GIANTS Game Engine v9 Documentation

Table of Contents

1. **Chapter 01:** Overview
    1. Enable development controls
    2. Runtime development key short-cuts
    3. Console Command Reference
2. **Chapter 02:** Editor
    1. Viewport
    2. Scenegraph
    3. Attributes
    4. Toolbar
    5. Terrain
    6. User Attributes
    7. Scripting
    8. Terrain Editing
    9. Replace Dialog
    10. Interactive Placement
    11. Other functionality
    12. Key short-cuts
3. **Chapter 03:** Debugger
    1. GIANTS Debugger
    2. Overview of panels
    3. Creating a new project
    4. Global and local variables
    5. Callstack panel
    6. Running new scripts during debugging
    7. Breakpoints
    8. Game output
    9. Searching in code
    10. Shortcuts
4. **Chapter 04:** Scripting API Reference
    1. Script Reference(1.2.0.1)
        1. General
        2. Animation
        3. AI
        4. I3D
        5. Handtools
        6. Events
        7. Objects
        8. Placeables
        9. Triggers
        10. Utils
        11. Vehicles
        12. Specializations
        13. animals
        14. AnimalHusbandryModules
        15. shop
        16. collections
        17. densityMaps
        18. economy
        19. effects

20. weather
21. farms
22. fieldJobs
23. dialogs
24. GUI
25. materials
26. misc
27. missions
28. modHub
29. Player
30. sounds
31. trainSystem
32. tutorials

Engine Reference(8.0.0.0)

33. General
34. Entity
35. Node
36. Scenegraph
37. Lighting
38. Camera
39. Shape
40. Particle System
41. Physics
42. Spline
43. Animation
44. Overlays
45. Sound
46. Input
47. XML
48. Network
49. Callbacks
50. Text Rendering
51. Terrain Detail
52. Tire Track
53. Editor
54. AI
55. Rendering
56. String
57. Math
58. I3D
59. Fillplanes
60. ModDownloadManager

2. Foundation Reference
    1. Scenegraph
    2. Input

5. **Chapter 05:** Tutorials
    1. Tutorial 1 - Load i3d files and basics
    2. Tutorial 2 - Light functions, global time and rendering text
    3. Tutorial 3 - User Attributes
    4. Tutorial 4 - Timers
    5. Tutorial 5 - Physics
    6. Tutorial 6 - Overlays
    7. Tutorial 7 - Audio
6. **Chapter 06:** Exporter

    1. Autodesk Maya[1] I3D Exporter
        1. Material Export Options
        2. Manual Installation
    2. Blender I3D Exporter
7. **Chapter 07:** Content Creation - Artwork Guide
    1. Autodesk Maya
    2. Texturing
8. **Chapter 08:** I3D Format
    1. Introduction
    2. Features
    3. Overview
    4. Specification
        1. General Layout
        2. Materials
        3. Shapes
        4. Dynamics
        5. Scenegraph
        6. Animation

Overview

Enable development controls

Open the file game.xml and change the value of controls from false to true.

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<game>
  ...
  <development>
    <controls>true</controls>
  </development>
</game>
```

Runtime development key short-cuts

Key	Function
~ or `	Toggle console
F2	Show frame rate
F3	Toggle frame rate limiter
F4	Wireframe mode
F5	Toggle debug rendering
F7	Toggle camera
F8	Toggle stats

Console command reference

help
Description

List all available commands

showFps
Description

Show frames per second

enableFramerateLimit

Description

Enable/disable frame per second limiter

framerateLimitFPS

Description

Frame per second limit attribute

listEntities

Description

Print detailed entity list

listResources

Description

Print detailed resource list

parallelRenderingAndPhysics

Description

Enable parallel rendering and physics

exit, quit or q

Description

Quits application

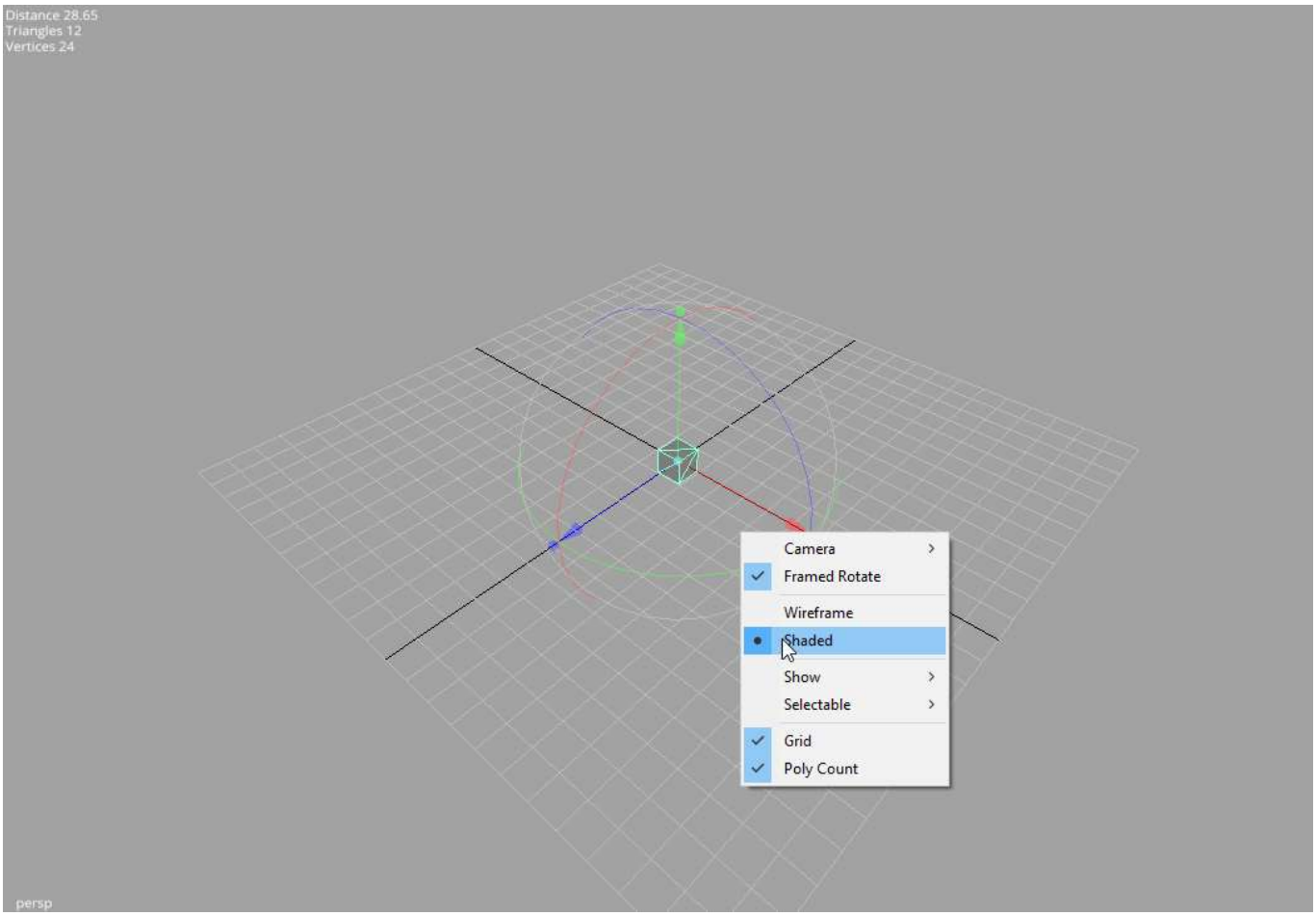
Editor

Table of Contents

1. Viewport
2. Scenegraph Panel
3. Attributes Panel
4. Toolbar
5. Terrain Editing
6. User Attributes
7. Animation Panel
8. Particle System Panel
9. Spline Editing
10. Scripting
11. Replace Dialog
12. Interactive Placement
13. Other functionality
14. Keyboard short-cuts

If you start up the Editor, you might not have all the panel windows open. To open other panel windows open the menu option window and enable the panel you want. You can adjust the width and height of the panels by dragging the outlines and if you want to close a window you can simply press the cross right of the panel name.

Viewport



Navigation

The navigation is quite like in Maya. If you have nothing open in your editor I suggest you quickly open a simple i3d file otherwise you won't see much of an effect. If its to dark in your scene just create a light by going to Create–>Light.

Some shortcuts:
LMB = Left Mouse Button
RMB = Right Mouse Button
MMB = Middle Mouse Button
Alt + LMB = Press and hold the Alt-Key and the left mouse button.

Alt + LMB rotates the camera.
Alt + RMB moves the camera forward and backward.
Alt + MMB causes the camera to pan.

If you don't have a middle mouse button, you can press LMB and RMB instead.

View Options

By simply right-clicking into the viewport panel you get the View Options. Sometimes it's useful too use different cameras in a scene. You can create them once and then with the View Options you can choose the camera to view at your scene from different angles quickly.

You might notice that if you rotate, the camera is rotated. If you have a big level this behaviour is very useful, but if you want to look at one particular object this can be quite awkward. To change the rotation to Framed Rotate you first have to select your object either by clicking on it directly or by choosing it from the scene-graph and the framing the selected object by pressing the F key. Now you can simply click (RMB) on the screen to open the View Options and choose Framed Rotate. Now the camera is rotating around your last framed object.

You can select the visual appearance of the object with Shaded (solid surface) or Wireframe (only edges of the object are visible)

If you have large scenes, you can toggle the visibility of lights, audio sources, physics and the cameras by checking them on the show submenu.

You can also use the Selectable submenu to toggle whether you can select lights, audio sources or cameras.

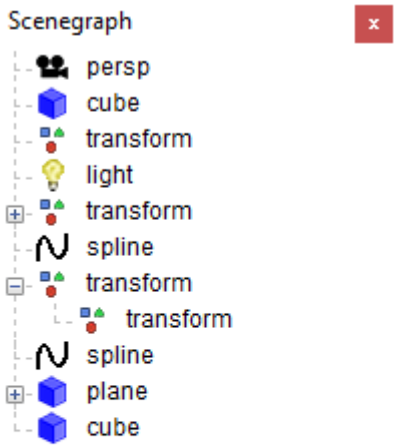
Furthermore you can toggle the grid and the polycount on and off.

Profile gives you the ability to choose different hardware profiles adequate for your system.

Debug can be used to find errors (e.g. on your 3D model)

Scenegraph Panel

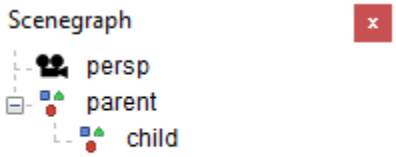
A very useful feature of the GIANTS Editor is the scenegraph. If you don't have it on the screen, just go to menu window and hit scenegraph. It is basically the same you have in Maya. It shows all objects you have in your scene and also the parent-child relationship between your objects.



Let's just make a little scene with some Transform Groups. The Transform Group is the basic building block of your scene. You can use a TransformGroup to move all the attached objects just by moving the Transform Group since the children inherit the transformations of the parent. To create a new Transform Group go to Create and hit TransformGroup. Now you can see it in the Scenegraph, its name is transform. To change the name just select it and go to the attributes panel. At the very top of the panel you can choose the name of your object. Make two TransformGroups and name them Parent and Child. If you move them around you see that their transformations are independent, the movement of the parent doesn't affect the child. Now let's make the appropriate relationship between the two. Select the child in the scenegraph panel and go to Edit->Cut. Now select your parent and hit Edit->Paste. Additionally by pressing MMB on the child, dragging it over parent and releasing MMB, the same as above will happen.

As you can see now, the child is now connected to the parent. If you now move around the parent, the

transformation of the child is affected too whereas the child can be moved around without affecting the parent.

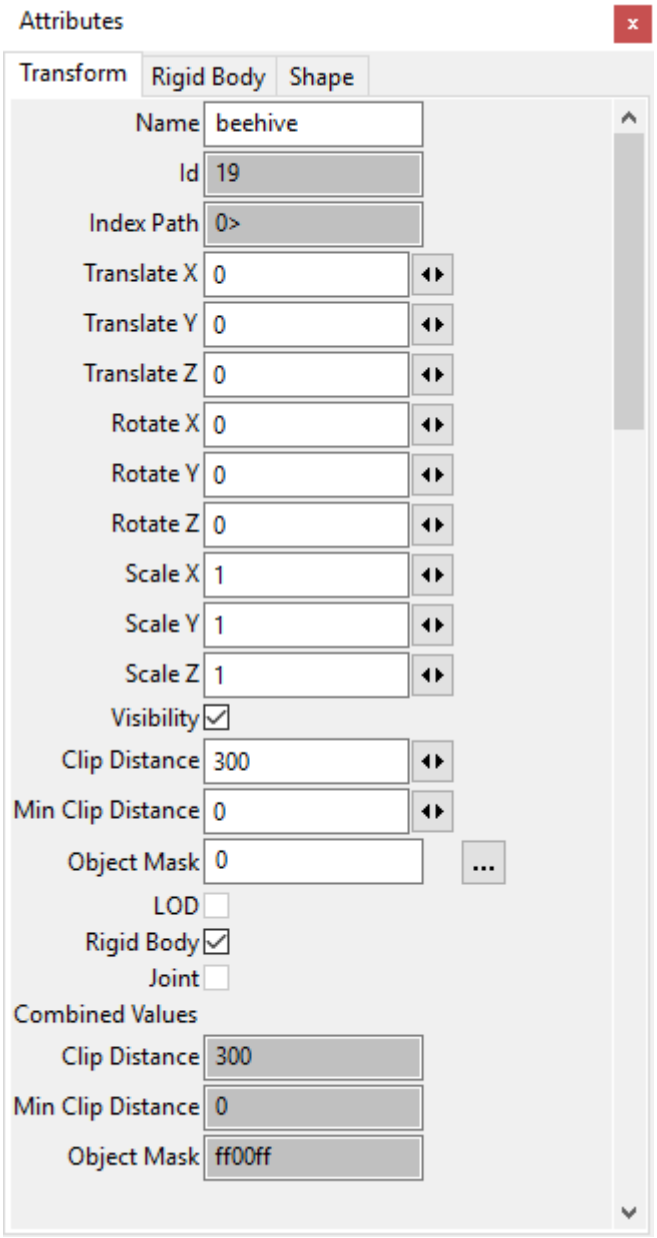


There can be 6 types of objects in your scene: Triangle Meshes, Splines, Cameras, Transform Groups, Lights, Terrains and Audio Sources.

It is also possible to use CTRL + C to copy objets, CTRL + X to cut objects and CTRL + V to paste objects. Notice that pasted objects are always children of the currently selected object. If you want to paste an object on the highest hierarchy level of the scenegraph be sure that nothing is selected in the scenegraph. This can be achieved by clicking at an empty space in the scenegraph panel.

Attributes Panel

If you don’t see the attributes panel go to Windows->Attributes then select an object in the scenegraph or the viewport. One important thing to notice is, that if the background color of an attribute turns red, you entered a value that isn’t allowed. (eg scale values other than 1 for a dynamic rigid body object) The background color of animated attributes is yellow.



Toolbar

The toolbar looks like this. If your pointing on the icons with your mouse, a text with the functionality of the tool is displayed.

Let’s quickly go through them from left to right.

File Operations

- Create a new i3d file
- Open an existing i3d file
- Open i3d file in text editor
- Reload i3d file
- Save the i3d file
- Save i3d file as

- Import an i3d file

History

- Undo last action
- Redo last action

Physics

- Play/Stop

If you hit the play icon, the physics will simulate.

Toggle Local- World Mode

A handy tool is the toggle Local- World Mode, it changes the orientation of the viewport transform gizmo from the local space of your object to world space.

Grid Snapping

Use this option to snap objects to a predefined grid.

Terrain and Terrain Foliage

- Terrain Sculpt Mode
- Terrain Detail Texture Paint Mode
- Terrain Info Layer Paint Mode
- Terrain Foliage Paint Mode

Texture Reload

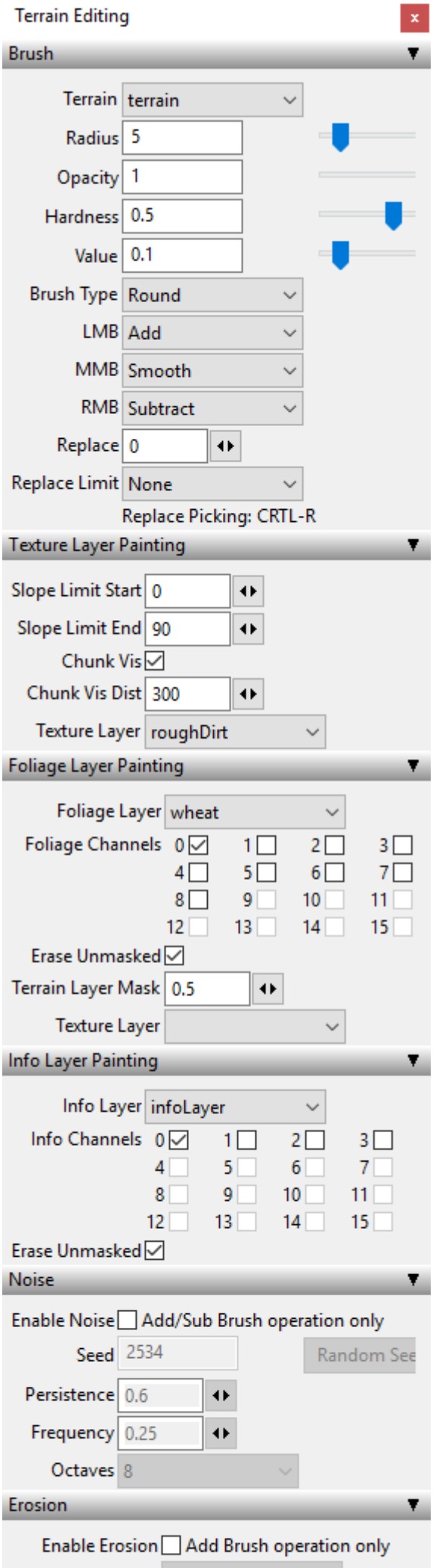
- Reload all textures
- Reload textures of selected objects

Script Editor

- Open Script Editor

Terrain Editing

To test the terrain sculpting, just open the terrain test scene. Now you have a terrain to play with. Open the terrain editing panel with window->Terrain Editing. (If the scenegraph panel is still open it might cut off a bit, just close it to get more space.)

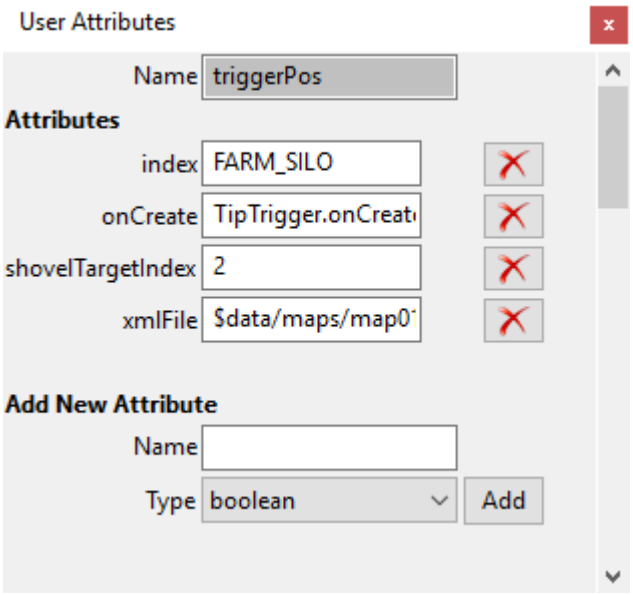

Terrain Editing
Brush
Terrain terrain
Radius 5
Opacity 1
Hardness 0.5
Value 0.1
Brush Type Round
LMB Add
MMB Smooth
RMB Subtract
Replace 0
Replace Limit None
Replace Picking: CRTL-R
Texture Layer Painting
Slope Limit Start 0
Slope Limit End 90
Chunk Vis
Chunk Vis Dist 300
Texture Layer roughDirt
Foliage Layer Painting
Foliage Layer wheat
Foliage Channels 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Erase Unmasked
Terrain Layer Mask 0.5
Texture Layer
Info Layer Painting
Info Layer infoLayer
Info Channels 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Erase Unmasked
Noise
Enable Noise Add/Sub Brush operation only
Seed 2534
Random See
Persistence 0.6
Frequency 0.25
Octaves 8
Erosion
Enable Erosion Add Brush operation only

If the terrain sculpt mode is active now, you can rise or lower the surface of your terrain by using RMB and LMB. MMB can be used to smooth and the value defines the force with which you pull or push. What happens here is, that you are painting a height map, that defines the height of each point of your terrain by giving it a grey-value. The attributes Opacity, Hardness and the Value are defining the "brush" you are painting the height map with. Just play around with the setting. You can change the radius of the brush with the shortcuts "V" and "B" and you can change the opacity value of the brush quickly with "N" and "M".

In the terrain texture paint mode next to the terrain sculpt icon you can colorize your terrain with different textures.
The used texture layers are defined in the i3d-file. (If you want to change the texture layers, you have to open the i3d with a text editor and change the layers there) Just choose your texture and paint onto your terrain. With the Chunk vis checkbox on, you can see exactly which texture layers you used in a chunk by what amout. You can use as many texture layers as you want, but you are limited to a maximum of four texture layers per chunk.

With the Terrain Foliage Paint Mode active, you can paint your foliage onto your terrain, it`s the same thing as if you would paint onto your terrain - not with a texture but with foliage. LMB adds foliage, RMB removes foliage.

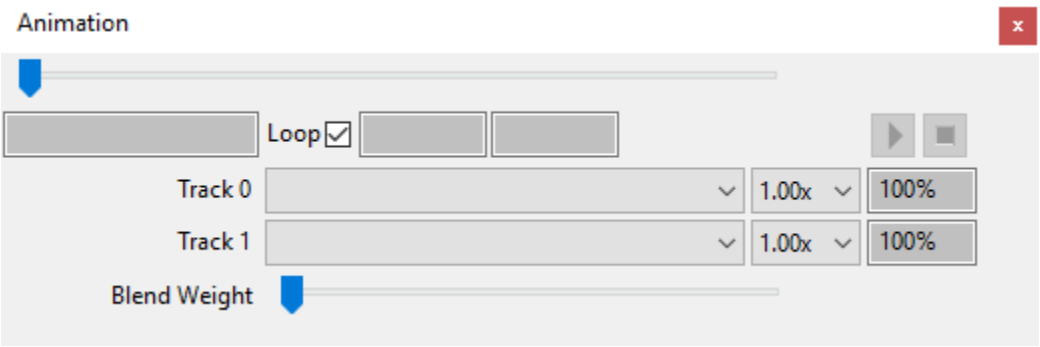
User Attributes



The user attributes panel is typically not visible if you open the editor for the first time. So go to Windows->User Attributes. Select an object in the scenegraph to see its user attributes.

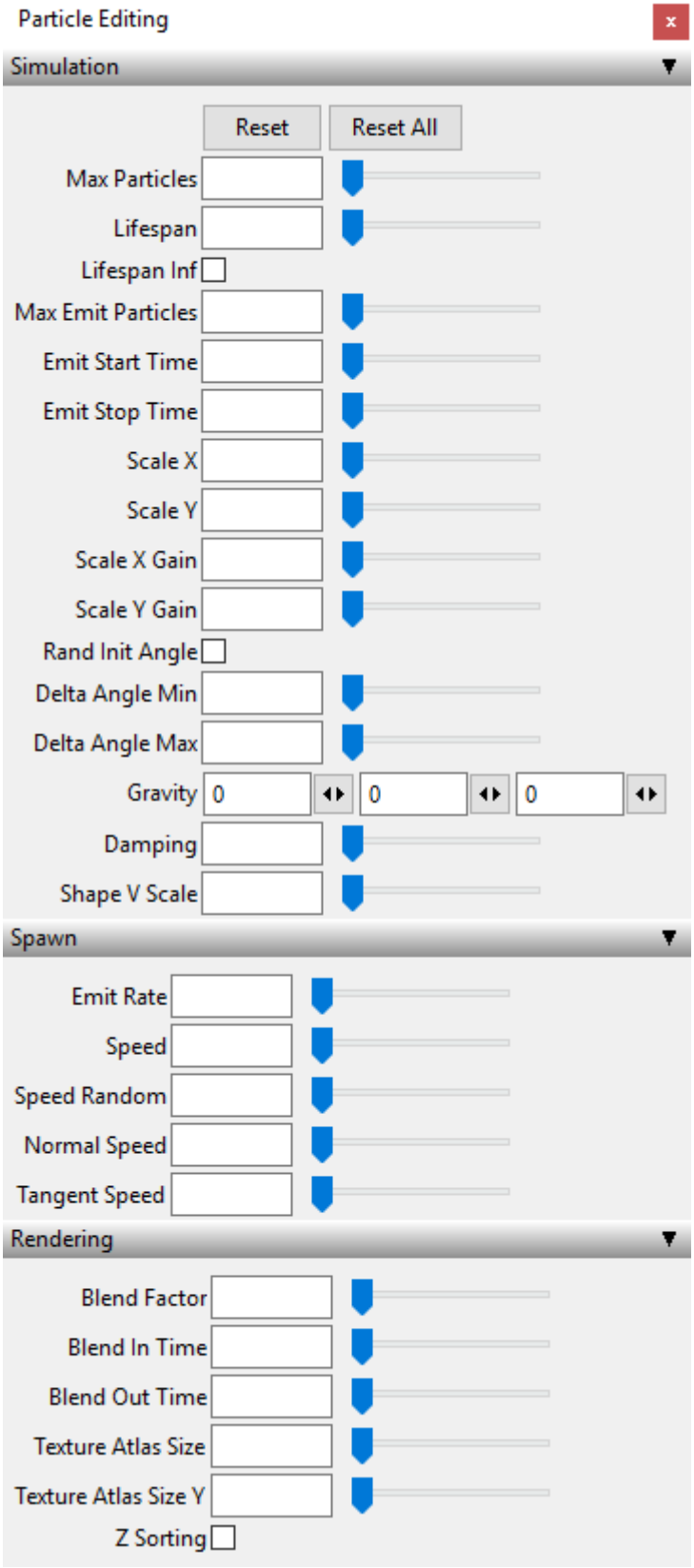
User Attributes can be defined in the editor and then be used in the engine (eg. within a script). This enables you to define object specific attributes for every object in the scene.

Animation Panel



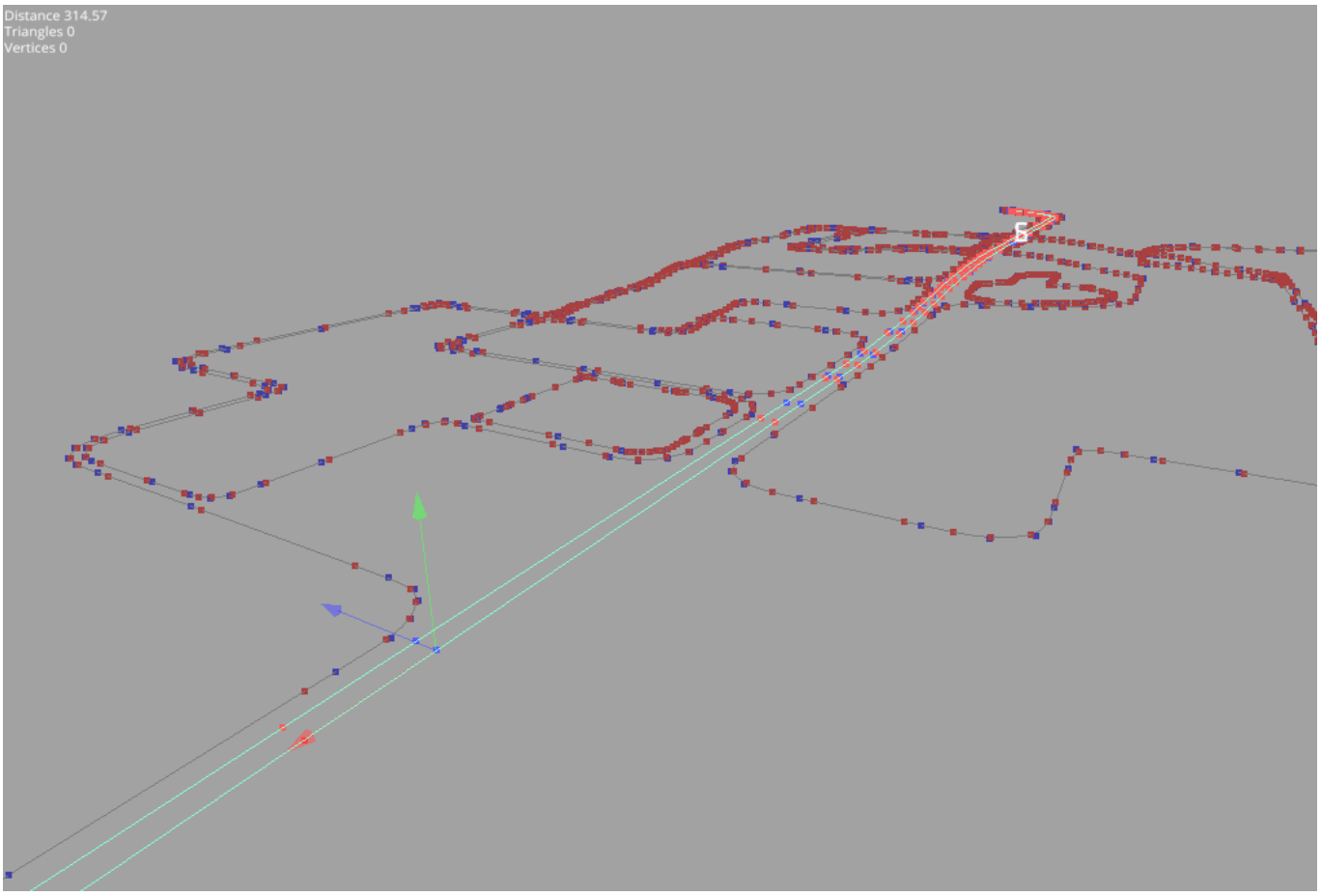
You can preview animation sets with the animation panel.

Particle System Panel



The particle system panel allows you to edit particle systems with real time preview.

Spline Editing



You can edit splines by picking a control vertex (CV) of a spline an moving it around. Delete or insert new control vertices with the keys delete and insert.

Scripting

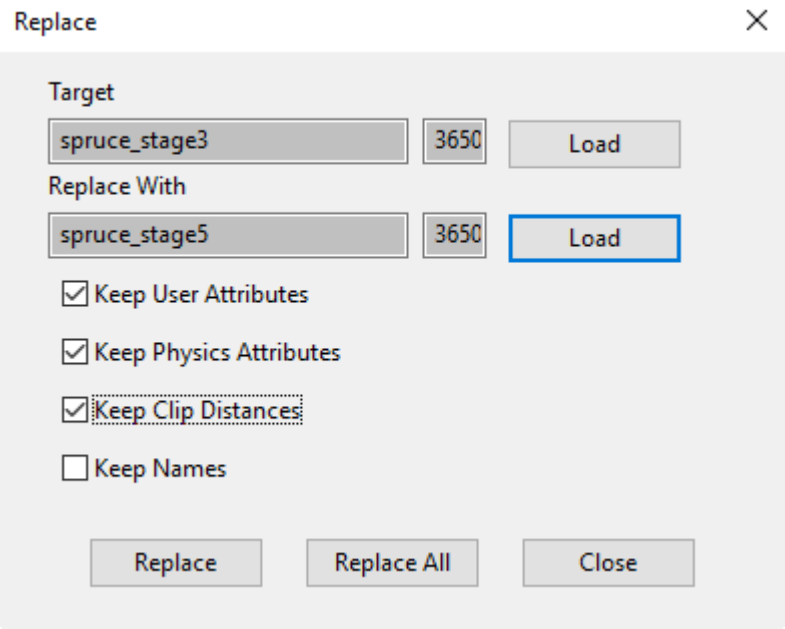
Here you can execute script snippets. With ENTER, you can add new lines. SHIFT+ENTER will execute the code in the text field.

Type in the following:
print("hello world from the GIANTS Editor");

Then hit SHIFT + ENTER and the string is printed out above.

The scripting can be useful for many things. For example you can run this script snippet to test the fog settings in an i3d scene:
setFog("exp2", 0.002, 0, 0.8, 0.81, 0.97);

Replace Dialog



If you want to replace an object in your scene, you can go to Edit->Replace to open the replace panel. Select the object you want to replace, then hit load. Do likewise with the object you want to replace with and choose whether you want to keep the User Attributes or not. Now you can replace one single object by hitting replace or you can replace all objects that are similar to the one you selected by hitting replace all.

Note: replace all does only work with single objects, but not with hierarchies of objects, however the single replace function works.
This feature is quite useful because you can use it to substitute an object reference feature within your i3d scene file.

Interactive Placement

This is a really nice feature of the editor, you should try it at least once!
To place an object on the surface of another object, you can simply select the object you want to place and then press CTRL + B + LMB, while pointing at your desired location. The selected object will then instantly be set to the location you've chosen.

Hint: you can do this with all your objects within your scenegraph and also with the camera or light sources. E.g. you can select the current camera and then place the camera at a location far away, allowing for fast relocation of the camera.

If you keep the LMB pressed and hit SHIFT or CRTL you can clone the selected object at the currect mouse position. SHIFT will just clone the object while CRTL will add a random rotation in the Y-axis. (Useful feature if you want to create a group with hundreds of randomly rotated objects like a forest)

Other functionality

Edit->Clear History: This clears the undo/redo history.

Edit->Move to Camera: Moves the selected object in front of the current camera

Navigation Speed: Moves the camera faster or slower. Use - and + to ajust the speed.

Default keyboard short-cuts

Key	Function
W A S D	Navigation
Alt + LMB	Rotate camera
Alt + MMB	Pan camera
Alt + RMB	Zoom camera
Alt + LMB + RMB	Move camera up or down and left or right camera
F	Frame selected object
-	Decrease navigation speed
+	Increase navigation speed
4	Wireframe mode
6	Shaded mode
Ctrl-S	Save
Ctrl-Z	Undo
Ctrl-W	Replace Dialog
Ctrl-X	Cut
Ctrl-C	Copy
Ctrl-V	Paste
Ctrl-Shift-C	Copy X,Y,Z components at once
Ctrl-Shift-V	Paste X,Y,Z components at once (can also be copied from a text source in the format "x y z")
Delete	Delete
Ctrl-D	Duplicate
Ctrl-F	Move to Camera
Ctrl-B	Interactive placement (hold left mouse button to move around)
Shift	Interactive placement paint
Ctrl	Interactive placement paint with random rotation around y axis
Ctrl-H	Hide object

Shift-H	Show object
Ctrl-G	Group objects
Ctrl-R	Pick replace value in viewport (Terrain edit modes)
V	Decrease brush radius
B	Increase brush radius
N	Decrease brush opacity
M	Increase brush opacity
F8	Toggle stats
Shift + Enter	Execute Script (Script Window)
X	Absolute grid snapping
J	Relative grid snapping
Delete	Delete spline control vertex
Insert	Insert new spline control vertex
Left	Previous spline control vertex
Right	Next spline control vertex
Up or Down	First spline control vertex
S	Stitch spline endpoints
O	Toggle spline open/close
R	Reverse spline
Ctrl-L	Create light

Debugger

Table of Contents

1. GIANTS Debugger/IDE
2. Overview of panels
3. Creating a new project
4. Global and local variables
5. Callstack panel
6. Running new scripts during debugging
7. Breakpoints
8. Game output
9. Searching in code
10. Shortcuts

GIANTS Debugger/IDE

The GIANTS Debugger is a new tool for creating, editing and debugging script mods. It works as an editor and a “remote” debugger in one. It interacts with the game state and provides you with information about the game state.

Possible use-case scenarios:

- to create or modify LUA and XML files
- using breakpoints to see how script is executed
- investigating a paused game state by looking through callstack or inspecting variables
- seeing when and how variables change
- live-coding by writing script while the game is running

The Debugger can be used on all kinds of mod configurations. These are the most common three configurations:

Working on a single mod (fast iteration time, but only compatible with singeplayer):

Unzip your mod and put the mod in the “mods” directory (by default in Documents/My Games/FarmingSimulator2019/mods). Make sure that the layout of the mod is correct, i.e. the modDesc.xml is directly in the mod folder (i.e. mods//modDesc.xml). Then create a new project in the Debugger. In “Project Settings”, set “Mod Name” to be the same name as the name of the mod folder and set the “Mod Directory” as a path to the directory of the mod (mods/). In case the mod zip file is already in the “mods” directory remove the zip file for the “mods”, because the zip file has priority over the directory when the game is loading.

Working on multiple mods at the same time (fast iteration time, but only compatible with singeplayer):

Follow the instructions above, but when you create a project in the Debugger, set the “Mod Directory” to “mods” directory instead of the specific mod directory. This way you can edit several mods at the same time. In this configuration, the “Mod Name” is not important, you can call it anything you like.

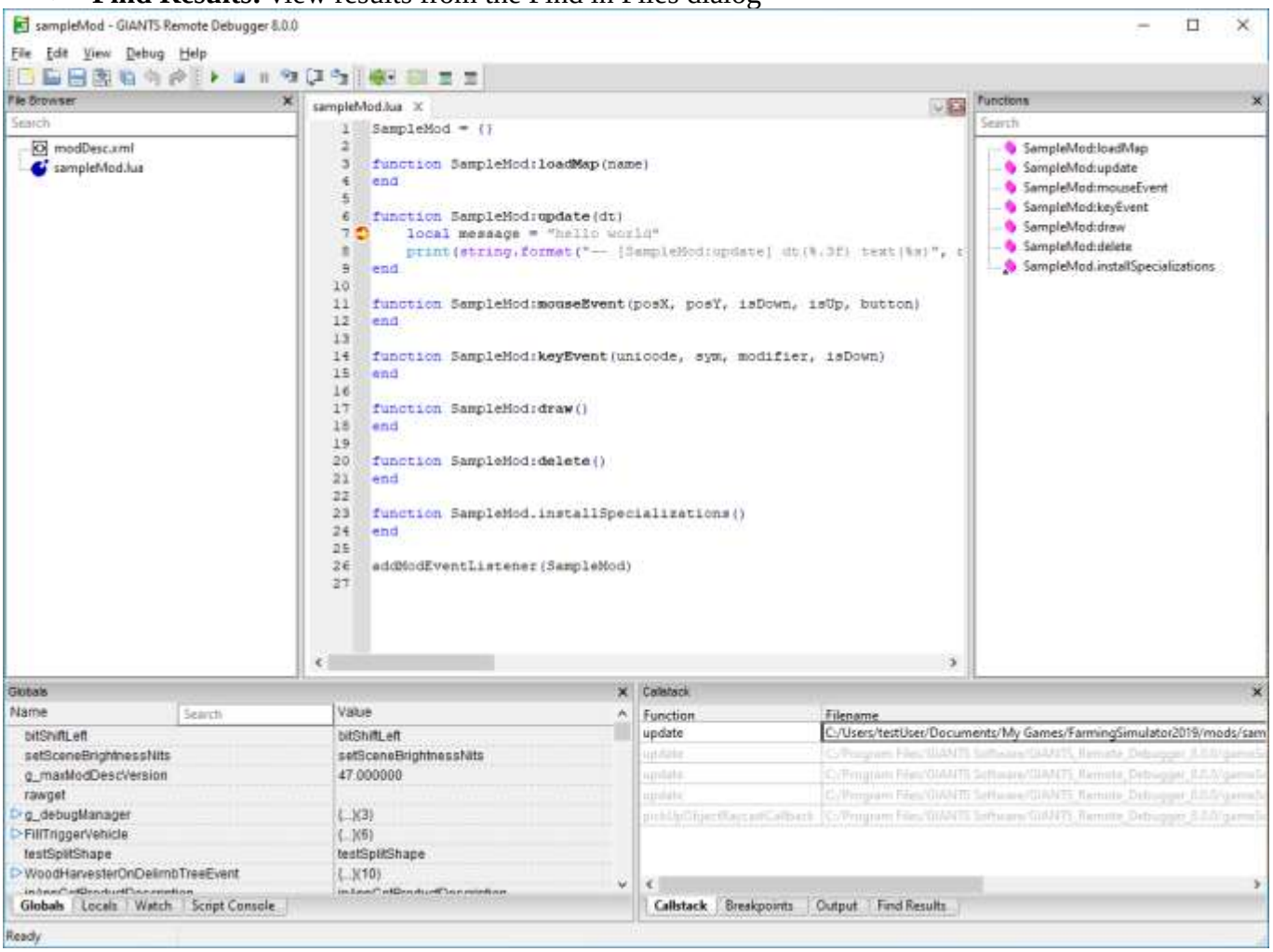
Working on a single mod outside the “mods” directory (also compatible with multiplayer):

This is the most complex configuration, and it is the only one that lets you debug mods in multiplayer. Place the mod zip file in the “mods” directory and unzip it somewhere else. In “Project Settings” set the “Mod Name” to match with the name of the zip file but without .zip at the end) and set the “Mod Directory” as a path to the unzipped directory. For every change you make in the Debugger, you will have to re-create the mod zip file and copy it to the “mods” directory before you start the game. External tools for zipping are useful for this purpose (not included).

Overview of panels

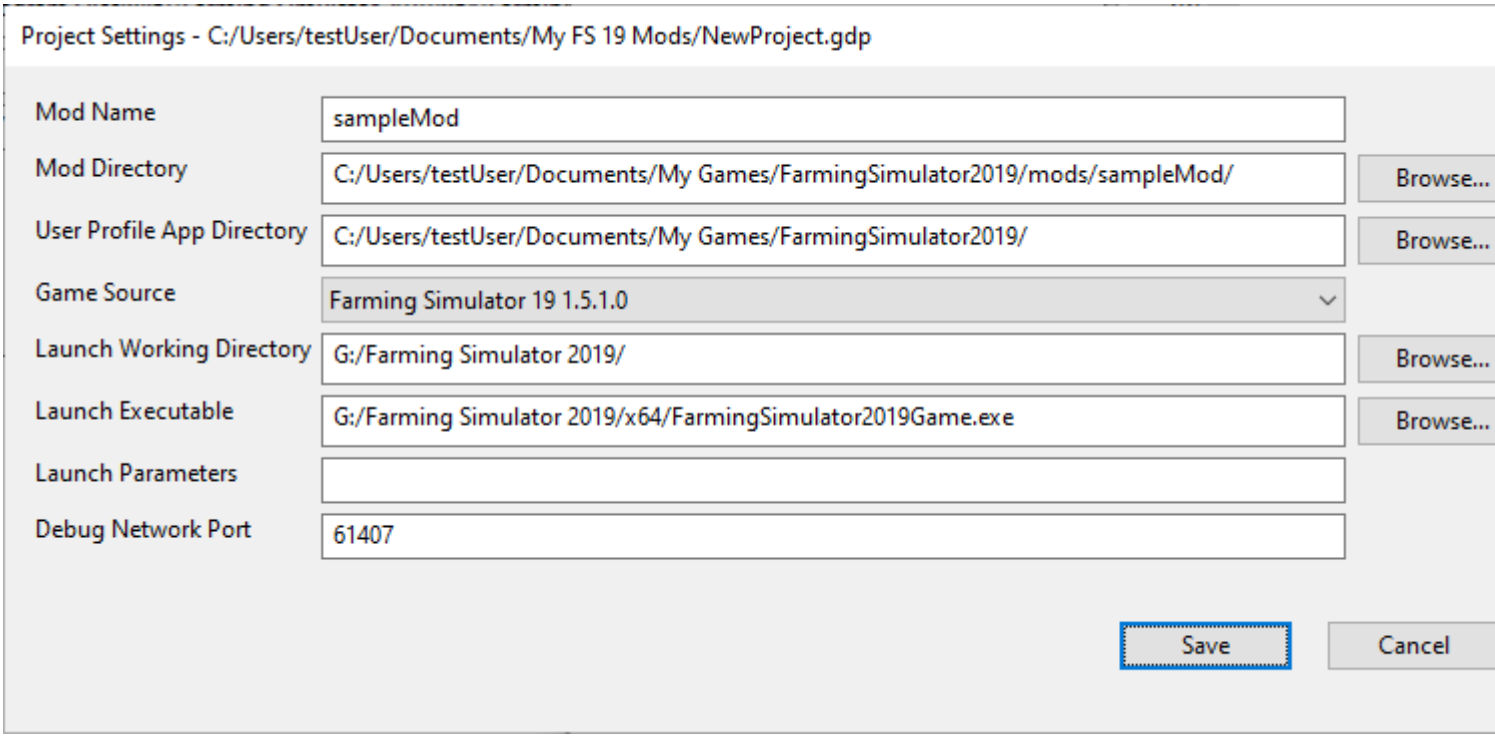
- **File Browser:** view files and folders in your project directory
- **Functions:** list of functions defined in the current file

- **Globals:** view global variables
- **Locals:** view local variables
- **Watch:** view the values of user-defined variables
- **Script console:** run user-defined parts of code
- **Callstack:** view current callstack
- **Breakpoints:** manage breakpoints
- **Output:** view log output
- **Find Results:** view results from the Find in Files dialog



Creating a new project

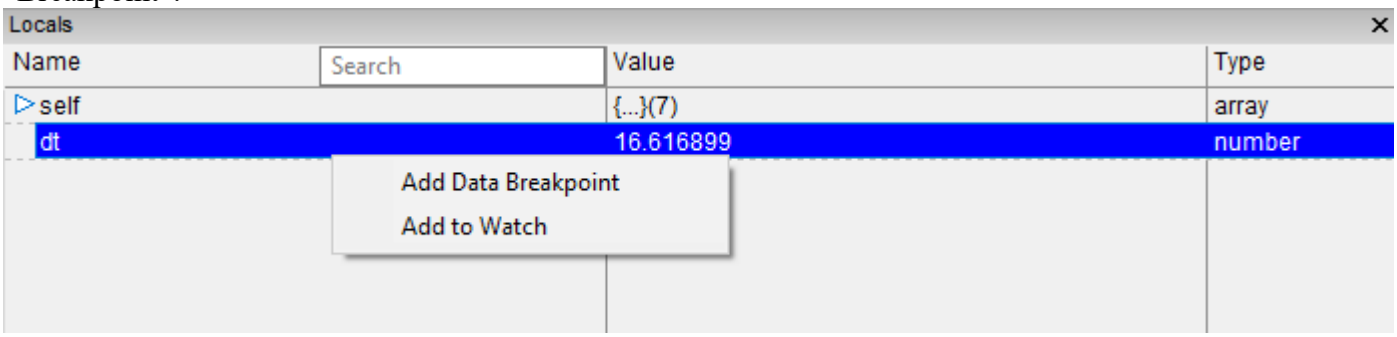
First you need to open or create a project.



Most of the required parameters are self-explanatory. The “Mod Directory” can be a path to a single mod or a “mods” directory if you are working on several mods at the same time. If you are working on a single mod, the “Mod Name” must have the same name as the .zip file. The “User Profile App Directory” should be saved as a path to a folder where the game stores user data (changes are not needed by default if the game configuration has not been changed manually). The debugger uses sockets to communicate with the game and listens at the port defined as the “Debug Network Port”.

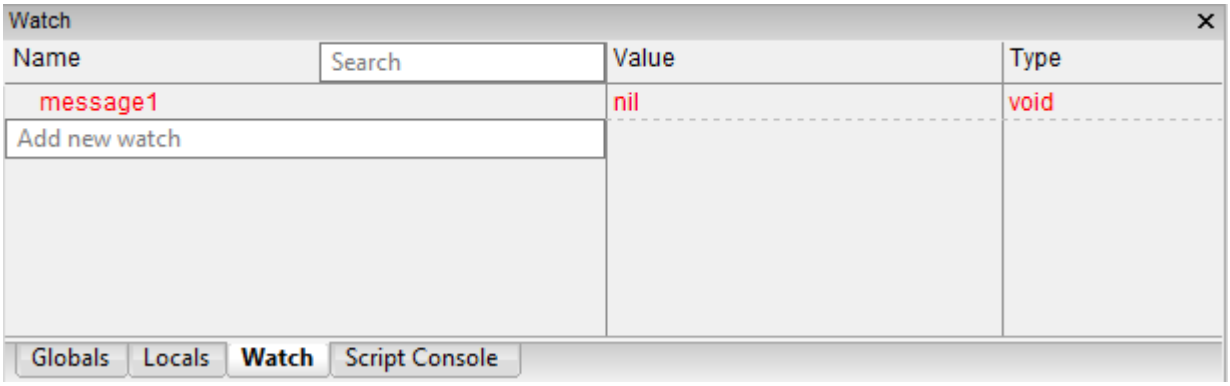
Global and local variables

The function of the “Globals” and “Locals” panels is the same: to explore the value of local and global variables at a current execution point (using breakpoints to halt execution and observe these variables, or by hitting the ‘break all’ button). By right-clicking on a variable, you can add it as a “Watch” or a “Breakpoint”.



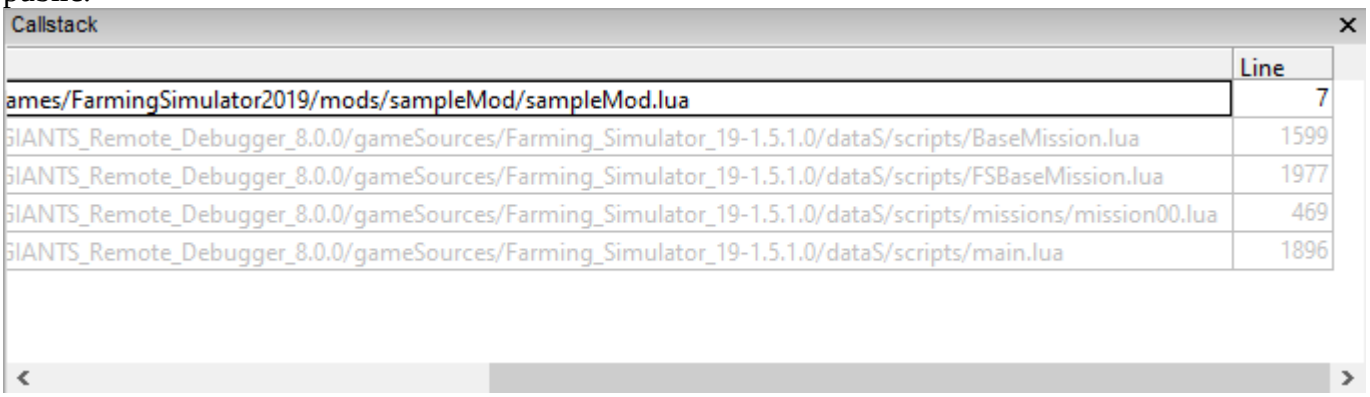
The ”Watch” panel is similar to the “Globals” and “Locals” panel, but instead of global or local variables, you can select variables that you would like to ‘watch’. Values are updated every time a breakpoint is hit or when you execute the code step-by-step. Variable names can be written like most other simple Lua statements:

- variableName
- object.member
- table[1]
- table[index]
- table[object.index]



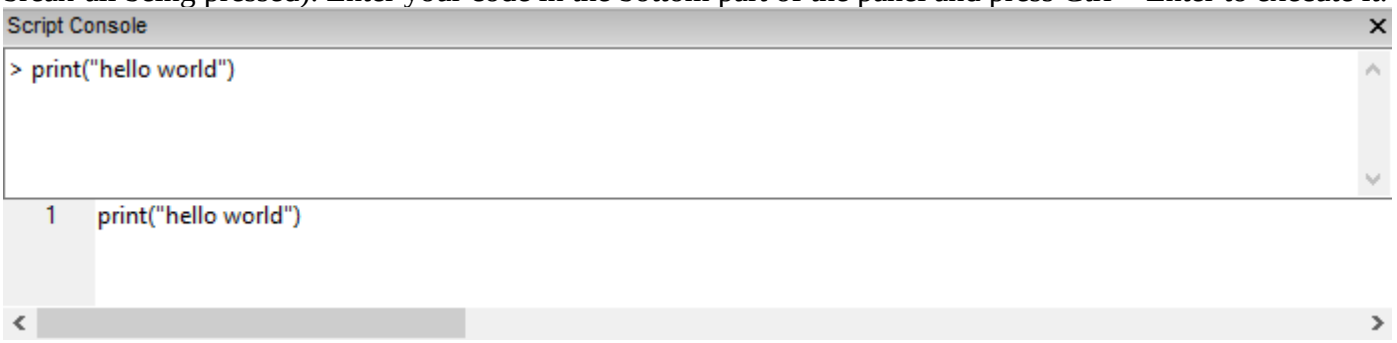
Callstack panel

The “Callstack” panel shows you the callstack at a current execution point. Double-click on a line to open a file at the location of the call. The Debugger even allows you step through some of the base game functions for which the source code has been made public. Grayed-out lines are parts of code that have not been made public.



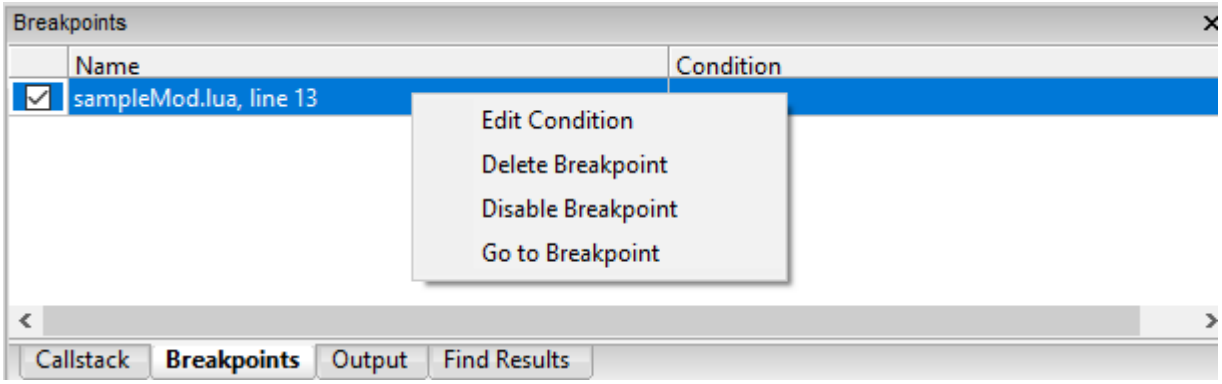
Running new scripts during debugging

The “Script Console” executes user-defined Lua code when execution is paused (by hitting a breakpoint or break-all being pressed). Enter your code in the bottom part of the panel and press Ctrl + Enter to execute it.

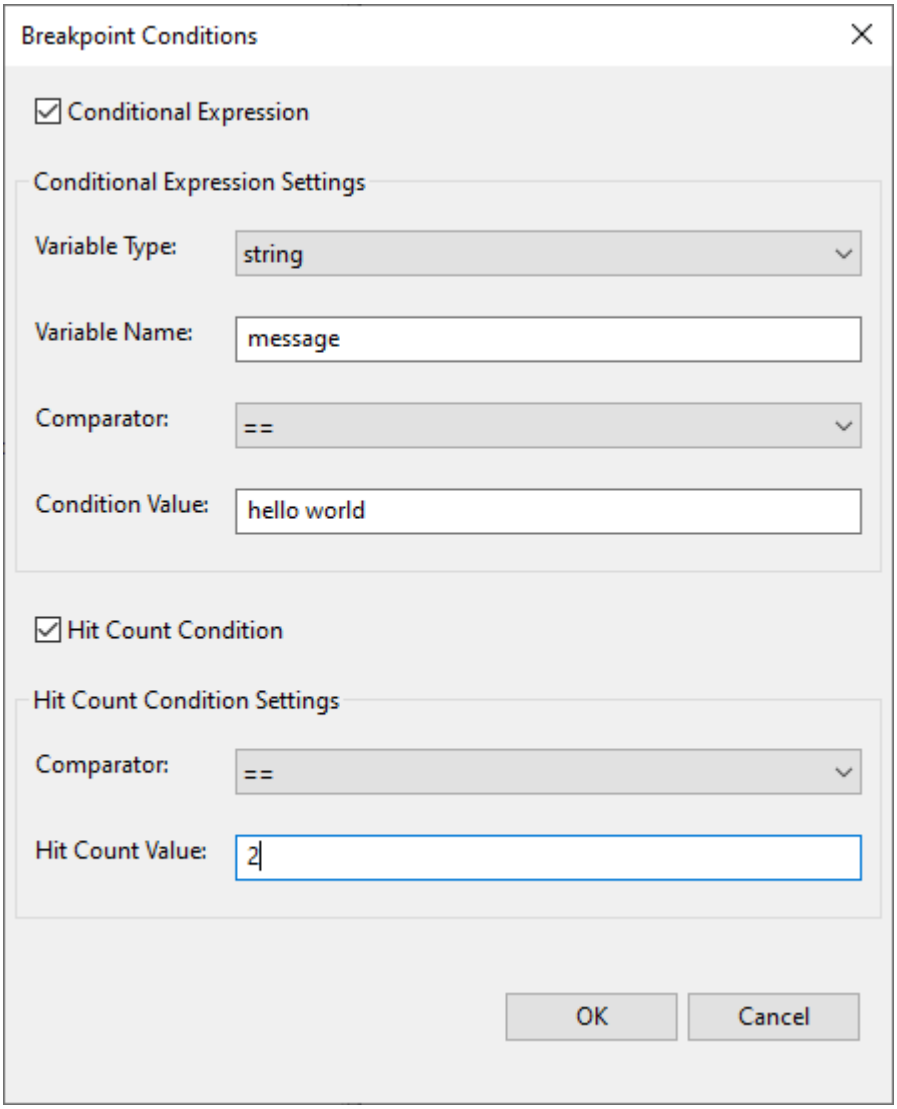


Breakpoints

The “Breakpoints” panel is where you manage breakpoints.
Regular breakpoints can be placed in the code by clicking on the vertical gray bar on the left of the code panel. You can also use F9 to enable/disable a breakpoint at the current position. The program will stop running when it tries to execute a line with a breakpoint.
Data breakpoints can be placed by right-clicking on a variable name in the “Globals”/”Locals”/”Watch” panel or by right-clicking on a name in the text editor. A data breakpoint will be hit when the data for the specified variable changes.

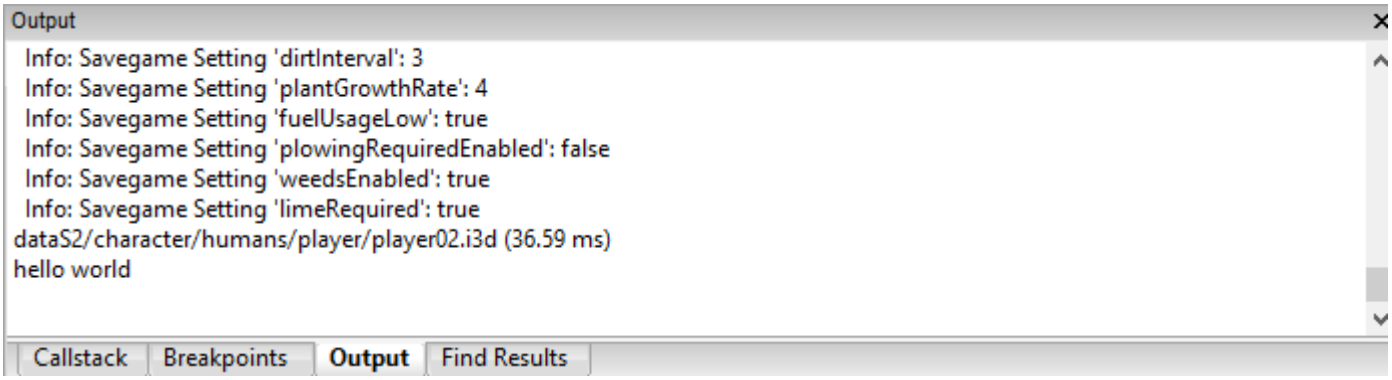


You can fine-tune a breakpoint by adding a conditional expression and/or hit count condition to it. These conditions are evaluated each time a breakpoint is hit and the code will only stop executing once the conditions are met.



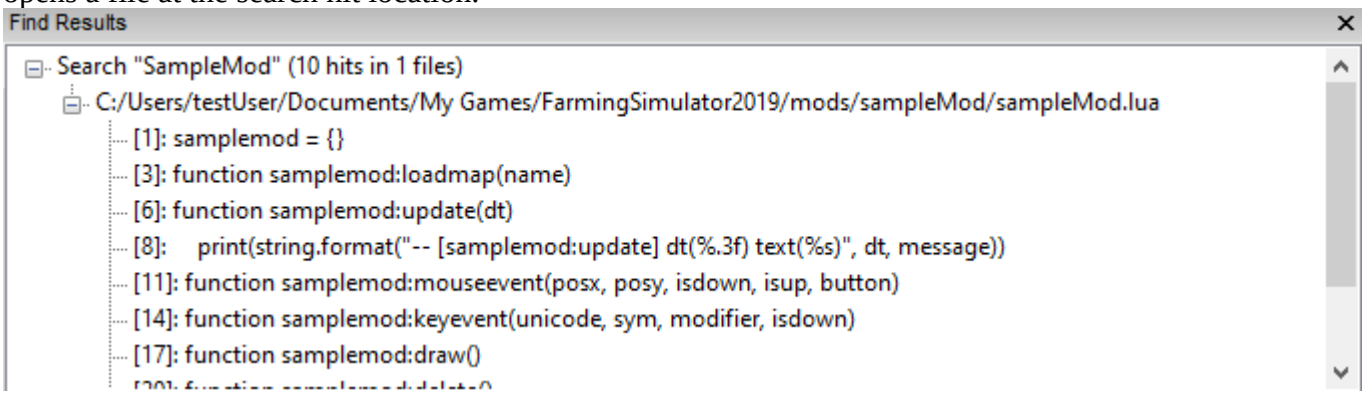
Game output

The “Output” panel shows the game’s log output. If a script error is reported, this panel will provide you with the file name and line number, so that you can double-click on it and view the code causing the error. The output is the same as the log file written by the game.



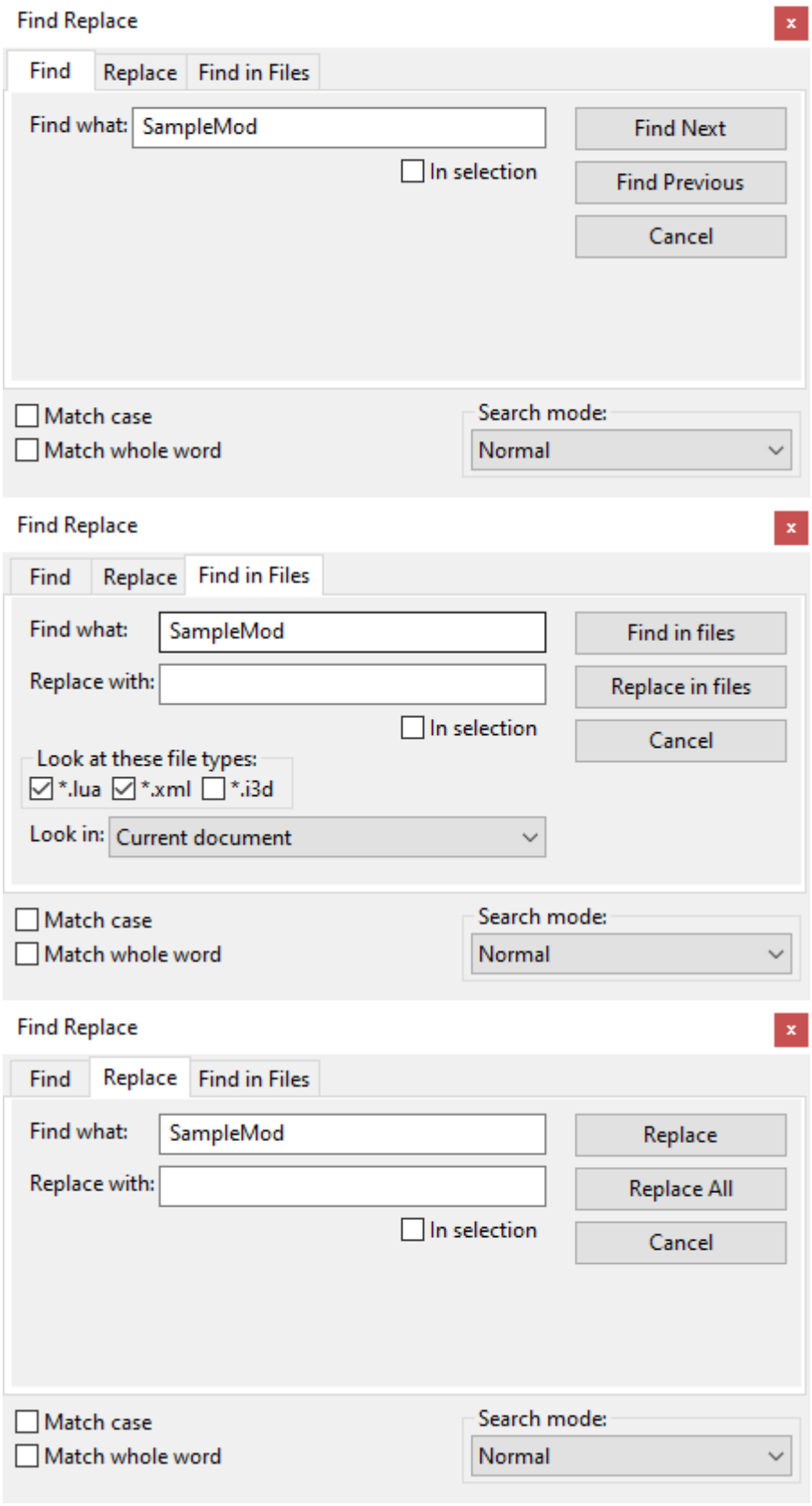
Searching in code

The “Find Results” panel shows search results from the “Find in Files” dialog. Double-clicking on a result opens a file at the search hit location.



The “Find Replace” window behaves in a similar way to other text editors. “Find” and “Replace” directly show the results in the text editor; while “Find in Files” searches through the entire file and shows results in the “Find Results” panel. Other search modes are available too:

- “Extended” search allows you to use special “escape” characters (\t = tab, \n = new line, \r = carriage return, \0 NULL character and \\ = backslash)
- “Regular Expression” searches with regular expressions. Captures are defined with brackets ‘(‘ and ‘)' in the search string and can be placed with $1-$9 in the replace string.



Default keyboard short-cuts

Key	Function
Ctrl-N	New file

Ctrl-O	Open file
Ctrl-S	Save file
Ctrl-Shift-S	Save all files
Ctrl-Q	Quit application
Ctrl-Z	Undo
Ctrl-Y	Redo
Ctrl-F	Find
Ctrl-H	Replace
Ctrl-Shift-F	Find in files
F3	Find next
Shift-F3	Find previous
Ctrl-Shift-G	Go to file
Ctrl-G	Go to line
Ctrl-K	Comment lines
Ctrl-Shift-K	Uncomment lines
Ctrl-W	Close tab
Ctrl-Shift-T	Reopen tab
Alt-Up	Move line up
Alt-Down	Move line down
Alt-Left	Navigate backwards
Alt-Right	Navigate forward
F5	Run / Continue execution
Ctrl-F5	Run Without Debugging
F7	Pause execution
F6	Stop execution
F9	Toggle Breakpoint
F11	Step Into
F10	Step Over
Shift-F11	Step Out

Scripting API Reference
Script Version: 1.2.0.1

AchievementMessage
Description

Achievement message display element.
-- Used to display a message in the HUD when a new achievement is unlocked.
--@category GUI

new
Description

Create a new instance of AchievementMessage.

Definition

new(string hudAtlasPath, table inputManager, table guiSoundPlayer, ContextActionDisplay ContextActionDisplay)

Arguments

string	hudAtlasPath	Path the HUD texture atlas
table	inputManager	InputBinding reference for achievement acknowledgment input
table	guiSoundPlayer	GuiSoundPlayer reference for playing a sound cue
ContextActionDisplay	ContextActionDisplay	reference for visibility checks (it occupies the same space on the HUD)

Return Values

table AchievementMessage instance

onMenuVisibilityChange
Description

Handle changes in menu visibility.
Keeps track of menu visibility state to avoid showing (or updating) an achievement message when the menu is active.

Definition

onMenuVisibilityChange()

showMessage
Description

Show an achievement message.
The message will be shown as soon as the player is not in a menu and there is no context action display visible.
Messages will however not be checked for overlaps. The caller has to make sure that achievements are displayed
sequentially.

Definition

showMessage(string title, string description, string iconFilename, table iconUVs, float duration)

Arguments

string title	Achievement title text
string description	Achievement description text
string iconFilename	Path to the achievement icon texture

table	iconUVs	UV coordinates of achievement icon with icon texture
float	duration	Maximum display duration in milliseconds

update
Description

Update display state.

Definition

update()

beginShowMessage
Description

Actually start showing the message when nothing is obstructing the view.

Definition

beginShowMessage()

hideMessage
Description

Hide the message, including animation.
Also removes the acknowledgment input events.

Definition

hideMessage()

draw
Description

Draw the achievement message if necessary.

Definition

draw()

storeScaledValues
Description

Store scaled positioning, size and offset values.

Definition

storeScaledValues()

setScale
Description

Set this element's scale.

Definition

setScale()

getBackgroundPosition
Description

Get the position of the background element, which provides this element's absolute position.

Definition

getBackgroundPosition(scale Current, float width)

Arguments

scale	Current	UI scale
float	width	Scaled background width in pixels

Return Values

float X position in screen space

float Y position in screen space

createBackground

Description

Create the empty background overlay

Definition

createBackground()

createComponents

Description

Create required display components.

Definition

createComponents()

createFrame

Description

Create the message frame.

Definition

createFrame()

createIcon

Description

Create the achievement icon.

Definition

createIcon()

AnimalScreen

Description

Animal Buying Screen.
AnimalScreen = {}

new

Description

Constructor

Definition

new(table target, table metatable)

Arguments

table target

table metatable

Return Values

table self instance

Code

```
53  function AnimalScreen:new(target, custom_mt, animalController, l10n, mess
54  local self = ScreenElement:new(target, custom_mt or AnimalScreen_mt)
55
56  self:registerControls(AnimalScreen.CONTROLS)
```

```
57
58  self.l10n = l10n
59  self.messageCenter = messageCenter
60  self.animalController = animalController
61
62  self.animalController:setSourceUpdateCallback(self.onSourceUpdate, self)
63  self.animalController:setTargetUpdateCallback(self.onTargetUpdate, self)
64  self.animalController:setNoValidHusbandryCallback(self.onNoValidHusbandry
65  self.animalController:setHusbandryIsFullCallback(self.onHusbandryIsFull,
66  self.animalController:setTrailerFullCallback(self.onTrailerIsFull, self)
67  self.animalController:setInvalidAnimalTypeCallback(self.onInvalidAnimalTy
68  self.animalController:setAnimalNotSupportedByTrailerCallback(self.onAnima
    self)
69  self.animalController:setNotEnoughMoneyCallback(self.onNotEnoughMoney, se
70  self.animalController:setCanNotAddToTrailerCallback(self.onCanNotAddToTra
71  self.animalController:setAnimalInUseCallback(self.onAnimalInUse, self)
72
73  self.isSourceSelected = true
74
75  self.isOpen = false
76  self.lastBalance = 0
77  self.sourceElementToAnimal = {}
78  self.targetElementToAnimal = {}
79
80  self.dofState = DepthOfFieldState.createFullscreenBlur()
81
82  return self
83  end
```

onOpen

Description

Callback on open

Definition

onOpen()

Code

```
87  function AnimalScreen:onOpen()
88  AnimalScreen:superClass().onOpen(self)
89
90  self.isOpen = true
91  self.isUpdating = false
92
93  g_gameStateManager:setGameState(GameState.MENU_ANIMAL_SHOP)
```

```
94
95  self.dofState:recordState() -- record current state before changing
96  self.dofState:apply() -- apply blur effect
97
98  self:updateScreen()
99
100 self.messageCenter:subscribe(MessageType.HUSBANDRY_ANIMALS_CHANGED,
    self.onAnimalsChanged, self)
101 end
```

onClose

Description

Callback on close

Definition

onClose(table element)

Arguments

table element

Code

```
106 function AnimalScreen:onClose(element)
107 AnimalScreen:superClass().onClose(self)
108 self.animalController:close()
109 self.isOpen = false
110 self.sourceElementToAnimal = {}
111
112 g_currentMission:resetGameState()
113
114 self.messageCenter:unsubscribeAll(self)
115
116 self.dofState:reset() -- reset depth of field parameters (blur
    effect)
117 end
```

onClickBack

Description

Callback on click back

Definition

onClickBack()

Code

```
262 function AnimalScreen:onClickBack()
263 AnimalScreen:superClass().onClickBack(self)
264 self:changeScreen(nil)
265 end
```

onClickOk

Description

Callback on click cancel

Definition

onClickOk()

Code

```
269  function AnimalScreen:onClickOk()
270  AnimalScreen:superClass().onClickOk(self)
271
272  if self.isSourceSelected then
273  self.animalController:moveToTarget(self.listSource.selectedIndex)
274  else
275  self.animalController:moveToSource(self.listTarget.selectedIndex)
276  end
277  end
```

BaseMission

Description

Input context name for identification, does not take care of proper context handling

new

Description

Create a new base mission

Definition

new(string baseDirectory, table customMt, table missionCollaborators)

Arguments

string	baseDirectory	Mission scripts base directory
table	customMt	Sub-class meta table
table	missionCollaborators	MissionCollaborators object containing a defined collection required object references for a mission.

initialize

Description

Initialize mission after instantiation.
Create complex members and call dependency methods in here so that mission instantiation cannot fail.

Definition

initialize()

createHUD

Description

Create the in-game HUD display.

Definition

createHUD()

addHelpButtonText

Description

Deprecated: Replaced by InputBinding:setActionEventText()

Definition

addHelpButtonText()

subscribeSettingsChangeMessages

Description

Subscribe to relevant game settings changes.

Definition

subscribeSettingsChangeMessages()

subscribeGuiOpenCloseMessages

Description

Subscribe to GUI notifications of opening and closing.

Definition

subscribeGuiOpenCloseMessages()

onBeforeMenuOpen

Description

Handle a menu open event.

Definition

onBeforeMenuOpen()

onAfterMenuClose

Description

Handle a menu closed event.

Definition

onAfterMenuClose()

registerActionEvents

Description

Register required input action events.

Definition

registerActionEvents()

registerPauseActionEvents

Description

Register action events for pause actions.
Event registration in this method is enclosed in an input binding registration context (InputBinding:beginActionEventsModification()).
Make sure that this is only called when all other registration-context altering code is done.

Definition

registerPauseActionEvents()

onPause

Description

Input event for "pause".

Definition

onPause()

onConsoleAcceptPause

Description

Input event for breaking pause on consoles.

Definition

onConsoleAcceptPause()

onToggleHelpText

Description

Input event for "toggle help text".

Definition

onToggleHelpText()

onSwitchVehicle

Description

Input event for "switch vehicle".

Definition

onSwitchVehicle(directionValue Numeric)

Arguments

directionValue Numeric value for switch direction, 1 is the next, -1 is the previous vehicle

Binding

Description

Binding of input to actions.
--@category Input
--@xmlConfig binding#device Device ID, must match a value returned by engine function getGamepadId() or one of the constants of InputDevice.DEFAULT_DEVICE_NAMES.

new

Description

Create a new Binding instance.

Definition

new(action InputAction, deviceId ID, axisNames List, axisComponent Component, inputComponent Component, index Binding)

Arguments

action	InputAction	reference which is being bound to an input
deviceId	ID	of the input device whose input is being bound to the given action
axisNames	List	of input axis names. If the list contains more than one axis, any axis before the last one is a modifier.
axisComponent	Component	of the last given axis to bind to the given action.
inputComponent	Component	of a physical axis (e.g. analog stick) to bind
index	Binding	index, 1 primary, 2 secondary, etc.

Return Values

bool true if allowed

New Binding instance

createFromXML

Description

Create a new Binding instance from an XML element.

Definition

createFromXML(int xmlFile, string elementTag, bool isLocked)

Arguments

int	xmlFile	XML file handle
string	elementTag	Tag of the element to parse as a Binding
bool	isLocked	If true, the binding belongs to a locked action

Return Values

boolean	isAllowed true if fillType is supported else false
Binding	instance initialized with values from XML and the given parameters

saveToXMLFile

Description

Save this binding to XML.

Definition

saveToXMLFile(xmlFile Input, elementTag Element)

Arguments

xmlFile	Input	binding settings XML file handle
elementTag	Element	tag of this binding

Return Values

boolean	isSupported true if fillType is supported else false

updateData

Description

Update a binding with a new input target.
Always use this method to change existing bindings or risk invalid state.

Definition

updateData(deviceId (New), axisNames List, inputComponent [optional])

Arguments

deviceId	(New)	device ID
axisNames	List	of input axis names. Make sure these are valid in the context of this binding's state (i.e. device and action axis type).
inputComponent	[optional]	Direction of a physical axis (e.g. analog stick) to bind, defaults to positive.

Return Values

float	freeCapacity free capacity

updateInput

Description

Update this binding's input state.

Definition

updateInput(inputValue Current)

Arguments

inputValue	Current	input value of the last axis in self.axisNames

Return Values

boolean	isAllowed true if toolType is allowed else false

setIsAnalog

Description

Set this binding's analog input flag.

Definition

setIsAnalog()

Return Values

table instance instance of object

setIndex

Description

Set this binding's index and update internal state.

Definition

setIndex()

Return Values

table instance instance of object

setActive

Description

Set this binding's active state.
Only active bindings are updated and can trigger action events.

Definition

setActive()

Return Values

integer	harvestPixelsSum	harvest of pixels sum
integer	harvestNumPixels	harvest number of pixels
float	sprayFactor	spray factor
float	plowFactor	plow factor
float	limeFactor	lime factor
float	weedFactor	weed factor
integer	growthState	growth state
float	maxArea	max area

setFrameTriggered

Description

Set the frame trigger flag which shows if this binding has triggered an event this frame.

Definition

setFrameTriggered()

getFrameTriggered

Description

Get the frame trigger flag which shows if this binding has triggered an event this frame.

Definition

getFrameTriggered()

setComboMask

Description

Set and store the combo bit mask for this binding.
This is agnostic of the actually bound device because a binding can only map input on one specific device.

Definition

setComboMask(int comboMask)

Arguments

int comboMask Combo mask

getComboMask

Description

Get the combo bit mask.

Definition

getComboMask()

Return Values

int Combo bit mask

hasCollisionWith

Description

Check if there is a collision between this binding and another one.
Two bindings collide if they bind the same input on the same axes of the same device.

Definition

hasCollisionWith(otherBinding Other)

Arguments

otherBinding Other binding to check for collision

Return Values

True if there is a collision.

isAlternativeTo

Description

Check if this binding is an alternative to another one on the same action.
Two bindings are alternatives if they bind different input on the same axis of the same device category to the same action.

Definition

isAlternativeTo(otherBinding Other)

Arguments

otherBinding Other binding to check if it's an alternative

Return Values

True if this and the other binding are alternatives of each other

clone

Description

Create a new instance of Binding with the same state as this instance.

Definition

clone()

Return Values

Cloned Binding instance

copyInputStateFrom

Description

Copy binding input state from another binding.

Definition

copyInputStateFrom()

Return Values

integer ret ret

integer total total

isSameSlot

Description

Check if this binding occupies the same binding slot as another binding on the same action.

Definition

isSameSlot()

getOppositeAxisComponent

Description

Get the opposite axis component for a given axis component.

Definition

getOppositeAxisComponent(axisComponent One)

Arguments

axisComponent One of the constant values of Binding.AXIS_COMPONENT

Return Values

integer changedValue changed value

getOppositeInputComponent

Description

Get the opposite input component for a given input component.

Definition

getOppositeInputComponent(inputComponent One)

Arguments

inputComponent One of the constant values of Binding.INPUT_COMPONENT

Return Values

integer changedArea changed area pixels

integer totalArea total area

makeId

Description

Make and assign a binding ID.

Definition

makeId()

needJapanesePlaystationButtonSwap

Description

Determine if we need to swap buttons for Japanese Playstation controllers by their convention.

Definition

needJapanesePlaystationButtonSwap()

Return Values

integer changedArea changed area pixels

integer totalArea total area pixles

swapJapanesePlaystationButtons

Description

Swaps buttons for Japanese Playstation controllers if contained in the given axis names.

Definition

swapJapanesePlaystationButtons(table axisNames)

Arguments

table axisNames Axis names array, will be changed in place

toString

Description

Get a string representation of this binding.

Definition

toString()

Return Values

integer numPixels number of pixels

integer totalNumPixels total number of pixels

BitmapElement

Description

Display element for images.
-- Used layers: "image" for the display image.
--@xmlConfig GuiElement#offset string [optional] Position offset of the displayed image relative to this element's origin in reference resolution, defaults to [0, 0]. Format: "[x]px [y]px".

setImageColor

Description

Set this element's image color.
Omitted (nil value) color values have no effect and the previously set value for that channel is used.

Definition

setImageColor(state GuiOverlay, r Red, g Green, b Blue, a Alpha)

Arguments

state GuiOverlay state for which the color is changed, use nil to set the default color

r Red color value

g Green color value

b Blue color value

a Alpha value (transparency)

setImageRotation

Description

Set this element's image overlay's rotation.

Definition

setImageRotation(float rotation)

Arguments

float rotation Rotation in radians

BoxLayoutElement

Description

Layout element which lays out child elements in regular rows or columns.
-- Exceptions are elements whose "layoutIgnore" property is true.
-- Used layers: "image" for a background image.
--@category GUI

--@xmlConfig GuiElement#alignmentX string [optional] Horizontal alignment of the layout, defaults to "left". Valid values are "left", "right" and "center".

getLayoutCells

Description

Extract a flow / cell data table from this box layout's elements.

Definition

getLayoutCells(ignoreVisibility Visibility)

Arguments

ignoreVisibility Visibility flag for element eligibility

Return Values

Flows and cells as nested tables: [flowIndex][cell index] = cell data {element, flowSize, lateralSize}

getLayoutSizes

Description

Calculate layout sizes based on flow cell data.

Definition

getLayoutSizes()

Return Values

List of lateral flow sizes, total lateral sizes (sum of lateral flow sizes), maximum flow size in direction of flow

getAlignmentOffset

Description

Get the layout space starting position offset for elements based on flow dimensions and alignments.

Definition

getAlignmentOffset(flowSize Flow, flowLateralSize Total)

Arguments

flowSize	Flow	size in flow direction (e.g. height for vertical flows)
flowLateralSize	Total	flow size orthogonal to flow direction (e.g. combined width of all columns for vertical flows)

Return Values

Layout X starting offset, layout Y starting offset, Layout X direction {-1|1}, Layout Y direction {-1|1}

getElementAlignmentOffset

Description

Calculate element offsets for margins and to counteract alignment updates in GuiElement.updateAbsolutePositions()

Definition

getElementAlignmentOffset(directionX Layout, directionY Layout)

Arguments

directionX Layout X direction as a signed factor {-1|1}

directionY Layout Y direction as a signed factor {-1|1}

Return Values

element X offset, element Y offset

applyCellPositions

Description

Apply layout positions to all given cells' elements.

Definition

applyCellPositions(flowCells List, offsetStartX Layout, offsetStartY Layout, directionX Layout, directionY Layout, lateralFlowSizes List)

Arguments

flowCells	List	of flows and their cell data
offsetStartX	Layout	X starting offset
offsetStartY	Layout	Y starting offset
directionX	Layout	X direction as a signed factor {-1|1}
directionY	Layout	Y direction as a signed factor {-1|1}
lateralFlowSizes	List	of lateral flow sizes (e.g. widths of columns in vertical flows)

focusLinkCells

Description

Link layout cells' elements for focus navigation.

Definition

focusLinkCells(flowCells List)

Arguments

flowCells List of flows and their cell data

focusLinkChildElement

Description

Link a child element to others for focus navigation

Definition

focusLinkChildElement(element Current, previousElement Element, firstElement First, lastElement Last)

Arguments

element	Current	element
previousElement	Element	at previous position in layout flow, will be nil when the first element is being processed
firstElement	First	element in layout
lastElement	Last	element in layout

invalidateLayout

Description

Invalidate the layout, will reposition all elements and update focus navigation.

Definition

invalidateLayout(ignoreVisibility If)

Arguments

ignoreVisibility If true, elements will be considered for layouting even if invisible

getFocusTarget

Description

Return the actual focus target when this box layout is being focused. This can be a previously focused element within
the layout (depends on rememberLastFocus attribute), a programmatically defined entry point based on the incoming
direction or simply the first layout element (when no other options apply).

Definition

getFocusTarget(incomingDirection Focus, moveDirection Actual)

Arguments

incomingDirection Focus	navigation direction from where this layout is being entered
moveDirection Actual	focus navigation direction per input

Return Values

GuiElement which should receive focus instead

ButtonElement

Description

Clickable button element.
-- Used layers: "image" for the background, "icon" for a button glyph.
-- All button UI callbacks do not require or provide any arguments.
--@category GUI
--@xmlConfig GuiElement#iconSize string [optional] Pixel size of button glyph in reference resolution. Format: "[width]px [height]px"

loadInputGlyphColors

Description

Load glyph overlay colors.

Definition

loadInputGlyphColors(profile If, xmlFile If, key XML)

Arguments

profile If	set, loads overlay properties from this button's GUI profile
xmlFile If	set, loads overlay properties from this button's XML configuration
key XML	base configuration node of this button

loadInputGlyph

Description

Load the actual input glyph symbols to display if an input action is defined on the button.

Definition

loadInputGlyph()

setKeyboardMode

Description

Set the keyboard mode flag.

Definition

setKeyboardMode()

setInputAction

Description

Set the input action for the display glyph by name.

Definition

setInputAction()

setImageUVs

Description

Set UV coordinates for the button background and/or icon.

Definition

setImageUVs()

getIsSelected

Description

Determine if this button is selected

Definition

getIsSelected()

getIsHighlighted

Description

Determine if this button is highlighted

Definition

getIsHighlighted()

getIconModifiedTextOffset

Description

Get modified text offset including changes from icon position and dimensions.

Definition

getIconModifiedTextOffset(float textOffsetX, float textOffsetY)

Arguments

float textOffsetX Screen space text X offset

float textOffsetY Screen space text Y offset

Return Values

float Modified X offset

float Modified Y offset

getTextOffset

Description

Get text offset from element position including modifications from icon.

Definition

getTextOffset()

getText2Offset

Description

Get shadow text offset from element position including modifications from icon.

Definition

getText2Offset()

getIconSize

Description

Get the current icon size in screen space.

Definition

getIconSize()

updateSize

Description

Update size of element depending on content

Definition

updateSize()

ButtonOverlay
Description

Keyboard button display overlay.
-- Overlay type which displays a keyboard key button symbol.
--@category GUI

delete
Description

Delete this button overlay.

Definition

delete()

setColor
Description

Set this overlay's background color.

Definition

setColor(r Red, g Green, b Blue, a Alpha)

Arguments

r	Red	channel [0, 1]
g	Green	channel [0, 1]
b	Blue	channel [0, 1]
a	Alpha	(transparency) channel [0, 1], 0 is fully transparent, 1 is opaque

renderButton
Description

Render this overlay with the given parameters.

Definition

renderButton(buttonText Text, posX Screen, posY Screen, height Button, alignment Text)

Arguments

buttonText	Text	to display as the key value, e.g. "A", "Space", "Ctrl", etc.
posX	Screen	x position
posY	Screen	y position
height	Button	display height
alignment	Text	alignment, one of the constants of RenderText.ALIGN_[...]

getButtonWidth
Description

Get the total display width of this button overlay for a given button text and height

Definition

getButtonWidth()

CareerScreen
Description

Career Screen.
Displays available save game slots.
-- @field savegameList Save game list

setIsWaitingForSaveGameInfo

Description

Set UI waiting state, which displays or hides a dialog that says to wait.

Definition

setIsWaitingForSaveGameInfo()

CharacterCreationScreen
Description

Character Selection Screen.

setCharacterIndex
Description

Set the index of the character. Is turned into a player and body pair.

Definition

setCharacterIndex()

updateCharacterWithSettings
Description

Update the character with the currently configured options.

Definition

updateCharacterWithSettings()

updateCharacter
Description

Update the character to match given options. Only reloads object if needed.

Definition

updateCharacter()

loadCharacterFinished
Description

Loading of the character object finished

Definition

loadCharacterFinished()

updateCharacterOptions
Description

Update the existing character with options that can be changed

Definition

updateCharacterOptions()

setHatHairNodeVisibility
Description

Set the visbility of the 'hat hair': special hair when wearing a hat

Definition

setHatHairNodeVisibility()

ChatDialog
Description

Multiplayer chat dialog
-- @field textElement User chat message input element

onMenuAxisUpDown
Description

Handle menu up/down input.
Scrolls through the chat history.

Definition

onMenuAxisUpDown()

ChatWindow
Description

HUD chat window.
-- Displays chat messages.
--@category GUI

new
Description

Create a new ChatWindow.

Definition

new(string hudAtlasPath)

Arguments

string hudAtlasPath Path to the HUD atlas texture.

Return Values

table ChatWindow instance

setChatMessages
Description

Set the chat message history reference for displaying.
The array is owned by the caller and must not be modified.

Definition

setChatMessages(table messages)

Arguments

table messages Messages array as {i={msg=<message text>, sender=<sender user nickname>}}

scrollChatMessages
Description

Scroll chat messages by a given amount.

Definition

scrollChatMessages(int delta, int numMessages)

Arguments

int delta Number of lines (positive or negative) to scroll

int numMessages Number of currently stored chat messages

onMenuVisibilityChange
Description

Handle menu visibility state change.

Definition

onMenuVisibilityChange()

update
Description

Update element state.

Definition

update()

draw

Description

Draw the chat window.

Definition

draw()

setScale

Description

Set this element's UI scale.

Definition

setScale()

getBackgroundPosition

Description

Get this element's base background position.

Definition

getBackgroundPosition(float uiScale)

Arguments

float uiScale Current UI scale factor

storeScaledValues

Description

Store scaled positioning, size and offset values.

Definition

storeScaledValues()

createBackground

Description

Create the background overlay.

Definition

createBackground()

ColorPickerDialog

Description

Color Picker Dialog
-- Lets the player pick from a set of colors.
-- @field buttonTemplate Color button template which is cloned per color

setColors

Description

Set colors after opening.

Definition

setColors(table colors, table defaultColor)

Arguments

table colors Array of colors {i={r, g, b, a}, or {i={name="name", color={r,g,b,a}}}

table defaultColor Default color setting as {r, g, b, a}, must be contained in colors array

focusLinkColorButtons

Description

Apply custom grid-based focus linking

Definition

focusLinkColorButtons()

setInitialFocus

Description

Set the initial focus when entering this dialog.

Definition

setInitialFocus()

setCallback

Description

Set the dialog callback.

Definition

setCallback()

onClickColorButton

Description

Handle activation of a color button.

Definition

onClickColorButton()

ContextActionDisplay

Description

Player context action display element.
-- Displays information about the current interaction context. Includes action names and current input scheme button
glyphs.
--@category GUI

new

Description

Create a new instance of ContextActionDisplay.

Definition

new(string hudAtlasPath)

Arguments

string hudAtlasPath Path to the HUD texture atlas

setContext

Description

Sets the current action context.
This must be called each frame when a given context is active. The highest priority context is displayed or the one
which was set the latest if two or more contexts have the same priority.

Definition

setContext(string contextAction, string contextIconName, string targetText, int priority, string actionText)

Arguments

string contextAction	Input action name of the context action
string contextIconName	Name of the icon to display for the action context, use one of ContextActionDisplay.CONTEXT_ICON
string targetText	Display text which describes the context action target
int priority	[optional, default=0] Context priority, a higher number has higher priority.
string actionText	[optional] Context action description, if different from context action description

update

Description

Update the context action display state.

Definition

update()

resetContext

Description

Reset context state after drawing.
The context must be set anew on each frame.

Definition

resetContext()

draw

Description

Draw the context action display.

Definition

draw()

setScale

Description

Set the scale of this element.

Definition

setScale()

storeScaledValues

Description

Store scaled positioning, size and offset values.

Definition

storeScaledValues()

getBackgroundPosition

Description

Get the position of the background element, which provides this element's absolute position.

Definition

getBackgroundPosition(scale Current, float width)

Arguments

scale Current	UI scale

float width Scaled background width in pixels

Return Values

float X position in screen space

float Y position in screen space

createBackground

Description

Create an empty background overlay as a base frame for this element.

Definition

createBackground()

createComponents

Description

Create display components.

Definition

createComponents(string hudAtlasPath, table inputDisplayManager)

Arguments

string hudAtlasPath Path to HUD atlas texture

table inputDisplayManager InputDisplayManager reference

createInputGlyph

Description

Create the input glyph element.

Definition

createInputGlyph()

createFrame

Description

Create the context display frame.

Definition

createFrame()

createActionIcons

Description

Create action context icons.
Only one of these will be visible at any time.

Definition

createActionIcons()

ControlsController

Description

Controls settings controller.
-- Handles the input controls model and control logic for settings screen controls pages.
--@category GUI

new

Description

Create a new ControlsController instance.

Definition

new()

setMessageCallback

Description

Set a callback for displaying status messages.
Callback signature: messageCallback(messageId, additionalText, addLine)

Definition

setMessageCallback(messageCallback A)

Arguments

messageCallback A	function which takes a message ID (see class constants), a optional list of additional strings and an optional addition flag (add instead of overwrite) to display a message.

setInputDoneCallback

Description

Set a callback to react on input capture completion.
Callback signature: inputDoneCallback(madeChange)

Definition

setInputDoneCallback(inputDoneCallback A)

Arguments

inputDoneCallback A	function to notify a collaborator that input gathering has finished and if there were any changes.

createDisplayAction

Description

Create a display action for displaying from bindings for a given axis direction.

Definition

createDisplayAction(deviceCategory Device, actionBinding Action, isAxisPositive If)

Arguments

deviceCategory	Device	category, only bindings of devices of this category are displayed
actionBinding	Action	binding table reference: {action=Action, bindings={1=binding1, 2=binding2, ...}}
isAxisPositive	If	true, the display action should show a positive action axis.

Return Values

{name="internalName", displayName="displayName",		inputTexts={1=text1, 2=text2, ...}, inputBindings={1=binding1, 2=binding2, ...}, positiveInput=[true|false]}

getDeviceCategoryActionBindings

Description

Get available action bindings per device category, including current changes (even before saving).

Definition

getDeviceCategoryActionBindings(deviceCategory Device)

Arguments

deviceCategory	Device category name

Return Values

List of DisplayActionBinding instances

getMouseAxisDisplayText

Description

Get display text for a single mouse axis.

Definition

getMouseAxisDisplayText()

getGamepadAxisDisplayText

Description

Get display text for a single gamepad axis.

Definition

getGamepadAxisDisplayText()

getBindingInputDisplayText

Description

Get display text for input axis names of a binding.

Definition

getBindingInputDisplayText(Binding Binding)

Arguments

Binding Binding reference

Return Values

Display

saveChanges

Description

Save changed bindings to user configuration.

Definition

saveChanges()

discardChanges

Description

Discard any previously made binding changes.

Definition

discardChanges()

loadBindings

Description

Load action input bindings.

Definition

loadBindings()

onClickInput

Description

Handles clicking an input binding button for a given device category and action.

Definition

onClickInput(deviceCategory Input, bindingId ID, actionBinding Action)

Arguments

deviceCategory	Input	device category
bindingId	ID	of binding on the current device. E.g. 1 for main binding, 2 for secondary binding, etc.
actionBinding	Action	binding table for the clicked action

Return Values

True if the controller starts listening for input, false otherwise (also if it is already listening!)

beginWaitForInput

Description

Begin waiting for input.
Overrides the global input hooks keyEvent() and mouseEvent() as well as the global update() loop, depending on which
device is being scanned. Also see InputBinding:startInputCapture().

Definition

beginWaitForInput(deviceCategory Category, bindingIndex Binding, DisplayActionBinding DisplayActionBinding)

Arguments

deviceCategory	Category	of device which is being scanned for input
bindingIndex	Binding	(primary, secondary, etc.) index which the scanned input will be assigned to.
DisplayActionBinding	DisplayActionBinding	instance which describes the action binding to re-map.

onAbortInputGathering

Description

Input gathering abort event.

Definition

onAbortInputGathering()

onDeleteInputBinding

Description

Input gathering binding deletion event.

Definition

onDeleteInputBinding(DisplayActionBinding DisplayActionBinding, gatheringState {binding=DisplayActionBinding)

Arguments

DisplayActionBinding	DisplayActionBinding	reference of the binding to delete
gatheringState	{binding=DisplayActionBinding	reference of the binding to change, bindingIndex=Binding index}

onCaptureKeyboardInput

Description

Input gathering keyboard input event.

Definition

onCaptureKeyboardInput(keyName Key, isModifier True, inputValue 1, gatheringState {binding=DisplayActionBinding)

Arguments

keyName	Key	name as defined in Input
isModifier	True	if the key is used as a modifier for a combined input
inputValue	1	for pressed, 0 for inactive
gatheringState	{binding=DisplayActionBinding	reference of the binding to change, bindingIndex=Binding index, keyState={[keyName]=[lastValue]}}

onCaptureMouseInput

Description

Input gathering mouse input event.

Definition

onCaptureMouseInput(inputAxisName Mouse, isModifier True, inputValue Input, gatheringState {binding=DisplayActionBinding)

Arguments

inputAxisName	Mouse	axis or button name as defined in Input
isModifier	True	if the input axis is used as a modifier for a combined input
inputValue	Input	value of the input axis
gatheringState	{binding=DisplayActionBinding	reference of the binding to change, bindingIndex=Binding index, mouseState={[axisName]=[lastValue]}}

onCaptureGamepadInput

Description

Input gathering gamepad / controller event.

Definition

onCaptureGamepadInput(deviceId Device, inputAxisName Axis, isModifier True, inputValue Input, gatheringState {binding=DisplayActionBinding)

Arguments

deviceId	Device	ID of the gamepad / controller which has received input
inputAxisName	Axis	or button name as defined in Input
isModifier	True	if the input axis is used as a modifier for a combined input
inputValue	Input	value of the input axis
gatheringState	{binding=DisplayActionBinding	reference of the binding to change, bindingIndex=Binding index, gamepadState={[deviceId]={[axisName]=[lastValue]}}}

endWaitForInput

Description

Stop waiting for input.
Lifts the input override and notifies the screen listener.

Definition

endWaitForInput(madeChange If)

Arguments

madeChange If true, the player has changed at least one binding.

lockInput

Description

Lock all UI input after capturing a new binding.

Definition

lockInput()

deleteBinding

Description

Delete a binding from its action binding table.

Definition

deleteBinding()

Return Values

bool True if an actual binding was deleted, false otherwise

assignKeyboardBinding

Description

Assign a (new) keyboard input binding.

Definition

assignKeyboardBinding(DisplayActionBinding DisplayActionBinding, bindingIndex Binding, keyames List)

Arguments

DisplayActionBinding	DisplayActionBinding	instance which holds information about the binding to be added or updated
bindingIndex	Binding	index to set, 1 primary, 2 secondary, etc.
keyames	List	of input key names to bind

Return Values

True if a binding change was made, false otherwise

validateMouseCombo

Description

Check if the given axis names contain a supported mouse combo.

Definition

validateMouseCombo()

assignMouseBinding

Description

Assign a (new) mouse input binding.

Definition

assignMouseBinding(DisplayActionBinding DisplayActionBinding, inputAxisNames List, inputDirection Numeric)

Arguments

DisplayActionBinding	DisplayActionBinding	instance which holds information about the binding to be added or updated
inputAxisNames	List	of input axis names (axes and buttons) to bind
inputDirection	Numeric	value which defines the physical input axis direction component to bind. Values greater than or equal to 0 will be interpreted as positive, lesser than 0 as negative.

Return Values

True if a binding change was made, false otherwise

assignGamepadBinding

Description

Assign a (new) gamepad / controller input binding.

Definition

assignGamepadBinding(DisplayActionBinding DisplayActionBinding, bindingIndex Binding, deviceId Gamepad, inputAxisNames List, inputDirection Numeric)

Arguments

DisplayActionBinding	DisplayActionBinding	instance which holds information about the binding to be added or updated
bindingIndex	Binding	index to set, 1 primary, 2 secondary, etc.
deviceId	Gamepad	/ controller device ID of the input to bind
inputAxisNames	List	of input axis names (axes and buttons) to bind
inputDirection	Numeric	value which defines the physical input axis direction component to bind. Values greater than or equal to 0 will be interpreted as positive, lesser than 0 as negative.

Return Values

True if a binding change was made, false otherwise

assignBinding

Description

Assign a (new) binding.
First tries to update an existing binding. If no binding with the given parameters exists to be updated, a new
binding is created instead, unless this would lead to an number of alternative bindings exceeding the maximum.

Definition

assignBinding(deviceId Binding, displayAction Cloned, bindingIndex Intended, inputAxisNames List, isPositiveAxis If, inputDirection Numeric, previousDeviceId [optional])

Arguments

deviceId	Binding	device ID
displayAction	Cloned	InputAction passed back from display and capture process
bindingIndex	Intended	binding index, 1 primary, 2 secondary, etc.
inputAxisNames	List	of input axis names to bind
isPositiveAxis	If	true, the binding covers the positive part of the action's logical axis (always true for half-axis / button / key actions). False covers the negative part.
inputDirection	Numeric	value which defines the physical input axis direction component to bind. Values greater than or equal to 0 will be interpreted as positive, lesser than 0 as negative.
previousDeviceId	[optional]	Previous device ID of an existing binding. If not set, will use the given deviceId parameter instead.

Return Values

True if binding has been assigned; reference to first colliding binding or nil

loadDefaultSettings

Description

Load default input binding settings from the game's profile template.
Handle with care, this will overwrite all the player's input binding settings.

Definition

loadDefaultSettings()

DataGrid

Description

Creating data grid

new

Description

@param table customMt custom metatable

Definition

new()

Return Values

table instance instance of object

Code

```
19  function DataGrid:new(numRows, numColumns, customMt)
20  local self = {}
21  setmetatable(self, customMt or DataGrid_mt)
22
23  self.grid = {}
24  self.numRows = numRows
25  self.numColumns = numColumns
26  for i=1, numRows do
27  table.insert(self.grid, {})
28  end
29
30  return self
31  end
```

delete

Description

Deletes data grid

Definition

delete()

Code

```
35  function DataGrid:delete()
36  self.grid = nil
37  end
```

getValue

Description

@param integer colIndex index of column

Definition

getValue()

Return Values

table value value at the given position

Code

```
44  function DataGrid:getValue(rowIndex, colIndex)
45  if rowIndex < 1 or rowIndex > self.numRows then
46  g_logManager:error("rowIndex out of bounds!")
47  printCallstack()
48  return nil
```

```
49  end
50  if colIndex < 1 or colIndex > self.numColumns then
51  g_logManager:error("colIndex out of bounds!")
52  printCallstack()
53  return nil
54  end
55
56  return self.grid[rowIndex][colIndex]
57  end
```

setValue

Description

@param integer colIndex index of column

Definition

setValue(table value)

Arguments

table value value at the given position

Code

```
64  function DataGrid:setValue(rowIndex, colIndex, value)
65  if rowIndex < 1 or rowIndex > self.numRows then
66  g_logManager:error("rowIndex out of bounds!")
67  printCallstack()
68  return false
69  end
70  if colIndex < 1 or colIndex > self.numColumns then
71  g_logManager:error("colIndex out of bounds!")
72  printCallstack()
73  return false
74  end
75
76  self.grid[rowIndex][colIndex] = value
77  return true
78  end
```

DepthOfFieldState

Description

Depth of field shader state.
-- This class wraps the depth of field shader parameter calls to allow convenient setting and resetting a specific
shader parameter state.
--@category GUI

new

Description

Create a new depth of field state.
This will record the currently set parameters which are restored when calling reset().

Definition

new(float nearCoCRadius, float nearBlurEnd, float farCoCRadius, float farBlurStart, float farBlurEnd)

Arguments

float nearCoCRadius	Near circle of confusion radius (nearCoCRadius = 0 means no near blur (pinhole camera))
float nearBlurEnd	Distance from the camera center where near blur ends
float farCoCRadius	Far circle of confusion radius (farCoCRadius = 0 means no far blur (pinhole camera))
float farBlurStart	Distance from the camera center where far blur starts
float farBlurEnd	Distance from the camera center where far blur ends

recordState

Description

Record the current shader parameters for reset.
This is called on instantiation. If another state should be recorded, call this again afterwards.

Definition

recordState()

setParameters

Description

Set one or more parameters after instantiation.
If any of the parameters is omitted, the currently stored value is used.

Definition

setParameters(float nearCoCRadius, float nearBlurEnd, float farCoCRadius, float farBlurStart, float farBlurEnd)

Arguments

float nearCoCRadius	Near circle of confusion radius (nearCoCRadius = 0 means no near blur (pinhole camera))
float nearBlurEnd	Distance from the camera center where near blur ends
float farCoCRadius	Far circle of confusion radius (farCoCRadius = 0 means no far blur (pinhole camera))
float farBlurStart	Distance from the camera center where far blur starts
float farBlurEnd	Distance from the camera center where far blur ends

apply

Description

Apply shader parameters which were provided on instantiation.

Definition

apply()

reset

Description

Reset the shader parameters to the state they were in when this object had been created.

Definition

reset()

createFullscreenBlur

Description

Create a depth of field state for a full screen blur effect.

Definition

createFullscreenBlur()

DirectSellDialog
Description

Vehicle selling or customization dialog
-- Opened when activating vehicle selling points.
-- @field headerText Dialog header text element

new
Description

Create a new DirectSellDialog instance.

Definition

new(table target, table custom_mt, table shopConfigScreen, table messageCenter)

Arguments

table target	DialogElement controller reference
table custom_mt	[optional] Sub-class meta table for inheritance
table shopConfigScreen	ShopConfigScreen reference for vehicle selling / customization
table messageCenter	MessageCenter reference for local network UI event handling

Return Values

table DirectSellDialog instance

onClickOk
Description

Handle "sell" button event.

Definition

onClickOk()

onClickActivate
Description

Handle "customize" button event.

Definition

onClickActivate()

DisplayActionBinding
Description

Holds information about an action binding for display purposes.
-- This class is used to transfer input binding information between ControlsController and UI components.
--@category Input

new
Description

Creates a new DisplayActionBinding.

Definition

new(action InputAction, isPositive If, displayName Display, Binding Bindings)

Arguments

action	InputAction of this binding

isPositive	If	true, this display action binding shows the binding for the positive input axis (always true for half-axis actions)
displayName	Display	name of this action binding for the given input axis direction.
Binding	Bindings	associated with the input axis component for the given action

setBindingDisplay

Description

Set display information for binding.

Definition

setBindingDisplay(Binding Binding, text Binding, column Display)

Arguments

Binding	Binding	reference contained in this instance
text	Binding	display text (button/key/axes names)
column	Display	column index (1 for primary, 2 for secondary, etc)

Return Values

integer	numPixels	number of pixels
integer	totalNumPixels	total number of pixels

EditFarmDialog

Description

Farm edit and create dialog.
-- Lets a player edit a farm's properties or create a new farm.
-- @field buttonTemplate Color button template which is cloned per color

new

Description

Create a new instance of EditFarmDialog.

Definition

new(table target, table custom_mt, table l10n, table farmManager)

Arguments

table target	Optional input target override
table custom_mt	Sub-class metatable
table l10n	I18N reference for localization
table farmManager	FarmManager reference for farm data access

setExistingFarm

Description

Set existing farm ID whose properties are to be modified.
If no farm ID is given, a new farm will be created.

Definition

setExistingFarm()

storeAvailableColors

Description

Find available farm colors and store them in fields.

Definition

storeAvailableColors()

onClickColorButton

Description

Handle activation of a color button.
Shadows parent method which exits the dialog right away.

Definition

onClickColorButton()

onClickActivate

Description

Handle dialog confirmation click / activation.
Depending on the mode, this will create a new farm or update an existing one.

Definition

onClickActivate()

Farm

Description

Permission types.

merge

Description

Merge another farm into this farm. Used for creating an SP game from an MP game. (this is mutating)

Definition

merge(table other)

Arguments

table other Another Farm

getFarmhouse

Description

Get the farmhouse associated with the farm.

Definition

getFarmhouse()

Return Values

table farmhouse or nil

getSpawnPoint

Description

Get the spawnpoint associated with the farm(house).

Definition

getSpawnPoint()

Return Values

integer spawnpoint node or the career spawnpoint node.

getSleepCamera

Description

Get the sleep camera.

Definition

getSleepCamera()

Return Values

integer Camera or 0 if no farmhouse.

getActiveUsers

Description

Get a list of active users. Useful for using their connection ID

Definition

getActiveUsers()

isUserFarmManager

Description

Determine if a user is a manager of this farm.

Definition

isUserFarmManager(userId User)

Arguments

userId User ID

Return Values

bool True if the user is a manager of this farm, false otherwise

getUserPermissions

Description

Get the farm permissions of a user.

Definition

getUserPermissions(userId User)

Arguments

userId User ID

Return Values

table Permission hash table {permission=<hasPermission>}

setUserPermission

Description

Set a user's permission in this farm.

Definition

setUserPermission(userId User, string permission, bool hasPermission)

Arguments

userId	User	ID
string	permission	Permission key from Farm.PERMISSION
bool	hasPermission	True if the permission is to be granted, false to be denied

promoteUser

Description

Promote a user to farm manager.

Definition

promoteUser()

demoteUser

Description

Demote a user from farm manager.

Definition

demoteUser()

setIsContractingFor
Description

Update contracting status

Definition

setIsContractingFor(noSendEvent boolean)

Arguments

noSendEvent boolean Send no event, forces setting of actual value without server feedback

changeBalance
Description

Add or remove money from the farm

Definition

changeBalance(number amount)

Arguments

number amount Amount to add (positive) or remove (negative)

getBalance
Description

Get the current account balance of the farm.

Definition

getBalance()

Return Values

float Account balance

getLoan
Description

Get the current loan of the farm.

Definition

getLoan()

Return Values

float

getHandTools
Description

Get a list of filenames for accessible handtools

Definition

getHandTools()

addUser
Description

Add a new user to the farm. This adds it to the players and active players list.
And also gives default permissions.

Definition

addUser()

removeUser
Description

Remove a user from the farm.

Definition

removeUser()

onUserJoinGame

Description

Called when a user joins the game. Active users is updated, and for spectator.
a new user might be added. Server only.

Definition

onUserJoinGame()

onUserQuitGame

Description

Called when a user quits the game. The active user list is updated. Server only.

Definition

onUserQuitGame()

FarmManager

Description

Multiplayer

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
27  function FarmManager:new(customMt)
28  local self = AbstractManager:new(customMt or FarmManager_mt)
29
30  return self
31  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
43  function FarmManager:loadMapData(xmlFile)
44  FarmManager:superClass().loadMapData(self)
45
46  if g_currentMission:getIsServer() then
```

```
47  g_currentMission:addUpdateable(self)
48
49  -- Create spectator farm
50  local spectatorFarm = Farm:new(true, g_client ~= nil, nil, true)
51  spectatorFarm.farmId = FarmManager.SPECTATOR_FARM_ID
52  spectatorFarm.isSpectator = true
53
54  spectatorFarm:register()
55
56  table.insert(self.farms, spectatorFarm)
57  self.farmIdToFarm[spectatorFarm.farmId] = spectatorFarm
58  end
59
60  addConsoleCommand("gsSetFarm", "Set farm for current player or
    vehicle", "consoleCommandSetFarm", self)
61
62  if g_addTestCommands then
63  addConsoleCommand("debugCreateFarm", "Create a new farm",
    "consoleCommandCreateFarm", self)
64  end
65  end
```

unloadMapData

Description

Unload data on mission delete

Definition

unloadMapData()

Code

```
69  function FarmManager:unloadMapData()
70  g_currentMission:removeUpdateable(self)
71
72  removeConsoleCommand("gsSetFarm")
73  if g_addTestCommands then
74  removeConsoleCommand("debugCreateFarm")
75  end
76
77  FarmManager:superClass().unloadMapData(self)
78  end
```

saveToXMLFile

Description

Write field mission data to savegame file

Definition

saveToXMLFile(string xmlFilename)

Arguments

string xmlFilename file path

Return Values

boolean true if loading was successful else false

Code

```
84  function FarmManager:saveToXMLFile(xmlFilename)
85  local xmlFile = createXMLFile("farmsXML", xmlFilename, "farms");
86
87  local index = 0
88  for i, farm in ipairs(self.farms) do
89  if farm.farmId ~= 0 then
90  local key = string.format("farms.farm(%d)", index)
91
92  farm:saveToXMLFile(xmlFile, key)
93  index = index + 1
94  end
95  end
96
97  saveXMLFile(xmlFile)
98  delete(xmlFile)
99  end
```

loadFromXMLFile

Description

Load fieldjob data from xml savegame file

Definition

loadFromXMLFile(string filename)

Arguments

string filename xml filename

Code

```
104  function FarmManager:loadFromXMLFile(xmlFilename)
105  if xmlFilename == nil then
106  self:loadDefaults()
107  return false
108  end
109
110  local xmlFile = loadXMLFile("TempXML", xmlFilename)
111  if not xmlFile then
112  return false
113  end
114
```

```
115 local i = 0
116 while true do
117 local key = string.format("farms.farm(%d)", i)
118 if not hasXMLProperty(xmlFile, key) then
119 break
120 end
121
122 local farm = Farm:new(true, g_client ~= nil)
123
124 if not farm:loadFromXMLFile(xmlFile, key) then
125 farm:delete()
126 else
127 farm:register()
128
129 table.insert(self.farms, farm)
130 self.farmIdToFarm[farm.farmId] = farm
131 end
132
133 i = i + 1
134 end
135
136 self:mergeFarmsForSingleplayer()
137
138 -- Emulate an MP player join for local MP or SP game
139 if g_currentMission:getIsClient() then
140 local uniqueUserId =
    g_currentMission.missionDynamicInfo.isMultiplayer and
    getUniqueUserId() or FarmManager.SINGLEPLAYER_UUID
141 self:playerJoinedGame(uniqueUserId,
    g_currentMission:getServerUserId())
142 end
143
144 g_fieldManager:updateFieldOwnership()
145
146 delete(xmlFile)
147
148 return true
149 end
```

mergeFarmlandsForSingleplayer

Description

Second step of merging: transfer all lands to the singleplayer farm

Definition

mergeFarmlandsForSingleplayer()

mergeObjectsForSingleplayer

Description

Third step of merging: move vehicles and bales to singleplayer farm

Definition

mergeObjectsForSingleplayer()

delete

Description

Deletes field mission manager

Definition

delete()

Code

```
236  function FarmManager:delete()
237  end
```

update

Description

Updates field mission ownage data from xml savegame file

Definition

update(string filename)

Arguments

string filename xml filename

Code

```
242  function FarmManager:update(dt)
243  if g_currentMission:getIsClient() then
244  if self.spFarmWasMerged and not self.mergedMessageShown then
245  g_gui:showInfoDialog({visible=true,
     text=g_i18n:getText("ui_farmedMergedSP"),
     dialogType=DialogElement.TYPE_INFO, isCloseAllowed=true})
246  self.mergedMessageShown = true
247  end
248  end
249  end
```

getFarmForUniqueUserId

Description

Get farm for given userId. To be used when player is not in the game

Definition

getFarmForUniqueUserId()

getFarmByUserId

Description

Get farm for given userId. To be used when player is in the game.

Definition

getFarmByUserId()

updateFarms

Description

On client, update the list of farms and set farm for given farmId

Definition

updateFarms()

getFarms

Description

Get the array of known farms.
Callers should not modify this array.

Definition

getFarms()

transferMoney

Description

Transfer an amount of money from the current user's farm to a destination farm.
Triggers a network event which checks farm balances and applies the change. Successful execution requires the current
user to have permission to transfer money as well as their current farm to have a sufficient balance.

Definition

transferMoney()

removeUserFromFarm

Description

Remove player from their farm. Only works if the caller has permission (master user, farm manager)

Definition

removeUserFromFarm()

removeFarm

Description

Farm has been destroyed. Remove from lists

Definition

removeFarm()

FieldInfoDisplay

Description

HUD field information display element.
-- Displays dynamic information about the field which the player is currently standing in (or close by).
--@category GUI

new

Description

Create a new instance of FieldInfoDisplay.

Definition

new(string hudAtlasPath)

Arguments

string hudAtlasPath Path to the HUD texture atlas

setupRows

Description

Set up rows data structures.

Definition

setupRows()

setPlayer

Description

Set the local player reference.

Definition

setPlayer()

setFruitType

Description

Set the current fruit type to display.

Definition

setFruitType(int fruitTypeIndex, int fruitGrowthState)

Arguments

int fruitTypeIndex Index of the field fruit type or a number <= 0 to clear the fruit type.

int fruitGrowthState Current growth state of the given fruit type

setOwnerFarmId

Description

Set the field owner

Definition

setOwnerFarmId(int ownerFarmId)

Arguments

int ownerFarmId Current owner farm id

setFertilization

Description

Set the fertilization factor info to display.

Definition

setFertilization(float fertilizationFactor)

Arguments

float fertilizationFactor Current fertilization factor to display, this will be converted to a percentage. Values < 0 clear this info.

setWeed

Description

Set the weed factor info to display.

Definition

setWeed(float fertilizationFactor)

Arguments

float fertilizationFactor Current weed factor to display, this will be converted to a percentage. Values < 0 clear this info.

setPlowingRequired

Description

Set the plowing required display.

Definition

setPlowingRequired(bool isRequired)

Arguments

bool isRequired If true, will display that the current field needs to be plowed. Otherwise, the info is hidden.

setLimeRequired

Description

Set the lime required display.

Definition

setLimeRequired(bool isRequired)

Arguments

bool isRequired If true, will display that the current field needs lime. Otherwise, the info is hidden.

addCustomText

Description

Add a custom text row.
The custom text will be added in order of calls to this function after the default information.

Definition

addCustomText(string leftText, string rightText, table leftColor)

Arguments

string leftText Text to be displayed on the left side in bold print

string rightText [optional] Text to be displayed on the right side in regular print

table leftColor [optional, default=<white>] Color of left text as an array {r, g, b, a}

Return Values

int Display row index of the newly added custom text or 0 if it could not be added

clearCustomText

Description

Clear a custom text row previously added by FieldInfoDisplay:addCustomText().

Definition

clearCustomText(int rowIndex)

Arguments

int rowIndex [optional] Custom text row index as returned by FieldInfoDisplay:addCustomText(). If no value is provided, all custom text is cleared.

clearInfoRow

Description

Clear a single info row's data.

Definition

clearInfoRow()

clearFieldData

Description

Clear all previously set field data.
This does not clear custom text rows, which need to be cleared specifically using clearCustomText().

Definition

clearFieldData()

onFieldDataUpdateFinished

Description

Called when FSDensityMapUtil.getFieldStatusAsync() in update() has finished.

Definition

onFieldDataUpdateFinished(table data)

Arguments

table data Field information data as provided by processing in FSDensityMapUtil.getFieldStatusAsync()

updateSize

Description

Update the info display size depending on used rows.

Definition

updateSize()

draw

Description

Draw the display.

Definition

draw()

drawText

Description

Draw text parts of this display element.

Definition

drawText()

getBackgroundPosition

Description

Get the scaled background position.

Definition

getBackgroundPosition()

setScale

Description

Set this element's UI scale factor.

Definition

setScale(float uiScale)

Arguments

float uiScale UI scale factor

storeScaledValues

Description

Store scaled position and size values.

Definition

storeScaledValues()

createBackground

Description

Create the background overlay.

Definition

createBackground()

createComponents

Description

Create required display components.

Definition

createComponents()

createFrame

Description

Create the background frame element.

Definition

createFrame()

createRowListContainer

Description

Create row list container.

Definition

createRowListContainer()

createSeparators

Description

Create row separators.

Definition

createSeparators()

FillLevelsDisplay

Description

Vehicle HUD fill levels display element.
-- Displays fill level bars for the current vehicle configuration
--@category GUI

new

Description

Creates a new FillLevelsDisplay instance.

Definition

new(string hudAtlasPath)

Arguments

string hudAtlasPath Path to the HUD texture atlas.

setVehicle

Description

Set the currently controlled vehicle which provides display data.

Definition

setVehicle(table vehicle)

Arguments

table vehicle Currently controlled vehicle

updateFillLevelBuffers

Description

Update fill levels data.

Definition

updateFillLevelBuffers()

updateFillLevelFrames

Description

Update fill level frames display state.

Definition

updateFillLevelFrames()

update

Description

Update the fill levels state.

Definition

update()

draw

Description

Draw this element.

Definition

draw()

setScale

Description

Set this element's scale.

Definition

setScale()

getBackgroundPosition

Description

Get the position of the background element, which provides this element's absolute position.

Definition

getBackgroundPosition(scale Current, float width)

Arguments

scale Current UI scale

float width Scaled background width in pixels

Return Values

float X position in screen space

float Y position in screen space

storeScaledValues

Description

Calculate and store scaling values based on the current UI scale.

Definition

storeScaledValues()

createBackground

Description

Create an empty background overlay as a base frame for this element.

Definition

createBackground()

refreshFillTypes

Description

Refresh fill type data and elements.

Definition

refreshFillTypes(table fillTypeManager)

Arguments

table fillTypeManager FillTypeManager reference

createFillTypeFrames

Description

Create fill type frames for all known fill types.

Definition

createFillTypeFrames()

createFillTypeFrame

Description

Create a fill type frame for the display of a fill type level state.

Definition

createFillTypeFrame()

createFillTypeIcon

Description

Create an icon for a fill type.

Definition

createFillTypeIcon()

createFillTypeBar

Description

Create a fill type bar used to display a fill level.
The newly created bar is added to the given parent frame and an internal collection indexable by fill type index.

Definition

createFillTypeBar(string hudAtlasPath, table frame, float baseX, float baseY, table fillType)

Arguments

string	hudAtlasPath	Path to HUD texture atlas
table	frame	Parent frame HUD element
float	baseX	Origin X position in screen space
float	baseY	Origin Y position in screen space

table	fillType	Fill type whose fill level is represented by the created bar

FocusManager

Description

The FocusManager controls which element in the menu system is currently focused and allows menu control with only keyboard or gamepad.
For each participating gui element the focus state and the next focused gui element in each direction is stored. This data is set up directly in the xml file of the gui screen and loaded through the loadElementFromXML() function or manually loaded within code by using
the loadElementFromCustomValues() method.
Focus handling is independent for every screen in the GUI. To swap screens the setGui() method
has to be used.
The focus system is then controlled with 5 actions: MENU_UP, MENU_DOWN, MENU_RIGHT and MENU_LEFT to
change the currently focused element in the specified direction and MENU_ACCEPT to activate
the currently focused element.
-- When using dynamically changing objects which cannot be set directly in the XML file of the screen
the method createLinkageSystemForElements() can be used to set up direction links between the
passed elements automatically.
--@category GUI

setGui

Description

Set the active GUI for focus input.

Definition

setGui(Gui Screen)

Arguments

Gui Screen root GuiElement

getElementById

Description

Get a focusable GuiElement in the current view by its ID.

Definition

getElementById()

getFocusedElement

Description

Get the currently focused GuiElement

Definition

getFocusedElement()

serveAutoFocusId

Description

Get a new automatic focus ID.
It's based on a simple integer increment and will be unique unless billions of elements require an ID.

Definition

serveAutoFocusId()

loadElementFromXML

Description

Load GuiElement focus data from its XML definition.
This is called at the end of GuiElement:loadFromXML().

Definition

loadElementFromXML()

loadElementFromCustomValues

Description

Add an element to the focus system with custom values.
The caller should ensure that explicitly set focus IDs are unique. If a duplicate ID is encountered, only the first
element with that focus ID is considered for focusing. The method returns a boolean value to indicate any problems
with data assignment. Callers can evaluate the value to check if the given parameters were valid. If in doubt or
when no elaborate focus navigation is needed, rely on automatic focus ID generation by omitting the ID parameter
(or set it to nil).

Definition

loadElementFromCustomValues(element Element, focusId Focus, focusChangeData Custom, focusActive If, isAlwaysFocusedOnOpen If)

Arguments

element	Element	to add to focus system
focusId	Focus	ID for element
focusChangeData	Custom	focus navigation data for the element (map of direction to focus ID)
focusActive	If	true, the element should be focused right now
isAlwaysFocusedOnOpen	If	true, the element is supposed to be focused when its parent view is opened.

Return Values

True if the element and all of its children could be set up with the given values, false otherwise.

removeElement

Description

Remove a GuiElement from the current focus context.

Definition

removeElement()

linkElements

Description

Links an element's focus navigation to another element for a given direction.
The link is unidirectional from source to target. If bi-directional links are desired, call this method again with
swapped arguments.

Definition

linkElements(sourceElement Source, direction Navigation, targetElement Target)

Arguments

sourceElement	Source	element which receives the focus link.
direction	Navigation	direction for the link, is not required to be the actual visual direction.
targetElement	Target	element

inputEvent

Description

Handles input and changes focus if required and possible.

Definition

inputEvent(action Name, value Input, eventUsed Usage)

Arguments

action	Name	of navigation action which triggered the event, see InputAction
value	Input	value [-1, 1]
eventUsed	Usage	flag, no action is taken if this is true

Return Values

True if the input event has been consumed, false otherwise

getDirectionForAxisValue

Description

Get a direction value for a given menu input action and value

Definition

getDirectionForAxisValue()

isFocusInputLocked

Description

Checks if the focus manager has an input lock on input.

Definition

isFocusInputLocked(inputAxis InputAction, value Axis, True if)

Arguments

inputAxis	InputAction	axis or action code
value	Axis	value [-1, 1] or nil if not a directional axis
True	if	locked, false otherwise

lockFocusInput

Description

Locks a given input axis action's input for a time. Until the delay has passed, the focus manager will not react to
that input.

Definition

lockFocusInput(inputAxis InputAction, delay Delay, value Axis)

Arguments

inputAxis	InputAction	axis or action code
delay	Delay	in ms
value	Axis	value [-1, 1], only relevant to identify directional axes

releaseMovementFocusInput

Description

Release a focus movement input lock on an action.
Called by the UI input handling code. Avoid calling this for anything else.

Definition

releaseMovementFocusInput(action Focus)

Arguments

action Focus movement input action name

resetFocusInputLocks

Description

Reset all locks of focus input.

Definition

resetFocusInputLocks()

getClosestPointOnBoundingBox

Description

Given a point and bounding box, get the closest other point on the bounding box circumference. If the point lies
within the bounding box, it is returned unchanged.

Definition

getClosestPointOnBoundingBox(x Point, y Point, boxMinX Bounding, boxMinY Bounding, boxMaxX Bounding, boxMaxY Bounding)

Arguments

x	Point	X
y	Point	Y
boxMinX	Bounding box minimum point X	
boxMinY	Bounding box minimum point Y	
boxMaxX	Bounding box maximum point X	
boxMaxY	Bounding box maximum point Y	

Return Values

Closest point x, y

getShortestBoundingBoxVector

Description

Calculate the shortest connecting line segment between two bounding boxes. Overlapping boxes will result in flipped
directions, so take care.

Definition

getShortestBoundingBoxVector()

checkElementDistance

Description

Checks the distance between two GuiElements with the aim of incrementally finding the closest other element in a
direction within a screen view.

Definition

checkElementDistance(curElement Current, other Other, dirX Scan, dirY Scan, curElementOffsetY Position, closestOther Previously, closestDistanceSq Squared)

Arguments

curElement	Current	checking GuiElement
other	Other	GuiElement to compare

dirX	Scan	direction vector x component, normalized to unit length
dirY	Scan	direction vector y component, normalized to unit length
curElementOffsetY	Position	y offset of current element's bounding volume, used when checking for wrap-around
closestOther	Previously	closest other GuiElement
closestDistanceSq	Squared	distance from the current checking element to the previously closest other GuiElement

getNextFocusElement

Description

Find the next other element to the one provided in a given navigation direction

Definition

getNextFocusElement(element GUI, direction Direction)

Arguments

element	GUI	element which needs a focus link
direction	Direction	constant [TOP | BOTTOM | LEFT | RIGHT]

Return Values

Next GUI	element in given direction which can be linked, actual scanning direction used (may change in wrap around scenarios)

getNestedFocusTarget

Description

Get an element's focus target at the deepest nesting depth, e.g. when multiple nested layouts point down to their
child elements until only a single element is left which points to itself.

Definition

getNestedFocusTarget(element GuiElement, direction Focus)

Arguments

element	GuiElement	whose focus target needs to be retrieved
direction	Focus	navigation direction

Return Values

Focus

updateFocus

Description

Update the current focus target.

Definition

updateFocus(element GuiElement, isFocusMoving Only, direction Focus, updateOnly If)

Arguments

element	GuiElement	which should be the new focus target
isFocusMoving	Only	move focus if this is true
direction	Focus	navigation movement direction, one of FocusManager.[TOP | BOTTOM | LEFT | RIGHT]
updateOnly	If	true, only updates the lock state of focus movement for the given parameters

setHighlight

Description

Activate a highlight on an element. Highlighted elements are only visually marked and do not receive focus activation.
Only one element will be highlighted at any time, usually corresponding to the current mouse over target.

Definition

setHighlight(element Element)

Arguments

element Element to be highlighted.

unsetHighlight

Description

Remove highlight status from an element.

Definition

unsetHighlight(element Highlighted)

Arguments

element Highlighted element to revert

setFocus

Description

Set focus on a GuiElement or its focus target.
Applies overlay state and triggers onFocusEnter() on the target.

Definition

setFocus(element Element, direction Focus, ... Variable)

Arguments

element Element whose focus target (usually itself) receives focus.
direction Focus navigation direction
... Variable arguments to pass on to the onFocusEnter callback of the target element

Return Values

True if focus has changed, false otherwise

unsetFocus

Description

Removes focus from an element.
Applies overlay state and triggers onFocusLeave() on the target.

Definition

unsetFocus(element Element, ... Variable)

Arguments

element Element which should lose focus
... Variable arguments to pass on to the onFocusLeave callback of the target element

setElementFocusOverlayState

Description

Set an elements focus overlay state for displaying.

Definition

setElementFocusOverlayState(element Target, isFocus If, handlePreviousState [optional])

Arguments

element Target element

isFocus	If	true, the element's state will be set to focused. Otherwise, it's state will be either restored or set to normal.
handlePreviousState	[optional]	If true or undefined, makes the element store its previous overlay state before modification or restore it when isFocused is false.

requireLock

Description

Globally lock focus input.

Definition

requireLock()

releaseLock

Description

Release the global focus input lock.

Definition

releaseLock()

isLocked

Description

Check if focus input is locked.

Definition

isLocked()

isDirectionLocked

Description

Determine if focus navigation in a given direction is currently locked.

Definition

isDirectionLocked(direction Navigation)

Arguments

direction Navigation direction as defined in constants

Return Values

True if navigation in given direction is locked

hasFocus

Description

Determine if a GuiElement is currently focused.

Definition

hasFocus()

getFocusOverrideFunction

Description

Get a closure override function for elements' getFocusOverride() methods.

Definition

getFocusOverrideFunction(forDirections List, substitute Element, useSubstituteForFocus (Optional))

Arguments

forDirections	List	of directions to override
substitute	Element	to substitute as focus target in overridden direction

useSubstituteForFocus (Optional)	If true, the substitute parameter will be used as the origin for finding the next focus target in the overridden direction.

FrameElement
Description

Base display frame element. All GUI views (partial and full screen) inherit from this.
-- This element provides the functionality to register control IDs, which are then exposed as fields of the concrete
descendant class (e.g. MainScreen or PasswordDialog). The control IDs must be assigned verbatim to any control
in the corresponding configuration XML file. If a registered ID is not used, the field will not be assigned and
access will fail. Available field IDs are documented as field properties per class. When creating a new view, take
care to include the call to registerControls() in the constructor to declare and expose control elements as fields.
--@category GUI

clone

Description

Override of GuiElement:clone().
Also exposes registered control element fields.

Definition

clone()

copyAttributes

Description

Override of GuiElement:copyAttributes().
Also resets registered control IDs so they can be exposed as fields again.

Definition

copyAttributes()

getRootElement

Description

Get the frame's root GuiElement instance.
This is the first and only direct child of a FrameElement, as defined by the GUI instantiation logic. This method
will always return a GuiElement instance, even if a new one must be created first.

Definition

getRootElement()

registerControls

Description

Register a collection of control IDs for direct access in GUI views.

Definition

registerControls(controlIDs Table)

Arguments

controlIDs Table	which holds control IDs as values, as they are required to be present in the view configuration.

exposeControlsAsFields

Description

Adds registered controls as fields to this FrameElement instance.
Called by the GUI system after loading.
The new fields will have the same name as the registered ID, so make sure there are no collision to avoid overrides
and that IDs are also valid as identifiers in Lua. If a control has been registered but no corresponding element is
available (e.g. when sub-classing and omitting some elements), the field will remain undefined. It's up to callers
to ensure that field configuration and usage in views matches.

Definition

exposeControlsAsFields(viewName View)

Arguments

viewName View name of this frame element

disableInputForDuration

Description

Set input disabling to a given duration.

Definition

disableInputForDuration(float duration)

Arguments

float duration Input disabling duration in milliseconds

isInputDisabled

Description

Check if input is currently disabled.

Definition

isInputDisabled()

setChangeScreenCallback

Description

Set a callback for requesting a view change from within a frame or screen view.

Definition

setChangeScreenCallback(func callback)

Arguments

func callback	Function reference, signature: function(sourceFrameElement, targetScreenClass, returnScreenClass)

setInputContextCallback

Description

Set a callback function for requesting a custom menu input context for this frame.

Definition

setInputContextCallback(func callback)

Arguments

func callback Function reference, signature: function(isContextActive)

setPlaySampleCallback

Description

Set a callback function for requesting to play a sound sample.

Definition

setPlaySampleCallback(func callback)

Arguments

func callback Function reference, signature: function(sampleName)

changeScreen

Description

Request a view change via the callback defined by setChangeScreenCallback().

Definition

changeScreen(table targetScreenClass)

Arguments

table targetScreenClass Class table of requested view (ScreenElement descendant, must be full view)

toggleCustomInputContext

Description

Request toggling of a custom menu input context for this frame via the callback defined by setInputContextCallback().

Definition

toggleCustomInputContext(bool isContextActive, string contextName)

Arguments

bool	isContextActive	If true, will activate a custom menu input context. Otherwise, will clear a previously activated context.
string	contextName	Name of the custom input context. Use a unique identifier value.

playSample

Description

Request playing a sound sample identified by name.

Definition

playSample(string sampleName)

Arguments

string sampleName Sample name, use one of GuiSoundPlayer.SOUND_SAMPLES

FSBaseMission

Description

(Un)loading stations

initialize

Description

Initialize mission after instantiation.
Create complex members and call dependency methods in here so that mission instantiation cannot fail.

Definition

initialize()

setHarvestScaleRatio

Description

Sets harvest ratios for fertilizer, plow, lime and weed factor

Definition

setHarvestScaleRatio(float sprayRatio, float plowRatio, float limeRatio, float weedRatio)

Arguments

float sprayRatio fertilizer ratio
float plowRatio plow ratio
float limeRatio lime ratio
float weedRatio weed ratio

getHarvestScaleMultiplier

Description

Get harvest multiplier based on fertilizer, plow, lime and weed factor, 1 = best

Definition

getHarvestScaleMultiplier(integer fruitTypeIndex, float sprayFactor, float plowFactor, float limeFactor, float weedFactor)

Arguments

integer fruitTypeIndex fruit type index
float sprayFactor fertilizer factor
float plowFactor plow factor
float limeFactor lime factor
float weedFactor weed factor

Return Values

float multiplier harvest multiplier

updateMenuAccessibleVehicles

Description

Check which vehicles are accessible and notify the menu.

Definition

updateMenuAccessibleVehicles()

addOwnedItem

Description

Add an item to the owned items collection.
Also notifies the shop.

Definition

addOwnedItem()

removeOwnedItem

Description

Remove an item from the owned items collection.
Also notifies the shop.

Definition

removeOwnedItem()

addLeasedItem

Description

Add an item to the leased items collection.
Also notifies the shop.

Definition

addLeasedItem()

removeLeasedItem

Description

Remove an item from the leased items collection.
Also notifies the shop.

Definition

removeLeasedItem()

updatePauseInputContext

Description

Set or revert the pause input context based on current state.

Definition

updatePauseInputContext()

updateSaving

Description

Update saving process.
Notifies the UI to display dialogs.

Definition

updateSaving()

onYesNoSavegameSelectDevice

Description

Savegame device selection confirmation dialog response callback.

Definition

onYesNoSavegameSelectDevice()

onSaveGameUpdateComplete

Description

Savegame overwrite completion callback.

Definition

onSaveGameUpdateComplete(int errorCode)

Arguments

int errorCode Saving process status code, one of Savegame.ERROR_...

onYesNoSavegameOverwrite

Description

Savegame overwrite confirmation dialog response callback.

Definition

onYesNoSavegameOverwrite()

showAttachContext

Description

Display vehicle attachment context information for the current frame.

Definition

showAttachContext()

showTipContext

Description

Display vehicle tipping context information for the current frame.

Definition

showTipContext()

showFuelContext

Description

Display vehicle refueling context information for the current frame.

Definition

showFuelContext()

showFillDogBowlContext

Description

Display dog companion food bowl fill context information for the current frame.

Definition

showFillDogBowlContext()

getMoney

Description

Get money for farm or current player farm

Definition

getMoney(farmId integer)

Arguments

farmId integer if nil, then current farm (client only)

getHasPlayerPermission

Description

Get whether the current player has given permission.

Definition

getHasPlayerPermission(string permission, table connection, integer farmId, boolean checkClient)

Arguments

string	permission	Permission to check
table	connection	[optional] Client connection to check permission for
integer	farmId	[optional] Limit permission to given farm (otherwise any farm)
boolean	checkClient	[optional] if true it will only check the client permission

getIngameMap

Description

Get the in-game map rendering component.
Should only be used for UI.

Definition

getIngameMap()

startSaveCurrentGame

Description

Start saving the game.

Definition

startSaveCurrentGame(bool isDediSaving)

Arguments

bool isDediSaving If true, this is a saving call on a dedicated server

saveSavegame

Description

Actually save the game locally.

Definition

saveSavegame()

getDoghouse

Description

Returns a doghouse

Definition

getDoghouse(integer farmId)

Arguments

integer farmId

Return Values

table instance of the doghouse

Code

```
3144  function FSBaseMission:getDoghouse(farmId)
3145  for _, doghouse in pairs(self.doghouses) do
3146  if doghouse:getOwnerFarmId() == farmId then
3147  return doghouse
3148  end
3149  end
3150  return nil
3151  end
```

registerHusbandry

Description

Register a husbandry in the mission.

Definition

registerHusbandry()

unregisterHusbandry

Description

Unregister a husbandry in the mission.

Definition

unregisterHusbandry()

getVehicleName

Description

Extract a concrete vehicle's display name from its store data.

Definition

getVehicleName(table vehicle)

Arguments

table vehicle Vehicle instance

Return Values

string Vehicle display name

registerActionEvents
Description

Register required input action events.

Definition

registerActionEvents()

registerPauseActionEvents
Description

Register action events for pause actions.
Event registration in this method is enclosed in an input binding registration context (InputBinding:beginActionEventsModification()).
Make sure that this is only called when all other registration-context altering code is done.

Definition

registerPauseActionEvents()

onToggleMenu
Description

Input event for "open menu".

Definition

onToggleMenu()

onToggleStore
Description

Input event for "open shop".

Definition

onToggleStore()

toggleChat
Description

Toggle chat display.
Called from input to open the chat dialog and from the chat dialog to close it again.

Definition

toggleChat()

onToggleRadio
Description

Input event for "toggle radio".

Definition

onToggleRadio()

onChangeTimescale
Description

Input event for "increase timescale" and "decrease timescale"

Definition

onChangeTimescale()

onShowHelpIconsChanged
Description

Handle a settings change for showing help icons.

Definition

onShowHelpIconsChanged()

onRadioVehicleOnlyChanged

Description

Handle a settings change for the radio vehicle only setting.
If the current radio state is active, the radio will start playing if the context fits (e.g. turn to global, and
player is on foot).

Definition

onRadioVehicleOnlyChanged()

onRadioIsActiveChanged

Description

Handle a settings change for the radio active state.

Definition

onRadioIsActiveChanged()

setRadioActionEventsState

Description

Set the active state for the radio action events.

Definition

setRadioActionEventsState()

subscribeMessages

Description

Subscribe to FS-specific event messages.
The BaseMission parent class unsubscribes from messages in delete(). No clean-up required in sub-classes like this.

Definition

subscribeMessages()

notifyPlayerFarmChanged

Description

Handle a local client changing the farm.

Definition

notifyPlayerFarmChanged()

GameInfoDisplay

Description

HUD general game information display.
-- Displays current game information. This includes weather, current account balance and time settings.
--@category GUI

new

Description

Create a new instance of GameInfoDisplay.

Definition

new(string hudAtlasPath)

Arguments

string hudAtlasPath Path to the HUD texture atlas

setMoneyUnit

Description

Set the money unit for displaying the account balance.

Definition

setMoneyUnit(int moneyUnit)

Arguments

int moneyUnit Money unit ID, any of [GS_MONEY_EURO | GS_MONEY_POUND | GS_MONEY_DOLLAR]. Invalid values are substituted by GS_MONEY_DOLLAR.

setMissionStats

Description

Set the MissionStats reference for displaying information.

Definition

setMissionStats(table missionStats)

Arguments

table missionStats MissionStats reference, do not change

setMissionInfo

Description

Set the mission information reference for base information display.

Definition

setMissionInfo(table missionInfo)

Arguments

table missionInfo MissionInfo reference, do not change

setEnvironment

Description

Set the environment reference to use for weather information display.

Definition

setEnvironment(table environment)

Arguments

table environment Environment reference, do not change

setMoneyVisible

Description

Set visibility of the money display.

Definition

setMoneyVisible()

setTimeVisible

Description

Set visibility of time display.

Definition

setTimeVisible()

setTemperatureVisible

Description

Set visibility of temperature display.

Definition

setTemperatureVisible()

setWeatherVisible

Description

Set visibility of weather display.

Definition

setWeatherVisible()

setTutorialVisible

Description

Set visibility of tutorial progress display.

Definition

setTutorialVisible()

setTutorialProgress

Description

Set the current tutorial progress values.

Definition

setTutorialProgress(float progress)

Arguments

float progress Progress expressed as a number between 0 and 1

update

Description

Update the game info display state.

Definition

update()

updateTime

Description

Update time display.

Definition

updateTime()

updateTemperature

Description

Update temperature display.

Definition

updateTemperature()

updateWeather

Description

Update weather display

Definition

updateWeather()

getVisibleWidth
Description

Get the game info display's width based on its visible info boxes.

Definition

getVisibleWidth()

Return Values

float Game info display width of visible elements in screen space

updateSizeAndPositions
Description

Update sizes and positions of this elements and its children.

Definition

updateSizeAndPositions()

draw
Description

Draw the game info display.

Definition

draw()

drawMoneyText
Description

Draw the text part of the money display.

Definition

drawMoneyText()

drawTimeText
Description

Draw the text part of the time display.

Definition

drawTimeText()

drawTemperatureText
Description

Draw the text part of the temperature display.

Definition

drawTemperatureText()

drawTutorialText
Description

Draw the text part of the tutorial progress display.

Definition

drawTutorialText()

animateWeatherChange
Description

Make an animation for a weather change.

Definition

animateWeatherChange()

addActiveWeatherAnimation

Description

Animate a weather icon becoming active.

Definition

addActiveWeatherAnimation()

addInactiveWeatherAnimation

Description

Animate a weather icon becoming inactive.

Definition

addInactiveWeatherAnimation()

addBecomeCurrentWeatherAnimation

Description

Animate a weather icon becoming the current weather icon.

Definition

addBecomeCurrentWeatherAnimation()

addWeatherPositionAnimation

Description

Animate weather icon position changes.

Definition

addWeatherPositionAnimation()

getBackgroundPosition

Description

Get this element's base background position.

Definition

getBackgroundPosition(float uiScale)

Arguments

float uiScale Current UI scale factor

setScale

Description

Set this element's UI scale factor.

Definition

setScale(float uiScale)

Arguments

float uiScale UI scale factor

storeScaledValues

Description

Store scaled positioning, size and offset values.

Definition

storeScaledValues()

createBackground

Description

Create the background overlay.

Definition

createBackground()

createComponents

Description

Create required display components.
Also adds a separator HUDElement instance as a field (".separator") to all info boxes.

Definition

createComponents()

createMoneyBox

Description

Create the money display box.

Definition

createMoneyBox()

createTimeBox

Description

Create the time display box.

Definition

createTimeBox()

createTemperatureBox

Description

Create the temperature display box.

Definition

createTemperatureBox()

createWeatherBox

Description

Create the weather display box.

Definition

createWeatherBox()

createWeatherIcon

Description

Create a weather icon for current and upcoming weather conditions.

Definition

createWeatherIcon(string hudAtlasPath, int weatherId, float boxHeight, table uvs, table color)

Arguments

string	hudAtlasPath	Path to HUD texture atlas
int	weatherId	Weather condition ID, as defined in WeatherType... constants
float	boxHeight	Screen space height of the box which will hold this icon
table	uvs	UV coordinates of the weather icon in the HUD texture atlas
table	color	Color RGBA array

Return Values

table Weather icon HUDElement instance

createTemperatureIcon

Description

Create a temperature icon to display stable or changing temperatures.

Definition

createTemperatureIcon(string hudAtlasPath, float leftX, float bottomY, float boxHeight, table uvs, table color)

Arguments

string hudAtlasPath Path to HUD texture atlas

float	leftX	Screen space left X position of newly created icon
float	bottomY	Screen space bottom Y position of the parent box
float	boxHeight	Screen space height of the parent box
table	uvs	UV coordinates of the icon in the HUD texture atlas
table	color	Color RGBA array

Return Values

table Temperature icon HUDElement instance

createClockHand

Description

Create a rotatable clock hand icon element.

Definition

createClockHand(string hudAtlasPath, float posX, float posY, table size, table uvs, table color, table pivot)

Arguments

string hudAtlasPath Path to HUD texture atlas

float	posX	Screen space X position of the clock hand
float	posY	Screen space Y position of the clock hand
table	size	Pixel size vector {width, height}
table	uvs	UV coordinates of the icon in the HUD texture atlas
table	color	Color RGBA array
table	pivot	UV pixel space rotation pivot coordinates

Return Values

table Clock hand HUDElement instance

createTimeScaleArrow

Description

Create a time scale arrow icon element.

Definition

createTimeScaleArrow(string hudAtlasPath, float posX, float posY, table size, table uvs)

Arguments

string hudAtlasPath Path to HUD texture atlas

float	posX	Screen space X position of the arrow
float	posY	Screen space Y position of the arrow
table	size	Pixel size vector {width, height}
table	uvs	UV coordinates of the icon in the HUD texture atlas

Return Values

table Time scale arrow icon HUDElement instance

createVerticalSeparator

Description

Create and return a vertical separator element.

Definition

createVerticalSeparator()

createTutorialBox

Description

Create the tutorial progress box.

Definition

createTutorialBox()

GamepadSigninScreen

Description

Sign-In Screen before Main Menu.
-- Used in console version.
-- @field startText Button prompt text to start the game.

inputEvent

Description

GUI input event callback.
See GuiElement:inputEvent().

Definition

inputEvent()

signIn

Description

Event function for button sign in.

Definition

signIn()

GamePausedDisplay

Description

HUD game pause display element.
-- Displays a customizable message when the game is paused.
--@category GUI

new

Description

Create a new GamePausedDisplay.

Definition

new(string hudAtlasPath)

Arguments

string hudAtlasPath Path to the HUD atlas texture.

Return Values

table GamePausedDisplay instance

setPauseText

Description

Set a custom text to display.

Definition

setPauseText()

onMenuVisibilityChange

Description

Handle menu visibility state change.

Definition

onMenuVisibilityChange()

setScale

Description

Set uniform UI scale.

Definition

setScale()

storeScaledValues

Description

Store scaled positioning, size and offset values.

Definition

storeScaledValues()

createBackground

Description

Get this element's base background position.

Definition

createBackground()

createComponents

Description

Create required display components.

Definition

createComponents()

Gui

Description

Graphical User Interface controller.
-- Builds UI from configurations, provides dialog display and propagates UI input.
--@category GUI

loadProfiles

Description

Load UI profile data from XML.

Definition

loadProfiles(xmlFilename UI)

Arguments

xmlFilename UI profiles definition XML file path, relative to application root.

loadGui

Description

Load a UI screen view's elements from an XML definition.

Definition

loadGui(xmlFilename View, name Screen, controller FrameElement, isFrame [optional,)

Arguments

xmlFilename	View	definition XML file path, relative to application root.
name	Screen	name
controller	FrameElement	instance which serves as the controller for loaded elements
isFrame	[optional,	default=false] If true, will interpret the loaded view as a frame to be used in multiple places.

Return Values

Root GuiElement instance of loaded view or nil if the definition XML file could not be loaded.

loadGuiRec

Description

Recursively load and build a UI configuration.

Definition

loadGuiRec(xmlFile Opened, xmlNodePath Current, parentGuiElement Current, target Target)

Arguments

xmlFile	Opened	GUI configuration XML file
xmlNodePath	Current	XML node path
parentGuiElement	Current	parent GuiElement
target	Target	of newly instantiated elements

resolveFrameReference

Description

Tries resolving a frame reference.
If no frame has been loaded with the name given by the reference, then the reference element itself is returned.
Otherwise, the registered frame is cloned and returned.

Definition

resolveFrameReference(self Gui, frameRefElement FrameReferenceElement)

Arguments

self	Gui	instance
frameRefElement	FrameReferenceElement	instance to resolve

Return Values

Cloned FrameElement instance or frameRefElement if resolution failed.

getProfile

Description

Get a UI profile by name.

Definition

getProfile()

getIsGuiVisible

Description

Determine if any menu or dialog is visible.

Definition

getIsGuiVisible()

getIsDialogVisible

Description

Determine if any dialog is visible.

Definition

getIsDialogVisible()

getActionEventId

Description

Get a menu action event ID by its name.
Modify GUI action events with care, as they are globally defined for an entire session.

Definition

getActionEventId()

showGui

Description

Display and return a screen identified by name.

Definition

showGui()

Return Values

Root GuiElement of screen or nil if the name did not match any known screen.

showDialog

Description

Display a dialog identified by name.

Definition

showDialog()

Return Values

Root GuiElement of dialog or nil if the name did not match any known dialog.

closeDialogByName

Description

Close a dialog identified by name.

Definition

closeDialogByName()

closeDialog

Description

Close a dialog identified by its root GuiElement.
This is always called when a dialog is closed, usually by itself.

Definition

closeDialog()

closeAllDialogs

Description

Close all open dialogs.

Definition

closeAllDialogs()

registerMenuInput

Description

Register menu input.

Definition

registerMenuInput()

mouseEvent

Description

GUI mouse event hook.
This is used primarily for mouse location checks, as button inputs are handled by InputBinding.

Definition

mouseEvent()

keyEvent

Description

GUI key event hook.
This is used for GuiElements which need to have direct access to raw key input, such as TextInputElement.

Definition

keyEvent()

update

Description

Update the GUI.
Propagates update to all active UI elements.

Definition

update()

draw

Description

Draw the GUI.
Propagates draw calls to all active UI elements.

Definition

draw()

notifyControls

Description

Notify controls of an action input with a value.

Definition

notifyControls(action Action, value Action)

Arguments

action Action name as defined by loaded actions, see also scripts/input/InputAction.lua

value Action value [-1, 1]

Return Values

True if any control has consumed the action event

onMenuInput

Description

Event callback for menu input.

Definition

onMenuInput()

onReleaseMovement

Description

Event callback for released movement menu input.

Definition

onReleaseMovement()

hasElementInputFocus

Description

Determine if a given GuiElement has input focus.

Definition

hasElementInputFocus()

getScreenInstanceByClass

Description

Get a screen controller instance by class for special cases.

Definition

getScreenInstanceByClass(table screenClass)

Arguments

table screenClass Class table of the requested screen

Return Values

table ScreenElement descendant instance of the given class or nil if no such instance was registered

changeScreen

Description

Change the currently displayed screen.

Definition

changeScreen(table source, table screenClass, table returnScreenClass)

Arguments

table source	Source screen instance
table screenClass	Class table of the requested screen to change to or nil to close the GUI
table returnScreenClass	[optional] Class table of the screen which will be opened on a "back" action in the new screen.

Return Values

table Root GuiElement instance of target screen

makeChangeScreenClosure

Description

Make a change screen callback function which encloses the Gui self reference.
This avoids passing the reference around as a parameter or global reference.

Definition

makeChangeScreenClosure()

toggleCustomInputContext

Description

Toggle a custom menu input context for one of the managed frames or screens.

Definition

toggleCustomInputContext()

makeToggleCustomInputContextClosure

Description

Make a toggle custom input context function which encloses the Gui self reference.

Definition

makeToggleCustomInputContextClosure()

makePlaySampleClosure

Description

Make a play sample function which encloses the Gui self reference.

Definition

makePlaySampleClosure()

assignPlaySampleCallback

Description

Assign the play sample closure to a GUI element which has the PlaySampleMixin included.

Definition

assignPlaySampleCallback(table guiElement)

Arguments

table guiElement GuiElement instance

Return Values

table The GuiElement instance given in the guiElement parameter

enterMenuContext

Description

Enter a new menu input context.
Menu views which require special input should provide a context name to not collide with the base menu input scheme.

Definition

enterMenuContext(string contextName)

Arguments

string contextName [optional] Custom menu input context name

leaveMenuContext

Description

Leave a menu input context.
This wraps the input context setting to check if the menu is actually active and the context should be reverted to
the state before entering the menu (or a custom menu input context within the menu).

Definition

leaveMenuContext()

addFrame

Description

Add the instance of a specific reusable GUI frame.

Definition

addFrame()

addScreen

Description

Add the instance of a specific GUI screen.

Definition

addScreen()

setCurrentMission

Description

Set the current mission reference for GUI screens.

Definition

setCurrentMission()

setEconomyManager

Description

Set the current economy manager for GUI screens.
The manager is initialized during mission loading.

Definition

setEconomyManager()

loadMapData

Description

Let the GUI (and its components) process map data when it's loaded.

Definition

loadMapData(int mapXmlFile)

Arguments

int mapXmlFile Map configuration XML file handle, do not close, will be handled by caller.

setClient

Description

Set the network client reference for GUI screens.

Definition

setClient()

setIsMultiplayer

Description

Set multiplayer state on screens which need to know about it.

Definition

setIsMultiplayer(bool isMultiplayer)

Arguments

bool isMultiplayer If true, the GUI needs work in multiplayer mode. Otherwise, it's single player.

initGuiLibrary

Description

Source in UI modules.

Definition

initGuiLibrary(baseDir Base)

Arguments

baseDir Base scripts directory

GuiElement

Description

GUI Element base class.
All elements displayed in the game UI must be instances or descendants of this class.
-- All XML configuration properties as declared below (and in subclasses) are mirrored in guiProfiles.xml as key-value
pairs in the form of &ltValue name="property_name" value="value" /&gt. Profiles are able to inherit from other
profiles, so take care to check their hierarchy if any of your settings do not seem to have any effect. Directly set
properties in the XML configuration will always override profile values, however.
-- Layer properties, prefixed with "[layer]", interact with an overlay system and provide display images. Usable
layers, whose names are substituted for the prefix, are primarily "image" and "icon". UI elements define layer names
on their own and read them from these generated properties. Whenever an element requires a layer, it is described
in its documentation such as this one. Example for an icon layer focus color property: iconFocusedColor="0.9 0.1 0.5 1.0".
-- A note regarding callbacks: All callbacks are called on an element's target first. When GUI elements are created from
configuration, their top-level view (e.g. MainScreen) is the callback target, i.e. MainScreen:callbackName() is
executed. Unless an element's target has been set to another value explicitly via code, this will always be the case.
--@category GUI
--@xmlConfig GuiElement#id string [optional] Element ID. Must be unique per GUI screen. If set on an element, will expose it as an indexable field on the view class (e.g. id="buttonOk" -> MainScreen.buttonOk). Use this feature sparingly, as it creates implicit coupling which leads to tedious debugging.

new

Description

Create a new GuiElement.

Definition

new(target Target)

Arguments

target Target ScreenElement instance

loadFromXML

Description

Load element data from an XML definition file.

Definition

loadFromXML(xmlFile Definition, key XML)

Arguments

xmlFile	Definition	XML file handle
key	XML	node path to this element's definition

loadProfile

Description

Load profile data for this element.

Definition

loadProfile(profile Loaded, applyProfile If)

Arguments

profile	Loaded GUI profile
applyProfile	If true, will re-calculate some dynamic properties. Use this when setting profiles dynamically at run time.

applyProfile

Description

Apply a GUI profile with a given name to this element.

Definition

applyProfile(profileName Name, bool force)

Arguments

profileName	Name	of the profile to apply to this element
bool	force	[optional] If true, will apply all profile settings, including position and size.

deleteFrame

Description

Delete any created frame overlays.

Definition

deleteFrame()

delete

Description

Delete this GuiElement.
Also deletes all child elements and removes itself from its parent and focus.

Definition

delete()

clone

Description

Create a deep copy clone of this GuiElement.

Definition

clone(parent Target, includeId [optional,, suppressOnCreate [optional,)

Arguments

parent	Target	parent element of the cloned element
includeId	[optional, default=false]	If true, will also clone ID values
suppressOnCreate	[optional, default=false]	If true, will not trigger the "onCreate" callback

copyAttributes

Description

Copy all attributes from a source GuiElement to this GuiElement.

Definition

copyAttributes()

onGuiSetupFinished

Description

Called on a screen view's root GuiElement when all elements in a screen view have been created.
The event is propagated to all children, depth-first.

Definition

onGuiSetupFinished()

createFrame

Description

Create frame overlays if this element has a frame.

Definition

createFrame()

toggleFrameSide

Description

Toggle a frame side's visibility identified by index.
If this element has no frame this will have no effect.

Definition

toggleFrameSide(int sideIndex)

Arguments

int sideIndex Index of the frame side, use one of GuiElement.FRAME_...

updateFramePosition

Description

Update the frame overlay positions if necessary.

Definition

updateFramePosition()

cutFrameBordersHorizontal

Description

Cut horizontal frame borders if a vertical frame side is thicker.

Definition

cutFrameBordersHorizontal()

cutFrameBordersVertical

Description

Cut vertical frame borders if a horizontal frame side is thicker.

Definition

cutFrameBordersVertical()

mouseEvent

Description

Mouse event hook for mouse movement checks.

Definition

mouseEvent()

inputEvent

Description

Handles an input event on a menu input action.
Input is first passed to the current GUI view, then to the focused element, then to the focus manager for navigation.
When a GUI element receives input, it should always propagate the input to its parent element first so they may
override behavior and/or set the event used flag. If properly inherited from GuiElement and descendants, this
behavior is guaranteed.

Definition

inputEvent(action Name, value Input, eventUsed If)

Arguments

action	Name	of input action which was triggered.
value	Input	value in the range of [-1, 1] for full axes and [0, 1] for half axes (includes buttons)
eventUsed	If	true, the event has been used by an input handler and should only be acted upon in exceptional cases.

Return Values

True if the input event has been handled, false otherwise.

keyEvent

Description

Key event hook for raw keyboard input.

Definition

keyEvent()

Return Values

True if the keyboard input has been processed by this element.

update

Description

Update this GuiElement.

Definition

update()

draw

Description

Draw this GuiElement.
If defined, triggers the "onDrawCallback".

Definition

draw()

onOpen

Description

Called on the root element of a screen view when it is opened.
This raises the "onOpenCallback" if defined and propagates to all children.

Definition

onOpen()

onClose

Description

Called on the root element of a screen view when it is closed.
This raises the "onCloseCallback" if defined and propagates to all children.

Definition

onClose()

shouldFocusChange

Description

Determine if this GuiElement should change focus in a given direction.

Definition

shouldFocusChange()

canReceiveFocus

Description

Determine if this GuiElement can receive focus.

Definition

canReceiveFocus()

onFocusLeave

Description

Called when this element loses focus.
This propagates to all children.

Definition

onFocusLeave()

onFocusEnter

Description

Called when this element becomes focused.
This propagates to all children.

Definition

onFocusEnter()

onFocusActivate

Description

Called when this element has focus and the focus activation action is triggered.
This propagates to all children.

Definition

onFocusActivate()

onHighlight

Description

Called when this element is highlighted.
This propagates to all children.

Definition

onHighlight()

onHighlightRemove

Description

Called when this element loses the highlight.
This propagates to all children.

Definition

onHighlightRemove()

storeOverlayState

Description

Store the current overlay state while another overrides it (e.g. pressed state)

Definition

storeOverlayState()

restoreOverlayState

Description

Restore a previously stored overlay state after an overriding state has expired

Definition

restoreOverlayState()

getHandleFocus

Description

Determine if this element can receive focus.

Definition

getHandleFocus()

setHandleFocus

Description

Set this elements capability to receive focus.

Definition

setHandleFocus()

addElement

Description

Add a child GuiElement to this GuiElement.

Definition

addElement()

removeElement

Description

Remove a child GuiElement from this GuiElement.

Definition

removeElement()

unlinkElement

Description

Safely remove this GuiElement from its parent, if it has a parent.

Definition

unlinkElement()

updateAbsolutePosition

Description

Update this elements absolute screen position.
This needs to be called whenever a position, alignment, origin or size value changes.

Definition

updateAbsolutePosition()

reset

Description

Resets the state of this GuiElement and its children.

Definition

reset()

isChildOf

Description

Check if this element is the child of another element.
This checks the full parent hierarchy.

Definition

isChildOf()

getFocusTarget

Description

Get the actual focus target, in case a child or parent element needs to be targeted instead.

Definition

getFocusTarget(incomingDirection (Optional), moveDirection (Optional))

Arguments

incomingDirection	(Optional)	If specified, may return different targets for different incoming directions.
moveDirection	(Optional)	Actual movement direction per input. This is the opposing direction of incomingDirection.

Return Values

GuiElement	Actual element to focus.

setPosition

Description

Set this element's position.

Definition

setPosition()

move

Description

Modify this element's position (i.e. translate position).

Definition

move()

setAbsolutePosition

Description

Directly set the absolute screen position of this GuiElement.
Also updates children accordingly.

Definition

setAbsolutePosition()

setSize
Description

Set this element's size.

Definition

setSize()

setVisible
Description

Set this element's visibility.

Definition

setVisible()

getIsVisible
Description

Determine if this element is visible.
This checks both its current alpha value (set by fadeIn() / fadeOut()) as well as the visibility flag. If the parent
is invisible, then so is this element.

Definition

getIsVisible()

setDisabled
Description

Set this element's disabled state.
Disabled elements can be displayed differently and do not respond to input actions.

Definition

setDisabled(disabled If, doNotUpdateChildren If)

Arguments

disabled	If true, disables the element. False enables it again.
doNotUpdateChildren	If true, does not apply the disabled state to child elements.

getIsDisabled
Description

Determine if this element is disabled.

Definition

getIsDisabled()

getIsSelected
Description

Determine if this element is currently selected.

Definition

getIsSelected()

getIsHighlighted
Description

Determine if this element is currently highlighted.

Definition

getIsHighlighted()

fadeIn

Description

Fade this element into visibility.

Definition

fadeIn()

fadeOut

Description

Fade this element out of visibility.

Definition

fadeOut()

setAlpha

Description

Directly set this element's alpha (transparency) value

Definition

setAlpha(alpha Transparency)

Arguments

alpha Transparency value in the floating point range of [0, 1], where 0 is invisible and 1 is opaque.

getIsActive

Description

Determine if this element is active (not disabled and visible).
Does not take alpha value into account.

Definition

getIsActive()

setSoundSuppressed

Description

Toggle a flag to suppress UI sounds issued by this element or by the FocusManager when handling this element.
This setting will propagate to children.

Definition

setSoundSuppressed()

getSoundSuppressed

Description

Get the sound suppression flag from this element.
If the flag is set to true, no sounds should be played when interacting with this element.

Definition

getSoundSuppressed()

findDescendantsRec

Description

Recursively add descendant elements of a root to an accumulator list. If a predicate function is given, it is

evaluated per element and only elements which yield a true value for the function are added to the accumulator.

Definition

findDescendantsRec(accumulator List, rootElement Current, predicateFunction [optional])

Arguments

accumulator	List	which receives descendant elements
rootElement	Current	element root whose direction children are added (after optional evaluation)
predicateFunction	[optional]	If specified, will be evaluated per element (see getDescendants)

getDescendants

Description

Get all contained elements of this element in the entire hierarchy.
Descendants are traversed depth-first, meaning that if elements have been properly added, the element order mirrors
the order in the XML configuration (lines). Use this method sparingly, especially on high-level elements.
Optionally, a predicate function can be passed which filters descendant elements. The function must return true for
any desired element and false otherwise.

Definition

getDescendants(predicateFunction [optional])

Arguments

predicateFunction	[optional]	A function which determines if a descendant element should be returned. Must take a GuiElement as an argument and return true if that element should be returned or false otherwise.

Return Values

List of this element's descendants in depth-first order with contiguous numeric indices.

getFirstDescendant

Description

Get the first descendant element of this element which matches a predicate function.
This is a shorthand for getDescendants() which returns just the first element matching the predicate function or nil
if no matching element exists.

Definition

getFirstDescendant(predicateFunction A)

Arguments

predicateFunction	A	function which determines if a descendant element should be returned. Must take a GuiElement as an argument and return true if that element should be returned or false otherwise.

Return Values

First matching descendant element in depth-first order or nil, if no element matched the predicate function

getDescendantById

Description

Get a descendant element of this element by its ID.
This is a shorthand for getDescendants() with an ID matching predicate function.

Definition

getDescendantById(id Element)

Arguments

id Element id

Return Values

element or nil

getDescendantByName

Description

Get a descendant element of this element by its name.
This is a shorthand for getDescendants() with an ID matching predicate function.

Definition

getDescendantByName(name Element)

Arguments

name Element name

Return Values

element or nil

updatePositionForOrigin

Description

Get the bit mask value for a position origin string value.

Definition

updatePositionForOrigin()

updateScreenAlign

Description

Get the bit mask value for a screen alignment string value.

Definition

updateScreenAlign()

getParentBorders

Description

Get the bottom left and top right corners of this element's parent's border rectangle.
If this element has no parent, the full screen's borders are returned (i.e. {0, 0, 1, 1})

Definition

getParentBorders()

Return Values

parent element or full screen borders in an array: {minX, minY, maxX, maxY}

getBorders

Description

Get this element's border rectangle represented by minimum and maximum points.

Definition

getBorders()

Return Values

element borders in an array: {minX, minY, maxX, maxY}

getCenter

Description

Get this element's center position.

Definition

getCenter()

applyScreenAlignment

Description

Apply screen alignment to this element and its children.
Scales size, position and margin depending on alignment settings.

Definition

applyScreenAlignment()

setOverlayState

Description

Set this element's overlay state

Definition

setOverlayState(overlayState Overlay)

Arguments

overlayState Overlay state identified by one of the GuiOverlay.STATE_[...] constants.

getOverlayState

Description

Get this element's overlay state.

Definition

getOverlayState()

addCallback

Description

Add a callback to this element which was defined in its XML definition.
If this element has a target, the given function name will be called on the target. Otherwise, the function is
assumed to be global.

Definition

addCallback(xmlFile XML, key XML, funcName Name)

Arguments

xmlFile	XML	file handle
key	XML	node path of this GuiElement's definition.
funcName	Name	of the callback function

raiseCallback

Description

Raise a previously added callback by name.

Definition

raiseCallback()

extractIndexAndNameFromID

Description

Try to extract a field name and index from an element ID.
IDs in configurations may be indexed on definition (e.g. fillTypes[2]). This function extracts the list name and
index if such a case is found. Otherwise, it will return no index and the original element ID.

Definition

extractIndexAndNameFromID(elementId Element)

Arguments

elementId Element ID, to be used as a field name on a ScreenElement view.

Return Values

index or nil, field name

setId

Description

Try setting this element's ID from its XML definition.

Definition

setId()

include

Description

Include a mixin in this element.
See GuiMixin.lua for the details on usage and implementation of mixins.

Definition

include(guiMixinType Class)

Arguments

guiMixinType Class table reference of a descendant of GuiMixin

toString

Description

Get a nice string representation for this GUI element.

Definition

toString()

GuiMixin

Description

GuiElement mixin base class.
-- Implements base functionality for GUI element mixins. All other GUI mixins should be descendants of this class.
-- Added methods:
GuiElement:hasIncluded(mixinType) Test if the GuiElement has included a mixin of the given type (class).
--@category GUI

new

Description

Create a new GuiMixin instance.
Subclasses need to provide their class type table for identification.

Definition

new(class Class, mixinType Class)

Arguments

class Class metatable

mixinType Class type table

Return Values

New

addTo

Description

Add a mixin to a GuiElement.
Adds mixin methods to the element which can then be used. A mixin's state is located in "element[mixinType]".

Definition

addTo()

hasIncluded

Description

Determine if a GuiElement has a mixin type included.

Definition

hasIncluded(GuiElement GuiElement, mixinType GuiMixin)

Arguments

GuiElement	GuiElement	instance
mixinType	GuiMixin	class reference

cloneMixin

Description

Clone mixin states for a mixin type from a source to a destination GuiElement instance.

Definition

cloneMixin()

clone

Description

Clone this mixin's state from a source to a destination GuiElement instance.

Definition

clone()

GuiOverlay
Description

GUI overlay manager.
-- Handles creation, loading and basic rendering of GUI overlays. This module has no interaction with Overlay.lua.
--@category GUI

loadOverlay

Description

Loads overlay data from XML or a profile into a table to turn it into an overlay.

Definition

loadOverlay()

loadXMLUVs

Description

Load overlay UV data from XML.

Definition

loadXMLUVs()

loadProfileUVs

Description

Load overlay UV data from a profile.

Definition

loadProfileUVs()

loadXMLColors

Description

Load overlay color data from XML.

Definition

loadXMLColors()

loadProfileColors

Description

Load overlay color data from a profile.

Definition

loadProfileColors()

createOverlay

Description

(Re-)Create an overlay.

Definition

createOverlay(Overlay Overlay, filename Path)

Arguments

Overlay	Overlay	table, see loadOverlay()
filename	Path	to image file (can also be a URL for web images)

Return Values

Overlay table with added image data

copyOverlay

Description

Copy an overlay.

Definition

copyOverlay()

deleteOverlay

Description

Delete an overlay.
Primarily releases the associated image file handle.

Definition

deleteOverlay()

getOverlayColor

Description

Get an overlay's color for a given overlay state.

Definition

getOverlayColor(Overlay Overlay, state GuiOverlay.STATE_[...])

Arguments

Overlay	Overlay	table
state	GuiOverlay.STATE_[...]	constant value

Return Values

Color as {red, green, blue, alpha} with all values in the range of [0, 1]

getOverlayUVs

Description

Get an overlay's UV coordinates for a given overlay state.

Definition

getOverlayUVs(Overlay Overlay, state GuiOverlay.STATE_[...])

Arguments

Overlay	Overlay	table
state	GuiOverlay.STATE_[...]	constant value

Return Values

UV coordinates as {u1, v1, u2, v2, u3, v3, u4x, v4} with all values in the range of [0, 1]

renderOverlay

Description

Renders an overlay with the given parameters.

Definition

renderOverlay(Overlay Overlay, posX Screen, posY Screen, sizeX Screen, sizeY Screen, state GuiOverlay.STATE_[...])

Arguments

Overlay	Overlay	table
posX	Screen	x position
posY	Screen	y position
sizeX	Screen	x size
sizeY	Screen	y size
state	GuiOverlay.STATE_[...]	constant for the required display state

GuiProfile

Description

GUI element display profile.
-- Holds GuiElement property data for re-use similar to a HTML/CSS definition.
--@category GUI

new

Description

Create a new GuiProfile.

Definition

new(profiles Reference, traits Reference)

Arguments

profiles	Reference to loaded profiles table for inheritance checking.
traits	Reference to loaded traits table for inheritance checking.

Return Values

New GuiProfile instance

loadFromXML

Description

Load profile data from XML.

Definition

loadFromXML(xmlFile XML, key Profile, presets Table, isTrait Whether)

Arguments

xmlFile	XML	file handle
key	Profile	XML element node path
presets	Table	of presets for symbol resolution, {preset name=preset value}
isTrait	Whether	this profile is a trait

Return Values

True if profile values could be loaded, false otherwise.

getValue

Description

Get a string value from this profile (and its ancestors) by name.

Definition

getValue(name Name, default Default)

Arguments

name	Name	of attribute value to retrieve
default	Default	value to use if the attribute is not defined.

getBool

Description

Get a boolean value from this profile (and its ancestors) by name.

Definition

getBool(name Name, default Default)

Arguments

name	Name	of attribute value to retrieve
default	Default	value to use if the attribute is not defined.

getNumber

Description

Get a number value from this profile (and its ancestors) by name.

Definition

getNumber(name Name, default Default)

Arguments

name	Name	of attribute value to retrieve
default	Default	value to use if the attribute is not defined.

GuiSoundPlayer

Description

GUI sound player.
-- This class loads known GUI sound samples as non-spatial samples to be played in the GUI (menu and HUD).
--@category GUI

new

Description

Create a new GuiSoundPlayer instance.

Definition

new(table soundManager)

Arguments

table soundManager SoundManager reference

loadSounds

Description

Load GUI sound samples from definitions.

Definition

loadSounds()

playSample

Description

Play a GUI sound sample identified by name.
The sample must have been loaded when the GUI was created.

Definition

playSample(string sampleName)

Arguments

string sampleName	Name of the sample to play, use one of the identifiers in GuiSoundPlayer.SOUND_SAMPLES.

GuiUtils

Description

GUI utility functions.
--@category GUI

getNormalizedValues

Description

Transform an attribute string representing a list of numbers into an array and normalize the values.

Definition

getNormalizedValues(str Attribute, refSize Reference, defaultValue Default)

Arguments

str	Attribute	string containing numbers, either raw or with a pixel unit designation on each number (e.g. "12 24" or "12px 24px")
refSize	Reference	size for normalization, e.g. a reference screen resolution used to scale pixel values, {sizeX, sizeY}
defaultValue	Default	value to return if the "str" parameter value is nil

Return Values

array of normalized values

get2DArray

Description

Transform an attribute string representing a 2D array into an actual array.

Definition

get2DArray(str Attribute, defaultValue Default)

Arguments

str	Attribute	string containing exactly 2 numbers
defaultValue	Default	value to return if the "str" parameter value is nil or invalid for transformation.

Return Values

array of the 2 converted values as numbers: {value1, value2}

get4DArray

Description

Transform an attribute string representing a 4D array into an actual array.

Definition

get4DArray(str Attribute, defaultValue Default)

Arguments

str	Attribute	string containing exactly 4 numbers
defaultValue	Default	value to return if the "str" parameter value is nil or invalid for transformation.

Return Values

array of the 4 converted values as numbers: {value1, value2, value3, value4}

getColorArray

Description

Transform an attribute string representing a 4D color array into an actual array.

Definition

getColorArray(str Attribute, defaultValue Default)

Arguments

str	Attribute	string containing exactly 4 numbers
defaultValue	Default	value to return if the "str" parameter value is nil or invalid for transformation.

Return Values

array of the 4 converted values as numbers: {red, green, blue, alpha}

getUVs

Description

Transform an attribute string representing a UV array into an actual array and normalize the values.

Definition

getUVs(str Attribute, ref Texture, defaultValue Default)

Arguments

str	Attribute	string containing exactly 4 numbers, order and format: "x[px] y[px] sizeX[px] sizeY[px]"
ref	Texture	reference size used to normalize pixel UV coordinates into unit sized UV coordinates
defaultValue	Default	value to return if the "str" parameter value is nil or invalid for transformation.

Return Values

array of the UV coordinates as {u1, v1, u2, v2, u3, v3, u4, v4}

checkOverlayOverlap

Description

Check if a point lies within or a hotspot overlaps an overlay.

Definition

checkOverlayOverlap(posX Point, posY Point, overlayX Overlay, overlayY Overlay, overlaySizeX Overlay, overlaySizeY Overlay, hotspot If)

Arguments

posX	Point	or hotspot x position
posY	Point	or hotspot y position
overlayX	Overlay	x position

overlayY	Overlay y position	
overlaySizeX	Overlay width	
overlaySizeY	Overlay height	
hotspot	If	provided as an array having 4 numbers for the bounding points of a rectangle {minX, minY, maxX, maxY}, will be checked if it overlaps the overlay area given by the other parameters.

HighPressureWasherPlaceable
Description

HighPressureWasher Activatable

new

Description

Creating placeable high pressure washer

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean	isServer	is server
boolean	isClient	is client
table	customMt	custom metatable

Return Values

table instance Instance of object

Code

```
20  function HighPressureWasher:new(isServer, isClient, customMt)
21  local mt = customMt
22  if mt == nil then
23  mt = HighPressureWasher_mt
24  end
25
26  local self = Placeable:new(isServer, isClient, mt)
27  registerObjectClassName(self, "HighPressureWasher")
28
29  return self
30  end
```

load

Description

Load high pressure washer

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position
float	y	z world position

float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Return Values

boolean success success

Code

```
43  function HighPressureWasher:load(xmlFilename, x,y,z, rx,ry,rz, initRandom)
44  if not HighPressureWasher:superClass().load(self, xmlFilename, x,y,z, rx,ry,rz, initRandom) then
45  return false
46  end
47
48  local xmlFile = loadXMLFile("TempXML", xmlFilename)
49
50  self.lanceNode = I3DUtil.indexToObject(self.nodeId, getXMLString(xmlFile, "placeable.highPressureWasher.lance#index"))
51  self.handtoolXML = Utils.getFilename(getXMLString(xmlFile, "placeable.highPressureWasher.handtool#filename"), self.baseDirectory)
52  self.playerInRangeDistance = Utils.getNoNil(getXMLFloat(xmlFile, "placeable.highPressureWasher.playerInRangeDistance"), 3)
53  self.actionRadius = Utils.getNoNil(getXMLFloat(xmlFile, "placeable.highPressureWasher.actionRadius#distance"), 15)
54
55  if self.isClient then
56  self.hpwSamples = {}
57  if self.isClient then
58  self.hpwSamples.compressor = g_soundManager:loadSampleFromXML(xmlFile, "placeable.highPressureWasher.sounds", "compressor", self.baseDirectory, self.nodeId, 0, AudioGroup.VEHICLE, nil, self)
59  self.hpwSamples.switch = g_soundManager:loadSampleFromXML(xmlFile, "placeable.highPressureWasher.sounds", "switch", self.baseDirectory, self.nodeId, 1, AudioGroup.VEHICLE, nil, self)
60  end
61
62  local filename = getXMLString(xmlFile, "placeable.highPressureWasher.exhaust#filename")
63  if filename ~= nil then
64  local i3dNode = g_i3DManager:loadSharedI3DFile(filename, self.baseDirectory, false, false, false)
65  if i3dNode ~= 0 then
```

```
66  local linkNode = Utils.getNoNil(I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile,
    "placeable.highPressureWasher.exhaust#index")), self.nodeId)
67  self.exhaustFilename = filename
68  self.exhaustNode = getChildAt(i3dNode, 0)
69  link(linkNode, self.exhaustNode)
70  setVisibility(self.exhaustNode, false)
71  delete(i3dNode)
72  end
73  end
74  end
75
76  delete(xmlFile)
77
78  self.isPlayerInRange = false
79  self.isTurnedOn = false
80  self.isTurningOff = false
81  self.turnOffTime = 0
82  self.turnOffDuration = 500
83  self.activatable = HighPressureWasherActivatable:new(self)
84  self.lastInRangePosition = {0,0,0}
85
86  return true
87  end
```

delete

Description

Deleting placeable high pressure washer

Definition

delete()

Code

```
91  function HighPressureWasher:delete()
92  self:setIsTurnedOn(false, nil, false)
93  if self.isClient then
94  if self.exhaustFilename ~= nil then
95  g_i3DManager:releaseSharedI3DFile(self.exhaustFilename,
    self.baseDirectory, true)
96  end
97  g_soundManager:deleteSamples(self.hpwSamples)
98  end
99
100 unregisterObjectClassName(self)
```

```
101  g_currentMission:removeActivatableObject(self.activatable)
102  HighPressureWasher:superClass().delete(self)
103  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
109  function HighPressureWasher:readStream(streamId, connection)
110  HighPressureWasher:superClass().readStream(self, streamId,
     connection)
111  if connection:getIsServer() then
112  local isTurnedOn = streamReadBool(streamId)
113  if isTurnedOn then
114  local player = NetworkUtil.readNodeObject(streamId)
115  if player ~= nil then
116  self:setIsTurnedOn(isTurnedOn, player, true)
117  end
118  end
119  end
120  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
126  function HighPressureWasher:writeStream(streamId, connection)
127  HighPressureWasher:superClass().writeStream(self, streamId,
     connection)
128  if not connection:getIsServer() then
129  streamWriteBool(streamId, self.isTurnedOn)
130  if self.isTurnedOn then
131  NetworkUtil.writeNodeObject(streamId, self.currentPlayer)
132  end
```

```
133 end
134 end
```

activateHandtool

Description

Activate hand tool

Definition

activateHandtool(table player)

Arguments

table player player

Code

```
139 function HighPressureWasher:activateHandtool(player)
140 self:setIsTurnedOn(true, player, true)
141 end
```

update

Description

Update

Definition

update(float dt)

Arguments

float dt time since last call in ms

Code

```
146 function HighPressureWasher:update(dt)
147 HighPressureWasher:superClass().update(self, dt)
148
149 if self.currentPlayer ~= nil then
150 local isPlayerInRange = self:getIsPlayerInRange(self.actionRadius,
    self.currentPlayer)
151 if isPlayerInRange then
152 self.lastInRangePosition = {getTranslation(self.currentPlayer.rootNode)}
153 else
154 local kx, _, kz = getWorldTranslation(self.nodeId)
155 local px, py, pz = getWorldTranslation(self.currentPlayer.rootNode)
156 local len = MathUtil.vector2Length(px-kx, pz-kz)
157
158 local x,y,z = unpack(self.lastInRangePosition)
159 x = kx + ((px-kx) / len) * (self.actionRadius-0.00001*dt)
160 z = kz + ((pz-kz) / len) * (self.actionRadius-0.00001*dt)
161 self.currentPlayer:moveToAbsoluteInternal(x, py, z)
162 self.lastInRangePosition = {x,y,z}
163
164 if self.currentPlayer == g_currentMission.player then
```

```
165  g_currentMission:showBlinkingWarning(g_i18n:getText("warning_hpwRangeRes
     4000)
166  end
167  end
168  end
169
170  if self.isClient then
171  if self.isTurningOff then
172  if g_currentMission.time > self.turnOffTime then
173  self.isTurningOff = false
174  g_soundManager:stopSample(self.hpwSamples.compressor)
175  end
176  end
177  end
178
179  self:raiseActive()
180  end
```

updateTick

Description

updateTick

Definition

updateTick(float dt)

Arguments

float dt time since last call in ms

Code

```
197  function HighPressureWasher:updateTick(dt)
198  HighPressureWasher:superClass().updateTick(self, dt)
199  local isPlayerInRange, player =
     self:getIsPlayerInRange(self.playerInRangeDistance)
200
201  if isPlayerInRange and
     g_currentMission.accessHandler:canPlayerAccess(self, player) then
202  self.playerInRange = player
203  self.isPlayerInRange = true
204  g_currentMission:addActivatableObject(self.activatable)
205  else
206  self.playerInRange = nil
207  self.isPlayerInRange = false
208  g_currentMission:removeActivatableObject(self.activatable)
209  end
210  end
```

setIsTurnedOn

Description

Set is turned on

Definition

setIsTurnedOn(boolean isTurnedOn, table player, boolean noEventSend)

Arguments

boolean	isTurnedOn	is turned on
table	player	player
boolean	noEventSend	no event send

Code

```
217 function HighPressureWasher:setIsTurnedOn(isTurnedOn, player, noEventSen
218 HPWPlaceableTurnOnEvent.sendEvent(self, isTurnedOn, player, noEventSend)
219
220 if self.isTurnedOn ~= isTurnedOn then
221 if isTurnedOn then
222 self.isTurnedOn = isTurnedOn
223
224 if player ~= nil then
225 self.currentPlayer = player
226 self.currentPlayer:addDeleteListener(self, "onPlayerDelete")
227 if noEventSend ~= true then
228 self.currentPlayer:equipHandtool(self.handtoolXML, true, noEventSend)
229 self.currentPlayer.baseInformation.currentHandtool:addDeleteListener(sel
    "onHandtoolDelete")
230 end
231 end
232
233 if self.isClient then
234 g_soundManager:playSample(self.hpwSamples.switch)
235 g_soundManager:playSample(self.hpwSamples.compressor)
236
237 if self.isTurningOff then
238 self.isTurningOff = false
239 end
240 setVisibility(self.lanceNode, false)
241 end
242 else
243 self:onDeactivate()
244 end
245 if self.exhaustNode ~= nil then
246 setVisibility(self.exhaustNode, isTurnedOn)
247 end
```

```
248 end
249 end
```

onPlayerDelete

Description

Definition

onPlayerDelete()

Code

```
253 function HighPressureWasher:onPlayerDelete()
254 self.currentPlayer = nil
255 self:setIsTurnedOn(false, nil, nil)
256 end
```

onHandtoolDelete

Description

Definition

onHandtoolDelete()

Code

```
260 function HighPressureWasher:onHandtoolDelete()
261 self.currentPlayer = nil
262 self:setIsTurnedOn(false, nil, nil)
263 end
```

onDeactivate

Description

On deactivate

Definition

onDeactivate()

Code

```
267 function HighPressureWasher:onDeactivate()
268 if self.isClient then
269 g_soundManager:playSample(self.hpwSamples.switch)
270 g_soundManager:stopSample(self.hpwSamples.washing, true)
271 self.isTurningOff = true
272 self.turnOffTime = g_currentMission.time + self.turnOffDuration
273 end
274 self.isTurnedOn = false
275 setVisibility(self.lanceNode, true)
276 if self.currentPlayer ~= nil then
277 if self.currentPlayer:hasHandtoolEquipped() then
278 self.currentPlayer.baseInformation.currentHandtool:removeDeleteListener
    "onHandtoolDelete")
279 self.currentPlayer:unequipHandtool()
280 end
281 self.currentPlayer:removeDeleteListener(self, "onPlayerDelete")
```

```
282  self.currentPlayer = nil
283  end
284  end
```

getIsActiveForInput

Description

Get is active for input

Definition

getIsActiveForInput()

Return Values

boolean isActiveForInput is active for input

Code

```
289  function HighPressureWasher:getIsActiveForInput()
290  if self.isTurnedOn and self.currentPlayer ==
     g_currentMission.player and not g_gui:getIsGuiVisible() then
291  return true
292  end
293  return false
294  end
```

getIsActiveForSound

Description

Get is active for sound

Definition

getIsActiveForSound()

Return Values

boolean isActiveForSound is active for sound

Code

```
299  function HighPressureWasher:getIsActiveForSound()
300  return self:getIsActiveForInput()
301  end
```

canBeSold

Description

Get can be sold in current state

Definition

canBeSold()

Return Values

boolean

string warningMessage warning message displayed in the shop

Code

```
307  function HighPressureWasher:canBeSold()
308  local warning = g_i18n:getText("shop_messageReturnVehicleInUse")
309  if self.currentPlayer ~= nil then
310  return false, warning
```

```
311 end
312 return true, nil
313 end
```

Horse
Description

Minimum daily riding time for a horse below which it will lose fitness.

new

Description
Definition

new()

Code

```
22 function Horse:new(isServer, isClient, owner, fillTypeIndex,
   customMt)
23 local self = RideableAnimal:new(isServer, isClient, owner,
   fillTypeIndex, customMt or Horse_mt)
24
25 self.name = g_animalNameManager:getRandomName()
26
27 self.fitnessScale = 0.0
28 self.fitnessScaleSent = 0.0
29
30 self.healthScale = 1.0
31 self.healthScaleSent = 1.0
32
33 self.ridingTimer = 0.0
34
35 self.horseDirtyFlag = self:getNextDirtyFlag()
36
37 return self
38 end
```

readStream

Description
Definition

readStream()

Code

```
42 function Horse:readStream(streamId)
43 Horse:superClass().readStream(self, streamId)
44
45 self.name = streamReadString(streamId)
46 self.fitnessScale = streamReadFloat32(streamId)
47 self.healthScale = streamReadFloat32(streamId)
48 self.ridingTimer = streamReadFloat32(streamId)
```

```
49  end
```

writeStream

Description

Definition

writeStream()

Code

```
53  function Horse:writeStream(streamId)
54  Horse:superClass().writeStream(self, streamId)
55
56  streamWriteString(streamId, self.name)
57  streamWriteFloat32(streamId, self.fitnessScale)
58  streamWriteFloat32(streamId, self.healthScale)
59  streamWriteFloat32(streamId, self.ridingTimer)
60  end
```

readUpdateStream

Description

Definition

readUpdateStream()

Code

```
64  function Horse:readUpdateStream(streamId, timestamp, connection)
65  Horse:superClass().readUpdateStream(self, streamId, timestamp,
    connection)
66
67  -- server to client only
68  if connection:getIsServer() then
69  if streamReadBool(streamId) then
70  self.fitnessScale =
    NetworkUtil.readCompressedPercentages(streamId, 7)
71  self.healthScale = NetworkUtil.readCompressedPercentages(streamId,
    7)
72  end
73  end
74  end
```

writeUpdateStream

Description

Definition

writeUpdateStream()

Code

```
78  function Horse:writeUpdateStream(streamId, connection, dirtyMask)
79  Horse:superClass().writeUpdateStream(self, streamId, connection,
    dirtyMask)
80
81  -- server to client only
```

```
82  if not connection:getIsServer() then
83  if streamWriteBool(streamId, bitAND(dirtyMask,
    self.horseDirtyFlag) ~= 0) then
84  NetworkUtil.writeCompressedPercentages(streamId,
    self.fitnessScaleSent, 7)
85  NetworkUtil.writeCompressedPercentages(streamId,
    self.healthScaleSent, 7)
86  end
87  end
88  end
```

loadFromXMLFile

Description

Definition

loadFromXMLFile()

Code

```
92  function Horse:loadFromXMLFile(xmlFile, key)
93  Horse:superClass().loadFromXMLFile(self, xmlFile, key)
94
95  self.name = getXMLString(xmlFile, key.."#name") or self.name
96  self.fitnessScale = getXMLFloat(xmlFile, key.."#fitnessScale") or
    self.fitnessScale
97  self.healthScale = getXMLFloat(xmlFile, key.."#healthScale") or
    self.healthScale
98  self.ridingTimer = getXMLFloat(xmlFile, key.."#ridingTimer") or
    self.ridingTimer
99  self.fitnessScaleSent = self.fitnessScale
100 end
```

saveToXMLFile

Description

Definition

saveToXMLFile()

Code

```
104 function Horse:saveToXMLFile(xmlFile, key, usedModNames)
105 Horse:superClass().saveToXMLFile(self, xmlFile, key,
    usedModNames)
106
107 setXMLString(xmlFile, key.."#name", self.name)
108 setXMLFloat(xmlFile, key.."#fitnessScale", self.fitnessScale)
109 setXMLFloat(xmlFile, key.."#healthScale", self.healthScale)
110 setXMLFloat(xmlFile, key.."#ridingTimer", self.ridingTimer)
111 end
```

setName

Description

Definition

setName()

Code

```
115  function Horse:setName(name)
116
117  if self.name ~= nil and self.name ~= name then
118  self.name = name
119  g_messageCenter:publish(MessageType.HUSBANDRY_ANIMALS_CHANGED,
     self.owner)
120  end
121  end
```

getName

Description

Definition

getName()

Code

```
125  function Horse:getName()
126  return self.name
127  end
```

getValueScale

Description

Definition

getValueScale()

Code

```
131  function Horse:getValueScale()
132  -- dirt scale should only count 10%
133  return 0.90 * (self.fitnessScale * self.healthScale) + 0.10 * (1-
     self.dirtScale)
134  end
```

setFitnessScale

Description

Definition

setFitnessScale()

Code

```
138  function Horse:setFitnessScale(scale, noEventSend)
139  self.fitnessScale = scale
140  if math.abs(self.fitnessScaleSent - self.fitnessScale) > 0.01
     then
141  self.fitnessScaleSent = self.fitnessScale
142  g_messageCenter:publish(MessageType.HUSBANDRY_ANIMALS_CHANGED,
     self.owner)
143
144  if noEventSend == nil or not noEventSend then
145  self:raiseDirtyFlags(self.horseDirtyFlag)
```

```
146 end
147 end
148 end
```

getFitnessScale

Description

Definition

getFitnessScale()

Code

```
152 function Horse:getFitnessScale()
153 return self.fitnessScale
154 end
```

setHealthScale

Description

Definition

setHealthScale()

Code

```
158 function Horse:setHealthScale(scale, noEventSend)
159 self.healthScale = scale
160 if math.abs(self.healthScaleSent - self.healthScale) > 0.01 then
161 self.healthScaleSent = self.healthScale
162 g_messageCenter:publish(MessageType.HUSBANDRY_ANIMALS_CHANGED,
    self.owner)
163
164 if noEventSend == nil or not noEventSend then
165 self:raiseDirtyFlags(self.horseDirtyFlag)
166 end
167 end
168 end
```

getHealthScale

Description

Definition

getHealthScale()

Code

```
172 function Horse:getHealthScale()
173 return self.healthScale
174 end
```

getTodaysRidingTime

Description

Get the riding time of this horse for the current day in milliseconds.

Definition

getTodaysRidingTime()

deactivateRiding

Description

Definition

deactivateRiding()

Code

```
183  function Horse:deactivateRiding(noEventSend)
184  if self.rideableVehicle ~= nil then
185  self.rideableVehicle:setFitnessChangedCallback(nil, nil)
186  end
187
188  Horse:superClass().deactivateRiding(self, noEventSend)
189  end
```

onLoadedRideable

Description

Definition

onLoadedRideable()

Code

```
193  function Horse:onLoadedRideable(rideableVehicle, vehicleLoadState, argum
194  Horse:superClass().onLoadedRideable(self, rideableVehicle, vehicleLoadSt
     arguments)
195
196  if self.rideableVehicle ~= nil then
197  self.rideableVehicle:setFitnessChangedCallback(self.onFitnessChangedCall
     self)
198  end
199  end
```

onFitnessChangedCallback

Description

Definition

onFitnessChangedCallback()

Code

```
203  function Horse:onFitnessChangedCallback(deltaTime)
204  self.ridingTimer = self.ridingTimer + deltaTime
205  end
```

updateFitness

Description

Definition

updateFitness()

Code

```
209  function Horse:updateFitness()
210  if self.isServer then
211  local fitness = self:getFitnessScale()
212
213  if self.ridingTimer < Horse.DAILY_MINIMUM_RIDING_TIME then
214  fitness = fitness - 0.02
```

```
215  else -- add capped gain
216  local fitnessGain = math.min(0.1, 0.1 * self.ridingTimer /
     Horse.DAILY_TARGET_RIDING_TIME)
217  fitness = fitness + fitnessGain
218  end
219
220  fitness = MathUtil.clamp(fitness, 0.0, 1.0)
221  self:setFitnessScale(fitness)
222  end
223
224  self.ridingTimer = 0.0
225  end
```

HUD

Description

Heads-up display.
-- The HUD displays information for the player when in game.
--@category GUI

new

Description

Create a new HUD instance.

Definition

new(bool isServer, bool isServer, bool isConsoleVersion, table messageCenter, table l10n, table inputManager, table inputDisplayManager, table modManager, table fillTypeManager, table fruitTypeManager, table guiSoundPlayer)

Arguments

bool	isServer	If true, the running game instance is a server
bool	isServer	If true, the running game instance is a client
bool	isConsoleVersion	If true, we are running on console
table	messageCenter	MessageCenter reference for message subscriptions
table	l10n	I18N reference for text localization
table	inputManager	InputBinding reference
table	inputDisplayManager	InputDisplayManager reference
table	modManager	ModManager reference
table	fillTypeManager	FillTypeManager reference
table	fruitTypeManager	FruitTypeManager reference
table	guiSoundPlayer	GuiSoundPlayer reference

createDisplayComponents

Description

Create all required display components.

Definition

createDisplayComponents(uiScale Current)

Arguments

uiScale	Current	UI scale

delete
Description

Delete the HUD and all its display components.

Definition

delete()

subscribeMessages
Description

Subscribe to relevant state messages.

Definition

subscribeMessages()

setScale
Description

Set the scale of the HUD.

Definition

setScale(float scale)

Arguments

float scale New scale value

drawControlledEntityHUD
Description

Draw the HUD components for the currently controlled entity (player or vehicle).

Definition

drawControlledEntityHUD()

drawInputHelp
Description

Draw the input help display panel and vehicle schema.

Definition

drawInputHelp()

drawTopNotification
Description

Draw a notification at the top center of the screen if one has been set.

Definition

drawTopNotification()

drawBlinkingWarning
Description

Draw a blinking warning if necessary.

Definition

drawBlinkingWarning()

drawPresentationVersion
Description

Draw the presentation mode logo.

Definition

drawPresentationVersion()

drawFading

Description

Draw the screen fade effect.

Definition

drawFading()

drawOverlayAtPositionWithDimensions

Description

Draw an overlay at a given screen space position with given dimensions.
Omitting a parameter will use the last setting of that parameter (or initial setting if parameter is never supplied).

Definition

drawOverlayAtPositionWithDimensions(table overlay, float screenX, float screenY, float screenWidth, float screenHeight)

Arguments

table	overlay	Overlay instance to draw
float	screenX	Drawing origin X position in screen space
float	screenY	Drawing origin Y position in screen space
float	screenWidth	Drawing horizontal size in screen space
float	screenHeight	Drawing vertical size in screen space

drawOverlayAtPosition

Description

Draw an overlay at a given screen space position.
Omitting a parameter will use the last setting of that parameter.

Definition

drawOverlayAtPosition(table overlay, float screenX, float screenY)

Arguments

table	overlay	Overlay instance to draw
float	screenX	Drawing origin X position in screen space
float	screenY	Drawing origin Y position in screen space

drawSideNotification

Description

Draw notification texts at the side of the screen.

Definition

drawSideNotification()

drawBaseHUD

Description

Draw the regular HUD for career mode.

Definition

drawBaseHUD()

drawCommunicationDisplay

Description

Draw the platform communication display (chat or speaker icons).

Definition

drawCommunicationDisplay()

setTutorialProgress

Description

Set the tutorial progress.

Definition

setTutorialProgress(float progress)

Arguments

float progress Progress expressed as a number between 0 and 1

drawGamePaused

Description

Draw a pause notification with a message text and synchronizing background if the menu is visible.

Definition

drawGamePaused(bool beforeMissionStart)

Arguments

bool beforeMissionStart	If true, draw a special overlay for cases when the menu has not yet loaded but the world is already visible.

drawVehicleName

Description

Draw the current vehicle name if it's active.

Definition

drawVehicleName()

drawInGameMessageAndIcon

Description

Draw the message window and icon at the bottom of the screen.

Definition

drawInGameMessageAndIcon()

drawMissionCompleted

Description

Show a "mission accomplished" message.

Definition

drawMissionCompleted()

drawMissionFailed

Description

Show a "mission failed" message.

Definition

drawMissionFailed()

showInGameMessage

Description

Display an in-game message in a window at the bottom.
This calls InGameMessage:showMessage(title, message, duration, controls, callback, target).

Definition

showInGameMessage()

showBlinkingWarning

Description

Display a blinking warning text.

Definition

showBlinkingWarning(string text, int duration, int priority)

Arguments

string	text	Warning text
int	duration	[optional, default=2000] Duration of warning visibility in milliseconds
int	priority	[optional, default=0] Warning priority value. If a new warning is triggered with the same or higher priority as a current one, the current warning is replaced.

addMoneyChange

Description

Accumulate a money transaction amount by type until shown by HUD.showMoneyChange().

Definition

addMoneyChange(int moneyType, float amount)

Arguments

int	moneyType	Type of transaction from EconomyManager.MONEY_TYPE_XYZ
float	amount	Amount of money

showMoneyChange

Description

Display a money change notification at the right side of the screen.

Definition

showMoneyChange(int moneyType, string text)

Arguments

int	moneyType	Type of transaction from EconomyManager.MONEY_TYPE_XYZ
string	text	Specific money type label (e.g. "loan interest")

addExtraPrintText

Description

Add an extra text to display in the input help panel for this frame.

Definition

addExtraPrintText()

showVehicleName

Description

Display the name of a vehicle at the bottom of the screen for a short time.

Definition

showVehicleName(string vehicleName)

Arguments

string	vehicleName	Name of a vehicle

addSideNotification

Description

Add a notification to be displayed at the right side of the screen.

Definition

addSideNotification(table color)

Arguments

table color Color as an RGBA array

addTopNotification

Description

Add a notification to be displayed in a frame at the top of the screen.

Definition

addTopNotification(string title, string text, string info, table iconKey, int duration)

Arguments

string	title	Notification title
string	text	Notification message text
string	info	Additional info text
table	iconKey	[optional] Icon key for a display icon, use a value from TopNotification.ICON
int	duration	[optional] Display duration in milliseconds. Negative values or nil default to a long-ish standard duration.

hideTopNotification

Description

Hide a current notification shown by HUD:addTopNotification().

Definition

hideTopNotification()

getIsFading

Description

Check if the HUD is currently fading the screen.

Definition

getIsFading()

setGameInfoPartVisibility

Description

Set the visibility of the game info display parts by flags.

Definition

setGameInfoPartVisibility(partFlags Combination)

Arguments

partFlags Combination of values from HUD.GAME_INFO_PART

onMenuVisibilityChange

Description

Handle menu visibility changes.
Keeps track of the menu visibility to block relevant processes.

Definition

onMenuVisibilityChange()

onPauseGameChange

Description

Handle game pause state changes.

Definition

onPauseGameChange(bool isPaused, string pauseText)

Arguments

bool isPaused [optional] If true, the game is currently paused. If not set, will not change the pause state.
string pauseText [optional] Text to display on the pause element

setIsVisible

Description

Set HUD visibility.

Definition

setIsVisible(bool isVisible)

Arguments

bool isVisible If true, the HUD is made visible. If false, it's made invisible.

setInputHelpVisible

Description

Set input help visibility.

Definition

setInputHelpVisible(bool isVisible)

Arguments

bool isVisible If true, the input help is displayed. If false, it's made invisible

setFieldInfoVisible

Description

Set field info display visibility.

Definition

setFieldInfoVisible(bool isVisible)

Arguments

bool isVisible If true, the field info display is shown. If false, it's hidden.

addCustomInputHelpEntry

Description

Add a custom input help entry which is displayed until removed.
Custom entries will be displayed in order of addition after any automatically detected input help entries and before
vehicle extensions.

Definition

addCustomInputHelpEntry()

clearCustomInputHelpEntries

Description

Clear all custom input help entries.

Definition

clearCustomInputHelpEntries()

getIsVisible

Description

Check if the HUD is visible.

Definition

getIsVisible()

Return Values

bool If true, the HUD is currently visible.

setControlledVehicle

Description

Set current controlled vehicle.

Definition

setControlledVehicle(table vehicle)

Arguments

table vehicle Vehicle reference or nil (not controlling a vehicle)

setIsControllingPlayer

Description

Set current player controlling state.

Definition

setIsControllingPlayer(bool isControllingPlayer)

Arguments

bool isControllingPlayer If true, the player is controlling their character on foot.

setMoneyUnit

Description

Set the money unit (currency) to display.

Definition

setMoneyUnit(int unit)

Arguments

int unit One of [GS_MONEY_EURO | GS_MONEY_POUND], any other value will display a universal dollar sign.

showAchievementMessage

Description

Display an achievement message.

Definition

showAchievementMessage()

showAttachContext

Description

Display vehicle attachment context information for the current frame.

Definition

showAttachContext()

showTipContext

Description

Display vehicle tipping context information for the current frame.

Definition

showTipContext()

showFuelContext

Description

Display vehicle refueling context information for the current frame.

Definition

showFuelContext()

showFillDogBowlContext

Description

Display dog bowl refilling context information for the current frame.

Definition

showFillDogBowlContext()

setPlayer

Description

Set the reference to the current player.

Definition

setPlayer()

setConnectedUsers

Description

Set the references to the currently connected users.

Definition

setConnectedUsers()

updateMessageAndIcon

Description

Update in-game message window and icon.

Definition

updateMessageAndIcon()

update

Description

Update base HUD state.

Definition

update()

updateBlinkingWarning

Description

Update the blinking warning if one is set.

Definition

updateBlinkingWarning()

updateMap

Description

Update the in-game map.

Definition

updateMap()

updateVehicleName

Description

Update vehicle name display when a vehicle was entered.

Definition

updateVehicleName()

fadeScreen

Description

Fade the screen.

Definition

fadeScreen(int direction, int duration, func callbackFunc, table callbackTarget)

Arguments

int	direction	If > 0 will fade to opaque (i.e. black screen), if < 0 will fade to transparent (i.e. clear)
int	duration	Target duration of the fade animation in milliseconds
func	callbackFunc	[optional] Callback function which is called when the fading animation completes
table	callbackTarget	[optional] Callback function target which is passed to the callback function as its first argument if provided

loadIngameMap

Description

Load graphics and settings for the in-game map.

Definition

loadIngameMap(string ingameMapFilename, int ingameMapWidth, int ingameMapHeight)

Arguments

string	ingameMapFilename	Path to map image
int	ingameMapWidth	Width of the map (X)
int	ingameMapHeight	Height of the map (Y)

setIngameMapSize

Description

Set the size index of the in-game map.

Definition

setIngameMapSize(int sizeIndex)

Arguments

int	sizeIndex	One of [IngameMap.STATE_MINIMAP | IngameMap.STATE_MAP | IngameMap.STATE_OFF]

getIngameMap

Description

Get the in-game map reference.

Definition

getIngameMap()

isInGameMessageActive

Description

Check if the in-game message is currently active.

Definition

isInGameMessageActive()

mouseEvent

Description

Mouse event function.
Only delegates input to components.

Definition

mouseEvent()

setEnvironment

Description

Set the environment reference to use for weather information display.

Definition

setEnvironment(table environment)

Arguments

table environment Environment reference, do not change

setMissionInfo

Description

Set the mission information reference for base information display.

Definition

setMissionInfo(table missionInfo)

Arguments

table missionInfo MissionInfo reference, do not change

setMissionStats

Description

Set the mission statistics reference for base information display.

Definition

setMissionStats(table missionStats)

Arguments

table missionStats MissionStats reference, do not change

setInGameIconOnPickup

Description

Set the in-game icon display when picking up an egg or nugget.

Definition

setInGameIconOnPickup()

scrollChatMessages

Description

Scroll chat messages by a given amount.

Definition

scrollChatMessages(int delta, int numMessages)

Arguments

int delta Number of lines (positive or negative) to scroll

int numMessages Number of currently stored chat messages

setChatWindowVisible

Description

Set chat window visibility.
When disabled, the chat display will linger for some time.

Definition

setChatWindowVisible()

setChatMessagesReference

Description

Set the chat messages array reference for the chat window.
The array is owned by the caller and must not be modified by consumers.

Definition

setChatMessagesReference()

registerInput

Description

Register input events of HUD components.

Definition

registerInput()

HUDDisplayElement

Description

HUD display element whose subclasses implement more complex HUD display subsystems.
--@category GUI

new

Description

Create a new HUD display element.

Definition

new(table subClass, table overlay, table parentHudElement)

Arguments

table subClass	Subclass metatable for inheritance
table overlay	Wrapped Overlay instance
table parentHudElement	[optional] Parent HUD element of the newly created HUD element

Return Values

table HUDDisplayElement instance

setVisible

Description

Set this element's visibility with optional animation.

Definition

setVisible(bool isVisible, bool animate)

Arguments

bool isVisible	True is visible, false is not.
bool animate	If true, the element will play an animation before applying the visibility change.

setScale

Description

Simplification of scale setter because these high-level elements always use a uniform scale.

Definition

setScale()

storeOriginalPosition

Description

Store the current element position as its original positions.

Definition

storeOriginalPosition()

getHidingTranslation

Description

Get the screen space translation for hiding.
Override in sub-classes if a different translation is required.

Definition

getHidingTranslation()

Return Values

float Screen space X translation

float Screen space Y translation

animationSetPositionX

Description

Animation setter function for X position.

Definition

animationSetPositionX()

animationSetPositionY

Description

Animation setter function for Y position.

Definition

animationSetPositionY()

animateHide

Description

Animate this element on hiding.

Definition

animateHide()

animateShow

Description

Animate this element on showing.

Definition

animateShow()

onAnimateVisibilityFinished

Description

Called when a hiding or showing animation has finished.

Definition

onAnimateVisibilityFinished()

HUDElement

Description

Lightweight HUD UI element.
-- Wraps an Overlay instance to display and provides a transform hierarchy of child HUDElement instances.
--@category GUI

new

Description

Create a new HUD element.

Definition

new(table subClass, table overlay, table parentHudElement)

Arguments

table subClass	Subclass metatable for inheritance
table overlay	Wrapped Overlay instance
table parentHudElement	[optional] Parent HUD element of the newly created HUD element

Return Values

table HUDElement instance

delete

Description

Delete this HUD element and all its children.
This will also delete the overlay and thus release its engine handle.

Definition

delete()

addChild

Description

Add a child HUD element to this element.

Definition

addChild(table childHudElement)

Arguments

table childHudElement HUDElement instance which is added as a child.

removeChild

Description

Remove a child HUD element from this element.

Definition

removeChild(table childHudElement)

Arguments

table childHudElement HUDElement instance which is removed as a child.

setPosition

Description

Set a HUD element's absolute screen space position.
If the element has any children, they will be moved with this element.

Definition

setPosition()

setRotation

Description

Set this HUD element's rotation.
Does not affect children. If no center position is given, the element's pivot values are used (default to 0)

Definition

setRotation(float rotation, float centerX, float centerY)

Arguments

float rotation Rotation in radians

float centerX [optional] Rotation pivot X position offset from overlay position in screen space

float centerY [optional] Rotation pivot Y position offset from overlay position in screen space

setRotationPivot

Description

Set this HUD element's rotation pivot point.

Definition

setRotationPivot(float pivotX, float pivotY)

Arguments

float pivotX Pivot x position offset from element position in screen space

float pivotY Pivot y position offset from element position in screen space

getRotationPivot

Description

Get this HUD element's rotation pivot point.

Definition

getRotationPivot()

Return Values

float Pivot x position offset from element position in screen space

float Pivot y position offset from element position in screen space

getPosition

Description

Get this HUD element's position.

Definition

getPosition()

Return Values

float X position in screen space

float Y position in screen space

setScale

Description

Set this HUD element's scale.
This will move and scale children proportionally.

Definition

setScale(float scaleWidth, float scaleHeight)

Arguments

float scaleWidth Width scale factor

float scaleHeight Height scale factor

getScale

Description

Get this HUD element's scale.

Definition

getScale()

Return Values

width	scale factor
height	scale factor

setAlignment

Description

Set this HUD element's positional alignment.
See Overlay:setAlignment for positioning logic.

Definition

setAlignment(int vertical, int horizontal)

Arguments

int vertical	Vertical alignment value [Overlay.ALIGN_VERTICAL_BOTTOM | Overlay.ALIGN_VERTICAL_MIDDLE | Overlay.ALIGN_VERTICAL_TOP]
int horizontal	Horizontal alignment value [Overlay.ALIGN_HORIZONTAL_LEFT | Overlay.ALIGN_HORIZONTAL_CENTER | Overlay.ALIGN_HORIZONTAL_RIGHT]

setVisible

Description

Set this HUD element's visibility.

Definition

setVisible()

getVisible

Description

Get this HUD element's visibility.

Definition

getVisible()

getColor

Description

Get this HUD element's color.

Definition

getColor()

Return Values

float Red	value
float Green	value
float Blue	value
float Alpha	value

getAlpha

Description

Get this HUD element's color alpha value.

Definition

getAlpha()

Return Values

float Alpha value

getWidth

Description

Get this HUD element's width in screen space.

Definition

getWidth()

getHeight

Description

Get this HUD element's height in screen space.

Definition

getHeight()

setDimension

Description

Set this HUD element's width and height.
Either value can be omitted (== nil) for no change.

Definition

setDimension()

resetDimensions

Description

Reset this HUD element's dimensions to their default values.
Resets width, height, scale and pivot.

Definition

resetDimensions()

setColor

Description

Set this HUD element overlay's color.
Children are unaffected.

Definition

setColor()

setAlpha

Description

Set this HUD element overlay's color alpha value only.

Definition

setAlpha()

setImage

Description

Set this HUD element overlay's image file.

Definition

setImage()

setUVs

Description

Set this HUD element overlay's UV coordinates.

Definition

setUVs()

update

Description

Update this HUD element's state.

Definition

update()

draw

Description

Draw this HUD element and all of its children in order of addition.

Definition

draw()

scalePixelToScreenVector

Description

Convert a vector from pixel values into scaled screen space values.

Definition

scalePixelToScreenVector(table vector2D)

Arguments

table vector2D Array of two pixel values

scalePixelToScreenHeight

Description

Convert a vertical pixel value into scaled screen space value.

Definition

scalePixelToScreenHeight(float height)

Arguments

float height Vertical pixel value

scalePixelToScreenWidth

Description

Convert a horizontal pixel value into scaled screen space value.

Definition

scalePixelToScreenWidth(float width)

Arguments

float width Horizontal pixel value

normalizeUVPivot

Description

Convert a texture space pivot to an element-local pivot.

Definition

normalizeUVPivot(table uvPivot, table uvs)

Arguments

table uvPivot Array of two pixel pivot coordinates in texture space

table uvs Array of UV coordinates as {x, y, width, height}

HUDFrameElement

Description

HUD background frame element.
-- Displays a transparent frame with a thick bottom bar for use as a background in HUD elements.
--@category GUI

new

Description

Create a new instance of FrameElement.

Definition

new(string hudAtlasPath, float posX, float posY, float width, float height, table parent)

Arguments

string	hudAtlasPath	Path to the HUD atlas texture
float	posX	Initial X position in screen space
float	posY	Initial Y position in screen space
float	width	Frame width in screen space
float	height	Frame height in screen space
table	parent	[optional] Parent HUDElement which will receive this frame as its child element

createComponents

Description

Create display components.

Definition

createComponents()

setDimension

Description

Set frame element dimensions.
Override from HUDElement to preserve border positioning and sizes.

Definition

setDimension()

HUDPopupMessage

Description

HUD popup message.
-- Displays a modal popup message which requires a player input to be accepted / dismissed or expires after a given
time.
--@category GUI

new

Description

Create a new instance of HUDPopupMessage.

Definition

new(string hudAtlasPath, l10n I18N, inputManager InputBinding, InputDisplayManager InputDisplayManager, IngameMap IngameMap)

Arguments

string	hudAtlasPath	Path to the HUD texture atlas
l10n	I18N	reference for text localization
inputManager	InputBinding	reference for custom input context handling
InputDisplayManager	InputDisplayManager	for input glyph display
IngameMap	IngameMap	reference used to notify the map when a message is shown

Return Values

table HUDPopupMessage instance

showMessage

Description

Show a new message.

Definition

showMessage(string title, string message, int duration, table controls, function callback, table target)

Arguments

string	title	Title text
string	message	Main message text
int	duration	Message display duration in milliseconds. If set to 0, will cause the message to be displayed for a duration derived from the message length. If set to <0, will cause the message to be displayed for a very long time.
table	controls	[optional] Array of InputHelpElement instance for input hint row display
function	callback	[optional] Function to be called when the message is acknowledged or expires
table	target	[optional] Callback target which is passed as the first argument to the given callback function

setPaused

Description

Set the game's paused state on this element.
If the game is paused, the message timer is stopped and no new messages are displayed.

Definition

setPaused()

getVisible

Description

Get this HUD element's visibility.

Definition

getVisible()

DisplayElement:getHidingTranslation

Description

Get the screen space translation for hiding.
Override in sub-classes if a different translation is required.

Definition

DisplayElement:getHidingTranslation()

Return Values

float Screen space X translation

float Screen space Y translation

onMenuVisibilityChange

Description

Handle menu visibility changes.

Definition

onMenuVisibilityChange()

assignCurrentMessage

Description

Assign a new current message and adjust display state accordingly.
This also resizes the message box according to the required space.

Definition

assignCurrentMessage()

getTitleHeight

Description

Get the display height of the current message's title.

Definition

getTitleHeight()

getTextHeight

Description

Get the display height of the current message's text.

Definition

getTextHeight()

getInputRowsHeight

Description

Get the display height of the current message's input rows.

Definition

getInputRowsHeight()

animateHide

Description

Animate this element on showing.

Definition

animateHide()

startMessage

Description

Start displaying a message dequeued from the currently pending messages.
Sets all required display and input state.

Definition

startMessage()

finishMessage

Description

Finish displaying a message after it has either elapsed or been acknowledged by the player.
Resets display and input state and triggers any provided message callback.

Definition

finishMessage()

update

Description

Update this element's state.

Definition

update()

updateCurrentMessage

Description

Update the current message.
Disables this popup when time runs out and dequeues a pending messages for displaying.

Definition

updateCurrentMessage()

updateButtonGlyphs

Description

Update button glyphs when the player input mode has changed.

Definition

updateButtonGlyphs()

setInputActive

Description

Enable / disable input events for message confirmation / skipping.

Definition

setInputActive()

onConfirmMessage

Description

Event function for either InputAction.SKIP_MESSAGE_BOX or
InputAction.MENU_ACCEPT.

Definition

onConfirmMessage()

draw

Description

Draw the message.

Definition

draw()

getBackgroundPosition

Description

Get this element's base background position.

Definition

getBackgroundPosition(float uiScale)

Arguments

float uiScale Current UI scale factor

setScale
Description

Set uniform UI scale.

Definition

setScale()

setDimension
Description

Set this HUD element's width and height.

Definition

setDimension()

storeScaledValues
Description

Store scaled positioning, size and offset values.

Definition

storeScaledValues()

createBackground
Description

Create the background overlay.

Definition

createBackground()

createComponents
Description

Create required display components.

Definition

createComponents()

createInputRow
Description

Create components for an input button row.

Definition

createInputRow()

HUDTextDisplay
Description

HUD text display.
-- Displays a formatted single-line text with optional animations.
--@category GUI

new
Description

Create a new HUDTextDisplay.

Definition

new(float posX, float posY, float textSize, int textAlignment, table textColor, bool textBool)

Arguments

float	posX	Screen space X position of the text display
float	posY	Screen space Y position of the text display
float	textSize	Text size in reference resolution pixels
int	textAlignment	Text alignment as one of RenderText.[ALIGN_LEFT | ALIGN_CENTER | ALIGN_RIGHT]
table	textColor	Text display color as an array {r, g, b, a}
bool	textBool	If true, will render the text in bold

Return Values

table HUDTextDisplay instance

setText

Description

Set the text to display.

Definition

setText(string text, float textSize, int textAlignment, table textColor, bool textBool)

Arguments

string	text	Display text
float	textSize	Text size in reference resolution pixels
int	textAlignment	Text alignment as one of RenderText.[ALIGN_LEFT | ALIGN_CENTER | ALIGN_RIGHT]
table	textColor	Text display color as an array {r, g, b, a}
bool	textBool	If true, will render the text in bold

setScale

Description

Set the text display UI scale.

Definition

setScale()

setVisible

Description

Set this element's visibility.

Definition

setVisible(bool isVisible, bool animate)

Arguments

bool	isVisible	Visibility state
bool	animate	If true, will play the currently set animation on becoming visible or and reset it when necessary.

setAlpha

Description

Set the global alpha value for this text display.
The alpha value will be multiplied with any text color alpha channel value.

Definition

setAlpha()

setTextColorChannels

Description

Set the text color by channels.
Use for dynamic changes and animation.

Definition

setTextColorChannels()

setTextShadow

Description

Set the text shadow state.

Definition

setTextShadow(bool isShadowEnabled, table shadowColor)

Arguments

bool	isShadowEnabled	If true, will cause a shadow to be rendered under the text
table	shadowColor	Shadow text color as an array {r, g, b, a}

setAnimation

Description

Set an animation tween (sequence) for this text display.
The animation can be played when calling HUDTextDisplay:setVisible() with the "animate" paramter set to true.

Definition

setAnimation()

update

Description

Update this element's state.

Definition

update()

draw

Description

Draw the text.

Definition

draw()

HusbandryModuleFoodSpillage

Description

Creating manager

delete

Description

Deletes instance

Definition

delete()

Code

```
29  function HusbandryModuleFoodSpillage:delete()
30  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
34  function HusbandryModuleFoodSpillage:initDataStructures()
35  HusbandryModuleFoodSpillage:superClass().initDataStructures(self)
36
37  self.spillageAreas = {}
38  self.foodToDrop = 0
39  self.spillageFillType = FillType.UNKNOWN
40  self.lineOffset = 0
41  self.cleanlinessFactor = 0.0
42  self.hasCleanliness = false
43  end
```

load

Description

Loads data from xml

Definition

load(table xmlFile, string xmlKey, table rootNode)

Arguments

table	xmlFile	handle
string	xmlKey	from which to read the configuration
table	rootNode	of the husbandry

Return Values

boolean true if loading was successful else false

Code

```
51  function HusbandryModuleFoodSpillage:load(xmlFile, configKey,
    rootNode, owner)
52  if not HusbandryModuleFoodSpillage:superClass().load(self,
    xmlFile, configKey, rootNode, owner) then
53  return false
54  end
55
56  if not hasXMLProperty(xmlFile, configKey) then
57  return false
58  end
59
60  local i = 0
61  while true do
62  local areaKey = string.format("%s.area(%d)", configKey, i)
63  if not hasXMLProperty(xmlFile, areaKey) then
64  break
```

```
65  end
66  local start = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, areaKey .. "#startNode"))
67  local width = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, areaKey .. "#widthNode"))
68  local height = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, areaKey .. "#heightNode"))
69
70  if start ~= nil and width ~= nil and height ~= nil then
71  table.insert(self.spillageAreas, {start = start, width = width,
    height = height})
72  end
73  i = i + 1
74  end
75
76  local spillageFillType = getXMLString(xmlFile, configKey ..
    "#fillType")
77  if spillageFillType ~= nil then
78  local fillTypeIndex =
    g_fillTypeManager:getFillTypeIndexByName(spillageFillType)
79  if fillTypeIndex ~= nil then
80  self.spillageFillType = fillTypeIndex
81  end
82  end
83
84  self.hasCleanliness = true
85
86  return self.spillageFillType ~= nil and #self.spillageAreas > 0
87  end
```

readStream

Description

Reads network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId	network stream identification
table	connection connection information

Code

```
94  function HusbandryModuleFoodSpillage:readStream(streamId,
    connection)
95  HusbandryModuleFoodSpillage:superClass().readStream(self,
    streamId, connection)
96
```

```
97  if self.hasCleanliness then
98  self.cleanlinessFactor = streamReadUInt8(streamId) / 255
99  end
100 end
```

writeStream

Description

Writes network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
106 function HusbandryModuleFoodSpillage:writeStream(streamId,
    connection)
107 HusbandryModuleFoodSpillage:superClass().writeStream(self,
    streamId, connection)
108
109 if self.hasCleanliness then
110 local cleanliness = math.floor(self.cleanlinessFactor * 255 +
    0.5)
111 streamWriteUInt8(streamId, cleanliness)
112 end
113 end
```

readUpdateStream

Description

Read updates from network stream

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId network stream identification

integer timestamp

table connection connection information

Code

```
120 function HusbandryModuleFoodSpillage:readUpdateStream(streamId,
    timestamp, connection)
121 HusbandryModuleFoodSpillage:superClass().readUpdateStream(self,
    streamId, timestamp, connection)
122
123 if self.hasCleanliness then
124 self.cleanlinessFactor = streamReadUInt8(streamId) / 255
125 end
126 end
```

writeUpdateStream

Description

Write updates from network stream

Definition

writeUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId network stream identification

table connection connection information

integer dirtyMask is used to check if we need to update

Code

```
133 function HusbandryModuleFoodSpillage:writeUpdateStream(streamId, connection, dirtyMask)
134 HusbandryModuleFoodSpillage:superClass().writeUpdateStream(self, streamId, connection, dirtyMask)
135
136 if self.hasCleanliness then
137 local cleanliness = math.floor(self.cleanlinessFactor * 255 + 0.5)
138 streamWriteUInt8(streamId, cleanliness)
139 end
140 end
```

onIntervalUpdate

Description

Update water usage

Definition

onIntervalUpdate(float dayToInterval)

Arguments

float dayToInterval

Code

```
145 function HusbandryModuleFoodSpillage:onIntervalUpdate(dayToInterval)
146 HusbandryModuleFoodSpillage:superClass().onIntervalUpdate(self, dayToInterval)
147 self:updateCleanlinessFactor(dayToInterval)
148 end
```

loadFromXMLFile

Description

Loads information from attributes and node. Retrives from xml file information.

Definition

loadFromXMLFile(table xmlFile, string key)

Arguments

table xmlFile XML file handler

string key XML base key

Code

```
183 function HusbandryModuleFoodSpillage:loadFromXMLFile(xmlFile,
    key)
184 HusbandryModuleFoodSpillage:superClass().loadFromXMLFile(self,
    xmlFile, key)
185
186 if self.hasCleanliness then
187 self.cleanlinessFactor = Utils.getNoNil(getXMLFloat(xmlFile,
    key.."#cleanlinessFactor"), self.cleanlinessFactor)
188 self.foodToDrop = Utils.getNoNil(getXMLFloat(xmlFile,
    key.."#foodToDrop"), self.foodToDrop)
189 end
190 end
```

updateFoodSpillage

Description

Update spillage mechanics.

Definition

updateFoodSpillage(float spillageDelta)

Arguments

float spillageDelta amount of food to increase

Return Values

float food dropped

Code

```
205 function HusbandryModuleFoodSpillage:updateFoodSpillage(spillageDelta)
206 local foodDropped = 0
207
208 if self.hasCleanliness and self.cleanlinessFactor > 0 and
    spillageDelta >
    g_densityMapHeightManager:getMinValidLiterValue(self.spillageFillType)
    then
209 local i = math.random(1, #self.spillageAreas)
210 local spillageArea = self.spillageAreas[i]
211 local xs,_,zs = getWorldTranslation(spillageArea.start)
212 local xw,_,zw = getWorldTranslation(spillageArea.width)
213 local xh,_,zh = getWorldTranslation(spillageArea.height)
214 local ux, uz = xw - xs, zw - zs
215 local vx, vz = xh - xs, zh - zs
216 local vLength = MathUtil.vector2Length(vx,vz)
217 local sx = xs + (math.random() * ux) + (math.random() * vx)
218 local sz = zs + (math.random() * uz) + (math.random() * vz)
219 local ex = xs + (math.random() * ux) + (math.random() * vx)
220 local ez = zs + (math.random() * uz) + (math.random() * vz)
```

```
221 local sy = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, sx, 0.0, sz)
222 local ey = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, ex, 0.0, ez)
223 local dropped, lineOffset = DensityMapHeightUtil.tipToGroundAroundLine(nil, spillageDelta, self.spillageFillType, sx,sy,sz, ex,ey,ez, 0, vLength, self.lineOffset, false, nil)
224
225 foodDropped = dropped
226 self.lineOffset = lineOffset
227 end
228
229 return foodDropped
230 end
```

getFoodSpillageLevel

Description

Get spillage level

Definition

getFoodSpillageLevel()

Return Values

float spillage total

Code

```
235 function HusbandryModuleFoodSpillage:getFoodSpillageLevel()
236 local totalLevel = 0
237 for _, spillageArea in ipairs(self.spillageAreas) do
238 local xs,_,zs = getWorldTranslation(spillageArea.start)
239 local xw,_,zw = getWorldTranslation(spillageArea.width)
240 local xh,_,zh = getWorldTranslation(spillageArea.height)
241 local fillLevel = DensityMapHeightUtil.getFillLevelAtArea(self.spillageFillType, xs, zs, xw, zw, xh, zh)
242 totalLevel = totalLevel + fillLevel
243 end
244 return totalLevel
245 end
```

getSpillageFactor

Description

Definition

getSpillageFactor()

Code

```
249 function HusbandryModuleFoodSpillage:getSpillageFactor()
```

```
250  if self.hasCleanliness then
251  return self.cleanlinessFactor
252  end
253
254  return nil
255  end
```

I3DUtil

Description

Sets the world direction for a given object

checkChildIndex

Description

Checks if given index is valid

Definition

checkChildIndex(integer node, string index)

Arguments

integer	node	id of an object
string	index	index

Return Values

boolean	isActivateable	is activateable
boolean	isValid	true if index is valid else false

indexToObject

Description

Returns index of object

Definition

indexToObject(table components, string index, table mappings)

Arguments

table	components	components (also (integer) id of root node possible)
string	index	index
table	mappings	id to index mapping table

Return Values

table	instance	instance of object
integer	id	id of object
integer	id	id of used root node

setNumberShaderByValue

Description

Sets the number shader attributes

Definition

setNumberShaderByValue(integer numbers, float value, integer precision, boolean showZero)

Arguments

integer	numbers	id of the node containing the number shapes
float	value	value
integer	precision	precision

boolean showZero true if zero should be shown

Return Values

table self returns the instance

wakeUpObject

Description

Wake up a collision object

Definition

wakeUpObject(integer node)

Arguments

integer node id of a collision object

Return Values

bool return true if successful

setWorldDirection

Description

limitedAxis 3: 0 degree = x axis, 90 degree = y axis

Definition

setWorldDirection(integer node, float dirX, float dirY, float dirZ, float upX, float upY, float upZ, integer limitedAxis, float minRot, float maxRot)

Arguments

integer	node	id of object
float	dirX	dirX
float	dirY	dirY
float	dirZ	dirZ
float	upX	upX
float	upY	upY
float	upZ	upZ
integer	limitedAxis	index of limited axis
float	minRot	minRot
float	maxRot	maxRot

Return Values

float returns the change delta

setDirection

Description

Sets direction for a given object (only if the direction is valid)

Definition

setDirection(integer node, float dirX, float dirY, float dirZ, float upX, float upY, float upZ)

Arguments

integer node id of object

float	dirX	dirX
float	dirY	dirY
float	dirZ	dirZ
float	upX	upX
float	upY	upY

float	upZ	upZ

Return Values

table self returns the instance

setShaderParameterRec

Description

Sets a shader parameter recursivly

Definition

setShaderParameterRec(integer node, string shaderParam, float x, float y, float z, float w)

Arguments

integer	node	id of the top node
string	shaderParam	the name of the shader param
float	x	x
float	y	y
float	z	z
float	w	w

Return Values

bool return true if successful

getNodesByShaderParam

Description

Gets a list of nodes with shaderParam assigned to it

Definition

getNodesByShaderParam(integer node, string shaderParam, table nodes)

Arguments

integer	node	id of the top node
string	shaderParam	the name of the shader param
table	nodes	a list of nodes

Return Values

IndexChangeSubjectMixin

Description

Index change subject mixin.
-- Add this mixin to a GuiElement to implement an observer pattern for index changes (e.g. paging, options, lists).
-- Added methods:
GuiElement:addIndexChangeObserver(observer, indexChangeCallback): Register an observer to be notified on index changes
observer is passed to the index change callback as the first argument, typically you would use "self"
indexChangeCallback must be a function with the signature function(observer, index, count)
GuiElement:notifyIndexChange(index, count): Called by the decorated GuiElement when the index or number of indexed items changes, triggers callbacks
index is the new index after the current change
count is the number of indexed items which may or may not have changed
--@category GUI

addTo

Description

See GuiMixin:addTo().

Definition

addTo()

addIndexChangeObserver

Description

Add an index change observer with a callback.

Definition

addIndexChangeObserver(GuiElement Decorated, observer Observer, indexChangeCallback Function(observer,)

Arguments

GuiElement	Decorated	GuiElement instance which has received this method
observer	Observer	object instance
indexChangeCallback	Function(observer,	index, count), where index is the new index and count the current number of indexable items

notifyIndexChange

Description

Notify observers of an index change.

Definition

notifyIndexChange(GuiElement Decorated, index New, count Indexable)

Arguments

GuiElement	Decorated	GuiElement instance which has received this method
index	New	index
count	Indexable	item count

clone

Description

Clone this mixin's state from a source to a destination GuiElement instance.

Definition

clone()

IndexStateElement

Description

Index state display element.
-- Displays a visual element per screen page to let the player see an indication of their current list, page or multi-
option selection index. The element points at a UI element which supports indexed access, such as ListElement,
PageElement or MultiTextOption. For other cases, UI views directly manipulate the states (e.g. cycling hints in the
loading screen).
--@category GUI
--@xmlConfig GuiElement#stateElementTemplateId string ID of a descendant, sibling or sibling's descendant GUI element which is duplicated per page. The element should be configured to have different visuals for normal and selected GuiOverlay states (layers).

locateStateElementTemplate

Description

Find and store the state element template which is configured in #stateElementTemplateId.
The function searches from this elements parent downwards, so any descendant of this element or its siblings may
contain the state element template.

Definition

locateStateElementTemplate()

locateIndexableElement

Description

Find and store the indexable element which is configured in #indexableElementId.
The function first locates the current view root (if possible) and then searches downwards. The search is broader
than locateStateElementTemplate() to allow more flexibility in UI design and configuration.

Definition

locateIndexableElement()

onIndexChange

Description

Event handler for index changes.

Definition

onIndexChange(index New, count (New))

Arguments

index New	index
count (New)	item count

setPageCount

Description

Set the page count.
Clears current page elements and rebuilds as many as needed from the configured template.

Definition

setPageCount(count Number, initialIndex [optional])

Arguments

count	Number of page indicators to display
initialIndex [optional]	Page index to set after rebuilding

setPageIndex

Description

Set the current page index and update display states.

Definition

setPageIndex(index New)

Arguments

index New	page index

IngameMap

Description

In-game map display element.
-- This class is used to display the game map both in the HUD as well as in the in-game menu.
--@category GUI

new

Description

Create a new instance of IngameMap.

Definition

new(string hudAtlasPath, table inputDisplayManager)

Arguments

string	hudAtlasPath	Path to the HUD atlas texture
table	inputDisplayManager	InputDisplayManager reference

delete

Description

Delete this element and all of its components.

Definition

delete()

setFullscreen

Description

Set full-screen mode (for map overview) without affecting the mini-map state.

Definition

setFullscreen()

getHotspotIndex

Description

Get the index of a hotspot in the hotspots list.

Definition

getHotspotIndex()

cycleVisibleHotspot

Description

Get the next or previous visible hotspot in order from a current hotspot.

Definition

cycleVisibleHotspot(table currentHotspot, table categoriesHash, int direction)

Arguments

table	currentHotspot	Currently selected hotspot
table	categoriesHash	Table of valid hotspot categories to cycle, keys are categories
int	direction	1 for next, -1 for previous

getHeight

Description

Override from HUDElement.
Return zero height when turned off, because the map is only invisible.

Definition

getHeight()

getRequiredHeight

Description

Get required display height on screen including all auxiliary elements and texts.

Definition

getRequiredHeight()

setSize
Description

Set the map size.

Definition

setSize(float width, float height)

Arguments

float width Width of the map in screen space

float height Height of the map in screen space

setPosition
Description

Set the map's position.
This sets the actual map display's position to the given values. This elements background is offset accordingly.

Definition

setPosition(float posX, float posY)

Arguments

float posX New map X position in screen space

float posY New map Y position in screen space

setIsVisible
Description

Set the map's visibility (= active state).

Definition

setIsVisible()

onToggleMapSize
Description

Called when the input for map size toggle is pressed and then released.

Definition

onToggleMapSize()

registerInput
Description

Register map size toggle input event.

Definition

registerInput()

update
Description

Update the map's state.

Definition

update()

updatePlayerPosition
Description

Update the data about the player's current position.

Definition

updatePlayerPosition()

updatePlayerArrow

Description

Update the arrow indicating the player's current position.

Definition

updatePlayerArrow()

updateMapHeightZoomFactor

Description

Update map zoom and visibility state.

Definition

updateMapHeightZoomFactor()

Return Values

bool True if the left border of the map has been reached
bool True if the right border of the map has been reached
bool True if the top border of the map has been reached
bool True if the bottom border of the map has been reached

updateInputGlyphs

Description

Update input display glyphs with the current input context.

Definition

updateInputGlyphs()

updateMapUVs

Description

Update map overlay UVs based on the currently focused position, e.g. a player or hotspot.

Definition

updateMapUVs()

Return Values

boolean true if loading was successful else false
bool True if the left border of the map has been reached
bool True if the right border of the map has been reached
bool True if the top border of the map has been reached
bool True if the bottom border of the map has been reached

draw

Description

Draw the map as hud element

Definition

draw()

Return Values

boolean true if loading was successful else false

drawPlayerArrow

Description

Draw the current player's arrow.

Definition

drawPlayerArrow()

Return Values

table achievement achievement object

drawMapLabel

Description

Draw the map label on top of the map display.

Definition

drawMapLabel()

Return Values

table instance instance of object

drawMap

Description

Draw the map.

Definition

drawMap(float alpha, bool isStandalone)

Arguments

float alpha [optional] Map opacity value

bool isStandalone [optional] If true, will draw the map border and other elements.

Return Values

boolean true if loading was successful else false

drawPointsOfInterest

Description

Draw map hotspot and arrow elements.

Definition

drawPointsOfInterest()

Return Values

boolean true if loading was successful else false

drawOtherPlayerArrows

Description

Draw arrows for other players' positions.

Definition

drawOtherPlayerArrows()

Return Values

boolean true if added successful else false

drawEnterableArrows

Description

Draw arrows for enterable vehicles controlled by players.

Definition

drawEnterableArrows()

Return Values

table helper a random helper object

renderHotspots

Description

Draw all known hotspots on the map.

Definition

renderHotspots()

Return Values

integer helperIndex a random helper index

drawHotspot

Description

Draw a single hotspot on the map.

Definition

drawHotspot()

Return Values

table helper the helper object

drawPlayersCoordinates

Description

Draw the player's current coordinates as text.

Definition

drawPlayersCoordinates()

Return Values

table helper the helper object

drawLatencyToServer

Description

Draw current latency to server as text.

Definition

drawLatencyToServer()

Return Values

boolean success true if helper is marked else false

setScale

Description

Set this element's scale.

Definition

setScale(float uiScale)

Arguments

float uiScale Current UI scale applied to both width and height of elements

Return Values

integer numOfHelpers total number of helpers

storeScaledValues

Description

Store scaled positioning, size and offset values.

Definition

storeScaledValues()

Return Values

table instance instance of object

getBackgroundPosition

Description

Get the base position of the entire element.

Definition

getBackgroundPosition()

Return Values

boolean true if loading was successful else false

createBackground

Description

Create the empty background overlay.

Definition

createBackground()

Return Values

boolean true if loading was successful else false

createComponents

Description

Create required display components.

Definition

createComponents(string hudAtlasPath)

Arguments

string hudAtlasPath Path to the HUD texture atlas

Return Values

boolean true if loading was successful else false

createFrame

Description

Create the map frame.

Definition

createFrame()

Return Values

boolean true if added successful else false

createToggleMapSizeGlyph

Description

Create the input glyph for map size toggling.

Definition

createToggleMapSizeGlyph()

Return Values

table npc a random npc object

createPlayerMapArrow

Description

Create the map arrow for the player's position.

Definition

createPlayerMapArrow()

Return Values

integer npcIndex a random npc index

createOtherMapArrowOverlay

Description

Create an arrow overlay used for other player's and their vehicles in multiplayer.

Definition

createOtherMapArrowOverlay()

Return Values

table npc the npc object

IngameMapElement

Description

In-game map element.
-- Controls input on the map in the in-game menu with objectives, vehicles, etc. The actual map rendering is deferred to
the map component of the current mission. The map reference and terrain size must be set during mission
initialization via the setIngameMap() and setTerrainSize() methods.
--@category GUI
--@xmlConfig GuiElement#cursorId string ID of element to use as a cursor on the map.

addCursorDeadzone

Description

Add a dead zone wherein the map will not react to cursor inputs.
Used this to designate areas where other controls should receive cursor input which would otherwise be used up by
the map (e.g. in full-screen mode in the map overview screen in-game). The deadzones will also restrict cursor
movement.

Definition

addCursorDeadzone()

clearCursorDeadzones

Description

Clear cursor dead zones.

Definition

clearCursorDeadzones()

isCursorInDeadzones

Description

Check if a cursor position is within one of the stored deadzones.

Definition

isCursorInDeadzones()

mouseEvent

Description

Custom mouse event handling for the in-game map.
Directly handles zoom, click and drag events on the map. See input events and

IngameMapElement:checkAndResetMouse()
for the state checking code required to bypass player mouse input bindings.

Definition

mouseEvent()

updateBaseMapValues

Description

Update base map element values for displaying as an embedded UI element.

Definition

updateBaseMapValues()

setIngameMap

Description

Set the IngameMap reference to use for display.

Definition

setIngameMap()

setTerrainSize

Description

Set the current map's terrain size for map display.

Definition

setTerrainSize()

registerActionEvents

Description

Register non-GUI input action events.

Definition

registerActionEvents()

removeActionEvents

Description

Remove non-GUI input action events.

Definition

removeActionEvents()

onHorizontalCursorInput

Description

Event function for horizontal cursor input bound to
InputAction.AXIS_LOOK_LEFTRIGHT_VEHICLE.

Definition

onHorizontalCursorInput()

onVerticalCursorInput

Description

Event function for vertical cursor input bound to
InputAction.AXIS_LOOK_UPDOWN_VEHICLE.

Definition

onVerticalCursorInput()

onAccept

Description

Event function for gamepad cursor accept input bound to InputAction.INGAMEMAP_ACCEPT.

Definition

onAccept()

onZoomInput

Description

Event function for map zoom input bound to InputAction.AXIS_ACCELERATE_VEHICLE and InputAction.AXIS_BRAKE_VEHICLE.

Definition

onZoomInput(inputValue Zoom, direction Zoom)

Arguments

inputValue	Zoom input value
direction	Zoom input sign value, 1 for zoom in, -1 for zoom out

checkAndResetMouse

Description

Check if mouse input was active before a bound input was triggered and queue a reset of the mouse state for the next
frame.
Mouse input continuously sets the mouse input flag (self.useMouse) but does not receive any events when the mouse
is inert. Therefore we need to set and reset the state each frame to make sure we can seamlessly switch between mouse
and gamepad input on the map element while at the same time preventing any player bindings from interfering with the
custom mouse input logic of this class.

Definition

checkAndResetMouse()

InGameMenu

Description

In-Game Menu.
-- Displays the main in-game menu with several pages, depending on the game state and mode (e.g. tutorial or
multiplayer). This menu can be extended and modified by adding and removing pages. Default pages for the base game
are always loaded but can also be removed (effectively just disabled). Custom pages can be entirely new or modified
sub-classes of the default pages. See methods InGameMenu:addPage() and InGameMenu:removePage() for details.
--@category GUI
-- @field header Header panel

new

Description

Create a new instance of InGameMenu.

Definition

new(table target, table custom_mt, table messageCenter, table l10n, table savegameController, table inputManager, table fruitTypeManager, table fillTypeManager, table storeManager, table shopController, table shopConfigScreen, table placementScreen, bool isConsoleVersion)

Arguments

table target	Callback target
table custom_mt	Sub-class meta table
table messageCenter	MessageCenter reference
table l10n	I18N reference for text localization
table savegameController	SavegameController reference for savegame persistence
table inputManager	InputBinding reference
table fruitTypeManager	FruitTypeManager reference for fruit type information access
table fillTypeManager	FillTypeManager reference for fill type information access
table storeManager	StoreManager reference
table shopController	ShopController reference
table shopConfigScreen	ShopConfigScreen reference
table placementScreen	PlacementScreen reference
bool isConsoleVersion	If true, the game is running on a console

Return Values

table instance instance of object

setMode

Description

Set the menu mode.
Switches the menu between pause menu mode or shop mode.

Definition

setMode(int menuMode)

Arguments

int menuMode One of [InGameMenu.MODE_MENU | InGameMenu.MODE_SHOP]

Return Values

table instance instance of object

setInGameMap

Description

Set the in-game map component to use by pages.

Definition

setInGameMap()

Return Values

table brandColor brandColor object

setTerrainSize

Description

Set the current terrain size to use by pages.

Definition

setTerrainSize()

Return Values

table farmhouse or nil

setMissionFruitTypes

Description

Set the known fruit types when loaded for the mission.

Definition

setMissionFruitTypes()

Return Values

integer spawnpoint node or the career spawnpoint node.

setSellingStations

Description

Set the current list of selling stations for displaying.

Definition

setSellingStations()

Return Values

integer Camera or 0 if no farmhouse.

setAccessibleVehicles

Description

Set the current list of accessible vehicles for displaying.

Definition

setAccessibleVehicles()

Return Values

bool True if the user is a manager of this farm, false otherwise

setBanStorage

Description

Set the reference to the ban storage.

Definition

setBanStorage()

Return Values

table Permission hash table {permission=<hasPermission>}

setConnectedUsers

Description

Set the current list of connected users for displaying.

Definition

setConnectedUsers()

Return Values

float Account balance

setClient

Description

Set the network client reference.

Definition

setClient()

Return Values

float

setServer
Description

Set the network server reference.

Definition

setServer()

Return Values

table instance instance of object

updateHasMasterRights
Description

Update master rights status.

Definition

updateHasMasterRights()

Return Values

boolean true if loading was successful else false

updateGarageItems
Description

Update garage items display data.

Definition

updateGarageItems()

Return Values

boolean true if loading was successful else false

onLoadMapFinished
Description

Called when the mission is fully loaded.
Used for late initialization of UI components which rely on mission information.

Definition

onLoadMapFinished()

Return Values

table data Data for the menu

initializePausePages
Description

Initialize pages for the pause mode.

Definition

initializePausePages()

Return Values

table instance Instance of object

initializeShopPages
Description

Initialize pages for the shop mode.

Definition

initializeShopPages()

Return Values

boolean true if loading was successful else false

setupMenuPages

Description

Set up displayed menu pages and their tabs.

Definition

setupMenuPages()

Return Values

table instance instance of object

setupMenuButtonInfo

Description

Define default properties and retrieval collections for menu buttons.

Definition

setupMenuButtonInfo()

Return Values

boolean true if loading was successful else false

addPageTab

Description

Add a page tab in the menu header.
Call this synchronously with InGameMenu:registerPage() to ensure a correct order of pages and tabs.

Definition

addPageTab()

Return Values

fruit type index to be planted

setPageTabEnabled

Description

Set enabled state of a page tab in the header.

Definition

setPageTabEnabled()

Return Values

float sprayFactor the spray factor of the given field

rebuildTabList

Description

Rebuild page tab list in order.

Definition

rebuildTabList()

Return Values

float plowFactor the plow factor of the given field

setEnvironment

Description

Set environment reference on loading.

Definition

setEnvironment()

Return Values

float plowFactor the lime factor of the given field

setMissionInfo

Description

Set mission info data on loading.

Definition

setMissionInfo()

Return Values

float plowFactor the weed factor of the given field

setPlayerFarm

Description

Set the player's current farm reference.

Definition

setPlayerFarm()

Return Values

float area area found

float totalArea total area checked

setPlayer

Description

Set the reference to the current player.

Definition

setPlayer()

setCurrentUserId

Description

Set the current user ID.

Definition

setCurrentUserId()

Return Values

table instance instance of object

setManureTriggers

Description

Set manure triggers of the current map/mission.

Definition

setManureTriggers()

Return Values

boolean true if loading was successful else false

setHusbandries

Description

Set the reference to the current husbandries collection.

Definition

setHusbandries()

Return Values

boolean true if loading was successful else false

leaveCurrentGame

Description

Reset menu state and go back to the main menu.

Definition

leaveCurrentGame()

Return Values

List of vehicles. Each element is a table with filanema and configuration properties.

Reward

exitMenu

Description

Exit the menu if allowed.

Definition

exitMenu()

exitMenuFromConfig

Description

Exit the menu from the shop configuration screen.

Definition

exitMenuFromConfig()

reset

Description

Reset menu state (and all pages).

Definition

reset()

Return Values

float multiplier harvest multiplier

onOpen

Description

Handle in-game menu opening event.

Definition

onOpen()

Return Values

table instance of the doghouse

onClose

Description

Handle in-game menu closing event.

Definition

onClose()

Return Values

string Vehicle display name

onButtonSaveGame

Description

Button function for saving the game.

Definition

onButtonSaveGame()

Return Values

mixed value Value of the setting. The type depends on the setting

onButtonBack

Description

Button function for backing out of the menu.

Definition

onButtonBack()

Return Values

boolean successful Returns true, if the setting was changed

onButtonQuit

Description

Button function for quitting the game to the main menu.

Definition

onButtonQuit()

Return Values

table instance instance of object

onButtonGarage

Description

Button function for switching to the garage view.

Definition

onButtonGarage()

Return Values

table self instance

onButtonShop

Description

Button function for switching to the shop view from the garage.

Definition

onButtonShop()

Return Values

True if the element and all of its children could be set up with the given values, false otherwise.

onButtonRepair

Description

Button function for repairing an owned vehicle in the garage.

Definition

onButtonRepair()

Return Values

True if the input event has been consumed, false otherwise

onYesNoRepairDialog

Description

Handle confirmation of vehicle repairing.

Definition

onYesNoRepairDialog()

Return Values

Closest point x, y

onVehicleRepairEvent

Description

Handle a local vehicle repaired event.
Updates the garage view buttons if currently viewing the repaired vehicle.

Definition

onVehicleRepairEvent()

Return Values

Next GUI element in given direction which can be linked, actual scanning direction used (may change in wrap around scenarios)

onButtonAcceptItem

Description

Button function for explicit clicking of item details action buttons.

Definition

onButtonAcceptItem()

Return Values

Focus

onButtonSwitchOwnedLeased

Description

Button function for switching owned / leased items in the garage.

Definition

onButtonSwitchOwnedLeased()

Return Values

True if focus has changed, false otherwise

startPlacementMode

Description

Enter placement mode.

Definition

startPlacementMode()

Return Values

True if navigation in given direction is locked

startSavingGameDisplay

Description

Enter saving mode.

Definition

startSavingGameDisplay()

Return Values

Root GuiElement instance of loaded view or nil if the definition XML file could not be loaded.

updatePages

Description

Update page enabled states.

Definition

updatePages()

Return Values

Cloned FrameElement instance or frameRefElement if resolution failed.

updatePageTabDisplay

Description

Update page tabs display after any page changes.

Definition

updatePageTabDisplay()

Return Values

Root GuiElement of screen or nil if the name did not match any known screen.

clearMenuButtonActions

Description

Clear menu button actions, events and callbacks.

Definition

clearMenuButtonActions()

Return Values

Root GuiElement of dialog or nil if the name did not match any known dialog.

assignMenuButtonInfo

Description

Assign menu button information to the in-game menu buttons.

Definition

assignMenuButtonInfo()

Return Values

True if any control has consumed the action event

setPageSelectorTitles

Description

Get page titles from currently visible pages and apply to the selector element.

Definition

setPageSelectorTitles()

Return Values

table ScreenElement descendant instance of the given class or nil if no such instance was registered

setShopDetailMode

Description

Switch to or from shop detail mode.
This can switch to a category's item list, the owned object or leased object overview by passing in the corresponding
detail page.

Definition

setShopDetailMode()

Return Values

table Root GuiElement instance of target screen

update

Description

Update the menu state each frame.
This uses a state machine approach for the game saving process.

Definition

update()

Return Values

table The GuiElement instance given in the guiElement parameter

openFinancesScreen

Description

Directly switch to the finances screen.

Definition

openFinancesScreen()

Return Values

New

openFarmsScreen

Description

Directly switch to the farms screen.

Definition

openFarmsScreen()

Return Values

table with added image data

setMasterServerConnectionFailed

Description

Notify the menu that the master server could not be connected to.

Definition

setMasterServerConnectionFailed()

Return Values

Color as {red, green, blue, alpha} with all values in the range of [0, 1]

setMasterUserLocal

Description

Set the menu state to master user.
Called by the mission if the current player is a master user.

Definition

setMasterUserLocal()

Return Values

UV coordinates as {u1, v1, u2, v2, u3, v3, u4x, v4} with all values in the range of [0, 1]

showConfigurationScreen

Description

Show the vehicle configuration screen for a given store item.

Definition

showConfigurationScreen()

Return Values

New GuiProfile instance

inputEvent

Description

Custom input handling to check shop toggle button when in shop mode.

Definition

inputEvent()

Return Values

True if profile values could be loaded, false otherwise.

onMenuActionClick

Description

Handle a menu action click by calling one of the menu button callbacks.

Definition

onMenuActionClick()

Return Values

int Camera node ID (view point, child of camera base node)

int Camera base node ID (view target, parent of camera node)

bool True if no callback was present and no action was taken, false otherwise

onClickOk

Description

Handle menu confirmation input event.

Definition

onClickOk()

onClickBack

Description

Handle menu back input event.

Definition

onClickBack()

Return Values

Camera X world space position

Camera Z world space position

Camera Y rotation in radians

onClickCancel

Description

Handle menu cancel input event.
Bound to quite the game to the main menu.

Definition

onClickCancel()

onClickActivate

Description

Handle menu active input event.
Bound to save the game.

Definition

onClickActivate()

onMoneyChanged

Description

Handle a balance change in game.

Definition

onMoneyChanged()

onSlotUsageChanged

Description

Handle a change of the current slot usage.

Definition

onSlotUsageChanged()

Return Values

x direction movement [-1, 1]

Z direction movement [-1, 1]

onSelectItemBuyDetail

Description

Handle selection of a detail item element when in buy mode.

Definition

onSelectItemBuyDetail()

onSelectItemSellDetail

Description

Handle selection of a detail item element in the garage.

Definition

onSelectItemSellDetail()

Return Values

array of normalized values

onYesNoRestartTutorial

Description

Handle tutorial restart confirmation dialog response.

Definition

onYesNoRestartTutorial()

Return Values

array of the 2 converted values as numbers: {value1, value2}

onYesNoEnd

Description

Server end game confirmation dialog response callback.

Definition

onYesNoEnd()

Return Values

array of the 4 converted values as numbers: {value1, value2, value3, value4}

onClickPageSelection

Description

Handle activation of page selection.

Definition

onClickPageSelection()

Return Values

array of the 4 converted values as numbers: {red, green, blue, alpha}

onPagePrevious

Description

Handle previous page event.

Definition

onPagePrevious()

Return Values

array of the UV coordinates as {u1, v1, u2, v2, u3, v3, u4, v4}

onPageNext

Description

Handle next page event.

Definition

onPageNext()

Return Values

bool True if overlay generation has started, false if generation is already in progress or an invalid overlay type has been provided

onPageChange

Description

Handle changing to another menu page.

Definition

onPageChange()

Return Values

bool True if overlay generation has started, false if generation is already in progress or an invalid overlay type has been provided

updateButtonsPanel

Description

Update the buttons panel when a given page is visible.

Definition

updateButtonsPanel()

Return Values

bool True if overlay generation has started, false if generation is already in progress or an invalid overlay type has been provided

getPageButtonInfo

Description

Get button actions and display information for a given menu page.

Definition

getPageButtonInfo()

Return Values

bool True if overlay generation has started, false if generation is already in progress or an invalid overlay type has been provided

onPageUpdate

Description

Handle a page being disabled.

Definition

onPageUpdate()

Return Values

array of display information, {i={colors={true=[{r,g,b,a} colorblind], false=[{r,g,b,a} default], iconFilename=path, iconUVs={u1, v1, u2, v2, u3, v3, u4, v4}, description=text, fruitTypeIndex=index}}

onConnectionFailedDialogClick

Description

Handle saving game confirmation when losing master server connection.

Definition

onConnectionFailedDialogClick()

Return Values

array of display information, {i={colors={true={i={r,g,b,a}}, false={i={r,g,b,a}}}, description=text}}

onVehiclesChanged

Description

Called from the mission controller when vehicles are modified.

Definition

onVehiclesChanged()

Return Values

array of display information, {i={colors={true={i={r,g,b,a}}, false={i={r,g,b,a}}}, description=text}}

onClickBrand

Description

Handle clicking on a brand in shop mode.

Definition

onClickBrand()

Return Values

float X position in screen space

float Y position in screen space

onClickItemCategory

Description

Handle clicking on an item category in shop mode.

Definition

onClickItemCategory()

notifyValidateSavegameList

Description

Notify the player when validating the savegame list.
Will prompt to select a storage device or inform that the chosen device has no more space available.

Definition

notifyValidateSavegameList()

Return Values

float Width scale factor

float Height scale factor

notifyStartSaving

Description

Notify the player that the game is saving, block input via dialog until the operation finishes.

Definition

notifyStartSaving()

notifySaveComplete

Description

Notify the menu when a save completes successfully so we can set the saving state flag.

Definition

notifySaveComplete()

Return Values

table SettingsModel instance

notifySavegameNotSaved

Description

Notify the player that the savegame could not be saved.

Definition

notifySavegameNotSaved()

Return Values

table Currently active (changed) settings value

notifyOverwriteSavegame

Description

Prompt the player to confirm overwriting an existing savegame.

Definition

notifyOverwriteSavegame()

Return Values

bool True if any setting has been changed, false otherwise

notifySaveFailedNoSpace

Description

Notify the player that saving failed because the current device has no more space, ask to select another device.

Definition

notifySaveFailedNoSpace()

Return Values

bool True if any setting has been changed, false otherwise

registerPage

Description

Register a page frame element in the menu.
This does not add the page to the paging component of the menu.

Definition

registerPage(table pageFrameElement, int position, function enablingPredicateFunction)

Arguments

table	pageFrameElement	Page FrameElement instance
int	position	[optional] Page position index in menu
function	enablingPredicateFunction	[optional] A function which returns the current enabling state of the page at any time. If the function returns true, the page should be enabled. If no argument is given, the page is always enabled.

Return Values

bool True if no callback was present and no action was taken, false otherwise

unregisterPage

Description

Unregister a page frame element identified by class from the menu.
This does not remove the page from the paging component of the menu or the corresponding page tab from the header.

Definition

unregisterPage(table pageFrameClass)

Arguments

table pageFrameClass FrameElement descendant class of a page which was previously registered

Return Values

bool	True	if there was a page of the given class and it was unregistered
table	Unregistered page controller instance or nil	
table	Unregistered page root GuiElement instance or nil	
table	Unregistered page tab ListElement instance of nil	
bool	True	if there was a page of the given class and it was unregistered
table	Unregistered page controller instance or nil	
table	Unregistered page root GuiElement instance or nil	
table	Unregistered page tab ListElement instance of nil	

addPage

Description

Add a page frame to be displayed in the menu at runtime.
The page will be part of the in-game menu until restarting the game or removing the page again.
-- @usage
local frameController = CustomFrameElement:new(...) -- create controller instance
g_gui:loadGui("<frame layout XML path>", "<frame name>", frameController, true) -- load UI components, attach controller
local enablePage = function() -- define predicate function which returns true when the page should be enabled
return self.example.isSkyBlue
end

self:addPage(frameController, position, enablePage) -- add page to menu (self)
-- @param table pageFrameElement FrameElement instance which is used as a page.

Definition

addPage(int position, string tabIconFilename, table tabIconUVs, function enablingPredicateFunction)

Arguments

int	position	Position index of added page, starting from left at 1 going to right.
string	tabIconFilename	Path to the texture file which contains the icon for the page tab in the header
table	tabIconUVs	UV array for the tab icon. Format: {x, y, width, height} in pixels on the texture.
function	enablingPredicateFunction	[optional] A function which returns the current enabling state of the page at any time. If the function returns true, the page should be enabled. If no argument is given, the page is always enabled.

removePage

Description

Remove a page from the menu at runtime by its class type.
The removed page is also deleted, including all of its children. Note that this method removes the page for an entire
game run, because the UI is loaded on game start. If you only need to disable a page, use InGameMenu:setPageEnabled()
instead.
-- The method will not remove game default pages, but only disable them.
-- @usage
self:removePage(CustomFrameElement) -- where CustomFrameElement is a sub-class of FrameElement
-- @param table pageFrameClass Class table of a FrameElement sub-class

Definition

removePage()

setPageEnabled

Description

Set the enabled state of a page identified by its controller class type.
This will also set the controller's state, so it can react to being enabled/disabled. The setting will persist
through calls to InGameMenu:reset() and must be reverted manually, if necessary.

Definition

setPageEnabled(table pageFrameClass, bool isEnabled)

Arguments

table	pageFrameClass	Class table of a FrameElement sub-class
bool	isEnabled	True for enabled, false for disabled

updateGarageButtonInfo

Description

Update the button information for the garage.

Definition

updateGarageButtonInfo()

Return Values

{name="internalName", displayName="displayName",	inputTexts={1=text1, 2=text2, ...}, inputBindings={1=binding1, 2=binding2, ...}, positiveInput=[true|false]}

makeSelfCallback

Description

Make a callback which encloses the self reference and handles arbitrary arguments afterwards.

Definition

makeSelfCallback()

Return Values

List of DisplayActionBinding instances

InGameMenuAnimalsFrame

Description

In-game menu animals statistics frame.
-- Displays information for all owned animal pens and horses.
--@category GUI

new

Description

Create a new InGameMenuAnimalsFrame instance.

Definition

new(table subclass_mt, table messageCenter, table l10n, table animalManager, table animalFoodManager, table fillTypeManager)

Arguments

table subclass_mt	[optional] Meta table of subclass
table messageCenter	MessageCenter reference for local message subscriptions
table l10n	I18N reference for localization
table animalManager	AnimalManager reference for animal type and information resolution
table animalFoodManager	AnimalFoodManager reference for food mixture information
table fillTypeManager	FillTypeManager reference for fill type resolution

Return Values

Display	
table	InGameMenuAnimalsFrame instance

initialize

Description

Late initialization.

Definition

initialize()

Return Values

True if the controller starts listening for input, false otherwise (also if it is already listening!)

setHusbandries

Description

Set the reference to the husbandries collection.
The husbandries are defined as {<husbandry ID> = AnimalHusbandry}

Definition

setHusbandries()

Return Values

bool True if an actual binding was deleted, false otherwise

buildLivestockListItem

Description

Build a livestock list item for a husbandry.

Definition

buildLivestockListItem()

Return Values

True if a binding change was made, false otherwise

buildHorseListItems

Description

Build animal list items for all horses in a horse husbandry.

Definition

buildHorseListItems()

Return Values

True if a binding change was made, false otherwise

rebuildAnimalList

Description

Rebuild the animal list from husbandry data.

Definition

rebuildAnimalList()

Return Values

True if a binding change was made, false otherwise

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

True if binding has been assigned; reference to first colliding binding or nil

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

table DirectSellDialog instance

setStatusBarValue

Description

Modify a status bar's value, applying the correct size and color as needed.

Definition

setStatusBarValue(table statusBarElement, float value, float startOffset, table profiles, float overrideStatusValue)

Arguments

table	statusBarElement	BitmapElement instance which receives a new size and profile depending on the value
float	value	Fractional value in the range of [0, 1]
float	startOffset	Start offset value in the range of [0, 1], value will be displayed from there on until 1.
table	profiles	Large or small status bar profile table, either InGameMenuAnimalsFrame.PROFILE.STATUS_BAR_LARGE or InGameMenuAnimalsFrame.PROFILE.STATUS_BAR_SMALL
float	overrideStatusValue	[optional] If provided, tests this value against the color thresholds instead of the given status bar value

Return Values

Flows and cells as nested tables: [flowIndex][cell index] = cell data {element, flowSize, lateralSize}

displayRequirement

Description

Display a husbandry requirements row with the given data.
If no label text is specified, the row will be hidden.

Definition

displayRequirement()

Return Values

List of lateral flow sizes, total lateral sizes (sum of lateral flow sizes), maximum flow size in direction of flow

displayCondition

Description

Display a husbandry conditions row with the given data.
If no label text is specified, the row will be hidden.

Definition

displayCondition()

Return Values

Layout X starting offset, layout Y starting offset, Layout X direction {-1|1}, Layout Y direction {-1|1}

sumFillLevelInfos

Description

Sum up the values of an array of animal husbandry fill level info tables.
The fill level info tables have the form of {fillType=fillType, fillLevel=fillLevel, capacity=capacity}. The method
assumes that the array contains only different instances of the same fill type.

Definition

sumFillLevelInfos()

Return Values

element X offset, element Y offset

updateLivestockHusbandryProductionDisplay

Description

Update production display for a livestock husbandry.
WIP, no assets ready for testing!

Definition

updateLivestockHusbandryProductionDisplay()

Return Values

which should receive focus instead

updateHusbandryConditionsDisplay

Description

Update base conditions (dirt, water, straw) display for a husbandry.

Definition

updateHusbandryConditionsDisplay()

Return Values

float Modified X offset

float Modified Y offset

updateHusbandryFoodDisplay

Description

Update food levels and capacities display for a husbandry.

Definition

updateHusbandryFoodDisplay()

displayHorse

Description

Display data of a single horse in the detail view.

Definition

displayHorse()

Return Values

True if the input event has been handled, false otherwise.

displayLivestock

Description

Display data of a livestock husbandry (pigs, cows, etc.) in the detail view.

Definition

displayLivestock()

Return Values

True if the keyboard input has been processed by this element.

updateMenuButtons

Description

Update contextual menu buttons.

Definition

updateMenuButtons()

Return Values

Actual element to focus.

renameCurrentHorse

Description

Rename the currently selected horse if the new name has been confirmed.

Definition

renameCurrentHorse()

Return Values

List of this element's descendants in depth-first order with contiguous numeric indices.

getFoodDescription

Description

Get a formatted food description text for an animal type.

Definition

getFoodDescription()

Return Values

First matching descendant element in depth-first order or nil, if no element matched the predicate function

onButtonRename

Description

Handle "rename" button activation when a horse is selected.

Definition

onButtonRename()

Return Values

element or nil

onListSelectionChanged

Description

Handle animal list selection changes.

Definition

onListSelectionChanged()

Return Values

element or nil

InGameMenuContractsFrame

Description

Exposed controls usable as fields in this frame, identified in configuration.

new

Description

Create a new InGameMenuContractsFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional] Meta table of subclass

Return Values

parent element or full screen borders in an array: {minX, minY, maxX, maxY}

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

element borders in an array: {minX, minY, maxX, maxY}

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

float X aspect scale factor

float Y aspect scale factor

setButtonsForState

Description

Update the state of the frame buttons.

Definition

setButtonsForState()

updateList

Description

Request a mission list update, but remember the position of the selection
If the mission is still in the list, the selection stays

Definition

updateList()

Return Values

index or nil, field name

updateDetailContents

Description

Update the contents of the contract detail screen (context sensitive)

Definition

updateDetailContents()

Return Values

Actual element to focus.

updateFarmersBox

Description

Update the content of the box with farmer information, using the field.

Definition

updateFarmersBox()

Return Values

bool True if the page changed

InGameMenuFinancesFrame

Description

Financial overview of the player's farm for the in-game menu.
-- Displays current balance and loan situation as well as past and present incomes and

expenses.
--@category GUI

new

Description

Create a new InGameMenuFinancesFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional] Meta table of subclass

Return Values

Page

initialize

Description

Initialize the finances frame after component creation.

Definition

initialize()

Return Values

Page container GuiElement instance

setClient

Description

Set networking client reference.

Definition

setClient()

Return Values

int Page index

setEnvironment

Description

Set the mission environment reference.

Definition

setEnvironment()

Return Values

int Page mapping index

setPlayerFarm

Description

Set the player's current farm reference.

Definition

setPlayerFarm()

Return Values

bool True if the event was not used, false if it was used.

setHasMasterRights

Description

Set the player's master rights status.

Definition

setHasMasterRights()

Return Values

bool True if the event was not used, false if it was used.

getPastDays

Description

Get the array of days before today counting back to the number of displayed past days.
The method resolves days to display labels which can be used directly.

Definition

getPastDays()

Return Values

bool True if the event was not used, false if it was used.

setupFinancesTable

Description

Late setup of finances table called on initialization.

Definition

setupFinancesTable()

Return Values

bool True if the event was not used, false if it was used.

updateBalance

Description

Update the current balance display text.

Definition

updateBalance()

Return Values

bool True if the event was not used, false if it was used.

updateLoan

Description

Update the loan display text.

Definition

updateLoan()

Return Values

bool True if the event was not used, false if it was used.

updateDayTotals

Description

Update the expenses/incomes total for all displayed days.

Definition

updateDayTotals()

Return Values

number of list items in data source

updateFinancesFooter

Description

Update statically positioned totals display.

Definition

updateFinancesFooter()

Return Values

table Array of button info as {i={inputAction=<action name>, text=<optional display text>, callback=<optional callback>}}

updateFinancesTable

Description

Update table display data.

Definition

updateFinancesTable()

Return Values

DataCell or nil if not found.

updateFinances

Description

Update finances display.

Definition

updateFinances()

Return Values

DataCell or nil if not found.

updateFinancesLoanButtons

Description

Update loan button active states.

Definition

updateFinancesLoanButtons()

Return Values

List of SortCell for the requested column name

updateMoneyUnit

Description

Update the current money unit.
Also applies the unit change to the borrow/repay button labels.

Definition

updateMoneyUnit()

Return Values

index of selected row

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

dataRow instance or nil, selected data index or 0

buildDataRow

Description

Build a table data row for a given financial statistic.

Definition

buildDataRow()

Return Values

dataRow instance or nil (element not part of table row)

onButtonBorrow

Description

Handle borrow money action.

Definition

onButtonBorrow()

Return Values

number of data view rows

onButtonRepay

Description

Handle repay money action.

Definition

onButtonRepay()

Return Values

Actual element to focus.

InGameMenuFrameElement

Description

Base class for frame elements for the in-game menu.
--@category GUI

new

Description

Create a new InGameMenuFrameElement instance.

Definition

new()

initialize

Description

Late initialization of a menu frame.
Override in sub-classes.

Definition

initialize()

getHasCustomMenuButtons
Description

Check if this menu frame requires menu button customization.

Definition

getHasCustomMenuButtons()

getMenuButtonInfo
Description

Get custom menu button information.

Definition

getMenuButtonInfo()

Return Values

table Array	of button info as {i={inputAction=<action name>, text=<optional display text>, callback=<optional callback>}}

setMenuButtonInfo
Description

Set custom menu button information.

Definition

setMenuButtonInfo(table menuButtonInfo)

Arguments

table menuButtonInfo	Array of button info as {i={inputAction=<action name>, text=<optional display text>, callback=<optional callback>}} or nil to reset.

setMenuButtonInfoDirty
Description

Set the menu button info dirty flag which causes the menu to update the buttons from this element's information.

Definition

setMenuButtonInfoDirty()

isMenuButtonInfoDirty
Description

Get the menu button info dirty state (has changed).

Definition

isMenuButtonInfoDirty()

clearMenuButtonInfoDirty
Description

Clear menu button dirty flag.

Definition

clearMenuButtonInfoDirty()

getMainElementSize
Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

getMainElementPosition
Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

requestClose
Description

Request to close the frame.
Frames can contain logic (e.g. saving pending changes) which should be handled before closing. Use this method in
sub-classes request closing the frame so it can wrap up first. If a callback is provided and the initial request
could not close the frame, the callback will be called as soon as the frame can be closed.

Definition

requestClose()

onFrameOpen
Description

Called when this frame is opened by its container.

Definition

onFrameOpen()

onFrameClose
Description

Called when this frame is closed by its container.

Definition

onFrameClose()

InGameMenuGameSettingsFrame
Description

Current savegame settings page for the in-game menu.
--@category GUI

new
Description

Create a new InGameMenuGameSettingsFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional] Meta table of subclass

Return Values

The new sorting order

initialize
Description

Initialize the game settings frame with concrete page references.
This must be called after loading, since page frames are cloned and references cannot be injected in the constructor.

Definition

initialize()

Return Values

table AchievementMessage instance

setMissionInfo

Description

Set the current mission's info.
Required to get current settings and to modify game name.

Definition

setMissionInfo()

Return Values

float X position in screen space

float Y position in screen space

setManureTriggers

Description

Set the currently known manure triggers in the game for helper manure refill settings.

Definition

setManureTriggers()

setHasMasterRights

Description

Set master rights status of the current game instance / player.

Definition

setHasMasterRights()

Return Values

table ChatWindow instance

updateGameSettings

Description

Update settings display with values from the current mission info / savegame.

Definition

updateGameSettings()

Return Values

float X position in screen space

float Y position in screen space

updatePauseButtonState

Description

Update the pause buttons visuals for pausing/unpausing based on the game state.

Definition

updatePauseButtonState()

assignStaticTexts

Description

Assign static option settings texts.

Definition

assignStaticTexts()

Return Values

int Display row index of the newly added custom text or 0 if it could not be added

assignTimeScaleTexts

Description

Assign time scale setting texts.

Definition

assignTimeScaleTexts()

Return Values

float X position in screen space

float Y position in screen space

assignDirtTexts

Description

Assign dirt setting texts.

Definition

assignDirtTexts()

assignPlantGrowthTexts

Description

Assign plant growth texts.

Definition

assignPlantGrowthTexts()

Return Values

float Game info display width of visible elements in screen space

assignAutoSaveTexts

Description

Assign auto save texts.

Definition

assignAutoSaveTexts()

Return Values

table Weather icon HUDElement instance

assignDynamicTexts

Description

Assign game state dependent setting texts.

Definition

assignDynamicTexts()

Return Values

table Temperature icon HUDElement instance

updateToolTipBoxVisibility

Description

Update visibility of tool tip box, only show when there is text to display.

Definition

updateToolTipBoxVisibility()

Return Values

table Clock hand HUDElement instance

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

table Time scale arrow icon HUDElement instance

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

table GamePausedDisplay instance

onEnterPressedSavegameName

Description

Handle accepting savegame name input.

Definition

onEnterPressedSavegameName()

Return Values

bool If true, the HUD is currently visible.

onToolTipBoxTextChanged

Description

Handle changing of the tool tip text.

Definition

onToolTipBoxTextChanged()

Return Values

table HUDDisplayElement instance

InGameMenuGeneralSettingsFrame

Description

General game settings page for the in-game menu.
--@category GUI

new

Description

Create a new InGameMenuGeneralSettingsFrame instance.

Definition

new(table subclass_mt, table settingsModel)

Arguments

table subclass_mt [optional] Meta table of subclass

table settingsModel SettingsModel reference which handles settings display state across the UI

Return Values

float Screen space X translation
float Screen space Y translation

updateGeneralSettings

Description

Update display values from settings.

Definition

updateGeneralSettings()

updateToolTipBoxVisibility

Description

Update visibility of tool tip box, only show when there is text to display.

Definition

updateToolTipBoxVisibility()

Return Values

table HUDElement instance

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

float Pivot x position offset from element position in screen space
float Pivot y position offset from element position in screen space

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

onClickCheckbox

Description

Handle clicks on a check box.

Definition

onClickCheckbox()

Return Values

float X position in screen space
float Y position in screen space

onClickMultiOption

Description

Handle clicks on a multi option element.

Definition

onClickMultiOption()

onClickNativeHelp

Description

Handle a button click on the native help button which is only visible and active on XBOX.

Definition

onClickNativeHelp()

Return Values

width scale factor

height scale factor

onToolTipBoxTextChanged

Description

Handle changing of the tool tip text.

Definition

onToolTipBoxTextChanged()

InGameMenuHelpFrame

Description

Exposed controls usable as fields in this frame, identified in configuration.

new

Description

Create a new InGameMenuHelpFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional] Meta table of subclass

Return Values

float Red value

float Green value

float Blue value

float Alpha value

setMissionBaseDirectory

Description

Set the current mission's base directory to let this frame load images from that location.

Definition

setMissionBaseDirectory()

createList

Description

Set up UI elements for a given category index.

Definition

createList()

loadHelpLine

Description

Load help category texts.

Definition

loadHelpLine()

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

float Alpha value

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

table HUDPopupMessage instance

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

float Screen space X translation

float Screen space Y translation

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

onHelpLineListSelectionChanged

Description

Handle a change of row in the current category's help list.

Definition

onHelpLineListSelectionChanged()

Return Values

table HUDTextDisplay instance

InGameMenuMapFrame

Description

Map overview frame of the in-game menu.
-- Displays the current map with markers for points of interest and terrain overlays.
--@category GUI

new

Description

Create a new InGameMenuMapFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional] Meta table of subclass

Return Values

bool True if the left border of the map has been reached

bool True if the right border of the map has been reached

bool True if the top border of the map has been reached

bool True if the bottom border of the map has been reached

createInputGlyphs

Description

Create input help glyphs.

Definition

createInputGlyphs()

initialize

Description

Initialize map frame after GUI setup.

Definition

initialize()

onLoadMapFinished

Description

Called by InGameMenu when a map has finished loading.

Definition

onLoadMapFinished()

toggleMapInput

Description

Toggle map overview-specific input.
Make sure this is called exactly once for activation and deactivation each when the map frame is opened or closed.

Definition

toggleMapInput()

Return Values

bool True if the left border of the map has been reached

bool True if the right border of the map has been reached

bool True if the top border of the map has been reached

bool True if the bottom border of the map has been reached

disableAlternateBindings

Description

Disable alternate bindings for menu navigation.
This will disable some default bindings which interfere with camera controls (e.g. D-Pad on controller). Whenever any
input event is modified, this method must be called again afterwards.

Definition

disableAlternateBindings()

getContextBoxPositionAndOrientation

Description

Get the context box position and orientation for a given hotspot.

Definition

getContextBoxPositionAndOrientation()

Return Values

float	Screen	space X position
float	Screen	space Y position
string	Orientation	value, one of InGameMenuMapFrame.CONTEXT_BOX_ORIENTATION
float	Rotation	angle value

updateContextBoxPosition

Description

Update the position of the hotspot context box if it's active.

Definition

updateContextBoxPosition()

setInGameMap

Description

Set the IngameMap reference in this frame's required IngameMapElement for display.

Definition

setInGameMap()

Return Values

float X position in screen space after the last glyph

setTerrainSize

Description

Set the terrain size for use in the ingameMapElement.

Definition

setTerrainSize()

Return Values

float X position in screen space after the last glyph

setMissionFruitTypes

Description

Set the fruit types used in the current mission.

Definition

setMissionFruitTypes()

Return Values

float Screen space height used by the combo header (0 if invisible)

setClient

Description

Set the client connection object for event dispatching.

Definition

setClient()

Return Values

table Input help elements

float Screen space height used by the returned help elements

setPlayerFarm

Description

Set the current player's farm ID.

Definition

setPlayerFarm()

assignFilterData

Description

Assign filter data after map loading.

Definition

assignFilterData()

Return Values

int Maximum number of entries to show

assignCropTypeFilterData

Description

Assign display data for the crop type filters.

Definition

assignCropTypeFilterData()

Return Values

boolean isAllowed isAllowed

assignGroundStateFilterData

Description

Assign display data for the ground state filters (growth and soil state).

Definition

assignGroundStateFilterData()

Return Values

table Combo InputGlyphElement instance

assignGroundStateColors

Description

Assign colors for ground state (growth or soil) elements.

Definition

assignGroundStateColors()

Return Values

table Combo header HUDElement

resetUIDeadzones

Description

Reset the UI dead zones for the map.

Definition

resetUIDeadzones()

Return Values

float Display height in screen space

setStaticUIDeadzone

Description

Set a static UI dead zone for the map to make it ignore cursor/mouse input within that screen region.
This dead zone is always in effect, regardless of map mode.

Definition

setStaticUIDeadzone()

Return Values

float Modified input help panel drawing vertical offset

onOverviewOverlayFinished

Description

Called when the overview overlay is finished for display.

Definition

onOverviewOverlayFinished()

Return Values

table SideNotification instance

onFarmlandOverlayFinished

Description

Called when the farmland overlay is finished for display.

Definition

onFarmlandOverlayFinished()

Return Values

table SpeakerDisplay instance

generateOverviewOverlay

Description

(Re-)Generate the map overlay for crop types, growth and soil states.

Definition

generateOverviewOverlay()

Return Values

table Overlay instance

generateFarmlandOverlay

Description

(Re-)Generate the map overlay for farmlands.

Definition

generateFarmlandOverlay()

Return Values

table HUDElement instance

setFilterIconState

Description

Set the display state of a filter icon based on the current filter settings.

Definition

setFilterIconState()

Return Values

table TopNotification instance

toggleFarmlandsHotspotFilterSettings

Description

Toggle hotspot filter settings for the farmlands view mode.

Definition

toggleFarmlandsHotspotFilterSettings()

Return Values

float Screen space X translation

float Screen space Y translation

showContextInput

Description

Show input buttons based on the current map selection context.

Definition

showContextInput()

updateContextInputBarVisibility

Description

Update the context input bar's visibility.
Turns it invisible if there are no buttons visile, otherwise turns it visible.

Definition

updateContextInputBarVisibility()

Return Values

bool If true, the HUD extension should be drawn in the current frame.

showContextMarker

Description

Show marker button according to context.

Definition

showContextMarker()

Return Values

float Modified input help panel drawing vertical offset

showContextBox

Description

Show the context box for a selected hotspot.

Definition

showContextBox()

Return Values

table HUD extension instance or nil of no extension has been registered for the given specialization

hideContextBox

Description

Hide the hotspot context box.

Definition

hideContextBox()

Return Values

table Array	of overlay descriptions: {overlay=overlay, x=0, y=0, rotation=0, invertX=false, invisibleBorderRight=vehicle.schemaOverlay.invisibleBorderRight, invisibleBorderLeft=vehicle.schemaOverlay.invisibleBorderLeft}
float	Screen space height of root vehicle schema overlay

setMapSelectionItem

Description

Set the map selection to a hotspot.

Definition

setMapSelectionItem()

getHotspotData

Description

Get display data for a selectable map hotspot.

Definition

getHotspotData()

Return Values

float	Minimum	X position (left)
float	Maximum	X position (right)
string	Description	text
string	Display	image file path
table	Display	image UVs
table	Vehicle	instance if a vehicle has been selected, or nil otherwise

setColorBlindMode

Description

Set the current color blind mode.
Updates colors on buttons and overlays.

Definition

setColorBlindMode(bool isActive)

Arguments

bool isActive	If true, color blind mode is active

initializeFilterButtonState

Description

Initialize filter buttons with the current filter state.

Definition

initializeFilterButtonState()

Return Values

table	Schema Overlay instance

resetFarmlandSelection

Description

Reset farmland selection state.

Definition

resetFarmlandSelection()

Return Values

table InGameMenuAnimalsFrame instance

checkPlaceablesOnFarmland

Description

Check if there are any placeables owned by the player on a given farmland.

Definition

checkPlaceablesOnFarmland()

Return Values

float	Screen	space X position
float	Screen	space Y position
string	Orientation	value, one of InGameMenuMapFrame.CONTEXT_BOX_ORIENTATION
float	Rotation	angle value

onMoneyChanged

Description

Handle changes to farm balance values.

Definition

onMoneyChanged(int farmId, float newBalance)

Arguments

int	farmId	ID of farm whose current balance has changed
float	newBalance	New balance value of the given farm

onDrawPostIngameMap

Description

Called after the in-game map has been drawn.
Draws the mode-dependent state overlay on top of the map

Definition

onDrawPostIngameMap()

onClickMapOverviewSelector

Description

Handle clicking the map overview selector which switches map overlay context (crop types, growth states, soil states).

Definition

onClickMapOverviewSelector()

onFilterButtonSelect

Description

Handle filter button focus or highlight activation.

Definition

onFilterButtonSelect()

Return Values

string	Description	text
string	Display	image file path
table	Display	image UVs
table	Vehicle	instance if a vehicle has been selected, or nil otherwise

onFilterButtonUnselect

Description

Handle filter button focus or highlight deactivation.

Definition

onFilterButtonUnselect()

setFilterButtonDisplayEnabled

Description

Set the display state of a filter button.

Definition

setFilterButtonDisplayEnabled()

toggleFilter

Description

Toggle filter state on filter button click.

Definition

toggleFilter()

onClickCropFilter

Description

Handle activation of a crop type filter.

Definition

onClickCropFilter(table element, int fruitTypeIndex)

Arguments

table element Clicked button element

int fruitTypeIndex Fruit type index as valid in the current mission.

Return Values

int List index corresponding to the farm ID or 1 if it could not be found or was the spectator farm ID

onClickGrowthFilter

Description

Handle activation of a growth state filter.

Definition

onClickGrowthFilter(table element, int growthStateIndex)

Arguments

table element Clicked button element

int growthStateIndex Growth state index

Return Values

dataRow instance with prices data for the given selling point

onClickSoilFilter

Description

Handle activation of a soil state filter.

Definition

onClickSoilFilter(table element, int growthStateIndex)

Arguments

table element Clicked button element

int growthStateIndex Soil state index

Return Values

float Sorting value, see TableElement:setCustomSortFunction(...)

onClickHotspot

Description

Handle clicking / activating a map hotspot.

Definition

onClickHotspot()

Return Values

float Current	fill level >= 0 or a value < 0 if no storage exists for the requested fill type index
float Total	capacity for the fill type >= 0 or a value < 0 if no storage exists for the requested fill type index

onClickMap

Description

Handle clicking within the map.

Definition

onClickMap()

onFarmlandStateChanged

Description

Called from FarmlandManager whenever a significant farmland change (mainly ownership) occurs.

Definition

onFarmlandStateChanged()

Return Values

table TableElement.DataRow instance

onVehicleReset

Description

Handle ResetVehicleEvent local event.

Definition

onVehicleReset()

Return Values

dataRow instance with vehicle data for the given vehicle

onClickSwitchMapMode

Description

Handle activation of map mode switch button.

Definition

onClickSwitchMapMode()

Return Values

float Sorting value, see TableElement:setCustomSortFunction(...)

onYesNoReset

Description

Dialog confirmation callback for resetting a vehicle.

Definition

onYesNoReset()

Return Values

True if the server matches the current filter settings, false otherwise

notifyPause

Description

Notify this frame when pausing the game.

Definition

notifyPause()

Return Values

dataRow instance with server data

selectFirstHotspot

Description

Select the first visible hotspot on the map

Definition

selectFirstHotspot()

Return Values

table LandscapingScreen instance

updateInputGlyphTransform

Description

Fit an input glyph into its corresponding placeholder GuiElement instance.

Definition

updateInputGlyphTransform()

Return Values

table new LandscapingScreenController instance

updateInputGlyphs

Description

Update input glyphs when input context changes.

Definition

updateInputGlyphs()

Return Values

int Node ID of the actual indicator shape in the loaded indicator asset

int Node ID of the attached light source of the loaded indicator asset

onYesNoBuyFarmland

Description

Handle confirmation result of farmland buying dialog.

Definition

onYesNoBuyFarmland()

onYesNoSellFarmland

Description

Handle confirmation result of farmland selling dialog.

Definition

onYesNoSellFarmland()

Return Values

table categories list of categories

registerInput

Description

Register input actions.

Definition

registerInput()

Return Values

table PlacementScreen instance

onMenuActivate

Description

Menu activate event, bound to vehicle enter, place visit or buying farmlands.

Definition

onMenuActivate()

Return Values

True if the placement is valid, false otherwise

string Reason of being invalid

onMenuCancel

Description

Menu cancel event, bound to hotspot tagging, vehicle reset or selling farmlands.

Definition

onMenuCancel()

onSwitchVehicle

Description

Switch vehicle action event, bound to cycle visible hotspots.

Definition

onSwitchVehicle()

Return Values

table ShopConfigScreen instance

InGameMenuMultiplayerFarmsFrame

Description

Exposed controls usable as fields in this frame, identified in configuration.

new

Description

Create a new InGameMenuMultiplayerFarmsFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional] Meta table of subclass

Return Values

float Fuel capacity in liters

initialize

Description

Late initialization after frame cloning.

Definition

initialize()

Return Values

float Base price
float Upgrade price
bool True if there are changes

setCurrentUserId

Description

Set the reference to the current user.

Definition

setCurrentUserId()

setUsers

Description

Set the current users reference.

Definition

setUsers()

setPlayer

Description

Set the reference to the current player.

Definition

setPlayer()

Return Values

int Number of used config option elements

setPlayerFarm

Description

Set the reference to the current player's farm.

Definition

setPlayerFarm()

Return Values

table New collection of filtered owned items

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

table Array of brands in the form of {i={id = brand.index, iconFilename = brand.image, label = brand.title}}

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

table Array of vehicle categories in the form of {i={id = category.index, iconFilename = category.image, label = category.title}}

buildFarmListItems

Description

Build the farm list items from the known farms.

Definition

buildFarmListItems()

Return Values

table Array of tool categories in the form of {i={id = category.index, iconFilename = category.image, label = category.title}}

updateFarmList

Description

Update farm list.

Definition

updateFarmList()

Return Values

table Array of object categories in the form of {i={id = category.index, iconFilename = category.image, label = category.title}}

getListFarmIndex

Description

Get the list index for a given farm ID.

Definition

getListFarmIndex(int farmId)

Arguments

int farmId Farm ID

Return Values

table Array of placeable categories in the form of {i={id = category.index, iconFilename = category.image, label = category.title}}

int List index corresponding to the farm ID or 1 if it could not be found or was the spectator farm ID

updateFarmBalance

Description

Update a farm's balance display.

Definition

updateFarmBalance()

Return Values

int Next usable attribute slot index after these fill types

updateFarmPlayers

Description

Update displayed player names on a farm.

Definition

updateFarmPlayers()

Return Values

int Number of attributes used for text data

updateMenuButtons

Description

Update context menu buttons.

Definition

updateMenuButtons()

Return Values

table instance Instance of object

joinFarm

Description

Let the current player join a given farm.
The player will leave their current farm (can be spectator farm) and join the new farm if possible. This sends
a PlayerSetFarmEvent which will check any farm password on the server side. In response, a PlayerSetFarmAnswerEvent
is sent back. The method will try using a previously recorded farm password if one has been received during a join
event.

Definition

joinFarm()

Return Values

boolean success success

leaveFarm

Description

Let the current player leave their farm.
The player will join the spectator farm and be able to choose a new farm to join.

Definition

leaveFarm()

Return Values

deleteFarm

Description

Delete a given farm
The current player must have admin privileges on the server and the farm must be empty.

Definition

deleteFarm()

Return Values

float

float

float

float

float

float

editFarm
Description

Show a dialog to edit farm properties.
The current player must have farm manager or administrator privileges on the server.

Definition

editFarm()

createFarm
Description

Show a dialog to create a new farm
The current player must have administrator privileges on the server.

Definition

createFarm()

onPlayerSetFarmAnswer
Description

Handle an answer to PlayerSetFarmEvent.

Definition

onPlayerSetFarmAnswer(int answerState, int farmId, string password)

Arguments

int	answerState	PlayerSetFarmAnswerEvent.STATE member
int	farmId	If the state is OK, will contain the newly joined farm
string	password	If the state is OK, will contain the newly joined farm's password

onFarmPasswordEntered
Description

Handle confirmation of farm password input when joining a farm.

Definition

onFarmPasswordEntered()

onPermissionChanged
Description

Handle changes to user permissions.
If the current user's permissions change, update the menu buttons in case they were elevated to farm manager status.

Definition

onPermissionChanged()

onFarmCreated
Description

Handle creation of a new farm.

Definition

onFarmCreated()

Return Values

table self instance of class event

onFarmsChanged
Description

Handle state update of any farm.

Definition

onFarmsChanged()

Return Values

table instance instance of event

onPlayerFarmChanged

Description

Handle a player changing their farm.

Definition

onPlayerFarmChanged()

Return Values

table self instance of class event

onFarmMoneyChanged

Description

Handle a balance change of a farm.

Definition

onFarmMoneyChanged()

Return Values

table instance instance of event

onClickLeft

Description

Handle clicking on the left navigation button.

Definition

onClickLeft()

Return Values

table self instance of class event

onClickRight

Description

Handle clicking on the right navigation button.

Definition

onClickRight()

Return Values

table instance instance of event

onClickFarm

Description

Handle a click / activation of a farm item.

Definition

onClickFarm()

Return Values

table instance Instance of object

onDoubleClickFarm

Description

Handle a double click of a farm item.

Definition

onDoubleClickFarm()

Return Values

boolean success success

onSelectionChanged

Description

Handle a farm selection change.

Definition

onSelectionChanged()

Return Values

float dailyUpkeep daily up keep

onDeleteFarmYesNo

Description

Handle farm deletion dialog confirmation.

Definition

onDeleteFarmYesNo()

Return Values

float sellPrice sell price

InGameMenuMultiplayerUsersFrame

Description

Multiplayer user management frame for the in-game menu.
-- Displays a user list and allows modification of permissions as well as money transfers between farms. Administrators
can log in using a password and kick/ban/unban users when logged in.
--@category GUI

new

Description

Create a new InGameMenuMultiplayerUsersFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional] Meta table of subclass

Return Values

boolean isActiveForInput is active for input

initialize

Description

Late initialization.

Definition

initialize()

Return Values

table self instance of class event

onFrameOpen

Description

Called when this frame is opened by its container.

Definition

onFrameOpen()

Return Values

table instance instance of event

onFrameClose

Description

Called when this frame is closed by its container.

Definition

onFrameClose()

Return Values

table instance instance of object

setupUserListFocusContext

Description

Set up the user list to store an instance flag when navigating users or not for menu button context.

Definition

setupUserListFocusContext()

Return Values

integer id i3d rootnode

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

boolean isValid true if index is valid else false

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

integer id id of object

integer id id of used root node

setPlayerFarm

Description

Set the reference to the current player's farm.

Definition

setPlayerFarm()

setCurrentUserId

Description

Set the current user ID.

Definition

setCurrentUserId()

Return Values

New Binding instance

setBanStorage

Description

Set the ban storage reference.

Definition

setBanStorage()

Return Values

instance initialized with values from XML and the given parameters

setUsers

Description

Set the current users reference.

Definition

setUsers()

Return Values

int Combo bit mask

getSortedUsers

Description

Get a new sorted array of known users for displaying.
Users are grouped by farms, starting with the current player's farm (unless they're a spectator). Within groups,
users are sorted alphabetically by name.

Definition

getSortedUsers()

Return Values

True if there is a collision.

buildUserDisplayInfo

Description

Build a display name for a given user and their special privilege flags.

Definition

buildUserDisplayInfo()

Return Values

Cloned Binding instance

rebuildUserList

Description

Update connected player information.

Definition

rebuildUserList(table missionUsers)

Arguments

table missionUsers Array of users managed by the current mission instance.

Return Values

updateMenuButtons

Description

Update menu buttons based on the current user's admin status.

Definition

updateMenuButtons()

Return Values

instance initialized with values from XML

updateBalance

Description

Update the current balance display.

Definition

updateBalance()

Return Values

Cloned InputAction instance

setCurrentBalance

Description

Set the current money balance display.

Definition

setCurrentBalance(float balance, string balanceString)

Arguments

float balance Current balance of the current player

string balanceString Properly formatted money string

Return Values

table InputBinding instance

updateDisplay

Description

Update all display states and text.

Definition

updateDisplay()

Return Values

table Set of required device categories, {<category> = true}

onButtonBan

Description

Handle a ban button activation.

Definition

onButtonBan()

Return Values

bool True if there are any configured bindings for the given device, false otherwise

onYesNoBan

Description

Handle ban confirmation dialog result.

Definition

onYesNoBan()

Return Values

GS_INPUT_HELP_MODE_KEYBOARD or GS_INPUT_HELP_MODE_GAMEPAD

onButtonKick

Description

Handle a kick button activation.

Definition

onButtonKick()

Return Values

GS_INPUT_HELP_MODE_KEYBOARD or GS_INPUT_HELP_MODE_GAMEPAD

onYesNoKick

Description

Handle kick confirmation dialog result.

Definition

onYesNoKick()

Return Values

True if the parameters are valid, false otherwise.

onButtonUnBan

Description

Handle an unban button activation

Definition

onButtonUnBan()

Return Values

bool True if the event could be registered, false otherwise

string event ID if successful, empty string otherwise

table Action reference of a colliding action if there would be a collision, nil otherwise

onButtonShowProfile

Description

Handle a show user profile button activation.

Definition

onButtonShowProfile()

onButtonInviteFriends

Description

Handle an invite friends button activation

Definition

onButtonInviteFriends()

onButtonAdminLogin

Description

Handle an admin login button activation.

Definition

onButtonAdminLogin()

Return Values

bool True if there would be a collision, false otherwise

table Colliding action if there would be a collision, nil otherwise

onUserSelected

Description

Handle selection changes in the user list.

Definition

onUserSelected()

onClickPermission

Description

Handle clicking a permission check box.

Definition

onClickPermission(table checkboxElement, bool isActive)

Arguments

table checkboxElement ToggleButtonElement which received the click

bool isActive New checked state

Return Values

array of tuples: {i={action=InputAction, event=InputEvent}}

onClickTransferButton

Description

Handle clicking the transfer button

Definition

onClickTransferButton()

Return Values

Gamepad combo mask, Mouse combo mask

transferMoney

Description

Transfer money from the current player's farm to the selected farm.

Definition

transferMoney()

Return Values

instance or nil if ID is invalid

onClickRemoveFromFarm

Description

Handle clicking the remove player from farm button.

Definition

onClickRemoveFromFarm()

Return Values

instance or nil if ID is invalid

onYesNoRemoveFromFarm

Description

Handle remove player from farm confirmation dialog result.

Definition

onYesNoRemoveFromFarm()

Return Values

instance or nil if ID is invalid

onClickPromote

Description

Handle clicking the promote player to farm manager button.

Definition

onClickPromote()

Return Values

float Mouse cursor X position in screen space

float Mouse cursor Y position in screen space

onYesNoPromoteToFarmManager

Description

Handle promote user confirmation dialog result.

Definition

onYesNoPromoteToFarmManager()

onClickContractor

Description

Handle clicking the contractor toggle button.

Definition

onClickContractor()

Return Values

previousDevices Table of internal ID -> InputDevice before the current initialization

onYesNoToggleContractorState

Description

Handle contractor state toggle confirmation dialog result.

Definition

onYesNoToggleContractorState()

Return Values

List of device information tables, format: {deviceId=[device ID], name=[device name]}

onFarmMoneyChanged

Description

Handle a local money change event for any farm.

Definition

onFarmMoneyChanged()

Return Values

Modifier bit mask

onFarmsChanged

Description

Handle state update of any farm.

Definition

onFarmsChanged()

Return Values

True if the binding's device ID could be resolved, false otherwise

onPlayerFarmChanged

Description

Handle a player changing their farm.

Definition

onPlayerFarmChanged()

Return Values

function results table {checkFunctionRef=[functionResult]}

onPermissionChanged

Description

Handle a local player permission change event.

Definition

onPermissionChanged()

Return Values

onContractingStateChanged

Description

Handle a local contracting state change event.

Definition

onContractingStateChanged()

Return Values

bool True if the new binding could be added

table Reference to an Action containing a binding which collided with the added binding or nil

onAdminPassword

Description

Handle admin login password dialog response.

Definition

onAdminPassword()

onAdminLoginSuccess

Description

Handle successfully logging in as an admin.

Definition

onAdminLoginSuccess()

Return Values

True if a binding could be found and updated, false if there was a collision

and Action reference of a collision as {collisionBinding=Binding, collisionAction=Action} or nil

InGameMenuPricesFrame

Description

Prices overview frame of the in-game menu.
-- Displays current prices of sellable goods and produce.
--@category GUI

new

Description

Create a new InGameMenuPricesFrame instance.

Definition

new(table subclass_mt, table l10n, table fillTypeManager)

Arguments

table subclass_mt	[optional] Meta table of subclass
table l10n	I18N reference
table fillTypeManager	FillTypeManager reference

makeTableFocusOverrideFunction

Description

Create a focus override function for the prices table which scrolls horizontally if possible instead of exiting the
table.

Definition

makeTableFocusOverrideFunction()

Return Values

True if all given axes are currently active, input value of the non-modified axis (last in list)

getStationName

Description

Get the display name of a selling station.

Definition

getStationName()

Return Values

True if all axes are active (digital interpretation), input value of the non-modified axis (last in list)

setSellingStations

Description

Set the current list of selling stations for displaying.

Definition

setSellingStations()

Return Values

Input value of the axis in the range [-1, 1] for full axes or [0, 1] for half axes.

updateVerticalSlider

Description

Update item counts and handle size of the vertical slider.

Definition

updateVerticalSlider()

Return Values

setupPriceTable

Description

Initial setup of the prices table and required data.

Definition

setupPriceTable()

Return Values

Ordered action bindings in the form of {i={action=InputAction, bindings={Binding}}}

updatePriceTable

Description

Update prices table display data.

Definition

updatePriceTable()

Return Values

updateHeaderIcons

Description

Update table headers with fill type icons according to current horizontal slider position.

Definition

updateHeaderIcons()

Return Values

table Array of events

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

table First active event or InputEvent.NO_EVENT

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

x motion scale, Y motion scale

onChangedPriceSlider

Description

Handle horizontal slider changes.

Definition

onChangedPriceSlider()

Return Values

True if the user settings have passed the integrity check, false otherwise.

onClickPriceHeader

Description

Handle price table header clicks.
Causes the table to be sorted by the clicked header column's numeric price values.

Definition

onClickPriceHeader()

Return Values

True if the axis name represents a physical full axis, false otherwise

onClickSellingPointHeader

Description

Handle selling point table header clicks.
Causes the table to be sorted by selling point names.

Definition

onClickSellingPointHeader()

Return Values

True if the input name represents an analog input

onClickPrices

Description

Handle table row clicks.

Definition

onClickPrices()

Return Values

True if the axis is at zero position, false otherwise

onDoubleClickPrices

Description

Handle table row double-clicks.
Sets a map marker on the selected selling point or clears it again.

Definition

onDoubleClickPrices()

Return Values

New InputDevice instance

onDataBindSellingPoint

Description

Bind an element to the selling point data.

Definition

onDataBindSellingPoint()

Return Values

Device ID or empty string

onDataBindPrice

Description

Bind an element to the indexed price data.

Definition

onDataBindPrice()

Return Values

Device name or empty string

onDataBindSiloCapacityLabel

Description

Bind an element to the silo capacity label data.

Definition

onDataBindSiloCapacityLabel()

Return Values

Prefix number, or -1 if none exists; raw engine-issued device ID

onDataBindSiloCapacityValue

Description

Binding an element to the indexed silo capacity value data.

Definition

onDataBindSiloCapacityValue()

Return Values

instance containing the symbol overlays in its "buttons" field

setSellingPointData

Description

Set data for a selling point table cell.

Definition

setSellingPointData(table dataCell, table sellingStation)

Arguments

table dataCell TableElement.DataCell instance for a selling point cell

table sellingStation SellingStation instance of the current data row

Return Values

array of help elements (see InputDisplayManager:makeHelpElement() for structure)

setPriceData

Description

Set data for a price table cell.

Definition

setPriceData(table dataCell, int priceIndex, table sellingStation)

Arguments

table dataCell TableElement.DataCell instance for a price cell

int priceIndex Price column index

table sellingStation SellingStation instance of the current data row

Return Values

Hash table of currently active combo button action names, {action name=true}

buildDataRow

Description

Build a DataRow from selling point properties to add to the prices table

Definition

buildDataRow(table sellingStation)

Arguments

table sellingStation SellingStation reference

Return Values

Axis name or nil if not found, internal ID of binding device

dataRow instance with prices data for the given selling point

sortPrices

Description

Table sorting function for prices.

Definition

sortPrices(table sortCell1, table sortCell1)

Arguments

table sortCell1 TableElement.SortCell instance representing the first cell to compare

table sortCell1 TableElement.SortCell instance representing the second cell to compare

Return Values

instance or nil if no overlay is available for the action's bindings

float Sorting value, see TableElement:setCustomSortFunction(...)

getStorageFillLevel

Description

Get the storage fill level for a given fill type index.

Definition

getStorageFillLevel(int index, bool farmSilo)

Arguments

int index Fill type index of the requested storage fill level

bool farmSilo If true, only counts storage of owned farm silos

Return Values

key text (e.g. "F" for "KEY_f") for the keyboard action binding or nil if no resolution is possible

float Current fill level >= 0 or a value < 0 if no storage exists for the requested fill type index

float Total capacity for the fill type >= 0 or a value < 0 if no storage exists for the requested fill type index

InGameMenuStatisticsFrame

Description

Game statistics display frame for the in-game menu.
--@category GUI

new

Description

Create a new InGameMenuStatisticsFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional] Meta table of subclass

Return Values

controller symbols configuration key name

updateStatistics

Description

Update statistics data in tables.

Definition

updateStatistics()

Return Values

controller symbols configuration key name

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

New InputEvent instance

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

table instance instance of object

buildDataRow

Description

Build a table data row for a given statistics attribute.

Definition

buildDataRow(table statAttribute)

Arguments

table statAttribute Statistics attribute as defined in FarmStats

Return Values

boolean true if loading was successful else false

table TableElement.DataRow instance

InGameMenuTutorialFrame

Description

Exposed controls usable as fields in this frame, identified in configuration.

new

Description

Create a new InGameMenuFinancesFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional] Meta table of subclass

Return Values

string cutterEffectType the real cutterEffect name, nil if not defined

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

table cutterEffects cutter effects

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

table instance instance of object

InGameMenuVehiclesFrame

Description

Exposed controls usable as fields in this frame, identified in configuration.

new

Description

Create a new InGameMenuVehiclesFrame instance.

Definition

new(table subclass_mt, table messageCenter, table l10n, table storeManager, table brandManager)

Arguments

table subclass_mt	Sub-class meta table
table messageCenter	MessageCenter reference for notifications
table l10n	I180N reference for display text resolution
table storeManager	StoreManager reference for vehicle store data look-ups
table brandManager	BrandManager reference for vehicle brand data look-ups

Return Values

boolean true if loading was successful else false

initialize

Description

Late initialization.

Definition

initialize()

Return Values

string materialType the real material name, nil if not defined

updateGarage

Description

Update the garage view.

Definition

updateGarage()

Return Values

integer materialId id of material

updateVerticalSlider

Description

Update item counts and handle size of the vertical slider.

Definition

updateVerticalSlider()

Return Values

table instance instance of object

setAccessibleVehicles

Description

Set a list of vehicles currently accessible by the player.

Definition

setAccessibleVehicles()

Return Values

boolean true if loading was successful else false

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

string particleType the real particle name, nil if not defined

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

table particleSystem particleSystem

makeTableHeaderFocusOverrideFunction

Description

Create a focus override function for a table header which modifies the scroll bar instead of navigation when
vertical navigation input is received.

Definition

makeTableHeaderFocusOverrideFunction()

Return Values

table instance instance of object

setNameData

Description

Set data for a vehicle name table cell.

Definition

setNameData()

Return Values

boolean true if loading was successful else false

setAgeData
Description

Set data for a vehicle age table cell.

Definition

setAgeData()

Return Values

boolean true if loading was successful else false

setOperatingHoursData
Description

Set data for a vehicle operating hours table cell.

Definition

setOperatingHoursData()

Return Values

table instance instance of object

setDamageData
Description

Set data for a vehicle damage table cell.

Definition

setDamageData()

Return Values

table instance instance of object

setLeasingData
Description

Set data for a vehicle leasing costs table cell.

Definition

setLeasingData()

Return Values

table instance instance of object

setValueData
Description

Set data for a vehicle selling value table cell.

Definition

setValueData()

Return Values

boolean true if loading was successful else false

buildDataRow
Description

Build a DataRow from vehicle properties to add to the table.

Definition

buildDataRow(table vehicle)

Arguments

table vehicle Vehicle reference

Return Values

table baleType baleType object

dataRow instance with vehicle data for the given vehicle

sortAttributes

Description

Table sorting function for numeric values.

Definition

sortAttributes(table sortCell1, table sortCell1)

Arguments

table sortCell1 TableElement.SortCell instance representing the first cell to compare

table sortCell1 TableElement.SortCell instance representing the second cell to compare

Return Values

table baleType baleType object

float Sorting value, see TableElement:setCustomSortFunction(...)

onClickVehicleHeader

Description

Handle clicks on the vehicle name header.
Switches table to default string sorting.

Definition

onClickVehicleHeader()

Return Values

table bale bale

onClickAttributeHeader

Description

Handle clicks on an attribute header.
Switches table to numerical sorting.

Definition

onClickAttributeHeader()

Return Values

string baleKey bale key

InputAction

Description

Logical input action.
-- Game components react to actions to alter their state. Available actions are loaded from "dataS/inputActions.xml".
Default action names are available for use as constants in the form of InputAction.ACTION_NAME (see end of file).
Additional actions (e.g. from mods) will be dynamically added as such constants at run time.
--@category Input
--@xmlConfig action#name Action name. This serves as a unique identifier in both configuration and code.

new

Description

Create a new InputAction instance

Definition

new(name Action, axisType Type, isLocked If, ignoreComboMask If, displayNamePositive [optional], displayNameNegative [optional])

Arguments

name	Action	name, unique
axisType	Type	of action axis (HALF or FULL)
isLocked	If	true, bindings for this action cannot be changed in the game
ignoreComboMask	If	true, non-combo bindings will trigger this action even if a combo button is currently active
displayNamePositive	[optional]	Display name of the action in positive direction for the current localization setting. This is the default display name.
displayNameNegative	[optional]	Display name of the action in negative direction for the current localization setting. Only used for full axes.

Return Values

InputAction

createFromXML

Description

Create a new InputAction instance from an XML element.

Definition

createFromXML(xmlFile XML, elementTag Tag)

Arguments

xmlFile	XML	file handle
elementTag	Tag	of the element to parse as an InputAction

Return Values

integer	numPixels	number of pixels
InputAction	instance	initialized with values from XML

addBinding

Description

Add a Binding to this action.

Definition

addBinding()

Return Values

integer	changedArea	changed area pixels
integer	totalArea	total area pixels

removeBinding

Description

Remove a binding from this action.

Definition

removeBinding()

getBindings

Description

Get the bindings for this action.

Definition

getBindings()

Return Values

integer changedArea changed area pixels

integer totalArea total area pixels

clearBindings

Description

Clear this action's bindings.

Definition

clearBindings()

setPrimaryKeyboardBinding

Description

Let this action know which binding is its primary keyboard binding.
Stores the input axes as a concatenated string for comparisons.

Definition

setPrimaryKeyboardBinding(Binding Binding)

Arguments

Binding Binding instance which is the primary keyboard binding for this action

Return Values

integer numChangedPixels number of changed pixels

isFullAxis

Description

Determine if this action represents a logical full axis.

Definition

isFullAxis()

Return Values

table color tire track color

getIgnoreComboMask

Description

Determine if bindings of this action should trigger any associated events if the bindings are not combos but a combo
button is currently pressed.

Definition

getIgnoreComboMask()

Return Values

string encodedString the encoded string

clone

Description

Create a new InputAction instance with the same state as this instance.

Definition

clone()

Return Values

string decodedString the decoded string

Cloned InputAction instance

toString

Description

Get a string representation for this InputAction.

Definition

toString()

Return Values

table copy the copied table

InputBinding

Description

Input binding manager.
-- Loads and saves action input bindings and provides action trigger information.
-- Methods of note (see the documentation of the actual methods for more detailed descriptions):
registerActionEvent() Register an event callback for an action (for action values, see InputAction.lua)
removeActionEvent() Removes a previously registered event. Also see removeActionEventsByTarget().
setActionEvent...() Group of methods to adjust event states, e.g. activity or input display hint parameters.
-- Input event usage scenarios:
1. Long lifetime components / much interaction: Register event at creation of a component, modify event in component
update method depending on its state, remove event at component destruction
2. Short lifetime components / little interaction: Register event when necessary (e.g. when in range of an object),
remove again when no interaction is possible anymore
-- Player input settings are loaded from the file "inputBindings.xml" in their profile directory. If the file is not
present or corrupted whenever input is loaded, it will be restored from a suitable template. If a binding of a locked
action is changed directly on the file system, it will be overwritten with its template counterpart on the next
loading call to ensure that critical inputs are always available (e.g. menu navigation).
--@category Input

new

Description

Create the InputBinding instance.

Definition

new(table logManager, table modManager, table messageCenter, bool isConsoleVersion)

Arguments

table logManager	LogManager reference
table modManager	ModManager reference
table messageCenter	MessageCenter reference
bool isConsoleVersion	If true, the game is running on console

Return Values

table copy	the copied table
table InputBinding	instance

load

Description

Load the input binding configuration.

Definition

load(bool isInitializing, bool forceDefaultBindings)

Arguments

bool isInitializing	[optional, default=false] If true, loads the input bindings in initialization mode and saves at the end of loading to store any resolved device IDs to the user settings.
bool forceDefaultBindings	[optional, default=false] If true, will forcibly load default bindings and override any user input settings.

Return Values

boolean isElementOfList true if element is part of the list, else false

loadDefaultBindings

Description

Load default input bindings.
Will apply defaults for newly connected devices or override all bindings with template defaults if requested.

Definition

loadDefaultBindings(bool forceReplace)

Arguments

bool forceReplace	[optional, default=false] If true, will overwrite all user bindings with the default templates.

Return Values

integer index index of first occurrence

loadModActions

Description

Load actions defined by mods.

Definition

loadModActions()

Return Values

boolean areEqual true if lists are equal, else false

loadModBindingDefaults

Description

Load default mod input binding data.

Definition

loadModBindingDefaults()

Return Values

any value value of the random element

assignPlatformBindingPaths

Description

Assign bindings configuration file paths depending on platform and devices.

Definition

assignPlatformBindingPaths()

Return Values

table set the converted set

overwriteSettingsWithDefault

Description

Overwrites user input settings with the default template.

Definition

overwriteSettingsWithDefault(forceOverwrite If)

Arguments

forceOverwrite If	true, will overwrite an existing settings file. If false, will only perform the operation if no user settings are present.

Return Values

table list the converted list

restoreInputContexts

Description

Restore input contexts after reloading the input settings, e.g. when devices change.

Definition

restoreInputContexts()

Return Values

table hash the converted hash

setShowMouseCursor

Description

Set the visibility state of the mouse cursor.

Definition

setShowMouseCursor(bool doShow, bool saveCursorPosition)

Arguments

bool doShow	If true, the cursor will be shown. Otherwise, it will be hidden.
bool saveCursorPosition	[optional] If true, saves the current cursor position to restore when showing it again

Return Values

boolean areEqual true if sets are equal, else false

getShowMouseCursor

Description

Determine if the mouse cursor is currently being shown.

Definition

getShowMouseCursor()

Return Values

boolean isSubset true if set1 is a subset of set2

getInputHelpMode

Description

Get the current input help mode for the input hint display.
If the input help mode has been set to automatic, the last button or key input will determine which device scheme
is used.

Definition

getInputHelpMode()

Return Values

boolean	isSubset true if set1 is a real subset of set2
GS_INPUT_HELP_MODE_KEYBOARD or	GS_INPUT_HELP_MODE_GAMEPAD

getLastInputMode

Description

Get the current input mode, independent of input help mode settings.

Definition

getLastInputMode()

Return Values

boolean	hasIntersection true if set1 and set2 have an intersection
GS_INPUT_HELP_MODE_KEYBOARD or	GS_INPUT_HELP_MODE_GAMEPAD

validateActionEventParameters

Description

Validate parameters for registerActionEvent().
Prints warnings if there are problems and returns the validation status.

Definition

validateActionEventParameters()

Return Values

table intersection	the intersection of both sets
True if	the parameters are valid, false otherwise.

registerActionEvent

Description

Register an event callback for an input action.
Use InputAction constants for action names. At least one of the trigger parameters must be set to true to receive
events. This methods also checks if there would be any input binding collision when an event is added for the given
action. If there are any collisions, the registration will fail.
-- Note the interaction of the "down" and "always" triggers: If "down" is set and not "always", only an input down-flank
will raise an event. If both "down" and "always" are set, an event is raised as long as the input is pressed beyond a
threshold. If only "always" and not "down" is set, an event will be raised on each frame with the current input value
even if no input is active.
-- @param string actionName Name of action, see InputAction

Definition

registerActionEvent(table targetObject, function eventCallback, bool triggerUp, bool triggerDown, bool triggerAlways, bool startActive, table callbackState)

Arguments

table	targetObject	Event target, first argument to event callback
function	eventCallback	Event callback, called when the action has input. Signature: callback(targetObject, actionName, inputValue, callbackState)
bool	triggerUp	If true, the event fires once when an input signal of the given action goes to "up" state, e.g. a released key.

bool	triggerDown	If true, the event fires once when an input signal of the given action goes to "down" state, e.g. a pressed key.
bool	triggerAlways	If true, the event fires on any signal input change, potentially every frame. This is mostly useful for required continuous axis input, e.g. steering actions.
bool	startActive	[optional] If true, the event is active right after registration and will be checked for input collisions. Otherwise, set its activation state using InputBinding:setActionEventActive() and handle collisions yourself.
table	callbackState	[optional] An arbitrary value or reference which serves as a closure state within callbacks, useful e.g. when checking input state between events without requiring additional class fields.

Return Values

table	substraction	the substraction of both set
bool	True	if the event could be registered, false otherwise
string	event	ID if successful, empty string otherwise
table	Action	reference of a colliding action if there would be a collision, nil otherwise

checkEventCollision

Description

Check if adding an event for a given action name would cause colliding input bindings to be active.

Definition

checkEventCollision(string actionName)

Arguments

string	actionName	Name of an InputAction

Return Values

table	union	the union of both sets
bool	True	if there would be a collision, false otherwise
table	Colliding action	if there would be a collision, nil otherwise

beginActionEventsModification

Description

Start registering action events in a given context.
This method must be accompanied by a finalizing call to InputBinding:endActionEventsModification() when registration is complete. While a registration context is set, all calls to InputBinding:registerActionEvent() will add events to that context. Also, derived event collections (i.e. data structures for display and fast iteration) will only be updated when InputBinding:endActionEventsModification() is called instead of each time an event is added.
If no context with the given name exists, a new empty one will be created and added to the known contexts.

Definition

beginActionEventsModification(string inContext, bool createNew)

Arguments

string	inContext	Name of the input context which receives any registered events until endActionEventsModification is called.
bool	createNew	[optional, default=false] If true, will create a new context of the given name, potentially overwriting any existing one.

Return Values

table filtered list (non-deep copy)

endActionEventsModification

Description

End registering action events in a context.
Resets the registration context set in InputBinding:beginActionEventsModification() for any later registrations.

Definition

endActionEventsModification()

Return Values

table filtered list (non-deep copy)

refreshEventCollections

Description

Refresh derived event collections for retrieval purposes.

Definition

refreshEventCollections()

Return Values

integer sign the sign of the value

storeDisplayActionEvents

Description

Store an array of action-event tuples for all action events which need an input hint display.

Definition

storeDisplayActionEvents()

Return Values

boolean true if value is nan else false

storeEventBindings

Description

Store all bindings associated with registered and active events in collections for iteration.

Definition

storeEventBindings()

Return Values

number rounted value

iterateEvents

Description

Iterates the action event collection, calling a given function on each event.
If the processing function returns a "truthy" value, the iteration is stopped.

Definition

iterateEvents(processingFunction function(event,)

Arguments

processingFunction function(event, actionName, actionEventList, actionEventListIndex)

Return Values

float value radian angle

removeEventInternal

Description

Internal function for event removal.

Definition

removeEventInternal()

Return Values

float value interpolated value

removeActionEvent

Description

Remove a previously registered action event by ID.

Definition

removeActionEvent(eventId Event)

Arguments

eventId Event ID as returned by registerActionEvent().

Return Values

float value interpolated value

removeActionEventsByActionName

Description

Remove all previously registered action events which are triggered by an action identified by name.

Definition

removeActionEventsByActionName(actionName If)

Arguments

actionName If an event is triggered by the action of that name, it is removed.

Return Values

float alpha alpha

removeActionEventsByTarget

Description

Remove all previously registered action events which have a given target.

Definition

removeActionEventsByTarget(targetObject If)

Arguments

targetObject If an event has this object as a target, it is removed.

Return Values

float value value

getDisplayActionEvents

Description

Get action-event tuples for registered events which require an input hint.
Note that the data is live and should be treated as read-only.

Definition

getDisplayActionEvents()

Return Values

float isOutOfBounds is out of bounds

array of tuples: {i={action=InputAction, event=InputEvent}}

setActionEventText

Description

Set a specific action text to display as an input hint for a given event.
Use this to adjust display texts depending on context, e.g. "Attach" / "Detach" for the vehicle attach action.
Additionally, it serves as a way to display neutral full axis action names, e.g. "Move player" instead of any
specific label depending on the input direction.

Definition

setActionEventText(eventId ID, actionText Localized)

Arguments

eventId	ID	of a registered event
actionText	Localized	display text for the event's action

Return Values

float value the floored percent value

setActionEventIcon

Description

Set the name of an icon to display as an input hint instead of text for a given event.
If an icon has been specified for an event with this function, any text setting will have no effect.

Definition

setActionEventIcon(eventId ID, iconName Input)

Arguments

eventId	ID	of a registered event
iconName	Input	hint icon name as defined in axisIcons.xml or InputHelpElement.AXIS_ICON

Return Values

float value the floored clamped value

setActionEventTextVisibility

Description

Set the visibility of an action event input hint display.
The most basic controls, such as player movement, do not need any input hints and can be hidden this way. If more
than one event is registered on an action, make sure to only set one of them to visible. The system will
automatically set all events after the first per action to invisible to ensure a valid ground state.

Definition

setActionEventTextVisibility(eventId ID, isVisible If)

Arguments

eventId	ID	of a registered event
isVisible	If	false, hides the input hint for this event. Default is visible.

Return Values

float angle the resized angle in the range -pi to pi

setActionEventTextPriority

Description

Set the priority of an action event input hint display.
Use any of the following script constants: GS_PRIO_VERY_HIGH, GS_PRIO_HIGH,

GS_PRIO_NORMAL, GS_PRIO_LOW,
GS_PRIO_VERY_LOW. Higher priority will be shown first in the input hint display box.

Definition

setActionEventTextPriority(eventId ID, priority Priority)

Arguments

eventId ID of a registered event

priority Priority number value, lower is more important. Use GS_PRIO_[...] constants.

Return Values

float angle the radian difference in range -pi to pi

setActionEventActive

Description

Set the active state of an action event.
Only active events can be triggered by input and be displayed as input hints.

Definition

setActionEventActive(eventId ID, isActive New)

Arguments

eventId ID of a registered event

isActive New active state

Return Values

float length length

setActionEventsActiveByTarget

Description

Set the active state of all action events targeting a given object.
Only active events can be triggered by input and be displayed as input hints.

Definition

setActionEventsActiveByTarget(targetObject Action, isActive New)

Arguments

targetObject Action event target object

isActive New active state

Return Values

float length square length

getComboCommandPressedMask

Description

Get the combo input masks for any currently active gamepad and mouse combo inputs.
The returned masks are bit-wise combined masks made from the
InputBinding.COMBO_MASK_[...] constants.

Definition

getComboCommandPressedMask()

Return Values

float x normalized x

float y normalized y

Gamepad combo mask, Mouse combo mask

getComboActionNameForAxisSet

Description

Get the combo action name for a given set of modifier axes.

Definition

getComboActionNameForAxisSet()

getInternalIdByDeviceId

Description

Resolve a device ID sting to the engine-internal integer ID.

Definition

getInternalIdByDeviceId()

Return Values

float length length

getDeviceByInternalId

Description

Retrieve an InputDevice by its internal device ID.

Definition

getDeviceByInternalId(internalDeviceId Internally)

Arguments

internalDeviceId Internally assigned device ID (= device index assigned by engine, zero based)

Return Values

float length square length

InputDevice instance or nil if ID is invalid

assignLastInputHelpMode

Description

Assign an input help mode as the last used mode.
This triggers a message center notification to all subscribed components if the input help mode has changed.

Definition

assignLastInputHelpMode()

Return Values

float x normalized x

float y normalized y

float z normalized z

startBindingChanges

Description

Prepare for incoming binding changes.
Must be called before InputBinding:startInputCapture() to guarantee a valid state.

Definition

startBindingChanges()

commitBindingChanges

Description

Commit all binding changes since the last call to InputBinding:startBindingChanges().
Accepts the current, modified binding state and notifies any change listeners.

Definition

commitBindingChanges()

rollbackBindingChanges

Description

Rolls back all binding changes since the last calls to InputBinding:startBindingChanges().

Definition

rollbackBindingChanges()

Return Values

float x scaled x

float y scaled y

float z scaled z

startInputCapture

Description

Capture all input and run a callback if input has been received on a device.
Overrides the global input event functions.

Definition

startInputCapture(isKeyboard True, isMouse True, callbackTarget If, callbackState Free, inputCallback Called, abortCallback Called, deleteCallback Called)

Arguments

isKeyboard	True	if target device is keyboard input
isMouse	True	if target device is mouse input
callbackTarget	If	specified, will be supplied as the first argument to the callback function (usually caller self)
callbackState	Free	parameter which is returned with callbacks to the callback target
inputCallback	Called	when the target device receives input. function(deviceId, inputAxisName, isModifier, inputValue, callbackState)
abortCallback	Called	when an abort input is received on any device. function()
deleteCallback	Called	when a delete input is received on any device. function(callbackState)

captureKeyboardInput

Description

Capture keyboard input.
Fires an input callback as soon as a key is pressed. Modifier keys (e.g. shift, alt) only trigger callbacks when
they are pressed alongside a regular key. Modifier callbacks always arrive before non-modifier key callbacks.

Definition

captureKeyboardInput(abortCallback function(), deleteCallback function(), inputCallback function(deviceId,)

Arguments

abortCallback	function()	to be called when the abort key is pressed
deleteCallback	function()	to be called when the delete key is pressed
inputCallback	function(deviceId, axisName, isModifier, inputValue), see startInputCapture()	

captureMouseInput

Description

Capture mouse input.
Axis input continuously fires callbacks while the player is moving the mouse. Button

callbacks are triggered once
when a mouse button is held down and again when it is released.

Definition

captureMouseInput(callback function(deviceId,)

Arguments

callback function(deviceId, axisName, isModifier, inputValue), see startInputCapture()

Return Values

float x clamped x

float y clamped y

float z clamped z

captureGamepadInput

Description

Capture gamepad / controller input.
Continously fires callbacks for all known gamepad buttons and axes for each frame.

Definition

captureGamepadInput(callback function(deviceId,)

Arguments

callback function(deviceId, axisName, isModifier, inputValue), see startInputCapture()

stopInputGathering

Description

Stop the current input gathering run.
Restores global input event functions.

Definition

stopInputGathering()

restoreDefaultBindings

Description

Restore default bindings

Definition

restoreDefaultBindings()

Return Values

float x interpolated x

float y interpolated y

float z interpolated z

clearState

Description

Clear relevant state for loading and resetting.

Definition

clearState()

loadActions

Description

Load input actions which are later bound to input axes.

Definition

loadActions(xmlFile Action, modName [optional])

Arguments

xmlFile	Action	definition XML file handle
modName	[optional]	Name of the mod which defines actions. If not set, actions are assumed to be default game actions.

resetDeviceInformation

Description

Reset stored device information.
Clears all device information and returns the previous device list for comparisons. Call this for
initialization and before enumerating system devices.

Definition

resetDeviceInformation()

Return Values

float	x	interpolated x
float	y	interpolated y
float	z	interpolated z

previousDevices Table of internal ID -> InputDevice before the current initialization

createDefaultDevices

Description

(Re-)Creates the internal keyboard and mouse virtual device.

Definition

createDefaultDevices()

enumerateGamepadDevices

Description

Find gamepad devices in the system and store their information.

Definition

enumerateGamepadDevices(previousDevices Table)

Arguments

previousDevices Table of internal ID -> InputDevice before the current initialization

getGamepadDevices

Description

Get a list of recognized gamepad (and other controller) input devices.

Definition

getGamepadDevices()

Return Values

float x	transformed x
float y	transformed y
float z	transformed z

List of device information tables, format: {deviceId=[device ID], name=[device name]}

loadDeviceSettingsFromXML

Description

Load device input settings (other than bindings) from an input binding XML file.

Definition

loadDeviceSettingsFromXML(xmlFile XML)

Arguments

xmlFile XML file handle

applyGamepadDeadzones

Description

Apply deadzone settings to all gamepad device axes.
If no deadzone value had been configured for a valid axis, the global default deadzone value is applied.

Definition

applyGamepadDeadzones()

initializeGamepadMapping

Description

Initialize gamepad settings from XML.

Definition

initializeGamepadMapping()

Return Values

dot the dot product

validateAndRepairComboActionBindings

Description

Check if combo action bindings are valid and restore proper bindings if necessary.

Definition

validateAndRepairComboActionBindings()

Return Values

float x x

float y y

float z z

loadActionBindingsFromXML

Description

Load bindings of actions to input axes from a configuration XML file.

Definition

loadActionBindingsFromXML(xmlFile Configuration, silentIgnoreDuplicates [optional], modName [optional], doNotReplace [optional])

Arguments

xmlFile	Configuration	XML file handle
silentIgnoreDuplicates	[optional]	If true, will not give a warning on duplicate bindings and just ignore them
modName	[optional]	Name of the mod for which action bindings are loaded. If not set, will load bindings for the default game.
doNotReplace	[optional]	If true, will not replace existing bindings with a loaded ones

loadBindingsFromXML

Description

Load all bindings within an XML actionBindings definition element.

Definition

loadBindingsFromXML()

storeComboInputMappings

Description

Store input axis names of combo command actions for combo mask checking.

Definition

storeComboInputMappings()

Return Values

float y rotation

getBindingComboMask

Description

Get a combo modifier mask for a binding.
Because input axis names are different, this is safe to use for any binding as long as the calling context sticks
to either gamepad or mouse input bindings to check.

Definition

getBindingComboMask(Binding Binding)

Arguments

Binding Binding instance

Return Values

float x x direction

float z z direction

Modifier bit mask

assignComboMasks

Description

Assign combination input bitmasks on actions based on their bindings.
This will update all actions and should be run whenever any binding changes. Axis inputs will have two valid bindings
for modifiers (component + and -). Ensure that players cannot set differently modified inputs on a single axis
action.

Definition

assignComboMasks()

storeLinkedBindings

Description

Store binding links based on connections set in InputAction.LINKED_ACTIONS.

Definition

storeLinkedBindings()

Return Values

float x limited x direction

float z limited z direction

assignActionPrimaryBindings

Description

Assign primary keyboard bindings to actions if applicable.

Definition

assignActionPrimaryBindings()

adjustBindingSlotIndex

Description

Adjust a binding's slot index.

Definition

adjustBindingSlotIndex(Binding Binding, usedSlots Set)

Arguments

Binding	Binding to adjust
usedSlots	Set of used slot indices for the binding's context (action, device, axis)

Return Values

float x x position

float x z position

resolveBindingDevice

Description

Resolve a binding's device ID to an engine-issued internal device ID.
Replaces default device IDs in bindings with the ID of the first device of the corresponding category. This
effectively binds default or template bindings to first devices. Also updates the binding's internal device ID field.
If a binding's device is missing, the method tries resolving a suitable replacement device. First, device IDs are
checked to see if the missing device simply changed order (and therefore its prefix). Then, the device names are
compared. If no identifiers match, the binding is assigned to the first device of the category of the missing device.
If all else fails, the binding is assigned to the first controller, possibly losing some bindings due to collisions.

Definition

resolveBindingDevice(Binding Binding)

Arguments

Binding Binding whose device ID is checked and replaced, if necessary

Return Values

True if the binding's device ID could be resolved, false otherwise

resolveFirstDeviceOfCategoryToBinding

Description

Resolve and assign the first device of a given category to a binding.

Definition

resolveFirstDeviceOfCategoryToBinding()

Return Values

float dot product

resolveSameIdDeviceWithPrefixToBinding

Description

Resolve and assign the last device with a given base ID (without prefix) to a binding.

Definition

resolveSameIdDeviceWithPrefixToBinding()

Return Values

float x x part of quaternion

float y y part of quaternion

float z z part of quaternion

float w w part of quaternion

resolveMissingDeviceByNameToBinding

Description

Resolve and assign the last device with a given name to a binding.

Definition

resolveMissingDeviceByNameToBinding()

resolveDefaultDeviceToBinding

Description

Resolve and assign the first known device to a binding.

Definition

resolveDefaultDeviceToBinding()

checkBindings

Description

Run a a list of functions on all known bindings in the context of a given action if provided.
The functions' return values are written to a results table which is returned at the end. Callers can retrieve
function results by indexing the results table with the corresponding function reference.

Definition

checkBindings(action Action, checkFunctions List)

Arguments

action Action reference for the current context, e.g. check bindings for an action

checkFunctions List of functions with the signature function(Binding), return [bool hasResult], result

Return Values

function results table {checkFunctionRef=[functionResult]}

validateBinding

Description

Validate a new binding before it's added to the system.
Returns boolean flags for constraints and a colliding binding if there was one.

Definition

validateBinding(Binding New)

Arguments

Binding New binding instance

Return Values

float x x part of quaternion

float y y part of quaternion

float z z part of quaternion

float w w part of quaternion

addBinding

Description

Add a new input binding.
Checks for duplicates and collisions and stores the binding only if everything's okay.

Definition

addBinding(table binding, bool silentIgnoreDuplicates, bool doNotReplace)

Arguments

table	binding	New binding
bool	silentIgnoreDuplicates	If true, will not print warnings on duplicate bindings
bool	doNotReplace	If true, will not replace existing bindings

Return Values

bool	True	if the new binding could be added
table	Reference to an Action containing a binding which collided with the added binding or nil	

updateBinding

Description

Update an existing binding with new device ID, input axes and input component data.

Definition

updateBinding(findDeviceId Binding, findActionName Name, findBindingIndex Binding, findAxisComponent Logical, deviceId New, axisNames List, inputComponent Physical)

Arguments

findDeviceId	Binding	device ID
findActionName	Name	of binding action
findBindingIndex	Binding	index, 1 primary, 2 secondary, etc.
findAxisComponent	Logical	binding axis direction component to assign, always positive for buttons/keys/half-axes
deviceId	New	device ID
axisNames	List	of new input axis names
inputComponent	Physical	input axis direction component to bind (direction of last axis in axisNames)

Return Values

True	if	a binding could be found and updated, false if there was a collision
Binding and Action reference of a collision as {collisionBinding=Binding, collisionAction=Action} or nil		

deleteBinding

Description

Delete a binding.

Definition

deleteBinding(findDeviceId Binding, findActionName Name, findBindingIndex Binding, findAxisComponent Logical)

Arguments

findDeviceId	Binding	device ID
findActionName	Name	of binding action
findBindingIndex	Binding	index, 1 primary, 2 secondary, etc.
findAxisComponent	Logical	binding axis direction component to assign, always positive for buttons/keys/half-axes

getKeyboardMouseInputActiveAndValue

Description

Check if a keyboard or mouse input is active.

Definition

getKeyboardMouseInputActiveAndValue(axes List, axisDirection 1, inputDirection 1)

Arguments

axes	List	of input axis names which are all required to be pressed for the input to be active
axisDirection	1	or -1, determining the direction of the bound logical axis.
inputDirection	1	or -1, determining the direction of the requested physical axis component.

Return Values

float x	x part of quaternion
float y	y part of quaternion
float z	z part of quaternion
float w	w part of quaternion
True if	all given axes are currently active, input value of the non-modified axis (last in list)

getGamepadInputActiveAndValue

Description

Scan gamepad / controller binding input.
Both buttons and controller physical axes are checked for a given device and list of logical axes.

Definition

getGamepadInputActiveAndValue(internalDeviceId Engine-internal, axes List, axisDirection 1, inputDirection 1)

Arguments

internalDeviceId	Engine-internal	device ID of device to scan
axes	List	of axis names to scan
axisDirection	1	or -1, determining the direction of the bound logical axis component.
inputDirection	1	or -1, determining the direction of the requested physical axis component.

Return Values

True if all axes are active (digital interpretation), input value of the non-modified axis (last in list)

getGamepadAxisValue

Description

Scan a controller axis on a device for input.

Definition

getGamepadAxisValue()

Return Values

Input value of the axis in the range [-1, 1] for full axes or [0, 1] for half axes.

update

Description

Update input for the current frame.

Definition

update(dt Delta)

Arguments

dt Delta time in milliseconds

checkGamepadActive

Description

Check if any gamepad button or axis is pressed and set the input help mode if necessary.

Definition

checkGamepadActive()

Return Values

float rotation the final rotation

updateMouseInput

Description

Update mouse input state.

Definition

updateMouseInput()

Return Values

float x x part of quaternion

float y y part of quaternion

float z z part of quaternion

float w w part of quaternion

finalizeMouseInput

Description

Finalize mouse input processing for the current frame.
Wraps the mouse position around if required and resets movement accumulators.

Definition

finalizeMouseInput()

clearActiveBindingBuffer

Description

Clear an active binding buffer for re-use to avoid table allocation each frame.
Automatically adds device entries if new devices are added.

Definition

clearActiveBindingBuffer()

updateEventBindings

Description

Update input bindings associated with all registered and active events.

Definition

updateEventBindings()

hasBindingForPressedMouseComboMask

Description

Check if there is currently any binding matching the pressed mouse combo mask.
This is used to override the combo mask ignore flag for mouse look in special cases. E.g. when rotating a tool axis
with a mouse dragging movement, mouse look should be disabled for the other axis as well, even if it has no active
binding with the same combo button (which would have been handled by input shadowing).

Definition

hasBindingForPressedMouseComboMask()

Return Values

float x x part of quaternion

float y y part of quaternion
float z z part of quaternion
float w w part of quaternion

shadowLinkedBindings

Description

Apply binding input shadow flag to bindings of linked actions for currently active input.

Definition

shadowLinkedBindings()

updateComboBindings

Description

Update binding input for combo action bindings.

Definition

updateComboBindings()

updateInput

Description

Update input bindings with the current input state and trigger events.

Definition

updateInput()

updateBindingInput

Description

Update the input state of a single action-to-input binding.

Definition

updateBindingInput()

Return Values

float x x part of quaternion
float y y part of quaternion
float z z part of quaternion
float w w part of quaternion

resetContinuousEventBindings

Description

Reset binding input state for bindings used in continuous trigger events.

Definition

resetContinuousEventBindings(bool checkComboMasks, int gamepadComboMask, int gamepadMouseMask)

Arguments

bool	checkComboMasks	[optional, default=false] If true, checks binding combo masks against provided parameters
int	gamepadComboMask	[optional] Gamepad combo input bit mask
int	gamepadMouseMask	[optional] Mouse combo input bit mask

neutralizeEventBindingInput

Description

Set an event's binding's input to a neutral value and notify the event's listeners of that neutral position.

Definition

neutralizeEventBindingInput()

updateDebugDisplay

Description

Update and debug information display.

Definition

updateDebugDisplay()

saveToXMLFile

Description

Save action bindings to player custom settings.

Definition

saveToXMLFile()

Return Values

float distance distance to rectangle

saveDeviceSettings

Description

Save all device settings.
Updates and adds settings for active devices.

Definition

saveDeviceSettings(xmlFile Input)

Arguments

xmlFile Input settings XML file handle

Return Values

float distance distance to line segment

getActionList

Description

Get a deep copy of the action definitions list.

Definition

getActionList()

Return Values

boolean hasIntersection true if both lines intersect

float	t1	x position
float	t2	z position

getActionByName

Description

Get a loaded InputAction by name.
Only use the returned action to assign references and do not change its state.

Definition

getActionByName(actionName Name)

Arguments

actionName Name of the action to be returned

Return Values

InputAction

disableAlternateBindingsForAction

Description

Disable alternate bindings for a given action.
This setting is only valid in the current input context and will be overwritten the next time any input event is
changed.

Definition

disableAlternateBindingsForAction(string actionName)

Arguments

string actionName InputAction name

setContext

Description

Set the current input context.
If no context of the given name exists, it will be created.

Definition

setContext(string name, bool createNew, bool deletePrevious)

Arguments

string	name	Input context name
bool	createNew	[optional, default=false] If true, will create a clear context with the given name, overriding any other with the same name.
bool	deletePrevious	[optional, default=false] If true, will delete the previous context after setting the new context.

Return Values

boolean hasIntersection	true if the line segment is completly in the rectangle OR if it intersects with one of the four rectangle sides

revertContext

Description

Revert the input context to the previously set input context.
Use this method only if you can guarantee a valid state, e.g. when calling setContext in a modal interaction context.

Definition

revertContext(bool deleteCurrent)

Arguments

bool deleteCurrent	[optional, default=false] If true, will delete the current input context before reverting to the previous context.

Return Values

float pos1X	x pos of intersection 1, else nil
float pos1Y	y pos of intersection 1, else nil
float pos2X	x pos of intersection 2, else nil
float pos2Z	z pos of intersection 2, else nil

setPreviousContext

Description

Set the previous context for another context.
This modifies the previous context value which is used to revert input contexts.

Definition

setPreviousContext(string forContextName, string previousContextName)

Arguments

string forContextName	Name of the input context whose previous context will be set
string previousContextName	New previous context name

clearAllContexts

Description

Clear all contexts except the root.
The current context will also be set to root.

Definition

clearAllContexts()

getContextName

Description

Get the name of the current input context.

Definition

getContextName()

getActionBindingsCopy

Description

Get an ordered copy of the action bindings table for manipulation.

Definition

getActionBindingsCopy(onlyAssignable If)

Arguments

onlyAssignable	If true, will only copy assignable action bindings

Return Values

boolean	hasIntersection true if spheres have an intersection else false
Ordered action	bindings in the form of {i={action=InputAction, bindings={Binding}}}

getActionBindings

Description

Get the current action bindings table.
The returned table is the current live state and can thus contain unconfirmed modifications.
Treat the table as
read-only so as not to interfere with the correct workings of this class. Only use this getter for initializations
and otherwise rely on setBindingChangeCallback() to receive a confirmed working state.

Definition

getActionBindings()

Return Values

float	area area in hectars

getEventsForActionName

Description

Get an array of registered events in the current input context for a given action name.

Definition

getEventsForActionName(string actionName)

Arguments

string actionName Action name as valid in InputAction

Return Values

float brightness brightness

table Array of events

getFirstActiveEventForActionName

Description

Get the first active event in the current input context for a given action name.

Definition

getFirstActiveEventForActionName(string actionName)

Arguments

string actionName Action name as valid in InputAction

Return Values

float averageSpeed average speed

table First active event or InputEvent.NO_EVENT

setMouseMotionScale

Description

Set the mouse motion scale.
The given scale factor is multiplied with the default mouse motion scale of 0.75 on the X and Y axes.

Definition

setMouseMotionScale(scale Mouse)

Arguments

scale Mouse motion scale factor

Return Values

float speed speed

getMouseMotionScale

Description

Get the current mouse motion scale.

Definition

getMouseMotionScale()

Return Values

float speedRandom speed random

x motion scale, Y motion scale

setBindingChangeCallback

Description

Set a callback for binding changes.
The callback function must accept the action bindings table as a single argument.

Definition

setBindingChangeCallback(callback Callback)

Arguments

callback Callback function which takes a single table parameter in the form of {InputAction: {i=Binding}}

Return Values

float normalSpeed normal speed

setEventChangeCallback

Description

Set a callback for event changes.
The callback function must accept the action-event tuples in an array as a single argument.

Definition

setEventChangeCallback(callback Callback)

Arguments

callback Callback	function which takes a single table parameter in the form of {i={action=InputAction, event=InputEvent}}

Return Values

float tangentSpeed tangent speed

notifyBindingChanges

Description

Notify a listener of binding changes.
The notification callback is set via setBindingChangeCallback().

Definition

notifyBindingChanges()

Return Values

float spriteScaleX X sprite scale

notifyEventChanges

Description

Notify a listener of event changes.
The notification callback is set via setEventChangeCallback().

Definition

notifyEventChanges()

Return Values

float spriteScaleY Y sprite scale

notifyInputModeChange

Description

Notify components of an input mode change via the message center.

Definition

notifyInputModeChange()

Return Values

float spriteScaleXGain X sprite scale gain

checkSettingsIntegrity

Description

Check if user input bindings are corrupted and compare input configuration XML file versions.

Definition

checkSettingsIntegrity()

Return Values

float spriteScaleYGain	Y sprite scale gain
True if	the user settings have passed the integrity check, false otherwise.

checkDefaultInputExclusiveActionBindings

Description

Checks loaded default input bindings according to exclusive locked action groups defined in InputAction.
This is a developer check to verify critical input plausibility.

Definition

checkDefaultInputExclusiveActionBindings()

Return Values

True if there is a collision, false otherwise

getIsPhysicalFullAxis

Description

Determine if an axis name represents a physical full axis on any device.

Definition

getIsPhysicalFullAxis(inputAxisName Axis)

Arguments

inputAxisName Axis name

Return Values

object instance or nil if no object could be created

True if no placeable object could be created at the requested position because there was no valid space, false otherwise

True if the axis name represents a physical full axis, false otherwise

getIsHalfAxis

Description

Determine if an axis name represents a physical half-axis on any device.

Definition

getIsHalfAxis(inputAxisName Axis, True if)

Arguments

inputAxisName Axis name

True if the axis name represents a physical half-axis, false otherwise

getIsAnalogInput

Description

Determine if an input name represents an analog (i.e. continuous) physical input.

Definition

getIsAnalogInput(inputName Input)

Arguments

inputName Input name (potentially a key, button or axis identifier)

Return Values

object instance or nil if no object could be created

True if no placeable object could be created at the requested position because there was no valid space, false otherwise

True if the input name represents an analog input

getIsKeyboardInput

Description

Determine if an input name list represents keyboard input.

Definition

getIsKeyboardInput()

getIsMouseInput

Description

Determine if an input name list represents mouse input.

Definition

getIsMouseInput()

Return Values

float Ray origin X position in world space
float Ray origin Y position in world space
float Ray origin Z position in world space
float Ray direction X component
float Ray direction Y component
float Ray direction Z component

getIsMouseWheelInput

Description

Determine if an input name list represents a single mouse wheel step input.

Definition

getIsMouseWheelInput()

getIsGamepadInput

Description

Determine if an input name list represents gamepad / controller input.

Definition

getIsGamepadInput()

getIsDPadInput

Description

Check if an input name list represents a single D-Pad input.

Definition

getIsDPadInput()

isAxisZero

Description

Check if the input value of an axis can be considered zero.

Definition

isAxisZero(value Axis)

Arguments

value Axis input value

Return Values

True if the axis is at zero position, false otherwise

getMaximumBindingCountForDeviceId

Description

Get the maximum number of bindings per action for a given device ID.

Definition

getMaximumBindingCountForDeviceId()

getDeviceCategory

Description

Get the device category for a device identified by its engine-internal ID

Definition

getDeviceCategory()

Return Values

any_type unpackedValues returns unpacked values found in string

InputDevice

Description

Game input device.
-- The class stores name and IDs of a device as well as physical to logical key / button / axis mappings.
--@category Input
--@xmlConfig device#id Device ID, must match a value returned by engine function getGamepadId() or one of the constants
of InputDevice.DEFAULT_DEVICE_NAMES.

new

Description

Create a new InputDevice instance.

Definition

new(internalId Internal, deviceId Device, deviceName Device, category Device)

Arguments

internalId	Internal	device ID as issued by the engine (~ device index)
deviceId	Device	ID as received by the operating system
deviceName	Device	name as assembled by the engine
category	Device	category, use one of the values of InputDevice.CATEGORY

Return Values

table values	values
New InputDevice instance	

loadSettingsFromXML

Description

Load device settings from an XML file.
This will load settings other than input bindings, e.g. sensitivity for gamepad axes.

Definition

loadSettingsFromXML(xmlFile XML, deviceElement XML)

Arguments

xmlFile	XML file handle
deviceElement	XML data element for this device

Return Values

table values radian values

saveSettingsToXML

Description

Save device settings to an XML file.

Definition

saveSettingsToXML(xmlFile XML, deviceElement XML)

Arguments

xmlFile XML file handle

deviceElement XML data element for this device

Return Values

table result text elements

isController

Description

Determine if this device is a controller/gamepad.

Definition

isController()

Return Values

boolean startsWidth true if given string starts with pattern else false

loadIdFromXML

Description

Load a device's ID from an XML file.

Definition

loadIdFromXML(xmlFile XML, deviceElement XML)

Arguments

xmlFile XML file handle

deviceElement XML data element for this device

Return Values

boolean startsWidth true if given string ends with pattern else false

Device ID or empty string

loadNameFromXML

Description

Load a device's name from an XML file.

Definition

loadNameFromXML(xmlFile XML, deviceElement XML)

Arguments

xmlFile XML file handle

deviceElement XML data element for this device

Return Values

string trimedString trimed text

Device name or empty string

loadCategoryFromXML

Description

Load a device's category from an XML file.

Definition

loadCategoryFromXML(xmlFile XML, deviceElement XML, Device category)

Arguments

xmlFile XML file handle

deviceElement XML data element for this device

Device category

Return Values

any_type value not nil value

getDeviceIdPrefix

Description

Get a device ID prefix' numeric value, if it exists.

Definition

getDeviceIdPrefix()

Return Values

float lx x direction

float lz z direction

Prefix number, or -1 if none exists; raw engine-issued device ID

getPrefixedDeviceId

Description

Get a device ID prefixed with a given number.

Definition

getPrefixedDeviceId()

toString

Description

Get a string representation for this device.

Definition

toString()

Return Values

float lx average x direction

float lz average z direction

InputDisplayManager

Description

Bridge-component between input handling and UI.
-- Serves data and display elements for input controls. Accesses InputBinding data and is being accessed by UI
components.
--@category Input

load

Description

Load the required base data for this manager.
Loads controller symbols / UI overlays.

Definition

load()

setDevGamepadLabelMapping

Description

Set up development-only gamepad label function if necessary.

Definition

setDevGamepadLabelMapping()

Return Values

boolean isAttached is attached

loadControllerSymbolsAndOverlays

Description

Load controller symbols and create UI overlays.
Reads a UV values from an XML configuration to create UI overlays from a texture atlas.

Definition

loadControllerSymbolsAndOverlays()

Return Values

boolean doesBlock implement does block

createButtonOverlay

Description

Create a single button overlay.

Definition

createButtonOverlay()

Return Values

integer aiToolReverserDirectionNode reverser direction node of ai tool

loadAxisIcons

Description

Load vehicle axis input hint icons.
The icons are kept in memory as overlays until the game is closed.

Definition

loadAxisIcons(xmlFile XML, modPath [optional])

Arguments

xmlFile XML file handle

modPath [optional] Path to loaded mod, will load defaults if set to nil

Return Values

float maxTurnRadius max turn radius

loadModAxisIcons

Description

Load axis icons defined by mods.

Definition

loadModAxisIcons()

Return Values

float leftAreaPercentage left area percentage

float rightAreaPercentage right area percentage

addContextBindings

Description

Add bindings of an action to a list if they fit the required context (gamepad or keyboard and mouse),

Definition

addContextBindings(contextBindings Bindings, action InputAction, isContextGamepad If)

Arguments

contextBindings	Bindings	list for the required context
action	InputAction	reference
isContextGamepad	If	true, we require gamepad bindings. Otherwise, keyboard and mouse is needed.

getActionBindingsForContext

Description

Get bindings from one or two actions for the current input context (either gamepad or keyboard and mouse).

Definition

getActionBindingsForContext()

Return Values

float area	area found
float totalArea	total area checked

getKeyboardBindings

Description

Get bindings for one or two actions for keyboard input, exclusively.

Definition

getKeyboardBindings()

resolveModifierSymbols

Description

Resolve modifier symbols and add them to an overlays collection.

Definition

resolveModifierSymbols(table overlays, table separators, table firstContextBinding)

Arguments

table overlays	Overlays collection which receives resolved modifier symbols
table separators	Array of separator types inbetween symbols
table firstContextBinding	Binding reference

Return Values

float area	area found
float totalArea	total area checked

resolveAccumulatedSymbolPermutations

Description

Recursively permutate symbol names to resolve dynamically ordered names to fixed definitions.

Definition

resolveAccumulatedSymbolPermutations()

resolveUnmodifiedSymbols

Description

Resolve unmodified binding axis names to symbol overlays and add those to the given overlays collection.

Definition

resolveUnmodifiedSymbols(table overlays, table contextBindings, bool isContextGamepad)

Arguments

table overlays	Overlays collection which receives resolved symbols
table contextBindings	Array of Binding instances
bool isContextGamepad	If true, resolve gamepad symbols. Otherwise, resolve mouse symbols.

Return Values

float area	area found
float totalArea	total area checked

addRegularSymbols

Description

Add regular input display symbols to an overlays array.

Definition

addRegularSymbols()

addComboSymbols

Description

Add combo input display symbols to an overlays array.

Definition

addComboSymbols()

Return Values

table instance	instance of object

getControllerSymbolOverlays

Description

Get input symbol overlays for one or two actions.
Use the second action parameter to display complementary actions, e.g. action1 is moving forward/backward, action 2
is moving left/right.

Definition

getControllerSymbolOverlays(actionName1 First, actionName2 [optional])

Arguments

actionName1 First	action name as defined and/or loaded in InputAction
actionName2 [optional]	Second action name as defined and/or loaded in InputAction

Return Values

integer	numOfConfigurationTypes number of configuration types
InputHelpElement instance	containing the symbol overlays in its "buttons" field

requireSymbolAccumulation

Description

Determine if an action and bindings combinations requires their input symbols to be accumulated to get a combined
symbol for better display.

Definition

requireSymbolAccumulation()

Return Values

integer	numOfConfigurationTypes number of configuration types

makeHelpElement

Description

Make a help element to display in the HUD.

Definition

makeHelpElement()

Return Values

string name name of config

onActionEventsChanged

Description

Called when action events in the input system have changed.

Definition

onActionEventsChanged(displayActionEvents Array)

Arguments

displayActionEvents	Array	of action-event tuples which require an input hint display, {i={action=InputAction, event=InputEvent}}

Return Values

integer index index of config

sortEventHelpElements

Description

Table sorting function for event help elements.
Priority > primaryKeyboardInput > text

Definition

sortEventHelpElements()

Return Values

table configuration configuration

sortEventHelpElementsGamepad

Description

Sorting function for gamepad mode

Definition

sortEventHelpElementsGamepad()

Return Values

any_type value value of attribute

storeEventHelpElements

Description

Store display help elements for currently active action events.

Definition

storeEventHelpElements()

Return Values

boolean configurationHasBeenBought configuration has been bought

storeComboHelpElements

Description

Store combo button help elements for any modifier buttons required by currently active action events.

Definition

storeComboHelpElements()

Return Values

table color color (r, g, b)

getEventHelpElementForAction

Description

Get the input help element for a given action.

Definition

getEventHelpElementForAction(inputActionName Name)

Arguments

inputActionName Name of the requested input action.

Return Values

integer material material

getEventHelpElements

Description

Get input help elements for the current input context.

Definition

getEventHelpElements(pressedComboMask Bit, isContextGamepad If)

Arguments

pressedComboMask	Bit	mask of the currently pressed combo button input
isContextGamepad	If	true, the player should be shown input hints for gamepad input. Otherwise, keyboard / mouse input help is required.

Return Values

any_type	value	value of config
array	of	help elements (see InputDisplayManager:makeHelpElement() for structure)

getComboHelpElements

Description

Get input help elements for currently available combo buttons.

Definition

getComboHelpElements(isContextGamepad If)

Arguments

isContextGamepad	If	true, the player should be shown input hints for gamepad input. Otherwise, keyboard / mouse input help is required.

Return Values

string	configKey	key of configuration
integer	configIndex	index of configuration
Hash	table	of currently active combo button action names, {action name=true}

getPrefix

Description

Get the controller symbol prefix for a device identified by its internal ID.

Definition

getPrefix()

getPlusOverlay

Description

Get the "plus" sign overlay.
This overlay is (re-)used in several places. Take care not to invalidate its state between uses.

Definition

getPlusOverlay()

Return Values

table color color (r, g, b)

getOrOverlay

Description

Get the "or" sign overlay.
This overlay is (re-)used in several places. Take care not to invalidate its state between uses.

Definition

getOrOverlay()

Return Values

table instance instance of object

getKeyboardKeyOverlay

Description

Get the keyboard key glyph overlay (ButtonOverlay instance).
This overlay is (re-)used in several places. Take care not to invalidate its state between uses.

Definition

getKeyboardKeyOverlay()

Return Values

boolean true if loading was successful else false

getFirstBindingAxisAndDeviceForActionName

Description

Get the first input axis name of the first binding of an input action identified by name.

Definition

getFirstBindingAxisAndDeviceForActionName(inputActionName Name, axisComponent [optional,, isGamepad [optional,)

Arguments

inputActionName	Name	of an action as defined in InputAction
axisComponent	[optional,	default=POSITIVE] Binding.AXIS_COMPONENT value
isGamepad	[optional,	default=false] If true, return the first gamepad binding. Otherwise, keyboard is requested.

Return Values

boolean success true if added else false

Axis name or nil if not found, internal ID of binding device

getGamepadInputActionOverlay

Description

Get an input action glyph overlay for the first gamepad button of the first binding of a given input action.
The method is simplified by design to mainly provide menu button glyphs. It will only return an overlay for the
first input axis, i.e. a simple button binding without modifiers. The returned overlay is a new instance and callers
need to delete it after use via GuiOverlay.deleteOverlay().

Definition

getGamepadInputActionOverlay(inputActionName Name, axisComponent [optional,)

Arguments

inputActionName Name	of an action as defined in InputAction
axisComponent	[optional, default=POSITIVE] Binding.AXIS_COMPONENT value

Return Values

boolean hasPrerequisite	true if all prerequisite specializations are loaded
Overlay instance	or nil if no overlay is available for the action's bindings

getKeyboardInputActionKey

Description

Get an input action key text for the key of the first binding of a given input action.

Definition

getKeyboardInputActionKey(inputActionName Name, axisComponent [optional,)

Arguments

inputActionName Name	of an action as defined in InputAction
axisComponent	[optional, default=POSITIVE] Binding.AXIS_COMPONENT value

Return Values

boolean hasPrerequisite	true if all prerequisite specializations are loaded
key text	(e.g. "F" for "KEY_f") for the keyboard action binding or nil if no resolution is possible

getGamepadInputSymbolName

Description

Get the controller symbol key name for a single gamepad / controller binding.
For full axes, call this for each axis component binding and concatenate the symbol keys with a space inbetween.

Definition

getGamepadInputSymbolName(internalDeviceId Internal, axisName Input, isAxisInput If)

Arguments

internalDeviceId	Internal	gamepad device ID
axisName	Input	axis name
isAxisInput	If	true, the bound input axis is an analog physical axis

Return Values

boolean hasPrerequisite	true if all prerequisite specializations are loaded
controller symbols	configuration key name

getMouseInputSymbolName

Description

Get the controller symbol key name for a single mouse binding.
For full axes, call this for each axis component binding and concatenate the symbol keys with a space inbetween.

Definition

getMouseInputSymbolName(axisNames Mouse)

Arguments

axisNames Mouse	binding axis names

Return Values

boolean canStart	can start ai
controller symbols	configuration key name

onActionBindingsChanged

Description

Called when action bindings have changed.

Definition

onActionBindingsChanged(actionBindings Action)

Arguments

actionBindings Action bindings table, {InputAction: {Binding}}

Return Values

boolean canContiue can contiue ai

InputEvent

Description

Input event.
-- Holds information about an input event for an action, target, event callback and event trigger conditions. Also
stores input hint display information, because only actions with associated events need hints. When an action has
several events, callers need to make sure only one of the events' visibility is active.
--@category Input

new

Description

Create a new instance of InputEvent.
This constructor also assigns an ID to the instance, which should be used as a handle in most cases instead of the
actual instance reference.

Definition

new(actionName Name, targetObject Event, eventCallback Event, triggerUp If, triggerDown If, triggerAlways If, startActive [optional,, callbackState [optional])

Arguments

actionName	Name	of action, see InputAction
targetObject	Event	target, first argument to event callback
eventCallback	Event	callback, called when the action has input. Signature: callback(targetObject, actionName, inputValue, callbackState, isAnalog)
triggerUp	If	true, the event fires once when an input signal of the given action goes to "up" state, e.g. a released key.
triggerDown	If	true, the event fires once when an input signal of the given action goes to "down" state, e.g. a pressed key.
triggerAlways	If	true, the event fires on any signal input change, potentially every frame. This is mostly useful for required continuous axis input, e.g. steering actions.
startActive	[optional,	default=false] If true, the event is active right after registration.
callbackState	[optional]	An arbitrary value or reference which serves as a closure state within callbacks, useful e.g. when checking input state between events without requiring class fields.

Return Values

float direction shape angle

New InputEvent instance

setIgnoreComboMask

Description

Set the ignore flag for input binding combo masks.
If true, bindings which can trigger this event can be activated even if the do not match the currently pressed combo
mask. This is required for special cases, mostly player camera input, which should still be active when a combo
button is pressed.

Definition

setIgnoreComboMask()

Return Values

boolean isNeeded collision box is needed

getIgnoreComboMask

Description

Get the ignore flag for input binding combo masks.

Definition

getIgnoreComboMask()

Return Values

string attributes stats attributes

notifyInput

Description

Notify this event of the input status of an associated action.
-- Note the interaction of the "down" and "always" triggers: If "down" is set and not "always", only an input down-flank
will raise an event. If both "down" and "always" are set, an event is raised as long as the input is pressed beyond a
threshold. If only "always" and not "down" is set, an event will be raised on each frame with the current input value
even if no input is active.
-- @param isUp If true, the action input had an "up" flank, i.e. the input was in "pressed" state and has returned to zero position

Definition

notifyInput(isDown If, isPressed If, inputValue Action, isAnalog True)

Arguments

isDown	If	true, the action input had a "down" flank, i.e. the input has crossed the "down" threshold
isPressed	If	true, the action input is in "pressed" state, i.e. the magnitude of the input is above the "down" threshold
inputValue	Action	input value. This is not guaranteed to be in the range [-1, 1], because of sensitivity settings. Event handlers may clamp input values at their own discretion.
isAnalog	True	if the binding which provides the input value is analog.

Return Values

boolean deactivateOnLeave deactivate on leaving

frameReset

Description

Reset this event after a frame.

Definition

frameReset()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

makeId

Description

Make this InputEvent instance's ID.

Definition

makeId()

Return Values

boolean exists animation axists

getTriggerCode

Description

Return trigger code

Definition

getTriggerCode()

Return Values

boolean isPlaying animation is playing

initializeDisplayText

Description

Initialize this event's input hint display text with information from its associated action. Defaults to the action's positive display name. For most half-axis actions, this will be correct. If a more specific
text is required depending on the current player context, apply a different text through the input system using
InputBinding:setActionEventText().

Definition

initializeDisplayText(InputAction InputAction)

Arguments

InputAction InputAction which this event references.

Return Values

float animTime real animation time in ms

InputGlyphElement

Description

Input glyph display element.
-- Displays a key or button glyph for an input.
--@category GUI

new

Description

Create a new instance of InputGlyphElement.

Definition

new(table inputDisplayManager, float baseWidth, float baseHeight)

Arguments

table	inputDisplayManager	InputDisplayManager reference
float	baseWidth	Default width of this element in screen space
float	baseHeight	Default height of this element in screen space

Return Values

table npc the npc object

delete

Description

Delete this element and release its resources.

Definition

delete()

Return Values

table instance instance of object

deleteOverlayCopies

Description

Delete any overlay copies.

Definition

deleteOverlayCopies()

Return Values

boolean true if loading was successful else false

setScale

Description

Set the scale of this element.

Definition

setScale(float widthScale, float heightScale)

Arguments

float widthScale Width scale factor

float heightScale Height scale factor

Return Values

bool true if successful

setUpperCase

Description

Set the glyph text to be displayed in all upper case or not.
This resets the lower case setting if upper case is enabled.

Definition

setUpperCase()

Return Values

bool true if successful

setLowerCase

Description

Set the glyph text to be displayed in all lower case or not.
This resets the upper case setting if lower case is enabled.

Definition

setLowerCase()

Return Values

bool true if normalized

setBold

Description

Set the glyph text to be displayed in bold print or not.

Definition

setBold()

Return Values

bool true if normalized

setKeyboardGlyphColor

Description

Set the button frame color for the keyboard glyphs.

Definition

setKeyboardGlyphColor(table color)

Arguments

table color Color as an RGBA array

Return Values

table returns a food group or nil if nothing is found

setButtonGlyphColor

Description

Set the color for button glyphs.

Definition

setButtonGlyphColor(table color)

Arguments

table color Color as an RGBA array

Return Values

table returns a food group or nil if nothing is found

setAction

Description

Set the action whose input glyphs need to be displayed by this element.

Definition

setAction(string actionName, string actionText, float actionTextSize, bool noModifiers, bool copyOverlays)

Arguments

string	actionName	InputAction name
string	actionText	[optional] Additional action text to display after the glyph
float	actionTextSize	[optional] Additional action text size in screen space
bool	noModifiers	[optional] If true, will only show the input glyph of the last unmodified input binding axis
bool	copyOverlays	[optional] If true, will create and handle a separate copy of an input glyph. Do not use this when updating the action each frame!

Return Values

table returns a food mixture or nil if nothing is found

setActions

Description

Set multiple actions whose input glyphs need to be displayed by this element.
If exactly two actions are passed in, they will be interpreted as belonging to the same axis and

the system tries
to resolved the actions to a combined glyph. Otherwise, the glyphs will just be displayed in order of the actions.

Definition

setActions(table actionNames, string actionText, float actionTextSize, bool noModifiers, bool copyOverlays)

Arguments

table	actionNames	InputAction names array
string	actionText	[optional] Additional action text to display after the glyph
float	actionTextSize	[optional] Additional action text size in screen space
bool	noModifiers	[optional] If true, will only show the input glyph of the last unmodified input binding axis
bool	copyOverlays	[optional] If true, will create and handle a separate copy of an input glyph. Do not use this when updating the action each frame!

Return Values

int Food	consumption type, one of [AnimalFoodManager.FOOD_CONSUME_TYPE_SERIAL | AnimalFoodManager.FOOD_CONSUME_TYPE_PARALLEL]

updateDisplayText

Description

Update the display text from the set action text according to current casing settings.

Definition

updateDisplayText()

Return Values

table returns a food group or nil if not found

getGlyphWidth

Description

Get the screen space width required by the glyphs used to display input in the current input context.

Definition

getGlyphWidth()

Return Values

float returns a production weight between 0 and 1

draw

Description

Draw the input glyph(s).

Definition

draw()

Return Values

table instance Instance of object

drawControllerButtons

Description

Draw controller button glyphs.

Definition

drawControllerButtons(table Array, float posX, float posY)

Arguments

table Array of controller button glyph overlays
float posX Initial drawing X position in screen space
float posY Initial drawing Y position in screen space

Return Values

bool returns true if loading is fine
float X position in screen space after the last glyph

drawKeyboardKeys

Description

Draw keyboard key glyphs.

Definition

drawKeyboardKeys(table Array, float posX, float posY)

Arguments

table Array of keyboard key names
float posX Initial drawing X position in screen space
float posY Initial drawing Y position in screen space

Return Values

bool returns true if module init after placement is fine
float X position in screen space after the last glyph

drawActionText

Description

Draw the action text after the input glyphs.

Definition

drawActionText(float posX, float posY)

Arguments

float posX Drawing X position in screen space
float posY Drawing Y position in screen space

Return Values

InputHelpDisplay

Description

HUD input help display.
-- Displays controls and further information for the current input context.
--@category GUI

new

Description

Create a new instance of InputHelpDisplay.

Definition

new(string hudAtlasPath)

Arguments

string hudAtlasPath Path to the HUD texture atlas

Return Values

addHelpText

Description

Add a help text line for this frame.
Will be cleared after the current frame.

Definition

addHelpText()

Return Values

bool returns true if initialization is fine

setVehicle

Description

Set the currently controlled vehicle.

Definition

setVehicle(table vehicle)

Arguments

table vehicle Vehicle reference or nil if no vehicle is controlled.

Return Values

string fill level

getHidingTranslation

Description

Override of HUDDisplayElement.
Moves out to the left to hide.

Definition

getHidingTranslation()

Return Values

integer total animals. Default is zero

addCustomEntry

Description

Add a custom input help entry which is displayed in the current input context until removed.
Custom entries will be displayed in order of addition after any automatically detected input help entries and before
vehicle extensions.

Definition

addCustomEntry()

Return Values

string return animal type. nil if not available

clearCustomEntries

Description

Clear all custom input help entries in the current context.

Definition

clearCustomEntries()

Return Values

table of cosumed food (result[fillTypeIndex]=amount)

update

Description

Update the input help's state.

Definition

update()

Return Values

table Production fill type information as {i={j={fillType=fillType, fillLevel=fillLevel, capacity=capacity}}}

updateSizeAndPositions

Description

Update sizes and positions of this elements and its children.

Definition

updateSizeAndPositions()

Return Values

refreshHUDExtensions

Description

Create any required HUD extensions when the current vehicle configuration changes.

Definition

refreshHUDExtensions()

Return Values

bool true if registration went well

updateHUDExtensions

Description

Update HUD extensions if controlled vehicles have changed.

Definition

updateHUDExtensions()

Return Values

bool true if registration went well

collectVehicleSpecializations

Description

Recursively get vehicle specializations for a vehicle configuration.

Definition

collectVehicleSpecializations()

Return Values

float dirty factor [0-1]

getAvailableHeight

Description

Get the available screen space height for displaying input help.

Definition

getAvailableHeight()

Return Values

table instance instance of object

updateInputContext

Description

Update display elements with the current input context.

Definition

updateInputContext()

Return Values

boolean true if loading was successful else false

updateEntries

Description

Update entry glyphs and visibility with the current input help elements.

Definition

updateEntries(table helpElements)

Arguments

table helpElements Array of InputHelpElement for the current input context

Return Values

bool true if successful

updateComboHeaders

Description

Update combo header state.

Definition

updateComboHeaders(bool useGamepadButtons, int pressedComboMaskMouse, int pressedComboMaskGamepad)

Arguments

bool	useGamepadButtons	If true, check gamepad input. Otherwise, check keyboard / mouse.
int	pressedComboMaskMouse	Bit mask of pressed mouse combo actions
int	pressedComboMaskGamepad	Bit mask of pressed gamepad combo actions

Return Values

bool	true if animal is in the husbandry
float	Screen space height used by the combo header (0 if invisible)

updateComboInputGlyphs

Description

Update visibility and color of combo input glyphs.

Definition

updateComboInputGlyphs(table comboActionStatus, int pressedComboMaskMouse, int pressedComboMaskGamepad)

Arguments

table	comboActionStatus	Hashtable of combo action names which are currently available
int	pressedComboMaskMouse	Bit mask of pressed mouse combo actions
int	pressedComboMaskGamepad	Bit mask of pressed gamepad combo actions

Return Values

table

getInputHelpElements

Description

Get input help elements based on the current input context.

Definition

getInputHelpElements(float availableHeight, int pressedComboMaskGamepad, int pressedComboMaskMouse, bool useGamepadButtons)

Arguments

float	availableHeight	Maximum available height to use for input help elements
int	pressedComboMaskGamepad	Bit mask of pressed gamepad combo buttons
int	pressedComboMaskMouse	Bit mask of pressed mouse combo buttons
bool	useGamepadButtons	If true, we should draw gamepad / controller combo button glyphs

Return Values

table	instance	instance of object
table	Input	help elements
float	Screen	space height used by the returned help elements

setAnimationAvailableHeight

Description

Set the current animation value for available height.

Definition

setAnimationAvailableHeight()

Return Values

boolean true if loading was successful else false

setAnimationOffset

Description

Set the current animation position offset.

Definition

setAnimationOffset()

Return Values

bool true if successful

animateHide

Description

Animate this element on hiding.

Definition

animateHide()

Return Values

bool true if successful

animateShow

Description

Animate this element on showing.

Definition

animateShow()

Return Values

float amount effectively changed

onAnimateVisibilityFinished

Description

Called when a hiding or showing animation has finished.

Definition

onAnimateVisibilityFinished()

Return Values

float delta of amount changed

onInputDevicesChanged

Description

Called when the connected input devices have changed.

Definition

onInputDevicesChanged()

Return Values

float returns the fillLevel of a fillType

draw

Description

Draw the input help.
Only draws if the element is visible and there are any help elements.

Definition

draw()

Return Values

float returns the fillLevel of all fillType

drawControlsLabel

Description

Draw the "controls" label on top of the display frame.

Definition

drawControlsLabel()

Return Values

float returns the capacity of a fillType

drawHelpInfos

Description

Draw icons and text in help entries.

Definition

drawHelpInfos()

Return Values

float returns a progress of the fillType between 0 and 1. Default is 0.

drawVehicleHUDExtensions

Description

Draw vehicle HUD extensions.

Definition

drawVehicleHUDExtensions()

Return Values

float actual delta filled

setScale

Description

Set this element's UI scale.

Definition

setScale(float uiScale)

Arguments

float uiScale UI scale factor

Return Values

storeScaledValues

Description

Store scaled positioning, size and offset values.

Definition

storeScaledValues()

Return Values

table module instance or nil of no module has been registered

getBackgroundPosition

Description

Get this element's base background position in screen space.

Definition

getBackgroundPosition()

Return Values

table instance instance of object

getTopLeftPosition

Description

Get the current top left position of the input help, including animation.

Definition

getTopLeftPosition()

Return Values

boolean true if loading was successful else false

getMaxEntryCount

Description

Get the maximum number of help entries which may be shown at any time.

Definition

getMaxEntryCount()

Return Values

float returns the capacity of a fillType

int Maximum number of entries to show

getIsHelpElementAllowed

Description

Returns if it is still allowed to add more help elements

Definition

getIsHelpElementAllowed(table helpElements, table helpElement)

Arguments

table helpElements existing table of help elements

table helpElement help element to add

Return Values

float amount effectively changed

boolean isAllowed isAllowed

setDimension

Description

Set this element's dimensions.
Override from HUDElement which adjusts frame with offset.

Definition

setDimension()

Return Values

boolean true if loading was successful else false

createBackground

Description

Create the background overlay for positioning.

Definition

createBackground()

Return Values

float food dropped

createComponents

Description

Create required display components.

Definition

createComponents()

Return Values

float spillage total

createFrame

Description

Create the frame around input help elements.

Definition

createFrame()

Return Values

table instance instance of object

createVerticalSeparator

Description

Create a vertical separator element.

Definition

createVerticalSeparator()

Return Values

boolean true if loading was successful else false

createHorizontalSeparator

Description

Create a horizontal separator element.

Definition

createHorizontalSeparator()

Return Values

table instance instance of object

createComboInputGlyph

Description

Create an input glyph for displaying combo input buttons.

Definition

createComboInputGlyph(float posX, float posY, float width, float height, string actionName)

Arguments

float	posX	Screen space X position
float	posY	Screen space Y position
float	width	Screen space width
float	height	Screen space height
string	actionName	InputAction name of the combo action whose input glyphs need to be displayed

Return Values

boolean	true if loading was successful else false
table	Combo InputGlyphElement instance

createComboHeader

Description

Create a combo input glyph header.

Definition

createComboHeader(hudAtlasPath Path, frameX Screen, frameY Screen, table combos, table boxSize, table separatorPositions)

Arguments

hudAtlasPath	Path	to HUD texture atlas
frameX	Screen	space X position of the display frame
frameY	Screen	space Y position of the display frame
table	combos	Array of input combination descriptions as defined in InputBinding
table	boxSize	2D vector which holds the pixel size of one combo input box within the header
table	separatorPositions	Array of 2D vectors of separator pixel positions

Return Values

float	manure dropped
table	Combo header HUDElement

createMouseComboHeader

Description

Create the mouse combo header.

Definition

createMouseComboHeader()

Return Values

float manure fill level

createControllerComboHeader

Description

Create the gamepad / controller combo header.

Definition

createControllerComboHeader()

Return Values

float manure used

createEntry

Description

Create an input help entry box.

Definition

createEntry()

Return Values

table instance instance of object

createEntries

Description

Create all required input help entry boxes.
The boxes are modified as required to reflect the current input context.

Definition

createEntries()

Return Values

boolean true if loading was successful else false

InputHelpElement

Description

Input help element.
-- Holds help and hint information about the current input context for display in the HUD.
--@category Input

getActionNames

Description

Get an array of action names which were validly set in the constructor.

Definition

getActionNames()

Return Values

float animTime animation time [0..1]

InputHelper

Description

Maps key IDs to localization glyph symbols.

MouseHelper.getInputDisplayText

Description

Get a display string for a list of mouse input axis names

Definition

MouseHelper.getInputDisplayText()

Return Values

float duration duration in ms

KeyboardHelper.getDisplayKeyName

Description

Get a display name for a given key ID.
Encapsulates engine getKeyName() function.

Definition

KeyboardHelper.getDisplayKeyName()

Return Values

boolean rightOrder returns true if parts are in right order

KeyboardHelper.getInputDisplayText

Description

Get a display string for a list of keyboard input keys.

Definition

KeyboardHelper.getInputDisplayText()

Return Values

boolean rightOrder returns true if parts are in reverse right order

GamepadHelper.getInputDisplayText

Description

Get a display string for a list of gamepad/controller input button/axis names and an associated device.

Definition

GamepadHelper.getInputDisplayText(inputList List, internalDeviceId ID)

Arguments

inputList	List of input button / axis names
internalDeviceId	ID of device as issued by the engine

Return Values

float ret limited value

JoinGameScreen

Description

Join Multiplayer Game Screen.
-- @field mainBox Layout box for most of the screen, used to quickly show / hide everything.

onClickHeader

Description

Handle clicks on table header elements
Triggers sorting and a view update.

Definition

onClickHeader()

Return Values

table instance instance of object

onServerNameChanged

Description

Handle changes in the server name filter field

Definition

onServerNameChanged(element TextInputElement, text New)

Arguments

element TextInputElement instance

text New text

Return Values

table baleType baleType object

saveFilterSettings

Description

Save the current filter settings.

Definition

saveFilterSettings()

Return Values

table brandColor brandColor object

getSelectedServer

Description

Get the server info of the currently selected server

Definition

getSelectedServer()

Return Values

table instance instance of object

setStartButtonState

Description

Assign the "Start Game" button state.
Enables the button if the currently selected server is valid for playing. Disables it otherwise.

Definition

setStartButtonState()

Return Values

boolean true if loading was successful else false

filterServer

Description

Filter a server with the current filter settings.

Definition

filterServer(server Server)

Arguments

server Server information (see onServerInfo()) for data structure

Return Values

boolean true if loading was successful else false

True if the server matches the current filter settings, false otherwise

setTableFiltersAndSorting

Description

Apply server filters to server list.

Definition

setTableFiltersAndSorting()

Return Values

boolean success success

rebuildServerList

Description

Update the server table.
This causes a full rebuild of the table data.

Definition

rebuildServerList()

Return Values

table instance instance of object

buildServerDataRow

Description

Build a DataRow from server properties to add to the server list GuiElement.

Definition

buildServerDataRow()

Return Values

table instance instance of object

dataRow instance with server data

ListItemElement

Description

List item element to be used and laid out by ListElement and TableElement.
-- Used layers: "image" for a background image.
--@category GUI
--@xmlConfig GuiElement#allowSelected bool [optional] If false, this element cannot be selected, only focused or activated.

getFocusTarget

Description

Get the actual focus target, in case a child or parent element needs to be targeted instead.
Override from BitmapElement: Only focuses children if #autoSelectChildren is true

Definition

getFocusTarget(incomingDirection (Optional), moveDirection (Optional))

Arguments

incomingDirection (Optional) If specified, may return different targets for different incoming directions.

moveDirection (Optional) Actual movement direction per input. This is the opposing direction of incomingDirection.

Return Values

GuiElement Actual element to focus.

MainScreen

Description

Main Menu Screen.
-- @field backgroundImage Main background image

setupNotifications

Description

Set up notifications display and animation.

Definition

setupNotifications()

Return Values

float nearCoCRadius	Near circle of confusion radius (nearCoCRadius = 0 means no near blur (pinhole camera))
float nearBlurEnd	Distance from the camera center where near blur ends
float farCoCRadius	Far circle of confusion radius (farCoCRadius = 0 means no far blur (pinhole camera))
float farBlurStart	Distance from the camera center where far blur starts
float farBlurEnd	Distance from the camera center where far blur ends

setNotificationButtonsDisabled

Description

Set notification buttons disabled state.
If fewer than 2 notifications are available to show, the cycle buttons will always be disabled.

Definition

setNotificationButtonsDisabled()

resetNotifications

Description

Reset notification data and display.

Definition

resetNotifications()

cycleNotification

Description

Cycle through available notifications.

Definition

cycleNotification(int signedDelta)

Arguments

int signedDelta 1 or -1 for next or previous

onNotificationBoxClick

Description

Handle an activation of the notification open button or input action.

Definition

onNotificationBoxClick()

updateNotifications

Description

Update notifications.

Definition

updateNotifications()

Return Values

float nearCoCRadius	Near circle of confusion radius (nearCoCRadius = 0 means no near blur (pinhole camera))
float nearBlurEnd	Distance from the camera center where near blur ends
float farCoCRadius	Far circle of confusion radius (farCoCRadius = 0 means no far blur (pinhole camera))
float farBlurStart	Distance from the camera center where far blur starts
float farBlurEnd	Distance from the camera center where far blur ends

updateFading
Description

Update screen fading values.

Definition

updateFading()

MapDataGrid
Description

Creating data grid

new
Description

@param table customMt custom metatable

Definition

new()

Return Values

table instance instance of object

Code

```
19  function MapDataGrid:new(mapSize, blocksPerRowColumn, customMt)
20  local self = DataGrid:new(blocksPerRowColumn, blocksPerRowColumn,
    customMt or MapDataGrid_mt)
21
22  self.blocksPerRowColumn = blocksPerRowColumn
23  self.mapSize = mapSize
24  self.blockSize = self.mapSize/self.blocksPerRowColumn
25
26  return self
27  end
```

getValueAtWorldPos
Description

@param float worldZ world position z

Definition

getValueAtWorldPos()

Return Values

table value value at the given position

Code

```
34  function MapDataGrid:getValueAtWorldPos(worldX, worldZ)
35  local rowIndex, colIndex = self:getRowColumnFromWorldPos(worldX,
    worldZ)
36  return self:getValue(rowIndex, colIndex), rowIndex, colIndex
37  end
```

setValueAtWorldPos
Description

@param float worldZ world position z

Definition

setValueAtWorldPos(table value)

Arguments

table value value at the given position

Code

```
44  function MapDataGrid:setValueAtWorldPos(worldX, worldZ, value)
45  local rowIndex, colIndex = self:getRowColumnFromWorldPos(worldX,
    worldZ)
46  self:setValue(rowIndex, colIndex, value)
47  end
```

getRowColumnFromWorldPos

Description

@param float worldZ world position z

Definition

getRowColumnFromWorldPos()

Return Values

integer row row

integer column column

Code

```
55  function MapDataGrid:getRowColumnFromWorldPos(worldX, worldZ)
56  local mapSize = self.mapSize
57  local blocksPerRowColumn = self.blocksPerRowColumn
58
59  local x = (worldX + mapSize*0.5) / mapSize
60  local z = (worldZ + mapSize*0.5) / mapSize
61
62  local row = MathUtil.clamp(math.ceil(blocksPerRowColumn*z), 1,
    blocksPerRowColumn)
63  local column = MathUtil.clamp(math.ceil(blocksPerRowColumn*x), 1,
    blocksPerRowColumn)
64
65  -- log(worldX, worldZ, " -> ", (worldX + self.mapSize*0.5),
    (worldZ + self.mapSize*0.5), z, x, row, column)
66
67  return row, column
68  end
```

MapHotspot

Description

UV coordinates in the map hotspot atlas.

setRawTextOffset

Description

Set the text offset using a raw value string in the form of "<X>px <Y>px".

Definition

setRawTextOffset()

Return Values

boolean true if loading was successful else false

setOwnerFarmId

Description

Set the farm this hotspot should be shown for. Use AccessHandler.EVERYONE for showing it to everybody (default)

Definition

setOwnerFarmId()

Return Values

bool is true if available Pallet has been assigned with a vehicle object

MapOverlayGenerator

Description

Map overlay generator.
-- Provides density map based data overlays on top of an in-game map.

new

Description

Create a MapOverlayGenerator instance.

Definition

new(table l10n, table fruitTypeManager, table fillTypeManager, table farmlandManager, table farmManager)

Arguments

table l10n	I18N reference for text localization
table fruitTypeManager	FruitTypeManager reference for fruit type resolution
table fillTypeManager	FillTypeManager reference for fruit fill type resolution
table farmlandManager	FarmlandManager reference for farm land data access
table farmManager	FarmManager reference for farm land ownership data access

delete

Description

Delete this instance and any used overlays.

Definition

delete()

setMissionFruitTypes

Description

Set the valid fruit types of the current mission.

Definition

setMissionFruitTypes()

setColorBlindMode

Description

Set the color blind mode for overlay creation.

Definition

setColorBlindMode()

buildFruitTypeMapOverlay

Description

Build the map overlay for fruit types.

Definition

buildFruitTypeMapOverlay()

buildGrowthStateMapOverlay

Description

Build the map overlay for growth states.

Definition

buildGrowthStateMapOverlay()

buildSoilStateMapOverlay

Description

Build the map overlay for soil states.

Definition

buildSoilStateMapOverlay()

buildFarmlandsMapOverlay

Description

Build the map overlay for farm lands.

Definition

buildFarmlandsMapOverlay()

generateOverlay

Description

Generate a map overlay of a given type.
This is an internal generic interfacing method and should not be called externally. Consumers should use one of the
specific public methods instead, e.g. MapOverlayGenerator:generateFruitTypeOverlay().

Definition

generateOverlay(int mapOverlayType, function finishedCallback, table overlayState)

Arguments

int	mapOverlayType	Overlay type as one of MapOverlayGenerator.OVERLAY_TYPE.[CROPS|GROWTH|SOIL|FARMLANDS]
function	finishedCallback	Function which is called with the overlay ID as its argument when processing is finished, signature: function(overlayId)
table	overlayState	Overlay state data which defines parameters (e.g. colors) of the map overlays.

Return Values

bool True	if overlay generation has started, false if generation is already in progress or an invalid overlay type has been provided

generateFruitTypeOverlay

Description

Generate a fruit type overlay.

Definition

generateFruitTypeOverlay(function finishedCallback, table fruitTypeFilter)

Arguments

function	finishedCallback	Called when generation is finished, signature: function(overlayId)

table	fruitTypeFilter	Map of fruit type indices to booleans. If the value is true, the fruit type will be displayed.

Return Values

bool True	if overlay generation has started, false if generation is already in progress or an invalid overlay type has been provided

generateGrowthStateOverlay

Description

Generate a growth state overlay.

Definition

generateGrowthStateOverlay(function finishedCallback, table fruitTypeFilter)

Arguments

function	finishedCallback	Called when generation is finished, signature: function(overlayId)
table	fruitTypeFilter	Map of growth state indices (MapOverlayGenerator.GROWTH_STATE_INDEX members) to booleans. If the value is true, the growth state will be displayed.

Return Values

bool True	if overlay generation has started, false if generation is already in progress or an invalid overlay type has been provided

generateSoilStateOverlay

Description

Generate a soil state overlay.

Definition

generateSoilStateOverlay(function finishedCallback, table fruitTypeFilter)

Arguments

function	finishedCallback	Called when generation is finished, signature: function(overlayId)
table	fruitTypeFilter	Map of soil state indices (MapOverlayGenerator.SOIL_STATE_INDEX members) to booleans. If the value is true, the soil state will be displayed.

Return Values

bool True	if overlay generation has started, false if generation is already in progress or an invalid overlay type has been provided

generateFarmlandOverlay

Description

Generate a farm land overlay.

Definition

generateFarmlandOverlay(function finishedCallback, table mapPosition)

Arguments

function	finishedCallback	Called when generation is finished, signature: function(overlayId)
table	mapPosition	Map position vector of a selection position. If the position is within a farm land, it will be highlighted.

checkOverlayFinished

Description

Check if any overlay is currently being generated and triggers a callback when it's finished.

Definition

checkOverlayFinished()

reset

Description

Reset overlay generation state.

Definition

reset()

update

Description

Update the state each frame.
Checks the overlay generation state.

Definition

update()

getDisplayCropTypes

Description

Get display information for crop types.
Override to add new crop types or change information.

Definition

getDisplayCropTypes()

Return Values

display information, {i={colors={true=[{r,g,b,a} colorblind], false=[{r,g,b,a} default], array of iconFilename=path, iconUVs={u1, v1, u2, v2, u3, v3, u4, v4}, description=text, fruitTypeIndex=index}}

Code

```
373 function MapOverlayGenerator:getDisplayCropTypes()
374 local cropTypes = {}
375 -- load crop type icon definitions in order of map configuration
376 for i, fruitType in ipairs(self.missionFruitTypes) do
377 if fruitType.shownOnMap and fruitType.fruitTypeIndex ~= FruitType.WEED t
    weed is known as a fruit type, but is handled as a soil state
378 local fillableIndex =
    self.fruitTypeManager:getFillTypeIndexByFruitTypeIndex(fruitType.fruitTy
379 local fillable = self.fillTypeManager:getFillTypeByIndex(fillableIndex)
380
381 local iconFilename = fillable.hudOverlayFilenameSmall
382 local iconUVs = Overlay.DEFAULT_UVS -- default crop type icons are separ
    files, use full texture
383 local description = fillable.title
384
385 table.insert(cropTypes, {
386 colors = {[false] = fruitType.defaultColor, [true] = fruitType.colorBlin
387 iconFilename = iconFilename,
388 iconUVs = iconUVs,
389 description = description,
```

```
390     fruitTypeIndex = fruitType.fruitTypeIndex,
391     foliageId = fruitType.foliageId
392     })
393 end
394 end
395
396 return cropTypes
397 end
```

getDisplayGrowthStates

Description

Get display information for growth states.
Growth states can be represented in multiple colors per state, so colors are defined in arrays per color blind mode.
Override to add new growth states or change information.

Definition

getDisplayGrowthStates()

Return Values

array of display information, {i={colors={true={i={r,g,b,a}}, false={i={r,g,b,a}}}, description=text}}

Code

```
404 function MapOverlayGenerator:getDisplayGrowthStates()
405 return {
406 -- indices are contiguous, so this definition is a valid array:
407 [MapOverlayGenerator.GROWTH_STATE_INDEX.CULTIVATED] = {
408 colors = {
409 [true] = {MapOverlayGenerator.FRUIT_COLOR_CULTIVATED[true]},
410 [false] = {MapOverlayGenerator.FRUIT_COLOR_CULTIVATED[false]}
411 },
412 description =
    self.l10n:getText(MapOverlayGenerator.L10N_SYMBOL.GROWTH_MAP_CULTIVATED)
413 },
414 [MapOverlayGenerator.GROWTH_STATE_INDEX.GROWING] = {
415 colors = MapOverlayGenerator.FRUIT_COLORS_GROWING,
416 description =
    self.l10n:getText(MapOverlayGenerator.L10N_SYMBOL.GROWTH_MAP_GROWING)
417 },
418 [MapOverlayGenerator.GROWTH_STATE_INDEX.HARVEST] = {
419 colors = MapOverlayGenerator.FRUIT_COLORS_HARVEST,
420 description =
    self.l10n:getText(MapOverlayGenerator.L10N_SYMBOL.GROWTH_MAP_HARVEST)
421 },
422 [MapOverlayGenerator.GROWTH_STATE_INDEX.HARVESTED] = {
423 colors = {
```

```
424     [true] = {MapOverlayGenerator.FRUIT_COLOR_CUT},
425     [false] = {MapOverlayGenerator.FRUIT_COLOR_CUT}
426     },
427     description =
        self.l10n:getText(MapOverlayGenerator.L10N_SYMBOL.GROWTH_MAP_HARVESTED)
428     },
429     [MapOverlayGenerator.GROWTH_STATE_INDEX.PLOWED] = {
430     colors = {
431     [true] = {MapOverlayGenerator.FRUIT_COLOR_PLOWED[true]},
432     [false] = {MapOverlayGenerator.FRUIT_COLOR_PLOWED[false]}
433     },
434     description =
        self.l10n:getText(MapOverlayGenerator.L10N_SYMBOL.GROWTH_MAP_PLOWED)
435     },
436     [MapOverlayGenerator.GROWTH_STATE_INDEX.TOPPING] = {
437     colors = {
438     [true] = {MapOverlayGenerator.FRUIT_COLOR_REMOVE_TOPS[true]},
439     [false] = {MapOverlayGenerator.FRUIT_COLOR_REMOVE_TOPS[false]}
440     },
441     description =
        self.l10n:getText(MapOverlayGenerator.L10N_SYMBOL.GROWTH_MAP_TOPPING)
442     },
443     [MapOverlayGenerator.GROWTH_STATE_INDEX.WITHERED] = {
444     colors = {
445     [true] = {MapOverlayGenerator.FRUIT_COLOR_WITHERED[true]},
446     [false] = {MapOverlayGenerator.FRUIT_COLOR_WITHERED[false]}
447     },
448     description =
        self.l10n:getText(MapOverlayGenerator.L10N_SYMBOL.GROWTH_MAP_WITHERED)
449     }
450     }
451     end
```

getDisplaySoilStates

Description

Get display information for soil states.
Soil states can be represented in multiple colors per state, so colors are defined in arrays per color blind mode.
Override to add new soil states or change information.

Definition

getDisplaySoilStates()

Return Values

array of display information, {i={colors={true={i={r,g,b,a}}, false={i={r,g,b,a}}}, description=text}}

Code

```
458 function MapOverlayGenerator:getDisplaySoilStates()
459 local weedType = self.fruitTypeManager:getWeedFruitType()
460 local fillableIndex =
    self.fruitTypeManager:getFillTypeIndexByFruitTypeIndex(FruitType.WEED)
461 local weedFillable =
    self.fillTypeManager:getFillTypeByIndex(fillableIndex)
462
463 return {
464 [MapOverlayGenerator.SOIL_STATE_INDEX.WEEDS] = {
465 colors = {
466 [true] = {weedType.colorBlindMapColor},
467 [false] = {weedType.defaultMapColor}
468 },
469 description = weedFillable.title
470 },
471 [MapOverlayGenerator.SOIL_STATE_INDEX.FERTILIZED] = {
472 colors = MapOverlayGenerator.FRUIT_COLORS_FERTILIZED,
473 description =
    self.l10n:getText(MapOverlayGenerator.L10N_SYMBOL.SOIL_MAP_FERTILIZED)
474 },
475 [MapOverlayGenerator.SOIL_STATE_INDEX.NEEDS_PLOWING] = {
476 colors = {
477 [true] = {MapOverlayGenerator.FRUIT_COLOR_NEEDS_PLOWING[true]},
478 [false] = {MapOverlayGenerator.FRUIT_COLOR_NEEDS_PLOWING[false]}
479 },
480 description =
    self.l10n:getText(MapOverlayGenerator.L10N_SYMBOL.SOIL_MAP_NEED_PLOWING)
481 },
482 [MapOverlayGenerator.SOIL_STATE_INDEX.NEEDS_LIME] = {
483 colors = {
484 [true] = {MapOverlayGenerator.FRUIT_COLOR_NEEDS_LIME[true]},
485 [false] = {MapOverlayGenerator.FRUIT_COLOR_NEEDS_LIME[false]}
486 },
487 description =
    self.l10n:getText(MapOverlayGenerator.L10N_SYMBOL.SOIL_MAP_NEED_LIME)
488 }
489 }
490 end
```

MapSelectionScreen
Description

Map Selection Screen.
-- Used when starting a new game.
-- @field mapSelector Map selection option at top of view

onCreateMapImage

Description

Creation event for map preview BitmapElement instances.
This gets called during onOpen when map data is iterated and added, so we can set the image state according to the
currently processed map.

Definition

onCreateMapImage(element BitmapElement)

Arguments

element BitmapElement instance which shows a map preview image

MessageCenter

Description

Message center message types.
Use these identifiers to subscribe to and publish messages.

subscribe

Description

Subscribe the target to given message type.

Definition

subscribe(integer messageType, function callback, table callbackTarget, any argument)

Arguments

integer	messageType	Type of message
function	callback	Callback function
table	callbackTarget	Optional target
any	argument	Optional argument, will be last in callback argument list

Return Values

table implement implement

unsubscribe

Description

Unsubscribe the observer from message
This is relatively slow, do not use in :update or :draw

Definition

unsubscribe(integer messageType, table callbackTarget)

Arguments

integer	messageType	Type of message
table	callbackTarget	Observer object

Return Values

boolean success success

unsubscribeAll

Description

Unsubscribe the observer from all messages

Definition

unsubscribeAll(table callbackTarget)

Arguments

table callbackTarget Observer object

Return Values

boolean supportsHardAttach attacher joint supports hard attach

publish

Description

Publish a message with given type and possible arguments

Definition

publish(integer messageType, table arguments)

Arguments

integer messageType Type of the message, used with the subscriptions

table arguments Optional arguments passed to function

Return Values

boolean success success

publishDelayed

Description

Publish a message with given type and possible arguments. This message is handled in the next frame.
Useful for within networking code

Definition

publishDelayed(integer messageType, table arguments)

Arguments

integer messageType Type of the message, used with the subscriptions

table arguments Optional arguments passed to function

Return Values

float totalMass total mass

MissionCollaborators

Description

Mission class collaborator collection for initialization.
-- Serves as an explicitly defined value-object for Mission class constructors to pass in required collaborator
references without needing to change the constructor signature for each new collaborator class.

new

Description

Create a new MissionCollaborators instance.
The constructor declares fields for reference which are used by the Mission class constructors. Stick to only
assigning to these fields when using this class or add a suitable declaration below when more references are needed.

Definition

new()

Return Values

float total amount of consumed pto torque in kNm

MissionManager
Description

Dismiss a (finished or cancelled) mission. Deletes it completely. Calls dismiss on the mission to handle money exchange to farm.

new
Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
31  function MissionManager:new(customMt)
32  local self = AbstractManager:new(customMt or MissionManager_mt)
33
34  self.missionTypes = {}
35  self.missionTypeIdToType = {}
36
37  self.defaultMissionMapWidth = 512
38  self.defaultMissionMapHeight = 512
39  self.missionMapNumChannels = 4
40
41  -- Only on new game start. Reset is done when quitting a game.
42  -- This value is changed very early in the loading process
43  self.numTransportTriggers = 0
44  self.transportTriggers = {}
45
46  return self
47  end
```

loadMapData
Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
74  function MissionManager:loadMapData(xmlFile)
75  MissionManager:superClass().loadMapData(self)
76
77  self:createMissionMap()
78
```

```
79  if g_currentMission:getIsServer() then
80  g_currentMission:addUpdateable(self)
81
82  self.missionNextGenerationTime = g_currentMission.time +
    MissionManager.MISSION_GENERATION_INTERVAL
83
84  self.fieldDataDmod =
    DensityMapModifier:new(g_currentMission.terrainDetailId,
    g_currentMission.sprayFirstChannel,
    g_currentMission.sprayNumChannels)
85  self.fieldDataFilter =
    DensityMapFilter:new(g_currentMission.terrainDetailId,
    g_currentMission.terrainDetailTypeFirstChannel,
    g_currentMission.terrainDetailTypeNumChannels)
86  self.fieldDataFilter:setValueCompareParams("greater", 0)
87  end
88
89  local p = getXMLString(xmlFile, "map.transportMissions#filename")
90  if p ~= nil then
91  local path = Utils.getFilename(p, g_currentMission.baseDirectory)
92  if path ~= nil and path ~= "" then
93  self:loadTransportMissions(path)
94  end
95  end
96
97  local p = getXMLString(xmlFile, "map.missionVehicles#filename")
98  if p ~= nil then
99  local path = Utils.getFilename(p, g_currentMission.baseDirectory)
100 if path ~= nil then
101 self:loadMissionVehicles(path)
102 end
103 end
104
105 if g_currentMission:getIsServer() then
106 -- Generate a weighted list of transport missions
107 for _, missionType in ipairs(self.missionTypes) do
108 if missionType.category == MissionManager.CATEGORY_TRANSPORT then
109 for i = 1, missionType.priority do
110 table.insert(self.possibleTransportMissionsWeighted, missionType)
111 end
112 end
113 end
```

```
114 end
115
116 g_messageCenter:subscribe(MessageType.MISSION_DELETED,
    self.onMissionDeleted, self)
117
118 if g_addTestCommands then
119 addConsoleCommand("gsGenerateFieldMission", "Force generating a
    new mission for given field", "consoleGenerateFieldMission",
    self)
120 addConsoleCommand("gsMissionLoadAllVehicles", "Loading and
    unloading all field mission vehicles",
    "consoleLoadAllFieldMissionVehicles", self)
121
122 addConsoleCommand("gsMissionHarvestField", "Harvest a field and
    print the liters", "consoleHarvestField", self)
123 end
124 end
```

loadTransportMissions

Description

Load map-configured transport missions

Definition

loadTransportMissions()

loadMissionVehicles

Description

Load vehicles for use with field missions

Definition

loadMissionVehicles()

unloadMapData

Description

Unload data on mission delete

Definition

unloadMapData()

Code

```
349 function MissionManager:unloadMapData()
350 g_messageCenter:unsubscribeAll()
351 g_currentMission:removeUpdateable(self)
352
353 self.numTransportTriggers = 0
354 self.transportTriggers = {}
355
356 self.fieldDataDmod = nil
357 self.fieldDataFilter = nil
```

```
358
359  self:destroyMissionMap()
360
361  if g_addTestCommands then
362  removeConsoleCommand("gsGenerateFieldMission")
363  removeConsoleCommand("gsMissionLoadAllVehicles")
364  removeConsoleCommand("gsMissionHarvestField")
365  end
366
367  MissionManager:superClass().unloadMapData(self)
368  end
```

saveToXMLFile

Description

Write field mission data to savegame file

Definition

saveToXMLFile(string xmlFilename)

Arguments

string xmlFilename file path

Return Values

boolean true if loading was successful else false

Code

```
374  function MissionManager:saveToXMLFile(xmlFilename)
375  local xmlFile = createXMLFile("missionXML", xmlFilename,
     "missions")
376
377  if xmlFile ~= nil then
378  for k, mission in ipairs(self.missions) do
379  local missionKey = string.format("missions.mission(%d)", k - 1)
380
381  setXMLString(xmlFile, missionKey.."#type", mission.type.name)
382  if mission.activeMissionId ~= nil then
383  setXMLInt(xmlFile, missionKey.."#activeId",
     mission.activeMissionId)
384  end
385
386  mission:saveToXMLFile(xmlFile, missionKey)
387  end
388
389  saveXMLFile(xmlFile)
390  delete(xmlFile)
391  end
```

```
392
393 return false
394 end
```

loadFromXMLFile

Description

Load fieldjob data from xml savegame file

Definition

loadFromXMLFile(string filename)

Arguments

string filename xml filename

Code

```
399 function MissionManager:loadFromXMLFile(xmlFilename)
400 if xmlFilename == nil then
401 return false
402 end
403
404 local xmlFile = loadXMLFile("missionsXML", xmlFilename)
405 if not xmlFile then
406 return false
407 end
408
409 -- Active missions
410 local i = 0
411 while true do
412 local key = string.format("missions.mission(%d)", i)
413 if not hasXMLProperty(xmlFile, key) then
414 break
415 end
416
417 local missionTypeName = getXMLString(xmlFile, key.."#type")
418 local missionType = self:getMissionType(missionTypeName)
419
420 if missionType ~= nil then
421 local mission = missionType.class:new(true, g_client ~= nil)
422 mission.type = missionType
423 mission.activeMissionId = getXMLInt(xmlFile, key .. "#activeId")
    -- can be nil
424 self:assignGenerationTime(mission)
425
426 if not mission:loadFromXMLFile(xmlFile, key) then
427 mission:delete()
```

```
428 else
429 if mission.field ~= nil then
430 self.fieldToMission[mission.field.fieldId] = mission
431 end
432
433 if mission.type.category == MissionManager.CATEGORY_TRANSPORT
    then
434 self.numTransportMissions = self.numTransportMissions + 1
435 end
436
437 mission:register()
438 table.insert(self.missions, mission)
439 end
440 else
441 print("Warning: Mission type '" .. tostring(missionType) .. "'
    not found!")
442 end
443
444 i = i + 1
445 end
446
447 -- If there are any active missions, find any associated vehicles
448 if table.getn(self.missions) > 0 then
449 for _, vehicle in pairs(g_currentMission.vehicles) do
450 if vehicle.activeMissionId ~= nil then
451 local mission =
    self:getMissionForActiveMissionId(vehicle.activeMissionId)
452 if mission ~= nil and mission.vehicles ~= nil then
453 table.insert(mission.vehicles, vehicle)
454 end
455 end
456 end
457 end
458
459 delete(xmlFile)
460
461 return true
462 end
```

delete

Description

Deletes field mission manager

Definition

delete()

Code

```
466  function MissionManager:delete()
467  end
```

update

Description

Updates field mission ownage data from xml savegame file

Definition

update(string filename)

Arguments

string filename xml filename

Code

```
472  function MissionManager:update(dt)
473  if g_currentMission:getIsServer() then
474  self.generationTimer = self.generationTimer -
     g_currentMission.missionInfo.timeScale * dt
475
476  self:updateMissions(dt)
477
478  if table.getn(self.missions) < MissionManager.MAX_MISSIONS and
     self.generationTimer < g_time then
479  self:generateMissions(dt)
480
481  -- Limit generation
482  self.generationTimer = MissionManager.MISSION_GENERATION_INTERVAL
483  end
484  end
485  end
```

getTransportMissionConfig

Description

Get mission configuration given the name

Definition

getTransportMissionConfig()

getTransportMissionConfigById

Description

Get a mission config given an ID. Used by TransportMission:readStream

Definition

getTransportMissionConfigById()

hasFarmActiveMission

Description

Get whether given farm has an active mission

Definition

hasFarmActiveMission()

startMission

Description

Start given mission for a farm.

Definition

startMission()

cancelMission

Description

Cancel mission: sets it to finished without success

Definition

cancelMission()

deleteMission

Description

Delete a mission

Definition

deleteMission()

dismissMission

Description

On a client it sends an event instead

Definition

dismissMission()

getActiveMissions

Description

Get a list of active missions

Definition

getActiveMissions()

getIsAnyMissionActive

Description

Get whether any mission is currently running

Definition

getIsAnyMissionActive()

canMissionStillRun

Description

Test whether the given mission is still able to run

Definition

canMissionStillRun()

addTransportMissionTrigger

Description

Add a new transport trigger. Requires triggerId and index properties.

Definition

addTransportMissionTrigger()

removeTransportMissionTrigger

Description

Remove a transport trigger.

Definition

removeTransportMissionTrigger()

generateNewFieldMission

Description

Generate a mission for given field. Returns nil if field already has an active mission

Definition

generateNewFieldMission()

assignGenerationTime

Description

Generation time is used for reliably sorting

Definition

assignGenerationTime()

getRandomVehicleGroup

Description

Get a randomly chosen group of vehicles fitting for given mission type and field size

Definition

getRandomVehicleGroup(missionType type, fieldSize size)

Arguments

missionType type of mission (string)
fieldSize size of field: 'SMALL', 'MEDIUM', 'LARGE'

Return Values

List of vehicles. Each element is a table with filanema and configuration properties.
Reward

getVehicleGroupFromIdentifier

Description

Used on the client. Make sure it never crashes on nil (patches)

Definition

getVehicleGroupFromIdentifier()

getFreeActiveMissionId

Description

Get a free activeMissionId, used for active missions only.

Definition

getFreeActiveMissionId()

validateMissionOnField

Description

Validate missions on given field, after something happened to the field (event)

Definition

validateMissionOnField()

getMissionMapValue

Description

Get a value at given world coordinates

Definition

getMissionMapValue()

getMissionForActiveMissionId

Description

Get the mission associated with the activeMissionId

Definition

getMissionForActiveMissionId()

MixerWagonHUDExtension

Description

Custom HUD drawing extension for MixerWagon.
-- Displays the fill levels of the mixer wagon.
--@category GUI

new

Description

Create a new instance of MixerWagonHUDExtension.

Definition

new(table vehicle, float uiScale, table uiTextColor, float uiTextSize)

Arguments

table	vehicle	Vehicle which has the specialization required by a sub-class
float	uiScale	Current UI scale
table	uiTextColor	HUD text drawing color as an RGBA array
float	uiTextSize	HUD text size

Return Values

table instance instance of object

canDraw

Description

Determine if the HUD extension should be drawn.

Definition

canDraw()

Return Values

boolean true if loading was successful else false

getDisplayHeight

Description

Get this HUD extension's display height.

Definition

getDisplayHeight()

Return Values

table instance instance of object

float	Display	height in screen space

draw

Description

Draw mixing ratio information for a mixing wagon when it is active.

Definition

draw(float leftPosX, float rightPosX, float posY)

Arguments

float leftPosX	Left input help panel column start position
float rightPosX	Right input help panel column start position
float posY	Current input help panel drawing vertical offset

Return Values

boolean true	if loading was successful else false
float	Modified input help panel drawing vertical offset

ModHubCategoriesFrame

Description

ModHub categories frame
-- Displays categories.
--@category GUI

new

Description

Create a new ModHubCategoriesFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional]	Meta table of subclass

initialize

Description

Initialize with categories to be displayed.
Categories must be provided as an array of tables like {id=<id>, iconFilename=<path>, label=<text>}. This will add
a category element per entry to the display list.

Definition

initialize(table categories, function categoryClickedCallback, table headerIconUVs, string headerText)

Arguments

table	categories	Category definitions array
function	categoryClickedCallback	Notification callback for category activation(click/enter), signature: function(selectedId, baseCategoryIconUVs, baseCategoryLabel, clickedCategoryLabel)
table	headerIconUVs	UV coordinates of the header icon to display
string	headerText	Header text to display

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

float nearCoCRadius	Near circle of confusion radius (nearCoCRadius = 0 means no near blur (pinhole camera))
float nearBlurEnd	Distance from the camera center where near blur ends
float farCoCRadius	Far circle of confusion radius (farCoCRadius = 0 means no far blur (pinhole camera))
float farBlurStart	Distance from the camera center where far blur starts
float farBlurEnd	Distance from the camera center where far blur ends

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

updateScrollButtons

Description

Update scroll button visibility based on the currently visible list items.

Definition

updateScrollButtons()

onClickCategory

Description

Handle a click / button activation on a category.

Definition

onClickCategory()

onDoubleClickCategory

Description

Handle a double click on a category.

Definition

onDoubleClickCategory()

onCategorySelected

Description

Handle navigation selection of a category element.

Definition

onCategorySelected()

Return Values

boolean isAllowed state change is allowed

onClickLeft

Description

Handle click on left navigation button.

Definition

onClickLeft()

Return Values

table instance instance of object

onClickRight

Description

Handle click on right navigation button.

Definition

onClickRight()

Return Values

boolean true if loading was successful else false

onScroll

Description

Handle a list scrolling event.

Definition

onScroll()

Return Values

boolean success success

ModHubDetailsFrame

Description

Shop items frame for the in-game menu shop.
-- Displays purchasable items of a common category in a horizontal list layout.
--@category GUI

new

Description

Create a new ModHubDetailsFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional] Meta table of subclass

Return Values

integer fillTypeIndex the fillType index

setCategory

Description

Set the category to display.

Definition

setCategory()

Return Values

table fillType the fillType object

setModInfo

Description

Set an ordered array of ShopDisplayItem instances to display in this frame.

Definition

setModInfo()

Return Values

table fillTypes list of fillTypes

getMainElementSize
Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

table fillTypeCategory fillType category object

getMainElementPosition
Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

table success true if added else false

ModHubItemsFrame
Description

Shop items frame for the in-game menu shop.
-- Displays purchasable items of a common category in a horizontal list layout.
--@category GUI

new
Description

Create a new ModHubItemsFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional] Meta table of subclass

Return Values

table fillTypes list of fillTypes

setItemClickCallback
Description

Set the callback to use when an item is activated (for buying or selling).

Definition

setItemClickCallback()

Return Values

table fillTypes fill types

setItemSelectCallback
Description

Set the callback to use when an item is selected in the view.

Definition

setItemSelectCallback()

Return Values

integer converterIndex index of converterIndex

setCategory

Description

Set the category to display.

Definition

setCategory()

Return Values

table converterData converter data

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

table sample sample

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

table instance instance of object

updateScrollButtons

Description

Update scroll button visibility based on the currently visible list items.

Definition

updateScrollButtons()

Return Values

boolean true if loading was successful else false

onClickItem

Description

Handle a click / button activation on an item.

Definition

onClickItem()

Return Values

boolean success success

onDoubleClickItem

Description

Handle a double-click on an item.

Definition

onDoubleClickItem()

Return Values

table fruitType fruitType type object

onClickLeft

Description

Handle click on left navigation button.

Definition

onClickLeft()

Return Values

table fruit the fruit object

onClickRight

Description

Handle click on right navigation button.

Definition

onClickRight()

Return Values

table fruit the fruit object

onItemSelected

Description

Handle selection of an item.

Definition

onItemSelected()

Return Values

table fruitTypes a list of fruitTypes

ModHubScreen

Description

Mode

new

Description

Create a new instance of ModHubScreen.

Definition

new(table target, table custom_mt, table messageCenter, table l10n, table inputManager, table modHubController, bool isConsoleVersion)

Arguments

table target	Callback target
table custom_mt	Sub-class meta table
table messageCenter	MessageCenter reference
table l10n	I18N reference for text localization
table inputManager	InputBinding reference
table modHubController	modHubController reference
bool isConsoleVersion	If true, the game is running on a console

Return Values

table fruitTypeCategory fruitType category object

reset

Description

Reset menu state (and all pages).

Definition

reset()

Return Values

table success true if added else false

clickBackCallback

Description

Button function for backing out of the menu.

Definition

clickBackCallback()

Return Values

table fruitTypes list of fruitTypes

clearMenuButtonActions

Description

Clear menu button actions, events and callbacks.

Definition

clearMenuButtonActions()

Return Values

table fruitTypes fruit types

assignMenuButtonInfo

Description

Assign menu button information to the in-game menu buttons.

Definition

assignMenuButtonInfo()

Return Values

table fillTypes fill types

onMenuActionClick

Description

Handle a menu action click by calling one of the menu button callbacks.

Definition

onMenuActionClick()

Return Values

table fillTypes fill types

bool True if no callback was present and no action was taken, false otherwise

onClickOk

Description

Handle menu confirmation input event.

Definition

onClickOk()

Return Values

float literPerSqm liter per sqm

onClickBack

Description

Handle menu back input event.

Definition

onClickBack()

Return Values

integer converterIndex index of converterIndex

onClickCancel

Description

Handle menu cancel input event.
Bound to quite the game to the main menu.

Definition

onClickCancel()

Return Values

table converterData converter data

onClickActivate

Description

Handle menu active input event.
Bound to save the game.

Definition

onClickActivate()

Return Values

table instance instance of object

updateButtonsPanel

Description

Update the buttons panel when a given page is visible.

Definition

updateButtonsPanel()

Return Values

boolean true if loading was successful else false

makeSelfCallback

Description

Make a callback which encloses the self reference and handles arbitrary arguments afterwards.

Definition

makeSelfCallback()

Return Values

table gameplayHintGroup a random gameplay hint group

MultiTextOptionElement

Description

Multiple choice text element.
-- This element requires a specific configuration setup to be properly used: In the configuration, it must contain the
following child elements in this order: 1. ButtonElement, 2. ButtonElement, 3. TextElement, 4. TextElement. The first
three elements are mandatory, as they are the buttons to change this element's value and the label which displays the

value. The fourth (text) element is optional and used as a header label for this element if defined.
--@category GUI
--@xmlConfig GuiElement#wrap bool [optional] If false, values will not cycle when the end or start of the value list is reached.

disableButtonSounds

Description

Disable automatic playing of sound samples in child buttons.

Definition

disableButtonSounds()

inputLeft

Description

Trigger a "left" input.

Definition

inputLeft(isShoulderButton If)

Arguments

isShoulderButton If true, assume a shoulder button input (PC or console)

inputRight

Description

Trigger a "right" input.

Definition

inputRight(isShoulderButton If)

Arguments

isShoulderButton If true, assume a shoulder button input (PC or console)

MultiValueTween

Description

Tween class which handles multiple values at the same time.
-- Start and end values must be passed in as arrays. The setter function must be able to handle as many arguments as
there were entries in the start and end values arrays: setter called as function(unpack(values)).
--@category GUI

new

Description

Create a new Tween.

Definition

new(table subClass, function setterFunction, table startValues, table endValues, float duration)

Arguments

table	subClass	Subclass metatable for inheritance
function	setterFunction	Values setter function. Signature: callback(v1, ..., vn) or callback(target, v1, ..., vn).
table	startValues	Original values
table	endValues	Target values
float	duration	Duration of tween in milliseconds

setTarget
Description

Set a callback target for this tween.
If a target has been set, the setter function must support receiving the target as its first argument.

Definition

setTarget()

tweenValue
Description

Get the current tween value.

Definition

tweenValue()

applyValue
Description

Apply a value via the setter function.

Definition

applyValue()

Overlay
Description

Image display overlay.
-- This class is used to display textures or usually parts thereof as rectangular panels in the UI or on the HUD.
Example usages include button icons, showing images in the main menu or drawing the in-game map.
--@category GUI

new
Description

Create a new Overlay.

Definition

new(overlayFilename File, x Screen, y Screen, width Display, height Display)

Arguments

overlayFilename	File	path of the source texture
x	Screen	position x
y	Screen	position y
width	Display	width
height	Display	height

delete
Description

Delete this overlay.
Releases the texture file handle.

Definition

delete()

setColor
Description

Set this overlay's color.
The color is multiplied with the texture color. For no modification of the texture color, use full opaque white,
i.e. {1, 1, 1, 1}.

Definition

setColor(r Red, g Green, b Blue, a Alpha)

Arguments

r Red channel
g Green channel
b Blue channel
a Alpha channel

setUVs

Description

Set this overlay's UVs which define the area to be displayed within the target texture.

Definition

setUVs(uvs UV)

Arguments

uvs UV coordinates in the form of {u1, v1, u2, v2, u3, v3, u4, v4}

setPosition

Description

Set this overlay's position.

Definition

setPosition()

getPosition

Description

Get this overlay's position.

Definition

getPosition()

Return Values

float X position in screen space
float Y position in screen space

setDimension

Description

Set this overlay's width and height.
Either value can be omitted (== nil) for no change.

Definition

setDimension()

resetDimensions

Description

Reset width, height and scale to initial values set in the constructor.

Definition

resetDimensions()

setInvertX

Description

Set horizontal flipping state.

Definition

setInvertX(invertX If)

Arguments

invertX If true, will set the overlay to display its image flipped horizontally

setRotation

Description

Set this overlay's rotation.

Definition

setRotation(rotation Rotation, centerX Rotation, centerY Rotation)

Arguments

rotation Rotation in radians

centerX Rotation pivot X position offset from overlay position in screen space

centerY Rotation pivot Y position offset from overlay position in screen space

setScale

Description

Set this overlay's scale.
Multiplies the scale values with the initial dimensions and sets those as the current dimensions.

Definition

setScale(float scaleWidth, float scaleHeight)

Arguments

float scaleWidth Width scale factor

float scaleHeight Height scale factor

getScale

Description

Get this overlay's scale values.

Definition

getScale()

Return Values

float Width scale factor

float Height scale factor

render

Description

Render this overlay.

Definition

render()

setAlignment

Description

Set this overlay's alignment.

Definition

setAlignment(vertical Vertical, horizontal Horizontal)

Arguments

vertical Vertical alignment value, one of Overlay.ALIGN_VERTICAL_[...]

horizontal Horizontal alignment value, one of Overlay.ALIGN_HORIZONTAL_[...]

setIsVisible

Description

Set this overlay's visibility.

Definition

setIsVisible()

setImage

Description

Set a different image for this overlay.
The previously targeted image's handle will be released.

Definition

setImage(overlayFilename File)

Arguments

overlayFilename File path to new target image

PagingElement

Description

Paging control element.
-- Organizes grouped elements into pages to be displayed one at a time. To set it up, one defines several
same-sized container elements (e.g. bare GuiElement) as children of the PagingElement to hold the pages' contents.
The pages should be given #name properties which are resolved to a localization text with a prepended "ui_" prefix.
On loading, any named child element of this PagingElement will be added as a page.
--@category GUI
--@xmlConfig GuiElement#onPageChange callback [optional]
onPageChangeCallback(pageIndex, pageMappingIndex, element) Called when the page changes. Receives the current index in all pages, the current index in all enabled pages and this element.

setPage

Description

Set the current page index.
Indices are based on mapped pages, which are all available and visible pages added to this element.

Definition

setPage(pageMappingIndex Index)

Arguments

pageMappingIndex Index of page in page mappings

Return Values

bool True if the page changed

getNextID

Description

Get a new page ID based on an internal counter.

Definition

getNextID()

addPage

Description

Add a new page.

Definition

addPage(id Page, element Page, title Page, index [optional])

Arguments

id	Page	ID
element	Page	container GuiElement
title	Page	title for displaying
index	[optional]	Insertion index for page. If undefined or outside of valid range, will add the page at the end.

getVisiblePagesCount

Description

Get the number of currently visible pages.

Definition

getVisiblePagesCount()

getPageIdByElement

Description

Get a page ID by the page container element reference.

Definition

getPageIdByElement(element Page)

Arguments

element Page container GuiElement instance

Return Values

Page

getPageElementByIndex

Description

Get a page container element by page index.

Definition

getPageElementByIndex(pageIndex Index)

Arguments

pageIndex Index of page in all (incl. disabled and invisible) pages

Return Values

Page container GuiElement instance

getPageIndexByElement

Description

Get the page index in this element's page array for a given element reference.

Definition

getPageIndexByElement(table element)

Arguments

table element GuiElement instance

Return Values

int Page index

getPageMappingIndexByElement

Description

Get the page index in this elements page mapping array (visible pages) for a given element reference.

Definition

getPageMappingIndexByElement(table element)

Arguments

table element GuiElement instance

Return Values

int Page mapping index

removePageByElement

Description

Remove a page, identified by its page container element, from this element.
The method only removes the page from the display collection. Callers must take care of the page element's clean-up,
e.g. removing it from the element hierarchy.

Definition

removePageByElement(pageElement Page)

Arguments

pageElement Page container element

getCurrentPageId

Description

Get the page ID of the currently displayed page.

Definition

getCurrentPageId()

getPageMappingIndex

Description

Get the index of a page in the page mappings (only visible pages) by page ID.

Definition

getPageMappingIndex()

getIsPageDisabled

Description

Determine if a page, identified by page ID, is disabled.

Definition

getIsPageDisabled()

getPageById

Description

Get a page by ID.

Definition

getPageById()

setPageIdDisabled

Description

Set a page's disabled state.

Definition

setPageIdDisabled(pageId Page, disabled True)

Arguments

pageId	Page ID
disabled True	for disabled, false for enabled

getPageTitles

Description

Get page titles of all pages which were visible at the latest call to updatePageMapping()

Definition

getPageTitles()

Placeable

Description

Base Class for placeables
-- Note about terrain modification on placement:
If terrain modification is enabled using the placeable.leveling#requireLeveling attribute, the configuration needs
at least one leveling area to be defined (ramps are optional). These areas represent parallelograms which are passed
to terrain modification functions. They are defined by start, width and height nodes which in this case should be
set up to form rectangular shapes. For the best results, create a new transform group for each area at the starting
positions and add separate nodes for the three defining points in the placeable object I3D file.
--@category Placeables
--@xmlConfig placeable.filename string File name of associated I3D file

onCreateGlowMaterial

Description

On create glow material

Definition

onCreateGlowMaterial(empty empty, integer id)

Arguments

empty	empty empty
integer id	id of node

Code

```
90  function Placeable.onCreateGlowMaterial(_, id)
91  if getHasShaderParameter(id, "colorScale") then
92  Placeable.GLOW_MATERIAL = getMaterial(id, 0)
93  end
94  end
```

new

Description

Creating placeable

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean	isServer	is server
boolean	isClient	is client
table	customMt	custom metatable

Return Values

table	instance Instance of object

Code

```
106  function Placeable:new(isServer, isClient, customMt)
107  local self = Object:new(isServer, isClient, customMt or
     Placeable_mt)
108  self.nodeId = 0
109
110  self.useRandomYRotation = false
111  self.useManualYRotation = false
112  self.placementSizeX = 1
113  self.placementSizeZ = 1
114  self.placementTestSizeX = 1
115  self.placementTestSizeZ = 1
116  self.requireLeveling = false
117  self.maxSmoothDistance = 3
118  self.maxSlope = MathUtil.degToRad(45)
119  self.maxEdgeAngle = MathUtil.degToRad(45)
120  self.smoothingGroundType = nil
121
122  self.triggerMarkers = {}
123  self.clearAreas = {}
124  self.levelAreas = {}
125  self.rampAreas = {}
126  self.foliageAreas = {}
127  self.samples = {}
128  self.pickObjects = {}
129  self.animatedObjects = {}
130  self.triggerMarkers = {}
131  self.mapHotspots = {}
132  self.isolated = false
133  self.isDeleted = false
134  self.useMultiRootNode = false
135  self.price = 0
```

```
136 self.age = 0
137 self.isInPreviewMode = nil
138 self.placementPositionSnapSize = 0
139 self.placementRotationSnapAngle = 0
140
141 -- defines that a placeable is placed on a map directly, and with
    a unique ID
142 self.mapBoundId = nil
143
144 registerObjectClassName(self, "Placeable")
145 return self
146 end
```

delete

Description

Deleting placeable

Definition

delete()

Code

```
150 function Placeable:delete()
151 self.isDeleted = true
152 if self.i3dFilename ~= nil then
153 g_i3DManager:releaseSharedI3DFile(self.i3dFilename, nil, true)
154 end
155
156 for _, node in ipairs(self.triggerMarkers) do
157 g_currentMission:removeTriggerMarker(node)
158 end
159
160 if #self.rampAreas > 0 then
161 for _, area in pairs(self.rampAreas) do
162 if area.previewShape then
163 g_i3DManager:releaseSharedI3DFile(Placeable.RAMP_PREVIEW_PATH,
    nil, true)
164 end
165 end
166 end
167
168 for _, animatedObject in ipairs(self.animatedObjects) do
169 animatedObject:delete()
170 end
171
```

```
172 for _, hotspot in ipairs(self.mapHotspots) do
173 g_currentMission.hud:removeMapHotspot(hotspot)
174 hotspot:delete()
175 end
176
177 if g_currentMission ~= nil and g_currentMission.environment ~=
    nil then
178 g_currentMission.environment:removeDayChangeListener(self)
179 g_currentMission.environment:removeWeatherChangeListener(self)
180 g_currentMission.environment:removeHourChangeListener(self)
181 end
182
183 unregisterObjectClassName(self)
184 g_currentMission:removeItemToSave(self)
185 g_currentMission:removePlaceable(self)
186 for _, node in pairs(self.pickObjects) do
187 g_currentMission:removeNodeObject(node)
188 end
189
190 if self.nodeId ~= 0 and entityExists(self.nodeId) then
191 self:setCollisionMask(self.nodeId, 0)
192 setVisibility(self.nodeId, false)
193 else
194 self.nodeId = 0
195 end
196
197 if self.isClient then
198 g_soundManager:deleteSamples(self.samples)
199 end
200
201 g_currentMission:removeOwnedItem(self)
202
203 g_currentMission:addPlaceableToDelete(self, 300)
204 Placeable:superClass().delete(self)
205 end
```

deleteFinal

Description

Deleting placeable final

Definition

deleteFinal()

Code

```
209  function Placeable:deleteFinal()
210  if self.nodeId ~= 0 then
211  delete(self.nodeId)
212  self.nodeId = 0
213  end
214  end
```

setCollisionMask

Description

Set collision mask of node and its children

Definition

setCollisionMask(integer nodeId, integer mask)

Arguments

integer nodeId id of node

integer mask collision mask

Code

```
220  function Placeable:setCollisionMask(nodeId, mask)
221  setCollisionMask(nodeId, mask)
222  local numChildren = getNumOfChildren(nodeId)
223  for i=0,numChildren-1 do
224  local childId = getChildAt(nodeId, i)
225  self:setCollisionMask(childId, mask)
226  end
227  end
```

getIsPlayerInRange

Description

Returns true if player is in range

Definition

getIsPlayerInRange(float distance, table player)

Arguments

float distance distance

table player player

Return Values

boolean isInRange is in range

Code

```
234  function Placeable:getIsPlayerInRange(distance, player)
235  if self.nodeId ~= 0 then
236  distance = Utils.getNoNil(distance, 10)
237  if player == nil then
238  for _, player in pairs(g_currentMission.players) do
239  if self:isInActionDistance(player, self.nodeId, distance) then
240  return true, player
241  end
```

```
242 end
243 else
244 return self:isInActionDistance(player, self.nodeId, distance), player
245 end
246 end
247 return false, nil
248 end
```

isInActionDistance

Description

Returns true if player is in range

Definition

isInActionDistance(table player, integer refNode, float distance)

Arguments

table player player
integer refNode id of reference node
float distance distance

Return Values

boolean isInRange is in range

Code

```
256 function Placeable:isInActionDistance(player, refNode, distance)
257 local x,_,z = getWorldTranslation(refNode)
258 local px,_,pz = getWorldTranslation(player.rootNode)
259 local dx,dz = px-x, pz-z
260 if dx*dx + dz*dz < distance*distance then
261 return true
262 end
263
264 return false
265 end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID
table connection connection

Code

```
271 function Placeable:readStream(streamId, connection)
272 Placeable:superClass().readStream(self, streamId, connection)
273 if connection:getIsServer() then
```

```
274 local configFileName = NetworkUtil.convertFromNetworkFilename(streamReadString(streamId))
275 local x=streamReadFloat32(streamId)
276 local y=streamReadFloat32(streamId)
277 local z=streamReadFloat32(streamId)
278 local rx=NetworkUtil.readCompressedAngle(streamId)
279 local ry=NetworkUtil.readCompressedAngle(streamId)
280 local rz=NetworkUtil.readCompressedAngle(streamId)
281 local isNew = self.configFileName == nil
282 if isNew then
283 self:load(configFileName, x,y,z, rx,ry,rz, false, false)
284 end
285 self.age = streamReadUInt16(streamId)
286 self.price = streamReadInt32(streamId)
287
288 if isNew then
289 self:finalizePlacement()
290 end
291
292 for _, animatedObject in ipairs(self.animatedObjects) do
293 local animatedObjectId = NetworkUtil.readNodeObjectId(streamId)
294 animatedObject:readStream(streamId, connection)
295 g_client:finishRegisterObject(animatedObject, animatedObjectId)
296 end
297 end
298 end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId	stream ID
table	connection connection

Code

```
304 function Placeable:writeStream(streamId, connection)
305 Placeable:superClass().writeStream(self, streamId, connection)
306 if not connection:getIsServer() then
307 streamWriteString(streamId, NetworkUtil.convertToNetworkFilename(self.configFileName))
308 local x,y,z = getTranslation(self.nodeId)
```

```
309 local x_rot,y_rot,z_rot = getRotation(self.nodeId)
310 streamWriteFloat32(streamId, x)
311 streamWriteFloat32(streamId, y)
312 streamWriteFloat32(streamId, z)
313 NetworkUtil.writeCompressedAngle(streamId, x_rot)
314 NetworkUtil.writeCompressedAngle(streamId, y_rot)
315 NetworkUtil.writeCompressedAngle(streamId, z_rot)
316 streamWriteUInt16(streamId, self.age)
317 streamWriteInt32(streamId, self.price)
318
319 for _, animatedObject in ipairs(self.animatedObjects) do
320 NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(animatedObject))
321 animatedObject:writeStream(streamId, connection)
322 g_server:registerObjectInStream(connection, animatedObject)
323 end
324 end
325 end
```

createNode

Description

Create node

Definition

createNode(string i3dFilename)

Arguments

string i3dFilename i3d file name

Return Values

boolean success success

Code

```
362 function Placeable:createNode(i3dFilename)
363 self.i3dFilename = i3dFilename
364 local nodeRoot = g_i3DManager:loadSharedI3DFile(i3dFilename, nil,
    false, false)
365 if nodeRoot == 0 then
366 return false
367 end
368
369 if self.useMultiRootNode then
370 link(getRootNode(), nodeRoot)
371 self.nodeId = nodeRoot
372 else
373 local nodeId = getChildAt(nodeRoot, 0)
```

```
374  if nodeId == 0 then
375  delete(nodeRoot)
376  return false
377  end
378  link(getRootNode(), nodeId)
379  delete(nodeRoot)
380  self.nodeId = nodeId
381  end
382
383  return true
384  end
```

setPreviewMaterials

Description

Sets the preview material to all nodes in the placeable

Definition

setPreviewMaterials(integer node, table nodeTable)

Arguments

integer node	id of node
table	nodeTable table to save the nodes

Code

```
390  function Placeable:setPreviewMaterials(node, nodeTable)
391  if getHasClassId(node, ClassIds.SHAPE) then
392  nodeTable[node] = node
393  setMaterial(node, Placeable.GLOW_MATERIAL, 0)
394  end
395
396  local numChildren = getNumOfChildren(node)
397  for i=0, numChildren-1 do
398  self:setPreviewMaterials(getChildAt(node, i), nodeTable)
399  end
400  end
```

setPlaceablePreviewState

Description

Set placable preview state

Definition

setPlaceablePreviewState(int state)

Arguments

int state	Placement state [Placeable.PREVIEW_STATE.CHECKING | Placeable.PREVIEW_STATE.VALID | Placeable.PREVIEW_STATE.INVALID]

Code

```
405  function Placeable:setPlaceablePreviewState(state)
406  if not self.isInPreviewMode then
```

```
407 self.isInPreviewMode = true
408
409 self.previewGlowingNodes = {}
410 if Placeable.GLOW_MATERIAL ~= nil then
411 -- replace materials with glowing material
412 self:setPreviewMaterials(self.nodeId, self.previewGlowingNodes)
413
414 -- load preview shapes for ramps
415 for _, area in pairs(self.rampAreas) do
416 local rampNode =
    g_i3DManager:loadSharedI3DFile(Placeable.RAMP_PREVIEW_PATH)
417 if rampNode ~= 0 then
418 link(area.root, rampNode) -- link the ramp root to the area ->
    can rotate the entire area hierarchy for preview
419 area.previewNode = rampNode
420
421 local rampRootTransform = getChildAt(rampNode, 0)
422 local rampShape = getChildAt(rampRootTransform, 0)
423 area.previewShape = rampShape
424
425 setMaterial(rampShape, Placeable.GLOW_MATERIAL, 0)
426
427 -- scale the ramp preview to the area dimensions, this assumes a
    rectangular area
428 local scaleX, _, scaleZ = getScale(rampShape)
429 local startX, startY, startZ = getWorldTranslation(area.start)
430 local widthX, widthY, widthZ = getWorldTranslation(area.width)
431 local heightX, heightY, heightZ =
    getWorldTranslation(area.height)
432
433 local width = MathUtil.vector3Length(widthX - startX, widthY -
    startY, widthZ - startZ)
434 local height = MathUtil.vector3Length(heightX - startX, heightY -
    startY, heightZ - startZ)
435
436 setScale(rampShape, width, Placeable.RAMP_PREVIEW_THICKNESS,
    height)
437 end
438 end
439 end
440 end
441 if Placeable.GLOW_MATERIAL ~= nil then
```

```
442 local color = Placeable.PREVIEW_COLOR[state]
443 for node in pairs(self.previewGlowingNodes) do
444 setShaderParameter(node, "colorScale", color[1], color[2],
    color[3], color[4], false)
445 end
446
447 local rampColor = Placeable.PREVIEW_RAMP_COLOR[state]
448 for _, area in pairs(self.rampAreas) do
449 if area.previewShape ~= nil then
450 setShaderParameter(area.previewShape, "colorScale", rampColor[1],
    rampColor[2], rampColor[3], rampColor[4], false)
451 end
452 end
453 end
454 end
```

load

Description

Load placeable

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position
float	y	z world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Return Values

boolean success success

Code

```
467 function Placeable:load(xmlFilename, x,y,z, rx,ry,rz, initRandom)
468 self.configFileName = xmlFilename
469 self.customEnvironment, self.baseDirectory =
    Utils.getModNameAndBaseDirectory(xmlFilename)
470
471 local xmlFile = loadXMLFile("TempXML", xmlFilename)
472 if xmlFile == 0 then
473 return false
474 end
475 local i3dFilename = getXMLString(xmlFile, "placeable.filename")
```

```
476 self.placementSizeX = Utils.getNoNil(getXMLFloat(xmlFile,
    "placeable.placement#sizeX"), self.placementSizeX)
477 self.placementSizeZ = Utils.getNoNil(getXMLFloat(xmlFile,
    "placeable.placement#sizeZ"), self.placementSizeZ)
478 self.placementTestSizeX = Utils.getNoNil(getXMLFloat(xmlFile,
    "placeable.placement#testSizeX"), self.placementSizeX)
479 self.placementTestSizeZ = Utils.getNoNil(getXMLFloat(xmlFile,
    "placeable.placement#testSizeZ"), self.placementSizeZ)
480 self.useRandomYRotation = Utils.getNoNil(getXMLBool(xmlFile,
    "placeable.placement#useRandomYRotation"),
    self.useRandomYRotation)
481 self.useManualYRotation = Utils.getNoNil(getXMLBool(xmlFile,
    "placeable.placement#useManualYRotation"),
    self.useManualYRotation)
482 self.alignToWorldY = Utils.getNoNil(getXMLBool(xmlFile,
    "placeable.placement#alignToWorldY"), true)
483 self.placementPositionSnapSize =
    math.abs(Utils.getNoNil(getXMLFloat(xmlFile,
    "placeable.placement#placementPositionSnapSize"), 0.0))
484 self.placementRotationSnapAngle =
    math.rad(math.abs(Utils.getNoNil(getXMLFloat(xmlFile,
    "placeable.placement#placementRotationSnapAngle"), 0.0)))
485
486 self.incomePerHour = getXMLFloat(xmlFile,
    "placeable.incomePerHour" ..
    g_currentMission.missionInfo.difficulty)
487 -- fallback for old single value format
488 if self.incomePerHour == nil then
489 self.incomePerHour = Utils.getNoNil(getXMLFloat(xmlFile,
    "placeable.incomePerHour"), 0)
490 if g_currentMission.missionInfo.difficulty == 1 then
491 self.incomePerHour = self.incomePerHour * 1.5
492 elseif g_currentMission.missionInfo.difficulty == 3 then
493 self.incomePerHour = self.incomePerHour / 1.5
494 end
495 end
496
497 local storeItem =
    g_storeManager:getItemByXMLFilename(self.configFileName)
498 if storeItem ~= nil then
499 if self.price == 0 or self.price == nil then
500 self.price = StoreItemUtil.getDefaultPrice(storeItem)
501 end
502
503 if g_currentMission ~= nil and storeItem.canBeSold then
```

```
504 g_currentMission.environment:addDayChangeListener(self)
505 end
506 end
507
508 if i3dFilename == nil then
509 delete(xmlFile)
510 return false
511 end
512 self.i3dFilename = Utils.getFilename(i3dFilename,
    self.baseDirectory)
513
514 if not self:createNode(self.i3dFilename) then
515 delete(xmlFile)
516 return false
517 end
518 self:initPose(x,y,z, rx,ry,rz, initRandom)
519
520 if hasXMLProperty(xmlFile, "placeable.dayNightObjects") then
521 local i = 0
522 while true do
523 local key =
    string.format("placeable.dayNightObjects.dayNightObject(%d)", i)
524 if not hasXMLProperty(xmlFile, key) then
525 break
526 end
527
528 local node = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, key.."#node"))
529 if node ~= nil then
530 if self.dayNightObjects == nil then
531 self.dayNightObjects = {}
532 g_currentMission.environment:addWeatherChangeListener(self)
533 end
534
535 local visibleDay = getXMLBool(xmlFile, key.."#visibleDay")
536 local visibleNight = getXMLBool(xmlFile, key.."#visibleNight")
537 local intensityDay = getXMLFloat(xmlFile, key.."#intensityDay")
538 local intensityNight = getXMLFloat(xmlFile,
    key.."#intensityNight")
539
```

```
540 table.insert(self.dayNightObjects, {node=node,
    visibleDay=visibleDay, visibleNight=visibleNight,
    intensityDay=intensityDay, intensityNight=intensityNight})
541 end
542 i = i + 1
543 end
544 end
545
546 -- load leveling properties
547 self.requireLeveling = Utils.getNoNil(getXMLBool(xmlFile,
    "placeable.leveling#requireLeveling"), self.requireLeveling)
548 if self.requireLeveling then
549 self.maxSmoothDistance = Utils.getNoNil(getXMLFloat(xmlFile,
    "placeable.leveling#maxSmoothDistance"), 3)
550 self.maxSlope =
    MathUtil.degToRad(Utils.getNoNil(getXMLFloat(xmlFile,
    "placeable.leveling#maxSlope"), 45))
551 self.maxEdgeAngle =
    MathUtil.degToRad(Utils.getNoNil(getXMLFloat(xmlFile,
    "placeable.leveling#maxEdgeAngle"), 45))
552 self.smoothingGroundType = getXMLString(xmlFile,
    "placeable.leveling#smoothingGroundType")
553 end
554
555 self:loadAreasFromXML(self.clearAreas, xmlFile,
    "placeable.clearAreas.clearArea(%d)", false, false)
556 self:loadAreasFromXML(self.levelAreas, xmlFile,
    "placeable.leveling.levelAreas.levelArea(%d)", false, true) --
    load leveling info
557 self:loadAreasFromXML(self.rampAreas, xmlFile,
    "placeable.leveling.rampAreas.rampArea(%d)", true, true) -- load
    ramps and leveling info
558 self:loadAreasFromXML(self.foliageAreas, xmlFile,
    "placeable.foliageAreas.foliageArea(%d)", false, false, true) --
    load ramps and leveling info
559
560 if hasXMLProperty(xmlFile, "placeable.tipOcclusionUpdateArea")
    then
561 local sizeX = getXMLFloat(xmlFile,
    "placeable.tipOcclusionUpdateArea#sizeX")
562 local sizeZ = getXMLFloat(xmlFile,
    "placeable.tipOcclusionUpdateArea#sizeZ")
563
564 if sizeX ~= nil and sizeZ ~= nil then
```

```
565 local centerX = Utils.getNoNil(getXMLFloat(xmlFile,
    "placeable.tipOcclusionUpdateArea#centerX"), 0)
566 local centerZ = Utils.getNoNil(getXMLFloat(xmlFile,
    "placeable.tipOcclusionUpdateArea#centerZ"), 0)
567 self.tipOcclusionUpdateArea = {centerX, centerZ, sizeX, sizeZ}
568 end
569 end
570
571 if not self.alignToWorldY then
572 self.pos1Node = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, "placeable.placement#pos1Node"))
573 self.pos2Node = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, "placeable.placement#pos2Node"))
574 self.pos3Node = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, "placeable.placement#pos3Node"))
575 if self.pos1Node == nil or self.pos2Node == nil or self.pos3Node
    == nil then
576 self.alignToWorldY = true
577 print("Warning: Pos1Node, Pos2Node and Pos3Node has to be set
    when alignToWorldY is false!")
578 end
579 end
580
581 if hasXMLProperty(xmlFile, "placeable.animatedObjects") then
582 local i = 0
583 while true do
584 local animationKey =
    string.format("placeable.animatedObjects.animatedObject(%d)", i)
585 if not hasXMLProperty(xmlFile, animationKey) then
586 break
587 end
588
589 local animatedObject = AnimatedObject:new(self.isServer,
    self.isClient)
590 if not animatedObject:load(self.nodeId, xmlFile, animationKey)
    then
591 print("Error: Failed to load animated object " .. tostring(i))
592 else
593 if self.isServer then
594 animatedObject:register(true)
595 end
596
597 table.insert(self.animatedObjects, animatedObject)
```

```
598 end
599
600 i = i + 1
601 end
602 end
603
604 local i = 0
605 while true do
606 local triggerMarkerKey =
    string.format("placeable.triggerMarkers.triggerMarker(%d)", i)
607 if not hasXMLProperty(xmlFile, triggerMarkerKey) then
608 break
609 end
610
611 local node = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, triggerMarkerKey.."#node"))
612 if node ~= nil then
613 table.insert(self.triggerMarkers, node)
614 g_currentMission:addTriggerMarker(node)
615 end
616
617 i = i + 1
618 end
619
620 if hasXMLProperty(xmlFile, "placeable.hotspots") then
621 local i = 0
622 while true do
623 local hotspotKey =
    string.format("placeable.hotspots.hotspot(%d)", i)
624 if not hasXMLProperty(xmlFile, hotspotKey) then
625 break
626 end
627
628 local hotspot = self:loadHotspotFromXML(xmlFile, hotspotKey)
629 if hotspot ~= nil then
630 g_currentMission:addMapHotspot(hotspot)
631 table.insert(self.mapHotspots, hotspot)
632 end
633
634 i = i + 1
635 end
```

```
636 end
637
638 if self.isClient then
639 self.samples.idle = g_soundManager:loadSampleFromXML(xmlFile,
    "placeable.sounds", "idle", self.baseDirectory, self.nodeId, 1,
    AudioGroup.ENVIRONMENT, nil, nil)
640 end
641
642 delete(xmlFile)
643
644 return true
645 end
```

loadAreasFromXML

Description

Load area definitions from XML into an area array.

Definition

loadAreasFromXML(table areaArray, string xmlFile, string xmlPathTemplate, bool isRamp)

Arguments

table	areaArray	Array instance which receives loaded area definitions.
string	xmlFile	Loaded XML file handle
string	xmlPathTemplate	XML element path template for an area type
bool	isRamp	If true, the targeted areas are ramps

loadAreaFromXML

Description

Load a single area definition from XML.
Areas are defined by three nodes: start, width and height. The start node denotes the origin of the area, while width
and height provide the dimensions of the spanned parallelogram. Usage of areas include clear areas (foliage is
cleared) and leveling areas (terrain is leveled) around placeable objects.

Definition

loadAreaFromXML(table area, integer xmlFile, string key, bool isRamp)

Arguments

table	area	Empty area definition table which receives the loaded node IDs.
integer	xmlFile	ID of the XML file
string	key	String Key to the XML element
bool	isRamp	If true, the targeted areas are ramps

Return Values

boolean success success

Code

```
685 function Placeable:loadAreaFromXML(area, xmlFile, key, isRamp,
    isLeveling)
686 local start = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, key .. "#startNode"))
```

```
687 local width = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, key .. "#widthNode"))
688 local height = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, key .. "#heightNode"))
689
690 if start ~= nil and width ~= nil and height ~= nil then
691 area.root = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, key .. "#rootNode"))
692 area.start = start
693 area.width = width
694 area.height = height
695 area.texture = getXMLString(xmlFile, key .. "#texture")
696
697 if isRamp then
698 local rx, ry, rz = getRotation(area.root)
699 area.baseRotation = {rx, ry, rz} -- store base rotation for
    resetting during preview
700 local rampSlope = getXMLFloat(xmlFile, key .. "#maxSlope")
701 area.maxSlope = rampSlope and MathUtil.degToRad(rampSlope) or
    self.maxSlope
702 end
703
704 if isLeveling then
705 area.groundType = getXMLString(xmlFile, key .. "#groundType")
706 end
707
708 return true
709 end
710
711 return false
712 end
```

loadFoliageAreaFromXML

Description

Load foliage definitons from XML.

Definition

loadFoliageAreaFromXML()

Code

```
716 function Placeable:loadFoliageAreaFromXML(area, xmlFile, key)
717 local rootNode = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, key .. "#rootNode"))
718 local fruitType = getXMLString(xmlFile, key .. "#fruitType")
```

```
719 local fruitTypeDesc = g_fruitTypeManager:getFruitTypeByName(fruitType)
720 local state = getXMLInt(xmlFile, key .. "#state")
721
722 if rootNode ~= nil and fruitTypeDesc ~= nil then
723 area.fruitType = fruitTypeDesc.index
724 area.fieldDimensions = rootNode
725 area.fruitState = Utils.getNoNil(state, fruitTypeDesc.maxHarvestingGrowthState - 1)
726
727 return true
728 end
729
730 return false
731 end
```

loadHotspotFromXML

Description

Load hotspot from XML definitons

Definition

loadHotspotFromXML()

Code

```
735 function Placeable:loadHotspotFromXML(xmlFile, key)
736 local name = Utils.getNoNil(getXMLString(xmlFile, key.."#name"), "")
737
738 local category = Utils.getNoNil(getXMLString(xmlFile, key.."#category"), "CATEGORY_TRIGGER")
739 if MapHotspot[category] ~= nil then
740 category = MapHotspot[category]
741 else
742 category = MapHotspot.CATEGORY_DEFAULT
743 end
744
745 local hotspot = MapHotspot:new(name, category)
746
747 local text = g_i18n:convertText(Utils.getNoNil(getXMLString(xmlFile, key.."#fullName"), ""))
748 if text ~= "" then
749 local showName = Utils.getNoNil(getXMLBool(xmlFile, key.."#showName"), false)
750
```

```
751 hotspot:setText(text, not showName)
752 end
753
754 local imageFilename = getXMLString(xmlFile,
    key.."#imageFilename")
755 if imageFilename ~= nil then
756 imageFilename = Utils.getFilename(imageFilename,
    self.baseDirectory)
757 end
758
759 local imageUVs =
    StringUtil.getVectorNFromString(getXMLString(xmlFile,
    key.."#imageUVs"), 4)
760 local imageName = getXMLString(xmlFile, key.."#imageName")
761 if imageName ~= nil and MapHotspot.UV[imageName] ~= nil then
762 imageUVs = MapHotspot.UV[imageName]
763 end
764 if imageUVs ~= nil then
765 imageUVs = getNormalizedUVs(imageUVs)
766 end
767 if imageUVs ~= nil then
768 local baseColor =
    StringUtil.getVectorNFromString(getXMLString(xmlFile,
    key.."#baseColor"), 4)
769 hotspot:setBorderedImage(imageFilename, imageUVs, baseColor)
770 end
771
772 local linkNode = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, key .. "#linkNode"))
773 if linkNode == nil then
774 linkNode = self.nodeId
775 end
776 hotspot:setLinkedNode(linkNode)
777
778 local width = getXMLFloat(xmlFile, key.."#width")
779 local height = getXMLFloat(xmlFile, key.."#height")
780 if width ~= nil and heigth ~= nil then
781 hotspot:setSize(width, height)
782 end
783
784 hotspot:setBlinking(Utils.getNoNil(getXMLBool(xmlFile,
    key.."#blinking"), false))
```

```
785 hotspot:setPersistent(Utils.getNoNil(getXMLBool(xmlFile,
    key.."#persistent"), false))
786 hotspot:setRenderLast(Utils.getNoNil(getXMLBool(xmlFile,
    key.."#renderLast"), false))
787
788 local textSize = getXMLInt(xmlFile, key.."#textSize")
789 if textSize ~= nil then
790 _, textSize = getNormalizedScreenValues(0, textSize)
791 hotspot:setTextOptions(textSize)
792 end
793
794 local hotspotTextOffset = Utils.getNoNil(getXMLString(xmlFile,
    key .. "#hotspotTextOffset"), "0px 0px")
795 hotspot:setRawTextOffset(hotspotTextOffset)
796
797 local textColor =
    StringUtil.getVectorNFromString(getXMLString(xmlFile,
    key.."#textColor"), 4)
798 hotspot:setTextOptions(nil, nil, nil, textColor)
799
800 return hotspot
801 end
```

finalizePlacement

Description

Called if placeable is placed

Definition

finalizePlacement()

Code

```
805 function Placeable:finalizePlacement()
806 if self.isInPreviewMode then
807 print("Error: Can't finalize placement of preview placeables")
808 end
809 if not self.isolated then
810
811 self:alignToTerrain()
812
813 addToPhysics(self.nodeId)
814 g_currentMission:addPlaceable(self)
815 g_currentMission:addItemToSave(self)
816 g_currentMission:addOwnedItem(self)
817 self:collectPickObjects(self.nodeId)
818
```

```
819 for _, node in pairs(self.pickObjects) do
820 g_currentMission:addNodeObject(node, self)
821 end
822
823 local missionInfo = g_currentMission.missionInfo
824 if self.isServer then
825 if not self.isLoadedFromSavegame or
826 (missionInfo.isValid and not
    (missionInfo:getIsTipCollisionValid(g_currentMission) and
    missionInfo:getIsPlacementCollisionValid(g_currentMission))) then
827 self:setTipOcclusionAreaDirty()
828 end
829 end
830 end
831
832 if self.isClient then
833 g_soundManager:playSample(self.samples.idle)
834 end
835
836 -- initially update dayNightObjects
837 self:weatherChanged()
838 g_currentMission.environment:addHourChangeListener(self)
839
840 local x,_,z = getWorldTranslation(self.nodeId)
841 self.farmlandId =
    g_farmlandManager:getFarmlandIdAtWorldPosition(x, z)
842
843 for _, hotspot in ipairs(self.mapHotspots) do
844 hotspot:setOwnerFarmId(self:getOwnerFarmId())
845 end
846
847 g_messageCenter:publish(MessageType.FARM_PROPERTY_CHANGED,
    {self:getOwnerFarmId()})
848 end
```

setTipOcclusionAreaDirty

Description

Set tip occlusion area dirty

Definition

setTipOcclusionAreaDirty()

Code

```
852 function Placeable:setTipOcclusionAreaDirty()
```

```
853  if self.tipOcclusionUpdateArea ~= nil and self.nodeId ~= 0 then
854  local x, z, sizeX, sizeZ = unpack(self.tipOcclusionUpdateArea)
855  local x1,_,z1 = localToWorld(self.nodeId, x+sizeX*0.5, 0,
     z+sizeZ*0.5)
856  local x2,_,z2 = localToWorld(self.nodeId, x-sizeX*0.5, 0,
     z+sizeZ*0.5)
857  local x3,_,z3 = localToWorld(self.nodeId, x+sizeX*0.5, 0, z-
     sizeZ*0.5)
858  local x4,_,z4 = localToWorld(self.nodeId, x-sizeX*0.5, 0, z-
     sizeZ*0.5)
859  local minX = math.min(math.min(x1, x2), math.min(x3, x4))
860  local maxX = math.max(math.max(x1, x2), math.max(x3, x4))
861  local minZ = math.min(math.min(z1, z2), math.min(z3, z4))
862  local maxZ = math.max(math.max(z1, z2), math.max(z3, z4))
863  g_densityMapHeightManager:setCollisionMapAreaDirty(minX, minZ,
     maxX, maxZ)
864  end
865  end
```

alignToTerrain

Description

Align placeable to terrain

Definition

alignToTerrain()

Code

```
869  function Placeable:alignToTerrain()
870  if not self.alignToWorldY then
871  local x1,y1,z1 = getWorldTranslation(self.nodeId)
872  y1 = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode,
     x1,y1,z1)
873  setTranslation(self.nodeId, x1,y1,z1)
874  local x2,y2,z2 = getWorldTranslation(self.pos1Node)
875  y2 = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode,
     x2,y2,z2)
876  local x3,y3,z3 = getWorldTranslation(self.pos2Node)
877  y3 = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode,
     x3,y3,z3)
878  local x4,y4,z4 = getWorldTranslation(self.pos3Node)
879  y4 = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode,
     x4,y4,z4)
880  local dirX = x2 - x1
881  local dirY = y2 - y1
882  local dirZ = z2 - z1
883  local dir2X = x3 - x4
```

```
884 local dir2Y = y3 - y4
885 local dir2Z = z3 - z4
886 local upX,upY,upZ = MathUtil.crossProduct(dir2X, dir2Y, dir2Z,
    dirX, dirY, dirZ)
887 setDirection(self.nodeId, dirX, dirY, dirZ, upX,upY,upZ)
888 end
889 end
```

clearFoliageAndTipAreas

Description

Clear foliage and tipAny from clearAreas

Definition

clearFoliageAndTipAreas()

Code

```
893 function Placeable:clearFoliageAndTipAreas()
894 if self.isServer then
895 for _, areas in pairs{self.clearAreas, self.levelAreas,
    self.rampAreas} do
896 for _, area in pairs(areas) do
897 local x,_,z = getWorldTranslation(area.start)
898 local x1,_,z1 = getWorldTranslation(area.width)
899 local x2,_,z2 = getWorldTranslation(area.height)
900 -- clear foliage
901 FSDensityMapUtil.removeFieldArea(x, z, x1, z1, x2, z2)
902 FSDensityMapUtil.removeWeedArea(x, z, x1, z1, x2, z2)
903 FSDensityMapUtil.eraseTireTrack(x, z, x1, z1, x2, z2)
904 -- clear tipAny
905 DensityMapHeightUtil.clearArea(x, z, x1, z1, x2, z2)
906 end
907 end
908
909 -- Add foliage again
910 for _, area in pairs(self.foliageAreas) do
911 FieldUtil.setAreaFruit(area.fieldDimensions, area.fruitType,
    area.fruitState)
912 end
913 end
914 end
```

initPose

Description

Initialize pose

Definition

initPose(float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

float	x	x world position
float	y	y world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Code

```
925  function Placeable:initPose(x,y,z, rx,ry,rz, initRandom)
926  setTranslation(self.nodeId, x, y, z)
927  setRotation(self.nodeId, rx, ry, rz)
928  end
```

collectPickObjects

Description

Collect pick objects

Definition

collectPickObjects(integer node)

Arguments

integer node node id

Code

```
933  function Placeable:collectPickObjects(node)
934  if getRigidBodyType(node) ~= "NoRigidBody" then
935  table.insert(self.pickObjects, node)
936  end
937  local numChildren = getNumOfChildren(node)
938  for i=1, numChildren do
939  self:collectPickObjects(getChildAt(node, i-1))
940  end
941  end
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key, boolean resetVehicles)

Arguments

integer	xmlFile	id of xml object
string	key	key
boolean	resetVehicles	reset vehicles

Return Values

boolean success success

Code

```
949  function Placeable:loadFromXMLFile(xmlFile, key, resetVehicles)
```

```
950  local x,y,z = StringUtil.getVectorFromString(getXMLString(xmlFile, key.."#position"))
951  local xRot,yRot,zRot = StringUtil.getVectorFromString(getXMLString(xmlFile, key.."#rotation"))
952  if x == nil or y == nil or z == nil or xRot == nil or yRot == nil or zRot == nil then
953  return false
954  end
955
956  xRot = math.rad(xRot)
957  yRot = math.rad(yRot)
958  zRot = math.rad(zRot)
959
960  local xmlFilename = getXMLString(xmlFile, key.."#filename")
961  if xmlFilename == nil then
962  return false
963  end
964  xmlFilename = NetworkUtil.convertFromNetworkFilename(xmlFilename)
965
966  if self:load(xmlFilename, x,y,z, xRot, yRot, zRot, false, false) then
967  self.age = Utils.getNoNil(getXMLFloat(xmlFile, key.."#age"), 0)
968  self.price = Utils.getNoNil(getXMLInt(xmlFile, key.."#price"), self.price)
969
970  -- Use a call so any sub-objects of the placeable can be updated
971  self:setOwnerFarmId(Utils.getNoNil(getXMLInt(xmlFile, key .. "#farmId"), AccessHandler.EVERYONE), true)
972
973  self.mapBoundId = Utils.getNoNil(getXMLString(xmlFile, key .. "#mapBoundId"), self.mapBoundId)
974  self.isLoadedFromSavegame = true
975  self:finalizePlacement()
976
977  for i, animatedObject in ipairs(self.animatedObjects) do
978  animatedObject:loadFromXMLFile(xmlFile, string.format("%s.animatedObjects.animatedObject(%d)", key, i - 1))
979  end
980
981  return true
```

```
982  else
983  return false
984  end
985  end
```

saveToXMLFile

Description

Get save attributes and nodes

Definition

saveToXMLFile(string nodeIdent)

Arguments

string nodeIdent node ident

Return Values

string attributes attributes

string nodes nodes

Code

```
992  function Placeable:saveToXMLFile(xmlFile, key, usedModNames)
993  local x,y,z = getTranslation(self.nodeId)
994  local xRot,yRot,zRot = getRotation(self.nodeId)
995
996  setXMLString(xmlFile, key.."#filename",
     HTMLUtil.encodeToHTML(NetworkUtil.convertToNetworkFilename(self.configF
997  setXMLString(xmlFile, key.."#position", string.format("%.4f %.4f %.4f",
998  setXMLString(xmlFile, key.."#rotation", string.format("%.4f %.4f %.4f",
     math.deg(xRot), math.deg(yRot), math.deg(zRot)))
999  setXMLInt(xmlFile, key.."#age", self.age)
1000 setXMLFloat(xmlFile, key.."#price", self.price)
1001 setXMLInt(xmlFile, key.."#farmId", self:getOwnerFarmId())
1002
1003 if self.mapBoundId ~= nil then
1004 setXMLString(xmlFile, key.."#mapBoundId", self.mapBoundId)
1005 end
1006
1007 for i, animatedObject in ipairs(self.animatedObjects) do
1008 animatedObject:saveToXMLFile(xmlFile,
     string.format("%s.animatedObjects.animatedObject(%d)", key, i - 1), use
1009 end
1010 end
```

update

Description

Update

Definition

update(float dt)

Arguments

float dt time since last call in ms

Code

```
1020  function Placeable:update(dt)
1021  end
```

getPrice

Description

Returns price

Definition

getPrice()

Return Values

integer price price

Code

```
1029  function Placeable:getPrice()
1030  return self.price
1031  end
```

canBuy

Description

Returns true if we can place a building
checking item count

Definition

canBuy()

Return Values

Code

```
1037  function Placeable:canBuy()
1038  local storeItem =
      g_storeManager:getItemByXMLFilename(self.configFileName)
1039  local enoughItems = storeItem.maxItemCount == nil or
      (storeItem.maxItemCount ~= nil and
      g_currentMission:getNumOfItems(storeItem,
      g_currentMission:getFarmId()) < storeItem.maxItemCount)
1040  return enoughItems
1041  end
```

onBuy

Description

Called on buy

Definition

onBuy()

Code

```
1053  function Placeable:onBuy()
1054  end
```

onSell

Description

Called on sell

Definition

onSell()

Code

```
1058  function Placeable:onSell()
1059  if self.isServer then
1060  self:setTipOcclusionAreaDirty()
1061  end
1062
1063  g_messageCenter:publish(MessageType.FARM_PROPERTY_CHANGED,
      {self:getOwnerFarmId()})
1064  end
```

getDailyUpkeep

Description

Returns daily up keep

Definition

getDailyUpkeep()

Return Values

integer dailyUpkeep daily up keep

Code

```
1069  function Placeable:getDailyUpkeep()
1070  local storeItem =
      g_storeManager:getItemByXMLFilename(self.configFileName)
1071  local multiplier = 1
1072  if storeItem.lifetime ~= nil and storeItem.lifetime ~= 0 then
1073  local ageMultiplier = math.min(self.age/storeItem.lifetime, 1)
1074  multiplier = 1 + EconomyManager.MAX_DAILYUPKEEP_MULTIPLIER *
      ageMultiplier
1075  end
1076  return StoreItemUtil.getDailyUpkeep(storeItem, nil) * multiplier
1077  end
```

getSellPrice

Description

Returns sell price

Definition

getSellPrice()

Return Values

integer sellPrice sell price

Code

```
1082  function Placeable:getSellPrice()
1083  local priceMultiplier = 0.5
```

```
1084  local storeItem = g_storeManager:getItemByXMLFilename(self.configFileName)
1085  local maxVehicleAge = storeItem.lifetime
1086
1087  if maxVehicleAge ~= nil and maxVehicleAge ~= 0 then
1088  priceMultiplier = priceMultiplier * math.exp(-3.5 * math.min(self.age/maxVehicleAge, 1))
1089  end
1090
1091  return math.floor(self.price * math.max(priceMultiplier, 0.05))
1092  end
```

isMapBound

Description

Get wether the placeable is bound to the map and should be matched with the original map definition.
Does not currently influence anything. Once it doe: the position and rotation of the placeable will be updated
By the map info. It will also be removed if it was from the map.

Definition

isMapBound()

hourChanged

Description

Called if hour changed

Definition

hourChanged()

Code

```
1103  function Placeable:hourChanged()
1104  if self.isServer then
1105  if self.incomePerHour ~= 0 then
1106  g_currentMission:addMoney(self.incomePerHour, self:getOwnerFarmId(), "propertyIncome")
1107  g_currentMission:addMoneyChange(self.incomePerHour, self:getOwnerFarmId(), EconomyManager.MONEY_TYPE_PROPERTY_INCOME)
1108  end
1109  end
1110  end
```

dayChanged

Description

Called if day changed

Definition

dayChanged()

Code

```
1114  function Placeable:dayChanged()
1115  self.age = self.age + 1
1116  end
```

weatherChanged

Description

Called if weather changed

Definition

weatherChanged()

Code

```
1120  function Placeable:weatherChanged()
1121  if g_currentMission ~= nil and g_currentMission.environment ~=
      nil and self.dayNightObjects ~= nil then
1122  for _, dayNightObject in pairs(self.dayNightObjects) do
1123  if dayNightObject.visibleDay ~= nil and
      dayNightObject.visibleNight ~= nil then
1124  setVisibility(dayNightObject.node,
      (g_currentMission.environment.isSunOn and
      dayNightObject.visibleDay) or (dayNightObject.visibleNight and
      (not g_currentMission.environment.isSunOn and not
      g_currentMission.environment.weather:getIsRaining())))
1125
1126  elseif dayNightObject.intensityDay ~= nil and
      dayNightObject.intensityNight ~= nil then
1127  local intensity = dayNightObject.intensityNight
1128  if g_currentMission.environment.isSunOn then
1129  intensity = dayNightObject.intensityDay
1130  end
1131
1132  local _,y,z,w = getShaderParameter(dayNightObject.node,
      "lightControl")
1133  setShaderParameter(dayNightObject.node, "lightControl",
      intensity, y, z, w, false)
1134  end
1135  end
1136  end
1137  end
```

getTerrainModificationBlockingAreas

Description

Get an array of areas which this placeable is going to modify on placement.
These areas are used to modify the placement blocking bit vector map to avoid interfering with terrain modifications
of other placeables.

Definition

getTerrainModificationBlockingAreas()

Return Values

{i = {x, y, z, side1x, side1y, side1z, side2x, side2y, side2z}}, all numbers in world space coordinates

loadSpecValueIncomePerHour

Description

Loads capacity spec value

Definition

loadSpecValueIncomePerHour(integer xmlFile, string customEnvironment)

Arguments

integer xmlFile id of xml object

string customEnvironment custom environment

Return Values

table capacityAndUnit capacity and unit

Code

```
1167  function Placeable.loadSpecValueIncomePerHour(xmlFile, customEnvironment)
1168      if not hasXMLProperty(xmlFile, "placeable.incomePerHour1") then
1169          return nil
1170      end
1171
1172      local incomePerHour = {}
1173      incomePerHour[1] = Utils.getNoNil(getXMLFloat(xmlFile, "placeable.incomePerHour1"), 0)
1174      incomePerHour[2] = Utils.getNoNil(getXMLFloat(xmlFile, "placeable.incomePerHour2"), 0)
1175      incomePerHour[3] = Utils.getNoNil(getXMLFloat(xmlFile, "placeable.incomePerHour3"), 0)
1176      return incomePerHour
1177  end
```

getSpecValueIncomePerHour

Description

Returns value of income per hour

Definition

getSpecValueIncomePerHour(table storeItem, table realItem)

Arguments

table storeItem store item

table realItem real item

Return Values

integer incomePerHour income per hour

Code

```
1184  function Placeable.getSpecValueIncomePerHour(storeItem, realItem)
1185      if storeItem.specs.incomePerHour == nil then
1186          return nil
1187      end
```

```
1188  return string.format(g_i18n:getText("shop_incomeValue"),
      g_i18n:formatMoney(storeItem.specs.incomePerHour[g_currentMission.missi
1189  end
```

PlacementScreen
Description

Object Placement HUD Screen.
-- Shown in-game when placing objects.
-- @field messageText Hint message display text

new

Description

Create a new PlacementScreen instance.

Definition

new(table target, table custom_mt, table messageCenter, table placementController)

Arguments

table target	ScreenElement controller instance
table custom_mt	[optional] Sub-class meta table for inheritance
table messageCenter	MessageCenter reference for local network UI event handling
table placementController	PlacementScreenController instance which handles placement logic

Return Values

table instance	instance of object
table PlacementScreen	instance

mouseEvent

Description

Handle mouse input event.
Custom logic for PC version mouse input.

Definition

mouseEvent()

Return Values

table instance instance of object

updateMessageText

Description

Update message text, if present.

Definition

updateMessageText()

Return Values

boolean true if loading was successful else false

setPlacementItem

Description

Set a placeable object for placement or selling.

Definition

setPlacementItem(item Placement, isSellMode If, obj Sellable)

Arguments

item	Placement item from the store

isSellMode	If	true, we want to sell the given item
obj	Sellable	object

Return Values

boolean success success

onMouseModeChanged

Description

Called when the mouse input mode changes.

Definition

onMouseModeChanged()

Return Values

table instance instance of object

handleControllerMessage

Description

Handles messages dispatched to this view by the controller.

Definition

handleControllerMessage(messageId Message, text Message, callback [optional], callbackTarget [optional])

Arguments

messageId	Message	ID as defined in PlacementScreenController.MESSAGE
text	Message	text to display
callback	[optional]	Callback for message dialog completion
callbackTarget	[optional]	Callback target which is given as the first argument to the callback if supplied

Return Values

boolean true if loading was successful else false

PlacementScreenController

Description

Placeable placement controller.
-- Handles placement and selling logic for placeable objects in the PlacementScreen view.
--@category GUI

new

Description

Create a new PlacementScreenController instance.

Definition

new(table l10n, table inputManager, table placeableTypeManager)

Arguments

table l10n	I18N reference for string localization.
table inputManager	InputBinding reference for input action registration
table placeableTypeManager	PlaceableTypeManager reference

Return Values

table categories a list of categories

setMouseModeChangeCallback

Description

Set the callback used to trigger a mouse mode change.

Definition

setMouseModeChangeCallback(function callback)

Arguments

function callback Mouse mode callback function, signature: function(isMouseMode)

Return Values

table category the corresponding category

setMessageDispatchCallback

Description

Set the callback used to dispatch UI messages.

Definition

setMessageDispatchCallback(function callback)

Arguments

function callback Message dispatch callback function, signature: function(messageId, text, callbackFunction, callbackTarget)

Return Values

table instance instance of object

setExitCallback

Description

Set the callback used to exit placement mode.

Definition

setExitCallback(function callback)

Arguments

function callback Exit callback, signature: function()

Return Values

table instance instance of object

setClient

Description

Set the reference to the network client.

Definition

setClient()

Return Values

table instance instance of object

setCurrentMission

Description

Set the reference to the current mission controller when starting a game.

Definition

setCurrentMission()

Return Values

table mod the mod object

setHUD

Description

Set the reference to the in-game HUD.

Definition

setHUD()

Return Values

boolean success true if mod was removed, else false

initialize

Description

Initialize the controller.

Definition

initialize()

Return Values

table mod the mod object

initializeCamera

Description

Set up the placement camera.

Definition

initializeCamera()

Return Values

table mod the mod object

reset

Description

Reset controller state, e.g. when leaving the controlled view.

Definition

reset()

Return Values

table mod the mod object

updateCameraMovement

Description

Update camera position and orientation based on player input.

Definition

updateCameraMovement(dt Delta, movementMultiplier Speed)

Arguments

dt Delta time in milliseconds

movementMultiplier Speed factor for movement

Return Values

table mods a list of mods

getMouseEdgeScrollingMovement

Description

Get camera movement for mouse edge scrolling.

Definition

getMouseEdgeScrollingMovement()

Return Values

table mods a list of multiplayer mods

x direction movement [-1, 1]

Z	direction movement [-1, 1]

applyCameraMovement

Description

Apply a movement to the camera (and view).

Definition

applyCameraMovement(moveX X, moveZ Z, movementMultiplier Movement)

Arguments

moveX	X	direction movement [-1, 1]
moveZ	Z	direction movement [-1, 1]
movementMultiplier	Movement	speed factor

Return Values

table mods a list of active mods

updateCameraPosition

Description

Update the camera position and orientation based on terrain and zoom state.

Definition

updateCameraPosition()

Return Values

integer numMods number of mods

updatePlaceablePreview

Description

Update the placeable preview.

Definition

updatePlaceablePreview()

Return Values

integer numMods number of valid mods

acceptSelection

Description

Accept the current selection (selling or placing object).

Definition

acceptSelection()

Return Values

boolean areAvailable true if all hashes are available else false

sellWarningInfoOk

Description

Called when the player acknowledges the sell warning.

Definition

sellWarningInfoOk()

Return Values

boolean isAvailable true if hash is available else false

onSellCallback

Description

Called when the player confirms selling a placeable.

Definition

onSellCallback()

Return Values

table instance instance of object

updateSlots

Description

Update placeable object slots usage.

Definition

updateSlots()

Return Values

boolean true if loading was successful else false

onPlaceableBought

Description

Called on BuyPlaceableEvent success.

Definition

onPlaceableBought()

Return Values

boolean success true if added else false

onPlaceableBuyFailed

Description

Called on BuyPlaceableEvent failure.

Definition

onPlaceableBuyFailed()

Return Values

table instance instance of object

onPlaceableSold

Description

Called on SellPlaceableEvent success.

Definition

onPlaceableSold()

Return Values

table instance instance of object

onPlaceableSellFailed

Description

Called on SellPlaceableEvent failure.

Definition

onPlaceableSellFailed()

Return Values

table instance instance of object

buyPlaceable

Description

Buy a currently previewed placeable.
This operation also makes some final checks and dispatches information dialogs if anything goes wrong.

Definition

buyPlaceable()

Return Values

boolean true if loading was successful else false

findSellObjectAt

Description

Find a sellable object under a cursor position.
The results of the search are processed in PlacementScreenController.onSellObjectRaycast().

Definition

findSellObjectAt(posX Cursor, posY Cursor)

Arguments

posX Cursor X position in screen space

posY Cursor Y position in screen space

Return Values

table baleType baleType object

sellPlaceable

Description

Sell a given placeable object.

Definition

sellPlaceable()

Return Values

integer toolTypeIndex tool type index

setPlacementItem

Description

Set a placeable object for placement or selling.

Definition

setPlacementItem(item Placement, isSellMode If, obj Sellable)

Arguments

item	Placement	item from the store
isSellMode	If	true, we want to sell the given item
obj	Sellable	object

Return Values

table instance instance of object

calculatePlacementHeight

Description

Calculate the placement height for a given world position based on the current player input height factor.

Definition

calculatePlacementHeight()

Return Values

boolean true if loading was successful else false

isPlacementValid

Description

Check if a preview placement is valid (without terrain modification).

Definition

isPlacementValid(Placeable Placeable, x Preview, y Preview, Z Preview, yRot Placeable, distance Distance)

Arguments

Placeable	Placeable	instance
x	Preview	X position in world space
y	Preview	Y position in world space
Z	Preview	Z position in world space
yRot	Placeable	Y rotation in radians
distance	Distance	from checking ray origin to ground

Return Values

boolean	true	if loading was successful else false
True	if	the placement is valid, false otherwise
string	Reason	of being invalid

onPlacementRaycast

Description

Called when the placement raycast finishes.

Definition

onPlacementRaycast(hitObjectId ID, x Raycast, y Raycast, Z Raycast, distance Distance)

Arguments

hitObjectId	ID	of the first hit object
x	Raycast	hit X position in world space
y	Raycast	hit Y position in world space
Z	Raycast	hit Z position in world space
distance	Distance	from ray origin to hit position

Return Values

table sprayType sprayType object

startPlacementTerrainValidation

Description

Start validation of a required terrain deformation for the current placement preview.

Definition

startPlacementTerrainValidation()

Return Values

table sprayType the sprayType object

addPlaceableLevelingArea

Description

Add a leveling area of a placeable to a terrain deformation object.

Definition

addPlaceableLevelingArea(terrainDeform TerrainDeformation, levelArea Table, terrainBrushId Terrain)

Arguments

terrainDeform	TerrainDeformation	instance
levelArea	Table	which holds area nodes, {start=origin node, width=first side area delimiter node, height=second side area delimiter node}
terrainBrushId	Terrain	brush ID, currently this is a map layer index (zero-based)

Return Values

table sprayType the sprayType object

addPlaceableRampArea

Description

Add a leveling area for a ramp of a placeable to a terrain deformation object.

Definition

addPlaceableRampArea(terrainDeform TerrainDeformation, rampArea Table, terrainBrushId Terrain, maxRampSlope Maximum, terrainRootNode Root)

Arguments

terrainDeform	TerrainDeformation	instance
rampArea	Table	which holds area nodes, {start=origin node, width=first side area delimiter node, height=second side area delimiter node, baseRotation={rx, ry, rz}, previewShape=nodeId}
terrainBrushId	Terrain	brush ID, currently this is a map layer index (zero-based)
maxRampSlope	Maximum	ramp slope as an angle in radians
terrainRootNode	Root	node of the terrain for height map check

Return Values

string fillTypeName the sprayType name

onTerrainValidationFinished

Description

Called when a terrain deformation validation has finished.

Definition

onTerrainValidationFinished(canDeform If, displacedVolume Volume, overlapsBlockedArea If)

Arguments

canDeform	If	true, the terrain deformation could be performed
displacedVolume	Volume	displaced by the terrain deformation operation in cubic meters (m3)
overlapsBlockedArea	If	true, the terrain modification overlaps a blocked area.

Return Values

integer fillTypeIndex the sprayType index

onTerrainDeformationFinished

Description

Called when an actual (not just preview) terrain deformation operation has finished.

Definition

onTerrainDeformationFinished(canDeform If, displacedVolume Volume)

Arguments

canDeform	If	true, the terrain deformation was performed

displacedVolume Volume displaced by the terrain deformation operation in cubic meters (m3)

Return Values

table sprayType the sprayType object

cancelTerrainDeformation

Description

Cancel a currently active terrain deformation process, if necessary.
This will cancel any ongoing deformation checks and operations and release all relevant blocking flags.

Definition

cancelTerrainDeformation()

Return Values

integer sprayTypeIndex the sprayType index

onSellObjectRaycast

Description

Called when the sell object raycast finishes.

Definition

onSellObjectRaycast(hitObjectId ID, x Raycast, y Raycast, Z Raycast, distance Distance)

Arguments

hitObjectId	ID	of the first hit object
x	Raycast	hit X position in world space
y	Raycast	hit Y position in world space
Z	Raycast	hit Z position in world space
distance	Distance	from ray origin to hit position

Return Values

table sprayTypes list of sprayTypes

determineCameraPosition

Description

Determine the current camera position and orientation.

Definition

determineCameraPosition()

Return Values

table	instance	instance of object
Camera X		world space position
Camera Z		world space position
Camera Y		rotation in radians

canBuy

Description

Determine if there is enough storage room to buy a currently previewed placeable object.

Definition

canBuy()

Return Values

table instance instance of object

canPlace

Description

Determine if a previewed placeable object can be placed at its current position.

Definition

canPlace()

Return Values

boolean true if loading was successful else false

resetInputState

Description

Reset event input state.

Definition

resetInputState()

Return Values

table baleType baleType object

registerActionEvents

Description

Register required action events.

Definition

registerActionEvents()

Return Values

string toolTypeName tool type name

removeActionEvents

Description

Remove action events registered on this screen.

Definition

removeActionEvents()

Return Values

integer toolTypeIndex tool type index

updateActionEvents

Description

Update action event activity states.

Definition

updateActionEvents()

Return Values

integer numToolTypes number of tool types

PlayerSetFarmAnswerEvent

Description

Player farm setting answer event.
-- Triggered in response to PlayerSetFarmEvent.

emptyNew

Description

Create an empty instance

Definition

emptyNew()

Return Values

table instance Instance of object

Code

```
27  function PlayerSetFarmAnswerEvent:emptyNew()
28  local self = Event:new(PlayerSetFarmAnswerEvent_mt)
29  return self
30  end
```

new

Description

Create an instance of PlayerSetFarmAnswerEvent.

Definition

new(int answerState, int farmId, string password)

Arguments

int	answerState	
int	farmId	Farm ID
string	password	Password used for PlayerSetFarmEvent

Return Values

table instance Instance of PlayerSetFarmAnswerEvent

Code

```
38  function PlayerSetFarmAnswerEvent:new(answerState, farmId,
    password)
39  local self = PlayerSetFarmAnswerEvent:emptyNew()
40
41  self.answerState = answerState
42  self.farmId = farmId
43  self.password = password
44
45  return self
46  end
```

writeStream

Description

Writes network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer	streamId	network stream identification
table	connection	connection information

Code

```
52  function PlayerSetFarmAnswerEvent:writeStream(streamId,
    connection)
53  streamWriteUIntN(streamId, self.answerState,
    PlayerSetFarmAnswerEvent.SEND_NUM_BITS)
```

```
54  streamWriteUIntN(streamId, self.farmId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
55
56  local passwordCorrect = self.answerState ==
    PlayerSetFarmAnswerEvent.STATE.OK
57  local passwordSet = self.password ~= nil
58  if streamWriteBool(streamId, passwordCorrect and passwordSet) then
59  streamWriteString(streamId, self.password)
60  end
61  end
```

readStream

Description

Reads network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
67  function PlayerSetFarmAnswerEvent:readStream(streamId, connection)
68  self.answerState = streamReadUIntN(streamId,
    PlayerSetFarmAnswerEvent.SEND_NUM_BITS)
69  self.farmId = streamReadUIntN(streamId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
70
71  if streamReadBool(streamId) then
72  self.password = streamReadString(streamId)
73  end
74
75  self:run(connection)
76  end
```

run

Description

Run event

Definition

run(table connection)

Arguments

table connection connection information

Code

```
81  function PlayerSetFarmAnswerEvent:run(connection)
82  if not connection:getIsServer() then -- server side, should not
    happen
```

```
83  g_logManager:devWarning("PlayerSetFarmAnswerEvent is a server to
    client only event")
84  else -- client side
85  if self.answerState == PlayerSetFarmAnswerEvent.STATE.OK then
86  g_messageCenter:publish(PlayerSetFarmAnswerEvent,
    {self.answerState, self.farmId, self.password})
87  elseif self.answerState ==
    PlayerSetFarmAnswerEvent.STATE.PASSWORD_REQUIRED then
88  g_messageCenter:publish(PlayerSetFarmAnswerEvent,
    {self.answerState, self.farmId})
89  end
90  end
91  end
```

PlaySampleMixin

Description

Play UI sound sample mixin.
-- Add this mixin to a GuiElement to enable it to play UI sounds.
-- Added methods:
GuiElement:setPlaySampleCallback(callback): Set a callback for playing UI sound samples, signature: function(sampleName).
GuiElement:playSample(index, count): Called by the decorated GuiElement to play a sound sample using a name from GuiSoundPlayer.SOUND_SAMPLES.
GuiElement:disablePlaySample(): Permanently disables playing samples for special cases (i.e. separate sound logic)
--@category GUI

addTo

Description

See GuiMixin:addTo().

Definition

addTo()

setPlaySampleCallback

Description

Set a callback to play a UI sound sample.

Definition

setPlaySampleCallback(table guiElement, function callback)

Arguments

table	guiElement	GuiElement instance
function	callback	Play sample callback, signature: function(sampleName)

playSample

Description

Request playing a UI sound sample identified by name.

Definition

playSample(table guiElement, string sampleName)

Arguments

table guiElement GuiElement instance
string sampleName Sample name, use one of GuiSoundPlayer.SOUND_SAMPLES.

disablePlaySample

Description

Permanently disable playing samples on the decorated GuiElement for special cases.

Definition

disablePlaySample()

clone

Description

Clone this mixin's state from a source to a destination GuiElement instance.

Definition

clone()

PolygonChain

Description

Creating data grid

new

Description

@param table customMt custom metatable

Definition

new()

Return Values

table instance instance of object

Code

```
19  function PolygonChain:new(customMt)
20  local self = {}
21  setmetatable(self, customMt or PolygonChain_mt)
22
23  self.controlNodes = {}
24
25  return self
26  end
```

delete

Description

Deletes data grid

Definition

delete()

Code

```
30  function PolygonChain:delete()
31  self.controlNodes = nil
32  end
```

RenderElement

Description

Render display as an overlay
--@category GUI
--@xmlConfig RenderElement#filename string Path to the i3d to be rendered in the overlay

createScene

Description

Create the scene and the overlay. Call destroyScene to clean up resources.

Definition

createScene()

destroyScene

Description

Destroy the scene and the overlay, cleaning up resources.

Definition

destroyScene()

SavegameController
Description

Savegame persistence controller.
-- Handles loading and saving of mission game states.

new

Description

Create a new instance of SavegameController.

Definition

new()

Return Values

float maxPtoRpm max pto rpm

loadSavegames

Description

Load the savegame meta data list.

Definition

loadSavegames()

Return Values

float neededPower needed power

resetStorageDeviceSelection

Description

Reset the storage device selection for saving.

Definition

resetStorageDeviceSelection()

Return Values

string l10n l10n text

float neededPower needed power in kw

float neededPower needed power in hp

updateSavegames

Description

Start updating the savegame list in the engine.

Definition

updateSavegames()

onSaveGameUpdateComplete

Description

Called when updating the save game list in the engine has finished.

Definition

onSaveGameUpdateComplete()

locateBackups

Description

Locate backups for a savegame instance in its backup path.

Definition

locateBackups(string backupBasePath, string backupDirBase)

Arguments

string	backupBasePath	Backup base path
string	backupDirBase	Savegame-specific backup base directory

Return Values

boolean	hasPrerequisite	true if all prerequisite specializations are loaded

assignBackupDeleteFlags

Description

Check and mark backups for deletion, based on creation time.

Definition

assignBackupDeleteFlags()

Return Values

boolean

createBackup

Description

Create an actual savegame backup.

Definition

createBackup(table savegame, string backupBasePath, string backupDirFull, string backupDirBase)

Arguments

table	savegame	Savegame to back up
string	backupBasePath	Backup base path
string	backupDirFull	Full savegame-specific backup directory
string	backupDirBase	Savegame-specific backup base directory

Return Values

boolean	updated	part was updated

backupSavegame

Description

Create a backup of the given savegame.

Definition

backupSavegame()

Return Values

boolean isActive is active

onSaveStartComplete

Description

Called when saving can begin or there is an IO problem with the target path.

Definition

onSaveStartComplete()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

onSaveComplete

Description

Called when the saving game process has finished.

Definition

onSaveComplete()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

onSavegameDeleted

Description

Called when a savegame has been deleted by the engine or an error occurred.

Definition

onSavegameDeleted()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

getCanStartGame

Description

Check if a savegame can be started.

Definition

getCanStartGame()

Return Values

boolean canFold ridge markers can be folden

getCanDeleteGame

Description

Check if a savegame can be deleted.

Definition

getCanDeleteGame()

Return Values

boolean success success

getSavegame

Description

Get a savegame by index.

Definition

getSavegame()

Return Values

boolean success success

getIsSaving

Description

Check if a game is being saved.

Definition

getIsSaving()

Return Values

boolean isActive speed rotating part is active

getSavingErrorCode

Description

Get the current saving error code.

Definition

getSavingErrorCode()

Return Values

boolean isActive work area is active

getIsWaitingForSavegameInfo

Description

Check if the save game info is currently being updated.

Definition

getIsWaitingForSavegameInfo()

Return Values

float dirtMultiplier current wear multiplier

getNumberOfSavegames

Description

Get the number of known savegames from the engine.

Definition

getNumberOfSavegames()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

isStorageDeviceUnavailable

Description

Check if the storage device is unavailable

Definition

isStorageDeviceUnavailable()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

ScreenElement

Description

Base screen element. All full-screen GUI views inherit from this.
-- ScreenElement inherits from FrameElement and has no additional configuration, but contains UI logic shared across all

full screen views.
--@category GUI
-- @field pageSelector Paging navigation controls container, only defined if the view supports paging by configuration

onClickOk

Description

Handle OK click event.

Definition

onClickOk()

Return Values

bool True if the event was not used, false if it was used.

onClickActivate

Description

Handle activate click event.

Definition

onClickActivate()

Return Values

bool True if the event was not used, false if it was used.

onClickCancel

Description

Handle cancel click event.

Definition

onClickCancel()

Return Values

bool True if the event was not used, false if it was used.

onPagePrevious

Description

Handle previous page event.

Definition

onPagePrevious()

onPageNext

Description

Handle next page event.

Definition

onPageNext()

onClickBack

Description

Handle back click event.

Definition

onClickBack(bool forceBack, bool usedMenuButton)

Arguments

bool forceBack	If true, the screen must allow going back

bool usedMenuButton If true, the menu action key/button was used to trigger this event

Return Values

bool True if the event was not used, false if it was used.

setReturnScreenClass

Description

Set the class of the return screen which should be opened when the "back" action is triggered on this screen.

Definition

setReturnScreenClass()

setNextScreenClickSoundMuted

Description

Mute the next click sound. Used to override click sounds for the activate/cancel actions

Definition

setNextScreenClickSoundMuted()

SettingsAdvancedFrame

Description

Advanced graphic settings frame.
--@category GUI
SettingsAdvancedFrame = {}

Parent

InGameMenuFrameElement

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

boolean true if loading was successful else false

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

table self new instance of object

SettingsConsoleFrame

Description

Console settings frame.
--@category GUI
SettingsConsoleFrame = {}

Parent

InGameMenuFrameElement

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

boolean success success

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

table self new instance of object

SettingsControlsFrame

Description

Controls binding settings frame.
-- This is only supposed to be active in the PC version of the game.
--@category GUI

new

Description

Create a new SettingsControlsFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional] Meta table of subclass

Return Values

boolean success success

initialize

Description

Late initialization.

Definition

initialize(table controlsController)

Arguments

table controlsController	New controls controller instance which provides input capture logic for control bindings. This controller must not be shared between components.

Return Values

float	x	x translation
float	y	y translation
float	z	z translation
float	rx	x rotation
float	ry	y rotation
float	rz	z rotation
float	sx	x scale

float	sy	y scale
float	sz	z scale
integer	visibility	visibility

requestClose

Description

Request to close this frame.
If there are pending changes, a saving prompt will be shown and direct closing denied.

Definition

requestClose()

onYesNoSaveControls

Description

Handle saving confirmation on leaving the frame.

Definition

onYesNoSaveControls()

revertChanges

Description

Revert any binding changes since the last save.

Definition

revertChanges()

saveChanges

Description

Save binding changes.

Definition

saveChanges()

updateMenuButtons

Description

Update menu button information.

Definition

updateMenuButtons()

updateHeader

Description

Update header elements' state.

Definition

updateHeader()

setDevicePage

Description

Set the device page.

Definition

setDevicePage(bool toKeyboard)

Arguments

bool toKeyboard	If true, set the page to keyboard, otherwise set it to gamepad

switchDevice
Description

Switch the current binding device (kb / mouse or gamepads).
Switches between tables if possible.

Definition

switchDevice()

setupControlsView
Description

Set up the controls view with input binding tables.

Definition

setupControlsView()

assignDeviceTableData
Description

Assigns table data to all device input binding table elements.
If called after initialization, this will completely rebuild the data. This may be desired or required when new
devices are detected.

Definition

assignDeviceTableData()

Return Values

boolean success success

setRequestButtonUpdateCallback
Description

Set a callback for requesting a button display update (e.g. when switching device pages).

Definition

setRequestButtonUpdateCallback()

Return Values

table instance Instance of object

setControlsMessage
Description

Set the controls notification message.

Definition

setControlsMessage(messageId ID, additionalText [optional], addLine [optional])

Arguments

messageId	ID	of message as defined as a constant in ControlsController.
additionalText	[optional]	More text to add to the message. Must always be a list, since it can be used to hold formatting arguments.
addLine	[optional]	If true, will add the message to the previous text instead of overwriting it.

Return Values

boolean success success

notifyInputGatheringFinished
Description

Notify the screen that input gathering has finished.
Unlocks navigation and closes any open dialogs.

Definition

notifyInputGatheringFinished(madeChange True)

Arguments

madeChange True if any action has been assigned a key/button

Return Values

integer fillType current fill type id

showInputPrompt

Description

Show an input prompt dialog.

Definition

showInputPrompt()

Return Values

integer fillLevel current fill level

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

boolean canBeSold bale can be sold

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

table instance Instance of object

bindControls

Description

Store data bindings for a device category.
The data bindings have an abstract field name (e.g. "action") as key and a table column name as value. The column
name must correspond to the name of a table row template element's name, which is also configured as a column name
in the table configuration.

Definition

bindControls(table bindings, int deviceCategory)

Arguments

table bindings Data binding table {name: column}

int deviceCategory Device category for which these bindings are stored

Return Values

table instance Instance of object

getActionDataFromClickedTableButton

Description

Get the action data associated with a clicked action input button.

Definition

getActionDataFromClickedTableButton(tableButton Clicked, isKeyboard If)

Arguments

tableButton Clicked	action input button element
isKeyboard If	true, the input action device is keyboard and mouse, otherwise it's a gamepad/controller

Return Values

boolean success success

updateDisplay

Description

Update the controls display data.

Definition

updateDisplay()

Return Values

boolean success success

onInputClicked

Description

Handle clicks on input action buttons.

Definition

onInputClicked(int deviceCategory, int bindingId, table actionData)

Arguments

int	deviceCategory	Input action device category
int	bindingId	Binding ID as defined in ControlsController (primary, secondary, tertiary)
table	actionData	Input action data

Return Values

table instance Instance of object

onClickDefaults

Description

Handle "reset to defaults" button activation.

Definition

onClickDefaults()

Return Values

boolean success success

onClickKeyboardHeader

Description

Handle clicking the keyboard controls header.

Definition

onClickKeyboardHeader()

Return Values

boolean success success

onClickGamepadHeader

Description

Handle clicking the gamepad controls header.

Definition

onClickGamepadHeader()

Return Values

float offset offset

SettingsDeviceFrame

Description

Console settings frame.
--@category GUI
SettingsDeviceFrame = {}

Parent

InGameMenuFrameElement

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

boolean isCloserToFront is closer to front

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

boolean canInteract can interact

update

Description

Get the frame's main content element's screen position.

Definition

update()

Return Values

boolean canClose can close silo

updateController

Description

Update controller and mouse sensitivity and deadzone settings values.

Definition

updateController()

Return Values

boolean canOpen can open silo

SettingsGeneralFrame

Description

General game settings frame.
--@category GUI
-- @field inputHelpMode In-game input display help mode option (auto, keyboard, gamepad)

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

float x x world position
float y y world position
float z z world position

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

SettingsModel
Description

Settings menu model.
-- Provides an interface model between game settings and the UI for re-use between several components. The model keeps
a common, transient state until saved. When saving, the settings are applied to the global game settings and written
to the player's configuration file.
--@category GUI

new

Description

Create a new instance.

Definition

new(table gameSettings, int settingsFileHandle, table l10n, table soundMixer)

Arguments

table	gameSettings	GameSettings object which holds the currently active and applied game settings
int	settingsFileHandle	Engine file handle of the player's settings file
table	l10n	I18N reference for localized display string resolution
table	soundMixer	SoundMixer reference for direct application of volume settings

Return Values

table SettingsModel instance

initialize

Description

Initialize model.
Read current configuration settings and populate valid display and configuration option values.

Definition

initialize()

addManagedSettings

Description

Add managed valid settings which receive their initial value from the loaded game settings or the engine.

Definition

addManagedSettings()

addSetting

Description

Add a setting to the model.
Reader and writer functions need to be provided which transform display values (usually indices) to actual setting
values and interact with the current game setting or engine states. Writer function can have side-effects, such as
directly applying values to the engine state or modifying dependent sub-settings.

Definition

addSetting(string gameSettingsKey, function readerFunction, function writerFunction, boolean noRestartRequired)

Arguments

string	gameSettingsKey	Key of the setting in GameSettings
function	readerFunction	Function which reads and processes the setting value identified by the key, signature: function(settingsKey)
function	writerFunction	Function which processes and writes the setting value identified by the key, signature: function(value, settingsKey)
boolean	noRestartRequired	true if no restart is required to apply the setting

setValue

Description

Set a settings value.

Definition

setValue(string settingKey, table value)

Arguments

string	settingKey	Setting key, use one of the values in SettingsModel.SETTING.
table	value	New setting value

getValue

Description

Get a settings value.

Definition

getValue(string settingKey)

Arguments

string	settingKey	Setting key, use one of the values in SettingsModel.SETTING.

Return Values

table	Currently active (changed) settings value

setSettingsFileHandle

Description

Set the settings file handle when it changes (e.g. possible in the gamepad sign-in process).

Definition

setSettingsFileHandle()

refresh

Description

Refresh settings values from their reader functions.
Use this when other components might have changed the settings state and the model needs to reflect those changes
now.

Definition

refresh()

refreshChangedValue

Description

Refresh settings values from their reader functions.
Use this when other components might have changed the settings state and the model needs to reflect those changes
now.

Definition

refreshChangedValue()

hasChanges

Description

Check if any setting has been changed in the model.

Definition

hasChanges()

Return Values

bool True if any setting has been changed, false otherwise

needsRestartToApplyChanges

Description

Check if any setting has been changed in the model.

Definition

needsRestartToApplyChanges()

Return Values

bool True if any setting has been changed, false otherwise

applyChanges

Description

Apply the currently held, transient settings to the game settings.

Definition

applyChanges(bool doSave)

Arguments

bool doSave If true, the changes will also be persisted to storage.

saveChanges

Description

Save the game settings which may have been modified by this model.
This will not apply transient changes but only persist the currently applied game settings.

Definition

saveChanges()

createControlDisplayValues

Description

Populate value and string lists for control elements display.

Definition

createControlDisplayValues()

getBrightnessTexts

Description

Get valid brightness option texts.

Definition

getBrightnessTexts()

getFovYTexts

Description

Get valid FOV Y option texts.

Definition

getFovYTexts()

getUiScaleTexts

Description

Get valid UI scale texts.

Definition

getUiScaleTexts()

getAudioVolumeTexts

Description

Get valid audio volume texts.

Definition

getAudioVolumeTexts()

getCameraSensitivityTexts

Description

Get valid camera sensitivity texts.

Definition

getCameraSensitivityTexts()

getVehicleArmSensitivityTexts

Description

Get valid camera sensitivity texts.

Definition

getVehicleArmSensitivityTexts()

getSteeringBackSpeedTexts

Description

Get valid camera sensitivity texts.

Definition

getSteeringBackSpeedTexts()

getMoneyUnitTexts

Description

Get valid money unit (=currency) texts.

Definition

getMoneyUnitTexts()

getDistanceUnitTexts

Description

Get valid distance unit texts.

Definition

getDistanceUnitTexts()

getTemperatureUnitTexts

Description

Get valid temperature unit texts.

Definition

getTemperatureUnitTexts()

getAreaUnitTexts

Description

Get valid area unit texts.

Definition

getAreaUnitTexts()

getRadioModeTexts

Description

Get valid radio mode texts.

Definition

getRadioModeTexts()

makeDefaultReaderFunction

Description

Build the default reader function.
Reads a value directly from the game settings.

Definition

makeDefaultReaderFunction()

makeDefaultWriterFunction

Description

Build the default writer function.
Writes a value directly to the game settings.

Definition

makeDefaultWriterFunction()

addDirectSetting

Description

Add a setting which can be directly read and written from and to the game settings.

Definition

addDirectSetting()

addPerformanceClassSetting

Description

Add the performance class setting.

Definition

addPerformanceClassSetting()

addWindowModeSetting

Description

Add the window mode setting.

Definition

addWindowModeSetting()

addLanguageSetting

Description

Add the language setting.

Definition

addLanguageSetting()

addMPLanguageSetting

Description

Add the language setting.

Definition

addMPLanguageSetting()

addBrightnessSetting

Description

Add the brightness setting.

Definition

addBrightnessSetting()

addVSyncSetting

Description

Add the vertical synchronization setting.

Definition

addVSyncSetting()

addFovYSetting

Description

Add the vertical field of view setting.

Definition

addFovYSetting()

addUIScaleSetting

Description

Add the UI scale setting.

Definition

addUIScaleSetting()

addCameraSensitivitySetting

Description

Add the camera sensitivity setting.

Definition

addCameraSensitivitySetting()

addVehicleArmSensitivitySetting

Description

Add the vehicleArm sensitivity setting.

Definition

addVehicleArmSensitivitySetting()

addMasterVolumeSetting

Description

Add the master volume setting.

Definition

addMasterVolumeSetting()

addMusicVolumeSetting

Description

Add the music volume setting.

Definition

addMusicVolumeSetting()

addEnvironmentVolumeSetting

Description

Add the environment volume setting.

Definition

addEnvironmentVolumeSetting()

addVehicleVolumeSetting

Description

Add the vehicle volume setting.

Definition

addVehicleVolumeSetting()

addRadioVolumeSetting

Description

Add the radio volume setting.

Definition

addRadioVolumeSetting()

addVolumeGUISetting

Description

Add the gui volume setting.

Definition

addVolumeGUISetting()

addSteeringBackSpeedSetting

Description

Add the steering back speed setting.

Definition

addSteeringBackSpeedSetting()

convertBrightnessToGamma

Description

Convert a settings brightness value to a game gamma value.

Definition

convertBrightnessToGamma()

convertGammaToBrightness

Description

Convert a game gamma value to a settings brightness value.

Definition

convertGammaToBrightness()

getVSyncByIndex

Description

Convert a v-sync option value to boolean.

Definition

getVSyncByIndex()

getVSyncIndex

Description

Convert a v-sync boolean value to an index value.

Definition

getVSyncIndex()

SettingsScreen

Description

Main Menu Settings Screen.
--@category GUI
-- @field pagingElement Paging controller element

updatePages

Description

Update page enabled states.

Definition

updatePages()

registerPage

Description

Register a page frame element in the menu.
This does not add the page to the paging component of the menu.

Definition

registerPage(table pageFrameElement, int position, function enablingPredicateFunction)

Arguments

table	pageFrameElement	Page FrameElement instance
int	position	[optional] Page position index in menu
function	enablingPredicateFunction	[optional] A function which returns the current enabling state of the page at any time. If the function returns true, the page should be enabled. If no argument is given, the page is always enabled.

Return Values

table self instance of class event

unregisterPage

Description

Unregister a page frame element identified by class from the menu.
This does not remove the page from the paging component of the menu or the corresponding page tab from the header.

Definition

unregisterPage(table pageFrameClass)

Arguments

table pageFrameClass FrameElement descendant class of a page which was previously registered

Return Values

table instance	instance of event
bool True	if there was a page of the given class and it was unregistered
table	Unregistered page controller instance or nil
table	Unregistered page root GuiElement instance or nil
table	Unregistered page tab ListElement instance of nil

addPageTab

Description

Add a page tab in the menu header.
Call this synchronously with SettingsScreen:registerPage() to ensure a correct order of pages and tabs.

Definition

addPageTab()

Return Values

table self instance of class event

updatePageTabDisplay

Description

Update page tabs display after any page changes.

Definition

updatePageTabDisplay()

Return Values

table instance instance of event

setPageTabEnabled

Description

Set enabled state of a page tab in the header.

Definition

setPageTabEnabled()

Return Values

table instance Instance of object

rebuildTabList

Description

Rebuild page tab list in order.

Definition

rebuildTabList()

Return Values

table instance Instance of object

setupMenuButtonInfo

Description

Define default properties and retrieval collections for menu buttons.

Definition

setupMenuButtonInfo()

Return Values

table instance Instance of object

onClickPageSelection

Description

Handle activation of page selection.

Definition

onClickPageSelection()

Return Values

table instance Instance of object

onPagePrevious

Description

Handle previous page event.

Definition

onPagePrevious()

Return Values

table instance Instance of object

onPageNext

Description

Handle next page event.

Definition

onPageNext()

Return Values

table instance Instance of object

onPageChange

Description

Handle changing to another menu page.

Definition

onPageChange()

Return Values

boolean allowsAutoDelete allows auto delete

updateButtonsPanel

Description

Update the buttons panel when a given page is visible.

Definition

updateButtonsPanel()

Return Values

boolean hasMoved has moved

clearMenuButtonActions

Description

Clear menu button actions, events and callbacks.

Definition

clearMenuButtonActions()

Return Values

boolean inScope in scope

assignMenuButtonInfo

Description

Assign menu button information to the in-game menu buttons.

Definition

assignMenuButtonInfo()

Return Values

float priority priority

initializePageDisplay

Description

Initialize page display labels from page titles.

Definition

initializePageDisplay()

Return Values

string rigidBodyType rigid body type

onClickOk

Description

Handle menu confirmation input event.

Definition

onClickOk()

Return Values

table instance Instance of object

ShopCategoriesFrame

Description

Shop categories frame for the in-game menu shop.
-- Displays categories/brands or purchasable items in a tile-layout.
--@category GUI

new

Description

Create a new ShopCategoriesFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional] Meta table of subclass

Return Values

boolean success success

initialize

Description

Initialize with categories to be displayed.
Categories must be provided as an array of tables like {id=<id>, iconFilename=<path>, label=<text>}. This will add
a category element per entry to the display list.

Definition

initialize(table categories, function categoryClickedCallback, table headerIconUVs, string headerText)

Arguments

table	categories	Category definitions array
function	categoryClickedCallback	Notification callback for category activation(click/enter), signature: function(selectedId, baseCategoryIconUVs, baseCategoryLabel, clickedCategoryLabel)
table	headerIconUVs	UV coordinates of the header icon to display
string	headerText	Header text to display

Return Values

table instance Instance of object

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

table self new instance of object

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

boolean success success

updateScrollButtons

Description

Update scroll button visibility based on the currently visible list items.

Definition

updateScrollButtons()

Return Values

boolean success success

onClickCategory

Description

Handle a click / button activation on a category.

Definition

onClickCategory()

Return Values

boolean allow allow fill type

onDoubleClickCategory

Description

Handle a double click on a category.

Definition

onDoubleClickCategory()

Return Values

float fillLevel fill level

onCategorySelected

Description

Handle navigation selection of a category element.

Definition

onCategorySelected()

Return Values

float freeCapacity free capacity

onClickLeft

Description

Handle click on left navigation button.

Definition

onClickLeft()

Return Values

table instance Instance of object

onClickRight

Description

Handle click on right navigation button.

Definition

onClickRight()

Return Values

table instance Instance of object

onScroll

Description

Handle a list scrolling event.

Definition

onScroll()

Return Values

table instance Instance of object

ShopConfigScreen

Description

Vehicle Shop and Configuration Screen.
--@category GUI
-- @field shopMoneyBox Layout box for player money information

new

Description

Create the shop configuration screen.

Definition

new(table shopController, table messageCenter, table l10n, table i3dManager, table brandManager, table configurationManager, table vehicleTypeManager, table inputManager, table inputDisplayManager)

Arguments

table shopController	ShopController instance, shared across all shop screens.
table messageCenter	MessageCenter reference for local network UI event handling
table l10n	I18N localization manager reference
table i3dManager	I3DManager reference for asset loading
table brandManager	BrandManager reference
table configurationManager	ConfigurationManager reference
table vehicleTypeManager	VehicleTypeManager reference
table inputManager	InputBinding reference for camera input handling
table inputDisplayManager	InputDisplayManager reference for camera input glyph display

Return Values

boolean isActivateable	is activateable
table	ShopConfigScreen instance

createInputGlyphs

Description

Create input help glyphs.

Definition

createInputGlyphs()

Return Values

table instance Instance of object

createFadeAnimations

Description

Create animations for fading the screen.

Definition

createFadeAnimations()

Return Values

boolean success success

fadeScreen

Description

Setter function for fading animations.

Definition

fadeScreen()

Return Values

table instance Instance of object

createWorkshop

Description

Load the workshop background geometry.
This will not yet link the workshop into the scene graph. That operation is done on the fly on open and close.

Definition

createWorkshop()

Return Values

boolean success success

setWorkshopNode

Description

Set the node ID of the workshop model when loaded.

Definition

setWorkshopNode()

Return Values

boolean success success

createCamera

Description

Create and initialize the workshop camera.

Definition

createCamera()

Return Values

string attributes attributes

string nodes nodes

resetCamera

Description

Reset camera rotation and distance fields.
The changes will be applied on the next update when the camera is repositioned.

Definition

resetCamera()

delete

Description

Delete this screen instance and release resources.

Definition

delete()

Return Values

table instance Instance of object

updateBalanceText

Description

Update the current balance display.

Definition

updateBalanceText()

Return Values

boolean success success

processStoreItemUpkeep

Description

Process and return daily upkeep cost of a store item.

Definition

processStoreItemUpkeep()

Return Values

boolean success success

processStoreItemPowerOutput

Description

Process and return the power ouput of a store item.

Definition

processStoreItemPowerOutput()

Return Values

string attributes attributes

string nodes nodes

processStoreItemFuelCapacity

Description

Process and return the fuel capacity of a store item.

Definition

processStoreItemFuelCapacity()

processStoreItemDefCapacity

Description

Process and return the fuel capacity of a store item.

Definition

processStoreItemDefCapacity()

Return Values

table instance Instance of object

processStoreItemMaxSpeed

Description

Process and retuen the maximum speed of a store item.

Definition

processStoreItemMaxSpeed()

Return Values

boolean success success

processStoreItemCapacity

Description

Process and return the fill capacity and unit of a store item.

Definition

processStoreItemCapacity()

Return Values

table instance Instance of object

processStoreItemWorkingWidth

Description

Process and return the working width of a store item.

Definition

processStoreItemWorkingWidth()

Return Values

processStoreItemWorkingSpeed

Description

Process and return the working speed of a store item.

Definition

processStoreItemWorkingSpeed()

Return Values

processStoreItemPowerNeeded

Description

Process and return the power requirement of a store item.

Definition

processStoreItemPowerNeeded()

Return Values

bool returns true if placement successful

processAttributeData

Description

Process a store item's attribute data.
Changes icons and display texts.

Definition

processAttributeData()

Return Values

boolean success success

getConfigurationCostsAndChanges

Description

Check the base and upgrade cost of a storeItem or current vehicle and whether or not there are any changes.

Definition

getConfigurationCostsAndChanges()

Return Values

string	attributes	attributes
string	nodes	nodes
float	Base	price
float	Upgrade	price
bool	True	if there are changes

updatePriceData

Description

Update price display data.

Definition

updatePriceData()

updateData

Description

Update display data for a store item and/or concrete vehicle.

Definition

updateData()

Return Values

bool return true if doghouse can be activated to fill bowl

overrideOptionFocus

Description

Override an option element's focus behavior.
If the first option is focused and navigation goes up, the option slider should be triggered instead. Vice versa for
the last option and down direction.

Definition

overrideOptionFocus()

Return Values

getDefaultConfigurationColorIndex

Description

Get the default color index of a color configuration.

Definition

getDefaultConfigurationColorIndex(string configName, table configItems, table vehicle)

Arguments

string	configName	Configuration attribute name of the color
table	configItems	Array of configuration attribute options
table	vehicle	[optional] Existing vehicle which is being configured

Return Values

bool true if registration went well

disableUnusedOptions

Description

Disable unused option elements.
During configuration loading, the UI elements are populated in order until no more elements or attributes are
available. If there are more UI elements than attributes, disable those elements now.

Definition

disableUnusedOptions(int currentOptionIndex, int currentColorIndex)

Arguments

int currentOptionIndex Index of first unused configuration option element

int currentColorIndex Index of first unused color picker element

Return Values

bool true if registration went well

updateButtons

Description

Update button states.

Definition

updateButtons()

Return Values

table instance Instance of object

loadCurrentConfiguration

Description

Load the current configuration of a given vehicle store item.

Definition

loadCurrentConfiguration()

Return Values

boolean success success

onVehicleLoaded

Description

Handles asynchronous vehicle loading event.

Definition

onVehicleLoaded()

Return Values

boolean isActivateable is activateable

updateSlider

Description

Update visibility and values of the options slider.

Definition

updateSlider()

Return Values

table instance Instance of object

onSliderChanged

Description

Handle slider change events.
Function must be targeted by screen configuration.

Definition

onSliderChanged()

Return Values

boolean success success

updateDisplay

Description

Update all display data based on a store item and / or vehicle.

Definition

updateDisplay(table storeItem, table vehicle, int scrollValue, bool doNotReload)

Arguments

table	storeItem	Store item reference
table	vehicle	Vehicle reference (for customizing)
int	scrollValue	Current scroll value in cases where there are more options than display elements.
bool	doNotReload	If true, the vehicle is not reloaded (use e.g. when scrolling)

Return Values

boolean success success

setCurrentMission

Description

Set the current mission reference at the start of the mission.

Definition

setCurrentMission()

Return Values

boolean success success

setEconomyManager

Description

Set the economy manager reference at the start of the mission

Definition

setEconomyManager()

Return Values

string attributes attributes

string nodes nodes

loadMapData

Description

Process map data.
Extracts the workshop position if possible.

Definition

loadMapData()

setWorkshopWorldPosition

Description

Set the workshop world position if a map has a custom position defined.

Definition

setWorkshopWorldPosition()

Return Values

boolean showInfo show info

deletePreviewVehicles

Description

Delete all preview vehicles and clear stored array.

Definition

deletePreviewVehicles()

Return Values

float fillDelta real fill delta

setStoreItem

Description

Set the current store item or vehicle to be modified.

Definition

setStoreItem()

Return Values

float isAllowed is allowed

setRequestExitCallback

Description

Set a callback to request exiting this screen and the entire shop at once.

Definition

setRequestExitCallback()

Return Values

table self instance of class event

shouldFocusChange

Description

Override and shadow of GuiElement:shouldFocusChange().
Always allows focus change.

Definition

shouldFocusChange()

Return Values

table instance instance of event

setConfigPrice

Description

Set the price label for a given configuration.

Definition

setConfigPrice()

Return Values

table instance Instance of object

onPickColor

Description

Handle the result of the color picking dialog.

Definition

onPickColor()

Return Values

boolean success success

selectFirstConfig

Description

Select the first configuration option.

Definition

selectFirstConfig()

Return Values

table instance Instance of object

processStoreItemConfigurationSet

Description

Process a configuration set of a store item.

Definition

processStoreItemConfigurationSet(table storeItem, table configSet, table vehicle)

Arguments

table storeItem StoreItem reference

table configSet StoreItem configuration set {name = name, configurations={i=configName}}

table vehicle [optional] Existing Vehicle instance reference

Return Values

boolean success success

processStoreItemSubConfigurationOption

Description

Process a sub-configuration option of a store item.
Sub-configurations are fine-grained configuration sets within vehicle configurations, e.g. wheel brands with distinct
options per brand.

Definition

processStoreItemSubConfigurationOption(table storeItem, string configName, table vehicle)

Arguments

table storeItem StoreItem reference

string configName Vehicle configuration name

table vehicle [optional] Existing Vehicle instance

Return Values

boolean isActiveForInput is active for input

processStoreItemConfigurationOption

Description

Process a configuration option of a store item.

Definition

processStoreItemConfigurationOption(table storeItem, string configName, table configItems, table vehicle, bool isSubConfigOption)

Arguments

table	storeItem	StoreItem reference
string	configName	Vehicle configuration name
table	configItems	Array of configuration items (see StoreItemUtil.addConfigurationItem() for structure)
table	vehicle	[optional] Existing Vehicle instance

bool	isSubConfigOption	[optional, default=false] If true, treats this option as the selection of a sub-configuration set

Return Values

boolean isActiveForSound is active for sound

processStoreItemColorOption

Description

Process a color option of a store item.

Definition

processStoreItemColorOption(table storeItem, string configName, table colorItems, int colorPickerIndex, table vehicle)

Arguments

table	storeItem	StoreItem reference
string	configName	Vehicle configuration name
table	colorItems	Array of color configuration items
int	colorPickerIndex	Element index in this screen of the color picker to populate with the configuration data
table	vehicle	[optional] Existing Vehicle instance

Return Values

boolean

string warningMessage warning message displayed in the shop

processStoreItemConfigurations

Description

Process store item configurations into more convenient data structures for display.
The processed data is stored in instance fields.

Definition

processStoreItemConfigurations()

updateConfigSetOptionElement

Description

Update a configuration option element with the configuration set selection.

Definition

updateConfigSetOptionElement()

Return Values

table self instance of class event

updateConfigOptionElement

Description

Update a configuration option element with a regular configuration option.

Definition

updateConfigOptionElement()

Return Values

table instance instance of event

updateSubConfigOptionElement

Description

Update a configuration option element with a sub-configuration selection.

Definition

updateSubConfigOptionElement()

Return Values

table instance Instance of object

updateConfigOptionsData

Description

Update display data in config options based on the current scroll value.

Definition

updateConfigOptionsData()

Return Values

boolean isInRange is in range

int Number of used config option elements

updateConfigOptionsNavigation

Description

Update config option navigation behavior.
Overrides focus navigation between elements and including scrolling.

Definition

updateConfigOptionsNavigation(int scrollValue, int usedConfigElementCount, int usedColorElementCount)

Arguments

int scrollValue	Current scrolling value
int usedConfigElementCount	Number of config option elements used
int usedColorElementCount	Number of color option elements used

Return Values

boolean isInRange is in range

updateConfigOptionsDisplay

Description

Update configuration display elements based on the currently available configurations and scrolling value.

Definition

updateConfigOptionsDisplay()

Return Values

boolean success success

update

Description

Update the configuration screen's state.

Definition

update()

Return Values

boolean success success

updateCamera

Description

Update camera orientation and position.

Definition

updateCamera()

Return Values

boolean success success

updateDepthOfField

Description

Update the depth of field blur effect.

Definition

updateDepthOfField()

Return Values

boolean success success

draw

Description

Draw the shop config screen.
Override for custom drawing.

Definition

draw()

Return Values

string attributes attributes

string nodes nodes

onOpen

Description

Event function called when opening this screen.

Definition

onOpen()

onClose

Description

Event function called when closing this screen.

Definition

onClose()

Return Values

integer price price

onClickOk

Description

Handle a click or button activation of the "Buy" button.

Definition

onClickOk()

Return Values

onYesNoBuy

Description

Buying confirmation dialog callback.

Definition

onYesNoBuy()

Return Values

integer dailyUpkeep daily up keep

onVehicleBought

Description

Handle vehicle buy event.

Definition

onVehicleBought()

Return Values

integer sellPrice sell price

onClickActivate

Description

Handle a click or button activation of the "Lease" button.

Definition

onClickActivate()

Return Values

{i = {x, y, z, side1x, side1y, side1z, side2x, side2y, side2z}}, all numbers in world space coordinates

onYesNoLease

Description

Handle leasing dialog confirmation.

Definition

onYesNoLease()

Return Values

table capacityAndUnit capacity and unit

onClickBack

Description

Handle back button activation.

Definition

onClickBack()

Return Values

integer incomePerHour income per hour

onCallback

Description

Trigger the callback set in setCallbacks() to notify another component of the current configuration.

Definition

onCallback(bool leaseItem, table storeItem, table configurations, float price)

Arguments

bool	leaseItem	If true, the item should be leased
table	storeItem	StoreItem instance
table	configurations	Item configuration attributes table
float	price	Item configuration price

Return Values

table instance Instance of object

updateInputGlyphs

Description

Update input glyphs when input context changes.

Definition

updateInputGlyphs()

Return Values

boolean success success

registerInputActions

Description

Register required input action events.

Definition

registerInputActions()

Return Values

boolean success success

disableAlternateBindings

Description

Disable alternate bindings for menu navigation.
This will disable some default bindings which interfere with camera controls (e.g. D-Pad on controller). Whenever any
input event is modified, this method must be called again afterwards.

Definition

disableAlternateBindings()

Return Values

string attributes attributes

string nodes nodes

onCameraLeftRight

Description

Handle input for camera left/right.

Definition

onCameraLeftRight()

onCameraUpDown

Description

Handle input for camera up down.

Definition

onCameraUpDown()

Return Values

boolean success success

onCameraZoom

Description

Handle input for camera zoom.

Definition

onCameraZoom()

Return Values

boolean success success

updateInput

Description

Update input for this frame.

Definition

updateInput()

Return Values

string attributes attributes

string nodes nodes

limitXRotation

Description

Limit camera X rotation by the constant maximum and the maximum camera height.

Definition

limitXRotation()

updateInputContext

Description

Update input context and activate suitable events.

Definition

updateInputContext()

Return Values

table capacityAndUnit capacity and unit

ShopController

Description

Shop buying/selling process controller.
-- Handle buying and selling logic, display data and synchronization of shop screens.
--@category GUI

new

Description

Create a new ShopController.

Definition

new(table messageCenter, table l10n, table storeManager, table brandManager, table fillTypeManager)

Arguments

table messageCenter	MessageCenter reference for local network UI event handling
table l10n	I18N reference
table storeManager	StoreManager reference for store item loading
table brandManager	BrandManager reference for brands loading
table fillTypeManager	FillTypeManager reference for fill type data access

Return Values

integer incomePerHour income per hour

subscribeEvents

Description

Subscribe to receive notifications for shop events.

Definition

subscribeEvents()

Return Values

table instance Instance of object

addBrandForDisplay

Description

Add a brand loaded from store items to the display collection.

Definition

addBrandForDisplay()

Return Values

boolean success success

addCategoryForDisplay

Description

Add a category loaded from store items to a suitable display collection.

Definition

addCategoryForDisplay()

Return Values

boolean success success

load

Description

Load brands and items category data.

Definition

load()

Return Values

string attributes attributes

string nodes nodes

setClient

Description

Set the network client on mission loading.

Definition

setClient()

setCurrentMission

Description

Set the current mission after loading.

Definition

setCurrentMission()

Return Values

table capacityAndUnit capacity and unit

setUpdateShopItemsCallback

Description

Set the callback to trigger a shop item list update.

Definition

setUpdateShopItemsCallback()

Return Values

integer incomePerHour income per hour

setUpdateAllItemsCallback

Description

Set the callback to trigger a full shop update.

Definition

setUpdateAllItemsCallback()

Return Values

table instance Instance of object

setSwitchToConfigurationCallback

Description

Set the callback to switch views to the vehicle configuration screen.

Definition

setSwitchToConfigurationCallback()

Return Values

boolean success success

setStartPlacementModeCallback

Description

Set the callback to enter placement mode.

Definition

setStartPlacementModeCallback()

Return Values

boolean success success

filterOwnedItemsByFarmId

Description

Filter a collection of owned items by their farm ID.
The given collection will not be modified.

Definition

filterOwnedItemsByFarmId(table ownedFarmItems, int farmId)

Arguments

table ownedFarmItems Collection of owned farm items (for structure see BaseMission.addItemToList)
int farmId Farm ID

Return Values

boolean success success
table New collection of filtered owned items

setOwnedFarmItems

Description

Set the array of owned items for the garage view of the current player's farm.

Definition

setOwnedFarmItems(table ownedFarmItems, int playerFarmId)

Arguments

table ownedFarmItems Collection of owned farm items (for structure see BaseMission.addItemToList)
int playerFarmId Farm ID

Return Values

table instance Instance of object

setLeasedFarmItems

Description

Set the array of leased items for the garage view of the current player's farm.

Definition

setLeasedFarmItems(table leasedFarmItems, int playerFarmId)

Arguments

table leasedFarmItems Collection of owned farm items (for structure see BaseMission.addItemToList)
int playerFarmId Farm ID

Return Values

boolean success success

update

Description

Update the shop controller state.
Delays buy events by one frame.

Definition

update()

Return Values

boolean success success

makeDisplayItem

Description

Make a display item out of a store item and optionally a concrete object.

Definition

makeDisplayItem()

Return Values

boolean success success

getOwnedItems

Description

Get an array of items owned by the player's farm.

Definition

getOwnedItems()

Return Values

table instance Instance of object

getLeasedVehicles

Description

Get an array of vehicles leased by the player's farm.

Definition

getLeasedVehicles()

Return Values

boolean success success

getOwnedFarmItems

Description

Get the collection of owned farm items for the current player's farm.

Definition

getOwnedFarmItems()

Return Values

boolean isActivateable is activateable

getLeasedFarmItems

Description

Get the collection of leased farm items for the current player's farm.

Definition

getLeasedFarmItems()

Return Values

table instance Instance of object

getBrands

Description

Get an array of known brands structured for display.

Definition

getBrands()

Return Values

bool	true	if ok
table	Array	of brands in the form of {i={id = brand.index, iconFilename = brand.image, label = brand.title}}

getVehicleCategories

Description

Get an array of known vehicle categories structured for display.

Definition

getVehicleCategories()

Return Values

table returns the graphics root node

table	Array	of vehicle categories in the form of {i={id = category.index, iconFilename = category.image, label = category.title}}

getToolCategories

Description

Get an array of tool categories structured for display.

Definition

getToolCategories()

Return Values

bool	true	if input is allowed.
table	Array	of tool categories in the form of {i={id = category.index, iconFilename = category.image, label = category.title}}

getObjectCategories

Description

Get an array of object categories structured for display.

Definition

getObjectCategories()

Return Values

float	posX	x position of player
float	posY	y position of player
float	posZ	z position of player
float	graphicsRotY	rotation of the player
table	Array	of object categories in the form of {i={id = category.index, iconFilename = category.image, label = category.title}}

getPlaceableCategories

Description

Get an array of placeable categories structured for display.

Definition

getPlaceableCategories()

Return Values

table	Array	of placeable categories in the form of {i={id = category.index, iconFilename = category.image, label = category.title}}

getItemsByBrand

Description

Get store items for a given brand ID.

Definition

getItemsByBrand()

getItemsByCategory

Description

Get shop display items for a given category name.

Definition

getItemsByCategory()

canBeBought

Description

Check if a store item can be bought at a given price and considering the current game state.

Definition

canBeBought()

Return Values

bool	returns true if distance to player root node is lower than clip distance

buy

Description

Buy an item.

Definition

buy(table storeItem, bool outsideBuy)

Arguments

table storeItem	StoreItem to buy

bool outsideBuy If true, means that an item is "bought" without anyone paying for it, e.g. as an achievement bonus

Return Values

float returns calculated priority

buyVehicle

Description

Buy a vehicle.

Definition

buyVehicle()

Return Values

string that will be displayed on console

onYesNoBuyObject

Description

Buy object confirmation dialog callback.

Definition

onYesNoBuyObject()

Return Values

string that will be displayed on console

buyObject

Description

Buy an object.

Definition

buyObject()

Return Values

string that will be displayed on console

buyHandTool

Description

Buy a hand tool.

Definition

buyHandTool()

Return Values

bool returns true object that was hit is valid

sell

Description

Sell an item.

Definition

sell(table item)

Arguments

table item Item to sell

Return Values

bool returns true object that was hit is valid

sellWarningInfoClickOk

Description

Show dialog for selling confirmation.

Definition

sellWarningInfoClickOk()

Return Values

table returns the handtool

onSellCallback

Description

Selling confirmation dialog callback.

Definition

onSellCallback()

Return Values

string Filename of currently equipped hand tool or empty string if no hand tool is equipped

onSellItem

Description

Event handling function which is called when selling an item has been confirmed.

Definition

onSellItem()

Return Values

always returns false

sellPlaceable

Description

Sell a placeable object.
Activates placement mode if the player owns more than one instance of the given placeable.

Definition

sellPlaceable()

Return Values

string that will be displayed on console

sellHandTool

Description

Sell a hand tool.

Definition

sellHandTool()

Return Values

table instance instance of object

sellVehicle

Description

Sell a vehicle.

Definition

sellVehicle()

Return Values

boolean true if loading was successful else false

setConfigurations

Description

Set a buying configuration is used when sending buy events during update().

Definition

setConfigurations()

Return Values

boolean true if loading was successful else false

finalizeBuy

Description

Finalize a buying (or leasing) process and trigger events according to the requested item type during update().

Definition

finalizeBuy()

Return Values

boolean true if added successful else false

onHandToolSellEvent

Description

Event callback on local SellHandToolEvent execution.

Definition

onHandToolSellEvent()

Return Values

table player the player object

onHandToolSold

Description

Event callback on successful SellHandToolEvent.

Definition

onHandToolSold()

Return Values

table player the player object

onHandToolSellFailed

Description

Event callback on failed SellHandToolEvent.

Definition

onHandToolSellFailed()

Return Values

integer number number of models

onVehicleBuyEvent

Description

Event callback on local BuyVehicleEvent execution.

Definition

onVehicleBuyEvent()

Return Values

table instance Instance of object

onVehicleBought
Description

Event callback on successful BuyVehicleEvent.

Definition

onVehicleBought()

Return Values

table instance Instance of object

onVehicleBuyFailed
Description

Event callback on failed BuyVehicleEvent.

Definition

onVehicleBuyFailed()

Return Values

table instance Instance of object

onObjectBuyEvent
Description

Event callback on local BuyObjectEvent execution.

Definition

onObjectBuyEvent()

Return Values

table instance Instance of PlayerSetFarmAnswerEvent

onObjectBought
Description

Event callback on successful BuyObjectEvent.

Definition

onObjectBought()

Return Values

table instance Instance of object

onObjectBuyFailed
Description

Event callback on failed BuyObjectEvent.

Definition

onObjectBuyFailed()

Return Values

table instance Instance of object

onHandToolBuyEvent
Description

Event callback on local BuyHandToolEvent execution.

Definition

onHandToolBuyEvent()

Return Values

table instance Instance of object

onHandToolBought
Description

Event callback on successful BuyHandToolEvent.

Definition

onHandToolBought()

Return Values

table instance Instance of object

onHandToolBuyFailed
Description

Event callback on failed BuyHandToolEvent.

Definition

onHandToolBuyFailed()

Return Values

table instance instance of object

onVehicleSellEvent
Description

Event callback on local SellVehicleEvent execution.

Definition

onVehicleSellEvent()

Return Values

bool returns true if player can interact with object

onVehicleSold
Description

Event callback on successful SellVehicleEvent.

Definition

onVehicleSold()

Return Values

table instance instance of object

onVehicleSellFailed
Description

Event callback on failed SellVehicleEvent.

Definition

onVehicleSellFailed()

Return Values

bool returns true if player can feed an animal

onPlaceableSellEvent
Description

Event callback on local SellPlaceableEvent execution.

Definition

onPlaceableSellEvent()

Return Values

table instance instance of object

onPlaceableSold

Description

Event callback on successful SellPlaceableEvent.

Definition

onPlaceableSold()

Return Values

bool returns true if player can interact with an animal

onPlaceableSellFailed

Description

Event callback on failed SellPlaceableEvent.

Definition

onPlaceableSellFailed()

Return Values

table instance instance of object

onBoughtCallback

Description

Buying process termination (success or failure) information dialog callback.

Definition

onBoughtCallback()

Return Values

bool returns true if player can pet an animal

onSoldCallback

Description

Selling process termination (success or failure) information dialog callback.

Definition

onSoldCallback()

Return Values

table instance instance of object

brandSortFunction

Description

Brand sorting function.

Definition

brandSortFunction()

Return Values

bool returns true if player can ride an animal

categorySortFunction

Description

Category sorting function.
Uses the category order index which was set when loading. This restores category order to the XML definition.

Definition

categorySortFunction()

Return Values

table instance instance of object

displayItemSortFunction

Description

Display item sorting function (for owned and leased objects in the garage).
First sorts by the category order index provided to the display item, then by price within the same category. As a
final tie-breaker for vehicles owned at mission start, the object ID is compared to guarantee a sort order
resolution.

Definition

displayItemSortFunction()

Return Values

table instance instance of object

ShopDisplayItem

Description

Display data class for shop items.
-- Both store items and concrete items can be represented by this class to be displayed in the shop for buying or
selling.

new

Description

Create a new ShopDisplayItem instance.

Definition

new(table storeItem, table concreteItem, table attributeIconProfiles, table attributeValues, table fillTypeFilenames, table seedTypeFilenames, string functionText, int orderValue)

Arguments

table	storeItem	Store item definition table, see StoreManager.lua.
table	concreteItem	Concrete game object of the given store item type or ShopDisplayItem.NO_CONCRETE_ITEM.
table	attributeIconProfiles	Array of UI profiles for attribute icons
table	attributeValues	Array of value display strings for attributes which correspond to the icons
table	fillTypeFilenames	Array of fill type icon file names
table	seedTypeFilenames	Array of seed fill type icon file names
string	functionText	Description text of the item's function
int	orderValue	Display item category order value, lower is higher order

ShopItemsFrame

Description

Shop items frame for the in-game menu shop.
-- Displays purchasable items of a common category in a horizontal list layout.
--@category GUI

new

Description

Create a new ShopItemsFrame instance.

Definition

new(table subclass_mt)

Arguments

table subclass_mt [optional] Meta table of subclass

Return Values

bool true if player can crouch

setItemClickCallback

Description

Set the callback to use when an item is activated (for buying or selling).

Definition

setItemClickCallback()

Return Values

table instance instance of object

setItemSelectCallback

Description

Set the callback to use when an item is selected in the view.

Definition

setItemSelectCallback()

Return Values

bool true if player can idle

setHeader

Description

Set header icon and text.

Definition

setHeader()

Return Values

table instance instance of object

setCategory

Description

Set the category to display.

Definition

setCategory()

Return Values

bool true if player can idle

setShowBalance

Description

Set the balance elements' visibility.

Definition

setShowBalance()

Return Values

table instance instance of object

setShowNavigation

Description

Set the navigation header's visibility.

Definition

setShowNavigation()

Return Values

bool true if state is available

setCurrentBalance

Description

Set the current money balance display.

Definition

setCurrentBalance(float balance, string balanceString)

Arguments

float balance Current balance of the current player

string balanceString Properly formatted money string

Return Values

table instance instance of object

setSlotsUsage

Description

Set the current slot usage display for consoles.

Definition

setSlotsUsage()

Return Values

bool true if player can idle

setDisplayItems

Description

Set an ordered array of ShopDisplayItem instances to display in this frame.

Definition

setDisplayItems()

Return Values

table instance instance of object

updateScrollButtons

Description

Update scroll button visibility based on the currently visible list items.

Definition

updateScrollButtons()

Return Values

bool true if player can jump

getStoreItemDisplayPrice

Description

Get a store items price (buy or sell value) for displaying.

Definition

getStoreItemDisplayPrice()

Return Values

table instance instance of object

assignItemFillTypesData

Description

Assign fill types data to detail box.
Creates icons for fill types.

Definition

assignItemFillTypesData(string baseIconProfile, table iconFilenames, int attributeIndex)

Arguments

string	baseIconProfile	UI profile for the fill type base icon
table	iconFilenames	Array of filenames of fill type icons
int	attributeIndex	Index of attribute slot to use

Return Values

table	player state	
int	Next	usable attribute slot index after these fill types

assignItemTextData

Description

Assign text data to detail box.

Definition

assignItemTextData(table displayItem)

Arguments

table	displayItem	ShopDisplayItem which holds item attribute data

Return Values

bool	true	if player state is available
int	Number of attributes used for text data	

assignItemAttributeData

Description

Assign display data of a selected ShopDisplayItem to the attribute elements.

Definition

assignItemAttributeData(table displayItem)

Arguments

table	displayItem	ShopDisplayItem which holds item attribute data

Return Values

bool	true	if player state is active

getMainElementSize

Description

Get the frame's main content element's screen size.

Definition

getMainElementSize()

Return Values

table	instance	instance of object

getMainElementPosition

Description

Get the frame's main content element's screen position.

Definition

getMainElementPosition()

Return Values

bool true if player can idle

onClickItem

Description

Handle a click / button activation on an item.

Definition

onClickItem()

Return Values

table instance instance of object

onDoubleClickItem

Description

Handle a double-click on an item.

Definition

onDoubleClickItem()

Return Values

bool true if player can run

onClickLeft

Description

Handle click on left navigation button.

Definition

onClickLeft()

Return Values

bool true if player can run

onClickRight

Description

Handle click on right navigation button.

Definition

onClickRight()

Return Values

table instance instance of object

onScroll

Description

Handle a list scroll event.

Definition

onScroll()

Return Values

bool true if player can swim

onItemSelected

Description

Handle selection of an item.

Definition

onItemSelected()

Return Values

table instance instance of object

SideNotification

Description

HUD side notification element.
-- Displays notifications issued by other game components at the side of the screen.
--@category GUI

new

Description

Create a new SideNotification.

Definition

new(string hudAtlasPath)

Arguments

string hudAtlasPath Path to the HUD atlas texture

Return Values

table instance instance of object
table SideNotification instance

addNotification

Description

Add a notification message to display.

Definition

addNotification(string text, table color, int displayDuration)

Arguments

string text Display message text
table color Color array as {r, g, b, a}
int displayDuration Display duration of message in milliseconds

Return Values

table instance instance of object

update

Description

Update notifications state.

Definition

update()

Return Values

table self instance

draw

Description

Draw the notifications.

Definition

draw()

Return Values

table instance instance of object

getBackgroundPosition
Description

Get this element's base background position.

Definition

getBackgroundPosition(float uiScale)

Arguments

float uiScale Current UI scale factor

Return Values

boolean true if loading was successful else false

setScale
Description

Set uniform UI scale.

Definition

setScale()

Return Values

boolean true if loading was successful else false

updateSizeAndPositions
Description

Update sizes and positions of this elements and its children.

Definition

updateSizeAndPositions()

Return Values

table animals list all animals

storeScaledValues
Description

Store scaled positioning, size and offset values.

Definition

storeScaledValues()

Return Values

table animal the animal object

createBackground
Description

Create the background overlay.

Definition

createBackground()

Return Values

table animal the animal object

createComponents
Description

Create required display components.

Definition

createComponents()

Return Values

table animal the animal object

SiloExtensionPlaceable

Description

When trying to sell an extension that is required to store all fills, show a warning before selling the extension and the contents.

load

Description

Load silo extension

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position
float	y	z world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Return Values

boolean success success

Code

```
46  function SiloExtensionPlaceable:load(xmlFilename, x,y,z, rx,ry,rz, initRandom)
47  if not SiloExtensionPlaceable:superClass().load(self, xmlFilename, x,y,z, rx,ry,rz, initRandom) then
48  return false
49  end
50
51  local xmlFile = loadXMLFile("TempXML", xmlFilename)
52
53  local storageKey = "placeable.storage"
54  if hasXMLProperty(xmlFile, storageKey) then
55  local storageNode = I3DUtil.indexToObject(self.nodeId, getXMLString(xmlFile, storageKey.."#node"))
56  if storageNode ~= nil then
57  self.storage = Storage:new(self.isServer, self.isClient)
58  self.storage:load(storageNode, xmlFile, storageKey)
59  else
60  g_logManager:xmlWarning(xmlFilename, "Missing 'node' for storage '%s'!", storageKey)
```

```
61  end
62  else
63  g_logManager:xmlWarning(xmlFilename, "Missing 'storage' for
    siloExtension '%s'!", xmlFilename)
64  end
65
66  delete(xmlFile)
67
68  return true
69  end
```

finalizePlacement

Description

Called if placeable is placed

Definition

finalizePlacement()

Code

```
73  function SiloExtensionPlaceable:finalizePlacement()
74  SiloExtensionPlaceable:superClass().finalizePlacement(self)
75
76  local lastFoundUnloadingStations =
    g_currentMission:getStorageTargetsInRange(g_currentMission.unloadingStati
    self.storage)
77  local lastFoundLoadingStations =
    g_currentMission:getStorageTargetsInRange(g_currentMission.loadingStation
    self.storage)
78
79  self.storage:setOwnerFarmId(self:getOwnerFarmId(), true)
80  g_currentMission:addStorage(self.storage)
81  self.storage:register(true)
82
83  g_currentMission:addStorageToUnloadingStations(self.storage,
    lastFoundUnloadingStations, false)
84  g_currentMission:addStorageToLoadingStations(self.storage,
    lastFoundLoadingStations, false)
85  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId	stream ID
table	connection connection

Code

```
91  function SiloExtensionPlaceable:readStream(streamId, connection)
92  SiloExtensionPlaceable:superClass().readStream(self, streamId,
    connection)
93  if connection:getIsServer() then
94  local storageId = NetworkUtil.readNodeObjectId(streamId)
95  self.storage:readStream(streamId, connection)
96  g_client:finishRegisterObject(self.storage, storageId)
97  end
98  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID
table connection connection

Code

```
104 function SiloExtensionPlaceable:writeStream(streamId, connection)
105 SiloExtensionPlaceable:superClass().writeStream(self, streamId,
    connection)
106 if not connection:getIsServer() then
107 NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(self.storage))
108 self.storage:writeStream(streamId, connection)
109 g_server:registerObjectInStream(connection, self.storage)
110 end
111 end
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key, boolean resetVehicles)

Arguments

integer xmlFile id of xml object
string key key
boolean resetVehicles reset vehicles

Return Values

boolean success success

Code

```
127 function SiloExtensionPlaceable:loadFromXMLFile(xmlFile, key,
    resetVehicles)
```

```
128  if not SiloExtensionPlaceable:superClass().loadFromXMLFile(self, xmlFile, key, resetVehicles) then
129  return false
130  end
131
132  if not self.storage:loadFromXMLFile(xmlFile, key..".storage") then
133  return false
134  end
135
136  return true
137  end
```

saveToXMLFile

Description

Get save attributes and nodes

Definition

saveToXMLFile(string nodeIdent)

Arguments

string nodeIdent node ident

Return Values

string attributes attributes

string nodes nodes

Code

```
144  function SiloExtensionPlaceable:saveToXMLFile(xmlFile, key, usedModNames)
145  SiloExtensionPlaceable:superClass().saveToXMLFile(self, xmlFile, key, usedModNames)
146
147  self.storage:saveToXMLFile(xmlFile, key..".storage", usedModNames)
148  end
```

canBeSold

Description

show a warning before selling the extension and the contents.

Definition

canBeSold()

SliderElement

Description

Draggable viewport slider element.
-- If #hasButtons is true or not present, this element requires 2 ButtonElement instances as the first children, which
provide another way to scroll in addition to clicking the bar or dragging the handle.
-- Used layers: "image" for a background image, "sliderImage" for a slider handle background image.

-- Implicit callback:
onSliderValueChanged(element, newValue) is called on all children and the target data element when the slider value
changes, if those elements have the method defined.
--@category GUI
--@xmlConfig GuiElement#direction string [optional] Slider orientation, defaults to "x". Valid values: "x" for horizontal, "y" for vertical.

updateSliderButtons

Description

Update the disabled-ness of the slider buttons depending on current state

Definition

updateSliderButtons()

canReceiveFocus

Description

Determine if this SliderElement can receive focus.

Definition

canReceiveFocus()

SpeakerDisplay
Description

Player speaker display for consoles.
-- Displays currently speaking players for consoles only. This display is used in place of the chat window.
--@category GUI

new

Description

Create a new SpeakerDisplay.

Definition

new(string hudAtlasPath, table ingameMap)

Arguments

string	hudAtlasPath	Path to the HUD atlas texture.
table	ingameMap	IngameMap reference for positioning

Return Values

string	animal type
table	SpeakerDisplay instance

setUsers

Description

Set the references to the currently connected users.

Definition

setUsers()

Return Values

table

onMenuVisibilityChange

Description

Handle menu visibility state change.

Definition

onMenuVisibilityChange()

Return Values

table store information [shopItemName, canBeBought, imageFilename, price]

updateSpeakingState

Description

Update current speaking state for all connected users.

Definition

updateSpeakingState()

Return Values

table instance instance of object

updateVisibility

Description

Update visibility states based on active speakers.

Definition

updateVisibility()

Return Values

boolean true if loading was successful else false

update

Description

Update the display state each frame.

Definition

update()

Return Values

table returns a food group or nil if nothing is found

setScale

Description

Set this element's UI scale.

Definition

setScale()

Return Values

table instance Instance of object

getBackgroundPosition

Description

Get this element's base background position.

Definition

getBackgroundPosition(float uiScale)

Arguments

float uiScale Current UI scale factor

Return Values

bool true if load is successful

storeScaledValues

Description

Store scaled positioning, size and offset values.

Definition

storeScaledValues()

Return Values

integer id of the animal group

createBackground

Description

Create the background overlay.

Definition

createBackground()

Return Values

integer number of trees found

createSpeakerLine

Description

Create a line for an active speaker.

Definition

createSpeakerLine()

Return Values

bool true to continue counting trees

createComponents

Description

Create required display components.

Definition

createComponents()

Return Values

table instance instance of object

SpeedMeterDisplay

Description

Vehicle HUD speed meter display element.
-- Displays gauges for current speed, fuel level and vehicle wear / damage. Also shows operating time, textual speed
display and cruise control state.
--@category GUI

new

Description

Create a new SpeedMeterDisplay instance.

Definition

new(string hudAtlasPath)

Arguments

string hudAtlasPath Path to the HUD texture atlas

Return Values

boolean success success

getBasePosition

Description

Get this element's base position as a reference for other component's positioning.

Definition

getBasePosition()

Return Values

string attributes attributes

string nodes nodes

createComponents

Description

Create display components for the speed meter.
Components are created with an implicit scale of 1. Scaling should only ever happen after initialization.

Definition

createComponents(string hudAtlasPath)

Arguments

string hudAtlasPath Path to the HUD texture atlas

delete

Description

Delete this instance and all managed components.

Definition

delete()

Return Values

boolean inScope in scope

setVehicle

Description

Set the current vehicle which provides the data for the speed meter.

Definition

setVehicle(table vehicle)

Arguments

table vehicle Vehicle reference

Return Values

float priority priority

update

Description

Update the state of the speed meter.

Definition

update()

Return Values

float closest distance squared in m

updateOperatingTime

Description

Update operating time drawing parameters.

Definition

updateOperatingTime()

Return Values

integer returns number of animals

updateCruiseControl

Description

Update cruise control drawing parameters.

Definition

updateCruiseControl()

Return Values

bool returns true if there is water

updateGaugeIndicator

Description

Update a gauge indicator needle.

Definition

updateGaugeIndicator(table Indicator, float radiusX, float radiusY, float rotation)

Arguments

table	Indicator	HUDElement
float	radiusX	Radius X component of distance to the gauge center
float	radiusY	Radius Y component of distance to the gauge center
float	rotation	Rotation angle of the indicator in radians

Return Values

float	separateForceX	x component of steering force. default is 0.
float	separateForceY	y component of steering force. default is 0.
float	separateForceZ	z component of steering force. default is 0.

updateGaugeFillSegments

Description

Update gauge fill segments.

Definition

updateGaugeFillSegments(table fillSegments, float gaugeValue)

Arguments

table	fillSegments	Array of segment elements
float	gaugeValue	Current gauge indication value [0, 1]

updateGaugePartialSegments

Description

Update partial gauge segment elements.

Definition

updateGaugePartialSegments(table partialSegments, float indicatorRotation, int rotationDirection, float gaugeRadiusX, float gaugeRadiusY, float gaugeMinAngle, flaot fullSegmentAngle, float detailSegmentAngle, bool isPartialOnly)

Arguments

table	partialSegments	Array of segment elements
float	indicatorRotation	Current indicator rotation angle in radians
int	rotationDirection	Direction value for the rotation, 1 is counter-clockwise, -1 is clockwise

float	gaugeRadiusX	Radius X component of distance to gauge center in screen space
float	gaugeRadiusY	Radius Y component of distance to gauge center in screen space
float	gaugeMinAngle	Angle of gauge zero position in radians
flaot	fullSegmentAngle	Angle which spans a filled segment in radians
float	detailSegmentAngle	Smallest angle which can be covered by a partial segment in radians
bool	isPartialOnly	If true, the current gauge has no fill segments and uses only partial segments to display values

updateSpeedGauge

Description

Update the speed gauge state.

Definition

updateSpeedGauge()

Return Values

float	wanderForceX	x component of steering force. default is 0.
float	wanderForceY	y component of steering force. default is 0.
float	wanderForceZ	z component of steering force. default is 0.
float	wanderAngle	new wandering angle in rad. default is 0.

updateDamageGauge

Description

Update the damage gauge state.

Definition

updateDamageGauge()

getVehicleFuelLevelAndCapacity

Description

Get fuel level and capacity of a vehicle.

Definition

getVehicleFuelLevelAndCapacity()

updateFuelGauge

Description

Update the fuel gauge state.

Definition

updateFuelGauge()

onAnimateVisibilityFinished

Description

Override of HUDDisplayElement.
Also updates the scaled values which are relative to the current position.

Definition

onAnimateVisibilityFinished()

Return Values

float	seekForceX	x component of steering force. default is 0.
float	seekForceY	y component of steering force. default is 0.
float	seekForceZ	z component of steering force. default is 0.

draw
Description

Draw the speed meter.

Definition

draw()

drawOperatingTimeText
Description

Draw vehicle operating time if set.

Definition

drawOperatingTimeText(table vehicle)

Arguments

table vehicle Current vehicle

drawCruiseControlText
Description

Draw the text portion of the cruise control element.

Definition

drawCruiseControlText()

Return Values

bool true is any player is close to ground

drawSpeedText
Description

Draw the current speed in text.

Definition

drawSpeedText()

Return Values

bool true is any player is close

fadeFuelGauge
Description

Fade the fuel gauge elements.

Definition

fadeFuelGauge()

Return Values

table instance Instance of object

animateFuelGaugeToggle
Description

Animate (de-)activation of the fuel gauge.

Definition

animateFuelGaugeToggle()

Return Values

bool returns true if load is successful

fadeDamageGauge
Description

Fade the damage gauge elements.

Definition

fadeDamageGauge()

Return Values

float x world position. default is 0

float x world position. default is 0

float x world position. default is 0

animateDamageGaugeToggle

Description

Animate (de-)activation of the damage gauge.

Definition

animateDamageGaugeToggle()

setScale

Description

Set the speed meter scale.
Overrides HUDElement.setScale().

Definition

setScale(float uiScale)

Arguments

float uiScale UI scale factor, applied to both width and height dimensions

storeGaugeCenterPosition

Description

Calculate and store the gauge center position, including the current UI scale.

Definition

storeGaugeCenterPosition(float baseX, float baseY)

Arguments

float baseX Gauge background element X position in screen space

float baseY Gauge background element Y position in screen space

Return Values

bool returns true if animals are spawned

storeScaledValues

Description

Calculate and store scaling values based on the current UI scale.

Definition

storeScaledValues(float baseX, float baseY)

Arguments

float baseX Gauge background element X position in screen space

float baseY Gauge background element Y position in screen space

Return Values

bool returns true if all tests are validated

getBackgroundPosition

Description

Get the position of the background element, which provides the SpeedMeterDisplay's absolute position.

Definition

getBackgroundPosition(float backgroundWidth)

Arguments

float backgroundWidth Scaled background width in pixels

Return Values

integer number of trees found

createBackground

Description

Create the background overlay for the speed meter.

Definition

createBackground(string hudAtlasPath)

Arguments

string hudAtlasPath Path to the HUD texture atlas

Return Values

bool true to continue counting trees

table Overlay instance

createGaugeBackground

Description

Create the gauge background.

Definition

createGaugeBackground()

Return Values

bool return true is on field

createSideGaugeBackground

Description

Create a side gauge background element.

Definition

createSideGaugeBackground()

Return Values

bool returns true if there is water

createGaugeIconElements

Description

Create gauge icons.

Definition

createGaugeIconElements()

Return Values

bool returns true if current time in hours range

createHorizontalSeparator

Description

Create the horizontal separator HUD element.

Definition

createHorizontalSeparator()

Return Values

integer returns the number of animals to spawn

createCruiseControlElement

Description

Create the cruise control HUD element.

Definition

createCruiseControlElement()

Return Values

bool returns true if animals are spawned

createOperatingTimeElement

Description

Create the operating time HUD element.

Definition

createOperatingTimeElement()

Return Values

string that will be displayed on console

createIndicator

Description

Create a movable indicator needle element.

Definition

createIndicator(string hudAtlasPath, table size, table uvs, table color, table pivot)

Arguments

string	hudAtlasPath	Path to the HUD texture atlas
table	size	Pixel size of the indicator as an array {width, height}
table	uvs	UV coordinates of the indicator texture part as an array {x, y, width, height}
table	color	Color of the indicator as an RGBA array
table	pivot	Rotation pivot offset of the indicator as an array {x, y}

Return Values

string	that will be displayed on console
table	HUDElement instance

createGaugeFillElements

Description

Create fill elements for large segments of a gauge.

Definition

createGaugeFillElements(string hudAtlasPath, float baseX, float baseY, float gaugeStartAngle, float gaugeEndAngle, float fillSegmentAngle, table radius, table segmentSize, table segmentPivot, table segmentUVs, table segmentColor)

Arguments

string	hudAtlasPath	Path to the HUD texture atlas
float	baseX	Base X position of the gauge element in screen space
float	baseY	Base Y position of the gauge element in screen space
float	gaugeStartAngle	Gauge starting angle position

float	gaugeEndAngle	Gauge ending angle position
float	fillSegmentAngle	Angle spanned by a fill segment
table	radius	Radius of the gauge as an array {x, y}
table	segmentSize	Pixel size of a fill segment as an array {width, height}
table	segmentPivot	Rotation pivot offset of a fill segment as an array {x, y}
table	segmentUVs	UV coordinates of a fill segment as an array {x, y, width, height}
table	segmentColor	Color of a fill segment as an RGBA array

Return Values

string that will be displayed on console

createGaugePartialElements

Description

Create gauge elements which span the spaces inbetween fill segments and their indicator needle.

Definition

createGaugePartialElements(string hudAtlasPath, float baseX, float baseY, table fullSegmentSize, table segmentPivot, table segmentColor, table gaugeSegmentUVs)

Arguments

string	hudAtlasPath	Path to the HUD texture atlas
float	baseX	Base X position of the gauge element in screen space
float	baseY	Base Y position of the gauge element in screen space
table	fullSegmentSize	Pixel size of a full segment as an array {width, height}
table	segmentPivot	Rotation pivot offset of a segment as an array {x, y}
table	segmentColor	Color of a segment as an RGBA array
table	gaugeSegmentUVs	Array of segment UV arrays ordered by size {sizeIndex={x, y, width, height}}

Return Values

string that will be displayed on console

createSpeedGaugeIndicator

Description

Create the indicator needle for the speed gauge.

Definition

createSpeedGaugeIndicator()

Return Values

createSpeedGaugeElements

Description

Create the segment elements for the speed gauge.

Definition

createSpeedGaugeElements()

Return Values

createDamageGaugeIndicator

Description

Create the indicator needle for the damage gauge.

Definition

createDamageGaugeIndicator()

Return Values

string that will be displayed on console

createDamageGaugeElements

Description

Create the gauge segments for the damage gauge.

Definition

createDamageGaugeElements()

Return Values

string that will be displayed on console

createFuelGaugeIndicator

Description

Create the indicator needle for the fuel gauge.

Definition

createFuelGaugeIndicator()

Return Values

table instance instance of object

createFuelGaugeElements

Description

Create the segment elements for the fuel gauge.

Definition

createFuelGaugeElements()

Return Values

table instance instance of object

StartMissionInfo

Description

Structured data for mission starts.
-- This serves as a data transfer object between GUI screens when setting up a game.

new

Description

Create a new StartMissionInfo instance.

Definition

new()

Return Values

boolean detachAllowed detach is allowed

reset

Description

Reset all information for a new setup.

Definition

reset()

Return Values

float wearMultiplier current wear multiplier

StartupScreen

Description

Game Startup Screen.
-- Shows splash and intro videos, leads on to main menu.

exposeControlsAsFields

Description

Duck-typed dummy function to make this class behave like a ScreenElement on initialization.
See Gui:loadGui() for the processing part which requires this.

Definition

exposeControlsAsFields()

Return Values

bool true if player can idle

StoreManager

Description

Category type for grouping (e.g. in the shop UI)

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
26  function StoreManager:new(customMt)
27  local self = AbstractManager:new(customMt or StoreManager_mt)
28
29  return self
30  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
34  function StoreManager:initDataStructures()
35  self.numOfCategories = 0
36  self.categories = {}
37  self.items = {}
38  self.xmlFilenameToItem = {}
39  self.modStoreItems = {}
40
41  self.specTypes = {}
42  self.nameToSpecType = {}
43  end
```

loadMapData

Description

Load manager data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
48  function StoreManager:loadMapData(xmlFile, missionInfo,
    baseDirectory)
49  StoreManager:superClass().loadMapData(self)
50
51  -- local all store categories
52  local categoryXMLFile = loadXMLFile("storeCategoriesXML",
    "dataS/storeCategories.xml")
53  local i = 0
54  while true do
55  local baseXMLName = string.format("categories.category(%d)", i)
56
57  if not hasXMLProperty(categoryXMLFile, baseXMLName) then
58  break
59  end
60  local name = getXMLString(categoryXMLFile, baseXMLName .. "#name")
61  local title = getXMLString(categoryXMLFile, baseXMLName ..
    "#title")
62  local imageFilename = getXMLString(categoryXMLFile, baseXMLName ..
    "#image")
63  local type = getXMLString(categoryXMLFile, baseXMLName .. "#type")
64
65  if title ~= nil and title:sub(1, 6) == "$l10n_" then
66  title = g_i18n:getText(title:sub(7))
67  end
68
69  self:addCategory(name, title, imageFilename, type, "")
70
71  i = i + 1
72  end
73  delete(categoryXMLFile)
74
75  -- now load all storeitems
76
77  local storeItemsFilename = "dataS/storeItems.xml"
```

```
78  if g_isPresentationVersionSpecialStore then
79  storeItemsFilename = "dataS/storeItems_presentationVersion.xml"
80  end
81
82  self:loadItemsFromXML(storeItemsFilename)
83
84  if xmlFile ~= nil then
85  local mapStoreItemsFilename = getXMLString(xmlFile,
    "map.storeItems#filename")
86  if mapStoreItemsFilename ~= nil then
87  mapStoreItemsFilename = Utils.getFilename(mapStoreItemsFilename,
    baseDirectory)
88  self:loadItemsFromXML(mapStoreItemsFilename)
89  end
90  end
91
92  for _, item in ipairs(self.modStoreItems) do
93  g_deferredLoadingManager:addSubtask(function()
94  self:loadItem(item.xmlFilename, item.baseDir,
    item.customEnvironment, item.isMod, item.isBundleItem,
    item.dlcTitle)
95  end)
96  end
97
98  return true
99  end
```

addCategory

Description

Adds a new store category

Definition

addCategory(string name, string title, string imageFilename, string baseDir)

Arguments

string name	category index name
string title	category title
string imageFilename	image
string baseDir	base directory

Return Values

boolean true if adding was successful else false

Code

```
128  function StoreManager:addCategory(name, title, imageFilename,
     type, baseDir)
129  if name == nil or name == "" then
```

```
130 print("Warning: Could not register store category. Name is missing or empty!")
131 return false
132 end
133 if not ClassUtil.getIsValidIndexName(name) then
134 print("Warning: '"..tostring(name).."' is no valid name for a category!")
135 return false
136 end
137 if title == nil or title == "" then
138 print("Warning: Could not register store category. Title is missing or empty!")
139 return false
140 end
141 if imageFilename == nil or imageFilename == "" then
142 print("Warning: Could not register store category. Image is missing or empty!")
143 return false
144 end
145 if baseDir == nil then
146 print("Warning: Could not register store category. Basedirectory not defined!")
147 return false
148 end
149
150 name = name:upper()
151
152 if self.categories[name] == nil then
153 self.numOfCategories = self.numOfCategories + 1
154
155 local category = {}
156 category.name = name
157 category.title = title
158 category.image = Utils.getFilename(imageFilename, baseDir)
159 category.type = StoreManager.CATEGORY_TYPE[type] ~= nil and type or StoreManager.CATEGORY_TYPE.NONE
160 category.orderId = self.numOfCategories
161
162 self.categories[name] = category
163 return true
164 end
165 return false
```

```
166 end
```

removeCategory

Description

Removes a store category

Definition

removeCategory(string name)

Arguments

string name category index name

Code

```
171 function StoreManager:removeCategory(name)
172 if not ClassUtil.getIsValidIndexName(name) then
173 print("Warning: '"..tostring(name).."' is no valid name for a category!")
174 return
175 end
176
177 name = name:upper()
178
179 for _, item in pairs(self.items) do
180 if item.category == name then
181 item.category = "MISC"
182 end
183 end
184 self.categories[name] = nil
185 end
```

getCategoryByName

Description

Gets a store category by name

Definition

getCategoryByName(string name)

Arguments

string name category index name

Return Values

table category the category object

Code

```
191 function StoreManager:getCategoryByName(name)
192 if name ~= nil then
193 return self.categories[name:upper()]
194 end
195 return nil
196 end
```

addSpecType

Description

Adds a new spec type

Definition

addSpecType(string name, string profile, function loadFunc, function getValueFunc)

Arguments

string	name	spec type index name
string	profile	spec type gui profile
function	loadFunc	the loading function pointer
function	getValueFunc	the get value function pointer

Code

```
204 function StoreManager:addSpecType(name, profile, loadFunc, getValueFunc)
205 if not ClassUtil.getIsValidIndexName(name) then
206 print("Warning: '"..tostring(name).."' is no valid name for a spec type!")
207 return
208 end
209
210 name = name
211
212 if self.nameToSpecType == nil then
213 printCallstack()
214 end
215
216 if self.nameToSpecType[name] ~= nil then
217 print("Error: spec type name '" ..name.. "' is already in use!")
218 return
219 end
220
221 local specType = {}
222 specType.name = name
223 specType.profile = profile
224 specType.loadFunc = loadFunc
225 specType.getValueFunc = getValueFunc
226
227 self.nameToSpecType[name] = specType
228 table.insert(self.specTypes, specType)
229 end
```

getSpecTypes

Description

Gets all spec types

Definition

getSpecTypes()

Return Values

table specTypes a list of spec types

Code

```
234  function StoreManager:getSpecTypes()
235  return self.specTypes
236  end
```

getSpecTypeByName

Description

Gets a spec type by name

Definition

getSpecTypeByName(string name)

Arguments

string name spec type index name

Return Values

table specType the corresponding spectype

Code

```
242  function StoreManager:getSpecTypeByName(name)
243  if not ClassUtil.getIsValidIndexName(name) then
244  print("Warning: '"..tostring(name).."' is no valid name for a
     spec type!")
245  return
246  end
247
248  return self.nameToSpecType[name]
249  end
```

addItem

Description

Adds a new store item

Definition

addItem(table storeItem)

Arguments

table storeItem the storeitem object

Return Values

boolean wasSuccessfull true if added else false

Code

```
255  function StoreManager:addItem(storeItem)
256  if self.xmlFilenameToItem[storeItem.xmlFilenameLower] ~= nil then
257  return false
258  end
259
260  table.insert(self.items, storeItem)
```

```
261 storeItem.id = #self.items
262 self.xmlFilenameToItem[storeItem.xmlFilenameLower] = storeItem
263 return true
264 end
```

removeItemByIndex

Description

Removes a storeitem by index

Definition

removeItemByIndex(integer index)

Arguments

integer index storeitem index

Code

```
269 function StoreManager:removeItemByIndex(index)
270 local item = self.items[index]
271 if item ~= nil then
272 self.xmlFilenameToItem[item.xmlFilenameLower] = nil
273
274 -- item.id must always match the index in the arry, thus swap the
    last to the removed position and reduce size
275 local numItems = table.getn(self.items)
276 if index < numItems then
277 self.items[index] = self.items[numItems]
278 self.items[index].id = index
279 end
280 table.remove(self.items, numItems)
281 end
282 end
```

getItems

Description

Gets all storeitems

Definition

getItems()

Return Values

table items a list of all store items

Code

```
287 function StoreManager:getItems()
288 return self.items
289 end
```

getItemByIndex

Description

Gets a store item by index

Definition

getItemByIndex(integer index)

Arguments

integer index store item index

Return Values

table storeItem the storeitem oject

Code

```
295  function StoreManager:getItemByIndex(index)
296  if index ~= nil then
297  return self.items[index]
298  end
299  return nil
300  end
```

getItemByXMLFilename

Description

Gets a store item xml filename

Definition

getItemByXMLFilename(string xmlFilename)

Arguments

string xmlFilename storeitem xml filename

Return Values

table storeItem the storeitem object

Code

```
306  function StoreManager:getItemByXMLFilename(xmlFilename)
307  if xmlFilename ~= nil then
308  return self.xmlFilenameToItem[xmlFilename:lower()]
309  end
310  end
```

getItemByCustomEnvironment

Description

Gets a store item xml filename

Definition

getItemByCustomEnvironment(string xmlFilename)

Arguments

string xmlFilename storeitem xml filename

Return Values

table storeItem the storeitem object

Code

```
316  function
     StoreManager:getItemByCustomEnvironment(customEnvironment)
317  local items = {}
318  for _, item in ipairs(self.items) do
319  if item.customEnvironment == customEnvironment then
320  table.insert(items, item)
```

```
321 end
322 end
323
324 return items
325 end
```

loadItem

Description

Loads a storeitem from xml file

Definition

loadItem(string xmlFilename, string baseDir, string customEnvironment, boolean isMod, boolean isBundleItem, string dlcTitle)

Arguments

string	xmlFilename	the storeitem xml filename
string	baseDir	the base directory
string	customEnvironment	a custom environment
boolean	isMod	true if item is a mod, else false
boolean	isBundleItem	true if item is bundleItem, else false
string	dlcTitle	optional dlc title

Return Values

table storeItem the storeitem object

Code

```
340 function StoreManager:loadItem(xmlFilename, baseDir,
    customEnvironment, isMod, isBundleItem, dlcTitle)
341 local xmlFilename = Utils.getFilename(xmlFilename, baseDir)
342 local xmlFile = loadXMLFile("storeItemXML", xmlFilename)
343 local baseXMLName = getXMLRootName(xmlFile)
344 local storeDataXMLName = baseXMLName..".storeData"
345
346 if not hasXMLProperty(xmlFile, storeDataXMLName) then
347 g_logManager:xmlError(xmlFilename, "No storeData found. StoreItem
    will be ignored!")
348 delete(xmlFile)
349 return nil
350 end
351
352 local isValid = true
353 local name = XMLUtil.getXMLI18NValue(xmlFile, storeDataXMLName,
    getXMLString, "name", nil, customEnvironment, true)
354 if name == nil then
355 g_logManager:xmlWarning(xmlFilename, "Name missing for storeitem.
    Ignoring store item!")
356 isValid = false
357 end
```

```
358
359 local imageFilename = Utils.getNoNil(getXMLString(xmlFile,
    storeDataXMLName..".image"), "")
360 if imageFilename == "" then
361 g_logManager:xmlWarning(xmlFilename, "Image icon is missing for
    storeitem. Ignoring store item!")
362 isValid = false
363 end
364
365 if not isValid then
366 delete(xmlFile)
367 return nil
368 end
369
370 local storeItem = {}
371 storeItem.name = name
372 storeItem.xmlFilename = xmlFilename
373 storeItem.xmlFilenameLower = xmlFilename:lower()
374 storeItem.imageFilename = Utils.getFilename(imageFilename, baseDir)
375 storeItem.functions = StoreItemUtil.getFunctionsFromXML(xmlFile,
    storeDataXMLName, customEnvironment)
376 storeItem.specs = StoreItemUtil.getSpecsFromXML(self.specTypes,
    xmlFile, customEnvironment)
377 storeItem.brandIndex = StoreItemUtil.getBrandIndexFromXML(xmlFile,
    storeDataXMLName, xmlFilename)
378 storeItem.species = Utils.getNoNil(getXMLString(xmlFile,
    storeDataXMLName..".species"), "")
379 storeItem.canBeSold = Utils.getNoNil(getXMLBool(xmlFile,
    storeDataXMLName..".canBeSold"), true)
380 storeItem.showInStore = Utils.getNoNil(getXMLBool(xmlFile,
    storeDataXMLName..".showInStore"), not isBundleItem)
381 storeItem.isBundleItem = isBundleItem
382 storeItem.allowLeasing = Utils.getNoNil(getXMLBool(xmlFile,
    storeDataXMLName..".allowLeasing"), true)
383 storeItem.maxItemCount = getXMLInt(xmlFile,
    storeDataXMLName..".maxItemCount")
384 storeItem.rotation = Utils.getNoNilRad(getXMLFloat(xmlFile,
    storeDataXMLName..".rotation"), 0)
385 storeItem.shopTranslationOffset =
    StringUtil.getVectorNFromString(getXMLString(xmlFile,
    storeDataXMLName..".shopTranslationOffset"), 3)
386 storeItem.shopRotationOffset =
    StringUtil.getRadiansFromString(getXMLString(xmlFile,
    storeDataXMLName..".shopRotationOffset"), 3)
```

```
387 storeItem.shopHeight = Utils.getNoNil(getXMLFloat(xmlFile,
    storeDataXMLName..".shopHeight"), 0)
388 storeItem.shopFoldingState = Utils.getNoNil(getXMLBool(xmlFile,
    storeDataXMLName..".shopFoldingState"), 0)
389 storeItem.sharedVramUsage, storeItem.perInstanceVramUsage,
    storeItem.ignoreVramUsage =
    StoreItemUtil.getVRamUsageFromXML(xmlFile, storeDataXMLName)
390 storeItem.dlcTitle = dlcTitle
391 storeItem.isMod = isMod
392 storeItem.customEnvironment = customEnvironment
393
394 local categoryName = getXMLString(xmlFile,
    storeDataXMLName..".category")
395 local category = self:getCategoryByName(categoryName)
396 if category == nil then
397 g_logManager:xmlWarning(xmlFilename, "Invalid category '%s' in store
    data! Using 'misc' instead!", tostring(categoryName))
398 category = self:getCategoryByName("misc")
399 end
400 storeItem.categoryName = category.name
401
402 storeItem.configurations =
    StoreItemUtil.getConfigurationsFromXML(xmlFile, baseXMLName,
    baseDir, customEnvironment, isMod, storeItem)
403 storeItem.subConfigurations =
    StoreItemUtil.getSubConfigurationsFromXML(storeItem.configurations)
404 storeItem.configurationSets =
    StoreItemUtil.getConfigurationSetsFromXML(storeItem, xmlFile,
    baseXMLName, baseDir, customEnvironment, isMod)
405 storeItem.price = Utils.getNoNil(getXMLFloat(xmlFile,
    storeDataXMLName..".price"), 0)
406 if storeItem.price < 0 then
407 g_logManager:xmlWarning(xmlFilename, "Price has to be greater than
    0. Using default 10.000 instead!")
408 storeItem.price = 10000
409 end
410 storeItem.dailyUpkeep = Utils.getNoNil(getXMLFloat(xmlFile,
    storeDataXMLName..".dailyUpkeep"), 0)
411 storeItem.runningLeasingFactor = Utils.getNoNil(getXMLFloat(xmlFile,
    storeDataXMLName..".runningLeasingFactor"),
    EconomyManager.DEFAULT_RUNNING_LEASING_FACTOR)
412 storeItem.lifetime = Utils.getNoNil(getXMLFloat(xmlFile,
    storeDataXMLName..".lifetime"), 600)
413
414
```

```
415 if hasXMLProperty(xmlFile, storeDataXMLName..".bundleElements") then
416 local bundleInfo = {bundleItems={}, attacherInfo={}}
417 local price = 0
418 local lifetime = math.huge
419 local dailyUpkeep = 0
420 local runningLeasingFactor = 0
421 local i = 0
422 while true do
423 local bundleKey =
    string.format(storeDataXMLName..".bundleElements.bundleElement(%d)",
    i)
424 if not hasXMLProperty(xmlFile, bundleKey) then
425 break
426 end
427 local bundleXmlFile = getXMLString(xmlFile,
    bundleKey..".xmlFilename")
428 local offset =
    StringUtil.getVectorNFromString(Utils.getNoNil(getXMLString(xmlFile,
    bundleKey..".offset"), "0 0 0"), 3)
429 local rotation = math.rad(Utils.getNoNil(getXMLFloat(xmlFile,
    bundleKey..".yRotation"), 0))
430 if bundleXmlFile ~= nil then
431 local completePath = Utils.getFilename(bundleXmlFile, baseDir)
432 local item = self:getItemByXMLFilename(completePath)
433 if item == nil then
434 item = self:loadItem(bundleXmlFile, baseDir, customEnvironment,
    isMod, true, dlcTitle)
435 end
436 if item ~= nil then
437 price = price + item.price
438 dailyUpkeep = dailyUpkeep + item.dailyUpkeep
439 runningLeasingFactor = runningLeasingFactor +
    item.runningLeasingFactor
440 lifetime = math.min(lifetime, item.lifetime)
441 table.insert(bundleInfo.bundleItems, {item=item,
    xmlFilename=item.xmlFilename, offset=offset, rotation=rotation,
    price=item.price})
442 end
443 end
444 i = i + 1
445 end
446 local i = 0
447 while true do
```

```
448 local attachKey = string.format(storeDataXMLName..".attacherInfo.attach(%d)", i)
449 if not hasXMLProperty(xmlFile, attachKey) then
450 break
451 end
452 local bundleElement0 = getXMLInt(xmlFile, attachKey.."#bundleElement0")
453 local bundleElement1 = getXMLInt(xmlFile, attachKey.."#bundleElement1")
454 local attacherJointIndex = getXMLInt(xmlFile, attachKey.."#attacherJointIndex")
455 local inputAttacherJointIndex = getXMLInt(xmlFile, attachKey.."#inputAttacherJointIndex")
456 if bundleElement0 ~= nil and bundleElement1 ~= nil and attacherJointIndex ~= nil and inputAttacherJointIndex ~= nil then
457 table.insert(bundleInfo.attacherInfo, {bundleElement0=bundleElement0, bundleElement1=bundleElement1, attacherJointIndex=attacherJointIndex, inputAttacherJointIndex=inputAttacherJointIndex})
458 end
459 i = i + 1
460 end
461
462 storeItem.price = price
463 storeItem.dailyUpkeep = dailyUpkeep
464 storeItem.runningLeasingFactor = runningLeasingFactor
465 storeItem.lifetime = lifetime
466 storeItem.bundleInfo = bundleInfo
467 end
468
469 self:addItem(storeItem)
470
471 delete(xmlFile)
472
473 return storeItem
474 end
```

TableElement

Description

Table GUI element.
-- Allows sorting by columns when clicking on header elements. Header elements should ideally be defined just before
the table itself, but never within the table.
--@category GUI
--@xmlConfig GuiElement#rowTemplateName string Element Name of row template.

The template will be replicated to fill the table's view and then discarded during initialization.

onGuiSetupFinished

Description

Called when the GUI is completely loaded. Link relevant collaborators of this table.

Definition

onGuiSetupFinished()

buildTableRows

Description

Build table row GUI elements.
First look for the row template, then replicate it as many times as configured (#itemsPerCol attribute)

Definition

buildTableRows()

processCellElements

Description

Process cell elements for cell-based navigation.

Definition

processCellElements()

applyAlternatingBackgroundsToRows

Description

Apply alternating background GUI profiles to rows if the required profiles have been specified (see configuration
properties).

Definition

applyAlternatingBackgroundsToRows()

addRow

Description

Add a data row to the end of this table.

Definition

addRow(dataRow DataRow, refreshView boolean)

Arguments

dataRow	DataRow	which holds an ID and column values as DataCell instances
refreshView	boolean	If true, triggers a refresh of the table's view.

removeRow

Description

Remove a row from this table.

Definition

removeRow(index Index, refreshView boolean)

Arguments

index	Index	of row to be removed.
refreshView	boolean	If true, triggers a refresh of the table's view.

clearData

Description

Clear all data from the table.

Definition

clearData(refreshView boolean)

Arguments

refreshView boolean If true, triggers a refresh of the table's view.

getViewDataCell

Description

Get a cell identified by row index and column name from this table's current view data.

Definition

getViewDataCell(rowIndex Index, colName string)

Arguments

rowIndex Index of row in view

colName string Name of data column

Return Values

DataCell or nil if not found.

getDataCell

Description

Get a cell identified by row index and column name from this table's data.

Definition

getDataCell(rowIndex Index, colName string)

Arguments

rowIndex Index of row in data

colName string Name of data column

Return Values

DataCell or nil if not found.

setCellText

Description

Set the text of a specific cell.
If the cell element does not have a setText() method, this method has no effect. Make sure to only target TextElement
descendants.

Definition

setCellText(rowIndex Index, colName string, text string, refreshView boolean)

Arguments

rowIndex	Index	of row in data
colName	string	Name of data column
text	string	New text value
refreshView	boolean	If true, triggers a refresh of the table's view. Use this when changing sortable column data.

setCellVisibility

Description

Set visibility of a specific cell.

Definition

setCellVisibility(rowIndex Index, colName string, isVisible boolean)

Arguments

rowIndex	Index	of row in data
colName	string	Name of data column
isVisible	boolean	New visibility value of the cell element

setCellOverrideGuiProfile

Description

Apply a temporary overriding GUI profile to a specific cell.
GUI profiles are defined in dataS/guiProfiles.xml. If this is called without a profile name argument, the original
profile is applied on the next table refresh.

Definition

setCellOverrideGuiProfile(rowIndex Index, colName string, profileName string)

Arguments

rowIndex	Index	of row in data
colName	string	Name of data column
profileName	string	Name of GUI profile to override or nil to reset.

disableSorting

Description

Disables all sorting, does not restore item order.

Definition

disableSorting()

deleteListItems

Description

Delete all list items.
Override from ListElement: Clears all data and updates the view.

Definition

deleteListItems()

onClickHeader

Description

Handle clicks on a header element for sorting.
This must be called by the containing screen on callback. The view update needs to be triggered separately.

Definition

onClickHeader(headerElement The)

Arguments

headerElement	The	header which has been clicked.

getSortableColumn

Description

Get a columns data in SortCell instances to apply sorting.

Definition

getSortableColumn(columnName Name)

Arguments

columnName Name of the data column, must be configured in #columnNames

Return Values

List of SortCell for the requested column name

setCustomSortFunction

Description

Set a custom sorting function to expand on the default data sorting function.

Definition

setCustomSortFunction(sortFunction Sorting, useBeforeData If)

Arguments

sortFunction	Sorting	function which takes two SortCell instances to compare and returns a numeric value; value < 0 : e1 < e2, value == 0 : e1 == e2, value > 0 : e1 > e2
useBeforeData	If	true, the custom function will be the first order function. Otherwise, elements will be ordered by data first and custom sorting second.

setProfileOverrideFilterFunction

Description

Set a filter function for the application of GUI profile overrides.

Definition

setProfileOverrideFilterFunction(filterFunction Must)

Arguments

filterFunction	Must	take a DataRow instance and return true if the override profile should be applied or false otherwise.

getSortFunction

Description

Get the table sorting function. If a custom sorting function has been set, the standard data cell sorting will be decorated by it. When setting a custom sorting function, another parameter is given which determines if the custom function should be the first or second criterion in sorting.

Definition

getSortFunction(isAscending True)

Arguments

isAscending True if data should be sorted in ascending order, false if descending

updateSortedView

Description

Updates the sorted data view with the required sorting order

Definition

updateSortedView(columnName Name, sortingOrder TableHeaderElement.SORTING_X)

Arguments

columnName	Name	of the column by which rows will be sorted or nil to revert to unsorted state
sortingOrder	TableHeaderElement.SORTING_X	value for sort order

invalidateLayout

Description

Rebuild the table layout.
This will reposition all row elements and fade them out. If they receive data, they are made visible again.

Definition

invalidateLayout()

updateSelectedIndex

Description

Update the selected index after view / data changes.
The method tries getting the new selected index by matching data row IDs.

Definition

updateSelectedIndex()

updateRows

Description

Apply data and attributes from visible items to row elements.

Definition

updateRows()

scrollToItemInView

Description

Scrolls to an item so that it naturally comes into view.
If the target index is above, this will set the item at the top. Likewise, if the index is below the current view,
the item will be shown at the bottom.

Definition

scrollToItemInView(index Target)

Arguments

index Target index to scroll to

updateView

Description

Update the table's view. Use this for external updates after data changes.
Fills in data into row elements according to current data and sorting order.

Definition

updateView(refocus If)

Arguments

refocus If true, the view will be scrolled to the currently selected item after the update.

getItemIndexByRealRowColumn

Description

Get item index from a row selection.
Override from ListElement: only handles rows.

Definition

getItemIndexByRealRowColumn(realRow Row)

Arguments

realRow Row data selection index

setSelectionByRealRowAndColumn

Description

Set selection based on list row and column.
Override from ListElement: reads data index from selected table row to select it.

Definition

setSelectionByRealRowAndColumn(realRow Row)

Arguments

realRow Row selection index

updateRowSelection

Description

Update selection state of table rows

Definition

updateRowSelection()

Return Values

index of selected row

getItemFactor

Description

Get an item factor for visual proportions.
Override from ListElement: Simplified table interaction with slider element to require straight indices only.
Therefore this only needs to return 1 now.

Definition

getItemFactor()

Return Values

scrollTo

Description

Scroll to a data view index position.

Definition

scrollTo(index Data, updateSlider If)

Arguments

index	Data index. The view will update to show the indexed data item at the top.
updateSlider If	true, will update the slider position if present

setSelectedIndex

Description

Set the selected index in table data view.
Scrolls to the newly selected row index if necessary.

Definition

setSelectedIndex(index Numeric, force If)

Arguments

index Numeric	index in table data view
force If	true, always forces a selection changed callback, even if the index remains the same

getSelectedElement

Description

Get the selected element.
Override from ListElement: Instead of returning raw GuiElement instances, this will provide the selected DataRow.

Definition

getSelectedElement()

Return Values

dataRow instance or nil, selected data index or 0

getSelectedTableRow

Description

Get the currently selected, visible TableRow instance.

Definition

getSelectedTableRow()

getDataRowForElement

Description

Get the data row which corresponds to the visual position of a given element.
This is useful for cases when getSelectedElement() does not currently represent a valid state, e.g. in click
callbacks of buttons contained in table rows (called before the table's mouse event).

Definition

getDataRowForElement(element GuiElement)

Arguments

element GuiElement instance which should be part of this table's row elements

Return Values

dataRow instance or nil (element not part of table row)

getItemCount

Description

Get the number of data view rows.
Override from ListElement.

Definition

getItemCount()

Return Values

number of data view rows

updateItemPositions

Description

Update item and data indices of rows.

Definition

updateItemPositions()

shouldFocusChange

Description

Determine if focus should change from this element in a given direction.
Override from ListElement: Change focus when moving out to the side or selection is at the start/end of the table.

Definition

shouldFocusChange(direction Focus)

Arguments

direction Focus navigation direction

onFocusEnter

Description

Handle receiving focus event

Definition

onFocusEnter()

getFocusTarget

Description

Get the actual focus target, in case a child or parent element needs to be targeted instead.

Definition

getFocusTarget(incomingDirection (Optional), moveDirection (Optional))

Arguments

incomingDirection	(Optional)	If specified, may return different targets for different incoming directions.
moveDirection	(Optional)	Actual movement direction per input. This is the opposing direction of incomingDirection.

Return Values

GuiElement Actual element to focus.

onMouseUp

Description

Handle mouse button up (after down) event
Override from ListElement: also update visual selection status of table row elements.

Definition

onMouseUp()

inputEvent

Description

Handle navigation input on this table.

Definition

inputEvent()

delayNavigationInput

Description

Lock and delay up and down directions for focus navigation.

Definition

delayNavigationInput()

TableHeaderElement

Description

Table header element to use within tables.
-- Children which serve as sorting icons are to be marked in the screen XML configuration by setting the name attribute
to "iconAscending" or "iconDescending" depending on which sorting state they are intended to represent. Headers must
be defined outside of tables, because anything within a table is considered a table item.

--@category GUI
--@xmlConfig GuiElement#targetTableId string Configured ID of the decorated table

addElement

Description

Add a child element to this GUI element. This element searches for marked children to use as sorting icons.

Definition

addElement(element Element)

Arguments

element Element to add.

toggleSorting

Description

Toggle this header's sorting display state, if allowed.

Definition

toggleSorting()

Return Values

The new sorting order

disableSorting

Description

Disable sorting on this header by setting its sorting state to OFF.

Definition

disableSorting()

updateSortingDisplay

Description

Update the header's display with its new state.

Definition

updateSortingDisplay()

TerrainDeformation
Description

Terrain deformation.
-- This class wraps terrain deformation engine functions for convenient usage.
--@category Terrain

new

Description

Create a new terrain deformation object.
Take care to call "apply" on each such object to ensure proper handling and clean-up on the engine side.

Definition

new(terrainNode Terrain)

Arguments

terrainNode Terrain root node, typically retrieved from "currentMission.terrainRootNode" or similar.

addArea

Description

Add a parallelogram area.
Terrain will be modified to fit to this area and receive the provided material. Several areas can be added per
deformation process.

Definition

addArea(x Parallelogram, y Parallelogram, Z Parallelogram, side1X Parallelogram, side1Y Parallelogram, side1Z Parallelogram, side2X Parallelogram, side2Y Parallelogram, side2Z Parallelogram, terrainBrushId Terrain)

Arguments

x	Parallelogram origin X position in world space
y	Parallelogram origin Y position in world space
Z	Parallelogram origin Z position in world space
side1X	Parallelogram side 1 X position offset relative to origin
side1Y	Parallelogram side 1 Y position offset relative to origin
side1Z	Parallelogram side 1 Z position offset relative to origin
side2X	Parallelogram side 2 X position offset relative to origin
side2Y	Parallelogram side 2 Y position offset relative to origin
side2Z	Parallelogram side 2 Z position offset relative to origin
terrainBrushId Terrain	material ID, currently just the map layer index to apply

setOutsideAreaBrush

Description

Set terrain material for areas outside of the given parallelograms but within the smoothing radius.

Definition

setOutsideAreaBrush(brushId Terrain)

Arguments

brushId Terrain	material ID, currently just the map layer index to apply

setOutsideAreaConstraints

Description

Set the constraints for smoothing the area outside the added parallelogram areas.

Definition

setOutsideAreaConstraints(maxSmoothDistance Radius, maxSlope Maximum, maxEdgeAngle Maximum)

Arguments

maxSmoothDistance	Radius	around parallelogram areas to apply smoothing
maxSlope	Maximum	allowed slope of terrain created by smoothing in radians
maxEdgeAngle	Maximum	allowed angle between individual polygons in the smoothed terrain in radians, this allows reducing the rate of change of the slope

getBlockedAreaMapSize

Description

Get the size of the area map needed for the terrain

Definition

getBlockedAreaMapSize()

setDynamicObjectCollisionMask

Description

Set the collision mask used to detect dynamic objects that are in the way of the deformation

Definition

setDynamicObjectCollisionMask()

setDynamicObjectMaxDisplacement

Description

Set the max displacement allowed for dynamic objects

Definition

setDynamicObjectMaxDisplacement()

setBlockedAreaMap

Description

Set an area map of terrain which must not be modified.
When a "1" appears in the channel for a terrain cell, the terrain remains unmodified at that point. Smoothing occurs
around the blocked areas as well as around the parallelogram areas.

Definition

setBlockedAreaMap(bitVectorMapId ID, channel Bit)

Arguments

bitVectorMapId	ID	of a bit vector map which needs to have the size as returned by getBlockedAreaMapSize()
channel	Bit	index of the channel within the bit vector map to test for blocking information

setBlockedAreaMaxDisplacement

Description

Set the max displacement allowed for the blocked area before a blocked failure occurs

Definition

setBlockedAreaMaxDisplacement()

apply

Description

Apply the terrain deformation.

Definition

apply(previewOnly If, callbackFunc Name, callbackObject Object)

Arguments

previewOnly	If	true, will only return callback data without committing changes to the terrain. Otherwise, the modification will be applied directly, if possible.
callbackFunc	Name	of a callback function which is called when calculation is finished. Signature: function(callbackObject, canDeform, displacedVolume, blocked)
callbackObject	Object	which holds callbackFunc and first argument to callback if set. If no callbackObject is provided, the callback function is assumed to be global and will be called as such.

cancel

Description

Cancel an ongoing terrain deformation process.
This will cancel calculations after apply() has been called and before the callback. Otherwise it has no effect.

Definition

cancel()

TextElement
Description

Text display element
--@category GUI
--@xmlConfig GuiElement#textColor string [optional] Normal state text color, defaults to white [1, 1, 1, 1]. Format: "[r] [g] [b] [a]" where each value is in the range of [0.0, 1.0]. The first three values represent red, green and blue color channels and the last is the alpha (translucency) value. When setting this property in guiProfiles.xml, preset values as defined at the top of that file may also be used.

getTextPosition

Description

Get the text position for drawing for the current alignment settings and including modifications to text.

Definition

getTextPosition()

TextInputDialog
Description

Text input dialog.
-- @field textElement Text input element

new

Description

Create a new TextInputDialog instance.

Definition

new()

TextInputElement
Description

Text input element which captures strings from player input.
-- Used layers: "cursor" for a text input cursor icon.
-- TODO: IME property docs
--@category GUI
--@xmlConfig GuiElement#imeKeyboardType string [optional] Input method editor keyboard type, defaults to "normal". TODO: add valid types based on engine code

setCaptureInput

Description

Set input capturing state.
When capturing, the standard input bindings are disabled (input context switch).

Definition

setCaptureInput()

inputEvent

Description

Handle GUI input events.
Reacts to confirmation and cancel actions. Also see GuiElement.inputEvent().

Definition

inputEvent()

ToggleButtonElement
Description

Toggle button
-- TODO: Refactor child display element retrieval
-- Used layers: "image" for the background.
--@category GUI
--@xmlConfig GuiElement#isChecked bool [optional] If true, the button is initialized in checked state.
--@xmlConfig GuiElement#onClick callback [optional] onClick(element, isChecked) Called when the element is clicked. Receives this element and the current toggle state as a boolean (true for checked, false for unchecked).

getFocusTarget

Description

Get the actual focus target of this element.

Definition

getFocusTarget()

TopNotification
Description

HUD top notification element.
-- Displays notifications issued by other game components at the top of the screen.
--@category GUI

new

Description

Create a new TopNotification.

Definition

new(string hudAtlasPath)

Arguments

string hudAtlasPath Path to the HUD atlas texture.

Return Values

table value value at the given position

table TopNotification instance

setNotification

Description

Set a notification to be displayed in a frame at the top of the screen.
If another notification is being displayed, it is immediately replaced by this new one.

Definition

setNotification(string title, string text, string info, table iconKey, int duration)

Arguments

string	title	Notification title
string	text	Notification message text
string	info	Additional info text
table	iconKey	[optional] Icon key for a display icon, use a value from TopNotification.ICON
int	duration	[optional] Display duration in milliseconds. Negative values or nil default to a long-ish standard duration.

Return Values

table instance instance of object

getHidingTranslation

Description

Get the screen space translation for hiding.
Override in sub-classes if a different translation is required.

Definition

getHidingTranslation()

Return Values

table value value at the given position

float Screen space X translation

float Screen space Y translation

update

Description

Update notification state.

Definition

update()

Return Values

integer row row

integer column column

draw

Description

Draw notification.

Definition

draw()

getBackgroundPosition

Description

Get this element's base background position.

Definition

getBackgroundPosition(float uiScale)

Arguments

float uiScale Current UI scale factor

Return Values

table instance instance of object

setScale

Description

Set uniform UI scale.

Definition

setScale()

Return Values

table instance instance of object

storeScaledValues

Description

Store scaled positioning, size and offset values.

Definition

storeScaledValues()

Return Values

boolean true if loading was successful else false

createBackground

Description

Create the background overlay.

Definition

createBackground()

Return Values

boolean canTip can tip to ground

createComponents

Description

Create required display components.

Definition

createComponents()

Return Values

integer fillType fill type found

Tween

Description

Tween class which linearly interpolates a quantity from a start value to an end value over a given duration.
--@category GUI

new

Description

Create a new Tween.

Definition

new(table subClass, function setterFunction, float startValue, float endValue, float duration)

Arguments

table	subClass	Subclass metatable for inheritance
function	setterFunction	Value setter function. Signature: callback(value) or callback(target, value).
float	startValue	Original value
float	endValue	Target value
float	duration	Duration of tween in milliseconds

getDuration

Description

Get this tween's duration in milliseconds.

Definition

getDuration()

getFinished

Description

Check if this tween has finished.

Definition

getFinished()

reset

Description

Reset this tween to play it again.

Definition

reset()

setTarget

Description

Set a callback target for this tween.
If a target has been set, the setter function must support receiving the target as its first argument.

Definition

setTarget()

update

Description

Update the tween's state.

Definition

update()

tweenValue

Description

Get the current tween value.

Definition

tweenValue()

applyValue

Description

Apply a value via the setter function.

Definition

applyValue()

TweenSequence

Description

Tween sequence.
-- Allows setting up more complex tweening by defining sequences of tweens, intervals and callbacks. A sequence is
itself a Tween, so you may even define and add sub-sequences.
-- Before a sequence reacts to update() calls, it must be started with start(). This also applies after resetting.
-- Adding tweens, callbacks and intervals will append them to the current sequence.
Insertion of tweens and callbacks
will insert them at the given relative instants, allowing for overlapping tweens and arbitrary callback times.
Inserting an interval will push pack all later instants by the given time.
--@category GUI

new

Description

Create a new TweenSequence.

Definition

new(table functionTarget)

Arguments

table functionTarget	[optional] Target table which is supplied by default to all tween setter functions and callbacks as the first argument. If not specified, the setters and callbacks will be called with one value only.

insertTween

Description

Insert a tween at a given instant.

Definition

insertTween(table tween, float instant)

Arguments

table tween	Tween instance
float instant	Time in milliseconds after sequence start

addTween

Description

Add a tween to the end of the sequence.

Definition

addTween(table tween)

Arguments

table tween	Tween instance

insertInterval

Description

Insert an interval at the given instant.
This will push back all later instants by the interval. Use this to insert pauses into the sequence.

Definition

insertInterval(float interval, float instant)

Arguments

float interval	Interval time in milliseconds
float instant	Time in milliseconds after sequence start

addInterval

Description

Add an interval at the end of the sequence.
Use this to add a pause to the sequence.

Definition

addInterval()

insertCallback

Description

Insert a callback at the given instant.

Definition

insertCallback(function callback, table callbackState, float instant)

Arguments

function	callback	Callback function with signature of either callback(target, value) or callback(value)
table	callbackState	Any value which is passed to the callback as its first (no target) or second (with target) argument
float	instant	Time in milliseconds after sequence start

addCallback

Description

Add a callback at the end of the sequence.

Definition

addCallback(function callback, table callbackState)

Arguments

function	callback	Callback function with signature of either callback(target, value) or callback(value)
table	callbackState	Any value which is passed to the callback as its first (no target) or second (with target) argument

getDuration

Description

Get this tween's duration in milliseconds.

Definition

getDuration()

setTarget

Description

Set a callback target for this tween.
If a target has been set, the setter function must support receiving the target as its first argument.

Definition

setTarget()

setLooping

Description

Set the looping state for this sequence.

Definition

setLooping(bool isLooping)

Arguments

bool isLooping If true, will restart the sequence when finished, including callbacks!

start

Description

Start the sequence.
A sequence will only update its state when it has been started.

Definition

start()

stop

Description

Stop the sequence.

Definition

stop()

reset

Description

Reset the sequence to its initial state.

Definition

reset()

update

Description

Update the sequence state over time.

Definition

update()

updateTweens

Description

Update active sequence tweens.

Definition

updateTweens(float lastInstant, float dt)

Arguments

float lastInstant	Last instant which received an update
float dt	Delta time

updateCallbacks

Description

Update callback states.

Definition

updateCallbacks()

UnBanDialog

Description

Un-ban dialog.
-- Displays a list of banned users to allow administrators to lift their bans.
-- @field dialogElement Main dialog frame GUI element which holds other elements

new

Description

Create a new UnBanDialog instance.

Definition

new()

rebuildBanList

Description

(Re-)Builds the ban list from ban storage data.

Definition

rebuildBanList()

updateButtons

Description

Update unban buttons visibility.

Definition

updateButtons()

setCallback

Description

Set the function which is called when this dialog returns.

Definition

setCallback()

closeAndCallback

Description

Close and call the notification callback.

Definition

closeAndCallback()

onListSelectionChanged

Description

Handle changes in the ban list selection.

Definition

onListSelectionChanged()

onClickBack

Description

Handle activation of the back button event.

Definition

onClickBack()

onClickCancel

Description

Handle activation of the cancel button event which unbans a single player.

Definition

onClickCancel()

onClickActivate

Description

Handle activation of the activate button event which unbans all users.

Definition

onClickActivate()

VehicleHUDExtension

Description

Custom vehicle HUD drawing extension.
-- This serves as the base class for custom specific drawing cases of vehicles in the HUD, e.g. MixerWagon fill levels.
-- To create new HUD extensions for vehicle specializations:
1. sub-class this base class
2. source() the sub-class module after its corresponding specialization's table has been declared
3. call VehicleHUDExtension.registerHUDExtension([specialization], [HUDextension])

in sub-class module
--@category GUI

new

Description

Base constructor for vehicle HUD extensions.

Definition

new(table class_mt, table vehicle, float uiScale, table uiTextColor, float uiTextSize)

Arguments

table	class_mt	Sub-class metatable
table	vehicle	Vehicle which has the specialization required by a sub-class
float	uiScale	Current UI scale
table	uiTextColor	HUD text drawing color as an RGBA array
float	uiTextSize	HUD text size

Return Values

float	fillLevel fill level found

delete

Description

Delete this instance and clean up resources.

Definition

delete()

Return Values

float	densityHeight	density height
float	deltaDensityHeight	delta of density height to terrain underneath

addComponentForCleanup

Description

Add a display component for cleanup on delete().
Added components must support delete() themselves or they will be ignored.

Definition

addComponentForCleanup()

getDisplayHeight

Description

Get this HUD extension's display height.
Override in subclasses.

Definition

getDisplayHeight()

Return Values

float	physicsDensityHeight	density height
float	deltaPhysicsDensityHeight	delta of physics collision to terrain underneath

canDraw

Description

Determine if this HUD extension is in a valid state for a call to draw() in the current frame.
Override in sub-classes with custom logic.

Definition

canDraw()

Return Values

bool If true, the HUD extension should be drawn in the current frame.

draw

Description

Draw HUD extension.

Definition

draw(float leftPosX, float rightPosX, float posY)

Arguments

float leftPosX	Left input help panel column start position
float rightPosX	Right input help panel column start position
float posY	Current input help panel drawing vertical offset

Return Values

float dropped	real fill level dropped
float lineOffset	line offset
float Modified	input help panel drawing vertical offset

registerHUDExtension

Description

Register a HUD extension for a specialization.

Definition

registerHUDExtension(table specializationType, table hudExtensionType)

Arguments

table specializationType	Vehicle specialization class type table
table hudExtensionType	HUD extension class type table corresponding to the given vehicle specialization

createHUDExtensionForSpecialization

Description

HUD extension factory method, creates a HUD extension for a given vehicle specialization.

Definition

createHUDExtensionForSpecialization(table spec, table vehicle, float uiScale, table uiTextColor, float uiTextSize)

Arguments

table spec	Specialization reference
table vehicle	Vehicle which has the given specialization
float uiScale	Current UI scale
table uiTextColor	HUD text drawing color as an RGBA array
float uiTextSize	HUD text size

Return Values

float dropped	real fill level dropped
float lineOffset	line offset
table HUD	extension instance or nil of no extension has been registered for the given specialization

hasHUDExtensionForSpecialization

Description

Check if there is a HUD extension for a given specialization.

Definition

hasHUDExtensionForSpecialization()

VehicleSchemaDisplay
Description

HUD vehicle schema display.
-- Displays a schematic view of the current vehicle configuration.
--@category GUI

new
Description

Create a new instance of VehicleSchemaDisplay.

Definition

new(table modManager)

Arguments

table modManager ModManager reference

Return Values

float fillLevel fill level removed

delete
Description

Delete this element.
Also deletes all loaded vehicle schema overlays.

Definition

delete()

Return Values

float fillLevel fill level changed

loadVehicleSchemaOverlays
Description

Load vehicle schema overlays from global and mod definitions.

Definition

loadVehicleSchemaOverlays()

Return Values

table instance Instance of object

loadVehicleSchemaOverlaysFromXML
Description

Load and create vehicle schema overlays from XML definitions.

Definition

loadVehicleSchemaOverlaysFromXML(int xmlFile, string modPath)

Arguments

int xmlFile XML file handle of vehicle schema definitions

string modPath Path to the current mod description or nil for the base game

Return Values

boolean true if loading was successful else false

setVehicle
Description

Set the currently controlled vehicle to display its schematic view.

Definition

setVehicle(table vehicle)

Arguments

table vehicle Vehicle reference

Return Values

table instance instance of object

lateSetDocked

Description

Animation method to set docked state at a delayed time by callback.

Definition

lateSetDocked()

Return Values

boolean true if loading was successful else false

setDocked

Description

Set the schema's docking state.
This element's position is updated based on the docking state.

Definition

setDocked(bool isDocked)

Arguments

bool isDocked If true, the schema should be display docked to the HUD input help display. Otherwise, it will take the input help's place in the top left corner.

Return Values

boolean true if loading was successful else false

draw

Description

Draw the vehicle schema display.
Only draws the schema if a controlled vehicle is set.

Definition

draw()

Return Values

boolean true if loading was successful else false

animateDocking

Description

Animate docking / undocking from input help display.

Definition

animateDocking(float startX, float startY, float targetX, float targetY, bool isDocking)

Arguments

float startX	Screen space starting X position of animation
float startY	Screen space starting Y position of animation
float targetX	Screen space target X position of animation

float targetY	Screen space target Y position of animation
bool isDocking	If true, moving to docking position. If false, moving to stand-alone position.

Return Values

integer mapHandle	id of bitvector

collectVehicleSchemaDisplayOverlays

Description

Recursively get vehicle schema overlay parts for a vehicle configuration.

Definition

collectVehicleSchemaDisplayOverlays()

Return Values

boolean isOwned	true if farm owns world position point, else false

getVehicleSchemaOverlays

Description

Get a vehicle configuration's schema overlays, including the root vehicle.

Definition

getVehicleSchemaOverlays()

Return Values

integer	farmId	id of farm. Returns 0 if land is not owned by anyone
table	Array	of overlay descriptions: {overlay=overlay, x=0, y=0, rotation=0, invertX=false, invisibleBorderRight=vehicle.schemaOverlay.invisibleBorderRight, invisibleBorderLeft=vehicle.schemaOverlay.invisibleBorderLeft}
float		Screen space height of root vehicle schema overlay

getSchemaDelimiters

Description

Get minimum and maximum screen space X positions of vehicle schema overlay descriptions.
The returned positions are relative to the position of the root vehicle schema overlay.

Definition

getSchemaDelimiters(table overlayDescriptions)

Arguments

table overlayDescriptions	Array of overlay descriptions, see VehicleSchemaDisplay:getVehicleSchemaOverlays()

Return Values

integer	farmlandId	farmland id. if 0, world position is no valid/buyable farmland
float	Minimum	X position (left)
float	Maximum	X position (right)

drawVehicleSchemaOverlays

Description

Draw vehicle schema icons for a given vehicle.

Definition

drawVehicleSchemaOverlays(table vehicle)

Arguments

table vehicle	Current vehicle

Return Values

boolean isValid true if id is valid, else false

getSchemaOverlayForState

Description

Get a schema overlay for a given vehicle's schema overlay data and current state.

Definition

getSchemaOverlayForState(table schemaOverlayData, bool isTurnedOn, bool isSelected, bool isImplement)

Arguments

table	schemaOverlayData	VehicleSchemaOverlayData instance of the current vehicle
bool	isTurnedOn	True if the vehicle is currently turned on, i.e. its function is active
bool	isSelected	True if the vehicle is currently selected for input
bool	isImplement	True if the vehicle is an implement (i.e. attached to a motorized vehicle), false if it's the root vehicle

Return Values

table	farmland farmland object
table	Schema Overlay instance

setScale

Description

Set this element's UI scale.

Definition

setScale(float uiScale)

Arguments

float uiScale UI scale factor

Return Values

table farmlands all available farmlands

storeScaledValues

Description

Store scaled positioning, size and offset values.

Definition

storeScaledValues()

Return Values

farmlandIds table list of farmland ids owned by given farm id

getBackgroundPosition

Description

Get the vehicle schema's base background position.

Definition

getBackgroundPosition(bool isDocked, float uiScale)

Arguments

bool	isDocked	If true, the vehicle schema is docked to the input help display
float	uiScale	Current UI scale

Return Values

float	localPosX local position x
float	localPosZ local position z

createBackground

Description

Create an empty background positioning overlay.

Definition

createBackground()

VehicleSchemaOverlayData

Description

Vehicle schema overlay data.
-- The game HUD draws vehicle schemas based on this data.
--@category Vehicles

new

Description

Create a new VehicleSchemaOverlayData instance.

Definition

new(float offsetX, float offsetY, string schemaNameDefault, string schemaNameOn, string schemaNameSelected, string schemaNameSelectedOn, float invisibleBorderRight, float invisibleBorderLeft)

Arguments

float	offsetX	Schema X position offset as a fraction of the schema overlay size width
float	offsetY	Schema Y position offset as a fraction of the schema overlay size height
string	schemaNameDefault	Name of schema overlay for the vehicle default state
string	schemaNameOn	Name of schema overlay for the vehicle turned on state
string	schemaNameSelected	Name of schema overlay for the vehicle selected state
string	schemaNameSelectedOn	Name of schema overlay for the vehicle selected and turned on state
float	invisibleBorderRight	Right margin width of schema overlay display expressed as a fraction of the overlay width
float	invisibleBorderLeft	Left margin width of schema overlay display expressed as a fraction of the overlay width

addAttacherJoint

Description

Add attacher joint information.

Definition

addAttacherJoint(float attacherOffsetX, float attacherOffsetY, float rotation, bool invertX, liftedOffsetX X, liftedOffsetY Y)

Arguments

float	attacherOffsetX	Attached vehicle schema overlay X offset expressed as a fraction of the overlay width
float	attacherOffsetY	Attached vehicle schema overlay Y offset expressed as a fraction of the overlay height
float	rotation	Attached vehicle schema rotation
bool	invertX	If true, the attached vehicle schema needs to be flipped horizontally (e.g. front attachments)
liftedOffsetX	X	position offset in lifted position in reference resolution pixels
liftedOffsetY	Y	position offset in lifted position in reference resolution pixels

WorkMode

Description

This is the specialization for switchable work modes
-- @author Stefan Maurus

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
21  function WorkMode.prerequisitesPresent(specializations)
22  return SpecializationUtil.hasSpecialization(WorkArea,
    specializations)
23  and SpecializationUtil.hasSpecialization(AnimatedVehicle,
    specializations)
24  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
59  function WorkMode:onLoad(savegame)
60  local spec = self.spec_workMode
61
62  spec.state = 1
63  spec.stateMax = 0
64
65  local baseKey = "vehicle.workModes"
66
67  spec.foldMaxLimit = Utils.getNoNil(getXMLFloat(self.xmlFile,
    string.format("%s#foldMaxLimit", baseKey)), 1)
68  spec.foldMinLimit = Utils.getNoNil(getXMLFloat(self.xmlFile,
    string.format("%s#foldMinLimit", baseKey)), 0)
69  spec.allowChangeOnLowered =
    Utils.getNoNil(getXMLBool(self.xmlFile,
    string.format("%s#allowChangeOnLowered", baseKey)), true)
70
```

```
spec.workModes = {}
local i = 0
while true do
local key = string.format("%s.workMode(%d)", baseKey, i)
if not hasXMLProperty(self.xmlFile, key) then
break
end

local entry = {}
entry.name = g_i18n:getText( getXMLString(self.xmlFile, key .. "#name") )
local inputBindingName = getXMLString(self.xmlFile, key .. "#inputBindingName")
if inputBindingName ~= nil then
if InputAction[inputBindingName] ~= nil then
entry.inputAction = InputAction[inputBindingName]
end
end

entry.turnedOnAnimations = {}
local j = 0
while true do
local key2 = string.format("%s.turnedOnAnimations.turnedOnAnimation(%d)", key, j)
if not hasXMLProperty(self.xmlFile, key2) then
break
end

local turnedOnAnimation = {}
turnedOnAnimation.name = getXMLString(self.xmlFile, key2.."#name")
turnedOnAnimation.turnOnFadeTime = Utils.getNoNil(getXMLFloat(self.xmlFile, key2.."#turnOnFadeTime"), 1) * 1000
turnedOnAnimation.turnOffFadeTime = Utils.getNoNil(getXMLFloat(self.xmlFile, key2.."#turnOffFadeTime"), 1) * 1000
turnedOnAnimation.speedScale = Utils.getNoNil(getXMLFloat(self.xmlFile, key2.."#speedScale"), 1)

turnedOnAnimation.speedDirection = 0
turnedOnAnimation.currentSpeed = 0
```

```
104
105 if self:getAnimationExists(turnedOnAnimation.name) then
106 table.insert(entry.turnedOnAnimations, turnedOnAnimation)
107 end
108
109 j = j + 1
110 end
111
112 entry.loweringAnimations = {}
113 j = 0
114 while true do
115 local key2 =
    string.format("%s.loweringAnimations.loweringAnimation(%d)", key,
    j)
116 if not hasXMLProperty(self.xmlFile, key2) then
117 break
118 end
119
120 local loweringAnimation = {}
121 loweringAnimation.name = getXMLString(self.xmlFile,
    key2.."#name")
122 loweringAnimation.speed =
    Utils.getNoNil(getXMLFloat(self.xmlFile, key2.."#speed"), 1)
123
124 if self:getAnimationExists(loweringAnimation.name) then
125 table.insert(entry.loweringAnimations, loweringAnimation)
126 end
127
128 j = j + 1
129 end
130
131 entry.workAreas = {}
132 j = 0
133 while true do
134 local key2 = string.format("%s.workAreas.workArea(%d)", key, j)
135 if not hasXMLProperty(self.xmlFile, key2) then
136 break
137 end
138
139 local workArea = {}
```

```
140 workArea.workAreaIndex = Utils.getNoNil(getXMLInt(self.xmlFile,
    key2.."#workAreaIndex"), j+1)
141 workArea.dropAreaIndex = Utils.getNoNil(getXMLInt(self.xmlFile,
    key2.."#dropAreaIndex"), j+1)
142
143 table.insert(entry.workAreas, workArea)
144
145 j = j + 1
146 end
147
148 entry.animations = {}
149 j = 0
150 while true do
151 local animKey = string.format("%s.animation(%d)", key, j)
152 if not hasXMLProperty(self.xmlFile, animKey) then
153 break
154 end
155
156 local animation = {}
157 animation.animName = getXMLString(self.xmlFile, animKey ..
    "#name")
158 animation.animSpeed = Utils.getNoNil(getXMLFloat(self.xmlFile,
    animKey .. "#speed"), 1.0)
159 animation.stopTime = getXMLFloat(self.xmlFile, animKey ..
    "#stopTime")
160 animation.repeatAfterUnfolding =
    Utils.getNoNil(getXMLBool(self.xmlFile, animKey ..
    "#repeatAfterUnfolding"), false)
161 animation.repeatStartTime = getXMLFloat(self.xmlFile, animKey ..
    "#repeatStartTime")
162 animation.repeated = false
163
164 if self:getAnimationExists(animation.animName) then
165 table.insert(entry.animations, animation)
166 end
167
168 j = j + 1
169 end
170
171 entry.windrowerEffects = g_effectManager:loadEffect(self.xmlFile,
    string.format("%s.windrowerEffect", key), self.components, self,
    self.i3dMappings)
```

```
172  entry.animationNodes =
     g_animationManager:loadAnimations(self.xmlFile,
     key..".animationNodes", self.components, self, self.i3dMappings)
173
174  table.insert(spec.workModes, entry)
175  i = i + 1
176  end
177
178  spec.stateMax = table.getn(spec.workModes)
179  if spec.stateMax > ((2^WorkMode.WORKMODE_SEND_NUM_BITS) - 1) then
180  print("Error: WorkMode only supports
     "..((2^WorkMode.WORKMODE_SEND_NUM_BITS) - 1).." modes!")
181  end
182
183  if spec.stateMax > 0 then
184  self:setWorkMode(1, true)
185  end
186
187  spec.accumulatedFruitType = FruitType.UNKNOWN
188  spec.dirtyFlag = self:getNextDirtyFlag()
189  end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
234  function WorkMode:onReadStream(streamId, connection)
235  local spec = self.spec_workMode
236  if spec.stateMax == 0 then
237  return
238  end
239
240  local state = streamReadUIntN(streamId,
     WorkMode.WORKMODE_SEND_NUM_BITS)
241  self:setWorkMode(state, true)
242  end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
248 function WorkMode:onWriteStream(streamId, connection)
249 local spec = self.spec_workMode
250 if spec.stateMax == 0 then
251 return
252 end
253
254 streamWriteUIntN(streamId, spec.state,
    WorkMode.WORKMODE_SEND_NUM_BITS)
255 end
```

onReadUpdateStream

Description

Called on on update

Definition

onReadUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId stream ID

integer timestamp timestamp

table connection connection

Code

```
262 function WorkMode:onReadUpdateStream(streamId, timestamp,
    connection)
263 if connection:getIsServer() then
264 if streamReadBool(streamId) then
265 local spec = self.spec_workMode
266
267 local mode = spec.workModes[spec.state]
268 for _, effect in ipairs(mode.windrowerEffects) do
269 if streamReadBool(streamId) then
270 effect.lastChargeTime = g_currentMission.time
271 end
272 end
273
274 spec.accumulatedFruitType = streamReadUIntN(streamId, 6)
275 end
276 end
```

```
277 end
```

onWriteUpdateStream

Description

Called on on update

Definition

onWriteUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Code

```
284 function WorkMode:onWriteUpdateStream(streamId, connection, dirtyMask)
285 if not connection:getIsServer() then
286 local spec = self.spec_workMode
287 if streamWriteBool(streamId, bitAND(dirtyMask, spec.dirtyFlag) ~= 0) then
288 local mode = spec.workModes[spec.state]
289
290 for _, effect in ipairs(mode.windrowerEffects) do
291 streamWriteBool(streamId, effect.lastChargeTime + 500 > g_currentMission.time)
292 end
293
294 streamWriteUIntN(streamId, spec.accumulatedFruitType, 6)
295 end
296 end
297 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt)

Arguments

float dt	time since last call in ms

Code

```
302 function WorkMode:onUpdate(dt, isActiveForInput, isSelected)
303 local spec = self.spec_workMode
304 if spec.stateMax == 0 then
305 return
306 end
307
```

```
308 if self.isClient then
309 local mode = spec.workModes[spec.state]
310
311 for _, turnedOnAnimation in pairs(mode.turnedOnAnimations) do
312 if turnedOnAnimation.speedDirection ~= 0 then
313 local duration = turnedOnAnimation.turnOnFadeTime
314 if turnedOnAnimation.speedDirection == -1 then
315 duration = turnedOnAnimation.turnOffFadeTime
316 end
317 turnedOnAnimation.currentSpeed = MathUtil.clamp(turnedOnAnimation.curren
    turnedOnAnimation.speedDirection * dt/duration, 0, 1)
318 self:setAnimationSpeed(turnedOnAnimation.name,
    turnedOnAnimation.currentSpeed*turnedOnAnimation.speedScale)
319 if turnedOnAnimation.speedDirection == -1 and turnedOnAnimation.currentS
    then
320 self:stopAnimation(turnedOnAnimation.name, true)
321 end
322
323 if turnedOnAnimation.currentSpeed == 1 or turnedOnAnimation.currentSpeed
324 turnedOnAnimation.speedDirection = 0
325 end
326 end
327 end
328
329 for _, effect in pairs(mode.windrowerEffects) do
330 if effect.lastChargeTime + 500 > g_currentMission.time then
331 local fillType =
    g_fruitTypeManager:getWindrowFillTypeIndexByFruitTypeIndex(spec.accumula
332 if fillType ~= nil then
333 effect:setFillType(fillType)
334 if not effect:isRunning() then
335 g_effectManager:startEffect(effect)
336 end
337 end
338 else
339 if effect.turnOffRequiredEffect == 0 or (effect.turnOffRequiredEffect ~=
    mode.windrowerEffects[effect.turnOffRequiredEffect]:isRunning()) then
340 g_effectManager:stopEffect(effect)
341 end
342 end
343 end
```

```
344 end
345
346 if self.isServer then
347 local mode = spec.workModes[spec.state]
348
349 local fruitType
350 local workAreaCharge
351 for _, area in ipairs(mode.workAreas) do
352 local workArea = self.spec_workArea.workAreas[area.workAreaIndex]
353 if workArea ~= nil then
354 if workArea.lastValidPickupFruitType ~= FruitType.UNKNOWN then
355 fruitType = workArea.lastValidPickupFruitType
356 end
357 workAreaCharge = workAreaCharge or workArea.lastPickupLiters ~= 0
358 end
359 end
360
361 if fruitType ~= nil then
362 if fruitType ~= spec.accumulatedFruitType then
363 spec.accumulatedFruitType = fruitType
364 self:raiseDirtyFlags(spec.dirtyFlag)
365 end
366 end
367
368 for _, effect in pairs(mode.windrowerEffects) do
369 if workAreaCharge then
370 effect.lastChargeTime = g_currentMission.time
371 self:raiseDirtyFlags(spec.dirtyFlag)
372 end
373 end
374 end
375 end
```

onDraw

Description

Called on draw

Definition

onDraw(boolean isActiveForInput, boolean isSelected)

Arguments

boolean isActiveForInput	true if vehicle is active for input
boolean isSelected	true if vehicle is selected

Code

```
421 function WorkMode:onDraw(isActiveForInput, isSelected)
422 local spec = self.spec_workMode
423 if spec.stateMax == 0 then
424 return
425 else
426 local mode = spec.workModes[spec.state]
427 g_currentMission:addExtraPrintText(string.format(g_i18n:getText("action_
    mode.name))
428
429 local allowWorkModeChange = self:getIsWorkModeChangeAllowed()
430
431 local actionEvent = spec.actionEvents[InputAction.TOGGLE_WORKMODE]
432 if actionEvent ~= nil then
433 g_inputBinding:setActionEventActive(actionEvent.actionEventId, allowWork
434 end
435
436 for _, workMode in ipairs(spec.workModes) do
437 if workMode.inputAction ~= nil then
438 actionEvent = spec.actionEvents[workMode.inputAction]
439 if actionEvent ~= nil then
440 g_inputBinding:setActionEventActive(actionEvent.actionEventId, allowWork
441 end
442 end
443 end
444 end
445 end
```

onDeactivate

Description

Called on deactivate

Definition

onDeactivate()

Code

```
449 function WorkMode:onDeactivate()
450 WorkMode.deactivateWindrowerEffects(self)
451 end
```

deactivateWindrowerEffects

Description

Called on deactivate

Definition

deactivateWindrowerEffects()

Code

```
455 function WorkMode:deactivateWindrowerEffects()
456 if self.isClient then
457 local spec = self.spec_workMode
458 for _, mode in pairs(spec.workModes) do
459 if mode.windrowerEffects ~= nil then
460 g_effectManager:stopEffects(mode.windrowerEffects)
461 end
462 end
463 end
464 end
```

setWorkMode

Description

Change work mode

Definition

setWorkMode(integer state, boolean noEventSend)

Arguments

integer	state	new state
boolean	noEventSend	no event send

Code

```
470 function WorkMode:setWorkMode(state, noEventSend)
471 local spec = self.spec_workMode
472
473 if noEventSend == nil or noEventSend == false then
474 if g_server ~= nil then
475 g_server:broadcastEvent(SetWorkModeEvent:new(self, state), nil,
    nil, self)
476 else
477 g_client:getServerConnection():sendEvent(SetWorkModeEvent:new(self,
    state))
478 end
479 end
480
481 if state ~= spec.state then
482 local currentMode = spec.workModes[spec.state]
483 if currentMode.animations ~= nil then
484 for _,anim in pairs(currentMode.animations) do
485 local curTime = self:getAnimationTime(anim.animName)
486 if anim.stopTime == nil then
487 self:playAnimation(anim.animName, -anim.animSpeed, curTime,
    noEventSend)
488 end
489 end
```

```
490 g_animationManager:stopAnimations(currentMode.animationNodes)
491 end
492
493 local newMode = spec.workModes[state]
494 if newMode.animations ~= nil then
495 for _,anim in pairs(newMode.animations) do
496 local curTime = self:getAnimationTime(anim.animName)
497 if anim.stopTime ~= nil then
498 self:setAnimationStopTime(anim.animName, anim.stopTime)
499 local speed = 1.0
500 if curTime > anim.stopTime then
501 speed = -1.0
502 end
503 self:playAnimation(anim.animName, speed, curTime, noEventSend)
504 else
505 self:playAnimation(anim.animName, anim.animSpeed, curTime,
    noEventSend)
506 end
507 end
508 if self:getIsTurnedOn() then
509 g_animationManager:stopAnimations(newMode.animationNodes)
510 end
511 end
512
513 for _, effect in pairs(currentMode.windrowerEffects) do
514 g_effectManager:stopEffect(effect)
515 end
516 end
517
518 local workAreaSpec = self.spec_workArea
519 if workAreaSpec ~= nil then
520 local workAreas = workAreaSpec.workAreas
521 for _, workArea in pairs(spec.workModes[state].workAreas) do
522 local workAreaToSet = workAreas[workArea.workAreaIndex]
523 if workAreaToSet ~= nil then
524 workAreaToSet.dropWindrowWorkAreaIndex = workArea.dropAreaIndex --
    windrower
525 workAreaToSet.dropAreaIndex = workArea.dropAreaIndex -- mower
526 end
527 end
528 end
```

```
529
530  spec.state = state
531  end
```

getIsWorkModeChangeAllowed

Description

Returns if work mode change is allowed

Definition

getIsWorkModeChangeAllowed()

Return Values

boolean isAllowed is allowed

Code

```
557  function WorkMode:getIsWorkModeChangeAllowed()
558  local spec = self.spec_workMode
559
560  if self.getFoldAnimTime ~= nil then
561  if self:getFoldAnimTime() > spec.foldMaxLimit or
     self:getFoldAnimTime() < spec.foldMinLimit then
562  return false
563  end
564  end
565
566  if not spec.allowChangeOnLowered then
567  local attacherVehicle = self:getAttacherVehicle()
568  if attacherVehicle ~= nil then
569  local index =
     attacherVehicle:getAttacherJointIndexFromObject(self)
570  local attacherJoint =
     attacherVehicle:getAttacherJointByJointDescIndex(index)
571
572  if attacherJoint.moveDown then
573  return false
574  end
575  end
576  end
577
578  return true
579  end
```

onTurnedOff

Description

Called on turn off

Definition

onTurnedOff(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
584  function WorkMode:onTurnedOff()
585  local spec = self.spec_workMode
586  if spec.stateMax == 0 then
587  return
588  end
589
590  if self.isClient then
591  WorkMode.deactivateWindrowerEffects(self)
592  local mode = spec.workModes[spec.state]
593
594  for _, turnedOnAnimation in pairs(mode.turnedOnAnimations) do
595  turnedOnAnimation.speedDirection = -1
596  end
597  g_animationManager:stopAnimations(mode.animationNodes)
598  end
599  end
```

onTurnedOn

Description

Called on turn on

Definition

onTurnedOn(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
604  function WorkMode:onTurnedOn()
605  local spec = self.spec_workMode
606  if spec.stateMax == 0 then
607  return
608  end
609
610  if self.isClient then
611  local mode = spec.workModes[spec.state]
612
613  for _, turnedOnAnimation in pairs(mode.turnedOnAnimations) do
614  turnedOnAnimation.speedDirection = 1
615  self:playAnimation(turnedOnAnimation.name,
     turnedOnAnimation.currentSpeed*turnedOnAnimation.speedScale,
     self:getAnimationTime(turnedOnAnimation.name), true)
616  end
```

```
617 g_animationManager:startAnimations(mode.animationNodes)
618 end
619 end
```

onSetLowered

Description

Called on change lowering state

Definition

onSetLowered(boolean lowered)

Arguments

boolean lowered attachable is lowered

Code

```
624 function WorkMode:onSetLowered(lowered)
625 local spec = self.spec_workMode
626 if spec.stateMax == 0 then
627 return
628 end
629
630 if self.getFoldAnimTime ~= nil then
631 local foldAnimTime = self:getFoldAnimTime()
632 if foldAnimTime ~= 1 and foldAnimTime ~= 0 and foldAnimTime ~=
    self.foldMiddleAnimTime then
633 spec.playDelayedLoweringAnimation = lowered
634 return
635 end
636 end
637
638 local mode = spec.workModes[spec.state]
639
640 for _, loweringAnimation in pairs(mode.loweringAnimations) do
641 if lowered then
642 if self:getAnimationTime(loweringAnimation.name) < 1 then
643 self:playAnimation(loweringAnimation.name,
    loweringAnimation.speed, nil, true)
644 end
645 else
646 if self:getAnimationTime(loweringAnimation.name) > 0 then
647 self:playAnimation(loweringAnimation.name, -
    loweringAnimation.speed, nil, true)
648 end
649 end
650 end
```

```
651 end
```

onFoldStateChanged

Description

Called on fold state change

Definition

onFoldStateChanged(integer direction)

Arguments

integer direction direction of folding

Code

```
656 function WorkMode:onFoldStateChanged(direction, moveToMiddle)
657 local spec = self.spec_workMode
658 if spec.stateMax == 0 then
659 return
660 end
661
662 if direction > 0 then
663 local mode = spec.workModes[spec.state]
664
665 for _, anim in pairs(mode.animations) do
666 if anim.repeatAfterUnfolding then
667 anim.repeated = false
668 end
669 end
670 end
671 end
```

AnimationManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
18 function AnimationManager:new(customMt)
19 local self = AbstractManager:new(customMt or AnimationManager_mt)
20
21 return self
22 end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
26  function AnimationManager:initDataStructures()
27  self.runningAnimations = {}
28  self.registeredAnimationClasses = {}
29  end
```

AITurnStrategy

Description

validateCollisionBox

Description

checks if calculation of box is valid (check for NAN)

Definition

validateCollisionBox()

HelperManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
18  function HelperManager:new(customMt)
19  local self = AbstractManager:new(customMt or HelperManager_mt)
20  return self
21  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
25  function HelperManager:initDataStructures()
26  self.numHelpers = 0
27  self.helpers = {}
28  self.nameToIndex = {}
29  self.indexToHelper = {}
30  self.availableHelpers = {}
31  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
42  function HelperManager:loadMapData(xmlFile, missionInfo,
    baseDirectory)
43  HelperManager:superClass().loadMapData(self)
44
45  self:loadDefaultTypes()
46  return XMLUtil.loadDataFromMapXML(xmlFile, "helpers",
    baseDirectory, self, self.loadHelpers, missionInfo, baseDirectory)
47  end
```

loadHelpers

Description

Load data on map load

Definition

loadHelpers()

Return Values

boolean true if loading was successful else false

Code

```
52  function HelperManager:loadHelpers(xmlFile, missionInfo,
    baseDirectory, isBaseType)
53  local i = 0
54  while true do
55  local key = string.format("map.helpers.helper(%d)", i)
56  if not hasXMLProperty(xmlFile, key) then
57  break
58  end
59
60  local name = getXMLString(xmlFile, key.."#name")
61  local title = getXMLString(xmlFile, key.."#title")
62  local filename = getXMLString(xmlFile, key.."#filename")
63
64  self:addHelper(name, title, filename, baseDirectory, isBaseType)
65
66  i = i + 1
67  end
68
```

```
69  return true
70  end
```

addHelper

Description

Adds a new helper

Definition

addHelper(string name, string filename, string baseDir)

Arguments

string name	helper index name
string filename	helper config filename
string baseDir	the base directory

Return Values

boolean true if added successful else false

Code

```
78  function HelperManager:addHelper(name, title, filename, baseDir,
    isBaseType)
79  if not ClassUtil.getIsValidIndexName(name) then
80  print("Warning: '"..tostring(name).."' is not a valid name for a
    helper. Ignoring helper!")
81  return nil
82  end
83
84  name = name:upper()
85
86  if isBaseType and self.nameToIndex[name] ~= nil then
87  print("Warning: Helper '"..tostring(name).."' already exists.
    Ignoring helper!")
88  return nil
89  end
90
91  local helper = self.helpers[name]
92  if helper == nil then
93  if filename == nil or filename == "" then
94  print("Warning: Missing helper config file for helper
    '"..tostring(name).."'. Ignoring helper!")
95  return nil
96  end
97
98  self.numHelpers = self.numHelpers + 1
99
100 helper = {}
101 helper.name = name
```

```
102  helper.index = self.numHelpers
103  helper.title = name
104  if title ~= nil then
105  helper.title = g_i18n:convertText(title)
106  end
107  helper.filename = Utils.getFilename(filename, baseDir)
108
109  self.helpers[name] = helper
110  self.nameToIndex[name] = self.numHelpers
111  self.indexToHelper[self.numHelpers] = helper
112  table.insert(self.availableHelpers, helper)
113  else
114  if title ~= nil then
115  helper.title = g_i18n:convertText(title)
116  end
117  if filename ~= nil then
118  helper.filename = Utils.getFilename(filename, baseDir)
119  end
120  end
121
122  return helper
123  end
```

getRandomHelper

Description

Gets a random helper

Definition

getRandomHelper()

Return Values

table helper a random helper object

Code

```
128  function HelperManager:getRandomHelper()
129  return self.availableHelpers[math.random(1,
     #self.availableHelpers)]
130  end
```

getRandomIndex

Description

Gets a random helper index

Definition

getRandomIndex()

Return Values

integer helperIndex a random helper index

Code

```
135  function HelperManager:getRandomIndex()
136  return math.random(1, self.numHelpers)
137  end
```

getHelperByIndex

Description

Gets a helper by index

Definition

getHelperByIndex(integer index)

Arguments

integer index the helper index

Return Values

table helper the helper object

Code

```
143  function HelperManager:getHelperByIndex(index)
144  if index ~= nil then
145  return self.indexToHelper[index]
146  end
147  return nil
148  end
```

getHelperByName

Description

Gets a helper by index name

Definition

getHelperByName(string name)

Arguments

string name the helper index name

Return Values

table helper the helper object

Code

```
154  function HelperManager:getHelperByName(name)
155  if name ~= nil then
156  name = name:upper()
157  return self.helpers[name]
158  end
159  return nil
160  end
```

useHelper

Description

Marks a helper as 'in use'

Definition

useHelper(table helper)

Arguments

table helper the helper object

Return Values

boolean success true if helper is marked else false

Code

```
166  function HelperManager:useHelper(helper)
167  for k, h in pairs(self.availableHelpers) do
168  if h == helper then
169  table.remove(self.availableHelpers, k)
170  return true
171  end
172  end
173  return false
174  end
```

releaseHelper

Description

Marks a helper as 'not in use'

Definition

releaseHelper(table helper)

Arguments

table helper the helper object

Code

```
179  function HelperManager:releaseHelper(helper)
180  table.insert(self.availableHelpers, helper)
181  end
```

getNumOfHelpers

Description

Gets number of helpers

Definition

getNumOfHelpers()

Return Values

integer numOfHelpers total number of helpers

Code

```
186  function HelperManager:getNumOfHelpers()
187  return self.numHelpers
188  end
```

NPCManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
18  function NPCManager:new(customMt)
19  local self = AbstractManager:new(customMt or NPCManager_mt)
20  return self
21  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
25  function NPCManager:initDataStructures()
26  self.numNpcs = 0
27  self.npcs = {}
28  self.nameToIndex = {}
29  self.indexToNpc = {}
30  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
41  function NPCManager:loadMapData(xmlFile, missionInfo,
    baseDirectory)
42  NPCManager:superClass().loadMapData(self)
43
44  self:loadDefaultTypes(missionInfo, baseDirectory)
45  return XMLUtil.loadDataFromMapXML(xmlFile, "npcs", baseDirectory,
    self, self.loadNPCs, missionInfo, baseDirectory)
46  end
```

loadNPCs

Description

Load data on map load

Definition

loadNPCs()

Return Values

boolean true if loading was successful else false

Code

```lua
function NPCManager:loadNPCs(xmlFile, missionInfo, baseDirectory, isBaseType)

local i = 0
while true do
local key = string.format("map.npcs.npc(%d)", i)
if not hasXMLProperty(xmlFile, key) then
break
end

local name = getXMLString(xmlFile, key.."#name")
local title = getXMLString(xmlFile, key.."#title")
local category = getXMLString(xmlFile, key.."#category")
local imageFilename = getXMLString(xmlFile, key.."#imageFilename")

self:addNPC(name, title, category, imageFilename, baseDirectory, isBaseType)

i = i + 1
end

return true
end
```

saveToXMLFile

Description

Write data to savegame file

Definition

saveToXMLFile(string xmlFilename)

Arguments

string xmlFilename file path

Return Values

boolean true if loading was successful else false

Code

```lua
function NPCManager:saveToXMLFile(xmlFilename)
-- save npcs to xml
local xmlFile = createXMLFile("npcsXML", xmlFilename, "npcs")
if xmlFile ~= nil then
for k, npc in ipairs(self.indexToNpc) do
local npcKey = string.format("npcs.npc(%d)", k-1)
setXMLString(xmlFile, npcKey.."#name", npc.name)
setXMLInt(xmlFile, npcKey.."#finishedMissions", npc.finishedMissions)
```

```
85  end
86
87  saveXMLFile(xmlFile)
88  delete(xmlFile)
89
90  return true
91  end
92
93  return false
94  end
```

loadFromXMLFile

Description

Load data from xml savegame file

Definition

loadFromXMLFile(string filename)

Arguments

string filename xml filename

Code

```
 99  function NPCManager:loadFromXMLFile(xmlFilename)
100  if xmlFilename == nil then
101  return false
102  end
103
104  local xmlFile = loadXMLFile("npcXML", xmlFilename)
105
106  if xmlFile == 0 then
107  return false
108  end
109
110  local i = 0
111  while true do
112  local key = string.format("npcs.npc(%d)", i)
113  if not hasXMLProperty(xmlFile, key) then
114  break
115  end
116
117  local name = getXMLString(xmlFile, key.."#name")
118  local npc = self:getNPCByName(name)
119  if npc ~= nil then
120  npc.finishedMissions = Utils.getNoNil(getXMLInt(xmlFile,
     key.."#finishedMissions"), 0)
```

```
121 else
122 print("Warning: Npc '"..tostring(name).."' not found!")
123 end
124 i = i + 1
125 end
126
127 delete(xmlFile)
128
129 return true
130 end
```

addNPC

Description

Adds a new npc

Definition

addNPC(string name, string title, string category, string imageFilename, string baseDir)

Arguments

string name	npc index name
string title	npc real name
string category	npc category (e.g. young, middle, old, female)
string imageFilename	npc image filename
string baseDir	the base directory

Return Values

boolean true if added successful else false

Code

```
140 function NPCManager:addNPC(name, title, category, imageFilename,
    baseDir, isBaseType)
141 if not ClassUtil.getIsValidIndexName(name) then
142 print("Warning: '"..tostring(name).."' is not a valid name for a
    npc. Ignoring npc!")
143 return nil
144 end
145
146 name = name:upper()
147
148 if isBaseType and self.nameToIndex[name] ~= nil then
149 print("Warning: NPC '"..tostring(name).."' already exists.
    Ignoring npc!")
150 return nil
151 end
152
153 local npc = self.npcs[name]
154 if npc == nil then
```

```
155 if title == nil or title == "" then
156 print("Warning: '"..tostring(title).."' is not a valid title for
    a npc. Ignoring npc!")
157 return nil
158 end
159 if category == nil or category == "" then
160 print("Warning: '"..tostring(category).."' is not a valid
    category for a npc. Ignoring npc!")
161 return nil
162 end
163 if imageFilename == nil or imageFilename == "" then
164 print("Warning: Missing npc image file for npc
    '"..tostring(name).."'. Ignoring npc!")
165 return nil
166 end
167
168 self.numNpcs = self.numNpcs + 1
169 npc = {}
170 npc.name = name
171 npc.title = g_i18n:convertText(title)
172 npc.index = self.numNpcs
173 npc.imageFilename = Utils.getFilename(imageFilename, baseDir)
174 npc.finishedMissions = 0
175
176 self.npcs[name] = npc
177 self.nameToIndex[name] = self.numNpcs
178 self.indexToNpc[self.numNpcs] = npc
179 else
180 if title ~= nil and title ~= "" then
181 npc.title = g_i18n:convertText(title)
182 end
183 if imageFilename ~= nil and imageFilename ~= "" then
184 npc.imageFilename = Utils.getFilename(imageFilename, baseDir)
185 end
186 end
187
188 return npc
189 end
```

getRandomNPC

Description

Gets a random npc

Definition

getRandomNPC()

Return Values

table npc a random npc object

Code

```
194  function NPCManager:getRandomNPC()
195  return self.indexToNpc[self:getRandomIndex()]
196  end
```

getRandomIndex

Description

Gets a random npc index

Definition

getRandomIndex()

Return Values

integer npcIndex a random npc index

Code

```
201  function NPCManager:getRandomIndex()
202  return math.random(1, self.numNpcs)
203  end
```

getNPCByIndex

Description

Gets a npc by index

Definition

getNPCByIndex(integer index)

Arguments

integer index the npc index

Return Values

table npc the npc object

Code

```
209  function NPCManager:getNPCByIndex(index)
210  if index ~= nil then
211  return self.indexToNpc[index]
212  end
213  return nil
214  end
```

getNPCByName

Description

Gets a npc by index name

Definition

getNPCByName(string name)

Arguments

string name the npc index name

Return Values

table npc the npc object

Code

```
220  function NPCManager:getNPCByName(name)
221  if name ~= nil then
222  name = name:upper()
223  return self.npcs[name]
224  end
225  return nil
226  end
```

I3DManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of basket trigger object

table instance instance of object

Code

```
17  function I3DManager:new(customMt)
18  local self = AbstractManager:new(customMt or I3DManager_mt)
19
20  return self
21  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

boolean success success

Code

```
25  function I3DManager:initDataStructures()
26  self.sharedI3DFiles = {}
27  self.sharedI3DFilesPendingCallbacks = {}
28  end
```

loadSharedI3DFile

Description

Loads an i3D file. A cache system is used for faster loading

Definition

loadSharedI3DFile(string filename, string baseDir, boolean callOnCreate, boolean addToPhysics, boolean verbose, function asyncCallbackFunction, table asyncCallbackObject, table asyncCallbackArguments)

Arguments

string	filename	filename
string	baseDir	baseDir
boolean	callOnCreate	true if onCreate i3d callbacks should be called
boolean	addToPhysics	true if collisions should be added to physics
boolean	verbose	verbose
function	asyncCallbackFunction	a callback function
table	asyncCallbackObject	callback function target object
table	asyncCallbackArguments	a list of arguments

Return Values

table	instance	instance
integer	id	i3d rootnode

loadSharedI3DFileFinished

Description

Called once i3d loading is finished

Definition

loadSharedI3DFileFinished(integer nodeId, table arguments)

Arguments

integer	nodeId	i3d node id
table	arguments	a list of arguments

Return Values

boolean	isActivateable	is activateable

fillSharedI3DFileCache

Description

Adds an i3d file to cache

Definition

fillSharedI3DFileCache(string filename, string baseDir)

Arguments

string	filename	filename
string	baseDir	baseDir

Return Values

table	instance	instance of basket trigger object

releaseSharedI3DFile

Description

Releases one instance. If autoDelete is true and instance count <= 0. I3d will be removed from cache

Definition

releaseSharedI3DFile(string filename, string baseDir, boolean autoDelete)

Arguments

string	filename	filename
string	baseDir	baseDir

boolean autoDelete true if file should be removed from cache if instance count is <= 0

Return Values

boolean success success

deleteSharedI3DFiles

Description

Clears i3d cache

Definition

deleteSharedI3DFiles()

Return Values

table instance instance of object

Chainsaw

Description

Class for chainsaws

new

Description

Creating chainsaw object

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean isServer is server

boolean isClient is client

table customMt custom metatable

Return Values

table instance instance of object

table instance Instance of object

Code

```
35  function Chainsaw:new(isServer, isClient, customMt)
36  local self = HandTool:new(isServer, isClient, customMt or
    Chainsaw_mt)
37  return self
38  end
```

load

Description

Load chainsaw from xml file

Definition

load(string xmlFilename, table player)

Arguments

string xmlFilename xml file name

table player player

Return Values

bool true if player can idle

boolean success success

Code

```
45  function Chainsaw:load(xmlFilename, player)
46  if not Chainsaw:superClass().load(self, xmlFilename, player) then
47  return false
48  end
49
50  local xmlFile = loadXMLFile("TempXML", xmlFilename)
51
52  self.rotateInput = 0
53  self.activatePressed = false
54  self.eventIdRotateHandtool = ""
55
56  self.rotationZ = 0
57  self.rotationSpeedZ = 0.003
58  self.cutSizeY = 1.1
59  self.cutSizeZ = 1.0
60  self.isCutting = false
61  self.waitingForResetAfterCut = false
62  self.cutNode = getChildAt(self.rootNode, 0)
63  self.graphicsNode = getChildAt(self.cutNode, 0)
64  self.chainNode = getChildAt(self.graphicsNode, 0)
65  self.psNode = getChildAt(self.graphicsNode, 1)
66  self.cutPositionNode = getChildAt(self.graphicsNode, 5)
67
68  self.pricePerSecond = Utils.getNoNil(getXMLFloat(xmlFile,
    "handTool.chainsaw.pricePerMinute"), 50) / 1000
69  self.quicktapThreshold = Utils.getNoNil(getXMLFloat(xmlFile,
    "handTool.chainsaw#quicktapThreshold"), 0.0) * 1000
70  if self.isClient then
71  self.particleSystems = {}
72
73  local i = 0
74  while true do
75  local keyPS =
    string.format("handTool.chainsaw.particleSystems.emitterShape(%d)",
    i)
76  if not hasXMLProperty(xmlFile, keyPS) then
77  break
78  end
79  local emitterShape = I3DUtil.indexToObject(self.rootNode,
    getXMLString(xmlFile, keyPS.."#node"))
80  local particleType = getXMLString(xmlFile, keyPS.."#particleType")
```

```
81  if emitterShape ~= nil then
82  local fillType = FillType.WOODCHIPS
83  local particleSystem =
    g_particleSystemManager:getParticleSystem(fillType, particleType)
84  if particleSystem ~= nil then
85  table.insert(self.particleSystems,
    ParticleUtil.copyParticleSystem(xmlFile, keyPS, particleSystem,
    emitterShape))
86  end
87  end
88  i = i + 1
89  end
90
91  if #self.particleSystems == 0 then
92  self.particleSystems = nil
93  end
94
95  self.handNode = Utils.getNoNil(I3DUtil.indexToObject(self.rootNode,
    getXMLString(xmlFile, "handTool.chainsaw.handNode#node")),
    self.rootNode)
96  self.handNodeRotation =
    StringUtil.getRadiansFromString(Utils.getNoNil(getXMLString(xmlFile,
    "handTool.chainsaw.handNode#rotation"), "0 0 0"), 3)
97
98  self.equipmentUVs =
    StringUtil.getVectorNFromString(Utils.getNoNil(getXMLString(xmlFile,
    "handTool.chainsaw.equipment#uvs"), "0 0"), 2)
99
100 self.chains = g_animationManager:loadAnimations(xmlFile,
    "handTool.chainsaw.chain", self.rootNode, self, nil)
101 self.samples = {}
102 self.samples.start = g_soundManager:loadSampleFromXML(xmlFile,
    "handTool.chainsaw.sounds", "start", self.baseDirectory,
    self.rootNode, 1, AudioGroup.VEHICLE, nil, nil)
103 self.samples.stop = g_soundManager:loadSampleFromXML(xmlFile,
    "handTool.chainsaw.sounds", "stop", self.baseDirectory,
    self.rootNode, 1, AudioGroup.VEHICLE, nil, nil)
104 self.samples.idle = g_soundManager:loadSampleFromXML(xmlFile,
    "handTool.chainsaw.sounds", "idle", self.baseDirectory,
    self.rootNode, 0, AudioGroup.VEHICLE, nil, nil)
105 self.samples.cutStart = g_soundManager:loadSampleFromXML(xmlFile,
    "handTool.chainsaw.sounds", "cutStart", self.baseDirectory,
    self.rootNode, 1, AudioGroup.VEHICLE, nil, nil)
```

```
106 self.samples.cutStop = g_soundManager:loadSampleFromXML(xmlFile, "handTool.chainsaw.sounds", "cutStop", self.baseDirectory, self.rootNode, 1, AudioGroup.VEHICLE, nil, nil)
107 self.samples.cutLoop = g_soundManager:loadSampleFromXML(xmlFile, "handTool.chainsaw.sounds", "cutLoop", self.baseDirectory, self.rootNode, 0, AudioGroup.VEHICLE, nil, nil)
108 self.samples.activeStart = g_soundManager:loadSampleFromXML(xmlFile, "handTool.chainsaw.sounds", "activeStart", self.baseDirectory, self.rootNode, 1, AudioGroup.VEHICLE, nil, nil)
109 self.samples.activeStop = g_soundManager:loadSampleFromXML(xmlFile, "handTool.chainsaw.sounds", "activeStop", self.baseDirectory, self.rootNode, 1, AudioGroup.VEHICLE, nil, nil)
110 self.samples.activeLoop = g_soundManager:loadSampleFromXML(xmlFile, "handTool.chainsaw.sounds", "activeLoop", self.baseDirectory, self.rootNode, 0, AudioGroup.VEHICLE, nil, nil)
111 self.samplesQuicktap = {}
112 local j = 0
113 while true do
114 local sampleQuicktap = g_soundManager:loadSampleFromXML(xmlFile, "handTool.chainsaw.sounds.quickTapSounds", string.format("quickTap(%d)", j), self.baseDirectory, self.rootNode, 1, AudioGroup.VEHICLE, nil, nil)
115 if sampleQuicktap == nil then
116 break
117 end
118 table.insert(self.samplesQuicktap, sampleQuicktap)
119 j = j + 1
120 end
121 self.samplesQuicktapCount = j
122
123 self.samplesTree = {}
124 self.samplesTree.cut = g_soundManager:loadSampleFromXML(xmlFile, "handTool.chainsaw.treeSounds", "cut", self.baseDirectory, self.rootNode, 1, AudioGroup.VEHICLE, nil, nil)
125 --self.samplesTree.falling = g_soundManager:loadSampleFromXML(xmlFile, "handTool.chainsaw.treeSounds", "falling", self.baseDirectory, self.rootNode, 1, AudioGroup.VEHICLE, nil, nil)
126 self.samplesBranch = {}
127 local k = 0
128 while true do
129 local sampleBranch = g_soundManager:loadSampleFromXML(xmlFile, "handTool.chainsaw.branchSounds", string.format("branch(%d)", k), self.baseDirectory, self.rootNode, 1, AudioGroup.VEHICLE, nil, nil)
130 if sampleBranch == nil then
```

```
131 break
132 end
133 table.insert(self.samplesBranch, sampleBranch)
134 k = k + 1
135 end
136 self.samplesBranchCount = k
137 self.samplesBranchActiveTimer = 0
138
139 self.samplesTreeLinkNode = createTransformGroup("cutSoundLinkNode")
140 link(self.cutNode, self.samplesTreeLinkNode)
141
142 if self.samplesTree.cut ~= nil and self.samplesTree.cut.soundNode ~=
    nil then
143 link(self.samplesTreeLinkNode, self.samplesTree.cut.soundNode)
144 --link(self.samplesTreeLinkNode, self.samplesTree.falling.soundNode)
145 end
146
147 self.soundFSM = FSMUtil.create()
148 self.soundFSM:addState(Chainsaw.SOUND_STATES.START,
    ChainsawSoundStateStart:new(Chainsaw.SOUND_STATES.START, self,
    self.soundFSM))
149 self.soundFSM:addState(Chainsaw.SOUND_STATES.STOP,
    ChainsawSoundStateStop:new(Chainsaw.SOUND_STATES.STOP, self,
    self.soundFSM))
150 self.soundFSM:addState(Chainsaw.SOUND_STATES.IDLE,
    ChainsawSoundStateIdle:new(Chainsaw.SOUND_STATES.IDLE, self,
    self.soundFSM))
151 self.soundFSM:addState(Chainsaw.SOUND_STATES.ACTIVE,
    ChainsawSoundStateActive:new(Chainsaw.SOUND_STATES.ACTIVE, self,
    self.soundFSM))
152 self.soundFSM:addState(Chainsaw.SOUND_STATES.CUT,
    ChainsawSoundStateCut:new(Chainsaw.SOUND_STATES.CUT, self,
    self.soundFSM))
153 self.soundFSM:addState(Chainsaw.SOUND_STATES.QUICKTAP,
    ChainsawSoundStateQuicktap:new(Chainsaw.SOUND_STATES.QUICKTAP, self,
    self.soundFSM))
154
155 local filename = getXMLString(xmlFile,
    "handTool.chainsaw.ringSelector#file")
156 if filename ~= nil then
157 local i3dNode = g_i3DManager:loadSharedI3DFile(filename,
    self.baseDirectory, false, false, false)
158 if i3dNode ~= 0 then
159 self.ringSelectorFilename = filename
```

```
160  self.ringSelector = getChildAt(i3dNode, 0)
161  self.ringSelectorScaleOffset = Utils.getNoNil(getXMLFloat(xmlFile,
     "handTool.chainsaw.ringSelector#scaleOffset"), 0.3)
162  setVisibility(self.ringSelector, false)
163  link(player.chainsawSplitShapeFocus, self.ringSelector)
164  delete(i3dNode)
165  end
166  end
167  end
168
169  self.lastWorkTime = 0
170  self.maxWorkTime = 300
171
172  self.moveSpeedY = 0.0001
173  self.speedFactor = 0
174  self.defaultCutDuration = 8.0 -- in s
175  self.maxTrunkWidthSq = 1.0 -- in m
176  self.outDuration = 0.15 -- in s
177  self.inDuration = 0.15 -- in s
178  self.cutTimer = 0.0 -- in s
179  self.outTimer = self.outDuration -- in s
180  self.transitionAlpha = 0.0 -- [0,1]
181  self.cameraTransitionState = Chainsaw.CAMERA_TRANSITION_STATES.NONE
     -- 0=in 1=cut 2=out
182  self.minRotationZ = math.rad(90) -- in rad
183  self.maxRotationZ = math.rad(-90) -- in rad
184  self.maxModelTranslation = 0.0 -- in m
185  self.cutFocusDistance = -1.0
186  self.startCameraDirectionY = { 0, 1, 0 }
187  self.startCameraDirectionZ = { 0, 0, 1 }
188  self.endCameraDirectionY = { 0, 1, 0 }
189  self.endCameraDirectionZ = { 0, 0, 1 }
190  self.startChainsawPosition = { 0, 0, 0 }
191  self.endChainsawPosition = { 0, 0, 0 }
192  self.showNotOwnedWarning = false
193
194  self.isCutting = false
195  self.isHorizontalCut = false
196
197  delete(xmlFile)
198
```

```
199  return true
200  end
```

delete

Description

Deleting chainsaw

Definition

delete()

Return Values

table instance instance of object

Code

```
204  function Chainsaw:delete()
205  if self.isClient then
206  ParticleUtil.deleteParticleSystems(self.particleSystems)
207
208  if self.ringSelector ~= nil then
209  delete(self.ringSelector)
210  end
211  for _,sample in pairs(self.samplesTree) do
212  g_soundManager:deleteSample(sample)
213  end
214  for _,sample in pairs(self.samplesBranch) do
215  g_soundManager:deleteSample(sample)
216  end
217  for _,sample in pairs(self.samples) do
218  g_soundManager:deleteSample(sample)
219  end
220  if self.ringSelectorFilename ~= nil then
221  g_i3DManager:releaseSharedI3DFile(self.ringSelectorFilename,
     self.baseDirectory, true)
222  end
223  g_animationManager:deleteAnimations(self.chains)
224  end
225
226  Chainsaw:superClass().delete(self)
227  end
```

cutRaycastCallback

Description

Saves raycast information

Definition

cutRaycastCallback(integer hitObjectId, float x, float y, float z, float distance)

Arguments

integer hitObjectId
float x
float y
float z
float distance

Return Values

bool true if player can swim

Code

```
236 function Chainsaw:cutRaycastCallback(hitObjectId, x, y, z, distance)
237 setWorldTranslation(self.player.chainsawCameraFocus, x, y, z)
238 self.cutFocusDistance = distance
239 end
```

updateCutRaycast

Description

Cast ray from the player camera

Definition

updateCutRaycast()

Return Values

bool true if player can swim

Code

```
243 function Chainsaw:updateCutRaycast()
244 self.cutFocusDistance = -1.0
245 setTranslation(self.player.chainsawCameraFocus, 0, 0, 0)
246 local cameraPosition = {getWorldTranslation(self.player.cameraNode)}
247 local worldDirection = {unProject(0.52, 0.4, 1)} -- Transform vector from screen space into world space
248 local treeCollisionMask = 16789504 -- bits: 12,13,24
249
250 worldDirection[1],
251 worldDirection[2],
252 worldDirection[3] = MathUtil.vector3Normalize(worldDirection[1], worldDirection[2], worldDirection[3])
253 raycastClosest(cameraPosition[1], cameraPosition[2], cameraPosition[3],
254 worldDirection[1], worldDirection[2], worldDirection[3],
255 "cutRaycastCallback", self.player.cutDetectionDistance,
256 self, treeCollisionMask)
257 end
```

testTooLow

Description

Definition

testTooLow(integer shape, float minY, float maxY, float minZ, float maxZ)

Arguments

integer shape
float minY
float maxY
float minZ
float maxZ

Return Values

table instance instance of object

Code

```
267 function Chainsaw:testTooLow(shape, minY, maxY, minZ, maxZ)
268 local cutTooLow = false
269 local _,y1,_ = localToLocal(self.player.chainsawSplitShapeFocus,
    shape, 0,minY,minZ)
270 local _,y3,_ = localToLocal(self.player.chainsawSplitShapeFocus,
    shape, 0,maxY,minZ)
271 local _,y4,_ = localToLocal(self.player.chainsawSplitShapeFocus,
    shape, 0,maxY,maxZ)
272
273 cutTooLow = y1 < 0.01 or y1 < 0.01 or y3 < 0.03 or y4 < 0.01
274 if not cutTooLow then
275 local x1,y1,z1 =
    localToWorld(self.player.chainsawSplitShapeFocus, 0,minY,minZ)
276 local x2,y2,z2 =
    localToWorld(self.player.chainsawSplitShapeFocus, 0,minY,maxZ)
277 local x3,y3,z3 =
    localToWorld(self.player.chainsawSplitShapeFocus, 0,maxY,minZ)
278 local x4,y4,z4 =
    localToWorld(self.player.chainsawSplitShapeFocus, 0,maxY,maxZ)
279 local h1 =
    getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode,
    x1,y1,z1)
280 local h2 =
    getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode,
    x2,y2,z2)
281 local h3 =
    getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode,
    x3,y3,z3)
282 local h4 =
    getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode,
    x4,y4,z4)
283 cutTooLow = h1 < 0.01 or h2 < 0.01 or h3 < 0.03 or h4 < 0.01
284 end
285 if cutTooLow then
286 return true
```

```
287 end
288 return false
289 end
```

getLookAt

Description

Given a target position we calculate a up and towards direction (result is in world reference)

Definition

getLookAt(integer camera, float target, float target, float target)

Arguments

integer camera identifier
float target position x
float target position y
float target position z

Return Values

table instance Instance of object
float
float
float
float
float
float

Code

```
303 function Chainsaw:getLookAt(cameraNode, targetX, targetY, targetZ)
304 local xx, xy, xz = 0, 0, 0
305 local yx, yy, yz = 0, 0, 0
306 local zx, zy, zz = 0, 0, 0
307 local nodePosition = {getWorldTranslation(cameraNode)}
308 local nodeUpDirection = {localDirectionToWorld(getParent(cameraNode), 0, -1, 0)}
309
310 zx = nodePosition[1] - targetX
311 zy = nodePosition[2] - targetY
312 zz = nodePosition[3] - targetZ
313 zx, zy, zz = MathUtil.vector3Normalize(zx, zy, zz)
314 xx, xy, xz = MathUtil.crossProduct(zx, zy, zz, nodeUpDirection[1], nodeUpDirection[2], nodeUpDirection[3])
315 xx, xy, xz = MathUtil.vector3Normalize(xx, xy, xz)
316 yx, yy, yz = MathUtil.crossProduct(zx, zy, zz, xx, xy, xz)
317 yx, yy, yz = MathUtil.vector3Normalize(yx, yy, yz)
318
319 return yx, yy, yz, zx, zy, zz
```

```
320 end
```

getCutStartEnd

Description

Function to calculate the start and the end of the cut by using the ringSelector

Definition

getCutStartEnd()

Return Values

table instance Instance of object

Code

```
324 function Chainsaw:getCutStartEnd()
325 local selectorPosition = {getWorldTranslation(self.ringSelector)}
326 local selectorScale = {getScale(self.ringSelector)}
327 local cutDirection = {localDirectionToWorld(self.ringSelector, 0, 1, 0)}
328 local cutStartposition = { selectorPosition[1] - 0.5 * selectorScale[1] * cutDirection[1],
329 selectorPosition[2] - 0.5 * selectorScale[2] * cutDirection[2],
330 selectorPosition[3] - 0.5 * selectorScale[3] * cutDirection[3] }
331 local cutEndposition = { selectorPosition[1] + 0.5 * selectorScale[1] * cutDirection[1],
332 selectorPosition[2] + 0.5 * selectorScale[2] * cutDirection[2],
333 selectorPosition[3] + 0.5 * selectorScale[3] * cutDirection[3] }
334 return cutStartposition[1], cutStartposition[2], cutStartposition[3], cutEndposition[1], cutEndposition[2], cutEndposition[3]
335 end
```

calculateCutDuration

Description

Definition

calculateCutDuration(float dt, bool true)

Arguments

float dt in milliseconds

bool true if the player is cutting

Return Values

table instance Instance of object

Code

```
341 function Chainsaw:calculateCutDuration()
342 local startX, startY, startZ, endX, endY, endZ = self:getCutStartEnd()
343 local trunkWidthSq = MathUtil.vector3LengthSq(endX - startX, endY - startY, endZ - startZ)
344 trunkWidthSq = MathUtil.clamp(trunkWidthSq, 0.0, self.maxTrunkWidthSq)
```

```
345 local cutDuration = trunkWidthSq * self.defaultCutDuration / self.maxTrunkWidthSq
346 return cutDuration
347 end
```

updateCuttingTimers

Description

Definition

updateCuttingTimers(float dt, bool true)

Arguments

float dt in milliseconds

bool true if the player is cutting

Return Values

table instance Instance of object

Code

```
353 function Chainsaw:updateCuttingTimers(dt, isCutting)
354 local dtInSec = dt * 0.001
355 self.transitionAlpha = 0.0
356
357 if isCutting then
358 local cutDuration = self:calculateCutDuration()
359 if (self.cutTimer == 0.0) then
360 self.outTimer = 0.0
361 self.cameraTransitionState = Chainsaw.CAMERA_TRANSITION_STATES.IN
362 elseif self.cutTimer == self.inDuration then
363 self.cameraTransitionState = Chainsaw.CAMERA_TRANSITION_STATES.CUT
364 end
365
366 if (self.cutTimer >= 0.0) and (self.cutTimer < self.inDuration) then
367 self.cutTimer = math.min(self.cutTimer + dtInSec, self.inDuration)
368 self.transitionAlpha = MathUtil.clamp(self.cutTimer, 0.0, self.inDuration) / self.inDuration
369 elseif (self.cutTimer >= self.inDuration) and (self.cutTimer < cutDuration) then
370 local restCutDuration = math.max(cutDuration - self.inDuration, 0)
371
372 self.cutTimer = math.min(self.cutTimer + dtInSec, cutDuration)
373 self.transitionAlpha = MathUtil.clamp(self.cutTimer - self.inDuration, 0, restCutDuration) / restCutDuration
374 else
```

```
375  self.transitionAlpha = 1.0
376  end
377  else
378  cutDuration = self.defaultCutDuration
379  if self.outTimer == 0.0 then
380  self.cutTimer = 0.0
381  self.cameraTransitionState =
     Chainsaw.CAMERA_TRANSITION_STATES.OUT
382  end
383
384  if (self.outTimer >= 0.0) and (self.outTimer < self.outDuration)
     then
385  self.outTimer = math.min(self.outTimer + dtInSec,
     self.outDuration)
386  self.transitionAlpha = MathUtil.clamp(self.outTimer, 0.0,
     self.outDuration) / self.outDuration
387  end
388  end
389  end
```

resetTransitionState

Description

Definition

resetTransitionState()

Return Values

table instance Instance of object

Code

```
393  function Chainsaw:resetTransitionState()
394  if (self.cameraTransitionState ~=
     Chainsaw.CAMERA_TRANSITION_STATES.NONE) then
395  self.cameraTransitionState =
     Chainsaw.CAMERA_TRANSITION_STATES.NONE
396  end
397  end
```

updateCuttingCamera

Description

This function updates the timers: cuttimer and the outTimer, depending if the player is cutting or not: we switch from player camera to cutting camera as to not lose focus. We rotate the cutting camera towards the cut target. We the player has finished cutting we move the cutting camera towards the player camera and switch back to the player camera.

Definition

updateCuttingCamera(bool isCutting)

Arguments

bool isCutting true if the player is cutting

Return Values

table instance Instance of object

Code

```
402 function Chainsaw:updateCuttingCamera(isCutting)
403 if isCutting then
404 if self.cameraTransitionState ==
    Chainsaw.CAMERA_TRANSITION_STATES.IN then
405 setRotation(self.player.cuttingCameraNode, 0, 0, 0)
406 local yx, yy, yz =
    localDirectionToWorld(self.player.cuttingCameraNode, 0, 1, 0)
407 local zx, zy, zz =
    localDirectionToWorld(self.player.cuttingCameraNode, 0, 0, 1)
408 local startX, startY, startZ, _, _, _ = self:getCutStartEnd()
409
410 self.startCameraDirectionY = {yx, yy, yz}
411 self.startCameraDirectionZ = {zx, zy, zz}
412 self.endCameraDirectionY[1], self.endCameraDirectionY[2],
    self.endCameraDirectionY[3],
413 self.endCameraDirectionZ[1], self.endCameraDirectionZ[2],
    self.endCameraDirectionZ[3] =
    self:getLookAt(self.player.cuttingCameraNode, startX, startY,
    startZ)
414 elseif self.cameraTransitionState ==
    Chainsaw.CAMERA_TRANSITION_STATES.CUT then
415 local startX, startY, startZ, endX, endY, endZ =
    self:getCutStartEnd()
416
417 self.startCameraDirectionY[1], self.startCameraDirectionY[2],
    self.startCameraDirectionY[3],
418 self.startCameraDirectionZ[1], self.startCameraDirectionZ[2],
    self.startCameraDirectionZ[3] =
    self:getLookAt(self.player.cuttingCameraNode, startX, startY,
    startZ)
419 self.endCameraDirectionY[1], self.endCameraDirectionY[2],
    self.endCameraDirectionY[3],
420 self.endCameraDirectionZ[1], self.endCameraDirectionZ[2],
    self.endCameraDirectionZ[3] =
    self:getLookAt(self.player.cuttingCameraNode, endX, endY, endZ)
421 end
422 else
423 if self.cameraTransitionState ==
    Chainsaw.CAMERA_TRANSITION_STATES.OUT then
424 local yx, yy, yz =
    localDirectionToWorld(self.player.cuttingCameraNode, 0, 1, 0)
425 local zx, zy, zz =
    localDirectionToWorld(self.player.cuttingCameraNode, 0, 0, 1)
```

```
self.startCameraDirectionY = {yx, yy, yz}
self.startCameraDirectionZ = {zx, zy, zz}
setRotation(self.player.cuttingCameraNode, 0, 0, 0)
yx, yy, yz = localDirectionToWorld(self.player.cuttingCameraNode, 0, 1, 0)
zx, zy, zz = localDirectionToWorld(self.player.cuttingCameraNode, 0, 0, 1)
self.endCameraDirectionY = {yx, yy, yz}
self.endCameraDirectionZ = {zx, zy, zz}
end
end

local currentCamera = getCamera()
if isCutting or self.outTimer < self.outDuration then
if (currentCamera ~= self.player.cuttingCameraNode) then
g_currentMission:addSpecialCamera(self.player.cuttingCameraNode)
setCamera(self.player.cuttingCameraNode)
end

local smoothDirY = { MathUtil.lerp(self.startCameraDirectionY[1], self.endCameraDirectionY[1], self.transitionAlpha),
MathUtil.lerp(self.startCameraDirectionY[2], self.endCameraDirectionY[2], self.transitionAlpha),
MathUtil.lerp(self.startCameraDirectionY[3], self.endCameraDirectionY[3], self.transitionAlpha) }
local smoothDirZ = { MathUtil.lerp(self.startCameraDirectionZ[1], self.endCameraDirectionZ[1], self.transitionAlpha),
MathUtil.lerp(self.startCameraDirectionZ[2], self.endCameraDirectionZ[2], self.transitionAlpha),
MathUtil.lerp(self.startCameraDirectionZ[3], self.endCameraDirectionZ[3], self.transitionAlpha) }

smoothDirY = {
worldDirectionToLocal(getParent(self.player.cuttingCameraNode), smoothDirY[1], smoothDirY[2], smoothDirY[3]) }
smoothDirZ = {
worldDirectionToLocal(getParent(self.player.cuttingCameraNode), smoothDirZ[1], smoothDirZ[2], smoothDirZ[3]) }
local d,e,f =
getWorldTranslation(self.player.chainsawSplitShapeFocus)
setDirection(self.player.cuttingCameraNode, smoothDirZ[1], smoothDirZ[2], smoothDirZ[3], smoothDirY[1], smoothDirY[2], smoothDirY[3])
else
```

```
455  if (currentCamera ~= self.player.cameraNode) then
456  setRotation(self.player.cuttingCameraNode, 0, 0, 0)
457  g_currentMission:removeSpecialCamera(self.player.cuttingCameraNode)
458  setCamera(self.player.cameraNode)
459  end
460  end
461  end
```

updateChainsawModel

Description

Definition

updateChainsawModel(bool isCutting)

Arguments

bool isCutting

Return Values

table instance Instance of object

Code

```
466  function Chainsaw:updateChainsawModel(isCutting)
467  local currentPos = {getWorldTranslation(self.graphicsNode)}
468
469  if isCutting then
470  local startPos = {}
471  local endPos = {}
472  startPos[1], startPos[2], startPos[3], endPos[1], endPos[2],
     endPos[3] = self:getCutStartEnd()
473
474  if self.cameraTransitionState ==
     Chainsaw.CAMERA_TRANSITION_STATES.IN then
475  self.startChainsawPosition = currentPos
476  self.endChainsawPosition = startPos
477  elseif self.cameraTransitionState ==
     Chainsaw.CAMERA_TRANSITION_STATES.CUT then
478  self.startChainsawPosition = startPos
479  self.endChainsawPosition = endPos
480  end
481  else
482  if self.cameraTransitionState ==
     Chainsaw.CAMERA_TRANSITION_STATES.OUT then
483  self.startChainsawPosition = currentPos
484  setTranslation(self.graphicsNode, 0, 0, 0)
485  self.endChainsawPosition =
     {getWorldTranslation(self.graphicsNode)}
486  end
```

```
487 end
488
489 if isCutting or self.outTimer < self.outDuration then
490 local smoothPosition = { MathUtil.lerp(
    self.startChainsawPosition[1], self.endChainsawPosition[1],
    self.transitionAlpha ),
491 MathUtil.lerp( self.startChainsawPosition[2],
    self.endChainsawPosition[2], self.transitionAlpha ),
492 MathUtil.lerp( self.startChainsawPosition[3],
    self.endChainsawPosition[3], self.transitionAlpha )}
493 local offset = {localToLocal(self.cutPositionNode,
    self.graphicsNode, 0,0,0)}
494 local cutDirection = {localDirectionToWorld(self.ringSelector, 0,
    0, offset[3])}
495 local destination = {smoothPosition[1] - cutDirection[1],
    smoothPosition[2] - cutDirection[2], smoothPosition[3] -
    cutDirection[3]}
496 local modelTranslation =
    {worldToLocal(getParent(self.graphicsNode), destination[1],
    destination[2], destination[3])}
497 local distance = MathUtil.vector3Length(modelTranslation[1],
    modelTranslation[2], modelTranslation[3])
498
499 if distance > self.maxModelTranslation then
500 modelTranslation =
    {MathUtil.vector3Normalize(modelTranslation[1],
    modelTranslation[2], modelTranslation[3])}
501 modelTranslation = {modelTranslation[1]*self.maxModelTranslation,
    modelTranslation[2]*self.maxModelTranslation,
    modelTranslation[3]*self.maxModelTranslation}
502 local screen = { project(destination[1], destination[2],
    destination[3]) }
503 setTranslation(self.graphicsNode, modelTranslation[1],
    modelTranslation[2], modelTranslation[3])
504 local graph = {getWorldTranslation(self.graphicsNode)}
505 local screen2 = { project(graph[1], graph[2], graph[3]) }
506 local world2 = { unProject(screen[1], screen[2], screen2[3]) }
507
508 setWorldTranslation(self.graphicsNode, world2[1], world2[2],
    world2[3])
509 else
510 setTranslation(self.graphicsNode, modelTranslation[1],
    modelTranslation[2], modelTranslation[3])
511 end
512 else
```

```
513  setTranslation(self.graphicsNode, 0, 0, 0)
514  end
515  end
```

getCutShapeInformation

Description

Preparing information for the splitShape methods

Definition

getCutShapeInformation(float x, float y, float z, float nx, float ny, float nz, float yx, float yy, float yz)

Arguments

float x
float y
float z
float nx
float ny
float nz
float yx
float yy
float yz

Return Values

table instance Instance of object

Code

```
528  function Chainsaw:getCutShapeInformation()
529  local x,y,z = getWorldTranslation(self.player.chainsawSplitShapeFocus)
530  local nx,ny,nz = localDirectionToWorld(self.player.chainsawSplitShapeFocus, 1,0,0)
531  local yx,yy,yz = localDirectionToWorld(self.player.chainsawSplitShapeFocus, 0,1,0)
532  return x, y, z, nx, ny, nz, yx, yy, yz
533  end
```

update

Description

Update

Definition

update(float dt, boolean allowInput)

Arguments

float	dt	time since last call in ms
boolean	allowInput	allow input

Return Values

table instance instance of object

Code

```
539  function Chainsaw:update(dt, allowInput)
```

```
Chainsaw:superClass().update(self, dt, allowInput)

if self.isServer then
local price = self.pricePerSecond * (dt / 1000)
g_farmManager:getFarmById(self.player.farmId).stats:updateStats("expense
g_currentMission:addMoney(-price, self.player.farmId, "vehicleRunningCos
end

if self.isClient then
if not self.isCutting then
self:updateCutRaycast()
end

if self.showNotOwnedWarning then
g_currentMission:showBlinkingWarning(g_i18n:getText("warning_youDontHave
2000)
self.showNotOwnedWarning = false -- reset so it can be set to true later
end
end

self.shouldDelimb = false

local lockPlayerInput = false

if allowInput then
local isCutting = false
local hasBeenCut = false

setRotation(self.graphicsNode, math.rad(math.random(-1, 1)) * 0.1, math.
1)) * 0.1, math.rad(-180))

if self.curSplitShape == nil then
lockPlayerInput = self.rotateInput ~= 0

if self.rotateInput ~= 0 then
self.rotationZ = MathUtil.clamp(self.rotationZ + self.rotationSpeedZ * s
dt, self.maxRotationZ, self.minRotationZ )
setRotation(self.rootNode, self.handNodeRotation[1], self.handNodeRotati
self.handNodeRotation[3] + self.rotationZ)
setRotation(self.player.chainsawCameraFocus, 0, 0, -self.rotationZ)
end
```

```
577 end
578
579 local shape = 0
580 if not self.waitingForResetAfterCut then
581 if self.curSplitShape ~= nil or self.cutTimer == 0 then
582 local minY,maxY, minZ,maxZ
583 if self.curSplitShape == nil or not entityExists(self.curSplitShape) the
584 self.curSplitShape = nil
585
586 local x, y, z, nx, ny, nz, yx, yy, yz = self:getCutShapeInformation()
587 shape, minY, maxY, minZ, maxZ = findSplitShape(x, y, z, nx, ny, nz, yx,
    self.cutSizeY, self.cutSizeZ)
588
589 if shape ~= nil and shape ~= 0 then
590 if self:isCuttingAllowed(x, y, z) then
591 self.showNotOwnedWarning = false
592
593 local cutTooLow = self:testTooLow(shape, minY, maxY, minZ, maxZ)
594 local outsideRange = (self.cutFocusDistance < 0.0 or self.cutFocusDistan
    self.player.cutDetectionDistance)
595 if cutTooLow or outsideRange then
596 self.player.walkingIsLocked = false
597 self.curSplitShape = nil
598 shape, minY,maxY, minZ,maxZ = 0, nil,nil, nil,nil
599 end
600 else
601 self.showNotOwnedWarning = true
602 end
603 end
604 self.curSplitShapeMinY = minY
605 self.curSplitShapeMaxY = maxY
606 self.curSplitShapeMinZ = minZ
607 self.curSplitShapeMaxZ = maxZ
608 else
609 shape = self.curSplitShape
610 end
611
612 self:updateRingSelector(shape)
613 end
614 end
615
```

```
616 if self.activatePressed then
617 self.speedFactor = math.min(self.speedFactor + dt/self.maxWorkTime, 1)
618
619 if not self.waitingForResetAfterCut then
620 local inRange = (self.cutFocusDistance >= self.player.minCutDistance and
    self.cutFocusDistance < self.player.maxCutDistance)
621 self.shouldDelimb = inRange
622
623 if (self.curSplitShape ~= nil or self.cutTimer == 0) and inRange then
624 if self.curSplitShape ~= nil and entityExists(self.curSplitShape) then
625 lockPlayerInput = true
626 local x, y, z, nx, ny, nz, yx, yy, yz = self:getCutShapeInformation()
627 local minY, maxY, minZ, maxZ = testSplitShape(self.curSplitShape, x, y,
    yy, yz, self.cutSizeY, self.cutSizeZ)
628
629 if minY == nil then
630 -- cancel cutting if shape can't be cut anymore from current position
631 self.player.walkingIsLocked = false
632 self.curSplitShape = nil
633 else
634 local cutTooLow = self:testTooLow(self.curSplitShape, minY,maxY, minZ,ma
635 if cutTooLow then
636 self.player.walkingIsLocked = false
637 self.curSplitShape = nil
638 end
639 end
640
641 self.curSplitShapeMinY = minY
642 self.curSplitShapeMaxY = maxY
643 self.curSplitShapeMinZ = minZ
644 self.curSplitShapeMaxZ = maxZ
645 else
646 if shape ~= 0 then
647 self.player.walkingIsLocked = true
648 self.curSplitShape = shape
649 end
650 end
651
652 if self.curSplitShape ~= nil then
653 local x,y,z, nx,ny,nz, yx,yy,yz = self:getCutShapeInformation()
654
```

```lua
if self:isCuttingAllowed(x, y, z) then
isCutting = true
end

if self.cutTimer > 0 then
self.lastWorkTime = math.min(self.lastWorkTime, self.maxWorkTime*0.7)
end

local cutDuration = self:calculateCutDuration()
if self.cutTimer >= cutDuration then
if g_currentMission:getIsServer() then
ChainsawUtil.cutSplitShape(self.curSplitShape, x, y, z, nx, ny, nz, yx,
self.cutSizeY, self.cutSizeZ, self.player.farmId)
else
g_client:getServerConnection():sendEvent(ChainsawCutEvent:new(self.curSp
nx, ny, nz, yx, yy, yz, self.cutSizeY, self.cutSizeZ, self.player.farmId
end

hasBeenCut = true
self.waitingForResetAfterCut = true
self.player.walkingIsLocked = false
self.curSplitShape = nil
self.curSplitShapeMinY = nil
self:updateRingSelector(0)
end
end
end
end
else
self.speedFactor = math.max(self.speedFactor - dt / self.maxWorkTime, 0)
self.waitingForResetAfterCut = false
self.player.walkingIsLocked = false
self.curSplitShape = nil
self.curSplitShapeMinY = nil
self.lastWorkTime = math.max(self.lastWorkTime - dt, 0)
self.workUpPlayed = false
end

self.player:lockInput(lockPlayerInput)

self:updateCuttingTimers(dt, isCutting)
```

```
694  self:updateCuttingCamera(isCutting)
695  self:updateChainsawModel(isCutting)
696  self:updateDelimb()
697  self:setCutting(isCutting, self.rotationZ > 0.7, hasBeenCut)
698
699  self.soundFSM:update(dt)
700  end
701  self:updateParticles()
702
703  self.rotateInput = 0
704  self.activatePressed = false
705  end
```

updateDelimb

Description

Update delimb mechanic (removing leaves, ...)

Definition

updateDelimb()

Return Values

boolean true if loading was successful else false

Code

```
713  function Chainsaw:updateDelimb()
714  if self.shouldDelimb then
715  local x,y,z = getWorldTranslation(self.player.chainsawSplitShapeFocus)
716  local nx,ny,nz = localDirectionToWorld(self.player.chainsawSplitShapeFoc
     1,0,0)
717  local yx,yy,yz = localDirectionToWorld(self.player.chainsawSplitShapeFoc
     0,1,0)
718  if g_server == nil then
719  g_client:getServerConnection():sendEvent(ChainsawDelimbEvent:new(self.pl
     x, y, z, nx, ny, nz, yx, yy, yz, false))
720  else
721  local ret = findAndRemoveSplitShapeAttachments(x, y, z, nx, ny, nz, yx,
     yz, 0.7, self.cutSizeY, self.cutSizeZ)
722  if ret then
723  self:setOnDelimb(true)
724  end
725  end
726  end
727  end
```

updateParticles

Description

Update particle system

Definition

updateParticles()

Return Values

table the brand object or nil of an error occured

Code

```
731  function Chainsaw:updateParticles()
732  if self.particleSystems ~= nil then
733  local active = false
734  if self.isCutting and ((self.samplesBranchActiveTimer >
     g_currentMission.time) or (self.curSplitShapeMinY ~= nil and
     self.curSplitShapeMaxY ~= nil and self.cutTimer >
     self.inDuration)) then
735  active = true
736  end
737  if self.isCutting and not self.player.isEntered then
738  active = true
739  end
740  for _, ps in pairs(self.particleSystems) do
741  ParticleUtil.setEmittingState(ps, active)
742  end
743  end
744  end
```

updateRingSelector

Description

Update ring selector

Definition

updateRingSelector(integer shape)

Arguments

integer shape

Return Values

table brand the brand object

Code

```
749  function Chainsaw:updateRingSelector(shape)
750  if self.ringSelector ~= nil then
751  local hasShape = (shape ~= nil) and (shape ~= 0)
752
753  if g_woodCuttingMarkerEnabled and hasShape then
754  local inDetectionRange = false
755  local inCutRange = false
756
757  if self.cutFocusDistance ~= nil and (self.cutFocusDistance >= 0.0
     and self.cutFocusDistance < self.player.cutDetectionDistance)
     then
```

```
758 inDetectionRange = true
759 inCutRange = (self.cutFocusDistance >= self.player.minCutDistance
    and self.cutFocusDistance < self.player.maxCutDistance)
760 end
761 if not getVisibility(self.ringSelector) and inDetectionRange then
762 local x, y, z = getWorldTranslation(self.ringSelector)
763 if self:isCuttingAllowed(x, y, z) then
764 setVisibility(self.ringSelector, true)
765 else
766 setVisibility(self.ringSelector, false)
767 end
768 elseif getVisibility(self.ringSelector) and not inDetectionRange
    then
769 setVisibility(self.ringSelector, false)
770 end
771
772 if getVisibility(self.ringSelector) then
773 if inCutRange then
774 setShaderParameter(self.ringSelector, "colorScale", 0.395, 0.925,
    0.115, 1, false)
775 else
776 setShaderParameter(self.ringSelector, "colorScale", 0.098, 0.450,
    0.960, 1, false)
777 end
778
779 if self.curSplitShapeMinY ~= nil then
780 local scale = math.max(self.curSplitShapeMaxY -
    self.curSplitShapeMinY + self.ringSelectorScaleOffset,
    self.curSplitShapeMaxZ-self.curSplitShapeMinZ +
    self.ringSelectorScaleOffset)
781 setScale(self.ringSelector, 1, scale, scale)
782
783 local a,b,c = localToWorld(self.player.chainsawSplitShapeFocus,
    0, (self.curSplitShapeMinY+self.curSplitShapeMaxY)*0.5,
    (self.curSplitShapeMinZ+self.curSplitShapeMaxZ)*0.5)
784 local x, y, z = worldToLocal(getParent(self.ringSelector), a,b,c)
785 setTranslation(self.ringSelector, x,y,z)
786 else
787 setScale(self.ringSelector, 1, 1, 1)
788 end
789 end
790 elseif getVisibility(self.ringSelector) then
```

```
791  setVisibility(self.ringSelector, false)
792  end
793  end
794  end
```

setCutting

Description

Set cutting

Definition

setCutting(boolean isCutting, boolean isHorizontalCut, boolean hasBeenCut, boolean noEventSend)

Arguments

boolean isCutting	is cutting
boolean isHorizontalCut	is horizontal cut
boolean hasBeenCut	
boolean noEventSend	no event send

Return Values

table brand the brand object

Code

```
802  function Chainsaw:setCutting(isCutting, isHorizontalCut,
     hasBeenCut, noEventSend)
803  ChainsawStateEvent.sendEvent(self.player, isCutting,
     isHorizontalCut, hasBeenCut, noEventSend)
804  self.isCutting = isCutting
805  self.isHorizontalCut = isHorizontalCut
806  self.hasBeenCut = hasBeenCut
807
808  if not self.player.isOwner then
809  self.player:setCuttingAnim(isCutting, isHorizontalCut)
810  end
811  end
```

setOnDelimb

Description

Set on delimb

Definition

setOnDelimb(boolean state)

Arguments

boolean state new state

Return Values

integer brandIndex the brand index

Code

```
816  function Chainsaw:setOnDelimb(state)
817  if state == true then
818  end
```

```
819 end
```

setHandNode

Description

Set hand node

Definition

setHandNode(integer handNode)

Arguments

integer handNode hand node id

Return Values

boolean true if storeitem is a vehicle, else false

Code

```
824 function Chainsaw:setHandNode(handNode)
825 Chainsaw:superClass().setHandNode(self, handNode)
826 if self.currentHandNode ~= handNode then
827 link(handNode, self.rootNode)
828 self.currentHandNode = handNode
829 setRotation(self.rootNode, unpack(self.handNodeRotation))
830 local x, y, z = getWorldTranslation(self.handNode)
831 x, y, z = worldToLocal(getParent(self.rootNode), x, y, z)
832 local a, b, c = getTranslation(self.rootNode)
833 setTranslation(self.rootNode, a - x, b - y, c - z)
834 end
835 end
```

onActivate

Description

On activate

Definition

onActivate(boolean allowInput)

Arguments

boolean allowInput allow input

Return Values

boolean true if storeitem is an animal, else false

Code

```
844 function Chainsaw:onActivate(allowInput)
845 Chainsaw:superClass().onActivate(self)
846
847 self.rotationZ = 0.0
848 setRotation(self.rootNode, self.handNodeRotation[1],
    self.handNodeRotation[2], self.handNodeRotation[3])
849 setRotation(self.player.chainsawCameraFocus, 0, 0,
    self.rotationZ)
850
```

```
851 self.startTime = g_currentMission.time
852 if not self.player.isOwner then
853 self.player.visualInformation:setProtectiveUV(self.equipmentUVs)
854 self.player:setWoodWorkVisibility(true)
855 end
856
857 if self.isClient then
858 g_animationManager:startAnimations(self.chains)
859 end
860
861 self.cutTimer = 0.0
862 setTranslation(self.graphicsNode, 0, 0, 0)
863
864 self.soundFSM:changeState(Chainsaw.SOUND_STATES.START)
865 end
```

onDeactivate

Description

On deactivate

Definition

onDeactivate(boolean allowInput)

Arguments

boolean allowInput allow input

Return Values

boolean true if storeitem is a placeable, else false

Code

```
870 function Chainsaw:onDeactivate(allowInput)
871 Chainsaw:superClass().onDeactivate(self)
872
873 self.speedFactor = 0
874 self.curSplitShape = nil
875 self.player:lockInput(false)
876 self.player.walkingIsLocked = false
877 self.player:setWoodWorkVisibility(false)
878 if self.isClient then
879 g_animationManager:stopAnimations(self.chains)
880 self.cutTimer = 0.0
881 setTranslation(self.graphicsNode, 0, 0, 0)
882 if self.particleSystems ~= nil then
883 for _, ps in pairs(self.particleSystems) do
884 ParticleUtil.resetNumOfEmittedParticles(ps)
885 ParticleUtil.setEmittingState(ps, false)
```

```
886 end
887 end
888 if getVisibility(self.ringSelector) then
889 setVisibility(self.ringSelector, false)
890 end
891 end
892 self.soundFSM:changeState(Chainsaw.SOUND_STATES.STOP)
893 end
```

registerActionEvents

Description

Definition

registerActionEvents()

Return Values

boolean true if storeitem is an object, else false

Code

```
897 function Chainsaw:registerActionEvents()
898 Chainsaw:superClass().registerActionEvents(self, allowInput)
899 local eventId = ""
900 _, eventId = g_inputBinding:registerActionEvent(InputAction.AXIS_ROTATE_HANDTOOL, self, self.onInputRotate, false, false, true, true)
901 g_inputBinding:setActionEventText(eventId, g_i18n:getText("action_rotate"))
902 self.eventIdRotateHandtool = eventId
903 end
```

onInputRotate

Description

Event function for chainsaw rotation bound to InputAction.AXIS_ROTATE_HANDTOOL.

Definition

onInputRotate()

Return Values

boolean true if storeitem is a handtool, else false

Code

```
907 function Chainsaw:onInputRotate(_, inputValue)
908 self.rotateInput = self.rotateInput + inputValue
909 end
```

ChainsawSoundStates

Description

Start:new

Description

Creating instance of sound state.

Definition

Start:new()

Return Values

integer	sharedVramUsage	the shared vram usage
integer	perInstanceVramUsage	the per instance vram usage
boolean	ignoreVramUsage	true if vram usage should be ignored else false

Code

```
18  function ChainsawSoundStateStart:new(id, owner, stateMachine,
    custom_mt)
19  local self = SimpleState:new(id, owner, stateMachine,
    ChainsawSoundStateStart_mt)
20  return self
21  end
```

Start:activate

Description

Activate method

Definition

Start:activate()

Code

```
25  function ChainsawSoundStateStart:activate(parms)
26  ChainsawSoundStateStart:superClass().activate(self, parms)
27  g_soundManager:playSample(self.owner.samples.start)
28  self.stateMachine:changeState(Chainsaw.SOUND_STATES.IDLE)
29  end
```

Stop:new

Description

Creating instance of sound state.

Definition

Stop:new()

Code

```
41  function ChainsawSoundStateStop:new(id, owner, stateMachine,
    custom_mt)
42  local self = SimpleState:new(id, owner, stateMachine,
    ChainsawSoundStateStop_mt)
43  return self
44  end
```

Stop:activate

Description

Activate method

Definition

Stop:activate()

Return Values

table	configurations	a list of configurations

Code

```
48  function ChainsawSoundStateStop:activate(parms)
```

```
49  ChainsawSoundStateStop:superClass().activate(self, parms)
50  g_soundManager:playSample(self.owner.samples.stop)
51  end
```

Idle:new

Description

Creating instance of sound state.

Definition

Idle:new()

Return Values

table configuration sets

Code

```
63  function ChainsawSoundStateIdle:new(id, owner, stateMachine,
    custom_mt)
64  local self = SimpleState:new(id, owner, stateMachine,
    ChainsawSoundStateIdle_mt)
65  self.activeTimer = 0.0
66  return self
67  end
```

Idle:deactivate

Description

Deactivate method

Definition

Idle:deactivate()

Return Values

table instance instance of object

Code

```
71  function ChainsawSoundStateIdle:deactivate()
72  ChainsawSoundStateIdle:superClass().deactivate(self)
73  g_soundManager:stopSample(self.owner.samples.idle)
74  self.activeTimer = 0.0
75  end
```

Idle:update

Description

update method

Definition

Idle:update(float dt)

Arguments

float dt in ms

Return Values

boolean true if loading was successful else false

Code

```
80  function ChainsawSoundStateIdle:update(dt)
```

```
81  ChainsawSoundStateIdle:superClass().update(self, dt)
82
83  if not g_soundManager:getIsSamplePlaying(self.owner.samples.start) and not g_soundManager:getIsSamplePlaying(self.owner.samples.idle) then
84  g_soundManager:playSample(self.owner.samples.idle)
85  end
86
87  if self.owner.isCutting then
88  self.stateMachine:changeState(Chainsaw.SOUND_STATES.CUT)
89  elseif self.owner.activatePressed then
90  self.activeTimer = self.activeTimer + dt
91
92  if self.activeTimer > self.owner.quicktapThreshold then
93  self.stateMachine:changeState(Chainsaw.SOUND_STATES.ACTIVE)
94  end
95  else
96  if self.activeTimer > 0.0 and self.activeTimer < self.owner.quicktapThreshold then
97  self.stateMachine:changeState(Chainsaw.SOUND_STATES.QUICKTAP)
98  end
99  end
100 end
```

Active:new

Description

Creating instance of sound state.

Definition

Active:new()

Return Values

boolean true if adding was successful else false

Code

```
112 function ChainsawSoundStateActive:new(id, owner, stateMachine, custom_mt)
113 local self = SimpleState:new(id, owner, stateMachine, ChainsawSoundStateActive_mt)
114 return self
115 end
```

Active:activate

Description

Activate method

Definition

Active:activate()

Return Values

table category the category object

Code

```
119  function ChainsawSoundStateActive:activate(parms)
120  ChainsawSoundStateActive:superClass().activate(self, parms)
121  local shouldInitiateStart = false
122  shouldInitiateStart = not (parms ~= nil and parms.alreadyActive)
123  if shouldInitiateStart then
124  g_soundManager:playSample(self.owner.samples.activeStart)
125  end
126  end
```

Active:deactivate

Description

Deactivate method

Definition

Active:deactivate()

Return Values

table specTypes a list of spec types

Code

```
130  function ChainsawSoundStateActive:deactivate()
131  ChainsawSoundStateActive:superClass().deactivate(self)
132  g_soundManager:stopSample(self.owner.samples.activeStart)
133  g_soundManager:stopSample(self.owner.samples.activeLoop)
134  end
```

Active:update

Description

update method

Definition

Active:update(float dt)

Arguments

float dt in ms

Return Values

table specType the corresponding spectype

Code

```
139  function ChainsawSoundStateActive:update(dt)
140  ChainsawSoundStateActive:superClass().update(self, dt)
141  if self.owner.isCutting then
142  local parms = {}
143  parms.alreadyActive = not
     g_soundManager:getIsSamplePlaying(self.owner.samples.activeStart)
144  self.stateMachine:changeState(Chainsaw.SOUND_STATES.CUT, parms)
145  elseif self.owner.activatePressed then
```

```
146 if not g_soundManager:getIsSamplePlaying(self.owner.samples.activeStart) and not g_soundManager:getIsSamplePlaying(self.owner.samples.activeLoop) then
147 g_soundManager:playSample(self.owner.samples.activeLoop)
148 end
149 else
150 self.stateMachine:changeState(Chainsaw.SOUND_STATES.IDLE)
151 g_soundManager:playSample(self.owner.samples.activeStop)
152 end
153 end
```

Cut:new

Description

Creating instance of sound state.

Definition

Cut:new()

Return Values

boolean wasSuccessfull true if added else false

Code

```
165 function ChainsawSoundStateCut:new(id, owner, stateMachine, custom_mt)
166 local self = SimpleState:new(id, owner, stateMachine, ChainsawSoundStateCut_mt)
167 return self
168 end
```

Cut:activate

Description

Activate method

Definition

Cut:activate()

Return Values

table items a list of all store items

Code

```
172 function ChainsawSoundStateCut:activate(parms)
173 ChainsawSoundStateCut:superClass().activate(self, parms)
174 local shouldInitiateStart = false
175 shouldInitiateStart = not (parms ~= nil and parms.alreadyActive)
176 if shouldInitiateStart then
177 g_soundManager:playSample(self.owner.samples.cutStart)
178 end
179 end
```

Cut:deactivate

Description

Deactivate method

Definition

Cut:deactivate()

Return Values

table storeItem the storeitem oject

Code

```
183  function ChainsawSoundStateCut:deactivate()
184  ChainsawSoundStateCut:superClass().deactivate(self)
185  g_soundManager:stopSample(self.owner.samples.cutStart)
186  g_soundManager:stopSample(self.owner.samples.cutLoop)
187  end
```

Cut:update

Description

update method

Definition

Cut:update(float dt)

Arguments

float dt in ms

Return Values

table storeItem the storeitem object

Code

```
192  function ChainsawSoundStateCut:update(dt)
193  ChainsawSoundStateCut:superClass().update(self, dt)
194  if not self.owner.isCutting then
195  if self.owner.activatePressed then
196  local parms = {}
197  parms.alreadyActive = not
     g_soundManager:getIsSamplePlaying(self.owner.samples.cutStart) or
     g_soundManager:getIsSamplePlaying(self.owner.samples.cutLoop)
198  self.stateMachine:changeState(Chainsaw.SOUND_STATES.ACTIVE,
     parms)
199  g_soundManager:playSample(self.owner.samples.cutStop)
200  else
201  self.stateMachine:changeState(Chainsaw.SOUND_STATES.IDLE)
202  g_soundManager:playSample(self.owner.samples.cutStop)
203  end
204  else
205  if not
     g_soundManager:getIsSamplePlaying(self.owner.samples.cutStart)
     and not
     g_soundManager:getIsSamplePlaying(self.owner.samples.cutLoop)
     then
```

```
206 g_soundManager:playSample(self.owner.samples.cutLoop)
207 end
208 end
209 end
```

Quicktap:new

Description

Creating instance of sound state.

Definition

Quicktap:new()

Return Values

table storeItem the storeitem object

Code

```
221 function ChainsawSoundStateQuicktap:new(id, owner, stateMachine,
    custom_mt)
222 local self = SimpleState:new(id, owner, stateMachine,
    ChainsawSoundStateQuicktap_mt)
223 return self
224 end
```

Quicktap:activate

Description

Activate method

Definition

Quicktap:activate()

Return Values

table storeItem the storeitem object

Code

```
228 function ChainsawSoundStateQuicktap:activate(parms)
229 ChainsawSoundStateQuicktap:superClass().activate(self, parms)
230
231 if self.owner.samplesQuicktapCount > 0 then
232 local idx = math.floor(math.random(1,
    self.owner.samplesQuicktapCount))
233 local sample = self.owner.samplesQuicktap[idx]
234 g_soundManager:playSample(sample)
235 end
236 self.stateMachine:changeState(Chainsaw.SOUND_STATES.IDLE)
237 end
```

HandTool

Description

Class for handtools

registerHandTool

Description

Register handtool type

Definition

registerHandTool(string typeName, table classObject)

Arguments

string	typeName	name of new type
table	classObject	class object

Return Values

table sample sample object

new

Description

Creating handtool object

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean	isServer	is server
boolean	isClient	is client
table	customMt	custom metatable

Return Values

bool	True	if the definition and parameters are valid
bool	True	if an external XML file is loaded in place of the given parameter, caller must delete the handle afterwards!
int	XML	file handle, either the one passed in as an argument or an alternate external sound file definition which must be released by the caller
string	Sound	definition parent element key which may have changed according to an alternate external sound file definition
int	Link	node target for spatial sound samples
table	instance	Instance of object

Code

```
30  function HandTool:new(isServer, isClient, customMt)
31  local mt = customMt
32  if mt == nil then
33  mt = HandTool_mt
34  end
35
36  local self = Object:new(isServer, isClient, mt)
37  self.static = true
38  self.player = nil
39  self.owner = nil
40  self.price = 0
41  self.age = 0
42  self.activatePressed = false
43  return self
44  end
```

load

Description

Load chainsaw from xml file

Definition

load(string xmlFilename, table player)

Arguments

string xmlFilename xml file name

table player player

Return Values

boolean success success

Code

```
51  function HandTool:load(xmlFilename, player)
52  self.configFileName = xmlFilename
53
54  self.customEnvironment, self.baseDirectory =
    Utils.getModNameAndBaseDirectory(xmlFilename)
55
56  local xmlFile = loadXMLFile("TempXML", xmlFilename)
57  if xmlFile == 0 then
58  return false
59  end
60  local i3dFilename = getXMLString(xmlFile, "handTool.filename")
61  self.position =
    StringUtil.getVectorNFromString(Utils.getNoNil(getXMLString(xmlFile,
    "handTool.position#value"), "0 0 0"), 3)
62  self.rotation =
    StringUtil.getRadiansFromString(Utils.getNoNil(getXMLString(xmlFile,
    "handTool.rotation#value"), "0 0 0"), 3)
63
64  if i3dFilename == nil then
65  delete(xmlFile)
66  return false
67  end
68  self.i3dFilename = Utils.getFilename(i3dFilename,
    self.baseDirectory)
69
70  local node = g_i3DManager:loadSharedI3DFile(self.i3dFilename)
71  self.rootNode = getChildAt(node, 0)
72  self.player = player
73
74  local storeItem =
    g_storeManager:getItemByXMLFilename(self.configFileName)
75  if self.price == 0 or self.price == nil then
76  self.price = StoreItemUtil.getDefaultPrice(storeItem)
```

```
77  end
78
79  if g_currentMission ~= nil and storeItem.canBeSold then
80  g_currentMission.environment:addDayChangeListener(self)
81  end
82
83  self.targets = {}
84  IKUtil.loadIKChainTargets(xmlFile, "handTool.targets",
    self.rootNode, self.targets, nil)
85
86  link(player.toolsRootNode, self.rootNode)
87  setTranslation(self.rootNode, self.position[1], self.position[2],
    self.position[3])
88  setRotation(self.rootNode, self.rotation[1], self.rotation[2],
    self.rotation[3])
89
90  delete(node)
91  delete(xmlFile)
92  setVisibility(self.rootNode, false)
93
94  self.isActive = false
95
96  return true
97  end
```

delete

Description

Deleting handtool

Definition

delete()

Code

```
104 function HandTool:delete()
105 self:removeActionEvents()
106
107 if g_currentMission ~= nil then
108 g_currentMission.environment:removeDayChangeListener(self)
109 end
110 if self.player.isOwner then
111 local farmId = self.player.farmId
112 local farm = g_farmManager:getFarmById(farmId)
113 if farm ~= nil then -- only remove from farm if it still exists
    (not the case when leaving a game)
```

```
114 farm:removeHandTool(self)
115 end
116 end
117 if self.rootNode ~= nil and self.rootNode ~= 0 then
118 g_i3DManager:releaseSharedI3DFile(self.i3dFilename, nil, true)
119 delete(self.rootNode)
120 end
121 HandTool:superClass().delete(self)
122 end
```

onActivate

Description

On activate

Definition

onActivate(boolean allowInput)

Arguments

boolean allowInput allow input

Code

```
133 function HandTool:onActivate(allowInput)
134 setVisibility(self.rootNode, true)
135 self.isActive = true
136 self:raiseActive()
137
138 if self.player.isOwner then
139 self:registerActionEvents()
140 end
141 end
```

onDeactivate

Description

On deactivate

Definition

onDeactivate(boolean allowInput)

Arguments

boolean allowInput allow input

Code

```
146 function HandTool:onDeactivate(allowInput)
147 setVisibility(self.rootNode, false)
148 self.isActive = false
149 self:removeActionEvents()
150 end
```

getDailyUpkeep

Description

Get daily up keep

Definition

getDailyUpkeep()

Return Values

table sample sample object

float dailyUpkeep daily up keep

Code

```
163 function HandTool:getDailyUpkeep()
164 local storeItem = g_storeManager:getItemByXMLFilename(self.configFileName)
165 local multiplier = 1
166
167 if storeItem.lifetime ~= nil and storeItem.lifetime ~= 0 then
168 local ageMultiplier = math.min(self.age / storeItem.lifetime, 1)
169 multiplier = EconomyManager.MAX_DAILYUPKEEP_MULTIPLIER * ageMultiplier
170 end
171
172 return StoreItemUtil.getDailyUpkeep(storeItem, nil) * multiplier
173 end
```

getSellPrice

Description

Get sell price

Definition

getSellPrice()

Return Values

boolean isPlaying true if sample is playing else false

float sellPrice sell price

Code

```
178 function HandTool:getSellPrice()
179 local priceMultiplier = 0.5
180 local storeItem = g_storeManager:getItemByXMLFilename(self.configFileName)
181 local maxVehicleAge = storeItem.lifetime
182
183 if maxVehicleAge ~= nil and maxVehicleAge ~= 0 then
184 priceMultiplier = priceMultiplier * math.exp(-3.5 * math.min(self.age/maxVehicleAge, 1))
185 end
186
187 return math.floor(self.price * math.max(priceMultiplier, 0.05))
188 end
```

dayChanged

Description

Called if day changed

Definition

dayChanged()

Return Values

boolean isIndoor true if indoor mode is active else false

Code

```
192  function HandTool:dayChanged()
193  self.age = self.age + 1
194  end
```

registerActionEvents

Description

Definition

registerActionEvents()

Return Values

boolean isIndoor true if inside building mode is active else false

Code

```
198  function HandTool:registerActionEvents()
199  end
```

removeActionEvents

Description

Definition

removeActionEvents()

Return Values

float factor the modifier factor

Code

```
203  function HandTool:removeActionEvents()
204  g_inputBinding:removeActionEventsByTarget(self)
205  end
```

HighPressureWasherLance

Description

Class for high pressure washer lance

new

Description

Definition

new()

Return Values

table instance instance of object

integer group audio group

Code

```
16  function HighPressureWasherLance:new(isServer, isClient, customMt)
17  local self = HighPressureWasherLance:superClass().new(self,
    isServer, isClient, customMt or HighPressureWasherLance_mt)
18
19  self.foundVehicle = nil
```

```
20  self.doWashing = false
21  self.washDistance = 10.0
22  self.washMultiplier = 1.0
23  self.pricePerSecond = 10
24  self.isHPWLance = true
25
26  return self
27  end
```

load

Description

Definition

load()

Code

```
31  function HighPressureWasherLance:load(xmlFilename, player)
32  if not HighPressureWasherLance:superClass().load(self, xmlFilename,
    player) then
33  return false
34  end
35
36  -- Lance model
37  local xmlFile = loadXMLFile("TempXML", xmlFilename)
38  self.lanceNode = I3DUtil.indexToObject(self.rootNode,
    getXMLString(xmlFile,
    "handTool.highPressureWasherLance.lance#node"))
39
40  local lancePosition = {}
41  local lanceRotation = {}
42  if self.player == g_currentMission.player then
43  lancePosition =
    StringUtil.getVectorNFromString(Utils.getNoNil(getXMLString(xmlFile,
    "handTool.highPressureWasherLance.lance.firstPerson#position"), "0 0
    0"), 3)
44  lanceRotation =
    StringUtil.getRadiansFromString(Utils.getNoNil(getXMLString(xmlFile,
    "handTool.highPressureWasherLance.lance.firstPerson#rotation"), "0 0
    0"), 3)
45  else
46  lancePosition =
    StringUtil.getVectorNFromString(Utils.getNoNil(getXMLString(xmlFile,
    "handTool.highPressureWasherLance.lance.thirdPerson#position"), "0 0
    0"), 3)
47  lanceRotation =
    StringUtil.getRadiansFromString(Utils.getNoNil(getXMLString(xmlFile,
    "handTool.highPressureWasherLance.lance.thirdPerson#rotation"), "0 0
    0"), 3)
```

```
48  end
49  self.lanceRaycastNode = I3DUtil.indexToObject(self.rootNode,
    getXMLString(xmlFile,
    "handTool.highPressureWasherLance.lance#raycastNode"))
50  self.washDistance = Utils.getNoNil(getXMLFloat(xmlFile,
    "handTool.highPressureWasherLance.lance#washDistance"), 10)
51  self.washMultiplier = Utils.getNoNil(getXMLFloat(xmlFile,
    "handTool.highPressureWasherLance.lance#washMultiplier"), 1)
52  self.pricePerSecond = Utils.getNoNil(getXMLFloat(xmlFile,
    "handTool.highPressureWasherLance.lance#pricePerMinute"), 10) / 1000
53
54  setTranslation(self.rootNode, unpack(lancePosition))
55  setRotation(self.rootNode, unpack(lanceRotation))
56
57  self.effects = g_effectManager:loadEffect(xmlFile,
    "handTool.highPressureWasherLance.effects", self.rootNode, self)
58  g_effectManager:setFillType(self.effects, FillType.WATER)
59
60  -- Sounds
61  self.washingSample = g_soundManager:loadSampleFromXML(xmlFile,
    "handTool.highPressureWasherLance.sounds", "washing",
    self.baseDirectory, self.rootNode, 0, AudioGroup.VEHICLE, nil, self)
62
63  return true
64  end
```

delete

Description

Deleting

Definition

delete()

Return Values

table	instance	instance of object
integer	group	audio group

Code

```
68  function HighPressureWasherLance:delete()
69  g_effectManager:deleteEffects(self.effects)
70  g_soundManager:deleteSample(self.washingSample)
71  HighPressureWasherLance:superClass().delete(self)
72  end
```

onDeactivate

Description

Definition

onDeactivate()

Code

```
76  function HighPressureWasherLance:onDeactivate()
77  self:setIsWashing(false, true, true)
78  HighPressureWasherLance:superClass().onDeactivate(self)
79  end
```

update

Description

Definition

update()

Return Values

table new Landscaping instance

Code

```
 83  function HighPressureWasherLance:update(dt, allowInput)
 84  HighPressureWasherLance:superClass().update(self, dt)
 85
 86  if allowInput then
 87  self:setIsWashing(self.activatePressed, false, false)
 88  end
 89
 90  if self.isServer then
 91  if self.doWashing then
 92  self.foundVehicle = nil
 93  self:cleanVehicle(self.player.cameraNode, dt)
 94  if self.lanceRaycastNode ~= nil then
 95  self:cleanVehicle(self.lanceRaycastNode, dt)
 96  end
 97  if self.foundVehicle ~= nil then
 98  local farmId = self.foundVehicle:getOwnerFarmId()
 99  local price = self.pricePerSecond * (dt / 1000)
100  local stats = g_farmManager:getFarmById(self.player.farmId).stats
101  stats:updateStats("expenses", price)
102  g_currentMission:addMoney(-price, farmId, "vehicleRunningCost")
103  end
104  end
105  end
106  self.activatePressed = false
107  self:raiseActive()
108  end
```

setIsWashing

Description

Set is washing

Definition

setIsWashing(boolean doWashing, boolean force, boolean noEventSend)

Arguments

boolean doWashing	do washing
boolean force	force
boolean noEventSend	no event send

Return Values

table New LandscapingSculptEvent instance

Code

```
115  function HighPressureWasherLance:setIsWashing(doWashing, force, noEventSend)
116      HPWLanceStateEvent.sendEvent(self.player, doWashing, noEventSend)
117      if self.doWashing ~= doWashing then
118          if doWashing then
119              g_effectManager:setFillType(self.effects, FillType.WATER)
120              g_effectManager:startEffects(self.effects)
121              g_soundManager:playSample(self.washingSample)
122          else
123              if force then
124                  g_effectManager:resetEffects(self.effects)
125              else
126                  g_effectManager:stopEffects(self.effects)
127              end
128              g_soundManager:stopSample(self.washingSample)
129          end
130          self.doWashing = doWashing
131      end
132  end
```

cleanVehicle

Description

Clean vehicle

Definition

cleanVehicle(integer node, float dt)

Arguments

integer	node	node id
float	dt	time since last call in ms

Return Values

table	instance instance of object
integer	group audio group

Code

```
138  function HighPressureWasherLance:cleanVehicle(node, dt)
139      local x, y, z = getWorldTranslation(node)
140      local dx, dy, dz = localDirectionToWorld(node, 0, 0, -1)
```

```
141 local lastFoundVehicle = self.foundVehicle
142 raycastAll(x, y, z, dx, dy, dz, "washRaycastCallback",
    self.washDistance, self, 32+64+128+256+4096+8194)
143
144 if self.foundVehicle ~= nil and lastFoundVehicle ~=
    self.foundVehicle then
145 self.foundVehicle:addDirtAmount(-self.washMultiplier * dt /
    self.foundVehicle:getWashDuration())
146 end
147 end
```

washRaycastCallback

Description

Wash raycast callback

Definition

washRaycastCallback(integer hitActorId, float x, float y, float z, float distance, float nx, float ny, float nz, integer subShapeIndex, integer hitShapeId)

Arguments

integer	hitActorId	id of hit object actor
float	x	x raycast position
float	y	y raycast position
float	z	z raycast position
float	distance	distance to raycast position
float	nx	x component of hit surface normal (unused)
float	ny	y component of hit surface normal (unused)
float	nz	z component of hit surface normal (unused)
integer	subShapeIndex	sub shape index of hit object
integer	hitShapeId	id of hit object shape

Code

```
161 function HighPressureWasherLance:washRaycastCallback(hitActorId, x, y,
    z, distance, nx, ny, nz, subShapeIndex, hitShapeId)
162 local vehicle = g_currentMission.nodeToObject[hitActorId]
163 if hitActorId ~= hitShapeId then
164 -- object is a compoundChild. Try to find the compound
165 local parentId = hitShapeId
166 while parentId ~= 0 do
167 if g_currentMission.nodeToObject[parentId] ~= nil then
168 -- found valid compound
169 vehicle = g_currentMission.nodeToObject[parentId]
170 break
171 end
172 parentId = getParent(parentId)
173 end
174 end
```

```
175
176  if vehicle ~= nil and vehicle.getAllowsWashingByType ~= nil and
     vehicle:getAllowsWashingByType(Washable.WASHTYPE_HIGH_PRESSURE_WASHER)
     then
177  self.foundVehicle = vehicle
178  return false
179  end
180  return true
181  end
```

getIsActiveForInput

Description

Get is active for input

Definition

getIsActiveForInput()

Return Values

boolean isActivateable is activateable

boolean isActiveForInput is active for input

Code

```
187  function HighPressureWasherLance:getIsActiveForInput()
188  if self.player == g_currentMission.player and not
     g_gui:getIsGuiVisible() then
189  return true
190  end
191  return false
192  end
```

AIConveyorBeltSetAngleEvent

Description

Event for conveyor belt angle

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function AIConveyorBeltSetAngleEvent:emptyNew()
16  local self = Event:new(AIConveyorBeltSetAngleEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table vehicle, integer currentAngle)

Arguments

table vehicle vehicle

integer currentAngle current angle

Code

```
24  function AIConveyorBeltSetAngleEvent:new(vehicle, currentAngle)
25  local self = AIConveyorBeltSetAngleEvent:emptyNew()
26  self.currentAngle = currentAngle;
27  self.vehicle = vehicle;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function AIConveyorBeltSetAngleEvent:readStream(streamId,
    connection)
36  self.vehicle = NetworkUtil.readNodeObject(streamId);
37  self.currentAngle = streamReadInt8(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function AIConveyorBeltSetAngleEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.vehicle);
47  streamWriteInt8(streamId, self.currentAngle);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function AIConveyorBeltSetAngleEvent:run(connection)
54  self.vehicle:setAIConveyorBeltAngle(self.currentAngle, true);
55  if not connection:getIsServer() then
56  g_server:broadcastEvent(AIConveyorBeltSetAngleEvent:new(self.vehicle,
    self.currentAngle), nil, connection, self.vehicle);
57  end;
58  end;
```

AIVehicleIsBlockedEvent

Description

Event for ai block

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function AIVehicleIsBlockedEvent:emptyNew()
16  local self = Event:new(AIVehicleIsBlockedEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table object, boolean isBlocked)

Arguments

table object object

boolean isBlocked is blocked

Code

```
24  function AIVehicleIsBlockedEvent:new(object, isBlocked)
25  local self = AIVehicleIsBlockedEvent:emptyNew()
26  self.object = object
27  self.isBlocked = isBlocked
28  return self
```

```
29  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function AIVehicleIsBlockedEvent:readStream(streamId, connection)
36  self.object = NetworkUtil.readNodeObject(streamId)
37  self.isBlocked = streamReadBool(streamId)
38  self:run(connection)
39  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function AIVehicleIsBlockedEvent:writeStream(streamId, connection)
46  NetworkUtil.writeNodeObject(streamId, self.object)
47  streamWriteBool(streamId, self.isBlocked)
48  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function AIVehicleIsBlockedEvent:run(connection)
54  if self.object ~= nil then
55  if self.isBlocked then
56  self.object:aiBlock()
57  else
58  self.object:aiContinue()
```

```
59  end
60  end
61  end
```

AIVehicleSetStartedEvent

Description

Event for ai start

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function AIVehicleSetStartedEvent:emptyNew()
16  local self = Event:new(AIVehicleSetStartedEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table object, integer reason, boolean isStarted, integer helper)

Arguments

table	object	object
integer	reason	reason
boolean	isStarted	is started
integer	helper	helper id

Code

```
26  function AIVehicleSetStartedEvent:new(object, reason, isStarted,
    helper, startedFarmId)
27  local self = AIVehicleSetStartedEvent:emptyNew()
28  self.object = object
29  self.isStarted = isStarted
30  self.reason = reason
31  self.startedFarmId = startedFarmId
32  if helper ~= nil then
33  self.helperIndex = helper.index
34  end
35  return self
36  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
42  function AIVehicleSetStartedEvent:readStream(streamId, connection)
43  self.object = NetworkUtil.readNodeObject(streamId)
44  self.isStarted = streamReadBool(streamId)
45
46  if not self.isStarted then
47  self.reason = streamReadUIntN(streamId,
    AIVehicle.NUM_BITS_REASONS)
48  else
49  self.helperIndex = streamReadUInt8(streamId)
50  end
51
52  self.startedFarmId = streamReadUIntN(streamId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
53
54  self:run(connection)
55  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
61  function AIVehicleSetStartedEvent:writeStream(streamId,
    connection)
62  NetworkUtil.writeNodeObject(streamId, self.object)
63  streamWriteBool(streamId, self.isStarted)
64
65  if not self.isStarted then
66  streamWriteUIntN(streamId, self.reason,
    AIVehicle.NUM_BITS_REASONS)
67  else
```

```
68  streamWriteUInt8(streamId, self.helperIndex)
69  end
70
71  streamWriteUIntN(streamId, self.startedFarmId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
72  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
77  function AIVehicleSetStartedEvent:run(connection)
78  if self.isStarted then
79  self.object:startAIVehicle(self.helperIndex, true,
    self.startedFarmId)
80  else
81  self.object:stopAIVehicle(self.reason, true)
82  end
83  if not connection:getIsServer() then
84  for _, v in pairs(g_server.clientConnections) do
85  if v ~= connection and not v:getIsLocal() then
86  v:sendEvent(AIVehicleSetStartedEvent:new(self.object, self.reason,
    self.isStarted,
    g_helperManager:getHelperByIndex(self.helperIndex),
    self.startedFarmId))
87  end
88  end
89  end
90  end
```

AnimatedVehicleStartEvent

Description

Event for animation start

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function AnimatedVehicleStartEvent:emptyNew()
16  local self = Event:new(AnimatedVehicleStartEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, string name, float speed, float animTime)

Arguments

table	object	object
string	name	name of animation
float	speed	speed of animation
float	animTime	time of animation

Code

```
26  function AnimatedVehicleStartEvent:new(object, name, speed,
    animTime)
27  local self = AnimatedVehicleStartEvent:emptyNew()
28  self.name = name;
29  self.speed = speed;
30  self.animTime = animTime;
31  self.object = object;
32  return self;
33  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer	streamId	streamId
integer	connection	connection

Code

```
39  function AnimatedVehicleStartEvent:readStream(streamId,
    connection)
40  self.object = NetworkUtil.readNodeObject(streamId);
41  self.name = streamReadString(streamId);
42  self.speed = streamReadFloat32(streamId);
43  self.animTime = streamReadFloat32(streamId);
44
45  self.object:playAnimation(self.name, self.speed, self.animTime,
    true);
```

```
46  if not connection:getIsServer() then
47  g_server:broadcastEvent(AnimatedVehicleStartEvent:new(self.object,
    self.name, self.speed, self.animTime), nil, connection,
    self.object);
48  end;
49  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
55  function AnimatedVehicleStartEvent:writeStream(streamId,
    connection)
56  NetworkUtil.writeNodeObject(streamId, self.object);
57  streamWriteString(streamId, self.name);
58  streamWriteFloat32(streamId, self.speed);
59  streamWriteFloat32(streamId, self.animTime);
60  end;
```

AnimatedVehicleStopEvent

Description

Event for animation stop

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function AnimatedVehicleStopEvent:emptyNew()
16  local self = Event:new(AnimatedVehicleStopEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, string name)

Arguments

table object object
string name name

Code

```
24  function AnimatedVehicleStopEvent:new(object, name)
25  local self = AnimatedVehicleStopEvent:emptyNew()
26  self.name = name;
27  self.object = object;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId
integer connection connection

Code

```
35  function AnimatedVehicleStopEvent:readStream(streamId, connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.name = streamReadString(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId
integer connection connection

Code

```
45  function AnimatedVehicleStopEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteString(streamId, self.name);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function AnimatedVehicleStopEvent:run(connection)
54  AnimatedVehicle.stopAnimation(self.object, self.name, true);
55  if not connection:getIsServer() then
56  g_server:broadcastEvent(AnimatedVehicleStopEvent:new(self.object,
    self.name), nil, connection, self.object);
57  end;
58  end;
```

BaleLoaderStateEvent

Description

Event for bale loader state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function BaleLoaderStateEvent:emptyNew()
16  local self = Event:new(BaleLoaderStateEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, integer stateId, integer nearestBaleServerId)

Arguments

table	object	object
integer	stateId	stateId
integer	nearestBaleServerId	nearestBaleServerId

Code

```
25  function BaleLoaderStateEvent:new(object, stateId,
    nearestBaleServerId)
26  local self = BaleLoaderStateEvent:emptyNew()
27  self.object = object;
28  self.stateId = stateId;
29  assert(nearestBaleServerId ~= nil or self.stateId ~=
    BaleLoader.CHANGE_GRAB_BALE);
30  self.nearestBaleServerId = nearestBaleServerId;
```

```
31  return self;
32  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
38  function BaleLoaderStateEvent:readStream(streamId, connection)
39  self.object = NetworkUtil.readNodeObject(streamId);
40
41  self.stateId = streamReadInt8(streamId);
42  if self.stateId == BaleLoader.CHANGE_GRAB_BALE then
43  self.nearestBaleServerId = NetworkUtil.readNodeObjectId(streamId);
44  end;
45  self:run(connection);
46  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
52  function BaleLoaderStateEvent:writeStream(streamId, connection)
53  NetworkUtil.writeNodeObject(streamId, self.object);
54  streamWriteInt8(streamId, self.stateId);
55  if self.stateId == BaleLoader.CHANGE_GRAB_BALE then
56  NetworkUtil.writeNodeObjectId(streamId, self.nearestBaleServerId);
57  end;
58  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
63  function BaleLoaderStateEvent:run(connection)
64  self.object:doStateChange(self.stateId, self.nearestBaleServerId);
65  end;
```

BalerCreateBaleEvent

Description

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
14  function BalerCreateBaleEvent:emptyNew()
15  local self = Event:new(BalerCreateBaleEvent_mt);
16  return self;
17  end;
```

new

Description

Create new instance of event

Definition

new(table object, integer baleFillType, float baleTime)

Arguments

table	object	object
integer	baleFillType	bale fill type
float	baleTime	bale time

Code

```
24  function BalerCreateBaleEvent:new(object, baleFillType, baleTime)
25  local self = BalerCreateBaleEvent:emptyNew()
26  self.baleFillType = baleFillType;
27  self.baleTime = baleTime;
28  self.object = object;
29  return self;
30  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
36  function BalerCreateBaleEvent:readStream(streamId, connection)
37  self.object = NetworkUtil.readNodeObject(streamId);
38  self.baleTime = streamReadFloat32(streamId);
39  self.baleFillType = streamReadUIntN(streamId,
    FillTypeManager.SEND_NUM_BITS);
40  self:run(connection);
41  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
47  function BalerCreateBaleEvent:writeStream(streamId, connection)
48  NetworkUtil.writeNodeObject(streamId, self.object);
49  streamWriteFloat32(streamId, self.baleTime);
50  streamWriteUIntN(streamId, self.baleFillType,
    FillTypeManager.SEND_NUM_BITS);
51  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
56  function BalerCreateBaleEvent:run(connection)
57  self.object:createBale(self.baleFillType);
58  self.object:setBaleTime(table.getn(self.object.spec_baler.bales),
    self.baleTime);
59  end;
```

BalerSetBaleTimeEvent

Description

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
14  function BalerSetBaleTimeEvent:emptyNew()
15  local self = Event:new(BalerSetBaleTimeEvent_mt);
16  return self;
17  end;
```

BalerSetBaleTimeEvent:new

Description

Create new instance of event

Definition

BalerSetBaleTimeEvent:new(table object, integer bale, float baleTime)

Arguments

table	object	object
integer	bale	bale id
float	baleTime	bale time

Code

```
24  function BalerSetBaleTimeEvent:new(object, bale, baleTime)
25  local self = BalerSetBaleTimeEvent:emptyNew()
26  self.object = object;
27  self.bale = bale;
28  self.baleTime = baleTime;
29  return self;
30  end;
```

BalerSetBaleTimeEvent:readStream

Description

Called on client side on join

Definition

BalerSetBaleTimeEvent:readStream(integer streamId, integer connection)

Arguments

integer streamId	streamId
integer connection	connection

Code

```
36  function BalerSetBaleTimeEvent:readStream(streamId, connection)
37  self.object = NetworkUtil.readNodeObject(streamId);
38  self.bale = streamReadInt32(streamId);
39  self.baleTime = streamReadFloat32(streamId);
40  self:run(connection);
41  end;
```

BalerSetBaleTimeEvent:writeStream

Description

Called on server side on join

Definition

BalerSetBaleTimeEvent:writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
47  function BalerSetBaleTimeEvent:writeStream(streamId, connection)
48  NetworkUtil.writeNodeObject(streamId, self.object);
49  streamWriteInt32(streamId, self.bale);
50  streamWriteFloat32(streamId, self.baleTime);
51  end;
```

BalerSetBaleTimeEvent:run

Description

Run action on receiving side

Definition

BalerSetBaleTimeEvent:run(integer connection)

Arguments

integer connection connection

Code

```
56  function BalerSetBaleTimeEvent:run(connection)
57  self.object:setBaleTime(self.bale, self.baleTime);
58  end;
```

BalerSetIsUnloadingBaleEvent

Description

Event for baler is unloading state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function BalerSetIsUnloadingBaleEvent:emptyNew()
16  local self = Event:new(BalerSetIsUnloadingBaleEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, boolean isUnloadingBale)

Arguments

table object object

boolean isUnloadingBale is unloading bale

Code

```
24  function BalerSetIsUnloadingBaleEvent:new(object, isUnloadingBale)
25  local self = BalerSetIsUnloadingBaleEvent:emptyNew()
26  self.object = object;
27  self.isUnloadingBale = isUnloadingBale;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function BalerSetIsUnloadingBaleEvent:readStream(streamId,
    connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.isUnloadingBale = streamReadBool(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function BalerSetIsUnloadingBaleEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteBool(streamId, self.isUnloadingBale);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function BalerSetIsUnloadingBaleEvent:run(connection)
54  if not connection:getIsServer() then
55  g_server:broadcastEvent(self, false, connection, self.object);
56  end;
57  self.object:setIsUnloadingBale(self.isUnloadingBale, true);
58  end;
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table object, boolean isUnloadingBale, boolean noEventSend)

Arguments

table	object	object
boolean	isUnloadingBale	isUnloadingBale
boolean	noEventSend	no event send

Code

```
65  function BalerSetIsUnloadingBaleEvent.sendEvent(object, isUnloadingBale,
    noEventSend)
66  if noEventSend == nil or noEventSend == false then
67  if g_server ~= nil then
68  g_server:broadcastEvent(BalerSetIsUnloadingBaleEvent:new(object,
    isUnloadingBale), nil, nil, object);
69  else
70  g_client:getServerConnection():sendEvent(BalerSetIsUnloadingBaleEvent:new
    isUnloadingBale));
71  end;
72  end;
73  end;
```

BaleWrapperStateEvent

Description

Event for bale wrapper state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function BaleWrapperStateEvent:emptyNew()
16  local self = Event:new(BaleWrapperStateEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, integer stateId, integer nearestBaleServerId)

Arguments

table	object	object
integer	stateId	state id
integer	nearestBaleServerId	server id of nearest bale

Code

```
25  function BaleWrapperStateEvent:new(object, stateId,
    nearestBaleServerId)
26  local self = BaleWrapperStateEvent:emptyNew()
27  self.object = object;
28  self.stateId = stateId;
29  assert(nearestBaleServerId ~= nil or self.stateId ~=
    BaleWrapper.CHANGE_GRAB_BALE);
30  self.nearestBaleServerId = nearestBaleServerId;
31  return self;
32  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId	streamId
integer connection	connection

Code

```
38  function BaleWrapperStateEvent:readStream(streamId, connection)
39  self.object = NetworkUtil.readNodeObject(streamId);
40  self.stateId = streamReadInt8(streamId);
41  if self.stateId == BaleWrapper.CHANGE_GRAB_BALE then
42  self.nearestBaleServerId = NetworkUtil.readNodeObjectId(streamId);
43  end;
```

```
44  self:run(connection);
45  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId
integer connection connection

Code

```
51  function BaleWrapperStateEvent:writeStream(streamId, connection)
52  NetworkUtil.writeNodeObject(streamId, self.object);
53  streamWriteInt8(streamId, self.stateId);
54  if self.stateId == BaleWrapper.CHANGE_GRAB_BALE then
55  NetworkUtil.writeNodeObjectId(streamId, self.nearestBaleServerId);
56  end;
57  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
62  function BaleWrapperStateEvent:run(connection)
63  self.object:doStateChange(self.stateId, self.nearestBaleServerId);
64  end;
```

BunkerSiloCloseEvent

Description

Event for bunker silo close

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table instance instance of event
table self instance of class event

Code

```
13  function BunkerSiloCloseEvent:emptyNew()
```

```
14  local self = Event:new(BunkerSiloCloseEvent_mt);
15  return self;
16  end;
```

new

Description

Create new instance of event

Definition

new(table bunkerSilo)

Arguments

table bunkerSilo bunkerSilo

Return Values

table self instance of class event

table instance instance of event

Code

```
22  function BunkerSiloCloseEvent:new(bunkerSilo)
23  local self = BunkerSiloCloseEvent:emptyNew()
24  self.bunkerSilo = bunkerSilo;
25  return self;
26  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Return Values

table instance instance of event

Code

```
32  function BunkerSiloCloseEvent:readStream(streamId, connection)
33  if not connection:getIsServer() then
34  self.bunkerSilo = NetworkUtil.readNodeObject(streamId);
35  end;
36  self:run(connection);
37  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Return Values

table self instance of class event

Code

```
43  function BunkerSiloCloseEvent:writeStream(streamId, connection)
44  if connection:getIsServer() then
45  NetworkUtil.writeNodeObject(streamId, self.bunkerSilo);
46  end;
47  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Return Values

table instance instance of event

Code

```
52  function BunkerSiloCloseEvent:run(connection)
53  if not connection:getIsServer() then
54  self.bunkerSilo:setState(BunkerSilo.STATE_CLOSED);
55  end;
56  end;
```

BunkerSiloOpenEvent

Description

Event for bunker silo open

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

table self instance of class event

Code

```
13  function BunkerSiloOpenEvent:emptyNew()
14  local self = Event:new(BunkerSiloOpenEvent_mt);
15  return self;
16  end;
```

new

Description

Create new instance of event

Definition

new(table bunkerSilo, float x, float y, float z)

Arguments

table	bunkerSilo	bunkerSilo
float	x	x opening position
float	y	y opening position
float	z	z opening position

Return Values

table instance instance of event

table instance instance of event

Code

```
25  function BunkerSiloOpenEvent:new(bunkerSilo, x,y,z)
26  local self = BunkerSiloOpenEvent:emptyNew()
27  self.bunkerSilo = bunkerSilo;
28  self.x = x;
29  self.y = y;
30  self.z = z;
31  return self;
32  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer	streamId
integer	connection

Return Values

table self instance of class event

Code

```
38  function BunkerSiloOpenEvent:readStream(streamId, connection)
39  if not connection:getIsServer() then
40  self.bunkerSilo = NetworkUtil.readNodeObject(streamId);
41  self.x = streamReadFloat32(streamId);
42  self.y = streamReadFloat32(streamId);
43  self.z = streamReadFloat32(streamId);
44  end;
45  self:run(connection);
46  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Return Values

table instance instance of event

Code

```
52  function BunkerSiloOpenEvent:writeStream(streamId, connection)
53  if connection:getIsServer() then
54  NetworkUtil.writeNodeObject(streamId, self.bunkerSilo);
55  streamWriteFloat32(streamId, self.x);
56  streamWriteFloat32(streamId, self.y);
57  streamWriteFloat32(streamId, self.z);
58  end;
59  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Return Values

table motorInstance motor instance

Code

```
64  function BunkerSiloOpenEvent:run(connection)
65  if not connection:getIsServer() then
66  self.bunkerSilo:openSilo(self.x, self.y, self.z);
67  end;
68  end;
```

ChainsawCutEvent

Description

Event for cutting

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

boolean true if storeitem is configurable, else false

table self instance of class event

Code

```
14  function ChainsawCutEvent:emptyNew()
15  local self = Event:new(ChainsawCutEvent_mt);
16  return self;
17  end;
```

new

Description

Create new instance of event

Definition

new(integer splitShapeId, float x, float y, float z, float nx, float ny, float nz, float yx, float yy, float yz, float cutSizeY, float cutSizeZ)

Arguments

integer splitShapeId id of split shape

float	x	x
float	y	y
float	z	z
float	nx	nx
float	ny	ny
float	nz	nz
float	yx	yx
float	yy	yy
float	yz	yz
float	cutSizeY	y cut size
float	cutSizeZ	z cut size

Return Values

boolean true if storeitem is leaseable, else false

table instance instance of event

Code

```
34  function ChainsawCutEvent:new(splitShapeId, x,y,z, nx,ny,nz,
    yx,yy,yz, cutSizeY, cutSizeZ, farmId)
35  local self = ChainsawCutEvent:emptyNew()
36  self.splitShapeId = splitShapeId;
37  self.x,self.y,self.z = x,y,z;
38  self.nx,self.ny,self.nz = nx,ny,nz;
39  self.yx,self.yy,self.yz = yx,yy,yz;
40  self.cutSizeY,self.cutSizeZ = cutSizeY,cutSizeZ;
41  self.farmId = farmId
42  return self;
43  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Return Values

integer configId the default config id

Code

```
49  function ChainsawCutEvent:readStream(streamId, connection)
50  if not connection:getIsServer() then
51  local splitShapeId = readSplitShapeIdFromStream(streamId);
52  local x = streamReadFloat32(streamId);
53  local y = streamReadFloat32(streamId);
54  local z = streamReadFloat32(streamId);
55  local nx = streamReadFloat32(streamId);
56  local ny = streamReadFloat32(streamId);
57  local nz = streamReadFloat32(streamId);
58  local yx = streamReadFloat32(streamId);
59  local yy = streamReadFloat32(streamId);
60  local yz = streamReadFloat32(streamId);
61  local cutSizeY = streamReadFloat32(streamId);
62  local cutSizeZ = streamReadFloat32(streamId);
63  local farmId = streamReadUIntN(streamId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
64
65  if splitShapeId ~= 0 then
66  ChainsawUtil.cutSplitShape(splitShapeId, x,y,z, nx,ny,nz,
    yx,yy,yz, cutSizeY, cutSizeZ, farmId);
67  end
68  end
69  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Return Values

integer the default price

Code

```
75  function ChainsawCutEvent:writeStream(streamId, connection)
76  if connection:getIsServer() then
77  writeSplitShapeIdToStream(streamId, self.splitShapeId);
78  streamWriteFloat32(streamId, self.x);
79  streamWriteFloat32(streamId, self.y);
80  streamWriteFloat32(streamId, self.z);
81  streamWriteFloat32(streamId, self.nx);
82  streamWriteFloat32(streamId, self.ny);
83  streamWriteFloat32(streamId, self.nz);
84  streamWriteFloat32(streamId, self.yx);
85  streamWriteFloat32(streamId, self.yy);
86  streamWriteFloat32(streamId, self.yz);
87  streamWriteFloat32(streamId, self.cutSizeY);
88  streamWriteFloat32(streamId, self.cutSizeZ);
89  streamWriteFloat32(streamId, self.farmId)
90  end
91  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Return Values

integer the daily upkeep

Code

```
96  function ChainsawCutEvent:run(connection)
97  print("Error: ChainsawCutEvent is not allowed to be executed on a
    local client");
98  end;
```

ChainsawDelimbEvent

Description

Event for delimb

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

integer cost of the storeitem

table self instance of class event

Code

```
14  function ChainsawDelimbEvent:emptyNew()
15  local self = Event:new(ChainsawDelimbEvent_mt);
16  return self;
17  end;
```

new

Description

Create new instance of event

Definition

new(table player, float x, float y, float z, float nx, float ny, float nz, float yx, float yy, float yz, boolean onDelimb)

Arguments

table	player	player
float	x	x
float	y	y
float	z	z
float	nx	nx
float	ny	ny
float	nz	nz
float	yx	yx
float	yy	yy
float	yz	yz
boolean	onDelimb	on delimb

Return Values

table	config	object
table	instance	instance of event

Code

```
33  function ChainsawDelimbEvent:new(player, x,y,z, nx,ny,nz,
    yx,yy,yz, onDelimb)
34  local self = ChainsawDelimbEvent:emptyNew()
35  self.player = player;
36  self.x, self.y, self.z = x, y, z;
37  self.nx, self.ny, self.nz = nx, ny, nz;
38  self.yx, self.yy, self.yz = yx, yy, yz;
39  self.onDelimb = onDelimb;
40  return self;
41  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Return Values

table functions list of storeitem functions

Code

```
47  function ChainsawDelimbEvent:readStream(streamId, connection)
48  if not connection:getIsServer() then -- server side
49  self.player = NetworkUtil.readNodeObject(streamId);
50  self.x = streamReadFloat32(streamId);
51  self.y = streamReadFloat32(streamId);
52  self.z = streamReadFloat32(streamId);
53  self.nx = streamReadFloat32(streamId);
54  self.ny = streamReadFloat32(streamId);
55  self.nz = streamReadFloat32(streamId);
56  self.yx = streamReadFloat32(streamId);
57  self.yy = streamReadFloat32(streamId);
58  self.yz = streamReadFloat32(streamId);
59  self.onDelimb = false;
60  if self.player ~= nil then
61  local chainsaw = self.player.currentTool;
62  if chainsaw ~= nil then
63  local ret =
    findAndRemoveSplitShapeAttachments(self.x,self.y,self.z,
    self.nx,self.ny,self.nz, self.yx,self.yy,self.yz, 0.7,
    chainsaw.cutSizeY, chainsaw.cutSizeZ);
64  if ret then
65  self.onDelimb = true;
66  connection:sendEvent(self);
67  end;
68  end;
69  end;
70  else -- client side
71  self.player = NetworkUtil.readNodeObject(streamId);
72  self.onDelimb = streamReadBool(streamId);
73  if self.player ~= nil and self.player.currentTool ~= nil then
74  if self.player.currentTool.setOnDelimb ~= nil then
75  self.player.currentTool:setOnDelimb(self.onDelimb);
76  end;
77  end;
78  end
79  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Return Values

table specs list of storeitem specs

Code

```
 85  function ChainsawDelimbEvent:writeStream(streamId, connection)
 86  if connection:getIsServer() then -- client
 87  NetworkUtil.writeNodeObject(streamId, self.player);
 88  streamWriteFloat32(streamId, self.x);
 89  streamWriteFloat32(streamId, self.y);
 90  streamWriteFloat32(streamId, self.z);
 91  streamWriteFloat32(streamId, self.nx);
 92  streamWriteFloat32(streamId, self.ny);
 93  streamWriteFloat32(streamId, self.nz);
 94  streamWriteFloat32(streamId, self.yx);
 95  streamWriteFloat32(streamId, self.yy);
 96  streamWriteFloat32(streamId, self.yz);
 97  else
 98  NetworkUtil.writeNodeObject(streamId, self.player);
 99  streamWriteBool(streamId, self.onDelimb);
100  end
101  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Return Values

integer brandIndex the brandindex

Code

```
106  function ChainsawDelimbEvent:run(connection)
107  print("Error: ChainsawDelimbEvent is not allowed to be executed
     on a local client");
108  end;
```

ChainsawStateEvent

Description

Event for chainsaw state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table instance instance of object
table self instance of class event

Code

```
14  function ChainsawStateEvent:emptyNew()
15  local self = Event:new(ChainsawStateEvent_mt)
16  return self
17  end
```

new

Description

Create new instance of event

Definition

new(table player, boolean isCutting, boolean isHorizontalCut)

Arguments

table player player
boolean isCutting is cutting
boolean isHorizontalCut is horizontal cutting

Return Values

boolean true if loading was successful else false
table instance instance of event

Code

```
25  function ChainsawStateEvent:new(player, isCutting,
    isHorizontalCut, hasBeencut)
26  local self = ChainsawStateEvent:emptyNew()
27  self.player = player
28  self.isCutting = isCutting
29  self.isHorizontalCut = isHorizontalCut
30  self.hasBeenCut = hasBeencut
31  return self
32  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Return Values

table instance instance of object

Code

```
38  function ChainsawStateEvent:readStream(streamId, connection)
39  self.player = NetworkUtil.readNodeObject(streamId)
40  self.isCutting = streamReadBool(streamId)
41  self.isHorizontalCut = streamReadBool(streamId)
42  self.hasBeenCut = streamReadBool(streamId)
43  self:run(connection)
44  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Return Values

boolean true if loading was successful else false

Code

```
50  function ChainsawStateEvent:writeStream(streamId, connection)
51  NetworkUtil.writeNodeObject(streamId, self.player)
52  streamWriteBool(streamId, self.isCutting)
53  streamWriteBool(streamId, self.isHorizontalCut)
54  streamWriteBool(streamId, self.hasBeenCut)
55  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Return Values

boolean true if loading was successful else false

Code

```
60  function ChainsawStateEvent:run(connection)
61  if not connection:getIsServer() then
62  g_server:broadcastEvent(self, false, connection, self.player)
```

```
63  end
64
65  local currentTool = self.player.baseInformation.currentHandtool
66  if currentTool ~= nil and currentTool.setCutting ~= nil then
67  currentTool:setCutting(self.isCutting, self.isHorizontalCut,
    self.hasBeenCut, true)
68  end
69  end
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table player, boolean isCutting, boolean isHorizontalCut, boolean noEventSend)

Arguments

table	player	player
boolean	isCutting	is cutting
boolean	isHorizontalCut	is horizontal cutting
boolean	noEventSend	no event send

Return Values

table sample sample object

Code

```
77  function ChainsawStateEvent.sendEvent(player, isCutting,
    isHorizontalCut, hasBeenCut, noEventSend)
78  local currentTool = player.baseInformation.currentHandtool
79  if currentTool ~= nil and currentTool.setCutting ~= nil and
    (currentTool.isCutting ~= isCutting or currentTool.hasBeenCut ~=
    hasBeenCut) then
80  if noEventSend == nil or noEventSend == false then
81  if g_server ~= nil then
82  g_server:broadcastEvent(ChainsawStateEvent:new(player, isCutting,
    isHorizontalCut, hasBeenCut), nil, nil, player)
83  else
84  g_client:getServerConnection():sendEvent(ChainsawStateEvent:new(player,
    isCutting, isHorizontalCut, hasBeenCut))
85  end
86  end
87  end
88  end
```

CombineStrawEnableEvent

Description

Event for straw enable state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function CombineStrawEnableEvent:emptyNew()
16  local self = Event:new(CombineStrawEnableEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table vehicle, boolean isSwathActive)

Arguments

table vehicle vehicle

boolean isSwathActive is straw enabled

Code

```
24  function CombineStrawEnableEvent:new(vehicle, isSwathActive)
25  local self = CombineStrawEnableEvent:emptyNew()
26  self.vehicle = vehicle;
27  self.isSwathActive = isSwathActive;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function CombineStrawEnableEvent:readStream(streamId, connection)
36  self.vehicle = NetworkUtil.readNodeObject(streamId);
37  self.isSwathActive = streamReadBool(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function CombineStrawEnableEvent:writeStream(streamId, connection)
46  NetworkUtil.writeNodeObject(streamId, self.vehicle);
47  streamWriteBool(streamId, self.isSwathActive);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function CombineStrawEnableEvent:run(connection)
54  self.vehicle:setIsSwathActive(self.isSwathActive, true);
55  if not connection:getIsServer() then
56  g_server:broadcastEvent(CombineStrawEnableEvent:new(self.vehicle,
    self.isSwathActive), nil, connection, self.vehicle);
57  end;
58  end;
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, boolean isSwathActive, boolean noEventSend)

Arguments

table vehicle vehicle

boolean isSwathActive is straw enabled

boolean noEventSend no event send

Code

```
65  function CombineStrawEnableEvent.sendEvent(vehicle, isSwathActive,
    noEventSend)
66  if noEventSend == nil or noEventSend == false then
67  if g_server ~= nil then
68  g_server:broadcastEvent(CombineStrawEnableEvent:new(vehicle, isSwathActiv
    nil, nil, vehicle);
69  else
```

```
70  g_client:getServerConnection():sendEvent(CombineStrawEnableEvent:new(vehi
    isSwathActive));
71  end;
72  end;
73  end;
```

CylinderedEasyControlChangeEvent

Description

Event for straw enable state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function CylinderedEasyControlChangeEvent:emptyNew()
16  local self = Event:new(CylinderedEasyControlChangeEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table vehicle, boolean isEasyControlActive)

Arguments

table vehicle vehicle

boolean isEasyControlActive is easy control enabled

Code

```
24  function CylinderedEasyControlChangeEvent:new(vehicle,
    isEasyControlActive)
25  local self = CylinderedEasyControlChangeEvent:emptyNew()
26  self.vehicle = vehicle
27  self.isEasyControlActive = isEasyControlActive
28  return self
29  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function CylinderedEasyControlChangeEvent:readStream(streamId,
    connection)
36  self.vehicle = NetworkUtil.readNodeObject(streamId)
37  self.isEasyControlActive = streamReadBool(streamId)
38  self:run(connection)
39  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function CylinderedEasyControlChangeEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.vehicle)
47  streamWriteBool(streamId, self.isEasyControlActive)
48  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function CylinderedEasyControlChangeEvent:run(connection)
54  if self.vehicle ~= nil then
55  self.vehicle:setIsEasyControlActive(self.isEasyControlActive, true)
56  end
57  if not connection:getIsServer() then
58  g_server:broadcastEvent(CylinderedEasyControlChangeEvent:new(self.vehicle
    self.isEasyControlActive), nil, connection, self.vehicle)
59  end
60  end
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, boolean isEasyControlActive, boolean noEventSend)

Arguments

table	vehicle	vehicle
boolean	isEasyControlActive	is easy control enabled
boolean	noEventSend	no event send

Code

```
67  function CylinderedEasyControlChangeEvent.sendEvent(vehicle, isEasyContro
    noEventSend)
68  if noEventSend == nil or noEventSend == false then
69  if g_server ~= nil then
70  g_server:broadcastEvent(CylinderedEasyControlChangeEvent:new(vehicle,
    isEasyControlActive), nil, nil, vehicle)
71  elseif g_client ~= nil then
72  g_client:getServerConnection():sendEvent(CylinderedEasyControlChangeEvent
    isEasyControlActive))
73  end
74  end
75  end
```

DrivableToggleLowerAllEvent

Description

Event for toggle lower all

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function DrivableToggleLowerAllEvent:emptyNew()
16  local self = Event:new(DrivableToggleLowerAllEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table vehicle)

Arguments

table vehicle vehicle

Code

```
23  function DrivableToggleLowerAllEvent:new(vehicle)
24  local self = DrivableToggleLowerAllEvent:emptyNew()
25  self.vehicle = vehicle;
26  return self;
27  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
33  function DrivableToggleLowerAllEvent:readStream(streamId,
    connection)
34  self.vehicle = NetworkUtil.readNodeObject(streamId);
35  self:run(connection);
36  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
42  function DrivableToggleLowerAllEvent:writeStream(streamId,
    connection)
43  NetworkUtil.writeNodeObject(streamId, self.vehicle);
44  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
49  function DrivableToggleLowerAllEvent:run(connection)
50  self.vehicle:toggleLowerAllImplements(true);
```

```
51  if not connection:getIsServer() then
52  g_server:broadcastEvent(DrivableToggleLowerAllEvent:new(self.vehicle),
    nil, connection, self.object);
53  end;
54  end;
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, boolean noEventSend)

Arguments

table vehicle vehicle

boolean noEventSend no event send

Code

```
60  function DrivableToggleLowerAllEvent.sendEvent(vehicle, noEventSend)
61  if noEventSend == nil or noEventSend == false then
62  if g_server ~= nil then
63  g_server:broadcastEvent(DrivableToggleLowerAllEvent:new(vehicle), nil, ni
    vehicle);
64  else
65  g_client:getServerConnection():sendEvent(DrivableToggleLowerAllEvent:new
66  end;
67  end;
68  end;
```

FillUnitUnloadEvent

Description

Event for turned on state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function FillUnitUnloadEvent:emptyNew()
16  local self = Event:new(FillUnitUnloadEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table object)

Arguments

table object object

Code

```
23  function FillUnitUnloadEvent:new(object)
24  local self = FillUnitUnloadEvent:emptyNew()
25  self.object = object
26  return self
27  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
38  function FillUnitUnloadEvent:readStream(streamId, connection)
39  if not connection:getIsServer() then
40  self.object = NetworkUtil.readNodeObject(streamId)
41  end
42  self:run(connection)
43  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
49  function FillUnitUnloadEvent:writeStream(streamId, connection)
50  if connection:getIsServer() then
51  NetworkUtil.writeNodeObject(streamId, self.object)
52  end
53  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
58  function FillUnitUnloadEvent:run(connection)
59  if not connection:getIsServer() then
60  if self.object ~= nil then
61  local success = self.object:unloadFillUnits(true)
62  if not success then
63  connection:sendEvent(FillUnitUnloadEvent:newServerToClient())
64  end
65  end
66  else
67  g_currentMission:addIngameNotification(FSBaseMission.INGAME_NOTIFICATION_
    g_i18n:getText("fillUnit_unload_nospace"))
68  end
69  end
```

FoldableSetFoldDirectionEvent

Description

Event for set folding direction

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function FoldableSetFoldDirectionEvent:emptyNew()
16  local self = Event:new(FoldableSetFoldDirectionEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, integer direction, boolean moveToMiddle)

Arguments

table	object	object
integer	direction	direction
boolean	moveToMiddle	move to middle

Code

```
25  function FoldableSetFoldDirectionEvent:new(object, direction,
    moveToMiddle)
26  local self = FoldableSetFoldDirectionEvent:emptyNew()
27  self.object = object;
28  self.direction = MathUtil.sign(direction);
29  self.moveToMiddle = moveToMiddle;
30  return self;
31  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
37  function FoldableSetFoldDirectionEvent:readStream(streamId,
    connection)
38  self.object = NetworkUtil.readNodeObject(streamId);
39  self.direction = streamReadUIntN(streamId, 2)-1;
40  self.moveToMiddle = streamReadBool(streamId);
41  self:run(connection);
42  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
48  function FoldableSetFoldDirectionEvent:writeStream(streamId,
    connection)
49  NetworkUtil.writeNodeObject(streamId, self.object);
50  streamWriteUIntN(streamId, self.direction+1, 2);
51  streamWriteBool(streamId, self.moveToMiddle);
52  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
57  function FoldableSetFoldDirectionEvent:run(connection)
58  if self.object ~= nil then
59  self.object:setFoldState(self.direction, self.moveToMiddle, true);
60  end;
61  if not connection:getIsServer() then
62  g_server:broadcastEvent(FoldableSetFoldDirectionEvent:new(self.object,
    self.direction, self.moveToMiddle), nil, connection, self.object);
63  end;
64  end;
```

GreenhouseSetIsWaterTankFillingEvent

Description

Event for greenhouse tank filling state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function GreenhouseSetIsWaterTankFillingEvent:emptyNew()
16  local self = Event:new(GreenhouseSetIsWaterTankFillingEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, boolean isFilling, table trailer)

Arguments

table object object

boolean isFilling is filling

table trailer trailer

Return Values

table instance instance of event

Code

```
26  function GreenhouseSetIsWaterTankFillingEvent:new(object,
    isFilling, trailer)
```

```
27  local self = GreenhouseSetIsWaterTankFillingEvent:emptyNew()
28  self.object = object;
29  self.isFilling = isFilling;
30  self.trailer = trailer;
31  return self;
32  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
38  function GreenhouseSetIsWaterTankFillingEvent:readStream(streamId,
    connection)
39  self.object = NetworkUtil.readNodeObject(streamId);
40  self.isFilling = streamReadBool(streamId);
41  if self.isFilling and not connection:getIsServer() then
42  self.trailer = NetworkUtil.readNodeObject(streamId);
43  end;
44  self:run(connection);
45  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
51  function
    GreenhouseSetIsWaterTankFillingEvent:writeStream(streamId,
    connection)
52  NetworkUtil.writeNodeObject(streamId, self.object);
53  streamWriteBool(streamId, self.isFilling);
54  if self.isFilling and connection:getIsServer() then
55  NetworkUtil.writeNodeObject(streamId, self.trailer);
56  end;
57  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
62  function GreenhouseSetIsWaterTankFillingEvent:run(connection)
63  if not connection:getIsServer() then
64  g_server:broadcastEvent(self, false, connection, self.object);
65  end;
66  self.object:setIsWaterTankFilling(self.isFilling, self.trailer,
    true);
67  end;
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table object, boolean isFilling, table trailer, boolean noEventSend)

Arguments

table	object	object
boolean	isFilling	is filling
table	trailer	trailer
boolean	noEventSend	no event send

Code

```
75  function GreenhouseSetIsWaterTankFillingEvent.sendEvent(object, isFilling
    noEventSend)
76  if isFilling ~= object.isWaterTankFilling then
77  if noEventSend == nil or noEventSend == false then
78  if g_server ~= nil then
79  g_server:broadcastEvent(GreenhouseSetIsWaterTankFillingEvent:new(object,
    trailer), nil, nil, object);
80  else
81  assert(not isFilling or (trailer ~= nil));
82  g_client:getServerConnection():sendEvent(GreenhouseSetIsWaterTankFillingE
    isFilling, trailer));
83  end;
84  end;
85  end;
86  end;
```

HonkEvent

Description

Event for honking

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function HonkEvent:emptyNew()
16  local self = Event:new(HonkEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, boolean isPlaying)

Arguments

table object object

boolean isPlaying honk is playing

Code

```
24  function HonkEvent:new(object, isPlaying)
25  local self = HonkEvent:emptyNew()
26  self.object = object;
27  self.isPlaying = isPlaying;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function HonkEvent:readStream(streamId, connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.isPlaying = streamReadBool(streamId);
38  self:run(connection);
```

```
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer	streamId	streamId
integer	connection	connection

Code

```
45  function HonkEvent:writeStream(streamId, connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteBool(streamId, self.isPlaying);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer	connection	connection

Code

```
53  function HonkEvent:run(connection)
54  self.object:playHonk(self.isPlaying, true);
55  if not connection:getIsServer() then
56  g_server:broadcastEvent(HonkEvent:new(self.object,
    self.isPlaying), nil, connection, self.object);
57  end;
58  end;
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, boolean isPlaying, boolean noEventSend)

Arguments

table	vehicle	vehicle
boolean	isPlaying	honk is playing
boolean	noEventSend	no event send

Code

```
65  function HonkEvent.sendEvent(vehicle, isPlaying, noEventSend)
66  if vehicle.spec_honk ~= nil and vehicle.spec_honk.isPlaying ~=
    isPlaying then
```

```
67  if noEventSend == nil or noEventSend == false then
68  if g_server ~= nil then
69  g_server:broadcastEvent(HonkEvent:new(vehicle, isPlaying), nil,
    nil, vehicle);
70  else
71  g_client:getServerConnection():sendEvent(HonkEvent:new(vehicle,
    isPlaying));
72  end;
73  end;
74  end;
75  end;
```

HPWLanceStateEvent

Description

Event for hpw state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table	instance	instance of object
integer	group	audio group
table	self	instance of class event

Code

```
15  function HPWLanceStateEvent:emptyNew()
16  local self = Event:new(HPWLanceStateEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table object, boolean doWashing)

Arguments

table	object	object
boolean	doWashing	do washing

Return Values

table instance instance of event

Code

```
25  function HPWLanceStateEvent:new(player, doWashing)
26  local self = HPWLanceStateEvent:emptyNew()
27  self.player = player
```

```
28  self.doWashing = doWashing
29  return self
30  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Return Values

table instance instance of basket trigger object

Code

```
36  function HPWLanceStateEvent:readStream(streamId, connection)
37  self.player = NetworkUtil.readNodeObject(streamId)
38  self.doWashing = streamReadBool(streamId)
39  self:run(connection)
40  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Return Values

boolean success success

Code

```
46  function HPWLanceStateEvent:writeStream(streamId, connection)
47  NetworkUtil.writeNodeObject(streamId, self.player)
48  streamWriteBool(streamId, self.doWashing)
49  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Return Values

table self returns the instance

Code

```
54  function HPWLanceStateEvent:run(connection)
55  if not connection:getIsServer() then
56  g_server:broadcastEvent(self, false, connection, self.player)
57  end
58  local currentTool = self.player.baseInformation.currentHandtool
59  if currentTool ~= nil and currentTool.setIsWashing ~= nil then
60  currentTool:setIsWashing(self.doWashing, false, true)
61  end
62  end
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table object, boolean doWashing, boolean noEventSend)

Arguments

table	object	object
boolean	doWashing	do washing
boolean	noEventSend	no event send

Return Values

bool true if level has changed

Code

```
69  function HPWLanceStateEvent.sendEvent(player, doWashing, noEventSend)
70  local currentTool = player.baseInformation.currentHandtool
71  if currentTool ~= nil and currentTool.setIsWashing ~= nil and doWashing
    ~= currentTool.doWashing then
72  if noEventSend == nil or noEventSend == false then
73  if g_server ~= nil then
74  g_server:broadcastEvent(HPWLanceStateEvent:new(player, doWashing), nil,
    nil, player)
75  else
76  g_client:getServerConnection():sendEvent(HPWLanceStateEvent:new(player,
    doWashing))
77  end
78  end
79  end
80  end
```

HPWPlaceableTurnOnEvent

Description

Event for hpw turn on state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function HPWPlaceableTurnOnEvent:emptyNew()
16  local self = Event:new(HPWPlaceableTurnOnEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table object, boolean isTurnedOn, table player)

Arguments

table	object	object
boolean	isTurnedOn	is turned on
table	player	player

Return Values

table instance instance of event

Code

```
26  function HPWPlaceableTurnOnEvent:new(object, isTurnedOn, player)
27  local self = HPWPlaceableTurnOnEvent:emptyNew()
28  self.object = object
29  self.isTurnedOn = isTurnedOn
30  self.player = player
31
32  return self
33  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer	streamId
integer	connection

Code

```
39  function HPWPlaceableTurnOnEvent:readStream(streamId, connection)
40  self.object = NetworkUtil.readNodeObject(streamId)
```

```
41  self.isTurnedOn = streamReadBool(streamId)
42  if self.isTurnedOn then
43  self.player = NetworkUtil.readNodeObject(streamId)
44  end
45  self:run(connection)
46  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
52  function HPWPlaceableTurnOnEvent:writeStream(streamId, connection)
53  NetworkUtil.writeNodeObject(streamId, self.object)
54  streamWriteBool(streamId, self.isTurnedOn)
55  if self.isTurnedOn then
56  NetworkUtil.writeNodeObject(streamId, self.player)
57  end
58  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
63  function HPWPlaceableTurnOnEvent:run(connection)
64  if not connection:getIsServer() then
65  g_server:broadcastEvent(self, false, connection, self.object)
66  end
67  self.object:setIsTurnedOn(self.isTurnedOn, self.player, true)
68  end
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table object, boolean isTurnedOn, table player, boolean noEventSend)

Arguments

table	object	object
boolean	isTurnedOn	is turned on
table	player	player
boolean	noEventSend	no event send

Code

```
76  function HPWPlaceableTurnOnEvent.sendEvent(object, isTurnedOn, player, noEventSend)
77      if isTurnedOn ~= object.isTurnedOn then
78          if noEventSend == nil or noEventSend == false then
79              if g_server ~= nil then
80                  g_server:broadcastEvent(HPWPlaceableTurnOnEvent:new(object, isTurnedOn, player), nil, nil, object)
81              else
82                  g_client:getServerConnection():sendEvent(HPWPlaceableTurnOnEvent:new(obje isTurnedOn, player))
83              end
84          end
85      end
86  end
```

JumpEvent

Description

Event for honking

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function JumpEvent:emptyNew()
16      local self = Event:new(JumpEvent_mt)
17      return self
18  end
```

new

Description

Create new instance of event

Definition

new(table object, boolean isPlaying)

Arguments

table	object	object
boolean	isPlaying	honk is playing

Code

```
24  function JumpEvent:new(object)
25  local self = JumpEvent:emptyNew()
26  self.object = object
27  return self
28  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
34  function JumpEvent:readStream(streamId, connection)
35  if not connection:getIsServer() then
36  self.object = NetworkUtil.readNodeObject(streamId)
37  self:run(connection)
38  end
39  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function JumpEvent:writeStream(streamId, connection)
46  if connection:getIsServer() then
47  NetworkUtil.writeNodeObject(streamId, self.object)
48  end
49  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
54  function JumpEvent:run(connection)
55  self.object:jump(true)
56  end
```

MixerWagonBaleNotAcceptedEvent

Description

Event for honking

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function MixerWagonBaleNotAcceptedEvent:emptyNew()
16  local self = Event:new(MixerWagonBaleNotAcceptedEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table vehicle)

Arguments

table vehicle vehicle

Code

```
23  function MixerWagonBaleNotAcceptedEvent:new(vehicle)
24  local self = MixerWagonBaleNotAcceptedEvent:emptyNew()
25  self.vehicle = vehicle
26  return self
27  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
33  function MixerWagonBaleNotAcceptedEvent:readStream(streamId, connection)
```

```
34  self.vehicle = NetworkUtil.readNodeObject(streamId)
35  g_currentMission:addIngameNotification(FSBaseMission.INGAME_NOTIFICATION_
    g_i18n:getText("warning_baleNotSupported"))
36  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
42  function MixerWagonBaleNotAcceptedEvent:writeStream(streamId,
    connection)
43  NetworkUtil.writeNodeObject(streamId, self.vehicle)
44  end
```

MowerToggleWindrowDropEvent

Description

Event for mower toggle drop

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function MowerToggleWindrowDropEvent:emptyNew()
16  local self = Event:new(MowerToggleWindrowDropEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, boolean useMowerWindrowDropAreas)

Arguments

table object object

boolean useMowerWindrowDropAreas use mower windrow drop areas

Code

```
24  function MowerToggleWindrowDropEvent:new(object,
    useMowerWindrowDropAreas)
```

```
25  local self = MowerToggleWindrowDropEvent:emptyNew()
26  self.object = object;
27  self.useMowerWindrowDropAreas = useMowerWindrowDropAreas;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function MowerToggleWindrowDropEvent:readStream(streamId,
    connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.useMowerWindrowDropAreas = streamReadBool(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function MowerToggleWindrowDropEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteBool(streamId, self.useMowerWindrowDropAreas);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function MowerToggleWindrowDropEvent:run(connection)
54  if not connection:getIsServer() then
55  g_server:broadcastEvent(self, false, connection, self.object);
56  end;
57  if self.object ~= nil then
58  self.object:setUseMowerWindrowDropAreas(self.useMowerWindrowDropAreas,
    true);
59  end;
60  end;
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, boolean useMowerWindrowDropAreas, boolean noEventSend)

Arguments

table	vehicle	vehicle
boolean	useMowerWindrowDropAreas	use mower windrow drop areas
boolean	noEventSend	no event send

Code

```
67  function MowerToggleWindrowDropEvent.sendEvent(vehicle, useMowerWindrowDr
    noEventSend)
68  if useMowerWindrowDropAreas ~= vehicle.useMowerWindrowDropAreas then
69  if noEventSend == nil or noEventSend == false then
70  if g_server ~= nil then
71  g_server:broadcastEvent(MowerToggleWindrowDropEvent:new(vehicle,
    useMowerWindrowDropAreas), nil, nil, vehicle);
72  else
73  g_client:getServerConnection():sendEvent(MowerToggleWindrowDropEvent:new
    useMowerWindrowDropAreas));
74  end;
75  end;
76  end;
77  end;
```

PickupSetStateEvent

Description

Event for lower and lift pickup

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function PickupSetStateEvent:emptyNew()
16  local self = Event:new(PickupSetStateEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, boolean isPickupLowered)

Arguments

table object object

boolean isPickupLowered is pickup lowered

Code

```
24  function PickupSetStateEvent:new(object, isPickupLowered)
25  local self = PickupSetStateEvent:emptyNew()
26  self.object = object;
27  self.isPickupLowered = isPickupLowered;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function PickupSetStateEvent:readStream(streamId, connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.isPickupLowered = streamReadBool(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function PickupSetStateEvent:writeStream(streamId, connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteBool(streamId, self.isPickupLowered);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function PickupSetStateEvent:run(connection)
54  if not connection:getIsServer() then
55  g_server:broadcastEvent(self, false, connection, self.object);
56  end;
57  if self.object ~= nil then
58  self.object:setPickupState(self.isPickupLowered, true);
59  end;
60  end;
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, boolean isPickupLowered, boolean noEventSend)

Arguments

table	vehicle	vehicle
boolean	isPickupLowered	is pickup lowered
boolean	noEventSend	no event send

Code

```
67  function PickupSetStateEvent.sendEvent(vehicle, isPickupLowered,
    noEventSend)
68  if isPickupLowered ~= vehicle.spec_pickup.isLowered then
69  if noEventSend == nil or noEventSend == false then
70  if g_server ~= nil then
71  g_server:broadcastEvent(PickupSetStateEvent:new(vehicle,
    isPickupLowered), nil, nil, vehicle);
72  else
73  g_client:getServerConnection():sendEvent(PickupSetStateEvent:new(vehicle,
    isPickupLowered));
```

```
74  end;
75  end;
76  end;
77  end;
```

PlantLimitToFieldEvent

Description

Event for limit to field state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function PlantLimitToFieldEvent:emptyNew()
16  local self = Event:new(PlantLimitToFieldEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, boolean plantLimitToField)

Arguments

table object object

boolean plantLimitToField plant is limited to field

Code

```
24  function PlantLimitToFieldEvent:new(object, plantLimitToField)
25  local self = PlantLimitToFieldEvent:emptyNew()
26  self.object = object;
27  self.plantLimitToField = plantLimitToField;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function PlantLimitToFieldEvent:readStream(streamId, connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.plantLimitToField = streamReadBool(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function PlantLimitToFieldEvent:writeStream(streamId, connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteBool(streamId, self.plantLimitToField);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function PlantLimitToFieldEvent:run(connection)
54  self.object:setPlantLimitToField(self.plantLimitToField, true);
55  if not connection:getIsServer() then
56  g_server:broadcastEvent(PlantLimitToFieldEvent:new(self.object,
    self.plantLimitToField), nil, connection, self.object);
57  end;
58  end;
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, boolean isPickupLowered, boolean noEventSend)

Arguments

table vehicle vehicle

boolean isPickupLowered is pickup lowered
boolean noEventSend no event send

Code

```
65  function PlantLimitToFieldEvent.sendEvent(vehicle, plantLimitToField,
    noEventSend)
66  if noEventSend == nil or noEventSend == false then
67  if g_server ~= nil then
68  g_server:broadcastEvent(PlantLimitToFieldEvent:new(vehicle,
    plantLimitToField), nil, nil, vehicle);
69  else
70  g_client:getServerConnection():sendEvent(PlantLimitToFieldEvent:new(vehic
    plantLimitToField));
71  end;
72  end;
73  end;
```

PlayerPermissionsEvent

Description

sendEvent

Description

Create an instance

Definition

sendEvent(table player, integer farmId, bool noEventSend)

Arguments

table player player instance
integer farmId farm identification
bool noEventSend if false will send the event

Code

```
88  function PlayerPermissionsEvent.sendEvent(userId, permissions,
    isFarmManager, noEventSend)
89  if noEventSend == nil or noEventSend == false then
90  local event = PlayerPermissionsEvent:new(userId, permissions,
    isFarmManager)
91
92  if g_server ~= nil then
93  local farm = g_farmManager:getFarmByUserId(userId)
94  local player = farm.userIdToPlayer[userId]
95
96  g_server:broadcastEvent(event, nil, nil, player)
97  else
98  g_client:getServerConnection():sendEvent(event)
99  end
100 end
101 end
```

PlowLimitToFieldEvent
Description

Event for limit to field state

emptyNew
Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function PlowLimitToFieldEvent:emptyNew()
16  local self = Event:new(PlowLimitToFieldEvent_mt);
17  return self;
18  end;
```

new
Description

Create new instance of event

Definition

new(table object, boolean plowLimitToField)

Arguments

table object object

boolean plowLimitToField plow is limited to field

Code

```
24  function PlowLimitToFieldEvent:new(object, plowLimitToField)
25  local self = PlowLimitToFieldEvent:emptyNew()
26  self.object = object;
27  self.plowLimitToField = plowLimitToField;
28  return self;
29  end;
```

readStream
Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function PlowLimitToFieldEvent:readStream(streamId, connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.plowLimitToField = streamReadBool(streamId);
```

```
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function PlowLimitToFieldEvent:writeStream(streamId, connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteBool(streamId, self.plowLimitToField);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function PlowLimitToFieldEvent:run(connection)
54  self.object:setPlowLimitToField(self.plowLimitToField, true);
55  if not connection:getIsServer() then
56  g_server:broadcastEvent(PlowLimitToFieldEvent:new(self.object,
    self.plowLimitToField), nil, connection, self.object);
57  end;
58  end;
```

PlowRotationEvent

Description

Event for plow rotation

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function PlowRotationEvent:emptyNew()
```

```
16  local self = Event:new(PlowRotationEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, boolean rotationMax)

Arguments

table object object

boolean rotationMax rotation max

Code

```
24  function PlowRotationEvent:new(object, rotationMax)
25  local self = PlowRotationEvent:emptyNew()
26  self.object = object;
27  self.rotationMax = rotationMax;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function PlowRotationEvent:readStream(streamId, connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.rotationMax = streamReadBool(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function PlowRotationEvent:writeStream(streamId, connection)
```

```
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteBool(streamId, self.rotationMax);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function PlowRotationEvent:run(connection)
54  self.object:setRotationMax(self.rotationMax, true);
55  if not connection:getIsServer() then
56  g_server:broadcastEvent(PlowRotationEvent:new(self.object,
    self.rotationMax), nil, connection, self.object);
57  end;
58  end;
```

ReceivingHopperSetCreateBoxesEvent

Description

Event for toggle box creation

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function ReceivingHopperSetCreateBoxesEvent:emptyNew()
16  local self = Event:new(ReceivingHopperSetCreateBoxesEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, boolean state)

Arguments

table object object

boolean state state

Code

```
24  function ReceivingHopperSetCreateBoxesEvent:new(object, state)
25  local self = ReceivingHopperSetCreateBoxesEvent:emptyNew()
26  self.object = object;
27  self.state = state;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function ReceivingHopperSetCreateBoxesEvent:readStream(streamId,
    connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.state = streamReadBool(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function ReceivingHopperSetCreateBoxesEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteBool(streamId, self.state);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function ReceivingHopperSetCreateBoxesEvent:run(connection)
54  self.object:setCreateBoxes(self.state, true);
55  if not connection:getIsServer() then
56  g_server:broadcastEvent(ReceivingHopperSetCreateBoxesEvent:new(self.objec
    self.state), nil, connection, self.object);
57  end;
58  end;
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, boolean state, boolean noEventSend)

Arguments

table	vehicle	vehicle
boolean	state	state
boolean	noEventSend	no event send

Code

```
65  function ReceivingHopperSetCreateBoxesEvent.sendEvent(vehicle, state, noE
66  if state ~= vehicle.state then
67  if noEventSend == nil or noEventSend == false then
68  if g_server ~= nil then
69  g_server:broadcastEvent(ReceivingHopperSetCreateBoxesEvent:new(vehicle, s
    nil, vehicle);
70  else
71  g_client:getServerConnection():sendEvent(ReceivingHopperSetCreateBoxesEve
    state));
72  end;
73  end;
74  end;
75  end;
```

ReverseDrivingSetStateEvent

Description

Event for reverse driving state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function ReverseDrivingSetStateEvent:emptyNew()
16  local self = Event:new(ReverseDrivingSetStateEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table vehicle, boolean isReverseDriving)

Arguments

table vehicle vehicle

boolean isReverseDriving is reverse driving

Code

```
24  function ReverseDrivingSetStateEvent:new(vehicle,
    isReverseDriving)
25  local self = ReverseDrivingSetStateEvent:emptyNew()
26  self.vehicle = vehicle;
27  self.isReverseDriving = isReverseDriving;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function ReverseDrivingSetStateEvent:readStream(streamId,
    connection)
36  self.vehicle = NetworkUtil.readNodeObject(streamId);
37  self.isReverseDriving = streamReadBool(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function ReverseDrivingSetStateEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.vehicle);
47  streamWriteBool(streamId, self.isReverseDriving);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function ReverseDrivingSetStateEvent:run(connection)
54  if not connection:getIsServer() then
55  g_server:broadcastEvent(self, false, connection, self.vehicle);
56  end;
57  if self.vehicle ~= nil then
58  self.vehicle:setIsReverseDriving(self.isReverseDriving, true);
59  end;
60  end;
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, boolean isReverseDriving, boolean noEventSend)

Arguments

table	vehicle	vehicle
boolean	isReverseDriving	is reverse driving
boolean	noEventSend	no event send

Code

```
67  function ReverseDrivingSetStateEvent.sendEvent(vehicle, isReverseDriving,
    noEventSend)
68  if isReverseDriving ~= vehicle.isReverseDriving then
69  if noEventSend == nil or noEventSend == false then
70  if g_server ~= nil then
71  g_server:broadcastEvent(ReverseDrivingSetStateEvent:new(vehicle,
    isReverseDriving), nil, nil, vehicle);
72  else
```

```
73  g_client:getServerConnection():sendEvent(ReverseDrivingSetStateEvent:new(
    isReverseDriving));
74  end;
75  end;
76  end;
77  end;
```

RidgeMarkerSetStateEvent

Description

Event for ridge marker state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function RidgeMarkerSetStateEvent:emptyNew()
16  local self = Event:new(RidgeMarkerSetStateEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table vehicle, boolean state)

Arguments

table vehicle vehicle

boolean state state

Code

```
24  function RidgeMarkerSetStateEvent:new(vehicle, state)
25  local self = RidgeMarkerSetStateEvent:emptyNew()
26  self.vehicle = vehicle
27  self.state = state
28  assert(state >= 0 and state < RidgeMarker.MAX_NUM_RIDGEMARKERS)
29  return self
30  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
36  function RidgeMarkerSetStateEvent:readStream(streamId, connection)
37  self.vehicle = NetworkUtil.readNodeObject(streamId)
38  self.state = streamReadUIntN(streamId, RidgeMarker.SEND_NUM_BITS)
39  self:run(connection)
40  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
46  function RidgeMarkerSetStateEvent:writeStream(streamId,
    connection)
47  NetworkUtil.writeNodeObject(streamId, self.vehicle)
48  streamWriteUIntN(streamId, self.state, RidgeMarker.SEND_NUM_BITS)
49  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
54  function RidgeMarkerSetStateEvent:run(connection)
55  self.vehicle:setRidgeMarkerState(self.state, true)
56  if not connection:getIsServer() then
57  g_server:broadcastEvent(RidgeMarkerSetStateEvent:new(self.vehicle,
    self.state), nil, connection, self.vehicle)
58  end
59  end
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, integer state, boolean noEventSend)

Arguments

table	vehicle	vehicle
integer	state	discharge state
boolean	noEventSend	no event send

Code

```
66  function RidgeMarkerSetStateEvent.sendEvent(vehicle, state, noEventSend)
67  if noEventSend == nil or noEventSend == false then
68  if g_server ~= nil then
69  g_server:broadcastEvent(RidgeMarkerSetStateEvent:new(vehicle, state), nil
    nil, self)
70  else
71  g_client:getServerConnection():sendEvent(RidgeMarkerSetStateEvent:new(veh
    state))
72  end
73  end
74  end
```

SetCoverStateEvent

Description

Event for cover state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function SetCoverStateEvent:emptyNew()
16  return Event:new(SetCoverStateEvent_mt)
17  end
```

new

Description

Create new instance of event

Definition

new(table vehicle, integer state)

Arguments

table	vehicle	vehicle
integer	state	cover state

Code

```
23  function SetCoverStateEvent:new(vehicle, state)
24  local self = SetCoverStateEvent:emptyNew()
25
```

```
26  self.vehicle = vehicle
27  self.state = state
28
29  return self
30  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
36  function SetCoverStateEvent:readStream(streamId, connection)
37  self.vehicle = NetworkUtil.readNodeObject(streamId)
38  self.state = streamReadUIntN(streamId, Cover.SEND_NUM_BITS)
39  self:run(connection)
40  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
46  function SetCoverStateEvent:writeStream(streamId, connection)
47  NetworkUtil.writeNodeObject(streamId, self.vehicle)
48  streamWriteUIntN(streamId, self.state, Cover.SEND_NUM_BITS)
49  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
54  function SetCoverStateEvent:run(connection)
55  if not connection:getIsServer() then
```

```
56  g_server:broadcastEvent(self, false, connection, self.vehicle)
57  end
58  if self.vehicle ~= nil then
59  self.vehicle:setCoverState(self.state, true)
60  end
61  end
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, integer state, boolean noEventSend)

Arguments

table	vehicle	vehicle
integer	state	cover state
boolean	noEventSend	no event send

Code

```
68  function SetCoverStateEvent.sendEvent(vehicle, state, noEventSend)
69  if vehicle.spec_cover.state ~= state then
70  if noEventSend == nil or noEventSend == false then
71  if g_server ~= nil then
72  g_server:broadcastEvent(SetCoverStateEvent:new(vehicle, state), nil,
    nil, vehicle)
73  else
74  g_client:getServerConnection():sendEvent(SetCoverStateEvent:new(vehicle,
    state))
75  end
76  end
77  end
78  end
```

SetCrabSteeringEvent

Description

Event for steering mode

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function SetCrabSteeringEvent:emptyNew()
```

```
16  local self = Event:new(SetCrabSteeringEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, integer state)

Arguments

table object object

integer state state

Code

```
24  function SetCrabSteeringEvent:new(vehicle, state)
25  local self = SetCrabSteeringEvent:emptyNew()
26  self.vehicle = vehicle;
27  self.state = state;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function SetCrabSteeringEvent:readStream(streamId, connection)
36  self.vehicle = NetworkUtil.readNodeObject(streamId);
37  self.state = streamReadUIntN(streamId,
    CrabSteering.STEERING_SEND_NUM_BITS);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function SetCrabSteeringEvent:writeStream(streamId, connection)
46  NetworkUtil.writeNodeObject(streamId, self.vehicle);
47  streamWriteUIntN(streamId, self.state,
    CrabSteering.STEERING_SEND_NUM_BITS);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function SetCrabSteeringEvent:run(connection)
54  self.vehicle:setCrabSteering(self.state, true);
55  if not connection:getIsServer() then
56  g_server:broadcastEvent(SetCrabSteeringEvent:new(self.vehicle,
    self.state), nil, connection, self.object);
57  end;
58  end;
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, integer state, boolean noEventSend)

Arguments

table	vehicle	vehicle
integer	state	state
boolean	noEventSend	no event send

Code

```
65  function SetCrabSteeringEvent.sendEvent(vehicle, state, noEventSend)
66  if noEventSend == nil or noEventSend == false then
67  if g_server ~= nil then
68  g_server:broadcastEvent(SetCrabSteeringEvent:new(vehicle, state), nil,
    nil, vehicle);
69  else
70  g_client:getServerConnection():sendEvent(SetCrabSteeringEvent:new(vehicle
    state));
71  end;
72  end;
73  end;
```

SetCruiseControlSpeedEvent

Description

Event for cruise control speed

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function SetCruiseControlSpeedEvent:emptyNew()
16  local self = Event:new(SetCruiseControlSpeedEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table vehicle, float speed)

Arguments

table vehicle vehicle

float speed speed

Code

```
24  function SetCruiseControlSpeedEvent:new(vehicle, speed)
25  local self = SetCruiseControlSpeedEvent:emptyNew()
26  self.speed = speed;
27  self.vehicle = vehicle;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function SetCruiseControlSpeedEvent:readStream(streamId,
    connection)
36  self.vehicle = NetworkUtil.readNodeObject(streamId);
37  self.speed = streamReadUInt8(streamId);
```

```
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function SetCruiseControlSpeedEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.vehicle);
47  streamWriteUInt8(streamId, self.speed);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function SetCruiseControlSpeedEvent:run(connection)
54  if not connection:getIsServer() then
55  g_server:broadcastEvent(self, false, connection, self.vehicle);
56  end;
57  self.vehicle:setCruiseControlMaxSpeed(self.speed);
58  end;
```

SetCruiseControlStateEvent

Description

Event for cruise control state event

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function SetCruiseControlStateEvent:emptyNew()
```

```
16  local self = Event:new(SetCruiseControlStateEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table vehicle, integer state)

Arguments

table vehicle vehicle

integer state state

Code

```
24  function SetCruiseControlStateEvent:new(vehicle, state)
25  local self = SetCruiseControlStateEvent:emptyNew()
26  self.state = state;
27  self.vehicle = vehicle;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function SetCruiseControlStateEvent:readStream(streamId,
    connection)
36  self.vehicle = NetworkUtil.readNodeObject(streamId);
37  self.state = streamReadUIntN(streamId, 2);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function SetCruiseControlStateEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.vehicle);
47  streamWriteUIntN(streamId, self.state, 2);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function SetCruiseControlStateEvent:run(connection)
54  self.vehicle:setCruiseControlState(self.state, true);
55  end;
```

SetDischargeStateEvent

Description

Event for dicharge state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function SetDischargeStateEvent:emptyNew()
16  local self = Event:new(SetDischargeStateEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table vehicle, integer state)

Arguments

table vehicle vehicle

integer state discharge state

Code

```
24  function SetDischargeStateEvent:new(vehicle, state)
25  local self = SetDischargeStateEvent:emptyNew()
```

```
26  self.vehicle = vehicle
27  self.state = state
28  return self
29  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function SetDischargeStateEvent:readStream(streamId, connection)
36  self.vehicle = NetworkUtil.readNodeObject(streamId)
37  self.state = streamReadUIntN(streamId, 2)
38  self:run(connection)
39  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function SetDischargeStateEvent:writeStream(streamId, connection)
46  NetworkUtil.writeNodeObject(streamId, self.vehicle)
47  streamWriteUIntN(streamId, self.state, 2)
48  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function SetDischargeStateEvent:run(connection)
54  self.vehicle:setDischargeState(self.state, true)
55  if not connection:getIsServer() then
```

```
56  g_server:broadcastEvent(SetDischargeStateEvent:new(self.vehicle,
    self.state), nil, connection, self.vehicle)
57  end
58  end
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, integer state, boolean noEventSend)

Arguments

table	vehicle	vehicle
integer	state	discharge state
boolean	noEventSend	no event send

Code

```
65  function SetDischargeStateEvent.sendEvent(vehicle, state, noEventSend)
66  if noEventSend == nil or noEventSend == false then
67  if g_server ~= nil then
68  g_server:broadcastEvent(SetDischargeStateEvent:new(vehicle, state), nil,
    nil, vehicle)
69  else
70  g_client:getServerConnection():sendEvent(SetDischargeStateEvent:new(vehic
    state))
71  end
72  end
73  end
```

SetFillUnitIsFillingEvent

Description

Event for toggle filling

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function SetFillUnitIsFillingEvent:emptyNew()
16  local self = Event:new(SetFillUnitIsFillingEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table vehicle, boolean isFilling)

Arguments

table vehicle vehicle

boolean isFilling is filling state

Code

```
24  function SetFillUnitIsFillingEvent:new(vehicle, isFilling)
25  local self = SetFillUnitIsFillingEvent:emptyNew()
26  self.vehicle = vehicle
27  self.isFilling = isFilling
28  return self
29  end
```

readStream

Description

Called on client side

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function SetFillUnitIsFillingEvent:readStream(streamId,
    connection)
36  self.vehicle = NetworkUtil.readNodeObject(streamId)
37  self.isFilling = streamReadBool(streamId)
38  self:run(connection)
39  end
```

writeStream

Description

Called on server side

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function SetFillUnitIsFillingEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.vehicle)
47  streamWriteBool(streamId, self.isFilling)
48  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function SetFillUnitIsFillingEvent:run(connection)
54  self.vehicle:setFillUnitIsFilling(self.isFilling, true)
55  if not connection:getIsServer() then
56  g_server:broadcastEvent(SetFillUnitIsFillingEvent:new(self.vehicle,
    self.isFilling), nil, connection, self.vehicle)
57  end
58  end
```

SetMotorTurnedOnEvent

Description

Event for motor turned on state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function SetMotorTurnedOnEvent:emptyNew()
16  local self = Event:new(SetMotorTurnedOnEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, boolean turnedOn)

Arguments

table object object

boolean turnedOn is turned on

Code

```
24  function SetMotorTurnedOnEvent:new(object, turnedOn)
25  local self = SetMotorTurnedOnEvent:emptyNew()
26  self.object = object;
27  self.turnedOn = turnedOn;
```

```
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function SetMotorTurnedOnEvent:readStream(streamId, connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.turnedOn = streamReadBool(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function SetMotorTurnedOnEvent:writeStream(streamId, connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteBool(streamId, self.turnedOn);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function SetMotorTurnedOnEvent:run(connection)
54  if self.turnedOn then
55  self.object:startMotor(true);
56  else
57  self.object:stopMotor(true);
```

```
58  end;
59  if not connection:getIsServer() then
60  g_server:broadcastEvent(SetMotorTurnedOnEvent:new(self.object,
    self.turnedOn), nil, connection, self.object);
61  end;
62  end;
```

SetPipeStateEvent

Description

Event for pipe state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function SetPipeStateEvent:emptyNew()
16  local self = Event:new(SetPipeStateEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, integer pipeState)

Arguments

table object object

integer pipeState pipe state

Code

```
24  function SetPipeStateEvent:new(object, pipeState)
25  local self = SetPipeStateEvent:emptyNew()
26  self.object = object;
27  self.pipeState = pipeState;
28  assert(self.pipeState >= 0 and self.pipeState < 8);
29  return self;
30  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
36  function SetPipeStateEvent:readStream(streamId, connection)
37  self.object = NetworkUtil.readNodeObject(streamId);
38  self.pipeState = streamReadUIntN(streamId, 3);
39  self:run(connection);
40  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
46  function SetPipeStateEvent:writeStream(streamId, connection)
47  NetworkUtil.writeNodeObject(streamId, self.object);
48  streamWriteUIntN(streamId, self.pipeState, 3);
49  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
54  function SetPipeStateEvent:run(connection)
55  self.object:setPipeState(self.pipeState, true);
56  if not connection:getIsServer() then
57  g_server:broadcastEvent(SetPipeStateEvent:new(self.object,
    self.pipeState), nil, connection, self.object);
58  end;
59  end;
```

SetSeedIndexEvent

Description

Set seed index event

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function SetSeedIndexEvent:emptyNew()
16  local self = Event:new(SetSeedIndexEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table object, integer seedIndex)

Arguments

table object object

integer seedIndex index of seed

Code

```
24  function SetSeedIndexEvent:new(object, seedIndex)
25  local self = SetSeedIndexEvent:emptyNew()
26  self.object = object
27  self.seedIndex = seedIndex
28  return self
29  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function SetSeedIndexEvent:readStream(streamId, connection)
36  self.object = NetworkUtil.readNodeObject(streamId)
37  self.seedIndex = streamReadUInt8(streamId)
38  self:run(connection)
39  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function SetSeedIndexEvent:writeStream(streamId, connection)
46  NetworkUtil.writeNodeObject(streamId, self.object)
47  streamWriteUInt8(streamId, self.seedIndex)
48  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function SetSeedIndexEvent:run(connection)
54  if not connection:getIsServer() then
55  g_server:broadcastEvent(self, false, connection, self.object)
56  end
57  self.object:setSeedIndex(self.seedIndex, true)
58  end
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table object, integer seedIndex, boolean noEventSend)

Arguments

table	object	object
integer	seedIndex	index of seed
boolean	noEventSend	no event send

Code

```
65  function SetSeedIndexEvent.sendEvent(object, seedIndex, noEventSend)
66  if noEventSend == nil or noEventSend == false then
67  if g_server ~= nil then
68  g_server:broadcastEvent(SetSeedIndexEvent:new(object, seedIndex), nil,
    nil, object)
69  else
70  g_client:getServerConnection():sendEvent(SetSeedIndexEvent:new(object,
    seedIndex))
71  end
```

```
72  end
73  end
```

SetTurnedOnEvent

Description

Event for turned on state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function SetTurnedOnEvent:emptyNew()
16  local self = Event:new(SetTurnedOnEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, boolean isTurnedOn)

Arguments

table object object

boolean isTurnedOn is turned on state

Code

```
24  function SetTurnedOnEvent:new(object, isTurnedOn)
25  local self = SetTurnedOnEvent:emptyNew()
26  self.object = object;
27  self.isTurnedOn = isTurnedOn;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function SetTurnedOnEvent:readStream(streamId, connection)
```

```
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.isTurnedOn = streamReadBool(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer	streamId	streamId
integer	connection	connection

Code

```
45  function SetTurnedOnEvent:writeStream(streamId, connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteBool(streamId, self.isTurnedOn);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer	connection	connection

Code

```
53  function SetTurnedOnEvent:run(connection)
54  if not connection:getIsServer() then
55  g_server:broadcastEvent(self, false, connection, self.object);
56  end;
57  if self.object ~= nil then
58  self.object:setIsTurnedOn(self.isTurnedOn, true);
59  end;
60  end;
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table object, boolean isTurnedOn, boolean noEventSend)

Arguments

table	object	object
boolean	isTurnedOn	is turned on state

boolean noEventSend no event send

Code

```
67  function SetTurnedOnEvent.sendEvent(vehicle, isTurnedOn, noEventSend)
68  if noEventSend == nil or noEventSend == false then
69  if g_server ~= nil then
70  g_server:broadcastEvent(SetTurnedOnEvent:new(vehicle, isTurnedOn),
    nil, nil, vehicle);
71  else
72  g_client:getServerConnection():sendEvent(SetTurnedOnEvent:new(vehicle,
    isTurnedOn));
73  end;
74  end;
75  end;
```

SetWorkModeEvent

Description

Event for work modes

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function SetWorkModeEvent:emptyNew()
16  local self = Event:new(SetWorkModeEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table object, integer state)

Arguments

table object object

integer state state

Code

```
24  function SetWorkModeEvent:new(vehicle, state)
25  local self = SetWorkModeEvent:emptyNew()
26  self.vehicle = vehicle
27  self.state = state
28  return self
```

```
29  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function SetWorkModeEvent:readStream(streamId, connection)
36  self.vehicle = NetworkUtil.readNodeObject(streamId)
37  self.state = streamReadUIntN(streamId,
    WorkMode.WORKMODE_SEND_NUM_BITS)
38  self:run(connection)
39  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function SetWorkModeEvent:writeStream(streamId, connection)
46  NetworkUtil.writeNodeObject(streamId, self.vehicle)
47  streamWriteUIntN(streamId, self.state,
    WorkMode.WORKMODE_SEND_NUM_BITS)
48  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function SetWorkModeEvent:run(connection)
54  self.vehicle:setWorkMode(self.state, true)
55  if not connection:getIsServer() then
56  g_server:broadcastEvent(SetWorkModeEvent:new(self.vehicle,
    self.state), nil, connection, self.object)
```

```
57  end
58  end
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, integer state, boolean noEventSend)

Arguments

table	vehicle	vehicle
integer	state	state
boolean	noEventSend	no event send

Code

```
65  function SetWorkModeEvent.sendEvent(vehicle, state, noEventSend)
66  if noEventSend == nil or noEventSend == false then
67  if g_server ~= nil then
68  g_server:broadcastEvent(SetWorkModeEvent:new(vehicle, state), nil,
    nil, vehicle)
69  else
70  g_client:getServerConnection():sendEvent(SetWorkModeEvent:new(vehicle,
    state))
71  end
72  end
73  end
```

TensionBeltsEvent

Description

Event for tension belts state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function TensionBeltsEvent:emptyNew()
16  local self = Event:new(TensionBeltsEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table object, boolean isActive, integer beltId)

Arguments

table object object
boolean isActive belt is active
integer beltId id of belt

Code

```
25  function TensionBeltsEvent:new(object, isActive, beltId)
26  local self = TensionBeltsEvent:emptyNew()
27  self.object = object
28  self.isActive = isActive
29  self.beltId = beltId
30  return self
31  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId
integer connection connection

Code

```
37  function TensionBeltsEvent:readStream(streamId, connection)
38  self.object = NetworkUtil.readNodeObject(streamId);
39  if not streamReadBool(streamId) then
40  self.beltId = streamReadUIntN(streamId,
    TensionBelts.NUM_SEND_BITS)+1
41  end
42  self.isActive = streamReadBool(streamId)
43  self:run(connection)
44  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId
integer connection connection

Code

```
50  function TensionBeltsEvent:writeStream(streamId, connection)
51  NetworkUtil.writeNodeObject(streamId, self.object)
```

```
52  streamWriteBool(streamId, self.beltId == nil)
53  if self.beltId ~= nil then
54  streamWriteUIntN(streamId, self.beltId-1,
    TensionBelts.NUM_SEND_BITS)
55  end
56  streamWriteBool(streamId, self.isActive)
57  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
62  function TensionBeltsEvent:run(connection)
63  if not connection:getIsServer() then
64  g_server:broadcastEvent(self, false, connection, self.object)
65  end
66  self.object:setTensionBeltsActive(self.isActive, self.beltId,
    true)
67  end
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, boolean isActive, integer beltId, boolean noEventSend)

Arguments

table	vehicle	vehicle
boolean	isActive	belt is active
integer	beltId	id of belt
boolean	noEventSend	no event send

Code

```
75  function TensionBeltsEvent.sendEvent(vehicle, isActive, beltId,
    noEventSend)
76  if noEventSend == nil or noEventSend == false then
77  if g_server ~= nil then
78  g_server:broadcastEvent(TensionBeltsEvent:new(vehicle, isActive,
    beltId), nil, nil, vehicle)
79  else
80  g_client:getServerConnection():sendEvent(TensionBeltsEvent:new(vehicle,
    isActive, beltId))
```

```
81  end
82  end
83  end
```

TensionBeltsRefreshEvent

Description

Event for tension belts state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function TensionBeltsRefreshEvent:emptyNew()
16  local self = Event:new(TensionBeltsRefreshEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table object, boolean isActive, integer beltId)

Arguments

table	object	object
boolean	isActive	belt is active
integer	beltId	id of belt

Code

```
25  function TensionBeltsRefreshEvent:new(object)
26  local self = TensionBeltsRefreshEvent:emptyNew()
27  self.object = object
28  return self
29  end
```

readStream

Description

Called on client side

Definition

readStream(integer streamId, integer connection)

Arguments

integer	streamId	streamId
integer	connection	connection

Code

```
35  function TensionBeltsRefreshEvent:readStream(streamId, connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self:run(connection)
38  end
```

writeStream

Description

Called on server side

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
44  function TensionBeltsRefreshEvent:writeStream(streamId,
    connection)
45  NetworkUtil.writeNodeObject(streamId, self.object)
46  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
51  function TensionBeltsRefreshEvent:run(connection)
52  if not connection:getIsServer() then
53  g_server:broadcastEvent(self, false, connection, self.object)
54  end
55  self.object:refreshTensionBelts()
56  end
```

TrailerToggleTipEvent

Description

Event for toggle trailer tipping

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function TrailerToggleTipEvent:emptyNew()
16  local self = Event:new(TrailerToggleTipEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, boolean isStart, table tipTrigger, integer tipSideIndex)

Arguments

table	object	object
boolean	isStart	is start
table	tipTrigger	tip trigger
integer	tipSideIndex	index of tip side

Code

```
26  function TrailerToggleTipEvent:new(object, isStart, tipSideIndex)
27  local self = TrailerToggleTipEvent:emptyNew()
28  self.isStart = isStart;
29  self.tipSideIndex = Utils.getNoNil(tipSideIndex, 1);
30  assert(self.tipSideIndex <= 15);
31  self.object = object;
32  return self;
33  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer	streamId	streamId
integer	connection	connection

Code

```
39  function TrailerToggleTipEvent:readStream(streamId, connection)
40  self.object = NetworkUtil.readNodeObject(streamId);
41  self.isStart = streamReadBool(streamId);
42  if self.isStart then
43  self.tipSideIndex = streamReadUIntN(streamId, 4);
44  end;
45  self:run(connection);
46  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
52  function TrailerToggleTipEvent:writeStream(streamId, connection)
53  NetworkUtil.writeNodeObject(streamId, self.object);
54  streamWriteBool(streamId, self.isStart);
55  if self.isStart then
56  streamWriteUIntN(streamId, self.tipSideIndex, 4);
57  end;
58  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
63  function TrailerToggleTipEvent:run(connection)
64  if not connection:getIsServer() then
65  g_server:broadcastEvent(self, false, connection, self.object);
66  end;
67
68  if self.isStart then
69  self.object:startTipping(self.tipSideIndex, true);
70  else
71  self.object:stopTipping(true);
72  end;
73  end;
```

TreePlanterLoadPalletEvent

Description

Event for loading of pallet on tree planter

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function TreePlanterLoadPalletEvent:emptyNew()
16  local self = Event:new(TreePlanterLoadPalletEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, integer palletObjectId)

Arguments

table object object

integer palletObjectId object id of pallet

Code

```
24  function TreePlanterLoadPalletEvent:new(object, palletObjectId)
25  local self = TreePlanterLoadPalletEvent:emptyNew()
26  self.object = object;
27  self.palletObjectId = palletObjectId;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function TreePlanterLoadPalletEvent:readStream(streamId,
    connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.palletObjectId = NetworkUtil.readNodeObjectId(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function TreePlanterLoadPalletEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  NetworkUtil.writeNodeObjectId(streamId, self.palletObjectId);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function TreePlanterLoadPalletEvent:run(connection)
54  if not connection:getIsServer() then
55  g_server:broadcastEvent(self, false, connection, self.object);
56  end;
57  self.object:loadPallet(self.palletObjectId, true);
58  end;
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table object, integer palletObjectId, boolean noEventSend)

Arguments

table	object	object
integer	palletObjectId	object id of pallet
boolean	noEventSend	no event send

Code

```
65  function TreePlanterLoadPalletEvent.sendEvent(object, palletObjectId,
    noEventSend)
66  if noEventSend == nil or noEventSend == false then
67  if g_server ~= nil then
68  g_server:broadcastEvent(TreePlanterLoadPalletEvent:new(object, palletObje
    nil, nil, object);
69  else
70  g_client:getServerConnection():sendEvent(TreePlanterLoadPalletEvent:new(o
    palletObjectId));
71  end;
```

```
72  end;
73  end;
```

VehicleAttachEvent

Description

Event for attaching

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
14  function VehicleAttachEvent:emptyNew()
15  local self = Event:new(VehicleAttachEvent_mt)
16  return self
17  end
```

new

Description

Create new instance of event

Definition

new(table vehicle, table implement, integer inputJointIndex, integer jointIndex, boolean startLowered)

Arguments

table	vehicle	vehicle
table	implement	implement
integer	inputJointIndex	index of input attacher joint
integer	jointIndex	index of attacher joint
boolean	startLowered	start in lowered state

Return Values

table instance instance of event

Code

```
27  function VehicleAttachEvent:new(vehicle, implement,
    inputJointIndex, jointIndex, startLowered)
28  local self = VehicleAttachEvent:emptyNew()
29  self.jointIndex = jointIndex
30  self.inputJointIndex = inputJointIndex
31  self.vehicle = vehicle
32  self.implement = implement
33  self.startLowered = startLowered
34  assert(self.jointIndex >= 0 and self.jointIndex < 127)
35  return self
```

```
36  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
42  function VehicleAttachEvent:readStream(streamId, connection)
43  self.vehicle = NetworkUtil.readNodeObject(streamId)
44  self.implement = NetworkUtil.readNodeObject(streamId)
45  self.jointIndex = streamReadUIntN(streamId, 7)
46  self.inputJointIndex = streamReadUIntN(streamId, 7)
47  self.startLowered = streamReadBool(streamId)
48  self:run(connection)
49  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
55  function VehicleAttachEvent:writeStream(streamId, connection)
56  NetworkUtil.writeNodeObject(streamId, self.vehicle)
57  NetworkUtil.writeNodeObject(streamId, self.implement)
58  streamWriteUIntN(streamId, self.jointIndex, 7)
59  streamWriteUIntN(streamId, self.inputJointIndex, 7)
60  streamWriteBool(streamId, self.startLowered)
61  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
66  function VehicleAttachEvent:run(connection)
67  self.vehicle:attachImplement(self.implement, self.inputJointIndex,
    self.jointIndex, true, nil, self.startLowered)
68  if not connection:getIsServer() then
69  g_server:broadcastEvent(self, nil, connection, self.object)
70  end
71  end
```

VehicleBrokenEvent

Description

Event for enter request

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
14  function VehicleBrokenEvent:emptyNew()
15  local self = Event:new(VehicleBrokenEvent_mt)
16  return self
17  end
```

new

Description

Create new instance of event

Definition

new(table object, table playerStyle)

Arguments

table object object

table playerStyle info

Return Values

table instance instance of event

Code

```
24  function VehicleBrokenEvent:new(object)
25  local self = VehicleBrokenEvent:emptyNew()
26  self.object = object
27  return self
28  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
34  function VehicleBrokenEvent:readStream(streamId, connection)
35  self.object = NetworkUtil.readNodeObject(streamId)
36  self:run(connection)
37  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
43  function VehicleBrokenEvent:writeStream(streamId, connection)
44  NetworkUtil.writeNodeObject(streamId, self.object)
45  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
50  function VehicleBrokenEvent:run(connection)
51  if self.object ~= nil then
52  self.object:setBroken()
53  end
54  end
```

VehicleBundleAttachEvent

Description

Event for bundle attaching

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
14  function VehicleBundleAttachEvent:emptyNew()
15  local self = Event:new(VehicleBundleAttachEvent_mt)
16  return self
17  end
```

new

Description

Create new instance of event

Definition

new(table bundles)

Arguments

table bundles bundles

Return Values

table instance instance of event

Code

```
23  function VehicleBundleAttachEvent:new(bundles)
24  local self = VehicleBundleAttachEvent:emptyNew()
25  self.bundles = bundles
26  return self
27  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
33  function VehicleBundleAttachEvent:readStream(streamId, connection)
34  local numBundles = streamReadUInt8(streamId)
35  for i=1, numBundles do
36  local v1 = NetworkUtil.readNodeObjectId(streamId)
37  local v2 = NetworkUtil.readNodeObjectId(streamId)
38  local inputJointIndex = streamReadUIntN(streamId, 7)
39  local jointIndex = streamReadUIntN(streamId, 7)
40  table.insert(g_currentMission.vehiclesToAttach, {v1id=v1, v2id=v2,
    inputJointIndex=inputJointIndex, jointIndex=jointIndex})
41  end
42  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
48  function VehicleBundleAttachEvent:writeStream(streamId,
    connection)
49  streamWriteUInt8(streamId, #self.bundles)
50  for i=1, #self.bundles do
51  local bundle = self.bundles[i]
52  NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(bundle.v1))
53  NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(bundle.v2))
54  streamWriteUIntN(streamId, bundle.input, 7)
55  streamWriteUIntN(streamId, bundle.attacher, 7)
56  end
57  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
62  function VehicleBundleAttachEvent:run(connection)
63  self.vehicle:attachImplement(self.implement, self.inputJointIndex,
    self.jointIndex, true, nil, self.startLowered)
64  if not connection:getIsServer() then
65  g_server:broadcastEvent(self, nil, connection, self.object)
66  end
67  end
```

VehicleDetachEvent

Description

Event for detaching

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
14  function VehicleDetachEvent:emptyNew()
15  local self = Event:new(VehicleDetachEvent_mt)
16  return self
17  end
```

new

Description

Create new instance of event

Definition

new(table vehicle, table implement)

Arguments

table vehicle vehicle

table implement implement

Return Values

table instance instance of event

Code

```
24  function VehicleDetachEvent:new(vehicle, implement)
25  local self = VehicleDetachEvent:emptyNew()
26  self.implement = implement
27  self.vehicle = vehicle
28  return self
29  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function VehicleDetachEvent:readStream(streamId, connection)
36  self.vehicle = NetworkUtil.readNodeObject(streamId)
37  self.implement = NetworkUtil.readNodeObject(streamId)
38  if connection:getIsServer() then
39  self.vehicle:detachImplementByObject(self.implement, true)
40  else
41  self.vehicle:detachImplementByObject(self.implement)
```

```
42  end
43  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
49  function VehicleDetachEvent:writeStream(streamId, connection)
50  NetworkUtil.writeNodeObject(streamId, self.vehicle)
51  NetworkUtil.writeNodeObject(streamId, self.implement)
52  end
```

VehicleEnterRequestEvent

Description

Event for enter request

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
14  function VehicleEnterRequestEvent:emptyNew()
15  local self = Event:new(VehicleEnterRequestEvent_mt)
16  return self
17  end
```

new

Description

Create new instance of event

Definition

new(table object, table playerStyle)

Arguments

table object object

table playerStyle info

Return Values

table instance instance of event

Code

```
24  function VehicleEnterRequestEvent:new(object, playerStyle, farmId)
```

```
25  local self = VehicleEnterRequestEvent:emptyNew()
26  self.object = object
27  self.objectId = NetworkUtil.getObjectId(self.object)
28  self.farmId = farmId
29  self.playerStyle = playerStyle
30  return self
31  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
37  function VehicleEnterRequestEvent:readStream(streamId, connection)
38  self.objectId = NetworkUtil.readNodeObjectId(streamId)
39  self.farmId = streamReadUIntN(streamId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
40  if self.playerStyle == nil then
41  self.playerStyle = PlayerStyle:new()
42  end
43  self.playerStyle:readStream(streamId, connection)
44  self.object = NetworkUtil.getObject(self.objectId)
45  self:run(connection)
46  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
52  function VehicleEnterRequestEvent:writeStream(streamId,
    connection)
53  NetworkUtil.writeNodeObjectId(streamId, self.objectId)
54  streamWriteUIntN(streamId, self.farmId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
55  self.playerStyle:writeStream(streamId, connection)
```

```
56  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
61  function VehicleEnterRequestEvent:run(connection)
62  --if self.object.isControlled == false then
63  local enterableSpec = self.object.spec_enterable
64  if self.object ~= nil and enterableSpec ~= nil and
    enterableSpec.isControlled == false then
65  self.object:setOwner(connection)
66  self.object.controllerFarmId = self.farmId
67  g_server:broadcastEvent(VehicleEnterResponseEvent:new(self.objectId,
    false, self.playerStyle, self.farmId), true, connection,
    self.object)
68  connection:sendEvent(VehicleEnterResponseEvent:new(self.objectId,
    true, self.playerStyle, self.farmId))
69  end
70  end
```

VehicleEnterResponseEvent

Description

Event for enter response

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
14  function VehicleEnterResponseEvent:emptyNew()
15  local self = Event:new(VehicleEnterResponseEvent_mt)
16  return self
17  end
```

new

Description

Create new instance of event

Definition

new(table id, boolean isOwner, table playerStyle)

Arguments

table	id	id
boolean	isOwner	is owner
table	playerStyle	

Return Values

table instance instance of event

Code

```
25  function VehicleEnterResponseEvent:new(id, isOwner, playerStyle,
    farmId)
26  local self = VehicleEnterResponseEvent:emptyNew()
27  self.id = id
28  self.isOwner = isOwner
29  self.playerStyle = playerStyle
30  self.farmId = farmId
31  return self
32  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId	streamId
integer connection	connection

Code

```
38  function VehicleEnterResponseEvent:readStream(streamId,
    connection)
39  self.id = NetworkUtil.readNodeObjectId(streamId)
40  self.isOwner = streamReadBool(streamId)
41  if self.playerStyle == nil then
42  self.playerStyle = PlayerStyle:new()
43  end
44  self.playerStyle:readStream(streamId, connection)
45  self.farmId = streamReadUIntN(streamId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
46  self:run(connection)
47  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
53  function VehicleEnterResponseEvent:writeStream(streamId,
    connection)
54  NetworkUtil.writeNodeObjectId(streamId, self.id)
55  streamWriteBool(streamId, self.isOwner)
56  self.playerStyle:writeStream(streamId, connection)
57  streamWriteUIntN(streamId, self.farmId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
58  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
63  function VehicleEnterResponseEvent:run(connection)
64  local object = NetworkUtil.getObject(self.id)
65  if self.isOwner then
66  g_currentMission:onEnterVehicle(object, self.playerStyle,
    self.farmId)
67  else
68  local enterableSpec = object.spec_enterable
69  if enterableSpec ~= nil then
70  if not enterableSpec.isEntered then
71  object:enterVehicle(false, self.playerStyle, self.farmId)
72  end
73  end
74  end
75  end
```

VehicleLeaveEvent

Description

Event for leaving

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
14  function VehicleLeaveEvent:emptyNew()
15  local self = Event:new(VehicleLeaveEvent_mt)
16  return self
17  end
```

new

Description

Create new instance of event

Definition

new(table object)

Arguments

table object object

Return Values

table instance instance of event

Code

```
23  function VehicleLeaveEvent:new(object)
24  local self = VehicleLeaveEvent:emptyNew()
25  self.object = object
26  return self
27  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
33  function VehicleLeaveEvent:readStream(streamId, connection)
34  self.object = NetworkUtil.readNodeObject(streamId)
35  self:run(connection)
36  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
42  function VehicleLeaveEvent:writeStream(streamId, connection)
43  NetworkUtil.writeNodeObject(streamId, self.object)
44  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
49  function VehicleLeaveEvent:run(connection)
50  if not connection:getIsServer() then
51  if self.object.owner ~= nil then
52  self.object:setOwner(nil)
53  self.object.controllerFarmId = nil
54  end
55  g_server:broadcastEvent(VehicleLeaveEvent:new(self.object), nil,
    connection, self.object)
56  end
57  self.object:leaveVehicle()
58  end
```

VehicleLowerImplementEvent

Description

Event for lowering implement

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
14  function VehicleLowerImplementEvent:emptyNew()
15  local self = Event:new(VehicleLowerImplementEvent_mt)
16  return self
17  end
```

new

Description

Create new instance of event

Definition

new(table vehicle, integer jointIndex, boolean moveDown)

Arguments

table	vehicle	vehicle
integer	jointIndex	index of joint
boolean	moveDown	move down

Return Values

table instance instance of event

Code

```
25  function VehicleLowerImplementEvent:new(vehicle, jointIndex, moveDown)
26  local self = VehicleLowerImplementEvent:emptyNew()
27  self.jointIndex = jointIndex
28  self.vehicle = vehicle
29  self.moveDown = moveDown
30  return self
31  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId	streamId
integer connection	connection

Code

```
37  function VehicleLowerImplementEvent:readStream(streamId, connection)
38  self.vehicle = NetworkUtil.readNodeObject(streamId)
39  self.jointIndex = streamReadInt8(streamId)
40  self.moveDown = streamReadBool(streamId)
41  self:run(connection)
42  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId	streamId
integer connection	connection

Code

```
48  function VehicleLowerImplementEvent:writeStream(streamId, connection)
```

```
49  NetworkUtil.writeNodeObject(streamId, self.vehicle)
50  streamWriteInt8(streamId, self.jointIndex)
51  streamWriteBool(streamId, self.moveDown)
52  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
57  function VehicleLowerImplementEvent:run(connection)
58  self.vehicle:setJointMoveDown(self.jointIndex, self.moveDown, true)
59  if not connection:getIsServer() then
60  g_server:broadcastEvent(VehicleLowerImplementEvent:new(self.vehicle,
    self.jointIndex, self.moveDown), nil, connection, self.object)
61  end
62  end
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, integer jointIndex, boolean moveDown, boolean noEventSend)

Arguments

table	vehicle	vehicle
integer	jointIndex	index of joint
boolean	moveDown	move down
boolean	noEventSend	no event send

Code

```
70  function VehicleLowerImplementEvent.sendEvent(vehicle, jointIndex, moveDo
    noEventSend)
71  if noEventSend == nil or noEventSend == false then
72  if g_server ~= nil then
73  g_server:broadcastEvent(VehicleLowerImplementEvent:new(vehicle, jointInde
    moveDown), nil, nil, vehicle)
74  else
75  g_client:getServerConnection():sendEvent(VehicleLowerImplementEvent:new(v
    jointIndex, moveDown))
76  end
77  end
78  end
```

VehicleSetBeaconLightEvent
Description

Event for beacon light state

emptyNew
Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function VehicleSetBeaconLightEvent:emptyNew()
16  local self = Event:new(VehicleSetBeaconLightEvent_mt);
17  return self;
18  end;
```

new
Description

Create new instance of event

Definition

new(table object, boolean active)

Arguments

table object object

boolean active active

Code

```
24  function VehicleSetBeaconLightEvent:new(object, active)
25  local self = VehicleSetBeaconLightEvent:emptyNew()
26  self.active = active;
27  self.object = object;
28  return self;
29  end;
```

readStream
Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function VehicleSetBeaconLightEvent:readStream(streamId,
    connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
```

```
37  self.active = streamReadBool(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function VehicleSetBeaconLightEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteBool(streamId, self.active);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function VehicleSetBeaconLightEvent:run(connection)
54  self.object:setBeaconLightsVisibility(self.active, true, true);
55  if not connection:getIsServer() then
56  g_server:broadcastEvent(VehicleSetBeaconLightEvent:new(self.object,
    self.active), nil, connection, self.object);
57  end;
58  end;
```

VehicleSetLightEvent

Description

Event for light state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function VehicleSetLightEvent:emptyNew()
16  local self = Event:new(VehicleSetLightEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, integer lightsTypesMask)

Arguments

table	object	object
integer	lightsTypesMask	light types mask

Code

```
24  function VehicleSetLightEvent:new(object, lightsTypesMask)
25  local self = VehicleSetLightEvent:emptyNew()
26  self.lightsTypesMask = lightsTypesMask;
27  self.object = object;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer	streamId	streamId
integer	connection	connection

Code

```
35  function VehicleSetLightEvent:readStream(streamId, connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.lightsTypesMask = streamReadInt32(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer	streamId	streamId
integer	connection	connection

Code

```
45  function VehicleSetLightEvent:writeStream(streamId, connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteInt32(streamId, self.lightsTypesMask);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function VehicleSetLightEvent:run(connection)
54  self.object:setLightsTypesMask(self.lightsTypesMask, true, true);
55  if not connection:getIsServer() then
56  g_server:broadcastEvent(VehicleSetLightEvent:new(self.object,
    self.lightsTypesMask), nil, connection, self.object);
57  end;
58  end;
```

VehicleSetTurnLightEvent

Description

Event for turn light state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function VehicleSetTurnLightEvent:emptyNew()
16  local self = Event:new(VehicleSetTurnLightEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, integer state)

Arguments

table object object

integer state state

Code

```
24  function VehicleSetTurnLightEvent:new(object, state)
25  local self = VehicleSetTurnLightEvent:emptyNew()
26  self.object = object;
27  self.state = state;
28  assert(state >= 0 and state <= Lights.TURNLIGHT_HAZARD);
29  return self;
30  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
36  function VehicleSetTurnLightEvent:readStream(streamId, connection)
37  self.object = NetworkUtil.readNodeObject(streamId);
38  self.state = streamReadUIntN(streamId,
    Lights.turnLightSendNumBits);
39  self:run(connection);
40  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
46  function VehicleSetTurnLightEvent:writeStream(streamId,
    connection)
47  NetworkUtil.writeNodeObject(streamId, self.object);
48  streamWriteUIntN(streamId, self.state,
    Lights.turnLightSendNumBits);
49  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
54  function VehicleSetTurnLightEvent:run(connection)
55  self.object:setTurnLightState(self.state, true, true);
56  if not connection:getIsServer() then
57  g_server:broadcastEvent(VehicleSetTurnLightEvent:new(self.object,
    self.state), nil, connection, self.object);
58  end;
59  end;
```

WaterTrailerSetIsFillingEvent

Description

Event for water trailer filling

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function WaterTrailerSetIsFillingEvent:emptyNew()
16  local self = Event:new(WaterTrailerSetIsFillingEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table vehicle, boolean isFilling)

Arguments

table vehicle vehicle

boolean isFilling is filling

Code

```
24  function WaterTrailerSetIsFillingEvent:new(vehicle, isFilling)
25  local self = WaterTrailerSetIsFillingEvent:emptyNew()
26  self.vehicle = vehicle
27  self.isFilling = isFilling
28  return self
29  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function WaterTrailerSetIsFillingEvent:readStream(streamId,
    connection)
36  self.vehicle = NetworkUtil.readNodeObject(streamId)
37  self.isFilling = streamReadBool(streamId)
38  self:run(connection)
39  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function WaterTrailerSetIsFillingEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.vehicle)
47  streamWriteBool(streamId, self.isFilling)
48  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function WaterTrailerSetIsFillingEvent:run(connection)
54  if not connection:getIsServer() then
55  g_server:broadcastEvent(self, false, connection, self.vehicle)
56  end
57  self.vehicle:setIsWaterTrailerFilling(self.isFilling, true)
58  end
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table vehicle, boolean isFilling, boolean noEventSend)

Arguments

table	vehicle	vehicle
boolean	isFilling	is filling
boolean	noEventSend	no event send

Code

```
65  function WaterTrailerSetIsFillingEvent.sendEvent(vehicle, isFilling, noEv
66  if noEventSend == nil or noEventSend == false then
67  if g_server ~= nil then
68  g_server:broadcastEvent(WaterTrailerSetIsFillingEvent:new(vehicle, isFill
    nil, vehicle)
69  else
70  g_client:getServerConnection():sendEvent(WaterTrailerSetIsFillingEvent:ne
    isFilling))
71  end
72  end
73  end
```

WearableRepairEvent

Description

Event for repairing

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function WearableRepairEvent:emptyNew()
16  local self = Event:new(WearableRepairEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table vehicle)

Arguments

table vehicle vehicle

Code

```
23  function WearableRepairEvent:new(vehicle, atSellingPoint)
24  local self = WearableRepairEvent:emptyNew()
25  self.vehicle = vehicle
26  self.atSellingPoint = atSellingPoint
27  return self
28  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
34  function WearableRepairEvent:readStream(streamId, connection)
35  self.vehicle = NetworkUtil.readNodeObject(streamId)
36  self.atSellingPoint = streamReadBool(streamId)
37  self:run(connection)
38  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
44  function WearableRepairEvent:writeStream(streamId, connection)
45  NetworkUtil.writeNodeObject(streamId, self.vehicle)
46  streamWriteBool(streamId, self.atSellingPoint)
47  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
52  function WearableRepairEvent:run(connection)
```

```
53  if not connection:getIsServer() then
54  if self.vehicle.repairVehicle ~= nil then
55  self.vehicle:repairVehicle(self.atSellingPoint)
56
57  g_server:broadcastEvent(self) -- broadcast for UI updates
58  g_messageCenter:publish(MessageType.VEHICLE_REPAIRED,
    {self.vehicle, self.atSellingPoint})
59  end
60  else
61  g_messageCenter:publish(MessageType.VEHICLE_REPAIRED,
    {self.vehicle, self.atSellingPoint})
62  end
63  end
```

WoodHarvesterCutTreeEvent

Description

Event for cut tree

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function WoodHarvesterCutTreeEvent:emptyNew()
16  local self = Event:new(WoodHarvesterCutTreeEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, float length)

Arguments

table object object

float length length

Code

```
24  function WoodHarvesterCutTreeEvent:new(object, length)
25  local self = WoodHarvesterCutTreeEvent:emptyNew()
26  self.object = object;
27  self.length = length;
28  return self;
```

```
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function WoodHarvesterCutTreeEvent:readStream(streamId,
    connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.length = streamReadFloat32(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function WoodHarvesterCutTreeEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteFloat32(streamId, self.length);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function WoodHarvesterCutTreeEvent:run(connection)
54  if not connection:getIsServer() then
55  g_server:broadcastEvent(WoodHarvesterCutTreeEvent:new(self.object,
    self.length), nil, connection, self.object);
56  end;
```

```
57  self.object:cutTree(self.length, true);
58  end;
```

sendEvent

Description

Broadcast event from server to all clients, if called on client call function on server and broadcast it to all clients

Definition

sendEvent(table object, float length, boolean noEventSend)

Arguments

table	object	object
float	length	length
boolean	noEventSend	no event send

Code

```
65  function WoodHarvesterCutTreeEvent.sendEvent(object, length, noEventSend)
66  if noEventSend == nil or noEventSend == false then
67  if g_server ~= nil then
68  g_server:broadcastEvent(WoodHarvesterCutTreeEvent:new(object, length), ni
    nil, object);
69  else
70  g_client:getServerConnection():sendEvent(WoodHarvesterCutTreeEvent:new(ob
    length));
71  end;
72  end;
73  end;
```

WoodHarvesterOnCutTreeEvent

Description

Event for on cut tree

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function WoodHarvesterOnCutTreeEvent:emptyNew()
16  local self = Event:new(WoodHarvesterOnCutTreeEvent_mt)
17  return self
18  end
```

new

Description

Create new instance of event

Definition

new(table object, float radius)

Arguments

table object object

float radius radius

Code

```
24  function WoodHarvesterOnCutTreeEvent:new(object, radius)
25  local self = WoodHarvesterOnCutTreeEvent:emptyNew()
26  self.object = object
27  self.radius = radius
28  return self
29  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function WoodHarvesterOnCutTreeEvent:readStream(streamId,
    connection)
36  self.object = NetworkUtil.readNodeObject(streamId)
37  self.radius = streamReadFloat32(streamId)
38  self:run(connection)
39  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function WoodHarvesterOnCutTreeEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.object)
47  streamWriteFloat32(streamId, self.radius)
48  end
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function WoodHarvesterOnCutTreeEvent:run(connection)
54  if not connection:getIsServer() then
55  g_server:broadcastEvent(WoodHarvesterOnCutTreeEvent:new(self.object,
    self.radius), nil, connection, self.object)
56  end
57  SpecializationUtil.raiseEvent(self.object, "onCutTree", self.radius)
58  end
```

WoodHarvesterOnDelimbTreeEvent

Description

Event for delimb tree state

emptyNew

Description

Create instance of Event class

Definition

emptyNew()

Return Values

table self instance of class event

Code

```
15  function WoodHarvesterOnDelimbTreeEvent:emptyNew()
16  local self = Event:new(WoodHarvesterOnDelimbTreeEvent_mt);
17  return self;
18  end;
```

new

Description

Create new instance of event

Definition

new(table object, boolean state)

Arguments

table object object

boolean state state

Code

```
24  function WoodHarvesterOnDelimbTreeEvent:new(object, state)
25  local self = WoodHarvesterOnDelimbTreeEvent:emptyNew()
26  self.object = object;
27  self.state = state;
28  return self;
29  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
35  function WoodHarvesterOnDelimbTreeEvent:readStream(streamId,
    connection)
36  self.object = NetworkUtil.readNodeObject(streamId);
37  self.state = streamReadBool(streamId);
38  self:run(connection);
39  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
45  function WoodHarvesterOnDelimbTreeEvent:writeStream(streamId,
    connection)
46  NetworkUtil.writeNodeObject(streamId, self.object);
47  streamWriteBool(streamId, self.state);
48  end;
```

run

Description

Run action on receiving side

Definition

run(integer connection)

Arguments

integer connection connection

Code

```
53  function WoodHarvesterOnDelimbTreeEvent:run(connection)
54  if not connection:getIsServer() then
55  g_server:broadcastEvent(WoodHarvesterOnDelimbTreeEvent:new(self.object,
    self.state), nil, connection, self.object);
56  end;
57  self.object:onDelimbTree(self.state);
```

```
58  end;
```

AnimatedMapObject

Description

Class for animated map objects

Parent

AnimatedObject

onCreate

Description

Creating animated object

Definition

onCreate(integer id)

Arguments

integer id node id

Return Values

boolean success success

Code

```
17  function AnimatedMapObject:onCreate(id)
18  local object = AnimatedMapObject:new(g_server ~= nil, g_client ~= nil)
19  if object:load(id) then
20  g_currentMission:addOnCreateLoadedObject(object)
21  g_currentMission:addOnCreateLoadedObjectToSave(object)
22  object:register(true)
23  else
24  object:delete()
25  end
26  end
```

new

Description

Creating new instance of animated object class

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean	isServer	is server
boolean	isClient	is client
table	customMt	custom metatable

Return Values

boolean	isActive	speed rotating part is active
table	self	new instance of object

Code

```
34  function AnimatedMapObject:new(isServer, isClient, customMt)
35  local self = AnimatedObject:new(isServer, isClient, customMt or AnimatedObject_mt)
```

```
36  return self
37  end
```

load

Description

Load animated object attributes from object

Definition

load(integer nodeId)

Arguments

integer nodeId id of object to load from

Return Values

integer direction direction

boolean success success

Code

```
49  function AnimatedMapObject:load(nodeId)
50  local xmlFilename = getUserAttribute(nodeId, "xmlFilename")
51
52  if xmlFilename == nil then
53  print("Error: Missing 'xmlFilename' user attribute for
    AnimatedMapObject node '"..getName(nodeId).."'!")
54  return false
55  end
56
57  local baseDir = g_currentMission.loadingMapBaseDirectory
58  if baseDir == "" then
59  baseDir = Utils.getNoNil(self.baseDirectory, baseDir)
60  end
61  xmlFilename = Utils.getFilename(xmlFilename, baseDir)
62
63  local index = getUserAttribute(nodeId, "index")
64  if index == nil then
65  print("Error: Missing 'index' user attribute for
    AnimatedMapObject node '"..getName(nodeId).."'!")
66  return false
67  end
68
69  local xmlFile = loadXMLFile("AnimatedObject", xmlFilename)
70  if xmlFile == 0 then
71  return false
72  end
73
74  -- Find the index in the XML
75  local key
```

```
76  local i = 0
77  while true do
78  local objectKey =
    string.format("animatedObjects.animatedObject(%d)", i)
79  if not hasXMLProperty(xmlFile, objectKey) then
80  break
81  end
82
83  local configIndex = getXMLString(xmlFile, objectKey.."#index")
84  if configIndex == index then
85  key = objectKey
86  break
87  end
88  i = i + 1
89  end
90
91  if key == nil then
92  print("Error: index '"..index.."' not found in AnimatedObject xml
    '"..xmlFilename.."'!")
93  return false
94  end
95
96  local result = AnimatedMapObject:superClass().load(self, nodeId,
    xmlFilename, index)
97
98  delete(xmlFile)
99
100 return result
101 end
```

AnimatedObject

Description

Class for animated objects

onCreate

Description

Creating animated object

Definition

onCreate(integer id)

Arguments

integer id node id

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
19  function AnimatedObject:onCreate(id)
20  local object = AnimatedObject:new(g_server ~= nil, g_client ~= nil)
21  if object:load(id) then
22  g_currentMission:addOnCreateLoadedObject(object)
23  g_currentMission:addOnCreateLoadedObjectToSave(object)
24  object:register(true)
25  else
26  object:delete()
27  end
28  end
```

new

Description

Creating new instance of animated object class

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean	isServer	is server
boolean	isClient	is client
table	customMt	custom metatable

Return Values

boolean	hasPrerequisite	true if all prerequisite specializations are loaded
table	self	new instance of object

Code

```
36  function AnimatedObject:new(isServer, isClient, customMt)
37  local self = Object:new(isServer, isClient, customMt or AnimatedObject_mt)
38  self.nodeId = 0
39  self.isMoving = false
40  self.wasPressed = false
41  self.baseDirectory = nil
42  self.customEnvironment = g_currentMission.loadingMapModName
43
44  -- input controls fields:
45  self.controls = {}
46  self.controls.active = false
47  self.controls.posAction = nil
48  self.controls.negAction = nil
49  self.controls.posText = nil
50  self.controls.negText = nil
51  self.controls.posActionEventId = nil
52  self.controls.negActionEventId = nil
```

```
53
54  return self
55  end
```

load

Description

Load animated object from object with given configuration file

Definition

load(integer nodeId, xmlFilename string, index integer)

Arguments

integer	nodeId	id of object
xmlFilename	string	Path of the xml configuration
index	integer	Configuration index within the xml file

Return Values

boolean	hasPrerequisite	true if all prerequisite specializations are loaded
boolean	success	success

Code

```
63  function AnimatedObject:load(nodeId, xmlFile, key)
64  self.nodeId = nodeId
65
66  self.samples = {}
67
68  local success = true
69  self.saveId = getXMLString(xmlFile, key.."#saveId")
70  if self.saveId == nil then
71  self.saveId = "AnimatedObject_"..getName(nodeId)
72  end
73
74  local animKey = key .. ".animation"
75
76  self.animation = {}
77  self.animation.parts = {}
78  self.animation.duration = Utils.getNoNil(getXMLFloat(xmlFile,
    animKey.."#duration"), 3) * 1000
79  if self.animation.duration == 0 then
80  self.animation.duration = 1000
81  end
82  self.animation.time = 0
83  self.animation.direction = 0
84
85  self.animation.targetTime = 0
86  self.animation.lastTime = 0
```

```
87  self.animation.timeIsDirty = false
88
89  self.interpolationDuration = 50+30
90  self.interpolationAlpha = 0
91
92
93  local i = 0
94  while true do
95  local partKey = string.format("%s.part(%d)", animKey, i)
96  if not hasXMLProperty(xmlFile, partKey) then
97  break
98  end
99
100 local node = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, partKey.."#node"))
101 if node ~= nil then
102 local part = {}
103 part.node = node
104 part.animCurve = AnimCurve:new(linearInterpolatorN)
105 local hasFrames = false
106 local j = 0
107 while true do
108 local frameKey = string.format("%s.keyFrame(%d)", partKey, j)
109 if not hasXMLProperty(xmlFile, frameKey) then
110 break
111 end
112
113 local keyTime = getXMLFloat(xmlFile, frameKey.."#time")
114 local keyframe = {self:loadFrameValues(xmlFile, frameKey, node)}
115 keyframe.time = keyTime
116 part.animCurve:addKeyframe(keyframe)
117 hasFrames = true
118
119 j = j + 1
120 end
121
122 if hasFrames then
123 table.insert(self.animation.parts, part)
124 end
125 end
```

```
126 i = i + 1
127 end
128
129 local initialTime = Utils.getNoNil(getXMLFloat(xmlFile,
    animKey.."#initialTime"), 0)*1000
130 self:setAnimTime(initialTime / self.animation.duration, true)
131
132 local startTime = getXMLFloat(xmlFile,
    key..".openingHours#startTime")
133 local endTime = getXMLFloat(xmlFile,
    key..".openingHours#endTime")
134 if startTime ~= nil and endTime ~= nil then
135 local disableIfClosed = Utils.getNoNil(getXMLBool(xmlFile,
    key..".openingHours#disableIfClosed"), false)
136 local closedText = getXMLString(xmlFile,
    key..".openingHours#closedText")
137 if closedText ~= nil then
138 if g_i18n:hasText(closedText, self.customEnvironment) then
139 closedText = g_i18n:getText(closedText, self.customEnvironment)
140 end
141 end
142 self.openingHours = {startTime=startTime, endTime=endTime,
    disableIfClosed=disableIfClosed, closedText=closedText}
143 g_currentMission.environment:addHourChangeListener(self)
144 end
145
146 self.isEnabled = true
147
148
149 local triggerId = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, key..".controls#triggerNode"))
150 if triggerId ~= nil then
151 self.controls.triggerId = triggerId
152
153 addTrigger(self.controls.triggerId, "triggerCallback", self)
154 for i=0, getNumOfChildren(self.controls.triggerId)-1 do
155 addTrigger(getChildAt(self.controls.triggerId, i),
    "triggerCallback", self)
156 end
157
158 local posAction = getXMLString(xmlFile,
    key..".controls#posAction")
```

```
159 if posAction ~= nil then
160 if InputAction[posAction] then
161 self.controls.posAction = posAction
162
163 local posText = getXMLString(xmlFile, key..".controls#posText")
164 if posText ~= nil then
165 if g_i18n:hasText(posText, self.customEnvironment) then
166 posText = g_i18n:getText(posText, self.customEnvironment)
167 end
168 self.controls.posActionText = posText
169 end
170
171 local negText = getXMLString(xmlFile, key..".controls#negText")
172 if negText ~= nil then
173 if g_i18n:hasText(negText, self.customEnvironment) then
174 negText = g_i18n:getText(negText, self.customEnvironment)
175 end
176 self.controls.negActionText = negText
177 end
178
179 local negAction = getXMLString(xmlFile,
    key..".controls#negAction")
180 if negAction ~= nil then
181 if InputAction[negAction] then
182 self.controls.negAction = negAction
183 else
184 print("Warning: Negative direction action '"..negAction.."' not
    defined!")
185 end
186 end
187 else
188 print("Warning: Positive direction action '"..posAction.."' not
    defined!")
189 end
190 end
191 end
192
193 if g_client ~= nil then
194 local soundsKey = key .. ".sounds"
```

```
195 	self.sampleMoving = g_soundManager:loadSampleFromXML(xmlFile, soundsKey, "moving", self.baseDirectory, self.nodeId, 1, AudioGroup.ENVIRONMENT, nil, nil)
196 	self.samplePosEnd = g_soundManager:loadSampleFromXML(xmlFile, soundsKey, "posEnd", self.baseDirectory, self.nodeId, 1, AudioGroup.ENVIRONMENT, nil, nil)
197 	self.sampleNegEnd = g_soundManager:loadSampleFromXML(xmlFile, soundsKey, "negEnd", self.baseDirectory, self.nodeId, 1, AudioGroup.ENVIRONMENT, nil, nil)
198 	end
199 
200 	self.animatedObjectDirtyFlag = self:getNextDirtyFlag()
201 
202 	return success
203 	end
```

loadFrameValues

Description

Load frame values from xml

Definition

loadFrameValues(integer fileId, string key, integer node)

Arguments

integer	fileId	xml file id
string	key	key
integer	node	node id

Return Values

float	dirtMultiplier	current dirt multiplier
float	x	x translation
float	y	y translation
float	z	z translation
float	rx	x rotation
float	ry	y rotation
float	rz	z rotation
float	sx	x scale
float	sy	y scale
float	sz	z scale
integer	visibility	visibility

Code

```
220 	function AnimatedObject:loadFrameValues(xmlFile, key, node)
221 	local rx,ry,rz = StringUtil.getVectorFromString(getXMLString(xmlFile, key.."#rotation"))
222 	local x,y,z = StringUtil.getVectorFromString(getXMLString(xmlFile, key.."#translation"))
```

```
223  local sx,sy,sz = StringUtil.getVectorFromString(getXMLString(xmlFile, key.."#scale"))
224  local isVisible = Utils.getNoNil(getXMLBool(xmlFile, key.."#visibility"), true)
225
226  local drx,dry,drz = getRotation(node)
227  rx = Utils.getNoNilRad(rx, drx)
228  ry = Utils.getNoNilRad(ry, dry)
229  rz = Utils.getNoNilRad(rz, drz)
230  local dx,dy,dz = getTranslation(node)
231  x = Utils.getNoNil(x, dx)
232  y = Utils.getNoNil(y, dy)
233  z = Utils.getNoNil(z, dz)
234  local dsx,dsy,dsz = getScale(node)
235  sx = Utils.getNoNil(sx, dsx)
236  sy = Utils.getNoNil(sy, dsy)
237  sz = Utils.getNoNil(sz, dsz)
238
239  local visibility = 1
240  if not isVisible then
241  visibility = 0
242  end
243
244  return x, y, z, rx, ry, rz, sx, sy, sz, visibility
245  end
```

delete

Description

Delete animated object

Definition

delete()

Return Values

float wearMultiplier current wear multiplier

Code

```
249  function AnimatedObject:delete()
250  if self.controls.triggerId ~= nil then
251  removeTrigger(self.controls.triggerId)
252  for i=0, getNumOfChildren(self.controls.triggerId)-1 do
253  removeTrigger(getChildAt(self.controls.triggerId, i))
254  end
255  self.controls.triggerId = nil
```

```
256 end
257
258 if self.sampleMoving ~= nil then
259 g_soundManager:deleteSample(self.sampleMoving)
260 self.sampleMoving = nil
261 end
262 if self.samplePosEnd ~= nil then
263 g_soundManager:deleteSample(self.samplePosEnd)
264 self.samplePosEnd = nil
265 end
266 if self.sampleNegEnd ~= nil then
267 g_soundManager:deleteSample(self.sampleNegEnd)
268 self.sampleNegEnd = nil
269 end
270
271 g_currentMission.environment:removeHourChangeListener(self)
272
273 AnimatedObject:superClass().delete(self)
274 end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Return Values

table objectsInTensionBeltRange object in belt range

integer numObjectsIntensionBeltRange number of objects in belt range

Code

```
280 function AnimatedObject:readStream(streamId, connection)
281 AnimatedObject:superClass().readStream(self, streamId,
    connection)
282 if connection:getIsServer() then
283 local animTime = streamReadFloat32(streamId)
284 self:setAnimTime(animTime, true)
285 end
286 end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
292 function AnimatedObject:writeStream(streamId, connection)
293 AnimatedObject:superClass().writeStream(self, streamId,
    connection)
294 if not connection:getIsServer() then
295 streamWriteFloat32(streamId, self.animation.time)
296 end
297 end
```

readUpdateStream

Description

Called on client side on update

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId stream ID

integer timestamp timestamp

table connection connection

Return Values

table objectIdsToUnmount table with object ids to unmount

integer numObjects number of objects to unmount

Code

```
304 function AnimatedObject:readUpdateStream(streamId, timestamp,
    connection)
305 AnimatedObject:superClass().readUpdateStream(self, streamId,
    timestamp, connection)
306 if connection:getIsServer() then
307 local isDirty = streamReadBool(streamId)
308 if isDirty then
309 local animTime = streamReadFloat32(streamId)
310 --self:setAnimTime(animTime)
311
312 self.animation.targetTime = animTime
313 self.animation.lastTime = self.animation.time
314
315 local deltaTime = g_client.tickDuration
316 if self.lastPhysicsNetworkTime ~= nil then
317 deltaTime = math.min(g_packetPhysicsNetworkTime -
    self.lastPhysicsNetworkTime, 3*g_client.tickDuration)
```

```
318 end
319 self.lastPhysicsNetworkTime = g_packetPhysicsNetworkTime
320
321 -- interpTimeLeft is the time we would've continued
    interpolating. This should always be about g_clientInterpDelay.
322 -- If we extrapolated, reset to the full delay
323 local interpTimeLeft = g_clientInterpDelay
324 if self.animation.timeIsDirty and self.interpolationAlpha < 1
    then
325 interpTimeLeft = (1-self.interpolationAlpha) *
    self.interpolationDuration
326 interpTimeLeft = interpTimeLeft * 0.95 + g_clientInterpDelay *
    0.05 -- slighly blend towards the expected delay
327 interpTimeLeft = math.min(interpTimeLeft, 3*g_clientInterpDelay)
    -- clamp to some reasonable maximum
328 end
329 self.interpolationDuration = interpTimeLeft + deltaTime
330 self.interpolationAlpha = 0
331
332 self.animation.timeIsDirty = true
333 end
334 end
335 end
```

writeUpdateStream

Description

Called on server side on update

Definition

writeUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Code

```
342 function AnimatedObject:writeUpdateStream(streamId, connection,
    dirtyMask)
343 AnimatedObject:superClass().writeUpdateStream(self, streamId,
    connection, dirtyMask)
344 if not connection:getIsServer() then
345 if streamWriteBool(streamId, bitAND(dirtyMask,
    self.animatedObjectDirtyFlag) ~= 0) then
346 streamWriteFloat32(streamId, self.animation.time)
347 end
348 end
```

```
349 end
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key)

Arguments

integer	xmlFile	id of xml object
string	key	key

Return Values

boolean	inRange	player is in range
table	belt	nearest belt
boolean	success	success

Code

```
356 function AnimatedObject:loadFromXMLFile(xmlFile, key)
357 local animTime = getXMLFloat(xmlFile, key .. "#time")
358 if animTime ~= nil then
359 self.animation.direction = Utils.getNoNil(getXMLInt(xmlFile,
    key.."#direction"), 0)
360 self:setAnimTime(animTime, true)
361 end
362
363 AnimatedObject.hourChanged(self)
364
365 return true
366 end
```

registerActionEventsWhenInRange

Description

Register action events when the player is in range.

Definition

registerActionEventsWhenInRange()

removeActionEvents

Description

Remove action events.
Call as early as possible to avoid input conflicts.

Definition

removeActionEvents()

Return Values

boolean	hasPrerequisite	true if all prerequisite specializations are loaded

onAnimationInputToggle

Description

Event function for animation toggle (e.g. open/close door with one input).

Definition

onAnimationInputToggle()

Return Values

float dirtMultiplier current dirt multiplier

onAnimationInputContinuous

Description

Event function for continuous animation (e.g. keep button pressed to raise/lower something).

Definition

onAnimationInputContinuous()

Return Values

float dirtMultiplier current wear multiplier

update

Description

Called on update

Definition

update(float dt)

Arguments

float dt time since last call in ms

Return Values

boolean isActive speed rotating part is active

Code

```
483 function AnimatedObject:update(dt)
484 AnimatedObject:superClass().update(self, dt)
485
486 -- Update availability of the input actions
487 local deactivateInput = false
488 if self.playerInRange then
489 if self.isEnabled then
490 if not self.controls.active then
491 self:registerActionEventsWhenInRange()
492 end
493 self:updateActionEventTexts()
494 else
495 deactivateInput = true
496 if self.openingHours ~= nil and self.openingHours.closedText ~=
    nil then
497 g_currentMission:addExtraPrintText(self.openingHours.closedText)
498 end
499 end
500 else
501 deactivateInput = true
502 end
```

```
503
504 if deactivateInput and self.controls.active then
505 self:removeActionEvents()
506 end
507
508 local finishedAnimation = false
509
510 -- former updateTick()
511 if self.isServer then
512 if self.animation.direction ~= 0 then
513 local newAnimTime = MathUtil.clamp(self.animation.time +
    (self.animation.direction*dt)/self.animation.duration, 0, 1)
514
515 self:setAnimTime(newAnimTime)
516 if newAnimTime == 0 or newAnimTime == 1 then
517 self.animation.direction = 0
518 finishedAnimation = true
519 end
520 end
521
522 if self.animation.time ~= self.animation.timeSend then
523 self.animation.timeSend = self.animation.time
524 self:raiseDirtyFlags(self.animatedObjectDirtyFlag)
525 end
526 end
527
528 if not self.isServer and self.animation.timeIsDirty then
529 local maxIntrpAlpha = 1.2
530 local interpolationAlpha = self.interpolationAlpha +
    g_physicsDtUnclamped / self.interpolationDuration
531
532 if interpolationAlpha >= maxIntrpAlpha then
533 interpolationAlpha = maxIntrpAlpha
534 self.animation.timeIsDirty = false
535 end
536
537 self.interpolationAlpha = interpolationAlpha
538 local animTime = self.animation.lastTime + (interpolationAlpha *
    (self.animation.targetTime - self.animation.lastTime))
539 self:setAnimTime(animTime)
540 end
```

```
541
542 if self.sampleMoving ~= nil then
543 if self.isMoving and self.animation.direction ~= 0 then
544 if not self.sampleMoving.isPlaying then
545 g_soundManager:playSample(self.sampleMoving)
546 self.sampleMoving.isPlaying = true
547 end
548 else
549 if self.sampleMoving.isPlaying then
550 g_soundManager:stopSample(self.sampleMoving)
551 self.sampleMoving.isPlaying = false
552 end
553 end
554 end
555
556 if finishedAnimation and self.animation.direction == 0 then
557 if self.samplePosEnd ~= nil and self.animation.time == 1 then
558 g_soundManager:playSample(self.samplePosEnd)
559 elseif self.sampleNegEnd ~= nil and self.animation.time == 0 then
560 g_soundManager:playSample(self.sampleNegEnd)
561 end
562 end
563
564 self.isMoving = false
565
566 if self.animation.direction ~= 0 then
567 self:raiseActive()
568 end
569 end
```

updateTick

Description

Called on update tick

Definition

updateTick(float dt)

Arguments

float dt time since last call in ms

Return Values

boolean checkSpeedlimit check speed limit

Code

```
574 function AnimatedObject:updateTick(dt)
575 AnimatedObject:superClass().updateTick(self, dt)
```

```
576 end
```

setAnimTime

Description

Set animation time

Definition

setAnimTime(float t)

Arguments

float t time

Return Values

float speedLimit speed limit

Code

```
581 function AnimatedObject:setAnimTime(t, omitSound)
582 t = MathUtil.clamp(t, 0, 1)
583
584 for _, part in pairs(self.animation.parts) do
585 local v = part.animCurve:get(t)
586 self:setFrameValues(part.node, v)
587 end
588
589 self.animation.time = t
590 self.isMoving = true
591 end
```

setFrameValues

Description

Set frame values

Definition

setFrameValues(integer node, table v)

Arguments

integer node node id

table v values

Return Values

table object object of sapling pallet

Code

```
597 function AnimatedObject:setFrameValues(node, v)
598 setTranslation(node, v[1], v[2], v[3])
599 setRotation(node, v[4], v[5], v[6])
600 setScale(node, v[7], v[8], v[9])
601 setVisibility(node, v[10] == 1)
602 end
```

hourChanged

Description

Called on hour change

Definition

hourChanged()

Return Values

table self new instance

Code

```
606  function AnimatedObject:hourChanged()
607  if self.isServer then
608  local currentHour = g_currentMission.environment.currentHour
609  if self.openingHours ~= nil then
610  if currentHour >= self.openingHours.startTime and currentHour <
     self.openingHours.endTime then
611  if not self.openingHours.isOpen then
612  if self.isServer then
613  self.animation.direction = 1
614  end
615  self.openingHours.isOpen = true
616  end
617  if self.openingHours.disableIfClosed then
618  self.isEnabled = true
619  end
620  else
621  if self.openingHours.isOpen then
622  if self.isServer then
623  self.animation.direction = -1
624  end
625  self.openingHours.isOpen = false
626  end
627  if self.openingHours.disableIfClosed then
628  self.isEnabled = false
629  end
630  end
631  end
632  end
633  end
```

triggerCallback

Description

Trigger callback

Definition

triggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of object that calls callback
boolean	onEnter	called on enter
boolean	onLeave	called on leave
boolean	onStay	called on stay

Return Values

boolean	isActivateable is activateable

Code

```
642 function AnimatedObject:triggerCallback(triggerId, otherId, onEnter, onLeave, onStay)
643 if g_currentMission.missionInfo:isa(FSCareerMissionInfo) then
644 if onEnter or onLeave then
645 if g_currentMission.player ~= nil and otherId == g_currentMission.player.rootNode then
646 if onEnter then
647 self.playerInRange = g_currentMission.player
648 else
649 self.playerInRange = nil
650 end
651 self:raiseActive()
652 end
653 end
654 end
655 end
```

Bale

Description

Class for bales

new

Description

Creating bale object

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean	isServer	is server
boolean	isClient	is client
table	customMt	customMt

Return Values

boolean	hasPrerequisite	true if all prerequisite specializations are loaded
table	instance	Instance of object

Code

```
18 function Bale:new(isServer, isClient, customMt)
19 local mt = customMt;
20 if mt == nil then
```

```
21  mt = Bale_mt;
22  end;
23
24  local self = MountableObject:new(isServer, isClient, mt);
25
26  self.forcedClipDistance = 150;
27  registerObjectClassName(self, "Bale");
28
29  self.fillType = FillType.STRAW;
30  self.fillLevel = 0;
31  self.wrappingState = 0;
32  self.baleValueScale = 1;
33  self.canBeSold = true
34
35  g_currentMission:addLimitedObject(FSBaseMission.LIMITED_OBJECT_TYPE_BALE,
    self);
36
37  return self;
38  end;
```

delete

Description

Deleting bale object

Definition

delete()

Return Values

boolean isTurnedOn vehicle is turned on

Code

```
42  function Bale:delete()
43  if self.i3dFilename ~= nil then
44  g_i3DManager:releaseSharedI3DFile(self.i3dFilename, nil, true);
45  end
46  g_currentMission:removeLimitedObject(FSBaseMission.LIMITED_OBJECT_TYPE_BA
    self);
47  unregisterObjectClassName(self);
48  g_currentMission:removeItemToSave(self);
49  Bale:superClass().delete(self);
50  end;
```

readUpdateStream

Description

Called on client side on update

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId stream ID
integer timestamp timestamp
table connection connection

Return Values

boolean canBeTurnedOn vehicle can be turned on

Code

```
57  function Bale:readUpdateStream(streamId, timestamp, connection)
58  if connection:getIsServer() then
59  if self.supportsWrapping then
60  self:setWrappingState(streamReadUInt8(streamId)/255);
61  end
62  end
63  Bale:superClass().readUpdateStream(self, streamId, timestamp,
    connection);
64  end;
```

writeUpdateStream

Description

Called on server side on update

Definition

writeUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId stream ID
table connection connection
integer dirtyMask dirty mask

Return Values

boolean allow allow turn on

Code

```
71  function Bale:writeUpdateStream(streamId, connection, dirtyMask)
72  if not connection:getIsServer() then
73  if self.supportsWrapping then
74  streamWriteUInt8(streamId, MathUtil.clamp(self.wrappingState*255,
    0, 255));
75  end
76  end
77  Bale:superClass().writeUpdateStream(self, streamId, connection,
    dirtyMask);
78  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Return Values

string warningText turn on not allowed warning text

Code

```
84  function Bale:readStream(streamId, connection)
85  local i3dFilename = NetworkUtil.convertFromNetworkFilename(streamReadString(streamId));
86  if self.nodeId == 0 then
87  self:createNode(i3dFilename);
88  end;
89  self.fillLevel = streamReadFloat32(streamId);
90  Bale:superClass().readStream(self, streamId, connection);
91  g_currentMission:addItemToSave(self);
92
93  self.baleValueScale = streamReadFloat32(streamId);
94  if self.supportsWrapping then
95  self:setWrappingState(streamReadUInt8(streamId)/255);
96  setShaderParameter(self.meshNode, "wrappingState", self.wrappingState, 0, 0, 0, false);
97
98  local r = streamReadFloat32(streamId);
99  local g = streamReadFloat32(streamId);
100 local b = streamReadFloat32(streamId);
101 local a = streamReadFloat32(streamId);
102 self:setColor(r, g, b, a);
103 end;
104 end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Return Values

boolean isActive work area is active

Code

```
110 function Bale:writeStream(streamId, connection)
```

```
111  streamWriteString(streamId,
     NetworkUtil.convertToNetworkFilename(self.i3dFilename));
112  streamWriteFloat32(streamId, self.fillLevel);
113  Bale:superClass().writeStream(self, streamId, connection);
114
115  streamWriteFloat32(streamId, self.baleValueScale);
116
117  if self.supportsWrapping then
118  streamWriteUInt8(streamId, MathUtil.clamp(self.wrappingState*255,
     0, 255));
119
120  streamWriteFloat32(streamId, self.wrappingColor[1]);
121  streamWriteFloat32(streamId, self.wrappingColor[2]);
122  streamWriteFloat32(streamId, self.wrappingColor[3]);
123  streamWriteFloat32(streamId, self.wrappingColor[4]);
124  end
125  end;
```

mount

Description

Mount bale to object

Definition

mount(table object, integer node, float x, float y, float z, float rx, float ry, float rz)

Arguments

table	object	target object
integer	node	target node id
float	x	x position
float	y	z position
float	z	z position
float	rx	rx rotation
float	ry	ry rotation
float	rz	rz rotation

Return Values

boolean isReady is ready for ai work

Code

```
137  function Bale:mount(object, node, x,y,z, rx,ry,rz)
138  g_currentMission:removeItemToSave(self);
139  Bale:superClass().mount(self, object, node, x,y,z, rx,ry,rz);
140  end;
```

unmount

Description

Unmount bale

Definition

unmount()

Return Values

boolean isOperating is operating

Code

```
144  function Bale:unmount()
145  if Bale:superClass().unmount(self) then
146  g_currentMission:addItemToSave(self);
147  return true;
148  end
149  return false;
150  end;
```

setNodeId

Description

Set node id

Definition

setNodeId(integer nodeId)

Arguments

integer nodeId node Id

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
155  function Bale:setNodeId(nodeId)
156  Bale:superClass().setNodeId(self, nodeId);
157
158  local isRoundbale = Utils.getNoNil(getUserAttribute(nodeId,
     "isRoundbale"), false);
159  local defaultFillLevel = 2100*2;
160  if isRoundbale then
161  defaultFillLevel = 2000*2;
162  end
163  if getUserAttribute(nodeId, "baleValue") ~= nil then
164  print("Warning: bale 'baleValue' is not supported anymore. Use
     'baleValueScale' instead and adjust the creating vehicles.");
165  end
166  local meshIndex = Utils.getNoNil(getUserAttribute(nodeId,
     "baleMeshIndex"), "1|0");
167  self.meshNode = I3DUtil.indexToObject(nodeId, meshIndex);
168
169  self.meshNodes = {self.meshNode}
170
171  self.supportsWrapping = Utils.getNoNil(getUserAttribute(nodeId,
     "supportsWrapping"), false);
```

```
172 if self.supportsWrapping then
173 self.wrappingColor = {1,1,1,1};
174 end
175
176 self.fillLevel = defaultFillLevel
177 self.baleValueScale =
    Utils.getNoNil(tonumber(getUserAttribute(nodeId,
    "baleValueScale")), 1);
178
179 self.fillType = FillType.STRAW;
180 local fillTypeStr = getUserAttribute(nodeId, "fillType");
181 if fillTypeStr ~= nil then
182 local fillTypeIndex =
    g_fillTypeManager:getFillTypeIndexByName(fillTypeStr)
183 if fillTypeIndex ~= nil then
184 self.fillType = fillTypeIndex;
185 end
186 elseif Utils.getNoNil(getUserAttribute(nodeId, "isHaybale"),
    false) then
187 self.fillType = FillType.DRYGRASS_WINDROW;
188 end
189
190 local baleWidth = tonumber(getUserAttribute(nodeId,
    "baleWidth"));
191 local baleHeight = tonumber(getUserAttribute(nodeId,
    "baleHeight"));
192 local baleLength = tonumber(getUserAttribute(nodeId,
    "baleLength"));
193 local baleDiameter = tonumber(getUserAttribute(nodeId,
    "baleDiameter"));
194 if baleDiameter ~= nil and baleWidth ~= nil then
195 self.baleDiameter = baleDiameter;
196 self.baleWidth = baleWidth;
197 elseif baleHeight ~= nil and baleWidth ~= nil and baleLength ~=
    nil then
198 self.baleHeight = baleHeight;
199 self.baleWidth = baleWidth;
200 self.baleLength = baleLength;
201 else
202 local isRoundbale = Utils.getNoNil(getUserAttribute(nodeId,
    "isRoundbale"), false);
203 if isRoundbale then
204 self.baleDiameter = 1.8;
```

```
205   self.baleWidth = 1.2;
206   else
207   self.baleHeight = 0.8;
208   self.baleWidth = 1.2;
209   self.baleLength = 2.4;
210   end
211   end
212   end;
```

createNode

Description

Load node from i3d file

Definition

createNode(string i3dFilename)

Arguments

string i3dFilename i3d file name

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
217   function Bale:createNode(i3dFilename)
218   self.i3dFilename = i3dFilename;
219   self.customEnvironment, self.baseDirectory =
      Utils.getModNameAndBaseDirectory(i3dFilename);
220   local baleRoot = g_i3DManager:loadSharedI3DFile(i3dFilename);
221
222   local baleId = getChildAt(baleRoot, 0);
223   link(getRootNode(), baleId);
224   delete(baleRoot);
225
226   self:setNodeId(baleId);
227   end;
```

load

Description

Load bale

Definition

load(string i3dFilename, float x, float y, float z, float rx, float ry, float rz, integer fillLevel)

Arguments

string	i3dFilename	i3d file name
float	x	x world position
float	y	z world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation

float	rz	rz world rotation
integer	fillLevel	fill level

Return Values

boolean success success

Code

```
239  function Bale:load(i3dFilename, x,y,z, rx,ry,rz, fillLevel)
240  self.i3dFilename = i3dFilename;
241  self.customEnvironment, self.baseDirectory =
     Utils.getModNameAndBaseDirectory(i3dFilename);
242  self:createNode(i3dFilename);
243  setTranslation(self.nodeId, x, y, z);
244  setRotation(self.nodeId, rx, ry, rz);
245  if fillLevel ~= nil then
246  self.fillLevel = fillLevel;
247  end
248  g_currentMission:addItemToSave(self);
249  end;
```

loadFromMemory

Description

Setting node id and i3d file name

Definition

loadFromMemory(integer nodeId, string i3dFilename)

Arguments

integer	nodeId	node id
string	i3dFilename	i3d file name

Return Values

boolean isActive work area is active

Code

```
255  function Bale:loadFromMemory(nodeId, i3dFilename)
256  self.i3dFilename = i3dFilename;
257  self.customEnvironment, self.baseDirectory =
     Utils.getModNameAndBaseDirectory(i3dFilename);
258  self:setNodeId(nodeId);
259  end;
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key, boolean resetVehicles)

Arguments

integer	xmlFile	id of xml object
string	key	key
boolean	resetVehicles	reset vehicles

Return Values

boolean checkSpeedlimit check speed limit

boolean success success

Code

```
267  function Bale:loadFromXMLFile(xmlFile, key, resetVehicles)
268
269  local x,y,z = StringUtil.getVectorFromString(getXMLString(xmlFile, key.."#position"));
270  local xRot,yRot,zRot = StringUtil.getVectorFromString(getXMLString(xmlFile, key.."#rotation"));
271  if x == nil or y == nil or z == nil or xRot == nil or yRot == nil or zRot == nil then
272  return false;
273  end;
274  xRot = math.rad(xRot)
275  yRot = math.rad(yRot)
276  zRot = math.rad(zRot)
277
278  local filename = getXMLString(xmlFile, key.."#filename");
279  if filename == nil then
280  return false;
281  end;
282  filename = NetworkUtil.convertFromNetworkFilename(filename);
283  local rootNode = g_i3DManager:loadSharedI3DFile(filename);
284  if rootNode == 0 then
285  return false;
286  end;
287
288  local ret = false;
289  local node = getChildAt(rootNode, 0);
290  if node ~= nil and node ~= 0 then
291  setTranslation(node, x,y,z);
292  setRotation(node, xRot,yRot,zRot);
293  link(getRootNode(), node);
294  ret = true;
295  end;
296  delete(rootNode);
297  if not ret then
298  return false;
299  end;
```

```
300
301 self:setOwnerFarmId(Utils.getNoNil(getXMLInt(xmlFile, key ..
    "#farmId"), AccessHandler.EVERYONE), true)
302
303 self:loadFromMemory(node, filename);
304
305
306 local fillLevel = getXMLFloat(xmlFile, key.."#fillLevel");
307 if fillLevel ~= nil then
308 self.fillLevel = fillLevel;
309 end;
310
311 local attributes = {}
312 self:loadExtraAttributesFromXMLFile(attributes, xmlFile, key,
    resetVehicles)
313 self:applyExtraAttributes(attributes)
314
315 return true;
316 end;
```

getValue

Description

Get price value of bale

Definition

getValue()

Return Values

float dirtMultiplier current dirt multiplier

Code

```
357 function Bale:getValue()
358 local pricePerLiter =
    g_currentMission.economyManager:getPricePerLiter(self.fillType);
359 return self.fillLevel * pricePerLiter * self.baleValueScale;
360 end;
```

getFillType

Description

Get fill type of bale

Definition

getFillType()

Return Values

float dirtMultiplier current wear multiplier

integer fillType current fill type id

Code

```
365 function Bale:getFillType()
```

```
366  return self.fillType;
367  end;
```

getFillLevel

Description

Get fill level of bale

Definition

getFillLevel()

Return Values

boolean success success

integer fillLevel current fill level

Code

```
372  function Bale:getFillLevel()
373  return self.fillLevel;
374  end;
```

setFillLevel

Description

Set fill level of bale

Definition

setFillLevel(integer fillLevel)

Arguments

integer fillLevel fill level

Return Values

float speedLimit speed limit

Code

```
379  function Bale:setFillLevel(fillLevel)
380  self.fillLevel = fillLevel;
381  end;
```

setCanBeSold

Description

Set if bale can be sold

Definition

setCanBeSold(boolean canBeSold)

Arguments

boolean canBeSold bale can be sold

Return Values

float totalMass total mass

Code

```
386  function Bale:setCanBeSold(canBeSold)
387  self.canBeSold = canBeSold
388  end
```

getCanBeSold

Description

Returns if bale can be sold

Definition

getCanBeSold()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

boolean canBeSold bale can be sold

Code

```
393 function Bale:getCanBeSold()
394 return self.canBeSold
395 end
```

setWrappingState

Description

Set wrapping state of bale

Definition

setWrappingState(boolean wrappingState)

Arguments

boolean wrappingState new wrapping state

Return Values

boolean allow allow turn on

Code

```
400 function Bale:setWrappingState(wrappingState)
401 if self.supportsWrapping then
402 if self.wrappingState ~= wrappingState then
403 self:raiseDirtyFlags(self.physicsObjectDirtyFlag);
404 end
405 self.wrappingState = wrappingState;
406 setShaderParameter(self.meshNode, "wrappingState",
    self.wrappingState, 0, 0, 0, false);
407 end
408 end
```

setColor

Description

Set color of bale

Definition

setColor(float r, float g, float b, float a)

Arguments

float r red channel value

float g green channel value

float b blue channel value

float a alpha channel value

Return Values

float dirtMultiplier current dirt multiplier

Code

```
416 function Bale:setColor(r, g, b, a)
```

```
417  r, g, b, a = Utils.getNoNil(r, 1), Utils.getNoNil(g, 1), Utils.getNoNil(b, 1), Utils.getNoNil(a, 1);
418  self.wrappingColor = {r, g, b, a};
419
420  if getHasShaderParameter(self.meshNode, "colorScale") then
421  setShaderParameter(self.meshNode, "colorScale", r, g, b, a, false);
422  end;
423  end;
```

getMeshNodes

Description

Set wrapping state of bale

Definition

getMeshNodes()

Return Values

float dirtMultiplier current wear multiplier

table meshNodes mesh nodes

Code

```
428  function Bale:getMeshNodes()
429  return self.meshNodes
430  end
```

Basketball

Description

Class for basketballs

Parent

PhysicsObject

onCreate

Description

Creating basketball

Definition

onCreate(integer id)

Arguments

integer id node id

Return Values

boolean success success

Code

```
15  function Basketball:onCreate(id)
16  local basketball = Basketball:new(g_server ~= nil, g_client ~= nil)
17  local x, y, z = getWorldTranslation(id)
18  local rx, ry, rz = getWorldRotation(id)
19  local filename = Utils.getNoNil(getUserAttribute(id, "filename"), "$data/objects/basketball/basketball.i3d")
```

```
20  filename = Utils.getFilename(filename,
    g_currentMission.loadingMapBaseDirectory);
21
22  if basketball:load(filename, x, y, z, rx, ry, rz) then
23  g_currentMission:addOnCreateLoadedObject(basketball)
24  basketball:register(true)
25  else
26  basketball:delete()
27  end
28  end
```

new

Description

Creating basketball object

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean	isServer	is server
boolean	isClient	is client
table	customMt	customMt

Return Values

boolean	hasPrerequisite	true if all prerequisite specializations are loaded
table	instance	Instance of object

Code

```
36  function Basketball:new(isServer, isClient, customMt)
37  local mt = customMt
38  if mt == nil then
39  mt = Basketball_mt
40  end
41
42  local self = PhysicsObject:new(isServer, isClient, mt)
43
44  self.forcedClipDistance = 150
45  registerObjectClassName(self, "Basketball")
46
47  return self
48  end
```

delete

Description

Deleting basketball object

Definition

delete()

Return Values

boolean isAllowed is allowed

Code

```
52  function Basketball:delete()
53  if self.i3dFilename ~= nil then
54  g_i3DManager:releaseSharedI3DFile(self.i3dFilename, nil, true)
55  end
56  unregisterObjectClassName(self)
57  Basketball:superClass().delete(self)
58  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Return Values

string attributes attributes

Code

```
64  function Basketball:readStream(streamId, connection)
65  if connection:getIsServer() then
66  local i3dFilename = NetworkUtil.convertFromNetworkFilename(streamReadString(streamId))
67  if self.nodeId == 0 then
68  self:createNode(i3dFilename)
69  end
70  Basketball:superClass().readStream(self, streamId, connection)
71  end
72  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Return Values

boolean success success

Code

```
78  function Basketball:writeStream(streamId, connection)
```

```
79  if not connection:getIsServer() then
80  streamWriteString(streamId,
    NetworkUtil.convertToNetworkFilename(self.i3dFilename))
81  Basketball:superClass().writeStream(self, streamId, connection)
82  end
83  end
```

createNode

Description

Load node from i3d file

Definition

createNode(string i3dFilename)

Arguments

string i3dFilename i3d file name

Return Values

float sellPrice sell price

Code

```
88  function Basketball:createNode(i3dFilename)
89  self.i3dFilename = i3dFilename
90  self.customEnvironment, self.baseDirectory =
    Utils.getModNameAndBaseDirectory(i3dFilename)
91  local basketballRoot = g_i3DManager:loadSharedI3DFile(i3dFilename)
92
93  local basketballId = getChildAt(basketballRoot, 0)
94  link(getRootNode(), basketballId)
95  delete(basketballRoot)
96
97  self:setNodeId(basketballId)
98  end
```

load

Description

Load Basketball

Definition

load(string i3dFilename, float x, float y, float z, float rx, float ry, float rz)

Arguments

string i3dFilename i3d file name

float	x	x world position
float	y	z world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation

Return Values

boolean isOnField is on field

Code

```
109  function Basketball:load(i3dFilename, x,y,z, rx,ry,rz)
110  self.i3dFilename = i3dFilename
111  self.customEnvironment, self.baseDirectory =
     Utils.getModNameAndBaseDirectory(i3dFilename)
112  self:createNode(i3dFilename)
113  setTranslation(self.nodeId, x, y, z)
114  setRotation(self.nodeId, rx, ry, rz)
115
116  return true
117  end;
```

Bga

Description

Class for bga

onCreate

Description

Creating bga object

Definition

onCreate(integer id)

Arguments

integer id node id

Return Values

integer parentComponent id of parent component node

Code

```
15  function Bga:onCreate(id)
16  g_logManager:error("BGA onCreate is deprecated!")
17  end
```

new

Description

Creating bga object

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean isServer is server

boolean isClient is client

table customMt customMt

Return Values

float lastSpeed last speed

table instance Instance of object

Code

```
25  function Bga:new(isServer, isClient, customMt)
26  local self = Object:new(isServer, isClient, customMt or Bga_mt)
```

```
27
28  self.nodeId = 0
29
30  self.digestateSilo = {}
31
32  self.bunker = {}
33  self.bunker.updateTimer = 0
34  self.bunker.money = 0
35
36  self.bgaDirtyFlag = self:getNextDirtyFlag()
37
38  return self
39  end
```

load

Description

Load bga

Definition

load(integer nodeId)

Arguments

integer nodeId node id

Return Values

table owner owner

boolean success success

Code

```
45  function Bga:load(id, xmlFile, key)
46
47  self.nodeId = id
48
49  local siloKey = key..".digestateSilo"
50
51  local loadingNode = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, siloKey..".loadingStation#node"))
52  if loadingNode ~= nil then
53  local loadingStation = LoadingStation:new(self.isServer,
    self.isClient)
54  if loadingStation:load(loadingNode, xmlFile,
    siloKey..".loadingStation") then
55  self.digestateSilo.loadingStation = loadingStation
56  end
57  end
58
59  if self.digestateSilo.loadingStation == nil then
```

```
60  g_logManager:warning("Could not load loading station for
    '%s.loadingStation'!", siloKey)
61  return false
62  end
63
64  local storages = {}
65  local i = 0
66  while true do
67  local storageKey = string.format("%s.storages.storage(%d)",
    siloKey, i)
68  if not hasXMLProperty(xmlFile, storageKey) then
69  break
70  end
71
72  local storageNode = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, storageKey.."#node"))
73  if storageNode ~= nil then
74  local storage = Storage:new(self.isServer, self.isClient)
75  if storage:load(storageNode, xmlFile, storageKey) then
76  table.insert(storages, storage)
77  storage:setOwnerFarmId(self:getOwnerFarmId(), true)
78  else
79  g_logManager:warning("Could not load storage for '%s'!",
    storageKey)
80  end
81  else
82  g_logManager:warning("Missing 'node' for storage '%s'!",
    storageKey)
83  end
84
85  i = i + 1
86  end
87
88  if #storages == 0 then
89  g_logManager:warning("Missing digestate silo storages for bga!")
90  return false
91  end
92
93  self.digestateSilo.storages = storages
94
95
```

```
96  local bunkerKey = "placeable.bga.bunker"
97  local unloadingNode = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, bunkerKey..".unloadingStation#node"))
98  if unloadingNode ~= nil then
99  local unloadingStation = BgaSellStation:new(self.isServer,
    self.isClient, self)
100 if unloadingStation:load(unloadingNode, xmlFile,
    bunkerKey..".unloadingStation") then
101 self.bunker.unloadingStation = unloadingStation
102 unloadingStation:addTargetStorage(self)
103 end
104 end
105
106 if self.bunker.unloadingStation == nil then
107 g_logManager:warning("Could not load unloading station for
    '%s.unloadingStation'!", bunkerKey)
108 return false
109 end
110
111 self.bunker.slots = {}
112 self.bunker.fillTypeToSlot = {}
113 local i = 0
114 while true do
115 local slotKey = string.format("%s.slot(%d)", bunkerKey, i)
116 if not hasXMLProperty(xmlFile, slotKey) then
117 break
118 end
119
120 local slot = {}
121 slot.capacity = getXMLInt(xmlFile, slotKey.."#capacity") or 40000
122 slot.litersPerSecond = getXMLFloat(xmlFile,
    slotKey.."#litersPerSecond") or 1
123 slot.fillTypes = {}
124 slot.fillLevel = 0
125
126 slot.display = DigitalDisplay:new()
127 if not slot.display:load(self.nodeId, xmlFile,
    slotKey..".display") then
128 slot.display = nil
129 else
130 slot.display:setValue(0)
131 end
```

```
132
133
134 local found = false
135 local j = 0
136 while true do
137 local fillTypeKey = string.format("%s.fillType(%d)", slotKey, j)
138 if not hasXMLProperty(xmlFile, fillTypeKey) then
139 break
140 end
141
142 local fillTypeCategories = getXMLString(xmlFile, fillTypeKey ..
    "#fillTypeCategories")
143 local fillTypeNames = getXMLString(xmlFile, fillTypeKey ..
    "#fillTypes")
144 local fillTypes
145
146 if fillTypeCategories ~= nil and fillTypeNames == nil then
147 fillTypes =
    g_fillTypeManager:getFillTypesByCategoryNames(fillTypeCategories,
    "Warning: '"..tostring(key).. "' has invalid fillTypeCategory
    '%s'.")
148 elseif fillTypeCategories == nil and fillTypeNames ~= nil then
149 fillTypes = g_fillTypeManager:getFillTypesByNames(fillTypeNames,
    "Warning: '"..tostring(key).. "' has invalid fillType '%s'.")
150 else
151 g_logManager:warning("Missing fillTypeCategories or fillTypes
    attribute for bga slot '%s'!", slotKey)
152 end
153
154 for _,fillTypeIndex in pairs(fillTypes) do
155 if self.bunker.fillTypeToSlot[fillTypeIndex] == nil then
156 self.bunker.fillTypeToSlot[fillTypeIndex] = slot
157
158 if slot.fillTypes[fillTypeIndex] == nil then
159 found = true
160 slot.fillTypes[fillTypeIndex] = {}
161 slot.fillTypes[fillTypeIndex].fillLevel = 0
162 slot.fillTypes[fillTypeIndex].scale = getXMLFloat(xmlFile,
    fillTypeKey.."#scale") or 1
163 slot.fillTypes[fillTypeIndex].pricePerLiter =
    getXMLFloat(xmlFile, fillTypeKey.."#pricePerLiter") or 1
164 else
```

```
165     g_logManager:warning("'%s' already used for '%s'!",
        g_fillTypeManager:getFillTypeNameByIndex(fillTypeIndex),
        fillTypeKey)
166     end
167     else
168     g_logManager:warning("'%s' already used by another slot for
        '%s'!", g_fillTypeManager:getFillTypeNameByIndex(fillTypeIndex),
        slotKey)
169     end
170     end
171
172     j = j + 1
173     end
174
175     if found then
176     table.insert(self.bunker.slots, slot)
177     else
178     g_logManager:warning("No fillTypes defined for slot '%s'!",
        slotKey)
179     end
180
181     i = i + 1
182     end
183
184     self.samples = {}
185     if self.isClient then
186     self.samples.work = g_soundManager:loadSampleFromXML(xmlFile,
        "placeable.bga.sounds", "work", self.baseDirectory, self.nodeId,
        0, AudioGroup.ENVIRONMENT, nil, nil)
187     end
188
189     g_currentMission.environment:addDayChangeListener(self)
190
191     return true
192     end
```

delete

Description

Deleting bga object

Definition

delete()

Return Values

integer

Code

```
212  function Bga:delete()
213  if self.digestateSilo.loadingStation ~= nil then
214  self.digestateSilo.loadingStation:delete()
215  end
216
217  for _, storage in ipairs(self.digestateSilo.storages) do
218  g_currentMission:removeStorage(storage)
219  storage:delete()
220  end
221
222  if self.bunker.unloadingStation ~= nil then
223  g_currentMission:removeUnloadingStation(self.bunker.unloadingStation)
224  self.bunker.unloadingStation:delete()
225  end
226
227  if self.isClient then
228  for _,sample in pairs(self.samples) do
229  g_soundManager:deleteSample(sample)
230  end
231  end
232
233  g_currentMission.environment:removeDayChangeListener(self)
234  g_messageCenter:unsubscribeAll(self)
235
236  Bga:superClass().delete(self)
237  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID
table connection connection

Return Values

boolean isFromVehicle is from vehicle

Code

```
289  function Bga:readStream(streamId, connection)
290  if connection:getIsServer() then
291  local loadingStationId = NetworkUtil.readNodeObjectId(streamId)
```

```
292 self.digestateSilo.loadingStation:readStream(streamId,
    connection)
293 g_client:finishRegisterObject(self.digestateSilo.loadingStation,
    loadingStationId)
294
295 for _, storage in ipairs(self.digestateSilo.storages) do
296 local storageId = NetworkUtil.readNodeObjectId(streamId)
297 storage:readStream(streamId, connection)
298 g_client:finishRegisterObject(storage, storageId)
299 end
300
301 local unloadingStationId = NetworkUtil.readNodeObjectId(streamId)
302 self.bunker.unloadingStation:readStream(streamId, connection)
303 g_client:finishRegisterObject(self.bunker.unloadingStation,
    unloadingStationId)
304
305 for _, slot in ipairs(self.bunker.slots) do
306 slot.fillLevel = streamReadInt32(streamId)
307 if slot.display ~= nil then
308 slot.display:setValue(slot.fillLevel)
309 end
310 end
311 end
312 end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Return Values

boolean isOperating is operating

Code

```
318 function Bga:writeStream(streamId, connection)
319 if not connection:getIsServer() then
320 NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(self.digestateSilo.loadingStation))
321 self.digestateSilo.loadingStation:writeStream(streamId,
    connection)
322 g_server:registerObjectInStream(connection,
    self.digestateSilo.loadingStation)
```

```
323
324 for _, storage in ipairs(self.digestateSilo.storages) do
325 NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(storage))
326 storage:writeStream(streamId, connection)
327 g_server:registerObjectInStream(connection, storage)
328 end
329
330 NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(self.bunker.unloadingStation))
331 self.bunker.unloadingStation:writeStream(streamId, connection)
332 g_server:registerObjectInStream(connection,
    self.bunker.unloadingStation)
333
334 for _, slot in ipairs(self.bunker.slots) do
335 streamWriteInt32(streamId, slot.fillLevel)
336 end
337 end
338 end
```

readUpdateStream

Description

Called on client side on update

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId	stream ID
integer timestamp	timestamp
table	connection connection

Return Values

float totalMass total mass

Code

```
345 function Bga:readUpdateStream(streamId, timestamp, connection)
346 if connection:getIsServer() then
347 local fillLevel = 0
348 for _, slot in ipairs(self.bunker.slots) do
349 slot.fillLevel = streamReadInt32(streamId)
350 fillLevel = fillLevel + slot.fillLevel
351 if slot.display ~= nil then
352 slot.display:setValue(slot.fillLevel)
353 end
354 end
355
```

```
356  if self.isClient then
357  if fillLevel > 0 then
358  if not g_soundManager:getIsSamplePlaying(self.samples.work) then
359  g_soundManager:playSample(self.samples.work)
360  end
361  else
362  if g_soundManager:getIsSamplePlaying(self.samples.work) then
363  g_soundManager:stopSample(self.samples.work)
364  end
365  end
366  end
367  end
368  end
```

writeUpdateStream

Description

Called on server side on update

Definition

writeUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer	streamId	stream ID
table	connection	connection
integer	dirtyMask	dirty mask

Return Values

boolean	success success

Code

```
375  function Bga:writeUpdateStream(streamId, connection, dirtyMask)
376  if not connection:getIsServer() then
377  for _, slot in ipairs(self.bunker.slots) do
378  streamWriteInt32(streamId, slot.fillLevel)
379  end
380  end
381  end
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key, boolean resetVehicles)

Arguments

integer	xmlFile	id of xml object
string	key	key
boolean	resetVehicles	reset vehicles

Return Values

boolean success success

boolean success success

Code

```
389 function Bga:loadFromXMLFile(xmlFile, key)
390 self.bunker.money = getXMLFloat(xmlFile, key.."#money") or 0
391
392 if not self.digestateSilo.loadingStation:loadFromXMLFile(xmlFile,
    key..".digestateSilo.loadingStation") then
393 return false
394 end
395
396 local i = 0
397 while true do
398 local storageKey = string.format("%s.digestateSilo.storage(%d)",
    key, i)
399 if not hasXMLProperty(xmlFile, storageKey) then
400 break
401 end
402
403 local index = getXMLInt(xmlFile, storageKey.."#index")
404 if index ~= nil then
405 if self.digestateSilo.storages[index] ~= nil then
406 if not
    self.digestateSilo.storages[index]:loadFromXMLFile(xmlFile,
    storageKey) then
407 return false
408 end
409 else
410 g_logManager:warning("Could not load digestateSilo storage. Given
    'index' '%d' is not defined!", index)
411 end
412 end
413
414 i = i + 1
415 end
416
417 local i = 0
418 while true do
419 local slotKey = string.format("%s.slot(%d)", key, i)
420 if not hasXMLProperty(xmlFile, slotKey) then
421 break
422 end
```

```
423
424 local index = getXMLInt(xmlFile, slotKey.."#index")
425 if index ~= nil then
426 local slot = self.bunker.slots[index]
427 if slot ~= nil then
428 local j = 0
429 while true do
430 local fillTypeKey = string.format("%s.fillType(%d)", slotKey, j)
431 if not hasXMLProperty(xmlFile, fillTypeKey) then
432 break
433 end
434
435 local fillTypeName = getXMLString(xmlFile,
    fillTypeKey.."#fillType")
436 local fillTypeIndex =
    g_fillTypeManager:getFillTypeIndexByName(fillTypeName)
437 if fillTypeIndex ~= nil and slot.fillTypes[fillTypeIndex] ~= nil
    then
438 slot.fillTypes[fillTypeIndex].fillLevel = getXMLFloat(xmlFile,
    fillTypeKey.."#fillLevel") or 0
439 slot.fillLevel = slot.fillLevel +
    slot.fillTypes[fillTypeIndex].fillLevel
440 end
441
442 j = j + 1
443 end
444 end
445
446 if slot.fillLevel > 0 then
447 self.bunker.isFilled = true
448 self:raiseActive()
449
450 if slot.display ~= nil then
451 slot.display:setValue(slot.fillLevel)
452 end
453 end
454 end
455
456 i = i + 1
457 end
458
```

```
459  return true
460  end
```

update

Description

Update

Definition

update(float dt)

Arguments

float dt time since last call in ms

Return Values

boolean success success

Code

```
489  function Bga:update(dt)
490  if self.bunker.isFilled then
491  self:raiseActive()
492  end
493
494  if self.isClient then
495  if self.bunker.isFilled then
496  if not g_soundManager:getIsSamplePlaying(self.samples.work) then
497  g_soundManager:playSample(self.samples.work)
498  end
499  else
500  if g_soundManager:getIsSamplePlaying(self.samples.work) then
501  g_soundManager:stopSample(self.samples.work)
502  end
503  end
504  end
505  end
```

updateTick

Description

UpdateTick

Definition

updateTick(float dt)

Arguments

float dt time since last call in ms

Return Values

string Schema overlay name which was modified if necessary

Code

```
510  function Bga:updateTick(dt)
511  if self.isServer and self.bunker.isFilled then
512  self.bunker.updateTimer = self.bunker.updateTimer + dt
```

```
513
514 if self.bunker.updateTimer > 1000 then
515 self.bunker.isFilled = false
516
517 for _, slot in ipairs(self.bunker.slots) do
518 slot.fillLevel = 0
519
520 for _, data in pairs(slot.fillTypes) do
521 if data.fillLevel > 0 then
522 local deltaLiters =
    math.min(slot.litersPerSecond*g_currentMission.missionInfo.timeScale,
    data.fillLevel)
523 data.fillLevel = data.fillLevel - deltaLiters
524
525 local converted = deltaLiters * data.scale
526 if converted > 0 then
527 self:addDigestate( converted)
528 end
529
530 if deltaLiters > 0 then
531 self:raiseDirtyFlags(self.bgaDirtyFlag)
532 end
533 end
534
535 slot.fillLevel = slot.fillLevel + data.fillLevel
536 end
537
538 if slot.fillLevel > 0 then
539 self.bunker.isFilled = true
540 end
541
542 if slot.display ~= nil then
543 slot.display:setValue(slot.fillLevel)
544 end
545 end
546
547 self.bunker.updateTimer = 0
548 end
549 end
550 end
```

BunkerSilo

Description

Class for bunker silo

onCreate

Description

Creating bunker silo object

Definition

onCreate(integer id)

Arguments

integer id node id

Return Values

float limit limit

boolean doCheckSpeedLimit do check speed limit

Code

```
23  function BunkerSilo:onCreate(id)
24  g_logManager:error("BunkerSilo.onCreate is deprecated!")
25  end
```

new

Description

Creating bunker silo object

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean isServer is server

boolean isClient is client

table customMt customMt

Return Values

table instance Instance of object

Code

```
33  function BunkerSilo:new(isServer, isClient, customMt)
34
35  local self = Object:new(isServer, isClient, customMt or
    BunkerSilo_mt)
36
37  self.nodeId = 0
38  self.interactionTriggerNode = nil
39
40  self.bunkerSiloArea = {}
41  self.bunkerSiloArea.offsetFront = 0
42  self.bunkerSiloArea.offsetBack = 0
43
44  self.acceptedFillTypes = {}
45
```

```
46  self.inputFillType = FillType.CHAFF
47  self.outputFillType = FillType.SILAGE
48  self.fermentingFillType = FillType.TARP
49
50  self.isOpenedAtFront = false
51  self.isOpenedAtBack = false
52  self.distanceToCompactedFillLevel = 100
53
54  self.fermentingTime = 0
55  self.fermentingDuration = 6*60*60*1000 -- 6hours (ingame)
56  self.fermentingPercent = 0
57
58  self.fillLevel = 0
59  self.compactedFillLevel = 0
60  self.compactedPercent = 0
61  self.emptyThreshold = 100
62
63  self.playerInRange = false
64  self.vehiclesInRange = {}
65  self.numVehiclesInRange = 0
66
67  self.siloIsFullWarningTimer = 0
68  self.siloIsFullWarningDuration = 2000
69
70  self.updateTimer = 0
71
72  self.activatable = BunkerSiloActivatable:new(self)
73
74  self.state = BunkerSilo.STATE_FILL
75
76  self.bunkerSiloDirtyFlag = self:getNextDirtyFlag()
77
78  return self
79  end
```

load

Description

Load bunker silo

Definition

load(integer nodeId)

Arguments

integer nodeId node id

Return Values

float dailyUpkeep daily up keep

boolean success success

Code

```
85  function BunkerSilo:load(id, xmlFile, key)
86
87  self.nodeId = id
88
89  self.bunkerSiloArea.start = I3DUtil.indexToObject(id,
    getXMLString(xmlFile, key..".area#startNode"))
90  self.bunkerSiloArea.width = I3DUtil.indexToObject(id,
    getXMLString(xmlFile, key..".area#widthNode"))
91  self.bunkerSiloArea.height = I3DUtil.indexToObject(id,
    getXMLString(xmlFile, key..".area#heightNode"))
92
93  self.bunkerSiloArea.sx, self.bunkerSiloArea.sy, self.bunkerSiloArea.sz =
    getWorldTranslation(self.bunkerSiloArea.start)
94  self.bunkerSiloArea.wx, self.bunkerSiloArea.wy, self.bunkerSiloArea.wz =
    getWorldTranslation(self.bunkerSiloArea.width)
95  self.bunkerSiloArea.hx, self.bunkerSiloArea.hy, self.bunkerSiloArea.hz =
    getWorldTranslation(self.bunkerSiloArea.height)
96
97  self.bunkerSiloArea.dhx = self.bunkerSiloArea.hx - self.bunkerSiloArea.s
98  self.bunkerSiloArea.dhy = self.bunkerSiloArea.hy - self.bunkerSiloArea.s
99  self.bunkerSiloArea.dhz = self.bunkerSiloArea.hz - self.bunkerSiloArea.s
100 self.bunkerSiloArea.dhx_norm, self.bunkerSiloArea.dhy_norm,
    self.bunkerSiloArea.dhz_norm =
    MathUtil.vector3Normalize(self.bunkerSiloArea.dhx,
    self.bunkerSiloArea.dhy, self.bunkerSiloArea.dhz)
101
102 self.bunkerSiloArea.dwx = self.bunkerSiloArea.wx - self.bunkerSiloArea.s
103 self.bunkerSiloArea.dwy = self.bunkerSiloArea.wy - self.bunkerSiloArea.s
104 self.bunkerSiloArea.dwz = self.bunkerSiloArea.wz - self.bunkerSiloArea.s
105 self.bunkerSiloArea.dwx_norm, self.bunkerSiloArea.dwy_norm,
    self.bunkerSiloArea.dwz_norm =
    MathUtil.vector3Normalize(self.bunkerSiloArea.dwx,
    self.bunkerSiloArea.dwy, self.bunkerSiloArea.dwz)
106
107 self.interactionTriggerNode = I3DUtil.indexToObject(id,
    getXMLString(xmlFile, key..".interactionTrigger#node"))
108 if self.interactionTriggerNode ~= nil then
109 addTrigger(self.interactionTriggerNode, "interactionTriggerCallback",
    self)
```

```
110 end
111
112 self.acceptedFillTypes = {}
113 local data = StringUtil.splitString(" ", getXMLString(xmlFile,
    key.."#acceptedFillTypes") or "chaff grass_windrow dryGrass_windrow")
114 for i=1, table.getn(data) do
115 local fillTypeIndex = g_fillTypeManager:getFillTypeIndexByName(data[i])
116 if fillTypeIndex ~= nil then
117 self.acceptedFillTypes[fillTypeIndex] = true
118 else
119 g_logManager:warning("'%s' is an invalid fillType for bunkerSilo '%s'!",
    tostring(data[i]), key.."#acceptedFillTypes")
120 end
121 end
122
123 local inputFillTypeName = getXMLString(xmlFile, key.."#inputFillType") o
    "chaff"
124 local inputFillTypeIndex =
    g_fillTypeManager:getFillTypeIndexByName(inputFillTypeName)
125 if inputFillTypeIndex ~= nil then
126 self.inputFillType = inputFillTypeIndex
127 else
128 g_logManager:warning("'%s' is an invalid input fillType for bunkerSilo
    '%s'!", tostring(inputFillTypeName), key.."#inputFillType")
129 end
130
131 local outputFillTypeName = getXMLString(xmlFile, key.."#outputFillType")
    or "silage"
132 local outputFillTypeIndex =
    g_fillTypeManager:getFillTypeIndexByName(outputFillTypeName)
133 if outputFillTypeIndex ~= nil then
134 self.outputFillType = outputFillTypeIndex
135 else
136 g_logManager:warning("'%s' is an invalid output fillType for bunkerSilo
    '%s'!", tostring(outputFillTypeName), key.."#outputFillType")
137 end
138
139 if self.isServer then
140 g_densityMapHeightManager:setConvertingFillTypeAreas(self.bunkerSiloArea
    self.acceptedFillTypes, self.inputFillType)
141 end
142
```

```
143 self.distanceToCompactedFillLevel = getXMLFloat(xmlFile,
    key.."#distanceToCompactedFillLevel") or
    self.distanceToCompactedFillLevel
144 self.fermentingDuration = (getXMLFloat(xmlFile, key.."#fermentingHours")
    or 6) * 1000 * 60 * 60
145 self.openingLength = getXMLFloat(xmlFile, key.."#openingLength") or 5
146
147 self.fillLevel = 0
148
149 -- adjust timings to difficulty
150 local difficultyMultiplier = g_currentMission.missionInfo.difficulty
151 self.fermentingDuration = self.fermentingDuration*difficultyMultiplier
152 self.distanceToCompactedFillLevel =
    self.distanceToCompactedFillLevel/difficultyMultiplier
153
154 -- just for tutorial 12 (feeder)
155 self.isTutorialSilo = Utils.getNoNil(getXMLBool(xmlFile,
    key.."#isTutorialSilo"), false )
156
157 self:setState(BunkerSilo.STATE_FILL)
158
159 return true
160 end
```

delete

Description

Deleting bunker silo object

Definition

delete()

Return Values

string xml xml

Code

```
164 function BunkerSilo:delete()
165 g_currentMission:removeOnCreateLoadedObjectToSave(self)
166 if self.interactionTriggerNode ~= nil then
167 removeTrigger(self.interactionTriggerNode)
168 end
169
170 g_densityMapHeightManager:removeFixedFillTypesArea(self.bunkerSiloArea)
171 g_densityMapHeightManager:removeConvertingFillTypeAreas(self.bunkerSiloA
172
173 g_currentMission:removeActivatableObject(self.activatable)
174 BunkerSilo:superClass().delete(self)
```

```
175  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Return Values

table self instance of class event

Code

```
181  function BunkerSilo:readStream(streamId, connection)
182  BunkerSilo:superClass().readStream(self, streamId, connection)
183  if connection:getIsServer() then
184  local state = streamReadUIntN(streamId, 3)
185  self:setState(state)
186  self.isOpenedAtFront = streamReadBool(streamId)
187  self.isOpenedAtBack = streamReadBool(streamId)
188  self.fillLevel = streamReadFloat32(streamId)
189  self.compactedPercent = math.floor( (streamReadUIntN(streamId, 8)
     / 2.55) + 0.5)
190  self.fermentingPercent = math.floor( (streamReadUIntN(streamId,
     8) / 2.55) + 0.5)
191  end
192  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Return Values

table instance instance of event

Code

```
198  function BunkerSilo:writeStream(streamId, connection)
199  BunkerSilo:superClass().writeStream(self, streamId, connection)
200  if not connection:getIsServer() then
201  streamWriteUIntN(streamId, self.state, 3)
202  streamWriteBool(streamId, self.isOpenedAtFront)
```

```
203 streamWriteBool(streamId, self.isOpenedAtBack)
204 streamWriteFloat32(streamId, self.fillLevel)
205 streamWriteUIntN(streamId, 2.55*self.compactedPercent, 8)
206 streamWriteUIntN(streamId, 2.55*self.fermentingPercent, 8)
207 end
208 end
```

readUpdateStream

Description

Called on client side on update

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId stream ID
integer timestamp timestamp
table connection connection

Return Values

table self instance of class event

Code

```
215 function BunkerSilo:readUpdateStream(streamId, timestamp,
    connection)
216 BunkerSilo:superClass().readUpdateStream(self, streamId,
    timestamp, connection)
217 if connection:getIsServer() then
218 if streamReadBool(streamId) then
219 local state = streamReadUIntN(streamId, 3)
220 if state ~= self.state then
221 self:setState(state, true)
222 end
223
224 self.fillLevel = streamReadFloat32(streamId)
225 self.isOpenedAtFront = streamReadBool(streamId)
226 self.isOpenedAtBack = streamReadBool(streamId)
227
228 if self.state == BunkerSilo.STATE_FILL then
229 self.compactedPercent = math.floor( (streamReadUIntN(streamId, 8)
    / 2.55) + 0.5)
230 elseif self.state == BunkerSilo.STATE_CLOSED or self.state ==
    BunkerSilo.STATE_FERMENTED then
231 self.fermentingPercent = math.floor( (streamReadUIntN(streamId,
    8) / 2.55) + 0.5)
232 end
233 end
```

```
234 end
235 end
```

writeUpdateStream

Description

Called on server side on update

Definition

writeUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Return Values

table instance instance of event

Code

```
242 function BunkerSilo:writeUpdateStream(streamId, connection, dirtyMask)
243 BunkerSilo:superClass().writeUpdateStream(self, streamId, connection, dirtyMask)
244 if not connection:getIsServer() then
245 if streamWriteBool(streamId, bitAND(dirtyMask, self.bunkerSiloDirtyFlag) ~= 0) then
246 streamWriteUIntN(streamId, self.state, 3)
247
248 streamWriteFloat32(streamId, self.fillLevel)
249 streamWriteBool(streamId, self.isOpenedAtFront)
250 streamWriteBool(streamId, self.isOpenedAtBack)
251
252 if self.state == BunkerSilo.STATE_FILL then
253 streamWriteUIntN(streamId, 2.55*self.compactedPercent, 8)
254 elseif self.state == BunkerSilo.STATE_CLOSED or self.state == BunkerSilo.STATE_FERMENTED then
255 streamWriteUIntN(streamId, 2.55*self.fermentingPercent, 8)
256 end
257 end
258 end
259 end
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key)

Arguments

integer xmlFile id of xml object
string key key

Return Values

table self instance of class event
boolean success success

Code

```
267 function BunkerSilo:loadFromXMLFile(xmlFile, key)
268
269 local state = getXMLInt(xmlFile, key.."#state")
270 if state ~= nil then
271 if state >= 0 and state < BunkerSilo.NUM_STATES then
272 self:setState(state)
273 end
274 end
275
276 local fillLevel = getXMLFloat(xmlFile, key.."#fillLevel")
277 if fillLevel ~= nil then
278 self.fillLevel = fillLevel
279 end
280 local compactedFillLevel = getXMLFloat(xmlFile,
    key.."#compactedFillLevel")
281 if compactedFillLevel ~= nil then
282 self.compactedFillLevel = MathUtil.clamp(compactedFillLevel, 0,
    self.fillLevel)
283 end
284 self.compactedPercent =
    MathUtil.getFlooredPercent(math.min(self.compactedFillLevel,
    self.fillLevel), self.fillLevel)
285
286 local fermentingTime = getXMLFloat(xmlFile,
    key.."#fermentingTime")
287 if fermentingTime ~= nil then
288 self.fermentingTime = MathUtil.clamp(fermentingTime, 0,
    self.fermentingDuration)
289 self.fermentingPercent =
    MathUtil.getFlooredPercent(self.fermentingTime,
    self.fermentingDuration)
290 end
291
292 self.isOpenedAtFront = Utils.getNoNil(getXMLBool(xmlFile,
    key.."#openedAtFront"), false)
293 self.isOpenedAtBack = Utils.getNoNil(getXMLBool(xmlFile,
    key.."#openedAtBack"), false)
```

```
294
295 if self.isOpenedAtFront then
296 self.bunkerSiloArea.offsetFront = self:getBunkerAreaOffset(true,
    0, self.outputFillType)
297 else
298 self.bunkerSiloArea.offsetFront = self:getBunkerAreaOffset(true,
    0, self.fermentingFillType)
299 end
300 if self.isOpenedAtBack then
301 self.bunkerSiloArea.offsetBack = self:getBunkerAreaOffset(false,
    0, self.outputFillType)
302 else
303 self.bunkerSiloArea.offsetBack = self:getBunkerAreaOffset(false,
    0, self.fermentingFillType)
304 end
305
306 if self.fillLevel > 0 and self.state == BunkerSilo.STATE_DRAIN
    then
307 local area = self.bunkerSiloArea
308 local offWx = area.wx - area.sx
309 local offWz = area.wz - area.sz
310 local offW = math.sqrt(offWx*offWx + offWz*offWz)
311
312 local offHx = area.hx - area.sx
313 local offHz = area.hz - area.sz
314 local offH = math.sqrt(offHx*offHx + offHz*offHz)
315
316 if offW > 0.001 and offH > 0.001 then
317 -- offset by 0.9m in each direction (and max 45%)
318 local offWScale = math.min(0.45, 0.9 / offW)
319 offWx = offWx * offWScale
320 offWz = offWz * offWScale
321
322 local offHScale = math.min(0.45, 0.9 / offH)
323 offHx = offHx * offHScale
324 offHz = offHz * offHScale
325
326 local innerFillLevel1 =
    DensityMapHeightUtil.getFillLevelAtArea(self.fermentingFillType,
    area.sx+offWx+offHx,area.sz+offWz+offHz, area.wx-
    offWx+offHx,area.wz-offWz+offHz, area.hx+offWx-
    offHx,area.hz+offWz-offHz)
```

```
327 local innerFillLevel2 = DensityMapHeightUtil.getFillLevelAtArea(self.outputFillType, area.sx+offWx+offHx,area.sz+offWz+offHz, area.wx-offWx+offHx,area.wz-offWz+offHz, area.hx+offWx-offHx,area.hz+offWz-offHz)
328 local innerFillLevel = innerFillLevel1 + innerFillLevel2
329 if innerFillLevel < self.emptyThreshold*0.5 then
330 DensityMapHeightUtil.removeFromGroundByArea(area.sx,area.sz, area.wx,area.wz, area.hx,area.hz, self.fermentingFillType)
331 DensityMapHeightUtil.removeFromGroundByArea(area.sx,area.sz, area.wx,area.wz, area.hx,area.hz, self.outputFillType)
332 self:setState(BunkerSilo.STATE_FILL, false)
333 end
334 end
335 elseif self.state == BunkerSilo.STATE_FILL then
336 local area = self.bunkerSiloArea
337 local fermentingFillLevel, fermentingPixels, totalFermentingPixels = DensityMapHeightUtil.getFillLevelAtArea(self.fermentingFillType, area.sx,area.sz, area.wx,area.wz, area.hx,area.hz)
338 -- Set to fermented state if more than 50% of the area is filled with fermenting fill type
339 if fermentingFillLevel > self.emptyThreshold and fermentingPixels > 0.5 * totalFermentingPixels then
340 local inputFillLevel, inputPixels, totalInputPixels = DensityMapHeightUtil.getFillLevelAtArea(self.inputFillType, area.sx,area.sz, area.wx,area.wz, area.hx,area.hz)
341 -- Only change if less than 10% is filled with input type (chaff) (ie. the silo is not being filled)
342 if inputPixels < 0.1*totalInputPixels then
343 self:setState(BunkerSilo.STATE_FERMENTED, false)
344 end
345 end
346 end
347
348 return true
349 end
```

update

Description

Update

Definition

update(float dt)

Arguments

float dt time since last call in ms

Return Values

table instance instance of event

Code

```
363 function BunkerSilo:update(dt)
364
365 if self:getCanInteract(true) then
366 local fillTypeIndex = self.inputFillType
367 if self.state == BunkerSilo.STATE_CLOSED or self.state == BunkerSilo.STA
    or self.state == BunkerSilo.STATE_DRAIN then
368 fillTypeIndex = self.outputFillType
369 end
370 local fillTypeName = ""
371 local fillType = g_fillTypeManager:getFillTypeByIndex(fillTypeIndex)
372 if fillType ~= nil then
373 fillTypeName = fillType.title
374 end
375 if self.state == BunkerSilo.STATE_FILL then
376 g_currentMission:addExtraPrintText(g_i18n:getText("info_fillLevel")..str
    %s: %d", fillTypeName, self.fillLevel))
377 g_currentMission:addExtraPrintText(g_i18n:getText("info_compacting")..st
    %d%%", self.compactedPercent))
378 elseif self.state == BunkerSilo.STATE_CLOSED or self.state ==
    BunkerSilo.STATE_FERMENTED then
379 g_currentMission:addExtraPrintText(g_i18n:getText("info_fermenting")..st
    %s: %d%%", fillTypeName, self.fermentingPercent))
380 elseif self.state == BunkerSilo.STATE_DRAIN then
381 g_currentMission:addExtraPrintText(g_i18n:getText("info_fillLevel")..str
    %s: %d", fillTypeName, self.fillLevel))
382 end
383 end
384
385 if self.state == BunkerSilo.STATE_CLOSED then
386 if self.isServer then
387 self.fermentingTime = math.min(self.fermentingDuration, self.fermentingT
    dt*g_currentMission.missionInfo.timeScale)
388 local fermentingPercent = MathUtil.getFlooredPercent(self.fermentingTime
    self.fermentingDuration)
389 if fermentingPercent ~= self.fermentingPercent then
390 self.fermentingPercent = fermentingPercent
391 self:raiseDirtyFlags(self.bunkerSiloDirtyFlag)
392 end
393 if self.fermentingTime >= self.fermentingDuration then
394 self:setState(BunkerSilo.STATE_FERMENTED, true)
```

```
395 end
396 end
397 end
398
399 if self.isServer then
400 if self.state == BunkerSilo.STATE_FILL then
401 for vehicle,state in pairs(self.vehiclesInRange) do
402 if state then
403 if vehicle:getIsActive() then
404 local distance = vehicle.lastMovedDistance
405 if distance > 0 then
406 local mass = vehicle:getTotalMass(false)
407
408 local refNode = vehicle.components[1].node
409 local scale = (mass / BunkerSilo.COMPACTING_BASE_MASS)
410
411 local bunkerSiloCompacter
412 if vehicle.getBunkerSiloCompacter ~= nil then
413 bunkerSiloCompacter = vehicle:getBunkerSiloCompacter()
414 end
415 if bunkerSiloCompacter ~= nil then
416 if bunkerSiloCompacter.refNode ~= nil then
417 refNode = bunkerSiloCompacter.refNode
418 end
419 if bunkerSiloCompacter.scale ~= nil then
420 scale = (mass / BunkerSilo.COMPACTING_BASE_MASS) * bunkerSiloCompacter.s
421 end
422 end
423
424 local deltaCompact = distance*scale*self.distanceToCompactedFillLevel
425
426 local wheels = vehicle:getWheels()
427 local numWheels = #wheels
428 if numWheels > 0 then
429 local wheelsOnSilo = 0
430 local wheelsInAir = 0
431 for _, wheel in ipairs(wheels) do
432 if wheel.contact == Wheels.WHEEL_GROUND_HEIGHT_CONTACT then
433 wheelsOnSilo = wheelsOnSilo + 1
434 elseif wheel.contact == Wheels.WHEEL_NO_CONTACT then
```

```
435     wheelsInAir = wheelsInAir + 1
436     end
437     end
438     if wheelsOnSilo > 0 then
439     deltaCompact = deltaCompact * ((wheelsOnSilo + wheelsInAir) / numWheels)
440     else
441     deltaCompact = 0
442     end
443     end
444
445     if deltaCompact > 0 then
446     local compactedFillLevel = math.min(self.compactedFillLevel + deltaCompa
        self.fillLevel)
447     if compactedFillLevel ~= self.compactedFillLevel then
448     self.compactedFillLevel = compactedFillLevel
449     self.compactedPercent = MathUtil.getFlooredPercent(math.min(self.compact
        self.fillLevel), self.fillLevel)
450     self:raiseDirtyFlags(self.bunkerSiloDirtyFlag)
451     end
452     end
453     end
454     end
455     end
456     end
457     end
458     end
459
460     -- for chaff tutorial: always take the highest fill level of all bunker
461     if g_currentMission ~= nil and g_currentMission.bunkerScore ~= nil then
462     if g_currentMission.bunkerScore < self.fillLevel then
463     g_currentMission.bunkerScore = self.fillLevel
464     end
465     end
466
467     self:raiseActive()
468     end
```

updateTick

Description

UpdateTick

Definition

updateTick(float dt)

Arguments

float dt time since last call in ms

Return Values

table instance Instance of object

Code

```
473  function BunkerSilo:updateTick(dt)
474  if self.isServer then
475  self.updateTimer = self.updateTimer - dt
476  if self.updateTimer <= 0 then
477  self.updateTimer = 200 + math.random()*100 -- update every 200 to
     300ms
478
479  local oldFillLevel = self.fillLevel
480  self:updateFillLevel()
481  if oldFillLevel ~= self.fillLevel then
482  self:raiseDirtyFlags(self.bunkerSiloDirtyFlag)
483  end
484  end
485  end
486  if not self.adjustedOpeningLength then
487  self.adjustedOpeningLength = true
488  self.openingLength = math.max(self.openingLength,
     DensityMapHeightUtil.getDefaultMaxRadius(self.outputFillType)+1)
489  end
490  end
```

setState

Description

Set state

Definition

setState(boolean state, boolean showNotification)

Arguments

boolean state new state

boolean showNotification show notification

Return Values

boolean success success

Code

```
524  function BunkerSilo:setState(state, showNotification)
525
526  if state ~= self.state then
527
528  if state == BunkerSilo.STATE_FILL then
529
```

```
530 self.fermentingTime = 0
531 self.fermentingPercent = 0
532 self.compactedFillLevel = 0
533 self.isOpenedAtFront = false
534 self.isOpenedAtBack = false
535 self.bunkerSiloArea.offsetFront = 0
536 self.bunkerSiloArea.offsetBack = 0
537
538 if showNotification then
539 g_currentMission:addIngameNotification(FSBaseMission.INGAME_NOTIFICATION
    g_i18n:getText("ingameNotification_bunkerSiloIsEmpty"))
540 end
541
542 if self.isServer then
543 g_densityMapHeightManager:removeFixedFillTypesArea(self.bunkerSiloArea)
544 g_densityMapHeightManager:setConvertingFillTypeAreas(self.bunkerSiloArea
    self.acceptedFillTypes, self.inputFillType)
545 end
546
547 elseif state == BunkerSilo.STATE_CLOSED then
548
549 if self.isServer then
550 -- change fillType
551 local area = self.bunkerSiloArea
552 local offsetFront = self:getBunkerAreaOffset(true, 0, self.inputFillType
553 local offsetBack = self:getBunkerAreaOffset(false, 0, self.inputFillType
554
555 local x0 = area.sx + (offsetFront * area.dhx_norm)
556 local z0 = area.sz + (offsetFront * area.dhz_norm)
557 local x1 = x0 + area.dwx
558 local z1 = z0 + area.dwz
559 local x2 = area.sx + area.dhx - (offsetBack * area.dhx_norm)
560 local z2 = area.sz + area.dhz - (offsetBack * area.dhz_norm)
561
562 local changed = DensityMapHeightUtil.changeFillTypeAtArea(x0,z0, x1,z1,
    self.inputFillType, self.fermentingFillType)
563
564 g_densityMapHeightManager:removeFixedFillTypesArea(self.bunkerSiloArea)
565 g_densityMapHeightManager:removeConvertingFillTypeAreas(self.bunkerSiloA
566 end
567
```

```
568 if showNotification then
569 g_currentMission:addIngameNotification(FSBaseMission.INGAME_NOTIFICATION
    g_i18n:getText("ingameNotification_bunkerSiloCovered"))
570 end
571
572 elseif state == BunkerSilo.STATE_FERMENTED then
573
574 if showNotification then
575 g_currentMission:addIngameNotification(FSBaseMission.INGAME_NOTIFICATION
    g_i18n:getText("ingameNotification_bunkerSiloDoneFermenting"))
576 end
577
578 elseif state == BunkerSilo.STATE_DRAIN then
579
580 self.bunkerSiloArea.offsetFront = 0
581 self.bunkerSiloArea.offsetBack = 0
582
583 if showNotification then
584 g_currentMission:addIngameNotification(FSBaseMission.INGAME_NOTIFICATION
    g_i18n:getText("ingameNotification_bunkerSiloOpened"))
585 end
586
587 if self.isServer then
588 g_densityMapHeightManager:removeConvertingFillTypeAreas(self.bunkerSiloA
589 local fillTypes = {}
590 fillTypes[self.outputFillType] = true
591 g_densityMapHeightManager:setFixedFillTypesArea(self.bunkerSiloArea,
    fillTypes)
592 end
593
594 end
595
596 self.state = state
597 if self.isServer then
598 self:raiseDirtyFlags(self.bunkerSiloDirtyFlag)
599 end
600 end
601 end
```

openSilo

Description

Open silo

Definition

openSilo(float px, float py, float pz)

Arguments

float px x player position
float py y player position
float pz z player position

Return Values

float newX new x position
float newY new y position
float newZ new z position

Code

```
608  function BunkerSilo:openSilo(px,py,pz)
609  self:setState(BunkerSilo.STATE_DRAIN, true)
610
611  self.bunkerSiloArea.offsetFront = self:getBunkerAreaOffset(true,
     0, self.fermentingFillType)
612  self.bunkerSiloArea.offsetBack = self:getBunkerAreaOffset(false,
     0, self.fermentingFillType)
613
614  -- check which side is closer to player
615  local openAtFront = self:getIsCloserToFront(px,py,pz)
616  if openAtFront and not self.isOpenedAtFront then
617  self:switchFillTypeAtOffset(true,
     self.bunkerSiloArea.offsetFront, self.openingLength)
618  self.isOpenedAtFront = true
619  self:raiseDirtyFlags(self.bunkerSiloDirtyFlag)
620  elseif not self.isOpenedAtBack then
621  self:switchFillTypeAtOffset(false,
     self.bunkerSiloArea.offsetBack, self.openingLength)
622  self.isOpenedAtBack = true
623  self:raiseDirtyFlags(self.bunkerSiloDirtyFlag)
624  end
625  end
```

getBunkerAreaOffset

Description

Get bunker area offset

Definition

getBunkerAreaOffset(boolean updateAtFront, float offset, integer fillType)

Arguments

boolean	updateAtFront	update at front
float	offset	offset
integer	fillType	fill type

Return Values

float offset offset

Code

```
633 function BunkerSilo:getBunkerAreaOffset(updateAtFront, offset,
    fillType)
634 local area = self.bunkerSiloArea
635
636 local hx, hz = area.dhx_norm, area.dhz_norm
637 local hl = MathUtil.vector3Length(area.dhx, area.dhy, area.dhz)
638
639 while offset <= (hl - 1) do
640 local pos = offset
641 if not updateAtFront then
642 pos = hl - offset - 1
643 end
644 local d1x,d1z = pos*hx, pos*hz
645 local d2x,d2z = (pos+1)*hx, (pos+1)*hz
646
647 local a0x, a0z = area.sx + d1x, area.sz + d1z
648 local a1x, a1z = area.wx + d1x, area.wz + d1z
649 local a2x, a2z = area.sx + d2x, area.sz + d2z
650
651 local fillLevel =
    DensityMapHeightUtil.getFillLevelAtArea(fillType, a0x,a0z,
    a1x,a1z, a2x,a2z)
652 if fillLevel > 0 then
653 return offset
654 end
655 offset = offset + 1
656 end
657
658 return math.max(hl - 1, 0)
659 end
```

switchFillTypeAtOffset

Description

Switch fill type at offset

Definition

switchFillTypeAtOffset(boolean switchAtFront, float offset, float length)

Arguments

boolean switchAtFront switch at front

float	offset	offset
float	length	length

Code

```lua
function BunkerSilo:switchFillTypeAtOffset(switchAtFront, offset, length)

    local fillType = self.fermentingFillType
    local newFillType = self.outputFillType

    local a0x, a0z = nil, nil
    local a1x, a1z = nil, nil
    local a2x, a2z = nil, nil

    local area = self.bunkerSiloArea

    if switchAtFront then
        a0x, a0z = area.sx + (offset * area.dhx_norm), area.sz + (offset * area.dhz_norm)
        a1x, a1z = a0x + area.dwx, a0z + area.dwz
        a2x, a2z = area.sx + ((offset + length) * area.dhx_norm), area.sz + ((offset + length) * area.dhz_norm)
    else
        a0x, a0z = area.hx - (offset * area.dhx_norm), area.hz - (offset * area.dhz_norm)
        a1x, a1z = a0x + area.dwx, a0z + area.dwz
        a2x, a2z = area.hx - ((offset + length) * area.dhx_norm), area.hz - ((offset + length) * area.dhz_norm)
    end

    DensityMapHeightUtil.changeFillTypeAtArea(a0x,a0z, a1x,a1z, a2x,a2z, fillType, newFillType)

end
```

getIsCloserToFront

Description

Get is closer to front

Definition

getIsCloserToFront(float ix, float iy, float iz)

Arguments

float ix	x position
float iy	y position
float iz	z position

Return Values

boolean	hasCollision	has collision
float	collisionDistance	distance to collision
float	normalX	normal x

float	normalY	normal y
float	normalZ	normal z
float	normalDotDir	normal dot direction
boolean	isCloserToFront	is closer to front

Code

```
697 function BunkerSilo:getIsCloserToFront(ix,iy,iz)
698 local area = self.bunkerSiloArea
699
700 local x = area.sx + (0.5*area.dwx) + (area.offsetFront * area.dhx_norm)
701 local y = area.sy + (0.5*area.dwy) + (area.offsetFront * area.dhy_norm)
702 local z = area.sz + (0.5*area.dwz) + (area.offsetFront * area.dhz_norm)
703 local distFront = MathUtil.vector3Length(x-ix, y-iy, z-iz)
704
705 local x = area.sx + (0.5*area.dwx) + area.dhx - (area.offsetBack * area.dhx_norm)
706 local y = area.sy + (0.5*area.dwy) + area.dhy - (area.offsetBack * area.dhy_norm)
707 local z = area.sz + (0.5*area.dwz) + area.dhz - (area.offsetBack * area.dhz_norm)
708 local distBack = MathUtil.vector3Length(x-ix, y-iy, z-iz)
709
710 return distFront < distBack
711 end
```

getCanInteract

Description

Get can interact with silo

Definition

getCanInteract(boolean showInformationOnly)

Arguments

boolean showInformationOnly show information only

Return Values

boolean canInteract can interact

Code

```
717 function BunkerSilo:getCanInteract(showInformationOnly)
718 if showInformationOnly then
719 if (g_currentMission.controlPlayer and self.playerInRange) then
720 return true
721 end
722 if not g_currentMission.controlPlayer then
723 for vehicle in pairs(self.vehiclesInRange) do
```

```
724 if vehicle:getIsActiveForInput(true) then
725 return true
726 end
727 end
728 end
729 else
730 if (g_currentMission.controlPlayer and self.playerInRange) then
731 --if next(self.vehiclesInRange) == nil then
732 return true
733 --end
734 end
735 end
736 return false
737 end
```

getCanCloseSilo

Description

Get can close silo

Definition

getCanCloseSilo()

Return Values

boolean canClose can close silo

Code

```
742 function BunkerSilo:getCanCloseSilo()
743 return self.state == BunkerSilo.STATE_FILL and self.fillLevel > 0
    and self.compactedPercent >= 100
744 end
```

getCanOpenSilo

Description

Get can open silo

Definition

getCanOpenSilo()

Return Values

boolean canOpen can open silo

Code

```
749 function BunkerSilo:getCanOpenSilo()
750 if not (self.state == BunkerSilo.STATE_FERMENTED or self.state ==
    BunkerSilo.STATE_DRAIN) then
751 return false
752 end
753 local ix,iy,iz = self:getInteractionPosition()
754 if ix ~= nil then
755 local closerToFront = self:getIsCloserToFront(ix,iy,iz)
```

```
756 if closerToFront and not self.isOpenedAtFront then
757 return true
758 end
759 if not closerToFront and not self.isOpenedAtBack then
760 return true
761 end
762 end
763 return false
764 end
```

getInteractionPosition

Description

Get interact position

Definition

getInteractionPosition()

Return Values

float x x world position
float y y world position
float z z world position

Code

```
778 function BunkerSilo:getInteractionPosition()
779 if g_currentMission.controlPlayer and self.playerInRange then
780 return getWorldTranslation(g_currentMission.player.rootNode)
781 else
782 if self.vehiclesInRange[g_currentMission.currentVehicle] ~= nil then
783 return
    getWorldTranslation(self.vehiclesInRange[g_currentMission.currentVehicle
784 end
785 end
786 return nil
787 end
```

interactionTriggerCallback

Description

interactionTriggerCallback

Definition

interactionTriggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay, integer otherId)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

integer otherId id of other actor

Code

```
797 function BunkerSilo:interactionTriggerCallback(triggerId, otherId,
    onEnter, onLeave, onStay, otherShapeId)
798 if onEnter or onLeave then
799 if g_currentMission.player ~= nil and otherId ==
    g_currentMission.player.rootNode then
800 if onEnter then
801 self.playerInRange = true
802 g_currentMission:removeActivatableObject(self.activatable) -- make sure
    is not added twice
803 g_currentMission:addActivatableObject(self.activatable)
804 else
805 self.playerInRange = false
806 if self.numVehiclesInRange == 0 then
807 g_currentMission:removeActivatableObject(self.activatable)
808 end
809 end
810 else
811 --local vehicle = g_currentMission.nodeToObject[otherId]
812 local vehicle = g_currentMission.nodeToObject[otherShapeId]
813 if vehicle ~= nil then
814 if onEnter then
815 if self.vehiclesInRange[vehicle] == nil then
816 self.vehiclesInRange[vehicle] = true
817 self.numVehiclesInRange = self.numVehiclesInRange + 1
818
819 g_currentMission:removeActivatableObject(self.activatable) -- make sure
    is not added twice
820 g_currentMission:addActivatableObject(self.activatable)
821
822 -- add callback if shovel
823 if vehicle.setChangedFillLevelCallback ~= nil then
824 vehicle:setChangedFillLevelCallback(BunkerSilo.onChangedFillLevelCallbac
    self)
825 end
826 end
827 else
828 if self.vehiclesInRange[vehicle] then
829 self.vehiclesInRange[vehicle] = nil
830 self.numVehiclesInRange = self.numVehiclesInRange - 1
831
```

```
832 if self.numVehiclesInRange == 0 and not self.playerInRange then
833 g_currentMission:removeActivatableObject(self.activatable)
834 end
835
836 -- remove callback if shovel
837 if vehicle.setChangedFillLevelCallback ~= nil then
838 vehicle:setChangedFillLevelCallback(nil)
839 end
840 end
841 end
842 end
843 end
844 end
845 end
```

onChangedFillLevelCallback

Description

Called if fill level changed

Definition

onChangedFillLevelCallback(table vehicle, integer fillDelta, integer fillType)

Arguments

table	vehicle	vehicle
integer	fillDelta	fill delta
integer	fillType	fill type

Return Values

table self instance of class event

Code

```
852 function BunkerSilo.onChangedFillLevelCallback(self, vehicle,
    fillDelta, fillType)
853 if fillDelta >= 0 then
854 return
855 end
856
857 local area = self.bunkerSiloArea
858
859 local x,y,z = getWorldTranslation(vehicle.components[1].node)
860 local closerToFront = self:getIsCloserToFront(x,y,z)
861
862 if vehicle.shovel ~= nil and vehicle.shovel.pickUp ~= nil and
    vehicle.shovel.pickUp.node ~= nil then
863 x,y,z = localToWorld(vehicle.shovel.pickUp.node, 0, 0,
    vehicle.shovel.pickUp.length)
864 end
```

```
865
866 local length = self.openingLength
867
868 if closerToFront then
869 if self.isOpenedAtFront then
870 local p1 = MathUtil.getProjectOnLineParameter(x,z,
    area.sx,area.sz, area.dhx_norm,area.dhz_norm)
871 if p1 > area.offsetFront - length then
872 local offset = self:getBunkerAreaOffset(true, area.offsetFront,
    self.fermentingFillType)
873 local targetOffset = math.max(p1, offset) + length
874
875 self:switchFillTypeAtOffset(true, area.offsetFront, targetOffset
    - area.offsetFront)
876 area.offsetFront = targetOffset
877 end
878 end
879 else
880 if self.isOpenedAtBack then
881 local p1 = MathUtil.getProjectOnLineParameter(x,z,
    area.hx,area.hz, -area.dhx_norm,-area.dhz_norm)
882 if p1 > area.offsetBack - length then
883 local offset = self:getBunkerAreaOffset(true, area.offsetBack,
    self.fermentingFillType)
884 local targetOffset = math.max(p1, offset) + length
885
886 self:switchFillTypeAtOffset(false, area.offsetBack, targetOffset
    - area.offsetBack)
887 area.offsetBack = targetOffset
888 end
889 end
890 end
891 end
```

Farmland
Description

Class for field definitions

new
Description

Create field definition object

Definition

new()

Return Values

table instance Instance of object

Code

```
25  function Farmland:new(customMt)
26  local self = setmetatable({}, customMt or Farmland_mt)
27
28  self.isOwned = false
29  self.xWorldPos = 0
30  self.zWorldPos = 0
31
32  return self
33  end
```

load

Description

Load farmland data from xml

Definition

load(integer xmlFile, string key)

Arguments

integer xmlFile handle of xml file

string key current xml element key

Return Values

boolean true if loading was successful else false

Code

```
40  function Farmland:load(xmlFile, key)
41  self.id = getXMLInt(xmlFile, key.."#id")
42
43  if self.id == nil or self.id == 0 then
44  print("Error: Invalid farmland id '"..tostring(self.id).."'!")
45  return false
46  end
47
48  self.name = Utils.getNoNil(getXMLString(xmlFile, key.."#name"),
    "")
49  self.areaInHa = Utils.getNoNil(getXMLFloat(xmlFile,
    key.."#areaInHa"), 2.5)
50
51  self.fixedPrice = getXMLFloat(xmlFile, key.."#price")
52  if self.fixedPrice == nil then
53  self.priceFactor = Utils.getNoNil(getXMLFloat(xmlFile,
    key.."#priceScale"), 1)
54  end
55  self.price = self.fixedPrice or 1
56
```

```
57  self:updatePrice()
58
59  local npc = g_npcManager:getNPCByIndex(getXMLInt(xmlFile,
    key.."#npcIndex"))
60  self.npcIndex = g_npcManager:getRandomIndex()
61  if npc ~= nil then
62  self.npcIndex = npc.index
63  end
64
65  -- Names are used with custom NPC sets
66  local npc = g_npcManager:getNPCByName(getXMLString(xmlFile,
    key.."#npcName"))
67  if npc ~= nil then
68  self.npcIndex = npc.index
69  end
70
71  self.isOwned = false
72  self.defaultFarmProperty = Utils.getNoNil(getXMLBool(xmlFile,
    key.."#defaultFarmProperty"), false)
73
74  return true
75  end
```

delete

Description

Delete field definition object

Definition

delete()

Code

```
79  function Farmland:delete()
80  end
```

setFarmlandIndicatorPosition

Description

Set farmland area indicator world position

Definition

setFarmlandIndicatorPosition(float xWorldPos, float zWorldPos)

Arguments

float xWorldPos farmland indicator x world position

float zWorldPos farmland size in ha

Code

```
86  function Farmland:setFarmlandIndicatorPosition(xWorldPos,
    zWorldPos)
87  self.xWorldPos, self.zWorldPos = xWorldPos, zWorldPos
```

```
88  end
```

setArea

Description

Set farmland area

Definition

setArea(float areaInHa)

Arguments

float areaInHa farmland size in ha

Code

```
93  function Farmland:setArea(areaInHa)
94  self.areaInHa = areaInHa
95  if self.fixedPrice == nil then
96  self:updatePrice()
97  end
98  end
```

Field

Description

Class for field definitions

new

Description

Create ai field definition object

Definition

new()

Return Values

table instance Instance of object

Code

```
19  function Field:new(customMt)
20  local self = {}
21  setmetatable(self, customMt or Field_mt)
22
23  self.fieldId = 0
24  self.posX = 0
25  self.posZ = 0
26  self.rootNode = nil
27  self.name = nil
28  self.mapHotspot = nil
29  self.fieldMissionAllowed = true
30  self.fieldGrassMission = false
31  self.fieldAngle = 0.0
32  self.fieldDimensions = nil
33  self.fieldArea = 1.0
```

```
34  self.getFieldStatusPartitions = {}
35  self.setFieldStatusPartitions = {}
36  self.maxFieldStatusPartitions = {}
37  self.isAIActive = true
38  self.fruitType = nil -- current fruit in the field, as seen by FJM
39  self.lastCheckedTime = nil
40
41  self.currentMission = nil
42
43  return self
44  end
```

load

Description

Load Field data from node

Definition

load(integer id)

Arguments

integer id ai field node id

Return Values

boolean true if loading was successful else false

Code

```
50  function Field:load(id)
51  self.rootNode = id
52  self.name = Utils.getNoNil(getUserAttribute(id, "name"), "")
53
54  self.fieldMissionAllowed = Utils.getNoNil(getUserAttribute(id,
    "fieldMissionAllowed"), true)
55  self.fieldGrassMission = Utils.getNoNil(getUserAttribute(id,
    "fieldGrassMission"), false)
56
57  local fieldDimensions = I3DUtil.indexToObject(id,
    getUserAttribute(id, "fieldDimensionIndex"))
58  if fieldDimensions == nil then
59  print("Warning: No fieldDimensionIndex defined for Field
    '"..getName(id).."'!")
60  return false
61  end
62  local angleRad =
    math.rad(Utils.getNoNil(tonumber(getUserAttribute(id,
    "fieldAngle")), 0))
63
```

```
64  self.fieldAngle = FSDensityMapUtil.convertToDensityMapAngle(angleRad, g_currentMission.terrainDetailAngleMaxValue)
65  self.fieldDimensions = fieldDimensions
66
67  FieldUtil.updateFieldPartitions(self, self.getFieldStatusPartitions, 900)
68  FieldUtil.updateFieldPartitions(self, self.setFieldStatusPartitions, 400)
69  FieldUtil.updateFieldPartitions(self, self.maxFieldStatusPartitions, 10000000)
70
71  self.posX, self.posZ = FieldUtil.getCenterOfField(self)
72
73  self.nameIndicator = I3DUtil.indexToObject(id, getUserAttribute(id, "nameIndicatorIndex")) -- this is where the field number appears on the ingamemap
74  if self.nameIndicator ~= nil then
75  local x, _, z = getWorldTranslation(self.nameIndicator)
76  self.posX, self.posZ = x, z
77  end
78
79  self.farmland = nil
80
81  return true
82  end
```

delete

Description

Delete field definition object

Definition

delete()

Code

```
86  function Field:delete()
87  if self.mapHotspot == nil then
88  g_currentMission:removeMapHotspot(self.mapHotspot)
89  self.mapHotspot:delete()
90  self.mapHotspot = nil
91  end
92  end
```

HelpIcons

Description

Class for help icons

onCreate

Description

Creating help icons

Definition

onCreate(integer id)

Arguments

integer id node id

Return Values

float maxClutchTorque max clutch torque

Code

```
17  function HelpIcons:onCreate(id)
18  local helpIcons = HelpIcons:new(id);
19  g_currentMission:addNonUpdateable(helpIcons);
20  g_currentMission.helpIconsBase = helpIcons;
21  end;
```

new

Description

Creating help icons

Definition

new(integer name)

Arguments

integer name node id

Return Values

float rotInertia rotation inertia

table instance Instance of object

Code

```
27  function HelpIcons:new(name)
28  local self = {};
29  setmetatable(self, HelpIcons_mt);
30
31  self.me = name;
32  local num = getNumOfChildren(self.me);
33
34  self.helpIcons = {};
35  for i = 0, num - 1 do
36  local helpIconTriggerId = getChildAt(self.me, i);
37  local helpIconId = getChildAt(helpIconTriggerId, 0);
38  local helpIconCustomNumber =
    Utils.getNoNil(getUserAttribute(helpIconTriggerId,
    "customNumber"), 0);
39  addTrigger(helpIconTriggerId, "triggerCallback", self);
40  local helpIcon = {helpIconTriggerId = helpIconTriggerId,
    helpIconId = helpIconId, helpIconCustomNumber =
    helpIconCustomNumber};
```

```
41  table.insert(self.helpIcons, helpIcon);
42  end;
43  self.visible = true;
44
45  return self;
46  end;
```

delete

Description

Deleting help icons

Definition

delete()

Return Values

float dampingRate damping rate [t m^2 s^-1]

Code

```
50  function HelpIcons:delete()
51  for _, helpIcon in pairs(self.helpIcons) do
52  removeTrigger(helpIcon.helpIconTriggerId);
53  end;
54  end;
```

triggerCallback

Description

Trigger callback

Definition

triggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

Return Values

float dampingRate damping rate [t m^2 s^-1]

Code

```
71  function HelpIcons:triggerCallback(triggerId, otherId, onEnter, onLeave,
    onStay)
72  if onEnter then -- and g_currentMission.player ~= nil and otherId ==
    g_currentMission.player.rootNode and g_currentMission.controlPlayer ther
73  -- only trigger if the player or a vehicle controlled by the player ente
74  if (g_currentMission.player ~= nil and otherId ==
    g_currentMission.player.rootNode and g_currentMission.controlPlayer) or
    (g_currentMission.controlledVehicle ~= nil and
    g_currentMission.controlledVehicle ==
    g_currentMission.nodeToObject[otherId]) then
```

```
75  local missionInfo = g_currentMission.missionInfo
76
77  for i, helpIcon in ipairs(self.helpIcons) do -- order is important for
    savegame
78  if helpIcon.helpIconTriggerId == triggerId then
79  if getVisibility(helpIcon.helpIconId) then
80  setVisibility(helpIcon.helpIconId, false);
81  setCollisionMask(helpIcon.helpIconTriggerId, 0);
82
83  -- update help icon string
84  missionInfo.foundHelpIcons = "";
85  for _, helpIcon in ipairs(self.helpIcons) do
86  if getVisibility(helpIcon.helpIconId) then
87  missionInfo.foundHelpIcons = missionInfo.foundHelpIcons .. "0";
88  else
89  missionInfo.foundHelpIcons = missionInfo.foundHelpIcons .. "1";
90  end;
91  end;
92
93  local messageNumber = helpIcon.helpIconCustomNumber;
94  if messageNumber == 0 then
95  messageNumber = i;
96  end;
97  g_currentMission.inGameMessage:showMessage(g_i18n:getText("helpIcon_titl
    .. messageNumber), g_i18n:getText("helpIcon_text" .. messageNumber), 0);
98  end;
99  end;
100 end;
101
102 end;
103 end;
104 end;
```

showHelpIcons

Description

Show help icons

Definition

showHelpIcons(boolean visible, boolean clearIconStates)

Arguments

boolean visible	visible
boolean clearIconStates	clear icon states

Return Values

float dampingRate damping rate [t m^2 s^-1]

Code

```
110  function HelpIcons:showHelpIcons(visible, clearIconStates)
111  self.visible = visible;
112
113  local oldStates = g_currentMission.missionInfo.foundHelpIcons;
114
115  for i, helpIcon in ipairs(self.helpIcons) do
116  local isVisible = visible
117  if clearIconStates == nil or not clearIconStates then
118  isVisible = isVisible and string.sub(oldStates, i, i) == "0";
119  end;
120
121  setVisibility(helpIcon.helpIconId, isVisible);
122  if isVisible then
123  setCollisionMask(helpIcon.helpIconTriggerId, 3145728);
124  else
125  setCollisionMask(helpIcon.helpIconTriggerId, 0);
126  end;
127  end;
128
129  end;
```

deleteHelpIcon

Description

Delete help icon

Definition

deleteHelpIcon(integer i)

Arguments

integer i id of help icon

Return Values

float maxMotorTorque max motor torque

Code

```
134  function HelpIcons:deleteHelpIcon(i)
135  if self.helpIcons[i] ~= nil then
136  setVisibility(self.helpIcons[i].helpIconId, false);
137  setCollisionMask(self.helpIcons[i].helpIconTriggerId, 0);
138  end;
139  end;
```

LoadingStation

Description

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Return Values

float brakeForce brake force

Code

```
 98  function LoadingStation:readStream(streamId, connection)
 99  LoadingStation:superClass().readStream(self, streamId,
     connection)
100  if connection:getIsServer() then
101  for _, loadTrigger in ipairs(self.loadTriggers) do
102  local loadTriggerId = NetworkUtil.readNodeObjectId(streamId)
103  loadTrigger:readStream(streamId, connection)
104  g_client:finishRegisterObject(loadTrigger, loadTriggerId)
105  end
106  end
107  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Return Values

float minRpm min rpm

Code

```
113  function LoadingStation:writeStream(streamId, connection)
114  LoadingStation:superClass().writeStream(self, streamId,
     connection)
115  if not connection:getIsServer() then
116  for _, loadTrigger in ipairs(self.loadTriggers) do
117  NetworkUtil.writeNodeObjectId(streamId,
     NetworkUtil.getObjectId(loadTrigger))
118  loadTrigger:writeStream(streamId, connection)
119  g_server:registerObjectInStream(connection, loadTrigger)
120  end
121  end
122  end
```

getIsFillAllowedToFarm

Description

Get whether the given farm is allowed to fill objects from this loading station.

Definition

getIsFillAllowedToFarm()

Return Values

float maxRpm max rpm

NightIllumination

Description

Class for NightIllumination objects which are used for building windows that are illuminated at night

onCreate

Description

Creating NightIllumination object

Definition

onCreate(integer id)

Arguments

integer id ID of the node

Return Values

float minRequiredRpm min required rpm

float minRequiredRpm max required rpm

Code

```
19  function NightIllumination:onCreate(id)
20  g_currentMission:addNonUpdateable(NightIllumination:new(id))
21  end
```

new

Description

Creating NightIllumination object

Definition

new(integer name)

Arguments

integer name ID of the node

Return Values

table instance Instance of object

Code

```
27  function NightIllumination:new(id)
28  local self = {}
29  setmetatable(self, NightIllumination_mt)
30
31  g_currentMission.environment:addWeatherChangeListener(self)
32
33  self.id = id
34  self.windowsId = 0
```

```
35  self.lightsId = 0
36
37  if getNumOfChildren(id) > 0 then
38  self.windowsId = getChildAt(id, 0)
39  end
40  if getNumOfChildren(id) > 1 then
41  self.lightsId = getChildAt(id, 1)
42  end
43
44  self.lightIntensity = Utils.getNoNil(getUserAttribute(self.id,
    "lightIntensity"), 1.0)
45
46  -- set windows to dark (using dashboardLightsShader)
47  if self.windowsId ~= 0 then
48  setShaderParameter(self.windowsId, "lightControl", 0, 0, 0, 0,
    false)
49  end
50
51  -- make lights invisible
52  if self.lightsId ~= 0 then
53  setVisibility(self.lightsId, false)
54  end
55
56  return self
57  end
```

delete

Description

Remove Object from WeatherChangeListeners

Definition

delete()

Return Values

float lastMotorRpm last motor rpm

Code

```
61  function NightIllumination:delete()
62  if g_currentMission.environment ~= nil then
63  g_currentMission.environment:removeWeatherChangeListener(self)
64  end
65  end
```

weatherChanged

Description

Change illumination of night objects

Definition

weatherChanged()

Return Values

float lastMotorRpm last motor rpm

Code

```
69  function NightIllumination:weatherChanged()
70  if g_currentMission ~= nil and g_currentMission.environment ~= nil
    then
71  local isLightNeeded = not (g_currentMission.environment.isSunOn
    and not g_currentMission.environment.weather:getIsRaining())
72
73  if self.windowsId ~= 0 then
74  if isLightNeeded then
75  setShaderParameter(self.windowsId, "lightControl",
    self.lightIntensity, 0, 0, 0, false)
76  else
77  setShaderParameter(self.windowsId, "lightControl", 0, 0, 0, 0,
    false)
78  end
79  end
80
81  if self.lightsId ~= 0 then
82  setVisibility(self.lightsId, isLightNeeded)
83  end
84  end
85  end
```

Nightlight2

Description

Class for nightlight objects which are blending in on night

onCreate

Description

Creating nightlight object

Definition

onCreate(integer id)

Arguments

integer id ID of the node

Return Values

float appliedTorque torque [kN]

Code

```
15  function Nightlight2:onCreate(id)
16  g_currentMission:addNonUpdateable(Nightlight2:new(id))
17  end
```

new

Description

Creating nightlight object

Definition

new(integer name)

Arguments

integer name ID of the node

Return Values

float externalTorque external torque [kN]

table instance Instance of object

Code

```
23  function Nightlight2:new(id)
24  local self = {}
25  setmetatable(self, Nightlight2_mt)
26
27  self.id = id
28  self.switchCollision = Utils.getNoNil(getUserAttribute(id,
    "switchCollision"), false)
29
30  if self.switchCollision then
31  self.collisionMask = getCollisionMask(id)
32  end
33
34  self:setVisibility(false)
35
36  g_currentMission.environment:addWeatherChangeListener(self)
37
38  return self
39  end
```

delete

Description

Remove Object from WeatherChangeListeners

Definition

delete()

Return Values

float torque external torque [kN]

Code

```
43  function Nightlight2:delete()
44  if g_currentMission.environment ~= nil then
45  g_currentMission.environment:removeWeatherChangeListener(self)
46  end
47  end
```

weatherChanged

Description

Change visibility of night object

Definition

weatherChanged()

Return Values

float equalizedMotorRpm equalized motor rpm

Code

```
59  function Nightlight2:weatherChanged()
60  if g_currentMission ~= nil and g_currentMission.environment ~= nil
    then
61  self:setVisibility(not (g_currentMission.environment.isSunOn and
    not g_currentMission.environment.weather:getIsRaining()))
62  end
63  end
```

NightlightFlicker

Description

NightlightFlickers are flickering lights that are only active at night or during bad weather

onCreate

Description

Creating nightlightflicker

Definition

onCreate(integer id)

Arguments

integer id node id

Return Values

float ptoMotorRpmRatio pto motor rpm ratio

Code

```
15  function NightlightFlicker:onCreate(id)
16  g_currentMission:addUpdateable(NightlightFlicker:new(id));
17  end;
```

new

Description

Creating nightlightflicker

Definition

new(integer name)

Arguments

integer name node id

Return Values

float nonClampedMotorRpm non clamped motor rpm

table instance Instance of object

Code

```
23  function NightlightFlicker:new(id)
24  local self = {};
```

```lua
setmetatable(self, NightlightFlicker_mt);

self.id = id;
self.isVisible = false;
self.isFlickerActive = false;
self.nextFlicker = 0;
self.flickerDuration = 100;
setVisibility(self.id, self.isVisible);

g_currentMission.environment:addWeatherChangeListener(self);

return self;
end;
```

update

Description

Update flickering

Definition

update(float dt)

Arguments

float dt time since last call in ms

Return Values

float nonClampedMotorRpm non clamped motor rpm

Code

```lua
function NightlightFlicker:update(dt)
if self.isVisible then

self.nextFlicker = self.nextFlicker - dt;
if self.nextFlicker <= 0 then
self.isFlickerActive = true;
setVisibility(self.id, false);
self.nextFlicker = math.floor(math.random() * 1500 +
self.flickerDuration + 10); -- set next flicker at least 10ms
after this one
end;

if self.isFlickerActive then
self.flickerDuration = self.flickerDuration - dt;
if self.flickerDuration <= 0 then
self.isFlickerActive = false;
self.flickerDuration = math.floor(math.random() * 200);
setVisibility(self.id, true);
```

```
61  end;
62  end;
63
64  end;
65  end;
```

weatherChanged

Description

Change visibility of night object

Definition

weatherChanged()

Return Values

float clutchRpm clutch rpm

Code

```
69  function NightlightFlicker:weatherChanged()
70  if g_currentMission ~= nil and g_currentMission.environment ~= nil
    then
71  self.isVisible = not (g_currentMission.environment.isSunOn and not
    g_currentMission.environment.weather:getIsRaining());
72  setVisibility(self.id, self.isVisible);
73  end;
74  end;
```

PhysicsObject

Description

Class for physics objects

new

Description

Creating physics object

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean	isServer	is server
boolean	isClient	is client
table	customMt	customMt

Return Values

table	torqueCurve	torque curve
table	instance	Instance of object

Code

```
17  function PhysicsObject:new(isServer, isClient, customMt)
18
19  local self = Object:new(isServer, isClient, customMt or
    PhysicsObject_mt)
20
21  self.nodeId = 0
```

```
22  self.networkTimeInterpolator = InterpolationTime:new(1.2)
23  self.forcedClipDistance = 60
24
25  self.physicsObjectDirtyFlag = self:getNextDirtyFlag()
26
27  return self
28  end
```

delete

Description

Deleting physics object

Definition

delete()

Return Values

float torque torque

Code

```
32  function PhysicsObject:delete()
33  removeWakeUpReport(self.nodeId)
34  g_currentMission:removeNodeObject(self.nodeId)
35  delete(self.nodeId)
36  self.nodeId = 0
37  PhysicsObject:superClass().delete(self)
38  end
```

getAllowsAutoDelete

Description

Get allows auto delete

Definition

getAllowsAutoDelete()

Return Values

float torque torque

boolean allowsAutoDelete allows auto delete

Code

```
43  function PhysicsObject:getAllowsAutoDelete()
44  return true
45  end
```

loadOnCreate

Description

Load on create

Definition

loadOnCreate(integer nodeId)

Arguments

integer nodeId node id

Return Values

float maxForwardSpeed maximum forward speed

Code

```
50  function PhysicsObject:loadOnCreate(nodeId)
51  self:setNodeId(nodeId)
52  if not self.isServer then
53  self:onGhostRemove()
54  end
55  end
```

setNodeId

Description

Set node id

Definition

setNodeId(integer nodeId)

Arguments

integer nodeId node Id

Return Values

float maxBackwardSpeed maximum backward speed

Code

```
60  function PhysicsObject:setNodeId(nodeId)
61  self.nodeId = nodeId
62  setRigidBodyType(self.nodeId, self:getDefaultRigidBodyType())
63  addToPhysics(self.nodeId)
64
65  local x, y, z = getTranslation(self.nodeId)
66  local xRot, yRot, zRot = getRotation(self.nodeId)
67  self.sendPosX, self.sendPosY, self.sendPosZ = x, y, z
68  self.sendRotX, self.sendRotY, self.sendRotZ = xRot, yRot, zRot
69
70  if not self.isServer then
71  local quatX, quatY, quatZ, quatW = mathEulerToQuaternion(xRot,
    yRot, zRot)
72  self.positionInterpolator = InterpolatorPosition:new(x, y, z)
73  self.quaternionInterpolator = InterpolatorQuaternion:new(quatX,
    quatY, quatZ, quatW)
74  end
75
76  self:addChildsToNodeObject(self.nodeId)
77  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Return Values

float physicalMaxForwardSpeed physical maximum forward speed

Code

```
83  function PhysicsObject:readStream(streamId, connection)
84  assert(self.nodeId ~= 0)
85  if connection:getIsServer() then
86  local paramsXZ = g_currentMission.vehicleXZPosCompressionParams
87  local paramsY = g_currentMission.vehicleYPosCompressionParams
88  local x = NetworkUtil.readCompressedWorldPosition(streamId,
    paramsXZ)
89  local y = NetworkUtil.readCompressedWorldPosition(streamId,
    paramsY)
90  local z = NetworkUtil.readCompressedWorldPosition(streamId,
    paramsXZ)
91  local xRot = NetworkUtil.readCompressedAngle(streamId)
92  local yRot = NetworkUtil.readCompressedAngle(streamId)
93  local zRot = NetworkUtil.readCompressedAngle(streamId)
94
95  local quatX, quatY, quatZ, quatW =
    mathEulerToQuaternion(xRot,yRot,zRot)
96  self:setWorldPositionQuaternion(x, y, z, quatX, quatY, quatZ,
    quatW, true)
97
98  self.networkTimeInterpolator:reset()
99  end
100 end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Return Values

float physicalMaxBackwardSpeed physical maximum backward speed

Code

```
106 function PhysicsObject:writeStream(streamId, connection)
107 if not connection:getIsServer() then
```

```
108 local x,y,z = getTranslation(self.nodeId)
109 local xRot,yRot,zRot = getRotation(self.nodeId)
110 local paramsXZ = g_currentMission.vehicleXZPosCompressionParams
111 local paramsY = g_currentMission.vehicleYPosCompressionParams
112 NetworkUtil.writeCompressedWorldPosition(streamId, x, paramsXZ)
113 NetworkUtil.writeCompressedWorldPosition(streamId, y, paramsY)
114 NetworkUtil.writeCompressedWorldPosition(streamId, z, paramsXZ)
115 NetworkUtil.writeCompressedAngle(streamId, xRot)
116 NetworkUtil.writeCompressedAngle(streamId, yRot)
117 NetworkUtil.writeCompressedAngle(streamId, zRot)
118 end
119 end
```

readUpdateStream

Description

Called on client side on update

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId	stream ID
integer timestamp	timestamp
table	connection connection

Return Values

float physicalMaxSpeed physical maximum speed

Code

```
126 function PhysicsObject:readUpdateStream(streamId, timestamp,
    connection)
127 if connection:getIsServer() then
128 if streamReadBool(streamId) then
129 local paramsXZ = g_currentMission.vehicleXZPosCompressionParams
130 local paramsY = g_currentMission.vehicleYPosCompressionParams
131 local x = NetworkUtil.readCompressedWorldPosition(streamId,
    paramsXZ)
132 local y = NetworkUtil.readCompressedWorldPosition(streamId,
    paramsY)
133 local z = NetworkUtil.readCompressedWorldPosition(streamId,
    paramsXZ)
134 local xRot = NetworkUtil.readCompressedAngle(streamId)
135 local yRot = NetworkUtil.readCompressedAngle(streamId)
136 local zRot = NetworkUtil.readCompressedAngle(streamId)
137
138 local quatX, quatY, quatZ, quatW =
    mathEulerToQuaternion(xRot,yRot,zRot)
```

```
139  self.positionInterpolator:setTargetPosition(x, y, z)
140  self.quaternionInterpolator:setTargetQuaternion(quatX, quatY,
     quatZ, quatW)
141  self.networkTimeInterpolator:startNewPhaseNetwork()
142  end
143  end
144  end
```

writeUpdateStream

Description

Called on server side on update

Definition

writeUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Return Values

float bestGearRatio best gear ratio

Code

```
151  function PhysicsObject:writeUpdateStream(streamId, connection,
     dirtyMask)
152  if not connection:getIsServer() then
153  if streamWriteBool(streamId, bitAND(dirtyMask,
     self.physicsObjectDirtyFlag) ~= 0) then
154  local paramsXZ = g_currentMission.vehicleXZPosCompressionParams
155  local paramsY = g_currentMission.vehicleYPosCompressionParams
156  NetworkUtil.writeCompressedWorldPosition(streamId, self.sendPosX,
     paramsXZ)
157  NetworkUtil.writeCompressedWorldPosition(streamId, self.sendPosY,
     paramsY)
158  NetworkUtil.writeCompressedWorldPosition(streamId, self.sendPosZ,
     paramsXZ)
159
160  NetworkUtil.writeCompressedAngle(streamId, self.sendRotX)
161  NetworkUtil.writeCompressedAngle(streamId, self.sendRotY)
162  NetworkUtil.writeCompressedAngle(streamId, self.sendRotZ)
163  end
164  end
165  end
```

update

Description

Update

Definition

update(float dt)

Arguments

float dt time since last call in ms

Return Values

float bestGear best gear

float gearRatio gear ratio

Code

```
170 function PhysicsObject:update(dt)
171 if not self.isServer then
172 self.networkTimeInterpolator:update(dt)
173 local interpolationAlpha = self.networkTimeInterpolator:getAlpha()
174 local posX, posY, posZ =
    self.positionInterpolator:getInterpolatedValues(interpolationAlpha)
175 local quatX, quatY, quatZ, quatW =
    self.quaternionInterpolator:getInterpolatedValues(interpolationAlpha)
176 self:setWorldPositionQuaternion(posX, posY, posZ, quatX, quatY,
    quatZ, quatW, false)
177
178 if self.networkTimeInterpolator:isInterpolating() then
179 self:raiseActive()
180 end
181 else
182 if not getIsSleeping(self.nodeId) then
183 self:raiseActive()
184 end
185 end
186 end
```

updateMove

Description

Update move

Definition

updateMove()

Return Values

boolean hasMoved has moved

Code

```
192 function PhysicsObject:updateMove()
193 local x, y, z = getTranslation(self.nodeId)
194 local xRot, yRot, zRot = getRotation(self.nodeId)
195 local hasMoved = math.abs(self.sendPosX-x)>0.005 or
    math.abs(self.sendPosY-y)>0.005 or math.abs(self.sendPosZ-
    z)>0.005 or
196 math.abs(self.sendRotX-xRot)>0.02 or math.abs(self.sendRotY-
    yRot)>0.02 or math.abs(self.sendRotZ-zRot)>0.02
```

```
197
198  if hasMoved then
199  self:raiseDirtyFlags(self.physicsObjectDirtyFlag)
200  self.sendPosX, self.sendPosY, self.sendPosZ = x, y ,z
201  self.sendRotX, self.sendRotY, self.sendRotZ = xRot, yRot, zRot
202  end
203
204  return hasMoved
205  end
```

updateTick

Description

updateTick

Definition

updateTick(float dt)

Arguments

float dt time since last call in ms

Return Values

float adjustedAcceleratorPedal	the adjusted accelerator pedal for the current gear situation (e.g. 0 while switching gears)

Code

```
210  function PhysicsObject:updateTick(dt)
211  if self.isServer then
212  self:updateMove()
213  end
214  end
```

testScope

Description

Test scope

Definition

testScope(float x, float y, float z, float coeff)

Arguments

float	x	x position
float	y	y position
float	z	z position
float	coeff	coeff

Return Values

float	minGearRatio	minimum gear ratio
float	maxGearRatio	maximum gear ratio
boolean	inScope	in scope

Code

```
223  function PhysicsObject:testScope(x,y,z, coeff)
224  local x1, y1, z1 = getWorldTranslation(self.nodeId)
```

```
225 local dist = (x1-x)*(x1-x) + (y1-y)*(y1-y) + (z1-z)*(z1-z)
226 local clipDist = math.min(getClipDistance(self.nodeId)*coeff,
    self.forcedClipDistance)
227 if dist < clipDist*clipDist then
228 return true
229 else
230 return false
231 end
232 end
```

getUpdatePriority

Description

Get update priority

Definition

getUpdatePriority(float skipCount, float x, float y, float z, float coeff, table connection)

Arguments

float	skipCount	skip count
float	x	x position
float	y	y position
float	z	z position
float	coeff	coeff
table	connection	connection

Return Values

float priority priority

Code

```
243 function PhysicsObject:getUpdatePriority(skipCount, x, y, z,
    coeff, connection)
244 local x1, y1, z1 = getWorldTranslation(self.nodeId)
245 local dist = math.sqrt((x1-x)*(x1-x) + (y1-y)*(y1-y) + (z1-
    z)*(z1-z))
246 local clipDist = math.min(getClipDistance(self.nodeId)*coeff,
    self.forcedClipDistance)
247 return (1-dist/clipDist)* 0.8 + 0.5*skipCount * 0.2
248 end
```

onGhostRemove

Description

On ghost remove

Definition

onGhostRemove()

Return Values

integer maxRpm current max rpm

Code

```
252 function PhysicsObject:onGhostRemove()
253 setVisibility(self.nodeId, false)
```

```
254  removeFromPhysics(self.nodeId)
255  end
```

onGhostAdd

Description

On ghost add

Definition

onGhostAdd()

Return Values

boolean continue continue

Code

```
259  function PhysicsObject:onGhostAdd()
260  setVisibility(self.nodeId, true)
261  addToPhysics(self.nodeId)
262  end
```

setWorldPositionQuaternion

Description

Set pose

Definition

setWorldPositionQuaternion(float x, float y, float z, float xRot, float yRot, float zRot, float w_rot)

Arguments

float x	x position
float y	z position
float z	z position
float xRot	x rotation
float yRot	y rotation
float zRot	z rotation
float w_rot	w rotation

Return Values

table instance instance of object

Code

```
273  function PhysicsObject:setWorldPositionQuaternion(x, y, z, quatX,
     quatY, quatZ, quatW, changeInterp)
274  if not self.isServer then
275  if changeInterp then
276  self.positionInterpolator:setPosition(x, y, z)
277  self.quaternionInterpolator:setQuaternion(quatX, quatY, quatZ,
     quatW)
278  end
279  end
280
281  setTranslation(self.nodeId, x, y, z)
282  setQuaternion(self.nodeId, quatX, quatY, quatZ, quatW)
```

```
283 end
```

getDefaultRigidBodyType

Description

Get default rigid body type

Definition

getDefaultRigidBodyType()

Return Values

boolean true if loading was successful else false

string rigidBodyType rigid body type

Code

```
288 function PhysicsObject:getDefaultRigidBodyType()
289 if self.isServer then
290 return "Dynamic"
291 else
292 return "Kinematic"
293 end
294 end
```

addChildsToNodeObject

Description

Add childs to node object

Definition

addChildsToNodeObject(integer nodeId)

Arguments

integer nodeId id of node

Return Values

boolean success true if added else false

Code

```
299 function PhysicsObject:addChildsToNodeObject(nodeId)
300 for i=0,getNumOfChildren(nodeId)-1 do
301 self:addChildsToNodeObject(getChildAt(nodeId, i))
302 end
303 local rigidBodyType = getRigidBodyType(nodeId)
304 if rigidBodyType ~= "NoRigidBody" then
305 g_currentMission:addNodeObject(nodeId, self)
306
307 if self.isServer then
308 addWakeUpReport(nodeId, "onPhysicObjectWakeUpCallback", self)
309 end
310 end
311 end
```

Rotator

Description

Rotators rotate around their y axis

onCreate

Description

Creating rotator

Definition

onCreate(integer id)

Arguments

integer id node id

Return Values

integer i index of tire type

Code

```
15  function Rotator:onCreate(id)
16  g_currentMission:addUpdateable(Rotator:new(id))
17  end
```

new

Description

Creating rotator

Definition

new(integer name)

Arguments

integer name node id

Return Values

float diffRotSpeed rot speed [rad/sec]

table instance Instance of object

Code

```
23  function Rotator:new(name)
24  local self = {}
25  setmetatable(self, Rotator_mt)
26
27  self.axisTable = {0, 0, 0}
28  self.me = name
29  self.speed = Utils.getNoNil(getUserAttribute(name, "speed"),
    0.0012)
30  local axis = Utils.getNoNil(getUserAttribute(name, "axis"), 3)
31  self.axisTable[axis] = 1
32
33  return self
34  end
```

update

Description

Update

Definition

update(float dt)

Arguments

float dt time since last call in ms

Return Values

float tireFriction tire friction

Code

```
42  function Rotator:update(dt)
43  rotate(self.me, self.axisTable[1] * self.speed * dt,
    self.axisTable[2] * self.speed * dt, self.axisTable[3] *
    self.speed * dt)
44  end
```

SellingStation

Description

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key)

Arguments

integer xmlFile id of xml object

string key key

Return Values

integer groundType ground type

boolean success success

Code

```
227 function SellingStation:loadFromXMLFile(xmlFile, key)
228 local i=0
229 while true do
230 local statsKey = string.format(key..".stats(%d)", i)
231 if not hasXMLProperty(xmlFile, statsKey) then
232 break
233 end
234 local fillTypeStr = getXMLString(xmlFile, statsKey.."#fillType")
235 local fillType =
    g_fillTypeManager:getFillTypeIndexByName(fillTypeStr)
236 if fillType ~= nil and self.acceptedFillTypes[fillType] then
237 self.totalReceived[fillType] =
    Utils.getNoNil(getXMLFloat(xmlFile, statsKey.."#received"), 0)
238 self.totalPaid[fillType] = Utils.getNoNil(getXMLFloat(xmlFile,
    statsKey.."#paid"), 0)
239 self.pricingDynamics[fillType]:loadFromXMLFile(xmlFile, statsKey)
240 end
241 i = i + 1
242 end
```

```
243
244  return true
245  end
```

initPricingDynamics

Description

Initialize pricing dynamics

Definition

initPricingDynamics()

Return Values

table instance instance of object

SimParticleSystem

Description

This pre-simulates a particle system so it doesn't start at zero when the game begins

onCreate

Description

Creating SimParticleSystem

Definition

onCreate(integer id)

Arguments

integer id node id

Code

```
15  function SimParticleSystem:onCreate(id)
16  g_currentMission:addNonUpdateable(SimParticleSystem:new(id));
17  end;
```

new

Description

Creating SimParticleSystem

Definition

new(integer name)

Arguments

integer name node id

Return Values

table instance Instance of object

Code

```
23  function SimParticleSystem:new(name)
24  local self = {};
25  setmetatable(self, SimParticleSystem_mt);
26  self.id = name;
27
28  local particleSystem = nil;
29
30  if getHasClassId(self.id, ClassIds.SHAPE) then
```

```
31  local geometry = getGeometry(self.id);
32  if geometry ~= 0 then
33  if getHasClassId(geometry, ClassIds.PRECIPITATION) then
34  particleSystem = geometry;
35  end;
36  end;
37  end;
38
39  if particleSystem ~= nil then
40  local lifespan = getParticleSystemLifespan(particleSystem);
41  addParticleSystemSimulationTime(particleSystem, lifespan);
42  end;
43
44  return self;
45  end;
```

Storage

Description

new

Description

Creating new instance of storage class

Definition

new(boolean isServer, boolean isClient)

Arguments

boolean isServer is server

boolean isClient is client

Return Values

table self new instance of object

Code

```
23  function Storage:new(isServer, isClient, customMt)
24  local self = Object:new(isServer, isClient, customMt or
    Storage_mt)
25
26  self.unloadingStations = {}
27  self.loadingStations = {}
28
29  self.rootNode = 0
30
31  return self
32  end
```

load

Description

Load storage attributes from object

Definition

load(integer id)

Arguments

integer id id of object

Return Values

boolean success success

Code

```
38  function Storage:load(id, xmlFile, key)
39  self.rootNode = id
40
41  self.capacityPerFillType = getXMLFloat(xmlFile, key ..
    "#capacityPerFillType") or 100000
42  self.costsPerFillLevelAndDay = getXMLFloat(xmlFile, key ..
    "#costsPerFillLevelAndDay") or 0
43
44  self.fillTypes = {}
45  self.fillLevels = {}
46  self.sortedFillTypes = {}
47
48  local fillTypeCategories = getXMLString(xmlFile, key ..
    "#fillTypeCategories")
49  local fillTypeNames = getXMLString(xmlFile, key .. "#fillTypes")
50  local fillTypes
51
52  if fillTypeCategories ~= nil and fillTypeNames == nil then
53  fillTypes =
    g_fillTypeManager:getFillTypesByCategoryNames(fillTypeCategories,
    "Warning: '"..tostring(key).. "' has invalid fillTypeCategory
    '%s'.")
54  elseif fillTypeCategories == nil and fillTypeNames ~= nil then
55  fillTypes = g_fillTypeManager:getFillTypesByNames(fillTypeNames,
    "Warning: '"..tostring(key).. "' has invalid fillType '%s'.")
56  else
57  print("Warning: '"..tostring(key).. "' a 'Storage' entry needs
    either the 'fillTypeCategories' or 'fillTypes' attribute.")
58  return false
59  end
60
61  for _,fillType in pairs(fillTypes) do
62  self.fillTypes[fillType] = true
63  end
64
65  for fillType,_ in pairs(self.fillTypes) do
```

```
66  table.insert(self.sortedFillTypes, fillType)
67  self.fillLevels[fillType] = 0
68  end
69  table.sort(self.sortedFillTypes)
70
71  self.storageDirtyFlag = self:getNextDirtyFlag()
72
73  g_messageCenter:subscribe(MessageType.FARM_DELETED,
    self.farmDestroyed, self)
74
75  return true
76  end
```

delete

Description

Deleting storage

Definition

delete()

Code

```
80  function Storage:delete()
81  if self.rootNode ~= 0 and entityExists(self.rootNode) then
82  delete(self.rootNode)
83  end
84
85  g_currentMission.environment:removeHourChangeListener(self)
86  g_messageCenter:unsubscribeAll(self)
87
88  Storage:superClass().delete(self)
89  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId	stream ID
table	connection connection

Code

```
95  function Storage:readStream(streamId, connection)
96  Storage:superClass().readStream(self, streamId, connection)
97
98  for _, fillType in ipairs(self.sortedFillTypes) do
```

```
 99  local fillLevel = 0
100  if streamReadBool(streamId) then
101  fillLevel = streamReadFloat32(streamId)
102  end
103  self:setFillLevel(fillLevel, fillType)
104  end
105  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId	stream ID
table	connection connection

Code

```
111  function Storage:writeStream(streamId, connection)
112  Storage:superClass().writeStream(self, streamId, connection)
113
114  for _, fillType in ipairs(self.sortedFillTypes) do
115  local fillLevel = self.fillLevels[fillType]
116  if streamWriteBool(streamId, fillLevel > 0) then
117  streamWriteFloat32(streamId, fillLevel)
118  end
119  end
120  end
```

readUpdateStream

Description

Called on client side on update

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId	stream ID
integer timestamp	timestamp
table	connection connection

Code

```
127  function Storage:readUpdateStream(streamId, timestamp,
     connection)
128  Storage:superClass().readUpdateStream(self, streamId, timestamp,
     connection)
129  if connection:getIsServer() then
130  if streamReadBool(streamId) then
```

```
131 for _, fillType in ipairs(self.sortedFillTypes) do
132 local fillLevel = 0
133 if streamReadBool(streamId) then
134 fillLevel = streamReadFloat32(streamId)
135 end
136 self:setFillLevel(fillLevel, fillType)
137 end
138 end
139 end
140 end
```

writeUpdateStream

Description

Called on server side on update

Definition

writeUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Code

```
147 function Storage:writeUpdateStream(streamId, connection,
    dirtyMask)
148 Storage:superClass().writeUpdateStream(self, streamId,
    connection, dirtyMask)
149 if not connection:getIsServer() then
150 if streamWriteBool(streamId, bitAND(dirtyMask,
    self.storageDirtyFlag) ~= 0) then
151 for _, fillType in ipairs(self.sortedFillTypes) do
152 local fillLevel = self.fillLevels[fillType]
153 if streamWriteBool(streamId, fillLevel > 0) then
154 streamWriteFloat32(streamId, fillLevel)
155 end
156 end
157 end
158 end
159 end
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key)

Arguments

integer xmlFile id of xml object

string key key

Return Values

boolean success success

Code

```
166 function Storage:loadFromXMLFile(xmlFile, key)
167 self:setOwnerFarmId(Utils.getNoNil(getXMLInt(xmlFile, key .. "#farmId"), AccessHandler.EVERYONE), true)
168
169 local i = 0
170 while true do
171 local siloKey = string.format(key .. ".node(%d)", i)
172 if not hasXMLProperty(xmlFile, siloKey) then
173 break
174 end
175
176 local fillTypeStr = getXMLString(xmlFile, siloKey.."#fillType")
177 local fillLevel = math.max(Utils.getNoNil(getXMLFloat(xmlFile, siloKey.."#fillLevel"), 0), 0)
178 local fillTypeIndex = g_fillTypeManager:getFillTypeIndexByName(fillTypeStr)
179
180 if fillTypeIndex ~= nil then
181 if self.fillLevels[fillTypeIndex] ~= nil then
182 self:setFillLevel(fillLevel, fillTypeIndex, nil)
183 else
184 print("Warning: Filltype '"..fillTypeStr.."' not supported by Storage "..getName(self.rootNode))
185 end
186 else
187 print("Error: Invalid filltype '"..fillTypeStr.."'")
188 end
189
190 i = i + 1
191 end
192
193 return true
194 end
```

getIsFillTypeSupported

Description

Returns if storage allows fill type

Definition

getIsFillTypeSupported(integer fillType)

Arguments

integer fillType fill type

Return Values

boolean allow allow fill type

Code

```
224  function Storage:getIsFillTypeSupported(fillType)
225  return self.fillTypes[fillType] == true
226  end
```

getFillLevel

Description

Returns fill level

Definition

getFillLevel(integer fillType)

Arguments

integer fillType fill type

Return Values

float fillLevel fill level

Code

```
232  function Storage:getFillLevel(fillType)
233  return self.fillLevels[fillType] or 0
234  end
```

setFillLevel

Description

Set fill level of storage

Definition

setFillLevel(float fillLevel, integer fillType)

Arguments

float fillLevel new fill level

integer fillType fill type

Code

```
244  function Storage:setFillLevel(fillLevel, fillType)
245  fillLevel = MathUtil.clamp(fillLevel, 0,
     self.capacityPerFillType)
246  if self.fillLevels[fillType] ~= nil and fillLevel ~=
     self.fillLevels[fillType] then
247  self.fillLevels[fillType] = fillLevel
248
249  if self.isServer then
250  self:raiseDirtyFlags(self.storageDirtyFlag)
251  end
252  end
253  end
```

getFreeCapacity

Description

Returns free capacity

Definition

getFreeCapacity(integer fillType)

Arguments

integer fillType fill type

Return Values

float freeCapacity free capacity

Code

```
259 function Storage:getFreeCapacity(fillType)
260 if self.fillLevels[fillType] ~= nil then
261 return math.max(self.capacityPerFillType -
    self.fillLevels[fillType], 0)
262 end
263 return 0
264 end
```

hourChanged

Description

Called on hour change

Definition

hourChanged()

Code

```
272 function Storage:hourChanged()
273 if self.isServer then
274 local difficultyMultiplier =
    g_currentMission.missionInfo.buyPriceMultiplier
275 local fillLevelFactor = difficultyMultiplier *
    self.costsPerFillLevelAndDay / 24
276
277 local costs = 0
278 for _, fillLevel in pairs(self.fillLevels) do
279 costs = costs + fillLevel * fillLevelFactor
280 end
281
282 g_currentMission:addMoney(-costs, self:getOwnerFarmId(),
    "propertyMaintenance")
283 g_currentMission:addMoneyChange(-costs, self:getOwnerFarmId(),
    EconomyManager.MONEY_TYPE_PROPERTY_MAINTENANCE)
284 end
285 end
```

SunAdmirer

Description

Class for objects which are visible when the sun is out

onCreate

Description

Creating sun admirer object

Definition

onCreate(integer id)

Arguments

integer id ID of the node

Code

```
15  function SunAdmirer:onCreate(id)
16  g_currentMission:addNonUpdateable(SunAdmirer:new(id))
17  end
```

new

Description

Creating nightlight object

Definition

new(integer name)

Arguments

integer name ID of the node

Return Values

table instance Instance of object

Code

```
23  function SunAdmirer:new(id)
24  local self = {}
25  setmetatable(self, SunAdmirer_mt)
26
27  self.id = id
28  self.switchCollision = Utils.getNoNil(getUserAttribute(id,
    "switchCollision"), false)
29
30  if self.switchCollision then
31  self.collisionMask = getCollisionMask(id)
32  end
33
34  self:setVisibility(true)
35
36  g_currentMission.environment:addWeatherChangeListener(self)
37
38  return self
39  end
```

delete

Description

Remove Object from WeatherChangeListeners

Definition

delete()

Code

```
43  function SunAdmirer:delete()
44  if g_currentMission.environment ~= nil then
45  g_currentMission.environment:removeWeatherChangeListener(self)
46  end
47  end
```

weatherChanged

Description

Change visibility of sun object

Definition

weatherChanged()

Code

```
59  function SunAdmirer:weatherChanged()
60  if g_currentMission ~= nil and g_currentMission.environment ~= nil
    then
61  self:setVisibility(g_currentMission.environment.isSunOn and not
    g_currentMission.environment.weather:getIsRaining())
62  end
63  end
```

TourIcons

Description

Tour icons are part of the (optional) guided tour at the career game's start

onCreate

Description

Creating tour icons

Definition

onCreate(integer id)

Arguments

integer id node id

Code

```
15  function TourIcons:onCreate(id)
16  local tourIcons = TourIcons:new(id)
17  table.insert(g_currentMission.updateables, tourIcons)
18  g_currentMission.tourIconsBase = tourIcons
19  end
```

new

Description

Creating tour icons

Definition

new(integer id)

Arguments

integer id node id

Return Values

table instance Instance of object

Code

```
25  function TourIcons:new(name)
26  local self = {}
27  setmetatable(self, TourIcons_mt)
28
29  self.me = name
30  local num = getNumOfChildren(self.me)
31
32  self.tourIcons = {}
33  for i = 0, num - 1 do
34  local tourIconTriggerId = getChildAt(self.me, i)
35  local tourIconId = getChildAt(tourIconTriggerId, 0)
36  addTrigger(tourIconTriggerId, "triggerCallback", self)
37  setVisibility(tourIconId, false)
38  local tourIcon = {tourIconTriggerId = tourIconTriggerId,
    tourIconId = tourIconId}
39  table.insert(self.tourIcons, tourIcon)
40  end
41
42  self.visible = false
43  self.mapHotspot = nil
44  self.currentTourIconNumber = 1
45  self.alpha = 0.25
46  self.alphaDirection = 1
47  self.startTourDialog = false
48  self.startTourDialogDelay = 0
49  self.permanentMessageDelay = 0
50  self.isPaused = false
51  self.pauseTime = 0
52  self.soldStuffAtGrainElevator = false
53
54  _, self.permanentTextSize = getNormalizedScreenValues(0, 28)
55
56  return self
57  end
```

delete

Description

Deleting tour icons

Definition

delete()

Code

```
61  function TourIcons:delete()
62  for _, tourIcon in pairs(self.tourIcons) do
63  removeTrigger(tourIcon.tourIconTriggerId)
64  end
65  end
```

showTourDialog

Description

Show tour yes/no dialog

Definition

showTourDialog()

Code

```
69  function TourIcons:showTourDialog()
70  g_gui:showYesNoDialog({text=g_i18n:getText("tour_text_start"),
    title="", callback=self.reactToDialog, target=self})
71  end
```

reactToDialog

Description

React to tour dialog

Definition

reactToDialog(boolean yes)

Arguments

boolean yes answer to dialog

Code

```
76  function TourIcons:reactToDialog(yes)
77  if yes then
78  self.visible = true
79  self:activateNextIcon()
80  -- hide all non-tour question marks
81  if g_currentMission.helpIconsBase ~= nil then
82  g_currentMission.helpIconsBase:showHelpIcons(false, true)
83  end
84  else
85  self.visible = false
86  g_currentMission.hud:showInGameMessage("",
    g_i18n:getText("tour_text_abort"), -1)
87  end
88
```

```
89  end
```

update

Description

Update

Definition

update(float dt)

Arguments

float dt time since last call in ms

Code

```
94  function TourIcons:update(dt)
95  if not g_currentMission.missionInfo.isValid and g_server ~= nil and
    self.initDone == nil and g_currentMission:getIsTourSupported() then
96  self.initDone = true
97
98  -- prepare fields
99  local field = g_fieldManager:getFieldByIndex(24)
100 local fruitDesc = g_fruitTypeManager:getFruitTypeByIndex(FruitType.WHEAT
101 for i = 1,table.getn(field.maxFieldStatusPartitions) do
102 g_fieldManager:setFieldPartitionStatus(field,
    field.maxFieldStatusPartitions, i, fruitDesc.index,
    FieldManager.FIELDSTATE_GROWING, fruitDesc.maxHarvestingGrowthState, 3,
    true, g_currentMission.plowCounterMaxValue, 0,
    g_currentMission.limeCounterMaxValue)
103 end
104
105 local field = g_fieldManager:getFieldByIndex(25)
106 local fruitDesc = g_fruitTypeManager:getFruitTypeByIndex(FruitType.CANOI
107 for i = 1,table.getn(field.maxFieldStatusPartitions) do
108 g_fieldManager:setFieldPartitionStatus(field,
    field.maxFieldStatusPartitions, i, fruitDesc.index,
    FieldManager.FIELDSTATE_HARVESTED, 0, 0, false,
    g_currentMission.plowCounterMaxValue, 0,
    g_currentMission.limeCounterMaxValue)
109 end
110
111 local field = g_fieldManager:getFieldByIndex(26)
112 for i = 1,table.getn(field.maxFieldStatusPartitions) do
113 g_fieldManager:setFieldPartitionStatus(field,
    field.maxFieldStatusPartitions, i, nil, FieldManager.FIELDSTATE_CULTIVAT
    0, 0, false, g_currentMission.plowCounterMaxValue, 0,
    g_currentMission.limeCounterMaxValue)
114 end
115
116 local field = g_fieldManager:getFieldByIndex(19)
```

```
117 for i = 1,table.getn(field.maxFieldStatusPartitions) do
118 g_fieldManager:setFieldPartitionStatus(field,
    field.maxFieldStatusPartitions, i, nil, FieldManager.FIELDSTATE_CULTIVAT
    0, 0, false, g_currentMission.plowCounterMaxValue, 0,
    g_currentMission.limeCounterMaxValue)
119 end
120 end
121
122 if self.startTourDialog then
123 self.startTourDialogDelay = self.startTourDialogDelay - dt
124 local showDialog = true
125 if g_currentMission.cameraFlightManager ~= nil then
126 showDialog = g_currentMission.cameraFlightManager.careerStartFlightPlaye
127 end
128 if showDialog then
129 if self.startTourDialogDelay < 0 then
130 self.startTourDialog = false
131 self:showTourDialog()
132 end
133 end
134 end
135
136 if self.isPaused then
137 if self.pauseTime > 0 then
138 self.pauseTime = self.pauseTime - dt
139 else
140 self.pauseTime = 0
141 self.isPaused = false
142 self:activateNextIcon()
143 end
144 end
145
146 if self.visible and not self.isPaused then
147 -- show current permanent message on screen if no ingame message or any
    other GUI screen is displayed
148 if not g_currentMission.hud:isInGameMessageVisible() and not
    g_gui:getIsGuiVisible() then
149
150 if false then --g_i18n:hasText("tour_permanentText" ..
    self.currentTourIconNumber - 1) then
151
152 if self.permanentMessageDelay > 0 then
```

```
153 self.permanentMessageDelay = self.permanentMessageDelay - dt
154 self.alpha = 0.25
155 self.alphaDirection = 1
156 else
157
158 setTextColor(1, 1, 1, self.alpha)
159 setTextBold(true)
160 setTextAlignment(RenderText.ALIGN_CENTER)
161 setTextWrapWidth(0.35)
162 if GS_IS_CONSOLE_VERSION then
163 setTextWrapWidth(0.295)
164 end
165 renderText(0.5, 0.93, self.permanentTextSize,
    g_i18n:getText("tour_permanentText" .. self.currentTourIconNumber - 1))
166 setTextWrapWidth(0)
167 setTextAlignment(RenderText.ALIGN_LEFT)
168 setTextBold(false)
169 setTextColor(1, 1, 1, 1)
170
171 self.alpha = self.alpha + self.alphaDirection * (dt / 600)
172 if self.alpha > 1 or self.alpha < 0.25 then
173 self.alphaDirection = self.alphaDirection * -1
174 self.alpha = MathUtil.clamp(self.alpha, 0.25, 1)
175 end
176 end
177
178 end
179 end
180
181 -- handle special cases
182
183 --# harvesting
184
185 -- icon #3 activates as soon as the player enters the tour's combine
    harvester
186 if self.currentTourIconNumber <= 3 then
187 if g_currentMission.controlledVehicle ~= nil and
    g_currentMission.controlledVehicle ==
    g_currentMission.tourVehicles["tourCombine"] then
188 self.currentTourIconNumber = 3
189 -- self:activateNextIcon()
```

```
190 self.pauseTime = 1000
191 self.isPaused = true
192 end
193 end
194
195 -- wait for player to attach the cutter
196 if self.currentTourIconNumber == 4 then
197 if g_currentMission.controlledVehicle ~= nil and
    g_currentMission.controlledVehicle ==
    g_currentMission.tourVehicles["tourCombine"] then
198 if
    g_currentMission.tourVehicles["tourCombine"].spec_combine.numAttachedCut
    > 0 then
199 -- self:activateNextIcon()
200 self.pauseTime = 1000
201 self.isPaused = true
202 end
203 end
204 end
205
206 -- wait for player to activate the cutter
207 if self.currentTourIconNumber == 5 then
208 if g_currentMission.controlledVehicle ~= nil and
    g_currentMission.controlledVehicle ==
    g_currentMission.tourVehicles["tourCombine"] then
209 if g_currentMission.controlledVehicle:getIsTurnedOn() then
210 -- self:activateNextIcon()
211 self.pauseTime = 1000
212 self.isPaused = true
213 end
214 end
215 end
216
217 -- wait for player to activate the helper
218 if self.currentTourIconNumber == 7 then
219 if g_currentMission.tourVehicles["tourCombine"]:getIsTurnedOn() and
    g_currentMission.tourVehicles["tourCombine"]:getIsAIActive() then
220 -- self:activateNextIcon()
221 self.pauseTime = 1000
222 self.isPaused = true
223 end
224 end
```

```
225
226 --# cultivating
227
228 -- wait for player to enter tractor1 and attach cultivator
229 if self.currentTourIconNumber == 9 then
230 if g_currentMission.controlledVehicle ~= nil and
    g_currentMission.controlledVehicle ==
    g_currentMission.tourVehicles["tourTractor1"] then
231 if g_currentMission.tourVehicles["tourCultivator"]:getRootVehicle() ==
    g_currentMission.tourVehicles["tourTractor1"]
232 and g_currentMission.tourVehicles["tourWeight1"]:getRootVehicle() ==
    g_currentMission.tourVehicles["tourTractor1"] then
233 -- self:activateNextIcon()
234 self.pauseTime = 1000
235 self.isPaused = true
236 end
237 end
238 end
239
240 if self.currentTourIconNumber == 11 then
241 -- self:activateNextIcon()
242 self.pauseTime = 1000
243 self.isPaused = true
244 end
245
246 --# sowing
247
248 -- wait for player to enter tractor2 and attach sowingMachine
249 if self.currentTourIconNumber == 13 then
250 if g_currentMission.controlledVehicle ~= nil and
    g_currentMission.controlledVehicle ==
    g_currentMission.tourVehicles["tourTractor2"] then
251 if g_currentMission.tourVehicles["tourSowingMachine"]:getRootVehicle() =
    g_currentMission.tourVehicles["tourTractor2"]
252 and g_currentMission.tourVehicles["tourWeight2"]:getRootVehicle() ==
    g_currentMission.tourVehicles["tourTractor2"] then
253 -- self:activateNextIcon()
254 self.pauseTime = 1000
255 self.isPaused = true
256 end
257 end
258 end
```

```
259
260
261 --# overloading / tipping
262
263 -- wait for player to enter tractor3 and attach trailer
264 if self.currentTourIconNumber == 16 then
265 if g_currentMission.controlledVehicle ~= nil and
    g_currentMission.controlledVehicle ==
    g_currentMission.tourVehicles["tourTractor3"] then
266 if g_currentMission.tourVehicles["tourTrailer"]:getRootVehicle() ==
    g_currentMission.tourVehicles["tourTractor3"] then
267 -- self:activateNextIcon()
268 self.pauseTime = 1000
269 self.isPaused = true
270 end
271 end
272 end
273
274 if self.currentTourIconNumber == 17 then
275 if g_currentMission.controlledVehicle ~= nil and
    g_currentMission.controlledVehicle ==
    g_currentMission.tourVehicles["tourTractor3"] then
276 if g_currentMission.tourVehicles["tourTrailer"]:getRootVehicle() ==
    g_currentMission.tourVehicles["tourTractor3"] then
277
278 local tractor = g_currentMission.tourVehicles["tourTractor3"]
279 local trailer = g_currentMission.tourVehicles["tourTrailer"]
280 local combine = g_currentMission.tourVehicles["tourCombine"]
281
282 local x,y,z = localToWorld(combine.components[1].node, 6,0,0)
283 self.mapHotspot:setWorldPosition(x, z)
284
285 local xv,yv,zv = getWorldTranslation(tractor.components[1].node)
286 local dist = MathUtil.vector3Length(x-xv, y-yv, z-zv)
287 if dist < 10 then
288 g_currentMission:setMapTargetHotspot(nil)
289 else
290 g_currentMission:setMapTargetHotspot(self.mapHotspot)
291 end
292
293 local fillUnitIndex = 1
294 local dischargeNode = combine:getCurrentDischargeNode()
```

```
295 if trailer:getFillUnitFillLevel(fillUnitIndex) > 0 and dischargeNode ~=
    and dischargeNode.dischargeObject == trailer then
296 -- self:activateNextIcon()
297 self.pauseTime = 1000
298 self.isPaused = true
299 end
300 end
301 else
302 -- If the harvester is full or stopped, and a player goes into the harve
    to open the pipe, the trailer vehicle
303 -- is not controlled so above code does not trigger. This cascades and
    breaks the tour.
304 if g_currentMission.tourVehicles["tourTrailer"]:getFillUnitFillLevel(1)
    400 then
305 self.pauseTime = 1000
306 self.isPaused = true
307 end
308 end
309 end
310
311 if self.currentTourIconNumber == 19 then
312 -- self:activateNextIcon()
313 self.pauseTime = 3000
314 self.isPaused = true
315 end
316
317 if self.currentTourIconNumber == 20 then
318 if g_currentMission.controlledVehicle ~= nil and
    g_currentMission.controlledVehicle ==
    g_currentMission.tourVehicles["tourTractor3"] then
319 if g_currentMission.tourVehicles["tourTrailer"]:getRootVehicle() ==
    g_currentMission.tourVehicles["tourTractor3"] then
320 local trailer = g_currentMission.tourVehicles["tourTrailer"]
321
322 local unloadingStation
323 for _, station in pairs(g_currentMission.unloadingStations) do
324 if station.owningPlaceable ~= nil and station.owningPlaceable.mapBoundId
    "sellingStationRestaurant" then
325 unloadingStation = station
326 break
327 end
328 end
```

```
329
330 local dischargeNode = trailer:getCurrentDischargeNode()
331
332 if dischargeNode ~= nil and dischargeNode.currentDischargeObject ~= nil
    dischargeNode.currentDischargeObject:getTarget() == unloadingStation the
333 -- self:activateNextIcon()
334 self.pauseTime = 3000
335 self.isPaused = true
336 end
337 end
338 end
339 end
340
341 --# shop
342 if self.currentTourIconNumber == 22 then
343 -- self:activateNextIcon()
344 self.pauseTime = 1000
345 self.isPaused = true
346 end
347
348 end
349 end
```

makeIconVisible

Description

Make tour icon visable

Definition

makeIconVisible(integer tourIconId)

Arguments

integer tourIconId id of tour icon

Code

```
354 function TourIcons:makeIconVisible(tourIconId)
355 -- make next icon visible
356 setVisibility(tourIconId, true)
357 local x, _, z = getWorldTranslation(tourIconId)
358
359 if self.mapHotspot == nil then
360 self.mapHotspot = MapHotspot:new("guide",
    MapHotspot.CATEGORY_TOUR)
361 self.mapHotspot:setImage(nil,
    getNormalizedUVs(MapHotspot.UV.HIGHLIGHT_MARKER), {0.2705, 0.6514,
    0.0802, 1})
362 self.mapHotspot:setPersistent(true)
```

```
363 self.mapHotspot:setRenderLast(true)
364 self.mapHotspot:setBlinking(true)
365
366 g_currentMission:addMapHotspot(self.mapHotspot)
367 end
368
369 self.mapHotspot:setWorldPosition(x, z)
370
371
372 -- Find 'hidden' icon used internally only
373 local x,y,z = getWorldTranslation(tourIconId)
374 local h =
    getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode,
    x,y,z)
375 if y > h then
376 g_currentMission:setMapTargetHotspot(self.mapHotspot)
377 self.mapHotspot.enabled = true
378 else
379 g_currentMission:setMapTargetHotspot(nil)
380 self.mapHotspot.enabled = false
381 end
382
383 g_currentMission.hud.ingameMap:toggleSize(IngameMap.STATE_MINIMAP,
    true)
384 end
```

triggerCallback

Description

Trigger callback

Definition

triggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

Code

```
393 function TourIcons:triggerCallback(triggerId, otherId, onEnter,
    onLeave, onStay)
394 if onEnter then
395 if self.tourIcons[self.currentTourIconNumber] ~= nil then
```

```
396 if self.tourIcons[self.currentTourIconNumber].tourIconTriggerId == triggerId and getVisibility(self.tourIcons[self.currentTourIconNumber].tourIconId) then
397 self:activateNextIcon()
398 end
399 end
400 end
401 end
```

activateNextIcon

Description

Activate next icon

Definition

activateNextIcon()

Code

```
405 function TourIcons:activateNextIcon()
406
407 -- make all previous icons invisible (also handles cases where player ma
    icons)
408 for i = 1, self.currentTourIconNumber do
409 local tourIcon = self.tourIcons[i]
410 if getVisibility(tourIcon.tourIconId) then
411 setVisibility(tourIcon.tourIconId, false)
412 setCollisionMask(tourIcon.tourIconTriggerId, 0)
413 end
414 end
415
416 if self.tourIcons[self.currentTourIconNumber + 1] ~= nil then
417 self:makeIconVisible(self.tourIcons[self.currentTourIconNumber + 1].tour
418 else
419 -- end of tour!
420 if self.mapHotspot ~= nil then
421 g_currentMission:removeMapHotspot(self.mapHotspot)
422 self.mapHotspot:delete()
423 self.mapHotspot = nil
424 end
425 -- re-display non-tour help icons
426 if g_gameSettings:getValue("showHelpIcons") then
427 if g_currentMission.helpIconsBase ~= nil then
428 g_currentMission.helpIconsBase:showHelpIcons(true, true)
429 end
```

```
430 end
431
432 self.visible = false
433 end
434
435 local title = g_i18n:getText("ui_tour")
436 local text = ""
437 local controls = {}
438
439 if self.currentTourIconNumber == 1 then
440
441 text = g_i18n:getText("tour_text_part01_lookAndWalk")
442 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.TOGGLE_MAP
    g_i18n:getText("action_toggleMapView")))
443
444 local useGamepadButtons = g_inputBinding:getInputHelpMode() == GS_INPUT_
445 if useGamepadButtons then
446 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.AXIS_MOVE_
    InputAction.AXIS_MOVE_SIDE_PLAYER, g_i18n:getText("action_movePlayer")))
447 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.AXIS_LOOK_
    InputAction.AXIS_LOOK_LEFTRIGHT_PLAYER, g_i18n:getText("action_lookPlaye
448 else
449 -- special case on PC
450 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.AXIS_MOVE_
    InputAction.AXIS_MOVE_SIDE_PLAYER, g_i18n:getText("action_movePlayer")))
451
452 -- to show a mouse icon, instead of 'arrow keys' on keyboard
453 -- local lookAroundSymbol =
    g_inputDisplayManager.controllerSymbols["mouse_MOUSE_BUTTON_NONE"] -- TO
    access, too error prone
454 -- if lookAroundSymbol ~= nil then
455 -- table.insert(controls, {overlays = {lookAroundSymbol.overlay}, text =
    g_i18n:getText("action_lookPlayer")} )
456 -- end
457 end
458
459 elseif self.currentTourIconNumber == 2 then -- # harvesting
460
461 text = g_i18n:getText("tour_text_part01_enterCombine")
```

```
462 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.ENTER, nil
    g_i18n:getText("input_ENTER")))
463
464 elseif self.currentTourIconNumber == 3 then
465
466 text = g_i18n:getText("tour_text_part01_attachHeader")
467 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.ATTACH, ni
    g_i18n:getText("input_ATTACH")))
468
469 elseif self.currentTourIconNumber == 4 then
470
471 text = g_i18n:getText("tour_text_part01_turnOnCombine")
472 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.SWITCH_IMP
    g_i18n:getText("action_switchImplement")))
473 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.IMPLEMENT_
    g_i18n:getText("action_unfold")))
474 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.IMPLEMENT_
    g_i18n:getText("action_turnOn")))
475
476 elseif self.currentTourIconNumber == 5 then
477
478 text = g_i18n:getText("tour_text_part01_startHarvesting")
479 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.AXIS_ACCEL
    InputAction.AXIS_BRAKE_VEHICLE, g_i18n:getText("action_accelerate")))
480 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.AXIS_MOVE_
    g_i18n:getText("action_steer")))
481
482 elseif self.currentTourIconNumber == 6 then
483
484 text = g_i18n:getText("tour_text_part01_startHelperHarvesting")
485 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.TOGGLE_AI,
    g_i18n:getText("input_TOGGLE_AI")))
486
487 elseif self.currentTourIconNumber == 7 then
488
489 text = g_i18n:getText("tour_text_part01_finished")
```

```
490 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.ENTER, nil
    g_i18n:getText("action_exitVehicle")))
491
492 elseif self.currentTourIconNumber == 8 then -- # cultivating
493
494 text = g_i18n:getText("tour_text_part02_enterTractor01")
495 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.ENTER, nil
    g_i18n:getText("input_ENTER")))
496 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.ATTACH, ni
    g_i18n:getText("input_ATTACH")))
497
498 elseif self.currentTourIconNumber == 9 then
499
500 text = g_i18n:getText("tour_text_part02_startCultivating")
501 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.LOWER_IMPI
    g_i18n:getText("input_LOWER_IMPLEMENT")))
502
503 elseif self.currentTourIconNumber == 10 then
504
505 text = g_i18n:getText("tour_text_part02_enoughCultivating")
506 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.TOGGLE_AI,
    g_i18n:getText("input_TOGGLE_AI")))
507
508 elseif self.currentTourIconNumber == 11 then
509
510 text = g_i18n:getText("tour_text_part02_finished")
511 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.SWITCH_VEH
    g_i18n:getText("input_SWITCH_VEHICLE")))
512 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.SWITCH_VEH
    g_i18n:getText("input_SWITCH_VEHICLE_BACK")))
513
514 elseif self.currentTourIconNumber == 12 then -- # sowing
515
516 text = g_i18n:getText("tour_text_part03_enterTractor01")
517 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.ENTER, nil
    g_i18n:getText("input_ENTER")))
```

```
518 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.ATTACH, ni
    g_i18n:getText("input_ATTACH")))
519
520 elseif self.currentTourIconNumber == 13 then
521
522 text = g_i18n:getText("tour_text_part03_startSowing")
523 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.IMPLEMENT_
    g_i18n:getText("action_chooseSeed")))
524 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.LOWER_IMPI
    g_i18n:getText("input_LOWER_IMPLEMENT")))
525 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.IMPLEMENT_
    g_i18n:getText("action_turnOn")))
526
527 elseif self.currentTourIconNumber == 14 then
528
529 text = g_i18n:getText("tour_text_part03_finished")
530 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.SWITCH_VEH
    g_i18n:getText("input_SWITCH_VEHICLE")))
531 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.SWITCH_VEH
    g_i18n:getText("input_SWITCH_VEHICLE_BACK")))
532
533 elseif self.currentTourIconNumber == 15 then -- # trailer / tipping and
534
535 text = g_i18n:getText("tour_text_part04_enterTractor01")
536 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.ENTER, nil
    g_i18n:getText("input_ENTER")))
537 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.ATTACH, ni
    g_i18n:getText("input_ATTACH")))
538
539 elseif self.currentTourIconNumber == 16 then
540
541 text = g_i18n:getText("tour_text_part04_alignToHarvester")
542
543 elseif self.currentTourIconNumber == 17 then
544
545 text = g_i18n:getText("tour_text_part04_unloadWheat")
```

```
546
547 elseif self.currentTourIconNumber == 18 then
548
549 text = g_i18n:getText("tour_text_part04_pricesInfo")
550 table.insert(controls, g_inputDisplayManager:getControllerSymbolOverlays
    nil, g_i18n:getText("input_MENU")))
551
552 elseif self.currentTourIconNumber == 19 then
553
554 text = g_i18n:getText("tour_text_part04_sellWheat")
555 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.TOGGLE_TIP
    g_i18n:getText("input_TOGGLE_TIPSTATE")))
556
557 elseif self.currentTourIconNumber == 20 then
558
559 text = g_i18n:getText("tour_text_part04_doneSelling")
560
561 elseif self.currentTourIconNumber == 21 then -- # shop
562
563 text = g_i18n:getText("tour_text_part05_visitShop")
564 table.insert(controls,
    g_inputDisplayManager:getControllerSymbolOverlays(InputAction.TOGGLE_STO
    g_i18n:getText("input_TOGGLE_STORE")))
565
566 elseif self.currentTourIconNumber == 22 then
567
568 text = g_i18n:getText("tour_text_end")
569
570 end
571
572 if g_currentMission.controlledVehicle ~= nil and
    g_currentMission.controlledVehicle.setCruiseControlState ~= nil then
573 g_currentMission.controlledVehicle:setCruiseControlState(Drivable.CRUISE
574 end
575
576 g_currentMission.hud:showInGameMessage(title, text, -1, controls)
577
578
579 self.currentTourIconNumber = self.currentTourIconNumber + 1
580
581 self.permanentMessageDelay = 250
```

```
582  end
```

UnloadingStation
Description

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID
table connection connection

Code

```
 99  function UnloadingStation:readStream(streamId, connection)
100  UnloadingStation:superClass().readStream(self, streamId,
     connection)
101  if connection:getIsServer() then
102  for _, unloadTrigger in ipairs(self.unloadTriggers) do
103  local unloadTriggerId = NetworkUtil.readNodeObjectId(streamId)
104  unloadTrigger:readStream(streamId, connection)
105  g_client:finishRegisterObject(unloadTrigger, unloadTriggerId)
106  end
107  end
108  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID
table connection connection

Code

```
114  function UnloadingStation:writeStream(streamId, connection)
115  UnloadingStation:superClass().writeStream(self, streamId,
     connection)
116  if not connection:getIsServer() then
117  for _, unloadTrigger in ipairs(self.unloadTriggers) do
118  NetworkUtil.writeNodeObjectId(streamId,
     NetworkUtil.getObjectId(unloadTrigger))
119  unloadTrigger:writeStream(streamId, connection)
120  g_server:registerObjectInStream(connection, unloadTrigger)
121  end
122  end
```

```
123 end
```

VehicleSellingPoint

Description

Class for vehicle selling point

new

Description

Creating vehicle selling point

Definition

new(integer id)

Arguments

integer id node id

Return Values

table instance Instance of object

Code

```
16 function VehicleSellingPoint:new(id)
17 local self = {}
18 setmetatable(self, VehicleSellingPoint_mt)
19
20 self.id = id
21
22 self.vehicleInRange = {}
23 self.currentVehicle = nil
24 self.currentVehicleId = 0
25
26 self.activateText =
   g_i18n:getText("action_configSellSpecificVehicle")
27
28 self.isEnabled = true
29 self.objectActivated = false
30
31 return self
32 end
```

delete

Description

Deleting vehicle selling point

Definition

delete()

Code

```
50 function VehicleSellingPoint:delete()
51 g_messageCenter:unsubscribeAll(self)
52
53 if self.playerTrigger ~= nil then
```

```
54  removeTrigger(self.playerTrigger)
55  self.playerTrigger = nil
56  end
57  if self.sellTriggerNode ~= nil then
58  removeTrigger(self.sellTriggerNode)
59  self.sellTriggerNode = nil
60  end
61
62  self.sellIcon = nil
63  end
```

getIsActivatable

Description

Get is activateable

Definition

getIsActivatable()

Return Values

boolean isActivateable is activateable

Code

```
68  function VehicleSellingPoint:getIsActivatable()
69  return self.isEnabled and g_currentMission.controlPlayer and
    (self:getOwnerFarmId() == 0 or g_currentMission:getFarmId() ==
    self:getOwnerFarmId())
70  end
```

onActivateObject

Description

On activate object

Definition

onActivateObject()

Code

```
78  function VehicleSellingPoint:onActivateObject()
79  self:determineCurrentVehicle()
80  g_gui:showDirectSellDialog({vehicle=self.currentVehicle,
    owner=self, ownWorkshop=self.ownWorkshop})
81  end
```

update

Description

Update

Definition

update(float dt)

Arguments

float dt time since last call in ms

Code

```
86  function VehicleSellingPoint:update(dt)
87  end
```

triggerCallback

Description

Trigger callback

Definition

triggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

Code

```
96  function VehicleSellingPoint:triggerCallback(triggerId, otherId, onEnter, onLeave, onStay)
97  if self.isEnabled and (not g_isPresentationVersion or g_isPresentationVersionShopEnabled) and g_currentMission.missionInfo:isa(FSCareerMissionInfo) then
98  if onEnter or onLeave then
99  if g_currentMission.player ~= nil and otherId == g_currentMission.player.rootNode then
100 if onEnter then
101 if not self.objectActivated then
102 g_currentMission:addActivatableObject(self)
103 self.objectActivated = true
104 end
105 else
106 if self.objectActivated then
107 g_currentMission:removeActivatableObject(self)
108 self.objectActivated = false
109 end
110 end
111 end
112 end
113 end
114 end
```

sellAreaTriggerCallback

Description

Sell area trigger callback

Definition

sellAreaTriggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

Code

```
123 function VehicleSellingPoint:sellAreaTriggerCallback(triggerId, otherId, onEnter, onLeave, onStay, otherShapeId)
124 if otherShapeId ~= nil and (onEnter or onLeave) then
125 if onEnter then
126 self.vehicleInRange[otherShapeId] = true
127 elseif onLeave then
128 self.vehicleInRange[otherShapeId] = nil
129 end
130
131 self:determineCurrentVehicle()
132 end
133 end
```

determineCurrentVehicle

Description

Determine current vehicle

Definition

determineCurrentVehicle()

Code

```
137 function VehicleSellingPoint:determineCurrentVehicle()
138 self.currentVehicle = nil
139 for vehicleId, inRange in pairs(self.vehicleInRange) do
140 if inRange ~= nil then
141 self.currentVehicle = g_currentMission.nodeToObject[vehicleId]
142
143 if self.currentVehicle ~= nil then
144 if not SpecializationUtil.hasSpecialization(Rideable, self.currentVehicle.specializations) or self.currentVehicle:getOwnerFarmId() ~= self:getOwnerFarmId() then
145 break
146 end
147 end
148 end
```

```
149  -- invalid vehicle (not existing or left), remove from list
150  self.vehicleInRange[vehicleId] = nil
151  end
152  end
```

updateIconVisibility

Description

Turn the icon on or off depending on the current game and the players farm

Definition

updateIconVisibility()

VendingMachine

Description

onActivate

Description

Event function for activation input.

Definition

onActivate()

BeehivePlaceable

Description

Class for placeable beehives

Parent

Placeable

new

Description

Creating placeable beehive

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean	isServer	is server
boolean	isClient	is client
table	customMt	custom metatable

Return Values

table	instance	Instance of object

Code

```
18  function BeehivePlaceable:new(isServer, isClient, customMt)
19  local mt = customMt
20  if mt == nil then
21  mt = BeehivePlaceable_mt
22  end
23
24  local self = Placeable:new(isServer, isClient, mt)
25
26  registerObjectClassName(self, "BeehivePlaceable")
```

```
27
28  return self
29  end
```

delete

Description

Deleting placeable beehive

Definition

delete()

Code

```
33  function BeehivePlaceable:delete()
34  unregisterObjectClassName(self)
35  g_currentMission.environment:removeHourChangeListener(self)
36  if self.particleSystem ~= nil then
37  ParticleUtil.deleteParticleSystem(self.particleSystem)
38  end
39  BeehivePlaceable:superClass().delete(self)
40  end
```

deleteFinal

Description

Deleting placeable beehive final

Definition

deleteFinal()

Code

```
44  function BeehivePlaceable:deleteFinal()
45  BeehivePlaceable:superClass().deleteFinal(self)
46  end
```

load

Description

Load beehive

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position
float	y	z world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Return Values

boolean success success

Code

```
59  function BeehivePlaceable:load(xmlFilename, x,y,z, rx,ry,rz, initRandom)
60  if not BeehivePlaceable:superClass().load(self, xmlFilename, x,y,z, rx,ry,rz, initRandom) then
61  return false
62  end
63
64  local xmlFile = loadXMLFile("TempXML", xmlFilename)
65
66  self.particleSystem = {}
67  ParticleUtil.loadParticleSystem(xmlFile, self.particleSystem, "placeable.particleSystem", self.nodeId, true, nil, self.baseDirectory)
68
69  delete(xmlFile)
70
71  return true
72  end
```

finalizePlacement

Description

Called if placeable is placed

Definition

finalizePlacement()

Code

```
76  function BeehivePlaceable:finalizePlacement()
77  BeehivePlaceable:superClass().finalizePlacement(self)
78  end
```

hourChanged

Description

Called if hour changed

Definition

hourChanged()

Code

```
82  function BeehivePlaceable:hourChanged()
83  if self.isServer then
84  g_currentMission:addMoney(self.incomePerHour, self:getOwnerFarmId(), "propertyIncome")
85  g_currentMission:addMoneyChange(self.incomePerHour, self:getOwnerFarmId(), EconomyManager.MONEY_TYPE_PROPERTY_INCOME)
86  end
```

```
87  end
```

BgaPlaceable

Description

Class for bga

Parent

Placeable

new

Description

Creating placeable silo

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean isServer	is server
boolean isClient	is client
table	customMt custom metatable

Return Values

table instance Instance of object

Code

```
24  function BgaPlaceable:new(isServer, isClient, customMt)
25  local self = Placeable:new(isServer, isClient, customMt or
    BgaPlaceable_mt)
26
27  registerObjectClassName(self, "BgaPlaceable")
28
29  return self
30  end
```

delete

Description

Deleting placeable silo

Definition

delete()

Code

```
34  function BgaPlaceable:delete()
35  self.bga:delete()
36
37  for _, bunkerSilo in ipairs(self.bunkerSilos) do
38  bunkerSilo:delete()
39  end
40
41  g_farmlandManager:removeStateChangeListener(self)
42
43  unregisterObjectClassName(self)
```

```
44  BgaPlaceable:superClass().delete(self)
45  end
```

load

Description

Load silo

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position
float	y	z world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Return Values

boolean success success

Code

```
58  function BgaPlaceable:load(xmlFilename, x,y,z, rx,ry,rz, initRandom)
59  if not BgaPlaceable:superClass().load(self, xmlFilename, x,y,z, rx,ry,rz, initRandom) then
60  return false
61  end
62
63  local xmlFile = loadXMLFile("TempXML", xmlFilename)
64
65  self.bga = Bga:new(self.isServer, self.isClient)
66  if not self.bga:load(self.nodeId, xmlFile, "placeable.bga") then
67  self.bga:delete()
68  delete(xmlFile)
69  return false
70  end
71
72  self.bunkerSilos = {}
73
74  local i = 0
75  while true do
76  local bunkerKey = string.format("placeable.bunkerSilos.bunkerSilo(%d)", i)
77  if not hasXMLProperty(xmlFile, bunkerKey) then
```

```
78  break
79  end
80
81  local bunkerSilo = BunkerSilo:new(self.isServer, self.isClient)
82  if bunkerSilo:load(self.nodeId, xmlFile, bunkerKey) then
83  table.insert(self.bunkerSilos, bunkerSilo)
84  else
85  bunkerSilo:delete()
86  end
87
88  i = i + 1
89  end
90
91  delete(xmlFile)
92
93  return true
94  end
```

finalizePlacement

Description

Called if placeable is placed

Definition

finalizePlacement()

Code

```
98  function BgaPlaceable:finalizePlacement()
99  BgaPlaceable:superClass().finalizePlacement(self)
100
101 self.bga:register(true)
102
103 for _, bunkerSilo in ipairs(self.bunkerSilos) do
104 bunkerSilo:register(true)
105 end
106
107 if self.isServer then
108 self:updateOwnership(true)
109 end
110
111 g_farmlandManager:addStateChangeListener(self)
112 end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
118 function BgaPlaceable:readStream(streamId, connection)
119 BgaPlaceable:superClass().readStream(self, streamId, connection)
120 if connection:getIsServer() then
121 local bgaId = NetworkUtil.readNodeObjectId(streamId)
122 self.bga:readStream(streamId, connection)
123 g_client:finishRegisterObject(self.bga, bgaId)
124
125 for _, bunkerSilo in ipairs(self.bunkerSilos) do
126 local bunkerSiloId = NetworkUtil.readNodeObjectId(streamId)
127 bunkerSilo:readStream(streamId, connection)
128 g_client:finishRegisterObject(bunkerSilo, bunkerSiloId)
129 end
130 end
131 end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
137 function BgaPlaceable:writeStream(streamId, connection)
138 BgaPlaceable:superClass().writeStream(self, streamId, connection)
139 if not connection:getIsServer() then
140 NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(self.bga))
141 self.bga:writeStream(streamId, connection)
142 g_server:registerObjectInStream(connection, self.bga)
143
144 for _, bunkerSilo in ipairs(self.bunkerSilos) do
145 NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(bunkerSilo))
146 bunkerSilo:writeStream(streamId, connection)
147 g_server:registerObjectInStream(connection, bunkerSilo)
```

```
148 end
149 end
150 end
```

collectPickObjects

Description

Collect pick objects

Definition

collectPickObjects(integer node)

Arguments

integer node node id

Code

```
167 function BgaPlaceable:collectPickObjects(node)
168 local foundNode = false
169 for _, unloadTrigger in
    ipairs(self.bga.bunker.unloadingStation.unloadTriggers) do
170 if node == unloadTrigger.exactFillRootNode then
171 foundNode = true
172 break
173 end
174 end
175
176 if not foundNode then
177 for _, loadTrigger in
    ipairs(self.bga.digestateSilo.loadingStation.loadTriggers) do
178 if node == loadTrigger.triggerNode then
179 foundNode = true
180 break
181 end
182 end
183 end
184
185 if not foundNode then
186 BgaPlaceable:superClass().collectPickObjects(self, node)
187 end
188 end
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key, boolean resetVehicles)

Arguments

integer	xmlFile	id of xml object

string key key
boolean resetVehicles reset vehicles

Return Values

boolean success success

Code

```
196 function BgaPlaceable:loadFromXMLFile(xmlFile, key, resetVehicles)
197 if not BgaPlaceable:superClass().loadFromXMLFile(self, xmlFile, key, resetVehicles) then
198 return false
199 end
200
201 self.bga:loadFromXMLFile(xmlFile, key..".bga", resetVehicles)
202
203 local i = 0
204 while true do
205 local bunkerKey = string.format("%s.bunkerSilo(%d)", key, i)
206 if not hasXMLProperty(xmlFile, bunkerKey) then
207 break
208 end
209
210 local index = getXMLInt(xmlFile, bunkerKey.."#index")
211
212 if index ~= nil then
213 if self.bunkerSilos[index] ~= nil then
214 if not self.bunkerSilos[index]:loadFromXMLFile(xmlFile, bunkerKey) then
215 return false
216 end
217 else
218 g_logManager:warning("Could not load bunkersilo. Given 'index' '%d' is not defined!", index)
219 end
220 end
221
222 i = i + 1
223 end
224
225 return true
226 end
```

saveToXMLFile

Description

Get save attributes and nodes

Definition

saveToXMLFile(string nodeIdent)

Arguments

string nodeIdent node ident

Return Values

string attributes attributes

string nodes nodes

Code

```
233 function BgaPlaceable:saveToXMLFile(xmlFile, key, usedModNames)
234 BgaPlaceable:superClass().saveToXMLFile(self, xmlFile, key, usedModNames)
235
236 self.bga:saveToXMLFile(xmlFile, key..".bga", usedModNames)
237
238 for k, bunkerSilo in ipairs(self.bunkerSilos) do
239 local bunkerKey = string.format("%s.bunkerSilo(%d)", key, k-1)
240 setXMLInt(xmlFile, bunkerKey .. "#index", k)
241 bunkerSilo:saveToXMLFile(xmlFile, bunkerKey, usedModNames)
242 end
243 end
```

BunkerSiloPlaceable

Description

new

Description

Creating placeable silo

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean isServer is server

boolean isClient is client

table customMt custom metatable

Return Values

table instance Instance of object

Code

```
26 function BunkerSiloPlaceable:new(isServer, isClient, customMt)
27 local self = Placeable:new(isServer, isClient, customMt or BgaPlaceable_mt)
28
29 registerObjectClassName(self, "BunkerSiloPlaceable")
30
31 return self
32 end
```

delete

Description

Deleting placeable silo

Definition

delete()

Code

```
36  function BunkerSiloPlaceable:delete()
37  for _, bunkerSilo in ipairs(self.bunkerSilos) do
38  bunkerSilo:delete()
39  end
40
41  unregisterObjectClassName(self)
42  BunkerSiloPlaceable:superClass().delete(self)
43  end
```

load

Description

Load silo

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position
float	y	z world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Return Values

boolean success success

Code

```
56  function BunkerSiloPlaceable:load(xmlFilename, x,y,z, rx,ry,rz,
    initRandom)
57  if not BunkerSiloPlaceable:superClass().load(self, xmlFilename,
    x,y,z, rx,ry,rz, initRandom) then
58  return false
59  end
60
61  local xmlFile = loadXMLFile("TempXML", xmlFilename)
62
63  self.bunkerSilos = {}
64
```

```lua
local i = 0
while true do
local bunkerKey = string.format("placeable.bunkerSilos.bunkerSilo(%d)", i)
if not hasXMLProperty(xmlFile, bunkerKey) then
break
end

local bunkerSilo = BunkerSilo:new(self.isServer, self.isClient)
if bunkerSilo:load(self.nodeId, xmlFile, bunkerKey) then
table.insert(self.bunkerSilos, bunkerSilo)
else
bunkerSilo:delete()
end

i = i + 1
end

delete(xmlFile)

return true
end
```

finalizePlacement

Description

Called if placeable is placed

Definition

finalizePlacement()

Code

```lua
function BunkerSiloPlaceable:finalizePlacement()
BunkerSiloPlaceable:superClass().finalizePlacement(self)

for _, bunkerSilo in ipairs(self.bunkerSilos) do
bunkerSilo:register(true)
end
end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
109 function BunkerSiloPlaceable:readStream(streamId, connection)
110 BunkerSiloPlaceable:superClass().readStream(self, streamId,
    connection)
111 if connection:getIsServer() then
112 for _, bunkerSilo in ipairs(self.bunkerSilos) do
113 local bunkerSiloId = NetworkUtil.readNodeObjectId(streamId)
114 bunkerSilo:readStream(streamId, connection)
115 g_client:finishRegisterObject(bunkerSilo, bunkerSiloId)
116 end
117 end
118 end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
124 function BunkerSiloPlaceable:writeStream(streamId, connection)
125 BunkerSiloPlaceable:superClass().writeStream(self, streamId,
    connection)
126 if not connection:getIsServer() then
127 for _, bunkerSilo in ipairs(self.bunkerSilos) do
128 NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(bunkerSilo))
129 bunkerSilo:writeStream(streamId, connection)
130 g_server:registerObjectInStream(connection, bunkerSilo)
131 end
132 end
133 end
```

collectPickObjects

Description

Collect pick objects

Definition

collectPickObjects(integer node)

Arguments

integer node node id

Code

```
138  function BunkerSiloPlaceable:collectPickObjects(node)
139  -- local foundNode = false
140  -- for _, unloadTrigger in
     ipairs(self.bga.bunker.unloadingStation.unloadTriggers) do
141  -- if node == unloadTrigger.exactFillRootNode then
142  -- foundNode = true
143  -- break
144  -- end
145  -- end
146
147  if not foundNode then
148  BunkerSiloPlaceable:superClass().collectPickObjects(self, node)
149  end
150  end
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key, boolean resetVehicles)

Arguments

integer	xmlFile	id of xml object
string	key	key
boolean	resetVehicles	reset vehicles

Return Values

boolean success success

Code

```
158  function BunkerSiloPlaceable:loadFromXMLFile(xmlFile, key,
     resetVehicles)
159  if not BunkerSiloPlaceable:superClass().loadFromXMLFile(self,
     xmlFile, key, resetVehicles) then
160  return false
161  end
162
163  local i = 0
164  while true do
165  local bunkerKey = string.format("%s.bunkerSilo(%d)", key, i)
166  if not hasXMLProperty(xmlFile, bunkerKey) then
167  break
168  end
169
170  local index = getXMLInt(xmlFile, bunkerKey.."#index")
```

```
171
172     if index ~= nil then
173     if self.bunkerSilos[index] ~= nil then
174     if not self.bunkerSilos[index]:loadFromXMLFile(xmlFile,
        bunkerKey) then
175     return false
176     end
177     else
178     g_logManager:warning("Could not load bunkersilo. Given 'index'
        '%d' is not defined!", index)
179     end
180     end
181
182     i = i + 1
183     end
184
185     return true
186     end
```

saveToXMLFile

Description

Get save attributes and nodes

Definition

saveToXMLFile(string nodeIdent)

Arguments

string nodeIdent node ident

Return Values

string attributes attributes

string nodes nodes

Code

```
193     function BunkerSiloPlaceable:saveToXMLFile(xmlFile, key,
        usedModNames)
194     BunkerSiloPlaceable:superClass().saveToXMLFile(self, xmlFile,
        key, usedModNames)
195
196     for k, bunkerSilo in ipairs(self.bunkerSilos) do
197     local bunkerKey = string.format("%s.bunkerSilo(%d)", key, k-1)
198     setXMLInt(xmlFile, bunkerKey .. "#index", k)
199     bunkerSilo:saveToXMLFile(xmlFile, bunkerKey, usedModNames)
200     end
201     end
```

BuyingStationPlaceable

Description

new

Description

Creating placeable silo

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean	isServer	is server
boolean	isClient	is client
table	customMt	custom metatable

Return Values

table instance Instance of object

Code

```
26  function BuyingStationPlaceable:new(isServer, isClient, customMt)
27  local self = Placeable:new(isServer, isClient, customMt or
    SiloPlaceable_mt)
28
29  registerObjectClassName(self, "BuyingStationPlaceable")
30
31  return self
32  end
```

delete

Description

Deleting placeable silo

Definition

delete()

Code

```
36  function BuyingStationPlaceable:delete()
37  if self.buyingStation ~= nil then
38  self.buyingStation:delete()
39  end
40
41  unregisterObjectClassName(self)
42  BuyingStationPlaceable:superClass().delete(self)
43  end
```

load

Description

Load silo

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position

float	y	z world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Return Values

boolean success success

Code

```
56  function BuyingStationPlaceable:load(xmlFilename, x,y,z, rx,ry,rz, initRandom)
57  if not BuyingStationPlaceable:superClass().load(self, xmlFilename, x,y,z, rx,ry,rz, initRandom) then
58  return false
59  end
60
61  local xmlFile = loadXMLFile("TempXML", xmlFilename)
62
63  self.buyingStation = BuyingStation:new(self.isServer, self.isClient)
64  self.buyingStation:load(self.nodeId, xmlFile, "placeable.buyingStation")
65  self.buyingStation.owningPlaceable = self
66
67  delete(xmlFile)
68
69  return true
70  end
```

finalizePlacement

Description

Called if placeable is placed

Definition

finalizePlacement()

Code

```
74  function BuyingStationPlaceable:finalizePlacement()
75  BuyingStationPlaceable:superClass().finalizePlacement(self)
76
77  self.buyingStation:register(true)
78  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
84  function BuyingStationPlaceable:readStream(streamId, connection)
85  BuyingStationPlaceable:superClass().readStream(self, streamId,
    connection)
86  if connection:getIsServer() then
87  local buyingStationId = NetworkUtil.readNodeObjectId(streamId)
88  self.buyingStation:readStream(streamId, connection)
89  g_client:finishRegisterObject(self.buyingStation, buyingStationId)
90  end
91  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
97  function BuyingStationPlaceable:writeStream(streamId, connection)
98  BuyingStationPlaceable:superClass().writeStream(self, streamId,
    connection)
99  if not connection:getIsServer() then
100 NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(self.buyingStation))
101 self.buyingStation:writeStream(streamId, connection)
102 g_server:registerObjectInStream(connection, self.buyingStation)
103 end
104 end
```

collectPickObjects

Description

Collect pick objects

Definition

collectPickObjects(integer node)

Arguments

integer node node id

Code

```
109 function BuyingStationPlaceable:collectPickObjects(node)
110 local foundNode = false
```

```
111  if not foundNode then
112  for _, loadTrigger in ipairs(self.buyingStation.loadTriggers) do
113  if node == loadTrigger.triggerNode then
114  foundNode = true
115  break
116  end
117  end
118  end
119
120  if not foundNode then
121  BuyingStationPlaceable:superClass().collectPickObjects(self,
     node)
122  end
123  end
```

Doghouse

Description

new

Description

Creating instance and initializing member variables

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean isServer	is server
boolean isClient	is client
table	customMt custom meta table

Return Values

table instance Instance of object

Code

```
14  function Doghouse:new(isServer, isClient, customMt)
15  local self = Placeable:new(isServer, isClient, customMt or
    Doghouse_mt)
16
17  registerObjectClassName(self, "Doghouse")
18  -- dog
19  self.dog = Dog:new()
20
21  -- trigger
22  self.triggerNode = nil
23  self.isActivatable = false
24  self.activateText = g_i18n:getText("action_doghouseFillbowl")
25
26  -- network
```

```
27  self.forcedClipDistance = 80
28
29  if not g_currentMission.gameStarted then
30  g_currentMission:registerObjectToCallOnMissionStart(self)
31  end
32
33  return self
34  end
```

delete

Description

Delete instance

Definition

delete()

Code

```
38  function Doghouse:delete()
39  self.dog:delete()
40  self.dog = nil
41  unregisterObjectClassName(self)
42  if self.triggerNode ~= nil then
43  removeTrigger(self.triggerNode)
44  end
45  Doghouse:superClass().delete(self)
46  end
```

getCanBePlacedAt

Description

Returns true if we can place a building

Definition

getCanBePlacedAt()

Return Values

Code

```
51  function Doghouse:getCanBePlacedAt(x, y, z, distance)
52  local canBePlaced = Doghouse:superClass().getCanBePlacedAt(self,
    x, y, z, distance)
53  return canBePlaced and not self:isDoghouseRegistered()
54  end
```

canBuy

Description

Returns true if we can place a building

Definition

canBuy()

Return Values

Code

```
59  function Doghouse:canBuy()
60  local canBuy = AnimalHusbandry:superClass().canBuy(self)
61  return canBuy and not self:isDoghouseRegistered(),
    g_i18n:getText("warning_onlyOneOfThisItemAllowedPerFarm")
62  end
```

finalizePlacement

Description

Finalize placement

Definition

finalizePlacement()

Return Values

bool returns true if placement successful

Code

```
67  function Doghouse:finalizePlacement()
68  Doghouse:superClass().finalizePlacement(self)
69
70  -- dog specific information
71  local xmlFile = loadXMLFile("TempXML", self.configFileName)
72
73  if xmlFile == 0 then
74  return false
75  end
76
77  self.spawnNode = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, "placeable.dog#spawnNode"))
78  local posX, posY, posZ = getWorldTranslation(self.spawnNode)
79  local dogXMLConfig = getXMLString(xmlFile,
    "placeable.dog#xmlFilename")
80  self.dog:init(self, dogXMLConfig, posX, posY, posZ, self.isServer,
    self.isClient)
81  self:registerDoghouseToMission()
82
83  self.namePlateNode = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, "placeable.nameplate#node"))
84
85  -- player interaction trigger
86  self.triggerNode = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, "placeable.playerInteractionTrigger#node"))
87  if self.triggerNode ~= nil then
88  addTrigger(self.triggerNode, "playerInteractionTriggerCallback",
    self)
89  end
```

```
90
91  self.foodNode = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, "placeable.bowl#foodNode"))
92  delete(xmlFile)
93  if self.foodNode ~= nil then
94  setVisibility(self.foodNode, false)
95  return true
96  end
97  return false
98  end
```

onSell

Description

Called on sell of the placeable

Definition

onSell()

Code

```
102  function Doghouse:onSell()
103  Doghouse:superClass().onSell(self)
104  if self.isServer then
105  self:unregisterDoghouseToMission()
106  end
107  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId	stream ID
table	connection connection

Code

```
113  function Doghouse:readStream(streamId, connection)
114  Doghouse:superClass().readStream(self, streamId, connection)
115  if connection:getIsServer() then
116  self.dog:setName(Utils.getNoNil(streamReadString(streamId), ""))
117  end
118  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
124 function Doghouse:writeStream(streamId, connection)
125 Doghouse:superClass().writeStream(self, streamId, connection)
126 if not connection:getIsServer() then
127 streamWriteString(streamId, self.dog.name)
128 end
129 end
```

writeUpdateStream

Description

Write update network stream

Definition

writeUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId network stream identification

table connection connection information

integer dirtyMask

Code

```
136 function Doghouse:writeUpdateStream(streamId, connection,
    dirtyMask)
137 Doghouse:superClass().writeUpdateStream(self, streamId,
    connection, dirtyMask)
138 self.dog:writeUpdateStream(streamId, connection, dirtyMask)
139 end
```

readUpdateStream

Description

Read update network stream

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId network stream identification

integer timestamp

table connection connection information

Code

```
146 function Doghouse:readUpdateStream(streamId, timestamp,
    connection)
147 Doghouse:superClass().readUpdateStream(self, streamId, timestamp,
    connection)
148 self.dog:readUpdateStream(streamId, timestamp, connection)
149 end
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key, boolean resetVehicles)

Arguments

integer	xmlFile	id of xml object
string	key	key
boolean	resetVehicles	reset vehicles

Return Values

boolean	success	success

Code

```
157 function Doghouse:loadFromXMLFile(xmlFile, key, resetVehicles)
158 local result = Doghouse:superClass().loadFromXMLFile(self,
    xmlFile, key, resetVehicles)
159
160 if result then
161 self.dog:setName(getXMLString(xmlFile, key.."#animalName"))
162 return result
163 end
164 return result
165 end
```

saveToXMLFile

Description

Get save attributes and nodes

Definition

saveToXMLFile(string nodeIdent)

Arguments

string	nodeIdent	node ident

Return Values

string	attributes	attributes
string	nodes	nodes

Code

```
172 function Doghouse:saveToXMLFile(xmlFile, key, usedModNames)
173 Doghouse:superClass().saveToXMLFile(self, xmlFile, key,
    usedModNames)
174
175 setXMLString(xmlFile, key.."#animalName",
    HTMLUtil.encodeToHTML(self.dog.name))
176 end
```

testScope

Description

Test scope

Definition

testScope(float x, float y, float z, float coeff)

Arguments

float x	x position
float y	y position
float z	z position
float coeff	coeff

Return Values

boolean inScope in scope

Code

```
185  function Doghouse:testScope(x,y,z, coeff)
186  local distance, clipDistance = getCompanionClosestDistance(self.dog.dogInstance, x, y, z)
187  local clipDist = math.min(clipDistance * coeff, self.forcedClipDistance)
188
189  if distance < clipDist then
190  return true
191  else
192  return false
193  end
194  end
```

getUpdatePriority

Description

Get update priority

Definition

getUpdatePriority(float skipCount, float x, float y, float z, float coeff, table connection)

Arguments

float skipCount	skip count
float x	x position
float y	y position
float z	z position
float coeff	coeff
table connection	connection

Return Values

float priority priority

Code

```
205  function Doghouse:getUpdatePriority(skipCount, x, y, z, coeff, connection)
206  local distance, clipDistance = getCompanionClosestDistance(self.dog.dogInstance, x, y, z)
207  local clipDist = math.min(clipDistance * coeff, self.forcedClipDistance)
208  local result = (1.0 - distance / clipDist) * 0.8 + 0.5 * skipCount * 0.2
209
```

```
210 return result
211 end
```

onGhostRemove

Description

On ghost remove

Definition

onGhostRemove()

Code

```
215 function Doghouse:onGhostRemove()
216 setVisibility(self.nodeId, false)
217 removeFromPhysics(self.nodeId)
218 self.dog:setVisibility(false)
219 end
```

onGhostAdd

Description

On ghost add

Definition

onGhostAdd()

Code

```
223 function Doghouse:onGhostAdd()
224 setVisibility(self.nodeId, true)
225 addToPhysics(self.nodeId)
226 self.dog:setVisibility(true)
227 end
```

updateTick

Description

Update network tick

Definition

updateTick(float dt)

Arguments

float dt time since last call in ms

Code

```
232 function Doghouse:updateTick(dt)
233 Doghouse:superClass().updateTick(self, dt)
234
235 self.dog:updateTick(dt)
236 end
```

drawDogName

Description

Definition

drawDogName()

Code

```
252 function Doghouse:drawDogName()
253 -- @Todo: adust text width
254 setTextColor(0.843, 0.745, 0.705, 1.0)
255 setTextAlignment(RenderText.ALIGN_CENTER)
256 local x,y,z = getWorldTranslation(self.namePlateNode)
257 local rx,ry,rz = getWorldRotation(self.namePlateNode)
258 renderText3D(x,y,z, rx,ry,rz, 0.04, self.dog.name)
259 setTextAlignment(RenderText.ALIGN_LEFT)
260 end
```

playerInteractionTriggerCallback

Description

Callback when interaction trigger is activated

Definition

playerInteractionTriggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

Code

```
269 function Doghouse:playerInteractionTriggerCallback(triggerId,
    otherId, onEnter, onLeave, onStay)
270 if g_currentMission.player ~= nil and otherId ==
    g_currentMission.player.rootNode and
    g_currentMission.player.farmId == self:getOwnerFarmId() then
271 if onEnter then
272 self.isActivatable = true
273 elseif onLeave then
274 self.isActivatable = false
275 end
276 end
277 self:raiseActive()
278 end
```

onActivateObject

Description

Callback on bowl activated

Definition

onActivateObject()

Code

```
282 function Doghouse:onActivateObject()
```

```
283  self:raiseActive()
284  end
```

getIsActivatable

Description

Checks if doghouse can be activated and bowl is empty

Definition

getIsActivatable()

Return Values

bool return true if doghouse can be activated to fill bowl

Code

```
289  function Doghouse:getIsActivatable()
290  return self.isActivatable
291  end
```

drawActivate

Description

Draw HUD icon

Definition

drawActivate()

Code

```
295  function Doghouse:drawActivate()
296  g_currentMission:showFillDogBowlContext()
297  end
```

onPlayerLeave

Description

Teleport animal back to doghouse

Definition

onPlayerLeave()

Code

```
301  function Doghouse:onPlayerLeave()
302  self:raiseActive()
303  end
```

isDoghouseRegistered

Description

Returns true if a doghouse is registered

Definition

isDoghouseRegistered()

Return Values

Code

```
308  function Doghouse:isDoghouseRegistered()
309  local dogHouse =
     g_currentMission:getDoghouse(self:getOwnerFarmId())
310  return dogHouse ~= nil
```

```
311 end
```

registerDoghouseToMission

Description

Registers the doghouse to the mission game.

Definition

registerDoghouseToMission()

Return Values

bool true if registration went well

Code

```
316 function Doghouse:registerDoghouseToMission()
317 if not self:isDoghouseRegistered() and self.dog ~= nil then
318 g_currentMission.doghouses[self.dog] = self
319 return true
320 end
321 return false
322 end
```

unregisterDoghouseToMission

Description

Registers the doghouse to the mission game.

Definition

unregisterDoghouseToMission()

Return Values

bool true if registration went well

Code

```
327 function Doghouse:unregisterDoghouseToMission()
328 if self.dog ~= nil then
329 g_currentMission.doghouses[self.dog] = nil
330 return true
331 end
332 return false
333 end
```

FarmhousePlaceable

Description

Class for placeable farmhouse

Parent

Placeable

new

Description

Creating placeable farmhouse

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean	isServer	is server
boolean	isClient	is client
table	customMt	custom metatable

Return Values

table instance Instance of object

Code

```
25  function FarmhousePlaceable:new(isServer, isClient, customMt)
26  local self = Placeable:new(isServer, isClient, customMt or
    FarmhousePlaceable_mt)
27
28  registerObjectClassName(self, "FarmhousePlaceable")
29
30  return self
31  end
```

delete

Description

Deleting placeable farmhouse

Definition

delete()

Code

```
35  function FarmhousePlaceable:delete()
36  unregisterObjectClassName(self)
37
38  if self.sleepingTrigger ~= nil then
39  removeTrigger(self.sleepingTrigger)
40  end
41
42  g_currentMission:removeActivatableObject(self)
43  self.objectActivated = false
44
45  FarmhousePlaceable:superClass().delete(self)
46  end
```

load

Description

Load farmhouse

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position
float	y	z world position

float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Return Values

boolean success success

Code

```
59  function FarmhousePlaceable:load(xmlFilename, x,y,z, rx,ry,rz,
    initRandom)
60  if not FarmhousePlaceable:superClass().load(self, xmlFilename,
    x,y,z, rx,ry,rz, initRandom) then
61  return false
62  end
63
64  local xmlFile = loadXMLFile("TempXML", xmlFilename)
65
66  self.spawnNode = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, "placeable.farmhouse#spawnNode"))
67
68  self.sleepingTrigger = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, "placeable.farmhouse.sleeping#triggerNode"))
69  if self.sleepingTrigger ~= nil then
70  addTrigger(self.sleepingTrigger, "sleepingTriggerCallback", self)
71  end
72  self.sleepingCamera = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, "placeable.farmhouse.sleeping#cameraNode"))
73
74  self.activateText = g_i18n:getText("ui_inGameSleep")
75
76  self.isEnabled = true
77  self.objectActivated = false
78
79
80  delete(xmlFile)
81
82  return true
83  end
```

getIsActivatable

Description

Get is activateable

Definition

getIsActivatable()

Return Values

boolean isActivateable is activateable

Code

```
134 function FarmhousePlaceable:getIsActivatable()
135 return g_currentMission:getFarmId() == self:getOwnerFarmId()
136 end
```

onActivateObject

Description

On activate object

Definition

onActivateObject()

Code

```
144 function FarmhousePlaceable:onActivateObject()
145 g_sleepManager:showDialog()
146 end
```

GreenhousePlaceable

Description

Class for placeable greenhouse

new

Description

Creating placeable greenhouse

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean	isServer	is server
boolean	isClient	is client
table	customMt	custom metatable

Return Values

table instance Instance of object

Code

```
20 function GreenhousePlaceable:new(isServer, isClient, customMt)
21 local self = Placeable:new(isServer, isClient, customMt or
   GreenhousePlaceable_mt)
22
23 registerObjectClassName(self, "GreenhousePlaceable")
24
25 self.waterTankCapacity = 1000
26 self.waterTankFillLevel = 1000
27 self.sentWaterTankFillLevel = self.waterTankFillLevel
28 self.waterTankUsagePerHour = 0
29 self.waterTankFillLitersPerSecond = 200
30 self.waterTrailers = {}
```

```
31  self.isFruitAlive = true
32  self.displayFruit = false
33
34  self.manureFillLevel = 0
35  self.manureUsagePerHour = 0
36  self.manureCapacity = 200
37  self.manurePlaneMinY = 0
38  self.manurePlaneMaxY = 1
39  self.sentManureFillLevel = self.manureFillLevel
40
41  self.vehiclesInRange = {}
42  self.playerInRange = false
43
44  self.greenhousePlaceableDirtyFlag = self:getNextDirtyFlag()
45
46  self.waterTrailerActivatable =
    GreenhousePlaceableWaterTankActivatable:new(self)
47
48  return self
49  end
```

delete

Description

Deleting placeable greenhouse

Definition

delete()

Code

```
53  function GreenhousePlaceable:delete()
54  if self.shovelTarget ~= nil then
55  self.shovelTarget:delete()
56  end
57
58  g_currentMission:removeActivatableObject(self.waterTrailerActivatable)
59  unregisterObjectClassName(self)
60  g_currentMission.environment:removeHourChangeListener(self)
61
62  if self.waterTankTriggerNode ~= nil then
63  removeTrigger(self.waterTankTriggerNode)
64  end
65
66  if g_client ~= nil then
67  for _,sample in pairs(self.samples) do
```

```
68  g_soundManager:deleteSample(sample)
69  end
70  end
71
72  GreenhousePlaceable:superClass().delete(self)
73  end
```

deleteFinal

Description

Deleting placeable greenhouse final

Definition

deleteFinal()

Code

```
77  function GreenhousePlaceable:deleteFinal()
78  GreenhousePlaceable:superClass().deleteFinal(self)
79  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
85  function GreenhousePlaceable:readStream(streamId, connection)
86  GreenhousePlaceable:superClass().readStream(self, streamId,
    connection)
87  if connection:getIsServer() then
88  local waterTankFillLevel =
    streamReadUInt8(streamId)/255*self.waterTankCapacity
89  self:setWaterTankFillLevel(waterTankFillLevel)
90  local manureFillLevel =
    streamReadUInt8(streamId)/255*self.manureCapacity
91  self:setManureFillLevel(manureFillLevel)
92  end
93  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
 99  function GreenhousePlaceable:writeStream(streamId, connection)
100  GreenhousePlaceable:superClass().writeStream(self, streamId, connection)
101  if not connection:getIsServer() then
102  streamWriteUInt8(streamId, math.floor(self.waterTankFillLevel/self.waterTankCapacity * 255))
103  streamWriteUInt8(streamId, math.floor(self.manureFillLevel/self.manureCapacity * 255))
104  end
105  end
```

readUpdateStream

Description

Called on client side on update

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId stream ID

integer timestamp timestamp

table connection connection

Code

```
112  function GreenhousePlaceable:readUpdateStream(streamId, timestamp, connection)
113  GreenhousePlaceable:superClass().readUpdateStream(self, streamId, timestamp, connection)
114  if connection:getIsServer() then
115  local waterTankFillLevel = streamReadUInt8(streamId)/255*self.waterTankCapacity
116  self:setWaterTankFillLevel(waterTankFillLevel)
117  local manureFillLevel = streamReadUInt8(streamId)/255*self.manureCapacity
118  self:setManureFillLevel(manureFillLevel)
119  end
120  end
```

writeUpdateStream

Description

Called on server side on update

Definition

writeUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId stream ID

table	connection	connection
integer	dirtyMask	dirty mask

Code

```
127  function GreenhousePlaceable:writeUpdateStream(streamId,
     connection, dirtyMask)
128  GreenhousePlaceable:superClass().writeUpdateStream(self,
     streamId, connection, dirtyMask)
129  if not connection:getIsServer() then
130  streamWriteUInt8(streamId,
     math.floor(self.waterTankFillLevel/self.waterTankCapacity * 255))
131  streamWriteUInt8(streamId,
     math.floor(self.manureFillLevel/self.manureCapacity * 255))
132  end
133  end
```

createNode

Description

Create node

Definition

createNode(string i3dFilename)

Arguments

string i3dFilename i3d file name

Return Values

boolean success success

Code

```
139  function GreenhousePlaceable:createNode(i3dFilename)
140  if not GreenhousePlaceable:superClass().createNode(self,
     i3dFilename) then
141  return false
142  end
143
144  return true
145  end
```

load

Description

Load greenhouse

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position
float	y	z world position
float	z	z world position
float	rx	rx world rotation

float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Return Values

boolean success success

Code

```
158 function GreenhousePlaceable:load(xmlFilename, x,y,z, rx,ry,rz, initRandom)
159 if not GreenhousePlaceable:superClass().load(self, xmlFilename, x,y,z, rx,ry,rz, initRandom) then
160 return false
161 end
162
163 local xmlFile = loadXMLFile("TempXML", xmlFilename)
164
165 local fruitAlive = I3DUtil.indexToObject(self.nodeId, getXMLString(xmlFile, "placeable.fruit#alive"))
166 local fruitDead = I3DUtil.indexToObject(self.nodeId, getXMLString(xmlFile, "placeable.fruit#dead"))
167 if fruitAlive ~= nil then
168 self.fruitAlive = fruitAlive
169 end
170 if fruitDead ~= nil then
171 self.fruitDead = fruitDead
172 end
173
174 self.waterTankTriggerNode = I3DUtil.indexToObject(self.nodeId, getXMLString(xmlFile, "placeable.waterTank#triggerNode"))
175 self.waterTankCapacity = Utils.getNoNil(getXMLFloat(xmlFile, "placeable.waterTank#capacity"), self.waterTankCapacity)
176 self.waterTankUsagePerHour = Utils.getNoNil(getXMLFloat(xmlFile, "placeable.waterTank#usagePerHour"), self.waterTankUsagePerHour)
177 self.waterTankFillLitersPerSecond = Utils.getNoNil(getXMLFloat(xmlFile, "placeable.waterTank#fillLitersPerSecond"), self.waterTankFillLitersPerSecond)
178
179 self.samples = {}
180 if g_client ~= nil then
181 self.samples.refuel = g_soundManager:loadSampleFromXML(xmlFile, "placeable.waterTank.sounds", "refuel", self.baseDirectory, self.nodeId, 0, AudioGroup.VEHICLE, nil, nil)
182 end
183
```

```
184 self.manurePlane = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, "placeable.manurePlane#node"))
185 self.shovelTargetNode = I3DUtil.indexToObject(self.nodeId,
    Utils.getNoNil(getXMLString(xmlFile,
    "placeable.manurePlane#shovelTargetIndex"), getXMLString(xmlFile,
    "placeable.manurePlane#collisionNode")))
186 local minY, maxY =
    StringUtil.getVectorFromString(getXMLString(xmlFile,
    "placeable.manurePlane#minMaxY"))
187 if minY ~= nil and maxY ~= nil then
188 self.manurePlaneMinY = minY
189 self.manurePlaneMaxY = maxY
190 end
191 self.manureCapacity = Utils.getNoNil(getXMLFloat(xmlFile,
    "placeable.manurePlane#capacity"), self.manureCapacity)
192 self.manureUsagePerHour = Utils.getNoNil(getXMLFloat(xmlFile,
    "placeable.manurePlane#usagePerHour"), self.manureUsagePerHour)
193
194 delete(xmlFile)
195
196 self:setWaterTankFillLevel(0)
197 self.sentWaterTankFillLevel = self.waterTankFillLevel
198
199 self:setManureFillLevel(0)
200 self.sentManureFillLevel = self.manureFillLevel
201
202 return true
203 end
```

finalizePlacement

Description

Called if placeable is placed

Definition

finalizePlacement()

Code

```
207 function GreenhousePlaceable:finalizePlacement()
208 GreenhousePlaceable:superClass().finalizePlacement(self)
209 g_currentMission.environment:addHourChangeListener(self)
210 if self.waterTankTriggerNode ~= nil then
211 addTrigger(self.waterTankTriggerNode, "onWaterTankTrigger", self)
212 end
213
214 if self.shovelTargetNode ~= nil then
```

```
215  if self.shovelTargetNode ~= nil then
216  self.shovelTarget = ShovelTarget:new(self)
217  if not self.shovelTarget:load(self.shovelTargetNode) then
218  self.shovelTarget:delete()
219  self.shovelTarget = nil
220  end
221  end
222  end
223  end
```

initPose

Description

Initialize pose

Definition

initPose(float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

float	x	x world position
float	y	y world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Code

```
234  function GreenhousePlaceable:initPose(x,y,z, rx,ry,rz,
     initRandom)
235  GreenhousePlaceable:superClass().initPose(self, x,y,z, rx,ry,rz,
     initRandom)
236  end
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key, boolean resetVehicles)

Arguments

integer	xmlFile	id of xml object
string	key	key
boolean	resetVehicles	reset vehicles

Return Values

boolean success success

Code

```
244  function GreenhousePlaceable:loadFromXMLFile(xmlFile, key,
     resetVehicles)
245
```

```
246 if not GreenhousePlaceable:superClass().loadFromXMLFile(self,
    xmlFile, key, resetVehicles) then
247 return false
248 end
249
250 local waterTankFillLevel = getXMLFloat(xmlFile,
    key.."#waterTankFillLevel")
251 if waterTankFillLevel ~= nil then
252 self:setWaterTankFillLevel(waterTankFillLevel)
253 end
254 local manureFillLevel = getXMLFloat(xmlFile,
    key.."#manureFillLevel")
255 if manureFillLevel ~= nil then
256 self:setManureFillLevel(manureFillLevel)
257 end
258
259 return true
260 end
```

saveToXMLFile

Description

Get save attributes and nodes

Definition

saveToXMLFile(string nodeIdent)

Arguments

string nodeIdent node ident

Return Values

string attributes attributes

string nodes nodes

Code

```
267 function GreenhousePlaceable:saveToXMLFile(xmlFile, key,
    usedModNames)
268 GreenhousePlaceable:superClass().saveToXMLFile(self, xmlFile,
    key, usedModNames)
269
270 setXMLFloat(xmlFile, key.."#waterTankFillLevel",
    self.waterTankFillLevel)
271 setXMLFloat(xmlFile, key.."#manureFillLevel",
    self.manureFillLevel)
272 end
```

update

Description

Update

Definition

update(float dt)

Arguments

float dt time since last call in ms

Code

```
277  function GreenhousePlaceable:update(dt)
278  GreenhousePlaceable:superClass().update(self, dt)
279  if self:getShowInfo() then
280  g_currentMission:addExtraPrintText(g_i18n:getText("info_waterFillLevel")
     ".."math.floor(self.waterTankFillLevel).."
     (".."math.floor(100*self.waterTankFillLevel/self.waterTankCapacity).."%)"
281  g_currentMission:addExtraPrintText(g_i18n:getText("info_manureFillLevel"
     ".."math.floor(self.manureFillLevel).."
     (".."math.floor(100*self.manureFillLevel/self.manureCapacity).."%)")
282  end
283  end
```

updateTick

Description

updateTick

Definition

updateTick(float dt)

Arguments

float dt time since last call in ms

Code

```
288  function GreenhousePlaceable:updateTick(dt)
289  if self.isServer then
290
291  if self.waterTankFillLevel ~= self.sentWaterTankFillLevel then
292  self:raiseDirtyFlags(self.greenhousePlaceableDirtyFlag)
293
294  self.sentWaterTankFillLevel = self.waterTankFillLevel
295  end
296  if self.manureFillLevel ~= self.sentManureFillLevel then
297  self:raiseDirtyFlags(self.greenhousePlaceableDirtyFlag)
298
299  self.sentManureFillLevel = self.manureFillLevel
300  end
301
302  if self.isWaterTankFilling then
303  local disableFilling = false
304  local waterFillLevel =
     self.waterTankFillTrailer:getFillLevel(FillType.WATER)
305  if waterFillLevel > 0 then
```

```
306 local oldFillLevel = self.waterTankFillLevel
307
308 local delta = self.waterTankFillLitersPerSecond*dt*0.001
309 delta = math.min(delta, waterFillLevel)
310
311 self:setWaterTankFillLevel(self.waterTankFillLevel + delta)
312
313 local delta = self.waterTankFillLevel - oldFillLevel
314 if delta <= 0 then
315 disableFilling = true
316 end
317 self.waterTankFillTrailer:setFillLevel(waterFillLevel - delta,
    FillType.WATER, true)
318 else
319 disableFilling = true
320 end
321 if disableFilling then
322 self:setIsWaterTankFilling(false)
323 end
324 end
325
326 end
327 end
```

getShowInfo

Description

Returns true if info should show

Definition

getShowInfo()

Return Values

boolean showInfo show info

Code

```
332 function GreenhousePlaceable:getShowInfo()
333 if (g_currentMission.controlPlayer and self.playerInRange) then
334 return true
335 end
336 if not g_currentMission.controlPlayer then
337 for vehicle in pairs(self.vehiclesInRange) do
338 if vehicle:getIsActiveForInput() then
339 return true
340 end
341 end
```

```
342 end
343 return false
344 end
```

hourChanged

Description

Called if hour changed

Definition

hourChanged()

Code

```
348 function GreenhousePlaceable:hourChanged()
349 if self.isServer then
350 self:setWaterTankFillLevel(self.waterTankFillLevel -
    self.waterTankUsagePerHour)
351 self:setManureFillLevel(self.manureFillLevel -
    self.manureUsagePerHour)
352
353 if self.isFruitAlive then
354 local income = self.incomePerHour
355 if self.manureFillLevel > 0 then
356 income = income * 2
357 end
358 g_currentMission:addMoney(income, self:getOwnerFarmId(),
    "propertyIncome")
359 g_currentMission:addMoneyChange(income, self:getOwnerFarmId(),
    EconomyManager.MONEY_TYPE_PROPERTY_INCOME)
360 end
361 end
362 end
```

setWaterTankFillLevel

Description

Set water tank fill level

Definition

setWaterTankFillLevel(float fillLevel)

Arguments

float fillLevel new fill level

Code

```
367 function GreenhousePlaceable:setWaterTankFillLevel(fillLevel)
368 self.waterTankFillLevel = MathUtil.clamp(fillLevel, 0,
    self.waterTankCapacity)
369
370 self.isFruitAlive = self.waterTankFillLevel > 0.0001
371
```

```
372  if self.waterTankFillLevel > 0 then
373  self.displayFruit = true
374  end
375
376  if self.fruitAlive ~= nil then
377  setVisibility(self.fruitAlive, self.isFruitAlive and
     self.displayFruit)
378  end
379  if self.fruitDead ~= nil then
380  setVisibility(self.fruitDead, not self.isFruitAlive and
     self.displayFruit)
381  end
382  end
```

addFillLevelFromTool_OLD

Description

Add fill level from tool to trigger

Definition

addFillLevelFromTool_OLD(table object, float fillDelta, integer fillType, integer toolType)

Arguments

table	object	object
float	fillDelta	delta to fill
integer	fillType	fill type index
integer	toolType	tool type index

Return Values

float fillDelta real fill delta

Code

```
391  function GreenhousePlaceable:addFillLevelFromTool_OLD(object,
     fillDelta, fillType, toolType)
392  fillDelta = math.min(fillDelta, self.manureCapacity-
     self.manureFillLevel)
393
394  if fillDelta > 0 then
395  self:setManureFillLevel(self.manureFillLevel+fillDelta)
396  end
397
398  return fillDelta
399  end
```

getAllowFillTypeFromTool

Description

Returns if shovel fill type is allowed

Definition

getAllowFillTypeFromTool()

Return Values

float isAllowed is allowed

Code

```
404  function GreenhousePlaceable:getAllowFillTypeFromTool(fillType)
405  return fillType == FillType.MANURE
406  end
```

setManureFillLevel

Description

Set manure fill level

Definition

setManureFillLevel(float fillLevel)

Arguments

float fillLevel new fill level

Code

```
411  function GreenhousePlaceable:setManureFillLevel(fillLevel)
412  self.manureFillLevel = MathUtil.clamp(fillLevel, 0,
     self.manureCapacity)
413
414  if self.manurePlane ~= nil then
415  setVisibility(self.manurePlane, self.manureFillLevel > 0.0001)
416  local x,y,z = getTranslation(self.manurePlane)
417  y = self.manurePlaneMinY +
     self.manureFillLevel/self.manureCapacity * (self.manurePlaneMaxY
     - self.manurePlaneMinY)
418
419  setTranslation(self.manurePlane, x,y,z)
420  end
421  end
```

setIsWaterTankFilling

Description

Set is water tank filling

Definition

setIsWaterTankFilling(boolean isWaterTankFilling, table trailer, boolean noEventSend)

Arguments

boolean	isWaterTankFilling	is water tank filling
table	trailer	trailer
boolean	noEventSend	no event send

Code

```
428  function
     GreenhousePlaceable:setIsWaterTankFilling(isWaterTankFilling,
     trailer, noEventSend)
429  GreenhouseSetIsWaterTankFillingEvent.sendEvent(self,
     isWaterTankFilling, trailer, noEventSend)
```

```
430  if self.isWaterTankFilling ~= isWaterTankFilling then
431  self.isWaterTankFilling = isWaterTankFilling
432  self.waterTankFillTrailer = trailer
433  end
434  if g_client ~= nil then
435  if isWaterTankFilling then
436  g_soundManager:playSample(self.samples.refuel)
437  else
438  g_soundManager:stopSample(self.samples.refuel)
439  end
440  end
441  end
```

addWaterTrailer

Description

Add water trailer in range

Definition

addWaterTrailer(table trailer)

Arguments

table trailer trailer

Code

```
446  function GreenhousePlaceable:addWaterTrailer(trailer)
447  if table.getn(self.waterTrailers) == 0 then
448  g_currentMission:addActivatableObject(self.waterTrailerActivatable)
449  end
450  table.insert(self.waterTrailers, trailer)
451  end
```

removeWaterTrailer

Description

Remove water trailer in range

Definition

removeWaterTrailer(table trailer)

Arguments

table trailer trailer

Code

```
456  function GreenhousePlaceable:removeWaterTrailer(trailer)
457  for i=1, table.getn(self.waterTrailers) do
458  if self.waterTrailers[i] == trailer then
459  table.remove(self.waterTrailers, i)
460  break
461  end
462  end
```

```
463 if table.getn(self.waterTrailers) == 0 then
464 if self.isServer then
465 self:setIsWaterTankFilling(false)
466 end
467 g_currentMission:removeActivatableObject(self.waterTrailerActivatable)
468 end
469 end
```

onWaterTankTrigger

Description

Water tank trigger

Definition

onWaterTankTrigger(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay, integer otherShapeId)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay
integer	otherShapeId	id of other actor

Code

```
479 function GreenhousePlaceable:onWaterTankTrigger(triggerId,
    otherActorId, onEnter, onLeave, onStay, otherShapeId)
480 if onEnter or onLeave then
481
482 if g_currentMission.player ~= nil and otherActorId ==
    g_currentMission.player.rootNode then
483 if onEnter then
484 self.playerInRange = true
485 else
486 self.playerInRange = false
487 end
488 else
489 local vehicle = g_currentMission.nodeToObject[otherActorId]
490 if vehicle ~= nil then
491 if onEnter then
492 self.vehiclesInRange[vehicle] = true
493 else
494 self.vehiclesInRange[vehicle] = nil
495 end
496 end
497
```

```
498
499  local trailer = g_currentMission.objectToTrailer[otherShapeId]
500  if trailer ~= nil and trailer:allowFillType(FillType.WATER,
     false) then
501  if onEnter then
502  self:addWaterTrailer(trailer)
503  else -- onLeave
504  self:removeWaterTrailer(trailer)
505  end
506  end
507  end
508  end
509  end
```

HeatingPlantPlaceable

Description

Class for placeable heating plants

Parent

Placeable

new

Description

Creating placeable high pressure washer

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean isServer	is server
boolean isClient	is client
table	customMt custom metatable

Return Values

table instance Instance of object

Code

```
18  function HeatingPlantPlaceable:new(isServer, isClient, customMt)
19  local mt = customMt;
20  if mt == nil then
21  mt = HeatingPlantPlaceable_mt;
22  end;
23
24  local self = Placeable:new(isServer, isClient, mt);
25
26  registerObjectClassName(self, "HeatingPlantPlaceable");
27
28  self.heatingPlantDirtyFlag = self:getNextDirtyFlag();
```

```
29
30  return self;
31  end;
```

delete

Description

Deleting heating plant

Definition

delete()

Code

```
35  function HeatingPlantPlaceable:delete()
36  if self.tipTrigger ~= nil then
37  self.tipTrigger:removeUpdateEventListener(self)
38  self.tipTrigger:delete()
39  end
40
41  if self.exhaustEffect ~= nil then
42  g_i3DManager:releaseSharedI3DFile(self.exhaustEffect.filename,
    self.baseDirectory, true);
43  end
44
45  g_animationManager:deleteAnimations(self.animationNodes)
46
47  unregisterObjectClassName(self);
48  HeatingPlantPlaceable:superClass().delete(self);
49  end;
```

load

Description

Load heating plant

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position
float	y	z world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Return Values

boolean success success

Code

```
62  function HeatingPlantPlaceable:load(xmlFilename, x,y,z, rx,ry,rz, initRandom)
63  if not HeatingPlantPlaceable:superClass().load(self, xmlFilename, x,y,z, rx,ry,rz, initRandom) then
64  return false;
65  end;
66
67  local xmlFile = loadXMLFile("TempXML", xmlFilename);
68
69  self.tipTrigger = TipTrigger:new(self.isServer, self.isClient)
70  if self.tipTrigger:load(I3DUtil.indexToObject(self.nodeId, getXMLString(xmlFile, "placeable.heatingPlant#tipTrigger"))) then
71  self.tipTrigger:register(true)
72  self.tipTrigger:addUpdateEventListener(self)
73  else
74  self.tipTrigger:delete()
75  self.tipTrigger = nil
76  end
77
78  if self.isClient then
79  self.animationNodes = g_animationManager:loadAnimations(xmlFile, "placeable.animationNodes", self.nodeId, self, nil)
80
81  local filename = getXMLString(xmlFile, "placeable.exhaust#filename");
82  local node = I3DUtil.indexToObject(self.nodeId, getXMLString(xmlFile, "placeable.exhaust#index"));
83  if filename ~= nil then
84  local i3dNode = g_i3DManager:loadSharedI3DFile(filename, self.baseDirectory, false, false, false);
85  if i3dNode ~= 0 then
86  self.exhaustEffect = {}
87  self.exhaustEffect.node = getChildAt(i3dNode, 0);
88  self.exhaustEffect.filename = filename
89  link(Utils.getNoNil(node, self.nodeId), self.exhaustEffect.node);
90  setVisibility(self.exhaustEffect.node, false);
91  setShaderParameter(self.exhaustEffect.node, "param", 0, 0, 0, 0.4, false);
92  delete(i3dNode);
93  end
94  end
```

```
95  end
96
97  self.workingTimeDuration = 120*1000
98  self.workingTime = 0
99
100 delete(xmlFile);
101
102 return true;
103 end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID
table connection connection

Code

```
109 function HeatingPlantPlaceable:readStream(streamId, connection)
110 HeatingPlantPlaceable:superClass().readStream(self, streamId,
    connection);
111 if connection:getIsServer() then
112 local tipTriggerId = NetworkUtil.readNodeObjectId(streamId);
113 self.tipTrigger:readStream(streamId, connection);
114 g_client:finishRegisterObject(self.tipTrigger, tipTriggerId);
115 end
116 end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID
table connection connection

Code

```
122 function HeatingPlantPlaceable:writeStream(streamId, connection)
123 HeatingPlantPlaceable:superClass().writeStream(self, streamId,
    connection);
124 if not connection:getIsServer() then
125 NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(self.tipTrigger));
```

```
126 	self.tipTrigger:writeStream(streamId, connection);
127 	g_server:registerObjectInStream(connection, self.tipTrigger);
128 	end
129 	end
```

readUpdateStream

Description

Called on client side on update

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId	stream ID
integer timestamp	timestamp
table	connection connection

Code

```
136 	function HeatingPlantPlaceable:readUpdateStream(streamId,
	timestamp, connection)
137 	HeatingPlantPlaceable:superClass().readUpdateStream(self,
	streamId, timestamp, connection);
138 	if connection:getIsServer() then
139 	local workingTimeBiggerZero = streamReadBool(streamId)
140 	if workingTimeBiggerZero then
141 	self.workingTime = self.workingTimeDuration
142 	if self.exhaustEffect ~= nil then
143 	setVisibility(self.exhaustEffect.node, true);
144 	end
145 	end
146 	end;
147 	end;
```

writeUpdateStream

Description

Called on server side on update

Definition

writeUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Code

```
154 	function HeatingPlantPlaceable:writeUpdateStream(streamId,
	connection, dirtyMask)
155 	HeatingPlantPlaceable:superClass().writeUpdateStream(self,
	streamId, connection, dirtyMask);
156 	if not connection:getIsServer() then
```

```
157   streamWriteBool(streamId, self.workingTime > 0)
158   end;
159   end;
```

collectPickObjects

Description

Collect pick objects

Definition

collectPickObjects(integer node)

Arguments

integer node node id

Code

```
164   function HeatingPlantPlaceable:collectPickObjects(node)
165   if self.tipTrigger == nil or self.tipTrigger.shovelTarget == nil
      or self.tipTrigger.shovelTarget.nodeId ~= node then
166   HeatingPlantPlaceable:superClass().collectPickObjects(self, node)
167   end
168   end;
```

update

Description

Update

Definition

update(float dt)

Arguments

float dt time since last call in ms

Code

```
173   function HeatingPlantPlaceable:update(dt)
174   HeatingPlantPlaceable:superClass().update(self, dt);
175
176   if self.isClient then
177   if self.workingTime > 0 then
178   self.workingTime = self.workingTime - dt
179   if self.workingTime <= 0 then
180   if self.exhaustEffect ~= nil then
181   setVisibility(self.exhaustEffect.node, false);
182   end
183   g_animationManager:stopAnimations(self.animationNodes)
184   end
185   end
186   end
187   end
```

updateTick

Description

Update tick

Definition

updateTick(float dt)

Arguments

float dt time since last call in ms

Code

```
192  function HeatingPlantPlaceable:updateTick(dt)
193  HeatingPlantPlaceable:superClass().updateTick(self, dt);
194  end
```

onUpdateEvent

Description

Called if someone filled in something

Definition

onUpdateEvent(table trigger, float fillDelta, integer fillType, table trailer, table tipTriggerTarget)

Arguments

table	trigger	trigger
float	fillDelta	delta filled in
integer	fillType	fill type filled in
table	trailer	object of trailer
table	tipTriggerTarget	tip trigger target

Code

```
203  function HeatingPlantPlaceable:onUpdateEvent(trigger, fillDelta,
     fillType, trailer, tipTriggerTarget)
204  if self.exhaustEffect ~= nil then
205  setVisibility(self.exhaustEffect.node, true);
206  end
207  if self.isServer then
208  self:raiseDirtyFlags(self.heatingPlantDirtyFlag);
209  end
210  self.workingTime = self.workingTimeDuration
211
212  g_animationManager:startAnimations(self.animationNodes)
213  end
```

SellingStationPlaceable

Description

new

Description

Creating placeable silo

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean isServer	is server

boolean	isClient	is client
table	customMt	custom metatable

Return Values

table	instance	Instance of object

Code

```
26  function SellingStationPlaceable:new(isServer, isClient, customMt)
27  local self = Placeable:new(isServer, isClient, customMt or
    SellStationPlaceable_mt)
28
29  registerObjectClassName(self, "SellingStationPlaceable")
30
31  return self
32  end
```

delete

Description

Deleting placeable silo

Definition

delete()

Code

```
36  function SellingStationPlaceable:delete()
37  if self.sellingStation ~= nil then
38  g_currentMission:removeUnloadingStation(self.sellingStation)
39  self.sellingStation:delete()
40  end
41
42  unregisterObjectClassName(self)
43  SellingStationPlaceable:superClass().delete(self)
44  end
```

load

Description

Load silo

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position
float	y	z world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation

boolean initRandom initialize random

Return Values

boolean success success

Code

```
57  function SellingStationPlaceable:load(xmlFilename, x,y,z, rx,ry,rz, initRandom)
58  if not SellingStationPlaceable:superClass().load(self, xmlFilename, x,y,z, rx,ry,rz, initRandom) then
59  return false
60  end
61
62  local xmlFile = loadXMLFile("TempXML", xmlFilename)
63
64  self.sellingStation = SellingStation:new(self.isServer, self.isClient)
65  self.sellingStation:load(self.nodeId, xmlFile, "placeable.sellingStation")
66  self.sellingStation.owningPlaceable = self
67
68  delete(xmlFile)
69
70  return true
71  end
```

finalizePlacement

Description

Called if placeable is placed

Definition

finalizePlacement()

Code

```
75  function SellingStationPlaceable:finalizePlacement()
76  SellingStationPlaceable:superClass().finalizePlacement(self)
77  self.sellingStation:register(true)
78  g_currentMission:addUnloadingStation(self.sellingStation)
79  end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
85  function SellingStationPlaceable:readStream(streamId, connection)
86  SellingStationPlaceable:superClass().readStream(self, streamId,
    connection)
87  if connection:getIsServer() then
88  local sellingStationId = NetworkUtil.readNodeObjectId(streamId)
89  self.sellingStation:readStream(streamId, connection)
90  g_client:finishRegisterObject(self.sellingStation,
    sellingStationId)
91  end
92  end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
98  function SellingStationPlaceable:writeStream(streamId,
    connection)
99  SellingStationPlaceable:superClass().writeStream(self, streamId,
    connection)
100 if not connection:getIsServer() then
101 NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(self.sellingStation))
102 self.sellingStation:writeStream(streamId, connection)
103 g_server:registerObjectInStream(connection, self.sellingStation)
104 end
105 end
```

collectPickObjects

Description

Collect pick objects

Definition

collectPickObjects(integer node)

Arguments

integer node node id

Code

```
110 function SellingStationPlaceable:collectPickObjects(node)
111 local foundNode = false
112 for _, unloadTrigger in
    ipairs(self.sellingStation.unloadTriggers) do
113 if node == unloadTrigger.exactFillRootNode then
```

```
114 foundNode = true
115 break
116 end
117 end
118
119 if not foundNode then
120 SellingStationPlaceable:superClass().collectPickObjects(self,
    node)
121 end
122 end
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key, boolean resetVehicles)

Arguments

integer	xmlFile	id of xml object
string	key	key
boolean	resetVehicles	reset vehicles

Return Values

boolean success success

Code

```
130 function SellingStationPlaceable:loadFromXMLFile(xmlFile, key,
    resetVehicles)
131 if not SellingStationPlaceable:superClass().loadFromXMLFile(self,
    xmlFile, key, resetVehicles) then
132 return false
133 end
134
135 if not self.sellingStation:loadFromXMLFile(xmlFile,
    key..".sellingStation") then
136 return false
137 end
138
139 return true
140 end
```

saveToXMLFile

Description

Get save attributes and nodes

Definition

saveToXMLFile(string nodeIdent)

Arguments

string nodeIdent node ident

Return Values

string attributes attributes

string nodes nodes

Code

```
147 function SellingStationPlaceable:saveToXMLFile(xmlFile, key,
    usedModNames)
148 SellingStationPlaceable:superClass().saveToXMLFile(self, xmlFile,
    key, usedModNames)
149
150 self.sellingStation:saveToXMLFile(xmlFile,
    key..".sellingStation", usedModNames)
151 end
```

SiloPlaceable

Description

Class for placeable silo

Parent

Placeable

new

Description

Creating placeable silo

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean isServer is server

boolean isClient is client

table customMt custom metatable

Return Values

table instance Instance of object

Code

```
18 function SiloPlaceable:new(isServer, isClient, customMt)
19 local self = Placeable:new(isServer, isClient, customMt or
   SiloPlaceable_mt)
20
21 registerObjectClassName(self, "SiloPlaceable")
22
23 return self
24 end
```

delete

Description

Deleting placeable silo

Definition

delete()

Code

```
28  function SiloPlaceable:delete()
29  if self.unloadingStation ~= nil then
30  self.unloadingStation:delete()
31  end
32
33  if self.loadingStation ~= nil then
34  self.loadingStation:delete()
35  end
36
37  for _, storage in ipairs(self.storages) do
38  g_currentMission:removeStorage(storage)
39  storage:delete()
40  end
41
42  unregisterObjectClassName(self)
43  SiloPlaceable:superClass().delete(self)
44  end
```

load

Description

Load silo

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position
float	y	z world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Return Values

boolean success success

Code

```
57  function SiloPlaceable:load(xmlFilename, x,y,z, rx,ry,rz,
    initRandom)
58  if not SiloPlaceable:superClass().load(self, xmlFilename, x,y,z,
    rx,ry,rz, initRandom) then
59  return false
60  end
61
```

```
62  local xmlFile = loadXMLFile("TempXML", xmlFilename)
63
64  self.storagePerFarm = Utils.getNoNil(getXMLBool(xmlFile,
    "placeable.storages#perFarm"), false)
65  self.foreignSilo = Utils.getNoNil(getXMLBool(xmlFile,
    "placeable.storages#foreignSilo"), self.storagePerFarm) -- Shows
    as foreign silo in the menu
66
67  self.unloadingStation = UnloadingStation:new(self.isServer,
    self.isClient)
68  self.unloadingStation:load(self.nodeId, xmlFile,
    "placeable.unloadingStation")
69  self.unloadingStation.owningPlaceable = self
70  self.unloadingStation.hasStoragePerFarm = self.storagePerFarm
71
72  self.loadingStation = LoadingStation:new(self.isServer,
    self.isClient)
73  self.loadingStation:load(self.nodeId, xmlFile,
    "placeable.loadingStation")
74  self.loadingStation.owningPlaceable = self
75  self.loadingStation.hasStoragePerFarm = self.storagePerFarm
76
77  local numStorageSets = self.storagePerFarm and
    FarmManager.MAX_NUM_FARMS or 1
78  if not g_currentMission.missionDynamicInfo.isMultiplayer then
79  numStorageSets = 1
80  end
81
82  self.storages = {}
83  local i = 0
84  while true do
85  local storageKey =
    string.format("placeable.storages.storage(%d)", i)
86  if not hasXMLProperty(xmlFile, storageKey) then
87  break
88  end
89
90  local storageNode = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, storageKey.."#node"))
91  if storageNode ~= nil then
92  for i = 1, numStorageSets do
93  local storage = Storage:new(self.isServer, self.isClient)
94  if storage:load(storageNode, xmlFile, storageKey) then
```

```
95  storage.ownerFarmId = i
96  storage.foreignSilo = self.foreignSilo -- Pass along for usage by
    prices menu
97  table.insert(self.storages, storage)
98  end
99  end
100 else
101 g_logManager:xmlWarning(xmlFilename, "Missing 'node' for storage
    '%s'!", storageKey)
102 end
103
104 i = i + 1
105 end
106
107 delete(xmlFile)
108
109 return true
110 end
```

finalizePlacement

Description

Called if placeable is placed

Definition

finalizePlacement()

Code

```
114 function SiloPlaceable:finalizePlacement()
115 SiloPlaceable:superClass().finalizePlacement(self)
116
117 self.unloadingStation:register(true)
118 g_currentMission:addUnloadingStation(self.unloadingStation)
119
120 self.loadingStation:register(true)
121 g_currentMission:addLoadingStation(self.loadingStation)
122
123 for _, storage in ipairs(self.storages) do
124 if not self.storagePerFarm then
125 storage:setOwnerFarmId(self:getOwnerFarmId(), true)
126 end
127
128 g_currentMission:addStorage(storage)
129
130 storage:register(true)
```

```
131
132 g_currentMission:addStorageToUnloadingStations(storage, {self.unloadingStation}, false)
133 g_currentMission:addStorageToLoadingStations(storage, {self.loadingStation}, false)
134 end
135 end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId	stream ID
table	connection connection

Code

```
161 function SiloPlaceable:readStream(streamId, connection)
162 SiloPlaceable:superClass().readStream(self, streamId, connection)
163 if connection:getIsServer() then
164 local unloadingStationId = NetworkUtil.readNodeObjectId(streamId)
165 self.unloadingStation:readStream(streamId, connection)
166 g_client:finishRegisterObject(self.unloadingStation, unloadingStationId)
167
168 local loadingStationId = NetworkUtil.readNodeObjectId(streamId)
169 self.loadingStation:readStream(streamId, connection)
170 g_client:finishRegisterObject(self.loadingStation, loadingStationId)
171
172 for _, storage in ipairs(self.storages) do
173 local storageId = NetworkUtil.readNodeObjectId(streamId)
174 storage:readStream(streamId, connection)
175 g_client:finishRegisterObject(storage, storageId)
176 end
177 end
178 end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
184 function SiloPlaceable:writeStream(streamId, connection)
185 SiloPlaceable:superClass().writeStream(self, streamId,
    connection)
186 if not connection:getIsServer() then
187 NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(self.unloadingStation))
188 self.unloadingStation:writeStream(streamId, connection)
189 g_server:registerObjectInStream(connection,
    self.unloadingStation)
190
191 NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(self.loadingStation))
192 self.loadingStation:writeStream(streamId, connection)
193 g_server:registerObjectInStream(connection, self.loadingStation)
194
195 for _, storage in ipairs(self.storages) do
196 NetworkUtil.writeNodeObjectId(streamId,
    NetworkUtil.getObjectId(storage))
197 storage:writeStream(streamId, connection)
198 g_server:registerObjectInStream(connection, storage)
199 end
200 end
201 end
```

collectPickObjects

Description

Collect pick objects

Definition

collectPickObjects(integer node)

Arguments

integer node node id

Code

```
218 function SiloPlaceable:collectPickObjects(node)
219 local foundNode = false
220 for _, unloadTrigger in
    ipairs(self.unloadingStation.unloadTriggers) do
221 if node == unloadTrigger.exactFillRootNode then
222 foundNode = true
223 break
224 end
225 end
```

```
226
227 if not foundNode then
228 for _, loadTrigger in ipairs(self.loadingStation.loadTriggers) do
229 if node == loadTrigger.triggerNode then
230 foundNode = true
231 break
232 end
233 end
234 end
235
236 if not foundNode then
237 SiloPlaceable:superClass().collectPickObjects(self, node)
238 end
239 end
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key, boolean resetVehicles)

Arguments

integer	xmlFile	id of xml object
string	key	key
boolean	resetVehicles	reset vehicles

Return Values

boolean success success

Code

```
247 function SiloPlaceable:loadFromXMLFile(xmlFile, key,
    resetVehicles)
248 if not SiloPlaceable:superClass().loadFromXMLFile(self, xmlFile,
    key, resetVehicles) then
249 return false
250 end
251
252 if not self.unloadingStation:loadFromXMLFile(xmlFile,
    key..".unloadingStation") then
253 return false
254 end
255
256 local i = 0
257 while true do
258 local storageKey = string.format("%s.storage(%d)", key, i)
259 if not hasXMLProperty(xmlFile, storageKey) then
```

```
260     break
261     end
262
263     local index = getXMLInt(xmlFile, storageKey.."#index")
264
265     if index ~= nil then
266     if self.storages[index] ~= nil then
267     if not self.storages[index]:loadFromXMLFile(xmlFile, storageKey)
        then
268     return false
269     end
270     else
271     g_logManager:warning("Could not load storage. Given 'index' '%d'
        is not defined!", index)
272     end
273     end
274
275     i = i + 1
276     end
277
278     return true
279     end
```

saveToXMLFile

Description

Get save attributes and nodes

Definition

saveToXMLFile(string nodeIdent)

Arguments

string nodeIdent node ident

Return Values

string attributes attributes

string nodes nodes

Code

```
286     function SiloPlaceable:saveToXMLFile(xmlFile, key, usedModNames)
287     SiloPlaceable:superClass().saveToXMLFile(self, xmlFile, key,
        usedModNames)
288
289     self.unloadingStation:saveToXMLFile(xmlFile,
        key..".unloadingStation", usedModNames)
290
291     for k, storage in ipairs(self.storages) do
292     local storageKey = string.format("%s.storage(%d)", key, k-1)
```

```
293   setXMLInt(xmlFile, storageKey .. "#index", k)
294   storage:saveToXMLFile(xmlFile, storageKey, usedModNames)
295   end
296   end
```

SolarCollectorPlaceable

Description

Class for placeable solar collectors

Parent

Placeable

new

Description

Creating placeable solar collector

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean isServer	is server
boolean isClient	is client
table	customMt custom metatable

Return Values

table instance Instance of object

Code

```
18  function SolarCollectorPlaceable:new(isServer, isClient, customMt)
19  local mt = customMt;
20  if mt == nil then
21  mt = SolarCollectorPlaceable_mt;
22  end;
23
24  local self = Placeable:new(isServer, isClient, mt);
25
26  registerObjectClassName(self, "SolarCollectorPlaceable");
27
28  self.headNode = 0;
29  self.incomePerHour = 0;
30  self.headRotationRandom = 0;
31  return self;
32  end;
```

delete

Description

Deleting solar collector

Definition

delete()

Code

```
36  function SolarCollectorPlaceable:delete()
37  unregisterObjectClassName(self);
38  g_currentMission.environment:removeHourChangeListener(self);
39  SolarCollectorPlaceable:superClass().delete(self);
40  end;
```

deleteFinal

Description

Deleting placeable solar collector final

Definition

deleteFinal()

Code

```
44  function SolarCollectorPlaceable:deleteFinal()
45  SolarCollectorPlaceable:superClass().deleteFinal(self);
46  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
52  function SolarCollectorPlaceable:readStream(streamId, connection)
53  if connection:getIsServer() then
54  self.headRotationRandom=NetworkUtil.readCompressedAngle(streamId);
55  end;
56  SolarCollectorPlaceable:superClass().readStream(self, streamId,
    connection);
57  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
63  function SolarCollectorPlaceable:writeStream(streamId, connection)
64  if not connection:getIsServer() then
```

```
65  NetworkUtil.writeCompressedAngle(streamId,
    self.headRotationRandom);
66  end;
67  SolarCollectorPlaceable:superClass().writeStream(self, streamId,
    connection);
68  end;
```

createNode

Description

Create node

Definition

createNode(string i3dFilename)

Arguments

string i3dFilename i3d file name

Return Values

boolean success success

Code

```
82  function SolarCollectorPlaceable:createNode(i3dFilename)
83  if not SolarCollectorPlaceable:superClass().createNode(self,
    i3dFilename) then
84  return false;
85  end;
86
87  if getNumOfChildren(self.nodeId) < 1 then
88  delete(self.nodeId);
89  self.nodeId = 0;
90  return false;
91  end;
92  self.headNode = getChildAt(self.nodeId, 0);
93
94  return true;
95  end;
```

load

Description

Load solar collector

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position
float	y	z world position
float	z	z world position
float	rx	rx world rotation

float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Return Values

boolean success success

Code

```
108  function SolarCollectorPlaceable:load(xmlFilename, x,y,z, rx,ry,rz, initRandom)
109  if not SolarCollectorPlaceable:superClass().load(self, xmlFilename, x,y,z, rx,ry,rz, initRandom) then
110  return false;
111  end;
112
113  return true;
114  end;
```

finalizePlacement

Description

Called if placeable is placed

Definition

finalizePlacement()

Code

```
118  function SolarCollectorPlaceable:finalizePlacement()
119  SolarCollectorPlaceable:superClass().finalizePlacement(self);
120  g_currentMission.environment:addHourChangeListener(self);
121  end
```

initPose

Description

Initialize pose

Definition

initPose(float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

float	x	x world position
float	y	y world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Code

```
132  function SolarCollectorPlaceable:initPose(x,y,z, rx,ry,rz, initRandom)
133  SolarCollectorPlaceable:superClass().initPose(self, x,y,z, rx,ry,rz, initRandom);
```

```
134
135 local rotVariation = 0.1;
136 self.headRotationRandom = math.rad(75-90) + math.random() *
    2*rotVariation - rotVariation; -- 75-90 = rotate to light (75 y
    rot) x axis
137
138 self:updateHeadRotation();
139 end;
```

updateHeadRotation

Description

Updates the rotation of the head

Definition

updateHeadRotation()

Code

```
143 function SolarCollectorPlaceable:updateHeadRotation()
144
145 local headRotation = math.rad(75-90);
146 if g_currentMission.environment.sunLightId ~= nil then
147 local dx,_,dz =
    localDirectionToWorld(g_currentMission.environment.sunLightId,
    0,0,1);
148 headRotation = math.atan2(dx, dz);
149 end
150 headRotation = headRotation + self.headRotationRandom;
151
152 local dx,_,dz = worldDirectionToLocal(self.nodeId,
    math.sin(headRotation),0,math.cos(headRotation));
153 setDirection(self.headNode, dx,0,dz, 0,1,0);
154 end;
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key, boolean resetVehicles)

Arguments

integer	xmlFile	id of xml object
string	key	key
boolean	resetVehicles	reset vehicles

Return Values

boolean success success

Code

```
162 function SolarCollectorPlaceable:loadFromXMLFile(xmlFile, key,
    resetVehicles)
```

```
163 local headRotationRandom = getXMLFloat(xmlFile,
    key.."#headRotationRandom");
164 if headRotationRandom == nil then
165 return false;
166 end;
167
168 self.headRotationRandom = headRotationRandom;
169
170 if not SolarCollectorPlaceable:superClass().loadFromXMLFile(self,
    xmlFile, key, resetVehicles) then
171 return false;
172 end;
173
174 return true;
175 end;
```

hourChanged

Description

Called if hour changed

Definition

hourChanged()

Code

```
188 function SolarCollectorPlaceable:hourChanged()
189 if self.isServer then
190 g_currentMission:addMoney(self.incomePerHour,
    self:getOwnerFarmId(), "propertyIncome");
191 g_currentMission:addMoneyChange(self.incomePerHour,
    self:getOwnerFarmId(),
    EconomyManager.MONEY_TYPE_PROPERTY_INCOME);
192 end
193 self:updateHeadRotation();
194 end
```

WindTurbinePlaceable

Description

Class for placeable wind turbines

Parent

Placeable

new

Description

Creating placeable wind turbine

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean isServer is server
boolean isClient is client
table customMt custom metatable

Return Values

table instance Instance of object

Code

```
18  function WindTurbinePlaceable:new(isServer, isClient, customMt)
19  local mt = customMt;
20  if mt == nil then
21  mt = WindTurbinePlaceable_mt;
22  end;
23
24  local self = Placeable:new(isServer, isClient, mt);
25
26  registerObjectClassName(self, "WindTurbinePlaceable");
27
28  self.rotationNode = 0;
29  self.headNode = 0;
30  self.incomePerHour = 0;
31  return self;
32  end;
```

delete

Description

Deleting placeable wind turbine

Definition

delete()

Code

```
36  function WindTurbinePlaceable:delete()
37  unregisterObjectClassName(self);
38  g_currentMission.environment:removeHourChangeListener(self);
39  WindTurbinePlaceable:superClass().delete(self);
40  end;
```

deleteFinal

Description

Deleting placeable wind turbine

Definition

deleteFinal()

Code

```
44  function WindTurbinePlaceable:deleteFinal()
45  WindTurbinePlaceable:superClass().deleteFinal(self);
46  end;
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
52  function WindTurbinePlaceable:readStream(streamId, connection)
53  if connection:getIsServer() then
54  self.headRotation=NetworkUtil.readCompressedAngle(streamId);
55  end;
56  WindTurbinePlaceable:superClass().readStream(self, streamId,
    connection);
57  end;
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
63  function WindTurbinePlaceable:writeStream(streamId, connection)
64  if not connection:getIsServer() then
65  NetworkUtil.writeCompressedAngle(streamId, self.headRotation);
66  end;
67  WindTurbinePlaceable:superClass().writeStream(self, streamId,
    connection);
68  end;
```

createNode

Description

Create node

Definition

createNode(string i3dFilename)

Arguments

string i3dFilename i3d file name

Return Values

boolean success success

Code

```
82  function WindTurbinePlaceable:createNode(i3dFilename)
```

```
83  if not WindTurbinePlaceable:superClass().createNode(self, i3dFilename) then
84  return false;
85  end;
86
87  if getNumOfChildren(self.nodeId) < 1 then
88  delete(self.nodeId);
89  self.nodeId = 0;
90  return false;
91  end;
92  self.headNode = getChildAt(self.nodeId, 0);
93  if getNumOfChildren(self.headNode) < 1 then
94  delete(self.nodeId);
95  self.nodeId = 0;
96  return false;
97  end;
98  self.rotationNode = getChildAt(self.headNode, 0);
99
100 return true;
101 end;
```

load

Description

Load wind turbine

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position
float	y	z world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Return Values

boolean success success

Code

```
114 function WindTurbinePlaceable:load(xmlFilename, x,y,z, rx,ry,rz, initRandom)
115 if not WindTurbinePlaceable:superClass().load(self, xmlFilename, x,y,z, rx,ry,rz, initRandom) then
116 return false;
```

```
117 end;
118
119 return true;
120 end;
```

finalizePlacement

Description

Called if placeable is placed

Definition

finalizePlacement()

Code

```
124 function WindTurbinePlaceable:finalizePlacement()
125 WindTurbinePlaceable:superClass().finalizePlacement(self);
126 g_currentMission.environment:addHourChangeListener(self);
127 end
```

initPose

Description

Initialize pose

Definition

initPose(float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

float	x	x world position
float	y	y world position
float	z	z world position
float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Code

```
138 function WindTurbinePlaceable:initPose(x,y,z, rx,ry,rz,
    initRandom)
139 WindTurbinePlaceable:superClass().initPose(self, x,y,z, rx,ry,rz,
    initRandom);
140
141 if initRandom == nil or initRandom == true then
142 local rotVariation = 0.2;
143 self.headRotation = 0.7 + math.random() * 2*rotVariation -
    rotVariation;
144 end;
145 rotate(self.rotationNode, 0,0,math.random()*math.pi*2);
146 self:updateHeadRotation();
147 end;
```

updateHeadRotation

Description

Updating head rotation

Definition

updateHeadRotation()

Code

```
151 function WindTurbinePlaceable:updateHeadRotation()
152 local dx,_,dz = worldDirectionToLocal(self.nodeId,
    math.sin(self.headRotation),0,math.cos(self.headRotation));
153 setDirection(self.headNode, dx,0,dz, 0,1,0);
154 end;
```

loadFromXMLFile

Description

Loading from attributes and nodes

Definition

loadFromXMLFile(integer xmlFile, string key, boolean resetVehicles)

Arguments

integer	xmlFile	id of xml object
string	key	key
boolean	resetVehicles	reset vehicles

Return Values

boolean success success

Code

```
162 function WindTurbinePlaceable:loadFromXMLFile(xmlFile, key,
    resetVehicles)
163 local headRotation = getXMLFloat(xmlFile, key.."#headRotation");
164 if headRotation == nil then
165 return false;
166 end;
167
168 self.headRotation = headRotation;
169
170 if not WindTurbinePlaceable:superClass().loadFromXMLFile(self,
    xmlFile, key, resetVehicles) then
171 return false;
172 end;
173
174 return true;
175 end;
```

update

Description

Update

Definition

update(float dt)

Arguments

float dt time since last call in ms

Code

```
186  function WindTurbinePlaceable:update(dt)
187  if self.rotationNode ~= 0 then
188  rotate(self.rotationNode, 0,0,-0.0025*dt);
189  end;
190  end;
```

hourChanged

Description

Called if hour changed

Definition

hourChanged()

Code

```
194  function WindTurbinePlaceable:hourChanged()
195  if self.isServer then
196  g_currentMission:addMoney(self.incomePerHour,
     self:getOwnerFarmId(), "propertyIncome");
197  g_currentMission:addMoneyChange(self.incomePerHour,
     self:getOwnerFarmId(),
     EconomyManager.MONEY_TYPE_PROPERTY_INCOME);
198  end
199  end
```

WoodSellStationPlaceable

Description

new

Description

Creating woodsell station

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean isServer	is server
boolean isClient	is client
table	customMt custom metatable

Return Values

table instance Instance of object

Code

```
26  function WoodSellStationPlaceable:new(isServer, isClient,
    customMt)
27  local self = Placeable:new(isServer, isClient, customMt or
    WoodSellStationPlaceable_mt)
28
29  registerObjectClassName(self, "WoodSellStationPlaceable")
30
```

```
31  self.woodInTrigger = {}
32
33  return self
34  end
```

delete

Description

Deleting trigger

Definition

delete()

Code

```
38  function WoodSellStationPlaceable:delete()
39  if self.mapHotspot ~= nil then
40  g_currentMission:removeMapHotspot(self.mapHotspot)
41  self.mapHotspot:delete()
42  end
43
44  if self.woodSellTrigger ~= nil then
45  removeTrigger(self.woodSellTrigger)
46  self.woodSellTrigger = nil
47  end
48
49  if self.sellTrigger ~= nil then
50  g_currentMission:removeActivatableObject(self)
51  removeTrigger(self.sellTrigger)
52  self.sellTrigger = nil
53  end
54
55  unregisterObjectClassName(self)
56  WoodSellStationPlaceable:superClass().delete(self)
57  end
```

load

Description

Load woodsell station

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, boolean initRandom)

Arguments

string	xmlFilename	xml file name
float	x	x world position
float	y	z world position
float	z	z world position

float	rx	rx world rotation
float	ry	ry world rotation
float	rz	rz world rotation
boolean	initRandom	initialize random

Return Values

boolean success success

Code

```
70  function WoodSellStationPlaceable:load(xmlFilename, x,y,z, rx,ry,rz, initRandom)
71  if not WoodSellStationPlaceable:superClass().load(self, xmlFilename, x,y,z, rx,ry,rz, initRandom) then
72  return false
73  end
74
75  local xmlFile = loadXMLFile("TempXML", xmlFilename)
76
77  self.appearsOnPDA = Utils.getNoNil(getXMLBool(xmlFile, "placeable.woodSellStation#appearsOnPDA"), false)
78  local rawName = Utils.getNoNil(getXMLString(xmlFile, "placeable.woodSellStation#stationName"), "WoodSellStation")
79  self.stationName = g_i18n:convertText(rawName) -- returns input if it cannot be resolved
80
81  local woodSellTrigger = I3DUtil.indexToObject(self.nodeId, getXMLString(xmlFile, "placeable.woodSellStation#triggerNode"))
82  if woodSellTrigger == nil then
83  g_logManager:xmlWarning(xmlFilename, "Missing wood trigger node in 'placeable.woodSellStation#triggerNode'!")
84  delete(xmlFile)
85  return false
86  end
87
88  local colMask = getCollisionMask(woodSellTrigger)
89  if bitAND(SplitTypeManager.COLLISIONMASK_TRIGGER, colMask) == 0 then
90  g_logManager:xmlWarning(xmlFilename, "Invalid collision mask for wood trigger 'placeable.woodSellStation#triggerNode'. Bit 24 needs to be set!")
91  delete(xmlFile)
92  return false
93  end
94
95  addTrigger(woodSellTrigger, "woodTriggerCallback", self)
```

```
96  self.woodSellTrigger = woodSellTrigger
97
98
99  local sellTrigger = I3DUtil.indexToObject(self.nodeId,
    getXMLString(xmlFile, "placeable.woodSellStation#sellTrigger"))
100 if sellTrigger == nil then
101 g_logManager:xmlWarning(xmlFilename, "Missing sell trigger in
    'placeable.woodSellStation#sellTrigger'!")
102 delete(xmlFile)
103 return false
104 end
105
106 colMask = getCollisionMask(sellTrigger)
107 if bitAND(Player.COLLISIONMASK_TRIGGER, colMask) == 0 then
108 g_logManager:xmlWarning(xmlFilename, "Invalid collision mask for
    sell trigger 'placeable.woodSellStation#triggerNode'. Bit 20
    needs to be set!")
109 delete(xmlFile)
110 return false
111 end
112
113 if self.appearsOnPDA then
114 local mapPosition = self.woodSellTrigger
115 local mapPositionIndex = getUserAttribute(self.woodSellTrigger,
    "mapPositionIndex")
116 if mapPositionIndex ~= nil then
117 mapPosition = I3DUtil.indexToObject(self.woodSellTrigger,
    mapPositionIndex)
118 if mapPosition == nil then
119 mapPosition = self.woodSellTrigger
120 end
121 end
122 local filenames = nil
123 local imageUVs = getNormalizedUVs(MapHotspot.UV.SELLING_POINT)
124 local x, _, z = getWorldTranslation(mapPosition)
125
126 local hotspotTextOffset = Utils.getNoNil(getXMLString(xmlFile,
    "placeable.woodSellStation#hotspotTextOffset"), "0px 0px")
127
128 self.mapHotspot = MapHotspot:new("woodSellStation",
    MapHotspot.CATEGORY_TRIGGER)
129 self.mapHotspot:setText(self.stationName)
```

```
130 self.mapHotspot:setBorderedImage(nil, imageUVs, {0.9559, 0.5647, 0.0423, 1})
131 self.mapHotspot:setWorldPosition(x, z)
132 self.mapHotspot:setRawTextOffset(hotspotTextOffset)
133
134 g_currentMission:addMapHotspot(self.mapHotspot)
135 end
136
137 addTrigger(sellTrigger, "woodSellTriggerCallback", self)
138 self.sellTrigger = sellTrigger
139
140 self.activateText = g_i18n:getText("action_sellWood")
141 self.objectActivated = false
142
143 self.updateEventListeners = {}
144
145 delete(xmlFile)
146
147 return true
148 end
```

addUpdateEventListener

Description

Add listener to update listeners

Definition

addUpdateEventListener()

Code

```
205 function WoodSellStationPlaceable:addUpdateEventListener(listener)
206 if listener ~= nil then
207 self.updateEventListeners[listener] = listener
208 end
209 end
```

removeUpdateEventListener

Description

Remove listener from update listeners

Definition

removeUpdateEventListener()

Code

```
213 function WoodSellStationPlaceable:removeUpdateEventListener(listener)
214 if listener ~= nil then
```

```
215  self.updateEventListeners[listener] = nil
216  end
217  end
```

getIsActivatable

Description

Returns true if wood can be sold

Definition

getIsActivatable()

Return Values

boolean isActivateable is activateable

Code

```
275  function WoodSellStationPlaceable:getIsActivatable()
276  return g_currentMission.controlPlayer
277  end
```

onActivateObject

Description

Called on activate object

Definition

onActivateObject()

Code

```
285  function WoodSellStationPlaceable:onActivateObject()
286  g_currentMission:addActivatableObject(self)
287  self.objectActivated = true
288  self:sellWood(g_currentMission:getFarmId())
289  end
```

BasketTrigger

Description

Class for basket triggers

onCreate

Description

On create basket trigger

Definition

onCreate(integer id)

Arguments

integer id id of trigger node

Code

```
17  function BasketTrigger:onCreate(id)
18  local trigger = BasketTrigger:new();
19  if trigger:load(id) then
20  g_currentMission:addNonUpdateable(trigger);
21  else
22  trigger:delete();
```

```
23  end
24  end;
```

new

Description

Creating basket trigger object

Definition

new(table mt)

Arguments

table mt custom metatable (optional)

Return Values

table instance instance of basket trigger object

Code

```
30  function BasketTrigger:new(mt)
31  local self = {};
32  if mt == nil then
33  mt = BasketTrigger_mt;
34  end
35  setmetatable(self, mt);
36
37  self.triggerId = 0;
38  self.nodeId = 0;
39
40  return self;
41  end;
```

load

Description

Load basket trigger

Definition

load(integer nodeId)

Arguments

integer nodeId id of node

Return Values

boolean success success

Code

```
47  function BasketTrigger:load(nodeId)
48  self.nodeId = nodeId;
49
50  self.triggerId = I3DUtil.indexToObject(nodeId,
    getUserAttribute(nodeId, "triggerIndex"));
51  if self.triggerId == nil then
52  self.triggerId = nodeId;
53  end
```

```
54  addTrigger(self.triggerId, "triggerCallback", self);
55
56  self.triggerObjects = {};
57
58  self.isEnabled = true;
59
60  return true;
61  end;
```

delete

Description

Delete basket trigger

Definition

delete()

Code

```
65  function BasketTrigger:delete()
66  removeTrigger(self.triggerId);
67  end;
```

triggerCallback

Description

Trigger callback

Definition

triggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

Code

```
76  function BasketTrigger:triggerCallback(triggerId, otherActorId, onEnter, onStay, otherShapeId)
77  if self.isEnabled then
78
79  if onEnter then
80  local object = g_currentMission:getNodeObject(otherActorId);
81  if object.thrownFromPosition ~= nil then
82  self.triggerObjects[otherActorId] = true;
83  end
84
85  elseif onLeave then
86  if self.triggerObjects[otherActorId] then
```

```
87  self.triggerObjects[otherActorId] = false;
88
89  local object = g_currentMission:getNodeObject(otherActorId);
90  local x,y,z = worldToLocal(self.triggerId,
    object.thrownFromPosition[1],object.thrownFromPosition[2],object.thrownFr
91  local dist = MathUtil.vector3Length(x,y,z);
92  end
93  end;
94  end;
95  end;
```

FillPlane

Description

new

Description

Creates a new instance of the class

Definition

new(table customMt)

Arguments

table customMt meta table

Return Values

table self returns the instance

Code

```
23  function FillPlane:new(customMt)
24  if customMt == nil then
25  customMt = FillPlane_mt
26  end
27  local self = {}
28  setmetatable(self, customMt)
29
30  self:initDataStructures()
31
32  return self
33  end
```

delete

Description

Destructor

Definition

delete()

Code

```
37  function FillPlane:delete()
38  end
```

initDataStructures

Description

Init class members

Definition

initDataStructures()

Code

```
42  function FillPlane:initDataStructures()
43  self.node = nil
44  self.maxCapacity = 0
45  self.moveMinY = 0
46  self.moveMaxY = 0
47  self.loaded = false
48  self.colorChange = false
49  end
```

load

Description

Loads fill plane

Definition

load(table rootNode, string xmlFile, string xmlNode)

Arguments

table	rootNode of the object
string xmlFile	file to read
string xmlNode	xmlNode to read from

Code

```
56  function FillPlane:load(rootNode, xmlFile, xmlNode)
57  local fillPlaneNodeStr =
    XMLUtil.getValueFromXMLFileOrUserAttribute(xmlFile, xmlNode,
    "node", getXMLString, rootNode)
58
59  if fillPlaneNodeStr ~= nil then
60  local fillPlaneNode = I3DUtil.indexToObject(rootNode,
    fillPlaneNodeStr)
61
62  if fillPlaneNode ~= nil then
63  self.node = fillPlaneNode
64  self.moveMinY =
    Utils.getNoNil(XMLUtil.getValueFromXMLFileOrUserAttribute(xmlFile,
    xmlNode, "minY", getXMLFloat, rootNode), 0)
65  self.moveMaxY =
    Utils.getNoNil(XMLUtil.getValueFromXMLFileOrUserAttribute(xmlFile,
    xmlNode, "maxY", getXMLFloat, rootNode), 0)
66  self.colorChange =
    Utils.getNoNil(XMLUtil.getValueFromXMLFileOrUserAttribute(xmlFile,
    xmlNode, "colorChange", getXMLBool, rootNode), false)
```

```
67  assert(self.moveMinY <= self.moveMaxY)
68  self.loaded = self.node ~= nil
69  local x, _, z = getTranslation(self.node)
70  setTranslation(self.node, x, self.moveMinY, z)
71  end
72  end
73  end
```

setState

Description

Changes fill levels visuals

Definition

setState(table instance)

Arguments

table instance target to check fillLevel

Return Values

bool true if level has changed

Code

```
79  function FillPlane:setState(state)
80  if self.loaded then
81  local delta = self.moveMaxY - self.moveMinY
82  local y = math.min(self.moveMinY + delta * state, self.moveMaxY)
83  local x, oldY, z = getTranslation(self.node)
84  setTranslation(self.node, x, y, z)
85
86  return oldY ~= y
87  end
88
89  return false
90  end
```

setColorScale

Description

Sets fill plane color shader

Definition

setColorScale(float[] a)

Arguments

float[] a float array for r, g, b

Code

```
95  function FillPlane:setColorScale(colorScale)
96  if self.loaded then
97  setShaderParameter(self.node, "colorScale", colorScale[1],
    colorScale[2], colorScale[3], 0, false)
98  end
```

```
99  end
```

FillTrigger

Description

Class for fill triggers

onCreate

Description

On create fill trigger

Definition

onCreate(integer id)

Arguments

integer id id of trigger node

Code

```
13  function FillTrigger:onCreate(id)
14  g_currentMission:addNonUpdateable(FillTrigger:new(id))
15  end
```

new

Description

Create fill trigger object

Definition

new(integer id, table sourceObject, integer fillUnitIndex, table customMt)

Arguments

integer	id	id of trigger node
table	sourceObject	sourceObject
integer	fillUnitIndex	fillUnitIndex
table	customMt	custom metatable (optional)

Return Values

table instance instance of gas station trigger

Code

```
24  function FillTrigger:new(id, sourceObject, fillUnitIndex,
    fillLitersPerSecond, defaultFillType, customMt)
25  local self = {}
26  setmetatable(self, customMt or FillTrigger_mt)
27
28  self.customEnvironment = g_currentMission.loadingMapModName
29
30  self.triggerId = id
31  addTrigger(id, "fillTriggerCallback", self)
32
33  -- place sound at the same position as the trigger
34  self.soundNode = createTransformGroup("fillTriggerSoundNode")
35  link(getParent(id), self.soundNode)
36  setTranslation(self.soundNode, getTranslation(id))
```

```
37
38  self.sourceObject = sourceObject
39  self.vehiclesTriggerCount = {}
40  self.fillUnitIndex = fillUnitIndex
41  self.fillLitersPerSecond = fillLitersPerSecond
42  self.appearsOnPDA = Utils.getNoNil(getUserAttribute(id,
    "appearsOnPDA"), true)
43  self.isEnabled = true
44
45  self.fillTypeIndex = FillType.DIESEL
46
47  if self.appearsOnPDA and sourceObject == nil then
48  local mapPosition = id
49  local mapPositionIndex = getUserAttribute(id, "mapPositionIndex")
50  if mapPositionIndex ~= nil then
51  mapPosition = I3DUtil.indexToObject(id, mapPositionIndex)
52  if mapPosition == nil then
53  mapPosition = id
54  end
55  end
56
57  local x, _, z = getWorldTranslation(mapPosition)
58
59  local fullViewName = Utils.getNoNil(getUserAttribute(id,
    "stationName"), "map_fuelStation")
60  if g_i18n:hasText(fullViewName, self.customEnvironment) then
61  fullViewName = g_i18n:getText(fullViewName,
    self.customEnvironment)
62  end
63
64  self.mapHotspot = MapHotspot:new("fuelStation",
    MapHotspot.CATEGORY_DEFAULT)
65  self.mapHotspot:setText(fullViewName)
66  self.mapHotspot:setWorldPosition(x, z)
67  self.mapHotspot:setBorderedImage(nil,
    getNormalizedUVs(MapHotspot.UV.GAS_STATION))
68
69  g_currentMission:addMapHotspot(self.mapHotspot)
70  end
71
72  self.moneyChangeId = getMoneyTypeId()
```

```
73
74  return self
75  end
```

delete

Description

Delete fill trigger

Definition

delete()

Code

```
79  function FillTrigger:delete()
80  -- remove the gas stations from all vehicles that are triggered by
    this trigger
81  for vehicle,count in pairs(self.vehiclesTriggerCount) do
82  if count > 0 then
83  if vehicle.removeFillUnitTrigger ~= nil then
84  vehicle:removeFillUnitTrigger(self)
85  end
86  end
87  end
88
89  if self.mapHotspot ~= nil then
90  g_currentMission:removeMapHotspot(self.mapHotspot)
91  self.mapHotspot:delete()
92  end
93
94  g_soundManager:deleteSample(self.sample)
95
96  removeTrigger(self.triggerId)
97  end
```

onVehicleDeleted

Description

Called if vehicle gets out of trigger

Definition

onVehicleDeleted(table vehicle)

Arguments

table vehicle vehicle

Code

```
102 function FillTrigger:onVehicleDeleted(vehicle)
103 self.vehiclesTriggerCount[vehicle] = nil
104 g_currentMission:showMoneyChange(self.moneyChangeId,
    g_i18n:getText("finance_purchaseFuel"))
```

```
105 end
```

fillVehicle

Description

Fill vehicle

Definition

fillVehicle(table vehicle, float delta)

Arguments

table vehicle vehicle to fill

float delta delta

Return Values

float delta real delta

Code

```
112 function FillTrigger:fillVehicle(vehicle, delta, dt)
113 if self.fillLitersPerSecond ~= nil then
114 delta = math.max(delta, self.fillLitersPerSecond*0.001*dt)
115 end
116
117 local farmId = vehicle:getActiveFarm()
118
119 if self.sourceObject ~= nil then
120 local sourceFuelFillLevel =
    self.sourceObject:getFillUnitFillLevel(self.fillUnitIndex)
121 if sourceFuelFillLevel > 0 and
    g_currentMission.accessHandler:canFarmAccess(farmId,
    self.sourceObject) then
122 delta = math.min(delta, sourceFuelFillLevel)
123 if delta <= 0 then
124 return 0
125 end
126 else
127 return 0
128 end
129 end
130
131 local fillType = self:getCurrentFillType()
132
133 local fillUnitIndex =
    vehicle:getFirstValidFillUnitToFill(fillType)
134 if fillUnitIndex == nil then
135 return 0
136 end
137
```

```
138 delta = vehicle:addFillUnitFillLevel(farmId, fillUnitIndex,
    delta, fillType, ToolType.TRIGGER, nil)
139
140 if delta > 0 then
141 if self.sourceObject ~= nil then
142 self.sourceObject:addFillUnitFillLevel(farmId,
    self.fillUnitIndex, -delta, fillType, ToolType.TRIGGER, nil)
143 else
144 local price = delta *
    g_currentMission.economyManager:getPricePerLiter(fillType)
145 g_farmManager:getFarmById(farmId).stats:updateStats("expenses",
    price)
146
147 local userId = g_currentMission:getServerUserId()
148 local user =
    g_currentMission:getUserByConnection(vehicle:getOwner())
149 if user ~= nil then
150 userId = user.userId
151 end
152
153 g_currentMission:addMoney(-price, farmId, "purchaseFuel")
154 g_currentMission:addMoneyChange(-price, farmId,
    self.moneyChangeId)
155 end
156 end
157
158 return delta
159 end
```

getIsActivatable

Description

Returns true if is activateable

Definition

getIsActivatable(table vehicle)

Arguments

table vehicle vehicle

Return Values

boolean isActivateable is activateable

Code

```
165 function FillTrigger:getIsActivatable(vehicle)
166 if self.sourceObject ~= nil then
167 if self.sourceObject:getFillUnitFillLevel(self.fillUnitIndex) > 0 and
    g_currentMission.accessHandler:canFarmAccess(vehicle:getActiveFarm(),
    self.sourceObject) then
```

```
168 return true
169 end
170 end
171
172 return false
173 end
```

fillTriggerCallback

Description

Trigger callback

Definition

fillTriggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

Code

```
182 function FillTrigger:fillTriggerCallback(triggerId, otherId, onEnter, onLeave, onStay)
183 if self.isEnabled and (onEnter or onLeave) then
184 local vehicle = g_currentMission:getNodeObject(otherId)
185 if vehicle ~= nil and vehicle.addFillUnitTrigger ~= nil and vehicle.removeFillUnitTrigger ~= nil and vehicle ~= self and vehicle ~= self.sourceObject then
186 local count = Utils.getNoNil(self.vehiclesTriggerCount[vehicle], 0)
187 if onEnter then
188 local fillType = self:getCurrentFillType()
189 local fillUnitIndex = vehicle:getFirstValidFillUnitToFill(fillType)
190 if fillUnitIndex ~= nil then
191 self.vehiclesTriggerCount[vehicle] = count + 1
192
193 if count == 0 then
194 vehicle:addFillUnitTrigger(self, fillType, fillUnitIndex)
195 end
196 end
197 else
198 self.vehiclesTriggerCount[vehicle] = count - 1
199 if count <= 1 then
200 self.vehiclesTriggerCount[vehicle] = nil
```

```
201 vehicle:removeFillUnitTrigger(self)
202 g_currentMission:showMoneyChange(self.moneyChangeId,
    g_i18n:getText("finance_purchaseFuel"))
203 end
204 end
205 end
206 end
207 end
```

InsideBuildingTrigger

Description

Class for InsideBuildingTriggers

onCreate

Description

On create InsideBuildingTrigger

Definition

onCreate(integer id)

Arguments

integer id id of trigger node

Code

```
15 function InsideBuildingTrigger.onCreate(_, id)
16 local trigger = InsideBuildingTrigger:new()
17 if trigger:load(id) then
18 g_currentMission:addNonUpdateable(trigger)
19 else
20 trigger:delete()
21 end
22 end
```

new

Description

Creating InsideBuildingTrigger object

Definition

new(table customMt)

Arguments

table customMt custom metatable (optional)

Return Values

table instance instance of basket trigger object

Code

```
28 function InsideBuildingTrigger:new(customMt)
29 local self = {}
30 setmetatable(self, customMt or InsideBuildingTrigger_mt)
31
32 self.triggerId = 0
```

```
33  self.nodeId = 0
34
35  return self
36  end
```

load

Description

Load InsideBuildingTrigger

Definition

load(integer nodeId)

Arguments

integer nodeId id of node

Return Values

boolean success success

Code

```
42  function InsideBuildingTrigger:load(nodeId)
43  self.nodeId = nodeId
44
45  self.triggerId = I3DUtil.indexToObject(nodeId,
    getUserAttribute(nodeId, "triggerIndex"))
46  if self.triggerId == nil then
47  self.triggerId = nodeId
48  end
49  addTrigger(self.triggerId, "insideBuildingTriggerCallback", self)
50
51  self.isEnabled = true
52
53  return true
54  end
```

delete

Description

Delete InsideBuildingTrigger

Definition

delete()

Code

```
58  function InsideBuildingTrigger:delete()
59  removeTrigger(self.triggerId)
60  end
```

insideBuildingTriggerCallback

Description

Trigger callback

Definition

insideBuildingTriggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

Code

```
69  function InsideBuildingTrigger:insideBuildingTriggerCallback(triggerId, otherActorId, onEnter, onLeave, onStay, otherShapeId)
70  -- log(g_currentMission.player.rootNode, triggerId, otherActorId, onEnter, onLeave, onStay, otherShapeId)
71  if g_currentMission.player ~= nil and g_currentMission.player.rootNode == otherActorId then
72  if self.isEnabled then
73  if onEnter then
74  g_currentMission:setIsInsideBuilding(true)
75  elseif onLeave then
76  g_currentMission:setIsInsideBuilding(false)
77  end
78  end
79  end
80  end
```

LoadTrigger

Description

farmIdForFillableObject

Description

Get the farm id for given object. If none can be found, SPECTATOR is used.

Definition

farmIdForFillableObject()

LoanTrigger

Description

Class for loan triggers

onCreate

Description

On create loan trigger

Definition

onCreate(integer id)

Arguments

integer id id of trigger node

Code

```
15  function LoanTrigger:onCreate(id)
16  g_currentMission:addNonUpdateable(LoanTrigger:new(id))
17  end
```

new

Description

Create loan trigger object

Definition

new(integer name)

Arguments

integer name id of trigger node

Return Values

table instance instance

Code

```
23  function LoanTrigger:new(name)
24  local self = {}
25  setmetatable(self, LoanTrigger_mt)
26
27  if g_currentMission:getIsClient() then
28  self.triggerId = name
29  addTrigger(name, "triggerCallback", self)
30  end
31
32  self.loanSymbol = getChildAt(name, 0)
33
34  self.activateText = g_i18n:getText("action_checkFinances")
35
36  self.isEnabled = true
37  self.objectActivated = false
38
39  g_messageCenter:subscribe(MessageType.PLAYER_FARM_CHANGED,
    self.playerFarmChanged, self)
40
41  self:updateIconVisibility()
42
43  return self
44  end
```

delete

Description

Delete loan trigger

Definition

delete()

Code

```
48  function LoanTrigger:delete()
49  g_messageCenter:unsubscribeAll(self)
50
51  if self.triggerId ~= nil then
52  removeTrigger(self.triggerId)
53  end
54  self.loanSymbol = nil
55  g_currentMission:removeActivatableObject(self)
56  end
```

getIsActivatable

Description

Returns true if is activateable

Definition

getIsActivatable()

Return Values

boolean isActivateable is activateable

Code

```
61  function LoanTrigger:getIsActivatable()
62  return self.isEnabled and g_currentMission.controlPlayer and
    g_currentMission:getFarmId() ~= FarmManager.SPECTATOR_FARM_ID
63  end
```

onActivateObject

Description

Called on activate object

Definition

onActivateObject()

Code

```
71  function LoanTrigger:onActivateObject()
72  g_gui:showGui("InGameMenu")
73  g_messageCenter:publish(MessageType.GUI_INGAME_OPEN_FINANCES_SCREEN)
74
75  g_currentMission:addActivatableObject(self)
76  self.objectActivated = true
77  end
```

triggerCallback

Description

Trigger callback

Definition

triggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

Code

```
86  function LoanTrigger:triggerCallback(triggerId, otherId, onEnter, onLeave, onStay)
87  if self.isEnabled and (not g_isPresentationVersion or g_isPresentationVersionShopEnabled) and g_currentMission.missionInfo:isa(FSCareerMissionInfo) then
88  if onEnter or onLeave then
89  if g_currentMission.player ~= nil and otherId == g_currentMission.player.rootNode then
90  if onEnter then
91  if not self.objectActivated then
92  g_currentMission:addActivatableObject(self)
93  self.objectActivated = true
94  end
95  else
96  if self.objectActivated then
97  g_currentMission:removeActivatableObject(self)
98  self.objectActivated = false
99  end
100 end
101 end
102 end
103 end
104 end
```

updateIconVisibility

Description

Turn the icon on or off depending on the current game and the players farm

Definition

updateIconVisibility()

RainDropFactorTrigger

Description

Class for RainDropFactorTriggers

onCreate

Description

On create RainDropFactorTrigger

Definition

onCreate(integer id)

Arguments

integer id id of trigger node

Code

```
15  function RainDropFactorTrigger:onCreate(id)
16  local trigger = RainDropFactorTrigger:new();
17  if trigger:load(id) then
18  g_currentMission:addNonUpdateable(trigger);
19  else
20  trigger:delete();
21  end
22  end;
```

new

Description

Creating RainDropFactorTrigger object

Definition

new(table mt)

Arguments

table mt custom metatable (optional)

Return Values

table instance instance of basket trigger object

Code

```
28  function RainDropFactorTrigger:new(mt)
29  local self = {};
30  if mt == nil then
31  mt = RainDropFactorTrigger_mt;
32  end
33  setmetatable(self, mt);
34
35  self.triggerId = 0;
36  self.nodeId = 0;
37
38  return self;
39  end;
```

load

Description

Load RainDropFactorTrigger

Definition

load(integer nodeId)

Arguments

integer nodeId id of node

Return Values

boolean success success

Code

```
45  function RainDropFactorTrigger:load(nodeId)
46  self.nodeId = nodeId;
47
48  self.triggerId = I3DUtil.indexToObject(nodeId,
    getUserAttribute(nodeId, "triggerIndex"));
49  if self.triggerId == nil then
50  self.triggerId = nodeId;
51  end
52  addTrigger(self.triggerId, "triggerCallback", self);
53
54  self.triggerObjects = {};
55
56  self.isEnabled = true;
57
58  return true;
59  end;
```

delete

Description

Delete RainDropFactorTrigger

Definition

delete()

Code

```
63  function RainDropFactorTrigger:delete()
64  removeTrigger(self.triggerId);
65  end;
```

triggerCallback

Description

Trigger callback

Definition

triggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

Code

```
74  function RainDropFactorTrigger:triggerCallback(triggerId,
    otherActorId, onEnter, onLeave, onStay, otherShapeId)
75  if self.isEnabled then
```

```
76  if onEnter then
77  if g_currentMission.environment ~= nil then
78  -- g_currentMission.environment.globalRainDropFactor = 0.0;
79  end
80  elseif onLeave then
81  if g_currentMission.environment ~= nil then
82  -- g_currentMission.environment.globalRainDropFactor = 1.0;
83  end
84  end;
85  end;
86  end;
```

ShopTrigger

Description

Class for shop triggers to open shop gui

onCreate

Description

On create shop trigger

Definition

onCreate(integer id)

Arguments

integer id trigger node id

Code

```
15  function ShopTrigger:onCreate(id)
16  g_currentMission:addNonUpdateable(ShopTrigger:new(id))
17  end
```

new

Description

Creating shop trigger object

Definition

new(integer name)

Arguments

integer name trigger node id

Return Values

table instance instance of object

Code

```
23  function ShopTrigger:new(name)
24  local self = {}
25  setmetatable(self, ShopTrigger_mt)
26
27  if g_currentMission:getIsClient() then
28  self.triggerId = name
29  addTrigger(name, "triggerCallback", self)
```

```
30  end
31
32  self.shopSymbol = getChildAt(name, 0)
33  self.shopPlayerSpawn = getChildAt(name, 1)
34
35  self.objectActivated = false
36  self.isEnabled = true
37
38  g_messageCenter:subscribe(MessageType.PLAYER_FARM_CHANGED,
    self.playerFarmChanged, self)
39
40  self:updateIconVisibility()
41
42  self.activateText = g_i18n:getText("action_activateShop")
43
44  return self
45  end
```

delete

Description

Deleting shop trigger

Definition

delete()

Code

```
49  function ShopTrigger:delete()
50  g_messageCenter:unsubscribeAll(self)
51  if self.triggerId ~= nil then
52  removeTrigger(self.triggerId)
53  end
54  self.shopSymbol = nil
55  g_currentMission:removeActivatableObject(self)
56  end
```

getIsActivatable

Description

Returns true if shop can be opened

Definition

getIsActivatable()

Return Values

boolean isActivateable is activateable

Code

```
61  function ShopTrigger:getIsActivatable()
```

```
62  return self.isEnabled and g_currentMission.controlPlayer and
    g_currentMission:getFarmId() ~= FarmManager.SPECTATOR_FARM_ID
63  end
```

onActivateObject

Description

Called on activate object

Definition

onActivateObject()

Code

```
71  function ShopTrigger:onActivateObject()
72  g_currentMission:addActivatableObject(self)
73  self.objectActivated = true
74
75  local inGameMenu = g_gui:getScreenInstanceByClass(InGameMenu)
76  inGameMenu:setMode(InGameMenu.MODE_SHOP)
77  g_gui:changeScreen(nil, InGameMenu)
78
79  local x,y,z = getWorldTranslation(self.shopPlayerSpawn)
80  local dx, _, dz = localDirectionToWorld(self.shopPlayerSpawn, 0,
    0, -1)
81  g_currentMission.player:moveToAbsolute(x,y,z)
82  g_currentMission.player.rotY =
    MathUtil.getYRotationFromDirection(dx, dz)
83  end
```

triggerCallback

Description

Trigger callback

Definition

triggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

Code

```
92  function ShopTrigger:triggerCallback(triggerId, otherId, onEnter,
    onLeave, onStay)
93  if self.isEnabled and (not g_isPresentationVersion or
    g_isPresentationVersionShopEnabled) and
    g_currentMission.missionInfo:isa(FSCareerMissionInfo) then
94  if onEnter or onLeave then
```

```
95  if g_currentMission.player ~= nil and otherId ==
    g_currentMission.player.rootNode then
96  if onEnter then
97  if not self.objectActivated then
98  g_currentMission:addActivatableObject(self)
99  self.objectActivated = true
100 end
101 else
102 if self.objectActivated then
103 g_currentMission:removeActivatableObject(self)
104 self.objectActivated = false
105 end
106 end
107 end
108 end
109 end
110 end
```

updateIconVisibility

Description

Turn the icon on or off depending on the current game and the players farm

Definition

updateIconVisibility()

TransportMissionTrigger

Description

Class for transport mission triggers

onCreate

Description

On create mission trigger

Definition

onCreate(integer id)

Arguments

integer id trigger node id

Code

```
19  function TransportMissionTrigger:onCreate(id)
20  g_currentMission:addNonUpdateable(TransportMissionTrigger:new(id))
21  end
```

new

Description

Creating mission trigger object

Definition

new(integer name)

Arguments

integer name trigger node id

Return Values

table instance instance of object

Code

```
27  function TransportMissionTrigger:new(id)
28  local self = {}
29  setmetatable(self, TransportMissionTrigger_mt)
30
31  self.triggerId = id
32  self.index = getUserAttribute(self.triggerId, "index")
33
34  addTrigger(id, "triggerCallback", self)
35
36  self.isEnabled = true
37
38  g_missionManager:addTransportMissionTrigger(self)
39
40  -- Hide until needed
41  self:setMission(nil)
42
43  return self
44  end
```

delete

Description

Deleting shop trigger

Definition

delete()

Code

```
48  function TransportMissionTrigger:delete()
49  removeTrigger(self.triggerId)
50
51  g_missionManager:removeTransportMissionTrigger(self)
52  end
```

triggerCallback

Description

Trigger callback

Definition

triggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer triggerId id of trigger

integer otherId id of actor
boolean onEnter on enter
boolean onLeave on leave
boolean onStay on stay

Code

```
72  function TransportMissionTrigger:triggerCallback(triggerId,
    otherId, onEnter, onLeave, onStay)
73  if self.isEnabled and self.mission ~= nil then
74  if onEnter then
75  self.mission:objectEnteredTrigger(self, otherId)
76  elseif onLeave then
77  self.mission:objectLeftTrigger(self, otherId)
78  end
79  end
80  end
```

UnloadFeedingTrough

Description

new

Description

Creates a new instance of the class

Definition

new(bool isServer, bool isClient, table customMt)

Arguments

bool isServer true if we are server
bool isClient true if we are client
table customMt meta table

Return Values

table self returns the instance

Code

```
25  function UnloadFeedingTrough:new(isServer, isClient, customMt)
26  local self = UnloadTrigger:new(isServer, isClient, customMt or
    UnloadFeedingTrough_mt)
27
28  self.animalPlaces = {}
29
30  return self
31  end
```

load

Description

Loads elements of the class

Definition

load(table rootNode, string xmlFile, string xmlNode)

Arguments

table rootNode of the object
string xmlFile file to read
string xmlNode xmlNode to read from

Return Values

bool return true if successful

Code

```
39  function UnloadFeedingTrough:load(rootNode, xmlFile, xmlNode,
    target)
40  local returnValue = UnloadFeedingTrough:superClass().load(self,
    rootNode, xmlFile, xmlNode, target)
41
42  if returnValue then
43  self:loadAnimalPlaces(rootNode, xmlFile, xmlNode)
44  end
45  return returnValue
46  end
```

loadAnimalPlaces

Description

Loads animal places

Definition

loadAnimalPlaces(table rootNode, string xmlFile, string xmlNode)

Arguments

table rootNode of the object
string xmlFile file to read
string xmlNode xmlNode to read from

Code

```
53  function UnloadFeedingTrough:loadAnimalPlaces(rootNode, xmlFile,
    xmlNode)
54  -- print(string.format("-- [UnloadFeedingTrough:loadAnimalPlaces]
    rootNode(%s) xmlFile(%s) xmlNode(%s)", tostring(rootNode),
    tostring(xmlFile), tostring(xmlNode)))
55  local animalPlacesNode =
    XMLUtil.getValueFromXMLFileOrUserAttribute(xmlFile, xmlNode,
    "animalPlacesNode", getXMLString, rootNode)
56  if animalPlacesNode ~= nil then
57  local animalPlaces = I3DUtil.indexToObject(rootNode,
    animalPlacesNode)
58  -- print(string.format("-- [UnloadFeedingTrough:loadAnimalPlaces]
    animalPlacesNode(%s) animalPlaces(%s)",
    tostring(animalPlacesNode), tostring(animalPlaces)))
59  if animalPlaces ~= nil and self.target~= nil and
    self.target.husbandryId ~= nil then
60  for i = 1, getNumOfChildren(animalPlaces) do
61  local animalPlaceId = getChildAt(animalPlaces, i - 1)
```

```
62  local animalPlace = addFeedingPlace(self.target.husbandryId, animalPlaceId, 0.0)
63  table.insert(self.animalPlaces, animalPlace)
64  end
65  end
66  end
67  end
```

loadFillTypes

Description

Overriding Unload Trigger method with empty method. The animal husbandry module is taking care of setting up the filltypes

Definition

loadFillTypes(table rootNode, string xmlFile, string xmlNode)

Arguments

table rootNode of the object
string xmlFile file to read
string xmlNode xmlNode to read from

Code

```
74  function UnloadFeedingTrough:loadFillTypes(rootNode, xmlFile, xmlNode)
75  end
```

initFillTypesFromFoodGroups

Description

Setup fillTypes from animal food groups

Definition

initFillTypesFromFoodGroups(table foodGroups)

Arguments

table foodGroups

Code

```
81  function UnloadFeedingTrough:initFillTypesFromFoodGroups(foodGroups)
82  self.fillTypes = {}
83  for _, foodGroup in pairs(foodGroups) do
84  for _, fillTypeIndex in pairs(foodGroup.fillTypes) do
85  self.fillTypes[fillTypeIndex] = true
86  end
87  end
88  end
```

addFillUnitFillLevel

Description

Changes fill levels from a tool

Definition

addFillUnitFillLevel(float deltaFillLevel, integer fillType, table fillInfo, integer toolType)

Arguments

float	deltaFillLevel
integer	fillType
table	fillInfo
integer	toolType

Return Values

float returns the change delta

Code

```
101 function UnloadFeedingTrough:addFillUnitFillLevel(farmId, fillUnitIndex, fillLevelDelta, fillTypeIndex, toolType, fillPositionData)
102 local foodMixture = g_animalFoodManager:getFoodMixtureByFillType(fillTypeIndex)
103 local delta = 0
104 if foodMixture ~= nil then
105 for _, ingredient in ipairs(foodMixture.ingredients) do
106 local ingredientFillType = ingredient.fillTypes[1]
107 local ingredientFillLevel = fillLevelDelta * ingredient.weight
108 delta = delta + UnloadFeedingTrough:superClass().addFillUnitFillLevel(self, farmId, fillUnitIndex, ingredientFillLevel, ingredientFillType, toolType, fillPositionData)
109 end
110 else
111 delta = UnloadFeedingTrough:superClass().addFillUnitFillLevel(self, farmId, fillUnitIndex, fillLevelDelta, fillTypeIndex, toolType, fillPositionData)
112 end
113
114 self:updateAnimalPlaces(delta)
115 return delta
116 end
```

updateAnimalPlaces

Description

Changes animal places visuals

Definition

updateAnimalPlaces()

Code

```
120 function UnloadFeedingTrough:updateAnimalPlaces(delta)
121 if delta ~= nil and delta > 0 then
122 for _, animalPlace in pairs(self.animalPlaces) do
123 updateFeedingPlace(self.target.husbandryId, animalPlace, delta)
124 end
```

```
125  end
126  end
```

UnloadTrigger

Description

new

Description

Creates a new instance of the class

Definition

new(bool isServer, bool isClient, table customMt)

Arguments

bool isServer true if we are server
bool isClient true if we are client
table customMt meta table

Return Values

table self returns the instance

Code

```
32  function UnloadTrigger:new(isServer, isClient, customMt)
33  local self = Object:new(isServer, isClient, customMt or
    UnloadTrigger_mt)
34
35  self.baleTriggerNode = nil
36  self.balesInTrigger = {}
37  self.fillTypes = {}
38  self.avoidFillTypes = {}
39  self.acceptedToolTypes = {}
40
41  return self
42  end
```

load

Description

Loads elements of the class

Definition

load(table rootNode, string xmlFile, string xmlNode)

Arguments

table rootNode of the object
string xmlFile file to read
string xmlNode xmlNode to read from

Return Values

bool return true if successful

Code

```
50  function UnloadTrigger:load(rootNode, xmlFile, xmlNode, target)
51  self:loadBaleTrigger(rootNode, xmlFile, xmlNode)
52
```

```
53  local exactFillRootNode = XMLUtil.getValueFromXMLFileOrUserAttribute(xmlFile, xmlNode, "exactFillRootNode", getXMLString, rootNode)
54  self.exactFillRootNode = I3DUtil.indexToObject(rootNode, exactFillRootNode)
55
56  if self.baleTriggerNode == nil and self.exactFillRootNode == nil then
57  g_logManager:warning("Missing exactFillRootNode or baleTrigger for unloadTrigger")
58  return false
59  end
60
61  if self.exactFillRootNode ~= nil then
62  local colMask = getCollisionMask(self.exactFillRootNode)
63  if bitAND(FillUnit.EXACTFILLROOTNODE_MASK, colMask) == 0 then
64  g_logManager:warning("Invalid exactFillRootNode collision mask for unloadTrigger. Bit 30 needs to be set!")
65  return false
66  end
67
68  g_currentMission:addNodeObject(self.exactFillRootNode, self)
69  end
70
71  if target ~= nil then
72  self:setTarget(target)
73  end
74
75  self:loadFillTypes(rootNode, xmlFile, xmlNode)
76  self:loadAcceptedToolType(rootNode, xmlFile, xmlNode)
77  self:loadAvoidFillTypes(rootNode, xmlFile, xmlNode)
78  self.isEnabled = true
79
80  return true
81  end
```

loadAcceptedToolType

Description

Loads accepted tool type

Definition

loadAcceptedToolType(table rootNode, string xmlFile, string xmlNode)

Arguments

table	rootNode of the object

string xmlFile file to read
string xmlNode xmlNode to read from

Code

```
88  function UnloadTrigger:loadAcceptedToolType(rootNode, xmlFile, xmlNode)
89  local acceptedToolTypeNames = XMLUtil.getValueFromXMLFileOrUserAttribute(xmlFile, xmlNode, "acceptedToolTypes", getXMLString, rootNode)
90  local acceptedToolTypes = StringUtil.getVectorFromString(acceptedToolTypeNames)
91
92  if acceptedToolTypes ~= nil then
93  for _,acceptedToolType in pairs(acceptedToolTypes) do
94  local toolTypeInt = g_toolTypeManager:getToolTypeIndexByName(acceptedToolType)
95  self.acceptedToolTypes[toolTypeInt] = true
96  end
97  else
98  self.acceptedToolTypes = nil
99  end
100 end
```

loadAvoidFillTypes

Description

Loads avoid fill Types

Definition

loadAvoidFillTypes(table rootNode, string xmlFile, string xmlNode)

Arguments

table rootNode of the object
string xmlFile file to read
string xmlNode xmlNode to read from

Code

```
107 function UnloadTrigger:loadAvoidFillTypes(rootNode, xmlFile, xmlNode)
108 local avoidFillTypeCategories = XMLUtil.getValueFromXMLFileOrUserAttribute(xmlFile, xmlNode, "avoidFillTypeCategories", getXMLString, rootNode)
109 local avoidFillTypeNames = XMLUtil.getValueFromXMLFileOrUserAttribute(xmlFile, xmlNode, "avoidFillTypes", getXMLString, rootNode)
110 local avoidFillTypes = nil
111
112 if avoidFillTypeCategories ~= nil and avoidFillTypeNames == nil then
113 avoidFillTypes = g_fillTypeManager:getFillTypesByCategoryNames(avoidFillTypeCategories, "Warning: UnloadTrigger has invalid avoidFillTypeCategory '%s'.")
```

```
114  elseif avoidFillTypeCategories == nil and avoidFillTypeNames ~= nil
     then
115  avoidFillTypes =
     g_fillTypeManager:getFillTypesByNames(avoidFillTypeNames, "Warning:
     UnloadTrigger has invalid avoidFillType '%s'.")
116  end
117  if avoidFillTypes ~= nil then
118  for _,fillType in pairs(avoidFillTypes) do
119  self.avoidFillTypes[fillType] = true
120  end
121  else
122  self.avoidFillTypes = nil
123  end
124  end
```

loadFillTypes

Description

Loads fill Types

Definition

loadFillTypes(table rootNode, string xmlFile, string xmlNode)

Arguments

table	rootNode of the object
string xmlFile	file to read
string xmlNode	xmlNode to read from

Code

```
131  function UnloadTrigger:loadFillTypes(rootNode, xmlFile, xmlNode)
132  local fillTypeCategories =
     XMLUtil.getValueFromXMLFileOrUserAttribute(xmlFile, xmlNode,
     "fillTypeCategories", getXMLString, rootNode)
133  local fillTypeNames =
     XMLUtil.getValueFromXMLFileOrUserAttribute(xmlFile, xmlNode,
     "fillTypes", getXMLString, rootNode)
134  local fillTypes = nil
135
136  if fillTypeCategories ~= nil and fillTypeNames == nil then
137  fillTypes =
     g_fillTypeManager:getFillTypesByCategoryNames(fillTypeCategories,
     "Warning: UnloadTrigger has invalid fillTypeCategory '%s'.")
138  elseif fillTypeNames ~= nil then
139  fillTypes = g_fillTypeManager:getFillTypesByNames(fillTypeNames,
     "Warning: UnloadTrigger has invalid fillType '%s'.")
140  end
141  if fillTypes ~= nil then
142  for _, fillType in pairs(fillTypes) do
```

```
143 self.fillTypes[fillType] = true
144 end
145 else
146 self.fillTypes = nil
147 end
148 end
```

loadBaleTrigger

Description

Loads bale trigger

Definition

loadBaleTrigger(table rootNode, string xmlFile, string xmlNode)

Arguments

table	rootNode of the object
string xmlFile	file to read
string xmlNode	xmlNode to read from

Code

```
155 function UnloadTrigger:loadBaleTrigger(rootNode, xmlFile,
    xmlNode)
156 local baleTriggerNode =
    XMLUtil.getValueFromXMLFileOrUserAttribute(xmlFile, xmlNode,
    "baleTriggerIndex", getXMLString, rootNode)
157 if baleTriggerNode ~= nil then
158 g_logManager:warning("'baleTriggerIndex' is not supported anymore
    for unloadTrigger! Please use 'baleTriggerNode' instead!")
159 end
160
161 baleTriggerNode =
    XMLUtil.getValueFromXMLFileOrUserAttribute(xmlFile, xmlNode,
    "baleTriggerNode", getXMLString, rootNode)
162
163 if baleTriggerNode ~= nil then
164 self.baleTriggerNode = I3DUtil.indexToObject(rootNode,
    baleTriggerNode)
165 if self.baleTriggerNode ~= nil then
166 addTrigger(self.baleTriggerNode, "baleTriggerCallback", self)
167 end
168 end
169
170 local baleDeleteLitersPerSecond =
    XMLUtil.getValueFromXMLFileOrUserAttribute(xmlFile, xmlNode,
    "baleDeleteLitersPerSecond", getXMLInt, rootNode)
171 if baleDeleteLitersPerSecond ~= nil then
172 self.baleDeleteLitersPerMS = baleDeleteLitersPerSecond * 0.0001
```

```
173 end
174 end
```

delete

Description

Delete instance

Definition

delete()

Code

```
178 function UnloadTrigger:delete()
179 if self.baleTriggerNode ~= nil and self.baleTriggerNode ~= 0 then
180 removeTrigger(self.baleTriggerNode)
181 self.baleTriggerNode = 0
182 end
183
184 UnloadTrigger:superClass().delete(self)
185 end
```

setTarget

Description

Connects object using the trigger to the trigger

Definition

setTarget(table object)

Arguments

table object target on which the unload trigger is attached

Code

```
190 function UnloadTrigger:setTarget(object)
191 assert(object.getIsFillTypeAllowed ~= nil)
192 assert(object.getIsToolTypeAllowed ~= nil)
193 assert(object.addFillLevelFromTool ~= nil)
194 assert(object.getFreeCapacity ~= nil)
195
196 self.target = object
197 end
```

update

Description

Update method

Definition

update(float dt)

Arguments

float dt delta time

Code

```
206 function UnloadTrigger:update(dt)
```

```
207 UnloadTrigger:superClass().update(self, dt)
208
209 self:updateBales(dt)
210 end
```

updateBales

Description

Update bale mechanics

Definition

updateBales(float dt)

Arguments

float dt delta time

Code

```
215 function UnloadTrigger:updateBales(dt)
216 for index, bale in ipairs(self.balesInTrigger) do
217 if bale ~= nil and bale.nodeId ~= 0 then
218 if bale.dynamicMountJointIndex == nil then
219 local fillType = bale:getFillType()
220 local fillLevel = bale:getFillLevel()
221 local fillInfo = nil
222
223 local delta = bale:getFillLevel()
224 if self.baleDeleteLitersPerMS ~= nil then
225 delta = self.baleDeleteLitersPerMS * dt
226 end
227
228 if delta > 0 then
229 self.target:addFillLevelFromTool(bale:getOwnerFarmId(), delta,
    fillType, fillInfo, ToolType.BALE)
230 bale:setFillLevel(fillLevel - delta)
231 local newFillLevel = bale:getFillLevel()
232 if newFillLevel < 0.01 then
233 bale:delete()
234 table.remove(self.balesInTrigger, index)
235 break
236 end
237 end
238 end
239 else
240 table.remove(self.balesInTrigger, index)
241 end
242 end
```

```
243
244 if #self.balesInTrigger > 0 then
245 self:raiseActive()
246 end
247 end
```

getFillUnitIndexFromNode

Description

Returns default value '1'

Definition

getFillUnitIndexFromNode(integer node)

Arguments

integer node scenegraph node

Code

```
252 function UnloadTrigger:getFillUnitIndexFromNode(node)
253 return 1
254 end
```

getFillUnitExactFillRootNode

Description

Returns exactFillRootNode

Definition

getFillUnitExactFillRootNode(integer fillUnitIndex)

Arguments

integer fillUnitIndex index of fillunit

Code

```
259 function UnloadTrigger:getFillUnitExactFillRootNode(fillUnitIndex)
260 return self.exactFillRootNode
261 end
```

addFillUnitFillLevel

Description

Increase fill level

Definition

addFillUnitFillLevel(integer fillUnitIndex, float fillLevelDelta, integer fillTypeIndex, table toolType, table fillPositionData)

Arguments

integer fillUnitIndex
float fillLevelDelta
integer fillTypeIndex
table toolType
table fillPositionData

Return Values

Code

```
271 function UnloadTrigger:addFillUnitFillLevel(farmId, fillUnitIndex, fillLevelDelta, fillTypeIndex, toolType, fillPositionData)
272 local applied = self.target:addFillLevelFromTool(farmId, fillLevelDelta, fillTypeIndex, fillPositionData, toolType)
273 return applied
274 end
```

getFillUnitAllowsFillType

Description

Checks if fill type is allowed

Definition

getFillUnitAllowsFillType(integer fillUnitIndex, integer fillType)

Arguments

integer fillUnitIndex

integer fillType

Return Values

bool true if allowed

Code

```
290 function UnloadTrigger:getFillUnitAllowsFillType(fillUnitIndex, fillType)
291 return self:getIsFillTypeAllowed(fillType)
292 end
```

getIsFillTypeAllowed

Description

Checks if fillType is allowed

Definition

getIsFillTypeAllowed(integer fillType)

Arguments

integer fillType

Return Values

boolean isAllowed true if fillType is supported else false

Code

```
298 function UnloadTrigger:getIsFillTypeAllowed(fillType)
299 return self:getIsFillTypeSupported(fillType) and self:getFillUnitFreeCapacity(1, fillType) > 0
300 end
```

getIsFillTypeSupported

Description

Checks if fillType is supported

Definition

getIsFillTypeSupported(integer fillType)

Arguments

integer fillType

Return Values

boolean isSupported true if fillType is supported else false

Code

```
306  function UnloadTrigger:getIsFillTypeSupported(fillType)
307  local accepted = true
308
309  if self.target ~= nil then
310  if not self.target:getIsFillTypeAllowed(fillType) then
311  accepted = false
312  end
313  else
314  if self.fillTypes ~= nil then
315  if not self.fillTypes[fillType] then
316  accepted = false
317  end
318  end
319  end
320
321  if self.avoidFillTypes ~= nil then
322  if self.avoidFillTypes[fillType] then
323  accepted = false
324  end
325  end
326
327  return accepted
328  end
```

getFillUnitFreeCapacity

Description

Returns the free capacity

Definition

getFillUnitFreeCapacity(integer fillUnitIndex, integer fillTypeIndex)

Arguments

integer fillUnitIndex fill unit index

integer fillTypeIndex fill type index

Return Values

float freeCapacity free capacity

Code

```
343  function UnloadTrigger:getFillUnitFreeCapacity(fillUnitIndex,
     fillTypeIndex, farmId)
344  if self.target.getFreeCapacity ~= nil then
345  return self.target:getFreeCapacity(fillTypeIndex, farmId)
346  end
```

```
347  return 0
348  end
```

getIsToolTypeAllowed

Description

Checks if toolType is allowed

Definition

getIsToolTypeAllowed(integer toolType)

Arguments

integer toolType

Return Values

boolean isAllowed true if toolType is allowed else false

Code

```
354  function UnloadTrigger:getIsToolTypeAllowed(toolType)
355  local accepted = true
356
357  if self.acceptedToolTypes ~= nil then
358  if self.acceptedToolTypes[toolType] ~= true then
359  accepted = false
360  end
361  end
362
363  if accepted then
364  return self.target:getIsToolTypeAllowed(toolType)
365  else
366  return false
367  end
368  end
```

baleTriggerCallback

Description

Callback method for the bale trigger

Definition

baleTriggerCallback(integer triggerId, integer otherId, bool onEnter, bool onLeave, bool onStay, integer otherShapeId)

Arguments

integer triggerId
integer otherId
bool onEnter
bool onLeave
bool onStay
integer otherShapeId

Code

```
function UnloadTrigger:baleTriggerCallback(triggerId, otherId, onEnter, onLeave, onStay, otherShapeId)
if self.isEnabled then
local object = g_currentMission:getNodeObject(otherId)
if object ~= nil and object:isa(Bale) then
if onEnter then
local fillType = object:getFillType()

if self.target:getIsFillTypeAllowed(fillType) and self.target:getIsToolTypeAllowed(ToolType.BALE) then
if self.target:getFreeCapacity(fillType) > 0 then
self:raiseActive()
table.insert(self.balesInTrigger, object)
end
end

if self.fillTypes ~= nil and self.fillTypes[fillType] ~= true then
return
end
if self.avoidFillTypes ~= nil and self.avoidFillTypes[fillType] == true then
return
end
if self.acceptedToolTypes ~= nil and self.acceptedToolTypes[ToolType.BALE] ~= true then
return
end

if self.target:getIsFillTypeAllowed(fillType) and self.target:getIsToolTypeAllowed(ToolType.BALE) then
self:raiseActive()
table.insert(self.balesInTrigger, object)
end
elseif onLeave then
for index,bale in ipairs(self.balesInTrigger) do
if bale == object then
table.remove(self.balesInTrigger, index)
break
end
end
end
end
```

```
414 end
415 end
416 end
```

WeighStation

Description

Class for weigh stations

onCreate

Description

On create weigh station

Definition

onCreate(integer id)

Arguments

integer id id of weigh station node

Code

```
13 function WeighStation:onCreate(id)
14 g_currentMission:addNonUpdateable(WeighStation:new(id))
15 end
```

new

Description

Create new weigh station object

Definition

new(integer trigger)

Arguments

integer trigger id id of trigger node

Return Values

table instance instance of object

Code

```
21 function WeighStation:new(triggerId)
22 local self = {}
23 setmetatable(self, WeighStation_mt)
24
25 local nodeId = triggerId
26 self.triggerId = triggerId
27 addTrigger(triggerId, "triggerCallback", self)
28
29 self.isEnabled = true
30 self.triggerVehicles = {}
31
32 local weightDisplayIndex = getUserAttribute(nodeId,
   "weightDisplayIndex")
33 if weightDisplayIndex ~= nil then
```

```
34  self.displayNumbers = I3DUtil.indexToObject(nodeId,
    weightDisplayIndex)
35  end
36
37  return self
38  end
```

delete

Description

Delete weigh station

Definition

delete()

Code

```
42  function WeighStation:delete()
43  if self.triggerId ~= nil then
44  removeTrigger(self.triggerId)
45  self.triggerId = nil
46  end
47  end
```

updateDisplayNumbers

Description

Write new mass into the display

Definition

updateDisplayNumbers(float mass)

Arguments

float mass mass

Code

```
52  function WeighStation:updateDisplayNumbers(mass)
53  if self.displayNumbers ~= nil then
54  I3DUtil.setNumberShaderByValue(self.displayNumbers,
    math.floor(mass), 0)
55  end
56  end
```

updateWeight

Description

Get mass of vehicles in trigger and update display

Definition

updateWeight()

Code

```
60  function WeighStation:updateWeight()
61  local mass = 0
62  for vehicle, _ in pairs(self.triggerVehicles) do
63  mass = mass + vehicle:getTotalMass()
```

```
64  end
65  self:updateDisplayNumbers(mass*1000)
66  end
```

triggerCallback

Description

Trigger callback

Definition

triggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

Code

```
75  function WeighStation:triggerCallback(triggerId, otherId, onEnter, onLeave, onStay)
76  if self.isEnabled and (onEnter or onLeave) then
77  local vehicle = g_currentMission.nodeToObject[otherId]
78  if onEnter then
79  if vehicle ~= nil then
80  if self.triggerVehicles[vehicle] == nil then
81  self.triggerVehicles[vehicle] = 0
82  end
83  self.triggerVehicles[vehicle] = self.triggerVehicles[vehicle] + 1
84  end
85  else
86  if vehicle ~= nil then
87  self.triggerVehicles[vehicle] = self.triggerVehicles[vehicle] - 1
88  if self.triggerVehicles[vehicle] == 0 then
89  self.triggerVehicles[vehicle] = nil
90  end
91  end
92  end
93
94  self:updateWeight()
95  end
96  end
```

ConfigurationUtil

Description

Vehicle configuration util class

addBoughtConfiguration

Description

Add bought configuration

Definition

addBoughtConfiguration(string name, integer id)

Arguments

string	name of bought configuration type
integer id	id of bought configuration

Code

```
23  function ConfigurationUtil.addBoughtConfiguration(object, name,
    id)
24  if g_configurationManager:getConfigurationIndexByName(name) ~= nil
    then
25  if object.boughtConfigurations[name] == nil then
26  object.boughtConfigurations[name] = {}
27  end
28  object.boughtConfigurations[name][id] = true
29  end
30  end
```

hasBoughtConfiguration

Description

Returns true if configuration has been bought

Definition

hasBoughtConfiguration(string name, integer id)

Arguments

string	name of bought configuration type
integer id	id of bought configuration

Return Values

boolean configurationHasBeenBought configuration has been bought

Code

```
37  function ConfigurationUtil.hasBoughtConfiguration(object, name,
    id)
38  if object.boughtConfigurations[name] ~= nil and
    object.boughtConfigurations[name][id] then
39  return true
40  end
41  return false
42  end
```

setConfiguration

Description

Set configuration value

Definition

setConfiguration(string name, integer id)

Arguments

string name name of configuration type

integer id id of configuration value

Code

```
48  function ConfigurationUtil.setConfiguration(object, name, id)
49  object.configurations[name] = id
50  end
```

getColorByConfigId

Description

Returns color of config id

Definition

getColorByConfigId(string configName, integer configId)

Arguments

string configName name if config

integer configId id of config to get color

Return Values

table color color (r, g, b)

Code

```
58  function ConfigurationUtil.getColorByConfigId(object, configName,
    configId)
59  local configId = object.configurations[configName]
60  if configId ~= nil then
61  local item =
    g_storeManager:getItemByXMLFilename(object.configFileName)
62  local config = item.configurations[configName][configId]
63  if config ~= nil then
64  return config.color
65  end
66  end
67
68  return nil
69  end
```

getMaterialByConfigId

Description

Returns material of config id

Definition

getMaterialByConfigId(string configName, integer configId)

Arguments

string configName name if config

integer configId id of config to get color

Return Values

integer material material

Code

```
76  function ConfigurationUtil.getMaterialByConfigId(object,
    configName, configId)
77  local configId = object.configurations[configName]
78  if configId ~= nil then
79  local item =
    g_storeManager:getItemByXMLFilename(object.configFileName)
80  local config = item.configurations[configName][configId]
81  if config ~= nil then
82  return config.material
83  end
84  end
85
86  return nil
87  end
```

applyDesign

Description

Apply design config

Definition

applyDesign(integer xmlFile, integer configDesignId)

Arguments

integer xmlFile	id of xml object
integer configDesignId	id of design to apply

Code

```
93  function ConfigurationUtil.applyDesign(object, xmlFile,
    configDesignId)
94  local designKey =
    string.format("vehicle.designConfigurations.designConfiguration(%d)",
    configDesignId-1)
95  if not hasXMLProperty(xmlFile, designKey) then
96  print("Warning: Invalid design configuration " .. configDesignId);
97  return
98  end
99  local i = 0
100 while true do
101 local materialKey = string.format(designKey..".material(%d)", i)
102 if not hasXMLProperty(xmlFile, materialKey) then
103 break
104 end
105 local baseMaterialNode = I3DUtil.indexToObject(object.components,
    getXMLString(xmlFile, materialKey.."#node"), object.i3dMappings);
106 local refMaterialNode = I3DUtil.indexToObject(object.components,
    getXMLString(xmlFile, materialKey.."#refNode"), object.i3dMappings);
107 if baseMaterialNode ~= nil and refMaterialNode ~= nil then
```

```
108 local oldMaterial = getMaterial(baseMaterialNode, 0)
109 local newMaterial = getMaterial(refMaterialNode, 0)
110 for _, component in pairs(object.components) do
111 ConfigurationUtil.replaceMaterialRec(object, component.node,
    oldMaterial, newMaterial);
112 end;
113 end
114
115 local materialName = getXMLString(xmlFile, materialKey .. "#name")
116 if materialName ~= nil then
117 local shaderParameterName = getXMLString(xmlFile, materialKey ..
    "#shaderParameter")
118 if shaderParameterName ~= nil then
119 local colorStr = getXMLString(xmlFile, materialKey.."#color")
120 if colorStr ~= nil then
121 local color = g_brandColorManager:getBrandColorByName(colorStr)
122 if color == nil then
123 color = ConfigurationUtil.getColorFromString(colorStr)
124 end
125 if color ~= nil then
126 local materialId = getXMLInt(xmlFile, materialKey.."#materialId")
127 if object.setBaseMaterialColor ~= nil then
128 object:setBaseMaterialColor(materialName, shaderParameterName, color,
    materialId)
129 end
130 end
131 end
132 end
133 end
134
135 i = i + 1
136 end
137 ObjectChangeUtil.updateObjectChanges(xmlFile,
    "vehicle.designConfigurations.designConfiguration", configDesignId,
    object.components, object);
138 end
```

replaceMaterialRec

Description

Replace material of node

Definition

replaceMaterialRec(integer node, integer oldMaterial, integer newMaterial)

Arguments

integer node	id of node
integer oldMaterial	id of old material
integer newMaterial	id of new material

Code

```
145 function ConfigurationUtil.replaceMaterialRec(object, node, oldMaterial, newMaterial)
146 if getHasClassId(node, ClassIds.SHAPE) then
147 local nodeMaterial = getMaterial(node, 0);
148 if nodeMaterial == oldMaterial then
149 setMaterial(node, newMaterial, 0)
150 end
151 end;
152
153 local numChildren = getNumOfChildren(node);
154 if numChildren > 0 then
155 for i=0, numChildren-1 do
156 ConfigurationUtil.replaceMaterialRec(object, getChildAt(node, i), oldMaterial, newMaterial)
157 end;
158 end;
159 end
```

setColor

Description

Sets color of vehicle

Definition

setColor(integer xmlFile, string configName, integer configColorId)

Arguments

integer xmlFile	id of xml object
string configName	name of config
integer configColorId	id of config color to use

Code

```
166 function ConfigurationUtil.setColor(object, xmlFile, configName, configColorId)
167 local color = ConfigurationUtil.getColorByConfigId(object, configName, configColorId)
168 if color ~= nil then
169 local r,g,b,a = unpack(color);
170 local i = 0;
171 while true do
172 local colorKey = string.format("vehicle.%sConfigurations.colorNode(%d)", configName, i)
173 if not hasXMLProperty(xmlFile, colorKey) then
```

```
174 break;
175 end
176
177 local node = I3DUtil.indexToObject(object.components,
    getXMLString(xmlFile, colorKey .. "#node"), object.i3dMappings);
178 if node ~= nil then
179 if getHasClassId(node, ClassIds.SHAPE) then
180 if Utils.getNoNil(getXMLBool(xmlFile, colorKey .. "#recursive"),
    false) then
181 I3DUtil.setShaderParameterRec(node, "colorScale", r, g, b, a);
182 else
183 setShaderParameter(node, "colorScale", r, g, b, a, false);
184 end
185 else
186 print("Warning: Could not set vehicle color to
    '"..getName(node).."' because node is not a shape!")
187 end
188 end
189 i = i + 1;
190 end
191 end
192 end
```

getConfigurationValue

Description

Get value of configuration

Definition

getConfigurationValue(integer xmlFile, string key, string subKey, string param, function xmlFunc, any_type defaultValue, string fallbackConfigKey, string fallbackOldgKey)

Arguments

integer	xmlFile	id of xml object
string	key	key
string	subKey	sub key
string	param	parameter
function	xmlFunc	xml function
any_type	defaultValue	default value
string	fallbackConfigKey	fallback config key
string	fallbackOldgKey	fallback old key

Return Values

any_type value value of config

Code

```
205 function ConfigurationUtil.getConfigurationValue(xmlFile, key,
    subKey, param, xmlFunc, defaultValue, fallbackConfigKey,
    fallbackOldKey)
```

```
206 if type(subKey) == "table" then
207 printCallstack();
208 end
209 local value = nil;
210 if key ~= nil then
211 value = xmlFunc(xmlFile, key..subKey..param);
212 end;
213
214 if value == nil and fallbackConfigKey ~= nil then
215 value = xmlFunc(xmlFile, fallbackConfigKey..subKey..param); --
    Check for default configuration (xml index 0)
216 end;
217 if value == nil and fallbackOldKey ~= nil then
218 value = xmlFunc(xmlFile, fallbackOldKey..subKey..param); --
    Fallback to old xml setup
219 end;
220 return Utils.getNoNil(value, defaultValue); -- using default
    value
221 end;
```

getXMLConfigurationKey

Description

Get xml configuration key

Definition

getXMLConfigurationKey(integer xmlFile, integer index, string key, string defaultKey, string configurationKey)

Arguments

integer	xmlFile	id of xml object
integer	index	index
string	key	key
string	defaultKey	default key
string	configurationKey	configuration key

Return Values

string	configKey	key of configuration
integer	configIndex	index of configuration

Code

```
232 function ConfigurationUtil.getXMLConfigurationKey(xmlFile, index,
    key, defaultKey, configurationKey)
233 local configIndex = Utils.getNoNil(index, 1);
234 local configKey = string.format(key.."(%d)", configIndex-1);
235 if index ~= nil and not hasXMLProperty(xmlFile, configKey) then
236 print("Warning: Invalid "..configurationKey.." index
    '"..tostring(index).."' in '"..key.."'. Using default
    "..configurationKey.." settings instead!");
```

```
237 end;
238
239 if not hasXMLProperty(xmlFile, configKey) then
240 configKey = key.."(0)";
241 end;
242 if not hasXMLProperty(xmlFile, configKey) then
243 configKey = defaultKey;
244 end;
245
246 return configKey, configIndex;
247 end;
```

getConfigColorSingleItemLoad

Description

Get config color single item load

Definition

getConfigColorSingleItemLoad(integer xmlFile, string baseXMLName, string baseDir, string customEnvironment, boolean isMod, table configItem)

Arguments

integer	xmlFile	id of xml object
string	baseXMLName	base xml name
string	baseDir	base directory
string	customEnvironment	custom environment
boolean	isMod	is mod
table	configItem	config item

Code

```
257 function ConfigurationUtil.getConfigColorSingleItemLoad(xmlFile,
    baseXMLName, baseDir, customEnvironment, isMod, configItem)
258
259 local colorStr = Utils.getNoNil(getXMLString(xmlFile,
    baseXMLName.."#color"), "1 1 1 1")
260 local color = g_brandColorManager:getBrandColorByName(colorStr)
261 if color == nil then
262 color = ConfigurationUtil.getColorFromString(colorStr)
263 end
264
265 configItem.color = color
266 configItem.material = getXMLInt(xmlFile,
    baseXMLName.."#material")
267
268 configItem.name = XMLUtil.getXMLI18NValue(xmlFile,
    baseXMLName.."#name", getXMLString, "", "", customEnvironment,
    false)
269 end
```

getConfigColorPostLoad

Description

Get config color post load

Definition

getConfigColorPostLoad(integer xmlFile, string baseKey, string baseDir, string customEnvironment, boolean isMod, table configurationItems)

Arguments

integer	xmlFile	id of xml object
string	baseKey	base key
string	baseDir	base directory
string	customEnvironment	custom environment
boolean	isMod	is mod
table	configurationItems	config items

Code

```
279 function ConfigurationUtil.getConfigColorPostLoad(xmlFile, baseKey, baseDir, customEnvironment, isMod, configurationItems, storeItem)
280 local defaultColorIndex = getXMLInt(xmlFile, baseKey.."#defaultColorIndex")
281
282 if Utils.getNoNil(getXMLBool(xmlFile, baseKey.."#useDefaultColors"), false) then
283 local price = Utils.getNoNil(getXMLInt(xmlFile, baseKey.."#price"), 1000);
284
285 for i, color in pairs(g_vehicleColors) do
286 local configItem = StoreItemUtil.addConfigurationItem(configurationItems, "", "", price, 0, false)
287 if color.r ~= nil and color.g ~= nil and color.b ~= nil then
288 configItem.color = {color.r, color.g, color.b, 1}
289 elseif color.brandColor ~= nil then
290 configItem.color = g_brandColorManager:getBrandColorByName(color.brandColor)
291
292 if configItem.color == nil then
293 configItem.color = {1, 1, 1, 1}
294 g_logManager:warning("Unable to find brandColor '%s' in g_vehicleColors", color.brandColor)
295 end
296 end
297
298 configItem.name = g_i18n:convertText(color.name)
299
```

```
300 if i == defaultColorIndex then
301 configItem.isDefault = true
302 configItem.price = 0
303 end
304 end
305 end;
306
307 if defaultColorIndex == nil then
308 local defaultIsDefined = false
309 for _, item in ipairs(configurationItems) do
310 if item.isDefault ~= nil and item.isDefault then
311 defaultIsDefined = true
312 end
313 end
314
315 if not defaultIsDefined then
316 if #configurationItems > 0 then
317 configurationItems[1].isDefault = true
318 configurationItems[1].price = 0
319 end
320 end
321 end
322 end;
```

getStoreAddtionalConfigData

Description

Get store additional config data

Definition

getStoreAddtionalConfigData(integer xmlFile, string baseXMLName, string baseDir, string customEnvironment, boolean isMod, table configItem)

Arguments

integer	xmlFile	id of xml object
string	baseXMLName	base xml name
string	baseDir	base directory
string	customEnvironment	custom environment
boolean	isMod	is mod
table	configItem	config item

Code

```
338 function ConfigurationUtil.getStoreAddtionalConfigData(xmlFile,
    baseXMLName, baseDir, customEnvironment, isMod, configItem)
339 configItem.vehicleType = getXMLString(xmlFile,
    baseXMLName.."#vehicleType")
340 end
```

getColorFromString

Description

Get color from string

Definition

getColorFromString(string colorString)

Arguments

string colorString color rgba string or brand color identifier

Return Values

table color color (r, g, b)

Code

```
346 function ConfigurationUtil.getColorFromString(colorString)
347 if colorString ~= nil then
348 local colorVector = g_brandColorManager:getBrandColorByName(colorString) or {StringUtil.getVectorFromString(colorString)}
349
350 if colorVector == nil or #colorVector < 3 or #colorVector > 4 then
351 print("Error: Invalid color string '" .. colorString .. "'")
352 return nil
353 end
354 return colorVector;
355 end
356 return nil;
357 end
```

DebugUtil

Description

DebugUtil

drawDebugNode

Description

Draws a debug node and optional a text at world position of given node

Definition

drawDebugNode(integer id, string text)

Arguments

integer id node id

string text text

drawDebugCircle

Description

Draw debug circle

Definition

drawDebugCircle(float x, float y, float z, float radius, integer steps)

Arguments

float x world x position

float	y	world y position
float	z	world z position
float	radius	radius
integer	steps	steps

drawDebugCubeAtWorldPos

Description

Draw debug cube at world position

Definition

drawDebugCubeAtWorldPos(float x, float y, float z, float dirX, float dirY, float dirZ, float upX, float upY, float upZ, float sizeX, float sizeY, float sizeZ, float r, float g, float b)

Arguments

float x	world x center position
float y	world y center position
float z	world z center position
float dirX	x direction
float dirY	y direction
float dirZ	z direction
float upX	x up of vector
float upY	y up of vector
float upZ	z up of vector
float sizeX	x size
float sizeY	y size
float sizeZ	z size
float r	red value
float g	green value
float b	blue value

drawDebugCube

Description

Draw debug cube at given node

Definition

drawDebugCube(integer id, float sizeX, float sizeY, float sizeZ, float r, float g, float b)

Arguments

integer	id	node id
float	sizeX	x size
float	sizeY	y size
float	sizeZ	z size
float	r	red value
float	g	green value
float	b	blue value

drawSimpleDebugCube

Description

Draw debug cube

Definition

drawSimpleDebugCube(float x, float y, float z, float width, float red, float green, float blue)

Arguments

float x	world x center position
float y	world y center position
float z	world z center position
float width	
float red	
float green	
float blue	

drawDebugReferenceAxisFromNode

Description

Draw debug reference axis fom Node

Definition

drawDebugReferenceAxisFromNode(node to)

Arguments

node to draw a reference axis from

drawDebugReferenceAxis

Description

Draw debug reference axis

Definition

drawDebugReferenceAxis(float x, float y, float z, float up, float up, float up, float direction, float direction, float direction)

Arguments

float x	world x center position
float y	world y center position
float z	world z center position
float up	x
float up	y
float up	z
float direction	x
float direction	y
float direction	z

drawDebugParallelogram

Description

Draw debug parallelogram

Definition

drawDebugParallelogram(float x, float z, float widthX, float widthZ, float heightX, float heightZ, float heightOffset, float r, float g, float b, float a)

Arguments

float x	world x center position
float z	world z center position
float widthX	widthX
float widthZ	widthZ

float heightX	heightX
float heightZ	heightZ
float heightOffset	heightOffset
float r	red
float g	green
float b	blue
float a	alpha

printTableRecursively

Description

Print a table recursively

Definition

printTableRecursively(table inputTable, string inputIndent, integer depth, integer maxDepth)

Arguments

table	inputTable	table to print
string	inputIndent	input indent
integer	depth	current depth
integer	maxDepth	max depth

printCallingFunctionLocation

Description

Print the script call location of a function call which uses this function.

Definition

printCallingFunctionLocation()

FSDensityMapUtil

Description

FSDensityMapUtil

cutFruitArea

Description

Cut fruit area

Definition

cutFruitArea(integer fruitId, float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ, boolean destroySpray, boolean destroySeedingWidth, boolean useMinForageState)

Arguments

integer	fruitId	fruit id
float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z
float	heightWorldX	height world X
float	heightWorldZ	height world Z
boolean	destroySpray	destroy spray
boolean	destroySeedingWidth	destroy seeding width
boolean	useMinForageState	use min forange state

Return Values

integer	harvestPixelsSum	harvest of pixels sum
integer	harvestNumPixels	harvest number of pixels
float	sprayFactor	spray factor
float	plowFactor	plow factor
float	limeFactor	lime factor
float	weedFactor	weed factor
integer	growthState	growth state
float	maxArea	max area

getFruitArea

Description

Get fruit area

Definition

getFruitArea(integer fruitId, float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ, boolean allowPreparing, boolean useMinForageState)

Arguments

integer	fruitId	fruit id
float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z
float	heightWorldX	height world X
float	heightWorldZ	height world Z
boolean	allowPreparing	allow preparing
boolean	useMinForageState	use min forage state

Return Values

integer	ret	ret
integer	total	total

updateRollerArea

Description

Update roller area

Definition

updateRollerArea(float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ)

Arguments

float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z
float	heightWorldX	height world X
float	heightWorldZ	height world Z

Return Values

integer	changedValue	changed value

updateCultivatorArea

Description

Update cultivator area

Definition

updateCultivatorArea(float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ, boolean createField, boolean commonForced, float angle, integer blockedSprayType)

Arguments

float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z
float	heightWorldX	height world X
float	heightWorldZ	height world Z
boolean	createField	createField
boolean	commonForced	common createField
float	angle	angle
integer	blockedSprayType	blockedSprayType

Return Values

integer	changedArea	changed area pixels
integer	totalArea	total area

updatePlowArea

Description

Update plow area

Definition

updatePlowArea(float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ, boolean forced, boolean commonForced, float angle)

Arguments

float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z
float	heightWorldX	height world X
float	heightWorldZ	height world Z
boolean	forced	forced
boolean	commonForced	common forced
float	angle	angle

Return Values

integer	changedArea	changed area pixels
integer	totalArea	total area pixles

updateDestroyCommonArea

Description

Update destroy common area

Definition

updateDestroyCommonArea(float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ, boolean limitToField)

Arguments

float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z
float	heightWorldX	height world X
float	heightWorldZ	height world Z
boolean	limitToField	limit to field

updateSprayArea

Description

Update spray area

Definition

updateSprayArea(float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, integer sprayTypeIndex)

Arguments

float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z
float	heightWorldX	height world X
integer	sprayTypeIndex	spray type index

Return Values

integer	numPixels	number of pixels
integer	totalNumPixels	total number of pixels

updateHerbicideArea

Description

Update herbicide area

Definition

updateHerbicideArea(float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ)

Arguments

float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z
float	heightWorldX	height world X
float	heightWorldZ	height world Z

Return Values

integer	numPixels	number of pixels
integer	totalNumPixels	total number of pixels

updateWeederArea

Description

Update weeder area

Definition

updateWeederArea(float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ)

Arguments

float startWorldX	start world X
float startWorldZ	start world Z
float widthWorldX	width world X
float widthWorldZ	width world Z
float heightWorldX	height world X
float heightWorldZ	height world Z

Return Values

integer numPixels number of pixels

removeWeedArea

Description

Remove weed area

Definition

removeWeedArea(float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ, integer maxGrowthState)

Arguments

float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z
float	heightWorldX	height world X
float	heightWorldZ	height world Z
integer	maxGrowthState	max growth state of the weed

setWeedArea

Description

Set weed area

Definition

setWeedArea(float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ, integer value)

Arguments

float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z
float	heightWorldX	height world X
float	heightWorldZ	height world Z
integer	value	state of the weed

resetSprayArea

Description

Reset spray area

Definition

resetSprayArea(float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ, boolean force)

Arguments

float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z
float	heightWorldX	height world X
float	heightWorldZ	height world Z
boolean	force	force

updateSowingArea

Description

Update sowing area

Definition

updateSowingArea(integer fruitId, float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ, float angle, integer growthState)

Arguments

integer	fruitId	fruit id
float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z
float	heightWorldX	height world X
float	heightWorldZ	height world Z
float	angle	angle
integer	growthState	fruit growth state value

Return Values

integer	changedArea	changed area pixels
integer	totalArea	total area pixels

updateDirectSowingArea

Description

Update direct sowing area

Definition

updateDirectSowingArea(integer fruitId, float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ, float angle, integer growthState)

Arguments

integer	fruitId	fruit id
float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z

float	heightWorldX	height world X
float	heightWorldZ	height world Z
float	angle	angle
integer	growthState	fruit growth state value

Return Values

integer	changedArea	changed area pixels
integer	totalArea	total area pixels

updateFruitPreparerArea

Description

Update fruit preparer area

Definition

updateFruitPreparerArea(integer fruitId, float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ, float startDropWorldX, float startDropWorldZ, float widthDropWorldX, float widthDropWorldZ, float heightDropWorldX, float heightDropWorldZ)

Arguments

integer	fruitId	fruit id
float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z
float	heightWorldX	height world X
float	heightWorldZ	height world Z
float	startDropWorldX	start drop world X
float	startDropWorldZ	start drop world Z
float	widthDropWorldX	width drop world X
float	widthDropWorldZ	width drop world Z
float	heightDropWorldX	height drop world X
float	heightDropWorldZ	height drop world Z

Return Values

integer	numChangedPixels	number of changed pixels

eraseTireTrack

Description

Erase tire track on given parallelogram

Definition

eraseTireTrack(float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ)

Arguments

float	startWorldX	start world X
float	startWorldZ	start world Z
float	widthWorldX	width world X
float	widthWorldZ	width world Z
float	heightWorldX	height world X
float	heightWorldZ	height world Z

getTireTrackColorFromDensityBits

Description

Returns tire track color from given density bits

Definition

getTireTrackColorFromDensityBits(integer densityBits)

Arguments

integer densityBits density bits

Return Values

table color tire track color

FSMUtil

Description

create

Description

Creates an instance of SimpleStateMachine; consumer is responsible for deletion of instance.

Definition

create()

Code

```
15  function FSMUtil.create()
16  local fsm = SimpleStateMachine:new()
17  return fsm
18  end
```

GameSettings

Description

Class handling global settings. Global instance is g_gameSettings

new

Description

Create a new instance

Definition

new(table customMt, table messageCenter)

Arguments

table customMt	Sub-class meta table
table messageCenter	MessageCenter reference for settings change notifications

getValue

Description

Returns the setting value with the given name
Settings in the game:
-maxNumMirrors
-lightsProfile
-realBeaconLights
-motorStopTimerDuration
-uiScale
-fovY
-isTrainTabbable
-radioVehicleOnly
-radioIsActive

-useColorblindMode
-easyArmControl
-useMiles
-showTriggerMarker
-resetCamera
-useWorldCamera
-invertYLook
-showHelpIcons
-showHelpMenu
-radioVolume
-vehicleVolume
-environmentVolume
-cameraSensitivity
-vehicleArmSensitivity
-ingameMapState
-ingameMapFilter
-moneyUnit
-masterVolume
-musicVolume

Definition

getValue(string name)

Arguments

string name name of the setting

Return Values

mixed value Value of the setting. The type depends on the setting

setValue

Description

Changed the setting value with the given name

Definition

setValue(string name, mixed value, boolean doSave)

Arguments

string	name	name of the setting
mixed	value	value to set the setting to
boolean	doSave	If true, the settings are saved persistently, otherwise it will only be saved when another call triggers it

Return Values

boolean successful Returns true, if the setting was changed

HTMLUtil

Description

encodeToHTML

Description

Returns a html encoded string

Definition

encodeToHTML(string str, boolean inCData)

Arguments

string	str	input string
boolean	inCData	true if text is in cdata element

Return Values

string encodedString the encoded string

decodeFromHTML

Description

Returns a html decoded string

Definition

decodeFromHTML(string str)

Arguments

string str input string

Return Values

string decodedString the decoded string

ListUtil

Description

copyTable

Description

Returns a copy of the given table. It's not a deep copy!

Definition

copyTable(table sourceTable)

Arguments

table sourceTable the source table

Return Values

table copy the copied table

copyTableRecursively

Description

Returns a full copy of the given table, including all child tables

Definition

copyTableRecursively(table sourceTable)

Arguments

table sourceTable the source table

Return Values

table copy the copied table

addElementToList

Description

Adds an element to the list if element is not part of the list

Definition

addElementToList(table list, table element)

Arguments

table list a list

table element an element

removeElementFromList

Description

Removes an element from a given list

Definition

removeElementFromList(table list, table element)

Arguments

table list a list

table element an element

hasListElement

Description

Checks if list contains an element

Definition

hasListElement(table list, table element)

Arguments

table list a list

table element an element

Return Values

boolean isElementOfList true if element is part of the list, else false

findListElementFirstIndex

Description

Gets the index of the first occurrence

Definition

findListElementFirstIndex(table list, table element, object defaultReturn)

Arguments

table list a list

table element an element

object defaultReturn a default return value

Return Values

integer index index of first occurrence

areListsEqual

Description

Checks if two lists are equal

Definition

areListsEqual(table list1, table list2, boolean orderIndependent)

Arguments

table list1 list one

table list2 list two

boolean orderIndependent true the order of the elements has to be the same

Return Values

boolean areEqual true if lists are equal, else false

getRandomElement

Description

Gets a random element from given list

Definition

getRandomElement(table list)

Arguments

table list list to get element from

Return Values

any value value of the random element

listToSet

Description

Converts a list to a set

Definition

listToSet(table list)

Arguments

table list a list

Return Values

table set the converted set

setToList

Description

Converts a set to a list

Definition

setToList(table set)

Arguments

table set a set

Return Values

table list the converted list

setToHash

Description

Converts a set to a hash

Definition

setToHash(table set)

Arguments

table set a set

Return Values

table hash the converted hash

areSetsEqual

Description

Checks if two sets are equal

Definition

areSetsEqual(table set1, table set2)

Arguments

table set1 set one

table set2 set two

Return Values

boolean areEqual true if sets are equal, else false

isSubset

Description

Checks if set1 is a subset of set2

Definition

isSubset(table set1, table set2)

Arguments

table set1 set one

table set2 set two

Return Values

boolean isSubset true if set1 is a subset of set2

isRealSubset

Description

Checks if set1 is a real subset of set2

Definition

isRealSubset(table set1, table set2)

Arguments

table set1 set one

table set2 set two

Return Values

boolean isSubset true if set1 is a real subset of set2

hasSetIntersection

Description

Checks if set1 and set2 have an intersection

Definition

hasSetIntersection(table set1, table set2)

Arguments

table set1 set one

table set2 set two

Return Values

boolean hasIntersection true if set1 and set2 have an intersection

getSetIntersection

Description

Gets the intersection of two sets

Definition

getSetIntersection(table set1, table set2)

Arguments

table set1 set one

table set2 set two

Return Values

table intersection the intersection of both sets

getSetSubtraction

Description

Gets the substraction of two sets

Definition

getSetSubtraction(table set1, table set2)

Arguments

table set1 set one

table set2 set two

Return Values

table substraction the substraction of both set

getSetUnion

Description

Gets the union of two sets

Definition

getSetUnion(table set1, table set2)

Arguments

table set1 set one

table set2 set two

Return Values

table union the union of both sets

filter

Description

Filters a list. Returns a copy (see functional programming)

Definition

filter(table list, function closure)

Arguments

table list list to filter

function closure filter

Return Values

table filtered list (non-deep copy)

ifilter

Description

Filters a list. Returns a copy (see functional programming). Indexed version.

Definition

ifilter(table list, function closure)

Arguments

table list indexed list to filter

function closure filter

Return Values

table filtered list (non-deep copy)

MathUtil

Description

sign

Description

Returns the sign of the given value

Definition

sign(float x)

Arguments

float x a value

Return Values

integer sign the sign of the value

isNan

Description

Returns of the given value is nan

Definition

isNan(float value)

Arguments

float value a value

Return Values

boolean true if value is nan else false

round

Description

Returns the rounded value

Definition

round(float value, float precision)

Arguments

float value a value

float precision a precision

Return Values

number rounted value

degToRad

Description

Returns the radian value of a given angle in degress

Definition

degToRad(float degValue)

Arguments

float degValue angle in degrees

Return Values

float value radian angle

lerp

Description

Returns interpolated value between two given values

Definition

lerp(float v1, float v2, float alpha)

Arguments

float v1 start value

float v2 end value

float alpha alpha

Return Values

float value interpolated value

lerp3

Description

Returns interpolated value between two vectors

Definition

lerp3(float x1, float y1, float z1, float x2, float y2, float z2, float alpha)

Arguments

float x1 start x value
float y1 start y value
float z1 start z value
float x2 end x value
float y2 end y value
float z2 end z value
float alpha alpha

Return Values

float value interpolated value

inverseLerp

Description

Returns alpha based on current value

Definition

inverseLerp(float v1, float v2, float cv)

Arguments

float v1 start value
float v2 end value
float cv current value

Return Values

float alpha alpha

clamp

Description

Returns value between given min and max

Definition

clamp(float value, float minVal, float maxVal)

Arguments

float value to clamp
float minVal min value
float maxVal max value

Return Values

float value value

getIsOutOfBounds

Description

Returns true if the value is out of the given range

Definition

getIsOutOfBounds(float value, float limit1, float limit2)

Arguments

float value value
float limit1 limit 1
float limit2 limit 2

Return Values

float isOutOfBounds is out of bounds

getFlooredPercent

Description

Returns the floored percent

Definition

getFlooredPercent(float value, float maxValue)

Arguments

float value a value

float maxValue max value

Return Values

float value the floored percent value

getFlooredBounded

Description

Returns the floored clamped value

Definition

getFlooredBounded(float value, float minValue, float maxValue)

Arguments

float value a value

float minValue min value

float maxValue max value

Return Values

float value the floored clamped value

getValidLimit

Description

Returns a valid limit in the range of -pi to pi

Definition

getValidLimit(float limit)

Arguments

float limit a radian angle

Return Values

float angle the resized angle in the range -pi to pi

getAngleDifference

Description

Returns the difference between two rad angles

Definition

getAngleDifference(float alpha, float beta)

Arguments

float alpha a radian angle

float beta a radian angle

Return Values

float angle the radian difference in range -pi to pi

vector2Length

Description

Returns length of 2d vector

Definition

vector2Length(float x, float y)

Arguments

float x x

float y y

Return Values

float length length

vector2LengthSq

Description

Returns squared length of 2d vector

Definition

vector2LengthSq(float x, float y)

Arguments

float x x

float y y

Return Values

float length square length

vector2Normalize

Description

Returns normalized vector

Definition

vector2Normalize(float x, float y)

Arguments

float x x

float y y

Return Values

float x normalized x

float y normalized y

vector3Length

Description

Returns length of 3d vector

Definition

vector3Length(float x, float y, float z)

Arguments

float x x

float y y

float z z

Return Values

float length length

vector3LengthSq

Description

Returns squared length of 3d vector

Definition

vector3LengthSq(float x, float y, float z)

Arguments

float x x

float y y

float z z

Return Values

float length square length

vector3Normalize

Description

Returns normalized vector

Definition

vector3Normalize(float x, float y, float z)

Arguments

float x x

float y y

float z z

Return Values

float x normalized x

float y normalized y

float z normalized z

vector3SetLength

Description

Returns a scaled vector

Definition

vector3SetLength(float x, float y, float z, float length)

Arguments

float x x

float y y

float z z

float length scale length

Return Values

float x scaled x

float y scaled y

float z scaled z

vector3Clamp

Description

Returns a clamped vector based on min and max vector length

Definition

vector3Clamp(float x, float y, float z, float minVal, float maxVal)

Arguments

float x x

float y y
float z z
float minVal min length
float maxVal max length

Return Values

float x clamped x
float y clamped y
float z clamped z

vector3Lerp

Description

Returns a linear interpolated vector based on given alpha

Definition

vector3Lerp(float x1, float y1, float z1, float x2, float y2, float z2, float alpha)

Arguments

float x1 x1
float y1 y1
float z1 z1
float x2 x2
float y2 y2
float z2 z2
float alpha alpha value

Return Values

float x interpolated x
float y interpolated y
float z interpolated z

vector3ArrayLerp

Description

Returns a linear interpolated vector based on given alpha

Definition

vector3ArrayLerp(table v1, table v2, float alpha)

Arguments

table v1 vector1
table v2 vector2
float alpha alpha value

Return Values

float x interpolated x
float y interpolated y
float z interpolated z

vector3Transformation

Description

Transform a vector by matrix multiplication, the matrix is given row by row

Definition

vector3Transformation(float x, float y, float z, float m11, float m12, float m13, float m21, float m22, float m23, float m31, float m32, float m33)

Arguments

float x x
float y y
float z z
float m11 m11
float m12 m12
float m13 m13
float m21 m21
float m22 m22
float m23 m23
float m31 m31
float m32 m32
float m33 m33

Return Values

float x transformed x
float y transformed y
float z transformed z

dotProduct

Description

Returns the dot product of 2 vectors

Definition

dotProduct(float ax, float ay, float az, float bx, float by, float bz)

Arguments

float ax ax
float ay ay
float az az
float bx bx
float by by
float bz bz

Return Values

dot the dot product

crossProduct

Description

Returns the cross product of 2 vectors

Definition

crossProduct(float ax, float ay, float az, float bx, float by, float bz)

Arguments

float ax ax
float ay ay
float az az
float bx bx
float by by
float bz bz

Return Values

float x x

float y y

float z z

getYRotationFromDirection

Description

Returns the angle to rotate from the z axis around the y axis (if x==0, the angle is 0 or 180°). This is unlike the default specification, where the rotation is 0 at the x axis

Definition

getYRotationFromDirection(float dx, float dz)

Arguments

float dx dx

float dz dz

Return Values

float y rotation

getDirectionFromYRotation

Description

Returns the x and z direction based on the given y rotation

Definition

getDirectionFromYRotation(float rotY)

Arguments

float rotY y rotation

Return Values

float x x direction

float z z direction

getRotationLimitedVector2

Description

Returns an 2d vector limited by min and max rotation

Definition

getRotationLimitedVector2(float x, float y, float minRot, float maxRot)

Arguments

float x	x direction
float y	y direction
float minRot	min rot
float maxRot	max rot

Return Values

float x limited x direction

float z limited z direction

projectOnLine

Description

Returns the projected point on a given line

Definition

projectOnLine(float px, float pz, float lineX, float lineX, float normlineDirX, float normlineDirZ)

Arguments

float px	x position
float pz	z position
float lineX	x position
float lineX	x position
float normlineDirX	normalized x direction
float normlineDirZ	normalized z direction

Return Values

float x x position

float x z position

getProjectOnLineParameter

Description

Returns the dot product on a given line

Definition

getProjectOnLineParameter(float px, float pz, float lineX, float lineZ, float normlineDirX, float normlineDirZ)

Arguments

float px	x position
float pz	z position
float lineX	x position
float lineZ	x position
float normlineDirX	normalized x direction
float normlineDirZ	normalized z direction

Return Values

float dot product

quaternionMult

Description

Returns the multiplication of two quaternions

Definition

quaternionMult(float x, float y, float z, float w, float x1, float y1, float z1, float w1)

Arguments

float x	x1
float y	y1
float z	z1
float w	w1
float x1	x2
float y1	y2
float z1	z2
float w1	w2

Return Values

float x	x part of quaternion
float y	y part of quaternion
float z	z part of quaternion
float w	w part of quaternion

quaternionNormalized

Description

Returns the normalized quaternion

Definition

quaternionNormalized(float x, float y, float z, float w)

Arguments

float x x
float y y
float z z
float w w

Return Values

float x x part of quaternion
float y y part of quaternion
float z z part of quaternion
float w w part of quaternion

slerpQuaternion

Description

Returns the linear interpolated quaternion

Definition

slerpQuaternion(float x1, float y1, float z1, float w1, float x2, float y2, float z2, float w2, float t)

Arguments

float x1 x1
float y1 y1
float z1 z1
float w1 w1
float x2 x2
float y2 y2
float z2 z2
float w2 w2
float t interpolation value

Return Values

float x x part of quaternion
float y y part of quaternion
float z z part of quaternion
float w w part of quaternion

normalizeRotationForShortestPath

Description

Returns normalized rotation for the shortest path from current rotation to target rotation

Definition

normalizeRotationForShortestPath(float targetRotation, float curRotation)

Arguments

float targetRotation the target rotation

float curRotation the current rotation

Return Values

float rotation the final rotation

nlerpQuaternionShortestPath

Description

Returns the normalized shortest path from current quaternion to target quaternion

Definition

nlerpQuaternionShortestPath(float x1, float y1, float z1, float w1, float x2, float y2, float z2, float w2, float t)

Arguments

float x1 x1

float y1 y1

float z1 z1

float w1 w1

float x2 x2

float y2 y2

float z2 z2

float w2 w2

float t alpha

Return Values

float x x part of quaternion

float y y part of quaternion

float z z part of quaternion

float w w part of quaternion

slerpQuaternionShortestPath

Description

Returns the shortest path from current quaternion to target quaternion

Definition

slerpQuaternionShortestPath(float x1, float y1, float z1, float w1, float x2, float y2, float z2, float w2, float t)

Arguments

float x1 x1

float y1 y1

float z1 z1

float w1 w1

float x2 x2

float y2 y2

float z2 z2

float w2 w2

float t alpha

Return Values

float x x part of quaternion

float y y part of quaternion

float z z part of quaternion

float w w part of quaternion

quaternionMadShortestPath

Description

Returns the shortest path from current quaternion to target quaternion (mad = multiply and add)

Definition

quaternionMadShortestPath(float x, float y, float z, float w, float x1, float y1, float z1, float w1, float t)

Arguments

float x	x1
float y	y1
float z	z1
float w	w1
float x1	x2
float y1	y2
float z1	z2
float w1	w2
float t	alpha

Return Values

float x	x part of quaternion
float y	y part of quaternion
float z	z part of quaternion
float w	w part of quaternion

getDistanceToRectangle2D

Description

Returns the distance to a rectangle

Definition

getDistanceToRectangle2D(float posX, float posZ, float sx, float sz, float dx, float dz, float length, float widthHalf)

Arguments

float posX	x position
float posZ	z position
float sx	x position of rectangle
float sz	z position of rectangle
float dx	x direction of rectangle
float dz	z direction of rectangle
float length	length of rectangle
float widthHalf	half width of rectangle

Return Values

float distance	distance to rectangle

getSignedDistanceToLineSegment2D

Description

Returns the distance to a line segment

Definition

getSignedDistanceToLineSegment2D(float x, float z, float sx, float sz, float dx, float dz, float length)

Arguments

float x	x position
float z	z position
float sx	x position of rectangle
float sz	z position of rectangle
float dx	x direction of rectangle
float dz	z direction of rectangle
float length	length of rectangle

Return Values

float distance distance to line segment

getLineLineIntersection2D

Description

Returns the line-line intersection point

Definition

getLineLineIntersection2D(float x1, float z1, float dirX1, float dirZ1, float x2, float z2, float dirX2, float dirZ2)

Arguments

float x1	x1 position
float z1	z1 position
float dirX1	line1 x direction
float dirZ1	line1 z direction
float x2	x2 position
float z2	z2 position
float dirX2	line2 x direction
float dirZ2	line2 z direction

Return Values

boolean hasIntersection true if both lines intersect

float	t1	x position
float	t2	z position

hasRectangleLineIntersection2D

Description

Returns if rectangle and line have intersection

Definition

hasRectangleLineIntersection2D(float x1, float z1, float dirX1, float dirZ1, float dirX2, float dirZ2, float x3, float z3, float dirX3, float dirZ3)

Arguments

float x1	x1 position
float z1	z1 position
float dirX1	rectangle x1 direction
float dirZ1	rectangle z1 direction
float dirX2	rectangle x2 direction
float dirZ2	rectangle z2 direction

float x3 line x position
float z3 line z position
float dirX3 line x direction
float dirZ3 line z direction

Return Values

boolean hasIntersection true if the line segment is completly in the rectangle OR if it intersects with one of the four rectangle sides

getCircleCircleIntersection

Description

Returns circle-circle intersection points

Definition

getCircleCircleIntersection(float x1, float y1, float r1, float x2, float y2, float r2)

Arguments

float x1 x1 position
float y1 y1 position
float r1 radius 1
float x2 x2 position
float y2 y2 position
float r2 radius 2

Return Values

float pos1X x pos of intersection 1, else nil
float pos1Y y pos of intersection 1, else nil
float pos2X x pos of intersection 2, else nil
float pos2Z z pos of intersection 2, else nil

hasSphereSphereIntersection

Description

Returns if 2 spheres have an intersection

Definition

hasSphereSphereIntersection(float x1, float y1, float z1, float r1, float x2, float y2, float z2, float r2)

Arguments

float x1 x1 position
float y1 y1 position
float z1 z1 position
float r1 radius 1
float x2 x2 position
float y2 y2 position
float z2 z2 position
float r2 radius 2

Return Values

boolean hasIntersection true if spheres have an intersection else false

areaToHa

Description

Converts a area to hectars

Definition

areaToHa(float area, float pixelToSqm)

Arguments

float area area

float pixelToSqm pixel density

Return Values

float area area in hectars

inchToM

Description

Converts an inch distance to meters

Definition

inchToM(inchValue the)

Arguments

inchValue the inch value to convert

mToInch

Description

Converts an meter distance to inch

Definition

mToInch(float meterValue)

Arguments

float meterValue the meter value to convert

getBrightnessFromColor

Description

Returns the brightness of a color [0..1]

Definition

getBrightnessFromColor(float r, float g, float b)

Arguments

float r red value

float g green value

float b blue value

Return Values

float brightness brightness

ObjectChangeUtil

Description

ObjectChangeUtil

loadObjectChangeFromXML

Description

Load object change from xml

Definition

loadObjectChangeFromXML(integer xmlFile, string key, table objects, integer rootNode, table parent)

Arguments

integer xmlFile file id of xml file

string	key	key
table	objects	table to insert loaded objects
integer	rootNode	id of root node
table	parent	parent

Code

```
14  function ObjectChangeUtil.loadObjectChangeFromXML(xmlFile, key, objects, rootNode, parent)
15  local i = 0
16  while true do
17  local nodeKey = string.format(key .. ".objectChange(%d)", i)
18  if not hasXMLProperty(xmlFile, nodeKey) then
19  break
20  end
21  local node = I3DUtil.indexToObject(rootNode, getXMLString(xmlFile, nodeKey .. "#node"), parent.i3dMappings)
22  if node ~= nil then
23  local object = {}
24  object.node = node
25  ObjectChangeUtil.loadValuesFromXML(xmlFile, nodeKey, node, object, parent)
26  table.insert(objects, object)
27  end
28  i = i + 1
29  end
30  end
```

loadValuesFromXML

Description

Load object values from xml

Definition

loadValuesFromXML(integer xmlFile, string key, integer node, table object, table parent)

Arguments

integer	xmlFile	file id of xml file
string	key	key
integer	node	node id to load from
table	object	table to insert loaded data
table	parent	parent

Code

```
39  function ObjectChangeUtil.loadValuesFromXML(xmlFile, key, node, object, parent)
40  object.visibilityActive = getXMLBool(xmlFile, key.."#visibilityActive")
41  object.visibilityInactive = getXMLBool(xmlFile, key.."#visibilityInactive")
```

```
42  object.translationActive = StringUtil.getVectorNFromString(getXMLString(xmlFile, key.."#translationActive"), 3)
43  object.translationInactive = StringUtil.getVectorNFromString(getXMLString(xmlFile, key.."#translationInactive"), 3)
44  object.rotationActive = StringUtil.getRadiansFromString(getXMLString(xmlFile, key.."#rotationActive"), 3)
45  object.rotationInactive = StringUtil.getRadiansFromString(getXMLString(xmlFile, key.."#rotationInactive"), 3)
46  object.scaleActive = StringUtil.getVectorNFromString(getXMLString(xmlFile, key.."#scaleActive"), 3)
47  object.scaleInactive = StringUtil.getVectorNFromString(getXMLString(xmlFile, key.."#scaleInactive"), 3)
48
49  XMLUtil.checkDeprecatedXMLElements(xmlFile, "", key.."#collisionActive", key.."#compoundChildActive or #rigidBodyTypeActive") --FS17 to FS19
50  XMLUtil.checkDeprecatedXMLElements(xmlFile, "", key.."#collisionInactive", key.."#compoundChildInactive or #rigidBodyTypeInactive") --FS17 to FS19
51
52  object.massActive = nil
53  object.massInactive = nil
54  local massActive = getXMLFloat(xmlFile, key.."#massActive")
55  if massActive ~= nil then
56  object.massActive = massActive / 1000
57  end
58  local massInactive = getXMLFloat(xmlFile, key.."#massInactive")
59  if massInactive ~= nil then
60  object.massInactive = massInactive / 1000
61  end
62  object.centerOfMassActive = StringUtil.getVectorNFromString(getXMLString(xmlFile, key.."#centerOfMassActive"), 3)
63  object.centerOfMassInactive = StringUtil.getVectorNFromString(getXMLString(xmlFile, key.."#centerOfMassInactive"), 3)
64  object.compoundChildActive = getXMLBool(xmlFile, key.."#compoundChildActive")
65  object.compoundChildInactive = getXMLBool(xmlFile, key.."#compoundChildInactive")
```

```
66  object.rigidBodyTypeActive = getXMLString(xmlFile,
    key.."#rigidBodyTypeActive")
67  object.rigidBodyTypeInactive = getXMLString(xmlFile,
    key.."#rigidBodyTypeInactive")
68
69  if object.rigidBodyTypeActive ~= nil then
70  local t = object.rigidBodyTypeActive
71  if t ~= "Static" and t ~= "Dynamic" and t ~= "Kinematic" and t ~=
    "NoRigidBody" then
72  g_logManager:warning("Invalid rigidBodyTypeActive '%s' for object
    change node '%s'. Use 'Static', 'Dynamic', 'Kinematic' or
    'NoRigidBody'!", t, key)
73  object.rigidBodyTypeActive = nil
74  end
75  end
76  if object.rigidBodyTypeInactive ~= nil then
77  local t = object.rigidBodyTypeInactive
78  if t ~= "Static" and t ~= "Dynamic" and t ~= "Kinematic" and t ~=
    "NoRigidBody" then
79  g_logManager:warning("Invalid rigidBodyTypeInactive '%s' for
    object change node '%s'. Use 'Static', 'Dynamic', 'Kinematic' or
    'NoRigidBody'!", t, key)
80  object.rigidBodyTypeInactive = nil
81  end
82  end
83
84  if parent ~= nil and parent.loadObjectChangeValuesFromXML ~= nil
    then
85  parent:loadObjectChangeValuesFromXML(xmlFile, key, node, object)
86  end
87  end
```

setObjectChanges

Description

Set object changes

Definition

setObjectChanges(table object, boolean isActive, table target, function updateFunc)

Arguments

table	object	objects to change
boolean	isActive	is active
table	target	target for updateFunc
function	updateFunc	function to update

Code

```
95  function ObjectChangeUtil.setObjectChanges(objects, isActive,
    target, updateFunc)
96  for _, object in pairs(objects) do
97  ObjectChangeUtil.setObjectChange(object, isActive, target,
    updateFunc)
98  end
99  end
```

setObjectChange

Description

Set object change

Definition

setObjectChange(table object, boolean isActive, table target, function updateFunc)

Arguments

table	object	objects to change
boolean	isActive	is active
table	target	target for updateFunc
function	updateFunc	function to update

Code

```
107 function ObjectChangeUtil.setObjectChange(object, isActive,
    target, updateFunc)
108 if isActive then
109 if object.visibilityActive ~= nil then
110 setVisibility(object.node, object.visibilityActive)
111 end
112 if object.translationActive ~= nil then
113 setTranslation(object.node, object.translationActive[1],
    object.translationActive[2], object.translationActive[3])
114 end
115 if object.rotationActive ~= nil then
116 setRotation(object.node, object.rotationActive[1],
    object.rotationActive[2], object.rotationActive[3])
117 end
118 if object.scaleActive ~= nil then
119 setScale(object.node, object.scaleActive[1],
    object.scaleActive[2], object.scaleActive[3])
120 end
121 if object.massActive ~= nil then
122 setMass(object.node, object.massActive)
123
124 if target ~= nil and target.components ~= nil then
125 for _, component in ipairs(target.components) do
126 if component.node == object.node then
```

```
component.defaultMass = object.massActive
target:setMassDirty()
end
end
end
end
if object.centerOfMassActive ~= nil then
setCenterOfMass(object.node, unpack(object.centerOfMassActive))
end
if object.compoundChildActive ~= nil then
setIsCompoundChild(object.node, object.compoundChildActive)
end
if object.rigidBodyTypeActive ~= nil then
setRigidBodyType(object.node, object.rigidBodyTypeActive)
end
else
if object.visibilityInactive ~= nil then
setVisibility(object.node, object.visibilityInactive)
end
if object.translationInactive ~= nil then
setTranslation(object.node, object.translationInactive[1],
object.translationInactive[2], object.translationInactive[3])
end
if object.rotationInactive ~= nil then
setRotation(object.node, object.rotationInactive[1],
object.rotationInactive[2], object.rotationInactive[3])
end
if object.scaleInactive ~= nil then
setScale(object.node, object.scaleInactive[1],
object.scaleInactive[2], object.scaleInactive[3])
end
if object.massInactive ~= nil then
setMass(object.node, object.massInactive)

if target ~= nil and target.components ~= nil then
for _, component in ipairs(target.components) do
if component.node == object.node then
component.defaultMass = object.massInactive
target:setMassDirty()
end
end
```

```
165 end
166 end
167 if object.centerOfMassInactive ~= nil then
168 setCenterOfMass(object.node, unpack(object.centerOfMassInactive))
169 end
170 if object.compoundChildInactive ~= nil then
171 setIsCompoundChild(object.node, object.compoundChildInactive)
172 end
173 if object.rigidBodyTypeInactive ~= nil then
174 setRigidBodyType(object.node, object.rigidBodyTypeInactive)
175 end
176 end
177 if target ~= nil then
178 if target.setObjectChangeValues ~= nil then
179 target:setObjectChangeValues(object, isActive)
180 end
181 if updateFunc ~= nil then
182 updateFunc(target, object.node)
183 end
184 end
185 end
```

updateObjectChanges

Description

Update object changes

Definition

updateObjectChanges(integer xmlFile, string key, integer configKey, integer rootNode, table parent)

Arguments

integer	xmlFile	file id of xml file
string	key	key
integer	configKey	id of used config
integer	rootNode	id of root node
table	parent	parent

Code

```
194 function ObjectChangeUtil.updateObjectChanges(xmlFile, key,
    configKey, rootNode, parent)
195 local i = 0
196 local activeI = (configKey - 1)
197 while true do
198 local objectChangeKey = string.format(key.."(%d)", i)
199 if not hasXMLProperty(xmlFile, objectChangeKey) then
```

```
200 break
201 end
202 if i ~= activeI then
203 local objects = {}
204 ObjectChangeUtil.loadObjectChangeFromXML(xmlFile,
    objectChangeKey, objects, rootNode, parent)
205 ObjectChangeUtil.setObjectChanges(objects, false, parent)
206 end
207 i = i + 1
208 end
209
210 -- Set the active config last so that it can overwrite settings
    of inactive configurations
211 if i > activeI then
212 local objectChangeKey = string.format(key.."(%d)", activeI)
213 local objects = {}
214 ObjectChangeUtil.loadObjectChangeFromXML(xmlFile,
    objectChangeKey, objects, rootNode, parent)
215 ObjectChangeUtil.setObjectChanges(objects, true, parent)
216 end
217 end
```

ParticleUtil

Description

ParticleUtil

loadParticleSystem

Description

Load particle system

Definition

loadParticleSystem(integer xmlId, table particleSystem, string baseString, table linkNodes, boolean defaultEmittingState, string defaultPsFile, string baseDir, integer defaultLinkNode)

Arguments

integer	xmlId	id of xml file
table	particleSystem	table to add particle system data
string	baseString	base string to load data from xml
table	linkNodes	link nodes
boolean	defaultEmittingState	default emitting state
string	defaultPsFile	path to default ps file
string	baseDir	base directory
integer	defaultLinkNode	id of default link node

deleteParticleSystem

Description

Delete particle system

Definition

deleteParticleSystem(table particleSystem)

Arguments

table particleSystem particle system

deleteParticleSystems

Description

Delete particle systems

Definition

deleteParticleSystems(table particleSystems)

Arguments

table particleSystems particle systems

setEmittingState

Description

Set emitting state of particle system

Definition

setEmittingState(table particleSystem, boolean state, boolean resetStartTimer, boolean resetStopTimer)

Arguments

table	particleSystem	particle system
boolean	state	emitting state
boolean	resetStartTimer	reset start timer
boolean	resetStopTimer	reset stop timer

getParticleSystemAverageSpeed

Description

Returns average speed of particle system

Definition

getParticleSystemAverageSpeed(table particleSystem)

Arguments

table particleSystem particle system

Return Values

float averageSpeed average speed

setParticleSystemTimeScale

Description

Setting time scale of particle system

Definition

setParticleSystemTimeScale(table particleSystem, float scale)

Arguments

table	particleSystem	particle system
float	scale	time scale

setEmitCountScale

Description

Setting emit count scale of particle system

Definition

setEmitCountScale(table particleSystem, float scale)

Arguments

table particleSystem particle system

float scale emit count scale

setParticleLifespan

Description

Setting particle system lifespan

Definition

setParticleLifespan(table particleSystem, float lifespan)

Arguments

table particleSystem particle system

float lifespan lifespan

setParticleStartStopTime

Description

Sets start and stop time of particle system

Definition

setParticleStartStopTime(table particleSystem, float startTime, float stopTime)

Arguments

table particleSystem particle system

float startTime start time

float stopTime stop time

getParticleSystemSpeed

Description

Returns speed of particle system

Definition

getParticleSystemSpeed(table particleSystem)

Arguments

table particleSystem particle system

Return Values

float speed speed

setParticleSystemSpeed

Description

Sets speed of particle system

Definition

setParticleSystemSpeed(table particleSystem, float speed)

Arguments

table particleSystem particle system

float speed speed

getParticleSystemSpeedRandom

Description

Returns speed random of particle system

Definition

getParticleSystemSpeedRandom(table particleSystem)

Arguments

table particleSystem particle system

Return Values

float speedRandom speed random

setParticleSystemSpeedRandom

Description

Sets speed random of particle system

Definition

setParticleSystemSpeedRandom(table particleSystem, float speedRandom)

Arguments

table particleSystem particle system

float speedRandom speed random

getParticleSystemNormalSpeed

Description

Returns normal speed of particle system

Definition

getParticleSystemNormalSpeed(table particleSystem)

Arguments

table particleSystem particle system

Return Values

float normalSpeed normal speed

setParticleSystemNormalSpeed

Description

Sets normal speed of particle system

Definition

setParticleSystemNormalSpeed(table particleSystem, float normalSpeed)

Arguments

table particleSystem particle system

float normalSpeed normal speed

getParticleSystemTangentSpeed

Description

Returns tangent speed of particle system

Definition

getParticleSystemTangentSpeed(table particleSystem)

Arguments

table particleSystem particle system

Return Values

float tangentSpeed tangent speed

setParticleSystemTangentSpeed

Description

Sets tangent speed of particle system

Definition

setParticleSystemTangentSpeed(table particleSystem, float tangentSpeed)

Arguments

table particleSystem particle system
float tangentSpeed tangent speed

getParticleSystemSpriteScaleX

Description

Returns X sprite scale of particle system

Definition

getParticleSystemSpriteScaleX(table particleSystem)

Arguments

table particleSystem particle system

Return Values

float spriteScaleX X sprite scale

setParticleSystemSpriteScaleX

Description

Sets X sprite scale of particle system

Definition

setParticleSystemSpriteScaleX(table particleSystem, float spriteScaleX)

Arguments

table particleSystem particle system
float spriteScaleX X sprite scale

getParticleSystemSpriteScaleY

Description

Returns Y sprite scale of particle system

Definition

getParticleSystemSpriteScaleY(table particleSystem)

Arguments

table particleSystem particle system

Return Values

float spriteScaleY Y sprite scale

setParticleSystemSpriteScaleY

Description

Sets Y sprite scale of particle system

Definition

setParticleSystemSpriteScaleY(table particleSystem, float spriteScaleY)

Arguments

table particleSystem particle system
float spriteScaleY Y sprite scale

getParticleSystemSpriteScaleXGain

Description

Returns X sprite scale gain of particle system

Definition

getParticleSystemSpriteScaleXGain(table particleSystem)

Arguments

table particleSystem particle system

Return Values

float spriteScaleXGain X sprite scale gain

setParticleSystemSpriteScaleXGain

Description

Sets X sprite scale gain of particle system

Definition

setParticleSystemSpriteScaleXGain(table particleSystem, float spriteScaleXGain)

Arguments

table particleSystem particle system

float spriteScaleXGain X sprite scale gain

getParticleSystemSpriteScaleYGain

Description

Returns Y sprite scale gain of particle system

Definition

getParticleSystemSpriteScaleYGain(table particleSystem)

Arguments

table particleSystem particle system

Return Values

float spriteScaleYGain Y sprite scale gain

setParticleSystemSpriteScaleYGain

Description

Sets Y sprite scale gain of particle system

Definition

setParticleSystemSpriteScaleYGain(table particleSystem, float spriteScaleYGain)

Arguments

table particleSystem particle system

float spriteScaleYGain Y sprite scale gain

resetNumOfEmittedParticles

Description

Resets number of emitted particles

Definition

resetNumOfEmittedParticles(table particleSystem)

Arguments

table particleSystem particle system

PlacementUtil

Description

isInsideRestrictedZone

Description

Check if a placeable object at a given position lies within a restricted zone or below the water line.

Definition

isInsideRestrictedZone(places Array, placable Placeable, x Testing, y Testing, Z Testing)

Arguments

places	Array	of restricted zones (see PlacementUtil.createRestrictedZone() for data definition)
placable	Placeable	instance
x	Testing	X position in world space
y	Testing	Y position in world space
Z	Testing	Z position in world space

isInsidePlacementPlaces

Description

Check if a placeable object at a given position lies within store or loading spaces.

Definition

isInsidePlacementPlaces(places Array, placable Placeable, x Testing, y Testing, Z Testing)

Arguments

places	Array	of places (see PlacementUtil.createPlace() for data definition)
placable	Placeable	instance
x	Testing	X position in world space
y	Testing	Y position in world space
Z	Testing	Z position in world space

hasObjectOverlap

Description

Test if a placeable's placement test volume at a given position and orientation would collide with another object.
The test is performed with a placeable's "placementTestSizeX" and "placementTestSizeZ" attribute values. The given
coordinates must represent a point either on an object or on the terrain to generate a correct result.

Definition

hasObjectOverlap(Placeable Placeable, x Testing, y Testing, Z Testing, rotY Testing)

Arguments

Placeable	Placeable	instance
x	Testing	X position in world space
y	Testing	Y position in world space
Z	Testing	Z position in world space
rotY	Testing	rotation in radians

Return Values

True if there is a collision, false otherwise

hasOverlapWithPoint

Description

Test if a placeable at given position and rotation would overlap with any player position

Definition

hasOverlapWithPoint()

loadPlaceableFromXML

Description

Load a placeable object from an XML definition file.

Definition

loadPlaceableFromXML(xmlFilename Path, x New, y New, Z New, rx New, ry New, rz New, moveMode True)

Arguments

xmlFilename	Path to placeable XML definition file
x	New object X position in world space
y	New object Y position in world space
Z	New object Z position in world space
rx	New object X rotation in radians
ry	New object Y rotation in radians
rz	New object Z rotation in radians
moveMode	True if the placeable can be moved around for placement, false otherwise

Return Values

Placeable object instance or nil if no object could be created		
True	if	no placeable object could be created at the requested position because there was no valid space, false otherwise

loadPlaceable

Description

Load a placeable object from a validated XML definition.

Definition

loadPlaceable(placeableType Type, xmlFilename Path, x New, y New, Z New, rx New, ry New, rz New, moveMode True)

Arguments

placeableType	Type	name of the requested placeable object
xmlFilename	Path	to placeable XML definition file
x	New	object X position in world space
y	New	object Y position in world space
Z	New	object Z position in world space
rx	New	object X rotation in radians
ry	New	object Y rotation in radians
rz	New	object Z rotation in radians
moveMode	True	if the placeable can be moved around for placement, false if the placeable is to be created on the spot

Return Values

Placeable object instance or nil if no object could be created		
True	if	no placeable object could be created at the requested position because there was no valid space, false otherwise

getPlaceableAreaByNodes

Description

Get a placeable area based on start, width and height nodes.
The area is returned as an array of 9 numbers, i.e. 1 world space point and 2 world space vectors. The point denotes
the area origin and the vectors define the orientation and length of the area sides.
return {worldStartX, worldStartY, worldStartZ, side1X, side1Y, side1Z, side2X, side2Y, side2Z}

Definition

getPlaceableAreaByNodes()

StringUtil
Description

getVectorFromString

Description

Returns vector from string separated by a whitespace

Definition

getVectorFromString(string input)

Arguments

string input input

Return Values

any_type unpackedValues returns unpacked values found in string

getVectorNFromString

Description

Returns vector N from string separated by a whitespace

Definition

getVectorNFromString(string input, integer num)

Arguments

string input input
integer num number of values

Return Values

table values values

getRadiansFromString

Description

Returns radian vector N from string separated by a whitespace

Definition

getRadiansFromString(string input, integer num)

Arguments

string input input
integer num number of values

Return Values

table values radian values

parseList

Description

Returns an array of elements from the string, split at separator, and passed through lambda

Definition

parseList(string str, string seprator, function lambda)

Arguments

string str input string
string seprator list separator
function lambda function to apply to each list element, e.g. tonumber

splitString

Description

Returns text elements splitted by splitPattern

Definition

splitString(string splitPattern, string text)

Arguments

string splitPattern splitting pattern

string text a text

Return Values

table result text elements

startsWith

Description

Returns if a string starts with given sequence

Definition

startsWith(string str, string find)

Arguments

string str a string

string find the start pattern

Return Values

boolean startsWidth true if given string starts with pattern else false

endsWith

Description

Returns if a string ends with given sequence

Definition

endsWith(string str, string find)

Arguments

string str a string

string find the start pattern

Return Values

boolean startsWidth true if given string ends with pattern else false

trim

Description

Returns a trimed string (no whitespaces at start and end)

Definition

trim(string str)

Arguments

string str a string

Return Values

string trimedString trimed text

Utils

Description

Utils

getNoNil

Description

Returns second parameter if the first is nil

Definition

getNoNil(any_type value, any_type setTo)

Arguments

any_type value value

any_type setTo set to value

Return Values

any_type value not nil value

XMLUtil

Description

XMLUtil
Class for various xml operations

getXMLStringWithDefault

Description

Searches for an XML attribute in two places

Definition

getXMLStringWithDefault(DomXMLFile xmlFile, string key, string defkey, string overridekey, string attr)

Arguments

DomXMLFile	xmlFile	the XML file to search
string	key	the main key to look under
string	defkey	the secondary key to look under, if the main key fails (can be nil)
string	overridekey	the override key, which can override the main key (can be nil)
string	attr	the attribute to find

getXMLIntWithDefault

Description

Searches for an XML int attribute in two places

Definition

getXMLIntWithDefault(DomXMLFile xmlFile, string key, string defkey, string overridekey, string attr)

Arguments

DomXMLFile	xmlFile	the XML file to search
string	key	the main key to look under
string	defkey	the secondary key to look under, if the main key fails (can be nil)
string	overridekey	the override key, which can override the main key (can be nil)
string	attr	the attribute to find

getXMLFloatWithDefault

Description

Searches for an XML float attribute in two places

Definition

getXMLFloatWithDefault(DomXMLFile xmlFile, string key, string defkey, string overridekey, string attr)

Arguments

DomXMLFile	xmlFile	the XML file to search
string	key	the main key to look under

string	defkey	the secondary key to look under, if the main key fails (can be nil)
string	overridekey	the override key, which can override the main key (can be nil)
string	attr	the attribute to find

getXMLOverwrittenValue

Description

Tries to retrieve xml value with given xmlFunc, returns nil if xml-value equals "-" otherwise xml-value if exists otherwise fallbackValue

Definition

getXMLOverwrittenValue(DomXMLFile xmlFile, string key, string subKey, args param, function xmlFunc, fallbackValue fallbackValue, valueFunc additional, ... variable)

Arguments

DomXMLFile	xmlFile	the XML file to search
string	key	the main key
string	subKey	the secondary key to append to the main key
args	param	parameters passed to the xmlFunc
function	xmlFunc	the xml function to be used for accessing the key
fallbackValue	fallbackValue	to be returned if key does not exist
valueFunc	additional	function to be applied to value retrieved from xml file
...	variable	number of arguments additionally passed to valueFunc

loadDataFromMapXML

Description

Loads the map specific data from the map xml. Can either be inlined in the map xml directly (<xmlKey>...</xmlKey>) or can be referenced with an external file (<xmlKey filename="..." />)

Definition

loadDataFromMapXML(DomXMLFile mapXMLFile, string xmlKey, object loadTarget, function loadFunc, vararg vararg)

Arguments

DomXMLFile	mapXMLFile	the map XML file
string	xmlKey	the xml key specifying the data to be loaded
object	loadTarget	the object to call the load function on
function	loadFunc	the load function to be called with the data
vararg	vararg	any data that should be passed to the load function

AIVehicleUtil

Description

Util class for various ai vehicle functions

driveToPoint

Description

Drive vehicle to given point

Definition

driveToPoint(table self, float dt, float acceleration, boolean allowedToDrive, boolean moveForwards, float tX, float tY, float maxSpeed, boolean doNotSteer)

Arguments

table	self	object of vehicle to move

float	dt	time since last call in ms
float	acceleration	acceleration
boolean	allowedToDrive	allowed to drive
boolean	moveForwards	move forwards
float	tX	local space x position
float	tY	local space y position
float	maxSpeed	speed limit
boolean	doNotSteer	do not steer

Code

```
28  function AIVehicleUtil.driveToPoint(self, dt, acceleration,
    allowedToDrive, moveForwards, tX, tZ, maxSpeed, doNotSteer)
29
30  if self.firstTimeRun then
31
32  if allowedToDrive then
33
34  local tX_2 = tX * 0.5
35  local tZ_2 = tZ * 0.5
36
37  local d1X, d1Z = tZ_2, -tX_2
38  if tX > 0 then
39  d1X, d1Z = -tZ_2, tX_2
40  end
41
42  local hit,_,f2 = MathUtil.getLineLineIntersection2D(tX_2,tZ_2,
    d1X,d1Z, 0,0, tX, 0)
43
44  if doNotSteer == nil or not doNotSteer then
45  local rotTime = 0
46  if hit and math.abs(f2) < 100000 then
47  local radius = tX * f2
48  rotTime = self.wheelSteeringDuration * ( math.atan(1/radius) /
    math.atan(1/self.maxTurningRadius) )
49  end
50
51  local targetRotTime
52  if rotTime >= 0 then
53  targetRotTime = math.min(rotTime, self.maxRotTime)
54  else
55  targetRotTime = math.max(rotTime, self.minRotTime)
56  end
57
```

```
58  if targetRotTime > self.rotatedTime then
59  self.rotatedTime = math.min(self.rotatedTime +
    dt*self:getAISteeringSpeed(), targetRotTime)
60  else
61  self.rotatedTime = math.max(self.rotatedTime -
    dt*self:getAISteeringSpeed(), targetRotTime)
62  end
63
64  -- adjust maxSpeed
65  local steerDiff = targetRotTime - self.rotatedTime
66  local fac = math.abs(steerDiff) / math.max(self.maxRotTime, -
    self.minRotTime)
67  local speedReduction = 1.0 - math.pow(fac, 0.25)
68
69  -- if the speed is decreased to less than 1.5km/h we do not
    accelrate anymore
70  if maxSpeed * speedReduction < 1.5 then
71  acceleration = 0
72  speedReduction = 1.5 / maxSpeed
73  end
74
75  maxSpeed = maxSpeed * speedReduction
76  end
77  end
78
79  self:getMotor():setSpeedLimit(math.min(maxSpeed,
    self:getCruiseControlSpeed()))
80  if self:getCruiseControlState() ~=
    Drivable.CRUISECONTROL_STATE_ACTIVE then
81  self:setCruiseControlState(Drivable.CRUISECONTROL_STATE_ACTIVE)
82  end
83
84  if not allowedToDrive then
85  acceleration = 0
86  end
87  if not moveForwards then
88  acceleration = -acceleration
89  end
90
91  WheelsUtil.updateWheelsPhysics(self, dt,
    self.lastSpeedReal*self.movingDirection, acceleration, not
    allowedToDrive, true)
```

```
92
93  end
94
95  end
```

driveInDirection

Description

Drive in given direction

Definition

driveInDirection(table self, float dt, float steeringAngleLimit, float acceleration, float slowAcceleration, float slowAngleLimit, boolean allowedToDrive, boolean moveForwards, float lx, float lz, float maxSpeed, float slowDownFactor)

Arguments

table	self	object of vehicle
float	dt	time since last call in ms
float	steeringAngleLimit	limit for steering angle
float	acceleration	acceleration
float	slowAcceleration	slow acceleration
float	slowAngleLimit	limit of slow angle
boolean	allowedToDrive	allow to drive
boolean	moveForwards	move forwards
float	lx	x direction
float	lz	z direction
float	maxSpeed	max speed
float	slowDownFactor	slow down factor

Code

```
111 function AIVehicleUtil.driveInDirection(self, dt, steeringAngleLimit, acceleration, slowAcceleration, slowAngleLimit, allowedToDrive, moveForwards, lx, lz, maxSpeed, slowDownFactor)
112
113 local angle = 0
114 if lx ~= nil and lz ~= nil then
115 local dot = lz
116 angle = math.deg(math.acos(dot))
117 if angle < 0 then
118 angle = angle+180
119 end
120
121 local turnLeft = lx > 0.00001
122 if not moveForwards then
123 turnLeft = not turnLeft
124 end
125
```

```
126 local targetRotTime
127 if turnLeft then
128 --rotate to the left
129 targetRotTime =
    self.maxRotTime*math.min(angle/steeringAngleLimit, 1)
130 else
131 --rotate to the right
132 targetRotTime =
    self.minRotTime*math.min(angle/steeringAngleLimit, 1)
133 end
134
135 if targetRotTime > self.rotatedTime then
136 self.rotatedTime = math.min(self.rotatedTime +
    dt*self:getAISteeringSpeed(), targetRotTime)
137 else
138 self.rotatedTime = math.max(self.rotatedTime -
    dt*self:getAISteeringSpeed(), targetRotTime)
139 end
140 end
141
142
143 if self.firstTimeRun then
144 local acc = acceleration
145 if maxSpeed ~= nil and maxSpeed ~= 0 then
146 if math.abs(angle) >= slowAngleLimit then
147 maxSpeed = maxSpeed * slowDownFactor
148 end
149 self.motor:setSpeedLimit(maxSpeed)
150
151 if self.cruiseControl.state ~=
    Drivable.CRUISECONTROL_STATE_ACTIVE then
152 self:setCruiseControlState(Drivable.CRUISECONTROL_STATE_ACTIVE)
153 end
154 else
155 if math.abs(angle) >= slowAngleLimit then
156 acc = slowAcceleration
157 end
158 end
159 if not allowedToDrive then
160 acc = 0
161 end
```

```
162  if not moveForwards then
163  acc = -acc
164  end
165  WheelsUtil.updateWheelsPhysics(self, dt,
     self.lastSpeedReal*self.movingDirection, acc, not allowedToDrive,
     true)
166  end
167  end
```

getDriveDirection

Description

Returns drive direction

Definition

getDriveDirection(integer refNode, float x, float y, float z)

Arguments

integer refNode id of ref node

float	x	world x
float	y	world y
float	z	world z

Return Values

float lx x direction

float lz z direction

Code

```
177  function AIVehicleUtil.getDriveDirection(refNode, x, y, z)
178  local lx, _, lz = worldToLocal(refNode, x, y, z)
179
180  local length = MathUtil.vector2Length(lx, lz)
181  if length > 0.00001 then
182  length = 1/length
183  lx = lx*length
184  lz = lz*length
185  end
186  return lx, lz
187  end
```

getAverageDriveDirection

Description

Returns average drive direction between 2 given vectors

Definition

getAverageDriveDirection(integer refNode, float x, float y, float z, float x2, float y2, float z2)

Arguments

integer refNode id of ref node

float	x	world x 1
float	y	world y 1

float	z	world z 1
float	x2	world x 2
float	y2	world y 2
float	z2	world z 2

Return Values

float lx average x direction

float lz average z direction

Code

```
200 function AIVehicleUtil.getAverageDriveDirection(refNode, x, y, z, x2, y2, z2)
201 local lx, _, lz = worldToLocal(refNode, (x+x2)*0.5, (y+y2)*0.5, (z+z2)*0.5)
202
203 local length = MathUtil.vector2Length(lx, lz)
204 if length > 0.00001 then
205 lx = lx/length
206 lz = lz/length
207 end
208 return lx, lz, length
209 end
```

getAttachedImplementsAllowTurnBackward

Description

Returns if trailer or trailer low is attached

Definition

getAttachedImplementsAllowTurnBackward(table vehicle)

Arguments

table vehicle vehicle to check

Return Values

boolean isAttached is attached

Code

```
215 function AIVehicleUtil.getAttachedImplementsAllowTurnBackward(vehicle)
216 if vehicle.getAIAllowTurnBackward ~= nil then
217 if not vehicle:getAIAllowTurnBackward() then
218 return false
219 end
220 end
221
222 if vehicle.getAttachedImplements ~= nil then
223 for _, implement in pairs(vehicle:getAttachedImplements()) do
224 local object = implement.object
225 if object ~= nil then
226 if object.getAIAllowTurnBackward ~= nil then
```

```
227 if not object:getAIAllowTurnBackward() then
228 return false
229 end
230 end
231
232 if not AIVehicleUtil.getAttachedImplementsAllowTurnBackward(object) then
233 return false
234 end
235 end
236 end
237 end
238
239 return true
240 end
```

getAIToolReverserDirectionNode

Description

Returns reverser direction node of attached ai tool

Definition

getAIToolReverserDirectionNode(table vehicle)

Arguments

table vehicle vehicle to check

Return Values

integer aiToolReverserDirectionNode reverser direction node of ai tool

Code

```
270 function AIVehicleUtil.getAIToolReverserDirectionNode(vehicle)
271 for _, implement in pairs(vehicle:getAttachedImplements()) do
272 if implement.object ~= nil then
273 local reverserNode = implement.object:getAIToolReverserDirectionNode()
274
275 local attachedReverserNode = AIVehicleUtil.getAIToolReverserDirectionNode(implement.object)
276 reverserNode = reverserNode or attachedReverserNode
277
278 return reverserNode
279 end
280 end
281 end
```

getMaxToolRadius

Description

Returns max tool turn radius

Definition

getMaxToolRadius(table implement)

Arguments

table implement implement to check

Return Values

float maxTurnRadius max turn radius

Code

```
287 function AIVehicleUtil.getMaxToolRadius(implement)
288 local radius = 0
289
290 local _, rotationNode, wheels =
    implement.object:getAITurnRadiusLimitation()
291
292 -- collect the max manual defined turn radius of all vehicles,
    not only valid ai implements
293 local rootVehicle = implement.object:getRootVehicle()
294 local retRadius =
    AIVehicleUtil.getAttachedImplementsMaxTurnRadius(rootVehicle)
295
296 if retRadius ~= -1 then
297 radius = retRadius
298 end
299
300 if rotationNode then
301 local activeInputAttacherJoint =
    implement.object:getActiveInputAttacherJoint()
302 local refNode = rotationNode
303
304 -- If the refNode is any attacher joint, we always use the
    currently used attacher joint
305 for _, inputAttacherJoint in
    pairs(implement.object:getInputAttacherJoints()) do
306 if refNode == inputAttacherJoint.node then
307 refNode = activeInputAttacherJoint.node
308 break
309 end
310 end
311
312 local rx,_,rz = localToLocal(refNode,
    implement.object.components[1].node, 0,0,0)
313
314 for _, wheel in pairs(wheels) do
```

```
315 local nx,_,nz = localToLocal(wheel.repr,
    implement.object.components[1].node, 0,0,0)
316
317 local x,z = nx-rx, nz-rz
318 local cx,cz = 0,0
319
320 -- get max rotation
321 local rotMax
322 if refNode == activeInputAttacherJoint.node then
323 local attacherVehicle = implement.object:getAttacherVehicle()
324 local jointDesc =
    attacherVehicle:getAttacherJointDescFromObject(implement.object)
325 rotMax = math.max(jointDesc.upperRotLimit[2],
    jointDesc.lowerRotLimit[2]) *
    activeInputAttacherJoint.lowerRotLimitScale[2]
326 else
327 for _,compJoint in pairs(implement.object.componentJoints) do
328 if refNode == compJoint.jointNode then
329 rotMax = compJoint.rotLimit[2]
330 break
331 end
332 end
333 end
334
335 -- calc turning radius
336 local x1 = x*math.cos(rotMax) - z*math.sin(rotMax)
337 local z1 = x*math.sin(rotMax) + z*math.cos(rotMax)
338
339 local dx = -z1
340 local dz = x1
341 if wheel.steeringAxleScale ~= 0 and wheel.steeringAxleRotMax ~= 0
    then
342 local tmpx, tmpz = dx, dz
343 dx = tmpx*math.cos(wheel.steeringAxleRotMax) -
    tmpz*math.sin(wheel.steeringAxleRotMax)
344 dz = tmpx*math.sin(wheel.steeringAxleRotMax) +
    tmpz*math.cos(wheel.steeringAxleRotMax)
345 end
346
347 local hit,f1,_ = MathUtil.getLineLineIntersection2D(cx,cz, 1,0,
    x1,z1, dx,dz)
348 if hit then
```

```
349     radius = math.max(radius, math.abs(f1))
350 end
351 end
352 end
353
354 return radius
355 end
```

updateInvertLeftRightMarkers

Description

Update invertation of ai left and right markers on vehicle

Definition

updateInvertLeftRightMarkers(table rootAttacherVehicle, table vehicle)

Arguments

table	rootAttacherVehicle	root attacher vehicle
table	vehicle	vehicle

Code

```
361 function AIVehicleUtil.updateInvertLeftRightMarkers(rootAttacherVehicle, vehicle)
362     if vehicle.getAIMarkers ~= nil then
363         local leftMarker, rightMarker, _ = vehicle:getAIMarkers()
364         if leftMarker ~= nil and rightMarker ~= nil then
365             local lX, _, _ = localToLocal(leftMarker, rootAttacherVehicle:getAIVehicleDirectionNode(), 0,0,0)
366             local rX, _, _ = localToLocal(rightMarker, rootAttacherVehicle:getAIVehicleDirectionNode(), 0,0,0)
367
368             if rX > lX then
369                 vehicle:setAIMarkersInverted()
370             end
371         end
372     end
373 end
```

getValidityOfTurnDirections

Description

Checks fruits on left and right side of vehicle to decide the turn direction

Definition

getValidityOfTurnDirections(table vehicle, float checkFrontDistance, table turnData)

Arguments

table	vehicle	vehicle to check
float	checkFrontDistance	distance to check in front of vehicle
table	turnData	properties for turning

Return Values

float leftAreaPercentage left area percentage

float rightAreaPercentage right area percentage

Code

```
382 function AIVehicleUtil.getValidityOfTurnDirections(vehicle, turnData)
383 -- let's check the area at/around the marker which is farest behind of vehicle
384 local directionNode = vehicle:getAIVehicleDirectionNode()
385 local attachedAIImplements = vehicle:getAttachedAIImplements()
386 local checkFrontDistance = 5
387
388 local leftAreaPercentage = 0
389 local rightAreaPercentage = 0
390
391 local minZ = math.huge
392 local maxZ = -math.huge
393 for _,implement in pairs(attachedAIImplements) do
394 local leftMarker, rightMarker, backMarker = implement.object:getAIMarkers()
395
396 local _,_,zl = localToLocal(leftMarker, directionNode, 0,0,0)
397 local _,_,zr = localToLocal(rightMarker, directionNode, 0,0,0)
398 local _,_,zb = localToLocal(backMarker, directionNode, 0,0,0)
399
400 minZ = math.min(minZ, zl, zr, zb)
401 maxZ = math.max(maxZ, zl, zr, zb)
402 end
403
404 local sideDistance
405 if turnData == nil then
406 local minAreaWidth = math.huge
407 for _,implement in pairs(attachedAIImplements) do
408 local leftMarker, rightMarker, _ = implement.object:getAIMarkers()
409
410 local lx, _, _ = localToLocal(leftMarker, directionNode, 0,0,0)
411 local rx, _, _ = localToLocal(rightMarker, directionNode, 0,0,0)
412 minAreaWidth = math.min(minAreaWidth, math.abs(lx-rx))
413 end
414 sideDistance = minAreaWidth
415 else
```

```
416 sideDistance = math.abs(turnData.sideOffsetRight -
    turnData.sideOffsetLeft)
417 end
418
419 local dx, dz = vehicle.aiDriveDirection[1],
    vehicle.aiDriveDirection[2]
420 local sx, sz = -dz, dx
421
422 for _,implement in pairs(attachedAIImplements) do
423 local leftMarker, rightMarker, _ =
    implement.object:getAIMarkers()
424
425 local lx, ly, lz = localToLocal(leftMarker, directionNode, 0,0,0)
426 local rx, ry, rz = localToLocal(rightMarker, directionNode,
    0,0,0)
427
428 local width = math.abs(lx-rx)
429 local length = checkFrontDistance + (maxZ - minZ) +
    math.max(sideDistance*1.3 + 2, checkFrontDistance) -- 1.3~tan(53)
    allows detecting back along a field side with angle 53 (and 2m
    extra compensates for some variances, or higher angles with small
    tools)
430
431 lx, _, lz = localToWorld(directionNode, lx,ly,maxZ +
    checkFrontDistance)
432 rx, _, rz = localToWorld(directionNode, rx,ry,maxZ +
    checkFrontDistance)
433
434 local lSX = lx
435 local lSZ = lz
436 local lWX = lSX - sx * width
437 local lWZ = lSZ - sz * width
438 local lHX = lSX - dx * length
439 local lHZ = lSZ - dz * length
440
441 local rSX = rx
442 local rSZ = rz
443 local rWX = rSX + sx * width
444 local rWZ = rSZ + sz * width
445 local rHX = rSX - dx * length
446 local rHZ = rSZ - dz * length
447
```

```
448 local lArea, lTotal = AIVehicleUtil.getAIAreaOfVehicle(implement.object, lSX,lSZ, lWX,lWZ, lHX,lHZ, false)
449 local rArea, rTotal = AIVehicleUtil.getAIAreaOfVehicle(implement.object, rSX,rSZ, rWX,rWZ, rHX,rHZ, false)
450
451 if lTotal > 0 then
452 leftAreaPercentage = leftAreaPercentage + (lArea / lTotal)
453 end
454 if rTotal > 0 then
455 rightAreaPercentage = rightAreaPercentage + (rArea / rTotal)
456 end
457
458 -- just visual debuging
459 if VehicleDebug.state == VehicleDebug.DEBUG_AI then
460 local lSY = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, lSX,0,lSZ)+2
461 local lWY = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, lWX,0,lWZ)+2
462 local lHY = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, lHX,0,lHZ)+2
463 local rSY = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, rSX,0,rSZ)+2
464 local rWY = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, rWX,0,rWZ)+2
465 local rHY = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, rHX,0,rHZ)+2
466
467 vehicle:addAIDebugLine({lSX,lSY,lSZ}, {lWX,lWY,lWZ}, {0.5, 0.5, 0.5})
468 vehicle:addAIDebugLine({lSX,lSY,lSZ}, {lHX,lHY,lHZ}, {0.5, 0.5, 0.5})
469 vehicle:addAIDebugLine({rSX,rSY,rSZ}, {rWX,rWY,rWZ}, {0.5, 0.5, 0.5})
470 vehicle:addAIDebugLine({rSX,rSY,rSZ}, {rHX,rHY,rHZ}, {0.5, 0.5, 0.5})
471 end
472 end
```

```
473
474 leftAreaPercentage = leftAreaPercentage /
    table.getn(attachedAIImplements)
475 rightAreaPercentage = rightAreaPercentage /
    table.getn(attachedAIImplements)
476
477 return leftAreaPercentage, rightAreaPercentage
478 end
```

checkImplementListForValidGround

Description

Returns if valid ground to work on is found for ai vehicle

Definition

checkImplementListForValidGround(table vehicle, float lookAheadDist, float lookAheadSize)

Arguments

table	vehicle	vehicle to check
float	lookAheadDist	look a head distance
float	lookAheadSize	look a head size

getAIAreaOfVehicle

Description

Returns amount of fruit to work for ai vehicle is in given area

Definition

getAIAreaOfVehicle(table vehicle, float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ)

Arguments

table	vehicle	vehicle
float	startWorldX	start world x
float	startWorldZ	start world z
float	widthWorldX	width world x
float	widthWorldZ	width world z
float	heightWorldX	height world x
float	heightWorldZ	height world z

Return Values

float area	area found
float totalArea	total area checked

Code

```
528 function AIVehicleUtil.getAIAreaOfVehicle(vehicle, startWorldX,
    startWorldZ, widthWorldX, widthWorldZ, heightWorldX,
    heightWorldZ, ignoreProhibitedValues)
529 local terrainDetailRequiredValueRanges =
    vehicle:getAITerrainDetailRequiredRange()
530 local terrainDetailProhibitValueRanges =
    vehicle:getAITerrainDetailProhibitedRange()
531
```

```
532 local fruitRequirements = vehicle:getAIFruitRequirements()
533 local useDensityHeightMap, useWindrowFruitType =
    vehicle:getAIFruitExtraRequirements()
534
535 local fruitProhibitions = vehicle:getAIFruitProhibitions()
536
537 if ignoreProhibitedValues then
538 terrainDetailProhibitValueRanges = {}
539 fruitProhibitions = {}
540 end
541
542 if not useDensityHeightMap then
543 return AIVehicleUtil.getAIFruitArea(startWorldX, startWorldZ,
    widthWorldX, widthWorldZ, heightWorldX, heightWorldZ,
    terrainDetailRequiredValueRanges,
    terrainDetailProhibitValueRanges, fruitRequirements,
    fruitProhibitions, useWindrowFruitType)
544 else
545 return AIVehicleUtil.getAIDensityHeightArea(startWorldX,
    startWorldZ, widthWorldX, widthWorldZ, heightWorldX,
    heightWorldZ, terrainDetailRequiredValueRanges,
    terrainDetailProhibitValueRanges, fruitRequirements,
    fruitProhibitions, useWindrowFruitType)
546 end
547 end
```

getAIFruitArea

Description

Returns amount of fruit to work is in given area

Definition

getAIFruitArea(float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ, table terrainDetailRequiredValueRanges, table terrainDetailProhibitValueRanges, table fruitRequirements, table fruitProhibitions, boolean useWindrowed)

Arguments

float	startWorldX	start world x
float	startWorldZ	start world z
float	widthWorldX	width world x
float	widthWorldZ	width world z
float	heightWorldX	height world x
float	heightWorldZ	height world z
table	terrainDetailRequiredValueRanges	terrain detail required value ranges
table	terrainDetailProhibitValueRanges	terrain detail prohibit value ranges
table	fruitRequirements	required fruit type
table	fruitProhibitions	prohibited fruit type

boolean useWindrowed use windrow

Return Values

float area area found

float totalArea total area checked

Code

```
564 function AIVehicleUtil.getAIFruitArea(startWorldX, startWorldZ,
    widthWorldX, widthWorldZ, heightWorldX, heightWorldZ,
    terrainDetailRequiredValueRanges,
    terrainDetailProhibitValueRanges, fruitRequirements,
    fruitProhibitions, useWindrowed)
565 local query = g_currentMission.fieldCropsQuery
566
567 for _, fruitRequirement in pairs(fruitRequirements) do
568 if fruitRequirement.fruitType ~= FruitType.UNKNOWN then
569 local ids = g_currentMission.fruits[fruitRequirement.fruitType]
570 if ids ~= nil and ids.id ~= 0 then
571 if useWindrowed then
572 return 0, 1
573 end
574
575 local desc =
    g_fruitTypeManager:getFruitTypeByIndex(fruitRequirement.fruitType)
576 query:addRequiredCropType(ids.id,
    fruitRequirement.minGrowthState+1,
    fruitRequirement.maxGrowthState+1, desc.startStateChannel,
    desc.numStateChannels,
    g_currentMission.terrainDetailTypeFirstChannel,
    g_currentMission.terrainDetailTypeNumChannels)
577 end
578 end
579 end
580
581 for _, fruitProhibition in pairs(fruitProhibitions) do
582 if fruitProhibition.fruitType ~= FruitType.UNKNOWN then
583 local ids = g_currentMission.fruits[fruitProhibition.fruitType]
584 if ids ~= nil and ids.id ~= 0 then
585 local desc =
    g_fruitTypeManager:getFruitTypeByIndex(fruitProhibition.fruitType)
586 query:addProhibitedCropType(ids.id,
    fruitProhibition.minGrowthState+1,
    fruitProhibition.maxGrowthState+1, desc.startStateChannel,
    desc.numStateChannels,
    g_currentMission.terrainDetailTypeFirstChannel,
    g_currentMission.terrainDetailTypeNumChannels)
587 end
```

```
588 end
589 end
590
591 for _,valueRange in pairs(terrainDetailRequiredValueRanges) do
592 query:addRequiredGroundValue(valueRange[1], valueRange[2],
    valueRange[3], valueRange[4])
593 end
594
595 for _,valueRange in pairs(terrainDetailProhibitValueRanges) do
596 query:addProhibitedGroundValue(valueRange[1], valueRange[2],
    valueRange[3], valueRange[4])
597 end
598
599 local x,z, widthX,widthZ, heightX,heightZ =
    MathUtil.getXZWidthAndHeight(startWorldX, startWorldZ,
    widthWorldX, widthWorldZ, heightWorldX, heightWorldZ)
600 return query:getParallelogram(x,z, widthX,widthZ, heightX,heightZ,
    true)
601 end
```

getAIDensityHeightArea

Description

Returns amount of density height to work is in given area

Definition

getAIDensityHeightArea(float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ, table terrainDetailRequiredValueRanges, table terrainDetailProhibitValueRanges, integer requiredfruittype, integer requiredMinGrowthState, integer requiredMaxGrowthState, integer prohibitedFruitType, integer prohibitedMinGrowthState, integer prohibitedMaxGrowthState, boolean useWindrowed)

Arguments

float	startWorldX	start world x
float	startWorldZ	start world z
float	widthWorldX	width world x
float	widthWorldZ	width world z
float	heightWorldX	height world x
float	heightWorldZ	height world z
table	terrainDetailRequiredValueRanges	terrain detail required value ranges
table	terrainDetailProhibitValueRanges	terrain detail prohibit value ranges
integer	requiredfruittype	required fruit type
integer	requiredMinGrowthState	required min growth state
integer	requiredMaxGrowthState	required max growth state
integer	prohibitedFruitType	prohibited fruit type
integer	prohibitedMinGrowthState	prohibited min growth state
integer	prohibitedMaxGrowthState	prohibited max growth state

boolean useWindrowed use windrow

Return Values

float area area found

float totalArea total area checked

Code

```
622 function AIVehicleUtil.getAIDensityHeightArea(startWorldX, startWorldZ,
    widthWorldZ, heightWorldX, heightWorldZ, terrainDetailRequiredValueRange
    terrainDetailProhibitValueRanges, fruitRequirements, fruitProhibitions,
623 -- first check if we are on a field
624
625 local data = g_currentMission.densityMapModifiers.getAIDensityHeightArea
626 local modifier = data.modifier
627 local filter = data.filter
628
629 modifier:setParallelogramWorldCoords(startWorldX, startWorldZ, widthWorl
    widthWorldZ, heightWorldX, heightWorldZ, "ppp")
630 filter:setValueCompareParams("greater", 0)
631
632 local _, detailArea, _ = modifier:executeGet(filter)
633 if detailArea == 0 then
634 return 0, 0
635 end
636
637 local retArea, retTotalArea = 0, 0
638 for _, fruitRequirement in pairs(fruitRequirements) do
639 if fruitRequirement.fruitType ~= FruitType.UNKNOWN then
640 local fillType
641 if useWindrowed then
642 fillType =
    g_fruitTypeManager:getWindrowFillTypeIndexByFruitTypeIndex(fruitRequiren
643 else
644 fillType =
    g_fruitTypeManager:getFruitTypeIndexByFillTypeIndex(fruitRequirement.fru
645 end
646 local _, area, totalArea = DensityMapHeightUtil.getFillLevelAtArea(fillT
    startWorldX,startWorldZ, widthWorldX,widthWorldZ, heightWorldX,heightWor
647 retArea, retTotalArea = retArea+area, totalArea
648 end
649 end
650
651 return retArea, retTotalArea
652 end
```

ConfigurationManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
17  function ConfigurationManager:new(customMt)
18  local self = AbstractManager:new(customMt or
    ConfigurationManager_mt)
19
20  self:initDataStructures()
21
22  return self
23  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
27  function ConfigurationManager:initDataStructures()
28  self.configurations = {}
29  self.intToConfigurationName = {}
30  self.configurationNameToInt = {}
31  end
```

addConfigurationType

Description

Register new configuration type

Definition

addConfigurationType(string name, string title, function preLoadFunc, function singleItemLoadFunc, function postLoadFunc, integer selectorType)

Arguments

string	name	name of config
string	title	title displayed in shop
function	preLoadFunc	called before loading of configuration
function	singleItemLoadFunc	called on single item load
function	postLoadFunc	called after loading of configuration
integer	selectorType	selector type [ConfigurationManager.SELECTOR_MULTIOPTION ConfigurationManager.SELECTOR_COLOR]

getNumOfConfigurationTypes

Description

Returns number of configuration types

Definition

getNumOfConfigurationTypes()

Return Values

integer numOfConfigurationTypes number of configuration types

Code

```
73  function ConfigurationManager:getNumOfConfigurationTypes()
74  return table.getn(self.intToConfigurationName)
75  end
```

getConfigurationTypes

Description

Returns a table of the available configuration types

Definition

getConfigurationTypes()

Return Values

integer numOfConfigurationTypes number of configuration types

Code

```
80  function ConfigurationManager:getConfigurationTypes()
81  return self.intToConfigurationName
82  end
```

getConfigurationNameByIndex

Description

Returns configuration name by given index

Definition

getConfigurationNameByIndex(integer index)

Arguments

integer index index of config

Return Values

string name name of config

Code

```
88  function ConfigurationManager:getConfigurationNameByIndex(index)
89  return self.intToConfigurationName[index]
90  end
```

getConfigurationIndexByName

Description

Returns configuration index by given name

Definition

getConfigurationIndexByName(string name)

Arguments

string name name of config

Return Values

integer index index of config

Code

```
96  function ConfigurationManager:getConfigurationIndexByName(name)
97  return self.configurationNameToInt[name]
98  end
```

getConfigurationDescByName

Description

Returns configuration desc by name

Definition

getConfigurationDescByName(string name)

Arguments

string name name of config

Return Values

table configuration configuration

Code

```
104  function ConfigurationManager:getConfigurationDescByName(name)
105  return self.configurations[name]
106  end
```

getConfigurationAttribute

Description

Returns configuration attribute by given name and attribute

Definition

getConfigurationAttribute(string configurationName, string attribute)

Arguments

string configurationName name of config

string attribute name of attribute

Return Values

any_type value value of attribute

Code

```
113  function ConfigurationManager:getConfigurationAttribute(configurationName, attribute)
114  local config = self:getConfigurationDescByName(configurationName)
115  return config[attribute]
116  end
```

SpecializationManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
17  function SpecializationManager:new(customMt)
18  local self = AbstractManager:new(customMt or
    SpecializationManager_mt)
19
20  return self
21  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
25  function SpecializationManager:initDataStructures()
26  self.specializations = {}
27  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
32  function SpecializationManager:loadMapData()
33  SpecializationManager:superClass().loadMapData(self)
34
35  local xmlFile = loadXMLFile("SpecializationsXML",
    "dataS/specializations.xml")
36  local i=0
37  while true do
38  local baseName =
    string.format("specializations.specialization(%d)", i)
39
40  local typeName = getXMLString(xmlFile, baseName.. "#name")
41  if typeName == nil then
42  break
43  end
44  local className = getXMLString(xmlFile, baseName.. "#className")
45  local filename = getXMLString(xmlFile, baseName.. "#filename")
46
```

```
47  g_deferredLoadingManager:addSubtask(function()
48  self:addSpecialization(typeName, className, filename, "")
49  end)
50  i = i+1
51  end
52  delete(xmlFile)
53
54  g_deferredLoadingManager:addSubtask(function()
55  print(" Loaded specializations")
56  end)
57
58  return true
59  end
```

addSpecialization

Description

Adds a new vehicleType

Definition

addSpecialization(string name, string className, string filename, string customEnvironment)

Arguments

string name	specialization name
string className	classname
string filename	filename
string customEnvironment	a custom environment

Return Values

boolean success true if added else false

Vehicle

Description

registerInteractionFlag

Description

Register interaction flag

Definition

registerInteractionFlag(string name)

Arguments

string name name of flag

Code

```
47  function Vehicle.registerInteractionFlag(name)
48  local key = "INTERACTION_FLAG_"..string.upper(name)
49  if Vehicle[key] == nil then
50  Vehicle.NUM_INTERACTION_FLAGS = Vehicle.NUM_INTERACTION_FLAGS + 1
51  Vehicle[key] = Vehicle.NUM_INTERACTION_FLAGS
52  end
53  end
```

saveStatsToXMLFile

Description

Get xml states attributes

Definition

saveStatsToXMLFile()

Return Values

string attributes attributes

Code

```
862 function Vehicle:saveStatsToXMLFile(xmlFile, key)
863 local isTabbable = self.getIsTabbable == nil or self:getIsTabbable()
864 if self.isDeleted or not self.isVehicleSaved or not isTabbable then
865 return false
866 end
867 local name = "Unknown"
868 local categoryName = "unknown"
869 local storeItem = g_storeManager:getItemByXMLFilename(self.configFileName)
870 if storeItem ~= nil then
871 if storeItem.name ~= nil then
872 name = tostring(storeItem.name)
873 end
874 if storeItem.categoryName ~= nil and storeItem.categoryName ~= "" then
875 categoryName = tostring(storeItem.categoryName)
876 end
877 end
878
879 setXMLString(xmlFile, key.."#name", HTMLUtil.encodeToHTML(name))
880 setXMLString(xmlFile, key.."#category", HTMLUtil.encodeToHTML(categoryName))
881 setXMLString(xmlFile, key.."#type", HTMLUtil.encodeToHTML(tostring(self.typeName)))
882
883 if self.components[1] ~= nil and self.components[1].node ~= 0 then
884 local x,y,z = getWorldTranslation(self.components[1].node)
885 setXMLFloat(xmlFile, key.."#x", x)
886 setXMLFloat(xmlFile, key.."#y", y)
887 setXMLFloat(xmlFile, key.."#z", z)
888 end
```

```
889
890 for id, spec in pairs(self.specializations) do
891 local name = self.specializationNames[id]
892
893 if spec.saveStatsToXMLFile ~= nil then
894 spec.saveStatsToXMLFile(self, xmlFile, key)
895 end
896 end
897
898 return true
899 end
```

readStream

Description

Called on client side on join

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId	stream ID
table	connection connection

Code

```
905 function Vehicle:readStream(streamId, connection)
906 Vehicle:superClass().readStream(self, streamId)
907 local configFile =
    NetworkUtil.convertFromNetworkFilename(streamReadString(streamId))
908 local typeName = streamReadString(streamId)
909
910 local configurations = {}
911 local numConfigs = streamReadUIntN(streamId,
    ConfigurationUtil.SEND_NUM_BITS)
912 for i=1, numConfigs do
913 local configNameId = streamReadUIntN(streamId,
    ConfigurationUtil.SEND_NUM_BITS)
914 local configId = streamReadUInt16(streamId)
915
916 local configName =
    g_configurationManager:getConfigurationNameByIndex(configNameId+1)
917 if configName ~= nil then
918 configurations[configName] = configId+1
919 end
920 end
921
922 local boughtConfigurations = {}
```

```
923  local numConfigs = streamReadUIntN(streamId, ConfigurationUtil.SEND_NUM_BITS)
924  for i=1, numConfigs do
925  local configNameId = streamReadUIntN(streamId, ConfigurationUtil.SEND_NUM_BITS)
926  local configName = g_configurationManager:getConfigurationNameByIndex(configNameId+1)
927  boughtConfigurations[configName] = {}
928  local numBoughtConfigIds = streamReadUInt16(streamId)
929  for j=1, numBoughtConfigIds do
930  local boughtConfigId = streamReadUInt16(streamId)
931  boughtConfigurations[configName][boughtConfigId + 1] = true
932  end
933  end
934
935  if self.configFileName == nil then
936  local vehicleData = {}
937  vehicleData.filename = configFile
938  vehicleData.isAbsolute = false
939  vehicleData.typeName = typeName
940  vehicleData.posX = 0
941  vehicleData.posY = nil
942  vehicleData.posZ = 0
943  vehicleData.yOffset = 0
944  vehicleData.rotX = 0
945  vehicleData.rotY = 0
946  vehicleData.rotZ = 0
947  vehicleData.isVehicleSaved = true
948  vehicleData.price = 0
949  vehicleData.propertyState = Vehicle.PROPERTY_STATE_NONE
950  -- assign parent class Object's ownerFarmId field here to ensure ownership synchronization in MP:
951  vehicleData.ownerFarmId = self.ownerFarmId
952  vehicleData.isLeased = 0
953  vehicleData.configurations = configurations
954  vehicleData.boughtConfigurations = boughtConfigurations
955  self:load(vehicleData)
956  end
957
958  -- remove from physics to set static components correctly
959  self:removeFromPhysics()
```

```
960
961 local paramsXZ = self.highPrecisionPositionSynchronization and
    g_currentMission.vehicleXZPosHighPrecisionCompressionParams or
    g_currentMission.vehicleXZPosCompressionParams
962 local paramsY = self.highPrecisionPositionSynchronization and
    g_currentMission.vehicleYPosHighPrecisionCompressionParams or
    g_currentMission.vehicleYPosCompressionParams
963 for i=1, table.getn(self.components) do
964 local component = self.components[i]
965 local x = NetworkUtil.readCompressedWorldPosition(streamId,
    paramsXZ)
966 local y = NetworkUtil.readCompressedWorldPosition(streamId, paramsY)
967 local z = NetworkUtil.readCompressedWorldPosition(streamId,
    paramsXZ)
968 local x_rot = NetworkUtil.readCompressedAngle(streamId)
969 local y_rot = NetworkUtil.readCompressedAngle(streamId)
970 local z_rot = NetworkUtil.readCompressedAngle(streamId)
971
972 local qx,qy,qz,qw = mathEulerToQuaternion(x_rot,y_rot,z_rot)
973 self:setWorldPositionQuaternion(x,y,z, qx,qy,qz,qw, i, true)
974
975 component.networkInterpolators.position:setPosition(x,y,z)
976 component.networkInterpolators.quaternion:setQuaternion(qx,qy,qz,qw)
977 end
978 self.networkTimeInterpolator:reset()
979
980 -- add to physics again
981 self:addToPhysics()
982
983 self.serverMass = streamReadFloat32(streamId)
984 self.age = streamReadUInt16(streamId)
985 self:setOperatingTime(streamReadFloat32(streamId), true)
986 self.price = streamReadInt32(streamId)
987 self.propertyState = streamReadUIntN(streamId, 2)
988
989 SpecializationUtil.raiseEvent(self, "onReadStream", streamId,
    connection)
990 end
```

writeStream

Description

Called on server side on join

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId stream ID

table connection connection

Code

```
996  function Vehicle:writeStream(streamId, connection)
997  Vehicle:superClass().writeStream(self, streamId)
998  streamWriteString(streamId,
     NetworkUtil.convertToNetworkFilename(self.configFileName))
999  streamWriteString(streamId, self.typeName)
1000
1001 local numConfigs = 0
1002 for _,_ in pairs(self.configurations) do
1003 numConfigs = numConfigs + 1
1004 end
1005
1006 streamWriteUIntN(streamId, numConfigs,
     ConfigurationUtil.SEND_NUM_BITS)
1007 for configName, configId in pairs(self.configurations) do
1008 local configNameId =
     g_configurationManager:getConfigurationIndexByName(configName)
1009 streamWriteUIntN(streamId, configNameId-1,
     ConfigurationUtil.SEND_NUM_BITS)
1010 streamWriteUInt16(streamId, configId-1)
1011 end
1012
1013 local numBoughtConfigs = 0
1014 for _,_ in pairs(self.boughtConfigurations) do
1015 numBoughtConfigs = numBoughtConfigs + 1
1016 end
1017
1018 streamWriteUIntN(streamId, numBoughtConfigs,
     ConfigurationUtil.SEND_NUM_BITS)
1019 for configName, configIds in pairs(self.boughtConfigurations) do
1020 local numBoughtConfigIds = 0
1021 for _,_ in pairs(configIds) do
1022 numBoughtConfigIds = numBoughtConfigIds + 1
1023 end
1024 local configNameId =
     g_configurationManager:getConfigurationIndexByName(configName)
1025 streamWriteUIntN(streamId, configNameId-1,
     ConfigurationUtil.SEND_NUM_BITS)
```

```
1026  streamWriteUInt16(streamId, numBoughtConfigIds)
1027  for id, _ in pairs(configIds) do
1028  streamWriteUInt16(streamId, id-1)
1029  end
1030  end
1031
1032  local paramsXZ = self.highPrecisionPositionSynchronization and
      g_currentMission.vehicleXZPosHighPrecisionCompressionParams or
      g_currentMission.vehicleXZPosCompressionParams
1033  local paramsY = self.highPrecisionPositionSynchronization and
      g_currentMission.vehicleYPosHighPrecisionCompressionParams or
      g_currentMission.vehicleYPosCompressionParams
1034  for i=1, table.getn(self.components) do
1035  local component = self.components[i]
1036  local x,y,z = getWorldTranslation(component.node)
1037  local x_rot,y_rot,z_rot = getWorldRotation(component.node)
1038  NetworkUtil.writeCompressedWorldPosition(streamId, x, paramsXZ)
1039  NetworkUtil.writeCompressedWorldPosition(streamId, y, paramsY)
1040  NetworkUtil.writeCompressedWorldPosition(streamId, z, paramsXZ)
1041  NetworkUtil.writeCompressedAngle(streamId, x_rot)
1042  NetworkUtil.writeCompressedAngle(streamId, y_rot)
1043  NetworkUtil.writeCompressedAngle(streamId, z_rot)
1044  end
1045
1046  streamWriteFloat32(streamId, self.serverMass)
1047  streamWriteUInt16(streamId, self.age)
1048  streamWriteFloat32(streamId, self.operatingTime)
1049  streamWriteInt32(streamId, self.price)
1050  streamWriteUIntN(streamId, self.propertyState, 2)
1051
1052  SpecializationUtil.raiseEvent(self, "onWriteStream", streamId,
      connection)
1053  end
```

readUpdateStream

Description

Called on client side on update

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer	streamId	stream ID
integer	timestamp	timestamp
table	connection	connection

Code

```
1060  function Vehicle:readUpdateStream(streamId, timestamp, connection)
1061  if connection.isServer then
1062  local hasUpdate = streamReadBool(streamId)
1063  if hasUpdate then
1064  self.networkTimeInterpolator:startNewPhaseNetwork()
1065
1066  local paramsXZ = self.highPrecisionPositionSynchronization and
      g_currentMission.vehicleXZPosHighPrecisionCompressionParams or
      g_currentMission.vehicleXZPosCompressionParams
1067  local paramsY = self.highPrecisionPositionSynchronization and
      g_currentMission.vehicleYPosHighPrecisionCompressionParams or
      g_currentMission.vehicleYPosCompressionParams
1068  for i=1, table.getn(self.components) do
1069  local component = self.components[i]
1070  if not component.isStatic then
1071  local x = NetworkUtil.readCompressedWorldPosition(streamId, paramsXZ)
1072  local y = NetworkUtil.readCompressedWorldPosition(streamId, paramsY)
1073  local z = NetworkUtil.readCompressedWorldPosition(streamId, paramsXZ)
1074  local x_rot = NetworkUtil.readCompressedAngle(streamId)
1075  local y_rot = NetworkUtil.readCompressedAngle(streamId)
1076  local z_rot = NetworkUtil.readCompressedAngle(streamId)
1077  local qx,qy,qz,qw = mathEulerToQuaternion(x_rot,y_rot,z_rot)
1078
1079  component.networkInterpolators.position:setTargetPosition(x,y,z)
1080  component.networkInterpolators.quaternion:setTargetQuaternion(qx,qy,qz,
1081  end
1082  end
1083  SpecializationUtil.raiseEvent(self, "onReadPositionUpdateStream",
      streamId, connection)
1084  end
1085  end
1086
1087  if Vehicle.debugNetworkUpdate then
1088  print("--------------------------------------------------------------")
1089  print(self.configFileName)
1090  for _, spec in ipairs(self.eventListeners["readUpdateStream"]) do
1091  local className = ClassUtil.getClassName(spec)
1092  local startBits = streamGetReadOffset(streamId)
1093  spec["readUpdateStream"](self, streamId, timestamp, connection)
1094  print(" "..tostring(className).." read " .. streamGetReadOffset(streamI
      startBits .. " bits")
```

```
1095  end
1096  else
1097  SpecializationUtil.raiseEvent(self, "onReadUpdateStream", streamId,
      timestamp, connection)
1098  end
1099  end
```

writeUpdateStream

Description

Called on server side on update

Definition

writeUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Code

```
1106  function Vehicle:writeUpdateStream(streamId, connection,
      dirtyMask)
1107  if not connection.isServer then
1108  if streamWriteBool(streamId, bitAND(dirtyMask,
      self.vehicleDirtyFlag) ~= 0) then
1109
1110  local paramsXZ = self.highPrecisionPositionSynchronization and
      g_currentMission.vehicleXZPosHighPrecisionCompressionParams or
      g_currentMission.vehicleXZPosCompressionParams
1111  local paramsY = self.highPrecisionPositionSynchronization and
      g_currentMission.vehicleYPosHighPrecisionCompressionParams or
      g_currentMission.vehicleYPosCompressionParams
1112  for i=1, table.getn(self.components) do
1113  local component = self.components[i]
1114  if not component.isStatic then
1115  local x,y,z = getWorldTranslation(component.node)
1116  local x_rot,y_rot,z_rot = getWorldRotation(component.node)
1117  NetworkUtil.writeCompressedWorldPosition(streamId, x, paramsXZ)
1118  NetworkUtil.writeCompressedWorldPosition(streamId, y, paramsY)
1119  NetworkUtil.writeCompressedWorldPosition(streamId, z, paramsXZ)
1120  NetworkUtil.writeCompressedAngle(streamId, x_rot)
1121  NetworkUtil.writeCompressedAngle(streamId, y_rot)
1122  NetworkUtil.writeCompressedAngle(streamId, z_rot)
1123  end
1124  end
1125  SpecializationUtil.raiseEvent(self,
      "onWritePositionUpdateStream", streamId, connection, dirtyMask)
```

```
1126  end
1127  end
1128
1129  if Vehicle.debugNetworkUpdate then
1130  print("------------------------------------------------------------")
1131  print(self.configFileName)
1132  for _, spec in ipairs(self.eventListeners["writeUpdateStream"]) do
1133  local className = ClassUtil.getClassName(spec)
1134  local startBits = streamGetWriteOffset(streamId)
1135  spec["writeUpdateStream"](self, streamId, connection, dirtyMask)
1136  print(" "..tostring(className).." Wrote " .. streamGetWriteOffset(streamId)-startBits .. " bits")
1137  end
1138  else
1139  SpecializationUtil.raiseEvent(self, "onWriteUpdateStream", streamId, connection, dirtyMask)
1140  end
1141  end
```

updateTick

Description

updateTick

Definition

updateTick(float dt)

Arguments

float dt time since last call in ms

Code

```
1281  function Vehicle:updateTick(dt)
1282
1283  local isActive = self:getIsActive()
1284  local isActiveForInput = self:getIsActiveForInput()
1285  local isSelected = self:getIsSelected()
1286
1287  self.wasTooFast = false
1288  if self.isServer then
1289  if self.synchronizePosition then
1290  local hasOwner = self:getOwner() ~= nil
1291  for i=1, table.getn(self.components) do
1292  local component = self.components[i]
1293  if not component.isStatic then
1294  local x,y,z = getWorldTranslation(component.node)
```

```
1295  local x_rot,y_rot,z_rot=getWorldRotation(component.node)
1296  local sentTranslation = component.sentTranslation
1297  local sentRotation = component.sentRotation
1298  if hasOwner or
1299  math.abs(x-sentTranslation[1]) > 0.005 or
1300  math.abs(y-sentTranslation[2]) > 0.005 or
1301  math.abs(z-sentTranslation[3]) > 0.005 or
1302  math.abs(x_rot-sentRotation[1]) > 0.1 or
1303  math.abs(y_rot-sentRotation[2]) > 0.1 or
1304  math.abs(z_rot-sentRotation[3]) > 0.1
1305  then
1306  self:raiseDirtyFlags(self.vehicleDirtyFlag)
1307  sentTranslation[1] = x
1308  sentTranslation[2] = y
1309  sentTranslation[3] = z
1310  sentRotation[1] = x_rot
1311  sentRotation[2] = y_rot
1312  sentRotation[3] = z_rot
1313
1314  self.lastMoveTime = g_currentMission.time
1315  end
1316  end
1317  end
1318  end
1319
1320  -- is the vehicle sunken in the water?
1321  self.showTailwaterDepthWarning = false
1322  if not self.isBroken and not g_gui:getIsGuiVisible() then
1323  local tailwaterDepth = self:getTailwaterDepth()
1324  if tailwaterDepth > self.thresholdTailwaterDepthWarning then
1325  self.showTailwaterDepthWarning = true
1326  if tailwaterDepth > self.thresholdTailwaterDepth then
1327  self:setBroken()
1328  end
1329  end
1330  end
1331
1332  local rootAttacherVehicle = self:getRootVehicle()
1333  if rootAttacherVehicle ~= nil and rootAttacherVehicle ~= self
      then
```

```
1334  rootAttacherVehicle.showTailwaterDepthWarning = rootAttacherVehicle.showTailwaterDepthWarning or self.showTailwaterDepthWarning
1335  end
1336  end
1337
1338  if self:getIsOperating() then
1339  self:setOperatingTime(self.operatingTime + dt)
1340  end
1341
1342  SpecializationUtil.raiseEvent(self, "onUpdateTick", dt, isActiveForInput, isSelected)
1343  SpecializationUtil.raiseEvent(self, "onPostUpdateTick", dt, isActiveForInput, isSelected)
1344  end
```

drawUIInfo

Description

Draw UI info

Definition

drawUIInfo()

Code

```
1371  function Vehicle:drawUIInfo()
1372  if g_showVehicleDistance then
1373  local dist = calcDistanceFrom(self.rootNode, getCamera())
1374  if dist <= 350 then
1375  Utils.renderTextAtWorldPosition(x,y+1,z, string.format("%.0f", dist), getCorrectTextSize(0.02), 0)
1376  end
1377  end
1378  end
```

addNodeObjectMapping

Description

Add component nodes to list

Definition

addNodeObjectMapping(table list)

Arguments

table list list

Code

```
1396  function Vehicle:addNodeObjectMapping(list)
1397  for _,v in pairs(self.components) do
1398  list[v.node] = self
1399  end
```

```
1400  end
```

removeNodeObjectMapping

Description

Remove component nodes from list

Definition

removeNodeObjectMapping(table list)

Arguments

table list list

Code

```
1405  function Vehicle:removeNodeObjectMapping(list)
1406  for _,v in pairs(self.components) do
1407  list[v.node] = nil
1408  end
1409  end
```

addToPhysics

Description

Add vehicle to physics

Definition

addToPhysics()

Return Values

boolean success success

Code

```
1414  function Vehicle:addToPhysics()
1415
1416  if not self.isAddedToPhysics then
1417  local lastMotorizedNode = nil
1418  for _, component in pairs(self.components) do
1419  addToPhysics(component.node)
1420  if component.motorized then
1421  if lastMotorizedNode ~= nil then
1422  if self.isServer then
1423  addVehicleLink(lastMotorizedNode, component.node)
1424  end
1425  end
1426  lastMotorizedNode = component.node
1427  end
1428  end
1429
1430  self.isAddedToPhysics = true
1431
1432  if self.isServer then
```

```
1433  for _, jointDesc in pairs(self.componentJoints) do
1434  self:createComponentJoint(self.components[jointDesc.componentIndices[1]],
      self.components[jointDesc.componentIndices[2]], jointDesc)
1435  end
1436
1437  -- if rootnode is sleeping all other components are sleeping as well
1438  addWakeUpReport(self.rootNode, "onVehicleWakeUpCallback", self)
1439  end
1440
1441  for _, collisionPair in pairs(self.collisionPairs) do
1442  setPairCollision(collisionPair.component1.node,
      collisionPair.component2.node, collisionPair.enabled)
1443  end
1444
1445  self:setMassDirty()
1446  end
1447
1448  return true
1449  end
```

removeFromPhysics

Description

Remove vehicle from physics

Definition

removeFromPhysics()

Code

```
1453  function Vehicle:removeFromPhysics()
1454  for _, component in pairs(self.components) do
1455  removeFromPhysics(component.node)
1456  end
1457  -- invalidate wheel shapes and component joints (removing the
      components removes the wheels and joints too)
1458  if self.isServer then
1459  for _, jointDesc in pairs(self.componentJoints) do
1460  jointDesc.jointIndex = 0
1461  end
1462  removeWakeUpReport(self.rootNode)
1463  end
1464  self.isAddedToPhysics = false
1465
1466  return true
1467  end
```

setRelativePosition

Description

Set relative position of vehicle

Definition

setRelativePosition(float positionX, float offsetY, float positionZ, float yRot)

Arguments

float	positionX	x position
float	offsetY	y offset
float	positionZ	z position
float	yRot	y rotation

Code

```
1475  function Vehicle:setRelativePosition(positionX, offsetY, positionZ, yRot)
1476  -- position the vehicle
1477  local terrainHeight = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, positionX, 300, positionZ)
1478
1479  self:setAbsolutePosition(positionX, terrainHeight+offsetY, positionZ, 0, yRot, 0)
1480  end
```

setWorldPosition

Description

Set world position and rotation of component

Definition

setWorldPosition(float x, float y, float z, float xRot, float yRot, float zRot, integer i, boolean changeInterp)

Arguments

float	x	x position
float	y	y position
float	z	z position
float	xRot	x rotation
float	yRot	y rotation
float	zRot	z rotation
integer	i	index if component
boolean	changeInterp	change interpolation

Code

```
1524  function Vehicle:setWorldPosition(x,y,z, xRot,yRot,zRot, i, changeInterp)
1525  local component = self.components[i]
1526  setWorldTranslation(component.node, x,y,z)
1527  setWorldRotation(component.node, xRot,yRot,zRot)
1528  if changeInterp then
1529  local qx, qy, qz, qw = mathEulerToQuaternion(xRot,yRot,zRot)
```

```
1530  component.networkInterpolators.quaternion:setQuaternion(qx, qy, qz, qw)
1531  component.networkInterpolators.position:setPosition(x,y,z)
1532  end
1533  end
```

setWorldPositionQuaternion

Description

Set world position and quaternion rotation of component

Definition

setWorldPositionQuaternion(float x, float y, float z, float qx, float qy, float qz, float qw, integer i, boolean changeInterp)

Arguments

float	x	x position
float	y	y position
float	z	z position
float	qx	x rotation
float	qy	y rotation
float	qz	z rotation
float	qw	w rotation
integer	i	index if component
boolean	changeInterp	change interpolation

Code

```
1546  function Vehicle:setWorldPositionQuaternion(x,y,z, qx,qy,qz,qw, i, changeInterp)
1547  local component = self.components[i]
1548  setWorldTranslation(component.node, x,y,z)
1549  setWorldQuaternion(component.node, qx,qy,qz,qw)
1550  if changeInterp then
1551  component.networkInterpolators.quaternion:setQuaternion(qx, qy, qz, qw)
1552  component.networkInterpolators.position:setPosition(x,y,z)
1553  end
1554  end
```

getPrice

Description

Returns price

Definition

getPrice(float price)

Arguments

float price price

Code

```
1578  function Vehicle:getPrice()
1579  return self.price
```

```
1580  end
```

getSellPrice

Description

Get sell price

Definition

getSellPrice()

Return Values

float sellPrice sell price

Code

```
1585  function Vehicle:getSellPrice()
1586  local priceMultiplier = 0.75
1587  local storeItem = g_storeManager:getItemByXMLFilename(self.configFileName)
1588  local maxVehicleAge = storeItem.lifetime
1589
1590  if maxVehicleAge ~= nil and maxVehicleAge ~= 0 then
1591  local ageMultiplier = 0.5 * math.min(self.age/maxVehicleAge, 1)
1592  local operatingTime = self.operatingTime / (1000*60*60)
1593  local operatingTimeMultiplier = 0.5 * math.min(operatingTime / (maxVehicleAge*EconomyManager.LIFETIME_OPERATINGTIME_RATIO), 1)
1594  priceMultiplier = priceMultiplier * math.exp(-3.5 * (ageMultiplier+operatingTimeMultiplier))
1595  end
1596
1597  return math.max(math.floor(self:getPrice() * math.max(priceMultiplier, 0.05)) - self:getRepairPrice(true), 0)
1598  end
```

getIsOnField

Description

Returns true if vehicle is on a field

Definition

getIsOnField()

Return Values

boolean isOnField is on field

Code

```
1603  function Vehicle:getIsOnField()
1604  local densityBits = 0
1605  for _,component in pairs(self.components) do
1606  local wx, wy, wz = getWorldTranslation(component.node)
1607
1608  local h = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, wx, wy, wz)
```

```
1609  if h-1 > wy then -- 1m threshold since ground tools are working slightly under the ground
1610  break
1611  end
1612
1613  local bits = getDensityAtWorldPos(g_currentMission.terrainDetailId, wx, wy, wz)
1614  densityBits = bitOR(densityBits, bits)
1615  if densityBits ~= 0 then
1616  return true
1617  end
1618  end
1619
1620  return false
1621  end
```

getParentComponent

Description

Get parent component of node

Definition

getParentComponent(integer node)

Arguments

integer node id of node

Return Values

integer parentComponent id of parent component node

Code

```
1627  function Vehicle:getParentComponent(node)
1628  while node ~= 0 do
1629  if self:getIsVehicleNode(node) then
1630  return node
1631  end
1632  node = getParent(node)
1633  end
1634  return 0
1635  end
```

getLastSpeed

Description

Returns last speed in kph

Definition

getLastSpeed(boolean useAttacherVehicleSpeed)

Arguments

boolean useAttacherVehicleSpeed use speed of attacher vehicle

Return Values

float lastSpeed last speed

Code

```
1641  function Vehicle:getLastSpeed(useAttacherVehicleSpeed)
1642  if useAttacherVehicleSpeed then
1643  if self.attacherVehicle ~= nil then
1644  return self.attacherVehicle:getLastSpeed(true)
1645  end
1646  end
1647
1648  return self.lastSpeed * 3600
1649  end
```

getOwner

Description

Get owner of vehicle

Definition

getOwner()

Return Values

table owner owner

Code

```
1659  function Vehicle:getOwner()
1660  if self.owner ~= nil then
1661  return self.owner
1662  end
1663
1664  return nil
1665  end
```

getActiveFarm

Description

Get the active farm. Preferable the one of the controlling player. Otherwise the owner.

Definition

getActiveFarm()

Return Values

integer

getIsVehicleNode

Description

Returns true if node is from vehicle

Definition

getIsVehicleNode(integer nodeId)

Arguments

integer nodeId node id

Return Values

boolean isFromVehicle is from vehicle

Code

```
1677  function Vehicle:getIsVehicleNode(nodeId)
1678  return self.vehicleNodes[nodeId] ~= nil
1679  end
```

getIsOperating

Description

Returns true if is operating

Definition

getIsOperating()

Return Values

boolean isOperating is operating

Code

```
1684  function Vehicle:getIsOperating()
1685  return false
1686  end
```

getTotalMass

Description

Returns total mass of vehicle (optional including attached vehicles)

Definition

getTotalMass(boolean onlyGivenVehicle)

Arguments

boolean onlyGivenVehicle use only the given vehicle, if false or nil it includes all attachables

Return Values

float totalMass total mass

Code

```
2070  function Vehicle:getTotalMass(onlyGivenVehicle)
2071  if self.isServer then
2072  local mass = 0
2073
2074  for _, component in ipairs(self.components) do
2075  mass = mass + component.mass
2076  end
2077
2078  return mass
2079  end
2080
2081  return 0
2082  end
```

getFillLevelInformation

Description

Get fill level information

Definition

getFillLevelInformation(table fillLevelInformations)

Arguments

table fillLevelInformations fill level informations

Code

```
2087  function Vehicle:getFillLevelInformation(fillLevelInformations)
2088  end
```

activate

Description

Called on activate

Definition

activate()

Code

```
2092  function Vehicle:activate()
2093  SpecializationUtil.raiseEvent(self, "onActivate")
2094  end
```

deactivate

Description

Called on deactivate

Definition

deactivate()

Code

```
2098  function Vehicle:deactivate()
2099  if self:getDeactivateOnLeave() then
2100  SpecializationUtil.raiseEvent(self, "onDeactivate")
2101  end
2102  end
```

setComponentJointFrame

Description

Set component joint frame

Definition

setComponentJointFrame(integer jointDesc, integer anchorActor)

Arguments

integer jointDesc joint desc index

integer anchorActor anchor actor

Code

```
2109  function Vehicle:setComponentJointFrame(jointDesc, anchorActor)
2110  if anchorActor == 0 then
2111  local localPoses = jointDesc.jointLocalPoses[1]
2112  localPoses.trans[1], localPoses.trans[2], localPoses.trans[3] =
      localToLocal(jointDesc.jointNode,
      self.components[jointDesc.componentIndices[1]].node, 0, 0, 0)
```

```
2113    localPoses.rot[1], localPoses.rot[2], localPoses.rot[3] =
        localRotationToLocal(jointDesc.jointNode,
        self.components[jointDesc.componentIndices[1]].node, 0, 0, 0)
2114    else
2115    local localPoses = jointDesc.jointLocalPoses[2]
2116    localPoses.trans[1], localPoses.trans[2], localPoses.trans[3] =
        localToLocal(jointDesc.jointNodeActor1,
        self.components[jointDesc.componentIndices[2]].node, 0, 0, 0)
2117    localPoses.rot[1], localPoses.rot[2], localPoses.rot[3] =
        localRotationToLocal(jointDesc.jointNodeActor1,
        self.components[jointDesc.componentIndices[2]].node, 0, 0, 0)
2118    end
2119
2120    local jointNode = jointDesc.jointNode
2121    if anchorActor == 1 then
2122    jointNode = jointDesc.jointNodeActor1
2123    end
2124
2125    if jointDesc.jointIndex ~= 0 then
2126    setJointFrame(jointDesc.jointIndex, anchorActor, jointNode)
2127    end
2128    end
```

setComponentJointRotLimit

Description

Set component joint rot limit

Definition

setComponentJointRotLimit(integer componentJoint, integer axis, float minLimit, float maxLimit)

Arguments

integer	componentJoint	index of component joint
integer	axis	axis
float	minLimit	min limit
float	maxLimit	max limit

Code

```
2137    function Vehicle:setComponentJointRotLimit(componentJoint, axis,
        minLimit, maxLimit)
2138    if self.isServer then
2139    componentJoint.rotLimit[axis] = maxLimit
2140    componentJoint.rotMinLimit[axis] = minLimit
2141
2142    if componentJoint.jointIndex ~= 0 then
2143    if minLimit <= maxLimit then
```

```
2144    setJointRotationLimit(componentJoint.jointIndex, axis-1, true, minLimit, maxLimit)
2145    else
2146    setJointRotationLimit(componentJoint.jointIndex, axis-1, false, 0, 0)
2147    end
2148    end
2149    end
2150    end
```

setComponentJointTransLimit

Description

Set component joint trans limit

Definition

setComponentJointTransLimit(integer componentJoint, integer axis, float minLimit, float maxLimit)

Arguments

integer	componentJoint	index of component joint
integer	axis	axis
float	minLimit	min limit
float	maxLimit	max limit

Code

```
2158    function Vehicle:setComponentJointTransLimit(componentJoint, axis, minLimit, maxLimit)
2159    if self.isServer then
2160    componentJoint.transLimit[axis] = maxLimit
2161    componentJoint.transMinLimit[axis] = minLimit
2162
2163    if componentJoint.jointIndex ~= 0 then
2164    if minLimit <= maxLimit then
2165    setJointTranslationLimit(componentJoint.jointIndex, axis-1, true, minLimit, maxLimit)
2166    else
2167    setJointTranslationLimit(componentJoint.jointIndex, axis-1, false, 0, 0)
2168    end
2169    end
2170    end
2171    end
```

loadComponentFromXML

Description

Load component from xml

Definition

loadComponentFromXML(table component, integer xmlFile, string key, table rootPosition, integer i)

Arguments

table	component	component
integer	xmlFile	id of xml object
string	key	key
table	rootPosition	root position (x, y, z)
integer	i	component index

Return Values

boolean success success

Code

```
2181  function Vehicle:loadComponentFromXML(component, xmlFile, key, rootPosition, i)
2182  if not self.isServer then
2183  if getRigidBodyType(component.node) == "Dynamic" then
2184  setRigidBodyType(component.node, "Kinematic")
2185  end
2186  end
2187  link(getRootNode(), component.node)
2188  if i == 1 then
2189  rootPosition[1], rootPosition[2], rootPosition[3] = getTranslation(component.node)
2190  if rootPosition[2] ~= 0 then
2191  g_logManager:xmlWarning(self.configFileName, "Y-Translation of component 1 (node 0>) has to be 0. Current value is: %.5f", rootPosition[2])
2192  end
2193  end
2194
2195  if getRigidBodyType(component.node) == "Static" then
2196  component.isStatic = true
2197  elseif getRigidBodyType(component.node) == "Kinematic" then
2198  component.isKinematic = true
2199  elseif getRigidBodyType(component.node) == "Dynamic" then
2200  component.isDynamic = true
2201  end
2202
2203  -- the position of the first component is the zero
2204  translate(component.node, -rootPosition[1], -rootPosition[2], -rootPosition[3])
2205  local x,y,z = getTranslation(component.node)
2206  local rx,ry,rz = getRotation(component.node)
2207  component.originalTranslation = {x,y,z}
```

```
2208  component.originalRotation = {rx,ry,rz}
2209
2210  component.sentTranslation = {x,y,z}
2211  component.sentRotation = {rx,ry,rz}
2212
2213  component.defaultMass = nil
2214  component.mass = nil
2215
2216  local mass = getXMLFloat(xmlFile, key.."#mass")
2217  if mass ~= nil then
2218  if mass < 10 then
2219  g_logManager:xmlDevWarning(self.configFileName, "Mass is lower
      than 10kg for '%s'. Mass unit is kilogramms. Is this correct?",
      key)
2220  end
2221  if component.isDynamic then
2222  setMass(component.node, mass/1000)
2223  end
2224
2225  component.defaultMass = mass/1000
2226  component.mass = component.defaultMass
2227  component.lastMass = component.mass
2228  else
2229  g_logManager:xmlWarning(self.configFileName, "Missing 'mass' for
      '%s'. Using default mass 500kg instead!", key)
2230  component.defaultMass = 0.5
2231  component.mass = 0.5
2232  component.lastMass = component.mass
2233  end
2234
2235  local comStr = getXMLString(xmlFile, key .. "#centerOfMass");
2236  if comStr ~= nil then
2237  local com = StringUtil.getVectorNFromString(comStr, 3)
2238  if com ~= nil then
2239  setCenterOfMass(component.node, com[1], com[2], com[3])
2240  else
2241  g_logManager:xmlWarning(self.configFileName, "Invalid center of
      mass given for '%s'. Ignoring this definition", key)
2242  end
2243  end
2244  local count = getXMLInt(xmlFile, key .. "#solverIterationCount")
```

```
2245  if count ~= nil then
2246  setSolverIterationCount(component.node, count)
2247  component.solverIterationCount = count
2248  end
2249  component.motorized = getXMLBool(xmlFile, key .. "#motorized") -
      - Note: motorized is nil if not set in the xml, and can be set
      by the wheels
2250  self.vehicleNodes[component.node] = {component=component}
2251  local clipDistance = getClipDistance(component.node)
2252  if clipDistance >= 1000000 and getVisibility(component.node)
      then
2253  local defaultClipdistance = 300
2254  g_logManager:xmlWarning(self.configFileName, "No clipdistance is
      set to component node '%s' (%s>). Set default clipdistance
      '%d'", getName(component.node), i-1, defaultClipdistance)
2255  setClipDistance(component.node, defaultClipdistance)
2256  end
2257
2258  component.collideWithAttachables =
      Utils.getNoNil(getXMLBool(xmlFile,
      key.."#collideWithAttachables"), false)
2259
2260  if getRigidBodyType(component.node) ~= "NoRigidBody" then
2261  if getLinearDamping(component.node) > 0.01 then
2262  g_logManager:xmlDevWarning(self.configFileName, "Non-zero linear
      damping (%.4f) for component node '%s' (%s>). Is this correct?",
      getLinearDamping(component.node), getName(component.node), i-1)
2263  elseif getAngularDamping(component.node) > 0.05 then
2264  g_logManager:xmlDevWarning(self.configFileName, "Large angular
      damping (%.4f) for component node '%s' (%s>). Is this correct?",
      getAngularDamping(component.node), getName(component.node), i-1)
2265  elseif getAngularDamping(component.node) < 0.0001 then
2266  g_logManager:xmlDevWarning(self.configFileName, "Zero damping
      for component node '%s' (%s>). Is this correct?",
      getName(component.node), i-1)
2267  end
2268  end
2269
2270  local name = getName(component.node)
2271  if not StringUtil.endsWith(name, "component"..i) then
2272  g_logManager:xmlDevWarning(self.configFileName, "Name of
      component '%d' ('%s') does not correpond with the component
      naming convention! (vehicleName_componentName_component%d)", i,
      name, i)
```

```
2273  end
2274
2275  return true
2276  end
```

loadComponentJointFromXML

Description

Load component joints from xml

Definition

loadComponentJointFromXML(table jointDesc, integer xmlFile, string key, integer componentJointI, integer jointNode, integer index1, integer index2)

Arguments

table	jointDesc	joint desc
integer	xmlFile	id of xml object
string	key	key
integer	componentJointI	component joint index
integer	jointNode	id of joint node
integer	index1	index of component 1
integer	index2	index of component 2

Return Values

boolean success success

Code

```
2288  function Vehicle:loadComponentJointFromXML(jointDesc, xmlFile, key, componentJointI, jointNode, index1, index2)
2289  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, key .. "#indexActor1", key .. "#nodeActor1") --FS17 to FS19
2290
2291  jointDesc.componentIndices = {index1, index2}
2292  jointDesc.jointNode = jointNode
2293  jointDesc.jointNodeActor1 = Utils.getNoNil(I3DUtil.indexToObject(self.components, getXMLString(xmlFile, key.."#nodeActor1"), self.i3dMappings), jointNode)
2294  if self.isServer then
2295  if self.components[index1] == nil or self.components[index2] == nil then
2296  g_logManager:xmlWarning(self.configFileName, "Invalid component indices (component1: %d, component2: %d) for component joint %d. Indices start with 1!", index1, index2, componentJointI)
2297  return false
2298  end
2299
2300  local x, y, z = StringUtil.getVectorFromString(getXMLString(xmlFile, key.."#rotLimit"))
```

```
2301  local rotLimits = { math.rad(Utils.getNoNil(x, 0)),
      math.rad(Utils.getNoNil(y, 0)), math.rad(Utils.getNoNil(z, 0)) }
2302  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#transLimit"))
2303  local transLimits = { Utils.getNoNil(x, 0), Utils.getNoNil(y,
      0), Utils.getNoNil(z, 0) }
2304  jointDesc.rotLimit = rotLimits
2305  jointDesc.transLimit = transLimits
2306
2307  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#rotMinLimit"))
2308  local rotMinLimits = { Utils.getNoNilRad(x, nil),
      Utils.getNoNilRad(y, nil), Utils.getNoNilRad(z, nil) }
2309
2310  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#transMinLimit"))
2311  local transMinLimits = { x,y,z }
2312
2313  for i=1,3 do
2314  if rotMinLimits[i] == nil then
2315  if rotLimits[i] >= 0 then
2316  rotMinLimits[i] = -rotLimits[i]
2317  else
2318  rotMinLimits[i] = rotLimits[i]+1
2319  end
2320  end
2321  if transMinLimits[i] == nil then
2322  if transLimits[i] >= 0 then
2323  transMinLimits[i] = -transLimits[i]
2324  else
2325  transMinLimits[i] = transLimits[i]+1
2326  end
2327  end
2328  end
2329
2330  jointDesc.jointLocalPoses = {}
2331  local trans = {localToLocal(jointDesc.jointNode,
      self.components[index1].node, 0, 0, 0)}
2332  local rot = {localRotationToLocal(jointDesc.jointNode,
      self.components[index1].node, 0, 0, 0)}
```

```
2333  jointDesc.jointLocalPoses[1] = {trans=trans, rot=rot}
2334
2335  local trans = {localToLocal(jointDesc.jointNodeActor1,
      self.components[index2].node, 0, 0, 0)}
2336  local rot = {localRotationToLocal(jointDesc.jointNodeActor1,
      self.components[index2].node, 0, 0, 0)}
2337  jointDesc.jointLocalPoses[2] = {trans=trans, rot=rot}
2338
2339  jointDesc.rotMinLimit = rotMinLimits
2340  jointDesc.transMinLimit = transMinLimits
2341
2342  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#rotLimitSpring"))
2343  local rotLimitSpring = { Utils.getNoNil(x, 0), Utils.getNoNil(y,
      0), Utils.getNoNil(z, 0) }
2344  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#rotLimitDamping"))
2345  local rotLimitDamping = { Utils.getNoNil(x, 1),
      Utils.getNoNil(y, 1), Utils.getNoNil(z, 1) }
2346  jointDesc.rotLimitSpring = rotLimitSpring
2347  jointDesc.rotLimitDamping = rotLimitDamping
2348
2349  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#rotLimitForceLimit"))
2350  local rotLimitForceLimit = { Utils.getNoNil(x, -1),
      Utils.getNoNil(y, -1), Utils.getNoNil(z, -1) }
2351  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#transLimitForceLimit"))
2352  local transLimitForceLimit = { Utils.getNoNil(x, -1),
      Utils.getNoNil(y, -1), Utils.getNoNil(z, -1) }
2353  jointDesc.rotLimitForceLimit = rotLimitForceLimit
2354  jointDesc.transLimitForceLimit = transLimitForceLimit
2355
2356  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#transLimitSpring"))
2357  local transLimitSpring = { Utils.getNoNil(x, 0),
      Utils.getNoNil(y, 0), Utils.getNoNil(z, 0) }
2358  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#transLimitDamping"))
```

```
2359  local transLimitDamping = { Utils.getNoNil(x, 1),
      Utils.getNoNil(y, 1), Utils.getNoNil(z, 1) }
2360  jointDesc.transLimitSpring = transLimitSpring
2361  jointDesc.transLimitDamping = transLimitDamping
2362
2363  jointDesc.zRotationXOffset = 0
2364  local zRotationNode = I3DUtil.indexToObject(self.components,
      getXMLString(xmlFile, key.."#zRotationNode"), self.i3dMappings)
2365  if zRotationNode ~= nil then
2366  jointDesc.zRotationXOffset,_,_ = localToLocal(zRotationNode,
      jointNode, 0,0,0)
2367  end
2368
2369  jointDesc.isBreakable = Utils.getNoNil(getXMLBool(xmlFile,
      key.."#breakable"), false)
2370  if jointDesc.isBreakable then
2371  jointDesc.breakForce = Utils.getNoNil(getXMLFloat(xmlFile,
      key.."#breakForce"), 10)
2372  jointDesc.breakTorque = Utils.getNoNil(getXMLFloat(xmlFile,
      key.."#breakTorque"), 10)
2373  end
2374  jointDesc.enableCollision = Utils.getNoNil(getXMLBool(xmlFile,
      key.."#enableCollision"), false)
2375
2376  -- Rotational drive
2377  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#maxRotDriveForce"))
2378  local maxRotDriveForce = { Utils.getNoNil(x, 0),
      Utils.getNoNil(y, 0), Utils.getNoNil(z, 0) }
2379  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#rotDriveVelocity"))
2380  local rotDriveVelocity = { Utils.getNoNilRad(x, nil),
      Utils.getNoNilRad(y, nil), Utils.getNoNilRad(z, nil) } --
      convert from deg/s to rad/s
2381  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#rotDriveRotation"))
2382  local rotDriveRotation = { Utils.getNoNilRad(x, nil),
      Utils.getNoNilRad(y, nil), Utils.getNoNilRad(z, nil) } --
      convert from deg to rad
2383  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#rotDriveSpring"))
```

```
2384  local rotDriveSpring = { Utils.getNoNil(x, 0), Utils.getNoNil(y,
      0), Utils.getNoNil(z, 0) }
2385  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#rotDriveDamping"))
2386  local rotDriveDamping = { Utils.getNoNil(x, 0),
      Utils.getNoNil(y, 0), Utils.getNoNil(z, 0) }
2387
2388  jointDesc.rotDriveVelocity = rotDriveVelocity
2389  jointDesc.rotDriveRotation = rotDriveRotation
2390  jointDesc.rotDriveSpring = rotDriveSpring
2391  jointDesc.rotDriveDamping = rotDriveDamping
2392  jointDesc.maxRotDriveForce = maxRotDriveForce
2393
2394  -- Translational drive
2395  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#transDriveVelocity"))
2396  local transDriveVelocity = { x,y,z }
2397
2398  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#transDrivePosition"))
2399  local transDrivePosition = { x,y,z }
2400
2401  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#transDriveSpring"))
2402  local transDriveSpring = { Utils.getNoNil(x, 0),
      Utils.getNoNil(y, 0), Utils.getNoNil(z, 0) }
2403
2404  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#transDriveDamping"))
2405  local transDriveDamping = { Utils.getNoNil(x, 1),
      Utils.getNoNil(y, 1), Utils.getNoNil(z, 1) }
2406
2407  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      key.."#maxTransDriveForce"))
2408  local maxTransDriveForce = { Utils.getNoNil(x, 0),
      Utils.getNoNil(y, 0), Utils.getNoNil(z, 0) }
2409
2410  jointDesc.transDriveVelocity = transDriveVelocity
2411  jointDesc.transDrivePosition = transDrivePosition
```

```
2412  jointDesc.transDriveSpring = transDriveSpring
2413  jointDesc.transDriveDamping = transDriveDamping
2414  jointDesc.maxTransDriveForce = maxTransDriveForce
2415
2416  jointDesc.jointIndex = 0
2417  end
2418
2419  return true
2420  end
```

createComponentJoint

Description

Create component joint between two components

Definition

createComponentJoint(table component1, table component2, table jointDesc)

Arguments

table component1	component 1
table component2	component 2
table jointDesc	joint desc

Return Values

boolean success success

Code

```
2428  function Vehicle:createComponentJoint(component1, component2,
      jointDesc)
2429  if component1 == nil or component2 == nil or jointDesc == nil then
2430  g_logManager:xmlWarning(self.configFileName, "Could not create
      component joint. No component1, component2 or jointDesc given!")
2431  return false
2432  end
2433
2434  local constr = JointConstructor:new()
2435  constr:setActors(component1.node, component2.node)
2436
2437  local localPoses1 = jointDesc.jointLocalPoses[1]
2438  local localPoses2 = jointDesc.jointLocalPoses[2]
2439  constr:setJointLocalPositions(localPoses1.trans[1],
      localPoses1.trans[2], localPoses1.trans[3], localPoses2.trans[1],
      localPoses2.trans[2], localPoses2.trans[3])
2440  constr:setJointLocalRotations(localPoses1.rot[1], localPoses1.rot[2],
      localPoses1.rot[3], localPoses2.rot[1], localPoses2.rot[2],
      localPoses2.rot[3])
2441  --constr:setJointTransforms(jointDesc.jointNode,
      jointDesc.jointNodeActor1)
2442
```

```
2443  constr:setRotationLimitSpring(jointDesc.rotLimitSpring[1],
      jointDesc.rotLimitDamping[1], jointDesc.rotLimitSpring[2],
      jointDesc.rotLimitDamping[2], jointDesc.rotLimitSpring[3],
      jointDesc.rotLimitDamping[3])
2444  constr:setTranslationLimitSpring(jointDesc.transLimitSpring[1],
      jointDesc.transLimitDamping[1], jointDesc.transLimitSpring[2],
      jointDesc.transLimitDamping[2], jointDesc.transLimitSpring[3],
      jointDesc.transLimitDamping[3])
2445  constr:setZRotationXOffset(jointDesc.zRotationXOffset)
2446  for i=1, 3 do
2447  if jointDesc.rotLimit[i] >= jointDesc.rotMinLimit[i] then
2448  constr:setRotationLimit(i-1, jointDesc.rotMinLimit[i],
      jointDesc.rotLimit[i])
2449  end
2450
2451  if jointDesc.transLimit[i] >= jointDesc.transMinLimit[i] then
2452  constr:setTranslationLimit(i-1, true, jointDesc.transMinLimit[i],
      jointDesc.transLimit[i])
2453  else
2454  constr:setTranslationLimit(i-1, false, 0, 0)
2455  end
2456  end
2457
2458  constr:setRotationLimitForceLimit(jointDesc.rotLimitForceLimit[1],
      jointDesc.rotLimitForceLimit[2], jointDesc.rotLimitForceLimit[3])
2459  constr:setTranslationLimitForceLimit(jointDesc.transLimitForceLimit[1],
      jointDesc.transLimitForceLimit[2], jointDesc.transLimitForceLimit[3])
2460
2461  if jointDesc.isBreakable then
2462  constr:setBreakable(jointDesc.breakForce, jointDesc.breakTorque)
2463  end
2464  constr:setEnableCollision(jointDesc.enableCollision)
2465
2466  for i=1,3 do
2467  if jointDesc.maxRotDriveForce[i] > 0.0001 and
      (jointDesc.rotDriveVelocity[i] ~= nil or jointDesc.rotDriveRotation[i]
      ~= nil) then
2468  local pos = Utils.getNoNil(jointDesc.rotDriveRotation[i], 0)
2469  local vel = Utils.getNoNil(jointDesc.rotDriveVelocity[i], 0)
2470  constr:setAngularDrive(i-1, jointDesc.rotDriveRotation[i] ~= nil,
      jointDesc.rotDriveVelocity[i] ~= nil, jointDesc.rotDriveSpring[i],
      jointDesc.rotDriveDamping[i], jointDesc.maxRotDriveForce[i], pos, vel)
2471  end
```

```
2472  if jointDesc.maxTransDriveForce[i] > 0.0001 and
      (jointDesc.transDriveVelocity[i] ~= nil or
      jointDesc.transDrivePosition[i] ~= nil) then
2473  local pos = Utils.getNoNil(jointDesc.transDrivePosition[i], 0)
2474  local vel = Utils.getNoNil(jointDesc.transDriveVelocity[i], 0)
2475  constr:setLinearDrive(i-1, jointDesc.transDrivePosition[i] ~= nil,
      jointDesc.transDriveVelocity[i] ~= nil, jointDesc.transDriveSpring[i],
      jointDesc.transDriveDamping[i], jointDesc.maxTransDriveForce[i], pos,
      vel)
2476  end
2477  end
2478
2479  jointDesc.jointIndex = constr:finalize()
2480
2481  return true
2482  end
```

prefixSchemaOverlayName

Description

Add a mod prefix to a schema overlay name to match name loading in HUD which avoids name collisions.
First checks if there is a matching default schema overlay name and only adds the prefix if there is none.

Definition

prefixSchemaOverlayName(string baseName, string prefix)

Arguments

string baseName Base schema overlay name

string prefix Name prefix to add if the baseName parameter is not one of the default schema overlays

Return Values

string Schema overlay name which was modified if necessary

loadSchemaOverlay

Description

Load schema overlay data from xml file
The HUD draws the schema from this information and handles all required visual components.

Definition

loadSchemaOverlay(integer xmlFile)

Arguments

integer xmlFile id of xml object

Code

```
2502  function Vehicle:loadSchemaOverlay(xmlFile)
2503  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName, "vehicle.schemaOverlay#file") --FS17 to
      FS19
```

```
2504  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName, "vehicle.schemaOverlay#width") --FS17 to
      FS19
2505  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName, "vehicle.schemaOverlay#height") --FS17 to
      FS19
2506  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName,
      "vehicle.schemaOverlay#invisibleBorderRight",
      "vehicle.base.schemaOverlay#invisibleBorderRight") --FS17 to
      FS19
2507  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName,
      "vehicle.schemaOverlay#invisibleBorderLeft",
      "vehicle.base.schemaOverlay#invisibleBorderLeft") --FS17 to FS19
2508  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName,
      "vehicle.schemaOverlay#attacherJointPosition",
      "vehicle.base.schemaOverlay#attacherJointPosition") --FS17 to
      FS19
2509  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName, "vehicle.schemaOverlay#basePosition",
      "vehicle.base.schemaOverlay#basePosition") --FS17 to FS19
2510  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName, "vehicle.schemaOverlay#fileSelected") --
      FS17 to FS19
2511  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName, "vehicle.schemaOverlay#fileTurnedOn") --
      FS17 to FS19
2512  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName,
      "vehicle.schemaOverlay#fileSelectedTurnedOn") --FS17 to FS19
2513
2514  if hasXMLProperty(xmlFile, "vehicle.base.schemaOverlay") then
2515  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      "vehicle.schemaOverlay.attacherJoint",
      "vehicle.attacherJoints.attacherJoint.schema") -- FS17
2516
2517  local x, y =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      "vehicle.base.schemaOverlay#attacherJointPosition"))
2518  local baseX, baseY =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      "vehicle.base.schemaOverlay#basePosition"))
2519
2520  if baseX == nil then
2521  baseX = x
2522  end
```

```
2523
2524  if baseY == nil then
2525  baseY = y
2526  end
2527
2528  local schemaNameDefault = getXMLString(xmlFile,
      "vehicle.base.schemaOverlay.default#name") or ""
2529  local schemaNameTurnedOn = getXMLString(xmlFile,
      "vehicle.base.schemaOverlay.turnedOn#name") or ""
2530  local schemaNameSelected = getXMLString(xmlFile,
      "vehicle.base.schemaOverlay.selected#name") or ""
2531  local schemaNameSelectedTurnedOn = getXMLString(xmlFile,
      "vehicle.base.schemaOverlay.turnedOnSelected#name") or ""
2532
2533  local modPrefix = self.customEnvironment or ""
2534  schemaNameDefault =
      Vehicle.prefixSchemaOverlayName(schemaNameDefault, modPrefix)
2535  schemaNameTurnedOn =
      Vehicle.prefixSchemaOverlayName(schemaNameTurnedOn, modPrefix)
2536  schemaNameSelected =
      Vehicle.prefixSchemaOverlayName(schemaNameSelected, modPrefix)
2537  schemaNameSelectedTurnedOn =
      Vehicle.prefixSchemaOverlayName(schemaNameSelectedTurnedOn,
      modPrefix)
2538
2539  self.schemaOverlay = VehicleSchemaOverlayData.new(
2540  baseX, baseY,
2541  schemaNameDefault,
2542  schemaNameTurnedOn,
2543  schemaNameSelected,
2544  schemaNameSelectedTurnedOn,
2545  getXMLFloat(xmlFile,
      "vehicle.base.schemaOverlay#invisibleBorderRight"),
2546  getXMLFloat(xmlFile,
      "vehicle.base.schemaOverlay#invisibleBorderLeft"))
2547  end
2548  end
```

dayChanged

Description

Called if day changed

Definition

dayChanged()

Code

```
2556  function Vehicle:dayChanged()
2557  self.age = self.age + 1
2558  end
```

getSpeedLimit

Description

Get speed limit

Definition

getSpeedLimit(boolean onlyIfWorking)

Arguments

boolean onlyIfWorking only if working

Return Values

float limit limit

boolean doCheckSpeedLimit do check speed limit

Code

```
2641  function Vehicle:getSpeedLimit(onlyIfWorking)
2642  local limit = math.huge
2643  local doCheckSpeedLimit = self:doCheckSpeedLimit()
2644  if onlyIfWorking == nil or (onlyIfWorking and doCheckSpeedLimit)
      then
2645  limit = self.speedLimit
2646
2647  local damage = self:getVehicleDamage()
2648  if damage > 0 then
2649  limit = limit * (1 - damage *
      Vehicle.DAMAGED_SPEEDLIMIT_REDUCTION)
2650  end
2651  end
2652
2653  local attachedImplements
2654  if self.getAttachedImplements ~= nil then
2655  attachedImplements = self:getAttachedImplements()
2656  end
2657  if attachedImplements ~= nil then
2658  for _, implement in pairs(attachedImplements) do
2659  if implement.object ~= nil then
2660  local speed, implementDoCheckSpeedLimit =
      implement.object:getSpeedLimit(onlyIfWorking)
2661  if onlyIfWorking == nil or (onlyIfWorking and
      implementDoCheckSpeedLimit) then
2662  limit = math.min(limit, speed)
2663  end
```

```
2664    doCheckSpeedLimit = doCheckSpeedLimit or implementDoCheckSpeedLimit
2665    end
2666    end
2667    end
2668    return limit, doCheckSpeedLimit
2669    end
```

getDailyUpkeep

Description

Get daily up keep

Definition

getDailyUpkeep()

Return Values

float dailyUpkeep daily up keep

Code

```
2682    function Vehicle:getDailyUpkeep()
2683    local storeItem = g_storeManager:getItemByXMLFilename(self.configFileName)
2684
2685    local multiplier = 1
2686    if storeItem.lifetime ~= nil and storeItem.lifetime ~= 0 then
2687    local ageMultiplier = 0.3 * math.min(self.age/storeItem.lifetime, 1)
2688    local operatingTime = self.operatingTime / (1000*60*60)
2689    local operatingTimeMultiplier = 0.7 * math.min(operatingTime / (storeItem.lifetime*EconomyManager.LIFETIME_OPERATINGTIME_RATIO), 1)
2690    multiplier = 1 + EconomyManager.MAX_DAILYUPKEEP_MULTIPLIER * (ageMultiplier+operatingTimeMultiplier)
2691    end
2692
2693    return StoreItemUtil.getDailyUpkeep(storeItem, self.configurations) * multiplier
2694    end
```

getReloadXML

Description

Get reload xml

Definition

getReloadXML(table vehicle)

Arguments

table vehicle vehicle

Return Values

string xml xml

Code

```
2767  function Vehicle.getReloadXML(vehicle)
2768
2769  local vehicleXMLFile = createXMLFile("vehicleXMLFile", "", "vehicles")
2770  if vehicleXMLFile ~= nil then
2771  local key = string.format("vehicles.vehicle(%d)", 0)
2772
2773  setXMLInt(vehicleXMLFile, key.."#id", 1)
2774  setXMLString(vehicleXMLFile, key.."#filename",
      HTMLUtil.encodeToHTML(NetworkUtil.convertToNetworkFilename(vehicle.conf
2775
2776  vehicle:saveToXMLFile(vehicleXMLFile, key, {})
2777
2778  return vehicleXMLFile
2779  end
2780
2781  return nil
2782  end
```

VehicleCamera

Description

new

Description

Creating vehicle camera

Definition

new(boolean isServer, boolean isClient, table customMt)

Arguments

boolean	isServer	is server
boolean	isClient	is client
table	customMt	custom metatable

Return Values

table instance Instance of object

Code

```
23  function VehicleCamera:new(vehicle, customMt)
24
25  local instance = {}
26  if customMt ~= nil then
27  setmetatable(instance, customMt)
28  else
29  setmetatable(instance, VehicleCamera_mt)
30  end
31
32
```

```
33  instance.vehicle = vehicle
34  instance.isActivated = false
35
36  instance.limitRotXDelta = 0
37
38  instance.raycastDistance = 0
39  instance.normalX = 0
40  instance.normalY = 0
41  instance.normalZ = 0
42
43  instance.raycastNodes = {}
44  instance.disableCollisionTime = -1
45
46  instance.lookAtPosition = {0,0,0}
47  instance.lookAtLastTargetPosition = {0,0,0}
48  instance.position = {0,0,0}
49  instance.lastTargetPosition = {0,0,0}
50
51  instance.lastInputValues = {}
52  instance.lastInputValues.upDown = 0
53  instance.lastInputValues.leftRight = 0
54
55  instance.isCollisionEnabled = not
    g_modIsLoaded["disableVehicleCameraCollision"]
56
57  return instance
58  end
```

loadFromXML

Description

Load vehicle camera from xml file

Definition

loadFromXML(integer xmlFile, string key)

Arguments

integer xmlFile id of xml object

string key key

Return Values

boolean success success

Code

```
65  function VehicleCamera:loadFromXML(xmlFile, key)
66  XMLUtil.checkDeprecatedXMLElements(xmlFile,
    self.vehicle.configFileName, key .. "#index", "#node") -- FS17 to
    FS19
```

```
67
68  local camIndexStr = getXMLString(xmlFile, key .. "#node")
69  self.cameraNode = I3DUtil.indexToObject(self.vehicle.components,
    camIndexStr, self.vehicle.i3dMappings)
70  if self.cameraNode == nil or not getHasClassId(self.cameraNode,
    ClassIds.CAMERA) then
71  g_logManager:xmlWarning(self.vehicle.configFileName, "Invalid
    camera node for camera '%s'. Must be a camera type!", key)
72  return false
73  end
74
75  self.fovY = calculateFovY(self.cameraNode)
76  setFovY(self.cameraNode, self.fovY)
77
78  self.isRotatable = Utils.getNoNil(getXMLBool(xmlFile, key ..
    "#rotatable"), false)
79  self.limit = Utils.getNoNil(getXMLBool(xmlFile, key .. "#limit"),
    false)
80  if self.limit then
81  self.rotMinX = getXMLFloat(xmlFile, key .. "#rotMinX")
82  self.rotMaxX = getXMLFloat(xmlFile, key .. "#rotMaxX")
83
84  self.transMin = getXMLFloat(xmlFile, key .. "#transMin")
85  self.transMax = getXMLFloat(xmlFile, key .. "#transMax")
86  if self.rotMinX == nil or self.rotMaxX == nil or self.transMin ==
    nil or self.transMax == nil then
87  g_logManager:xmlWarning(self.vehicle.configFileName, "Missing
    'rotMinX', 'rotMaxX', 'transMin' or 'transMax' for camera '%s'",
    key)
88  return false
89  end
90  end
91
92  self.isInside = Utils.getNoNil(getXMLBool(xmlFile, key ..
    "#isInside"), false)
93  if self.isInside then
94  self.defaultLowPassGain = Utils.getNoNil(getXMLFloat(xmlFile, key
    .. "#defaultLowPassGain"), 0.5)
95  self.defaultVolume = Utils.getNoNil(getXMLFloat(xmlFile, key ..
    "#defaultVolume"), 0.9)
96  else
97  self.defaultLowPassGain = Utils.getNoNil(getXMLFloat(xmlFile, key
    .. "#defaultLowPassGain"), 1.0)
```

```
98  self.defaultVolume = Utils.getNoNil(getXMLFloat(xmlFile, key ..
    "#defaultVolume"), 1.0)
99  end
100 self.allowHeadTracking = Utils.getNoNil(getXMLBool(xmlFile, key
    .. "#allowHeadTracking"), self.isInside)
101
102 local shadowBoxIndexStr = getXMLString(xmlFile, key ..
    "#shadowFocusBox")
103 self.shadowFocusBoxNode =
    I3DUtil.indexToObject(self.vehicle.components, shadowBoxIndexStr,
    self.vehicle.i3dMappings)
104 if self.shadowFocusBoxNode ~= nil and not
    getHasClassId(self.shadowFocusBoxNode, ClassIds.SHAPE) then
105 g_logManager:xmlWarning(self.vehicle.configFileName, "Invalid
    camera shadow focus box '%s'. Must be a shape and cpu mesh",
    getName(shadowFocusBoxNode))
106 self.shadowFocusBoxNode = nil;
107 end
108
109 if self.isInside and self.shadowFocusBoxNode == nil then
110 g_logManager:xmlDevWarning(self.vehicle.configFileName, "Missing
    shadow focus box for indoor camera '%s'", key)
111 end
112
113 self.useOutdoorSounds = Utils.getNoNil(getXMLBool(xmlFile, key ..
    "#useOutdoorSounds"), not self.isInside)
114
115 if self.isRotatable then
116 self.rotateNode = I3DUtil.indexToObject(self.vehicle.components,
    getXMLString(xmlFile, key .. "#rotateNode"),
    self.vehicle.i3dMappings)
117 self.hasExtraRotationNode = self.rotateNode ~= nil
118 end
119
120 local rotation =
    StringUtil.getRadiansFromString(getXMLString(xmlFile,
    key.."#rotation"), 3)
121 if rotation ~= nil then
122 local rotationNode = self.cameraNode
123 if self.rotateNode ~= nil then
124 rotationNode = self.rotateNode
125 end
126 setRotation(rotationNode, unpack(rotation))
127 end
```

```
128 local translation = StringUtil.getVectorNFromString(getXMLString(xmlFile, key.."#translation"), 3)
129 if translation ~= nil then
130 setTranslation(self.cameraNode, unpack(translation))
131 end
132
133 self.allowTranslation = (self.rotateNode ~= nil and self.rotateNode ~= self.cameraNode)
134
135 self.useMirror = Utils.getNoNil(getXMLBool(xmlFile, key .. "#useMirror"), false)
136 self.useWorldXZRotation = getXMLBool(xmlFile, key .. "#useWorldXZRotation") -- overrides the ingame setting
137 self.resetCameraOnVehicleSwitch = getXMLBool(xmlFile, key .. "#resetCameraOnVehicleSwitch") -- overrides the ingame setting
138
139 self.positionSmoothingParameter = 0
140 self.lookAtSmoothingParameter = 0
141 local useDefaultPositionSmoothing = Utils.getNoNil(getXMLBool(xmlFile, key .. "#useDefaultPositionSmoothing"), true)
142 if useDefaultPositionSmoothing then
143 if self.isInside then
144 self.positionSmoothingParameter = 0.128 -- 0.095
145 self.lookAtSmoothingParameter = 0.176 -- 0.12
146 else
147 self.positionSmoothingParameter = 0.016
148 self.lookAtSmoothingParameter = 0.022
149 end
150 end
151 self.positionSmoothingParameter = Utils.getNoNil(getXMLFloat(xmlFile, key .. "#positionSmoothingParameter"), self.positionSmoothingParameter)
152 self.lookAtSmoothingParameter = Utils.getNoNil(getXMLFloat(xmlFile, key .. "#lookAtSmoothingParameter"), self.lookAtSmoothingParameter)
153
154 local useHeadTracking = g_gameSettings:getValue("isHeadTrackingEnabled") and isHeadTrackingAvailable() and self.allowHeadTracking
155 if useHeadTracking then
156 self.positionSmoothingParameter = 0
157 self.lookAtSmoothingParameter = 0
```

```
158 end
159
160 self.cameraPositionNode = self.cameraNode
161 if self.positionSmoothingParameter > 0 then
162 -- create a node which indicates the target position of the
    camera
163 self.cameraPositionNode =
    createTransformGroup("cameraPositionNode")
164 local camIndex = getChildIndex(self.cameraNode)
165 link(getParent(self.cameraNode), self.cameraPositionNode,
    camIndex)
166 local x,y,z = getTranslation(self.cameraNode)
167 local rx,ry,rz = getRotation(self.cameraNode)
168 setTranslation(self.cameraPositionNode, x,y,z)
169 setRotation(self.cameraPositionNode, rx,ry,rz)
170
171 unlink(self.cameraNode)
172 end
173 self.rotYSteeringRotSpeed =
    math.rad(Utils.getNoNil(getXMLFloat(xmlFile, key ..
    "#rotYSteeringRotSpeed"), 0))
174
175 if self.rotateNode == nil or self.rotateNode == self.cameraNode
    then
176 self.rotateNode = self.cameraPositionNode
177 end
178
179 if useHeadTracking then
180 local dx,dy,dz = localDirectionToLocal(self.cameraPositionNode,
    getParent(self.cameraPositionNode), 0,0,1)
181 local tx,ty,tz = localToLocal(self.cameraPositionNode,
    getParent(self.cameraPositionNode), 0,0,0)
182 self.headTrackingNode = createTransformGroup("headTrackingNode")
183 link(getParent(self.cameraPositionNode), self.headTrackingNode)
184 setTranslation(self.headTrackingNode, tx,ty,tz)
185 if math.abs(dx)+math.abs(dz) > 0.0001 then
186 setDirection(self.headTrackingNode, dx, dy, dz, 0,1,0)
187 else
188 setRotation(self.headTrackingNode, 0,0,0)
189 end
190 end
191
```

```
192 self.origRotX, self.origRotY, self.origRotZ = getRotation(self.rotateNode)
193 self.rotX = self.origRotX
194 self.rotY = self.origRotY
195 self.rotZ = self.origRotZ
196
197 self.origTransX, self.origTransY, self.origTransZ = getTranslation(self.cameraPositionNode)
198 self.transX = self.origTransX
199 self.transY = self.origTransY
200 self.transZ = self.origTransZ
201
202 local transLength = MathUtil.vector3Length(self.origTransX, self.origTransY, self.origTransZ) + 0.00001 -- prevent devision by zero
203 self.zoom = transLength
204 self.zoomTarget = transLength
205 self.zoomLimitedTarget = -1
206
207 local trans1OverLength = 1.0/transLength
208 self.transDirX = trans1OverLength*self.origTransX
209 self.transDirY = trans1OverLength*self.origTransY
210 self.transDirZ = trans1OverLength*self.origTransZ
211 if self.allowTranslation then
212 if transLength <= 0.01 then
213 g_logManager:xmlWarning(self.vehicle.configFileName, "Invalid camera translation for camera '%s'. Distance needs to be bigger than 0.01", key)
214 end
215 end
216
217 table.insert(self.raycastNodes, self.rotateNode)
218 local i=0
219 while true do
220 local raycastKey = key..string.format(".raycastNode(%d)", i)
221 if not hasXMLProperty(xmlFile, raycastKey) then
222 break
223 end
224
225 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.vehicle.configFileName, raycastKey .. "#index", raycastKey .. "#node") --FS17 to FS19
```

```
226
227 local node = I3DUtil.indexToObject(self.vehicle.components,
    getXMLString(xmlFile, raycastKey .. "#node"),
    self.vehicle.i3dMappings)
228 if node ~= nil then
229 table.insert(self.raycastNodes, node)
230 end
231
232 i=i+1
233 end
234
235 local sx,sy,sz = getScale(self.cameraNode)
236 if sx ~= 1 or sy ~= 1 or sz ~= 1 then
237 g_logManager:xmlWarning(self.vehicle.configFileName, "Vehicle
    camera with scale found for camera '%s'. Resetting to scale 1",
    key)
238 setScale(self.cameraNode, 1,1,1)
239 end
240
241 self.headTrackingPositionOffset = {0, 0, 0}
242 self.headTrackingRotationOffset = {0, 0, 0}
243
244 return true
245 end
```

delete

Description

Deleting vehicle camera

Definition

delete()

Code

```
249 function VehicleCamera:delete()
250 if self.cameraNode ~= nil and self.positionSmoothingParameter > 0
    then
251 delete(self.cameraNode)
252 self.cameraNode = nil
253 end
254 setShadowFocusBox(0)
255 end
```

zoomSmoothly

Description

Zoom camera smoothly

Definition

zoomSmoothly(float offset)

Arguments

float offset offset

Code

```
260  function VehicleCamera:zoomSmoothly(offset)
261  --self.zoomTarget = self.zoomTarget + offset
262  --if self.limit then
263  -- self.zoomTarget = math.min(self.transMax,
     math.max(self.transMin, self.zoomTarget))
264  --end
265  --self.zoomTarget = math.max(self.zoomTarget, 0.01)
266
267  local zoomTarget = self.zoomTarget
268  if self.transMin ~= nil and self.transMax ~= nil and
     self.transMin ~= self.transMax then
269  zoomTarget = math.min(self.transMax, math.max(self.transMin,
     self.zoomTarget + offset))
270  end
271  self.zoomTarget = zoomTarget
272
273  end
```

raycastCallback

Description

Raycast callback

Definition

raycastCallback(integer transformId, float x, float y, float z, float distance, float nx, float ny, float nz)

Arguments

integer transformId id raycasted object

float	x	x raycast position
float	y	y raycast position
float	z	z raycast position
float	distance	distance to raycast position
float	nx	normal x
float	ny	normal y
float	nz	normal z

Code

```
285  function VehicleCamera:raycastCallback(transformId, x, y, z,
     distance, nx, ny, nz)
286  self.raycastDistance = distance
287  self.normalX = nx
288  self.normalY = ny
289  self.normalZ = nz
```

```
290 self.raycastTransformId = transformId
291 end
```

update

Description

Update

Definition

update(float dt)

Arguments

float dt time since last call in ms

Code

```
296 function VehicleCamera:update(dt)
297 local target = self.zoomTarget
298 if self.zoomLimitedTarget >= 0 then
299 target = math.min(self.zoomLimitedTarget, self.zoomTarget)
300 end
301 self.zoom = target + ( math.pow(0.99579, dt) * (self.zoom -
    target) )
302
303 --
304 if self.lastInputValues.upDown ~= 0 then
305 local value = self.lastInputValues.upDown *
    g_gameSettings:getValue(GameSettings.SETTING.CAMERA_SENSITIVITY)
306 self.lastInputValues.upDown = 0
307
308 if self.isRotatable then
309 if self.isActivated and not g_gui:getIsGuiVisible() then
310 if self.limitRotXDelta > 0.001 then
311 self.rotX = math.min(self.rotX - value, self.rotX)
312 elseif self.limitRotXDelta < -0.001 then
313 self.rotX = math.max(self.rotX - value, self.rotX)
314 else
315 self.rotX = self.rotX - value
316 end
317
318 if self.limit then
319 self.rotX = math.min(self.rotMaxX, math.max(self.rotMinX,
    self.rotX))
320 end
321 end
322 end
323 end
```

```
324
325 if self.lastInputValues.leftRight ~= 0 then
326 local value = self.lastInputValues.leftRight *
    g_gameSettings:getValue(GameSettings.SETTING.CAMERA_SENSITIVITY)
327 self.lastInputValues.leftRight = 0
328
329 if self.isRotatable then
330 if self.isActivated and not g_gui:getIsGuiVisible() then
331 self.rotY = self.rotY - value
332 end
333 end
334 end
335
336 --
337 if g_gameSettings:getValue("isHeadTrackingEnabled") and
    isHeadTrackingAvailable() and self.allowHeadTracking and
    self.headTrackingNode ~= nil then
338 local tx,ty,tz = getHeadTrackingTranslation()
339 local pitch,yaw,roll = getHeadTrackingRotation()
340 if pitch ~= nil then
341 local camParent = getParent(self.cameraNode)
342 local ctx,cty,ctz;
343 local crx,cry,crz;
344 if camParent ~= 0 then
345 ctx, cty, ctz = localToLocal(self.headTrackingNode, camParent,
    tx, ty, tz);
346 crx, cry, crz = localRotationToLocal(self.headTrackingNode,
    camParent, pitch,yaw,roll);
347 else
348 ctx, cty, ctz = localToWorld(self.headTrackingNode, tx, ty, tz);
349 crx, cry, crz = localRotationToWorld(self.headTrackingNode,
    pitch,yaw,roll);
350 end
351
352 setRotation(self.cameraNode, crx, cry, crz)
353 setTranslation(self.cameraNode, ctx, cty, ctz)
354 end
355 else
356
357 self:updateRotateNodeRotation()
358
359
```

```
360 if self.limit then
361 -- adjust rotation to avoid clipping with terrain
362 if self.isRotatable and ((self.useWorldXZRotation == nil and
    g_gameSettings:getValue("useWorldCamera")) or
    self.useWorldXZRotation) then
363
364 local numIterations = 4
365 for i=1, numIterations do
366 local transX, transY, transZ = self.transDirX*self.zoom,
    self.transDirY*self.zoom, self.transDirZ*self.zoom
367 local x,y,z = localToWorld(getParent(self.cameraPositionNode),
    transX, transY, transZ)
368
369 local terrainHeight =
    DensityMapHeightUtil.getHeightAtWorldPos(x,0,z)
370
371 local minHeight = terrainHeight + 0.9
372 if y < minHeight then
373 local h = math.sin(self.rotX)*self.zoom
374 local h2 = h-(minHeight-y)
375 self.rotX = math.asin(MathUtil.clamp(h2/self.zoom, -1, 1))
376 self:updateRotateNodeRotation()
377 else
378 break
379 end
380 end
381 end
382
383 -- adjust zoom to avoid collision with objects
384 if self.allowTranslation then
385
386 self.limitRotXDelta = 0
387 local hasCollision, collisionDistance, nx,ny,nz, normalDotDir =
    self:getCollisionDistance()
388 if hasCollision then
389 local distOffset = 0.1
390 if normalDotDir ~= nil then
391 local absNormalDotDir = math.abs(normalDotDir)
392 distOffset = MathUtil.lerp(1.2, 0.1,
    absNormalDotDir*absNormalDotDir*(3-2*absNormalDotDir))
393 end
394 collisionDistance = math.max(collisionDistance-distOffset, 0.01)
```

```
395 self.disableCollisionTime = g_currentMission.time+400
396 self.zoomLimitedTarget = collisionDistance
397 if collisionDistance < self.zoom then
398 self.zoom = collisionDistance
399 end
400 if self.isRotatable and nx ~= nil and collisionDistance <
    self.transMin then
401 local _,lny,_ = worldDirectionToLocal(self.rotateNode, nx,ny,nz)
402 if lny > 0.5 then
403 self.limitRotXDelta = 1
404 elseif lny < -0.5 then
405 self.limitRotXDelta = -1
406 end
407 end
408 else
409 if self.disableCollisionTime <= g_currentMission.time then
410 self.zoomLimitedTarget = -1
411 end
412 end
413 end
414
415 end
416 self.transX, self.transY, self.transZ = self.transDirX*self.zoom,
    self.transDirY*self.zoom, self.transDirZ*self.zoom
417 setTranslation(self.cameraPositionNode, self.transX, self.transY,
    self.transZ)
418
419 if self.positionSmoothingParameter > 0 then
420
421 local interpDt = g_physicsDt
422 if g_server == nil then
423 -- on clients, we interpolate the vehicles with dt, thus we need
    to use the same for camera interpolation
424 interpDt = dt
425 end
426 if interpDt > 0 then
427 local xlook,ylook,zlook = getWorldTranslation(self.rotateNode)
428 local lookAtPos = self.lookAtPosition
429 local lookAtLastPos = self.lookAtLastTargetPosition
430 lookAtPos[1],lookAtPos[2],lookAtPos[3] =
    self:getSmoothed(self.lookAtSmoothingParameter,
```

```
    lookAtPos[1],lookAtPos[2],lookAtPos[3], xlook,ylook,zlook,
    lookAtLastPos[1],lookAtLastPos[2],lookAtLastPos[3], interpDt)
431 lookAtLastPos[1],lookAtLastPos[2],lookAtLastPos[3] =
    xlook,ylook,zlook
432
433 local x,y,z = getWorldTranslation(self.cameraPositionNode)
434 local pos = self.position
435 local lastPos = self.lastTargetPosition
436 pos[1],pos[2],pos[3] =
    self:getSmoothed(self.positionSmoothingParameter,
    pos[1],pos[2],pos[3], x,y,z, lastPos[1],lastPos[2],lastPos[3],
    interpDt)
437 lastPos[1],lastPos[2],lastPos[3] = x,y,z
438
439 self:setSeparateCameraPose()
440 end
441 end
442
443 end
444
445 end
```

getSmoothed

Description

Get smooth camera position

Definition

getSmoothed(float alpha, float curX, float curY, float curZ, float targetX, float targetY, float targetZ, float lastTargetX, float lastTargetY, float lastTargetZ, float dt)

Arguments

float alpha	alpha
float curX	current x position
float curY	current y position
float curZ	current z position
float targetX	target x position
float targetY	target y position
float targetZ	target z position
float lastTargetX	last target x position
float lastTargetY	last target y position
float lastTargetZ	last target z position
float dt	time since last call in ms

Return Values

float newX	new x position
float newY	new y position
float newZ	new z position

onActivate

Description

Called on activate

Definition

onActivate()

Code

```
495 function VehicleCamera:onActivate()
496
497 self.isActivated = true
498 if (self.resetCameraOnVehicleSwitch == nil and
    g_gameSettings:getValue("resetCamera")) or self.resetCameraOnVehicleSwit
    then
499 self:resetCamera()
500 end
501 setCamera(self.cameraNode)
502 if self.shadowFocusBoxNode then
503 setShadowFocusBox(self.shadowFocusBoxNode)
504 end
505
506 if self.positionSmoothingParameter > 0 then
507 local xlook,ylook,zlook = getWorldTranslation(self.rotateNode)
508 self.lookAtPosition[1] = xlook
509 self.lookAtPosition[2] = ylook
510 self.lookAtPosition[3] = zlook
511 self.lookAtLastTargetPosition[1] = xlook
512 self.lookAtLastTargetPosition[2] = ylook
513 self.lookAtLastTargetPosition[3] = zlook
514 local x,y,z = getWorldTranslation(self.cameraPositionNode)
515 self.position[1] = x
516 self.position[2] = y
517 self.position[3] = z
518 self.lastTargetPosition[1] = x
519 self.lastTargetPosition[2] = y
520 self.lastTargetPosition[3] = z
521
522
523 local rx,ry,rz = getWorldRotation(self.rotateNode)
524
525 setRotation(self.cameraNode, rx,ry,rz)
526 setTranslation(self.cameraNode, x,y,z)
527 end
```

```
528
529 self.lastInputValues = {}
530 self.lastInputValues.upDown = 0
531 self.lastInputValues.leftRight = 0
532
533 -- activate action event callbacks
534 local _, actionEventId1 =
    g_inputBinding:registerActionEvent(InputAction.AXIS_LOOK_UPDOWN_VEHICLE,
    self, VehicleCamera.actionEventLookUpDown, false, false, true, true, nil
535 local _, actionEventId2 =
    g_inputBinding:registerActionEvent(InputAction.AXIS_LOOK_LEFTRIGHT_VEHIC
    self, VehicleCamera.actionEventLookLeftRight, false, false, true, true,
    nil)
536 g_inputBinding:setActionEventTextVisibility(actionEventId1, false)
537 g_inputBinding:setActionEventTextVisibility(actionEventId2, false)
538 end
```

onDeactivate

Description

Called on deactivate

Definition

onDeactivate()

Code

```
542 function VehicleCamera:onDeactivate()
543 self.isActivated = false
544 setShadowFocusBox(0)
545
546 -- remove action event callbacks
547 g_inputBinding:removeActionEventsByTarget(self)
548 end
```

resetCamera

Description

Reset camera to original pose

Definition

resetCamera()

Code

```
571 function VehicleCamera:resetCamera()
572 self.rotX = self.origRotX
573 self.rotY = self.origRotY
574 self.rotZ = self.origRotZ
575
576 self.transX = self.origTransX
577 self.transY = self.origTransY
```

```
578  self.transZ = self.origTransZ
579
580  local transLength = MathUtil.vector3Length(self.origTransX,
     self.origTransY, self.origTransZ)
581  self.zoom = transLength
582  self.zoomTarget = transLength
583  self.zoomLimitedTarget = -1
584
585  self:updateRotateNodeRotation()
586  setTranslation(self.cameraPositionNode, self.transX, self.transY,
     self.transZ)
587
588  if self.positionSmoothingParameter > 0 then
589  local xlook,ylook,zlook = getWorldTranslation(self.rotateNode)
590  self.lookAtPosition[1] = xlook
591  self.lookAtPosition[2] = ylook
592  self.lookAtPosition[3] = zlook
593  local x,y,z = getWorldTranslation(self.cameraPositionNode)
594  self.position[1] = x
595  self.position[2] = y
596  self.position[3] = z
597
598  self:setSeparateCameraPose()
599  end
600  end
```

updateRotateNodeRotation

Description

Update rotation node rotation

Definition

updateRotateNodeRotation()

Code

```
604  function VehicleCamera:updateRotateNodeRotation()
605  local rotY = self.rotY
606  if self.rotYSteeringRotSpeed ~= nil and self.rotYSteeringRotSpeed ~= 0 a
     self.vehicle.spec_articulatedAxis ~= nil and
     self.vehicle.spec_articulatedAxis.interpolatedRotatedTime ~= nil then
607  rotY = rotY +
     self.vehicle.spec_articulatedAxis.interpolatedRotatedTime*self.rotYSteer
608  end
609
610  if (self.useWorldXZRotation == nil and g_gameSettings:getValue("useWorlc
     or self.useWorldXZRotation then
```

```
611 local upx,upy,upz = 0,1,0
612
613 local dx,_,dz = localDirectionToWorld(getParent(self.rotateNode), 0,0,1)
614 local invLen = 1/math.sqrt(dx*dx + dz*dz)
615 dx = dx*invLen
616 dz = dz*invLen
617
618
619
620 local newDx = math.cos(self.rotX) * (math.cos(rotY)*dx + math.sin(rotY)*
621 local newDy = -math.sin(self.rotX)
622 local newDz = math.cos(self.rotX) * (-math.sin(rotY)*dx + math.cos(rotY)
623
624
625 newDx,newDy,newDz = worldDirectionToLocal(getParent(self.rotateNode),
    newDx,newDy,newDz)
626 upx,upy,upz = worldDirectionToLocal(getParent(self.rotateNode), upx,upy,
627
628 -- worst case check
629 if math.abs(MathUtil.dotProduct(newDx,newDy,newDz, upx,upy,upz)) > ( 0.9
    MathUtil.vector3Length(newDx,newDy,newDz) * MathUtil.vector3Length(upx,u
    then
630 setRotation(self.rotateNode, self.rotX, rotY, self.rotZ)
631 else
632 setDirection(self.rotateNode, newDx,newDy,newDz, upx,upy,upz)
633 end
634 else
635 setRotation(self.rotateNode, self.rotX, rotY, self.rotZ)
636 end
637 end
```

setSeparateCameraPose

Description

Set separate camera pose

Definition

setSeparateCameraPose()

Code

```
641 function VehicleCamera:setSeparateCameraPose()
642 if self.rotateNode ~= self.cameraPositionNode then
643 local dx = self.position[1] - self.lookAtPosition[1]
644 local dy = self.position[2] - self.lookAtPosition[2]
645 local dz = self.position[3] - self.lookAtPosition[3]
```

```
646
647 local upx,upy,upz = 0,1,0
648 if math.abs(dx) < 0.001 and math.abs(dz) < 0.001 then
649 upx = 0.1
650 end
651
652 setDirection(self.cameraNode, dx,dy,dz, upx,upy,upz)
653 else
654 local rx,ry,rz = getWorldRotation(self.rotateNode)
655 setRotation(self.cameraNode, rx,ry,rz)
656 end
657 setTranslation(self.cameraNode,
    self.position[1],self.position[2],self.position[3])
658 end
```

getCollisionDistance

Description

Get distance to collision

Definition

getCollisionDistance()

Return Values

boolean	hasCollision	has collision
float	collisionDistance	distance to collision
float	normalX	normal x
float	normalY	normal y
float	normalZ	normal z
float	normalDotDir	normal dot direction

Code

```
668 function VehicleCamera:getCollisionDistance()
669 if not self.isCollisionEnabled then
670 return false, nil, nil, nil, nil, nil
671 end
672
673 local raycastMask = 32+64+128+256+4096
674
675 local targetCamX, targetCamY, targetCamZ =
    localToWorld(self.rotateNode, self.transDirX*self.zoomTarget,
    self.transDirY*self.zoomTarget, self.transDirZ*self.zoomTarget)
676
677 local hasCollision = false
678 local collisionDistance = -1
679 local normalX,normalY,normalZ
680 local normalDotDir
```

```
681 for _, raycastNode in ipairs(self.raycastNodes) do
682
683 hasCollision = false
684
685 local nodeX, nodeY, nodeZ = getWorldTranslation(raycastNode)
686 local dirX, dirY, dirZ = targetCamX-nodeX, targetCamY-nodeY,
    targetCamZ-nodeZ
687 local dirLength = MathUtil.vector3Length(dirX, dirY, dirZ)
688 dirX = dirX / dirLength
689 dirY = dirY / dirLength
690 dirZ = dirZ / dirLength
691
692 local startX = nodeX
693 local startY = nodeY
694 local startZ = nodeZ
695 local currentDistance = 0
696 local minDistance = self.transMin
697
698 while true do
699 if (dirLength-currentDistance) <= 0 then
700 break
701 end
702 self.raycastDistance = 0
703 raycastClosest(startX, startY, startZ, dirX, dirY, dirZ,
    "raycastCallback", dirLength-currentDistance, self, raycastMask,
    true)
704
705 if self.raycastDistance ~= 0 then
706 currentDistance = currentDistance + self.raycastDistance+0.001
707 local ndotd = MathUtil.dotProduct(self.normalX, self.normalY,
    self.normalZ, dirX, dirY, dirZ)
708
709 local isAttachedVehicle = false
710 local object =
    g_currentMission:getNodeObject(self.raycastTransformId)
711 if object ~= nil then
712 if object ~= self.vehicle then
713 local attached1 = object.getIsAttachedTo ~= nil and
    object:getIsAttachedTo(self.vehicle)
714 local attached2 = self.vehicle.getIsAttachedTo ~= nil and
    self.vehicle:getIsAttachedTo(object)
715 isAttachedVehicle = attached1 or attached2
```

```
716
717 local mountObject = object.dynamicMountObject
718 if mountObject ~= nil and (mountObject == self.vehicle or
    mountObject:getRootVehicle() == self.vehicle) then
719 isAttachedVehicle = true
720 end
721 end
722 end
723
724 if isAttachedVehicle or object == self.vehicle then --
    isAttachedNode or isDynamicallyMounted then
725 if ndotd > 0 then
726 minDistance = math.max(minDistance, currentDistance)
727 end
728 else
729 hasCollision = true
730 -- we take the distance from the rotate node
731 if raycastNode == self.rotateNode then
732 normalX,normalY,normalZ = self.normalX, self.normalY,
    self.normalZ
733 collisionDistance = math.max(self.transMin, currentDistance)
734 normalDotDir = ndotd
735 end
736 break
737 end
738 startX = nodeX+dirX*currentDistance
739 startY = nodeY+dirY*currentDistance
740 startZ = nodeZ+dirZ*currentDistance
741 else
742 break
743 end
744 end
745 if not hasCollision then
746 break
747 end
748 end
749
750 return hasCollision, collisionDistance, normalX,normalY,normalZ,
    normalDotDir
751 end
```

VehicleMotor
Description

Class for vehicle motors

new

Description

Creating new motor

Definition

new(integer minRpm, integer maxRpm, float maxForwardSpeed, float maxBackwardSpeed, table torqueCurve, float brakeForce, float forwardGearRatios, float backwardGearRatios, float minForwardGearRatio, float maxForwardGearRatio, float minBackwardGearRatio, float maxBackwardGearRatio, integer ptoMotorRpmRatio)

Arguments

integer	minRpm	min rpm
integer	maxRpm	max rpm
float	maxForwardSpeed	max forward speed
float	maxBackwardSpeed	max backward speed
table	torqueCurve	torque curve (AnimCurve)
float	brakeForce	brake force
float	forwardGearRatios	list of gear ratios to use when driving forwards (in decreasing order)
float	backwardGearRatios	list of gear ratios to use when driving backwards (in decreasing order)
float	minForwardGearRatio	min forward gear ratio
float	maxForwardGearRatio	max forward gear ratio
float	minBackwardGearRatio	min backward gear ratio
float	maxBackwardGearRatio	max backward gear ratio
integer	ptoMotorRpmRatio	pto motor rpm ratio

Return Values

table motorInstance motor instance

Code

```
36  function VehicleMotor:new(vehicle, minRpm, maxRpm,
    maxForwardSpeed, maxBackwardSpeed, torqueCurve, brakeForce,
    forwardGearRatios, backwardGearRatios, minForwardGearRatio,
    maxForwardGearRatio, minBackwardGearRatio, maxBackwardGearRatio,
    ptoMotorRpmRatio, minSpeed)
37
38  local self = {}
39  setmetatable(self, VehicleMotor_mt)
40
41  self.vehicle = vehicle
42  self.minRpm = minRpm
43  self.maxRpm = maxRpm
44  self.minSpeed = minSpeed
45  self.maxForwardSpeed = maxForwardSpeed -- speed in m/s
46  self.maxBackwardSpeed = maxBackwardSpeed
47
48  self.maxClutchTorque = 5 -- amount of torque that can be
    transferred from motor to clutch/wheels [t m s^-2]
```

```
49
50  self.torqueCurve = torqueCurve
51  self.brakeForce = brakeForce
52
53  self.gear = 0
54  self.minGearRatio = 0
55  self.maxGearRatio = 0
56
57  self.forwardGearRatios = forwardGearRatios
58  self.backwardGearRatios = backwardGearRatios
59  self.minForwardGearRatio = minForwardGearRatio
60  self.maxForwardGearRatio = maxForwardGearRatio
61  self.minBackwardGearRatio = minBackwardGearRatio
62  self.maxBackwardGearRatio = maxBackwardGearRatio
63
64  self.manualTargetGear = nil
65  self.targetGear = 0
66  self.previousGear = 0
67  self.gearChangeTimer = -1
68  self.gearChangeTime = 250
69  self.autoGearChangeTimer = -1
70  self.autoGearChangeTime = 1000
71
72  self.lastRealMotorRpm = 0
73  self.lastMotorRpm = 0
74
75  self.rpmLimit = math.huge
76  self.speedLimit = math.huge -- Speed limit in km/h
77  self.speedLimitAcc = math.huge
78
79  self.accelerationLimit = 2 -- m s^-2
80
81  self.equalizedMotorRpm = 0
82
83  self.requiredMotorPower = 0
84
85  if self.maxForwardSpeed == nil then
86  self.maxForwardSpeed =
    self:calculatePhysicalMaximumForwardSpeed()
87  end
```

```
88  if self.maxBackwardSpeed == nil then
89  self.maxBackwardSpeed =
    self:calculatePhysicalMaximumBackwardSpeed()
90  end
91
92  self.peakMotorTorque = self.torqueCurve:getMaximum()
93
94  -- Calculate peak power. Assume we have a linear interpolation on
    the torque values
95  -- For each segment, find the maximum power (D[torque(x, i) * x]
    == 0) and take the maximum segment
96  -- D[ ((x-x0) / (x1-x0) (y1-y0) + y0) x] == 0
97  -- -> (x1 y0 - x0 y1) / (2 (y0 - y1)) if y0 != y1
98  self.peakMotorPower = 0
99  self.peakMotorPowerRotSpeed = 0
100 local numKeyFrames = #self.torqueCurve.keyframes
101 if numKeyFrames >= 2 then
102 for i=2,numKeyFrames do
103 local v0 = self.torqueCurve.keyframes[i-1]
104 local v1 = self.torqueCurve.keyframes[i]
105 local torque0 = self.torqueCurve:getFromKeyframes(v0, v0, i-1, i-
    1, 0)
106 local torque1 = self.torqueCurve:getFromKeyframes(v1, v1, i, i,
    0)
107 local rpm, torque
108 if math.abs(torque0 - torque1) > 0.0001 then
109 rpm = (v1.time * torque0 - v0.time * torque1) / (2.0 * (torque0 -
    torque1))
110 rpm = math.min(math.max(rpm, v0.time), v1.time)
111 torque = self.torqueCurve:getFromKeyframes(v0, v1, i-1, i,
    (v1.time - rpm) / (v1.time - v0.time))
112 else
113 rpm = v0.time
114 torque = torque0
115 end
116 local power = torque * rpm
117 if power > self.peakMotorPower then
118 self.peakMotorPower = power
119 self.peakMotorPowerRotSpeed = rpm
120 end
121 end
122 -- Convert from rpm to rad/s
```

```
123 self.peakMotorPower = self.peakMotorPower * math.pi/30
124 self.peakMotorPowerRotSpeed = self.peakMotorPowerRotSpeed *
    math.pi/30
125 else
126 local v = self.torqueCurve.keyframes[1]
127 local rotSpeed = v.time*math.pi/30
128 local torque = self.torqueCurve:getFromKeyframes(v, v, i, i, 0)
129 self.peakMotorPower = rotSpeed*torque
130 self.peakMotorPowerRotSpeed = rotSpeed
131 end
132
133 self.ptoMotorRpmRatio = ptoMotorRpmRatio
134
135 self.rotInertia = 0.001 -- Rotational inertia of the motor,
    mostly defined by the flywheel [t m^2]
136 self.dampingRateFullThrottle = 0.00015 -- Damping rate of the
    motor if the acceleration pedal is 1 [t m^2 s^-1]
137 self.dampingRateZeroThrottleClutchEngaged = 0.003 -- Damping rate
    of the motor if the acceleration pedal is 0 and the clutch is
    engaged [t m^2 s^-1]
138 self.dampingRateZeroThrottleClutchDisengaged = 0.001 -- Damping
    rate of the motor if the acceleration pedal is 0 and the clutch
    is disengaged [t m^2 s^-1]
139
140
141
142 -- Motor properties as read from the physics engine
143 self.gearRatio = 0
144 self.motorRotSpeed = 0 -- motor rotation speed [rad/s]
145 self.motorRotAcceleration = 0 -- motor rotation acceleration
    [rad/s^2]
146 self.motorRotAccelerationSmoothed = 0 -- motor rotation
    acceleration smoothed [rad/s^2]
147
148 self.motorAvailableTorque = 0 -- torque that was available to the
    physics simulation [kN == t m/s^2]
149 self.motorAppliedTorque = 0 -- torque that was applied (<=
    available), can be smaller when acceleration/speed is limited [kN
    == t m/s^2]
150 self.motorExternalTorque = 0 -- torque that was removed from the
    motor and was not applied to the wheels (e.g. PTO) [kN == t
    m/s^2]
151
```

```
152 self.differentialRotSpeed = 0 -- rotation speed of the main differential [rad/s]
153 self.differentialRotAcceleration = 0 -- rotation accleration of the main differential [rad/s^2]
154 self.differentialRotAccelerationSmoothed = 0 -- smoothed rotation accleration of the main differential [rad/s^2]
155
156 return self
157 end
```

setLowBrakeForce

Description

Set low brake force

Definition

setLowBrakeForce(float lowBrakeForceScale, float lowBrakeForceSpeedLimit)

Arguments

float lowBrakeForceScale low brake force scale
float lowBrakeForceSpeedLimit low brake force speed limit

Code

```
163 function VehicleMotor:setLowBrakeForce(lowBrakeForceScale, lowBrakeForceSpeedLimit)
164 self.lowBrakeForceScale = lowBrakeForceScale
165 self.lowBrakeForceSpeedLimit = lowBrakeForceSpeedLimit
166 end
```

getMaxClutchTorque

Description

Returns max clutch torque

Definition

getMaxClutchTorque()

Return Values

float maxClutchTorque max clutch torque

Code

```
171 function VehicleMotor:getMaxClutchTorque()
172 return self.maxClutchTorque
173 end
```

getRotInertia

Description

Returns rotation inertia

Definition

getRotInertia()

Return Values

float rotInertia rotation inertia

Code

```
178 function VehicleMotor:getRotInertia()
```

```
179  return self.rotInertia
180  end
```

setRotInertia

Description

Sets rotation inertia

Definition

setRotInertia(float rotInertia)

Arguments

float rotInertia rotation inertia

Code

```
185  function VehicleMotor:setRotInertia(rotInertia)
186  self.rotInertia = rotInertia
187  end
```

getDampingRateFullThrottle

Description

Returns the damping rate of the motor if the acceleration pedal is 1

Definition

getDampingRateFullThrottle()

Return Values

float dampingRate damping rate [t m^2 s^-1]

Code

```
192  function VehicleMotor:getDampingRateFullThrottle()
193  return self.dampingRateFullThrottle
194  end
```

getDampingRateZeroThrottleClutchEngaged

Description

Returns the damping rate of the motor if the acceleration pedal is 0 and the clutch is engaged

Definition

getDampingRateZeroThrottleClutchEngaged()

Return Values

float dampingRate damping rate [t m^2 s^-1]

Code

```
199  function VehicleMotor:getDampingRateZeroThrottleClutchEngaged()
200  return self.dampingRateZeroThrottleClutchEngaged
201  end
```

getDampingRateZeroThrottleClutchDisengaged

Description

Returns the damping rate of the motor if the acceleration pedal is 0 and the clutch is disengaged

Definition

getDampingRateZeroThrottleClutchDisengaged()

Return Values

float dampingRate damping rate [t m^2 s^-1]

Code

```
206 function VehicleMotor:getDampingRateZeroThrottleClutchDisengaged()
207 return self.dampingRateZeroThrottleClutchDisengaged
208 end
```

setDampingRateFullThrottle

Description

Sets the damping rate of the motor if the acceleration pedal is 1

Definition

setDampingRateFullThrottle(float dampingRate)

Arguments

float dampingRate new damping rate [t m^2 s^-1]

Code

```
213 function VehicleMotor:setDampingRateFullThrottle(dampingRate)
214 self.dampingRateFullThrottle = dampingRate
215 end
```

setDampingRateZeroThrottleClutchEngaged

Description

Sets the damping rate of the motor if the acceleration pedal is 0 and the clutch is engaged

Definition

setDampingRateZeroThrottleClutchEngaged(float dampingRate)

Arguments

float dampingRate new damping rate [t m^2 s^-1]

Code

```
220 function VehicleMotor:setDampingRateZeroThrottleClutchEngaged(dampingRate)
221 self.dampingRateZeroThrottleClutchEngaged = dampingRate
222 end
```

setDampingRateZeroThrottleClutchDisengaged

Description

Sets the damping rate of the motor if the acceleration pedal is 0 and the clutch is disengaged

Definition

setDampingRateZeroThrottleClutchDisengaged(float dampingRate)

Arguments

float dampingRate new damping rate [t m^2 s^-1]

Code

```
227 function VehicleMotor:setDampingRateZeroThrottleClutchDisengaged(dampingRate)
228 self.dampingRateZeroThrottleClutchDisengaged = dampingRate
229 end
```

setGearChangeTime

Description

Sets the time it takes change gears

Definition

setGearChangeTime(float gearChangeTime)

Arguments

float gearChangeTime gear change time [ms]

Code

```
234 function VehicleMotor:setGearChangeTime(gearChangeTime)
235 self.gearChangeTime = gearChangeTime
236 self.gearChangeTimer = math.min(self.gearChangeTimer,
    gearChangeTime)
237 end
```

setAutoGearChangeTime

Description

Sets the time that needs to pass since the last gear change until an automatic gear change is allowed

Definition

setAutoGearChangeTime(float autoGearChangeTime)

Arguments

float autoGearChangeTime automatic gear change time [ms]

Code

```
242 function VehicleMotor:setAutoGearChangeTime(autoGearChangeTime)
243 self.autoGearChangeTime = autoGearChangeTime
244 self.autoGearChangeTimer = math.min(self.autoGearChangeTimer,
    autoGearChangeTime)
245 end
```

getPeakTorque

Description

Returns max torque

Definition

getPeakTorque()

Return Values

float maxMotorTorque max motor torque

Code

```
250 function VehicleMotor:getPeakTorque()
251 return self.peakMotorTorque
252 end
```

getBrakeForce

Description

Returns brake force

Definition

getBrakeForce()

Return Values

float brakeForce brake force

Code

```
257  function VehicleMotor:getBrakeForce()
258  return self.brakeForce
259  end
```

getMinRpm

Description

Returns min rpm

Definition

getMinRpm()

Return Values

float minRpm min rpm

Code

```
264  function VehicleMotor:getMinRpm()
265  return self.minRpm
266  end
```

getMaxRpm

Description

Returns max rpm

Definition

getMaxRpm()

Return Values

float maxRpm max rpm

Code

```
271  function VehicleMotor:getMaxRpm()
272  return self.maxRpm
273  end
```

getRequiredMotorRpmRange

Description

Returns the currently required motor rpm range (e.g. defined by the activated pto)

Definition

getRequiredMotorRpmRange()

Return Values

float minRequiredRpm min required rpm

float minRequiredRpm max required rpm

Code

```
279  function VehicleMotor:getRequiredMotorRpmRange()
280  local motorPtoRpm =
     math.min(PowerConsumer.getMaxPtoRpm(self.vehicle)*self.ptoMotorRpmRatio,
     self.maxRpm)
281  if motorPtoRpm ~= 0 then
282  return motorPtoRpm, motorPtoRpm
283  end
284  return self.minRpm, self.maxRpm
```

```
285 end
```

getLastMotorRpm

Description

Returns last motor rpm damped

Definition

getLastMotorRpm()

Return Values

float lastMotorRpm last motor rpm

Code

```
290 function VehicleMotor:getLastMotorRpm()
291 return self.lastMotorRpm
292 end
```

getLastRealMotorRpm

Description

Returns last motor rpm real

Definition

getLastRealMotorRpm()

Return Values

float lastMotorRpm last motor rpm

Code

```
297 function VehicleMotor:getLastRealMotorRpm()
298 return self.lastRealMotorRpm
299 end
```

setLastRpm

Description

Sets last motor rpm

Definition

setLastRpm(float lastRpm)

Arguments

float lastRpm new last motor rpm

Code

```
304 function VehicleMotor:setLastRpm(lastRpm)
305 self.lastRealMotorRpm = lastRpm
306
307 self.lastMotorRpm = self.lastMotorRpm * 0.95 +
    self.lastRealMotorRpm * 0.05
308 end
```

getMotorAppliedTorque

Description

Returns the last applied torque to the motor

Definition

getMotorAppliedTorque()

Return Values

float appliedTorque torque [kN]

Code

```
313  function VehicleMotor:getMotorAppliedTorque()
314  return self.motorAppliedTorque
315  end
```

getMotorExternalTorque

Description

Returns the last applied external torque (torque used by external power consumers like the PTO)

Definition

getMotorExternalTorque()

Return Values

float externalTorque external torque [kN]

Code

```
320  function VehicleMotor:getMotorExternalTorque()
321  return self.motorExternalTorque
322  end
```

getMotorAvailableTorque

Description

Returns the last total available motor torque

Definition

getMotorAvailableTorque()

Return Values

float torque external torque [kN]

Code

```
327  function VehicleMotor:getMotorAvailableTorque()
328  return self.motorAvailableTorque
329  end
```

getEqualizedMotorRpm

Description

Returns equalized motor rpm

Definition

getEqualizedMotorRpm()

Return Values

float equalizedMotorRpm equalized motor rpm

Code

```
334  function VehicleMotor:getEqualizedMotorRpm()
335  return self.equalizedMotorRpm
336  end
```

setEqualizedMotorRpm

Description

Sets equalized motor rpm

Definition

setEquализedMotorRpm(float equalizedMotorRpm)

Arguments

float equalizedMotorRpm equalized motor rpm

Code

```
341  function VehicleMotor:setEqualizedMotorRpm(rpm)
342  self.equalizedMotorRpm = rpm
343  self:setLastRpm(rpm)
344  end
```

getPtoMotorRpmRatio

Description

Returns pto motor rpm ratio

Definition

getPtoMotorRpmRatio()

Return Values

float ptoMotorRpmRatio pto motor rpm ratio

Code

```
349  function VehicleMotor:getPtoMotorRpmRatio()
350  return self.ptoMotorRpmRatio
351  end
```

getNonClampedMotorRpm

Description

Returns non clamped motor rpm

Definition

getNonClampedMotorRpm()

Return Values

float nonClampedMotorRpm non clamped motor rpm

Code

```
356  function VehicleMotor:getNonClampedMotorRpm()
357  return self.motorRotSpeed * 30 / math.pi
358  end
```

getMotorRotSpeed

Description

Returns non clamped motor rpm

Definition

getMotorRotSpeed()

Return Values

float nonClampedMotorRpm non clamped motor rpm

Code

```
363  function VehicleMotor:getMotorRotSpeed()
364  return self.motorRotSpeed
365  end
```

getClutchRotSpeed

Description

Returns clutch rpm

Definition

getClutchRotSpeed()

Return Values

float clutchRpm clutch rpm

Code

```
371  function VehicleMotor:getClutchRotSpeed()
372  return self.differentialRotSpeed * self.gearRatio
373  end
```

getTorqueCurve

Description

Returns torque curve

Definition

getTorqueCurve()

Return Values

table torqueCurve torque curve

Code

```
378  function VehicleMotor:getTorqueCurve()
379  return self.torqueCurve
380  end
```

getTorque

Description

Returns torque of the motor at the current rpm with the given accelerator pedal

Definition

getTorque(float acceleration)

Arguments

float acceleration acceleration

Return Values

float torque torque

Code

```
386  function VehicleMotor:getTorque(acceleration)
387  -- Note: the torque curve is undefined outside the min/max rpm
     range. Clamping makes the curve flat at the outside range
388  local torque =
     self:getTorqueCurveValue(MathUtil.clamp(self.motorRotSpeed *
     30/math.pi, self.minRpm, self.maxRpm))
389  torque = torque * math.abs(acceleration)
390  return torque
391  end
```

getTorqueCurveValue

Description

Returns torque of the motor at the given rpm

Definition

getTorqueCurveValue(float rpm)

Arguments

float rpm rpm

Return Values

float torque torque

Code

```
397 function VehicleMotor:getTorqueCurveValue(rpm)
398 local damage = 1 - (self.vehicle:getVehicleDamage() * VehicleMotor.DAMAGE_TORQUE_REDUCTION)
399 return self:getTorqueCurve():get(rpm) * damage
400 end
```

getMaximumForwardSpeed

Description

Returns maximum forward speed

Definition

getMaximumForwardSpeed()

Return Values

float maxForwardSpeed maximum forward speed

Code

```
416 function VehicleMotor:getMaximumForwardSpeed()
417 return self.maxForwardSpeed
418 end
```

getMaximumBackwardSpeed

Description

Returns maximum backward speed

Definition

getMaximumBackwardSpeed()

Return Values

float maxBackwardSpeed maximum backward speed

Code

```
423 function VehicleMotor:getMaximumBackwardSpeed()
424 return self.maxBackwardSpeed
425 end
```

calculatePhysicalMaximumForwardSpeed

Description

Returns physical maximum forward speed

Definition

calculatePhysicalMaximumForwardSpeed()

Return Values

float physicalMaxForwardSpeed physical maximum forward speed

Code

```
430 function VehicleMotor:calculatePhysicalMaximumForwardSpeed()
431 return VehicleMotor.calculatePhysicalMaximumSpeed(self.minForwardGearRatio, self.forwardGearRatios, self.maxRpm)
432 end
```

calculatePhysicalMaximumBackwardSpeed

Description

Returns physical maximum backward speed

Definition

calculatePhysicalMaximumBackwardSpeed()

Return Values

float physicalMaxBackwardSpeed physical maximum backward speed

Code

```
437 function VehicleMotor:calculatePhysicalMaximumBackwardSpeed()
438 return VehicleMotor.calculatePhysicalMaximumSpeed(self.minBackwardGearRatio, self.backwardGearRatios, self.maxRpm)
439 end
```

calculatePhysicalMaximumSpeed

Description

Returns physical maximum speed

Definition

calculatePhysicalMaximumSpeed(float minGearRatio, table gearRatios, integer maxRpm)

Arguments

float	minGearRatio	min gear ratio
table	gearRatios	gear ratios
integer	maxRpm	max rpm

Return Values

float physicalMaxSpeed physical maximum speed

Code

```
447 function VehicleMotor.calculatePhysicalMaximumSpeed(minGearRatio, gearRatios, maxRpm)
448 local minRatio
449 if minGearRatio ~= nil then
450 minRatio = minGearRatio
451 else
452 minRatio = math.huge
453 for _, ratio in pairs(gearRatios) do
454 minRatio = math.min(minRatio, ratio)
455 end
456 end
457 return maxRpm * math.pi / (30 * minRatio)
458 end
```

update

Description

Update the state of the motor (sync with physics simulation)

Definition

update(float dt)

Arguments

float dt time since last call in ms

Code

```
463 function VehicleMotor:update(dt)
464 local vehicle = self.vehicle
465 if next(vehicle.spec_motorized.differentials) ~= nil and
    vehicle.spec_motorized.motorizedNode ~= nil then
466 -- Only update the physics values if a physics simulation was performed
467 if g_physicsDtNonInterpolated > 0.0 then
468 local lastMotorRotSpeed = self.motorRotSpeed;
469 local lastDiffRotSpeed = self.differentialRotSpeed
470 self.motorRotSpeed, self.differentialRotSpeed, self.gearRatio =
    getMotorRotationSpeed(vehicle.spec_motorized.motorizedNode)
471
472 self.motorAvailableTorque, self.motorAppliedTorque,
    self.motorExternalTorque =
    getMotorTorque(vehicle.spec_motorized.motorizedNode)
473
474
475 local motorRotAcceleration = (self.motorRotSpeed - lastMotorRotSpeed) /
    (g_physicsDtNonInterpolated*0.001)
476 self.motorRotAcceleration = motorRotAcceleration
477 self.motorRotAccelerationSmoothed = 0.8 *
    self.motorRotAccelerationSmoothed + 0.2 * motorRotAcceleration
478
479 local diffRotAcc = (self.differentialRotSpeed - lastDiffRotSpeed) /
    (g_physicsDtNonInterpolated*0.001)
480 self.differentialRotAcceleration = diffRotAcc
481 self.differentialRotAccelerationSmoothed = 0.8 *
    self.differentialRotAccelerationSmoothed + 0.2 * diffRotAcc
482
483 --print(string.format("update rpms: %.2f %.2f acc: %.2f",
    self.motorRotSpeed*30/math.pi,
    self.differentialRotSpeed*self.gearRatio*30/math.pi,
    motorRotAcceleration))
484 end
485
486 self.requiredMotorPower = math.huge
487
```

```
488 else
489 local _, gearRatio = self:getMinMaxGearRatio()
490 self.differentialRotSpeed =
    WheelsUtil.computeDifferentialRotSpeedNonMotor(vehicle)
491 self.motorRotSpeed = math.max(self.differentialRotSpeed * gearRatio, 0)
492 self.gearRatio = gearRatio
493 end
494
495 -- the clamped motor rpm always is higher-equal than the required rpm by
    the pto
496 --local ptoRpm =
    math.min(PowerConsumer.getMaxPtoRpm(self.vehicle)*self.ptoMotorRpmRatio,
    self.maxRpm)
497 -- smoothing for raise/fall of ptoRpm
498 if self.lastPtoRpm == nil then
499 self.lastPtoRpm = self.minRpm
500 end
501 local ptoRpm =
    PowerConsumer.getMaxPtoRpm(self.vehicle)*self.ptoMotorRpmRatio
502 if ptoRpm > self.lastPtoRpm then
503 self.lastPtoRpm = math.min(ptoRpm, self.lastPtoRpm +
    self.maxRpm*dt/2000)
504 elseif ptoRpm < self.lastPtoRpm then
505 self.lastPtoRpm = math.max(self.minRpm, self.lastPtoRpm -
    self.maxRpm*dt/1000)
506 end
507 local ptoRpm = math.min(self.lastPtoRpm, self.maxRpm)
508
509 local clampedMotorRpm = math.max(self.motorRotSpeed*30/math.pi, ptoRpm,
    self.minRpm)
510
511 self:setLastRpm(clampedMotorRpm)
512
513 self.equalizedMotorRpm = clampedMotorRpm
514 end
```

getBestGearRatio

Description

Returns best gear ratio

Definition

getBestGearRatio(float wheelSpeedRpm, float minRatio, float maxRatio, float accSafeMotorRpm, float requiredMotorPower, float requiredMotorRpm)

Arguments

float wheelSpeedRpm	wheel speed rpm

float minRatio	min ratio
float maxRatio	max ratio
float accSafeMotorRpm	acc save motor rpm
float requiredMotorPower	the required motor power [kW] (can be bigger than what the motor can actually achieve)
float requiredMotorRpm	fixed motor rpm to be used (if not 0)

Return Values

float bestGearRatio best gear ratio

Code

```
526 function VehicleMotor:getBestGearRatio(wheelSpeedRpm, minRatio, maxRatio, accSafeMotorRpm, requiredMotorPower, requiredMotorRpm)
527
528 if requiredMotorRpm ~= 0 then
529 local gearRatio = math.max(requiredMotorRpm-accSafeMotorRpm, requiredMotorRpm*0.8) / math.max(wheelSpeedRpm, 0.001)
530 gearRatio = MathUtil.clamp(gearRatio, minRatio, maxRatio)
531 return gearRatio
532 end
533
534 -- Use a minimum wheel rpm to avoid that gearRatio is ignored
535 wheelSpeedRpm = math.max(wheelSpeedRpm, 0.0001)
536
537 local bestMotorPower = 0
538 local bestGearRatio = minRatio
539 --local bestRPM = 0
540 -- TODO make this more efficient
541 for gearRatio = minRatio, maxRatio, 0.5 do
542 local motorRpm = wheelSpeedRpm * gearRatio
543 if motorRpm > self.maxRpm - accSafeMotorRpm then
544 break
545 end
546 local motorPower = self:getTorqueCurveValue(math.max(motorRpm, self.minRpm)) * motorRpm *math.pi/30
547 if motorPower > bestMotorPower then
548 bestMotorPower = motorPower
549 bestGearRatio = gearRatio
550 --bestRPM = motorRpm
551 end
552
553 if motorPower >= requiredMotorPower then
554 break
555 end
```

```
556 end
557 --print(string.format("Selected best gear: %f, %.2fkW rpm %.2f
    wheel %.2f", bestGearRatio, bestMotorPower, bestRPM,
    wheelSpeedRpm,))
558
559 return bestGearRatio
560 end
```

getBestGear

Description

Returns best gear

Definition

getBestGear(float acceleration, float wheelSpeedRpm, float accSafeMotorRpm, float requiredMotorPower, float requiredMotorRpm)

Arguments

float acceleration	acceleration
float wheelSpeedRpm	wheel speed rpm
float accSafeMotorRpm	acc save motor rpm
float requiredMotorPower	required wheel torque
float requiredMotorRpm	required motor rpm

Return Values

float bestGear	best gear
float gearRatio	gear ratio

Code

```
571 function VehicleMotor:getBestGear(acceleration, wheelSpeedRpm,
    accSafeMotorRpm, requiredMotorPower, requiredMotorRpm)
572 if math.abs(acceleration) < 0.001 then
573 acceleration = 1
574 if wheelSpeedRpm < 0 then
575 acceleration = -1
576 end
577 end
578 if acceleration > 0 then
579 if self.minForwardGearRatio ~= nil then
580 local wheelSpeedRpm = math.max(wheelSpeedRpm, 0)
581 local bestGearRatio = self:getBestGearRatio(wheelSpeedRpm,
    self.minForwardGearRatio, self.maxForwardGearRatio,
    accSafeMotorRpm, requiredMotorPower, requiredMotorRpm)
582 return 1, bestGearRatio
583 else
584 return 1, self.forwardGearRatios[1]
585 end
586 else
587 if self.minBackwardGearRatio ~= nil then
```

```
588  local wheelSpeedRpm = math.max(-wheelSpeedRpm, 0)
589  local bestGearRatio = self:getBestGearRatio(wheelSpeedRpm,
     self.minBackwardGearRatio, self.maxBackwardGearRatio,
     accSafeMotorRpm, requiredMotorPower, requiredMotorRpm)
590  return -1, -bestGearRatio
591  else
592  return -1, -self.backwardGearRatios[1]
593  end
594  end
595  end
```

updateGear

Description

Update gear

Definition

updateGear(float acceleratorPedal)

Arguments

float acceleratorPedal acceleratorPedal

Return Values

float adjustedAcceleratorPedal	the adjusted accelerator pedal for the current gear situation (e.g. 0 while switching gears)

Code

```
818  function VehicleMotor:updateGear(acceleratorPedal, dt)
819  local adjAcceleratorPedal = acceleratorPedal
820  if self.gearChangeTimer >= 0 then
821  self.gearChangeTimer = self.gearChangeTimer - dt;
822  if self.gearChangeTimer < 0 then
823  if self.targetGear > 0 then
824  -- give the logic another chance to choose a better gear at the
     time when the gear change should happen
825  self.gear = self:findGearChangeTargetGear(self.targetGear,
     self.previousGear, self.forwardGearRatios, 1, acceleratorPedal,
     dt)
826  self.minGearRatio = self.forwardGearRatios[self.gear]
827  else
828  self.gear = -self:findGearChangeTargetGear(-self.targetGear, -
     self.previousGear, self.backwardGearRatios, -1, acceleratorPedal,
     dt)
829  self.minGearRatio = -self.backwardGearRatios[-self.gear]
830  end
831  self.maxGearRatio = self.minGearRatio
832  end
833  adjAcceleratorPedal = 0
834  else
```

```
local gearSign = 0
if acceleratorPedal > 0 then
if self.minForwardGearRatio ~= nil then
self.minGearRatio = self.minForwardGearRatio
self.maxGearRatio = self.maxForwardGearRatio
else
gearSign = 1
end
elseif acceleratorPedal < 0 then
if self.minBackwardGearRatio ~= nil then
self.minGearRatio = -self.minBackwardGearRatio
self.maxGearRatio = -self.maxBackwardGearRatio
else
gearSign = -1
end
else
if self.maxGearRatio > 0 then
if self.minForwardGearRatio == nil then
gearSign = 1
end
elseif self.maxGearRatio < 0 then
if self.minBackwardGearRatio == nil then
gearSign = -1
end
end
end

self.autoGearChangeTimer = self.autoGearChangeTimer - dt
local newGear = self.gear
if g_manualGearShift then
if self.manualTargetGear ~= nil then
newGear = self.manualTargetGear
self.manualTargetGear = nil
end
else
if gearSign > 0 then
if self.gear <= 0 then
newGear = 1
else
if self.autoGearChangeTimer <= 0 then
```

```
875  newGear = self:findGearChangeTargetGearPrediction(self.gear,
     self.forwardGearRatios, 1, self.autoGearChangeTimer,
     acceleratorPedal, dt)
876  end
877  newGear = math.min(math.max(newGear, 1), #self.forwardGearRatios)
878  end
879  elseif gearSign < 0 then
880  if self.gear >= 0 then
881  newGear = -1
882  else
883  if self.autoGearChangeTimer <= 0 then
884  newGear = -self:findGearChangeTargetGearPrediction(-self.gear,
     self.backwardGearRatios, -1, self.autoGearChangeTimer,
     acceleratorPedal, dt)
885  end
886  newGear = math.max(math.min(newGear, -1), -
     #self.backwardGearRatios)
887  end
888  end
889  end
890  if newGear ~= self.gear then
891  self.targetGear = newGear
892  self.previousGear = self.gear
893  self.gear = 0
894  self.minGearRatio = 0
895  self.maxGearRatio = 0
896  self.autoGearChangeTimer = self.autoGearChangeTime
897  self.gearChangeTimer = self.gearChangeTime
898  adjAcceleratorPedal = 0
899  end
900  end
901  return adjAcceleratorPedal
902  end
```

getMinMaxGearRatio

Description

Returns currently selected minimum and maximum gear ratio
Gear ratios for driving backwards are negative. Min/max always refers to the absolute value
For regular gear box transmission, the minimum and maximum gear ratios are identical

Definition

getMinMaxGearRatio()

Return Values

float minGearRatio minimum gear ratio

float maxGearRatio maximum gear ratio

Code

```
910  function VehicleMotor:getMinMaxGearRatio()
911  return self.minGearRatio, self.maxGearRatio
912  end
```

getCurMaxRpm

Description

Returns current max rpm

Definition

getCurMaxRpm()

Return Values

integer maxRpm current max rpm

Code

```
921  function VehicleMotor:getCurMaxRpm()
922  local maxRpm = self.maxRpm
923
924  local gearRatio = self:getGearRatio()
925  if gearRatio ~= 0 then
926  --local speedLimit = self.speedLimit * 0.277778
927  local speedLimit = math.min(self.speedLimit,
     math.max(self.speedLimitAcc, self.vehicle.lastSpeedReal*3600)) *
     0.277778
928  if gearRatio > 0 then
929  speedLimit = math.min(speedLimit, self.maxForwardSpeed)
930  else
931  speedLimit = math.min(speedLimit, self.maxBackwardSpeed)
932  end
933
934  maxRpm = math.min(maxRpm, speedLimit * 30 / math.pi *
     math.abs(gearRatio))
935  end
936
937  maxRpm = math.min(maxRpm, self.rpmLimit)
938  return maxRpm
939  end
```

setSpeedLimit

Description

Sets speed limit

Definition

setSpeedLimit(float limit)

Arguments

float limit new limit

Code

```
944  function VehicleMotor:setSpeedLimit(limit)
945  self.speedLimit = math.max(limit, self.minSpeed)
946  end
```

setRpmLimit

Description

Sets rpm limit

Definition

setRpmLimit(float limit)

Arguments

float limit new limit

Code

```
963  function VehicleMotor:setRpmLimit(rpmLimit)
964  self.rpmLimit = rpmLimit
965  end
```

VehiclePlacementCallback

Description

Placement callback

new

Description

Create instance of class

Definition

new()

Code

```
11  function VehiclePlacementCallback:new()
12  local instance = {}
13  setmetatable(instance, VehiclePlacementCallback_mt)
14
15  return instance
16  end
```

callback

Description

Raycast callback

Definition

callback(integer transformId, float x, float y, float z, float distance)

Arguments

integer transformId id raycasted object

float	x	x raycast position
float	y	y raycast position
float	z	z raycast position
float	distance	distance to raycast position

Return Values

boolean continue continue

Code

```
26  function VehiclePlacementCallback:callback(transformName, x, y, z, distance)
27  self.raycastHitName = transformName
28  self.x = x
29  self.y = y
30  self.z = z
31  self.distance = distance
32
33  return true
34  end
```

VehicleTypeManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
17  function VehicleTypeManager:new(customMt)
18  local self = AbstractManager:new(customMt or VehicleTypeManager_mt)
19
20  return self
21  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
25  function VehicleTypeManager:initDataStructures()
26  self.vehicleTypes = {}
27  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
32  function VehicleTypeManager:loadMapData()
33  VehicleTypeManager:superClass().loadMapData(self)
34
35  local xmlFile = loadXMLFile("VehicleTypesXML",
    "dataS/vehicleTypes.xml")
36
37  local i = 0
38  while true do
39  local key = string.format("vehicleTypes.type(%d)", i)
40  if not hasXMLProperty(xmlFile, key) then
41  break
42  end
43
44  g_deferredLoadingManager:addSubtask(function()
45  self:loadVehicleTypeFromXML(xmlFile, key, nil, nil, nil)
46  end)
47
48  i = i + 1
49  end
50
51  g_deferredLoadingManager:addSubtask(function()
52  delete(xmlFile)
53  end)
54
55  g_deferredLoadingManager:addSubtask(function()
56  print(" Loaded vehicle types")
57  end)
58
59  return true
60  end
```

addVehicleType

Description

Adds a new vehicleType

Definition

addVehicleType(string typeName, string className, string filename, table specializationNames, string customEnvironment)

Arguments

string typeName	vehicle type name
string className	classname

string	filename	filename
table	specializationNames	list of specializations
string	customEnvironment	a custom environment

Return Values

boolean success true if added else false

WheelsUtil

Description

registerTireType

Description

Register new tire type

Definition

registerTireType(string name, table frictionCoeffs, table frictionCoeffsWer)

Arguments

string	name	name of new tire type
table	frictionCoeffs	friction coeffs
table	frictionCoeffsWer	friction coeffs wet

Code

```
30  function WheelsUtil.registerTireType(name, frictionCoeffs, frictionCoeffsWet)
31  local tireType = WheelsUtil.getTireType(name)
32  if tireType ~= nil then
33  print("Warning: Adding duplicate tire type '"..name.."'")
34  return
35  end
36
37  local function getNoNilCoeffs(frictionCoeffs)
38  local localCoeffs = {}
39  if frictionCoeffs[1] == nil then
40  localCoeffs[1] = 1.15
41  for i=2,WheelsUtil.NUM_GROUNDS do
42  if frictionCoeffs[i] ~= nil then
43  localCoeffs[1] = frictionCoeffs[i]
44  break
45  end
46  end
47  else
48  localCoeffs[1] = frictionCoeffs[1]
49  end
50  for i=2,WheelsUtil.NUM_GROUNDS do
51  localCoeffs[i] = frictionCoeffs[i] or frictionCoeffs[i-1]
52  end
53  return localCoeffs
```

```
54  end
55
56  local tireType = {}
57  tireType.name = name
58  tireType.frictionCoeffs = getNoNilCoeffs(frictionCoeffs)
59  tireType.frictionCoeffsWet = getNoNilCoeffs(frictionCoeffsWet or
    frictionCoeffs)
60  table.insert(WheelsUtil.tireTypes, tireType)
61  end
```

getTireType

Description

Returns tire type index

Definition

getTireType(string name)

Arguments

string name name of tire type

Return Values

integer i index of tire type

Code

```
67  function WheelsUtil.getTireType(name)
68  for i, t in pairs(WheelsUtil.tireTypes) do
69  if t.name == name then
70  return i
71  end
72  end
73  return nil
74  end
```

updateWheelsPhysics

Description

Updates wheel physics

Definition

updateWheelsPhysics(float dt, float currentSpeed, float acceleration, boolean doHandbrake, boolean stopAndGoBraking)

Arguments

float	dt	time since last call in ms
float	currentSpeed	signed current speed (m/ms)
float	acceleration	target acceleration [-1,1]
boolean	doHandbrake	do handbrake
boolean	stopAndGoBraking	if false, the acceleration needs to be 0 before a change of direction is allowed

Code

```
194  function WheelsUtil.updateWheelsPhysics(self, dt, currentSpeed,
     acceleration, doHandbrake, stopAndGoBraking)
```

```
195 --print("function
    WheelsUtil.updateWheelsPhysics("..tostring(self)..",
    "..tostring(dt)..", "..tostring(currentSpeed)..",
    "..tostring(acceleration)..", "..tostring(doHandbrake)..",
    "..tostring(stopAndGoBraking))
196
197 local acceleratorPedal = 0
198 local brakePedal = 0
199
200 local reverserDirection = 1
201 if self.spec_drivable ~= nil then
202 reverserDirection = self.spec_drivable.reverserDirection
203 acceleration = acceleration * reverserDirection
204 end
205
206 local motor = self.spec_motorized.motor
207
208 local absCurrentSpeed = math.abs(currentSpeed)
209 local accSign = MathUtil.sign(acceleration)
210
211 self.nextMovingDirection =
    Utils.getNoNil(self.nextMovingDirection, 0)
212
213 local automaticBrake = false
214
215 if math.abs(acceleration) < 0.001 then
216 automaticBrake = true
217
218 -- Non-stop&go only allows change of direction if the vehicle
    speed is smaller than 1km/h or the direction has already changed
    (e.g. because the brakes are not hard enough)
219 if stopAndGoBraking or currentSpeed * self.nextMovingDirection <
    0.0003 then
220 self.nextMovingDirection = 0
221 end
222 else
223 -- Disable the known moving direction if the vehicle is driving
    more than 5km/h (0.0014 * 3600 = 5.04km/h) in the opposite
    direction
224 if self.nextMovingDirection * currentSpeed < -0.0014 then
225 self.nextMovingDirection = 0
226 end
227
```

```
228 -- Continue accelerating if we want to go in the same direction
229 -- or if the vehicle is only moving slowly in the wrong direction
    (0.0003 * 3600 = 1.08 km/h) and we are allowed to change
    direction
230 if accSign == self.nextMovingDirection or (currentSpeed * accSign
    > -0.0003 and (stopAndGoBraking or self.nextMovingDirection ==
    0)) then
231 acceleratorPedal = acceleration
232 brakePedal = 0
233 self.nextMovingDirection = accSign
234 else
235 acceleratorPedal = 0
236 brakePedal = math.abs(acceleration)
237 if stopAndGoBraking then
238 self.nextMovingDirection = accSign
239 end
240 end
241 end
242
243 if automaticBrake then
244 acceleratorPedal = 0
245 end
246
247 acceleratorPedal = motor:updateGear(acceleratorPedal, dt)
248
249 if motor.gear == 0 and motor.targetGear ~= 0 then
250 -- brake automatically if the vehicle is rolling backwards while
    shifting
251 if currentSpeed * MathUtil.sign(motor.targetGear) < 0 then
252 automaticBrake = true
253 end
254 end
255
256 if automaticBrake then
257 local isSlow = absCurrentSpeed < motor.lowBrakeForceSpeedLimit
258 local isArticulatedSteering = self.spec_articulatedAxis ~= nil
    and self.spec_articulatedAxis.componentJoint ~= nil and
    math.abs(self.rotatedTime) > 0.01
259
260 if (isSlow or doHandbrake) and not isArticulatedSteering then
261 brakePedal = 1
262 else
```

```
263 -- interpolate between lowBrakeForce and 1 if speed is below 3.6
    km/h
264 local factor = math.min(absCurrentSpeed / 0.001, 1)
265 brakePedal = MathUtil.lerp(1, motor.lowBrakeForceScale, factor)
266 end
267 end
268
269 -- ToDo: move to Lights ?!
270 if self.spec_lights ~= nil then
271 if self.setBrakeLightsVisibility ~= nil then
272 self:setBrakeLightsVisibility(not automaticBrake and
    math.abs(brakePedal) > 0)
273 end
274
275 if self.setReverseLightsVisibility ~= nil then
276 self:setReverseLightsVisibility((currentSpeed < -0.0006 or
    acceleratorPedal < 0) and reverserDirection == 1)
277 end
278 end
279
280 acceleratorPedal, brakePedal =
    WheelsUtil.getSmoothedAcceleratorAndBrakePedals(self,
    acceleratorPedal, brakePedal, dt)
281
282 --active braking if over the speed limit
283 local overSpeedLimit = self:getLastSpeed() -
    motor:getSpeedLimit()
284 if overSpeedLimit > 0 then
285 brakePedal = math.max(math.min(overSpeedLimit/5, 1), brakePedal)
286 acceleratorPedal = math.min(0, acceleratorPedal)
287 end
288
289 if next(self.spec_motorized.differentials) ~= nil and
    self.spec_motorized.motorizedNode ~= nil then
290
291 local absAcceleratorPedal = math.abs(acceleratorPedal)
292 local minGearRatio, maxGearRatio = motor:getMinMaxGearRatio()
293
294 local maxSpeed;
295 if maxGearRatio >= 0 then
296 maxSpeed = motor:getMaximumForwardSpeed()
297 else
```

```
298 maxSpeed = motor:getMaximumBackwardSpeed()
299 end
300
301 local acceleratorPedalControlsSpeed = false
302 if acceleratorPedalControlsSpeed then
303 maxSpeed = maxSpeed * absAcceleratorPedal
304 if absAcceleratorPedal > 0.001 then
305 absAcceleratorPedal = 1
306 end
307 end
308 maxSpeed = math.min(maxSpeed, motor:getSpeedLimit() / 3.6)
309 local maxAcceleration = motor:getAccelerationLimit()
310 local minMotorRpm, maxMotorRpm = motor:getRequiredMotorRpmRange()
311
312 local neededPtoTorque =
    PowerConsumer.getTotalConsumedPtoTorque(self) /
    motor:getPtoMotorRpmRatio();
313
314 --print(string.format("set vehicle props: accPed=%.1f speed=%.1f
    gearRatio=[%.1f %.1f] rpm=[%.1f %.1f]", absAcceleratorPedal,
    maxSpeed, minGearRatio, maxGearRatio, minMotorRpm, maxMotorRpm))
315 controlVehicle(self.spec_motorized.motorizedNode,
    absAcceleratorPedal, maxSpeed, maxAcceleration,
    minMotorRpm*math.pi/30, maxMotorRpm*math.pi/30, minGearRatio,
    maxGearRatio, motor:getMaxClutchTorque(), neededPtoTorque)
316 end
317
318 self:brake(brakePedal)
319 end
```

updateWheelPhysics

Description

Update wheel physics

Definition

updateWheelPhysics(table wheel, boolean doHandbrake, float brakePedal, float dt)

Arguments

table	wheel	wheel
boolean	doHandbrake	doHandbrake
float	brakePedal	brake pedal
float	dt	dt

Code

```
328 function WheelsUtil.updateWheelPhysics(self, wheel, brakePedal,
    dt)
329 WheelsUtil.updateWheelSteeringAngle(self, wheel, dt)
```

```
330
331 if self.isServer and self.isAddedToPhysics then
332 local brakeForce = self:getBrakeForce() * brakePedal
333 setWheelShapeProps(wheel.node, wheel.wheelShape, wheel.torque,
    brakeForce*wheel.brakeFactor, wheel.steeringAngle,
    wheel.rotationDamping)
334 end
335 end
```

updateWheelHasGroundContact

Description

Update check if wheel has ground contact

Definition

updateWheelHasGroundContact(table wheel)

Arguments

table wheel wheel

Code

```
340 function WheelsUtil.updateWheelHasGroundContact(wheel)
341
342 local x,_,_ = getWheelShapeContactPoint(wheel.node,
    wheel.wheelShape)
343 wheel.hasGroundContact = x~=nil
344 --return wheel.hasGroundContact
345 end
```

updateWheelSteeringAngle

Description

Update wheel steering angle

Definition

updateWheelSteeringAngle(table wheel, float dt)

Arguments

table wheel wheel

float dt time since last call in ms

Code

```
351 function WheelsUtil.updateWheelSteeringAngle(self, wheel, dt)
352
353 local steeringAngle = wheel.steeringAngle
354 local rotatedTime = self.rotatedTime
355
356 if wheel.steeringAxleScale ~= nil and wheel.steeringAxleScale ~=
    0 then
357 local steeringAxleAngle = 0
358 if self.spec_attachable ~= nil then
359 steeringAxleAngle = self.spec_attachable.steeringAxleAngle
```

```
360 end
361 steeringAngle = MathUtil.clamp(steeringAxleAngle *
    wheel.steeringAxleScale, wheel.steeringAxleRotMin,
    wheel.steeringAxleRotMax)
362 elseif wheel.versatileYRot and
    self:getIsVersatileYRotActive(wheel) then
363 if self.isServer then
364 if wheel.forceVersatility or wheel.hasGroundContact then
365 steeringAngle = Utils.getVersatileRotation(wheel.repr,
    wheel.node, dt, wheel.positionX, wheel.positionY,
    wheel.positionZ, wheel.steeringAngle, wheel.rotMin, wheel.rotMax)
366 end
367 end
368 elseif wheel.rotSpeed ~= nil and wheel.rotMax ~= nil and
    wheel.rotMin ~= nil then
369 if rotatedTime > 0 or wheel.rotSpeedNeg == nil then
370 steeringAngle = rotatedTime * wheel.rotSpeed
371 else
372 steeringAngle = rotatedTime * wheel.rotSpeedNeg
373 end
374 if steeringAngle > wheel.rotMax then
375 steeringAngle = wheel.rotMax
376 elseif steeringAngle < wheel.rotMin then
377 steeringAngle = wheel.rotMin
378 end
379 if self.updateSteeringAngle ~= nil then
380 steeringAngle = self:updateSteeringAngle(wheel, dt,
    steeringAngle)
381 end
382 end
383 wheel.steeringAngle = steeringAngle
384 end
```

computeDifferentialRotSpeedNonMotor

Description

Compute differential rot speed from properties of vehicle other than the motor, e.g. rot speed of wheels or linear speed of vehicle

Definition

computeDifferentialRotSpeedNonMotor()

Return Values

float diffRotSpeed rot speed [rad/sec]

Code

```
389 function WheelsUtil.computeDifferentialRotSpeedNonMotor(self)
```

```
390 if self.isServer and self.spec_wheels ~= nil and
    #self.spec_wheels.wheels ~= 0 then
391 local wheelSpeed = 0
392 local numWheels = 0
393 for _, wheel in pairs(self.spec_wheels.wheels) do
394 local axleSpeed = getWheelShapeAxleSpeed(wheel.node,
    wheel.wheelShape) -- rad/sec
395 if wheel.hasGroundContact then
396 wheelSpeed = wheelSpeed + axleSpeed * wheel.radius
397 numWheels = numWheels+1
398 end
399 end
400
401 if numWheels > 0 then
402 return wheelSpeed/numWheels
403 end
404 return 0
405 else
406 -- v = w*r => w = v/r
407 -- differentials have embeded gear so that r can be considered 1
408 return self.lastSpeedReal*1000
409 end
410 end
```

updateVisualWheel

Description

Update wheel graphics

Definition

updateVisualWheel(table wheel, float x, float y, float z, float x, float suspensionLength)

Arguments

table	wheel	wheel
float	x	x position
float	y	y position
float	z	z position
float	x	x drive rotation
float	suspensionLength	length of suspension

Code

```
472 function WheelsUtil.updateVisualWheel(self, wheel, x, y, z,
    xDrive, suspensionLength)
473 local changed = false
474
475 local steeringAngle = wheel.steeringAngle
476 if not wheel.showSteeringAngle then
```

```
477 steeringAngle = 0
478 end
479
480 local _, oldY, _ = getRotation(wheel.repr)
481 local dirX, dirY, dirZ = localDirectionToLocal(wheel.repr,
    getParent(wheel.repr), 0, -1, 0)
482 if math.abs(steeringAngle-oldY) >
    WheelsUtil.STEERING_ANGLE_THRESHOLD then
483 setRotation(wheel.repr, 0, steeringAngle, 0)
484 changed = true
485 end
486
487 local oldX, _, _ = getRotation(wheel.driveNode)
488 if math.abs(xDrive-oldX) > WheelsUtil.STEERING_ANGLE_THRESHOLD
    then
489 setRotation(wheel.driveNode, xDrive, 0, 0)
490 changed = true
491 end
492
493 if wheel.wheelTire ~= nil then
494 local x, y, z, _ = getShaderParameter(wheel.wheelTire,
    "morphPosition")
495 local deformation = MathUtil.clamp((wheel.deltaY+0.04-
    suspensionLength)*0.7, 0, wheel.maxDeformation)
496 if math.abs(deformation - wheel.deformation) > 0.01 then
497 wheel.deformation = deformation
498 setShaderParameter(wheel.wheelTire, "morphPosition", x, y, z,
    deformation, false)
499
500 if wheel.additionalWheels ~= nil then
501 for _, additionalWheel in pairs(wheel.additionalWheels) do
502 local x, y, z, _ = getShaderParameter(additionalWheel.wheelTire,
    "morphPosition")
503 setShaderParameter(additionalWheel.wheelTire, "morphPosition", x,
    y, z, deformation, false)
504 end
505 end
506 changed = true
507 end
508
509 suspensionLength = suspensionLength+deformation
510 end
```

```
511
512 suspensionLength = suspensionLength - wheel.deltaY
513
514 if math.abs(wheel.lastMovement-suspensionLength) >
    WheelsUtil.SUSPENSION_THRESHOLD then
515 local transRatio = wheel.transRatio
516 local movement = suspensionLength * transRatio
517 setTranslation(wheel.repr, wheel.startPositionX + dirX*movement,
    wheel.startPositionY + dirY*movement, wheel.startPositionZ +
    dirZ*movement)
518 changed = true
519 if transRatio < 1 then
520 movement = suspensionLength*(1-transRatio)
521 setTranslation(wheel.driveNode, wheel.driveNodeStartPosX +
    dirX*movement, wheel.driveNodeStartPosY + dirY*movement,
    wheel.driveNodeStartPosZ + dirZ*movement)
522 end
523
524 wheel.lastMovement = suspensionLength
525 end
526
527 if wheel.steeringNode ~= nil then
528 local refAngle = wheel.steeringNodeMaxRot
529 local refTrans = wheel.steeringNodeMaxTransX
530 local refRot = wheel.steeringNodeMaxRotY
531 if steeringAngle < 0 then
532 refAngle = wheel.steeringNodeMinRot
533 refTrans = wheel.steeringNodeMinTransX
534 refRot = wheel.steeringNodeMinRotY
535 end
536 local steering = 0
537 if refAngle ~= 0 then
538 steering = steeringAngle / refAngle
539 end
540
541 if wheel.steeringNodeMinTransX ~= nil then
542 local x,y,z = getTranslation(wheel.steeringNode)
543 x = refTrans * steering
544 setTranslation(wheel.steeringNode, x, y, z)
545 end
546 if wheel.steeringNodeMinRotY ~= nil then
```

```
547 local rotX,rotY,rotZ = getRotation(wheel.steeringNode)
548 rotY = refRot * steering
549 setRotation(wheel.steeringNode, rotX, rotY, rotZ)
550 end
551 end
552
553 if wheel.fenderNode ~= nil then
554 local angleDif = 0
555 if steeringAngle > wheel.fenderRotMax then
556 angleDif = wheel.fenderRotMax - steeringAngle
557 elseif steeringAngle < wheel.fenderRotMin then
558 angleDif = wheel.fenderRotMin - steeringAngle
559 end
560 setRotation(wheel.fenderNode, 0, angleDif, 0)
561 end
562
563 return changed
564 end
```

getTireFriction

Description

Returns tire friction

Definition

getTireFriction(integer tireType, integer groundType, float wetScale)

Arguments

integer	tireType	tire type index
integer	groundType	ground type index
float	wetScale	wet scale

Return Values

float tireFriction tire friction

Code

```
572 function WheelsUtil.getTireFriction(tireType, groundType,
    wetScale)
573 if wetScale == nil then
574 wetScale = 0
575 end
576 local coeff =
    WheelsUtil.tireTypes[tireType].frictionCoeffs[groundType]
577 local coeffWet =
    WheelsUtil.tireTypes[tireType].frictionCoeffsWet[groundType]
578 return coeff + (coeffWet-coeff)*wetScale
579 end
```

getGroundType

Description

Get ground type

Definition

getGroundType(boolean isField, boolean isRoad, float depth)

Arguments

boolean isField is on field

boolean isRoad is on road

float depth depth of terrain

Return Values

integer groundType ground type

Code

```
587 function WheelsUtil.getGroundType(isField, isRoad, depth)
588 -- terrain softness:
589 -- [ 0, 0.1]: road
590 -- [0.1, 0.8]: hard terrain
591 -- [0.8, 1 ]: soft terrain
592 if isField then
593 return WheelsUtil.GROUND_FIELD
594 elseif isRoad or depth < 0.1 then
595 return WheelsUtil.GROUND_ROAD
596 else
597 if depth > 0.8 then
598 return WheelsUtil.GROUND_SOFT_TERRAIN
599 else
600 return WheelsUtil.GROUND_HARD_TERRAIN
601 end
602 end
603 end
```

WorkAreaTypeManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
19 function WorkAreaTypeManager:new(customMt)
20 local self = AbstractManager:new(customMt or
   WorkAreaTypeManager_mt)
21
```

```
22  return self
23  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
27  function WorkAreaTypeManager:initDataStructures()
28  self.workAreaTypes = {}
29  self.workAreaTypeNameToInt = {}
30  self.workAreaTypeNameToDesc = {}
31  WorkAreaType = self.workAreaTypeNameToInt
32  end
```

AIConveyorBelt

Description

Specialization for AI converyor belts

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
19  function AIConveyorBelt.prerequisitesPresent(specializations)
20  return SpecializationUtil.hasSpecialization(AIVehicle,
    specializations) and
    SpecializationUtil.hasSpecialization(Motorized, specializations)
21  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
47  function AIConveyorBelt:onLoad(savegame)
48  local spec = self.spec_aiConveyorBelt
```

```
49
50  spec.isAllowed = hasXMLProperty(self.xmlFile,
    "vehicle.ai.conveyorBelt")
51
52  spec.minAngle = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.ai.conveyorBelt#minAngle"), 5)
53  spec.maxAngle = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.ai.conveyorBelt#maxAngle"), 45)
54  spec.stepSize = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.ai.conveyorBelt#stepSize"), 5)
55  spec.currentAngle = spec.maxAngle
56
57  spec.minTargetWorldYRot = 0
58  spec.maxTargetWorldYRot = 0
59  spec.currentDirection = 0
60  spec.currentSpeed = 0
61
62  spec.speed = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.ai.conveyorBelt#speed"), 1)
63  spec.direction = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.ai.conveyorBelt#direction"), -1)
64  end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
82  function AIConveyorBelt:onReadStream(streamId, connection)
83  self:setAIConveyorBeltAngle(streamReadInt8(streamId), true)
84  end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
90  function AIConveyorBelt:onWriteStream(streamId, connection)
91  streamWriteInt8(streamId, self.spec_aiConveyorBelt.currentAngle)
92  end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
99  function AIConveyorBelt:onUpdate(dt, isActiveForInput,
    isSelected)
100 local spec = self.spec_aiConveyorBelt
101 if self.isServer then
102 if self:getIsAIActive() then
103 spec.currentDirection, spec.currentSpeed =
    self:getDirectionAndSpeedToTargetAngle(spec.currentDirection,
    spec.minTargetWorldYRot, spec.maxTargetWorldYRot)
104
105 self:getMotor():setSpeedLimit(math.abs(spec.currentSpeed *
    spec.speed))
106 WheelsUtil.updateWheelsPhysics(self, dt, spec.currentSpeed *
    spec.speed * spec.direction, spec.currentDirection *
    spec.direction, false, true)
107 end
108 end
109 end
```

onUpdateTick

Description

Called on updateTick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
116 function AIConveyorBelt:onUpdateTick(dt, isActiveForInput,
    isSelected)
117 local spec = self.spec_aiConveyorBelt
```

```
118  if self.isClient then
119  local actionEvent = spec.actionEvents[InputAction.IMPLEMENT_EXTRA3]
120  if actionEvent ~= nil then
121  g_inputBinding:setActionEventActive(actionEvent.actionEventId, isActiveForInput)
122  if isActiveForInput then
123  g_inputBinding:setActionEventText(actionEvent.actionEventId, string.format(g_i18n:getText("action_conveyorBeltChangeAngle"), string.format("%.0f", spec.currentAngle)))
124  end
125  end
126  end
127  end
```

AIImplement

Description

Specialization for AI Implements

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17  function AIImplement.prerequisitesPresent(specializations)
18  return true
19  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
97  function AIImplement:onLoad(savegame)
98  local spec = self.spec_aiImplement
99
100 local baseName = "vehicle.ai"
101
```

```
102 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, baseName .. ".areaMarkers#leftIndex",
    baseName .. ".areaMarkers#leftNode") -- FS17 to FS 19
103 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, baseName .. ".areaMarkers#rightIndex",
    baseName .. ".areaMarkers#rightNode") -- FS17 to FS 19
104 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, baseName .. ".areaMarkers#backIndex",
    baseName .. ".areaMarkers#backNode") -- FS17 to FS 19
105
106 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, baseName .. ".sizeMarkers#leftIndex",
    baseName .. ".sizeMarkers#leftNode") -- FS17 to FS 19
107 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, baseName .. ".sizeMarkers#rightIndex",
    baseName .. ".sizeMarkers#rightNode") -- FS17 to FS 19
108 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, baseName .. ".sizeMarkers#backIndex",
    baseName .. ".sizeMarkers#backNode") -- FS17 to FS 19
109
110 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, baseName ..
    ".trafficCollisionTrigger#index", baseName ..
    ".collisionTrigger#node") -- FS17 to FS 19
111 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, baseName .. ".trafficCollisionTrigger#node",
    baseName .. ".collisionTrigger#node") -- FS17 to FS 19
112 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, baseName .. ".collisionTrigger#index",
    baseName .. ".collisionTrigger#node") -- FS17 to FS 19
113 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.aiLookAheadSize#value", baseName ..
    ".lookAheadSize#value") -- FS17 to FS 19
114
115 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, baseName ..
    ".toolReverserDirectionNode#index", baseName ..
    ".toolReverserDirectionNode#node") -- FS17 to FS 19
116 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, baseName .. ".turningRadiusLimiation",
    baseName .. ".turningRadiusLimitation") -- FS17 to FS 19
117
118 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, baseName .. ".forceTurnNoBackward#value",
    baseName .. ".allowTurnBackward#value (inverted)") -- FS17 to FS
    19
119
120
```

```
121 spec.minTurningRadius = getXMLFloat(self.xmlFile, baseName ..
    ".minTurningRadius#value")
122
123 spec.leftMarker = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseName .. ".areaMarkers#leftNode"),
    self.i3dMappings)
124 spec.rightMarker = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseName .. ".areaMarkers#rightNode"),
    self.i3dMappings)
125 spec.backMarker = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseName .. ".areaMarkers#backNode"),
    self.i3dMappings)
126 spec.aiMarkersInverted = false
127
128 spec.sizeLeftMarker = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseName .. ".sizeMarkers#leftNode"),
    self.i3dMappings)
129 spec.sizeRightMarker = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseName .. ".sizeMarkers#rightNode"),
    self.i3dMappings)
130 spec.sizeBackMarker = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseName .. ".sizeMarkers#backNode"),
    self.i3dMappings)
131
132 spec.collisionTrigger = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseName .. ".collisionTrigger#node"),
    self.i3dMappings)
133 if spec.collisionTrigger ~= nil then
134 local rigidBodyType = getRigidBodyType(spec.collisionTrigger)
135 if rigidBodyType ~= "Kinematic" then
136 g_logManager:xmlWarning(self.configFileName,
    "'aiCollisionTrigger' is not a kinematic body type")
137 end
138 end
139
140 spec.needsLowering = Utils.getNoNil(getXMLBool(self.xmlFile,
    baseName .. ".needsLowering#value"), true)
141 spec.lowerIfAnyIsLowerd = Utils.getNoNil(getXMLBool(self.xmlFile,
    baseName .. ".needsLowering#lowerIfAnyIsLowerd"), false)
142
143 spec.allowTurnBackward = Utils.getNoNil(getXMLBool(self.xmlFile,
    baseName .. ".allowTurnBackward#value"), true)
144
145 spec.toolReverserDirectionNode =
    I3DUtil.indexToObject(self.components, getXMLString(self.xmlFile,
    baseName .. ".toolReverserDirectionNode#node"), self.i3dMappings)
```

```
146
147 spec.turningRadiusLimitation = {}
148 spec.turningRadiusLimitation.rotationJoint =
    I3DUtil.indexToObject(self.components, getXMLString(self.xmlFile,
    baseName .. ".turningRadiusLimitation#rotationJointNode"),
    self.i3dMappings)
149 if spec.turningRadiusLimitation.rotationJoint ~= nil then
150 spec.turningRadiusLimitation.wheelIndices =
    {StringUtil.getVectorFromString(getXMLString(self.xmlFile,
    baseName .. ".turningRadiusLimitation#wheelIndices"))}
151 end
152
153 spec.turningRadiusLimitation.radius = getXMLFloat(self.xmlFile,
    baseName .. ".turningRadiusLimitation#radius")
154
155 spec.lookAheadSize = Utils.getNoNil(getXMLFloat(self.xmlFile,
    baseName .. ".lookAheadSize#value"), 2)
156 spec.useAttributesOfAttachedImplement =
    Utils.getNoNil(getXMLBool(self.xmlFile, baseName ..
    ".useAttributesOfAttachedImplement#value"), false)
157
158 spec.hasNoFullCoverageArea =
    Utils.getNoNil(getXMLString(self.xmlFile, baseName ..
    ".hasNoFullCoverageArea#value"), false)
159
160 spec.terrainDetailRequiredValueRanges = {}
161 spec.terrainDetailProhibitedValueRanges = {}
162
163 spec.requiredFruitTypes = {}
164 spec.prohibitedFruitTypes = {}
165
166 spec.isLineStarted = false
167 end
```

AIVehicle

Description

Specialization for AI vehicles

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
62  function AIVehicle.prerequisitesPresent(specializations)
63  return SpecializationUtil.hasSpecialization(Drivable,
    specializations)
64  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
152 function AIVehicle:onLoad(savegame)
153 local spec = self.spec_aiVehicle
154
155 spec.aiSteeringSpeed = Utils.getNoNil(getXMLFloat(spec.xmlFile,
    "vehicle.ai.steeringSpeed"), 1)*0.001
156
157 spec.isActive = false
158
159 spec.aiImplementList = {}
160 spec.aiImplementDataDirtyFlag = true
161
162 spec.taskList = {}
163
164 spec.driveStrategies = {}
165
166 spec.didNotMoveTimeout = Utils.getNoNil(
    getXMLFloat(spec.xmlFile, "vehicle.ai.didNotMoveTimeout#value"),
    5000)
167 if getXMLBool(spec.xmlFile,
    "vehicle.ai.didNotMoveTimeout#deactivated") then
168 spec.didNotMoveTimeout = math.huge
169 end
170
171 spec.didNotMoveTimer = spec.didNotMoveTimeout
172
173 spec.debugTexts = {}
174 spec.debugLines = {}
175
```

```
176  spec.aiTrafficCollision = nil
177  spec.aiTrafficCollisionRemoveDelay = 0
178  spec.aiTrafficCollisionTranslation = {0, 0, 20}
179  spec.aiTrafficCollisionScale = {40, 30, 40}
180
181  spec.pricePerMS = Utils.getNoNil(getXMLFloat(spec.xmlFile,
     "vehicle.ai.pricePerHour"), 2000)/60/60/1000
182
183  spec.steeringNode = I3DUtil.indexToObject(self.components,
     getXMLString(self.xmlFile, "vehicle.ai.steeringNode#node"),
     self.i3dMappings)
184  end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
190  function AIVehicle:onReadStream(streamId, connection)
191  if streamReadBool(streamId) then
192  local helperIndex = streamReadUInt8(streamId)
193  local farmId = streamReadUIntN(streamId,
     FarmManager.FARM_ID_SEND_NUM_BITS)
194  self:startAIVehicle(helperIndex, true, farmId)
195  end
196  end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
202  function AIVehicle:onWriteStream(streamId, connection)
203  local spec = self.spec_aiVehicle
204  if streamWriteBool(streamId, self:getIsAIActive()) then
205  streamWriteUInt8(streamId, spec.currentHelper.index)
```

```
206 streamWriteUIntN(streamId, spec.startedFarmId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
207 end
208 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
215 function AIVehicle:onUpdate(dt, isActiveForInput, isSelected)
216 local spec = self.spec_aiVehicle
217
218 if VehicleDebug.state == VehicleDebug.DEBUG_AI and
    isActiveForInput then
219 if #spec.debugTexts > 0 then
220 for i, text in pairs(spec.debugTexts) do
221 renderText(0.7, 0.92-(0.02*i), 0.02, text)
222 end
223 end
224 if #spec.debugLines > 0 then
225 for _, l in pairs(spec.debugLines) do
226 drawDebugLine(l.s[1],l.s[2],l.s[3], l.c[1],l.c[2],l.c[3],
    l.e[1],l.e[2],l.e[3], l.c[1],l.c[2],l.c[3])
227 end
228 end
229 end
230
231 if spec.aiImplementDataDirtyFlag then
232 spec.aiImplementDataDirtyFlag = false
233 self:updateAIImplementData()
234 end
235
236 if spec.isServer then
237 if self:getIsAIActive() then
238 if spec.driveStrategies ~= nil and #spec.driveStrategies > 0 then
239 for i=1,#spec.driveStrategies do
240 local driveStrategy = spec.driveStrategies[i]
```

```
241 driveStrategy:update(dt)
242 end
243 end
244 end
245 end
246
247 -- as long as the ai is turned on we raise active (e.g. if the
    vehicle stops due a collision the vehicle collision may sleep and
    stop raising active)
248 if self:getIsAIActive() then
249 self:raiseActive()
250 end
251 end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
258 function AIVehicle:onUpdateTick(dt, isActiveForInput, isSelected)
259 local spec = self.spec_aiVehicle
260
261 self:clearAIDebugTexts()
262 self:clearAIDebugLines()
263
264 if self.isServer then
265 if self:getIsAIActive() then
266 local difficultyMultiplier =
    g_currentMission.missionInfo.buyPriceMultiplier;
267 local price = -dt * difficultyMultiplier * spec.pricePerMS
268
269 -- If field was not owned, it is a mission. Increase the price
    for balancing.
270 if self.getLastTouchedFarmlandFarmId ~= nil and
    self:getLastTouchedFarmlandFarmId() == 0 then
271 price = price * MissionManager.AI_PRICE_MULTIPLIER
272 end
273
```

```
274 g_currentMission:addMoney(price, spec.startedFarmId,
    "wagePayment");
275 g_currentMission:addMoneyChange(price, spec.startedFarmId,
    FSBaseMission.MONEY_TYPE_AI)
276
277 if spec.driveStrategies ~= nil and #spec.driveStrategies > 0 then
278 local vX,vY,vZ =
    getWorldTranslation(self:getAIVehicleSteeringNode())
279
280 local tX, tZ, moveForwards, maxSpeedStra, maxSpeed,
    distanceToStop
281 for i=1,#spec.driveStrategies do
282 local driveStrategy = spec.driveStrategies[i]
283 tX, tZ, moveForwards, maxSpeedStra, distanceToStop =
    driveStrategy:getDriveData(dt, vX,vY,vZ)
284 maxSpeed = math.min(maxSpeedStra or math.huge, maxSpeed or
    math.huge)
285 if tX ~= nil or not self:getIsAIActive() then
286 break
287 end
288 end
289
290 if tX == nil then
291 if self:getIsAIActive() then -- check if AI is still active,
    because it might have been kicked by a strategy
292 self:stopAIVehicle(AIVehicle.STOP_REASON_REGULAR)
293 end
294 end
295
296 if not self:getIsAIActive() then
297 return
298 end
299
300 local acceleration = 1.0
301
302 local minimumSpeed = 5
303 local lookAheadDistance = 5
304
305 local distSpeed = math.max(minimumSpeed, maxSpeed * math.min(1,
    distanceToStop/lookAheadDistance))
306 local speedLimit, _ = self:getSpeedLimit()
307 maxSpeed = math.min(maxSpeed, distSpeed, speedLimit)
```

```
maxSpeed = math.min(maxSpeed, self:getCruiseControlMaxSpeed())

local isAllowedToDrive = maxSpeed ~= 0

local pX, _, pZ = worldToLocal(self:getAIVehicleSteeringNode(), tX,vY,tZ)
if not moveForwards and self.spec_articulatedAxis ~= nil then
if self.spec_articulatedAxis.aiRevereserNode ~= nil then
pX, _, pZ = worldToLocal(self.spec_articulatedAxis.aiRevereserNode, tX,vY,tZ)
end
end

AIVehicleUtil.driveToPoint(self, dt, acceleration, isAllowedToDrive, moveForwards, pX, pZ, maxSpeed)

-- worst case check: did not move but should have moved
if isAllowedToDrive and self:getLastSpeed() < 0.5 then
spec.didNotMoveTimer = spec.didNotMoveTimer - dt
else
spec.didNotMoveTimer = spec.didNotMoveTimeout
end

if spec.didNotMoveTimer < 0 then
self:stopAIVehicle(AIVehicle.STOP_REASON_BLOCKED_BY_OBJECT)
end
end

if #spec.taskList > 0 then
for i, task in pairs(spec.taskList) do
self:addAIDebugText(string.format("AI TASK: %d - %s", i, task.getFunc))
if task.getObject ~= nil then
if task.getObject[task.getFunc](task.getObject, unpack(task.getParams)) then
task.setObject[task.setFunc](task.setObject, unpack(task.setParams))
spec.taskList[i] = nil
end
else
task.setObject[task.setFunc](task.setObject, unpack(task.setParams))
```

```
343 spec.taskList[i] = nil
344 end
345 end
346 end
347
348 self:raiseAIEvent("onAIActive", "onAIImplementActive")
349 else
350 if spec.aiTrafficCollisionRemoveDelay > 0 then
351 spec.aiTrafficCollisionRemoveDelay =
    spec.aiTrafficCollisionRemoveDelay - dt
352 if spec.aiTrafficCollisionRemoveDelay <= 0 then
353 if spec.aiTrafficCollision ~= nil then
354 if entityExists(spec.aiTrafficCollision) then
355 delete(spec.aiTrafficCollision)
356 end
357 end
358 spec.aiTrafficCollisionRemoveDelay = 0
359 end
360 end
361 end
362 end
363
364 if self.isClient then
365 local actionEvent = spec.actionEvents[InputAction.TOGGLE_AI]
366 if actionEvent ~= nil then
367 local showAction = false
368
369 if self:getIsActiveForInput(true, true) then
370 -- if ai is active we always display the dismiss helper action
371 showAction = self:getCanStartAIVehicle() or self:getIsAIActive()
372
373 if showAction then
374 if self:getIsAIActive() then
375 g_inputBinding:setActionEventText(actionEvent.actionEventId,
    g_i18n:getText("action_dismissEmployee"))
376 else
377 g_inputBinding:setActionEventText(actionEvent.actionEventId,
    g_i18n:getText("action_hireEmployee"))
378 end
379 end
380 end
```

```
381
382 g_inputBinding:setActionEventActive(actionEvent.actionEventId,
    showAction)
383 end
384 end
385 end
```

getCanStartAIVehicle

Description

Returns true if ai can start

Definition

getCanStartAIVehicle()

Return Values

boolean canStart can start ai

Code

```
390 function AIVehicle:getCanStartAIVehicle()
391 local spec = self.spec_aiVehicle
392
393 if self:getAIVehicleDirectionNode() == nil then
394 return false
395 end
396
397 if g_currentMission.disableAIVehicle then
398 return false
399 end
400
401 if AIVehicle.numHirablesHired >= g_currentMission.maxNumHirables
    then
402 return false
403 end
404
405 if #spec.aiImplementList == 0 then
406 return false
407 end
408
409 return true
410 end
```

getCanAIVehicleContinueWork

Description

Returns true if ai can contiue

Definition

getCanAIVehicleContinueWork()

Return Values

boolean canContiue can contiue ai

Code

```
415  function AIVehicle:getCanAIVehicleContinueWork()
416  for _, implement in ipairs(self:getAttachedAIImplements()) do
417  if not implement.object:getCanAIImplementContinueWork() then
418  return false
419  end
420  end
421
422  if SpecializationUtil.hasSpecialization(AIImplement,
     self.specializations) then
423  if not self:getCanAIImplementContinueWork() then
424  return false
425  end
426  end
427
428  return true
429  end
```

startAIVehicle

Description

Starts ai vehicle

Definition

startAIVehicle(integer helperIndex, boolean noEventSend)

Arguments

integer helperIndex index of hired helper

boolean noEventSend no event send

Code

```
463  function AIVehicle:startAIVehicle(helperIndex, noEventSend, startedFarmI
464  local spec = self.spec_aiVehicle
465
466  if not self:getIsAIActive() then
467  if helperIndex ~= nil then
468  spec.currentHelper = g_helperManager:getHelperByIndex(helperIndex)
469  else
470  spec.currentHelper = g_helperManager:getRandomHelper()
471  end
472
473  g_helperManager:useHelper(spec.currentHelper)
474
475  spec.startedFarmId = startedFarmId
```

```
476
477 g_farmManager:getFarmById(startedFarmId).stats:updateStats("workersHired
    1)
478
479 if noEventSend == nil or noEventSend == false then
480 local event = AIVehicleSetStartedEvent:new(self, nil, true,
    spec.currentHelper, startedFarmId)
481 if g_server ~= nil then
482 g_server:broadcastEvent(event, nil, nil, self)
483 else
484 g_client:getServerConnection():sendEvent(event)
485 end
486 end
487
488 AIVehicle.numHirablesHired = AIVehicle.numHirablesHired + 1
489
490 if self.setRandomVehicleCharacter ~= nil then
491 self:setRandomVehicleCharacter()
492 end
493
494 local hotspotX, _, hotspotZ = getWorldTranslation(spec.rootNode)
495
496 local _, textSize = getNormalizedScreenValues(0, 9)
497 local _, textOffsetY = getNormalizedScreenValues(0, 18)
498 local width, height = getNormalizedScreenValues(24, 24)
499 spec.mapAIHotspot = MapHotspot:new("helper", MapHotspot.CATEGORY_AI)
500 spec.mapAIHotspot:setSize(width, height)
501 spec.mapAIHotspot:setLinkedNode(spec.components[1].node)
502 spec.mapAIHotspot:setText(spec.currentHelper.name)
503 spec.mapAIHotspot:setImage(nil, getNormalizedUVs(MapHotspot.UV.HELPER),
    {0.052, 0.1248, 0.672, 1})
504 spec.mapAIHotspot:setBackgroundImage(nil,
    getNormalizedUVs(MapHotspot.UV.HELPER))
505 spec.mapAIHotspot:setIconScale(0.7)
506 spec.mapAIHotspot:setTextOptions(textSize, nil, textOffsetY, {1, 1, 1, 1
    Overlay.ALIGN_VERTICAL_MIDDLE)
507 g_currentMission:addMapHotspot(spec.mapAIHotspot)
508
509 spec.isActive = true
510
511 if spec.isServer then
```

```
512 self:updateAIImplementData()
513 self:updateAIDriveStrategies()
514 end
515
516 self:raiseAIEvent("onAIStart", "onAIImplementStart")
517 self:requestActionEventUpdate()
518
519 local collisionRoot = g_i3DManager:loadSharedI3DFile(AIVehicle.TRAFFIC_COLLISION_BOX_FILENAME, self.baseDirectory, false, true, false)
520 if collisionRoot ~= nil and collisionRoot ~= 0 then
521 local collision = getChildAt(collisionRoot, 0)
522
523 link(self.components[1].node, collision)
524
525 setTranslation(collision, unpack(spec.aiTrafficCollisionTranslation))
526 setScale(collision, unpack(spec.aiTrafficCollisionScale))
527 spec.aiTrafficCollision = collision
528
529 delete(collisionRoot)
530 end
531 end
532 end
```

stopAIVehicle

Description

Stops ai vehicle

Definition

stopAIVehicle(integer reason, boolean noEventSend)

Arguments

integer	reason	reason
boolean	noEventSend	no event send

Code

```
538 function AIVehicle:stopAIVehicle(reason, noEventSend)
539 local spec = self.spec_aiVehicle
540
541 if self:getIsAIActive() then
542 if noEventSend == nil or noEventSend == false then
543 local event = AIVehicleSetStartedEvent:new(self, reason, false, nil, spec.startedFarmId)
544
545 if g_server ~= nil then
546 g_server:broadcastEvent(event, nil, nil, self)
```

```
547 else
548 g_client:getServerConnection():sendEvent(event)
549 end
550 end
551
552 if self.isClient then
553 if g_currentMission.player ~= nil then
554 if g_currentMission.player.farmId == spec.startedFarmId then
555 if reason ~= nil and reason ~= AIVehicle.STOP_REASON_USER then
556 local notificationType = FSBaseMission.INGAME_NOTIFICATION_CRITICAL
557 if reason == AIVehicle.STOP_REASON_REGULAR then
558 notificationType = FSBaseMission.INGAME_NOTIFICATION_OK
559 end
560
561 g_currentMission:addIngameNotification(notificationType,
    string.format(g_i18n:getText(AIVehicle.REASON_TEXT_MAPPING[reason]),
    spec.currentHelper.name))
562 end
563 end
564 end
565 end
566
567 g_helperManager:releaseHelper(spec.currentHelper)
568 spec.currentHelper = nil
569
570 g_farmManager:updateFarmStats(spec.startedFarmId, "workersHired", -
    1)
571
572 AIVehicle.numHirablesHired = math.max(AIVehicle.numHirablesHired -
    1, 0)
573
574 if self.restoreVehicleCharacter ~= nil then
575 self:restoreVehicleCharacter()
576 end
577
578 if spec.mapAIHotspot ~= nil then
579 g_currentMission:removeMapHotspot(spec.mapAIHotspot)
580 spec.mapAIHotspot:delete()
581 spec.mapAIHotspot = nil
582 end
583
```

```
584 self:setCruiseControlState(Drivable.CRUISECONTROL_STATE_OFF, true)
585 if spec.isServer then
586 WheelsUtil.updateWheelsPhysics(self, 0,
    spec.lastSpeedReal*spec.movingDirection, 0, true, true)
587
588 if spec.driveStrategies ~= nil and #spec.driveStrategies > 0 then
589 for i=#spec.driveStrategies,1,-1 do
590 spec.driveStrategies[i]:delete()
591 table.remove(spec.driveStrategies, i)
592 end
593 spec.driveStrategies = {}
594 end
595 end
596
597 spec.isActive = false
598 spec.isTurning = false
599
600 -- move the collision far under the ground and remove it 200ms
    delayed (avoids problems with the traffic doesn't get the trigger
    onLeave callback of the collision box)
601 setTranslation(spec.aiTrafficCollision, 0, -1000, 0)
602 spec.aiTrafficCollisionRemoveDelay = 200
603
604 self:raiseAIEvent("onAIEnd", "onAIImplementEnd")
605 self:requestActionEventUpdate()
606 end
607 end
```

getDirectionSnapAngle

Description

Get direction shape angle

Definition

getDirectionSnapAngle()

Return Values

float direction shape angle

Code

```
635 function AIVehicle:getDirectionSnapAngle()
636 return 0
637 end
```

updateAIImplementData

Description

Fills aiImplementList with vehicles to use by ai

Definition

updateAIImplementData()

Code

```
641 function AIVehicle:updateAIImplementData()
642 local spec = self.spec_aiVehicle
643
644 spec.aiImplementList = {}
645 self:addVehicleToAIImplementList(spec.aiImplementList)
646 end
```

updateAIDriveStrategies

Description

Set drive strategies depending on the vehicle

Definition

updateAIDriveStrategies()

Code

```
658 function AIVehicle:updateAIDriveStrategies()
659 local spec = self.spec_aiVehicle
660
661 if #spec.aiImplementList > 0 then
662 if spec.driveStrategies ~= nil and #spec.driveStrategies > 0 then
663 for i=#spec.driveStrategies,1,-1 do
664 spec.driveStrategies[i]:delete()
665 table.remove(spec.driveStrategies, i)
666 end
667 spec.driveStrategies = {}
668 end
669
670 local foundCombine = false
671 local foundBaler = false
672 for _,implement in pairs(spec.aiImplementList) do
673 if SpecializationUtil.hasSpecialization(Combine,
    implement.object.specializations) then
674 foundCombine = true
675 end
676 if SpecializationUtil.hasSpecialization(Baler,
    implement.object.specializations) then
677 foundBaler = true
678 end
679 end
680
681 foundCombine = foundCombine or
    SpecializationUtil.hasSpecialization(Combine,
    spec.specializations)
```

```
682  if foundCombine then
683  local driveStrategyCombine = AIDriveStrategyCombine:new()
684  driveStrategyCombine:setAIVehicle(self)
685  table.insert(spec.driveStrategies, driveStrategyCombine)
686  end
687
688  foundBaler = foundBaler or
     SpecializationUtil.hasSpecialization(Baler, spec.specializations)
689  if foundBaler then
690  local driveStrategyCombine = AIDriveStrategyBaler:new()
691  driveStrategyCombine:setAIVehicle(self)
692  table.insert(spec.driveStrategies, driveStrategyCombine)
693  end
694
695  local driveStrategyCollision = AIDriveStrategyCollision:new()
696  local driveStrategyStraight = AIDriveStrategyStraight:new()
697
698  driveStrategyCollision:setAIVehicle(self)
699  driveStrategyStraight:setAIVehicle(self)
700
701  table.insert(spec.driveStrategies, driveStrategyCollision)
702  table.insert(spec.driveStrategies, driveStrategyStraight)
703  end
704  end
```

saveStatsToXMLFile

Description

Returns string with states for game stats xml file

Definition

saveStatsToXMLFile()

Return Values

string attributes stats attributes

Code

```
723  function AIVehicle:saveStatsToXMLFile(xmlFile, key)
724  setXMLBool(xmlFile, key.."#isAIActive", self:getIsAIActive())
725  end
```

onEnterVehicle

Description

Called on enter vehicle

Definition

onEnterVehicle(boolean isControlling)

Arguments

boolean isControlling is player controlling the vehicle

Code

```
811  function AIVehicle:onEnterVehicle(isControlling)
812  self:setAIMapHotspotVisibility(false)
813  end
```

onLeaveVehicle

Description

Called on leaving the vehicle

Definition

onLeaveVehicle()

Code

```
817  function AIVehicle:onLeaveVehicle()
818  self:setAIMapHotspotVisibility(true)
819  end
```

getDeactivateOnLeave

Description

Get deactivate on leaving

Definition

getDeactivateOnLeave()

Return Values

boolean deactivateOnLeave deactivate on leaving

Code

```
824  function AIVehicle:getDeactivateOnLeave(superFunc)
825  return superFunc(self) and not self:getIsAIActive()
826  end
```

AnimatedVehicle

Description

Class for all AnimatedVehicles

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
20  function AnimatedVehicle.prerequisitesPresent(specializations)
21  return true
22  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
70  function AnimatedVehicle:onLoad(savegame)
71  local spec = self.spec_animatedVehicle
72
73  spec.animations = {}
74
75  local i = 0
76  while true do
77  local key = string.format("vehicle.animations.animation(%d)", i)
78  if not hasXMLProperty(self.xmlFile, key) then
79  break
80  end
81
82  local animation = {}
83
84  if self:loadAnimation(self.xmlFile, key, animation) then
85  spec.animations[animation.name] = animation
86  end
87
88  i = i + 1
89  end
90
91  spec.activeAnimations = {}
92  spec.numActiveAnimations = 0
93  end
```

onPostLoad

Description

Called after loading

Definition

onPostLoad(table savegame)

Arguments

table savegame savegame

Code

```
98  function AnimatedVehicle:onPostLoad(savegame)
99  local spec = self.spec_animatedVehicle
100 for name, animation in pairs(spec.animations) do
```

```
101 if animation.resetOnStart then
102 self:playAnimation(name, -1, nil, true)
103 AnimatedVehicle.updateAnimationByName(self, name, 9999999)
104 end
105 end
106 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
113 function AnimatedVehicle:onUpdate(dt, isActiveForInput,
    isSelected)
114 AnimatedVehicle.updateAnimations(self, dt)
115
116 if self.spec_animatedVehicle.numActiveAnimations > 0 then
117 self:raiseActive()
118 end
119 end
```

initializeAnimationParts

Description

Initialize parts of animation

Definition

initializeAnimationParts(table animation)

Arguments

table animation animation

Code

```
338 function AnimatedVehicle:initializeAnimationParts(animation)
339 local numParts = table.getn(animation.parts)
340
341 for i, part in ipairs(animation.parts) do
342 self:initializeAnimationPart(animation, part, i, numParts)
343 end
344
345 for i, part in ipairs(animation.parts) do
346 self:postInitializeAnimationPart(animation, part, i, numParts)
347 end
```

```
348 end
```

initializeAnimationPart

Description

Initialize part of animation

Definition

initializeAnimationPart(table part)

Arguments

table part part

Code

```
353 function AnimatedVehicle:initializeAnimationPart(animation, part,
    i, numParts)
354 -- rot, trans, scale
355 AnimatedVehicle.initializeAnimationPartAttribute(self, animation,
    part, i, numParts, "nextRotPart", "prevRotPart", "startRot",
    "endRot", "rotation")
356 AnimatedVehicle.initializeAnimationPartAttribute(self, animation,
    part, i, numParts, "nextTransPart", "prevTransPart",
    "startTrans", "endTrans", "translation")
357 AnimatedVehicle.initializeAnimationPartAttribute(self, animation,
    part, i, numParts, "nextScalePart", "prevScalePart",
    "startScale", "endScale", "scale")
358
359 --shader params
360 AnimatedVehicle.initializeAnimationPartAttribute(self, animation,
    part, i, numParts, "nextShaderPart", "prevShaderPart",
    "shaderStartValues", "shaderEndValues", "shaderParameter")
361
362 -- animation clips
363 AnimatedVehicle.initializeAnimationPartAttribute(self, animation,
    part, i, numParts, "nextClipPart", "prevClipPart",
    "clipStartTime", "clipEndTime", "animation clip")
364
365 -- dependent animations
366 AnimatedVehicle.initializeAnimationPartAttribute(self, animation,
    part, i, numParts, "nextDependentAnimPart",
    "prevDependentAnimPart", "dependentAnimStartTime",
    "dependentAnimEndTime", "dependent animation", nil, nil,
    "dependentAnim")
367
368 if self.isServer then
369 -- joint limits
370 AnimatedVehicle.initializeAnimationPartAttribute(self, animation,
    part, i, numParts, "nextRotLimitPart", "prevRotLimitPart",
    "startRotMinLimit", "endRotMinLimit", "joint rot limit",
    "startRotMaxLimit", "endRotMaxLimit", "componentJoint")
```

```
371 AnimatedVehicle.initializeAnimationPartAttribute(self, animation, part, i, numParts, "nextTransLimitPart", "prevTransLimitPart", "startTransMinLimit", "startTransMinLimit", "joint trans limit", "startTransMaxLimit", "endTransMaxLimit", "componentJoint")
372 end
373 end
```

postInitializeAnimationPart

Description

Post Initialize part of animation (normally used to set default start value if not set by the end value of the previous part)

Definition

postInitializeAnimationPart(table part)

Arguments

table part part

Code

```
378 function AnimatedVehicle:postInitializeAnimationPart(animation, part, i, numParts)
379 if part.endRot ~= nil and part.startRot == nil then
380 local x,y,z = getRotation(part.node)
381 part.startRot = {x,y,z}
382 end
383 if part.endTrans ~= nil and part.startTrans == nil then
384 local x,y,z = getTranslation(part.node)
385 part.startTrans = {x,y,z}
386 end
387 if part.endScale ~= nil and part.startScale == nil then
388 local x,y,z = getScale(part.node)
389 part.startScale = {x,y,z}
390 end
391 if self.isServer then
392 if part.endRotMinLimit ~= nil and part.startRotMinLimit == nil then
393 local rotLimit = part.componentJoint.rotMinLimit
394 part.startRotMinLimit = {rotLimit[1], rotLimit[2], rotLimit[3]}
395 end
396 if part.endRotMaxLimit ~= nil and part.startRotMaxLimit == nil then
397 local rotLimit = part.componentJoint.rotLimit
398 part.startRotMaxLimit = {rotLimit[1], rotLimit[2], rotLimit[3]}
399 end
400 if part.endTransMinLimit ~= nil and part.startTransMinLimit == nil then
401 local transLimit = part.componentJoint.transMinLimit
```

```
402 part.startTransMinLimit = {transLimit[1], transLimit[2],
    transLimit[3]}
403 end
404 if part.endTransMaxLimit ~= nil and part.startTransMaxLimit ==
    nil then
405 local transLimit = part.componentJoint.transLimit
406 part.startTransMaxLimit = {transLimit[1], transLimit[2],
    transLimit[3]}
407 end
408 end
409 end
```

playAnimation

Description

Play animation

Definition

playAnimation(string name, float speed, float animTime, boolean noEventSend)

Arguments

string	name	name of animation
float	speed	speed
float	animTime	start time
boolean	noEventSend	no event send

Code

```
418 function AnimatedVehicle:playAnimation(name, speed, animTime, noEventSen
419 local spec = self.spec_animatedVehicle
420
421 local animation = spec.animations[name]
422 if animation ~= nil then
423 SpecializationUtil.raiseEvent(self, "onPlayAnimation", name)
424
425 if speed == nil then
426 speed = animation.currentSpeed
427 end
428
429 -- skip animation if speed is not set or 0 to allow skipping animations
    xml speed attribute set to 0
430 if speed == nil or speed == 0 then
431 return
432 end
433
434 if animTime == nil then
435 if self:getIsAnimationPlaying(name) then
436 animTime = self:getAnimationTime(name)
```

```
437     elseif speed > 0 then
438     animTime = 0
439     else
440     animTime = 1
441     end
442     end
443     if noEventSend == nil or noEventSend == false then
444     if g_server ~= nil then
445     g_server:broadcastEvent(AnimatedVehicleStartEvent:new(self, name, speed,
        animTime), nil, nil, self)
446     else
447     g_client:getServerConnection():sendEvent(AnimatedVehicleStartEvent:new(s
        name, speed, animTime))
448     end
449     end
450
451     if spec.activeAnimations[name] == nil then
452     spec.activeAnimations[name] = animation
453     spec.numActiveAnimations = spec.numActiveAnimations + 1
454     SpecializationUtil.raiseEvent(self, "onStartAnimation", name)
455     end
456     animation.currentSpeed = speed
457     animation.currentTime = animTime*animation.duration
458     self:resetAnimationValues(animation)
459
460     self:raiseActive()
461     end
462     end
```

stopAnimation

Description

Stop animation

Definition

stopAnimation(string name, boolean noEventSend)

Arguments

string	name	name of animation
boolean	noEventSend	no event send

Code

```
468     function AnimatedVehicle:stopAnimation(name, noEventSend)
469     local spec = self.spec_animatedVehicle
470
471     if noEventSend == nil or noEventSend == false then
```

```
472 if g_server ~= nil then
473 g_server:broadcastEvent(AnimatedVehicleStopEvent:new(self, name), nil, r
    self)
474 else
475 g_client:getServerConnection():sendEvent(AnimatedVehicleStopEvent:new(se
    name))
476 end
477 end
478 local animation = spec.animations[name]
479 if animation ~= nil then
480 SpecializationUtil.raiseEvent(self, "onStopAnimation", name)
481 animation.stopTime = nil
482 end
483
484 if spec.activeAnimations[name] ~= nil then
485 spec.numActiveAnimations = spec.numActiveAnimations - 1
486 spec.activeAnimations[name] = nil
487 SpecializationUtil.raiseEvent(self, "onFinishAnimation", name)
488 end
489 end
```

getAnimationExists

Description

Returns true if animation exits

Definition

getAnimationExists(string name)

Arguments

string name name of animation

Return Values

boolean exists animation axists

Code

```
495 function AnimatedVehicle:getAnimationExists(name)
496 local spec = self.spec_animatedVehicle
497
498 return spec.animations[name] ~= nil
499 end
```

getIsAnimationPlaying

Description

Returns true if animation is playing

Definition

getIsAnimationPlaying(string name)

Arguments

string name name of animation

Return Values

boolean isPlaying animation is playing

Code

```
505  function AnimatedVehicle:getIsAnimationPlaying(name)
506  local spec = self.spec_animatedVehicle
507
508  return spec.activeAnimations[name] ~= nil
509  end
```

getRealAnimationTime

Description

Returns real animation time

Definition

getRealAnimationTime(string name)

Arguments

string name name of animation

Return Values

float animTime real animation time in ms

Code

```
515  function AnimatedVehicle:getRealAnimationTime(name)
516  local spec = self.spec_animatedVehicle
517
518  local animation = spec.animations[name]
519  if animation ~= nil then
520  return animation.currentTime
521  end
522  return 0
523  end
```

setRealAnimationTime

Description

Set animation real time

Definition

setRealAnimationTime(string name, float animTime, boolean update)

Arguments

string	name	name of animation
float	animTime	real animation time in ms
boolean	update	update animation

Code

```
530  function AnimatedVehicle:setRealAnimationTime(name, animTime,
     update)
531  local spec = self.spec_animatedVehicle
532
533  local animation = spec.animations[name]
```

```
534  if animation ~= nil then
535  if update == nil or update then
536  local currentSpeed = animation.currentSpeed
537  animation.currentSpeed = 1
538  if animation.currentTime > animTime then
539  animation.currentSpeed = -1
540  end
541
542  self:resetAnimationValues(animation)
543
544  local dtToUse, _ =
     AnimatedVehicle.updateAnimationCurrentTime(self, animation,
     99999999, animTime)
545  AnimatedVehicle.updateAnimation(self, animation, dtToUse, false)
546  animation.currentSpeed = currentSpeed
547  else
548  animation.currentTime = animTime
549  end
550  end
551  end
```

getAnimationTime

Description

Returns animation time

Definition

getAnimationTime(string name)

Arguments

string name name of animation

Return Values

float animTime animation time [0..1]

Code

```
557  function AnimatedVehicle:getAnimationTime(name)
558  local spec = self.spec_animatedVehicle
559
560  local animation = spec.animations[name]
561  if animation ~= nil then
562  return animation.currentTime/animation.duration
563  end
564  return 0
565  end
```

setAnimationTime

Description

Set animation time

Definition

setAnimationTime(string name, float animTime, boolean update)

Arguments

string	name	name of animation
float	animTime	animation time [0..1]
boolean	update	update animation

Code

```
572  function AnimatedVehicle:setAnimationTime(name, animTime, update)
573  local spec = self.spec_animatedVehicle
574
575  if spec.animations == nil then
576  printCallstack()
577  end
578
579  local animation = spec.animations[name]
580  if animation ~= nil then
581  self:setRealAnimationTime(name, animTime*animation.duration,
     update)
582  end
583  end
```

getAnimationDuration

Description

Returns duration of animation

Definition

getAnimationDuration(string name)

Arguments

string name name of animation

Return Values

float duration duration in ms

Code

```
589  function AnimatedVehicle:getAnimationDuration(name)
590  local spec = self.spec_animatedVehicle
591
592  local animation = spec.animations[name]
593  if animation ~= nil then
594  return animation.duration
595  end
596  return 1
597  end
```

setAnimationSpeed

Description

Sets speed of animation

Definition

setAnimationSpeed(string name, float speed)

Arguments

string name name of animation
float speed speed

Code

```
603 function AnimatedVehicle:setAnimationSpeed(name, speed)
604 local spec = self.spec_animatedVehicle
605
606 local animation = spec.animations[name]
607 if animation ~= nil then
608 local speedReversed = false
609 if (animation.currentSpeed > 0) ~= (speed > 0) then
610 speedReversed = true
611 end
612 animation.currentSpeed = speed
613 if self:getIsAnimationPlaying(name) and speedReversed then
614 self:resetAnimationValues(animation)
615 end
616 end
617 end
```

setAnimationStopTime

Description

Sets animation stop time

Definition

setAnimationStopTime(string name, float stopTime)

Arguments

string name name of animation
float stopTime stop time [0..1]

Code

```
623 function AnimatedVehicle:setAnimationStopTime(name, stopTime)
624 local spec = self.spec_animatedVehicle
625
626 local animation = spec.animations[name]
627 if animation ~= nil then
628 animation.stopTime = stopTime*animation.duration
629 end
630 end
```

resetAnimationValues

Description

Resets animation values

Definition

resetAnimationValues(table animation)

Arguments

table animation animation

Code

```
635  function AnimatedVehicle:resetAnimationValues(animation)
636  AnimatedVehicle.findCurrentPartIndex(animation)
637  for _, part in ipairs(animation.parts) do
638  self:resetAnimationPartValues(part)
639  end
640  end
```

resetAnimationPartValues

Description

Resets animation part

Definition

resetAnimationPartValues(table part)

Arguments

table part part to reset

Code

```
645  function AnimatedVehicle:resetAnimationPartValues(part)
646  part.curRot = nil
647  part.speedRot = nil
648  part.curTrans = nil
649  part.speedTrans = nil
650  part.curScale = nil
651  part.speedScale = nil
652  part.curVisibility = nil
653  part.curRotMinLimit = nil
654  part.curRotMaxLimit = nil
655  part.speedRotLimit = nil
656  part.curTransMinLimit = nil
657  part.curTransMaxLimit = nil
658  part.speedTransLimit = nil
659  part.shaderCurValues = nil
660  part.curClipTime = nil
661  part.curDependentAnimTime = nil
662  end
```

animPartSorter

Description

Returns true if anim parts are in the right order

Definition

animPartSorter(table a, table b)

Arguments

table a part a to check

table b part b to check

Return Values

boolean rightOrder returns true if parts are in right order

Code

```
763  function AnimatedVehicle.animPartSorter(a, b)
764  if a.startTime < b.startTime then
765  return true
766  elseif a.startTime == b.startTime then
767  return a.duration < b.duration
768  end
769  return false
770  end
```

animPartSorterReverse

Description

Returns true if anim parts are in the reverse right order

Definition

animPartSorterReverse(table a, table b)

Arguments

table a part a to check

table b part b to check

Return Values

boolean rightOrder returns true if parts are in reverse right order

Code

```
777  function AnimatedVehicle.animPartSorterReverse(a, b)
778  local endTimeA = a.startTime + a.duration
779  local endTimeB = b.startTime + b.duration
780  if endTimeA > endTimeB then
781  return true
782  elseif endTimeA == endTimeB then
783  return a.startTime > b.startTime
784  end
785  return false
786  end
```

getMovedLimitedValue

Description

Returns moved limited value

Definition

getMovedLimitedValue(float currentValue, float destValue, float speed, float dt)

Arguments

float currentValue current value

float destValue	dest value
float speed	speed
float dt	time since last call in ms

Return Values

float ret limited value

Code

```
795 function AnimatedVehicle.getMovedLimitedValue(currentValue, destValue, speed, dt)
796 local limitF = math.min
797 -- we are moving towards -inf, we need to check for the maximum
798 if destValue < currentValue then
799 limitF = math.max
800 elseif destValue == currentValue then
801 return currentValue
802 end
803 local ret = limitF(currentValue + speed * dt, destValue)
804 return ret
805 end
```

setMovedLimitedValues3

Description

Sets moved limited values (3)

Definition

setMovedLimitedValues3(table currentValues, table destValues, table speeds, float dt)

Arguments

table currentValues	current values
table destValues	dest values
table speeds	speeds
float dt	time since last call in ms

Code

```
813 function AnimatedVehicle.setMovedLimitedValues3(currentValues, destValues, speeds, dt)
814 local hasChanged = false
815 for i=1,3 do
816 local newValue = AnimatedVehicle.getMovedLimitedValue(currentValues[i], destValues[i], speeds[i], dt)
817 if currentValues[i] ~= newValue then
818 hasChanged = true
819 currentValues[i] = newValue
820 end
821 end
822 return hasChanged
823 end
```

setMovedLimitedValues4

Description

Sets moved limited values (4)

Definition

setMovedLimitedValues4(table currentValues, table destValues, table speeds, float dt)

Arguments

table currentValues	current values
table destValues	dest values
table speeds	speeds
float dt	time since last call in ms

Code

```
831  function AnimatedVehicle.setMovedLimitedValues4(currentValues, destValues, speeds, dt)
832  local hasChanged = false
833  for i=1,4 do
834  local newValue = AnimatedVehicle.getMovedLimitedValue(currentValues[i], destValues[i], speeds[i], dt)
835  if currentValues[i] ~= newValue then
836  hasChanged = true
837  currentValues[i] = newValue
838  end
839  end
840  return hasChanged
841  end
```

findCurrentPartIndex

Description

Find current playing part

Definition

findCurrentPartIndex(table animation)

Arguments

table animation animation

Return Values

integer index of current playing part

Code

```
847  function AnimatedVehicle.findCurrentPartIndex(animation)
848  if animation.currentSpeed > 0 then
849  -- find the first part that is being played at the current time
850  animation.currentPartIndex = table.getn(animation.parts)+1
851  for i, part in ipairs(animation.parts) do
852  if part.startTime+part.duration >= animation.currentTime then
853  animation.currentPartIndex = i
```

```
854 break
855 end
856 end
857 else
858 -- find the last part that is being played at the current time
    (the first in partsReverse)
859 animation.currentPartIndex = table.getn(animation.partsReverse)+1
860 for i, part in ipairs(animation.partsReverse) do
861 if part.startTime <= animation.currentTime then
862 animation.currentPartIndex = i
863 break
864 end
865 end
866 end
867 end
```

getDurationToEndOfPart

Description

Returns duration to the end of current part

Definition

getDurationToEndOfPart(table part, table anim)

Arguments

table part part

table anim animation

Return Values

float duration duration to end of current part

Code

```
874 function AnimatedVehicle.getDurationToEndOfPart(part, anim)
875 if anim.currentSpeed > 0 then
876 return part.startTime+part.duration - anim.currentTime
877 else
878 return anim.currentTime - part.startTime
879 end
880 end
```

getNextPartIsPlaying

Description

Get next part is playing

Definition

getNextPartIsPlaying(table nextPart, table prevPart, table anim, boolean default)

Arguments

table nextPart next part

table prevPart previous part

table anim animation

boolean default default value

Return Values

boolean isPlaying next part is playing

Code

```
889  function AnimatedVehicle.getNextPartIsPlaying(nextPart, prevPart, anim, default)
890  if anim.currentSpeed > 0 then
891  if nextPart ~= nil then
892  return nextPart.startTime > anim.currentTime
893  end
894  else
895  if prevPart ~= nil then
896  return prevPart.startTime + prevPart.duration < anim.currentTime
897  end
898  end
899  return default
900  end
```

updateAnimations

Description

Update animations

Definition

updateAnimations(float dt)

Arguments

float dt time since last call in ms

Code

```
905  function AnimatedVehicle.updateAnimations(self, dt)
906  local spec = self.spec_animatedVehicle
907
908  for _, anim in pairs(spec.activeAnimations) do
909  local dtToUse, stopAnim = AnimatedVehicle.updateAnimationCurrentTime(self, anim, dt, anim.stopTime)
910  AnimatedVehicle.updateAnimation(self, anim, dtToUse, stopAnim)
911  end
912  end
```

updateAnimationByName

Description

Update animation by name

Definition

updateAnimationByName(string animName, float dt)

Arguments

string animName name of animation to update

float	dt	time since last call in ms

Code

```
function AnimatedVehicle.updateAnimationByName(self, animName, dt)
    local spec = self.spec_animatedVehicle

    local anim = spec.animations[animName]
    if anim ~= nil then
        local dtToUse, stopAnim = AnimatedVehicle.updateAnimationCurrentTime(self, anim, dt, anim.stopTime)
        AnimatedVehicle.updateAnimation(self, anim, dtToUse, stopAnim)
    end
end
```

updateAnimationCurrentTime

Description

Update current animation time

Definition

updateAnimationCurrentTime(table anim, float dt, float stopTime)

Arguments

table	anim	animation
float	dt	time since last call in ms
float	stopTime	stop time

Return Values

float	dtToUse	dt to use
boolean	stopAnimation	stop animation

Code

```
function AnimatedVehicle.updateAnimationCurrentTime(self, anim, dt, stopTime)
    anim.currentTime = anim.currentTime + dt*anim.currentSpeed

    local absSpeed = math.abs(anim.currentSpeed)
    local dtToUse = dt*absSpeed
    local stopAnim = false
    if stopTime ~= nil then
        if anim.currentSpeed > 0 then
            if stopTime <= anim.currentTime then
                dtToUse = dtToUse-(anim.currentTime-stopTime)
                anim.currentTime = stopTime
                stopAnim = true
            end
        else
```

```
949 if stopTime >= anim.currentTime then
950 dtToUse = dtToUse-(stopTime-anim.currentTime)
951 anim.currentTime = stopTime
952 stopAnim = true
953 end
954 end
955 end
956 return dtToUse, stopAnim
957 end
```

updateAnimation

Description

Update animation

Definition

updateAnimation(table anim, float dtToUse, boolean stopAnimation)

Arguments

table	anim	animation
float	dtToUse	dt to use
boolean	stopAnimation	stop animation

Code

```
964 function AnimatedVehicle.updateAnimation(self, anim, dtToUse,
    stopAnim)
965 local spec = self.spec_animatedVehicle
966
967 local numParts = table.getn(anim.parts)
968 local parts = anim.parts
969 if anim.currentSpeed < 0 then
970 parts = anim.partsReverse
971 end
972
973 if dtToUse > 0 then
974 local hasChanged = false
975 local nothingToChangeYet = false
976 for partI=anim.currentPartIndex, numParts do
977 local part = parts[partI]
978
979 local isInRange = true
980 if part.requiredAnimation ~= nil then
981 local time = self:getAnimationTime(part.requiredAnimation)
982 if time < part.requiredAnimationRange[1] or time >
    part.requiredAnimationRange[2] then
983 isInRange = false
```

```
984  end
985  end
986
987  if (part.direction == 0 or ((part.direction > 0) ==
     (anim.currentSpeed >= 0))) and isInRange then
988  local durationToEnd =
     AnimatedVehicle.getDurationToEndOfPart(part, anim)
989
990  -- is this part not playing yet?
991  if durationToEnd > part.duration then
992  nothingToChangeYet = true
993  break
994  end
995
996  local realDt = dtToUse
997
998  if anim.currentSpeed > 0 then
999  local startT = anim.currentTime-dtToUse
1000 if startT < part.startTime then
1001 realDt = dtToUse - part.startTime + startT
1002 end
1003 else
1004 local startT = anim.currentTime+dtToUse
1005 local endTime = part.startTime + part.duration
1006 if startT > endTime then
1007 realDt = dtToUse - (startT - endTime)
1008 end
1009 end
1010
1011 durationToEnd = durationToEnd+realDt
1012
1013 if self:updateAnimationPart(anim, part, durationToEnd, dtToUse,
     realDt) then
1014 if self.setMovingToolDirty ~= nil then
1015 self:setMovingToolDirty(part.node)
1016 end
1017 hasChanged = true
1018 end
1019 end
1020
1021 if partI == anim.currentPartIndex then
```

```
1022  -- is this part finished?
1023  if (anim.currentSpeed > 0 and part.startTime + part.duration <
      anim.currentTime) or
1024  (anim.currentSpeed <= 0 and part.startTime > anim.currentTime)
1025  then
1026  self:resetAnimationPartValues(part)
1027  anim.currentPartIndex = anim.currentPartIndex+1
1028  end
1029  end
1030  end
1031  if not nothingToChangeYet and not hasChanged and
      anim.currentPartIndex >= numParts then
1032  -- end the animation
1033  if anim.currentSpeed > 0 then
1034  anim.currentTime = anim.duration
1035  else
1036  anim.currentTime = 0
1037  end
1038  stopAnim = true
1039  end
1040  end
1041  if stopAnim or anim.currentPartIndex > numParts or
      anim.currentPartIndex < 1 then
1042  if not stopAnim then
1043  if anim.currentSpeed > 0 then
1044  anim.currentTime = anim.duration
1045  else
1046  anim.currentTime = 0
1047  end
1048  end
1049  anim.currentTime = math.min(math.max(anim.currentTime, 0),
      anim.duration)
1050  anim.stopTime = nil
1051  if spec.activeAnimations[anim.name] ~= nil then
1052  spec.numActiveAnimations = spec.numActiveAnimations - 1
1053  spec.activeAnimations[anim.name] = nil
1054  SpecializationUtil.raiseEvent(self, "onFinishAnimation",
      anim.name)
1055  end
1056
1057  if anim.looping then
```

```
1058  -- restart animation
1059  self:setAnimationTime(anim.name, math.abs((anim.duration-
      anim.currentTime) - 1), true)
1060  self:playAnimation(anim.name, anim.currentSpeed, nil, true)
1061  end
1062  end
1063  end
```

updateAnimationPart

Description

Update animation part

Definition

updateAnimationPart(table anim, table part, float durationToEnd, float dtToUse, float realDt)

Arguments

table	anim	animation
table	part	part
float	durationToEnd	duration to end
float	dtToUse	dt to use
float	realDt	real dt

Code

```
1072  function AnimatedVehicle:updateAnimationPart(animation, part,
      durationToEnd, dtToUse, realDt)
1073  local hasPartChanged = false
1074  if part.startRot ~= nil and (durationToEnd > 0 or
      AnimatedVehicle.getNextPartIsPlaying(part.nextRotPart,
      part.prevRotPart, animation, true)) then
1075  local destRot = part.endRot
1076  if animation.currentSpeed < 0 then
1077  destRot = part.startRot
1078  end
1079  if part.curRot == nil then
1080  local x,y,z = getRotation(part.node)
1081  part.curRot = {x,y,z}
1082  local invDuration = 1.0/math.max(durationToEnd, 0.001)
1083  part.speedRot = {(destRot[1]-x)*invDuration, (destRot[2]-
      y)*invDuration, (destRot[3]-z)*invDuration}
1084  end
1085  if AnimatedVehicle.setMovedLimitedValues3(part.curRot, destRot,
      part.speedRot, realDt) then
1086  setRotation(part.node, part.curRot[1], part.curRot[2],
      part.curRot[3])
1087  hasPartChanged = true
1088  end
1089  end
```

```
1090  if part.startTrans ~= nil and (durationToEnd > 0 or AnimatedVehicle.getNextPartIsPlaying(part.nextTransPart, part.prevTransPart, animation, true)) then
1091  local destTrans = part.endTrans
1092  if animation.currentSpeed < 0 then
1093  destTrans = part.startTrans
1094  end
1095  if part.curTrans == nil then
1096  local x,y,z = getTranslation(part.node)
1097  part.curTrans = {x,y,z}
1098  local invDuration = 1.0/math.max(durationToEnd, 0.001)
1099  part.speedTrans = {(destTrans[1]-x)*invDuration, (destTrans[2]-y)*invDuration, (destTrans[3]-z)*invDuration}
1100  end
1101  if AnimatedVehicle.setMovedLimitedValues3(part.curTrans, destTrans, part.speedTrans, realDt) then
1102  setTranslation(part.node, part.curTrans[1], part.curTrans[2], part.curTrans[3])
1103  hasPartChanged = true
1104  end
1105  end
1106  if part.startScale ~= nil and (durationToEnd > 0 or AnimatedVehicle.getNextPartIsPlaying(part.nextScalePart, part.prevScalePart, animation, true)) then
1107  local destScale = part.endScale
1108  if animation.currentSpeed < 0 then
1109  destScale = part.startScale
1110  end
1111  if part.curScale == nil then
1112  local x,y,z = getScale(part.node)
1113  part.curScale = {x,y,z}
1114  local invDuration = 1.0/math.max(durationToEnd, 0.001)
1115  part.speedScale = {(destScale[1]-x)*invDuration, (destScale[2]-y)*invDuration, (destScale[3]-z)*invDuration}
1116  end
1117  if AnimatedVehicle.setMovedLimitedValues3(part.curScale, destScale, part.speedScale, realDt) then
1118  setScale(part.node, part.curScale[1], part.curScale[2], part.curScale[3])
1119  hasPartChanged = true
1120  end
1121  end
```

```
1122  if part.shaderParameter ~= nil and (durationToEnd > 0 or
      AnimatedVehicle.getNextPartIsPlaying(part.nextShaderPart,
      part.prevShaderPart, animation, true)) then
1123  local destValues = part.shaderEndValues
1124  if animation.currentSpeed < 0 then
1125  destValues = part.shaderStartValues
1126  end
1127  if part.shaderCurValues == nil then
1128  local x,y,z,w = getShaderParameter(part.node,
      part.shaderParameter)
1129  part.shaderCurValues = {x,y,z,w}
1130  local invDuration = 1.0 / math.max(durationToEnd, 0.001)
1131  part.speedShader = {(destValues[1]-x)*invDuration,
      (destValues[2]-y)*invDuration, (destValues[3]-z)*invDuration,
      (destValues[4]-w)*invDuration}
1132  end
1133  if AnimatedVehicle.setMovedLimitedValues4(part.shaderCurValues,
      destValues, part.speedShader, realDt) then
1134  setShaderParameter(part.node, part.shaderParameter,
      part.shaderCurValues[1], part.shaderCurValues[2],
      part.shaderCurValues[3], part.shaderCurValues[4], false)
1135  hasPartChanged = true
1136  end
1137  end
1138
1139  if part.animationClip ~= nil and (durationToEnd > 0 or
      AnimatedVehicle.getNextPartIsPlaying(part.nextClipPart,
      part.prevClipPart, animation, true)) then
1140  local destValue = part.clipEndTime
1141  if animation.currentSpeed < 0 then
1142  destValue = part.clipStartTime
1143  end
1144  local forceUpdate = false
1145  if part.curClipTime == nil then
1146  local oldClipIndex =
      getAnimTrackAssignedClip(part.animationCharSet, 0)
1147  clearAnimTrackClip(part.animationCharSet, 0);
1148  assignAnimTrackClip(part.animationCharSet, 0,
      part.animationClipIndex);
1149
1150  part.curClipTime = part.clipStartTime
1151  if oldClipIndex == part.animationClipIndex then
1152  part.curClipTime = getAnimTrackTime(part.animationCharSet, 0)
```

```
1153  end
1154
1155  local invDuration = 1.0 / math.max(durationToEnd, 0.001)
1156  part.speedClip = (destValue-part.curClipTime)*invDuration
1157  forceUpdate = true
1158  end
1159  local newTime =
      AnimatedVehicle.getMovedLimitedValue(part.curClipTime,
      destValue, part.speedClip, realDt)
1160  if newTime ~= part.curClipTime or forceUpdate then
1161  part.curClipTime = newTime
1162
1163  enableAnimTrack(part.animationCharSet, 0);
1164  setAnimTrackTime(part.animationCharSet, 0, newTime, true);
1165  disableAnimTrack(part.animationCharSet, 0);
1166
1167  hasPartChanged = true
1168  end
1169  end
1170
1171  if part.visibility ~= nil then
1172  if part.curVisibility == nil then
1173  part.curVisibility = getVisibility(part.node)
1174  end
1175  if part.visibility ~= part.curVisibility then
1176  part.curVisibility = part.visibility
1177  setVisibility(part.node, part.visibility)
1178  hasPartChanged = true
1179  end
1180  end
1181
1182  if part.dependentAnim ~= nil and (durationToEnd > 0 or
      AnimatedVehicle.getNextPartIsPlaying(part.nextDependentAnimPart,
      part.prevDependentAnimPart, animation, true)) then
1183  if self:getAnimationExists(part.dependentAnim) then
1184  local destValue = part.dependentAnimEndTime
1185  if animation.currentSpeed < 0 then
1186  destValue = part.dependentAnimStartTime
1187  end
1188  local forceUpdate = false
1189  if part.curDependentAnimTime == nil then
```

```
1190    part.curDependentAnimTime =
        self:getAnimationTime(part.dependentAnim)
1191
1192    local invDuration = 1.0 / math.max(durationToEnd, 0.001)
1193    part.speedDependentAnim = (destValue-
        part.curDependentAnimTime)*invDuration
1194    forceUpdate = true
1195    end
1196    local newTime =
        AnimatedVehicle.getMovedLimitedValue(part.curDependentAnimTime,
        destValue, part.speedDependentAnim, realDt)
1197    if newTime ~= part.curDependentAnimTime or forceUpdate then
1198    part.curDependentAnimTime = newTime
1199    self:setAnimationTime(part.dependentAnim, newTime, true)
1200    hasPartChanged = true
1201    end
1202    else
1203    g_logManager:xmlWarning(self.configFileName, "Unable to find
        dependent animation '%s'", part.dependentAnim)
1204    end
1205    end
1206
1207    if self.isServer then
1208    if part.startRotMinLimit ~= nil and (durationToEnd > 0 or
        AnimatedVehicle.getNextPartIsPlaying(part.nextRotLimitPart,
        part.prevRotLimitPart, animation, true)) then
1209    local destRotMinLimit = part.endRotMinLimit
1210    local destRotMaxLimit = part.endRotMaxLimit
1211    if animation.currentSpeed < 0 then
1212    destRotMinLimit = part.startRotMinLimit
1213    destRotMaxLimit = part.startRotMaxLimit
1214    end
1215    if part.curRotMinLimit == nil then
1216    local x,y,z = unpack(part.componentJoint.rotMinLimit)
1217    part.curRotMinLimit = {x,y,z}
1218    local invDuration = 1.0/math.max(durationToEnd, 0.001)
1219    part.speedRotMinLimit = {(destRotMinLimit[1]-x)*invDuration,
        (destRotMinLimit[2]-y)*invDuration, (destRotMinLimit[3]-
        z)*invDuration}
1220    end
1221    if part.curRotMaxLimit == nil then
1222    local x,y,z = unpack(part.componentJoint.rotLimit)
```

```
part.curRotMaxLimit = {x,y,z}
local invDuration = 1.0/math.max(durationToEnd, 0.001)
part.speedRotMaxLimit = {(destRotMaxLimit[1]-x)*invDuration, (destRotMaxLimit[2]-y)*invDuration, (destRotMaxLimit[3]-z)*invDuration}
end
for i=1, 3 do
local newRotMinLimit = AnimatedVehicle.getMovedLimitedValue(part.curRotMinLimit[i], destRotMinLimit[i], part.speedRotMinLimit[i], realDt)
local newRotMaxLimit = AnimatedVehicle.getMovedLimitedValue(part.curRotMaxLimit[i], destRotMaxLimit[i], part.speedRotMaxLimit[i], realDt)
if newRotMinLimit ~= part.curRotMinLimit[i] or newRotMaxLimit ~= part.curRotMaxLimit[i] then
part.curRotMinLimit[i] = newRotMinLimit
part.curRotMaxLimit[i] = newRotMaxLimit
self:setComponentJointRotLimit(part.componentJoint, i, newRotMinLimit, newRotMaxLimit)
hasPartChanged = true
end
end
end
if part.startTransMinLimit ~= nil and (durationToEnd > 0 or AnimatedVehicle.getNextPartIsPlaying(part.nextTransLimitPart, part.prevTransLimitPart, animation, true)) then
local destTransMinLimit = part.endTransMinLimit
local destTransMaxLimit = part.endTransMaxLimit
if animation.currentSpeed < 0 then
destTransMinLimit = part.startTransMinLimit
destTransMaxLimit = part.startTransMaxLimit
end
if part.curTransMinLimit == nil then
local x,y,z = unpack(part.componentJoint.transMinLimit)
part.curTransMinLimit = {x,y,z}
local invDuration = 1.0/math.max(durationToEnd, 0.001)
part.speedTransMinLimit = {(destTransMinLimit[1]-x)*invDuration, (destTransMinLimit[2]-y)*invDuration, (destTransMinLimit[3]-z)*invDuration}
end
if part.curTransMaxLimit == nil then
local x,y,z = unpack(part.componentJoint.transLimit)
part.curTransMaxLimit = {x,y,z}
```

```
1254  local invDuration = 1.0/math.max(durationToEnd, 0.001)
1255  part.speedTransMaxLimit = {(destTransMaxLimit[1]-x)*invDuration,
      (destTransMaxLimit[2]-y)*invDuration, (destTransMaxLimit[3]-
      z)*invDuration}
1256  end
1257  for i=1, 3 do
1258  local newTransMinLimit =
      AnimatedVehicle.getMovedLimitedValue(part.curTransMinLimit[i],
      destTransMinLimit[i], part.speedTransMinLimit[i], realDt)
1259  local newTransMaxLimit =
      AnimatedVehicle.getMovedLimitedValue(part.curTransMaxLimit[i],
      destTransMaxLimit[i], part.speedTransMaxLimit[i], realDt)
1260  if newTransMinLimit ~= part.curTransMinLimit[i] or
      newTransMaxLimit ~= part.curTransMaxLimit[i] then
1261  part.curTransMinLimit[i] = newTransMinLimit
1262  part.curTransMaxLimit[i] = newTransMaxLimit
1263  self:setComponentJointTransLimit(part.componentJoint, i,
      newTransMinLimit, newTransMaxLimit)
1264  hasPartChanged = true
1265  end
1266  end
1267  end
1268  end
1269
1270  return hasPartChanged
1271  end
```

ArticulatedAxis

Description

> **This is the specialization for vehicles which steer with an articulated axis (excavators, loaders ...)**

prerequisitesPresent

Description

> Checks if all prerequisite specializations are loaded

Definition

> prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17  function ArticulatedAxis.prerequisitesPresent(specializations)
18  return SpecializationUtil.hasSpecialization(Drivable,
    specializations)
19  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
30  function ArticulatedAxis:onLoad(savegame)
31  local xmlFile = self.xmlFile
32  local spec = self.spec_articulatedAxis
33
34  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.articulatedAxis.rotatingPart(0)#index",
    "vehicle.articulatedAxis.rotatingPart(0)#node") -- FS17
35
36  local index = getXMLInt(xmlFile,
    "vehicle.articulatedAxis#componentJointIndex")
37  if index ~= nil then
38  if index == 0 then
39  g_logManager:xmlWarning(self.configFileName, "Invalid component
    joint index '0' for articulatedAxis. Indices start with 1!")
40  else
41  local componentJoint = self.componentJoints[index]
42  local rotSpeed = getXMLFloat(xmlFile,
    "vehicle.articulatedAxis#rotSpeed")
43  local rotMax = getXMLFloat(xmlFile,
    "vehicle.articulatedAxis#rotMax")
44  local rotMin = getXMLFloat(xmlFile,
    "vehicle.articulatedAxis#rotMin")
45  if componentJoint ~= nil and rotSpeed ~= nil and rotMax ~= nil
    and rotMin ~= nil then
46  spec.rotSpeed = math.rad(rotSpeed)
47  spec.rotMax = math.rad(rotMax)
48  spec.rotMin = math.rad(rotMin)
49
50  spec.componentJoint = componentJoint
51  spec.anchorActor = Utils.getNoNil(getXMLInt(xmlFile,
    "vehicle.articulatedAxis#anchorActor"), 0)
52  spec.rotationNode = I3DUtil.indexToObject(self.components,
    getXMLString(xmlFile, "vehicle.articulatedAxis#rotNode"),
    self.i3dMappings)
53  if spec.rotationNode == nil then
```

```
54  spec.rotationNode = spec.componentJoint.jointNode
55  end
56
57  spec.curRot = 0
58
59  local i = 0
60  spec.rotatingParts = {}
61  while true do
62  local key =
    string.format("vehicle.articulatedAxis.rotatingPart(%d)", i)
63  if not hasXMLProperty(xmlFile, key) then
64  break
65  end
66
67  local node = I3DUtil.indexToObject(self.components,
    getXMLString(xmlFile, key .. "#node"), self.i3dMappings)
68  if node ~= nil then
69  local rotatingPart = {}
70  rotatingPart.node = node
71  rotatingPart.defRot = {getRotation(node)}
72  rotatingPart.posRot =
    StringUtil.getRadiansFromString(getXMLString(xmlFile, key ..
    "#posRot"), 3)
73  rotatingPart.negRot =
    StringUtil.getRadiansFromString(getXMLString(xmlFile, key ..
    "#negRot"), 3)
74  rotatingPart.negRotFactor = Utils.getNoNil(getXMLFloat(xmlFile,
    key .. "#negRotFactor"), 1)
75  rotatingPart.posRotFactor = Utils.getNoNil(getXMLFloat(xmlFile,
    key .. "#posRotFactor"), 1)
76  rotatingPart.invertSteeringAngle =
    Utils.getNoNil(getXMLBool(xmlFile, key ..
    "#invertSteeringAngle"), false)
77  table.insert(spec.rotatingParts, rotatingPart)
78  else
79  g_logManager:xmlWarning(self.configFileName, "Failed to load
    rotation part '%s'", key)
80  end
81  i = i + 1
82  end
83
84  -- adjust steering values
85  local maxRotTime = rotMax/rotSpeed
```

```
86  local minRotTime = rotMin/rotSpeed
87  if minRotTime > maxRotTime then
88  local temp = minRotTime
89  minRotTime = maxRotTime
90  maxRotTime = temp
91  end
92  if maxRotTime > self.maxRotTime then
93  self.maxRotTime = maxRotTime
94  end
95  if minRotTime < self.minRotTime then
96  self.minRotTime = minRotTime
97  end
98
99  self.maxRotation = rotMax
100 self.wheelSteeringDuration = MathUtil.sign(rotSpeed) * rotMax /
    rotSpeed
101
102 -- adjust variables used by AIVehicleUtil
103 local aiReverserNodeString = getXMLString(xmlFile,
    "vehicle.articulatedAxis#aiRevereserNode")
104 if aiReverserNodeString ~= nil then
105 spec.aiRevereserNode = I3DUtil.indexToObject(self.components,
    aiReverserNodeString, self.i3dMappings)
106 end
107
108 local maxTurningRadius = 0
109 local specWheels = self.spec_wheels
110 for i=1,2 do
111 local rootNode =
    self.components[componentJoint.componentIndices[i]].node
112
113 for _, wheel in ipairs(specWheels.wheels) do
114 if self:getParentComponent(wheel.repr) == rootNode then
115
116 local wx,_,wz = localToLocal(wheel.driveNode, rootNode, 0,0,0)
117 local dx1 = 1
118 if wx < 0 then
119 dx1 = -1
120 end
121 local dz1 = math.tan( math.max(wheel.rotMin, wheel.rotMax) )
122 if wz > 0 then
```

```
123  dz1 = -dz1
124  end
125
126  local x2, z2 = 0, 0
127  local dx2 = 1
128  if wx < 0 then
129  dx2 = -1
130  end
131  local dz2 = math.tan( math.max(rotMin, rotMax) )
132  if wz < 0 then
133  dz2 = -dz2
134  end
135
136  -- normalize directions
137  local l1 = MathUtil.vector2Length(dx1, dz1)
138  dx1, dz1 = dx1 / l1, dz1 / l1
139
140  local l2 = MathUtil.vector2Length(dx2, dz2)
141  dx2, dz2 = dx2 / l2, dz2 / l2
142
143  local intersect, _, f2 =
     MathUtil.getLineLineIntersection2D(wx,wz, dx1,dz1, x2,z2,
     dx2,dz2)
144  if intersect then
145  local radius = math.abs(f2)
146  maxTurningRadius = math.max(maxTurningRadius, radius)
147  end
148  end
149  end
150  end
151
152  if maxTurningRadius ~= 0 then
153  self.maxTurningRadius = maxTurningRadius
154  end
155  end
156  end
157  end
158
159  spec.interpolatedRotatedTime = 0
160  end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
176 function ArticulatedAxis:onUpdate(dt, isActiveForInput, isSelected)
177 local spec = self.spec_articulatedAxis
178 if spec.componentJoint ~= nil then
179 -- interpolatedRotatedTime to manipulate camera rot
180 if spec.interpolatedRotatedTime < self.rotatedTime then
181 spec.interpolatedRotatedTime = math.min(self.rotatedTime, spec.interpolatedRotatedTime + math.abs(spec.rotSpeed) * dt/500)
182 elseif spec.interpolatedRotatedTime > self.rotatedTime then
183 spec.interpolatedRotatedTime = math.max(self.rotatedTime, spec.interpolatedRotatedTime - math.abs(spec.rotSpeed) * dt/500)
184 end
185
186 local steeringAngle = MathUtil.clamp(self.rotatedTime * spec.rotSpeed, spec.rotMin, spec.rotMax)
187 if self.updateArticulatedAxisRotation ~= nil then
188 steeringAngle = self:updateArticulatedAxisRotation(steeringAngle, dt)
189 end
190
191 if math.abs(steeringAngle - spec.curRot) > 0.000001 then
192 if self.isServer then
193 setRotation(spec.rotationNode, 0, steeringAngle, 0)
194 self:setComponentJointFrame(spec.componentJoint, spec.anchorActor)
195 spec.curRot = steeringAngle
196 end
197
198 if self.isClient then
199 local percent = 0
200 if steeringAngle > 0 then
201 percent = steeringAngle / spec.rotMax
202 elseif steeringAngle < 0 then
```

```
203 percent = steeringAngle / spec.rotMin
204 end
205
206 for _,rotPart in pairs(spec.rotatingParts) do
207 local rx,ry,rz
208 if (steeringAngle > 0 and not rotPart.invertSteeringAngle) or
    (steeringAngle < 0 and rotPart.invertSteeringAngle) then
209 rx,ry,rz = MathUtil.vector3ArrayLerp(rotPart.defRot,
    rotPart.posRot, math.min(1,percent*rotPart.posRotFactor))
210 else
211 rx,ry,rz = MathUtil.vector3ArrayLerp(rotPart.defRot,
    rotPart.negRot, math.min(1,percent*rotPart.negRotFactor))
212 end
213 setRotation(rotPart.node, rx,ry,rz)
214 if self.setMovingToolDirty ~= nil then
215 self:setMovingToolDirty(rotPart.node)
216 end
217 end
218 end
219 end
220 end
221 end
```

Attachable

Description

This is the specialization for all vehicles that may be attached

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17 function Attachable.prerequisitesPresent(specializations)
18 return true
19 end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
97  function Attachable:onLoad(savegame)
98  local spec = self.spec_attachable
99
100 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "v
    "vehicle.inputAttacherJoints.inputAttacherJoint") -- FS15 to FS17
101 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "v
    "vehicle.inputAttacherJoints.inputAttacherJoint#needsLowering") -- FS15
102 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "v
    "vehicle.inputAttacherJoints.inputAttacherJoint#allowsLowering") -- FS15
103 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "v
    "vehicle.inputAttacherJoints.inputAttacherJoint#isDefaultLowered") -- FS
104 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.forceSelectionOnAttach#value",
    "vehicle.inputAttacherJoints.inputAttacherJoint#forceSelectionOnAttach")
105 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "v
    "vehicle.attacherJoint#topReferenceNode") -- FS15 to FS17
106 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "v
    "vehicle.attacherJoint#rootNode") -- FS15 to FS17
107
108 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "v
    "vehicle.attachable.inputAttacherJoints") -- FS17 to FS19
109 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.inputAttacherJointConfigurations", "vehicle.attachable.inputAtt
    FS17 to FS19
110 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "v
    "vehicle.attachable.brakeForce") -- FS17 to FS19
111 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "v
    "vehicle.attachable.steeringAxleAngleScale") -- FS17 to FS19
112 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "v
    "vehicle.attachable.support") -- FS17 to FS19
113 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "v
    "vehicle.attachable.lowerAnimation") -- FS17 to FS19
114 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "v
    "vehicle.attachable.toolCameras") -- FS17 to FS19
115 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.attachable.toolCameras#count", "vehicle.attachable.toolCameras"
116 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.attachable.toolCameras.toolCamera1", "vehicle.attachable.toolCa
117 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.attachable.toolCameras.toolCamera2", "vehicle.attachable.toolCa
118 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.attachable.toolCameras.toolCamera3", "vehicle.attachable.toolCa
```

```
119
120 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.foldable.foldingParts#onlyFoldOnDetach", "vehicle.attachable#al
    FS17 to FS19
121 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.maximalAirConsumptionPerFullStop", "vehicle.attachable.airConsu
    second at full brake power)") --FS17 to FS19
122
123 spec.attacherJoint = nil
124
125 spec.inputAttacherJoints = {}
126 local i = 0
127 while true do
128 local key = string.format("vehicle.attachable.inputAttacherJoints.inputA
129 if not hasXMLProperty(self.xmlFile, key) then
130 break
131 end
132
133 local inputAttacherJoint = {}
134 if self:loadInputAttacherJoint(self.xmlFile, key, inputAttacherJoint, i)
135 table.insert(spec.inputAttacherJoints, inputAttacherJoint)
136 end
137
138 i = i + 1
139 end
140
141 if self.configurations["inputAttacherJoint"] ~= nil then
142 local attacherConfigs =
    string.format("vehicle.attachable.inputAttacherJointConfigurations.input
    self.configurations["inputAttacherJoint"]-1)
143 local i=0
144 while true do
145 local baseName = string.format(attacherConfigs..".inputAttacherJoint(%d)
146 if not hasXMLProperty(self.xmlFile, baseName) then
147 break
148 end
149 local inputAttacherJoint = {}
150 if self:loadInputAttacherJoint(self.xmlFile, baseName, inputAttacherJoin
151 table.insert(spec.inputAttacherJoints, inputAttacherJoint)
152 end
153 i = i + 1
154 end
```

```
155 ObjectChangeUtil.updateObjectChanges(self.xmlFile,
    "vehicle.attachable.inputAttacherJointConfigurations.inputAttacherJointC
    self.configurations["inputAttacherJoint"], self.components, self)
156 end
157
158 spec.brakeForce = Utils.getNoNil(getXMLFloat(self.xmlFile, "vehicle.atta
159 spec.airConsumerUsage = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.attachable.airConsumer#usage"), 0)
160
161 spec.allowFoldingWhileAttached = Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.attachable#allowFoldingWhileAttached"), true)
162 spec.allowFoldingWhileLowered = Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.attachable#allowFoldingWhileLowered"), true)
163
164 spec.updateWheels = true
165 spec.updateSteeringAxleAngle = true
166
167 spec.isSelected = false
168 spec.attachTime = 0
169
170 spec.steeringAxleAngle = 0
171 spec.steeringAxleTargetAngle = 0
172
173 spec.steeringAxleAngleScaleStart = Utils.getNoNil(getXMLFloat(self.xmlFi
    "vehicle.attachable.steeringAxleAngleScale#startSpeed"), 10)
174 spec.steeringAxleAngleScaleEnd = Utils.getNoNil(getXMLFloat(self.xmlFile
    "vehicle.attachable.steeringAxleAngleScale#endSpeed"), 30)
175 spec.steeringAxleUpdateBackwards = Utils.getNoNil(getXMLBool(self.xmlFil
    "vehicle.attachable.steeringAxleAngleScale#backwards"), false)
176 spec.steeringAxleAngleSpeed = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.attachable.steeringAxleAngleScale#speed"), 0.001)
177 spec.steeringAxleUseSuperAttachable = Utils.getNoNil(getXMLFloat(self.xm
    "vehicle.attachable.steeringAxleAngleScale#useSuperAttachable"), false)
178 spec.steeringAxleTargetNode = I3DUtil.indexToObject(self.components, get
    "vehicle.attachable.steeringAxleAngleScale#targetNode"), self.i3dMapping
179 spec.steeringAxleAngleMinRot = Utils.getNoNilRad(getXMLFloat(self.xmlFil
    "vehicle.attachable.steeringAxleAngleScale#minRot"), 0)
180 spec.steeringAxleAngleMaxRot = Utils.getNoNilRad(getXMLFloat(self.xmlFil
    "vehicle.attachable.steeringAxleAngleScale#maxRot"), 0)
181
182 spec.supportAnimation = getXMLString(self.xmlFile, "vehicle.attachable.s
183
184 spec.lowerAnimation = getXMLString(self.xmlFile, "vehicle.attachable.low
```

```
185 spec.lowerAnimationSpeed = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.attachable.lowerAnimation#speed"), 1)
186 spec.lowerAnimationDirectionOnDetach = Utils.getNoNil(getXMLFloat(self.x
    "vehicle.attachable.lowerAnimation#directionOnDetach"), 0)
187
188 spec.toolCameras = {}
189 i = 0
190 while true do
191 local cameraKey = string.format("vehicle.attachable.toolCameras.toolCame
192 if not hasXMLProperty(self.xmlFile, cameraKey) then
193 break
194 end
195
196 local camera = VehicleCamera:new(self)
197 if camera:loadFromXML(self.xmlFile, cameraKey) then
198 table.insert(spec.toolCameras, camera)
199 end
200
201 i = i + 1
202 end
203
204 spec.isHardAttached = false
205 end
```

onPostLoad

Description

Called after loading

Definition

onPostLoad(table savegame)

Arguments

table savegame savegame

Code

```
210 function Attachable:onPostLoad(savegame)
211 local spec = self.spec_attachable
212
213 if spec.supportAnimation ~= nil and self.playAnimation ~= nil
    then
214 self:playAnimation(spec.supportAnimation, 1, nil, true)
215 AnimatedVehicle.updateAnimationByName(self,
    spec.supportAnimation, 9999999)
216 end
217
218 if savegame ~= nil and not savegame.resetVehicles then
```

```
219 if spec.lowerAnimation ~= nil and self.playAnimation ~= nil then
220 local lowerAnimTime = getXMLFloat(savegame.xmlFile,
    savegame.key..".attachable#lowerAnimTime")
221 if lowerAnimTime ~= nil then
222 local speed = 1
223 if lowerAnimTime < 0.5 then
224 speed = -1
225 end
226 self:playAnimation(spec.lowerAnimation, speed, nil, true)
227 self:setAnimationTime(spec.lowerAnimation, lowerAnimTime)
228 AnimatedVehicle.updateAnimationByName(self, spec.lowerAnimation,
    9999999)
229
230 if self.updateCylinderedInitial ~= nil then
231 self:updateCylinderedInitial(false)
232 end
233 end
234 end
235 end
236
237 for _, inputAttacherJoint in pairs(spec.inputAttacherJoints) do
238 if self.getMovingPartByNode ~= nil then
239 if inputAttacherJoint.steeringBarLeftNode ~= nil then
240 local movingPart =
    self:getMovingPartByNode(inputAttacherJoint.steeringBarLeftNode)
241 if movingPart ~= nil then
242 inputAttacherJoint.steeringBarLeftMovingPart = movingPart
243 else
244 inputAttacherJoint.steeringBarLeftNode = nil
245 end
246 end
247 if inputAttacherJoint.steeringBarRightNode ~= nil then
248 local movingPart =
    self:getMovingPartByNode(inputAttacherJoint.steeringBarRightNode)
249 if movingPart ~= nil then
250 inputAttacherJoint.steeringBarRightMovingPart = movingPart
251 else
252 inputAttacherJoint.steeringBarRightNode = nil
253 end
254 end
255 else
```

```
256 inputAttacherJoint.steeringBarLeftNode = nil
257 inputAttacherJoint.steeringBarRightNode = nil
258 end
259 end
260
261 if self.brake ~= nil then
262 self:brake(spec.brakeForce, true)
263 end
264
265 if #spec.inputAttacherJoints > 0 then
266 g_currentMission:addAttachableVehicle(self)
267 end
268 end
```

onPreDelete

Description

Called on before deleting

Definition

onPreDelete()

Code

```
272 function Attachable:onPreDelete()
273 local spec = self.spec_attachable
274
275 if spec.attacherVehicle ~= nil then
276 spec.attacherVehicle:detachImplementByObject(self, true)
277 end
278
279 g_currentMission:removeAttachableVehicle(self)
280 end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId
integer connection connection

Code

```
294 function Attachable:onReadStream(streamId, connection)
295 if streamReadBool(streamId) then
296 local object = NetworkUtil.readNodeObject(streamId)
297 local inputJointDescIndex = streamReadInt8(streamId)
```

```
298 local jointDescIndex = streamReadInt8(streamId)
299 local moveDown = streamReadBool(streamId)
300 local implementIndex = streamReadInt8(streamId)
301 if object ~= nil then
302 object:attachImplement(self, inputJointDescIndex, jointDescIndex,
    true, implementIndex)
303 object:setJointMoveDown(jointDescIndex, moveDown, true)
304 end
305 end
306 end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
312 function Attachable:onWriteStream(streamId, connection)
313 local spec = self.spec_attachable
314
315 streamWriteBool(streamId, spec.attacherVehicle ~= nil)
316 if spec.attacherVehicle ~= nil then
317 local attacherJointVehicleSpec =
    spec.attacherVehicle.spec_attacherJoints
318 local implementIndex =
    spec.attacherVehicle:getImplementIndexByObject(self)
319 local implement =
    attacherJointVehicleSpec.attachedImplements[implementIndex]
320 local inputJointDescIndex = spec.inputAttacherJointDescIndex
321 local jointDescIndex = implement.jointDescIndex
322 local jointDesc =
    attacherJointVehicleSpec.attacherJoints[jointDescIndex]
323 local moveDown = jointDesc.moveDown
324 NetworkUtil.writeNodeObject(streamId, spec.attacherVehicle)
325 streamWriteInt8(streamId, inputJointDescIndex)
326 streamWriteInt8(streamId, jointDescIndex)
327 streamWriteBool(streamId, moveDown)
328 streamWriteInt8(streamId, implementIndex)
329 end
330 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
337 function Attachable:onUpdate(dt, isActiveForInput, isSelected)
338 local spec = self.spec_attachable
339
340 if spec.updateSteeringAxleAngle then
341 if self:getLastSpeed() > 0.1 then
342 local baseVehicle = self:getSteeringAxleBaseVehicle()
343 if baseVehicle ~= nil and (self.movingDirection >= 0 or
    spec.steeringAxleUpdateBackwards) then
344 local yRot =
    Utils.getYRotationBetweenNodes(self.steeringAxleNode,
    baseVehicle.steeringAxleNode)
345 if math.abs(yRot) > 1.57 then
346 if yRot > 0 then
347 yRot = yRot - 3.14
348 else
349 yRot = yRot + 3.14
350 end
351 end
352
353 local startSpeed = spec.steeringAxleAngleScaleStart
354 local endSpeed = spec.steeringAxleAngleScaleEnd
355 local scale = MathUtil.clamp(1 + (self:getLastSpeed()-startSpeed)
    * 1.0/(startSpeed-endSpeed), 0, 1)
356 spec.steeringAxleTargetAngle = yRot*scale
357 elseif self:getLastSpeed() > 0.2 then
358 spec.steeringAxleTargetAngle = 0
359 end
360
361 local dir = MathUtil.sign(spec.steeringAxleTargetAngle -
    spec.steeringAxleAngle)
362 if dir == 1 then
```

```
363 spec.steeringAxleAngle = math.min(spec.steeringAxleAngle + dir*dt*spec.steeringAxleAngleSpeed, spec.steeringAxleTargetAngle)
364 else
365 spec.steeringAxleAngle = math.max(spec.steeringAxleAngle + dir*dt*spec.steeringAxleAngleSpeed, spec.steeringAxleTargetAngle)
366 end
367
368 if spec.steeringAxleTargetNode ~= nil then
369 local angle = MathUtil.clamp(spec.steeringAxleAngle, spec.steeringAxleAngleMinRot, spec.steeringAxleAngleMaxRot)
370 setRotation(spec.steeringAxleTargetNode, 0, angle, 0)
371 self:setMovingToolDirty(spec.steeringAxleTargetNode)
372 end
373 end
374 end
375 end
```

loadInputAttacherJoint

Description

Called on loading

Definition

loadInputAttacherJoint(table savegame)

Arguments

table savegame savegame

Code

```
380 function Attachable:loadInputAttacherJoint(xmlFile, key, inputAttacherJoint, index)
381 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName, key .. "#index", key .. "#node") -- FS17 to FS19
382 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName, key .. "#indexVisual", key .. "#nodeVisual") -- FS17 to FS19
383 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName, key .. "#ptoInputNode", "vehicle.powerTakeOffs.input") -- FS17 to FS19
384 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName, key .. "#lowerDistanceToGround", key..".distanceToGround#lower") -- FS17 to FS19
385 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName, key .. "#upperDistanceToGround", key..".distanceToGround#upper") -- FS17 to FS19
386
387 local node = I3DUtil.indexToObject(self.components, getXMLString(xmlFile, key .. "#node"), self.i3dMappings)
388 if node ~= nil then
389 inputAttacherJoint.node = node
```

```
390 inputAttacherJoint.heightNode = I3DUtil.indexToObject(self.components, getXMLString(xmlFile, key .. ".heightNode#node"), self.i3dMappings)
391 if inputAttacherJoint.heightNode ~= nil then
392 inputAttacherJoint.heightNodeOffset = {localToLocal(inputAttacherJoint.heightNode, node, 0, 0, 0)}
393 end
394
395 local jointTypeStr = getXMLString(xmlFile, key .. "#jointType")
396 local jointType
397 if jointTypeStr ~= nil then
398 jointType = AttacherJoints.jointTypeNameToInt[jointTypeStr]
399 if jointType == nil then
400 g_logManager:xmlWarning(self.configFileName, "Invalid jointType '%s' for inputAttacherJoint '%s'!", tostring(jointTypeStr), key)
401 end
402 else
403 g_logManager:xmlWarning(self.configFileName, "Missing jointType for inputAttacherJoint '%s'!", key)
404 end
405 if jointType == nil then
406 local needsTrailerJoint = Utils.getNoNil(getXMLBool(xmlFile, key .. "#needsTrailerJoint"), false)
407 local needsLowTrailerJoint = Utils.getNoNil(getXMLBool(xmlFile, key .. "#needsLowJoint"), false)
408 if needsTrailerJoint then
409 if needsLowTrailerJoint then
410 jointType = AttacherJoints.JOINTTYPE_TRAILERLOW
411 else
412 jointType = AttacherJoints.JOINTTYPE_TRAILER
413 end
414 else
415 jointType = AttacherJoints.JOINTTYPE_IMPLEMENT
416 end
417 end
418 inputAttacherJoint.jointType = jointType
419
420 inputAttacherJoint.jointOrigTrans = { getTranslation(inputAttacherJoint.node) }
421 inputAttacherJoint.topReferenceNode = I3DUtil.indexToObject(self.components, getXMLString(xmlFile, key .. "#topReferenceNode"), self.i3dMappings)
```

```
422 inputAttacherJoint.rootNode =
    Utils.getNoNil(I3DUtil.indexToObject(self.components,
    getXMLString(xmlFile, key .. "#rootNode"), self.i3dMappings),
    self.components[1].node)
423 inputAttacherJoint.rootNodeBackup = inputAttacherJoint.rootNode
424 inputAttacherJoint.allowsDetaching =
    Utils.getNoNil(getXMLBool(xmlFile, key .. "#allowsDetaching"),
    true)
425 inputAttacherJoint.fixedRotation =
    Utils.getNoNil(getXMLBool(xmlFile, key .. "#fixedRotation"),
    false)
426 inputAttacherJoint.hardAttach =
    Utils.getNoNil(getXMLBool(xmlFile, key .. "#hardAttach"), false)
427 if inputAttacherJoint.hardAttach and #self.components > 1 then
428 g_logManager:xmlWarning(self.configFileName, "hardAttach only
    available for single component vehicles! InputAttacherJoint
    '%s'!", key)
429 inputAttacherJoint.hardAttach = false
430 end
431 inputAttacherJoint.visualNode =
    I3DUtil.indexToObject(self.components, getXMLString(xmlFile, key
    .. "#nodeVisual"), self.i3dMappings)
432 if inputAttacherJoint.hardAttach and
    inputAttacherJoint.visualNode ~= nil then
433 inputAttacherJoint.visualNodeData = {
434 parent = getParent(inputAttacherJoint.visualNode),
435 translation = { getTranslation(inputAttacherJoint.visualNode) },
436 rotation = { getRotation(inputAttacherJoint.visualNode) },
437 index = getChildIndex(inputAttacherJoint.visualNode)
438 }
439 end
440
441
442 if jointType == AttacherJoints.JOINTTYPE_IMPLEMENT or jointType
    == AttacherJoints.JOINTTYPE_CUTTER then
443 if getXMLFloat(xmlFile, key .. ".distanceToGround#lower") == nil
    then
444 g_logManager:xmlWarning(self.configFileName, "Missing
    '.distanceToGround#lower' for inputAttacherJoint '%s'!", key)
445 end
446 if getXMLFloat(xmlFile, key .. ".distanceToGround#upper") == nil
    then
447 g_logManager:xmlWarning(self.configFileName, "Missing
    '.distanceToGround#upper' for inputAttacherJoint '%s'!", key)
448 end
```

```
449 end
450 inputAttacherJoint.lowerDistanceToGround =
    Utils.getNoNil(getXMLFloat(xmlFile, key ..
    ".distanceToGround#lower"), 0.7)
451 inputAttacherJoint.upperDistanceToGround =
    Utils.getNoNil(getXMLFloat(xmlFile, key ..
    ".distanceToGround#upper"), 1.0)
452 if inputAttacherJoint.lowerDistanceToGround >
    inputAttacherJoint.upperDistanceToGround then
453 g_logManager:xmlWarning(self.configFileName,
    "distanceToGround#lower may not be larger than
    distanceToGround#upper for inputAttacherJoint '%s'. Switching
    values!", key)
454 local copy = inputAttacherJoint.lowerDistanceToGround
455 inputAttacherJoint.lowerDistanceToGround =
    inputAttacherJoint.upperDistanceToGround
456 inputAttacherJoint.upperDistanceToGround = copy
457 end
458
459 inputAttacherJoint.lowerRotationOffset =
    math.rad(Utils.getNoNil(getXMLFloat(xmlFile, key ..
    "#lowerRotationOffset"), 0))
460
461
462 local defaultUpperRotationOffset = 8
463 if jointType == AttacherJoints.JOINTTYPE_CUTTER or jointType ==
    AttacherJoints.JOINTTYPE_WHEELLOADER or jointType ==
    AttacherJoints.JOINTTYPE_TELEHANDLER or jointType ==
    AttacherJoints.JOINTTYPE_FRONTLOADER then
464 defaultUpperRotationOffset = 0
465 end
466
467 inputAttacherJoint.upperRotationOffset =
    math.rad(Utils.getNoNil(getXMLFloat(xmlFile, key ..
    "#upperRotationOffset"), defaultUpperRotationOffset))
468
469 inputAttacherJoint.allowsJointRotLimitMovement =
    Utils.getNoNil(getXMLBool(xmlFile, key ..
    "#allowsJointRotLimitMovement"), true)
470 inputAttacherJoint.allowsJointTransLimitMovement =
    Utils.getNoNil(getXMLBool(xmlFile, key ..
    "#allowsJointTransLimitMovement"), true)
471
472 inputAttacherJoint.needsToolbar =
    Utils.getNoNil(getXMLBool(xmlFile, key .. "#needsToolbar"),
    false)
```

```
473 if inputAttacherJoint.needsToolbar and jointType ~= AttacherJoints.JOINTTYPE_IMPLEMENT then
474 g_logManager:xmlWarning(self.configFileName, "'needsToolbar' requires jointType 'implement' for inputAttacherJoint '%s'!", key)
475 inputAttacherJoint.needsToolbar = false
476 end
477
478 inputAttacherJoint.steeringBarLeftNode = I3DUtil.indexToObject(self.components, getXMLString(xmlFile, key .. "#steeringBarLeftNode"), self.i3dMappings)
479 inputAttacherJoint.steeringBarRightNode = I3DUtil.indexToObject(self.components, getXMLString(xmlFile, key .. "#steeringBarRightNode"), self.i3dMappings)
480
481 --load joint limit scales
482 local x, y, z = StringUtil.getVectorFromString(getXMLString(xmlFile, key .. "#upperRotLimitScale"))
483 inputAttacherJoint.upperRotLimitScale = { Utils.getNoNil(x, 0), Utils.getNoNil(y, 0), Utils.getNoNil(z, 0) }
484 local x, y, z = StringUtil.getVectorFromString(getXMLString(xmlFile, key .. "#lowerRotLimitScale"))
485 if jointType == AttacherJoints.JOINTTYPE_IMPLEMENT then
486 inputAttacherJoint.lowerRotLimitScale = { Utils.getNoNil(x, 0), Utils.getNoNil(y, 0), Utils.getNoNil(z, 1) }
487 else
488 inputAttacherJoint.lowerRotLimitScale = { Utils.getNoNil(x, 1), Utils.getNoNil(y, 1), Utils.getNoNil(z, 1) }
489 end
490
491 local x, y, z = StringUtil.getVectorFromString(getXMLString(xmlFile, key .. "#upperTransLimitScale"))
492 inputAttacherJoint.upperTransLimitScale = { Utils.getNoNil(x, 0), Utils.getNoNil(y, 0), Utils.getNoNil(z, 0) }
493 local x, y, z = StringUtil.getVectorFromString(getXMLString(xmlFile, key .. "#lowerTransLimitScale"))
494 inputAttacherJoint.lowerTransLimitScale = { Utils.getNoNil(x, 0), Utils.getNoNil(y, 1), Utils.getNoNil(z, 0) }
495
496 local x, y, z = StringUtil.getVectorFromString(getXMLString(xmlFile, key.."#rotLimitSpring"))
```

```
497 inputAttacherJoint.rotLimitSpring = { Utils.getNoNil(x, 0),
    Utils.getNoNil(y, 0), Utils.getNoNil(z, 0) }
498 local x, y, z =
    StringUtil.getVectorFromString(getXMLString(xmlFile,
    key.."#rotLimitDamping"))
499 inputAttacherJoint.rotLimitDamping = { Utils.getNoNil(x, 1),
    Utils.getNoNil(y, 1), Utils.getNoNil(z, 1) }
500 local x, y, z =
    StringUtil.getVectorFromString(getXMLString(xmlFile,
    key.."#rotLimitForceLimit"))
501 inputAttacherJoint.rotLimitForceLimit = { Utils.getNoNil(x, -1),
    Utils.getNoNil(y, -1), Utils.getNoNil(z, -1) }
502
503 local x, y, z =
    StringUtil.getVectorFromString(getXMLString(xmlFile,
    key.."#transLimitSpring"))
504 inputAttacherJoint.transLimitSpring = { Utils.getNoNil(x, 0),
    Utils.getNoNil(y, 0), Utils.getNoNil(z, 0) }
505 local x, y, z =
    StringUtil.getVectorFromString(getXMLString(xmlFile,
    key.."#transLimitDamping"))
506 inputAttacherJoint.transLimitDamping = { Utils.getNoNil(x, 1),
    Utils.getNoNil(y, 1), Utils.getNoNil(z, 1) }
507 local x, y, z =
    StringUtil.getVectorFromString(getXMLString(xmlFile,
    key.."#transLimitForceLimit"))
508 inputAttacherJoint.transLimitForceLimit = { Utils.getNoNil(x, -
    1), Utils.getNoNil(y, -1), Utils.getNoNil(z, -1) }
509
510 inputAttacherJoint.attacherHeight = getXMLFloat(xmlFile, key ..
    "#attacherHeight")
511 if inputAttacherJoint.attacherHeight == nil then
512 if jointType == AttacherJoints.JOINTTYPE_TRAILER then
513 inputAttacherJoint.attacherHeight = 0.9
514 elseif jointType == AttacherJoints.JOINTTYPE_TRAILERLOW then
515 inputAttacherJoint.attacherHeight = 0.55
516 end
517 end
518
519 local defaultNeedsLowering = true
520 local defaultAllowsLowering = false
521 if inputAttacherJoint.jointType ==
    AttacherJoints.JOINTTYPE_TRAILER or inputAttacherJoint.jointType
    == AttacherJoints.JOINTTYPE_TRAILERLOW then
522 defaultNeedsLowering = false
```

```
523 end
524 if inputAttacherJoint.jointType ~= AttacherJoints.JOINTTYPE_TRAILER and inputAttacherJoint.jointType ~= AttacherJoints.JOINTTYPE_TRAILERLOW then
525 defaultAllowsLowering = true
526 end
527 inputAttacherJoint.needsLowering = Utils.getNoNil(getXMLBool(xmlFile, key.. "#needsLowering"), defaultNeedsLowering)
528 inputAttacherJoint.allowsLowering = Utils.getNoNil(getXMLBool(xmlFile, key.. "#allowsLowering"), defaultAllowsLowering)
529 inputAttacherJoint.isDefaultLowered = Utils.getNoNil(getXMLBool(xmlFile, key.. "#isDefaultLowered"), false)
530 inputAttacherJoint.useFoldingLoweredState = Utils.getNoNil(getXMLBool(xmlFile, key.. "#useFoldingLoweredState"), false)
531 inputAttacherJoint.forceSelection = Utils.getNoNil(getXMLBool(self.xmlFile, key.."#forceSelectionOnAttach"), true)
532
533 inputAttacherJoint.dependentAttacherJoints = {}
534 local k = 0
535 while true do
536 local dependentKey = string.format(key .. ".dependentAttacherJoint(%d)", k)
537 if not hasXMLProperty(xmlFile, dependentKey) then
538 break
539 end
540 local attacherJointIndex = getXMLInt(xmlFile, dependentKey.."#attacherJointIndex")
541 if attacherJointIndex ~= nil then
542 table.insert(inputAttacherJoint.dependentAttacherJoints, attacherJointIndex)
543 end
544 k = k + 1
545 end
546
547 -- reset values if hardAttach is active
548 if inputAttacherJoint.hardAttach then
549 inputAttacherJoint.needsLowering = false
550 inputAttacherJoint.allowsLowering = false
551 inputAttacherJoint.isDefaultLowered = false
```

```
552 inputAttacherJoint.upperRotationOffset = 0
553 end
554
555 inputAttacherJoint.changeObjects = {}
556 ObjectChangeUtil.loadObjectChangeFromXML(xmlFile, key,
    inputAttacherJoint.changeObjects, self.components, self)
557
558 return true
559 end
560
561 return false
562 end
```

getAllowsLowering

Description

Returns true if tool can be lowered

Definition

getAllowsLowering()

Return Values

boolean detachAllowed detach is allowed

string warning [optional] warning text to display

Code

```
605 function Attachable:getAllowsLowering()
606 local inputAttacherJoint = self:getActiveInputAttacherJoint()
607 if inputAttacherJoint ~= nil then
608 if not inputAttacherJoint.allowsLowering then
609 return false, nil
610 end
611 end
612
613 return true, nil
614 end
```

getIsImplementChainLowered

Description

Returns true if vehicle itself is lowered and all parent vehicles are lowered until the first vehicle is not attachable (e.g. all implements until a tractor is in the chain)

Definition

getIsImplementChainLowered(boolean defaultIsLowered)

Arguments

boolean defaultIsLowered default value if lowering is not allowed

Return Values

boolean isLowered implement chain is lowered

Code

```
620 function Attachable:getIsImplementChainLowered(defaultIsLowered)
621 if not self:getIsLowered(defaultIsLowered) then
622 return false
623 end
624
625 local attacherVehicle = self:getAttacherVehicle()
626 if attacherVehicle ~= nil then
627 if attacherVehicle.getAllowsLowering ~= nil then
628 if attacherVehicle:getAllowsLowering() then
629 if not attacherVehicle:getIsImplementChainLowered(defaultIsLowered) then
630 return false
631 end
632 end
633 end
634 end
635
636 return true
637 end
```

getIsInWorkPosition

Description

Returns true if it is in work position

Definition

getIsInWorkPosition()

Return Values

boolean inWorkPosition is in work position

Code

```
642 function Attachable:getIsInWorkPosition()
643 return true
644 end
```

getAttachbleAirConsumerUsage

Description

Returns air consumer usage

Definition

getAttachbleAirConsumerUsage()

Return Values

float usage usage

Code

```
649 function Attachable:getAttachbleAirConsumerUsage()
650 return self.spec_attachable.airConsumerUsage
651 end
```

isDetachAllowed

Description

Returns true if detach is allowed

Definition

isDetachAllowed()

Return Values

boolean detachAllowed detach is allowed

string warning [optional] warning text to display

Code

```
function Attachable:isDetachAllowed()
local spec = self.spec_attachable
if spec.attacherJoint ~= nil then
if spec.attacherJoint.allowsDetaching == false then
return false, nil
end
end

-- block detach while the attaching is still in progress
local attacherVehicle = self:getAttacherVehicle()
if attacherVehicle ~= nil then
local implement = attacherVehicle:getImplementByObject(self)
if implement ~= nil and implement.attachingIsInProgress then
return false
end
end

return true, nil
end
```

getIsInputAttacherActive

Description

Returns true if input attacher is active and can be used to attach

Definition

getIsInputAttacherActive(table inputAttacherJoint)

Arguments

table inputAttacherJoint input attacher joint

Return Values

boolean isActive input attacher is active

Code

```
function Attachable:getIsInputAttacherActive(inputAttacherJoint)
return true
end
```

getSteeringAxleBaseVehicle

Description

Returns vehicle used to calculate steering axle

Definition

getSteeringAxleBaseVehicle()

Code

```
687 function Attachable:getSteeringAxleBaseVehicle()
688 local spec = self.spec_attachable
689
690 if spec.steeringAxleUseSuperAttachable then
691 if spec.attacherVehicle ~= nil then
692 return spec.attacherVehicle.attacherVehicle
693 end
694 end
695
696 return spec.attacherVehicle
697 end
```

attachableAddToolCameras

Description

Add tool cameras to root attacher vehicle

Definition

attachableAddToolCameras()

Code

```
701 function Attachable:attachableAddToolCameras()
702 local spec = self.spec_attachable
703
704 if #spec.toolCameras > 0 then
705 local rootAttacherVehicle = self:getRootVehicle()
706 if rootAttacherVehicle ~= nil then
707 if rootAttacherVehicle.addToolCameras ~= nil then
708 rootAttacherVehicle:addToolCameras(spec.toolCameras)
709 end
710 end
711 end
712 end
```

attachableRemoveToolCameras

Description

Remove tool cameras from root attacher vehicle

Definition

attachableRemoveToolCameras()

Code

```
716 function Attachable:attachableRemoveToolCameras()
717 local spec = self.spec_attachable
```

```
718
719 if #spec.toolCameras > 0 then
720 local rootAttacherVehicle = self:getRootVehicle()
721 if rootAttacherVehicle ~= nil then
722 if rootAttacherVehicle.removeToolCameras ~= nil then
723 rootAttacherVehicle:removeToolCameras(spec.toolCameras)
724 end
725 end
726 end
727 end
```

preAttach

Description

Called before vehicle gets attached

Definition

preAttach(table attacherVehicle, integer inputAttacherJointDescIndex)

Arguments

table	attacherVehicle	attacher vehicle
integer	inputAttacherJointDescIndex	index of input attacher joint

Code

```
733 function Attachable:preAttach(attacherVehicle,
    inputJointDescIndex, jointDescIndex)
734 local spec = self.spec_attachable
735
736 spec.attacherVehicle = attacherVehicle
737 spec.attacherJoint =
    spec.inputAttacherJoints[inputJointDescIndex]
738 spec.inputAttacherJointDescIndex = inputJointDescIndex
739
740 SpecializationUtil.raiseEvent(self, "onPreAttach",
    attacherVehicle, inputJointDescIndex, jointDescIndex)
741 end
```

postAttach

Description

Called if vehicle gets attached

Definition

postAttach(table attacherVehicle, integer inputJointDescIndex, integer jointDescIndex)

Arguments

table	attacherVehicle	attacher vehicle
integer	inputJointDescIndex	index of input attacher joint
integer	jointDescIndex	index of attacher joint it gets attached to

Code

```
748 function Attachable:postAttach(attacherVehicle, inputJointDescIndex,
    jointDescIndex)
749 local spec = self.spec_attachable
750
751 -- only activate tool if root vehicle is controlled (activated). We don'
    want to activate a tool that is attached during loading
752 local rootVehicle = self:getRootVehicle()
753 if rootVehicle ~= nil and rootVehicle.getIsControlled ~= nil and
    rootVehicle:getIsControlled() then
754 self:activate()
755 end
756
757 if self.setLightsTypesMask ~= nil then
758 local lightsSpecAttacherVehicle = attacherVehicle.spec_lights
759 if lightsSpecAttacherVehicle ~= nil then
760 self:setLightsTypesMask(lightsSpecAttacherVehicle.lightsTypesMask, true,
    true)
761 self:setBeaconLightsVisibility(lightsSpecAttacherVehicle.beaconLightsAct
    true, true)
762 self:setTurnLightState(lightsSpecAttacherVehicle.turnLightState, true, t
763 end
764 end
765
766 spec.attachTime = g_currentMission.time
767
768 if spec.supportAnimation ~= nil and self.playAnimation ~= nil then
769 self:playAnimation(spec.supportAnimation, -1, nil, true)
770 if self.propertyState == Vehicle.PROPERTY_STATE_SHOP_CONFIG then
771 AnimatedVehicle.updateAnimationByName(self, spec.supportAnimation, 9999
772 end
773 end
774
775 self:attachableAddToolCameras()
776
777 ObjectChangeUtil.setObjectChanges(spec.attacherJoint.changeObjects, true
    self, self.setMovingToolDirty)
778
779 local jointDesc =
    attacherVehicle:getAttacherJointByJointDescIndex(jointDescIndex)
780 if jointDesc.steeringBarLeftNode ~= nil and
    spec.attacherJoint.steeringBarLeftMovingPart ~= nil then
781 for _,movingPart in pairs(self.spec_cylindered.movingParts) do
```

```
782 if movingPart.referencePoint == spec.attacherJoint.steeringBarLeftMovingPart.referencePoint and movingPa ~= spec.attacherJoint.steeringBarLeftMovingPart then
783 movingPart.referencePoint = jointDesc.steeringBarLeftNode
784 end
785 end
786 spec.attacherJoint.steeringBarLeftMovingPart.referencePoint = jointDesc.steeringBarLeftNode
787 end
788 if jointDesc.steeringBarRightNode ~= nil and spec.attacherJoint.steeringBarRightMovingPart ~= nil then
789 for _,movingPart in pairs(self.spec_cylindered.movingParts) do
790 if movingPart.referencePoint == spec.attacherJoint.steeringBarRightMovingPart.referencePoint and movingP ~= spec.attacherJoint.steeringBarRightMovingPart then
791 movingPart.referencePoint = jointDesc.steeringBarRightNode
792 end
793 end
794 spec.attacherJoint.steeringBarRightMovingPart.referencePoint = jointDesc.steeringBarRightNode
795 end
796
797 SpecializationUtil.raiseEvent(self, "onPostAttach", attacherVehicle, inputJointDescIndex, jointDescIndex)
798 end
```

setLowered

Description

Set attachables lowering state

Definition

setLowered(boolean lowered)

Arguments

boolean lowered attachable is lowered

Code

```
837 function Attachable:setLowered(lowered)
838 local spec = self.spec_attachable
839
840 if spec.lowerAnimation ~= nil and self.playAnimation ~= nil then
841 if lowered then
842 self:playAnimation(spec.lowerAnimation, spec.lowerAnimationSpeed, nil, true)
843 else
844 self:playAnimation(spec.lowerAnimation, -spec.lowerAnimationSpeed, nil, true)
```

```
845 end
846 end
847
848 if spec.attacherJoint ~= nil then
849 for _, dependentAttacherJointIndex in
    pairs(spec.attacherJoint.dependentAttacherJoints) do
850 if self.getAttacherJoints ~= nil then
851 local attacherJoints = self:getAttacherJoints()
852 if attacherJoints[dependentAttacherJointIndex] ~= nil then
853 self:setJointMoveDown(dependentAttacherJointIndex, lowered, true)
854 else
855 g_logManager:xmlWarning(self.configFileName, "Failed to lower
    dependent attacher joint index '%d', No attacher joint defined!",
    dependentAttacherJointIndex)
856 end
857 else
858 g_logManager:xmlWarning(self.configFileName, "Failed to lower
    dependent attacher joint index '%d', AttacherJoint specialization
    is missing!", dependentAttacherJointIndex)
859 end
860 end
861 end
862
863 SpecializationUtil.raiseEvent(self, "onSetLowered", lowered)
864 end
```

setLoweredAll

Description

Set attachables lowering all state

Definition

setLoweredAll(boolean doLowering)

Arguments

boolean doLowering do lowering

Code

```
870 function Attachable:setLoweredAll(doLowering, jointDescIndex)
871 self:getAttacherVehicle():handleLowerImplementByAttacherJointIndex(joint
    doLowering)
872
873 SpecializationUtil.raiseEvent(self, "onSetLoweredAll", doLowering, joint
874 end
```

onDeactivate

Description

Called on deactivate

Definition

onDeactivate()

Code

```
903  function Attachable:onDeactivate()
904  if self.brake ~= nil then
905  local spec = self.spec_attachable
906  self:brake(spec.brakeForce, true)
907  end
908  end
```

getIsOperating

Description

Returns if vehicle is operating

Definition

getIsOperating()

Return Values

boolean isOperating is operating

Code

```
946  function Attachable:getIsOperating(superFunc)
947  local spec = self.spec_attachable
948
949  local isOperating = superFunc(self)
950
951  if not isOperating and spec.attacherVehicle ~= nil then
952  isOperating = spec.attacherVehicle:getIsOperating()
953  end
954
955  return isOperating
956  end
```

getCanAIImplementContinueWork

Description

Returns true if vehicle is ready for ai work

Definition

getCanAIImplementContinueWork()

Return Values

boolean isReady is ready for ai work

Code

```
1009  function Attachable:getCanAIImplementContinueWork(superFunc)
1010  local spec = self.spec_attachable
1011
1012  local isReady = true
1013  if spec.lowerAnimation ~= nil then
1014  local time = self:getAnimationTime(spec.lowerAnimation)
```

```
1015  isReady = time == 1 or time == 0
1016  end
1017
1018  local jointDesc =
      spec.attacherVehicle:getAttacherJointDescFromObject(self)
1019  if jointDesc.allowsLowering and self:getAINeedsLowering() then
1020  if jointDesc.moveDown then
1021  isReady = (jointDesc.moveAlpha == jointDesc.lowerAlpha or
      jointDesc.moveAlpha == jointDesc.upperAlpha) and isReady
1022  end
1023  end
1024
1025  return isReady
1026  end
```

AttacherJointControl

Description

Specialization to control the attacherJoint height

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17  function
    AttacherJointControl.prerequisitesPresent(specializations)
18  return SpecializationUtil.hasSpecialization(Attachable,
    specializations)
19  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
51  function AttacherJointControl:onLoad(savegame)
52  local spec = self.spec_attacherJointControl
53
```

```
54  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.attacherJointControl.control1",
    "vehicle.attacherJointControl.control with #controlFunction
    'controlAttacherJointHeight'") -- FS17
55  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.attacherJointControl.control2",
    "vehicle.attacherJointControl.control with #controlFunction
    'controlAttacherJointTilt'") -- FS17
56
57  local baseKey = "vehicle.attacherJointControl"
58  spec.maxTiltAngle = Utils.getNoNilRad(getXMLFloat(self.xmlFile,
    baseKey .. "#maxTiltAngle"), math.rad(25))
59  spec.heightTargetAlpha = -1
60
61  spec.controls = {}
62  spec.nameToControl = {}
63  local i = 0
64  while true do
65  local key = string.format("%s.control(%d)", baseKey, i)
66  if not hasXMLProperty(self.xmlFile, key) then
67  break
68  end
69
70  local control = {}
71
72  local controlFunc = getXMLString(self.xmlFile, key ..
    "#controlFunction")
73  if controlFunc ~= nil and self[controlFunc] ~= nil then
74  control.func = self[controlFunc]
75
76  if control.func == self.controlAttacherJointHeight then
77  spec.heightController = control
78  end
79  else
80  g_logManager:xmlWarning(self.configFileName, "Unknown control
    function '%s' for attacher joint control '%s'",
    tostring(controlFunc), key)
81  break
82  end
83
84  local actionBindingName = getXMLString(self.xmlFile, key ..
    "#controlAxis")
```

```
85  if actionBindingName ~= nil and InputAction[actionBindingName] ~=
    nil then
86  control.controlAction = InputAction[actionBindingName]
87  else
88  g_logManager:xmlWarning(self.configFileName, "Unknown control axis
    '%s' for attacher joint control '%s'",
    tostring(actionBindingName), key)
89  break
90  end
91
92  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, key.."#controlAxisIcon", key.."#iconName") --
    FS17 to FS19
93
94  local iconName = getXMLString(self.xmlFile, key .. "#iconName") or
    ""
95  if InputHelpElement.AXIS_ICON[iconName] == nil then
96  -- add the mod name as a prefix to match axis icon loading name
    collision avoidance
97  iconName = (self.customEnvironment or "") .. iconName
98  end
99
100 control.axisActionIcon = iconName
101
102 control.invertAxis = Utils.getNoNil( getXMLString(self.xmlFile,
    key .. "#invertControlAxis"), false )
103 control.mouseSpeedFactor = Utils.getNoNil(
    getXMLFloat(self.xmlFile, key .. "#mouseSpeedFactor"), 1.0 )
104 control.moveAlpha = 0
105
106 spec.nameToControl[actionBindingName] = control
107 table.insert(spec.controls, control)
108
109 i = i + 1
110 end
111
112 if self.isClient then
113 spec.lastMoveTime = 0
114
115 spec.samples = {}
116 spec.samples.hydraulic =
    g_soundManager:loadSampleFromXML(self.xmlFile, baseKey..".sounds",
```

```
    "hydraulic", self.baseDirectory, self.components, 0,
    AudioGroup.VEHICLE, self.i3dMappings, self)
end

spec.jointDesc = nil
spec.dirtyFlag = self:getNextDirtyFlag()
end

function AttacherJointControl:onDelete()
if self.isClient then
local spec = self.spec_attacherJointControl
g_soundManager:deleteSample(spec.samples.hydraulic)
end
end

---Called on client side on join
-- @param integer streamId streamId
-- @param integer connection connection
-- @includeCode
function AttacherJointControl:onReadStream(streamId, connection)
if connection:getIsServer() then
if streamReadBool(streamId) then
local spec = self.spec_attacherJointControl
for _, control in ipairs(spec.controls) do
local moveAlpha = streamReadFloat32(streamId)
self:controlAttacherJoint(control, moveAlpha)
end
end
end
end

---Called on server side on join
-- @param integer streamId streamId
-- @param integer connection connection
-- @includeCode
function AttacherJointControl:onWriteStream(streamId, connection)
if not connection:getIsServer() then
local spec = self.spec_attacherJointControl
if streamWriteBool(streamId, spec.jointDesc ~= nil) then
for _, control in ipairs(spec.controls) do
```

```
155 streamWriteFloat32(streamId, control.moveAlpha)
156 end
157 end
158 end
159 end
160
161 ---Called on on update
162 -- @param integer streamId stream ID
163 -- @param integer timestamp timestamp
164 -- @param table connection connection
165 -- @includeCode
166 function AttacherJointControl:onReadUpdateStream(streamId,
    timestamp, connection)
167 if not connection:getIsServer() then
168 if streamReadBool(streamId) then
169 local spec = self.spec_attacherJointControl
170 for _, control in ipairs(spec.controls) do
171 local moveAlpha = streamReadFloat32(streamId)
172 self:controlAttacherJoint(control, moveAlpha)
173 end
174 end
175 end
176 end
177
178 ---Called on on update
179 -- @param integer streamId stream ID
180 -- @param table connection connection
181 -- @param integer dirtyMask dirty mask
182 -- @includeCode
183 function AttacherJointControl:onWriteUpdateStream(streamId,
    connection, dirtyMask)
184 if connection:getIsServer() then
185 local spec = self.spec_attacherJointControl
186 if streamWriteBool(streamId, bitAND(dirtyMask, spec.dirtyFlag) ~=
    0) then
187 for _, control in ipairs(spec.controls) do
188 streamWriteFloat32(streamId, control.moveAlpha)
189 end
190 end
191 end
192 end
```

```
193
194 ---Called on update
195 -- @param float dt time since last call in ms
196 -- @param boolean isActiveForInput true if vehicle is active for
    input
197 -- @param boolean isSelected true if vehicle is selected
198 -- @includeCode
199 function AttacherJointControl:onUpdate(dt, isActiveForInput,
    isSelected)
200 local spec = self.spec_attacherJointControl
201 local control = spec.heightController
202
203 --update attacher joint height by target alpha
204 if control ~= nil then
205 if spec.jointDesc ~= nil then
206 if spec.heightTargetAlpha ~= -1 then
207 local diff = spec.heightTargetAlpha - control.moveAlpha + 0.0001
208 local moveTime = diff / (spec.jointDesc.upperAlpha -
    spec.jointDesc.lowerAlpha) * spec.jointDesc.moveTime
209 local moveStep = dt / moveTime * diff
210 if diff > 0 then
211 moveStep = -moveStep
212 end
213 local newAlpha = control.moveAlpha + moveStep
214 self:controlAttacherJoint(control, newAlpha)
215 if math.abs(spec.heightTargetAlpha - newAlpha) < 0.01 then
216 spec.heightTargetAlpha = -1
217 end
218 end
219 end
220 end
221
222 if spec.lastMoveTime + 100 > g_time then
223 if not g_soundManager:getIsSamplePlaying(spec.samples.hydraulic)
    then
224 g_soundManager:playSample(spec.samples.hydraulic)
225 end
226 else
227 if g_soundManager:getIsSamplePlaying(spec.samples.hydraulic) then
228 g_soundManager:stopSample(spec.samples.hydraulic)
229 end
```

```
230 end
231 end
232
233 ---Updates attacher joint move alpha and update joint frame
234 -- @param table control control
235 -- @param float moveAlpha move alpha
236 -- @includeCode
237 function AttacherJointControl:controlAttacherJoint(control,
    moveAlpha)
238 local spec = self.spec_attacherJointControl
239 local jointDesc = spec.jointDesc
240
241 if jointDesc ~= nil then
242 moveAlpha = control.func(self, moveAlpha)
243
244 local attacherVehicle = self:getAttacherVehicle()
245 attacherVehicle:updateAttacherJointRotation(jointDesc, self)
246 if self.isServer and jointDesc.jointIndex ~= 0 then
247 setJointFrame(jointDesc.jointIndex, 0, jointDesc.jointTransform)
248 end
249 end
250
251 spec.lastMoveTime = g_time
252
253 control.moveAlpha = moveAlpha
254 self:raiseDirtyFlags(spec.dirtyFlag)
255 end
256
257 ---Updates attacher joint height by given mve alpha value
258 -- @param float moveAlpha move alpha
259 -- @return float moveAlpha limited move alpha
260 -- @includeCode
261 function
    AttacherJointControl:controlAttacherJointHeight(moveAlpha)
262 local spec = self.spec_attacherJointControl
263 local jointDesc = spec.jointDesc
264
265 if moveAlpha == nil then
266 moveAlpha = jointDesc.moveAlpha
267 end
268
```

```
269 moveAlpha = MathUtil.clamp(moveAlpha, jointDesc.upperAlpha,
    jointDesc.lowerAlpha)
270 if jointDesc.rotationNode ~= nil then
271 setRotation(jointDesc.rotationNode,
    MathUtil.vector3ArrayLerp(jointDesc.upperRotation,
    jointDesc.lowerRotation, moveAlpha))
272 end
273 if jointDesc.rotationNode2 ~= nil then
274 setRotation(jointDesc.rotationNode2,
    MathUtil.vector3ArrayLerp(jointDesc.upperRotation2,
    jointDesc.lowerRotation2, moveAlpha))
275 end
276
277 spec.lastHeightAlpha = moveAlpha
278
279 return moveAlpha
280 end
281
282 ---Updates attacher joint tilt by given mve alpha value
283 -- @param float moveAlpha move alpha
284 -- @return float moveAlpha limited move alpha
285 -- @includeCode
286 function AttacherJointControl:controlAttacherJointTilt(moveAlpha)
287 local spec = self.spec_attacherJointControl
288
289 if moveAlpha == nil then
290 moveAlpha = 0.5
291 end
292
293 moveAlpha = MathUtil.clamp(moveAlpha, 0, 1)
294 local angle = spec.maxTiltAngle * -(moveAlpha - 0.5)
295
296 spec.jointDesc.upperRotationOffset =
    spec.jointDesc.upperRotationOffsetBackup + angle
297 spec.jointDesc.lowerRotationOffset =
    spec.jointDesc.lowerRotationOffsetBackup + angle
298
299 return moveAlpha
300 end
301
302 ---Returns direction boolean for lifting and lowering the
    implement by button press
```

```
303 -- @return boolean direction direction
304 -- @includeCode
305 function AttacherJointControl:getControlAttacherJointDirection()
306 local spec = self.spec_attacherJointControl
307 if spec.heightTargetAlpha ~= -1 then
308 return spec.heightTargetAlpha == spec.jointDesc.upperAlpha
309 end
310
311 local lastAlpha = spec.heightController.moveAlpha
312 return math.abs(lastAlpha - spec.jointDesc.lowerAlpha) >
    math.abs(lastAlpha - spec.jointDesc.upperAlpha)
313 end
314
315 ---Called on loading
316 -- @param table savegame savegame
317 -- @includeCode
318 function AttacherJointControl:loadInputAttacherJoint(superFunc,
    xmlFile, key, inputAttacherJoint, i)
319 if not superFunc(self, xmlFile, key, inputAttacherJoint, i) then
320 return false
321 end
322
323 inputAttacherJoint.isControllable =
    Utils.getNoNil(getXMLBool(xmlFile, key .. "#isControllable"),
    false)
324
325 return true
326 end
327
328 function
    AttacherJointControl:registerLoweringActionEvent(superFunc,
    actionEventsTable, inputAction, target, callback, triggerUp,
    triggerDown, triggerAlways, startActive, callbackState,
    customIconName)
329 local spec = self.spec_attacherJointControl
330 if spec.heightController then
331 local _, actionEventId = self:addActionEvent(actionEventsTable,
    InputAction.LOWER_IMPLEMENT, self,
    AttacherJointControl.actionEventAttacherJointControlSetPoint,
    triggerUp, triggerDown, triggerAlways, startActive, callbackState,
    customIconName)
332 g_inputBinding:setActionEventTextPriority(actionEventId,
    GS_PRIO_HIGH)
```

```
333
334 if inputAction == InputAction.LOWER_IMPLEMENT then
335 return
336 end
337 end
338
339 superFunc(self, actionEventsTable, inputAction, target, callback,
    triggerUp, triggerDown, triggerAlways, startActive, callbackState,
    customIconName)
340 end
341
342 function
    AttacherJointControl:getLoweringActionEventState(superFunc)
343 local spec = self.spec_attacherJointControl
344 if spec.heightController then
345 local showText = spec.jointDesc ~= nil
346
347 local text
348 if showText then
349 if self:getControlAttacherJointDirection() then
350 text = string.format(g_i18n:getText("action_lowerOBJECT"),
    self.typeDesc)
351 else
352 text = string.format(g_i18n:getText("action_liftOBJECT"),
    self.typeDesc)
353 end
354 end
355
356 return showText, text
357 end
358
359 return superFunc(self)
360 end
361
362 function AttacherJointControl:getCanBeSelected(superFunc)
363 return true
364 end
365
366 function
    AttacherJointControl:onRegisterActionEvents(isActiveForInput)
367 if self.isClient then
368 local spec = self.spec_attacherJointControl
```

```
369 self:clearActionEventsTable(spec.actionEvents)
370
371 if isActiveForInput then
372 if spec.jointDesc ~= nil then
373 for _, control in ipairs(spec.controls) do
374 local _, actionEventId = self:addActionEvent(spec.actionEvents,
    control.controlAction, self,
    AttacherJointControl.actionEventAttacherJointControl, false,
    false, true, true, nil, control.axisActionIcon)
375 g_inputBinding:setActionEventTextPriority(actionEventId,
    GS_PRIO_NORMAL)
376 end
377 end
378 end
379 end
380 end
381
382 ---Called if vehicle gets attached
383 -- @param table attacherVehicle attacher vehicle
384 -- @param integer inputJointDescIndex index of input attacher
    joint
385 -- @param integer jointDescIndex index of attacher joint it gets
    attached to
386 -- @includeCode
387 function AttacherJointControl:onPostAttach(attacherVehicle,
    inputJointDescIndex, jointDescIndex)
388 local spec = self.spec_attacherJointControl
389
390 local inputAttacherJoints = self:getInputAttacherJoints()
391 if inputAttacherJoints[inputJointDescIndex] ~= nil and
    inputAttacherJoints[inputJointDescIndex].isControllable then
392 local attacherJoints = attacherVehicle:getAttacherJoints()
393 local jointDesc = attacherJoints[jointDescIndex]
394 jointDesc.allowsLoweringBackup = jointDesc.allowsLowering
395 jointDesc.allowsLowering = false
396 jointDesc.upperRotationOffsetBackup =
    jointDesc.upperRotationOffset
397 jointDesc.lowerRotationOffsetBackup =
    jointDesc.lowerRotationOffset
398
399 spec.jointDesc = jointDesc
400
401 -- reset control values
```

```
402 for _, control in ipairs(spec.controls) do
403 control.moveAlpha = control.func(self)
404 end
405
406 --lift after attach
407 spec.heightTargetAlpha = spec.jointDesc.upperAlpha
408
409 self:requestActionEventUpdate()
410 end
411 end
412
413 ---Called if vehicle gets detached
414 -- @param table attacherVehicle attacher vehicle
415 -- @param table implement implement
416 -- @includeCode
417 function AttacherJointControl:onPreDetach(attacherVehicle,
    implement)
418 local spec = self.spec_attacherJointControl
419
420 if spec.jointDesc ~= nil then
421 spec.jointDesc.allowsLowering =
    spec.jointDesc.allowsLoweringBackup
422 spec.jointDesc.upperRotationOffset =
    spec.jointDesc.upperRotationOffsetBackup
423 spec.jointDesc.lowerRotationOffset =
    spec.jointDesc.lowerRotationOffsetBackup
424 spec.jointDesc = nil
425 end
426 end
427
428 function
    AttacherJointControl.actionEventAttacherJointControl(self,
    actionName, inputValue, callbackState, isAnalog)
429 if math.abs(inputValue) > 0 then
430 local spec = self.spec_attacherJointControl
431 local control = spec.nameToControl[actionName]
432
433 -- multiply by 0.025 to have to same speed as in fs17
434 local changedAlpha = inputValue * control.mouseSpeedFactor * 0.025
435 if control.invertAxis then
436 changedAlpha = -changedAlpha
437 end
```

```
438
439 self:controlAttacherJoint(control, control.moveAlpha +
    changedAlpha)
440 end
441 end
442
443 function
    AttacherJointControl.actionEventAttacherJointControlSetPoint(self,
    actionName, inputValue, callbackState, isAnalog)
444 local spec = self.spec_attacherJointControl
445 if spec.jointDesc ~= nil then
446 if self:getControlAttacherJointDirection() then
447 spec.heightTargetAlpha = spec.jointDesc.lowerAlpha
448 else
449 spec.heightTargetAlpha = spec.jointDesc.upperAlpha
450 end
451 end
452 end
453
```

AttacherJoints

Description

This is the specialization for vehicles with attacherJoints

registerJointType

Description

Registration of attacher joint type

Definition

registerJointType(string name)

Arguments

string name name if attacher type

Code

```
30 function AttacherJoints.registerJointType(name)
31 local key = "JOINTTYPE_"..string.upper(name)
32 if AttacherJoints[key] == nil then
33 AttacherJoints.NUM_JOINTTYPES = AttacherJoints.NUM_JOINTTYPES+1
34 AttacherJoints[key] = AttacherJoints.NUM_JOINTTYPES
35 AttacherJoints.jointTypeNameToInt[name] =
   AttacherJoints.NUM_JOINTTYPES
36 end
37 end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
163 function AttacherJoints:onLoad(savegame)
164
165 local spec = self.spec_attacherJoints
166
167 spec.attacherJointCombos = {}
168 spec.attacherJointCombos.duration = Utils.getNoNil(getXMLFloat(self.xmlF
    "vehicle.attacherJoints#comboDuration"), 2) * 1000
169 spec.attacherJointCombos.currentTime = 0
170 spec.attacherJointCombos.direction = -1
171 spec.attacherJointCombos.isRunning = false
172 spec.attacherJointCombos.joints = {}
173
174 spec.attacherJoints = {}
175 local i=0
176 while true do
177 local baseName = string.format("vehicle.attacherJoints.attacherJoint(%d)
178 if not hasXMLProperty(self.xmlFile, baseName) then
179 break
180 end
181 local attacherJoint = {}
182 if self:loadAttacherJointFromXML(attacherJoint, self.xmlFile, baseName,
183 table.insert(spec.attacherJoints, attacherJoint)
184 attacherJoint.index = #spec.attacherJoints
185 end
186 i = i + 1
187 end
188
189 if self.configurations["attacherJoint"] ~= nil then
190 local attacherConfigs =
    string.format("vehicle.attacherJoints.attacherJointConfigurations.attach
    self.configurations["attacherJoint"]-1)
191 local i=0
192 while true do
193 local baseName = string.format(attacherConfigs..".attacherJoint(%d)", i)
194 if not hasXMLProperty(self.xmlFile, baseName) then
195 break
```

```
196 end
197 local attacherJoint = {}
198 if self:loadAttacherJointFromXML(attacherJoint, self.xmlFile, baseName,
199 table.insert(spec.attacherJoints, attacherJoint)
200 end
201 i = i + 1
202 end
203 ObjectChangeUtil.updateObjectChanges(self.xmlFile,
    "vehicle.attacherJoints.attacherJointConfigurations.attacherJointConfigu
    self.configurations["attacherJoint"], self.components, self)
204 end
205
206 spec.attachedImplements = {}
207 spec.selectedImplement = nil
208
209 -- data structure to store information about eventually attachable vehic
210 spec.attachableInfo = {}
211 spec.attachableInfo.attacherVehicle = nil
212 spec.attachableInfo.attacherVehicleJointDescIndex = nil
213 spec.attachableInfo.attachable = nil
214 spec.attachableInfo.attachableJointDescIndex = nil
215
216 if self.isClient then
217 spec.samples = {}
218 spec.isHydraulicSamplePlaying = false
219 spec.samples.hydraulic = g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.attacherJoints.sounds", "hydraulic", self.baseDirectory, self.c
    AudioGroup.VEHICLE, self.i3dMappings, self)
220 spec.samples.attach = g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.attacherJoints.sounds", "attach", self.baseDirectory, self.comp
    AudioGroup.VEHICLE, self.i3dMappings, self)
221 end
222
223 if self.isClient and g_isDevelopmentVersion then
224 for k, attacherJoint in ipairs(spec.attacherJoints) do
225 if spec.samples.attach == nil and attacherJoint.sampleAttach == nil then
226 g_logManager:xmlDevWarning(self.configFileName, "Missing attach sound fo
227 end
228 if attacherJoint.rotationNode ~= nil and spec.samples.hydraulic == nil t
229 g_logManager:xmlDevWarning(self.configFileName, "Missing hydraulic sound
    k)
230 end
```

```
231 end
232 end
233
234 spec.dirtyFlag = self:getNextDirtyFlag()
235 end
```

onPostLoad

Description

Called after loading

Definition

onPostLoad(table savegame)

Arguments

table savegame savegame

Code

```
240 function AttacherJoints:onPostLoad(savegame)
241 local spec = self.spec_attacherJoints
242
243 for _, attacherJoint in pairs(spec.attacherJoints) do
244 attacherJoint.jointOrigRot = {
    getRotation(attacherJoint.jointTransform) }
245 attacherJoint.jointOrigTrans = {
    getTranslation(attacherJoint.jointTransform) }
246 if attacherJoint.transNode ~= nil then
247 attacherJoint.transNodeMinY =
    Utils.getNoNil(attacherJoint.transNodeMinY,
    attacherJoint.jointOrigTrans[2])
248 attacherJoint.transNodeMaxY =
    Utils.getNoNil(attacherJoint.transNodeMaxY,
    attacherJoint.jointOrigTrans[2])
249 _, attacherJoint.transNodeOffsetY, _ =
    localToLocal(attacherJoint.jointTransform,
    attacherJoint.transNode, 0, 0, 0)
250 _, attacherJoint.transNodeMinY, _ =
    localToLocal(getParent(attacherJoint.transNode), self.rootNode,
    0, attacherJoint.transNodeMinY, 0)
251 _, attacherJoint.transNodeMaxY, _ =
    localToLocal(getParent(attacherJoint.transNode), self.rootNode,
    0, attacherJoint.transNodeMaxY, 0)
252 end
253
254 if attacherJoint.bottomArm ~= nil then
255 setRotation(attacherJoint.bottomArm.rotationNode,
    attacherJoint.bottomArm.rotX, attacherJoint.bottomArm.rotY,
    attacherJoint.bottomArm.rotZ)
256 if self.setMovingToolDirty ~= nil then
```

```
257 self:setMovingToolDirty(attacherJoint.bottomArm.rotationNode)
258 end
259 end
260 if attacherJoint.rotationNode ~= nil then
261 setRotation(attacherJoint.rotationNode, attacherJoint.rotX,
    attacherJoint.rotY, attacherJoint.rotZ)
262 end
263 end
264
265 if savegame ~= nil and not savegame.resetVehicles then
266 if spec.attacherJointCombos ~= nil then
267 local comboDirection = getXMLInt(savegame.xmlFile,
    savegame.key..".attacherJoints#comboDirection")
268 if comboDirection ~= nil then
269 spec.attacherJointCombos.direction = comboDirection
270 if comboDirection == 1 then
271 spec.attacherJointCombos.currentTime =
    spec.attacherJointCombos.duration
272 end
273 end
274 end
275 end
276 end
```

onPreDelete

Description

Called on before deleting

Definition

onPreDelete()

Code

```
280 function AttacherJoints:onPreDelete()
281 local spec = self.spec_attacherJoints
282
283 for i=table.getn(spec.attachedImplements), 1, -1 do
284 self:detachImplement(1, true)
285 end
286 end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
290 function AttacherJoints:onDelete()
291 local spec = self.spec_attacherJoints
292
293 if self.isClient then
294 for _, jointDesc in pairs(spec.attacherJoints) do
295 g_soundManager:deleteSample(jointDesc.sampleAttach)
296 end
297
298 g_soundManager:deleteSample(spec.samples.hydraulic)
299 g_soundManager:deleteSample(spec.samples.attach)
300 end
301 end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
345 function AttacherJoints:onReadStream(streamId, connection)
346 local numImplements = streamReadInt8(streamId)
347 for i=1, numImplements do
348 local object = NetworkUtil.readNodeObject(streamId)
349 local inputJointDescIndex = streamReadInt8(streamId)
350 local jointDescIndex = streamReadInt8(streamId)
351 local moveDown = streamReadBool(streamId)
352 if object ~= nil then
353 self:attachImplement(object, inputJointDescIndex, jointDescIndex,
    true, i)
354 self:setJointMoveDown(jointDescIndex, moveDown, true)
355 end
356 end
357 end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId
integer connection connection

Code

```
363 function AttacherJoints:onWriteStream(streamId, connection)
364 local spec = self.spec_attacherJoints
365
366 -- write attached implements
367 streamWriteInt8(streamId, table.getn(spec.attachedImplements))
368 for i=1, table.getn(spec.attachedImplements) do
369 local implement = spec.attachedImplements[i]
370 local inputJointDescIndex =
    implement.object.spec_attachable.inputAttacherJointDescIndex
371 local jointDescIndex = implement.jointDescIndex
372 local jointDesc = spec.attacherJoints[jointDescIndex]
373 local moveDown = jointDesc.moveDown
374 NetworkUtil.writeNodeObject(streamId, implement.object)
375 streamWriteInt8(streamId, inputJointDescIndex)
376 streamWriteInt8(streamId, jointDescIndex)
377 streamWriteBool(streamId, moveDown)
378 end
379 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float dt time since last call in ms
boolean isActiveForInput true if vehicle is active for input
boolean isSelected true if vehicle is selected

Code

```
387 function AttacherJoints:onUpdate(dt, isActiveForInput,
    isSelected)
388 local spec = self.spec_attacherJoints
389
390 for _, implement in pairs(spec.attachedImplements) do
391 if implement.object ~= nil then
392 if self.isServer or (self.updateLoopIndex ==
    implement.object.updateLoopIndex) then
393 self:updateAttacherJointGraphics(implement, dt)
394 end
395 end
```

```
396 end
397
398 if not self.isServer and self.getAttacherVehicle ~= nil then
399 local attacherVehicle = self:getAttacherVehicle()
400 if attacherVehicle ~= nil then
401 if self.updateLoopIndex == attacherVehicle.updateLoopIndex then
402 local implement = attacherVehicle:getImplementByObject(self)
403 if implement ~= nil then
404 attacherVehicle:updateAttacherJointGraphics(implement, dt)
405 end
406 end
407 end
408 end
409 end
```

onPostUpdate

Description

Called on after update

Definition

onPostUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
416 function AttacherJoints:onPostUpdate(dt, isActiveForInput,
    isSelected)
417 local spec = self.spec_attacherJoints
418
419 for _,implement in pairs(spec.attachedImplements) do
420 if not implement.attachingIsInProgress then
421 local attacherJoint =
    implement.object:getActiveInputAttacherJoint()
422 if attacherJoint ~= nil then
423 if
    spec.attacherJoints[implement.jointDescIndex].steeringBarLeftNode
    ~= nil and attacherJoint.steeringBarLeftMovingPart ~= nil then
424 Cylindered.updateMovingPart(self,
    attacherJoint.steeringBarLeftMovingPart, nil, true)
425 end
426 if
    spec.attacherJoints[implement.jointDescIndex].steeringBarRightNode
    ~= nil and attacherJoint.steeringBarRightMovingPart ~= nil then
```

```
427 Cylindered.updateMovingPart(self,
    attacherJoint.steeringBarRightMovingPart, nil, true)
428 end
429 end
430 end
431 end
432 end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
439 function AttacherJoints:onUpdateTick(dt, isActiveForInput, isSelected)
440 local spec = self.spec_attacherJoints
441
442 local playHydraulicSound = false
443
444 for _, implement in pairs(spec.attachedImplements) do
445 if implement.object ~= nil then
446 local jointDesc = spec.attacherJoints[implement.jointDescIndex]
447
448 if not implement.object.isHardAttached then
449 if self.isServer then
450 if implement.attachingIsInProgress then
451 local done = true
452 for i=1,3 do
453 local lastRotLimit = implement.attachingRotLimit[i]
454 local lastTransLimit = implement.attachingTransLimit[i]
455 implement.attachingRotLimit[i] = math.max(0, implement.attachingRotLimit
    implement.attachingRotLimitSpeed[i] * dt)
456 implement.attachingTransLimit[i] = math.max(0,
    implement.attachingTransLimit[i] - implement.attachingTransLimitSpeed[i]
    dt)
457 if (implement.attachingRotLimit[i] > 0 or implement.attachingTransLimit[
    0) or
458 (lastRotLimit > 0 or lastTransLimit > 0)
459 then
```

```
460 done = false
461 end
462 end
463 implement.attachingIsInProgress = not done
464
465 if done then
466 if implement.object.spec_attachable.attacherJoint.hardAttach and
    self:getIsHardAttachAllowed(implement.jointDescIndex) then
467 self:hardAttachImplement(implement)
468 end
469 self:postAttachImplement(implement)
470 end
471 end
472 end
473 if not implement.attachingIsInProgress then
474 local jointFrameInvalid = false
475 if jointDesc.allowsLowering then
476 local moveAlpha = Utils.getMovedLimitedValue(jointDesc.moveAlpha,
    jointDesc.lowerAlpha, jointDesc.upperAlpha, jointDesc.moveTime, dt, not
    jointDesc.moveDown)
477 if moveAlpha ~= jointDesc.moveAlpha then
478 playHydraulicSound = true
479 jointDesc.moveAlpha = moveAlpha
480 jointDesc.moveLimitAlpha = 1- (moveAlpha-jointDesc.lowerAlpha) /
    (jointDesc.upperAlpha-jointDesc.lowerAlpha)
481 jointFrameInvalid = true
482 self:updateAttacherJointRotationNodes(jointDesc, jointDesc.moveAlpha)
483 self:updateAttacherJointRotation(jointDesc, implement.object)
484 end
485 end
486
487 jointFrameInvalid = jointFrameInvalid or jointDesc.jointFrameInvalid
488 if jointFrameInvalid then
489 jointDesc.jointFrameInvalid = false
490 if self.isServer then
491 setJointFrame(jointDesc.jointIndex, 0, jointDesc.jointTransform)
492 end
493 end
494 end
495 if self.isServer then
496 local force = implement.attachingIsInProgress
```

```
497 if force or (jointDesc.allowsLowering and jointDesc.allowsJointLimitMove
    then
498 if jointDesc.jointIndex ~= nil and jointDesc.jointIndex ~= 0 then
499 if force or
    implement.object.spec_attachable.attacherJoint.allowsJointRotLimitMoveme
    then
500 for i=1,3 do
501 local newRotLimit = MathUtil.lerp( math.max(implement.attachingRotLimit[
    implement.upperRotLimit[i]),
502 math.max(implement.attachingRotLimit[i], implement.lowerRotLimit[i]),
    jointDesc.moveLimitAlpha)
503 if force or math.abs(newRotLimit - implement.jointRotLimit[i]) > 0.0005
504 local rotLimitDown = -newRotLimit
505 local rotLimitUp = newRotLimit
506 if i == 3 then
507 if jointDesc.lockDownRotLimit then
508 rotLimitDown = math.min(-implement.attachingRotLimit[i], 0)
509 end
510 if jointDesc.lockUpRotLimit then
511 rotLimitUp = math.max(implement.attachingRotLimit[i], 0)
512 end
513 end
514 setJointRotationLimit(jointDesc.jointIndex, i-1, true, rotLimitDown,
    rotLimitUp)
515 implement.jointRotLimit[i] = newRotLimit
516 end
517 end
518 end
519
520 if force or
    implement.object.spec_attachable.attacherJoint.allowsJointTransLimitMove
    then
521 for i=1,3 do
522 local newTransLimit = MathUtil.lerp(
    math.max(implement.attachingTransLimit[i], implement.upperTransLimit[i])
523 math.max(implement.attachingTransLimit[i], implement.lowerTransLimit[i])
    jointDesc.moveLimitAlpha)
524
525 if force or math.abs(newTransLimit - implement.jointTransLimit[i]) > 0.0
    then
526 local transLimitDown = -newTransLimit
527 local transLimitUp = newTransLimit
528 if i == 2 then
```

```
529 if jointDesc.lockDownTransLimit then
530 transLimitDown = math.min(-implement.attachingTransLimit[i], 0)
531 end
532 if jointDesc.lockUpTransLimit then
533 transLimitUp = math.max(implement.attachingTransLimit[i], 0)
534 end
535 end
536
537 setJointTranslationLimit(jointDesc.jointIndex, i-1, true, transLimitDown
    transLimitUp)
538 implement.jointTransLimit[i] = newTransLimit
539 end
540 end
541 end
542 end
543 end
544 end
545 end
546 end
547 end
548
549 if self.isClient and spec.samples.hydraulic ~= nil then
550 if playHydraulicSound then
551 if not spec.isHydraulicSamplePlaying then
552 g_soundManager:playSample(spec.samples.hydraulic)
553 spec.isHydraulicSamplePlaying = true
554 end
555 else
556 if spec.isHydraulicSamplePlaying then
557 g_soundManager:stopSample(spec.samples.hydraulic)
558 spec.isHydraulicSamplePlaying = false
559 end
560 end
561 end
562
563 local combos = spec.attacherJointCombos
564 if combos ~= nil and combos.isRunning then
565 for _, joint in pairs(combos.joints) do
566 local doLowering
567 if combos.direction == 1 and combos.currentTime >= joint.time then
```

```
568 doLowering = true
569 elseif combos.direction == -1 and combos.currentTime <= combos.duration-
    joint.time then
570 doLowering = false
571 end
572
573 if doLowering ~= nil then
574 local implement = self:getImplementFromAttacherJointIndex(joint.jointInd
575 if implement ~= nil then
576 if implement.object.setLoweredAll ~= nil then
577 implement.object:setLoweredAll(doLowering, joint.jointIndex)
578 end
579 end
580 end
581 end
582
583 if (combos.direction == -1 and combos.currentTime == 0) or
584 (combos.direction == 1 and combos.currentTime == combos.duration) then
585 combos.isRunning = false
586 end
587
588 combos.currentTime = MathUtil.clamp(combos.currentTime + dt*combos.direc
    0, combos.duration)
589 end
590
591 -- find attachables in range
592 local info = spec.attachableInfo
593 info.attacherVehicle = nil
594
595 if self.isClient then
596 if spec.actionEvents ~= nil then
597 local attachActionEvent = spec.actionEvents[InputAction.ATTACH]
598 if attachActionEvent ~= nil then
599 local visible = false
600
601 if self:getCanToggleAttach() then
602 info.attacherVehicle, info.attacherVehicleJointDescIndex, info.attachabl
    info.attachableJointDescIndex = AttacherJoints.findVehicleInAttachRange(
    AttacherJoints.MAX_ATTACH_DISTANCE_SQ, AttacherJoints.MAX_ATTACH_ANGLE)
603
604 local text = ""
```

```
605 local prio = GS_PRIO_VERY_LOW
606
607 local selectedVehicle = self:getSelectedVehicle()
608 if selectedVehicle ~= nil and not selectedVehicle.isDeleted and
    selectedVehicle.isDetachAllowed ~= nil and selectedVehicle:isDetachAllow
    then
609 visible = true
610 text = g_i18n:getText("action_detach")
611 end
612
613 if info.attacherVehicle ~= nil then
614 if g_currentMission.accessHandler:canFarmAccess(self:getActiveFarm(),
    info.attachable) then
615 visible = true
616 text = g_i18n:getText("action_attach")
617 g_currentMission:showAttachContext(info.attachable)
618 prio = GS_PRIO_VERY_HIGH
619 else
620 g_currentMission:addExtraPrintText(g_i18n:getText("info_attach_not_allow
621 end
622 end
623
624 g_inputBinding:setActionEventText(attachActionEvent.actionEventId, text)
625 g_inputBinding:setActionEventTextPriority(attachActionEvent.actionEventI
    prio)
626 end
627
628 g_inputBinding:setActionEventTextVisibility(attachActionEvent.actionEver
    visible)
629 end
630
631 local lowerActionEvent = spec.actionEvents[InputAction.LOWER_IMPLEMENT]
632 if lowerActionEvent ~= nil then
633 local showLower = false
634 local text = ""
635 local selectedImplement = self:getSelectedImplement()
636 for _,attachedImplement in pairs(spec.attachedImplements) do
637 if attachedImplement == selectedImplement then
638 showLower, text = attachedImplement.object:getLoweringActionEventState()
639 break
640 end
```

```
641 end
642
643 g_inputBinding:setActionEventActive(lowerActionEvent.actionEventId,
    showLower)
644 g_inputBinding:setActionEventText(lowerActionEvent.actionEventId, text)
645 g_inputBinding:setActionEventTextPriority(lowerActionEvent.actionEventId
    GS_PRIO_NORMAL)
646 end
647 end
648 end
649 end
```

onDraw

Description

Called on draw

Definition

onDraw(boolean isActiveForInput, boolean isSelected)

Arguments

boolean isActiveForInput	true if vehicle is active for input
boolean isSelected	true if vehicle is selected

Code

```
655 function AttacherJoints:onDraw(isActiveForInput, isSelected)
656 local spec = self.spec_attacherJoints
657
658 if self == g_currentMission.controlledVehicle then
659 -- call draw on all attached implements, selection check is done
    in the implement
660 for _, implement in ipairs(spec.attachedImplements) do
661 local object = implement.object
662 if object ~= nil then
663 if object.draw ~= nil then
664 object.draw(object, isActiveForInput, isSelected)
665 end
666 end
667 end
668 end
669 end
```

loadAttachmentsFromXMLFile

Description

Called on loading

Definition

loadAttachmentsFromXMLFile(table savegame)

Arguments

table savegame savegame

Code

```
674 function AttacherJoints:loadAttachmentsFromXMLFile(xmlFile, key, idsToVehicle)
675 local spec = self.spec_attacherJoints
676
677 local i = 0
678 while true do
679 local attachmentKey = string.format("%s.attachment(%d)", key, i)
680 if not hasXMLProperty(xmlFile, attachmentKey) then
681 break
682 end
683
684 local attachmentId = getXMLString(xmlFile, attachmentKey.."#attachmentId")
685 local jointIndex = getXMLInt(xmlFile, attachmentKey.."#jointIndex")
686 local inputJointDescIndex = Utils.getNoNil(getXMLInt(xmlFile, attachmentKey.."#inputJointDescIndex"), 1)
687 if attachmentId ~= nil and jointIndex ~= nil then
688 local attachment = idsToVehicle[attachmentId]
689
690 local inputAttacherJoints
691 if attachment ~= nil and attachment.getInputAttacherJoints ~= nil then
692 inputAttacherJoints = attachment:getInputAttacherJoints()
693 end
694 local inputAttacherJoint
695 if inputAttacherJoints ~= nil then
696 inputAttacherJoint = inputAttacherJoints[inputJointDescIndex]
697 end
698
699 if inputAttacherJoint ~= nil and spec.attacherJoints[jointIndex] ~= nil and spec.attacherJoints[jointIndex].jointIndex == 0 then
700 self:attachImplement(attachment, inputJointDescIndex, jointIndex, true, nil, nil, false)
701
702 local moveDown = getXMLBool(xmlFile, attachmentKey.."#moveDown")
703 if moveDown ~= nil then
704 self:setJointMoveDown(jointIndex, moveDown, true)
705 end
706 end
```

```
707 end
708
709 i = i + 1
710 end
711 end
```

updateAttacherJointGraphics

Description

Update attacher joint graphics

Definition

updateAttacherJointGraphics(table implement, float dt)

Arguments

table	implement	implement
float	dt	time since last call in ms

Code

```
811 function AttacherJoints:updateAttacherJointGraphics(implement, dt)
812 local spec = self.spec_attacherJoints
813
814 if implement.object ~= nil then
815 local jointDesc = spec.attacherJoints[implement.jointDescIndex]
816
817 local attacherJoint = implement.object:getActiveInputAttacherJoint()
818
819 if jointDesc.topArm ~= nil and attacherJoint.topReferenceNode ~= nil the
820 local ax, ay, az = getWorldTranslation(jointDesc.topArm.rotationNode)
821 local bx, by, bz = getWorldTranslation(attacherJoint.topReferenceNode)
822
823 local x, y, z = worldDirectionToLocal(getParent(jointDesc.topArm.rotatio
824 local distance = MathUtil.vector3Length(x,y,z)
825
826 local _, upY, upZ = 0,1,0
827 if math.abs(y) > 0.99*distance then
828 -- direction and up is parallel
829 upY = 0
830 if y > 0 then
831 upZ = 1
832 else
833 upZ = -1
834 end
835 end
836
837 -- different approach I) rotate actual direction of topArm by 90degree a
```

```
838 local alpha = math.rad(-90)
839 -- check if rotationNode is at back of tractor => inverted rotation dire
    rotating TG in i3d
840 local px,py,pz = getWorldTranslation(jointDesc.topArm.rotationNode)
841 local _,_,lz = worldToLocal(self.components[1].node, px,py,pz)
842 if lz < 0 then
843 alpha = math.rad(90)
844 end
845
846 local dx, dy, dz = localDirectionToWorld(jointDesc.topArm.rotationNode,
847 dx, dy, dz = worldDirectionToLocal(getParent(jointDesc.topArm.rotationNo
848 local upX = dx
849 local upY = math.cos(alpha)*dy - math.sin(alpha)*dz
850 local upZ = math.sin(alpha)*dy + math.cos(alpha)*dz
851
852 setDirection(jointDesc.topArm.rotationNode, x*jointDesc.topArm.zScale, y
    z*jointDesc.topArm.zScale, upX, upY, upZ)
853 if jointDesc.topArm.translationNode ~= nil and not implement.attachingIs
854 local translation = (distance-jointDesc.topArm.referenceDistance)
855 setTranslation(jointDesc.topArm.translationNode, 0, 0, translation*joint
856 if jointDesc.topArm.scaleNode ~= nil then
857 setScale(jointDesc.topArm.scaleNode, 1, 1,
    math.max((translation+jointDesc.topArm.scaleReferenceDistance)/jointDesc
    0))
858 end
859 end
860 end
861 if jointDesc.bottomArm ~= nil then
862 local ax, ay, az = getWorldTranslation(jointDesc.bottomArm.rotationNode)
863 local bx, by, bz = getWorldTranslation(attacherJoint.node)
864
865 local x, y, z = worldDirectionToLocal(getParent(jointDesc.bottomArm.rota
866 local distance = MathUtil.vector3Length(x,y,z)
867 local upX, upY, upZ = 0,1,0
868 if math.abs(y) > 0.99*distance then
869 -- direction and up is parallel
870 upY = 0
871 if y > 0 then
872 upZ = 1
873 else
874 upZ = -1
```

```
875 end
876 end
877 local dirX = 0
878 if not jointDesc.bottomArm.lockDirection then
879 dirX = x*jointDesc.bottomArm.zScale
880 end
881 setDirection(jointDesc.bottomArm.rotationNode, dirX, y*jointDesc.bottomA
    z*jointDesc.bottomArm.zScale, upX, upY, upZ)
882 if jointDesc.bottomArm.translationNode ~= nil and not implement.attachir
883 setTranslation(jointDesc.bottomArm.translationNode, 0, 0, (distance-
    jointDesc.bottomArm.referenceDistance)*jointDesc.bottomArm.zScale)
884 end
885 if self.setMovingToolDirty ~= nil then
886 self:setMovingToolDirty(jointDesc.bottomArm.rotationNode)
887 end
888
889 if attacherJoint.needsToolbar and jointDesc.bottomArm.toolbar ~= nil the
890 local parent = getParent(jointDesc.bottomArm.toolbar)
891
892 local _, yDir, zDir = localDirectionToLocal(attacherJoint.node, jointDes
893 local xDir, yDir, zDir = localDirectionToLocal(jointDesc.rootNode, parer
894
895 local _, yUp, zUp = localDirectionToLocal(attacherJoint.node, jointDesc.
896 local xUp, yUp, zUp = localDirectionToLocal(jointDesc.rootNode, parent,
897
898 setDirection(jointDesc.bottomArm.toolbar, xDir, yDir, zDir, xUp, yUp, zU
899 end
900 end
901 end
902 end
```

calculateAttacherJointMoveUpperLowerAlpha

Description

Calculate move upper and lower alpha of attacher joint

Definition

calculateAttacherJointMoveUpperLowerAlpha(table jointDesc, table object)

Arguments

table jointDesc	joint desc of used attacher
table object	object of attached vehicle

Code

```
908 function
    AttacherJoints:calculateAttacherJointMoveUpperLowerAlpha(jointDesc,
    object)
```

```
909 local objectAtttacherJoint = object.spec_attachable.attacherJoint
910
911 if jointDesc.allowsLowering then
912
913 local lowerDistanceToGround = jointDesc.lowerDistanceToGround
914 local upperDistanceToGround = jointDesc.upperDistanceToGround
915
916 if objectAtttacherJoint.heightNode ~= nil and
    jointDesc.rotationNode ~= nil then
917 self:updateAttacherJointRotationNodes(jointDesc, 1)
918
919 setRotation(jointDesc.jointTransform,
    unpack(jointDesc.jointOrigRot))
920 local x, y, z = localToLocal(jointDesc.jointTransform,
    jointDesc.rootNode, 0, 0, 0)
921 local delta = jointDesc.lowerDistanceToGround - y
922 local hx, hy, hz = localToLocal(jointDesc.jointTransform,
    jointDesc.rootNode, objectAtttacherJoint.heightNodeOffset[1],
    objectAtttacherJoint.heightNodeOffset[2],
    objectAtttacherJoint.heightNodeOffset[3])
923 lowerDistanceToGround = hy + delta
924
925 self:updateAttacherJointRotationNodes(jointDesc, 0)
926 x, y, z = localToLocal(jointDesc.jointTransform,
    jointDesc.rootNode, 0, 0, 0)
927 delta = jointDesc.upperDistanceToGround - y
928 hx, hy, hz = localToLocal(jointDesc.jointTransform,
    jointDesc.rootNode, objectAtttacherJoint.heightNodeOffset[1],
    objectAtttacherJoint.heightNodeOffset[2],
    objectAtttacherJoint.heightNodeOffset[3])
929 upperDistanceToGround = hy + delta
930 end
931
932 local upperAlpha =
    MathUtil.clamp((objectAtttacherJoint.upperDistanceToGround -
    upperDistanceToGround) / (lowerDistanceToGround -
    upperDistanceToGround), 0, 1)
933 local lowerAlpha =
    MathUtil.clamp((objectAtttacherJoint.lowerDistanceToGround -
    upperDistanceToGround) / (lowerDistanceToGround -
    upperDistanceToGround), 0, 1)
934
935 if objectAtttacherJoint.allowsLowering and jointDesc.allowsLowering
    then
936 return upperAlpha, lowerAlpha
```

```
937 else
938 if objectAtttacherJoint.isDefaultLowered then
939 return lowerAlpha,lowerAlpha
940 else
941 return upperAlpha,upperAlpha
942 end
943 end
944 end
945
946 if objectAtttacherJoint.isDefaultLowered then
947 return 1,1
948 else
949 return 0,0
950 end
951 end
```

updateAttacherJointRotation

Description

Update attacher joint rotations depending on move alpha

Definition

updateAttacherJointRotation(table jointDesc, table object)

Arguments

table jointDesc	joint desc of used attacher
table object	object of attached vehicle

Code

```
957 function AttacherJoints:updateAttacherJointRotation(jointDesc,
    object)
958 local objectAtttacherJoint = object.spec_attachable.attacherJoint
959
960 -- rotate attacher such that
961 local targetRot =
    MathUtil.lerp(objectAtttacherJoint.upperRotationOffset,
    objectAtttacherJoint.lowerRotationOffset, jointDesc.moveAlpha)
962 local curRot = MathUtil.lerp(jointDesc.upperRotationOffset,
    jointDesc.lowerRotationOffset, jointDesc.moveAlpha)
963 local rotDiff = targetRot - curRot
964
965 setRotation(jointDesc.jointTransform,
    unpack(jointDesc.jointOrigRot))
966 rotateAboutLocalAxis(jointDesc.jointTransform, rotDiff, 0, 0, 1)
967 end
```

attachImplement

Description

Attach implement

Definition

attachImplement(table object, integer inputJointDescIndex, boolean noEventSend, integer index, boolean startLowered, boolean noSmoothAttach)

Arguments

table	object	object of vehicle to attach
integer	inputJointDescIndex	index of input attacher joint to use
boolean	noEventSend	no event send
integer	index	index of attached implement
boolean	startLowered	start vehicle on lower state
boolean	noSmoothAttach	dont use smooth attach

Return Values

boolean success success

createAttachmentJoint

Description

Create attacher joint between vehicle and implement

Definition

createAttachmentJoint(table implement, boolean noSmoothAttach)

Arguments

table	implement	implement to attach
boolean	noSmoothAttach	dont use smooth attach

Code

```
1154  function AttacherJoints:createAttachmentJoint(implement, noSmoothAttach
1155
1156  local spec = self.spec_attacherJoints
1157  local jointDesc = spec.attacherJoints[implement.jointDescIndex]
1158  local objectAtttacherJoint =
      implement.object.spec_attachable.attacherJoint
1159
1160  if self.isServer and objectAtttacherJoint ~= nil then
1161  if getRigidBodyType(jointDesc.rootNode) ~= "Dynamic" or
      getRigidBodyType(objectAtttacherJoint.rootNode) ~= "Dynamic" then
1162  return
1163  end
1164
1165  local xNew = jointDesc.jointOrigTrans[1] +
      jointDesc.jointPositionOffset[1]
1166  local yNew = jointDesc.jointOrigTrans[2] +
      jointDesc.jointPositionOffset[2]
1167  local zNew = jointDesc.jointOrigTrans[3] +
      jointDesc.jointPositionOffset[3]
1168
```

```
1169  -- transform offset position to world coord and to jointTransform coord
      to get position offset dependend on angle and position
1170  local x,y,z = localToWorld(getParent(jointDesc.jointTransform), xNew,
      yNew, zNew)
1171  local x1,y1,z1 = worldToLocal(jointDesc.jointTransform, x,y,z)
1172
1173  -- move jointTransform to offset pos
1174  setTranslation(jointDesc.jointTransform, xNew, yNew, zNew)
1175
1176  -- transform it to implement position and angle
1177  x,y,z = localToWorld(objectAtttacherJoint.node,x1,y1,z1)
1178  local x2,y2,z2 = worldToLocal(getParent(objectAtttacherJoint.node),
      x,y,z)
1179  setTranslation(objectAtttacherJoint.node, x2,y2, z2)
1180
1181
1182  local constr = JointConstructor:new()
1183  constr:setActors(jointDesc.rootNode, objectAtttacherJoint.rootNode)
1184  constr:setJointTransforms(jointDesc.jointTransform,
      objectAtttacherJoint.node)
1185  --constr:setBreakable(20, 10)
1186
1187  implement.jointRotLimit = {}
1188  implement.jointTransLimit = {}
1189
1190  implement.lowerRotLimit = {}
1191  implement.lowerTransLimit = {}
1192
1193  implement.upperRotLimit = {}
1194  implement.upperTransLimit = {}
1195
1196  if noSmoothAttach == nil or not noSmoothAttach then
1197  local dx,dy,dz = localToLocal(objectAtttacherJoint.node,
      jointDesc.jointTransform, 0,0,0)
1198  local _,y,z = localDirectionToLocal(objectAtttacherJoint.node,
      jointDesc.jointTransform, 0,1,0)
1199  local rX = math.atan2(z,y)
1200  local x,_,z = localDirectionToLocal(objectAtttacherJoint.node,
      jointDesc.jointTransform, 0,0,1)
1201  local rY = math.atan2(x,z)
1202  local x,y,_ = localDirectionToLocal(objectAtttacherJoint.node,
      jointDesc.jointTransform, 1,0,0)
```

```
1203  local rZ = math.atan2(y,x)
1204  implement.attachingTransLimit = { math.abs(dx), math.abs(dy),
      math.abs(dz) }
1205  implement.attachingRotLimit = { math.abs(rX), math.abs(rY), math.abs(rZ
      }
1206  implement.attachingTransLimitSpeed = {}
1207  implement.attachingRotLimitSpeed = {}
1208  for i=1,3 do
1209  implement.attachingTransLimitSpeed[i] = implement.attachingTransLimit[i
      / 500
1210  implement.attachingRotLimitSpeed[i] = implement.attachingRotLimit[i] /
      500
1211  end
1212  implement.attachingIsInProgress = true
1213  else
1214  implement.attachingTransLimit = { 0,0,0 }
1215  implement.attachingRotLimit = { 0,0,0 }
1216  end
1217
1218  for i=1, 3 do
1219  local lowerRotLimit =
      jointDesc.lowerRotLimit[i]*objectAtttacherJoint.lowerRotLimitScale[i]
1220  local upperRotLimit =
      jointDesc.upperRotLimit[i]*objectAtttacherJoint.upperRotLimitScale[i]
1221  if objectAtttacherJoint.fixedRotation then
1222  lowerRotLimit = 0
1223  upperRotLimit = 0
1224  end
1225
1226  local upperTransLimit =
      jointDesc.lowerTransLimit[i]*objectAtttacherJoint.lowerTransLimitScale[
1227  local lowerTransLimit =
      jointDesc.upperTransLimit[i]*objectAtttacherJoint.upperTransLimitScale[
1228  implement.lowerRotLimit[i] = lowerRotLimit
1229  implement.upperRotLimit[i] = upperRotLimit
1230
1231  implement.lowerTransLimit[i] = upperTransLimit
1232  implement.upperTransLimit[i] = lowerTransLimit
1233
1234  if not jointDesc.allowsLowering then
1235  implement.upperRotLimit[i] = lowerRotLimit
1236  implement.upperTransLimit[i] = upperTransLimit
```

```
1237  end
1238
1239  local rotLimit = lowerRotLimit
1240  local transLimit = upperTransLimit
1241  if jointDesc.allowsLowering and jointDesc.allowsJointLimitMovement ther
1242  if objectAtttacherJoint.allowsJointRotLimitMovement then
1243  rotLimit = MathUtil.lerp(upperRotLimit, lowerRotLimit,
      jointDesc.moveAlpha)
1244  end
1245  if objectAtttacherJoint.allowsJointTransLimitMovement then
1246  transLimit = MathUtil.lerp(lowerTransLimit, upperTransLimit,
      jointDesc.moveAlpha)
1247  end
1248  end
1249
1250  local limitRot = rotLimit
1251  local limitTrans = transLimit
1252  if noSmoothAttach == nil or not noSmoothAttach then
1253  limitRot = math.max(rotLimit, implement.attachingRotLimit[i])
1254  limitTrans = math.max(transLimit, implement.attachingTransLimit[i])
1255  end
1256
1257  constr:setRotationLimit(i-1, -limitRot, limitRot)
1258  implement.jointRotLimit[i] = limitRot
1259  constr:setTranslationLimit(i-1, true, -limitTrans, limitTrans)
1260  implement.jointTransLimit[i] = limitTrans
1261  end
1262
1263  if jointDesc.enableCollision then
1264  constr:setEnableCollision(true)
1265  else
1266  for _, component in pairs(self.components) do
1267  if component.node ~= jointDesc.rootNodeBackup and not
      component.collideWithAttachables then
1268  setPairCollision(component.node, objectAtttacherJoint.rootNode, false)
1269  end
1270  end
1271  end
1272
1273  local springX = math.max(jointDesc.rotLimitSpring[1],
      objectAtttacherJoint.rotLimitSpring[1])
```

```
1274  local springY = math.max(jointDesc.rotLimitSpring[2],
      objectAtttacherJoint.rotLimitSpring[2])
1275  local springZ = math.max(jointDesc.rotLimitSpring[3],
      objectAtttacherJoint.rotLimitSpring[3])
1276  local dampingX = math.max(jointDesc.rotLimitDamping[1],
      objectAtttacherJoint.rotLimitDamping[1])
1277  local dampingY = math.max(jointDesc.rotLimitDamping[2],
      objectAtttacherJoint.rotLimitDamping[2])
1278  local dampingZ = math.max(jointDesc.rotLimitDamping[3],
      objectAtttacherJoint.rotLimitDamping[3])
1279  local forceLimitX =
      Utils.getMaxJointForceLimit(jointDesc.rotLimitForceLimit[1],
      objectAtttacherJoint.rotLimitForceLimit[1])
1280  local forceLimitY =
      Utils.getMaxJointForceLimit(jointDesc.rotLimitForceLimit[2],
      objectAtttacherJoint.rotLimitForceLimit[2])
1281  local forceLimitZ =
      Utils.getMaxJointForceLimit(jointDesc.rotLimitForceLimit[3],
      objectAtttacherJoint.rotLimitForceLimit[3])
1282  constr:setRotationLimitSpring(springX, dampingX, springY, dampingY,
      springZ, dampingZ)
1283  constr:setRotationLimitForceLimit(forceLimitX, forceLimitY, forceLimitZ
1284
1285  local springX = math.max(jointDesc.transLimitSpring[1],
      objectAtttacherJoint.transLimitSpring[1])
1286  local springY = math.max(jointDesc.transLimitSpring[2],
      objectAtttacherJoint.transLimitSpring[2])
1287  local springZ = math.max(jointDesc.transLimitSpring[3],
      objectAtttacherJoint.transLimitSpring[3])
1288  local dampingX = math.max(jointDesc.transLimitDamping[1],
      objectAtttacherJoint.transLimitDamping[1])
1289  local dampingY = math.max(jointDesc.transLimitDamping[2],
      objectAtttacherJoint.transLimitDamping[2])
1290  local dampingZ = math.max(jointDesc.transLimitDamping[3],
      objectAtttacherJoint.transLimitDamping[3])
1291  local forceLimitX =
      Utils.getMaxJointForceLimit(jointDesc.transLimitForceLimit[1],
      objectAtttacherJoint.transLimitForceLimit[1])
1292  local forceLimitY =
      Utils.getMaxJointForceLimit(jointDesc.transLimitForceLimit[2],
      objectAtttacherJoint.transLimitForceLimit[2])
1293  local forceLimitZ =
      Utils.getMaxJointForceLimit(jointDesc.transLimitForceLimit[3],
      objectAtttacherJoint.transLimitForceLimit[3])
1294  constr:setTranslationLimitSpring(springX, dampingX, springY, dampingY,
      springZ, dampingZ)
```

```
1295  constr:setTranslationLimitForceLimit(forceLimitX, forceLimitY,
      forceLimitZ)
1296
1297  jointDesc.jointIndex = constr:finalize()
1298
1299  -- restore implement attacher joint position (to ensure correct bottom
      arm alignment)
1300  setTranslation(objectAtttacherJoint.node,
      unpack(objectAtttacherJoint.jointOrigTrans))
1301  else
1302  jointDesc.jointIndex = 0
1303  end
1304  end
```

hardAttachImplement

Description

Hard attach implement

Definition

hardAttachImplement(table implement)

Arguments

table implement implement to attach

Code

```
1309  function AttacherJoints:hardAttachImplement(implement)
1310  local spec = self.spec_attacherJoints
1311
1312  local implements = {}
1313  local attachedImplements
1314  if implement.object.getAttachedImplements ~= nil then
1315  attachedImplements = implement.object:getAttachedImplements()
1316  end
1317  if attachedImplements ~= nil then
1318  for i = table.getn(attachedImplements), 1, -1 do
1319  local impl = attachedImplements[i]
1320  local object = impl.object
1321  local jointDescIndex = impl.jointDescIndex
1322  local jointDesc = implement.object.spec_attacherJoints.attacherJoints[j
1323  local inputJointDescIndex = object.spec_attachable.inputAttacherJointDe
1324  local moveDown = jointDesc.moveDown
1325  table.insert(implements, 1, {object=object, implementIndex=i,
      jointDescIndex=jointDescIndex, inputJointDescIndex=inputJointDescIndex,
      moveDown=moveDown})
1326  implement.object:detachImplement(1, true)
1327  end
```

```
1328  end
1329
1330  local attacherJoint = spec.attacherJoints[implement.jointDescIndex]
1331  local implementJoint = implement.object.spec_attachable.attacherJoint
1332
1333  local baseVehicleComponentNode = self:getParentComponent(attacherJoint.
1334  local attachedVehicleComponentNode =
      implement.object:getParentComponent(implement.object.spec_attachable.at
1335
1336  -- remove all components from physics
1337  local currentVehicle = self
1338  while currentVehicle ~= nil do
1339  currentVehicle:removeFromPhysics()
1340  currentVehicle = currentVehicle.attacherVehicle
1341  end
1342  implement.object:removeFromPhysics()
1343
1344  -- set valid baseVehicle compound
1345  if spec.attacherVehicle == nil then
1346  setIsCompound(baseVehicleComponentNode, true)
1347  end
1348  -- set attachedVehicle to compound child
1349  setIsCompoundChild(attachedVehicleComponentNode, true)
1350
1351  -- set direction and local position
1352  local dirX, dirY, dirZ = localDirectionToLocal(attachedVehicleComponent
      implementJoint.node, 0, 0, 1)
1353  local upX, upY, upZ = localDirectionToLocal(attachedVehicleComponentNod
      implementJoint.node, 0, 1, 0)
1354  setDirection(attachedVehicleComponentNode, dirX, dirY, dirZ, upX, upY,
1355  local x,y,z = localToLocal(attachedVehicleComponentNode, implementJoint
1356  setTranslation(attachedVehicleComponentNode, x, y, z)
1357  link(attacherJoint.jointTransform, attachedVehicleComponentNode)
1358
1359  -- link visual and set to correct position
1360  if implementJoint.visualNode ~= nil and attacherJoint.jointTransformVis
1361  local dirX, dirY, dirZ = localDirectionToLocal(implementJoint.visualNod
      implementJoint.node, 0, 0, 1)
1362  local upX, upY, upZ = localDirectionToLocal(implementJoint.visualNode,
      implementJoint.node, 0, 1, 0)
1363  setDirection(implementJoint.visualNode, dirX, dirY, dirZ, upX, upY, upZ
```

```
1364  local x,y,z = localToLocal(implementJoint.visualNode, implementJoint.no
1365  setTranslation(implementJoint.visualNode, x, y, z)
1366  link(attacherJoint.jointTransformVisual, implementJoint.visualNode)
1367  end
1368
1369  implement.object.isHardAttached = true
1370
1371  -- add to physics again
1372  local currentVehicle = self
1373  while currentVehicle ~= nil do
1374  currentVehicle:addToPhysics()
1375  currentVehicle = currentVehicle.attacherVehicle
1376  end
1377
1378  -- set new joint rootNodes
1379  for _, attacherJoint in pairs(implement.object.spec_attacherJoints.atta
1380  attacherJoint.rootNode = self.rootNode
1381  end
1382
1383  for _, impl in pairs(implements) do
1384  implement.object:attachImplement(impl.object, impl.inputJointDescIndex,
      impl.jointDescIndex, true, impl.implementIndex, impl.moveDown, true)
1385  end
1386
1387  if self.isServer then
1388  self:raiseDirtyFlags(self.vehicleDirtyFlag)
1389  end
1390
1391  return true
1392  end
```

hardDetachImplement

Description

Hard detach implement

Definition

hardDetachImplement(table implement)

Arguments

table implement implement to detach

Code

```
1397  function AttacherJoints:hardDetachImplement(implement)
1398  -- restore original joint rootNode
```

```
for _, attacherJoint in pairs(implement.object.spec_attacherJoints.attacherJoints) do
attacherJoint.rootNode = attacherJoint.rootNodeBackup
end

local implementJoint = implement.object.spec_attachable.attacherJoint

local attachedVehicleComponentNode = implement.object:getParentComponent(implementJoint.node)

local currentVehicle = self
while currentVehicle ~= nil do
currentVehicle:removeFromPhysics()
currentVehicle = currentVehicle.attacherVehicle
end
--implement.object:removeFromPhysics()

setIsCompound(attachedVehicleComponentNode, true)

local x,y,z = getWorldTranslation(attachedVehicleComponentNode)
setTranslation(attachedVehicleComponentNode, x,y,z)
local dirX, dirY, dirZ = localDirectionToWorld(implement.object.rootNode, 0, 0, 1)
local upX, upY, upZ = localDirectionToWorld(implement.object.rootNode, 0, 1, 0)
setDirection(attachedVehicleComponentNode, dirX, dirY, dirZ, upX, upY, upZ)
link(getRootNode(), attachedVehicleComponentNode)

if implementJoint.visualNode ~= nil and getParent(implementJoint.visualNode) ~= implementJoint.visualNodeData.parent then
link(implementJoint.visualNodeData.parent, implementJoint.visualNode, implementJoint.visualNodeData.index)
setRotation(implementJoint.visualNode, implementJoint.visualNodeData.rotation[1], implementJoint.visualNodeData.rotation[2], implementJoint.visualNodeData.rotation[3])
setTranslation(implementJoint.visualNode, implementJoint.visualNodeData.translation[1], implementJoint.visualNodeData.translation[2], implementJoint.visualNodeData.translation[3])
end
```

```
1428
1429  local currentVehicle = self
1430  while currentVehicle ~= nil do
1431  currentVehicle:addToPhysics()
1432  currentVehicle = currentVehicle.attacherVehicle
1433  end
1434  implement.object:addToPhysics()
1435  implement.object.isHardAttached = false
1436
1437  if self.isServer then
1438  self:raiseDirtyFlags(self.vehicleDirtyFlag)
1439  end
1440
1441  return true
1442  end
```

detachImplement

Description

Detach implement

Definition

detachImplement(integer implementIndex, boolean noEventSend)

Arguments

integer	implementIndex	index of implement in self.attachedImplements
boolean	noEventSend	no event send

Return Values

boolean	success success

Code

```
1449  function AttacherJoints:detachImplement(implementIndex, noEventSend)
1450  local spec = self.spec_attacherJoints
1451
1452  if noEventSend == nil or noEventSend == false then
1453  if g_server ~= nil then
1454  g_server:broadcastEvent(VehicleDetachEvent:new(self,
      spec.attachedImplements[implementIndex].object), nil, nil, self)
1455  else
1456  -- Send detach request to server and return
1457  local implement = spec.attachedImplements[implementIndex]
1458  if implement.object ~= nil then
1459  g_client:getServerConnection():sendEvent(VehicleDetachEvent:new(self,
      implement.object))
1460  end
1461  return
1462  end
```

```
1463  end
1464
1465  local implement = spec.attachedImplements[implementIndex]
1466
1467  SpecializationUtil.raiseEvent(self, "onPreDetachImplement",
      implement)
1468  implement.object:preDetach(self, implement)
1469
1470  local jointDesc
1471  if implement.object ~= nil then
1472  jointDesc = spec.attacherJoints[implement.jointDescIndex]
1473  if jointDesc.transNode ~= nil then
1474  setTranslation(jointDesc.transNode,
      unpack(jointDesc.transNodeOrgTrans))
1475  end
1476  if not implement.object.isHardAttached then
1477  if self.isServer then
1478  if jointDesc.jointIndex ~= 0 then
1479  removeJoint(jointDesc.jointIndex)
1480  end
1481
1482  if not jointDesc.enableCollision then
1483  for _, component in pairs(self.components) do
1484  if component.node ~= jointDesc.rootNodeBackup and not
      component.collideWithAttachables then
1485  local attacherJoint = implement.object:getActiveInputAttacherJoint()
1486  setPairCollision(component.node, attacherJoint.rootNode, true)
1487  end
1488  end
1489  end
1490  end
1491  end
1492  jointDesc.jointIndex = 0
1493  end
1494
1495  ObjectChangeUtil.setObjectChanges(jointDesc.changeObjects, false,
      self, self.setMovingToolDirty)
1496
1497  if implement.object ~= nil then
1498  local object = implement.object
1499
```

```
1500  if object.isHardAttached then
1501  self:hardDetachImplement(implement)
1502  end
1503
1504  if self.isClient then
1505  if jointDesc.topArm ~= nil then
1506  setRotation(jointDesc.topArm.rotationNode, jointDesc.topArm.rotX,
      jointDesc.topArm.rotY, jointDesc.topArm.rotZ)
1507  if jointDesc.topArm.translationNode ~= nil then
1508  setTranslation(jointDesc.topArm.translationNode, 0, 0, 0)
1509  end
1510  if jointDesc.topArm.scaleNode ~= nil then
1511  setScale(jointDesc.topArm.scaleNode, 1, 1, 1)
1512  end
1513  if jointDesc.topArm.toggleVisibility then
1514  setVisibility(jointDesc.topArm.rotationNode, false)
1515  end
1516  end
1517  if jointDesc.bottomArm ~= nil then
1518  setRotation(jointDesc.bottomArm.rotationNode,
      jointDesc.bottomArm.rotX, jointDesc.bottomArm.rotY,
      jointDesc.bottomArm.rotZ)
1519  if jointDesc.bottomArm.translationNode ~= nil then
1520  setTranslation(jointDesc.bottomArm.translationNode, 0, 0, 0)
1521  end
1522  if self.setMovingToolDirty ~= nil then
1523  self:setMovingToolDirty(jointDesc.bottomArm.rotationNode)
1524  end
1525  if jointDesc.bottomArm.toolbar ~= nil then
1526  setVisibility(jointDesc.bottomArm.toolbar, false)
1527  end
1528  if jointDesc.bottomArm.toggleVisibility then
1529  setVisibility(jointDesc.bottomArm.rotationNode, false)
1530  end
1531  end
1532  end
1533  -- restore original translation
1534  setTranslation(jointDesc.jointTransform,
      unpack(jointDesc.jointOrigTrans))
1535  local attacherJoint = object:getActiveInputAttacherJoint()
```

```
1536  setTranslation(attacherJoint.node,
      unpack(attacherJoint.jointOrigTrans))
1537  if jointDesc.rotationNode ~= nil then
1538  setRotation(jointDesc.rotationNode, jointDesc.rotX, jointDesc.rotY,
      jointDesc.rotZ)
1539  end
1540
1541  SpecializationUtil.raiseEvent(self, "onPostDetachImplement",
      implementIndex)
1542  object:postDetach(implementIndex)
1543  end
1544
1545  table.remove(spec.attachedImplements, implementIndex)
1546
1547  self:playDetachSound(jointDesc)
1548
1549  local data = {attacherVehicle=self, attachedVehicle=implement.object}
1550  implement.object:raiseStateChange(Vehicle.STATE_CHANGE_DETACH, data)
1551  local rootVehicle = self:getRootVehicle()
1552  rootVehicle:raiseStateChange(Vehicle.STATE_CHANGE_DETACH, data)
1553
1554  self:getRootVehicle():updateSelectableObjects()
1555  self:getRootVehicle():setSelectedVehicle(self, nil, true)
1556  self:getRootVehicle():requestActionEventUpdate() -- do action event
      update independent of a successful selection (important since we
      cannot select every vehicle)
1557  implement.object:updateSelectableObjects()
1558  implement.object:setSelectedVehicle(implement.object, nil, true)
1559  implement.object:requestActionEventUpdate() -- do action event update
      independent of a successful selection (important since we cannot
      select every vehicle)
1560
1561  return true
1562  end
```

detachImplementByObject

Description

Detach implement by object of implement

Definition

detachImplementByObject(table object, boolean noEventSend)

Arguments

table	object	object of implement to detach
boolean	noEventSend	no event send

Return Values

boolean success success

Code

```
1569  function AttacherJoints:detachImplementByObject(object, noEventSend)
1570  local spec = self.spec_attacherJoints
1571
1572  for i,implement in ipairs(spec.attachedImplements) do
1573  if implement.object == object then
1574  self:detachImplement(i, noEventSend)
1575  break
1576  end
1577  end
1578
1579  return true
1580  end
```

playAttachSound

Description

Play attach sound

Definition

playAttachSound(table jointDesc)

Arguments

table jointDesc joint desc

Return Values

boolean success success

Code

```
1619  function AttacherJoints:playAttachSound(jointDesc)
1620  local spec = self.spec_attacherJoints
1621
1622  if self.isClient then
1623  if jointDesc ~= nil and jointDesc.sampleAttach ~= nil then
1624  g_soundManager:playSample(jointDesc.sampleAttach)
1625  else
1626  g_soundManager:playSample(spec.samples.attach)
1627  end
1628  end
1629
1630  return true
1631  end
```

playDetachSound

Description

Play detach sound

Definition

playDetachSound(table jointDesc)

Arguments

table jointDesc joint desc

Return Values

boolean success success

Code

```
1637  function AttacherJoints:playDetachSound(jointDesc)
1638  local spec = self.spec_attacherJoints
1639
1640  if self.isClient then
1641  if jointDesc ~= nil and jointDesc.sampleAttach ~= nil then
1642  g_soundManager:playSample(jointDesc.sampleAttach)
1643  else
1644  g_soundManager:playSample(spec.samples.attach)
1645  end
1646  end
1647
1648  return true
1649  end
```

detachingIsPossible

Description

Returns true if it is possible to detach selected implement

Definition

detachingIsPossible()

Return Values

boolean possibleToDetach possible to detach selected implement

Code

```
1654  function AttacherJoints:detachingIsPossible()
1655  local implement =
      self:getImplementByObject(self:getSelectedVehicle())
1656  if implement ~= nil then
1657  local object = implement.object
1658  if object ~= nil and object.attacherVehicle ~= nil and
      object:isDetachAllowed() then
1659  local implementIndex =
      object.attacherVehicle:getImplementIndexByObject(object)
1660  if implementIndex ~= nil then
1661  return true
1662  end
1663  end
1664  end
1665  return false
```

```
1666  end
```

getImplementIndexByJointDescIndex

Description

Returns implement index in 'self.attachedImplements' by jointDescIndex

Definition

getImplementIndexByJointDescIndex(integer jointDescIndex)

Arguments

integer jointDescIndex joint desc index

Return Values

integer index index of implement

Code

```
1672  function AttacherJoints:getImplementIndexByJointDescIndex(jointDescIndex)
1673      local spec = self.spec_attacherJoints
1674
1675      for i, implement in pairs(spec.attachedImplements) do
1676          if implement.jointDescIndex == jointDescIndex then
1677              return i
1678          end
1679      end
1680
1681      return nil
1682  end
```

getImplementByJointDescIndex

Description

Returns implement by jointDescIndex

Definition

getImplementByJointDescIndex(integer jointDescIndex)

Arguments

integer jointDescIndex joint desc index

Return Values

table implement implement

Code

```
1688  function AttacherJoints:getImplementByJointDescIndex(jointDescIndex)
1689      local spec = self.spec_attacherJoints
1690
1691      for i, implement in pairs(spec.attachedImplements) do
1692          if implement.jointDescIndex == jointDescIndex then
1693              return implement
1694          end
1695      end
```

```
1696
1697  return nil
1698  end
```

getImplementIndexByObject

Description

Returns implement index in 'self.attachedImplements' by object

Definition

getImplementIndexByObject(table object)

Arguments

table object object of attached implement

Return Values

integer index index of implement

Code

```
1704  function AttacherJoints:getImplementIndexByObject(object)
1705  local spec = self.spec_attacherJoints
1706
1707  for i, implement in pairs(spec.attachedImplements) do
1708  if implement.object == object then
1709  return i
1710  end
1711  end
1712
1713  return nil
1714  end
```

getImplementByObject

Description

Returns implement by object

Definition

getImplementByObject(table object)

Arguments

table object object of attached implement

Return Values

table implement implement

Code

```
1720  function AttacherJoints:getImplementByObject(object)
1721  local spec = self.spec_attacherJoints
1722
1723  for i, implement in pairs(spec.attachedImplements) do
1724  if implement.object == object then
1725  return implement
1726  end
1727  end
```

```
1728
1729  return nil
1730  end
```

activateAttachments

Description

Call "activate" on all attachments

Definition

activateAttachments()

Code

```
1745  function AttacherJoints:activateAttachments()
1746  local spec = self.spec_attacherJoints
1747
1748  for _,v in pairs(spec.attachedImplements) do
1749  if v.object ~= nil then
1750  v.object:activate()
1751  end
1752  end
1753  end
```

deactivateAttachments

Description

Call "deactivate" on all attachments

Definition

deactivateAttachments()

Code

```
1757  function AttacherJoints:deactivateAttachments()
1758  local spec = self.spec_attacherJoints
1759
1760  for _,v in pairs(spec.attachedImplements) do
1761  if v.object ~= nil then
1762  v.object:deactivate()
1763  end
1764  end
1765  end
```

deactivateAttachmentsLights

Description

Call "deactivateLights" on all attachments

Definition

deactivateAttachmentsLights()

Code

```
1769  function AttacherJoints:deactivateAttachmentsLights()
1770  local spec = self.spec_attacherJoints
```

```
1771
1772  for _,v in pairs(spec.attachedImplements) do
1773  if v.object ~= nil and v.object.deactivateLights ~= nil then
1774  v.object:deactivateLights()
1775  end
1776  end
1777  end
```

setJointMoveDown

Description

Set joint move down

Definition

setJointMoveDown(integer jointDescIndex, boolean moveDown, boolean noEventSend)

Arguments

integer	jointDescIndex	index of joint desc
boolean	moveDown	move down
boolean	noEventSend	no event send

Return Values

boolean success success

Code

```
1785  function AttacherJoints:setJointMoveDown(jointDescIndex,
      moveDown, noEventSend)
1786  local spec = self.spec_attacherJoints
1787
1788  VehicleLowerImplementEvent.sendEvent(self, jointDescIndex,
      moveDown, noEventSend)
1789  local jointDesc = spec.attacherJoints[jointDescIndex]
1790  jointDesc.moveDown = moveDown
1791
1792  local implementIndex =
      self:getImplementIndexByJointDescIndex(jointDescIndex)
1793  if implementIndex ~= nil then
1794  local implement = spec.attachedImplements[implementIndex]
1795  if implement.object ~= nil then
1796  implement.object:setLowered(moveDown)
1797  end
1798  end
1799
1800  return true
1801  end
```

getIsHardAttachAllowed

Description

Returns if attacher joint supports hard attach

Definition

getIsHardAttachAllowed(integer jointDescIndex)

Arguments

integer jointDescIndex index of joint

Return Values

boolean supportsHardAttach attacher joint supports hard attach

Code

```
1807  function AttacherJoints:getIsHardAttachAllowed(jointDescIndex)
1808  local spec = self.spec_attacherJoints
1809
1810  return spec.attacherJoints[jointDescIndex].supportsHardAttach
1811  end
```

loadAttacherJointFromXML

Description

Load attacher joint from xml

Definition

loadAttacherJointFromXML(table attacherJoint, integer fileId, string baseName, integer index)

Arguments

table	attacherJoint	attacherJoint
integer	fileId	xml file id
string	baseName	baseName
integer	index	index of attacher joint

Code

```
1819  function AttacherJoints:loadAttacherJointFromXML(attacherJoint,
      xmlFile, baseName, index)
1820  local spec = self.spec_attacherJoints
1821
1822  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#index", baseName .. "#node") -- FS17
1823  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#indexVisual", baseName .. "#nodeVisual") -- FS17
1824  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#ptoOutputNode", "vehicle.powerTakeOffs.output") --
      FS17 to FS19
1825  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#lowerDistanceToGround",
      baseName.."".distanceToGround#lower") -- FS17 to FS19
1826  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#upperDistanceToGround",
      baseName.."".distanceToGround#upper") -- FS17 to FS19
1827  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#rotationNode", baseName.."".rotationNode#node") --
      FS17 to FS19
```

```
1828  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#upperRotation",
      baseName..".rotationNode#upperRotation") -- FS17 to FS19
1829  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#lowerRotation",
      baseName..".rotationNode#lowerRotation") -- FS17 to FS19
1830  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#startRotation",
      baseName..".rotationNode#startRotation") -- FS17 to FS19
1831  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#rotationNode2", baseName..".rotationNode2#node") --
      FS17 to FS19
1832  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#upperRotation2",
      baseName..".rotationNode2#upperRotation") -- FS17 to FS19
1833  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#lowerRotation2",
      baseName..".rotationNode2#lowerRotation") -- FS17 to FS19
1834  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#transNode", baseName..".transNode#node") -- FS17 to
      FS19
1835  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#transNodeMinY", baseName..".transNode#minY") -- FS17
      to FS19
1836  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#transNodeMaxY", baseName..".transNode#maxY") -- FS17
      to FS19
1837  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
      baseName .. "#transNodeHeight", baseName..".transNode#height") --
      FS17 to FS19
1838
1839
1840  local node = I3DUtil.indexToObject(self.components,
      getXMLString(xmlFile, baseName.. "#node"), self.i3dMappings)
1841  if node == nil then
1842  g_logManager:xmlWarning(self.configFileName, "Missing node for
      attacherJoint '%s'", baseName)
1843  return false
1844  end
1845
1846  attacherJoint.jointTransform = node
1847
1848  attacherJoint.jointTransformVisual =
      I3DUtil.indexToObject(self.components, getXMLString(xmlFile,
      baseName .. "#nodeVisual"), self.i3dMappings)
```

```
1849  attacherJoint.supportsHardAttach =
      Utils.getNoNil(getXMLBool(xmlFile,
      baseName.."#supportsHardAttach"), true)
1850
1851  local jointTypeStr = getXMLString(xmlFile, baseName.. "#jointType")
1852  local jointType
1853  if jointTypeStr ~= nil then
1854  jointType = AttacherJoints.jointTypeNameToInt[jointTypeStr]
1855  if jointType == nil then
1856  g_logManager:xmlWarning(self.configFileName, "Invalid jointType
      '%s' for attacherJoint '%s'!", tostring(jointTypeStr), baseName)
1857  end
1858  end
1859  if jointType == nil then
1860  jointType = AttacherJoints.JOINTTYPE_IMPLEMENT
1861  end
1862  attacherJoint.jointType = jointType
1863  attacherJoint.allowsJointLimitMovement =
      Utils.getNoNil(getXMLBool(xmlFile,
      baseName.."#allowsJointLimitMovement"), true)
1864  attacherJoint.allowsLowering = Utils.getNoNil(getXMLBool(xmlFile,
      baseName.."#allowsLowering"), true)
1865  attacherJoint.isDefaultLowered = Utils.getNoNil(getXMLBool(xmlFile,
      baseName.."#isDefaultLowered"), false)
1866
1867  attacherJoint.allowDetachingWhileLifted =
      Utils.getNoNil(getXMLBool(xmlFile,
      baseName.."#allowDetachingWhileLifted"), true)
1868
1869  if jointType == AttacherJoints.JOINTTYPE_TRAILER or jointType ==
      AttacherJoints.JOINTTYPE_TRAILERLOW then
1870  attacherJoint.allowsLowering = false
1871  end
1872
1873  attacherJoint.canTurnOnImplement =
      Utils.getNoNil(getXMLBool(xmlFile,
      baseName.."#canTurnOnImplement"), true)
1874
1875  local rotationNode = I3DUtil.indexToObject(self.components,
      getXMLString(xmlFile, baseName.. ".rotationNode#node"),
      self.i3dMappings)
1876  if rotationNode ~= nil then
1877  attacherJoint.rotationNode = rotationNode
```

```
1878  local x, y, z = StringUtil.getVectorFromString(getXMLString(xmlFile, baseName..".rotationNode#lowerRotation"))
1879  attacherJoint.lowerRotation = { math.rad(Utils.getNoNil(x, 0)), math.rad(Utils.getNoNil(y, 0)), math.rad(Utils.getNoNil(z, 0)) }
1880
1881  local x, y, z = StringUtil.getVectorFromString(getXMLString(xmlFile, baseName..".rotationNode#upperRotation"))
1882  local rx,ry,rz = getRotation(rotationNode)
1883  attacherJoint.upperRotation = { Utils.getNoNilRad(x, rx), Utils.getNoNilRad(y, ry), Utils.getNoNilRad(z, rz) }
1884
1885  local startRot = StringUtil.getRadiansFromString(getXMLString(xmlFile, baseName..".rotationNode#startRotation"), 3)
1886  if startRot ~= nil then
1887  attacherJoint.rotX, attacherJoint.rotY, attacherJoint.rotZ = startRot[1],startRot[2],startRot[3]
1888  else
1889  attacherJoint.rotX, attacherJoint.rotY, attacherJoint.rotZ = getRotation(rotationNode)
1890  end
1891
1892  local lowerValues = {attacherJoint.lowerRotation[1], attacherJoint.lowerRotation[2], attacherJoint.lowerRotation[3]}
1893  local upperValues = {attacherJoint.upperRotation[1], attacherJoint.upperRotation[2], attacherJoint.upperRotation[3]}
1894
1895  for i=1, 3 do
1896  local l = lowerValues[i]
1897  local u = upperValues[i]
1898
1899  if l > u then
1900  upperValues[i] = l
1901  lowerValues[i] = u
1902  end
1903  end
1904
1905  attacherJoint.rotX = MathUtil.clamp(attacherJoint.rotX, lowerValues[1], upperValues[1])
1906  attacherJoint.rotY = MathUtil.clamp(attacherJoint.rotY, lowerValues[2], upperValues[2])
```

```
1907    attacherJoint.rotZ = MathUtil.clamp(attacherJoint.rotZ,
        lowerValues[3], upperValues[3])
1908    end
1909    local rotationNode2 = I3DUtil.indexToObject(self.components,
        getXMLString(xmlFile, baseName.. ".rotationNode2#node"),
        self.i3dMappings)
1910    if rotationNode2 ~= nil then
1911    attacherJoint.rotationNode2 = rotationNode2
1912    local x, y, z =
        StringUtil.getVectorFromString(getXMLString(xmlFile,
        baseName..".rotationNode2#lowerRotation"))
1913    if x ~= nil and y ~= nil and z ~= nil then
1914    attacherJoint.lowerRotation2 = { math.rad(Utils.getNoNil(x, 0)),
        math.rad(Utils.getNoNil(y, 0)), math.rad(Utils.getNoNil(z, 0)) }
1915    else
1916    attacherJoint.lowerRotation2 = { -attacherJoint.lowerRotation[1], -
        attacherJoint.lowerRotation[2], -attacherJoint.lowerRotation[3] }
1917    end
1918
1919    local x, y, z =
        StringUtil.getVectorFromString(getXMLString(xmlFile,
        baseName..".rotationNode2#upperRotation"))
1920    if x ~= nil and y ~= nil and z ~= nil then
1921    attacherJoint.upperRotation2 = { math.rad(Utils.getNoNil(x, 0)),
        math.rad(Utils.getNoNil(y, 0)), math.rad(Utils.getNoNil(z, 0)) }
1922    else
1923    attacherJoint.upperRotation2 = { -attacherJoint.upperRotation[1], -
        attacherJoint.upperRotation[2], -attacherJoint.upperRotation[3] }
1924    end
1925    end
1926
1927    attacherJoint.transNode = I3DUtil.indexToObject(self.components,
        getXMLString(xmlFile, baseName..".transNode#node"),
        self.i3dMappings)
1928    if attacherJoint.transNode ~= nil then
1929    attacherJoint.transNodeOrgTrans =
        {getTranslation(attacherJoint.transNode)}
1930    attacherJoint.transNodeHeight = Utils.getNoNil(getXMLFloat(xmlFile,
        baseName..".transNode#height"), 0.12)
1931    attacherJoint.transNodeMinY = getXMLFloat(xmlFile,
        baseName..".transNode#minY")
1932    attacherJoint.transNodeMaxY = getXMLFloat(xmlFile,
        baseName..".transNode#maxY")
1933    end
1934
```

```
1935  -- lowerDistanceToGround is a mandatory attribute if a rotationNode
      is available
1936  if (attacherJoint.rotationNode ~= nil or attacherJoint.transNode ~=
      nil) and getXMLFloat(xmlFile, baseName..".distanceToGround#lower")
      == nil then
1937  g_logManager:xmlWarning(self.configFileName, "Missing
      '.distanceToGround#lower' for attacherJoint '%s'. Use console
      command 'gsVehicleAnalyze' to get correct values!", baseName)
1938  end
1939  attacherJoint.lowerDistanceToGround =
      Utils.getNoNil(getXMLFloat(xmlFile,
      baseName..".distanceToGround#lower"), 0.7)
1940
1941  -- upperDistanceToGround is a mandatory attribute if a rotationNode
      is available
1942  if (attacherJoint.rotationNode ~= nil or attacherJoint.transNode ~=
      nil) and getXMLFloat(xmlFile, baseName..".distanceToGround#upper")
      == nil then
1943  g_logManager:xmlWarning(self.configFileName, "Missing
      '.distanceToGround#upper' for attacherJoint '%s'. Use console
      command 'gsVehicleAnalyze' to get correct values!", baseName)
1944  end
1945  attacherJoint.upperDistanceToGround =
      Utils.getNoNil(getXMLFloat(xmlFile,
      baseName..".distanceToGround#upper"), 1.0)
1946
1947  if attacherJoint.lowerDistanceToGround >
      attacherJoint.upperDistanceToGround then
1948  g_logManager:xmlWarning(self.configFileName,
      "distanceToGround#lower may not be larger than
      distanceToGround#upper for attacherJoint '%s'. Switching values!",
      baseName)
1949  local copy = attacherJoint.lowerDistanceToGround
1950  attacherJoint.lowerDistanceToGround =
      attacherJoint.upperDistanceToGround
1951  attacherJoint.upperDistanceToGround = copy
1952  end
1953
1954  attacherJoint.lowerRotationOffset =
      math.rad(Utils.getNoNil(getXMLFloat(xmlFile,
      baseName.."#lowerRotationOffset"), 0))
1955  attacherJoint.upperRotationOffset =
      math.rad(Utils.getNoNil(getXMLFloat(xmlFile,
      baseName.."#upperRotationOffset"), 0))
1956
```

```
1957  attacherJoint.lockDownRotLimit = Utils.getNoNil(getXMLBool(xmlFile, baseName.."#lockDownRotLimit"), false)
1958  attacherJoint.lockUpRotLimit = Utils.getNoNil(getXMLBool(xmlFile, baseName.."#lockUpRotLimit"), false)
1959  -- only use translimit in +y. Set -y to 0
1960  attacherJoint.lockDownTransLimit = Utils.getNoNil(getXMLBool(xmlFile, baseName.."#lockDownTransLimit"), true)
1961  attacherJoint.lockUpTransLimit = Utils.getNoNil(getXMLBool(xmlFile, baseName.."#lockUpTransLimit"), false)
1962
1963  local lowerRotLimitStr = "20 20 20"
1964  if jointType ~= AttacherJoints.JOINTTYPE_IMPLEMENT then
1965  lowerRotLimitStr = "0 0 0"
1966  end
1967  local lx, ly, lz = StringUtil.getVectorFromString(Utils.getNoNil(getXMLString(xmlFile, baseName.."#lowerRotLimit"), lowerRotLimitStr))
1968  attacherJoint.lowerRotLimit = {}
1969  attacherJoint.lowerRotLimit[1] = math.rad(math.abs(Utils.getNoNil(lx, 20)))
1970  attacherJoint.lowerRotLimit[2] = math.rad(math.abs(Utils.getNoNil(ly, 20)))
1971  attacherJoint.lowerRotLimit[3] = math.rad(math.abs(Utils.getNoNil(lz, 20)))
1972  local ux, uy, uz = StringUtil.getVectorFromString(getXMLString(xmlFile, baseName.."#upperRotLimit"))
1973  attacherJoint.upperRotLimit = {}
1974  attacherJoint.upperRotLimit[1] = math.rad(math.abs(Utils.getNoNil(Utils.getNoNil(ux, lx), 20)))
1975  attacherJoint.upperRotLimit[2] = math.rad(math.abs(Utils.getNoNil(Utils.getNoNil(uy, ly), 20)))
1976  attacherJoint.upperRotLimit[3] = math.rad(math.abs(Utils.getNoNil(Utils.getNoNil(uz, lz), 20)))
1977
1978  local lowerTransLimitStr = "0.5 0.5 0.5"
1979  if jointType ~= AttacherJoints.JOINTTYPE_IMPLEMENT then
1980  lowerTransLimitStr = "0 0 0"
1981  end
1982  local lx, ly, lz = StringUtil.getVectorFromString(Utils.getNoNil(getXMLString(xmlFile, baseName.."#lowerTransLimit"), lowerTransLimitStr))
1983  attacherJoint.lowerTransLimit = {}
1984  attacherJoint.lowerTransLimit[1] = math.abs(Utils.getNoNil(lx, 0))
```

```
1985  attacherJoint.lowerTransLimit[2] = math.abs(Utils.getNoNil(ly, 0))
1986  attacherJoint.lowerTransLimit[3] = math.abs(Utils.getNoNil(lz, 0))
1987  local ux, uy, uz =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      baseName.."#upperTransLimit"))
1988  attacherJoint.upperTransLimit = {}
1989  attacherJoint.upperTransLimit[1] =
      math.abs(Utils.getNoNil(Utils.getNoNil(ux, lx), 0))
1990  attacherJoint.upperTransLimit[2] =
      math.abs(Utils.getNoNil(Utils.getNoNil(uy, ly), 0))
1991  attacherJoint.upperTransLimit[3] =
      math.abs(Utils.getNoNil(Utils.getNoNil(uz, lz), 0))
1992
1993
1994  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      baseName.."#jointPositionOffset"))
1995  attacherJoint.jointPositionOffset = {}
1996  attacherJoint.jointPositionOffset[1] = Utils.getNoNil(x, 0)
1997  attacherJoint.jointPositionOffset[2] = Utils.getNoNil(y, 0)
1998  attacherJoint.jointPositionOffset[3] = Utils.getNoNil(z, 0)
1999
2000  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      baseName.."#rotLimitSpring"))
2001  attacherJoint.rotLimitSpring = { Utils.getNoNil(x, 0),
      Utils.getNoNil(y, 0), Utils.getNoNil(z, 0) }
2002  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      baseName.."#rotLimitDamping"))
2003  attacherJoint.rotLimitDamping = { Utils.getNoNil(x, 1),
      Utils.getNoNil(y, 1), Utils.getNoNil(z, 1) }
2004  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      baseName.."#rotLimitForceLimit"))
2005  attacherJoint.rotLimitForceLimit = { Utils.getNoNil(x, -1),
      Utils.getNoNil(y, -1), Utils.getNoNil(z, -1) }
2006
2007  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      baseName.."#transLimitSpring"))
2008  attacherJoint.transLimitSpring = { Utils.getNoNil(x, 0),
      Utils.getNoNil(y, 0), Utils.getNoNil(z, 0) }
2009  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      baseName.."#transLimitDamping"))
```

```
2010  attacherJoint.transLimitDamping = { Utils.getNoNil(x, 1),
      Utils.getNoNil(y, 1), Utils.getNoNil(z, 1) }
2011  local x, y, z =
      StringUtil.getVectorFromString(getXMLString(xmlFile,
      baseName.."#transLimitForceLimit"))
2012  attacherJoint.transLimitForceLimit = { Utils.getNoNil(x, -1),
      Utils.getNoNil(y, -1), Utils.getNoNil(z, -1) }
2013
2014  attacherJoint.moveDefaultTime = Utils.getNoNil(getXMLFloat(xmlFile,
      baseName.."#moveTime"), 0.5)*1000
2015  attacherJoint.moveTime = attacherJoint.moveDefaultTime
2016
2017  attacherJoint.enableCollision = Utils.getNoNil(getXMLBool(xmlFile,
      baseName.."#enableCollision"), false)
2018
2019  local topArmFilename = getXMLString(xmlFile, baseName..
      ".topArm#filename")
2020  if topArmFilename ~= nil then
2021  local baseNode = I3DUtil.indexToObject(self.components,
      getXMLString(xmlFile, baseName.. ".topArm#baseNode"),
      self.i3dMappings)
2022  if baseNode ~= nil then
2023  local i3dNode =
      g_i3DManager:loadSharedI3DFile(topArmFilename,self.baseDirectory,
      false, false, false)
2024  if i3dNode ~= 0 then
2025  local rootNode = getChildAt(i3dNode, 0)
2026  link(baseNode, rootNode)
2027  delete(i3dNode)
2028  setTranslation(rootNode, 0,0,0)
2029  local translationNode = getChildAt(rootNode, 0)
2030  local referenceNode = getChildAt(translationNode, 0)
2031
2032
2033  local topArm = {}
2034  topArm.rotationNode = rootNode
2035  topArm.rotX, topArm.rotY, topArm.rotZ = 0,0,0
2036  topArm.translationNode = translationNode
2037
2038  local _,_,referenceDistance = getTranslation(referenceNode)
2039  topArm.referenceDistance = referenceDistance
2040
2041  topArm.zScale = 1
```

```
2042  local zScale = MathUtil.sign(Utils.getNoNil(getXMLFloat(xmlFile,
      baseName.. ".topArm#zScale"), 1))
2043  if zScale < 0 then
2044  topArm.rotY = math.pi
2045  setRotation(rootNode, topArm.rotX, topArm.rotY, topArm.rotZ)
2046  end
2047
2048  if getNumOfChildren(rootNode) > 1 then
2049  topArm.scaleNode = getChildAt(rootNode, 1)
2050  local scaleReferenceNode = getChildAt(topArm.scaleNode, 0)
2051  local _,_,scaleReferenceDistance =
      getTranslation(scaleReferenceNode)
2052  topArm.scaleReferenceDistance = scaleReferenceDistance
2053  end
2054
2055  topArm.toggleVisibility = Utils.getNoNil(getXMLBool(xmlFile,
      baseName.. ".topArm#toggleVisibility"), false)
2056  if topArm.toggleVisibility then
2057  setVisibility(topArm.rotationNode, false)
2058  end
2059
2060  local colorValueStr = getXMLString(xmlFile,
      baseName..".topArm#color")
2061  local colorValue =
      g_brandColorManager:getBrandColorByName(colorValueStr)
2062  if colorValue == nil then
2063  colorValue = StringUtil.getVectorNFromString(colorValueStr, 3)
2064  end
2065
2066  local colorValue2Str = getXMLString(xmlFile,
      baseName..".topArm#color2")
2067  local colorValue2 =
      g_brandColorManager:getBrandColorByName(colorValue2Str)
2068  if colorValue2 == nil then
2069  colorValue2 = StringUtil.getVectorNFromString(colorValue2Str, 3)
2070  end
2071
2072  if colorValue ~= nil then
2073  local material = getXMLInt(xmlFile, baseName..".topArm#material")
2074  I3DUtil.setShaderParameterRec(rootNode, "colorMat0", colorValue[1],
      colorValue[2], colorValue[3], material or colorValue[4])
2075  end
```

```
2076  if colorValue2 ~= nil then
2077  local material2 = getXMLInt(xmlFile, baseName..".topArm#material2")
2078  I3DUtil.setShaderParameterRec(rootNode, "colorMat1",
      colorValue2[1], colorValue2[2], colorValue2[3], material2 or
      colorValue2[4])
2079  end
2080
2081  attacherJoint.topArm = topArm
2082  end
2083  end
2084  else
2085  local rotationNode = I3DUtil.indexToObject(self.components,
      getXMLString(xmlFile, baseName.. ".topArm#rotationNode"),
      self.i3dMappings)
2086  local translationNode = I3DUtil.indexToObject(self.components,
      getXMLString(xmlFile, baseName.. ".topArm#translationNode"),
      self.i3dMappings)
2087  local referenceNode = I3DUtil.indexToObject(self.components,
      getXMLString(xmlFile, baseName.. ".topArm#referenceNode"),
      self.i3dMappings)
2088  if rotationNode ~= nil then
2089  local topArm = {}
2090  topArm.rotationNode = rotationNode
2091  topArm.rotX, topArm.rotY, topArm.rotZ = getRotation(rotationNode)
2092  if translationNode ~= nil and referenceNode ~= nil then
2093  topArm.translationNode = translationNode
2094
2095  local x,y,z = getTranslation(translationNode)
2096  if math.abs(x) >= 0.0001 or math.abs(y) >= 0.0001 or math.abs(z) >=
      0.0001 then
2097  g_logManager:xmlWarning(self.configFileName, "TopArm translation of
      attacherJoint '%s' is not 0/0/0!", baseName)
2098  end
2099  topArm.referenceDistance = calcDistanceFrom(referenceNode,
      translationNode)
2100  end
2101  topArm.zScale = MathUtil.sign(Utils.getNoNil(getXMLFloat(xmlFile,
      baseName.. ".topArm#zScale"), 1))
2102  topArm.toggleVisibility = Utils.getNoNil(getXMLBool(xmlFile,
      baseName.. ".topArm#toggleVisibility"), false)
2103  if topArm.toggleVisibility then
2104  setVisibility(topArm.rotationNode, false)
2105  end
```

```
2106
2107    attacherJoint.topArm = topArm
2108    end
2109    end
2110
2111    local rotationNode = I3DUtil.indexToObject(self.components,
        getXMLString(xmlFile, baseName.. ".bottomArm#rotationNode"),
        self.i3dMappings)
2112    local translationNode = I3DUtil.indexToObject(self.components,
        getXMLString(xmlFile, baseName.. ".bottomArm#translationNode"),
        self.i3dMappings)
2113    local referenceNode = I3DUtil.indexToObject(self.components,
        getXMLString(xmlFile, baseName.. ".bottomArm#referenceNode"),
        self.i3dMappings)
2114    if rotationNode ~= nil then
2115    local bottomArm = {}
2116    bottomArm.rotationNode = rotationNode
2117    local startRot =
        StringUtil.getRadiansFromString(getXMLString(xmlFile,
        baseName..".bottomArm#startRotation"), 3)
2118    if startRot ~= nil then
2119    bottomArm.rotX, bottomArm.rotY, bottomArm.rotZ =
        startRot[1],startRot[2],startRot[3]
2120    else
2121    bottomArm.rotX, bottomArm.rotY, bottomArm.rotZ =
        getRotation(rotationNode)
2122    end
2123    if translationNode ~= nil and referenceNode ~= nil then
2124    bottomArm.translationNode = translationNode
2125
2126    local x,y,z = getTranslation(translationNode)
2127    if math.abs(x) >= 0.0001 or math.abs(y) >= 0.0001 or math.abs(z) >=
        0.0001 then
2128    g_logManager:xmlWarning(self.configFileName, "BottomArm translation
        of attacherJoint '%s' is not 0/0/0!", baseName)
2129    end
2130    bottomArm.referenceDistance = calcDistanceFrom(referenceNode,
        translationNode)
2131    end
2132    bottomArm.zScale =
        MathUtil.sign(Utils.getNoNil(getXMLFloat(xmlFile, baseName..
        ".bottomArm#zScale"), 1))
2133    bottomArm.lockDirection = Utils.getNoNil(getXMLBool(xmlFile,
        baseName.. ".bottomArm#lockDirection"), true)
```

```
2134
2135  bottomArm.toggleVisibility = Utils.getNoNil(getXMLBool(xmlFile,
      baseName.. ".bottomArm#toggleVisibility"), false)
2136  if bottomArm.toggleVisibility then
2137  setVisibility(bottomArm.rotationNode, false)
2138  end
2139
2140  if jointType == AttacherJoints.JOINTTYPE_IMPLEMENT then
2141  local toolbarFilename = Utils.getNoNil(getXMLString(xmlFile,
      baseName.. ".toolbar#filename"), "$data/shared/assets/toolbar.i3d")
2142  local i3dNode = g_i3DManager:loadSharedI3DFile(toolbarFilename,
      self.baseDirectory, false, false, false)
2143  if i3dNode ~= 0 then
2144  local rootNode = getChildAt(i3dNode, 0)
2145  link(referenceNode, rootNode)
2146  delete(i3dNode)
2147  setTranslation(rootNode, 0,0,0)
2148  bottomArm.toolbar = rootNode
2149  setVisibility(rootNode, false)
2150  end
2151  end
2152
2153  attacherJoint.bottomArm = bottomArm
2154  end
2155
2156  if self.isClient then
2157  attacherJoint.sampleAttach =
      g_soundManager:loadSampleFromXML(xmlFile, baseName, "attachSound",
      self.baseDirectory, self.components, 1, AudioGroup.VEHICLE,
      self.i3dMappings, self)
2158  end
2159
2160  attacherJoint.steeringBarLeftNode =
      I3DUtil.indexToObject(self.components, getXMLString(xmlFile,
      baseName.. ".steeringBars#leftNode"), self.i3dMappings)
2161  attacherJoint.steeringBarRightNode =
      I3DUtil.indexToObject(self.components, getXMLString(xmlFile,
      baseName.. ".steeringBars#rightNode"), self.i3dMappings)
2162
2163  attacherJoint.changeObjects = {}
2164  ObjectChangeUtil.loadObjectChangeFromXML(xmlFile, baseName,
      attacherJoint.changeObjects, self.components, self)
```

```
2165  -- ObjectChangeUtil.setObjectChanges(attacherJoint.changeObjects,
      false, self, self.setMovingToolDirty)
2166
2167  attacherJoint.rootNode =
      Utils.getNoNil(I3DUtil.indexToObject(self.components,
      getXMLString(xmlFile, baseName.."#rootNode"), self.i3dMappings),
      self.components[1].node)
2168  attacherJoint.rootNodeBackup = attacherJoint.rootNode
2169  attacherJoint.jointIndex = 0
2170
2171  local t = getXMLFloat(xmlFile, baseName .. "#comboTime")
2172  if t ~= nil then
2173  table.insert(spec.attacherJointCombos.joints, {jointIndex = index +
      1, time = MathUtil.clamp(t, 0, 1) *
      spec.attacherJointCombos.duration})
2174  end
2175
2176  local schemaKey = baseName.. ".schema"
2177  if hasXMLProperty(xmlFile, schemaKey) then
2178  local x, y = StringUtil.getVectorFromString(getXMLString(xmlFile,
      schemaKey .. "#position"))
2179  local liftedOffsetX, liftedOffsetY =
      StringUtil.getVectorFromString(Utils.getNoNil(getXMLString(xmlFile,
      schemaKey.."#liftedOffset"), "0 5"))
2180
2181  self.schemaOverlay:addAttacherJoint(x, y,
2182  math.rad(Utils.getNoNil(getXMLFloat(xmlFile, schemaKey ..
      "#rotation"), 0)),
2183  Utils.getNoNil(getXMLBool(xmlFile, schemaKey .. "#invertX"),
      false),
2184  liftedOffsetX, liftedOffsetY)
2185  else
2186  g_logManager:xmlWarning(self.configFileName, "Missing schema
      overlay attacherJoint '%s'!", baseName)
2187  end
2188
2189  return true
2190  end
```

addToPhysics

Description

Add to physics

Definition

addToPhysics()

Return Values

boolean success success

Code

```lua
function AttacherJoints:addToPhysics(superFunc)
local spec = self.spec_attacherJoints

if not superFunc(self) then
return false
end

for _, implement in pairs(spec.attachedImplements) do
if not implement.object.isHardAttached then
self:createAttachmentJoint(implement)
end
end

return true
end
```

getTotalMass

Description

Returns total mass of vehicle (optional including attached vehicles)

Definition

getTotalMass(boolean onlyGivenVehicle)

Arguments

boolean onlyGivenVehicle use only the given vehicle, if false or nil it includes all attachables

Return Values

float totalMass total mass

Code

```lua
function AttacherJoints:getTotalMass(superFunc,
onlyGivenVehicle)
local spec = self.spec_attacherJoints
local mass = superFunc(self)

if onlyGivenVehicle == nil or not onlyGivenVehicle then
for _, implement in pairs(spec.attachedImplements) do
local object = implement.object
mass = mass + object:getTotalMass(onlyGivenVehicle)
end
end

return mass
end
```

getChildVehicles

Description

Inserts all child vehicles into the given table

Definition

getChildVehicles(table vehicles)

Arguments

table vehicles child vehicles table

Code

```
2275  function AttacherJoints:getChildVehicles(superFunc, vehicles)
2276  local spec = self.spec_attacherJoints
2277
2278  for _, implement in pairs(spec.attachedImplements) do
2279  implement.object:getChildVehicles(vehicles)
2280  end
2281
2282  return superFunc(self, vehicles)
2283  end
```

getAirConsumerUsage

Description

Returns air consumer usage of attached vehicles

Definition

getAirConsumerUsage()

Return Values

float usage air usage

Code

```
2288  function AttacherJoints:getAirConsumerUsage(superFunc)
2289  local spec = self.spec_attacherJoints
2290  local usage = superFunc(self)
2291
2292  for _, implement in pairs(spec.attachedImplements) do
2293  local object = implement.object
2294  if object.getAttachbleAirConsumerUsage ~= nil then
2295  usage = usage + object:getAttachbleAirConsumerUsage()
2296  end
2297  end
2298
2299  return usage
2300  end
```

isDetachAllowed

Description

Returns true if detach is allowed

Definition

isDetachAllowed()

Return Values

boolean detachAllowed detach is allowed

Code

```
2425  function AttacherJoints:isDetachAllowed(superFunc)
2426  if not superFunc(self) then
2427  return false
2428  end
2429
2430  local spec = self.spec_attacherJoints
2431  for attacherJointIndex, attacherJoint in
      ipairs(spec.attacherJoints) do
2432  if not attacherJoint.allowDetachingWhileLifted then
2433  if not attacherJoint.moveDown then
2434  local implement =
      self:getImplementByJointDescIndex(attacherJointIndex)
2435  if implement ~= nil then
2436  return false,
      string.format(g_i18n:getText("warning_lowerImplementFirst"),
      implement.object.typeDesc)
2437  end
2438  end
2439  end
2440  end
2441
2442  return true
2443  end
```

onActivate

Description

Called on activate

Definition

onActivate()

Code

```
2491  function AttacherJoints:onActivate()
2492  self:activateAttachments()
2493  end
```

onDeactivate

Description

Called on deactivate

Definition

onDeactivate()

Code

```
2497  function AttacherJoints:onDeactivate()
2498  self:deactivateAttachments()
2499  if self.isClient then
2500  local spec = self.spec_attacherJoints
2501  g_soundManager:stopSample(spec.samples.hydraulic)
2502  spec.isHydraulicSamplePlaying = false
2503  end
2504  end
```

onDeactivateLights

Description

Called on deactivating lights

Definition

onDeactivateLights()

Code

```
2518  function AttacherJoints:onDeactivateLights()
2519  self:deactivateAttachmentsLights()
2520  end
```

BaleGrab

Description

Class for a balegrab tool

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
38  function BaleGrab:onLoad(savegame)
39  local spec = self.spec_baleGrab
40
41  if self.isServer then
42  local attacherTriggerTriggerNode =
    I3DUtil.indexToObject(self.components, getXMLString(self.xmlFile,
    "vehicle.baleGrab#triggerNode"), self.i3dMappings)
43  local attacherTriggerRootNode =
    I3DUtil.indexToObject(self.components, getXMLString(self.xmlFile,
    "vehicle.baleGrab#rootNode"), self.i3dMappings)
44  local attacherTriggerJointNode =
    I3DUtil.indexToObject(self.components, getXMLString(self.xmlFile,
    "vehicle.baleGrab#jointNode"), self.i3dMappings)
```

```
45  local attacherJointTypeString =
    Utils.getNoNil(getXMLString(self.xmlFile,
    "vehicle.baleGrab#jointType"), "TYPE_AUTO_ATTACH_XYZ")
46  local attacherJointType = DynamicMountUtil.TYPE_AUTO_ATTACH_XYZ
47  if DynamicMountUtil[attacherJointTypeString] ~= nil then
48  attacherJointType = DynamicMountUtil[attacherJointTypeString]
49  end
50
51  if attacherTriggerTriggerNode ~= nil and attacherTriggerRootNode
    ~= nil and attacherTriggerJointNode ~= nil then
52  local forceAcceleration = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.baleGrab#forceAcceleration"), 20)
53  addTrigger(attacherTriggerTriggerNode, "baleGrabTriggerCallback",
    self)
54
55  local grabRefComponentJointIndex1 = getXMLInt(self.xmlFile,
    "vehicle.baleGrab#grabRefComponentJointIndex1")
56  local grabRefComponentJointIndex2 = getXMLInt(self.xmlFile,
    "vehicle.baleGrab#grabRefComponentJointIndex2")
57  local componentJoint1, componentJoint2
58  if grabRefComponentJointIndex1 ~= nil then
59  componentJoint1 =
    self.componentJoints[grabRefComponentJointIndex1+1]
60  end
61  if grabRefComponentJointIndex2 ~= nil then
62  componentJoint2 =
    self.componentJoints[grabRefComponentJointIndex2+1]
63  end
64  local rotDiffThreshold1 =
    math.rad(Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.baleGrab#rotDiffThreshold1"), 2))
65  local rotDiffThreshold2 =
    math.rad(Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.baleGrab#rotDiffThreshold2"), 2))
66  spec.dynamicMountAttacherTrigger = {
67  triggerNode=attacherTriggerTriggerNode,
    rootNode=attacherTriggerRootNode,
    jointNode=attacherTriggerJointNode,
    attacherJointType=attacherJointType,
    forceAcceleration=forceAcceleration,
68  componentJoint1=componentJoint1,
    rotDiffThreshold1=rotDiffThreshold1,
    cosRotDiffThreshold1=math.cos(rotDiffThreshold1),
69  componentJoint2=componentJoint2,
    rotDiffThreshold2=rotDiffThreshold2,
    cosRotDiffThreshold2=math.cos(rotDiffThreshold2)
```

```
70  }
71  end
72  spec.dynamicMountedObjects = {}
73  spec.pendingDynamicMountObjects = {}
74  end
75  end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
79  function BaleGrab:onDelete()
80  local spec = self.spec_baleGrab
81  if self.isServer then
82  for object,_ in pairs(spec.dynamicMountedObjects) do
83  object:unmountDynamic()
84  end
85  end
86  if spec.dynamicMountAttacherTrigger ~= nil then
87  removeTrigger(spec.dynamicMountAttacherTrigger.triggerNode)
88  end
89  end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
96  function BaleGrab:onUpdateTick(dt, isActiveForInput, isSelected)
97  if self.isServer then
98  local spec = self.spec_baleGrab
99  local attachTrigger = spec.dynamicMountAttacherTrigger
100
101 local isClosed = true
102 if attachTrigger.componentJoint1 ~= nil then
```

```
103 isClosed = self:isComponentJointOutsideLimit(attachTrigger.componentJoint1, attachTrigger.rotDiffThreshold1, attachTrigger.cosRotDiffThreshold1)
104 end
105 if isClosed and attachTrigger.componentJoint2 ~= nil then
106 isClosed = self:isComponentJointOutsideLimit(attachTrigger.componentJoint2, attachTrigger.rotDiffThreshold2, attachTrigger.cosRotDiffThreshold2)
107 end
108 if isClosed then
109 for object,_ in pairs(spec.pendingDynamicMountObjects) do
110 if spec.dynamicMountedObjects[object] == nil then
111 object:unmountDynamic()
112 local dynamicMountData = spec.dynamicMountAttacherTrigger
113 if object:mountDynamic(self, dynamicMountData.rootNode, dynamicMountData.jointNode, dynamicMountData.attacherJointType, dynamicMountData.forceAcceleration) then
114 self:addDynamicMountedObject(object)
115 end
116 end
117 end
118 else
119 for object,_ in pairs(spec.dynamicMountedObjects) do
120 self:removeDynamicMountedObject(object, false)
121 object:unmountDynamic()
122 end
123 end
124 end
125 end
```

addDynamicMountedObject

Description

Add dynamic mount object

Definition

addDynamicMountedObject(table object)

Arguments

table object object

Code

```
130 function BaleGrab:addDynamicMountedObject(object)
131 local spec = self.spec_baleGrab
132 spec.dynamicMountedObjects[object] = object
133 end
```

removeDynamicMountedObject

Description

Remove dynamic mount object

Definition

removeDynamicMountedObject(table object, boolean isDeleting)

Arguments

table	object	object
boolean	isDeleting	is deleting

Code

```
function BaleGrab:removeDynamicMountedObject(object, isDeleting)
local spec = self.spec_baleGrab
spec.dynamicMountedObjects[object] = nil
if isDeleting then
spec.pendingDynamicMountObjects[object] = nil
end
end
```

baleGrabTriggerCallback

Description

Trigger callback

Definition

baleGrabTriggerCallback(integer triggerId, integer otherActorId, boolean onEnter, boolean onLeave, boolean onStay, integer otherShapeId)

Arguments

integer	triggerId	id of trigger
integer	otherActorId	id of other actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay
integer	otherShapeId	id of other shape

Code

```
function BaleGrab:baleGrabTriggerCallback(triggerId, otherActorId, onEnter, onLeave, onStay, otherShapeId)
local spec = self.spec_baleGrab
if onEnter then
local object = g_currentMission:getNodeObject(otherActorId)
if object == nil then
object = g_currentMission.nodeToObject[otherActorId]
end
if object ~= nil and object ~= self and object.getSupportsMountDynamic ~= nil and object:getSupportsMountDynamic() then
spec.pendingDynamicMountObjects[object] = Utils.getNoNil(spec.pendingDynamicMountObjects[object], 0) + 1
```

```
164 end
165 elseif onLeave then
166 local object = g_currentMission:getNodeObject(otherActorId)
167 if object == nil then
168 object = g_currentMission.nodeToObject[otherActorId]
169 end
170 if object ~= nil then
171 if spec.pendingDynamicMountObjects[object] ~= nil then
172 local count = spec.pendingDynamicMountObjects[object]-1
173 if count == 0 then
174 spec.pendingDynamicMountObjects[object] = nil
175
176 if spec.dynamicMountedObjects[object] ~= nil then
177 self:removeDynamicMountedObject(object, false)
178 object:unmountDynamic()
179 end
180 else
181 spec.pendingDynamicMountObjects[object] = count
182 end
183 end
184 end
185 end
186 end
```

isComponentJointOutsideLimit

Description

Returns if component joint is outside the rotation limit

Definition

isComponentJointOutsideLimit(integer componentJoint, float maxRot, float cosMaxRot)

Arguments

integer	componentJoint	index of component joint
float	maxRot	max rotation
float	cosMaxRot	cos max rotation

Return Values

boolean isOutside is outside the rotation limit

Code

```
194 function BaleGrab:isComponentJointOutsideLimit(componentJoint, maxRot,
    cosMaxRot)
195 local x,_,z =
    localDirectionToLocal(self.components[componentJoint.componentIndices[2]
    componentJoint.jointNode, 0,0,1)
196 if (x >= 0) == (maxRot >= 0) then
197 if z <= cosMaxRot*math.sqrt(x*x + z*z) then
```

```
198  return true
199  end
200  end
201  return false
202  end
```

BaleLoader

Description

This is the specialization for automatic bale loaders

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
20  function BaleLoader.prerequisitesPresent(specializations)
21  return SpecializationUtil.hasSpecialization(FillUnit,
    specializations)
22  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
125  function BaleLoader:onLoad(savegame)
126  local spec = self.spec_baleLoader
127
128  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
     self.configFileName,
     "vehicle.baleloaderTurnedOnScrollers.baleloaderTurnedOnScroller",
     "vehicle.baleLoader.animationNodes.animationNode") --FS17 to FS19
129  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
     self.configFileName, "vehicle.baleGrabber",
     "vehicle.baleLoader.grabber") --FS17 to FS19
130  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
     self.configFileName, "vehicle.balePlaces",
     "vehicle.baleLoader.balePlaces") --FS17 to FS19
```

```
131 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.grabParticleSystem",
    "vehicle.baleLoader.grabber.grabParticleSystem") --FS17 to FS19
132 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.baleLoader#pickupRange",
    "vehicle.baleLoader.grabber#pickupRange") --FS17 to FS19
133 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.baleTypes",
    "vehicle.baleLoader.baleTypes") --FS17 to FS19
134
135 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.baleLoader#textTransportPosition",
    "vehicle.baleLoader.texts#transportPosition") --FS17 to FS19
136 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.baleLoader#textOperatingPosition",
    "vehicle.baleLoader.texts#operatingPosition") --FS17 to FS19
137 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.baleLoader#textUnload",
    "vehicle.baleLoader.texts#unload") --FS17 to FS19
138 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.baleLoader#textTilting",
    "vehicle.baleLoader.texts#tilting") --FS17 to FS19
139 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.baleLoader#textLowering",
    "vehicle.baleLoader.texts#lowering") --FS17 to FS19
140 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.baleLoader#textLowerPlattform",
    "vehicle.baleLoader.texts#lowerPlattform") --FS17 to FS19
141 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.baleLoader#textAbortUnloading",
    "vehicle.baleLoader.texts#abortUnloading") --FS17 to FS19
142 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.baleLoader#textUnloadHere",
    "vehicle.baleLoader.texts#unloadHere") --FS17 to FS19
143
144 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.baleLoader#rotatePlatformAnimName",
    "vehicle.baleLoader.animations#rotatePlatform") --FS17 to FS19
145 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.baleLoader#rotatePlatformBackAnimName",
    "vehicle.baleLoader.animations#rotatePlatformBack") --FS17 to
    FS19
146 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.baleLoader#rotatePlatformEmptyAnimName",
    "vehicle.baleLoader.animations#rotatePlatformEmpty") --FS17 to
    FS19
```

```
147
148 local baseKey = "vehicle.baleLoader"
149
150 spec.balesToLoad = {}
151
152 spec.balesToMount = {}
153
154 spec.isInWorkPosition = false
155 spec.grabberIsMoving = false
156 spec.synchronizeFillLevel = false
157 spec.synchronizeFullFillLevel = true
158
159 spec.rotatePlatformDirection = 0
160 spec.frontBalePusherDirection = 0
161
162 spec.emptyState = BaleLoader.EMPTY_NONE
163 spec.itemsToSave = {}
164
165 spec.texts = {}
166 spec.texts.transportPosition =
    Utils.getNoNil(getXMLString(self.xmlFile,
    baseKey..".texts#transportPosition"),
    "action_baleloaderTransportPosition")
167 spec.texts.operatingPosition =
    Utils.getNoNil(getXMLString(self.xmlFile,
    baseKey..".texts#operatingPosition"),
    "action_baleloaderOperatingPosition")
168 spec.texts.unload = Utils.getNoNil(getXMLString(self.xmlFile,
    baseKey..".texts#unload"), "action_baleloaderUnload")
169 spec.texts.tilting = Utils.getNoNil(getXMLString(self.xmlFile,
    baseKey..".texts#tilting"), "info_baleloaderTiltingTable")
170 spec.texts.lowering = Utils.getNoNil(getXMLString(self.xmlFile,
    baseKey..".texts#lowering"), "info_baleloaderLoweringTable")
171 spec.texts.lowerPlattform =
    Utils.getNoNil(getXMLString(self.xmlFile,
    baseKey..".texts#lowerPlattform"),
    "action_baleloaderLowerPlatform")
172 spec.texts.abortUnloading =
    Utils.getNoNil(getXMLString(self.xmlFile,
    baseKey..".texts#abortUnloading"),
    "action_baleloaderAbortUnloading")
173 spec.texts.unloadHere = Utils.getNoNil(getXMLString(self.xmlFile,
    baseKey..".texts#unloadHere"), "action_baleloaderUnloadHere")
174
```

```
175 spec.animations = {}
176 spec.animations.rotatePlatform =
    Utils.getNoNil(getXMLString(self.xmlFile,
    baseKey..".animations#rotatePlatform"), "rotatePlatform")
177 spec.animations.rotatePlatformBack =
    Utils.getNoNil(getXMLString(self.xmlFile,
    baseKey..".animations#rotatePlatformBack"), "rotatePlatform")
178 spec.animations.rotatePlatformEmpty =
    Utils.getNoNil(getXMLString(self.xmlFile,
    baseKey..".animations#rotatePlatformEmpty"), "rotatePlatform")
179
180 spec.moveBalePlacesAfterRotatePlatform =
    Utils.getNoNil(getXMLBool(self.xmlFile,
    baseKey.."#moveBalePlacesAfterRotatePlatform"), false)
181 spec.alwaysMoveBalePlaces =
    Utils.getNoNil(getXMLBool(self.xmlFile,
    baseKey.."#alwaysMoveBalePlaces"), false)
182 spec.transportPositionAfterUnloading =
    Utils.getNoNil(getXMLBool(self.xmlFile,
    baseKey.."#transportPositionAfterUnloading"), true)
183
184 spec.automaticUnloading = Utils.getNoNil(getXMLBool(self.xmlFile,
    baseKey.."#automaticUnloading"), false)
185 spec.resetEmptyRotateAnimation =
    Utils.getNoNil(getXMLBool(self.xmlFile,
    baseKey.."#resetEmptyRotateAnimation"), true)
186
187 spec.mountDynamic = Utils.getNoNil(getXMLBool(self.xmlFile,
    baseKey..".dynamicMount#enabled"), false)
188 spec.updateBaleJointNodePosition = {}
189 spec.dynamicMountMinTransLimits =
    StringUtil.getVectorNFromString(getXMLString(self.xmlFile,
    baseKey..".dynamicMount#minTransLimits"), 3)
190 spec.dynamicMountMaxTransLimits =
    StringUtil.getVectorNFromString(getXMLString(self.xmlFile,
    baseKey..".dynamicMount#maxTransLimits"), 3)
191
192 spec.fillUnitIndex = Utils.getNoNil(getXMLInt(self.xmlFile,
    baseKey.."#fillUnitIndex"), 1)
193
194 spec.baleGrabber = {}
195 spec.baleGrabber.grabNode =
    I3DUtil.indexToObject(self.components, getXMLString(self.xmlFile,
    baseKey..".grabber#grabNode"), self.i3dMappings)
196 spec.baleGrabber.pickupRange =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    baseKey..".grabber#pickupRange"), 3.0)
```

```
197 spec.baleGrabber.balesInTrigger = {}
198 spec.baleGrabber.trigger = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseKey..".grabber#triggerNode"),
    self.i3dMappings)
199 if spec.baleGrabber.trigger ~= nil then
200 addTrigger(spec.baleGrabber.trigger,
    "baleGrabberTriggerCallback", self)
201 else
202 g_logManager:xmlError(self.configFileName, "Bale grabber needs a
    valid trigger!")
203 end
204
205 spec.startBalePlace = {}
206 spec.startBalePlace.bales = {}
207 spec.startBalePlace.node = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile,
    baseKey..".balePlaces#startBalePlace"), self.i3dMappings)
208 if spec.startBalePlace.node ~= nil then
209 spec.startBalePlace.numOfPlaces =
    getNumOfChildren(spec.startBalePlace.node)
210 if spec.startBalePlace.numOfPlaces == 0 then
211 spec.startBalePlace.node = nil
212 else
213 spec.startBalePlace.origRot = {}
214 spec.startBalePlace.origTrans = {}
215 for i=1, spec.startBalePlace.numOfPlaces do
216 local node = getChildAt(spec.startBalePlace.node, i-1)
217 spec.startBalePlace.origRot[i] = {getRotation(node)}
218 spec.startBalePlace.origTrans[i] = {getTranslation(node)}
219 end
220 end
221 end
222 spec.startBalePlace.count = 0
223
224 spec.currentBalePlace = 1
225 spec.balePlaces = {}
226 local i = 0
227 while true do
228 local key = string.format("%s.balePlaces.balePlace(%d)", baseKey,
    i)
229 if not hasXMLProperty(self.xmlFile, key) then
230 break
```

```
231 end
232 local node = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, key.."#node"), self.i3dMappings)
233 local collision = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, key.."#collision"), self.i3dMappings)
234 if node ~= nil then
235 local entry = {}
236 entry.node = node
237 if collision ~= nil then
238 entry.collision = collision
239 setIsCompoundChild(entry.collision, false)
240 end
241 table.insert(spec.balePlaces, entry)
242 end
243 i = i + 1
244 end
245
246 if self.isClient then
247 local grabParticleSystem = {}
248 local psName = baseKey..".grabber.grabParticleSystem"
249 if ParticleUtil.loadParticleSystem(self.xmlFile,
    grabParticleSystem, psName, self.components, false, nil,
    self.baseDirectory) then
250 spec.grabParticleSystem = grabParticleSystem
251 spec.grabParticleSystemDisableTime = 0
252 spec.grabParticleSystemDisableDuration =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    psName.."#disableDuration"), 0.6)*1000
253 end
254
255 spec.samples = {}
256 spec.samples.grab =
    g_soundManager:loadSampleFromXML(self.xmlFile,
    baseKey..".sounds", "grab", self.baseDirectory, self.components,
    1, AudioGroup.VEHICLE, self.i3dMappings, self)
257 end
258
259 spec.allowedBaleTypes = {}
260 i = 0
261 while true do
262 local key = string.format("%s.baleTypes.baleType(%d)", baseKey,
    i)
263 if not hasXMLProperty(self.xmlFile, key) then
```

```
264 break
265 end
266 local minBaleDiameter = getXMLFloat(self.xmlFile,
    key.."#minBaleDiameter")
267 local maxBaleDiameter = getXMLFloat(self.xmlFile,
    key.."#maxBaleDiameter")
268 local minBaleWidth = getXMLFloat(self.xmlFile,
    key.."#minBaleWidth")
269 local maxBaleWidth = getXMLFloat(self.xmlFile,
    key.."#maxBaleWidth")
270 local minBaleHeight = getXMLFloat(self.xmlFile,
    key.."#minBaleHeight")
271 local maxBaleHeight = getXMLFloat(self.xmlFile,
    key.."#maxBaleHeight")
272 local minBaleLength = getXMLFloat(self.xmlFile,
    key.."#minBaleLength")
273 local maxBaleLength = getXMLFloat(self.xmlFile,
    key.."#maxBaleLength")
274 if minBaleDiameter ~= nil and maxBaleDiameter ~= nil and
    minBaleWidth ~= nil and maxBaleWidth ~= nil then
275 table.insert(spec.allowedBaleTypes, {
276 minBaleDiameter = MathUtil.round(minBaleDiameter, 2),
277 maxBaleDiameter = MathUtil.round(maxBaleDiameter, 2),
278 minBaleWidth = MathUtil.round(minBaleWidth, 2),
279 maxBaleWidth = MathUtil.round(maxBaleWidth, 2)})
280 elseif minBaleWidth ~= nil and maxBaleWidth ~= nil and
    minBaleHeight ~= nil and maxBaleHeight ~= nil and minBaleLength
    ~= nil and maxBaleLength ~= nil then
281 table.insert(spec.allowedBaleTypes, {
282 minBaleWidth = MathUtil.round(minBaleWidth, 2),
283 maxBaleWidth = MathUtil.round(maxBaleWidth, 2),
284 minBaleHeight = MathUtil.round(minBaleHeight, 2),
285 maxBaleHeight = MathUtil.round(maxBaleHeight, 2),
286 minBaleLength = MathUtil.round(minBaleLength, 2),
287 maxBaleLength = MathUtil.round(maxBaleLength, 2)
288 })
289 end
290 i = i + 1
291 end
292
293 spec.animationNodes =
    g_animationManager:loadAnimations(self.xmlFile,
    baseKey..".animationNodes", self.components, self,
    self.i3dMappings)
```

```
294 end
```

onPostLoad

Description

Called after loading

Definition

onPostLoad(table savegame)

Arguments

table savegame savegame

Code

```
299 function BaleLoader:onPostLoad(savegame)
300 if savegame ~= nil then
301 local spec = self.spec_baleLoader
302
303 if Utils.getNoNil(getXMLBool(savegame.xmlFile,
    savegame.key..".baleLoader#isInWorkPosition"), false) then
304 if not spec.isInWorkPosition then
305 spec.grabberIsMoving = true
306 spec.isInWorkPosition = true
307 BaleLoader.moveToWorkPosition(self, true)
308 end
309 end
310
311 spec.currentBalePlace = 1
312
313 spec.startBalePlace.count = 0
314 local numBales = 0
315 if not savegame.resetVehicles then
316 local i=0
317 while true do
318 local baleKey =
    savegame.key..string.format(".baleLoader.bale(%d)", i)
319 if not hasXMLProperty(savegame.xmlFile, baleKey) then
320 break
321 end
322 local filename = getXMLString(savegame.xmlFile,
    baleKey.."#filename")
323 if filename ~= nil then
324 filename = NetworkUtil.convertFromNetworkFilename(filename)
325
326 local x,y,z =
    StringUtil.getVectorFromString(getXMLString(savegame.xmlFile,
    baleKey.."#position"))
```

```
327 local xRot,yRot,zRot = StringUtil.getVectorFromString(getXMLString(savegame.xmlFile, baleKey.."#rotation"))
328 local fillLevel = getXMLFloat(savegame.xmlFile, baleKey.."#fillLevel")
329 local balePlace = getXMLInt(savegame.xmlFile, baleKey.."#balePlace")
330 local helper = getXMLInt(savegame.xmlFile, baleKey.."#helper")
331 local farmId = getXMLInt(savegame.xmlFile, baleKey.."#farmId")
332 if balePlace == nil or (balePlace > 0 and (x == nil or y == nil or z == nil or xRot == nil or yRot == nil or zRot == nil)) or (balePlace < 1 and helper == nil) then
333 print("Warning: Corrupt savegame, bale "..filename.." could not be loaded")
334 else
335 xRot = math.rad(xRot)
336 yRot = math.rad(yRot)
337 zRot = math.rad(zRot)
338
339 local translation
340 local rotation
341 if balePlace > 0 then
342 translation = {x,y,z}
343 rotation={xRot, yRot, zRot}
344 else
345 translation = {0,0,0}
346 rotation={0,0,0}
347 end
348 local parentNode = nil
349 local bales = nil
350 if balePlace < 1 then
351 if spec.startBalePlace.node ~= nil and helper <= spec.startBalePlace.numOfPlaces then
352 parentNode = getChildAt(spec.startBalePlace.node, helper-1)
353 if spec.startBalePlace.bales == nil then
354 spec.startBalePlace.bales = {}
355 end
356 bales = spec.startBalePlace.bales
357 spec.startBalePlace.count = spec.startBalePlace.count+1
358 end
359 elseif balePlace <= table.getn(spec.balePlaces) then
```

```
360 spec.currentBalePlace = math.max(spec.currentBalePlace,
    balePlace+1)
361 parentNode = spec.balePlaces[balePlace].node
362 if spec.balePlaces[balePlace].bales == nil then
363 spec.balePlaces[balePlace].bales = {}
364 end
365 bales = spec.balePlaces[balePlace].bales
366 end
367 if parentNode ~= nil then
368 local attributes = {}
369 Bale.loadExtraAttributesFromXMLFile(self, attributes,
    savegame.xmlFile, baleKey, savegame.resetVehicles)
370
371 numBales = numBales + 1
372 table.insert(spec.balesToLoad, {parentNode=parentNode,
    filename=filename, bales=bales, translation=translation,
    rotation=rotation, fillLevel=fillLevel, farmId=farmId,
    attributes=attributes})
373 end
374 end
375 end
376 i = i +1
377 end
378 end
379
380 -- update animations
381 BaleLoader.updateBalePlacesAnimations(self)
382 for i, place in pairs(spec.balePlaces) do
383 if place.collision ~= nil then
384 if i <= numBales then
385 setIsCompoundChild(place.collision, true)
386 else
387 setIsCompoundChild(place.collision, false)
388 end
389 end
390 end
391 end
392 end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
396 function BaleLoader:onDelete()
397 local spec = self.spec_baleLoader
398
399 -- avoid the bale nodes to be deleted twice (because they are
    linked the vehicle and because the Bale object is deleted)
400 for _, balePlace in pairs(spec.balePlaces) do
401 if balePlace.bales ~= nil then
402 for _, baleServerId in pairs(balePlace.bales) do
403 local bale = NetworkUtil.getObject(baleServerId)
404 if bale ~= nil then
405 if spec.mountDynamic then
406 self:unmountDynamicBale(bale)
407 else
408 bale:unmount()
409 end
410
411 -- if the vehicle is reloaded we remove the bale since it will be
    regenerated on load
412 if self.isReconfigurating ~= nil and self.isReconfigurating then
413 bale:delete()
414 end
415 end
416 end
417 end
418 end
419 for _, baleServerId in ipairs(spec.startBalePlace.bales) do
420 local bale = NetworkUtil.getObject(baleServerId)
421 if bale ~= nil then
422 if spec.mountDynamic then
423 self:unmountDynamicBale(bale)
424 else
425 bale:unmount()
426 end
427
428 -- if the vehicle is reloaded we remove the bale since it will be
    regenerated on load
429 if self.isReconfigurating ~= nil and self.isReconfigurating then
430 bale:delete()
431 end
```

```
end
end

if self.isClient then
if spec.grabParticleSystem ~= nil then
ParticleUtil.deleteParticleSystem(spec.grabParticleSystem)
end

for i,sample in pairs(spec.samples) do
g_soundManager:deleteSample(sample)
end
end

if spec.baleGrabber.trigger ~= nil then
removeTrigger(spec.baleGrabber.trigger)
end
end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
function BaleLoader:onReadStream(streamId, connection)
local spec = self.spec_baleLoader

spec.isInWorkPosition = streamReadBool(streamId)
spec.frontBalePusherDirection = streamReadIntN(streamId, 3)
spec.rotatePlatformDirection = streamReadIntN(streamId, 3)

-- note: we do not sync grabberMoveState, this may lead to visual artifacts for a few seconds at the bigging

if spec.isInWorkPosition then
BaleLoader.moveToWorkPosition(self)
end

local emptyState = streamReadUIntN(streamId, 4)
```

```
521
522 spec.currentBalePlace = streamReadInt8(streamId)
523
524 -- read bale at bale grabber
525 if streamReadBool(streamId) then
526 spec.baleGrabber.currentBale =
    NetworkUtil.readNodeObjectId(streamId)
527 spec.balesToMount[spec.baleGrabber.currentBale] =
    {serverId=spec.baleGrabber.currentBale,
    linkNode=spec.baleGrabber.grabNode, trans={0,0,0}, rot={0,0,0} }
528 end
529
530 -- read bales at start bale places
531 spec.startBalePlace.count = streamReadUIntN(streamId, 3)
532 for i=1, spec.startBalePlace.count do
533 local baleServerId = NetworkUtil.readNodeObjectId(streamId)
534
535 local attachNode = getChildAt(spec.startBalePlace.node, i-1)
536 spec.balesToMount[baleServerId] = {serverId=baleServerId,
    linkNode=attachNode, trans={0,0,0}, rot={0,0,0} }
537
538 table.insert(spec.startBalePlace.bales, baleServerId)
539 end
540
541
542 -- read bales at normal bale places
543 for i=1, table.getn(spec.balePlaces) do
544 local balePlace = spec.balePlaces[i]
545
546 local numBales = streamReadUIntN(streamId, 3)
547 if numBales > 0 then
548 balePlace.bales = {}
549
550 for baleI=1, numBales do
551 local baleServerId = NetworkUtil.readNodeObjectId(streamId)
552 local x = streamReadFloat32(streamId)
553 local y = streamReadFloat32(streamId)
554 local z = streamReadFloat32(streamId)
555
556 table.insert(balePlace.bales, baleServerId)
557
```

```
558 spec.balesToMount[baleServerId] = {serverId=baleServerId,
    linkNode=balePlace.node, trans={ x,y,z}, rot={0,0,0} }
559 end
560 end
561 end
562
563 -- read num bales on platform
564 -- read bales on baleloader
565 -- ignore
566
567 -- grabberMoveState ? int:nil
568 -- isInWorkPosition bool
569 -- emptyState ? int:nil
570 -- rotatePlatformDirection
571 -- frontBalePusherDirection
572 -- grabberIsMoving: bool
573
574
575 -- update animations
576 BaleLoader.updateBalePlacesAnimations(self)
577
578 if emptyState >= BaleLoader.EMPTY_TO_WORK then
579 self:doStateChange(BaleLoader.CHANGE_EMPTY_START)
580 AnimatedVehicle.updateAnimations(self, 99999999)
581 if emptyState >= BaleLoader.EMPTY_ROTATE_PLATFORM then
582 self:doStateChange(BaleLoader.CHANGE_EMPTY_ROTATE_PLATFORM)
583 AnimatedVehicle.updateAnimations(self, 99999999)
584 if emptyState >= BaleLoader.EMPTY_ROTATE1 then
585 self:doStateChange(BaleLoader.CHANGE_EMPTY_ROTATE1)
586 AnimatedVehicle.updateAnimations(self, 99999999)
587 if emptyState >= BaleLoader.EMPTY_CLOSE_GRIPPERS then
588 self:doStateChange(BaleLoader.CHANGE_EMPTY_CLOSE_GRIPPERS)
589 AnimatedVehicle.updateAnimations(self, 99999999)
590 if emptyState >= BaleLoader.EMPTY_HIDE_PUSHER1 then
591 self:doStateChange(BaleLoader.CHANGE_EMPTY_HIDE_PUSHER1)
592 AnimatedVehicle.updateAnimations(self, 99999999)
593 if emptyState >= BaleLoader.EMPTY_HIDE_PUSHER2 then
594 self:doStateChange(BaleLoader.CHANGE_EMPTY_HIDE_PUSHER2)
595 AnimatedVehicle.updateAnimations(self, 99999999)
596 if emptyState >= BaleLoader.EMPTY_ROTATE2 then
```

```
597  self:doStateChange(BaleLoader.CHANGE_EMPTY_ROTATE2)
598  AnimatedVehicle.updateAnimations(self, 99999999)
599  if emptyState >= BaleLoader.EMPTY_WAIT_TO_DROP then
600  self:doStateChange(BaleLoader.CHANGE_EMPTY_WAIT_TO_DROP)
601  AnimatedVehicle.updateAnimations(self, 99999999)
602  if emptyState == BaleLoader.EMPTY_CANCEL or emptyState ==
     BaleLoader.EMPTY_WAIT_TO_REDO then
603  self:doStateChange(BaleLoader.CHANGE_EMPTY_CANCEL)
604  AnimatedVehicle.updateAnimations(self, 99999999)
605  if emptyState == BaleLoader.EMPTY_WAIT_TO_REDO then
606  self:doStateChange(BaleLoader.CHANGE_EMPTY_WAIT_TO_REDO)
607  AnimatedVehicle.updateAnimations(self, 99999999)
608  end
609  elseif emptyState == BaleLoader.EMPTY_WAIT_TO_SINK or emptyState
     == BaleLoader.EMPTY_SINK then
610  self:doStateChange(BaleLoader.CHANGE_DROP_BALES)
611  AnimatedVehicle.updateAnimations(self, 99999999)
612
613  if emptyState == BaleLoader.EMPTY_SINK then
614  self:doStateChange(BaleLoader.CHANGE_SINK)
615  AnimatedVehicle.updateAnimations(self, 99999999)
616  end
617  end
618  end
619
620  end
621  end
622  end
623  end
624  end
625  end
626  end
627  spec.emptyState = emptyState
628  end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
634  function BaleLoader:onWriteStream(streamId, connection)
635  local spec = self.spec_baleLoader
636
637  streamWriteBool(streamId, spec.isInWorkPosition)
638  streamWriteIntN(streamId, spec.frontBalePusherDirection, 3)
639  streamWriteIntN(streamId, spec.rotatePlatformDirection, 3)
640
641  streamWriteUIntN(streamId, spec.emptyState, 4)
642
643  streamWriteInt8(streamId, spec.currentBalePlace)
644
645  -- write bale at bale grabber
646  if streamWriteBool(streamId, spec.baleGrabber.currentBale ~= nil)
     then
647  NetworkUtil.writeNodeObjectId(streamId,
     spec.baleGrabber.currentBale)
648  end
649
650  -- write bales at start bale places
651  streamWriteUIntN(streamId, spec.startBalePlace.count, 3)
652  for i=1, spec.startBalePlace.count do
653  local baleServerId = spec.startBalePlace.bales[i]
654  NetworkUtil.writeNodeObjectId(streamId, baleServerId)
655  end
656
657  -- write bales at normal bale places
658  for i=1, table.getn(spec.balePlaces) do
659  local balePlace = spec.balePlaces[i]
660
661  local numBales = 0
662  if balePlace.bales ~= nil then
663  numBales = table.getn(balePlace.bales)
664  end
665  streamWriteUIntN(streamId, numBales, 3)
666  if balePlace.bales ~= nil then
667  for baleI=1, numBales do
668  local baleServerId = balePlace.bales[baleI]
669  local bale = NetworkUtil.getObject(baleServerId)
670  local nodeId = bale.nodeId
```

```
671 local x,y,z = getTranslation(nodeId)
672 NetworkUtil.writeNodeObjectId(streamId, baleServerId)
673 streamWriteFloat32(streamId, x)
674 streamWriteFloat32(streamId, y)
675 streamWriteFloat32(streamId, z)
676 end
677 end
678 end
679 end
```

updateBalePlacesAnimations

Description

Update bale place animations

Definition

updateBalePlacesAnimations()

Code

```
683 function BaleLoader.updateBalePlacesAnimations(self)
684 local spec = self.spec_baleLoader
685
686 if spec.currentBalePlace > spec.startBalePlace.numOfPlaces or
    (spec.moveBalePlacesAfterRotatePlatform and spec.currentBalePlace
    > 1) then
687 local delta = 1
688 if spec.moveBalePlacesAfterRotatePlatform and not
    spec.alwaysMoveBalePlaces then
689 delta = 0
690 end
691 -- currentBalePlace-1 or -0 needs to be at the first position
692 self:playAnimation("moveBalePlaces", 1, 0, true)
693 self:setAnimationStopTime("moveBalePlaces",
    (spec.currentBalePlace-delta)/table.getn(spec.balePlaces))
694
695 AnimatedVehicle.updateAnimations(self, 99999999)
696 end
697
698 if spec.startBalePlace.count >= 1 then
699 self:playAnimation("balesToOtherRow", 20, nil, true)
700 AnimatedVehicle.updateAnimations(self, 99999999)
701 if spec.startBalePlace.count >= spec.startBalePlace.numOfPlaces
    then
702 BaleLoader.rotatePlatform(self)
703 end
704 end
```

```
705 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
712 function BaleLoader:onUpdate(dt, isActiveForInput, isSelected)
713 local spec = self.spec_baleLoader
714
715 if self.firstTimeRun then
716 for k,v in pairs(spec.balesToLoad) do
717 local baleObject = Bale:new(self.isServer, self.isClient)
718 local x,y,z = unpack(v.translation)
719 local rx,ry,rz = unpack(v.rotation)
720 baleObject:load(v.filename, x,y,z,rx,ry,rz, v.fillLevel)
721 baleObject.ownerFarmId = Utils.getNoNil(v.farmId, AccessHandler.EVERYONE
722
723 if spec.mountDynamic then
724 self:mountDynamicBale(baleObject, v.parentNode)
725 else
726 baleObject:mount(self, v.parentNode, x,y,z, rx,ry,rz)
727 end
728
729 baleObject:applyExtraAttributes(v.attributes)
730 baleObject:register()
731
732 table.insert(v.bales, NetworkUtil.getObjectId(baleObject))
733 spec.balesToLoad[k] = nil
734 end
735
736 for k, baleToMount in pairs(spec.balesToMount) do
737 local bale = NetworkUtil.getObject(baleToMount.serverId)
738 if bale ~= nil then
739 local x,y,z = unpack(baleToMount.trans)
740 local rx,ry,rz = unpack(baleToMount.rot)
741
```

```
742 if spec.mountDynamic then
743 self:mountDynamicBale(bale, baleToMount.linkNode)
744 else
745 bale:mount(self, baleToMount.linkNode, x,y,z, rx,ry,rz)
746 end
747
748 spec.balesToMount[k] = nil
749 end
750 end
751 end
752
753 if self.isClient then
754 if spec.grabParticleSystem ~= nil then
755 if spec.grabParticleSystemDisableTime ~= 0 and
    spec.grabParticleSystemDisableTime < g_currentMission.time then
756 ParticleUtil.setEmittingState(spec.grabParticleSystem, false)
757 spec.grabParticleSystemDisableTime = 0
758 end
759 end
760 end
761
762 -- check if grabber is still moving
763 if spec.grabberIsMoving then
764 if not self:getIsAnimationPlaying("baleGrabberTransportToWork") then
765 spec.grabberIsMoving = false
766 end
767 end
768
769 if self:getIsBaleGrabbingAllowed() then
770 if spec.baleGrabber.grabNode ~= nil and spec.baleGrabber.currentBale ==
771 -- find nearest bale
772 local nearestBale, nearestBaleType = BaleLoader.getBaleInRange(self,
    spec.baleGrabber.grabNode, spec.baleGrabber.balesInTrigger)
773 if nearestBale ~= nil then
774 if nearestBaleType == nil then
775 g_currentMission:showBlinkingWarning(g_i18n:getText("warning_baleNotSupp
    2000)
776 elseif self.isServer then
777 self:pickupBale(nearestBale, nearestBaleType)
778 end
779 end
```

```
780 end
781 end
782 if self.isServer then
783 if spec.grabberMoveState ~= nil then
784 if spec.grabberMoveState == BaleLoader.GRAB_MOVE_UP then
785 if not self:getIsAnimationPlaying("baleGrabberWorkToDrop") then
786 --BaleLoader.CHANGE_GRAB_MOVE_UP
787 -- switch to grabberMoveState
788 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_GRAB_MOVE_UP), true, nil, self)
789 end
790 elseif spec.grabberMoveState == BaleLoader.GRAB_DROP_BALE then
791 if not self:getIsAnimationPlaying(spec.currentBaleGrabberDropBaleAnimNam
792 --BaleLoader.CHANGE_GRAB_DROP_BALE
793 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_GRAB_DROP_BALE), true, nil, self)
794 end
795 elseif spec.grabberMoveState == BaleLoader.GRAB_MOVE_DOWN then
796 --BaleLoader.CHANGE_GRAB_MOVE_DOWN
797 if not self:getIsAnimationPlaying("baleGrabberWorkToDrop") then
798 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_GRAB_MOVE_DOWN), true, nil, self)
799 end
800 end
801 end
802 if spec.frontBalePusherDirection ~= 0 then
803 if not self:getIsAnimationPlaying("frontBalePusher") then
804 --BaleLoader.CHANGE_FRONT_PUSHER
805 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_FRONT_PUSHER), true, nil, self)
806 end
807 end
808 if spec.rotatePlatformDirection ~= 0 then
809 local name = spec.animations.rotatePlatform
810 if spec.rotatePlatformDirection < 0 then
811 name = spec.animations.rotatePlatformBack
812 end
813 if not self:getIsAnimationPlaying(name) and not
    self:getIsAnimationPlaying("moveBalePlacesExtrasOnce") then
814 --BaleLoader.CHANGE_ROTATE_PLATFORM
815 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_ROTATE_PLATFORM), true, nil, self)
```

```
816 end
817 end
818
819 if spec.emptyState ~= BaleLoader.EMPTY_NONE then
820 if spec.emptyState == BaleLoader.EMPTY_TO_WORK then
821 if not self:getIsAnimationPlaying("baleGrabberTransportToWork") then
822 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_EMPTY_ROTATE_PLATFORM), true, nil, self)
823 end
824 elseif spec.emptyState == BaleLoader.EMPTY_ROTATE_PLATFORM then
825 if not self:getIsAnimationPlaying(spec.animations.rotatePlatformEmpty) t
826 --BaleLoader.CHANGE_EMPTY_ROTATE1
827 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_EMPTY_ROTATE1), true, nil, self)
828 end
829 elseif spec.emptyState == BaleLoader.EMPTY_ROTATE1 then
830 if not self:getIsAnimationPlaying("emptyRotate") and not
    self:getIsAnimationPlaying("moveBalePlacesToEmpty") then
831 --BaleLoader.CHANGE_EMPTY_CLOSE_GRIPPERS
832 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_EMPTY_CLOSE_GRIPPERS), true, nil, self)
833 end
834 elseif spec.emptyState == BaleLoader.EMPTY_CLOSE_GRIPPERS then
835 if not self:getIsAnimationPlaying("closeGrippers") then
836 --BaleLoader.CHANGE_EMPTY_HIDE_PUSHER1
837 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_EMPTY_HIDE_PUSHER1), true, nil, self)
838 end
839 elseif spec.emptyState == BaleLoader.EMPTY_HIDE_PUSHER1 then
840 if not self:getIsAnimationPlaying("emptyHidePusher1") then
841 --BaleLoader.CHANGE_EMPTY_HIDE_PUSHER2
842 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_EMPTY_HIDE_PUSHER2), true, nil, self)
843 end
844 elseif spec.emptyState == BaleLoader.EMPTY_HIDE_PUSHER2 then
845 if self:getAnimationTime("moveBalePusherToEmpty") < 0.7 or not
    self:getIsAnimationPlaying("moveBalePusherToEmpty") then
846 --BaleLoader.CHANGE_EMPTY_ROTATE2
847 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_EMPTY_ROTATE2), true, nil, self)
848 end
849 elseif spec.emptyState == BaleLoader.EMPTY_ROTATE2 then
```

```
850 if not self:getIsAnimationPlaying("emptyRotate") then
851 --BaleLoader.CHANGE_EMPTY_WAIT_TO_DROP
852 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_EMPTY_WAIT_TO_DROP), true, nil, self)
853 end
854 elseif spec.emptyState == BaleLoader.EMPTY_SINK then
855 if not self:getIsAnimationPlaying("emptyRotate") and
856 not self:getIsAnimationPlaying("moveBalePlacesToEmpty") and
857 not self:getIsAnimationPlaying("emptyHidePusher1") and
858 not self:getIsAnimationPlaying(spec.animations.rotatePlatformEmpty)
859 then
860 --BaleLoader.CHANGE_EMPTY_STATE_NIL
861 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_EMPTY_STATE_NIL), true, nil, self)
862 end
863 elseif spec.emptyState == BaleLoader.EMPTY_CANCEL then
864 if not self:getIsAnimationPlaying("emptyRotate") then
865 --BaleLoader.CHANGE_EMPTY_WAIT_TO_REDO
866 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_EMPTY_WAIT_TO_REDO), true, nil, self)
867 end
868 end
869 end
870 end
871
872 -- re mount bale in grabber, which was unmounted to be saved as normal b
873 if spec.baleGrabber.currentBaleIsUnmounted then
874 spec.baleGrabber.currentBaleIsUnmounted = false
875 local bale = NetworkUtil.getObject(spec.baleGrabber.currentBale)
876 if bale ~= nil then
877 if spec.mountDynamic then
878 self:mountDynamicBale(bale, spec.baleGrabber.grabNode)
879 else
880 bale:mount(self, spec.baleGrabber.grabNode, 0,0,0, 0,0,0)
881 end
882 end
883 end
884 end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
891 function BaleLoader:onUpdateTick(dt, isActiveForInput, isSelected)
892 local spec = self.spec_baleLoader
893
894 if self.isClient then
895 local actionEvent = spec.actionEvents[InputAction.IMPLEMENT_EXTRA]
896 if actionEvent ~= nil then
897 local showAction = false
898 if spec.emptyState == BaleLoader.EMPTY_NONE then
899 if spec.grabberMoveState == nil then
900 if spec.isInWorkPosition then
901 g_inputBinding:setActionEventText(actionEvent.actionEventId, g_i18n:getText(spec.texts.transportPosition))
902 showAction = true
903 else
904 g_inputBinding:setActionEventText(actionEvent.actionEventId, g_i18n:getText(spec.texts.operatingPosition))
905 showAction = true
906 end
907 end
908 end
909
910 g_inputBinding:setActionEventActive(actionEvent.actionEventId, showAction)
911 end
912
913 actionEvent = spec.actionEvents[InputAction.IMPLEMENT_EXTRA2]
914 if actionEvent ~= nil then
915 g_inputBinding:setActionEventActive(actionEvent.actionEventId, spec.emptyState == BaleLoader.EMPTY_WAIT_TO_DROP)
916 end
917
918 actionEvent = spec.actionEvents[InputAction.IMPLEMENT_EXTRA3]
919 if actionEvent ~= nil then
```

```
920 local showAction = false
921
922 if spec.emptyState == BaleLoader.EMPTY_NONE then
923 if BaleLoader.getAllowsStartUnloading(self) then
924 g_inputBinding:setActionEventText(actionEvent.actionEventId,
    g_i18n:getText(spec.texts.unload))
925 showAction = true
926 end
927 elseif spec.emptyState == BaleLoader.EMPTY_WAIT_TO_DROP then
928 g_inputBinding:setActionEventText(actionEvent.actionEventId,
    g_i18n:getText(spec.texts.unloadHere))
929 showAction = true
930 elseif spec.emptyState == BaleLoader.EMPTY_WAIT_TO_SINK then
931 g_inputBinding:setActionEventText(actionEvent.actionEventId,
    g_i18n:getText(spec.texts.lowerPlattform))
932 showAction = true
933 elseif spec.emptyState == BaleLoader.EMPTY_WAIT_TO_REDO then
934 g_inputBinding:setActionEventText(actionEvent.actionEventId,
    g_i18n:getText(spec.texts.unload))
935 showAction = true
936 end
937
938 g_inputBinding:setActionEventActive(actionEvent.actionEventId,
    showAction)
939 end
940 end
941
942 if self.isServer then
943 if spec.automaticUnloading then
944 if spec.emptyState == BaleLoader.EMPTY_WAIT_TO_DROP then
945 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_BUTTON_EMPTY), true, nil, self)
946 end
947
948 if spec.emptyState == BaleLoader.EMPTY_WAIT_TO_SINK then
949 g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
    BaleLoader.CHANGE_SINK), true, nil, self)
950 end
951 end
952
953 local jointNodePositionChanged = false
954 -- move bale smoothly to the joint position
```

```
for i, jointNode in ipairs(spec.updateBaleJointNodePosition) do
local x, y, z = getTranslation(jointNode.node)

if jointNode.quaternion == nil then
local qx, qy, qz, qw = getQuaternion(jointNode.node)
jointNode.quaternion = {qx, qy, qz, qw}
end

jointNode.time = jointNode.time + dt
if jointNode.time < 1000 then
local qx, qy, qz, qw = 0, 0, 0, 1
if math.abs(jointNode.quaternion[2]) > 0.5 then
qx, qy, qz, qw = MathUtil.slerpQuaternionShortestPath(jointNode.quaternion[1], jointNode.quaternion[2], jointNode.quaternion[3], jointNode.quaternion[4], 0, 1, 0, 0, jointNode.time/1000)
elseif math.abs(jointNode.quaternion[2]) < 0.5 then
qx, qy, qz, qw = MathUtil.slerpQuaternionShortestPath(jointNode.quaternion[1], jointNode.quaternion[2], jointNode.quaternion[3], jointNode.quaternion[4], 0, 0, 0, 1, jointNode.time/1000)
end
setQuaternion(jointNode.node, qx, qy, qz, qw)
jointNodePositionChanged = true
end

if math.abs(x)+math.abs(y)+math.abs(z) > 0.001 then
local move = 0.0001*dt

local function moveValue(old, move)
local limit = MathUtil.sign(old) > 0 and math.max or math.min
return limit(old-MathUtil.sign(old)*move, 0)
end

setTranslation(jointNode.node, moveValue(x, move), moveValue(y, move), moveValue(z, move))

jointNodePositionChanged = true
elseif jointNode.time > 1000 then
table.remove(spec.updateBaleJointNodePosition, i)
end
end
```

```
990
991  -- update dynamic bale joints as soon one of the available
     animations is playing
992  local anyAnimationPlaying = false
993  for name,_ in pairs(self.spec_animatedVehicle.animations) do
994  if self:getIsAnimationPlaying(name) then
995  anyAnimationPlaying = true
996  end
997  end
998
999  if anyAnimationPlaying or jointNodePositionChanged then
1000 for _, balePlace in pairs(spec.balePlaces) do
1001 if balePlace.bales ~= nil then
1002 for _, baleServerId in pairs(balePlace.bales) do
1003 local bale = NetworkUtil.getObject(baleServerId)
1004 if bale ~= nil then
1005 if bale.dynamicMountJointIndex ~= nil then
1006 setJointFrame(bale.dynamicMountJointIndex, 0,
     bale.dynamicMountJointNode)
1007 end
1008 end
1009 end
1010 end
1011 end
1012 for _, baleServerId in ipairs(spec.startBalePlace.bales) do
1013 local bale = NetworkUtil.getObject(baleServerId)
1014 if bale ~= nil then
1015 if bale.dynamicMountJointIndex ~= nil then
1016 setJointFrame(bale.dynamicMountJointIndex, 0,
     bale.dynamicMountJointNode)
1017 end
1018 end
1019 end
1020 end
1021 end
1022 end
```

getBaleInRange

Description

Returns if nearest bale in range

Definition

getBaleInRange(integer refNode)

Arguments

integer refNode id of reference node

Return Values

table bale bale

integer nearestBaleType id of bale type

Code

```
1029  function BaleLoader.getBaleInRange(self, refNode, balesInTrigger)
1030  local spec = self.spec_baleLoader
1031
1032  local nearestDistance = spec.baleGrabber.pickupRange
1033  local nearestBale = nil
1034  local nearestBaleType = nil
1035
1036  for bale, state in pairs(balesInTrigger) do
1037  if state ~= nil and state > 0 then
1038
1039  local isValidBale = true
1040  for _, balePlace in pairs(spec.balePlaces) do
1041  if balePlace.bales ~= nil then
1042  for _, baleServerId in pairs(balePlace.bales) do
1043  local baleInPlace = NetworkUtil.getObject(baleServerId)
1044  if baleInPlace ~= nil and baleInPlace == bale then
1045  isValidBale = false
1046  end
1047  end
1048  end
1049  end
1050  for _, baleServerId in ipairs(spec.startBalePlace.bales) do
1051  local baleInPlace = NetworkUtil.getObject(baleServerId)
1052  if baleInPlace ~= nil and baleInPlace == bale then
1053  isValidBale = false
1054  end
1055  end
1056
1057  if isValidBale then
1058
1059  local distance = calcDistanceFrom(refNode, bale.nodeId)
1060  if distance < nearestDistance then
1061  local foundBaleType
1062  for _, baleType in pairs(spec.allowedBaleTypes) do
```

```
1063    if baleType.minBaleDiameter ~= nil then
1064    if bale.baleDiameter ~= nil and bale.baleWidth ~= nil and
1065    bale.baleDiameter >= baleType.minBaleDiameter and
        bale.baleDiameter <= baleType.maxBaleDiameter and
1066    bale.baleWidth >= baleType.minBaleWidth and bale.baleWidth <=
        baleType.maxBaleWidth
1067    then
1068    foundBaleType = baleType
1069    break
1070    end
1071    else
1072    if bale.baleWidth ~= nil and bale.baleHeight ~= nil and
        bale.baleLength ~= nil and
1073    bale.baleWidth >= baleType.minBaleWidth and bale.baleWidth <=
        baleType.maxBaleWidth and
1074    bale.baleHeight >= baleType.minBaleHeight and bale.baleHeight <=
        baleType.maxBaleHeight and
1075    bale.baleLength >= baleType.minBaleLength and bale.baleLength <=
        baleType.maxBaleLength
1076    then
1077    foundBaleType = baleType
1078    break
1079    end
1080    end
1081    end
1082    if foundBaleType ~= nil or nearestBaleType == nil then
1083    if foundBaleType ~= nil then
1084    nearestDistance = distance
1085    end
1086    nearestBale = bale
1087    nearestBaleType = foundBaleType
1088    end
1089    end
1090
1091    end
1092    end
1093    end
1094    return nearestBale, nearestBaleType
1095    end
```

onDeactivate

Description

Called on deactivate

Definition

onDeactivate()

Code

```
1099  function BaleLoader:onDeactivate()
1100  local spec = self.spec_baleLoader
1101  if spec.grabParticleSystem ~= nil then
1102  ParticleUtil.setEmittingState(spec.grabParticleSystem, false)
1103  end
1104  end
```

doStateChange

Description

Change bale loader state

Definition

doStateChange(integer id, integer nearestBaleServerId)

Arguments

integer id	id of new state
integer nearestBaleServerId	server id of nearest bale

Code

```
1110  function BaleLoader:doStateChange(id, nearestBaleServerId)
1111  local spec = self.spec_baleLoader
1112
1113  if id == BaleLoader.CHANGE_DROP_BALES then
1114  -- drop all bales to ground (and add to save by mission)
1115  spec.currentBalePlace = 1
1116  for _, balePlace in pairs(spec.balePlaces) do
1117  if balePlace.bales ~= nil then
1118  for _, baleServerId in pairs(balePlace.bales) do
1119  local bale = NetworkUtil.getObject(baleServerId)
1120  if bale ~= nil then
1121  if spec.mountDynamic then
1122  self:unmountDynamicBale(bale)
1123  else
1124  bale:unmount()
1125  end
1126
1127  -- clear bales in trigger table
1128  if spec.baleGrabber.balesInTrigger[bale] ~= nil then
1129  spec.baleGrabber.balesInTrigger[bale] = nil
1130  end
1131  end
1132  spec.balesToMount[baleServerId] = nil
```

```
1133  end
1134  balePlace.bales = nil
1135  end
1136  end
1137
1138  self:addFillUnitFillLevel(self:getOwnerFarmId(),
      spec.fillUnitIndex, -math.huge,
      self:getFillUnitFirstSupportedFillType(spec.fillUnitIndex),
      ToolType.UNDEFINED, nil)
1139  for _, place in pairs(spec.balePlaces) do
1140  if place.collision ~= nil then
1141  setIsCompoundChild(place.collision, false)
1142  end
1143  end
1144
1145  self:playAnimation("releaseFrontplattform", 1, nil, true)
1146  self:playAnimation("closeGrippers", -1, nil, true)
1147  spec.emptyState = BaleLoader.EMPTY_WAIT_TO_SINK
1148  elseif id == BaleLoader.CHANGE_SINK then
1149  if spec.resetEmptyRotateAnimation then
1150  self:playAnimation("emptyRotate", -1, nil, true)
1151  end
1152  self:playAnimation("moveBalePlacesToEmpty", -1, nil, true)
1153  self:playAnimation("emptyHidePusher1", -1, nil, true)
1154  self:playAnimation(spec.animations.rotatePlatformEmpty, -1, nil,
      true)
1155  if not spec.isInWorkPosition then
1156  self:playAnimation("closeGrippers", 1,
      self:getAnimationTime("closeGrippers"), true)
1157  end
1158  spec.emptyState = BaleLoader.EMPTY_SINK
1159  elseif id == BaleLoader.CHANGE_EMPTY_REDO then
1160  self:playAnimation("emptyRotate", 1, nil, true)
1161  spec.emptyState = BaleLoader.EMPTY_ROTATE2
1162  elseif id == BaleLoader.CHANGE_EMPTY_START then
1163  -- move to work position in case it is not there now
1164  BaleLoader.moveToWorkPosition(self)
1165  spec.emptyState = BaleLoader.EMPTY_TO_WORK
1166  elseif id == BaleLoader.CHANGE_EMPTY_CANCEL then
1167  self:playAnimation("emptyRotate", -1, nil, true)
1168  spec.emptyState = BaleLoader.EMPTY_CANCEL
```

```
1169  elseif id == BaleLoader.CHANGE_MOVE_TO_TRANSPORT then
1170  if spec.isInWorkPosition then
1171  spec.grabberIsMoving = true
1172  spec.isInWorkPosition = false
1173  g_animationManager:stopAnimations(spec.animationNodes)
1174  -- move to transport position
1175  BaleLoader.moveToTransportPosition(self)
1176  end
1177  elseif id == BaleLoader.CHANGE_MOVE_TO_WORK then
1178  if not spec.isInWorkPosition then
1179  spec.grabberIsMoving = true
1180  spec.isInWorkPosition = true
1181  g_animationManager:startAnimations(spec.animationNodes)
1182  -- move to work position
1183  BaleLoader.moveToWorkPosition(self)
1184  end
1185  elseif id == BaleLoader.CHANGE_GRAB_BALE then
1186  local bale = NetworkUtil.getObject(nearestBaleServerId)
1187  spec.baleGrabber.currentBale = nearestBaleServerId
1188  if bale ~= nil then
1189  if spec.mountDynamic then
1190  self:mountDynamicBale(bale, spec.baleGrabber.grabNode)
1191  else
1192  bale:mount(self, spec.baleGrabber.grabNode, 0,0,0, 0,0,0)
1193  end
1194
1195  spec.balesToMount[nearestBaleServerId] = nil
1196  else
1197  spec.balesToMount[nearestBaleServerId] =
      {serverId=nearestBaleServerId, linkNode=spec.baleGrabber.grabNode,
      trans={0,0,0}, rot={0,0,0} }
1198  end
1199  spec.grabberMoveState = BaleLoader.GRAB_MOVE_UP
1200  self:playAnimation("baleGrabberWorkToDrop", 1, nil, true)
1201
1202  self:addFillUnitFillLevel(self:getOwnerFarmId(),
      spec.fillUnitIndex, 1,
      self:getFillUnitFirstSupportedFillType(spec.fillUnitIndex),
      ToolType.UNDEFINED, nil)
1203  for i, place in pairs(spec.balePlaces) do
1204  if place.collision ~= nil then
```

```
1205  if i <= self:getFillUnitFillLevel(spec.fillUnitIndex) then
1206  setIsCompoundChild(place.collision, true)
1207  else
1208  setIsCompoundChild(place.collision, false)
1209  end
1210  end
1211  end
1212
1213  if self.isClient then
1214  g_soundManager:playSample(spec.samples.grab)
1215
1216  if spec.grabParticleSystem ~= nil then
1217  ParticleUtil.setEmittingState(spec.grabParticleSystem, true)
1218  spec.grabParticleSystemDisableTime = g_currentMission.time +
      spec.grabParticleSystemDisableDuration
1219  end
1220  end
1221  elseif id == BaleLoader.CHANGE_GRAB_MOVE_UP then
1222  spec.currentBaleGrabberDropBaleAnimName =
      self:getBaleGrabberDropBaleAnimName()
1223  self:playAnimation(spec.currentBaleGrabberDropBaleAnimName, 1,
      nil, true)
1224  spec.grabberMoveState = BaleLoader.GRAB_DROP_BALE
1225  elseif id == BaleLoader.CHANGE_GRAB_DROP_BALE then
1226  -- drop bale at platform
1227  if spec.startBalePlace.count < spec.startBalePlace.numOfPlaces and
      spec.startBalePlace.node ~= nil then
1228  local attachNode = getChildAt(spec.startBalePlace.node,
      spec.startBalePlace.count)
1229  local bale = NetworkUtil.getObject(spec.baleGrabber.currentBale)
1230  if bale ~= nil then
1231  if spec.mountDynamic then
1232  self:mountDynamicBale(bale, attachNode)
1233  else
1234  bale:mount(self, attachNode, 0,0,0, 0,0,0)
1235  end
1236
1237  spec.balesToMount[spec.baleGrabber.currentBale] = nil
1238  else
1239  spec.balesToMount[spec.baleGrabber.currentBale] =
      {serverId=spec.baleGrabber.currentBale, linkNode=attachNode,
      trans={0,0,0}, rot={0,0,0} }
```

```
1240  end
1241
1242  spec.startBalePlace.count = spec.startBalePlace.count + 1
1243  table.insert(spec.startBalePlace.bales,
      spec.baleGrabber.currentBale)
1244  spec.baleGrabber.currentBale = nil
1245  --setRotation(baleNode, 0, 0, 0)
1246  --setTranslation(baleNode, 0, 0, 0)
1247
1248  if spec.startBalePlace.count < spec.startBalePlace.numOfPlaces
      then
1249  spec.frontBalePusherDirection = 1
1250  self:playAnimation("balesToOtherRow", 1, nil, true)
1251  self:playAnimation("frontBalePusher", 1, nil, true)
1252  elseif spec.startBalePlace.count ==
      spec.startBalePlace.numOfPlaces then
1253  BaleLoader.rotatePlatform(self)
1254  end
1255
1256  self:playAnimation(spec.currentBaleGrabberDropBaleAnimName, -5,
      nil, true)
1257  self:playAnimation("baleGrabberWorkToDrop", -1, nil, true)
1258  spec.grabberMoveState = BaleLoader.GRAB_MOVE_DOWN
1259  end
1260  elseif id == BaleLoader.CHANGE_GRAB_MOVE_DOWN then
1261  spec.grabberMoveState = nil
1262  elseif id == BaleLoader.CHANGE_FRONT_PUSHER then
1263  if spec.frontBalePusherDirection > 0 then
1264  self:playAnimation("frontBalePusher", -1, nil, true)
1265  spec.frontBalePusherDirection = -1
1266  else
1267  spec.frontBalePusherDirection = 0
1268  end
1269  elseif id == BaleLoader.CHANGE_ROTATE_PLATFORM then
1270  if spec.rotatePlatformDirection > 0 then
1271  -- drop bales
1272  local balePlace = spec.balePlaces[spec.currentBalePlace]
1273  spec.currentBalePlace = spec.currentBalePlace + 1
1274  for i=1, table.getn(spec.startBalePlace.bales) do
1275  local node = getChildAt(spec.startBalePlace.node, i-1)
1276  local x,y,z = getTranslation(node)
```

```
1277  local rx,ry,rz = getRotation(node)
1278  local baleServerId = spec.startBalePlace.bales[i]
1279  local bale = NetworkUtil.getObject(baleServerId)
1280  if bale ~= nil then
1281  if spec.mountDynamic then
1282  self:mountDynamicBale(bale, balePlace.node)
1283  else
1284  bale:mount(spec, balePlace.node, x,y,z, rx,ry,rz)
1285  end
1286
1287  spec.balesToMount[baleServerId] = nil
1288  else
1289  spec.balesToMount[baleServerId] = {serverId=baleServerId,
      linkNode=balePlace.node, trans={ x,y,z}, rot={rx,ry,rz} }
1290  end
1291  end
1292  balePlace.bales = spec.startBalePlace.bales
1293  spec.startBalePlace.bales = {}
1294  spec.startBalePlace.count = 0
1295
1296  for i=1, spec.startBalePlace.numOfPlaces do
1297  local node = getChildAt(spec.startBalePlace.node, i-1)
1298  setRotation(node, unpack(spec.startBalePlace.origRot[i]))
1299  setTranslation(node, unpack(spec.startBalePlace.origTrans[i]))
1300  end
1301
1302  if spec.emptyState == BaleLoader.EMPTY_NONE then
1303  -- we are not waiting to start emptying, rotate back
1304  spec.rotatePlatformDirection = -1
1305  self:playAnimation(spec.animations.rotatePlatformBack, -1, nil,
      true)
1306
1307  if spec.moveBalePlacesAfterRotatePlatform then
1308  -- currentBalePlace+1 needs to be at the first position
1309  if spec.currentBalePlace <= table.getn(spec.balePlaces) or
      spec.alwaysMoveBalePlaces then
1310  self:playAnimation("moveBalePlaces", 1, (spec.currentBalePlace-
      1)/table.getn(spec.balePlaces), true)
1311  self:setAnimationStopTime("moveBalePlaces",
      (spec.currentBalePlace)/table.getn(spec.balePlaces))
1312  self:playAnimation("moveBalePlacesExtrasOnce", 1, nil, true)
```

```
1313  end
1314  end
1315  else
1316  spec.rotatePlatformDirection = 0
1317  end
1318  else
1319  spec.rotatePlatformDirection = 0
1320  end
1321  elseif id == BaleLoader.CHANGE_EMPTY_ROTATE_PLATFORM then
1322  spec.emptyState = BaleLoader.EMPTY_ROTATE_PLATFORM
1323  if spec.startBalePlace.count == 0 then
1324  self:playAnimation(spec.animations.rotatePlatformEmpty, 1, nil,
      true)
1325  else
1326  BaleLoader.rotatePlatform(self)
1327  end
1328  elseif id == BaleLoader.CHANGE_EMPTY_ROTATE1 then
1329  self:playAnimation("emptyRotate", 1, nil, true)
1330  self:setAnimationStopTime("emptyRotate", 0.2)
1331  local balePlacesTime = self:getRealAnimationTime("moveBalePlaces")
1332  self:playAnimation("moveBalePlacesToEmpty", 1.5,
      balePlacesTime/self:getAnimationDuration("moveBalePlacesToEmpty"),
      true)
1333  self:playAnimation("moveBalePusherToEmpty", 1.5,
      balePlacesTime/self:getAnimationDuration("moveBalePusherToEmpty"),
      true)
1334  spec.emptyState = BaleLoader.EMPTY_ROTATE1
1335  elseif id == BaleLoader.CHANGE_EMPTY_CLOSE_GRIPPERS then
1336  self:playAnimation("closeGrippers", 1, nil, true)
1337  spec.emptyState = BaleLoader.EMPTY_CLOSE_GRIPPERS
1338  elseif id == BaleLoader.CHANGE_EMPTY_HIDE_PUSHER1 then
1339  self:playAnimation("emptyHidePusher1", 1, nil, true)
1340  spec.emptyState = BaleLoader.EMPTY_HIDE_PUSHER1
1341  elseif id == BaleLoader.CHANGE_EMPTY_HIDE_PUSHER2 then
1342  self:playAnimation("moveBalePusherToEmpty", -2, nil, true)
1343  spec.emptyState = BaleLoader.EMPTY_HIDE_PUSHER2
1344  elseif id == BaleLoader.CHANGE_EMPTY_ROTATE2 then
1345  self:playAnimation("emptyRotate", 1,
      self:getAnimationTime("emptyRotate"), true)
1346  spec.emptyState = BaleLoader.EMPTY_ROTATE2
1347  elseif id == BaleLoader.CHANGE_EMPTY_WAIT_TO_DROP then
1348  -- wait for the user to react (abort or drop)
```

```
1349  spec.emptyState = BaleLoader.EMPTY_WAIT_TO_DROP
1350  elseif id == BaleLoader.CHANGE_EMPTY_STATE_NIL then
1351  spec.emptyState = BaleLoader.EMPTY_NONE
1352  if spec.transportPositionAfterUnloading then
1353  BaleLoader.moveToTransportPosition(self)
1354  if self.isServer then
1355  g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
      BaleLoader.CHANGE_MOVE_TO_TRANSPORT), true, nil, self)
1356  end
1357  end
1358  elseif id == BaleLoader.CHANGE_EMPTY_WAIT_TO_REDO then
1359  spec.emptyState = BaleLoader.EMPTY_WAIT_TO_REDO
1360  elseif id == BaleLoader.CHANGE_BUTTON_EMPTY then
1361  -- Server only code
1362  assert(self.isServer)
1363  if spec.emptyState ~= BaleLoader.EMPTY_NONE then
1364  if spec.emptyState == BaleLoader.EMPTY_WAIT_TO_DROP then
1365  -- BaleLoader.CHANGE_DROP_BALES
1366  g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
      BaleLoader.CHANGE_DROP_BALES), true, nil, self)
1367  elseif spec.emptyState == BaleLoader.EMPTY_WAIT_TO_SINK then
1368  -- BaleLoader.CHANGE_SINK
1369  g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
      BaleLoader.CHANGE_SINK), true, nil, self)
1370  elseif spec.emptyState == BaleLoader.EMPTY_WAIT_TO_REDO then
1371  -- BaleLoader.CHANGE_EMPTY_REDO
1372  g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
      BaleLoader.CHANGE_EMPTY_REDO), true, nil, self)
1373  end
1374  else
1375  --BaleLoader.CHANGE_EMPTY_START
1376  if BaleLoader.getAllowsStartUnloading(self) then
1377  g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
      BaleLoader.CHANGE_EMPTY_START), true, nil, self)
1378  end
1379  end
1380  elseif id == BaleLoader.CHANGE_BUTTON_EMPTY_ABORT then
1381  -- Server only code
1382  assert(self.isServer)
1383  if spec.emptyState ~= BaleLoader.EMPTY_NONE then
1384  if spec.emptyState == BaleLoader.EMPTY_WAIT_TO_DROP then
```

```
1385  --BaleLoader.CHANGE_EMPTY_CANCEL
1386  g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
      BaleLoader.CHANGE_EMPTY_CANCEL), true, nil, self)
1387  end
1388  end
1389  elseif id == BaleLoader.CHANGE_BUTTON_WORK_TRANSPORT then
1390  -- Server only code
1391  assert(self.isServer)
1392  if spec.emptyState == BaleLoader.EMPTY_NONE and
      spec.grabberMoveState == nil then
1393  if spec.isInWorkPosition then
1394  g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
      BaleLoader.CHANGE_MOVE_TO_TRANSPORT), true, nil, self)
1395  else
1396  g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
      BaleLoader.CHANGE_MOVE_TO_WORK), true, nil, self)
1397  end
1398  end
1399  end
1400  end
```

getAllowsStartUnloading

Description

Returns if allows to start unloading bales

Definition

getAllowsStartUnloading()

Return Values

boolean allowsUnloading allows unloading

Code

```
1405  function BaleLoader.getAllowsStartUnloading(self)
1406  local spec = self.spec_baleLoader
1407
1408  if self:getFillUnitFillLevel(spec.fillUnitIndex) == 0 then
1409  return false
1410  end
1411
1412  if spec.rotatePlatformDirection ~= 0 then
1413  return false
1414  end
1415
1416  if spec.frontBalePusherDirection ~= 0 then
1417  return false
1418  end
```

```
1419
1420  if spec.grabberIsMoving or spec.grabberMoveState ~= nil then
1421  return false
1422  end
1423
1424  if spec.emptyState ~= BaleLoader.EMPTY_NONE then
1425  return false
1426  end
1427
1428  return true
1429  end
```

rotatePlatform

Description

Rotate bale loader platform

Definition

rotatePlatform()

Code

```
1433  function BaleLoader.rotatePlatform(self)
1434  local spec = self.spec_baleLoader
1435
1436  spec.rotatePlatformDirection = 1
1437  self:playAnimation(spec.animations.rotatePlatform, 1, nil, true)
1438
1439  if (spec.currentBalePlace > 1 and not
      spec.moveBalePlacesAfterRotatePlatform) or
      spec.alwaysMoveBalePlaces then
1440  -- currentBalePlace needs to be at the first position
1441  self:playAnimation("moveBalePlaces", 1, (spec.currentBalePlace-
      1)/table.getn(spec.balePlaces), true)
1442  self:setAnimationStopTime("moveBalePlaces",
      (spec.currentBalePlace)/table.getn(spec.balePlaces))
1443  self:playAnimation("moveBalePlacesExtrasOnce", 1, nil, true)
1444  end
1445  end
```

moveToWorkPosition

Description

Move bale Loader to work position

Definition

moveToWorkPosition(boolean onLoad)

Arguments

boolean onLoad called on load

Code

```
1450  function BaleLoader.moveToWorkPosition(self, onLoad)
1451  local speed = 1
1452  if onLoad then
1453  speed = 9999
1454  end
1455
1456  self:playAnimation("baleGrabberTransportToWork", speed,
      MathUtil.clamp(self:getAnimationTime("baleGrabberTransportToWork"),
      0, 1), true)
1457  local animTime = nil
1458  if self:getAnimationTime("closeGrippers") ~= 0 then
1459  animTime = self:getAnimationTime("closeGrippers")
1460  end
1461  self:playAnimation("closeGrippers", -1, animTime, true)
1462  end
```

moveToTransportPosition

Description

Move to transport position

Definition

moveToTransportPosition()

Code

```
1466  function BaleLoader.moveToTransportPosition(self)
1467  self:playAnimation("baleGrabberTransportToWork", -1,
      MathUtil.clamp(self:getAnimationTime("baleGrabberTransportToWork"),
      0, 1), true)
1468  self:playAnimation("closeGrippers", 1,
      MathUtil.clamp(self:getAnimationTime("closeGrippers"), 0, 1), true)
1469  end
```

getBaleGrabberDropBaleAnimName

Description

Returns bale grabber drop bale animation name

Definition

getBaleGrabberDropBaleAnimName()

Return Values

string name name of bale grabber drop bale animation name

Code

```
1474  function BaleLoader:getBaleGrabberDropBaleAnimName()
1475  local spec = self.spec_baleLoader
1476  local name = string.format("baleGrabberDropBale%d",
      spec.startBalePlace.count)
1477  if self:getAnimationExists(name) then
1478  return name
```

```
1479  end
1480  return "baleGrabberDropBale"
1481  end
```

getIsBaleGrabbingAllowed

Description

Returns if bale grabbing is allowed

Definition

getIsBaleGrabbingAllowed()

Return Values

string name name of bale grabber drop bale animation name

Code

```
1486  function BaleLoader:getIsBaleGrabbingAllowed()
1487  local spec = self.spec_baleLoader
1488
1489  if not spec.isInWorkPosition then
1490  return false
1491  end
1492
1493  if spec.grabberIsMoving or spec.grabberMoveState ~= nil then
1494  return false
1495  end
1496
1497  if spec.startBalePlace.count >= spec.startBalePlace.numOfPlaces
      then
1498  return false
1499  end
1500
1501  if spec.frontBalePusherDirection ~= 0 then
1502  return false
1503  end
1504
1505  if spec.rotatePlatformDirection ~= 0 then
1506  return false
1507  end
1508
1509  if spec.alwaysMoveBalePlaces and
      self:getIsAnimationPlaying("moveBalePlaces") then
1510  return false
1511  end
1512
1513  if spec.emptyState ~= BaleLoader.EMPTY_NONE then
```

```
1514  return false
1515  end
1516
1517  if self:getFillUnitFreeCapacity(spec.fillUnitIndex) == 0 then
1518  return false
1519  end
1520
1521  return true
1522  end
```

pickupBale

Description

Pickup bale

Definition

pickupBale(table nearestBale, integer nearestBaleType)

Arguments

table nearestBale nearest bale

integer nearestBaleType nearest bale type

Code

```
1527  function BaleLoader:pickupBale(nearestBale, nearestBaleType)
1528  g_server:broadcastEvent(BaleLoaderStateEvent:new(self,
      BaleLoader.CHANGE_GRAB_BALE,
      NetworkUtil.getObjectId(nearestBale)), true, nil, self)
1529  end
```

BaleWrapper

Description

Class for all BaleWrappers

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
41  function BaleWrapper.prerequisitesPresent(specializations)
42  return SpecializationUtil.hasSpecialization(AnimatedVehicle,
    specializations)
43  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
83  function BaleWrapper:onLoad(savegame)
84  local spec = self.spec_baleWrapper
85
86  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.wrapper", "vehicle.baleWrapper") --
    FS17 to FS19
87  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.baleGrabber",
    "vehicle.baleWrapper.grabber") --FS17 to FS19
88  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.baleWrapper.grabber#index",
    "vehicle.baleWrapper.grabber#node") --FS17 to FS19
89  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.baleWrapper.grabber#index",
    "vehicle.baleWrapper.grabber#node") --FS17 to FS19
90  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.baleWrapper.roundBaleWrapper#baleIndex",
    "vehicle.baleWrapper.roundBaleWrapper#baleNode") --FS17 to FS19
91  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.baleWrapper.roundBaleWrapper#wrapperIndex",
    "vehicle.baleWrapper.roundBaleWrapper#wrapperNode") --FS17 to
    FS19
92  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.baleWrapper.squareBaleWrapper#baleIndex",
    "vehicle.baleWrapper.squareBaleWrapper#baleNode") --FS17 to FS19
93  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.baleWrapper.squareBaleWrapper#wrapperIndex",
    "vehicle.baleWrapper.squareBaleWrapper#wrapperNode") --FS17 to
    FS19
94
95  local baseKey = "vehicle.baleWrapper"
96
97  spec.roundBaleWrapper = {}
98  self:loadWrapperFromXML(spec.roundBaleWrapper, self.xmlFile,
    baseKey..".roundBaleWrapper")
99
100 spec.squareBaleWrapper = {}
```

```
101 self:loadWrapperFromXML(spec.squareBaleWrapper, self.xmlFile,
    baseKey..".squareBaleWrapper")
102
103
104 spec.currentWrapper = {}
105 spec.currentWrapperFoldMinLimit =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    baseKey.."#foldMinLimit"), 0)
106 spec.currentWrapperFoldMaxLimit =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    baseKey.."#foldMaxLimit"), 1)
107
108
109 spec.currentWrapper = spec.roundBaleWrapper
110 self:updateWrapNodes(false, true, 0)
111 spec.currentWrapper = spec.squareBaleWrapper
112 self:updateWrapNodes(false, true, 0)
113
114 spec.baleGrabber = {}
115 spec.baleGrabber.grabNode =
    I3DUtil.indexToObject(self.components, getXMLString(self.xmlFile,
    baseKey..".grabber#node"), self.i3dMappings)
116 spec.baleGrabber.nearestDistance =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    baseKey..".grabber#nearestDistance"), 3.0)
117
118 spec.baleToLoad = nil
119 spec.baleToMount = nil
120 spec.baleWrapperState = BaleWrapper.STATE_NONE
121 spec.grabberIsMoving = false
122 spec.hasBaleWrapper = true
123 spec.showInvalidBaleWarning = false
124 end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
357 function BaleWrapper:onDelete()
358 local spec = self.spec_baleWrapper
359
360 local baleId
```

```
361 if spec.currentWrapper.currentBale ~= nil then
362 baleId = spec.currentWrapper.currentBale
363 end
364 if spec.baleGrabber.currentBale ~= nil then
365 baleId = spec.baleGrabber.currentBale
366 end
367 if baleId ~= nil then
368 local bale = NetworkUtil.getObject(baleId)
369 if bale ~= nil then
370 -- unmount bale
371 bale:unmount()
372 end
373 end
374 if self.isClient then
375 g_soundManager:deleteSamples(spec.roundBaleWrapper.samples)
376 g_soundManager:deleteSamples(spec.squareBaleWrapper.samples)
377 end
378 end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
407 function BaleWrapper:onReadStream(streamId, connection)
408 if connection:getIsServer() then
409 local spec = self.spec_baleWrapper
410
411 local isRoundBaleWrapper = streamReadBool(streamId)
412 if isRoundBaleWrapper then
413 spec.currentWrapper = spec.roundBaleWrapper
414 else
415 spec.currentWrapper = spec.squareBaleWrapper
416 end
417
418 local wrapperState = streamReadUIntN(streamId,
    BaleWrapper.STATE_NUM_BITS)
419 if wrapperState >= BaleWrapper.STATE_MOVING_BALE_TO_WRAPPER then
```

```
420 local baleServerId
421 local isRoundBale
422 if wrapperState ~= BaleWrapper.STATE_WRAPPER_RESETTING_PLATFORM
    then
423 baleServerId = NetworkUtil.readNodeObjectId(streamId)
424 isRoundBale = streamReadBool(streamId)
425 end
426
427 if wrapperState == BaleWrapper.STATE_MOVING_BALE_TO_WRAPPER then
428 self:doStateChange(BaleWrapper.CHANGE_GRAB_BALE, baleServerId)
429 AnimatedVehicle.updateAnimations(self, 99999999)
430 elseif wrapperState == BaleWrapper.STATE_MOVING_GRABBER_TO_WORK
    then
431 self.baleGrabber.currentBale = baleServerId
432
433 self:doStateChange(BaleWrapper.CHANGE_DROP_BALE_AT_GRABBER)
434 AnimatedVehicle.updateAnimations(self, 99999999)
435
436 elseif wrapperState ~=
    BaleWrapper.STATE_WRAPPER_RESETTING_PLATFORM then
437 spec.currentWrapper = (isRoundBale and spec["roundBaleWrapper"])
    or spec["squareBaleWrapper"]
438
439 local attachNode = spec.currentWrapper.baleNode
440 spec.baleToMount = {serverId=baleServerId, linkNode=attachNode,
    trans={0,0,0}, rot={0,0,0} }
441 self:updateWrapNodes(true, false, 0)
442 spec.currentWrapper.currentBale = baleServerId
443
444 if wrapperState == BaleWrapper.STATE_WRAPPER_WRAPPING_BALE then
445 local wrapperTime = streamReadFloat32(streamId)
446 spec.currentWrapper.currentTime = wrapperTime
447 self:updateWrappingState(spec.currentWrapper.currentTime /
    spec.currentWrapper.animTime, true)
448 else
449 spec.currentWrapper.currentTime = spec.currentWrapper.animTime
450 self:updateWrappingState(1, true)
451
452 self:doStateChange(BaleWrapper.CHANGE_WRAPPING_BALE_FINSIHED)
453 AnimatedVehicle.updateAnimations(self, 99999999)
454 if wrapperState >= BaleWrapper.STATE_WRAPPER_DROPPING_BALE then
455 self:doStateChange(BaleWrapper.CHANGE_WRAPPER_START_DROP_BALE)
```

```
456 AnimatedVehicle.updateAnimations(self, 99999999)
457 end
458 end
459 else
460 -- simply set the state but do nothing else
461 spec.baleWrapperState =
    BaleWrapper.STATE_WRAPPER_RESETTING_PLATFORM
462 end
463 end
464 end
465 end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
471 function BaleWrapper:onWriteStream(streamId, connection)
472 if not connection:getIsServer() then
473 local spec = self.spec_baleWrapper
474
475 streamWriteBool(streamId, spec.currentWrapper ==
    spec.roundBaleWrapper)
476
477 local wrapperState = spec.baleWrapperState
478 streamWriteUIntN(streamId, wrapperState,
    BaleWrapper.STATE_NUM_BITS)
479
480 if wrapperState >= BaleWrapper.STATE_MOVING_BALE_TO_WRAPPER and
    wrapperState ~= BaleWrapper.STATE_WRAPPER_RESETTING_PLATFORM then
481 local bale
482 if wrapperState == BaleWrapper.STATE_MOVING_BALE_TO_WRAPPER then
483 NetworkUtil.writeNodeObjectId(streamId,
    spec.baleGrabber.currentBale)
484 bale = NetworkUtil.getObject(spec.baleGrabber.currentBale)
485 else
486 NetworkUtil.writeNodeObjectId(streamId,
    spec.currentWrapper.currentBale)
487 bale = NetworkUtil.getObject(spec.currentWrapper.currentBale)
```

```
488 end
489
490 streamWriteBool(streamId, (bale or {}).baleDiameter ~= nil)
491 end
492 if wrapperState == BaleWrapper.STATE_WRAPPER_WRAPPING_BALE then
493 streamWriteFloat32(streamId, spec.currentWrapper.currentTime)
494 end
495 end
496 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
503 function BaleWrapper:onUpdate(dt, isActiveForInput, isSelected)
504 local spec = self.spec_baleWrapper
505
506 if self.firstTimeRun then
507 if spec.baleToLoad ~= nil then
508 local v = spec.baleToLoad
509 spec.baleToLoad = nil
510 local baleObject = Bale:new(self.isServer, self.isClient)
511 local x,y,z = unpack(v.translation)
512 local rx,ry,rz = unpack(v.rotation)
513 baleObject:load(v.filename, x,y,z,rx,ry,rz, v.fillLevel)
514 baleObject:register()
515
516 if baleObject.nodeId ~= nil and baleObject.nodeId ~= 0 then
517 self:doStateChange(BaleWrapper.CHANGE_GRAB_BALE,
    NetworkUtil.getObjectId(baleObject))
518 self:doStateChange(BaleWrapper.CHANGE_DROP_BALE_AT_GRABBER)
519
520 baleObject.baleValueScale = v.baleValueScale
521 local wrapperState = math.min(v.wrapperTime /
    spec.currentWrapper.animTime, 1)
522 baleObject:setWrappingState(wrapperState)
```

```
523
524 self:doStateChange(BaleWrapper.CHANGE_WRAPPING_START)
525 spec.currentWrapper.currentTime = v.wrapperTime
526
527 local wrappingTime = spec.currentWrapper.currentTime /
    spec.currentWrapper.animTime
528 self:setAnimationTime(spec.currentWrapper.animations["wrapBale"].animNam
    wrappingTime)
529 self:updateWrappingState(wrappingTime)
530 end
531 end
532
533 if spec.baleToMount ~= nil then
534 local bale = NetworkUtil.getObject(spec.baleToMount.serverId)
535 if bale ~= nil then
536 local x,y,z = unpack(spec.baleToMount.trans)
537 local rx,ry,rz = unpack(spec.baleToMount.rot)
538 bale:mount(self, spec.baleToMount.linkNode, x,y,z, rx,ry,rz)
539 spec.baleToMount = nil
540
541 if spec.baleWrapperState == BaleWrapper.STATE_MOVING_BALE_TO_WRAPPER the
542 self:playMoveToWrapper(bale)
543 end
544 end
545 end
546 end
547
548 if spec.baleWrapperState == BaleWrapper.STATE_WRAPPER_WRAPPING_BALE then
549 local wrapper = spec.currentWrapper
550
551 wrapper.currentTime = wrapper.currentTime + dt
552 local wrappingTime = wrapper.currentTime / wrapper.animTime
553 self:updateWrappingState(wrappingTime)
554
555 if self.isClient then
556 if wrapper.wrappingSoundEndTime <= wrappingTime then
557 if g_soundManager:getIsSamplePlaying(wrapper.samples.wrap) then
558 g_soundManager:stopSample(wrapper.samples.wrap)
559 g_soundManager:playSample(wrapper.samples.stop)
560 end
561 else
```

```
562 if not g_soundManager:getIsSamplePlaying(wrapper.samples.wrap) then
563 g_soundManager:playSample(wrapper.samples.wrap)
564 end
565 end
566 end
567 end
568 end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
575 function BaleWrapper:onUpdateTick(dt, isActiveForInput, isSelected)
576 local spec = self.spec_baleWrapper
577
578 spec.showInvalidBaleWarning = false
579
580 if self:allowsGrabbingBale() then
581 if spec.baleGrabber.grabNode ~= nil and spec.baleGrabber.currentBale ==
582 -- find nearest bale
583 local nearestBale, nearestBaleType = BaleWrapper.getBaleInRange(self,
    spec.baleGrabber.grabNode, spec.baleGrabber.nearestDistance)
584 if nearestBale ~= nil then
585 if self.isServer and (nearestBaleType ~= nil or
    self:getIsBaleFillTypeSkiped(nearestBale)) then
586 self:pickupWrapperBale(nearestBale, nearestBaleType)
587 else
588 if spec.lastDroppedBale ~= nearestBale then
589 spec.showInvalidBaleWarning = true
590 end
591 end
592 end
593 end
594 end
595 if self.isServer then
596 if spec.baleWrapperState ~= BaleWrapper.STATE_NONE then
```

```
597 if spec.baleWrapperState == BaleWrapper.STATE_MOVING_BALE_TO_WRAPPER the
598 if not self:getIsAnimationPlaying(spec.currentWrapper.animations["moveToWrapper then
599 g_server:broadcastEvent(BaleWrapperStateEvent:new(self, BaleWrapper.CHANGE_DROP_BALE_AT_GRABBER), true, nil, self)
600 end
601 elseif spec.baleWrapperState == BaleWrapper.STATE_MOVING_GRABBER_TO_WORK
602 if not self:getIsAnimationPlaying(spec.currentWrapper.animations["moveToWrapper then
603 local bale = NetworkUtil.getObject(spec.currentWrapper.currentBale)
604 if bale ~= nil and not bale.supportsWrapping then
605 g_server:broadcastEvent(BaleWrapperStateEvent:new(self, BaleWrapper.CHANGE_WRAPPER_START_DROP_BALE), true, nil, self)
606 else
607 g_server:broadcastEvent(BaleWrapperStateEvent:new(self, BaleWrapper.CHANGE_WRAPPING_START), true, nil, self)
608 end
609 end
610 elseif spec.baleWrapperState == BaleWrapper.STATE_WRAPPER_DROPPING_BALE
611 if not self:getIsAnimationPlaying(spec.currentWrapper.animations["dropFromWrapp then
612 g_server:broadcastEvent(BaleWrapperStateEvent:new(self, BaleWrapper.CHANGE_WRAPPER_BALE_DROPPED), true, nil, self)
613 end
614 elseif spec.baleWrapperState == BaleWrapper.STATE_WRAPPER_RESETTING_PLAT
615 if not self:getIsAnimationPlaying(spec.currentWrapper.animations["resetAfterDro then
616 g_server:broadcastEvent(BaleWrapperStateEvent:new(self, BaleWrapper.CHANGE_WRAPPER_PLATFORM_RESET), true, nil, self)
617 end
618 end
619 end
620 end
621
622 -- update action event visibility and text
623 local actionEvent = spec.actionEvents[InputAction.IMPLEMENT_EXTRA3]
624 if actionEvent ~= nil then
625 g_inputBinding:setActionEventActive(actionEvent.actionEventId, spec.bale == BaleWrapper.STATE_WRAPPER_FINSIHED)
```

```
626 g_inputBinding:setActionEventText(actionEvent.actionEventId,
    g_i18n:getText(spec.currentWrapper.unloadBaleText))
627 end
628
629 if spec.setWrappingStateFinished then
630 g_server:broadcastEvent(BaleWrapperStateEvent:new(self,
    BaleWrapper.CHANGE_WRAPPING_BALE_FINSIHED), true, nil, self)
631 spec.setWrappingStateFinished = false
632 end
633 end
```

onDraw

Description

Called on draw

Definition

onDraw()

Code

```
637 function BaleWrapper:onDraw(isActiveForInput, isSelected)
638 if self.isClient then
639 local spec = self.spec_baleWrapper
640 if spec.showInvalidBaleWarning then
641 g_currentMission:showBlinkingWarning(g_i18n:getText("warning_baleNotSupp
    500)
642 end
643 end
644 end
```

allowsGrabbingBale

Description

Returns if allows bale grabbing

Definition

allowsGrabbingBale()

Return Values

boolean allows allows bale grabbing

Code

```
649 function BaleWrapper:allowsGrabbingBale()
650 local spec = self.spec_baleWrapper
651 local specFoldable = self.spec_foldable
652 if specFoldable ~= nil then
653 if specFoldable.foldAnimTime ~= nil and
    (specFoldable.foldAnimTime > spec.currentWrapperFoldMaxLimit or
    specFoldable.foldAnimTime < spec.currentWrapperFoldMinLimit) then
654 return false
655 end
```

```
656 end
657
658 return spec.baleWrapperState == BaleWrapper.STATE_NONE
659 end
```

updateWrapNodes

Description

Update bale wrap nodes

Definition

updateWrapNodes(boolean isWrapping, boolean isEmpty, float t, float wrapperRot)

Arguments

boolean	isWrapping	is wrapping
boolean	isEmpty	is empty
float	t	animation time
float	wrapperRot	rotation of wrapper

Code

```
667 function BaleWrapper:updateWrapNodes(isWrapping, isEmpty, t,
    wrapperRot)
668 local spec = self.spec_baleWrapper
669
670 if wrapperRot == nil then
671 wrapperRot = 0
672 end
673
674 for _, wrapNode in pairs(spec.currentWrapper.wrapNodes) do
675 local doShow = true
676 if wrapNode.maxWrapperRot ~= nil then
677 doShow = wrapperRot < wrapNode.maxWrapperRot
678 end
679 setVisibility(wrapNode.nodeId, doShow and ((isWrapping and
    wrapNode.wrapVisibility) or (isEmpty and
    wrapNode.emptyVisibility)))
680 end
681
682 if isWrapping then
683 local wrapperRotRepeat = MathUtil.sign(wrapperRot) * (wrapperRot
    % math.pi)
684 if wrapperRotRepeat < 0 then
685 wrapperRotRepeat = wrapperRotRepeat + math.pi
686 end
687
688 for _,wrapAnimNode in pairs(spec.currentWrapper.wrapAnimNodes) do
689 local v
```

```
690  if wrapAnimNode.useWrapperRot then
691  local rot = wrapperRot
692  if wrapAnimNode.repeatWrapperRot then
693  rot = wrapperRotRepeat
694  end
695  v = wrapAnimNode.animCurve:get(rot)
696  else
697  v = wrapAnimNode.animCurve:get(t)
698  end
699  if v ~= nil then
700  setTranslation(wrapAnimNode.nodeId, v[1], v[2], v[3])
701  setRotation(wrapAnimNode.nodeId, v[4], v[5], v[6])
702  setScale(wrapAnimNode.nodeId, v[7], v[8], v[9])
703  end
704  end
705  else
706  if not isEmpty then
707  for _,wrapAnimNode in pairs(spec.currentWrapper.wrapAnimNodes) do
708  if wrapAnimNode.normalizeRotationOnBaleDrop ~= 0 then
709  local rot = { getRotation(wrapAnimNode.nodeId) }
710  for i=1,3 do
711  rot[i] = wrapAnimNode.normalizeRotationOnBaleDrop *
     MathUtil.sign(rot[i]) * ( rot[i] % (2*math.pi) )
712  end
713  setRotation(wrapAnimNode.nodeId, rot[1],rot[2],rot[3])
714  end
715  end
716  end
717  end
718  end
```

updateWrappingState

Description

Update wrapping state

Definition

updateWrappingState(float t, boolean noEventSend)

Arguments

float	t	animation time
boolean	noEventSend	no event send

Code

```
724  function BaleWrapper:updateWrappingState(t, noEventSend)
725  local spec = self.spec_baleWrapper
```

```
726
727 t = math.min(t, 1)
728 local wrapperRot = 0
729 if spec.currentWrapper.animCurve ~= nil then
730 local v = spec.currentWrapper.animCurve:get(t)
731 if v ~= nil then
732 setRotation(spec.currentWrapper.baleNode, v[1]%(math.pi*2),
    v[2]%(math.pi*2), v[3]%(math.pi*2))
733 setRotation(spec.currentWrapper.wrapperNode, v[4]%(math.pi*2),
    v[5]%(math.pi*2), v[6]%(math.pi*2))
734 wrapperRot = v[3 + spec.currentWrapper.wrapperRotAxis]
735 elseif spec.currentWrapper.animations["wrapBale"].animName ~= nil then
736 t =
    self:getAnimationTime(spec.currentWrapper.animations["wrapBale"].animNam
737 end
738 if spec.currentWrapper.currentBale ~= nil then
739 local bale = NetworkUtil.getObject(spec.currentWrapper.currentBale)
740 if bale ~= nil then
741 if self.isServer then
742 local wrappingState = t
743 if table.getn(spec.currentWrapper.wrappingStateCurve.keyframes) > 0 then
744 wrappingState = spec.currentWrapper.wrappingStateCurve:get(t)
745 end
746 bale:setWrappingState(wrappingState)
747 end
748
749 if bale.setColor ~= nil then
750 local color = ConfigurationUtil.getColorByConfigId(self, "wrappingColor"
751 if color ~= nil then
752 local r, g, b, a = unpack(color)
753 bale:setColor(r, g, b, a)
754 end
755 end
756 end
757 end
758 end
759 self:updateWrapNodes(t > 0, false, t, wrapperRot)
760 if t == 1 then
761 if self.isServer and spec.baleWrapperState ==
    BaleWrapper.STATE_WRAPPER_WRAPPING_BALE and not noEventSend then
```

```
762 g_server:broadcastEvent(BaleWrapperStateEvent:new(self,
    BaleWrapper.CHANGE_WRAPPING_BALE_FINSIHED), true, nil, self)
763 end
764 end
765 end
```

playMoveToWrapper

Description

Play move to wrapper animation

Definition

playMoveToWrapper(table bale)

Arguments

table bale bale to move

Code

```
770 function BaleWrapper:playMoveToWrapper(bale)
771 local spec = self.spec_baleWrapper
772
773 spec.currentWrapper = spec.roundBaleWrapper
774 if bale.baleDiameter == nil then
775 spec.currentWrapper = spec.squareBaleWrapper
776 end
777
778 if spec.currentWrapper.animations["moveToWrapper"].animName ~= nil then
779 self:playAnimation(spec.currentWrapper.animations["moveToWrapper"].animN
    spec.currentWrapper.animations["moveToWrapper"].animSpeed, nil, true)
780 end
781 end
```

doStateChange

Description

Changed wrapper state

Definition

doStateChange(integer id, integer nearestBaleServerId)

Arguments

integer id id of new state

integer nearestBaleServerId server id of nearest bale

Code

```
787 function BaleWrapper:doStateChange(id, nearestBaleServerId)
788 local spec = self.spec_baleWrapper
789
790 if id == BaleWrapper.CHANGE_WRAPPING_START or (spec.baleWrapperState ~=
    BaleWrapper.STATE_WRAPPER_FINSIHED and id ==
    BaleWrapper.CHANGE_WRAPPER_START_DROP_BALE) then
```

```
791 if self:getIsBaleFillTypeSkiped(NetworkUtil.getObject(spec.currentWrapper.c then
792 if self.isServer then
793 spec.setWrappingStateFinished = true
794 end
795
796 return
797 end
798 end
799
800 if id == BaleWrapper.CHANGE_GRAB_BALE then
801 local bale = NetworkUtil.getObject(nearestBaleServerId)
802 spec.baleGrabber.currentBale = nearestBaleServerId
803 if bale ~= nil then
804 local x,y,z = localToLocal(bale.nodeId, getParent(spec.baleGrabber.grabN
805 setTranslation(spec.baleGrabber.grabNode, x,y,z)
806 bale:mount(self, spec.baleGrabber.grabNode, 0,0,0, 0,0,0)
807 spec.baleToMount = nil
808 spec:playMoveToWrapper(bale)
809 else
810 spec.baleToMount = {serverId=nearestBaleServerId, linkNode=spec.baleGrabber.grabNode, trans={0,0,0}, rot={0,0,0} }
811 end
812 spec.baleWrapperState = BaleWrapper.STATE_MOVING_BALE_TO_WRAPPER
813
814 elseif id == BaleWrapper.CHANGE_DROP_BALE_AT_GRABBER then
815 -- drop bale at wrapper
816 local attachNode = spec.currentWrapper.baleNode
817 local bale = NetworkUtil.getObject(spec.baleGrabber.currentBale)
818 if bale ~= nil then
819 bale:mount(self, attachNode, 0,0,0, 0,0,0)
820 spec.baleToMount = nil
821 else
822 spec.baleToMount = {serverId=spec.baleGrabber.currentBale, linkNode=atta trans={0,0,0}, rot={0,0,0} }
823 end
824 self:updateWrapNodes(true, false, 0)
825
826 spec.currentWrapper.currentBale = spec.baleGrabber.currentBale
827 spec.baleGrabber.currentBale = nil
```

```
828
829 if spec.currentWrapper.animations["moveToWrapper"].animName ~= nil then
830 if spec.currentWrapper.animations["moveToWrapper"].reverseAfterMove ther
831 self:playAnimation(spec.currentWrapper.animations["moveToWrapper"].animN
    spec.currentWrapper.animations["moveToWrapper"].animSpeed, nil, true)
832 end
833 end
834
835 spec.baleWrapperState = BaleWrapper.STATE_MOVING_GRABBER_TO_WORK
836
837 elseif id == BaleWrapper.CHANGE_WRAPPING_START then
838
839 spec.baleWrapperState = BaleWrapper.STATE_WRAPPER_WRAPPING_BALE
840 if self.isClient then
841 g_soundManager:playSample(spec.currentWrapper.samples.start)
842 g_soundManager:playSample(spec.currentWrapper.samples.wrap, 0,
    spec.currentWrapper.samples.start)
843 end
844
845 if spec.currentWrapper.animations["wrapBale"].animName ~= nil then
846 self:playAnimation(spec.currentWrapper.animations["wrapBale"].animName,
    spec.currentWrapper.animations["wrapBale"].animSpeed, nil, true)
847 end
848
849 if self.isServer then
850 for _, collision in pairs(spec.currentWrapper.collisions) do
851 setCollisionMask(collision.node, collision.activeCollisionMask);
852 end;
853 end;
854 elseif id == BaleWrapper.CHANGE_WRAPPING_BALE_FINSIHED then
855
856 if self.isClient then
857 g_soundManager:stopSample(spec.currentWrapper.samples.wrap)
858 g_soundManager:stopSample(spec.currentWrapper.samples.stop)
859 if spec.currentWrapper.wrappingSoundEndTime == 1 then
860 g_soundManager:playSample(spec.currentWrapper.samples.stop)
861 end
862
863 -- if the start sound is still playing (e.g. on loading if the wrapping
    directly set to 1) we stop it
864 if g_soundManager:getIsSamplePlaying(spec.currentWrapper.samples.start)
```

```
865 g_soundManager:stopSample(spec.currentWrapper.samples.start)
866 end
867 end
868
869 self:updateWrappingState(1, true)
870 spec.baleWrapperState = BaleWrapper.STATE_WRAPPER_FINSIHED
871
872 if self:getIsBaleFillTypeSkiped(NetworkUtil.getObject(spec.currentWrapper.c
    then
873 self:updateWrappingState(0, true)
874 end
875 elseif id == BaleWrapper.CHANGE_WRAPPER_START_DROP_BALE then
876
877 self:updateWrapNodes(false, false, 0)
878 if spec.currentWrapper.animations["dropFromWrapper"].animName ~= nil the
879 self:playAnimation(spec.currentWrapper.animations["dropFromWrapper"].ani
    spec.currentWrapper.animations["dropFromWrapper"].animSpeed, nil, true)
880 end
881 spec.baleWrapperState = BaleWrapper.STATE_WRAPPER_DROPPING_BALE
882
883 if self.isServer then
884 for _, collision in pairs(spec.currentWrapper.collisions) do
885 setCollisionMask(collision.node, collision.inActiveCollisionMask);
886 end;
887 end;
888 elseif id == BaleWrapper.CHANGE_WRAPPER_BALE_DROPPED then
889
890 local bale = NetworkUtil.getObject(spec.currentWrapper.currentBale)
891 if bale ~= nil then
892 bale:unmount()
893 end
894 spec.lastDroppedBale = bale
895 spec.currentWrapper.currentBale = nil
896 spec.currentWrapper.currentTime = 0
897 if spec.currentWrapper.animations["resetAfterDrop"].animName ~= nil then
898 self:playAnimation(spec.currentWrapper.animations["resetAfterDrop"].anim
    spec.currentWrapper.animations["resetAfterDrop"].animSpeed, nil, true)
899 end
900 spec.baleWrapperState = BaleWrapper.STATE_WRAPPER_RESETTING_PLATFORM
901
```

```
902 elseif id == BaleWrapper.CHANGE_WRAPPER_PLATFORM_RESET then
903
904 self:updateWrappingState(0)
905 self:updateWrapNodes(false, true, 0)
906 spec.baleWrapperState = BaleWrapper.STATE_NONE
907
908 elseif id == BaleWrapper.CHANGE_BUTTON_EMPTY then
909
910 -- Server only code
911 assert(self.isServer)
912 if spec.baleWrapperState == BaleWrapper.STATE_WRAPPER_FINSIHED then
913 g_server:broadcastEvent(BaleWrapperStateEvent:new(self,
    BaleWrapper.CHANGE_WRAPPER_START_DROP_BALE), true, nil, self)
914 end
915
916 end
917 end
```

getWrapperBaleType

Description

Returns type of wrapped bale

Definition

getWrapperBaleType(table bale)

Arguments

table bale unwrapped bale

Return Values

table baleType wrapped bale

Code

```
923 function BaleWrapper:getWrapperBaleType(bale)
924 local spec = self.spec_baleWrapper
925
926 local baleTypes
927 if bale.baleDiameter ~= nil then
928 baleTypes =
    spec.roundBaleWrapper.allowedBaleTypes[bale:getFillType()]
929 else
930 baleTypes =
    spec.squareBaleWrapper.allowedBaleTypes[bale:getFillType()]
931 end
932 if baleTypes ~= nil then
933 for _, baleType in pairs(baleTypes) do
934 if bale.baleDiameter ~= nil and bale.baleWidth ~= nil then
```

```
935  if bale.baleDiameter >= baleType.minBaleDiameter and
     bale.baleDiameter <= baleType.maxBaleDiameter and
936  bale.baleWidth >= baleType.minBaleWidth and bale.baleWidth <=
     baleType.maxBaleWidth
937  then
938  return baleType
939  end
940  elseif bale.baleHeight ~= nil and bale.baleWidth ~= nil and
     bale.baleLength ~= nil then
941  if bale.baleHeight >= baleType.minBaleHeight and bale.baleHeight
     <= baleType.maxBaleHeight and
942  bale.baleWidth >= baleType.minBaleWidth and bale.baleWidth <=
     baleType.maxBaleWidth and
943  bale.baleLength >= baleType.minBaleLength and bale.baleLength <=
     baleType.maxBaleLength
944  then
945  return baleType
946  end
947  end
948  end
949  end
950  return nil
951  end
```

pickupWrapperBale

Description

Pickup bale to wrap

Definition

pickupWrapperBale(table bale, integer baleType)

Arguments

table	bale	bale to pickup
integer	baleType	type of bale

Code

```
957  function BaleWrapper:pickupWrapperBale(bale, baleType)
958  if baleType ~= nil and bale.i3dFilename ~=
     baleType.wrapperBaleFilename then
959  local x,y,z = getWorldTranslation(bale.nodeId)
960  local rx,ry,rz = getWorldRotation(bale.nodeId)
961  local fillLevel = bale.fillLevel
962  local baleValueScale = bale.baleValueScale
963  bale:delete()
964
965  bale = Bale:new(self.isServer, self.isClient)
```

```
966  bale:load(baleType.wrapperBaleFilename, x,y,z, rx,ry,rz,
     fillLevel)
967  bale.baleValueScale = baleValueScale
968  bale:register()
969  end
970
971  -- found bale
972  g_server:broadcastEvent(BaleWrapperStateEvent:new(self,
     BaleWrapper.CHANGE_GRAB_BALE, NetworkUtil.getObjectId(bale)),
     true, nil, self)
973  end
```

getBaleInRange

Description

Returns if nearest bale in range

Definition

getBaleInRange(integer refNode, float distance)

Arguments

integer refNode id of reference node
float distance max distance

Return Values

table nearestBale nearest bale
integer nearestBaleType id of bale type

Code

```
1001  function BaleWrapper.getBaleInRange(self, refNode, distance)
1002  local nearestDistance = distance
1003  local nearestBale = nil
1004  local nearestBaleType
1005
1006  for _, item in pairs(g_currentMission.itemsToSave) do
1007  local bale = item.item
1008  if bale:isa(Bale) then
1009  local maxDist
1010  if bale.baleDiameter ~= nil then
1011  maxDist = math.min(bale.baleDiameter, bale.baleWidth)
1012  else
1013  maxDist = math.min(bale.baleLength, bale.baleHeight,
     bale.baleWidth)
1014  end
1015  local _,_,z = localToLocal(bale.nodeId, refNode, 0,0,0)
1016  if math.abs(z) < maxDist and calcDistanceFrom(refNode,
     bale.nodeId) < nearestDistance then
1017  local foundBaleType
```

```
1018  if not bale.supportsWrapping or bale.wrappingState < 0.99 then
1019  foundBaleType = self:getWrapperBaleType(bale)
1020  end
1021  if foundBaleType ~= nil or nearestBaleType == nil then
1022  if foundBaleType ~= nil then
1023  nearestDistance = distance
1024  end
1025  nearestBale = bale
1026  nearestBaleType = foundBaleType
1027  end
1028  end
1029  end
1030  end
1031  return nearestBale, nearestBaleType
1032  end
```

getIsFoldAllowed

Description

Returns if fold is allowed

Definition

getIsFoldAllowed(boolean onAiTurnOn)

Arguments

boolean onAiTurnOn called on ai turn on

Return Values

boolean allowsFold allows folding

Code

```
1038  function BaleWrapper:getIsFoldAllowed(superFunc, direction,
      onAiTurnOn)
1039  local spec = self.spec_baleWrapper
1040  if spec.baleWrapperState ~= BaleWrapper.STATE_NONE then
1041  return false
1042  end
1043
1044  return superFunc(self, direction, onAiTurnOn)
1045  end
```

onDeactivate

Description

Called on deactivating

Definition

onDeactivate()

Code

```
1067  function BaleWrapper:onDeactivate()
```

```
1068  local spec = self.spec_baleWrapper
1069  spec.showInvalidBaleWarning = false
1070  if self.isClient then
1071  for _, sample in pairs(spec.currentWrapper.samples) do
1072  g_soundManager:stopSample(sample)
1073  end
1074  end
1075  end
```

CCTDrivable

Description

Specialization class for CCTDrivables

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
20  function CCTDrivable.prerequisitesPresent(specializations)
21  return SpecializationUtil.hasSpecialization(Enterable,
    specializations)
22  end
```

registerFunctions

Description

Registers functions

Definition

registerFunctions(string vehicleType)

Arguments

string vehicleType type of vehicle

Code

```
27  function CCTDrivable.registerFunctions(vehicleType)
28  SpecializationUtil.registerFunction(vehicleType, "moveCCT",
    CCTDrivable.moveCCT)
29  SpecializationUtil.registerFunction(vehicleType,
    "getIsCCTOnGround", CCTDrivable.getIsCCTOnGround)
30  SpecializationUtil.registerFunction(vehicleType,
    "getCCTCollisionMask", CCTDrivable.getCCTCollisionMask)
31  SpecializationUtil.registerFunction(vehicleType,
    "getCCTWorldTranslation", CCTDrivable.getCCTWorldTranslation)
32  end
```

registerEventListeners

Description

Registers event listeners

Definition

registerEventListeners(string vehicleType)

Arguments

string vehicleType type of vehicle

Code

```
42  function CCTDrivable.registerEventListeners(vehicleType)
43  SpecializationUtil.registerEventListener(vehicleType, "onLoad",
    CCTDrivable)
44  SpecializationUtil.registerEventListener(vehicleType, "onDelete",
    CCTDrivable)
45  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
50  function CCTDrivable:onLoad(savegame)
51  local spec = self.spec_cctdrivable
52
53  spec.cctRadius = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.cctDrivable#cctRadius"), 1.0)
54  spec.cctHeight = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.cctDrivable#cctHeight"), 1.0)
55  spec.customOffset = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.cctDrivable#customOffset"), 0)
56  spec.cctCenterOffset = spec.cctRadius + spec.cctHeight*0.5
57
58  spec.kinematicCollisionMask = 4 -- 2147483648 + 1048576 + 4
59  spec.movementCollisionMask = 31 -- 2147483648 + 1048576 + 31
60  if self.isServer then
61  -- CCT
62  local mass = 1000
63  spec.cctNode = createTransformGroup("cctDrivable")
64  link(getRootNode(), spec.cctNode)
65  spec.controllerIndex = createCCT(spec.cctNode, spec.cctRadius,
    spec.cctHeight, 0.6, 45.0, 0.1, spec.kinematicCollisionMask, mass)
66  end
```

```
67  end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
71  function CCTDrivable:onDelete()
72  local spec = self.spec_cctdrivable
73
74  if spec.controllerIndex ~= nil then
75  removeCCT(spec.controllerIndex)
76  end
77  end
```

moveCCT

Description

Definition

moveCCT()

Code

```
81  function CCTDrivable:moveCCT(moveX, moveY, moveZ)
82  if self.isServer then
83  local spec = self.spec_cctdrivable
84  -- move cct
85  moveCCT(spec.controllerIndex, moveX, moveY, moveZ,
    spec.movementCollisionMask)
86  self:raiseActive()
87  end
88  end
```

getIsCCTOnGround

Description

Definition

getIsCCTOnGround()

Return Values

int returns the CCT index

Code

```
93  function CCTDrivable:getIsCCTOnGround()
94  local spec = self.spec_cctdrivable
95  if self.isServer then
96  local _, _, isOnGround =
    getCCTCollisionFlags(spec.controllerIndex)
97  return isOnGround
98  end
```

```
 99  return false
100  end
```

getCCTCollisionMask

Description

Definition

getCCTCollisionMask()

Return Values

int returns the collision mask

Code

```
105  function CCTDrivable:getCCTCollisionMask()
106  local spec = self.spec_cctdrivable
107  return spec.kinematicCollisionMask
108  end
```

getCCTWorldTranslation

Description

Definition

getCCTWorldTranslation()

Return Values

float x position of center of CCT

float y position of center of CCT

float z position of center of CCT

Code

```
115  function CCTDrivable:getCCTWorldTranslation()
116  local spec = self.spec_cctdrivable
117  local cctX, cctY, cctZ = getTranslation(spec.cctNode)
118  cctY = cctY-spec.cctCenterOffset
119  return cctX, cctY, cctZ
120  end
```

setWorldPosition

Description

Set world position and rotation of component

Definition

setWorldPosition(float x, float y, float z, float xRot, float yRot, float zRot, integer i, boolean changeInterp)

Arguments

float	x	x position
float	y	y position
float	z	z position
float	xRot	x rotation
float	yRot	y rotation
float	zRot	z rotation
integer	i	index if component
boolean	changeInterp	change interpolation

Code

```
132  function CCTDrivable:setWorldPosition(superFunc, x,y,z,
     xRot,yRot,zRot, i, changeInterp)
133  superFunc(self, x,y,z, xRot,yRot,zRot, i, changeInterp)
134  if self.isServer and i == 1 then
135  local spec = self.spec_cctdrivable
136  setTranslation(spec.cctNode, x, y + spec.cctCenterOffset, z)
137  end
138  end
```

setWorldPositionQuaternion

Description

Set world position and quaternion rotation of component

Definition

setWorldPositionQuaternion(float x, float y, float z, float qx, float qy, float qz, float qw, integer i, boolean changeInterp)

Arguments

float	x	x position
float	y	y position
float	z	z position
float	qx	x rotation
float	qy	y rotation
float	qz	z rotation
float	qw	w rotation
integer	i	index if component
boolean	changeInterp	change interpolation

Code

```
151  function CCTDrivable:setWorldPositionQuaternion(superFunc, x, y,
     z, qx, qy, qz, qw, i, changeInterp)
152  superFunc(self, x, y, z, qx, qy, qz, qw, i, changeInterp)
153  if self.isServer and i == 1 then
154  local spec = self.spec_cctdrivable
155  setTranslation(spec.cctNode, x, y + spec.cctCenterOffset, z)
156  end
157  end
```

Combine

Description

onPostLoad

Description

Called after loading

Definition

onPostLoad(table savegame)

Arguments

table	savegame	savegame

Code

```
146 function Combine:onPostLoad(savegame)
147 local spec = self.spec_combine
148
149 if savegame ~= nil then
150 if spec.swath.isAvailable then
151 local isSwathActive = Utils.getNoNil(getXMLBool(savegame.xmlFile,
    savegame.key .. ".combine#isSwathActive"), spec.isSwathActive)
152 self:setIsSwathActive(isSwathActive, true, true)
153 end
154 self:setWorkedHectars(Utils.getNoNil(getXMLFloat(savegame.xmlFile,
    savegame.key .. ".combine#workedHectars"), spec.workedHectars))
155 else
156 self:setIsSwathActive(spec.isSwathActive, true, true)
157 end
158
159 local ladder = spec.ladder
160 if ladder.animName ~= nil then
161 local time = 0
162 if self.getFoldAnimTime ~= nil then
163 local foldAnimTime = self:getFoldAnimTime()
164 if foldAnimTime > ladder.foldMaxLimit or foldAnimTime <
    ladder.foldMinLimit then
165 time = 1
166 end
167 end
168
169 if ladder.foldDirection ~= 1 then
170 time = 1 - time
171 end
172
173 self:setAnimationTime(ladder.animName, time, true)
174 end
175
176 if self:getFillUnitCapacity(spec.fillUnitIndex) == 0 then
177 g_logManager:xmlWarning(self.configFileName, "Capacity of fill
    unit '%d' for combine needs to be set greater 0 or not defined!
    (not defined = infinity)", spec.fillUnitIndex)
178 end
179 end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
function Combine:onDelete()
if self.isClient then
local spec = self.spec_combine

g_effectManager:deleteEffects(spec.fillEffects)
g_effectManager:deleteEffects(spec.strawEffects)
g_effectManager:deleteEffects(spec.chopperEffects)

g_animationManager:deleteAnimations(spec.animationNodes)

g_soundManager:deleteSamples(spec.samples)
end
end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
function Combine:onReadStream(streamId, connection)
local spec = self.spec_combine

spec.lastValidInputFruitType = streamReadUIntN(streamId, FruitTypeManager.SEND_NUM_BITS)
local combineIsFilling = streamReadBool(streamId)
local chopperPSenabled = streamReadBool(streamId)
local strawPSenabled = streamReadBool(streamId)

self:setCombineIsFilling(combineIsFilling, false, true)
self:setChopperPSEnabled(chopperPSenabled, false, true)
self:setStrawPSEnabled(strawPSenabled, false, true)

local isSwathActive = streamReadBool(streamId)
self:setIsSwathActive(isSwathActive, true)
```

```
224
225 local workedHectars = streamReadFloat32(streamId)
226 self:setWorkedHectars(workedHectars)
227 end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
233 function Combine:onWriteStream(streamId, connection)
234 local spec = self.spec_combine
235
236 streamWriteUIntN(streamId, spec.lastValidInputFruitType,
    FruitTypeManager.SEND_NUM_BITS)
237 streamWriteBool(streamId, spec.isFilling)
238 streamWriteBool(streamId, spec.chopperPSenabled)
239 streamWriteBool(streamId, spec.strawPSenabled)
240 streamWriteBool(streamId, spec.isSwathActive)
241
242 streamWriteFloat32(streamId, spec.workedHectars)
243 end
```

onReadUpdateStream

Description

Called on on update

Definition

onReadUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId stream ID

integer timestamp timestamp

table connection connection

Code

```
250 function Combine:onReadUpdateStream(streamId, timestamp,
    connection)
251 if connection:getIsServer() then
252 if streamReadBool(streamId) then
253 local spec = self.spec_combine
254 spec.lastValidInputFruitType = streamReadUIntN(streamId,
    FruitTypeManager.SEND_NUM_BITS)
```

```
255
256 local combineIsFilling = streamReadBool(streamId)
257 local chopperPSenabled = streamReadBool(streamId)
258 local strawPSenabled = streamReadBool(streamId)
259 self:setCombineIsFilling(combineIsFilling, false, true)
260 self:setChopperPSEnabled(chopperPSenabled, false, true)
261 self:setStrawPSEnabled(strawPSenabled, false, true)
262
263 local workedHectars = streamReadFloat32(streamId)
264 self:setWorkedHectars(workedHectars)
265 end
266 end
267 end
```

onWriteUpdateStream

Description

Called on on update

Definition

onWriteUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Code

```
274 function Combine:onWriteUpdateStream(streamId, connection,
    dirtyMask)
275 if not connection:getIsServer() then
276 local spec = self.spec_combine
277
278 if streamWriteBool(streamId, bitAND(dirtyMask, spec.dirtyFlag) ~=
    0) then
279 streamWriteUIntN(streamId, spec.lastValidInputFruitType,
    FruitTypeManager.SEND_NUM_BITS)
280 streamWriteBool(streamId, spec.isFilling)
281 streamWriteBool(streamId, spec.chopperPSenabled)
282 streamWriteBool(streamId, spec.strawPSenabled)
283
284 streamWriteFloat32(streamId, spec.workedHectars)
285 end
286 end
287 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
294 function Combine:onUpdate(dt, isActiveForInput, isSelected)
295 local spec = self.spec_combine
296
297 local isTurnedOn = self:getIsTurnedOn()
298 if isTurnedOn then
299 if self.isServer and spec.swath.isAvailable then
300 -- check for changes while threshing, e.g. activated straw swath and the
    (has no windrow)
301 local fruitType =
    g_fruitTypeManager:getFruitTypeIndexByFillTypeIndex(self:getFillUnitFill
302 if spec.isSwathActive and fruitType ~= nil and fruitType ~= FruitType.UN
303 local fruitDesc = g_fruitTypeManager:getFruitTypeByIndex(fruitType)
304 if not fruitDesc.hasWindrow then
305 self:setIsSwathActive(false)
306 end
307 end
308 end
309 end
310
311 if self.isServer then
312 if self:getFillUnitFillLevel(spec.fillUnitIndex) < 0.0001 then
313 spec.lastDischargeTime = g_time
314 end
315 end
316 end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
323 function Combine:onUpdateTick(dt, isActiveForInput, isSelected)
324 local spec = self.spec_combine
325
326 if self.isServer then
327 spec.lastArea = spec.lastCuttersArea
328 spec.lastAreaZeroTime = spec.lastAreaZeroTime + dt
329 if spec.lastArea > 0 then
330 spec.lastAreaZeroTime = 0
331 spec.lastAreaNonZeroTime = g_currentMission.time
332 end
333 spec.lastInputFruitType = spec.lastCuttersInputFruitType
334 spec.lastCuttersArea = 0
335 spec.lastCuttersInputFruitType = FruitType.UNKNOWN
336 spec.lastCuttersFruitType = FruitType.UNKNOWN
337
338 if spec.lastInputFruitType ~= FruitType.UNKNOWN then
339 spec.lastValidInputFruitType = spec.lastInputFruitType
340 end
341
342 local inputBuffer = spec.processing.inputBuffer
343
344 if spec.lastAreaZeroTime > 500 then
345 if spec.fillDisableTime == nil then
346 spec.fillDisableTime = g_currentMission.time +
    spec.processing.toggleTime
347 end
348 end
349 if spec.fillEnableTime ~= nil and spec.fillEnableTime <=
    g_currentMission.time then
350 self:setCombineIsFilling(true, false, false)
351 spec.fillEnableTime = nil
352 end
353 if spec.fillDisableTime ~= nil and spec.fillDisableTime <=
    g_currentMission.time then
354 self:setCombineIsFilling(false, false, false)
355 spec.fillDisableTime = nil
356 end
357
358 if self:getIsTurnedOn() then
```

```
359 local stats = g_currentMission:farmStats(self:getLastTouchedFarmlandFarmId())
360 stats:updateStats("threshedTime", dt / (1000 * 60))
361 stats:updateStats("workedTime", dt / (1000 * 60))
362 end
363
364 inputBuffer.slotTimer = inputBuffer.slotTimer - dt
365 if inputBuffer.slotTimer < 0 then
366 inputBuffer.slotTimer = inputBuffer.slotDuration
367
368 inputBuffer.fillIndex = inputBuffer.fillIndex + 1
369 if inputBuffer.fillIndex > inputBuffer.slotCount then
370 inputBuffer.fillIndex = 1
371 end
372
373 local lastDropIndex = inputBuffer.dropIndex
374 inputBuffer.dropIndex = inputBuffer.dropIndex + 1
375 if inputBuffer.dropIndex > inputBuffer.slotCount then
376 inputBuffer.dropIndex = 1
377 end
378
379 inputBuffer.buffer[inputBuffer.dropIndex].liters = inputBuffer.buffer[inputBuffer.dropIndex].liters + inputBuffer.buffer[lastDropIndex].liters
380 inputBuffer.buffer[inputBuffer.dropIndex].inputLiters = inputBuffer.buffer[inputBuffer.dropIndex].inputLiters + inputBuffer.buffer[lastDropIndex].liters --inputLiters
381
382 inputBuffer.buffer[lastDropIndex].area = 0
383 inputBuffer.buffer[lastDropIndex].realArea = 0
384 inputBuffer.buffer[lastDropIndex].liters = 0
385 inputBuffer.buffer[lastDropIndex].inputLiters = 0
386 end
387
388 if spec.isFilling ~= spec.sentIsFilling or spec.chopperPSenabled ~= spec.sentChopperPSenabled or spec.strawPSenabled ~= spec.sentStrawPSenabled then
389 self:raiseDirtyFlags(spec.dirtyFlag)
390 spec.sentIsFilling = spec.isFilling
391 spec.sentChopperPSenabled = spec.chopperPSenabled
392 spec.sentStrawPSenabled = spec.strawPSenabled
393 end
```

```
394 end
395 end
```

onDraw

Description

Called on draw

Definition

onDraw(boolean isActiveForInput, boolean isSelected)

Arguments

boolean isActiveForInput	true if vehicle is active for input
boolean isSelected	true if vehicle is selected

Code

```
401 function Combine:onDraw(isActiveForInput, isSelected)
402 if self:getIsTurnedOn() and not self:getIsThreshingAllowed(false) then
403 g_currentMission:showBlinkingWarning(g_i18n:getText("warning_doNotThresh
    2000)
404 end
405 end
```

setIsSwathActive

Description

Set is straw enabled

Definition

setIsSwathActive(boolean isSwathActive, boolean noEventSend, boolean force)

Arguments

boolean isSwathActive	new state
boolean noEventSend	no event send
boolean force	force action

Code

```
531 function Combine:setIsSwathActive(isSwathActive, noEventSend,
    force)
532 local spec = self.spec_combine
533
534 if isSwathActive ~= spec.isSwathActive or force then
535 CombineStrawEnableEvent.sendEvent(self, isSwathActive,
    noEventSend)
536 spec.isSwathActive = isSwathActive
537
538 local anim = spec.chopper.animName
539 if self.playAnimation ~= nil and anim ~= nil then
540 local dir = 1
541 if isSwathActive then
542 dir = -1
543 end
```

```
544 self:playAnimation(anim, dir, self:getAnimationTime(anim), true)
545
546 if force then
547 AnimatedVehicle.updateAnimationByName(self, anim, 9999999)
548 end
549 end
550
551 Combine.updateToggleStrawText(self)
552 end
553 end
```

setChopperPSEnabled

Description

Set chopper PS enabled

Definition

setChopperPSEnabled(boolean chopperPSenabled, boolean fruitTypeChanged, boolean isSynchronized)

Arguments

boolean chopperPSenabled	set enabled
boolean fruitTypeChanged	fruit type has changed
boolean isSynchronized	filling is synchronized

Code

```
606 function Combine:setChopperPSEnabled(chopperPSenabled,
    fruitTypeChanged, isSynchronized)
607 local spec = self.spec_combine
608
609 if spec.chopperPSenabled ~= chopperPSenabled or fruitTypeChanged
    then
610 if spec.chopperEffects ~= nil and (not chopperPSenabled or
    fruitTypeChanged) then
611 g_effectManager:stopEffects(spec.chopperEffects)
612 end
613
614 spec.chopperPSenabled = chopperPSenabled
615 if self.isServer and isSynchronized then
616 spec.sentChopperPSenabled = chopperPSenabled
617 end
618 if chopperPSenabled and self.isClient then
619 if spec.chopperEffects ~= nil then
620 g_effectManager:setFillType(spec.chopperEffects,
    self:getFillUnitLastValidFillType(spec.fillUnitIndex))
621 g_effectManager:startEffects(spec.chopperEffects)
622 end
```

```
623 end
624 end
625 end
```

setStrawPSEnabled

Description

Set straw PS enabled

Definition

setStrawPSEnabled(boolean strawPSenabled, boolean fruitTypeChanged, boolean isSynchronized)

Arguments

boolean strawPSenabled	set enabled
boolean fruitTypeChanged	fruit type has changed
boolean isSynchronized	filling is synchronized

Code

```
632 function Combine:setStrawPSEnabled(strawPSenabled, fruitTypeChanged, isSynchronized)
633 local spec = self.spec_combine
634
635 if spec.strawPSenabled ~= strawPSenabled or fruitTypeChanged then
636 if spec.strawEffects ~= nil and (not strawPSenabled or fruitTypeChanged) then
637 g_effectManager:stopEffects(spec.strawEffects)
638 end
639
640 spec.strawPSenabled = strawPSenabled
641 if self.isServer and isSynchronized then
642 spec.sentStrawPSenabled = strawPSenabled
643 end
644 if not strawPSenabled then
645 spec.strawToDrop = 0
646 end
647 if strawPSenabled and self.isClient then
648 if spec.strawEffects ~= nil then
649 g_effectManager:setFillType(spec.strawEffects, self:getFillUnitLastValidFillType(spec.fillUnitIndex))
650 g_effectManager:startEffects(spec.strawEffects)
651 end
652 end
653 end
654 end
```

setCombineIsFilling

Description

Set combine is filling

Definition

setCombineIsFilling(boolean isFilling, boolean fruitTypeChanged, boolean isSynchronized)

Arguments

boolean isFilling	combine is filling
boolean fruitTypeChanged	fruit type has changed
boolean isSynchronized	filling is synchronized

Code

```
661 function Combine:setCombineIsFilling(isFilling, fruitTypeChanged, isSynchronized)
662 local spec = self.spec_combine
663
664 if spec.isFilling ~= isFilling or fruitTypeChanged then
665 spec.isFilling = isFilling
666
667 if isFilling then
668 g_animationManager:startAnimations(spec.fillingAnimationNodes)
669 else
670 g_animationManager:stopAnimations(spec.fillingAnimationNodes)
671 end
672
673 g_animationManager:setFillType(spec.fillingAnimationNodes, self:getFillUnitLastValidFillType(spec.fillUnitIndex))
674
675 if self.isServer and isSynchronized then
676 spec.sentIsFilling = isFilling
677 end
678
679 if spec.fillEffects ~= nil and (not isFilling or fruitTypeChanged) then
680 g_effectManager:stopEffects(spec.fillEffects)
681 end
682
683 if isFilling then
684 if spec.fillEffects ~= nil then
685 g_effectManager:setFillType(spec.fillEffects, self:getFillUnitLastValidFillType(spec.fillUnitIndex))
686 g_effectManager:startEffects(spec.fillEffects)
687 end
688 end
689 end
690 end
```

startThreshing

Description

Start threshing

Definition

startThreshing()

Code

```
694 function Combine:startThreshing()
695 local spec = self.spec_combine
696
697 if spec.numAttachedCutters > 0 then
698 for _,cutter in pairs(spec.attachedCutters) do
699 if cutter ~= self then
700 local jointDescIndex =
    self:getAttacherJointIndexFromObject(cutter)
701 self:setJointMoveDown(jointDescIndex, true, true)
702 end
703
704 cutter:setIsTurnedOn(true, true)
705 end
706
707 if spec.threshingStartAnimation ~= nil and self.playAnimation ~=
    nil then
708 self:playAnimation(spec.threshingStartAnimation,
    spec.threshingStartAnimationSpeedScale,
    self:getAnimationTime(spec.threshingStartAnimation), true)
709 end
710
711 if self.isClient then
712 g_soundManager:stopSamples(spec.samples)
713 g_soundManager:playSample(spec.samples.start)
714 g_soundManager:playSample(spec.samples.work, 0,
    spec.samples.start)
715 end
716
717 SpecializationUtil.raiseEvent(self, "onStartThreshing")
718 end
719 end
```

stopThreshing

Description

Stop threshing

Definition

stopThreshing()

Code

```
723 function Combine:stopThreshing()
724 local spec = self.spec_combine
725
726 if self.isClient then
727 g_soundManager:stopSamples(spec.samples)
728 g_soundManager:playSample(spec.samples.stop)
729 end
730
731 self:setCombineIsFilling(false, false, true)
732
733 for cutter,_ in pairs(spec.attachedCutters) do
734 if cutter ~= self then
735 local jointDescIndex =
    self:getAttacherJointIndexFromObject(cutter)
736 self:setJointMoveDown(jointDescIndex, false, true)
737 end
738
739 cutter:setIsTurnedOn(false, true)
740 end
741
742 if spec.threshingStartAnimation ~= nil and spec.playAnimation ~=
    nil then
743 self:playAnimation(spec.threshingStartAnimation, -
    spec.threshingStartAnimationSpeedScale,
    self:getAnimationTime(spec.threshingStartAnimation), true)
744 end
745
746 SpecializationUtil.raiseEvent(self, "onStopThreshing")
747 end
```

setWorkedHectars

Description

Set worked hectars value and updated hud

Definition

setWorkedHectars(float hectars)

Arguments

float hectars new hectars value

Code

```
752 function Combine:setWorkedHectars(hectars)
753 local spec = self.spec_combine
754
755 spec.workedHectars = hectars
```

```
756
757 if self.isServer then
758 if math.abs(spec.workedHectars-spec.workedHectarsSent) > 0.01
    then
759 self:raiseDirtyFlags(spec.dirtyFlag)
760 spec.workedHectarsSent = spec.workedHectars
761 end
762 end
763 end
```

getIsThreshingAllowed

Description

Returns if threshing is allowed

Definition

getIsThreshingAllowed(boolean earlyWarning)

Arguments

boolean earlyWarning early warning

Return Values

boolean allows allows threshing

Code

```
859 function Combine:getIsThreshingAllowed(earlyWarning)
860 local spec = self.spec_combine
861
862 if spec.allowThreshingDuringRain then
863 return true
864 end
865
866 local rainScale =
    g_currentMission.environment.weather:getRainFallScale()
867 local timeSinceLastRain =
    g_currentMission.environment.weather:getTimeSinceLastRain()
868 if earlyWarning ~= nil and earlyWarning == true then
869 if rainScale <= 0.02 and timeSinceLastRain > 20 then
870 return true
871 end
872 else
873 if rainScale <= 0.1 and timeSinceLastRain > 20 then
874 return true
875 end
876 end
877
878 return false
879 end
```

getCanBeTurnedOn

Description

Returns if turn on is allowed

Definition

getCanBeTurnedOn()

Return Values

boolean allow allow turn on

Code

```
917  function Combine:getCanBeTurnedOn(superFunc)
918  local spec = self.spec_combine
919
920  if spec.numAttachedCutters <= 0 then
921  return false
922  end
923
924  for cutter, _ in pairs(spec.attachedCutters) do
925  if cutter ~= self then
926  if cutter.getCanBeTurnedOn ~= nil and not
     cutter:getCanBeTurnedOn() then
927  return false
928  end
929  end
930  end
931
932  return superFunc(self)
933  end
```

getTurnedOnNotAllowedWarning

Description

Returns turn on not allowed warning text

Definition

getTurnedOnNotAllowedWarning()

Return Values

string warningText turn on not allowed warning text

Code

```
938  function Combine:getTurnedOnNotAllowedWarning(superFunc)
939  if self:getIsActiveForInput() then
940  local spec = self.spec_combine
941  if not self:getCanBeTurnedOn() then
942  if spec.numAttachedCutters == 0 then
943  return spec.noCutterWarning
944  else
```

```lua
for cutter, _ in pairs(spec.attachedCutters) do
if cutter ~= self then
if cutter.getTurnedOnNotAllowedWarning ~= nil then
local warning = cutter:getTurnedOnNotAllowedWarning()
if warning ~= nil then
return warning
end
end
end
end
end
end

return superFunc(self)
end
```

getIsFoldAllowed

Description

Returns if fold is allowed

Definition

getIsFoldAllowed()

Return Values

boolean allowsFold allows folding

Code

```lua
function Combine:getIsFoldAllowed(superFunc, direction, onAiTurnOn)
local spec = self.spec_combine

if not self.allowFoldWhileThreshing and self:getIsTurnedOn() then
return false
end

local fillLevel = self:getFillUnitFillLevel(spec.fillUnitIndex)
if direction == spec.foldDirection and (fillLevel > spec.foldFillLevelThreshold and self:getFillUnitCapacity(spec.fillUnitIndex) ~= math.huge) then
return false
end

return superFunc(self, direction, onAiTurnOn)
end
```

onDeactivate

Description

Called on deactivate

Definition

onDeactivate()

Code

```
1066  function Combine:onDeactivate()
1067  local spec = self.spec_combine
1068
1069  self:setChopperPSEnabled(false, false, true)
1070  self:setStrawPSEnabled(false, false, true)
1071  self:setCombineIsFilling(false, false, true)
1072  spec.fillEnableTime = nil
1073  spec.fillDisableTime = nil
1074  end
```

onPostAttachImplement

Description

Called on attaching a implement

Definition

onPostAttachImplement(table implement)

Arguments

table implement implement to attach

Code

```
1079  function Combine:onPostAttachImplement(attachable,
      inputJointDescIndex, jointDescIndex)
1080  local spec = self.spec_combine
1081
1082  local attacherJoint = attachable:getActiveInputAttacherJoint()
1083  if attacherJoint ~= nil then
1084  if attacherJoint.jointType == AttacherJoints.JOINTTYPE_CUTTER or
      attacherJoint.jointType ==
      AttacherJoints.JOINTTYPE_CUTTERHARVESTER then
1085  self:addCutterToCombine(attachable)
1086  end
1087  end
1088  end
```

onPostDetachImplement

Description

Called on detaching a implement

Definition

onPostDetachImplement(integer implementIndex)

Arguments

integer implementIndex index of implement to detach

Code

```
1093  function Combine:onPostDetachImplement(implementIndex)
1094  local spec = self.spec_combine
1095
1096  local object = self:getObjectFromImplementIndex(implementIndex)
1097  if object ~= nil then
1098  local attacherJoint = object:getActiveInputAttacherJoint()
1099  if attacherJoint ~= nil then
1100  if attacherJoint.jointType == AttacherJoints.JOINTTYPE_CUTTER or
      attacherJoint.jointType ==
      AttacherJoints.JOINTTYPE_CUTTERHARVESTER then
1101  self:removeCutterFromCombine(object)
1102  end
1103  end
1104  end
1105  end
```

onTurnedOn

Description

Called on turn on

Definition

onTurnedOn(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
1110  function Combine:onTurnedOn()
1111  self:startThreshing()
1112  if self.isClient then
1113  local spec = self.spec_combine
1114  g_animationManager:startAnimations(spec.animationNodes)
1115  end
1116  end
```

onTurnedOff

Description

Called on turn off

Definition

onTurnedOff(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
1121  function Combine:onTurnedOff()
1122  self:stopThreshing()
1123  if self.isClient then
```

```
1124  local spec = self.spec_combine
1125  g_animationManager:stopAnimations(spec.animationNodes)
1126  end
1127  end
```

ConveyorBelt

Description

Class for conveyor belts to control unloading start time and unloading effects

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17  function ConveyorBelt.prerequisitesPresent(specializations)
18  return SpecializationUtil.hasSpecialization(Dischargeable,
    specializations)
19  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
41  function ConveyorBelt:onLoad(savegame)
42
43  local spec = self.spec_conveyorBelt
44
45  if self.isClient then
46  spec.animationNodes =
    g_animationManager:loadAnimations(self.xmlFile,
    "vehicle.conveyorBelt.animationNodes", self.components, self,
    self.i3dMappings)
47  end
48
49  spec.effects = g_effectManager:loadEffect(self.xmlFile,
    "vehicle.conveyorBelt.effects", self.components, self,
    self.i3dMappings)
```

```
50  spec.currentDelay = 0
51  table.sort(spec.effects, function(effect1, effect2) return
    effect1.startDelay < effect2.startDelay end)
52
53  for _, effect in pairs(spec.effects) do
54  if effect.planeFadeTime ~= nil then
55  spec.currentDelay = spec.currentDelay + effect.planeFadeTime
56  end
57  if effect.setScrollUpdate ~= nil then
58  effect:setScrollUpdate(false)
59  end
60  end
61  spec.maxDelay = spec.currentDelay
62
63  spec.morphStartPos = 0
64  spec.morphEndPos = 0
65  spec.isEffectDirty = false
66  spec.emptyFactor = 1
67
68  spec.dischargeNodeIndex = Utils.getNoNil(getXMLInt(self.xmlFile,
    "vehicle.conveyorBelt#dischargeNodeIndex"), 1)
69  self:setCurrentDischargeNodeIndex(spec.dischargeNodeIndex)
70  local dischargeNode =
    self:getDischargeNodeByIndex(spec.dischargeNodeIndex)
71  local capacity =
    self:getFillUnitCapacity(dischargeNode.fillUnitIndex)
72  spec.fillUnitIndex = dischargeNode.fillUnitIndex
73  spec.startFillLevel = capacity *
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.conveyorBelt#startPercentage"), 0.9)
74
75  local i = 0
76  while true do
77  local key = string.format("vehicle.conveyorBelt.offset(%d)", i)
78  if not hasXMLProperty(self.xmlFile, key) then
79  break
80  end
81
82  local movingToolNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, key.."#movingToolNode"),
    self.i3dMappings)
83  if movingToolNode ~= nil then
```

```
84  if spec.offsets == nil then
85  spec.offsets = {}
86  end
87
88  local offset = {}
89  offset.lastState = 0
90  offset.movingToolNode = movingToolNode
91  offset.effects = {}
92  local j = 0
93  while true do
94  local effectKey = string.format(key..".effect(%d)", j)
95  if not hasXMLProperty(self.xmlFile, effectKey) then
96  break
97  end
98
99  local effectIndex = Utils.getNoNil(getXMLInt(self.xmlFile,
    effectKey.."#index"), 0)
100 local effect = spec.effects[effectIndex]
101 if effect ~= nil and effect.setOffset ~= nil then
102 local entry = {}
103 entry.effect = effect
104 entry.minValue = Utils.getNoNil(getXMLFloat(self.xmlFile,
    effectKey.."#minOffset"), 0)*1000
105 entry.maxValue = Utils.getNoNil(getXMLFloat(self.xmlFile,
    effectKey.."#maxOffset"), 1)*1000
106 entry.inverted = Utils.getNoNil(getXMLBool(self.xmlFile,
    effectKey.."#inverted"), false)
107 table.insert(offset.effects, entry)
108 else
109 g_logManager:xmlWarning(self.configFileName, "Effect index '%d'
    not found!", effectIndex)
110 end
111 j = j + 1
112 end
113
114 table.insert(spec.offsets, offset)
115 else
116 g_logManager:xmlWarning(self.configFileName, "Missing
    movingToolNode for conveyor offset '%s'!", key)
117 end
118 i = i + 1
119 end
```

```
120 end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
155 function ConveyorBelt:onDelete()
156 local spec = self.spec_conveyorBelt
157 g_effectManager:deleteEffects(spec.effects)
158 g_animationManager:deleteAnimations(spec.animationNodes)
159 end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
166 function ConveyorBelt:onUpdateTick(dt, isActiveForInput,
    isSelected)
167 if self.isClient then
168 local spec = self.spec_conveyorBelt
169
170 local isBeltActive = self:getDischargeState() ~=
    Dischargeable.DISCHARGE_STATE_OFF
171 if isBeltActive then
172 local fillLevel = self:getFillUnitFillLevel(spec.fillUnitIndex)
173 if fillLevel > 0 then
174 local movedFactor = dt / spec.currentDelay
175 spec.morphStartPos = MathUtil.clamp(spec.morphStartPos +
    movedFactor, 0, 1)
176 spec.morphEndPos = MathUtil.clamp(spec.morphEndPos + movedFactor,
    0, 1)
177
178 -- we calculate the empty factor based on visual effect mesh and
    filllevel to get a smooth transition
179 local fillFactor =
    fillLevel/self:getFillUnitCapacity(spec.fillUnitIndex)
180 local visualFactor = spec.morphEndPos-spec.morphStartPos
```

```
181
182 spec.emptyFactor = 1
183 if visualFactor > fillFactor then
184 spec.emptyFactor = MathUtil.clamp(fillFactor/visualFactor, 0, 1)
185 else
186 local offset = fillFactor - visualFactor
187 spec.offset = offset
188 spec.morphStartPos = MathUtil.clamp(spec.morphStartPos -
    (offset/((1-spec.morphStartPos)*spec.currentDelay))*dt , 0, 1)
189 end
190
191 spec.isEffectDirty = true
192 end
193 end
194
195 if spec.isEffectDirty then
196 for _, effect in pairs(spec.effects) do
197 if effect.setMorphPosition ~= nil then
198 local effectStart = effect.startDelay/spec.currentDelay
199 local effectEnd = (effect.startDelay+effect.planeFadeTime-
    effect.offset)/spec.currentDelay
200 local offsetFactor = effect.offset/effect.planeFadeTime
201
202 local startMorphFactor = (spec.morphStartPos-
    effectStart)/(effectEnd-effectStart)
203 local startMorph = MathUtil.clamp(offsetFactor +
    startMorphFactor*(1-offsetFactor), offsetFactor, 1)
204
205 local endMorphFactor = (spec.morphEndPos-effectStart)/(effectEnd-
    effectStart)
206 local endMorph = MathUtil.clamp(offsetFactor + endMorphFactor*(1-
    offsetFactor), offsetFactor, 1)
207
208 --renderText(0.6, 0.8-i*0.015, 0.012, string.format("%d:
    effectStart %.4f effectEnd %.4f -> startMorph %.4f endMorph %.4f
    | offset %.4f | %.4f %.4f", i, effectStart, effectEnd,
    startMorph, endMorph, effect.offset/effect.planeFadeTime,
    (spec.morphStartPos-effectStart)/(effectEnd-effectStart),
    (spec.morphEndPos-effectStart)/(effectEnd-effectStart)))
209 effect:setMorphPosition(startMorph, endMorph)
210 end
211 end
212 spec.isEffectDirty = false
```

```
213 end
214 end
215 end
```

CrabSteering

Description

Class for vehicles with variable steering modes

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
18 function CrabSteering.prerequisitesPresent(specializations)
19 return SpecializationUtil.hasSpecialization(Drivable,
   specializations)
20 and SpecializationUtil.hasSpecialization(Wheels, specializations)
21 and SpecializationUtil.hasSpecialization(AnimatedVehicle,
   specializations)
22 end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
48 function CrabSteering:onLoad(savegame)
49 local spec = self.spec_crabSteering
50
51 spec.state = 1
52 spec.stateMax = -1
53
54 spec.distFromCompJointToCenterOfBackWheels =
   getXMLFloat(self.xmlFile,
   "vehicle.crabSteering#distFromCompJointToCenterOfBackWheels")
55 spec.aiSteeringModeIndex = Utils.getNoNil(getXMLInt(self.xmlFile,
   "vehicle.crabSteering#aiSteeringModeIndex"), 1)
```

```
56  spec.toggleSpeedFactor = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.crabSteering#toggleSpeedFactor"), 1)
57
58  spec.currentArticulatedAxisOffset = 0
59  spec.articulatedAxisOffsetChanged = false
60  spec.articulatedAxisLastAngle = 0
61  spec.articulatedAxisChangingTime = 0
62
63  local baseKey = "vehicle.crabSteering"
64
65  spec.steeringModes = {}
66  local i = 0
67  while true do
68  local key = string.format("%s.steeringMode(%d)", baseKey, i)
69  if not hasXMLProperty(self.xmlFile, key) then
70  break
71  end
72
73  local entry = {}
74  entry.name = g_i18n:getText(getXMLString(self.xmlFile, key ..
    "#name"), self.customEnvironment)
75
76  local inputBindingName = getXMLString(self.xmlFile, key ..
    "#inputBindingName")
77  if inputBindingName ~= nil then
78  if InputAction[inputBindingName] ~= nil then
79  entry.inputAction = InputAction[inputBindingName]
80  else
81  g_logManager:xmlWarning(self.configFileName, "Invalid
    inputBindingname '%s' for '%s'", tostring(inputBindingName), key)
82  end
83  end
84
85  entry.wheels = {}
86  local j = 0
87  while true do
88  local wheelKey = string.format("%s.wheel(%d)", key, j)
89  if not hasXMLProperty(self.xmlFile, wheelKey) then
90  break
91  end
92  local wheelEntry = {}
```

```
93  wheelEntry.wheelIndex = getXMLInt(self.xmlFile, wheelKey ..
    "#index")
94  wheelEntry.offset = math.rad(
    Utils.getNoNil(getXMLFloat(self.xmlFile, wheelKey .. "#offset"),
    0) )
95  wheelEntry.locked = Utils.getNoNil(getXMLBool(self.xmlFile,
    wheelKey .. "#locked"), false)
96
97  local wheels = self:getWheels()
98  if wheels[wheelEntry.wheelIndex] ~= nil then
99  wheels[wheelEntry.wheelIndex].steeringOffset = 0
100
101 wheels[wheelEntry.wheelIndex].rotSpeedBackUp =
    wheels[wheelEntry.wheelIndex].rotSpeed
102 else
103 g_logManager:xmlError(self.configFileName, "Invalid wheelIndex
    '%s' for '%s'", tostring(wheelEntry.wheelIndex), wheelKey)
104 end
105
106 table.insert(entry.wheels, wheelEntry)
107 j = j + 1
108 end
109
110 local specArticulatedAxis = self.spec_articulatedAxis
111 if specArticulatedAxis ~= nil and
    specArticulatedAxis.componentJoint ~= nil then
112 entry.articulatedAxis = {}
113 entry.articulatedAxis.rotSpeedBackUp =
    specArticulatedAxis.rotSpeed
114 entry.articulatedAxis.offset = math.rad(
    Utils.getNoNil(getXMLFloat(self.xmlFile, key ..
    ".articulatedAxis#offset"), 0) )
115 entry.articulatedAxis.locked =
    Utils.getNoNil(getXMLBool(self.xmlFile, key ..
    ".articulatedAxis#locked"), false)
116 entry.articulatedAxis.wheelIndices =
    {StringUtil.getVectorFromString(getXMLString(self.xmlFile, key ..
    ".articulatedAxis#wheelIndices"))}
117 end
118
119 entry.animations = {}
120 j = 0
121 while true do
122 local animKey = string.format("%s.animation(%d)", key, j)
```

```
123 if not hasXMLProperty(self.xmlFile, animKey) then
124 break
125 end
126 local animName = getXMLString(self.xmlFile, animKey .. "#name")
127 local animSpeed = Utils.getNoNil(getXMLFloat(self.xmlFile,
    animKey .. "#speed"), 1.0)
128 local stopTime = getXMLFloat(self.xmlFile, animKey ..
    "#stopTime")
129 if animName ~= nil and self:getAnimationExists(animName) then
130 table.insert(entry.animations, {animName=animName,
    animSpeed=animSpeed, stopTime=stopTime})
131 else
132 g_logManager:xmlWarning(self.configFileName, "Invalid animation
    '%s' for '%s'", tostring(animName), animKey)
133 end
134 j = j + 1
135 end
136
137 table.insert(spec.steeringModes, entry)
138 i = i + 1
139 end
140
141 spec.stateMax = table.getn(spec.steeringModes)
142 if spec.stateMax > ((2^CrabSteering.STEERING_SEND_NUM_BITS) - 1)
    then
143 g_logManager:xmlError(self.configFileName, "CrabSteering only
    supports %d steering modes!",
    (2^CrabSteering.STEERING_SEND_NUM_BITS) - 1)
144 end
145
146 if spec.stateMax > 0 then
147 self:setCrabSteering(1, true)
148 end
149 end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```lua
function CrabSteering:onReadStream(streamId, connection)
local spec = self.spec_crabSteering
if spec.stateMax == 0 then
return
end
spec.state = streamReadUIntN(streamId, CrabSteering.STEERING_SEND_NUM_BITS)
end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId
integer connection connection

Code

```lua
function CrabSteering:onWriteStream(streamId, connection)
local spec = self.spec_crabSteering
if spec.stateMax == 0 then
return
end
streamWriteUIntN(streamId, spec.state, CrabSteering.STEERING_SEND_NUM_BITS)
end
```

onReadUpdateStream

Description

Called on on update

Definition

onReadUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId stream ID
integer timestamp timestamp
table connection connection

Code

```lua
function CrabSteering:onReadUpdateStream(streamId, timestamp, connection)
local spec = self.spec_crabSteering
if spec.stateMax == 0 then
return
end

```

```
186 local specArticulatedAxis = self.spec_articulatedAxis
187 if specArticulatedAxis ~= nil and
    specArticulatedAxis.componentJoint ~= nil then
188 specArticulatedAxis.curRot = streamReadFloat32(streamId)
189 end
190 end
```

onWriteUpdateStream

Description

Called on on update

Definition

onWriteUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Code

```
197 function CrabSteering:onWriteUpdateStream(streamId, connection,
    dirtyMask)
198 local spec = self.spec_crabSteering
199 if spec.stateMax == 0 then
200 return
201 end
202
203 local specArticulatedAxis = self.spec_articulatedAxis
204 if specArticulatedAxis ~= nil and
    specArticulatedAxis.componentJoint ~= nil then
205 streamWriteFloat32(streamId, specArticulatedAxis.curRot)
206 end
207 end
```

onDraw

Description

Called on draw

Definition

onDraw(boolean isActiveForInput, boolean isSelected)

Arguments

boolean isActiveForInput	true if vehicle is active for input
boolean isSelected	true if vehicle is selected

Code

```
213 function CrabSteering:onDraw(isActiveForInput, isSelected)
214 local spec = self.spec_crabSteering
215 if spec.stateMax == 0 then
216 return
```

```
217 else
218 local mode = spec.steeringModes[spec.state]
219 g_currentMission:addExtraPrintText(string.format(g_i18n:getText("action_
    mode.name))
220 end
221 end
```

setCrabSteering

Description

Change crap steering mode

Definition

setCrabSteering(integer state, boolean noEventSend)

Arguments

integer	state	new state
boolean	noEventSend	no event send

Code

```
227 function CrabSteering:setCrabSteering(state, noEventSend)
228 local spec = self.spec_crabSteering
229
230 if noEventSend == nil or noEventSend == false then
231 if g_server ~= nil then
232 g_server:broadcastEvent(SetCrabSteeringEvent:new(self, state), nil,
    nil, self)
233 else
234 g_client:getServerConnection():sendEvent(SetCrabSteeringEvent:new(self,
    state))
235 end
236 end
237
238 if state ~= spec.state then
239 local currentMode = spec.steeringModes[spec.state]
240 if currentMode.animations ~= nil then
241 for _,anim in pairs(currentMode.animations) do
242 local curTime = self:getAnimationTime(anim.animName)
243 if anim.stopTime == nil then
244 self:playAnimation(anim.animName, -anim.animSpeed, curTime,
    noEventSend)
245 end
246 end
247 end
248 local newMode = spec.steeringModes[state]
249 if newMode.animations ~= nil then
250 for _,anim in pairs(newMode.animations) do
```

```
local curTime = self:getAnimationTime(anim.animName)
if anim.stopTime ~= nil then
self:setAnimationStopTime(anim.animName, anim.stopTime)
local speed = 1.0
if curTime > anim.stopTime then
speed = -1.0
end
self:playAnimation(anim.animName, speed, curTime, noEventSend)
else
self:playAnimation(anim.animName, anim.animSpeed, curTime, noEventSend)
end
end
end
end

spec.state = state
end
```

updateSteeringAngle

Description

Update steering angle depending of the selected steering mode

Definition

updateSteeringAngle(table wheel, float dt, float steeringAngle)

Arguments

table	wheel	wheel
float	dt	time since last call in ms
float	steeringAngle	steering angle

Return Values

float steeringAngle adjusted steering angle

Code

```
function CrabSteering:updateSteeringAngle(wheel, dt, steeringAngle)
local spec = self.spec_crabSteering
local specWheels = self.spec_wheels
local specDriveable = self.spec_drivable

if spec.stateMax == 0 then
return steeringAngle
end

local currentMode = spec.steeringModes[spec.state]
if currentMode.wheels == nil or table.getn(currentMode.wheels) == 0 then
return steeringAngle
```

```
287 end
288
289 for _, wheelProperties in pairs(currentMode.wheels) do
290 if specWheels.wheels[wheelProperties.wheelIndex] == wheel then
291 local rotScale =
    math.min(1.0/(self.lastSpeed*specDriveable.speedRotScale+specDriveable.s
    1)
292 local delta = dt*0.001*self.autoRotateBackSpeed*rotScale * spec.toggleSp
293
294 if wheel.steeringOffset < wheelProperties.offset then
295 wheel.steeringOffset = math.min(wheelProperties.offset, wheel.steeringOf
296 elseif wheel.steeringOffset > wheelProperties.offset then
297 wheel.steeringOffset = math.max(wheelProperties.offset, wheel.steeringOf
298 end
299
300 if not wheelProperties.locked then
301
302 local rotSpeed
303 if self.rotatedTime > 0 then
304 rotSpeed = (wheel.rotMax - wheel.steeringOffset) / self.wheelSteeringDur
305 if wheel.rotSpeedBackUp < 0 then
306 rotSpeed = (wheel.rotMin - wheel.steeringOffset) / self.wheelSteeringDur
307 end
308 else
309 rotSpeed = -(wheel.rotMin - wheel.steeringOffset) / self.wheelSteeringDu
310 if wheel.rotSpeedBackUp < 0 then
311 rotSpeed = -(wheel.rotMax - wheel.steeringOffset) / self.wheelSteeringDu
312 end
313 end
314
315 if wheel.rotSpeed < wheel.rotSpeedBackUp then
316 wheel.rotSpeed = math.min(wheel.rotSpeedBackUp, wheel.rotSpeed + delta)
317 elseif wheel.rotSpeed > wheel.rotSpeedBackUp then
318 wheel.rotSpeed = math.max(wheel.rotSpeedBackUp, wheel.rotSpeed - delta)
319 end
320 local f = wheel.rotSpeed / wheel.rotSpeedBackUp
321
322 steeringAngle = wheel.steeringOffset + (self.rotatedTime * f * rotSpeed)
323
324 else
325
```

```
326 if wheel.steeringAngle > wheel.steeringOffset or steeringAngle > wheel.s
327 steeringAngle = math.max(wheel.steeringOffset, math.min(wheel.steeringAn
    - delta)
328 elseif wheel.steeringAngle < wheel.steeringOffset or steeringAngle < whe
    then
329 steeringAngle = math.min(wheel.steeringOffset, math.max(wheel.steeringAn
    + delta)
330 end
331
332 if steeringAngle == wheel.steeringOffset then
333 wheel.rotSpeed = 0
334 else
335 if wheel.rotSpeed < 0 then
336 wheel.rotSpeed = math.min(0, wheel.rotSpeed + delta)
337 elseif wheel.rotSpeed > 0 then
338 wheel.rotSpeed = math.max(0, wheel.rotSpeed - delta)
339 end
340 end
341
342 end
343
344 steeringAngle = MathUtil.clamp(steeringAngle, wheel.rotMin, wheel.rotMax
345
346 break
347 end
348 end
349
350 return steeringAngle
351 end
```

updateArticulatedAxisRotation

Description

Update articulated axis rotation

Definition

updateArticulatedAxisRotation(float steeringAngle, float dt)

Arguments

float steeringAngle	steering angle
float dt	time since last call in ms

Return Values

float steeringAngle adjusted steering angle

Code

```
358 function CrabSteering:updateArticulatedAxisRotation(steeringAngle, dt)
359 local spec = self.spec_crabSteering
```

```
360 local specArticulatedAxis = self.spec_articulatedAxis
361 local specDriveable = self.spec_drivable
362
363 if spec.stateMax == 0 then
364 return steeringAngle
365 end
366
367 if not self.isServer then
368 return specArticulatedAxis.curRot
369 end
370
371 local currentMode = spec.steeringModes[spec.state]
372 if currentMode.articulatedAxis == nil then
373 return steeringAngle
374 end
375
376 --
377 local rotScale =
    math.min(1.0/(self.lastSpeed*specDriveable.speedRotScale+specDriveable.s
    1)
378 local delta = dt*0.001*self.autoRotateBackSpeed*rotScale * spec.toggleSp
379
380 if spec.currentArticulatedAxisOffset < currentMode.articulatedAxis.offse
381 spec.currentArticulatedAxisOffset = math.min(currentMode.articulatedAxis
    spec.currentArticulatedAxisOffset + delta)
382 elseif spec.currentArticulatedAxisOffset > currentMode.articulatedAxis.c
383 spec.currentArticulatedAxisOffset = math.max(currentMode.articulatedAxis
    spec.currentArticulatedAxisOffset - delta)
384 end
385
386 -- adjust rotSpeed
387 if currentMode.articulatedAxis.locked then
388 if specArticulatedAxis.rotSpeed > 0 then
389 specArticulatedAxis.rotSpeed = math.max(0, specArticulatedAxis.rotSpeed
390 elseif specArticulatedAxis.rotSpeed < 0 then
391 specArticulatedAxis.rotSpeed = math.min(0, specArticulatedAxis.rotSpeed
392 end
393 else
394 if specArticulatedAxis.rotSpeed > currentMode.articulatedAxis.rotSpeedBa
395 specArticulatedAxis.rotSpeed = math.max(currentMode.articulatedAxis.rotS
    specArticulatedAxis.rotSpeed - delta)
```

```
396 elseif specArticulatedAxis.rotSpeed < currentMode.articulatedAxis.rotSpe
397 specArticulatedAxis.rotSpeed = math.min(currentMode.articulatedAxis.rotS
    specArticulatedAxis.rotSpeed + delta)
398 end
399 end
400
401 local rotSpeed
402 if (self.rotatedTime) * (currentMode.articulatedAxis.rotSpeedBackUp) > 0
403 rotSpeed = (specArticulatedAxis.rotMax - spec.currentArticulatedAxisOffs
    self.wheelSteeringDuration
404 else
405 rotSpeed = (specArticulatedAxis.rotMin - spec.currentArticulatedAxisOffs
    self.wheelSteeringDuration
406 end
407
408 local f = math.abs(specArticulatedAxis.rotSpeed) /
    math.abs(currentMode.articulatedAxis.rotSpeedBackUp)
409 rotSpeed = rotSpeed * f
410
411 steeringAngle = spec.currentArticulatedAxisOffset + (math.abs(self.rotat
412
413 -- change rotation just if wheels are moving (so you don't have to steer
    direction while turning on crab steering)
414 if table.getn(currentMode.articulatedAxis.wheelIndices) > 0 and
    spec.distFromCompJointToCenterOfBackWheels ~= nil and self.movingDirecti
415 local wheels = self:getWheels()
416
417 local curRot = MathUtil.sign(currentMode.articulatedAxis.rotSpeedBackUp)
    specArticulatedAxis.curRot
418
419 local alpha = 0
420 local count = 0
421 for _,wheelIndex in pairs(currentMode.articulatedAxis.wheelIndices) do
422 alpha = alpha + wheels[wheelIndex].steeringAngle
423 count = count + 1
424 end
425 alpha = alpha / count
426 alpha = alpha - curRot
427
428 local v = 0
429 count = 0
430 for _,wheelIndex in pairs(currentMode.articulatedAxis.wheelIndices) do
```

```
local wheel = wheels[wheelIndex]
local axleSpeed = getWheelShapeAxleSpeed(wheel.node, wheel.wheelShape) -
if wheel.hasGroundContact then
local longSlip, _ = getWheelShapeSlip(wheel.node, wheel.wheelShape)
local fac = 1.0 - math.min(1.0, longSlip)
v = v + fac * axleSpeed * wheel.radius
count = count + 1
end
end
v = v / count
local h = v * 0.001 * dt
local g = math.sin(alpha) * h
local a = math.cos(alpha) * h
local ls = spec.distFromCompJointToCenterOfBackWheels
local beta = math.atan2(g, ls - a)

steeringAngle = MathUtil.sign(currentMode.articulatedAxis.rotSpeedBackUp

spec.articulatedAxisOffsetChanged = true
spec.articulatedAxisLastAngle = steeringAngle
else
local changingTime = spec.articulatedAxisChangingTime
if spec.articulatedAxisOffsetChanged then
changingTime = 2500
spec.articulatedAxisOffsetChanged = false
end

--smooth blending if steering change is from crab to normal
if changingTime > 0 then
local pos = changingTime / 2500
steeringAngle = steeringAngle * (1-pos) + spec.articulatedAxisLastAngle
spec.articulatedAxisChangingTime = changingTime - dt
end
end

steeringAngle = math.max(specArticulatedAxis.rotMin, math.min(specArticu
steeringAngle))

return steeringAngle
end
```

onAIImplementStart

Description

Called on start ai vehicle

Definition

onAIImplementStart()

Code

```
function CrabSteering:onAIImplementStart()
local spec = self.spec_crabSteering
if spec.stateMax > 0 then
self:setCrabSteering(spec.aiSteeringModeIndex, true)
end
end
```

Crawlers

Description

This is the specialization for crawlers

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
function Crawlers.prerequisitesPresent(specializations)
return SpecializationUtil.hasSpecialization(Wheels,
specializations)
end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
function Crawlers:onLoad(savegame)
local spec = self.spec_crawlers

local wheelConfigId = Utils.getNoNil(self.configurations["wheel"], 1)
local wheelKey =
string.format("vehicle.wheels.wheelConfigurations.wheelConfiguration(%d)"
wheelConfigId-1)
```

```
44
45  spec.crawlers = {}
46  local i = 0
47  while true do
48  local key = string.format(wheelKey..".crawlers.crawler(%d)", i)
49  if not hasXMLProperty(self.xmlFile, key) then
50  break
51  end
52
53  local crawler = {}
54  if self:loadCrawlerFromXML(self.xmlFile, key, crawler) then
55  table.insert(spec.crawlers, crawler)
56  end
57
58  i = i + 1
59  end
60  end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
64  function Crawlers:onDelete()
65  local spec = self.spec_crawlers
66  for _, crawler in pairs(spec.crawlers) do
67  if crawler.filename ~= nil then
68  g_i3DManager:releaseSharedI3DFile(crawler.filename,
    self.baseDirectory, true)
69  end
70  end
71  end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActive, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActive	true if vehicle is active
boolean	isActiveForInput	true if vehicle is active for input

boolean isSelected true if vehicle is selected

Code

```
79  function Crawlers:onUpdate(dt, isActiveForInput, isSelected)
80  local spec = self.spec_crawlers
81  for _, crawler in pairs(spec.crawlers) do
82  crawler.movedDistance = 0
83
84  if crawler.wheel ~= nil then
85  local newX, _, _ = getRotation(crawler.wheel.driveNode)
86  if crawler.lastRotation == nil then
87  crawler.lastRotation = newX
88  end
89
90  if newX - crawler.lastRotation < -math.pi then
91  crawler.lastRotation = crawler.lastRotation - 2*math.pi
92  elseif newX - crawler.lastRotation > math.pi then
93  crawler.lastRotation = crawler.lastRotation + 2*math.pi
94  end
95
96  crawler.movedDistance = crawler.wheel.radius * (newX-
    crawler.lastRotation)
97  crawler.lastRotation = newX
98  else
99  local newX, newY, newZ =
    getWorldTranslation(crawler.speedReferenceNode)
100 if crawler.lastPosition == nil then
101 crawler.lastPosition = {newX, newY, newZ}
102 end
103 local dx, dy, dz =
    worldDirectionToLocal(crawler.speedReferenceNode, newX-
    crawler.lastPosition[1], newY-crawler.lastPosition[2], newZ-
    crawler.lastPosition[3])
104 local movingDirection = 0
105 if dz > 0.0001 then
106 movingDirection = 1
107 elseif dz < -0.0001 then
108 movingDirection = -1
109 end
110 crawler.movedDistance = MathUtil.vector3Length(dx, dy, dz) *
    movingDirection
111 crawler.lastPosition[1] = newX
112 crawler.lastPosition[2] = newY
```

```
113 crawler.lastPosition[3] = newZ
114 end
115
116 for _, scrollerNode in pairs(crawler.scrollerNodes) do
117 scrollerNode.scrollPosition = (scrollerNode.scrollPosition +
    crawler.movedDistance*scrollerNode.scrollSpeed) %
    scrollerNode.scrollLength
118 if scrollerNode.shaderParameterComponent == 1 then
119 setShaderParameter(scrollerNode.node,
    scrollerNode.shaderParameterName,
    scrollerNode.scrollPosition,0,0,0, false)
120 else
121 setShaderParameter(scrollerNode.node,
    scrollerNode.shaderParameterName,
    0,scrollerNode.scrollPosition,0,0, false)
122 end
123 end
124
125 for _, rotatingPart in pairs(crawler.rotatingParts) do
126 rotate(rotatingPart.node, rotatingPart.speedScale *
    crawler.movedDistance, 0, 0)
127 end
128 end
129 end
```

loadCrawlerFromXML

Description

Load crawlers from xml

Definition

loadCrawlerFromXML(integer xmlFile)

Arguments

integer xmlFile id of xml object

Code

```
134 function Crawlers:loadCrawlerFromXML(xmlFile, key, crawler)
135 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, key.."#crawlerIndex", "Moved to external
    crawler config file") -- FS17 to FS19
136 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, key.."#length", "Moved to external crawler
    config file") -- FS17 to FS19
137 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, key.."#shaderParameterComponent", "Moved to
    external crawler config file") -- FS17 to FS19
138 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, key.."#shaderParameterName", "Moved to
    external crawler config file") -- FS17 to FS19
```

```
139 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, key.."#scrollLength", "Moved to external
    crawler config file") -- FS17 to FS19
140 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, key.."#scrollSpeed", "Moved to external
    crawler config file") -- FS17 to FS19
141 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, key.."#index", "Moved to external crawler
    config file") -- FS17 to FS19
142 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, key..".rotatingPart", "Moved to external
    crawler config file") -- FS17 to FS19
143
144 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, key.."#linkIndex", key.."#linkNode") -- FS17
    to FS19
145 local linkNode = I3DUtil.indexToObject(self.components,
    getXMLString(xmlFile, key.."#linkNode"), self.i3dMappings)
146 if linkNode == nil then
147 g_logManager:xmlWarning(self.configFileName, "Missing link node
    for crawler '%s'", key)
148 return false
149 end
150
151 local isLeft = Utils.getNoNil(getXMLBool(xmlFile, key ..
    "#isLeft"), false)
152 crawler.trackWidth = Utils.getNoNil(getXMLFloat(xmlFile,
    key.."#trackWidth"), 1)
153
154 local filename = getXMLString(xmlFile, key .. "#filename")
155 if not self:loadCrawlerFromConfigFile(crawler, filename,
    linkNode, isLeft) then
156 return false
157 end
158
159 local offset =
    StringUtil.getVectorNFromString(getXMLString(xmlFile, key ..
    "#offset"), 3)
160 if offset ~= nil then
161 setTranslation(crawler.loadedCrawler, unpack(offset))
162 end
163
164 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, key.."#speedRefWheel", key.."#wheelIndex") -
    - FS17 to FS19
```

```
165 local wheelIndex = getXMLInt(xmlFile, key.."#wheelIndex")
166 if wheelIndex ~= nil then
167 local wheels = self:getWheels()
168 if wheels[wheelIndex] ~= nil then
169 crawler.wheel = wheels[wheelIndex]
170 if not crawler.wheel.isSynchronized then
171 g_logManager:xmlWarning(self.configFileName, "Wheel for crawler
    '%s' in not synchronized! It won't rotate on the client side.",
    key)
172 end
173 end
174 end
175
176 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, key.."#speedRefNode",
    key.."#speedReferenceNode") -- FS17 to FS19
177 crawler.speedReferenceNode =
    Utils.getNoNil(I3DUtil.indexToObject(self.components,
    getXMLString(xmlFile, key.."#speedReferenceNode"),
    self.i3dMappings), linkNode)
178 crawler.movedDistance = 0
179
180 crawler.fieldDirtMultiplier = Utils.getNoNil(getXMLInt(xmlFile,
    key.."#fieldDirtMultiplier"), 150)
181 crawler.streetDirtMultiplier = Utils.getNoNil(getXMLInt(xmlFile,
    key.."#streetDirtMultiplier"), -300)
182 crawler.minDirtPercentage = Utils.getNoNil(getXMLInt(xmlFile,
    key.."#minDirtPercentage"), 0.35)
183
184 return true
185 end
```

Cultivator

Description

Class for all cultivators

initSpecialization

Description

Called on specialization initializing

Definition

initSpecialization()

Code

```
15 function Cultivator.initSpecialization()
16 g_workAreaTypeManager:addWorkAreaType("cultivator", true)
17 end
```

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
23  function Cultivator.prerequisitesPresent(specializations)
24  return SpecializationUtil.hasSpecialization(WorkArea,
    specializations)
25  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
53  function Cultivator:onLoad(savegame)
54
55  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.cultivator.directionNode#index",
    "vehicle.cultivator.directionNode#node") --FS17 to FS19
56
57  if self:getGroundReferenceNodeFromIndex(1) == nil then
58  print("Warning: No ground reference nodes in "..self.configFileName)
59  end
60
61  local spec = self.spec_cultivator
62
63  if self.isClient then
64  spec.samples = {}
65  spec.samples.work = g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.cultivator.sounds", "work", self.baseDirectory,
    self.components, 0, AudioGroup.VEHICLE, self.i3dMappings, self)
66  spec.isWorkSamplePlaying = false
67  end
68
```

```
69  spec.directionNode =
    Utils.getNoNil(I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, "vehicle.cultivator.directionNode#node"),
    self.i3dMappings), self.components[1].node)
70  spec.onlyActiveWhenLowered = Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.cultivator.onlyActiveWhenLowered#value"), true)
71  spec.isSubsoiler = Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.cultivator#isSubsoiler"), false)
72
73  if self.addAITerrainDetailRequiredRange ~= nil then
74  self:addAITerrainDetailRequiredRange(g_currentMission.plowValue,
    g_currentMission.plowValue,
    g_currentMission.terrainDetailTypeFirstChannel,
    g_currentMission.terrainDetailTypeNumChannels)
75  self:addAITerrainDetailRequiredRange(g_currentMission.sowingValue,
    g_currentMission.sowingValue,
    g_currentMission.terrainDetailTypeFirstChannel,
    g_currentMission.terrainDetailTypeNumChannels)
76  self:addAITerrainDetailRequiredRange(g_currentMission.sowingWidthValue,
    g_currentMission.sowingWidthValue,
    g_currentMission.terrainDetailTypeFirstChannel,
    g_currentMission.terrainDetailTypeNumChannels)
77  self:addAITerrainDetailRequiredRange(g_currentMission.grassValue,
    g_currentMission.grassValue,
    g_currentMission.terrainDetailTypeFirstChannel,
    g_currentMission.terrainDetailTypeNumChannels)
78  end
79
80  spec.startActivationTimeout = 2000
81  spec.startActivationTime = 0
82  spec.hasGroundContact = false
83  spec.isWorking = false
84  spec.limitToField = false
85  spec.forceLimitToField = true
86
87  spec.workAreaParameters = {}
88  spec.workAreaParameters.limitToField = spec.limitToField
89  spec.workAreaParameters.forceLimitToField = spec.forceLimitToField
90  spec.workAreaParameters.angle = 0
91  spec.workAreaParameters.lastChangedArea = 0
92  spec.workAreaParameters.lastTotalArea = 0
93  end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
97  function Cultivator:onDelete()
98  if self.isClient then
99  local spec = self.spec_cultivator
100 g_soundManager:deleteSamples(spec.samples)
101 end
102 end
```

doCheckSpeedLimit

Description

Returns if speed limit should be checked

Definition

doCheckSpeedLimit()

Return Values

boolean checkSpeedlimit check speed limit

Code

```
131 function Cultivator:doCheckSpeedLimit(superFunc)
132 return superFunc(self) or self:getIsImplementChainLowered()
133 end
```

getDoGroundManipulation

Description

Returns if tool does ground manipulation

Definition

getDoGroundManipulation()

Return Values

boolean doGroundManipulation do ground manipulation

Code

```
138 function Cultivator:getDoGroundManipulation(superFunc)
139 local spec = self.spec_cultivator
140
141 if not spec.isWorking then
142 return false
143 end
144
145 return superFunc(self)
146 end
```

getDirtMultiplier

Description

Returns current dirt multiplier

Definition

getDirtMultiplier()

Return Values

float dirtMultiplier current dirt multiplier

Code

```
151  function Cultivator:getDirtMultiplier(superFunc)
152  local spec = self.spec_cultivator
153
154  local multiplier = superFunc(self)
155  if spec.isWorking then
156  multiplier = multiplier + self:getWorkDirtMultiplier() *
     self:getLastSpeed() / spec.speedLimit
157  end
158
159  return multiplier
160  end
```

getWearMultiplier

Description

Returns current wear multiplier

Definition

getWearMultiplier()

Return Values

float dirtMultiplier current wear multiplier

Code

```
165  function Cultivator:getWearMultiplier(superFunc)
166  local spec = self.spec_cultivator
167  local multiplier = superFunc(self)
168
169  if spec.isWorking then
170  multiplier = multiplier + self:getWorkWearMultiplier() *
     self:getLastSpeed() / spec.speedLimit
171  end
172
173  return multiplier
174  end
```

loadWorkAreaFromXML

Description

Loads work areas from xml

Definition

loadWorkAreaFromXML(table workArea, integer xmlFile, string key)

Arguments

table	workArea	workArea
integer	xmlFile	id of xml object
string	key	key

Return Values

boolean success success

Code

```
182  function Cultivator:loadWorkAreaFromXML(superFunc, workArea,
     xmlFile, key)
183  local retValue = superFunc(self, workArea, xmlFile, key)
184
185  if workArea.type == WorkAreaType.DEFAULT then
186  workArea.type = WorkAreaType.CULTIVATOR
187  end
188
189  return retValue
190  end
```

onPostAttach

Description

Called if vehicle gets attached

Definition

onPostAttach(table attacherVehicle, integer inputJointDescIndex, integer jointDescIndex)

Arguments

table	attacherVehicle	attacher vehicle
integer	inputJointDescIndex	index of input attacher joint
integer	jointDescIndex	index of attacher joint it gets attached to

Code

```
215  function Cultivator:onPostAttach(attacherVehicle,
     inputJointDescIndex, jointDescIndex)
216  local spec = self.spec_cultivator
217  spec.startActivationTime = g_currentMission.time +
     spec.startActivationTimeout
218  end
```

onPreDetach

Description

Called if vehicle gets detached

Definition

onPreDetach(table attacherVehicle, table implement)

Arguments

table	attacherVehicle	attacher vehicle
table	implement	implement

Code

```
224  function Cultivator:onPreDetach(attacherVehicle, implement)
225  local spec = self.spec_cultivator
226  spec.limitToField = false
227  end
```

getDefaultSpeedLimit

Description

Returns default speed limit

Definition

getDefaultSpeedLimit()

Return Values

float speedLimit speed limit

Code

```
293  function Cultivator.getDefaultSpeedLimit()
294  return 15
295  end
```

Cutter

Description

onPreDetach

Description

Called if vehicle gets detached

Definition

onPreDetach(table attacherVehicle, table implement)

Arguments

table attacherVehicle attacher vehicle

table implement implement

Code

```
732  function Cutter:onPreDetach(attacherVehicle, implement)
733  if self.isClient then
734  Cutter.updateExtraObjects(self, true)
735  end
736  end
```

Cylindered

Description

This is the specialization for vehicles which have movable parts

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
19  function Cylindered.prerequisitesPresent(specializations)
20  return true
21  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
90  function Cylindered:onLoad(savegame)
91  local spec = self.spec_cylindered
92
93  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.movingParts", "vehicle.cylindered.movingParts") --FS17 to FS19
94  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.movingTools", "vehicle.cylindered.movingTools") --FS17 to FS19
95  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.cylinderedHydraulicSound", "vehicle.cylindered.sounds.hydraulic
    FS17 to FS19
96
97  spec.samples = {}
98  if self.isClient then
99  spec.samples.hydraulic = g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.cylindered.sounds", "hydraulic", self.baseDirectory, self.compo
    0, AudioGroup.VEHICLE, self.i3dMappings, self)
100 spec.isHydraulicSamplePlaying = false
101 end
102
103 spec.activeDirtyMovingParts = {}
104
105 local referenceNodes = {}
106 spec.nodesToMovingParts = {}
107 spec.movingParts = {}
108 self.anyMovingPartsDirty = false
109 spec.detachLockNodes = nil
110 local i = 0
111 while true do
112 local partKey = string.format("vehicle.cylindered.movingParts.movingPart
    i)
113 if not hasXMLProperty(self.xmlFile, partKey) then
114 break
115 end
116
117 local entry = {}
```

```
118 if self:loadMovingPartFromXML(self.xmlFile, partKey, entry) then
119 if referenceNodes[entry.node] == nil then
120 referenceNodes[entry.node] = {}
121 end
122 if spec.nodesToMovingParts[entry.node] == nil then
123 table.insert(referenceNodes[entry.node], entry)
124
125 self:loadDependentParts(self.xmlFile, partKey, entry)
126 self:loadDependentComponentJoints(self.xmlFile, partKey, entry)
127 self:loadCopyLocalDirectionParts(self.xmlFile, partKey, entry)
128 self:loadExtraDependentParts(self.xmlFile, partKey, entry)
129 self:loadDependentAnimations(self.xmlFile, partKey, entry)
130
131 entry.key = partKey
132 table.insert(spec.movingParts, entry)
133
134 if entry.isActiveDirty then
135 table.insert(spec.activeDirtyMovingParts, entry)
136 end
137
138 spec.nodesToMovingParts[entry.node] = entry
139 else
140 g_logManager:xmlWarning(self.configFileName, "Moving part with node '%s'
    already exists!", getName(entry.node))
141 end
142 end
143
144 i = i + 1
145 end
146
147 spec.isActiveDirtyTimeOffset = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.cylindered.movingParts#isActiveDirtyTimeOffset"), 0) * 1000
148 spec.isActiveDirtyTime = g_currentMission.time
149
150 -- find dependencies
151 for _, part in pairs(spec.movingParts) do
152 part.dependentParts = {}
153 for _, ref in pairs(part.dependentPartNodes) do
154 if referenceNodes[ref] ~= nil then
155 for _, p in pairs(referenceNodes[ref]) do
156 part.dependentParts[p] = p
```

```
157 p.isDependentPart = true
158 end
159 end
160 end
161 end
162
163 local function addMovingPart(part, newTable, allowDependentParts)
164 for _, addedPart in ipairs(newTable) do
165 if addedPart == part then
166 return
167 end
168 end
169
170 if part.isDependentPart == true then
171 if allowDependentParts ~= true then
172 return
173 end
174 end
175
176 table.insert(newTable, part)
177
178 for _, depPart in pairs(part.dependentParts) do
179 addMovingPart(depPart, newTable, true)
180 end
181 end
182
183 local newParts = {}
184 for _, part in ipairs(spec.movingParts) do
185 addMovingPart(part, newParts)
186 end
187 spec.movingParts = newParts
188
189 spec.controlGroups = {}
190 spec.controlGroupMapping = {}
191 spec.currentControlGroupIndex = 1
192 spec.controlGroupNames = {}
193 local i = 0
194 while true do
195 local groupKey =
    string.format("vehicle.cylindered.movingTools.controlGroups.controlGroup
    i)
```

```
196 if not hasXMLProperty(self.xmlFile, groupKey) then
197 break
198 end
199
200 local name = getXMLString(self.xmlFile, groupKey.."#name")
201 if name ~= nil then
202 table.insert(spec.controlGroupNames, g_i18n:convertText(name,
    self.customEnvironment))
203 end
204
205 i = i + 1
206 end
207
208 spec.nodesToMovingTools = {}
209 spec.movingTools = {}
210 local i = 0
211 while true do
212 local toolKey = string.format("vehicle.cylindered.movingTools.movingTool
    i)
213 if not hasXMLProperty(self.xmlFile, toolKey) then
214 break
215 end
216
217 local entry = {}
218 if self:loadMovingToolFromXML(self.xmlFile, toolKey, entry) then
219 if referenceNodes[entry.node] == nil then
220 referenceNodes[entry.node] = {}
221 end
222
223 if spec.nodesToMovingTools[entry.node] == nil then
224 table.insert(referenceNodes[entry.node], entry)
225
226 self:loadDependentMovingTools(self.xmlFile, toolKey, entry)
227 self:loadDependentParts(self.xmlFile, toolKey, entry)
228 self:loadDependentComponentJoints(self.xmlFile, toolKey, entry)
229 self:loadExtraDependentParts(self.xmlFile, toolKey, entry)
230 self:loadDependentAnimations(self.xmlFile, toolKey, entry)
231
232 entry.isActive = true
233 entry.key = toolKey
234 table.insert(spec.movingTools, entry)
```

```
235 spec.nodesToMovingTools[entry.node] = entry
236 else
237 g_logManager:xmlWarning(self.configFileName, "Moving tool with node '%s'
    already exists!", getName(entry.node))
238 end
239 end
240 i = i + 1
241 end
242
243 for _, groupIndex in ipairs(spec.controlGroups) do
244 local subSelectionIndex = self:addSubselection(groupIndex)
245 spec.controlGroupMapping[subSelectionIndex] = groupIndex
246 end
247
248 for _, part in pairs(spec.movingTools) do
249 part.dependentParts = {}
250 for _, ref in pairs(part.dependentPartNodes) do
251 if referenceNodes[ref] ~= nil then
252 for _, p in pairs(referenceNodes[ref]) do
253 part.dependentParts[p] = p
254 end
255 end
256 end
257 for i=#part.dependentMovingTools, 1, -1 do
258 local dependentTool = part.dependentMovingTools[i]
259 local tool = spec.nodesToMovingTools[dependentTool.node]
260 if tool ~= nil then
261 dependentTool.movingTool = tool
262 else
263 g_logManager:xmlWarning(self.configFileName, "Dependent moving tool '%s'
    defined. Ignoring it!", getName(dependentTool.node))
264 table.remove(part.dependentMovingTools, i)
265 end
266 end
267 end
268
269
270 local simpleKey = "vehicle.cylindered.movingTools.easyArmControl"
271 if hasXMLProperty(self.xmlFile, simpleKey) then
272 spec.easyArmControl = {}
```

```
spec.easyArmControl.rootNode = I3DUtil.indexToObject(self.components, getXMLString(self.xmlFile, simpleKey.."#rootNode"), self.i3dMappings)
spec.easyArmControl.targetNodeY = I3DUtil.indexToObject(self.components, getXMLString(self.xmlFile, simpleKey.."#node"), self.i3dMappings)
spec.easyArmControl.targetNodeZ = I3DUtil.indexToObject(self.components, getXMLString(self.xmlFile, simpleKey.."#targetNodeZ"), self.i3dMappings)
spec.easyArmControl.targetNodeY

if spec.easyArmControl.targetNodeZ ~= nil and spec.easyArmControl.target ~= nil then
local targetYTool = self:getMovingToolByNode(spec.easyArmControl.targetN
local targetZTool = self:getMovingToolByNode(spec.easyArmControl.targetN
if targetYTool ~= nil and targetZTool ~= nil then
spec.easyArmControl.targetNode = spec.easyArmControl.targetNodeZ
if getParent(spec.easyArmControl.targetNodeY) == spec.easyArmControl.targetNodeZ then
spec.easyArmControl.targetNode = spec.easyArmControl.targetNodeY
end
spec.easyArmControl.targetRefNode = I3DUtil.indexToObject(self.component getXMLString(self.xmlFile, simpleKey.."#refNode"), self.i3dMappings)
spec.easyArmControl.lastValidPositionY = {getTranslation(spec.easyArmControl.targetNodeY)}
spec.easyArmControl.lastValidPositionZ = {getTranslation(spec.easyArmControl.targetNodeZ)}
spec.easyArmControl.xRotationMaxDistance = getXMLFloat(self.xmlFile, simpleKey..".xRotationNodes#maxDistance") or 0
spec.easyArmControl.xRotationNodes = {}
spec.easyArmControl.zTranslationNodes = {}

i = 0
local maxTrans = 0
while true do
local transKey = string.format("%s.zTranslationNodes.zTranslationNode(%d simpleKey, i)
if not hasXMLProperty(self.xmlFile, transKey) then
break
end

local node = I3DUtil.indexToObject(self.components, getXMLString(self.xm transKey.."#node"), self.i3dMappings)
if node ~= nil then
local movingTool = self:getMovingToolByNode(node)
if movingTool ~= nil then
```

```
304 local maxDistance = math.abs(movingTool.transMin - movingTool.transMax)
305 maxTrans = maxTrans + maxDistance
306 movingTool.easyArmControlActive = false
307 table.insert(spec.easyArmControl.zTranslationNodes, {node=node,
    movingTool=movingTool, maxDistance=maxDistance, transFactor=0})
308 end
309 end
310
311 i = i + 1
312 end
313
314 for _, translationNode in ipairs(spec.easyArmControl.zTranslationNodes)
315 translationNode.transFactor = translationNode.maxDistance / maxTrans
316 end
317
318 for i=1, 2 do
319 local xRotKey = string.format("%s.xRotationNodes.xRotationNode%d", simpl
    i)
320 if not hasXMLProperty(self.xmlFile, xRotKey) then
321 g_logManager:xmlWarning(self.configFileName, "Missing second xRotation n
    for easy control!")
322 spec.easyArmControl = nil
323 break
324 end
325
326 local node = I3DUtil.indexToObject(self.components, getXMLString(self.xm
    xRotKey.."#node"), self.i3dMappings)
327 local refNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, xRotKey.."#refNode"), self.i3dMappings)
328 if node ~= nil and refNode ~= nil then
329 local movingTool = self:getMovingToolByNode(node)
330 if movingTool ~= nil then
331 movingTool.easyArmControlActive = false
332 table.insert(spec.easyArmControl.xRotationNodes, {node=node, refNode=ref
    movingTool=movingTool})
333 end
334 end
335 end
336 else
337 g_logManager:xmlError(self.configFileName, "Missing moving tools for eas
    control targets!")
338 spec.easyArmControl = nil
```

```
339 end
340 else
341 g_logManager:xmlError(self.configFileName, "Missing easy control targets
342 spec.easyArmControl = nil
343 end
344 end
345
346 if self.loadDashboardsFromXML ~= nil then
347 local dashboardData = {valueTypeToLoad = "movingTool",
348 valueObject = self,
349 valueFunc = Cylindered.getMovingToolDashboardState,
350 minFunc = 0,
351 maxFunc = 1,
352 additionalAttributesFunc = Cylindered.movingToolDashboardAttributes,
353 idleValue = 0.5}
354
355 self:loadDashboardsFromXML(self.xmlFile, "vehicle.cylindered.dashboards"
    dashboardData)
356 end
357
358 if self.isClient and g_isDevelopmentVersion then
359 if (#spec.movingParts > 0 or #spec.movingTools > 0) and spec.samples.hyd
    == nil then
360 g_logManager:xmlDevWarning(self.configFileName, "Missing cylindered hydr
    sound")
361 end
362 end
363
364 spec.cylinderedDirtyFlag = self:getNextDirtyFlag()
365 spec.cylinderedInputDirtyFlag = self:getNextDirtyFlag()
366
367 spec.isLoading = true
368 end
```

onPostLoad

Description

Called after loading

Definition

onPostLoad(table savegame)

Arguments

table savegame savegame

Code

```
373 function Cylindered:onPostLoad(savegame)
374 local spec = self.spec_cylindered
375
376 for _, tool in pairs(spec.movingTools) do
377 if self:getIsMovingToolActive(tool) then
378 if tool.startRot ~= nil then
379 tool.curRot[tool.rotationAxis] = tool.startRot
380 setRotation(tool.node, unpack(tool.curRot))
381 end
382 if tool.startTrans ~= nil then
383 tool.curTrans[tool.translationAxis] = tool.startTrans
384 setTranslation(tool.node, unpack(tool.curTrans))
385 end
386
387 if tool.delayedNode ~= nil then
388 self:setDelayedData(tool, true)
389 end
390
391 if tool.isIntitialDirty then
392 Cylindered.setDirty(self, tool)
393 end
394 end
395 end
396
397 for _, part in pairs(spec.movingParts) do
398 self:loadDependentAttacherJoints(self.xmlFile, part.key, part)
399 self:loadDependentWheels(self.xmlFile, part.key, part)
400 end
401
402 for _, tool in pairs(spec.movingTools) do
403 self:loadDependentAttacherJoints(self.xmlFile, tool.key, tool)
404 self:loadDependentWheels(self.xmlFile, tool.key, tool)
405 end
406
407 if self:allowLoadMovingToolStates() then
408 if savegame ~= nil and not savegame.resetVehicles then
409 local i = 0
410 for _, tool in ipairs(spec.movingTools) do
411 if tool.saving then
412 if self:getIsMovingToolActive(tool) then
```

```
local toolKey = string.format("%s.cylindered.movingTool(%d)", savegame.key, i)
local changed = false
if tool.transSpeed ~= nil then
local newTrans = getXMLFloat(savegame.xmlFile, toolKey.."#translation")
if newTrans ~= nil then
if tool.transMax ~= nil then
newTrans = math.min(newTrans, tool.transMax)
end
if tool.transMin ~= nil then
newTrans = math.max(newTrans, tool.transMin)
end
end
if newTrans ~= nil and math.abs(newTrans - tool.curTrans[tool.translationAxis]) > 0.0001 then
tool.curTrans = {getTranslation(tool.node)}
tool.curTrans[tool.translationAxis] = newTrans
setTranslation(tool.node, unpack(tool.curTrans))
changed = true
end
end
if tool.rotSpeed ~= nil then
local newRot = getXMLFloat(savegame.xmlFile, toolKey.."#rotation")
if newRot ~= nil then
if tool.rotMax ~= nil then
newRot = math.min(newRot, tool.rotMax)
end
if tool.rotMin ~= nil then
newRot = math.max(newRot, tool.rotMin)
end
end
if newRot ~= nil and math.abs(newRot - tool.curRot[tool.rotationAxis]) > 0.0001 then
tool.curRot = {getRotation(tool.node)}
tool.curRot[tool.rotationAxis] = newRot
setRotation(tool.node, unpack(tool.curRot))
changed = true
end
end
```

```
449 if tool.animSpeed ~= nil then
450 local animTime = getXMLFloat(savegame.xmlFile,
    toolKey.."#animationTime")
451 if animTime ~= nil then
452 if tool.animMinTime ~= nil then
453 animTime = math.max(animTime, tool.animMinTime)
454 end
455 if tool.animMaxTime ~= nil then
456 animTime = math.min(animTime, tool.animMaxTime)
457 end
458 end
459 tool.curAnimTime = animTime
460 self:setAnimationTime(tool.animName, animTime, true)
461 end
462 if changed then
463 Cylindered.setDirty(self, tool)
464 end
465
466 if tool.delayedNode ~= nil then
467 self:setDelayedData(tool, true)
468 end
469 end
470 i = i + 1
471 end
472
473 for _, dependentTool in pairs(tool.dependentMovingTools) do
474 Cylindered.updateRotationBasedLimits(self, tool, dependentTool)
475 end
476 end
477 end
478 end
479
480 self:updateEasyControl(9999, true)
481 self:updateCylinderedInitial(false)
482
483 -- force update after loading, e.g. savegame
484 spec.isActiveDirtyTime = g_currentMission.time +
    math.max(spec.isActiveDirtyTimeOffset, 1000)
485 end
```

onLoadFinished

Description

Called after loading

Definition

onLoadFinished(table savegame)

Arguments

table savegame savegame

Code

```
490 function Cylindered:onLoadFinished(savegame)
491 local spec = self.spec_cylindered
492 spec.isLoading = false
493
494 for i=1, table.getn(spec.movingTools) do
495 local tool = spec.movingTools[i]
496 if tool.delayedHistoryIndex ~= nil and tool.delayedHistoryIndex > 0 then
497 self:updateDelayedTool(tool, true)
498 end
499 end
500 end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
504 function Cylindered:onDelete()
505 local spec = self.spec_cylindered
506
507 if self.isClient then
508 g_soundManager:deleteSamples(spec.samples)
509 end
510
511 for _,movingTool in pairs(spec.movingTools) do
512 if movingTool.icon ~= nil then
513 movingTool.icon:delete()
514 movingTool.icon = nil
515 end
516 end
517 end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
546 function Cylindered:onReadStream(streamId, connection)
547 local spec = self.spec_cylindered
548
549 if self:allowLoadMovingToolStates() then
550 if connection:getIsServer() then
551 for i=1, table.getn(spec.movingTools) do
552 local tool = spec.movingTools[i]
553 tool.networkTimeInterpolator:reset()
554 if tool.transSpeed ~= nil then
555 local newTrans = streamReadFloat32(streamId)
556 tool.curTrans[tool.translationAxis] = newTrans
557 setTranslation(tool.node, unpack(tool.curTrans))
558 tool.networkInterpolators.translation:setValue(tool.curTrans[tool.transl
559 end
560 if tool.rotSpeed ~= nil then
561 local newRot = streamReadFloat32(streamId)
562 tool.curRot[tool.rotationAxis] = newRot
563 setRotation(tool.node, unpack(tool.curRot))
564 tool.networkInterpolators.rotation:setAngle(newRot)
565 end
566 if tool.animSpeed ~= nil then
567 local newAnimTime = streamReadFloat32(streamId)
568 tool.curAnimTime = newAnimTime
569 self:setAnimationTime(tool.animName, tool.curAnimTime)
570 tool.networkInterpolators.animation:setValue(newAnimTime)
571 end
572 if tool.delayedNode ~= nil then
573 self:setDelayedData(tool, true)
574 end
575 Cylindered.setDirty(self, tool)
576 end
577 end
578 end
579 end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
585 function Cylindered:onWriteStream(streamId, connection)
586 local spec = self.spec_cylindered
587
588 if self:allowLoadMovingToolStates() then
589 if not connection:getIsServer() then
590 for i=1, table.getn(spec.movingTools) do
591 local tool = spec.movingTools[i]
592 if tool.transSpeed ~= nil then
593 streamWriteFloat32(streamId, tool.curTrans[tool.translationAxis])
594 end
595 if tool.rotSpeed ~= nil then
596 streamWriteFloat32(streamId, tool.curRot[tool.rotationAxis])
597 end
598 if tool.animSpeed ~= nil then
599 streamWriteFloat32(streamId, tool.curAnimTime)
600 end
601 end
602 end
603 end
604 end
```

onReadUpdateStream

Description

Called on on update

Definition

onReadUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId stream ID

integer timestamp timestamp

table connection connection

Code

```
611 function Cylindered:onReadUpdateStream(streamId, timestamp,
    connection)
612 local spec = self.spec_cylindered
613
```

```
614 -- if server, read input from client
615 if not connection:getIsServer() then
616 if streamReadBool(streamId) then
617 for _, tool in ipairs(spec.movingTools) do
618 if tool.axisActionIndex ~= nil then
619 tool.move = (streamReadUIntN(streamId, 12) / 4095 * 2 - 1) * 5
620 if math.abs(tool.move) < 0.01 then
621 tool.move = 0
622 end
623 end
624 end
625 end
626 else
627 -- if client, read updated attributes
628 if streamReadBool(streamId) then
629 for _, tool in ipairs(spec.movingTools) do
630 if tool.dirtyFlag ~= nil then
631 if streamReadBool(streamId) then
632 tool.networkTimeInterpolator:startNewPhaseNetwork()
633
634 if tool.transSpeed ~= nil then
635 local newTrans = streamReadFloat32(streamId)
636 if math.abs(newTrans - tool.curTrans[tool.translationAxis]) >
    0.0001 then
637 tool.networkInterpolators.translation:setTargetValue(newTrans)
638 end
639 end
640 if tool.rotSpeed ~= nil then
641 local newRot
642 if tool.rotMin == nil or tool.rotMax == nil then
643 newRot = NetworkUtil.readCompressedAngle(streamId)
644 else
645 if tool.syncMinRotLimits then
646 tool.rotMin = streamReadFloat32(streamId)
647 end
648 if tool.syncMaxRotLimits then
649 tool.rotMax = streamReadFloat32(streamId)
650 end
651 newRot = NetworkUtil.readCompressedRange(streamId, tool.rotMin,
    tool.rotMax, tool.rotSendNumBits)
652 end
```

```
653 if math.abs(newRot - tool.curRot[tool.rotationAxis]) > 0.0001 then
654 tool.networkInterpolators.rotation:setTargetAngle(newRot)
655 end
656 end
657 if tool.animSpeed ~= nil then
658 local newAnimTime = NetworkUtil.readCompressedRange(streamId, tool.animMinTime, tool.animMaxTime, tool.animSendNumBits)
659 if math.abs(newAnimTime - tool.curAnimTime) > 0.0001 then
660 tool.networkInterpolators.animation:setTargetValue(newAnimTime)
661 end
662 end
663 end
664 end
665 end
666 end
667 end
668 end
```

onWriteUpdateStream

Description

Called on on update

Definition

onWriteUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Code

```
675 function Cylindered:onWriteUpdateStream(streamId, connection, dirtyMask)
676 local spec = self.spec_cylindered
677
678 -- if client, send input to server
679 if connection:getIsServer() then
680 if streamWriteBool(streamId, bitAND(dirtyMask, spec.cylinderedInputDirtyFlag) ~= 0) then
681 for _, tool in ipairs(spec.movingTools) do
682 if tool.axisActionIndex ~= nil then
683 local value = (MathUtil.clamp(tool.moveToSend / 5, -1, 1) + 1) / 2 * 4095
684 streamWriteUIntN(streamId, value, 12)
685 end
```

```
686 end
687 end
688 else
689 -- if server, send updated attributes
690 if streamWriteBool(streamId, bitAND(dirtyMask,
    spec.cylinderedDirtyFlag) ~= 0) then
691 for _, tool in ipairs(spec.movingTools) do
692 if tool.dirtyFlag ~= nil then
693 if streamWriteBool(streamId, bitAND(dirtyMask, tool.dirtyFlag) ~=
    0 and self:getIsMovingToolActive(tool)) then
694 if tool.transSpeed ~= nil then
695 streamWriteFloat32(streamId, tool.curTrans[tool.translationAxis])
696 end
697 if tool.rotSpeed ~= nil then
698 local rot = tool.curRot[tool.rotationAxis]
699 if tool.rotMin == nil or tool.rotMax == nil then
700 NetworkUtil.writeCompressedAngle(streamId, rot)
701 else
702 if tool.syncMinRotLimits then
703 streamWriteFloat32(streamId, tool.rotMin)
704 end
705 if tool.syncMaxRotLimits then
706 streamWriteFloat32(streamId, tool.rotMax)
707 end
708 NetworkUtil.writeCompressedRange(streamId, rot, tool.rotMin,
    tool.rotMax, tool.rotSendNumBits)
709 end
710 end
711 if tool.animSpeed ~= nil then
712 NetworkUtil.writeCompressedRange(streamId, tool.curAnimTime,
    tool.animMinTime, tool.animMaxTime, tool.animSendNumBits)
713 end
714 end
715 end
716 end
717 end
718 end
719 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActive, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActive	true if vehicle is active
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
727 function Cylindered:onUpdate(dt, isActiveForInput, isSelected)
728 local spec = self.spec_cylindered
729
730 spec.movingToolNeedsSound = false
731 spec.movingPartNeedsSound = false
732
733 self:updateEasyControl(dt)
734
735 if self.isServer then
736 for i=1,table.getn(spec.movingTools) do
737 local tool = spec.movingTools[i]
738
739 local rotSpeed = 0
740 local transSpeed = 0
741 local animSpeed = 0
742
743 local move = tool.move + tool.externalMove
744 if math.abs(move) > 0 then
745 tool.externalMove = 0
746
747 if tool.rotSpeed ~= nil then
748 rotSpeed = move*tool.rotSpeed
749 if tool.rotAcceleration ~= nil and math.abs(rotSpeed - tool.lastRotSpeed
    tool.rotAcceleration*dt then
750 if rotSpeed > tool.lastRotSpeed then
751 rotSpeed = tool.lastRotSpeed + tool.rotAcceleration*dt
752 else
753 rotSpeed = tool.lastRotSpeed - tool.rotAcceleration*dt
754 end
755 end
756 end
757 if tool.transSpeed ~= nil then
758 transSpeed = move*tool.transSpeed
```

```
759 if tool.transAcceleration ~= nil and math.abs(transSpeed -
    tool.lastTransSpeed) >= tool.transAcceleration*dt then
760 if transSpeed > tool.lastTransSpeed then
761 transSpeed = tool.lastTransSpeed + tool.transAcceleration*dt
762 else
763 transSpeed = tool.lastTransSpeed - tool.transAcceleration*dt
764 end
765 end
766 end
767 if tool.animSpeed ~= nil then
768 animSpeed = move*tool.animSpeed
769 if tool.animAcceleration ~= nil and math.abs(animSpeed - tool.lastAnimSp
    >= tool.animAcceleration*dt then
770 if animSpeed > tool.lastAnimSpeed then
771 animSpeed = tool.lastAnimSpeed + tool.animAcceleration*dt
772 else
773 animSpeed = tool.lastAnimSpeed - tool.animAcceleration*dt
774 end
775 end
776 end
777 else
778 -- decelerate
779 if tool.rotAcceleration ~= nil then
780 if tool.lastRotSpeed < 0 then
781 rotSpeed = math.min(tool.lastRotSpeed + tool.rotAcceleration*dt, 0)
782 else
783 rotSpeed = math.max(tool.lastRotSpeed - tool.rotAcceleration*dt, 0)
784 end
785 end
786 if tool.transAcceleration ~= nil then
787 if tool.lastTransSpeed < 0 then
788 transSpeed = math.min(tool.lastTransSpeed + tool.transAcceleration*dt, 0
789 else
790 transSpeed = math.max(tool.lastTransSpeed - tool.transAcceleration*dt, 0
791 end
792 end
793 if tool.animAcceleration ~= nil then
794 if tool.lastAnimSpeed < 0 then
795 animSpeed = math.min(tool.lastAnimSpeed + tool.animAcceleration*dt, 0)
796 else
797 animSpeed = math.max(tool.lastAnimSpeed - tool.animAcceleration*dt, 0)
```

```
798 end
799 end
800 end
801
802 local changed = false
803 if rotSpeed ~= nil and rotSpeed ~= 0 then
804 changed = changed or Cylindered.setToolRotation(self, tool, rotSpeed, dt
805 else
806 tool.lastRotSpeed = 0
807 end
808 if transSpeed ~= nil and transSpeed ~= 0 then
809 changed = changed or Cylindered.setToolTranslation(self, tool, transSpee
810 else
811 tool.lastTransSpeed = 0
812 end
813 if animSpeed ~= nil and animSpeed ~= 0 then
814 changed = changed or Cylindered.setToolAnimation(self, tool, animSpeed,
815 else
816 tool.lastAnimSpeed = 0
817 end
818
819 for _, dependentTool in pairs(tool.dependentMovingTools) do
820 if dependentTool.speedScale ~= nil then
821 local isAllowed = true
822 if dependentTool.requiresMovement then
823 if not changed then
824 isAllowed = false
825 end
826 end
827
828 if isAllowed then
829 dependentTool.movingTool.externalMove = dependentTool.speedScale * tool.
830 end
831 end
832
833 Cylindered.updateRotationBasedLimits(self, tool, dependentTool)
834
835 if dependentTool.minTransLimits ~= nil or dependentTool.maxTransLimits ~
    then
836 local state = Cylindered.getMovingToolState(self, tool)
```

```
837 if dependentTool.minTransLimits ~= nil then
838 dependentTool.movingTool.transMin =
    MathUtil.lerp(dependentTool.minTransLimits[1],
    dependentTool.minTransLimits[2], 1-state)
839 end
840 if dependentTool.maxTransLimits ~= nil then
841 dependentTool.movingTool.transMax =
    MathUtil.lerp(dependentTool.maxTransLimits[1],
    dependentTool.maxTransLimits[2], 1-state)
842 end
843 local transLimitChanged = Cylindered.setToolTranslation(self,
    dependentTool.movingTool, 0, 0)
844 if transLimitChanged then
845 Cylindered.setDirty(self, dependentTool.movingTool)
846 end
847 end
848 if dependentTool.minRotLimits ~= nil or dependentTool.maxRotLimits ~= ni
849 local state = Cylindered.getMovingToolState(self, tool)
850 if dependentTool.minRotLimits ~= nil then
851 dependentTool.movingTool.rotMin = MathUtil.lerp(dependentTool.minRotLimi
    dependentTool.minRotLimits[2], 1-state)
852 end
853 if dependentTool.maxRotLimits ~= nil then
854 dependentTool.movingTool.rotMax = MathUtil.lerp(dependentTool.maxRotLimi
    dependentTool.maxRotLimits[2], 1-state)
855 end
856
857 local rotLimitChanged = Cylindered.setToolRotation(self,
    dependentTool.movingTool, 0, 0)
858 if rotLimitChanged then
859 Cylindered.setDirty(self, dependentTool.movingTool)
860 end
861 end
862 end
863
864 if changed then
865 if tool.playSound then
866 spec.movingToolNeedsSound = true
867 end
868 Cylindered.setDirty(self, tool)
869 tool.networkPositionIsDirty = true
870 self:raiseDirtyFlags(tool.dirtyFlag)
```

```
871 self:raiseDirtyFlags(spec.cylinderedDirtyFlag)
872 end
873 end
874 else
875 -- client side
876 for i=1,table.getn(spec.movingTools) do
877 local tool = spec.movingTools[i]
878
879 tool.networkTimeInterpolator:update(dt)
880 local interpolationAlpha = tool.networkTimeInterpolator:getAlpha()
881 local changed = false
882
883 if self:getIsMovingToolActive(tool) then
884 if tool.rotSpeed ~= nil then
885 local newRot =
    tool.networkInterpolators.rotation:getInterpolatedValue(interpolationAlp
886 if math.abs(newRot - tool.curRot[tool.rotationAxis]) > 0.0001 then
887 changed = true
888 tool.curRot[tool.rotationAxis] = newRot
889 setRotation(tool.node, tool.curRot[1], tool.curRot[2], tool.curRot[3])
890 end
891 end
892
893 if tool.transSpeed ~= nil then
894 local newTrans =
    tool.networkInterpolators.translation:getInterpolatedValue(interpolation
895 if math.abs(newTrans - tool.curTrans[tool.translationAxis]) > 0.0001 the
896 changed = true
897 tool.curTrans[tool.translationAxis] = newTrans
898 setTranslation(tool.node, tool.curTrans[1], tool.curTrans[2],
    tool.curTrans[3])
899 end
900 end
901
902 if tool.animSpeed ~= nil then
903 local newAnimTime =
    tool.networkInterpolators.animation:getInterpolatedValue(interpolationAl
904 if math.abs(newAnimTime - tool.curAnimTime) > 0.0001 then
905 changed = true
906 tool.curAnimTime = newAnimTime
907 self:setAnimationTime(tool.animName, newAnimTime)
```

```
908  end
909  end
910
911  if changed then
912  Cylindered.setDirty(self, tool)
913  end
914  end
915
916  if tool.networkTimeInterpolator:isInterpolating() then
917  self:raiseActive()
918  end
919  end
920  end
921
922  for i=1, table.getn(spec.movingTools) do
923  local tool = spec.movingTools[i]
924  if tool.delayedHistoryIndex ~= nil and tool.delayedHistoryIndex > 0 then
925  self:updateDelayedTool(tool)
926  end
927  end
928  end
```

loadDependentMovingTools

Description

Load dependent moving tools from xml

Definition

loadDependentMovingTools(integer xmlFile, string baseName, table entry)

Arguments

integer	xmlFile	id of xml object
string	baseName	base name
table	entry	entry to add

Code

```
1538  function Cylindered:loadDependentMovingTools(xmlFile, baseName,
      entry)
1539  entry.dependentMovingTools = {}
1540
1541  local j = 0
1542  while true do
1543  local refBaseName =
      baseName..string.format(".dependentMovingTool(%d)", j)
1544  if not hasXMLProperty(xmlFile, refBaseName) then
1545  break
```

```
1546  end
1547
1548  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName, refBaseName.."#index",
      refBaseName.."#index") --FS17 to FS19
1549
1550  local node = I3DUtil.indexToObject(self.components,
      getXMLString(xmlFile, refBaseName.."#node"), self.i3dMappings)
1551  local speedScale = getXMLFloat(xmlFile,
      refBaseName.."#speedScale")
1552  local requiresMovement = Utils.getNoNil(getXMLBool(xmlFile,
      refBaseName.."#requiresMovement"), false)
1553
1554  local rotationBasedLimits =
      AnimCurve:new(Cylindered.limitInterpolator)
1555  local found = false
1556  local i = 0
1557  while true do
1558  local key = string.format("%s.limit(%d)",
      refBaseName..".rotationBasedLimits", i)
1559  if not hasXMLProperty(xmlFile, key) then
1560  break
1561  end
1562
1563  local keyFrame = self:loadRotationBasedLimits(xmlFile, key,
      entry)
1564  if keyFrame ~= nil then
1565  rotationBasedLimits:addKeyframe(keyFrame)
1566  found = true
1567  end
1568  i = i + 1
1569  end
1570  if not found then
1571  rotationBasedLimits = nil
1572  end
1573
1574  local minTransLimits = getXMLString(xmlFile,
      refBaseName.."#minTransLimits")
1575  local maxTransLimits = getXMLString(xmlFile,
      refBaseName.."#maxTransLimits")
1576  local minRotLimits = getXMLString(xmlFile,
      refBaseName.."#minRotLimits")
```

```
1577  local maxRotLimits = getXMLString(xmlFile, refBaseName.."#maxRotLimits")
1578  if node ~= nil and (rotationBasedLimits ~= nil or speedScale ~= nil or minTransLimits ~= nil or maxTransLimits ~= nil or minRotLimits ~= nil or maxRotLimits ~= nil) then
1579  local dependentTool = {}
1580  dependentTool.node = node
1581  dependentTool.rotationBasedLimits = rotationBasedLimits
1582  dependentTool.speedScale = speedScale
1583  dependentTool.requiresMovement = requiresMovement
1584  dependentTool.minTransLimits = StringUtil.getVectorNFromString(minTransLimits, 2)
1585  dependentTool.maxTransLimits = StringUtil.getVectorNFromString(maxTransLimits, 2)
1586  dependentTool.minRotLimits = StringUtil.getRadiansFromString(minRotLimits, 2)
1587  dependentTool.maxRotLimits = StringUtil.getRadiansFromString(maxRotLimits, 2)
1588  table.insert(entry.dependentMovingTools, dependentTool)
1589  end
1590
1591  j = j + 1
1592  end
1593  end
```

loadDependentParts

Description

Load dependent parts from xml

Definition

loadDependentParts(integer xmlFile, string baseName, table entry)

Arguments

integer	xmlFile	id of xml object
string	baseName	base name
table	entry	entry to add

Code

```
1600  function Cylindered:loadDependentParts(xmlFile, baseName, entry)
1601  entry.dependentPartNodes = {}
1602
1603  local j = 0
1604  while true do
1605  local refBaseName = baseName..string.format(".dependentPart(%d)", j)
1606  if not hasXMLProperty(xmlFile, refBaseName) then
1607  break
```

```
1608  end
1609
1610  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName, refBaseName.."#index",
      refBaseName.."#index") --FS17 to FS19
1611
1612  local node = I3DUtil.indexToObject(self.components,
      getXMLString(xmlFile, refBaseName.."#node"), self.i3dMappings)
1613  if node ~= nil then
1614  table.insert(entry.dependentPartNodes, node)
1615  end
1616
1617  j = j + 1
1618  end
1619  end
```

loadDependentComponentJoints

Description

Load component joints from xml

Definition

loadDependentComponentJoints(integer xmlFile, string baseName, table entry)

Arguments

integer	xmlFile	id of xml object
string	baseName	base name
table	entry	entry to add

Code

```
1626  function Cylindered:loadDependentComponentJoints(xmlFile,
      baseName, entry)
1627  if not self.isServer then
1628  return
1629  end
1630
1631  entry.componentJoints = {}
1632
1633  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName, baseName.."#componentJointIndex",
      baseName..".componentJoint#index") --FS15 to FS17
1634  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName, baseName.."#anchorActor",
      baseName..".componentJoint#anchorActor") --FS15 to FS17
1635
1636  local i = 0
1637  while true do
1638  local key = baseName .. string.format(".componentJoint(%d)", i)
```

```
1639    if not hasXMLProperty(xmlFile, key) then
1640    break
1641    end
1642    local index = getXMLInt(xmlFile, key .. "#index")
1643    if index ~= nil and self.componentJoints[index] ~= nil then
1644    local anchorActor = Utils.getNoNil(getXMLInt(xmlFile,
        key.."#anchorActor"), 0)
1645
1646    local componentJoint = self.componentJoints[index]
1647
1648    local jointEntry = {}
1649    jointEntry.componentJoint = componentJoint
1650    jointEntry.anchorActor = anchorActor
1651    jointEntry.index = index
1652
1653    local jointNode = componentJoint.jointNode
1654    if jointEntry.anchorActor == 1 then
1655    jointNode = componentJoint.jointNodeActor1
1656    end
1657
1658    local node =
        self.components[componentJoint.componentIndices[2]].node
1659    jointEntry.x, jointEntry.y, jointEntry.z = localToLocal(node,
        jointNode, 0,0,0)
1660    jointEntry.upX, jointEntry.upY, jointEntry.upZ =
        localDirectionToLocal(node, jointNode, 0,1,0)
1661    jointEntry.dirX, jointEntry.dirY, jointEntry.dirZ =
        localDirectionToLocal(node, jointNode, 0,0,1)
1662
1663    table.insert(entry.componentJoints, jointEntry)
1664    else
1665    g_logManager:xmlWarning(self.configFileName, "Invalid index for
        '%s'", key)
1666    end
1667
1668    i = i + 1
1669    end
1670    end
```

loadDependentAttacherJoints

Description

Load attacher joints from xml

Definition

loadDependentAttacherJoints(integer xmlFile, string baseName, table entry)

Arguments

integer	xmlFile	id of xml object
string	baseName	base name
table	entry	entry to add

Code

```
1677  function Cylindered:loadDependentAttacherJoints(xmlFile, baseName, entry)
1678  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, baseName.."#jointIndices", baseName..".attacherJoint#jointIndices") --FS15 to FS17
1679
1680  local indices = StringUtil.getVectorNFromString(getXMLString(xmlFile, baseName.. ".attacherJoint#jointIndices"))
1681  if indices ~= nil then
1682  entry.attacherJoints = {}
1683
1684  local availableAttacherJoints
1685  if self.getAttacherJoints ~= nil then
1686  availableAttacherJoints = self:getAttacherJoints()
1687  end
1688  if availableAttacherJoints ~= nil then
1689  for i=1, table.getn(indices) do
1690  if availableAttacherJoints[indices[i]] ~= nil then
1691  table.insert(entry.attacherJoints, availableAttacherJoints[indices[i]])
1692  end
1693  end
1694  end
1695  end
1696
1697  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, baseName.."#inputAttacherJoint", baseName..".inputAttacherJoint#value") --FS15 to FS17
1698
1699  entry.inputAttacherJoint = Utils.getNoNil(getXMLBool(xmlFile, baseName.. ".inputAttacherJoint#value"), false)
1700  end
```

loadDependentWheels

Description

Load wheels from xml

Definition

loadDependentWheels(integer xmlFile, string baseName, table entry)

Arguments

integer	xmlFile	id of xml object
string	baseName	base name
table	entry	entry to add

Code

```
1707  function Cylindered:loadDependentWheels(xmlFile, baseName, entry)
1708  if SpecializationUtil.hasSpecialization(Wheels, self.specializations) then
1709  local indices = StringUtil.getVectorNFromString(getXMLString(xmlFile, baseName.."#wheelIndices"))
1710  if indices ~= nil then
1711  entry.wheels = {}
1712  for _,wheelIndex in pairs(indices) do
1713  local wheel = self:getWheelFromWheelIndex(wheelIndex)
1714  if wheel ~= nil then
1715  table.insert(entry.wheels, wheel)
1716  else
1717  g_logManager:xmlWarning(self.configFileName, "Invalid wheelIndex '%s' for '%s'!", wheelIndex, baseName)
1718  end
1719  end
1720  end
1721  end
1722  end
```

loadDependentTranslatingParts

Description

Load translating parts

Definition

loadDependentTranslatingParts(integer xmlFile, string baseName, table entry)

Arguments

integer	xmlFile	id of xml object
string	baseName	base name
table	entry	entry to add

Code

```
1729  function Cylindered:loadDependentTranslatingParts(xmlFile, baseName, entry)
1730  entry.translatingParts = {}
1731  if entry.referencePoint ~= nil then
1732  local j = 0
1733  while true do
```

```
1734  local refBaseName = baseName..string.format(".translatingPart(%d)", j)
1735  if not hasXMLProperty(xmlFile, refBaseName) then
1736  break
1737  end
1738
1739  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, refBaseName.."#index", refBaseName.."#node") --FS15 to FS17
1740
1741  local node = I3DUtil.indexToObject(self.components, getXMLString(xmlFile, refBaseName.."#node"), self.i3dMappings)
1742  if node ~= nil then
1743  local transEntry = {}
1744  transEntry.node = node
1745  local x, y, z = getTranslation(node)
1746  transEntry.startPos = {x, y, z}
1747  local _, _, refZ = worldToLocal(node, getWorldTranslation(entry.referencePoint))
1748  transEntry.referenceDistance = refZ
1749  transEntry.referenceDistancePoint = I3DUtil.indexToObject(self.components, getXMLString(xmlFile, refBaseName.."#referenceDistancePoint"), self.i3dMappings)
1750
1751  transEntry.minZTrans = getXMLFloat(xmlFile, refBaseName.."#minZTrans")
1752  transEntry.maxZTrans = getXMLFloat(xmlFile, refBaseName.."#maxZTrans")
1753
1754  table.insert(entry.translatingParts, transEntry)
1755  end
1756  j = j + 1
1757  end
1758  end
1759  end
```

loadCopyLocalDirectionParts

Description

Load copy local direction parts from xml

Definition

loadCopyLocalDirectionParts(integer xmlFile, string baseName, table entry)

Arguments

integer xmlFile	id of xml object
string	baseName base name

table	entry	entry to add

Code

```
1811  function Cylindered:loadCopyLocalDirectionParts(xmlFile,
      baseName, entry)
1812  entry.copyLocalDirectionParts = {}
1813  local j = 0
1814  while true do
1815  local refBaseName =
      baseName..string.format(".copyLocalDirectionPart(%d)", j)
1816  if not hasXMLProperty(xmlFile, refBaseName) then
1817  break
1818  end
1819
1820  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
      self.configFileName, refBaseName.."#index",
      refBaseName.."#node") --FS15 to FS17
1821
1822  local node = I3DUtil.indexToObject(self.components,
      getXMLString(xmlFile, refBaseName.."#node"), self.i3dMappings)
1823  if node ~= nil then
1824  local copyLocalDirectionPart = {}
1825  copyLocalDirectionPart.node = node
1826  copyLocalDirectionPart.dirScale =
      StringUtil.getVectorNFromString(getXMLString(xmlFile,
      refBaseName.."#dirScale"), 3)
1827  copyLocalDirectionPart.upScale =
      StringUtil.getVectorNFromString(getXMLString(xmlFile,
      refBaseName.."#upScale"), 3)
1828
1829  self:loadDependentComponentJoints(xmlFile, refBaseName,
      copyLocalDirectionPart)
1830
1831  table.insert(entry.copyLocalDirectionParts,
      copyLocalDirectionPart)
1832  end
1833  j = j + 1
1834  end
1835  end
```

setMovingToolDirty

Description

Set moving tool dirty

Definition

setMovingToolDirty(integer node)

Arguments

integer node node id

Code

```
1855  function Cylindered:setMovingToolDirty(node)
1856  local spec = self.spec_cylindered
1857
1858  local tool = spec.nodesToMovingTools[node]
1859  if tool ~= nil then
1860  local x,y,z = getRotation(tool.node)
1861  tool.curRot[1] = x
1862  tool.curRot[2] = y
1863  tool.curRot[3] = z
1864  local x,y,z = getTranslation(tool.node)
1865  tool.curTrans[1] = x
1866  tool.curTrans[2] = y
1867  tool.curTrans[3] = z
1868  Cylindered.setDirty(self, tool)
1869
1870  if not self.isServer and self.isClient then
1871  tool.networkInterpolators.translation:setValue(tool.curTrans[tool.trans
1872  tool.networkInterpolators.rotation:setAngle(tool.curRot[tool.rotationAx
1873  end
1874  end
1875  end
```

updateCylinderedInitial

Description

Initial update of cylindered

Definition

updateCylinderedInitial(boolean placeComponents)

Arguments

boolean placeComponents place components

Code

```
1880  function Cylindered:updateCylinderedInitial(placeComponents,
      keepDirty)
1881  if placeComponents == nil then
1882  placeComponents = true
1883  end
1884
1885  if keepDirty == nil then
1886  keepDirty = false
1887  end
1888
```

```
1889  local spec = self.spec_cylindered
1890
1891  for _, part in pairs(spec.activeDirtyMovingParts) do
1892  Cylindered.setDirty(self, part)
1893  end
1894
1895  for _, tool in ipairs(spec.movingTools) do
1896  if tool.isDirty then
1897  Cylindered.updateWheels(self, tool)
1898  if self.isServer then
1899  Cylindered.updateComponentJoints(self, tool, placeComponents)
1900  end
1901  tool.isDirty = false or keepDirty
1902  end
1903
1904  self:updateDependentAnimations(tool, 9999)
1905  end
1906
1907  for _, part in ipairs(spec.movingParts) do
1908  if part.isDirty then
1909  Cylindered.updateMovingPart(self, part, placeComponents)
1910  Cylindered.updateWheels(self, part)
1911  part.isDirty = false or keepDirty
1912  end
1913
1914  self:updateDependentAnimations(part, 9999)
1915  end
1916  end
```

allowLoadMovingToolStates

Description

Returns if loading of moving tool stats from savegame is allowed

Definition

allowLoadMovingToolStates()

Return Values

boolean isAllowed is allowed

Code

```
1921  function Cylindered:allowLoadMovingToolStates(superFunc)
1922  return true
1923  end
```

isDetachAllowed

Description

Returns true if detach is allowed

Definition

isDetachAllowed()

Return Values

boolean detachAllowed detach is allowed

Code

```
1940  function Cylindered:isDetachAllowed(superFunc)
1941  local spec = self.spec_cylindered
1942  if spec.detachLockNodes ~= nil then
1943  for entry, data in pairs(spec.detachLockNodes) do
1944  local node = entry.node
1945  local rot = {getRotation(node)}
1946
1947  if data.detachingRotMinLimit ~= nil and rot[entry.rotationAxis]
      < data.detachingRotMinLimit then
1948  return false, nil
1949  end
1950  if data.detachingRotMaxLimit ~= nil and rot[entry.rotationAxis]
      > data.detachingRotMaxLimit then
1951  return false, nil
1952  end
1953
1954  local trans = {getTranslation(node)}
1955  if data.detachingTransMinLimit ~= nil and
      trans[entry.translationAxis] < data.detachingTransMinLimit then
1956  return false, nil
1957  end
1958  if data.detachingTransMaxLimit ~= nil and
      trans[entry.translationAxis] > data.detachingTransMaxLimit then
1959  return false, nil
1960  end
1961  end
1962  end
1963
1964  return superFunc(self)
1965  end
```

loadObjectChangeValuesFromXML

Description

Load object change from xml

Definition

loadObjectChangeValuesFromXML(integer xmlFile, string key, integer node, table object)

Arguments

integer xmlFile id of xml object
string key key
integer node node id
table object object

Code

```
1973  function Cylindered:loadObjectChangeValuesFromXML(superFunc,
      xmlFile, key, node, object)
1974  superFunc(self, xmlFile, key, node, object)
1975
1976  local spec = self.spec_cylindered
1977
1978  if spec.nodesToMovingTools ~= nil and
      spec.nodesToMovingTools[node] ~= nil then
1979  local movingTool = spec.nodesToMovingTools[node]
1980  object.movingToolRotMaxActive =
      Utils.getNoNilRad(getXMLFloat(xmlFile,
      key.."#movingToolRotMaxActive"), movingTool.rotMax)
1981  object.movingToolRotMaxInactive =
      Utils.getNoNilRad(getXMLFloat(xmlFile,
      key.."#movingToolRotMaxInactive"), movingTool.rotMax)
1982  object.movingToolRotMinActive =
      Utils.getNoNilRad(getXMLFloat(xmlFile,
      key.."#movingToolRotMinActive"), movingTool.rotMin)
1983  object.movingToolRotMinInactive =
      Utils.getNoNilRad(getXMLFloat(xmlFile,
      key.."#movingToolRotMinInactive"), movingTool.rotMin)
1984  end
1985  end
```

setObjectChangeValues

Description

Sets object change values

Definition

setObjectChangeValues(table object, boolean isActive)

Arguments

table object object
boolean isActive is active

Code

```
1991  function Cylindered:setObjectChangeValues(superFunc, object,
      isActive)
1992  superFunc(self, object, isActive)
1993
1994  local spec = self.spec_cylindered
1995
```

```
1996  if spec.nodesToMovingTools ~= nil and
      spec.nodesToMovingTools[object.node] ~= nil then
1997  local movingTool = spec.nodesToMovingTools[object.node]
1998  if isActive then
1999  movingTool.rotMax = object.movingToolRotMaxActive
2000  movingTool.rotMin = object.movingToolRotMinActive
2001  else
2002  movingTool.rotMax = object.movingToolRotMaxInactive
2003  movingTool.rotMin = object.movingToolRotMinInactive
2004  end
2005  end
2006  end
```

getWearMultiplier

Description

Returns current wear multiplier

Definition

getWearMultiplier()

Return Values

float wearMultiplier current wear multiplier

Code

```
2124  function Cylindered:getWearMultiplier(superFunc)
2125  local spec = self.spec_cylindered
2126  local multiplier = superFunc(self)
2127
2128  if spec.isHydraulicSamplePlaying then
2129  multiplier = multiplier + self:getWorkWearMultiplier()
2130  end
2131
2132  return multiplier
2133  end
```

onPostAttach

Description

Called if vehicle gets attached

Definition

onPostAttach(table attacherVehicle, integer inputJointDescIndex, integer jointDescIndex)

Arguments

table	attacherVehicle	attacher vehicle
integer	inputJointDescIndex	index of input attacher joint
integer	jointDescIndex	index of attacher joint it gets attached to

Code

```
2161  function Cylindered:onPostAttach(attacherVehicle,
      inputJointDescIndex, jointDescIndex)
```

```
2162  local spec = self.spec_cylindered
2163
2164  for _, tool in ipairs(spec.movingTools) do
2165  local changed = false
2166  if tool.transSpeed ~= nil then
2167  local trans = tool.curTrans[tool.translationAxis]
2168
2169  local changedTrans = false
2170  if tool.attachTransMax ~= nil and trans > tool.attachTransMax
      then
2171  trans = tool.attachTransMax
2172  changedTrans = true
2173  elseif tool.attachTransMin ~= nil and trans <
      tool.attachTransMin then
2174  trans = tool.attachTransMin
2175  changedTrans = true
2176  end
2177  if changedTrans then
2178  tool.curTrans[tool.translationAxis] = trans
2179  setTranslation(tool.node, unpack(tool.curTrans))
2180  changed = true
2181  end
2182  end
2183  if tool.rotSpeed ~= nil then
2184  local rot = tool.curRot[tool.rotationAxis]
2185
2186  local changedRot = false
2187  if tool.attachRotMax ~= nil and rot > tool.attachRotMax then
2188  rot = tool.attachRotMax
2189  changedRot = true
2190  elseif tool.attachRotMin ~= nil and rot < tool.attachRotMin then
2191  rot = tool.attachRotMin
2192  changedRot = true
2193  end
2194  if changedRot then
2195  tool.curRot[tool.rotationAxis] = rot
2196  setRotation(tool.node, unpack(tool.curRot))
2197  changed = true
2198  end
2199  end
2200  if changed then
```

```
2201  Cylindered.setDirty(self, tool)
2202  end
2203  end
2204  end
```

onDeactivate

Description

Called on deactivate

Definition

onDeactivate()

Code

```
2223  function Cylindered:onDeactivate()
2224  if self.isClient then
2225  local spec = self.spec_cylindered
2226  g_soundManager:stopSample(spec.samples.hydraulic)
2227  spec.isHydraulicSamplePlaying = false
2228  end
2229  end
```

setToolTranslation

Description

Set tool translation

Definition

setToolTranslation(table tool, float transSpeed, float dt)

Arguments

table	tool	tool
float	transSpeed	translation speed
float	dt	time since last call in ms

Return Values

boolean	changed	translation changed

Code

```
2237  function Cylindered.setToolTranslation(self, tool, transSpeed,
      dt, delta)
2238  local curTrans = { getTranslation(tool.node) }
2239  local newTrans = curTrans[tool.translationAxis]
2240  local oldTrans = newTrans
2241  if transSpeed ~= nil then
2242  newTrans = newTrans + transSpeed*dt
2243  else
2244  newTrans = newTrans + delta
2245  end
2246  if tool.transMax ~= nil then
2247  newTrans = math.min(newTrans, tool.transMax)
2248  end
```

```
2249  if tool.transMin ~= nil then
2250  newTrans = math.max(newTrans, tool.transMin)
2251  end
2252  local diff = newTrans - oldTrans
2253  if dt ~= 0 then
2254  tool.lastTransSpeed = diff/dt
2255  end
2256  if math.abs(diff) > 0.0001 then
2257  curTrans[tool.translationAxis] = newTrans
2258  tool.curTrans = curTrans
2259  setTranslation(tool.node, unpack(tool.curTrans))
2260
2261  SpecializationUtil.raiseEvent(self, "onMovingToolChanged", tool,
      transSpeed, dt)
2262
2263  return true
2264  end
2265
2266  return false
2267  end
```

setToolRotation

Description

Set tool rotation

Definition

setToolRotation(table tool, float rotSpeed, float dt, float delta)

Arguments

table	tool	tool
float	rotSpeed	rotation speed
float	dt	time since last call in ms
float	delta	delta rotation

Return Values

boolean changed rotation changed

Code

```
2276  function Cylindered.setToolRotation(self, tool, rotSpeed, dt,
      delta)
2277  local curRot = { getRotation(tool.node) }
2278  local newRot = curRot[tool.rotationAxis]
2279  if rotSpeed ~= nil then
2280  newRot = newRot + rotSpeed*dt
2281  else
2282  newRot = newRot + delta
2283  end
```

```
2284  if tool.rotMax ~= nil then
2285  newRot = math.min(newRot, tool.rotMax)
2286  end
2287  if tool.rotMin ~= nil then
2288  newRot = math.max(newRot, tool.rotMin)
2289  end
2290  local diff = newRot - curRot[tool.rotationAxis]
2291  if rotSpeed ~= nil then
2292  if dt ~= 0 then
2293  tool.lastRotSpeed = diff/dt
2294  end
2295  end
2296
2297  if math.abs(diff) > 0.0001 then
2298  -- wrap if not limited
2299  if tool.rotMin == nil and tool.rotMax == nil then
2300  if newRot > 2*math.pi then
2301  newRot = newRot - 2*math.pi
2302  end
2303  if newRot < 0 then
2304  newRot = newRot + 2*math.pi
2305  end
2306  end
2307  tool.curRot[tool.rotationAxis] = newRot
2308  setRotation(tool.node, unpack(tool.curRot))
2309
2310  SpecializationUtil.raiseEvent(self, "onMovingToolChanged", tool,
     rotSpeed, dt)
2311
2312  return true
2313  end
2314
2315  return false
2316  end
```

setToolAnimation

Description

Set tool animation

Definition

setToolAnimation(table tool, float animSpeed, float dt)

Arguments

table tool	tool

float animSpeed animation speed
float dt time since last call in ms

Return Values

boolean changed animation changed

Code

```
2324  function Cylindered.setToolAnimation(self, tool, animSpeed, dt)
2325  local newAnimTime = tool.curAnimTime+animSpeed*dt
2326
2327  if tool.animMaxTime ~= nil then
2328  newAnimTime = math.min(newAnimTime, tool.animMaxTime)
2329  end
2330  if tool.animMinTime ~= nil then
2331  newAnimTime = math.max(newAnimTime, tool.animMinTime)
2332  end
2333  local diff = newAnimTime - tool.curAnimTime
2334  if dt ~= 0 then
2335  tool.lastAnimSpeed = diff / dt
2336  end
2337  if math.abs(diff) > 0.0001 then
2338  tool.curAnimTime = newAnimTime
2339  self:setAnimationTime(tool.animName, newAnimTime)
2340  SpecializationUtil.raiseEvent(self, "onMovingToolChanged", tool,
      animSpeed, dt)
2341  return true
2342  end
2343
2344  return false
2345  end
```

getMovingToolState

Description

Returns moving tool state

Definition

getMovingToolState(table tool)

Arguments

table tool tool

Return Values

float state state of moving tool [0..1]

Code

```
2351  function Cylindered.getMovingToolState(self, tool)
2352  local state = 0
2353  if tool.rotMax ~= nil and tool.rotMin ~= nil then
```

```
2354  state = (tool.curRot[tool.rotationAxis]-tool.rotMin) /
      (tool.rotMax-tool.rotMin)
2355  else
2356  if tool.transMax ~= nil and tool.transMin ~= nil then
2357  state = (tool.curTrans[tool.translationAxis]-tool.transMin) /
      (tool.transMax-tool.transMin)
2358  end
2359  end
2360
2361  return state
2362  end
```

setDirty

Description

Set dirty

Definition

setDirty(table part)

Arguments

table part part to set dirty

Code

```
2367  function Cylindered.setDirty(self, part)
2368  if not part.isDirty or self.spec_cylindered.isLoading then --
      during loading we always allow setting dirty since we do not
      reset it
2369  part.isDirty = true
2370  self.anyMovingPartsDirty = true
2371
2372  if part.delayedNode ~= nil then
2373  self:setDelayedData(part)
2374  end
2375
2376  Cylindered.updateAttacherJoints(self, part)
2377  Cylindered.updateWheels(self, part)
2378
2379  for _, v in pairs(part.dependentParts) do
2380  Cylindered.setDirty(self, v)
2381  end
2382  end
2383  end
```

updateWheels

Description

Update wheel of part

Definition

updateWheels(table part)

Arguments

table part part

Code

```
2388  function Cylindered.updateWheels(self, part)
2389  if part.wheels ~= nil then
2390  for _, wheel in pairs(part.wheels) do
2391  wheel.steeringCenterOffsetX, wheel.steeringCenterOffsetY,
      wheel.steeringCenterOffsetZ = localToLocal(wheel.driveNode,
      wheel.repr, 0, 0, 0)
2392  wheel.steeringCenterOffsetX = -wheel.steeringCenterOffsetX
2393  wheel.steeringCenterOffsetY = -wheel.steeringCenterOffsetY
2394  wheel.steeringCenterOffsetZ = -wheel.steeringCenterOffsetZ
2395
2396  wheel.positionX, wheel.positionY, wheel.positionZ =
      localToLocal(getParent(wheel.repr), wheel.node,
      wheel.startPositionX-wheel.steeringCenterOffsetX,
      wheel.startPositionY-wheel.steeringCenterOffsetY,
      wheel.startPositionZ-wheel.steeringCenterOffsetZ)
2397  if wheel.useReprDirection then
2398  wheel.directionX, wheel.directionY, wheel.directionZ =
      localDirectionToLocal(getParent(wheel.repr), wheel.node, 0,-1,0)
2399  wheel.axleX, wheel.axleY, wheel.axleZ =
      localDirectionToLocal(getParent(wheel.repr), wheel.node, 1,0,0)
2400  elseif wheel.useDriveNodeDirection then
2401  wheel.directionX, wheel.directionY, wheel.directionZ =
      localDirectionToLocal(getParent(wheel.driveNode), wheel.node,
      0,-1,0)
2402  wheel.axleX, wheel.axleY, wheel.axleZ =
      localDirectionToLocal(getParent(wheel.driveNode), wheel.node,
      1,0,0)
2403  end
2404
2405  self:updateWheelBase(wheel)
2406  end
2407  end
2408  end
```

updateMovingPart

Description

Update moving part

Definition

updateMovingPart(table part, boolean placeComponents, boolean updateDependentParts)

Arguments

table part part

boolean placeComponents place components
boolean updateDependentParts update dependent parts

Code

```
2415  function Cylindered.updateMovingPart(self, part, placeComponents, updateDependentParts)
2416
2417  -- the local reference point must be referenceDistance away from the referencePoint
2418  local refX,refY,refZ
2419  local dirX, dirY, dirZ = 0,0,0
2420  if part.referencePoint ~= nil then
2421  if part.moveToReferenceFrame then
2422  local x,y,z = localToLocal(part.referenceFrame, getParent(part.node), part.referenceFrameOffset[1], part.referenceFrameOffset[2], part.referenceFrameOffset[3])
2423  setTranslation(part.node, x,y,z)
2424  end
2425  refX,refY,refZ = getWorldTranslation(part.referencePoint)
2426  if part.referenceDistance == 0 then
2427  if part.useLocalOffset then
2428  local lx, ly, lz = worldToLocal(part.node, refX, refY, refZ)
2429  dirX, dirY, dirZ = localDirectionToWorld(part.node, lx-part.localReferencePoint[1], ly-part.localReferencePoint[2], lz)
2430  else
2431  local x,y,z = getWorldTranslation(part.node)
2432  dirX, dirY, dirZ = refX - x, refY-y, refZ-z
2433  end
2434  else
2435  if part.updateLocalReferenceDistance then
2436  local _,y,z = worldToLocal(part.node, getWorldTranslation(part.localReferencePointNode))
2437  part.localReferenceDistance = MathUtil.vector2Length(y, z)
2438  end
2439  if part.referenceDistancePoint ~= nil then
2440  local _,_,z = worldToLocal(part.node, getWorldTranslation(part.referenceDistancePoint))
2441  part.referenceDistance = z
2442  end
2443
2444  if part.localReferenceTranslate then
2445  local _, ly, lz = worldToLocal(part.node, refX, refY, refZ)
2446
```

```
2447  -- calculate line-circle intersection
2448  if math.abs(ly) < part.referenceDistance then
2449  local dz =
      math.sqrt(part.referenceDistance*part.referenceDistance - ly*ly)
2450
2451  local z1 = (lz - dz) - part.localReferenceDistance
2452  local z2 = (lz + dz) - part.localReferenceDistance
2453  if math.abs(z2) < math.abs(z1) then
2454  z1 = z2
2455  end
2456  local parentNode = getParent(part.node)
2457  local tx,ty,tz = unpack(part.localReferenceTranslation)
2458  local _, _, coz = localToLocal(parentNode, part.node, tx,ty,tz)
2459  local ox,oy,oz = localDirectionToLocal(part.node, parentNode,
      0,0,z1-coz)
2460  setTranslation(part.node, tx+ox,ty+oy,tz+oz)
2461  end
2462  else
2463
2464  local r1 = part.localReferenceDistance
2465  local r2 = part.referenceDistance
2466  local _, ly, lz = worldToLocal(part.node, refX, refY, refZ)
2467  --print("intersect: "..ly .. " "..lz)
2468  local ix, iy, i2x, i2y =
      MathUtil.getCircleCircleIntersection(0,0, r1, ly, lz, r2)
2469
2470  if ix ~= nil then
2471  if i2x ~= nil then
2472  -- use the point which as the same angle side as the original
      configuration
2473  local side = ix*(lz-iy) - iy*(ly-ix)
2474  if (side < 0) ~= (part.localReferenceAngleSide < 0) then
2475  iy = i2y
2476  ix = i2x
2477  end
2478  end
2479  dirX, dirY, dirZ = localDirectionToWorld(part.node, 0, ix, iy)
2480  end
2481  end
2482  end
2483  else
```

```
2484  if part.alignToWorldY then
2485  dirX, dirY, dirZ = localDirectionToWorld(getRootNode(), 0, 1, 0)
2486  else
2487  dirX, dirY, dirZ = localDirectionToWorld(part.referenceFrame, 0,
      0, 1)
2488  end
2489  if part.moveToReferenceFrame then
2490  local x,y,z = localToLocal(part.referenceFrame,
      getParent(part.node), part.referenceFrameOffset[1],
      part.referenceFrameOffset[2], part.referenceFrameOffset[3])
2491  setTranslation(part.node, x,y,z)
2492  end
2493
2494  if part.doLineAlignment then
2495  local foundPoint = false
2496  for i=1, #part.orientationLineNodes-1 do
2497  local startNode = part.orientationLineNodes[i]
2498  local endNode = part.orientationLineNodes[i+1]
2499
2500  local _, sy, sz = localToLocal(startNode, part.referenceFrame,
      0, 0, 0)
2501  local _, ey, ez = localToLocal(endNode, part.referenceFrame, 0,
      0, 0)
2502  local _, cy, cz = localToLocal(part.node, part.referenceFrame,
      0, 0, 0)
2503
2504  local hasIntersection, i1y, i1z, i2y, i2z =
      MathUtil.getCircleLineIntersection(cy, cz, part.partLength, sy,
      sz, ey, ez)
2505  if hasIntersection then
2506  local targetY, targetZ
2507  if not MathUtil.getIsOutOfBounds(i1y, sy, ey) and not
      MathUtil.getIsOutOfBounds(i1z, sz, ez) then
2508  targetY, targetZ = i1y, i1z
2509  foundPoint = true
2510  end
2511  if not MathUtil.getIsOutOfBounds(i2y, sy, ey) and not
      MathUtil.getIsOutOfBounds(i2z, sz, ez) then
2512  targetY, targetZ = i2y, i2z
2513  foundPoint = true
2514  end
2515
2516  if foundPoint then
```

```
2517  dirX, dirY, dirZ = localDirectionToWorld(part.referenceFrame, 0, targetY, targetZ)
2518  local upX, upY, upZ = localDirectionToWorld(part.referenceFrame, 0, 1, 0)
2519
2520  I3DUtil.setWorldDirection(part.node, dirX, dirY, dirZ, upX, upY, upZ, part.limitedAxis, part.minRot, part.maxRot)
2521  break
2522  end
2523  end
2524  end
2525  end
2526  end
2527  if (dirX ~= 0 or dirY ~= 0 or dirZ ~= 0) and part.doDirectionAlignment then
2528  local upX, upY, upZ = localDirectionToWorld(part.referenceFrame, 0, 1, 0)
2529  if part.invertZ then
2530  dirX = -dirX
2531  dirY = -dirY
2532  dirZ = -dirZ
2533  end
2534
2535  I3DUtil.setWorldDirection(part.node, dirX, dirY, dirZ, upX, upY, upZ, part.limitedAxis, part.minRot, part.maxRot)
2536
2537  if part.scaleZ and part.localReferenceDistance ~= nil then
2538  local len = MathUtil.vector3Length(dirX, dirY, dirZ)
2539  setScale(part.node, 1, 1, len/part.localReferenceDistance)
2540  end
2541  end
2542  if part.doRotationAlignment then
2543  local x,y,z = getRotation(part.referenceFrame)
2544  setRotation(part.node, x*part.rotMultiplier, y*part.rotMultiplier, z*part.rotMultiplier)
2545  end
2546
2547  if part.referencePoint ~= nil then
2548  local numTranslatingParts = table.getn(part.translatingParts)
2549  if numTranslatingParts > 0 then
2550  local _, _, dist = worldToLocal(part.node, refX, refY, refZ)
2551  for i=1,numTranslatingParts do
```

```
2552  local translatingPart = part.translatingParts[i]
2553  local newZ = (dist - translatingPart.referenceDistance)/numTranslatingParts
2554
2555  if translatingPart.minZTrans ~= nil then
2556  newZ = math.max(translatingPart.minZTrans, newZ)
2557  end
2558
2559  if translatingPart.maxZTrans ~= nil then
2560  newZ = math.min(translatingPart.maxZTrans, newZ)
2561  end
2562
2563  setTranslation(translatingPart.node, translatingPart.startPos[1], translatingPart.startPos[2], newZ)
2564  end
2565  end
2566  end
2567
2568  if part.copyLocalDirectionParts ~= nil then
2569  for _,copyLocalDirectionPart in pairs(part.copyLocalDirectionParts) do
2570  local dx,dy,dz = localDirectionToWorld(part.node, 0,0,1)
2571  dx,dy,dz = worldDirectionToLocal(getParent(part.node), dx,dy,dz)
2572  dx = dx * copyLocalDirectionPart.dirScale[1]
2573  dy = dy * copyLocalDirectionPart.dirScale[2]
2574  dz = dz * copyLocalDirectionPart.dirScale[3]
2575
2576  local ux,uy,uz = localDirectionToWorld(part.node, 0,1,0)
2577  ux,uy,uz = worldDirectionToLocal(getParent(part.node), ux,uy,uz)
2578  ux = ux * copyLocalDirectionPart.upScale[1]
2579  uy = uy * copyLocalDirectionPart.upScale[2]
2580  uz = uz * copyLocalDirectionPart.upScale[3]
2581
2582  setDirection(copyLocalDirectionPart.node, dx,dy,dz, ux,uy,uz)
2583
2584  if self.isServer then
2585  Cylindered.updateComponentJoints(self, copyLocalDirectionPart, placeComponents)
2586  end
2587  end
2588  end
```

```
2589
2590  -- update component joint
2591  if self.isServer then
2592  Cylindered.updateComponentJoints(self, part, placeComponents)
2593  Cylindered.updateAttacherJoints(self, part)
2594  end
2595
2596  if updateDependentParts and part.dependentParts ~= nil then
2597  for _,part in pairs(part.dependentParts) do
2598  Cylindered.updateMovingPart(self, part, placeComponents,
      updateDependentParts)
2599  end
2600  end
2601
2602  part.isDirty = false
2603  end
```

updateComponentJoints

Description

Update component joints

Definition

updateComponentJoints(table entry, boolean placeComponents)

Arguments

table	entry	entry
boolean	placeComponents	place components

Code

```
2609  function Cylindered.updateComponentJoints(self, entry,
      placeComponents)
2610  if self.isServer then
2611  if entry.componentJoints ~= nil then
2612  for _,joint in ipairs(entry.componentJoints) do
2613  local componentJoint = joint.componentJoint
2614
2615  local jointNode = componentJoint.jointNode
2616  if joint.anchorActor == 1 then
2617  jointNode = componentJoint.jointNodeActor1
2618  end
2619
2620  if placeComponents then
2621  local node =
      self.components[componentJoint.componentIndices[2]].node
2622  local x,y,z = localToWorld(jointNode, joint.x, joint.y, joint.z)
```

```
2623  local upX,upY,upZ = localDirectionToWorld(jointNode, joint.upX,joint.upY,joint.upZ)
2624  local dirX,dirY,dirZ = localDirectionToWorld(jointNode, joint.dirX,joint.dirY,joint.dirZ)
2625  setWorldTranslation(node, x,y,z)
2626  I3DUtil.setWorldDirection(node, dirX,dirY,dirZ, upX,upY,upZ)
2627  end
2628
2629  self:setComponentJointFrame(componentJoint, joint.anchorActor)
2630  end
2631  end
2632  end
2633  end
```

updateAttacherJoints

Description

Update attacher joints

Definition

updateAttacherJoints(table entry)

Arguments

table entry entry

Code

```
2638  function Cylindered.updateAttacherJoints(self, entry)
2639  if self.isServer then
2640  if entry.attacherJoints ~= nil then
2641  for _,joint in ipairs(entry.attacherJoints) do
2642  if joint.jointIndex ~= 0 then
2643  setJointFrame(joint.jointIndex, 0, joint.jointTransform)
2644  end
2645  end
2646  end
2647
2648  if entry.inputAttacherJoint then
2649  if self.getAttacherVehicle ~= nil then
2650  local attacherVehicle = self:getAttacherVehicle()
2651  if attacherVehicle ~= nil then
2652
2653  local attacherJoints = attacherVehicle:getAttacherJoints()
2654  if attacherJoints ~= nil then
2655
2656  local jointDescIndex = attacherVehicle:getAttacherJointIndexFromObject(self)
```

```
2657  if jointDescIndex ~= nil then
2658  local jointDesc = attacherJoints[jointDescIndex]
2659
2660  local inputAttacherJoint = self:getActiveInputAttacherJoint()
2661  if inputAttacherJoint ~= nil then
2662  setJointFrame(jointDesc.jointIndex, 1, inputAttacherJoint.node )
2663  end
2664  end
2665  end
2666  end
2667  end
2668  end
2669  end
2670  end
```

CylinderedFoldable
Description

Class for cylindered foldables

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
19  function CylinderedFoldable.prerequisitesPresent(specializations)
20  return SpecializationUtil.hasSpecialization(Cylindered,
    specializations) and
    SpecializationUtil.hasSpecialization(Foldable, specializations)
21  end
```

Dischargeable
Description

getDischargeTargetObject

Description

Object found for possible discharging

Definition

getDischargeTargetObject()

getCurrentDischargeObject

Description

Object being discharged to

Definition

getCurrentDischargeObject()

Drivable
Description

Specialization class for Drivables

prerequisitesPresent
Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
26  function Drivable.prerequisitesPresent(specializations)
27  return SpecializationUtil.hasSpecialization(Enterable,
    specializations) and
    SpecializationUtil.hasSpecialization(Motorized, specializations)
28  end
```

onLoad
Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
69  function Drivable:onLoad(savegame)
70  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.steering#index",
    "vehicle.drivable.steeringWheel#node") --FS17 to FS19
71  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.steering#node",
    "vehicle.drivable.steeringWheel#node") --FS17 to FS19
72  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.cruiseControl",
    "vehicle.drivable.cruiseControl") --FS17 to FS19
73  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.indoorHud.cruiseControl",
    "vehicle.drivable.dashboards.dashboard with valueType
    'cruiseControl'") --FS17 to FS19
74  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.showChangeToolSelectionHelp") --
    FS17 to FS19
```

```
75  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.maxRotatedTimeSpeed#value") --FS17
    to FS19
76  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.speedRotScale#scale",
    "vehicle.drivable.speedRotScale#scale") --FS17 to FS19
77  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.speedRotScale#offset",
    "vehicle.drivable.speedRotScale#offset") --FS17 to FS19
78
79  local spec = self.spec_drivable
80
81  spec.allowPlayerControl = true
82  spec.showToolSelectionHud = true
83
84  spec.doHandbrake = false
85  spec.doHandbrakeSend = false
86  spec.reverserDirection = 1
87
88  -- attributes set by action events
89  spec.lastInputValues = {}
90  spec.lastInputValues.axisAccelerate = 0
91  spec.lastInputValues.axisBrake = 0
92  spec.lastInputValues.axisSteer = 0
93  spec.lastInputValues.axisSteerIsAnalog = false
94  spec.lastInputValues.cruiseControlValue = 0
95  spec.lastInputValues.cruiseControlState = 0
96
97  -- 'final' attrbutes calculated
98  spec.axisForward = 0
99  spec.axisForwardSend = 0
100
101 spec.axisSide = 0
102 spec.axisSideSend = 0
103 spec.axisSideLast = 0
104
105 spec.speedRotScale = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.drivable.speedRotScale#scale"), 80)
106 spec.speedRotScaleOffset =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.drivable.speedRotScale#offset"), 0.7)
107
```

```
108 local motor = self:getMotor()
109 spec.cruiseControl = {}
110 spec.cruiseControl.maxSpeed =
    Utils.getNoNil(getXMLInt(self.xmlFile,
    "vehicle.drivable.cruiseControl#maxSpeed"),
    math.ceil(motor:getMaximumForwardSpeed()*3.6))
111 spec.cruiseControl.minSpeed =
    Utils.getNoNil(getXMLInt(self.xmlFile,
    "vehicle.drivable.cruiseControl#minSpeed"), math.min(1,
    spec.cruiseControl.maxSpeed))
112 spec.cruiseControl.speed = spec.cruiseControl.maxSpeed
113 spec.cruiseControl.isActive =
    Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.drivable.cruiseControl#enabled"), true)
114 spec.cruiseControl.state = Drivable.CRUISECONTROL_STATE_OFF
115 spec.cruiseControl.topSpeedTime = 2000
116 spec.cruiseControl.changeDelay = 250
117 spec.cruiseControl.changeCurrentDelay = 0
118 spec.cruiseControl.changeMultiplier = 1
119 spec.cruiseControl.speedSent = spec.cruiseControl.speed
120
121 local node = I3DUtil.indexToObject( self.components,
    getXMLString(self.xmlFile,
    "vehicle.drivable.steeringWheel#node"), self.i3dMappings)
122 if node ~= nil then
123 spec.steeringWheel = {}
124 spec.steeringWheel.node = node
125 local _,ry,_ = getRotation(spec.steeringWheel.node)
126 spec.steeringWheel.lastRotation = ry
127 spec.steeringWheel.indoorRotation = math.rad( Utils.getNoNil(
    getXMLFloat(self.xmlFile,
    "vehicle.drivable.steeringWheel#indoorRotation"), 0 ) )
128 spec.steeringWheel.outdoorRotation = math.rad( Utils.getNoNil(
    getXMLFloat(self.xmlFile,
    "vehicle.drivable.steeringWheel#outdoorRotation"), 0 ) )
129 end
130
131 if self.loadDashboardsFromXML ~= nil then
132 local dashKey = "vehicle.drivable.dashboards"
133 self:loadDashboardsFromXML(self.xmlFile, dashKey,
    {valueTypeToLoad = "cruiseControl",
134 valueObject = spec.cruiseControl,
135 valueFunc = "speed"})
136
```

```
137 self:loadDashboardsFromXML(self.xmlFile, dashKey,
    {valueTypeToLoad = "directionForward",
138 valueObject = self,
139 valueFunc = "getIsDrivingForward"})
140
141 self:loadDashboardsFromXML(self.xmlFile, dashKey,
    {valueTypeToLoad = "directionBackward",
142 valueObject = self,
143 valueFunc = "getIsDrivingBackward"})
144
145 self:loadDashboardsFromXML(self.xmlFile, dashKey,
    {valueTypeToLoad = "movingDirection",
146 valueObject = self,
147 valueFunc = "getDrivingDirection",
148 minFunc = -1,
149 maxFunc = 1})
150
151 self:loadDashboardsFromXML(self.xmlFile, dashKey,
    {valueTypeToLoad = "cruiseControlActive",
152 valueObject = spec.cruiseControl,
153 valueFunc = "state",
154 valueCompare = Drivable.CRUISECONTROL_STATE_ACTIVE})
155
156 self:loadDashboardsFromXML(self.xmlFile, dashKey,
    {valueTypeToLoad = "accelerationAxis",
157 valueObject = self,
158 valueFunc = "getAccelerationAxis"})
159
160 self:loadDashboardsFromXML(self.xmlFile, dashKey,
    {valueTypeToLoad = "decelerationAxis",
161 valueObject = self,
162 valueFunc = "getDecelerationAxis"})
163
164 self:loadDashboardsFromXML(self.xmlFile, dashKey,
    {valueTypeToLoad = "ac_decelerationAxis",
165 valueObject = self,
166 valueFunc = "getAcDecelerationAxis"})
167
168 self:loadDashboardsFromXML(self.xmlFile, dashKey,
    {valueTypeToLoad = "steeringAngle",
169 valueObject = self,
170 valueFunc = "getDashboardSteeringAxis",
```

```
171 minFunc = -1,
172 maxFunc = 1})
173 end
174
175 if self.isClient then
176 spec.samples = {}
177 spec.samples.waterSplash =
    g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.drivable.sounds", "waterSplash", self.baseDirectory,
    self.components, 1, AudioGroup.VEHICLE, self.i3dMappings, self)
178 if self.isClient and g_isDevelopmentVersion then
179 if spec.samples.waterSplash == nil then
180 g_logManager:xmlDevWarning(self.configFileName, "Missing drivable
    waterSplash sound")
181 end
182 end
183 end
184
185 if savegame ~= nil then
186 local maxSpeed = getXMLInt(savegame.xmlFile,
    savegame.key..".drivable#cruiseControl")
187 self:setCruiseControlMaxSpeed(maxSpeed)
188 end
189
190 spec.dirtyFlag = self:getNextDirtyFlag()
191 end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
195 function Drivable:onDelete()
196 if self.isClient then
197 local spec = self.spec_drivable
198 for _,sample in pairs(spec.samples) do
199 g_soundManager:deleteSample(sample)
200 end
201 end
202 end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
213 function Drivable:onReadStream(streamId, connection)
214 local spec = self.spec_drivable
215
216 if spec.cruiseControl.isActive then
217 self:setCruiseControlState(streamReadUIntN(streamId, 2), true)
218 local speed = streamReadUInt8(streamId)
219 self:setCruiseControlMaxSpeed(speed)
220 spec.cruiseControl.speedSent = speed
221 end
222 end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
228 function Drivable:onWriteStream(streamId, connection)
229 local spec = self.spec_drivable
230
231 if spec.cruiseControl.isActive then
232 streamWriteUIntN(streamId, spec.cruiseControl.state, 2)
233 streamWriteUInt8(streamId, spec.cruiseControl.speed)
234 end
235 end
```

onReadUpdateStream

Description

Called on on update

Definition

onReadUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId stream ID

integer timestamp timestamp

table connection connection

Code

```
242 function Drivable:onReadUpdateStream(streamId, timestamp, connection)
243 local spec = self.spec_drivable
244
245 if streamReadBool(streamId) then
246 spec.axisForward = streamReadUIntN(streamId, 10) / 1023 * 2 - 1
247 spec.axisSide = streamReadUIntN(streamId, 10) / 1023 * 2 - 1
248 spec.doHandbrake = streamReadBool(streamId)
249 end
250 end
```

onWriteUpdateStream

Description

Called on on update

Definition

onWriteUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId stream ID
table connection connection
integer dirtyMask dirty mask

Code

```
257 function Drivable:onWriteUpdateStream(streamId, connection, dirtyMask)
258 local spec = self.spec_drivable
259
260 if streamWriteBool(streamId, bitAND(dirtyMask, spec.dirtyFlag) ~= 0) then
261 local axisForward = (spec.axisForward + 1) / 2 * 1023
262 streamWriteUIntN(streamId, axisForward, 10)
263
264 local axisSide = (spec.axisSide + 1) / 2 * 1023
265 streamWriteUIntN(streamId, axisSide, 10)
266
267 streamWriteBool(streamId, spec.doHandbrake)
268 end
269 end
```

setCruiseControlState

Description

Changes cruise control state

Definition

setCruiseControlState(integer state, boolean noEventSend)

Arguments

integer	state	new state
boolean	noEventSend	no event send

Code

```
418  function Drivable:setCruiseControlState(state, noEventSend)
419  local spec = self.spec_drivable
420  if spec.cruiseControl.state ~= state then
421  spec.cruiseControl.state = state
422  if not self.isServer and (noEventSend == nil or not noEventSend) then
423  g_client:getServerConnection():sendEvent(SetCruiseControlStateEvent:new(
     state))
424  end
425
426  if spec.toggleCruiseControlEvent ~= nil then
427  local text
428  if state ~= Drivable.CRUISECONTROL_STATE_ACTIVE then
429  text = g_i18n:getText("action_activateCruiseControl")
430  else
431  text = g_i18n:getText("action_deactivateCruiseControl")
432  end
433  g_inputBinding:setActionEventText(spec.toggleCruiseControlEvent, text)
434  end
435  end
436  end
```

setCruiseControlMaxSpeed

Description

Sets cruise control max speed

Definition

setCruiseControlMaxSpeed(float speed)

Arguments

float	speed	new max speed

Code

```
441  function Drivable:setCruiseControlMaxSpeed(speed)
442  local spec = self.spec_drivable
443
444  if speed ~= nil then
445  speed = MathUtil.clamp(speed, spec.cruiseControl.minSpeed,
     spec.cruiseControl.maxSpeed)
446  if spec.cruiseControl.speed ~= speed then
447  spec.cruiseControl.speed = speed
448  if spec.cruiseControl.state == Drivable.CRUISECONTROL_STATE_FULL
     then
```

```
449 spec.cruiseControl.state = Drivable.CRUISECONTROL_STATE_ACTIVE
450 end
451 end
452
453 if spec.cruiseControl.state ~= Drivable.CRUISECONTROL_STATE_OFF
    then
454 if spec.cruiseControl.state ==
    Drivable.CRUISECONTROL_STATE_ACTIVE then
455 local speed,_ = self:getSpeedLimit(true)
456 speed = math.min(speed, spec.cruiseControl.speed)
457 self:getMotor():setSpeedLimit(speed)
458 else
459 self:getMotor():setSpeedLimit(math.huge)
460 end
461 end
462 end
463 end
```

onCameraChanged

Description

Called on camera change

Definition

onCameraChanged(table camera, integer camIndex)

Arguments

table camera new camera

integer camIndex index of new camera

Code

```
570 function Drivable:onCameraChanged(camera, camIndex)
571 end
```

stopMotor

Description

Called on motor stop

Definition

stopMotor(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
576 function Drivable:stopMotor(noEventSend)
577 self:setCruiseControlState(Drivable.CRUISECONTROL_STATE_OFF,
    true)
578 end
```

updateVehiclePhysics

Description

Update vehicle physics

Definition

updateVehiclePhysics(float axisForward, boolean axisForwardIsAnalog, float axisSide, boolean axisSideIsAnalog, boolean doHandbrake, float dt)

Arguments

float	axisForward	axis forward value
boolean	axisForwardIsAnalog	forward axis is analog
float	axisSide	side axis
boolean	axisSideIsAnalog	side axis is analog
boolean	doHandbrake	do handbrake
float	dt	time since lsat call in ms

Code

```
641 function Drivable:updateVehiclePhysics(axisForward, axisSide, doHandbrake, dt)
642 local spec = self.spec_drivable
643
644 axisForward = axisForward
645 axisSide = self:getSteeringDirection() * axisSide
646
647 local acceleration = 0
648
649 if self:getIsMotorStarted() and self:getMotorStartTime() <= g_currentMission.time then
650 acceleration = axisForward
651 if math.abs(acceleration) > 0.8 then
652 self:setCruiseControlState(Drivable.CRUISECONTROL_STATE_OFF, true)
653 end
654 if spec.cruiseControl.state ~= Drivable.CRUISECONTROL_STATE_OFF then
655 acceleration = 1.0
656 end
657 end
658 if not self:getCanMotorRun() then
659 acceleration = 0
660 if self:getIsMotorStarted() then
661 self:stopMotor()
662 end
663 end
664
665 -- only update steering if a player is in the vehicle
666 if self.getIsControlled ~= nil and self:getIsControlled() then
```

```
667 local targetRotatedTime = 0
668 if self.maxRotTime ~= nil and self.minRotTime ~= nil then
669 if axisSide < 0 then
670 -- 0 to maxRotTime
671 targetRotatedTime = math.min(-self.maxRotTime * axisSide,
    self.maxRotTime)
672 else
673 -- 0 to minRotTime
674 targetRotatedTime = math.max(self.minRotTime * axisSide,
    self.minRotTime)
675 end
676 end
677 self.rotatedTime = targetRotatedTime
678 end
679
680 if self.firstTimeRun then
681 if self.spec_wheels ~= nil then
682 WheelsUtil.updateWheelsPhysics(self, dt,
    self.lastSpeedReal*self.movingDirection, acceleration,
    doHandbrake, g_currentMission.missionInfo.stopAndGoBraking)
683 end
684 end
685
686 return acceleration
687 end
```

DynamicallyLoadedParts

Description

loadDynamicallyPartsFromXML

Description

Load dynamically loaded parts from xml and link them to target

Definition

loadDynamicallyPartsFromXML(table dynamicallyLoadedPart, integer xmlFile, string key)

Arguments

table	dynamicallyLoadedPart	dynamically loaded part
integer	xmlFile	id of xml object
string	key	key

Return Values

boolean success success

Code

```
58 function
   DynamicallyLoadedParts:loadDynamicallyPartsFromXML(dynamicallyLoadedPart,
   xmlFile, key)
59 local filename = getXMLString(xmlFile, key .. "#filename")
```

```
60  if filename ~= nil then
61  dynamicallyLoadedPart.filename = filename
62  local i3dNode = g_i3DManager:loadSharedI3DFile(filename,
    self.baseDirectory, false, false, false)
63  if i3dNode ~= 0 then
64  local node = I3DUtil.indexToObject(i3dNode,
    Utils.getNoNil(getXMLString(xmlFile, key .. "#node"), "0|0"),
    self.i3dMappings)
65  local linkNode = I3DUtil.indexToObject(self.components,
    Utils.getNoNil(getXMLString(xmlFile, key .. "#linkNode"), "0>"),
    self.i3dMappings)
66  if linkNode == nil then
67  g_logManager:xmlWarning(self.configFileName, "Failed to load
    dynamicallyLoadedPart '%s'. Unable to find linkNode", key)
68  return false
69  end
70
71  local x,y,z = StringUtil.getVectorFromString(getXMLString(xmlFile, key ..
    "#position"))
72  if x ~= nil and y ~= nil and z ~= nil then
73  setTranslation(node, x,y,z)
74  end
75
76  local rotationNode = I3DUtil.indexToObject(i3dNode, getXMLString(xmlFile,
    key .. "#rotationNode"), self.i3dMappings) or node
77  local rotX, rotY, rotZ =
    StringUtil.getVectorFromString(getXMLString(xmlFile, key .. "#rotation"))
78  if rotX ~= nil and rotY ~= nil and rotZ ~= nil then
79  rotX = MathUtil.degToRad(rotX)
80  rotY = MathUtil.degToRad(rotY)
81  rotZ = MathUtil.degToRad(rotZ)
82  setRotation(rotationNode, rotX, rotY, rotZ)
83  end
84
85  local shaderParameterName = getXMLString(xmlFile, key ..
    "#shaderParameterName")
86  local x,y,z,w = StringUtil.getVectorFromString(getXMLString(xmlFile, key
    .. "#shaderParameter"))
87  if shaderParameterName ~= nil and x ~= nil and y ~= nil and z ~= nil and
    w ~= nil then
88  setShaderParameter(node, shaderParameterName, x, y, z, w, false)
89  end
90
```

```
91  link(linkNode, node)
92  delete(i3dNode)
93  return true
94  end
95  end
96
97  return false
98  end
```

Enterable
Description

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
226 function Enterable:onDelete()
227 local spec = self.spec_enterable
228
229 if spec.vehicleCharacter ~= nil then
230 spec.vehicleCharacter:delete()
231 end
232 for _, camera in ipairs(spec.cameras) do
233 camera:delete()
234 end
235
236 g_currentMission:removeEnterableVehicle(self)
237 g_currentMission:removeInteractiveVehicle(self)
238 end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId
integer connection connection

Code

```
244 function Enterable:onReadStream(streamId, connection)
245 local isControlled = streamReadBool(streamId)
246 if isControlled then
247 local playerStyle = PlayerStyle:new()
```

```
248 playerStyle:readStream(streamId, connection)
249 local farmId = streamReadUIntN(streamId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
250 self:enterVehicle(false, playerStyle, farmId)
251 end
252 end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
258 function Enterable:onWriteStream(streamId, connection)
259 local spec = self.spec_enterable
260 if streamWriteBool(streamId, spec.isControlled) then
261 spec.playerStyle:writeStream(streamId, connection)
262 streamWriteUIntN(streamId, spec.controllerFarmId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
263 end
264 end
```

saveStatsToXMLFile

Description

Returns string with name of controller for game stats xml

Definition

saveStatsToXMLFile()

Return Values

string statesAttributes states attributes

Code

```
269 function Enterable:saveStatsToXMLFile(xmlFile, key)
270 local spec = self.spec_enterable
271 if spec.isControlled and spec.playerStyle ~= nil and
    spec.playerStyle.playerName ~= nil then
272 setXMLString(xmlFile, key.."#controller",
    HTMLUtil.encodeToHTML(tostring(spec.playerStyle.playerName)))
273 end
274 return nil
275 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
282  function Enterable:onUpdate(dt, isActiveForInput, isSelected)
283  local spec = self.spec_enterable
284  for _,modifier in ipairs(spec.characterTargetNodeModifiers) do
285  self:updateCharacterTargetNodeModifier(dt, modifier)
286  end
287
288  if self:getIsControlled() then
289  if self.isClient then
290  if spec.vehicleCharacter ~= nil and
     spec.vehicleCharacter.isCharacterLoaded then
291  spec.vehicleCharacter:update(dt)
292  end
293
294  if self:getIsAdditionalCharacterActive() ~=
     spec.additionalCharacterActive then
295  spec.additionalCharacterActive = not
     spec.additionalCharacterActive
296
297  local character = self:getVehicleCharacter()
298  if character ~= nil then
299  local node = spec.defaultCharacterNode
300  local targets = spec.defaultCharacterTargets
301  if spec.additionalCharacterActive then
302  targets = spec.additionalCharacterTargets
303  node =spec.additionalCharacterNode
304  end
305
306  character:setIKChainTargets(targets)
307
308  character.characterNode = node
309  character.characterDistanceRefNode = node
310  link(node, character.skeletonThirdPerson)
311  character.visualInformation:linkProtectiveWear(node)
312  end
313  end
```

```
314 end
315
316 self:getRootVehicle():raiseActive()
317 end
318 end
```

onPostUpdate

Description

Called on postUpdate

Definition

onPostUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
325 function Enterable:onPostUpdate(dt, isActiveForInput, isSelected)
326 local spec = self.spec_enterable
327
328 if self.isClient then
329 if spec.isEntered and spec.vehicleCharacter ~= nil then
330 if spec.vehicleCharacter.characterSpineNode ~= nil and
    spec.vehicleCharacter.characterSpineSpeedDepended then
331 spec.vehicleCharacter:setSpineDirty(self.lastSpeedAcceleration)
332 end
333 end
334
335 if self.firstTimeRun then
336 if self:getIsEntered() then
337 spec.activeCamera:update(dt)
338 if not g_currentMission.hasSpecialCamera then
339 setCamera(spec.activeCamera.cameraNode)
340 end
341 end
342
343 -- update character visiblity
344 if self:getAllowCharacterVisibilityUpdate() then
345 if spec.vehicleCharacter ~= nil then
346 spec.vehicleCharacter:updateVisibility()
347 end
348 end
349 end
```

```
350
351 if self:getIsControlled() then
352 -- do mirror checks
353 if spec.activeCamera ~= nil and spec.activeCamera.useMirror then
354 self:setMirrorVisible(true)
355 end
356 end
357 end
358 end
```

enterVehicle

Description

Enter vehicle

Definition

enterVehicle(boolean isControlling, integer playerIndex, integer playerColorIndex)

Arguments

boolean	isControlling	is controlling vehicle
integer	playerIndex	index of player who enters the vehicle
integer	playerColorIndex	index of player color

Code

```
554 function Enterable:enterVehicle(isControlling, playerStyle, farmId)
555 local spec = self.spec_enterable
556
557 self:raiseActive()
558
559 spec.isControlled = true
560 spec.isEntered = isControlling
561 spec.playerStyle = playerStyle
562 spec.canUseEnter = false
563 spec.controllerFarmId = farmId
564
565 g_currentMission.controlledVehicles[self] = self
566
567 if spec.forceSelectionOnEnter then
568 local rootAttacherVehicle = self:getRootVehicle()
569 if rootAttacherVehicle ~= self then
570 rootAttacherVehicle:setSelectedImplementByObject(self)
571 end
572 end
573
574 if spec.isEntered then
575 -- if head tracking is available we want to use the first indoor camera
```

```
576 if g_gameSettings:getValue("isHeadTrackingEnabled") and isHeadTrackingAvailable() then
577 for i,camera in pairs(spec.cameras) do
578 if camera.isInside then
579 spec.camIndex = i
580 break
581 end
582 end
583 end
584
585 if g_gameSettings:getValue("resetCamera") then
586 spec.camIndex = 1
587 end
588 self:setActiveCameraIndex(spec.camIndex)
589 end
590
591 if not self:getIsAIActive() then
592 local playerModel = g_playerModelManager:getPlayerModelByIndex(spec.playerStyle.selectedMode
593 self:setVehicleCharacter(playerModel.xmlFilename, spec.playerStyle)
594
595 if spec.enterAnimation ~= nil and self.playAnimation ~= nil then
596 self:playAnimation(spec.enterAnimation, 1, nil, true)
597 end
598 end
599
600 SpecializationUtil.raiseEvent(self, "onEnterVehicle", isControlling)
601
602 -- activate actionEvents
603 if self.isClient then
604 self:requestActionEventUpdate()
605 end
606
607 if self.isServer and not isControlling and g_currentMission.trafficSyste
    nil and g_currentMission.trafficSystem.trafficSystemId ~= 0 then
608 addTrafficSystemPlayer(g_currentMission.trafficSystem.trafficSystemId,
    self.components[1].node)
609 end
610
611 self:activate()
612 end
```

leaveVehicle

Description

Leave vehicle

Definition

leaveVehicle()

Code

```
function Enterable:leaveVehicle()
local spec = self.spec_enterable

g_currentMission:removePauseListeners(self)

if spec.activeCamera ~= nil and spec.isEntered then
spec.activeCamera:onDeactivate()
g_soundManager:setIsIndoor(false)

local nC, blurEnd, fC, fBlurStart, fBlurEnd = unpack(spec.dofParams)
setDoFparams(nC, blurEnd, fC, fBlurStart, fBlurEnd)
end

if self.spec_rideable ~= nil then
-- To be called before "spec.vehicleCharacter:delete()"
self:setEquipmentVisibility(false)
self:unlinkReins()
end

spec.isControlled = false
spec.isEntered = false
spec.playerIndex = 0
spec.playerColorIndex = 0
spec.canUseEnter = true
spec.controllerFarmId = 0

g_currentMission.controlledVehicles[self] = nil
g_currentMission:setLastInteractionTime(200)

if spec.vehicleCharacter ~= nil and
self:getDisableVehicleCharacterOnLeave() then
spec.vehicleCharacter:delete()
end

if spec.enterAnimation ~= nil and self.playAnimation ~= nil then
```

```
650 self:playAnimation(spec.enterAnimation, -1, nil, true)
651 end
652
653 self:setMirrorVisible(false)
654
655 SpecializationUtil.raiseEvent(self, "onLeaveVehicle")
656
657 -- deactivate actionEvents
658 if self.isClient then
659 self:requestActionEventUpdate()
660 end
661
662 if self.isServer and not spec.isEntered and
    g_currentMission.trafficSystem ~= nil and
    g_currentMission.trafficSystem.trafficSystemId ~= 0 then
663 removeTrafficSystemPlayer(g_currentMission.trafficSystem.trafficSystemId
    self.components[1].node)
664 end
665
666 self:deactivate()
667 end
```

setActiveCameraIndex

Description

Change active camera index

Definition

setActiveCameraIndex(integer index)

Arguments

integer index index of camera to set

Code

```
672 function Enterable:setActiveCameraIndex(index)
673 local spec = self.spec_enterable
674
675 if spec.activeCamera == nil or not spec.activeCamera.isInside
    then
676 spec.dofParams = {getDoFparams()}
677 end
678
679 if spec.activeCamera ~= nil then
680 spec.activeCamera:onDeactivate()
681 end
682 spec.camIndex = index
683 if spec.camIndex > spec.numCameras then
```

```
684 spec.camIndex = 1
685 end
686 local activeCamera = spec.cameras[spec.camIndex]
687 spec.activeCamera = activeCamera
688 activeCamera:onActivate()
689
690 g_soundManager:setIsIndoor(not activeCamera.useOutdoorSounds)
691 self:setMirrorVisible(activeCamera.useMirror)
692
693 local nC, blurEnd, fC, fBlurStart, fBlurEnd =
    unpack(spec.dofParams)
694 if activeCamera.isInside then
695 setDoFparams(nC, 0.6, fC, fBlurStart, fBlurEnd)
696 else
697 setDoFparams(nC, blurEnd, fC, fBlurStart, fBlurEnd)
698 end
699
700 SpecializationUtil.raiseEvent(self, "onCameraChanged",
    activeCamera, spec.camIndex)
701 end
```

addToolCameras

Description

Add cameras from tool

Definition

addToolCameras(table cameras)

Arguments

table cameras cameras to add

Code

```
706 function Enterable:addToolCameras(cameras)
707 local spec = self.spec_enterable
708
709 for _,toolCamera in pairs(cameras) do
710 table.insert(spec.cameras, toolCamera)
711 end
712 spec.numCameras = #spec.cameras
713 end
```

removeToolCameras

Description

Remove cameras from tool

Definition

removeToolCameras(table cameras)

Arguments

table cameras cameras to remove

Code

```
718  function Enterable:removeToolCameras(cameras)
719  local spec = self.spec_enterable
720
721  for _,toolCamera in pairs(cameras) do
722  for j,camera in pairs(spec.cameras) do
723  if toolCamera == camera then
724  table.remove(spec.cameras, j)
725  break
726  end
727  end
728  end
729  spec.numCameras = #spec.cameras
730  if spec.activeCamera ~= nil then
731  spec.activeCamera:onDeactivate()
732  end
733  if spec.camIndex > spec.numCameras then
734  spec.camIndex = 1
735  end
736  local cameraId = getCamera()
737  self:setActiveCameraIndex(spec.camIndex)
738  setCamera(cameraId)
739  end
```

drawUIInfo

Description

Called on ui info draw, renders nicknames in multiplayer

Definition

drawUIInfo()

Code

```
885  function Enterable:drawUIInfo(superFunc)
886  local spec = self.spec_enterable
887
888  superFunc(self)
889
890  if not spec.isEntered and self.isClient and self:getIsActive()
     and spec.isControlled and not g_gui:getIsGuiVisible() and not
     g_flightAndNoHUDKeysEnabled then
891  local x,y,z = getWorldTranslation(spec.nicknameRendering.node)
892  local x1,y1,z1 = getWorldTranslation(getCamera())
```

```
local distSq = MathUtil.vector3LengthSq(x-x1,y-y1,z-z1)
if distSq <= 100*100 then
x = x + spec.nicknameRendering.offset[1]
y = y + spec.nicknameRendering.offset[2]
z = z + spec.nicknameRendering.offset[3]

Utils.renderTextAtWorldPosition(x,y,z, self:getControllerName(), getCorrectTextSize(0.02), 0)
end
end
end
```

getDistanceToNode

Description

Returns distance between given object and enterReferenceNode

Definition

getDistanceToNode(integer object)

Arguments

integer object id of object

Return Values

float distance distance

Code

```
function Enterable:getDistanceToNode(superFunc, node)
local spec = self.spec_enterable

local superDistance = superFunc(self, node)

if spec == nil or spec.enterReferenceNode == nil then
return superDistance
end

local px, _, pz = getWorldTranslation(node)
local vx, _, vz = getWorldTranslation(spec.enterReferenceNode)
local distance = MathUtil.vector2Length(px-vx, pz-vz)

if distance < spec.interactionRadius and distance < superDistance then
self.interactionFlag = Vehicle.INTERACTION_FLAG_ENTERABLE
return distance
end

return superDistance
```

```
928 end
```

getInteractionHelp

Description

Returns interaction help text

Definition

getInteractionHelp()

Return Values

string text text

Code

```
933 function Enterable:getInteractionHelp(superFunc)
934 if self.interactionFlag == Vehicle.INTERACTION_FLAG_ENTERABLE then
935 return g_i18n:getText("action_enter")
936 else
937 return superFunc(self)
938 end
939 end
```

interact

Description

Interact

Definition

interact()

Code

```
943 function Enterable:interact(superFunc)
944 if self.interactionFlag == Vehicle.INTERACTION_FLAG_ENTERABLE then
945 g_currentMission:requestToEnterVehicle(self)
946 else
947 superFunc(self)
948 end
949 end
```

getIsEnterable

Description

Get whether current player can enter this vehicle
Only works when isClient

Definition

getIsEnterable()

FertilizingCultivator

Description

Class for all fertilizing cultivators

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17  function FertilizingCultivator.prerequisitesPresent(specializations)
18  return SpecializationUtil.hasSpecialization(Cultivator, specializations) and SpecializationUtil.hasSpecialization(Sprayer, specializations)
19  end
```

FertilizingSowingMachine

Description

Class for all fertilizing sowingMachine

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17  function FertilizingSowingMachine.prerequisitesPresent(specializations)
18  return SpecializationUtil.hasSpecialization(SowingMachine, specializations) and SpecializationUtil.hasSpecialization(Sprayer, specializations)
19  end
```

FillTriggerVehicle

Description

Class for vehicles with a fill trigger such as fuel trailers

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17  function FillTriggerVehicle.prerequisitesPresent(specializations)
18  return SpecializationUtil.hasSpecialization(FillUnit,
    specializations)
19  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
33  function FillTriggerVehicle:onLoad(savegame)
34  local spec = self.spec_fillTriggerVehicle
35
36  local triggerNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile,
    "vehicle.fillTriggerVehicle#triggerNode"), self.i3dMappings)
37  if triggerNode ~= nil then
38  spec.fillUnitIndex = Utils.getNoNil(getXMLInt(self.xmlFile,
    "vehicle.fillTriggerVehicle#fillUnitIndex"), 1)
39  spec.litersPerSecond = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.fillTriggerVehicle#litersPerSecond"), 50)
40  spec.fillTrigger = FillTrigger:new(triggerNode, self,
    spec.fillUnitIndex, spec.litersPerSecond)
41  end
42  end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
46  function FillTriggerVehicle:onDelete()
47  local spec = self.spec_fillTriggerVehicle
48
49  if spec.fillTrigger ~= nil then
50  spec.fillTrigger:delete()
51  spec.fillTrigger = nil
52  end
53  end
```

FillUnit
Description
addFillUnitTrigger
Description

Adds fill trigger

Definition

addFillUnitTrigger(table trigger, integer fillTypeIndex)

Arguments

table trigger trigger
integer fillTypeIndex fillTypeIndex

Code

```
1484  function FillUnit:addFillUnitTrigger(trigger, fillTypeIndex, fillUnitIndex)
1485      local spec = self.spec_fillUnit
1486      if #spec.fillTrigger.triggers == 0 then
1487          g_currentMission:addActivatableObject(spec.fillTrigger.activatable)
1488          spec.fillTrigger.activatable:setFillType(fillTypeIndex)
1489      end
1490      ListUtil.addElementToList(spec.fillTrigger.triggers, trigger)
1491      SpecializationUtil.raiseEvent(self, "onAddedFillUnitTrigger", fillTypeIndex, fillUnitIndex, #spec.fillTrigger.triggers)
1492  end
```

removeFillUnitTrigger
Description

Removes fill trigger

Definition

removeFillUnitTrigger(table trigger)

Arguments

table trigger trigger

Code

```
1497  function FillUnit:removeFillUnitTrigger(trigger)
1498      local spec = self.spec_fillUnit
1499      ListUtil.removeElementFromList(spec.fillTrigger.triggers, trigger)
1500
1501      if self.isServer and trigger == spec.fillTrigger.currentTrigger then
1502          self:setFillUnitIsFilling(false)
1503      end
1504
1505      if #spec.fillTrigger.triggers == 0 then
1506          g_currentMission:removeActivatableObject(spec.fillTrigger.activatable)
1507      end
1508
```

```
1509  SpecializationUtil.raiseEvent(self, "onRemovedFillUnitTrigger",
      #spec.fillTrigger.triggers)
1510  end
```

getFillLevelInformation

Description

Get fill level information

Definition

getFillLevelInformation(table fillLevelInformations)

Arguments

table fillLevelInformations fill level informations

Code

```
1628  function FillUnit:getFillLevelInformation(superFunc,
      fillLevelInformations)
1629  superFunc(self, fillLevelInformations)
1630
1631  local spec = self.spec_fillUnit
1632  for _,fillUnit in pairs(spec.fillUnits) do
1633  if fillUnit.capacity > 0 and fillUnit.showOnHud then
1634  local fillType = fillUnit.fillType
1635  if fillUnit.fillTypeToDisplay ~= FillType.UNKNOWN then
1636  fillType = fillUnit.fillTypeToDisplay
1637  end
1638
1639  local added = false
1640  for _,fillLevelInformation in pairs(fillLevelInformations) do
1641  if fillLevelInformation.fillType == fillType then
1642  fillLevelInformation.fillLevel = fillLevelInformation.fillLevel
      + fillUnit.fillLevel
1643  fillLevelInformation.capacity = fillLevelInformation.capacity +
      fillUnit.capacity
1644  added = true
1645  break
1646  end
1647  end
1648  if not added then
1649  table.insert(fillLevelInformations, {fillType=fillType,
      fillLevel=fillUnit.fillLevel, capacity=fillUnit.capacity})
1650  end
1651  end
1652  end
1653  end
```

Foldable

Description

isDetachAllowed

Description

Returns true if detach is allowed

Definition

isDetachAllowed()

Return Values

boolean detachAllowed detach is allowed

string warning [optional] warning text to display

Code

```
729 function Foldable:isDetachAllowed(superFunc)
730 local spec = self.spec_foldable
731
732 if spec.foldAnimTime > spec.detachingMaxLimit or
    spec.foldAnimTime < spec.detachingMinLimit then
733 return false, spec.unfoldWarning
734 end
735
736 if not spec.allowDetachingWhileFolding then
737 if spec.foldAnimTime ~= spec.foldMiddleAnimTime and
    (spec.foldAnimTime > 0 and spec.foldAnimTime < 1) then
738 return false, spec.unfoldWarning
739 end
740 end
741
742 return superFunc(self)
743 end
```

getAllowsLowering

Description

Returns true if tool can be lowered

Definition

getAllowsLowering()

Return Values

boolean detachAllowed detach is allowed

string warning [optional] warning text to display

Code

```
749 function Foldable:getAllowsLowering(superFunc)
750 local spec = self.spec_foldable
751
752 if spec.foldAnimTime > spec.loweringMaxLimit or spec.foldAnimTime
    < spec.loweringMinLimit then
753 return false, spec.unfoldWarning
```

```
754 end
755
756 return superFunc(self)
757 end
```

getIsInputAttacherActive

Description

Returns true if input attacher is active and can be used to attach

Definition

getIsInputAttacherActive(table inputAttacherJoint)

Arguments

table inputAttacherJoint input attacher joint

Return Values

boolean isActive input attacher is active

Code

```
884 function Foldable:getIsInputAttacherActive(superFunc,
    inputAttacherJoint)
885 if inputAttacherJoint.foldMinLimit ~= nil and
    inputAttacherJoint.foldMaxLimit ~= nil then
886 local foldAnimTime = self:getFoldAnimTime()
887 if foldAnimTime < inputAttacherJoint.foldMinLimit or foldAnimTime
    > inputAttacherJoint.foldMaxLimit then
888 return false
889 end
890 end
891
892 return superFunc(self, inputAttacherJoint)
893 end
```

FruitPreparer

Description

This is the specialization for tool that prepare the fruits for harvesting. E.g. potato haulm toppers and sugar beet defoliators

initSpecialization

Description

Called on specialization initializing

Definition

initSpecialization()

Code

```
15 function FruitPreparer.initSpecialization()
16 g_workAreaTypeManager:addWorkAreaType("fruitPreparer", false)
17 end
```

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
23  function FruitPreparer.prerequisitesPresent(specializations)
24  return SpecializationUtil.hasSpecialization(WorkArea,
    specializations) and
    SpecializationUtil.hasSpecialization(TurnOnVehicle,
    specializations)
25  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
51  function FruitPreparer:onLoad(savegame)
52  local spec = self.spec_fruitPreparer
53
54  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.turnOnAnimation#name",
    "vehicle.turnOnVehicle.turnedAnimation#name") -- FS15 to FS17
55  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.turnOnAnimation#speed",
    "vehicle.turnOnVehicle.turnedAnimation#turnOnSpeedScale") -- FS15
    to FS17
56
57  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.fruitPreparer#useReelStateToTurnOn")
    -- FS17 to FS19
58  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.fruitPreparer#onlyActiveWhenLowered") -- FS17 to FS19
59  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.vehicle.fruitPreparerSound",
    "vehicle.fruitPreparer.sounds.work") -- FS17 to FS19
60  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.turnedOnRotationNodes.turnedOnRotationNode",
    "vehicle.fruitPreparer.animationNodes.animationNode",
    "fruitPreparer") -- FS17 to FS19
```

```
61
62  if self.isClient then
63  spec.samples = {}
64  spec.samples.work = g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.fruitPreparer.sounds", "work", self.baseDirectory,
    self.components, 0, AudioGroup.VEHICLE, self.i3dMappings, self)
65
66  spec.animationNodes =
    g_animationManager:loadAnimations(self.xmlFile,
    "vehicle.fruitPreparer.animationNodes", self.components, self,
    self.i3dMappings)
67  end
68
69  spec.fruitType = FruitType.UNKNOWN
70  local fruitType = getXMLString(self.xmlFile,
    "vehicle.fruitPreparer#fruitType")
71  if fruitType ~= nil then
72  local desc = g_fruitTypeManager:getFruitTypeByName(fruitType)
73  if desc ~= nil then
74  spec.fruitType = desc.index
75
76  if self.setAIFruitRequirements ~= nil then
77  self:setAIFruitRequirements(desc.index,
    desc.minPreparingGrowthState, desc.maxPreparingGrowthState)
78
79  local aiUsePreparedState = Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.fruitPreparer#aiUsePreparedState"), self.spec_cutter ~=
    nil)
80  if aiUsePreparedState then
81  self:addAIFruitRequirement(desc.index, desc.preparedGrowthState,
    desc.preparedGrowthState)
82  end
83  end
84  else
85  g_logManager:xmlWarning(self.configFileName, "Unable to find
    fruitType '%s' in fruitPreparer", fruitType)
86  end
87  else
88  g_logManager:xmlWarning(self.configFileName, "Missing fruitType in
    fruitPreparer")
89  end
90
91  spec.isWorking = false
```

```
92  end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
96  function FruitPreparer:onDelete()
97  if self.isClient then
98  local spec = self.spec_fruitPreparer
99  g_soundManager:deleteSamples(spec.samples)
100 g_animationManager:deleteAnimations(spec.animationNodes)
101 end
102 end
```

onReadUpdateStream

Description

Called on on update

Definition

onReadUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId	stream ID
integer timestamp	timestamp
table	connection connection

Code

```
109 function FruitPreparer:onReadUpdateStream(streamId, timestamp,
    connection)
110 if connection:getIsServer() then
111 local spec = self.spec_fruitPreparer
112 spec.isWorking = streamReadBool(streamId)
113 end
114 end
```

onWriteUpdateStream

Description

Called on on update

Definition

onWriteUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Code

```lua
function FruitPreparer:onWriteUpdateStream(streamId, connection, dirtyMask)
if not connection:getIsServer() then
local spec = self.spec_fruitPreparer
streamWriteBool(streamId, spec.isWorking)
end
end
```

onTurnedOn

Description

Called on turn on

Definition

onTurnedOn(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```lua
function FruitPreparer:onTurnedOn()
if self.isClient then
local spec = self.spec_fruitPreparer
g_soundManager:playSample(spec.samples.work)
g_animationManager:startAnimations(spec.animationNodes)
end
end
```

onTurnedOff

Description

Called on turn off

Definition

onTurnedOff(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```lua
function FruitPreparer:onTurnedOff()
if self.isClient then
local spec = self.spec_fruitPreparer
g_soundManager:stopSamples(spec.samples)
g_animationManager:stopAnimations(spec.animationNodes)
end
end
```

loadWorkAreaFromXML

Description

Loads work areas from xml

Definition

loadWorkAreaFromXML(table workArea, integer xmlFile, string key)

Arguments

table	workArea	workArea
integer	xmlFile	id of xml object
string	key	key

Return Values

boolean success success

Code

```
156 function FruitPreparer:loadWorkAreaFromXML(superFunc, workArea, xmlFile, key)
157 local retValue = superFunc(self, workArea, xmlFile, key)
158
159 if workArea.type == WorkAreaType.FRUITPREPARER then
160 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, key.."#dropStartIndex", key..".fruitPreparer#dropWorkAreaIndex") -- FS17 to FS19
161 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, key.."#dropWidthIndex", key..".fruitPreparer#dropWorkAreaIndex") -- FS17 to FS19
162 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, key.."#dropHeightIndex", key..".fruitPreparer#dropWorkAreaIndex") -- FS17 to FS19
163
164 workArea.dropWorkAreaIndex = getXMLInt(xmlFile, key..".fruitPreparer#dropWorkAreaIndex")
165 end
166
167 return retValue
168 end
```

getDoGroundManipulation

Description

Returns if tool does ground manipulation

Definition

getDoGroundManipulation()

Return Values

boolean doGroundManipulation do ground manipulation

Code

```
173 function FruitPreparer:getDoGroundManipulation(superFunc)
174 local spec = self.spec_fruitPreparer
175 return superFunc(self) and spec.isWorking
176 end
```

doCheckSpeedLimit

Description

Returns if speed limit should be checked

Definition

doCheckSpeedLimit()

Return Values

boolean checkSpeedlimit check speed limit

Code

```
181  function FruitPreparer:doCheckSpeedLimit(superFunc)
182  return superFunc(self) or (self:getIsTurnedOn() and
     (self.getIsImplementChainLowered == nil or
     self:getIsImplementChainLowered()))
183  end
```

getDefaultSpeedLimit

Description

Returns default speed limit

Definition

getDefaultSpeedLimit()

Return Values

float speedLimit speed limit

Code

```
192  function FruitPreparer.getDefaultSpeedLimit()
193  return 15
194  end
```

GroundAdjustedNodes

Description

groundAdjustRaycastCallback

Description

Raycast callback

Definition

groundAdjustRaycastCallback(integer transformId, float x, float y, float z, float distance)

Arguments

integer	transformId	id raycasted object
float	x	x raycast position
float	y	y raycast position
float	z	z raycast position
float	distance	distance to raycast position

Code

```
169  function
     GroundAdjustedNodes:groundAdjustRaycastCallback(transformId, x,
     y, z, distance)
170  self.lastRaycastDistance = distance;
171  end;
```

Honk

Description

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
41  function Honk:onDelete()
42  if self.isClient then
43  local spec = self.spec_honk
44  g_soundManager:deleteSample(spec.sample)
45  end
46  end
```

onLeaveVehicle

Description

Called on leaving the vehicle

Definition

onLeaveVehicle()

Code

```
66  function Honk:onLeaveVehicle()
67  self:playHonk(false, true)
68  end
```

HookLiftContainer

Description

Class for hooklift containers

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17  function HookLiftContainer.prerequisitesPresent(specializations)
18  return SpecializationUtil.hasSpecialization(AnimatedVehicle,
    specializations) and
    SpecializationUtil.hasSpecialization(Attachable, specializations)
19  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
36  function HookLiftContainer:onLoad(savegame)
37  local spec = self.spec_hookLiftContainer
38
39  spec.tiltContainerOnDischarge =
    Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.hookLiftContainer#tiltContainerOnDischarge"), true)
40  end
```

HookLiftTrailer

Description

Class for hooklift trailers

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17  function HookLiftTrailer.prerequisitesPresent(specializations)
18  return SpecializationUtil.hasSpecialization(AnimatedVehicle,
    specializations) and
    SpecializationUtil.hasSpecialization(Foldable, specializations)
19  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
45  function HookLiftTrailer:onLoad(savegame)
46  local spec = self.spec_hookLiftTrailer
47
48  spec.jointLimits = AnimCurve:new(linearInterpolatorN)
49  local i = 0
50  while true do
51  local key =
    string.format("vehicle.hookLiftTrailer.jointLimits.key(%d)", i)
```

```
52  if not hasXMLProperty(self.xmlFile, key) then
53  if i == 0 then
54  spec.jointLimits = nil
55  end
56  break
57  end
58  local t = getXMLFloat(self.xmlFile, key.."#time")
59  local rx,ry,rz =
    StringUtil.getVectorFromString(getXMLString(self.xmlFile,
    key.."#rotLimit"))
60  local tx,ty,tz =
    StringUtil.getVectorFromString(getXMLString(self.xmlFile,
    key.."#transLimit"))
61  rx = math.rad(Utils.getNoNil(rx, 0))
62  ry = math.rad(Utils.getNoNil(ry, 0))
63  rz = math.rad(Utils.getNoNil(rz, 0))
64  tx = Utils.getNoNil(tx, 0)
65  ty = Utils.getNoNil(ty, 0)
66  tz = Utils.getNoNil(tz, 0)
67  spec.jointLimits:addKeyframe({rx, ry, rz, tx, ty, tz, time = t})
68  i = i +1
69  end
70
71  spec.refAnimation = Utils.getNoNil(getXMLString(self.xmlFile,
    "vehicle.hookLiftTrailer.jointLimits#refAnimation"), "unfoldHand")
72  spec.unloadingAnimation =
    Utils.getNoNil(getXMLString(self.xmlFile,
    "vehicle.hookLiftTrailer.unloadingAnimation#name"), "unloading")
73  spec.unloadingAnimationSpeed =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.hookLiftTrailer.unloadingAnimation#speed"), 1)
74  spec.unloadingAnimationReverseSpeed =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.hookLiftTrailer.unloadingAnimation#reverseSpeed"), -1)
75
76  spec.texts = {}
77  spec.texts.unloadContainer =
    Utils.getNoNil(getXMLString(self.xmlFile,
    "vehicle.hookLiftTrailer.texts#unloadContainer"),
    "unload_container")
78  spec.texts.loadContainer =
    Utils.getNoNil(getXMLString(self.xmlFile,
    "vehicle.hookLiftTrailer.texts#loadContainer"), "load_container")
79  spec.texts.unloadArm = Utils.getNoNil(getXMLString(self.xmlFile,
    "vehicle.hookLiftTrailer.texts#unloadArm"), "unload_arm")
```

```
80  spec.texts.loadArm = Utils.getNoNil(getXMLString(self.xmlFile,
    "vehicle.hookLiftTrailer.texts#loadArm"), "load_arm")
81  end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
88  function HookLiftTrailer:onUpdateTick(dt, isActiveForInput,
    isSelected)
89  local spec = self.spec_hookLiftTrailer
90
91  if spec.attachedContainer ~= nil then
92  local animTime = self:getAnimationTime(spec.refAnimation)
93  spec.attachedContainer.object.allowsDetaching = animTime > 0.95
94
95  if self:getIsAnimationPlaying(spec.refAnimation) and
    spec.jointLimits ~= nil and not
    spec.attachedContainer.implement.attachingIsInProgress then
96  local v = spec.jointLimits:get(animTime)
97  for i=1,3 do
98  setJointRotationLimit(spec.attachedContainer.jointIndex, i-1,
    true, -v[i], v[i])
99  setJointTranslationLimit(spec.attachedContainer.jointIndex, i+2,
    true, -v[i+3], v[i+3])
100 end
101 end
102 end
103 end
```

onPostAttachImplement

Description

Called on attaching a implement

Definition

onPostAttachImplement(table implement)

Arguments

table	implement	implement to attach

Code

```lua
function HookLiftTrailer:onPostAttachImplement(attachable, inputJointDescIndex, jointDescIndex)
local spec = self.spec_hookLiftTrailer

local attacherJoint = attachable:getActiveInputAttacherJoint()
if attacherJoint ~= nil then
if attacherJoint.jointType == AttacherJoints.JOINTTYPE_HOOKLIFT then
local jointDesc = self:getAttacherJointByJointDescIndex(jointDescIndex)
spec.attachedContainer = {}
spec.attachedContainer.jointIndex = jointDesc.jointIndex
spec.attachedContainer.implement = self:getImplementByObject(attachable)
spec.attachedContainer.object = attachable

local foldableSpec = self.spec_foldable
foldableSpec.posDirectionText = spec.texts.unloadContainer
foldableSpec.negDirectionText = spec.texts.loadContainer
end
end
end
```

onPreDetachImplement

Description

Called on detaching a implement

Definition

onPreDetachImplement(integer implementIndex)

Arguments

integer implementIndex index of implement to detach

Code

```lua
function HookLiftTrailer:onPreDetachImplement(implement)
local spec = self.spec_hookLiftTrailer
if spec.attachedContainer ~= nil then
if implement == spec.attachedContainer.implement then
local foldableSpec = self.spec_foldable
foldableSpec.posDirectionText = spec.texts.unloadArm
foldableSpec.negDirectionText = spec.texts.loadArm
spec.attachedContainer = nil
end
end
end
```

startTipping

Description

Called on start tipping

Definition

startTipping()

Code

```
144  function HookLiftTrailer:startTipping()
145  local spec = self.spec_hookLiftTrailer
146  self:playAnimation(spec.unloadingAnimation,
     spec.unloadingAnimationSpeed,
     self:getAnimationTime(spec.unloadingAnimation), true)
147  end
```

stopTipping

Description

Called on stop tipping

Definition

stopTipping()

Code

```
151  function HookLiftTrailer:stopTipping()
152  local spec = self.spec_hookLiftTrailer
153  self:playAnimation(spec.unloadingAnimation,
     spec.unloadingAnimationReverseSpeed,
     self:getAnimationTime(spec.unloadingAnimation), true)
154  end
```

getIsTippingAllowed

Description

Returns if tipping is allowed

Definition

getIsTippingAllowed()

Return Values

boolean tippingAllowed tipping is allowed

Code

```
159  function HookLiftTrailer:getIsTippingAllowed()
160  local spec = self.spec_hookLiftTrailer
161  return self:getAnimationTime(spec.refAnimation) == 0
162  end
```

getIsFoldAllowed

Description

Returns if fold is allowed

Definition

getIsFoldAllowed()

Return Values

boolean allowsFold allows folding

Code

```lua
function HookLiftTrailer:getIsFoldAllowed(superFunc, direction, onAiTurnOn)
local spec = self.spec_hookLiftTrailer
if self:getAnimationTime(spec.unloadingAnimation) > 0 then
return false
end

return superFunc(self, direction, onAiTurnOn)
end
```

isDetachAllowed

Description

Returns true if detach is allowed

Definition

isDetachAllowed()

Return Values

boolean detachAllowed detach is allowed

Code

```lua
function HookLiftTrailer:isDetachAllowed(superFunc)
if self:getAnimationTime(self.spec_hookLiftTrailer.unloadingAnimation) == 0 then
return superFunc(self)
else
return false, nil
end
end
```

getDoConsumePtoPower

Description

Returns if should consume pto power

Definition

getDoConsumePtoPower()

Return Values

boolean consume consumePtoPower

Code

```lua
function HookLiftTrailer:getDoConsumePtoPower(superFunc)
local spec = self.spec_hookLiftTrailer
local doConsume = superFunc(self)

return doConsume or self:getIsAnimationPlaying(spec.refAnimation) or self:getIsAnimationPlaying(spec.unloadingAnimation)
end
```

getPtoRpm

Description

Returns rpm of pto

Definition

getPtoRpm()

Return Values

float rpm rpm of pto

Code

```
205 function HookLiftTrailer:getPtoRpm(superFunc)
206 local spec = self.spec_hookLiftTrailer
207 local rpm = superFunc(self)
208
209 if self:getIsAnimationPlaying(spec.refAnimation) or
    self:getIsAnimationPlaying(spec.unloadingAnimation) then
210 return self.spec_powerConsumer.ptoRpm
211 else
212 return rpm
213 end
214 end
```

Leveler

Description

Class for all Levelers

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17 function Leveler.prerequisitesPresent(specializations)
18 return SpecializationUtil.hasSpecialization(FillUnit,
   specializations)
19 end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
35 function Leveler:onLoad(savegame)
```

```lua
local spec = self.spec_leveler

XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "vehicle.leveler.levelerNode#index", "vehicle.leveler.levelerNode#node") -- FS17
XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "vehicle.levelerEffects", "vehicle.leveler.effects") -- FS17

spec.nodes = {}
local i = 0
while true do
local key = string.format("vehicle.leveler.levelerNode(%d)", i)
if not hasXMLProperty(self.xmlFile, key) then
break
end

local levelerNode = {}
if self:loadLevelerNodeFromXML(levelerNode, self.xmlFile, key) then
table.insert(spec.nodes, levelerNode)
end
i = i + 1
end

spec.pickUpDirection = Utils.getNoNil(getXMLInt(self.xmlFile, "vehicle.leveler.pickUpDirection"), 1.0)
spec.fillUnitIndex = getXMLInt(self.xmlFile, "vehicle.leveler#fillUnitIndex")
if not self:getFillUnitExists(spec.fillUnitIndex) then
g_logManager:xmlWarning(self.configFileName, "Unknown fillUnitIndex '%s' for leveler", tostring(spec.fillUnitIndex))
end

if self.isClient then
spec.effects = g_effectManager:loadEffect(self.xmlFile, "vehicle.leveler.effects", self.components, self, self.i3dMappings)
end
end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
72  function Leveler:onUpdate(dt, isActiveForInput, isSelected)
73  local spec = self.spec_leveler
74
75  if self.isClient then
76  local fillType = self:getFillUnitFillType(spec.fillUnitIndex)
77  local fillLevel = self:getFillUnitFillLevel(spec.fillUnitIndex)
78  local visible = fillLevel >
    2*g_densityMapHeightManager:getMinValidLiterValue(fillType)
79  if visible and fillType ~= FillType.UNKNOWN then
80  g_effectManager:setFillType(spec.effects, fillType)
81  g_effectManager:startEffects(spec.effects)
82
83  local fillPercentage = fillLevel / self:getFillUnitCapacity(spec.fillUni
84  for _, effect in pairs(spec.effects) do
85  effect:setFillLevel(fillPercentage)
86  effect:setLastVehicleSpeed(self.movingDirection * self:getLastSpeed())
87  end
88  else
89  g_effectManager:stopEffects(spec.effects)
90  end
91  end
92
93  if self.isServer then
94  for _, levelerNode in pairs(spec.nodes) do
95  local x0,y0,z0 = localToWorld(levelerNode.node, -levelerNode.width, 0,
    levelerNode.maxDropDirOffset)
96  local x1,y1,z1 = localToWorld(levelerNode.node, levelerNode.width, 0,
    levelerNode.maxDropDirOffset)
97  if not
    g_farmlandManager:getIsOwnedByFarmAtWorldPosition(g_currentMission:getFa
    x0, z0) or
98  not
    g_farmlandManager:getIsOwnedByFarmAtWorldPosition(g_currentMission:getFa
    x1, z1) then
99  break
100 end
```

```
101
102 local pickedUpFillLevel = 0
103 local fillType = self:getFillUnitFillType(spec.fillUnitIndex)
104 local fillLevel = self:getFillUnitFillLevel(spec.fillUnitIndex)
105
106 if fillType == FillType.UNKNOWN or fillLevel < g_densityMapHeightManager:getMinValidLiterValue(fillType) + 0.001 then
107 local newFillType = DensityMapHeightUtil.getFillTypeAtLine(x0,y0,z0, x1, 0.5*levelerNode.maxDropDirOffset)
108 if newFillType ~= FillType.UNKNOWN and newFillType ~= fillType then
109 self:addFillUnitFillLevel(self:getOwnerFarmId(), spec.fillUnitIndex, -math.huge)
110 fillType = newFillType
111 end
112 end
113 local heightType = g_densityMapHeightManager:getDensityMapHeightTypeByFillTypeIndex(fillTyp
114
115 if fillType ~= FillType.UNKNOWN and heightType ~= nil then
116 local innerRadius = 0
117 local outerRadius = DensityMapHeightUtil.getDefaultMaxRadius(fillType)
118 local capacity = self:getFillUnitCapacity(spec.fillUnitIndex)
119
120 -- pick up at node
121 if self:getIsLevelerPickupNodeActive(levelerNode) then
122 if spec.pickUpDirection == self.movingDirection then
123 local sx,sy,sz = localToWorld(levelerNode.node, -levelerNode.width, 0, 0
124 local ex,ey,ez = localToWorld(levelerNode.node, levelerNode.width, 0, 0)
125
126 fillLevel = self:getFillUnitFillLevel(spec.fillUnitIndex)
127 local delta = -(capacity-fillLevel)
128 local numHeightLimitChecks = levelerNode.numHeightLimitChecks
129 if numHeightLimitChecks > 0 then
130 local movementY = 0
131 for i=0,numHeightLimitChecks do
132 local t = i/numHeightLimitChecks
133 local xi = sx + (ex-sx)*t
134 local yi = sy + (ey-sy)*t
135 local zi = sz + (ez-sz)*t
136 local hi = DensityMapHeightUtil.getHeightAtWorldPos(xi,yi,zi)
137 movementY = math.max(movementY, hi-0.05 - yi) -- limit to 5cm below surf
```

```
138 end
139 if movementY > 0 then
140 sy = sy+movementY
141 ey = ey+movementY
142 end
143 end
144
145 levelerNode.lastPickUp, levelerNode.lineOffsetPickUp =
    DensityMapHeightUtil.tipToGroundAroundLine(self, delta, fillType, sx,sy-
    ex,ey-0.1,ez, innerRadius, outerRadius, levelerNode.lineOffsetPickUp, tr
    nil)
146
147 if levelerNode.lastPickUp < 0 then
148 self:addFillUnitFillLevel(self:getOwnerFarmId(), spec.fillUnitIndex, -
    levelerNode.lastPickUp, fillType, ToolType.UNDEFINED, nil)
149 pickedUpFillLevel = levelerNode.lastPickUp
150 end
151 end
152 end
153
154 -- drop at node
155 fillLevel = self:getFillUnitFillLevel(spec.fillUnitIndex)
156 if fillLevel > 0 then
157 local f = (fillLevel/capacity)
158 local width = MathUtil.lerp(levelerNode.minDropWidth, levelerNode.maxDro
    f)
159
160 local sx,sy,sz = localToWorld(levelerNode.node, -width, 0, 0)
161 local ex,ey,ez = localToWorld(levelerNode.node, width, 0, 0)
162
163 local yOffset = -0.1 -0.05
164
165 levelerNode.lastDrop1, levelerNode.lineOffsetDrop1 =
    DensityMapHeightUtil.tipToGroundAroundLine(self, fillLevel, fillType,
    sx,sy+yOffset,sz, ex,ey+yOffset,ez, innerRadius, outerRadius,
    levelerNode.lineOffsetDrop1, true, tipOcclusionAreas)
166 if levelerNode.lastDrop1 > 0 then
167 self:addFillUnitFillLevel(self:getOwnerFarmId(), spec.fillUnitIndex, -
    levelerNode.lastDrop1, fillType, ToolType.UNDEFINED, nil)
168 end
169 end
170
```

```
171 -- drop further at front
172 fillLevel = self:getFillUnitFillLevel(spec.fillUnitIndex)
173
174 if fillLevel > 0 then
175 local f = (fillLevel/capacity)
176 local width = MathUtil.lerp(levelerNode.minDropWidth, levelerNode.maxDro
    f)
177
178 local yOffset = MathUtil.lerp(levelerNode.minDropHeight,
    levelerNode.maxDropHeight, f)
179
180 local sx,sy,sz = localToWorld(levelerNode.node, -width, 0, 0)
181 local ex,ey,ez = localToWorld(levelerNode.node, width, 0, 0)
182 local dx,dy,dz = localDirectionToWorld(levelerNode.node, 0, 0, 1)
183
184 local backOffset = -outerRadius * spec.pickUpDirection * 1.5
185 local backLen = MathUtil.lerp(levelerNode.minDropDirOffset,
    levelerNode.maxDropDirOffset, f) - backOffset
186
187 local backX,backY,backZ = dx*backOffset,dy*backOffset,dz*backOffset
188 dx,dy,dz = dx*backLen,dy*backLen,dz*backLen
189
190 local terrainHeightUpdater =
    g_densityMapHeightManager:getTerrainDetailHeightUpdater()
191 if terrainHeightUpdater ~= nil then
192 addDensityMapHeightOcclusionArea(terrainHeightUpdater,
    sx+backX,sy+backY,sz+backZ, ex-sx,ey-sy,ez-sz, dx, dy, dz, true)
193 if width < levelerNode.width-0.05 then
194 -- fully block left and right of the inner block area
195 local sx2,sy2,sz2 = localToWorld(levelerNode.node, -levelerNode.width, 0
196 local ex2,ey2,ez2 = localToWorld(levelerNode.node, levelerNode.width, 0,
197
198 addDensityMapHeightOcclusionArea(terrainHeightUpdater,
    sx2+backX,sy2+backY,sz2+backZ, sx-sx2,sy-sy2,sz-sz2, dx, dy, dz, false)
199 addDensityMapHeightOcclusionArea(terrainHeightUpdater, ex +backX,ey +bac
    +backZ, ex2-ex,ey2-ey,ez2-ez, dx, dy, dz, false)
200 end
201 end
202
203 levelerNode.lastDrop2, levelerNode.lineOffsetDrop2 =
    DensityMapHeightUtil.tipToGroundAroundLine(self, fillLevel, fillType,
    sx,sy+yOffset,sz, ex,ey+yOffset,ez, 0, outerRadius,
    levelerNode.lineOffsetDrop2, true, nil)
```

```
204 if levelerNode.lastDrop2 > 0 then
205 self:addFillUnitFillLevel(self:getOwnerFarmId(), spec.fillUnitIndex, -
    levelerNode.lastDrop2, fillType, ToolType.UNDEFINED, nil)
206 end
207 end
208 end
209
210 -- call fill level changed callack to inform bunker silo about change
211 if pickedUpFillLevel < 0 and fillType ~= FillType.UNKNOWN then
212 if self.changedFillLevelCallback ~= nil then
213 if self.changedFillLevelCallbackTarget ~= nil then
214 self.changedFillLevelCallback(self.changedFillLevelCallbackTarget, self,
    pickedUpFillLevel, fillType)
215 else
216 self.changedFillLevelCallback(self, pickedUpFillLevel, fillType)
217 end
218 end
219 end
220 end
221 end
222 end
```

loadLevelerNodeFromXML

Description

Loads leveler node from xml

Definition

loadLevelerNodeFromXML(table levelerNode, integer xmlFile, string key)

Arguments

table	levelerNode	leveler node data
integer	xmlFile	id of xml object
string	key	key

Return Values

boolean success success

Code

```
230 function Leveler:loadLevelerNodeFromXML(levelerNode, xmlFile,
    key)
231 levelerNode.node = I3DUtil.indexToObject(self.components,
    getXMLString(xmlFile, key .. "#node"), self.i3dMappings)
232 if levelerNode.node ~= nil then
233 levelerNode.width = getXMLFloat(xmlFile, key .. "#width")
234
235 levelerNode.minDropWidth = Utils.getNoNil(getXMLFloat(xmlFile,
    key .. "#minDropWidth"), levelerNode.width*0.5)
```

```
236 levelerNode.maxDropWidth = Utils.getNoNil(getXMLFloat(xmlFile,
    key .. "#maxDropWidth"), levelerNode.width)
237 levelerNode.minDropHeight = Utils.getNoNil(getXMLFloat(xmlFile,
    key .. "#minDropHeight"), 0)
238 levelerNode.maxDropHeight = Utils.getNoNil(getXMLFloat(xmlFile,
    key .. "#maxDropHeight"), 1)
239 levelerNode.minDropDirOffset =
    Utils.getNoNil(getXMLFloat(xmlFile, key .. "#minDropDirOffset"),
    0.7)
240 levelerNode.maxDropDirOffset =
    Utils.getNoNil(getXMLFloat(xmlFile, key .. "#maxDropDirOffset"),
    0.7)
241 levelerNode.numHeightLimitChecks =
    Utils.getNoNil(getXMLFloat(xmlFile, key ..
    "#numHeightLimitChecks"), 6)
242
243 levelerNode.lineOffsetPickUp = nil
244 levelerNode.lineOffsetDrop = nil
245
246 levelerNode.lastPickUp = 0
247 levelerNode.lastDrop = 0
248 return true
249 end
250
251 return false
252 end
```

getIsLevelerPickupNodeActive

Description

Returns true if leveler pickup node is active

Definition

getIsLevelerPickupNodeActive(table levelerNode)

Arguments

table levelerNode pickup node data

Return Values

boolean isActive pickup node is active

Code

```
258 function Leveler:getIsLevelerPickupNodeActive(levelerNode)
259 return true
260 end
```

setChangedFillLevelCallback

Description

Set fill level changed callback (e.g. by bunker silo)

Definition

setChangedFillLevelCallback(function callback, table callbackTarget)

Arguments

function callback callback

table callbackTarget callback target

Code

```
266 function Leveler:setChangedFillLevelCallback(callback,
    callbackTarget)
267 self.changedFillLevelCallback = callback
268 self.changedFillLevelCallbackTarget = callbackTarget
269 end
```

Lights

Description

This is the specialization for vehicles with lights

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
27 function Lights.prerequisitesPresent(specializations)
28 return true
29 end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
84 function Lights:onLoad(savegame)
85
86 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
   self.configFileName, "vehicle.lights.low.light#decoration",
   "vehicle.lights.defaultLights#node") --FS17 to FS19
87 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
   self.configFileName, "vehicle.lights.high.light#decoration",
   "vehicle.lights.defaultLights#node") --FS17 to FS19
88 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
   self.configFileName, "vehicle.lights.low.light#realLight",
   "vehicle.lights.realLights.low.light#node") --FS17 to FS19
```

```
89  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.lights.high.light#realLight",
    "vehicle.lights.realLights.high.light#node") --FS17 to FS19
90  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.brakeLights.brakeLight#realLight",
    "vehicle.lights.realLights.high.brakeLight#node") --FS17 to FS19
91  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.brakeLights.brakeLight#decoration",
    "vehicle.lights.brakeLights.brakeLight#node") --FS17 to FS19
92  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.reverseLights.reverseLight#realLight",
    "vehicle.lights.realLights.high.reverseLight#node") --FS17 to
    FS19
93  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.reverseLights.reverseLight#decoration",
    "vehicle.lights.reverseLights.reverseLight#node") --FS17 to FS19
94  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.turnLights.turnLightLeft#realLight",
    "vehicle.lights.realLights.high.turnLightLeft#node") --FS17 to
    FS19
95  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.turnLights.turnLightLeft#decoration",
    "vehicle.lights.turnLights.turnLightLeft#node") --FS17 to FS19
96  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.turnLights.turnLightRight#realLight",
    "vehicle.lights.realLights.high.turnLightRight#node") --FS17 to
    FS19
97  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.turnLights.turnLightRight#decoration",
    "vehicle.lights.turnLights.turnLightRight#node") --FS17 to FS19
98  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.reverseLights.reverseLight#realLight",
    "vehicle.lights.realLights.high.reverseLight#node") --FS17 to
    FS19
99  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.reverseLights.reverseLight#decoration",
    "vehicle.lights.reverseLights.reverseLight#node") --FS17 to FS19
100
101 local spec = self.spec_lights
102
103 spec.shaderDefaultLights = {}
```

```
104 spec.shaderBrakeLights = {}
105 spec.shaderLeftTurnLights = {}
106 spec.shaderRightTurnLights = {}
107 spec.shaderReverseLights = {}
108
109 spec.realLights = {}
110 spec.realLights.low = {lightTypes={}, turnLightsLeft={},
    turnLightsRight={}, brakeLights={}, reverseLights={},
    interiorLights={}}
111 spec.realLights.high = {lightTypes={}, turnLightsLeft={},
    turnLightsRight={}, brakeLights={}, reverseLights={},
    interiorLights={}}
112
113 spec.defaultLights = {}
114 spec.brakeLights = {}
115 spec.reverseLights = {}
116 spec.turnLightsLeft = {}
117 spec.turnLightsRight = {}
118
119 spec.lightsTypesMask = 0
120 spec.currentLightState = 0
121 spec.numLightTypes = 0
122 spec.lightStates = {}
123
124 local registeredLightTypes = {}
125 local i = 0
126 while true do
127 local key = string.format("vehicle.lights.states.state(%d)", i)
128 if not hasXMLProperty(self.xmlFile, key) then
129 break
130 end
131 local lightTypes =
    {StringUtil.getVectorFromString(getXMLString(self.xmlFile,
    key.."#lightTypes"))}
132 for _, lightType in pairs(lightTypes) do
133 if registeredLightTypes[lightType] == nil then
134 registeredLightTypes[lightType] = lightType
135 spec.numLightTypes = spec.numLightTypes + 1
136 end
137 end
138
139 table.insert(spec.lightStates, lightTypes)
```

```
140 i = i + 1
141 end
142
143 local lightTypesStr = Utils.getNoNil(getXMLString(self.xmlFile,
    "vehicle.lights.states.aiState#lightTypes"), "0 1 2")
144 local lightTypes =
    {StringUtil.getVectorFromString(lightTypesStr)}
145 local lightsTypesMask = 0
146 for _, lightType in pairs(lightTypes) do
147 lightsTypesMask = bitOR(lightsTypesMask, 2^lightType)
148 end
149 spec.aiLightsTypesMask = lightsTypesMask
150
151 spec.sharedLights = {}
152 local i = 0
153 while true do
154 local key = string.format("vehicle.lights.sharedLight(%d)", i)
155 if not hasXMLProperty(self.xmlFile, key) then
156 break
157 end
158
159 local sharedLight = {}
160 if self:loadSharedLight(self.xmlFile, key, sharedLight) then
161 table.insert(spec.sharedLights, sharedLight)
162 end
163
164 i = i + 1
165 end
166
167 local realLightToLight = {}
168 self:loadRealLightSetup(self.xmlFile,
    "vehicle.lights.realLights.low", spec.realLights.low,
    realLightToLight)
169 self:loadRealLightSetup(self.xmlFile,
    "vehicle.lights.realLights.high", spec.realLights.high,
    realLightToLight)
170
171 self:loadVisualLights(self.xmlFile,
    "vehicle.lights.defaultLights.defaultLight", true,
    spec.defaultLights, spec.shaderDefaultLights)
172 self:loadVisualLights(self.xmlFile,
    "vehicle.lights.brakeLights.brakeLight", false, spec.brakeLights,
    spec.shaderBrakeLights)
```

```
173 self:loadVisualLights(self.xmlFile, "vehicle.lights.reverseLights.reverseLight", false, spec.reverseLights, spec.shaderReverseLights)
174 self:loadVisualLights(self.xmlFile, "vehicle.lights.turnLights.turnLightLeft", false, spec.turnLightsLeft, spec.shaderLeftTurnLights)
175 self:loadVisualLights(self.xmlFile, "vehicle.lights.turnLights.turnLightRight", false, spec.turnLightsRight, spec.shaderRightTurnLights)
176
177 spec.brakeLightsVisibility = false
178 spec.reverseLightsVisibility = false
179 spec.turnLightState = Lights.TURNLIGHT_OFF
180 spec.hasTurnLights = #spec.turnLightsLeft > 0 or #spec.turnLightsRight > 0
181 spec.turnLightRepetitionCount = 0
182
183 spec.actionEventsActiveChange = {}
184
185 spec.beaconLights = {}
186 local i = 0
187 while true do
188 local key = string.format("vehicle.lights.beaconLights.beaconLight(%d)", i)
189 if not hasXMLProperty(self.xmlFile, key) then
190 break
191 end
192
193 local node = I3DUtil.indexToObject(self.components, getXMLString(self.xmlFile, key.."#node"), self.i3dMappings)
194 if node ~= nil then
195 local lightXmlFilename = getXMLString(self.xmlFile, key.."#filename")
196 if lightXmlFilename ~= nil then
197 lightXmlFilename = Utils.getFilename(lightXmlFilename, self.baseDirectory)
198 local lightXmlFile = loadXMLFile("beaconLightXML", lightXmlFilename)
199
200 if lightXmlFile ~= nil and lightXmlFile ~= 0 then
201 local i3dFilename = getXMLString(lightXmlFile, "beaconLight.filename")
202 if i3dFilename ~= nil then
```

```
203 local i3dNode = g_i3DManager:loadSharedI3DFile(i3dFilename,
    self.baseDirectory, false, false, false)
204 if i3dNode ~= nil and i3dNode ~= 0 then
205 local rootNode = I3DUtil.indexToObject(i3dNode,
    getXMLString(lightXmlFile, "beaconLight.rootNode#node"))
206 local rotatorNode = I3DUtil.indexToObject(i3dNode,
    getXMLString(lightXmlFile, "beaconLight.rotator#node"))
207 local speed = getXMLFloat(lightXmlFile,
    "beaconLight.rotator#speed") or 0.015
208 local lightNode = I3DUtil.indexToObject(i3dNode,
    getXMLString(lightXmlFile, "beaconLight.light#node"))
209 local lightShaderNode = I3DUtil.indexToObject(i3dNode,
    getXMLString(lightXmlFile, "beaconLight.light#shaderNode"))
210 local intensity = getXMLFloat(lightXmlFile,
    "beaconLight.light#intensity") or 1000
211 local realLightNode = I3DUtil.indexToObject(i3dNode,
    getXMLString(lightXmlFile, "beaconLight.realLight#node"))
212
213 if rootNode ~= nil and (lightNode ~= nil or lightShaderNode ~=
    nil) then
214 link(node, rootNode)
215 setTranslation(rootNode, 0,0,0)
216
217 local light = {}
218 light.filename = i3dFilename
219 light.rootNode = rootNode
220 light.rotatorNode = rotatorNode
221 light.lightNode = lightNode
222 light.lightShaderNode = lightShaderNode
223 light.realLightNode = realLightNode
224 light.speed = speed
225 light.intensity = intensity
226
227 if realLightNode ~= nil then
228 light.defaultColor = {getLightColor(realLightNode)}
229 setVisibility(realLightNode, false)
230 end
231
232 if lightNode ~= nil then
233 setVisibility(lightNode, false)
234 end
235 if lightShaderNode ~= nil then
```

```
236 local _,y,z,w = getShaderParameter(lightShaderNode,
    "lightControl")
237 setShaderParameter(lightShaderNode, "lightControl", 0, y, z, w,
    false)
238 end
239
240 if light.speed > 0 then
241 local rot = math.random(0, math.pi*2)
242 if light.rotatorNode ~= nil then
243 setRotation(light.rotatorNode, 0, rot, 0)
244 end
245 end
246
247 table.insert(spec.beaconLights, light)
248 end
249
250 delete(i3dNode)
251 end
252 end
253
254 delete(lightXmlFile)
255 end
256 end
257 end
258
259 i = i + 1
260 end
261 spec.beaconLightsActive = false
262
263 if self.isClient ~= nil then
264 spec.samples = {}
265 spec.samples.toggleLights =
    g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.lights.sounds", "toggleLights", self.baseDirectory,
    self.components, 1, AudioGroup.VEHICLE, self.i3dMappings, self)
266 spec.samples.turnLight =
    g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.lights.sounds", "turnLight", self.baseDirectory,
    self.components, 1, AudioGroup.VEHICLE, self.i3dMappings, self)
267 end
268
269 if self.loadDashboardsFromXML ~= nil then
```

```
270 self:loadDashboardsFromXML(self.xmlFile,
    "vehicle.lights.dashboards", {valueTypeToLoad = "lightState",
271 valueObject = spec,
272 valueFunc = "lightsTypesMask",
273 additionalAttributesFunc = Lights.dashboardLightAttributes,
274 stateFunc = Lights.dashboardLightState})
275
276 self:loadDashboardsFromXML(self.xmlFile,
    "vehicle.lights.dashboards", {valueTypeToLoad = "turnLightLeft",
277 valueObject = spec,
278 valueFunc = "turnLightState",
279 valueCompare = {Lights.TURNLIGHT_LEFT, Lights.TURNLIGHT_HAZARD}})
280
281 self:loadDashboardsFromXML(self.xmlFile,
    "vehicle.lights.dashboards", {valueTypeToLoad = "turnLightRight",
282 valueObject = spec,
283 valueFunc = "turnLightState",
284 valueCompare = {Lights.TURNLIGHT_RIGHT,
    Lights.TURNLIGHT_HAZARD}})
285
286 self:loadDashboardsFromXML(self.xmlFile,
    "vehicle.lights.dashboards", {valueTypeToLoad =
    "turnLightHazard",
287 valueObject = spec,
288 valueFunc = "turnLightState",
289 valueCompare = {Lights.TURNLIGHT_HAZARD}})
290
291 self:loadDashboardsFromXML(self.xmlFile,
    "vehicle.lights.dashboards", {valueTypeToLoad = "turnLightAny",
292 valueObject = spec,
293 valueFunc = "turnLightState",
294 valueCompare = {Lights.TURNLIGHT_LEFT, Lights.TURNLIGHT_RIGHT,
    Lights.TURNLIGHT_HAZARD}})
295
296 self:loadDashboardsFromXML(self.xmlFile,
    "vehicle.lights.dashboards", {valueTypeToLoad = "beaconLight",
297 valueObject = spec,
298 valueFunc = "beaconLightsActive"})
299 end
300
301 spec.dirtyFlag = self:getNextDirtyFlag()
302 end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
306 function Lights:onDelete()
307 local spec = self.spec_lights
308 for _, beaconLight in pairs(spec.beaconLights) do
309 if beaconLight.filename ~= nil then
310 g_i3DManager:releaseSharedI3DFile(beaconLight.filename,
    self.baseDirectory, true)
311 end
312 end
313 for _, sharedLight in pairs(spec.sharedLights) do
314 g_i3DManager:releaseSharedI3DFile(sharedLight.filename,
    self.baseDirectory, true)
315 end
316 if self.isClient ~= nil then
317 for _,sample in pairs(spec.samples) do
318 g_soundManager:deleteSample(sample)
319 end
320 end
321 end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
327 function Lights:onReadStream(streamId, connection)
328
329 local lightsTypesMask = streamReadInt32(streamId)
330 self:setLightsTypesMask(lightsTypesMask, true, true)
331
332 local beaconLightsActive = streamReadBool(streamId)
333 self:setBeaconLightsVisibility(beaconLightsActive, true, true)
334
335 local turnLightState = streamReadUIntN(streamId,
    Lights.turnLightSendNumBits)
```

```
336 self:setTurnLightState(turnLightState, true, true)
337 end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer	streamId	streamId
integer	connection	connection

Code

```
343 function Lights:onWriteStream(streamId, connection)
344 local spec = self.spec_lights
345
346 streamWriteInt32(streamId, spec.lightsTypesMask)
347 streamWriteBool(streamId, spec.beaconLightsActive)
348 streamWriteUIntN(streamId, spec.turnLightState,
    Lights.turnLightSendNumBits)
349 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
385 function Lights:onUpdate(dt, isActiveForInput, isSelected)
386 local spec = self.spec_lights
387
388 if spec.beaconLightsActive then
389 for _, beaconLight in pairs(spec.beaconLights) do
390 if beaconLight.rotatorNode ~= nil then
391 rotate(beaconLight.rotatorNode, 0, beaconLight.speed*dt, 0)
392 end
393 end
394
395 self:raiseActive()
396 end
397
```

```
398 if spec.turnLightState ~= Lights.TURNLIGHT_OFF then
399 local alpha = MathUtil.clamp((math.cos(7*getShaderTimeSec()) +
    0.2), 0, 1)
400
401 if spec.turnLightState == Lights.TURNLIGHT_LEFT or
    spec.turnLightState == Lights.TURNLIGHT_HAZARD then
402 for _, light in pairs(spec.activeTurnLightSetup.turnLightsLeft)
    do
403 setLightColor(light.node, light.defaultColor[1]*alpha,
    light.defaultColor[2]*alpha, light.defaultColor[3]*alpha)
404 for i=0, getNumOfChildren(light.node)-1 do
405 setLightColor(getChildAt(light.node, i),
    light.defaultColor[1]*alpha, light.defaultColor[2]*alpha,
    light.defaultColor[3]*alpha)
406 end
407 end
408 end
409 if spec.turnLightState == Lights.TURNLIGHT_RIGHT or
    spec.turnLightState == Lights.TURNLIGHT_HAZARD then
410 for _, light in pairs(spec.activeTurnLightSetup.turnLightsRight)
    do
411 setLightColor(light.node, light.defaultColor[1]*alpha,
    light.defaultColor[2]*alpha, light.defaultColor[3]*alpha)
412 for i=0, getNumOfChildren(light.node)-1 do
413 setLightColor(getChildAt(light.node, i),
    light.defaultColor[1]*alpha, light.defaultColor[2]*alpha,
    light.defaultColor[3]*alpha)
414 end
415 end
416 end
417
418 self:raiseActive()
419 end
420
421 if self.isClient and spec.samples.turnLight ~= nil then
422 if spec.turnLightState > Lights.TURNLIGHT_OFF then
423 local turnLightRepetitionCount = math.floor((getShaderTimeSec()*7
    + math.acos(-0.2)) / (math.pi*2))
424 if spec.turnLightRepetitionCount ~= nil and
    turnLightRepetitionCount ~= spec.turnLightRepetitionCount then
425 g_soundManager:playSample(spec.samples.turnLight)
426 end
427 spec.turnLightRepetitionCount = turnLightRepetitionCount
```

```
428 end
429 end
430 end
```

getIsActiveForLights

Description

Returns if is active for lights

Definition

getIsActiveForLights()

Return Values

boolean isActive is active for lights

Code

```
452 function Lights:getIsActiveForLights()
453
454 if self.getIsEntered ~= nil and self:getIsEntered() and
    self:getCanToggleLight() then
455 return true
456 end
457 -- ToDo: fix this
458 if self.attacherVehicle ~= nil and (self.isSteerable == nil or
    self.isSteerable == false) then
459 return self.attacherVehicle:getIsActiveForLights()
460 end
461 return false
462 end
```

getCanToggleLight

Description

Returns if lights can be toggled

Definition

getCanToggleLight()

Return Values

boolean canBeToggled lights can be toggled

Code

```
467 function Lights:getCanToggleLight()
468 local spec = self.spec_lights
469
470 if self:getIsAIActive() then
471 return false
472 end
473
474 if spec.numLightTypes == 0 then
475 return false
476 end
```

```
477
478 if g_currentMission.controlledVehicle == self then
479 return true
480 else
481 return false
482 end
483 end
```

getUseHighProfile

Description

Returns if high profile is used

Definition

getUseHighProfile()

Return Values

boolean highProfileUsed high profile is used

Code

```
488 function Lights:getUseHighProfile()
489 local lightsProfile = g_gameSettings:getValue("lightsProfile")
490
491 return lightsProfile == GS_PROFILE_VERY_HIGH or (lightsProfile ==
    GS_PROFILE_HIGH and self:getIsActiveForLights())
492 end
```

setLightsTypesMask

Description

Set light type mask

Definition

setLightsTypesMask(integer lightsTypesMask, boolean force, boolean noEventSend)

Arguments

integer	lightsTypesMask	new light types mask
boolean	force	force action
boolean	noEventSend	no event send

Return Values

boolean changed mask has changed

Code

```
521 function Lights:setLightsTypesMask(lightsTypesMask, force, noEventSend)
522 local spec = self.spec_lights
523
524 if lightsTypesMask ~= spec.lightsTypesMask or force then
525 if noEventSend == nil or noEventSend == false then
526 if g_server ~= nil then
527 g_server:broadcastEvent(VehicleSetLightEvent:new(self,
    lightsTypesMask), nil, nil, self)
528 else
```

```
529 g_client:getServerConnection():sendEvent(VehicleSetLightEvent:new(self,
    lightsTypesMask))
530 end
531 end
532
533 if self.isClient then
534 g_soundManager:playSample(spec.samples.toggleLights)
535 end
536
537 local activeLightSetup = spec.realLights.low
538 local inactiveLightSetup = spec.realLights.high
539 if self:getUseHighProfile() then
540 activeLightSetup = spec.realLights.high
541 inactiveLightSetup = spec.realLights.low
542 end
543
544 -- deactivate inactive spec lights
545 for _, light in pairs(inactiveLightSetup.lightTypes) do
546 setVisibility(light.node, false)
547 end
548
549 local function getIsLightActive(light)
550 local lightActive = false
551 for _, lightType in pairs(light.lightTypes) do
552 if bitAND(lightsTypesMask, 2^lightType) ~= 0 then
553 lightActive = true
554 break
555 end
556 end
557 if light.enableDirection ~= nil then
558 local reverserDirection = 1.0
559 if self.getReverserDirection ~= nil then
560 reverserDirection = self:getReverserDirection()
561 end
562 lightActive = lightActive and light.enableDirection ==
    reverserDirection
563 end
564 if lightActive then
565 for _, excludedLightType in pairs(light.excludedLightTypes) do
566 if bitAND(lightsTypesMask, 2^excludedLightType) ~= 0 then
567 lightActive = false
```

```
568 break
569 end
570 end
571 end
572
573 return lightActive
574 end
575 -- activate real lights
576 for _,light in pairs(activeLightSetup.lightTypes) do
577 local isActive = getIsLightActive(light)
578 if isActive then
579 setVisibility(light.node, true)
580 else
581 local active = false
582 if light.brakeLight ~= nil then
583 if light.brakeLight.isActive then
584 active = true
585 end
586 end
587 setVisibility(light.node, active)
588 end
589
590 light.isActive = isActive
591 end
592 -- activate old corona lights
593 for _, light in pairs(spec.defaultLights) do
594 local isActive = getIsLightActive(light)
595 setVisibility(light.node, isActive)
596 end
597
598 -- set new lights
599 for _, light in pairs(spec.shaderDefaultLights) do
600 local isActive = getIsLightActive(light)
601 local value = 1
602 if not isActive then
603 value = 0
604 end
605 local _,y,z,w = getShaderParameter(light.node, "lightControl")
606 setShaderParameter(light.node, "lightControl",
    math.max(value*light.intensity, 0), y, z, w, false)
```

```
607
608 if light.toggleVisibility then
609 setVisibility(light.node, isActive)
610 end
611 end
612
613 spec.lightsTypesMask = lightsTypesMask
614
615 SpecializationUtil.raiseEvent(self, "onLightsTypesMaskChanged",
    lightsTypesMask)
616 end
617
618 return true
619 end
```

getLightsTypesMask

Description

Get light type mask

Definition

getLightsTypesMask()

Return Values

integer lightsTypesMask light types mask

Code

```
624 function Lights:getLightsTypesMask()
625 return self.spec_lights.lightsTypesMask
626 end
```

setBeaconLightsVisibility

Description

Toggle beacon light visibility

Definition

setBeaconLightsVisibility(boolean visibility, boolean force, boolean noEventSend)

Arguments

boolean visibility	new visibility state
boolean force	force action
boolean noEventSend	no event send

Return Values

boolean changed visibility has changed

Code

```
634 function Lights:setBeaconLightsVisibility(visibility, force, noEventSend
635 local spec = self.spec_lights
636
637 if visibility ~= spec.beaconLightsActive or force then
638 if noEventSend == nil or noEventSend == false then
```

```
639 if g_server ~= nil then
640 g_server:broadcastEvent(VehicleSetBeaconLightEvent:new(self, visibility)
    nil, nil, self)
641 else
642 g_client:getServerConnection():sendEvent(VehicleSetBeaconLightEvent:new
    visibility))
643 end
644 end
645
646 spec.beaconLightsActive = visibility
647 local realBeaconLights = g_gameSettings:getValue("realBeaconLights")
648 for _, beaconLight in pairs(spec.beaconLights) do
649 if realBeaconLights and beaconLight.realLightNode ~= nil then
650 setVisibility(beaconLight.realLightNode, visibility)
651 end
652 if beaconLight.lightNode ~= nil then
653 setVisibility(beaconLight.lightNode, visibility)
654 end
655 if beaconLight.lightShaderNode ~= nil then
656 local value = 1*beaconLight.intensity
657 if not visibility then
658 value = 0
659 end
660
661 local _,y,z,w = getShaderParameter(beaconLight.lightShaderNode,
    "lightControl")
662 setShaderParameter(beaconLight.lightShaderNode, "lightControl", value, y
    w, false)
663 end
664 end
665
666 SpecializationUtil.raiseEvent(self, "onBeaconLightsVisibilityChanged",
    visibility)
667 end
668
669 return true
670 end
```

getBeaconLightsVisibility

Description

Get beacon light state

Definition

getBeaconLightsVisibility()

Return Values

boolean state beacon light state

Code

```
675 function Lights:getBeaconLightsVisibility()
676 return self.spec_lights.beaconLightsActive
677 end
```

setTurnLightState

Description

Toggle turn light state

Definition

setTurnLightState(boolean state, boolean force, boolean noEventSend)

Arguments

boolean state	new state
boolean force	force action
boolean noEventSend	no event send

Return Values

boolean changed state has changed

Code

```
685 function Lights:setTurnLightState(state, force, noEventSend)
686 local spec = self.spec_lights
687
688 if state ~= spec.turnLightState or force then
689 if noEventSend == nil or noEventSend == false then
690 if g_server ~= nil then
691 g_server:broadcastEvent(VehicleSetTurnLightEvent:new(self, state), nil,
    nil, self)
692 else
693 g_client:getServerConnection():sendEvent(VehicleSetTurnLightEvent:new(se
    state))
694 end
695 end
696
697 local activeLightSetup = spec.realLights.low
698 local inactiveLightSetup = spec.realLights.high
699 if self:getUseHighProfile() then
700 activeLightSetup = spec.realLights.high
701 inactiveLightSetup = spec.realLights.low
702 end
703
704 spec.activeTurnLightSetup = activeLightSetup
705
706 -- deactivate inactive spec lights
```

```
707 for _, light in pairs(inactiveLightSetup.turnLightsLeft) do
708 setVisibility(light.node, false)
709 end
710 for _, light in pairs(inactiveLightSetup.turnLightsRight) do
711 setVisibility(light.node, false)
712 end
713 -- set correct state for active spec lights
714 for _, light in pairs(activeLightSetup.turnLightsLeft) do
715 setVisibility(light.node, state == Lights.TURNLIGHT_LEFT or state ==
    Lights.TURNLIGHT_HAZARD)
716 end
717 for _, light in pairs(activeLightSetup.turnLightsRight) do
718 setVisibility(light.node, state == Lights.TURNLIGHT_RIGHT or state ==
    Lights.TURNLIGHT_HAZARD)
719 end
720 -- update old corona light elements
721 for _, light in pairs(spec.turnLightsLeft) do
722 setVisibility(light.node, state == Lights.TURNLIGHT_LEFT or state ==
    Lights.TURNLIGHT_HAZARD)
723 end
724 for _, light in pairs(spec.turnLightsRight) do
725 setVisibility(light.node, state == Lights.TURNLIGHT_RIGHT or state ==
    Lights.TURNLIGHT_HAZARD)
726 end
727
728 -- update shader lights
729 local value = 1
730 if state ~= Lights.TURNLIGHT_LEFT and state ~= Lights.TURNLIGHT_HAZARD t
731 value = 0
732 end
733 for _, sharedTurnLight in ipairs(spec.shaderLeftTurnLights) do
734 local _,y,z,w = getShaderParameter(sharedTurnLight.node, "lightControl")
735 setShaderParameter(sharedTurnLight.node, "lightControl",
    value*sharedTurnLight.intensity, y, z, w, false)
736
737 if sharedTurnLight.toggleVisibility then
738 setVisibility(sharedTurnLight.node, value == 1)
739 end
740 end
741 value = 1
742 if state ~= Lights.TURNLIGHT_RIGHT and state ~= Lights.TURNLIGHT_HAZARD
    then
```

```
743 value = 0
744 end
745 for _, sharedTurnLight in ipairs(spec.shaderRightTurnLights) do
746 local _,y,z,w = getShaderParameter(sharedTurnLight.node, "lightControl")
747 setShaderParameter(sharedTurnLight.node, "lightControl",
    value*sharedTurnLight.intensity, y, z, w, false)
748
749 if sharedTurnLight.toggleVisibility then
750 setVisibility(sharedTurnLight.node, value == 1)
751 end
752 end
753
754 spec.turnLightState = state
755
756 SpecializationUtil.raiseEvent(self, "onTurnLightStateChanged", state)
757 end
758
759 return true
760 end
```

getTurnLightState

Description

Get turn light state

Definition

getTurnLightState()

Return Values

integer state turn light state

Code

```
765 function Lights:getTurnLightState()
766 return self.spec_lights.turnLightState
767 end
```

setBrakeLightsVisibility

Description

Set brake light visibility

Definition

setBrakeLightsVisibility(boolean visibility)

Arguments

boolean visibility new visibility

Return Values

boolean changed visibility has changed

Code

```
773 function Lights:setBrakeLightsVisibility(visibility)
774 local spec = self.spec_lights
```

```
775
776 if visibility ~= spec.brakeLightsVisibility then
777
778 local activeLightSetup = spec.realLights.low
779 local inactiveLightSetup = spec.realLights.high
780 if self:getUseHighProfile() then
781 activeLightSetup = spec.realLights.high
782 inactiveLightSetup = spec.realLights.low
783 end
784
785 -- deactivate inactive spec lights
786 for _, light in pairs(inactiveLightSetup.brakeLights) do
787 setVisibility(light.node, false)
788 end
789
790 -- set correct state for active spec lights
791 for _, light in pairs(activeLightSetup.brakeLights) do
792 light.isActive = visibility
793 if visibility then
794 if light.backLight ~= nil then
795 local color = light.backLight.defaultColor
796 setLightColor(light.node, color[1]*2, color[2]*2, color[3]*2)
797 for i=0, getNumOfChildren(light.node)-1 do
798 setLightColor(getChildAt(light.node, i), color[1]*2, color[2]*2,
    color[3]*2)
799 end
800 end
801 setVisibility(light.node, true)
802 light.isActive = true
803 else
804 local isVisible = false
805 if light.backLight ~= nil then
806 local color = light.backLight.defaultColor
807 setLightColor(light.node, color[1], color[2], color[3])
808 for i=0, getNumOfChildren(light.node)-1 do
809 setLightColor(getChildAt(light.node, i), color[1], color[2],
    color[3])
810 end
811 if light.backLight.isActive then
812 isVisible = true
813 end
```

```
814 end
815 setVisibility(light.node, isVisible)
816 end
817 end
818
819 for _, light in pairs(spec.brakeLights) do
820 setVisibility(light.node, visibility)
821 end
822
823 local dir = 1
824 if not visibility then
825 dir = -1
826 end
827 for _, sharedBrakeLight in ipairs(spec.shaderBrakeLights) do
828 local x,y,z,w = getShaderParameter(sharedBrakeLight.node,
    "lightControl")
829 setShaderParameter(sharedBrakeLight.node, "lightControl",
    math.max(x+dir*sharedBrakeLight.intensity, 0), y, z, w, false)
830
831 if sharedBrakeLight.toggleVisibility then
832 setVisibility(sharedBrakeLight.node, visibility)
833 end
834 end
835
836 spec.brakeLightsVisibility = visibility
837
838 SpecializationUtil.raiseEvent(self,
    "onBrakeLightsVisibilityChanged", visibility)
839 end
840
841 return true
842 end
```

setReverseLightsVisibility

Description

Set reverse light visibility

Definition

setReverseLightsVisibility(boolean visibility)

Arguments

boolean visibility new visibility

Return Values

boolean changed visibility has changed

Code

```lua
function Lights:setReverseLightsVisibility(visibility)
local spec = self.spec_lights

if visibility ~= spec.reverseLightsVisibility then

local activeLightSetup = spec.realLights.low
local inactiveLightSetup = spec.realLights.high
if self:getUseHighProfile() then
activeLightSetup = spec.realLights.high
inactiveLightSetup = spec.realLights.low
end

-- deactivate inactive spec lights
for _, light in pairs(inactiveLightSetup.reverseLights) do
setVisibility(light.node, false)
end

-- set correct state for active spec lights
for _, light in pairs(activeLightSetup.reverseLights) do
setVisibility(light.node, visibility)
end

-- update old corona light elements
for _, light in pairs(spec.reverseLights) do
setVisibility(light.node, visibility)
end

-- update shader lights
local dir = 1
if not visibility then
dir = -1
end
for _, sharedReverseLight in ipairs(spec.shaderReverseLights) do
local x,y,z,w = getShaderParameter(sharedReverseLight.node, "lightControl")
setShaderParameter(sharedReverseLight.node, "lightControl", math.max(x+dir*sharedReverseLight.intensity), y, z, w, false)

if sharedReverseLight.toggleVisibility then
setVisibility(sharedReverseLight.node, visibility)
end
```

```
887 end
888
889 spec.reverseLightsVisibility = visibility
890
891 SpecializationUtil.raiseEvent(self,
    "onReverseLightsVisibilityChanged", visibility)
892 end
893
894 return true
895 end
```

setInteriorLightsVisibility

Description

Set interior light visibility

Definition

setInteriorLightsVisibility(boolean visibility)

Arguments

boolean visibility new visibility

Return Values

boolean changed visibility has changed

Code

```
901 function Lights:setInteriorLightsVisibility(visibility)
902 local spec = self.spec_lights
903
904 -- interior lights are turned of between 8am and 10am and turn on
    between 4pm and 6pm
905 local brightness = 0
906 local hour = g_currentMission.environment.currentHour +
    g_currentMission.environment.currentMinute / 60
907 if hour < 10 then
908 brightness = 1-(hour - 8) / 2
909 end
910 if hour > 16 then
911 brightness = (hour - 16) / 2
912 end
913
914 brightness = MathUtil.clamp(brightness, 0, 1)
915
916 if brightness == 0 then
917 visibility = false
918 end
919
```

```
920  if visibility ~= spec.interiorLightsVisibility or brightness ~=
     spec.interiorLightsBrightness then
921  local activeLightSetup = spec.realLights.low
922  local inactiveLightSetup = spec.realLights.high
923  if self:getUseHighProfile() then
924  activeLightSetup = spec.realLights.high
925  inactiveLightSetup = spec.realLights.low
926  end
927
928  -- deactivate inactive spec lights
929  for _, light in pairs(inactiveLightSetup.interiorLights) do
930  setVisibility(light.node, false)
931  end
932
933  -- set correct state for active spec lights
934  for _, light in pairs(activeLightSetup.interiorLights) do
935  if visibility then
936  if light.startColor == nil then
937  light.startColor = {getLightColor(light.node)}
938  end
939
940  setLightColor(light.node, light.startColor[1]*brightness,
     light.startColor[2]*brightness, light.startColor[3]*brightness)
941  end
942
943  light.isActive = visibility
944  setVisibility(light.node, visibility)
945  end
946
947  spec.interiorLightsVisibility = visibility
948  spec.interiorLightsBrightness = brightness
949  end
950
951  return true
952  end
```

LivestockTrailer
Description

Class for livestock trailers

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
53  function LivestockTrailer:onLoad(savegame)
54  local spec = self.spec_livestockTrailer
55
56  spec.animalPlaces = {}
57  spec.animalTypeToPlaces = {}
58  local i = 0
59  while true do
60  local key = string.format("vehicle.livestockTrailer.animal(%d)",
    i)
61  if not hasXMLProperty(self.xmlFile, key) then
62  break
63  end
64
65  local place = {}
66  place.numUsed = 0
67
68  local animalTypeStr = getXMLString(self.xmlFile, key .. "#type")
69  local animalType =
    g_animalManager:getAnimalsByType(animalTypeStr)
70  if animalType == nil then
71  g_logManager:xmlWarning(self.configFileName, "Animal type '%s'
    could not be found!", animalTypeStr)
72  break
73  end
74  place.animalType = animalType.type
75
76  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, key.."#index", key.."#node") --FS17 to FS19
77
78  place.slots = {}
79  local parent = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, key .. "#node"), self.i3dMappings)
80  local numSlots = math.abs(getXMLInt(self.xmlFile,
    key.."#numSlots") or 0)
81  if numSlots > getNumOfChildren(parent) then
82  g_logManager:xmlWarning(self.configFileName, "numSlots is greater
    than available children for '%s'", key)
```

```
numSlots = getNumOfChildren(parent)
end
for j=0, numSlots-1 do
local slotNode = getChildAt(parent, j)
table.insert(place.slots, {linkNode=slotNode, loadedMesh=nil, place=place})
end

table.insert(spec.animalPlaces, place)
spec.animalTypeToPlaces[place.animalType] = place

i = i + 1
end

spec.loadedAnimals = {}
spec.loadedAnimalsIds = {}
spec.animalToSlots = {}
spec.animalsToLoad = nil
spec.loadingTrigger = nil

spec.currentAnimalType = nil

spec.dirtyFlag = self:getNextDirtyFlag()
end
```

Locomotive
Description

Class for all Locomotives

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
function Locomotive.prerequisitesPresent(specializations)
return SpecializationUtil.hasSpecialization(SplineVehicle, specializations) and SpecializationUtil.hasSpecialization(Drivable, specializations)
end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
58  function Locomotive:onLoad(savegame)
59  local spec = self.spec_locomotive
60
61  spec.powerArm = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, "vehicle.locomotive.powerArm#node"),
    self.i3dMappings)
62  spec.lastElectricitySplineTime = 0
63  spec.lastSplineTime = 0
64  spec.splineDiff = 0
65  spec.electricitySpline = nil
66
67  spec.lastVirtualRpm = self:getMotor():getMinRpm()
68  spec.speed = 0
69  spec.lastAcceleration = 0
70  spec.nextMovingDirection = 0
71
72  spec.state = Locomotive.STATE_NONE
73
74  spec.doStartCheck = true
75  end
```

MixerWagon

Description

Class for all mixer wagons

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
22  function MixerWagon.prerequisitesPresent(specializations)
```

```
23  return SpecializationUtil.hasSpecialization(Trailer,
    specializations) and
    SpecializationUtil.hasSpecialization(TurnOnVehicle,
    specializations)
24  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
53  function MixerWagon:onLoad(savegame)
54
55  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.mixerWagonBaleTrigger#index", "vehicle.mixerWagon.baleTrigger#r
    --FS17 to FS19
56  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.mixerWagon.baleTrigger#index", "vehicle.mixerWagon.baleTrigger#
    --FS19 to FS19
57
58  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.mixerWagonPickupStartSound", "vehicle.turnOnVehicle.sounds.star
    FS17 to FS19
59  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.mixerWagonPickupStopSound", "vehicle.turnOnVehicle.sounds.stop"
    FS17 to FS19
60  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.mixerWagonPickupSound", "vehicle.turnOnVehicle.sounds.work") --
    to FS19
61
62  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.mixerWagonRotatingParts.mixerWagonRotatingPart#type",
    "vehicle.mixerWagon.mixAnimationNodes.animationNode", "mixerWagonMix") -
    to FS19
63  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.mixerWagonRotatingParts.mixerWagonRotatingPart#type",
    "vehicle.mixerWagon.pickupAnimationNodes.animationNode", "mixerWagonPick
    -FS17 to FS19
64
65  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.mixerWagonRotatingParts.mixerWagonScroller",
    "vehicle.mixerWagon.pickupAnimationNodes.pickupAnimationNode") --FS17 to
66
67  local spec = self.spec_mixerWagon
```

```
68
69  if self.isClient then
70  spec.mixAnimationNodes = g_animationManager:loadAnimations(self.xmlFile,
    "vehicle.mixerWagon.mixAnimationNodes", self.components, self,
    self.i3dMappings)
71  spec.pickupAnimationNodes = g_animationManager:loadAnimations(self.xmlFi
    "vehicle.mixerWagon.pickupAnimationNodes", self.components, self,
    self.i3dMappings)
72  end
73
74  if self.isServer then
75  spec.baleTriggerNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, "vehicle.mixerWagon.baleTrigger#node"),
    self.i3dMappings)
76  if spec.baleTriggerNode ~= nil then
77  addTrigger(spec.baleTriggerNode, "mixerWagonBaleTriggerCallback", self)
78  end
79  end
80
81  spec.activeTimerMax = 5000
82  spec.activeTimer = 0
83
84  spec.fillUnitIndex = Utils.getNoNil(getXMLInt(self.xmlFile,
    "vehicle.mixerWagon#fillUnitIndex"), 1)
85
86  -- remove grass_windrow from fillTypes, we do not want to support grass
    mixer wagons becasue this would make hay "useless"
87  local fillUnit = self:getFillUnitByIndex(spec.fillUnitIndex)
88  if fillUnit ~= nil then
89  fillUnit.needsSaving = false
90  if fillUnit.supportedFillTypes[FillType.GRASS_WINDROW] then
91  fillUnit.supportedFillTypes[FillType.GRASS_WINDROW] = nil
92  end
93  end
94
95  -- disable sync of fillLevels for fillUnit - will be handled by mixerWag
96  fillUnit.synchronizeFillLevel = false
97
98  spec.mixerWagonFillTypes = {}
99  spec.fillTypeToMixerWagonFillType = {}
100 local sumRatio = 0
101 local i=0
```

```
102 while true do
103 local baseName = string.format("vehicle.mixerWagon.mixerWagonFillTypes.mixerWagonFillType
    i)
104 if not hasXMLProperty(self.xmlFile, baseName) then
105 break
106 end
107 local entry = {}
108 entry.fillTypes = {}
109
110 local j=0
111 while true do
112 local fillTypeKey = baseName..string.format(".fillType(%d)", j)
113 if not hasXMLProperty(self.xmlFile, fillTypeKey) then
114 break
115 end
116 local fillTypeStr = getXMLString(self.xmlFile, fillTypeKey.."#fillType")
117 if fillTypeStr ~= nil then
118 local fillTypeIndex = g_fillTypeManager:getFillTypeIndexByName(fillTypeS
119 -- we do not want to support grass in mixer wagons becasue this would ma
    "useless"
120 if fillTypeIndex ~= nil and fillTypeIndex ~= FillType.GRASS_WINDROW then
121 if spec.fillTypeToMixerWagonFillType[fillTypeIndex] == nil then
122 entry.fillTypes[fillTypeIndex] = true
123 spec.fillTypeToMixerWagonFillType[fillTypeIndex] = entry
124 else
125 g_logManager:xmlWarning(self.configFileName, "MixerWagonFillType '%s' in
    already used! Ignoring it!", fillTypeKey, fillTypeStr)
126 end
127 else
128 g_logManager:xmlWarning(self.configFileName, "FillType '%s' not defined
    mixerWagonFillType '%s'!", fillTypeStr, fillTypeKey)
129 end
130 end
131
132 j = j+1
133 end
134
135 entry.name = Utils.getNoNil(getXMLString(self.xmlFile, baseName.."#name"
    "unknown")
136 entry.minPercentage = Utils.getNoNil(getXMLFloat(self.xmlFile,
    baseName.."#minPercentage"), 0)*0.01
```

```
137 entry.maxPercentage = Utils.getNoNil(getXMLFloat(self.xmlFile,
    baseName.."#maxPercentage"), 100)*0.01
138 entry.ratio = entry.maxPercentage - entry.minPercentage
139 entry.fillLevel = 0
140
141 if next(entry.fillTypes) ~= nil then
142 sumRatio = sumRatio + entry.ratio
143 table.insert(spec.mixerWagonFillTypes, entry)
144 end
145 i = i+1
146 end
147
148 for i, entry in ipairs(spec.mixerWagonFillTypes) do
149 entry.ratio = entry.ratio / sumRatio
150 end
151
152 spec.dirtyFlag = self:getNextDirtyFlag()
153
154 if savegame ~= nil then
155 for i, entry in ipairs(spec.mixerWagonFillTypes) do
156 local fillTypeKey =
    savegame.key..string.format(".mixerWagon.fillType(%d)#fillLevel", i-1)
157 local fillLevel = Utils.getNoNil(getXMLFloat(savegame.xmlFile, fillTypeK
    0)
158
159 if fillLevel > 0 then
160 self:addFillUnitFillLevel(self:getOwnerFarmId(), spec.fillUnitIndex,
    fillLevel, next(entry.fillTypes), ToolType.UNDEFINED, nil)
161 end
162 end
163 end
164 end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
168 function MixerWagon:onDelete()
169 local spec = self.spec_mixerWagon
170 if self.isServer then
```

```
171 if spec.baleTriggerNode ~= nil then
172 removeTrigger(spec.baleTriggerNode)
173 end
174 end
175
176 if self.isClient then
177 g_animationManager:deleteAnimations(spec.mixAnimationNodes)
178 g_animationManager:deleteAnimations(spec.pickupAnimationNodes)
179 end
180 end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId
integer connection connection

Code

```
195 function MixerWagon:onReadStream(streamId, connection)
196 local spec = self.spec_mixerWagon
197 for _, entry in ipairs(spec.mixerWagonFillTypes) do
198 local fillLevel = streamReadFloat32(streamId)
199 if fillLevel > 0 then
200 self:addFillUnitFillLevel(self:getOwnerFarmId(),
    spec.fillUnitIndex, fillLevel, next(entry.fillTypes),
    ToolType.UNDEFINED, nil)
201 end
202 end
203 end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId
integer connection connection

Code

```
209 function MixerWagon:onWriteStream(streamId, connection)
210 local spec = self.spec_mixerWagon
211 for _, entry in ipairs(spec.mixerWagonFillTypes) do
```

```
212 streamWriteFloat32(streamId, entry.fillLevel)
213 end
214 end
```

onReadUpdateStream

Description

Called on on update

Definition

onReadUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId	stream ID
integer timestamp	timestamp
table	connection connection

Code

```
221 function MixerWagon:onReadUpdateStream(streamId, timestamp,
    connection)
222 if connection:getIsServer() then
223 if streamReadBool(streamId) then
224 local spec = self.spec_mixerWagon
225
226 for _, entry in ipairs(spec.mixerWagonFillTypes) do
227 local fillLevel = streamReadFloat32(streamId)
228 local delta = fillLevel - entry.fillLevel
229 if delta ~= 0 then
230 self:addFillUnitFillLevel(self:getOwnerFarmId(),
    spec.fillUnitIndex, delta, next(entry.fillTypes),
    ToolType.UNDEFINED, nil)
231 end
232 end
233 end
234 end
235 end
```

onWriteUpdateStream

Description

Called on on update

Definition

onWriteUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Code

```
242 function MixerWagon:onWriteUpdateStream(streamId, connection,
    dirtyMask)
```

```
243 if not connection:getIsServer() then
244 local spec = self.spec_mixerWagon
245 if streamWriteBool(streamId, bitAND(dirtyMask, spec.dirtyFlag) ~=
    0) then
246
247 for _, entry in ipairs(spec.mixerWagonFillTypes) do
248 streamWriteFloat32(streamId, entry.fillLevel)
249 end
250 end
251 end
252 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
259 function MixerWagon:onUpdate(dt, isActiveForInput, isSelected)
260 local spec = self.spec_mixerWagon
261
262 local tipState = self:getTipState()
263 local isTurnedOn = self:getIsTurnedOn()
264 local isDischarging = tipState == Trailer.TIPSTATE_OPENING or
    tipState == Trailer.TIPSTATE_OPEN
265
266 if spec.activeTimer > 0 or isTurnedOn or isDischarging then
267 spec.activeTimer = spec.activeTimer - dt
268 g_animationManager:startAnimations(spec.mixAnimationNodes)
269 else
270 g_animationManager:stopAnimations(spec.mixAnimationNodes)
271 end
272 end
```

mixerWagonBaleTriggerCallback

Description

Trigger callback

Definition

mixerWagonBaleTriggerCallback(integer triggerId, integer otherActorId, boolean onEnter, boolean onLeave, boolean onStay, integer otherShapeId)

Arguments

integer	triggerId	id of trigger
integer	otherActorId	id of other actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay
integer	otherShapeId	id of other shape

Code

```
282 function MixerWagon:mixerWagonBaleTriggerCallback(triggerId, otherActorI
    onEnter, onLeave, onStay, otherShapeId)
283 if onEnter then
284 -- this happens if a compound child of a deleted compound is entering
285 if otherActorId ~= 0 then
286 local bale = g_currentMission:getNodeObject(otherActorId)
287 if bale ~= nil then
288 if bale:isa(Bale) then
289 local spec = self.spec_mixerWagon
290 local fillLevel = bale:getFillLevel()
291 local fillTypeIndex = bale:getFillType()
292
293 if self:getFillUnitSupportsFillType(spec.fillUnitIndex, fillTypeIndex) a
    self:getFillUnitFreeCapacity(spec.fillUnitIndex) > 0 then
294 self:addFillUnitFillLevel(self:getOwnerFarmId(), spec.fillUnitIndex, fil
    fillTypeIndex, ToolType.BALE, nil)
295 bale:delete()
296 spec.activeTimer = spec.activeTimerMax
297 self:raiseDirtyFlags(spec.dirtyFlag)
298 else
299 if self.isClient then
300 g_currentMission:addIngameNotification(FSBaseMission.INGAME_NOTIFICATION
    g_i18n:getText("warning_baleNotSupported"))
301 end
302 g_server:broadcastEvent(MixerWagonBaleNotAcceptedEvent:new(self), nil, n
303 end
304 end
305 end
306 end
307 end
308 end
```

Motorized

Description

Class for all vehicles with motors

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
94  function Motorized:onLoad(savegame)
95  local spec = self.spec_motorized
96
97  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.turnedOnRotationNodes.turnedOnRotationNode#type",
    "vehicle.motor.animationNodes.animationNode", "motor") --FS17 to FS19
98  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.differentialConfigurations",
    "vehicle.motorized.differentialConfigurations") --FS17 to FS19
99  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.motorConfigurations", "vehicle.motorized.motorConfigurations")
    FS17 to FS19
100 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.fuelCapacity", "vehicle.fillUnit") --FS17 to FS19
101 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.maximalAirConsumptionPerFullStop",
    "vehicle.motorized.consumerConfigurations.consumerConfiguration.consumer
    fill type 'air')#usage (is now in usage per second at full brake power)'
    FS17 to FS19
102 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.indoorHud.rpm", "vehicle.motorized.dashboards.dashboard with
    valueType 'rpm'") --FS17 to FS19
103 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.indoorHud.speed", "vehicle.motorized.dashboards.dashboard with
    valueType 'speed'") --FS17 to FS19
104 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.indoorHud.fuelUsage", "vehicle.motorized.dashboards.dashboard v
    valueType 'fuelUsage'") --FS17 to FS19
105 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.indoorHud.fuel", "fillUnit.dashboard with valueType 'fillLevel'
    FS17 to FS19
106
107 spec.motorizedNode = nil
108 for _, component in pairs(self.components) do
109 if component.motorized then
110 spec.motorizedNode = component.node
111 break
112 end
```

```
113 end
114
115 self:loadDifferentials(self.xmlFile, self.differentialIndex)
116 self:loadMotor(self.xmlFile, self.configurations["motor"])
117 self:loadSounds(self.xmlFile, self.configurations["motor"])
118
119 self:loadConsumerConfiguration(self.xmlFile, spec.consumerConfigurationI
120
121 if self.isClient then
122 self:loadExhaustEffects(self.xmlFile)
123 end
124
125 spec.stopMotorOnLeave = true
126
127 spec.motorStartDuration = 0
128 if spec.samples ~= nil and spec.samples.motorStart ~= nil then
129 spec.motorStartDuration = spec.samples.motorStart.duration
130 end
131 spec.motorStartDuration =
    Utils.getNoNil(Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.motorized.motorStartDuration"), spec.motorStartDuration), 0)
132 spec.consumersEmptyWarning =
    g_i18n:getText(Utils.getNoNil(getXMLString(self.xmlFile,
    "vehicle.motorized#consumersEmptyWarning"), "warning_motorFuelEmpty"))
133
134 spec.turnOnText = g_i18n:getText(Utils.getNoNil(getXMLString(self.xmlFil
    "vehicle.motorized#turnOnText"), "action_startMotor"))
135 spec.turnOffText = g_i18n:getText(Utils.getNoNil(getXMLString(self.xmlFi
    "vehicle.motorized#turnOffText"), "action_stopMotor"))
136
137 spec.speedDisplayScale = 1
138 spec.motorStartTime = 0
139 spec.lastRoundPerMinute = 0
140 spec.actualLoadPercentage = 0
141 spec.smoothedLoadPercentage = 0
142 spec.maxDecelerationDuringBrake = 0
143 spec.showTurnOnMotorWarning = 0
144
145 spec.isMotorStarted = false
146 spec.motorStopTimerDuration =
    g_gameSettings:getValue("motorStopTimerDuration")
147 spec.motorStopTimer = spec.motorStopTimerDuration
```

```
148
149 spec.motorTemperature = {}
150 spec.motorTemperature.value = 20
151 spec.motorTemperature.valueSend = 20
152 spec.motorTemperature.valueMax = 120
153 spec.motorTemperature.valueMin = 20
154 spec.motorTemperature.heatingPerMS = 1.5 / 1000 -- delta °C per ms, at f
    load
155 spec.motorTemperature.coolingByWindPerMS = 1.00 / 1000
156
157 spec.motorFan = {}
158 spec.motorFan.enabled = false
159 spec.motorFan.enableTemperature = 95
160 spec.motorFan.disableTemperature = 85
161 spec.motorFan.coolingPerMS = 3.0 / 1000
162
163 spec.lastFuelUsage = 0
164 spec.lastDefUsage = 0
165 spec.lastVehicleDamage = 0
166
167 if self.loadDashboardsFromXML ~= nil then
168 self:loadDashboardsFromXML(self.xmlFile, "vehicle.motorized.dashboards",
    {valueTypeToLoad = "rpm",
169 valueObject = spec.motor,
170 valueFunc = "getLastMotorRpm",
171 minFunc = 0,
172 maxFunc = "getMaxRpm"})
173
174 self:loadDashboardsFromXML(self.xmlFile, "vehicle.motorized.dashboards",
    {valueTypeToLoad = "speed",
175 valueObject = self,
176 valueFunc = "getLastSpeed",
177 minFunc = 0,
178 maxFunc = self:getMotor():getMaximumForwardSpeed()*3.6})
179
180 self:loadDashboardsFromXML(self.xmlFile, "vehicle.motorized.dashboards",
    {valueTypeToLoad = "speedDir",
181 valueObject = self,
182 valueFunc = Motorized.getDashboardSpeedDir,
183 minFunc = -self:getMotor():getMaximumBackwardSpeed()*3.6,
184 maxFunc = self:getMotor():getMaximumForwardSpeed()*3.6,
```

```
185   centerFunc = 0})
186
187   self:loadDashboardsFromXML(self.xmlFile, "vehicle.motorized.dashboards",
      {valueTypeToLoad = "fuelUsage",
188   valueObject = spec,
189   valueFunc = "lastFuelUsage"})
190
191   self:loadDashboardsFromXML(self.xmlFile, "vehicle.motorized.dashboards",
      {valueTypeToLoad = "motorTemperature",
192   valueObject = spec.motorTemperature,
193   valueFunc = "value",
194   minFunc = "valueMin",
195   maxFunc = "valueMax"})
196
197   self:loadDashboardsFromXML(self.xmlFile, "vehicle.motorized.dashboards",
      {valueTypeToLoad = "motorTemperatureWarning",
198   valueObject = spec.motorTemperature,
199   valueFunc = "value",
200   additionalAttributesFunc = Dashboard.warningAttributes,
201   stateFunc = Dashboard.warningState})
202   end
203
204   spec.animationNodes = g_animationManager:loadAnimations(self.xmlFile,
      "vehicle.motorized.animationNodes", self.components, self, self.i3dMappi
205
206   spec.dirtyFlag = self:getNextDirtyFlag()
207   end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
302   function Motorized:onUpdate(dt, isActiveForInput, isSelected)
303   local spec = self.spec_motorized
304
305   local accInput = 0
306   if self.getAxisForward ~= nil then
```

```
307 accInput = self:getAxisForward()
308 end
309
310 if self:getIsMotorStarted() then
311 spec.motor:update(dt)
312 if self.getCruiseControlState ~= nil then
313 if self:getCruiseControlState() ~= Drivable.CRUISECONTROL_STATE_OFF then
314 accInput = 1
315 end
316 end
317
318 if self.isServer then
319 self:updateConsumers(dt, accInput)
320
321 -- update motor properties on damage change to update the torque reducti
322 local damage = self:getVehicleDamage()
323 if math.abs(damage - spec.lastVehicleDamage) > 0.05 then
324 self:updateMotorProperties()
325 spec.lastVehicleDamage = self:getVehicleDamage()
326 end
327 end
328
329 if self.isClient then
330 -- update sounds
331 local samples = spec.samples
332 if g_soundManager:getIsSamplePlaying(spec.motorSamples[1], 1.5*dt) then
333 -- air compressor fill sound
334 if samples.airCompressorStart ~= nil and samples.airCompressorStop ~= ni
    samples.airCompressorRun ~= nil then
335 if spec.consumersByFillTypeName ~= nil and spec.consumersByFillTypeName.
    nil then
336 local consumer = spec.consumersByFillTypeName.air
337
338 if not consumer.doRefill then
339 if g_soundManager:getIsSamplePlaying(samples.airCompressorRun) then
340 g_soundManager:stopSample(samples.airCompressorRun)
341 g_soundManager:playSample(samples.airCompressorStop)
342 end
343 else
344 if not g_soundManager:getIsSamplePlaying(samples.airCompressorRun) then
```

```
345 if not g_soundManager:getIsSamplePlaying(samples.airCompressorStart, 1.5
    spec.brakeCompressor.playSampleRunTime == nil then
346 g_soundManager:playSample(samples.airCompressorStart)
347 spec.playSampleRunTime = g_currentMission.time +
    samples.airCompressorStart.duration
348 end
349 if not g_soundManager:getIsSamplePlaying(samples.airCompressorStart) the
350 spec.brakeCompressor.playSampleRunTime = nil
351 g_soundManager:stopSample(samples.airCompressorStart)
352 g_soundManager:playSample(samples.airCompressorRun)
353 end
354 end
355 end
356 end
357 end
358
359 -- random zsch sound
360 if spec.compressionSoundTime <= g_currentMission.time then
361 g_soundManager:playSample(samples.airRelease)
362 spec.compressionSoundTime = g_currentMission.time + math.random(10000, 4
363 end
364
365 local isBraking = self:getDecelerationAxis() > 0 and self:getLastSpeed()
366
367 -- brake zsch sound
368 if samples.compressedAir ~= nil then
369 if isBraking then
370 samples.compressedAir.brakeTime = samples.compressedAir.brakeTime + dt
371 else
372 if samples.compressedAir.brakeTime > 0 then
373 samples.compressedAir.lastBrakeTime = samples.compressedAir.brakeTime
374 samples.compressedAir.brakeTime = 0
375
376 g_soundManager:playSample(samples.compressedAir)
377 end
378 end
379 end
380
381 --brake sound
382 if samples.brake ~= nil then
383 if isBraking then
```

```
384 if not spec.isBrakeSamplePlaying then
385 g_soundManager:playSample(samples.brake)
386 spec.isBrakeSamplePlaying = true
387 end
388 else
389 if spec.isBrakeSamplePlaying then
390 g_soundManager:stopSample(samples.brake)
391 spec.isBrakeSamplePlaying = false
392 end
393 end
394 end
395
396 -- reverse driving beep
397 if samples.reverseDrive ~= nil then
398 local reverserDirection = self.getReverserDirection == nil and 1 or
    self:getReverserDirection()
399 local isReverseDriving = self:getLastSpeed() > spec.reverseDriveThresho
    self.movingDirection ~= reverserDirection
400 if not g_soundManager:getIsSamplePlaying(samples.reverseDrive) and
    isReverseDriving then
401 g_soundManager:playSample(samples.reverseDrive)
402 elseif not isReverseDriving then
403 g_soundManager:stopSample(samples.reverseDrive)
404 end
405 end
406 end
407 end
408
409 if self.isServer then
410 if not self:getIsAIActive() then
411 if self.lastMovedDistance > 0 then
412 g_currentMission:farmStats(self:getOwnerFarmId()):updateStats("traveled
    self.lastMovedDistance*0.001)
413 end
414 end
415 end
416
417 spec.showTurnOnMotorWarning = false
418 else
419 spec.showTurnOnMotorWarning = accInput ~= 0
420 end
```

```
421 end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
428 function Motorized:onUpdateTick(dt, isActiveForInput, isSelected)
429 local spec = self.spec_motorized
430
431 if self.isServer then
432 -- motor load
433 local loadPercentage = spec.motor:getMotorAppliedTorque() /
    math.max(spec.motor:getMotorAvailableTorque(), 0.0001)
434
435 spec.actualLoadPercentage = loadPercentage
436 spec.smoothedLoadPercentage = 0.95*spec.smoothedLoadPercentage +
    0.05*loadPercentage
437
438 -- force stop of motor if player is far away from vehicle for a
    certain amount of time
439 if not g_currentMission.missionInfo.automaticMotorStartEnabled then
440 if spec.isMotorStarted and not self:getIsAIActive() then
441
442 local isEntered = self.getIsEntered ~= nil and self:getIsEntered()
443 local isControlled = self.getIsControlled ~= nil and
    self:getIsControlled()
444
445 if not isEntered and not isControlled then
446
447 local isPlayerInRange = false
448
449 for _, player in pairs(g_currentMission.players) do
450 if player.isControlled then
451 local distance = calcDistanceFrom(self.rootNode, player.rootNode)
452 if distance < 250 then
453 isPlayerInRange = true
454 break
```

```
455 end
456 end
457 end
458 if not isPlayerInRange then
459 for _, enterable in pairs(g_currentMission.enterables) do
460 if enterable.spec_enterable ~= nil and
    enterable.spec_enterable.isControlled then
461 local distance = calcDistanceFrom(self.rootNode, enterable.rootNode)
462 if distance < 250 then
463 isPlayerInRange = true
464 break
465 end
466 end
467 end
468 end
469
470 if isPlayerInRange then
471 spec.motorStopTimer = spec.motorStopTimerDuration
472 else
473 spec.motorStopTimer = spec.motorStopTimer - dt
474 if spec.motorStopTimer <= 0 then
475 self:stopMotor()
476 end
477 end
478
479 end
480 end
481 end
482
483 if spec.isMotorStarted then
484 self:updateMotorTemperature(dt)
485 end
486 end
487
488 if self.isClient then
489 if self:getIsMotorStarted() then
490 if spec.exhaustParticleSystems ~= nil then
491 for _, ps in pairs(spec.exhaustParticleSystems) do
492 local scale = MathUtil.lerp(spec.exhaustParticleSystems.minScale,
    spec.exhaustParticleSystems.maxScale,
    spec.motor:getEqualizedMotorRpm() / spec.motor:getMaxRpm())
```

```
493 ParticleUtil.setEmitCountScale(spec.exhaustParticleSystems, scale)
494 ParticleUtil.setParticleLifespan(ps, ps.originalLifespan * scale)
495 end
496 end
497
498 if spec.exhaustFlap ~= nil then
499 local minRandom = -0.1
500 local maxRandom = 0.1
501 local angle = MathUtil.lerp(minRandom, maxRandom, math.random()) +
    spec.exhaustFlap.maxRot * (spec.motor:getEqualizedMotorRpm() /
    spec.motor:getMaxRpm())
502 angle = MathUtil.clamp(angle, 0, spec.exhaustFlap.maxRot)
503 setRotation(spec.exhaustFlap.node, angle, 0, 0)
504 end
505
506 if spec.exhaustEffects ~= nil then
507 local lastSpeed = self:getLastSpeed()
508
509 spec.currentDirection = {localDirectionToWorld(self.rootNode, 0, 0,
    1)}
510 if spec.lastDirection == nil then
511 spec.lastDirection = spec.currentDirection
512 end
513
514 local x,_,z = worldDirectionToLocal(self.rootNode,
    spec.lastDirection[1], spec.lastDirection[2], spec.lastDirection[3])
515 local dot = z
516 dot = dot / MathUtil.vector2Length(x,z)
517 local angle = math.acos(dot)
518 if x < 0 then
519 angle = -angle
520 end
521 local steeringPercent = math.abs((angle / dt) /
    spec.exhaustEffectMaxSteeringSpeed)
522 spec.lastDirection = spec.currentDirection
523
524 for _, effect in pairs(spec.exhaustEffects) do
525 local rpmScale = spec.motor:getEqualizedMotorRpm() /
    spec.motor:getMaxRpm()
526 local scale = MathUtil.lerp(effect.minRpmScale, effect.maxRpmScale,
    rpmScale)
527 local forwardXRot = 0
```

```
local forwardZRot = 0
local steerXRot = 0
local steerZRot = 0

local r = MathUtil.lerp(effect.minRpmColor[1], effect.maxRpmColor[1], rpmScale)
local g = MathUtil.lerp(effect.minRpmColor[2], effect.maxRpmColor[2], rpmScale)
local b = MathUtil.lerp(effect.minRpmColor[3], effect.maxRpmColor[3], rpmScale)
local a = MathUtil.lerp(effect.minRpmColor[4], effect.maxRpmColor[4], rpmScale)
setShaderParameter(effect.effectNode, "exhaustColor", r, g, b, a, false)

-- speed rotation
if self.movingDirection == 1 then
local percent = MathUtil.clamp(lastSpeed/effect.maxForwardSpeed, 0, 1)
forwardXRot = effect.xzRotationsForward[1] * percent
forwardZRot = effect.xzRotationsForward[2] * percent
elseif self.movingDirection == -1 then
local percent = MathUtil.clamp(lastSpeed/effect.maxBackwardSpeed, 0, 1)
forwardXRot = effect.xzRotationsBackward[1] * percent
forwardZRot = effect.xzRotationsBackward[2] * percent
end

-- steering rotation
if angle > 0 then
steerXRot = effect.xzRotationsRight[1] * steeringPercent
steerZRot = effect.xzRotationsRight[2] * steeringPercent
elseif angle < 0 then
steerXRot = effect.xzRotationsLeft[1] * steeringPercent
steerZRot = effect.xzRotationsLeft[2] * steeringPercent
end
-- target rotations
local targetXRot = effect.xzRotationsOffset[1] + forwardXRot + steerXRot
local targetZRot = effect.xzRotationsOffset[2] + forwardZRot + steerZRot

```

```
561 -- damping
562 if targetXRot > effect.xRot then
563 effect.xRot = math.min(effect.xRot + 0.003*dt, targetXRot)
564 else
565 effect.xRot = math.max(effect.xRot - 0.003*dt, targetXRot)
566 end
567 if targetZRot > effect.xRot then
568 effect.zRot = math.min(effect.zRot + 0.003*dt, targetZRot)
569 else
570 effect.zRot = math.max(effect.zRot - 0.003*dt, targetZRot)
571 end
572 setShaderParameter(effect.effectNode, "param", effect.xRot,
    effect.zRot, 0, scale, false)
573 end
574 end
575 end
576
577 -- display permanent fuel empty warning if automatic motor start is
    enabled
578 if self:getIsActiveForInput(true) then
579 if g_currentMission.missionInfo.automaticMotorStartEnabled then
580 if not self:getCanMotorRun() then
581 local warning = self:getMotorNotAllowedWarning()
582 if warning ~= nil then
583 g_currentMission:showBlinkingWarning(warning, 500)
584 end
585 else
586 --start motor if fuel got filled
587 if not self:getIsMotorStarted() then
588 self:startMotor(true)
589 end
590 end
591 end
592
593 -- hide / show start motor input action if setting is changed on the
    fly
594 local actionEvent =
    spec.actionEvents[InputAction.TOGGLE_MOTOR_STATE]
595 if actionEvent ~= nil then
596 if not g_currentMission.missionInfo.automaticMotorStartEnabled then
597 local text
```

```
598
599 g_inputBinding:setActionEventActive(actionEvent.actionEventId, true)
600 if self:getIsMotorStarted() then
601 g_inputBinding:setActionEventTextPriority(actionEvent.actionEventId,
    GS_PRIO_VERY_LOW)
602 text = spec.turnOffText
603 else
604 g_inputBinding:setActionEventTextPriority(actionEvent.actionEventId,
    GS_PRIO_VERY_HIGH)
605 text = spec.turnOnText
606 end
607
608 g_inputBinding:setActionEventText(actionEvent.actionEventId, text)
609 else
610 g_inputBinding:setActionEventActive(actionEvent.actionEventId,
    false)
611 end
612 end
613 end
614 end
615 end
```

onDraw

Description

Called on draw

Definition

onDraw(boolean isActiveForInput, boolean isSelected)

Arguments

boolean isActiveForInput	true if vehicle is active for input
boolean isSelected	true if vehicle is selected

Code

```
621 function Motorized:onDraw(isActiveForInput, isSelected)
622 local spec = self.spec_motorized
623 if spec.showTurnOnMotorWarning then
624 g_currentMission:showBlinkingWarning(g_i18n:getText("warning_motorNotSta
    2000)
625 end
626 end
```

loadDifferentials

Description

Load differentials from xml

Definition

loadDifferentials(integer xmlFile, integer configDifferentialIndex)

Arguments

integer xmlFile	id of xml object
integer configDifferentialIndex	index of differential config

Code

```
632 function Motorized:loadDifferentials(xmlFile, configDifferentialIndex)
633 local key,_ = ConfigurationUtil.getXMLConfigurationKey(xmlFile,
    configDifferentialIndex,
    "vehicle.motorized.differentialConfigurations.differentialConfiguration"
    "vehicle.motorized.differentials", "differentials")
634
635 local spec = self.spec_motorized
636
637 spec.differentials = {}
638 if self.isServer and spec.motorizedNode ~= nil then
639 local i = 0
640 while true do
641 local key = string.format(key..".differentials.differential(%d)", i)
642 if not hasXMLProperty(xmlFile, key) then
643 break
644 end
645 local torqueRatio = Utils.getNoNil(getXMLFloat(xmlFile,
    key.."#torqueRatio"), 0.5)
646 local maxSpeedRatio = Utils.getNoNil(getXMLFloat(xmlFile,
    key.."#maxSpeedRatio"), 1.3)
647
648 local indices = {-1, -1}
649 local indexIsWheel = {false, false}
650 for j=1,2 do
651 local wheelIndex = getXMLInt(xmlFile, key ..
    string.format("#wheelIndex%d", j))
652 if wheelIndex ~= nil then
653 if self:getWheelFromWheelIndex(wheelIndex) ~= nil then
654 indices[j] = wheelIndex
655 indexIsWheel[j] = true
656 else
657 g_logManager:xmlWarning(self.configFileName, "Unable to find wheelIndex
    '%d' for differential '%s' (Indices start at 1)", wheelIndex, key)
658 end
659 else
660 local diffIndex = getXMLInt(xmlFile, key ..
    string.format("#differentialIndex%d", j))
661 if diffIndex ~= nil then
662 indices[j] = diffIndex - 1
```

```
663 indexIsWheel[j] = false
664
665 if diffIndex == 0 then
666 g_logManager:xmlWarning(self.configFileName, "Unable to find
    differentialIndex '0' for differential '%s' (Indices start at 1)", key)
667 end
668 end
669 end
670 end
671
672 if indices[1] ~= -1 and indices[2] ~= -1 then
673 table.insert(spec.differentials, {
674 torqueRatio = torqueRatio,
675 maxSpeedRatio = maxSpeedRatio,
676 diffIndex1 = indices[1],
677 diffIndex1IsWheel = indexIsWheel[1],
678 diffIndex2 = indices[2],
679 diffIndex2IsWheel = indexIsWheel[2] } )
680 end
681
682 i = i + 1
683 end
684
685 if #spec.differentials == 0 then
686 g_logManager:xmlWarning(self.configFileName, "No differentials defined")
687 end
688 end
689 end
```

loadMotor

Description

Load motor from xml file

Definition

loadMotor(integer xmlFile, integer motorId)

Arguments

integer xmlFile id of xml object

integer motorId index of motor configuration

Code

```
695 function Motorized:loadMotor(xmlFile, motorId)
696 local key, motorId = ConfigurationUtil.getXMLConfigurationKey(xmlFile, m
    "vehicle.motorized.motorConfigurations.motorConfiguration", "vehicle.mot
697
```

```
698 local spec = self.spec_motorized
699
700 local fallbackConfigKey = "vehicle.motorized.motorConfigurations.motorCo
701 local fallbackOldKey = "vehicle"
702
703 spec.motorType = ConfigurationUtil.getConfigurationValue(xmlFile, key, "
    getXMLString, "vehicle", fallbackConfigKey, fallbackOldKey)
704 spec.motorStartAnimation = ConfigurationUtil.getConfigurationValue(xmlFi
    "#startAnimationName", getXMLString, "vehicle", fallbackConfigKey, fallb
705
706 spec.fuelCapacity = ConfigurationUtil.getConfigurationValue(xmlFile, key
    getXMLFloat, 500, fallbackConfigKey, fallbackOldKey)
707 spec.consumerConfigurationIndex = ConfigurationUtil.getConfigurationValu
    "#consumerConfigurationIndex", "", getXMLInt, 1, fallbackConfigKey, fall
708
709 local wheelKey, _ = ConfigurationUtil.getXMLConfigurationKey(xmlFile,
    self.configurations["wheel"], "vehicle.wheels.wheelConfigurations.wheelC
    "vehicle.wheels", "wheels")
710
711 ObjectChangeUtil.updateObjectChanges(xmlFile,
    "vehicle.motorized.motorConfigurations.motorConfiguration", motorId, sel
712
713 local motorMinRpm = ConfigurationUtil.getConfigurationValue(xmlFile, key
    getXMLFloat, 1000, fallbackConfigKey, fallbackOldKey)
714 local motorMaxRpm = ConfigurationUtil.getConfigurationValue(xmlFile, key
    getXMLFloat, 1800, fallbackConfigKey, fallbackOldKey)
715 local minSpeed = ConfigurationUtil.getConfigurationValue(xmlFile, key, "
    getXMLFloat, 1, fallbackConfigKey, fallbackOldKey)
716 local maxForwardSpeed = ConfigurationUtil.getConfigurationValue(xmlFile,
    "#maxForwardSpeed", getXMLFloat, nil, fallbackConfigKey, fallbackOldKey)
717 local maxBackwardSpeed = ConfigurationUtil.getConfigurationValue(xmlFile
    "#maxBackwardSpeed", getXMLFloat, nil, fallbackConfigKey, fallbackOldKey
718 if maxForwardSpeed ~= nil then
719 maxForwardSpeed = maxForwardSpeed/3.6
720 end
721 if maxBackwardSpeed ~= nil then
722 maxBackwardSpeed = maxBackwardSpeed/3.6
723 end
724
725 local maxWheelSpeed = ConfigurationUtil.getConfigurationValue(xmlFile, v
    "#maxForwardSpeed", getXMLFloat, nil, nil, "vehicle.wheels")
726 if maxWheelSpeed ~= nil then
727 maxForwardSpeed = maxWheelSpeed/3.6
```

```
728 end
729 local accelerationLimit = ConfigurationUtil.getConfigurationValue(xmlFil
    "#accelerationLimit", getXMLFloat, 2.0, fallbackConfigKey, fallbackOldKe
730
731 local brakeForce = ConfigurationUtil.getConfigurationValue(xmlFile, key,
    "#brakeForce", getXMLFloat, 10, fallbackConfigKey, fallbackOldKey)*2
732 local lowBrakeForceScale = ConfigurationUtil.getConfigurationValue(xmlFi
    "#lowBrakeForceScale", getXMLFloat, 0.5, fallbackConfigKey, fallbackOldK
733 local lowBrakeForceSpeedLimit = ConfigurationUtil.getConfigurationValue
    ".motor", "#lowBrakeForceSpeedLimit", getXMLFloat, 1, fallbackConfigKey,
734 local torqueScale = ConfigurationUtil.getConfigurationValue(xmlFile, key
    "#torqueScale", getXMLFloat, 1, fallbackConfigKey, fallbackOldKey)
735 local ptoMotorRpmRatio = ConfigurationUtil.getConfigurationValue(xmlFile
    "#ptoMotorRpmRatio", getXMLFloat, 4, fallbackConfigKey, fallbackOldKey)
736
737 local minForwardGearRatio = ConfigurationUtil.getConfigurationValue(xmlF
    ".transmission", "#minForwardGearRatio", getXMLFloat, nil, fallbackConfi
738 local maxForwardGearRatio = ConfigurationUtil.getConfigurationValue(xmlF
    ".transmission", "#maxForwardGearRatio", getXMLFloat, nil, fallbackConfi
739 local minBackwardGearRatio = ConfigurationUtil.getConfigurationValue(xml
    ".transmission", "#minBackwardGearRatio", getXMLFloat, nil, fallbackConf
740 local maxBackwardGearRatio = ConfigurationUtil.getConfigurationValue(xml
    ".transmission", "#maxBackwardGearRatio", getXMLFloat, nil, fallbackConf
741 local gearChangeTime = ConfigurationUtil.getConfigurationValue(xmlFile,
    "#gearChangeTime", getXMLFloat, nil, fallbackConfigKey, fallbackOldKey)
742 local autoGearChangeTime = ConfigurationUtil.getConfigurationValue(xmlFi
    ".transmission", "#autoGearChangeTime", getXMLFloat, nil, fallbackConfig
743 local axleRatio = ConfigurationUtil.getConfigurationValue(xmlFile, key,
    "#axleRatio", getXMLFloat, 1.0, fallbackConfigKey, fallbackOldKey)
744
745
746 if maxForwardGearRatio == nil or minForwardGearRatio == nil then
747 minForwardGearRatio = nil
748 maxForwardGearRatio = nil
749 else
750 minForwardGearRatio = minForwardGearRatio * axleRatio
751 maxForwardGearRatio = maxForwardGearRatio * axleRatio
752 end
753 if minBackwardGearRatio == nil or maxBackwardGearRatio == nil then
754 minBackwardGearRatio = nil
755 maxBackwardGearRatio = nil
756 else
757 minBackwardGearRatio = minBackwardGearRatio * axleRatio
```

```
758 maxBackwardGearRatio = maxBackwardGearRatio * axleRatio
759 end
760
761 -- Read forward gear ratios
762 local forwardGearRatios
763 if minForwardGearRatio == nil then
764 forwardGearRatios = self:loadGears(xmlFile, "forwardGear", key, motorId,
    fallbackOldKey, motorMaxRpm, axleRatio)
765 if forwardGearRatios == nil then
766 print("Warning: Missing forward gear ratios for motor in '"..self.config
767 forwardGearRatios = {1}
768 end
769 end
770 -- Read backward gear ratios
771 local backwardGearRatios
772 if minBackwardGearRatio == nil then
773 backwardGearRatios = self:loadGears(xmlFile, "backwardGear", key, motorI
    fallbackOldKey, motorMaxRpm, axleRatio)
774 if backwardGearRatios == nil then
775 print("Warning: Missing backward gear ratios for motor in '"..self.confi
776 backwardGearRatios = {1}
777 end
778 end
779
780 --local maxTorque = 0
781 local torqueCurve = AnimCurve:new(linearInterpolator1)
782 local torqueI = 0
783 local torqueBase = fallbackOldKey..".motor.torque" -- fallback to old mo
784 if key ~= nil and hasXMLProperty(xmlFile, fallbackConfigKey..".motor.tor
    default motor configuration
785 torqueBase = fallbackConfigKey..".motor.torque"
786 end
787 if key ~= nil and hasXMLProperty(xmlFile, key..".motor.torque(0)") then
    motor configuration
788 torqueBase = key..".motor.torque"
789 end
790
791 while true do
792 local torqueKey = string.format(torqueBase.."(%d)", torqueI)
793 local normRpm = getXMLFloat(xmlFile, torqueKey.."#normRpm")
794 local rpm
```

```
795 if normRpm == nil then
796 rpm = getXMLFloat(xmlFile, torqueKey.."#rpm")
797 else
798 rpm = normRpm * motorMaxRpm
799 end
800 local torque = getXMLFloat(xmlFile, torqueKey.."#torque")
801 if torque == nil or rpm == nil then
802 break
803 end
804 torqueCurve:addKeyframe({torque*torqueScale, time = rpm})
805 torqueI = torqueI +1
806 end
807
808 spec.motor = VehicleMotor:new(self, motorMinRpm, motorMaxRpm, maxForward
maxBackwardSpeed, torqueCurve, brakeForce, forwardGearRatios, backwardGe
minForwardGearRatio, maxForwardGearRatio, minBackwardGearRatio, maxBackw
ptoMotorRpmRatio, minSpeed)
809
810 local rotInertia = ConfigurationUtil.getConfigurationValue(xmlFile, key,
"#rotInertia", getXMLFloat, spec.motor:getRotInertia(), fallbackConfigKe
811 local dampingRateFullThrottle = ConfigurationUtil.getConfigurationValue
".motor", "#dampingRateFullThrottle", getXMLFloat, spec.motor:getDamping
fallbackConfigKey, fallbackOldKey)
812 local dampingRateZeroThrottleClutchEngaged = ConfigurationUtil.getConfig
key, ".motor", "#dampingRateZeroThrottleClutchEngaged", getXMLFloat,
spec.motor:getDampingRateZeroThrottleClutchEngaged(), fallbackConfigKey,
813 local dampingRateZeroThrottleClutchDisengaged =
ConfigurationUtil.getConfigurationValue(xmlFile, key, ".motor",
"#dampingRateZeroThrottleClutchDisengaged", getXMLFloat,
spec.motor:getDampingRateZeroThrottleClutchDisengaged(), fallbackConfigK
814 spec.motor:setRotInertia(rotInertia)
815 spec.motor:setDampingRateFullThrottle(dampingRateFullThrottle)
816 spec.motor:setDampingRateZeroThrottleClutchEngaged(dampingRateZeroThrott
817 spec.motor:setDampingRateZeroThrottleClutchDisengaged(dampingRateZeroThr
818 spec.motor:setLowBrakeForce(lowBrakeForceScale, lowBrakeForceSpeedLimit)
819 spec.motor:setAccelerationLimit(accelerationLimit)
820
821 if gearChangeTime ~= nil then
822 spec.motor:setGearChangeTime(gearChangeTime*1000)
823 end
824 if autoGearChangeTime ~= nil then
825 spec.motor:setAutoGearChangeTime(autoGearChangeTime*1000)
826 end
```

```
827 end
```

loadExhaustEffects

Description

Loading of exhaust effects from xml file

Definition

loadExhaustEffects(integer xmlFile, table exhaustEffects)

Arguments

integer xmlFile	id of xml object
table	exhaustEffects table to ass exhaustEffects

Code

```
861 function Motorized:loadExhaustEffects(xmlFile)
862 local spec = self.spec_motorized
863
864 spec.exhaustParticleSystems = {}
865 local exhaustParticleSystemCount = Utils.getNoNil(getXMLInt(xmlFile,
    "vehicle.motorized.exhaustParticleSystems#count"), 0)
866 for i=1, exhaustParticleSystemCount do
867 local namei =
    string.format("vehicle.motorized.exhaustParticleSystems.exhaustParticleS
    i)
868 local ps = {}
869 ParticleUtil.loadParticleSystem(xmlFile, ps, namei, self.components, fal
    self.baseDirectory)
870 ps.minScale = Utils.getNoNil(getXMLFloat(xmlFile,
    "vehicle.motorized.exhaustParticleSystems#minScale"), 0.5)
871 ps.maxScale = Utils.getNoNil(getXMLFloat(xmlFile,
    "vehicle.motorized.exhaustParticleSystems#maxScale"), 1)
872 table.insert(spec.exhaustParticleSystems, ps)
873 end
874 if #spec.exhaustParticleSystems == 0 then
875 spec.exhaustParticleSystems = nil
876 end
877
878 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
    "vehicle.motorized.exhaustFlap#index", "vehicle.motorized.exhaustFlap#no
    FS17 to FS19
879
880 local exhaustFlapIndex = getXMLString(xmlFile,
    "vehicle.motorized.exhaustFlap#node")
881 if exhaustFlapIndex ~= nil then
882 spec.exhaustFlap = {}
883 spec.exhaustFlap.node = I3DUtil.indexToObject(self.components, exhaustFl
    self.i3dMappings)
```

```
884 spec.exhaustFlap.maxRot = MathUtil.degToRad(Utils.getNoNil(getXMLFloat(x
    "vehicle.motorized.exhaustFlap#maxRot"),0))
885 end
886
887 spec.exhaustEffects = {}
888 local i = 0
889 while true do
890 local key = string.format("vehicle.motorized.exhaustEffects.exhaustEffec
    i)
891 if not hasXMLProperty(xmlFile, key) then
892 break
893 end
894
895 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName, key.."#
    key.."#node") --FS17 to FS19
896
897 local linkNode = I3DUtil.indexToObject(self.components, getXMLString(xml
    key.."#node"), self.i3dMappings)
898 local filename = getXMLString(xmlFile, key .. "#filename")
899 if filename ~= nil and linkNode ~= nil then
900 local i3dNode = g_i3DManager:loadSharedI3DFile(filename, self.baseDirect
    false, false, false)
901 if i3dNode ~= 0 then
902 local node = getChildAt(i3dNode, 0)
903 if getHasShaderParameter(node, "param") then
904 local effect = {}
905 effect.effectNode = node
906 effect.node = linkNode
907 effect.filename = filename
908 link(effect.node, effect.effectNode)
909 setVisibility(effect.effectNode, false)
910 delete(i3dNode)
911
912 effect.minRpmColor =
    StringUtil.getVectorNFromString(Utils.getNoNil(getXMLString(xmlFile,
    key.."#minRpmColor"), "0 0 0 1"), 4)
913 effect.maxRpmColor =
    StringUtil.getVectorNFromString(Utils.getNoNil(getXMLString(xmlFile,
    key.."#maxRpmColor"), "0.0384 0.0359 0.0627 2.0"), 4)
914 effect.minRpmScale = Utils.getNoNil(getXMLFloat(xmlFile, key.."#minRpmSc
    0.25)
915 effect.maxRpmScale = Utils.getNoNil(getXMLFloat(xmlFile, key.."#maxRpmSc
    0.95)
```

```
916 effect.maxForwardSpeed = Utils.getNoNil(getXMLFloat(xmlFile,
    key.."#maxForwardSpeed"), math.ceil(spec.motor:getMaximumForwardSpeed()*
917 effect.maxBackwardSpeed = Utils.getNoNil(getXMLFloat(xmlFile,
    key.."#maxBackwardSpeed"), math.ceil(spec.motor:getMaximumBackwardSpeed(
918
919 effect.xzRotationsOffset =
    StringUtil.getRadiansFromString(Utils.getNoNil(getXMLString(xmlFile,
    key.."#xzRotationsOffset"), "0 0"), 2)
920 effect.xzRotationsForward =
    StringUtil.getRadiansFromString(Utils.getNoNil(getXMLString(xmlFile,
    key.."#xzRotationsForward"), "0 0"), 2)
921 effect.xzRotationsBackward =
    StringUtil.getRadiansFromString(Utils.getNoNil(getXMLString(xmlFile,
    key.."#xzRotationsBackward"), "0 0"), 2)
922 effect.xzRotationsLeft =
    StringUtil.getRadiansFromString(Utils.getNoNil(getXMLString(xmlFile,
    key.."#xzRotationsLeft"), "0 0"), 2)
923 effect.xzRotationsRight =
    StringUtil.getRadiansFromString(Utils.getNoNil(getXMLString(xmlFile,
    key.."#xzRotationsRight"), "0 0"), 2)
924
925 effect.xRot = 0
926 effect.zRot = 0
927
928 table.insert(spec.exhaustEffects, effect)
929 end
930 end
931 end
932 i = i + 1
933 end
934 spec.exhaustEffectMaxSteeringSpeed = 0.001
935 end
```

loadSounds

Description

Load sounds from xml file

Definition

loadSounds(integer xmlFile)

Arguments

integer xmlFile id of xml object

Code

```
941 function Motorized:loadSounds(xmlFile, motorId)
942 if self.isClient then
943 local spec = self.spec_motorized
944
```

```
945 local baseString = "vehicle.motorized.sounds"
946 spec.samples = {}
947
948 spec.samples.motorStart =
    g_soundManager:loadSampleFromXML(xmlFile, baseString,
    "motorStart", self.baseDirectory, self.components, 1,
    AudioGroup.VEHICLE, self.i3dMappings, self)
949 spec.samples.motorStop =
    g_soundManager:loadSampleFromXML(xmlFile, baseString,
    "motorStop", self.baseDirectory, self.components, 1,
    AudioGroup.VEHICLE, self.i3dMappings, self)
950 spec.samples.gearbox = g_soundManager:loadSampleFromXML(xmlFile,
    baseString, "gearbox", self.baseDirectory, self.components, 0,
    AudioGroup.VEHICLE, self.i3dMappings, self)
951 spec.samples.retarder = g_soundManager:loadSampleFromXML(xmlFile,
    baseString, "retarder", self.baseDirectory, self.components, 0,
    AudioGroup.VEHICLE, self.i3dMappings, self)
952
953 spec.motorSamples = {}
954 local i = 0
955 while true do
956 local sample = g_soundManager:loadSampleFromXML(xmlFile,
    baseString, string.format("motor(%d)", i), self.baseDirectory,
    self.components, 0, AudioGroup.VEHICLE, self.i3dMappings, self)
957 if sample == nil then
958 break
959 end
960
961 table.insert(spec.motorSamples, sample)
962 i = i + 1
963 end
964
965 spec.samples.airCompressorStart =
    g_soundManager:loadSampleFromXML(xmlFile, baseString,
    "airCompressorStart", self.baseDirectory, self.components, 1,
    AudioGroup.VEHICLE, self.i3dMappings, self)
966 spec.samples.airCompressorStop =
    g_soundManager:loadSampleFromXML(xmlFile, baseString,
    "airCompressorStop", self.baseDirectory, self.components, 1,
    AudioGroup.VEHICLE, self.i3dMappings, self)
967 spec.samples.airCompressorRun =
    g_soundManager:loadSampleFromXML(xmlFile, baseString,
    "airCompressorRun", self.baseDirectory, self.components, 0,
    AudioGroup.VEHICLE, self.i3dMappings, self)
968
```

```
969 spec.samples.compressedAir =
    g_soundManager:loadSampleFromXML(xmlFile, baseString,
    "compressedAir", self.baseDirectory, self.components, 1,
    AudioGroup.VEHICLE, self.i3dMappings, self)
970 if spec.samples.compressedAir ~= nil then
971 spec.samples.compressedAir.brakeTime = 0
972 spec.samples.compressedAir.lastBrakeTime = 0
973 end
974
975 spec.samples.airRelease =
    g_soundManager:loadSampleFromXML(xmlFile, baseString,
    "airRelease", self.baseDirectory, self.components, 1,
    AudioGroup.VEHICLE, self.i3dMappings, self)
976
977 spec.samples.reverseDrive =
    g_soundManager:loadSampleFromXML(xmlFile, baseString,
    "reverseDrive", self.baseDirectory, self.components, 0,
    AudioGroup.VEHICLE, self.i3dMappings, self)
978 spec.reverseDriveThreshold = Utils.getNoNil(getXMLFloat(xmlFile,
    "vehicle.motorized.reverseDriveSound#threshold"), 4)
979
980 spec.brakeCompressor = {}
981 spec.brakeCompressor.capacity =
    Utils.getNoNil(getXMLFloat(xmlFile,
    "vehicle.motorized.brakeCompressor#capacity"), 6)
982 spec.brakeCompressor.refillFilllevel =
    math.min(spec.brakeCompressor.capacity,
    Utils.getNoNil(getXMLFloat(xmlFile,
    "vehicle.motorized.brakeCompressor#refillFillLevel"),
    spec.brakeCompressor.capacity/2))
983 spec.brakeCompressor.fillSpeed =
    Utils.getNoNil(getXMLFloat(xmlFile,
    "vehicle.motorized.brakeCompressor#fillSpeed"), 0.6) / 1000
984 spec.brakeCompressor.fillLevel = 0
985 spec.brakeCompressor.doFill = true
986
987 spec.isBrakeSamplePlaying = false
988 spec.samples.brake = g_soundManager:loadSampleFromXML(xmlFile,
    baseString, "brake", self.baseDirectory, self.components, 0,
    AudioGroup.VEHICLE, self.i3dMappings, self)
989
990 spec.compressionSoundTime = 0
991 end
992 end
```

getIsMotorStarted

Description

Returns if motor is stated

Definition

getIsMotorStarted()

Return Values

boolean isStarted motor is started

Code

```
1052  function Motorized:getIsMotorStarted()
1053  return self.spec_motorized.isMotorStarted
1054  end
```

startMotor

Description

Start motor

Definition

startMotor(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
1096  function Motorized:startMotor(noEventSend)
1097  if noEventSend == nil or noEventSend == false then
1098  if g_server ~= nil then
1099  g_server:broadcastEvent(SetMotorTurnedOnEvent:new(self, true), nil, nil
      self)
1100  else
1101  g_client:getServerConnection():sendEvent(SetMotorTurnedOnEvent:new(self
      true))
1102  end
1103  end
1104  local spec = self.spec_motorized
1105  if not spec.isMotorStarted then
1106  spec.isMotorStarted = true
1107
1108  if self.isClient then
1109  if spec.exhaustParticleSystems ~= nil then
1110  for _, ps in pairs(spec.exhaustParticleSystems) do
1111  ParticleUtil.setEmittingState(ps, true)
1112  end
1113  end
1114  if spec.exhaustEffects ~= nil then
1115  for _, effect in pairs(spec.exhaustEffects) do
1116  setVisibility(effect.effectNode, true)
1117  effect.xRot = effect.xzRotationsOffset[1]
1118  effect.zRot = effect.xzRotationsOffset[2]
```

```
1119  setShaderParameter(effect.effectNode, "param", effect.xRot, effect.zRot
      0, 0, false)
1120
1121  local color = effect.minRpmColor
1122  setShaderParameter(effect.effectNode, "exhaustColor", color[1],
      color[2], color[3], color[4], false)
1123  end
1124  end
1125
1126  g_soundManager:stopSample(spec.samples.motorStop)
1127
1128  g_soundManager:playSample(spec.samples.motorStart)
1129  g_soundManager:playSamples(spec.motorSamples, 0,
      spec.samples.motorStart)
1130  g_soundManager:playSample(spec.samples.gearbox, 0,
      spec.samples.motorStart)
1131  g_soundManager:playSample(spec.samples.retarder, 0,
      spec.samples.motorStart)
1132
1133  g_animationManager:startAnimations(spec.animationNodes)
1134
1135  if spec.motorStartAnimation ~= nil then
1136  self:playAnimation(spec.motorStartAnimation, 1, nil, true)
1137  end
1138  end
1139
1140  spec.motorStartTime = g_currentMission.time + spec.motorStartDuration
1141  spec.compressionSoundTime = g_currentMission.time + math.random(5000,
      20000)
1142  spec.lastRoundPerMinute = 0
1143
1144  SpecializationUtil.raiseEvent(self, "onStartMotor")
1145  end
1146
1147  if self.setDashboardsDirty ~= nil then
1148  self:setDashboardsDirty()
1149  end
1150
1151  end
```

stopMotor

Description

Stop motor

Definition

stopMotor(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
1156  function Motorized:stopMotor(noEventSend)
1157  if noEventSend == nil or noEventSend == false then
1158  if g_server ~= nil then
1159  g_server:broadcastEvent(SetMotorTurnedOnEvent:new(self, false), nil,
      nil, self)
1160  else
1161  g_client:getServerConnection():sendEvent(SetMotorTurnedOnEvent:new(self
      false))
1162  end
1163  end
1164
1165  local spec = self.spec_motorized
1166  if spec.isMotorStarted then
1167  spec.isMotorStarted = false
1168
1169  Motorized.turnOffImplement(self)
1170
1171  if self.isClient then
1172  if spec.exhaustParticleSystems ~= nil then
1173  for _, ps in pairs(spec.exhaustParticleSystems) do
1174  ParticleUtil.setEmittingState(ps, false)
1175  end
1176  end
1177
1178  if spec.exhaustEffects ~= nil then
1179  for _, effect in pairs(spec.exhaustEffects) do
1180  setVisibility(effect.effectNode, false)
1181  end
1182  end
1183  if spec.exhaustFlap ~= nil then
1184  setRotation(spec.exhaustFlap.node, 0, 0, 0)
1185  end
1186
1187  g_soundManager:stopSample(spec.samples.motorStart)
1188  g_soundManager:playSample(spec.samples.motorStop)
1189  g_soundManager:stopSamples(spec.motorSamples)
```

```
1190  g_soundManager:stopSample(spec.samples.gearbox)
1191  g_soundManager:stopSample(spec.samples.retarder)
1192
1193  g_soundManager:stopSample(spec.samples.airCompressorStart)
1194  g_soundManager:stopSample(spec.samples.airCompressorStop)
1195  g_soundManager:stopSample(spec.samples.airCompressorRun)
1196
1197  g_soundManager:stopSample(spec.samples.compressedAir)
1198  g_soundManager:stopSample(spec.samples.airRelease)
1199
1200  g_soundManager:stopSample(spec.samples.reverseDrive)
1201  g_soundManager:stopSample(spec.samples.brake)
1202  spec.isBrakeSamplePlaying = false
1203
1204  g_animationManager:stopAnimations(spec.animationNodes)
1205
1206  if spec.motorStartAnimation ~= nil then
1207  self:playAnimation(spec.motorStartAnimation, -1, nil, true)
1208  end
1209  end
1210
1211  SpecializationUtil.raiseEvent(self, "onStopMotor")
1212  end
1213
1214  if self.setDashboardsDirty ~= nil then
1215  self:setDashboardsDirty()
1216  end
1217  end
```

addToPhysics

Description

Add to physics

Definition

addToPhysics()

Return Values

boolean success success

Code

```
1442  function Motorized:addToPhysics(superFunc)
1443  if not superFunc(self) then
1444  return false
1445  end
1446
```

```
1447  if self.isServer then
1448  local spec = self.spec_motorized
1449
1450  if spec.motorizedNode ~= nil then
1451  if next(spec.differentials) ~= nil then
1452
1453  for _, differential in pairs(spec.differentials) do
1454  local diffIndex1 = differential.diffIndex1
1455  local diffIndex2 = differential.diffIndex2
1456
1457  if differential.diffIndex1IsWheel then
1458  diffIndex1 = self:getWheelFromWheelIndex(diffIndex1).wheelShape
1459  end
1460  if differential.diffIndex2IsWheel then
1461  diffIndex2 = self:getWheelFromWheelIndex(diffIndex2).wheelShape
1462  end
1463
1464  addDifferential( spec.motorizedNode,
1465  diffIndex1,
1466  differential.diffIndex1IsWheel,
1467  diffIndex2,
1468  differential.diffIndex2IsWheel,
1469  differential.torqueRatio,
1470  differential.maxSpeedRatio )
1471  end
1472
1473  self:updateMotorProperties()
1474
1475  controlVehicle(spec.motorizedNode, 0.0, 0.0, 0.0, 0.0,
     math.huge, 0.0, 0.0, 0.0, 0.0)
1476  end
1477  end
1478  end
1479  return true
1480  end
```

getIsOperating

Description

Returns if vehicle is operating

Definition

getIsOperating()

Return Values

boolean isOperating is operating

Code

```
1493  function Motorized:getIsOperating(superFunc)
1494  return superFunc(self) or self:getIsMotorStarted()
1495  end
```

getDeactivateOnLeave

Description

Returns if vehicle deactivates on leave

Definition

getDeactivateOnLeave()

Return Values

boolean deactivate vehicle deactivates on leave

Code

```
1500  function Motorized:getDeactivateOnLeave(superFunc)
1501  return superFunc(self) and
      g_currentMission.missionInfo.automaticMotorStartEnabled
1502  end
```

onEnterVehicle

Description

Called on enter vehicle

Definition

onEnterVehicle(boolean isControlling)

Arguments

boolean isControlling is player controlling the vehicle

Code

```
1548  function Motorized:onEnterVehicle(isControlling)
1549  if g_currentMission.missionInfo.automaticMotorStartEnabled then
1550  if self:getCanMotorRun() then
1551  self:startMotor(true)
1552  end
1553  end
1554
1555  --release sound of hand brake
1556  local samples = self.spec_motorized.samples
1557  if samples.compressedAir ~= nil then
1558  g_soundManager:playSample(samples.compressedAir,
      math.random(500, 2000))
1559  end
1560  end
```

onLeaveVehicle

Description

Called on leaving the vehicle

Definition

onLeaveVehicle()

Code

```
1564  function Motorized:onLeaveVehicle()
1565  local spec = self.spec_motorized
1566  if self:getStopMotorOnLeave() and
      g_currentMission.missionInfo.automaticMotorStartEnabled then
1567  self:stopMotor(true)
1568  end
1569
1570  if self.isServer then
1571  if spec.motorizedNode ~= nil then
1572  controlVehicle(spec.motorizedNode, 0.0, 0.0, 0.0, 0.0,
      math.huge, 0.0, 0.0, 0.0, 0.0)
1573  end
1574  end
1575  end
```

turnOffImplement

Description

Turn off implement and childs of implement

Definition

turnOffImplement(table object)

Arguments

table object object to turn off

Code

```
1623  function Motorized.turnOffImplement(object)
1624  if object.setIsTurnedOn ~= nil then
1625  object:setIsTurnedOn(false, true)
1626  end
1627  if object.attachedImplements ~= nil then
1628  for _,implement in pairs(object.attachedImplements) do
1629  if implement.object ~= nil then
1630  Motorized.turnOffImplement(implement.object)
1631  end
1632  end
1633  end
1634  end
```

Mountable

Description

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
64  function Mountable:onDelete()
65  local spec = self.spec_mountable
66
67  if spec.dynamicMountJointIndex ~= nil then
68  removeJointBreakReport(spec.dynamicMountJointIndex)
69  removeJoint(spec.dynamicMountJointIndex)
70  end
71  if spec.dynamicMountObject ~= nil then
72  spec.dynamicMountObject:removeDynamicMountedObject(self, true)
73  end
74  if spec.dynamicMountTriggerId ~= nil then
75  removeTrigger(spec.dynamicMountTriggerId)
76  end
77  end
```

onPostAttach

Description

Called if vehicle gets attached

Definition

onPostAttach(table attacherVehicle, integer inputJointDescIndex, integer jointDescIndex)

Arguments

table	attacherVehicle	attacher vehicle
integer	inputJointDescIndex	index of input attacher joint
integer	jointDescIndex	index of attacher joint it gets attached to

Code

```
240  function Mountable:onPostAttach(attacherVehicle,
     inputJointDescIndex, jointDescIndex)
241  self:unmountDynamic()
242  end
```

Mower

Description

Class for all mowers

initSpecialization

Description

Called on specialization initializing

Definition

initSpecialization()

Code

```
17  function Mower.initSpecialization()
18  g_workAreaTypeManager:addWorkAreaType("mower", false)
```

```
19  end
```

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
25  function Mower.prerequisitesPresent(specializations)
26  return SpecializationUtil.hasSpecialization(WorkArea,
    specializations) and
    SpecializationUtil.hasSpecialization(TurnOnVehicle,
    specializations)
27  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
61  function Mower:onLoad(savegame)
62  local spec = self.spec_mower
63
64  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.mowerEffects.mowerEffect",
    "vehicle.mower.dropEffects.dropEffect") --FS17 to FS19
65  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.mowerEffects.mowerEffect#mowerCutArea",
    "vehicle.mower.dropEffects.dropEffect#dropAreaIndex") --FS17 to
    FS19
66  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.turnedOnRotationNodes.turnedOnRotationNode#type",
    "vehicle.mower.turnOnNodes.turnOnNode", "mower") --FS17 to FS19
67  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.mowerStartSound",
    "vehicle.turnOnVehicle.sounds.start") --FS17 to FS19
68  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.mowerStopSound",
    "vehicle.turnOnVehicle.sounds.stop") --FS17 to FS19
```

```
69  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.mowerSound",
    "vehicle.turnOnVehicle.sounds.work") --FS17 to FS19
70
71  if self.isClient then
72  spec.animationNodes =
    g_animationManager:loadAnimations(self.xmlFile,
    "vehicle.mower.animationNodes", self.components, self,
    self.i3dMappings)
73
74  spec.dropEffects = {}
75  local i = 0
76  while true do
77  local key =
    string.format("vehicle.mower.dropEffects.dropEffect(%d)", i)
78  if not hasXMLProperty(self.xmlFile, key) then
79  break
80  end
81  local effects = g_effectManager:loadEffect(self.xmlFile, key,
    self.components, self, self.i3dMappings)
82  if effects ~= nil then
83  local dropEffect = {}
84  dropEffect.effects = effects
85  dropEffect.dropAreaIndex = Utils.getNoNil(getXMLInt(self.xmlFile,
    key .. "#dropAreaIndex"), 1)
86  dropEffect.workAreaIndex = getXMLInt(self.xmlFile, key ..
    "#workAreaIndex")
87  if self.spec_workArea.workAreas[dropEffect.dropAreaIndex] == nil
    then
88  g_logManager:xmlWarning(self.configFileName, "Invalid
    dropAreaIndex '%s' in '%s'", dropEffect.dropAreaIndex, key)
89  dropEffect.dropAreaIndex = nil
90  end
91  dropEffect.activeTime = -1
92  dropEffect.activeTimeDuration = 750
93  dropEffect.isActive = false
94  dropEffect.isActiveSent = false
95  table.insert(spec.dropEffects, dropEffect)
96  end
97  i = i + 1
98  end
99  end
100
```

```
101 if spec.dropAreas == nil then
102 spec.dropAreas = {}
103 end
104
105 spec.fruitTypeConverters = {}
106 local converter = getXMLString(self.xmlFile,
    "vehicle.mower#fruitTypeConverter")
107 if converter ~= nil then
108 local data = g_fruitTypeManager:getConverterDataByName(converter)
109 if data ~= nil then
110 for input, converted in pairs(data) do
111 spec.fruitTypeConverters[input] = converted
112 end
113 end
114 else
115 print(string.format("Warning: Missing fruit type converter in
    '%s'", self.configFileName))
116 end
117
118 spec.fillUnitIndex = getXMLFloat(self.xmlFile,
    "vehicle.mower#fillUnitIndex")
119 spec.pickupFillScale = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.mower#pickupFillScale"), 1)
120
121 spec.toggleWindrowDropEnableText = getXMLString(self.xmlFile,
    "vehicle.mower.toggleWindrowDrop#enableText")
122 spec.toggleWindrowDropDisableText = getXMLString(self.xmlFile,
    "vehicle.mower.toggleWindrowDrop#disableText")
123 if spec.toggleWindrowDropEnableText ~= nil then
124 spec.toggleWindrowDropEnableText =
    g_i18n:convertText(spec.toggleWindrowDropEnableText,
    self.customEnvironment)
125 end
126 if spec.toggleWindrowDropDisableText ~= nil then
127 spec.toggleWindrowDropDisableText =
    g_i18n:convertText(spec.toggleWindrowDropDisableText,
    self.customEnvironment)
128 end
129
130 spec.toggleWindrowDropAnimation = getXMLString(self.xmlFile,
    "vehicle.mower.toggleWindrowDrop#animationName")
```

```
131 spec.enableWindrowDropAnimationSpeed =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.mower.toggleWindrowDrop#animationEnableSpeed"), 1)
132 spec.disableWindrowDropAnimationSpeed =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.mower.toggleWindrowDrop#animationDisableSpeed"), -
    spec.enableWindrowDropAnimationSpeed)
133
134 spec.useWindrowDropAreas = Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.mower.toggleWindrowDrop#startEnabled"), false)
135
136 spec.workAreaParameters = {}
137 spec.workAreaParameters.lastChangedArea = 0
138 spec.workAreaParameters.lastTotalArea = 0
139
140 spec.dirtyFlag = self:getNextDirtyFlag()
141
142 if self.addAITerrainDetailRequiredRange ~= nil then
143 self:addAITerrainDetailRequiredRange(g_currentMission.grassValue,
    g_currentMission.grassValue,
    g_currentMission.terrainDetailTypeFirstChannel,
    g_currentMission.terrainDetailTypeNumChannels)
144 self:addAITerrainDetailRequiredRange(g_currentMission.sowingValue,
    g_currentMission.sowingValue,
    g_currentMission.terrainDetailTypeFirstChannel,
    g_currentMission.terrainDetailTypeNumChannels)
145 end
146
147 if self.addAIFruitRequirement ~= nil then
148 for inputFruitType, _ in pairs(spec.fruitTypeConverters) do
149 local desc =
    g_fruitTypeManager:getFruitTypeByIndex(inputFruitType)
150 self:addAIFruitRequirement(desc.index,
    desc.minHarvestingGrowthState, desc.maxHarvestingGrowthState)
151 end
152 end
153 end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
157 function Mower:onDelete()
```

```
158 if self.isClient then
159 local spec = self.spec_mower
160
161 for _, dropEffect in pairs(spec.dropEffects) do
162 g_effectManager:deleteEffects(dropEffect.effects)
163 end
164 g_animationManager:deleteAnimations(spec.animationNodes)
165 end
166 end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
172 function Mower:onReadStream(streamId, connection)
173 local spec = self.spec_mower
174 if spec.toggleWindrowDropEnableText ~= nil and
    spec.toggleWindrowDropDisableText ~= nil then
175 local useMowerWindrowDropAreas = streamReadBool(streamId)
176 self:setUseMowerWindrowDropAreas(useMowerWindrowDropAreas, true)
177 end
178 end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
184 function Mower:onWriteStream(streamId, connection)
185 local spec = self.spec_mower
186 if spec.toggleWindrowDropEnableText ~= nil and
    spec.toggleWindrowDropDisableText ~= nil then
187 streamWriteBool(streamId, spec.useWindrowDropAreas)
188 end
189 end
```

onReadUpdateStream

Description

Called on on update

Definition

onReadUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId	stream ID
integer timestamp	timestamp
table	connection connection

Code

```
196 function Mower:onReadUpdateStream(streamId, timestamp, connection)
197 if connection:getIsServer() then
198 local spec = self.spec_mower
199
200 if #spec.dropEffects > 0 then
201 if streamReadBool(streamId) then
202 for _, dropEffect in pairs(spec.dropEffects) do
203 self:setDropEffectEnabled(dropEffect, streamReadBool(streamId))
204 end
205 end
206 end
207 end
208 end
```

onWriteUpdateStream

Description

Called on on update

Definition

onWriteUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Code

```
215 function Mower:onWriteUpdateStream(streamId, connection, dirtyMask)
216 if not connection:getIsServer() then
217 local spec = self.spec_mower
218
219 if #spec.dropEffects > 0 then
220 if streamWriteBool(streamId, bitAND(dirtyMask, spec.dirtyFlag) ~= 0) then
```

```
221 for _, dropEffect in pairs(spec.dropEffects) do
222 streamWriteBool(streamId, dropEffect.isActiveSent)
223 end
224 end
225 end
226 end
227 end
```

setDropEffectEnabled

Description

Enable mower effect

Definition

setDropEffectEnabled(table mowerEffect, boolean isActive)

Arguments

table	mowerEffect	mower effect
boolean	isActive	new is active state

Code

```
330 function Mower:setDropEffectEnabled(dropEffect, isActive)
331 dropEffect.isActive = isActive
332 if self.isClient then
333 if isActive then
334 g_effectManager:setFillType(dropEffect.effects,
    dropEffect.fillType)
335 g_effectManager:startEffects(dropEffect.effects)
336 else
337 g_effectManager:stopEffects(dropEffect.effects)
338 end
339 end
340 end
```

setUseMowerWindrowDropAreas

Description

Toggle use of windrower drop areas

Definition

setUseMowerWindrowDropAreas(boolean useMowerWindrowDropAreas, boolean noEventSend)

Arguments

boolean	useMowerWindrowDropAreas	use mower windrow drop areas
boolean	noEventSend	no event send

Code

```
346 function
    Mower:setUseMowerWindrowDropAreas(useMowerWindrowDropAreas,
    noEventSend)
347 local spec = self.spec_mower
```

```
348 if useMowerWindrowDropAreas ~= spec.useWindrowDropAreas then
349 MowerToggleWindrowDropEvent.sendEvent(self,
    useMowerWindrowDropAreas, noEventSend)
350 spec.useWindrowDropAreas = useMowerWindrowDropAreas
351
352 if spec.toggleWindrowDropAnimation ~= nil and self.playAnimation
    ~= nil then
353 local speed = spec.enableWindrowDropAnimationSpeed
354 if not useMowerWindrowDropAreas then
355 speed = spec.disableWindrowDropAnimationSpeed
356 end
357 self:playAnimation(spec.toggleWindrowDropAnimation, speed, nil,
    true)
358 end
359 end
360 end
```

loadWorkAreaFromXML

Description

Loads work areas from xml

Definition

loadWorkAreaFromXML(table workArea, integer xmlFile, string key)

Arguments

table	workArea	workArea
integer	xmlFile	id of xml object
string	key	key

Return Values

boolean	success	success

Code

```
368 function Mower:loadWorkAreaFromXML(superFunc, workArea, xmlFile,
    key)
369 local retValue = superFunc(self, workArea, xmlFile, key)
370
371 if workArea.type == WorkAreaType.DEFAULT then
372 workArea.type = WorkAreaType.MOWER
373 end
374
375 if workArea.type == WorkAreaType.MOWER then
376 workArea.dropWindrow = Utils.getNoNil(getXMLBool(xmlFile, key ..
    ".mower#dropWindrow"), true)
377 workArea.dropAreaIndex = Utils.getNoNil(getXMLInt(xmlFile, key ..
    ".mower#dropAreaIndex"), 1)
378
379 workArea.lastPickupLiters = 0
```

```
380 end
381
382 if workArea.type == WorkAreaType.AUXILIARY then
383 workArea.litersToDrop = 0
384 if self.spec_mower.dropAreas == nil then
385 self.spec_mower.dropAreas = {}
386 end
387 table.insert(self.spec_mower.dropAreas, workArea)
388 end
389
390 return retValue
391 end
```

doCheckSpeedLimit

Description

Returns if speed limit should be checked

Definition

doCheckSpeedLimit()

Return Values

boolean checkSpeedlimit check speed limit

Code

```
411 function Mower:doCheckSpeedLimit(superFunc)
412 return superFunc(self) or (self:getIsTurnedOn() and
    (self.getIsLowered == nil or self:getIsLowered()))
413 end
```

onTurnedOn

Description

Called on turn off

Definition

onTurnedOn(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
422 function Mower:onTurnedOn()
423 if self.isClient then
424 local spec = self.spec_mower
425 g_animationManager:startAnimations(spec.animationNodes)
426 end
427 end
```

onTurnedOff

Description

Called on turn off

Definition

onTurnedOff(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
function Mower:onTurnedOff()
if self.isClient then
local spec = self.spec_mower
for _, dropEffect in pairs(spec.dropEffects) do
self:setDropEffectEnabled(dropEffect, false)
end

g_animationManager:stopAnimations(spec.animationNodes)
end
end
```

getDefaultSpeedLimit

Description

Returns default speed limit

Definition

getDefaultSpeedLimit()

Return Values

float speedLimit speed limit

Code

```
function Mower.getDefaultSpeedLimit()
return 20
end
```

Pipe

Description

onDischargeStateChanged

Description

Called on discharge state change

Definition

onDischargeStateChanged()

Code

```
function Pipe:onDischargeStateChanged(state)
if self.isClient then
local spec = self.spec_pipe
local dischargeNode = self:getCurrentDischargeNode()
local dischargeNodeIndex = nil
if dischargeNode ~= nil then
dischargeNodeIndex = dischargeNode.index
end
if dischargeNodeIndex == spec.dischargeNodeIndex then
```

```
901  if state == Dischargeable.DISCHARGE_STATE_OFF then
902  g_animationManager:stopAnimations(spec.animationNodes)
903  else
904  g_animationManager:startAnimations(spec.animationNodes)
905  g_animationManager:setFillType(spec.animationNodes,
     self:getFillUnitLastValidFillType(dischargeNode.fillUnitIndex))
906  end
907  end
908  end
909  end
```

Plow

Description

This is the specialization for plows

getWearMultiplier

Description

Returns current wear multiplier

Definition

getWearMultiplier()

Return Values

float dirtMultiplier current wear multiplier

Code

```
371  function Plow:getWearMultiplier(superFunc)
372  local multiplier = superFunc(self)
373
374  local spec = self.spec_plow
375  if spec.isWorking then
376  multiplier = multiplier + self:getWorkWearMultiplier() *
     self:getLastSpeed() / self.speedLimit
377  end
378
379  return multiplier
380  end
```

onPostAttach

Description

Called if vehicle gets attached

Definition

onPostAttach(table attacherVehicle, integer inputJointDescIndex, integer jointDescIndex)

Arguments

table	attacherVehicle	attacher vehicle
integer	inputJointDescIndex	index of input attacher joint
integer	jointDescIndex	index of attacher joint it gets attached to

Code

```lua
function Plow:onPostAttach(attacherVehicle, inputJointDescIndex, jointDescIndex)
local spec = self.spec_plow

spec.startActivationTime = g_currentMission.time + spec.startActivationTimeout
if spec.wasTurnAnimationStopped then
local dir = 1
if not spec.rotationMax then
dir = -1
end
self:playAnimation(spec.rotationPart.turnAnimation, dir, self:getAnimationTime(spec.rotationPart.turnAnimation), true)
spec.wasTurnAnimationStopped = false
end
end
```

onPreDetach

Description

Called if vehicle gets detached

Definition

onPreDetach(table attacherVehicle, table implement)

Arguments

table attacherVehicle attacher vehicle
table implement implement

Code

```lua
function Plow:onPreDetach(attacherVehicle, implement)
local spec = self.spec_plow

spec.limitToField = true
if self:getIsAnimationPlaying(spec.rotationPart.turnAnimation) then
self:stopAnimation(spec.rotationPart.turnAnimation, true)
spec.wasTurnAnimationStopped = true
end
end
```

PowerConsumer

Description

This is the specialization for all vehicles that consume power

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
22  function PowerConsumer.prerequisitesPresent(specializations)
23  return true
24  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
48  function PowerConsumer:onLoad(savegame)
49  local spec = self.spec_powerConsumer
50
51  spec.forceNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, "vehicle.powerConsumer#forceNode"),
    self.i3dMappings)
52  spec.forceDirNode =
    Utils.getNoNil(I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, "vehicle.powerConsumer#forceDirNode"),
    self.i3dMappings), spec.forceNode)
53  spec.forceFactor = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.powerConsumer#forceFactor"), 1.0)
54
55  spec.maxForce = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.powerConsumer#maxForce"), 0) -- kN
56  spec.forceDir = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.powerConsumer#forceDir"), 1)
57
58  spec.turnOnNotAllowedWarning =
    string.format(g_i18n:getText(Utils.getNoNil(getXMLString(self.xmlFile,
    "vehicle.powerConsumer#turnOnNotAllowedWarning"),
    "warning_insufficientPowerOutput")), self.typeDesc)
59
60  self:loadPowerSetup(self.xmlFile)
61  end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
68  function PowerConsumer:onUpdate(dt, isActiveForInput, isSelected)
69  if self.isServer then
70  local spec = self.spec_powerConsumer
71
72  if spec.forceNode ~= nil and self.movingDirection == spec.forceDir
    then
73  local multiplier = self:getPowerMultiplier()
74  if multiplier ~= 0 then
75  local frictionForce = spec.forceFactor * self.lastSpeedReal * 1000
    * self:getTotalMass(false) / (dt/1000)
76
77  local force = -math.min(frictionForce, spec.maxForce) *
    self.movingDirection * multiplier
78  local dx,dy,dz = localDirectionToWorld(spec.forceDirNode, 0, 0,
    force)
79  local px,py,pz = getCenterOfMass(spec.forceNode)
80
81  addForce(spec.forceNode, dx,dy,dz, px,py,pz, true)
82
83  if VehicleDebug.state == VehicleDebug.DEBUG_PHYSICS and
    self:getIsActiveForInput() then
84  local str = string.format("frictionForce=%.2f maxForce=%.2f ->
    force=%.2f", frictionForce, spec.maxForce, force)
85  renderText(0.7, 0.85, getCorrectTextSize(0.02), str)
86  end
87  end
88  end
89  end
90  end
```

loadPowerSetup

Description

Called on loading

Definition

loadPowerSetup(table savegame)

Arguments

table savegame savegame

Code

```
95  function PowerConsumer:loadPowerSetup(xmlFile)
96  local spec = self.spec_powerConsumer
97
98  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
    "vehicle.powerConsumer#neededPtoPower",
    "vehicle.powerConsumer#neededMinPtoPower and
    vehicle.powerConsumer#neededMaxPtoPower")
99
100 spec.neededMaxPtoPower = Utils.getNoNil(getXMLFloat(xmlFile,
    "vehicle.powerConsumer#neededMaxPtoPower"), 0)
101 spec.neededMinPtoPower = Utils.getNoNil(getXMLFloat(xmlFile,
    "vehicle.powerConsumer#neededMinPtoPower"),
    spec.neededMaxPtoPower) -- in kW at ptoRpm
102 if spec.neededMaxPtoPower < spec.neededMinPtoPower then
103 g_logManager:xmlWarning(self.configFileName,
    "'vehicle.powerConsumer#neededMaxPtoPower' is smaller than
    'vehicle.powerConsumer#neededMinPtoPower'")
104 end
105
106 spec.ptoRpm = Utils.getNoNil(getXMLFloat(xmlFile,
    "vehicle.powerConsumer#ptoRpm"), 0)
107 end
```

getPtoRpm

Description

Returns rpm of pto

Definition

getPtoRpm()

Return Values

float rpm rpm of pto

Code

```
112 function PowerConsumer:getPtoRpm()
113 if self:getDoConsumePtoPower() then
114 return self.spec_powerConsumer.ptoRpm
115 end
116
117 return 0
118 end
```

getDoConsumePtoPower

Description

Returns if should consume pto power

Definition

getDoConsumePtoPower()

Return Values

boolean consume consumePtoPower

Code

```
123  function PowerConsumer:getDoConsumePtoPower()
124  return self.getIsTurnedOn ~= nil and self:getIsTurnedOn()
125  end
```

getPowerMultiplier

Description

Returns power multiplier

Definition

getPowerMultiplier()

Return Values

float powerMultiplier current power multiplier

Code

```
130  function PowerConsumer:getPowerMultiplier()
131  return 1
132  end
```

getConsumedPtoTorque

Description

Returns consumed pto torque

Definition

getConsumedPtoTorque()

Return Values

float torque consumed pto torque in kNm

Code

```
138  function PowerConsumer:getConsumedPtoTorque(expected)
139  if self:getDoConsumePtoPower() or (expected ~= nil and expected)
     then
140  local spec = self.spec_powerConsumer
141
142  local rpm = spec.ptoRpm
143  if rpm > 0.001 then
144  local consumingLoad, count = self:getConsumingLoad()
145  if count > 0 then
146  consumingLoad = consumingLoad / count
147  else
148  consumingLoad = 1
149  end
150
151  local neededPtoPower = spec.neededMinPtoPower + (consumingLoad *
     (spec.neededMaxPtoPower - spec.neededMinPtoPower))
152  return neededPtoPower / (rpm*math.pi/30)
153  end
```

```
154 end
155
156 return 0
157 end
```

getCanBeTurnedOn

Description

Returns if turn on is allowed

Definition

getCanBeTurnedOn()

Return Values

boolean allow allow turn on

Code

```
166 function PowerConsumer:getCanBeTurnedOn(superFunc)
167 local rootVehicle = self:getRootVehicle()
168 if rootVehicle ~= nil and rootVehicle.getMotor ~= nil then
169 local rootMotor = rootVehicle:getMotor()
170
171 local torqueRequested = self:getConsumedPtoTorque(true)
172 torqueRequested = torqueRequested +
    PowerConsumer.getTotalConsumedPtoTorque(rootVehicle, self)
173 torqueRequested = torqueRequested /
    rootMotor:getPtoMotorRpmRatio()
174
175 -- 90% of motor torque should be more than requested torque,
    because we need some torque to accelerate the vehicle
176 if torqueRequested > 0 and torqueRequested >
    0.9*rootMotor:getPeakTorque() then
177 return false, true
178 end
179 end
180
181 if superFunc ~= nil then
182 return superFunc(self)
183 else
184 return true, false
185 end
186 end
```

getTurnedOnNotAllowedWarning

Description

Returns turn on not allowed warning text

Definition

getTurnedOnNotAllowedWarning()

Return Values

string warningText turn on not allowed warning text

Code

```
191 function PowerConsumer:getTurnedOnNotAllowedWarning(superFunc)
192 local spec = self.spec_powerConsumer
193 local _, notEnoughPower = PowerConsumer.getCanBeTurnedOn(self)
194 if notEnoughPower then
195 return spec.turnOnNotAllowedWarning
196 else
197 return superFunc(self)
198 end
199 end
```

getTotalConsumedPtoTorque

Description

Get total amount of consumed pto torque (from every attached vehicle)

Definition

getTotalConsumedPtoTorque(table excludeVehicle)

Arguments

table excludeVehicle exluded vehicle

Return Values

float total amount of consumed pto torque in kNm

Code

```
205 function PowerConsumer.getTotalConsumedPtoTorque(self,
    excludeVehicle)
206 local torque = 0
207 if self ~= excludeVehicle then
208 if self.getConsumedPtoTorque ~= nil then
209 torque = self:getConsumedPtoTorque()
210 end
211 end
212
213 if self.getAttachedImplements ~= nil then
214 local attachedImplements = self:getAttachedImplements()
215 for _, implement in pairs(attachedImplements) do
216 torque = torque +
    PowerConsumer.getTotalConsumedPtoTorque(implement.object,
    excludeVehicle)
217 end
218 end
219
220 return torque
221 end
```

getMaxPtoRpm

Description

Returns max pto rpm

Definition

getMaxPtoRpm()

Return Values

float maxPtoRpm max pto rpm

Code

```
226  function PowerConsumer.getMaxPtoRpm(self)
227  local rpm = 0
228
229  if self.getPtoRpm ~= nil then
230  rpm = self:getPtoRpm()
231  end
232
233  if self.getAttachedImplements ~= nil then
234  local attachedImplements = self:getAttachedImplements()
235  for _, implement in pairs(attachedImplements) do
236  rpm = math.max(rpm, PowerConsumer.getMaxPtoRpm(implement.object))
237  end
238  end
239
240  return rpm
241  end
```

loadSpecValueNeededPower

Description

Loads needed power spec value

Definition

loadSpecValueNeededPower(integer xmlFile, string customEnvironment)

Arguments

integer	xmlFile	id of xml object
string	customEnvironment	custom environment

Return Values

float neededPower needed power

Code

```
283  function PowerConsumer.loadSpecValueNeededPower(xmlFile,
     customEnvironment)
284  return getXMLString(xmlFile,
     "vehicle.storeData.specs.neededPower")
285  end
```

getSpecValueNeededPower

Description

Returns needed power spec value

Definition

getSpecValueNeededPower(table storeItem, table realItem)

Arguments

table storeItem store item
table realItem real item

Return Values

string l10n l10n text
float neededPower needed power in kw
float neededPower needed power in hp

Code

```
294 function PowerConsumer.getSpecValueNeededPower(storeItem, realItem)
295 if storeItem.specs.neededPower ~= nil then
296 local hp, kw = g_i18n:getPower(storeItem.specs.neededPower)
297 return string.format(g_i18n:getText("shop_neededPowerValue"), MathUtil.round(kw), MathUtil.round(hp))
298 end
299 return nil
300 end
```

RandomlyMovingParts

Description

Class for randomly moving parts

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17 function RandomlyMovingParts.prerequisitesPresent(specializations)
18 return true
19 end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
36  function RandomlyMovingParts:onLoad(savegame)
37  local spec = self.spec_randomlyMovingParts
38
39  spec.nodes = {}
40  local i = 0
41  while true do
42  local baseName =
    string.format("vehicle.randomlyMovingParts.randomlyMovingPart(%d)",
    i)
43  if not hasXMLProperty(self.xmlFile, baseName) then
44  break
45  end
46
47  local randomlyMovingPart = {}
48  if self:loadRandomlyMovingPartFromXML(randomlyMovingPart,
    self.xmlFile, baseName) then
49  table.insert(spec.nodes, randomlyMovingPart)
50  end
51
52  i = i + 1
53  end
54  end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
61  function RandomlyMovingParts:onUpdate(dt, isActiveForInput,
    isSelected)
62  local spec = self.spec_randomlyMovingParts
63  for _, part in pairs(spec.nodes) do
64  self:updateRandomlyMovingPart(part, dt)
65  end
66  end
```

loadRandomlyMovingPartFromXML

Description

Load randomly moving part from xml

Definition

loadRandomlyMovingPartFromXML(table part, integer xmlFile, string key)

Arguments

table	part	part
integer	xmlFile	id of xml object
string	key	key

Return Values

boolean

Code

```
74  function RandomlyMovingParts:loadRandomlyMovingPartFromXML(part, xmlFile, key)
75  if not hasXMLProperty(self.xmlFile, key) then
76  return false
77  end
78
79  XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName, key .. "#index", key .. "#node") --FS17 to FS19
80
81  local node = I3DUtil.indexToObject(self.components, getXMLString(xmlFile, key .. "#node"), self.i3dMappings)
82  if node == nil then
83  g_logManager:xmlWarning(self.configFileName, "Unknown node for randomlyMovingPart in '%s'", key)
84  return false
85  end
86
87  part.node = node
88
89  if self.getGroundReferenceNodeFromIndex ~= nil then
90  local refNodeIndex = getXMLInt(xmlFile, key .. "#refNodeIndex")
91  if refNodeIndex ~= nil then
92  if refNodeIndex ~= 0 then
93  local groundReferenceNode = self:getGroundReferenceNodeFromIndex(refNodeIndex)
94  if groundReferenceNode ~= nil then
95  part.groundReferenceNode = groundReferenceNode
96  end
97  else
98  g_logManager:xmlWarning(self.configFileName, "Unknown ground reference node in '%s'! Indices start with '0'", key.."#refNodeIndex")
99  end
```

```
100 end
101 end
102
103 local rx, ry, rz = getRotation(part.node)
104 local rotMean =
    StringUtil.getRadiansFromString(getXMLString(xmlFile, key ..
    "#rotMean"), 2)
105 if rotMean then
106 part.rotOrig = {rx, ry, rz}
107 part.rotCur = {rx, ry, rz}
108
109 part.rotAxis = getXMLInt(xmlFile, key .. "#rotAxis")
110
111 part.rotMean = rotMean
112 part.rotVar =
    StringUtil.getRadiansFromString(getXMLString(xmlFile, key ..
    "#rotVariance"), 2)
113
114 part.rotTimeMean =
    StringUtil.getVectorNFromString(getXMLString(xmlFile, key ..
    "#rotTimeMean"), 2)
115 part.rotTimeVar =
    StringUtil.getVectorNFromString(getXMLString(xmlFile, key ..
    "#rotTimeVariance"), 2)
116
117 part.pauseMean =
    StringUtil.getVectorNFromString(getXMLString(xmlFile, key ..
    "#pauseMean"), 2)
118 part.pauseVar =
    StringUtil.getVectorNFromString(getXMLString(xmlFile, key ..
    "#pauseVariance"), 2)
119
120 for i=1,2 do
121 part.rotTimeMean[i] = part.rotTimeMean[i] * 1000
122 part.rotTimeVar[i] = part.rotTimeVar[i] * 1000
123 part.pauseMean[i] = part.pauseMean[i] * 1000
124 part.pauseVar[i] = part.pauseVar[i] * 1000
125 end
126
127 part.rotTarget = {}
128 part.rotSpeed = {}
129 part.pause = {}
130
```

```
131 part.isSpeedDependent = Utils.getNoNil(getXMLBool(xmlFile, key .. "#isSpeedDependent"), false)
132
133 self:updateRotationTargetValues(part)
134 end
135
136 part.nextMoveTime = g_currentMission.time + part.pause[2]
137 part.curMoveDirection = 1
138 part.isActive = true
139
140 return true
141 end
```

updateRandomlyMovingPart

Description

Update randomly moving parts

Definition

updateRandomlyMovingPart(table part, float dt)

Arguments

table part part to update

float dt time since last call in ms

Return Values

boolean updated part was updated

Code

```
148 function RandomlyMovingParts:updateRandomlyMovingPart(part, dt)
149 if part.nextMoveTime < g_currentMission.time then
150 local speed = dt
151 if part.isSpeedDependent then
152 speed = speed * math.min(self:getLastSpeed() / self:getSpeedLimit(true), 1)
153 end
154
155 part.isActive = self:getIsRandomlyMovingPartActive(part)
156 if part.curMoveDirection > 0 then
157 if part.isActive then
158 part.rotCur[part.rotAxis] = math.min(part.rotTarget[1], part.rotCur[part.rotAxis] + (part.rotSpeed[1] * speed))
159 if part.rotCur[part.rotAxis] == part.rotTarget[1] then
160 part.curMoveDirection = -1
161 part.nextMoveTime = g_currentMission.time + part.pause[1]
162 end
163 end
164 else
```

```
165  part.rotCur[part.rotAxis] = math.max(part.rotTarget[2],
     part.rotCur[part.rotAxis] + (part.rotSpeed[2] * speed))
166  if part.rotCur[part.rotAxis] == part.rotTarget[2] then
167  -- start next movement only if active
168  if part.isActive then
169  part.curMoveDirection = 1
170  part.nextMoveTime = g_currentMission.time + part.pause[2]
171  self:updateRotationTargetValues(part)
172  end
173  end
174  end
175
176  setRotation(part.node, part.rotCur[1], part.rotCur[2],
     part.rotCur[3])
177
178  if self.setMovingToolDirty ~= nil then
179  self:setMovingToolDirty(part.node)
180  end
181
182  return true
183  else
184  return false
185  end
186  end
```

updateRotationTargetValues

Description

Update rotation target values

Definition

updateRotationTargetValues(table part)

Arguments

table part part to update

Code

```
191  function RandomlyMovingParts:updateRotationTargetValues(part)
192  for i=1,2 do
193  part.rotTarget[i] = part.rotMean[i] + (part.rotVar[i] * (-0.5 +
     math.random()))
194  end
195
196  for i=1,2 do
197  local rotTime = part.rotTimeMean[i] + (part.rotTimeVar[i] * (-0.5
     + math.random()))
198  if i == 1 then
```

```
199 part.rotSpeed[i] = (part.rotTarget[1] - part.rotTarget[2]) / rotTime
200 else
201 part.rotSpeed[i] = (part.rotTarget[2] - part.rotTarget[1]) / rotTime
202 end
203 end
204
205 for i=1,2 do
206 part.pause[i] = part.pauseMean[i] + (part.pauseVar[i] * (-0.5 + math.random()))
207 end
208 end
```

getIsRandomlyMovingPartActive

Description

Returns if randomly moving part is active

Definition

getIsRandomlyMovingPartActive(table part)

Arguments

table part part to check

Return Values

boolean isActive is active

Code

```
214 function RandomlyMovingParts:getIsRandomlyMovingPartActive(part)
215 local retValue = true
216 if part.groundReferenceNode ~= nil then
217 retValue = self:getIsGroundReferenceNodeActive(part.groundReferenceNode)
218 end
219
220 return retValue
221 end
```

ReceivingHopper

Description

Class for all receiving hoppers

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
21  function ReceivingHopper.prerequisitesPresent(specializations)
22  return SpecializationUtil.hasSpecialization(FillUnit,
    specializations) and
    SpecializationUtil.hasSpecialization(Dischargeable,
    specializations)
23  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
47  function ReceivingHopper:onLoad(savegame)
48  local spec = self.spec_receivingHopper
49
50  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.receivingHopper#unloadingDelay",
    "Dischargeable functionalities") -- FS17 to FS19
51  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.receivingHopper#unloadInfoIndex",
    "Dischargeable functionalities") -- FS17 to FS19
52  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.receivingHopper#dischargeInfoIndex",
    "Dischargeable functionalities") -- FS17 to FS19
53  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.receivingHopper.tipTrigger#index",
    "Dischargeable functionalities") -- FS17 to FS19
54  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.receivingHopper.boxTrigger#index",
    "Dischargeable functionalities") -- FS17 to FS19
55  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.receivingHopper.fillScrollerNodes.fillScrollerNode",
    "Dischargeable functionalities") -- FS17 to FS19
56  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.receivingHopper.fillEffect",
    "Dischargeable functionalities") -- FS17 to FS19
57  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.receivingHopper.fillEffect",
    "Dischargeable functionalities") -- FS17 to FS19
58  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
```

```
    "vehicle.receivingHopper.boxTrigger#litersPerMinute",
    "Dischargeable functionalities") -- FS17 to FS19
59  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.receivingHopper.raycastNode#index",
    "Dischargeable functionalities") -- FS17 to FS19
60  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.receivingHopper.raycastNode#raycastLength",
    "Dischargeable functionalities") -- FS17 to FS19
61
62  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.receivingHopper.boxTrigger#boxSpawnPlaceIndex",
    "vehicle.receivingHopper.boxes#spawnPlaceNode") -- FS17 to FS19
63  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.receivingHopper.boxTrigger.box(0)",
    "vehicle.receivingHopper.boxes.box(0)") -- FS17 to FS19
64
65  spec.fillUnitIndex = Utils.getNoNil(getXMLInt(self.xmlFile,
    "vehicle.receivingHopper#fillUnitIndex"), 1)
66  spec.spawnPlace = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile,
    "vehicle.receivingHopper.boxes#spawnPlaceNode"), self.i3dMappings)
67
68  spec.boxes = {}
69  local i = 0
70  while true do
71  local baseName =
    string.format("vehicle.receivingHopper.boxes.box(%d)", i)
72  if not hasXMLProperty(self.xmlFile, baseName) then
73  break
74  end
75
76  local fillTypeStr = getXMLString(self.xmlFile,
    baseName.."#fillType")
77  local filename = getXMLString(self.xmlFile, baseName.."#filename")
78  local fillTypeIndex =
    g_fillTypeManager:getFillTypeIndexByName(fillTypeStr)
79  if fillTypeIndex ~= nil then
80  spec.boxes[fillTypeIndex] = filename
81  else
82  g_logManager:xmlWarning(self.configFileName, "Invalid fillType
    '%s'", fillTypeStr)
83  end
84
```

```
85  i = i + 1
86  end
87
88  spec.createBoxes = false
89  spec.lastBox = nil
90  end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
97  function ReceivingHopper:onUpdateTick(dt, isActiveForInput,
    isSelected)
98  local spec = self.spec_receivingHopper
99  if spec.createBoxes then
100 if self:getDischargeState() == Dischargeable.DISCHARGE_STATE_OFF
    then
101 if self:getCanSpawnNextBox() then
102 self:createBox()
103 end
104 end
105 end
106 end
```

setCreateBoxes

Description

Toggle box creating

Definition

setCreateBoxes(boolean state, boolean noEventSend)

Arguments

boolean	state	new state
boolean	noEventSend	no event send

Code

```
112 function ReceivingHopper:setCreateBoxes(state, noEventSend)
113 local spec = self.spec_receivingHopper
114 if state ~= spec.createBoxes then
115 ReceivingHopperSetCreateBoxesEvent.sendEvent(self, state,
    noEventSend)
```

```
116
117 spec.createBoxes = state
118 ReceivingHopper.updateActionEvents(self)
119
120 spec.lastBox = nil
121 end
122 end
```

createBox

Description

Create box

Definition

createBox()

Code

```
161 function ReceivingHopper:createBox()
162 local spec = self.spec_receivingHopper
163 if self.isServer then
164 if spec.createBoxes then
165 local fillType = self:getFillUnitFillType(spec.fillUnitIndex)
166 if spec.boxes[fillType] ~= nil then
167 local x, _, z = getWorldTranslation(spec.spawnPlace)
168 local dirX, _, dirZ = localDirectionToWorld(spec.spawnPlace, 0,
    1, 0)
169 local yRot = MathUtil.getYRotationFromDirection(dirX, dirZ);
170 local xmlFilename = Utils.getFilename(spec.boxes[fillType],
    self.baseDirectory)
171 local vehicle = g_currentMission:loadVehicle(xmlFilename, x, nil,
    z, 0, yRot, true, 0, Vehicle.PROPERTY_STATE_OWNED,
    self:getOwnerFarmId(), nil, nil)
172 if vehicle ~= nil then
173 table.insert(g_currentMission.vehicles, vehicle)
174 spec.lastBox = vehicle
175 end
176 end
177 end
178 end
179 end
```

ReverseDriving

Description

This is the specialization for vehicles with reverse driving functionality

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
19  function ReverseDriving.prerequisitesPresent(specializations)
20  return SpecializationUtil.hasSpecialization(Drivable,
    specializations)
21  and SpecializationUtil.hasSpecialization(Enterable,
    specializations)
22  and SpecializationUtil.hasSpecialization(AnimatedVehicle,
    specializations)
23  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
58  function ReverseDriving:onLoad(savegame)
59  local spec = self.spec_reverseDriving
60
61  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.reverseDriving.steering#reversedIndex",
    "vehicle.reverseDriving.steeringWheel#node") --FS17 to FS19
62  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.reverseDriving.steering#reversedNode",
    "vehicle.reverseDriving.steeringWheel#node") --FS17 to FS19
63
64  spec.reversedCharacterTargets = {}
65  IKUtil.loadIKChainTargets(self.xmlFile, "vehicle.reverseDriving",
    self.components, spec.reversedCharacterTargets, self.i3dMappings)
66
67  local node = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile,
    "vehicle.reverseDriving.steeringWheel#node"), self.i3dMappings)
68  if node ~= nil then
69  spec.steeringWheel = {}
```

```
70  spec.steeringWheel.node = node
71  local _,ry,_ = getRotation(spec.steeringWheel.node)
72  spec.steeringWheel.lastRotation = ry
73  spec.steeringWheel.indoorRotation =
    math.rad(Utils.getNoNil(Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.reverseDriving.steeringWheel#indoorRotation"),
    getXMLFloat(self.xmlFile,
    "vehicle.drivable.steeringWheel#indoorRotation")), 0))
74  spec.steeringWheel.outdoorRotation =
    math.rad(Utils.getNoNil(Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.reverseDriving.steeringWheel#outdoorRotation"),
    getXMLFloat(self.xmlFile,
    "vehicle.drivable.steeringWheel#outdoorRotation")), 0))
75  end
76
77  spec.reverseDrivingAnimation =
    Utils.getNoNil(getXMLString(self.xmlFile,
    "vehicle.reverseDriving#animationName"), "reverseDriving")
78
79  if not self:getAnimationExists(spec.reverseDrivingAnimation) then
80  g_logManager:xmlError(self.configFileName, "ReverseDriving
    requires a animation in 'vehicle.reverseDriving#animationName'!")
81  end
82
83  spec.isChangingDirection = false
84  spec.isReverseDriving = false
85  spec.isSelectable = true
86
87  spec.smoothReverserDirection = 1
88  end
```

onPostLoad

Description

Called after loading

Definition

onPostLoad(table savegame)

Arguments

table savegame savegame

Code

```
93  function ReverseDriving:onPostLoad(savegame)
94  local spec = self.spec_reverseDriving
95
96  local character = self:getVehicleCharacter()
97  if character ~= nil then
```

```
 98  spec.defaultCharacterTargets = character:getIKChainTargets()
 99  end
100
101  local isReverseDriving = false
102  if savegame ~= nil then
103  isReverseDriving = Utils.getNoNil(getXMLBool(savegame.xmlFile,
     savegame.key..".reverseDriving#isActive"), false)
104  end
105
106  self:setIsReverseDriving(isReverseDriving, true)
107  spec.updateAnimationOnEnter = isReverseDriving
108  end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
119  function ReverseDriving:onReadStream(streamId, connection)
120  self:setIsReverseDriving(Utils.getNoNil(streamReadBool(streamId),
     false))
121  end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
127  function ReverseDriving:onWriteStream(streamId, connection)
128  streamWriteBool(streamId,
     self.spec_reverseDriving.isReverseDriving)
129  end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
136  function ReverseDriving:onUpdate(dt, isActiveForInput,
     isSelected)
137  local spec = self.spec_reverseDriving
138
139  if spec.isChangingDirection then
140  local character = self:getVehicleCharacter()
141  if character ~= nil then
142  character:setCharacterVisibility(false)
143  end
144
145  if not self:getIsEntered() then
146  if spec.updateAnimationOnEnter then
147  AnimatedVehicle.updateAnimations(self, 99999999)
148  spec.updateAnimationOnEnter = false
149  end
150  end
151
152  if not self:getIsAnimationPlaying(spec.reverseDrivingAnimation)
     then
153  self:reverseDirectionChanged(spec.reverserDirection)
154  end
155
156  local direction = (spec.isReverseDriving and 1) or -1
157  spec.smoothReverserDirection =
     MathUtil.clamp(spec.smoothReverserDirection - 0.001 * dt *
     direction, -1, 1)
158  end
159  end
```

reverseDirectionChanged

Description

Called after reverse drive change

Definition

reverseDirectionChanged(float direction)

Arguments

float	direction new direction

Code

```lua
function ReverseDriving:reverseDirectionChanged(direction)
local spec = self.spec_reverseDriving

spec.isChangingDirection = false
if spec.isReverseDriving then
self:setReverserDirection(-1)
spec.smoothReverserDirection = -1
else
self:setReverserDirection(1)
spec.smoothReverserDirection = 1
end

local character = self:getVehicleCharacter()
if character ~= nil then
if spec.isReverseDriving and next(spec.reversedCharacterTargets) ~= nil then
character:setIKChainTargets(spec.reversedCharacterTargets)
else
character:setIKChainTargets(spec.defaultCharacterTargets)
end

if character.meshThirdPerson ~= nil and not self:getIsEntered() then
character:updateVisibility()
end

character:setAllowCharacterUpdate(true)
end

if self.setLightsTypesMask ~= nil then
self:setLightsTypesMask(self.spec_lights.lightsTypesMask, true, true)
end

SpecializationUtil.raiseEvent(self, "onReverseDirectionChanged", direction)
end
```

setIsReverseDriving

Description

Toggle reverse driving

Definition

setIsReverseDriving(boolean isReverseDriving, boolean noEventSend)

Arguments

boolean isReverseDriving

boolean noEventSend no event send

Code

```
202 function ReverseDriving:setIsReverseDriving(isReverseDriving, noEventSend)
203 local spec = self.spec_reverseDriving
204 if isReverseDriving ~= spec.isReverseDriving then
205 spec.isChangingDirection = true
206 spec.isReverseDriving = isReverseDriving
207
208 local dir = (isReverseDriving and 1) or -1
209 self:playAnimation(spec.reverseDrivingAnimation, dir, self:getAnimationTime(spec.reverseDrivingAnimation), true)
210
211 -- deactivate update of character ik chains to prevent strange states after changing
212 local character = self:getVehicleCharacter()
213 if character ~= nil then
214 character:setAllowCharacterUpdate(false)
215 end
216
217 self:setReverserDirection(0)
218 SpecializationUtil.raiseEvent(self, "onStartReverseDirectionChange")
219 ReverseDrivingSetStateEvent.sendEvent(self, isReverseDriving, noEventSend)
220 end
221 end
```

Rideable

Description

Specialization class for Rideables

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
52  function Rideable.prerequisitesPresent(specializations)
53  return SpecializationUtil.hasSpecialization(CCTDrivable,
    specializations) and
54  SpecializationUtil.hasSpecialization(Washable, specializations)
55  end
```

registerFunctions

Description

Registers functions

Definition

registerFunctions(string vehicleType)

Arguments

string vehicleType type of vehicle

Code

```
60  function Rideable.registerFunctions(vehicleType)
61  SpecializationUtil.registerFunction(vehicleType, "jump",
    Rideable.jump)
62  SpecializationUtil.registerFunction(vehicleType, "resetInputs",
    Rideable.resetInputs)
63  SpecializationUtil.registerFunction(vehicleType,
    "updateKinematic", Rideable.updateKinematic)
64  SpecializationUtil.registerFunction(vehicleType, "testCCTMove",
    Rideable.testCCTMove)
65  SpecializationUtil.registerFunction(vehicleType,
    "updateAnimation", Rideable.updateAnimation)
66  SpecializationUtil.registerFunction(vehicleType, "updateSound",
    Rideable.updateSound)
67  SpecializationUtil.registerFunction(vehicleType, "updateFitness",
    Rideable.updateFitness)
68  SpecializationUtil.registerFunction(vehicleType, "updateDirt",
    Rideable.updateDirt)
69  SpecializationUtil.registerFunction(vehicleType,
    "calculateLegsDistance", Rideable.calculateLegsDistance)
70  SpecializationUtil.registerFunction(vehicleType,
    "setWorldPositionQuat", Rideable.setWorldPositionQuat)
71  SpecializationUtil.registerFunction(vehicleType,
    "setShaderParameter", Rideable.setShaderParameter)
72  SpecializationUtil.registerFunction(vehicleType,
    "getShaderParameter", Rideable.getShaderParameter)
73  SpecializationUtil.registerFunction(vehicleType,
    "updateFootsteps", Rideable.updateFootsteps)
74  SpecializationUtil.registerFunction(vehicleType, "getPosition",
    Rideable.getPosition)
75  SpecializationUtil.registerFunction(vehicleType, "getRotation",
    Rideable.getRotation)
```

```
76  SpecializationUtil.registerFunction(vehicleType, "setDirtScale",
    Rideable.setDirtScale)
77  SpecializationUtil.registerFunction(vehicleType, "getDirtScale",
    Rideable.getDirtScale)
78  SpecializationUtil.registerFunction(vehicleType,
    "setFitnessChangedCallback", Rideable.setFitnessChangedCallback)
79  SpecializationUtil.registerFunction(vehicleType,
    "setDirtChangedCallback", Rideable.setDirtChangedCallback)
80  SpecializationUtil.registerFunction(vehicleType,
    "isOnHusbandyGround", Rideable.isOnHusbandyGround)
81  SpecializationUtil.registerFunction(vehicleType,
    "setEquipmentVisibility", Rideable.setEquipmentVisibility)
82  SpecializationUtil.registerFunction(vehicleType, "abandonCheck",
    Rideable.abandonCheck)
83  SpecializationUtil.registerFunction(vehicleType,
    "getHoofSurfaceSound", Rideable.getHoofSurfaceSound)
84  SpecializationUtil.registerFunction(vehicleType, "removeRideable",
    Rideable.removeRideable)
85  SpecializationUtil.registerFunction(vehicleType, "setAnimal",
    Rideable.setAnimal)
86  SpecializationUtil.registerFunction(vehicleType,
    "groundRaycastCallback", Rideable.groundRaycastCallback)
87  SpecializationUtil.registerFunction(vehicleType, "unlinkReins",
    Rideable.unlinkReins)
88  SpecializationUtil.registerFunction(vehicleType,
    "updateInputText", Rideable.updateInputText)
89  SpecializationUtil.registerFunction(vehicleType,
    "setPlayerToEnter", Rideable.setPlayerToEnter)
90  SpecializationUtil.registerFunction(vehicleType, "endFade",
    Rideable.endFade)
91  end
```

registerEventListeners

Description

Registers event listeners

Definition

registerEventListeners(string vehicleType)

Arguments

string vehicleType type of vehicle

Code

```
96  function Rideable.registerEventListeners(vehicleType)
97  SpecializationUtil.registerEventListener(vehicleType, "onLoad",
    Rideable)
98  SpecializationUtil.registerEventListener(vehicleType, "onDelete",
    Rideable)
```

```
 99  SpecializationUtil.registerEventListener(vehicleType,
     "onReadStream", Rideable)
100  SpecializationUtil.registerEventListener(vehicleType,
     "onWriteStream", Rideable)
101  SpecializationUtil.registerEventListener(vehicleType,
     "onReadUpdateStream", Rideable)
102  SpecializationUtil.registerEventListener(vehicleType,
     "onWriteUpdateStream", Rideable)
103  SpecializationUtil.registerEventListener(vehicleType,
     "onReadPositionUpdateStream", Rideable)
104  SpecializationUtil.registerEventListener(vehicleType,
     "onWritePositionUpdateStream", Rideable)
105  SpecializationUtil.registerEventListener(vehicleType, "onUpdate",
     Rideable)
106  SpecializationUtil.registerEventListener(vehicleType,
     "onUpdateInterpolation", Rideable)
107  SpecializationUtil.registerEventListener(vehicleType,
     "onRegisterActionEvents", Rideable)
108  SpecializationUtil.registerEventListener(vehicleType,
     "onEnterVehicle", Rideable)
109  SpecializationUtil.registerEventListener(vehicleType,
     "onSetBroken", Rideable)
110  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
123  function Rideable:onLoad(savegame)
124  local spec = self.spec_rideable
125
126  -- Overwrite the Vehicle high precision setting. Otherwise the precision
     might lead to large jitter in speed in multiplayer
127  self.highPrecisionPositionSynchronization = true
128
129  self.isVehicleSaved = false
130  spec.currentDirtScale = 0
131  spec.abandonTimerDuration =
     g_gameSettings:getValue("horseAbandonTimerDuration")
132  spec.abandonTimer = spec.abandonTimerDuration
133  spec.fadeDuration = 400
```

```
spec.isRideableRemoved = false
spec.justSpawned = true
spec.meshNode = nil
spec.hairNode = nil
-- Animation
spec.animationNode = nil
spec.charsetId = nil
spec.animationPlayer = 0
spec.animationParameters = {}
spec.animationParameters.forwardVelocity = {id=1, value=0.0, type=1}
spec.animationParameters.verticalVelocity = {id=2, value=0.0, type=1}
spec.animationParameters.yawVelocity = {id=3, value=0.0, type=1}
spec.animationParameters.absForwardVelocity = {id=4, value=0.0, type=1}
spec.animationParameters.onGround = {id=5, value=false, type=0}
spec.animationParameters.inWater = {id=6, value=false, type=0}
spec.animationParameters.closeToGround = {id=7, value=false, type=0}
spec.animationParameters.leftRightWeight = {id=8, value=0.0, type=1}
spec.animationParameters.absYawVelocity = {id=9, value=0.0, type=1}
spec.animationParameters.halted = {id=10, value=false, type=0}
spec.animationParameters.smoothedForwardVelocity = {id=11, value=0.0,
type=1}
spec.animationParameters.absSmoothedForwardVelocity = {id=12, value=0.0,
type=1}

-- InputAction
spec.acceletateEventId = ""
spec.brakeEventId = ""
spec.steerEventId = ""
spec.jumpEventId = ""

-- movements
spec.currentTurnAngle = 0
spec.currentTurnSpeed = 0.0
spec.currentSpeed = 0.0
spec.currentSpeedY = 0.0
spec.isInWater = false
spec.cctMoveQueue = {}
spec.currentCCTPosX = 0.0
spec.currentCCTPosY = 0.0
spec.currentCCTPosZ = 0.0
spec.lastCCTPosX = 0.0
```

```
173 spec.lastCCTPosY = 0.0
174 spec.lastCCTPosZ = 0.0
175 spec.topSpeeds = {}
176 spec.topSpeeds[Rideable.GAITTYPES.BACKWARDS] =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.rideable#speedBackwards"), -1.0)
177 spec.topSpeeds[Rideable.GAITTYPES.STILL] = 0.0
178 spec.topSpeeds[Rideable.GAITTYPES.WALK] =
    Utils.getNoNil(getXMLFloat(self.xmlFile, "vehicle.rideable#speedWalk"),
    2.5)
179 spec.topSpeeds[Rideable.GAITTYPES.CANTER] =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.rideable#speedCanter"), 3.5)
180 spec.topSpeeds[Rideable.GAITTYPES.TROT] =
    Utils.getNoNil(getXMLFloat(self.xmlFile, "vehicle.rideable#speedTrot"),
    5.0)
181 spec.topSpeeds[Rideable.GAITTYPES.GALLOP] =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.rideable#speedGallop"), 10.0)
182 spec.minTurnRadius = {}
183 spec.minTurnRadius[Rideable.GAITTYPES.BACKWARDS] =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.rideable#minTurnRadiusBackwards"), 1.0)
184 spec.minTurnRadius[Rideable.GAITTYPES.STILL] = 1.0
185 spec.minTurnRadius[Rideable.GAITTYPES.WALK] =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.rideable#minTurnRadiusWalk"), 1.0)
186 spec.minTurnRadius[Rideable.GAITTYPES.CANTER] =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.rideable#minTurnRadiusCanter"), 2.5)
187 spec.minTurnRadius[Rideable.GAITTYPES.TROT] =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.rideable#minTurnRadiusTrot"), 5.0)
188 spec.minTurnRadius[Rideable.GAITTYPES.GALLOP] =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.rideable#minTurnRadiusGallop"), 10.0)
189 spec.groundRaycastResult = {}
190 spec.groundRaycastResult.y = 0.0
191 spec.groundRaycastResult.object = nil
192 spec.groundRaycastResult.distance = 0.0
193 spec.haltTimer = 0.0
194 spec.smoothedLeftRightWeight = 0.0
195
196 -- interpolation
197 -- spec.interpolationTime = InterpolationTime:new(1.0)
```

```
198 -- spec.interpolatorPosition = InterpolatorPosition:new(0.0, 0.0, 0.0)
199 -- spec.interpolatorQuaternion = InterpolatorQuaternion:new(0.0, 0.0,
    0.0, 1.0) -- only used on server side for rotation of camera
200 -- spec.interpolatorOnGround = InterpolatorValue:new(0.0)
201
202 -- steer
203 spec.maxAcceleration = 5 -- m/s^2
204 spec.maxDeceleration = 10 -- m/s^2
205 spec.gravity = -18.8
206
207 -- ground orientation
208 spec.frontCheckDistance = 0.0
209 spec.backCheckDistance = 0.0
210 spec.isOnGround = true
211 spec.isCloseToGround = true
212
213 assert(spec.topSpeeds[Rideable.GAITTYPES.MIN] <
    spec.topSpeeds[Rideable.GAITTYPES.MAX])
214 spec.maxTurnSpeed = math.rad(Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.rideable#turnSpeed"), 45.0)) -- xml: deg/s, script: rad/s
215 spec.jumpHeight = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.rideable#jumpHeight"), 2.0)
216
217 local function loadHoof(target, index, key)
218 local hoof = {}
219 hoof.node = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, key.."#node"), self.i3dMappings)
220 -- hoof.psNode = createTransformGroup("psLinkNode")
221 -- link(hoof.node, hoof.psNode)
222 hoof.onGround = false
223 hoof.psSlow = {}
224 ParticleUtil.loadParticleSystem(self.xmlFile, hoof.psSlow,
    key..".particleSystemSlow", getRootNode(), false, nil,
    self.baseDirectory)
225 hoof.psFast = {}
226 ParticleUtil.loadParticleSystem(self.xmlFile, hoof.psFast,
    key..".particleSystemFast", getRootNode(), false, nil,
    self.baseDirectory)
227 target[index] = hoof
228 end
229
230 -- Hooves
```

```
231 spec.hooves = {}
232 loadHoof(spec.hooves, Rideable.HOOVES.FRONT_LEFT,
    "vehicle.rideable.modelInfo.hoofFrontLeft")
233 loadHoof(spec.hooves, Rideable.HOOVES.FRONT_RIGHT,
    "vehicle.rideable.modelInfo.hoofFrontRight")
234 loadHoof(spec.hooves, Rideable.HOOVES.BACK_LEFT,
    "vehicle.rideable.modelInfo.hoofBackLeft")
235 loadHoof(spec.hooves, Rideable.HOOVES.BACK_RIGHT,
    "vehicle.rideable.modelInfo.hoofBackRight")
236
237 spec.frontCheckDistance =
    self:calculateLegsDistance(spec.hooves[Rideable.HOOVES.FRONT_LEFT].node,
    spec.hooves[Rideable.HOOVES.FRONT_RIGHT].node)
238 spec.backCheckDistance =
    self:calculateLegsDistance(spec.hooves[Rideable.HOOVES.BACK_LEFT].node,
    spec.hooves[Rideable.HOOVES.BACK_RIGHT].node)
239
240 spec.animationNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, "vehicle.rideable.modelInfo#animationNode"),
    self.i3dMappings)
241 spec.meshNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, "vehicle.rideable.modelInfo#meshNode"),
    self.i3dMappings)
242 spec.hairNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, "vehicle.rideable.modelInfo#hairNode"),
    self.i3dMappings)
243 spec.equipmentNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, "vehicle.rideable.modelInfo#equipmentNode"),
    self.i3dMappings)
244 spec.reinsNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, "vehicle.rideable.modelInfo#reinsNode"),
    self.i3dMappings)
245 spec.leftReinNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, "vehicle.rideable.modelInfo#reinLeftNode"),
    self.i3dMappings)
246 spec.rightReinNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, "vehicle.rideable.modelInfo#reinRightNode"),
    self.i3dMappings)
247 spec.leftReinParentNode = getParent(spec.leftReinNode)
248 spec.rightReinParentNode = getParent(spec.rightReinNode)
249
250 -- animation
251 if spec.animationNode ~= nil then
252 spec.charsetId = getAnimCharacterSet(spec.animationNode)
253 spec.animationPlayer = createConditionalAnimation()
254 for key, parameter in pairs(spec.animationParameters) do
```

```
255 conditionalAnimationRegisterParameter(spec.animationPlayer,
    parameter.id, parameter.type, key)
256 end
257 initConditionalAnimation(spec.animationPlayer, spec.charsetId,
    self.configFileName, "vehicle.conditionalAnimation")
258 setConditionalAnimationSpecificParameterIds(spec.animationPlayer,
    spec.animationParameters.absForwardVelocity.id,
    spec.animationParameters.absYawVelocity.id)
259 end
260
261 -- Sounds
262 spec.surfaceSounds = {}
263 spec.surfaceIdToSound = {}
264 spec.surfaceNameToSound = {}
265 spec.currentSurfaceSound = nil
266 for _, surfaceSound in pairs(g_currentMission.surfaceSounds) do
267 if surfaceSound.type == "hoofstep" and surfaceSound.sample ~= nil then
268 local sample = g_soundManager:cloneSample(surfaceSound.sample,
    self.components[1].node, self)
269 sample.sampleName = surfaceSound.name
270
271 table.insert(spec.surfaceSounds, sample)
272 spec.surfaceIdToSound[surfaceSound.materialId] = sample
273 spec.surfaceNameToSound[surfaceSound.name] = sample
274 end
275 end
276 spec.horseStopSound = g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.rideable.sounds", "halt", self.baseDirectory, self.components,
    1, AudioGroup.VEHICLE, self.i3dMappings, self)
277 spec.horseBreathSoundsNoEffort =
    g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.rideable.sounds", "breathingNoEffort", self.baseDirectory,
    self.components, 1, AudioGroup.VEHICLE, self.i3dMappings, self)
278 spec.horseBreathSoundsEffort =
    g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.rideable.sounds", "breathingEffort", self.baseDirectory,
    self.components, 1, AudioGroup.VEHICLE, self.i3dMappings, self)
279 spec.horseBreathIntervalNoEffort =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.rideable.sounds#breathIntervalNoEffort"), 1.0) * 1000.0
280 spec.horseBreathIntervalEffort =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.rideable.sounds#breathIntervalEffort"), 1.0) * 1000.0
```

```
281 spec.horseBreathMinIntervalIdle =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.rideable.sounds#minBreathIntervalIdle"), 1.0) * 1000.0
282 spec.horseBreathMaxIntervalIdle =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.rideable.sounds#maxBreathIntervalIdle"), 1.0) * 1000.0
283 spec.currentBreathTimer = 0.0
284
285 -- attributes set by action events
286 spec.inputValues = {}
287 spec.inputValues.axisSteer = 0.0
288 spec.inputValues.axisSteerSend = 0.0
289 spec.inputValues.currentGait = Rideable.GAITTYPES.STILL
290 self:resetInputs()
291
292 spec.interpolatorIsOnGround = InterpolatorValue:new(0.0)
293 if self.isServer then
294 spec.interpolatorTurnAngle = InterpolatorAngle:new(0.0)
295 end
296
297 -- Network
298 spec.dirtyFlag = self:getNextDirtyFlag()
299 end
```

calculateLegsDistance

Description

Gets legs distance from rootNode

Definition

calculateLegsDistance(integer left, integer right)

Arguments

integer left leg node

integer right leg node

Return Values

float distance from root node

Code

```
332 function Rideable:calculateLegsDistance(leftLegNode,
    rightLegNode)
333 local distance = 0.0
334 if leftLegNode ~= nil and rightLegNode ~= nil then
335 local dxL, dyL, dzL = localToLocal(leftLegNode, self.rootNode,
    0.0, 0.0, 0.0)
336 local dxR, dyR, dzR = localToLocal(rightLegNode, self.rootNode,
    0.0, 0.0, 0.0)
337 distance = (dzL + dzR) * 0.5
```

```
338 end
339 return distance
340 end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
344 function Rideable:onDelete()
345 local spec = self.spec_rideable
346 g_soundManager:deleteSamples(spec.surfaceSounds)
347 g_soundManager:deleteSample(spec.horseStopSound)
348 g_soundManager:deleteSample(spec.horseBreathSoundsNoEffort)
349 g_soundManager:deleteSample(spec.horseBreathSoundsEffort)
350
351 for _, d in pairs(spec.hooves) do
352 ParticleUtil.deleteParticleSystem(d.psSlow)
353 ParticleUtil.deleteParticleSystem(d.psFast)
354 end
355
356 if spec.animationPlayer ~= 0 then
357 delete(spec.animationPlayer)
358 spec.animationPlayer = 0
359 end
360 end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
366 function Rideable:onReadStream(streamId, connection)
367 local spec = self.spec_rideable
368 if connection:getIsServer() then
369 local isOnGround = streamReadBool(streamId)
370 if isOnGround then
371 spec.interpolatorIsOnGround:setValue(1.0)
```

```
372 else
373 spec.interpolatorIsOnGround:setValue(0.0)
374 end
375 end
376 if streamReadBool(streamId) then
377 local animal = NetworkUtil.readNodeObject(streamId)
378 self:setAnimal(animal)
379 end
380 if streamReadBool(streamId) then
381 local player = NetworkUtil.readNodeObject(streamId)
382 self:setPlayerToEnter(player)
383 end
384 end
```

onWriteStream

Description

Called on client side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId
integer connection connection

Code

```
390 function Rideable:onWriteStream(streamId, connection)
391 local spec = self.spec_rideable
392 if not connection:getIsServer() then
393 streamWriteBool(streamId, spec.isOnGround)
394 end
395 if streamWriteBool(streamId, spec.animal ~= nil) then
396 NetworkUtil.writeNodeObject(streamId, spec.animal)
397 end
398 if streamWriteBool(streamId, spec.playerToEnter ~= nil) then
399 NetworkUtil.writeNodeObject(streamId, spec.playerToEnter)
400 end
401 end
```

onReadUpdateStream

Description

Called on on update

Definition

onReadUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId stream ID

integer timestamp timestamp

table connection connection

Code

```
408 function Rideable:onReadUpdateStream(streamId, timestamp,
    connection)
409 local spec = self.spec_rideable
410
411 if not connection:getIsServer() then
412 spec.inputValues.axisSteer = streamReadFloat32(streamId)
413 spec.inputValues.currentGait = streamReadUInt8(streamId)
414 else
415 spec.haltTimer = streamReadFloat32(streamId)
416 if spec.haltTimer > 0 then
417 spec.inputValues.currentGait = Rideable.GAITTYPES.STILL
418 spec.inputValues.axisSteerSend = 0
419 end
420 end
421 end
```

onWriteUpdateStream

Description

Called on on update

Definition

onWriteUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId stream ID

table connection connection

integer dirtyMask dirty mask

Code

```
428 function Rideable:onWriteUpdateStream(streamId, connection,
    dirtyMask)
429 local spec = self.spec_rideable
430 if connection:getIsServer() then
431 streamWriteFloat32(streamId, spec.inputValues.axisSteerSend)
432 streamWriteUInt8(streamId, spec.inputValues.currentGait)
433 else
434 streamWriteFloat32(streamId, spec.haltTimer)
435 end
436 end
```

onReadPositionUpdateStream

Description

Definition

onReadPositionUpdateStream(integer streamId, integer connection)

Arguments

integer streamId	streamId
integer connection	connection

Code

```
function Rideable:onReadPositionUpdateStream(streamId, connection)
local spec = self.spec_rideable
local isOnGround = streamReadBool(streamId)
if isOnGround then
spec.interpolatorIsOnGround:setValue(1.0)
else
spec.interpolatorIsOnGround:setValue(0.0)
end
end
```

onWritePositionUpdateStream

Description

Definition

onWritePositionUpdateStream(integer streamId, integer connection)

Arguments

integer streamId	streamId
integer connection	connection

Code

```
function Rideable:onWritePositionUpdateStream(streamId, connection, dirtyMask)
local spec = self.spec_rideable
streamWriteBool(streamId, spec.isOnGround)
end
```

endFade

Description

Definition

endFade()

Code

```
function Rideable:endFade()
end
```

onUpdate

Description

Called on on update

Definition

onUpdate(float dt, bool isActiveForInput, bool isSelected)

Arguments

float dt	delta time
bool isActiveForInput	true if specializations is active for input
bool isSelected	true if specializations is selected

Code

```
476 function Rideable:onUpdate(dt, isActiveForInput, isSelected)
477 local spec = self.spec_rideable
478
479 if spec.playerToEnter ~= nil and spec.checkPlayerToEnter then
480 if spec.playerToEnter == g_currentMission.player then
481 g_currentMission:requestToEnterVehicle(self)
482 spec.checkPlayerToEnter = false
483 end
484 self:raiseActive()
485 end
486
487 local isEntered = self:getIsEntered()
488 local isControlled = self:getIsControlled()
489
490 if isEntered then
491 if isActiveForInput then
492 self:updateInputText()
493 end
494 if not self.isServer then
495 spec.inputValues.axisSteerSend = spec.inputValues.axisSteer
496 self:raiseDirtyFlags(spec.dirtyFlag)
497 self:resetInputs()
498 end
499 end
500
501 if spec.isOnGround and spec.justSpawned then
502 spec.justSpawned = false
503 if (not spec.checkPlayerToEnter) or (spec.checkPlayerToEnter and
    spec.playerToEnter == g_currentMission.player) then
504 g_currentMission:fadeScreen(-1, spec.fadeDuration, self.endFade,
    self)
505 end
506 end
507
508 self:updateAnimation(dt)
509 if self.isClient then
510 self:updateFootsteps(dt)
511 self:updateSound(dt)
512 end
513
```

```
514 if self.isServer then
515 self:updateFitness(dt)
516 self:updateDirt(dt)
517 end
518
519 if spec.haltTimer > 0 then
520 spec.inputValues.currentGait = Rideable.GAITTYPES.STILL
521 spec.haltTimer = spec.haltTimer - dt
522 end
523
524 if self.isServer then
525 if not isEntered and not isControlled and spec.playerToEnter ==
    nil then
526 self:abandonCheck(dt)
527 end
528 end
529 end
```

onSetBroken

Description

Definition

onSetBroken()

Code

```
600 function Rideable:onSetBroken()
601 self:removeRideable()
602 self:unlinkReins()
603 local spec = self.spec_rideable
604 g_currentMission:addIngameNotification(FSBaseMission.INGAME_NOTIFICATION
    string.format(g_i18n:getText("ingameNotification_horseInStable"),
    spec.animal:getName()))
605 end
```

testCCTMove

Description

Check if a requested CCT move was successful. We need a range for the error because of possible huge fps fluctuation.

Definition

testCCTMove()

Code

```
610 function Rideable:testCCTMove(dt)
611 local spec = self.spec_rideable
612 spec.lastCCTPosX, spec.lastCCTPosY, spec.lastCCTPosZ =
    spec.currentCCTPosX, spec.currentCCTPosY, spec.currentCCTPosZ
613 spec.currentCCTPosX, spec.currentCCTPosY, spec.currentCCTPosZ =
    getWorldTranslation(self.spec_cctdrivable.cctNode)
```

```
614
615 local expectedMovementX, expectedMovementZ = 0,0
616
617 while spec.cctMoveQueue[1] ~= nil and
    getIsPhysicsUpdateIndexSimulated(spec.cctMoveQueue[1].physicsIndex)
    do
618 expectedMovementX = expectedMovementX + spec.cctMoveQueue[1].moveX
619 expectedMovementZ = expectedMovementZ + spec.cctMoveQueue[1].moveZ
620 table.remove(spec.cctMoveQueue, 1)
621 end
622
623 local expectedMovement = math.sqrt(expectedMovementX *
    expectedMovementX + expectedMovementZ * expectedMovementZ)
624 if expectedMovement > 0.001*dt then -- only check if we are
    supposed to move faster than 3.6km/h
625 local movementX = spec.currentCCTPosX - spec.lastCCTPosX
626 local movementZ = spec.currentCCTPosZ - spec.lastCCTPosZ
627 local movement = math.sqrt(movementX * movementX + movementZ *
    movementZ)
628 if movement <= expectedMovement*0.7 then
629 --print(string.format("-- [Rideable:testCCTMove] movement(%.3f),
    expectedMovement(%.3f)", movement, expectedMovement))
630 spec.inputValues.currentGait = Rideable.GAITTYPES.STILL
631 spec.haltTimer = 900
632 if spec.horseStopSound ~= nil then
633 g_soundManager:playSample(spec.horseStopSound)
634 end
635 end
636 end
637 end
```

jump

Description

Definition

jump()

Code

```
641 function Rideable:jump()
642 local spec = self.spec_rideable
643
644 if not self.isServer then
645 g_client:getServerConnection():sendEvent(JumpEvent:new(self))
646 end
647
```

```
local velY = math.sqrt(-2.0 * spec.gravity * spec.jumpHeight)
spec.currentSpeedY = velY
end
```

resetInputs

Description

Resets all inputs

Definition

resetInputs()

Code

```
function Rideable:resetInputs()
local enterable = self.spec_enterable

local spec = self.spec_rideable
spec.inputValues.axisSteer = 0
end
```

updateKinematic

Description

Update animal kinematic; if we reach max speed? we fix velocity else we add force to accelerate. If we need to break, we add a break force. At the end we add gravity force and change direction when needed.

Definition

updateKinematic(float dt)

Arguments

float dt delta time in ms

Code

```
function Rideable:updateKinematic(dt)
local spec = self.spec_rideable
local dtInSec = dt*0.001

-- Update movement in current direction
local desiredSpeed = spec.topSpeeds[spec.inputValues.currentGait]
local maxSpeedChange = spec.maxAcceleration
if desiredSpeed == 0.0 then
maxSpeedChange = spec.maxDeceleration
end
maxSpeedChange = maxSpeedChange * dtInSec
if not spec.isOnGround then
-- reduce acceleration when in the air
maxSpeedChange = maxSpeedChange * 0.2
end

```

```lua
local speedChange = (desiredSpeed - spec.currentSpeed)
speedChange = MathUtil.clamp(speedChange, -maxSpeedChange, maxSpeedChange)

--local movement = (spec.currentSpeed + 0.5 * speedChange) * dtInSec
spec.currentSpeed = spec.currentSpeed + speedChange
local movement = spec.currentSpeed * dtInSec

-- Update gravity / vertical movement
local gravitySpeedChange = spec.gravity * dtInSec
spec.currentSpeedY = spec.currentSpeedY + gravitySpeedChange
local movementY = spec.currentSpeedY * dtInSec

-- Update rotation
local slowestSpeed = spec.topSpeeds[Rideable.GAITTYPES.WALK]
local fastestSpeed = spec.topSpeeds[Rideable.GAITTYPES.MAX]

local maxTurnSpeedChange = (MathUtil.clamp((fastestSpeed - spec.currentSpeed) / (fastestSpeed - slowestSpeed), 0, 1) * 0.4 + 0.8) -- Use smaller changes when walking slowly (between 0.8 and 1.2 rad/s^2)
maxTurnSpeedChange = maxTurnSpeedChange * dtInSec
if not spec.isOnGround then
-- reduce turn acceleration when in the air
maxTurnSpeedChange = maxTurnSpeedChange * 0.25
end

local desiredTurnSpeed = spec.maxTurnSpeed * spec.inputValues.axisSteer
local turnSpeedChange = (desiredTurnSpeed - spec.currentTurnSpeed)
turnSpeedChange = MathUtil.clamp(turnSpeedChange, -maxTurnSpeedChange, maxTurnSpeedChange)
spec.currentTurnSpeed = spec.currentTurnSpeed + turnSpeedChange
spec.currentTurnAngle = spec.currentTurnAngle + spec.currentTurnSpeed * dtInSec * (movement >= 0 and 1 or -1)

local movementX, movementZ = math.sin(spec.currentTurnAngle) * movement, math.cos(spec.currentTurnAngle) * movement
self:moveCCT(movementX, movementY, movementZ, true)
```

```
711  table.insert(spec.cctMoveQueue, {physicsIndex = getPhysicsUpdateIndex(), moveX = movementX, moveY = movementY, moveZ = movementZ})
712  end
```

groundRaycastCallback

Description

Callback used when raycast hists an object. Updates player information so it can be used to pickup the object.

Definition

groundRaycastCallback(integer hitObjectId, float x, float y, float z, float distance)

Arguments

integer	hitObjectId	scenegraph object id
float	x	world x hit position
float	y	world y hit position
float	z	world z hit position
float	distance	distance at which the cast hit the object

Return Values

bool returns true object that was hit is valid

Code

```
722  function Rideable:groundRaycastCallback(hitObjectId, x, y, z, distance)
723  local spec = self.spec_rideable
724  if hitObjectId == self.spec_cctdrivable.cctNode then
725  return true
726  end
727  -- DebugUtil.drawSimpleDebugCube(x, y, z, 0.05, 1, 0, 0)
728  spec.groundRaycastResult.y = y
729  spec.groundRaycastResult.object = hitObjectId
730  spec.groundRaycastResult.distance = distance
731
732  return false
733  end
```

updateAnimation

Description

Updates the parameters that will drive the animation

Definition

updateAnimation(float dt)

Arguments

float dt delta time in ms

Code

```
738  function Rideable:updateAnimation(dt)
739  local spec = self.spec_rideable
740  local params = spec.animationParameters
```

```
local speed = self.lastSignedSpeedReal * 1000.0
local smoothedSpeed = self.lastSignedSpeed * 1000.0
speed = MathUtil.clamp(speed,
spec.topSpeeds[Rideable.GAITTYPES.BACKWARDS],
spec.topSpeeds[Rideable.GAITTYPES.MAX])
smoothedSpeed = MathUtil.clamp(smoothedSpeed,
spec.topSpeeds[Rideable.GAITTYPES.BACKWARDS],
spec.topSpeeds[Rideable.GAITTYPES.MAX])

local turnSpeed
if self.isServer then
turnSpeed = (spec.interpolatorTurnAngle.targetValue -
spec.interpolatorTurnAngle.lastValue) /
(self.networkTimeInterpolator.interpolationDuration * 0.001)
else
local interpQuat =
self.components[1].networkInterpolators.quaternion
local lastDirX, lastDirY, lastDirZ =
mathQuaternionRotateVector(interpQuat.lastQuaternionX,
interpQuat.lastQuaternionY, interpQuat.lastQuaternionZ,
interpQuat.lastQuaternionW, 0,0,1)
local targetDirX, targetDirY, targetDirZ =
mathQuaternionRotateVector(interpQuat.targetQuaternionX,
interpQuat.targetQuaternionY, interpQuat.targetQuaternionZ,
interpQuat.targetQuaternionW, 0,0,1)
local lastTurnAngle =
MathUtil.getYRotationFromDirection(lastDirX, lastDirZ)
local targetTurnAngle =
MathUtil.getYRotationFromDirection(targetDirX, targetDirZ)
local turnAngleDiff = targetTurnAngle - lastTurnAngle
-- normalize to -180,180deg
if turnAngleDiff > math.pi then
turnAngleDiff = turnAngleDiff - 2*math.pi
elseif turnAngleDiff < -math.pi then
turnAngleDiff = turnAngleDiff + 2*math.pi
end
turnSpeed = turnAngleDiff /
(self.networkTimeInterpolator.interpolationDuration * 0.001)
end

local interpPos =
self.components[1].networkInterpolators.position
local speedY = (interpPos.targetPositionY -
interpPos.lastPositionY) /
(self.networkTimeInterpolator.interpolationDuration * 0.001)
```

```lua

local leftRightWeight = 0
if math.abs(speed) > 0.01 then
local closestGait = Rideable.GAITTYPES.STILL
local closestDiff = math.huge
for i=1,Rideable.GAITTYPES.MAX do
local diff = math.abs(speed - spec.topSpeeds[i])
if diff < closestDiff then
closestGait = i
closestDiff = diff
end
end
local minTurnRadius = spec.minTurnRadius[closestGait]
leftRightWeight = minTurnRadius * turnSpeed / speed
else
leftRightWeight = turnSpeed / spec.maxTurnSpeed
end
if leftRightWeight < spec.smoothedLeftRightWeight then
spec.smoothedLeftRightWeight = math.max(leftRightWeight, spec.smoothedLeftRightWeight-1/500*dt, -1)
else
spec.smoothedLeftRightWeight = math.min(leftRightWeight, spec.smoothedLeftRightWeight+1/500*dt, 1)
end

-- print(string.format("-- [Rideable:updateAnimation] \t%.6f\t%.6f\t%.6f\t%.6f\t%.6f\t%.6f\t%.6f\t%.6f\t%.6f",
-- leftRightWeight, spec.smoothedLeftRightWeight, MathUtil.clamp(leftRightWeight, -1.0, 1.0),
-- turnSpeed / spec.maxTurnSpeed, MathUtil.clamp(turnSpeed / spec.maxTurnSpeed, -1.0, 1.0),
-- turnSpeed, speed, self.movingDirection, self.lastSpeedAcceleration * 1000 * 1000))

params.forwardVelocity.value = speed
params.absForwardVelocity.value = math.abs(speed)
params.verticalVelocity.value = speedY
params.yawVelocity.value = turnSpeed
params.absYawVelocity.value = math.abs(turnSpeed)
params.leftRightWeight.value = spec.smoothedLeftRightWeight
params.onGround.value = spec.isOnGround or spec.justSpawned
```

```
803 params.closeToGround.value = spec.isCloseToGround
804 params.inWater.value = spec.isInWater
805 params.halted.value = spec.haltTimer > 0
806 params.smoothedForwardVelocity.value = smoothedSpeed
807 params.absSmoothedForwardVelocity.value = math.abs(smoothedSpeed)
808
809 -- horse animation
810 if spec.animationPlayer ~= 0 then
811 for _, parameter in pairs(params) do
812 if parameter.type == 0 then
813 setConditionalAnimationBoolValue(spec.animationPlayer,
    parameter.id, parameter.value)
814 elseif parameter.type == 1 then
815 setConditionalAnimationFloatValue(spec.animationPlayer,
    parameter.id, parameter.value)
816 end
817 end
818 updateConditionalAnimation(spec.animationPlayer, dt)
819 -- local x,y,z = getWorldTranslation(self.rootNode)
820 -- conditionalAnimationDebugDraw(spec.animationPlayer, x,y,z)
821 end
822 local isEntered = self.getIsEntered ~= nil and
    self:getIsEntered()
823 local isControlled = self.getIsControlled ~= nil and
    self:getIsControlled()
824
825 if isEntered or isControlled then
826 -- rider animation
827 local character = self:getVehicleCharacter()
828 if character ~= nil and character.animationCharsetId ~= 0 and
    character.animationPlayer ~= nil then
829 for _, parameter in pairs(params) do
830 if parameter.type == 0 then
831 setConditionalAnimationBoolValue(character.animationPlayer,
    parameter.id, parameter.value)
832 elseif parameter.type == 1 then
833 setConditionalAnimationFloatValue(character.animationPlayer,
    parameter.id, parameter.value)
834 end
835 end
836 updateConditionalAnimation(character.animationPlayer, dt)
837 -- local x,y,z = getWorldTranslation(self.rootNode)
```

```
838  -- conditionalAnimationDebugDraw(character.animationPlayer,
     x,y,z)
839  end
840  end
841  end
```

updateSound

Description

Definition

updateSound()

Code

```
845  function Rideable:updateSound(dt)
846  local spec = self.spec_rideable
847
848  if spec.horseBreathSoundsEffort ~= nil and
     spec.horseBreathSoundsNoEffort ~= nil and spec.isOnGround then
849  spec.currentBreathTimer = spec.currentBreathTimer - dt
850  spec.currentBreathTimer = math.max(spec.currentBreathTimer, 0.0)
851
852  if spec.currentBreathTimer == 0.0 then
853  if spec.inputValues.currentGait == Rideable.GAITTYPES.GALLOP then
854  g_soundManager:playSample(spec.horseBreathSoundsEffort)
855  spec.currentBreathTimer = spec.horseBreathIntervalEffort
856  else
857  g_soundManager:playSample(spec.horseBreathSoundsNoEffort)
858  if spec.inputValues.currentGait == Rideable.GAITTYPES.STILL then
859  spec.currentBreathTimer = spec.horseBreathMinIntervalIdle +
     (math.random() * (spec.horseBreathMaxIntervalIdle -
     spec.horseBreathMinIntervalIdle))
860  else
861  spec.currentBreathTimer = spec.horseBreathIntervalNoEffort
862  end
863  end
864  end
865  end
866  end
```

setWorldPositionQuat

Description

Set world position and quaternion rotation of component

Definition

setWorldPositionQuat(float x, float y, float z, float qx, float qy, float qz, float qw, integer i, boolean changeInterp)

Arguments

float	x	x position
float	y	y position
float	z	z position
float	qx	x rotation
float	qy	y rotation
float	qz	z rotation
float	qw	w rotation
integer	i	index if component
boolean	changeInterp	change interpolation

Code

```
879  function Rideable:setWorldPositionQuat(x,y,z, qx,qy,qz,qw,
     changeInterp)
880  setWorldTranslation(self.rootNode, x,y,z)
881  setWorldQuaternion(self.rootNode, qx,qy,qz,qw)
882  if changeInterp then
883  local spec = self.spec_rideable
884  spec.networkInterpolators.position:setPosition(x,y,z)
885  spec.networkInterpolators.quaternion:setQuaternion(qx, qy, qz,
     qw)
886  end
887  end
```

setShaderParameter

Description

Set shader parameter on the shape node

Definition

setShaderParameter(float x, float y, float z, float w, string parameterName)

Arguments

float	x	x parameter
float	y	y parameter
float	z	z parameter
float	w	z parameter
string	parameterName	parameter name

Code

```
896  function Rideable:setShaderParameter(x, y, z, w, parameterName)
897  local spec = self.spec_rideable
898  I3DUtil.setShaderParameterRec(spec.meshNode, parameterName, x, y,
     z, w)
899  I3DUtil.setShaderParameterRec(spec.hairNode, parameterName, x, y,
     z, w)
900  end
```

getShaderParameter

Description

Get shader parameter on the shape node

Definition

getShaderParameter(string parameterName)

Arguments

string parameterName parameter name

Return Values

float x x parameter

float y y parameter

float z z parameter

float w z parameter

Code

```
function Rideable:getShaderParameter(parameterName)
local spec = self.spec_rideable
local x, y, z, w = getShaderParameter(spec.meshNode, parameterName)
return x, y, z, w
end
```

onRegisterActionEvents

Description

Registers action events

Definition

onRegisterActionEvents()

Code

```
function Rideable:onRegisterActionEvents(isActiveForInput)
if self.isClient then
local spec = self.spec_rideable
self:clearActionEventsTable(spec.actionEvents)

if self:getIsActiveForInput(true) then
local actionEventId
_, actionEventId = self:addActionEvent(spec.actionEvents, InputAction.AXIS_ACCELERATE_VEHICLE, self, Rideable.actionEventAccelerate, false, true, false, true, nil)
g_inputBinding:setActionEventTextPriority(actionEventId, GS_PRIO_VERY_HIGH)
g_inputBinding:setActionEventTextVisibility(actionEventId, false)
spec.acceletateEventId = actionEventId

_, actionEventId = self:addActionEvent(spec.actionEvents, InputAction.AXIS_BRAKE_VEHICLE, self, Rideable.actionEventBrake, false, true, false, true, nil)
g_inputBinding:setActionEventTextPriority(actionEventId, GS_PRIO_HIGH)
g_inputBinding:setActionEventTextVisibility(actionEventId, false)
```

```
spec.brakeEventId = actionEventId

_, actionEventId = self:addActionEvent(spec.actionEvents, InputAction.AXIS_MOVE_SIDE_VEHICLE, self, Rideable.actionEventSteer, false, false, true, true, nil)
g_inputBinding:setActionEventTextVisibility(actionEventId, false)
spec.steerEventId = actionEventId

_, actionEventId = self:addActionEvent(spec.actionEvents, InputAction.JUMP, self, Rideable.actionEventJump, false, true, false, true, nil)
g_inputBinding:setActionEventTextPriority(actionEventId, GS_PRIO_VERY_LOW)
g_inputBinding:setActionEventTextVisibility(actionEventId, false)
spec.jumpEventId = actionEventId
end
end
end
```

onEnterVehicle

Description

Called on enter vehicle

Definition

onEnterVehicle(bool isControlling)

Arguments

bool isControlling

unlinkReins

Description

Definition

unlinkReins()

Code

```
function Rideable:unlinkReins()
if self.isClient then
local spec = self.spec_rideable

link(spec.leftReinParentNode, spec.leftReinNode)
link(spec.rightReinParentNode, spec.rightReinNode)
setVisibility(spec.reinsNode, false)
end
end
```

setEquipmentVisibility

Description

Called on leaving the vehicle

Definition

setEquipmentVisibility()

Code

```
 993  function Rideable:setEquipmentVisibility(val)
 994  if self.isClient then
 995  local spec = self.spec_rideable
 996
 997  if spec.equipmentNode ~= nil then
 998  setVisibility(spec.equipmentNode, val)
 999  setVisibility(spec.reinsNode, val)
1000  end
1001  end
1002  end
```

actionEventAccelerate

Description

Callback on accelerate event

Definition

actionEventAccelerate(table self, string actionName, float inputValue, string callbackState, bool isAnalog)

Arguments

table	self	instance
string	actionName	action name
float	inputValue	input value
string	callbackState	
bool	isAnalog	true if input is analog

Code

```
1011  function Rideable.actionEventAccelerate(self, actionName,
      inputValue, callbackState, isAnalog)
1012  local spec = self.spec_rideable
1013  local enterable = self.spec_enterable
1014
1015  if enterable.isEntered and enterable.isControlled and
      spec.haltTimer <= 0 and spec.isOnGround then
1016  spec.inputValues.currentGait =
      math.min(spec.inputValues.currentGait + 1,
      Rideable.GAITTYPES.MAX)
1017  end
1018  end
```

actionEventBrake

Description

Callback on brake event

Definition

actionEventBrake(table self, string actionName, float inputValue, string callbackState, bool isAnalog)

Arguments

table	self	instance
string	actionName	action name
float	inputValue	input value
string	callbackState	
bool	isAnalog	true if input is analog

Code

```
1027  function Rideable.actionEventBrake(self, actionName, inputValue, callbackState, isAnalog)
1028  local spec = self.spec_rideable
1029  if self:getIsEntered() and spec.haltTimer <= 0 and spec.isOnGround then
1030  spec.inputValues.currentGait = math.max(spec.inputValues.currentGait - 1, 1)
1031  end
1032  end
```

actionEventSteer

Description

Callback on steer event

Definition

actionEventSteer(table self, string actionName, float inputValue, string callbackState, bool isAnalog)

Arguments

table	self	instance
string	actionName	action name
float	inputValue	input value
string	callbackState	
bool	isAnalog	true if input is analog

Code

```
1041  function Rideable.actionEventSteer(self, actionName, inputValue, callbackState, isAnalog)
1042  local spec = self.spec_rideable
1043  if self:getIsEntered() and spec.haltTimer <= 0 then
1044  local spec = self.spec_rideable
1045  spec.inputValues.axisSteer = -inputValue
1046  end
1047  end
```

actionEventJump

Description

Callback on jump event

Definition

actionEventJump(table self, string actionName, float inputValue, string callbackState, bool isAnalog)

Arguments

table	self	instance
string	actionName	action name
float	inputValue	input value
string	callbackState	
bool	isAnalog	true if input is analog

Code

```
1056  function Rideable.actionEventJump(self, actionName, inputValue, callbackState, isAnalog)
1057      local spec = self.spec_rideable
1058
1059      local canJump = spec.isOnGround and spec.inputValues.currentGait >= Rideable.GAITTYPES.CANTER and not spec.isInWater
1060      if canJump then
1061          self:jump()
1062      end
1063  end
```

updateFootsteps

Description

Definition

updateFootsteps()

Code

```
1067  function Rideable:updateFootsteps(dt)
1068      local spec = self.spec_rideable
1069
1070      for k, hoofInfo in pairs(spec.hooves) do
1071          local posX, posY, posZ = getWorldTranslation(hoofInfo.node)
1072          spec.groundRaycastResult.object = 0
1073          spec.groundRaycastResult.y = posY - 1
1074          raycastClosest(posX, posY + Rideable.GROUND_RAYCAST_OFFSET, posZ, 0.0, -1.0, 0.0, "groundRaycastCallback", Rideable.GROUND_RAYCAST_MAXDISTANCE, self, Rideable.GROUND_RAYCAST_COLLISIONMASK)
1075
1076          local hitTerrain = spec.groundRaycastResult.object == g_currentMission.terrainRootNode
1077          local terrainY = spec.groundRaycastResult.y
1078          local onGround = ((posY - terrainY) < 0.05)
1079
1080          -- DebugUtil.drawDebugNode(hoofInfo.node, string.format("[%s] (%.6f/%.6f|%s)", getName(hoofInfo.node), posY, terrainY, tostring(((posY - terrainY) < 0.05))))
1081          if onGround and not hoofInfo.onGround then
```

```
1082  local r, g, b, _, _ = getTerrainAttributesAtWorldPos(g_currentMission.terrainRootNode, posX, posY, posZ, true, true, true, true, false)
1083  hoofInfo.onGround = true
1084  -- particles
1085  if spec.inputValues.currentGait < Rideable.GAITTYPES.CANTER then
1086  ParticleUtil.resetNumOfEmittedParticles(hoofInfo.psSlow)
1087  ParticleUtil.setEmittingState(hoofInfo.psSlow, true)
1088  setTranslation(hoofInfo.psSlow.emitterShape, posX, terrainY, posZ)
1089  setShaderParameter(hoofInfo.psSlow.shape, "psColor", r, g, b, 1, false)
1090  else
1091  ParticleUtil.resetNumOfEmittedParticles(hoofInfo.psFast)
1092  ParticleUtil.setEmittingState(hoofInfo.psFast, true)
1093  setTranslation(hoofInfo.psFast.emitterShape, posX, terrainY, posZ)
1094  setShaderParameter(hoofInfo.psFast.shape, "psColor", r, g, b, 1, false)
1095  end
1096
1097  local sample = self:getHoofSurfaceSound(posX, posY, posZ, hitTerrain)
1098  if sample ~= nil then
1099  hoofInfo.sampleDebug = string.format("%s - %s", sample.sampleName, sample.filename)
1100  g_soundManager:playSample(sample)
1101  end
1102
1103  elseif not onGround and hoofInfo.onGround then
1104  hoofInfo.onGround = false
1105  ParticleUtil.setEmittingState(hoofInfo.psSlow, false)
1106  ParticleUtil.setEmittingState(hoofInfo.psFast, false)
1107  end
1108  end
1109  end
```

updateDirt

Description

Definition

updateDirt()

Code

```
1113  function Rideable:updateDirt(dt)
1114  local spec = self.spec_rideable
```

```
1115
1116  if spec.dirtChangedCallbackFunc ~= nil and
      spec.inputValues.currentGait ~= Rideable.GAITTYPES.STILL then
1117  local delta = dt / (10 * 60 * 1000)
1118  local newScale = MathUtil.clamp(spec.currentDirtScale + delta,
      0.0, 1.0)
1119  self:setDirtScale(newScale)
1120  spec.dirtChangedCallbackFunc(spec.dirtChangedCallbackTarget,
      newScale)
1121  end
1122  end
```

updateFitness

Description

Definition

updateFitness()

Code

```
1126  function Rideable:updateFitness(dt)
1127  local spec = self.spec_rideable
1128
1129  if spec.fitnessChangedCallbackFunc ~= nil and spec.currentSpeed ~=
      0.0 then
1130  spec.fitnessChangedCallbackFunc(spec.fitnessChangedCallbackTarget,
      dt)
1131  end
1132  end
```

getHoofSurfaceSound

Description

Definition

getHoofSurfaceSound()

Code

```
1136  function Rideable:getHoofSurfaceSound(x, y, z, hitTerrain)
1137  local spec = self.spec_rideable
1138  local densityBits =
      getDensityAtWorldPos(g_currentMission.terrainDetailId, x, y, z)
1139  local hitTerrain = hitTerrain
1140
1141  local inWater = g_currentMission.waterY > y
1142
1143  if hitTerrain then
1144  local isOnField = densityBits ~= 0
1145  if isOnField then
1146  return spec.surfaceNameToSound["field"]
1147  elseif inWater then
```

```
1148    return spec.surfaceNameToSound["shallowWater"]
1149    else
1150    local _, _, _, _ , materialId = getTerrainAttributesAtWorldPos(g_currentMission.terrainRootNode, x, y, z, true, true, true, true, false)
1151    return spec.surfaceIdToSound[materialId]
1152    end
1153    else
1154    return spec.surfaceNameToSound["asphalt"]
1155    end
1156    end
```

getPosition

Description

Definition

getPosition()

Code

```
1160    function Rideable:getPosition()
1161    return getWorldTranslation(self.rootNode)
1162    end
```

getRotation

Description

Definition

getRotation()

Code

```
1166    function Rideable:getRotation()
1167    return getWorldRotation(self.rootNode)
1168    end
```

setDirtScale

Description

Definition

setDirtScale()

Code

```
1172    function Rideable:setDirtScale(scale)
1173    local spec = self.spec_rideable
1174    spec.currentDirtScale = scale
1175    local x, _, z, w = getShaderParameter(spec.meshNode, "RDT")
1176    I3DUtil.setShaderParameterRec(spec.meshNode, "RDT", x, spec.currentDirtScale, z, w)
1177    I3DUtil.setShaderParameterRec(spec.hairNode, "RDT", x, spec.currentDirtScale, z, w)
1178    end
```

getDirtScale

Description

Definition

getDirtScale()

Code

```
1182  function Rideable:getDirtScale()
1183  local spec = self.spec_rideable
1184  return spec.currentDirtScale
1185  end
```

setDirtChangedCallback

Description

Definition

setDirtChangedCallback()

Code

```
1189  function Rideable:setDirtChangedCallback(dirtChangedCallbackFunc, dirtChangedCallbackTarget)
1190  local spec = self.spec_rideable
1191  spec.dirtChangedCallbackFunc = dirtChangedCallbackFunc
1192  spec.dirtChangedCallbackTarget = dirtChangedCallbackTarget
1193  end
```

setFitnessChangedCallback

Description

Definition

setFitnessChangedCallback()

Code

```
1197  function Rideable:setFitnessChangedCallback(fitnessChangedCallbackFunc, fitnessChangedCallbackTarget)
1198  local spec = self.spec_rideable
1199  spec.fitnessChangedCallbackFunc = fitnessChangedCallbackFunc
1200  spec.fitnessChangedCallbackTarget = fitnessChangedCallbackTarget
1201  end
```

isOnHusbandyGround

Description

Definition

isOnHusbandyGround()

Code

```
1205  function Rideable:isOnHusbandyGround(deliveryArea)
1206  if deliveryArea ~= nil and deliveryArea.startNode ~= nil and deliveryArea.heightNode ~= nil and deliveryArea.heightNode ~= nil then
1207  local x, y, z = getWorldTranslation(self.rootNode)
1208  local xl, _, zl = worldToLocal(deliveryArea.startNode, x, y, z)
1209  local xw, _, _ = getTranslation(deliveryArea.widthNode)
1210  local _, _, zh = getTranslation(deliveryArea.heightNode)
```

```
1211  local result = (xl >= 0.0) and (zl >= 0.0) and (xl < xw) and (zl
      < zh)
1212  return result
1213  end
1214  return false
1215  end
```

abandonCheck

Description

Definition

abandonCheck()

Code

```
1219  function Rideable:abandonCheck(dt)
1220  local spec = self.spec_rideable
1221
1222  if spec.animal == nil then
1223  return
1224  end
1225  local isOnHusbandry = spec.animal:isOnHusbandyGround()
1226  if isOnHusbandry then
1227  self:removeRideable()
1228  g_currentMission:addIngameNotification(FSBaseMission.INGAME_NOTIFICATIO
      string.format(g_i18n:getText("ingameNotification_horseInStable"),
      spec.animal:getName()))
1229  return
1230  end
1231
1232  -- remove animal as soon as all players are out of range
1233  local range = 250
1234  local isPlayerInRange = false
1235  for _, player in pairs(g_currentMission.players) do
1236  if player.isControlled then
1237  local distance = calcDistanceFrom(self.rootNode, player.rootNode)
1238  if distance < range then
1239  isPlayerInRange = true
1240  break
1241  end
1242  end
1243  end
1244
1245  if not isPlayerInRange then
1246  for _, enterable in pairs(g_currentMission.enterables) do
```

```
1247  if enterable.spec_enterable ~= nil and enterable.spec_enterable.isContr
      then
1248  local distance = calcDistanceFrom(self.rootNode, enterable.rootNode)
1249  if distance < range then
1250  isPlayerInRange = true
1251  break
1252  end
1253  end
1254  end
1255  end
1256
1257  if isPlayerInRange then
1258  spec.abandonTimer = spec.abandonTimerDuration
1259  else
1260  spec.abandonTimer = spec.abandonTimer - dt
1261  if spec.abandonTimer <= 0 then
1262  self:removeRideable()
1263  g_currentMission:addIngameNotification(FSBaseMission.INGAME_NOTIFICATIO
      string.format(g_i18n:getText("ingameNotification_horseInStable"),
      spec.animal:getName()))
1264  end
1265  end
1266
1267  if spec.abandonTimer > 0 then
1268  self:raiseActive()
1269  end
1270  end
```

setAnimal

Description

Definition

setAnimal()

Code

```
1274  function Rideable:setAnimal(animal)
1275  local spec = self.spec_rideable
1276  spec.animal = animal
1277
1278  if animal ~= nil then
1279  -- set dirt
1280  local x, y, z, w = getShaderParameter(spec.meshNode, "RDT")
1281  local dirt = animal:getDirtScale()
1282  setShaderParameter(spec.meshNode, "RDT", x, dirt, z, w, false)
```

```
1283
1284  -- set texture
1285  local x, y, _, _ = getShaderParameter(spec.meshNode,
      "atlasInvSizeAndOffsetUV")
1286  local numTilesU = 1 / x
1287  local numTilesV = 1 / y
1288  local subType = animal:getSubType()
1289  local tileUIndex = subType.texture.tileUIndex
1290  local tileVIndex = subType.texture.tileVIndex
1291  local tileU = tileUIndex / numTilesU
1292  local tileV = tileVIndex / numTilesV
1293
1294  setShaderParameter(spec.meshNode, "atlasInvSizeAndOffsetUV", x,
      y, tileU, tileV, false)
1295
1296  if spec.hairNode ~= nil then
1297  local x, y, _, _ = getShaderParameter(spec.hairNode,
      "atlasInvSizeAndOffsetUV")
1298  setShaderParameter(spec.hairNode, "atlasInvSizeAndOffsetUV", x,
      y, tileU, tileV, false)
1299  end
1300  end
1301  end
```

setPlayerToEnter

Description

Definition

setPlayerToEnter()

Code

```
1304  function Rideable:setPlayerToEnter(player)
1305  local spec = self.spec_rideable
1306  spec.playerToEnter = player
1307  spec.checkPlayerToEnter = true
1308  self:raiseActive()
1309  end
```

removeRideable

Description

Definition

removeRideable()

Code

```
1313  function Rideable:removeRideable()
1314  local spec = self.spec_rideable
1315  if not spec.isRideableRemoved then
```

```
1316  spec.isRideableRemoved = true
1317
1318  local husbandry = spec.animal:getOwner()
1319  husbandry:removeRideable(spec.animal:getVisualId())
1320  end
1321  end
```

getName

Description

Definition

getName()

Code

```
1325  function Rideable:getName(superFunc)
1326  local spec = self.spec_rideable
1327  return spec.animal:getName()
1328  end
```

getFullName

Description

Definition

getFullName()

Code

```
1332  function Rideable:getFullName(superFunc)
1333  return self:getName()
1334  end
```

getCanBeReset

Description

Definition

getCanBeReset()

Code

```
1338  function Rideable:getCanBeReset(superFunc)
1339  return false
1340  end
```

updateDebugValues

Description

Definition

updateDebugValues()

Code

```
1344  function Rideable:updateDebugValues(values)
1345  local spec = self.spec_rideable
1346
1347  for k, hoofInfo in pairs(spec.hooves) do
1348  table.insert(values, {name="hoof sample " .. k,
      value=hoofInfo.sampleDebug})
1349  end
```

```
1350 end
```

updateInputText

Description

Definition

updateInputText()

Code

```
1354 function Rideable:updateInputText()
1355 local spec = self.spec_rideable
1356
1357 if spec.inputValues.currentGait == Rideable.GAITTYPES.BACKWARDS
     then
1358 g_inputBinding:setActionEventText(spec.acceletateEventId,
     g_i18n:getText("action_stop"))
1359 g_inputBinding:setActionEventActive(spec.acceletateEventId, true)
1360 g_inputBinding:setActionEventTextVisibility(spec.acceletateEventId,
     true)
1361
1362 g_inputBinding:setActionEventActive(spec.brakeEventId, false)
1363 g_inputBinding:setActionEventTextVisibility(spec.brakeEventId,
     false)
1364
1365 g_inputBinding:setActionEventActive(spec.jumpEventId, false)
1366 g_inputBinding:setActionEventTextVisibility(spec.jumpEventId,
     false)
1367 elseif spec.inputValues.currentGait == Rideable.GAITTYPES.STILL
     then
1368 g_inputBinding:setActionEventText(spec.acceletateEventId,
     g_i18n:getText("action_walk"))
1369 g_inputBinding:setActionEventActive(spec.acceletateEventId, true)
1370 g_inputBinding:setActionEventTextVisibility(spec.acceletateEventId,
     true)
1371
1372 g_inputBinding:setActionEventText(spec.brakeEventId,
     g_i18n:getText("action_walkBackwards"))
1373 g_inputBinding:setActionEventActive(spec.brakeEventId, true)
1374 g_inputBinding:setActionEventTextVisibility(spec.brakeEventId,
     true)
1375
1376 g_inputBinding:setActionEventActive(spec.jumpEventId, false)
1377 g_inputBinding:setActionEventTextVisibility(spec.jumpEventId,
     false)
1378 elseif spec.inputValues.currentGait == Rideable.GAITTYPES.WALK then
1379 g_inputBinding:setActionEventText(spec.acceletateEventId,
     g_i18n:getText("action_trot"))
```

```
g_inputBinding:setActionEventActive(spec.acceletateEventId, true)
g_inputBinding:setActionEventTextVisibility(spec.acceletateEventId, true)

g_inputBinding:setActionEventText(spec.brakeEventId, g_i18n:getText("action_stop"))
g_inputBinding:setActionEventActive(spec.brakeEventId, true)
g_inputBinding:setActionEventTextVisibility(spec.brakeEventId, true)

g_inputBinding:setActionEventActive(spec.jumpEventId, false)
g_inputBinding:setActionEventTextVisibility(spec.jumpEventId, false)
elseif spec.inputValues.currentGait == Rideable.GAITTYPES.TROT then
g_inputBinding:setActionEventText(spec.acceletateEventId, g_i18n:getText("action_canter"))
g_inputBinding:setActionEventActive(spec.acceletateEventId, true)
g_inputBinding:setActionEventTextVisibility(spec.acceletateEventId, true)

g_inputBinding:setActionEventText(spec.brakeEventId, g_i18n:getText("action_walk"))
g_inputBinding:setActionEventActive(spec.brakeEventId, true)
g_inputBinding:setActionEventTextVisibility(spec.brakeEventId, true)

g_inputBinding:setActionEventActive(spec.jumpEventId, false)
g_inputBinding:setActionEventTextVisibility(spec.jumpEventId, false)
elseif spec.inputValues.currentGait == Rideable.GAITTYPES.CANTER then
g_inputBinding:setActionEventText(spec.acceletateEventId, g_i18n:getText("action_gallop"))
g_inputBinding:setActionEventActive(spec.acceletateEventId, true)
g_inputBinding:setActionEventTextVisibility(spec.acceletateEventId, true)

g_inputBinding:setActionEventText(spec.brakeEventId, g_i18n:getText("action_trot"))
g_inputBinding:setActionEventActive(spec.brakeEventId, true)
g_inputBinding:setActionEventTextVisibility(spec.brakeEventId, true)

```

```
1409    g_inputBinding:setActionEventText(spec.jumpEventId,
        g_i18n:getText("input_JUMP"))
1410    g_inputBinding:setActionEventActive(spec.jumpEventId, true)
1411    g_inputBinding:setActionEventTextVisibility(spec.jumpEventId, true)
1412    elseif spec.inputValues.currentGait == Rideable.GAITTYPES.GALLOP
        then
1413    g_inputBinding:setActionEventActive(spec.acceletateEventId, false)
1414    g_inputBinding:setActionEventTextVisibility(spec.acceletateEventId,
        false)
1415
1416    g_inputBinding:setActionEventText(spec.brakeEventId,
        g_i18n:getText("action_canter"))
1417    g_inputBinding:setActionEventActive(spec.brakeEventId, true)
1418    g_inputBinding:setActionEventTextVisibility(spec.brakeEventId,
        true)
1419
1420    g_inputBinding:setActionEventText(spec.jumpEventId,
        g_i18n:getText("input_JUMP"))
1421    g_inputBinding:setActionEventActive(spec.jumpEventId, true)
1422    g_inputBinding:setActionEventTextVisibility(spec.jumpEventId, true)
1423    end
1424    end
```

RidgeMarker

Description

This is the specialization for tools which have a ridge marker

initSpecialization

Description

Called on specialization initializing

Definition

initSpecialization()

Code

```
20  function RidgeMarker.initSpecialization()
21  g_workAreaTypeManager:addWorkAreaType("ridgemarker", false)
22  end
```

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
28  function RidgeMarker.prerequisitesPresent(specializations)
29  return SpecializationUtil.hasSpecialization(AnimatedVehicle,
    specializations) and
    SpecializationUtil.hasSpecialization(WorkArea, specializations)
30  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
61  function RidgeMarker:onLoad(savegame)
62  local spec = self.spec_ridgeMarker
63
64  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.ridgeMarkers",
    "vehicle.ridgeMarker") -- FS17 to FS19
65  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.ridgeMarkers.ridgeMarker",
    "vehicle.ridgeMarker.marker") -- FS17 to FS19
66  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.ridgeMarker.ridgeMarker",
    "vehicle.ridgeMarker.marker") -- FS17 to FS19
67
68  local inputButtonStr = getXMLString(self.xmlFile,
    "vehicle.ridgeMarker#inputButton")
69  if inputButtonStr ~= nil then
70  spec.ridgeMarkerInputButton = InputAction[inputButtonStr]
71  end
72  spec.ridgeMarkerInputButton =
    Utils.getNoNil(spec.ridgeMarkerInputButton,
    InputAction.IMPLEMENT_EXTRA4)
73
74  spec.ridgeMarkers = {}
75  spec.workAreaToRidgeMarker = {}
76  local i = 0
77  while true do
78  local key = string.format("vehicle.ridgeMarker.marker(%d)", i)
79  if not hasXMLProperty(self.xmlFile, key) then
80  break
81  end
```

```
82
83  if table.getn(spec.ridgeMarkers) >= RidgeMarker.MAX_NUM_RIDGEMARKERS-1 then
84  g_logManager:xmlError(self.configFileName, "Too many ridgeMarker states. Only %d states are supported!", RidgeMarker.MAX_NUM_RIDGEMARKERS-1)
85  break
86  end
87
88  local ridgeMarker = {}
89  if self:loadRidgeMarker(self.xmlFile, key, ridgeMarker) then
90  table.insert(spec.ridgeMarkers, ridgeMarker)
91  spec.workAreaToRidgeMarker[ridgeMarker.workAreaIndex] = ridgeMarker
92  end
93
94  i = i + 1
95  end
96  spec.numRigdeMarkers = table.getn(spec.ridgeMarkers)
97
98  spec.ridgeMarkerMinFoldTime = Utils.getNoNil(getXMLFloat(self.xmlFile, "vehicle.ridgeMarker#foldMinLimit"), 0)
99  spec.ridgeMarkerMaxFoldTime = Utils.getNoNil(getXMLFloat(self.xmlFile, "vehicle.ridgeMarker#foldMaxLimit"), 1)
100 spec.foldDisableDirection = getXMLInt(self.xmlFile, "vehicle.ridgeMarker#foldDisableDirection")
101 spec.onlyActiveWhenLowered = Utils.getNoNil(getXMLBool(self.xmlFile, "vehicle.ridgeMarker#onlyActiveWhenLowered"), true)
102 spec.ridgeMarkerState = 0
103 spec.directionNode = I3DUtil.indexToObject(self.components, getXMLString(self.xmlFile, "vehicle.ridgeMarker#directionNode"), self.i3dMappings)
104 end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
130  function RidgeMarker:onReadStream(streamId, connection)
131  local spec = self.spec_ridgeMarker
132  if spec.numRigdeMarkers > 0 then
133  local state = streamReadUIntN(streamId,
     RidgeMarker.SEND_NUM_BITS)
134  self:setRidgeMarkerState(state, true)
135  if state ~= 0 then
136  AnimatedVehicle.updateAnimationByName(self,
     spec.ridgeMarkers[state].animName, 9999999)
137  end
138  end
139  end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
145  function RidgeMarker:onWriteStream(streamId, connection)
146  local spec = self.spec_ridgeMarker
147  if spec.numRigdeMarkers > 0 then
148  streamWriteUIntN(streamId, spec.ridgeMarkerState,
     RidgeMarker.SEND_NUM_BITS)
149  end
150  end
```

setRidgeMarkerState

Description

Toggle ridge marker state

Definition

setRidgeMarkerState(integer state, boolean noEventSend)

Arguments

integer state new state

boolean noEventSend no event send

Code

```
178  function RidgeMarker:setRidgeMarkerState(state, noEventSend)
179  local spec = self.spec_ridgeMarker
180
181  if spec.ridgeMarkerState ~= state then
```

```
182 RidgeMarkerSetStateEvent.sendEvent(self, state, noEventSend)
183
184 -- fold last state
185 if spec.ridgeMarkerState ~= 0 then
186 local animTime =
    self:getAnimationTime(spec.ridgeMarkers[spec.ridgeMarkerState].animName)
187 self:playAnimation(spec.ridgeMarkers[spec.ridgeMarkerState].animName, -1
    animTime, true)
188 end
189 spec.ridgeMarkerState = state
190
191 -- unfold new state
192 if spec.ridgeMarkerState ~= 0 then
193 if spec.ridgeMarkers[spec.ridgeMarkerState].liftedAnimTime ~= nil and no
    self:getIsLowered(true) then
194 self:setAnimationStopTime(spec.ridgeMarkers[spec.ridgeMarkerState].animN
    spec.ridgeMarkers[spec.ridgeMarkerState].liftedAnimTime)
195 end
196
197 local animTime =
    self:getAnimationTime(spec.ridgeMarkers[spec.ridgeMarkerState].animName)
198 self:playAnimation(spec.ridgeMarkers[spec.ridgeMarkerState].animName, 1,
    animTime, true)
199 end
200 end
201 end
```

canFoldRidgeMarker

Description

Return if ridge markers can be folden

Definition

canFoldRidgeMarker(integer state)

Arguments

integer state state

Return Values

boolean canFold ridge markers can be folden

Code

```
208 function RidgeMarker:canFoldRidgeMarker(state)
209 local spec = self.spec_ridgeMarker
210
211 local foldAnimTime = nil
212 if self.getFoldAnimTime ~= nil then
213 foldAnimTime = self:getFoldAnimTime()
```

```
214 if foldAnimTime < spec.ridgeMarkerMinFoldTime or foldAnimTime > spec.ridgeMarkerMaxFoldTime then
215 return false
216 end
217 end
218
219 local foldableSpec = self.spec_foldable
220 if state ~= 0 and not foldableSpec.moveToMiddle and spec.foldDisableDirection ~= nil and (spec.foldDisableDirection == foldableSpec.foldMoveDirection or foldableSpec.foldMoveDirection == 0) then
221 return false
222 end
223
224 return true
225 end
```

loadWorkAreaFromXML

Description

Loads work areas from xml

Definition

loadWorkAreaFromXML(table workArea, integer xmlFile, string key)

Arguments

table	workArea	workArea
integer	xmlFile	id of xml object
string	key	key

Return Values

boolean success success

Code

```
274 function RidgeMarker:loadWorkAreaFromXML(superFunc, workArea, xmlFile, key)
275 if not superFunc(self, workArea, xmlFile, key) then
276 return false
277 end
278
279 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, key ..".area#startIndexTest", key ..".testArea#startNode") -- FS17 to FS19
280 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, key ..".area#widthIndexTest", key ..".testArea#widthNode") -- FS17 to FS19
281 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, key ..".area#heightIndexTest", key ..".testArea#heightNode") -- FS17 to FS19
282
```

```
283 local startTest = I3DUtil.indexToObject(self.components,
    getXMLString(xmlFile, key ..".testArea#startNode"),
    self.i3dMappings)
284 local widthTest = I3DUtil.indexToObject(self.components,
    getXMLString(xmlFile, key ..".testArea#widthNode"),
    self.i3dMappings)
285 local heightTest = I3DUtil.indexToObject(self.components,
    getXMLString(xmlFile, key ..".testArea#heightNode"),
    self.i3dMappings)
286
287 if startTest ~= nil and widthTest ~= nil and heightTest ~= nil
    then
288 workArea.startTest = startTest
289 workArea.widthTest = widthTest
290 workArea.heightTest = heightTest
291
292 XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, key .."#animName") -- FS17 to FS19
293
294 if workArea.type == WorkAreaType.DEFAULT then
295 workArea.type = WorkAreaType.RIDGEMARKER
296 end
297 elseif workArea.type == WorkAreaType.RIDGEMARKER then
298 g_logManager:xmlWarning(self.configFileName, "Missing test area
    for ridge marker area '%s'", key)
299 end
300
301 return true
302 end
```

loadSpeedRotatingPartFromXML

Description

Loads speed rotating parts from xml

Definition

loadSpeedRotatingPartFromXML(table speedRotatingPart, integer xmlFile, string key)

Arguments

table	speedRotatingPart	speedRotatingPart
integer	xmlFile	id of xml object
string	key	key

Return Values

boolean	success	success

Code

```
310 function RidgeMarker:loadSpeedRotatingPartFromXML(superFunc,
    speedRotatingPart, xmlFile, key)
311 if not superFunc(self, speedRotatingPart, xmlFile, key) then
```

```
312 return false
313 end
314
315 speedRotatingPart.ridgeMarkerAnim = getXMLString(xmlFile, key ..
    "#ridgeMarkerAnim")
316 speedRotatingPart.ridgeMarkerAnimTimeMax =
    Utils.getNoNil(getXMLFloat(xmlFile, key ..
    "#ridgeMarkerAnimTimeMax"), 0.99)
317
318 return true
319 end
```

getIsSpeedRotatingPartActive

Description

Returns true if speed rotating part is active

Definition

getIsSpeedRotatingPartActive(table speedRotatingPart)

Arguments

table speedRotatingPart speedRotatingPart

Return Values

boolean isActive speed rotating part is active

Code

```
325 function RidgeMarker:getIsSpeedRotatingPartActive(superFunc,
    speedRotatingPart)
326 if speedRotatingPart.ridgeMarkerAnim ~= nil and
    self:getAnimationTime(speedRotatingPart.ridgeMarkerAnim) <
    speedRotatingPart.ridgeMarkerAnimTimeMax then
327 return false
328 end
329
330 return superFunc(self, speedRotatingPart)
331 end
```

getIsWorkAreaActive

Description

Returns true if work area is active

Definition

getIsWorkAreaActive(table workArea)

Arguments

table workArea workArea

Return Values

boolean isActive work area is active

Code

```
341 function RidgeMarker:getIsWorkAreaActive(superFunc, workArea)
342
```

```lua
if workArea.type == WorkAreaType.RIDGEMARKER then
local spec = self.spec_ridgeMarker
local ridgeMarker = spec.workAreaToRidgeMarker[workArea.index]

if ridgeMarker ~= nil then
local animTime = self:getAnimationTime(ridgeMarker.animName)
if animTime > ridgeMarker.maxWorkLimit or animTime < ridgeMarker.minWorkLimit then
return false
end

if spec.onlyActiveWhenLowered and not self:getIsLowered(false) then
return false
end
end
end

return superFunc(self, workArea)
end
```

onSetLowered

Description

Called on lowered state change

Definition

onSetLowered(boolean lowered)

Arguments

boolean lowered is lowered

Code

```lua
function RidgeMarker:onSetLowered(lowered)
local spec = self.spec_ridgeMarker
if lowered then
for _, ridgeMarker in pairs(spec.ridgeMarkers) do
if ridgeMarker.liftedAnimTime ~= nil then
local animTime = self:getAnimationTime(ridgeMarker.animName)
if animTime == ridgeMarker.liftedAnimTime then
self:playAnimation(ridgeMarker.animName, 1, animTime, true)
end
end
end
else
for _, ridgeMarker in pairs(spec.ridgeMarkers) do
```

```lua
if ridgeMarker.liftedAnimTime ~= nil then
local animTime = self:getAnimationTime(ridgeMarker.animName)
if animTime > ridgeMarker.liftedAnimTime then
self:setAnimationStopTime(ridgeMarker.animName, ridgeMarker.liftedAnimTime)
self:playAnimation(ridgeMarker.animName, -1, animTime, true)
end
end
end
end
```

onFoldStateChanged

Description

Called on folding state change

Definition

onFoldStateChanged(integer direction, boolean moveToMiddle)

Arguments

integer direction direction

boolean moveToMiddle move to middle position

Code

```lua
function RidgeMarker:onFoldStateChanged(direction, moveToMiddle)
if not moveToMiddle and direction > 0 then
self:setRidgeMarkerState(0, true)
end
end
```

Roller

Description

getWearMultiplier

Description

Returns current wear multiplier

Definition

getWearMultiplier()

Return Values

float dirtMultiplier current wear multiplier

Code

```lua
function Roller:getWearMultiplier(superFunc)
local spec = self.spec_roller
local multiplier = superFunc(self)

if spec.isWorking then
multiplier = multiplier + self:getWorkWearMultiplier()
end
```

```
113
114 return multiplier
115 end
```

onPostAttach

Description

Called after vehicle gets attached

Definition

onPostAttach(table attacherVehicle, integer inputJointDescIndex, integer jointDescIndex)

Arguments

table	attacherVehicle	attacher vehicle
integer	inputJointDescIndex	index of input attacher joint
integer	jointDescIndex	index of attacher joint it gets attached to

Code

```
161 function Roller:onPostAttach(attacherVehicle,
    inputJointDescIndex, jointDescIndex)
162 local spec = self.spec_roller
163 spec.startActivationTime = g_currentMission.time +
    spec.startActivationTimeout
164 end
```

Ropes

Description

Specialization for vehicles with animated ropes

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17 function Ropes.prerequisitesPresent(specializations)
18 return true
19 end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
function Ropes:onLoad(savegame)
local spec = self.spec_ropes

spec.ropes = {}
local i = 0
while true do
local key = string.format("vehicle.ropes.rope(%d)", i)
if not hasXMLProperty(self.xmlFile, key) then
break
end

local entry = {}
entry.baseNode = I3DUtil.indexToObject(self.components, getXMLString(self.xmlFile, key .. "#baseNode"), self.i3dMappings)
entry.targetNode = I3DUtil.indexToObject(self.components, getXMLString(self.xmlFile, key .. "#targetNode"), self.i3dMappings)
entry.baseParameters = StringUtil.getVectorNFromString(getXMLString(self.xmlFile, key .. "#baseParameters", 4))
entry.targetParameters = StringUtil.getVectorNFromString(getXMLString(self.xmlFile, key .. "#targetParameters", 4))

setShaderParameter(entry.baseNode, "cv0", entry.baseParameters[1], entry.baseParameters[2], entry.baseParameters[3], entry.baseParameters[4], false)
setShaderParameter(entry.baseNode, "cv1", 0, 0, 0, 0, false)
local x,y,z = localToLocal(entry.targetNode, entry.baseNode, entry.targetParameters[1], entry.targetParameters[2], entry.targetParameters[3])
setShaderParameter(entry.baseNode, "cv3", x, y, z, 0, false)

entry.baseParameterAdjusters = {}
local j = 0
while true do
local adjusterKey = string.format("%s.baseParameterAdjuster(%d)", key, j)
if not hasXMLProperty(self.xmlFile, adjusterKey) then
break
end

local adjusterNode = {}
```

```
65  if self:loadAdjusterNode(adjusterNode, self.xmlFile,
    adjusterKey)then
66  table.insert(entry.baseParameterAdjusters, adjusterNode)
67  end
68
69  j = j + 1
70  end
71
72  entry.targetParameterAdjusters = {}
73  j = 0
74  while true do
75  local adjusterKey =
    string.format("%s.targetParameterAdjuster(%d)", key, j)
76  if not hasXMLProperty(self.xmlFile, adjusterKey) then
77  break
78  end
79
80  local adjusterNode = {}
81  if self:loadAdjusterNode(adjusterNode, self.xmlFile,
    adjusterKey)then
82  table.insert(entry.targetParameterAdjusters, adjusterNode)
83  end
84
85  j = j + 1
86  end
87
88  table.insert(spec.ropes, entry)
89  i = i + 1
90  end
91
92  spec.isActiveTimeOffset = Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.ropes#isActiveDirtyTimeOffset"), 2) * 1000
93  spec.isActiveTime = g_currentMission.time +
    spec.isActiveTimeOffset
94  end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms

boolean isActiveForInput true if vehicle is active for input
boolean isSelected true if vehicle is selected

Code

```
101 function Ropes:onUpdate(dt, isActiveForInput, isSelected)
102 local spec = self.spec_ropes
103 if self.isClient then
104 spec.isActiveTime = g_currentMission.time +
    spec.isActiveTimeOffset
105 end
106 if spec.isActiveTime >= g_currentMission.time then
107 for _, rope in pairs(spec.ropes) do
108 local x, y, z =
    self:updateAdjusterNodes(rope.baseParameterAdjusters)
109 setShaderParameter(rope.baseNode, "cv0",
    rope.baseParameters[1]+x, rope.baseParameters[2]+y,
    rope.baseParameters[3]+z, 0, false)
110
111 x, y, z = localToLocal(rope.targetNode, rope.baseNode, 0, 0, 0)
112 setShaderParameter(rope.baseNode, "cv2", 0, 0, 0, 0, false)
113 setShaderParameter(rope.baseNode, "cv3", x, y, z, 0, false)
114
115 x, y, z = self:updateAdjusterNodes(rope.targetParameterAdjusters)
116 x, y, z = localToLocal(rope.targetNode, rope.baseNode,
    rope.targetParameters[1]+x, rope.targetParameters[2]+y,
    rope.targetParameters[3]+z)
117 setShaderParameter(rope.baseNode, "cv4", x,y,z,0, false)
118 end
119 end
120 end
```

SemiTrailerFront

Description

This is the specialization for semi trailer fronts

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17 function SemiTrailerFront.prerequisitesPresent(specializations)
```

```
18  return SpecializationUtil.hasSpecialization(AttacherJoints,
    specializations)
19  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
37  function SemiTrailerFront:onLoad(savegame)
38  local spec = self.spec_semiTrailerFront
39  spec.inputAttacherCurFade = 1
40  spec.inputAttacherFadeDir = 1
41  spec.inputAttacherFadeDuration = 1000
42
43  spec.joint = self.spec_attachable.inputAttacherJoints[1]
44  spec.joint.lowerRotLimitScaleBackup =
    {spec.joint.lowerRotLimitScale[1],
    spec.joint.lowerRotLimitScale[2],
    spec.joint.lowerRotLimitScale[3]}
45
46  spec.attachedSemiTrailerBack = nil
47  spec.inputAttacherImplement = nil
48  spec.doSemiTrailerLockCheck = true
49  end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
56  function SemiTrailerFront:onUpdate(dt, isActiveForInput, isSelected)
57  local spec = self.spec_semiTrailerFront
58
59  if spec.doSemiTrailerLockCheck then
60  spec.doSemiTrailerLockCheck = false
```

```
61  if spec.attachedSemiTrailerBack == nil then
62  spec.inputAttacherFadeDir = -1
63  end
64  end
65
66  if self.isServer and spec.inputAttacherImplement ~= nil and ((spec.inputA
    spec.inputAttacherFadeDir < 0) or (spec.inputAttacherCurFade < 1 and spec
    then
67  spec.inputAttacherCurFade =
    MathUtil.clamp(spec.inputAttacherCurFade+(spec.inputAttacherFadeDir*dt)/s
    0, 1)
68
69  local lowerRotLimitScale = spec.joint.lowerRotLimitScale
70  local lowerRotLimitScaleBackup = spec.joint.lowerRotLimitScaleBackup
71  lowerRotLimitScale[1] = lowerRotLimitScaleBackup[1]*spec.inputAttacherCur
72  lowerRotLimitScale[2] = lowerRotLimitScaleBackup[2]*spec.inputAttacherCur
73  lowerRotLimitScale[3] = lowerRotLimitScaleBackup[3]*spec.inputAttacherCur
74
75  local attacherVehicle = self:getAttacherVehicle()
76  if attacherVehicle ~= nil then
77  local attacherJoints = attacherVehicle:getAttacherJoints()
78  local jointDesc = attacherJoints[spec.inputAttacherImplement.jointDescInd
79  local lowerRotLimit = spec.inputAttacherImplement.lowerRotLimit
80  local x = lowerRotLimit[1]*lowerRotLimitScale[1]
81  local y = lowerRotLimit[2]*lowerRotLimitScale[2]
82  local z = lowerRotLimit[3]*lowerRotLimitScale[3]
83
84  setJointRotationLimit(jointDesc.jointIndex, 0, true, -x, x)
85  setJointRotationLimit(jointDesc.jointIndex, 1, true, -y, y)
86  setJointRotationLimit(jointDesc.jointIndex, 2, true, -z, z)
87  end
88  end
89  end
```

isDetachAllowed

Description

Returns true if detach is allowed

Definition

isDetachAllowed()

Return Values

boolean detachAllowed detach is allowed

Code

```
94  function SemiTrailerFront:isDetachAllowed(superFunc)
```

```lua
local canBeDatached, warning = superFunc(self)
if not canBeDatached then
return false, warning
end

local spec = self.spec_semiTrailerFront
return spec.attachedSemiTrailerBack ~= nil, nil
end
```

onPostAttachImplement

Description

Called on attaching a implement

Definition

onPostAttachImplement(table implement)

Arguments

table implement implement to attach

Code

```lua
function SemiTrailerFront:onPostAttachImplement(attachable, inputJointDescIndex, jointDescIndex)
local spec = self.spec_semiTrailerFront
spec.attachedSemiTrailerBack = attachable
spec.inputAttacherFadeDir = 1000
end
```

onPreDetachImplement

Description

Called on detaching a implement

Definition

onPreDetachImplement(integer implementIndex)

Arguments

integer implementIndex index of implement to detach

Code

```lua
function SemiTrailerFront:onPreDetachImplement(implement)
local spec = self.spec_semiTrailerFront
spec.attachedSemiTrailerBack = nil
spec.inputAttacherFadeDir = -1
end
```

onPreDetach

Description

Called if vehicle gets detached

Definition

onPreDetach(table attacherVehicle, table implement)

Arguments

table attacherVehicle attacher vehicle

table	implement	implement

Code

```
126  function SemiTrailerFront:onPreDetach(attacherVehicle, implement)
127  local spec = self.spec_semiTrailerFront
128  spec.inputAttacherImplement = nil
129  end
```

onPostAttach

Description

Called after vehicle was attached

Definition

onPostAttach(table attacherVehicle, integer inputJointDescIndex, integer jointDescIndex)

Arguments

table	attacherVehicle	attacher vehicle
integer	inputJointDescIndex	index of input attacher joint
integer	jointDescIndex	index of attacher joint it was attached to

Code

```
136  function SemiTrailerFront:onPostAttach(attacherVehicle,
     inputJointDescIndex, jointDescIndex)
137  local spec = self.spec_semiTrailerFront
138  spec.inputAttacherImplement =
     attacherVehicle:getImplementByObject(self)
139  end
```

Shovel

Description

getWearMultiplier

Description

Returns current wear multiplier

Definition

getWearMultiplier()

Return Values

float	wearMultiplier current wear multiplier

Code

```
315  function Shovel:getWearMultiplier(superFunc)
316  local spec = self.spec_shovel
317  local multiplier = superFunc(self)
318
319  if spec.loadingFillType ~= FillType.UNKNOWN then
320  multiplier = multiplier + self:getWorkWearMultiplier()
321  end
322
323  return multiplier
324  end
```

SowingMachine

Description

getWearMultiplier

Description

Returns current wear multiplier

Definition

getWearMultiplier()

Return Values

float dirtMultiplier current wear multiplier

Code

```
442 function SowingMachine:getWearMultiplier(superFunc)
443 local spec = self.spec_sowingMachine
444 local multiplier = superFunc(self)
445
446 if self.movingDirection > 0 and spec.isWorking and (not
    spec.needsActivation or self:getIsTurnedOn()) then
447 multiplier = multiplier + self:getWorkWearMultiplier() *
    self:getLastSpeed() / self.speedLimit
448 end
449
450 return multiplier
451 end
```

SpeedRotatingParts

Description

Specialization for vehicle with speedrotatingparts

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17 function SpeedRotatingParts.prerequisitesPresent(specializations)
18 return true
19 end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
44  function SpeedRotatingParts:onLoad(savegame)
45  local spec = self.spec_speedRotatingParts
46
47  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.speedRotatingParts.speedRotatingPart(0)#index",
    "vehicle.speedRotatingParts.speedRotatingPart(0)#node") -- FS17
48
49  spec.speedRotatingParts = {}
50  local i = 0
51  while true do
52  local baseName =
    string.format("vehicle.speedRotatingParts.speedRotatingPart(%d)",
    i)
53  if not hasXMLProperty(self.xmlFile, baseName) then
54  break
55  end
56  local speedRotatingPart = {}
57  if self:loadSpeedRotatingPartFromXML(speedRotatingPart,
    self.xmlFile, baseName) then
58  table.insert(spec.speedRotatingParts, speedRotatingPart)
59  end
60  i = i + 1
61  end
62
63  spec.dirtyFlag = self:getNextDirtyFlag()
64  end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
70  function SpeedRotatingParts:onReadStream(streamId, connection)
71  local spec = self.spec_speedRotatingParts
72  for _, speedRotatingPart in pairs(spec.speedRotatingParts) do
73  if speedRotatingPart.versatileYRot then
74  local yRot = streamReadUIntN(streamId, 9)
```

```
75  speedRotatingPart.steeringAngle = yRot / 511 * math.pi*2
76  end
77  end
78  end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId	streamId
integer connection	connection

Code

```
84  function SpeedRotatingParts:onWriteStream(streamId, connection)
85  local spec = self.spec_speedRotatingParts
86  for _, speedRotatingPart in pairs(spec.speedRotatingParts) do
87  if speedRotatingPart.versatileYRot then
88  streamWriteUIntN(streamId,
    MathUtil.clamp(math.floor(speedRotatingPart.steeringAngle /
    (math.pi*2) * 511), 0, 511), 9)
89  end
90  end
91  end
```

onReadUpdateStream

Description

Called on on update

Definition

onReadUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId	stream ID
integer timestamp	timestamp
table	connection connection

Code

```
98  function SpeedRotatingParts:onReadUpdateStream(streamId,
    timestamp, connection)
99  if connection.isServer then
100 local spec = self.spec_speedRotatingParts
101 local hasUpdate = streamReadBool(streamId)
102 if hasUpdate then
103 for _, speedRotatingPart in pairs(spec.speedRotatingParts) do
104 if speedRotatingPart.versatileYRot then
105 local yRot = streamReadUIntN(streamId, 9)
```

```
106     speedRotatingPart.steeringAngle = yRot / 511 * math.pi*2
107 end
108 end
109 end
110 end
111 end
```

onWriteUpdateStream

Description

Called on on update

Definition

onWriteUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer	streamId	stream ID
table	connection	connection
integer	dirtyMask	dirty mask

Code

```
118 function SpeedRotatingParts:onWriteUpdateStream(streamId,
    connection, dirtyMask)
119     if not connection.isServer then
120         local spec = self.spec_speedRotatingParts
121         if streamWriteBool(streamId, bitAND(dirtyMask, spec.dirtyFlag) ~=
    0) then
122             for _, speedRotatingPart in pairs(spec.speedRotatingParts) do
123                 if speedRotatingPart.versatileYRot then
124                     local yRot = speedRotatingPart.steeringAngle % (math.pi*2)
125                     streamWriteUIntN(streamId, MathUtil.clamp(math.floor(yRot /
    (math.pi*2) * 511), 0, 511), 9)
126                 end
127             end
128         end
129     end
130 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
137 function SpeedRotatingParts:onUpdate(dt, isActiveForInput,
    isSelected)
138 local spec = self.spec_speedRotatingParts
139
140 for _, speedRotatingPart in pairs(spec.speedRotatingParts) do
141 local isPartActive =
    self:getIsSpeedRotatingPartActive(speedRotatingPart)
142 if isPartActive or (speedRotatingPart.lastSpeed ~= 0 and not
    speedRotatingPart.stopIfNotActive) then
143 self:updateSpeedRotatingPart(speedRotatingPart, dt, isPartActive)
144 end
145 end
146 end
```

loadSpeedRotatingPartFromXML

Description

Loads speed rotating parts from xml

Definition

loadSpeedRotatingPartFromXML(table speedRotatingPart, integer xmlFile, string key)

Arguments

table	speedRotatingPart	speedRotatingPart
integer	xmlFile	id of xml object
string	key	key

Return Values

boolean	success	success

Code

```
154 function
    SpeedRotatingParts:loadSpeedRotatingPartFromXML(speedRotatingPart,
    xmlFile, key)
155 speedRotatingPart.repr = I3DUtil.indexToObject(self.components,
    getXMLString(xmlFile, key.."#node"), self.i3dMappings)
156 speedRotatingPart.shaderNode =
    I3DUtil.indexToObject(self.components, getXMLString(xmlFile,
    key.."#shaderNode"), self.i3dMappings)
157 if speedRotatingPart.shaderNode ~= nil then
158 speedRotatingPart.useShaderRotation =
    Utils.getNoNil(getXMLBool(xmlFile, key.."#useRotation"), true)
159 local scale = Utils.getNoNil(getXMLString(xmlFile,
    key.."#scrollScale"), "1 0")
160 speedRotatingPart.scrollScale =
    StringUtil.getVectorNFromString(scale, 2)
161 end
162
163 if speedRotatingPart.repr == nil and speedRotatingPart.shaderNode
    == nil then
```

```
164 g_logManager:xmlWarning(self.configFileName, "Invalid
    speedRotationPart node '%s' in '%s'",
    tostring(getXMLString(xmlFile, key.."#node")), key)
165 return false
166 end
167 speedRotatingPart.driveNode =
    Utils.getNoNil(I3DUtil.indexToObject(self.components,
    getXMLString(xmlFile, key .. "#driveNode"), self.i3dMappings),
    speedRotatingPart.repr)
168
169 local componentIndex = getXMLInt(xmlFile,
    key.."#refComponentIndex")
170 if componentIndex ~= nil and self.components[componentIndex] ~=
    nil then
171 speedRotatingPart.componentNode =
    self.components[componentIndex].node
172 else
173 local node = Utils.getNoNil(speedRotatingPart.driveNode,
    speedRotatingPart.shaderNode)
174 speedRotatingPart.componentNode = self:getParentComponent(node)
175 end
176
177 speedRotatingPart.xDrive = 0
178 local wheelIndex = getXMLInt(xmlFile, key.."#wheelIndex")
179 if wheelIndex ~= nil then
180 if self.getWheels == nil then
181 g_logManager:xmlWarning(self.configFileName, "wheelIndex for
    speedRotatingPart '%s' given, but no wheels loaded/defined", key)
182 else
183 local wheels = self:getWheels()
184 local wheel = wheels[wheelIndex]
185 if wheel == nil then
186 g_logManager:xmlWarning(self.configFileName, "Invalid wheel index
    '%s' for speedRotatingPart '%s'", tostring(wheelIndex), key)
187 return false
188 end
189 speedRotatingPart.wheel = wheel
190 speedRotatingPart.lastWheelXRot = 0
191 end
192 end
193
```

```
194 speedRotatingPart.dirRefNode =
    I3DUtil.indexToObject(self.components, getXMLString(xmlFile,
    key.."#dirRefNode"), self.i3dMappings)
195 speedRotatingPart.dirFrameNode =
    I3DUtil.indexToObject(self.components, getXMLString(xmlFile,
    key.."#dirFrameNode"), self.i3dMappings)
196 speedRotatingPart.alignDirection =
    Utils.getNoNil(getXMLBool(xmlFile, key .. "#alignDirection"),
    false)
197 speedRotatingPart.applySteeringAngle =
    Utils.getNoNil(getXMLBool(xmlFile, key .. "#applySteeringAngle"),
    false)
198 speedRotatingPart.useWheelReprTranslation =
    Utils.getNoNil(getXMLBool(xmlFile, key ..
    "#useWheelReprTranslation"), true)
199 speedRotatingPart.updateXDrive =
    Utils.getNoNil(getXMLBool(xmlFile, key .. "#updateXDrive"), true)
200
201 speedRotatingPart.versatileYRot =
    Utils.getNoNil(getXMLBool(xmlFile, key .. "#versatileYRot"),
    false)
202 if speedRotatingPart.versatileYRot and speedRotatingPart.repr ==
    nil then
203 g_logManager:xmlWarning(self.configFileName, "Versatile
    speedRotationPart '%s' does not support shaderNodes", key)
204 return false
205 end
206
207 local minYRot = getXMLFloat(xmlFile, key .. "#minYRot")
208 if minYRot ~= nil then
209 speedRotatingPart.minYRot = math.rad(minYRot)
210 end
211 local maxYRot = getXMLFloat(xmlFile, key .. "#maxYRot")
212 if maxYRot ~= nil then
213 speedRotatingPart.maxYRot = math.rad(maxYRot)
214 end
215 speedRotatingPart.steeringAngle = 0
216 speedRotatingPart.steeringAngleSent = 0
217
218 speedRotatingPart.wheelScale = getXMLFloat(xmlFile, key ..
    "#wheelScale")
219 if speedRotatingPart.wheelScale == nil then
220 local baseRadius = 1.0
221 local radius = 1.0
```

```
222 if speedRotatingPart.wheel ~= nil then
223 baseRadius = speedRotatingPart.wheel.radius
224 radius = speedRotatingPart.wheel.radius
225 end
226 speedRotatingPart.wheelScale = baseRadius /
    Utils.getNoNil(getXMLFloat(xmlFile, key.."#radius"), radius)
227 end
228
229 speedRotatingPart.wheelScaleBackup = speedRotatingPart.wheelScale
230
231 speedRotatingPart.onlyActiveWhenLowered =
    Utils.getNoNil(getXMLBool(xmlFile, key ..
    "#onlyActiveWhenLowered"), false)
232 speedRotatingPart.stopIfNotActive =
    Utils.getNoNil(getXMLBool(xmlFile, key .. "#stopIfNotActive"),
    false)
233 speedRotatingPart.fadeOutTime =
    Utils.getNoNil(getXMLFloat(xmlFile, key .. "#fadeOutTime"), 3) *
    1000
234 speedRotatingPart.activationSpeed =
    Utils.getNoNil(getXMLFloat(xmlFile, key .. "#activationSpeed"), 1)
235 speedRotatingPart.speedReferenceNode =
    I3DUtil.indexToObject(self.components, getXMLString(xmlFile,
    key.."#speedReferenceNode"), self.i3dMappings)
236 if speedRotatingPart.speedReferenceNode ~= nil and
    speedRotatingPart.speedReferenceNode ==
    speedRotatingPart.driveNode then
237 g_logManager:xmlWarning(self.configFileName, "Ignoring
    speedRotationPart '%s' because speedReferenceNode is identical
    with driveNode. Need to be different!", key)
238 return false
239 end
240 speedRotatingPart.lastSpeed = 0
241 speedRotatingPart.lastDir = 1
242
243 return true
244 end
```

getIsSpeedRotatingPartActive

Description

Returns true if speed rotating part is active

Definition

getIsSpeedRotatingPartActive(table speedRotatingPart)

Arguments

table speedRotatingPart speedRotatingPart

Return Values

boolean isActive speed rotating part is active

Code

```
250 function SpeedRotatingParts:getIsSpeedRotatingPartActive(speedRotatingPart)
251 if speedRotatingPart.onlyActiveWhenLowered then
252 if self.getIsLowered ~= nil and not self:getIsLowered() then
253 return false
254 else
255 return true
256 end
257 end
258
259 return true
260 end
```

getSpeedRotatingPartDirection

Description

Return direction of speed rotating part

Definition

getSpeedRotatingPartDirection(table speedRotatingPart)

Arguments

table speedRotatingPart speed rotating part

Return Values

integer direction direction

Code

```
266 function SpeedRotatingParts:getSpeedRotatingPartDirection(speedRotatingPart)
267 return 1
268 end
```

SplineVehicle

Description

Class for all SplineVehicles

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17 function SplineVehicle.prerequisitesPresent(specializations)
```

```
18  return true
19  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
44  function SplineVehicle:onLoad(savegame)
45  local spec = self.spec_splineVehicle
46
47  spec.frontNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile,
    "vehicle.splineVehicle.dollies#frontNode"), self.i3dMappings)
48  spec.backNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile,
    "vehicle.splineVehicle.dollies#backNode"), self.i3dMappings)
49
50  spec.frontToBackDistance = calcDistanceFrom(spec.frontNode,
    spec.backNode)
51
52  spec.dolly1Node = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile,
    "vehicle.splineVehicle.dollies#dolly1Node"), self.i3dMappings)
53  spec.dolly2Node = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile,
    "vehicle.splineVehicle.dollies#dolly2Node"), self.i3dMappings)
54
55  spec.dollyToDollyDistance = calcDistanceFrom(spec.dolly1Node,
    spec.dolly2Node)
56
57  spec.rootNodeToBackDistance = calcDistanceFrom(spec.backNode,
    self.rootNode)
58  spec.rootNodeToFrontDistance = calcDistanceFrom(spec.frontNode,
    self.rootNode)
59
60  spec.alignDollys = Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.splineVehicle.dollies#alignDollys"), true)
61  spec.splinePosition = 0
62  spec.lastSplinePosition = 0
63  spec.currentSplinePosition = 0
64  spec.lastSplinePositionDelta = 0
```

```
65
66  spec.lastSplinePositionSpeed = 0
67  spec.splinePositionSpeedReal = 0
68  spec.splinePositionSpeed = 0
69  spec.splinePositionAcceleration = 0
70
71  spec.splineSpeed = 0
72
73  spec.firstUpdate = true
74  end
```

StrawBlower

Description

Class for all sprayers

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17  function StrawBlower.prerequisitesPresent(specializations)
18  return SpecializationUtil.hasSpecialization(FillUnit,
    specializations)
19  and SpecializationUtil.hasSpecialization(Trailer, specializations)
20  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
42  function StrawBlower:onLoad(savegame)
43  local spec = self.spec_strawBlower
44
45  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.strawBlower.baleTrigger#index",
    "vehicle.strawBlower.baleTrigger#node") --FS17 to FS19
```

```
46  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.strawBlower.doorAnimation#name",
    "vehicle.foldable.foldingParts.foldingPart.animationName") --FS17
    to FS19
47
48  spec.triggeredBales = {}
49
50  if self.isServer then
51  spec.triggerId = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile,
    "vehicle.strawBlower.baleTrigger#node"), self.i3dMappings)
52  if spec.triggerId ~= nil then
53  addTrigger(spec.triggerId, "strawBlowerBaleTriggerCallback", self)
54  end
55  end
56
57  if self.isClient then
58  spec.animationNodes =
    g_animationManager:loadAnimations(self.xmlFile,
    "vehicle.strawBlower.animationNodes", self.components, self,
    self.i3dMappings)
59
60  spec.samples = {}
61  spec.samples.start =
    g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.strawBlower.sounds", "start", self.baseDirectory,
    self.components, 1, AudioGroup.VEHICLE, self.i3dMappings, self)
62  spec.samples.stop = g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.strawBlower.sounds", "stop", self.baseDirectory,
    self.components, 1, AudioGroup.VEHICLE, self.i3dMappings, self)
63  spec.samples.work = g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.strawBlower.sounds", "work", self.baseDirectory,
    self.components, 0, AudioGroup.VEHICLE, self.i3dMappings, self)
64  end
65
66  spec.fillUnitIndex = Utils.getNoNil(getXMLInt(self.xmlFile,
    "vehicle.strawBlower#fillUnitIndex"), 1)
67
68  -- we need to send the full fill level rather than percentages
    since we change the capacity
69  local fillUnit = self:getFillUnitByIndex(spec.fillUnitIndex)
70  fillUnit.synchronizeFullFillLevel = true
71
72  if savegame ~= nil and not savegame.resetVehicles then
```

```
73  self:addFillUnitFillLevel(self:getOwnerFarmId(),
    spec.fillUnitIndex, -math.huge, FillType.UNKNOWN,
    ToolType.UNDEFINED)
74  end
75  end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
79  function StrawBlower:onDelete()
80  local spec = self.spec_strawBlower
81
82  if self.isServer then
83  if spec.triggerId ~= nil then
84  removeTrigger(spec.triggerId)
85  end
86
87  for bale, _ in pairs(spec.triggeredBales) do
88  if entityExists(bale.nodeId) then
89  I3DUtil.wakeUpObject(bale.nodeId)
90  end
91  end
92  end
93
94  if self.isClient then
95  g_soundManager:deleteSamples(spec.samples)
96  g_animationManager:deleteAnimations(spec.animationNodes)
97  end
98  end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```lua
function StrawBlower:onUpdateTick(dt, isActiveForInput, isSelected)
local spec = self.spec_strawBlower

if self.isServer then
if spec.currentBale == nil and self:getFillUnitSupportsToolType(spec.fillUnitIndex, ToolType.BALE) then
local bale = next(spec.triggeredBales)
if bale ~= nil then
self:setFillUnitCapacity(spec.fillUnitIndex, bale:getFillLevel())
self:addFillUnitFillLevel(self:getOwnerFarmId(), spec.fillUnitIndex, -math.huge, FillType.UNKNOWN, ToolType.UNDEFINED)
self:addFillUnitFillLevel(self:getOwnerFarmId(), spec.fillUnitIndex, bale:getFillLevel(), bale:getFillType(), ToolType.BALE)
spec.currentBale = bale
end
end
end
end
```

strawBlowerBaleTriggerCallback

Description

Trigger callback

Definition

strawBlowerBaleTriggerCallback(integer triggerId, integer otherActorId, boolean onEnter, boolean onLeave, boolean onStay, integer otherShapeId)

Arguments

integer	triggerId	id of trigger
integer	otherActorId	id of other actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay
integer	otherShapeId	id of other shape

Code

```lua
function StrawBlower:strawBlowerBaleTriggerCallback(triggerId, otherActorId, onEnter, onLeave, onStay, otherShapeId)
local spec = self.spec_strawBlower

if onEnter then
-- this happens if a compound child of a deleted compound is entering
if otherActorId ~= 0 then
```

```
135 local object = g_currentMission:getNodeObject(otherActorId)
136 if object ~= nil then
137 if object:isa(Bale) and
    g_currentMission.accessHandler:canFarmAccess(self:getActiveFarm(),
    object) then
138 if self:getFillUnitSupportsFillType(spec.fillUnitIndex,
    object:getFillType()) then
139 spec.triggeredBales[object] =
    Utils.getNoNil(spec.triggeredBales[object], 0) + 1
140 end
141 end
142 end
143 end
144 elseif onLeave then
145 if otherActorId ~= 0 then
146 local object = g_currentMission:getNodeObject(otherActorId)
147 if object ~= nil then
148 local triggerCount = spec.triggeredBales[object]
149 if triggerCount ~= nil then
150 if triggerCount == 1 then
151 spec.triggeredBales[object] = nil
152 if object == spec.currentBale then
153 spec.currentBale = nil
154 self:addFillUnitFillLevel(self:getOwnerFarmId(),
    spec.fillUnitIndex, -math.huge, FillType.UNKNOWN,
    ToolType.UNDEFINED)
155 end
156 else
157 spec.triggeredBales[object] = triggerCount-1
158 end
159 end
160 end
161 end
162 end
163 end
```

onFillUnitFillLevelChanged

Description

Set unit fill level

Definition

onFillUnitFillLevelChanged(integer fillUnitIndex, float fillLevel, integer fillType, boolean force, table fillInfo)

Arguments

integer	fillUnitIndex	index of fill unit
float	fillLevel	new fill level
integer	fillType	fill type
boolean	force	force action
table	fillInfo	fill info for fill volume

Code

```
189 function StrawBlower:onFillUnitFillLevelChanged(fillUnitIndex, fillLevelDelta, fillTypeIndex, toolType, fillPositionData, appliedDelta)
190 local spec = self.spec_strawBlower
191 if fillUnitIndex == spec.fillUnitIndex then
192 local newFillLevel = self:getFillUnitFillLevel(spec.fillUnitIndex)
193
194 if self.isServer then
195 local bale = spec.currentBale
196 if bale ~= nil then
197 if newFillLevel <= 0 then
198 if self.removeDynamicMountedObject ~= nil then
199 self:removeDynamicMountedObject(bale)
200 end
201
202 local baleOwner = bale:getOwnerFarmId()
203
204 bale:delete()
205 spec.currentBale = nil
206 spec.triggeredBales[bale] = nil
207
208 self:setFillUnitCapacity(spec.fillUnitIndex, 1)
209 self:addFillUnitFillLevel(baleOwner, spec.fillUnitIndex, -math.huge, FillType.UNKNOWN, ToolType.UNDEFINED)
210 elseif newFillLevel < bale:getFillLevel() and fillTypeIndex == bale:getFillType() then
211 bale:setFillLevel(newFillLevel)
212 end
213 end
214 else
215 if newFillLevel <= 0 then
216 self:setFillUnitCapacity(spec.fillUnitIndex, 1)
217 end
218 end
219 end
```

```
220 end
```

onDischargeStateChanged

Description

Called on discharge state change

Definition

onDischargeStateChanged()

Code

```
224 function StrawBlower:onDischargeStateChanged(state)
225 local spec = self.spec_strawBlower
226 local samples = spec.samples
227 if self.isClient then
228 if state ~= Dischargeable.DISCHARGE_STATE_OFF then
229 g_soundManager:stopSample(samples.work)
230 g_soundManager:stopSample(samples.stop)
231 g_soundManager:playSample(samples.start)
232 g_soundManager:playSample(samples.work, 0, samples.start)
233 g_animationManager:startAnimations(spec.animationNodes)
234 else
235 g_soundManager:stopSample(samples.start)
236 g_soundManager:stopSample(samples.work)
237 g_soundManager:playSample(samples.stop)
238 g_animationManager:stopAnimations(spec.animationNodes)
239 end
240 end
241 end
```

StumpCutter

Description

This is the specialization for stump cutters

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17 function StumpCutter.prerequisitesPresent(specializations)
18 return SpecializationUtil.hasSpecialization(TurnOnVehicle,
   specializations)
19 end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
43  function StumpCutter:onLoad(savegame)
44  local spec = self.spec_stumpCutter
45
46  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.turnedOnRotationNodes.turnedOnRotationNode",
    "vehicle.stumpCutter.animationNodes.animationNode", "stumbCutter")
    --FS17 to FS19
47  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.stumpCutterStartSound",
    "vehicle.stumpCutter.sounds.start") --FS17 to FS19
48  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.stumpCutterIdleSound",
    "vehicle.stumpCutter.sounds.idle") --FS17 to FS19
49  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.stumpCutterWorkSound",
    "vehicle.stumpCutter.sounds.work") --FS17 to FS19
50  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.stumpCutterStopSound",
    "vehicle.stumpCutter.sounds.stop") --FS17 to FS19
51  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.stumpCutter.emitterShape(0)",
    "vehicle.stumpCutter.effects.effectNode") --FS17 to FS19
52  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.stumpCutter.particleSystem(0)",
    "vehicle.stumpCutter.effects.effectNode") --FS17 to FS19
53
54  local baseKey = "vehicle.stumpCutter"
55
56  spec.cutNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseKey .. "#cutNode"),
    self.i3dMappings)
57  spec.cutSizeY = Utils.getNoNil(getXMLFloat(self.xmlFile, baseKey
    .. "#cutSizeY"), 1)
58  spec.cutSizeZ = Utils.getNoNil(getXMLFloat(self.xmlFile, baseKey
    .. "#cutSizeZ"), 1)
59
```

```
60  spec.cutFullTreeThreshold =
    Utils.getNoNil(getXMLFloat(self.xmlFile, baseKey ..
    "#cutFullTreeThreshold"), 0.4)
61  spec.cutPartThreshold = Utils.getNoNil(getXMLFloat(self.xmlFile,
    baseKey .. "#cutPartThreshold"), 0.2)
62
63  if self.isClient then
64  spec.samples = {}
65  spec.samples.start =
    g_soundManager:loadSampleFromXML(self.xmlFile, baseKey ..
    ".sounds", "start", self.baseDirectory, self.components, 1,
    AudioGroup.VEHICLE, self.i3dMappings, self)
66  spec.samples.stop = g_soundManager:loadSampleFromXML(self.xmlFile,
    baseKey .. ".sounds", "stop", self.baseDirectory, self.components,
    1, AudioGroup.VEHICLE, self.i3dMappings, self)
67  spec.samples.idle = g_soundManager:loadSampleFromXML(self.xmlFile,
    baseKey .. ".sounds", "idle", self.baseDirectory, self.components,
    0, AudioGroup.VEHICLE, self.i3dMappings, self)
68  spec.samples.work = g_soundManager:loadSampleFromXML(self.xmlFile,
    baseKey .. ".sounds", "work", self.baseDirectory, self.components,
    0, AudioGroup.VEHICLE, self.i3dMappings, self)
69
70  spec.maxWorkFadeTime = 1000
71  spec.workFadeTime = 0
72
73  spec.effects = g_effectManager:loadEffect(self.xmlFile,
    baseKey..".effects", self.components, self, self.i3dMappings)
74  spec.animationNodes =
    g_animationManager:loadAnimations(self.xmlFile, baseKey ..
    ".animationNodes", self.components, self, self.i3dMappings)
75  end
76
77  spec.maxCutTime = Utils.getNoNil(getXMLFloat(self.xmlFile, baseKey
    .. "#maxCutTime"), 4000)
78  spec.nextCutTime = spec.maxCutTime
79  spec.maxResetCutTime = Utils.getNoNil(getXMLFloat(self.xmlFile,
    baseKey .. "#maxResetCutTime"), 1000)
80  spec.resetCutTime = spec.maxResetCutTime
81  end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```lua
function StumpCutter:onDelete()
if self.isClient then
local spec = self.spec_stumpCutter
g_effectManager:deleteEffects(spec.effects)
g_soundManager:deleteSamples(spec.samples)
g_animationManager:deleteAnimations(spec.animationNodes)
end
end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```lua
function StumpCutter:onUpdateTick(dt, isActiveForInput, isSelected)
if self:getIsTurnedOn() then
local spec = self.spec_stumpCutter

if spec.cutNode ~= nil then
spec.curLenAbove = 0

local x,y,z = getWorldTranslation(spec.cutNode)
local nx,ny,nz = localDirectionToWorld(spec.cutNode, 1,0,0)
local yx,yy,yz = localDirectionToWorld(spec.cutNode, 0,1,0)
if spec.curSplitShape ~= nil then
if testSplitShape(spec.curSplitShape, x,y,z, nx,ny,nz, yx,yy,yz, spec.cutSizeY, spec.cutSizeZ) == nil then
spec.curSplitShape = nil
end
end
if spec.curSplitShape == nil then
local shape, _, _, _, _ = findSplitShape(x,y,z, nx,ny,nz, yx,yy,yz, spec.cutSizeY, spec.cutSizeZ)
if shape ~= 0 then
spec.curSplitShape = shape
end
end
```

```
120
121 if VehicleDebug.state == VehicleDebug.DEBUG_ATTRIBUTES then
122 local x1, y1, z1 = localToWorld(spec.cutNode, 0, 0,
    spec.cutSizeZ)
123 local x2, y2, z2 = localToWorld(spec.cutNode, 0, spec.cutSizeY,
    0)
124 DebugUtil.drawDebugAreaRectangle(x, y, z, x1, y1, z1, x2, y2, z2,
    false, 0, 1, 0)
125 end
126
127 local isWorking = false
128 if spec.curSplitShape ~= nil then
129 local lenBelow, lenAbove =
    getSplitShapePlaneExtents(spec.curSplitShape, x,y,z, nx,ny,nz)
130 isWorking = lenAbove >= spec.cutPartThreshold
131
132 spec.workFadeTime = math.min(spec.maxWorkFadeTime,
    spec.workFadeTime + dt)
133 if self.isServer then
134 spec.resetCutTime = spec.maxResetCutTime
135 if spec.nextCutTime > 0 then
136 spec.nextCutTime = spec.nextCutTime - dt
137 if spec.nextCutTime <= 0 then
138 -- cut
139 local _,ly,_ = worldToLocal(spec.curSplitShape, x,y,z)
140 if (lenBelow <= spec.cutFullTreeThreshold or ly <
    spec.cutPartThreshold+0.01) and lenAbove < 1 then -- only delete
    tree if lenAbove < 1 to avoid full tree deletion
141 -- Delete full tree: Not much left below the cut or cutting near
    the ground
142 self:crushSplitShape(spec.curSplitShape)
143 spec.curSplitShape = nil
144 elseif lenAbove >= spec.cutPartThreshold then
145 -- Normal cut: Splitting 20cm below the top
146 spec.nextCutTime = spec.maxCutTime
147 local curSplitShape = spec.curSplitShape
148 spec.curSplitShape = nil
149 spec.curLenAbove = lenAbove
150 splitShape(curSplitShape, x,y,z, nx,ny,nz, yx,yy,yz,
    spec.cutSizeY, spec.cutSizeZ, "stumpCutterSplitShapeCallback",
    self)
151 else
152 spec.curSplitShape = nil
```

```
153  spec.nextCutTime = spec.maxCutTime
154  end
155  end
156  end
157  end
158  else
159  spec.workFadeTime = math.max(0, spec.workFadeTime - dt)
160  if self.isServer then
161  if spec.resetCutTime > 0 then
162  spec.resetCutTime = spec.resetCutTime - dt
163  if spec.resetCutTime <= 0 then
164  spec.nextCutTime = spec.maxCutTime
165  end
166  end
167  end
168  end
169
170  if self.isClient then
171  if isWorking then
172  g_effectManager:setFillType(spec.effects, FillType.WOODCHIPS)
173  g_effectManager:startEffects(spec.effects)
174  if not g_soundManager:getIsSamplePlaying(spec.samples.work) then
175  g_soundManager:playSample(spec.samples.work)
176  end
177  else
178  g_effectManager:stopEffects(spec.effects)
179  if g_soundManager:getIsSamplePlaying(spec.samples.work) then
180  g_soundManager:stopSample(spec.samples.work)
181  end
182  end
183  end
184  end
185  end
186  end
```

onDeactivate

Description

Called on deactivate

Definition

onDeactivate()

Code

```
190 function StumpCutter:onDeactivate()
191 if self.isClient then
192 local spec = self.spec_stumpCutter
193 g_effectManager:stopEffects(spec.effects)
194 end
195 end
```

onTurnedOn

Description

Called on turn on

Definition

onTurnedOn(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
200 function StumpCutter:onTurnedOn()
201 if self.isClient then
202 local spec = self.spec_stumpCutter
203 g_soundManager:stopSamples(spec.samples)
204 g_soundManager:playSample(spec.samples.start)
205 g_soundManager:playSample(spec.samples.idle, 0,
    spec.samples.start)
206 g_animationManager:startAnimations(spec.animationNodes)
207 end
208 end
```

onTurnedOff

Description

Called on turn off

Definition

onTurnedOff(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
213 function StumpCutter:onTurnedOff()
214 if self.isClient then
215 local spec = self.spec_stumpCutter
216 spec.workFadeTime = 0
217 g_effectManager:stopEffects(spec.effects)
218 g_soundManager:stopSamples(spec.samples)
219 g_soundManager:playSample(spec.samples.stop)
220 g_animationManager:stopAnimations(spec.animationNodes)
221 end
```

```
222 end
```

crushSplitShape

Description

Crush slit shape

Definition

crushSplitShape(integer shape)

Arguments

integer shape shape

Code

```
227 function StumpCutter:crushSplitShape(shape)
228 if self.isServer then
229 delete(shape)
230 end
231 end
```

stumpCutterSplitShapeCallback

Description

Split shape callback

Definition

stumpCutterSplitShapeCallback(integer shape, boolean isBelow, boolean isAbove, float minY, float maxY, float minZ, float maxZ)

Arguments

integer	shape	shape
boolean	isBelow	is below
boolean	isAbove	is above
float	minY	min y split position
float	maxY	max y split position
float	minZ	min z split position
float	maxZ	max z split position

Code

```
242 function StumpCutter:stumpCutterSplitShapeCallback(shape, isBelow, isAbove, minY, maxY, minZ, maxZ)
243 local spec = self.spec_stumpCutter
244
245 if not isBelow then
246 if spec.curLenAbove < 1 then -- split tree if lenAbove > 1 to avoid fulltree deletion
247 self:crushSplitShape(shape)
248 end
249 else
250 local yPos = minY + (maxY-minY)/2
251 local zPos = minZ + (maxZ-minZ)/2
252 local _,y,_ = localToWorld(spec.cutNode, -0.05, yPos, zPos)
```

```
253 local height = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, getWorldTranslation(spec.cutNode))
254 if y < height then
255 self:crushSplitShape(shape)
256 else
257 spec.curSplitShape = shape
258 end
259 end
260 end
```

getDirtMultiplier

Description

Returns current dirt multiplier

Definition

getDirtMultiplier()

Return Values

float dirtMultiplier current dirt multiplier

Code

```
265 function StumpCutter:getDirtMultiplier(superFunc)
266 local multiplier = superFunc(self)
267
268 local spec = self.spec_stumpCutter
269 if spec.curSplitShape ~= nil then
270 multiplier = multiplier + self:getWorkDirtMultiplier()
271 end
272
273 return multiplier
274 end
```

getWearMultiplier

Description

Returns current wear multiplier

Definition

getWearMultiplier()

Return Values

float wearMultiplier current wear multiplier

Code

```
279 function StumpCutter:getWearMultiplier(superFunc)
280 local multiplier = superFunc(self)
281
282 local spec = self.spec_stumpCutter
283 if spec.curSplitShape ~= nil then
284 multiplier = multiplier + self:getWorkWearMultiplier()
```

```
285 end
286
287 return multiplier
288 end
```

Tedder

Description

onTurnedOn

Description

Called on turn off

Definition

onTurnedOn(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
386 function Tedder:onTurnedOn()
387 if self.isClient then
388 local spec = self.spec_tedder
389 g_animationManager:startAnimations(spec.animationNodes)
390 end
391 end
```

onTurnedOff

Description

Called on turn off

Definition

onTurnedOff(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
396 function Tedder:onTurnedOff()
397 if self.isClient then
398 local spec = self.spec_tedder
399 g_animationManager:stopAnimations(spec.animationNodes)
400 for _, effect in ipairs(spec.effects) do
401 g_effectManager:stopEffects(effect.effects)
402 end
403 end
404 end
```

TensionBeltObject

Description

Class for vehicles which can be mounted by tension belts

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
30  function TensionBeltObject:onLoad(savegame)
31  local spec = self.spec_tensionBeltObject
32
33  spec.supportsTensionBelts =
    Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.tensionBeltObject#supportsTensionBelts"), true)
34
35  spec.meshNodes = {}
36  local i = 0
37  while true do
38  local baseKey =
    string.format("vehicle.tensionBeltObject.meshNodes.meshNode(%d)",
    i)
39  if not hasXMLProperty(self.xmlFile, baseKey) then
40  break
41  end
42
43  local node = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseKey.."#node"), self.i3dMappings)
44  if node ~= nil then
45  table.insert(spec.meshNodes, node)
46  end
47
48  i = i + 1
49  end
50  end
```

TensionBelts

Description

Class for vehicle which can dynamically mount objects (e.g. bales and pallets)

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```lua
function TensionBelts:onLoad(savegame)
local spec = self.spec_tensionBelts

local tensionBeltConfigurationId = Utils.getNoNil(self.configurations["t
local configKey =
string.format("vehicle.tensionBelts.tensionBeltsConfigurations.tensionBe
tensionBeltConfigurationId -1)
ObjectChangeUtil.updateObjectChanges(self.xmlFile,
"vehicle.tensionBelts.tensionBeltsConfigurations.tensionBeltsConfigurati
self.components, self)


spec.hasTensionBelts = true
if not hasXMLProperty(self.xmlFile, configKey) then
spec.hasTensionBelts = false
return
end

spec.belts = {}
spec.activatable = TensionBeltsActivatable:new(self)

spec.totalInteractionRadius = Utils.getNoNil(getXMLFloat(self.xmlFile,
configKey.."#totalInteractionRadius"), 6)
spec.interactionRadius = Utils.getNoNil(getXMLFloat(self.xmlFile, config
spec.interactionBaseNode = Utils.getNoNil(I3DUtil.indexToObject(self.com
configKey.."#interactionBasenode"), self.i3dMappings), self.rootNode)
spec.activationTrigger = I3DUtil.indexToObject(self.components, getXMLSt
configKey.."#activationTrigger"), self.i3dMappings)
if spec.activationTrigger ~= nil then
addTrigger(spec.activationTrigger, "tensionBeltActivationTriggerCallback
end

spec.isPlayerInTrigger = false
spec.checkSizeOffsets = {0*0.5, 5*0.5, 3*0.5}
spec.numObjectsIntensionBeltRange = 0

local tensionBeltType = Utils.getNoNil(getXMLString(self.xmlFile, config
local beltData = g_tensionBeltManager:getBeltData(tensionBeltType)

if beltData ~= nil then
spec.tensionBelts = {}
spec.singleBelts = {}
```

```
99  spec.sortedBelts = {}
100 spec.width = Utils.getNoNil(getXMLFloat(self.xmlFile, configKey.."#width
101 spec.ratchetPosition = getXMLFloat(self.xmlFile, configKey.."#ratchetPos
102 spec.useHooks = Utils.getNoNil(getXMLBool(self.xmlFile, configKey.."#use
103 spec.maxEdgeLength = Utils.getNoNil(getXMLFloat(self.xmlFile, configKey.
104 spec.geometryBias = Utils.getNoNil(getXMLFloat(self.xmlFile, configKey..
105 spec.defaultOffsetSide = Utils.getNoNil(getXMLFloat(self.xmlFile, config
106 spec.defaultOffset = Utils.getNoNil(getXMLFloat(self.xmlFile, configKey.
107 spec.defaultHeight = Utils.getNoNil(getXMLFloat(self.xmlFile, configKey.
108 spec.beltData = beltData
109 spec.linkNode = I3DUtil.indexToObject(self.components, getXMLString(self
    self.i3dMappings)
110 spec.rootNode = Utils.getNoNil(I3DUtil.indexToObject(self.components, ge
    configKey.."#rootNode"), self.i3dMappings), self.components[1].node)
111 spec.jointNode = Utils.getNoNil(I3DUtil.indexToObject(self.components, g
    configKey.."#jointNode"), self.i3dMappings), spec.rootNode)
112 spec.checkTimerDuration = 500
113 spec.checkTimer = spec.checkTimerDuration
114
115 if getRigidBodyType(spec.jointNode) ~= "Dynamic" and getRigidBodyType(sp
116 print("Error: Given jointNode '"..getName(spec.jointNode).."' has invali
    'Dynamic' or 'Kinematic'! Using '"..getName(self.components[1].node).."'
117 spec.jointNode = self.components[1].node
118 end
119
120 local rigidBodyType = getRigidBodyType(spec.jointNode)
121 spec.isDynamic = rigidBodyType == "Dynamic"
122
123 local x,y,z = localToLocal(spec.linkNode, spec.jointNode, 0,0,0)
124 local rx,ry,rz = localRotationToLocal(spec.linkNode, spec.jointNode, 0,0
125 spec.linkNodePosition = {x, y, z}
126 spec.linkNodeRotation = {rx, ry, rz}
127
128 local i = 0
129 while true do
130 local key = string.format(configKey..".tensionBelt(%d)", i)
131 if not hasXMLProperty(self.xmlFile, key) then
132 break
133 end
134
135 if #spec.sortedBelts == 2^TensionBelts.NUM_SEND_BITS then
```

```
136 print("Warning: Max number of tension belts is ".. 2^TensionBelts.NUM_SE
137 break
138 end
139
140 local startNode = I3DUtil.indexToObject(self.components, getXMLString(se
    self.i3dMappings)
141 local endNode = I3DUtil.indexToObject(self.components, getXMLString(self
    self.i3dMappings)
142 if startNode ~= nil and endNode ~= nil then
143 local x,y,_ = localToLocal(endNode, startNode, 0, 0, 0)
144
145 if math.abs(x) < 0.0001 and math.abs(y) < 0.0001 then
146 if spec.linkNode == nil then
147 spec.linkNode = getParent(startNode)
148 end
149 if spec.startNode == nil then
150 spec.startNode = startNode
151 end
152 spec.endNode = endNode
153
154 local offsetLeft = getXMLFloat(self.xmlFile, key.."#offsetLeft")
155 local offsetRight = getXMLFloat(self.xmlFile, key.."#offsetRight")
156 local offset = getXMLFloat(self.xmlFile, key.."#offset")
157 local height = getXMLFloat(self.xmlFile, key.."#height")
158
159 local intersectionNodes = {}
160 local j = 0
161 while true do
162 local intersectionKey = string.format(key..".intersectionNode(%d)", j)
163 if not hasXMLProperty(self.xmlFile, intersectionKey) then
164 break
165 end
166 local node = I3DUtil.indexToObject(self.components, getXMLString(self.xm
    self.i3dMappings)
167 if node ~= nil then
168 table.insert(intersectionNodes, node)
169 end
170 j = j + 1
171 end
172
```

```
173 local belt = {id=i+1, startNode=startNode, endNode=endNode, offsetLeft=o
    offset=offset, height=height, mesh=nil, intersectionNodes=intersectionNo
174 spec.singleBelts[belt] = belt
175 table.insert(spec.sortedBelts, belt)
176 else
177 print("Warning: x and y position of endNode need to be 0 for tension bel
    '"..self.configFileName.."'!")
178 end
179 end
180
181 i = i + 1
182 end
183
184 local minX, minZ = math.huge, math.huge
185 local maxX, maxZ = -math.huge, -math.huge
186 for _, belt in pairs(spec.singleBelts) do
187 local sx,_,sz = localToLocal(belt.startNode, spec.interactionBaseNode, 0
188 local ex,_,ez = localToLocal(belt.endNode, spec.interactionBaseNode, 0,
189 minX = math.min(minX, sx, ex)
190 minZ = math.min(minZ, sz, ez)
191 maxX = math.max(maxX, sx, ex)
192 maxZ = math.max(maxZ, sz, ez)
193 end
194
195 spec.interactionBasePointX = (maxX + minX) / 2
196 spec.interactionBasePointZ = (maxZ + minZ) / 2
197
198 for _, belt in pairs(spec.singleBelts) do
199 local sx,_,sz = localToLocal(belt.startNode, spec.interactionBaseNode, 0
200 local sl = MathUtil.vector2Length(spec.interactionBasePointX-sx, spec.in
201 local el = MathUtil.vector2Length(spec.interactionBasePointX-sx, spec.in
202 spec.totalInteractionRadius = math.max(spec.totalInteractionRadius, sl,
203 end
204
205 spec.hasTensionBelts = #spec.sortedBelts > 0
206 else
207 print("Warning: No belt data found for tension belts in '"..self.configF
208 end
209
210 spec.checkBoxes = {}
211 spec.objectsToJoint = {}
```

```
212  spec.isPlayerInRange = false
213  spec.currentBelt = nil
214  spec.areBeltsFasten = false
215  end
```

onPostLoad

Description

Called after loading

Definition

onPostLoad(table savegame)

Arguments

table savegame savegame

Code

```
220  function TensionBelts:onPostLoad(savegame)
221  if savegame ~= nil then
222  local spec = self.spec_tensionBelts
223
224  spec.beltsToLoad = {}
225  local i = 0
226  while true do
227  local key = string.format("%s.tensionBelts.belt(%d)",
     savegame.key, i)
228  if not hasXMLProperty(savegame.xmlFile, key) then
229  break
230  end
231  if getXMLBool(savegame.xmlFile, key.."#isActive") then
232  table.insert(spec.beltsToLoad, i+1)
233  end
234  i = i + 1
235  end
236  end
237  end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
263  function TensionBelts:onDelete()
264  local spec = self.spec_tensionBelts
265
266  spec.isPlayerInRange = false
```

```
267  g_currentMission:removeActivatableObject(spec.activatable)
268  self:setTensionBeltsActive(false, nil, true)
269
270  if spec.activationTrigger ~= nil then
271  removeTrigger(spec.activationTrigger)
272  spec.activationTrigger = nil
273  end
274  end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
280  function TensionBelts:onReadStream(streamId, connection)
281  local spec = self.spec_tensionBelts
282
283  if not spec.hasTensionBelts then
284  return
285  end
286  if spec.tensionBelts ~= nil then
287  spec.beltsToLoad = {}
288  for k, _ in ipairs(spec.sortedBelts) do
289  local beltActive = streamReadBool(streamId)
290  if beltActive then
291  table.insert(spec.beltsToLoad, k)
292  end
293  end
294  end
295  end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
301 function TensionBelts:onWriteStream(streamId, connection)
302 local spec = self.spec_tensionBelts
303
304 if not spec.hasTensionBelts then
305 return
306 end
307 if spec.tensionBelts ~= nil then
308 for _, belt in ipairs(spec.sortedBelts) do
309 streamWriteBool(streamId, belt.mesh ~= nil)
310 end
311 end
312 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
319 function TensionBelts:onUpdate(dt, isActiveForInput, isSelected)
320 local spec = self.spec_tensionBelts
321
322 if not spec.hasTensionBelts then
323 return
324 end
325
326 if spec.beltsToLoad ~= nil then
327 for _, k in pairs(spec.beltsToLoad) do
328 local noEventSend = false
329 if not self.isServer then
330 -- do not resend event to server after sync. Only send event after loadi
    make sure client have visible tension belts
331 noEventSend = true
332 end
333 self:setTensionBeltsActive(true, spec.sortedBelts[k].id, noEventSend)
334 end
335 spec.beltsToLoad = nil
```

```
336 end
337
338 if self.isServer and spec.isDynamic then
339 local x,y,z = localToLocal(spec.linkNode, spec.jointNode, 0,0,0)
340 local rx,ry,rz = localRotationToLocal(spec.linkNode, spec.jointNode, 0,0
341
342 local isDirty = false
343 if math.abs(x - spec.linkNodePosition[1]) > 0.001 or math.abs(y -
    spec.linkNodePosition[2]) > 0.001 or math.abs(z - spec.linkNodePosition[
    0.001 or
344 math.abs(rx - spec.linkNodeRotation[1]) > 0.001 or math.abs(ry -
    spec.linkNodeRotation[2]) > 0.001 or math.abs(rz - spec.linkNodeRotation
    0.001 then
345 isDirty = true
346 end
347 if isDirty then
348 spec.linkNodePosition[1], spec.linkNodePosition[2], spec.linkNodePositio
    x, y, z
349 spec.linkNodeRotation[1], spec.linkNodeRotation[2], spec.linkNodeRotatio
    rx, ry, rz
350 for _, joint in pairs(spec.objectsToJoint) do
351 setJointFrame(joint.jointIndex, 0, joint.jointTransform)
352 end
353 end
354 end
355
356 if TensionBelts.debugRendering then
357 for belt,_ in pairs(spec.belts) do
358 DebugUtil.drawDebugNode(belt)
359 for i=0, getNumOfChildren(belt)-1 do
360 DebugUtil.drawDebugNode(getChildAt(belt, i))
361 end
362 end
363
364 if spec.checkBoxes ~= nil then
365 for _, box in pairs(spec.checkBoxes) do
366 local p = box.points
367 local c = box.color
368
369 -- bottom
370 drawDebugLine(p[1][1],p[1][2],p[1][3], c[1],c[2],c[3], p[2][1],p[2][2],p
    c[1],c[2],c[3])
```

```
371 drawDebugLine(p[2][1],p[2][2],p[2][3], c[1],c[2],c[3], p[3][1],p[3][2],p
    c[1],c[2],c[3])
372 drawDebugLine(p[3][1],p[3][2],p[3][3], c[1],c[2],c[3], p[4][1],p[4][2],p
    c[1],c[2],c[3])
373 drawDebugLine(p[4][1],p[4][2],p[4][3], c[1],c[2],c[3], p[1][1],p[1][2],p
    c[1],c[2],c[3])
374 -- top
375 drawDebugLine(p[5][1],p[5][2],p[5][3], c[1],c[2],c[3], p[6][1],p[6][2],p
    c[1],c[2],c[3])
376 drawDebugLine(p[6][1],p[6][2],p[6][3], c[1],c[2],c[3], p[7][1],p[7][2],p
    c[1],c[2],c[3])
377 drawDebugLine(p[7][1],p[7][2],p[7][3], c[1],c[2],c[3], p[8][1],p[8][2],p
    c[1],c[2],c[3])
378 drawDebugLine(p[8][1],p[8][2],p[8][3], c[1],c[2],c[3], p[5][1],p[5][2],p
    c[1],c[2],c[3])
379 -- left
380 drawDebugLine(p[1][1],p[1][2],p[1][3], c[1],c[2],c[3], p[5][1],p[5][2],p
    c[1],c[2],c[3])
381 drawDebugLine(p[4][1],p[4][2],p[4][3], c[1],c[2],c[3], p[8][1],p[8][2],p
    c[1],c[2],c[3])
382 -- right
383 drawDebugLine(p[2][1],p[2][2],p[2][3], c[1],c[2],c[3], p[6][1],p[6][2],p
    c[1],c[2],c[3])
384 drawDebugLine(p[3][1],p[3][2],p[3][3], c[1],c[2],c[3], p[7][1],p[7][2],p
    c[1],c[2],c[3])
385 drawDebugPoint(p[9][1], p[9][2], p[9][3], 1, 1, 1, 1)
386 end
387 end
388
389 local a,b,c = localToWorld(spec.interactionBaseNode,
    spec.interactionBasePointX, 0, spec.interactionBasePointZ)
390 drawDebugPoint(a,b,c, 0,0,1, 1,1,1,1)
391
392 for i=0, 360-10, 10 do
393 local x,y,z = localToWorld(spec.interactionBaseNode,
    spec.interactionBasePointX+math.cos(math.rad(i))*spec.totalInteractionRa
    spec.interactionBasePointZ+math.sin(math.rad(i))*spec.totalInteractionRa
394 drawDebugPoint(x, y, z, 1,1,1,1)
395 for _, belt in pairs(spec.singleBelts) do
396 local x,y,z = localToWorld(belt.startNode,
    math.cos(math.rad(i))*spec.interactionRadius, 0,
    math.sin(math.rad(i))*spec.interactionRadius)
397 drawDebugPoint(x, y, z, 0,1,0,1)
```

```
398 local x,y,z = localToWorld(belt.endNode,
    math.cos(math.rad(i))*spec.interactionRadius, 0,
    math.sin(math.rad(i))*spec.interactionRadius)
399 drawDebugPoint(x, y, z, 1,0,0,1)
400 end
401 end
402 end
403
404 if spec.isPlayerInTrigger or spec.isPlayerInRange then
405 self:raiseActive()
406 end
407
408 -- create tension belt preview and control activation
409 local currentBelt = nil
410 spec.isPlayerInRange, currentBelt = self:getIsPlayerInTensionBeltsRange
411 if spec.isPlayerInRange then
412 if currentBelt ~= spec.currentBelt then
413 if spec.currentBelt ~= nil and spec.currentBelt.dummy ~= nil then
414 delete(spec.currentBelt.dummy)
415 spec.currentBelt.dummy = nil
416 end
417 spec.currentBelt = currentBelt
418
419 if spec.currentBelt ~= nil and spec.currentBelt.mesh == nil then
420 local objects, _ = self:getObjectToMount(spec.currentBelt)
421 self:createTensionBelt(spec.currentBelt, true, objects)
422 end
423 end
424 g_currentMission:addActivatableObject(spec.activatable)
425 else
426 g_currentMission:removeActivatableObject(spec.activatable)
427 if spec.currentBelt ~= nil and spec.currentBelt.dummy ~= nil then
428 delete(spec.currentBelt.dummy)
429 spec.currentBelt.dummy = nil
430 spec.currentBelt = nil
431 end
432 end
433 end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
440 function TensionBelts:onUpdateTick(dt, isActiveForInput,
    isSelected)
441 local spec = self.spec_tensionBelts
442
443 if not spec.hasTensionBelts then
444 return
445 end
446
447 if self.isServer and spec.tensionBelts ~= nil then
448 spec.checkTimer = spec.checkTimer - dt
449 if spec.checkTimer < 0 then
450 -- check if entities have been deleted
451 local needUpdate = false
452 for phyiscObject, _ in pairs(spec.objectsToJoint) do
453 if not entityExists(phyiscObject) then
454 spec.objectsToJoint[phyiscObject] = nil
455 needUpdate = true
456 break
457 end
458 end
459 if needUpdate then
460 self:refreshTensionBelts()
461 end
462
463 spec.checkTimer = spec.checkTimerDuration
464 end
465 end
466 end
```

onDraw

Description

Called on draw

Definition

onDraw(boolean isActiveForInput, boolean isSelected)

Arguments

boolean isActiveForInput true if vehicle is active for input
boolean isSelected true if vehicle is selected

Code

```
472 function TensionBelts:onDraw(isActiveForInput, isSelected)
473 local spec = self.spec_tensionBelts
474
475 if not spec.hasTensionBelts then
476 return
477 end
478
479 if isActiveForInput and isSelected then
480 if spec.areBeltsFasten then
481 g_currentMission:addHelpButtonText(g_i18n:getText("action_unfastenTensio
    InputBinding.TOGGLE_TENSION_BELTS, nil, GS_PRIO_NORMAL)
482 else
483 g_currentMission:addHelpButtonText(g_i18n:getText("action_fastenTensionB
    InputBinding.TOGGLE_TENSION_BELTS, nil, GS_PRIO_NORMAL)
484 end
485 end
486
487 -- for _, d in pairs(spec.objectsToJoint) do
488 -- local node = d.jointTransform
489 -- DebugUtil.drawDebugNode(node, getName(node))
490 -- end
491 end
```

setTensionBeltsActive

Description

Set tensionbelts active state

Definition

setTensionBeltsActive(boolean isActive, integer beltId)

Arguments

boolean isActive new active state
integer beltId id of belt to set state (if id is nil it uses every belt)

Code

```
601 function TensionBelts:setTensionBeltsActive(isActive, beltId,
    noEventSend)
602 local spec = self.spec_tensionBelts
603
604 if spec.tensionBelts ~= nil then
605 TensionBeltsEvent.sendEvent(self, isActive, beltId, noEventSend)
606
607 local belt = nil
```

```
608 if beltId ~= nil then
609 belt = spec.sortedBelts[beltId]
610 end
611
612 if isActive then
613 local objects, _ = self:getObjectToMount(belt)
614 if belt == nil then
615 for _, belt in pairs(spec.singleBelts) do
616 if belt.mesh == nil then
617 self:createTensionBelt(belt, false, objects)
618 end
619 end
620 else
621 if belt.mesh == nil then
622 self:createTensionBelt(belt, false, objects)
623 end
624 end
625
626 if self.isServer then
627 for _, object in pairs(objects) do
628 self:lockTensionBeltObject(object.physics, spec.objectsToJoint,
    spec.isDynamic, spec.jointNode)
629 end
630 end
631 else
632 if belt == nil then
633 for _, belt in pairs(spec.singleBelts) do
634 self:removeTensionBelt(belt)
635 end
636 else
637 self:removeTensionBelt(belt)
638 end
639
640 if self.isServer then
641 -- remove joints
642 local objectIds, _ = self:getObjectsToUnmount(belt)
643 for _, objectId in pairs(objectIds) do
644 self:freeTensionBeltObject(objectId, spec.objectsToJoint,
    spec.isDynamic)
645 end
646 end
```

```
647 end
648
649 self:updateFastenState()
650 end
651 end
```

updateFastenState

Description

Update 'self.areBeltsFasten'

Definition

updateFastenState()

Code

```
655 function TensionBelts:updateFastenState()
656 local spec = self.spec_tensionBelts
657
658 local unfastenBelts = false
659 for _, belt in pairs(spec.singleBelts) do
660 if belt.mesh == nil then
661 unfastenBelts = true
662 break
663 end
664 end
665
666 spec.areBeltsFasten = not unfastenBelts
667 end
```

createTensionBelt

Description

Create tension belt

Definition

createTensionBelt(table belt, boolean isDummy, table object)

Arguments

table	belt	belt to use
boolean	isDummy	create dummy belt
table	object	objects to fasten

Code

```
674 function TensionBelts:createTensionBelt(belt, isDummy, objects)
675 local spec = self.spec_tensionBelts
676
677 local tensionBelt = TensionBeltGeometryConstructor:new()
678
679 local beltData = spec.beltData
680
```

```
681 tensionBelt:setWidth(spec.width)
682 tensionBelt:setMaxEdgeLength(spec.maxEdgeLength)
683 if isDummy then
684 tensionBelt:setMaterial(beltData.dummyMaterial.materialId)
685 tensionBelt:setUVscale(beltData.dummyMaterial.uvScale)
686 else
687 tensionBelt:setMaterial(beltData.material.materialId)
688 tensionBelt:setUVscale(beltData.material.uvScale)
689 end
690
691 if spec.ratchetPosition ~= nil and beltData.ratchet ~= nil then
692 tensionBelt:addAttachment(0, spec.ratchetPosition,
    beltData.ratchet.sizeRatio*spec.width)
693 end
694
695 if spec.useHooks and beltData.hook ~= nil then
696 tensionBelt:addAttachment(0, 0,
    beltData.hook.sizeRatio*spec.width)
697 tensionBelt:addAttachment(1, 0,
    beltData.hook.sizeRatio*spec.width)
698 end
699
700 tensionBelt:setFixedPoints(belt.startNode, belt.endNode)
701 tensionBelt:setGeometryBias(spec.geometryBias)
702 tensionBelt:setLinkNode(spec.linkNode)
703
704 for _, pointNode in pairs(belt.intersectionNodes) do
705 local x,y,z = getWorldTranslation(pointNode)
706 local dirX, dirY, dirZ = localDirectionToWorld(pointNode, 1, 0,
    0)
707 tensionBelt:addIntersectionPoint(x,y,z, dirX, dirY, dirZ)
708 end
709
710 for _, object in pairs(objects) do
711 for _, node in pairs(object.visuals) do
712 tensionBelt:addShape(node, 0, 1, 0, 1)
713 end
714 end
715
716 local beltShape, _, beltLength = tensionBelt:finalize()
717 if beltShape ~= 0 then
```

```
718 if isDummy then
719 belt.dummy = beltShape
720 else
721 local currentIndex = 0
722 if spec.ratchetPosition ~= nil and beltData.ratchet ~= nil then
723 if getNumOfChildren(beltShape) > currentIndex then
724 local scale = spec.width
725 local ratched = clone(beltData.ratchet.node, false, false, false)
726 link(getChildAt(beltShape, 0), ratched)
727 setScale(ratched, scale, scale, scale)
728 currentIndex = currentIndex + 1
729 end
730 end
731 if spec.useHooks then
732 if beltData.hook ~= nil and getNumOfChildren(beltShape) >
    currentIndex+1 then
733 local scale = spec.width
734 local hookStart = clone(beltData.hook.node, false, false, false)
735 link(getChildAt(beltShape, currentIndex), hookStart)
736 setScale(hookStart, scale, scale, scale)
737
738 local hookEnd = clone(beltData.hook.node, false, false, false)
739 link(getChildAt(beltShape, currentIndex+1), hookEnd)
740 setRotation(hookEnd, 0, math.pi, 0)
741 setTranslation(hookEnd, 0, 0, beltData.hook.sizeRatio*spec.width)
742 setScale(hookEnd, scale, scale, scale)
743 currentIndex = currentIndex + 2
744
745 setShaderParameter(beltShape, "beltClipOffsets", 0,
    beltData.hook.sizeRatio*spec.width, beltLength-
    beltData.hook.sizeRatio*spec.width, beltLength, false)
746 end
747 end
748
749 belt.mesh = beltShape
750 spec.belts[beltShape] = beltShape
751 if belt.dummy ~= nil then
752 delete(belt.dummy)
753 belt.dummy = nil
754 end
755 end
```

```
756
757 return beltShape
758 end
759
760 return nil
761 end
```

removeTensionBelt

Description

Remove tension belt

Definition

removeTensionBelt(table belt)

Arguments

table belt belt to remove

Code

```
766 function TensionBelts:removeTensionBelt(belt)
767 if belt.mesh ~= nil then
768 local spec = self.spec_tensionBelts
769
770 spec.belts[belt.mesh] = nil
771 delete(belt.mesh)
772 belt.mesh = nil
773 if spec.currentBelt == belt then
774 spec.currentBelt = nil
775 end
776 end
777 end
```

getObjectToMount

Description

Returns objects in belt range

Definition

getObjectToMount(table belt)

Arguments

table belt belt to check

Return Values

table objectsInTensionBeltRange object in belt range

integer numObjectsIntensionBeltRange number of objects in belt range

Code

```
784 function TensionBelts:getObjectToMount(belt)
785 local spec = self.spec_tensionBelts
786
787 local markerStart = spec.startNode
788 local markerEnd = spec.endNode
```

```
789 local offsetLeft = nil
790 local offsetRight = nil
791 local offset = nil
792 local height = nil
793 if belt ~= nil then
794 markerStart = belt.startNode
795 markerEnd = belt.endNode
796
797 offsetLeft = belt.offsetLeft
798 offsetRight = belt.offsetRight
799 offset = belt.offset
800 height = belt.height
801 if offsetLeft == nil then
802 if spec.sortedBelts[belt.id-1] ~= nil and
    spec.sortedBelts[belt.id-1].mesh ~= nil then
803 local x,_,_ = localToLocal(markerStart, spec.sortedBelts[belt.id-
    1].startNode, 0, 0, 0)
804 offsetLeft = math.abs(x)
805 end
806 end
807 if offsetRight == nil then
808 if spec.sortedBelts[belt.id+1] ~= nil and
    spec.sortedBelts[belt.id+1].mesh ~= nil then
809 local x,_,_ = localToLocal(markerStart,
    spec.sortedBelts[belt.id+1].startNode, 0, 0, 0)
810 offsetRight = math.abs(x)
811 end
812 end
813
814 end
815
816 if offsetLeft == nil then
817 offsetLeft = spec.defaultOffsetSide
818 end
819 if offsetRight == nil then
820 offsetRight = spec.defaultOffsetSide
821 end
822 if offset == nil then
823 offset = spec.defaultOffset
824 end
825 if height == nil then
```

```
826 height = spec.defaultHeight
827 end
828
829 local sizeX = (offsetLeft+offsetRight) * 0.5
830 local sizeY = height * 0.5
831 local _, _, width = localToLocal(markerEnd, markerStart, 0, 0, 0)
832 local sizeZ = width*0.5 - 2*offset
833
834 local centerX = (offsetLeft-offsetRight)*0.5
835 local centerY = height*0.5
836 local centerZ = width*0.5
837 local x,y,z = localToWorld(markerStart, centerX, centerY,
    centerZ)
838
839 if TensionBelts.debugRendering then
840 local box = {}
841 box.points = {}
842 local colorR = math.random(0, 1)
843 local colorG = math.random(0, 1)
844 local colorB = math.random(0, 1)
845 box.color = {colorR,colorG,colorB}
846
847 local blx, bly, blz = localToWorld(markerStart, centerX-sizeX,
    centerY-sizeY, centerZ-sizeZ)
848 local brx, bry, brz = localToWorld(markerStart, centerX+sizeX,
    centerY-sizeY, centerZ-sizeZ)
849 local flx, fly, flz = localToWorld(markerStart, centerX-sizeX,
    centerY-sizeY, centerZ+sizeZ)
850 local frx, fry, frz = localToWorld(markerStart, centerX+sizeX,
    centerY-sizeY, centerZ+sizeZ)
851
852 local tblx, tbly, tblz = localToWorld(markerStart, centerX-sizeX,
    centerY+sizeY, centerZ-sizeZ)
853 local tbrx, tbry, tbrz = localToWorld(markerStart, centerX+sizeX,
    centerY+sizeY, centerZ-sizeZ)
854 local tflx, tfly, tflz = localToWorld(markerStart, centerX-sizeX,
    centerY+sizeY, centerZ+sizeZ)
855 local tfrx, tfry, tfrz = localToWorld(markerStart, centerX+sizeX,
    centerY+sizeY, centerZ+sizeZ)
856
857 table.insert(box.points, { blx, bly, blz } ) -- lower: lb
858 table.insert(box.points, { brx, bry, brz } ) -- rb
```

```
859  table.insert(box.points, { frx, fry, frz } ) -- rf
860  table.insert(box.points, { flx, fly, flz } ) -- lf
861
862  table.insert(box.points, { tblx, tbly, tblz } ) -- upper: lb
863  table.insert(box.points, { tbrx, tbry, tbrz } ) -- rb
864  table.insert(box.points, { tfrx, tfry, tfrz } ) -- rf
865  table.insert(box.points, { tflx, tfly, tflz } ) -- lf
866  table.insert(box.points, { x, y, z } ) -- center
867
868  spec.checkBoxes[markerStart] = box
869  end
870
871  local rx,ry,rz = getWorldRotation(markerStart)
872  spec.objectsInTensionBeltRange = {}
873  spec.numObjectsIntensionBeltRange = 0
874
875  -- collision mask : all bits except bit 13, 23, 30
876  overlapBox(x, y, z, rx, ry, rz, sizeX, sizeY, sizeZ,
     "objectOverlapCallback", self, 3212828671, true, false, true)
877
878  return spec.objectsInTensionBeltRange,
     spec.numObjectsIntensionBeltRange
879  end
```

getObjectsToUnmount

Description

Returns objects to unmount if given belt will be removed

Definition

getObjectsToUnmount(table belt)

Arguments

table belt belt to check

Return Values

table objectIdsToUnmount table with object ids to unmount

integer numObjects number of objects to unmount

Code

```
886  function TensionBelts:getObjectsToUnmount(belt)
887  local spec = self.spec_tensionBelts
888
889  local objectIdsToUnmount = {}
890  local numObjects = 0
891  -- copy mounted objects table
892  for objectId, _ in pairs(spec.objectsToJoint) do
893  objectIdsToUnmount[objectId] = objectId
```

```
894 numObjects = numObjects + 1
895 end
896
897 -- remove objects mounted by other active belts
898 for _, otherBelt in pairs(spec.singleBelts) do
899 if otherBelt.mesh ~= nil and otherBelt ~= belt then
900 local objectToMount,_ = self:getObjectToMount(otherBelt)
901 for _, object in pairs(objectToMount) do
902 -- remove objects mounted by other belts
903 if objectIdsToUnmount[object.physics] ~= nil then
904 objectIdsToUnmount[object.physics] = nil
905 numObjects = numObjects - 1
906 end
907 end
908 end
909 end
910
911 return objectIdsToUnmount, numObjects
912 end
```

objectOverlapCallback

Description

Overlap callback

Definition

objectOverlapCallback(integer transformId)

Arguments

integer transformId id of found transform

Code

```
917 function TensionBelts:objectOverlapCallback(transformId)
918 if transformId ~= 0 and getHasClassId(transformId,
    ClassIds.SHAPE) then
919 local spec = self.spec_tensionBelts
920
921 local object = g_currentMission:getNodeObject(transformId)
922 if object ~= nil then
923 if object.getSupportsTensionBelts ~= nil and
    object:getSupportsTensionBelts() then
924 if object.getMeshNodes ~= nil and object.dynamicMountObject ==
    nil then
925 local nodeId = object:getTensionBeltNodeId()
926 if spec.objectsInTensionBeltRange[nodeId] == nil then
927 local nodes = object:getMeshNodes()
```

```
928  if nodes ~= nil then
929  spec.objectsInTensionBeltRange[nodeId] = {physics=nodeId,
     visuals=nodes}
930  spec.numObjectsIntensionBeltRange =
     spec.numObjectsIntensionBeltRange + 1
931  end
932  end
933  end
934  end
935  elseif getSplitType(transformId) ~= 0 then
936  local rigidBodyType = getRigidBodyType(transformId)
937  if (rigidBodyType == "Dynamic" or rigidBodyType == "Kinematic")
     and spec.objectsInTensionBeltRange[transformId] == nil then
938  spec.objectsInTensionBeltRange[transformId] =
     {physics=transformId, visuals={transformId}}
939  spec.numObjectsIntensionBeltRange =
     spec.numObjectsIntensionBeltRange + 1
940  end
941  end
942  end
943
944  return true
945  end
```

getIsPlayerInTensionBeltsRange

Description

Returns if player is in tension belt range

Definition

getIsPlayerInTensionBeltsRange()

Return Values

boolean	inRange	player is in range
table	belt	nearest belt

Code

```
951  function TensionBelts:getIsPlayerInTensionBeltsRange()
952
953  if g_currentMission.player == nil then
954  return false, nil
955  end
956
957  local spec = self.spec_tensionBelts
958
959  local px, _, pz =
     getWorldTranslation(g_currentMission.player.rootNode)
```

```
960 local vx, _, vz = localToWorld(spec.interactionBaseNode,
    spec.interactionBasePointX, 0, spec.interactionBasePointZ)
961 local currentBelt = nil
962 local distance = math.huge
963
964 if MathUtil.vector2Length(px-vx, pz-vz) <
    spec.totalInteractionRadius then
965 if spec.tensionBelts ~= nil then
966 for _, belt in pairs(spec.singleBelts) do
967 local sx, _, sz = getWorldTranslation(belt.startNode)
968 local ex, _, ez = getWorldTranslation(belt.endNode)
969 local sDistance = MathUtil.vector2Length(px-sx, pz-sz)
970 local eDistance = MathUtil.vector2Length(px-ex, pz-ez)
971 if (sDistance < distance and sDistance < spec.interactionRadius)
    or (eDistance < distance and eDistance < spec.interactionRadius)
    then
972 currentBelt = belt
973 distance = math.min(sDistance, eDistance)
974 end
975 end
976 end
977
978 if distance < spec.interactionRadius then
979 return true, currentBelt
980 end
981 end
982
983 return false, nil
984 end
```

TreePlanter

Description

Class for all tree planters

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
20  function TreePlanter.prerequisitesPresent(specializations)
21  return SpecializationUtil.hasSpecialization(TurnOnVehicle,
    specializations)
22  and SpecializationUtil.hasSpecialization(FillUnit,
    specializations)
23  and SpecializationUtil.hasSpecialization(GroundReference,
    specializations)
24  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
66  function TreePlanter:onLoad(savegame)
67  local spec = self.spec_treePlanter
68
69  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.treePlanterSound", "vehicle.treePlanter.sounds.work") --FS17
    to FS19
70  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.turnedOnRotationNodes.turnedOnRotationNode(0)",
    "vehicle.treePlanter.animationNodes.animationNode") --FS17 to FS19
71
72  local baseKey = "vehicle.treePlanter"
73
74  if self.isClient then
75  spec.samples = {}
76  spec.samples.work = g_soundManager:loadSampleFromXML(self.xmlFile,
    baseKey..".sounds", "work", self.baseDirectory, self.components, 0,
    AudioGroup.VEHICLE, self.i3dMappings, self)
77  spec.isWorkSamplePlaying = false
78
79  spec.animationNodes = g_animationManager:loadAnimations(self.xmlFile,
    baseKey..".animationNodes", self.components, self, self.i3dMappings)
80  end
81
82  spec.node = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseKey.."#node"), self.i3dMappings)
83  spec.minDistance = Utils.getNoNil(getXMLFloat(self.xmlFile,
    baseKey.."#minDistance"), 20) -- distance to next tree
84
```

```
85  spec.palletTrigger = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseKey.."#palletTrigger"),
    self.i3dMappings)
86  if spec.palletTrigger ~= nil then
87  addTrigger(spec.palletTrigger, "palletTriggerCallback", self)
88  else
89  g_logManager:xmlWarning(self.configFileName, "TreePlanter requires a
    palletTrigger!")
90  end
91  spec.palletsInTrigger = {}
92
93  local refNodeIndex = Utils.getNoNil(getXMLInt(self.xmlFile,
    baseKey.."#refNodeIndex"), 1)
94  spec.groundReferenceNode =
    self:getGroundReferenceNodeFromIndex(refNodeIndex)
95  if spec.groundReferenceNode == nil then
96  g_logManager:xmlWarning(self.configFileName, "No groundReferenceNode
    specified or invalid groundReferenceNode index in '%s'",
    baseKey.."#refNodeIndex")
97  end
98
99  spec.activatable = TreePlanterActivatable:new(self)
100
101 spec.saplingPalletGrabNode =
    Utils.getNoNil(I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseKey.."#saplingPalletGrabNode"),
    self.i3dMappings), self.rootNode)
102 spec.saplingPalletMountNode =
    Utils.getNoNil(I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseKey.."#saplingPalletMountNode"),
    self.i3dMappings), self.rootNode)
103 spec.mountedSaplingPallet = nil
104
105 spec.fillUnitIndex = Utils.getNoNil(getXMLInt(self.xmlFile,
    baseKey.."#fillUnitIndex"), 1)
106 spec.nearestPalletDistance = Utils.getNoNil(getXMLInt(self.xmlFile,
    baseKey.."#palletMountingRange"), 6.0)
107
108 spec.currentTree = 1
109 spec.lastTreePos = nil
110
111 spec.showFieldNotOwnedWarning = false
112 spec.showRestrictedZoneWarning = false
113 spec.showTooManyTreesWarning = false
```

```
114 spec.hasGroundContact = false
115
116 spec.limitToField = true
117 spec.forceLimitToField = false
118
119 -- attributes for AI
120 if self.addAITerrainDetailRequiredRange ~= nil then
121 self:addAITerrainDetailRequiredRange(g_currentMission.cultivatorValue,
    g_currentMission.cultivatorValue,
    g_currentMission.terrainDetailTypeFirstChannel,
    g_currentMission.terrainDetailTypeNumChannels)
122 self:addAITerrainDetailRequiredRange(g_currentMission.plowValue,
    g_currentMission.plowValue,
    g_currentMission.terrainDetailTypeFirstChannel,
    g_currentMission.terrainDetailTypeNumChannels)
123 self:addAITerrainDetailRequiredRange(g_currentMission.sowingValue,
    g_currentMission.sowingValue,
    g_currentMission.terrainDetailTypeFirstChannel,
    g_currentMission.terrainDetailTypeNumChannels)
124 end
125
126 if self.setAIFruitProhibitions ~= nil then
127 self:setAIFruitProhibitions(FruitType.POPLAR, 1, 5)
128 end
129
130 spec.dirtyFlag = self:getNextDirtyFlag()
131
132 if savegame ~= nil and not savegame.resetVehicles then
133 spec.lastTreePos =
    StringUtil.getVectorNFromString(getXMLString(savegame.xmlFile,
    savegame.key..".treePlanter#lastTreePos"), 3)
134
135 spec.palletHadBeenMounted = getXMLBool(savegame.xmlFile, savegame.key
    .. ".treePlanter#palletHadBeenMounted")
136 end
137 end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
141 function TreePlanter:onDelete()
142 local spec = self.spec_treePlanter
```

```
143
144 if self.isClient then
145 g_soundManager:deleteSamples(spec.samples)
146 g_animationManager:deleteAnimations(spec.animationNodes)
147
148 if spec.activatable ~= nil then
149 g_currentMission:removeActivatableObject(spec.activatable)
150 end
151 end
152
153 if spec.mountedSaplingPallet ~= nil then
154 spec.mountedSaplingPallet:unmount()
155 spec.mountedSaplingPallet = nil
156 end
157
158 if spec.palletTrigger ~= nil then
159 removeTrigger(spec.palletTrigger)
160 end
161 end
```

removeMountedObject

Description

Remove mounted object

Definition

removeMountedObject(integer object, boolean isDeleting)

Arguments

integer	object	object to remove
boolean	isDeleting	called on delete

Code

```
178 function TreePlanter:removeMountedObject(object, isDeleting)
179 local spec = self.spec_treePlanter
180
181 if spec.mountedSaplingPallet == object then
182 spec.mountedSaplingPallet:unmount()
183 spec.mountedSaplingPallet = nil
184 end
185 end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
191 function TreePlanter:onReadStream(streamId, connection)
192 if streamReadBool(streamId) then
193 local spec = self.spec_treePlanter
194 spec.palletIdToMount = NetworkUtil.readNodeObjectId(streamId)
195 end
196 end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
202 function TreePlanter:onWriteStream(streamId, connection)
203 local spec = self.spec_treePlanter
204 streamWriteBool(streamId, spec.mountedSaplingPallet ~= nil)
205 if spec.mountedSaplingPallet ~= nil then
206 local palletId =
    NetworkUtil.getObjectId(spec.mountedSaplingPallet)
207 NetworkUtil.writeNodeObjectId(streamId, palletId)
208 end
209 end
```

onReadUpdateStream

Description

Called on on update

Definition

onReadUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId stream ID

integer timestamp timestamp

table connection connection

Code

```
216 function TreePlanter:onReadUpdateStream(streamId, timestamp,
    connection)
217 if connection:getIsServer() then
218 local spec = self.spec_treePlanter
219 if streamReadBool(streamId) then
```

```
220     spec.hasGroundContact = streamReadBool(streamId)
221     spec.showFieldNotOwnedWarning = streamReadBool(streamId)
222     spec.showRestrictedZoneWarning = streamReadBool(streamId)
223 end
224 end
225 end
```

onWriteUpdateStream

Description

Called on on update

Definition

onWriteUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer	streamId	stream ID
table	connection	connection
integer	dirtyMask	dirty mask

Code

```
232 function TreePlanter:onWriteUpdateStream(streamId, connection, dirtyMask)
233     if not connection:getIsServer() then
234         local spec = self.spec_treePlanter
235         if streamWriteBool(streamId, bitAND(dirtyMask, spec.dirtyFlag) ~= 0) then
236             streamWriteBool(streamId, spec.hasGroundContact)
237             streamWriteBool(streamId, spec.showFieldNotOwnedWarning)
238             streamWriteBool(streamId, spec.showRestrictedZoneWarning)
239         end
240     end
241 end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
248 function TreePlanter:onUpdate(dt, isActiveForInput, isSelected)
249     local spec = self.spec_treePlanter
250 
251     if self.firstTimeRun then
```

```
252 local pallet
253 if spec.palletIdToMount ~= nil then
254 pallet = NetworkUtil.getObject(spec.palletIdToMount)
255 elseif spec.palletHadBeenMounted then
256 spec.palletHadBeenMounted = nil
257 pallet = TreePlanter.getSaplingPalletInRange(self,
    spec.saplingPalletMountNode, spec.palletsInTrigger)
258 end
259
260 if pallet ~= nil then
261 pallet:mount(self, spec.saplingPalletMountNode, 0,0,0, 0,0,0)
262
263 local fillUnitIndex = 1
264 local fillType = pallet:getFillUnitFillType(fillUnitIndex)
265 local fillLevel = pallet:getFillUnitFillLevel(fillUnitIndex)
266 local capacity = pallet:getFillUnitCapacity(fillUnitIndex)
267 if fillType ~= FillType.UNKNOWN and fillLevel > 0 then
268 self:setFillUnitCapacity(spec.fillUnitIndex, capacity)
269
270 if self.isServer then
271 self:addFillUnitFillLevel(self:getOwnerFarmId(),
    spec.fillUnitIndex, -math.huge, fillType, ToolType.UNDEFINED)
272 self:addFillUnitFillLevel(self:getOwnerFarmId(),
    spec.fillUnitIndex, fillLevel, fillType, ToolType.UNDEFINED)
273 end
274 end
275
276 spec.mountedSaplingPallet = pallet
277 g_currentMission:removeActivatableObject(spec.activatable)
278 spec.palletIdToMount = nil
279 end
280 end
281
282 if self.isClient then
283 local nearestSaplingPallet = nil
284 if spec.mountedSaplingPallet == nil then
285 nearestSaplingPallet = TreePlanter.getSaplingPalletInRange(self,
    spec.saplingPalletGrabNode, spec.palletsInTrigger)
286 end
287
288 if spec.nearestSaplingPallet ~= nearestSaplingPallet then
```

```
289 spec.nearestSaplingPallet = nearestSaplingPallet
290
291 if nearestSaplingPallet ~= nil then
292 g_currentMission:addActivatableObject(spec.activatable)
293 else
294 g_currentMission:removeActivatableObject(spec.activatable)
295 end
296 end
297 end
298
299 if spec.mountedSaplingPallet ~= nil then
300 spec.mountedSaplingPallet:raiseActive()
301 end
302 end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
309 function TreePlanter:onUpdateTick(dt, isActiveForInput, isSelected)
310 local spec = self.spec_treePlanter
311
312 spec.showTooManyTreesWarning = false
313 local showFieldNotOwnedWarning = false
314 local showRestrictedZoneWarning = false
315
316 if self.isServer then
317 local hasGroundContact = false
318 if spec.groundReferenceNode ~= nil then
319 hasGroundContact =
    self:getIsGroundReferenceNodeActive(spec.groundReferenceNode)
320 end
321
322 if spec.hasGroundContact ~= hasGroundContact then
323 self:raiseDirtyFlags(spec.dirtyFlag)
324 spec.hasGroundContact = hasGroundContact
```

```
325 end
326 end
327
328 if self:getIsAIActive() then
329 if not g_currentMission.missionInfo.helperBuySeeds then
330 if spec.mountedSaplingPallet == nil then
331 local rootVehicle = self:getRootVehicle()
332 rootVehicle:stopAIVehicle(AIVehicle.STOP_REASON_OUT_OF_FILL)
333 end
334 end
335 end
336
337 if spec.hasGroundContact then
338 if self:getIsTurnedOn() then
339 if self.isServer then
340 local fillLevel = self:getFillUnitFillLevel(spec.fillUnitIndex)
341 local fillType = self:getFillUnitFillType(spec.fillUnitIndex)
342
343 if g_currentMission.missionInfo.helperBuySeeds then
344 if self:getIsAIActive() then
345 fillType = FillType.POPLAR
346 end
347 end
348
349 if fillLevel > 0 or g_currentMission.missionInfo.helperBuySeeds then
350 if fillType == FillType.TREESAPLINGS then
351 if self:getLastSpeed() > 1 then
352 local x,y,z = getWorldTranslation(spec.node)
353 if g_currentMission.accessHandler:canFarmAccessLand(self:getActiveFarm()
    z) then
354 if not PlacementUtil.isInsideRestrictedZone(g_currentMission.restrictedZ
    nil, x, y, z) then
355 if y > g_currentMission.waterY then
356 if spec.lastTreePos ~= nil then
357 local distance = MathUtil.vector3Length(x-spec.lastTreePos[1], y-
    spec.lastTreePos[2], z-spec.lastTreePos[3])
358 if distance > spec.minDistance then
359 self:createTree()
360 end
361 else
362 self:createTree()
```

```
363 end
364 end
365 else
366 showRestrictedZoneWarning = true
367 end
368 else
369 showFieldNotOwnedWarning = true
370 end
371 end
372 elseif fillType ~= FillType.UNKNOWN then
373 local x,_,z = getWorldTranslation(spec.node)
374 if g_currentMission.accessHandler:canFarmAccessLand(g_currentMission:getFar
    x, z) then
375 local width = math.sqrt( g_currentMission:getFruitPixelsToSqm() ) * 0.5
376
377 local sx,_,sz = localToWorld(spec.node, -width,0,width)
378 local wx,_,wz = localToWorld(spec.node, width,0,width)
379 local hx,_,hz = localToWorld(spec.node, -width,0,3*width)
380
381 local fruitType =
    g_fruitTypeManager:getFruitTypeIndexByFillTypeIndex(fillType)
382 local fruitDesc = g_fruitTypeManager:getFruitTypeByIndex(fruitType)
383
384 local dx,_,dz = localDirectionToWorld(spec.node, 0, 0, 1)
385 local angleRad = MathUtil.getYRotationFromDirection(dx, dz)
386 if fruitDesc ~= nil and fruitDesc.directionSnapAngle ~= 0 then
387 angleRad = math.floor(angleRad / fruitDesc.directionSnapAngle + 0.5) *
    fruitDesc.directionSnapAngle
388 end
389 local angle = FSDensityMapUtil.convertToDensityMapAngle(angleRad,
    g_currentMission.terrainDetailAngleMaxValue)
390
391 -- cultivate
392 local limitToField = spec.limitToField or spec.forceLimitToField
393 local limitGrassDestructionToField = spec.limitToField or
    spec.forceLimitToField
394 FSDensityMapUtil.updateCultivatorArea(sx,sz, wx,wz, hx,hz, not limitToFi
    not limitGrassDestructionToField, angle, nil)
395 FSDensityMapUtil.eraseTireTrack(sx,sz, wx,wz, hx,hz)
396
397 -- plant, shift area
```

```
398 sx,_,sz = localToWorld(spec.node, -width,0,-3*width)
399 wx,_,wz = localToWorld(spec.node, width,0,-3*width)
400 hx,_,hz = localToWorld(spec.node, -width,0,-width)
401 local area, _ = FSDensityMapUtil.updateSowingArea(fruitType, sx,sz, wx,w
    hx,hz, angle, 2)
402
403 local usage = fruitDesc.seedUsagePerSqm * area
404
405 local stats = g_farmManager:getFarmById(self:getActiveFarm()).stats
406 if self:getIsAIActive() and g_currentMission.missionInfo.helperBuySeeds
407 local price = usage *
    g_currentMission.economyManager:getCostPerLiter(FillType.SEEDS, false) *
    - increase price if AI is active to reward the player's manual work
408 stats:updateStats("expenses", price)
409 g_currentMission:addMoney(-price, self:getActiveFarm(), "purchaseSeeds")
410 else
411 self:addFillUnitFillLevel(self:getOwnerFarmId(), spec.fillUnitIndex, -us
    fillType, ToolType.UNDEFINED)
412 end
413
414 local lastHa = MathUtil.areaToHa(area, g_currentMission:getFruitPixelsTo
415 stats:updateStats("seedUsage", usage) -- TODO(JK): this is incorrect. de
    on controlling/ai
416 stats:updateStats("sownHectares", lastHa)
417 stats:updateStats("sownTime", dt/(1000*60))
418 stats:updateStats("workedHectares", lastHa)
419 stats:updateStats("workedTime", dt/(1000*60))
420 else
421 showFieldNotOwnedWarning = true
422 end
423 end
424
425 end
426 end
427 end
428 end
429
430 if self.isServer then
431 if spec.showFieldNotOwnedWarning ~= showFieldNotOwnedWarning or
    spec.showRestrictedZoneWarning ~= showRestrictedZoneWarning then
432 spec.showFieldNotOwnedWarning = showFieldNotOwnedWarning
433 spec.showRestrictedZoneWarning = showRestrictedZoneWarning
```

```
434 self:raiseDirtyFlags(spec.dirtyFlag)
435 end
436 end
437
438 if self.isClient then
439 if self:getIsTurnedOn() and spec.hasGroundContact and self:getLastSpeed
    then
440 if not spec.isWorkSamplePlaying then
441 g_soundManager:playSample(spec.samples.work)
442 spec.isWorkSamplePlaying = true
443 end
444 else
445 if spec.isWorkSamplePlaying then
446 g_soundManager:stopSample(spec.samples.work)
447 spec.isWorkSamplePlaying = false
448 end
449 end
450
451 local actionEvent = spec.actionEvents[InputAction.IMPLEMENT_EXTRA3]
452 if actionEvent ~= nil then
453 local showAction = false
454
455 if isActiveForInput then
456 local fillType = self:getFillUnitFillType(spec.fillUnitIndex)
457 if fillType ~= FillType.UNKNOWN and fillType ~= FillType.TREESAPLINGS th
458 if g_currentMission:getHasPlayerPermission("createFields", self:getOwner
    then
459 if not spec.forceLimitToField then
460 showAction = true
461 end
462 end
463 end
464
465 if showAction then
466 if spec.limitToField then
467 g_inputBinding:setActionEventText(actionEvent.actionEventId,
    g_i18n:getText("action_allowCreateFields"))
468 else
469 g_inputBinding:setActionEventText(actionEvent.actionEventId,
    g_i18n:getText("action_limitToFields"))
470 end
```

```
471 end
472 end
473
474 g_inputBinding:setActionEventActive(actionEvent.actionEventId, showActio
475 end
476 end
477 end
```

onDraw

Description

Called on draw

Definition

onDraw(boolean isActiveForInput, boolean isSelected)

Arguments

boolean isActiveForInput	true if vehicle is active for input
boolean isSelected	true if vehicle is selected

Code

```
483 function TreePlanter:onDraw(isActiveForInput, isSelected)
484 local spec = self.spec_treePlanter
485
486 if isActiveForInput then
487 if self:getFillUnitFillLevel(spec.fillUnitIndex) <= 0 then
488 g_currentMission:addExtraPrintText(g_i18n:getText("info_firstFillTheTool
489 end
490 end
491
492 if spec.showFieldNotOwnedWarning then
493 g_currentMission:showBlinkingWarning(g_i18n:getText("warning_youDontHave
494 end
495
496 if spec.showRestrictedZoneWarning then
497 g_currentMission:showBlinkingWarning(g_i18n:getText("warning_actionNotAl
498 end
499
500 if spec.showTooManyTreesWarning then
501 g_currentMission:showBlinkingWarning(g_i18n:getText("warning_tooManyTree
502 end
503 end
```

onTurnedOn

Description

Called on turn off

Definition

onTurnedOn(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
508  function TreePlanter:onTurnedOn()
509  if self.isClient then
510  local spec = self.spec_treePlanter
511  g_animationManager:startAnimations(spec.animationNodes)
512  end
513  end
```

onTurnedOff

Description

Called on turn off

Definition

onTurnedOff(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
518  function TreePlanter:onTurnedOff()
519  if self.isClient then
520  local spec = self.spec_treePlanter
521  g_animationManager:stopAnimations(spec.animationNodes)
522  g_soundManager:stopSamples(spec.samples)
523  spec.isWorkSamplePlaying = false
524  end
525  end
```

onFillUnitFillLevelChanged

Description

Set unit fill level

Definition

onFillUnitFillLevelChanged(integer fillUnitIndex, float fillLevel, integer fillType, boolean force, table fillPositionData)

Arguments

integer	fillUnitIndex	index of fill unit
float	fillLevel	new fill level
integer	fillType	fill type
boolean	force	force action
table	fillPositionData	fill info for fill volume

Code

```
534  function TreePlanter:onFillUnitFillLevelChanged(fillUnitIndex,
     fillLevelDelta, fillType, toolType, fillPositionData,
     appliedDelta)
535  local spec = self.spec_treePlanter
```

```
536 local object = spec.mountedSaplingPallet
537
538 if object ~= nil and fillLevelDelta < 0 then
539 local fillUnits = object:getFillUnits()
540 for objectFillUnitIndex, _ in pairs(fillUnits) do
541 if object:getFillUnitFillType(fillUnitIndex) == fillType then
542 object:addFillUnitFillLevel(self:getOwnerFarmId(),
    objectFillUnitIndex, fillLevelDelta, fillType,
    ToolType.UNDEFINED)
543 --delete pallet if it is empty
544 if object:getFillUnitFillLevel(objectFillUnitIndex) <= 0.5 then -
    - use threshold instead if int to avoid floating point issues
545 spec.palletsInTrigger[object] = nil
546 g_currentMission:removeVehicle(object);
547 spec.mountedSaplingPallet = nil
548 end
549
550 break;
551 end
552 end
553 end
554 end
```

setPlantLimitToField

Description

Set plant limit to field state

Definition

setPlantLimitToField(boolean plantLimitToField, boolean noEventSend)

Arguments

boolean plantLimitToField	plant limit to field state
boolean noEventSend	no event send

Code

```
560 function TreePlanter:setPlantLimitToField(plantLimitToField,
    noEventSend)
561 local spec = self.spec_treePlanter
562
563 if spec.limitToField ~= plantLimitToField then
564 spec.limitToField = plantLimitToField
565
566 PlantLimitToFieldEvent.sendEvent(self, plantLimitToField,
    noEventSend)
567 end
568 end
```

createTree

Description

Create tree on current position

Definition

createTree()

Code

```
572 function TreePlanter:createTree()
573 local spec = self.spec_treePlanter
574
575 if not g_treePlantManager:canPlantTree() then
576 spec.showTooManyTreesWarning = true
577 return
578 end
579
580 if self.isServer and spec.mountedSaplingPallet ~= nil then
581 local x,y,z = getWorldTranslation(spec.node)
582 local yRot = math.random() * 2*math.pi
583
584 g_treePlantManager:plantTree(1, x, y, z, 0, yRot, 0, 0)
585 spec.lastTreePos = {x,y,z}
586
587 local stats = g_farmManager:getFarmById(self:getActiveFarm()).stats
588
589 if g_currentMission.missionInfo.helperBuySeeds and self:getIsAIActive()
590 local storeItem =
    g_storeManager:getItemByXMLFilename(spec.mountedSaplingPallet.configFile
591 local pricePerSapling = 1.5 * (storeItem.price /
    spec.mountedSaplingPallet:getCapacity())
592
593 stats:updateStats("expenses", pricePerSapling)
594 g_currentMission:addMoney(-pricePerSapling, self:getActiveFarm(),
    "purchaseSeeds")
595 else
596 -- use 0.9999 instead of 1 to compansate float precision on mp sync
597 self:addFillUnitFillLevel(self:getOwnerFarmId(), spec.fillUnitIndex, -0.
    self:getFillUnitFillType(spec.fillUnitIndex), ToolType.UNDEFINED)
598 end
599
600 -- increase tree plant counter for achievements
601 stats:updateStats("plantedTreeCount", 1)
602 end
```

```
603 end
```

loadPallet

Description

Called on loading

Definition

loadPallet(table savegame)

Arguments

table savegame savegame

Code

```
608 function TreePlanter:loadPallet(palletObjectId, noEventSend)
609 local spec = self.spec_treePlanter
610
611 TreePlanterLoadPalletEvent.sendEvent(self, palletObjectId,
    noEventSend)
612
613 spec.palletIdToMount = palletObjectId
614 end
```

getDirtMultiplier

Description

Returns current dirt multiplier

Definition

getDirtMultiplier()

Return Values

float dirtMultiplier current dirt multiplier

Code

```
619 function TreePlanter:getDirtMultiplier(superFunc)
620 local multiplier = superFunc(self)
621
622 local spec = self.spec_treePlanter
623 if spec.hasGroundContact then
624 multiplier = multiplier + self:getWorkDirtMultiplier() *
    self:getLastSpeed() / self.speedLimit
625 end
626
627 return multiplier
628 end
```

getWearMultiplier

Description

Returns current wear multiplier

Definition

getWearMultiplier()

Return Values

float dirtMultiplier current wear multiplier

Code

```
633  function TreePlanter:getWearMultiplier(superFunc)
634  local multiplier = superFunc(self)
635
636  local spec = self.spec_treePlanter
637  if spec.hasGroundContact then
638  multiplier = multiplier + self:getWorkWearMultiplier() *
     self:getLastSpeed() / self.speedLimit
639  end
640
641  return multiplier
642  end
```

getIsSpeedRotatingPartActive

Description

Returns true if speed rotating part is active

Definition

getIsSpeedRotatingPartActive(table speedRotatingPart)

Arguments

table speedRotatingPart speedRotatingPart

Return Values

boolean isActive speed rotating part is active

Code

```
648  function TreePlanter:getIsSpeedRotatingPartActive(superFunc,
     speedRotatingPart)
649  local spec = self.spec_treePlanter
650
651  if not spec.hasGroundContact then
652  return false
653  end
654
655  return superFunc(self, speedRotatingPart)
656  end
```

doCheckSpeedLimit

Description

Returns if speed limit should be checked

Definition

doCheckSpeedLimit()

Return Values

boolean checkSpeedlimit check speed limit

Code

```
674  function TreePlanter:doCheckSpeedLimit(superFunc)
```

```
675  return superFunc(self) or (self:getIsTurnedOn() and
     self:getIsImplementChainLowered())
676  end
```

getDefaultSpeedLimit

Description

Returns default speed limit

Definition

getDefaultSpeedLimit()

Return Values

float speedLimit speed limit

Code

```
770  function TreePlanter.getDefaultSpeedLimit()
771  return 5
772  end
```

getSaplingPalletInRange

Description

Returns nearest sapling pallet in range

Definition

getSaplingPalletInRange(integer refNode)

Arguments

integer refNode id of reference node

Return Values

table object object of sapling pallet

Code

```
778  function TreePlanter.getSaplingPalletInRange(self, refNode,
     palletsInTrigger)
779  local spec = self.spec_treePlanter
780
781  local nearestDistance = spec.nearestPalletDistance
782  local nearestSaplingPallet = nil
783
784  for object, state in pairs(palletsInTrigger) do
785  if state ~= nil and state > 0 then
786
787  if object ~= spec.mountedSaplingPallet then
788
789  local distance = calcDistanceFrom(refNode, object.rootNode)
790  if distance < nearestDistance then
791  local validPallet = false
792
793  local fillUnits = object:getFillUnits()
794  for fillUnitIndex, _ in pairs(fillUnits) do
```

```
795  local filltype = object:getFillUnitFillType(fillUnitIndex)
796  if filltype ~= FillType.UNKNOWN then
797  if self:getFillUnitSupportsFillType(spec.fillUnitIndex, filltype)
     then
798  if object:getFillUnitFillLevel(fillUnitIndex) > 0 then
799  validPallet = true
800  break
801  end
802  end
803  end
804  end
805
806  if validPallet then
807  nearestSaplingPallet = object
808  end
809  end
810
811  end
812  end
813  end
814  return nearestSaplingPallet
815  end
```

Activatable:new

Description

Returns new instance of class

Definition

Activatable:new(table treePlanter)

Arguments

table treePlanter object of treePlanter

Return Values

table self new instance

Code

```
838  function TreePlanterActivatable:new(treePlanterVehicle)
839  local self = {}
840  setmetatable(self, TreePlanterActivatable_mt)
841
842  self.treePlanterVehicle = treePlanterVehicle
843  self.activateText =
     string.format(g_i18n:getText("action_refillOBJECT"),
     self.treePlanterVehicle.typeDesc)
844
845  return self
```

```
846 end
```

Activatable:getIsActivatable

Description

Returns if is activateable

Definition

Activatable:getIsActivatable()

Return Values

boolean isActivateable is activateable

Code

```
851 function TreePlanterActivatable:getIsActivatable()
852 if self.treePlanterVehicle:getRootVehicle() ~= g_currentMission.controlledVehicle then
853 return false
854 end
855
856 if self.treePlanterVehicle.spec_treePlanter.mountedSaplingPallet == nil and self.treePlanterVehicle.spec_treePlanter.nearestSaplingPallet ~= nil then
857 return true
858 end
859 return false
860 end
```

Activatable:onActivateObject

Description

Called on activate object

Definition

Activatable:onActivateObject()

Code

```
864 function TreePlanterActivatable:onActivateObject()
865 self.treePlanterVehicle:loadPallet(NetworkUtil.getObjectId(self.treePlar
866 end
```

TreeSaw

Description

This is the specialization for all tree saws

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17  function TreeSaw.prerequisitesPresent(specializations)
18  return SpecializationUtil.hasSpecialization(TurnOnVehicle,
    specializations)
19  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
37  function TreeSaw:onLoad(savegame)
38  local spec = self.spec_treeSaw
39
40  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.turnedOnRotationNodes.turnedOnRotationNode",
    "vehicle.treeSaw.animationNodes.animationNode", "stumbCutter") --
    FS17 to FS19
41  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName,
    "vehicle.treeSaw.cutParticleSystems.emitterShape(0)",
    "vehicle.treeSaw.effects.effectNode") --FS17 to FS19
42  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.treeSaw.sawSound",
    "vehicle.treeSaw.sounds.saw") --FS17 to FS19
43  XMLUtil.checkDeprecatedXMLElements(self.xmlFile,
    self.configFileName, "vehicle.treeSaw.cutSound",
    "vehicle.treeSaw.sounds.cut") --FS17 to FS19
44
45  local baseKey = "vehicle.treeSaw"
46
47  if self.isClient then
48  spec.animationNodes =
    g_animationManager:loadAnimations(self.xmlFile,
    baseKey..".animationNodes", self.components, self,
    self.i3dMappings)
49  spec.effects = g_effectManager:loadEffect(self.xmlFile, baseKey ..
    ".effects", self.components, self, self.i3dMappings)
50
51  spec.samples = {}
```

```
52  spec.samples.cut = g_soundManager:loadSampleFromXML(self.xmlFile,
    baseKey .. ".sounds", "cut", self.baseDirectory, self.components,
    0, AudioGroup.VEHICLE, self.i3dMappings, self)
53  spec.samples.saw = g_soundManager:loadSampleFromXML(self.xmlFile,
    baseKey .. ".sounds", "saw", self.baseDirectory, self.components,
    0, AudioGroup.VEHICLE, self.i3dMappings, self)
54  end
55
56  spec.cutNode = I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, baseKey .. ".cutNode#node"),
    self.i3dMappings)
57  spec.cutSizeY = Utils.getNoNil(getXMLFloat(self.xmlFile, baseKey
    .. ".cutNode#sizeY"), 1)
58  spec.cutSizeZ = Utils.getNoNil(getXMLFloat(self.xmlFile, baseKey
    .. ".cutNode#sizeZ"), 1)
59  spec.lengthAboveThreshold =
    Utils.getNoNil(getXMLFloat(self.xmlFile, baseKey ..
    ".cutNode#lengthAboveThreshold"), 0.3)
60  spec.lengthBelowThreshold =
    Utils.getNoNil(getXMLFloat(self.xmlFile, baseKey ..
    ".cutNode#lengthBelowThreshold"), 0.3)
61  spec.cutTimerDuration = Utils.getNoNil(getXMLFloat(self.xmlFile,
    baseKey .. ".cutNode#timer"), 1)*1000
62
63  spec.curSplitShape = nil
64  spec.cutTimer = -1
65  spec.isCutting = false
66  spec.warnTreeNotOwned = false
67  end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
71  function TreeSaw:onDelete()
72  if self.isClient then
73  local spec = self.spec_treeSaw
74  g_effectManager:deleteEffects(spec.effects)
75  g_soundManager:deleteSamples(spec.samples)
76  g_animationManager:deleteAnimations(spec.animationNodes)
77  end
78  end
```

onReadUpdateStream

Description

Called on on update

Definition

onReadUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId	stream ID
integer timestamp	timestamp
table	connection connection

Code

```
85  function TreeSaw:onReadUpdateStream(streamId, timestamp,
    connection)
86  if connection:getIsServer() then
87  local spec = self.spec_treeSaw
88  spec.isCutting = streamReadBool(streamId)
89  end
90  end
```

onWriteUpdateStream

Description

Called on on update

Definition

onWriteUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Code

```
 97  function TreeSaw:onWriteUpdateStream(streamId, connection,
     dirtyMask)
 98  if not connection:getIsServer() then
 99  local spec = self.spec_treeSaw
100  streamWriteBool(streamId, spec.isCutting)
101  end
102  end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
function TreeSaw:onUpdate(dt, isActiveForInput, isSelected)
local spec = self.spec_treeSaw

-- Verify that the split shapes still exist (possible that someone has cut them)
if self.isServer then
if spec.curSplitShape ~= nil then
if not entityExists(spec.curSplitShape) then
spec.curSplitShape = nil
if g_server ~= nil then
spec.cutTimer = -1
end
end
end

spec.isCutting = spec.curSplitShape ~= nil
if spec.curSplitShape ~= nil then
if spec.cutTimer > 0 then
spec.cutTimer = math.max(spec.cutTimer - dt, 0)
end

-- cut
if spec.cutTimer == 0 then
spec.cutTimer = -1

local x,y,z = getWorldTranslation(spec.cutNode)
local nx,ny,nz = localDirectionToWorld(spec.cutNode, 1,0,0)
local yx,yy,yz = localDirectionToWorld(spec.cutNode, 0,1,0)

ChainsawUtil.cutSplitShape(spec.curSplitShape, x,y,z, nx,ny,nz, yx,yy,yz, spec.cutSizeY, spec.cutSizeZ, self:getActiveFarm())
spec.curSplitShape = nil
end
end
end

-- effect and sound for cut
if self.isClient then
if spec.cutTimer > 0 then
g_effectManager:setFillType(spec.effects, FillType.WOODCHIPS)
g_effectManager:startEffects(spec.effects)
```

```
148 if not g_soundManager:getIsSamplePlaying(spec.samples.cut) then
149 g_soundManager:playSample(spec.samples.cut)
150 end
151 else
152 g_effectManager:stopEffects(spec.effects)
153 if g_soundManager:getIsSamplePlaying(spec.samples.cut) then
154 g_soundManager:stopSample(spec.samples.cut)
155 end
156 end
157 end
158 end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
165 function TreeSaw:onUpdateTick(dt, isActiveForInput, isSelected)
166 local spec = self.spec_treeSaw
167 spec.warnTreeNotOwned = false
168
169 if self:getIsTurnedOn() then
170 if spec.cutNode ~= nil then
171 local x,y,z = getWorldTranslation(spec.cutNode)
172 local nx,ny,nz = localDirectionToWorld(spec.cutNode, 1,0,0)
173 local yx,yy,yz = localDirectionToWorld(spec.cutNode, 0,1,0)
174
175 if spec.curSplitShape == nil then
176 local shape, _, _, _, _ = findSplitShape(x,y,z, nx,ny,nz, yx,yy,yz,
    spec.cutSizeY, spec.cutSizeZ)
177 if shape ~= 0 then
178 if
    g_currentMission.accessHandler:canFarmAccessLand(self:getActiveFarm(),
    x, z) then
179 spec.curSplitShape = shape
180 spec.cutTimer = spec.cutTimerDuration
181 else
182 spec.warnTreeNotOwned = true
```

```
183 end
184 end
185 end
186
187 if spec.curSplitShape ~= nil then
188 local minY,maxY, minZ,maxZ = testSplitShape(spec.curSplitShape, x,y,z,
    nx,ny,nz, yx,yy,yz, spec.cutSizeY, spec.cutSizeZ)
189 if minY == nil then
190 spec.curSplitShape = nil
191 else
192 -- check if cut would be below y=0 (tree CoSy)
193 local cutTooLow = false
194 local _,y,_ = localToLocal(spec.cutNode, spec.curSplitShape,
    0,minY,minZ)
195 cutTooLow = cutTooLow or y < 0.01
196 local _,y,_ = localToLocal(spec.cutNode, spec.curSplitShape,
    0,minY,maxZ)
197 cutTooLow = cutTooLow or y < 0.01
198 local _,y,_ = localToLocal(spec.cutNode, spec.curSplitShape,
    0,maxY,minZ)
199 cutTooLow = cutTooLow or y < 0.01
200 local _,y,_ = localToLocal(spec.cutNode, spec.curSplitShape,
    0,maxY,maxZ)
201 cutTooLow = cutTooLow or y < 0.01
202 if cutTooLow then
203 spec.curSplitShape = nil
204 end
205 end
206 end
207
208 if spec.curSplitShape ~= nil then
209 local lenBelow, lenAbove =
    getSplitShapePlaneExtents(spec.curSplitShape, x,y,z, nx,ny,nz)
210 if lenAbove < spec.lengthAboveThreshold or lenBelow <
    spec.lengthBelowThreshold then
211 spec.curSplitShape = nil
212 end
213 end
214
215 if spec.curSplitShape == nil and spec.cutTimer > -1 then
216 spec.cutTimer = -1
217 end
```

```
218 end
219 end
220 end
```

onDraw

Description

Called on draw

Definition

onDraw(boolean isActiveForInput, boolean isSelected)

Arguments

boolean isActiveForInput	true if vehicle is active for input
boolean isSelected	true if vehicle is selected

Code

```
226 function TreeSaw:onDraw(isActiveForInput, isSelected)
227 local spec = self.spec_treeSaw
228
229 if spec.isCutting then
230 g_currentMission:addExtraPrintText(g_i18n:getText("info_cutting"))
231 end
232
233 if spec.warnTreeNotOwned then
234 g_currentMission:showBlinkingWarning(g_i18n:getText("warning_youDontHave
    1000)
235 end
236 end
```

onDeactivate

Description

Called on deactivate

Definition

onDeactivate()

Code

```
240 function TreeSaw:onDeactivate()
241 local spec = self.spec_treeSaw
242
243 spec.curSplitShape = nil
244 spec.cutTimer = -1
245 end
```

onTurnedOn

Description

Called on turn on

Definition

onTurnedOn()

Code

```
249 function TreeSaw:onTurnedOn()
250 if self.isClient then
251 local spec = self.spec_treeSaw
252 g_animationManager:startAnimations(spec.animationNodes)
253 g_soundManager:playSample(spec.samples.saw)
254 end
255 end
```

onTurnedOff

Description

Called on turn off

Definition

onTurnedOff()

Code

```
259 function TreeSaw:onTurnedOff()
260 local spec = self.spec_treeSaw
261
262 spec.curSplitShape = nil
263 spec.cutTimer = -1
264
265 if self.isClient then
266 g_animationManager:stopAnimations(spec.animationNodes)
267 g_effectManager:stopEffects(spec.effects)
268 g_soundManager:stopSamples(spec.samples)
269 end
270 end
```

TurnOnVehicle

Description

Class for all vehicles that can be turned on

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
74 function TurnOnVehicle:onLoad(savegame)
75
76 XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
   "vehicle.turnOnSettings#turnOffText", "vehicle.turnOnVehicle#turnOffText
   to FS17
```

```
77  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.turnOnSettings#turnOnText", "vehicle.turnOnVehicle#turnOnText")
    FS17
78  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.turnOnSettings#needsSelection") --FS15 to FS17
79  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.turnOnSettings#isAlwaysTurnedOn",
    "vehicle.turnOnVehicle#isAlwaysTurnedOn") --FS15 to FS17
80  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.turnOnSettings#toggleButton", "vehicle.turnOnVehicle#toggleButt
    to FS17
81  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.turnOnSettings#animationName",
    "vehicle.turnOnVehicle.turnedAnimation#name") --FS15 to FS17
82  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.turnOnSettings#turnOnSpeedScale",
    "vehicle.turnOnVehicle.turnedAnimation#turnOnSpeedScale") --FS15 to FS1
83  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.turnOnSettings#turnOffSpeedScale",
    "vehicle.turnOnVehicle.turnedAnimation#turnOffSpeedScale") --FS15 to FS
84  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.turnedOnRotationNodes.turnedOnRotationNode#type",
    "vehicle.turnOnVehicle.animationNodes.animationNode", "turnOn") --FS17 t
85  XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName,
    "vehicle.foldable.foldingParts#turnOffOnFold",
    "vehicle.turnOnVehicle#turnOffIfNotAllowed") --FS17 to FS19
86
87  local spec = self.spec_turnOnVehicle
88
89  local turnOnButtonStr = getXMLString(self.xmlFile,
    "vehicle.turnOnVehicle#toggleButton")
90  if turnOnButtonStr ~= nil then
91  spec.toggleTurnOnInputBinding = InputAction[turnOnButtonStr]
92  end
93  spec.toggleTurnOnInputBinding = Utils.getNoNil(spec.toggleTurnOnInputBin
    InputAction.IMPLEMENT_EXTRA)
94
95  spec.turnOffText = Utils.getNoNil(getXMLString(self.xmlFile,
    "vehicle.turnOnVehicle#turnOffText"), "action_turnOffOBJECT")
96  spec.turnOnText = Utils.getNoNil(getXMLString(self.xmlFile,
    "vehicle.turnOnVehicle#turnOnText"), "action_turnOnOBJECT")
97  spec.isTurnedOn = false
98  spec.isAlwaysTurnedOn = Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.turnOnVehicle#isAlwaysTurnedOn"), false)
99  spec.turnedOnByAttacherVehicle = Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.turnOnVehicle#turnedOnByAttacherVehicle"), false)
```

```
100 spec.turnOffIfNotAllowed = Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.turnOnVehicle#turnOffIfNotAllowed"), false)
101 spec.motorNotStartedWarning =
    string.format(g_i18n:getText(Utils.getNoNil(getXMLString(self.xmlFile,
    "vehicle.turnOnVehicle#motorNotStartedWarning"), "warning_motorNotStarte
    self.typeDesc)
102
103 spec.aiRequiresTurnOn = Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.turnOnVehicle#aiRequiresTurnOn"), true)
104
105 if self.isClient then
106 spec.animationNodes = g_animationManager:loadAnimations(self.xmlFile,
    "vehicle.turnOnVehicle.animationNodes", self.components, self, self.i3dN
107
108 local turnOnAnimation = getXMLString(self.xmlFile,
    "vehicle.turnOnVehicle.turnedAnimation#name")
109 if turnOnAnimation ~= nil then
110 spec.turnOnAnimation = {}
111 spec.turnOnAnimation.name = turnOnAnimation
112 spec.turnOnAnimation.turnOnSpeedScale = Utils.getNoNil(getXMLFloat(self.
    "vehicle.turnOnVehicle.turnedAnimation#turnOnSpeedScale"), 1)
113 spec.turnOnAnimation.turnOffSpeedScale = Utils.getNoNil(getXMLFloat(self
    "vehicle.turnOnVehicle.turnedAnimation#turnOffSpeedScale"), -
    spec.turnOnAnimation.turnOnSpeedScale)
114 end
115
116 spec.turnedOnAnimations = {}
117 local i = 0
118 while true do
119 local baseKey = string.format("vehicle.turnOnVehicle.turnedOnAnimation(%
120 if not hasXMLProperty(self.xmlFile, baseKey) then
121 break
122 end
123
124 local entry = {}
125 local name = getXMLString(self.xmlFile, baseKey.."#name")
126 if name ~= nil then
127 entry.name = name
128 entry.turnOnFadeTime = Utils.getNoNil(getXMLFloat(self.xmlFile,
    baseKey.."#turnOnFadeTime"), 1) * 1000
129 entry.turnOffFadeTime = Utils.getNoNil(getXMLFloat(self.xmlFile,
    baseKey.."#turnOffFadeTime"), 1) * 1000
```

```
130 entry.speedScale = Utils.getNoNil(getXMLFloat(self.xmlFile,
    baseKey.."#speedScale"), 1)
131 entry.speedDirection = 0
132 entry.currentSpeed = 0
133
134 table.insert(spec.turnedOnAnimations, entry)
135 end
136
137 i = i + 1
138 end
139
140 spec.activatableFillUnits = {}
141 local i = 0
142 while true do
143 local key =
    string.format("vehicle.turnOnVehicle.activatableFillUnits.activatableFil
    i)
144 if not hasXMLProperty(self.xmlFile, key) then
145 break
146 end
147
148 local fillUnitIndex = getXMLInt(self.xmlFile, key.."#index")
149 if fillUnitIndex ~= nil then
150 spec.activatableFillUnits[fillUnitIndex] = true
151 end
152
153 i = i + 1
154 end
155
156 spec.samples = {}
157 spec.samples.start = g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.turnOnVehicle.sounds", "start", self.baseDirectory, self.compon
    AudioGroup.VEHICLE, self.i3dMappings, self)
158 spec.samples.stop = g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.turnOnVehicle.sounds", "stop", self.baseDirectory, self.compone
    AudioGroup.VEHICLE, self.i3dMappings, self)
159 spec.samples.work = g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.turnOnVehicle.sounds", "work", self.baseDirectory, self.compone
    AudioGroup.VEHICLE, self.i3dMappings, self)
160 end
161 end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
function TurnOnVehicle:onDelete()
if self.isClient then
local spec = self.spec_turnOnVehicle
g_soundManager:deleteSamples(spec.samples)
g_animationManager:deleteAnimations(spec.animationNodes)
end
end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
function TurnOnVehicle:onReadStream(streamId, connection)
local turnedOn = streamReadBool(streamId)
self:setIsTurnedOn(turnedOn, true)
end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer streamId streamId

integer connection connection

Code

```
function TurnOnVehicle:onWriteStream(streamId, connection)
local spec = self.spec_turnOnVehicle
streamWriteBool(streamId, spec.isTurnedOn)
end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
196 function TurnOnVehicle:onUpdate(dt, isActiveForInput, isSelected)
197 local spec = self.spec_turnOnVehicle
198
199 local isTurnedOn = self:getIsTurnedOn()
200
201 if self.isClient then
202 if not spec.isAlwaysTurnedOn and not
    spec.turnedOnByAttacherVehicle then
203 -- update activity of actionEvent
204 if spec.actionEvents ~= nil then
205 local actionEvent =
    spec.actionEvents[spec.toggleTurnOnInputBinding]
206 if actionEvent ~= nil and actionEvent.actionEventId ~= nil then
207 local state = self:getCanToggleTurnedOn()
208
209 if state then
210 local text
211 if isTurnedOn then
212 text = string.format(g_i18n:getText("action_turnOffOBJECT"),
    self.typeDesc)
213 else
214 text = string.format(g_i18n:getText("action_turnOnOBJECT"),
    self.typeDesc)
215 end
216 g_inputBinding:setActionEventText(actionEvent.actionEventId,
    text)
217 end
218
219 g_inputBinding:setActionEventActive(actionEvent.actionEventId,
    state)
220 end
221 end
222 end
223 end
224
225 if self.isClient then
226 if self.playAnimation ~= nil then
```

```
227 for _, animation in ipairs(spec.turnedOnAnimations) do
228 if animation.speedDirection ~= 0 then
229 local duration = animation.turnOnFadeTime
230 if animation.speedDirection == -1 then
231 duration = animation.turnOffFadeTime
232 end
233 animation.currentSpeed = MathUtil.clamp(animation.currentSpeed +
    animation.speedDirection * dt/duration, 0, 1)
234 self:setAnimationSpeed(animation.name,
    animation.currentSpeed*animation.speedScale)
235 if animation.speedDirection == -1 and animation.currentSpeed == 0
    then
236 self:stopAnimation(animation.name, true)
237 end
238
239 if animation.currentSpeed == 1 or animation.currentSpeed == 0
    then
240 animation.speedDirection = 0
241 end
242 end
243 end
244 end
245 end
246 end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
253 function TurnOnVehicle:onUpdateTick(dt, isActiveForInput,
    isSelected)
254 if self.isServer then
255 local spec = self.spec_turnOnVehicle
256 if spec.turnOffIfNotAllowed then
257 if not self:getCanBeTurnedOn() then
258 if self:getIsTurnedOn() then
259 self:setIsTurnedOn(false)
```

```
260 else
261 if self.getAttacherVehicle ~= nil then
262 local attacherVehicle = self:getAttacherVehicle()
263 if attacherVehicle ~= nil then
264 if attacherVehicle.setIsTurnedOn ~= nil and
    attacherVehicle:getIsTurnedOn() then
265 attacherVehicle:setIsTurnedOn(false)
266 end
267 end
268 end
269 end
270 end
271 end
272 end
273 end
```

setIsTurnedOn

Description

Set is turned on state

Definition

setIsTurnedOn(boolean isTurnedOn, boolean noEventSend)

Arguments

boolean isTurnedOn new is is turned on state

boolean noEventSend no event send

Code

```
279 function TurnOnVehicle:setIsTurnedOn(isTurnedOn, noEventSend)
280 local spec = self.spec_turnOnVehicle
281
282 if isTurnedOn ~= spec.isTurnedOn then
283 SetTurnedOnEvent.sendEvent(self, isTurnedOn, noEventSend)
284 spec.isTurnedOn = isTurnedOn
285
286 local actionEvent = spec.actionEvents[InputAction.TOGGLE_COVER]
287 local text
288
289 if spec.isTurnedOn then
290 SpecializationUtil.raiseEvent(self, "onTurnedOn")
291 text = string.format(g_i18n:getText(spec.turnOffText),
    self.typeDesc)
292 else
293 SpecializationUtil.raiseEvent(self, "onTurnedOff")
294 text = string.format(g_i18n:getText(spec.turnOnText),
    self.typeDesc)
```

```
295 end
296
297 if actionEvent ~= nil then
298 g_inputBinding:setActionEventText(actionEvent.actionEventId,
    text)
299 end
300 end
301 end
```

getIsTurnedOn

Description

Returns if vehicle is turned on

Definition

getIsTurnedOn()

Return Values

boolean isTurnedOn vehicle is turned on

Code

```
348 function TurnOnVehicle:getIsTurnedOn()
349 local spec = self.spec_turnOnVehicle
350
351 return spec.isAlwaysTurnedOn or spec.isTurnedOn
352 end
```

getCanBeTurnedOn

Description

Returns if vehicle can be turned on

Definition

getCanBeTurnedOn()

Return Values

boolean canBeTurnedOn vehicle can be turned on

Code

```
357 function TurnOnVehicle:getCanBeTurnedOn()
358 local spec = self.spec_turnOnVehicle
359
360 if spec.isAlwaysTurnedOn then
361 return false
362 end
363
364 if self.getInputAttacherJoint ~= nil then
365 local inputAttacherJoint = self:getInputAttacherJoint()
366
367 if inputAttacherJoint ~= nil and inputAttacherJoint.canBeTurnedOn
    ~= nil and not inputAttacherJoint.canBeTurnedOn then
368 return false
```

```
369 end
370 end
371
372 if self.getIsMotorStarted ~= nil then
373 return self:getIsMotorStarted()
374 else
375 local rootAttacherVehicle = self:getRootVehicle()
376 if rootAttacherVehicle ~= self then
377 if rootAttacherVehicle.getIsMotorStarted ~= nil then
378 return rootAttacherVehicle:getIsMotorStarted()
379 end
380 end
381 end
382
383 return true
384 end
```

getCanToggleTurnedOn

Description

Returns if user is allowed to turn on the vehicle

Definition

getCanToggleTurnedOn()

Return Values

boolean allow allow turn on

Code

```
389 function TurnOnVehicle:getCanToggleTurnedOn()
390 local spec = self.spec_turnOnVehicle
391
392 if spec.isAlwaysTurnedOn then
393 return false
394 end
395
396 if spec.turnedOnByAttacherVehicle then
397 return false
398 end
399
400 return true
401 end
```

getTurnedOnNotAllowedWarning

Description

Returns turn on not allowed warning text

Definition

getTurnedOnNotAllowedWarning()

Return Values

string warningText turn on not allowed warning text

Code

```
406  function TurnOnVehicle:getTurnedOnNotAllowedWarning()
407  local spec = self.spec_turnOnVehicle
408
409  if self.getIsMotorStarted ~= nil then
410  if not self:getIsMotorStarted() then
411  return spec.motorNotStartedWarning
412  end
413  else
414  local rootAttacherVehicle = self:getRootVehicle()
415  if rootAttacherVehicle.getIsMotorStarted ~= nil then
416  if not rootAttacherVehicle:getIsMotorStarted() then
417  return spec.motorNotStartedWarning
418  end
419  end
420  end
421
422  return nil
423  end
```

loadInputAttacherJoint

Description

Called on loading

Definition

loadInputAttacherJoint(table savegame)

Arguments

table savegame savegame

Code

```
433  function TurnOnVehicle:loadInputAttacherJoint(superFunc, xmlFile,
     key, inputAttacherJoint, i)
434  if not superFunc(self, xmlFile, key, inputAttacherJoint, i) then
435  return false
436  end
437
438  inputAttacherJoint.canBeTurnedOn =
     Utils.getNoNil(getXMLBool(xmlFile, key.. "#canBeTurnedOn"), true)
439
440  return true
441  end
```

loadWorkAreaFromXML

Description

Loads work areas from xml

Definition

loadWorkAreaFromXML(table workArea, integer xmlFile, string key)

Arguments

table	workArea	workArea
integer	xmlFile	id of xml object
string	key	key

Code

```
448  function TurnOnVehicle:loadWorkAreaFromXML(superFunc, workArea, xmlFile, key)
449  local retValue = superFunc(self, workArea, xmlFile, key)
450
451  workArea.needsSetIsTurnedOn = Utils.getNoNil( getXMLBool(xmlFile, key.."#needsSetIsTurnedOn"), true )
452
453  return retValue
454  end
```

getIsWorkAreaActive

Description

Returns true if work area is active

Definition

getIsWorkAreaActive(table workArea)

Arguments

table workArea workArea

Return Values

boolean isActive work area is active

Code

```
460  function TurnOnVehicle:getIsWorkAreaActive(superFunc, workArea)
461  if not self:getIsTurnedOn() and workArea.needsSetIsTurnedOn then
462  return false
463  end
464
465  return superFunc(self, workArea)
466  end
```

getCanAIImplementContinueWork

Description

Returns true if vehicle is ready for ai work

Definition

getCanAIImplementContinueWork()

Return Values

boolean isReady is ready for ai work

Code

```
471 function TurnOnVehicle:getCanAIImplementContinueWork(superFunc)
472 local ret = false
473 if self:getCanBeTurnedOn() then
474 if self:getIsTurnedOn() then
475 ret = true
476 end
477 end
478
479 if not self:getAIRequiresTurnOn() then
480 ret = true
481 end
482
483 -- on headland the tool can be turned off
484 if not self:getIsAIImplementInLine() then
485 ret = true
486 end
487
488 return superFunc(self) and ret
489 end
```

getIsOperating

Description

Returns if vehicle is operating

Definition

getIsOperating()

Return Values

boolean isOperating is operating

Code

```
494 function TurnOnVehicle:getIsOperating(superFunc)
495 if self:getIsTurnedOn() then
496 return true
497 end
498
499 return superFunc(self)
500 end
```

onDeactivate

Description

Called on deactivate

Definition

onDeactivate()

Code

```
615 function TurnOnVehicle:onDeactivate()
```

```
616  self:setIsTurnedOn(false, true)
617  end
```

onPreDetach

Description

Called if vehicle gets detached

Definition

onPreDetach(table attacherVehicle, table implement)

Arguments

table attacherVehicle attacher vehicle
table implement implement

Code

```
632  function TurnOnVehicle:onPreDetach(attacherVehicle, implement)
633  self:setIsTurnedOn(false, true)
634  end
```

WaterTrailer

Description

Class for water trailers

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
16  function WaterTrailer.prerequisitesPresent(specializations)
17  return SpecializationUtil.hasSpecialization(FillUnit,
    specializations)
18  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
36  function WaterTrailer:onLoad(savegame)
37  local spec = self.spec_waterTrailer
38
```

```
39  local fillUnitIndex = getXMLInt(self.xmlFile,
    "vehicle.waterTrailer#fillUnitIndex")
40  if fillUnitIndex ~= nil then
41  spec.fillUnitIndex = fillUnitIndex
42  spec.fillLitersPerSecond =
    Utils.getNoNil(getXMLFloat(self.xmlFile,
    "vehicle.waterTrailer#fillLitersPerSecond"), 500)
43  spec.waterFillNode =
    Utils.getNoNil(I3DUtil.indexToObject(self.components,
    getXMLString(self.xmlFile, "vehicle.waterTrailer#fillNode"),
    self.i3dMappings), self.components[1].node)
44  end
45
46  spec.isFilling = false
47  spec.activatableAdded = false
48  spec.activatable = WaterTrailerActivatable:new(self)
49
50  if self.isClient then
51  spec.samples = {}
52  spec.samples.refill =
    g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.waterTrailer.sounds", "refill", self.baseDirectory,
    self.components, 0, AudioGroup.VEHICLE, self.i3dMappings, self)
53  end
54  end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
58  function WaterTrailer:onDelete()
59  local spec = self.spec_waterTrailer
60  if spec.activatableAdded then
61  g_currentMission:removeActivatableObject(spec.activatable)
62  spec.activatableAdded = false
63  end
64
65  if self.isClient then
66  g_soundManager:deleteSamples(spec.samples)
67  end
68  end
```

onReadStream

Description

Called on client side on join

Definition

onReadStream(integer streamId, integer connection)

Arguments

integer	streamId	streamId
integer	connection	connection

Code

```
74  function WaterTrailer:onReadStream(streamId, connection)
75  local isFilling = streamReadBool(streamId)
76  self:setIsWaterTrailerFilling(isFilling, true)
77  end
```

onWriteStream

Description

Called on server side on join

Definition

onWriteStream(integer streamId, integer connection)

Arguments

integer	streamId	streamId
integer	connection	connection

Code

```
83  function WaterTrailer:onWriteStream(streamId, connection)
84  local spec = self.spec_waterTrailer
85  streamWriteBool(streamId, spec.isFilling)
86  end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
93  function WaterTrailer:onUpdateTick(dt, isActiveForInput,
    isSelected)
94  local spec = self.spec_waterTrailer
95
96  local _,y,_ = getWorldTranslation(spec.waterFillNode)
97  local isNearWater = (y <= g_currentMission.waterY + 0.2)
98
```

```
 99  if isNearWater then
100  if not spec.activatableAdded then
101  g_currentMission:addActivatableObject(spec.activatable)
102  spec.activatableAdded = true
103  end
104  else
105  if spec.activatableAdded then
106  g_currentMission:removeActivatableObject(spec.activatable)
107  spec.activatableAdded = false
108  end
109  end
110
111  if self.isServer then
112  if spec.isFilling then
113  -- stop filling if not near the water anymore
114  if not isNearWater then
115  self:setIsWaterTrailerFilling(false)
116  end
117  end
118
119  if spec.isFilling then
120  if self:getFillUnitAllowsFillType(spec.fillUnitIndex,
     FillType.WATER) then
121  local delta = self:addFillUnitFillLevel(self:getOwnerFarmId(),
     spec.fillUnitIndex, spec.fillLitersPerSecond*dt*0.001,
     FillType.WATER, ToolType.TRIGGER, nil)
122  if delta <= 0 then
123  self:setIsWaterTrailerFilling(false)
124  end
125  end
126  end
127  end
128  end
```

onDraw

Description

Called on draw

Definition

onDraw(boolean isActiveForInput, boolean isSelected)

Arguments

boolean isActiveForInput	true if vehicle is active for input
boolean isSelected	true if vehicle is selected

Code

```
134 function WaterTrailer:onDraw(isActiveForInput, isSelected)
135 local spec = self.spec_waterTrailer
136 if self:getFillUnitFillLevel(spec.fillUnitIndex) <= 0 and
    self:getFillUnitCapacity(spec.fillUnitIndex) ~= 0 then
137 g_currentMission:addExtraPrintText(g_i18n:getText("info_firstFillTheTool
138 end
139 end
```

setIsWaterTrailerFilling

Description

Set is water trailer filling state

Definition

setIsWaterTrailerFilling(boolean isFilling, boolean noEventSend)

Arguments

boolean isFilling	new is filling state
boolean noEventSend	no event send

Code

```
145 function WaterTrailer:setIsWaterTrailerFilling(isFilling,
    noEventSend)
146 local spec = self.spec_waterTrailer
147 if isFilling ~= spec.isFilling then
148 WaterTrailerSetIsFillingEvent.sendEvent(self, isFilling,
    noEventSend)
149
150 spec.isFilling = isFilling
151
152 if self.isClient then
153 if isFilling then
154 g_soundManager:playSample(spec.samples.refill)
155 else
156 g_soundManager:stopSample(spec.samples.refill)
157 end
158 end
159 end
160 end
```

onPreDetach

Description

Called if vehicle gets detached

Definition

onPreDetach(table attacherVehicle, table implement)

Arguments

table attacherVehicle	attacher vehicle
table implement	implement

Code

```
166 function WaterTrailer:onPreDetach(attacherVehicle, implement)
167 local spec = self.spec_waterTrailer
168 if spec.activatableAdded then
169 g_currentMission:removeActivatableObject(spec.activatable)
170 spec.activatableAdded = false
171 end
172 end
```

Weeder

Description

Class for all weeders

initSpecialization

Description

Called on specialization initializing

Definition

initSpecialization()

Code

```
15 function Weeder.initSpecialization()
16 g_workAreaTypeManager:addWorkAreaType("weeder", true)
17 end
```

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
23 function Weeder.prerequisitesPresent(specializations)
24 return SpecializationUtil.hasSpecialization(WorkArea,
   specializations) and
25 SpecializationUtil.hasSpecialization(AttacherJoints,
   specializations)
26 end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
54  function Weeder:onLoad(savegame)
55  local spec = self.spec_weeder
56
57  if self.isClient then
58  spec.samples = {}
59  spec.samples.work = g_soundManager:loadSampleFromXML(self.xmlFile,
    "vehicle.weeder.sounds", "work", self.baseDirectory, self.components,
    0, AudioGroup.VEHICLE, self.i3dMappings, self)
60  spec.isWorkSamplePlaying = false
61  end
62
63  spec.startActivationTimeout = 2000
64  spec.startActivationTime = 0
65
66  spec.maxGrowthState = Utils.getNoNil(getXMLInt(self.xmlFile,
    "vehicle.weeder.maxGrowthState"), 2)
67
68  spec.workAreaParameters = {}
69  spec.workAreaParameters.lastArea = 0
70
71  spec.isWorking = false
72
73  -- ai setup
74  if self.addAITerrainDetailRequiredRange ~= nil then
75  self:addAITerrainDetailRequiredRange(g_currentMission.sowingValue,
    g_currentMission.sowingValue,
    g_currentMission.terrainDetailTypeFirstChannel,
    g_currentMission.terrainDetailTypeNumChannels)
76  self:addAITerrainDetailRequiredRange(g_currentMission.sowingWidthValue,
    g_currentMission.sowingWidthValue,
    g_currentMission.terrainDetailTypeFirstChannel,
    g_currentMission.terrainDetailTypeNumChannels)
77  end
78
79  local fruitType = g_fruitTypeManager:getFruitTypeByName("weed")
80  if fruitType ~= nil then
81  self:setAIFruitRequirements(fruitType.index, 1, 1)
82  end
83  end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
87  function Weeder:onDelete()
88  if self.isClient then
89  local spec = self.spec_weeder
90  g_soundManager:deleteSamples(spec.samples)
91  end
92  end
```

loadWorkAreaFromXML

Description

Loads work areas from xml

Definition

loadWorkAreaFromXML(table workArea, integer xmlFile, string key)

Arguments

table	workArea	workArea
integer	xmlFile	id of xml object
string	key	key

Return Values

boolean success success

Code

```
157  function Weeder:loadWorkAreaFromXML(superFunc, workArea, xmlFile,
     key)
158  if workArea.type == WorkAreaType.DEFAULT then
159  workArea.type = WorkAreaType.WEEDER
160  end
161
162  return superFunc(self, workArea, xmlFile, key)
163  end
```

getIsWorkAreaActive

Description

Returns true if work area is active

Definition

getIsWorkAreaActive(table workArea)

Arguments

table workArea workArea

Return Values

boolean isActive work area is active

Code

```
169  function Weeder:getIsWorkAreaActive(superFunc, workArea)
170  if workArea.type == WorkAreaType.WEEDER then
```

```
171  local isActive = true
172  if workArea.requiresGroundContact and
     workArea.groundReferenceNode ~= nil then
173  isActive = isActive and
     self:getIsGroundReferenceNodeActive(workArea.groundReferenceNode)
174  end
175  if isActive and workArea.disableBackwards then
176  isActive = isActive and self.movingDirection > 0
177  end
178  return isActive
179  end
180
181  return superFunc(self, workArea)
182  end
```

doCheckSpeedLimit

Description

Returns if speed limit should be checked

Definition

doCheckSpeedLimit()

Return Values

boolean checkSpeedlimit check speed limit

Code

```
187  function Weeder:doCheckSpeedLimit(superFunc)
188  return superFunc(self) or self:getIsImplementChainLowered()
189  end
```

getDirtMultiplier

Description

Returns current dirt multiplier

Definition

getDirtMultiplier()

Return Values

float dirtMultiplier current dirt multiplier

Code

```
194  function Weeder:getDirtMultiplier(superFunc)
195  local spec = self.spec_weeder
196  local multiplier = superFunc(self)
197
198  if spec.isWorking then
199  multiplier = multiplier + self:getWorkDirtMultiplier() *
     self:getLastSpeed() / self.speedLimit
200  end
201
```

```
202 return multiplier
203 end
```

getWearMultiplier

Description

Returns current wear multiplier

Definition

getWearMultiplier()

Return Values

float dirtMultiplier current wear multiplier

Code

```
208 function Weeder:getWearMultiplier(superFunc)
209 local spec = self.spec_weeder
210 local multiplier = superFunc(self)
211
212 if spec.isWorking then
213 multiplier = multiplier + self:getWorkWearMultiplier() *
    self:getLastSpeed() / self.speedLimit
214 end
215
216 return multiplier
217 end
```

loadGroundParticleMapping

Description

Load ground particle mapping from xml file

Definition

loadGroundParticleMapping(integer xmlFile, string key, table mapping, integer index, integer i3dNode)

Arguments

integer	xmlFile	id of xml object
string	key	key
table	mapping	mapping
integer	index	index
integer	i3dNode	id of i3d node

Return Values

boolean success success

Code

```
227 function Weeder:loadGroundParticleMapping(superFunc, xmlFile,
    key, mapping, index, i3dNode)
228 if not superFunc(self, xmlFile, key, mapping, index, i3dNode)
    then
229 return false
230 end
231
```

```
232  mapping.adjustColor = Utils.getNoNil(getXMLBool(xmlFile,
     key.."#adjustColor"), false)
233  if mapping.adjustColor then
234  local spec = self.spec_weeder
235  if spec.colorParticleSystems == nil then
236  spec.colorParticleSystems = {}
237  end
238  mapping.lastColor = {}
239
240  table.insert(spec.colorParticleSystems, mapping)
241  end
242
243  return true
244  end
```

onPostAttach

Description

Called if vehicle gets attached

Definition

onPostAttach(table attacherVehicle, integer inputJointDescIndex, integer jointDescIndex)

Arguments

table	attacherVehicle	attacher vehicle
integer	inputJointDescIndex	index of input attacher joint
integer	jointDescIndex	index of attacher joint it gets attached to

Code

```
292  function Weeder:onPostAttach(attacherVehicle,
     inputJointDescIndex, jointDescIndex)
293  local spec = self.spec_weeder
294  spec.startActivationTime = g_currentMission.time +
     spec.startActivationTimeout
295  end
```

getDefaultSpeedLimit

Description

Returns default speed limit

Definition

getDefaultSpeedLimit()

Return Values

float speedLimit speed limit

Code

```
300  function Weeder.getDefaultSpeedLimit()
301  return 15
302  end
```

Wheels

Description

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
993  function Wheels:onUpdate(dt, isActiveForInput, isSelected)
994  local spec = self.spec_wheels
995
996  -- interpolation of wheel properties
997  if not self.isServer and self.isClient then
998  spec.networkTimeInterpolator:update(dt)
999  local interpolationAlpha = spec.networkTimeInterpolator:getAlpha()
1000
1001 self.rotatedTime =
     self.rotatedTimeInterpolator:getInterpolatedValue(interpolationAlpha)
1002
1003 for i=1, table.getn(spec.wheels) do
1004 local wheel = spec.wheels[i]
1005 wheel.netInfo.x, wheel.netInfo.y, wheel.netInfo.z =
     wheel.networkInterpolators.position:getInterpolatedValues(interpolatior
1006 wheel.netInfo.xDrive =
     wheel.networkInterpolators.xDrive:getInterpolatedValue(interpolationAlp
1007 wheel.netInfo.suspensionLength =
     wheel.networkInterpolators.suspensionLength:getInterpolatedValue(interp
1008
1009 if wheel.driveGroundParticleSystems ~= nil then
1010 for _,ps in pairs(wheel.driveGroundParticleSystems) do
1011 setTranslation(ps.emitterShape, wheel.netInfo.x + ps.offsets[1], wheel.
     ps.offsets[2], wheel.netInfo.z + ps.offsets[3])
1012 end
1013 end
1014 end
1015
1016 if spec.networkTimeInterpolator:isInterpolating() then
1017 self:raiseActive()
1018 end
1019 end
```

```
1020
1021  if self.firstTimeRun then
1022
1023  for _,wheel in pairs(spec.wheels) do
1024  self:updateWheelContact(wheel)
1025  self:updateWheelSink(wheel, dt)
1026  self:updateWheelFriction(wheel, dt)
1027  self:updateWheelTireTracks(wheel)
1028  self:updateWheelDensityMapHeight(wheel, dt)
1029  self:updateWheelDestruction(wheel, dt)
1030
1031  WheelsUtil.updateWheelPhysics(self, wheel, spec.brakePedal, dt)
1032  local changed = WheelsUtil.updateWheelGraphics(self, wheel, dt)
1033  if wheel.updateWheelChock and changed then
1034  for _, wheelChock in ipairs(wheel.wheelChocks) do
1035  self:updateWheelChockPosition(wheelChock, false)
1036  end
1037  end
1038  end
1039
1040  if self:getAreSurfaceSoundsActive() then
1041  -- update surface sounds
1042  if spec.surfaceSounds ~= nil then
1043  local currentSound = self:getCurrentSurfaceSound()
1044
1045  if currentSound ~= spec.currentSurfaceSound then
1046  if spec.currentSurfaceSound ~= nil then
1047  g_soundManager:stopSample(spec.currentSurfaceSound)
1048  end
1049  if currentSound ~= nil then
1050  g_soundManager:playSample(currentSound)
1051  end
1052
1053  spec.currentSurfaceSound = currentSound
1054  end
1055  end
1056  end
1057  end
1058
1059  if self.isServer then
```

```
1060    self:raiseDirtyFlags(spec.dirtyFlag)
1061    end
1062    end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActive, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActive	true if vehicle is active
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
1087    function Wheels:onUpdateTick(dt, isActiveForInput, isSelected)
1088    local spec = self.spec_wheels
1089
1090    for _, wheel in pairs(spec.wheels) do
1091    if wheel.rotSpeedLimit ~= nil then
1092    local dir = -1
1093    if self:getLastSpeed() <= wheel.rotSpeedLimit then
1094    dir = 1
1095    end
1096
1097    wheel.currentRotSpeedAlpha =
        MathUtil.clamp(wheel.currentRotSpeedAlpha + dir*(dt/1000), 0, 1)
1098    wheel.rotSpeed = wheel.rotSpeedDefault * wheel.currentRotSpeedAlpha
1099    wheel.rotSpeedNeg = wheel.rotSpeedNegDefault *
        wheel.currentRotSpeedAlpha
1100    end
1101    end
1102
1103    if self.isClient then
1104    local speed = self:getLastSpeed()
1105    local groundWetness =
        g_currentMission.environment.weather:getGroundWetness()
1106    local groundIsWet = groundWetness > 0.2
1107    for _,wheel in pairs(spec.wheels) do
1108    if wheel.driveGroundParticleSystems ~= nil then
1109    local states = wheel.driveGroundParticleStates
1110    local enableSoilPS = false
1111    if wheel.lastTerrainValue > 0 and wheel.lastTerrainValue < 5 then
```

```
1112  enableSoilPS = (speed > 1) -- and wheel.sink > 0
1113  end
1114  local sizeScale = 2 * wheel.width * wheel.radiusOriginal
1115
1116  states.driving_dry = enableSoilPS
1117  states.driving_wet = enableSoilPS and groundIsWet
1118  states.driving_dust = not groundIsWet
1119
1120  for psName, state in pairs(states) do
1121  local ps = wheel.driveGroundParticleSystems[psName]
1122
1123  if state then
1124  if self.movingDirection < 0 then
1125  setRotation(ps.emitterShape, 0, math.pi+wheel.steeringAngle, 0)
1126  else
1127  setRotation(ps.emitterShape, 0, wheel.steeringAngle, 0)
1128  end
1129
1130  local scale
1131  if psName ~= "driving_dust" and self.isServer and self.isClient
      then
1132  local wheelSpeed = getWheelShapeAxleSpeed(wheel.node,
      wheel.wheelShape)*30/math.pi * wheel.radius * 2 * 0.001885
1133  local wheelSlip = math.pow(wheelSpeed/self.lastSpeedReal*100, 2.5)
1134  scale = self:getDriveGroundParticleSystemsScale(ps, wheelSpeed) *
      wheelSlip
1135  else
1136  scale = self:getDriveGroundParticleSystemsScale(ps,
      self.lastSpeedReal)
1137  end
1138
1139  if ps.isTintable then
1140  -- interpolate between different ground colors to avoid
      unrealisitic particle color changes
1141  if ps.lastColor == nil then
1142  ps.lastColor =
      {ps.wheel.lastColor[1],ps.wheel.lastColor[2],ps.wheel.lastColor[3]}
1143  ps.targetColor =
      {ps.wheel.lastColor[1],ps.wheel.lastColor[2],ps.wheel.lastColor[3]}
1144  ps.currentColor =
      {ps.wheel.lastColor[1],ps.wheel.lastColor[2],ps.wheel.lastColor[3]}
1145  ps.alpha = 1
```

```
1146  end
1147
1148  if ps.alpha ~= 1 then
1149  ps.alpha = math.min(ps.alpha + dt/1000, 1)
1150  ps.currentColor = {MathUtil.vector3ArrayLerp(ps.lastColor,
      ps.targetColor, ps.alpha)}
1151  if ps.alpha == 1 then
1152  ps.lastColor[1] = ps.currentColor[1]
1153  ps.lastColor[2] = ps.currentColor[2]
1154  ps.lastColor[3] = ps.currentColor[3]
1155  end
1156  end
1157
1158  if ps.alpha == 1 and ps.wheel.lastColor[1] ~= ps.targetColor[1] and
      ps.wheel.lastColor[2] ~= ps.targetColor[2] and
      ps.wheel.lastColor[3] ~= ps.targetColor[3] then
1159  ps.alpha = 0
1160  ps.targetColor[1] = ps.wheel.lastColor[1]
1161  ps.targetColor[2] = ps.wheel.lastColor[2]
1162  ps.targetColor[3] = ps.wheel.lastColor[3]
1163  end
1164  end
1165
1166  if scale > 0 then
1167  ParticleUtil.setEmittingState(ps, true)
1168  if ps.isTintable then
1169  setShaderParameter(ps.shape, "psColor", ps.currentColor[1],
      ps.currentColor[2], ps.currentColor[3], 1, false)
1170  end
1171  else
1172  ParticleUtil.setEmittingState(ps, false)
1173  end
1174
1175  -- emit count
1176  local maxSpeed = (50 / 3.6)
1177  local circum = wheel.radiusOriginal
1178  local maxWheelRpm = maxSpeed / circum
1179  local wheelRotFactor = Utils.getNoNil(wheel.netInfo.xDriveSpeed, 0)
      / maxWheelRpm
1180  local emitScale = scale * wheelRotFactor * sizeScale
1181  ParticleUtil.setEmitCountScale(ps, MathUtil.clamp(emitScale,
      ps.minScale, ps.maxScale))
```

```
1182
1183  -- speeds
1184  local speedFactor = 1.0
1185  ParticleUtil.setParticleSystemSpeed(ps, ps.particleSpeed *
      speedFactor)
1186  ParticleUtil.setParticleSystemSpeedRandom(ps,
      ps.particleRandomSpeed * speedFactor)
1187  else
1188  ParticleUtil.setEmittingState(ps, false)
1189  end
1190
1191  states[psName] = false
1192  end
1193  end
1194  end
1195  end
1196  end
```

getTotalMass

Description

Returns total mass of vehicle (optional including attached vehicles)

Definition

getTotalMass(boolean onlyGivenVehicle)

Arguments

boolean onlyGivenVehicle use only the given vehicle, if false or nil it includes all attachables

Return Values

float totalMass total mass

Code

```
1239  function Wheels:getTotalMass(superFunc, onlyGivenVehicle)
1240  local mass = superFunc(self)
1241
1242  local spec = self.spec_wheels
1243  for _, wheel in pairs(spec.wheels) do
1244  mass = mass + wheel.mass
1245  end
1246
1247  return mass
1248  end
```

WoodCrusher

Description

This is the specialization for wood crushers

prerequisitesPresent

Description

Checks if all prerequisite specializations are loaded

Definition

prerequisitesPresent(table specializations)

Arguments

table specializations specializations

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
17  function WoodCrusher.prerequisitesPresent(specializations)
18  return SpecializationUtil.hasSpecialization(TurnOnVehicle,
    specializations) and
    SpecializationUtil.hasSpecialization(FillUnit, specializations)
19  end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
45  function WoodCrusher:onLoad(savegame)
46  local spec = self.spec_woodCrusher
47
48  WoodCrusher.loadWoodCrusher(self, spec, self.xmlFile,
    self.components)
49
50  local moveColDisableCollisionPairs =
    Utils.getNoNil(getXMLBool(self.xmlFile,
    "vehicle.woodCrusher#moveColDisableCollisionPairs"), true)
51  if moveColDisableCollisionPairs then
52  for _, component in pairs(self.components) do
53  for _, node in pairs(spec.moveColNodes) do
54  setPairCollision(component.node, node, false)
55  end
56  end
57  end
58
59  spec.fillUnitIndex = Utils.getNoNil(getXMLInt(self.xmlFile,
    "vehicle.woodCrusher#fillUnitIndex"), 1)
60  end
```

onDelete

Description

Called on deleting

Definition

onDelete()

Code

```
64  function WoodCrusher:onDelete()
65  WoodCrusher.deleteWoodCrusher(self, self.spec_woodCrusher)
66  end
```

onReadUpdateStream

Description

Called on on update

Definition

onReadUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId	stream ID
integer timestamp	timestamp
table	connection connection

Code

```
73  function WoodCrusher:onReadUpdateStream(streamId, timestamp,
    connection)
74  if connection:getIsServer() then
75  local spec = self.spec_woodCrusher
76  if streamReadBool(streamId) then
77  spec.crushingTime = 1000
78  else
79  spec.crushingTime = 0
80  end
81  end
82  end
```

onWriteUpdateStream

Description

Called on on update

Definition

onWriteUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	stream ID
table	connection connection
integer dirtyMask	dirty mask

Code

```
89  function WoodCrusher:onWriteUpdateStream(streamId, connection,
    dirtyMask)
90  if not connection:getIsServer() then
91  local spec = self.spec_woodCrusher
```

```
92  streamWriteBool(streamId, spec.crushingTime > 0)
93  end
94  end
```

onUpdate

Description

Called on update

Definition

onUpdate(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
101 function WoodCrusher:onUpdate(dt, isActiveForInput, isSelected)
102 WoodCrusher.updateWoodCrusher(self, self.spec_woodCrusher, dt,
    self:getIsTurnedOn())
103 end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
110 function WoodCrusher:onUpdateTick(dt, isActiveForInput,
    isSelected)
111 WoodCrusher.updateTickWoodCrusher(self, self.spec_woodCrusher,
    dt, self:getIsTurnedOn())
112
113 local spec = self.spec_woodCrusher
114 -- turn on/off automatically if tree node is in trigger
115 if self.isServer then
116 if g_currentMission.missionInfo.automaticMotorStartEnabled then
117 if spec.turnOnAutomatically and self.setIsTurnedOn ~= nil then
118 if next(spec.moveTriggerNodes) ~= nil then
119 if self.getIsMotorStarted ~= nil then
120 if not self:getIsMotorStarted() then
121 self:startMotor()
122 end
```

```
else
if self.attacherVehicle ~= nil then
if self.attacherVehicle.getIsMotorStarted ~= nil then
if not self.attacherVehicle:getIsMotorStarted() then
self.attacherVehicle:startMotor()
end
end
end
end

if not self.isControlled and not self:getIsTurnedOn() and
self:getCanBeTurnedOn() then
self:setIsTurnedOn(true)
end
spec.turnOffTimer = 3000
else
if self:getIsTurnedOn() then
if spec.turnOffTimer == nil then
spec.turnOffTimer = 3000
end
spec.turnOffTimer = spec.turnOffTimer - dt

if spec.turnOffTimer < 0 then
local rootAttacherVehicle = self:getRootVehicle()

if not rootAttacherVehicle.isControlled then
if self.getIsMotorStarted ~= nil then
if self:getIsMotorStarted() then
self:stopMotor()
end
end

self:setIsTurnedOn(false)
end
end
end
end
end
end
end
```

```
162 end
```

onTurnedOn

Description

Called on turn on

Definition

onTurnedOn(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
167 function WoodCrusher:onTurnedOn()
168 WoodCrusher.turnOnWoodCrusher(self, self.spec_woodCrusher)
169 end
```

onTurnedOff

Description

Called on turn off

Definition

onTurnedOff(boolean noEventSend)

Arguments

boolean noEventSend no event send

Code

```
174 function WoodCrusher:onTurnedOff()
175 WoodCrusher.turnOffWoodCrusher(self, self.spec_woodCrusher)
176 end
```

getCanBeTurnedOn

Description

Returns if user is allowed to turn on the vehicle

Definition

getCanBeTurnedOn()

Return Values

boolean allow allow turn on

Code

```
181 function WoodCrusher:getCanBeTurnedOn(superFunc)
182 local spec = self.spec_woodCrusher
183 if spec.turnOnAutomatically then
184 return false
185 end
186
187 return superFunc(self)
188 end
```

getDirtMultiplier

Description

Returns current dirt multiplier

Definition

getDirtMultiplier()

Return Values

float dirtMultiplier current dirt multiplier

Code

```
193  function WoodCrusher:getDirtMultiplier(superFunc)
194  local multiplier = superFunc(self)
195
196  local spec = self.spec_woodCrusher
197  if spec.crushingTime > 0 then
198  multiplier = multiplier + self:getWorkDirtMultiplier()
199  end
200
201  return multiplier
202  end
```

getWearMultiplier

Description

Returns current wear multiplier

Definition

getWearMultiplier()

Return Values

float dirtMultiplier current wear multiplier

Code

```
207  function WoodCrusher:getWearMultiplier(superFunc)
208  local multiplier = superFunc(self)
209
210  local spec = self.spec_woodCrusher
211  if spec.crushingTime > 0 then
212  multiplier = multiplier + self:getWorkWearMultiplier()
213  end
214
215  return multiplier
216  end
```

onCrushedSplitShape

Description

Called on crush split shape

Definition

onCrushedSplitShape(table splitType, float volume)

Arguments

table splitType split type

float volume volume

Code

```
222 function WoodCrusher:onCrushedSplitShape(splitType, volume)
223 local spec = self.spec_woodCrusher
224 self:addFillUnitFillLevel(self:getOwnerFarmId(),
    spec.fillUnitIndex, volume * 1000 * splitType.woodChipsPerLiter,
    FillType.WOODCHIPS, ToolType.UNDEFINED)
225 end
```

loadWoodCrusher

Description

Load wood crusher from xml file

Definition

loadWoodCrusher(integer xmlFile, integer rootNode)

Arguments

integer xmlFile id of xml object

integer rootNode id of root node

Code

```
231 function WoodCrusher.loadWoodCrusher(self, woodCrusher, xmlFile,
    rootNode)
232 woodCrusher.vehicle = self
233
234 woodCrusher.woodCrusherSplitShapeCallback =
    WoodCrusher.woodCrusherSplitShapeCallback
235 woodCrusher.woodCrusherMoveTriggerCallback =
    WoodCrusher.woodCrusherMoveTriggerCallback
236
237 local xmlRoot = getXMLRootName(xmlFile)
238
239 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
    xmlRoot .. ".woodCrusher.moveTrigger(0)#index", xmlRoot ..
    ".woodCrusher.moveTriggers.trigger#node") --FS17 to FS19
240 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
    xmlRoot .. ".woodCrusher.moveCollision(0)#index", xmlRoot ..
    ".woodCrusher.moveCollisions.collision#node") --FS17 to FS19
241 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
    xmlRoot .. ".woodCrusher.emitterShape(0)", xmlRoot ..
    ".woodCrusher.crushEffects with effectClass 'ParticleEffect'") --
    FS17 to FS19
242 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
    xmlRoot .. ".woodCrusherStartSound", xmlRoot ..
    ".woodCrusher.sounds.start") --FS17 to FS19
243 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
    xmlRoot .. ".woodCrusherIdleSound", xmlRoot ..
    ".woodCrusher.sounds.idle") --FS17 to FS19
244 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
    xmlRoot .. ".woodCrusherWorkSound", xmlRoot ..
    ".woodCrusher.sounds.work") --FS17 to FS19
```

```
245 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
    xmlRoot .. ".woodCrusherStopSound", xmlRoot ..
    ".woodCrusher.sounds.stop") --FS17 to FS19
246 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
    xmlRoot .. ".turnedOnRotationNodes.turnedOnRotationNode#type",
    xmlRoot .. ".woodCrusher.animationNodes.animationNode",
    "woodCrusher") --FS17 to FS19
247 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName,
    xmlRoot .. ".turnedOnScrollers.turnedOnScroller", xmlRoot ..
    ".woodCrusher.animationNodes.animationNode") --FS17 to FS19
248
249 local baseKey = xmlRoot .. ".woodCrusher"
250 woodCrusher.cutNode = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, baseKey.."#cutNode"), self.i3dMappings)
251 woodCrusher.mainDrumRefNode = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, baseKey.."#mainDrumRefNode"),
    self.i3dMappings)
252
253 woodCrusher.moveTriggers = {}
254 local i = 0
255 while true do
256 local key = string.format("%s.moveTriggers.trigger(%d)", baseKey,
    i)
257 if not hasXMLProperty(xmlFile, key) then
258 break
259 end
260
261 local node = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, key .. "#node"), self.i3dMappings)
262 if node ~= nil then
263 table.insert(woodCrusher.moveTriggers, node)
264 end
265 i = i + 1
266 end
267
268 woodCrusher.moveColNodes = {}
269 i = 0
270 while true do
271 local key = string.format("%s.moveCollisions.collision(%d)",
    baseKey, i)
272 if not hasXMLProperty(xmlFile, key) then
273 break
274 end
```

```
275
276 local node = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, key .. "#node"), self.i3dMappings)
277 if node ~= nil then
278 table.insert(woodCrusher.moveColNodes, node)
279 end
280 i = i + 1
281 end
282
283 woodCrusher.moveVelocityZ = Utils.getNoNil(getXMLFloat(xmlFile,
    baseKey.."#moveVelocityZ"), 0.8) -- m/s
284 woodCrusher.moveMaxForce = Utils.getNoNil(getXMLFloat(xmlFile,
    baseKey.."#moveMaxForce"), 7) -- input is kN
285 woodCrusher.downForceNode = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, baseKey.."#downForceNode"),
    self.i3dMappings)
286 woodCrusher.downForce = Utils.getNoNil(getXMLFloat(xmlFile,
    baseKey.."#downForce"), 2)
287 woodCrusher.cutSizeY = Utils.getNoNil(getXMLFloat(xmlFile,
    baseKey.."#cutSizeY"), 1)
288 woodCrusher.cutSizeZ = Utils.getNoNil(getXMLFloat(xmlFile,
    baseKey.."#cutSizeZ"), 1)
289
290 woodCrusher.moveTriggerNodes = {}
291 if self.isServer and woodCrusher.moveTriggers ~= nil then
292 for _,node in pairs(woodCrusher.moveTriggers) do
293 addTrigger(node, "woodCrusherMoveTriggerCallback", woodCrusher)
294 end
295 end
296
297 woodCrusher.crushNodes = {}
298 woodCrusher.crushingTime = 0
299 woodCrusher.turnOnAutomatically =
    Utils.getNoNil(getXMLBool(xmlFile, baseKey ..
    "#automaticallyTurnOn"), false)
300
301 if self.isClient then
302 woodCrusher.crushEffects = g_effectManager:loadEffect(xmlFile,
    baseKey..".crushEffects", rootNode, self, self.i3dMappings)
303
304 woodCrusher.animationNodes =
    g_animationManager:loadAnimations(xmlFile,
    baseKey..".animationNodes", rootNode, self, self.i3dMappings)
```

```
305
306 woodCrusher.isWorkSamplePlaying = false
307 woodCrusher.samples = {}
308 woodCrusher.samples.start =
    g_soundManager:loadSampleFromXML(xmlFile, baseKey..".sounds",
    "start", self.baseDirectory, rootNode, 1, AudioGroup.VEHICLE,
    self.i3dMappings, self)
309 woodCrusher.samples.stop =
    g_soundManager:loadSampleFromXML(xmlFile, baseKey..".sounds",
    "stop", self.baseDirectory, rootNode, 1, AudioGroup.VEHICLE,
    self.i3dMappings, self)
310 woodCrusher.samples.work =
    g_soundManager:loadSampleFromXML(xmlFile, baseKey..".sounds",
    "work", self.baseDirectory, rootNode, 0, AudioGroup.VEHICLE,
    self.i3dMappings, self)
311 woodCrusher.samples.idle =
    g_soundManager:loadSampleFromXML(xmlFile, baseKey..".sounds",
    "idle", self.baseDirectory, rootNode, 0, AudioGroup.VEHICLE,
    self.i3dMappings, self)
312 end
313 end
```

deleteWoodCrusher

Description

Delete wood crusher

Definition

deleteWoodCrusher()

Code

```
317 function WoodCrusher.deleteWoodCrusher(self, woodCrusher)
318 if self.isServer and woodCrusher.moveTriggers ~= nil then
319 for _,node in pairs(woodCrusher.moveTriggers) do
320 removeTrigger(node)
321 end
322 end
323
324 if self.isClient then
325 g_effectManager:deleteEffects(woodCrusher.crushEffects)
326 g_soundManager:deleteSamples(woodCrusher.samples)
327 g_animationManager:deleteAnimations(woodCrusher.animationNodes)
328 end
329 end
```

updateWoodCrusher

Description

Update wood crusher

Definition

updateWoodCrusher(float dt)

Arguments

float dt time since last call in ms

Code

```
334  function WoodCrusher.updateWoodCrusher(self, woodCrusher, dt,
     isTurnedOn)
335  if isTurnedOn then
336  if self.isServer then
337  for node in pairs(woodCrusher.crushNodes) do
338  WoodCrusher.crushSplitShape(self, woodCrusher, node)
339  woodCrusher.crushNodes[node] = nil
340  woodCrusher.moveTriggerNodes[node] = nil
341  end
342
343  local x,y,z = getTranslation(woodCrusher.mainDrumRefNode)
344  local _, ty, _ = 0, 0, 0
345  local maxTreeSizeY = 0
346  for id in pairs(woodCrusher.moveTriggerNodes) do
347  if not entityExists(id) then
348  woodCrusher.moveTriggerNodes[id] = nil
349  else
350  if woodCrusher.downForceNode ~= nil then
351  local x,y,z = getWorldTranslation(woodCrusher.downForceNode)
352  local nx,ny,nz = localDirectionToWorld(woodCrusher.downForceNode,
     1,0,0)
353  local yx,yy,yz = localDirectionToWorld(woodCrusher.downForceNode,
     0,1,0)
354
355  local minY,maxY, minZ,maxZ = testSplitShape(id, x,y,z, nx,ny,nz,
     yx,yy,yz, woodCrusher.cutSizeY, woodCrusher.cutSizeZ)
356  if minY ~= nil then
357  local cx,cy,cz = localToWorld(woodCrusher.downForceNode, 0,
     (minY+maxY)*0.5, (minZ+maxZ)*0.5)
358  local downX,downY,downZ =
     localDirectionToWorld(woodCrusher.downForceNode, 0,-
     woodCrusher.downForce,0)
359  addForce(id, downX, downY, downZ, cx,cy,cz, false)
360  if woodCrusher.mainDrumRefNode ~= nil then
361  maxTreeSizeY = math.max(maxTreeSizeY, maxY)
362  end
363  end
364
```

```
365 end
366 end
367 end
368 if woodCrusher.mainDrumRefNode ~= nil then
369 if maxTreeSizeY > 0 then
370 local a,b,c = localToWorld(woodCrusher.downForceNode, 0, maxTreeSizeY, 0)
371 _, ty, _ = worldToLocal(getParent(woodCrusher.mainDrumRefNode), a,b,c)
372 end
373 if ty > y then
374 y = math.min(y + 0.0003*dt, ty)
375 else
376 y = math.max(y - 0.0003*dt, ty)
377 end
378 setTranslation(woodCrusher.mainDrumRefNode, x,y,z)
379 end
380
381 if next(woodCrusher.moveTriggerNodes) ~= nil or woodCrusher.crushingTime > 0 then
382 self:raiseActive()
383 end
384 end
385 end
386 end
```

updateTickWoodCrusher

Description

Update tick wood crusher

Definition

updateTickWoodCrusher(float dt)

Arguments

float dt time since last call in ms

Code

```
391 function WoodCrusher.updateTickWoodCrusher(self, woodCrusher, dt, isTurnedOn)
392 if isTurnedOn then
393 if self.isServer then
394 if woodCrusher.cutNode ~= nil and next(woodCrusher.moveTriggerNodes) ~= nil then
395 local x,y,z = getWorldTranslation(woodCrusher.cutNode)
396 local nx,ny,nz = localDirectionToWorld(woodCrusher.cutNode, 1,0,0)
```

```
local yx,yy,yz = localDirectionToWorld(woodCrusher.cutNode, 0,1,0)
for id in pairs(woodCrusher.moveTriggerNodes) do
local lenBelow, lenAbove = getSplitShapePlaneExtents(id, x,y,z, nx,ny,nz)
if lenAbove ~= nil and lenBelow ~= nil then
if lenBelow <= 0.4 then
woodCrusher.moveTriggerNodes[id] = nil
WoodCrusher.crushSplitShape(self, woodCrusher, id)
elseif lenAbove >= 0.2 then
local minY = splitShape(id, x,y,z, nx,ny,nz, yx,yy,yz, woodCrusher.cutSizeY, woodCrusher.cutSizeZ, "woodCrusherSplitShapeCallback", woodCrusher)
if minY ~= nil then
woodCrusher.moveTriggerNodes[id] = nil
end
end
end
end
end
end

if woodCrusher.crushingTime > 0 then
woodCrusher.crushingTime = math.max(woodCrusher.crushingTime - dt, 0)
end

local isCrushing = woodCrusher.crushingTime > 0

if self.isClient then
if isCrushing then
g_effectManager:setFillType(woodCrusher.crushEffects, FillType.WOODCHIPS)
g_effectManager:startEffects(woodCrusher.crushEffects)
else
g_effectManager:stopEffects(woodCrusher.crushEffects)
end

if isTurnedOn and isCrushing then
if not woodCrusher.isWorkSamplePlaying then
g_soundManager:playSample(woodCrusher.samples.work)
```

```
433 woodCrusher.isWorkSamplePlaying = true
434 end
435 else
436 if woodCrusher.isWorkSamplePlaying then
437 g_soundManager:stopSample(woodCrusher.samples.work)
438 woodCrusher.isWorkSamplePlaying = false
439 end
440 end
441 end
442 end
```

turnOnWoodCrusher

Description

Turn on wood crusher

Definition

turnOnWoodCrusher()

Code

```
446 function WoodCrusher.turnOnWoodCrusher(self, woodCrusher)
447 if self.isServer and woodCrusher.moveColNodes ~= nil then
448 for _,node in pairs(woodCrusher.moveColNodes) do
449 setFrictionVelocity(node, woodCrusher.moveVelocityZ)
450 end
451 end
452
453 if self.isClient then
454 g_soundManager:stopSamples(woodCrusher.samples)
455 woodCrusher.isWorkSamplePlaying = false
456 g_soundManager:playSample(woodCrusher.samples.start)
457 g_soundManager:playSample(woodCrusher.samples.idle, 0,
    woodCrusher.samples.start)
458
459 if self.isClient then
460 g_animationManager:startAnimations(woodCrusher.animationNodes)
461 end
462 end
463 end
```

turnOffWoodCrusher

Description

Turn off wood crusher

Definition

turnOffWoodCrusher()

Code

```
467 function WoodCrusher.turnOffWoodCrusher(self, woodCrusher)
468 if self.isServer then
469 for node in pairs(woodCrusher.crushNodes) do
470 WoodCrusher.crushSplitShape(self, woodCrusher, node)
471 woodCrusher.crushNodes[node] = nil
472 end
473 if woodCrusher.moveColNodes ~= nil then
474 for _,node in pairs(woodCrusher.moveColNodes) do
475 setFrictionVelocity(node, 0.0)
476 end
477 end
478 end
479
480 if self.isClient then
481 g_effectManager:stopEffects(woodCrusher.crushEffects)
482 g_soundManager:stopSamples(woodCrusher.samples)
483 g_soundManager:playSample(woodCrusher.samples.stop)
484 woodCrusher.isWorkSamplePlaying = false
485
486 if self.isClient then
487 g_animationManager:stopAnimations(woodCrusher.animationNodes)
488 end
489 end
490 end
```

crushSplitShape

Description

Crush split shape

Definition

crushSplitShape(integer shape)

Arguments

integer shape id of shape

Code

```
495 function WoodCrusher.crushSplitShape(self, woodCrusher, shape)
496 local splitType = g_splitTypeManager:getSplitTypeByIndex(getSplitType(shape))
497 if splitType ~= nil and splitType.woodChipsPerLiter > 0 then
498 local volume = getVolume(shape)
499 delete(shape)
500 woodCrusher.crushingTime = 1000
501 self:onCrushedSplitShape(splitType, volume)
502 end
```

```
503 end
```

woodCrusherSplitShapeCallback

Description

Split shape callback

Definition

woodCrusherSplitShapeCallback(integer shape, boolean isBelow, boolean isAbove, float minY, float maxY, float minZ, float maxZ)

Arguments

integer	shape	shape
boolean	isBelow	is below
boolean	isAbove	is above
float	minY	min y split position
float	maxY	max y split position
float	minZ	min z split position
float	maxZ	max z split position

Code

```
514 function WoodCrusher.woodCrusherSplitShapeCallback(self, shape, isBelow, isAbove, minY, maxY, minZ, maxZ)
515 if not isBelow then
516 self.crushNodes[shape] = shape
517 end
518 end
```

woodCrusherMoveTriggerCallback

Description

Trigger callback

Definition

woodCrusherMoveTriggerCallback(integer triggerId, integer otherActorId, boolean onEnter, boolean onLeave, boolean onStay, integer otherShapeId)

Arguments

integer	triggerId	id of trigger
integer	otherActorId	id of other actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay
integer	otherShapeId	id of other shape

Code

```
528 function WoodCrusher.woodCrusherMoveTriggerCallback(self, triggerId, otherActorId, onEnter, onLeave, onStay, otherShapeId)
529 local vehicle = g_currentMission.nodeToObject[otherActorId]
530 if vehicle == nil and getRigidBodyType(otherActorId) == "Dynamic" then
531 local splitType = g_splitTypeManager:getSplitTypeByIndex(getSplitType(otherActorId))
532 if splitType ~= nil and splitType.woodChipsPerLiter > 0 then
```

```
533 if onEnter then
534 self.moveTriggerNodes[otherActorId] =
    Utils.getNoNil(self.moveTriggerNodes[otherActorId],0)+1
535 self.vehicle:raiseActive()
536 elseif onLeave then
537 local c = self.moveTriggerNodes[otherActorId]
538 if c ~= nil then
539 c = c-1
540 if c == 0 then
541 self.moveTriggerNodes[otherActorId] = nil
542 else
543 self.moveTriggerNodes[otherActorId] = c
544 end
545 end
546 end
547 end
548 end
549 end
```

WorkArea

Description

Class for all WorkAreas

initSpecialization

Description

Called on specialization initializing

Definition

initSpecialization()

Code

```
15 function WorkArea.initSpecialization()
16 g_workAreaTypeManager:addWorkAreaType("default", false)
17 g_workAreaTypeManager:addWorkAreaType("auxiliary", false)
18 end
```

onLoad

Description

Called on loading

Definition

onLoad(table savegame)

Arguments

table savegame savegame

Code

```
57 function WorkArea:onLoad(savegame)
58 local spec = self.spec_workArea
```

```lua

XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "vehicle.workAreas.workArea(0)#startIndex", "vehicle.workAreas.workArea(0).area#startIndex") --FS17 to FS19
XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "vehicle.workAreas.workArea(0)#widthIndex", "vehicle.workAreas.workArea(0).area#widthIndex") --FS17 to FS19
XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "vehicle.workAreas.workArea(0)#heightIndex", "vehicle.workAreas.workArea(0).area#heightIndex") --FS17 to FS19
XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "vehicle.workAreas.workArea(0)#foldMinLimit", "vehicle.workAreas.workArea(0).folding#minLimit") --FS17 to FS19
XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "vehicle.workAreas.workArea(0)#foldMaxLimit", "vehicle.workAreas.workArea(0).folding#maxLimit") --FS17 to FS19
XMLUtil.checkDeprecatedXMLElements(self.xmlFile, self.configFileName, "vehicle.workAreas.workArea(0)#refNodeIndex", "vehicle.workAreas.workArea(0).groundReferenceNode#index") --FS17 to FS19

spec.workAreas = {}
local i = 0
while true do
local key = string.format("vehicle.workAreas.workArea(%d)", i)
if not hasXMLProperty(self.xmlFile, key) then
break
end
local workArea = {}
if self:loadWorkAreaFromXML(workArea, self.xmlFile, key) then
table.insert(spec.workAreas, workArea)
workArea.index = #spec.workAreas
end
i = i + 1
end

spec.workAreaByType = {}
for _, area in pairs(spec.workAreas) do
if spec.workAreaByType[area.type] == nil then
spec.workAreaByType[area.type] = {}
end
table.insert(spec.workAreaByType[area.type], area)
end
```

```
89
90  spec.lastAccessedFarmlandOwner = 0
91  spec.showFarmlandNotOwnedWarning = false
92  end
```

onUpdateTick

Description

Called on update tick

Definition

onUpdateTick(float dt, boolean isActiveForInput, boolean isSelected)

Arguments

float	dt	time since last call in ms
boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
127 function WorkArea:onUpdateTick(dt, isActiveForInput, isSelected)
128 local spec = self.spec_workArea
129 SpecializationUtil.raiseEvent(self, "onStartWorkAreaProcessing",
    dt, spec.workAreas)
130
131 spec.showFarmlandNotOwnedWarning = false
132 -- do not reset last accessed farmland owner
133 -- spec.lastAccessedFarmlandOwner = 0
134 local hasProcessed = false
135
136 for _,workArea in ipairs(spec.workAreas) do
137 if workArea.type ~= WorkAreaType.AUXILIARY then
138 workArea.lastWorkedHectares = 0
139
140 local isAreaActive = self:getIsWorkAreaActive(workArea)
141 if isAreaActive and workArea.requiresOwnedFarmland then
142 local isOwned = false
143 local farmId = self:getActiveFarm()
144 if farmId == nil then -- Shop
145 farmId = AccessHandler.EVERYONE
146 end
147
148 local xs,_,zs = getWorldTranslation(workArea.start)
149 local isAccessible, farmlandOwner =
    self:getIsAccessibleAtWorldPosition(farmId, xs, zs,
    workArea.type)
150 if isAccessible then
151 if farmlandOwner ~= nil then
```

```
152 spec.lastAccessedFarmlandOwner = farmlandOwner
153 end
154 isOwned = true
155 else
156 local xw,_,zw = getWorldTranslation(workArea.width)
157 if self:getIsAccessibleAtWorldPosition(farmId, xw, zw,
    workArea.type) then
158 isOwned = true
159 else
160 local xh,_,zh = getWorldTranslation(workArea.height)
161 if self:getIsAccessibleAtWorldPosition(farmId, xh, zh,
    workArea.type) then
162 isOwned = true
163 else
164 local x = xw + (xh - xs)
165 local z = zw + (zh - zs)
166 if self:getIsAccessibleAtWorldPosition(farmId, x, z,
    workArea.type) then
167 isOwned = true
168 end
169 end
170 end
171 end
172
173 if not isOwned then
174 spec.showFarmlandNotOwnedWarning = true
175 isAreaActive = false
176 end
177 end
178
179 if isAreaActive then
180 if workArea.preprocessingFunction ~= nil then
181 workArea.preprocessingFunction(self, workArea, dt)
182 end
183
184 if workArea.processingFunction ~= nil then
185 local realArea, _ = workArea.processingFunction(self, workArea,
    dt)
186
187 workArea.lastWorkedHectares = MathUtil.areaToHa(realArea,
    g_currentMission:getFruitPixelsToSqm()) -- 4096px are mapped to
    2048m
```

```
188
189 if workArea.lastWorkedHectares > 0 then
190 self:setWorkAreaProcessingTime(workArea, g_currentMission.time)
191 end
192
193 -- Adding an area of interest for the wildlife spawners / keep
    (xw,zw) and (xh,zh)
194 if g_wildlifeSpawnerManager ~= nil and realArea > 0 then
195 local workAreaType =
    g_workAreaTypeManager:getWorkAreaTypeByIndex(workArea.type)
196 if workAreaType.attractWildlife then
197 local xw,_,zw = getWorldTranslation(workArea.width)
198 local xh,_,zh = getWorldTranslation(workArea.height)
199
200 local radius = 3.0
201 local posX = 0.5 * xw + 0.5 * xh
202 local posZ = 0.5 * zw + 0.5 * zh
203 local lifeTime = 0
204 g_wildlifeSpawnerManager:addAreaOfInterest(lifeTime, posX, posZ,
    radius)
205 end
206 end
207 end
208
209 if workArea.postprocessingFunction ~= nil then
210 workArea.postprocessingFunction(self, workArea, dt)
211 end
212
213 hasProcessed = true
214 end
215 end
216 end
217
218 SpecializationUtil.raiseEvent(self, "onEndWorkAreaProcessing",
    dt, hasProcessed)
219 end
```

onDraw

Description

Called on draw

Definition

onDraw(boolean isActiveForInput, boolean isSelected)

Arguments

boolean	isActiveForInput	true if vehicle is active for input
boolean	isSelected	true if vehicle is selected

Code

```
225 function WorkArea:onDraw(isActiveForInput, isSelected)
226 local spec = self.spec_workArea
227 if spec.showFarmlandNotOwnedWarning then
228 -- If this vehicle is a mission vehicle, the land-not-owned warning only
    the mission field
229 -- that could be because the land is unowned by the player but not for,
    used on owned land,
230 -- which is not allowed to prevent cheating with free vehicles. Therefor
    vehicle, we
231 -- show a different warning
232 if self.propertyState == Vehicle.PROPERTY_STATE_MISSION then
233 g_currentMission:showBlinkingWarning(g_i18n:getText("warning_cantUseMiss
234 else
235 g_currentMission:showBlinkingWarning(g_i18n:getText("warning_youDontHave
236 end
237 end
238 end
```

loadWorkAreaFromXML

Description

Loads work areas from xml

Definition

loadWorkAreaFromXML(table workArea, integer xmlFile, string key)

Arguments

table	workArea	workArea
integer	xmlFile	id of xml object
string	key	key

Return Values

boolean	success	success

Code

```
246 function WorkArea:loadWorkAreaFromXML(workArea, xmlFile, key)
247 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName, key ..
    ".area#startIndex", key .. ".area#startNode") --FS17 to FS19
248 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName, key ..
    ".area#widthIndex", key .. ".area#widthNode") --FS17 to FS19
249 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName, key ..
    ".area#heightIndex", key .. ".area#heightNode") --FS17 to FS19
250
251 local start = I3DUtil.indexToObject(self.components, getXMLString(xmlFil
    key .. ".area#startNode"), self.i3dMappings)
```

```
252 local width = I3DUtil.indexToObject(self.components, getXMLString(xmlFil
    key .. ".area#widthNode"), self.i3dMappings)
253 local height = I3DUtil.indexToObject(self.components, getXMLString(xmlFi
    key .. ".area#heightNode"), self.i3dMappings)
254
255 if start ~= nil and width ~= nil and height ~= nil then
256 local areaTypeStr = getXMLString(xmlFile, key .."#type")
257 workArea.type = g_workAreaTypeManager:getWorkAreaTypeIndexByName(areaTyp
    or WorkAreaType.DEFAULT
258
259 if workArea.type == nil then
260 g_logManager:xmlWarning(self.configFileName, "Invalid workArea type '%s'
    workArea '%s'!", areaTypeStr, key)
261 return false
262 end
263
264 workArea.isSynchronized = Utils.getNoNil(getXMLBool(xmlFile,
    key.."#isSynchronized"), true)
265 workArea.requiresGroundContact = Utils.getNoNil(getXMLBool(xmlFile, key
    "#requiresGroundContact"), true)
266
267 if workArea.type ~= WorkAreaType.AUXILIARY then
268
269 if workArea.requiresGroundContact then
270 XMLUtil.checkDeprecatedXMLElements(xmlFile, self.configFileName, key ..
    "#refNodeIndex", key .. ".groundReferenceNode#index") --FS17 to FS19
271 local groundReferenceNodeIndex = getXMLInt(xmlFile, key
    ..".groundReferenceNode#index")
272 if groundReferenceNodeIndex == nil then
273 g_logManager:xmlWarning(self.configFileName, "Missing groundReference
    'groundReferenceNode#index' for workArea '%s'. Add
    requiresGroundContact=\"false\" if groundContact is not required!", key)
274 return false
275 end
276 local groundReferenceNode =
    self:getGroundReferenceNodeFromIndex(groundReferenceNodeIndex)
277 if groundReferenceNode ~= nil then
278 workArea.groundReferenceNode = groundReferenceNode
279 else
280 g_logManager:xmlWarning(self.configFileName, "Invalid groundReferenceNod
    index for workArea '%s'!", key)
281 return false
282 end
```

```
283 end
284
285 workArea.disableBackwards = Utils.getNoNil(getXMLBool(xmlFile, key ..
    "#disableBackwards"), true)
286
287 workArea.functionName = getXMLString(xmlFile, key .. "#functionName")
288 if workArea.functionName == nil then
289 g_logManager:xmlWarning(self.configFileName, "Missing 'functionName' for
    workArea '%s'!", key)
290 return false
291 else
292 if self[workArea.functionName] == nil then
293 g_logManager:xmlWarning(self.configFileName, "Given functionName '%s' no
    defined. Please add missing function or specialization!",
    tostring(workArea.functionName))
294 return false
295 end
296 workArea.processingFunction = self[workArea.functionName]
297 end
298
299 workArea.preprocessFunctionName = getXMLString(xmlFile, key ..
    "#preprocessFunctionName")
300 if workArea.preprocessFunctionName ~= nil then
301 if self[workArea.preprocessFunctionName] == nil then
302 g_logManager:xmlWarning(self.configFileName, "Given preprocessFunctionNa
    '%s' not defined. Please add missing function or specialization!",
    tostring(workArea.preprocessFunctionName))
303 return false
304 end
305 workArea.preprocessingFunction = self[workArea.preprocessFunctionName]
306 end
307
308 workArea.postprocessFunctionName = getXMLString(xmlFile, key ..
    "#postprocessFunctionName")
309 if workArea.postprocessFunctionName ~= nil then
310 if self[workArea.postprocessFunctionName] == nil then
311 g_logManager:xmlWarning(self.configFileName, "Given postprocessFunctionN
    '%s' not defined. Please add missing function or specialization!",
    tostring(workArea.postprocessFunctionName))
312 return false
313 end
314 workArea.postprocessingFunction = self[workArea.postprocessFunctionName]
315 end
```

```lua

workArea.requiresOwnedFarmland = Utils.getNoNil(getXMLBool(xmlFile, key.."#requiresOwnedFarmland"), true)
end

workArea.lastProcessingTime = 0
workArea.start = start
workArea.width = width
workArea.height = height

return true
end

return false
end

function WorkArea:getWorkAreaByIndex(workAreaIndex)
local spec = self.spec_workArea
return spec.workAreas[workAreaIndex]
end

---Returns true if work area is active
-- @param table workArea workArea
-- @return boolean isActive work area is active
-- @includeCode
function WorkArea:getIsWorkAreaActive(workArea)

local isActive = true
if workArea.requiresGroundContact == true and workArea.groundReferenceNodenil then
isActive = self:getIsGroundReferenceNodeActive(workArea.groundReferenceN
end

if isActive and workArea.disableBackwards then
isActive = isActive and self.movingDirection > 0
end

return isActive
end

---Sets work area processing time
```

```lua
-- @param table workArea workArea
-- @param float time time
-- @includeCode
function WorkArea:setWorkAreaProcessingTime(workArea, time)
workArea.lastProcessingTime = time
end

---Returns true if work area is processing
-- @param table workArea workArea
-- @includeCode
function WorkArea:getIsWorkAreaProcessing(workArea)
return workArea.lastProcessingTime + 200 >= g_currentMission.time
end

---Returns positions of active work areas by type
-- @param integer areaType area type
-- @param boolean needsFieldProperty needs field property
-- @return table workAreasSend work areas send
-- @return boolean showFarmlandNotOwnedWarning show field not owned warn
-- @return float area size of areas
-- @includeCode
function WorkArea:getTypedNetworkAreas(areaType, needsFieldProperty)
local workAreasSend = {}
local area = 0
local typedWorkAreas = self:getTypedWorkAreas(areaType)
local showFarmlandNotOwnedWarning = false

for _, workArea in pairs(typedWorkAreas) do
if self:getIsWorkAreaActive(workArea) then
local x,_,z = getWorldTranslation(workArea.start)

local isAccessible = not needsFieldProperty
if needsFieldProperty then
local farmId = g_currentMission:getFarmId()
isAccessible = g_currentMission.accessHandler:canFarmAccessLand(farmId,
or g_missionManager:getIsMissionWorkAllowed(farmId, x, z, areaType)
end

if isAccessible then
local x1,_,z1 = getWorldTranslation(workArea.width)
```

```
394 local x2,_,z2 = getWorldTranslation(workArea.height)
395 area = area + math.abs((z1-z)*(x2-x) - (x1-x)*(z2-z))
396 table.insert(workAreasSend, {x,z,x1,z1,x2,z2})
397 else
398 showFarmlandNotOwnedWarning = true
399 end
400 end
401 end
402
403 return workAreasSend, showFarmlandNotOwnedWarning, area
404 end
405
406 ---Returns areas by type
407 -- @param integer areaType area type
408 -- @return table workAreas work areas
409 -- @includeCode
410 function WorkArea:getTypedWorkAreas(areaType)
411 local spec = self.spec_workArea
412 return Utils.getNoNil(spec.workAreaByType[areaType], {})
413 end
414
415 ---Returns if at least one area from type is active
416 -- @param integer areaType area type
417 -- @return boolean isActive isActive
418 -- @return table typedWorkAreas areas from given type
419 -- @includeCode
420 function WorkArea:getIsTypedWorkAreaActive(areaType)
421 local isActive = false
422 local typedWorkAreas = self:getTypedWorkAreas(areaType)
423 for _, workArea in pairs(typedWorkAreas) do
424 if self:getIsWorkAreaActive(workArea) then
425 isActive = true
426 break
427 end
428 end
429
430 return isActive, typedWorkAreas
431 end
432
433 function WorkArea:getIsFarmlandNotOwnedWarningShown()
```

```lua
return self.spec_workArea.showFarmlandNotOwnedWarning
end

---Loads speed rotating parts from xml
-- @param table speedRotatingPart speedRotatingPart
-- @param integer xmlFile id of xml object
-- @param string key key
-- @return boolean success success
-- @includeCode
function WorkArea:loadSpeedRotatingPartFromXML(superFunc, speedRotatingP
xmlFile, key)
if not superFunc(self, speedRotatingPart, xmlFile, key) then
return false
end

speedRotatingPart.workAreaIndex = getXMLInt(xmlFile, key.."#workAreaInde

return true
end

---Returns true if speed rotating part is active
-- @param table speedRotatingPart speedRotatingPart
-- @return boolean isActive speed rotating part is active
-- @includeCode
function WorkArea:getIsSpeedRotatingPartActive(superFunc, speedRotatingP
if speedRotatingPart.workAreaIndex ~= nil then
local spec = self.spec_workArea
local workArea = spec.workAreas[speedRotatingPart.workAreaIndex]
if workArea == nil then
speedRotatingPart.workAreaIndex = nil
g_logManager:xmlWarning(self.configFileName, "Invalid workAreaIndex '%s'
Indexing starts with 1!", tostring(speedRotatingPart.workAreaIndex))
return true
end

if not
self:getIsWorkAreaProcessing(spec.workAreas[speedRotatingPart.workAreaIn
then
return false
end
end
```

```
471
472 return superFunc(self, speedRotatingPart)
473 end
474
```

AnimalFoodManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
48 function AnimalFoodManager:new(customMt)
49 local self = AbstractManager:new(customMt or AnimalFoodManager_mt)
50
51 return self
52 end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
56 function AnimalFoodManager:initDataStructures()
57 self.foodGroups = {}
58 self.foodMixtures = {}
59 self.animalFoodMixtures = {}
60 end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
65 function AnimalFoodManager:loadMapData(xmlFile, missionInfo)
66 AnimalFoodManager:superClass().loadMapData(self)
67
```

```
68  local filename = Utils.getFilename(getXMLString(xmlFile,
    "map.husbandryFood#filename"), g_currentMission.baseDirectory)
69  if filename == nil or filename == "" then
70  print("Error: Could not load husbandry food configuration file '"
    .. tostring(filename) .. "'!")
71  return false
72  end
73  local foodGroupsLoaded = false
74  local mixturesLoaded = false
75  local foodGroupsNormalized = false
76  local mixturesNormalized = false
77  local animalFoodXmlFile = loadXMLFile("animalFood", filename)
78
79  if animalFoodXmlFile ~= nil then
80  foodGroupsLoaded = self:loadFoodGroups(animalFoodXmlFile)
81  mixturesLoaded = self:loadMixtures(animalFoodXmlFile)
82  foodGroupsNormalized = self:normalizeFoodGroupWeights()
83  mixturesNormalized = self:normalizeMixtureWeights()
84  delete(animalFoodXmlFile)
85  end
86  return foodGroupsLoaded and mixturesLoaded and
    foodGroupsNormalized and mixturesNormalized
87  end
```

loadFoodGroups

Description

Loads food groups

Definition

loadFoodGroups(table xmlFile)

Arguments

table xmlFile

Return Values

bool true if successful

Code

```
93  function AnimalFoodManager:loadFoodGroups(xmlFile)
94  local i = 0
95
96  while true do
97  local animalKey =
    string.format("animalFood.animals.animalFoodGroups(%d)", i)
98  if not hasXMLProperty(xmlFile, animalKey) then
99  break
100 end
```

```
101 local animalType = getXMLString(xmlFile, animalKey.."#type")
102 if animalType ~= nil then
103 animalType = animalType:upper()
104 self.foodGroups[animalType] = {}
105 self.foodGroups[animalType].content = {}
106 self.foodGroups[animalType].consumptionType =
    AnimalFoodManager.FOOD_CONSUME_TYPE_SERIAL
107
108 local foodProcessTypeString = getXMLString(xmlFile, animalKey ..
    "#consumptionType")
109 foodProcessTypeString = foodProcessTypeString:upper()
110 if foodProcessTypeString == "PARALLEL" then
111 self.foodGroups[animalType].consumptionType =
    AnimalFoodManager.FOOD_CONSUME_TYPE_PARALLEL
112 end
113
114 local j = 0
115 while true do
116 local groupKey = string.format("%s.foodGroup(%d)", animalKey, j)
117 if not hasXMLProperty(xmlFile, groupKey) then
118 break
119 end
120
121 local foodGroup = {}
122 foodGroup.title =
    g_i18n:convertText(Utils.getNoNil(getXMLString(xmlFile,
    groupKey.."#title"), ""))
123 foodGroup.productionWeight = Utils.getNoNil(getXMLFloat(xmlFile,
    groupKey.."#productionWeight"), 0.0)
124 foodGroup.fillTypes = {}
125 local fillTypesStr = Utils.getNoNil(getXMLString(xmlFile,
    groupKey.."#fillTypes"), "")
126 local warning = string.format("Warning: FillType '%s' undefined
    for foodGroups of '%s'. Ignoring it!", tostring(fillTypeName),
    tostring(animalType))
127 foodGroup.fillTypes =
    g_fillTypeManager:getFillTypesByNames(fillTypesStr, warning)
128 table.insert(self.foodGroups[animalType].content, foodGroup)
129 j = j + 1
130 end
131 end
132 i = i + 1
133 end
```

```
134 return true
135 end
```

loadMixtures

Description

Loads food groups

Definition

loadMixtures(table xmlFile)

Arguments

table xmlFile

Return Values

bool true if successful

Code

```
141 function AnimalFoodManager:loadMixtures(xmlFile)
142 local i = 0
143
144 while true do
145 local mixtureKey =
    string.format("animalFood.foodMixtures.foodMixture(%d)", i)
146 if not hasXMLProperty(xmlFile, mixtureKey) then
147 break
148 end
149
150 local mixtureFillTypeName = Utils.getNoNil(getXMLString(xmlFile,
    mixtureKey.."#fillType"), "")
151 local animalType = Utils.getNoNil(getXMLString(xmlFile,
    mixtureKey.."#type"), "")
152 if animalType ~= nil then
153 animalType = animalType:upper()
154 if self.animalFoodMixtures[animalType] == nil then
155 self.animalFoodMixtures[animalType] = {}
156 end
157
158 local mixtureFillType =
    g_fillTypeManager:getFillTypeByName(mixtureFillTypeName)
159 if mixtureFillType ~= nil then
160 local fillType = mixtureFillType.index
161 table.insert(self.animalFoodMixtures[animalType], fillType)
162 self.foodMixtures[fillType] = {}
163 self.foodMixtures[fillType].ingredients = {}
164
165 local j = 0
166 while true do
```

```
167 local ingredientKey = string.format("%s.ingredient(%d)", mixtureKey, j)
168 if not hasXMLProperty(xmlFile, ingredientKey) then
169 break
170 end
171
172 local ingredient = {}
173 local warning = string.format("Warning: FillType '%s' undefined for mixture of '%s'. Ignoring it!", tostring(fillTypeName), tostring(mixtureFillTypeName))
174 local fillTypesStr = Utils.getNoNil(getXMLString(xmlFile, ingredientKey.."#fillTypes"), "")
175
176 ingredient.fillTypes = g_fillTypeManager:getFillTypesByNames(fillTypesStr, warning)
177 ingredient.weight = Utils.getNoNil(getXMLFloat(xmlFile, ingredientKey.."#weight"), 0.0)
178 table.insert(self.foodMixtures[fillType].ingredients, ingredient)
179 j = j + 1
180 end
181 else
182 print("Warning: FillType '"..tostring(fillTypeName).."' undefined for mixtures. Ignoring it!")
183 return false
184 end
185 end
186 i = i + 1
187 end
188 return true
189 end
```

normalizeFoodGroupWeights

Description

Normalize food groups weight if parallel consumption type

Definition

normalizeFoodGroupWeights()

Return Values

bool true if normalized

Code

```
194 function AnimalFoodManager:normalizeFoodGroupWeights()
195 local oneGroupHasNoWeight = false
196 for _, animalFoodGroup in pairs(self.foodGroups) do
197 local sumWeigths = 0
```

```
198  for _, foodGroup in pairs(animalFoodGroup.content) do
199  sumWeigths = sumWeigths + foodGroup.productionWeight
200  end
201
202  if sumWeigths > 0 then
203  if animalFoodGroup.consumptionType ==
     AnimalFoodManager.FOOD_CONSUME_TYPE_PARALLEL then
204  for _, foodGroup in pairs(animalFoodGroup.content) do
205  foodGroup.productionWeight = foodGroup.productionWeight /
     sumWeigths
206  end
207  end
208  else
209  oneGroupHasNoWeight = true
210  end
211  end
212  return not oneGroupHasNoWeight
213  end
```

normalizeMixtureWeights

Description

Normalize mixture's ingredient weight

Definition

normalizeMixtureWeights()

Return Values

bool true if normalized

Code

```
218  function AnimalFoodManager:normalizeMixtureWeights()
219  local oneGroupHasNoWeight = false
220  for _, mixture in pairs(self.foodMixtures) do
221  local sumWeigths = 0
222  for _, ingredient in pairs(mixture.ingredients) do
223  sumWeigths = sumWeigths + ingredient.weight
224  end
225  if sumWeigths > 0 then
226  for _, ingredient in pairs(mixture.ingredients) do
227  ingredient.weight = ingredient.weight / sumWeigths
228  end
229  else
230  oneGroupHasNoWeight = true
231  end
232  end
```

```
233 return not oneGroupHasNoWeight
234 end
```

getFoodGroupByAnimalIndex

Description

Retrieves food group by animal index

Definition

getFoodGroupByAnimalIndex(integer animalIndex)

Arguments

integer animalIndex

Return Values

table returns a food group or nil if nothing is found

Code

```
240 function AnimalFoodManager:getFoodGroupByAnimalIndex(animalIndex)
241 if animalIndex ~= nil then
242 local animalType = g_animalManager:getAnimalType(animalIndex)
243 return self.foodGroups[animalType].content
244 end
245 return nil
246 end
```

getFoodGroupByAnimalType

Description

Retrieves food group by animal type

Definition

getFoodGroupByAnimalType(string animalType)

Arguments

string animalType

Return Values

table returns a food group or nil if nothing is found

Code

```
252 function AnimalFoodManager:getFoodGroupByAnimalType(animalType)
253 if animalType ~= nil then
254 return self.foodGroups[animalType].content
255 end
256 return nil
257 end
```

getFoodMixturesByAnimalType

Description

Retrieves food mixtures by animal type

Definition

getFoodMixturesByAnimalType(string animalType)

Arguments

string animalType

Return Values

table returns a food mixture or nil if nothing is found

Code

```
263 function AnimalFoodManager:getFoodMixturesByAnimalType(animalType)
264 if animalType ~= nil then
265 return self.animalFoodMixtures[animalType]
266 end
267 return nil
268 end
```

getFoodConsumptionTypeByAnimalType

Description

Retrieve food consumption type by animal type.

Definition

getFoodConsumptionTypeByAnimalType(string animalType)

Arguments

string animalType Animal type

Return Values

int Food consumption type, one of [AnimalFoodManager.FOOD_CONSUME_TYPE_SERIAL | AnimalFoodManager.FOOD_CONSUME_TYPE_PARALLEL]

Code

```
278 function AnimalFoodManager:getFoodConsumptionTypeByAnimalType(animalType)
279 if animalType ~= nil then
280 return self.foodGroups[animalType].consumptionType
281 else
282 return nil
283 end
284 end
```

getFoodGroupProductionWeight

Description

Retrieves production weight of a fill type from the food group of an animal type

Definition

getFoodGroupProductionWeight(string animalIndex, integer fillType)

Arguments

string animalIndex

integer fillType

Return Values

float production weight

Code

```
291 function AnimalFoodManager:getFoodGroupProductionWeight(animalType,
    fillType)
```

```
292 local foodGroup = self:getFoodGroupByFillType(animalType, fillType)
293
294 if foodGroup ~= nil then
295 return foodGroup.productionWeight
296 end
297 return 0.0
298 end
```

getFoodGroupByFillType

Description

Retrieves food group by fill type

Definition

getFoodGroupByFillType(string animalIndex, integer fillType)

Arguments

string animalIndex

integer fillType

Return Values

table returns a food group or nil if not found

Code

```
305 function AnimalFoodManager:getFoodGroupByFillType(animalType, fillTypeIndex)
306 local animalFoodGroups = self.foodGroups[animalType]
307
308 if animalFoodGroups ~= nil then
309 for _, foodGroup in pairs(animalFoodGroups.content) do
310 for _, foodGroupFillTypeIndex in pairs(foodGroup.fillTypes) do
311 if foodGroupFillTypeIndex == fillTypeIndex then
312 return foodGroup
313 end
314 end
315 end
316 end
317 return nil
318 end
```

consumeFood

Description

Consumes food(fillType) in fillLevels and returns the production weight associated to it

Definition

consumeFood(string animalType, float amountToConsume., table fillLevels)

Arguments

string animalType

float amountToConsume. Is modified, is zero all food is consumed. If not enough food, amount is > 0.

table	fillLevels	array of floats indexed to fillTypeIndices. Is modified depending on how much food is conumed.

Return Values

float returns a production weight between 0 and 1

Code

```
327  function AnimalFoodManager:consumeFood(animalType, amountToConsume, fillLevels, consumedFood)
328  local production = 0.0
329  local animalFoodGroups = self.foodGroups[animalType]
330
331  -- print(string.format("-- [AnimalFoodManager:consumeFood] animalType(%s) animalFoodGroups(%s)", tostring(animalType), tostring(animalFoodGroups)))
332  if animalFoodGroups ~= nil then
333  -- print(string.format("-- [AnimalFoodManager:consumeFood] consumptionType(%d)", animalFoodGroups.consumptionType))
334  if animalFoodGroups.consumptionType == AnimalFoodManager.FOOD_CONSUME_TYPE_SERIAL then
335  production = self:consumeFoodSerially(amountToConsume, animalFoodGroups.content, fillLevels, consumedFood)
336  elseif animalFoodGroups.consumptionType == AnimalFoodManager.FOOD_CONSUME_TYPE_PARALLEL then
337  production = self:consumeFoodParallelly(amountToConsume, animalFoodGroups.content, fillLevels, consumedFood)
338  end
339  end
340  return production
341  end
```

changeFillLevels

Description

Definition

changeFillLevels()

Code

```
345  function AnimalFoodManager:changeFillLevels(fillDelta, fillLevels, fillTypeIndex)
346  local old = fillLevels[fillTypeIndex]
347  fillLevels[fillTypeIndex] = math.max(0, old + fillDelta)
348  local new = fillLevels[fillTypeIndex]
349  return new - old
350  end
```

consumeFoodGroup

Description

Definition

consumeFoodGroup()

Code

```
354 function AnimalFoodManager:consumeFoodGroup(foodGroup, amount, fillLevels, consumedFood)
355 for _, fillTypeIndex in pairs(foodGroup.fillTypes) do
356 if fillLevels[fillTypeIndex] ~= nil then
357 local currentFillLevel = fillLevels[fillTypeIndex]
358 local amountToConsume = -math.min(amount, currentFillLevel)
359 local deltaConsumed = self:changeFillLevels(amountToConsume, fillLevels, fillTypeIndex)
360
361 -- print(string.format("-- [AnimalFoodManager:consumeFoodGroup] foodGroup(%s) amount(%.3f) amountToConsume(%.3f) deltaConsumed(%.3f) fillTypeIndex(%s/%d) newAmount(%.3f)",
362 -- foodGroup.title,
363 -- amount,
364 -- amountToConsume,
365 -- deltaConsumed,
366 -- g_fillTypeManager:getFillTypeNameByIndex(fillTypeIndex), fillTypeIndex,
367 -- math.max(amount + deltaConsumed, 0.0)
368 -- ))
369 amount = math.max(amount + deltaConsumed, 0.0)
370 consumedFood[fillTypeIndex] = -deltaConsumed
371 if amount == 0.0 then
372 -- print(string.format("-- [AnimalFoodManager:consumeFoodGroup][A] EXIT"))
373 return amount
374 end
375 end
376 end
377 -- print(string.format("-- [AnimalFoodManager:consumeFoodGroup][B] EXIT restAmount(%.3f) ", amount))
378 return amount
379 end
```

consumeFoodSerially

Description

Definition

consumeFoodSerially()

Code

```
383 function AnimalFoodManager:consumeFoodSerially(amount, foodGroups, fillLevels, consumedFood)
384 local productionWeight = 0.0
```

```lua
local totalAmountToConsume = amount

-- print(string.format("--[AnimalFoodManager:consumeFoodSerially] ENTER"))
for _, foodGroup in ipairs(foodGroups) do
local oldAmount = amount
amount = self:consumeFoodGroup(foodGroup, amount, fillLevels, consumedFood)
local deltaProdWeight = ((oldAmount - amount) / totalAmountToConsume) * foodGroup.productionWeight
-- print(string.format("--[AnimalFoodManager:consumeFoodSerially] deltaProdWeight(%.3f) amount(%.3f) oldAmount(%.3f) totalAmountToConsume(%.3f) foodGroup.productionWeight(%.3f)",
-- deltaProdWeight,
-- amount,
-- oldAmount,
-- totalAmountToConsume,
-- foodGroup.productionWeight
-- ))

productionWeight = productionWeight + deltaProdWeight
end
-- print(string.format("--[AnimalFoodManager:consumeFoodSerially] EXIT: productionWeight(%.3f)", productionWeight))
return productionWeight
end
```

getTotalFillLevelInGroupByFillTypeIndex

Description

Definition

getTotalFillLevelInGroupByFillTypeIndex()

Code

```lua
function AnimalFoodManager:getTotalFillLevelInGroupByFillTypeIndex(animalType, fillLevels, fillTypeIndex)
local foodGroup = self:getFoodGroupByFillType(animalType, fillTypeIndex)

if foodGroup ~= nil then
return self:getTotalFillLevelInGroup(foodGroup, fillLevels)
end
return 0.0
end
```

getTotalFillLevelInGroup

Description

Definition

getTotalFillLevelInGroup()

Code

```lua
function AnimalFoodManager:getTotalFillLevelInGroup(foodGroup, fillLevels)
local totalFillLevel = 0.0
for _, fillTypeIndex in pairs(foodGroup.fillTypes) do
totalFillLevel = totalFillLevel + fillLevels[fillTypeIndex]
end
return totalFillLevel
end
```

consumeFoodParallelly

Description

Definition

consumeFoodParallelly()

Code

```lua
function AnimalFoodManager:consumeFoodParallelly(amount, foodGroups, fillLevels, consumedFood)
local productionWeight = 0.0
local totalAmountToConsume = amount
local hasFoodToConsume = amount > 0.0
local nbActiveGroups = #foodGroups
-- print(string.format("--[AnimalFoodManager:consumeFoodParallelly] CALLED!"))

while ((nbActiveGroups > 0) and hasFoodToConsume) do
-- print(string.format("--[AnimalFoodManager:consumeFoodParallelly] amount(%.3f)", amount))
local groupsToProcess = {}
local minAmountToConsume = -1.0

for _, foodGroup in pairs(foodGroups) do
local totalFillLevelInGroup = self:getTotalFillLevelInGroup(foodGroup, fillLevels)
if totalFillLevelInGroup > 0.0 then
if minAmountToConsume > 0.0 then
minAmountToConsume = math.min(totalFillLevelInGroup, minAmountToConsume)
else
minAmountToConsume = totalFillLevelInGroup
end
```

```
449 -- print(string.format("--
    [AnimalFoodManager:consumeFoodParallelly]
    totalFillLevelInGroup(%.3f)", totalFillLevelInGroup))
450 table.insert(groupsToProcess, foodGroup)
451 end
452 end
453 nbActiveGroups = #groupsToProcess
454 if nbActiveGroups > 0 then
455 local distributedAmount = amount / nbActiveGroups
456 minAmountToConsume = math.min(distributedAmount,
    minAmountToConsume)
457 -- print(string.format("--
    [AnimalFoodManager:consumeFoodParallelly] nbActiveGroups(%d)
    minAmountToConsume(%.3f)", nbActiveGroups, minAmountToConsume))
458 for _, foodGroup in pairs(groupsToProcess) do
459 local oldAmount = minAmountToConsume
460 local restAmount = self:consumeFoodGroup(foodGroup,
    minAmountToConsume, fillLevels, consumedFood)
461 local deltaProdWeight = ((oldAmount - restAmount) /
    distributedAmount) * foodGroup.productionWeight
462 productionWeight = productionWeight + deltaProdWeight
463 amount = amount - (minAmountToConsume + restAmount)
464 -- print(string.format("--
    [AnimalFoodManager:consumeFoodParallelly] amount(%.3f)
    restAmount(%.3f)", amount, restAmount))
465 end
466 end
467 hasFoodToConsume = (amount > 0.0)
468 end
469 -- print(string.format("--
    [AnimalFoodManager:consumeFoodParallelly]
    productionWeight(%.3f)", productionWeight))
470 -- print(string.format("--
    [AnimalFoodManager:consumeFoodParallelly] EXIT!"))
471 return productionWeight
472 end
```

AnimalHusbandry

Description

onCreate

Description

Keeping this function for backward compatibility. Called when when object is created by level designer.

Definition

onCreate(integer transformId)

Arguments

integer transformId id of the object

Code

```
35  function AnimalHusbandry:onCreate(transformId)
36  local typeName = getUserAttribute(transformId, "typeName")
37  local xmlFile = getUserAttribute(transformId, "xmlFile")
38
39  if typeName ~= nil and xmlFile ~= nil then
40  xmlFile = Utils.getFilename(xmlFile,
    g_currentMission.loadingMapBaseDirectory)
41
42  local husbandry = AnimalHusbandry:new(g_server ~= nil, g_client ~=
    nil)
43
44  self.baseDirectory = g_currentMission.loadingMapBaseDirectory
45  self.customEnvironment = g_currentMission.loadingMapModName
46
47  local x, y, z = getWorldTranslation(transformId)
48  local rx, ry, rz = getWorldRotation(transformId)
49
50  if husbandry:load(xmlFile, x, y, z, rx, ry, rz, false) then
51  g_currentMission:addOnCreateLoadedObject(husbandry)
52  g_currentMission:addOnCreateLoadedObjectToSave(husbandry)
53  husbandry:register(true)
54  husbandry:finalizePlacement()
55  else
56  husbandry:delete()
57  end
58  else
59  print("Error: missing typeName and xmlFile attribute for animal
    husbandry in "..getName(transformId))
60  end
61  end
```

new

Description

Creating instance and initializing member variables

Definition

new(boolean isServer, boolean isClient, table mt)

Arguments

boolean	isServer	is server
boolean	isClient	is client
table	mt	custom meta table

Return Values

table instance Instance of object

Code

```
69  function AnimalHusbandry:new(isServer, isClient, mt)
70  if mt == nil then
71  mt = AnimalHusbandry_mt
72  end
73  local self = Placeable:new(isServer, isClient, mt)
74
75  self.useMultiRootNode = true
76  self.modulesByName = {}
77  self.modulesById = {}
78  -- husbandry information
79  self.updateMinutes = 0
80  self.saveId = ""
81  self.hasStatistics = false
82  self.husbandryDirtyFlag = self:getNextDirtyFlag()
83
84  -- productivity
85  self.globalProductionFactor = 0.0
86
87  registerObjectClassName(self, "AnimalHusbandry")
88
89  return self
90  end
```

load

Description

Loads information from xml file and registers the instance to the current Mission

Definition

load(string xmlFilename, float x, float y, float z, float rx, float ry, float rz, initRandom)

Arguments

string	xmlFilename	XML filename
float	x	world x position
float	y	world y position
float	z	world z position
float	rx	x rotation in rad
float	ry	y rotation in rad
float	rz	z rotation in rad
initRandom		

Return Values

bool returns true if loading is fine

Code

```
103 function AnimalHusbandry:load(xmlFilename, x, y, z, rx, ry, rz,
    initRandom)
104 if not AnimalHusbandry:superClass().load(self, xmlFilename, x, y,
    z, rx, ry, rz, initRandom) then
105 return false
106 end
107
108 local success = self:loadModules(xmlFilename)
109
110 if success then
111 local xmlFile = loadXMLFile("AnimalHusbandryXML", xmlFilename)
112 local key = string.format("placeable.husbandry")
113 self.saveId = Utils.getNoNil(getXMLString(xmlFile, key ..
    "#saveId"), "Animals_" .. self:getAnimalType())
114 self.hasStatistics = Utils.getNoNil(getXMLBool(xmlFile, key ..
    "#hasStatistics"), false)
115 delete(xmlFile)
116 end
117 return success
118 end
```

loadModules

Description

Loads all modules declared in xml file

Definition

loadModules(table xmlFile)

Arguments

table xmlFile handle to the xml config file

Code

```
123 function AnimalHusbandry:loadModules(xmlFilename)
124 local xmlFile = loadXMLFile("TempXML", xmlFilename)
125 local key = string.format("placeable.husbandry")
126 local i = 0
127
128 if xmlFile == 0 then
129 return false
130 end
131 while true do
132 local moduleNameKey = string.format("%s.modules.module(%d)#name",
    key, i)
133 if not hasXMLProperty(xmlFile, moduleNameKey) then
134 break
135 end
```

```
136 local moduleName = getXMLString(xmlFile, moduleNameKey)
137 local newModule = HusbandryModuleBase.createModule(moduleName)
138
139 if newModule ~= nil then
140 local moduleConfigKey =
    string.format("%s.modules.module(%d).config", key, i)
141 local success = newModule:load(xmlFile, moduleConfigKey,
    self.nodeId, self)
142
143 if not success then
144 return false
145 end
146 self.modulesByName[moduleName] = newModule
147 table.insert(self.modulesById, newModule)
148 end
149 i = i + 1
150 end
151 delete(xmlFile)
152 return true
153 end
```

modulesFinalizePlacement

Description

Definition

modulesFinalizePlacement()

Return Values

bool returns true if module init after placement is fine

Code

```
158 function AnimalHusbandry:modulesFinalizePlacement()
159 for _, moduleInstance in pairs(self.modulesById) do
160 local success = moduleInstance:finalizePlacement()
161 if success ~= true then
162 return false
163 end
164 end
165 return true
166 end
```

getCanBePlacedAt

Description

Returns true if we can place a building

Definition

getCanBePlacedAt()

Return Values

Code

```
171  function AnimalHusbandry:getCanBePlacedAt(x, y, z, distance)
172  local canBePlaced = AnimalHusbandry:superClass().getCanBePlacedAt(self, x, y, z, distance)
173  return canBePlaced and not self:isSimilarHusbandryRegistered()
174  end
```

canBuy

Description

Returns true if we can place a building

Definition

canBuy()

Return Values

Code

```
179  function AnimalHusbandry:canBuy()
180  local canBuy = AnimalHusbandry:superClass().canBuy(self)
181  return canBuy and not self:isSimilarHusbandryRegistered(), g_i18n:getText("warning_onlyOneOfThisItemAllowedPerFarm")
182  end
```

finalizePlacement

Description

Called when player places the husbandry.

Definition

finalizePlacement()

Return Values

bool returns true if initialization is fine

Code

```
197  function AnimalHusbandry:finalizePlacement()
198  AnimalHusbandry:superClass().finalizePlacement(self)
199
200  local success = self:modulesFinalizePlacement() and self:registerHusbandryToMission()
201  if not success then
202  return false
203  end
204
205  if self.isServer then
206  g_currentMission.environment:addHourChangeListener(self)
207  end
208  return true
209  end
```

onSell

Description

Called on sell

Definition

onSell()

Code

```
213 function AnimalHusbandry:onSell()
214 AnimalHusbandry:superClass().onSell(self)
215 
216 for i = #self.modulesById, 1, -1 do
217 self.modulesById[i]:onSell()
218 end
219 
220 if self.isServer then
221 g_currentMission.environment:removeMinuteChangeListener(self)
222 end
223 end
```

delete

Description

Deletes instance

Definition

delete()

Code

```
227 function AnimalHusbandry:delete()
228 g_currentMission:removeOnCreateLoadedObjectToSave(self)
229 self:unregisterHusbandryToMission()
230 
231 for _, moduleInstance in pairs(self.modulesById) do
232 moduleInstance:delete()
233 end
234 
235 if self.isServer then
236 if self.palletSpawnerTriggerId ~= nil then
237 removeTrigger(self.palletSpawnerTriggerId)
238 end
239 end
240 
241 AnimalHusbandry:superClass().delete(self)
242 end
```

loadFromXMLFile

Description

Loads information from attributes and node. Retrives from xml file information from: initial amount of animals. manure, pallet, fill level, cleanliness.

Definition

loadFromXMLFile(table xmlFile, string key)

Arguments

table xmlFile XML file handler

string key XML base key

Code

```
248 function AnimalHusbandry:loadFromXMLFile(xmlFile, key)
249 AnimalHusbandry:superClass().loadFromXMLFile(self, xmlFile, key,
    nil)
250
251 self.globalProductionFactor = getXMLFloat(xmlFile,
    key.."#globalProductionFactor") or self.globalProductionFactor
252
253 local i = 0
254 while true do
255 local moduleKey = string.format("%s.module(%d)", key, i)
256 local moduleNameKey = string.format("%s#name", moduleKey)
257 if not hasXMLProperty(xmlFile, moduleNameKey) then
258 break
259 end
260 local moduleName = getXMLString(xmlFile, moduleNameKey)
261 local moduleInstance = self:getModuleByName(moduleName)
262
263 if moduleInstance ~= nil then
264 moduleInstance:loadFromXMLFile(xmlFile, moduleKey)
265 end
266 i = i + 1
267 end
268 return true
269 end
```

readStream

Description

Reads network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
291 function AnimalHusbandry:readStream(streamId, connection)
292 AnimalHusbandry:superClass().readStream(self, streamId,
    connection)
```

```
293
294 if connection:getIsServer() then
295 for _, moduleInstance in ipairs(self.modulesById) do
296 moduleInstance:readStream(streamId, connection)
297 end
298 end
299 end
```

writeStream

Description

Writes network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
305 function AnimalHusbandry:writeStream(streamId, connection)
306 AnimalHusbandry:superClass().writeStream(self, streamId,
    connection)
307 if not connection:getIsServer() then
308 for _, moduleInstance in ipairs(self.modulesById) do
309 moduleInstance:writeStream(streamId, connection)
310 end
311 end
312 end
```

readUpdateStream

Description

Read updates from network stream

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId network stream identification

integer timestamp

table connection connection information

Code

```
319 function AnimalHusbandry:readUpdateStream(streamId, timestamp,
    connection)
320 AnimalHusbandry:superClass().readUpdateStream(self, streamId,
    timestamp, connection)
321
322 if connection:getIsServer() then
323 if streamReadBool(streamId) then
```

```
324     for _, moduleInstance in ipairs(self.modulesById) do
325     moduleInstance:readUpdateStream(streamId, timestamp, connection)
326     end
327     self.globalProductionFactor = streamReadFloat32(streamId)
328     end
329     end
330     end
```

writeUpdateStream

Description

Write updates from network stream

Definition

writeUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId	network stream identification
table	connection connection information
integer dirtyMask	is used to check if we need to update

Code

```
337     function AnimalHusbandry:writeUpdateStream(streamId, connection,
        dirtyMask)
338     AnimalHusbandry:superClass().writeUpdateStream(self, streamId,
        connection, dirtyMask)
339     if not connection:getIsServer() then
340     local isDirty = bitAND(dirtyMask, self.husbandryDirtyFlag) ~= 0
341     if streamWriteBool(streamId, isDirty) then
342     for _, moduleInstance in ipairs(self.modulesById) do
343     moduleInstance:writeUpdateStream(streamId, connection, dirtyMask)
344     end
345     streamWriteFloat32(streamId, self.globalProductionFactor)
346     end
347     end
348     end
```

onIntervalModulesUpdate

Description

Update method for modules called when an interval has passed

Definition

onIntervalModulesUpdate(float dayInterval)

Arguments

float dayInterval

Code

```
366     function AnimalHusbandry:onIntervalModulesUpdate(dayInterval)
367     for _, moduleInstance in pairs(self.modulesById) do
368     moduleInstance:onIntervalUpdate(dayInterval)
```

```
369 end
370 end
```

hourChanged

Description

Called by the environment when an hour has changed. Updates: egg production, straw usage, food usage, production, animal reproduction, dirt, water usage, manure and pallet

Definition

hourChanged()

Code

```
374 function AnimalHusbandry:hourChanged()
375 for _, moduleInstance in pairs(self.modulesById) do
376 moduleInstance:onHourChanged()
377 end
378
379 if self:getNumOfAnimals() > 0 then
380 if self.isServer then
381 local dayInterval = 1 / 24
382
383 self:onIntervalModulesUpdate(dayInterval)
384 self:updateGlobalProductionFactor()
385 self:raiseDirtyFlags(self.husbandryDirtyFlag)
386 end
387 end
388 end
```

updateGlobalProductionFactor

Description

Definition

updateGlobalProductionFactor()

Code

```
392 function AnimalHusbandry:updateGlobalProductionFactor()
393 self.globalProductionFactor = 0.0
394 if self:hasWater() then
395 if self:hasStraw() then
396 self.globalProductionFactor = 0.1
397 end
398 local foodSpillageFactor = self:getFoodSpillageFactor() or 0
399 self.globalProductionFactor = self.globalProductionFactor + (1.0
    - foodSpillageFactor) * 0.05
400 self.globalProductionFactor = self.globalProductionFactor +
    self:getFoodProductionFactor() * 0.85
401 -- test: self.globalProductionFactor = 1.0
402 end
```

```
403 end
```

getFluidStatsText

Description

Format text to display fill level

Definition

getFluidStatsText(float fillLevel, bool alwaysExact)

Arguments

float fillLevel

bool alwaysExact if true will change text accordingly

Return Values

string fill level

Code

```
410 function AnimalHusbandry:getFluidStatsText(fillLevel,
    alwaysExact)
411 fillLevel = Utils.getNoNil(fillLevel,0)
412 if self.isServer or alwaysExact or fillLevel == 0 then
413 return string.format("%1.0f",
    math.floor(g_i18n:getFluid(fillLevel)))
414 else
415 return string.format("~%1.0f",
    math.floor(g_i18n:getFluid(fillLevel)))
416 end
417 end
```

collectPickObjects

Description

Collect pick objects / empty as we do not use pick object for animal husbandry

Definition

collectPickObjects(integer node)

Arguments

integer node node id

Code

```
422 function AnimalHusbandry:collectPickObjects(node)
423 end
```

getModuleById

Description

Definition

getModuleById()

Code

```
431 function AnimalHusbandry:getModuleById(moduleId)
432 local moduleinstance = self.modulesById[moduleId]
433
434 return moduleinstance
435 end
```

getModuleByName

Description

Definition

getModuleByName()

Code

```
439 function AnimalHusbandry:getModuleByName(moduleName)
440 if HusbandryModuleBase.hasModule(moduleName) then
441 return self.modulesByName[moduleName]
442 end
443 g_logManager:warning("Animal module '%s' is not registered!",
    tostring(moduleName))
444 return nil
445 end
```

setModuleParameters

Description

Initializes capacity and daily usage parameters for the specified module. Called by the animal module

Definition

setModuleParameters(string moduleName, float capacity, float usagePerDay)

Arguments

string	moduleName
float	capacity
float	usagePerDay

Code

```
452 function AnimalHusbandry:setModuleParameters(moduleName,
    capacity, usagePerDay)
453 local moduleInstance = self:getModuleByName(moduleName)
454
455 if moduleInstance ~= nil then
456 moduleInstance:setCapacity(capacity)
457 moduleInstance:setSingleAnimalUsagePerDay(usagePerDay)
458 end
459 end
```

getNumOfAnimals

Description

Definition

getNumOfAnimals()

Return Values

integer total animals. Default is zero

Code

```
464 function AnimalHusbandry:getNumOfAnimals()
465 local animalsModule = self:getModuleByName("animals")
466
```

```
467  if animalsModule ~= nil then
468  return animalsModule:getNumOfAnimals()
469  end
470  return 0
471  end
```

getAnimalType

Description

Definition

getAnimalType()

Return Values

string return animal type. nil if not available

Code

```
476  function AnimalHusbandry:getAnimalType()
477  local animalsModule = self:getModuleByName("animals")
478
479  if animalsModule ~= nil then
480  return animalsModule:getAnimalType()
481  end
482
483  return nil
484  end
```

getConsumedFood

Description

Returns the amount of food consumed

Definition

getConsumedFood()

Return Values

table of cosumed food (result[fillTypeIndex]=amount)

Code

```
489  function AnimalHusbandry:getConsumedFood()
490  local foodModule = self:getModuleByName("food")
491
492  if foodModule ~= nil then
493  return foodModule:getConsumedFood()
494  end
495  return nil
496  end
```

getFoodProductionFactor

Description

Definition

getFoodProductionFactor()

Code

```lua
function AnimalHusbandry:getFoodProductionFactor()
local foodModule = self:getModuleByName("food")

if foodModule ~= nil then
return foodModule:getFoodFactor()
end
return 0.0
end
```

hasStraw

Description

Definition

hasStraw()

Code

```lua
function AnimalHusbandry:hasStraw()
local strawModule = self:getModuleByName("straw")

if strawModule ~= nil then
return strawModule:hasStraw()
end
return true
end
```

hasWater

Description

Definition

hasWater()

Code

```lua
function AnimalHusbandry:hasWater()
local waterModule = self:getModuleByName("water")

if waterModule ~= nil then
return waterModule:hasWater()
end
return true
end
```

getFoodFilltypeInfo

Description

Definition

getFoodFilltypeInfo()

Code

```lua
function AnimalHusbandry:getFoodFilltypeInfo()
local foodModule = self:getModuleByName("food")

```

```
536  if foodModule ~= nil then
537  return foodModule:getFilltypeInfos()
538  end
539  return AnimalHusbandry.NO_FILLTYPE_INFOS
540  end
```

getWaterFilltypeInfo

Description

Definition

getWaterFilltypeInfo()

Code

```
544  function AnimalHusbandry:getWaterFilltypeInfo()
545  local waterModule = self:getModuleByName("water")
546
547  if waterModule ~= nil then
548  return waterModule:getFilltypeInfos()
549  end
550  return AnimalHusbandry.NO_FILLTYPE_INFOS
551  end
```

getStrawFilltypeInfo

Description

Definition

getStrawFilltypeInfo()

Code

```
555  function AnimalHusbandry:getStrawFilltypeInfo()
556  local strawModule = self:getModuleByName("straw")
557
558  if strawModule ~= nil then
559  return strawModule:getFilltypeInfos()
560  end
561  return AnimalHusbandry.NO_FILLTYPE_INFOS
562  end
```

getPalletsFilltypeInfo

Description

Definition

getPalletsFilltypeInfo()

Code

```
566  function AnimalHusbandry:getPalletsFilltypeInfo()
567  local palletsModule = self:getModuleByName("pallets")
568
569  if palletsModule ~= nil then
570  return palletsModule:getFilltypeInfos()
571  end
```

```
572  return AnimalHusbandry.NO_FILLTYPE_INFOS
573  end
```

getLiquidManureFilltypeInfo

Description

Definition

getLiquidManureFilltypeInfo()

Code

```
577  function AnimalHusbandry:getLiquidManureFilltypeInfo()
578  local liquidManureModule = self:getModuleByName("liquidManure")
579
580  if liquidManureModule ~= nil then
581  return liquidManureModule:getFilltypeInfos()
582  end
583  return AnimalHusbandry.NO_FILLTYPE_INFOS
584  end
```

getManureFilltypeInfo

Description

Definition

getManureFilltypeInfo()

Code

```
588  function AnimalHusbandry:getManureFilltypeInfo()
589  local manureModule = self:getModuleByName("manure")
590
591  if manureModule ~= nil then
592  return manureModule:getFilltypeInfos()
593  end
594  return AnimalHusbandry.NO_FILLTYPE_INFOS
595  end
```

getProductionFilltypeInfo

Description

Get production fill type information for this husbandry.
This will return fill type information in an array per fill type with tables in the form of {fillType=fillType, fillLevel=fillLevel, capacity=capacity}.
The output order is set to manure first, liquid manure (slurry) second and the main husbandry product (e.g. milk, egg pallets) third.

Definition

getProductionFilltypeInfo()

Return Values

table Production	fill type information as {i={j={fillType=fillType, fillLevel=fillLevel, capacity=capacity}}}

Code

```
602  function AnimalHusbandry:getProductionFilltypeInfo()
603  local manureProduction = self:getManureFilltypeInfo()
```

```
604 local liquidManureProduction = self:getLiquidManureFilltypeInfo()
605 local milkProduction = self:getMilkFilltypeInfo()
606 local palletsProduction = self:getPalletsFilltypeInfo()
607
608 local result = {}
609 if manureProduction ~= AnimalHusbandry.NO_FILLTYPE_INFOS then
610 table.insert(result, manureProduction)
611 end
612
613 if liquidManureProduction ~= AnimalHusbandry.NO_FILLTYPE_INFOS
    then
614 table.insert(result, liquidManureProduction)
615 end
616
617 if milkProduction ~= AnimalHusbandry.NO_FILLTYPE_INFOS then
618 table.insert(result, milkProduction)
619 elseif palletsProduction ~= AnimalHusbandry.NO_FILLTYPE_INFOS
    then
620 table.insert(result, palletsProduction)
621 end
622
623 return result
624 end
```

getFoodSpillageFactor

Description

Definition

getFoodSpillageFactor()

Code

```
628 function AnimalHusbandry:getFoodSpillageFactor()
629 local foodSpillageModule = self:getModuleByName("foodSpillage")
630
631 if foodSpillageModule ~= nil then
632 return foodSpillageModule:getSpillageFactor()
633 end
634
635 return nil
636 end
```

getMilkFilltypeInfo

Description

Definition

getMilkFilltypeInfo()

Code

```
640 function AnimalHusbandry:getMilkFilltypeInfo()
641 local milkModule = self:getModuleByName("milk")
642
643 if milkModule ~= nil then
644 return milkModule:getFilltypeInfos()
645 end
646 return AnimalHusbandry.NO_FILLTYPE_INFOS
647 end
```

isSimilarHusbandryRegistered

Description

Returns true if a husbandry with same animal type exists for the current farmId

Definition

isSimilarHusbandryRegistered()

Return Values

Code

```
652 function AnimalHusbandry:isSimilarHusbandryRegistered()
653 local animalsModule = self:getModuleByName("animals")
654
655 if animalsModule ~= nil then
656 local animalType = animalsModule:getAnimalType()
657 if animalType ~= nil then
658 for _, husbandry in pairs(g_currentMission.husbandries) do
659 if animalType == husbandry:getAnimalType() and
    husbandry:getOwnerFarmId() == self:getOwnerFarmId() then
660 return true
661 end
662 end
663 end
664 end
665 return false
666 end
```

registerHusbandryToMission

Description

Registers the husbandry to the mission game.

Definition

registerHusbandryToMission()

Return Values

bool true if registration went well

Code

```
671 function AnimalHusbandry:registerHusbandryToMission()
672 if not self:isSimilarHusbandryRegistered() then
```

```
673 local animalsModule = self:getModuleByName("animals")
674 if animalsModule ~= nil then
675 local husbandryId = animalsModule.husbandryId
676 g_currentMission:registerHusbandry(husbandryId, self)
677 return true
678 end
679 end
680 return false
681 end
```

unregisterHusbandryToMission

Description

Registers the husbandry to the mission game.

Definition

unregisterHusbandryToMission()

Return Values

bool true if registration went well

Code

```
686 function AnimalHusbandry:unregisterHusbandryToMission()
687 local animalsModule = self:getModuleByName("animals")
688
689 if animalsModule ~= nil then
690 local husbandryId = animalsModule.husbandryId
691 if husbandryId ~= nil then
692 g_currentMission:unregisterHusbandry(husbandryId)
693 return true
694 end
695 end
696 return false
697 end
```

getAnimalSubTypeCount

Description

Registers the husbandry to the mission game.

Definition

getAnimalSubTypeCount()

Code

```
701 function AnimalHusbandry:getAnimalSubTypeCount()
702 local count = 0
703 local animalsModule = self:getModuleByName("animals")
704
705 if animalsModule ~= nil then
706 count = animalsModule.numSubTypes
```

```
707 end
708 return count
709 end
```

getAnimals

Description

Registers the husbandry to the mission game.

Definition

getAnimals()

Code

```
713 function AnimalHusbandry:getAnimals()
714 local animalsModule = self:getModuleByName("animals")
715
716 if animalsModule ~= nil then
717 return animalsModule:getAnimals()
718 end
719 end
```

getTypedAnimals

Description

Registers the husbandry to the mission game.

Definition

getTypedAnimals()

Code

```
723 function AnimalHusbandry:getTypedAnimals()
724 local animalsModule = self:getModuleByName("animals")
725
726 if animalsModule ~= nil then
727 return animalsModule:getTypedAnimals()
728 end
729 end
```

addSingleAnimal

Description

Registers the husbandry to the mission game.

Definition

addSingleAnimal()

Code

```
770 function AnimalHusbandry:addSingleAnimal(animal)
771 local animalsModule = self:getModuleByName("animals")
772
773 if animalsModule ~= nil then
774 animalsModule:addSingleAnimal(animal)
775 end
```

```
776 end
```

removeAnimals

Description

Registers the husbandry to the mission game.

Definition

removeAnimals()

Code

```
788 function AnimalHusbandry:removeAnimals(animals)
789 local animalsModule = self:getModuleByName("animals")
790
791 if animalsModule ~= nil then
792 animalsModule:removeAnimals(animals)
793 end
794 end
```

removeSingleAnimal

Description

Registers the husbandry to the mission game.

Definition

removeSingleAnimal()

Code

```
798 function AnimalHusbandry:removeSingleAnimal(animal)
799 local animalsModule = self:getModuleByName("animals")
800
801 if animalsModule ~= nil then
802 animalsModule:removeSingleAnimal(animal)
803 end
804 end
```

addRideable

Description

Definition

addRideable()

Code

```
808 function AnimalHusbandry:addRideable(visualId, player)
809 local animalsModule = self:getModuleByName("animals")
810
811 if animalsModule ~= nil then
812 animalsModule:addRideable(visualId, player)
813 end
814 end
```

removeRideable

Description

Definition

removeRideable()

Code

```
818  function AnimalHusbandry:removeRideable(visualId)
819  local animalsModule = self:getModuleByName("animals")
820
821  if animalsModule ~= nil then
822  animalsModule:removeRideable(visualId)
823  animalsModule:updateVisualDirt()
824  end
825  end
```

isRideableInOnHusbandryGround

Description

Definition

isRideableInOnHusbandryGround()

Code

```
829  function AnimalHusbandry:isRideableInOnHusbandryGround(visualId)
830  local animalsModule = self:getModuleByName("animals")
831
832  if animalsModule ~= nil then
833  return animalsModule:isRideableInOnHusbandryGround(visualId)
834  end
835  return false
836  end
```

getGlobalProductionFactor

Description

Definition

getGlobalProductionFactor()

Code

```
877  function AnimalHusbandry:getGlobalProductionFactor()
878  return self.globalProductionFactor
879  end
```

getFilltype

Description

Definition

getFilltype()

Code

```
883  function AnimalHusbandry:getFilltype(subTypeId)
884  local animalsModule = self:getModuleByName("animals")
885
886  if animalsModule ~= nil then
887  local fillType = animalsModule:getFillType(subTypeId)
888  return fillType
```

```
889 end
890 return nil
891 end
```

getGlobalDirtFactor

Description

Definition

getGlobalDirtFactor()

Return Values

float dirty factor [0-1]

Code

```
896 function AnimalHusbandry:getGlobalDirtFactor()
897 local animalsModule = self:getModuleByName("animals")
898 if animalsModule ~= nil then
899 return animalsModule:getDirtFactor()
900 end
901
902 return 0
903 end
```

renameAnimal

Description

Rename a visible animal identified by subtype and animal ID.

Definition

renameAnimal()

Code

```
907 function AnimalHusbandry:renameAnimal(animalId, name,
    noEventSend)
908 local animalsModule = self:getModuleByName("animals")
909
910 if animalsModule ~= nil then
911 animalsModule:renameAnimal(animalId, name, noEventSend)
912 end
913 end
```

getAnimalName

Description

Retrieves animal name

Definition

getAnimalName()

Code

```
917 function AnimalHusbandry:getAnimalName(visualId)
918 local animalsModule = self:getModuleByName("animals")
919
920 if animalsModule ~= nil then
```

```
921  return animalsModule:getAnimalName(visualId)
922  end
923  return ""
924  end
```

getReproductionTimePerDay

Description

Definition

getReproductionTimePerDay()

Code

```
928  function AnimalHusbandry:getReproductionTimePerDay(fillTypeIndex)
929  local animalsModule = self:getModuleByName("animals")
930  if animalsModule ~= nil then
931  return animalsModule:getReproductionTimePerDay(fillTypeIndex)
932  end
933
934  return 0
935  end
```

getTimeUntilNextAnimal

Description

Definition

getTimeUntilNextAnimal()

Code

```
939  function AnimalHusbandry:getTimeUntilNextAnimal(fillTypeIndex)
940  local animalsModule = self:getModuleByName("animals")
941
942  if animalsModule ~= nil then
943  return animalsModule:getTimeUntilNextAnimal(fillTypeIndex)
944  end
945  return nil
946  end
```

AnimalHusbandryNoMorePalletSpaceEvent

Description

emptyNew

Description

Creating empty instance

Definition

emptyNew()

Return Values

table instance instance of object

Code

```
12  function AnimalHusbandryNoMorePalletSpaceEvent:emptyNew()
13  local self = Event:new(AnimalHusbandryNoMorePalletSpaceEvent_mt)
```

```
14  return self
15  end
```

new

Description

Creating instance

Definition

new(table animalHusbandry)

Arguments

table animalHusbandry instance of animal husbandry

Return Values

table instance instance of object

Code

```
21  function AnimalHusbandryNoMorePalletSpaceEvent:new(animalHusbandry)
22  local self = AnimalHusbandryNoMorePalletSpaceEvent:emptyNew()
23  self.animalHusbandry = animalHusbandry
24  return self
25  end
```

readStream

Description

Reads from network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
32  function AnimalHusbandryNoMorePalletSpaceEvent:readStream(streamId, conne
33  local animalHusbandry = NetworkUtil.readNodeObject(streamId)
34  if animalHusbandry ~= nil then
35  local fillType = g_fillTypeManager:getFillTypeByIndex(animalHusbandry.palletFillType)
36  if fillType ~= nil then
37  local animalTypeString = g_i18n:getText( "ui_statisticView_" .. tostring(animalHusbandry.typeName), g_currentMission.missionInfo.customEnvironment)
38  local text = string.format( g_i18n:getText("ingameNotification_palletSpawnerBlocked"), fillType.title animalTypeString )
39  g_currentMission:addIngameNotification(FSBaseMission.INGAME_NOTIFICATION_ text)
40  end
41  end
```

```
42  end
```

writeStream

Description

Writes in network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
48  function AnimalHusbandryNoMorePalletSpaceEvent:writeStream(streamId, connection)
49  NetworkUtil.writeNodeObject(streamId, self.animalHusbandry)
50  end
```

run

Description

Run event

Definition

run(table connection)

Arguments

table connection connection information

Code

```
55  function AnimalHusbandryNoMorePalletSpaceEvent:run(connection)
56  end
```

AnimalLoadingTrigger

Description

onCreate

Description

Callback of scenegraph object

Definition

onCreate(integer id)

Arguments

integer id nodeid that the trigger is created from

Code

```
20  function AnimalLoadingTrigger:onCreate(id)
21  local trigger = AnimalLoadingTrigger:new(g_server ~= nil, g_client ~= nil)
22  if trigger ~= nil then
23  g_currentMission:addOnCreateLoadedObject(trigger)
24  trigger:load(id)
25  trigger:register(true)
26  end
```

```
27  end
```

new

Description

Creates an instance of the class

Definition

new(bool isServer, bool isClient)

Arguments

bool isServer

bool isClient

Return Values

table self instance

Code

```
34  function AnimalLoadingTrigger:new(isServer, isClient)
35  local self = Object:new(isServer, isClient,
    AnimalLoadingTrigger_mt)
36
37  self.customEnvironment = g_currentMission.loadingMapModName
38  self.isDealer = false
39  self.triggerNode = nil
40  self.appearsOnPDA = true
41  self.title = g_i18n:getText("ui_farm")
42
43  self.animals = nil
44
45  self.activateText = g_i18n:getText("animals_openAnimalScreen",
    self.customEnvironment)
46  self.isActivatableAdded = false
47  self.isPlayerInRange = false
48
49  self.isEnabled = false
50
51  self.loadingVehicle = nil
52  self.activatedTarget = nil
53  self.objectActivated = false
54
55  return self
56  end
```

load

Description

Loads information from scenegraph node.

Definition

load(integer id)

Arguments

integer id nodeid that the trigger is created from

Code

```
function AnimalLoadingTrigger:load(node, husbandry)
self.husbandry = husbandry
self.isDealer = Utils.getNoNil(getUserAttribute(node, "isDealer"), false)

if self.isDealer then
local animalTypesString = getUserAttribute(node, "animalTypes")
if animalTypesString ~= nil then
local animalTypes = StringUtil.splitString(" ", animalTypesString)
for _, animalTypeStr in pairs(animalTypes) do
local animalType = g_animalManager:getAnimalsByType(animalTypeStr)
if animalType ~= nil then
if self.animalTypes == nil then
self.animalTypes = {}
end

table.insert(self.animalTypes, animalType)
else
g_logManager:warning("Invalid animal type '%s' for animalLoadingTrigger '%s'!", animalTypeStr, getName(node))
end
end
end
end

self.triggerNode = node
addTrigger(self.triggerNode, "triggerCallback", self)

self.appearsOnPDA = Utils.getNoNil(getUserAttribute(node, "appearsOnPDA"), self.isDealer)
self.title = g_i18n:getText(Utils.getNoNil(getUserAttribute(node, "title"), "ui_farm"), self.customEnvironment)
self.isEnabled = not g_isPresentationVersion or g_isPresentationVersionShopEnabled

if self.appearsOnPDA then
local mapPosition = node
```

```
93  local mapPositionIndex = getUserAttribute(node,
    "mapPositionIndex")
94
95  if mapPositionIndex ~= nil then
96  mapPosition = I3DUtil.indexToObject(node, mapPositionIndex)
97  if mapPosition == nil then
98  mapPosition = node
99  end
100 end
101
102 local x, _, z = getWorldTranslation(mapPosition)
103
104 local fullViewName = Utils.getNoNil(getUserAttribute(node,
    "stationName"), "animals_dealer")
105 if g_i18n:hasText(fullViewName) then
106 fullViewName = g_i18n:getText(fullViewName,
    self.customEnvironment)
107 end
108
109 self.mapHotspot = MapHotspot:new("livestockDealer",
    MapHotspot.CATEGORY_DEFAULT)
110 self.mapHotspot:setText(fullViewName)
111 self.mapHotspot:setBorderedImage(nil,
    getNormalizedUVs(MapHotspot.UV.ANIMAL_VENDOR))
112 self.mapHotspot:setWorldPosition(x, z)
113
114 g_currentMission:addMapHotspot(self.mapHotspot)
115 end
116
117 return self
118 end
```

delete

Description

Deletes instance

Definition

delete()

Code

```
122 function AnimalLoadingTrigger:delete()
123 if self.mapHotspot ~= nil then
124 g_currentMission:removeMapHotspot(self.mapHotspot)
125 self.mapHotspot:delete()
126 end
```

```
127
128 g_currentMission:removeActivatableObject(self)
129
130 if self.triggerNode ~= nil then
131 removeTrigger(self.triggerNode)
132 self.triggerNode = nil
133 end
134 end
```

triggerCallback

Description

Callback when trigger changes state

Definition

triggerCallback(integer triggerId, integer otherId, bool onEnter, bool onLeave, bool onStay)

Arguments

integer	triggerId
integer	otherId
bool	onEnter
bool	onLeave
bool	onStay

Code

```
143 function AnimalLoadingTrigger:triggerCallback(triggerId, otherId,
    onEnter, onLeave, onStay)
144 if self.isEnabled and (onEnter or onLeave) then
145 local vehicle = g_currentMission.nodeToObject[otherId]
146 if vehicle ~= nil and vehicle.getSupportsAnimalType ~= nil then
147 if onEnter then
148 self:setLoadingTrailer(vehicle)
149 elseif onLeave then
150 if vehicle == self.loadingVehicle then
151 self:setLoadingTrailer(nil)
152 end
153 if vehicle == self.activatedTarget then
154 -- close dialog!
155 g_animalScreen:onVehicleLeftTrigger()
156 self.objectActivated = false
157 end
158 end
159 elseif g_currentMission.player ~= nil and otherId ==
    g_currentMission.player.rootNode then
160 if onEnter then
161 self.isPlayerInRange = true
162 else
```

```
163     self.isPlayerInRange = false
164 end
165 self:updateActivatableObject()
166 end
167 end
168 end
```

updateActivatableObject

Description

Adds or removes the trigger as an activable object to the mission

Definition

updateActivatableObject()

Code

```
172 function AnimalLoadingTrigger:updateActivatableObject()
173 if self.loadingVehicle ~= nil or self.isPlayerInRange then
174 if not self.isActivatableAdded then
175 self.isActivatableAdded = true
176 g_currentMission:addActivatableObject(self)
177 end
178 else
179 if self.isActivatableAdded and self.loadingVehicle == nil and not
    self.isPlayerInRange then
180 g_currentMission:removeActivatableObject(self)
181 self.isActivatableAdded = false
182 self.objectActivated = false
183 end
184 end
185 end
```

setLoadingTrailer

Description

Sets the loading trailer

Definition

setLoadingTrailer(table loadingVehicle)

Arguments

table loadingVehicle

Code

```
190 function AnimalLoadingTrigger:setLoadingTrailer(loadingVehicle)
191 if self.loadingVehicle ~= nil and
    self.loadingVehicle.setLoadingTrigger ~= nil then
192 self.loadingVehicle:setLoadingTrigger(nil)
193 end
194 self.loadingVehicle = loadingVehicle
```

```
195 if self.loadingVehicle ~= nil and
    self.loadingVehicle.setLoadingTrigger ~= nil then
196 self.loadingVehicle:setLoadingTrigger(self)
197 end
198
199 self:updateActivatableObject()
200 end
```

getIsActivatable

Description

Checks if can be activated

Definition

getIsActivatable(table vehicle)

Arguments

table vehicle unused variable

Code

```
205 function AnimalLoadingTrigger:getIsActivatable(vehicle)
206 local canAccess = self.husbandry == nil or
    self.husbandry:getOwnerFarmId() == g_currentMission:getFarmId()
207
208 if g_gui.currentGui == nil and self.isEnabled and canAccess and
    g_currentMission:getHasPlayerPermission("tradeAnimals") then
209 local rootAttacherVehicle = nil
210 if self.loadingVehicle ~= nil then
211 rootAttacherVehicle = self.loadingVehicle:getRootVehicle()
212 end
213
214 return self.isPlayerInRange or rootAttacherVehicle ==
    g_currentMission.controlledVehicle
215 end
216 return false
217 end
```

drawActivate

Description

Draw

Definition

drawActivate()

Code

```
221 function AnimalLoadingTrigger:drawActivate()
222 end
```

onActivateObject

Description

Callback on activate

Definition

onActivateObject()

Code

```
226  function AnimalLoadingTrigger:onActivateObject()
227  g_currentMission:removeActivatableObject(self)
228  self.isActivatableAdded = false
229
230  self.objectActivated = true
231  self.activatedTarget = self.loadingVehicle
232
233  local controller = g_animalScreen:getController()
234  controller:setTrailer(self.loadingVehicle)
235  controller:setHusbandry(self.husbandry)
236  controller:setLoadingTrigger(self)
237  controller:initialize()
238  g_gui:showGui("AnimalScreen")
239  end
```

AnimalManager

Description

new

Description

Creating manager

Definition

new(table meta)

Arguments

table meta table

Return Values

table instance instance of object

Code

```
21  function AnimalManager:new(customMt)
22  local self = AbstractManager:new(customMt or AnimalManager_mt)
23
24  return self
25  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
29  function AnimalManager:initDataStructures()
30  self.numAnimals = 0
```

```
31  self.animals = {}
32  self.typeToAnimal = {}
33  self.indexToAnimal = {}
34  self.fillTypeToAnimal = {}
35  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData(table xmlFile, table missionInfo)

Arguments

table xmlFile XML file handle

table missionInfo mission information

Return Values

boolean true if loading was successful else false

Code

```
42  function AnimalManager:loadMapData(xmlFile, missionInfo)
43  AnimalManager:superClass().loadMapData(self)
44
45  local filename = Utils.getFilename(getXMLString(xmlFile,
    "map.husbandryAnimals#filename"), g_currentMission.baseDirectory)
46  if filename == nil or filename == "" then
47  g_logManager:error("Could not load husbandry config file '%s'!",
    tostring(filename))
48  return false
49  end
50  local animalXmlFile = loadXMLFile("husbandry", filename)
51
52  if animalXmlFile ~= nil then
53  self:loadAnimals(animalXmlFile, g_currentMission.baseDirectory)
54  delete(animalXmlFile)
55
56  return self.numAnimals ~= 0
57  end
58  return false
59  end
```

loadAnimals

Description

Load data on map load

Definition

loadAnimals(table xmlFile)

Arguments

table xmlFile XML file handle

Return Values

boolean true if loading was successful else false

Code

```
65  function AnimalManager:loadAnimals(xmlHandle, baseDirectory)
66  if xmlHandle == 0 then
67  return false
68  end
69  self.animals = {}
70  self.typeToAnimal = {}
71  local i = 0
72  while true do
73  local animalKey = string.format("animals.animal(%d)", i)
74  if not hasXMLProperty(xmlHandle, animalKey) then
75  break
76  end
77  local animal = {}
78  animal.type = getXMLString(xmlHandle, animalKey.. "#type")
79  animal.class = getXMLString(xmlHandle, animalKey.. "#class")
80
81  if not ClassUtil.getIsValidClassName(animal.class) then
82  g_logManager:error("Invalid animal class name '%s'!",
    tostring(animal.class))
83  return false
84  end
85  if ClassUtil.getClassObject(animal.class) == nil then
86  g_logManager:error("Animal class '%s' not defined !",
    tostring(animal.class))
87  return false
88  end
89
90  local hasName = Utils.getNoNil(getXMLBool(xmlHandle,
    animalKey.."#hasName"), false)
91  animal.stats = {}
92  animal.stats.breeding = Utils.getNoNil(getXMLString(xmlHandle,
    animalKey.. "#statsBreeding"), "")
93  animal.subTypes = {}
94  if animal.type ~= nil then
95  animal.type = animal.type:upper()
96  -- Specific animal subtype info
97  local j = 0
98  while true do
```

```
99  local subTypeKey = string.format("%s.subType(%d)", animalKey, j)
100 if not hasXMLProperty(xmlHandle, subTypeKey) then
101 break
102 end
103 local animalSubType = {}
104 local fillTypeName = getXMLString(xmlHandle,
    subTypeKey.."#fillTypeName")
105 local fillType =
    g_fillTypeManager:getFillTypeByName(fillTypeName)
106 if fillType ~= nil then
107 animalSubType.storeInfo = {}
108 animalSubType.output = {}
109 animalSubType.input = {}
110 animalSubType.breeding = {}
111 animalSubType.texture = {}
112 animalSubType.dirt = {}
113 animalSubType.dummy = {}
114 -- General info
115 self.numAnimals = self.numAnimals + 1
116 animalSubType.index = self.numAnimals
117 animalSubType.type = animal.type
118 animalSubType.hasName = hasName
119 animalSubType.subTypeId = j + 1 -- in lua context we use a 1-
    based array. Only when we call engine do we translate to 0-based
    array
120 animalSubType.fillType = fillType.index
121 animalSubType.fillTypeDesc = fillType
122 animalSubType.subTypeName =
    g_i18n:convertText(Utils.getNoNil(getXMLString(xmlHandle,
    subTypeKey.."#name"), ""))
123 -- Store
124 animalSubType.storeInfo.shopItemName =
    g_i18n:convertText(Utils.getNoNil(getXMLString(xmlHandle,
    subTypeKey..".store#itemName"), ""))
125 animalSubType.storeInfo.canBeBought =
    Utils.getNoNil(getXMLBool(xmlHandle,
    subTypeKey..".store#canBeBought"), false)
126 animalSubType.storeInfo.imageFilename =
    Utils.getNoNil(getXMLString(xmlHandle,
    subTypeKey..".store#image"), "")
127 animalSubType.storeInfo.imageFilename =
    Utils.getFilename(animalSubType.storeInfo.imageFilename,
    baseDirectory)
```

```
128 animalSubType.storeInfo.buyPrice =
    Utils.getNoNil(getXMLFloat(xmlHandle,
    subTypeKey..".store#buyPrice"), 0.0)
129 animalSubType.storeInfo.sellPrice =
    Utils.getNoNil(getXMLFloat(xmlHandle,
    subTypeKey..".store#sellPrice"), 0.0)
130 -- Output
131 animalSubType.output.milkPerDay =
    Utils.getNoNil(getXMLFloat(xmlHandle,
    subTypeKey..".output#milkPerDay"), 0)
132 animalSubType.output.manurePerDay =
    Utils.getNoNil(getXMLFloat(xmlHandle,
    subTypeKey..".output#manurePerDay"), 0)
133 animalSubType.output.liquidManurePerDay =
    Utils.getNoNil(getXMLFloat(xmlHandle,
    subTypeKey..".output#liquidManurePerDay"), 0)
134 animalSubType.output.palletsPerDay =
    Utils.getNoNil(getXMLFloat(xmlHandle,
    subTypeKey..".output#palletsPerDay"), 0)
135 animalSubType.output.foodSpillagePerDay =
    Utils.getNoNil(getXMLFloat(xmlHandle,
    subTypeKey..".output#foodSpillagePerDay"), 0)
136 -- Input
137 animalSubType.input.strawPerDay =
    Utils.getNoNil(getXMLFloat(xmlHandle,
    subTypeKey..".input#strawPerDay"), 0)
138 animalSubType.input.waterPerDay =
    Utils.getNoNil(getXMLFloat(xmlHandle,
    subTypeKey..".input#waterPerDay"), 0)
139 animalSubType.input.foodPerDay =
    Utils.getNoNil(getXMLFloat(xmlHandle,
    subTypeKey..".input#foodPerDay"), 0)
140 -- texture
141 animalSubType.texture.tileUIndex = getXMLInt(xmlHandle,
    subTypeKey..".texture#tileUIndex")
142 animalSubType.texture.tileVIndex = getXMLInt(xmlHandle,
    subTypeKey..".texture#tileVIndex")
143 -- dirt
144 animalSubType.dirt.cleanDuration = (getXMLFloat(xmlHandle,
    subTypeKey..".dirt#cleanDuration") or 0.5) * 1000
145 -- dummy
146 animalSubType.dummy.filename =
    Utils.getFilename(Utils.getNoNil(getXMLString(xmlHandle,
    subTypeKey..".dummy#filename"), baseDirectory))
147 animalSubType.dummy.meshNodeStr = getXMLString(xmlHandle,
    subTypeKey..".dummy#meshNode") or "0"
```

```
148 animalSubType.dummy.hairNodeStr = getXMLString(xmlHandle,
    subTypeKey..".dummy#hairNode")
149
150 -- Breeding
151 animalSubType.breeding.birthRatePerDay =
    Utils.getNoNil(getXMLFloat(xmlHandle,
    subTypeKey..".breeding#birthRatePerDay"), 0)
152 -- Riding
153 animalSubType.rideableFileName =
    Utils.getFilename(Utils.getNoNil(getXMLString(xmlHandle,
    subTypeKey..".rideable#filename"), ""))
154 -- Search tables
155 self.indexToAnimal[animalSubType.index] = animalSubType
156 self.fillTypeToAnimal[animalSubType.fillType] = animalSubType
157
158 table.insert(animal.subTypes, animalSubType)
159 else
160 g_logManager:warning("FillType '%s' for animal '%s' not defined.
    Ignoring animal!", tostring(fillTypeName), animal.subTypeName)
161 end
162 j = j + 1
163 end
164
165 if #animal.subTypes > 0 then
166 self.typeToAnimal[animal.type] = animal
167 table.insert(self.animals, animal)
168 else
169 g_logManager:error("No sub types defined for animal '%s'.
    Ignoring animal!", animal.type)
170 end
171 else
172 g_logManager:warning("Animal '%d' has no type. Ignoring animal!",
    i)
173 end
174 i = i + 1
175 end
176 end
```

getAnimals

Description

Gets all animals

Definition

getAnimals()

Return Values

table animals list all animals

Code

```
181  function AnimalManager:getAnimals()
182  return self.animals
183  end
```

getAnimalByIndex

Description

Gets an animal by index

Definition

getAnimalByIndex(integer index)

Arguments

integer index the animal index

Return Values

table animal the animal object

Code

```
189  function AnimalManager:getAnimalByIndex(index)
190  if index ~= nil then
191  return self.indexToAnimal[index]
192  end
193  return nil
194  end
```

getAnimalsByType

Description

Gets an animal by index name

Definition

getAnimalsByType(string name)

Arguments

string name the animal index name

Return Values

table animal the animal object

Code

```
200  function AnimalManager:getAnimalsByType(animalType)
201  if ClassUtil.getIsValidIndexName(animalType) then
202  animalType = animalType:upper()
203  return self.typeToAnimal[animalType]
204  end
205  return nil
206  end
```

getAnimalByFillType

Description

Gets an animal by fillType

Definition

getAnimalByFillType(integer fillType)

Arguments

integer fillType the fillType index

Return Values

table animal the animal object

Code

```
212  function AnimalManager:getAnimalByFillType(fillType)
213  if fillType ~= nil then
214  return self.fillTypeToAnimal[fillType]
215  end
216  return nil
217  end
```

getAnimalType

Description

Gets an animal type by animal index

Definition

getAnimalType(integer animalIndex)

Arguments

integer animalIndex

Return Values

string animal type

Code

```
223  function AnimalManager:getAnimalType(animalIndex)
224  for _, animal in pairs(self.typeToAnimal) do
225  for _, animalSubtype in pairs(animal.subTypes) do
226  if animalSubtype.index == animalIndex then
227  return animal.type
228  end
229  end
230  end
231  return nil
232  end
```

getFillType

Description

Gets an animal fillType

Definition

getFillType(integer animalIndex, integer animal)

Arguments

integer animalIndex

integer animal subtype Index

Return Values

table

Code

```
239 function AnimalManager:getFillType(animalIndex, subtypeIndex)
240 local animal = self.indexToAnimal[animalIndex]
241
242 if animal ~= nil then
243 local animalSubType = animal.subTypes[subtypeIndex]
244
245 if animalSubType ~= nil then
246 local fillType =
    g_fillTypeManager:getFillTypeByIndex(animalSubType.fillType)
247 if fillType ~= nil then
248 return fillType
249 end
250 end
251 end
252 g_logManager:devWarning("Warning: could not find fillType for
    animal id(%d) subTypeId(%d)!", animalIndex, subtypeIndex)
253 return nil
254 end
```

getStoreInfo

Description

Gets an animal store information

Definition

getStoreInfo(integer animalIndex, integer animal)

Arguments

integer animalIndex

integer animal subtype Index

Return Values

table store information [shopItemName, canBeBought, imageFilename, price]

Code

```
261 function AnimalManager:getStoreInfo(animalIndex, subtypeIndex)
262 local animal = self.indexToAnimal[animalIndex]
263
264 if animal ~= nil then
265 local animalSubType = animal.subTypes[subtypeIndex]
266 return animalSubType.storeInfo
267 end
268 return nil
269 end
```

AnimalNameManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
19  function AnimalNameManager:new(customMt)
20  local self = AbstractManager:new(customMt or AnimalNameManager_mt)
21
22  return self
23  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
27  function AnimalNameManager:initDataStructures()
28  self.names = {}
29  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
34  function AnimalNameManager:loadMapData(xmlFile, missionInfo)
35  AnimalNameManager:superClass().loadMapData(self)
36
37  local filename = Utils.getFilename(getXMLString(xmlFile,
    "map.animalNames#filename"), g_currentMission.baseDirectory)
38  if filename == nil or filename == "" then
39  print("Error: Could not load animal name configuration file '" ..
    tostring(filename) .. "'!")
40  return false
41  end
42
43  self:loadNamesFromXML(filename)
44
45  return true
```

```
46  end
```

getRandomName

Description

Definition

getRandomName()

Return Values

table returns a food group or nil if nothing is found

Code

```
77  function AnimalNameManager:getRandomName()
78  local index = math.random(#self.names)
79
80  if index > 0 then
81  return self.names[index]
82  end
83  return ""
84  end
```

CrowsWildlife

Description

new

Description

Creating instance

Definition

new(table mt)

Arguments

table mt custom meta table

Return Values

table instance Instance of object

Code

```
37  function CrowsWildlife:new(mt)
38  if mt == nil then
39  mt = CrowsWildlife_mt
40  end
41  local self = LightWildlife:new(mt)
42
43  self.animalStates = {}
44  for _, stateEntry in pairs(CrowsWildlife.CROW_STATES) do
45  table.insert(self.animalStates, stateEntry)
46  end
47  self.tree = nil
48  self.soundFSM = FSMUtil.create()
49  self.soundFSM:addState(CrowsWildlife.CROW_SOUND_STATES.NONE,
    CrowSoundStateDefault:new(CrowsWildlife.CROW_SOUND_STATES.NONE, self,
    self.soundFSM))
```

```
50  self.soundFSM:addState(CrowsWildlife.CROW_SOUND_STATES.CALM_GROUND,
    CrowSoundStateCalmGround:new(CrowsWildlife.CROW_SOUND_STATES.CALM_GROUND,
    self, self.soundFSM))
51  self.soundFSM:addState(CrowsWildlife.CROW_SOUND_STATES.CALM_AIR,
    CrowSoundStateCalmAir:new(CrowsWildlife.CROW_SOUND_STATES.CALM_AIR, self,
    self.soundFSM))
52  self.soundFSM:addState(CrowsWildlife.CROW_SOUND_STATES.BUSY,
    CrowSoundStateBusy:new(CrowsWildlife.CROW_SOUND_STATES.BUSY, self,
    self.soundFSM))
53  self.soundFSM:addState(CrowsWildlife.CROW_SOUND_STATES.TAKEOFF,
    CrowSoundStateTakeOff:new(CrowsWildlife.CROW_SOUND_STATES.TAKEOFF, self,
    self.soundFSM))
54  self.soundFSM:changeState(CrowsWildlife.CROW_SOUND_STATES.NONE)
55  return self
56  end
```

delete

Description

Delete instance

Definition

delete()

Code

```
60  function CrowsWildlife:delete()
61  for _,sample in pairs(self.samplesFlyAway) do
62  g_soundManager:deleteSample(sample)
63  end
64  g_soundManager:deleteSample(self.sampleBusy)
65  g_soundManager:deleteSample(self.sampleCalm)
66  end
```

load

Description

Load xml file

Definition

load(string xmlFilename)

Arguments

string xmlFilename xml filename to load

Return Values

bool true if load is successful

Code

```
72  function CrowsWildlife:load(xmlFilename)
73  CrowsWildlife:superClass().load(self, xmlFilename)
74
75  local xmlFile = loadXMLFile("TempXML", self.xmlFilename)
76  if xmlFile == 0 then
```

```
77  self.xmlFilename = nil
78  return false
79  end
80
81  self.samples = {}
82  self.samples.flyAway = {}
83  local i = 0
84  while true do
85  local sampleFlyAway = g_soundManager:loadSampleFromXML(xmlFile,
    "wildlifeAnimal.sounds.flyAways", string.format("flyAway(%d)",
    i), self.baseDirectory, self.soundsNode, 1,
    AudioGroup.ENVIRONMENT, nil, nil)
86  if sampleFlyAway == nil then
87  break
88  end
89  table.insert(self.samples.flyAway, sampleFlyAway)
90  i = i + 1
91  end
92  self.samples.flyAwayCount = i
93  self.samples.calmGround = {}
94  local j = 0
95  while true do
96  local sampleCalmGround =
    g_soundManager:loadSampleFromXML(xmlFile,
    "wildlifeAnimal.sounds.calmGrounds",
    string.format("calmGround(%d)", j), self.baseDirectory,
    self.soundsNode, 1, AudioGroup.ENVIRONMENT, nil, nil)
97  if sampleCalmGround == nil then
98  break
99  end
100 table.insert(self.samples.calmGround, sampleCalmGround)
101 j = j + 1
102 end
103 self.samples.calmCount = j
104 self.samples.busy = g_soundManager:loadSampleFromXML(xmlFile,
    "wildlifeAnimal.sounds", "busy", self.baseDirectory,
    self.soundsNode, 0, AudioGroup.ENVIRONMENT, nil, nil)
105 self.samples.calmAir = g_soundManager:loadSampleFromXML(xmlFile,
    "wildlifeAnimal.sounds", "calmAir", self.baseDirectory,
    self.soundsNode, 0, AudioGroup.ENVIRONMENT, nil, nil)
106 delete(xmlFile)
107 return true
108 end
```

createAnimals

Description

Create animals

Definition

createAnimals(string name, float spawnPosX, float spawnPosY, float spawnPosZ, integer nbAnimals)

Arguments

string	name	name of animals to spawn
float	spawnPosX	world x position
float	spawnPosY	world y position
float	spawnPosZ	world z position
integer	nbAnimals	amount of animals to spawn

Return Values

integer id of the animal group

Code

```
118  function CrowsWildlife:createAnimals(name, spawnPosX, spawnPosY, spawnPosZ, nbAnimals)
119  if #self.animals == 0 then
120  self.soundFSM:changeState(CrowsWildlife.CROW_SOUND_STATES.CALM_GROUND)
121  end
122  local id = CrowsWildlife:superClass().createAnimals(self, name, spawnPosX, spawnPosY, spawnPosZ, nbAnimals)
123  return id
124  end
```

update

Description

update

Definition

update()

Code

```
128  function CrowsWildlife:update(dt)
129  CrowsWildlife:superClass().update(self, dt)
130
131  if #self.animals > 0 then
132  self.soundFSM:update(dt)
133  elseif self.soundFSM.currentState.id ~= CrowsWildlife.CROW_SOUND_STATES.NONE then
134  self.soundFSM:changeState(CrowsWildlife.CROW_SOUND_STATES.NONE)
135  end
136  end
```

searchTree

Description

Search tree around a radius

Definition

searchTree(float x, float y, float z, radius radius)

Arguments

float	x	x world position from which areas are checked
float	y	y world position from which areas are checked
float	z	z world position from which areas are checked
radius	radius	radius of the test in m

Return Values

integer number of trees found

Code

```
145  function CrowsWildlife:searchTree(x, y, z, radius)
146  overlapSphere(x, y, z, radius, "treeSearchCallback", self, 2,
     false, true, false)
147  end
```

treeSearchCallback

Description

Tree count callback

Definition

treeSearchCallback(integer transformId)

Arguments

integer transformId - transformId of the element detected in the overlap test

Return Values

bool true to continue counting trees

Code

```
153  function CrowsWildlife:treeSearchCallback(transformId)
154  self.tree = nil
155  if transformId ~= 0 and getHasClassId(transformId,
     ClassIds.SHAPE) then
156  local object = getParent(transformId)
157  if object ~= nil and getSplitType(transformId) ~= 0 then
158  self.tree = object
159  end
160  end
161  return true
162  end
```

getAverageLocationOfIdleAnimals

Description

Get average location of all idle animals

Definition

getAverageLocationOfIdleAnimals()

Code

```
166  function CrowsWildlife:getAverageLocationOfIdleAnimals()
167  local nbIdleAnimals = 0
```

```
168 local accPosX, accPosZ = 0.0, 0.0
169 for _, animal in pairs(self.animals) do
170 local currentState = animal.stateMachine.currentState.id
171 if currentState == "idle_walk" or currentState == "idle_eat" or
    currentState == "idle_attention" then
172 local posX, posY, posZ = getWorldTranslation(animal.i3dNodeId)
173 accPosX, accPosZ = accPosX + posX, accPosZ + posZ
174 nbIdleAnimals = nbIdleAnimals + 1
175 end
176 end
177 if nbIdleAnimals > 0 then
178 accPosX, accPosZ = accPosX / nbIdleAnimals, accPosZ /
    nbIdleAnimals
179 local terrainHeight = self:getTerrainHeightWithProps(accPosX,
    accPosZ)
180 return true, accPosX, terrainHeight, accPosZ
181 end
182 return false, 0.0, 0.0, 0.0
183 end
```

CrowsWildlifeSoundStates

Description

SoundStateDefault:new

Description

Creating instance of state.

Definition

SoundStateDefault:new()

Code

```
17 function CrowSoundStateDefault:new(id, owner, stateMachine,
   custom_mt)
18 local self = SimpleState:new(id, owner, stateMachine, custom_mt or
   CrowSoundStateDefault_mt)
19 return self
20 end
```

SoundStateTakeOff:new

Description

Creating instance of state.

Definition

SoundStateTakeOff:new()

Code

```
31 function CrowSoundStateTakeOff:new(id, owner, stateMachine,
   custom_mt)
32 local self = SimpleState:new(id, owner, stateMachine, custom_mt or
   CrowSoundStateTakeOff_mt)
```

```
33  self.currentIdx = 0
34  return self
35  end
```

SoundStateTakeOff:activate

Description

Activate method; place sounds node in the middle of idle crows; play a random sample

Definition

SoundStateTakeOff:activate()

Code

```
39  function CrowSoundStateTakeOff:activate(parms)
40  CrowSoundStateTakeOff:superClass().activate(self, parms)
41
42  if self.owner.samples.flyAwayCount > 0 then
43  self.currentIdx = math.floor(math.random(1,
    self.owner.samples.flyAwayCount))
44  g_soundManager:playSample(self.owner.samples.flyAway[self.currentIdx])
45  else
46  self.stateMachine:changeState(CrowsWildlife.CROW_SOUND_STATES.CALM_GROUND
47  end
48  end
```

SoundStateTakeOff:deactivate

Description

Deactivate method

Definition

SoundStateTakeOff:deactivate()

Code

```
52  function CrowSoundStateTakeOff:deactivate()
53  CrowSoundStateTakeOff:superClass().deactivate(self)
54
55  if self.currentIdx > 0 then
56  local sample = self.owner.samples.flyAway[self.currentIdx]
57  g_soundManager:stopSample(sample)
58  self.currentIdx = 0
59  end
60  end
```

SoundStateTakeOff:update

Description

Update method

Definition

SoundStateTakeOff:update()

Code

```
64  function CrowSoundStateTakeOff:update(dt)
65  CrowSoundStateTakeOff:superClass().update(self, dt)
66
67  if self.currentIdx > 0 then
68  local sample = self.owner.samples.flyAway[self.currentIdx]
69  if not g_soundManager:getIsSamplePlaying(sample) then
70  self.currentIdx = 0
71  self.stateMachine:changeState(CrowsWildlife.CROW_SOUND_STATES.BUSY)
72  end
73  end
74  end
```

SoundStateBusy:new

Description

Creating instance of state.

Definition

SoundStateBusy:new()

Code

```
85  function CrowSoundStateBusy:new(id, owner, stateMachine,
    custom_mt)
86  local self = SimpleState:new(id, owner, stateMachine, custom_mt or
    CrowSoundStateBusy_mt)
87  self.minTimeUntilNextSamplePlay = 6.0 * 1000.0
88  self.maxTimeUntilNextSamplePlay = 8.0 * 1000.0
89  self.timer = 0.0
90  return self
91  end
```

SoundStateBusy:activate

Description

Activate method; place sounds node in the middle of idle crows; play a random sample

Definition

SoundStateBusy:activate()

Code

```
95  function CrowSoundStateBusy:activate(parms)
96  CrowSoundStateBusy:superClass().activate(self, parms)
97
98  g_soundManager:playSample(self.owner.samples.busy)
99  self.timer = self.minTimeUntilNextSamplePlay + (math.random() *
    (self.maxTimeUntilNextSamplePlay -
    self.minTimeUntilNextSamplePlay))
100 end
```

SoundStateBusy:deactivate

Description

Deactivate method

Definition

SoundStateBusy:deactivate()

Code

```
104  function CrowSoundStateBusy:deactivate()
105  CrowSoundStateBusy:superClass().deactivate(self)
106
107  g_soundManager:stopSample(self.owner.samples.busy)
108  end
```

SoundStateBusy:update

Description

Update method

Definition

SoundStateBusy:update()

Code

```
112  function CrowSoundStateBusy:update(dt)
113  CrowSoundStateBusy:superClass().update(self, dt)
114
115  self.timer = math.max(self.timer - dt, 0.0)
116  if (self.timer == 0.0) then
117  self.stateMachine:changeState(CrowsWildlife.CROW_SOUND_STATES.CALM_AIR)
118  end
119  end
```

SoundStateCalmGround:new

Description

Creating instance of state.

Definition

SoundStateCalmGround:new()

Code

```
130  function CrowSoundStateCalmGround:new(id, owner, stateMachine,
     custom_mt)
131  local self = SimpleState:new(id, owner, stateMachine, custom_mt
     or CrowSoundStateCalmGround_mt)
132  self.minTimeUntilNextSamplePlay = 6.0 * 1000.0
133  self.maxTimeUntilNextSamplePlay = 8.0 * 1000.0
134  self.timer = 0.0
135  self.currentIdx = 0
136  return self
137  end
```

SoundStateCalmGround:activate

Description

Activate method; place sounds node in the middle of idle crows; play a random sample

Definition

SoundStateCalmGround:activate()

Code

```
141  function CrowSoundStateCalmGround:activate(parms)
142  CrowSoundStateCalmGround:superClass().activate(self, parms)
143
144  if self.owner.samples.calmCount > 0 then
145  local birdsOnGround, soundPosX, soundPosY, soundPosZ =
     self.owner:getAverageLocationOfIdleAnimals()
146
147  self.timer = self.minTimeUntilNextSamplePlay + (math.random() *
     (self.maxTimeUntilNextSamplePlay - self.minTimeUntilNextSamplePlay))
148  if birdsOnGround then
149  setWorldTranslation(self.owner.soundsNode, soundPosX, soundPosY,
     soundPosZ)
150  self.currentIdx = math.floor(math.random(1,
     self.owner.samples.calmCount))
151  g_soundManager:playSample(self.owner.samples.calmGround[self.currentIdx]
152  end
153  end
154  end
```

SoundStateCalmGround:deactivate

Description

Deactivate method

Definition

SoundStateCalmGround:deactivate()

Code

```
158  function CrowSoundStateCalmGround:deactivate()
159  CrowSoundStateCalmGround:superClass().deactivate(self)
160
161  if self.currentIdx > 0 then
162  local sample = self.owner.samples.calmGround[self.currentIdx]
163  g_soundManager:stopSample(sample)
164  self.currentIdx = 0
165  end
166  end
```

SoundStateCalmGround:update

Description

Update method

Definition

SoundStateCalmGround:update()

Code

```
170 function CrowSoundStateCalmGround:update(dt)
171 CrowSoundStateCalmGround:superClass().update(self, dt)
172
173 if self.currentIdx > 0 then
174 local sample = self.owner.samples.calmGround[self.currentIdx]
175 if not g_soundManager:getIsSamplePlaying(sample) then
176 self.currentIdx = 0
177 end
178 end
179
180 self.timer = math.max(self.timer - dt, 0.0)
181 if (self.timer == 0.0) and (self.owner.samples.calmCount > 0) then
182 self.timer = self.minTimeUntilNextSamplePlay + (math.random() *
(self.maxTimeUntilNextSamplePlay - self.minTimeUntilNextSamplePlay))
183
184 local birdsOnGround, soundPosX, soundPosY, soundPosZ =
self.owner:getAverageLocationOfIdleAnimals()
185 if birdsOnGround then
186 setWorldTranslation(self.owner.soundsNode, soundPosX, soundPosY,
soundPosZ)
187 self.currentIdx = math.floor(math.random(1,
self.owner.samples.calmCount))
188 g_soundManager:playSample(self.owner.samples.calmGround[self.currentIdx]
189 end
190 end
191 end
```

SoundStateCalmAir:new

Description

Creating instance of state.

Definition

SoundStateCalmAir:new()

Code

```
202 function CrowSoundStateCalmAir:new(id, owner, stateMachine,
custom_mt)
203 local self = SimpleState:new(id, owner, stateMachine, custom_mt
or CrowSoundStateCalmAir_mt)
204 self.minTimeUntilNextSamplePlay = 6.0 * 1000.0
205 self.maxTimeUntilNextSamplePlay = 8.0 * 1000.0
206 self.timer = 0.0
207 return self
208 end
```

SoundStateCalmAir:activate

Description

Activate method; place sounds node in the middle of idle crows; play a random sample

Definition

SoundStateCalmAir:activate()

Code

```
212 function CrowSoundStateCalmAir:activate(parms)
213 CrowSoundStateCalmAir:superClass().activate(self, parms)
214
215 g_soundManager:playSample(self.owner.samples.calmAir)
216 self.timer = self.minTimeUntilNextSamplePlay + (math.random() *
    (self.maxTimeUntilNextSamplePlay -
    self.minTimeUntilNextSamplePlay))
217 end
```

SoundStateCalmAir:deactivate

Description

Deactivate method

Definition

SoundStateCalmAir:deactivate()

Code

```
221 function CrowSoundStateCalmAir:deactivate()
222 CrowSoundStateCalmAir:superClass().deactivate(self)
223
224 g_soundManager:stopSample(self.owner.samples.calmAir)
225 end
```

SoundStateCalmAir:update

Description

Update method

Definition

SoundStateCalmAir:update()

Code

```
229 function CrowSoundStateCalmAir:update(dt)
230 CrowSoundStateCalmAir:superClass().update(self, dt)
231
232 self.timer = math.max(self.timer - dt, 0.0)
233 if (self.timer == 0.0) then
234 self.stateMachine:changeState(CrowsWildlife.CROW_SOUND_STATES.CALM_GROUN
235 end
236 end
```

CrowsWildlifeStates

Description

StateDefault:new

Description

Creating instance of state.

Definition

StateDefault:new()

Code

```
17  function CrowStateDefault:new(id, owner, stateMachine, custom_mt)
18  local self = SimpleState:new(id, owner, stateMachine, custom_mt or
    CrowStateDefault_mt)
19  return self
20  end
```

StateDefault:activate

Description

Activate method

Definition

StateDefault:activate()

Code

```
24  function CrowStateDefault:activate(parms)
25  CrowStateDefault:superClass().activate(self, parms)
26
27  local foundgroundTarget =
    self.owner:chooseRandomTargetNotInWater(2.0, 2.5, 0.0, 0.0)
28  if foundgroundTarget then
29  self.stateMachine:changeState("idle_walk")
30  else
31  self.owner:chooseRandomTargetNotInWater(40.0, 50.0, 5.0, 10.0,
    self.owner.steering.targetX, self.owner.steering.targetZ)
32  self.stateMachine:changeState("takeOff")
33  end
34  end
```

StateFlyGlide:new

Description

Creating instance of state.

Definition

StateFlyGlide:new()

Code

```
45  function CrowStateFlyGlide:new(id, owner, stateMachine, custom_mt)
46  local self = SimpleState:new(id, owner, stateMachine, custom_mt or
    CrowStateFlyGlide_mt)
47  self.glideTimer = 10.0 * 1000.0
48  self.timer = 0.0
49  return self
50  end
```

StateFlyGlide:activate

Description

Activate method

Definition

StateFlyGlide:activate()

Code

```
54  function CrowStateFlyGlide:activate(parms)
55  CrowStateFlyGlide:superClass().activate(self, parms)
56  self.timer = self.glideTimer
57  self.owner.steering.seekPercent = 0.6
58  self.owner.steering.wanderPercent = 0.3
59  self.owner.steering.separationPercent = 0.1
60  self.owner.steering.maxForce = 0.1
61  -- self.owner.steering.maxVelocity = 1.5 + math.random() * 1.0
62  end
```

StateFlyGlide:deactivate

Description

Deactivate method

Definition

StateFlyGlide:deactivate()

Code

```
66  function CrowStateFlyGlide:deactivate()
67  CrowStateFlyGlide:superClass().deactivate(self)
68  self.owner.steering.maxForce = 0.5
69  end
```

StateFlyGlide:update

Description

update method

Definition

StateFlyGlide:update(float dt)

Arguments

float dt in ms

Code

```
74  function CrowStateFlyGlide:update(dt)
75  CrowStateFlyGlide:superClass().update(self, dt)
76
77  self.timer = math.max(self.timer - dt, 0.0)
78  if self.timer == 0.0 or self.owner:isNearTarget(0.5) then
79  local x, y, z = getWorldTranslation(self.owner.i3dNodeId)
80  local terrainHeight =
    self.owner.spawner:getTerrainHeightWithProps(x, z)
81  y = math.max(y - terrainHeight, 5.0)
```

```
82  self.owner:chooseRandomTargetNotInWater(5.0, 10.0, y - 5.0, y + 5.0)
83  self.stateMachine:changeState("fly")
84  end
85  end
```

StateFly:new

Description

Creating instance of state.

Definition

StateFly:new()

Code

```
96  function CrowStateFly:new(id, owner, stateMachine, custom_mt)
97  local self = SimpleState:new(id, owner, stateMachine, custom_mt or CrowStateFly_mt)
98  return self
99  end
```

StateFly:activate

Description

Activate method

Definition

StateFly:activate()

Code

```
103 function CrowStateFly:activate(parms)
104 CrowStateFly:superClass().activate(self, parms)
105 self.owner.steering.seekPercent = 0.8
106 self.owner.steering.wanderPercent = 0.1
107 self.owner.steering.separationPercent = 0.1
108 self.owner.steering.maxVelocity = 8.2 + math.random() * 2.0
109 self.owner.steering.maxForce = 1.0
110 end
```

StateFly:deactivate

Description

Activate method

Definition

StateFly:deactivate()

Code

```
114 function CrowStateFly:deactivate()
115 CrowStateFly:superClass().deactivate(self)
116 self.owner.steering.maxForce = 0.5
117 end
```

StateFly:update

Description

update method

Definition

StateFly:update(float dt)

Arguments

float dt in ms

Code

```
122 function CrowStateFly:update(dt)
123 CrowStateFly:superClass().update(self, dt)
124
125 if self.owner:isNearTarget(0.5) then
126 if self.owner:isNearGround(0.5) then
127 self.stateMachine:changeState("land")
128 else
129 local choice = math.random(1, 3)
130 if choice == 1 then -- glide
131 local x, y, z = getWorldTranslation(self.owner.i3dNodeId)
132 local terrainHeight =
    self.owner.spawner:getTerrainHeightWithProps(x, z)
133 y = math.max(y - terrainHeight, 5.0)
134 self.owner:chooseRandomTargetNotInWater(10.0, 5.0, y - 5.0, y)
135 self.stateMachine:changeState("fly_glide")
136 elseif choice == 2 then -- fly some more
137 local x, y, z = getWorldTranslation(self.owner.i3dNodeId)
138 local terrainHeight =
    self.owner.spawner:getTerrainHeightWithProps(x, z)
139 y = math.max(y - terrainHeight, 5.0)
140 self.owner:chooseRandomTargetNotInWater(5.0, 15.0, y, y + 5.0)
141 elseif choice == 3 then -- land
142 self.owner:chooseRandomTargetNotInWater(35.0, 50.0, 0.0, 0.0)
143 end
144 end
145 end
146 end
```

StateLand:new

Description

Creating instance of state.

Definition

StateLand:new()

Code

```
157 function CrowStateLand:new(id, owner, stateMachine, custom_mt)
158 local self = SimpleState:new(id, owner, stateMachine, custom_mt
    or CrowStateLand_mt)
```

```
159  return self
160  end
```

StateLand:activate

Description

Activate method

Definition

StateLand:activate()

Code

```
164  function CrowStateLand:activate(parms)
165  CrowStateLand:superClass().activate(self, parms)
166  self.owner.steering.seekPercent = 1.0
167  self.owner.steering.wanderPercent = 0.0
168  self.owner.steering.separationPercent = 0.0
169  self.owner.steering.maxVelocity = 3.2 + math.random() * 2.0
170  end
```

StateLand:update

Description

update method

Definition

StateLand:update(float dt)

Arguments

float dt in ms

Code

```
175  function CrowStateLand:update(dt)
176  CrowStateLand:superClass().update(self, dt)
177  if self.owner:isNearGround(0.0) then
178  self.stateMachine:changeState("idle_walk")
179  end
180  end
```

StateTakeOff:new

Description

Creating instance of state.

Definition

StateTakeOff:new()

Code

```
191  function CrowStateTakeOff:new(id, owner, stateMachine, custom_mt)
192  local self = SimpleState:new(id, owner, stateMachine, custom_mt
     or CrowStateTakeOff_mt)
193  return self
194  end
```

StateTakeOff:activate

Description

Activate method

Definition

StateTakeOff:activate()

Code

```
198  function CrowStateTakeOff:activate(parms)
199  CrowStateTakeOff:superClass().activate(self, parms)
200  self.owner.steering.seekPercent = 0.6
201  self.owner.steering.wanderPercent = 0.4
202  self.owner.steering.separationPercent = 0.0
203  self.owner.steering.maxVelocity = 24.2 + math.random() * 2.0
204  self.owner.steering.maxForce = 10.5
205
206  if self.owner.spawner.soundFSM.currentState.id ==
     CrowsWildlife.CROW_SOUND_STATES.CALM_GROUND then
207  self.owner.spawner.soundFSM:changeState(CrowsWildlife.CROW_SOUND_STATES.
208  end
209  end
```

StateTakeOff:deactivate

Description

Activate method

Definition

StateTakeOff:deactivate()

Code

```
213  function CrowStateTakeOff:deactivate()
214  CrowStateTakeOff:superClass().deactivate(self)
215  self.owner.steering.maxForce = 0.5
216  end
```

StateTakeOff:update

Description

update method

Definition

StateTakeOff:update(float dt)

Arguments

float dt in ms

Code

```
221  function CrowStateTakeOff:update(dt)
222  CrowStateTakeOff:superClass().update(self, dt)
223
224  self.stateMachine:changeState("fly")
225  end
```

StateIdleWalk:new

Description

Creating instance of state.

Definition

StateIdleWalk:new()

Code

```
236 function CrowStateIdleWalk:new(id, owner, stateMachine,
    custom_mt)
237 local self = SimpleState:new(id, owner, stateMachine, custom_mt
    or CrowStateIdleWalk_mt)
238
239 return self
240 end
```

StateIdleWalk:activate

Description

Activate method

Definition

StateIdleWalk:activate()

Code

```
244 function CrowStateIdleWalk:activate(parms)
245 CrowStateIdleWalk:superClass().activate(self, parms)
246 self.owner.steering.seekPercent = 0.3
247 self.owner.steering.wanderPercent = 0.7
248 self.owner.steering.separationPercent = 0.0
249 self.owner.steering.maxVelocity = 0.7 + math.random() * 0.1
250 end
```

StateIdleWalk:update

Description

update method

Definition

StateIdleWalk:update(float dt)

Arguments

float dt in ms

Code

```
255 function CrowStateIdleWalk:update(dt)
256 CrowStateIdleWalk:superClass().update(self, dt)
257
258 if self.owner:isNearestPlayerClose(15.0) then
259 self.stateMachine:changeState("idle_attention")
260 elseif self.owner:isNearTarget(0.5) then
261 local choice = math.random(1, 2)
262 if choice == 1 then -- eat
263 self.stateMachine:changeState("idle_eat")
264 elseif choice == 2 then -- walk some more
```

```
265  self.owner:chooseRandomTargetNotInWater(0.0, 5.0, 0.0, 0.0)
266  end
267  end
268  end
```

StateIdleEat:new

Description

Creating instance of state.

Definition

StateIdleEat:new()

Code

```
279  function CrowStateIdleEat:new(id, owner, stateMachine, custom_mt)
280  local self = SimpleState:new(id, owner, stateMachine, custom_mt
     or CrowStateIdleEat_mt)
281  self.eatTimer = 1.0 * 1000.0
282  self.timer = 0.0
283  return self
284  end
```

StateIdleEat:activate

Description

Activate method

Definition

StateIdleEat:activate()

Code

```
288  function CrowStateIdleEat:activate(parms)
289  CrowStateIdleEat:superClass().activate(self, parms)
290  self.timer = self.eatTimer
291  self.owner.steering.seekPercent = 0.0
292  self.owner.steering.wanderPercent = 0.0
293  self.owner.steering.separationPercent = 0.0
294  self.owner.steering.maxVelocity = 0.0
295  end
```

StateIdleEat:update

Description

update method

Definition

StateIdleEat:update(float dt)

Arguments

float dt in ms

Code

```
300  function CrowStateIdleEat:update(dt)
301  CrowStateIdleEat:superClass().update(self, dt)
```

```
302
303 self.timer = math.max(self.timer - dt, 0.0)
304 if self.timer == 0.0 then
305 local foundgroundTarget =
    self.owner:chooseRandomTargetNotInWater(2.0, 2.5, 0.0, 0.0)
306 if foundgroundTarget then
307 self.stateMachine:changeState("idle_walk")
308 else
309 self.owner:chooseRandomTargetNotInWater(40.0, 50.0, 5.0, 10.0,
    self.owner.steering.targetX, self.owner.steering.targetZ)
310 self.stateMachine:changeState("takeOff")
311 end
312 end
313 end
```

StateIdleAttention:new

Description

Creating instance of state.

Definition

StateIdleAttention:new()

Code

```
324 function CrowStateIdleAttention:new(id, owner, stateMachine,
    custom_mt)
325 local self = SimpleState:new(id, owner, stateMachine, custom_mt
    or CrowStateIdleAttention_mt)
326 self.attentionTimer = 2.0 * 1000.0
327 self.timer = 0.0
328 return self
329 end
```

StateIdleAttention:activate

Description

Activate method

Definition

StateIdleAttention:activate()

Code

```
333 function CrowStateIdleAttention:activate(parms)
334 CrowStateIdleAttention:superClass().activate(self, parms)
335 self.timer = self.attentionTimer
336 self.owner.steering.seekPercent = 0.0
337 self.owner.steering.wanderPercent = 0.0
338 self.owner.steering.separationPercent = 0.0
339 self.owner.steering.maxVelocity = 0.0
340 end
```

StateIdleAttention:update

Description

update method

Definition

StateIdleAttention:update(float dt)

Arguments

float dt in ms

Code

```
345 function CrowStateIdleAttention:update(dt)
346 CrowStateIdleAttention:superClass().update(self, dt)
347
348 self.timer = math.max(self.timer - dt, 0.0)
349 local playerTooClose, posX, _, posZ =
    self.owner:isNearestPlayerClose(10.0)
350 if playerTooClose then
351 self.owner:chooseRandomTargetNotInWater(20.0, 25.0, 25.0, 30.0,
    posX, posZ)
352 self.owner.spawner:searchTree(self.owner.steering.targetX,
    self.owner.steering.targetY, self.owner.steering.targetZ, 20.0)
353 if self.owner.spawner.tree ~= nil then
354 local treeX, treeY, treeZ =
    getWorldTranslation(self.owner.spawner.tree)
355 self.owner.steering.targetX, self.owner.steering.targetZ = treeX,
    treeZ
356 end
357 self.stateMachine:changeState("takeOff")
358 elseif self.timer == 0.0 then
359 local choice = math.random(1, 2)
360
361 if choice == 1 then -- eat
362 self.stateMachine:changeState("idle_eat")
363 elseif choice == 2 then -- walk some more
364 local foundgroundTarget =
    self.owner:chooseRandomTargetNotInWater(2.0, 2.5, 0.0, 0.0)
365 if foundgroundTarget then
366 self.stateMachine:changeState("idle_walk")
367 else
368 self.owner:chooseRandomTargetNotInWater(40.0, 50.0, 5.0, 10.0,
    self.owner.steering.targetX, self.owner.steering.targetZ)
369 self.stateMachine:changeState("takeOff")
370 end
371 end
```

```
372 end
373 end
```

Dog

Description

new

Description

Creating

Definition

new()

Return Values

table instance instance of object

Code

```
18 function Dog:new(customMt)
19 local self = {}
20 setmetatable(self, customMt or Dog_mt)
21
22 self.dogInstance = 0
23 self.animalId = 0
24 self.dirtyFlag = 0
25 self.spawner = nil
26 self.entityFollow = nil
27 self.isServer = false
28 self.isClient = false
29 self.isActivable = false
30 self.isStaying = false
31 self.abandonTimer = 0.0
32 self.abandonTimerDuration = 6000
33 self.abandonRange = 100
34 self.name = ""
35 return self
36 end
```

delete

Description

Delete

Definition

delete()

Code

```
40 function Dog:delete()
41 if self.isServer then
42 g_currentMission.environment:removeHourChangeListener(self)
43 end
```

```
44  if self.dogInstance ~= nil and self.dogInstance ~= 0 then
45  delete(self.dogInstance)
46  end
47  end
```

init

Description

Definition

init()

Code

```
51  function Dog:init(spawner, xmlFilename, spawnX, spawnY, spawnZ,
    isServer, isClient)
52  self.spawner = spawner
53  self.animalId = 0
54  self.dirtyFlag = spawner:getNextDirtyFlag()
55  self.dogInstance = createAnimalCompanionManager("dog",
    xmlFilename, "dog", spawnX, spawnY, spawnZ,
    g_currentMission.terrainRootNode, isServer, isClient, 1)
56  setCompanionWaterLevel(self.dogInstance, g_currentMission.waterY)
57  setCompanionTrigger(self.dogInstance, self.animalId,
    "playerInteractionTriggerCallback", self)
58  setCompanionCommonSteeringParameters(self.dogInstance,
    self.animalId, 1.5, 2.5, MathUtil.degToRad(20.0), 0.2)
59  setCompanionWanderSteeringParameters(self.dogInstance,
    self.animalId, spawnX, spawnY, spawnZ, 1.0, 0.2, 0.01)
60  setCompanionArriveSteeringParameters(self.dogInstance,
    self.animalId, 1.0, 0.5)
61  setCompanionBehaviorWanderParameters(self.dogInstance,
    self.animalId, 10.0)
62  self.name = g_animalNameManager:getRandomName()
63
64  self.isServer = isServer
65  self.isClient = isClient
66
67  if self.isServer then
68  g_currentMission.environment:addHourChangeListener(self)
69  end
70  end
```

writeUpdateStream

Description

Write update network stream

Definition

writeUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId network stream identification
table connection connection information
integer dirtyMask

Code

```
77  function Dog:writeUpdateStream(streamId, connection, dirtyMask)
78  if not connection:getIsServer() then
79  writeAnimalCcompanionManagerToStream(self.dogInstance, streamId)
80  end
81  end
```

readUpdateStream

Description

Read update network stream

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId network stream identification
integer timestamp
table connection connection information

Code

```
88  function Dog:readUpdateStream(streamId, timestamp, connection)
89  if connection:getIsServer() then
90  readAnimalCompanionManagerFromStream(self.dogInstance, streamId,
    g_clientInterpDelay, g_packetPhysicsNetworkTime,
    g_client.tickDuration)
91  end
92  end
```

update

Description

Update

Definition

update(float dt)

Arguments

float dt time since last call in ms

Code

```
97  function Dog:update(dt)
98  -- companionDebugDraw(self.dogInstance, self.animalId, true, true, true)
99
100 if self.isStaying and self:isAbandoned(dt) then
101 local spawnX, spawnY, spawnZ = getWorldTranslation(self.spawner.spawnNod
102 setCompanionPosition(self.dogInstance, self.animalId, spawnX, spawnY,
    spawnZ)
103 self:idleWander()
104 self:resetSteeringParms()
```

```
105  self.isStaying = false
106  g_currentMission:addIngameNotification(FSBaseMission.INGAME_NOTIFICATION
     g_i18n:getText("ingameNotification_dogInDogHouse"))
107  end
108  end
```

updateTick

Description

Update network tick

Definition

updateTick(float dt)

Arguments

float dt time since last call in ms

Code

```
113  function Dog:updateTick(dt)
114  if (getAnimalCompanionNeedNetworkUpdate(self.dogInstance)) then
115  self.spawner:raiseDirtyFlags(self.dirtyFlag)
116  end
117  end
```

hourChanged

Description

Called by the environment when an hour has changed.

Definition

hourChanged()

Code

```
121  function Dog:hourChanged()
122  if not self.isServer then
123  return
124  end
125  setCompanionDaytime(self.dogInstance,
     g_currentMission.environment.dayTime)
126  end
```

setName

Description

Definition

setName()

Code

```
130  function Dog:setName(name)
131  self.name = name
132  end
```

setVisibility

Description

Definition

setVisibility()

Code

```
136  function Dog:setVisibility(state)
137  setCompanionsVisibility(self.dogInstance, state)
138  setCompanionsPhysicsUpdate(self.dogInstance, state)
139  end
```

playerInteractionTriggerCallback

Description

Callback when dog interaction trigger is activated

Definition

playerInteractionTriggerCallback(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	id of trigger
integer	otherId	id of actor
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

Code

```
148  function Dog:playerInteractionTriggerCallback(triggerId, otherId,
     onEnter, onLeave, onStay)
149  if g_currentMission.player ~= nil and otherId ==
     g_currentMission.player.rootNode and
     g_currentMission.player.farmId == self.spawner:getOwnerFarmId()
     then
150  if onEnter then
151  self.isActivable = true
152  elseif onLeave then
153  self.isActivable = false
154  end
155  end
156  self.spawner:raiseActive()
157  end
```

followEntity

Description

Activate follow player

Definition

followEntity(integer scenegraph)

Arguments

integer scenegraph node of the ball

Code

```
162  function Dog:followEntity(playerNode)
163  self.entityFollow = playerNode
164  setCompanionBehaviorIdleWander(self.dogInstance, self.animalId)
```

```
165  setCompanionFollowEntity(self.dogInstance, self.animalId,
     self.entityFollow, 3.0, 2.0, 6.0)
166  self.isStaying = false
167  end
```

goToSpawn

Description

Activate fetch ball behavior

Definition

goToSpawn(integer scenegraph)

Arguments

integer scenegraph node of the ball

Code

```
172  function Dog:goToSpawn()
173  self.entityFollow = nil
174  setCompanionBehaviorIdleWander(self.dogInstance, self.animalId)
175  setCompanionGotoEntity(self.dogInstance, self.animalId,
     self.spawner.spawnNode, 3.0, 2.0, 2.5)
176  end
```

feed

Description

Accessor to change dog name

Definition

feed()

Code

```
180  function Dog:feed()
181  if self.spawner.foodNode ~= nil then
182  setVisibility(self.spawner.foodNode, true)
183  setCompanionFeed(self.dogInstance, self.animalId,
     self.spawner.foodNode, 1.0, 0.55, 1.5)
184  end
185  end
```

fetchBall

Description

Activate fetch ball behavior

Definition

fetchBall(integer scenegraph)

Arguments

integer scenegraph node of the ball

Code

```
190  function Dog:fetchBall(playerNode, ballNode)
191  setCompanionFetch(self.dogInstance, self.animalId, ballNode, 1.0,
     0.5, playerNode, 3.0, 2.0, 3.0, 8.0)
192  end
```

pet

Description

Activate dog petting response behavior

Definition

pet()

Code

```
196  function Dog:pet()
197  setCompanionPet(self.dogInstance, self.animalId)
198  self.spawner:raiseActive()
199  end
```

idleStay

Description

Activate dog petting response behavior

Definition

idleStay()

Code

```
203  function Dog:idleStay()
204  self.entityFollow = nil
205  setCompanionBehaviorIdleStay(self.dogInstance, self.animalId)
206  self.spawner:raiseActive()
207  self.isStaying = true
208  end
```

idleWander

Description

Activate dog petting response behavior

Definition

idleWander()

Code

```
212  function Dog:idleWander()
213  setCompanionBehaviorIdleWander(self.dogInstance, self.animalId)
214  self.spawner:raiseActive()
215  end
```

isAbandoned

Description

Definition

isAbandoned()

Code

```
219  function Dog:isAbandoned(dt)
220  local isEntityInRange = false
221  for _, player in pairs(g_currentMission.players) do
222  if player.isControlled then
```

```
local entityX, entityY, entityZ = getWorldTranslation(player.rootNode)
local distance, _ = getCompanionClosestDistance(self.dogInstance, entityX, entityY, entityZ)
if distance < self.abandonRange then
isEntityInRange = true
break
end
end
end

if not isEntityInRange then
for _, enterable in pairs(g_currentMission.enterables) do
if enterable.spec_enterable ~= nil and enterable.spec_enterable.isControlled then
local entityX, entityY, entityZ = getWorldTranslation(enterable.rootNode)
local distance, _ = getCompanionClosestDistance(self.dogInstance, entityX, entityY, entityZ)
if distance < self.abandonRange then
isEntityInRange = true
break
end
end
end
end

if isEntityInRange then
self.abandonTimer = self.abandonTimerDuration
else
self.abandonTimer = self.abandonTimer - dt
if self.abandonTimer <= 0 then
return true
end
end

if self.abandonTimer > 0 then
self.spawner:raiseActive()
end
return false
end
```

resetSteeringParms

Description

Definition

resetSteeringParms()

Code

```
262 function Dog:resetSteeringParms()
263 local spawnX, spawnY, spawnZ = getWorldTranslation(self.spawner.spawnNode)
264 setCompanionCommonSteeringParameters(self.dogInstance, self.animalId, 1.5, 2.5, MathUtil.degToRad(20.0), 0.2)
265 setCompanionWanderSteeringParameters(self.dogInstance, self.animalId, spawnX, spawnY, spawnZ, 1.0, 0.2, 0.01)
266 setCompanionArriveSteeringParameters(self.dogInstance, self.animalId, 1.0, 0.5)
267 end
```

LightWildlife

Description

StateDefault:new

Description

Creating instance of sound state.

Definition

StateDefault:new()

Code

```
22 function LightWildlifeStateDefault:new(id, owner, stateMachine, custom_mt)
23 local self = SimpleState:new(id, owner, stateMachine, LightWildlifeStateDefault_mt)
24 return self
25 end
```

new

Description

new

Definition

new()

Code

```
40 function LightWildlife:new(customMt)
41 local self = {}
42
43 local mt = customMt
44 if mt == nil then
45 mt = LightWildlife_mt
46 end
47 setmetatable(self, mt)
48
```

```
49  self.type = ""
50  self.i3dFilename = ""
51  self.animals = {}
52  self.animalStates = {}
53  local defaultState = {id="default",
    class=LightWildlifeStateDefault}
54  table.insert(self.animalStates, defaultState)
55  self.soundsNode = createTransformGroup("lightWildlifeSounds")
56  return self
57  end
```

load

Description

load

Definition

load()

Code

```
61  function LightWildlife:load(xmlFilename)
62  self.xmlFilename = Utils.getFilename(xmlFilename,
    self.baseDirectory)
63  local xmlFile = loadXMLFile("TempXML", self.xmlFilename)
64  if xmlFile == 0 then
65  self.xmlFilename = nil
66  return false
67  end
68  local key = "wildlifeAnimal"
69  if hasXMLProperty(xmlFile, key) then
70  self.type = getXMLString(xmlFile, key .. "#type")
71  self.randomSpawnRadius = Utils.getNoNil(getXMLFloat(xmlFile, key
    .. "#randomSpawnRadius"), 0.0)
72  self.i3dFilename = getXMLString(xmlFile, key ..
    ".asset#filename")
73  self.shaderNodeString = getXMLString(xmlFile, key ..
    ".animations#shaderNode")
74  self.shaderParmId = getXMLString(xmlFile, key ..
    ".animations#shaderParameterId")
75  self.shaderParmOpcode = getXMLString(xmlFile, key ..
    ".animations#shaderParameterOpcode")
76  self.shaderParmSpeed = getXMLString(xmlFile, key ..
    ".animations#shaderParameterSpeed")
77  self.animations = {}
78  self.animations["default"] = {name="default", opcode=0,
    speed=0.0, transitionTimer=0.0}
79  local i = 0
```

```
80  while true do
81  local animkey = string.format(key .. ".animations.animation(%d)",
    i)
82  if not hasXMLProperty(xmlFile, animkey) then
83  break
84  end
85  local state = Utils.getNoNil(getXMLString(xmlFile,
    animkey.."#conditionState"), "")
86  local animation = {}
87  animation.opcode = Utils.getNoNil(getXMLInt(xmlFile,
    animkey.."#opcode"), 0)
88  animation.speed = Utils.getNoNil(getXMLFloat(xmlFile,
    animkey.."#speed"), 0.0)
89  animation.transitionTimer = Utils.getNoNil(getXMLFloat(xmlFile,
    animkey.."#transitionTimer"), 1.0) * 1000.0
90  self.animations[state] = animation
91  i = i + 1
92  end
93  if self.type ~= nil and self.i3dFilename ~= nil then
94  delete(xmlFile)
95  return true
96  end
97  end
98  delete(xmlFile)
99  return false
100 end
```

createAnimals

Description

createAnimals

Definition

createAnimals()

Code

```
104 function LightWildlife:createAnimals(name, spawnPosX, spawnPosY,
    spawnPosZ, nbAnimals)
105 if #self.animals == 0 then
106 for i = 1, nbAnimals do
107 local nodeId = g_i3DManager:loadSharedI3DFile(self.i3dFilename,
    self.baseDirectory, false, false, false)
108 link(getRootNode(), nodeId)
109 local shaderNode = I3DUtil.indexToObject(nodeId,
    self.shaderNodeString)
110 local animal = LightWildlifeAnimal:new(self, i, nodeId,
    shaderNode)
```

```
111
112 animal:init(spawnPosX, spawnPosZ, self.randomSpawnRadius,
    self.animalStates)
113 table.insert(self.animals, animal)
114 end
115 setWorldTranslation(self.soundsNode, spawnPosX, spawnPosY,
    spawnPosZ)
116 return 1
117 end
118 return 0
119 end
```

delete

Description

delete

Definition

delete()

Code

```
123 function LightWildlife:delete()
124 self:removeAllAnimals()
125 end
```

removeAllAnimals

Description

delete

Definition

removeAllAnimals()

Code

```
129 function LightWildlife:removeAllAnimals()
130 for _, animal in pairs(self.animals) do
131 g_i3DManager:releaseSharedI3DFile(self.i3dFilename,
    self.baseDirectory, true)
132 delete(animal.i3dNodeId)
133 end
134 self.animals = {}
135 end
```

update

Description

update

Definition

update()

Code

```
139 function LightWildlife:update(dt)
140 for _, animal in pairs(self.animals) do
```

```
141 animal:update(dt)
142 animal:updateAnimation(dt)
143 end
144 end
```

removeFarAwayAnimals

Description

removeFarAwayAnimals

Definition

removeFarAwayAnimals()

Code

```
148 function LightWildlife:removeFarAwayAnimals(maxDistance, refPosX,
    refPosY, refPosZ)
149 local removeCount = 0
150
151 for i=#self.animals, 1, -1 do
152 local x, y, z = getWorldTranslation(self.animals[i].i3dNodeId)
153 local deltaX = refPosX - x
154 local deltaY = refPosY - y
155 local deltaZ = refPosZ - z
156 local distSq = deltaX * deltaX + deltaY * deltaY + deltaZ *
    deltaZ
157
158 if distSq > (maxDistance * maxDistance) then
159 g_i3DManager:releaseSharedI3DFile(self.i3dFilename,
    self.baseDirectory, false)
160 delete(self.animals[i].i3dNodeId)
161 table.remove(self.animals, i)
162 removeCount = removeCount + 1
163 end
164 end
165 return removeCount
166 end
```

getClosestDistance

Description

getClosestDistance

Definition

getClosestDistance(float refPosX, float refPosY, float refPosZ)

Arguments

float refPosX reference x world position
float refPosY reference x world position
float refPosZ reference x world position

Return Values

float closest distance squared in m

Code

```
174  function LightWildlife:getClosestDistance(refPosX, refPosY, refPosZ)
175  local closestDistSq = nil
176
177  for _, animal in pairs(self.animals) do
178  local x, y, z = getWorldTranslation(animal.i3dNodeId)
179  local deltaX = refPosX - x
180  local deltaY = refPosY - y
181  local deltaZ = refPosZ - z
182  local distSq = deltaX * deltaX + deltaY * deltaY + deltaZ * deltaZ
183
184  if closestDistSq == nil or (closestDistSq ~= nil and distSq < closestDistSq) then
185  closestDistSq = distSq
186  end
187  end
188  if closestDistSq == nil then
189  closestDistSq = 0.0
190  end
191  return closestDistSq
192  end
```

countSpawned

Description

CountSpawned

Definition

countSpawned()

Return Values

integer returns number of animals

Code

```
197  function LightWildlife:countSpawned()
198  return #self.animals
199  end
```

getIsInWater

Description

Check if position is in water

Definition

getIsInWater(float x, float y, float z)

Arguments

float x x world position from which areas are checked

float y y world position from which areas are checked
float z z world position from which areas are checked

Return Values

bool returns true if there is water

Code

```
207  function LightWildlife:getIsInWater(x, y, z)
208  local terrainHeight = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, x, 0, z)
209  return g_currentMission.waterY >= terrainHeight
210  end
```

getTerrainHeightWithProps

Description

Definition

getTerrainHeightWithProps()

Code

```
214  function LightWildlife:getTerrainHeightWithProps(x, z)
215  local terrainY = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, x, 0, z)
216  local offset = 5.0
217  local distance = 5.0
218  local collisionMask = 63
219
220  self.groundY = -1.0
221  raycastClosest(x, terrainY + offset, z, 0.0, -1.0, 0.0, "groundRaycastCallback", 5.0, self, collisionMask)
222  return math.max(terrainY, self.groundY)
223  end
```

groundRaycastCallback

Description

Definition

groundRaycastCallback()

Code

```
227  function LightWildlife:groundRaycastCallback(hitObjectId, x, y, z, distance)
228  if hitObjectId ~= nil then
229  local objectType = getRigidBodyType(hitObjectId)
230  if objectType ~= "Dynamic" and objectType ~= "Kinematic" then
231  self.groundY = y
232  return false
233  end
234  end
```

```
235
236  return true
237  end
```

LightWildLifeAnimal

Description

lifeAnimal:new

Description

new

Definition

lifeAnimal:new()

Code

```
16  function LightWildlifeAnimal:new(spawner, id, nodeId, shaderNode,
    customMt)
17  local self = {}
18  setmetatable(self, customMt or LightWildlifeAnimal_mt)
19
20  self.stateMachine = FSMUtil.create()
21  self.spawner = spawner
22  self.id = id
23  self.i3dNodeId = nodeId
24  self.shaderNode = shaderNode
25
26  self.animation = {}
27  self.animation.currentOpcode = 0
28  self.animation.currentSpeed = 0.0
29  self.animation.transitionTimer = 0.0
30  self.animation.transitionCurrentTimer = 0.0
31  self.animation.transitionOpcode = 0
32  self.animation.transitionSpeed = 0.0
33
34  self.steering = {}
35  self.steering.targetX = 0.0
36  self.steering.targetY = 0.0
37  self.steering.targetZ = 0.0
38  self.steering.speed = 0.0
39  self.steering.velocityX = 0.0
40  self.steering.velocityY = 0.0
41  self.steering.velocityZ = 0.0
42  self.steering.wanderAngle = 0.0
43  self.steering.maxVelocity = 0.0
44  self.steering.seekPercent = 0.0
```

```
45  self.steering.wanderPercent = 0.0
46  self.steering.separationPercent = 0.0
47  self.steering.wanderX = 0.0
48  self.steering.wanderY = 0.0
49  self.steering.wanderZ = 0.0
50  self.steering.wanderTimer = 0.0
51  self.steering.maxForce = 0.5
52  self.steering.mass = 1.0
53  self.steering.radius = 1.0
54  self.steering.radiusSq = self.steering.radius *
    self.steering.radius
55  return self
56  end
```

lifeAnimal:delete

Description

delete

Definition

lifeAnimal:delete()

Code

```
60  function LightWildlifeAnimal:delete()
61  self.stateMachine:changeState("default")
62  end
```

lifeAnimal:init

Description

init

Definition

lifeAnimal:init()

Code

```
66  function LightWildlifeAnimal:init(spawnPosX, spawnPosZ, radius,
    states)
67  local offsetX = math.random() * radius
68  local offsetZ = math.random() * radius
69  local posX = spawnPosX + offsetX
70  local posZ = spawnPosZ + offsetZ
71  local posY = self.spawner:getTerrainHeightWithProps(posX, posZ)
72
73  setTranslation(self.i3dNodeId, posX, posY, posZ)
74  setRotation(self.i3dNodeId, 0, math.rad(math.random(1, 360)), 0)
75  setShaderParameter(self.shaderNode, self.spawner.shaderParmId,
    self.id, 0, 0, 0, false)
76
```

```
77  for _, state in pairs(states) do
78  self.stateMachine:addState(state.id, state.class:new(state.id,
    self, self.stateMachine))
79  end
80  self.stateMachine:changeState("default")
81  end
```

lifeAnimal:update

Description

update

Definition

lifeAnimal:update()

Code

```
85  function LightWildlifeAnimal:update(dt)
86  self:updateKinematic(dt)
87  self.stateMachine:update(dt)
88  end
```

lifeAnimal:updateKinematic

Description

Update animal kinematic parameter. Move crow using a mix of seek and wander behavior.

Definition

lifeAnimal:updateKinematic(table animal, float dt)

Arguments

table animal animal structure information

float dt delta time in ms

Code

```
 98  function LightWildlifeAnimal:updateKinematic(dt)
 99  local dtInSec = dt * 0.001
100
101  local seekForceX, seekForceY, seekForceZ = 0.0, 0.0, 0.0
102  if self.steering.seekPercent > 0.0 then
103  seekForceX, seekForceY, seekForceZ = self:calculateSeekForce()
104  seekForceX, seekForceY, seekForceZ =
     MathUtil.vector3Clamp(seekForceX, seekForceY, seekForceZ, 0.0,
     self.steering.maxForce)
105  end
106  local wanderForceX, wanderForceY, wanderForceZ = 0.0, 0.0, 0.0
107  if self.steering.wanderPercent > 0.0 then
108  wanderForceX, wanderForceY, wanderForceZ =
     self:calculateWanderForce()
109  wanderForceX, wanderForceY, wanderForceZ =
     MathUtil.vector3Clamp(wanderForceX, wanderForceY, wanderForceZ,
     0.0, self.steering.maxForce)
110  end
```

```
111 local separateForceX, separateForceY, separateForceZ = 0.0, 0.0,
    0.0
112 if self.steering.separationPercent > 0.0 then
113 separateForceX, separateForceY, separateForceZ =
    self:calculateSeparateForce()
114 separateForceX, separateForceY, separateForceZ =
    MathUtil.vector3Clamp(separateForceX, separateForceY,
    separateForceZ, 0.0, self.steering.maxForce)
115 end
116
117 local totalForceX = self.steering.seekPercent * seekForceX +
    self.steering.wanderPercent * wanderForceX +
    self.steering.separationPercent * separateForceX
118 local totalForceY = self.steering.seekPercent * seekForceY +
    self.steering.wanderPercent * wanderForceY +
    self.steering.separationPercent * separateForceY
119 local totalForceZ = self.steering.seekPercent * seekForceZ +
    self.steering.wanderPercent * wanderForceZ +
    self.steering.separationPercent * separateForceZ
120
121 totalForceX = totalForceX / self.steering.mass
122 totalForceY = totalForceY / self.steering.mass
123 totalForceZ = totalForceZ / self.steering.mass
124
125 self.steering.velocityX = self.steering.velocityX + totalForceX
126 self.steering.velocityY = self.steering.velocityY + totalForceY
127 self.steering.velocityZ = self.steering.velocityZ + totalForceZ
128 self.steering.velocityX, self.steering.velocityY,
    self.steering.velocityZ =
    MathUtil.vector3Clamp(self.steering.velocityX,
    self.steering.velocityY, self.steering.velocityZ, 0.0,
    self.steering.maxVelocity)
129
130 local positionX, positionY, positionZ =
    getWorldTranslation(self.i3dNodeId)
131 -- DEBUG
132 -- DebugUtil.drawDebugNode(self.i3dNodeId, string.format("[%d]
    state(%s) \nForces: max(%.3f) Seek(%.3f,%.3f,%.3f)
    Wander(%.3f,%.3f,%.3f) Separate(%.3f,%.3f,%.3f)\nWeights:
    seek(%.3f) wander(%.3f) separate(%.3f)\n maxVel(%.3f)
    vel(%.3f,%.3f,%.3f)",
133 -- self.id, self.stateMachine.currentState.id,
134 -- self.steering.maxForce, seekForceX, seekForceY, seekForceZ,
    wanderForceX, wanderForceY, wanderForceZ, separateForceX,
    separateForceY, separateForceZ,
```

```
135 -- self.steering.seekPercent, self.steering.wanderPercent,
    self.steering.separationPercent,
136 -- self.steering.maxVelocity, self.steering.velocityX,
    self.steering.velocityY, self.steering.velocityZ))
137 -- drawDebugLine(positionX, positionY, positionZ,1,0,0,
    self.steering.targetX,self.steering.targetY,self.steering.targetZ,
    1,0,0)
138 --
139 local terrainY = self.spawner:getTerrainHeightWithProps(positionX,
    positionZ)
140 positionX = positionX + self.steering.velocityX * dtInSec
141 positionY = math.max(positionY + self.steering.velocityY *
    dtInSec, terrainY)
142 positionZ = positionZ + self.steering.velocityZ * dtInSec
143
144 if self.steering.velocityX > 0.0 or self.steering.velocityY > 0.0
    or self.steering.velocityZ > 0.0 then
145 local newDirX, newDirY, newDirZ =
    MathUtil.vector3Normalize(self.steering.velocityX,
    self.steering.velocityY, self.steering.velocityZ)
146 setDirection(self.i3dNodeId, newDirX, newDirY, newDirZ, 0, 1, 0)
147 end
148 setWorldTranslation(self.i3dNodeId, positionX, positionY,
    positionZ)
149 end
```

lifeAnimal:calculateSeparateForce

Description

Calculate steering separate force

Definition

lifeAnimal:calculateSeparateForce()

Return Values

float separateForceX x component of steering force. default is 0.

float separateForceY y component of steering force. default is 0.

float separateForceZ z component of steering force. default is 0.

Code

```
156 function LightWildlifeAnimal:calculateSeparateForce()
157 local positionX, positionY, positionZ =
    getWorldTranslation(self.i3dNodeId)
158 local count = 0
159 local sumX, sumY, sumZ = 0.0, 0.0, 0.0
160
161 for _, animal in pairs(self.spawner.animals) do
162 local animalX, animalY, animalZ =
    getWorldTranslation(animal.i3dNodeId)
```

```
163 local deltaX, deltaY, deltaZ = positionX - animalX, positionY - animalY, positionZ - animalZ
164 local deltaXSq, deltaYSq, deltaZSq = deltaX * deltaX, deltaY * deltaY, deltaZ * deltaZ
165 local distSq = deltaXSq + deltaYSq + deltaZSq
166
167 if distSq > 0 and distSq < self.steering.radiusSq then
168 local dist = math.sqrt(distSq)
169 local normDeltaX, normDeltaY, normDeltaZ = MathUtil.vector3Normalize(deltaX, deltaY, deltaZ)
170
171 normDeltaX, normDeltaY, normDeltaZ = normDeltaX / dist, normDeltaY / dist, normDeltaZ / dist
172 sumX, sumY, sumZ = sumX + normDeltaX, sumY + normDeltaY, sumZ + normDeltaZ
173 count = count + 1
174 end
175 end
176
177 if count > 0 then
178 sumX, sumY, sumZ = sumX / count, sumY / count, sumZ / count
179 sumX, sumY, sumZ = MathUtil.vector3SetLength(sumX, sumY, sumZ, self.steering.maxVelocity)
180 return sumX - self.steering.velocityX, sumY - self.steering.velocityY, sumZ - self.steering.velocityZ
181 end
182 return 0.0, 0.0, 0.0
183 end
```

lifeAnimal:calculateWanderForce

Description

Calculate steering wander force

Definition

lifeAnimal:calculateWanderForce()

Return Values

float wanderForceX x component of steering force. default is 0.
float wanderForceY y component of steering force. default is 0.
float wanderForceZ z component of steering force. default is 0.
float wanderAngle new wandering angle in rad. default is 0.

Code

```
191 function LightWildlifeAnimal:calculateWanderForce()
192 if self.steering.velocityX ~= 0.0 and self.steering.velocityY ~= 0.0 and self.steering.velocityZ ~= 0.0 then
193 local circleCenterX = self.steering.velocityX
```

```
194  local circleCenterY = self.steering.velocityY
195  local circleCenterZ = self.steering.velocityZ
196  local circleDistance = 5.0
197  local circleRadius = 1.0
198  local angleChange = MathUtil.degToRad(35)
199  self.steering.wanderAngle = self.steering.wanderAngle +
     math.random() * angleChange - angleChange * 0.5
200  circleCenterX, circleCenterY, circleCenterZ =
     MathUtil.vector3Normalize(circleCenterX, circleCenterY,
     circleCenterZ)
201  circleCenterX = circleCenterX * circleDistance
202  circleCenterY = circleCenterY * circleDistance
203  circleCenterZ = circleCenterZ * circleDistance
204
205  local displacementX, displacementY, displacementZ =
     math.cos(self.steering.wanderAngle) * circleRadius, 0.0,
     math.sin(self.steering.wanderAngle) * circleRadius
206  local wanderForceX = circleCenterX + displacementX
207  local wanderForceY = circleCenterY + displacementY
208  local wanderForceZ = circleCenterZ + displacementZ
209  return wanderForceX, wanderForceY, wanderForceZ
210  end
211  return 0.0, 0.0, 0.0, 0.0
212  end
```

lifeAnimal:calculateSeekForce

Description

Calculate steering seek force towards target

Definition

lifeAnimal:calculateSeekForce(float positionX, float positionY, float positionZ)

Arguments

float positionX animal x world position
float positionY animal y world position
float positionZ animal z world position

Return Values

float seekForceX x component of steering force. default is 0.
float seekForceY y component of steering force. default is 0.
float seekForceZ z component of steering force. default is 0.

Code

```
222  function LightWildlifeAnimal:calculateSeekForce()
223  local positionX, positionY, positionZ =
     getWorldTranslation(self.i3dNodeId)
224  local dirToTargetX = self.steering.targetX - positionX
225  local dirToTargetY = self.steering.targetY - positionY
```

```
226 local dirToTargetZ = self.steering.targetZ - positionZ
227
228 if dirToTargetX ~= 0.0 or dirToTargetY ~= 0.0 or dirToTargetZ ~=
    0.0 then
229 dirToTargetX, dirToTargetY, dirToTargetZ =
    MathUtil.vector3Normalize(dirToTargetX, dirToTargetY,
    dirToTargetZ)
230 local desiredVelX = dirToTargetX * self.steering.maxVelocity
231 local desiredVelY = dirToTargetY * self.steering.maxVelocity
232 local desiredVelZ = dirToTargetZ * self.steering.maxVelocity
233 return desiredVelX - self.steering.velocityX, desiredVelY -
    self.steering.velocityY, desiredVelZ - self.steering.velocityZ
234 end
235 return 0.0, 0.0, 0.0
236 end
```

lifeAnimal:setTarget

Description

Sets target

Definition

lifeAnimal:setTarget()

Code

```
240 function LightWildlifeAnimal:setTarget(x, y, z)
241 self.steering.targetX = x
242 self.steering.targetY = y
243 self.steering.targetZ = z
244 end
```

lifeAnimal:isNearTarget

Description

Returns true if distance to target is less than 10cm

Definition

lifeAnimal:isNearTarget()

Code

```
248 function LightWildlifeAnimal:isNearTarget(distance)
249 local positionX, positionY, positionZ =
    getWorldTranslation(self.i3dNodeId)
250 local dirToTargetX = self.steering.targetX - positionX
251 local dirToTargetY = self.steering.targetY - positionY
252 local dirToTargetZ = self.steering.targetZ - positionZ
253 local distanceSq = dirToTargetX * dirToTargetX + dirToTargetY *
    dirToTargetY + dirToTargetZ * dirToTargetZ
254
255 return distanceSq < (distance * distance)
```

```
256 end
```

lifeAnimal:isNearGround

Description

Checks distance to ground

Definition

lifeAnimal:isNearGround(float distance)

Arguments

float distance in m

Return Values

bool true is any player is close to ground

Code

```
262 function LightWildlifeAnimal:isNearGround(distanceToGround)
263 local positionX, positionY, positionZ = getWorldTranslation(self.i3dNodeId)
264 local groundY = self.spawner:getTerrainHeightWithProps(positionX, positionZ)
265
266 return (positionY - groundY) <= distanceToGround
267 end
```

lifeAnimal:isNearestPlayerClose

Description

Calculate nearest distance to players or vehicle

Definition

lifeAnimal:isNearestPlayerClose(float distance)

Arguments

float distance in m

Return Values

bool true is any player is close

Code

```
273 function LightWildlifeAnimal:isNearestPlayerClose(distance)
274 local testDistanceSq = distance * distance
275 local positionX, positionY, positionZ = getWorldTranslation(self.i3dNodeId)
276 local minDistSq = nil
277 local nearPosX, nearPosY, nearPosZ = 0.0, 0.0, 0.0
278 for _, player in pairs(g_currentMission.players) do
279 if player.isControlled then
280 local playerX, playerY, playerZ = getWorldTranslation(player.rootNode)
281 local deltaX = playerX - positionX
282 local deltaY = playerY - positionY
283 local deltaZ = playerZ - positionZ
```

```
284 local distSq = deltaX * deltaX + deltaY * deltaY + deltaZ *
    deltaZ
285
286 if minDistSq == nil or (minDistSq ~= nil and distSq < minDistSq)
    then
287 minDistSq = distSq
288 nearPosX, nearPosY, nearPosZ = playerX, playerY, playerZ
289 end
290 end
291 end
292 for _, enterable in pairs(g_currentMission.enterables) do
293 if enterable:getIsEntered() then
294 local px, py, pz = getWorldTranslation(enterable.rootNode)
295 local deltaX = px - positionX
296 local deltaY = py - positionY
297 local deltaZ = pz - positionZ
298 local distSq = deltaX * deltaX + deltaY * deltaY + deltaZ *
    deltaZ
299
300 if minDistSq == nil or (minDistSq ~= nil and distSq < minDistSq)
    then
301 minDistSq = distSq
302 nearPosX, nearPosY, nearPosZ = px, py, pz
303 end
304 end
305 end
306 if minDistSq == nil then
307 return false
308 end
309 return minDistSq < testDistanceSq, nearPosX, nearPosY, nearPosZ
310 end
```

lifeAnimal:chooseRandomTarget

Description

Sets a random target

Definition

lifeAnimal:chooseRandomTarget(float min, float max, float min, float max, float fleePosX, float fleePosZ)

Arguments

float min	distance in m
float max	distance in m
float min	height
float max	height in m; 0.0 means ground height

float fleePosX flee position x
float fleePosZ flee position z

Code

```
320 function LightWildlifeAnimal:chooseRandomTarget(minDistance, maxDistance, minHeight, maxHeight, fleePosX, fleePosZ)
321 local finalDirX, finalDirZ = 0.0, 1.0
322
323 if fleePosX ~= nil and fleePosZ ~= nil then
324 local posX, _, posZ = getWorldTranslation(self.i3dNodeId)
325 local dirX = posX - fleePosX
326 local dirZ = posZ - fleePosZ
327 dirX, dirZ = MathUtil.vector2Normalize(dirX, dirZ)
328 finalDirX, finalDirZ = dirX, dirZ
329 else
330 local randomAngle = math.random() * 30.0
331 local angleChange = MathUtil.degToRad(randomAngle)
332 local x, z = -math.sin(angleChange), math.cos(angleChange)
333 local dirX, _, dirZ = localDirectionToWorld(self.i3dNodeId, x, 0.0, z)
334 finalDirX, finalDirZ = dirX, dirZ
335 end
336
337 local deltaDist = maxDistance - minDistance
338 local distance = minDistance + math.random() * deltaDist
339 local positionX, _, positionZ = getWorldTranslation(self.i3dNodeId)
340 local positionX, positionZ = positionX + finalDirX * distance, positionZ + finalDirZ * distance
341
342 local deltaHeight = maxHeight - minHeight
343 local positionY = (minHeight + math.random() * deltaHeight)
344 local terrainY = self.spawner:getTerrainHeightWithProps(positionX, positionZ)
345 if terrainY <= (g_currentMission.waterY + 0.1) then
346 positionY = positionY + (g_currentMission.waterY + 0.1)
347 else
348 positionY = positionY + terrainY
349 end
350
351 self:setTarget(positionX, positionY, positionZ)
352 end
```

lifeAnimal:chooseRandomTargetNotInWater

Description

Sets a random target not in water

Definition

lifeAnimal:chooseRandomTargetNotInWater(float min, float max, float min, float max, float fleePosX, float fleePosZ, bool true)

Arguments

float min	distance in m
float max	distance in m
float min	height
float max	height in m; 0.0 means ground height
float fleePosX	flee position x
float fleePosZ	flee position z
bool true	if found false if not

Code

```
363 function LightWildlifeAnimal:chooseRandomTargetNotInWater(minDistance, maxDistance, minHeight, maxHeight, fleePosX, fleePosZ)
364 local found = false
365 local numTry = 5
366
367 while not found and numTry > 0 do
368 self:chooseRandomTarget(minDistance, maxDistance, minHeight, maxHeight, fleePosX, fleePosZ)
369 found = not self.spawner:getIsInWater(self.steering.targetX, self.steering.targetY, self.steering.targetZ)
370 numTry = numTry - 1
371 end
372 return found
373 end
```

lifeAnimal:updateAnimation

Description

Update transition animation

Definition

lifeAnimal:updateAnimation(float dt)

Arguments

float dt delta time

Code

```
382 function LightWildlifeAnimal:updateAnimation(dt)
383 if self.animation.transitionCurrentTimer > 0.0 then
384 self.animation.transitionCurrentTimer = math.max(self.animation.transitionCurrentTimer - dt, 0.0)
385 if self.animation.transitionCurrentTimer == 0.0 then
```

```
386 setShaderParameter(self.shaderNode,
    self.spawner.shaderParmOpcode, self.animation.transitionOpcode,
    0, 0, 0, false)
387 setShaderParameter(self.shaderNode, self.spawner.shaderParmSpeed,
    self.animation.transitionSpeed, 0, 0, 0, false)
388 self.animation.currentOpcode = self.animation.transitionOpcode
389 self.animation.currentSpeed = self.animation.transitionSpeed
390 self.animation.transitionOpcode = 0
391 self.animation.transitionSpeed = 0.0
392 else
393 local alpha = 1.0 - (self.animation.transitionCurrentTimer /
    self.animation.transitionTimer)
394 setShaderParameter(self.shaderNode,
    self.spawner.shaderParmOpcode, self.animation.currentOpcode,
    self.animation.transitionOpcode, alpha, 0, false)
395 end
396 end
397 if not self:checkAnimationState() then
398 self:synchronizeAnimation()
399 end
400 end
```

lifeAnimal:synchronizeAnimation

Description

Definition

lifeAnimal:synchronizeAnimation()

Code

```
404 function LightWildlifeAnimal:synchronizeAnimation()
405 local currentState = self.stateMachine.currentState.id
406 local animation = self.spawner.animations[currentState]
407
408 if animation.transitionTimer > 0.0 then
409 self.animation.transitionCurrentTimer = animation.transitionTimer
410 self.animation.transitionTimer = animation.transitionTimer
411 self.animation.transitionOpcode = animation.opcode
412 self.animation.transitionSpeed = animation.speed
413 setShaderParameter(self.shaderNode,
    self.spawner.shaderParmOpcode, self.animation.currentOpcode,
    self.animation.transitionOpcode, 0, 0, false)
414 setShaderParameter(self.shaderNode, self.spawner.shaderParmSpeed,
    self.animation.currentSpeed, self.animation.transitionSpeed, 0,
    0, false)
415 end
416 end
```

lifeAnimal:checkAnimationState

Description

Definition

lifeAnimal:checkAnimationState()

Code

```
420 function LightWildlifeAnimal:checkAnimationState()
421 local currentState = self.stateMachine.currentState.id
422 local animation = self.spawner.animations[currentState]
423
424 if self.animation.transitionCurrentTimer > 0.0 and
    self.animation.transitionOpcode == animation.opcode then
425 return true
426 elseif self.animation.transitionCurrentTimer == 0.0 and
    self.animation.currentOpcode == animation.opcode then
427 return true
428 end
429 return false
430 end
```

RideableAnimal

Description

new

Description

Definition

new()

Code

```
18 function RideableAnimal:new(isServer, isClient, owner,
   fillTypeIndex, customMt)
19 local self = Animal:new(isServer, isClient, owner, fillTypeIndex,
   customMt or RideableAnimal_mt)
20
21 self.rideableVehicle = nil
22 self.enterNextRidable = true
23 self.loadAnimalId = nil
24
25 return self
26 end
```

loadFromXMLFile

Description

Definition

loadFromXMLFile()

Code

```
87 function RideableAnimal:loadFromXMLFile(xmlFile, key)
88 RideableAnimal:superClass().loadFromXMLFile(self, xmlFile, key)
```

```
89
90  local isRidingActive = getXMLBool(xmlFile,
    key.."#isRidingActive")
91  if isRidingActive then
92  local x,y,z =
    StringUtil.getVectorFromString(getXMLString(xmlFile,
    key.."#position"))
93  local xRot,yRot,zRot =
    StringUtil.getVectorFromString(getXMLString(xmlFile,
    key.."#rotation"))
94
95  xRot = math.rad(xRot or 0)
96  yRot = math.rad(yRot or 0)
97  zRot = math.rad(zRot or 0)
98
99  self.loadedRidingData = {position={x, y, z}, rotation={xRot,
    yRot, zRot}}
100 end
101 end
```

saveToXMLFile

Description

Definition

saveToXMLFile()

Code

```
105 function RideableAnimal:saveToXMLFile(xmlFile, key, usedModNames)
106 RideableAnimal:superClass().saveToXMLFile(self, xmlFile, key,
    usedModNames)
107
108 setXMLBool(xmlFile, key.."#isRidingActive", self.rideableVehicle
    ~= nil)
109
110 if self.rideableVehicle ~= nil then
111 local x,y,z = self.rideableVehicle:getPosition()
112 local xRot,yRot,zRot = self.rideableVehicle:getRotation()
113
114 setXMLString(xmlFile, key.."#position", string.format("%.4f %.4f
    %.4f", x, y, z))
115 setXMLString(xmlFile, key.."#rotation", string.format("%.4f %.4f
    %.4f", math.deg(xRot), math.deg(yRot), math.deg(zRot)))
116 end
117 end
```

activateRiding

Description

Definition

activateRiding()

Code

```
121 function RideableAnimal:activateRiding(player, noEventSend)
122
123 AnimalRidingEvent.sendEvent(self, true, player, noEventSend)
124
125 if self.owner.isServer then
126 local posX, posY, posZ = 0, nil, 0
127 local rotY = 0
128
129 if self.loadedRidingData ~= nil then
130 local pos = self.loadedRidingData.position
131 posX, posY, posZ = pos[1] or posX, pos[2] or posY, pos[3] or posZ
132 rotY = self.loadedRidingData.rotation[2]
133 else
134 local visualId = self.visualId
135 local husbandryId = self.module.husbandryId
136 posX, posY, posZ = getAnimalPosition(husbandryId, visualId)
137 _, rotY, _ = getAnimalRotation(husbandryId, visualId)
138 end
139
140 self.ridingPlayer = player
141 g_currentMission:loadVehicle(self.subType.rideableFileName, posX,
    posY, posZ, 0, rotY, true, 0.0, Vehicle.PROPERTY_STATE_OWNED,
    AccessHandler.EVERYONE, {}, nil, self.onLoadedRideable, self)
142 else
143 self.module:hideAnimal(self.visualId)
144 end
145 end
```

deactivateRiding

Description

Definition

deactivateRiding()

Code

```
149 function RideableAnimal:deactivateRiding(noEventSend)
150 AnimalRidingEvent.sendEvent(self, false, nil, noEventSend)
151
152 if self.rideableVehicle ~= nil then
153 self.rideableVehicle:setDirtChangedCallback(nil, nil)
154 self.dirtScale = self.rideableVehicle:getDirtScale()
155 if self.owner.isServer and not self.rideableVehicle.isDeleted
    then
```

```
156 g_currentMission:removeVehicle(self.rideableVehicle)
157 end
158
159 self.rideableVehicle = nil
160 end
161
162 self.ridingPlayer = nil
163 self.module:showAnimal(self.visualId)
164 end
```

onLoadedRideable

Description

Definition

onLoadedRideable()

Code

```
168 function RideableAnimal:onLoadedRideable(rideableVehicle,
    vehicleLoadState, arguments)
169 if rideableVehicle ~= nil then
170 self:finishRideable(rideableVehicle)
171 end
172 end
```

getCanBeRidden

Description

Definition

getCanBeRidden()

Code

```
207 function RideableAnimal:getCanBeRidden()
208 return self.rideableVehicle == nil
209 end
```

onDirtChangedCallback

Description

Definition

onDirtChangedCallback()

Code

```
213 function RideableAnimal:onDirtChangedCallback(dirtScale)
214 self:setDirtScale(dirtScale)
215 end
```

isOnHusbandyGround

Description

Definition

isOnHusbandyGround()

Code

```
219 function RideableAnimal:isOnHusbandyGround()
220 if self.rideableVehicle ~= nil then
```

```
221 return self.rideableVehicle:isOnHusbandyGround(self.module.rideableDeliveryArea
222 end
223 return false
224 end
```

WildlifeSpawner

Description

new

Description

Creating instance

Definition

new(table customMt)

Arguments

table customMt custom meta table

Return Values

table instance Instance of object

Code

```
40 function WildlifeSpawner:new(customMt)
41 local mt = customMt
42 if mt == nil then
43 mt = WildlifeSpawner_mt
44 end
45 local self = {}
46 setmetatable(self, mt)
47
48 self.collisionDetectionMask = 4096 -- dynamic_objects
49 self.maxCost = 0
50 self.checkTimeInterval = 0.0
51 self.nextCheckTime = 0.0
52 self.areas = {}
53 self.areasOfInterest = {}
54 self.totalCost = 0
55 self.treeCount = 0
56
57 -- Debug
58 -- if g_addCheatCommands then
59 self.debugAnimalList = {}
60 self.debugShow = false
61 self.debugShowId = WildlifeSpawner.DEBUGSHOWIDSTATES.NONE
62 self.debugShowSteering = false
63 self.debugShowAnimation = false
```

```
64  addConsoleCommand("gsToggleShowWildlife", "Toggle shows/hide all
    wildlife debug information.", "consoleCommandToggleShowWildlife",
    self)
65  addConsoleCommand("gsToggleShowWildlifeId", "Toggle shows/hide all
    wildlife animal id.", "consoleCommandToggleShowWildlifeId", self)
66  addConsoleCommand("gsToggleShowWildlifeSteering", "Toggle
    shows/hide animal steering information.",
    "consoleCommandToggleShowWildlifeSteering", self)
67  addConsoleCommand("gsToggleShowWildlifeAnimation", "Toggle
    shows/hide animal animation information.",
    "consoleCommandToggleShowWildlifeAnimation", self)
68  addConsoleCommand("gsAddWildlifeAnimalToDebug", "Adds an animal to
    a debug list.", "consoleCommandAddWildlifeAnimalToDebug", self)
69  addConsoleCommand("gsRemoveWildlifeAnimalToDebug", "Removes an
    animal to a debug list.",
    "consoleCommandRemoveWildlifeAnimalToDebug", self)
70  -- end
71
72  return self
73  end
```

delete

Description

Delete instance

Definition

delete()

Code

```
77  function WildlifeSpawner:delete()
78  self:removeAllAnimals()
79
80  removeConsoleCommand("gsToggleShowWildlife")
81  removeConsoleCommand("gsToggleShowWildlifeId")
82  removeConsoleCommand("gsToggleShowWildlifeSteering")
83  removeConsoleCommand("gsToggleShowWildlifeAnimation")
84  removeConsoleCommand("gsAddWildlifeAnimalToDebug")
85  removeConsoleCommand("gsRemoveWildlifeAnimalToDebug")
86  end
```

onConnectionClosed

Description

Definition

onConnectionClosed()

Code

```
90  function WildlifeSpawner:onConnectionClosed()
91  self:removeAllAnimals()
92  end
```

removeAllAnimals

Description

Definition

removeAllAnimals()

Code

```
96  function WildlifeSpawner:removeAllAnimals()
97  for _, area in pairs(self.areas) do
98  for _, species in pairs(area.species) do
99  for i=#species.spawned, 1, -1 do
100 if species.classType == "companionAnimal" then
101 local spawn = species.spawned[i]
102 if spawn.spawnId ~= nil then
103 delete(spawn.spawnId)
104 spawn.spawnId = nil
105 end
106 elseif species.classType == "lightWildlife" and
    species.lightWildlife ~= nil then
107 species.lightWildlife:removeAllAnimals()
108 end
109 table.remove(species.spawned, i)
110 end
111 end
112 end
113 self.totalCost = 0
114 end
```

loadMapData

Description

Loads xml file (areas and containing animals)

Definition

loadMapData(table xmlFile)

Arguments

table xmlFile XML file handle

Return Values

bool returns true if load is successful

Code

```
120 function WildlifeSpawner:loadMapData(xmlFile)
121 local filename = Utils.getFilename(getXMLString(xmlFile,
    "map.wildlife#filename"), g_currentMission.baseDirectory)
122 if filename == nil or filename == "" then
123 print("Error: Could not load wildlife config file
    '"..tostring(filename).."'!")
124 return false
```

```
125 end
126
127 local wildlifeXmlFile = loadXMLFile("wildlife", filename)
128
129 if wildlifeXmlFile ~= nil then
130 self.maxCost = Utils.getNoNil(getXMLInt(wildlifeXmlFile,
"wildlifeSpawner#maxCost"), 0)
131 self.checkTimeInterval = Utils.getNoNil(getXMLFloat(wildlifeXmlFile,
"wildlifeSpawner#checkTimeInterval"), 1.0) * 1000.0
132 self.maxAreaOfInterest = Utils.getNoNil(getXMLFloat(wildlifeXmlFile,
"wildlifeSpawner#maxAreaOfInterest"), 1)
133 self.areaOfInterestliveTime = Utils.getNoNil(getXMLFloat(wildlifeXmlFile
"wildlifeSpawner#areaOfInterestliveTime"), 1.0) * 1000.0
134 self.bypassRules = Utils.getNoNil(getXMLBool(wildlifeXmlFile,
"wildlifeSpawner#bypassRules"), false)
135 local i = 0
136 while true do
137 local areaBaseString = string.format("wildlifeSpawner.area(%d)", i)
138 if not hasXMLProperty(wildlifeXmlFile, areaBaseString) then
139 break
140 end
141 local newArea = {}
142 newArea.areaSpawnRadius = Utils.getNoNil(getXMLFloat(wildlifeXmlFile,
areaBaseString .. "#areaSpawnRadius"), 1.0)
143 newArea.areaMaxRadius = Utils.getNoNil(getXMLFloat(wildlifeXmlFile,
areaBaseString .. "#areaMaxRadius"), 1.0)
144 newArea.spawnCircleRadius = Utils.getNoNil(getXMLFloat(wildlifeXmlFile,
areaBaseString .. "#spawnCircleRadius"), 1.0)
145
146 newArea.species = {}
147 local j = 0
148 while true do
149 local speciesBaseString = string.format("%s.species(%d)", areaBaseStrin
j)
150 if not hasXMLProperty(wildlifeXmlFile, speciesBaseString) then
151 break
152 end
153 local classTypeString = getXMLString(wildlifeXmlFile, speciesBaseString
"#classType")
154 local classType = nil
155
156 if classTypeString ~= nil then
157 if string.lower(classTypeString) == "companionanimal" then
```

```
158 classType = "companionAnimal"
159 elseif string.lower(classTypeString) == "lightwildlife" then
160 classType = "lightWildlife"
161 end
162 end
163 if classType ~= nil then
164 local newSpecies = {}
165 newSpecies.classType = classType
166 newSpecies.name = getXMLString(wildlifeXmlFile, speciesBaseString ..
"#name")
167 newSpecies.configFilename = getXMLString(wildlifeXmlFile,
speciesBaseString .. "#config")
168 newSpecies.spawnRule = {}
169 newSpecies.spawnRule.hours = self:parseHours(getXMLString(wildlifeXmlFil
speciesBaseString .. ".spawnRules#hours"))
170 newSpecies.spawnRule.onField = getXMLBool(wildlifeXmlFile,
speciesBaseString .. ".spawnRules#onField")
171 newSpecies.spawnRule.hasTrees = getXMLBool(wildlifeXmlFile,
speciesBaseString .. ".spawnRules#hasTrees")
172 newSpecies.spawnRule.inWater = getXMLBool(wildlifeXmlFile,
speciesBaseString .. ".spawnRules#inWater")
173 newSpecies.cost = getXMLFloat(wildlifeXmlFile, speciesBaseString ..
".cost")
174 newSpecies.minCount = getXMLInt(wildlifeXmlFile, speciesBaseString ..
".minCount")
175 newSpecies.maxCount = getXMLInt(wildlifeXmlFile, speciesBaseString ..
".maxCount")
176 newSpecies.currentCount = 0
177 newSpecies.spawnCount = getXMLInt(wildlifeXmlFile, speciesBaseString ..
".spawnCount")
178 newSpecies.groupSpawnRadius = getXMLInt(wildlifeXmlFile, speciesBaseStri
.. ".groupSpawnRadius")
179 newSpecies.spawned = {}
180 newSpecies.lightWildlife = nil
181 if classType == "lightWildlife" then
182 if newSpecies.name == "crow" then
183 newSpecies.lightWildlife = CrowsWildlife:new()
184 newSpecies.lightWildlife:load(Utils.getNoNil(getXMLString(wildlifeXmlFil
speciesBaseString .. "#config"), ""))
185 end
186 end
187 table.insert(newArea.species, newSpecies)
188 end
```

```
189     j = j + 1
190     end
191     table.insert(self.areas, newArea)
192     i = i + 1
193     end
194     delete(wildlifeXmlFile)
195     return true
196     end
197     return false
198     end
```

unloadMapData

Description

Unload data on mission delete

Definition

unloadMapData()

Code

```
202     function WildlifeSpawner:unloadMapData()
203     self:removeAllAnimals()
204     end
```

parseHours

Description

Parse hours string

Definition

parseHours(string input)

Arguments

string input string to parse

Code

```
210     function WildlifeSpawner:parseHours(input)
211     local hoursResult = {}
212     if input ~= nil then
213     input = input:gsub('[^-,0-9]','') -- remove all but numbers, '-'
        and ','
214     local hourRangesStrings = StringUtil.splitString(",", input) --
        split by ','
215     local num = table.getn(hourRangesStrings)
216
217     for i = 1, num do
218     local hourFromStartString = StringUtil.splitString("-",
        hourRangesStrings[i]) -- split string by '-'
219     local num2 = table.getn(hourFromStartString)
220     local fromVal = 0
221     local toVal = 0
```

```lua

if num2 == 1 then
fromVal = tonumber(hourFromStartString[1])
toVal = tonumber(hourFromStartString[1])
elseif num2 == 2 then
fromVal = tonumber(hourFromStartString[1])
toVal = tonumber(hourFromStartString[2])
end

-- sanity checks
if (num2 == 1 or num2 == 2) and (fromVal <= toVal) and (fromVal >= 0 and fromVal <= 24) and (toVal >= 0 and toVal <= 24) then
table.insert(hoursResult, {from = fromVal, to = toVal})
end
end
end
return hoursResult
end
```

onGhostRemove

Description

On ghost remove callback

Definition

onGhostRemove()

Code

```lua
function WildlifeSpawner:onGhostRemove()
for _, area in pairs(self.areas) do
for _, species in pairs(area.species) do
if species.classType == "companionAnimal" then
for _, spawn in pairs(species.spawned) do
if spawn.spawnId ~= nil then
setCompanionsVisibility(spawn.spawnId, false)
setCompanionsPhysicsUpdate(spawn.spawnId, false)
end
end
end
end
end
end
```

onGhostAdd

Description

On ghost add callback

Definition

onGhostAdd()

Code

```
259  function WildlifeSpawner:onGhostAdd()
260  for _, area in pairs(self.areas) do
261  for _, species in pairs(area.species) do
262  if species.classType == "companionAnimal" then
263  for _, spawn in pairs(species.spawned) do
264  if spawn.spawnId ~= nil then
265  setCcompanionsVisibility(spawn.spawnId, true)
266  setCcompanionsPhysicsUpdate(spawn.spawnId, true)
267  end
268  end
269  end
270  end
271  end
272  end
```

update

Description

Update function. Regulate animal population.

Definition

update(float dt)

Arguments

float dt time since last call in ms

Code

```
277  function WildlifeSpawner:update(dt)
278  -- remove outdated areas of interest
279  self:updateAreaOfInterest(dt)
280  -- add animals
281  self.nextCheckTime = self.nextCheckTime - dt
282  if (self.nextCheckTime < 0.0) then
283  self.nextCheckTime = self.checkTimeInterval
284  self:updateSpawner()
285  end
286  -- remove animals too far away
287  self:removeFarAwayAnimals()
288
289  -- update animal context
290  for _, area in pairs(self.areas) do
291  for _, species in pairs(area.species) do
292  if species.classType == "companionAnimal" then
```

```
293  for _, spawn in pairs(species.spawned) do
294  if spawn.spawnId ~= nil then
295  setCompanionDaytime(spawn.spawnId,
     g_currentMission.environment.dayTime)
296  end
297  end
298  elseif species.classType == "lightWildlife" then
299  species.lightWildlife:update(dt)
300  end
301  end
302  end
303
304  if self.debugShow then
305  -- debug
306  self:debugDraw()
307  end
308  end
```

removeFarAwayAnimals

Description

Removing animals that are too far away

Definition

removeFarAwayAnimals()

Code

```
312  function WildlifeSpawner:removeFarAwayAnimals()
313  local passedTest, originX, originY, originZ =
     self:getPlayerCenter()
314
315  if passedTest then
316  for _, area in pairs(self.areas) do
317  for _, species in pairs(area.species) do
318  if species.classType == "companionAnimal" then
319  for i=#species.spawned, 1, -1 do
320  local spawn = species.spawned[i]
321  if spawn.spawnId ~= nil then
322  local distance, _ = getCompanionClosestDistance(spawn.spawnId,
     originX, originY, originZ)
323
324  if distance > area.areaMaxRadius then
325  delete(spawn.spawnId)
326  spawn.spawnId = nil
327  species.currentCount = species.currentCount - spawn.count
```

```
328 self.totalCost = self.totalCost - species.cost * spawn.count
329 table.remove(species.spawned, i)
330 end
331 end
332 end
333 elseif species.classType == "lightWildlife" and
    species.lightWildlife ~= nil then
334 local removedAnimalsCount =
    species.lightWildlife:removeFarAwayAnimals(area.areaMaxRadius,
    originX, originY, originZ)
335 if removedAnimalsCount > 0 then
336 species.currentCount = species.currentCount - removedAnimalsCount
337 self.totalCost = self.totalCost - species.cost *
    removedAnimalsCount
338 for i=#species.spawned, 1, -1 do
339 --local spawn = species.spawned[i]
340 if species.lightWildlife:countSpawned() == 0 then
341 table.remove(species.spawned, i)
342 end
343 end
344 end
345 end
346 end
347 end
348 end
349 end
```

getPlayerCenter

Description

Get player location from which the tests should be done

Definition

getPlayerCenter()

Return Values

float x world position. default is 0

float x world position. default is 0

float x world position. default is 0

Code

```
356 function WildlifeSpawner:getPlayerCenter()
357 if g_currentMission.controlPlayer and g_currentMission.player ~=
    nil then
358 local x, y, z =
    getWorldTranslation(g_currentMission.player.rootNode)
359 return true, x, y, z
```

```
360  elseif g_currentMission.controlledVehicle ~= nil then
361  local x,y,z = getWorldTranslation(g_currentMission.controlledVehicle.rootNode)
362  return true, x, y, z
363  end
364  return false, 0, 0, 0
365  end
```

updateSpawner

Description

Update spawner logic

Definition

updateSpawner()

Code

```
369  function WildlifeSpawner:updateSpawner()
370  local passedTest, x, y, z = self:getPlayerCenter()
371  if passedTest then
372  self:checkAreas(x, y, z)
373  end
374  end
```

trySpawnAtArea

Description

We try to spawn one animal type if the rules are valid

Definition

trySpawnAtArea(table species, float spawnCircleRadius, float testX, float testY, float testZ)

Arguments

table	species	species information
float	spawnCircleRadius	spawn circle radius in m
float	testX	x world position to test
float	testY	y world position to test
float	testZ	z world position to test

Return Values

bool returns true if animals are spawned

Code

```
384  function WildlifeSpawner:trySpawnAtArea(species, spawnCircleRadius, testX, testY, testZ)
385  for _, animalType in pairs(species) do
386  if self:checkArea(testX, testY, testZ, animalType.spawnRule, spawnCircleRadius) then
387  local spawnPosX = testX + math.random() * spawnCircleRadius
388  local spawnPosZ = testZ + math.random() * spawnCircleRadius
389  local spawnPosY = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, spawnPosX, 0, spawnPosZ) + 0.5
```

```
390  if self:spawnAnimals(animalType, spawnPosX, spawnPosY, spawnPosZ) then
391  return true
392  end
393  end
394  end
395  return false
396  end
```

checkAreas

Description

For each areas, we try to spawn first in areas of interests that have been registered by workAreas. If there is no spawn then, we try to spawn at a random position around the player.

Definition

checkAreas(float x, float y, float z)

Arguments

float x	x world position from which areas are checked
float y	y world position from which areas are checked
float z	z world position from which areas are checked

Code

```
403  function WildlifeSpawner:checkAreas(x, y, z)
404  local testX = x
405  local testZ = z
406  local testY = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, testX, 0, testZ) + 0.5
407
408  for _, area in pairs(self.areas) do
409  local hasSpawned = false
410  for _, interestArea in pairs(self.areasOfInterest) do
411  local distSq = (testX - interestArea.positionX) * (testX - interestArea.positionX) + (testZ - interestArea.positionZ) * (testZ - interestArea.positionZ)
412  if (distSq < (area.areaSpawnRadius * area.areaSpawnRadius)) then
413  hasSpawned = self:trySpawnAtArea(area.species, interestArea.radius, testX, testY, testZ)
414  if hasSpawned then
415  break
416  end
417  end
418  end
419  if not hasSpawned then
420  local angle = math.rad(math.random(0, 360))
```

```
421 testX = x + area.areaSpawnRadius * math.cos(angle) - area.areaSpawnRadius * math.sin(angle)
422 testZ = z + area.areaSpawnRadius * math.cos(angle) + area.areaSpawnRadius * math.sin(angle)
423 testY = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, testX, 0, testZ) + 0.5
424
425 self:trySpawnAtArea(area.species, area.spawnCircleRadius, testX, testY, testZ)
426 end
427 end
428 end
```

checkArea

Description

Check area with to see if we should spawn animals (trees amount, is a field, is in water, hours of the day)

Definition

checkArea(float x, float y, float z, table rules, float radius)

Arguments

float x	x world position from which areas are checked
float y	y world position from which areas are checked
float z	z world position from which areas are checked
table rules	rules to check against for the species
float radius	radius of the test

Return Values

bool returns true if all tests are validated

Code

```
438 function WildlifeSpawner:checkArea(x, y, z, rules, radius)
439 local nbTrees = self:countTrees(x, y, z, radius)
440 local isOnField = self:getIsOnField(x, y, z)
441 local isInWater = self:getIsInWater(x, y, z)
442 local hoursTest = self:checkHours(rules)
443 local onFieldTest = (rules.onField and isOnField) or (not rules.onField and not isOnField)
444 local hasTreesTest = (rules.hasTrees and nbTrees > 0) or (not rules.hasTrees and nbTrees == 0)
445 local inWaterTest = (rules.inWater and isInWater) or (not rules.inWater and not isInWater)
446 if self.bypassRules or (hoursTest and onFieldTest and hasTreesTest and inWaterTest) then
447 return true
448 end
449 return false
```

```
450 end
```

countTrees

Description

Count number of trees

Definition

countTrees(float x, float y, float z, radius radius)

Arguments

float	x	x world position from which areas are checked
float	y	y world position from which areas are checked
float	z	z world position from which areas are checked
radius	radius	radius of the test in m

Return Values

integer number of trees found

Code

```
459 function WildlifeSpawner:countTrees(x, y, z, radius)
460 self.treeCount = 0
461 overlapSphere(x, y, z, radius, "treeCountTestCallback", self, 2, false, true, false)
462 --overlapBox(x, y, z, 0, 1, 0, radius, radius, radius, "treeCountTestCallback", self, self.collisionDetectionMask, false, true, false)
463 return self.treeCount
464 end
```

treeCountTestCallback

Description

Tree count callback

Definition

treeCountTestCallback(integer transformId)

Arguments

integer transformId - transformId of the element detected in the overlap test

Return Values

bool true to continue counting trees

Code

```
470 function WildlifeSpawner:treeCountTestCallback(transformId)
471 if transformId ~= 0 and getHasClassId(transformId, ClassIds.SHAPE) then
472 local object = getParent(transformId)
473 if object ~= nil and getSplitType(transformId) ~= 0 then
474 self.treeCount = self.treeCount + 1
475 end
476 end
477 return true
478 end
```

getIsOnField

Description

Check if position is on a field

Definition

getIsOnField(float x, float y, float z)

Arguments

float x x world position from which areas are checked

float y y world position from which areas are checked

float z z world position from which areas are checked

Return Values

bool return true is on field

Code

```
486  function WildlifeSpawner:getIsOnField(x, y, z)
487  local densityBits = getDensityAtWorldPos(g_currentMission.terrainDetailId, x, y, z)
488  return densityBits ~= 0
489  end
```

getIsInWater

Description

Check if position is in water

Definition

getIsInWater(float x, float y, float z)

Arguments

float x x world position from which areas are checked

float y y world position from which areas are checked

float z z world position from which areas are checked

Return Values

bool returns true if there is water

Code

```
497  function WildlifeSpawner:getIsInWater(x, y, z)
498  local terrainHeight = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, x, 0, z)
499  return g_currentMission.waterY >= terrainHeight
500  end
```

checkHours

Description

Check daytime hour is in one of the valid hour ranges

Definition

checkHours()

Return Values

bool returns true if current time in hours range

Code

```lua
function WildlifeSpawner:checkHours(rules)
local currentHour = math.floor(g_currentMission.environment.dayTime / (60 * 60 * 1000))

for _, hours in pairs(rules.hours) do
if (currentHour >= hours.from and currentHour <= hours.to) then
return true
end
end
return false
end
```

countAnimalsTobeSpawned

Description

Calculate a random amount of animals that can be spawned for a species

Definition

countAnimalsTobeSpawned(table species)

Arguments

table species

Return Values

integer returns the number of animals to spawn

Code

```lua
function WildlifeSpawner:countAnimalsTobeSpawned(species)
local remainingAnimal = math.floor((self.maxCost - self.totalCost) / species.cost)

if species.minCount > remainingAnimal then
return 0
end
local deltaNbAnimals = species.maxCount - species.minCount
local nbAnimals = species.minCount + math.random(1, deltaNbAnimals)
nbAnimals = math.min(remainingAnimal, nbAnimals)
return nbAnimals
end
```

spawnAnimals

Description

Spawn animals

Definition

spawnAnimals(table species, float spawnPosX, float spawnPosY, float spawnPosZ)

Arguments

table species species to spawn

float spawnPosX x world position to spawn
float spawnPosY y world position to spawn
float spawnPosZ z world position to spawn

Return Values

bool returns true if animals are spawned

Code

```
540 function WildlifeSpawner:spawnAnimals(species, spawnPosX, spawnPosY, spawnPosZ)
541 local xmlFilename = Utils.getFilename(species.configFilename, g_currentMission.loadingMapBaseDirectory)
542
543 if species.name == nil or
544 xmlFilename == nil or
545 g_currentMission.terrainRootNode == nil or
546 species.currentCount >= species.maxCount then
547 return false
548 end
549 local nbAnimals = self:countAnimalsTobeSpawned(species)
550 if nbAnimals == 0 then
551 return false
552 end
553 local id = nil
554 if species.classType == "companionAnimal" then
555 id = createAnimalCompanionManager(species.name, xmlFilename, "wildlifeAnimal", spawnPosX, spawnPosY, spawnPosZ, g_currentMission.terrainRootNode, g_currentMission:getIsServer(), g_currentMission:getIsClient(), nbAnimals)
556 setCompanionWaterLevel(id, g_currentMission.waterY)
557 for animalId=0, nbAnimals - 1 do
558 setCompanionCommonSteeringParameters(id, animalId, 0.5, 3.0, MathUtil.degToRad(20.0), 0.2)
559 setCompanionWanderSteeringParameters(id, animalId, spawnPosX, spawnPosY, spawnPosZ, 10.0, 12.0, 0.01)
560 end
561 elseif species.classType == "lightWildlife" then
562 id = species.lightWildlife:createAnimals(species.name, spawnPosX, spawnPosY, spawnPosZ, nbAnimals)
563 end
564 if (id ~= nil) and (id ~= 0) then
565 table.insert(species.spawned, {spawnId = id, posX = spawnPosX, posY = spawnPosY, posZ = spawnPosZ, count=nbAnimals})
566 species.currentCount = species.currentCount + nbAnimals
567 self.totalCost = self.totalCost + species.cost * nbAnimals
```

```
568     return true
569     end
570     return false
571     end
```

debugDraw

Description

Display debug information

Definition

debugDraw()

Code

```
575     function WildlifeSpawner:debugDraw()
576     renderText(0.02, 0.95, 0.02, string.format("Wildlife
        Info\nCost(%d / %d)", self.totalCost, self.maxCost))
577     local passedTest, originX, originY, originZ =
        self:getPlayerCenter()
578
579     if passedTest then
580     for _, area in pairs(self.areas) do
581     for _, species in pairs(area.species) do
582     for _, spawn in pairs(species.spawned) do
583     if spawn.spawnId ~= nil then
584     local distance = 0.0
585
586     if species.classType == "companionAnimal" then
587     distance, _ = getCompanionClosestDistance(spawn.spawnId, originX,
        originY, originZ)
588     elseif species.classType == "lightWildlife" then
589     distance = species.lightWildlife:getClosestDistance(originX,
        originY, originZ)
590     distance = math.sqrt(distance)
591     end
592     local text = string.format("[%s][%d]\n- nearest player distance
        (%.3f)", species.name, spawn.spawnId, distance)
593
594     Utils.renderTextAtWorldPosition(spawn.posX, spawn.posY + 0.12,
        spawn.posZ, text, getCorrectTextSize(0.012), 0)
595     DebugUtil.drawDebugCubeAtWorldPos(spawn.posX, spawn.posY,
        spawn.posZ, 1, 0, 0, 0, 1, 0, 0.05, 0.05, 0.05, 1.0, 1.0, 0.0)
596     DebugUtil.drawDebugCircle(spawn.posX, spawn.posY, spawn.posZ,
        species.groupSpawnRadius, 10.0, {1.0, 1.0, 0.0})
597     end
598     end
```

```
599 end
600 end
601 end
602
603 for _, area in pairs(self.areas) do
604 for _, species in pairs(area.species) do
605 if species.classType == "companionAnimal" then
606 for _, spawn in pairs(species.spawned) do
607 for animalId=0, spawn.count - 1 do
608 local showAdditionalInfo = self:isInDebugList(spawn.spawnId,
    animalId)
609 local showId = (self.debugShowId ==
    WildlifeSpawner.DEBUGSHOWIDSTATES.SINGLE and showAdditionalInfo)
    or self.debugShowId == WildlifeSpawner.DEBUGSHOWIDSTATES.ALL
610 companionDebugDraw(spawn.spawnId, animalId, showId,
    showAdditionalInfo and self.debugShowSteering, showAdditionalInfo
    and self.debugShowAnimation)
611 end
612 end
613 end
614 end
615 end
616 end
```

updateAreaOfInterest

Description

Removes an area of interest if the time to live expires

Definition

updateAreaOfInterest(float dt)

Arguments

float dt delta time in ms

Code

```
621 function WildlifeSpawner:updateAreaOfInterest(dt)
622 for key, area in pairs(self.areasOfInterest) do
623 area.timeToLive = area.timeToLive - dt
624 if area.timeToLive <= 0.0 then
625 table.remove(self.areasOfInterest, key)
626 end
627 end
628 end
```

addAreaOfInterest

Description

Adds an area of interest to check

Definition

addAreaOfInterest(float liveTime, float posX, float posZ, float radius)

Arguments

float liveTime	how long the area is available
float posX	x world position
float posZ	z world position
float radius	radius of the area in m

Code

```
636  function WildlifeSpawner:addAreaOfInterest(liveTime, posX, posZ,
     radius)
637  if #self.areasOfInterest <= self.maxAreaOfInterest then
638  local info = {}
639  info.liveTime = liveTime
640  info.positionX = posX
641  info.positionZ = posZ
642  info.radius = radius
643  info.timeToLive = self.areaOfInterestliveTime
644  table.insert(self.areasOfInterest, info)
645  end
646  end
```

consoleCommandToggleShowWildlife

Description

Definition

consoleCommandToggleShowWildlife()

Return Values

string that will be displayed on console

Code

```
651  function WildlifeSpawner:consoleCommandToggleShowWildlife()
652  self.debugShow = not self.debugShow
653
654  return string.format("-- show Wildlife debug = %s",
     tostring(self.debugShow))
655  end
```

consoleCommandToggleShowWildlifeId

Description

Definition

consoleCommandToggleShowWildlifeId()

Return Values

string that will be displayed on console

Code

```
660  function WildlifeSpawner:consoleCommandToggleShowWildlifeId()
661  self.debugShowId = self.debugShowId + 1
662
```

```
663 if self.debugShowId > WildlifeSpawner.DEBUGSHOWIDSTATES.MAX then
664 self.debugShowId = WildlifeSpawner.DEBUGSHOWIDSTATES.NONE
665 end
666
667 local state = ""
668 if (self.debugShowId == WildlifeSpawner.DEBUGSHOWIDSTATES.NONE)
    then
669 state = "NONE"
670 elseif (self.debugShowId ==
    WildlifeSpawner.DEBUGSHOWIDSTATES.SINGLE) then
671 state = "SINGLE"
672 elseif (self.debugShowId ==
    WildlifeSpawner.DEBUGSHOWIDSTATES.ALL) then
673 state = "ALL"
674 end
675 return string.format("-- show Wildlife Id = %s", state)
676 end
```

consoleCommandToggleShowWildlifeSteering

Description

Definition

consoleCommandToggleShowWildlifeSteering()

Return Values

string that will be displayed on console

Code

```
681 function
    WildlifeSpawner:consoleCommandToggleShowWildlifeSteering()
682 self.debugShowSteering = not self.debugShowSteering
683
684 return string.format("-- show Wildlife Steering = %s",
    tostring(self.debugShowSteering))
685 end
```

consoleCommandToggleShowWildlifeAnimation

Description

Definition

consoleCommandToggleShowWildlifeAnimation()

Return Values

string that will be displayed on console

Code

```
690 function
    WildlifeSpawner:consoleCommandToggleShowWildlifeAnimation()
691 self.debugShowAnimation = not self.debugShowAnimation
692
693 return string.format("-- show Wildlife Animation = %s",
    tostring(self.debugShowAnimation))
```

```
694 end
```

animalExists

Description

Definition

animalExists()

Return Values

Code

```
699 function WildlifeSpawner:animalExists(spawnId, animalId)
700 for _, area in pairs(self.areas) do
701 for _, species in pairs(area.species) do
702 if species.classType == "companionAnimal" then
703 for _, spawn in pairs(species.spawned) do
704 if spawn.spawnId == spawnId and animalId < spawn.count then
705 return true
706 end
707 end
708 end
709 end
710 end
711 return false
712 end
```

isInDebugList

Description

Definition

isInDebugList()

Return Values

Code

```
717 function WildlifeSpawner:isInDebugList(spawnId, animalId)
718 for key, entry in pairs(self.debugAnimalList) do
719 if entry.spawnId == spawnId and entry.animalId == animalId then
720 return true
721 end
722 end
723 return false
724 end
```

consoleCommandAddWildlifeAnimalToDebug

Description

Definition

consoleCommandAddWildlifeAnimalToDebug()

Return Values

string that will be displayed on console

Code

```
729 function WildlifeSpawner:consoleCommandAddWildlifeAnimalToDebug(spawnId, animalId)
730 local argsTest = true
731 spawnId = tonumber(spawnId)
732 if spawnId == nil then
733 argsTest = false
734 end
735 animalId = tonumber(animalId)
736 if animalId == nil then
737 argsTest = false
738 end
739
740 if argsTest and self:animalExists(spawnId, animalId) then
741 table.insert(self.debugAnimalList, {spawnId = spawnId, animalId = animalId})
742 return string.format("-- added [spawn(%d)][animal(%d)] to debug list.", spawnId, animalId)
743 else
744 return string.format("-- gsAddWildlifeAnimalToDebug [spawnId][animalId]")
745 end
746 end
```

consoleCommandRemoveWildlifeAnimalToDebug

Description

Definition

consoleCommandRemoveWildlifeAnimalToDebug()

Return Values

string that will be displayed on console

Code

```
751 function WildlifeSpawner:consoleCommandRemoveWildlifeAnimalToDebug(spawnId, animalId)
752 local argsTest = true
753 spawnId = tonumber(spawnId)
754 if spawnId == nil then
755 argsTest = false
756 end
757 animalId = tonumber(animalId)
758 if animalId == nil then
759 argsTest = false
760 end
761 if argsTest then
```

```
762 for key, entry in pairs(self.debugAnimalList) do
763 if entry.spawnId == spawnId and entry.animalId == animalId then
764 table.remove(self.debugAnimalList, key)
765 return string.format("-- removed [spawn(%d)][animal(%d)] from debug list.", spawnId, animalId)
766 end
767 end
768 end
769 return string.format("-- gsRemoveWildlifeAnimalToDebug [spawnId][animalId]")
770 end
```

HusbandryModuleAnimals

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
32 function HusbandryModuleAnimal:new(customMt)
33 local self = HusbandryModuleBase:new(customMt or HusbandryModuleAnimal_mt)
34 return self
35 end
```

delete

Description

Deletes instance

Definition

delete()

Code

```
39 function HusbandryModuleAnimal:delete()
40 for i=#self.animals, 1, -1 do
41 local animal = self.animals[i]
42 self:removeSingleAnimal(animal, true)
43 animal:delete()
44 end
45 self:updateVisualAnimals()
46
47 if self.animalLoadingTrigger ~= nil then
48 self.animalLoadingTrigger:delete()
```

```
49  self.animalLoadingTrigger = nil
50  end
51
52  if self.husbandryId ~= 0 then
53  delete(self.husbandryId)
54  end
55  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
59  function HusbandryModuleAnimal:initDataStructures()
60  HusbandryModuleAnimal:superClass().initDataStructures(self)
61
62  self.animalType = ""
63
64  self.updateVisuals = false
65
66  self.animalsToAdd = nil
67  self.renamingTasks = nil
68  self.animals = {}
69  self.typedAnimals = {}
70
71  self.reproductionRatesPerDay = {}
72  self.newAnimalPercentages = {}
73
74  self.visualIdToAnimal = {}
75  self.visualAnimals = {}
76
77  self.maxNumAnimals = 0
78  self.carryingCapacity = 0
79
80  self.navMeshNode = nil
81  self.animalHusbandryXMLFilename = ""
82  self.placementRaycastDistance = 2.0
83  self.husbandryId = 0
84  self.animalLoadingTrigger = nil
85  self.rideableDeliveryArea = {}
```

```
 86  self.rideableDeliveryArea.startNode = nil
 87  self.rideableDeliveryArea.widthNode = nil
 88  self.rideableDeliveryArea.heightNode = nil
 89
 90  local profileClass = Utils.getPerformanceClassId()
 91  if profileClass >= GS_PROFILE_VERY_HIGH then
 92  self.maxNumVisualAnimals = 25
 93  elseif profileClass >= GS_PROFILE_HIGH then
 94  self.maxNumVisualAnimals = 20
 95  else
 96  self.maxNumVisualAnimals = 16
 97  end
 98
 99  self.isDirtyFlag = 0
100  end
```

load

Description

Loads data from xml

Definition

load(table xmlFile, string xmlKey, table rootNode)

Arguments

table	xmlFile	handle
string	xmlKey	from which to read the configuration
table	rootNode	of the husbandry

Return Values

boolean true if loading was successful else false

Code

```
108  function HusbandryModuleAnimal:load(xmlFile, configKey, rootNode,
     owner)
109  if not HusbandryModuleAnimal:superClass().load(self, xmlFile,
     configKey, rootNode, owner) then
110  return false
111  end
112
113  if not hasXMLProperty(xmlFile, configKey) then
114  return false
115  end
116
117  self.animalType = getXMLString(xmlFile, configKey .. "#type")
118  if self.animalType == nil then
119  g_logManager:error("Missing animal type for husbandry!")
120  return false
```

```
121 end
122
123 if self.animalType ~= nil then
124 local animals = g_animalManager:getAnimalsByType(self.animalType)
125 if animals == nil then
126 g_logManager:error("Animal type '%s' not found!",
    self.animalType)
127 return false
128 end
129
130 for _, subType in ipairs(animals.subTypes) do
131 self.reproductionRatesPerDay[subType.fillType] = 0
132 self.newAnimalPercentages[subType.fillType] = 0
133 end
134 end
135
136 self.navMeshNode = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, configKey .. "#navmeshNode"))
137 if self.navMeshNode == nil then
138 g_logManager:error("Invalid navMeshIndex in '%s'!",
    getName(rootNode))
139 return false
140 end
141
142 self.rideableDeliveryArea = {}
143 self.rideableDeliveryArea.startNode =
    I3DUtil.indexToObject(rootNode, getXMLString(xmlFile, configKey
    .. "#rideableDeliveryStartNode"))
144 self.rideableDeliveryArea.widthNode =
    I3DUtil.indexToObject(rootNode, getXMLString(xmlFile, configKey
    .. "#rideableDeliveryWidthNode"))
145 self.rideableDeliveryArea.heightNode =
    I3DUtil.indexToObject(rootNode, getXMLString(xmlFile, configKey
    .. "#rideableDeliveryHeightNode"))
146
147 self.animalHusbandryXMLFilename =
    Utils.getNoNil(getXMLString(xmlFile, configKey ..
    "#husbandryFileName"), "")
148 self.animalHusbandryXMLFilename =
    Utils.getFilename(self.animalHusbandryXMLFilename,
    g_currentMission.baseDirectory)
149 if self.animalHusbandryXMLFilename == "" then
150 g_logManager:error("Missing animal husbandry xml filename!")
```

```
151     return false
152     end
153
154     self.placementRaycastDistance =
        Utils.getNoNil(getXMLFloat(xmlFile, configKey ..
        "#placementRaycastDistance"), 2.0)
155
156     self.maxNumAnimals = Utils.getNoNil(getXMLInt(xmlFile, configKey
        .. "#maxNumAnimals"), 16)
157     self.carryingCapacity = Utils.getNoNil(getXMLInt(xmlFile,
        configKey .. "#carryingCapacity"), 16)
158
159     local animalLoadTriggerId = I3DUtil.indexToObject(rootNode,
        getXMLString(xmlFile, configKey .. "#animalLoadTriggerNode"))
160     if animalLoadTriggerId ~= nil then
161     self.animalLoadingTrigger =
        AnimalLoadingTrigger:new(self.owner.isServer,
        self.owner.isClient)
162     if self.animalLoadingTrigger ~= nil then
163     self.animalLoadingTrigger:load(animalLoadTriggerId, self.owner)
164     self.animalLoadingTrigger:register(true)
165     end
166     end
167
168     if self.animalType == "HORSE" then
169     self.maxNumVisualAnimals = math.min(self.maxNumAnimals, 16)
170     self.maxNumAnimals = math.min(self.maxNumAnimals, 16)
171     end
172
173     local maxAnimals = (2^HusbandryModuleAnimal.SEND_NUM_BITS) - 1
174     if self.maxNumAnimals > maxAnimals then
175     g_logManager:warning("Maximum number of animals reached! Maximum
        is '%d'!", maxAnimals)
176     self.maxNumAnimals = maxAnimals
177     end
178
179     self.isDirtyFlag = self.owner:getNextDirtyFlag()
180
181     return true
182     end
```

finalizePlacement

Description

Called when husbandry is placed

Definition

finalizePlacement()

Return Values

bool true if successful

Code

```
187  function HusbandryModuleAnimal:finalizePlacement()
188  if not HusbandryModuleAnimal:superClass().finalizePlacement(self)
     then
189  return false
190  end
191
192  self.husbandryId = createAnimalHusbandry(self.animalType,
     self.navMeshNode, self.animalHusbandryXMLFilename,
     self.placementRaycastDistance, true)
193
194  if self.husbandryId == 0 then
195  g_logManager:error("Could not create animal husbandry!")
196  return false
197  end
198
199  return true
200  end
```

readStream

Description

Reads network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
206  function HusbandryModuleAnimal:readStream(streamId, connection)
207  HusbandryModuleAnimal:superClass().readStream(self, streamId,
     connection)
208
209  self.animalsToAdd = {}
210
211  local numAnimals = streamReadUInt16(streamId)
212  for i = 1, numAnimals do
213  local animalObjectId = NetworkUtil.readNodeObjectId(streamId)
214  table.insert(self.animalsToAdd, animalObjectId)
215  end
```

```
216 end
```

writeStream

Description

Writes network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
222 function HusbandryModuleAnimal:writeStream(streamId, connection)
223 HusbandryModuleAnimal:superClass().writeStream(self, streamId,
    connection)
224
225 streamWriteUInt16(streamId, #self.animals)
226 for _, animal in ipairs(self.animals) do
227 local animalObjectId = NetworkUtil.getObjectId(animal)
228 NetworkUtil.writeNodeObjectId(streamId, animalObjectId)
229 end
230 end
```

loadFromXMLFile

Description

Loads information from attributes and node. Retrives from xml file information.

Definition

loadFromXMLFile(table xmlFile, string key)

Arguments

table xmlFile XML file handler

string key XML base key

Code

```
236 function HusbandryModuleAnimal:loadFromXMLFile(xmlFile, key)
237 HusbandryModuleAnimal:superClass().loadFromXMLFile(self, xmlFile,
    key)
238
239 self.animalsToAdd = {}
240 local i = 0
241 while true do
242 local animalKey = string.format("%s.animal(%d)", key, i)
243 if not hasXMLProperty(xmlFile, animalKey) then
244 break
245 end
246
247
```

```
248 local animal = Animal.createFromXMLFile(xmlFile, animalKey,
    self.owner.isServer, self.owner.isClient, self.owner)
249 if animal ~= nil then
250 animal:register()
251 table.insert(self.animalsToAdd, NetworkUtil.getObjectId(animal))
252 end
253
254 i = i + 1
255 end
256
257 local i = 0
258 while true do
259 local animalKey = string.format("%s.breeding(%d)", key, i)
260 if not hasXMLProperty(xmlFile, animalKey) then
261 break
262 end
263
264 local fillTypeName = getXMLString(xmlFile,
    animalKey.."#fillType")
265 local fillTypeIndex =
    g_fillTypeManager:getFillTypeIndexByName(fillTypeName)
266 if fillTypeIndex ~= nil then
267 self.newAnimalPercentages[fillTypeIndex] = getXMLFloat(xmlFile,
    animalKey.."#percentage") or 0
268 end
269
270 i = i + 1
271 end
272 end
```

saveToXMLFile

Description

Save module to xml

Definition

saveToXMLFile(integer xmlFile, string key, table usedModNames)

Arguments

integer	xmlFile	file handle
string	key	xml key
table	usedModNames	

Code

```
279 function HusbandryModuleAnimal:saveToXMLFile(xmlFile, key,
    usedModNames)
```

```
280 HusbandryModuleAnimal:superClass().saveToXMLFile(self, xmlFile,
    key, usedModNames)
281
282 for i, animal in ipairs(self.animals) do
283 local animalKey = string.format("%s.animal(%d)", key, i-1)
284 animal:saveToXMLFile(xmlFile, animalKey, usedModNames)
285 end
286
287 local i = 0
288 for fillTypeIndex, percentage in pairs(self.newAnimalPercentages)
    do
289 if percentage > 0 then
290 local animalKey = string.format("%s.breeding(%d)", key, i)
291 setXMLString(xmlFile, animalKey.."#fillType",
    g_fillTypeManager:getFillTypeNameByIndex(fillTypeIndex))
292 setXMLFloat(xmlFile, animalKey.."#percentage", percentage)
293 i = i + 1
294 end
295 end
296 end
```

update

Description

Update module

Definition

update(float dt)

Arguments

float dt delta time

Code

```
301 function HusbandryModuleAnimal:update(dt)
302 HusbandryModuleAnimal:superClass().update(self, dt)
303
304 local needsUpdate = self.updateVisuals or self.animalsToAdd ~=
    nil or self.renamingTasks ~= nil or self.pendingRideables ~= nil
305
306 if isHusbandryReady(self.husbandryId) then
307 if self.animalsToAdd ~= nil then
308 for i=#self.animalsToAdd, 1, -1 do
309 local id = self.animalsToAdd[i]
310 local animal = NetworkUtil.getObject(id)
311 if animal ~= nil then
312 self:addSingleAnimal(animal, true)
313 table.remove(self.animalsToAdd, i)
```

```
314 end
315 end
316 if #self.animalsToAdd == 0 then
317 self.animalsToAdd = nil
318 self:updateBreeding(0)
319 end
320 end
321
322 if self.renamingTasks ~= nil then
323 for i=#self.renamingTasks, 1, -1 do
324 local data = self.renamingTasks[i]
325 local animal = NetworkUtil.getObject(data.animalId)
326 if animal ~= nil then
327 animal:setName(data.name)
328 table.remove(self.renamingTasks, i)
329 end
330 end
331
332 if #self.renamingTasks == 0 then
333 self.renamingTasks = nil
334 end
335 end
336
337 if self.updateVisuals then
338 self:updateVisualAnimals()
339 self.updateVisuals = false
340 end
341
342 if self.pendingRideables ~= nil then
343 for i=#self.pendingRideables, 1, -1 do
344 local animal = self.pendingRideables[i]
345 if animal:tryToFinishRideable() then
346 table.remove(self.pendingRideables, i)
347 end
348 end
349
350 if #self.pendingRideables == 0 then
351 self.pendingRideables = nil
352 end
353 end
```

```
354
355 needsUpdate = needsUpdate or self.animalsToAdd ~= nil
356 end
357
358 return needsUpdate
359 end
```

getSupportsSubType

Description

Definition

getSupportsSubType()

Code

```
381 function HusbandryModuleAnimal:getSupportsSubType(subType)
382 return subType.type == self.animalType
383 end
```

addAnimals

Description

Definition

addAnimals()

Code

```
387 function HusbandryModuleAnimal:addAnimals(animals, noEventSend)
388 local sendAnimals = {}
389
390 for _, animal in ipairs(animals) do
391 if self:addSingleAnimal(animal, true) then
392 table.insert(sendAnimals, animal)
393 end
394 end
395
396 if self.owner.isServer and noEventSend == nil or not noEventSend
    then
397 g_server:broadcastEvent(AnimalAddEvent:new(self.owner,
    sendAnimals), nil, nil, nil, true)
398 end
399 end
```

addSingleAnimal

Description

Definition

addSingleAnimal()

Code

```
403 function HusbandryModuleAnimal:addSingleAnimal(animal,
    noEventSend)
404 if #self.animals >= self.maxNumAnimals then
405 return false
```

```
406 end
407
408 if not self:getSupportsSubType(animal:getSubType()) then
409 return false
410 end
411
412 table.insert(self.animals, animal)
413
414 local fillTypeIndex = animal:getFillTypeIndex()
415 if self.typedAnimals[fillTypeIndex] == nil then
416 self.typedAnimals[fillTypeIndex] = {}
417 end
418 table.insert(self.typedAnimals[fillTypeIndex], animal)
419
420 animal:setOwner(self.owner)
421
422 if self.owner.isServer and noEventSend == nil or not noEventSend
    then
423 g_server:broadcastEvent(AnimalAddEvent:new(self.owner, {animal}),
    nil, nil, nil, true)
424 end
425
426 g_messageCenter:publish(MessageType.HUSBANDRY_ANIMALS_CHANGED,
    self.owner)
427
428 self:updateAnimalParameters()
429
430 self.updateVisuals = true
431 self.owner:raiseActive()
432
433 if animal:isa(RideableAnimal) then
434 if not animal:getIsRideableSetupDone() then
435 if self.pendingRideables == nil then
436 self.pendingRideables = {}
437 end
438 table.insert(self.pendingRideables, animal)
439 self.owner:raiseActive()
440 end
441 end
442
443 return true
```

```
444 end
```

addPendingAnimal

Description

Definition

addPendingAnimal()

Code

```
448 function HusbandryModuleAnimal:addPendingAnimal(animalObjectId)
449 if self.animalsToAdd == nil then
450 self.animalsToAdd = {}
451 end
452 table.insert(self.animalsToAdd, animalObjectId)
453
454 self.owner:raiseActive()
455 end
```

removeAnimals

Description

Definition

removeAnimals()

Code

```
459 function HusbandryModuleAnimal:removeAnimals(animals,
    noEventSend)
460 local sendAnimals = {}
461
462 for _, animal in ipairs(animals) do
463 if self:removeSingleAnimal(animal, true) then
464 table.insert(sendAnimals, animal)
465 end
466 end
467
468 if self.owner.isServer and (noEventSend == nil or not
    noEventSend) then
469 g_server:broadcastEvent(AnimalRemoveEvent:new(self.owner,
    sendAnimals), nil, nil, nil, true)
470 end
471 end
```

removeSingleAnimal

Description

Definition

removeSingleAnimal()

Code

```
475 function HusbandryModuleAnimal:removeSingleAnimal(animal,
    noEventSend)
476 local found = false
```

```
477
478 for i, storedAnimal in ipairs(self.animals) do
479 if storedAnimal == animal then
480 if self.owner.isServer and (noEventSend == nil or not
    noEventSend) then
481 g_server:broadcastEvent(AnimalRemoveEvent:new(self.owner,
    {animal}), nil, nil, nil, true)
482 end
483
484 table.remove(self.animals, i)
485
486 local fillTypeIndex = animal:getFillTypeIndex()
487 table.remove(self.typedAnimals[fillTypeIndex], 1)
488
489 found = true
490 break
491 end
492 end
493
494 if found then
495 local visualId = animal:getVisualId()
496 if visualId ~= nil then
497 removeHusbandryAnimal(self.husbandryId, visualId)
498 self.visualIdToAnimal[visualId] = nil
499 animal:setVisualId(nil)
500
501 local fillType = animal:getFillTypeIndex()
502 for k, visual in ipairs(self.visualAnimals[fillType].visuals) do
503 if visual.id == visualId then
504 table.remove(self.visualAnimals[fillType].visuals, k)
505 break
506 end
507 end
508
509 if #self.visualAnimals[fillType].visuals == 0 then
510 self.visualAnimals[fillType] = nil
511 end
512 end
513
514 g_messageCenter:publish(MessageType.HUSBANDRY_ANIMALS_CHANGED,
    self.owner)
```

```
515
516  self:updateAnimalParameters()
517
518  self.updateVisuals = true
519  self.owner:raiseActive()
520
521  return true
522  end
523
524  return false
525  end
```

updateVisualAnimals

Description

Definition

updateVisualAnimals()

Code

```
529  function HusbandryModuleAnimal:updateVisualAnimals()
530
531  local numAnimals = #self.animals
532  local maxVisualAnimals = math.min(numAnimals,
     self.maxNumVisualAnimals)
533
534  local usableAnimals = {}
535
536  local percentages = {}
537  for _, animal in ipairs(self.animals) do
538  local fillType = animal:getFillTypeIndex()
539  if percentages[fillType] == nil then
540  percentages[fillType] = 0
541  usableAnimals[fillType] = {}
542  end
543  percentages[fillType] = percentages[fillType] + 1
544
545  if animal:getVisualId() == nil then
546  table.insert(usableAnimals[fillType], animal)
547  end
548  end
549
550  local instances = {}
551  local fillTypeToIntance = {}
552  local numInstances = 0
```

```
553 for fillType, num in pairs(percentages) do
554 local percentage = 0
555 if #self.animals > 0 then
556 percentage = num / numAnimals
557 end
558 local instance = {fillType=fillType, count=math.max(1,
    math.floor(maxVisualAnimals*percentage))}
559 table.insert(instances, instance)
560 fillTypeToIntance[fillType] = instance
561 numInstances = numInstances + instance.count
562 end
563
564 local function sort(a1, a2)
565 return a1.count < a2.count
566 end
567 table.sort(instances, sort)
568
569 -- if there's dif we reduce/increase each type by 1 until dif is
    0
570 local dif = maxVisualAnimals - numInstances
571 local i = 1
572 while dif ~= 0 do
573 local delta = MathUtil.sign(dif)
574 instances[i].count = instances[i].count + delta
575 dif = dif - delta
576
577 i = i + 1
578 if i > #instances then
579 i = 1
580 end
581 end
582
583 -- remove visual animals if not part of husbandry anymore
584 for k, visualAnimal in pairs(self.visualAnimals) do
585 if fillTypeToIntance[visualAnimal.fillType] == nil then
586 for i=1, #visualAnimal.visuals do
587 local visual = table.remove(visualAnimal.visuals, 1)
588 removeHusbandryAnimal(self.husbandryId, visual.id)
589 visual.animal:setVisualId(nil)
590 self.visualIdToAnimal[visual.id] = nil
591 end
```

```
592 self.visualAnimals[k] = nil
593 end
594 end
595
596 -- add or remove visual animals
597 for _, instance in ipairs(instances) do
598 local visualAnimal = self.visualAnimals[instance.fillType]
599 if visualAnimal == nil then
600 visualAnimal = {fillType=instance.fillType, visuals={}}
601 self.visualAnimals[instance.fillType] = visualAnimal
602 end
603
604 local dif = instance.count - #visualAnimal.visuals
605
606 if dif > 0 then
607 for i=1, dif do
608 local nextAnimal =
    table.remove(usableAnimals[visualAnimal.fillType], 1)
609 if nextAnimal == nil then
610 break
611 end
612
613 local subType = nextAnimal:getSubType()
614 local id = addHusbandryAnimal(self.husbandryId,
    subType.subTypeId-1)
615
616 if id ~= 0 then
617 setAnimalTextureTile(self.husbandryId, id,
    subType.texture.tileUIndex, subType.texture.tileVIndex)
618 self.visualIdToAnimal[id] = nextAnimal
619 nextAnimal:setVisualId(id)
620 table.insert(visualAnimal.visuals, {animal=nextAnimal, id=id})
621 end
622 end
623 elseif dif < 0 then
624 for j=1, math.abs(dif) do
625 local visual = table.remove(visualAnimal.visuals, 1)
626 if visual == nil then
627 break
628 end
629
```

```
630 removeHusbandryAnimal(self.husbandryId, visual.id)
631 visual.animal:setVisualId(nil)
632 self.visualIdToAnimal[visual.id] = nil
633 end
634 end
635 end
636
637 self:updateVisualDirt()
638 end
```

updateAnimalParameters

Description

Definition

updateAnimalParameters()

Code

```
642 function HusbandryModuleAnimal:updateAnimalParameters()
643
644 local averageWaterUsagePerDay = 0.0
645 local averageStrawUsagePerDay = 0.0
646 local averageFoodUsagePerDay = 0.0
647 local averageFoodSpillageProductionPerDay = 0.0
648 local averagePalletsProductionPerDay = 0.0
649 local averageManureProductionPerDay = 0.0
650 local averageLiquidManureProductionPerDay = 0.0
651 local averageMilkProductionPerDay = 0.0
652
653 for _, animal in ipairs(self.animals) do
654 local subType = animal:getSubType()
655
656 local input = subType.input
657 local output = subType.output
658 averageWaterUsagePerDay = averageWaterUsagePerDay +
    input.waterPerDay
659 averageStrawUsagePerDay = averageStrawUsagePerDay +
    input.strawPerDay
660 averageFoodUsagePerDay = averageFoodUsagePerDay +
    input.foodPerDay
661 averageFoodSpillageProductionPerDay =
    averageFoodSpillageProductionPerDay + output.foodSpillagePerDay
662 averagePalletsProductionPerDay = averagePalletsProductionPerDay +
    output.palletsPerDay
663 averageManureProductionPerDay = averageManureProductionPerDay +
    output.manurePerDay
```

```
664 averageLiquidManureProductionPerDay =
    averageLiquidManureProductionPerDay + output.liquidManurePerDay
665 averageMilkProductionPerDay = averageMilkProductionPerDay +
    output.milkPerDay
666 end
667
668 local numAnimals = #self.animals
669 averageWaterUsagePerDay = averageWaterUsagePerDay / numAnimals
670 averageStrawUsagePerDay = averageStrawUsagePerDay / numAnimals
671 averageFoodUsagePerDay = averageFoodUsagePerDay / numAnimals
672 averageFoodSpillageProductionPerDay =
    averageFoodSpillageProductionPerDay / numAnimals
673 averagePalletsProductionPerDay = averagePalletsProductionPerDay /
    numAnimals
674 averageManureProductionPerDay = averageManureProductionPerDay /
    numAnimals
675 averageLiquidManureProductionPerDay =
    averageLiquidManureProductionPerDay / numAnimals
676 averageMilkProductionPerDay = averageMilkProductionPerDay /
    numAnimals
677
678 local nbDays = 6.0
679 local maxNumAnimals = self.maxNumAnimals
680 local usageMultiplier = nbDays * maxNumAnimals
681 local waterCapacity = averageWaterUsagePerDay * usageMultiplier
682 local strawCapacity = averageStrawUsagePerDay * usageMultiplier
683 local foodCapacity = averageFoodUsagePerDay * usageMultiplier
684 local foodSpillageCapacity = averageFoodSpillageProductionPerDay
    * maxNumAnimals
685 local palletsCapacity = averagePalletsProductionPerDay *
    maxNumAnimals
686 local manureCapacity = averageManureProductionPerDay *
    maxNumAnimals
687 local liquidManureCapacity = averageLiquidManureProductionPerDay
    * maxNumAnimals
688 local milkCapacity = averageMilkProductionPerDay * maxNumAnimals
689
690 self.owner:setModuleParameters("water", waterCapacity,
    averageWaterUsagePerDay)
691 self.owner:setModuleParameters("straw", strawCapacity,
    averageStrawUsagePerDay)
692 self.owner:setModuleParameters("food", foodCapacity,
    averageFoodUsagePerDay)
```

```
693  self.owner:setModuleParameters("foodSpillage", foodSpillageCapacity, averageFoodSpillageProductionPerDay)
694  self.owner:setModuleParameters("pallets", palletsCapacity, averagePalletsProductionPerDay)
695  self.owner:setModuleParameters("manure", manureCapacity, averageManureProductionPerDay)
696  self.owner:setModuleParameters("liquidManure", liquidManureCapacity, averageLiquidManureProductionPerDay)
697  self.owner:setModuleParameters("milk", milkCapacity, averageMilkProductionPerDay)
698  end
```

addRideable

Description

Adds rideable vehicle in the husbandry

Definition

addRideable(integer husbandry)

Arguments

integer husbandry animal id

Code

```
703  function HusbandryModuleAnimal:addRideable(visualId, player)
704  local animal = self.visualIdToAnimal[visualId]
705  if animal ~= nil and animal:isa(RideableAnimal) then
706  if animal:getCanBeRidden() then
707  animal:activateRiding(player, false)
708  end
709  end
710  end
```

removeRideable

Description

Removes rideable vehicle in the husbandry

Definition

removeRideable(integer husbandry)

Arguments

integer husbandry animal id

Code

```
715  function HusbandryModuleAnimal:removeRideable(visualId)
716  local animal = self.visualIdToAnimal[visualId]
717  if animal ~= nil then
718  animal:deactivateRiding()
719  end
720  end
```

isRideableInOnHusbandryGround

Description

Definition

isRideableInOnHusbandryGround()

Return Values

bool true if animal is in the husbandry

Code

```
725  function HusbandryModuleAnimal:isRideableInOnHusbandryGround(visualId)
726  local animal = self.visualIdToAnimal[visualId]
727  if animal ~= nil and animal:isa(RideableAnimal) then
728  return animal:isOnHusbandyGround()
729  end
730  return false
731  end
```

getSupportsRiding

Description

Definition

getSupportsRiding()

Code

```
735  function HusbandryModuleAnimal:getSupportsRiding(visualId)
736  local animal = self.visualIdToAnimal[visualId]
737  return animal ~= nil and animal:isa(RideableAnimal)
738  end
```

getCanBeRidden

Description

Definition

getCanBeRidden()

Code

```
741  function HusbandryModuleAnimal:getCanBeRidden(visualId)
742  local animal = self.visualIdToAnimal[visualId]
743  if animal ~= nil and animal:isa(RideableAnimal) then
744  return animal:getCanBeRidden()
745  end
746
747  return false
748  end
```

hideAnimal

Description

Hide animal to husbandry

Definition

hideAnimal(integer visualId)

Arguments

integer visualId animal index in engine

Code

```
753  function HusbandryModuleAnimal:hideAnimal(visualId)
754  if visualId ~= nil and visualId ~= 0 then
755  hideAnimal(self.husbandryId, visualId)
756  end
757  end
```

showAnimal

Description

Show animal to husbandry

Definition

showAnimal(integer visualId)

Arguments

integer visualId animal index in engine

Code

```
762  function HusbandryModuleAnimal:showAnimal(visualId)
763  if visualId ~= nil and visualId ~= 0 then
764  showAnimal(self.husbandryId, visualId)
765  end
766  end
```

cleanAnimal

Description

Player cleans animal

Definition

cleanAnimal(integer visualId)

Arguments

integer visualId animal index in engine

Code

```
771  function HusbandryModuleAnimal:cleanAnimal(visualId, dt)
772  local animal = self.visualIdToAnimal[visualId]
773
774  if animal ~= nil and animal.clean ~= nil then
775  animal:clean(dt)
776  end
777
778  self:updateVisualDirt()
779  end
```

isAnimalDirty

Description

Definition

isAnimalDirty()

Code

```
783  function HusbandryModuleAnimal:isAnimalDirty(visualId)
784  local animal = self.visualIdToAnimal[visualId]
```

```
785
786  if animal ~= nil then
787  return animal:getDirtScale() > 0.0
788  end
789  return false
790  end
```

updateFitness

Description

Definition

updateFitness()

Code

```
794  function HusbandryModuleAnimal:updateFitness()
795  for _, animal in ipairs(self.animals) do
796  if animal ~= nil and animal:isa(Horse) then
797  return animal:updateFitness()
798  end
799  end
800  end
```

updateVisualDirt

Description

Definition

updateVisualDirt()

Code

```
804  function HusbandryModuleAnimal:updateVisualDirt()
805  for visualId, animal in pairs(self.visualIdToAnimal) do
806  local rough, _, tiling, x =
     getAnimalShaderParameter(self.husbandryId, visualId, "RDT")
807  setAnimalShaderParameter(self.husbandryId, visualId, "RDT",
     rough, animal:getDirtScale(), tiling, x)
808  end
809  end
```

updateBreeding

Description

Update reproduction information for the animals

Definition

updateBreeding(float dayToInterval)

Arguments

float dayToInterval

Code

```
814  function HusbandryModuleAnimal:updateBreeding(dayToInterval)
815  -- calculate reproduction rate ...
816
```

```
817 local totalAnimals = #self.animals
818 if self.owner.isServer then
819 for fillTypeIndex, animals in pairs(self.typedAnimals) do
820 local numAnimals = #animals
821 if numAnimals > 0 and totalAnimals < self.maxNumAnimals then
822 local subType = animals[1]:getSubType()
823 local birthRatePerDay = subType.breeding.birthRatePerDay
824 if birthRatePerDay > 0 then
825 local deltaTime = dayToInterval
826 local carryingCapacity = self.carryingCapacity
827 local birthIncrease = deltaTime * birthRatePerDay * ((carryingCapacity -
    carryingCapacity) * numAnimals
828
829 self.newAnimalPercentages[fillTypeIndex] = self.newAnimalPercentages[fil
    birthIncrease
830 if self.newAnimalPercentages[fillTypeIndex] > 1 then
831 local numNewAnimals = math.floor(self.newAnimalPercentages[fillTypeIndex
832
833 self.newAnimalPercentages[fillTypeIndex] = self.newAnimalPercentages[fil
    numNewAnimals
834 for i=1, numNewAnimals do
835 -- do not use local numAnimals because self.numAnimals gets updated when
    animal
836 if #self.animals >= self.maxNumAnimals then
837 break
838 end
839
840 local desc = g_animalManager:getAnimalsByType(subType.type)
841 local newAnimal = Animal.createFromFillType(self.owner.isServer, self.ow
    self.owner, fillTypeIndex)
842 newAnimal:register()
843 self:addSingleAnimal(newAnimal)
844
845 -- update stats
846 if desc.stats.breeding ~= "" then
847 g_currentMission:farmStats(self.owner:getOwnerFarmId()):updateStats(desc
    1)
848 end
849 end
850 end
851
852 -- values for stats
```

```
853     self.reproductionRatesPerDay[fillTypeIndex] = (self.owner:getGlobalProdu
        numAnimals * birthRatePerDay) * 0.5
854     else
855     self.reproductionRatesPerDay[fillTypeIndex] = 0
856     end
857     else
858     self.reproductionRatesPerDay[fillTypeIndex] = 0
859     end
860     end
861     end
862     end
```

getNumOfAnimals

Description

Gets the total number of animals

Definition

getNumOfAnimals(integer total)

Arguments

integer total number of animals

Code

```
867     function HusbandryModuleAnimal:getNumOfAnimals()
868     return #self.animals
869     end
```

getMaxNumOfAnimals

Description

Gets the maximum number of animals

Definition

getMaxNumOfAnimals(integer max)

Arguments

integer max number of animals

Code

```
874     function HusbandryModuleAnimal:getMaxNumOfAnimals()
875     return self.maxNumAnimals
876     end
```

getAnimalType

Description

Accessor to return the animal type

Definition

getAnimalType(string animalType)

Arguments

string animalType

Code

```
881     function HusbandryModuleAnimal:getAnimalType()
```

```
882  return self.animalType
883  end
```

getFillType

Description

Retrieves the filltype information of a subtype

Definition

getFillType()

Return Values

table

Code

```
897  function HusbandryModuleAnimal:getFillType(subTypeId)
898  local animals = g_animalManager:getAnimalsByType(self.animalType)
899  if animals ~= nil then
900  for _, animal in pairs(animals.subTypes) do
901  if animal.subTypeId == subTypeId then
902  local fillType =
     g_fillTypeManager:getFillTypeByIndex(animal.fillType)
903  if fillType ~= nil then
904  return fillType
905  end
906  end
907  end
908  end
909  return nil
910  end
```

getAnimals

Description

Definition

getAnimals()

Code

```
914  function HusbandryModuleAnimal:getAnimals()
915  return self.animals
916  end
```

getTypedAnimals

Description

Definition

getTypedAnimals()

Code

```
920  function HusbandryModuleAnimal:getTypedAnimals()
921  return self.typedAnimals
922  end
```

renameAnimal

Description
Definition

renameAnimal()

Code

```
926 function HusbandryModuleAnimal:renameAnimal(animalId, name, noEventSend)
927 if self.renamingTasks == nil then
928 self.renamingTasks = {}
929 end
930
931 table.insert(self.renamingTasks, {animalId=animalId, name=name})
932 self.owner:raiseActive()
933
934 AnimalNameEvent.sendEvent(self.owner, animalId, name, noEventSend)
935 end
```

getReproductionTimePerDay

Description
Definition

getReproductionTimePerDay()

Code

```
974 function HusbandryModuleAnimal:getReproductionTimePerDay(fillTypeIndex)
975 local reproductionRatePerDay = self.reproductionRatesPerDay[fillTypeIndex]
976 if reproductionRatePerDay == nil or reproductionRatePerDay == 0 then
977 return nil
978 end
979
980 local reproductionDuration = 1.0 / reproductionRatePerDay
981 local reproMins = reproductionDuration * 24 * 60
982
983 return reproMins
984 end
```

getAnimalName

Description
Definition

getAnimalName()

Code

```
988 function HusbandryModuleAnimal:getAnimalName(visualId)
989 local animal = self.visualIdToAnimal[visualId]
990 if animal ~= nil and animal:isa(Horse) then
```

```
991   return animal:getName()
992   end
993   return ""
994   end
```

HusbandryModuleBase

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
20   function HusbandryModuleBase:new(customMt)
21   if customMt == nil then
22   customMt = HusbandryModuleBase_mt
23   end
24
25   local self = {}
26   setmetatable(self, customMt)
27   self:initDataStructures()
28   self.owner = nil
29   return self
30   end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
34   function HusbandryModuleBase:initDataStructures()
35   self.fillLevels = {}
36   self.fillCapacity = 0.0
37   self.providedFillTypes = {}
38   self.singleAnimalUsagePerDay = 0.0
39   end
```

delete

Description

Deletes instance

Definition

delete()

Code

```
43  function HusbandryModuleBase:delete()
44  self.owner = nil
45  end
```

load

Description

Loads data from xml. Checking methods from husbandry used by modules are available.

Definition

load(table xmlFile, string xmlKey, table rootNode, table owner)

Arguments

table	xmlFile	handle
string	xmlKey	from which to read the configuration
table	rootNode	of the husbandry
table	owner	the husbandry

Return Values

boolean true if loading was successful else false

Code

```
54  function HusbandryModuleBase:load(xmlFile, configKey, rootNode,
    owner)
55  self.owner = owner
56  return true
57  end
```

finalizePlacement

Description

Called when husbandry is placed

Definition

finalizePlacement()

Return Values

bool true if successful

Code

```
62  function HusbandryModuleBase:finalizePlacement()
63  return true
64  end
```

onSell

Description

Called when husbandry is placed

Definition

onSell()

Return Values

bool true if successful

Code

```
69  function HusbandryModuleBase:onSell()
70  self.fillLevels = {}
```

```
71  self.fillCapacity = 0.0
72  self.providedFillTypes = {}
73  self.singleAnimalUsagePerDay = 0.0
74  end
```

onIntervalUpdate

Description

Updates module when a timed interval has passed

Definition

onIntervalUpdate(float dayToInterval)

Arguments

float dayToInterval

Code

```
79  function HusbandryModuleBase:onIntervalUpdate(dayToInterval)
80  end
```

readStream

Description

Reads network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
 96  function HusbandryModuleBase:readStream(streamId, connection)
 97  local nbFillLevel = streamReadUInt8(streamId)
 98
 99  for i=1, nbFillLevel do
100  local fillTypeIndex = streamReadUInt8(streamId)
101  local fillLevel = streamReadFloat32(streamId)
102  self.fillLevels[fillTypeIndex] = fillLevel
103  end
104  end
```

writeStream

Description

Writes network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
110  function HusbandryModuleBase:writeStream(streamId, connection)
```

```
111  local nbFillLevel = 0
112  for fillTypeIndex, fillLevel in pairs(self.fillLevels) do
113  nbFillLevel = nbFillLevel + 1
114  end
115
116  streamWriteUInt8(streamId, nbFillLevel)
117  for fillTypeIndex, fillLevel in pairs(self.fillLevels) do
118  streamWriteUInt8(streamId, fillTypeIndex)
119  streamWriteFloat32(streamId, fillLevel)
120  end
121  end
```

readUpdateStream

Description

Read updates from network stream

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId network stream identification
integer timestamp
table connection connection information

Code

```
128  function HusbandryModuleBase:readUpdateStream(streamId,
     timestamp, connection)
129  local nbFillLevel = streamReadUInt8(streamId)
130
131  for i=1, nbFillLevel do
132  local fillTypeIndex = streamReadUInt8(streamId)
133  local fillLevel = streamReadFloat32(streamId)
134  self.fillLevels[fillTypeIndex] = fillLevel
135  end
136  end
```

writeUpdateStream

Description

Write updates from network stream

Definition

writeUpdateStream(integer streamId, table connection, integer dirtyMask)

Arguments

integer streamId network stream identification
table connection connection information
integer dirtyMask is used to check if we need to update

Code

```
143 function HusbandryModuleBase:writeUpdateStream(streamId,
    connection, dirtyMask)
144 local nbFillLevel = 0
145 for fillTypeIndex, fillLevel in pairs(self.fillLevels) do
146 nbFillLevel = nbFillLevel + 1
147 end
148
149 streamWriteUInt8(streamId, nbFillLevel)
150 for fillTypeIndex, fillLevel in pairs(self.fillLevels) do
151 streamWriteUInt8(streamId, fillTypeIndex)
152 streamWriteFloat32(streamId, fillLevel)
153 end
154 end
```

loadFromXMLFile

Description

Loads information from attributes and node. Retrives from xml file information.

Definition

loadFromXMLFile(table xmlFile, string key)

Arguments

table	xmlFile XML file handler
string key	XML base key

Code

```
164 function HusbandryModuleBase:loadFromXMLFile(xmlFile, key)
165 self.fillCapacity = Utils.getNoNil(getXMLFloat(xmlFile,
    key.."#fillCapacity"), self.fillCapacity)
166
167 local i = 0
168 while true do
169 local fillLevelKey = key .. string.format(".fillLevel(%d)", i)
170
171 if not hasXMLProperty(xmlFile, fillLevelKey) then
172 break
173 end
174
175 local fillTypeName = getXMLString(xmlFile, fillLevelKey ..
    "#fillType")
176 local fillLevel = getXMLFloat(xmlFile, fillLevelKey ..
    "#fillLevel")
177 if fillTypeName ~= nil and fillLevel ~= nil then
178 local fillTypeIndex =
    g_fillTypeManager:getFillTypeIndexByName(fillTypeName)
179
```

```
180 if fillTypeIndex ~= nil then
181 self.fillLevels[fillTypeIndex] = fillLevel
182 self.providedFillTypes[fillTypeIndex] = true
183 end
184 end
185 i = i + 1
186 end
187 end
```

getIsFillTypeAllowed

Description

Check if fillType is allowed

Definition

getIsFillTypeAllowed(integer fillTypeIndex)

Arguments

integer fillTypeIndex index of the fillType to check

Code

```
217 function HusbandryModuleBase:getIsFillTypeAllowed(fillTypeIndex)
218 for fTypeIndex, state in pairs(self.providedFillTypes) do
219 if fTypeIndex == fillTypeIndex and state then
220 return true
221 end
222 end
223
224 return false
225 end
```

getIsToolTypeAllowed

Description

Checks if a tool type is allowed. (empty method)

Definition

getIsToolTypeAllowed(integer toolType)

Arguments

integer toolType

Code

```
230 function HusbandryModuleBase:getIsToolTypeAllowed(toolType)
231 return true
232 end
```

getHasSpaceForUnloading

Description

Checks if there is enough space to unload a fillType

Definition

getHasSpaceForUnloading(integer index)

Arguments

integer index of the fillType

Code

```
237 function HusbandryModuleBase:getHasSpaceForUnloading()
238 return self:getTotalFillLevel() <= self:getCapacity()
239 end
```

changeFillLevels

Description

Change amount of fillType

Definition

changeFillLevels(float amount, integer index)

Arguments

float amount to change the fill type

integer index of the fillType

Return Values

float amount effectively changed

Code

```
246 function HusbandryModuleBase:changeFillLevels(fillDelta,
    fillTypeIndex)
247 local delta = 0.0
248 if self.fillLevels[fillTypeIndex] ~= nil then
249 local oldFillLevel = self.fillLevels[fillTypeIndex]
250 local newFillLevel = oldFillLevel + fillDelta
251 newFillLevel = math.max(newFillLevel, 0.0)
252 delta = newFillLevel - oldFillLevel
253
254 local oldTotalFillLevel = self:getTotalFillLevel()
255 local capacity = self:getCapacity()
256 local newTotalFillLevel = oldTotalFillLevel + delta
257 newTotalFillLevel = MathUtil.clamp(newTotalFillLevel, 0.0,
    capacity)
258 delta = newTotalFillLevel - oldTotalFillLevel
259
260 self.fillLevels[fillTypeIndex] = self.fillLevels[fillTypeIndex] +
    delta
261 end
262 return delta
263 end
```

addFillLevelFromTool

Description

Called to change the amount of a specific fillType

Definition

addFillLevelFromTool(float deltaFillLevel, integer fillTypeIndex)

Arguments

float	deltaFillLevel amount to change
integer	fillTypeIndex index of the fillType

Return Values

float delta of amount changed

Code

```
270  function HusbandryModuleBase:addFillLevelFromTool(farmId, deltaFillLevel, fillTypeIndex)
271      if not self:getHasSpaceForUnloading() then
272          return 0
273      end
274
275      local changed = 0
276      changed = self:changeFillLevels(deltaFillLevel, fillTypeIndex)
277      return changed
278  end
```

getFillLevel

Description

Gets a fill level

Definition

getFillLevel(integer index)

Arguments

integer index of a fill type

Return Values

float returns the fillLevel of a fillType

Code

```
284  function HusbandryModuleBase:getFillLevel(fillTypeIndex)
285      return Utils.getNoNil(self.fillLevels[fillTypeIndex], 0.0)
286  end
```

getTotalFillLevel

Description

Gets total fill level

Definition

getTotalFillLevel()

Return Values

float returns the fillLevel of all fillType

Code

```
291  function HusbandryModuleBase:getTotalFillLevel()
292      local totalFillLevel = 0.0
293
294      for _, fillLevel in pairs(self.fillLevels) do
295          totalFillLevel = totalFillLevel + fillLevel
296      end
```

```
297  return totalFillLevel
298  end
```

setCapacity

Description

Sets a capacity

Definition

setCapacity(float newCapacity)

Arguments

float newCapacity

Code

```
303  function HusbandryModuleBase:setCapacity(newCapacity)
304  self.fillCapacity = newCapacity
305  end
```

getFreeCapacity

Description

Sets a capacity

Definition

getFreeCapacity()

Code

```
309  function HusbandryModuleBase:getFreeCapacity(fillTypeIndex)
310  return self:getCapacity() - self:getFillLevel(fillTypeIndex)
311  end
```

setSingleAnimalUsagePerDay

Description

Definition

setSingleAnimalUsagePerDay( )

Arguments

Code

```
316  function
     HusbandryModuleBase:setSingleAnimalUsagePerDay(usagePerDay)
317  self.singleAnimalUsagePerDay = usagePerDay
318  end
```

getCapacity

Description

Gets a fill capacity

Definition

getCapacity(integer index)

Arguments

integer index of a fill type

Return Values

float returns the capacity of a fillType

Code

```
324  function HusbandryModuleBase:getCapacity()
```

```
325 local capacity = self.fillCapacity
326 return capacity
327 end
```

getFillProgress

Description

Gets a fill progress

Definition

getFillProgress(integer index)

Arguments

integer index of a fill type

Return Values

float returns a progress of the fillType between 0 and 1. Default is 0.

Code

```
333 function HusbandryModuleBase:getFillProgress()
334 local capacity = self:getCapacity()
335 if capacity > 0.0 then
336 local progress = self:getTotalFillLevel() / capacity
337 progress = MathUtil.clamp(progress, 0.0, 1.0)
338 return progress
339 end
340 return 0.0
341 end
```

getProvidedFillTypes

Description

Definition

getProvidedFillTypes()

Code

```
345 function HusbandryModuleBase:getProvidedFillTypes()
346 return self.providedFillTypes
347 end
```

getAllFillLevels

Description

Definition

getAllFillLevels()

Code

```
351 function HusbandryModuleBase:getAllFillLevels()
352 local fillLevels = {}
353 local capacity = self:getCapacity()
354
355 for fillTypeIndex, fillLevel in pairs(self.fillLevels) do
356 fillLevels[fillTypeIndex] = fillLevel
357 end
```

```
358  return fillLevels, capacity
359  end
```

addFillLevelToFillableObject

Description

Increase fill level of a fillable object

Definition

addFillLevelToFillableObject(table fillableObject, integer fillTypeIndex, integer fillDelta, table fillInfo, toolType)

Arguments

table	fillableObject	fillable enity
integer	fillTypeIndex	fill type id
integer	fillDelta	amount to fill
table	fillInfo	fill information structure
toolType		

Return Values

float actual delta filled

Code

```
369  function HusbandryModuleBase:addFillLevelToFillableObject(fillableObject, fillUnitIndex, fillTypeIndex, fillDelta, fillInfo, toolType)
370  if fillableObject == nil or fillableObject == 0 or fillableObject == self then
371  return 0.0
372  end
373
374  local oldFillLevel = self:getFillLevel(fillTypeIndex)
375  fillDelta = math.min(fillDelta, oldFillLevel)
376
377  local actualDelta = fillableObject:addFillUnitFillLevel(self.owner:getOwnerFarmId(), fillUnitIndex, fillDelta, fillTypeIndex, ToolType.UNDEFINED, fillInfo)
378  self:changeFillLevels(-actualDelta, fillTypeIndex)
379  return actualDelta
380  end
```

getIsFillAllowedToFarm

Description

Definition

getIsFillAllowedToFarm(integer farmId)

Arguments

integer farmId

Return Values

Code

```
386  function HusbandryModuleBase:getIsFillAllowedToFarm(farmId)
```

```
387 return g_currentMission.accessHandler:canFarmAccess(farmId, self.owner)
388 end
```

registerModule

Description

Register a HUD extension for a specialization.

Definition

registerModule(string moduleName, table moduleType)

Arguments

string moduleName module name

table moduleType module class type table corresponding to the given module

Code

```
404 function HusbandryModuleBase.registerModule(moduleName, moduleType)
405 registry[moduleName] = moduleType
406 end
```

createModule

Description

Husbandry module factory method

Definition

createModule(table spec)

Arguments

table spec Specialization reference

Return Values

table module instance or nil of no module has been registered

Code

```
412 function HusbandryModuleBase.createModule(moduleName)
413 local moduleType = registry[moduleName]
414 local moduleInstance = nil
415 if moduleType ~= nil then
416 moduleInstance = moduleType.new()
417 end
418 return moduleInstance
419 end
```

hasModule

Description

Check if there is a module for a given specialization.

Definition

hasModule()

Code

```
423 function HusbandryModuleBase.hasModule(moduleName)
424 return registry[moduleName] ~= nil
```

```
425  end
```

HusbandryModuleFood

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
23  function HusbandryModuleFood:new(customMt)
24  local self = HusbandryModuleBase:new(customMt or
    HusbandryModuleFood_mt)
25
26  addConsoleCommand("gsHusbandryFillModuleFood", "Fills food
    module", "consoleHusbandryFillModuleFood", self)
27  self.foodGroupCapacities = {}
28  return self
29  end
```

delete

Description

Deletes instance

Definition

delete()

Code

```
33  function HusbandryModuleFood:delete()
34  if self.feedingTrough ~= nil then
35  self.feedingTrough:delete()
36  self.feedingTrough = nil
37  end
38  if self.fillPlane ~= nil then
39  self.fillPlane:delete()
40  self.fillPlane = nil
41  end
42
43  self.foodGroupCapacities = {}
44
45  removeConsoleCommand("gsHusbandryFillModuleFood")
46  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
50  function HusbandryModuleFood:initDataStructures()
51  HusbandryModuleFood:superClass().initDataStructures(self)
52  self.feedingTrough = nil
53  self.foodFactor = 0.0
54  self.consumedFood = {}
55  end
```

load

Description

Loads data from xml

Definition

load(table xmlFile, string xmlKey, table rootNode)

Arguments

table	xmlFile	handle
string	xmlKey	from which to read the configuration
table	rootNode	of the husbandry

Return Values

boolean true if loading was successful else false

Code

```
63  function HusbandryModuleFood:load(xmlFile, configKey, rootNode,
    owner)
64  if not HusbandryModuleFood:superClass().load(self, xmlFile,
    configKey, rootNode, owner) then
65  return false
66  end
67
68  if not hasXMLProperty(xmlFile, configKey) then
69  return false
70  end
71
72  local foodNodeId = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, configKey .. "#node"))
73  if foodNodeId ~= nil then
74  local feedingTrough = UnloadFeedingTrough:new(self.owner.isServer,
    self.owner.isClient)
75
76  if feedingTrough:load(foodNodeId, xmlFile, configKey, self) then
77  feedingTrough:register(true)
78  self.feedingTrough = feedingTrough
```

```
79  self:setupFoodGroups()
80
81  self.fillPlane = FillPlane:new()
82  self.fillPlane:load(rootNode, xmlFile, configKey..".fillPlane")
83
84  return true
85  else
86  feedingTrough:delete()
87  return false
88  end
89  end
90
91  return false
92  end
```

loadFromXMLFile

Description

Loads information from attributes and node. Retrives from xml file information.

Definition

loadFromXMLFile(table xmlFile, string key)

Arguments

table	xmlFile	XML file handler
string	key	XML base key

Code

```
98  function HusbandryModuleFood:loadFromXMLFile(xmlFile, key)
99  HusbandryModuleWater:superClass().loadFromXMLFile(self, xmlFile,
    key)
100
101 if self.fillPlane ~= nil then
102 self.fillPlane:setState(self:getFillProgress())
103 end
104 end
```

setupFoodGroups

Description

Definition

setupFoodGroups()

Code

```
108 function HusbandryModuleFood:setupFoodGroups()
109 if self.feedingTrough ~= nil then
110 local animalType = self.owner:getAnimalType()
111 local foodGroups =
    g_animalFoodManager:getFoodGroupByAnimalType(animalType)
```

```lua
local foodMixtures = g_animalFoodManager:getFoodMixturesByAnimalType(animalType)

if foodGroups ~= nil then
for _, foodGroup in pairs(foodGroups) do
for _, fillTypeIndex in pairs(foodGroup.fillTypes) do
self.fillLevels[fillTypeIndex] = 0.0
self.providedFillTypes[fillTypeIndex] = true
end
table.insert(self.foodGroupCapacities, {foodGroup=foodGroup,capacity=0.0})
end
if foodMixtures ~= nil then
for _, foodMixtureFillType in ipairs(foodMixtures) do
self.providedFillTypes[foodMixtureFillType] = true
end
end
self:setCapacity(0.0)
self.feedingTrough:initFillTypesFromFoodGroups(foodGroups)
else
print("Error: food group for animal type '" .. animalType .. "' not found")
end
end
end
```

setCapacity

Description

Sets a capacity

Definition

setCapacity(float newCapacity)

Arguments

float newCapacity

Code

```lua
function HusbandryModuleFood:setCapacity(newCapacity)
self.fillCapacity = 0.0
for _, foodGroupInfo in pairs(self.foodGroupCapacities) do
foodGroupInfo.capacity = newCapacity
end
end
```

getCapacity

Description

Gets a fill capacity

Definition

getCapacity(integer index)

Arguments

integer index of a fill type

Return Values

float returns the capacity of a fillType

Code

```
149 function HusbandryModuleFood:getCapacity()
150 local animalType = self.owner:getAnimalType()
151 local consumptionType = g_animalFoodManager:getFoodConsumptionTypeByAnimalType(animalType)
152 local foodCapacity = 0.0
153
154 if consumptionType == AnimalFoodManager.FOOD_CONSUME_TYPE_SERIAL then
155 for _, foodGroupInfo in pairs(self.foodGroupCapacities) do
156 foodCapacity = foodCapacity + foodGroupInfo.capacity
157 end
158 elseif consumptionType == AnimalFoodManager.FOOD_CONSUME_TYPE_PARALLEL then
159 for _, foodGroupInfo in pairs(self.foodGroupCapacities) do
160 foodCapacity = foodGroupInfo.capacity
161 break
162 end
163 else
164 foodCapacity = self.fillCapacity
165 end
166 return foodCapacity
167 end
```

getFreeCapacity

Description

Sets a capacity

Definition

getFreeCapacity()

Code

```
171 function HusbandryModuleFood:getFreeCapacity(fillTypeIndex)
172 local animalType = self.owner:getAnimalType()
173 local foodMixture = g_animalFoodManager:getFoodMixtureByFillType(fillTypeIndex)
174
175 if foodMixture ~= nil then
176 local checkedFoodGroups = {}
```

```
177 local capacity = 0
178 local fillLevel = 0
179 for _, ingredient in ipairs(foodMixture.ingredients) do
180 for _, fillType in ipairs(ingredient.fillTypes) do
181 local foodGroup =
    g_animalFoodManager:getFoodGroupByFillType(animalType, fillType)
182 if checkedFoodGroups[foodGroup] == nil then
183 for _, foodGroupInfo in pairs(self.foodGroupCapacities) do
184 if foodGroupInfo.foodGroup == foodGroup then
185 capacity = capacity + foodGroupInfo.capacity
186 break
187 end
188 end
189 fillLevel = fillLevel + self:getFillLevel(fillType)
190
191 checkedFoodGroups[foodGroup] = true
192 end
193 end
194 end
195
196 return capacity - fillLevel
197
198 else
199 local foodGroup =
    g_animalFoodManager:getFoodGroupByFillType(animalType,
    fillTypeIndex)
200 local capacity = 0.0
201 for _, foodGroupInfo in pairs(self.foodGroupCapacities) do
202 if foodGroupInfo.foodGroup == foodGroup then
203 capacity = foodGroupInfo.capacity
204 break
205 end
206 end
207
208 return capacity - self:getFillLevel(fillTypeIndex)
209 end
210 end
```

getFillLevel

Description

Definition

getFillLevel()

Code

```
214 function HusbandryModuleFood:getFillLevel(fillTypeIndex)
215 local animalType = self.owner:getAnimalType()
216 local fillLevel = g_animalFoodManager:getTotalFillLevelInGroupByFillTypeIndex(animalType, self.fillLevels, fillTypeIndex)
217 return Utils.getNoNil(fillLevel, 0.0)
218 end
```

getTotalFillLevel

Description

Definition

getTotalFillLevel()

Code

```
222 function HusbandryModuleFood:getTotalFillLevel()
223 local animalType = self.owner:getAnimalType()
224 local consumptionType = g_animalFoodManager:getFoodConsumptionTypeByAnimalType(animalType)
225 local totalFillLevel = 0.0
226
227 if consumptionType == AnimalFoodManager.FOOD_CONSUME_TYPE_SERIAL then
228 for _, foodGroupInfo in pairs(self.foodGroupCapacities) do
229 totalFillLevel = g_animalFoodManager:getTotalFillLevelInGroup(foodGroupInfo.foodGroup, self.fillLevels)
230 if totalFillLevel > 0.0 then
231 break
232 end
233 end
234 elseif consumptionType == AnimalFoodManager.FOOD_CONSUME_TYPE_PARALLEL and #self.foodGroupCapacities > 0 then
235 local nbGroups = #self.foodGroupCapacities
236 for _, foodGroupInfo in pairs(self.foodGroupCapacities) do
237 totalFillLevel = totalFillLevel + g_animalFoodManager:getTotalFillLevelInGroup(foodGroupInfo.foodGroup, self.fillLevels)
238 end
239 totalFillLevel = totalFillLevel / nbGroups
240 else
241 totalFillLevel = HusbandryModuleFood:superClass().getTotalFillLevel(self)
242 end
243 return totalFillLevel
244 end
```

onIntervalUpdate

Description

Update food usage

Definition

onIntervalUpdate(float dayToInterval)

Arguments

float dayToInterval

Code

```
249 function HusbandryModuleFood:onIntervalUpdate(dayToInterval)
250 HusbandryModuleFood:superClass().onIntervalUpdate(self,
    dayToInterval)
251
252 self.consumedFood = {}
253 -- print(string.format("-- [HusbandryModuleFood:onIntervalUpdate]
    CALLED"))
254 if self.singleAnimalUsagePerDay > 0.0 then
255 local totalNumAnimals = self.owner:getNumOfAnimals()
256 local foodNeeded = totalNumAnimals * self.singleAnimalUsagePerDay
    * dayToInterval
257 -- print(string.format("-- [HusbandryModuleFood:onIntervalUpdate]
    foodNeeded(%.3f) totalNumAnimals(%d)
    singleAnimalUsagePerDay(%.3f) dayToInterval(%.3f) ",
258 -- foodNeeded,
259 -- totalNumAnimals,
260 -- self.singleAnimalUsagePerDay,
261 -- dayToInterval))
262 if foodNeeded > 0.0 then
263 local animalType = self.owner:getAnimalType()
264 self.foodFactor = g_animalFoodManager:consumeFood(animalType,
    foodNeeded, self.fillLevels, self.consumedFood)
265
266 if self.fillPlane ~= nil then
267 self.fillPlane:setState(self:getFillProgress())
268 end
269 end
270 end
271 end
```

addFillLevelFromTool

Description

Definition

addFillLevelFromTool()

Code

```
275 function HusbandryModuleFood:addFillLevelFromTool(farmId, deltaFillLevel, fillTypeIndex)
276 local freeCapacity = self:getFreeCapacity(fillTypeIndex)
277 local changed = self:changeFillLevels(math.min(freeCapacity, deltaFillLevel), fillTypeIndex)
278
279 if deltaFillLevel > 0 and changed ~= 0 then
280 self.owner:updateGlobalProductionFactor()
281 end
282
283 if changed > 0 and self.feedingTrough ~= nil then
284 if self.feedingTrough:getIsFillTypeSupported(fillTypeIndex) then
285 self.feedingTrough:raiseActive()
286 if self.fillPlane ~= nil then
287 self.fillPlane:setState(self:getFillProgress())
288 self:updateFillPlaneColor()
289 end
290 end
291 end
292 return changed
293 end
```

changeFillLevels

Description

Change amount of fillType

Definition

changeFillLevels(float amount, integer index)

Arguments

float	amount to change the fill type
integer index	of the fillType

Return Values

float amount effectively changed

Code

```
300 function HusbandryModuleFood:changeFillLevels(fillDelta, fillTypeIndex)
301 local delta = 0.0
302 if self.fillLevels[fillTypeIndex] ~= nil then
303 local oldFillLevel = self.fillLevels[fillTypeIndex]
304 local newFillLevel = oldFillLevel + fillDelta
305 newFillLevel = math.max(newFillLevel, 0.0)
306 delta = newFillLevel - oldFillLevel
307
308 local oldTotalFillLevel = self:getFillLevel(fillTypeIndex)
```

```
309 local capacity = self:getCapacity()
310 local newTotalFillLevel = oldTotalFillLevel + delta
311 newTotalFillLevel = MathUtil.clamp(newTotalFillLevel, 0.0,
    capacity)
312 delta = newTotalFillLevel - oldTotalFillLevel
313
314 self.fillLevels[fillTypeIndex] = self.fillLevels[fillTypeIndex] +
    delta
315 end
316 return delta
317 end
```

getConsumedFood

Description

Definition

getConsumedFood()

Code

```
321 function HusbandryModuleFood:getConsumedFood()
322 return self.consumedFood
323 end
```

getFoodFactor

Description

Definition

getFoodFactor()

Code

```
327 function HusbandryModuleFood:getFoodFactor()
328 return self.foodFactor
329 end
```

consoleHusbandryFillModuleFood

Description

Definition

consoleHusbandryFillModuleFood()

Code

```
333 function
    HusbandryModuleFood:consoleHusbandryFillModuleFood(fillTypeName,
    value, length)
334 local usage = "gsHusbandryFillModuleFood fillTypeName value"
335 if fillTypeName == nil then
336 return "No filltype given. " .. usage
337 end
338 local fillTypeIndex =
    g_fillTypeManager:getFillTypeIndexByName(fillTypeName)
339 if fillTypeIndex == nil then
340 return "Invalid fillType. " .. usage
```

```
341 end
342 value = tonumber(value)
343 if value == nil then
344 return "No value given. " .. usage
345 end
346
347 self:addFillLevelFromTool(g_currentMission:getFarmId(), value,
    fillTypeIndex)
348
349 return "Filled ..."
350 end
```

updateFillPlaneColor

Description

Definition

updateFillPlaneColor()

Code

```
354 function HusbandryModuleFood:updateFillPlaneColor()
355 if self.fillPlane ~= nil and self.fillPlane.colorChange then
356 local colorScale = {0.0, 0.0, 0.0}
357
358 for filltypeIndex, state in pairs(self.feedingTrough.fillTypes)
    do
359 if state then
360 local fillType =
    g_fillTypeManager:getFillTypeByIndex(filltypeIndex)
361 local fillLevelRatio =
    self.feedingTrough.target:getFillLevel(filltypeIndex) /
    self.feedingTrough.target:getCapacity()
362 local fillPlaneColors = {1.0, 1.0, 1.0}
363
364 if fillType.fillPlaneColors ~= nil then
365 fillPlaneColors = fillType.fillPlaneColors
366 end
367 for i = 1, 3 do
368 colorScale[i] = MathUtil.clamp(colorScale[i] + fillLevelRatio *
    fillPlaneColors[i], 0.0, 1.0)
369 end
370 end
371 end
372 self.fillPlane:setColorScale(colorScale)
373 end
374 end
```

getFilltypeInfos

Description

Definition

getFilltypeInfos()

Code

```
function HusbandryModuleFood:getFilltypeInfos()
local result = {}

for _, foodGroupInfo in pairs(self.foodGroupCapacities) do
local foodGroup = foodGroupInfo.foodGroup
local totalFillLevel = g_animalFoodManager:getTotalFillLevelInGroup(foodGroupInfo.foodGroup, self.fillLevels)
local capacity = foodGroupInfo.capacity
table.insert(result, {foodGroup=foodGroup, fillLevel=totalFillLevel, capacity=capacity})
end
return result
end
```

HusbandryModuleLiquidManure

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
function HusbandryModuleLiquidManure:new(customMt)
return HusbandryModuleBase:new(customMt or HusbandryModuleLiquidManure_mt)
end
```

delete

Description

Deletes instance

Definition

delete()

Code

```
function HusbandryModuleLiquidManure:delete()
HusbandryModuleLiquidManure:superClass().delete(self)
if self.loadPlace ~= nil then
self.loadPlace:delete()
```

```
31  self.loadPlace = nil
32  end
33  if self.fillPlane ~= nil then
34  self.fillPlane:delete()
35  self.fillPlane = nil
36  end
37  end
```

load

Description

Loads data from xml

Definition

load(table xmlFile, string xmlKey, table rootNode)

Arguments

table	xmlFile	handle
string	xmlKey	from which to read the configuration
table	rootNode	of the husbandry

Return Values

boolean true if loading was successful else false

Code

```
45  function HusbandryModuleLiquidManure:load(xmlFile, configKey,
    rootNode, owner)
46  if not HusbandryModuleLiquidManure:superClass().load(self,
    xmlFile, configKey, rootNode, owner) then
47  return false
48  end
49
50  if not hasXMLProperty(xmlFile, configKey) then
51  return false
52  end
53
54  local liquidManureNodeId = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, configKey .. "#node"))
55  if liquidManureNodeId ~= nil then
56  local loadPlace = LoadTrigger:new(self.owner.isServer,
    self.owner.isClient)
57
58
59  if loadPlace:load(liquidManureNodeId, xmlFile, configKey) then
60  loadPlace:setSource(self)
61  loadPlace:register(true)
62
63  self.loadPlace = loadPlace
```

```
64  self.fillPlane = FillPlane:new()
65  self.fillPlane:load(rootNode, xmlFile, configKey..".fillPlane")
66
67  for fillTypeIndex, state in pairs(loadPlace.fillTypes) do
68  self.fillLevels[fillTypeIndex] = 0.0
69  self.providedFillTypes[fillTypeIndex] = state
70  self:setCapacity(0.0)
71  end
72
73  return true
74  else
75  loadPlace:delete()
76  end
77  end
78
79  return false
80  end
```

onIntervalUpdate

Description

Update water usage

Definition

onIntervalUpdate(float dayToInterval)

Arguments

float dayToInterval

Code

```
85  function HusbandryModuleLiquidManure:onIntervalUpdate(dayToInterval)
86  HusbandryModuleLiquidManure:superClass().onIntervalUpdate(self, dayToInterval)
87
88  if self.singleAnimalUsagePerDay > 0 then
89  local totalNumAnimals = self.owner:getNumOfAnimals()
90  local hasWater = self.owner:hasWater()
91
92  if hasWater then
93  local newLiquidManure = totalNumAnimals * self.singleAnimalUsagePerDay * dayToInterval
94
95  if newLiquidManure > 0 then
96  for fillTypeIndex, _ in pairs(self.fillLevels) do
97  self:changeFillLevels(newLiquidManure, fillTypeIndex)
```

```
 98  end
 99  if self.fillPlane ~= nil then
100  self.fillPlane:setState(self:getFillProgress())
101  end
102  end
103  end
104  end
105  end
```

getFilltypeInfos

Description

Definition

getFilltypeInfos()

Code

```
109  function HusbandryModuleLiquidManure:getFilltypeInfos()
110  local result = {}
111
112  for fillTypeIndex, _ in pairs(self.loadPlace.fillTypes) do
113  local fillType =
     g_fillTypeManager:getFillTypeByIndex(fillTypeIndex)
114  local capacity = self:getCapacity()
115  local fillLevel = self:getFillLevel(fillTypeIndex)
116
117  table.insert(result, {fillType=fillType, fillLevel=fillLevel,
     capacity=capacity})
118  end
119  return result
120  end
```

HusbandryModuleManure

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
24  function HusbandryModuleManure:new(customMt)
25  if customMt == nil then
26  customMt = HusbandryModuleManure_mt
27  end
28  local self = HusbandryModuleBase:new(customMt)
```

```
29  return self
30  end
```

delete

Description

Deletes instance

Definition

delete()

Code

```
34  function HusbandryModuleManure:delete()
35  -- unregistration in mission
36  g_currentMission:removeManureHeap(self.manureHeapName)
37
38  g_densityMapHeightManager:removeFixedFillTypesArea(self.manureArea)
39  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
43  function HusbandryModuleManure:initDataStructures()
44  HusbandryModuleManure:superClass().initDataStructures(self)
45
46  self.manureArea = nil
47  self.fillTypeIndex = 0
48  self.fillLevel = 0
49  self.manureToDrop = 0
50  self.manureToRemove = 0
51  self.lineOffsetManure = 0
52  self.manureHeapName = ""
53  end
```

load

Description

Loads data from xml

Definition

load(table xmlFile, string xmlKey, table rootNode)

Arguments

table	xmlFile	handle
string	xmlKey	from which to read the configuration
table	rootNode	of the husbandry

Return Values

boolean true if loading was successful else false

Code

```
61  function HusbandryModuleManure:load(xmlFile, configKey, rootNode,
    owner)
62  local result = HusbandryModuleManure:superClass().load(self,
    xmlFile, configKey, rootNode, owner)
63
64  if result ~= true then
65  return false
66  end
67  if not hasXMLProperty(xmlFile, configKey) then
68  return false
69  end
70
71  local manureAreaNodeStart = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, configKey .. "#startNode"))
72  local manureAreaNodeWidth = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, configKey .. "#widthNode"))
73  local manureAreaNodeHeight = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, configKey .. "#heightNode"))
74  local fillType = getXMLString(xmlFile, configKey .. "#fillType")
75  if manureAreaNodeStart ~= nil and manureAreaNodeWidth ~= nil and
    manureAreaNodeHeight ~= nil and fillType ~= nil then
76  self.fillTypeIndex =
    g_fillTypeManager:getFillTypeIndexByName(fillType)
77  self.manureArea = {start=manureAreaNodeStart,
    width=manureAreaNodeWidth, height=manureAreaNodeHeight}
78
79  local fillTypes = {}
80  fillTypes[self.fillTypeIndex] = true
81  g_densityMapHeightManager:setFixedFillTypesArea(self.manureArea,
    fillTypes)
82  else
83  result = false
84  end
85  return result
86  end
```

onIntervalUpdate

Description

Update water usage

Definition

onIntervalUpdate(float dayToInterval)

Arguments

float dayToInterval

Code

```
100 function HusbandryModuleManure:onIntervalUpdate(dayToInterval)
101 HusbandryModuleManure:superClass().onIntervalUpdate(self, dayToInterval)
102
103 if self.singleAnimalUsagePerDay > 0 then
104 local hasStraw = self.owner:hasStraw()
105
106 if hasStraw then
107 if self.manureToDrop > g_densityMapHeightManager:getMinValidLiterValue(self.fillTypeIndex) then
108 local maxManureToDrop = math.min(self.manureToDrop, 20.0 * g_densityMapHeightManager:getMinValidLiterValue(self.fillTypeIndex))
109 local dropped = self:updateManure(maxManureToDrop)
110
111 self.manureToDrop = self.manureToDrop - dropped
112 end
113
114 local totalNumAnimals = self.owner:getNumOfAnimals()
115 local newManure = totalNumAnimals * self.singleAnimalUsagePerDay * dayToInterval
116
117 self.manureToDrop = self.manureToDrop + newManure
118
119 local manureLevel = self:getManureLevel()
120 self.fillLevel = manureLevel + self.manureToDrop
121 end
122 end
123 end
```

loadFromXMLFile

Description

Loads information from attributes and node. Retrives from xml file information.

Definition

loadFromXMLFile(table xmlFile, string key)

Arguments

table xmlFile XML file handler

string key XML base key

Code

```
129 function HusbandryModuleManure:loadFromXMLFile(xmlFile, key)
130 HusbandryModuleManure:superClass().loadFromXMLFile(self, xmlFile, key)
```

```
131
132 self.fillLevel = Utils.getNoNil(getXMLFloat(xmlFile,
    key.."#fillLevel"), self.fillLevel)
133 self.manureToDrop = Utils.getNoNil(getXMLFloat(xmlFile,
    key.."#manureToDrop"), self.manureToDrop)
134 self.manureToRemove = Utils.getNoNil(getXMLFloat(xmlFile,
    key.."#manureToRemove"), self.manureToRemove)
135 end
```

updateManure

Description

Update manure mechanics

Definition

updateManure(float manureIncrease)

Arguments

float manureIncrease

Return Values

float manure dropped

Code

```
149 function HusbandryModuleManure:updateManure(manureIncrease)
150 local manureDropped = 0
151
152 if manureIncrease >
    g_densityMapHeightManager:getMinValidLiterValue(self.fillTypeIndex)
    then
153 local xs, _, zs = getWorldTranslation(self.manureArea.start)
154 local xw, _, zw = getWorldTranslation(self.manureArea.width)
155 local xh, _, zh = getWorldTranslation(self.manureArea.height)
156 local ux, uz = xw - xs, zw - zs
157 local vx, vz = xh - xs, zh - zs
158
159 --local uLength = MathUtil.vector2Length(ux, uz)
160 local vLength = MathUtil.vector2Length(vx, vz)
161
162 -- drop centered or along line from start to width
163 local sx = xs + 0.40 * ux + 0.5 * vx -- ( ux / uLength ) + ( vx /
    vLength )
164 local sz = zs + 0.40 * uz + 0.5 * vz -- ( uz / uLength ) + ( vz /
    vLength )
165 local sy =
    getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, sx, 0,
    sz) + 10.0
166
```

```
167 local ex = xs + 0.60 * ux + 0.5 * vx -- xw - ( ux / uLength ) + ( vx / vLength )
168 local ez = zs + 0.60 * uz + 0.5 * vz -- zw - ( uz / uLength ) + ( vz / vLength )
169 local ey = sy -- getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, ex, 0, ez) + 10
170
171 local dropped, lineOffset = DensityMapHeightUtil.tipToGroundAroundLine(nil, manureIncrease, self.fillTypeIndex, sx, sy, sz, ex, ey, ez, 0, vLength, self.lineOffsetManure, false, nil)
172
173 manureDropped = dropped
174 self.lineOffsetManure = lineOffset
175 end
176 return manureDropped
177 end
```

getManureLevel

Description

Get manure level from manure area

Definition

getManureLevel()

Return Values

float manure fill level

Code

```
182 function HusbandryModuleManure:getManureLevel()
183 local xs, _, zs = getWorldTranslation(self.manureArea.start)
184 local xw, _, zw = getWorldTranslation(self.manureArea.width)
185 local xh, _, zh = getWorldTranslation(self.manureArea.height)
186 local fillLevel = DensityMapHeightUtil.getFillLevelAtArea(self.fillTypeIndex, xs, zs, xw, zw, xh, zh)
187 return fillLevel
188 end
```

removeManure

Description

Remove amount of manure; called by Sprayer specialization

Definition

removeManure(float delta)

Arguments

float delta amount to remove

Return Values

float manure used

Code

```
194 function HusbandryModuleManure:removeManure(delta)
195 local used = 0
196
197 if self.manureToDrop >= delta then
198 self.manureToDrop = self.manureToDrop - delta
199 used = delta
200 else
201 self.manureToDrop = self.manureToDrop - delta
202 delta = math.abs(self.manureToDrop)
203 self.manureToDrop = 0
204 self.manureToRemove = self.manureToRemove + delta
205
206 local manureLevel = self:getManureLevel()
207 self.fillLevel = manureLevel + self.manureToDrop
208
209 if self.manureToRemove < manureLevel then
210 used = delta
211 if self.manureToRemove >
    g_densityMapHeightManager:getMinValidLiterValue(self.fillTypeIndex)
    then
212 local xs, _, zs = getWorldTranslation(self.manureArea.start)
213 local xw, _, zw = getWorldTranslation(self.manureArea.width)
214 local xh, _, zh = getWorldTranslation(self.manureArea.height)
215 local ux, uz = xw - xs, zw - zs
216 local vx, vz = xh - xs, zh - zs
217 local radius = MathUtil.vector2Length(xw - xh, zw - zh) * 0.5
218 local sx = xs + 0.5 * ux + 0.5 * vx
219 local sz = zs + 0.5 * uz + 0.5 * vz
220 local sy =
    getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode,
    sx,0,sz) + 10
221 local ex = xs + 0.5 * ux + 0.5 * vx
222 local ez = zs + 0.5 * uz + 0.5 * vz
223 local ey = sz
224 local dropped, _ = DensityMapHeightUtil.tipToGroundAroundLine(nil,
    -self.manureToRemove, self.fillTypeIndex, sx, sy, sz, ex, ey, ez,
    radius, radius, nil, false, nil)
225
226 self.manureToRemove = math.max(self.manureToRemove + dropped, 0)
227 if dropped == 0 then
228 used = 0
```

```
229 end
230 end
231 end
232 end
233 return used
234 end
```

getFilltypeInfos

Description

Definition

getFilltypeInfos()

Code

```
238 function HusbandryModuleManure:getFilltypeInfos()
239 local result = {}
240 local fillType = g_fillTypeManager:getFillTypeByIndex(self.fillTypeIndex)
241 local capacity = 0.0
242 local fillLevel = self.fillLevel
243
244 table.insert(result, {fillType=fillType, fillLevel=fillLevel, capacity=capacity})
245 return result
246 end
```

HusbandryModuleMilk

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
22 function HusbandryModuleMilk:new(customMt)
23 if customMt == nil then
24 customMt = HusbandryModuleMilk_mt
25 end
26 local self = HusbandryModuleBase:new(customMt)
27 return self
28 end
```

delete

Description

Deletes instance

Definition

delete()

Code

```
32  function HusbandryModuleMilk:delete()
33  HusbandryModuleMilk:superClass().delete(self)
34  if self.loadPlace ~= nil then
35  self.loadPlace:delete()
36  self.loadPlace = nil
37  end
38  end
```

load

Description

Loads data from xml

Definition

load(table xmlFile, string xmlKey, table rootNode)

Arguments

table	xmlFile	handle
string	xmlKey	from which to read the configuration
table	rootNode	of the husbandry

Return Values

boolean true if loading was successful else false

Code

```
46  function HusbandryModuleMilk:load(xmlFile, configKey, rootNode,
    owner)
47  local result = HusbandryModuleMilk:superClass().load(self,
    xmlFile, configKey, rootNode, owner)
48
49  if result ~= true then
50  return false
51  end
52  if not hasXMLProperty(xmlFile, configKey) then
53  return false
54  end
55
56  local milkTankNodeId = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, configKey .. "#node"))
57  if milkTankNodeId ~= nil then
58  local loadPlace = LoadTrigger:new(self.owner.isServer,
    self.owner.isClient)
59
60
61  if loadPlace:load(milkTankNodeId, xmlFile, configKey) then
62  loadPlace:setSource(self)
```

```
63  loadPlace:register(true)
64
65  self.loadPlace = loadPlace
66
67  for fillTypeIndex, state in pairs(loadPlace.fillTypes) do
68  self.fillLevels[fillTypeIndex] = 0.0
69  self.providedFillTypes[fillTypeIndex] = state
70  self:setCapacity(0.0)
71  end
72  return true
73  else
74  loadPlace:delete()
75  end
76  end
77  return false
78  end
```

onIntervalUpdate

Description

Update milk usage

Definition

onIntervalUpdate(float dayToInterval)

Arguments

float dayToInterval

Code

```
83  function HusbandryModuleMilk:onIntervalUpdate(dayToInterval)
84  HusbandryModuleMilk:superClass().onIntervalUpdate(self,
    dayToInterval)
85
86  if self.singleAnimalUsagePerDay > 0.0 then
87  local hasWater = self.owner:hasWater()
88
89  if hasWater then
90  local totalNumAnimals = self.owner:getNumOfAnimals()
91  local newMilk = self.owner:getGlobalProductionFactor() *
    totalNumAnimals * self.singleAnimalUsagePerDay * dayToInterval
92
93  if newMilk > 0 then
94  for fillTypeIndex, state in pairs(self.loadPlace.fillTypes) do
95  self:changeFillLevels(newMilk, fillTypeIndex)
96  end
97  end
```

```
 98  end
 99  end
100  end
```

getFilltypeInfos

Description

Definition

getFilltypeInfos()

Code

```
104  function HusbandryModuleMilk:getFilltypeInfos()
105  local result = {}
106
107  for fillTypeIndex, _ in pairs(self.loadPlace.fillTypes) do
108  local fillType = g_fillTypeManager:getFillTypeByIndex(fillTypeIndex)
109  local capacity = self:getCapacity()
110  local fillLevel = self:getFillLevel(fillTypeIndex)
111
112  table.insert(result, {fillType=fillType, fillLevel=fillLevel, capacity=capacity})
113  end
114  return result
115  end
```

HusbandryModulePallets

Description

load

Description

Loads data from xml

Definition

load(table xmlFile, string xmlKey, table rootNode)

Arguments

table	xmlFile	handle
string	xmlKey	from which to read the configuration
table	rootNode	of the husbandry

Return Values

boolean true if loading was successful else false

Code

```
54  function HusbandryModulePallets:load(xmlFile, configKey, rootNode, owner)
55  local result = HusbandryModulePallets:superClass().load(self, xmlFile, configKey, rootNode, owner)
56
57  if result ~= true then
58  return false
```

```
59  end
60  if not hasXMLProperty(xmlFile, configKey) then
61  return false
62  end
63
64  self.palletSpawnerStartNode = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, configKey .. "#startNode"))
65  self.palletSpawnerWidthNode = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, configKey .. "#widthNode"))
66  self.palletSpawnerHeightNode = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, configKey .. "#heightNode"))
67
68  local xw, yw, zw = getTranslation(self.palletSpawnerWidthNode)
69  if xw == 0.0 or math.abs(yw) > 0.001 or math.abs(zw) > 0.001 then
70  g_logManager:devWarning(string.format("Warning: width node of
    husbandry module pallets has incorrect parameters x(%.3f),
    y(%.3f), z(%.3f).", xw, yw, zw))
71  return false
72  end
73  local xh, yh, zh = getTranslation(self.palletSpawnerHeightNode)
74  if math.abs(xh) > 0.001 or math.abs(yh) > 0.001 or zh == 0.0 then
75  g_logManager:devWarning(string.format("Warning: height node of
    husbandry module pallets has incorrect parameters x(%.3f),
    y(%.3f), z(%.3f).", xh, yh, zh))
76  return false
77  end
78
79  self.palletSpawnerNode = self.palletSpawnerStartNode
80  self.palletSpawnerAreaSizeX = xw
81  self.palletSpawnerAreaSizeZ = zh
82
83  local fillTypeStr = getXMLString(xmlFile, configKey ..
    "#fillType")
84  self.palletFillTypeIndex =
    g_fillTypeManager:getFillTypeIndexByName(fillTypeStr)
85  self.palletConfigFilename = Utils.getNoNil(getXMLString(xmlFile,
    configKey .. "#filename"), "")
86  self.palletConfigFilename =
    Utils.getFilename(self.palletConfigFilename, self.baseDirectory)
87  self.palletSize = Utils.getNoNil(getXMLFloat(xmlFile, configKey ..
    "#size"), 2.0)
88  self.palletSize_half = 0.5 * self.palletSize
```

```
89  self.palletFillUnitIndex = getXMLFloat(xmlFile, configKey ..
    "#fillUnitIndex")
90  self.palletSpawnerFillDelta = 0
91  self.numObjectsInPalletSpawnerTrigger = 0
92  return self.palletSpawnerNode ~= nil and self.palletFillTypeIndex
    ~= nil and self.palletConfigFilename ~= "" and
    self.palletFillUnitIndex ~= nil
93  end
```

onIntervalUpdate

Description

Update pallets production

Definition

onIntervalUpdate(float dayToInterval)

Arguments

float dayToInterval

Code

```
98  function HusbandryModulePallets:onIntervalUpdate(dayToInterval)
99  HusbandryModulePallets:superClass().onIntervalUpdate(self, dayToInterval
100 local totalNumAnimals = self.owner:getNumOfAnimals()
101
102 if self.singleAnimalUsagePerDay > 0 and totalNumAnimals > 0 then
103 local productivity = self.owner:getGlobalProductionFactor()
104 local fillDelta = productivity * totalNumAnimals * self.singleAnimalUsag
    dayToInterval
105
106 self.palletSpawnerFillDelta = self.palletSpawnerFillDelta + fillDelta
107 if self.palletSpawnerFillDelta > 0 then
108 -- check if last pallet is still valid
109 if self.currentPallet ~= nil then
110 local fillUnitIndex = self.palletFillUnitIndex
111 if self.currentPallet:getFillUnitFreeCapacity(fillUnitIndex) == 0.0 then
112 self.currentPallet = nil
113 end
114 end
115 if self.currentPallet ~= nil then
116 if not entityExists(self.currentPallet.rootNode) then
117 self.currentPallet = nil
118 else
119 local x, _, z = localToLocal(self.currentPallet.rootNode, self.palletSpa
    0,0,0)
120 if x < 0 or z < 0 or x > self.palletSpawnerAreaSizeX or z >
    self.palletSpawnerAreaSizeZ then
```

```
121 self.currentPallet = nil
122 end
123 end
124 end
125 -- check if there is a pallet which can be filled
126 if self.currentPallet == nil then
127 self.availablePallet = nil
128 local x,y,z = localToWorld(self.palletSpawnerNode, 0.5 *
    self.palletSpawnerAreaSizeX, 0, 0.5 * self.palletSpawnerAreaSizeZ)
129 local rx,ry,rz = getWorldRotation(self.palletSpawnerNode)
130 local nbShapesOverlap = overlapBox(x, y - 5, z, rx, ry, rz, 0.5 *
    self.palletSpawnerAreaSizeX, 10, 0.5 * self.palletSpawnerAreaSizeZ,
    "palletSpawnerCollisionTestCallback", self, nil, true, false, true)
131 if self.availablePallet ~= nil then
132 self.currentPallet = self.availablePallet
133 end
134 end
135 if self.currentPallet == nil then
136 local rx, ry, rz = getWorldRotation(self.palletSpawnerNode)
137 local x, y, z = getWorldTranslation(self.palletSpawnerNode)
138 local canCreatePallet = false
139
140 for dx = self.palletSize_half, self.palletSpawnerAreaSizeX - self.pallet
    self.palletSize do
141 for dz = self.palletSize_half, self.palletSpawnerAreaSizeZ - self.pallet
    self.palletSize do
142 x, y, z = localToWorld(self.palletSpawnerNode, dx, 0, dz)
143 self.palletSpawnerCollisionObjectId = 0
144 local nbShapesOverlap = overlapBox(x, y - 5, z, rx, ry, rz, self.palletS
    10.0, self.palletSize_half, "palletSpawnerCollisionTestCallback", self,
    false, true)
145 if self.palletSpawnerCollisionObjectId == 0 then
146 canCreatePallet = true
147 break
148 end
149 end
150 if canCreatePallet then
151 break
152 end
153 end
154 local thresholdForDeletion = 1.0
155 if canCreatePallet and self.palletSpawnerFillDelta > thresholdForDeletio
```

```
156 self.currentPallet = g_currentMission:loadVehicle(self.palletConfigFiler
    nil, z, 1.2, ry, true, 0, Vehicle.PROPERTY_STATE_OWNED,
    self.owner:getOwnerFarmId(), nil, nil)
157 table.insert(self.pallets, self.currentPallet)
158 self.currentPallet:addDeleteListener(self)
159 end
160 end
161 if self.currentPallet ~= nil then
162 local fillUnitIndex = self.palletFillUnitIndex
163
164 self.palletSpawnerFillDelta =
    self.currentPallet:addFillUnitFillLevel(self.owner:getOwnerFarmId(),
    fillUnitIndex, self.palletSpawnerFillDelta, self.palletFillTypeIndex,
    ToolType.UNDEFINED)
165 else
166 -- @todo: replace with new system
167 -- g_server:broadcastEvent(AnimalHusbandryNoMorePalletSpaceEvent:new(sel
168 --
169 -- local fillType = g_fillTypeManager:getFillTypeByIndex(self.palletFill
170 -- if fillType ~= nil then
171 -- local animalTypeString = g_i18n:getText( "ui_statisticView_" ..
    tostring(self.owner:getAnimalType()),
    g_currentMission.missionInfo.customEnvironment)
172 -- local text =
    string.format(g_i18n:getText("ingameNotification_palletSpawnerBlocked"),
    fillType.title, animalTypeString )
173 --
    g_currentMission:addIngameNotification(FSBaseMission.INGAME_NOTIFICATION
    text)
174 -- end
175 end
176 end
177 end
178 end
```

loadFromXMLFile

Description

Loads information from attributes and node. Retrives from xml file information.

Definition

loadFromXMLFile(table xmlFile, string key)

Arguments

table	xmlFile	XML file handler
string	key	XML base key

Code

```
184 function HusbandryModulePallets:loadFromXMLFile(xmlFile, key)
```

```
185 HusbandryModulePallets:superClass().loadFromXMLFile(self,
    xmlFile, key)
186
187 self.palletSpawnerFillDelta = Utils.getNoNil(getXMLFloat(xmlFile,
    key.."#palletFillDelta"), self.palletSpawnerFillDelta)
188 end
```

palletSpawnerCollisionTestCallback

Description

Test that object colliding with pallet is a vehicle handling pallets and is not full

Definition

palletSpawnerCollisionTestCallback(integer transformId)

Arguments

integer transformId scenegraph id from colliding object

Return Values

bool is true if available Pallet has been assigned with a vehicle object

Code

```
200 function
    HusbandryModulePallets:palletSpawnerCollisionTestCallback(transformId)
201 if transformId ~= g_currentMission.terrainRootNode then
202 if transformId ~= self.palletSpawnerNode then
203 self.palletSpawnerCollisionObjectId = transformId
204 local object = g_currentMission:getNodeObject(transformId)
205 if object ~= nil and object.isa ~= nil and object:isa(Vehicle) and
    object.typeName == "pallet" then
206 local fillUnitObject = object.spec_fillUnit
207
208 if fillUnitObject ~= nil then
209 local fillUnits = fillUnitObject:getFillUnits()
210
211 for fillUnitIndex, fillUnit in pairs(fillUnits) do
212 if fillUnitIndex == self.palletFillTypeIndex then
213 if fillUnitObject:getFillUnitFillLevel(fillUnitIndex) <
    fillUnitObject:getFillUnitCapacity(fillUnitIndex) then
214 local x, _, z = localToLocal(fillUnitObject.rootNode,
    self.palletSpawnerNode, 0, 0, 0)
215
216 if not ((x < 0) or (z < 0) or (x > self.palletSpawnerAreaSizeX) or (z
    > self.palletSpawnerAreaSizeZ)) then
217 self.availablePallet = fillUnitObject
218 end
219 end
220 end
221 end
```

```
222 end
223 end
224 end
225 end
226 return self.availablePallet == nil
227 end
```

onDeleteObject

Description

Definition

onDeleteObject()

Code

```
231 function HusbandryModulePallets:onDeleteObject(object)
232 for i, pallet in pairs(self.pallets) do
233 if pallet == object then
234 table.remove(self.pallets, i)
235 break
236 end
237 end
238 end
```

getFilltypeInfos

Description

Definition

getFilltypeInfos()

Code

```
242 function HusbandryModulePallets:getFilltypeInfos()
243 local result = {}
244 local totalCapacity = 0.0
245 local totalFillLevel = 0.0
246 local fillType =
    g_fillTypeManager:getFillTypeByIndex(self.palletFillTypeIndex)
247 for i, pallet in pairs(self.pallets) do
248 local fillUnitObject = pallet.spec_fillUnit
249
250 if fillUnitObject ~= nil then
251 local fillUnits = fillUnitObject:getFillUnits()
252
253 for fillUnitIndex, fillUnit in pairs(fillUnits) do
254 local capacity =
    fillUnitObject:getFillUnitFillLevel(fillUnitIndex)
255 local fillLevel =
    fillUnitObject:getFillUnitCapacity(fillUnitIndex)
256
```

```
257 totalCapacity = totalCapacity + capacity
258 totalFillLevel = totalFillLevel + fillLevel
259 end
260 end
261 end
262 table.insert(result, {fillType=fillType, fillLevel=totalCapacity,
    capacity=totalFillLevel})
263 return result
264 end
```

HusbandryModuleStraw

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
22 function HusbandryModuleStraw:new(customMt)
23 return HusbandryModuleBase:new(customMt or
   HusbandryModuleStraw_mt)
24 end
```

delete

Description

Deletes instance

Definition

delete()

Code

```
28 function HusbandryModuleStraw:delete()
29 if self.unloadPlace ~= nil then
30 self.unloadPlace:delete()
31 self.unloadPlace = nil
32 end
33 if self.fillPlane ~= nil then
34 self.fillPlane:delete()
35 self.fillPlane = nil
36 end
37 end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
41  function HusbandryModuleStraw:initDataStructures()
42  HusbandryModuleStraw:superClass().initDataStructures(self)
43  self.unloadPlace = nil
44  end
```

load

Description

Loads data from xml

Definition

load(table xmlFile, string xmlKey, table rootNode)

Arguments

table	xmlFile	handle
string	xmlKey	from which to read the configuration
table	rootNode	of the husbandry

Return Values

boolean true if loading was successful else false

Code

```
52  function HusbandryModuleStraw:load(xmlFile, configKey, rootNode,
    owner)
53  if not HusbandryModuleStraw:superClass().load(self, xmlFile,
    configKey, rootNode, owner) then
54  return false
55  end
56
57  if not hasXMLProperty(xmlFile, configKey) then
58  return false
59  end
60
61  local strawNodeId = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, configKey .. "#node"))
62  if strawNodeId ~= nil then
63  local unloadPlace = UnloadTrigger:new(self.owner.isServer,
    self.owner.isClient)
64
65  if unloadPlace:load(strawNodeId, xmlFile, configKey, self) then
66  unloadPlace:register(true)
67  self.unloadPlace = unloadPlace
68
69  self.fillPlane = FillPlane:new()
70  self.fillPlane:load(rootNode, xmlFile, configKey..".fillPlane")
```

```
71
72  for fillTypeIndex, state in pairs(unloadPlace.fillTypes) do
73  self.fillLevels[fillTypeIndex] = 0.0
74  self.providedFillTypes[fillTypeIndex] = state
75  self:setCapacity(0.0)
76  end
77
78  return true
79  else
80  unloadPlace:delete()
81  return false
82  end
83  end
84
85  return false
86  end
```

loadFromXMLFile

Description

Loads information from attributes and node. Retrives from xml file information.

Definition

loadFromXMLFile(table xmlFile, string key)

Arguments

table	xmlFile	XML file handler
string	key	XML base key

Code

```
92  function HusbandryModuleStraw:loadFromXMLFile(xmlFile, key)
93  HusbandryModuleWater:superClass().loadFromXMLFile(self, xmlFile,
    key)
94
95  if self.fillPlane ~= nil then
96  self.fillPlane:setState(self:getFillProgress())
97  end
98  end
```

onIntervalUpdate

Description

Update straw usage

Definition

onIntervalUpdate(float dayToInterval)

Arguments

float	dayToInterval

Code

```
103  function HusbandryModuleStraw:onIntervalUpdate(dayToInterval)
```

```
104 HusbandryModuleStraw:superClass().onIntervalUpdate(self,
    dayToInterval)
105
106 if self.singleAnimalUsagePerDay > 0.0 then
107 local totalNumAnimals = self.owner:getNumOfAnimals()
108 local strawNeeded = totalNumAnimals *
    self.singleAnimalUsagePerDay * dayToInterval
109
110 if strawNeeded > 0.0 then
111 for fillTypeIndex, fillLevel in pairs(self.fillLevels) do
112 if fillLevel > 0.0 then
113 self:changeFillLevels(-math.min(strawNeeded, fillLevel),
    fillTypeIndex)
114
115 if self.fillPlane ~= nil then
116 self.fillPlane:setState(self:getFillProgress())
117 end
118 end
119 end
120 end
121 end
122 end
```

addFillLevelFromTool

Description

Definition

addFillLevelFromTool()

Code

```
126 function HusbandryModuleStraw:addFillLevelFromTool(farmId,
    deltaFillLevel, fillType)
127 local changed =
    HusbandryModuleStraw:superClass().addFillLevelFromTool(self,
    farmId, deltaFillLevel, fillType)
128
129 if deltaFillLevel > 0 and changed ~= 0 then
130 self.owner:updateGlobalProductionFactor()
131 end
132
133 if changed > 0 and self.unloadPlace ~= nil then
134 if self.unloadPlace:getIsFillTypeSupported(fillType) then
135 self.unloadPlace:raiseActive()
136 if self.fillPlane ~= nil then
137 self.fillPlane:setState(self:getFillProgress())
```

```
138 end
139 end
140 end
141 return changed
142 end
```

hasStraw

Description

Definition

hasStraw()

Code

```
146 function HusbandryModuleStraw:hasStraw()
147 local totalStraw = 0.0
148 for _, fillLevel in pairs(self.fillLevels) do
149 totalStraw = totalStraw + fillLevel
150 end
151 return totalStraw > 0.0
152 end
```

getFilltypeInfos

Description

Definition

getFilltypeInfos()

Code

```
156 function HusbandryModuleStraw:getFilltypeInfos()
157 local result = {}
158
159 for filltypeIndex, val in pairs(self.fillLevels) do
160 local fillType = g_fillTypeManager:getFillTypeByIndex(filltypeIndex)
161 local capacity = self.unloadPlace.target:getCapacity()
162 local fillLevel = self.unloadPlace.target:getFillLevel(filltypeIndex)
163
164 table.insert(result, {fillType=fillType, fillLevel=fillLevel, capacity=capacity})
165 end
166 return result
167 end
```

HusbandryModuleWater

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
22  function HusbandryModuleWater:new(customMt)
23  return HusbandryModuleBase:new(customMt or
    HusbandryModuleWater_mt)
24  end
```

delete

Description

Deletes instance

Definition

delete()

Code

```
28  function HusbandryModuleWater:delete()
29  if self.unloadPlace ~= nil then
30  self.unloadPlace:delete()
31  self.unloadPlace = nil
32  end
33  if self.fillPlane ~= nil then
34  self.fillPlane:delete()
35  self.fillPlane = nil
36  end
37  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
41  function HusbandryModuleWater:initDataStructures()
42  HusbandryModuleWater:superClass().initDataStructures(self)
43  self.unloadPlace = nil
44  end
```

load

Description

Loads data from xml

Definition

load(table xmlFile, string xmlKey, table rootNode)

Arguments

table	xmlFile	handle
string	xmlKey	from which to read the configuration

table rootNode of the husbandry

Return Values

boolean true if loading was successful else false

Code

```
52  function HusbandryModuleWater:load(xmlFile, configKey, rootNode,
    owner)
53  if not HusbandryModuleWater:superClass().load(self, xmlFile,
    configKey, rootNode, owner) then
54  return false
55  end
56
57  if not hasXMLProperty(xmlFile, configKey) then
58  return false
59  end
60
61  local waterNodeId = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, configKey .. "#node"))
62  if waterNodeId ~= nil then
63  local unloadPlace = UnloadTrigger:new(self.owner.isServer,
    self.owner.isClient)
64  if unloadPlace:load(waterNodeId, xmlFile, configKey, self) then
65  unloadPlace:register(true)
66  self.unloadPlace = unloadPlace
67  self.fillPlane = FillPlane:new()
68  self.fillPlane:load(rootNode, xmlFile, configKey..".fillPlane")
69
70  for fillTypeIndex, state in pairs(unloadPlace.fillTypes) do
71  self.fillLevels[fillTypeIndex] = 0.0
72  self.providedFillTypes[fillTypeIndex] = state
73  self:setCapacity(0.0)
74  end
75
76  return true
77  else
78  unloadPlace:delete()
79  return false
80  end
81  end
82
83  return false
84  end
```

loadFromXMLFile

Description

Loads information from attributes and node. Retrives from xml file information.

Definition

loadFromXMLFile(table xmlFile, string key)

Arguments

table	xmlFile	XML file handler
string	key	XML base key

Code

```
90  function HusbandryModuleWater:loadFromXMLFile(xmlFile, key)
91  HusbandryModuleWater:superClass().loadFromXMLFile(self, xmlFile,
    key)
92
93  if self.fillPlane ~= nil then
94  self.fillPlane:setState(self:getFillProgress())
95  end
96  end
```

onIntervalUpdate

Description

Update water usage

Definition

onIntervalUpdate(float dayToInterval)

Arguments

float dayToInterval

Code

```
101 function HusbandryModuleWater:onIntervalUpdate(dayToInterval)
102 HusbandryModuleWater:superClass().onIntervalUpdate(self,
    dayToInterval)
103
104 if self.singleAnimalUsagePerDay > 0.0 then
105 local totalNumAnimals = self.owner:getNumOfAnimals()
106 local waterNeeded = totalNumAnimals *
    self.singleAnimalUsagePerDay * dayToInterval
107
108 if waterNeeded > 0.0 then
109 for fillTypeIndex, fillLevel in pairs(self.fillLevels) do
110 if fillLevel > 0.0 then
111 self:changeFillLevels(-math.min(waterNeeded, fillLevel),
    fillTypeIndex)
112
113 if self.fillPlane ~= nil then
114 self.fillPlane:setState(self:getFillProgress())
115 end
```

```
116 end
117 end
118 end
119 end
120 end
```

addFillLevelFromTool

Description

Definition

addFillLevelFromTool()

Code

```
124 function HusbandryModuleWater:addFillLevelFromTool(farmId,
    deltaFillLevel, fillType)
125 local changed =
    HusbandryModuleWater:superClass().addFillLevelFromTool(self,
    farmId, deltaFillLevel, fillType)
126
127 if deltaFillLevel > 0 and changed ~= 0 then
128 self.owner:updateGlobalProductionFactor()
129 end
130
131 if changed > 0 and self.unloadPlace ~= nil then
132 if self.unloadPlace:getIsFillTypeSupported(fillType) then
133 self.unloadPlace:raiseActive()
134 if self.fillPlane ~= nil then
135 self.fillPlane:setState(self:getFillProgress())
136 end
137 end
138 end
139
140 return changed
141 end
```

hasWater

Description

Get a water multiplier

Definition

hasWater(float between)

Arguments

float between 0 and 1

Code

```
146 function HusbandryModuleWater:hasWater()
147 local totalWater = 0.0
148 for _, fillLevel in pairs(self.fillLevels) do
```

```
149 totalWater = totalWater + fillLevel
150 end
151 return totalWater > 0.0
152 end
```

getFilltypeInfos

Description

Definition

getFilltypeInfos()

Code

```
156 function HusbandryModuleWater:getFilltypeInfos()
157 local result = {}
158
159 for filltypeIndex, val in pairs(self.fillLevels) do
160 local fillType = g_fillTypeManager:getFillTypeByIndex(filltypeIndex)
161 local capacity = self.unloadPlace.target:getCapacity()
162 local fillLevel = self.unloadPlace.target:getFillLevel(filltypeIndex)
163
164 table.insert(result, {fillType=fillType, fillLevel=fillLevel, capacity=capacity})
165 end
166 return result
167 end
```

AnimalController

Description

new

Description

Create a new AnimalController.

Definition

new(table l10n)

Arguments

table l10n I18N reference

BrandManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
18  function BrandManager:new(customMt)
19  local self = AbstractManager:new(customMt or BrandManager_mt)
20
21  return self
22  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
26  function BrandManager:initDataStructures()
27  self.numOfBrands = 0
28  self.nameToIndex = {}
29  self.nameToBrand = {}
30  self.indexToBrand = {}
31
32  Brand = self.nameToIndex
33  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
38  function BrandManager:loadMapData(missionInfo)
39  BrandManager:superClass().loadMapData(self)
40
41  local xmlFile = loadXMLFile("brandsXML", "dataS/brands.xml")
42
43  local i = 0
44  while true do
45  local baseXMLName = string.format("brands.brand(%d)", i)
46
47  if not hasXMLProperty(xmlFile, baseXMLName) then
48  break
49  end
50  local name = getXMLString(xmlFile, baseXMLName .. "#name")
51  local title = getXMLString(xmlFile, baseXMLName .. "#title")
```

```
52  local image = getXMLString(xmlFile, baseXMLName .. "#image")
53
54  if title ~= nil and title:sub(1, 6) == "$l10n_" then
55  title = g_i18n:getText(title:sub(7))
56  end
57  self:addBrand(name, title, image, "", false)
58
59  i = i + 1
60  end
61  delete(xmlFile)
62
63  return true
64  end
```

addBrand

Description

Adds a new band

Definition

addBrand(string name, string title, string imageFilename, string baseDir, boolean isMod)

Arguments

string	name	the mapping name .e.g "HORSCH" for accessing a brand using the Brand.HORSCH enumeration
string	title	the displayed name of the brand in ui
string	imageFilename	the brand icon
string	baseDir	the image filename base directory
boolean	isMod	is a mod brand

Return Values

table the brand object or nil of an error occured

Code

```
74  function BrandManager:addBrand(name, title, imageFilename,
    baseDir, isMod)
75  if name == nil or name == "" then
76  print("Warning: Could not register brand. Name is missing or
    empty!")
77  return false
78  end
79  if title == nil or title == "" then
80  print("Warning: Could not register brand. Title is missing or
    empty!")
81  return false
82  end
83  if imageFilename == nil or imageFilename == "" then
84  print("Warning: Could not register brand. Image is missing or
    empty!")
```

```
85  return false
86  end
87  if baseDir == nil then
88  print("Warning: Could not register brand. Basedirectory not
    defined!")
89  return false
90  end
91
92  name = name:upper()
93
94  if ClassUtil.getIsValidIndexName(name) then
95  if self.nameToIndex[name] == nil then
96  self.numOfBrands = self.numOfBrands + 1
97  self.nameToIndex[name] = self.numOfBrands
98
99  local brand = {}
100 brand.index = self.numOfBrands
101 brand.name = name
102 brand.image = Utils.getFilename(imageFilename, baseDir)
103 brand.title = title
104 brand.isMod = isMod
105
106 self.nameToBrand[name] = brand
107 self.indexToBrand[self.numOfBrands] = brand
108
109 return brand
110 end
111 else
112 print("Warning: invalid brand name '" .. tostring(name) .. "'!
    Only capital letters allowed!")
113 end
114 end
```

getBrandByIndex

Description

Gets brand by index

Definition

getBrandByIndex(integer brandIndex)

Arguments

integer brandIndex brand index

Return Values

table brand the brand object

Code

```
120  function BrandManager:getBrandByIndex(brandIndex)
121  if brandIndex ~= nil then
122  return self.indexToBrand[brandIndex]
123  end
124  return nil
125  end
```

getBrandByName

Description

Gets brand by name

Definition

getBrandByName(string brandName)

Arguments

string brandName brand name

Return Values

table brand the brand object

Code

```
131  function BrandManager:getBrandByName(brandName)
132  if brandName ~= nil then
133  return self.nameToBrand[brandName:upper()]
134  end
135  return nil
136  end
```

getBrandIndexByName

Description

Gets brand index by name

Definition

getBrandIndexByName(string brandName)

Arguments

string brandName brand name

Return Values

integer brandIndex the brand index

Code

```
142  function BrandManager:getBrandIndexByName(brandName)
143  if brandName ~= nil then
144  return self.nameToIndex[brandName:upper()]
145  end
146  return nil
147  end
```

StoreItemUtil

Description

getIsVehicle

Description

Returns if a store item is a vehicle

Definition

getIsVehicle(table storeItem)

Arguments

table storeItem a storeitem object

Return Values

boolean true if storeitem is a vehicle, else false

getIsAnimal

Description

Returns if a store item is an animal

Definition

getIsAnimal(table storeItem)

Arguments

table storeItem a storeitem object

Return Values

boolean true if storeitem is an animal, else false

getIsPlaceable

Description

Returns if a store item is a placeable

Definition

getIsPlaceable(table storeItem)

Arguments

table storeItem a storeitem object

Return Values

boolean true if storeitem is a placeable, else false

getIsObject

Description

Returns if a store item is an object

Definition

getIsObject(table storeItem)

Arguments

table storeItem a storeitem object

Return Values

boolean true if storeitem is an object, else false

getIsHandTool

Description

Returns if a store item is a handtool

Definition

getIsHandTool(table storeItem)

Arguments

table storeItem a storeitem object

Return Values

boolean true if storeitem is a handtool, else false

getIsConfigurable

Description

Returns if a store item is configurable.
Checks if there are any configurations and also if any of the configurations has more than just one option.

Definition

getIsConfigurable(table storeItem)

Arguments

table storeItem a storeitem object

Return Values

boolean true if storeitem is configurable, else false

getIsLeasable

Description

Returns if a store item is leaseable

Definition

getIsLeasable(table storeItem)

Arguments

table storeItem a storeitem object

Return Values

boolean true if storeitem is leaseable, else false

getDefaultConfigId

Description

Get the default config id

Definition

getDefaultConfigId(table storeItem, string configurationName)

Arguments

table storeItem a storeitem object

string configurationName name of the configuration

Return Values

integer configId the default config id

getDefaultPrice

Description

Get the default price

Definition

getDefaultPrice(table storeItem, table configurations)

Arguments

table storeItem a storeitem object

table configurations list of configurations

Return Values

integer the default price

getDailyUpkeep

Description

Get the daily upkeep

Definition

getDailyUpkeep(table storeItem, table configurations)

Arguments

table storeItem	a storeitem object
table configurations	list of configurations

Return Values

integer the daily upkeep

getCosts

Description

Get the costs of storeitem

Definition

getCosts(table storeItem, table configurations, string costType)

Arguments

table storeItem	a storeitem object
table configurations	list of configurations
string costType	the cost type

Return Values

integer cost of the storeitem

addConfigurationItem

Description

Adds a configuration item to the given list

Definition

addConfigurationItem(table configurationItems, string name, string desc, float price, integer dailyUpkeep)

Arguments

table	configurationItems	a list of configurationItems
string	name	name of the configuration
string	desc	desc of the configuration
float	price	price of the configuration
integer	dailyUpkeep	dailyUpkeep of the configuration

Return Values

table config object

getFunctionsFromXML

Description

Gets the storeitem functions from xml

Definition

getFunctionsFromXML(integer xmlFile, string storeDataXMLName, string customEnvironment)

Arguments

integer	xmlFile	the xml handle
string	storeDataXMLName	name of the parent xml element
string	customEnvironment	a custom environment

Return Values

table functions list of storeitem functions

getSpecsFromXML

Description

Gets the storeitem specs from xml

Definition

getSpecsFromXML(table specTypes, integer xmlFile, string customEnvironment)

Arguments

table	specTypes	list of spec types
integer	xmlFile	the xml handle
string	customEnvironment	a custom environment

Return Values

table specs list of storeitem specs

getBrandIndexFromXML

Description

Gets the storeitem brand index from xml

Definition

getBrandIndexFromXML(integer xmlFile, string storeDataXMLName, string xmlFilename)

Arguments

integer	xmlFile	the xml handle
string	storeDataXMLName	name of the parent xml element
string	xmlFilename	filepath of the xml file

Return Values

integer brandIndex the brandindex

getVRamUsageFromXML

Description

Gets the storeitem vram usage from xml

Definition

getVRamUsageFromXML(integer xmlFile, string storeDataXMLName)

Arguments

integer	xmlFile	the xml handle
string	storeDataXMLName	name of the parent xml element

Return Values

integer	sharedVramUsage	the shared vram usage
integer	perInstanceVramUsage	the per instance vram usage
boolean	ignoreVramUsage	true if vram usage should be ignored else false

getConfigurationsFromXML

Description

Gets the storeitem configurations from xml

Definition

getConfigurationsFromXML(integer xmlFile, string baseXMLName, string baseDir, string customEnvironment, boolean isMod)

Arguments

integer	xmlFile	the xml handle

string	baseXMLName	the name of the base xml element
string	baseDir	the base directory
string	customEnvironment	a custom environment
boolean	isMod	true if the storeitem is a mod, else false

Return Values

table configurations a list of configurations

getConfigurationSetsFromXML

Description

Gets predefined configuration sets

Definition

getConfigurationSetsFromXML(table storeItem, integer xmlFile, string baseXMLName, string baseDir, string customEnvironment, boolean isMod)

Arguments

table	storeItem	a storeItem
integer	xmlFile	the xml handle
string	baseXMLName	the name of the base xml element
string	baseDir	the base directory
string	customEnvironment	a custom environment
boolean	isMod	true if the storeitem is a mod, else false

Return Values

table configuration sets

SimpleState

Description

new

Description

Creating instance of lightweight state machine.

Definition

new()

Code

```
17  function SimpleState:new(id, owner, stateMachine, custom_mt)
18  local self = {}
19  setmetatable(self, custom_mt or SimpleState_mt)
20  self.id = id
21  self.owner = owner
22  self.stateMachine = stateMachine
23  return self
24  end
```

delete

Description

Definition

delete()

Code

```
28  function SimpleState:delete()
```

```
29  end
```

activate

Description

Activate method

Definition

activate()

Code

```
33  function SimpleState:activate(parms)
34  end
```

deactivate

Description

Deactivate method

Definition

deactivate()

Code

```
38  function SimpleState:deactivate()
39  end
```

update

Description

update method

Definition

update(float dt)

Arguments

float dt in ms

Code

```
44  function SimpleState:update(dt)
45  end
```

SimpleStateMachine

Description

new

Description

Creating instance of lightweight state machine.

Definition

new()

Code

```
17  function SimpleStateMachine:new(custom_mt)
18  local self = {}
19  setmetatable(self, custom_mt or SimpleStateMachine_mt)
20
21  self.currentState = nil
22  self.states = {}
23  return self
```

```
24  end
```

delete

Description

Definition

delete()

Code

```
28  function SimpleStateMachine:delete()
29  self:reset()
30  end
```

addState

Description

Adds a state to the state machine

Definition

addState(string;integer stateId, state instance)

Arguments

string;integer stateId	
state	instance of the state

Code

```
36  function SimpleStateMachine:addState(stateId, state)
37  self.states[stateId] = state
38  end
```

removeState

Description

Removes a state

Definition

removeState(string;integer stateId)

Arguments

string;integer stateId	

Code

```
43  function SimpleStateMachine:removeState(stateId)
44  if self.currentState == self.states[stateId] then
45  self.currentState = nil
46  end
47  self.states[stateId] = nil
48  end
```

reset

Description

Clears all states

Definition

reset()

Code

```
52  function SimpleStateMachine:reset()
```

```
53  self.states = {}
54  self.currentState = nil
55  end
```

changeState

Description

Changes current state

Definition

changeState(string;integer stateId)

Arguments

string;integer stateId

Code

```
60  function SimpleStateMachine:changeState(stateId, parms)
61  if self.states[stateId] ~= nil then
62  if self.currentState ~= nil then
63  self.currentState:deactivate()
64  end
65  self.currentState = self.states[stateId]
66  self.currentState:activate(parms)
67  end
68  end
```

update

Description

Calls the update method of a state

Definition

update(float dt)

Arguments

float dt in ms

Code

```
73  function SimpleStateMachine:update(dt)
74  if self.currentState ~= nil then
75  self.currentState:update(dt)
76  end
77  end
```

DensityMapHeightManager

Description

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
57  function DensityMapHeightManager:loadMapData(xmlFile, missionInfo, baseDirectory)
58      DensityMapHeightManager:superClass().loadMapData(self)
59
60      self:loadDefaultTypes(missionInfo, baseDirectory)
61      return XMLUtil.loadDataFromMapXML(xmlFile, "densityMapHeightTypes", baseDirectory, self, self.loadDensityMapHeightTypes, missionInfo)
62  end
```

DensityMapHeightUtil

Description

getCanTipToGround

Description

Returns of can tip to ground

Definition

getCanTipToGround(integer fillType)

Arguments

integer fillType fill type

Return Values

boolean canTip can tip to ground

getFillTypeAtLine

Description

Returns fill type at line

Definition

getFillTypeAtLine(float sx, float sy, float sz, float ex, float ey, float ez, float radius)

Arguments

float sx	start x position
float sy	start y position
float sz	start z position
float ex	end x position
float ey	end y position
float ez	end z position
float radius	radius to check

Return Values

integer fillType fill type found

getFillLevelAtArea

Description

Returns fill level at area

Definition

getFillLevelAtArea(integer fillType, float x0, float z0, float x1, float z1, float x2, float z2)

Arguments

integer	fillType	fill type to check
float	x0	start x position

float	z0	start z position
float	x1	width x position
float	z1	width z position
float	x2	heigth x position
float	z2	height z position

Return Values

float fillLevel fill level found

getHeightAtWorldPos

Description

Returns density height at world pos

Definition

getHeightAtWorldPos(float x, float y, float z)

Arguments

float x world x position

float y world y position

float z world z position

Return Values

float densityHeight density height

float deltaDensityHeight delta of density height to terrain underneath

getCollisionHeightAtWorldPos

Description

Returns height of physics collision of density at world pos

Definition

getCollisionHeightAtWorldPos(float x, float y, float z)

Arguments

float x world x position

float y world y position

float z world z position

Return Values

float physicsDensityHeight density height

float deltaPhysicsDensityHeight delta of physics collision to terrain underneath

tipToGroundAroundLine

Description

Returns fill type at line

Definition

tipToGroundAroundLine(table vehicle, float delta, integer filltype, float sx, float sy, float sz, float ex, float ey, float ez, float innerRadius, float radius, float lineOffset, boolean limitToLineHeight, table occlusionAreas, boolean useOcclusionAreas, boolean applyChanges)

Arguments

table	vehicle	vehicle that is tipping
float	delta	delta to tip
integer	filltype	fill type to tip
float	sx	start x position

float	sy	start y position
float	sz	start z position
float	ex	end x position
float	ey	end y position
float	ez	end z position
float	innerRadius	inner radius
float	radius	radius
float	lineOffset	line offset
boolean	limitToLineHeight	limit to line height
table	occlusionAreas	occlusion areas
boolean	useOcclusionAreas	use occlusion areas
boolean	applyChanges	apply changes to density height map (default: true)

Return Values

float dropped	real fill level dropped
float lineOffset	line offset

getCanTipToGroundAroundLine

Description

Returns if it's possible to tip around ground line

Definition

getCanTipToGroundAroundLine(table vehicle, float delta, integer filltype, float sx, float sy, float sz, float ex, float ey, float ez, float innerRadius, float radius, float lineOffset, boolean limitToLineHeight, table occlusionAreas, boolean useOcclusionAreas)

Arguments

table	vehicle	vehicle that is tipping
float	delta	delta to tip
integer	filltype	fill type to tip
float	sx	start x position
float	sy	start y position
float	sz	start z position
float	ex	end x position
float	ey	end y position
float	ez	end z position
float	innerRadius	inner radius
float	radius	radius
float	lineOffset	line offset
boolean	limitToLineHeight	limit to line height
table	occlusionAreas	occlusion areas
boolean	useOcclusionAreas	use occlusion areas

Return Values

float dropped	real fill level dropped
float lineOffset	line offset

removeFromGroundByArea

Description

Removes density from fill type in given area

Definition

removeFromGroundByArea(float x0, float z0, float x1, float z1, float x2, float z2, integer fillType)

Arguments

float	x0	start x position
float	z0	start z position
float	x1	width x position
float	z1	width z position
float	x2	heigth x position
float	z2	height z position
integer	fillType	fill type to remove

Return Values

float fillLevel fill level removed

changeFillTypeAtArea

Description

Change fill type at area

Definition

changeFillTypeAtArea(float x0, float z0, float x1, float z1, float x2, float z2, integer fillType, integer newFillType)

Arguments

float	x0	start x position
float	z0	start z position
float	x1	width x position
float	z1	width z position
float	x2	heigth x position
float	z2	height z position
integer	fillType	old fill type
integer	newFillType	new fill type

Return Values

float fillLevel fill level changed

FarmlandManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
20  function FarmlandManager:new(customMt)
21  local self = AbstractManager:new(customMt or FarmlandManager_mt)
22  return self
23  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
27  function FarmlandManager:initDataStructures()
28  self.farmlands = {}
29  self.sortedFarmlandIds = {}
30  -- mapping table farmland id to farm id
31  self.farmlandMapping = {}
32  self.localMap = nil
33  self.localMapWidth = 0
34  self.localMapHeight = 0
35  self.numberOfBits = 8
36  self.stateChangeListener = {}
37  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
43  function FarmlandManager:loadMapData(xmlFile)
44  FarmlandManager:superClass().loadMapData(self)
45  return XMLUtil.loadDataFromMapXML(xmlFile, "farmlands",
    g_currentMission.baseDirectory, self, self.loadFarmlandData)
46  end
```

loadFarmlandData

Description

Load data on map load

Definition

loadFarmlandData()

Return Values

boolean true if loading was successful else false

Code

```
51  function FarmlandManager:loadFarmlandData(xmlFile)
52  local filename = Utils.getFilename(getXMLString(xmlFile,
    "map.farmlands#densityMapFilename"),
    g_currentMission.baseDirectory)
```

```
53
54  -- number of channels for farmland bit vector
55  self.numberOfBits = Utils.getNoNil(getXMLInt(xmlFile,
    "map.farmlands#numChannels"), 8)
56  self.pricePerHa = Utils.getNoNil(getXMLFloat(xmlFile,
    "map.farmlands#pricePerHa"), 60000)
57
58  FarmlandManager.NOT_BUYABLE_FARM_ID = 2^self.numberOfBits-1
59
60  -- load a bitvector
61  self.localMap = createBitVectorMap("FarmlandMap")
62  local success = loadBitVectorMapFromFile(self.localMap, filename,
    self.numberOfBits)
63  if not success then
64  print("Warning: Loading farmland file '"..tostring(filename).."'
    failed!")
65  return false
66  end
67
68  self.localMapWidth, self.localMapHeight =
    getBitVectorMapSize(self.localMap)
69
70  local farmlandSizeMapping = {}
71  local farmlandCenterData = {}
72  local numOfFarmlands = 0
73  local maxFarmlandId = 0
74  local missingFarmlandDefinitions = false
75  for x = 0, self.localMapWidth - 1 do
76  for y = 0, self.localMapHeight - 1 do
77  local value = getBitVectorMapPoint(self.localMap, x, y, 0,
    self.numberOfBits)
78
79  if value > 0 then
80  if self.farmlandMapping[value] == nil then
81  farmlandSizeMapping[value] = 0
82  farmlandCenterData[value] = {sumPosX=0, sumPosZ=0}
83  self.farmlandMapping[value] = FarmlandManager.NO_OWNER_FARM_ID
84  numOfFarmlands = numOfFarmlands + 1
85  maxFarmlandId = math.max(value, maxFarmlandId)
86  end
87
88  farmlandSizeMapping[value] = farmlandSizeMapping[value] + 1
```

```
89  farmlandCenterData[value].sumPosX =
    farmlandCenterData[value].sumPosX + (x-0.5)
90  farmlandCenterData[value].sumPosZ =
    farmlandCenterData[value].sumPosZ + (y-0.5)
91  else
92  missingFarmlandDefinitions = true
93  end
94  end
95  end
96
97  if missingFarmlandDefinitions then
98  print("Warning: Farmland-Ids not set for all pixel in farmland-
    infoLayer!")
99  end
100
101 local isNewSavegame = not g_currentMission.missionInfo.isValid
102
103 local i = 0
104 while true do
105 local key = string.format("map.farmlands.farmland(%d)", i)
106 if not hasXMLProperty(xmlFile, key) then
107 break
108 end
109
110 local farmland = Farmland:new()
111 if farmland:load(xmlFile, key) and self.farmlands[farmland.id] ==
    nil and self.farmlandMapping[farmland.id] ~= nil then
112 self.farmlands[farmland.id] = farmland
113 table.insert(self.sortedFarmlandIds, farmland.id)
114
115 local shouldAddDefaults = isNewSavegame and
    g_currentMission.missionInfo.hasInitiallyOwnedFarmlands and not
    g_currentMission.missionDynamicInfo.isMultiplayer
116 if farmland.defaultFarmProperty and shouldAddDefaults and
    g_currentMission:getIsServer() then
117 self:setLandOwnership(farmland.id,
    FarmManager.SINGLEPLAYER_FARM_ID)
118 end
119 else
120 if self.farmlandMapping[farmland.id] == nil then
121 print("Error: Farmland-Id " .. tostring(farmland.id) .. " not
    defined in farmland ownage file '"..filename.."'. Skipping
    farmland definition!")
```

```
122 end
123 if self.farmlands[farmland.id] ~= nil then
124 print("Error: Farmland-id '"..tostring(farmland.id).."' already exists! Ignore it!")
125 end
126 farmland:delete()
127 end
128
129 i = i + 1
130 end
131
132 for index, _ in pairs(self.farmlandMapping) do
133 if index ~= FarmlandManager.NOT_BUYABLE_FARM_ID and self.farmlands[index] == nil then
134 print("Error: Farmland-Id " .. tostring(index) .. " not defined in farmland xml file!")
135 end
136 end
137
138 local transformFactor = g_currentMission.terrainSize / self.localMapWidth
139 local pixelToSqm = transformFactor*transformFactor
140
141 for id, farmland in pairs(self.farmlands) do
142 local ha = MathUtil.areaToHa(farmlandSizeMapping[id], pixelToSqm)
143 farmland:setArea(ha)
144
145 local posX = ((farmlandCenterData[id].sumPosX / farmlandSizeMapping[id]) - self.localMapWidth*0.5) * transformFactor
146 local posZ = ((farmlandCenterData[id].sumPosZ / farmlandSizeMapping[id]) - self.localMapHeight*0.5) * transformFactor
147 self.farmlands[id]:setFarmlandIndicatorPosition(posX, posZ)
148 end
149
150 g_messageCenter:subscribe(MessageType.FARM_DELETED, self.farmDestroyed, self)
151
152 if g_currentMission:getIsServer() then
153 if g_addCheatCommands then
154 -- master user only cheats (will be added in setMasterUserLocal too)
```

```
155 addConsoleCommand("gsBuyFarmland", "Buys farmland with given id",
    "consoleCommandBuyFarmland", self)
156 addConsoleCommand("gsBuyAllFarmlands", "Buys all farmlands",
    "consoleCommandBuyAllFarmlands", self)
157 addConsoleCommand("gsSellFarmland", "Sells farmland with given
    id", "consoleCommandSellFarmland", self)
158 addConsoleCommand("gsSellAllFarmlands", "Sells all farmlands",
    "consoleCommandSellAllFarmlands", self)
159 end
160 end
161
162 return true
163 end
```

unloadMapData

Description

Unload data on mission delete

Definition

unloadMapData()

Code

```
167 function FarmlandManager:unloadMapData()
168 removeConsoleCommand("gsBuyFarmland")
169 removeConsoleCommand("gsBuyAllFarmlands")
170 removeConsoleCommand("gsSellFarmland")
171 removeConsoleCommand("gsSellAllFarmlands")
172
173 g_messageCenter:unsubscribeAll(self)
174
175 if self.localMap ~= nil then
176 delete(self.localMap)
177 self.localMap = nil
178 end
179 if (self.farmlands ~= nil) then
180 for _, farmland in pairs(self.farmlands) do
181 farmland:delete()
182 end
183 end
184
185 FarmlandManager:superClass().unloadMapData(self)
186 end
```

saveToXMLFile

Description

Write farmland ownage data to savegame file

Definition

saveToXMLFile(string xmlFilename)

Arguments

string xmlFilename file path

Return Values

boolean true if loading was successful else false

Code

```
192 function FarmlandManager:saveToXMLFile(xmlFilename)
193 -- save farmland to xml
194 local xmlFile = createXMLFile("farmlandsXML", xmlFilename,
    "farmlands")
195 if xmlFile ~= nil then
196 local index = 0
197 for farmlandId, farmId in pairs(self.farmlandMapping) do
198 local farmlandKey = string.format("farmlands.farmland(%d)",
    index)
199 setXMLInt(xmlFile, farmlandKey.."#id", farmlandId)
200 setXMLInt(xmlFile, farmlandKey.."#farmId", Utils.getNoNil(farmId,
    FarmlandManager.NO_OWNER_FARM_ID))
201 index = index + 1
202 end
203
204 saveXMLFile(xmlFile)
205 delete(xmlFile)
206
207 return true
208 end
209
210 return false
211 end
```

loadFromXMLFile

Description

Load farmland ownage data from xml savegame file

Definition

loadFromXMLFile(string filename)

Arguments

string filename xml filename

Code

```
216 function FarmlandManager:loadFromXMLFile(xmlFilename)
217 if xmlFilename == nil then
218 return false
219 end
```

```
220
221 local xmlFile = loadXMLFile("farmlandXML", xmlFilename)
222 if xmlFile == 0 then
223 return false
224 end
225
226 local farmlandCounter = 0
227 while true do
228 local key = string.format("farmlands.farmland(%d)",
    farmlandCounter)
229 local farmlandId = getXMLInt(xmlFile, key .. "#id")
230 if farmlandId == nil then
231 break
232 end
233 local farmId = getXMLInt(xmlFile, key .. "#farmId")
234 if farmId > FarmlandManager.NO_OWNER_FARM_ID then
235 self:setLandOwnership(farmlandId, farmId)
236 end
237
238 farmlandCounter = farmlandCounter + 1
239 end
240
241 delete(xmlFile)
242
243 g_farmManager:mergeFarmlandsForSingleplayer()
244
245 return true
246 end
```

delete

Description

Deletes farm land manager

Definition

delete()

Code

```
250 function FarmlandManager:delete()
251 end
```

getLocalMap

Description

Gets farmland bit vector handle

Definition

getLocalMap()

Return Values

integer mapHandle id of bitvector

Code

```
256  function FarmlandManager:getLocalMap()
257  return self.localMap
258  end
```

getIsOwnedByFarmAtWorldPosition

Description

Checks if farm owns given world position

Definition

getIsOwnedByFarmAtWorldPosition(integer farmId, float worldPosX, float worldPosZ)

Arguments

integer farmId farm id

float worldPosX world position x

float worldPosZ world position z

Return Values

boolean isOwned true if farm owns world position point, else false

Code

```
266  function FarmlandManager:getIsOwnedByFarmAtWorldPosition(farmId,
     worldPosX, worldPosZ)
267  if farmId == FarmlandManager.NO_OWNER_FARM_ID or farmId == nil or
     self.farmlandMapping[farmId] == nil then
268  return false
269  end
270  local farmlandId = self:getFarmlandIdAtWorldPosition(worldPosX,
     worldPosZ)
271  return self.farmlandMapping[farmlandId] == farmId
272  end
```

getFarmlandOwner

Description

Gets farmland owner

Definition

getFarmlandOwner(integer farmlandId)

Arguments

integer farmlandId farmland id

Return Values

integer farmId id of farm. Returns 0 if land is not owned by anyone

Code

```
278  function FarmlandManager:getFarmlandOwner(farmlandId)
279  if farmlandId == nil or self.farmlandMapping[farmlandId] == nil
     then
280  return FarmlandManager.NO_OWNER_FARM_ID
281  end
```

```
282
283 return self.farmlandMapping[farmlandId]
284 end
```

getFarmlandIdAtWorldPosition

Description

Gets farmland id at given world position

Definition

getFarmlandIdAtWorldPosition(float worldPosX, float worldPosZ)

Arguments

float worldPosX world position x

float worldPosZ world position z

Return Values

integer farmlandId farmland id. if 0, world position is no valid/buyable farmland

Code

```
291 function FarmlandManager:getFarmlandIdAtWorldPosition(worldPosX,
    worldPosZ)
292 local localPosX, localPosZ =
    self:convertWorldToLocalPosition(worldPosX, worldPosZ)
293 return getBitVectorMapPoint(self.localMap, localPosX, localPosZ,
    0, self.numberOfBits)
294 end
```

getIsValidFarmlandId

Description

Checks if given farmland-id is valid

Definition

getIsValidFarmlandId(integer farmlandId)

Arguments

integer farmlandId farmland id

Return Values

boolean isValid true if id is valid, else false

Code

```
311 function FarmlandManager:getIsValidFarmlandId(farmlandId)
312 if farmlandId == nil or farmlandId == 0 or farmlandId < 0 then
313 return false
314 end
315 if self:getFarmlandById(farmlandId) == nil then
316 return false
317 end
318 return true
319 end
```

setLandOwnership

Description

Sets farm land ownership

Definition

setLandOwnership(integer farmlandId, integer farmId)

Arguments

integer farmlandId farm land id

integer farmId farm id. set farmid to 0 to sell farm land

Code

```lua
function FarmlandManager:setLandOwnership(farmlandId, farmId)
if not self:getIsValidFarmlandId(farmlandId) then
return false
end
if farmId == nil or farmId < FarmlandManager.NO_OWNER_FARM_ID or farmId == FarmlandManager.NOT_BUYABLE_FARM_ID then
return false
end

local farmland = self:getFarmlandById(farmlandId)
if farmland == nil then
print("Warning: Farmland not defined in map!")
return
end

self.farmlandMapping[farmlandId] = farmId
farmland.isOwned = farmId ~= FarmlandManager.NO_OWNER_FARM_ID

for _, listener in pairs(self.stateChangeListener) do
listener:onFarmlandStateChanged(farmlandId, farmId)
end
end
```

getFarmlandById

Description

Gets farmland by id

Definition

getFarmlandById(integer farmlandId)

Arguments

integer farmlandId farmland id

Return Values

table farmland farmland object

Code

```lua
function FarmlandManager:getFarmlandById(farmlandId)
return self.farmlands[farmlandId]
end
```

getFarmlands

Description

Gets all farmlands

Definition

getFarmlands()

Return Values

table farmlands all available farmlands

Code

```
358 function FarmlandManager:getFarmlands()
359 return self.farmlands
360 end
```

getOwnedFarmlandIdsByFarmId

Description

Gets list of owned farmland ids for given farm

Definition

getOwnedFarmlandIdsByFarmId(integer farmId)

Arguments

integer farmId farm id

Return Values

farmlandIds table list of farmland ids owned by given farm id

Code

```
370 function FarmlandManager:getOwnedFarmlandIdsByFarmId(id)
371 local farmlandIds = {}
372 for farmlandId, farmId in pairs(self.farmlandMapping) do
373 if farmId == id then
374 table.insert(farmlandIds, farmlandId)
375 end
376 end
377 return farmlandIds
378 end
```

convertWorldToLocalPosition

Description

Converts world to local position

Definition

convertWorldToLocalPosition(float worldPosX, float worldPosZ)

Arguments

float worldPosX world position x

float worldPosZ world position z

Return Values

float localPosX local position x

float localPosZ local position z

Code

```
386 function FarmlandManager:convertWorldToLocalPosition(worldPosX,
    worldPosZ)
```

```
387  local terrainSize = g_currentMission.terrainSize
388  if worldPosX == nil then print(debug.traceback()) end
389  return math.floor(self.localMapWidth *
     (worldPosX+terrainSize*0.5) / terrainSize),
390  math.floor(self.localMapHeight * (worldPosZ+terrainSize*0.5) /
     terrainSize)
391  end
```

addStateChangeListener

Description

Adds a farmland state change listener

Definition

addStateChangeListener(table listener)

Arguments

table listener state listener

Code

```
404  function FarmlandManager:addStateChangeListener(listener)
405  if listener ~= nil and listener.onFarmlandStateChanged ~= nil
     then
406  self.stateChangeListener[listener] = listener
407  end
408  end
```

removeStateChangeListener

Description

Removes a farmland state change listener

Definition

removeStateChangeListener(table listener)

Arguments

table listener state listener

Code

```
413  function FarmlandManager:removeStateChangeListener(listener)
414  if listener ~= nil then
415  self.stateChangeListener[listener] = nil
416  end
417  end
```

EffectManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
18  function EffectManager:new(customMt)
19  local self = AbstractManager:new(customMt or EffectManager_mt)
20
21  return self
22  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
26  function EffectManager:initDataStructures()
27  self.runningEffects = {}
28  self.registeredEffectClasses = {}
29  end
```

WeatherTypeManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
19  function WeatherTypeManager:new(customMt)
20  return AbstractManager:new(customMt or WeatherTypeManager_mt)
21  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
25  function WeatherTypeManager:initDataStructures()
26  self.weatherTypes = {}
27  self.nameToIndex = {}
28  self.indexToName = {}
29  self:loadDefaultTypes()
30
31  WeatherType = self.nameToIndex
```

```
32  end
```

addWeatherType

Description

Adds a new weather type

Definition

addWeatherType(string name)

Arguments

string name weather type name

Return Values

table brandColor brandColor object

AccessHandler

Description

canPlayerAccess

Description

Determine if the currrent player can access given object

Definition

canPlayerAccess(table object, table player)

Arguments

table object

table player optional player

canFarmAccess

Description

Determine if the currrent player can access given object

Definition

canFarmAccess(table object, integer farmId, boolean [optional])

Arguments

table	object	
integer	farmId	farmId to check with (not NOBODY)
boolean	[optional]	Allow spectator farm when source and target farms are equal (0==0)

BanStorage

Description

setPath

Description

Set the path of the ban list XML file.

Definition

setPath()

addUser

Description

Add a user to the ban list.

Definition

addUser(string uniqueUserId, string nickname, bool saveImmediately)

Arguments

string uniqueUserId	Unique user identified

string nickname User display nickname
bool saveImmediately [optional, default=true] If true, will immediately persist the change

removeUser

Description

Remove a user from the ban list.

Definition

removeUser(string uniqueUserId, bool saveImmediately)

Arguments

string uniqueUserId Unique user identified
bool saveImmediately [optional, default=true] If true, will immediately persist the change

getBans

Description

Get the array of banned users.
Callers must not manipulate the returned array directly. See BanStorage:addUser() for array item structure.

Definition

getBans()

FarmStats

Description

getStatisticData

Description

Turn all data into text to display in the finances menu

Definition

getStatisticData()

Return Values

table data Data for the menu

addStatistic

Description

Add a statistic to the statistic data table

Definition

addStatistic()

merge

Description

Used to merge farms into each other when playing SP on a MP game

Definition

merge()

AbstractFieldMission

Description

resetField

Description

Resets a field to the state it has before a mission. Make sure it it completable

Definition

resetField()

completeField
Description

Sets the field to the completed state (100%)

Definition

completeField()

getFieldSize
Description

Get a categorized field size for leased vehicle groups.

Definition

getFieldSize()

getVehicleVariant
Description

Get variant of the vehicle needed.

Definition

getVehicleVariant(string? variant)

Arguments

string? variant variant name or nil for no filter

calculateVehicleUseCost
Description

Cost on reward for using farmers vehicles

Definition

calculateVehicleUseCost()

getMaxCutLiters
Description

Get the maximum amount of liters to be gotten from a field when harvested

Definition

getMaxCutLiters()

isSpawnSpaceAvailable
Description

Check if enough space is available at spawn to spawn all vehicles

Definition

isSpawnSpaceAvailable()

calculateReimbursement
Description

Calculate the money given for the contents of leased vehicles
Excludes diesel, def and air

Definition

calculateReimbursement()

getNPC
Description

Get the NPC giving this mission

Definition

getNPC()

AbstractMission
Description
validate
Description

Validate that the mission is still able to run. If false, it is deleted by the mission manager

Definition

validate()

FieldManager
Description
new
Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
33  function FieldManager:new(customMt)
34  local self = AbstractManager:new(customMt or FieldManager_mt)
35
36  return self
37  end
```

initDataStructures
Description

Initialize data structures

Definition

initDataStructures()

Code

```
41  function FieldManager:initDataStructures()
42  self.fields = {}
43  self.farmlandIdFieldMapping = {}
44  end
```

loadMapData
Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
49  function FieldManager:loadMapData(xmlFile)
50  FieldManager:superClass().loadMapData(self)
51
52  g_currentMission:addUpdateable(self)
53
54  self.setFieldPartitionModifier =
    DensityMapModifier:new(g_currentMission.terrainDetailId)
55  self.detailModifier =
    DensityMapModifier:new(g_currentMission.terrainDetailId)
56
57  local weedType = g_fruitTypeManager:getWeedFruitType()
58  if weedType ~= nil then
59  local ids = g_currentMission.fruits[weedType.index]
60  local weed = weedType.weed
61
62  self.weedModifier = DensityMapModifier:new(ids.id,
    weedType.startStateChannel, weedType.numStateChannels)
63  end
64
65  -- create list of valid/available fruit types
66  self.availableFruitTypeIndices = {}
67  self.minFieldGrowthStateTime = math.huge
68  for _, fruitType in ipairs(g_fruitTypeManager:getFruitTypes()) do
69  if fruitType.useForFieldJob and fruitType.allowsSeeding and
    fruitType.needsSeeding then
70  if g_currentMission.fruits[fruitType.index] ~= nil then
71  table.insert(self.availableFruitTypeIndices, fruitType.index)
72  self.minFieldGrowthStateTime =
    math.min(self.minFieldGrowthStateTime, fruitType.growthStateTime)
73  end
74  end
75  end
76  self.fruitTypesCount = table.getn(self.availableFruitTypeIndices)
77
78  self.fieldIndexToCheck = table.getn(self.fields)
79
80  -- Connect farmlands to fields first. We need the farmlands to
    skip overriding owned fields (in order to have working starter
    fields)
81  g_deferredLoadingManager:addSubtask(function()
82  for _, field in ipairs(self.fields) do
83  local posX, posZ = field:getCenterOfFieldWorldPosition()
```

```
84  local farmland = g_farmlandManager:getFarmlandAtWorldPosition(posX, posZ)
85
86  field:setFarmland(farmland)
87
88  if self.farmlandIdFieldMapping[farmland.id] == nil then
89  self.farmlandIdFieldMapping[farmland.id] = {}
90  end
91
92  table.insert(self.farmlandIdFieldMapping[farmland.id], field)
93  end
94  end)
95
96  -- New save game
97  if not g_currentMission.missionInfo.isValid and g_server ~= nil then
98  g_deferredLoadingManager:addSubtask(function()
99  local index = 1
100
101 for _, field in pairs(self.fields) do
102 if field:getIsAIActive() and field.fieldMissionAllowed and not field.farmland.isOwned then
103 local fruitIndex = self.availableFruitTypeIndices[index]
104 if field.fieldGrassMission then
105 fruitIndex = FruitType.GRASS
106 end
107
108 local fruitDesc = g_fruitTypeManager:getFruitTypeByIndex(fruitIndex)
109 local fieldState = FieldManager.FIELDSTATE_GROWING
110
111
112 local growthState
113 if fruitDesc.minHarvestingGrowthState == 0 and fruitDesc.maxHarvestingGrowthState == 0 and fruitDesc.cutState == 0 then
114 growthState = 2
115 else
116 if fruitDesc.minPreparingGrowthState ~= -1 then
117 growthState = math.random(1, fruitDesc.maxPreparingGrowthState + 1)
118 else
```

```
119 growthState = math.random(1, fruitDesc.maxHarvestingGrowthState +
    1)
120 end
121 end
122
123 if fruitIndex == FruitType.GRASS and growthState == 1 then
124 growthState = 2
125 end
126
127 local weedValue = 0
128 if fruitDesc.plantsWeed then
129 weedValue = 1
130 end
131
132 local plowFactor = 0
133 if not g_currentMission.missionInfo.plowingRequiredEnabled then
134 plowFactor = g_currentMission.plowCounterMaxValue
135 else
136 plowFactor = math.random(0, g_currentMission.plowCounterMaxValue)
137 end
138
139 local sprayFactor = math.random(0,
    g_currentMission.sprayLevelMaxValue)
140 local limeFactor = math.random(0,
    g_currentMission.limeCounterMaxValue)
141
142 for i = 1, table.getn(field.maxFieldStatusPartitions) do
143 self:setFieldPartitionStatus(field,
    field.maxFieldStatusPartitions, i, fruitIndex, fieldState,
    growthState, sprayFactor, false, plowFactor, weedValue,
    limeFactor)
144 end
145
146 index = index + 1
147 if index > self.fruitTypesCount then
148 index = 1
149 end
150 end
151 end
152 end)
153 elseif g_server ~= nil then
154 -- get current state of fields
```

```
155 for _, field in pairs(self.fields) do
156 g_deferredLoadingManager:addSubtask(function()
157 self:findFieldFruit(field)
158 end)
159 end
160 end
161
162 g_deferredLoadingManager:addSubtask(function()
163 self:findFieldSizes()
164 end)
165
166 g_deferredLoadingManager:addSubtask(function()
167 g_farmlandManager:addStateChangeListener(self)
168
169 if g_currentMission:getIsServer() then
170 if g_addCheatCommands then
171 addConsoleCommand("gsSetFieldFruit", "Sets a given fruit to
    field", "consoleCommandSetFieldFruit", self)
172 addConsoleCommand("gsSetFieldFruitAll", "Sets a given fruit to
    all fields", "consoleCommandSetFieldFruitAll", self)
173 addConsoleCommand("gsSetFieldGround", "Sets a given fruit to
    field", "consoleCommandSetFieldGround", self)
174 addConsoleCommand("gsSetFieldGroundAll", "Sets a given fruit to
    allfield", "consoleCommandSetFieldGroundAll", self)
175 end
176 end
177 end)
178
179 g_messageCenter:subscribe(MessageType.FARM_PROPERTY_CHANGED,
    self.farmPropertyChanged, self)
180 end
```

unloadMapData

Description

Unload data on mission delete

Definition

unloadMapData()

Code

```
184 function FieldManager:unloadMapData()
185 g_currentMission:removeUpdateable(self)
186 g_farmlandManager:removeStateChangeListener(self)
187
188 for _, field in pairs(self.fields) do
```

```
189  field:delete()
190  end
191
192  self.setFieldPartitionModifier = nil
193  self.detailModifier = nil
194  self.weedModifier = nil
195
196  g_messageCenter:unsubscribeAll(self)
197
198  removeConsoleCommand("gsSetFieldFruit")
199  removeConsoleCommand("gsSetFieldFruitAll")
200  removeConsoleCommand("gsSetFieldGround")
201  removeConsoleCommand("gsSetFieldGroundAll")
202
203
204  FieldManager:superClass().unloadMapData(self)
205  end
```

delete

Description

Deletes field manager

Definition

delete()

Code

```
209  function FieldManager:delete()
210  end
```

farmPropertyChanged

Description

Notified when a property of given farm changed. This could be the color which requires us to update the hotspots

Definition

farmPropertyChanged()

getFruitIndexForField

Description

The new fruit to plant in a field

Definition

getFruitIndexForField(Field Field)

Arguments

Field Field field to get a fruit for

Return Values

fruit type index to be planted

FieldUtil

Description

onCreate

Description

I3D-Callback to add a field

Definition

onCreate(integer id)

Arguments

integer id node id of the field element

getSprayFactor

Description

Returns spray factor of given field

Definition

getSprayFactor(table field)

Arguments

table field instance of an field

Return Values

float sprayFactor the spray factor of the given field

getPlowFactor

Description

Returns plow factor of given field

Definition

getPlowFactor(table field)

Arguments

table field instance of an field

Return Values

float plowFactor the plow factor of the given field

getLimeFactor

Description

Returns lime factor of given field

Definition

getLimeFactor(table field)

Arguments

table field instance of an field

Return Values

float plowFactor the lime factor of the given field

getWeedFactor

Description

Returns weed factor of given field

Definition

getWeedFactor(table field)

Arguments

table field instance of an field

Return Values

float plowFactor the weed factor of the given field

getMeasurementPositionOfField

Description

Get world position used to measure given field

Definition

getMeasurementPositionOfField()

getFruitArea

Description

Returns amount of fruit to work is in given area

Definition

getFruitArea(float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ, table terrainDetailRequiredValueRanges, table terrainDetailProhibitValueRanges, integer requiredfruittype, integer requiredMinGrowthState, integer requiredMaxGrowthState, integer prohibitedFruitType, integer prohibitedMinGrowthState, integer prohibitedMaxGrowthState, boolean useWindrowed)

Arguments

float	startWorldX	start world x
float	startWorldZ	start world z
float	widthWorldX	width world x
float	widthWorldZ	width world z
float	heightWorldX	height world x
float	heightWorldZ	height world z
table	terrainDetailRequiredValueRanges	terrain detail required value ranges
table	terrainDetailProhibitValueRanges	terrain detail prohibit value ranges
integer	requiredfruittype	required fruit type
integer	requiredMinGrowthState	required min growth state
integer	requiredMaxGrowthState	required max growth state
integer	prohibitedFruitType	prohibited fruit type
integer	prohibitedMinGrowthState	prohibited min growth state
integer	prohibitedMaxGrowthState	prohibited max growth state
boolean	useWindrowed	use windrow

Return Values

float area	area found
float totalArea	total area checked

Code

```
333 function FieldUtil.getFruitArea(startWorldX, startWorldZ,
    widthWorldX, widthWorldZ, heightWorldX, heightWorldZ,
    terrainDetailRequiredValueRanges,
    terrainDetailProhibitValueRanges, requiredFruitType,
    requiredMinGrowthState, requiredMaxGrowthState,
    prohibitedFruitType, prohibitedMinGrowthState,
    prohibitedMaxGrowthState, useWindrowed)
334 local query = g_currentMission.fieldCropsQuery
335
336 if requiredFruitType ~= FruitType.UNKNOWN then
```

```
local ids = g_currentMission.fruits[requiredFruitType]
if ids ~= nil and ids.id ~= 0 then
if useWindrowed then
return 0, 1
end
local desc = g_fruitTypeManager:getFruitTypeByIndex(requiredFruitType)
query:addRequiredCropType(ids.id, requiredMinGrowthState+1, requiredMaxGrowthState+1, desc.startStateChannel, desc.numStateChannels, g_currentMission.terrainDetailTypeFirstChannel, g_currentMission.terrainDetailTypeNumChannels)
end
end

if prohibitedFruitType ~= FruitType.UNKNOWN then
local ids = g_currentMission.fruits[prohibitedFruitType]
if ids ~= nil and ids.id ~= 0 then
local desc = g_fruitTypeManager:getFruitTypeByIndex(prohibitedFruitType)
query:addProhibitedCropType(ids.id, prohibitedMinGrowthState+1, prohibitedMaxGrowthState+1, desc.startStateChannel, desc.numStateChannels, g_currentMission.terrainDetailTypeFirstChannel, g_currentMission.terrainDetailTypeNumChannels)
end
end

for _,valueRange in pairs(terrainDetailRequiredValueRanges) do
query:addRequiredGroundValue(valueRange[1], valueRange[2], valueRange[3], valueRange[4])
end
for _,valueRange in pairs(terrainDetailProhibitValueRanges) do
query:addProhibitedGroundValue(valueRange[1], valueRange[2], valueRange[3], valueRange[4])
end
local x,z, widthX,widthZ, heightX,heightZ = MathUtil.getXZWidthAndHeight(startWorldX, startWorldZ, widthWorldX, widthWorldZ, heightWorldX, heightWorldZ)
return query:getParallelogram(x,z, widthX,widthZ, heightX,heightZ, true)
end
```

TransferMoneyDialog
Description
new

Description

Create a new TransferMoneyDialog instance.

Definition

new()

onClickActivate

Description

Handle "sell" button event.

Definition

onClickActivate()

ListElement

Description

List display element.
-- Layouts a list of ListItemElement instances which themselves can contain other elements. The list interacts with a
slider element for scrolling, if it is set up via configuration.
-- Use this list element for ordered displaying of a small to medium number of elements. For larger element counts or
more elaborate ordering logic, consider using the TableElement instead.
-- An important note:
- Even in a single column list, set both listItemHeight and listItemWidth. Both are always used to calculate mouse click targets.
--

onMouseDown

Description

Handles mouse button down event

Definition

onMouseDown()

onMouseUp

Description

Handles mouse button up (after down) event

Definition

onMouseUp()

getRowColumnForScreenPosition

Description

Get the list row and column indices for a given screen position.

Definition

getRowColumnForScreenPosition(posX Screen, posY Screen)

Arguments

posX Screen X position

posY Screen Y position

Return Values

row index, column index

updateItemPositionsInRange

Description

Update item positions for elements within a given item list range. The caller is responsible for index validity.

Definition

updateItemPositionsInRange(startIndex Starting, endIndex End)

Arguments

startIndex	Starting	index in self.listItems
endIndex	End	index in self.listItems

getItemCount

Description

Get the number of list items.

Definition

getItemCount()

Return Values

number of list items

getVisibleItemCount

Description

Get the number of visible items.

Definition

getVisibleItemCount()

Return Values

number of visible items

shouldFocusChange

Description

Determine if focus should change from this element in a given direction.
TODO: move navigation logic out of here, getter methods should never change object state!

Definition

shouldFocusChange(direction Focus)

Arguments

direction	Focus	navigation direction

ModHubController

Description

new

Description

Create a new ModHubController.

Definition

new(table messageCenter, table l10n, table storeManager, table brandManager, table fillTypeManager)

Arguments

table messageCenter	MessageCenter reference for local network UI event handling
table l10n	I18N reference
table storeManager	StoreManager reference for store item loading
table brandManager	BrandManager reference for brands loading
table fillTypeManager	FillTypeManager reference for fill type data access

Return Values

table fillType fillType object

load

Description

Load brands and items category data.

Definition

load()

Return Values

table fillType the fillType object

getCategories

Description

Get an array of categories

Definition

getCategories()

Return Values

string fillTypeName the fillType name

table categories list of categories

CutterEffectManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean isPlaying next part is playing

table instance instance of object

Code

```
19  function CutterEffectManager:new(customMt)
20  self = AbstractManager:new(customMt or CutterEffectManager_mt)
21
22  return self
23  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

float dtToUse dt to use

boolean stopAnimation stop animation

Code

```
27  function CutterEffectManager:initDataStructures()
```

```
28  self.nameToIndex = {}
29  self.cutterEffectTypes = {}
30  self.cutterEffects = {}
31  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
36  function CutterEffectManager:loadMapData()
37  CutterEffectManager:superClass().loadMapData(self)
38
39  self:addCutterEffectType("threshing")
40  self:addCutterEffectType("forage")
41  self:addCutterEffectType("center")
42  self:addCutterEffectType("left")
43  self:addCutterEffectType("right")
44
45  CutterEffectType = self.nameToIndex
46
47  return true
48  end
```

addCutterEffectType

Description

Adds a new cutterEffect type

Definition

addCutterEffectType(string name)

Arguments

string name name

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

getCutterEffectTypeByName

Description

Returns a cutterEffectType by name

Definition

getCutterEffectTypeByName(string name)

Arguments

string name name of cutterEffect type

Return Values

boolean	hasPrerequisite	true if all prerequisite specializations are loaded
string	cutterEffectType	the real cutterEffect name, nil if not defined

addCutterEffect

Description

Adds a new cutterEffect

Definition

addCutterEffect(integer fillTypeIndex, string materialType, integer materialIndex, integer materialId)

Arguments

integer	fillTypeIndex	filltype index
string	materialType	materialType
integer	materialIndex	material index
integer	materialId	internal material id

Return Values

boolean	detachAllowed	detach is allowed
string	warning	[optional] warning text to display

getCutterEffects

Description

Returns cutter effects for given properties

Definition

getCutterEffects(integer fruitTypeIndex, integer effectType, boolean isThreshing)

Arguments

integer	fruitTypeIndex	fruit type index
integer	effectType	effect type
boolean	isThreshing	is threshing

Return Values

table	cutterEffects	cutter effects

MaterialManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean	isLowered	implement chain is lowered
table	instance	instance of object

Code

```
19  function MaterialManager:new(customMt)
20  local self = AbstractManager:new(customMt or MaterialManager_mt)
21
22  return self
23  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

boolean inWorkPosition is in work position

Code

```
27  function MaterialManager:initDataStructures()
28  self.nameToIndex = {}
29  self.materialTypes = {}
30  self.materials = {}
31  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

float usage usage

boolean true if loading was successful else false

Code

```
36  function MaterialManager:loadMapData()
37  MaterialManager:superClass().loadMapData(self)
38
39  self:addMaterialType("fillplane")
40  self:addMaterialType("icon")
41  self:addMaterialType("unloading")
42  self:addMaterialType("smoke")
43  self:addMaterialType("straw")
44  self:addMaterialType("chopper")
45  self:addMaterialType("soil")
46  self:addMaterialType("sprayer")
47  self:addMaterialType("spreader")
48  self:addMaterialType("pipe")
49  self:addMaterialType("mower")
50  self:addMaterialType("belt")
51  self:addMaterialType("leveler")
52  self:addMaterialType("washer")
53  self:addMaterialType("pickup")
54
```

```
55  MaterialType = self.nameToIndex
56
57  return true
58  end
```

addMaterialType

Description

Adds a new material type

Definition

addMaterialType(string name)

Arguments

string name name

Return Values

boolean detachAllowed detach is allowed

string warning [optional] warning text to display

getMaterialTypeByName

Description

Returns a materialType by name

Definition

getMaterialTypeByName(string name)

Arguments

string name name of material type

Return Values

string materialType the real material name, nil if not defined

addMaterial

Description

Adds a new material type

Definition

addMaterial(integer fillTypeIndex, string materialType, integer materialIndex, integer materialId)

Arguments

integer fillTypeIndex filltype index

string materialType materialType

integer materialIndex material index

integer materialId internal material id

Return Values

boolean isActive input attacher is active

getMaterial

Description

Returns material for given properties

Definition

getMaterial(integer fillType, string materialTypeName, integer materialIndex)

Arguments

integer fillType fill type

string materialTypeName name of material type
integer materialIndex index of material

Return Values

boolean isOperating is operating
integer materialId id of material

MaterialUtil

Description

onCreateMaterial

Description

Called by material holder to create material

Definition

onCreateMaterial(any_type unused, integer id)

Arguments

any_type unused unused
integer id id

Return Values

boolean isReady is ready for ai work

onCreateParticleSystem

Description

Called by particle holder to create particle system

Definition

onCreateParticleSystem(any_type unused, integer id)

Arguments

any_type unused unused
integer id id

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

onCreateCutterEffect

Description

Called by cutter effect holder to create cutter effect

Definition

onCreateCutterEffect(any_type unused, integer id)

Arguments

any_type unused unused
integer id id

Return Values

boolean success success

ParticleSystemManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean success success

table instance instance of object

Code

```
19  function ParticleSystemManager:new(customMt)
20  local self = AbstractManager:new(customMt or
    ParticleSystemManager_mt)
21
22  return self
23  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

boolean success success

Code

```
27  function ParticleSystemManager:initDataStructures()
28  self.nameToIndex = {}
29  self.particleTypes = {}
30  self.particleSystems = {}
31  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean success success

boolean true if loading was successful else false

Code

```
36  function ParticleSystemManager:loadMapData()
37  ParticleSystemManager:superClass().loadMapData(self)
38
39  self:addParticleType("unloading")
40  self:addParticleType("smoke")
41  self:addParticleType("chopper")
42  self:addParticleType("straw")
43  self:addParticleType("cutter_chopper")
44  self:addParticleType("soil")
45  self:addParticleType("soil_smoke")
```

```
46  self:addParticleType("soil_chunks")
47  self:addParticleType("soil_big_chunks")
48  self:addParticleType("soil_harvesting")
49  self:addParticleType("spreader")
50  self:addParticleType("spreader_smoke")
51  self:addParticleType("windrower")
52  self:addParticleType("tedder")
53  self:addParticleType("weeder")
54  self:addParticleType("crusher_wood")
55  self:addParticleType("crusher_dust")
56  self:addParticleType("prepare_fruit")
57  self:addParticleType("cleaning_soil")
58  self:addParticleType("cleaning_dust")
59  self:addParticleType("washer_water")
60  self:addParticleType("chainsaw_wood")
61  self:addParticleType("chainsaw_dust")
62  self:addParticleType("pickup")
63  self:addParticleType("pickup_falling")
64  self:addParticleType("sowing")
65  self:addParticleType("loading")
66  self:addParticleType("driving_dust")
67  self:addParticleType("driving_dry")
68  self:addParticleType("driving_wet")
69
70  ParticleType = self.nameToIndex
71
72  return true
73  end
```

unloadMapData

Description

Unload data on mission delete

Definition

unloadMapData()

Return Values

boolean success success

Code

```
77  function ParticleSystemManager:unloadMapData()
78  for _, fillTypeParticleSystem in pairs(self.particleSystems) do
79  for _, ps in pairs(fillTypeParticleSystem) do
80  ParticleUtil.deleteParticleSystem(ps);
81  end
```

```
82  end
83
84  ParticleSystemManager:superClass().unloadMapData(self)
85  end
```

addParticleType

Description

Adds a new particle type

Definition

addParticleType(string name)

Arguments

string name name

Return Values

boolean possibleToDetach possible to detach selected implement

getParticleSystemTypeByName

Description

Returns a particleType by name

Definition

getParticleSystemTypeByName(string name)

Arguments

string name name of particle type

Return Values

integer index index of implement

string particleType the real particle name, nil if not defined

addParticleSystem

Description

Adds a new material type

Definition

addParticleSystem(integer fillTypeIndex, string particleType, integer materialIndex, integer materialId)

Arguments

integer fillTypeIndex filltype index

string particleType particleType

integer materialIndex material index

integer materialId internal material id

Return Values

table implement implement

getParticleSystem

Description

Returns particle system for given properties

Definition

getParticleSystem(integer fillType, string particleTypeName)

Arguments

integer fillType fill type

string particleTypeName name of particle type

Return Values

integer index index of implement

table particleSystem particleSystem

AbstractManager

Description

new

Description

Creating manager

Definition

new()

Return Values

float usage air usage

table instance instance of object

Code

```
17  function AbstractManager:new(customMt)
18  local self = setmetatable({}, customMt or AbstractManager_mt)
19
20  self:initDataStructures()
21  self.loadedMapData = false
22
23  return self
24  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

boolean detachAllowed detach is allowed

Code

```
28  function AbstractManager:initDataStructures()
29  end
```

load

Description

Loads initial manager

Definition

load()

Return Values

boolean isOutside is outside the rotation limit

boolean true if loading was successful else false

Code

```
34  function AbstractManager:load()
```

```
35  return true
36  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

boolean true if loading was successful else false

Code

```
41  function AbstractManager:loadMapData()
42  if g_isDevelopmentVersion and self.loadedMapData then
43  g_logManager:error("Manager map-data already loaded or not deleted after last game load!")
44  printCallstack()
45  end
46  self.loadedMapData = true
47  return true
48  end
```

unloadMapData

Description

Unload data on mission delete

Definition

unloadMapData()

Return Values

table bale bale

integer nearestBaleType id of bale type

Code

```
52  function AbstractManager:unloadMapData()
53  self.loadedMapData = false
54  self:initDataStructures()
55  end
```

AsyncManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```lua
function AsyncManager:new(customMt)
local self = setmetatable({}, customMt or AsyncManager_mt)

self:initDataStructures()

return self
end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

boolean allowsUnloading allows unloading

Code

```lua
function AsyncManager:initDataStructures()
self.firstTask = nil
self.lastTask = nil
self.currentRunningTask = nil
self.enabled = true
self.doTracing = false
self.executeSubTasksImmediately = self.doTracing and false -- reduces tracing spam when true
end
```

AutoSaveManager

Description

new

Description

Creating manager

Definition

new()

Return Values

string name name of bale grabber drop bale animation name

table instance instance of object

Code

```lua
function AutoSaveManager:new(customMt)
self = AbstractManager:new(customMt or AutoSaveManager_mt)

self.interval = 1000 * 60 * AutoSaveManager.DEFAULT_INTERVAL
self.time = self.interval

self.isPending = false
```

```
30  self.isActive = true
31
32  g_messageCenter:subscribe(MessageType.GUI_INGAME_OPEN,
    self.onOpenIngameMenu, self)
33  g_messageCenter:subscribe(MessageType.SAVEGAME_LOADED,
    self.onSavegameLoaded, self)
34
35  return self
36  end
```

BaleTypeManager
Description
new
Description

Creating manager

Definition

new()

Return Values

string name name of bale grabber drop bale animation name
table instance instance of object

Code

```
19  function BaleTypeManager:new(customMt)
20  local self = AbstractManager:new(customMt or BaleTypeManager_mt)
21  return self
22  end
```

initDataStructures
Description

Initialize data structures

Definition

initDataStructures()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
26  function BaleTypeManager:initDataStructures()
27  self.baleTypes = {}
28  self.nameToBaleType = {}
29  self.nameToIndex = {}
30  self.roundBales = {}
31  self.squareBales = {}
32
33  BaleType = self.nameToIndex
34  end
```

loadMapData
Description

Load data on map load

Definition

loadMapData()

Return Values

boolean allows allows bale grabbing

boolean true if loading was successful else false

Code

```
45  function BaleTypeManager:loadMapData(xmlFile, missionInfo, baseDirectory)
46      BaleTypeManager:superClass().loadMapData(self)
47
48      self:loadDefaultTypes(missionInfo)
49      return XMLUtil.loadDataFromMapXML(xmlFile, "baleTypes", baseDirectory, self, self.loadBaleTypes, missionInfo)
50  end
```

loadBaleTypeFromXML

Description

Loads and adds bale type from xml

Definition

loadBaleTypeFromXML(integer xmlFile, float key)

Arguments

integer xmlFile xml file id

float key xmlKey

Return Values

table baleType wrapped bale

table baleType baleType object

addBaleType

Description

Adds a new baleType

Definition

addBaleType(string name, float litersPerSecond)

Arguments

string name baleType index name

float litersPerSecond liter per second

Return Values

table nearestBale nearest bale

integer nearestBaleType id of bale type

table baleType baleType object

getBale

Description

Get bale

Definition

getBale(integer fillTypeIndex, float width, float height, float length, float diameter, boolean isRoundbale)

Arguments

integer	fillTypeIndex	fill type index
float	width	bale width
float	height	bale height
float	length	bale length
float	diameter	bale diameter
boolean	isRoundbale	is roundbale

Return Values

table bale bale

getBaleKey

Description

Get bale key

Definition

getBaleKey(string fillTypeName, boolean isRoundbale, float width, float height, float length, float diameter)

Arguments

string	fillTypeName	fill type name
boolean	isRoundbale	is roundbale
float	width	bale width
float	height	bale height
float	length	bale length
float	diameter	bale diameter

Return Values

boolean	allowsFold	allows folding
string	baleKey	bale key

BrandColorManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean	hasPrerequisite	true if all prerequisite specializations are loaded
table	instance	instance of object

Code

```
17  function BrandColorManager:new(customMt)
18  local self = AbstractManager:new(customMt or BrandColorManager_mt)
19  return self
20  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
24  function BrandColorManager:initDataStructures()
25  self.brandColors = {}
26  self:loadDefaultTypes()
27  end
```

loadBrandColorFromXML

Description

Loads and adds brand color from xml

Definition

loadBrandColorFromXML(integer xmlFile, float key)

Arguments

integer xmlFile xml file id

float key xmlKey

Return Values

int returns the CCT index

table baleType baleType object

addBrandColor

Description

Adds a new brand color

Definition

addBrandColor(string name, string value, boolean isBaseType)

Arguments

string name brand color name

string value value

boolean isBaseType is base type

Return Values

int returns the collision mask

table brandColor brandColor object

ConnectionHoseManager

Description

new

Description

Creating manager

Definition

new()

Return Values

float x position of center of CCT

float y position of center of CCT

float z position of center of CCT

table instance instance of object

Code

```
20  function ConnectionHoseManager:new(customMt)
21  local self = AbstractManager:new(customMt or
    ConnectionHoseManager_mt)
22
23  self:initDataStructures()
24
25  return self
26  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
30  function ConnectionHoseManager:initDataStructures()
31  self.typeByName = {}
32  ConnectionHoseType = self.typeByName
33  self.basicHoses = {}
34  self.sockets = {}
35  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
40  function ConnectionHoseManager:loadMapData(xmlFile, missionInfo,
    baseDirectory)
41  ConnectionHoseManager:superClass().loadMapData(self)
42  self:loadConnectionHosesFromXML(ConnectionHoseManager.DEFAULT_HOSES_FILEN
43  end
```

unloadMapData

Description

Load data on map load

Definition

unloadMapData()

Return Values

boolean allows allows threshing
boolean true if loading was successful else false

Code

```
48  function ConnectionHoseManager:unloadMapData()
49  for _, entry in ipairs(self.basicHoses) do
50  delete(entry.node)
51  end
52  for _, hoseType in pairs(self.typeByName) do
53  for _, adapter in pairs(hoseType.adapters) do
54  delete(adapter.node)
55  end
56  for _, hose in pairs(hoseType.hoses) do
57  delete(hose.materialNode)
58  end
59  end
60  for _, entry in pairs(self.sockets) do
61  delete(entry.node)
62  end
63
64  ConnectionHoseManager:superClass().unloadMapData(self)
65  end
```

loadConnectionHosesFromXML

Description

Loads fillTypes

Definition

loadConnectionHosesFromXML(table self, integer xmlFile, table missionInfo)

Arguments

table self target
integer xmlFile xml file handle
table missionInfo missionInfo

Return Values

boolean allow allow turn on
boolean success success

addConnectionHoseType

Description

Adds a new connection hose

Definition

addConnectionHoseType(string name)

Arguments

string name connectionHoseType index name

Return Values

string warningText turn on not allowed warning text

Code

```
268 function ConnectionHoseManager:addConnectionHoseType(name, desc)
269 name = name:upper()
270
271 if self.typeByName[name] == nil then
272 self.typeByName[name] = desc
273 else
274 print(string.format("Warning: connection hose type '%s' already exits!", name))
275 end
276 end
```

DeferredLoadingManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean allowsFold allows folding

table instance instance of object

Code

```
17 function DeferredLoadingManager:new(customMt)
18 local self = AsyncManager:new(customMt or DeferredLoadingManager_mt)
19
20 return self
21 end
```

FillTypeManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

table instance instance of object

Code

```
24 function FillTypeManager:new(customMt)
25 local self = AbstractManager:new(customMt or FillTypeManager_mt)
26 return self
27 end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
31  function FillTypeManager:initDataStructures()
32  self.fillTypes = {}
33  self.nameToFillType = {}
34  self.indexToFillType = {}
35  self.nameToIndex = {}
36  self.indexToName = {}
37
38  self.fillTypeConverters = {}
39  self.converterNameToIndex = {}
40  self.nameToConverter = {}
41
42  self.categories = {}
43  self.nameToCategoryIndex = {}
44  self.categoryIndexToFillTypes = {}
45  self.categoryNameToFillTypes = {}
46
47  self.fillTypeSamples = {}
48  self.fillTypeToSample = {}
49
50  FillType = self.nameToIndex
51  FillTypeCategory = self.categories
52  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

float steeringAngle adjusted steering angle

boolean true if loading was successful else false

Code

```
63  function FillTypeManager:loadMapData(xmlFile, missionInfo,
    baseDirectory)
64  FillTypeManager:superClass().loadMapData(self)
```

```
65
66  self:loadDefaultTypes()
67  return XMLUtil.loadDataFromMapXML(xmlFile, "fillTypes",
    baseDirectory, self, self.loadFillTypes, missionInfo,
    baseDirectory)
68  end
```

loadFillTypes

Description

Loads fillTypes

Definition

loadFillTypes(table self, integer xmlFile, table missionInfo)

Arguments

table	self	target
integer	xmlFile	xml file handle
table	missionInfo	missionInfo

Return Values

float	steeringAngle	adjusted steering angle
boolean	success	success

addFillType

Description

Adds a new fillType

Definition

addFillType(string name, string title, boolean showOnPriceTable, float pricePerLiter, float massPerLiter, float maxPhysicalSurfaceAngle, string hudOverlayFilename, string hudOverlayFilenameSmall, string customEnv)

Arguments

string	name	fillType index name
string	title	fillType full name
boolean	showOnPriceTable	show on price table
float	pricePerLiter	price per liter
float	massPerLiter	mass per liter
float	maxPhysicalSurfaceAngle	max surface angle
string	hudOverlayFilename	hud icon
string	hudOverlayFilenameSmall	hud icon small
string	customEnv	custom environment

Return Values

boolean	hasPrerequisite	true if all prerequisite specializations are loaded
table	fillType	fillType object

getFillTypeByIndex

Description

Gets a fillType by index

Definition

getFillTypeByIndex(integer index)

Arguments

integer index the fillType index

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

table fillType the fillType object

Code

```
310 function FillTypeManager:getFillTypeByIndex(index)
311 if index ~= nil then
312 return self.fillTypes[index]
313 end
314 return nil
315 end
```

getFillTypeNameByIndex

Description

Gets a fillTypeName by index

Definition

getFillTypeNameByIndex(integer index)

Arguments

integer index the fillType index

Return Values

boolean checkSpeedlimit check speed limit

string fillTypeName the fillType name

Code

```
321 function FillTypeManager:getFillTypeNameByIndex(index)
322 if index ~= nil then
323 return self.indexToName[index]
324 end
325 return nil
326 end
```

getFillTypeIndexByName

Description

Gets a fillType index by name

Definition

getFillTypeIndexByName(string name)

Arguments

string name the fillType index name

Return Values

boolean doGroundManipulation do ground manipulation

integer fillTypeIndex the fillType index

Code

```
332 function FillTypeManager:getFillTypeIndexByName(name)
333 if name ~= nil then
334 name = name:upper()
```

```
335     return self.nameToIndex[name]
336     end
337     return nil
338     end
```

getFillTypeByName

Description

Gets a fillType by index name

Definition

getFillTypeByName(string name)

Arguments

string name the fillType index name

Return Values

float dirtMultiplier current dirt multiplier

table fillType the fillType object

Code

```
344 function FillTypeManager:getFillTypeByName(name)
345     if ClassUtil.getIsValidIndexName(name) then
346         name = name:upper()
347         return self.nameToFillType[name]
348     end
349     return nil
350 end
```

getFillTypes

Description

Gets a list of fillTypes

Definition

getFillTypes()

Return Values

float dirtMultiplier current wear multiplier

table fillTypes list of fillTypes

Code

```
355 function FillTypeManager:getFillTypes()
356     return self.fillTypes
357 end
```

addFillTypeCategory

Description

Adds a new fillType category

Definition

addFillTypeCategory(string name)

Arguments

string name fillType category index name

Return Values

boolean	success	success
table	fillTypeCategory	fillType category object

addFillTypeToCategory

Description

Add fillType to category

Definition

addFillTypeToCategory(integer fillTypeIndex, integer categoryIndex)

Arguments

integer	fillTypeIndex	index of fillType
integer	categoryIndex	index of category

Return Values

float	speedLimit	speed limit
table	success	true if added else false

getFillTypesByCategoryNames

Description

Gets a list of fillTypes of the given category names

Definition

getFillTypesByCategoryNames(string name, string warning)

Arguments

string	name	fillType category index names
string	warning	a warning text shown if a category is not found

Return Values

boolean	hasPrerequisite	true if all prerequisite specializations are loaded
table	fillTypes	list of fillTypes

Code

```
407 function FillTypeManager:getFillTypesByCategoryNames(names, warning)
408 local fillTypes = {}
409 local alreadyAdded = {}
410 local categories = StringUtil.splitString(" ", names);
411 for _, categoryName in pairs(categories) do
412 categoryName = categoryName:upper()
413 local categoryFillTypes = self.categoryNameToFillTypes[categoryName];
414 if categoryFillTypes ~= nil then
415 for fillType, _ in pairs(categoryFillTypes) do
416 if alreadyAdded[fillType] == nil then
417 table.insert(fillTypes, fillType);
418 alreadyAdded[fillType] = true
419 end
420 end
421 else
```

```
422  if warning ~= nil then
423  print(string.format(warning, categoryName));
424  end
425  end
426  end
427  return fillTypes
428  end
```

getFillTypesByNames

Description

Gets list of fillTypes from string with fill type names

Definition

getFillTypesByNames(string fillTypes, string warning)

Arguments

string	fillTypes	fill types
string	warning	warning if fill type not found

Return Values

boolean	isAllowed	is allowed
table	fillTypes	fill types

addFillTypeConverter

Description

Adds a new fill type converter

Definition

addFillTypeConverter(string name)

Arguments

string	name	name

Return Values

boolean	detachAllowed	detach is allowed
integer	converterIndex	index of converterIndex

addFillTypeConversion

Description

Add fill type to fill type conversion

Definition

addFillTypeConversion(integer converter, integer sourceFillTypeIndex, integer targetFillTypeIndex, float conversionFactor)

Arguments

integer	converter	index of converter
integer	sourceFillTypeIndex	source fill type index
integer	targetFillTypeIndex	target fill type index
float	conversionFactor	factor of conversion

Return Values

float	wearMultiplier	current wear multiplier

getConverterDataByName

Description

Returns converter data by given name

Definition

getConverterDataByName(string converterName)

Arguments

string converterName name of converter

Return Values

boolean changed translation changed

table converterData converter data

getSampleByFillType

Description

Returns sound sample of fill type

Definition

getSampleByFillType(int fillType)

Arguments

int fillType fill type index

Return Values

boolean changed rotation changed

table sample sample

FruitTypeManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean changed animation changed

table instance instance of object

Code

```
29  function FruitTypeManager:new(customMt)
30  local self = AbstractManager:new(customMt or FruitTypeManager_mt)
31  return self
32  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

float state state of moving tool [0..1]

Code

```
36  function FruitTypeManager:initDataStructures()
37  self.fruitTypes = {}
```

```
38  self.indexToFruitType = {}
39  self.nameToIndex = {}
40  self.nameToFruitType = {}
41  self.fruitTypeIndexToFillType = {}
42  self.fillTypeIndexToFruitTypeIndex = {}
43
44  self.fruitTypeConverters = {}
45  self.converterNameToIndex = {}
46  self.nameToConverter = {}
47
48  self.windrowFillTypes = {}
49  self.fruitTypeIndexToWindrowFillTypeIndex = {}
50
51  self.numCategories = 0
52  self.categories = {}
53  self.indexToCategory = {}
54  self.categoryToFruitTypes = {}
55
56  self.weedFruitType = nil
57
58  FruitType = self.nameToIndex
59  FruitType.UNKNOWN = 0
60  FruitTypeCategory = self.categories
61  FruitTypeConverter = self.converterNameToIndex
62  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean hasPrerequisite	true if all prerequisite specializations are loaded
boolean true	if loading was successful else false

Code

```
74  function FruitTypeManager:loadMapData(xmlFile, missionInfo,
    baseDirectory)
75  FruitTypeManager:superClass().loadMapData(self)
76
77  self:loadDefaultTypes()
78  return XMLUtil.loadDataFromMapXML(xmlFile, "fruitTypes",
    baseDirectory, self, self.loadFruitTypes, missionInfo)
```

```
79 end
```

loadFruitTypes

Description

Loads fruitTypes

Definition

loadFruitTypes(table self, integer xmlFile)

Arguments

table	self	target
integer	xmlFile	xml file handle

Return Values

boolean	hasPrerequisite	true if all prerequisite specializations are loaded
boolean	success	success

addFruitType

Description

Adds a new fruitType

Definition

addFruitType(string name, boolean shownOnMap, boolean useForFieldJob)

Arguments

string	name	fruit index name
boolean	shownOnMap	show on map
boolean	useForFieldJob	use for field job

Return Values

boolean	success	success
table	fruitType	fruitType type object

loadFruitTypeWindrow

Description

Loads fruitType windrow data

Definition

loadFruitTypeWindrow(table fruitType, integer xmlFile, string key)

Arguments

table	fruitType	fruit type object
integer	xmlFile	xml file handle
string	key	xml key

Return Values

string	statesAttributes	states attributes

loadFruitTypeGrowth

Description

Loads fruitType growth data

Definition

loadFruitTypeGrowth(table fruitType, integer xmlFile, string key)

Arguments

table	fruitType	fruit type object
integer	xmlFile	xml file handle

string key xml key

Return Values

float distance distance

loadFruitTypeHarvest

Description

Loads fruitType harvest data

Definition

loadFruitTypeHarvest(table fruitType, integer xmlFile, string key)

Arguments

table fruitType fruit type object

integer xmlFile xml file handle

string key xml key

Return Values

string text text

loadFruitTypeCultivation

Description

Loads fruitType cultivation data

Definition

loadFruitTypeCultivation(table fruitType, integer xmlFile, string key)

Arguments

table fruitType fruit type object

integer xmlFile xml file handle

string key xml key

Return Values

table self instance of class event

loadFruitTypePreparing

Description

Loads fruitType preparing data

Definition

loadFruitTypePreparing(table fruitType, integer xmlFile, string key)

Arguments

table fruitType fruit type object

integer xmlFile xml file handle

string key xml key

Return Values

table self instance of class event

loadFruitTypeOptions

Description

Loads fruitType option data

Definition

loadFruitTypeOptions(table fruitType, integer xmlFile, string key)

Arguments

table fruitType fruit type object

integer xmlFile xml file handle
string key xml key

Return Values

table self instance of class event

loadFruitTypeWeedData

Description

Loads fruitType generic updater data

Definition

loadFruitTypeWeedData(table fruitType, integer xmlFile, string key)

Arguments

table fruitType fruit type object
integer xmlFile xml file handle
string key xml key

Return Values

table self instance of class event

loadFruitTypeMapColors

Description

Load fruit type map overlay color data.

Definition

loadFruitTypeMapColors(table fruitType, integer xmlFile, string key)

Arguments

table fruitType fruit type object
integer xmlFile xml file handle
string key xml key

Return Values

table self instance of class event

loadFruitTypeDestruction

Description

Load fruit type map overlay color data.

Definition

loadFruitTypeDestruction(table fruitType, integer xmlFile, string key)

Arguments

table fruitType fruit type object
integer xmlFile xml file handle
string key xml key

Return Values

table self instance of class event

getFruitTypeByIndex

Description

Gets a fruitType by index

Definition

getFruitTypeByIndex(integer index)

Arguments

integer index the fruit index

Return Values

table self instance of class event

table fruit the fruit object

Code

```
500 function FruitTypeManager:getFruitTypeByIndex(index)
501 if index ~= nil then
502 return self.indexToFruitType[index]
503 end
504 return nil
505 end
```

getFruitTypeByName

Description

Gets a fruitType by index name

Definition

getFruitTypeByName(string name)

Arguments

string name the fruit index name

Return Values

table self instance of class event

table fruit the fruit object

Code

```
518 function FruitTypeManager:getFruitTypeByName(name)
519 if name ~= nil then
520 name = name:upper()
521 return self.nameToFruitType[name]
522 end
523 return nil
524 end
```

getFruitTypes

Description

Gets a list of fruitTypes

Definition

getFruitTypes()

Return Values

table self instance of class event

table fruitTypes a list of fruitTypes

Code

```
529 function FruitTypeManager:getFruitTypes()
530 return self.fruitTypes
531 end
```

addFruitTypeCategory

Description

Adds a new fruitType category

Definition

addFruitTypeCategory(string name)

Arguments

string name fruit category index name

Return Values

table self instance of class event

table fruitTypeCategory fruitType category object

addFruitTypeToCategory

Description

Add fruitType to category

Definition

addFruitTypeToCategory(integer fruitTypeIndex, integer categoryIndex)

Arguments

integer fruitTypeIndex index of fruit type

integer categoryIndex index of category

Return Values

table self instance of class event

table success true if added else false

getFruitTypesByCategoryNames

Description

Gets a list of fruitTypes of the given category names

Definition

getFruitTypesByCategoryNames(string name, string warning)

Arguments

string name fruitType category index names

string warning a warning text shown if a category is not found

Return Values

table self instance of class event

table fruitTypes list of fruitTypes

Code

```
612 function FruitTypeManager:getFruitTypesByCategoryNames(names, warning)
613 local fruitTypes = {}
614 local alreadyAdded = {}
615 local categories = StringUtil.splitString(" ", names)
616 for _, categoryName in pairs(categories) do
617 categoryName = categoryName:upper()
618 local categoryIndex = self.categories[categoryName]
619 local categoryFruitTypes = self.categoryToFruitTypes[categoryIndex]
620 if categoryFruitTypes ~= nil then
621 for fruitType, _ in pairs(categoryFruitTypes) do
```

```
622  if alreadyAdded[fruitType] == nil then
623  table.insert(fruitTypes, fruitType)
624  alreadyAdded[fruitType] = true
625  end
626  end
627  else
628  if warning ~= nil then
629  print(string.format(warning, categoryName))
630  end
631  end
632  end
633  return fruitTypes
634  end
```

getFruitTypesByNames

Description

Gets list of fruitTypes from string with fruit type names

Definition

getFruitTypesByNames(string fruitTypes, string warning)

Arguments

string fruitTypes fruit types
string warning warning if fruit type not found

Return Values

table self instance of class event
table fruitTypes fruit types

getFillTypesByFruitTypeNames

Description

Gets a list if fillType from string with fruit type names

Definition

getFillTypesByFruitTypeNames(string names, string warning)

Arguments

string names fruit type names
string warning warning if fill type not found

Return Values

table self instance of class event
table fillTypes fill types

getFillTypesByFruitTypeCategoryName

Description

Gets a list of fillTypes from string with fruit type category names

Definition

getFillTypesByFruitTypeCategoryName(string fruitTypeCategories, string warning)

Arguments

string fruitTypeCategories fruit type categories

string warning warning if category not found

Return Values

table self instance of class event

table fillTypes fill types

getFillTypeLiterPerSqm

Description

Get fill type liter per sqm

Definition

getFillTypeLiterPerSqm(integer fillType, float defaultValue)

Arguments

integer fillType fill type

float defaultValue default value if fill type not found

Return Values

table self instance of class event

float literPerSqm liter per sqm

addFruitTypeConverter

Description

Adds a new fruit type converter

Definition

addFruitTypeConverter(string name)

Arguments

string name name

Return Values

table self instance of class event

integer converterIndex index of converterIndex

addFruitTypeConversion

Description

Add fruit type to fill type conversion

Definition

addFruitTypeConversion(integer converter, integer fruitTypeIndex, integer fillTypeIndex, float conversionFactor, float windrowConversionFactor)

Arguments

integer converter index of converter

integer fruitTypeIndex fruit type index

integer fillTypeIndex fill type index

float conversionFactor factor of conversion

float windrowConversionFactor factor of windrow conversion

Return Values

table self instance of class event

getConverterDataByName

Description

Returns converter data by given name

Definition

getConverterDataByName(string converterName)

Arguments

string converterName name of converter

Return Values

table self instance of class event

table converterData converter data

GameplayHintManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table self instance of class event

table instance instance of object

Code

```
17  function GameplayHintManager:new(customMt)
18  local self = AbstractManager:new(customMt or
    GameplayHintManager_mt)
19
20  return self
21  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

table self instance of class event

Code

```
25  function GameplayHintManager:initDataStructures()
26  self.gameplayHints = {}
27  self.isLoaded = false
28  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

table self instance of class event

boolean true if loading was successful else false

Code

```
33  function GameplayHintManager:loadMapData(xmlFile, missionInfo)
34  GameplayHintManager:superClass().loadMapData(self)
35
36  local filename = Utils.getFilename(getXMLString(xmlFile,
    "map.gameplayHints#filename"), g_currentMission.baseDirectory)
37  if filename == nil or filename == "" then
38  print("Error: Could not load gameplayHint config file
    '"..tostring(filename).."'!")
39  return false
40  end
41
42  local gameplayHintXmlFile = loadXMLFile("gameplayHints", filename)
43  local i = 0
44  while true do
45  local key = string.format("gameplayHints.gameplayHint(%d)", i)
46  if not hasXMLProperty(gameplayHintXmlFile, key) then
47  break
48  end
49
50  local text = getXMLString(gameplayHintXmlFile, key)
51  if text:sub(1,6) == "$l10n_" then
52  text = g_i18n:getText(text:sub(7), missionInfo.customEnvironment)
53  end
54  table.insert(self.gameplayHints, text)
55
56  i = i + 1
57  end
58  delete(gameplayHintXmlFile)
59
60  self.isLoaded = true
61
62  return true
63  end
```

getRandomGameplayHint

Description

Gets random gamplay hints

Definition

getRandomGameplayHint(integer numHints)

Arguments

integer numHints number of hints

Return Values

table self instance of class event

table gameplayHintGroup a random gameplay hint group

Code

```
69  function GameplayHintManager:getRandomGameplayHint(numberOfHints)
70  local hints = {}
71  local addedHints = {}
72
73  local numHints = #self.gameplayHints
74  while #hints < numberOfHints do
75  local hintId = math.random(1, numHints)
76  if addedHints[hintId] == nil then
77  table.insert(hints, self.gameplayHints[hintId])
78  addedHints[hintId] = hintId
79  end
80  end
81
82  return hints
83  end
```

GroundTypeManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table self instance of class event

table instance instance of object

Code

```
17  function GroundTypeManager:new(customMt)
18  local self = AbstractManager:new(customMt or GroundTypeManager_mt)
19  return self
20  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

table self instance of class event

Code

```
24  function GroundTypeManager:initDataStructures()
25  self.groundTypes = {}
26  self.groundTypeMappings = {}
27  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

table self instance of class event

boolean true if loading was successful else false

Code

```
58  function GroundTypeManager:loadMapData(xmlFile, missionInfo,
    baseDirectory)
59  GroundTypeManager:superClass().loadMapData(self)
60  self:loadGroundTypes()
61  return XMLUtil.loadDataFromMapXML(xmlFile, "groundTypeMappings",
    baseDirectory, self, self.loadGroundTypeMappings, missionInfo)
62  end
```

loadGroundTypeMappings

Description

Loads the groundTypeMappings from the map xml

Definition

loadGroundTypeMappings(entity xmlFile, table missionInfo, entity xmlFile)

Arguments

entity xmlFile xml file handle

table missionInfo missionInfo table

entity xmlFile xml file handle

Return Values

table self instance of class event

boolean success success

initTerrain

Description

Initialize the ground type manager with the data from the terrain

Definition

initTerrain(entity terrainRootNode)

Arguments

entity terrainRootNode the root node of the terrain

Return Values

table self instance of class event

Code

```
94  function GroundTypeManager:initTerrain(terrainRootNode)
```

```
 95  self.terrainLayerMappping = {}
 96  local numLayers = getTerrainNumOfLayers(terrainRootNode)
 97  for i=0,numLayers-1 do
 98  local layerName = getTerrainLayerName(terrainRootNode, i)
 99  self.terrainLayerMappping[layerName] = i
100  end
101  end
```

HelpLineManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table self instance of class event

table instance instance of object

Code

```
21  function HelpLineManager:new(customMt)
22  local self = AbstractManager:new(customMt or HelpLineManager_mt)
23
24  return self
25  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

table self instance of class event

Code

```
29  function HelpLineManager:initDataStructures()
30  self.categories = {}
31  self.categoryNames = {}
32  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

table self instance of class event

boolean true if loading was successful else false

Code

```
37  function HelpLineManager:loadMapData(xmlFile, missionInfo)
38  HelpLineManager:superClass().loadMapData(self)
39
40  local filename = Utils.getFilename(getXMLString(xmlFile,
    "map.helpline#filename"), g_currentMission.baseDirectory)
41  if filename == nil or filename == "" then
42  print("Error: Could not load helpline config file
    '"..tostring(filename).."'!")
43  return false
44  end
45
46  self:loadFromXML(filename, missionInfo)
47
48  return true
49  end
```

loadCategory

Description

Load a category and its items (pages)

Definition

loadCategory()

Return Values

table self instance of class event

loadPage

Description

Load a single helpline element (page)

Definition

loadPage()

Return Values

table self instance of class event

convertText

Description

Convert text to the proper language with the correct template replacements

Definition

convertText()

Return Values

table self instance of class event

getCategories

Description

Gets a list of all categories

Definition

getCategories()

Return Values

table self instance of class event

table categories a list of categories

Code

```
136  function HelpLineManager:getCategories()
137  return self.categories
138  end
```

getCategory

Description

Gets a category by category index

Definition

getCategory(integer categoryIndex)

Arguments

integer categoryIndex category index

Return Values

table self instance of class event

table category the corresponding category

Code

```
144  function HelpLineManager:getCategory(categoryIndex)
145  if categoryIndex ~= nil then
146  return self.categories[categoryIndex]
147  end
148  return nil
149  end
```

LogManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table self instance of class event

table instance instance of object

Code

```
17  function LogManager:new(customMt)
18  local self = AbstractManager:new(customMt or LogManager_mt)
19
20  return self
21  end
```

xmlWarning

Description

Prints a xml warning to console and logfile

Definition

xmlWarning(string xmlFilename, string warningMessage, List params)

Arguments

string xmlFilename the xml filename
string warningMessage the warning message. Can contain string-format placeholders
List params variable number of parameters. Depends on placeholders in warning message

Return Values

table self instance of class event

xmlError

Description

Prints a xml error to console and logfile

Definition

xmlError(string xmlFilename, string errorMessage, List params)

Arguments

string xmlFilename the xml filename
string errorMessage the error message. Can contain string-format placeholders
List params variable number of parameters. Depends on placeholders in error message

Return Values

table self instance of class event

xmlInfo

Description

Prints a xml info to console and logfile

Definition

xmlInfo(string xmlFilename, string infoMessage, List params)

Arguments

string xmlFilename the xml filename
string infoMessage the warning message. Can contain string-format placeholders
List params variable number of parameters. Depends on placeholders in warning message

Return Values

table self instance of class event

xmlDevWarning

Description

Prints a xml development warning to console and logfile

Definition

xmlDevWarning(string xmlFilename, string warningMessage, List params)

Arguments

string xmlFilename the xml filename
string warningMessage the warning message. Can contain string-format placeholders
List params variable number of parameters. Depends on placeholders in warning message

Return Values

table self instance of class event

xmlDevError

Description

Prints a xml development error to console and logfile

Definition

xmlDevError(string xmlFilename, string errorMessage, List params)

Arguments

string xmlFilename the xml filename
string errorMessage the error message. Can contain string-format placeholders
List params variable number of parameters. Depends on placeholders in error message

Return Values

table self instance of class event

xmlDevInfo

Description

Prints a xml development info to console and logfile

Definition

xmlDevInfo(string xmlFilename, string infoMessage, List params)

Arguments

string xmlFilename the xml filename
string infoMessage the info message. Can contain string-format placeholders
List params variable number of parameters. Depends on placeholders in info message

Return Values

table self instance of class event

warning

Description

Prints a warning to console and logfile

Definition

warning(string warningMessage, List params)

Arguments

string warningMessage the warning message. Can contain string-format placeholders
List params variable number of parameters. Depends on placeholders in warning message

Return Values

table self instance of class event

error

Description

Prints an error to console and logfile

Definition

error(string errorMessage, List params)

Arguments

string errorMessage the warning message. Can contain string-format placeholders
List params variable number of parameters. Depends on placeholders in warning message

Return Values

table self instance of class event

info

Description

Prints an info to console and logfile

Definition

info(string infoMessage, List params)

Arguments

string infoMessage the warning message. Can contain string-format placeholders

List params variable number of parameters. Depends on placeholders in warning message

Return Values

table self instance of class event

devWarning

Description

Prints a development warning to console and logfile

Definition

devWarning(string warningMessage, List params)

Arguments

string warningMessage the warning message. Can contain string-format placeholders

List params variable number of parameters. Depends on placeholders in warning message

Return Values

table self instance of class event

devError

Description

Prints a development error to console and logfile

Definition

devError(string errorMessage, List params)

Arguments

string errorMessage the warning message. Can contain string-format placeholders

List params variable number of parameters. Depends on placeholders in warning message

Return Values

table self instance of class event

devInfo

Description

Prints a development info to console and logfile

Definition

devInfo(string infoMessage, List params)

Arguments

string infoMessage the warning message. Can contain string-format placeholders

List params variable number of parameters. Depends on placeholders in warning message

Return Values

table self instance of class event

MapManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

table instance instance of object

Code

```
17  function MapManager:new(customMt)
18  local self = AbstractManager:new(customMt or MapManager_mt)
19
20  return self
21  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Code

```
25  function MapManager:initDataStructures()
26  self.maps = {}
27  self.idToMap = {}
28  end
```

addMapItem

Description

Adds new map item

Definition

addMapItem(string id, string scriptFilename, string className, string configFile, string defaultVehiclesXMLFilename, string title, string description, string iconFilename, string baseDirectory, table customEnvironment, boolean isMultiplayerSupported, boolean isModMap)

Arguments

string	id	id
string	scriptFilename	script filename
string	className	mission class name
string	configFile	map config file
string	defaultVehiclesXMLFilename	map default vehicles file
string	title	title
string	description	description text
string	iconFilename	map icon filename
string	baseDirectory	map base directory
table	customEnvironment	custom environment
boolean	isMultiplayerSupported	multiplayer support flag

boolean	isModMap	is mod map flag

Return Values

boolean	hasPrerequisite	true if all prerequisite specializations are loaded

loadMapFromXML

Description

Loads map from xml file and adds map item

Definition

loadMapFromXML(integer xmlFile, string baseName, string modDir, string modName, boolean isMultiplayerSupported, boolean isDLCFile)

Arguments

integer	xmlFile	id of xml file
string	baseName	base name
string	modDir	path to mod directory
string	modName	mod name
boolean	isMultiplayerSupported	multiplayer support flag
boolean	isDLCFile	is DLC flag

Return Values

boolean	detachAllowed	detach is allowed
string	warning	[optional] warning text to display

ModManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table	instance	instance of object

Code

```
17  function ModManager:new(customMt)
18  local self = AbstractManager:new(customMt or ModManager_mt)
19
20  return self
21  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

boolean	detachAllowed	detach is allowed
string	warning	[optional] warning text to display

Code

```
25  function ModManager:initDataStructures()
26  self.hashToMod = {}
27  self.nameToMod = {}
28  self.validMods = {}
29  self.multiplayerMods = {}
30  self.mods = {}
31  self.numMods = 0
32  end
```

addMod

Description

Adds a new mod

Definition

addMod(string name, string description, string version, string modDescVersion, string author, string iconFilename, string modName, string modDir, string modFile, boolean isMultiplayerSupported, string fileHash, string absBaseFilename, boolean isDirectory, boolean isDLC)

Arguments

string	name	title mod title
string	description	mod description
string	version	mod version
string	modDescVersion	mod desc version
string	author	mod author
string	iconFilename	mod icon filenamepath
string	modName	mod name
string	modDir	mod directory
string	modFile	mod file name
boolean	isMultiplayerSupported	defines if mod can be used in mp
string	fileHash	mod file md5 hash
string	absBaseFilename	absolute file path
boolean	isDirectory	true if directory, false if zip
boolean	isDLC	true if mod is a dlc

Return Values

table mod the mod object

Code

```
51  function ModManager:addMod(title, description, version,
    modDescVersion, author, iconFilename, modName, modDir, modFile,
    isMultiplayerSupported, fileHash, absBaseFilename, isDirectory,
    isDLC, isPreorderBonus)
52  if fileHash ~= nil and self.hashToMod[fileHash] ~= nil then
53  print("Error: Adding mod with same file hash twice. Title is " ..
    title .. " filehash: " .. fileHash)
54  return nil
55  end
56
```

```
57  self.numMods = self.numMods + 1
58
59  local mod = {}
60  mod.id = self.numMods
61  mod.title = title
62  mod.description = description
63  mod.version = version
64  mod.modDescVersion = modDescVersion
65  mod.author = author
66  mod.iconFilename = iconFilename
67  mod.isDLC = isDLC
68  mod.fileHash = fileHash
69  mod.modName = modName
70  mod.modDir = modDir
71  mod.modFile = modFile
72  mod.absBaseFilename = absBaseFilename
73  mod.isDirectory = isDirectory
74  mod.isMultiplayerSupported = isMultiplayerSupported
75  mod.isPreorderBonus = isPreorderBonus
76
77  table.insert(self.mods, mod)
78  self.nameToMod[modName] = mod
79  if fileHash ~= nil then
80  table.insert(self.validMods, mod)
81  self.hashToMod[fileHash] = mod
82  if isMultiplayerSupported then
83  table.insert(self.multiplayerMods, mod)
84  end
85  end
86
87  return mod
88  end
```

removeMod

Description

Removes a mod

Definition

removeMod(table mod)

Arguments

table mod the mod object

Return Values

boolean isActive input attacher is active

boolean success true if mod was removed, else false

Code

```
 94  function ModManager:removeMod(mod)
 95  if mod ~= nil then
 96  self.nameToMod[mod.modName] = nil
 97  if mod.fileHash ~= nil then
 98  self.hashToMod[mod.fileHash] = nil
 99  end
100
101  for index, modItem in ipairs(self.mods) do
102  if modItem == mod then
103  table.remove(self.mods, index)
104  break
105  end
106  end
107
108  for index, modItem in ipairs(self.validMods) do
109  if modItem == mod then
110  table.remove(self.validMods, index)
111  break
112  end
113  end
114
115  for index, modItem in ipairs(self.multiplayerMods) do
116  if modItem == mod then
117  table.remove(self.multiplayerMods, index)
118  break
119  end
120  end
121
122  return true
123  end
124
125  return false
126  end
```

getModByFileHash

Description

Gets a mod by filehash

Definition

getModByFileHash(string fileHash)

Arguments

string fileHash the file md5 hash

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

table mod the mod object

Code

```
132 function ModManager:getModByFileHash(fileHash)
133 return self.hashToMod[fileHash]
134 end
```

getModByName

Description

Gets a mod by name

Definition

getModByName(string modName)

Arguments

string modName the mod name

Return Values

boolean success success

table mod the mod object

Code

```
140 function ModManager:getModByName(modName)
141 return self.nameToMod[modName]
142 end
```

getModByIndex

Description

Gets a mod by indea

Definition

getModByIndex(integer index)

Arguments

integer index the mod index

Return Values

boolean doGroundManipulation do ground manipulation

table mod the mod object

Code

```
148 function ModManager:getModByIndex(index)
149 return self.mods[index]
150 end
```

getMods

Description

Gets a list of all mods

Definition

getMods()

Return Values

boolean checkSpeedlimit check speed limit

table	mods	a list of mods

Code

```
155  function ModManager:getMods()
156  return self.mods
157  end
```

getMultiplayerMods

Description

Gets a list of all multiplayer mods

Definition

getMultiplayerMods()

Return Values

float	speedLimit	speed limit
table	mods	a list of multiplayer mods

Code

```
162  function ModManager:getMultiplayerMods()
163  return self.multiplayerMods
164  end
```

getNumOfMods

Description

Gets total number of mods

Definition

getNumOfMods()

Return Values

boolean	hasPrerequisite	true if all prerequisite specializations are loaded
integer	numMods	number of mods

Code

```
169  function ModManager:getNumOfMods()
170  return #self.mods
171  end
```

getNumOfValidMods

Description

Gets total number of valid mods

Definition

getNumOfValidMods()

Return Values

boolean	hasPrerequisite	true if all prerequisite specializations are loaded
integer	numMods	number of valid mods

Code

```
176  function ModManager:getNumOfValidMods()
177  return #self.validMods
178  end
```

getAreAllModsAvailable

Description

Checks if all given mod hashes are available

Definition

getAreAllModsAvailable(table modHashs)

Arguments

table modHashs a list of modhashes

Return Values

boolean tippingAllowed tipping is allowed

boolean areAvailable true if all hashes are available else false

Code

```
184 function ModManager:getAreAllModsAvailable(modHashs)
185 for _, modHash in pairs(modHashs) do
186 if not self:getIsModAvailable(modHash) then
187 return false
188 end
189 end
190
191 return true
192 end
```

getIsModAvailable

Description

Checks if given hash is available

Definition

getIsModAvailable(table modHash)

Arguments

table modHash a mod hash

Return Values

boolean allowsFold allows folding

boolean isAvailable true if hash is available else false

Code

```
198 function ModManager:getIsModAvailable(modHash)
199 local modItem = self.hashToMod[modHash]
200 if modItem == nil or not modItem.isMultiplayerSupported then
201 return false
202 end
203
204 return true
205 end
```

isModMap

Description

Determines if given mod is a map-mod

Definition

isModMap(string modName)

Arguments

string modName Name of the mod

Return Values

boolean detachAllowed detach is allowed

PlaceableTypeManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean consume consumePtoPower

table instance instance of object

Code

```
17 function PlaceableTypeManager:new(customMt)
18 local self = AbstractManager:new(customMt or PlaceableTypeManager_mt)
19
20 return self
21 end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

float rpm rpm of pto

Code

```
25 function PlaceableTypeManager:initDataStructures()
26 self.placeableTypes = {}
27 end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

boolean true if loading was successful else false

Code

```
32 function PlaceableTypeManager:loadMapData()
33 PlaceableTypeManager:superClass().loadMapData(self)
```

```
34
35  local xmlFile = loadXMLFile("PlaceableTypesXML",
    "dataS/placeableTypes.xml")
36  local i=0
37  while true do
38  local baseName = string.format("placeableTypes.placeableType(%d)",
    i)
39
40  local typeName = getXMLString(xmlFile, baseName.. "#name")
41  if typeName == nil then
42  break
43  end
44  local className = getXMLString(xmlFile, baseName.. "#className")
45  local filename = getXMLString(xmlFile, baseName.. "#filename")
46
47  self:addPlaceableType(typeName, className, filename, "")
48  i = i+1
49  end
50  delete(xmlFile)
51
52  print(" Loaded placeable types")
53
54  return true
55  end
```

addPlaceableType

Description

Adds a new placeableType

Definition

addPlaceableType(string typeName, string className, string filename, string customEnvironment)

Arguments

string typeName	placeable type name
string className	class name
string filename	filename
string customEnvironment	customEnvironment

Return Values

boolean success success

boolean success true if added else false

PreorderBonusManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean isActive pickup node is active

table instance instance of object

Code

```
19  function PreorderBonusManager:new(customMt)
20  self = AbstractManager:new(customMt or PreorderBonusManager_mt)
21
22  self.spawnedMods = {}
23  self.modsToSpawn = {}
24
25  self.timer = 0
26
27  return self
28  end
```

SleepManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

table instance instance of object

Code

```
20  function SleepManager:new(customMt)
21  self = AbstractManager:new(customMt or SleepManager_mt)
22
23  self.isSleeping = false
24  self.wakeUpTime = 0
25
26  self.sleepingRanges = {}
27  self.sleepingRanges[1] = {19, 24}
28  self.sleepingRanges[2] = {0, 5}
29
30  self.requestedSleep = false
31  self.requestedTime = 0
32  self.requestCounter = 0
33  self.responseCounter = 0
34
```

```
35  return self
36  end
```

update

Description

Initialize data structures

Definition

update()

Return Values

boolean isActive is active for lights

Code

```
40  function SleepManager:update(dt)
41  if self.wakeUpTime < g_time and self.isSleeping then
42  self:stopSleep()
43  end
44
45  if self.requestedSleep then
46  if self.responseCounter == self.requestCounter then
47  self:startSleep(self.duration)
48  self.responseCounter = 0
49  self.requestedSleep = false
50  end
51
52  if self.requestedTime + SleepManager.TIME_TO_ANSWER_REQUEST <
    g_time then
53  self.responseCounter = 0
54  self.requestedSleep = false
55  end
56  end
57  end
```

SplitTypeManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean canBeToggled lights can be toggled

table instance instance of object

Code

```
19  function SplitTypeManager:new(customMt)
20  local self = AbstractManager:new(customMt or SplitTypeManager_mt)
```

```
21
22  return self
23  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

boolean highProfileUsed high profile is used

Code

```
27  function SplitTypeManager:initDataStructures()
28  self.typesByIndex = {}
29  end
```

loadMapData

Description

Loads initial manager

Definition

loadMapData()

Return Values

boolean changed mask has changed

boolean true if loading was successful else false

Code

```
34  function SplitTypeManager:loadMapData()
35  SplitTypeManager:superClass().loadMapData(self)
36  self:addSplitType("spruce", 1, 0.7, 7.0, true); -- density 0.47
37  self:addSplitType("pine", 2, 0.7, 7.0, true); -- density 0.52
38  self:addSplitType("larch", 3, 0.7, 7.0, true); -- density 0.59
39  self:addSplitType("birch", 4, 0.85, 7.2, false); -- density 0.65
40  self:addSplitType("beech", 5, 0.9, 7.4, false); -- density 0.69
41  self:addSplitType("maple", 6, 0.9, 7.4, false); -- density 0.65
42  self:addSplitType("oak", 7, 0.9, 7.4, false); -- density 0.67
43  self:addSplitType("ash", 8, 0.9, 7.4, false); -- density 0.69
44  self:addSplitType("locust", 9, 1.0, 7.8, false); -- density 0.73
45  self:addSplitType("mahogany", 10, 1.1, 8.0, false); -- density 0.8
46  self:addSplitType("poplar", 11, 0.7, 7.5, false); -- density 0.48
47
48  return true
49  end
```

addSplitType

Description

Adds a new baleType

Definition

addSplitType(string name, float litersPerSecond)

Arguments

string name baleType index name
float litersPerSecond liter per second

Return Values

integer lightsTypesMask light types mask
table baleType baleType object

getSplitTypeByIndex

Description

Returns tool type index by given name

Definition

getSplitTypeByIndex(string toolTypeName)

Arguments

string toolTypeName tool type name

Return Values

boolean changed visibility has changed
integer toolTypeIndex tool type index

Code

```
71  function SplitTypeManager:getSplitTypeByIndex(index)
72  if self.typesByIndex[index] ~= nil then
73  return self.typesByIndex[index]
74  end;
75  end;
```

SprayTypeManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean state beacon light state
table instance instance of object

Code

```
19  function SprayTypeManager:new(customMt)
20  local self = AbstractManager:new(customMt or SprayTypeManager_mt)
21  return self
22  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

boolean changed state has changed

Code

```
26  function SprayTypeManager:initDataStructures()
27  self.numSprayTypes = 0
28  self.sprayTypes = {}
29  self.nameToSprayType = {}
30  self.nameToIndex = {}
31  self.indexToName = {}
32  self.fillTypeIndexToSprayType = {}
33
34  SprayType = self.nameToIndex
35  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

integer state turn light state

boolean true if loading was successful else false

Code

```
46  function SprayTypeManager:loadMapData(xmlFile, missionInfo,
    baseDirectory)
47  SprayTypeManager:superClass().loadMapData(self)
48  self:loadDefaultTypes()
49  return XMLUtil.loadDataFromMapXML(xmlFile, "sprayTypes",
    baseDirectory, self, self.loadSprayTypes, missionInfo)
50  end
```

loadSprayTypes

Description

Load data on map load

Definition

loadSprayTypes()

Return Values

boolean changed visibility has changed

boolean true if loading was successful else false

Code

```
55  function SprayTypeManager:loadSprayTypes(xmlFile, missionInfo,
    isBaseType)
56  local i = 0
57  while true do
```

```
58  local key = string.format("map.sprayTypes.sprayType(%d)", i)
59  if not hasXMLProperty(xmlFile, key) then
60  break
61  end
62
63  local name = getXMLString(xmlFile, key.."#name")
64  local litersPerSecond = getXMLFloat(xmlFile,
    key.."#litersPerSecond")
65  local typeName = getXMLString(xmlFile, key.."#type")
66  local groundType = getXMLInt(xmlFile, key.."#groundType")
67
68  self:addSprayType(name, litersPerSecond, typeName, groundType,
    isBaseType)
69
70  i = i + 1
71  end
72
73  return true
74  end
```

addSprayType

Description

Adds a new sprayType

Definition

addSprayType(string name, float litersPerSecond)

Arguments

string	name	sprayType index name
float	litersPerSecond	liter per second

Return Values

boolean	changed	visibility has changed
table	sprayType	sprayType object

getSprayTypeByIndex

Description

Gets a sprayType by index

Definition

getSprayTypeByIndex(integer index)

Arguments

integer	index	the sprayType index

Return Values

boolean	changed	visibility has changed
table	sprayType	the sprayType object

Code

```
140  function SprayTypeManager:getSprayTypeByIndex(index)
```

```
141  if index ~= nil then
142  return self.sprayTypes[index]
143  end
144  return nil
145  end
```

getSprayTypeByName

Description

Gets a sprayType by name

Definition

getSprayTypeByName(string name)

Arguments

string name the sprayType name

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

table sprayType the sprayType object

Code

```
151  function SprayTypeManager:getSprayTypeByName(name)
152  if name ~= nil then
153  name = name:upper()
154  return self.nameToSprayType[name]
155  end
156  return nil
157  end
```

getFillTypeNameByIndex

Description

Gets a fillTypeName by index

Definition

getFillTypeNameByIndex(integer index)

Arguments

integer index the sprayType index

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

string fillTypeName the sprayType name

Code

```
163  function SprayTypeManager:getFillTypeNameByIndex(index)
164  if index ~= nil then
165  return self.indexToName[index]
166  end
167  return nil
168  end
```

getFillTypeIndexByName

Description

Gets a sprayType index by name

Definition

getFillTypeIndexByName(string name)

Arguments

string name the sprayType index name

Return Values

boolean isStarted motor is started

integer fillTypeIndex the sprayType index

Code

```
174 function SprayTypeManager:getFillTypeIndexByName(name)
175 if name ~= nil then
176 name = name:upper()
177 return self.nameToIndex[name]
178 end
179 return nil
180 end
```

getFillTypeByName

Description

Gets a sprayType by index name

Definition

getFillTypeByName(string name)

Arguments

string name the sprayType index name

Return Values

boolean success success

table sprayType the sprayType object

Code

```
186 function SprayTypeManager:getFillTypeByName(name)
187 if name ~= nil then
188 name = name:upper()
189 return self.nameToSprayType[name]
190 end
191 return nil
192 end
```

getSprayTypeIndexByFillTypeIndex

Description

Gets a sprayTypeIndex by fillType index

Definition

getSprayTypeIndexByFillTypeIndex(integer index)

Arguments

integer index the fillType index

Return Values

boolean isOperating is operating
integer sprayTypeIndex the sprayType index

Code

```
205 function SprayTypeManager:getSprayTypeIndexByFillTypeIndex(index)
206 if index ~= nil then
207 local sprayType = self.fillTypeIndexToSprayType[index]
208 if sprayType ~= nil then
209 return sprayType.index
210 end
211 end
212 return nil
213 end
```

getSprayTypes

Description

Gets a list of sprayTypes

Definition

getSprayTypes()

Return Values

boolean deactivate vehicle deactivates on leave
table sprayTypes list of sprayTypes

Code

```
218 function SprayTypeManager:getSprayTypes()
219 return self.sprayTypes
220 end
```

TensionBeltManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded
table instance instance of object

Code

```
18 function TensionBeltManager:new(customMt)
19 if customMt == nil then
20 customMt = TensionBeltManager_mt
21 end
22
23 local self = {}
24 setmetatable(self, customMt)
```

```
25
26  self:initDataStructures()
27
28  return self
29  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

boolean success success

Code

```
33  function TensionBeltManager:initDataStructures()
34  self.belts = {}
35  self.defaultBeltData = nil
36  end
```

ToolTypeManager

Description

new

Description

Creating manager

Definition

new()

Return Values

boolean checkSpeedlimit check speed limit

table instance instance of object

Code

```
19  function ToolTypeManager:new(customMt)
20  local self = AbstractManager:new(customMt or ToolTypeManager_mt)
21
22  return self
23  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Return Values

float speedLimit speed limit

Code

```
27  function ToolTypeManager:initDataStructures()
```

```
28  self.indexToName = {}
29  self.nameToInt = {}
30
31  ToolType = self.nameToInt
32  end
```

loadMapData

Description

Loads initial manager

Definition

loadMapData()

Return Values

float dirtMultiplier current wear multiplier
boolean true if loading was successful else false

Code

```
37  function ToolTypeManager:loadMapData()
38  ToolTypeManager:superClass().loadMapData(self)
39
40  self:addToolType("undefined")
41  self:addToolType("dischargeable")
42  self:addToolType("trigger")
43  self:addToolType("bale")
44
45  return true
46  end
```

addToolType

Description

Adds a new baleType

Definition

addToolType(string name, float litersPerSecond)

Arguments

string name baleType index name
float litersPerSecond liter per second

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded
table baleType baleType object

getToolTypeNameByIndex

Description

Returns tool type name by given index

Definition

getToolTypeNameByIndex(integer toolTypeIndex)

Arguments

integer toolTypeIndex tool type index

Return Values

float rpm rpm of pto

string toolTypeName tool type name

Code

```
70  function ToolTypeManager:getToolTypeNameByIndex(index)
71  if self.indexToName[index] ~= nil then
72  return self.indexToName[index]
73  end;
74
75  return "UNDEFINED"
76  end;
```

getToolTypeIndexByName

Description

Returns tool type index by given name

Definition

getToolTypeIndexByName(string toolTypeName)

Arguments

string toolTypeName tool type name

Return Values

boolean consume consumePtoPower

integer toolTypeIndex tool type index

Code

```
82  function ToolTypeManager:getToolTypeIndexByName(name)
83  name = name:upper()
84  if self.nameToInt[name] ~= nil then
85  return self.nameToInt[name]
86  end;
87
88  return ToolType.UNDEFINED
89  end;
```

getNumberOfToolTypes

Description

Returns number of tool types

Definition

getNumberOfToolTypes()

Return Values

float powerMultiplier current power multiplier

integer numToolTypes number of tool types

Code

```
94  function ToolTypeManager:getNumberOfToolTypes()
95  return table.getn(self.indexToName)
96  end;
```

TreePlantManager
Description
loadMapData
Description

Load data on map load

Definition

loadMapData()

Return Values

float	torque consumed pto torque in kNm
boolean true	if loading was successful else false

Code

```
60  function TreePlantManager:loadMapData(xmlFile, missionInfo, baseDirectory)
61      TreePlantManager:superClass().loadMapData(self)
62
63      self:loadDefaultTypes(missionInfo, baseDirectory)
64      return XMLUtil.loadDataFromMapXML(xmlFile, "treeTypes", baseDirectory, self, self.loadTreeTypes, missionInfo)
65  end
```

mission00
Description
Mission00:doPauseGame
Description

Called on GamePauseEvent.

Definition

Mission00:doPauseGame()

Return Values

boolean allow	allow turn on

Mission00:doUnpauseGame
Description

Called on GamePauseEvent.

Definition

Mission00:doUnpauseGame()

Return Values

string warningText	turn on not allowed warning text

ModCategoryInfo
Description
new
Description

Create a new ModCategoryInfo instance.

Definition

new()

Return Values

float dirtMultiplier	current wear multiplier

ModInfo
Description
new
Description

Create a new ModInfo instance.

Definition

new()

Return Values

boolean hasPrerequisite true if all prerequisite specializations are loaded

Player
Description

Player class.
--

new
Description

Creating player and initializing member variables

Definition

new(boolean isServer, boolean isClient)

Arguments

boolean isServer is server

boolean isClient is client

Return Values

table instance Instance of object

Code

```
70  function Player:new(isServer, isClient)
71  local self = Object:new(isServer, isClient, Player_mt)
72
73  self.isControlled = false
74  self.isOwner = false
75  self.isEntered = false
76  self.debugFlightModeWalkingSpeed = 0.016
77  self.debugFlightModeRunningFactor = 1
78
79  self.networkInformation = {}
80  self.networkInformation.creatorConnection = nil
81  self.networkInformation.history = {}
82  self.networkInformation.index = 0
83  if self.isServer then
84  self.networkInformation.sendIndex = 0
85  end
86  self.networkInformation.interpolationTime = InterpolationTime:new(1.0)
87  self.networkInformation.interpolatorPosition = InterpolatorPosition:new(
    0.0, 0.0)
```

```
88  self.networkInformation.interpolatorQuaternion =
    InterpolatorQuaternion:new(0.0, 0.0, 0.0, 1.0) -- only used on server si
    rotation of camera
89  self.networkInformation.interpolatorOnGround = InterpolatorValue:new(0.0
90  self.networkInformation.tickTranslation = {0.0, 0.0, 0.0}
91  self.networkInformation.dirtyFlag = self:getNextDirtyFlag()
92  self.networkInformation.updateTargetTranslationPhysicsIndex = -1
93  self.networkInformation.rotateObject = false
94  self.networkInformation.rotateObjectInputV = 0.0
95  self.networkInformation.rotateObjectInputH = 0.0
96
97  self.motionInformation = {}
98  self.motionInformation.translation = {0.0, 0.0, 0.0} -- in m
99  self.motionInformation.velocity = {0.0, 0.0, 0.0} -- in m/s
100 self.motionInformation.acceleration = {0.0, 0.0, 0.0} -- in m/s^2
101 self.motionInformation.accumulatedForces = {0.0, 0.0, 0.0} -- in N
102 self.motionInformation.damping = 0.8
103 self.motionInformation.mass = 80.0 -- in kg
104 self.motionInformation.inverseMass = 1.0 / self.motionInformation.mass -
    kg ^ -1
105 self.motionInformation.gravity = {0.0, -29.0, 0.0} -- in N
106 self.motionInformation.maxIdleSpeed = 0.1 -- in m/s
107 self.motionInformation.maxWalkingSpeed = 6.0 -- in m/s
108 self.motionInformation.maxRunningSpeed = 9.0 -- in m/s
109 self.motionInformation.maxSwimmingSpeed = 3.0 -- in m/s
110 self.motionInformation.maxCrouchingSpeed = 2.0 -- in m/s
111 self.motionInformation.maxFallingSpeed = 6.0 -- in m/s
112 self.motionInformation.maxCheatRunningSpeed = 34.0 -- in m/s
113 self.motionInformation.maxPresentationRunningSpeed = 128.0 -- in m/s
114 self.motionInformation.maxSpeedDelay = 0.1 -- in s (how long before max
    is reached)
115 self.motionInformation.brakeDelay = 0.001 -- in s (how long before veloc
    null)
116 self.motionInformation.brakeForce = {0.0, 0.0, 0.0} -- in N (force to ap
    stop the player gradually)
117 self.motionInformation.currentGroundSpeed = 0.0 -- in m/s
118 self.motionInformation.minimumFallingSpeed = -0.00001 -- in m/s
119 self.motionInformation.coveredGroundDistance = 0.0 -- in m
120 self.motionInformation.currentCoveredGroundDistance = 0.0
121 self.motionInformation.justMoved = false --
122 self.motionInformation.isBraking = false
123 self.motionInformation.lastSpeed = 0.0
```

```
124 self.motionInformation.currentSpeed = 0.0
125 self.motionInformation.isReverse = false
126 self.motionInformation.desiredSpeed = 0.0
127
128 self.baseInformation = {}
129 self.baseInformation.position = {0.0, 0.0, 0.0} -- position of self.root
130 self.baseInformation.lastPosition = {0.0, 0.0, 0.0} -- position of
    self.rootNode
131 self.baseInformation.direction = {0.0, 0.0, 0.0} -- direction of the
    self.cameraNode
132 self.baseInformation.isOnGround = true
133 self.baseInformation.isOnGroundPhysics = true
134 self.baseInformation.isCloseToGround = true
135 self.baseInformation.isInWater = false
136 self.baseInformation.wasInWater = false
137 self.baseInformation.waterLevel = -0.5
138 self.baseInformation.plungedInWater = false
139 self.baseInformation.plungedYVelocityThreshold = -2.0
140 self.baseInformation.isInDebug = false
141 self.baseInformation.capsuleHeight = 0.0
142 self.baseInformation.capsuleRadius = 0.0
143 self.baseInformation.tagOffset = {0.0, 1.8, 0.0}
144 self.baseInformation.translationAlphaDifference = 0.0
145 self.baseInformation.animDt = 0.0
146 self.baseInformation.isCrouched = false
147 self.baseInformation.isUsingChainsawHorizontal = false
148 self.baseInformation.isUsingChainsawVertical = false
149 self.baseInformation.currentHandtool = nil
150 self.baseInformation.capsuleTotalHeight = 0.0
151
152 self.inputInformation = {}
153 self.inputInformation.moveForward = 0.0
154 self.inputInformation.moveRight = 0.0
155 self.inputInformation.moveUp = 0.0 -- for debug flight mode
156 self.inputInformation.pitchCamera = 0.0
157 self.inputInformation.yawCamera = 0.0
158 self.inputInformation.runAxis = 0.0
159 self.inputInformation.crouchState = Player.BUTTONSTATES.RELEASED
160 self.inputInformation.interactState = Player.BUTTONSTATES.RELEASED
161
162 -- These are the parameters for the input registration and the eventId
```

```
163 self.inputInformation.registrationList = {}
164 self.inputInformation.registrationList[InputAction.AXIS_MOVE_SIDE_PLAYER
eventId="", callback=self.onInputMoveSide, triggerUp=false, triggerDown=
triggerAlways=true, activeType=Player.INPUT_ACTIVE_TYPE.IS_MOVEMENT,
callbackState=nil, text="", textVisibility=false }
165 self.inputInformation.registrationList[InputAction.AXIS_MOVE_FORWARD_PLA
{ eventId="", callback=self.onInputMoveForward, triggerUp=false,
triggerDown=false, triggerAlways=true,
activeType=Player.INPUT_ACTIVE_TYPE.IS_MOVEMENT, callbackState=nil, text
textVisibility=false }
166 self.inputInformation.registrationList[InputAction.AXIS_LOOK_LEFTRIGHT_P
= { eventId="", callback=self.onInputLookLeftRight, triggerUp=false,
triggerDown=false, triggerAlways=true,
activeType=Player.INPUT_ACTIVE_TYPE.IS_MOVEMENT, callbackState=nil, text
textVisibility=false }
167 self.inputInformation.registrationList[InputAction.AXIS_LOOK_UPDOWN_PLAY
{ eventId="", callback=self.onInputLookUpDown, triggerUp=false,
triggerDown=false, triggerAlways=true,
activeType=Player.INPUT_ACTIVE_TYPE.IS_MOVEMENT, callbackState=nil, text
textVisibility=false }
168 self.inputInformation.registrationList[InputAction.AXIS_RUN] = { eventId
callback=self.onInputRun, triggerUp=false, triggerDown=false,
triggerAlways=true, activeType=Player.INPUT_ACTIVE_TYPE.IS_MOVEMENT,
callbackState=nil, text="", textVisibility=false }
169 self.inputInformation.registrationList[InputAction.JUMP] = { eventId="",
callback=self.onInputJump, triggerUp=false, triggerDown=true,
triggerAlways=false, activeType=Player.INPUT_ACTIVE_TYPE.IS_MOVEMENT,
callbackState=nil, text="", textVisibility=GS_IS_CONSOLE_VERSION }
170 -- TODO: read from game settings? also needs to be applied to
PlayerStateCrouch.toggleMode. triggerAlways = not crouchToggleMode
171 self.inputInformation.registrationList[InputAction.CROUCH] = { eventId='
callback=self.onInputCrouch, triggerUp=false, triggerDown=true,
triggerAlways=true, activeType=Player.INPUT_ACTIVE_TYPE.IS_MOVEMENT,
callbackState=nil, text="", textVisibility=GS_IS_CONSOLE_VERSION }
172 self.inputInformation.registrationList[InputAction.ACTIVATE_OBJECT] = {
eventId="", callback=self.onInputActivateObject, triggerUp=false,
triggerDown=true, triggerAlways=false,
activeType=Player.INPUT_ACTIVE_TYPE.IS_MOVEMENT, callbackState=nil, text
textVisibility=true }
173 self.inputInformation.registrationList[InputAction.ROTATE_OBJECT_LEFT_RI
{ eventId="", callback=self.onInputRotateObjectHorizontally, triggerUp=f
triggerDown=false, triggerAlways=true,
activeType=Player.INPUT_ACTIVE_TYPE.IS_CARRYING, callbackState=nil,
text=g_i18n:getText("action_rotateObjectHorizontally"), textVisibility=t
174 self.inputInformation.registrationList[InputAction.ROTATE_OBJECT_UP_DOWN
eventId="", callback=self.onInputRotateObjectVertically, triggerUp=false
triggerDown=false, triggerAlways=true,
activeType=Player.INPUT_ACTIVE_TYPE.IS_CARRYING, callbackState=nil,
text=g_i18n:getText("action_rotateObjectVertically"), textVisibility=tru
```

```
175 self.inputInformation.registrationList[InputAction.ENTER] = { eventId="",
    callback=self.onInputEnter, triggerUp=false, triggerDown=true,
    triggerAlways=false, activeType=Player.INPUT_ACTIVE_TYPE.STARTS_ENABLED,
    callbackState=nil, text="", textVisibility=false }
176 self.inputInformation.registrationList[InputAction.TOGGLE_LIGHTS_FPS] =
    eventId="", callback=self.onInputToggleLight, triggerUp=false,
    triggerDown=true, triggerAlways=false,
    activeType=Player.INPUT_ACTIVE_TYPE.STARTS_ENABLED, callbackState=nil,
    text="", textVisibility=false }
177 self.inputInformation.registrationList[InputAction.THROW_OBJECT] = {
    eventId="", callback=self.onInputThrowObject, triggerUp=false,
    triggerDown=true, triggerAlways=false,
    activeType=Player.INPUT_ACTIVE_TYPE.STARTS_DISABLED, callbackState=nil,
    text=g_i18n:getText("input_THROW_OBJECT"), textVisibility=true }
178 self.inputInformation.registrationList[InputAction.INTERACT] = { eventId
    callback=self.onInputInteract, triggerUp=true, triggerDown=true,
    triggerAlways=false, activeType=Player.INPUT_ACTIVE_TYPE.STARTS_DISABLED
    callbackState=nil, text="", textVisibility=true }
179 self.inputInformation.registrationList[InputAction.NEXT_HANDTOOL] = {
    eventId="", callback=self.onInputCycleHandTool, triggerUp=false,
    triggerDown=true, triggerAlways=false,
    activeType=Player.INPUT_ACTIVE_TYPE.STARTS_DISABLED, callbackState=1,
    text=g_i18n:getText("input_NEXT_HANDTOOL"), textVisibility=false }
180 self.inputInformation.registrationList[InputAction.PREVIOUS_HANDTOOL] =
    eventId="", callback=self.onInputCycleHandTool, triggerUp=false,
    triggerDown=true, triggerAlways=false,
    activeType=Player.INPUT_ACTIVE_TYPE.STARTS_DISABLED, callbackState=-1,
    text=g_i18n:getText("input_PREVIOUS_HANDTOOL"), textVisibility=false }
181 self.inputInformation.registrationList[InputAction.DEBUG_PLAYER_ENABLE]
    eventId="", callback=self.onInputDebugFlyToggle, triggerUp=false,
    triggerDown=true, triggerAlways=false,
    activeType=Player.INPUT_ACTIVE_TYPE.IS_DEBUG, callbackState=nil, text="",
    textVisibility=false }
182 self.inputInformation.registrationList[InputAction.DEBUG_PLAYER_UP_DOWN]
    eventId="", callback=self.onInputDebugFlyUpDown, triggerUp=false,
    triggerDown=false, triggerAlways=true,
    activeType=Player.INPUT_ACTIVE_TYPE.IS_DEBUG, callbackState=nil, text="",
    textVisibility=false }
183 self.inputInformation.registrationList[InputAction.ACTIVATE_HANDTOOL] =
    eventId="", callback=self.onInputActivateHandtool, triggerUp=false,
    triggerDown=true, triggerAlways=true,
    activeType=Player.INPUT_ACTIVE_TYPE.STARTS_DISABLED, callbackState=nil,
    text=g_i18n:getText("input_ACTIVATE_HANDTOOL"), textVisibility=false }
184
185 self.soundInformation = {}
186 self.soundInformation.samples = {}
187
188 self.soundInformation.samples.swim = {}
189 self.soundInformation.samples.plunge = {}
```

```
190 self.soundInformation.samples.horseBrush = {}
191 self.soundInformation.distancePerFootstep = {}
192 self.soundInformation.distancePerFootstep.crouch = 0.5
193 self.soundInformation.distancePerFootstep.walk = 0.75
194 self.soundInformation.distancePerFootstep.run = 1.5
195 self.soundInformation.distanceSinceLastFootstep = 0.0
196 self.soundInformation.isSampleSwinPlaying = false
197
198 self.particleSystemsInformation = {}
199 self.particleSystemsInformation.systems = {}
200 self.particleSystemsInformation.systems.swim = {}
201 self.particleSystemsInformation.systems.plunge = {}
202 self.particleSystemsInformation.swimNode = 0
203 self.particleSystemsInformation.plungeNode = 0
204
205 self.animationInformation = {}
206 self.animationInformation.player = nil
207 self.animationInformation.parameters = {}
208 self.animationInformation.parameters.forwardVelocity = {id=1, value=0.0,
    type=1}
209 self.animationInformation.parameters.verticalVelocity = {id=2, value=0.0
    type=1}
210 self.animationInformation.parameters.yawVelocity = {id=3, value=0.0, typ
211 self.animationInformation.parameters.onGround = {id=4, value=false, type
212 self.animationInformation.parameters.inWater = {id=5, value=false, type=
213 self.animationInformation.parameters.isCrouched = {id=6, value=false, ty
214 self.animationInformation.parameters.absForwardVelocity = {id=7, value=0
    type=1}
215 self.animationInformation.parameters.isCloseToGround = {id=8, value=fals
    type=0}
216 self.animationInformation.parameters.isUsingChainsawHorizontal = {id=9,
    value=false, type=0}
217 self.animationInformation.parameters.isUsingChainsawVertical = {id=10,
    value=false, type=0}
218
219 -- @see Player.loadCustomization for the content of this struct
220 self.visualInformation = nil
221
222 -- cached info
223 self.animationInformation.oldYaw = 0.0 -- in rad
224 self.animationInformation.newYaw = 0.0 -- in rad
225 self.animationInformation.estimatedYawVelocity = 0.0 -- in rad/s
```

```lua

-- Player movement lock flag
self.walkingIsLocked = false

self.canRideAnimal = false
self.canEnterVehicle = false

self.isLightActive = false

self.rotX = 0
self.rotY = 0
self.cameraRotY = 0

self.graphicsRotY = 0
self.targetGraphicsRotY = 0

self.camera = 0

self.time = 0

self.lightNode = nil

self.clipDistance = 500

self.lastAnimPosX = 0
self.lastAnimPosY = 0
self.lastAnimPosZ = 0

self.walkDistance = 0
self.animUpdateTime = 0

self.debugFlightMode = false
self.debugFlightCoolDown = 0

self.requestedFieldData = false

self.playerStateMachine = PlayerStateMachine:new(self)

self.farmId = FarmManager.SPECTATOR_FARM_ID

```

```
266 return self
267 end
```

loadVisuals

Description

Loading visuals for the player

Definition

loadVisuals(table self, string xmlFilename, table playerStyle, table linkNode, bool isRealPlayer, table ikChains, function getParentFunc, function getParentFuncTarget, table parentObj)

Arguments

table	self	player object for loading the visuals
string	xmlFilename	XML filename
table	playerStyle	
table	linkNode	node to link the third person skeleton of the player on
bool	isRealPlayer	false if player is in a vehicle
table	ikChains	list of IK chains
function	getParentFunc	
function	getParentFuncTarget	
table	parentObj	parent object

Return Values

bool true if ok

Code

```
281 function Player.loadVisuals(self, xmlFilename, playerStyle, linkNode, isRealPlayer, ikChains, getParentFunc, getParentFuncTarget, parentObj)
282 self.xmlFilename = xmlFilename
283 self.customEnvironment, self.baseDirectory = Utils.getModNameAndBaseDirectory(xmlFilename)
284
285 local xmlFile = loadXMLFile("TempXML", xmlFilename)
286 if xmlFile == 0 then
287 return false
288 end
289
290 local filename = getXMLString(xmlFile, "player.filename")
291 self.filename = Utils.getFilename(filename, self.baseDirectory)
292 local rootNode = g_i3DManager:loadSharedI3DFile(self.filename, nil, nil, true)
293
294 self.graphicsRootNode = createTransformGroup("player_graphicsRootNode")
295
296 if isRealPlayer then
297 self.cameraNode = I3DUtil.indexToObject(rootNode, getXMLString(xmlFile, "player.camera#index"))
```

```
298 if self.cameraNode == nil then
299 g_logManager:devError("Error: Failed to find player camera in
'"..filename.."'")
300 end
301 self.camX, self.camY, self.camZ = getTranslation(self.cameraNode)
302 setNearClip(self.cameraNode, 0.15) -- lessens night light windows flicke
(standard value: 0.1)
303 setFarClip(self.cameraNode, 6000) -- prevents distance mountains from
clipping (standard value: 5000)
304 self.fovY = calculateFovY(self.cameraNode)
305 setFovY(self.cameraNode, self.fovY)
306
307 self.animRootThirdPerson = I3DUtil.indexToObject(rootNode,
getXMLString(xmlFile, "player.character.thirdPerson#animRootNode"))
308 if self.animRootThirdPerson == nil then
309 g_logManager:devError("Error: Failed to find animation root node in
'"..self.filename.."'")
310 end
311
312 -- Capsule information
313 self.baseInformation.capsuleHeight = getXMLFloat(xmlFile,
"player.character#physicsCapsuleHeight")
314 self.baseInformation.capsuleRadius = getXMLFloat(xmlFile,
"player.character#physicsCapsuleRadius")
315 self.baseInformation.capsuleTotalHeight = self.baseInformation.capsuleHe
+ 2.0 * self.baseInformation.capsuleRadius
316
317 -- first person view
318 self.cuttingCameraNode = I3DUtil.indexToObject(rootNode,
getXMLString(xmlFile, "player.character.firstPerson#cuttingCameraNode"))
319 setNearClip(self.cuttingCameraNode, 0.15)
320 setFarClip(self.cuttingCameraNode, 6000)
321 self.fovY = calculateFovY(self.cuttingCameraNode)
322 setFovY(self.cuttingCameraNode, self.fovY)
323 end
324
325 -- Avator customization
326 self.visualInformation = PlayerStyle:new()
327 self.visualInformation:copySelection(playerStyle)
328 self.visualInformation:loadXML(self, rootNode, xmlFile, "player.characte
329
330 self.skeletonThirdPerson = I3DUtil.indexToObject(rootNode,
getXMLString(xmlFile, "player.character.thirdPerson#skeleton"))
```

```
331
332 if self.skeletonThirdPerson == nil then
333 g_logManager:devError("Error: Failed to find skeleton root node in
    '"..self.filename.."'")
334 end
335
336 self.meshThirdPerson = I3DUtil.indexToObject(rootNode, getXMLString(xmlF
    "player.character.thirdPerson#mesh"))
337
338 if self.meshThirdPerson == nil then
339 g_logManager:devError("Error: Failed to find player mesh in
    '"..self.filename.."'")
340 end
341
342 if self.meshThirdPerson ~= nil then
343 setVisibility(self.meshThirdPerson, false)
344 setClipDistance(self.meshThirdPerson, 200)
345 end
346
347 self.thirdPersonSpineNode = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, "player.character.thirdPerson#spine"))
348 self.thirdPersonSuspensionNode =
    Utils.getNoNil(I3DUtil.indexToObject(rootNode, getXMLString(xmlFile,
    "player.character.thirdPerson#suspension")), self.thirdPersonSpineNode)
349 self.thirdPersonRightHandNode = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, "player.character.thirdPerson#rightHandNode"))
350 self.thirdPersonLeftHandNode = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, "player.character.thirdPerson#leftHandNode"))
351
352 self.lightNode = I3DUtil.indexToObject(rootNode, getXMLString(xmlFile,
    "player.light#index"))
353 if self.lightNode ~= nil then
354 setVisibility(self.lightNode, false)
355 end
356
357 self.pickUpKinematicHelperNode = I3DUtil.indexToObject(rootNode,
    getXMLString(xmlFile, "player.pickUpKinematicHelper#index"))
358 if self.pickUpKinematicHelperNode ~= nil then
359 self.pickUpKinematicHelperNodeChild =
    createTransformGroup("pickUpKinematicHelperNodeChild")
360 link(self.pickUpKinematicHelperNode, self.pickUpKinematicHelperNodeChild
361 end
362
```

```lua
-- IK Chains
local i = 0
while true do
local key = string.format("player.ikChains.ikChain(%d)", i)
if not hasXMLProperty(xmlFile, key) then
break
end
IKUtil.loadIKChain(xmlFile, key, rootNode, rootNode, ikChains)
i = i + 1
end
IKUtil.setIKChainInactive(ikChains, "spine")
if isRealPlayer then
IKUtil.deleteIKChain(ikChains, "rightFoot")
IKUtil.deleteIKChain(ikChains, "leftFoot")
IKUtil.deleteIKChain(ikChains, "rightArm")
IKUtil.deleteIKChain(ikChains, "leftArm")
IKUtil.deleteIKChain(ikChains, "spine")
end

-- Linking: Do not use indexToObject below
if self.meshThirdPerson ~= nil then
link(getRootNode(), self.meshThirdPerson)
end
if isRealPlayer then
self.chainsawCameraFocus = createTransformGroup("player_chainsawCameraFo
self.chainsawSplitShapeFocus =
createTransformGroup("player_chainsawSplitShapeFocus")
link(self.cameraNode, self.chainsawCameraFocus)
link(self.chainsawCameraFocus, self.chainsawSplitShapeFocus)

local cutFocusOffset =
StringUtil.getVectorNFromString(Utils.getNoNil(getXMLString(xmlFile,
"player.character.firstPerson#cutFocusOffset"), "0 0 -1"), 3)
setTranslation(self.chainsawSplitShapeFocus, unpack(cutFocusOffset))
local cutFocusRotation =
StringUtil.getVectorNFromString(Utils.getNoNil(getXMLString(xmlFile,
"player.character.firstPerson#cutFocusRotation"), "0 0 0"), 3)
local rotX2, rotY2, rotZ2 = unpack(cutFocusRotation)
setRotation(self.chainsawSplitShapeFocus, math.rad(rotX2), math.rad(rotY
math.rad(rotZ2))
self.minCutDistance = Utils.getNoNil(getXMLFloat(xmlFile,
"player.character.firstPerson#minCutDistance"), 0.5)
```

```
398 self.maxCutDistance = Utils.getNoNil(getXMLFloat(xmlFile,
    "player.character.firstPerson#maxCutDistance"), 1.0)
399 self.cutDetectionDistance = Utils.getNoNil(getXMLFloat(xmlFile,
    "player.character.firstPerson#cutDetectionDistance"), 10.0)
400 end
401
402 if isRealPlayer then
403 self.skeletonRootNode = createTransformGroup("player_skeletonRootNode")
404 self.foliageBendingNode = createTransformGroup("player_foliageBendingNod
405
406 link(getRootNode(), self.graphicsRootNode)
407 link(self.graphicsRootNode, self.cameraNode)
408 link(self.graphicsRootNode, self.skeletonRootNode)
409 link(self.graphicsRootNode, self.foliageBendingNode)
410
411 if self.animRootThirdPerson ~= nil then
412 link(self.skeletonRootNode, self.animRootThirdPerson)
413 if self.skeletonThirdPerson ~= nil then
414 link(self.animRootThirdPerson, self.skeletonThirdPerson)
415 end
416 end
417
418 self.visualInformation:linkProtectiveWear(self.skeletonRootNode)
419
420 if self.skeletonThirdPerson ~= nil and
    getNumOfChildren(self.skeletonThirdPerson) > 0 then
421 local animNode = g_animCache:getNode(AnimationCache.CHARACTER)
422 cloneAnimCharacterSet(animNode, getParent(self.skeletonThirdPerson))
423 local animCharsetId =
    getAnimCharacterSet(getChildAt(self.skeletonThirdPerson, 0))
424 self.animationInformation.player = createConditionalAnimation()
425
426 for key, parameter in pairs(self.animationInformation.parameters) do
427 conditionalAnimationRegisterParameter(self.animationInformation.player,
    parameter.id, parameter.type, key)
428 end
429 initConditionalAnimation(self.animationInformation.player, animCharsetId
    self.xmlFilename, "player.conditionalAnimation")
430 setConditionalAnimationSpecificParameterIds(self.animationInformation.pl
    self.animationInformation.parameters.absForwardVelocity.id,
    self.animationInformation.parameters.yawVelocity.id)
431 end
```

```
432
433 self.toolsRootNode = createTransformGroup("toolsRoot")
434
435 if self.isOwner then
436 local toolRotation =
    StringUtil.getVectorNFromString(Utils.getNoNil(getXMLString(xmlFile,
    "player.character.toolNode#firstPersonRotation"), "0 0 0"), 3)
437 local rotX, rotY, rotZ = unpack(toolRotation)
438 setRotation(self.toolsRootNode, math.rad(rotX), math.rad(rotY),
    math.rad(rotZ))
439 local toolTranslate =
    StringUtil.getVectorNFromString(Utils.getNoNil(getXMLString(xmlFile,
    "player.character.toolNode#firstPersonTranslation"), "0 0 0"), 3)
440 local transX, transY, transZ = unpack(toolTranslate)
441 setTranslation(self.toolsRootNode, transX, transY, transZ)
442 link(self.cuttingCameraNode, self.toolsRootNode)
443 else
444 local toolRotation =
    StringUtil.getVectorNFromString(Utils.getNoNil(getXMLString(xmlFile,
    "player.character.toolNode#thirdPersonRotation"), "0 0 0"), 3)
445 local rotX, rotY, rotZ = unpack(toolRotation)
446 setRotation(self.toolsRootNode, math.rad(rotX), math.rad(rotY),
    math.rad(rotZ))
447 local toolTranslate =
    StringUtil.getVectorNFromString(Utils.getNoNil(getXMLString(xmlFile,
    "player.character.toolNode#thirdPersonTranslation"), "0 0 0"), 3)
448 local transX, transY, transZ = unpack(toolTranslate)
449 setTranslation(self.toolsRootNode, transX, transY, transZ)
450 link(self.thirdPersonRightHandNode, self.toolsRootNode)
451 end
452
453 -- Fx
454 self.particleSystemsInformation.swimNode = createTransformGroup("swimFXN
455 link(getRootNode(), self.particleSystemsInformation.swimNode)
456 self.particleSystemsInformation.plungeNode =
    createTransformGroup("plungeFXNode")
457 link(getRootNode(), self.particleSystemsInformation.plungeNode)
458
459 ParticleUtil.loadParticleSystem(xmlFile,
    self.particleSystemsInformation.systems.swim, "player.particleSystems.sw
    self.particleSystemsInformation.swimNode, false, nil, self.baseDirectory
460 ParticleUtil.loadParticleSystem(xmlFile,
    self.particleSystemsInformation.systems.plunge,
```

```
    "player.particleSystems.plunge", self.particleSystemsInformation.plungeN
    false, nil, self.baseDirectory)
461 --
462 self.visualInformation:applySelection()
463 if self.isOwner then
464 self.visualInformation:setVisibility(false)
465 else
466 self.visualInformation:setVisibility(true)
467 end
468 else
469 link(linkNode, self.skeletonThirdPerson)
470 self.visualInformation:linkProtectiveWear(linkNode)
471
472 if self.pickUpKinematicHelperNode ~= nil then
473 delete(self.pickUpKinematicHelperNode)
474 self.pickUpKinematicHelperNode = nil
475 end
476 if self.lightNode ~= nil then
477 delete(self.lightNode)
478 self.lightNode = nil
479 end
480 if self.cameraNode ~= nil then
481 delete(self.cameraNode)
482 self.cameraNode = nil
483 end
484 local offset = {localToLocal(self.thirdPersonSpineNode,
    self.skeletonThirdPerson, 0, 0, 0)}
485
486 setTranslation(self.skeletonThirdPerson, -offset[1], -offset[2], -offset
487 self.visualInformation:applySelection()
488 self.visualInformation:setVisibility(true)
489 end
490
491 -- Sound
492 if isRealPlayer then
493 self.soundInformation.surfaceSounds = {}
494 self.soundInformation.surfaceIdToSound = {}
495 self.soundInformation.surfaceNameToSound = {}
496 self.soundInformation.currentSurfaceSound = nil
497
498 if not self.isOwner then
```

```
499 for _, surfaceSound in pairs(g_currentMission.surfaceSounds) do
500 if surfaceSound.type == "footstep" and surfaceSound.sample ~= nil then
501 local sample = g_soundManager:cloneSample(surfaceSound.sample, self.root
    self)
502 sample.sampleName = surfaceSound.name
503
504 table.insert(self.soundInformation.surfaceSounds, sample)
505 self.soundInformation.surfaceIdToSound[surfaceSound.materialId] = sample
506 self.soundInformation.surfaceNameToSound[surfaceSound.name] = sample
507 end
508 end
509
510 self.soundInformation.samples.swim =
    g_soundManager:loadSampleFromXML(xmlFile, "player.sounds.water", "swim",
    self.baseDirectory, self.rootNode, 0, AudioGroup.ENVIRONMENT, nil, nil)
511 self.soundInformation.samples.plunge =
    g_soundManager:loadSampleFromXML(xmlFile, "player.sounds.water", "plunge
    self.baseDirectory, self.rootNode, 1, AudioGroup.ENVIRONMENT, nil, nil)
512 self.soundInformation.samples.flashlight =
    g_soundManager:loadSampleFromXML(xmlFile, "player.sounds.tools",
    "flashlight", self.baseDirectory, self.rootNode, 1, AudioGroup.ENVIRONME
    nil, nil)
513 self.soundInformation.samples.horseBrush =
    g_soundManager:loadSampleFromXML(xmlFile, "player.sounds.tools",
    "horseBrush", self.baseDirectory, self.rootNode, 0, AudioGroup.ENVIRONME
    nil, nil)
514 else
515 for _, surfaceSound in pairs(g_currentMission.surfaceSounds) do
516 if surfaceSound.type == "footstep" and surfaceSound.sample ~= nil then
517 local sample = g_soundManager:cloneSample2D(surfaceSound.sample, self)
518 sample.sampleName = surfaceSound.name
519
520 table.insert(self.soundInformation.surfaceSounds, sample)
521 self.soundInformation.surfaceIdToSound[surfaceSound.materialId] = sample
522 self.soundInformation.surfaceNameToSound[surfaceSound.name] = sample
523 end
524 end
525
526 self.soundInformation.samples.swim =
    g_soundManager:loadSample2DFromXML(xmlFile, "player.sounds.water", "swim
    self.baseDirectory, 0, AudioGroup.ENVIRONMENT)
527 self.soundInformation.samples.plunge =
    g_soundManager:loadSample2DFromXML(xmlFile, "player.sounds.water", "plun
    self.baseDirectory, 1, AudioGroup.ENVIRONMENT)
```

```
528 	self.soundInformation.samples.flashlight = g_soundManager:loadSample2DFromXML(xmlFile, "player.sounds.tools", "flashlight", self.baseDirectory, 1, AudioGroup.ENVIRONMENT)
529 	self.soundInformation.samples.horseBrush = g_soundManager:loadSample2DFromXML(xmlFile, "player.sounds.tools", "horseBrush", self.baseDirectory, 0, AudioGroup.ENVIRONMENT)
530 	end
531 
532 	self.soundInformation.distancePerFootstep.crouch = Utils.getNoNil(getXMLFloat(xmlFile, "player.sounds.footsteps#distancePerFootstepCrouch"), 0.5)
533 	self.soundInformation.distancePerFootstep.walk = Utils.getNoNil(getXMLFloat(xmlFile, "player.sounds.footsteps#distancePerFootstepWalk"), 0.75)
534 	self.soundInformation.distancePerFootstep.run = Utils.getNoNil(getXMLFloat(xmlFile, "player.sounds.footsteps#distancePerFootstepRun"), 1.5)
535 	self.soundInformation.isSampleSwinPlaying = false
536 	end
537 	IKUtil.updateAlignNodes(ikChains, getParentFunc, getParentFuncTarget, parentObj)
538 
539 	delete(xmlFile)
540 	delete(rootNode)
541 	return true
542 	end
```

load

Description

Loading player information

Definition

load(string xmlFilename, string controllerName, integer playerColorIndex, table creatorConnection, bool isOwner)

Arguments

string	xmlFilename	XML filename containing player information
string	controllerName	name of the player controlling
integer	playerColorIndex	color index of the table g_playerColors[]
table	creatorConnection	
bool	isOwner	true is current player is owner

Code

```
551 	function Player:load(xmlFilename, playerStyle, creatorConnection, isOwner)
552 	self.networkInformation.creatorConnection = creatorConnection
553 	self.isOwner = isOwner
554 	self.rootNode = createTransformGroup("PlayerCCT")
```

```
555 link(getRootNode(), self.rootNode)
556
557 self.ikChains = {}
558 Player.loadVisuals(self, xmlFilename, playerStyle, nil, true,
    self.ikChains, self.getParentComponent, self, nil)
559
560 self.playerStateMachine:load()
561
562 self.isObjectInRange = false
563 self.isCarryingObject = false
564 self.pickedUpObject = nil
565 self.showTooHeavyWarning = false
566
567 local uiScale = g_gameSettings:getValue("uiScale")
568
569 -- TODO: move to HUD
570 self.pickedUpObjectWidth, self.pickedUpObjectHeight =
    getNormalizedScreenValues(80 * uiScale, 80 * uiScale)
571 self.pickedUpObjectOverlay = Overlay:new(g_baseHUDFilename, 0.5, 0.5,
    self.pickedUpObjectWidth, self.pickedUpObjectHeight)
572 self.pickedUpObjectOverlay:setAlignment(Overlay.ALIGN_VERTICAL_MIDDLE,
    Overlay.ALIGN_HORIZONTAL_CENTER)
573 self.pickedUpObjectHandUVs = getNormalizedUVs{0, 138, 80, 80}
574 self.pickedUpObjectAimingUVs = getNormalizedUVs{0, 48, 48, 48}
575 self.pickedUpObjectAimingWidth, self.pickedUpObjectAimingHeight =
    getNormalizedScreenValues(20 * uiScale, 20 * uiScale)
576 self.pickedUpObjectOverlay:setUVs(self.pickedUpObjectAimingUVs)
577 self.pickedUpObjectOverlay:setColor(1, 1, 1, 0.3)
578
579 self:moveToAbsoluteInternal(0, -200, 0)
580
581 self.controllerIndex = createCCT(self.rootNode,
    self.baseInformation.capsuleRadius,
    self.baseInformation.capsuleHeight, 0.6, 45.0, 0.1,
    Player.kinematicCollisionMask, self.motionInformation.mass)
582
583 self.lockedInput = false
584
585 if self.isOwner then
586 addConsoleCommand("gsToggleFlightAndNoHUDMode", "Enables/disables the
    flight (J) and no HUD (O) toggle keys",
    "consoleCommandToggleFlightAndNoHUDMode", self)
```

```
587  addConsoleCommand("gsToggleWoodCuttingMaker", "Enables/disables
     chainsaw woodcutting marker",
     "Player.consoleCommandToggleWoodCuttingMaker", nil)
588  addConsoleCommand("gsTogglePlayerDebug", "Enables/disables player
     debug information", "consoleCommandTogglePlayerDebug", self)
589  if g_addTestCommands then
590  addConsoleCommand("gsReloadIKChains", "Reloads IKChains",
     "Player.consoleCommandReloadIKChains", nil)
591  addConsoleCommand("gsToggleSuperStrength", "Enables/disables player
     super strength", "consoleCommandToggleSuperStrongMode", self)
592  end
593  end
594  end
```

getParentComponent

Description

Gets the parent node

Definition

getParentComponent(table node)

Arguments

table node this parameter is unused in this function

Return Values

table returns the graphics root node

Code

```
600  function Player:getParentComponent(node)
601  return self.graphicsRootNode
602  end
```

deleteVisuals

Description

Delete visual information for the player

Definition

deleteVisuals(table self, table ikChains)

Arguments

table self player object for unloading the visuals

table ikChains list of IK chains

Code

```
608  function Player.deleteVisuals(self, ikChains)
609  if self.filename ~= nil then
610  g_i3DManager:releaseSharedI3DFile(self.filename, nil, false)
611  self.filename = nil
612  end
613  for chainId, _ in pairs(ikChains) do
614  IKUtil.deleteIKChain(ikChains, chainId)
615  end
```

```
616
617 if self.particleSystemsInformation ~= nil then
618 if self.particleSystemsInformation.swimNode ~= nil then
619 delete(self.particleSystemsInformation.swimNode)
620 end
621 if self.particleSystemsInformation.plungeNode ~= nil then
622 delete(self.particleSystemsInformation.plungeNode)
623 end
624
625 ParticleUtil.deleteParticleSystem(self.particleSystemsInformation.system
626 ParticleUtil.deleteParticleSystem(self.particleSystemsInformation.system
627 end
628
629 self.visualInformation:unlinkProtectiveWear()
630
631 if self.thirdPersonSpineNode ~= nil then
632 delete(self.thirdPersonSpineNode)
633 self.thirdPersonSpineNode = nil
634 end
635 if self.skeletonThirdPerson ~= nil then
636 delete(self.skeletonThirdPerson)
637 self.skeletonThirdPerson = nil
638 end
639 if self.meshThirdPerson ~= nil then
640 delete(self.meshThirdPerson)
641 self.meshThirdPerson = nil
642 end
643 if self.graphicsRootNode ~= nil then
644 delete(self.graphicsRootNode)
645 self.graphicsRootNode = nil
646 end
647 end
```

delete

Description

Delete

Definition

delete()

Code

```
651 function Player:delete()
652 if self.isOwner then -- only remove action events if this Player instanc
    was controller by the current user
```

```
653 self:removeActionEvents()
654 end
655
656 for _, sample in pairs(self.soundInformation.samples) do
657 g_soundManager:deleteSample(sample)
658 end
659
660 g_soundManager:deleteSamples(self.soundInformation.surfaceSounds)
661
662 if self.isCarryingObject then
663 if g_server ~= nil then
664 self:pickUpObject(false)
665 end
666 end
667
668 if self.pickedUpObjectOverlay ~= nil then
669 self.pickedUpObjectOverlay:delete()
670 end
671
672 if self:hasHandtoolEquipped() then
673 self.baseInformation.currentHandtool:onDeactivate()
674 self.baseInformation.currentHandtool:delete()
675 self.baseInformation.currentHandtool = nil
676 end
677
678 delete(self.animationInformation.player)
679 self.animationInformation.player = nil
680 Player.deleteVisuals(self, self.ikChains)
681 removeCCT(self.controllerIndex)
682 delete(self.rootNode)
683 self.playerStateMachine:delete()
684 self:deleteStartleAnimalData()
685 self.lightNode = nil
686 if self.foliageBendingId ~= nil then
687 g_currentMission.foliageBendingSystem:destroyObject(self.foliageBendingI
688 self.foliageBendingId = nil
689 end
690
691 if self.isOwner then
692 removeConsoleCommand("gsToggleFlightAndNoHUDMode")
```

```
693  removeConsoleCommand("gsToggleWoodCuttingMaker")
694  removeConsoleCommand("gsTogglePlayerDebug")
695  removeConsoleCommand("gsReloadIKChains")
696  removeConsoleCommand("gsToggleSuperStrength")
697  end
698
699  Player:superClass().delete(self)
700  end
```

setCuttingAnim

Description

Set cutting animation

Definition

setCuttingAnim(bool isCutting, bool isHorizontalCut)

Arguments

bool isCutting	true if player is cutting
bool isHorizontalCut	true if player is cutting horizontaly

Code

```
706  function Player:setCuttingAnim(isCutting, isHorizontalCut)
707  if not isCutting and
     (self.baseInformation.isUsingChainsawHorizontal or
     self.baseInformation.isUsingChainsawVertical) then
708  self.baseInformation.isUsingChainsawHorizontal = false
709  self.baseInformation.isUsingChainsawVertical = false
710  elseif isCutting then
711  if isHorizontalCut then
712  self.baseInformation.isUsingChainsawHorizontal = true
713  self.baseInformation.isUsingChainsawVertical = false
714  else
715  self.baseInformation.isUsingChainsawHorizontal = false
716  self.baseInformation.isUsingChainsawVertical = true
717  end
718  end
719  end
```

readStream

Description

Reads from network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId	id of the stream to read
table	connection connection information

Code

```lua
function Player:readStream(streamId, connection)
Player:superClass().readStream(self, streamId)
local isOwner = streamReadBool(streamId)
local filename = NetworkUtil.convertFromNetworkFilename(streamReadString(streamId))
local x = streamReadFloat32(streamId)
local y = streamReadFloat32(streamId)
local z = streamReadFloat32(streamId)
local isControlled = streamReadBool(streamId)
if self.visualInformation == nil then
self.visualInformation = PlayerStyle:new()
end
self.visualInformation:readStream(streamId, connection)
self.farmId = streamReadUIntN(streamId, FarmManager.FARM_ID_SEND_NUM_BITS)
if self.filename == nil then
self:load(filename, self.visualInformation, connection, isOwner)
end
self:moveToAbsoluteInternal(x, y, z)

self:setLightIsActive(streamReadBool(streamId), true)

if isControlled ~= self.isControlled then
if isControlled then
self:onEnter(false)
else
self:onLeave()
end
end

local hasHandtool = streamReadBool(streamId)
if hasHandtool then
local handtoolFilename = NetworkUtil.convertFromNetworkFilename(streamReadString(streamId))
self:equipHandtool(handtoolFilename, true, true)
end
end
```

writeStream

Description

Writes in network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId id of the stream to read
table connection connection information

Code

```
765 function Player:writeStream(streamId, connection)
766 Player:superClass().writeStream(self, streamId)
767 streamWriteBool(streamId, connection == self.networkInformation.creator
768 streamWriteString(streamId, NetworkUtil.convertToNetworkFilename(self.xm
769 local x, y, z=getTranslation(self.rootNode)
770 streamWriteFloat32(streamId, x)
771 streamWriteFloat32(streamId, y)
772 streamWriteFloat32(streamId, z)
773 streamWriteBool(streamId, self.isControlled)
774 self.visualInformation:writeStream(streamId, connection)
775 streamWriteUIntN(streamId, self.farmId, FarmManager.FARM_ID_SEND_NUM_BIT
776
777 streamWriteBool(streamId, self.isLightActive)
778
779 local hasHandtool = self:hasHandtoolEquipped()
780 streamWriteBool(streamId, hasHandtool)
781 if hasHandtool then
782 streamWriteString(streamId,
    NetworkUtil.convertToNetworkFilename(self.baseInformation.currentHandtoo
783 end
784 end
```

readUpdateStream

Description

Reads from network stream via update

Definition

readUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId id of the stream to read
integer timestamp timestamp of the packet
table connection connection information

Code

```
791 function Player:readUpdateStream(streamId, timestamp, connection)
792 if connection:getIsServer() then
793 -- client code (read data from server)
794 local x = streamReadFloat32(streamId)
795 local y = streamReadFloat32(streamId)
796 local z = streamReadFloat32(streamId)
```

```
local alpha = streamReadFloat32(streamId)
self.cameraRotY = alpha

self.isObjectInRange = streamReadBool(streamId)
if self.isObjectInRange then
self.lastFoundObjectMass = streamReadFloat32(streamId)
else
self.lastFoundObjectMass = nil
end
self.isCarryingObject = streamReadBool(streamId)
local isOnGround = streamReadBool(streamId)

if self.isOwner then
local index = streamReadInt32(streamId)
--print("CLIENT ( "..tostring(self).." ): x/y/z = "..tostring(x).." / "..tostring(y).." / "..tostring(z))
-- remove history entries before the one sent by the server
while self.networkInformation.history[1] ~= nil and self.networkInformation.history[1].index <= index do
table.remove(self.networkInformation.history, 1)
end
-- set position sent by server
setCCTPosition(self.controllerIndex, x, y, z)
-- move cct from the rest of the history queue
for i=1, table.getn(self.networkInformation.history) do
moveCCT(self.controllerIndex,
self.networkInformation.history[i].movementX,
self.networkInformation.history[i].movementY,
self.networkInformation.history[i].movementZ,
Player.movementCollisionMask)
end
-- set target physics index to current index
self.networkInformation.updateTargetTranslationPhysicsIndex = getPhysicsUpdateIndex() -- update until the current physics index is simulated
else
local isControlled = streamReadBool(streamId)
if isControlled ~= self.isControlled then
self:moveToAbsoluteInternal(x, y, z)
if isControlled then
self:onEnter(false)
else
```

```
831 self:onLeave()
832 end
833 else
834 -- other clients: set position, refrain from using target index and
    start new network interpolation phase
835 setTranslation(self.rootNode, x, y, z)
836 self.networkInformation.interpolatorPosition:setTargetPosition(x, y,
    z)
837 if isOnGround then
838 self.networkInformation.interpolatorOnGround:setTargetValue(1.0)
839 else
840 self.networkInformation.interpolatorOnGround:setTargetValue(0.0)
841 end
842 self.networkInformation.updateTargetTranslationPhysicsIndex = -1
843 self.networkInformation.interpolationTime:startNewPhaseNetwork()
844 -- force update
845 self:raiseActive()
846 end
847 -- [animation]
848 self.baseInformation.isCrouched = streamReadBool(streamId)
849 --
850 end
851 else
852 -- server code (read data from client)
853 if connection == self.networkInformation.creatorConnection then
854 -- we received translation information from client and we ask the
    physics to move the avatar; we insert the current physics index in the
    history queue and force an update()
855 local movementX=streamReadFloat32(streamId)
856 local movementY=streamReadFloat32(streamId)
857 local movementZ=streamReadFloat32(streamId)
858
859 local qx = streamReadFloat32(streamId)
860 local qy = streamReadFloat32(streamId)
861 local qz = streamReadFloat32(streamId)
862 local qw = streamReadFloat32(streamId)
863
864 local index = streamReadInt32(streamId)
865 local isControlled = streamReadBool(streamId)
866
```

```
moveCCT(self.controllerIndex, movementX, movementY, movementZ, Player.movementCollisionMask)

self.networkInformation.interpolationTime:startNewPhaseNetwork()
self.networkInformation.interpolatorQuaternion:setTargetQuaternion(qx, qy, qz, qw)
local physicsIndex = getPhysicsUpdateIndex()
table.insert(self.networkInformation.history, {index=index, physicsIndex = physicsIndex})

self.networkInformation.updateTargetTranslationPhysicsIndex = physicsIndex -- update until the current physics index is simulated

self:raiseActive()
if isControlled ~= self.isControlled then
if isControlled then
self:onEnter(false)
else
self:onLeave()
end
end
-- [animation]
self.baseInformation.isCrouched = streamReadBool(streamId)
--
if self.isCarryingObject then
self.networkInformation.rotateObject = streamReadBool(streamId)
if self.networkInformation.rotateObject then
self.networkInformation.rotateObjectInputH = streamReadFloat32(streamId)
self.networkInformation.rotateObjectInputV = streamReadFloat32(streamId)
end
end
end
end
```

writeUpdateStream

Description

Writes to network stream via update

Definition

writeUpdateStream(integer streamId, integer timestamp, table connection)

Arguments

integer streamId id of the stream to read
integer timestamp timestamp of the packet
table connection connection information

Code

```lua
function Player:writeUpdateStream(streamId, connection, dirtyMask)
if not connection:getIsServer() then
-- server code (send data to client)
local x, y, z = getTranslation(self.rootNode)
--print("SERVER (
"..tostring(self).."/"..tostring(self.controllerName).."/"..tostring(sel
): x/y/z="..tostring(x).." / "..tostring(y).." / "..tostring(z).."
self.sendIndex="..tostring(self.networkInformation.sendIndex))
streamWriteFloat32(streamId, x)
streamWriteFloat32(streamId, y)
streamWriteFloat32(streamId, z)

local x, _, z = localDirectionToLocal(self.cameraNode, getParent(self.ca
local alpha = math.atan2(x, z)
streamWriteFloat32(streamId, alpha)

streamWriteBool(streamId, self.isObjectInRange)
if self.isObjectInRange then
streamWriteFloat32(streamId, self.lastFoundObjectMass)
end
streamWriteBool(streamId, self.isCarryingObject)
streamWriteBool(streamId, self.baseInformation.isOnGroundPhysics)
local isOwner = connection == self.networkInformation.creatorConnection
if isOwner then
streamWriteInt32(streamId, self.networkInformation.sendIndex)
else
streamWriteBool(streamId, self.isControlled)
-- [animation]
local isCrouching = self.playerStateMachine:isActive("crouch")
streamWriteBool(streamId, isCrouching)
--
end
else
-- client code (send data to server)
if self.isOwner then
-- sending translation information to the server and reset the translati
streamWriteFloat32(streamId, self.networkInformation.tickTranslation[1])
```

```
937  streamWriteFloat32(streamId, self.networkInformation.tickTranslation[2])
938  streamWriteFloat32(streamId, self.networkInformation.tickTranslation[3])
939  self.networkInformation.tickTranslation[1] = 0.0
940  self.networkInformation.tickTranslation[2] = 0.0
941  self.networkInformation.tickTranslation[3] = 0.0
942  local x, y, z, w = getQuaternion(self.cameraNode)
943  streamWriteFloat32(streamId, x) -- ? ToDo: Utils.writeCompressedQuaterni
944  streamWriteFloat32(streamId, y)
945  streamWriteFloat32(streamId, z)
946  streamWriteFloat32(streamId, w)
947
948  streamWriteInt32(streamId, self.networkInformation.index)
949  streamWriteBool(streamId, self.isControlled)
950  -- [animation]
951  local isCrouching = self.playerStateMachine:isActive("crouch")
952  streamWriteBool(streamId, isCrouching)
953  --
954  if self.isCarryingObject then
955  streamWriteBool(streamId, self.networkInformation.rotateObject)
956  if self.networkInformation.rotateObject then
957  streamWriteFloat32(streamId, self.networkInformation.rotateObjectInputH)
958  streamWriteFloat32(streamId, self.networkInformation.rotateObjectInputV)
959  end
960  end
961  end
962  end
963  end
```

mouseEvent

Description

Function called when mouse is moved (call from BaseMission).

Definition

mouseEvent(float posX, float posY, bool isDown, bool isUp, button)

Arguments

float	posX	position of the mouse
float	posY	position of the mouse
bool	isDown	
bool	isUp	
button		

Code

```
972  function Player:mouseEvent(posX, posY, isDown, isUp, button)
973  end
```

getIsInputAllowed

Description

A function to check if input is allowed. Player is entered and is a client as well as the gui is visible.

Definition

getIsInputAllowed()

Return Values

bool true if input is allowed.

Code

```
978 function Player:getIsInputAllowed()
979 return self.isEntered and self.isClient and not g_gui:getIsGuiVisible()
980 end
```

updateAnimationParameters

Description

Updates the parameters that will drive the animation

Definition

updateAnimationParameters(float dt)

Arguments

float dt delta time in ms

Code

```
985 function Player:updateAnimationParameters(dt)
986 local dx = (self.networkInformation.interpolatorPosition.targetPositionX - self.networkInformation.interpolatorPosition.lastPositionX)
987 local dy = (self.networkInformation.interpolatorPosition.targetPositionY - self.networkInformation.interpolatorPosition.lastPositionY)
988 local dz = (self.networkInformation.interpolatorPosition.targetPositionZ - self.networkInformation.interpolatorPosition.lastPositionZ)
989 local vx = dx / self.networkInformation.interpolationTime.interpolationDuration * 1000.0
990 local vy = dy / self.networkInformation.interpolationTime.interpolationDuration * 1000.0
991 local vz = dz / self.networkInformation.interpolationTime.interpolationDuration * 1000.0
992 local dirX, dirZ = math.sin(self.graphicsRotY), math.cos(self.graphicsRotY)
993 local estimatedForwardVelocity = vx * dirX + vz * dirZ
994
```

```
995  if self.baseInformation.animDt ~= nil and
     self.baseInformation.animDt ~= 0 then
996  self.animationInformation.oldYaw = self.animationInformation.newYaw
997  self.animationInformation.newYaw = self.cameraRotY
998  self.animationInformation.estimatedYawVelocity =
     MathUtil.getAngleDifference(self.animationInformation.newYaw,
     self.animationInformation.oldYaw) / (self.baseInformation.animDt *
     0.001)
999  self.baseInformation.animDt = 0
1000 end
1001
1002 local params = self.animationInformation.parameters
1003 params.forwardVelocity.value = estimatedForwardVelocity
1004 params.verticalVelocity.value = vy
1005 params.yawVelocity.value =
     self.animationInformation.estimatedYawVelocity
1006 params.onGround.value = self.baseInformation.isOnGround
1007 params.inWater.value = self.baseInformation.isInWater
1008 params.isCrouched.value = self.baseInformation.isCrouched
1009 params.absForwardVelocity.value =
     math.abs(estimatedForwardVelocity)
1010 params.isCloseToGround.value = self.baseInformation.isCloseToGround
1011 params.isUsingChainsawHorizontal.value =
     self.baseInformation.isUsingChainsawHorizontal
1012 params.isUsingChainsawVertical.value =
     self.baseInformation.isUsingChainsawVertical
1013
1014 for _, parameter in pairs(self.animationInformation.parameters) do
1015 if parameter.type == 0 then
1016 setConditionalAnimationBoolValue(self.animationInformation.player,
     parameter.id, parameter.value)
1017 elseif parameter.type == 1 then
1018 setConditionalAnimationFloatValue(self.animationInformation.player,
     parameter.id, parameter.value)
1019 end
1020 end
1021 end
```

updateWaterParms

Description

Updates information related to the player and the water level. It is used for particle effects when plunging and checking if the player is in water.

Definition

updateWaterParms()

Code

```
1025  function Player:updateWaterParms()
1026  local _, y, _ = getWorldTranslation(self.rootNode)
1027  local deltaWater = y - g_currentMission.waterY -
      self.baseInformation.capsuleTotalHeight * 0.5
1028  local waterLevel = self.baseInformation.waterLevel
1029  local velocityY = 0.0
1030
1031  if deltaWater < -50 then
1032  return
1033  end
1034
1035  if not self.isEntered then
1036  velocityY =
      self.animationInformation.parameters.verticalVelocity.value
1037  else
1038  velocityY = self.motionInformation.velocity[2]
1039  end
1040
1041  self.baseInformation.wasInWater = self.baseInformation.isInWater
1042  self.baseInformation.isInWater = deltaWater <= waterLevel
1043
1044  if not self.baseInformation.wasInWater and
      self.baseInformation.isInWater and velocityY <
      self.baseInformation.plungedYVelocityThreshold then
1045  self.baseInformation.plungedInWater = true
1046  else
1047  self.baseInformation.plungedInWater = false
1048  end
1049  end
```

update

Description

Main update function for the player. Taking care of: fx, water parms, sound, player states, motion, debug, action events, hand tools, animation, object picking, IK

Definition

update(float dt)

Arguments

float dt delta time in ms

Code

```
1054  function Player:update(dt)
1055  self.time = self.time + dt
1056
```

```
1057  -- print(string.format("-- [Player:update][%s] isEntered(%s) isServer(%s) isClient(%s) isControlled(%s)", tostring(self), tostring(self.isEntered), tostring(self.isServer), tostring(self.isClient), tostring(self.isControlled)))
1058  if not self.isEntered and self.isClient and self.isControlled then
1059  self:updateFX()
1060  end
1061
1062  -- check if the player is on ground
1063  if self.isServer or self.isEntered then
1064  local _, _, isOnGround = getCCTCollisionFlags(self.controllerIndex)
1065  self.baseInformation.isOnGroundPhysics = isOnGround
1066  end
1067
1068  if self.isClient and self.isControlled then
1069  self:updateWaterParms()
1070  self:updateSound()
1071  end
1072
1073  if self.isEntered and self.isClient and not g_gui:getIsGuiVisible() then
1074  if not g_currentMission.isPlayerFrozen then
1075  self:updatePlayerStates()
1076  self.playerStateMachine:update(dt)
1077  self:updateKinematic()
1078  self:integrateMotion(dt)
1079  self:movePlayer(dt)
1080  self:recordPositionInformation()
1081  self:cameraBob(dt)
1082  self:debugDraw()
1083  self.playerStateMachine:debugDraw(dt)
1084
1085  if not g_currentMission.hasSpecialCamera then
1086  -- JK: This is called all the time and does not seem needed
1087  -- setCamera(self.cameraNode)
1088  end
1089
1090  if not self.walkingIsLocked then
1091  self.rotX = self.rotX - self.inputInformation.pitchCamera * g_gameSettings:getValue(GameSettings.SETTING.CAMERA_SENSITIVITY)
```

```
1092  self.rotY = self.rotY - self.inputInformation.yawCamera *
      g_gameSettings:getValue(GameSettings.SETTING.CAMERA_SENSITIVITY)
1093
1094  self.rotX = math.min(1.2, math.max(-1.5, self.rotX))
1095  setRotation(self.cameraNode, self.rotX, self.rotY, 0)
1096  setRotation(self.foliageBendingNode, 0, self.rotY, 0)
1097
1098  local isSwimming = self.playerStateMachine:isActive("swim")
1099  end
1100  self:updateActionEvents()
1101  end
1102  end
1103
1104  if self:hasHandtoolEquipped() then
1105  self.baseInformation.currentHandtool:update(dt,
      self:getIsInputAllowed())
1106  end
1107
1108  self:updateInterpolation()
1109
1110  if self.isClient and self.isControlled then
1111  self:updateRotation(dt)
1112  end
1113
1114  -- animation
1115  if self.isClient and self.isControlled and not self.isEntered
      then
1116  self:updateAnimationParameters(dt)
1117  updateConditionalAnimation(self.animationInformation.player, dt)
1118
1119  -- Animation Debug
1120  -- local a,b,c = getWorldTranslation(self.rootNode)
1121  -- b = b + 1.0
1122  --
      conditionalAnimationDebugDraw(self.animationInformation.player,
      a,b,c)
1123  end
1124
1125  -- objects in front of player
1126  self:checkObjectInRange()
1127  -- if self.isEntered or (self.isControlled and
      self.networkInformation.interpolationTime.isDirty) then
```

```
1128  if self.isEntered or
      self.networkInformation.interpolationTime.isDirty then
1129  self:raiseActive()
1130  end
1131
1132  if self.isClient and self.isControlled and not self.isEntered
      and self.networkInformation.rotateObject then
1133  self:rotateObject(self.networkInformation.rotateObjectInputV,
      1.0, 0.0, 0.0)
1134  self:rotateObject(self.networkInformation.rotateObjectInputH,
      0.0, 1.0, 0.0)
1135  end
1136  self:resetInputsInformation()
1137  end
```

checkObjectInRange

Description

Definition

checkObjectInRange()

Code

```
1141  function Player:checkObjectInRange()
1142  -- handle picking up of objects
1143  if self.isControlled then
1144  if self.isServer then
1145  if not self.isCarryingObject then
1146  local x,y,z = localToWorld(self.cameraNode, 0,0,1.0)
1147  local dx,dy,dz = localDirectionToWorld(self.cameraNode, 0,0,-1)
1148  self.lastFoundObject = nil
1149  self.lastFoundObjectHitPoint = nil
1150  raycastAll(x,y,z, dx,dy,dz, "pickUpObjectRaycastCallback",
      Player.MAX_PICKABLE_OBJECT_DISTANCE, self)
1151  self.isObjectInRange = self.lastFoundObject ~= nil
1152  else
1153  -- check if object still exists
1154  if self.pickedUpObject ~= nil then
1155  if not entityExists(self.pickedUpObject) then
1156  Player.PICKED_UP_OBJECTS[self.pickedUpObject] = false
1157  self.pickedUpObject = nil
1158  self.pickedUpObjectJointId = nil
1159  self.isCarryingObject = false
1160  end
1161  end
1162  end
```

```
1163 end
1164 end
1165 end
```

updateTick

Description

Update function called when network ticks. Takes care of movements/position, player state machine, interpolation phase, physics, handtools. Inputs are reset here.

Definition

updateTick(float dt)

Arguments

float dt delta time in ms

Code

```
1170 function Player:updateTick(dt)
1171 self.playerStateMachine:updateTick(dt)
1172
1173 if self:hasHandtoolEquipped() then
1174 self.baseInformation.currentHandtool:updateTick(dt,
     self:getIsInputAllowed())
1175 end
1176 self:updateNetworkMovementHistory()
1177
1178 -- -------------------------
1179 -- @todo: check and add again
1180 -- -------------------------
1181 -- local xt, yt, zt = getTranslation(self.rootNode)
1182 -- --[[
1183 -- if GS_PLATFORM_TYPE == GS_PLATFORM_TYPE_PS4 or
     GS_PLATFORM_TYPE == GS_PLATFORM_TYPE_XBOXONE then
1184 -- xt = xt + self.movementX
1185 -- zt = zt + self.movementZ
1186 -- yt =
     getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, xt,
     300, zt) + self.height
1187 -- setTranslation(self.rootNode, xt, yt, zt)
1188 -- end
1189 -- --]]
1190 -- -------------------------
1191 if self.isServer and self.isControlled then
1192 if GS_PLATFORM_TYPE ~= GS_PLATFORM_TYPE_PC then
1193 local x, y, z = getTranslation(self.rootNode)
1194 local paramsXZ = g_currentMission.vehicleXZPosCompressionParams
```

```
1195  if not NetworkUtil.getIsWorldPositionInCompressionRange(x,
      paramsXZ) or
1196  not NetworkUtil.getIsWorldPositionInCompressionRange(z,
      paramsXZ) or
1197  getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, x,
      y, z) > y + 20
1198  then
1199  self:moveTo(g_currentMission.playerStartX,
      g_currentMission.playerStartY, g_currentMission.playerStartZ,
      g_currentMission.playerStartIsAbsolute, false)
1200  return
1201  end
1202  end
1203  end
1204  -- ------------------------
1205  end
```

updateActionEvents

Description

Update action event states and input hint display.

Definition

updateActionEvents()

Code

```
1209  function Player:updateActionEvents()
1210  -- light
1211  local isDark = g_currentMission.environment.currentHour > 20 or
      g_currentMission.environment.currentHour < 7
1212  local eventIdToggleLight =
      self.inputInformation.registrationList[InputAction.TOGGLE_LIGHTS_FPS].e
1213  if self.playerStateMachine:isAvailable("useLight") and isDark then
1214  g_inputBinding:setActionEventTextVisibility(eventIdToggleLight, isDark
      self.lightNode ~= nil)
1215  end
1216
1217  -- tools
1218  local eventIdNextHandTool =
      self.inputInformation.registrationList[InputAction.NEXT_HANDTOOL].event
1219  local eventIdPrevHandTool =
      self.inputInformation.registrationList[InputAction.PREVIOUS_HANDTOOL].e
1220  if self.playerStateMachine:isAvailable("cycleHandtool") then
1221  g_inputBinding:setActionEventActive(eventIdNextHandTool, true)
1222  g_inputBinding:setActionEventActive(eventIdPrevHandTool, true)
1223  g_inputBinding:setActionEventTextVisibility(eventIdNextHandTool, true)
1224  g_inputBinding:setActionEventTextVisibility(eventIdPrevHandTool, true)
```

```
1225  else
1226  g_inputBinding:setActionEventActive(eventIdNextHandTool, false)
1227  g_inputBinding:setActionEventActive(eventIdPrevHandTool, false)
1228  g_inputBinding:setActionEventTextVisibility(eventIdNextHandTool, false)
1229  g_inputBinding:setActionEventTextVisibility(eventIdPrevHandTool, false)
1230  end
1231
1232  local eventIdUseHandTool =
      self.inputInformation.registrationList[InputAction.ACTIVATE_HANDTOOL].e
1233  if self:hasHandtoolEquipped() then
1234  g_inputBinding:setActionEventActive(eventIdUseHandTool, true)
1235  g_inputBinding:setActionEventTextVisibility(eventIdUseHandTool, true)
1236  else
1237  g_inputBinding:setActionEventActive(eventIdUseHandTool, false)
1238  g_inputBinding:setActionEventTextVisibility(eventIdUseHandTool, false)
1239  end
1240
1241  local eventIdInteract =
      self.inputInformation.registrationList[InputAction.INTERACT].eventId
1242  if self.playerStateMachine:isAvailable("drop") then
1243  g_inputBinding:setActionEventText(eventIdInteract,
      g_i18n:getText("action_dropObject"))
1244  g_inputBinding:setActionEventActive(eventIdInteract, true)
1245  g_inputBinding:setActionEventTextVisibility(eventIdInteract, true)
1246  elseif self.playerStateMachine:isAvailable("pickup") then
1247  g_inputBinding:setActionEventText(eventIdInteract,
      string.format(g_i18n:getText("action_pickUpObject"), self.lastFoundObje
1248  g_inputBinding:setActionEventActive(eventIdInteract, true)
1249  g_inputBinding:setActionEventTextVisibility(eventIdInteract, true)
1250  elseif self.playerStateMachine:isAvailable("animalInteract") or
      self.playerStateMachine:isActive("animalInteract") then
1251  local animalInteractState = self.playerStateMachine:getState("animalInt
1252  local animalInteractText = string.format(g_i18n:getText("action_interac
      animalInteractState.interactText)
1253  g_inputBinding:setActionEventText(eventIdInteract, animalInteractText)
1254  g_inputBinding:setActionEventActive(eventIdInteract, true)
1255  g_inputBinding:setActionEventTextVisibility(eventIdInteract, true)
1256  elseif self.inputInformation.interactState == Player.BUTTONSTATES.RELEA
1257  g_inputBinding:setActionEventActive(eventIdInteract, false)
1258  g_inputBinding:setActionEventTextVisibility(eventIdInteract, false)
1259  end
1260
```

```lua
local eventIdActivateObject = self.inputInformation.registrationList[InputAction.ACTIVATE_OBJECT].eve
if self.playerStateMachine:isAvailable("activateObject") then
local activateObjectState = self.playerStateMachine:getState("activateO
g_inputBinding:setActionEventText(eventIdActivateObject, activateObjectState.activateText)
g_inputBinding:setActionEventActive(eventIdActivateObject, true)
g_inputBinding:setActionEventTextVisibility(eventIdActivateObject, true
elseif self.playerStateMachine:isAvailable("animalFeed") then
g_inputBinding:setActionEventText(eventIdActivateObject, g_i18n:getText("action_feedAnimal"))
g_inputBinding:setActionEventActive(eventIdActivateObject, true)
g_inputBinding:setActionEventTextVisibility(eventIdActivateObject, true
elseif self.playerStateMachine:isAvailable("animalPet") then
g_inputBinding:setActionEventText(eventIdActivateObject, g_i18n:getText("action_petAnimal"))
g_inputBinding:setActionEventActive(eventIdActivateObject, true)
g_inputBinding:setActionEventTextVisibility(eventIdActivateObject, true
else
g_inputBinding:setActionEventActive(eventIdActivateObject, false)
g_inputBinding:setActionEventTextVisibility(eventIdActivateObject, fals
end

-- enter vehicle or ride animal
self.canRideAnimal = self.playerStateMachine:isAvailable("animalRide")
self.canEnterVehicle = g_currentMission.interactiveVehicleInRange and g_currentMission.interactiveVehicleInRange:getIsEnterable()
local vehicleIsRideable = self.canEnterVehicle and SpecializationUtil.hasSpecialization(Rideable, g_currentMission.interactiveVehicleInRange.specializations)
local eventIdEnter = self.inputInformation.registrationList[InputAction.ENTER].eventId
if self.canEnterVehicle and not vehicleIsRideable then
g_inputBinding:setActionEventText(eventIdEnter, g_i18n:getText("button_enterVehicle"))
g_inputBinding:setActionEventActive(eventIdEnter, true)
g_inputBinding:setActionEventTextVisibility(eventIdEnter, true)
elseif self.canRideAnimal or vehicleIsRideable then
local rideableName = ""
if self.canRideAnimal then
local rideState = self.playerStateMachine:getState("animalRide")
rideableName = rideState:getRideableName()
```

```
1294  elseif vehicleIsRideable then
1295  rideableName = g_currentMission.interactiveVehicleInRange:getFullName()
1296  end
1297  g_inputBinding:setActionEventText(eventIdEnter,
      string.format(g_i18n:getText("action_rideAnimal"), rideableName))
1298  g_inputBinding:setActionEventActive(eventIdEnter, true)
1299  g_inputBinding:setActionEventTextVisibility(eventIdEnter, true)
1300  else
1301  g_inputBinding:setActionEventActive(eventIdEnter, false)
1302  g_inputBinding:setActionEventTextVisibility(eventIdEnter, false)
1303  end
1304
1305  local eventIdThrowObject =
      self.inputInformation.registrationList[InputAction.THROW_OBJECT].eventI
1306  if self.playerStateMachine:isAvailable("throw") then
1307  g_inputBinding:setActionEventActive(eventIdThrowObject, true)
1308  g_inputBinding:setActionEventTextVisibility(eventIdThrowObject, true)
1309  else
1310  g_inputBinding:setActionEventActive(eventIdThrowObject, false)
1311  g_inputBinding:setActionEventTextVisibility(eventIdThrowObject, false)
1312  end
1313
1314  local eventIdObjectRotateHorizontally =
      self.inputInformation.registrationList[InputAction.ROTATE_OBJECT_LEFT_R
1315  local eventIdObjectRotateVertically =
      self.inputInformation.registrationList[InputAction.ROTATE_OBJECT_UP_DOW
1316  if self.isCarryingObject then
1317  g_inputBinding:setActionEventActive(eventIdObjectRotateHorizontally, tr
1318  g_inputBinding:setActionEventTextVisibility(eventIdObjectRotateHorizont
1319  g_inputBinding:setActionEventActive(eventIdObjectRotateVertically, true
1320  g_inputBinding:setActionEventTextVisibility(eventIdObjectRotateVertical
1321  else
1322  g_inputBinding:setActionEventActive(eventIdObjectRotateHorizontally, fa
1323  g_inputBinding:setActionEventTextVisibility(eventIdObjectRotateHorizont
1324  g_inputBinding:setActionEventActive(eventIdObjectRotateVertically, fals
1325  g_inputBinding:setActionEventTextVisibility(eventIdObjectRotateVertical
1326  end
1327
1328  -- debug movements
1329  local eventIdDebugFlyToggle =
      self.inputInformation.registrationList[InputAction.DEBUG_PLAYER_ENABLE]
```

```
1330  g_inputBinding:setActionEventActive(eventIdDebugFlyToggle,
      g_flightAndNoHUDKeysEnabled)
1331  local eventIdDebugFlyUpDown =
      self.inputInformation.registrationList[InputAction.DEBUG_PLAYER_UP_DOWN
1332  g_inputBinding:setActionEventActive(eventIdDebugFlyUpDown,
      g_flightAndNoHUDKeysEnabled)
1333  end
```

updateInterpolation

Description

Updates interpolations for physics, camera and position

Definition

updateInterpolation()

Code

```
1337  function Player:updateInterpolation()
1338  if self.isEntered then
1339  local xt, yt, zt = getTranslation(self.rootNode)
1340  self.networkInformation.interpolatorPosition:setTargetPosition(xt, yt,
1341
1342  if self.baseInformation.isOnGroundPhysics then
1343  self.networkInformation.interpolatorOnGround:setTargetValue(1.0)
1344  else
1345  self.networkInformation.interpolatorOnGround:setTargetValue(0.0)
1346  end
1347  self.networkInformation.interpolationTime:startNewPhase(75)
1348  elseif self.networkInformation.updateTargetTranslationPhysicsIndex >= 0
1349  if getIsPhysicsUpdateIndexSimulated(self.networkInformation.updateTarge
1350  self.networkInformation.updateTargetTranslationPhysicsIndex = -1
1351  end
1352  local xt, yt, zt = getTranslation(self.rootNode)
1353  self.networkInformation.interpolatorPosition:setTargetPosition(xt, yt,
1354  if self.baseInformation.isOnGroundPhysics then
1355  self.networkInformation.interpolatorOnGround:setTargetValue(1.0)
1356  else
1357  self.networkInformation.interpolatorOnGround:setTargetValue(0.0)
1358  end
1359  self.networkInformation.interpolatorQuaternion:setTargetQuaternion(self
      self.networkInformation.interpolatorQuaternion.targetQuaternionY, self.
      self.networkInformation.interpolatorQuaternion.targetQuaternionW)
1360  self.networkInformation.interpolationTime:startNewPhase(75)
1361  end
1362
1363  if self.isControlled then
```

```
1364  local needsCameraInterp = self.isServer and not self.isEntered
1365  local needsPositionInterp = self.isClient
1366
1367  if needsCameraInterp or needsPositionInterp then
1368  if self.networkInformation.interpolationTime.isDirty then
1369  self.networkInformation.interpolationTime:update(g_physicsDtUnclamped)
1370  if needsCameraInterp then
1371  local qx, qy, qz, qw =
      self.networkInformation.interpolatorQuaternion:getInterpolatedValues(se
1372
1373  setQuaternion(self.cameraNode, qx, qy, qz, qw)
1374  end
1375  if needsPositionInterp then
1376  local x, y, z =
      self.networkInformation.interpolatorPosition:getInterpolatedValues(self
1377  setTranslation(self.graphicsRootNode, x, y - self.baseInformation.capsu
1378  local isOnGroundFloat =
      self.networkInformation.interpolatorOnGround:getInterpolatedValue(self.
1379  self.baseInformation.isOnGround = isOnGroundFloat > 0.9
1380  self.baseInformation.isCloseToGround = isOnGroundFloat > 0.5
1381  end
1382  end
1383  end
1384  end
1385  end
```

updateNetworkMovementHistory

Description

Definition

updateNetworkMovementHistory()

Code

```
1389  function Player:updateNetworkMovementHistory()
1390  if self.isEntered and self.isClient then
1391  self:raiseDirtyFlags(self.networkInformation.dirtyFlag)
1392  elseif self.isServer and not self.isEntered and self.isControlled then
1393  -- find the latest index, which is already simulated now
1394  local latestSimulatedIndex = -1
1395  while self.networkInformation.history[1] ~= nil and
      getIsPhysicsUpdateIndexSimulated(self.networkInformation.history[1].phy
      do
1396  latestSimulatedIndex = self.networkInformation.history[1].index
1397  table.remove(self.networkInformation.history, 1)
1398  end
```

```
1399  if latestSimulatedIndex >= 0 then
1400  self.networkInformation.sendIndex = latestSimulatedIndex
1401  self:raiseDirtyFlags(self.networkInformation.dirtyFlag)
1402  end
1403  end
1404  end
```

updateRotation

Description

Updates rotation of player avatar over the network

Definition

updateRotation(float dt)

Arguments

float dt delta time in ms

Code

```
1409  function Player:updateRotation(dt)
1410  if not self.isEntered then
1411  local animDt = 60
1412  self.animUpdateTime = self.animUpdateTime + dt
1413  if self.animUpdateTime > animDt then
1414  if self.isServer then
1415  local x, _, z = localDirectionToLocal(self.cameraNode,
      getParent(self.cameraNode), 0, 0, 1)
1416  local alpha = math.atan2(x, z)
1417  self.cameraRotY = alpha
1418  end
1419
1420  local x, y, z = getTranslation(self.graphicsRootNode)
1421  local dx, _, dz = x - self.lastAnimPosX, y - self.lastAnimPosY, z
      - self.lastAnimPosZ
1422  local dirX, dirZ = -math.sin(self.cameraRotY), -
      math.cos(self.cameraRotY)
1423  local movementDist = dx * dirX + dz * dirZ -- Note: |dir| = 1
1424
1425  if (dx * dx + dz * dz) < 0.001 then
1426  self.targetGraphicsRotY = self.cameraRotY + math.rad(180.0)
1427  else
1428  if movementDist > -0.001 then
1429  self.targetGraphicsRotY = math.atan2(dx, dz)
1430  else
1431  self.targetGraphicsRotY = math.atan2(-dx, -dz)
1432  end
```

```
1433  end
1434
1435  dirX, dirZ = -math.sin(self.targetGraphicsRotY), -
      math.cos(self.targetGraphicsRotY)
1436  movementDist = dx * dirX + dz * dirZ -- Note: |dir| = 1
1437
1438  movementDist = self.walkDistance * 0.2 + movementDist * 0.8
1439  self.walkDistance = movementDist
1440
1441  self.lastAnimPosX = x
1442  self.lastAnimPosY = y
1443  self.lastAnimPosZ = z
1444
1445  self.baseInformation.animDt = self.animUpdateTime
1446  self.animUpdateTime = 0
1447  end
1448
1449  self.targetGraphicsRotY =
      MathUtil.normalizeRotationForShortestPath(self.targetGraphicsRotY,
      self.graphicsRotY)
1450  local maxDeltaRotY = math.rad(0.5) * dt
1451  self.graphicsRotY = math.min(math.max(self.targetGraphicsRotY,
      self.graphicsRotY - maxDeltaRotY), self.graphicsRotY +
      maxDeltaRotY)
1452
1453  setRotation(self.skeletonRootNode, 0, self.graphicsRotY, 0)
1454  end
1455  end
```

getPositionData

Description

Let position information of the root node of the player

Definition

getPositionData()

Return Values

float posX	x position of player
float posY	y position of player
float posZ	z position of player
float graphicsRotY	rotation of the player

Code

```
1463  function Player:getPositionData()
1464  local posX, posY, posZ = getTranslation(self.rootNode)
1465  if self.isClient and self.isControlled and self.isEntered then
```

```
1466  return posX, posY, posZ, self.rotY
1467  else
1468  return posX, posY, posZ, self.graphicsRotY
1469  end
1470  end
```

setIKDirty

Description

Sets all ik chain node to dirty so that they are recalculated

Definition

setIKDirty()

Code

```
1474  function Player:setIKDirty()
1475  IKUtil.setIKChainDirty(self.ikChains, "rightFoot")
1476  IKUtil.setIKChainDirty(self.ikChains, "leftFoot")
1477  IKUtil.setIKChainDirty(self.ikChains, "rightArm")
1478  IKUtil.setIKChainDirty(self.ikChains, "leftArm")
1479  IKUtil.setIKChainDirty(self.ikChains, "spine")
1480  end
```

lockInput

Description

Locks player input

Definition

lockInput(bool locked)

Arguments

bool locked if true, will lock input

Code

```
1485  function Player:lockInput(locked)
1486  self.lockedInput = locked
1487  end
```

moveTo

Description

Moves the player to the given position, with the given y offset to the terrain

Definition

moveTo(float x, float y, float z, bool isAbsolute)

Arguments

float x	new x position
float y	new y position
float z	new z position

bool isAbsolute if true, Y coordinate is in absolute, not calculated from terrain height

Code

```
1495  function Player:moveTo(x, y, z, isAbsolute, isRootNode)
1496  if not self.isServer and self.isOwner then
```

```
1497  g_client:getServerConnection():sendEvent(PlayerTeleportEvent:new(x, y, z, isAbsolute, isRootNode))
1498  end
1499  if not isAbsolute then
1500  local terrainHeight = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, x, 300, z)
1501  y = terrainHeight + y
1502  end
1503  if not isRootNode then
1504  y = y + self.baseInformation.capsuleTotalHeight * 0.5
1505  end
1506  self:moveToAbsoluteInternal(x, y, z)
1507  end
```

moveToAbsolute

Description

Moves the player root node to the given position, such that the feet are at x, y, z

Definition

moveToAbsolute(float x, float y, float z)

Arguments

float x new x position
float y new y position
float z new z position

Code

```
1514  function Player:moveToAbsolute(x, y, z)
1515  self:moveTo(x, y, z, true, false)
1516  end
```

moveRootNodeToAbsolute

Description

Moves the player to the given position, such that the root node is at x, y, z

Definition

moveRootNodeToAbsolute(float x, float y, float z)

Arguments

float x new x position
float y new y position
float z new z position

Code

```
1523  function Player:moveRootNodeToAbsolute(x, y, z)
1524  self:moveTo(x, y, z, true, true)
1525  end
```

moveToExitPoint

Description

Moves player to vehicle exit node

Definition

moveToExitPoint(table exitVehicle)

Arguments

table exitVehicle vehicle class that will be used to get the exit node to place the player

Code

```
1530  function Player:moveToExitPoint(exitVehicle)
1531  local exitPoint
1532  if exitVehicle.getExitNode ~= nil then
1533  exitPoint = exitVehicle:getExitNode()
1534  end
1535  local x, y, z = getWorldTranslation(exitPoint)
1536  local terrainHeight = getTerrainHeightAtWorldPos(g_currentMission.terrainRootNode, x, 300, z)
1537  y = math.max(terrainHeight + 0.1, y + 0.9)
1538  self:moveToAbsolute(x, y, z)
1539  local dx, _, dz = localDirectionToWorld(exitPoint, 0, 0, -1)
1540  self.rotY = MathUtil.getYRotationFromDirection(dx, dz)
1541
1542  --self.targetGraphicsRotY = self.rotY
1543  --self.graphicsRotY = self.rotY
1544  --(I) setRotation(self.graphicsRootNode, 0, self.graphicsRotY, 0)
1545  setRotation(self.cameraNode, self.rotX, self.rotY, 0)
1546  end
```

setRotation

Description

Set player rotation

Definition

setRotation(float rotX, float rotY)

Arguments

float rotX set rotation x parameter; vertical X rotation is clamped

float rotY set rotation y parameter (graphics node, target graphics and camera)

Code

```
1552  function Player:setRotation(rotX, rotY)
1553  self.rotX = math.min(1.2, math.max(-1.5, rotX))
1554  self.rotY = rotY
1555  self.graphicsRotY = rotY
1556  self.cameraRotY = rotY
1557  self.targetGraphicsRotY = rotY
1558  end
```

moveToAbsoluteInternal

Description

Move player root node and graphics node to a specific position. Updates interpolation parameters

Definition

moveToAbsoluteInternal(float x, float y, float z)

Arguments

float x new x position

float y new y position

float z new z position

Code

```
1565  function Player:moveToAbsoluteInternal(x, y, z)
1566  setTranslation(self.rootNode, x, y, z)
1567  setTranslation(self.graphicsRootNode, x, y -
      self.baseInformation.capsuleTotalHeight * 0.5, z)
1568
1569  self.networkInformation.interpolationTime:reset()
1570  self.networkInformation.interpolatorPosition:setPosition(x,y,z)
1571  self.networkInformation.updateTargetTranslationPhysicsIndex = -1
1572
1573  self.lastAnimPosX = x
1574  self.lastAnimPosY = y
1575  self.lastAnimPosZ = z
1576  self.walkDistance = 0
1577
1578  self.baseInformation.lastPosition[1],
1579  self.baseInformation.lastPosition[2],
1580  self.baseInformation.lastPosition[3] =
      getTranslation(self.rootNode)
1581  self.baseInformation.position[1],
1582  self.baseInformation.position[2],
1583  self.baseInformation.position[3] = getTranslation(self.rootNode)
1584  end
```

drawUIInfo

Description

Renders UI information for the player

Definition

drawUIInfo()

Code

```
1588  function Player:drawUIInfo()
1589  if self.isClient and self.isControlled and not self.isEntered
      then
```

```
1590  if not g_gui:getIsGuiVisible() and not
      g_flightAndNoHUDKeysEnabled then
1591  local x, y, z = getTranslation(self.graphicsRootNode)
1592  local x1, y1, z1 = getWorldTranslation(getCamera())
1593  local diffX = x - x1
1594  local diffY = y - y1
1595  local diffZ = z - z1
1596  local dist = MathUtil.vector3LengthSq(diffX, diffY, diffZ)
1597  if dist <= 100 * 100 then
1598  y = y + self.baseInformation.tagOffset[2]
1599  Utils.renderTextAtWorldPosition(x, y, z,
      self.visualInformation.playerName, getCorrectTextSize(0.02), 0)
1600  end
1601  end
1602  end
1603  end
```

draw

Description

Draws overlay information

Definition

draw()

Code

```
1607  function Player:draw()
1608  if self:getIsInputAllowed() then
1609  if self:hasHandtoolEquipped() then
1610  self.baseInformation.currentHandtool:draw()
1611  elseif not self.isCarryingObject and self.isObjectInRange and not
      self:hasHandtoolEquipped() then
1612  if not g_flightAndNoHUDKeysEnabled and self.pickedUpObjectOverlay ~=
      nil then
1613  self.pickedUpObjectOverlay:setDimension(self.pickedUpObjectWidth,
      self.pickedUpObjectHeight)
1614  self.pickedUpObjectOverlay:setUVs(self.pickedUpObjectHandUVs)
1615  self.pickedUpObjectOverlay:render()
1616  end
1617  elseif not g_flightAndNoHUDKeysEnabled and self.pickedUpObjectOverlay
      ~= nil then
1618  self.pickedUpObjectOverlay:setDimension(self.pickedUpObjectAimingWidth,
      self.pickedUpObjectAimingHeight)
1619  self.pickedUpObjectOverlay:setUVs(self.pickedUpObjectAimingUVs)
1620  self.pickedUpObjectOverlay:render()
1621  end
```

```
1622  end
1623  end
```

onEnter

Description

Called when player enters mission. Sets player mesh visibility. Update traffic system with player info. Register player action events.

Definition

onEnter(bool isControlling)

Arguments

bool isControlling true if controlled

Code

```
1629  function Player:onEnter(isControlling)
1630  self:raiseActive()
1631  if self.foliageBendingNode ~= nil and self.foliageBendingId == nil then
1632  -- foliage bending
1633  self.foliageBendingId = g_currentMission.foliageBendingSystem:createRectangle(-0.5, 0.5, -0.5, 0.5, 0.4, self.foliageBendingNode)
1634  end
1635
1636  if self.isServer then
1637  self:setOwner(self.networkInformation.creatorConnection)
1638  end
1639  if isControlling or self.isServer then
1640  self:raiseDirtyFlags(self.networkInformation.dirtyFlag)
1641  end
1642
1643  self.baseInformation.lastPosition[1],
1644  self.baseInformation.lastPosition[2],
1645  self.baseInformation.lastPosition[3] = getTranslation(self.rootNode)
1646  self.baseInformation.position[1],
1647  self.baseInformation.position[2],
1648  self.baseInformation.position[3] = getTranslation(self.rootNode)
1649
1650  self.isControlled = true
1651  if isControlling then
1652  g_currentMission:addPauseListeners(self, Player.onPausGame)
1653  setRotation(self.cameraNode, 0, 0, 0)
1654  setCamera(self.cameraNode)
1655  self.isEntered = true
1656  self:setVisibility(false)
```

```
1657  self:registerActionEvents()
1658  else
1659  self:setVisibility(true)
1660  end
1661  if self.isServer and not self.isEntered and
      g_currentMission.trafficSystem ~= nil and
      g_currentMission.trafficSystem.trafficSystemId ~= 0 then
1662  addTrafficSystemPlayer(g_currentMission.trafficSystem.trafficSystemId,
      self.graphicsRootNode)
1663  end
1664  end
```

onLeaveVehicle

Description

Called when player leaves vehicle

Definition

onLeaveVehicle()

Code

```
1668  function Player:onLeaveVehicle()
1669  self.playerStateMachine:deactivateState("animalRide")
1670  end
```

onLeave

Description

Called when player Leaves mission. Update traffic system to ignore this player. Clear position history, visibility. Removes tools. Deregister from companion animal system. Moves to (0, -200, 0)

Definition

onLeave()

Code

```
1674  function Player:onLeave()
1675  -- return dog to it's house
1676  local dogHouse = g_currentMission:getDoghouse(self.farmId)
1677  if dogHouse ~= nil and dogHouse.dog.entityFollow == self.rootNode then
1678  dogHouse.dog:idleStay()
1679  end
1680  -- stop swim sound
1681  if self.soundInformation.isSampleSwinPlaying then
1682  g_soundManager:stopSample(self.soundInformation.samples.swim)
1683  self.soundInformation.isSampleSwinPlaying = false
1684  end
1685
1686  self:removeActionEvents()
1687  -- unregister animal interess
```

```
1688 for husbandryId, _ in pairs(g_currentMission.husbandries) do
1689 setAnimalInteressNode(husbandryId, 0)
1690 end
1691
1692 if self.foliageBendingId ~= nil then
1693 g_currentMission.foliageBendingSystem:destroyObject(self.foliageBending
1694 self.foliageBendingId = nil
1695 end
1696
1697 if self.isServer then
1698 self:setOwner(nil)
1699 end
1700 if self.isEntered or self.isServer then
1701 self:raiseDirtyFlags(self.networkInformation.dirtyFlag)
1702 end
1703 if self.isServer and not self.isEntered and g_currentMission.trafficSys
     ~= nil and g_currentMission.trafficSystem.trafficSystemId ~= 0 then
1704 removeTrafficSystemPlayer(g_currentMission.trafficSystem.trafficSystemI
     self.graphicsRootNode)
1705 end
1706
1707 g_currentMission:addPauseListeners(self)
1708
1709 --clear history
1710 self.networkInformation.history = {}
1711 self.isControlled = false
1712 self.isEntered = false
1713 self:setVisibility(false)
1714
1715 if self:hasHandtoolEquipped() then
1716 self.baseInformation.currentHandtool:onDeactivate()
1717 self.baseInformation.currentHandtool:delete()
1718 self.baseInformation.currentHandtool = nil
1719 end
1720 local dogHouse = g_currentMission:getDoghouse(self.farmId)
1721 if dogHouse ~= nil then
1722 dogHouse:onPlayerLeave()
1723 end
1724
1725 if self.lightNode ~= nil then
1726 setVisibility(self.lightNode, false)
```

```
1727  end
1728  self:moveToAbsoluteInternal(0, -200, 0)
1729  end
```

setVisibility

Description

Sets third person mesh visibility

Definition

setVisibility(bool visibility)

Arguments

bool visibility if true will update visibility accordingly.

Code

```
1734  function Player:setVisibility(visibility)
1735  if self.meshThirdPerson ~= nil then
1736  setVisibility(self.meshThirdPerson, visibility)
1737  self.visualInformation:setVisibility(visibility)
1738  end
1739  end
```

setWoodWorkVisibility

Description

Sets visibility for gloves and helmet. Gloves code is deprecated.

Definition

setWoodWorkVisibility(bool visibilityGloves, bool visibilityHelmet)

Arguments

bool visibilityGloves if true will update visibility of gloves accordingly.

bool visibilityHelmet if true will update visibility of helmet accordingly.

Code

```
1745  function Player:setWoodWorkVisibility(state, uvs)
1746  if self.isEntered then
1747  self.visualInformation:setProtectiveVisibility(false)
1748  end
1749  self.visualInformation:setProtectiveVisibility(state)
1750  end
```

testScope

Description

Check if a position is within clip distance

Definition

testScope(float x, float y, float z, float coeff)

Arguments

float x	world x position
float y	world y position
float z	world z position

float coeff parameter is unused

Return Values

bool returns true if distance to player root node is lower than clip distance

Code

```
1759  function Player:testScope(x, y, z, coeff)
1760  local x1, y1, z1 = getTranslation(self.rootNode)
1761  local dist = MathUtil.vector3Length(x1 - x, y1 - y, z1 - z)
1762  local clipDist = self.clipDistance
1763  if dist < clipDist * clipDist then
1764  return true
1765  else
1766  return false
1767  end
1768  end
```

onGhostRemove

Description

Deletes player

Definition

onGhostRemove()

Code

```
1772  function Player:onGhostRemove()
1773  self:delete()
1774  end
```

onGhostAdd

Description

Empty function

Definition

onGhostAdd()

Code

```
1778  function Player:onGhostAdd()
1779  end
```

getUpdatePriority

Description

Calculate a priority value from the position of the root node and the clpi distance

Definition

getUpdatePriority(float skipCount, float x, float y, float z, float coeff, table connection)

Arguments

float skipCount	
float x	world x position
float y	world y position
float z	world z position
float coeff	parameter is unused
table connection	structure containing connection information

Return Values

float returns calculated priority

Code

```
1790  function Player:getUpdatePriority(skipCount, x, y, z, coeff,
      connection)
1791  if self.owner == connection then
1792  return 50
1793  end
1794  local x1, y1, z1 = getTranslation(self.rootNode)
1795  local dist = MathUtil.vector3Length(x1 - x, y1 - y, z1 - z)
1796  local clipDist = self.clipDistance
1797  return (1 - dist / clipDist) * 0.8 + 0.5 * skipCount * 0.2
1798  end
```

consoleCommandToggleFlightAndNoHUDMode

Description

Toggle flight mode

Definition

consoleCommandToggleFlightAndNoHUDMode()

Return Values

string that will be displayed on console

Code

```
1803  function Player:consoleCommandToggleFlightAndNoHUDMode()
1804  g_flightAndNoHUDKeysEnabled = not g_flightAndNoHUDKeysEnabled
1805  if not g_flightAndNoHUDKeysEnabled then
1806  self.debugFlightMode = false -- force reset flight mode
1807  end
1808
1809  return "PlayerFlightAndNoHUDMode = " ..
      tostring(g_flightAndNoHUDKeysEnabled)
1810  end
```

consoleCommandToggleWoodCuttingMaker

Description

Toggle wood cutting marker

Definition

consoleCommandToggleWoodCuttingMaker(table unusedSelf)

Arguments

table unusedSelf unused parameter

Return Values

string that will be displayed on console

Code

```
1816  function Player.consoleCommandToggleWoodCuttingMaker(unusedSelf)
1817  g_woodCuttingMarkerEnabled = not g_woodCuttingMarkerEnabled
```

```
1818  return "WoodCuttingMarker = " ..
      tostring(g_woodCuttingMarkerEnabled)
1819  end
```

consoleCommandToggleSuperStrongMode

Description

Toggle super-strength mode

Definition

consoleCommandToggleSuperStrongMode()

Code

```
1823  function Player:consoleCommandToggleSuperStrongMode()
1824  if self.superStrengthEnabled then
1825  self.superStrengthEnabled = false
1826  Player.MAX_PICKABLE_OBJECT_MASS = 0.2
1827  return "Player now has normal strength"
1828  else
1829  self.superStrengthEnabled = true
1830  Player.MAX_PICKABLE_OBJECT_MASS = 25
1831  return "Player now has super-strength"
1832  end
1833  end
```

deleteStartleAnimalData

Description

Remove animal sound timer and sound itself (deprecated?)

Definition

deleteStartleAnimalData()

Code

```
1837  function Player:deleteStartleAnimalData()
1838  if (self.startleAnimalSoundTimerId) then
1839  removeTimer(self.startleAnimalSoundTimerId)
1840  self.startleAnimalSoundTimerId = nil
1841  end
1842  self:deleteStartleAnimalSound()
1843  end
```

deleteStartleAnimalSound

Description

Remove animal sound(deprecated?)

Definition

deleteStartleAnimalSound()

Code

```
1847  function Player:deleteStartleAnimalSound()
1848  if (self.startleAnimalSoundNode) then
```

```
1849  delete(self.startleAnimalSoundNode)
1850  self.startleAnimalSoundNode = nil
1851  end
1852  self.startleAnimalSoundTimerId = nil
1853  end
```

consoleCommandReloadIKChains

Description

Reloads IK chains. Used when modifying IK chains in the player configuration file.

Definition

consoleCommandReloadIKChains(table unusedSelf)

Arguments

table unusedSelf unused parameter

Return Values

string that will be displayed on console

Code

```
1859  function Player.consoleCommandReloadIKChains(unusedSelf)
1860  local xmlFile = loadXMLFile("TempXML",
      g_currentMission.player.xmlFilename)
1861
1862  local i = 0
1863  while true do
1864  local key = string.format("player.ikChains.ikChain(%d)", i)
1865  if not hasXMLProperty(xmlFile, key) then
1866  break
1867  end
1868
1869  local id = getXMLString(xmlFile, key.. "#id")
1870  local chain = g_currentMission.player.ikChains[id]
1871  if chain ~= nil then
1872  for k, node in pairs(chain.nodes) do
1873  local nodeKey = key..string.format(".node(%d)", k - 1)
1874  node.minRx = math.rad(Utils.getNoNil(getXMLFloat(xmlFile,
      nodeKey .. "#minRx"), -180))
1875  node.maxRx = math.rad(Utils.getNoNil(getXMLFloat(xmlFile,
      nodeKey .. "#maxRx"), 180))
1876  node.minRy = math.rad(Utils.getNoNil(getXMLFloat(xmlFile,
      nodeKey .. "#minRy"), -180))
1877  node.maxRy = math.rad(Utils.getNoNil(getXMLFloat(xmlFile,
      nodeKey .. "#maxRy"), 180))
1878  node.minRz = math.rad(Utils.getNoNil(getXMLFloat(xmlFile,
      nodeKey .. "#minRz"), -180))
1879  node.maxRz = math.rad(Utils.getNoNil(getXMLFloat(xmlFile,
      nodeKey .. "#maxRz"), 180))
```

```
1880    node.damping = math.rad(Utils.getNoNil(getXMLFloat(xmlFile,
        nodeKey .. "#damping"), 0))
1881
1882    chain.ikChainSolver:setJointTransformGroup(k - 1, node.node,
        node.minRx, node.maxRx, node.minRy, node.maxRy, node.minRz,
        node.maxRz, node.damping)
1883    end
1884    end
1885    i = i + 1
1886    end
1887
1888    g_currentMission.player:setIKDirty()
1889
1890    delete(xmlFile)
1891    end
```

groundRaycastCallback

Description

Callback used to get object below player

Definition

groundRaycastCallback(integer hitObjectId, float x, float y, float z, float distance)

Arguments

integer	hitObjectId	scenegraph object id
float	x	world x hit position
float	y	world y hit position
float	z	world z hit position
float	distance	distance at which the cast hit the object

Return Values

bool returns true object that was hit is valid

Code

```
1901    function Player:groundRaycastCallback(hitObjectId, x, y, z,
        distance)
1902    self.belowPlayerObject = hitObjectId
1903    return false
1904    end
```

pickUpObjectRaycastCallback

Description

Callback used when raycast hists an object. Updates player information so it can be used to pickup the object.

Definition

pickUpObjectRaycastCallback(integer hitObjectId, float x, float y, float z, float distance)

Arguments

integer	hitObjectId	scenegraph object id
float	x	world x hit position

float	y	world y hit position
float	z	world z hit position
float	distance	distance at which the cast hit the object

Return Values

bool returns true object that was hit is valid

Code

```
1915  function Player:pickUpObjectRaycastCallback(hitObjectId, x, y, z, distance)
1916  if hitObjectId ~= g_currentMission.terrainDetailId and Player.PICKED_UP_OBJECTS[hitObjectId] ~= true then
1917  if getRigidBodyType(hitObjectId) == "Dynamic" then
1918
1919  local mass = getMass(hitObjectId)
1920  -- check if mounted:
1921  local canBePickedUp = true
1922  local object = g_currentMission:getNodeObject(hitObjectId)
1923  if object ~= nil then
1924  if object.dynamicMountObject ~= nil then
1925  canBePickedUp = false
1926  end
1927  if object.getTotalMass ~= nil then
1928  mass = object:getTotalMass()
1929  end
1930  if object.getCanBePickedUp ~= nil then
1931  if not object:getCanBePickedUp() then
1932  canBePickedUp = false
1933  end
1934  end
1935  end
1936
1937  if canBePickedUp then
1938  self.lastFoundObject = hitObjectId
1939  self.lastFoundObjectMass = mass
1940  self.lastFoundObjectHitPoint = {x, y, z}
1941  end
1942
1943  -- only consider first potentially valid object
1944  return false
1945  end
1946  end
1947
```

```
1948  return true
1949  end
```

pickUpObject

Description

Picks up an object and links it via a spring mechanism.

Definition

pickUpObject(bool state, bool noEventSend)

Arguments

bool state	if true will join the object, else the joint is removed
bool noEventSend	unused parameter

Code

```
1955  function Player:pickUpObject(state, noEventSend)
1956  PlayerPickUpObjectEvent.sendEvent(self, state, noEventSend)
1957  if self.isServer then
1958  if state and (self.isObjectInRange and self.lastFoundObject ~=
      nil) and not self.isCarryingObject then
1959  local constr = JointConstructor:new()
1960
1961  -- disable collision with CCT
1962  self.pickedUpObjectCollisionMask =
      getCollisionMask(self.lastFoundObject)
1963  local newPickedUpObjectCollisionFlag =
      bitXOR(bitAND(self.pickedUpObjectCollisionMask,
      Player.movementCollisionMask), self.pickedUpObjectCollisionMask)
1964  setCollisionMask(self.lastFoundObject,
      newPickedUpObjectCollisionFlag)
1965
1966  constr:setActors(self.pickUpKinematicHelperNode,
      self.lastFoundObject)
1967
1968  for i=0, 2 do
1969  constr:setRotationLimit(i, 0, 0)
1970  constr:setTranslationLimit(i, true, 0, 0)
1971  end
1972
1973  -- set position of joint
1974  local mx, my, mz = getCenterOfMass(self.lastFoundObject)
1975  local wx, wy, wz = localToWorld(self.lastFoundObject, mx, my,
      mz)
1976  constr:setJointWorldPositions(wx, wy, wz, wx, wy, wz)
1977
1978  local nx, ny, nz = localDirectionToWorld(self.lastFoundObject,
      1, 0, 0)
```

```
1979  constr:setJointWorldAxes(nx, ny, nz, nx, ny, nz)
1980
1981  local yx, yy, yz = localDirectionToWorld(self.lastFoundObject,
      0, 1, 0)
1982  constr:setJointWorldNormals(yx, yy, yz, yx, yy, yz)
1983  constr:setEnableCollision(false)
1984
1985  setWorldTranslation(self.pickUpKinematicHelperNodeChild, wx, wy,
      wz)
1986  setWorldRotation(self.pickUpKinematicHelperNodeChild,
      getWorldRotation(self.lastFoundObject))
1987
1988  -- set spring/damper ?!
1989  local dampingRatio = 1.0
1990  local mass = getMass(self.lastFoundObject)
1991
1992  local rotationLimitSpring = {}
1993  local rotationLimitDamper = {}
1994  for i=1, 3 do
1995  rotationLimitSpring[i] = mass * 60
1996  rotationLimitDamper[i] = dampingRatio * 2 * math.sqrt( mass *
      rotationLimitSpring[i] )
1997  --print(" rotSpring/Damper =
      "..tostring(rotationLimitSpring[i])..." /
      "..tostring(rotationLimitDamper[i]))
1998  end
1999  constr:setRotationLimitSpring(rotationLimitSpring[1],
      rotationLimitDamper[1], rotationLimitSpring[2],
      rotationLimitDamper[2], rotationLimitSpring[3],
      rotationLimitDamper[3])
2000
2001  local translationLimitSpring = {}
2002  local translationLimitDamper = {}
2003  for i=1, 3 do
2004  translationLimitSpring[i] = mass * 60
2005  translationLimitDamper[i] = dampingRatio * 2 * math.sqrt( mass *
      translationLimitSpring[i] )
2006  --print(" transSpring/Damper =
      "..tostring(translationLimitSpring[i])..." /
      "..tostring(translationLimitDamper[i]))
2007  end
2008  constr:setTranslationLimitSpring(translationLimitSpring[1],
      translationLimitDamper[1], translationLimitSpring[2],
```

```
    translationLimitDamper[2], translationLimitSpring[3],
    translationLimitDamper[3])
2009
2010 local forceAcceleration = 4
2011 local forceLimit = forceAcceleration * mass * 40.0
2012 constr:setBreakable(forceLimit, forceLimit)
2013
2014 self.pickedUpObjectJointId = constr:finalize()
2015
2016 addJointBreakReport(self.pickedUpObjectJointId,
    "onPickedUpObjectJointBreak", self)
2017
2018 self.pickedUpObject = self.lastFoundObject
2019 self.isCarryingObject = true
2020 Player.PICKED_UP_OBJECTS[self.pickedUpObject] = true
2021
2022 local object =
    g_currentMission:getNodeObject(self.pickedUpObject)
2023 if object ~= nil then
2024 object.thrownFromPosition = nil
2025 end
2026 else
2027 if self.pickedUpObjectJointId ~= nil then
2028 setCollisionMask(self.pickedUpObject,
    self.pickedUpObjectCollisionMask)
2029 self.pickedUpObjectCollisionMask = 0
2030 removeJoint(self.pickedUpObjectJointId)
2031 self.pickedUpObjectJointId = nil
2032 self.isCarryingObject = false
2033 Player.PICKED_UP_OBJECTS[self.pickedUpObject] = false
2034
2035 if entityExists(self.pickedUpObject) then
2036 local vx, vy, vz = getLinearVelocity(self.pickedUpObject)
2037 vx = MathUtil.clamp(vx, -5, 5)
2038 vy = MathUtil.clamp(vy, -5, 5)
2039 vz = MathUtil.clamp(vz, -5, 5)
2040 setLinearVelocity(self.pickedUpObject, vx, vy, vz)
2041 -- setAngularVelocity(self.pickedUpObject, vx, vy, vz)
2042 end
2043
```

```
2044  local object = g_currentMission:getNodeObject(self.pickedUpObject)
2045  if object ~= nil then
2046  object.thrownFromPosition = nil
2047  end
2048  self.pickedUpObject = nil
2049  end
2050  end
2051  end
2052  end
```

setLightIsActive

Description

Definition

setLightIsActive()

Code

```
2056  function Player:setLightIsActive(isActive, noEventSend)
2057  if isActive ~= self.isLightActive then
2058  self.isLightActive = isActive
2059  PlayerToggleLightEvent.sendEvent(self, isActive, noEventSend)
2060
2061  setVisibility(self.lightNode, isActive)
2062  g_soundManager:playSample(self.soundInformation.samples.flashlight)
2063  end
2064  end
```

loadHandTool

Description

Loading hand tools for the player

Definition

loadHandTool(string xmlFilename)

Arguments

string xmlFilename XML filename

Return Values

table returns the handtool

Code

```
2070  function Player:loadHandTool(xmlFilename)
2071  if GS_IS_CONSOLE_VERSION and not fileExists(xmlFilename) then
2072  return nil
2073  end
2074  local xmlFile = loadXMLFile("TempXML", xmlFilename)
2075  local handToolType = getXMLString(xmlFile, "handTool.handToolType")
2076  delete(xmlFile)
```

```
2077  if handToolType ~= nil then
2078  local classObject = HandTool.handToolTypes[handToolType]
2079  if classObject == nil then
2080  local modName, _ = Utils.getModNameAndBaseDirectory(xmlFilename)
2081  if modName ~= nil then
2082  handToolType = modName.."."..handToolType
2083  classObject = HandTool.handToolTypes[handToolType]
2084  end
2085  end
2086  local handTool = nil
2087  if classObject ~= nil then
2088  handTool = classObject:new(self.isServer, self.isClient)
2089  else
2090  g_logManager:devError("Error: Invalid handtool type
      '"..handToolType.."'")
2091  end
2092  if handTool ~= nil then
2093  if not handTool:load(xmlFilename, self) then
2094  g_logManager:devError("Error: Failed to load handtool
      '"..xmlFilename.."'")
2095  handTool:delete()
2096  handTool = nil
2097  end
2098  end
2099
2100  return handTool
2101  end
2102  return nil
2103  end
```

equipHandtool

Description

Definition

equipHandtool()

Code

```
2107  function Player:equipHandtool(handtoolFilename, force, noEventSend)
2108  PlayerSetHandToolEvent.sendEvent(self, handtoolFilename, force,
      noEventSend)
2109
2110  if self:hasHandtoolEquipped() then
2111  if self.baseInformation.currentHandtool.configFileName ~=
      handtoolFilename or handtoolFilename == "" or force then
2112  self.baseInformation.currentHandtool:onDeactivate()
```

```
2113  self.baseInformation.currentHandtool:delete()
2114  self.baseInformation.currentHandtool = nil
2115  end
2116  if handtoolFilename ~= "" then
2117  self.baseInformation.currentHandtool =
      self:loadHandTool(handtoolFilename)
2118  end
2119  else
2120  if handtoolFilename ~= "" then
2121  self.baseInformation.currentHandtool =
      self:loadHandTool(handtoolFilename)
2122  end
2123  end
2124  -- if we have loaded a handtool, then we configure it
2125  if self:hasHandtoolEquipped() then
2126  self.baseInformation.currentHandtool:onActivate(self:getIsInputAllowed
2127  self.baseInformation.currentHandtool:setHandNode(self.toolsRootNode)
2128
2129  local ikTargets = self.baseInformation.currentHandtool.targets
2130  if ikTargets ~= nil then
2131  for ikChainId, target in pairs(ikTargets) do
2132  IKUtil.setTarget(self.ikChains, ikChainId, target)
2133  end
2134  self:setIKDirty()
2135  end
2136  end
2137  end
```

unequipHandtool

Description

Definition

unequipHandtool()

Code

```
2141  function Player:unequipHandtool()
2142  self:equipHandtool("", true)
2143  end
```

hasHandtoolEquipped

Description

Definition

hasHandtoolEquipped()

Code

```
2147  function Player:hasHandtoolEquipped()
2148  return self.baseInformation.currentHandtool ~= nil
```

```
2149  end
```

throwObject

Description

Throws an object. Activates dog to fetch a ball if conditions are met.

Definition

throwObject()

Code

```
2153  function Player:throwObject(noEventSend)
2154  PlayerThrowObjectEvent.sendEvent(self, noEventSend)
2155  if self.pickedUpObject ~= nil and self.pickedUpObjectJointId ~=
      nil then
2156  local pickedUpObject = self.pickedUpObject
2157  self:pickUpObject(false)
2158
2159  local mass = getMass(pickedUpObject)
2160
2161  local v = 8 * (1.1-mass/Player.MAX_PICKABLE_OBJECT_MASS) --
      speed between 0.8 and 8.8 based on mass of current object
2162  local vx, vy, vz = localDirectionToWorld(self.cameraNode, 0, 0,
      -v)
2163  setLinearVelocity(pickedUpObject, vx, vy, vz)
2164
2165  local object = g_currentMission:getNodeObject(pickedUpObject)
2166  if object ~= nil then
2167  object.thrownFromPosition = {getWorldTranslation(self.rootNode)}
2168  end
2169
2170  local isBall = Utils.getNoNil(getUserAttribute(pickedUpObject,
      "isBall"), false)
2171  local dogHouse = g_currentMission:getDoghouse(self.farmId)
2172  if isBall and dogHouse ~= nil and dogHouse.dog.isActivable then
2173  dogHouse.dog:fetchBall(self.rootNode, pickedUpObject)
2174  end
2175  elseif (self.isObjectInRange and self.lastFoundObject ~= nil)
      and not self.isCarryingObject then
2176  local mass = getMass(self.lastFoundObject)
2177
2178  if mass <= Player.MAX_PICKABLE_OBJECT_MASS then
2179  local v = 8 * (1.1 - mass / Player.MAX_PICKABLE_OBJECT_MASS) --
      speed between 0.8 and 8.8 based on mass of current object
2180  local halfSqrt = 0.707106781
2181  -- Add the impulse in 45deg towards the y axis of the camera
```

```
2182    local vx,vy,vz = localDirectionToWorld(self.cameraNode, 0.0, halfSqrt * v, -halfSqrt * v)
2183    setLinearVelocity(self.lastFoundObject, vx, vy, vz)
2184    end
2185    end
2186    end
```

onPickedUpObjectJointBreak

Description

Callback when picked-up object's joint is broken

Definition

onPickedUpObjectJointBreak(integer jointIndex, float breakingImpulse)

Arguments

integer jointIndex	index of the joint
float breakingImpulse	

Return Values

always returns false

Code

```
2193    function Player:onPickedUpObjectJointBreak(jointIndex, breakingImpulse)
2194    if jointIndex == self.pickedUpObjectJointId then
2195    self:pickUpObject(false)
2196    end
2197    -- Do not delete the joint internally, we already deleted it
2198    return false
2199    end
```

updateSound

Description

Update sound for the player: steps (when crouch, walk, run), swim, plunge

Definition

updateSound()

Code

```
2231    function Player:updateSound()
2232    -- Foot steps
2233    local distanceToCheck = -1.0
2234    local forwardVel = getConditionalAnimationFloatValue(self.animationInformation.player, self.animationInformation.parameters.forwardVelocity.id)
2235
2236    if not self.isEntered then
2237    if self.playerStateMachine:isActive("crouch") then
2238    distanceToCheck = self.soundInformation.distancePerFootstep.crouch
2239    elseif math.abs(forwardVel) <= self.motionInformation.maxWalkingSpeed then
```

```
distanceToCheck = self.soundInformation.distancePerFootstep.walk
elseif math.abs(forwardVel) > self.motionInformation.maxWalkingSpeed then
distanceToCheck = self.soundInformation.distancePerFootstep.run
end
else
if self.playerStateMachine:isActive("crouch") then
distanceToCheck = self.soundInformation.distancePerFootstep.crouch
elseif self.playerStateMachine:isActive("walk") then
distanceToCheck = self.soundInformation.distancePerFootstep.walk
elseif self.playerStateMachine:isActive("run") then
distanceToCheck = self.soundInformation.distancePerFootstep.run
end
end

if distanceToCheck > 0.0 then
local delta = self.motionInformation.coveredGroundDistance - self.soundInformation.distanceSinceLastFootstep
delta = delta - distanceToCheck

if delta > 0 then
if self.baseInformation.isOnGround then
local wx, wy, wz = getWorldTranslation(self.rootNode)
local sample, shallowWater = self:getCurrentSurfaceSound(wx, wy, wz)
if not self.baseInformation.isInWater and self.soundInformation.isSampleSwinPlaying then
g_soundManager:stopSample(self.soundInformation.samples.swim)
self.soundInformation.isSampleSwinPlaying = false
end

if self.baseInformation.isInWater and not shallowWater then
if not self.soundInformation.isSampleSwinPlaying then
g_soundManager:playSample(self.soundInformation.samples.swim)
self.soundInformation.isSampleSwinPlaying = true
end
else
if sample ~= nil then
g_soundManager:playSample(sample)
end
end
```

```
2277  end
2278  self.soundInformation.distanceSinceLastFootstep =
      self.motionInformation.coveredGroundDistance + delta
2279  end
2280  end
2281
2282  -- plunge
2283  if self.baseInformation.plungedInWater then
2284  g_soundManager:playSample(self.soundInformation.samples.plunge)
2285  end
2286  end
```

updateFX

Description

Update particle FX.

Definition

updateFX()

Code

```
2290  function Player:updateFX()
2291  -- swim
2292  -- if self.baseInformation.isInWater then -- and self.motionInformation
2293  -- local x, _, z = getWorldTranslation(self.rootNode)
2294  --
2295  -- setWorldTranslation(self.particleSystemsInformation.swimNode, x,
      g_currentMission.waterY, z)
2296  -- ParticleUtil.setEmittingState(self.particleSystemsInformation.system
2297  -- else
2298  --
      ParticleUtil.resetNumOfEmittedParticles(self.particleSystemsInformation
2299  -- ParticleUtil.setEmittingState(self.particleSystemsInformation.system
2300  -- end
2301
2302  -- plunge
2303  if self.baseInformation.plungedInWater then
2304  local x, _, z = getWorldTranslation(self.rootNode)
2305
2306  setWorldTranslation(self.particleSystemsInformation.plungeNode, x,
      g_currentMission.waterY, z)
2307  ParticleUtil.resetNumOfEmittedParticles(self.particleSystemsInformation
2308  ParticleUtil.setEmittingState(self.particleSystemsInformation.systems.p
2309  end
2310  end
```

addForceToMotion

Description

Accumulate a force that will be applied in the player's motion when integrated

Definition

addForceToMotion(float x, float y, float z)

Arguments

float x force x component in m/s^2

float y force y component in m/s^2

float z force z component in m/s^2

Code

```
2317  function Player:addForceToMotion(x, y, z)
2318  self.motionInformation.accumulatedForces[1] =
      self.motionInformation.accumulatedForces[1] + x
2319  self.motionInformation.accumulatedForces[2] =
      self.motionInformation.accumulatedForces[2] + y
2320  self.motionInformation.accumulatedForces[3] =
      self.motionInformation.accumulatedForces[3] + z
2321  end
```

addVelocityToMotion

Description

Adds a velocity that will be applied in the player's motion when integrated

Definition

addVelocityToMotion(float x, float y, float z)

Arguments

float x velocity x component in m/s

float y velocity y component in m/s

float z velocity z component in m/s

Code

```
2328  function Player:addVelocityToMotion(x, y, z)
2329  self.motionInformation.velocity[1] =
      self.motionInformation.velocity[1] + x
2330  self.motionInformation.velocity[2] =
      self.motionInformation.velocity[2] + y
2331  self.motionInformation.velocity[3] =
      self.motionInformation.velocity[3] + z
2332  end
```

setVelocityToMotion

Description

Overwrites the velocities in the motion information

Definition

setVelocityToMotion(float x, float y, float z)

Arguments

float x velocity x component in m/s

float y velocity y component in m/s

float z velocity z component in m/s

Code

```
2339  function Player:setVelocityToMotion(x, y, z)
2340  if x ~= nil then
2341  self.motionInformation.velocity[1] = x
2342  end
2343  if y ~= nil then
2344  self.motionInformation.velocity[2] = y
2345  end
2346  if z ~= nil then
2347  self.motionInformation.velocity[3] = z
2348  end
2349  end
```

setAccelerationToMotion

Description

Overwrites the ofrces in the motion information

Definition

setAccelerationToMotion(float x, float y, float z)

Arguments

float x velocity x component in m/s

float y velocity y component in m/s

float z velocity z component in m/s

Code

```
2356  function Player:setAccelerationToMotion(x, y, z)
2357  if (x ~= nil) then
2358  self.motionInformation.acceleration[1] = x
2359  end
2360  if (y ~= nil) then
2361  self.motionInformation.acceleration[2] = y
2362  end
2363  if (z ~= nil) then
2364  self.motionInformation.acceleration[3] = z
2365  end
2366  end
```

movePlayer

Description

Definition

movePlayer()

Code

```
2370  function Player:movePlayer(dt)
2371  self.debugFlightCoolDown = self.debugFlightCoolDown - 1
2372  if self.debugFlightMode then
```

```
2373  self.motionInformation.translation[2] =
      self.inputInformation.moveUp * dt
2374  end
2375
2376  self.networkInformation.tickTranslation[1] =
      self.networkInformation.tickTranslation[1] +
      self.motionInformation.translation[1]
2377  self.networkInformation.tickTranslation[2] =
      self.networkInformation.tickTranslation[2] +
      self.motionInformation.translation[2]
2378  self.networkInformation.tickTranslation[3] =
      self.networkInformation.tickTranslation[3] +
      self.motionInformation.translation[3]
2379  moveCCT(self.controllerIndex,
      self.motionInformation.translation[1],
      self.motionInformation.translation[2],
      self.motionInformation.translation[3],
      Player.movementCollisionMask)
2380
2381  self.networkInformation.index = self.networkInformation.index +
      1
2382  if not self.isServer then
2383  -- remove old history (above 100 entries)
2384  while table.getn(self.networkInformation.history) > 100 do
2385  table.remove(self.networkInformation.history, 1)
2386  end
2387  table.insert(self.networkInformation.history,
      {index=self.networkInformation.index,
      movementX=self.motionInformation.translation[1],
      movementY=self.motionInformation.translation[2],
      movementZ=self.motionInformation.translation[3]})
2388  end
2389
2390  self.motionInformation.translation[1] = 0.0
2391  self.motionInformation.translation[2] = 0.0
2392  self.motionInformation.translation[3] = 0.0
2393  end
```

cameraBob

Description

For water

Definition

cameraBob()

Code

```
2397  function Player:cameraBob(dt)
2398  local amplitude = 0.0
```

```
local period = 0.0
local isSwimming = self.playerStateMachine:isActive("swim")
local isWalking = self.playerStateMachine:isActive("walk")
local isCrouching = self.playerStateMachine:isActive("crouch")
local isRunning = self.playerStateMachine:isActive("run")

if isSwimming then
amplitude = 0.009
period = 2000.0
elseif isCrouching and isWalking then
amplitude = 0.03
period = 800.0
elseif isWalking then
amplitude = 0.03
period = 500.0
elseif isRunning then
amplitude = 0.06
period = 400.0
end

if (amplitude ~= 0.0 or period ~= 0.0) and
math.abs(self.motionInformation.desiredSpeed) > 0.0 then
local dtInSec = dt * 0.001
local expectedDistance = self.motionInformation.desiredSpeed *
dtInSec
local deltaDistance =
MathUtil.clamp(self.motionInformation.currentCoveredGroundDistance,
0.0, expectedDistance)
local amplitudeScale = deltaDistance / expectedDistance
local delta = amplitudeScale * amplitude * math.sin(math.pi * 2.0 *
self.time / period)
local cameraY = self.camY
local currentCamX, _, currentCamZ = getTranslation(self.cameraNode)

if not isCrouching then
setTranslation(self.cameraNode, currentCamX, cameraY + delta,
currentCamZ)
else
local crouchState = self.playerStateMachine:getState("crouch")
local crouchCamY = crouchState.crouchCameraY
setTranslation(self.cameraNode, currentCamX, crouchCamY + delta,
currentCamZ)
```

```
2434 end
2435 end
2436 end
```

integrateMotion

Description

Integrates player motion using velocity verlets

Definition

integrateMotion(float dt)

Arguments

float dt delta time in ms

Code

```
2441 function Player:integrateMotion(dt)
2442 -- GDC 2016 / Session Name Math for Game Programmers: Building A Better Jump / Speaker(s) Kyle Pittman
2443 if (dt > 0.0) then
2444 local dtInSec = dt * 0.001
2445
2446 local newAcceleration = {
     self.motionInformation.accumulatedForces[1] *
     self.motionInformation.inverseMass,
2447  self.motionInformation.accumulatedForces[2] *
     self.motionInformation.inverseMass,
2448  self.motionInformation.accumulatedForces[3] *
     self.motionInformation.inverseMass }
2449
2450 self.motionInformation.translation[1] =
     self.motionInformation.translation[1] +
     self.motionInformation.velocity[1] * dtInSec + 0.5 *
     self.motionInformation.acceleration[1] * dtInSec * dtInSec
2451 self.motionInformation.translation[2] =
     self.motionInformation.translation[2] +
     self.motionInformation.velocity[2] * dtInSec + 0.5 *
     self.motionInformation.acceleration[2] * dtInSec * dtInSec
2452 self.motionInformation.translation[3] =
     self.motionInformation.translation[3] +
     self.motionInformation.velocity[3] * dtInSec + 0.5 *
     self.motionInformation.acceleration[3] * dtInSec * dtInSec
2453
2454 self.motionInformation.velocity[1] =
     self.motionInformation.velocity[1] + 0.5 * (
     self.motionInformation.acceleration[1] + newAcceleration[1] ) *
     dtInSec
2455 self.motionInformation.velocity[2] =
     self.motionInformation.velocity[2] + 0.5 * (
     self.motionInformation.acceleration[2] + newAcceleration[2] ) *
     dtInSec
```

```
2456  self.motionInformation.velocity[3] =
      self.motionInformation.velocity[3] + 0.5 * (
      self.motionInformation.acceleration[3] + newAcceleration[3] ) *
      dtInSec
2457
2458  self.motionInformation.acceleration[1] = newAcceleration[1]
2459  self.motionInformation.acceleration[2] = newAcceleration[2]
2460  self.motionInformation.acceleration[3] = newAcceleration[3]
2461
2462  self.motionInformation.accumulatedForces[1] = 0.0
2463  self.motionInformation.accumulatedForces[2] = 0.0
2464  self.motionInformation.accumulatedForces[3] = 0.0
2465
2466  -- Lock Y translation so player is not moving underwater; Y
      accel and vel are reset
2467  local _, playerY, _ = getWorldTranslation(self.rootNode)
2468  local newPlayerY = playerY +
      self.motionInformation.translation[2]
2469  local waterY = g_currentMission.waterY +
      self.baseInformation.waterLevel
2470  local deltaWater = waterY - newPlayerY
2471  if deltaWater >= 0.0 then
2472  self.motionInformation.translation[2] =
      self.motionInformation.translation[2] + deltaWater
2473  self.motionInformation.velocity[2] = 0.0
2474  self.motionInformation.acceleration[2] = 0.0
2475  end
2476  end
2477  end
```

resetInputsInformation

Description

Reset input information inbetween frames. Input is accumulated by events until read, processed and reset.

Definition

resetInputsInformation()

Code

```
2481  function Player:resetInputsInformation()
2482  self.inputInformation.moveRight = 0
2483  self.inputInformation.moveForward = 0
2484  self.inputInformation.moveUp = 0
2485  self.inputInformation.pitchCamera = 0
2486  self.inputInformation.yawCamera = 0
```

```
2487  self.inputInformation.runAxis = 0
2488  self.inputInformation.crouchState = Player.BUTTONSTATES.RELEASED
2489  end
```

debugDraw

Description

Prints player debug information regarding motion and input.

Definition

debugDraw()

Code

```
2493  function Player:debugDraw()
2494  if (self.baseInformation.isInDebug) then
2495  setTextColor(1, 0, 0, 1)
2496  local line = 0.96
2497  renderText(0.05, line, 0.02, "[motion]") line = line - 0.02
2498  renderText(0.05, line, 0.02, string.format("isOnGround(%s) ",
      tostring(self.baseInformation.isOnGround))) line = line - 0.02
2499  renderText(0.05, line, 0.02, string.format("position(%3.4f,
      %3.4f, %3.4f)", self.baseInformation.position[1],
      self.baseInformation.position[2],
      self.baseInformation.position[3])) line = line - 0.02
2500  renderText(0.05, line, 0.02, string.format("lastPosition(%3.4f,
      %3.4f, %3.4f)", self.baseInformation.lastPosition[1],
      self.baseInformation.lastPosition[2],
      self.baseInformation.lastPosition[3])) line = line - 0.02
2501  renderText(0.05, line, 0.02, string.format("direction(%3.4f,
      %3.4f, %3.4f)", self.baseInformation.direction[1],
      self.baseInformation.direction[2],
      self.baseInformation.direction[3])) line = line - 0.02
2502  renderText(0.05, line, 0.02, string.format("mov(%3.4f, %3.4f,
      %3.4f)", self.motionInformation.translation[1],
      self.motionInformation.translation[2],
      self.motionInformation.translation[3])) line = line - 0.02
2503  renderText(0.05, line, 0.02, string.format("vel(%3.4f, %3.4f,
      %3.4f)", self.motionInformation.velocity[1],
      self.motionInformation.velocity[2],
      self.motionInformation.velocity[3])) line = line - 0.02
2504  renderText(0.05, line, 0.02, string.format("acc(%3.4f, %3.4f,
      %3.4f)", self.motionInformation.acceleration[1],
      self.motionInformation.acceleration[2],
      self.motionInformation.acceleration[3])) line = line - 0.02
2505  renderText(0.05, line, 0.02, string.format("accForces(%3.4f,
      %3.4f, %3.4f)", self.motionInformation.accumulatedForces[1],
      self.motionInformation.accumulatedForces[2],
      self.motionInformation.accumulatedForces[3])) line = line - 0.02
2506  renderText(0.05, line, 0.02, string.format("brakeForce(%3.4f,
      %3.4f, %3.4f)", self.motionInformation.brakeForce[1],
```

```
      self.motionInformation.brakeForce[2],
      self.motionInformation.brakeForce[3])) line = line - 0.02
2507  renderText(0.05, line, 0.02,
      string.format("distanceCovered(%.2f)",
      self.motionInformation.coveredGroundDistance)) line = line -
      0.02
2508  renderText(0.05, line, 0.02, string.format("inWater(%s)",
      tostring(self.baseInformation.isInWater))) line = line - 0.02
2509
2510  setTextColor(0, 1, 0, 1)
2511  line = line - 0.02
2512  renderText(0.05, line, 0.02, "[input]")line = line - 0.02
2513  renderText(0.05, line, 0.02, string.format("right(%3.4f)",
      self.inputInformation.moveRight))
2514  line = line - 0.02
2515  renderText(0.05, line, 0.02, string.format("forward(%3.4f)",
      self.inputInformation.moveForward))
2516  line = line - 0.02
2517  renderText(0.05, line, 0.02, string.format("pitch(%3.4f)",
      self.inputInformation.pitchCamera))
2518  line = line - 0.02
2519  renderText(0.05, line, 0.02, string.format("yaw(%3.4f)",
      self.inputInformation.yawCamera))
2520  line = line - 0.02
2521  renderText(0.05, line, 0.02, string.format("runAxis(%3.4f)",
      self.inputInformation.runAxis))
2522  line = line - 0.02
2523  renderText(0.05, line, 0.02, string.format("crouchState(%s)",
      tostring(self.inputInformation.crouchState)))
2524  line = line - 0.02
2525  renderText(0.05, line, 0.02, string.format("interactState(%s)",
      tostring(self.inputInformation.interactState)))
2526  line = line - 0.02
2527  end
2528  end
```

calculateDesiredSpeed

Description

Definition

calculateDesiredSpeed()

Code

```
2532  function Player:calculateDesiredSpeed()
2533  local inputRight = self.inputInformation.moveRight
2534  local inputForward = self.inputInformation.moveForward
2535  local isSwimming = self.playerStateMachine:isActive("swim")
```

```
2536 local isCrouching = self.playerStateMachine:isActive("crouch")
2537 local isFalling = self.playerStateMachine:isActive("fall")
2538 local isUsingHandtool = self:hasHandtoolEquipped()
2539 local baseSpeed = self.motionInformation.maxWalkingSpeed
2540 local runningSpeed = 0
2541 if isFalling then
2542 baseSpeed = self.motionInformation.maxFallingSpeed
2543 elseif isSwimming then
2544 if g_addTestCommands and not g_isPresentationVersion then
2545 if (self.inputInformation.runAxis > 0.0) then
2546 runningSpeed =
     self.motionInformation.maxPresentationRunningSpeed
2547 end
2548 else
2549 baseSpeed = self.motionInformation.maxSwimmingSpeed
2550 end
2551 elseif isCrouching then
2552 baseSpeed = self.motionInformation.maxCrouchingSpeed
2553 end
2554
2555 if ((inputForward ~= 0.0) or (inputRight ~= 0.0)) then
2556 local magnitude = math.sqrt(inputRight * inputRight +
     inputForward * inputForward)
2557 local desiredSpeed = MathUtil.clamp(magnitude, 0.0, 1.0) *
     baseSpeed
2558 local inputRun = self.inputInformation.runAxis
2559
2560 if (inputRun > 0.0) and not (isSwimming or isCrouching or
     isUsingHandtool) then -- check if we are running
2561 runningSpeed = self.motionInformation.maxRunningSpeed
2562
2563 if g_addTestCommands and not g_isPresentationVersion then
2564 runningSpeed =
     self.motionInformation.maxPresentationRunningSpeed
2565 elseif g_addCheatCommands and not g_isPresentationVersion and
     (g_currentMission.isMasterUser or
     g_currentMission:getIsServer()) then
2566 runningSpeed = self.motionInformation.maxCheatRunningSpeed
2567 end
2568 desiredSpeed = inputRun * runningSpeed
2569 end
2570
```

```
2571  local normInputRight = inputRight / magnitude
2572  local normInputForward = inputForward / magnitude
2573
2574  return desiredSpeed, normInputRight, normInputForward
2575  end
2576
2577  return 0.0
2578  end
```

recordPositionInformation

Description

Definition

recordPositionInformation()

Code

```
2582  function Player:recordPositionInformation()
2583  -- gather base information
2584  local prevPos = {}
2585  prevPos[1] = self.baseInformation.position[1]
2586  prevPos[2] = self.baseInformation.position[2]
2587  prevPos[3] = self.baseInformation.position[3]
2588
2589  self.baseInformation.lastPosition[1] = prevPos[1]
2590  self.baseInformation.lastPosition[2] = prevPos[2]
2591  self.baseInformation.lastPosition[3] = prevPos[3]
2592
2593  self.baseInformation.position[1],
2594  self.baseInformation.position[2],
2595  self.baseInformation.position[3] = getTranslation(self.rootNode)
2596
2597  local deltaPos = {self.baseInformation.position[1] -
      self.baseInformation.lastPosition[1],
2598  self.baseInformation.position[2] -
      self.baseInformation.lastPosition[2],
2599  self.baseInformation.position[3] -
      self.baseInformation.lastPosition[3]}
2600  local groundDistanceCovered =
      MathUtil.vector2Length(deltaPos[1], deltaPos[3])
2601  self.motionInformation.justMoved = groundDistanceCovered > 0.0
2602  if self.baseInformation.isOnGround then
2603  self.motionInformation.currentCoveredGroundDistance =
      groundDistanceCovered
2604  self.motionInformation.coveredGroundDistance =
      self.motionInformation.coveredGroundDistance +
      groundDistanceCovered
```

```
2605 end
2606 end
```

calculate2DDotProductAgainstVelocity

Description

Definition

calculate2DDotProductAgainstVelocity()

Code

```
2610 function Player:calculate2DDotProductAgainstVelocity(velocity,
     currentSpeed, vector)
2611 local normalizedVelX = velocity[1] / currentSpeed
2612 local normalizedVelZ = velocity[3] / currentSpeed
2613 local vectorMagnitude = math.sqrt(vector[1] * vector[1] +
     vector[3] * vector[3])
2614 local normalizedVectorX = vector[1] / vectorMagnitude
2615 local normalizedVectorZ = vector[3] / vectorMagnitude
2616 local dot = normalizedVelX * normalizedVectorX + normalizedVelZ
     * normalizedVectorZ
2617
2618 return dot
2619 end
```

resetBrake

Description

Definition

resetBrake()

Code

```
2623 function Player:resetBrake()
2624 self:setVelocityToMotion(0.0, 0.0, 0.0)
2625 self:setAccelerationToMotion(0.0, 0.0, 0.0)
2626 self.motionInformation.brakeForce = {0.0, 0.0, 0.0}
2627 self.motionInformation.isBraking = false
2628 end
```

updateKinematic

Description

Updates player movement depending on inputs (run, crouch, swim) and apply gravity

Definition

updateKinematic()

Code

```
2632 function Player:updateKinematic()
2633 local EPSILON = 0.00001
2634 local desiredSpeed, normInputRight, normInputForward =
     self:calculateDesiredSpeed()
2635 local velocity = self.motionInformation.velocity
```

```lua
local currentSpeed = math.sqrt(velocity[1] * velocity[1] + velocity[3] * velocity[3])

self.motionInformation.lastSpeed = self.motionInformation.currentSpeed
self.motionInformation.currentSpeed = currentSpeed

local mass = self.motionInformation.mass
local inAir = not (self.debugFlightMode or self.baseInformation.isOnGround or self.baseInformation.isInWater)
if inAir then
self:addForceToMotion(0.0, self.motionInformation.gravity[2] * mass, 0.0)
end
self.motionInformation.desiredSpeed = desiredSpeed

if desiredSpeed ~= 0.0 and self.motionInformation.isBraking then
self:resetBrake()
elseif desiredSpeed ~= 0.0 then
local currentWorldDirX, _, currentWorldDirZ = localDirectionToWorld(self.cameraNode, normInputRight, 0.0, normInputForward)
currentWorldDirX, currentWorldDirZ = MathUtil.vector2Normalize(currentWorldDirX, currentWorldDirZ)
local desiredVelocityX = currentWorldDirX * desiredSpeed
local desiredVelocityZ = currentWorldDirZ * desiredSpeed
local deltaVelX = (desiredVelocityX - velocity[1])
local accelerationX = deltaVelX / self.motionInformation.maxSpeedDelay
local forceMagnitudeX = accelerationX * mass
local deltaVelZ = (desiredVelocityZ - velocity[3])
local accelerationZ = deltaVelZ / self.motionInformation.maxSpeedDelay
local forceMagnitudeZ = accelerationZ * mass
self:addForceToMotion(forceMagnitudeX, 0.0, forceMagnitudeZ)
self.motionInformation.isReverse = false
if currentSpeed > 0.0 then
local worldDir = {currentWorldDirX, 0.0, currentWorldDirZ}
local dot = self:calculate2DDotProductAgainstVelocity(velocity, currentSpeed, worldDir)

if dot <= 0.0 then
```

```
2668    self.motionInformation.isReverse = true
2669    end
2670    end
2671    elseif (math.abs(velocity[1]) > EPSILON or math.abs(velocity[3])
        > EPSILON) then
2672    if not self.motionInformation.isBraking then
2673    local desiredVelocityX = 0.0
2674    local desiredVelocityZ = 0.0
2675    local accelerationX = (desiredVelocityX - velocity[1]) /
        self.motionInformation.brakeDelay
2676    local forceMagnitudeX = accelerationX * mass
2677    local accelerationZ = (desiredVelocityZ - velocity[3]) /
        self.motionInformation.brakeDelay
2678    local forceMagnitudeZ = accelerationZ * mass
2679
2680    self.motionInformation.brakeForce = {forceMagnitudeX, 0.0,
        forceMagnitudeZ}
2681    self:addForceToMotion(self.motionInformation.brakeForce[1],
        self.motionInformation.brakeForce[2],
        self.motionInformation.brakeForce[3])
2682    self.motionInformation.isBraking = true
2683    else
2684    local dotVelBrake =
        self:calculate2DDotProductAgainstVelocity(velocity,
        currentSpeed, self.motionInformation.brakeForce)
2685    if dotVelBrake >= 0.0 then
2686    self:resetBrake()
2687    else
2688    self:addForceToMotion(self.motionInformation.brakeForce[1],
        self.motionInformation.brakeForce[2],
        self.motionInformation.brakeForce[3])
2689    end
2690    end
2691    end
2692    end
```

updatePlayerStates

Description

Updates player state machine

Definition

updatePlayerStates()

Code

```
2696    function Player:updatePlayerStates()
2697    if self.playerStateMachine:isAvailable("fall") then
```

```
2698  self.playerStateMachine:activateState("fall")
2699  end
2700
2701  if self.baseInformation.isInWater and
      self.playerStateMachine:isAvailable("swim") then
2702  self.playerStateMachine:activateState("swim")
2703  self.playerStateMachine:deactivateState("crouch")
2704  end
2705
2706  if (self.inputInformation.moveForward ~= 0) or
      (self.inputInformation.moveRight ~= 0) then
2707  if (self.inputInformation.runAxis > 0.0) and
      self.playerStateMachine:isAvailable("run") then
2708  self.playerStateMachine:activateState("run")
2709  elseif self.playerStateMachine:isAvailable("walk") then
2710  self.playerStateMachine:activateState("walk")
2711  end
2712  else
2713  self.playerStateMachine:activateState("idle")
2714  end
2715  end
```

setWalkingLock

Description

Lock or unlock the player's movement. Sets the Player.walkingIsLocked flag and enables/disables movement action events accordingly.

Definition

setWalkingLock(isLocked If)

Arguments

isLocked If true, the player's movement is locked. Otherwise, it is released.

Code

```
2720  function Player:setWalkingLock(isLocked)
2721  self.walkingIsLocked = isLocked
2722
2723  for _, inputRegistration in
      pairs(self.inputInformation.registrationList) do
2724  if inputRegistration.activeType ==
      Player.INPUT_ACTIVE_TYPE.IS_MOVEMENT then
2725  g_inputBinding:setActionEventActive(inputRegistration.eventId,
      not isLocked)
2726  end
2727  end
2728  end
```

consoleCommandTogglePlayerDebug

Description

Toggle player debug info display

Definition

consoleCommandTogglePlayerDebug()

Return Values

string that will be displayed on console

Code

```
2733  function Player:consoleCommandTogglePlayerDebug()
2734  self.baseInformation.isInDebug = not
      self.baseInformation.isInDebug
2735  return "Player Debug = " ..
      tostring(self.baseInformation.isInDebug)
2736  end
```

registerActionEvents

Description

Register required player action events.

Definition

registerActionEvents()

Code

```
2774  function Player:registerActionEvents()
2775  -- register action events for the player context without switching
      (important when this is called from within the UI context)
2776  g_inputBinding:beginActionEventsModification(Player.INPUT_CONTEXT_NAME)
2777
2778  for actionId, inputRegisterEntry in
      pairs(self.inputInformation.registrationList) do
2779  local eventAdded = false
2780  local startActive = false
2781
2782  if inputRegisterEntry.activeType ==
      Player.INPUT_ACTIVE_TYPE.STARTS_ENABLED then
2783  startActive = true
2784  elseif inputRegisterEntry.activeType ==
      Player.INPUT_ACTIVE_TYPE.STARTS_DISABLED then
2785  startActive = false
2786  elseif inputRegisterEntry.activeType ==
      Player.INPUT_ACTIVE_TYPE.IS_MOVEMENT then
2787  startActive = not self.walkingIsLocked
2788  elseif inputRegisterEntry.activeType ==
      Player.INPUT_ACTIVE_TYPE.IS_CARRYING then
2789  startActive = self.isCarryingObject
2790  elseif inputRegisterEntry.activeType ==
      Player.INPUT_ACTIVE_TYPE.IS_DEBUG then
```

```
2791  startActive = self.baseInformation.isInDebug
2792  end
2793
2794  eventAdded, inputRegisterEntry.eventId =
      g_inputBinding:registerActionEvent(actionId, self,
      inputRegisterEntry.callback, inputRegisterEntry.triggerUp,
      inputRegisterEntry.triggerDown, inputRegisterEntry.triggerAlways,
      startActive, inputRegisterEntry.callbackState)
2795  if inputRegisterEntry.text ~= nil and inputRegisterEntry.text ~= ""
      then
2796  g_inputBinding:setActionEventText(inputRegisterEntry.eventId,
      inputRegisterEntry.text)
2797  end
2798
2799  g_inputBinding:setActionEventTextVisibility(inputRegisterEntry.eventId,
      inputRegisterEntry.textVisibility)
2800
2801  -- if eventAdded == false then
2802  -- g_logManager:devWarning("Cannot add player action event: %s.",
      tostring(actionId))
2803  -- end
2804  end
2805  -- reset registration context, update event data in input system:
2806  g_inputBinding:endActionEventsModification()
2807  end
```

removeActionEvents

Description

Remove all player action events.

Definition

removeActionEvents()

Code

```
2811  function Player:removeActionEvents()
2812  -- modify action events in player context without switching (important
      because this can be called from within the UI)
2813  g_inputBinding:beginActionEventsModification(Player.INPUT_CONTEXT_NAME)
2814  g_inputBinding:removeActionEventsByTarget(self)
2815  for _, inputRegisterEntry in
      pairs(self.inputInformation.registrationList) do
2816  inputRegisterEntry.eventId = ""
2817  end
2818  g_inputBinding:endActionEventsModification()
2819  end
```

onInputLookLeftRight

Description

Event function for player camera horizontal axis.

Definition

onInputLookLeftRight(nil , float inputValue)

Arguments

nil

float inputValue

Code

```
2825  function Player:onInputLookLeftRight(_, inputValue, _, _,
      isMouse)
2826  if not self.lockedInput then
2827  if isMouse then
2828  inputValue = inputValue * 0.001 * 16.666
2829  else
2830  inputValue = inputValue * g_currentDt *0.001
2831  end
2832  self.inputInformation.yawCamera =
      self.inputInformation.yawCamera + inputValue
2833  end
2834  self.inputInformation.isMouseRotation = isMouse
2835  end
```

onInputLookUpDown

Description

Event function for player camera vertical axis.

Definition

onInputLookUpDown(nil , float inputValue)

Arguments

nil

float inputValue

Code

```
2841  function Player:onInputLookUpDown(_, inputValue, _, _, isMouse)
2842  if not self.lockedInput then
2843  local pitchValue = g_gameSettings:getValue("invertYLook") and -
      inputValue or inputValue
2844  if isMouse then
2845  pitchValue = pitchValue * 0.001 * 16.666
2846  else
2847  pitchValue = pitchValue * g_currentDt *0.001
2848  end
2849  self.inputInformation.pitchCamera =
      self.inputInformation.pitchCamera + pitchValue
2850  end
```

```
2851  end
```

onInputMoveSide

Description

Event function for player strafe movement.

Definition

onInputMoveSide(nil , float inputValue)

Arguments

nil

float inputValue

Code

```
2857  function Player:onInputMoveSide(_, inputValue)
2858  if not self.lockedInput then
2859  self.inputInformation.moveRight =
      self.inputInformation.moveRight + inputValue
2860  end
2861  end
```

onInputMoveForward

Description

Event function for player forward/backward movement.

Definition

onInputMoveForward(nil , float inputValue)

Arguments

nil

float inputValue

Code

```
2867  function Player:onInputMoveForward(_, inputValue)
2868  if not self.lockedInput then
2869  self.inputInformation.moveForward =
      self.inputInformation.moveForward + inputValue
2870  end
2871  end
```

onInputRun

Description

Event function for player running.

Definition

onInputRun(nil , float inputValue)

Arguments

nil

float inputValue

Code

```
2877  function Player:onInputRun(_, inputValue)
2878  self.inputInformation.runAxis = inputValue
```

```
2879
2880  if self.debugFlightMode then
2881  if inputValue > 0 and self.debugFlightModeRunningFactor ~= 4.0
      then
2882  self.debugFlightModeRunningFactor = 4.0
2883  elseif inputValue == 0 and self.debugFlightModeRunningFactor ~=
      1.0 then
2884  self.debugFlightModeRunningFactor = 1.0
2885  end
2886  end
2887  end
```

onInputCrouch

Description

Event function for crouching.

Definition

onInputCrouch()

Code

```
2891  function Player:onInputCrouch(_, inputValue)
2892  if self.playerStateMachine:isAvailable("crouch") then
2893  self.playerStateMachine:activateState("crouch")
2894  end
2895
2896  self.inputInformation.crouchState = Player.BUTTONSTATES.PRESSED
2897  end
```

onInputRotateObjectHorizontally

Description

Event function for rotating object.

Definition

onInputRotateObjectHorizontally()

Code

```
2901  function Player:onInputRotateObjectHorizontally(_, inputValue)
2902  if self.pickedUpObjectJointId ~= nil and math.abs(inputValue) >
      0 then
2903  self:rotateObject(inputValue, 0.0, 1.0, 0.0)
2904  elseif self.isCarryingObject and self.isClient and
      self.isControlled then
2905  if inputValue ~= 0.0 then
2906  self.networkInformation.rotateObject = true
2907  else
2908  self.networkInformation.rotateObject = false
2909  end
2910  self.networkInformation.rotateObjectInputH = inputValue
```

```
2911  end
2912  end
```

onInputRotateObjectVertically

Description

Event function for rotating object.

Definition

onInputRotateObjectVertically()

Code

```
2916  function Player:onInputRotateObjectVertically(_, inputValue)
2917  if self.pickedUpObjectJointId ~= nil and math.abs(inputValue) > 0 then
2918  self:rotateObject(inputValue, 1.0, 0.0, 0.0)
2919  elseif self.isCarryingObject and self.isClient and self.isControlled then
2920  if inputValue ~= 0.0 then
2921  self.networkInformation.rotateObject = true
2922  else
2923  self.networkInformation.rotateObject = false
2924  end
2925  self.networkInformation.rotateObjectInputV = inputValue
2926  end
2927  end
```

rotateObject

Description

Rotates object

Definition

rotateObject()

Code

```
2931  function Player:rotateObject(inputValue, axisX, axisY, axisZ)
2932  local jointIndex = self.pickedUpObjectJointId
2933  local actor = 0
2934  local objectTransform = self.pickUpKinematicHelperNodeChild
2935  local rotX, rotY, rotZ = localDirectionToLocal(self.cameraNode, objectTransform, axisX, axisY, axisZ)
2936  local dtInSec = g_physicsDt * 0.001
2937  local rotation = math.rad(90.0) * dtInSec * inputValue
2938
2939  rotateAboutLocalAxis(objectTransform, rotation, rotX, rotY, rotZ)
2940  setJointFrame(jointIndex, actor, objectTransform)
2941  end
```

onInputJump

Description

Event function for jumping.

Definition

onInputJump()

Code

```
2945  function Player:onInputJump(_, inputValue)
2946  if self.playerStateMachine:isAvailable("jump") then
2947  self.playerStateMachine:activateState("jump")
2948  end
2949  end
```

onInputInteract

Description

Event function for interacting with an animal

Definition

onInputInteract()

Code

```
2953  function Player:onInputInteract(_, inputValue)
2954  -- Note: we need to store pressed state for animal cleaning (see
      PlayerStateAnimalInteract) which is a continuous
2955  -- action. Therefore, this event is called onUp and onDown. When
      the down event is received, input will be non-zero.
2956  if self.inputInformation.interactState ~=
      Player.BUTTONSTATES.PRESSED and inputValue ~= 0 then
2957  if self.playerStateMachine:isAvailable("drop") then
2958  self.playerStateMachine:activateState("drop")
2959  elseif self.playerStateMachine:isAvailable("pickup") then
2960  self.playerStateMachine:activateState("pickup")
2961  elseif self.playerStateMachine:isAvailable("animalInteract")
      then
2962  self.playerStateMachine:activateState("animalInteract")
2963  end
2964
2965  self.inputInformation.interactState =
      Player.BUTTONSTATES.PRESSED
2966  else
2967  self.inputInformation.interactState =
      Player.BUTTONSTATES.RELEASED
2968  end
2969  end
```

onInputActivateObject

Description

Event function for interacting with an animal

Definition

onInputActivateObject()

Code

```
function Player:onInputActivateObject(_, inputValue)
if self.playerStateMachine:isAvailable("activateObject") then
self.playerStateMachine:activateState("activateObject")
elseif self.playerStateMachine:isAvailable("animalFeed") then
self.playerStateMachine:activateState("animalFeed")
elseif self.playerStateMachine:isAvailable("animalPet") then
self.playerStateMachine:activateState("animalPet")
end
end
```

onInputToggleLight

Description

Event function for flashlight toggle.

Definition

onInputToggleLight()

Code

```
function Player:onInputToggleLight()
if self.playerStateMachine:isAvailable("useLight") then
self.playerStateMachine:activateState("useLight")
end
end
```

onInputCycleHandTool

Description

Event function for cycling through available hand tools.

Definition

onInputCycleHandTool(nil , nil , integer direction)

Arguments

nil

nil

integer direction direction in which the equipment is cycled through

Code

```
function Player:onInputCycleHandTool(_, _, direction)
if self.playerStateMachine:isAvailable("cycleHandtool") then
local cycleHandtoolState = self.playerStateMachine:getState("cycleHandtool")
cycleHandtoolState.cycleDirection = direction
self.playerStateMachine:activateState("cycleHandtool")
end
end
```

onInputThrowObject

Description

Event function for throwing an object.

Definition

onInputThrowObject()

Code

```
3006  function Player:onInputThrowObject(_, inputValue)
3007  if self.playerStateMachine:isAvailable("throw") then
3008  self.playerStateMachine:activateState("throw")
3009  end
3010  end
```

onInputDebugFlyToggle

Description

Event function for the debug flying toggle.

Definition

onInputDebugFlyToggle()

Code

```
3014  function Player:onInputDebugFlyToggle()
3015  if not self.walkingIsDisabled then
3016  if self.debugFlightCoolDown <= 0 then
3017  if g_flightAndNoHUDKeysEnabled then
3018  self.debugFlightMode = not self.debugFlightMode
3019  self.debugFlightCoolDown = 10
3020  end
3021  end
3022  end
3023  end
```

onInputDebugFlyUpDown

Description

Event function for the debug flying vertical movement.

Definition

onInputDebugFlyUpDown(nil , float inputValue)

Arguments

nil

float inputValue

Code

```
3029  function Player:onInputDebugFlyUpDown(_, inputValue)
3030  if not self.walkingIsDisabled then
3031  local move = inputValue * 0.5 * self.debugFlightModeWalkingSpeed
      * self.debugFlightModeRunningFactor
3032  self.inputInformation.moveUp = self.inputInformation.moveUp +
      move
3033  end
```

```
3034  end
```

onInputEnter

Description

Event function for enter

Definition

onInputEnter(nil , float inputValue)

Arguments

nil

float inputValue

Code

```
3040  function Player:onInputEnter(_, inputValue)
3041  if g_time > g_currentMission.lastInteractionTime + 200 then
3042  local enterVehicle = false
3043  if g_currentMission.interactiveVehicleInRange and
      g_currentMission.accessHandler:canFarmAccess(self.farmId,
      g_currentMission.interactiveVehicleInRange) then
3044  g_currentMission.interactiveVehicleInRange:interact()
3045  enterVehicle = true
3046  elseif self.canRideAnimal then
3047  self.playerStateMachine:activateState("animalRide")
3048  enterVehicle = true
3049  end
3050  end
3051  end
```

onInputActivateHandtool

Description

Definition

onInputActivateHandtool()

Code

```
3055  function Player:onInputActivateHandtool(_, inputValue)
3056  if self:hasHandtoolEquipped() then
3057  self.baseInformation.currentHandtool.activatePressed =
      inputValue ~= 0
3058  end
3059  end
```

PlayerModelManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
27  function PlayerModelManager:new(customMt)
28  local self = {}
29  setmetatable(self, customMt or PlayerModelManager_mt)
30
31  self:initDataStructures()
32
33  return self
34  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
38  function PlayerModelManager:initDataStructures()
39  self.playerModels = {}
40  self.nameToPlayerModel = {}
41  self.nameToIndex = {}
42  end
```

load

Description

Loads initial manager

Definition

load()

Return Values

boolean true if loading was successful else false

Code

```
47  function PlayerModelManager:load(xmlFilename)
48  local result = false
49  if xmlFilename ~= nil and xmlFilename ~= "" then
50  local xmlFile = loadXMLFile("TempXML", xmlFilename)
51  if xmlFile ~= nil and xmlFile ~= 0 then
52  local i = 0
53
54  while true do
55  local baseKey = "playerModels.playerModel"
56  local XMLFilenameKey = string.format("%s(%d)#xmlFilename",
    baseKey, i)
57  local genderKey = string.format("%s(%d)#gender", baseKey, i)
```

```
58  local defaultNameKey = string.format("%s(%d)#defaultName",
    baseKey, i)
59  local descKey = string.format("%s(%d)#desc", baseKey, i)
60
61  if not hasXMLProperty(xmlFile, XMLFilenameKey) or not
    hasXMLProperty(xmlFile, genderKey) or not hasXMLProperty(xmlFile,
    defaultNameKey) or not hasXMLProperty(xmlFile, descKey) then
62  break
63  end
64  local modelXMLFilename = getXMLString(xmlFile, XMLFilenameKey)
65  local modelGender = getXMLString(xmlFile, genderKey)
66  local modelDefaultName = getXMLString(xmlFile, defaultNameKey)
67  local modelDesc = getXMLString(xmlFile, descKey)
68
69  local modelGenderId = PlayerModelManager.GENDERS.GENDER_NONE
70  modelGender = modelGender:upper()
71  if modelGender == "FEMALE" then
72  modelGenderId = PlayerModelManager.GENDERS.GENDER_FEMALE
73  else
74  modelGenderId = PlayerModelManager.GENDERS.GENDER_MALE
75  end
76  if self:addPlayerModel(modelDefaultName, modelXMLFilename,
    modelDesc, modelGenderId) ~= nil then
77  result = true
78  end
79  i = i + 1
80  end
81  delete(xmlFile)
82  else
83  print(string.format("Warning: Cannot open xmlFilename('%s') is
    missing for player model manager.", tostring(xmlFilename)))
84  end
85  else
86  print(string.format("Warning: Config xmlFilename('%s') is missing
    for player model manager.", tostring(xmlFilename)))
87  end
88  return result
89  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
94  function PlayerModelManager:loadMapData(xmlFile)
95  return true
96  end
```

unloadMapData

Description

Unload data on mission delete

Definition

unloadMapData()

Code

```
100 function PlayerModelManager:unloadMapData()
101 end
```

addPlayerModel

Description

Adds a new player

Definition

addPlayerModel(string name, string xmlFilename, string description, string baseDir)

Arguments

string name	index name
string xmlFilename	xml filename
string description	description
string baseDir	the base directory

Return Values

boolean true if added successful else false

Code

```
110 function PlayerModelManager:addPlayerModel(name, xmlFilename,
    description, genderId)
111 if not ClassUtil.getIsValidIndexName(name) then
112 g_logManager:devWarning("Warning: '"..tostring(name).."' is not a
    valid name for a player. Ignoring it!")
113 return nil
114 end
115 if xmlFilename == nil or xmlFilename == "" then
116 g_logManager:devWarning("Warning: Config xmlFilename is missing
    for player '"..tostring(name).."'. Ignoring it!")
117 return nil
118 end
119
120 name = name:upper()
121
```

```
122  if self.nameToPlayerModel[name] == nil then
123  local numPlayerModels = #self.playerModels + 1
124  local playerModel = {}
125
126  playerModel.name = name
127  playerModel.index = numPlayerModels
128  playerModel.xmlFilename = Utils.getFilename(xmlFilename, nil)
129  playerModel.description = description
130  playerModel.genderId = genderId
131  table.insert(self.playerModels, playerModel)
132  self.nameToPlayerModel[name] = playerModel
133  self.nameToIndex[name] = numPlayerModels
134  return playerModel
135  else
136  g_logManager:devWarning("Warning: Player '"..tostring(name).."'
     already exists. Ignoring it!")
137  end
138
139  return nil
140  end
```

getPlayerModelByIndex

Description

Gets a player by index

Definition

getPlayerModelByIndex(integer index)

Arguments

integer index the player index

Return Values

table player the player object

Code

```
146  function PlayerModelManager:getPlayerModelByIndex(index)
147  if index ~= nil then
148  return self.playerModels[index]
149  end
150  return nil
151  end
```

getPlayerByName

Description

Gets a player by index name

Definition

getPlayerByName(string name)

Arguments

string name the player index name

Return Values

table player the player object

Code

```
157  function PlayerModelManager:getPlayerByName(name)
158  if name ~= nil then
159  name = name:upper()
160  return self.nameToPlayerModel[name]
161  end
162  return nil
163  end
```

getNumOfPlayerModels

Description

Gets number of available player models

Definition

getNumOfPlayerModels()

Return Values

integer number number of models

Code

```
168  function PlayerModelManager:getNumOfPlayerModels()
169  return #self.playerModels
170  end
```

PlayerPickUpObjectEvent

Description

emptyNew

Description

Create an empty instance

Definition

emptyNew()

Return Values

table instance Instance of object

Code

```
11  function PlayerPickUpObjectEvent:emptyNew()
12  local self = Event:new(PlayerPickUpObjectEvent_mt)
13  return self
14  end
```

new

Description

Create an instance

Definition

new(table player, bool state)

Arguments

table player player instance

bool state

Return Values

table instance Instance of object

Code

```
21  function PlayerPickUpObjectEvent:new(player, state)
22  self.player = player
23  self.state = state
24  return self
25  end
```

readStream

Description

Reads network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
31  function PlayerPickUpObjectEvent:readStream(streamId, connection)
32  self.player = NetworkUtil.readNodeObject(streamId)
33  self.state = streamReadBool(streamId)
34  self:run(connection)
35  end
```

writeStream

Description

Writes network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
41  function PlayerPickUpObjectEvent:writeStream(streamId, connection)
42  NetworkUtil.writeNodeObject(streamId, self.player)
43  streamWriteBool(streamId, self.state)
44  end
```

run

Description

Run event

Definition

run(table connection)

Arguments

table connection connection information

Code

```
49  function PlayerPickUpObjectEvent:run(connection)
50  if not connection:getIsServer() then
51  g_server:broadcastEvent(self, false, connection, self.player)
52  end
53
54  self.player:pickUpObject(self.state, true)
55  end
```

sendEvent

Description

Definition

sendEvent()

Code

```
59  function PlayerPickUpObjectEvent.sendEvent(player, state, noEventSend)
60  if noEventSend == nil or noEventSend == false then
61  if g_server ~= nil then
62  g_server:broadcastEvent(PlayerPickUpObjectEvent:new(player, state), nil,
    nil, player)
63  else
64  g_client:getServerConnection():sendEvent(PlayerPickUpObjectEvent:new(play
    state))
65  end
66  end
67  end
```

PlayerSetFarmEvent

Description

emptyNew

Description

Create an empty instance

Definition

emptyNew()

Return Values

table instance Instance of object

Code

```
13  function PlayerSetFarmEvent:emptyNew()
14  local self = Event:new(PlayerSetFarmEvent_mt)
15  return self
16  end
```

new

Description

Create an instance

Definition

new(table player, integer toolId)

Arguments

table player player instance

integer toolId tool identification

Return Values

table instance Instance of object

Code

```
23  function PlayerSetFarmEvent:new(player, farmId, password)
24  local self = PlayerSetFarmEvent:emptyNew()
25
26  self.player = player
27  self.farmId = farmId
28  self.password = password
29
30  return self
31  end
```

writeStream

Description

Writes network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
37  function PlayerSetFarmEvent:writeStream(streamId, connection)
38  NetworkUtil.writeNodeObject(streamId, self.player)
39  streamWriteUIntN(streamId, self.farmId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
40
41  if self.password ~= nil then
42  streamWriteBool(streamId, true)
43  streamWriteString(streamId, self.password)
44  else
45  streamWriteBool(streamId, false)
46  end
47  end
```

readStream

Description

Reads network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId network stream identification
table connection connection information

Code

```
53  function PlayerSetFarmEvent:readStream(streamId, connection)
54  self.player = NetworkUtil.readNodeObject(streamId)
55  self.farmId = streamReadUIntN(streamId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
56
57  if streamReadBool(streamId) then
58  self.password = streamReadString(streamId)
59  end
60
61  self:run(connection)
62  end
```

run

Description

Run event

Definition

run(table connection)

Arguments

table connection connection information

Code

```
67  function PlayerSetFarmEvent:run(connection)
68  if not connection:getIsServer() then --server side
69  local oldFarmId = self.player.farmId
70  local oldFarm = g_farmManager:getFarmById(oldFarmId)
71
72  local farm = g_farmManager:getFarmById(self.farmId)
73
74  if farm ~= nil then
75  local user = g_currentMission:getUserByUserId(self.player.userId)
76
77  -- admins can always join any farm, otherwise the password must be empty
78  if user.isMasterUser or farm.password == nil or farm.password == self.pa
79  oldFarm:removeUser(user.userId)
80  self.player:setFarm(self.farmId)
81  farm:addUser(user.userId, user.uniqueUserId, user.isMasterUser)
82
83  self.player:onFarmChange()
```

```
84
85  -- publish message about farm change
86  g_messageCenter:publish(MessageType.PLAYER_FARM_CHANGED, {self.player})
87
88  -- Force an update of the finance history
89  user.financesVersionCounter = 0
90
91  -- Finish handshake, lets the client record the password for the farm
92  connection:sendEvent(PlayerSetFarmAnswerEvent:new(PlayerSetFarmAnswerEve
    self.password))
93
94  -- Tell all players that a player has switched
95  g_server:broadcastEvent(PlayerSwitchedFarmEvent:new(oldFarmId, self.farm
96  else
97  -- let the client know that the correct password is required
98  connection:sendEvent(PlayerSetFarmAnswerEvent:new(PlayerSetFarmAnswerEve
    self.farmId))
99  end
100 end
101 else -- client side
102 self.player.farmId = self.farmId
103 self.player:onFarmChange()
104
105 g_messageCenter:publish(MessageType.PLAYER_FARM_CHANGED, {self.player})
106 end
107 end
```

sendEvent

Description

Create an instance

Definition

sendEvent(table player, integer farmId, bool noEventSend)

Arguments

table	player	player instance
integer	farmId	farm identification
bool	noEventSend	if false will send the event

Code

```
114 function PlayerSetFarmEvent.sendEvent(player, farmId, noEventSend)
115 if noEventSend == nil or noEventSend == false then
116 if g_server ~= nil then
117 g_server:broadcastEvent(PlayerSetFarmEvent:new(player, farmId), nil,
    nil, player)
118 else
```

```
119  g_client:getServerConnection():sendEvent(PlayerSetFarmEvent:new(player,
     farmId))
120  end
121  end
122  end
```

PlayerSetHandToolEvent

Description

emptyNew

Description

Create an empty instance

Definition

emptyNew()

Return Values

table instance Instance of object

Code

```
11  function PlayerSetHandToolEvent:emptyNew()
12  local self = Event:new(PlayerSetHandToolEvent_mt)
13  return self
14  end
```

new

Description

Create an instance

Definition

new(table player, string tool)

Arguments

table player player instance

string tool identification

Return Values

table instance Instance of object

Code

```
21  function PlayerSetHandToolEvent:new(player, handtoolFileName,
    force)
22  local self = PlayerSetHandToolEvent:emptyNew()
23  self.player = player
24  self.handtoolFileName = handtoolFileName
25  self.force = force
26  return self
27  end
```

readStream

Description

Reads network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
33  function PlayerSetHandToolEvent:readStream(streamId, connection)
34  self.player = NetworkUtil.readNodeObject(streamId)
35  self.handtoolFileName =
    NetworkUtil.convertFromNetworkFilename(streamReadString(streamId))
36  self.force = streamReadBool(streamId)
37  self:run(connection)
38  end
```

writeStream

Description

Writes network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
44  function PlayerSetHandToolEvent:writeStream(streamId, connection)
45  NetworkUtil.writeNodeObject(streamId, self.player)
46  streamWriteString(streamId,
    NetworkUtil.convertToNetworkFilename(self.handtoolFileName))
47  streamWriteBool(streamId, self.force)
48  end
```

run

Description

Run event

Definition

run(table connection)

Arguments

table connection connection information

Code

```
53  function PlayerSetHandToolEvent:run(connection)
54  if not connection:getIsServer() then
55  g_server:broadcastEvent(self, false, connection, self.player)
56  end
57
58  self.player:equipHandtool(self.handtoolFileName, self.force, true)
59  end
```

sendEvent

Description

Create an instance

Definition

sendEvent(table player, integer handtoolFileName, bool noEventSend)

Arguments

table	player	player instance
integer	handtoolFileName	tool identification
bool	noEventSend	if false will send the event

Code

```
66  function PlayerSetHandToolEvent.sendEvent(player, handtoolFileName, force
    noEventSend)
67  if noEventSend == nil or noEventSend == false then
68  if g_server ~= nil then
69  g_server:broadcastEvent(PlayerSetHandToolEvent:new(player,
    handtoolFileName, force), nil, nil, player)
70  else
71  g_client:getServerConnection():sendEvent(PlayerSetHandToolEvent:new(playe
    handtoolFileName, force))
72  end
73  end
74  end
```

PlayerStateActivateObject

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player	instance of player
table stateMachine	instance of the state machine manager

Return Values

table instance instance of object

Code

```
19  function PlayerStateActivateObject:new(player, stateMachine)
20  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateActivateObject_mt)
21
22  self.activateText = ""
23  self.object = nil
24  return self
25  end
```

isAvailable

Description

Definition

isAvailable()

Return Values

bool returns true if player can interact with object

Code

```
30  function PlayerStateActivateObject:isAvailable()
31  for key, object in pairs(g_currentMission.activatableObjects) do
32  if object:getIsActivatable() then
33  self.activateText = object.activateText
34  self.object = object
35  object:drawActivate();
36  return true
37  end
38  end
39  return false
40  end
```

activate

Description

Activate method.

Definition

activate()

Code

```
44  function PlayerStateActivateObject:activate()
45  PlayerStateActivateObject:superClass().activate(self)
46
47  self.object:onActivateObject()
48  for _, v in pairs(g_currentMission.activateListeners) do
49  v:onActivateObject(self.object)
50  end
51  g_currentMission:removeActivatableObject(self.object)
52  self:deactivate()
53  end
```

deactivate

Description

Activate method.

Definition

deactivate()

Code

```
57  function PlayerStateActivateObject:deactivate()
58  self.object = nil
59  self.activateText = ""
60  PlayerStateActivateObject:superClass().deactivate(self)
```

```
61  end
```

PlayerStateAnimalFeed

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player instance of player

table stateMachine instance of the state machine manager

Return Values

table instance instance of object

Code

```
19  function PlayerStateAnimalFeed:new(player, stateMachine)
20  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateAnimalFeed_mt)
21
22  self.isDog = false
23  return self
24  end
```

isAvailable

Description

Check if we can feed an animal.

Definition

isAvailable()

Return Values

bool returns true if player can feed an animal

Code

```
29  function PlayerStateAnimalFeed:isAvailable()
30  self.isDog = false
31  if self.player.isClient and self.player.isEntered and not
    g_gui:getIsGuiVisible() then
32  local playerHandsEmpty =
    self.player.baseInformation.currentHandtool == nil and not
    self.player.isCarryingObject
33  local dogHouse = g_currentMission:getDoghouse(self.player.farmId)
34
35  if playerHandsEmpty and dogHouse ~= nil and dogHouse.isActivatable
    and not getVisibility(dogHouse.foodNode) then
36  self.isDog = true
37  return true
38  end
```

```
39  end
40  return false
41  end
```

activate

Description

Activate method

Definition

activate()

Code

```
45  function PlayerStateAnimalFeed:activate()
46  PlayerStateAnimalFeed:superClass().activate(self)
47  local dogHouse = g_currentMission:getDoghouse(self.player.farmId)
48  dogHouse.dog:feed()
49  self:deactivate()
50  end
```

deactivate

Description

Deactivate method

Definition

deactivate()

Code

```
54  function PlayerStateAnimalFeed:deactivate()
55  PlayerStateAnimalFeed:superClass().deactivate(self)
56  self.isDog = false
57  end
```

PlayerStateAnimalInteract

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player instance of player

table stateMachine instance of the state machine manager

Return Values

table instance instance of object

Code

```
19  function PlayerStateAnimalInteract:new(player, stateMachine)
20  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateAnimalInteract_mt)
21
```

```
22  self.isDog = false
23  self.husbandryInfo = {}
24  self.castDistance = 1.5 -- in m
25  self.interactText = ""
26  return self
27  end
```

isAvailable

Description

Check if we can interact with an animal.

Definition

isAvailable()

Return Values

bool returns true if player can interact with an animal

Code

```
32  function PlayerStateAnimalInteract:isAvailable()
33  self.isDog = false
34  if self.player.isClient and self.player.isEntered and not
    g_gui:getIsGuiVisible() then
35  local playerHandsEmpty =
    self.player.baseInformation.currentHandtool == nil and not
    self.player.isCarryingObject
36  local dogHouse = g_currentMission:getDoghouse(self.player.farmId)
37
38  if playerHandsEmpty and dogHouse ~= nil and
    dogHouse.dog.isActivable then
39  self.isDog = true
40  if dogHouse.dog.entityFollow == nil then
41  self.interactText = g_i18n:getText("action_interactAnimalFollow")
42  else
43  self.interactText =
    g_i18n:getText("action_interactAnimalStopFollow")
44  end
45  return true
46  end
47  end
48
49  self:updateAnimal()
50
51  if self.husbandryInfo.husbandryId ~= nil then
52  local husbandry =
    g_currentMission.husbandries[self.husbandryInfo.husbandryId]
```

```
53  if husbandry ~= nil and husbandry:getCanBeRidden(self.husbandryInfo.visualId) and husbandry:isAnimalDirty(self.husbandryInfo.visualId) then
54  self.interactText = string.format(g_i18n:getText("action_interactAnimalClean"), husbandry:getAnimalName(self.husbandryInfo.visualId))
55  return true
56  end
57  end
58  self.interactText = ""
59  return false
60  end
```

activate

Description

Activate method. If animal is a dog, we pet him.

Definition

activate()

Code

```
64  function PlayerStateAnimalInteract:activate()
65  PlayerStateAnimalInteract:superClass().activate(self)
66
67  if self.isDog then
68  local dogHouse = g_currentMission:getDoghouse(self.player.farmId)
69
70  if dogHouse.dog.entityFollow ~= self.player.rootNode then
71  dogHouse.dog:followEntity(self.player.rootNode)
72  else
73  dogHouse.dog:goToSpawn()
74  end
75  self:deactivate()
76  else
77  if self.husbandryInfo.husbandryId ~= nil then
78  g_soundManager:playSample(self.player.soundInformation.samples.horseBrush
79  end
80  end
81  end
```

deactivate

Description

Deactivate method

Definition

deactivate()

Code

```
85  function PlayerStateAnimalInteract:deactivate()
86  PlayerStateAnimalInteract:superClass().deactivate(self)
87  g_soundManager:stopSample(self.player.soundInformation.samples.horseBrush
88  self.isDog = false
89  self.husbandryInfo = {}
90  end
```

update

Description

Update method

Definition

update(float dt)

Arguments

float dt delta time in ms

Code

```
103 function PlayerStateAnimalInteract:update(dt)
104 self:updateAnimal()
105
106 if self.husbandryInfo.husbandryId ~= nil and
    self.player.inputInformation.interactState ==
    Player.BUTTONSTATES.PRESSED then
107 local husbandry =
    g_currentMission.husbandries[self.husbandryInfo.husbandryId]
108 husbandry:cleanAnimal(self.husbandryInfo.visualId, dt)
109 else
110 self:deactivate()
111 end
112 end
```

PlayerStateAnimalPet

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player instance of player

table stateMachine instance of the state machine manager

Return Values

table instance instance of object

Code

```
19  function PlayerStateAnimalPet:new(player, stateMachine)
20  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateAnimalPet_mt)
```

```
21
22  self.isDog = false
23  return self
24  end
```

isAvailable

Description

Check if we can pet an animal.

Definition

isAvailable()

Return Values

bool returns true if player can pet an animal

Code

```
29  function PlayerStateAnimalPet:isAvailable()
30  self.isDog = false
31  if self.player.isClient and self.player.isEntered and not
    g_gui:getIsGuiVisible() then
32  local playerHandsEmpty =
    self.player.baseInformation.currentHandtool == nil and not
    self.player.isCarryingObject
33  local dogHouse = g_currentMission:getDoghouse(self.player.farmId)
34
35  if playerHandsEmpty and dogHouse ~= nil and
    dogHouse.dog.isActivable then
36  self.isDog = true
37  return true
38  end
39  end
40  return false
41  end
```

activate

Description

Activate method

Definition

activate()

Code

```
45  function PlayerStateAnimalPet:activate()
46  PlayerStateAnimalPet:superClass().activate(self)
47  local dogHouse = g_currentMission:getDoghouse(self.player.farmId)
48  dogHouse.dog:pet()
49  self:deactivate()
50  end
```

deactivate

Description

Deactivate method

Definition

deactivate()

Code

```
54  function PlayerStateAnimalPet:deactivate()
55  PlayerStateAnimalPet:superClass().deactivate(self)
56  self.isDog = false
57  end
```

PlayerStateAnimalRide

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player instance of player

table stateMachine instance of the state machine manager

Return Values

table instance instance of object

Code

```
19  function PlayerStateAnimalRide:new(player, stateMachine)
20  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateAnimalRide_mt)
21
22  self.testedHusbandryInfo = nil
23  self.castDistance = 1.5 -- in m
24  self.timeFadeToBlack = 250
25  return self
26  end
```

isAvailable

Description

Check if we can ride animal.

Definition

isAvailable()

Return Values

bool returns true if player can ride an animal

Code

```
31  function PlayerStateAnimalRide:isAvailable()
32  local cameraX, cameraY, cameraZ =
    localToWorld(self.player.cameraNode, 0.0, 0.0, 0.0)
```

```
33  local dirX, dirY, dirZ = localDirectionToWorld(self.player.cameraNode, 0.0, 0.0, -1.0)
34  local collisionMask = 268435456 -- bit 28
35  self.testedHusbandryInfo = nil
36  raycastClosest(cameraX, cameraY, cameraZ, dirX, dirY, dirZ, "animalRaycastCallback", self.castDistance, self, collisionMask)
37  if self.testedHusbandryInfo ~= nil then
38  return true
39  end
40  return false
41  end
```

activate

Description

Activate method.

Definition

activate()

Code

```
45  function PlayerStateAnimalRide:activate()
46  PlayerStateAnimalRide:superClass().activate(self)
47  g_currentMission:fadeScreen(1, self.timeFadeToBlack, self.endFadeToBlack, self)
48  end
```

endFadeToBlack

Description

Definition

endFadeToBlack()

Code

```
52  function PlayerStateAnimalRide:endFadeToBlack()
53  local husbandry = g_currentMission.husbandries[self.testedHusbandryInfo.husbandryId]
54  if husbandry ~= nil then
55  husbandry:addRideable(self.testedHusbandryInfo.visualId, self.player)
56  end
57  end
```

animalRaycastCallback

Description

Raycast to check if animal husbandry has been detected

Definition

animalRaycastCallback(float dt)

Arguments

float dt delta time in ms

Code

```
62  function PlayerStateAnimalRide:animalRaycastCallback(hitObjectId,
    x, y, z, distance)
63  local husbandryId, visualId =
    getAnimalFromCollisionNode(hitObjectId)
64
65  if husbandryId ~= 0 then
66  local husbandry = g_currentMission.husbandries[husbandryId]
67  if (husbandry ~= nil and husbandry:getOwnerFarmId() ==
    self.player.farmId) and husbandry:getSupportsRiding(visualId)and
    husbandry:getCanBeRidden(visualId) then
68  self.testedHusbandryInfo = {husbandryId=husbandryId,
    visualId=visualId}
69  return true
70  end
71  end
72
73  return false
74  end
```

getRideableName

Description

Definition

getRideableName()

Code

```
78  function PlayerStateAnimalRide:getRideableName()
79  local rideableName = ""
80  if self.testedHusbandryInfo ~= nil then
81  local husbandry =
    g_currentMission.husbandries[self.testedHusbandryInfo.husbandryId]
82  if husbandry ~= nil then
83  rideableName =
    husbandry:getAnimalName(self.testedHusbandryInfo.visualId)
84  end
85  end
86  return rideableName
87  end
```

PlayerStateBase

Description

new

Description

Creating instance of player state.

Definition

new(table player, table stateMachine, table custom_mt)

Arguments

table player	instance of player
table stateMachine	instance of the state machine manager
table custom_mt	meta table

Return Values

table instance instance of object

Code

```
20  function PlayerStateBase:new(player, stateMachine, custom_mt)
21  if custom_mt == nil then
22  custom_mt = PlayerStateBase_mt
23  end
24  local self = setmetatable({}, custom_mt)
25
26  self.isActive = false
27  self.isInDebugMode = false
28  self.player = player
29  self.stateMachine = stateMachine
30
31  return self
32  end
```

delete

Description

Load method

Definition

delete()

Code

```
36  function PlayerStateBase:delete()
37  end
```

load

Description

Load method

Definition

load()

Code

```
41  function PlayerStateBase:load()
42  end
```

activate

Description

Activate method

Definition

activate()

Code

```
46  function PlayerStateBase:activate()
```

```
47  self.isActive = true
48  end
```

deactivate

Description

Deactivate method

Definition

deactivate()

Code

```
52  function PlayerStateBase:deactivate()
53  self.isActive = false
54  end
```

toggleDebugMode

Description

Toggle debug mode

Definition

toggleDebugMode()

Code

```
58  function PlayerStateBase:toggleDebugMode()
59  if self.isInDebugMode then
60  self.isInDebugMode= false
61  else
62  self.isInDebugMode= true
63  end
64  end
```

inDebugMode

Description

Check if we are in debug mode

Definition

inDebugMode()

Code

```
68  function PlayerStateBase:inDebugMode()
69  return self.isInDebugMode
70  end
```

debugDraw

Description

Definition

debugDraw()

Code

```
74  function PlayerStateBase:debugDraw(dt)
75  end
```

getStateMachine

Description

Get manager

Definition

getStateMachine()

Code

```
79  function PlayerStateBase:getStateMachine()
80  return self.stateMachine
81  end
```

update

Description

Update method

Definition

update(float dt)

Arguments

float dt delta time in ms

Code

```
86  function PlayerStateBase:update(dt)
87  end
```

updateTick

Description

Network tick update method

Definition

updateTick(float dt)

Arguments

float dt delta time in ms

Code

```
92  function PlayerStateBase:updateTick(dt)
93  end
```

PlayerStateCrouch

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player instance of player

table stateMachine instance of the state machine manager

Return Values

table instance instance of object

Code

```
27  function PlayerStateCrouch:new(player, stateMachine)
28  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateCrouch_mt)
```

```
29
30  self.crouchTime = 0.25 -- in sec
31  self.crouchFactor = 0
32  self.crouchYOffset = 0.8
33  self.cameraY = 0.0
34  self.crouchCameraY = 0.0
35  self.progress = PlayerStateCrouch.PROGRESS_STATES.UP
36  self.toggleMode = false
37  return self
38  end
```

Base:load

Description

Load method.

Definition

Base:load()

Code

```
42  function PlayerStateBase:load()
43  self.cameraY = self.player.camY
44  end
```

isAvailable

Description

Check if state is available. Player cannot crouch while in water

Definition

isAvailable()

Return Values

bool true if player can crouch

Code

```
49  function PlayerStateCrouch:isAvailable()
50  if self.player.baseInformation.isInWater then
51  return false
52  end
53
54  if self.player:hasHandtoolEquipped() and
    self.player.baseInformation.currentHandtool.activatePressed then
55  return false
56  end
57  return true
58  end
```

deactivate

Description

Deactivate method. Player camera node translation is set.

Definition

deactivate()

Code

```
62  function PlayerStateCrouch:deactivate()
63  PlayerStateCrouch:superClass().deactivate(self)
64
65  setTranslation(self.player.cameraNode, 0.0, self.cameraY, 0.0)
66  end
```

update

Description

Update method. Slowly update camera Y position.

Definition

update()

Code

```
70  function PlayerStateCrouch:update(dt)
71  -- print(string.format("-- [PlayerStateCrouch:update] toggle(%s)
    progress(%d) inputState(%d) factor(%.3f)",
    tostring(self.toggleMode), self.progress,
    self.player.inputInformation.crouchState, self.crouchFactor))
72  self:processInput()
73  self:moveCamera(dt)
74
75  if self.crouchFactor == 1.0 then
76  self.progress = PlayerStateCrouch.PROGRESS_STATES.CROUCH
77  elseif self.crouchFactor == 0.0 then
78  self.progress = PlayerStateCrouch.PROGRESS_STATES.UP
79  end
80  end
```

processInput

Description

Definition

processInput()

Code

```
84  function PlayerStateCrouch:processInput()
85  if self.player.baseInformation.currentHandtool == nil or
    self.player:hasHandtoolEquipped() and not
    self.player.baseInformation.currentHandtool.activatePressed then
86  local crouchState = self.progress
87  local inputState = self.player.inputInformation.crouchState
88
89  local isCrouching = crouchState ==
    PlayerStateCrouch.PROGRESS_STATES.CROUCHING or crouchState ==
    PlayerStateCrouch.PROGRESS_STATES.CROUCH
```

```
90  local isStanding = crouchState == PlayerStateCrouch.PROGRESS_STATES.STANDING or crouchState == PlayerStateCrouch.PROGRESS_STATES.UP
91
92  if self.toggleMode then
93  if inputState == Player.BUTTONSTATES.PRESSED then
94  if isCrouching then
95  self.progress = PlayerStateCrouch.PROGRESS_STATES.STANDING
96  elseif isStanding then
97  self.progress = PlayerStateCrouch.PROGRESS_STATES.CROUCHING
98  end
99  end
100 else
101 if inputState == Player.BUTTONSTATES.PRESSED and isStanding then
102 self.progress = PlayerStateCrouch.PROGRESS_STATES.CROUCHING
103 elseif inputState == Player.BUTTONSTATES.RELEASED and isCrouching then
104 self.progress = PlayerStateCrouch.PROGRESS_STATES.STANDING
105 end
106 end
107 end
108 end
```

moveCamera

Description

Definition

moveCamera()

Code

```
112 function PlayerStateCrouch:moveCamera(dt)
113 if self.progress == PlayerStateCrouch.PROGRESS_STATES.CROUCHING then
114 local dtInSec = dt * 0.001
115 self.crouchFactor = math.min(1.0, self.crouchFactor + dtInSec / self.crouchTime)
116 elseif self.progress == PlayerStateCrouch.PROGRESS_STATES.CROUCH then
117 elseif self.progress == PlayerStateCrouch.PROGRESS_STATES.STANDING then
118 local dtInSec = dt * 0.001
119 self.crouchFactor = math.max(0.0, self.crouchFactor - dtInSec / self.crouchTime)
120 elseif self.progress == PlayerStateCrouch.PROGRESS_STATES.UP then
121 self:deactivate()
122 end
```

```
123  self.crouchCameraY = self.cameraY - self.crouchYOffset *
     self.crouchFactor
124  setTranslation(self.player.cameraNode, 0.0, self.crouchCameraY,
     0.0)
125  end
```

PlayerStateCycleHandtool

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player instance of player

table stateMachine instance of the state machine manager

Return Values

table instance instance of object

Code

```
19  function PlayerStateCycleHandtool:new(player, stateMachine)
20  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateCycleHandtool_mt)
21
22  self.cycleDirection = 1
23  return self
24  end
```

isAvailable

Description

Definition

isAvailable()

Return Values

bool true if player can idle

Code

```
29  function PlayerStateCycleHandtool:isAvailable()
30  local farm = g_farmManager:getFarmById(self.player.farmId)
31
32  return #farm.handTools > 0
33  end
```

activate

Description

Activate method.

Definition

activate()

Code

```lua
function PlayerStateCycleHandtool:activate()
PlayerStateCycleHandtool:superClass().activate(self)

local farm = g_farmManager:getFarmById(self.player.farmId)
local handTools = farm.handTools
local currentId = 0

if self.player:hasHandtoolEquipped() then
local currentConfigFileName =
self.player.baseInformation.currentHandtool.configFileName

for key, filename in pairs(handTools) do
if filename == currentConfigFileName then
currentId = key
break
end
end
end

currentId = currentId + self.cycleDirection
if currentId > #handTools then
currentId = 0
elseif currentId < 0 then
currentId = #handTools
end

if currentId == 0 then
self.player:equipHandtool("", false)
else
self.player:equipHandtool(handTools[currentId], false)
end

self:deactivate()
end
```

PlayerStateDrop

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player instance of player

table stateMachine instance of the state machine manager

Return Values

table instance instance of object

Code

```
19  function PlayerStateDrop:new(player, stateMachine)
20  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateDrop_mt)
21
22  return self
23  end
```

isAvailable

Description

Definition

isAvailable()

Return Values

bool true if player can idle

Code

```
28  function PlayerStateDrop:isAvailable()
29  if self.player.isCarryingObject then
30  return true
31  end
32  return false
33  end
```

activate

Description

Activate method.

Definition

activate()

Code

```
37  function PlayerStateDrop:activate()
38  PlayerStateDrop:superClass().activate(self)
39
40  self.player:pickUpObject(false)
41  self:deactivate()
42  end
```

PlayerStateFall

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player instance of player

table stateMachine instance of the state machine manager

Return Values

table instance instance of object

Code

```
19  function PlayerStateFall:new(player, stateMachine)
20  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateFall_mt)
21
22  return self
23  end
```

isAvailable

Description

Check if state is available. If player is not on the ground and is not in water and vertical velocity lower than

Definition

isAvailable()

Return Values

bool true if state is available

Code

```
28  function PlayerStateFall:isAvailable()
29  local isOnGround = self.player.baseInformation.isOnGround
30  local isInWater = self.player.baseInformation.isInWater
31  local verticalVelocity = self.player.motionInformation.velocity[2]
32
33  if not isOnGround and not isInWater and (verticalVelocity <
    self.player.motionInformation.minimumFallingSpeed) then
34  return true
35  end
36  return false
37  end
```

activate

Description

Activate method. Set vertical velocity.

Definition

activate()

Code

```
41  function PlayerStateFall:activate()
42  PlayerStateFall:superClass().activate(self)
43  end
```

deactivate

Description

Deactivate method.

Definition

deactivate()

Code

```
47  function PlayerStateFall:deactivate()
48  PlayerStateFall:superClass().deactivate(self)
49
50  self.player:setVelocityToMotion(nil, 0.0, nil)
51  end
```

update

Description

Update method. Will deactivate when player hits the ground or is in water

Definition

update(float dt)

Arguments

float dt delta time in ms

Code

```
56  function PlayerStateFall:update(dt)
57  local isOnGround = self.player.baseInformation.isOnGround
58  local isInWater = self.player.baseInformation.isInWater
59
60  if isOnGround or isInWater then
61  self:deactivate()
62  end
63  end
```

PlayerStateIdle

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player instance of player

table stateMachine instance of the state machine manager

Return Values

table instance instance of object

Code

```
19  function PlayerStateIdle:new(player, stateMachine)
20  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateIdle_mt)
```

```
21
22  return self
23  end
```

isAvailable

Description

Check if state is available. Always true

Definition

isAvailable()

Return Values

bool true if player can idle

Code

```
28  function PlayerStateIdle:isAvailable()
29  return true
30  end
```

update

Description

Update method. Will deactivate if player moves or if not on the ground.

Definition

update(float dt)

Arguments

float dt delta time in ms

Code

```
35  function PlayerStateIdle:update(dt)
36  local playerInputsCheck = (
    self.player.inputInformation.moveForward ~= 0) or
    (self.player.inputInformation.moveRight ~= 0)
37
38  if playerInputsCheck or not self.player.baseInformation.isOnGround
    then
39  self:deactivate()
40  end
41  end
```

PlayerStateJump

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player	instance of player
table stateMachine	instance of the state machine manager

Return Values

table instance instance of object

Code

```
19  function PlayerStateJump:new(player, stateMachine)
20  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateJump_mt)
21
22  self.desiredHeight = 1.0 -- in m
23  -- debug code
24  self.playerPos = {}
25  self.jumpDuration = 0.0
26
27  return self
28  end
```

delete

Description

Definition

delete()

Code

```
32  function PlayerStateJump:delete()
33  if self.player.isOwner then
34  removeConsoleCommand("gsToggleDebugStateJump")
35  end
36  end
```

load

Description

Load method

Definition

load()

Code

```
40  function PlayerStateJump:load()
41  if self.player.isOwner then
42  addConsoleCommand("gsToggleDebugStateJump", "Toggle debug mode for
    Jump", "consoleCommandDebugStateJump", self)
43  end
44  end
```

isAvailable

Description

Check if state is available. Check that player is on the ground.

Definition

isAvailable()

Return Values

bool true if player can jump

Code

```
49  function PlayerStateJump:isAvailable()
50  local isOnGround = self.player.baseInformation.isOnGround
51
52  return isOnGround and not g_currentMission:isInGameMessageActive()
53  end
```

activate

Description

Activate method. Set vertical velocity.

Definition

activate()

Code

```
57  function PlayerStateJump:activate()
58  PlayerStateJump:superClass().activate(self)
59  local desiredVerticalVelocity =
    self:calculateVerticalVelocityForHeight(self.desiredHeight)
60
61  self.player:setVelocityToMotion(nil, desiredVerticalVelocity, nil)
62
63  if self:inDebugMode() then
64  self.playerPos = {}
65  self.jumpDuration = 0.0
66  end
67  end
```

deactivate

Description

Deactivate method.

Definition

deactivate()

Code

```
71  function PlayerStateJump:deactivate()
72  PlayerStateJump:superClass().deactivate(self)
73
74  self.jumpWeight = 0
75  end
```

update

Description

Update method. If the fall player state is available, we deactive this state.

Definition

update(float dt)

Arguments

float dt delta time in ms

Code

```
80  function PlayerStateJump:update(dt)
81  if self.stateMachine:isAvailable("fall") then
82  if self:inDebugMode() then
83  local startPos = self.playerPos[1]
84  local endPos = self.playerPos[#self.playerPos]
85  local delta = endPos[2] - startPos[2]
86  print(string.format("[PlayerStateJump:update] End jump /
    duration(%.4f s) / height(%.4f).", self.jumpDuration * 0.001,
    delta))
87  end
88  self:deactivate()
89  end
90
91  -- debug code
92  if self:inDebugMode() then
93  local posX, posY, posZ =
    getWorldTranslation(g_currentMission.player.rootNode)
94  local newEntry = {posX, posY, posZ}
95
96  table.insert(self.playerPos, newEntry)
97  self.jumpDuration = self.jumpDuration + dt
98  end
99  end
```

debugDraw

Description

Debug draw method.

Definition

debugDraw(float dt)

Arguments

float dt delta time in ms

Code

```
104  function PlayerStateJump:debugDraw(dt)
105  for i=1, #self.playerPos do
106  DebugUtil.drawDebugCircle( self.playerPos[i][1],
     self.playerPos[i][2], self.playerPos[i][3], 0.1, 10)
107  end
108  end
```

consoleCommandDebugStateJump

Description

Console command to debug draw the jump state

Definition

consoleCommandDebugStateJump()

Code

```
112  function PlayerStateJump:consoleCommandDebugStateJump()
113  self:toggleDebugMode()
114  if self:inDebugMode() then
115  self.playerPos = {}
116  end
117  end
```

calculateVerticalVelocityForHeight

Description

Calculate the initial vertical velocity according to the desired height we want the player to reach.

Definition

calculateVerticalVelocityForHeight(float desiredHeight)

Arguments

float desiredHeight height in m we want the player to reach

Return Values

float initial vertical velocity in m/s

Code

```
124  function PlayerStateJump:calculateVerticalVelocityForHeight(
     desiredHeight )
125  if ( desiredHeight <= 0.0 ) or (self.player.motionInformation.gravity[2]
     >= 0.0) then
126  return 0.0
127  end
128
129  local result = math.sqrt(-2.0 * self.player.motionInformation.gravity[2]
     * desiredHeight)
130  -- local duration = math.sqrt( -2.0 * desiredHeight /
     self.player.motionInformation.gravity[2])
131  -- local result = 2.0 * desiredHeight / duration
132
133  if self:inDebugMode() then
134  print(string.format("[PlayerStateJump:CalculateVerticalVelocityForHeight
     gravity(%.4f) desiredHeight(%.4f) result(%.4f)",
     self.player.motionInformation.gravity[2], desiredHeight, result ))
135  end
136  return result
137  end
```

PlayerStateMachine

Description

new

Description

Creating instance of player state machine. Initializing member variables: player states and a table containing those states.

Definition

new(table player, table custom_mt)

Arguments

table player instance of player

table custom_mt meta table

Return Values

table instance instance of object

Code

```
19  function PlayerStateMachine:new(player, custom_mt)
20  if custom_mt == nil then
21  custom_mt = PlayerStateMachine_mt
22  end
23  local self = setmetatable({}, custom_mt)
24  self.player = player
25
26  -- State Machine information
27  self.playerStateIdle = PlayerStateIdle:new(self.player, self)
28  self.playerStateWalk = PlayerStateWalk:new(self.player, self)
29  self.playerStateRun = PlayerStateRun:new(self.player, self)
30  self.playerStateJump = PlayerStateJump:new(self.player, self)
31  self.playerStateSwim = PlayerStateSwim:new(self.player, self)
32  self.playerStateFall = PlayerStateFall:new(self.player, self)
33  self.playerStateCrouch = PlayerStateCrouch:new(self.player, self)
34  self.playerStateAnimalInteract =
    PlayerStateAnimalInteract:new(self.player, self)
35  self.playerStateAnimalRide =
    PlayerStateAnimalRide:new(self.player, self)
36  self.playerStateAnimalFeed =
    PlayerStateAnimalFeed:new(self.player, self)
37  self.playerStateAnimalPet = PlayerStateAnimalPet:new(self.player,
    self)
38  self.playerStatePickup = PlayerStatePickup:new(self.player, self)
39  self.playerStateDrop = PlayerStateDrop:new(self.player, self)
40  self.playerStateThrow = PlayerStateThrow:new(self.player, self)
41  self.playerStateActicateObject =
    PlayerStateActivateObject:new(self.player, self)
42  self.playerStateUseLight = PlayerStateUseLight:new(self.player,
    self)
43  self.playerStateCycleHandtool =
    PlayerStateCycleHandtool:new(self.player, self)
44
```

```
self.stateList = { ["idle"] = self.playerStateIdle,
["walk"] = self.playerStateWalk,
["run"] = self.playerStateRun,
["jump"] = self.playerStateJump,
["swim"] = self.playerStateSwim,
["fall"] = self.playerStateFall,
["crouch"] = self.playerStateCrouch,
["animalInteract"] = self.playerStateAnimalInteract,
["animalRide"] = self.playerStateAnimalRide,
["animalFeed"] = self.playerStateAnimalFeed,
["animalPet"] = self.playerStateAnimalPet,
["pickup"] = self.playerStatePickup,
["drop"] = self.playerStateDrop,
["throw"] = self.playerStateThrow,
["activateObject"] = self.playerStateActicateObject,
["useLight"] = self.playerStateUseLight,
["cycleHandtool"] = self.playerStateCycleHandtool,
}

-- field [from][to] : allowed
self.fsmTable = {}
self.fsmTable["walk"] = {}
self.fsmTable["walk"]["jump"] = true
self.fsmTable["walk"]["run"] = true
self.fsmTable["walk"]["swim"] = true
self.fsmTable["walk"]["crouch"] = true
self.fsmTable["walk"]["pickup"] = true
self.fsmTable["walk"]["drop"] = true
self.fsmTable["walk"]["throw"] = true
self.fsmTable["walk"]["activateObject"] = true
self.fsmTable["walk"]["useLight"] = true
self.fsmTable["walk"]["cycleHandtool"] = true
self.fsmTable["run"] = {}
self.fsmTable["run"]["jump"] = true
self.fsmTable["run"]["swim"] = true
self.fsmTable["run"]["pickup"] = true
self.fsmTable["run"]["drop"] = true
self.fsmTable["run"]["throw"] = true
self.fsmTable["run"]["activateObject"] = true
self.fsmTable["run"]["useLight"] = true
```

```
self.fsmTable["run"]["cycleHandtool"] = true
self.fsmTable["crouch"] = {}
self.fsmTable["crouch"]["walk"] = true
self.fsmTable["crouch"]["jump"] = true
self.fsmTable["crouch"]["swim"] = true
self.fsmTable["crouch"]["animalInteract"] = true
self.fsmTable["crouch"]["animalRide"] = true
self.fsmTable["crouch"]["animalPet"] = true
self.fsmTable["crouch"]["animalFeed"] = true
self.fsmTable["crouch"]["pickup"] = true
self.fsmTable["crouch"]["drop"] = true
self.fsmTable["crouch"]["throw"] = true
self.fsmTable["crouch"]["activateObject"] = true
self.fsmTable["crouch"]["useLight"] = true
self.fsmTable["crouch"]["cycleHandtool"] = true
self.fsmTable["fall"] = {}
self.fsmTable["fall"]["swim"] = true
self.fsmTable["fall"]["useLight"] = true
self.fsmTable["jump"] = {}
self.fsmTable["idle"] = {}
self.fsmTable["idle"]["jump"] = true
self.fsmTable["idle"]["crouch"] = true
self.fsmTable["idle"]["walk"] = true
self.fsmTable["idle"]["run"] = true
self.fsmTable["idle"]["animalInteract"] = true
self.fsmTable["idle"]["animalRide"] = true
self.fsmTable["idle"]["animalPet"] = true
self.fsmTable["idle"]["animalFeed"] = true
self.fsmTable["idle"]["pickup"] = true
self.fsmTable["idle"]["drop"] = true
self.fsmTable["idle"]["throw"] = true
self.fsmTable["idle"]["activateObject"] = true
self.fsmTable["idle"]["useLight"] = true
self.fsmTable["idle"]["cycleHandtool"] = true
self.fsmTable["swim"] = {}
self.fsmTable["swim"]["walk"] = true
self.fsmTable["swim"]["run"] = true
self.fsmTable["swim"]["useLight"] = true
self.fsmTable["animalInteract"] = {}
self.fsmTable["animalInteract"]["crouch"] = true
```

```
125  self.fsmTable["animalInteract"]["idle"] = true
126  self.fsmTable["animalInteract"]["walk"] = true
127  self.fsmTable["animalInteract"]["run"] = true
128  self.fsmTable["animalFeed"] = {}
129  self.fsmTable["animalFeed"]["crouch"] = true
130  self.fsmTable["animalFeed"]["idle"] = true
131  self.fsmTable["animalFeed"]["walk"] = true
132  self.fsmTable["animalFeed"]["run"] = true
133  self.fsmTable["animalPet"] = {}
134  self.fsmTable["animalPet"]["crouch"] = true
135  self.fsmTable["animalPet"]["idle"] = true
136  self.fsmTable["animalPet"]["walk"] = true
137  self.fsmTable["animalPet"]["run"] = true
138
139  self.debugMode = false
140  return self
141  end
```

delete

Description

Methods for deleting player state machine

Definition

delete()

Code

```
145  function PlayerStateMachine:delete()
146  if self.player.isOwner then
147  removeConsoleCommand("gsToggleDebugPlayerFSM")
148  end
149
150  for _, stateInstance in pairs(self.stateList) do
151  stateInstance:delete()
152  stateInstance = {}
153  end
154  end
```

getState

Description

Returns a player state

Definition

getState(string stateName)

Arguments

string stateName name of the state to search for

Return Values

table player state

Code

```
160  function PlayerStateMachine:getState(stateName)
161  return self.stateList[stateName]
162  end
```

isAvailable

Description

Check if a player state is available and not already active.

Definition

isAvailable(string stateName)

Arguments

string stateName

Return Values

bool true if player state is available

Code

```
168  function PlayerStateMachine:isAvailable(stateName)
169  if self.stateList[stateName] ~= nil then
170  local result = (self.stateList[stateName].isActive == false) and
     self.stateList[stateName]:isAvailable()
171
172  return result
173  end
174  return false
175  end
```

isActive

Description

Check if a player state is active

Definition

isActive(string stateName)

Arguments

string stateName

Return Values

bool true if player state is active

Code

```
181  function PlayerStateMachine:isActive(stateName)
182  if self.stateList[stateName] ~= nil then
183  return self.stateList[stateName].isActive
184  end
185  return false
186  end
```

update

Description

Execute all update methods of active player states

Definition

update(float dt)

Arguments

float dt delta time in ms

Code

```
191  function PlayerStateMachine:update(dt)
192  for stateName, stateInstance in pairs(self.stateList) do
193  if stateInstance.isActive then
194  stateInstance:update(dt)
195  end
196  end
197  end
```

updateTick

Description

Execute all update methods when network tick of active player states

Definition

updateTick(float dt)

Arguments

float dt delta time in ms

Code

```
202  function PlayerStateMachine:updateTick(dt)
203  for stateName, stateInstance in pairs(self.stateList) do
204  if stateInstance.isActive then
205  stateInstance:updateTick(dt)
206  end
207  end
208  end
```

debugDraw

Description

Execute all debug draw methods. Displays is states are active and available. Also draw internal player state debug method.

Definition

debugDraw(float dt)

Arguments

float dt delta time in ms

Code

```
213  function PlayerStateMachine:debugDraw(dt)
214  if self.debugMode then
215  setTextColor(1, 1, 0, 1)
216  renderText(0.05, 0.60, 0.02, "[state machine]")
217  local i = 0
218  for stateName, stateInstance in pairs(self.stateList) do
```

```
219  renderText(0.05, 0.58 - i * 0.02 , 0.02, string.format("- %s
     active(%s) isAvailable(%s)", stateName,
     tostring(stateInstance.isActive),
     tostring(stateInstance:isAvailable())))
220  i = i + 1
221  end
222  end
223
224  for stateName, stateInstance in pairs(self.stateList) do
225  if stateInstance.inDebugMode(self) then
226  stateInstance:debugDraw(dt)
227  end
228  end
229  end
```

activateState

Description

Activates a player state. Checks if active states allows to use the player state we want to activate. If allowed, the state is activated.

Definition

activateState(string stateNameTo)

Arguments

string stateNameTo the player state we want to activate

Code

```
234  function PlayerStateMachine:activateState(stateNameTo)
235  local allowed = true
236
237  for stateNameFrom, stateInstance in pairs(self.stateList) do
238  if stateInstance.isActive and (self.fsmTable[stateNameFrom] ==
     nil or not self.fsmTable[stateNameFrom][stateNameTo]) then
239  allowed = false
240  break
241  end
242  end
243
244  -- if self.debugMode then
245  -- print(string.format("-- [PlayerStateMachine:activateState]
     state(%s) allowed(%s) active(%s)", stateNameTo,
     tostring(allowed),
     tostring(self.stateList[stateNameTo].isActive)))
246  -- end
247  if allowed and (self.stateList[stateNameTo] ~= nil) and
     (self.stateList[stateNameTo].isActive == false) then
248  self.stateList[stateNameTo]:activate()
```

```
249 end
250 end
```

deactivateState

Description

Deactivates a player state

Definition

deactivateState(string stateName)

Arguments

string stateName

Code

```
255 function PlayerStateMachine:deactivateState(stateName)
256 if (self.stateList[stateName] ~= nil) and
    (self.stateList[stateName].isActive == true) then
257 self.stateList[stateName]:deactivate()
258 end
259 end
```

load

Description

Loads states

Definition

load()

Code

```
263 function PlayerStateMachine:load()
264 for _, stateInstance in pairs(self.stateList) do
265 stateInstance:load()
266 end
267
268 -- Console commands
269 -- self.player.isOwner is init at this point
270 if self.player.isOwner then
271 addConsoleCommand("gsToggleDebugPlayerFSM", "Toggle debug mode
    for player state machine",
    "consoleCommandDebugFinalStateMachine", self)
272 end
273 end
```

consoleCommandDebugFinalStateMachine

Description

Console command to toggle debug of player state machine

Definition

consoleCommandDebugFinalStateMachine()

Code

```
277 function PlayerStateMachine:consoleCommandDebugFinalStateMachine()
278 if self.debugMode then
279 self.debugMode = false
280 else
281 self.debugMode = true
282 end
283 end
```

PlayerStatePickup
Description
new
Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player instance of player
table stateMachine instance of the state machine manager

Return Values

table instance instance of object

Code

```
19 function PlayerStatePickup:new(player, stateMachine)
20 local self = PlayerStateBase:new(player, stateMachine, PlayerStatePickup_mt)
21
22 return self
23 end
```

isAvailable
Description
Definition

isAvailable()

Return Values

bool true if player can idle

Code

```
28 function PlayerStatePickup:isAvailable()
29 if self.player.isClient and self.player.isEntered and not self.player:hasHandtoolEquipped() then
30 if not self.player.isCarryingObject and self.player.isObjectInRange and r
   self.player.usesChainsaw then
31 if self.player.lastFoundObjectMass <= Player.MAX_PICKABLE_OBJECT_MASS the
32 return true
33 else
34 g_currentMission:addExtraPrintText(g_i18n:getText("warning_objectTooHeavy
```

```
35  end
36  end
37  end
38  return false
39  end
```

activate

Description

Activate method.

Definition

activate()

Code

```
43  function PlayerStatePickup:activate()
44  PlayerStatePickup:superClass().activate(self)
45
46  self.player:pickUpObject(true)
47  self:deactivate()
48  end
```

PlayerStateRun

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player	instance of player
table stateMachine	instance of the state machine manager

Return Values

table instance instance of object

Code

```
19  function PlayerStateRun:new(player, stateMachine)
20  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateRun_mt)
21
22  return self
23  end
```

isAvailable

Description

Check if state is available.

Definition

isAvailable()

Return Values

bool true if player can run

Code

```
28  function PlayerStateRun:isAvailable()
29  return self:canRun()
30  end
```

update

Description

Update method. Will deactivate if player stops moving, is not on the ground or if the run input is not pressed.

Definition

update(float dt)

Arguments

float dt delta time in ms

Code

```
35  function PlayerStateRun:update(dt)
36  local playerInputsCheck = (self.player.inputInformation.runAxis ~= 0.0) and (( self.player.inputInformation.moveForward ~= 0) or (self.player.inputInformation.moveRight ~= 0))
37
38  if (self:canRun() == false) or (playerInputsCheck == false) then
39  self:deactivate()
40  end
41  end
```

canRun

Description

Check that player is on the ground.

Definition

canRun()

Return Values

bool true if player can run

Code

```
46  function PlayerStateRun:canRun()
47  return self.player.baseInformation.isOnGround
48  end
```

PlayerStateSwim

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player	instance of player

table stateMachine instance of the state machine manager

Return Values

table instance instance of object

Code

```
20  function PlayerStateSwim:new(player, stateMachine)
21  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateSwim_mt)
22
23  return self
24  end
```

isAvailable

Description

Check if state is available if player is in water

Definition

isAvailable()

Return Values

bool true if player can swim

Code

```
29  function PlayerStateSwim:isAvailable()
30  local isInWater = self.player.baseInformation.isInWater
31
32  if isInWater then
33  return true
34  end
35  return false
36  end
```

update

Description

Update method. Will deactivate if player is not in water anymore

Definition

update(float dt)

Arguments

float dt delta time in ms

Code

```
41  function PlayerStateSwim:update(dt)
42  if not self.player.baseInformation.isInWater then
43  self:deactivate()
44  end
45  end
```

PlayerStateThrow

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player instance of player

table stateMachine instance of the state machine manager

Return Values

table instance instance of object

Code

```
19  function PlayerStateThrow:new(player, stateMachine)
20  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateThrow_mt)
21
22  return self
23  end
```

isAvailable

Description

Definition

isAvailable()

Return Values

bool true if player can idle

Code

```
28  function PlayerStateThrow:isAvailable()
29  if self.player.isClient and self.player.isEntered and not
    self.player:hasHandtoolEquipped() then
30  if self.player.isCarryingObject or (not
    self.player.isCarryingObject and (self.player.isObjectInRange and
    self.player.lastFoundObject ~= nil)) then
31  if self.player.lastFoundObjectMass <=
    Player.MAX_PICKABLE_OBJECT_MASS then
32  return true
33  end
34  end
35  end
36  return false
37  end
```

activate

Description

Activate method.

Definition

activate()

Code

```
41  function PlayerStateThrow:activate()
```

```
42  PlayerStateThrow:superClass().activate(self)
43
44  self.player:throwObject()
45  self:deactivate()
46  end
```

PlayerStateUseLight
Description
new
Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player instance of player
table stateMachine instance of the state machine manager

Return Values

table instance instance of object

Code

```
19  function PlayerStateUseLight:new(player, stateMachine)
20  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateUseLight_mt)
21
22  return self
23  end
```

isAvailable
Description
Definition

isAvailable()

Return Values

bool true if player can idle

Code

```
28  function PlayerStateUseLight:isAvailable()
29  if self.player.lightNode ~= nil and not
    g_currentMission:isInGameMessageActive() then
30  return true
31  end
32  return false
33  end
```

activate
Description

Activate method.

Definition

activate()

Code

```
37  function PlayerStateUseLight:activate()
38  PlayerStateUseLight:superClass().activate(self)
39
40  self.player:setLightIsActive(not self.player.isLightActive)
41  self:deactivate()
42  end
```

PlayerStateWalk

Description

new

Description

Creating instance of state.

Definition

new(table player, table stateMachine)

Arguments

table player instance of player

table stateMachine instance of the state machine manager

Return Values

table instance instance of object

Code

```
19  function PlayerStateWalk:new(player, stateMachine)
20  local self = PlayerStateBase:new(player, stateMachine,
    PlayerStateWalk_mt)
21
22  return self
23  end
```

isAvailable

Description

Check if state is available

Definition

isAvailable()

Return Values

bool true if player can swim

Code

```
28  function PlayerStateWalk:isAvailable()
29  return self:canWalk()
30  end
```

update

Description

Update method. Will deactivate if player is not moving anymore or if he starts running.

Definition

update(float dt)

Arguments

float dt delta time in ms

Code

```lua
function PlayerStateWalk:update(dt)
local playerInputsCheck = (self.player.inputInformation.moveForward ~= 0) or (self.player.inputInformation.moveRight ~= 0)

if not self:canWalk() or not playerInputsCheck then
self:deactivate()
end
end
```

canWalk

Description

Check if player is on the ground and he is not running

Definition

canWalk()

Return Values

bool true if player can swim

Code

```lua
function PlayerStateWalk:canWalk()
local isRunning = (self.player.inputInformation.runAxis ~= 0.0)

return self.player.baseInformation.isOnGround and not isRunning
end
```

PlayerStyle

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```lua
function PlayerStyle:new(customMt)
local self = {}
setmetatable(self, customMt or PlayerStyle_mt)

self.player = nil
self.selectedModelIndex = 1
self.selectedColorIndex = 0
self.selectedBodyIndex = 0
```

```
26  self.selectedHatIndex = 0
27  self.selectedAccessoryIndex = 0
28  self.selectedHairIndex = 0
29  self.selectedJacketIndex = 0
30  self.playerName = "player"
31  self.bodies = {}
32  self.playerHatNode = nil
33  self.playerAccessoryNode = nil
34  self.accessoryNode = nil
35  self.hatNode = nil
36  self.hatReferenceFilename = ""
37  self.accessoryReferenceFilename = ""
38  self.accessories = {}
39  self.hairs = {}
40  self.hairs.hatStyleNode = nil
41  self.hairs.styles = {}
42  self.jackets = {}
43  self.protection = {}
44  self.protection.helmetNode = nil
45  self.protection.glovesNode = nil
46  self.protection.isVisible = false
47  self.useDefault = false
48
49  return self
50  end
```

copySelection

Description

Definition

copySelection()

Code

```
54  function PlayerStyle:copySelection(other)
55  self.selectedModelIndex = other.selectedModelIndex
56  self.selectedColorIndex = other.selectedColorIndex
57  self.selectedBodyIndex = other.selectedBodyIndex
58  self.selectedHatIndex = other.selectedHatIndex
59  self.selectedAccessoryIndex = other.selectedAccessoryIndex
60  self.selectedHairIndex = other.selectedHairIndex
61  self.selectedJacketIndex = other.selectedJacketIndex
62  self.playerName = other.playerName
63  self.useDefault = other.useDefault
64  end
```

applySelection
Description
Definition

applySelection()

Code

```lua
function PlayerStyle:applySelection()
self:setBody(self.selectedBodyIndex)
self:setHair(self.selectedHairIndex)
self:setHat(self.selectedHatIndex)
self:setAccessory(self.selectedAccessoryIndex)
self:setJacket(self.selectedJacketIndex)
self:setColor(self.selectedColorIndex)
end
```

loadXML
Description
Definition

loadXML()

Code

```lua
function PlayerStyle:loadXML(player, playerRootNode, xmlFile, baseKey)
local i = 0

self.player = player
-- bodies
i = 0
while true do
local headKey = string.format("%s.playerStyle.bodies.variant(%d)#headNode", baseKey, i)
local armsKey = string.format("%s.playerStyle.bodies.variant(%d)#armNode", baseKey, i)
if not hasXMLProperty(xmlFile, headKey) or not hasXMLProperty(xmlFile, armsKey) then
break
end
local headNode = I3DUtil.indexToObject(playerRootNode, getXMLString(xmlFile, headKey))
local armsNode = I3DUtil.indexToObject(playerRootNode, getXMLString(xmlFile, armsKey))
table.insert(self.bodies, {headNode=headNode, armsNode=armsNode})
i = i + 1
end
```

```
96  -- hats
97  self.playerHatNode = I3DUtil.indexToObject(playerRootNode,
    getXMLString(xmlFile,
    string.format("%s.playerStyle.hats#rootNode", baseKey)))
98  self.hatReferenceFilename = getXMLString(xmlFile,
    string.format("%s.playerStyle.hats#filename", baseKey))
99  -- accessories
100 self.playerAccessoryNode = I3DUtil.indexToObject(playerRootNode,
    getXMLString(xmlFile,
    string.format("%s.playerStyle.hats#rootNode", baseKey)))
101 self.accessoryReferenceFilename = getXMLString(xmlFile,
    string.format("%s.playerStyle.hats#filename", baseKey))
102 -- hairs
103 self.hairs.hatStyleNode = I3DUtil.indexToObject(playerRootNode,
    getXMLString(xmlFile,
    string.format("%s.playerStyle.hairStyles#hatHairNode", baseKey)))
104 i = 0
105 while true do
106 local hairStyleKey =
    string.format("%s.playerStyle.hairStyles.variant(%d)", baseKey,
    i)
107 if not hasXMLProperty(xmlFile, hairStyleKey) then
108 break
109 end
110 local node = I3DUtil.indexToObject(playerRootNode,
    getXMLString(xmlFile, hairStyleKey.."#node"))
111 local name = Utils.getNoNil(getXMLString(xmlFile,
    hairStyleKey.."#name"), "")
112 local isHatHair = Utils.getNoNil(getXMLBool(xmlFile,
    hairStyleKey.."#isHatHair"), false)
113
114 if isHatHair and self.hairs.hatIndex == 0 then
115 self.hairs.hatIndex = i + 1
116 end
117
118 table.insert(self.hairs.styles, {node=node, name=name})
119 i = i + 1
120 end
121 -- jackets
122 i = 0
123 while true do
124 local jacketKey =
    string.format("%s.playerStyle.jackets.variant(%d)", baseKey, i)
125 if not hasXMLProperty(xmlFile, jacketKey) then
```

```
126     break
127     end
128     local node = I3DUtil.indexToObject(playerRootNode,
        getXMLString(xmlFile, jacketKey.."#node"))
129     local name = getXMLString(xmlFile, jacketKey.."#name")
130     table.insert(self.jackets, {node=node, name=name})
131     i = i + 1
132     end
133     -- chainsaw protection
134     self.protection.helmetNode =
        I3DUtil.indexToObject(playerRootNode, getXMLString(xmlFile,
        string.format("%s.playerStyle.helmet#node", baseKey)))
135     self.protection.glovesNode =
        I3DUtil.indexToObject(playerRootNode, getXMLString(xmlFile,
        string.format("%s.playerStyle.gloves#node", baseKey)))
136
137     -- default
138     if self.useDefault then
139     local defaultHatKey = string.format("%s.playerStyle#defaultHat",
        baseKey)
140     if hasXMLProperty(xmlFile, defaultHatKey) then
141     self.selectedHatIndex = getXMLInt(xmlFile, defaultHatKey)
142     end
143     local defaultAccessoryKey =
        string.format("%s.playerStyle#defaultAccessory", baseKey)
144     if hasXMLProperty(xmlFile, defaultAccessoryKey) then
145     self.selectedAccessoryIndex = getXMLInt(xmlFile,
        defaultAccessoryKey)
146     end
147     local defaultHairstyleKey =
        string.format("%s.playerStyle#defaultHaistyle", baseKey)
148     if hasXMLProperty(xmlFile, defaultHairstyleKey) then
149     self.selectedHairIndex = getXMLInt(xmlFile, defaultHairstyleKey)
150     end
151     local defaultBodyKey =
        string.format("%s.playerStyle#defaultBody", baseKey)
152     if hasXMLProperty(xmlFile, defaultBodyKey) then
153     self.selectedBodyIndex = getXMLInt(xmlFile, defaultBodyKey)
154     end
155     end
156     end
```

linkProtectiveWear

Description

Definition

linkProtectiveWear()

Code

```
160  function PlayerStyle:linkProtectiveWear(linkNode)
161  if self.protection.glovesNode ~= nil then
162  link(linkNode, self.protection.glovesNode)
163  end
164  end
```

unlinkProtectiveWear

Description

Definition

unlinkProtectiveWear()

Code

```
168  function PlayerStyle:unlinkProtectiveWear()
169  if self.protection.glovesNode ~= nil then
170  unlink(self.protection.glovesNode)
171  end
172  end
```

readStream

Description

Reads from network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId id of the stream to read

table connection connection information

Code

```
178  function PlayerStyle:readStream(streamId, connection)
179  self.selectedModelIndex = streamReadUIntN(streamId,
     PlayerModelManager.SEND_NUM_BITS)
180  self.selectedColorIndex = streamReadUInt8(streamId)
181  self.selectedBodyIndex = streamReadUInt8(streamId)
182  self.selectedHatIndex = streamReadUInt8(streamId)
183  self.selectedAccessoryIndex = streamReadUInt8(streamId)
184  self.selectedHairIndex = streamReadUInt8(streamId)
185  self.selectedJacketIndex = streamReadUInt8(streamId)
186  self.playerName = streamReadString(streamId)
187  end
```

writeStream

Description

Writes in network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId id of the stream to read

table connection connection information

Code

```
193 function PlayerStyle:writeStream(streamId, connection)
194 streamWriteUIntN(streamId, self.selectedModelIndex,
    PlayerModelManager.SEND_NUM_BITS)
195 streamWriteUInt8(streamId, self.selectedColorIndex)
196 streamWriteUInt8(streamId, self.selectedBodyIndex)
197 streamWriteUInt8(streamId, self.selectedHatIndex)
198 streamWriteUInt8(streamId, self.selectedAccessoryIndex)
199 streamWriteUInt8(streamId, self.selectedHairIndex)
200 streamWriteUInt8(streamId, self.selectedJacketIndex)
201 streamWriteString(streamId, self.playerName)
202 end
```

setColor

Description

Definition

setColor()

Code

```
206 function PlayerStyle:setColor(playerColorIndex)
207 if self.player ~= nil then
208 if self.player.meshThirdPerson ~= nil and
    self.player.meshThirdPerson ~= 0 and playerColorIndex > 0 and
    playerColorIndex <= table.getn(g_playerColors) then
209 local r, g, b, a = unpack(g_playerColors[playerColorIndex].value)
210
211 self.selectedColorIndex = playerColorIndex
212 setShaderParameter(self.player.meshThirdPerson, "colorScaleR", r,
    g, b, a, false)
213 end
214 else
215 g_logManager:devError("-- [PlayerStyle:setColor] Player not set.
    Missing call to PlayerStyle:loadXML().")
216 end
217 end
```

setBody

Description

Definition

setBody()

Code

```
221 function PlayerStyle:setBody(bodyIndex)
```

```
222  local result = false
223  if self.player ~= nil then
224  for key, body in ipairs(self.bodies) do
225  if key == bodyIndex then
226  result = true
227  setVisibility(body.headNode, true)
228  setVisibility(body.armsNode, true)
229  self.selectedBodyIndex = key
230  else
231  setVisibility(body.headNode, false)
232  setVisibility(body.armsNode, false)
233  end
234  end
235  else
236  g_logManager:devError("-- [PlayerStyle:setBody] Player not set.
     Missing call to PlayerStyle:loadXML().")
237  end
238  return result
239  end
```

removeHat

Description

Definition

removeHat()

Code

```
243  function PlayerStyle:removeHat()
244  local result = false
245
246  if self.player ~= nil and self.hatNode ~= nil then
247  unlink(self.hatNode)
248  delete(self.hatNode)
249  self.hatNode = nil
250  self.selectedAccessoryIndex = 0
251  result = true
252  end
253  return result
254  end
```

removeAccessory

Description

Definition

removeAccessory()

Code

```
258  function PlayerStyle:removeAccessory()
```

```
259  local result = false
260
261  if self.player ~= nil and self.accessoryNode ~= nil then
262  unlink(self.accessoryNode)
263  delete(self.accessoryNode)
264  self.accessoryNode = nil
265  self.selectedAccessoryIndex = 0
266  result = true
267  end
268  return result
269  end
```

setHair

Description

Definition

setHair()

Code

```
273  function PlayerStyle:setHair(hairIndex, isHatHair)
274  local result = false
275
276  if self.player ~= nil then
277  if isHatHair ~= nil and isHatHair then
278  if self.selectedHairIndex > 0 then
279  local currentHairStyleNode =
     self.hairs.styles[self.selectedHairIndex].node
280  setVisibility(currentHairStyleNode, false)
281  setVisibility(self.hairs.hatStyleNode, true)
282  end
283  result = true
284  else
285  for key, hairStyle in ipairs(self.hairs.styles) do
286  if key == hairIndex then
287  if self.selectedHatIndex == 0 then
288  setVisibility(hairStyle.node, true)
289  end
290  self.selectedHairIndex = key
291  result = true
292  else
293  setVisibility(hairStyle.node, false)
294  end
295  end
296  end
```

```
297 else
298 g_logManager:devError("-- [PlayerStyle:setHair] Player not set.
    Missing call to PlayerStyle:loadXML().")
299 end
300 return result
301 end
```

setHat

Description

Definition

setHat()

Code

```
305 function PlayerStyle:setHat(hatIndex)
306 local result = false
307
308 if self.player ~= nil and self.playerHatNode ~= nil then
309 if hatIndex == 0 and self.selectedHatIndex > 0 then
310 result = self:removeHat()
311 self:setHair(self.selectedHairIndex)
312 elseif hatIndex ~= self.selectedHatIndex or self.hatNode == nil
    then
313 self:removeHat()
314 local xmlFilename = Utils.getFilename(self.hatReferenceFilename)
315
316 if xmlFilename ~= nil and xmlFilename ~= "" then
317 local xmlFile = loadXMLFile("hatXML", xmlFilename)
318
319 if xmlFile ~= nil and xmlFile ~= 0 then
320 local hatKey =
    string.format("playerClothing.hats.hat(%d)#filename", hatIndex -
    1)
321 if hasXMLProperty(xmlFile, hatKey) then
322 local i3dFilename = getXMLString(xmlFile, hatKey)
323 self.hatNode = loadI3DFile(i3dFilename, false, false, false)
324
325 if self.hatNode ~= nil then
326 self:setHair(0, true)
327 link(self.playerHatNode, self.hatNode)
328 self.selectedHatIndex = hatIndex
329 result = true
330 end
331 end
332 delete(xmlFile)
```

```
333 end
334 end
335 end
336 else
337 g_logManager:devError("-- [PlayerStyle:setHat] Player not set.
    Missing call to PlayerStyle:loadXML().")
338 end
339 return result
340 end
```

setJacket

Description

Definition

setJacket()

Code

```
344 function PlayerStyle:setJacket(jacketIndex)
345 local result = false
346
347 if self.player ~= nil then
348 for key, jacket in ipairs(self.jackets) do
349 if key == jacketIndex then
350 result = true
351 setVisibility(jacket.node, true)
352 self.selectedJacketIndex = key
353 else
354 setVisibility(jacket.node, false)
355 end
356 end
357 else
358 g_logManager:devError("-- [PlayerStyle:setJacket] Player not set.
    Missing call to PlayerStyle:loadXML().")
359 end
360 return result
361 end
```

setAccessory

Description

Definition

setAccessory()

Code

```
365 function PlayerStyle:setAccessory(accessoryIndex)
366 local result = false
367
368 if self.player ~= nil and self.playerAccessoryNode ~= nil then
```

```
369  if accessoryIndex == 0 and self.selectedAccessoryIndex > 0 then
370  result = self:removeHat()
371  self:setHair(self.selectedHairIndex)
372  elseif accessoryIndex ~= self.selectedAccessoryIndex or
     self.accessoryNode == nil then
373  self:removeAccessory()
374  local xmlFilename =
     Utils.getFilename(self.accessoryReferenceFilename)
375
376  if xmlFilename ~= nil and xmlFilename ~= "" then
377  local xmlFile = loadXMLFile("accessoryXML", xmlFilename)
378
379  if xmlFile ~= nil and xmlFile ~= 0 then
380  local accessoryKey =
     string.format("playerClothing.accessories.accessory(%d)#filename",
     accessoryIndex - 1)
381  if hasXMLProperty(xmlFile, accessoryKey) then
382  local i3dFilename = getXMLString(xmlFile, accessoryKey)
383  self.accessoryNode = loadI3DFile(i3dFilename, false, false, false)
384
385  if self.accessoryNode ~= nil then
386  link(self.playerAccessoryNode, self.accessoryNode)
387  self.selectedAccessoryIndex = accessoryIndex
388  result = true
389  end
390  end
391  delete(xmlFile)
392  end
393  end
394  end
395  else
396  g_logManager:devError("-- [PlayerStyle:setAccessory] Player not
     set. Missing call to PlayerStyle:loadXML().")
397  end
398  return result
399  end
```

setProtectiveVisibility

Description

Definition

setProtectiveVisibility()

Code

```
403  function PlayerStyle:setProtectiveVisibility(state)
```

```
404  self.protection.isVisible = state
405
406  if self.player ~= nil then
407  if self.protection.helmetNode ~= nil then
408  setVisibility(self.protection.helmetNode, state)
409  end
410  if self.protection.glovesNode ~= nil then
411  setVisibility(self.protection.glovesNode, state)
412  end
413  self:updateHeadWearVisibility()
414  else
415  g_logManager:devError("-- [PlayerStyle:setProtectiveVisibility]
     Player not set. Missing call to PlayerStyle:loadXML().")
416  end
417  end
```

setProtectiveUV

Description

Definition

setProtectiveUV()

Code

```
421  function PlayerStyle:setProtectiveUV(uvs)
422  if self.protection.helmetNode ~= nil then
423  setShaderParameter(self.protection.helmetNode, "offsetUV",
     uvs[1], uvs[2], 0, 0, false)
424  end
425  if self.protection.glovesNode ~= nil then
426  setShaderParameter(self.protection.glovesNode, "offsetUV",
     uvs[1], uvs[2], 0, 0, false)
427  end
428  end
```

getProtectiveVisibility

Description

Definition

getProtectiveVisibility()

Code

```
432  function PlayerStyle:getProtectiveVisibility()
433  return self.protection.isVisible
434  end
```

updateHeadWearVisibility

Description

Definition

updateHeadWearVisibility()

Code

```lua
function PlayerStyle:updateHeadWearVisibility()
local isThirdPerson = getVisibility(self.player.meshThirdPerson)

if isThirdPerson then
local protectiveGearVisibility = self:getProtectiveVisibility()

if self.selectedHatIndex > 0 then
setVisibility(self.hatNode, not protectiveGearVisibility and isThirdPerson)
elseif self.selectedHairIndex > 0 then
local currentHairStyleNode = self.hairs.styles[self.selectedHairIndex].node
setVisibility(currentHairStyleNode, not protectiveGearVisibility)
setVisibility(self.hairs.hatStyleNode, protectiveGearVisibility)
end
end
end
```

setVisibility

Description

Definition

setVisibility()

Code

```lua
function PlayerStyle:setVisibility(state)
if self.player ~= nil then
if self.selectedBodyIndex > 0 then
local body = self.bodies[self.selectedBodyIndex]
setVisibility(body.headNode, state)
setVisibility(body.armsNode, state)
end
if self.hatNode ~= nil then
setVisibility(self.hatNode, state)
if self.selectedHairIndex > 0 then
setVisibility(self.hairs.hatStyleNode, state)
end
else
if self.selectedHairIndex > 0 then
local currentHairStyleNode = self.hairs.styles[self.selectedHairIndex].node
setVisibility(currentHairStyleNode, state)
end
end
```

```
474  if self.accessoryNode ~= nil then
475  setVisibility(self.accessoryNode, state)
476  end
477  if self.selectedJacketIndex > 0 then
478  local jacket = self.jackets[self.selectedJacketIndex]
479  setVisibility(jacket.node, state)
480  end
481  else
482  g_logManager:devError("-- [PlayerStyle:setVisibility] Player not
     set. Missing call to PlayerStyle:loadXML().")
483  end
484  end
```

PlayerSwitchedFarmEvent

Description

emptyNew

Description

Create an empty instance

Definition

emptyNew()

Return Values

table instance Instance of object

Code

```
15  function PlayerSwitchedFarmEvent:emptyNew()
16  local self = Event:new(PlayerSwitchedFarmEvent_mt)
17  return self
18  end
```

writeStream

Description

Writes network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
34  function PlayerSwitchedFarmEvent:writeStream(streamId, connection)
35  streamWriteUIntN(streamId, self.farmId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
36  streamWriteUIntN(streamId, self.oldFarmId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
37  NetworkUtil.writeNodeObjectId(streamId, self.userId)
38  end
```

readStream

Description

Reads network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
44  function PlayerSwitchedFarmEvent:readStream(streamId, connection)
45  self.farmId = streamReadUIntN(streamId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
46  self.oldFarmId = streamReadUIntN(streamId,
    FarmManager.FARM_ID_SEND_NUM_BITS)
47  self.userId = NetworkUtil.readNodeObjectId(streamId)
48
49  self:run(connection)
50  end
```

run

Description

Run event

Definition

run(table connection)

Arguments

table connection connection information

Code

```
55  function PlayerSwitchedFarmEvent:run(connection)
56  if connection:getIsServer() then -- on client
57  if self.oldFarmId ~= FarmManager.INVALID_FARM_ID then -- joined
    server
58  g_farmManager:getFarmById(self.oldFarmId):removeUser(self.userId)
59  end
60
61  if self.farmId ~= FarmManager.INVALID_FARM_ID then -- left server
62  g_farmManager:getFarmById(self.farmId):addUser(self.userId)
63  end
64
65  g_messageCenter:publish(MessageType.PLAYER_FARM_CHANGED,
    {self.player})
66  else -- on server, notify all clients (incl. self) of player farm
    switch
67  g_server:broadcastEvent(PlayerSwitchedFarmEvent:new(self.oldFarmId,
    self.farmId, self.userId), true)
68  end
```

```
69  end
```

PlayerTeleportEvent
Description
emptyNew
Description

Create an empty instance

Definition

emptyNew()

Return Values

table instance Instance of object

Code

```
11  function PlayerTeleportEvent:emptyNew()
12  local self = Event:new(PlayerTeleportEvent_mt)
13  return self
14  end
```

new
Description

Create an instance

Definition

new(float x, float y, float z, bool isAbsolute, bool isRootNode, float z)

Arguments

float x	world x position
float y	world y position
float z	world z position
bool isAbsolute	if not true, y is a delta from the terrain
bool isRootNode	if true, y is the root node location, otherwise y is the feet location
float z	world z position

Return Values

table instance Instance of object

Code

```
25  function PlayerTeleportEvent:new(x, y, z, isAbsolute, isRootNode)
26  local self = PlayerTeleportEvent:emptyNew()
27  self.x = x
28  self.y = y
29  self.z = z
30  self.isAbsolute = isAbsolute
31  self.isRootNode = isRootNode
32  return self
33  end
```

newExitVehicle
Description

Create an instance when player exits vehicle

Definition

newExitVehicle(table exitVehicle)

Arguments

table exitVehicle instance of the vehicle that the player exits

Return Values

table instance Instance of object

Code

```
39  function PlayerTeleportEvent:newExitVehicle(exitVehicle)
40  local self = PlayerTeleportEvent:emptyNew()
41  self.exitVehicle = exitVehicle
42  return self
43  end
```

readStream

Description

Reads network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
49  function PlayerTeleportEvent:readStream(streamId, connection)
50  if streamReadBool(streamId) then
51  self.exitVehicle = NetworkUtil.readNodeObject(streamId)
52  else
53  self.x = streamReadFloat32(streamId)
54  self.y = streamReadFloat32(streamId)
55  self.z = streamReadFloat32(streamId)
56  self.isAbsolute = streamReadBool(streamId)
57  self.isRootNode = streamReadBool(streamId)
58  end
59  self:run(connection)
60  end
```

writeStream

Description

Writes network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
66  function PlayerTeleportEvent:writeStream(streamId, connection)
```

```
67  if streamWriteBool(streamId, self.exitVehicle ~= nil) then
68  NetworkUtil.writeNodeObject(streamId, self.exitVehicle)
69  else
70  streamWriteFloat32(streamId, self.x)
71  streamWriteFloat32(streamId, self.y)
72  streamWriteFloat32(streamId, self.z)
73  streamWriteBool(streamId, self.isAbsolute)
74  streamWriteBool(streamId, self.isRootNode)
75  end
76  end
```

run

Description

Run event

Definition

run(table connection)

Arguments

table connection connection information

Code

```
81  function PlayerTeleportEvent:run(connection)
82  if not connection:getIsServer() then
83  local player = g_currentMission.connectionsToPlayer[connection]
84  if player ~= nil then
85  if self.exitVehicle ~= nil then
86  player:moveToExitPoint(self.exitVehicle)
87  elseif self.x ~= nil then
88  player:moveTo(self.x,self.y,self.z,self.isAbsolute,self.isRootNode)
89  end
90  end
91  end
92  end
```

PlayerThrowObjectEvent

Description

emptyNew

Description

Create an empty instance

Definition

emptyNew()

Return Values

table instance Instance of object

Code

```
11  function PlayerThrowObjectEvent:emptyNew()
12  local self = Event:new(PlayerThrowObjectEvent_mt)
```

```
13  return self
14  end
```

new

Description

Create an instance

Definition

new(table player)

Arguments

table player player instance

Return Values

table instance Instance of object

Code

```
20  function PlayerThrowObjectEvent:new(player)
21  self.player = player
22  return self
23  end
```

readStream

Description

Reads network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
29  function PlayerThrowObjectEvent:readStream(streamId, connection)
30  self.player = NetworkUtil.readNodeObject(streamId)
31  self:run(connection)
32  end
```

writeStream

Description

Writes network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
38  function PlayerThrowObjectEvent:writeStream(streamId, connection)
39  NetworkUtil.writeNodeObject(streamId, self.player)
40  end
```

run

Description

Run event

Definition

run(table connection)

Arguments

table connection connection information

Code

```
45  function PlayerThrowObjectEvent:run(connection)
46  if not connection:getIsServer() then
47  g_server:broadcastEvent(self, false, connection, self.player)
48  end
49
50  self.player:throwObject(true)
51  end
```

sendEvent

Description

Definition

sendEvent()

Code

```
55  function PlayerThrowObjectEvent.sendEvent(player, noEventSend)
56  if noEventSend == nil or noEventSend == false then
57  if g_server ~= nil then
58  g_server:broadcastEvent(PlayerThrowObjectEvent:new(player), nil, nil,
    player)
59  else
60  g_client:getServerConnection():sendEvent(PlayerThrowObjectEvent:new(playe
61  end
62  end
63  end
```

PlayerToggleLightEvent

Description

emptyNew

Description

Create an empty instance

Definition

emptyNew()

Return Values

table instance Instance of object

Code

```
11  function PlayerToggleLightEvent:emptyNew()
12  local self = Event:new(PlayerToggleLightEvent_mt)
13  return self
14  end
```

new

Description

Create an instance

Definition

new(table player)

Arguments

table player player instance

Return Values

table instance Instance of object

Code

```
20  function PlayerToggleLightEvent:new(player, isActive)
21  self.player = player
22  self.isActive = isActive
23  return self
24  end
```

readStream

Description

Reads network stream

Definition

readStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
30  function PlayerToggleLightEvent:readStream(streamId, connection)
31  self.player = NetworkUtil.readNodeObject(streamId)
32  self.isActive = streamReadBool(streamId)
33  self:run(connection)
34  end
```

writeStream

Description

Writes network stream

Definition

writeStream(integer streamId, table connection)

Arguments

integer streamId network stream identification

table connection connection information

Code

```
40  function PlayerToggleLightEvent:writeStream(streamId, connection)
41  NetworkUtil.writeNodeObject(streamId, self.player)
42  streamWriteBool(streamId, self.isActive)
43  end
```

run

Description

Run event

Definition

run(table connection)

Arguments

table connection connection information

Code

```
48  function PlayerToggleLightEvent:run(connection)
49  if not connection:getIsServer() then
50  g_server:broadcastEvent(self, false, connection, self.player)
51  end
52
53  self.player:setLightIsActive(self.isActive, true)
54  end
```

sendEvent

Description

Definition

sendEvent()

Code

```
58  function PlayerToggleLightEvent.sendEvent(player, active, noEventSend)
59  if noEventSend == nil or noEventSend == false then
60  if g_server ~= nil then
61  g_server:broadcastEvent(PlayerToggleLightEvent:new(player, active), nil,
    nil, player)
62  else
63  g_client:getServerConnection():sendEvent(PlayerToggleLightEvent:new(playe
    active))
64  end
65  end
66  end
```

AmbientSoundManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
25  function AmbientSoundManager:new(customMt)
26  local self = AbstractManager:new(customMt or
    AmbientSoundManager_mt)
```

```
27
28  return self
29  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
33  function AmbientSoundManager:initDataStructures()
34  self.ambient3DSounds = {}
35  self.nodeToSoundNode = {}
36  self.nodeToPolyChain = {}
37  self.polyChains = {}
38  self.rootNode = nil
39  self.lastGrid = nil
40  self.ambientXmlFilename = nil
41  self.ambient3DFilename = nil
42  self.initialized = false
43  self.numOfAmbient3DSounds = 0
44  self.blocksPerRowColumn = 8
45  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
50  function AmbientSoundManager:loadMapData(xmlFile, missionInfo)
51  AmbientSoundManager:superClass().loadMapData(self)
52
53  local xmlFilename = Utils.getFilename(getXMLString(xmlFile,
    "map.sounds#filename"), g_currentMission.baseDirectory)
54  if xmlFile == nil then
55  return false
56  end
57
58  local soundXmlFile = loadXMLFile("ambientSoundsXML", xmlFilename)
59  if soundXmlFile == nil or soundXmlFile == 0 then
```

```
60  g_logManager:xmlWarning(xmlFilename, "Warning: AmbientSounds
    could not load xmlFile!")
61  return false
62  end
63
64  self.initialized = true
65  self.mapSoundGrid = MapDataGrid:new(g_currentMission.mapWidth,
    self.blocksPerRowColumn)
66
67  -- initialize grid
68  for i=1, self.blocksPerRowColumn do
69  for j=1, self.blocksPerRowColumn do
70  self.mapSoundGrid:setValue(i, j, {})
71  end
72  end
73
74  self.ambientXmlFilename = xmlFilename
75  local filename = getXMLString(soundXmlFile,
    "sound.ambient3d#filename")
76  if filename ~= nil then
77  local ambient3DFilename = Utils.getFilename(filename,
    g_currentMission.baseDirectory)
78  self.ambient3DFilename = ambient3DFilename
79  end
80
81  self:loadAmbientSounds()
82
83  self.indoorVolumeFactor = getXMLFloat(soundXmlFile,
    "sound.ambient3d#indoorVolumeFactor") or 1
84  self.indoorLowpassGainFactor = getXMLFloat(soundXmlFile,
    "sound.ambient3d#indoorLowpassGainFactor") or 1
85
86  -- log("---> ", #self.splines)
87  -- for _, spline in ipairs(self.splines) do
88  -- for i=0, getSplineLength(spline), 5 do
89  -- local t = i / getSplineLength(spline)
90  -- local x,y,z = getSplinePosition(spline, t)
91  -- log(x, y, z, t, getNumOfChildren(spline))
92
93  -- local value, rowIndex, colIndex =
    self.mapSoundGrid:getValueAtWorldPos(x, z)
94  -- if value == nil then
```

```
95  -- value = {}
96  -- self.mapSoundGrid:setValue(rowIndex, colIndex, value)
97  -- end
98  -- value[spline] = {spline=spline}
99
100 -- for j=0, getNumOfChildren(spline)-1 do
101 -- log(getName(getChildAt(spline, j)))
102 -- local soundNode = self.nodeToSoundNode[getChildAt(spline, j)]
103 -- if soundNode ~= nil then
104 -- soundNode:setWorldPosition(x,y,z)
105 -- self:addSoundNodeToGrid(soundNode)
106 -- end
107 -- end
108 -- end
109 -- end
110
111 if g_addCheatCommands then
112 addConsoleCommand("gsReloadAmbientSounds", "Reload ambient sound
    system", "consoleCommandReloadAmbientSound", self)
113 addConsoleCommand("gsToggleAmbientSoundsDebug", "Toggles ambient
    sound system debugging",
    "consoleCommandToggleAmbientSoundsDebug", self)
114 end
115
116 delete(soundXmlFile)
117
118 return true
119 end
```

unloadMapData

Description

Unload data on mission delete

Definition

unloadMapData()

Code

```
123 function AmbientSoundManager:unloadMapData()
124 AmbientSoundManager:superClass().unloadMapData(self)
125
126 if self.initialized then
127 removeConsoleCommand("gsReloadAmbientSounds")
128 removeConsoleCommand("gsToggleAmbientSoundsDebug")
129 self:deleteAmbientSounds()
```

```
130  self.mapSoundGrid:delete()
131  end
132  end
```

AmbientSoundUtil

Description

onCreateSoundNode

Description

I3D-Callback to add a 3d sound

Definition

onCreateSoundNode(integer id)

Arguments

integer id node id of the 3d sound element

onCreatePolygonChain

Description

I3D-Callback to add a spline

Definition

onCreatePolygonChain(integer id)

Arguments

integer id node id of the 3d sound element

SoundManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
22  function SoundManager:new(customMt)
23  local self = AbstractManager:new(customMt or SoundManager_mt)
24
25  return self
26  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
30  function SoundManager:initDataStructures()
31  self.samples = {}
32  self.orderedSamples = {}
```

```
33  self.activeSamples = {}
34  self.activeSamplesSet = {}
35  self.currentSampleIndex = 1
36  self.oldRandomizationIndex = 1
37  self.isIndoor = false
38  self.isInsideBuilding = false
39
40  self.soundTemplates = {}
41  self.soundTemplateXMLFile = nil
42  self:loadSoundTemplates(SoundManager.DEFAULT_SOUND_TEMPLATES)
43
44  self.modifierTypeNameToIndex = {}
45  self.modifierTypeIndexToDesc = {}
46  SoundModifierType = self.modifierTypeNameToIndex
47
48  self.indoorStateChangedListeners = {}
49  end
```

registerModifierType

Description

Loads sound templates from xml file

Definition

registerModifierType()

Return Values

boolean true if loading was successful else false

Code

```
54  function SoundManager:registerModifierType(typeName, func,
    minFunc, maxFunc)
55  typeName = typeName:upper()
56
57  if SoundModifierType[typeName] == nil then
58  local desc = {}
59  desc.name = typeName
60  desc.index = #self.modifierTypeIndexToDesc + 1
61  desc.func = func
62  desc.minFunc = minFunc
63  desc.maxFunc = maxFunc
64
65  SoundModifierType[typeName] = desc.index
66  table.insert(self.modifierTypeIndexToDesc, desc)
67  end
68
```

```
69  return SoundModifierType[typeName]
70  end
```

loadSoundTemplates

Description

Loads sound templates from xml file

Definition

loadSoundTemplates()

Return Values

boolean true if loading was successful else false

Code

```
75  function SoundManager:loadSoundTemplates(xmlFilename)
76  local xmlFile = loadXMLFile("TempTemplates", xmlFilename)
77  if xmlFile ~= nil then
78  local i = 0
79  while true do
80  local key = string.format("soundTemplates.template(%d)", i)
81  if not hasXMLProperty(xmlFile, key) then
82  break
83  end
84
85  local name = getXMLString(xmlFile, key.."#name")
86  if name ~= nil then
87  if self.soundTemplates[name] == nil then
88  self.soundTemplates[name] = key
89  else
90  print(string.format("Warning: Sound template '%s' already
    exists!", name))
91  end
92  end
93
94  i = i + 1
95  end
96
97  self.soundTemplateXMLFile = xmlFile
98  return true
99  end
100
101 return false
102 end
```

cloneSample

Description

Returns a clone of the sample at the given link node

Definition

cloneSample(table sample, integer linkNode)

Arguments

table	sample	sample object
integer	linkNode	id of new link node

Return Values

table sample sample object

cloneSample2D

Description

Returns a clone of the sample at the given link node

Definition

cloneSample2D(table sample, integer linkNode)

Arguments

table	sample	sample object
integer	linkNode	id of new link node

Return Values

table sample sample object

validateSampleDefinition

Description

Validate a sample definition and parameters.

Definition

validateSampleDefinition(int xmlFile, string baseKey, string sampleName, string baseDir, string audioGroup, bool is2D, table components, table i3dMappings)

Arguments

int	xmlFile	Sample definition XML file handle
string	baseKey	Parent element key of sample
string	sampleName	Sample element name
string	baseDir	Sample file path base directory
string	audioGroup	Sample audio group
bool	is2D	If true, the sample is interpreted as a non-spatial sound sample
table	components	Path components targeting a node to which a spatial audio source is linked
table	i3dMappings	Mappings of I3D indices to path components for node link resolution

Return Values

bool	True	if the definition and parameters are valid
bool	True	if an external XML file is loaded in place of the given parameter, caller must delete the handle afterwards!
int	XML	file handle, either the one passed in as an argument or an alternate external sound file definition which must be released by the caller
string	Sound	definition parent element key which may have changed according to an alternate external sound file definition
int	Link	node target for spatial sound samples

loadSample2DFromXML

Description

Loads a 2D sample from XML.
This creates a sample using no spatial information to be used for global, client-only contexts (e.g. UI).

Definition

loadSample2DFromXML(int xmlFile, string baseKey, string sampleName, string baseDir, int loops, string audioGroup)

Arguments

int	xmlFile	Sample definition XML file handle
string	baseKey	Parent element key of sample
string	sampleName	Sample element name
string	baseDir	Sample file path base directory
int	loops	Loop count of sample, defaults to 1. A value of 0 will loop indefinitely.
string	audioGroup	Sample audio group

loadSampleFromXML

Description

Loads a sample from xml

Definition

loadSampleFromXML(integer xmlFile, string key, string sampleName, string baseDir, table components, integer loops)

Arguments

integer	xmlFile	xml-file handle
string	key	xml element key
string	sampleName	sample name
string	baseDir	base directory
table	components	components
integer	loops	number of loops

Return Values

table sample sample object

loadSampleAttributesFromXML

Description

Loads a sample attributes from xml

Definition

loadSampleAttributesFromXML(table sample, integer xmlFile, string key)

Arguments

table	sample	sample object
integer	xmlFile	xml-file handle
string	key	xml element key

loadModifiersFromXML

Description

Loads a sample modifiers from xml

Definition

loadModifiersFromXML(table sample, integer xmlFile, string key)

Arguments

table	sample	sample object

integer xmlFile xml-file handle
string key xml element key

loadRandomizationsFromXML

Description

Loads a sample modifiers from xml

Definition

loadRandomizationsFromXML(table sample, integer xmlFile, string key)

Arguments

table sample sample object
integer xmlFile xml-file handle
string key xml element key

updateSampleFade

Description

Updates sample fade

Definition

updateSampleFade(table sample, float dt)

Arguments

table sample sample object
float dt time since last call in ms

updateSampleModifiers

Description

Updates sample modifiers

Definition

updateSampleModifiers(table sample)

Arguments

table sample sample object

updateSampleAttributes

Description

Updates sample attributes

Definition

updateSampleAttributes(table sample, boolean force)

Arguments

table sample sample object
boolean force true if indoor / outdoor change should be forced

updateSampleRandomizations

Description

Updates sample modifiers

Definition

updateSampleRandomizations(table sample, table inputs)

Arguments

table sample sample object
table inputs a table of values

getSampleModifierValue

Description

Gets sample modifiers

Definition

getSampleModifierValue(table sample, table inputs, string debugString)

Arguments

table	sample	sample object
table	inputs	a table of values
string	debugString	a string

deleteSample

Description

Deletes the sample

Definition

deleteSample(table sample)

Arguments

table sample sample object

deleteSamples

Description

Deletes table of samples

Definition

deleteSamples(table samples)

Arguments

table samples table with sample objects

playSample

Description

Plays the sample

Definition

playSample(table sample)

Arguments

table sample sample object

playSamples

Description

Plays table of samples

Definition

playSamples(table samples, integer delay, table afterSample)

Arguments

table	samples	table with sample objects
integer	delay	delay in ms
table	afterSample	sample will be played as soon as afterSample has stoppped

stopSample

Description

Stops the sample

Definition

stopSample(table sample, boolean force)

Arguments

table sample sample object

boolean force ignore fade out and stop immediately

stopSamples

Description

Stops table of samples

Definition

stopSamples(table samples)

Arguments

table samples table with sample objects

setSampleVolume

Description

Sets the sample volume

Definition

setSampleVolume(table sample, float volume)

Arguments

table sample sample object

float volume volume

setSamplePitch

Description

Sets the sample pitch

Definition

setSamplePitch(table sample, float pitch)

Arguments

table sample sample object

float pitch pitch

getIsSamplePlaying

Description

Checks if a sample is playing

Definition

getIsSamplePlaying(table sample, float offset)

Arguments

table sample sample object

float offset an offset

Return Values

boolean isPlaying true if sample is playing else false

setIsIndoor

Description

Sets the indoor state

Definition

setIsIndoor(boolean true)

Arguments

boolean true if sound should be played as indoor, else false

getIsIndoor

Description

Checks if indoor mode is active

Definition

getIsIndoor()

Return Values

boolean isIndoor true if indoor mode is active else false

setIsInsideBuilding

Description

Sets the inside building state

Definition

setIsInsideBuilding(boolean true)

Arguments

boolean true if sound should be played as inside building, else false

getIsInsideBuilding

Description

Checks if inside building mode is active

Definition

getIsInsideBuilding()

Return Values

boolean isIndoor true if inside building mode is active else false

getModifierFactor

Description

Gets modifier factor

Definition

getModifierFactor(table sample, string modifierName)

Arguments

table sample sample object

string modifierName the modifier name

Return Values

float factor the modifier factor

SoundMixer

Description

new

Description

Creating sound node

Definition

new()

Return Values

table instance instance of object

integer group audio group

Code

```
function SoundMixer:new(customMt)

local self = {}
setmetatable(self, customMt or SoundMixer_mt)

g_messageCenter:subscribe(MessageType.GAME_STATE_CHANGED, self.onGameStateChanged, self)

local xmlFilename = "dataS/soundMixer.xml"
local xmlFile = loadXMLFile("soundMixerXML", xmlFilename)

self.volumeFactors = {}
for _, groupIndex in pairs(AudioGroup) do
self.volumeFactors[groupIndex] = 1
end

self.masterVolume = 1

self.gameStates = {}

if xmlFile ~= nil and xmlFile ~= 0 then
local i = 0
while true do
local key = string.format("soundMixer.gameState(%d)", i)
if not hasXMLProperty(xmlFile, key) then
break
end

local gameStateName = getXMLString(xmlFile, key.."#name")
local gameStateIndex = g_gameStateManager:getGameStateIndexByName(gameStateName)
if gameStateIndex ~= nil then
local gameState = {}
gameState.audioGroups = {}

local j = 0
while true do
local audioGroupKey = string.format("%s.audioGroup(%d)", key, j)
if not hasXMLProperty(xmlFile, audioGroupKey) then
```

```
55  break
56  end
57
58  local name = getXMLString(xmlFile, audioGroupKey.."#name")
59  local volume = getXMLFloat(xmlFile, audioGroupKey.."#volume") or
    1.0
60  local audioGroupIndex = AudioGroup.getAudioGroupIndexByName(name)
61
62  if audioGroupIndex ~= nil then
63  local group = {}
64  group.index = audioGroupIndex
65  group.volume = volume
66  gameState.audioGroups[audioGroupIndex] = group
67  else
68  print(string.format("Warning: Audio-Group '%s' is not defined for
    audio-group '%s'!", tostring(name), key))
69  end
70
71  j = j + 1
72  end
73
74  self.gameStates[gameStateIndex] = gameState
75  else
76  print(string.format("Warning: Game-State '%s' is not defined for
    state '%s'!", tostring(gameStateName), key))
77  end
78
79  i = i + 1
80  end
81  else
82  print("Error: SoundMixer could not load configuration file!")
83  end
84
85  delete(xmlFile)
86
87  self.volumes = {}
88  local gameState = self.gameStates[GameState.LOADING]
89  for _, groupIndex in ipairs(AudioGroup.groups) do
90  local volume = gameState.audioGroups[groupIndex].volume or 1
91  self.volumes[groupIndex] = {volume = volume, listeners = {}}
92  setAudioGroupVolume(groupIndex, volume)
```

```
93  end
94
95  return self
96  end
```

SoundNode
Description
new
Description

Creating sound node

Definition

new()

Return Values

table	instance instance of object
integer group	audio group

Code

```
19  function SoundNode:new(soundNode, group, customMt)
20
21  local self = {}
22  setmetatable(self, customMt or SoundNode_mt)
23
24  if not getHasClassId(soundNode, ClassIds.AUDIO_SOURCE) then
25  g_logManager:warning("SoundNode '" ..
    tostring(getName(soundNode)) .. "' is not an AUDIO_SOURCE!")
26  return nil
27  end
28
29  self.soundNode = soundNode
30  self.nodes = {}
31  self.currentNode = nil
32  self.nextNode = nil
33  self.outerRange = getAudioSourceRange(soundNode)
34  self.innerRange = getAudioSourceInnerRange(soundNode)
35  self.nextPlayTime = g_time
36  self.nextCheckTime = nil
37
38  local function addSoundSource(node, group)
39  local sample = getAudioSourceSample(node)
40  local duration = getSampleDuration(sample)
41  setSampleGroup(sample, group)
42  local volume = getSampleVolume(sample)
43  setAudioSourceAutoPlay(node, false)
```

```
44  stopSample(sample, 0, 0)
45  table.insert(self.nodes, {index=#self.nodes+1, node=node,
    sample=sample, volume=volume, duration=duration})
46  end
47
48  addSoundSource(soundNode, group)
49
50  for i=getNumOfChildren(soundNode)-1, 0, -1 do
51  local child = getChildAt(soundNode, i)
52  if getHasClassId(child, ClassIds.AUDIO_SOURCE) then
53  setAudioSourceRange(child, self.outerRange)
54  setAudioSourceInnerRange(child, self.innerRange)
55  addSoundSource(child, group)
56  end
57  end
58
59  self.retriggerMinDelay =
    Utils.getNoNil(getUserAttribute(soundNode, "retriggerMinDelay"),
    0) * 1000
60  self.retriggerMaxDelay =
    Utils.getNoNil(getUserAttribute(soundNode, "retriggerMaxDelay"),
    0) * 1000
61
62  -- if retriggerMinDelay = 0 we need to make sure that there are
    at least 2 sounds to calculate offsets and seamless playing
63  if self.retriggerMinDelay == 0 then
64  if #self.nodes == 1 then
65  -- copy sound
66  local node = self.nodes[1]
67  local newNode = clone(node.node, true, false, false)
68  setName(newNode, getName(newNode).."_copy")
69  addSoundSource(newNode, group)
70  end
71  end
72
73  local tx,ty,tz = getTranslation(soundNode)
74  local rx,ry,rz = getRotation(soundNode)
75
76  self.parent =
    createTransformGroup("soundGroup_"..getName(soundNode))
77  link(getParent(soundNode), self.parent, getChildIndex(soundNode))
78  setTranslation(self.parent, tx, ty, tz)
```

```
79  setRotation(self.parent, rx, ry, rz)
80
81  for _, soundNode in ipairs(self.nodes) do
82  link(self.parent, soundNode.node)
83  setTranslation(soundNode.node, 0, 0, 0)
84  setRotation(soundNode.node, 0, 0, 0)
85  end
86
87  self.playByDay = Utils.getNoNil(getUserAttribute(soundNode,
    "playByDay"), false)
88  self.playByNight = Utils.getNoNil(getUserAttribute(soundNode,
    "playByNight"), false)
89  if not self.playByDay and not self.playByNight then
90  g_logManager:warning("Ambient 3D sound '%s' has invalid time
    state. At least one of the states 'playByDay' or 'playByNight'
    need to be 'true'", getName(soundNode))
91  end
92  self.playDuringHail = Utils.getNoNil(getUserAttribute(soundNode,
    "playDuringHail"), false)
93  self.playDuringRain = Utils.getNoNil(getUserAttribute(soundNode,
    "playDuringRain"), false)
94  self.playDuringSun = Utils.getNoNil(getUserAttribute(soundNode,
    "playDuringSun"), false)
95  if not self.playDuringHail and not self.playDuringRain and not
    self.playDuringSun then
96  g_logManager:warning("Ambient 3D sound '%s' has invalid weather
    state. At least one of the states 'playDuringHail',
    'playDuringRain' or 'playDuringSun' need to be 'true'",
    getName(soundNode))
97  end
98  self.playInsideBuilding =
    Utils.getNoNil(getUserAttribute(soundNode, "playInsideBuilding"),
    true)
99
100 self.playExterior = Utils.getNoNil(getUserAttribute(soundNode,
    "playExterior"), false)
101 self.playInterior = Utils.getNoNil(getUserAttribute(soundNode,
    "playInterior"), false)
102 if not self.playExterior and not self.playInterior then
103 g_logManager:warning("Ambient 3D sound '%s' has invalid position
    state. At least one of the states 'playExterior' or
    'playInterior' need to be 'true'", getName(soundNode))
104 end
105
```

```
106 self.playHourStart = MathUtil.clamp(Utils.getNoNil(getUserAttribute(soundNode, "playHourStart"), 0), 0, 24) * 1000 * 60 * 60
107 self.playHourEnd = MathUtil.clamp(Utils.getNoNil(getUserAttribute(soundNode, "playHourEnd"), 24), 0, 24) * 1000 * 60 * 60
108 self.playHourInverted = Utils.getNoNil(getUserAttribute(soundNode, "playHourInverted"), false)
109 self.playHour = Utils.getNoNil(getUserAttribute(soundNode, "playHour"), false)
110
111 self.autoStop = Utils.getNoNil(getUserAttribute(soundNode, "autoStop"), true)
112 self.isLooping = Utils.getNoNil(getUserAttribute(soundNode, "isLooping"), false)
113 self.fadeOutDuration = Utils.getNoNil(getUserAttribute(soundNode, "fadeOutDuration"), 0) * 1000
114
115 return self
116 end
```

RailroadCaller

Description

new

Description

Creating sound node

Definition

new()

Return Values

table	instance instance of object
integer group	audio group

Code

```
19 function RailroadCaller:new(isServer, isClient, trainSystem, nodeId, customMt)
20
21 local self = {}
22 setmetatable(self, customMt or RailroadCaller_mt)
23
24 self.trainSystem = trainSystem
25 self.nodeId = nodeId
26 self.isServer = isServer
27 self.isClient = isClient
28
29 return self
```

```
30  end
```

getIsActivatable

Description

Returns true if shop can be opened

Definition

getIsActivatable()

Return Values

boolean isActivateable is activateable

Code

```
84  function RailroadCaller:getIsActivatable()
85  return g_currentMission.controlPlayer
86  end
```

onActivateObject

Description

Called on activate object

Definition

onActivateObject()

Code

```
94  function RailroadCaller:onActivateObject()
95  g_currentMission:addActivatableObject(self)
96  self.objectActivated = true
97  self:callRailroad()
98  end
```

RailroadCrossing

Description

new

Description

Creating sound node

Definition

new()

Return Values

table instance instance of object

integer group audio group

Code

```
19  function RailroadCrossing:new(isServer, isClient, trainSystem,
    nodeId, customMt)
20
21  local self = {}
22  setmetatable(self, customMt or RailroadCrossing_mt)
23
24  self.trainSystem = trainSystem
25  self.nodeId = nodeId
```

```
26  self.isServer = isServer
27  self.isClient = isClient
28
29  return self
30  end
```

Tutorial
Description

getMessages

Description

Get a list of all messages. Also has completion info

Definition

getMessages()

scripts

AchievementManager
Description

new

Description

Creating manager

Definition

new(table customMt, table messageCenter)

Arguments

table customMt	Sub-class meta table
table messageCenter	MessageCenter reference

Return Values

table instance instance of object

Code

```
18  function AchievementManager:new(customMt, messageCenter)
19  local self = AbstractManager:new(customMt or
    AchievementManager_mt)
20
21  self.messageCenter = messageCenter
22
23  return self
24  end
```

initDataStructures

Description

Initialize data structures

Definition

initDataStructures()

Code

```
28  function AchievementManager:initDataStructures()
29  self.achievementList = {}
30  self.achievementListById = {}
```

```
31  self.achievementListByName = {}
32  self.achievementPlates = nil
33  self.numberOfAchievements = 0
34  self.numberOfUnlockedAchievements = 0
35  self.achievementsValid = false
36  self.achievementTimer = 0
37  self.achievementTimeInterval = 1500
38  end
```

load

Description

Loads initial manager

Definition

load()

Return Values

boolean true if loading was successful else false

Code

```
43  function AchievementManager:load()
44
45  local usePlatinum = GS_PLATFORM_TYPE == GS_PLATFORM_TYPE_PS4
46  local xmlFile = loadXMLFile("achievementsXML",
    "dataS/achievements.xml")
47  local xmlFileContent = saveXMLFileToMemory(xmlFile)
48  initAchievements(xmlFileContent)
49  local i = 0
50
51  self.numberOfAchievements = 0
52
53  while true do
54  local key = string.format("achievements.achievement(%d)", i)
55  if not hasXMLProperty(xmlFile, key) then
56  break
57  end
58
59  local id = getXMLString(xmlFile, key.."#id")
60  local idName = getXMLString(xmlFile, key.."#idName")
61  local score = getXMLInt(xmlFile, key.."#score")
62  local targetScore = getXMLInt(xmlFile, key.."#targetScore")
63  local showScore = getXMLBool(xmlFile, key.."#showScore")
64  local imageFilename = getXMLString(xmlFile, key.."#imageFilename")
65  local imageSize = GuiUtils.get2DArray(getXMLString(xmlFile,
    key.."#imageSize"), {1024,1024})
```

```
66  local imageUVs = GuiUtils.getUVs(Utils.getNoNil(getXMLString(xmlFile, key.."#imageUVs"), "0 0 1 1"), imageSize)
67  local psnType = Utils.getNoNil(getXMLString(xmlFile, key.."#psn_type"), "")
68
69  if id ~= nil and idName ~= nil and (psnType ~= "P" or usePlatinum) then
70  local name = g_i18n:getText("achievement_name" .. idName)
71  local description = g_i18n:getText("achievement_desc" .. idName)
72  description = string.gsub(description, "$MEASURING_UNIT", g_i18n:getMeasuringUnit(true))
73  description = string.gsub(description, "$CURRENCY_SYMBOL", g_i18n:getCurrencySymbol(true))
74
75  self:addAchievement(id, idName, name, description, score, targetScore, showScore, imageFilename, imageUVs)
76  end
77
78  i = i + 1
79  end
80
81  delete(xmlFile)
82
83  self:loadAchievementsState()
84
85  return true
86  end
```

loadMapData

Description

Load data on map load

Definition

loadMapData()

Return Values

boolean true if loading was successful else false

Code

```
91  function AchievementManager:loadMapData()
92
93  if g_currentMission.missionInfo.isNewSPCareer then
94  self.startPlayTime = nil
95  self.startMoney = nil
96  self.startFieldJobMissionCount = nil
```

```lua
    self.startCultivatedHectares = nil
    self.startSownHectares = nil
    self.startFertilizedHectares = nil
    self.startThreshedHectares = nil
    self.startCutTreeCount = nil
    self.startBreedCowsCount = nil
    self.startBreedSheepCount = nil
    self.startBreedPigsCount = nil
    self.startBreedChickenCount = nil
else
    local stats = g_currentMission:farmStats()

    self.startPlayTime = math.floor(stats:getTotalValue("playTime") / 60 + 0.0001)

    -- TODO(JK): this does not work for spectator -> farm. Also, hide in MP
    local farm = g_farmManager:getFarmById(0)
    self.startMoney = farm.money

    self.startFieldJobMissionCount = stats:getTotalValue("fieldJobMissionCount")
    self.startCultivatedHectares = stats:getTotalValue("cultivatedHectares")
    self.startSownHectares = stats:getTotalValue("sownHectares")
    self.startFertilizedHectares = stats:getTotalValue("fertilizedHectares")
    self.startThreshedHectares = stats:getTotalValue("threshedHectares")

    self.startCutTreeCount = stats:getTotalValue("cutTreeCount")
    self.startBreedCowsCount = stats:getTotalValue("breedCowsCount")
    self.startBreedSheepCount = stats:getTotalValue("breedSheepCount")
    self.startBreedPigsCount = stats:getTotalValue("breedPigsCount")
    self.startBreedChickenCount = stats:getTotalValue("breedChickenCount")
end

return true
end
```

addAchievement

Description

Adds a new achievement

Definition

addAchievement(string id, string idName, string name, string description, integer score, integer targetScore, boolean showScore, string imageFilename, string imageUVs)

Arguments

string	id	achievement id
string	idName	achievement id name
string	name	achievement name
string	description	achievement description
integer	score	achievement score
integer	targetScore	achievement targetScore
boolean	showScore	achievement showScore
string	imageFilename	achievement imageFilename
string	imageUVs	achievement imageUVs

Return Values

table achievement achievement object

GameStateManager

Description

new

Description

Creating manager

Definition

new()

Return Values

table instance instance of object

Code

```
17  function GameStateManager:new(customMt)
18  local self = {}
19  setmetatable(self, customMt or GameStateManager_mt)
20
21  self.gameStateChangeListeners = {}
22  self.gameState = GameState.STARTING
23
24  return self
25  end
```

I18N

Description

getVolumeUnit

Description

Get the generic unit display text for volumes.

Definition

getVolumeUnit()

Return Values

table instance instance of gas station trigger

getCurrentDate

Description

Get the current date as a string formatted according to the current localization settings.

Definition

getCurrentDate()

Return Values

float delta real delta

formatMinutes

Description

Format a minute time value to a display string as "hh:mm h".

Definition

formatMinutes()

Return Values

boolean isActivateable is activateable

Engine Version: 8.0.0.0

General

drawDebugArrow

Description

Render an arrow. Only use for debug rendering

Definition

drawDebugArrow(float x, float y, float z, float dirX, float dirY, float dirZ, float tangX, float tangY, float tangZ, float r, float g, float b)

Arguments

float x	x position
float y	y position
float z	z position
float dirX	direction x coordinate
float dirY	direction y coordinate
float dirZ	direction z coordinate
float tangX	tangential x direction
float tangY	tangential y direction
float tangZ	tangential z direction
float r	red color component [0, 1]
float g	green color component [0, 1]
float b	blue color component [0, 1]

drawDebugLine

Description

Render a line. Only use for debug rendering

Definition

drawDebugLine(float x0, float y0, float z0, float r0, float g0, float b0, float x1, float y1, float z1, float r1, float g1, float b1)

Arguments

float x0 start x position
float y0 start y position
float z0 start z position
float r0 start red color component [0, 1]
float g0 start green color component [0, 1]
float b0 start blue color component [0, 1]
float x1 end x position
float y1 end y position
float z1 end z position
float r1 end red color component [0, 1]
float g1 end green color component [0, 1]
float b1 end blue color component [0, 1]

drawDebugPoint

Description

Render a point. Only use for debug rendering

Definition

drawDebugPoint(float x, float y, float z, float r, float g, float b, float a)

Arguments

float x x position
float y y position
float z z position
float r red color component [0, 1]
float g green color component [0, 1]
float b blue color component [0, 1]
float a alpha color component [0, 1]

print

Description

Print to console

Definition

print(any type arg1, any type ...)

Arguments

any type arg1 a value
any type ... another value

printCallstack

Description

print callstack

Definition

printCallstack()

source

Description

Source script file

Definition

source(string filename, ref environment)

Arguments

string filename name of script file

ref environment ref to custom environment

Entity

delete

Description

Delete Entity/Object

Definition

delete(integer objectId)

Arguments

integer objectId id of the object

getClassId

Description

Get class name of object

Definition

getClassId(integer objectId)

Arguments

integer objectId id of the object

Return Values

integer classId id of the class

getClassName

Description

Get class name of object

Definition

getClassName(integer objectId)

Arguments

integer objectId id of the object

Return Values

string className class name of object

getHasClassId

Description

Get has class id

Definition

getHasClassId(integer objectId, integer classId)

Arguments

integer objectId id of the object

integer classId id of the class

Return Values

boolean hasClassId has class id

Node

getChild

Description

Get child id

Definition

getChild(integer objectId, string childName)

Arguments

integer objectId id of the object

string childName child name

Return Values

integer childId id of child node

getChildAt

Description

Get child id at given index

Definition

getChildAt(integer objectId, integer index)

Arguments

integer objectId id of the object

integer index index of child

Return Values

integer childId id of child node

getChildIndex

Description

Get child index

Definition

getChildIndex(integer objectId)

Arguments

integer objectId objectid

Return Values

integer childIndex index of child node

getName

Description

Get object name

Definition

getName(integer objectId)

Arguments

integer objectId id of the object

Return Values

string objectName object name

getNumOfChildren

Description

Get number of children

Definition

getNumOfChildren(integer objectId)

Arguments

integer objectId id of the object

Return Values

integer numOfChildren number of children

getParent

Description

Get parent id

Definition

getParent(integer objectId)

Arguments

integer objectId id of the object

Return Values

integer parentId parent id

link

Description

Link node to another node

Definition

link(integer parentNodeId, integer childNodeId, integer index)

Arguments

integer parentNodeId id of parent node
integer childNodeId id of child node
integer index insert index (optional)

setName

Description

Set object name

Definition

setName(integer objectId, string objectName)

Arguments

integer objectId id of the object
string objectName new object name

unlink

Description

Unlink node from parent

Definition

unlink(integer objectId)

Arguments

integer objectId id of the object

Scenegraph

clone

Description

Clone scenegraph object

Definition

clone(integer objectId, boolean groupUnderParent, boolean callOnCreate, boolean addPhysics)

Arguments

integer objectId id of scenegraph object

boolean groupUnderParent	if true the clone will be linked to the the same parent as objectid, otherwise it is not linked to the scenegraph
boolean callOnCreate	call oncreate
boolean addPhysics	add node to physics

Return Values

string cloneId id of clone object

createTransformGroup

Description

Create transform group

Definition

createTransformGroup(string transformName)

Arguments

string transformName name of transform object

Return Values

integer transformId id of transform object

getClipDistance

Description

Get object clip distance

Definition

getClipDistance(integer objectId)

Arguments

integer objectId id of the object

Return Values

float distance clip distance

getEffectiveClipDistance

Description

Get effective object clip distance

Definition

getEffectiveClipDistance(integer objectId)

Arguments

integer objectId id of the object

Return Values

float distance effective clip distance

getEffectiveMinClipDistance

Description

Get effective minimum clip distance

Definition

getEffectiveMinClipDistance(integer objectId)

Arguments

integer objectId id of the object

Return Values

float minDist effective minimum clip distance

getMinClipDistance

Description

Get minimum clip distance

Definition

getMinClipDistance(integer objectId)

Arguments

integer objectId id of the object

Return Values

float minDist minimum clip distance

getObjectMask

Description

Get object mask

Definition

getObjectMask(integer objectId)

Arguments

integer objectId id of the object

Return Values

integer mask the object mask

getQuaternion

Description

gets quaternion

Definition

getQuaternion(integer objectId)

Arguments

integer objectId id of the object

Return Values

float x x value

float y y value

float z z value

float w w value

getRootNode

Description

Get root node

Definition

getRootNode(integer viewportIndex)

Arguments

integer viewportIndex index of viewport

Return Values

integer objectId id of the object

getRotation

Description

Get rotation of a transform object

Definition

getRotation(integer transformId)

Arguments

integer transformId id of transform object

Return Values

float x x value of rotation (radian units)

float y y value of rotation (radian units)

float z z value of rotation (radian units)

getScale

Description

Get scale of a transform object

Definition

getScale(integer transformId)

Arguments

integer transformId id of transform object

Return Values

float x x value of scale

float y y value of scale

float z z value of scale

getTranslation

Description

Get translation of a transform object

Definition

getTranslation(integer transformId)

Arguments

integer transformId id of transform object

Return Values

float x x value of translation

float y y value of translation

float z z value of translation

getUserAttribute

Description

Get user attribute value

Definition

getUserAttribute(integer objectId, string attributeName)

Arguments

integer	objectId	id of the object
string	attributeName	name of the user attribute

Return Values

any type	attributeValue	value of the user attribute, return type is the type of the attribute. returns nil if attributename doest not exist

getVisibility

Description

Get transform object visibility

Definition

getVisibility(integer transformId)

Arguments

integer transformId id of transform object

Return Values

boolean visibility visibility state

getWorldQuaternion

Description

Gets world quaternion

Definition

getWorldQuaternion(integer objectId)

Arguments

integer objectId id of the object

Return Values

float x x value

float y y value

float z z value

float w w value

getWorldRotation

Description

Get world rotation of a transform object

Definition

getWorldRotation(integer transformId)

Arguments

integer transformId id of transform object

Return Values

float x x value of world rotation (radian units)

float y y value of world rotation (radian units)

float z z value of world rotation (radian units)

getWorldTranslation

Description

Get world translation of a transform object

Definition

getWorldTranslation(integer transformId)

Arguments

integer transformId id of transform object

Return Values

float x x value of world translation

float y y value of world translation

float z z value of world translation

localDirectionToLocal

Description

Local space to local space transformation, only direction without translation

Definition

localDirectionToLocal(integer transformId, integer targetTransformId, float x, float y, float z)

Arguments

integer	transformId	id of transform object
integer	targetTransformId	id of target transform object
float	x	x value of local direction
float	y	y value of local direction
float	z	z value of local direction

Return Values

float x x value of local direction
float y y value of local direction
float z z value of local direction

localDirectionToWorld

Description

Local space to world space transformation, only direction without translation

Definition

localDirectionToWorld(integer transformId, float x, float y, float z)

Arguments

integer	transformId	id of transform object
float	x	x value of local position
float	y	y value of local position
float	z	z value of local position

Return Values

float x x value of world direction
float y y value of world direction
float z z value of world direction

localRotationToWorld

Description

Definition

localRotationToWorld()

localToLocal

Description

Local space to local space transformation

Definition

localToLocal(integer transformId, integer targetTransformId, float x, float y, float z)

Arguments

integer	transformId	id of transform object
integer	targetTransformId	id of target transform object
float	x	x value of local direction
float	y	y value of local direction
float	z	z value of local direction

Return Values

float x x value of local direction
float y y value of local direction

float z z value of local direction

localToWorld

Description

Local space to world space transformation

Definition

localToWorld(integer transformId, float x, float y, float z)

Arguments

integer transformId id of transform object

float	x	x value of local position
float	y	y value of local position
float	z	z value of local position

Return Values

float x x value of world position

float y y value of world position

float z z value of world position

project

Description

Transform vector from world space into screen space

Definition

project(float wx, float wy, float wz)

Arguments

float wx world space x coordinate

float wy world space y coordinate

float wz world space z coordinate

Return Values

float sx screen space x coordinate

float sy screen space y coordinate

float sz screen space z coordinate

rotateAboutLocalAxis

Description

Rotate about local axis

Definition

rotateAboutLocalAxis(integer transformId, float rotation, float x, float y, float z)

Arguments

integer transformId id of transform object

float	rotation	angle in radians
float	x	x value of local axis
float	y	y value of local axis
float	z	z value of local axis

setClipDistance

Description

Set object clip distance

Definition

setClipDistance(integer objectId, float distance)

Arguments

integer objectId id of the object

float distance clip distance

setDirection

Description

Set the direction of an object, the positive z-axis points towards the given direction. The y-axis lies in the direction-up-plane.

Definition

setDirection(integer transformId, float x, float y, float z, float upX, float upY, float upZ)

Arguments

integer transformId id of transform object

float	x	x value of direction
float	y	y value of direction
float	z	z value of direction
float	upX	x value of up vector
float	upY	y value of up vector
float	upZ	z value of up vector

setMinClipDistance

Description

Set minimum clip distance

Definition

setMinClipDistance(integer objectId, float minDist)

Arguments

integer objectId id of the object

float minDist minimum clip distance

setObjectMask

Description

Set object mask

Definition

setObjectMask(integer objectId, integer mask)

Arguments

integer objectId id of the object

integer mask the object mask to set

setQuaternion

Description

Sets quaternion

Definition

setQuaternion(integer objectId, float x, float y, float z, float w)

Arguments

integer objectId id of the object

float	x	x value
float	y	y value

float	z	z value
float	w	w value

setRootNode

Description

Set rootnode

Definition

setRootNode(integer objectId, ref)

Arguments

integer objectId id of the object

ref

setRotation

Description

Set rotation

Definition

setRotation(integer objectId, float z, float y, float x)

Arguments

integer objectId id of the object

float	z	z rotation (rad)
float	y	y rotation (rad)
float	x	x rotation (rad)

setScale

Description

Set scale of a transform object

Definition

setScale(integer transformId, float x, float y, float z)

Arguments

integer transformId id of transform object

float	x	x value of scale
float	y	y value of scale
float	z	z value of scale

setTranslation

Description

Set translation of a transform object

Definition

setTranslation(integer transformId, float x, float y, float z)

Arguments

integer transformId id of transform object

float	x	x value of translation
float	y	y value of translation
float	z	z value of translation

setUserAttribute

Description

Set user attribute value

Definition

setUserAttribute(integer objectId, string attributeName, string typeName, any type value)

Arguments

integer	objectId	id of the object
string	attributeName	name of the user attribute
string	typeName	name of the type ("Integer", "Float", "String", "Boolean")
any type	value	value of the user attribute, must match typename

setVisibility

Description

Set transform object visibility

Definition

setVisibility(integer transformId, boolean visibility)

Arguments

integer	transformId	id of transform object
boolean	visibility	visibility state

unProject

Description

Transform vector from screen space into world space

Definition

unProject(float sx, float sy, float sz)

Arguments

float sx	screen space x coordinate
float sy	screen space y coordinate
float sz	screen space z coordinate

Return Values

float wx	world space x coordinate
float wy	world space y coordinate
float wz	world space z coordinate

worldDirectionToLocal

Description

World space to local space transformation, only direction without translation

Definition

worldDirectionToLocal(integer transformId, float x, float y, float z)

Arguments

integer	transformId	id of transform object
float	x	x value of world direction
float	y	y value of world direction
float	z	z value of world direction

Return Values

float x	x value of local direction
float y	y value of local direction
float z	z value of local direction

worldToLocal

Description

World space to local space transformation

Definition

worldToLocal(integer transformId, float x, float y, float z)

Arguments

integer	transformId	id of transform object
float	x	x value of world position
float	y	y value of world position
float	z	z value of world position

Return Values

float	x	x value of local direction
float	y	y value of local direction
float	z	z value of local direction

Lighting

getLightRange

Description

Get range of a light

Definition

getLightRange(integer lightId)

Arguments

integer	lightId	id of light

Return Values

float	range	light range

setLightRange

Description

Set range of a light

Definition

setLightRange(integer lightId, float range)

Arguments

integer	lightId	id of light
float	range	light range

Camera

aimCamera

Description

Aim camera (spring/damper)

Definition

aimCamera(integer cameraId, float x, float y, float z, float distance, float dt, float springStrength)

Arguments

integer	cameraId	id of the camera
float	x	target x coordinate
float	y	target y coordinate
float	z	target z coordinate

float	distance	distance from target
float	dt	delta time
float	springStrength	spring strength

createCamera

Description

Create camera

Definition

createCamera(string cameraName, float fovy, float nearClip, float farClip)

Arguments

string	cameraName	camera name
float	fovy	field of view angle (degree)
float	nearClip	near clip
float	farClip	far clip

Return Values

integer cameraId id of the camera

getCamera

Description

Get camera

Definition

getCamera(integer cameraIndex)

Arguments

integer cameraIndex index of camera

Return Values

integer cameraId id of the camera

getFarClip

Description

Get the far clip distance

Definition

getFarClip(integer cameraId)

Arguments

integer cameraId id of the camera

Return Values

float farClip far clip distance

getFovy

Description

Get the field of view angle

Definition

getFovy(integer cameraId)

Arguments

integer cameraId id of the camera

Return Values

float fovy field of view angle (degree)

getNearClip

Description

Set the near clip distance

Definition

getNearClip(integer cameraId)

Arguments

integer cameraId id of the camera

Return Values

float nearClip near clip distance

setCamera

Description

Set camera

Definition

setCamera(integer cameraId, integer viewportId)

Arguments

integer cameraId id of the camera

integer viewportId id of the viewport

setFarClip

Description

Set the far clip distance

Definition

setFarClip(integer cameraId, float farClip)

Arguments

integer cameraId id of the camera

float farClip far clip distance

setFovy

Description

Set the field of view angle

Definition

setFovy(integer cameraId, float fovy)

Arguments

integer cameraId id of the camera

float fovy field of view angle (degree)

setNearClip

Description

Set the near clip distance

Definition

setNearClip(integer cameraId, float nearClip)

Arguments

integer cameraId id of the camera

float nearClip near clip distance

Shape

getGeometry

Description

Get shape geometry id

Definition

getGeometry(integer shapeId)

Arguments

integer shapeId shape id

Return Values

integer geometryId geometry id

getHasShaderParameter

Description

Get hash shader parameter

Definition

getHasShaderParameter(integer shapeId, string parameterName)

Arguments

integer shapeId shape id

string parameterName the name of the parameter

Return Values

boolean hasParam has parameter

getMaterial

Description

Get material by index

Definition

getMaterial(integer shapeId, integer materialIdx)

Arguments

integer shapeId shape id

integer materialIdx index of attached material

Return Values

integer materialId id of material

getNumMaterials

Description

Get number of materials

Definition

getNumMaterials(integer shapeId)

Arguments

integer shapeId shape id

Return Values

integer numMaterials number of materials

getShaderParameter

Description

Get shader parameter

Definition

getShaderParameter(integer shapeId, string parameterName)

Arguments

integer shapeId shape id

string parameterName the name of the parameter

Return Values

float x x value

float y y value

float z z value

float w w value

getSplitType

Description

Get split type

Definition

getSplitType(integer shapeId)

Arguments

integer shapeId shape id

Return Values

integer splitType split type

setMaterial

Description

Set material by index

Definition

setMaterial(integer shapeId, integer materialId, integer material index)

Arguments

integer shapeId shape id

integer materialId id of material

integer material index index of material

setReflectionMapObjectMask

Description

Set reflection object mask

Definition

setReflectionMapObjectMask(integer shapeId, integer mask, boolean isShared)

Arguments

integer shapeId shape id

integer mask the mask

boolean isShared is shared parameter

setShaderParameter

Description

Set shader parameter

Definition

setShaderParameter(integer shapeId, string parameterName, float x, float y, float z, float w, boolean shared)

Arguments

integer shapeId shape id

string parameterName the name of the parameter

float x x value

float	y	y value
float	z	z value
float	w	w value
boolean	shared	if true, the value is applied to all shapes with the same material

Particle System

addParticleSystemSimulationTime

Description

Add particle system simulation time

Definition

addParticleSystemSimulationTime(integer particleSystemId, float dt)

Arguments

integer	particleSystemId	particle system id
float	dt	value of simulation time to be added

getEmitStartTime

Description

Get emitter starting time.

Definition

getEmitStartTime(integer particleSystemId)

Arguments

integer particleSystemId particle system id

Return Values

float emitStartTime emitter start time

getEmitStopTime

Description

Get emitter stop time.

Definition

getEmitStopTime(integer particleSystemId)

Arguments

integer particleSystemId particle system id

Return Values

float emitStopTime emitter stop time

getEmitterShape

Description

Returns the emitter shape of the particle system

Definition

getEmitterShape(integer particleSystemId)

Arguments

integer particleSystemId particle system id

Return Values

integer shapeId id of the emitter shape

getParticleSystemAverageSpeed

Description

Get particle system average speed.

Definition

getParticleSystemAverageSpeed(integer particleSystemId)

Arguments

integer particleSystemId particle system id

Return Values

float normalSpeed value of normal speed

float tangentSpeed value of tangential speed

getParticleSystemLifespan

Description

Get particle system life span.

Definition

getParticleSystemLifespan(integer particleSystemId)

Arguments

integer particleSystemId particle system id

Return Values

float lifeSpan value of life span

resetEmitStartTimer

Description

Resets the start timer of emitted particles.

Definition

resetEmitStartTimer(integer particleSystemId, float time scale)

Arguments

integer particleSystemId particle system id

float time scale time scale (default is 1)

resetEmitStopTimer

Description

Resets the stop timer of emitted particles.

Definition

resetEmitStopTimer(integer particleSystemId, float time scale)

Arguments

integer particleSystemId particle system id

float time scale time scale (default is 1)

resetNumOfEmittedParticles

Description

Resets the counter of emitted particles. This is used if the maxEmit attribute is set for the particle system.

Definition

resetNumOfEmittedParticles(integer particleSystemId)

Arguments

integer particleSystemId particle system id

setEmitCountScale

Description

Set particle system count scale

Definition

setEmitCountScale(integer particleSystemId, float countScale)

Arguments

integer particleSystemId particle system id
float countScale value of count scale

setEmitStartTime

Description

Set emitter starting time.

Definition

setEmitStartTime(integer particleSystemId, float emitStartTime)

Arguments

integer particleSystemId particle system id
float emitStartTime emitter start time

setEmitStopTime

Description

Set emitter stop time.

Definition

setEmitStopTime(integer particleSystemId, float emitStopTime)

Arguments

integer particleSystemId particle system id
float emitStopTime emitter stop time

setEmitterShape

Description

Sets the emitter shape of the particle system

Definition

setEmitterShape(integer particleSystemId, integer shapeId)

Arguments

integer particleSystemId particle system id
integer shapeId id of the emitter shape

setEmittingState

Description

Set whether the particle system should emit new particles

Definition

setEmittingState(integer particleSystemId, boolean state)

Arguments

integer particleSystemId particle system id
boolean state if true, new particles are emitted

setParticleSystemLifespan

Description

Set particle system life span.

Definition

setParticleSystemLifespan(integer particleSystemId, float lifeSpan, boolean keepBlendTimes)

Arguments

integer	particleSystemId	particle system id
float	lifeSpan	value of life span
boolean	keepBlendTimes	if true, blend times are kept

setParticleSystemLifespan

Description

Set particle system life span.

Definition

setParticleSystemLifespan(integer particleSystemId, float lifeSpan, boolean keepBlendTimes)

Arguments

integer	particleSystemId	particle system id
float	lifeSpan	value of life span
boolean	keepBlendTimes	if true, blend times are kept

setParticleSystemTimeScale

Description

Sets the time scale for the particle simulation.

Definition

setParticleSystemTimeScale(integer particleSystemId, float timeScale)

Arguments

integer	particleSystemId	particle system id
float	timeScale	time scale for the particle simulation

Physics

addDifferential

Description

Add differential

Definition

addDifferential(integer objectId, integer diff0Index, boolean diffIndex0IsWheel, integer diff1Index, boolean diffIndex1IsWheel, float ratio, float bias)

Arguments

integer	objectId	id of the object
integer	diff0Index	differential 0 index
boolean	diffIndex0IsWheel	is wheel state
integer	diff1Index	differential 1 index
boolean	diffIndex1IsWheel	is wheel state
float	ratio	ratio
float	bias	bias

addForce

Description

Add force to object

Definition

addForce(integer transformId, float forceX, float forceY, float forceZ, float positionX, float positionY, float positionZ, boolean isPositionLocal)

Arguments

integer	transformId	id of transform object
float	forceX	force x
float	forceY	force y
float	forceZ	force z
float	positionX	position x
float	positionY	position y
float	positionZ	position z
boolean	isPositionLocal	is position local

addImpulse

Description

Add impulse to object

Definition

addImpulse(integer transformId, float impulseX, float impulseY, float impulseZ, float positionX, float positionY, float positionZ, boolean isPositionLocal)

Arguments

integer	transformId	id of transform object
float	impulseX	impulse x
float	impulseY	impulse y
float	impulseZ	impulse z
float	positionX	position x
float	positionY	position y
float	positionZ	position z
boolean	isPositionLocal	is position local

addToPhysics

Description

Add to physics

Definition

addToPhysics(integer transformId)

Arguments

integer transformId id of transform object

addTorque

Description

Adds torque to a collision

Definition

addTorque(integer objectId, float x, float y, float z)

Arguments

integer objectId id of the object

float	x	x torque
float	y	y torque
float	z	z torque

addTorqueImpulse

Description

Adds torque impulse to a collision

Definition

addTorqueImpulse(integer objectId, float x, float y, float z)

Arguments

integer	objectId	id of the object
float	x	x torque impulse
float	y	y torque impulse
float	z	z torque impulse

addVehicleLink

Description

Add vehicle link

Definition

addVehicleLink(integer transformId, integer transformId2)

Arguments

integer	transformId	id of transform object
integer	transformId2	id of second transform object

computeWheelShapeTireForces

Description

Calculate wheel shape tire forces

Definition

computeWheelShapeTireForces(integer transformId, integer wheelShapeIndex, float longSlipRatio, float latSlipAngle, float tireLoad)

Arguments

integer	transformId	id of transform object
integer	wheelShapeIndex	wheel shape index
float	longSlipRatio	longitudal slip ratio
float	latSlipAngle	lateral slip angle
float	tireLoad	tire load

Return Values

float	longForce	longitudal force
float	latForce	lateral force

createCCT

Description

Create character controller (y axis capsule based)

Definition

createCCT(integer transformId, float radius, float height, float stepOffset, float slopeLimit, float skinWidth, integer collisionMask, float mass)

Arguments

integer	transformId	id of transform object
float	radius	physics time scale
float	height	physics time scale
float	stepOffset	physics time scale
float	slopeLimit	physics time scale

float	skinWidth	skin width
integer	collisionMask	collision mask
float	mass	character mass [kg]

Return Values

integer characterIndex character index numbe

createWheelShape

Description

Create wheel shape

Definition

createWheelShape(integer transformId, float positionX, float positionY, float positionZ, float radius, float suspensionTravel, float spring, float damper, float mass, integer collisionMask, integer wheelShapeIndex)

Arguments

integer	transformId	id of transform object
float	positionX	position x
float	positionY	position y
float	positionZ	position z
float	radius	radius
float	suspensionTravel	suspension travel
float	spring	spring
float	damper	damper
float	mass	mass
integer	collisionMask	collision mask
integer	wheelShapeIndex	wheel shape index

Return Values

integer wheelShapeIndex wheel shape index

getAngularDamping

Description

Get angular damping

Definition

getAngularDamping(integer transformId)

Arguments

integer transformId id of transform object

Return Values

float angularDamping angular damping

getAngularVelocity

Description

Get angular velocity of transform object

Definition

getAngularVelocity(integer transformId)

Arguments

integer transformId id of transform object

Return Values

float velocityX x value of angular velocity

float velocityY y value of angular velocity
float velocityZ z value of angular velocity

getCCTCollisionFlags

Description

Get character controller collision flags

Definition

getCCTCollisionFlags(integer characterIndex)

Arguments

integer characterIndex character index number

Return Values

boolean side side flag
boolean up up flag
boolean down down flag

getCenterOfMass

Description

Get center of mass

Definition

getCenterOfMass(integer transformId)

Arguments

integer transformId id of transform object

Return Values

float x x position
float y y position
float z z position

getCollisionMask

Description

Get collision mask

Definition

getCollisionMask(integer transformId)

Arguments

integer transformId id of transform object

Return Values

integer mask collision mask

getDensity

Description

Get density

Definition

getDensity(integer transformId)

Arguments

integer transformId id of transform object

Return Values

float density density

getLinearDamping

Description

Get linear damping

Definition

getLinearDamping(integer transformId)

Arguments

integer transformId id of transform object

Return Values

float linearDamping linear damping

getLinearVelocity

Description

Get linear velocity of transform object

Definition

getLinearVelocity(integer transformId)

Arguments

integer transformId id of transform object

Return Values

float velocityX x value of velocity

float velocityY y value of velocity

float velocityZ z value of velocity

getMass

Description

Get mass

Definition

getMass(integer transformId)

Arguments

integer transformId id of transform object

Return Values

float mass mass

getMotorRotationSpeed

Description

Set vehicle properties

Definition

getMotorRotationSpeed(integer transformId)

Arguments

integer transformId id of transform object

Return Values

float motorRotSpeed motor rotation speed

float clutchRotSpeed clutch rotation speed

float motorLoad maximum motor load

getRigidBodyType

Description

Get rigid body type

Definition

getRigidBodyType(integer transformId)

Arguments

integer transformId id of transform object

Return Values

string type rigid body type ("static", "dynamic", "kinematic" or "norigidbody")

getVelocityAtLocalPos

Description

Get velocity at local position of transform object

Definition

getVelocityAtLocalPos(integer transformId, float positionX, float positionY, float positionZ)

Arguments

integer transformId id of transform object
float positionX x value of local position
float positionY y value of local position
float positionZ z value of local position

Return Values

float velocityX x value of velocity
float velocityY y value of velocity
float velocityZ z value of velocity

getVelocityAtWorldPos

Description

Get velocity at world position of transform object

Definition

getVelocityAtWorldPos(integer transformId, float positionX, float positionY, float positionZ)

Arguments

integer transformId id of transform object
float positionX x value of world position
float positionY y value of world position
float positionZ z value of world position

Return Values

float velocityX x value of velocity
float velocityY y value of velocity
float velocityZ z value of velocity

getVolume

Description

Get volume

Definition

getVolume(integer transformId)

Arguments

integer transformId id of transform object

Return Values

float volume volume

getWheelShapeAxleSpeed

Description

Get wheel shape axle speed

Definition

getWheelShapeAxleSpeed(integer transformId, integer wheelShapeIndex)

Arguments

integer transformId id of transform object

integer wheelShapeIndex wheel shape index

Return Values

float axleSpeed axle speed

getWheelShapeContactForce

Description

Get wheel shape contact force

Definition

getWheelShapeContactForce(integer transformId, integer wheelShapeIndex)

Arguments

integer transformId id of transform object

integer wheelShapeIndex wheel shape index

Return Values

float contactForce contact force

getWheelShapeContactNormal

Description

Get wheel shape contact normal

Definition

getWheelShapeContactNormal(integer transformId, integer wheelShapeIndex)

Arguments

integer transformId id of transform object

integer wheelShapeIndex wheel shape index

Return Values

float x x value of contact normal

float y y value of contact normal

float z z value of contact normal

getWheelShapeContactObject

Description

Get wheel shape contact object

Definition

getWheelShapeContactObject(integer transformId, integer wheelShapeIndex)

Arguments

integer transformId id of transform object

integer wheelShapeIndex wheel shape index

Return Values

integer wheelShapeContactObject wheel shape contact object

getWheelShapeContactPoint

Description

Get wheel shape contact point

Definition

getWheelShapeContactPoint(integer transformId, integer wheelShapeIndex)

Arguments

integer transformId id of transform object

integer wheelShapeIndex wheel shape index

Return Values

float positionX position x

float positionY position y

float positionZ position z

float contactSkinWidth skinwidth of contact object

getWheelShapePosition

Description

Get wheel shape contact point

Definition

getWheelShapePosition(integer transformId, integer wheelShapeIndex)

Arguments

integer transformId id of transform object

integer wheelShapeIndex wheel shape index

Return Values

float positionX position x

float positionY position y

float positionZ position z

float rotation rotation

getWheelShapeSlip

Description

Get wheel shape slip

Definition

getWheelShapeSlip(integer transformId, integer wheelShapeIndex)

Arguments

integer transformId id of transform object

integer wheelShapeIndex wheel shape index

Return Values

float wheelShapeSlip longitudal slip

float wheelShapeSlip lateral slip

JointConstructor:setActors

Description

Set joint actors

Definition

JointConstructor:setActors(integer actor1Id, integer actor2Id)

Arguments

integer actor1Id id of actor 1

integer actor2Id id of actor 2

JointConstructor:setJointTransforms

Description

Set joint transforms

Definition

JointConstructor:setJointTransforms(integer jointNode1, integer jointNode2)

Arguments

integer jointNode1 id of joint node 1

integer jointNode2 id of joint node 2

moveCCT

Description

Enqueue character movement

Definition

moveCCT(integer characterIndex, float x, float y, float z, integer collisionMasks)

Arguments

integer	characterIndex	character index number
float	x	x value
float	y	y value
float	z	z value
integer	collisionMasks	collision masks

overlapBox

Description

Overlap box objects

Definition

overlapBox(float x, float y, float z, float rx, float ry, float rz, float ex, float ey, float ez, string overlapFunctionCallback, object targetObject, integer collisionMask, boolean includeDynamics, boolean includeStatics, boolean exactTest)

Arguments

float	x	center x
float	y	center y
float	z	center z
float	rx	rotation x
float	ry	rotation y
float	rz	rotation z
float	ex	extent x
float	ey	extent y
float	ez	extent z
string	overlapFunctionCallback	overlap function callback
object	targetObject	target object (optional), the callback function is called as a member function of targetobject
integer	collisionMask	collisionMask
boolean	includeDynamics	include dynamics
boolean	includeStatics	include statics
boolean	exactTest	exact test

Return Values

integer numShapes number of shape overlaps

overlapSphere

Description

Overlap sphere objects

Definition

overlapSphere(float x, float y, float z, float radius, string overlapFunctionCallback, object targetObject, integer collisionMask, boolean includeDynamics, boolean includeStatics, boolean exactTest)

Arguments

float	x	center x
float	y	center y
float	z	center z
float	radius	radius
string	overlapFunctionCallback	overlap function callback
object	targetObject	target object (optional), the callback function is called as a member function of targetobject
integer	collisionMask	collisionMask
boolean	includeDynamics	include dynamics
boolean	includeStatics	include statics
boolean	exactTest	exact test

Return Values

integer numShapes number of shape overlaps

raycastAll

Description

Raycast objects, see raycast callback function

Definition

raycastAll(float x, float y, float z, float nx, float ny, float nz, string raycastFunctionCallback, float maxDistance, object targetObject, integer collisionMask, boolean generateNormal)

Arguments

float	x	origin x
float	y	origin y
float	z	origin z
float	nx	direction x
float	ny	direction y
float	nz	direction z
string	raycastFunctionCallback	raycast function callback. see raycast callback function
float	maxDistance	max distance
object	targetObject	target object (optional), the callback function is called as a member function of targetobject (default nil)
integer	collisionMask	collision mask (optional), the mask to filter colliding objects (default 0xffffffff)
boolean	generateNormal	generate normal (optional), if set and true, the normal at the collision point is passed as arguments to the callback function (default false)

Return Values

integer numShapes number of shapes hit

raycastClosest

Description

Raycast closest object, see raycast callback function

Definition

raycastClosest(float x, float y, float z, float nx, float ny, float nz, string raycastFunctionCallback, float maxDistance, object targetObject, integer collisionMask, boolean generateNormal)

Arguments

float	x	origin x
float	y	origin y
float	z	origin z
float	nx	direction x
float	ny	direction y
float	nz	direction z
string	raycastFunctionCallback	raycast function callback. see raycast callback function
float	maxDistance	max distance
object	targetObject	target object (optional), the callback function is called as a member function of targetobject (default nil)
integer	collisionMask	collision mask (optional), the mask to filter colliding objects (default 0xffffffff)
boolean	generateNormal	generate normal (optional), if set and true, the normal at the collision point is passed as arguments to the callback function (default false)

Return Values

integer numShapes number of shapes hit

removeAllDifferentials

Description

remove all differential

Definition

removeAllDifferentials(integer transformId)

Arguments

integer transformId id of transform object

removeCCT

Description

Remove character controller

Definition

removeCCT(integer characterIndex)

Arguments

integer characterIndex character index number

removeFromPhysics

Description

Remove from physics

Definition

removeFromPhysics(integer transformId)

Arguments

integer transformId id of transform object

setAngularDamping

Description

Set angular damping

Definition

setAngularDamping(integer transformId, float angularDamping)

Arguments

integer transformId id of transform object
float angularDamping angular damping

setAngularVelocity

Description

Set angular velocity of transform object

Definition

setAngularVelocity(integer transformId, float velocityX, float velocityY, float velocityZ)

Arguments

integer transformId id of transform object
float velocityX x value of angular velocity
float velocityY y value of angular velocity
float velocityZ z value of angular velocity

setCenterOfMass

Description

Set center of mass

Definition

setCenterOfMass(integer transformId, float x, float y, float z)

Arguments

integer transformId id of transform object
float x x position
float y y position
float z z position

setCollisionMask

Description

Set collision mask

Definition

setCollisionMask(integer transformId, integer mask)

Arguments

integer transformId id of transform object
integer mask collision mask

setFrictionVelocity

Description

Sets friction velocity to collision

Definition

setFrictionVelocity(integer objectId, float velocity)

Arguments

integer objectId id of the object

float velocity velocity

setJointDrive

Description

Set joint drive. Drives orientation if position drive or angular velocity if velocity drive.

Definition

setJointDrive(integer transformId, boolean isLinear, boolean isPosition, float valueX, float valueY, float valueZ, float angle)

Arguments

integer	transformId	id of transform object
boolean	isLinear	is linear
boolean	isPosition	is position
float	valueX	orientation axis x or angular velocity x
float	valueY	orientation axis y or angular velocity y
float	valueZ	orientation axis z or angular velocity z
float	angle	orientation angle

setLinearDamping

Description

Set linear damping

Definition

setLinearDamping(integer transformId, float linearDamping)

Arguments

integer	transformId	id of transform object
float	linearDamping	linear damping

setLinearVelocity

Description

Set linear velocity of transform object

Definition

setLinearVelocity(integer transformId, float velocityX, float velocityY, float velocityZ)

Arguments

integer	transformId	id of transform object
float	velocityX	x value of velocity
float	velocityY	y value of velocity
float	velocityZ	z value of velocity

setMass

Description

Set mass

Definition

setMass(integer transformId, float mass)

Arguments

integer	transformId	id of transform object
float	mass	mass

setRigidBodyType

Description

Set rigid body type

Definition

setRigidBodyType(integer transformId, string type)

Arguments

integer	transformId	id of transform object
string	type	rigid body type ("static", "dynamic", "kinematic" or "norigidbody")

setSolverIterationCount

Description

Set solver iteration count

Definition

setSolverIterationCount(integer transformId, integer count)

Arguments

integer	transformId	id of transform object
integer	count	number of iteration

setVehicleProps

Description

Set vehicle properties

Definition

setVehicleProps(integer transformId, float torque, float maxDriveSpeed, float gearRatio, float maxClutchTorque)

Arguments

integer	transformId	id of transform object
float	torque	torque
float	maxDriveSpeed	maximum driving speed
float	gearRatio	gear ratio
float	maxClutchTorque	maximum clutch torque

setWheelShapeForcePoint

Description

Set wheel shape force point

Definition

setWheelShapeForcePoint(integer transformId, integer wheelShapeIndex, float positionX, float positionY, float positionZ)

Arguments

integer	transformId	id of transform object
integer	wheelShapeIndex	wheel shape index
float	positionX	position x
float	positionY	position y
float	positionZ	position z

setWheelShapeProps

Description

Set wheel shape properties

Definition

setWheelShapeProps(integer transformId, integer wheelShapeIndex, float motorTorque, float brakeTorque, float steerAngle)

Arguments

integer	transformId	id of transform object
integer	wheelShapeIndex	wheel shape index
float	motorTorque	motor torque
float	brakeTorque	brake torque
float	steerAngle	steer angle

setWheelShapeTireFriction

Description

Set wheel shape tire friction

Definition

setWheelShapeTireFriction(integer transformId, integer wheelShapeIndex, float maxLongStiffness, float maxLatStiffness, float maxLatStiffnessTireLoad, float frictionMultiplier)

Arguments

integer	transformId	id of transform object
integer	wheelShapeIndex	wheel shape index
float	maxLongStiffness	longitudal stiffness [t/rad] (force per slip ratio)
float	maxLatStiffness	peak lateral stiffness [t/rad] (force per slip angle)
float	maxLatStiffnessTireLoad	tire load at which the peak lateral stiffness is reached [t]
float	frictionMultiplier	friction multiplier modeling ground friction properties

simulatePhysics

Description

Enable/disable physics simulation

Definition

simulatePhysics(boolean state)

Arguments

boolean	state	enable simulation state

simulatePhysicsTimeScale

Description

Time scale of physics simulation

Definition

simulatePhysicsTimeScale(float scale)

Arguments

float	scale	physics time scale

updateDifferential

Description

update differential

Definition

updateDifferential(integer transformId, integer index, float ratio, float bias)

Arguments

integer transformId id of transform object
integer index index of differential
float ratio ratio
float bias bias

Spline

getSplineCV

Description

Get spline control vertex

Definition

getSplineCV(integer shapeId, integer index)

Arguments

integer shapeId id of curve shape object
integer index control vertex index

Return Values

float x control vertex x coordinate
float y control vertex y coordinate
float z control vertex z coordinate

getSplineDirection

Description

Get spline direction

Definition

getSplineDirection(integer shapeId, float time)

Arguments

integer shapeId id of curve shape object
float time time

Return Values

float dirX x direction
float dirY y direction
float dirZ z direction

getSplineLength

Description

Get spline length

Definition

getSplineLength(integer shapeId)

Arguments

integer shapeId id of curve shape object

Return Values

float length length of spline

getSplineNumOfCV

Description

Get number of spline control vertices

Definition

getSplineNumOfCV(integer shapeId)

Arguments

integer shapeId id of curve shape object

Return Values

integer num number of spline control vertices

getSplineOrientation

Description

Get spline orientation

Definition

getSplineOrientation(integer shapeId, float time, float upDirX, float upDirY, float upDirZ)

Arguments

integer shapeId id of curve shape object
float time time [0,1]
float upDirX up direction x coordinate
float upDirY up direction y coordinate
float upDirZ up direction z coordinate

Return Values

float rx x rotation
float ry y rotation
float rz z rotation

getSplinePosition

Description

Get spline position

Definition

getSplinePosition(integer shapeId, float time)

Arguments

integer shapeId id of curve shape object
float time time [0,1]

Return Values

float x x coordinate
float y y coordinate
float z z coordinate

Animation

assignAnimTrackClip

Description

Assign clip to animation track

Definition

assignAnimTrackClip(integer characterSetId, integer trackId, integer clipIndex)

Arguments

integer characterSetId id of the characterset object
integer trackId track number
integer clipIndex clip index number

clearAnimTrackClip

Description

Clear animation track clip assignment

Definition

clearAnimTrackClip(integer characterSetId, integer trackId)

Arguments

integer characterSetId id of the characterset object

integer trackId track number

cloneAnimCharacterSet

Description

Clone anim character set

Definition

cloneAnimCharacterSet(integer objectId, integer targetId)

Arguments

integer objectId id of the object

integer targetId id of the target object

Return Values

boolean success was operation successful

disableAnimTrack

Description

Disable animation track

Definition

disableAnimTrack(integer characterSetId, integer trackId)

Arguments

integer characterSetId id of the characterset object

integer trackId track number

enableAnimTrack

Description

Enable animation track

Definition

enableAnimTrack(integer characterSetId, integer trackId)

Arguments

integer characterSetId id of the characterset object

integer trackId track number

getAnimCharacterSet

Description

Get animation character set id

Definition

getAnimCharacterSet(integer objectId)

Arguments

integer objectId id of the object

Return Values

integer characterSetId id of the characterset object

getAnimClipDuration

Description

Get the duration of the clip at the given index

Definition

getAnimClipDuration(integer characterSetId, integer index)

Arguments

integer characterSetId id of the characterset object
integer index clip index

Return Values

float duration duration

getAnimClipIndex

Description

Return the index of the clip with the given name

Definition

getAnimClipIndex(integer characterSetId, string clipName)

Arguments

integer characterSetId id of the characterset object
string clipName the name of the clip to find

Return Values

integer index index of the clip, -1 if clip was not found

getAnimNumOfClips

Description

Get number of clips

Definition

getAnimNumOfClips(integer characterSetId)

Arguments

integer characterSetId id of the characterset object

Return Values

integer numClips number of clips

getAnimTrackAssignedClip

Description

Get animation track assigned clip index number

Definition

getAnimTrackAssignedClip(integer characterSetId, integer trackId)

Arguments

integer characterSetId id of the characterset object
integer trackId track number

Return Values

integer clipIndex clip index number

getAnimTrackBlendWeight

Description

Get animation track blend weight

Definition

getAnimTrackBlendWeight(integer characterSetId, integer trackId)

Arguments

integer characterSetId id of the characterset object
integer trackId track number

Return Values

float weight blend weight

getAnimTrackTime

Description

Get animation track loop state

Definition

getAnimTrackTime(integer characterSetId, integer trackId)

Arguments

integer characterSetId id of the characterset object
integer trackId track number

Return Values

float time time

isAnimTrackClipAssigned

Description

Is clip assigned to animation track

Definition

isAnimTrackClipAssigned(integer characterSetId, integer trackId)

Arguments

integer characterSetId id of the characterset object
integer trackId track number

Return Values

boolean state assigned state

isAnimTrackEnabled

Description

Is animation track enabled

Definition

isAnimTrackEnabled(integer characterSetId, integer trackId)

Arguments

integer characterSetId id of the characterset object
integer trackId track number

Return Values

boolean state enable state

setAnimTrackBlendWeight

Description

Set animation track blend weight

Definition

setAnimTrackBlendWeight(integer characterSetId, integer trackId, float weight)

Arguments

integer characterSetId id of the characterset object
integer trackId track number
float weight blend weight

setAnimTrackLoopState

Description

Set animation track loop state

Definition

setAnimTrackLoopState(integer characterSetId, integer trackId, boolean loopState)

Arguments

integer	characterSetId	id of the characterset object
integer	trackId	track number
boolean	loopState	loop state

setAnimTrackSpeedScale

Description

Set animation track speed scale

Definition

setAnimTrackSpeedScale(integer characterSetId, integer trackId, float speedScale)

Arguments

integer	characterSetId	id of the characterset object
integer	trackId	track number
float	speedScale	speed scale

setAnimTrackTime

Description

Set animation track time

Definition

setAnimTrackTime(integer characterSetId, integer trackId, float time, boolean immediateUpdate (optional))

Arguments

integer	characterSetId	id of characterset object
integer	trackId	track number
float	time	time
boolean	immediateUpdate (optional)	if set to true, the animation is updated immediately, otherwise it is only updated the next frame. for performance reasons, only set this to true if you really need the immediate update.

Overlays

createImageOverlay

Description

Create overlay object

Definition

createImageOverlay(string textureFilename)

Arguments

string	textureFilename	texture file name

renderOverlay

Description

Render overlay

Definition

renderOverlay(integer overlayId, float x, float y, float width, float height)

Arguments

integer overlayId texture file name

float	x	x position [0,1]
float	y	y position [0,1]
float	width	width [0,1]
float	height	height [0,1]

setOverlayColor

Description

Set overlay color

Definition

setOverlayColor(integer overlayId, float red, float green, float blue, float alpha)

Arguments

integer overlayId texture file name

float	red	red value
float	green	green value
float	blue	blue value
float	alpha	alpha value

setOverlayRotation

Description

Set overlay rotation

Definition

setOverlayRotation(integer overlayId, float rotation, float x, float y)

Arguments

integer overlayId id of the overlay

float	rotation	rotation (rad)
float	x	x pos rotation center
float	y	y pos rotation center

setOverlayUVs

Description

Set overlay uv coordinates

Definition

setOverlayUVs(integer overlayId, float v0, float u0, float v1, float u1, float v2, float u2, float v3, float u3)

Arguments

integer overlayId texture file name

float	v0	v0 texture coordinate
float	u0	u0 texture coordinate
float	v1	v1 texture coordinate
float	u1	u1 texture coordinate
float	v2	v2 texture coordinate
float	u2	u2 texture coordinate
float	v3	v3 texture coordinate

float	u3	u3 texture coordinate

Sound

createAudioSource

Description

Create audio source object for 3D sounds

Definition

createAudioSource(string audioSourceName, string sampleFilename, float radius, float innerRadius, float volume, integer loops)

Arguments

string	audioSourceName	audio source name
string	sampleFilename	sample file name (.wav or .ogg)
float	radius	radius
float	innerRadius	inner radius
float	volume	volume [0,1]
integer	loops	number of loops (0: infinite loops)

createSample

Description

Create sample object

Definition

createSample(string objectName)

Arguments

string objectName sample object name

Return Values

integer sampleId id of the created sample

createStreamedSample

Description

Create streamed sample object

Definition

createStreamedSample(string objectName, boolean isBackgroundMusic)

Arguments

string	objectName	streamed sample object name
boolean	isBackgroundMusic	is background music

Return Values

integer sampleId id of the created sample

getAudioSourceSample

Description

Gets the sample id of an audio source

Definition

getAudioSourceSample(integer objectId)

Arguments

integer objectId id of the audio source

Return Values

integer sampleId sample id

getCurrentMasterVolume

Description

Get current master volume

Definition

getCurrentMasterVolume()

Return Values

float volume current master volume

getMasterVolume

Description

Get master volume

Definition

getMasterVolume()

Return Values

float volume master volume

getSampleDuration

Description

Get sample duration

Definition

getSampleDuration(integer sampleId)

Arguments

integer sampleId id of sample object

Return Values

float duration duration value

getSamplePitch

Description

Set sample pitch

Definition

getSamplePitch(integer sampleId)

Arguments

integer sampleId id of sample object

Return Values

float pitch pitch of sample object

getSamplePlayOffset

Description

Get sample play offset

Definition

getSamplePlayOffset(integer sampleId)

Arguments

integer sampleId id of the sample object

Return Values

float offset sample play offset

getSampleVelocity

Description

Get velocity of a sample object

Definition

getSampleVelocity(integer sampleId)

Arguments

integer sampleId id of the sample object

Return Values

float x velocity value towards x value

float y velocity value towards y value

float z velocity value towards z value

getSampleVolume

Description

Get sample volume

Definition

getSampleVolume(integer sampleId)

Arguments

integer sampleId id of the sample object

Return Values

float volume volume value

getStreamedSampleVolume

Description

Get streamed sample volume

Definition

getStreamedSampleVolume(integer sampleId)

Arguments

integer sampleId id of the streamed sample object

Return Values

float volume volume value

isSamplePlaying

Description

Is sample playing

Definition

isSamplePlaying(integer sampleId)

Arguments

integer sampleId id of the sample object

Return Values

boolean isPlaying is sample playing

loadSample

Description

Load sample object

Definition

loadSample(integer sampleId, string sampleFilename, boolean b3DSound)

Arguments

integer sampleId id of the sample object

string sampleFilename sample filename (.wav)
boolean b3DSound true=enables 3d sound-rendering, false=disables 3d sound-rendering

Return Values

boolean success was operation successful

loadStreamedSample

Description

Load streamed sample object

Definition

loadStreamedSample(integer sampleId, string bgmusicFilename)

Arguments

integer sampleId id of the streamed sample object
string bgmusicFilename streamed sample filename (.ogg)

pauseStreamedSample

Description

Pause streamed sample object

Definition

pauseStreamedSample(integer sampleId)

Arguments

integer sampleId id of the streamed sample object

playSample

Description

Play sample object

Definition

playSample(integer sampleId, integer loops, float volume, float offset)

Arguments

integer sampleId id of the sample object
integer loops number of loops [0,n] 0 = endless looping
float volume volume [0,1]
float offset offset in milliseconds

playStreamedSample

Description

Play streamed sample object

Definition

playStreamedSample(integer sampleId, integer repeat)

Arguments

integer sampleId id of the streamed sample object
integer repeat number of recurrences. 0 means endless loop

resumeStreamedSample

Description

Resume streamed sample

Definition

resumeStreamedSample(integer sampleId)

Arguments

integer sampleId id of the streamed sample object

setAudioCullingWorldProperties

Description

Set audio culling world properties

Definition

setAudioCullingWorldProperties(float gridMinX, float gridMinY, float gridMinZ, float gridMaxX, float gridMaxY, float gridMaxZ, integer gridSize)

Arguments

float	gridMinX	gridMinX
float	gridMinY	gridMinY
float	gridMinZ	gridMinZ
float	gridMaxX	gridMaxX
float	gridMaxY	gridMaxY
float	gridMaxZ	gridMaxZ
integer	gridSize	gridSize

setSamplePitch

Description

Set sample pitch

Definition

setSamplePitch(integer sampleId, float pitch)

Arguments

integer sampleId id of the sample object

float	pitch	pitch value [0.5-2.0]

setSampleVelocity

Description

Set velocity of a sample object

Definition

setSampleVelocity(integer sampleId, float x, float y, float z)

Arguments

integer sampleId id of the sample object

float	x	velocity value towards x value
float	y	velocity value towards y value
float	z	velocity value towards z value

setSampleVolume

Description

Set sample volume

Definition

setSampleVolume(integer sampleId, float volume)

Arguments

integer sampleId id of the sample object

float	volume	volume value

setStreamedSampleVolume

Description

Set streamed sample volume

Definition

setStreamedSampleVolume(integer sampleId, float volume)

Arguments

integer sampleId id of the streamed sample object
float volume volume value

stopSample

Description

Stop sample object

Definition

stopSample(integer sampleId)

Arguments

integer sampleId id of the sample object

stopStreamedSample

Description

Stop streamed sample

Definition

stopStreamedSample(integer sampleId)

Arguments

integer sampleId id of the streamed sample object

Input

getGamepadAxisLabel

Description

Get joystick/gamepad axis label

Definition

getGamepadAxisLabel(integer axisNumber, integer gamepadIndex)

Arguments

integer axisNumber axis number [0=input.axis_1, 5=input.axis_6]
integer gamepadIndex joystick/gampad index

Return Values

string axisLabel axis label

getGamepadButtonLabel

Description

Get joystick/gamepad button label

Definition

getGamepadButtonLabel(integer buttonNumber, integer gamepadIndex)

Arguments

integer buttonNumber button number [0=input.button_1, 15==input.button_16]
integer gamepadIndex joystick/gampad index

Return Values

string buttonLabel button label

getGamepadName

Description

Get name of joystick/gamepad

Definition

getGamepadName(integer gamepadIndex)

Arguments

integer gamepadIndex joystick/gampad index

Return Values

string gamepadName name of joystick/gampad

getInputAxis

Description

Get joystick/gamepad axis value

Definition

getInputAxis(integer axisNumber, integer gamepadIndex)

Arguments

integer axisNumber axis number [0=input.axis_1, 5=input.axis_6]

integer gamepadIndex joystick/gampad index

Return Values

float axisValue axis value [-1, 1]

getInputButton

Description

Get joystick/gamepad button value

Definition

getInputButton(integer buttonNumber, integer gamepadIndex)

Arguments

integer buttonNumber button number [0=input.button_1, 15==input.button_16]

integer gamepadIndex joystick/gampad index

Return Values

float buttonValue button value [0, 1]

getNumOfGamepads

Description

Get number of joysticks/gamepads

Definition

getNumOfGamepads()

Return Values

integer numOfGamepads number of joysticks/gampads

XML

createXMLFile

Description

Create an empty XML file

Definition

createXMLFile(string objectName, string filename, string rootNodeName)

Arguments

string objectName internal name for the object created

string filename filename (full path)
string rootNodeName name of the root node

Return Values

integer xmlId id of the xml object

getXMLBool

Description

Get XML file boolean attribute.

Definition

getXMLBool(integer xmlId, string attributePath)

Arguments

integer xmlId id of the xml object
string attributePath attribute path

Return Values

boolean value attribute value

getXMLFloat

Description

Get XML file float attribute.

Definition

getXMLFloat(integer xmlId, string attributePath)

Arguments

integer xmlId id of the xml object
string attributePath attribute path

Return Values

float value attribute value

getXMLInt

Description

Get XML file integer attribute.

Definition

getXMLInt(integer xmlId, string attributePath)

Arguments

integer xmlId id of the xml object
string attributePath attribute path

Return Values

integer value attribute value

getXMLString

Description

Get XML file string attribute.

Definition

getXMLString(integer xmlId, string attributePath)

Arguments

integer xmlId id of the xml object
string attributePath attribute path

Return Values

string value attribute value

hasXMLProperty

Description

Returns if an XML path is available in the file.

Definition

hasXMLProperty(integer xmlId, string attributePath)

Arguments

integer xmlId id of the xml object

string attributePath path to a node or an attribute

Return Values

boolean value is the given property available in the xml file

loadXMLFile

Description

Load XML file

Definition

loadXMLFile(string objectName, string filename)

Arguments

string objectName internal name for the object created

string filename filename (full path)

Return Values

integer xmlId id of the xml object

loadXMLFileFromMemory

Description

Load XML file from memory

Definition

loadXMLFileFromMemory(string objectName, string content)

Arguments

string objectName internal name for the object created

string content xml content

Return Values

integer xmlId id of the xml object

removeXMLProperty

Description

Remove XML property.

Definition

removeXMLProperty(integer xmlId, string propertyPath)

Arguments

integer xmlId id of the xml object

string propertyPath path to a node or an attribute

Return Values

boolean success was operation successful

saveXMLFile

Description

Save XML file

Definition

saveXMLFile(integer xmlId)

Arguments

integer xmlId id of the xml object

Return Values

boolean success was operation successful

saveXMLFileToMemory

Description

Save XML file to memory

Definition

saveXMLFileToMemory(integer xmlId)

Arguments

integer xmlId id of the xml object

Return Values

string memory id memory id

setXMLBool

Description

Set XML file boolean attribute.

Definition

setXMLBool(integer xmlId, string attributePath, boolean value)

Arguments

integer	xmlId	id of the xml object
string	attributePath	attribute path
boolean	value	value

setXMLFloat

Description

Set XML file float attribute.

Definition

setXMLFloat(integer xmlId, string attributePath, float value)

Arguments

integer	xmlId	id of the xml object
string	attributePath	attribute path
float	value	value

setXMLInt

Description

Set XML file integer attribute.

Definition

setXMLInt(integer xmlId, string attributePath, integer value)

Arguments

integer	xmlId	id of the xml object
string	attributePath	attribute path
integer	value	value

setXMLString

Description

Set XML file string attribute.

Definition

setXMLString(integer xmlId, string attributePath, string value)

Arguments

integer	xmlId	id of the xml object
string	attributePath	attribute path
string	value	value

Network

streamAlignReadToByteBoundary

Description

Stream align read to byte boundary

Definition

streamAlignReadToByteBoundary(integer streamId)

Arguments

integer streamId id of the network stream

streamAlignWriteToByteBoundary

Description

Stream align write to byte boundary

Definition

streamAlignWriteToByteBoundary(integer streamId)

Arguments

integer streamId id of the network stream

streamGetNumOfUnreadBits

Description

Stream get number of unread bits

Definition

streamGetNumOfUnreadBits(integer streamId)

Arguments

integer streamId id of the network stream

Return Values

integer value number of unread bit

streamGetReadOffset

Description

Stream get read offset

Definition

streamGetReadOffset(integer streamId)

Arguments

integer streamId id of the network stream

Return Values

integer offset read offset

streamGetWriteOffset

Description

Return the write pointer offset in bytes

Definition

streamGetWriteOffset(integer streamId)

Arguments

integer streamId id of the network stream

Return Values

integer offset write offset

streamReadBool

Description

Stream read boolean

Definition

streamReadBool(integer streamId)

Arguments

integer streamId id of the network stream

Return Values

boolean value value

streamReadFloat32

Description

Stream read 32bit float

Definition

streamReadFloat32(integer streamId)

Arguments

integer streamId id of the network stream

Return Values

float value value

streamReadInt16

Description

Stream read 16bit signed integer

Definition

streamReadInt16(integer streamId)

Arguments

integer streamId id of the network stream

Return Values

integer value value

streamReadInt32

Description

Stream read 32bit signed integer

Definition

streamReadInt32(integer streamId)

Arguments

integer streamId id of the network stream

Return Values

integer value value

streamReadInt8

Description

Stream read 8bit signed integer

Definition

streamReadInt8(integer streamId)

Arguments

integer streamId id of the network stream

Return Values

integer value value

streamReadIntN

Description

Stream read 32bit signed integer

Definition

streamReadIntN(integer streamId, integer numberofBits)

Arguments

integer streamId id of the network stream

integer numberofBits number of bits

Return Values

integer value value

streamReadManualTimestamp

Description

Stream read manual timestamp

Definition

streamReadManualTimestamp(integer streamId)

Arguments

integer streamId id of the network stream

Return Values

integer timestamp timestamp

streamReadString

Description

Stream read 32bit float

Definition

streamReadString(integer streamId)

Arguments

integer streamId id of the network stream

Return Values

string value value

streamReadUInt16

Description

Stream write 16-bit unsigned integer.

Definition

streamReadUInt16(integer streamId)

Arguments

integer streamId id of the network stream

Return Values

integer uint16 uint16

streamReadUInt8

Description

Stream read 8-bit unsigned integer.

Definition

streamReadUInt8(integer streamId)

Arguments

integer streamId id of the network stream

Return Values

integer uint8 uint8

streamReadUIntN

Description

Stream write N-bit unsigned integer. 0<N<32.

Definition

streamReadUIntN(integer streamId, integer numberOfBits)

Arguments

integer streamId id of the network stream

integer numberOfBits number of bits 2<n<32

Return Values

integer value value

streamSetReadOffset

Description

Set the read pointer to the given offset in bytes

Definition

streamSetReadOffset(integer streamId, integer offset)

Arguments

integer streamId id of the network stream

integer offset read offset in bytes

streamSetWriteOffset

Description

Set the write pointer to the given offset in bytes

Definition

streamSetWriteOffset(integer streamId, integer offset)

Arguments

integer streamId id of the network stream

integer offset write offset in bytes

streamWriteBool

Description

Stream write boolean

Definition

streamWriteBool(integer streamId, boolean value)

Arguments

integer streamId id of the network stream

boolean value value

Return Values

boolean success was operation successful

streamWriteFloat32

Description

Stream write 32bit float

Definition

streamWriteFloat32(integer streamId, float value)

Arguments

integer streamId id of the network stream

float value value

streamWriteInt16

Description

Stream write 16bit signed integer

Definition

streamWriteInt16(integer streamId, integer value)

Arguments

integer streamId id of the network stream

integer value value [-32768, 32768]

streamWriteInt32

Description

Stream write 32bit signed integer

Definition

streamWriteInt32(integer streamId, integer value)

Arguments

integer streamId id of the network stream

integer value value

streamWriteInt8

Description

Stream write 8bit signed integer

Definition

streamWriteInt8(integer streamId, integer value)

Arguments

integer streamId id of the network stream

integer value value [-127, 127]

streamWriteIntN

Description

Stream write N bit signed integer. 0 < N < 32.

Definition

streamWriteIntN(integer streamId, integer value, integer numberOfBits)

Arguments

integer streamId id of the network stream
integer value value [-2^(n-1), 2^(n-1)]
integer numberOfBits number of bits

streamWriteManualTimestamp

Description

Stream write manual timestamp

Definition

streamWriteManualTimestamp(integer streamId, integer timestampe)

Arguments

integer streamId id of the network stream
integer timestampe timestamp

streamWriteStream

Description

Stream write stream

Definition

streamWriteStream(integer streamId, integer partSize, boolean useReadStream)

Arguments

integer streamId id of the network stream
integer partSize part size bits
boolean useReadStream use read stream

streamWriteString

Description

Stream write string

Definition

streamWriteString(integer streamId, string value)

Arguments

integer streamId id of the network stream
string value value, max 64k characters

streamWriteTimestamp

Description

Stream write timestamp

Definition

streamWriteTimestamp(integer streamId)

Arguments

integer streamId id of the network stream

streamWriteUInt16

Description

Stream write 16-bit unsigned integer.

Definition

streamWriteUInt16(integer streamId, integer value)

Arguments

integer streamId id of the network stream
integer value value [0, 2^16]

streamWriteUInt8

Description

Stream write 8-bit unsigned integer.

Definition

streamWriteUInt8(integer streamId, integer uint8)

Arguments

integer streamId id of the network stream
integer uint8 uint 8

streamWriteUIntN

Description

Stream write N-bit unsigned integer. 0<N<32.

Definition

streamWriteUIntN(integer streamId, integer value, integer numberofBits)

Arguments

integer streamId id of the network stream
integer value value [0, 2^n]
integer numberofBits number of bits

Callbacks

createFunctionName

Description

Create callbacks are called during i3d loads

Definition

createFunctionName(integer objectId)

Arguments

integer objectId id of the object

draw

Description

This function is called once per frame to draw game objects. Call render functions here.

Definition

draw()

init

Description

This function is called once on startup. Create and load objects here.

Definition

init(string args)

Arguments

string args command line arguments

Return Values

boolean success was operation successful

keyEvent

Description

This function is called when a key event occurs.

Definition

keyEvent(float unicode, float sym, float modifier, boolean isDown)

Arguments

float	unicode	unicode value
float	sym	sym key
float	modifier	key modifier
boolean	isDown	is down state

mouseEvent

Description

This function is called when a mouse event occurs.

Definition

mouseEvent(float posX, float posY, boolean isDown, boolean isUp, float button)

Arguments

float	posX	x position [0,1]
float	posY	y position [0,1]
boolean	isDown	is down state
boolean	isUp	is up state
float	button	button number

onContactFunctionName

Description

This function is called when a contact event occurs.

Definition

onContactFunctionName(integer objectId, integer otherObjectId, boolean isStart, float normalForce, float tangentialForce)

Arguments

integer	objectId	id of the object
integer	otherObjectId	id of the other object
boolean	isStart	is start touch
float	normalForce	normal contact force
float	tangentialForce	tangential contact force

packetReceived

Description

This function is called when a network packet was received.

Definition

packetReceived(integer packetType, integer timestamp, integer streamId)

Arguments

integer	packetType	type of the packet. all available types are stored in the network table
integer	timestamp	timestamp of when the packet was sent
integer	streamId	id of the stream containing the packet data

raycastFunctionName

Description

This function is called when a raycast hit event occurs.
See raycastAll and raycastClosest

Definition

raycastFunctionName(integer hitObjectId, float x, float y, float z, float distance, float nx, float ny, float nz, integer subShapeIndex)

Arguments

integer	hitObjectId	id of the hit node
float	x	world impact x
float	y	world impact y
float	z	world impact z
float	distance	distance
float	nx	world normal x (only valid if generatenormal was true)
float	ny	world normal y (only valid if generatenormal was true)
float	nz	world normal nz (only valid if generatenormal was true)
integer	subShapeIndex	sub shape index of destructible object. negative if normal rigid body (check for subshapeindex >= 0)

Return Values

boolean continue continue raycasting

triggerFunctionName

Description

This function is called when a trigger event occurs.

Definition

triggerFunctionName(integer triggerId, integer otherId, boolean onEnter, boolean onLeave, boolean onStay)

Arguments

integer	triggerId	trigger id
integer	otherId	id of the other object
boolean	onEnter	on enter
boolean	onLeave	on leave
boolean	onStay	on stay

update

Description

This function is called once per frame to update game state.

Definition

update(float dt)

Arguments

float dt time between this frame and last frame in milliseconds

Text Rendering

getCanRenderUnicode

Description

Get can render unicode

Definition

getCanRenderUnicode(integer unicode)

Arguments

integer unicode unicode

Return Values

boolean canRender can render unicode

getTextHeight

Description

Get text height

Definition

getTextHeight(float fontSize, string utf8string)

Arguments

float fontSize font size

string utf8string utf8 encoded string

Return Values

float textHeight text height

integer numLines number of lines

getTextLength

Description

Get text length

Definition

getTextLength(float fontSize, string utf8string, integer maxNumLines)

Arguments

float fontSize font size

string utf8string utf8 encoded string

integer maxNumLines max numbers of lines

Return Values

integer numChars number of characters

getTextLineLength

Description

Get text length

Definition

getTextLineLength(float fontSize, string utf8string, float maxWidth)

Arguments

float fontSize font size

string utf8string utf8 encoded string

float maxWidth max width

Return Values

integer numChars number of characters

getTextWidth

Description

Get text width

Definition

getTextWidth(float fontSize, string utf8string)

Arguments

float fontSize font size

string utf8string utf8 encoded string

Return Values

float textWidth text width

renderText

Description

Render text to viewport. See "asciiToUtf8" to convert to utf8

Definition

renderText(float x, float y, float size, string str)

Arguments

float x x position [0, 1]

float y y position [0, 1]

float size font size

string str utf8 encoded string to print

setTextAlignment

Description

Set the text alignment to be used for the following renderText calls

Definition

setTextAlignment(integer alignment)

Arguments

integer alignment alignment (rendertext.align_left, rendertext.align_center or rendertext.align_right)

setTextBold

Description

Set the text boldness to be used for the following renderText calls

Definition

setTextBold(boolean isBold)

Arguments

boolean isBold bold state

setTextColor

Description

Set the text color to be used for the following renderText calls

Definition

setTextColor(float r, float g, float b, float a)

Arguments

float r red color component [0, 1]

float g green color component [0, 1]

float b blue color component [0, 1]

float a alpha (opacity) [0, 1]

setTextLineBounds

Description

Set text line bounds

Definition

setTextLineBounds(integer startLine, integer numLines)

Arguments

integer startLine start line

integer numLines number of shown lines

setTextWidthScale

Description

Set the text scale width to be used for the following renderText calls

Definition

setTextWidthScale(float scaleWidth)

Arguments

float scaleWidth scale width

setTextWidthScale

Description

Set the text scale width to be used for the following renderText calls

Definition

setTextWidthScale(float scaleWidth)

Arguments

float scaleWidth scale width

setTextWrapWidth

Description

Set text wrap width

Definition

setTextWrapWidth(float wrapWidth)

Arguments

float wrapWidth wrap width

Terrain Detail

addDensityMaskedParallelogram

Description

Add density masked parallelogram

Definition

addDensityMaskedParallelogram(integer transformId, float x, float z, float dX1, float dZ1, float dX2, float dZ2, integer firstChannel, integer numChannels, integer mask id, integer maskFirstChannel, integer maskNumChannels, integer value)

Arguments

integer	transformId	id of transform object
float	x	x position
float	z	z position
float	dX1	x value of 1st direction
float	dZ1	z value of 1st direction
float	dX2	x value of 2nd direction
float	dZ2	z value of 2nd direction
integer	firstChannel	first channel
integer	numChannels	number of channels
integer	mask id	mask id

integer	maskFirstChannel	mark first channel
integer	maskNumChannels	mask number of channels
integer	value	value

Return Values

integer	density	density
integer	numChangedPixels	number of changed pixels
integer	totalDelta	total number of changed pixels
integer	totalNumPixels	total number of pixels

addDensityParallelogram

Description

Set density masked parallelogram

Definition

addDensityParallelogram(integer transformId, float x, float z, float dX1, float dZ1, float dX2, float dZ2, integer firstChannel, integer numChannels, integer value)

Arguments

integer	transformId	id of transform object
float	x	x position
float	z	z position
float	dX1	x value of 1st direction
float	dZ1	z value of 1st direction
float	dX2	x value of 2nd direction
float	dZ2	z value of 2nd direction
integer	firstChannel	first channel
integer	numChannels	number of channels
integer	value	value

Return Values

integer	density	density
integer	numChangedPixels	number of changed pixels
integer	totalDelta	total number of changed pixels
integer	totalNumPixels	total number of pixels

getDensityAtWorldPos

Description

Get density at world position

Definition

getDensityAtWorldPos(integer transformId, float x, float z)

Arguments

integer	transformId	id of transform object
float	x	x position
float	z	z position

Return Values

integer density density

getDensityMapFileName

Description

Get density map file name

Definition

getDensityMapFileName(integer transformId)

Arguments

integer transformId id of transform object

Return Values

string fileName file name

getDensityMapSize

Description

Get density map file name

Definition

getDensityMapSize(integer transformId)

Arguments

integer transformId id of transform object

Return Values

integer mapSize map size

getDensityMaskedParallelogram

Description

Get density masked parallelogram

Definition

getDensityMaskedParallelogram(integer transformId, float x, float z, float dX1, float dZ1, float dX2, float dZ2, integer firstChannel, integer numChannels, integer id, integer maskFirstChannel, integer maskNumChannels)

Arguments

integer	transformId	id of transform object
float	x	x position
float	z	z position
float	dX1	x value of 1st direction
float	dZ1	z value of 1st direction
float	dX2	x value of 2nd direction
float	dZ2	z value of 2nd direction
integer	firstChannel	first channel
integer	numChannels	number of channels
integer	id	id
integer	maskFirstChannel	mask first channel
integer	maskNumChannels	mask number of channels

Return Values

integer	density	density
integer	numPixels	number of pixels
integer	totalNumPixels	total number of pixels

getDensityParallelogram

Description

Get density parallelogram

Definition

getDensityParallelogram(integer transformId, float x, float z, float dX1, float dZ1, float dX2, float dZ2, integer firstChannel, integer numChannels)

Arguments

integer	transformId	id of transform object
float	x	x position
float	z	z position
float	dX1	x value of 1st direction
float	dZ1	z value of 1st direction
float	dX2	x value of 2nd direction
float	dZ2	z value of 2nd direction
integer	firstChannel	first channel
integer	numChannels	number of channels

Return Values

integer	density	density
integer	numPixels	number of pixels
integer	totalNumPixels	total number of pixels

getDensityParallelogramArea

Description

Get density parallelogram area

Definition

getDensityParallelogramArea(integer transformId, float x, float z, float dX1, float dZ1, float dX2, float dZ2)

Arguments

integer	transformId	id of transform object
float	x	x position
float	z	z position
float	dX1	x value of 1st direction
float	dZ1	z value of 1st direction
float	dX2	x value of 2nd direction
float	dZ2	z value of 2nd direction

Return Values

integer	density	density

getDensityRegion

Description

Get density region

Definition

getDensityRegion(integer transformId, float x, float z, float width, float height, integer firstChannel, integer numChannels)

Arguments

integer	transformId	id of transform object
float	x	x position
float	z	z position
float	width	width
float	height	height

integer firstChannel first channel

integer numChannels number of channels

Return Values

integer density sum density sum

getDensityRegionWorld

Description

Get density region world

Definition

getDensityRegionWorld(integer transformId, float x, float z, float width, float height, integer firstChannel, integer numChannels)

Arguments

integer	transformId	id of transform object
float	x	x position
float	z	z position
float	width	width
float	height	height
integer	firstChannel	first channel
integer	numChannels	number of channels

Return Values

integer type type

getGrowthNumStates

Description

Get number of growth states

Definition

getGrowthNumStates(integer foliageId)

Arguments

integer foliageId id of the foliage

Return Values

integer numStates number of states

getGrowthStateTime

Description

Get growth state time

Definition

getGrowthStateTime(integer foliageId)

Arguments

integer foliageId id of the foliage

Return Values

float stateTime growth state time

getTerrainAttributesAtWorldPos

Description

Get terrain attributes at world pos

Definition

getTerrainAttributesAtWorldPos(integer terrainId, float x, float y, float z, boolean xComb, boolean yComb, boolean zComb, boolean wComb, boolean uComb)

Arguments

integer	terrainId	id of the terrain
float	x	x position
float	y	y position
float	z	z position
boolean	xComb	is x combined
boolean	yComb	is y combined
boolean	zComb	is z combined
boolean	wComb	is w combined
boolean	uComb	is u combined

Return Values

float	x	x value
float	y	y value
float	z	z value
float	w	w value
float	u	u value

getTerrainDetailByName

Description

Get terrain detail by name

Definition

getTerrainDetailByName(integer terrainId, string detailName)

Arguments

integer	terrainId	id of the terrain
string	detailName	detail name

Return Values

integer	detailId	detail id

getTerrainDetailName

Description

Get terrain detail name

Definition

getTerrainDetailName(integer terrainId)

Arguments

integer	terrainId	id of the terrain

Return Values

string	viewDistance	view distance

getTerrainDetailNumChannels

Description

Get number of terrain detail channels

Definition

getTerrainDetailNumChannels(integer terrain id)

Arguments

integer terrain id id of the terrain

Return Values

integer numChannels number of channels

getTerrainDetailViewDistance

Description

Get terrain detail view distance

Definition

getTerrainDetailViewDistance(integer terrainId)

Arguments

integer terrainId id of the terrain

Return Values

float viewDistance view distance

getTerrainHeightAtWorldPos

Description

Get terrain height at world pos

Definition

getTerrainHeightAtWorldPos(integer terrainId, float x, float y, float z)

Arguments

integer terrainId id of the terrain

float	x	x position
float	y	y position
float	z	z position

Return Values

float height terrain height

getTerrainNormalAtWorldPos

Description

Get terrain normal at world pos

Definition

getTerrainNormalAtWorldPos(integer terrainId, float x, float y, float z)

Arguments

integer terrainId id of the terrain

float	x	x position
float	y	y position
float	z	z position

Return Values

float nx x normal

float ny y normal

float nz z normal

getTerrainSize

Description

Get terrain size

Definition

getTerrainSize(integer terrainId)

Arguments

integer terrainId id of the terrain

Return Values

float size terrain size

setDensityCompareParams

Description

Sets density compare parameters

Default Value ("greater" -1)

Definition

setDensityCompareParams(integer transformId, string comparator, integer a, void b, void b, void b)

Arguments

integer	transformId	id of transform object
string	comparator	greater, between
integer	a	?
void	b	?
void	b	?
void	b	?

setDensityMaskedParallelogram

Description

Set density masked parallelogram

Definition

setDensityMaskedParallelogram(integer transformId, float x, float z, float dX1, float dZ1, float dX2, float dZ2, integer firstChannel, integer numChannels, integer id, integer maskFirstChannel, integer maskNumChannels, integer mask id)

Arguments

integer	transformId	id of transform object
float	x	x position
float	z	z position
float	dX1	x value of 1st direction
float	dZ1	z value of 1st direction
float	dX2	x value of 2nd direction
float	dZ2	z value of 2nd direction
integer	firstChannel	first channel
integer	numChannels	number of channels
integer	id	id
integer	maskFirstChannel	mark first channel
integer	maskNumChannels	mask number of channels
integer	mask id	mask id

Return Values

integer	density	density
integer	numPixels	number of pixels
integer	totalNumPixels	total number of pixels

setDensityMaskParams

Description

Sets density mask

Default Value ("greater" 0)

Definition

setDensityMaskParams(integer transformId, string comparator, integer a, void b, void b, void b)

Arguments

integer	transformId	id of transform object
string	comparator	greater, between
integer	a	?
void	b	?
void	b	?
void	b	?

setDensityNewTypeIndexMode

Description

Set Density new type index mode

Definition

setDensityNewTypeIndexMode(integer transformId, integer newTypeIndexMode)

Arguments

integer	transformId	id of transform object
integer	newTypeIndexMode	0=update_index,1=keep_index,2=set_index_to_zero

setDensityParallelogram

Description

Set density parallelogram

Definition

setDensityParallelogram(integer transformId, float x, float z, float dX1, float dZ1, float dX2, float dZ2, integer firstChannel, integer numChannels, integer value)

Arguments

integer	transformId	id of transform object
float	x	x position
float	z	z position
float	dX1	x value of 1st direction
float	dZ1	z value of 1st direction
float	dX2	x value of 2nd direction
float	dZ2	z value of 2nd direction
integer	firstChannel	first channel
integer	numChannels	number of channels
integer	value	value

Return Values

integer	density	density
integer	numPixels	number of pixels
integer	totalNumPixels	total number of pixels

setDensityReturnValueShift

Description

Set density return value shift

Definition

setDensityReturnValueShift(integer transformId, integer shift)

Arguments

integer transformId id of transform object
integer shift shift value

setDensityTypeIndexCompareMode

Description

Set density type index compare mode

Definition

setDensityTypeIndexCompareMode(integer transformId, integer indexComopareMode)

Arguments

integer transformId id of transform object
integer indexComopareMode 0=type_compare_equal, 1=type_compare_notequal, 2=type_compare_none

setEnableGrowth

Description

Set enable growth

Definition

setEnableGrowth(integer objectId, boolean isEnabled)

Arguments

integer objectId id of the object
boolean isEnabled is growth enabled

setFoliageGrowingEnabled

Description

Set foliage growing enabled

Definition

setFoliageGrowingEnabled(boolean enabled)

Arguments

boolean enabled foliage growing enabled

setTypeIndexMaskedParallelogram

Description

Set indexed mask parallelogram type

Definition

setTypeIndexMaskedParallelogram(integer transformId, float x, float z, float dX1, float dZ1, float dX2, float dZ2, integer mask id, integer firstChannel, integer numChannels)

Arguments

integer	transformId	id of transform object
float	x	x position
float	z	z position
float	dX1	x value of 1st direction
float	dZ1	z value of 1st direction

float	dX2	x value of 2nd direction
float	dZ2	z value of 2nd direction
integer	mask id	mask id
integer	firstChannel	first channel
integer	numChannels	number of channels

Return Values

integer	numChangedPixels	number of changed pixels
integer	totalNumPixels	total number of pixels

Tire Track

addTrackPoint

Description

Adds a tire track position

Definition

addTrackPoint(integer systemId, integer trackId, float x, float y, float z, float x, float y, float z, float r, float g, float b, float a, float uw, float dTheta)

Arguments

integer	systemId	id of tyre track object
integer	trackId	id of tyre track
float	x	x position of track object
float	y	y position of tyre track object
float	z	z position of tyre track object
float	x	x part of up vector
float	y	y part of up vector
float	z	z part of up vector
float	r	red color component [0, 1]
float	g	green color component [0, 1]
float	b	blue color component [0, 1]
float	a	alpha (used to determine the amount of dirt) [0, 1]
float	uw	bumpiness [0 1]
float	dTheta	wheel dtheta/dt (used to determine forward/backward motion of wheel)

createTrack

Description

Creates a tire track system

Definition

createTrack(integer systemId, float width, integer atlasIndex)

Arguments

integer	systemId	id of tyre track object
float	width	tyre track width
integer	atlasIndex	index of atlas

Return Values

integer	trackid	id of tyre track

cutTrack

Description

Relinquish Track Segments

Definition

cutTrack(integer systemId, integer trackId)

Arguments

integer systemId id of tyre track object

integer trackId id of tyre track

destroyTrack

Description

Destroys a tire track system

Definition

destroyTrack(integer systemId, integer trackId)

Arguments

integer systemId id of tyre track object

integer trackId id of tyre track

eraseParallelogram

Description

Erase Segments Inside Parallelogram

Definition

eraseParallelogram(integer systemId, float startWorldX, float startWorldZ, float widthWorldX, float widthWorldZ, float heightWorldX, float heightWorldZ)

Arguments

integer	systemId	id of tyre track object
float	startWorldX	parallelogram x start position in world space
float	startWorldZ	parallelogram y start position in world space
float	widthWorldX	parallelogram x width in world space
float	widthWorldZ	parallelogram z width in world space
float	heightWorldX	parallelogram x height in world space
float	heightWorldZ	parallelogram z height in world space

Editor

addSelection

Description

Add node to selection

Definition

addSelection(integer objectId)

Arguments

integer objectId id of the object

Return Values

integer selectionIndex index of the selected object

clearSelection

Description

Clear the selection

Definition

clearSelection()

getNumSelected

Description

Get number of selected nodes

Definition

getNumSelected()

Return Values

integer numSelected number of selected nodes

getSelection

Description

Get selected nodes

Definition

getSelection(integer selectionIndex)

Arguments

integer selectionIndex index of the selected object

Return Values

integer objectId id of the object

AI

buildNavMesh

Description

Build the navigation mesh based on the specified world data

Definition

buildNavMesh(integer worldNode, float cellSize, float cellHeight, float agentHeight, float agentRadius, float agentMaxClimb, float agentMaxSlope, float minRegionSize, float mergeRegionSize, float maxEdgeLength, float maxSimplificationError, integer navMeshBuildMask, float terrainDetail, string terrainCullInfoLayer, integer terrainCullInfoLayerChannels)

Arguments

integer	worldNode	node which contains the world data to be considered
float	cellSize	cell size for nav mesh rasterization [m]
float	cellHeight	cell height for nav mesh rasterization [m]
float	agentHeight	height of the agent [m]
float	agentRadius	radius of the agent [m]
float	agentMaxClimb	maximum height an agent can climb [m]
float	agentMaxSlope	maximum slope an agent can climb [radians]
float	minRegionSize	minimum side length of a separate region [#cells]
float	mergeRegionSize	merge regions with a smaller side length [#cells]
float	maxEdgeLength	maximum length of an edge in the nav mesh [m]
float	maxSimplificationError	maximum error the nav mesh simplification can introdcue
integer	navMeshBuildMask	mask to match potential shapes for nav mesh generation
float	terrainDetail	scale of the detail level used for terrain [0 no terrain, 1 use all terrain vertices]
string	terrainCullInfoLayer	info layer used for culling the terrain. No culling if string is empty
integer	terrainCullInfoLayerChannels	mask to specify which channels need to be set. Cull the terrain if none of the specified channels are set

Return Values

boolean success is true, if the navigation mesh was built successfully

createNavMesh

Description

Create a navigation mesh node.

Definition

createNavMesh(string name)

Arguments

string name name of the nav mesh

Return Values

integer navMeshId id of the nav mesh

Rendering

setFog

Description

Set fog properties

Definition

setFog(string fogType, float startDistanceOrDensity, float endDistance, float r, float g, float b)

Arguments

string	fogType	fog type ("none", "linear", "exp" or "exp2")
float	startDistanceOrDensity	start distance (linear) or density (exp and exp2)
float	endDistance	end distance (linear)
float	r	red color component [0, 1]
float	g	green color component [0, 1]
float	b	blue color component [0, 1]

String

asciiToUtf8

Description

Converts an ascii latin1 (ISO 88859-1) encoded string to an utf8 string

Definition

asciiToUtf8(string asciiString)

Arguments

string asciiString ascii string

Return Values

string utf8string utf8 encoded string

unicodeToUtf8

Description

Converts an unicode value to an utf8 string

Definition

unicodeToUtf8(integer unicode)

Arguments

integer unicode unicode value

Return Values

string utf8string utf8 encoded string representing the unicode

utf8Strlen

Description

Returns the length of an utf8 formated string

Definition

utf8Strlen(string utf8string)

Arguments

string utf8string utf8 formated string

Return Values

integer length length of the given string

utf8Substr

Description

Returns a sub string of an utf8 formated string

Definition

utf8Substr(string utf8string, integer startIndex, integer length)

Arguments

string utf8string utf8 formated string

integer startIndex zero based start index

integer length maximal length of the sub string (optional)

Return Values

string subString utf8 formated sub string

utf8ToLower

Description

Returns a lower case string of an utf8 formated string

Definition

utf8ToLower(string utf8string)

Arguments

string utf8string utf8 formated string

Return Values

string utf8string lower case string of the given string

utf8ToUnicode

Description

Converts an utf8 string to unicode

Definition

utf8ToUnicode(string utf8string)

Arguments

string utf8string utf8 encoded string

Return Values

integer unicode unicode value

integer utf8Len length of the given string

utf8ToUpper

Description

Return a upper case string of an utf8 formated string

Definition

utf8ToUpper(string utf8string)

Arguments

string utf8string utf8 formated string

Return Values

string utf8string upper case string of the given string

Math

base64Decode

Description

base64 decode

Definition

base64Decode(string input)

Arguments

string input string to decode

Return Values

bytearray output output array

base64Encode

Description

base64 encode

Definition

base64Encode(bytearray input)

Arguments

bytearray input input array

Return Values

string output output string

bitAND

Description

bit AND-Operation

Definition

bitAND(integer value1, integer value2)

Arguments

integer value1 value 1 for operation

integer value2 value 2 for operation

Return Values

integer result result value

bitHighestSet

Description

bit HighestSet-Operation

Definition

bitHighestSet(integer input)

Arguments

integer input input

Return Values

integer highestBit highest bit set

bitOR

Description

bit OR-Operation

Definition

bitOR(integer value1, integer value2)

Arguments

integer value1 value 1 for operation

integer value2 value 2 for operation

Return Values

integer result result from OR operation

bitShiftRight

Description

bit ShiftRight-Operation

Definition

bitShiftRight(integer value1, integer value2)

Arguments

integer value1 value 1 for operation

integer value2 value 2 for operation

Return Values

integer result result from operation

getLevenshteinDistance

Description

Get levenshtein distance

Definition

getLevenshteinDistance(string value1, string value2)

Arguments

string value1 value 1 for operation

string value2 value 2 for operation

Return Values

integer distance levenshtein distance

mathEulerToQuaternion

Description

Euler angle to quaternion

Definition

mathEulerToQuaternion(float x, float y, float z)

Arguments

float x x rot (rad)

float y y rot (rad)

float z z rot (rad)

Return Values

float qx x quaternion

float qy y quaternion

float qz z quaternion

float qw w quaternion

I3D

loadI3DFile

Description

Load I3D file

Definition

loadI3DFile(string filename, boolean addPhysics, boolean callOnCreate, boolean verbose)

Arguments

string	filename	i3d filename
boolean	addPhysics	add node to physics
boolean	callOnCreate	call oncreate
boolean	verbose	verbose mode

Return Values

integer rootNodeId id of root node

Fillplanes

createFillPlaneShape

Description

Creates a fill plane shape based on shapeId

Definition

createFillPlaneShape(integer shapeId, string shapeName, float volume, float deltaMax, float maxSurfaceAngle, float maxPhysicalSurfaceAngle, float maxSubDivEdgeLength, boolean createSidePlanes)

Arguments

integer	shapeId	shape id
string	shapeName	name of shape
float	volume	volume of the fill plane shape (m^3)
float	deltaMax	value of how much the fill plane can be excessed (in meter)
float	maxSurfaceAngle	maximum surface angle (geometric tuning value in radian)
float	maxPhysicalSurfaceAngle	maximum physical surface angle (this is usually material dependent in radian)
float	maxSubDivEdgeLength	defines the maximum edge-length of the subdivision surface (in meter)
boolean	createSidePlanes	state of side plane creation

Return Values

integer fillPlaneShapeId id of fill plane shape type

fillPlaneAdd

Description

Add material/volume to a fill plane

Definition

fillPlaneAdd(integer fillPlaneId, float dTvolume, float x, float y, float z, float dx1, float dy1, float dz1, float dx2, float dy2, float dz2)

Arguments

integer fillPlaneId id of fill plane

float	dTvolume	delta volume (m^3)
float	x	x position in world space

float	y	y position in world space
float	z	z position in world space
float	dx1	frame x position in world space
float	dy1	frame y position in world space
float	dz1	frame z position in world space
float	dx2	frame x position in world space
float	dy2	frame y position in world space
float	dz2	frame z position in world space

findPolyline

Description

find nearest polyline

Definition

findPolyline(integer fillPlaneId, float x, float z)

Arguments

integer	fillPlaneId	id of fill plane
float	x	x position in local space
float	z	z position in local space

Return Values

integer polyLineId id of polylineid

getFillPlaneHeightAtLocalPos

Description

Get the fill plane height at a specific position

Definition

getFillPlaneHeightAtLocalPos(integer fillPlaneId, float x, float z)

Arguments

integer	fillPlaneId	id of fill plane
float	x	x value of position
float	z	x value of position

Return Values

float height fillplane height at given position

setFillPlaneMaxPhysicalSurfaceAngle

Description

Set fill plane physical surface angle

Definition

setFillPlaneMaxPhysicalSurfaceAngle(integer fillPlaneId, float physicalSurfAngle)

Arguments

integer	fillPlaneId	id of fill plane
float	physicalSurfAngle	value of physical surface angle (in radian)

setPolylineTranslation

Description

Translate polyline

Definition

setPolylineTranslation(integer fillPlaneId, integer polyLineIdx, float dx, float dz)

Arguments

integer	fillPlaneId	id of fill plane
integer	polyLineIdx	polyline index
float	dx	x value of translation
float	dz	z value of translation

ModDownloadManager

drawDebugArrow

Description

Render an arrow. Only use for debug rendering

Definition

drawDebugArrow(float x, float y, float z, float dirX, float dirY, float dirZ, float tangX, float tangY, float tangZ, float r, float g, float b)

Arguments

float x	x position
float y	y position
float z	z position
float dirX	direction x coordinate
float dirY	direction y coordinate
float dirZ	direction z coordinate
float tangX	tangential x direction
float tangY	tangential y direction
float tangZ	tangential z direction
float r	red color component [0, 1]
float g	green color component [0, 1]
float b	blue color component [0, 1]

drawDebugLine

Description

Render a line. Only use for debug rendering

Definition

drawDebugLine(float x0, float y0, float z0, float r0, float g0, float b0, float x1, float y1, float z1, float r1, float g1, float b1)

Arguments

float x0	start x position
float y0	start y position
float z0	start z position
float r0	start red color component [0, 1]
float g0	start green color component [0, 1]
float b0	start blue color component [0, 1]
float x1	end x position
float y1	end y position
float z1	end z position
float r1	end red color component [0, 1]
float g1	end green color component [0, 1]
float b1	end blue color component [0, 1]

drawDebugPoint

Description

Render a point. Only use for debug rendering

Definition

drawDebugPoint(float x, float y, float z, float r, float g, float b, float a)

Arguments

float x	x position
float y	y position
float z	z position
float r	red color component [0, 1]
float g	green color component [0, 1]
float b	blue color component [0, 1]
float a	alpha color component [0, 1]

print

Description

Print to console

Definition

print(any type arg1, any type ...)

Arguments

any type arg1	a value
any type ...	another value

printCallstack

Description

print callstack

Definition

printCallstack()

source

Description

Source script file

Definition

source(string filename, ref environment)

Arguments

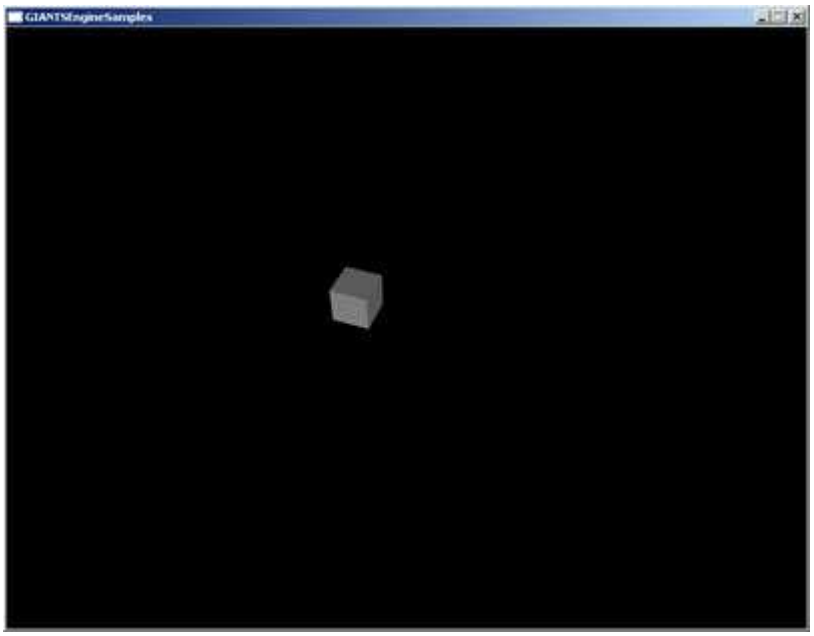
string filename	name of script file
ref environment	ref to custom environment

Tutorials

Tutorial 1 - Basic setup and loading i3d files

The goal of this tutorial is to teach you the very basics of the GIANTS engine. You simply load an i3d with a cube in it and then rotate the cube by using LUA script.

Sample name: SampleI3DLoading
Script source code file: *sample/i3dLoading/main.lua*



The goal of this tutorial is to show you

- how the engine uses ids
- how to load i3d files
- how to access objects within a loaded i3d
- what the base callback functions of the engine are
- how to access keyboard commands

New functions in this tutorial

init()
keyEvent()
update()
rotate()
getRootNode()
loadI3DFile()
link()
getChildAt()
print()
requestExit()

The most important script file is your main.lua. It is a lua file that is loaded and executed when your exe file is started.

The first thing you need to know is how the engine exactly knows where your main.lua is. Well there has to be an xml file in the same folder as your exe with exactly the same name as your exe. In this case it is named SampleI3DLoading.xml. Open it with a text editor.

```
<?xml version="1.0" encoding="iso-8859-1" standalone="no" ?>
<startup>
```

```
 <cmdline>game.exe -script sample/i3dLoading/main.lua</cmdline>
</startup>
```

In the third line you see where your main.lua is located. If you want to move your main.lua, you have to change the path in this xml as well.

So, let's have a look at this main.lua, open it with any text editor. There are 5 essential functions: init(), mouseEvent(), keyEvent(), update() and draw(). Let's look at these functions in detail:

init()

init() is executed once when you start your exe. Let's go through the function line by line:

```
local worldRootNode = getRootNode();
```

The variable worldRootNode is defined and assigned with the result of a function called getRootNode();

If you meet a new function and you don't exactly know what it does, simply look here and search for the function. Then you get detailed information about the function. In the tutorials those function descriptions are always blue.
Here is what you find if you search for the getRootNode function:

getRootNode

Description

Get scenegraph root node

Definition

function getRootNode()

Return Values

integer rootNodeId id of the root node

The descriptions of the functions are quite self-explanatory and can be really helpful.

It's important to see that the GIANTS engine works with ids also called handles. This function returns the id from the root node of your root scene graph.

Your entire world is attached to this root node. Each object you want to see later on must be a child of this root node.

Now we need to load the i3d that contains the cube we want to rotate later on.

```
local sceneToLoad = loadI3DFile("sample/i3dLoading/cube.i3d");
```

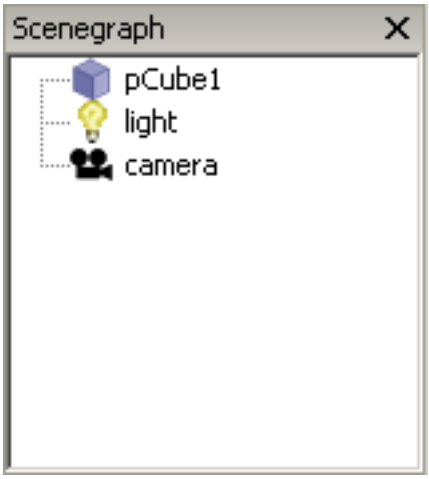
With this function, you load an i3d file, and its id (which is an integer) is returned. Now that we've got both things, the worldRootNode and the i3d we want to load, we have to link them together, which is the next line of code.

```
link(getRootNode(), sceneToLoad);
```

Now we want the cube's id to rotate. Be aware that the i3d we've already loaded is the whole scene which contains not only the cube, but also a light and the camera. So now we need to know which id the cube has. Since the cube is a child of the i3d we can use the following function:

```
cubeId = getChildAt(sceneToLoad, 0);
```

First you name the i3d that you want the child from and then you say which child you want. 0 is the first child, 1 is the second. You need to know which child is your cube. Just open the cube .i3d with the Editor and you can see that the cube is the first position in your scenegraph (reading from top down)



A very important thing you need in programming are prints. The function needed here is simply called print(). The thing you want to print are the parameters. Be careful with strings, they always need quotes. You can link strings and variables by using .. as shown in the example.

print("this is a print: cube id: " .. cubeId);

You can see the printed text in the in game console. Just press tilde '~' to make the console visible.

Now we have all the information we need, but one thing remains to be done. Since the camera is located right in the center (inside the cube actually) we won't be able to see anything. So we grab the camera and move it a bit. With the getCamera() function we first get the id of the camera, because the setTranslation function needs to know which id has to be translated. The next parameters are the desired x, y, z values.

setTranslation(getCamera(), 5, 10.0, 10.0);

Then we rotate the camera so that we can see our cube nicely. The function is similar to the setTranslation but you define the rotations around the three axes of course. You need a radian value, that's why the functions math.rad() are used. They convert a degree value into a radian value.

setRotation(getCamera(), math.rad(-45), math.rad(20), 0);

mouseEvent()

We don't use this function right now, but of course you can access the mouse commands with it.

```
function mouseEvent(posX, posY, isDown, isUp, button)
end;
```

keyEvent()

This function is used to access any keyboard commands. In this case, we only want to know if the escape key is pressed. If this is the case, then the program closes with the command requestExit();

```
function keyEvent(unicode, sym, modifier, isDown)
 -- Check for escape key
 if sym == Input.KEY_esc and isDown == true then
  -- Request quit
  requestExit();
```

```
 end;
end;
```

update(dt)

The update function is a function that is called after every frame. The parameter dt is the time that has passed since the last update call in milliseconds.

So, this is the place where we can finally make our cube rotate. For this we use the rotate(dt) function. It needs the id of the object that should rotate, in this case the cubeId, and also the rate of rotation on the x, y and z-axis. We just let it rotate on the x-axis with the value of 0.08 degree per millisecond.

```
function update(dt)
 rotate(cubeId, math.rad(0.08)*dt, 0, 0);
end;
```

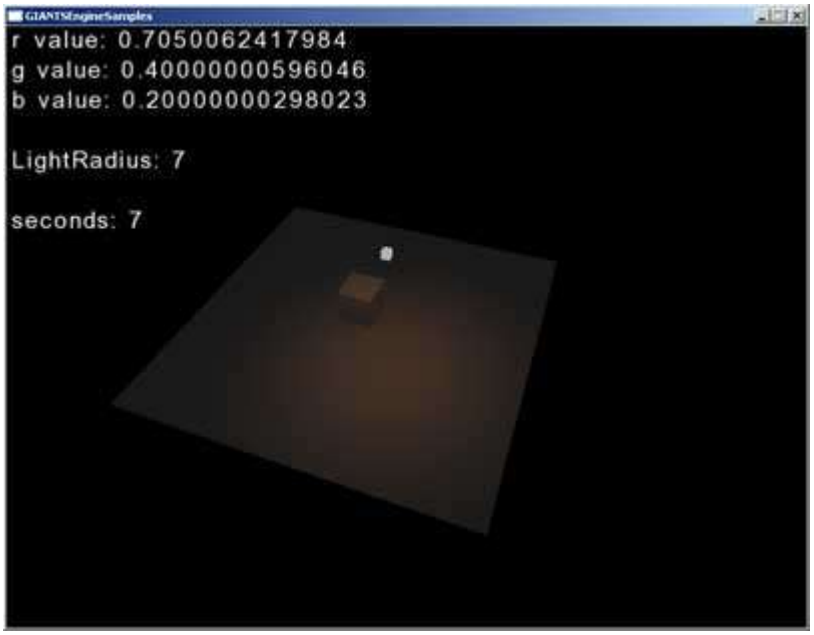
draw()

The draw function is needed to draw something onto the screen. This will be discussed later on.

```
function draw()
end;
```

Tutorial 2 - Light functions, global time and rendering text

w**Sample name:** SampleLighting
Script source code file: *sample/lighting/main.lua*



The goal of this tutorial is to show you

- how to manipulate lights in your scene
- how to handle time
- how to print a text on the screen

New functions in this tutorial

setLightDiffuseColor()
getLightDiffuseColor()
setLightRange()
getLightRange()
renderText()

The first thing you might notice is that there are two new additional variables that we're going to use:

```
local globalTime = 0;
local lightRadius = 7;
```

In this, and also in further tutorials, we will just cover the new functions and variables. So, if there is anything you don't understand, have a look at Tutorial 1.

init()

In the init() function we additionally store the id of the point light in the variable pointLightId.

```
pointLightId = getChildAt(id, 2);
```

When you open the i3d file cubeWithPointLight.i3d, you can see that the light is in third position. Since the function getChildAt()starts counting from 0, the argument 2 points to the third child.

Set the light radius to an initial value of 7.

```
setLightRange(pointLightId, lightRadius);
```

keyEvent()

Here are two more keys: Key 1 and Key 2. When they are pressed, the functions decreaseLightRadius and increaseLightRadius are called.

```
if sym == Input.KEY_1 and isDown == true then
 decreaseLightRadius();
end;

if sym == Input.KEY_2 and isDown == true then
 increaseLightRadius();
end;
```

increase- decreaseLightRadius()
These two functions change the variable lightRadius by 0.5 and then call the lightRadius function with the new value.

```
function increaseLightRadius()
 lightRadius = lightRadius + 0.5
 setLightRange(pointLightId,lightRadius);
end;
```

update(dt)

Two variables are new here:

Why do we need a globalTime here? Well, we want the light to rotate around the cube. Therefore, we need a value that constantly changes in corrrespondence with time. Of course you could also increase a number every frame, but since the framerate isn't constant we also use the time which has passed since the last frame and thus get an exact and constantly growing number that isn't affected by the different framerates. Remember that dt is the time passed between the last and the actual frame in milliseconds. So, if dt is multiplied by 0.001 you get seconds. We also need a distance to the center which is stored in the variable radius.

```
 globalTime = globalTime+dt*0.001;
 local radius = 2;
```

This function handles the translation of the pointLight using the globalTime and the radius 2.

```
 setTranslation(pointLightId, math.sin(globalTime)*radius, 3, math.cos(globalTime)*radius);
```

We also change the color of the diffuse color of the light. This is done with the setLightDiffuseColor() function. As arguments it uses the id, and then the three values for red, green and blue as floats between 0 and 1. As r we use the sinus of the globalTime.

```
 setLightDiffuseColor(pointLightId, math.sin(globalTime), 0.4, 0.2);
```

When you execute the program, you see that there is a white box where the point light is. This may seem a bit odd since you probably don't want this cube to be visible at all. When you open the file cubeWithPointLight.i3d and click on the plus left of the light in the scene graph, you see that we have attached a little cube to the light. We have done this to to give you a better visual feedback on where the pointLight is. On top of that, it shows also another important thing: as a child it inherits the translations of the parent and thus also moves with it.

draw()

Like the update function, the draw function is executed after every frame.

First we store the rgb values of the pointLight using the getLightDiffuseColor() function. It returns three different values. So be careful, you need to separate the different variables by commas!

local r, g, b = getLightDiffuseColor(pointLightId);

Then we use the function renderText() to render the result directly onto the screen.

renderText

Description

Render text to viewport

Definition

function renderText(float x, float y, float size, string str)

Arguments

float	*x*	x position [0, 1]
float	*y*	y position [0, 1]
float	*size*	font size
string	*str*	string to print

When using renderText you can also weld parts together with the .. operator to concatenate variables and strings like in the example.

```
renderText(0,0.95,0.05,"r value: " .. r);
renderText(0,0.90,0.05,"g value: " .. g);
renderText(0,0.85,0.05,"b value: " .. b);
```

The next line stores the value of the getLightRange() function in the lightRadius variable

```
local lightRadius = getLightRange(pointLightId);
```

and then the variable is rendered onto the screen:

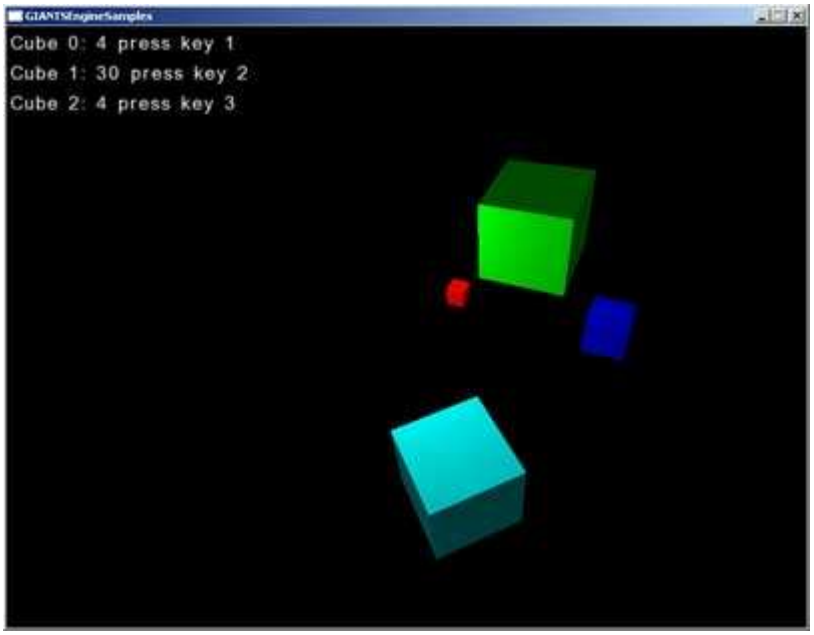
```
renderText(0,0.75,0.05,"LightRadius: " .. lightRadius);
```

With the last line we simply render the globalTime. Since it is a float we use math.floor to get an integer.

```
renderText(0,0.65,0.05,"seconds: " .. math.floor(globalTime));
```

Tutorial 3 - Using user attributes

Sample name: SampleUserAttributes
Script source code file: *sample/userAttributes/main.lua*



The goal of this tutorial is to show you

- how to create user attributes
- how to access user attributes from script
- how to use the onCreate script callback

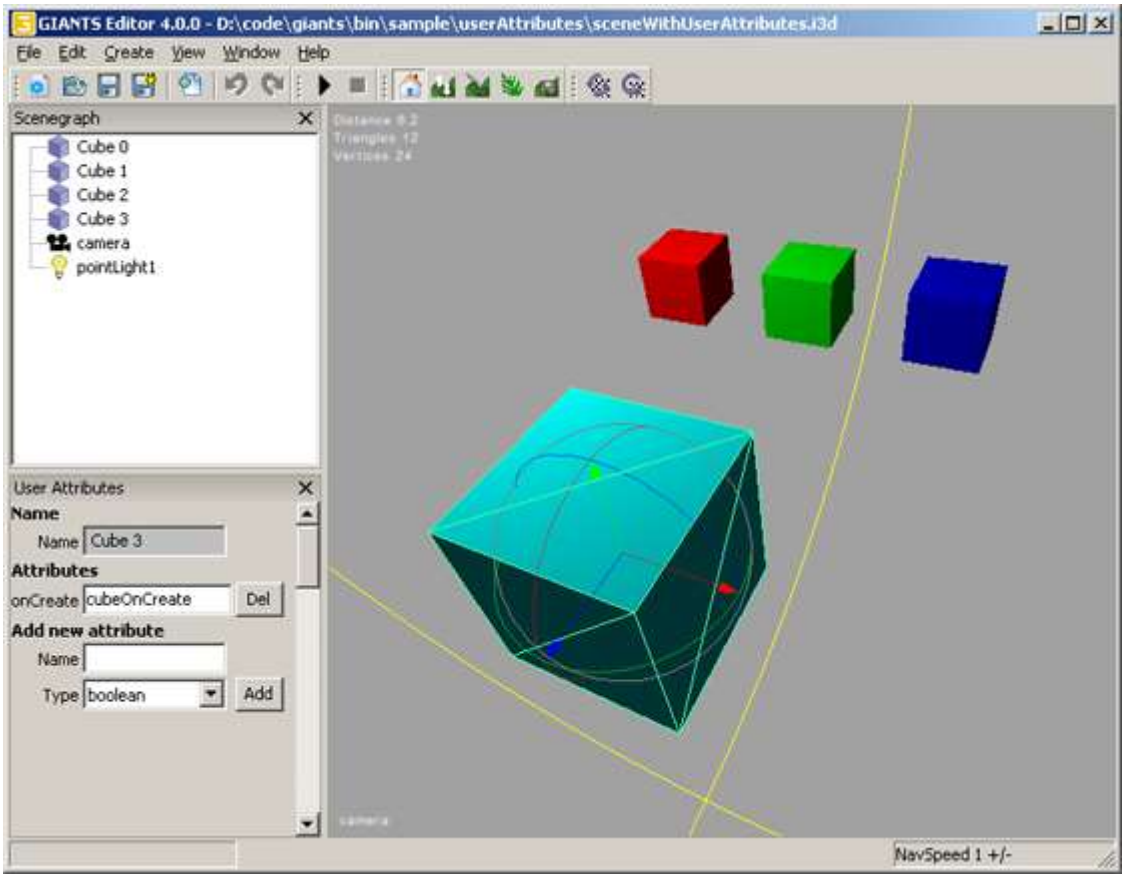
New functions in this tutorial

getUserAttribute()
setUserAttribute()
setScale()
translate()

Open sceneWithUserAttributes.i3d and main.lua which are located in \sample\userAttributes.

Run the sample SampleUserAttributes.exe. You see three cubes with different scales and one cube that is rotating. If you press 1,2 or 3, the related cube is translated onto the y-axis and the number of translations is also displayed.

If you look at sceneWithUserAttributes.i3d in the editor, you will see, that all three cubes in the back have the same size. Open the User Attributes window.



In the User Attributes window you see the name of your object, followed by the user attributes and at the bottom there is the possibility to create new attributes. Select the different objects to see what kind of user attributes they have.

You'll notice that both cubes in the back have two user attributes labeled count and size. The bigger cube in the front only has an onCreate script callback.

How to create new user attributes
To create your own attributes, just enter the name of your attribute and then choose the according type of your attribute. You can choose whether your attribute is a boolean, integer, float, a string or a script callback. Then hit add and your attribute will appear.

Let's have a look at the LUA code in main.lua. The very first line of code is the creation of an empty array called cubeIdArray.

init()

You should be familiar with the first two lines, otherwise have a look at Tutorial 1.
What follows is a for loop from 0 to 2 where the array cubeIdArray is filled up with three new arrays containing the cubeIds of the three cubes.

Then the variable size gets filled with the result of the getUserAttributes function. The getUserAttributes function needs the id of the object and the name of the attribute as arguments. Be careful here, the name has to be exactly the same as you have specified in the editor.

Finally, there is a check whether size has a value and then the function setScale is used to scale the cube accordingly. Try to enter new size values in the i3d and then see the result by running the program.

```
for index=0, 2 do
-- Get current cube id and store it for later
```

```
local cubeId = getChildAt(i3dRoot, index);
  cubeIdArray[index] = {};
  cubeIdArray[index].id = cubeId;

  -- Get scale value from user attributes
  local size = getUserAttribute(cubeId, "size");
  if size ~= nil then
    -- Set scale of current cube
    setScale(cubeId, size, size, size);
  end;
end;
```

keyEvent()

First we check if esc is pressed.

Then, if the keys 1,2 or 3 are pressed, the rest of the code is executed. The id of the corresponding cube is stored within the variable cubeId. Then the count of the cube is stored in the variable count by using the return value of the function **getUserAttribute**. If the count isn't nil, the count is increased by one using the **setUserAttribute** function.

The setUserAttribute function can either be used to create new attributes or to overwrite values of existing attributes. It uses the id of the object (cubeId), the name of the attribute("count"), the data type(Integer) and the new value as arguments(count+1). Finally, the corresponding cube is translated onto the y-axis with the value 0.1 using the translate function.

```
if isDown == true then

 -- Check for escape key
 if sym == Input.KEY_esc then
  -- Request quit
  requestExit();
 end;

 -- Check for key 1 to 3
 if sym >= Input.KEY_1 and sym <= Input.KEY_3 then

  -- Get id of current cube
  local cubeId = cubeIdArray[sym-Input.KEY_1].id;

  -- Get count user attribute
  local count = getUserAttribute(cubeId, "count");
  if count ~= nil then
    -- Set count user attribute
    setUserAttribute(cubeId, "count", "Integer", count+1);
    -- Translate cube a bit
    translate(cubeId, 0, 0.1, 0);
  end;

 end;

end;
```

cubeOnCreate(id)

This function isn't called anywhere in the LUA code. So what is it used for? As soon as the cube in the front gets loaded, its attribute script callback causes the function cubeOnCreate to run once. The script also provides the id of the object that caused the script callback. Be careful with the spelling, the function in your LUA script has to be exactly the same as your ScriptCallback in the i3d. What we do in the cubeOnCreate function is quite simple: we store the id in the new variable rotatingCubeId.

```
function cubeOnCreate(id)
  -- Store the id for later
  rotatingCubeId = id;
end;
```

update(dt)

Here we first check if rotatingCubeId exists and then we rotate the cube using the rotate function with the id we stored in the variable rotatingCubeId.

```
function update(dt)

  -- Rotate cube3
  if rotatingCubeId ~= nil then
    rotate(rotatingCubeId, 0, 0.05, 0);
  end;

end;
```

draw()

Here you can see how flexibly you can use the renderText function: using it within a for loop, for welding text and variables as well as for calling functions like the getUserAttributes in this case.

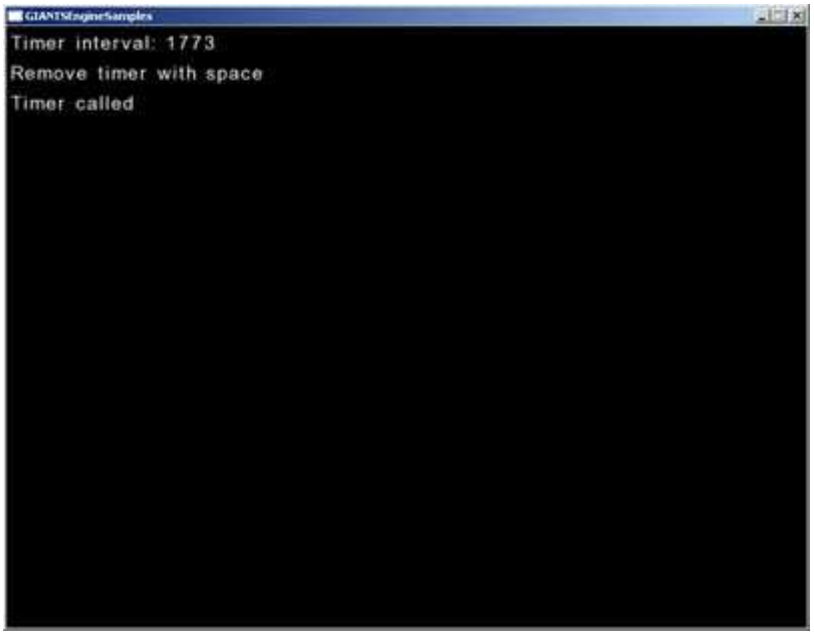
```
function draw()

  for index=0, 2 do
    renderText(0,0.95-index*0.05, 0.04,"Cube " .. index .. ": "
    .. getUserAttribute(cubeIdArray[index].id, "count") .. " press key " .. index+1);
  end;

end;
```

Tutorial 4 - Using timers

Sample name: SampleTimer
Script source code file: *sample/timer/main.lua*



The goal of this tutorial is to show you

- how to create and use timers

New functions in this tutorial

addTimer()
setTimerTime()
removeTimer()

Run the sample SampleTimer.exe. Only three lines of text are rendered. The first shows you a random timer interval, the second is a string that says that you can remove the timer with the space key and finally, there is a line showing a flashing text :"Timer called".

Before we start looking at the code we should get a rough idea about what a timer is, and what functionalities it has.

A timer is a simple object that has a certain set time and the name of the callback function. As soon as the timer is activated, the defined time is counted down to zero. When this happens, the timer calls the callback function. This behavior can be used in various situations, for instance if a trigger shouldn’t react immediately but only after a certain delay.

Open the the main.lua which is located in the directory /sample/timer.

At first, four variables are defined:
timerId = 0;
timerInterval = 0;
renderCount = 0;
timerRemoved = false;

init()

In the init function the first line calls the function setRandomTimerInterval. This function simply generates a random value between 1000 and 5000 for the variable timerInterval.

```
function setRandomTimerInterval()
  timerInterval = math.random(1000, 5000);
end;
```

The variable timerInterval is used immediately because a new timer is created with the line:

timerId = addTimer(timerInterval, "onTimer");

The variable timerId stores the Id of the new timer which is returned by the function addTimer. Here is how this function is described in the documentation:

addTimer

Description

Adds timer callback function

Definition

function addTimer(float time, string timerFunctionCallback, object instance)

Arguments

float	*time*	time in milliseconds
string	*triggerFunctionCallback*	trigger function callback
object	*instance*	instance object (optional)

Return Values

integer timerId timer id

The first argument is the time in milliseconds and the second argument is the function that is called when the time reaches zero. In this case, the variable timerInterval defines how long the timer will wait, until it calls its callback function. After this time, the function onTimer is called.

keyEvent()

If you hit the space key the timer is removed and the boolean timerRemoved is switched to true.

```
if sym == Input.KEY_space then
  -- Remove timer
  removeTimer(timerId);
  timerRemoved = true;
end;
```

onTimer()

This function is called as soon as the timer reaches zero.

The first line defines the variable renderCount to be 100. This defines the number of frames the line "timer called" is rendered in the draw function later on.

The second line defines a new timerInterval by calling the setRandomTimerInterval function again.

The third line sets a new timer time with the new timerInterval time. The setTimerTime function needs the id of the timer (timerId) and the new time (timerInterval). After the timer reaches zero again, the onTimer function is called again. You can remove a timer within the callback function by returning false.

```
-- Timer callback
function onTimer()
  renderCount = 100;
  setRandomTimerInterval();
  setTimerTime(timerId, timerInterval);
  return true;
end;
```

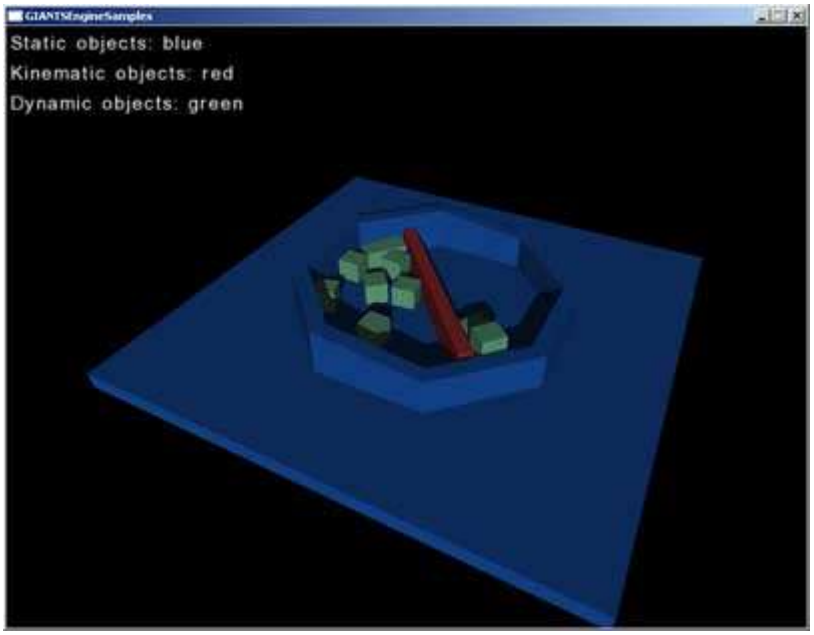
draw()

First we have an if-else construct that handles the two states, i.e. whether the timer is removed or not. Depending on the state of the boolean timerRemoved, the different renderText functions are executed.

The string "timer called" is only called if the variable renderCount is not zero. Remember that we set the renderCount to 100 in the onTimer function which is the trigger callback of the timer. With each frame the renderCount is decreased by 1.

Tutorial 5 - Objects with physics

Sample name: SamplePhysics
Script source code file: *sample/physics/main.lua*



The goal of this tutorial is to show you

- the difference between rigid body and no rigid body
- the difference between static, kinematic and dynamic objects
- how to create objects with physical behavior

Basically, there are two different types of objects available. Objects with rigid body and objects without rigid body.

No rigid body

Objects with no rigid body are only rendered. They can be moved freely and they don’t cause any collision, thus other objects can move through them. These objects only use a minimum of the engine's performance. If you export a new object from Maya without having changed the attributes, the rigid body is activated by default. You should turn it off in the editor, when you don’t need it.

Rigid body

Objects with rigid body behave like real objects. If another object with rigid body tries to go trough it, there will be a collision. Since the used up power of this behavior has to be calculated, it should only be used when necessary. There are three rigid body types:

Static

These objects offer collision, but they can’t be moved, nor do they react to gravity.

Kinematic

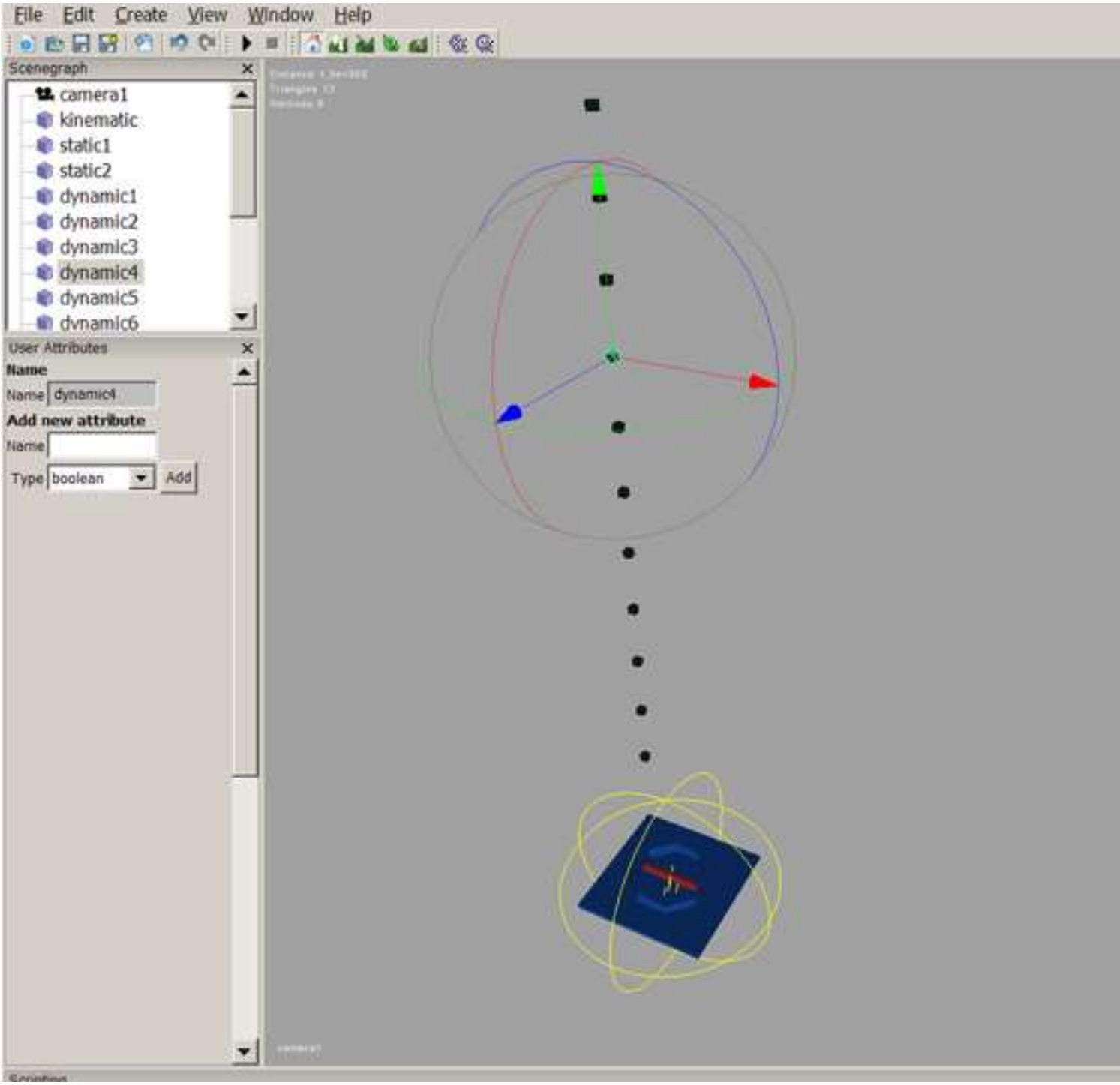
These objects offer collisions with dynamic objects and can be moved in a predefined way. E.g. if you have a train that goes from A to B along a predefined path and a collision occurs with a kinematic object, the train will simply continue its movement, while the dynamic object will be pushed away. If the same train hits a static or another kinematic object, nothing will happen.

Dynamic

This kind of objects react dynamically to collisions and can't be moved manually. The movement of dynamic objects is a result of external forces and isn’t predefined.

Run SamplePhysics.exe.

Since it is self-explanatory, just open the sceneWithPhysics.i3d which is located in /sample/physics to see how it is done.



You probably need to zoom out a bit to see the whole scene.

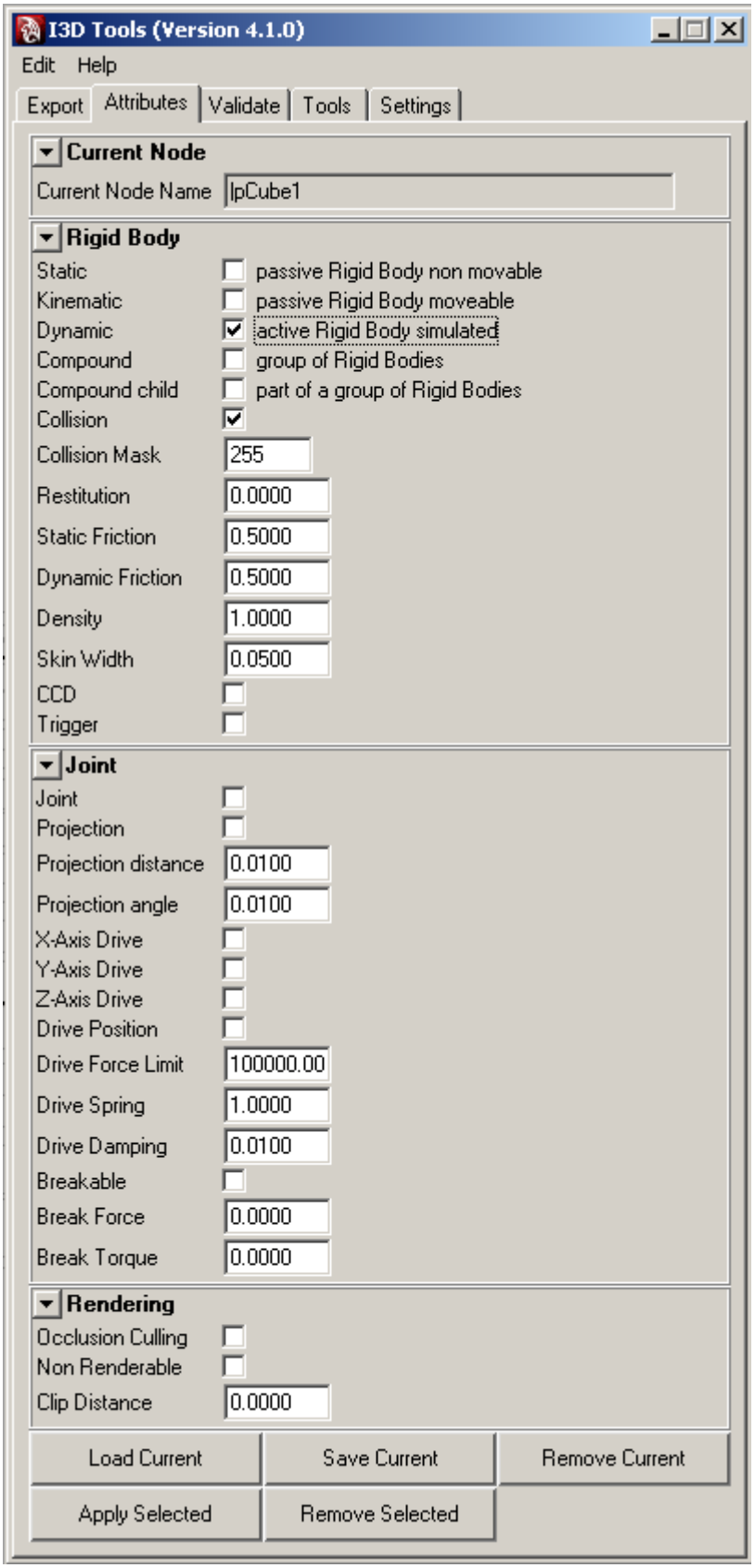
Select the different objects and look at the Rigid Body Type in the red marked area. In this scene all objects are rigid bodies, but the Rigid Body Type is different, depending on how their physics should react.

If you want to select a cube which is within the sphere of the multi-selection-tool, you'll first have to select something that is a bit further away.

Try to change the Rigid Body Type of the different objects and see what happens, when you run the

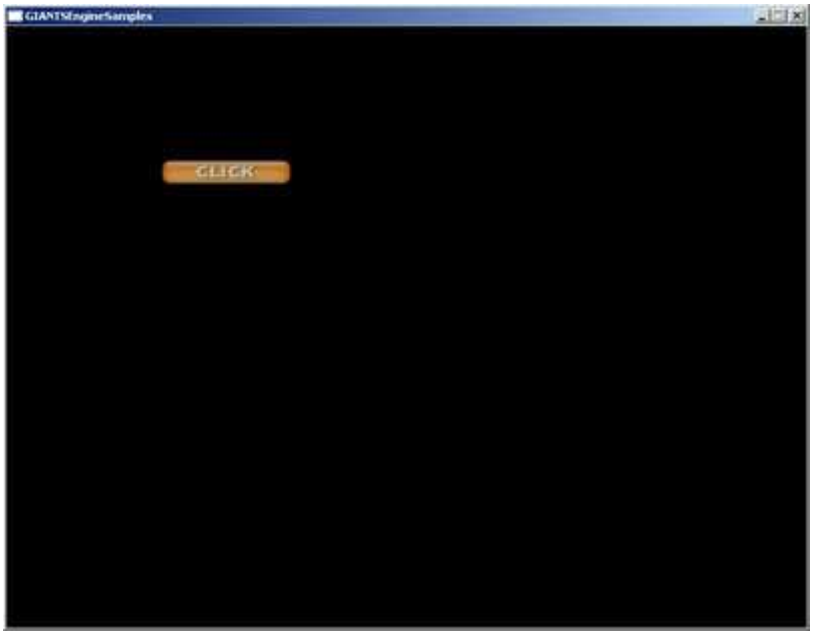
program. To change it, simply open the dropdown menu and select the desired Rigid Body Type. Hit enter, and don’t forget to save your scene.

Instead of defining the physical behavior of an object in the editor, you could specify these settings directly in Maya. Launch the i3d exporter and open the attributes panel where you can see all the Rigid Body options available. Just select the desired mesh, click *Load Current* and then set the Rigid Body settings as you wish. Don’t forget to click *Save Current* when you've done it. If you export the mesh as an i3d file, the rigid body settings are already included. This method has the advantage that attributes defined in Maya remain in the saved scene file. If you export the same scene more than once, you might save time by defining it once in Maya instead of redoing it every time in the editor.


I3D Tools (Version 4.1.0)
Edit
Help
Export
Attributes
Validate
Tools
Settings
Current Node
Current Node Name
lpCube1
Rigid Body
Static
passive Rigid Body non movable
Kinematic
passive Rigid Body moveable
Dynamic
active Rigid Body simulated
Compound
group of Rigid Bodies
Compound child
part of a group of Rigid Bodies
Collision
Collision Mask
255
Restitution
0.0000
Static Friction
0.5000
Dynamic Friction
0.5000
Density
1.0000
Skin Width
0.0500
CCD
Trigger
Joint
Joint
Projection
Projection distance
0.0100
Projection angle
0.0100
X-Axis Drive
Y-Axis Drive
Z-Axis Drive
Drive Position
Drive Force Limit
100000.00
Drive Spring
1.0000
Drive Damping
0.0100
Breakable
Break Force
0.0000
Break Torque
0.0000
Rendering
Occlusion Culling
Non Renderable
Clip Distance
0.0000
Load Current
Save Current
Remove Current
Apply Selected
Remove Selected

Tutorial 6 - Rendering overlays

Sample name: SampleOverlay
Script source code file: *sample/overlay/main.lua*



The goal of this tutorial is to show you

- how overlays work
- how to create overlays

New functions in this tutorial:

createOverlay()
setOverlayColor()
renderOverlay()

When you run sampleOverlay.exe you see an overlay displayed on the screen. If you click on it, its position changes randomly.

Open main.lua which is in /sample/overlay to take a look at the code.

First there are three variables: overlayId holds the id of the overlay, overlayX and overlayY are used to define the position of the overlay.

init()

In the first line, the variable overlayId is filled with the result of the function createOverlay which returns the id of the newly created overlay. The documentation describes the createOverlay functions as follows:

createOverlay

Description

Create overlay object

Definition

function createOverlay(string overlayName, string textureFilename)

Arguments

string overlayName overlay name
string textureFilename texture file name

Take note that you have to use the relative path to your file.

overlayId = createOverlay("overlay", "sample/overlay/overlay.png");

The second line consists of the setOverlayColor function.
Try to comment it out by adding -- in front of the line. Save the script and run the program. You see that this sample also works without the setOverlayColor function. It isn't needed to display an overlay, but it gives you the possibility to change its appearance. Here is how it is described in the documentation:

setOverlayColor

Description

Set overlay color

Definition

function setOverlayColor(integer overlayId, float red, float green, float blue, float alpha)

Arguments

integer	*overlayId*	overlay id
float	*red*	red value
float	*green*	green value
float	*blue*	blue value
float	*alpha*	alpha value

The used arguments in the example result in white with an opacity of 0.75.

setOverlayColor(overlayId, 1, 1, 1, 0.75);

Try the arguments 1, 0, 0, 0.75, to see the difference.

mouseEvent()

This function runs every time you move the mouse or click on it. If the mouse is inactive this function isn't executed. It provides the position of the cursor, and the state of the buttons which can be used within the function.
We first check, if the state of a mouse button is down, which is done by the isDown part of the if-construct. The rest of the if-construct is a simple hit test, it checks whether the cursor is within the overlay or not. If the if-construct returns true, you have clicked on the button. Then the overlayX and overlayY values are overwritten with a random number between 0.05 and 0.8.

```
function mouseEvent(posX, posY, isDown, isUp, button)
  -- Check if overlay has been clicked
  if isDown and posX > overlayX and posX < overlayX+0.16 and
    posY > overlayY and posY < overlayY+0.04 then
    overlayX = math.random(50, 800)/1000;
    overlayY = math.random(50, 900)/1000;
  end; end;
```

draw()

What happens here is essential : try commenting out the renderOverlay function once. If you run the sample now, nothing will be rendered at all. This function is needed to render an overlay. This allows you to display or hide components (e.g. menu options) depending on the state of your program.

Here is how the function is described in the documentation:

renderOverlay

Description

Render overlay

Definition

function renderOverlay(integer overlayId, float x1, float y1, float width, float height)

Arguments

integer	*overlayId*	overlay id
float	*x*	x position [0,1]
float	*y*	y position [0,1]
float	*width*	width [0,1]
float	*height*	height [0,1]

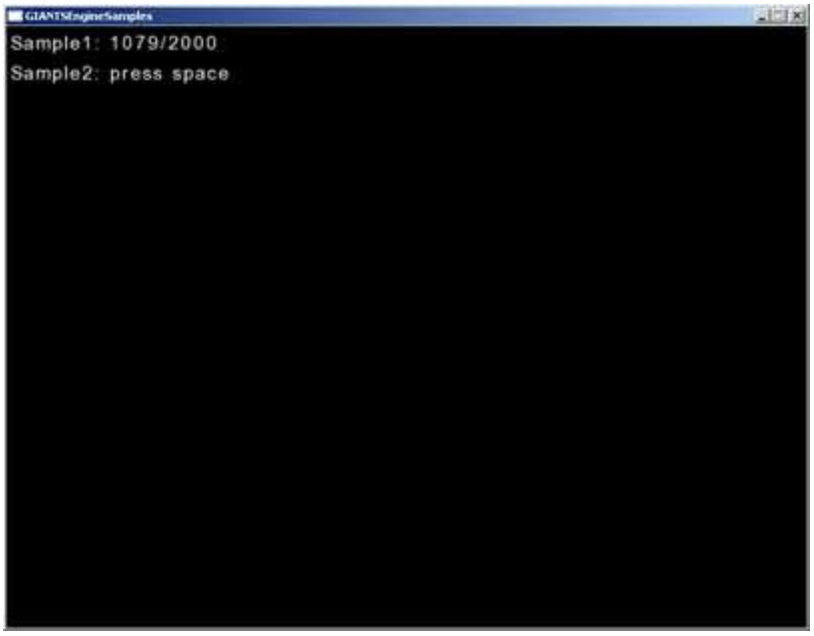
A width and height of 1 means that the overlay has exactly the size of the screen.

Since overlayX and overlayY change when you click on the overlay, it is instantly rendered in the new position.

```
function draw()
  -- Render overlay
  renderOverlay(overlayId, overlayX, overlayY, 0.16, 0.04);
end;
```

Tutorial 7 - Playing audio samples

Sample name: SampleAudio
Script source code file: *sample/audio/main.lua*



The goal of this tutorial is to show you

- how to use audio samples
- the difference between 2d and 3d sounds

New functions in this tutorial

createSample()
loadSample()
playSample()

Run sampleAudio.exe. Every two seconds you should hear a "pling" and if you press space you should hear a water splash.

Open main.lua which is in /sample/audio to see the code of this example.

First there are three variables: time, sample1Id and sample2Id

init()

The first line creates a sample using the createSample function and stores the id in the variable sample1Id. The only argument here is the name of the sample.

```
sample1Id = createSample("sample1");
```

The sample is like an empty shell which needs filling with an audio file. That's what the second line does. It needs the id of the sample and the filename as arguments. The last argument is a boolean and defines whether your sound is 3D or not.

3D sounds can be placed in your scene. They are surrounded by a sphere which defines the distance over which the sound can be heard. The closer you get to the sound the louder it is.

A 2D sound simply plays at full volume when it is triggered.

For the filename you need to indicate the relative path.

loadSample(sample1Id, "sample/audio/sample1.wav", false);

keyEvent()

If you press the space key, the sample2 is played via the playSample function. Here is how this function is described in the documentation:

playSample

Description

Play sample object

Definition

function playSample(integer objectId, integer loops, float volume, float offset)

Arguments

integer	*objectId*	id of sample object
integer	*loops*	number of loops [0,n] 0 = endless looping
float	*volume*	volume [0,1]
float	*offset*	offset in milliseconds

The offset defines from which position within the sample it starts playing. E.g. if you define an offset of 3000, the first three seconds of your sample are cut off the first time the sample gets played.

```
if sym == Input.KEY_space then
  playSample(sample2Id, 1, 1, 0);
end;
```

update()

First the time is increased and if it is larger than 2000 milliseconds the sample1 is played and the time is reset to 0.

```
function update(dt)

  time = time + dt;

  if time > 2000 then
    playSample(sample1Id, 1, 1, 0);
    time = 0;
  end;
end;
```

Exporter

Autodesk Maya I3D Exporter

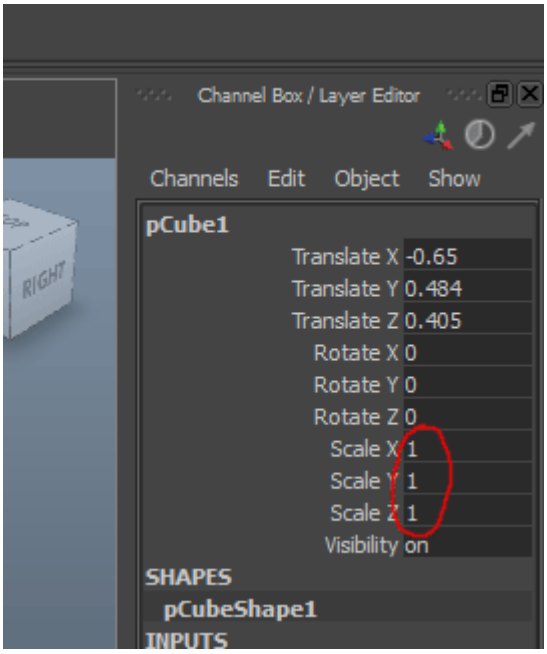
To generate i3d files of your 3d-models you can use the exporter plugin for Autodesk Maya. The GIANTS Editor and the GIANTS Engine can only load i3d files. This section of the documentation will show you, how to export i3d files with the I3D exporter plugins in Autodesk Maya.

The first thing you probably want to do is to install the exporter plugins in Autodesk Maya.

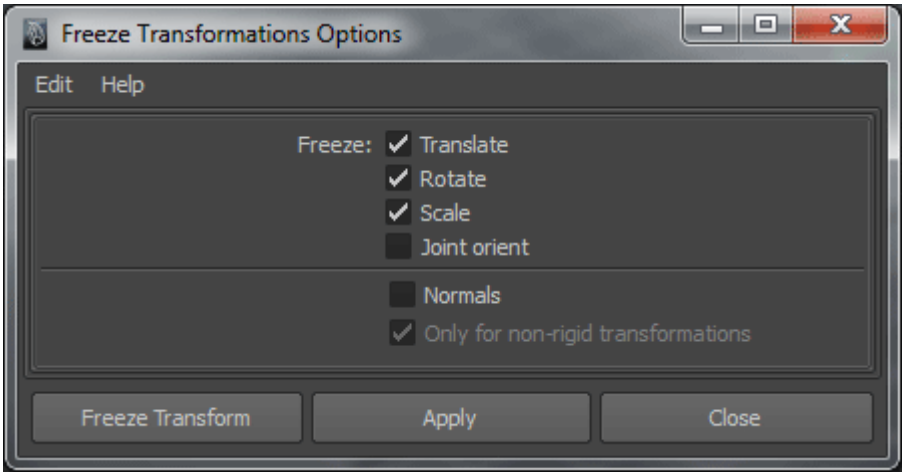
Preparing your 3d-model for export

Before you can actually use the exporter, it's wise to check certain things with your 3d-model in Maya which can cause issues later in the editor or the engine.

In the Channel-Box you see the transformations of your selected object.



The ScaleX, ScaleY and ScaleZ of your object should be 1 1 1.
If you just plan to use this object as a static mesh or with no physics at all you could export it with any scale you want, but if you want to simulate your object in the engine (as a dynamic or kinematic rigid body object) the scale must be 1 1 1 otherwise the physics simulation will produce incorrect results.

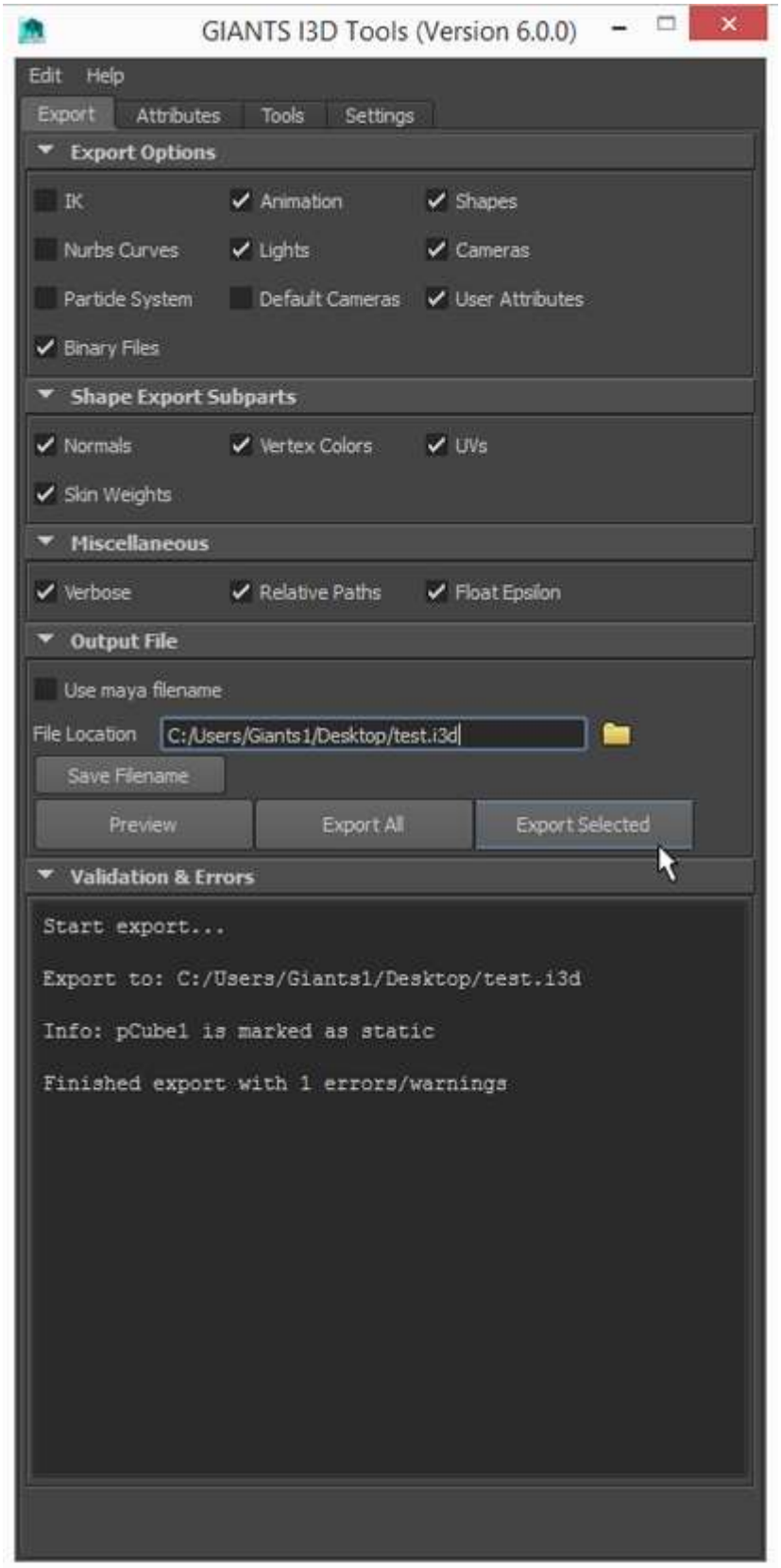


If you have an object with scale, you can easily get rid of the scale. Go to the Modify-Panel, and select the option box right of the Freeze Transformations. The Freeze Transformations dialog window opens. You can select the checkbox scale and hit apply.

Note: Please refer to the artwork guide for further information about asset conditioning for the engine.

I3D Exporter Usage

Start the exporter dialog with a click on the I3D exporter icon you have created on your custom shelf (*see Installing I3D exporter plugins*)



Export Panel
The export panel is quite self-explanatory. In the export options part you can include and exclude parts from exporting (IK, Animation, Shapes, Nurbs Curves, Lights, Cameras, Particle Systems and/or Default Cameras). The Shape Export Subparts section allows you to control which subshape attributes are exported (Normals, vertex Colors, Texture Coordinates and/or Skin Weights).

The miscellaneous section has this options:

- **Verbose**, display verbose information in output window during the export
- **Relative Paths**, generate relative paths for texture files
- **Float Epsilon**, truncate values within epsilon range to zero. Epsilon: [-1e-5, 1e-5]

Buttons

- **Preview**, exports whole scene to a temporary file and starts the editor
- **Export All**, exports the whole scene
- **Export**, exports only currently selected objects

Attributes Panel

In the attributes panel you can load and save attributes of your objects those attributes will be saved into the Maya file and are translated into the correct i3d attributes when you export to a i3d file.

The first thing you have to do here is, that you must press the "Load Current" button at the bottom in order to load the attibutes of your object. Otherwise everything is unchecked and is not showing the attributes that are currently active on you object. So be sure, to hit this button first.

Current Node
Here you can see the name of your current working object.

Rigid Body

This section handles all the attributes regarding to physics rigid bodies.

Joint
Here you can define your joint-attributes in detail. One thing you might miss here are the joint limits, they must be defined within Maya, since your object can have many joints with different limits. You can define the limit of a joint angle by the limit informations of your transform-object.

Rendering

- **Occlusion Culling**, Objects that are entirely covered by other objects may be culled and thus not rendered at all. This option can increase the rendering performace in the engine if you have big objects containing much smaller objects, for instance a house. If you activate the occlusion culling on the root-node of the house all its childs are also not rendered, if the house is complete covered behind other objects. However, you should not apply this attribute on too many small objects because it is one more task for the engine.
- **Non Renderable**, With this option a objects will not be rendered at any time. Use this option for collision geometry. If you have complex objects with sub-objects attached, be careful with this checkbox since this attribute is also going to affect all the attached children.
- **Clip Distance**, This value defines how far the object is still going to be rendered. If you have a large scene and tons of objects this is a powerful method to keep your framerate high.

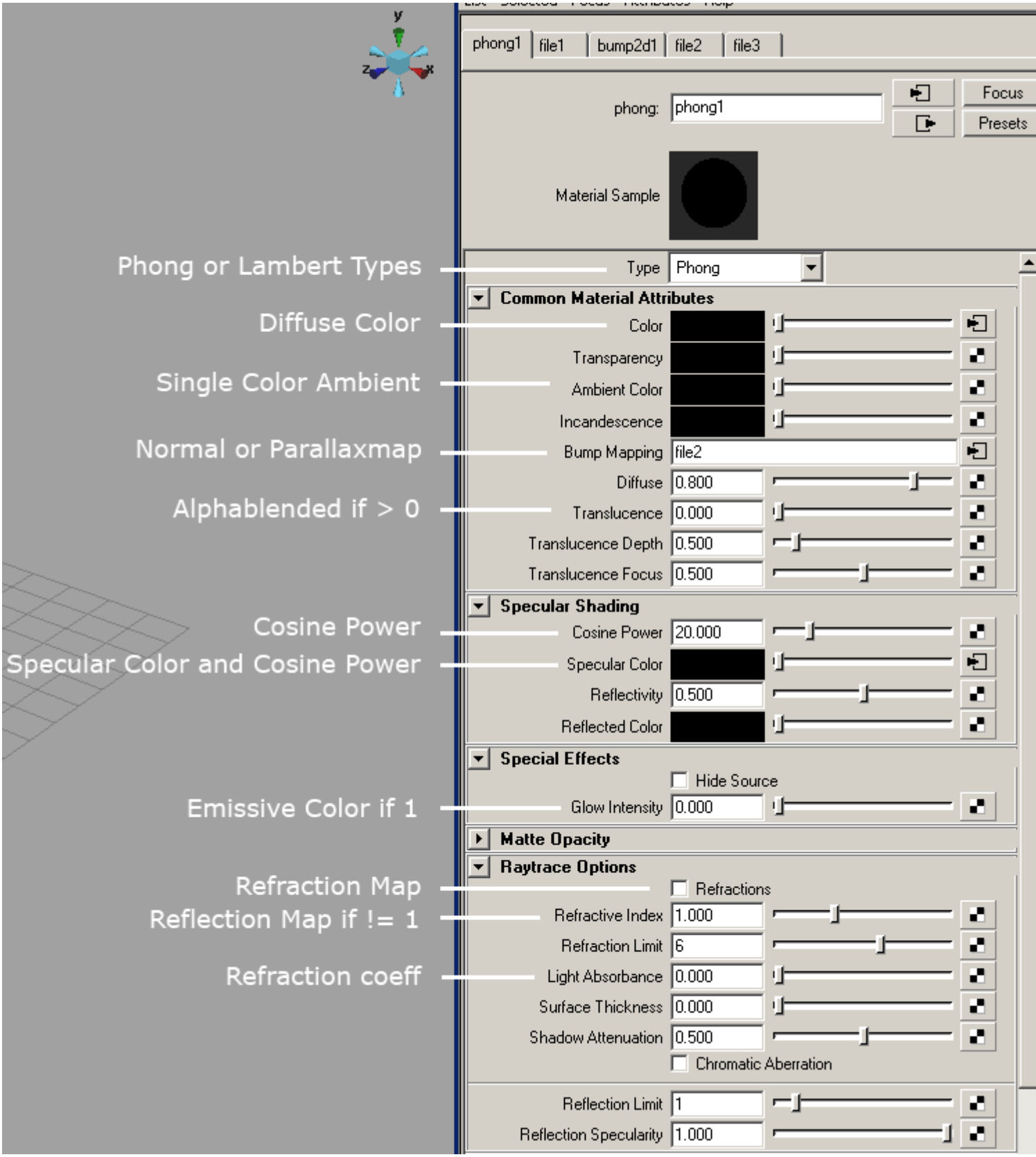
After you have defined all your attributes, you have to hit the "Save Current" button to save your attributes. If you have lots of objects with the same attributes, you can select them all, and then use apply selected to apply the current settings to all the selected objects. Remove current resets the attributes to the default values and if you want do reset multiple objects, you can select them all and hit "Remove Selected"

Validate Panel
With this panel you can validate if one of your objects have an incorrect pivot. In Maya your pivot can have a local and a world space attribute which you can find in the attribute editor on the right side. The x y and z values of both, local and world-space have to be 0 0 0 otherwise you will get a warning. You can fix incorrect pivots with the FreeezeToPivot option in the tools panel.

Tools
If you have validated your objects and a local pivot was not set to 0 0 0, you can quickly fix affected objects with the FreeezeToPivot button.

Material export options

Mapping between Maya and i3d material attributes



Material Attributes

- **Custom Shader:** Use the attribute name 'customShader' with a string type to add a custom shader to the material.
  The value specifies the path of the custom shader xml file relative to the maya file.
- **Custom Shader Parameter:** Use the attribute name 'customParameter_<parameterName>' with a string type to specify the value of a custom shader parameter. The values should be space separated. <parameterName> is the name of the parameter as specified in the custom shader.

- **Custom Shader Texture:**: Use the attribute 'customTexture_<textureName>' with a string type to specify the texture of a custom shader texture.
  The value specifies the path of the texture file relative to the maya file.
  <textureName> is the name of the texture as specified in the custom shader.

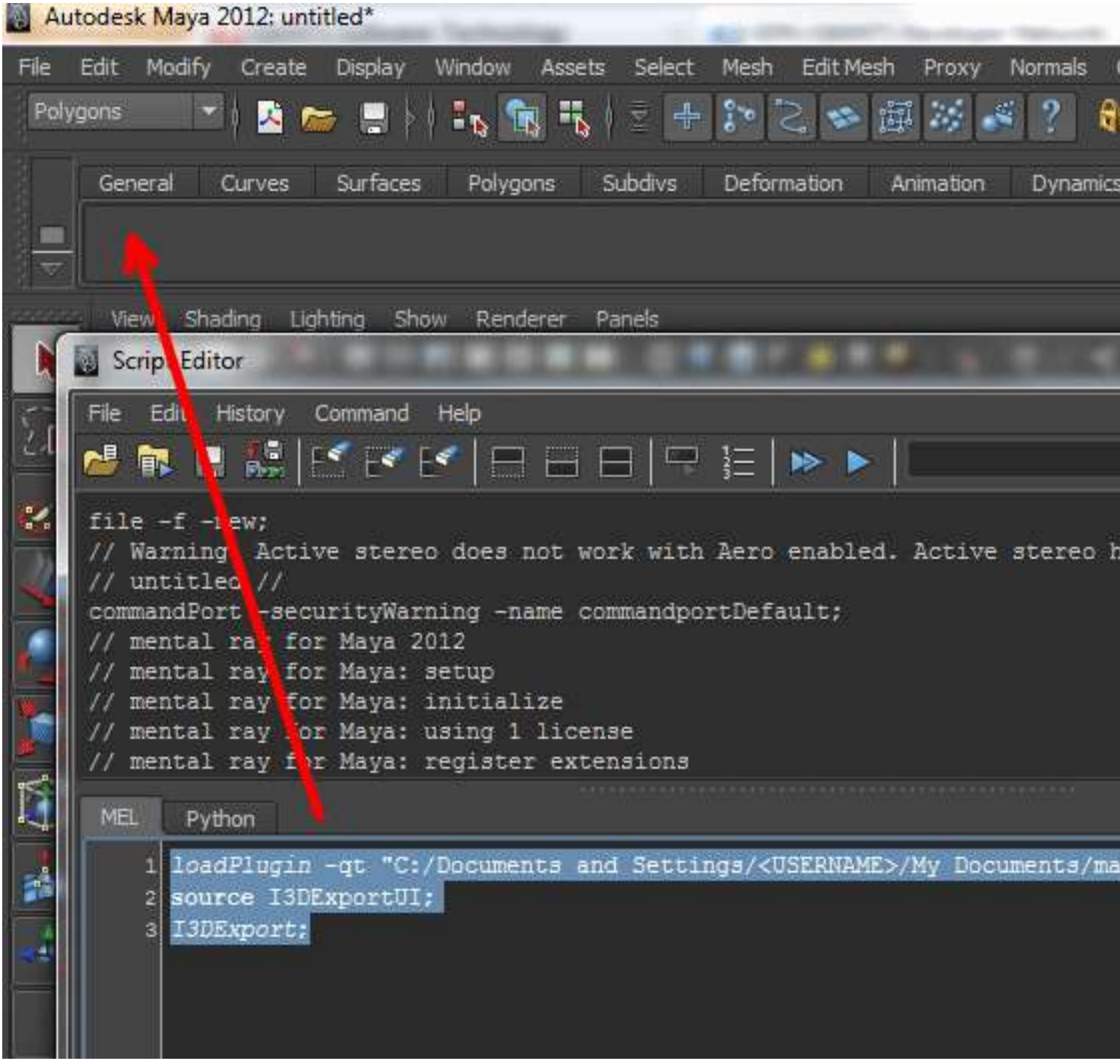
Install I3D Exporter Plugin Manually

Step 1

Place the files I3DExportUI.mel and I3DExporter2014-x64.mll into the folder "C:/Documents and Settings/<USERNAME>/My Documents/maya/scripts".

Step 2

Start Maya and open the Script Editor window.



Step 3

Select custom shelf tab

Step 4
Type the following commands in the lower portion of the Script Editor:
loadPlugin -qt "C:/Documents and Settings/<USERNAME>/My Documents/maya/scripts/I3DExporter2014-x64.mll";
source I3DExportUI;
I3DExport;
Highlight the commands, then use the middle mouse button, to drag the highlighted MEL commands to the shelf.

Note:

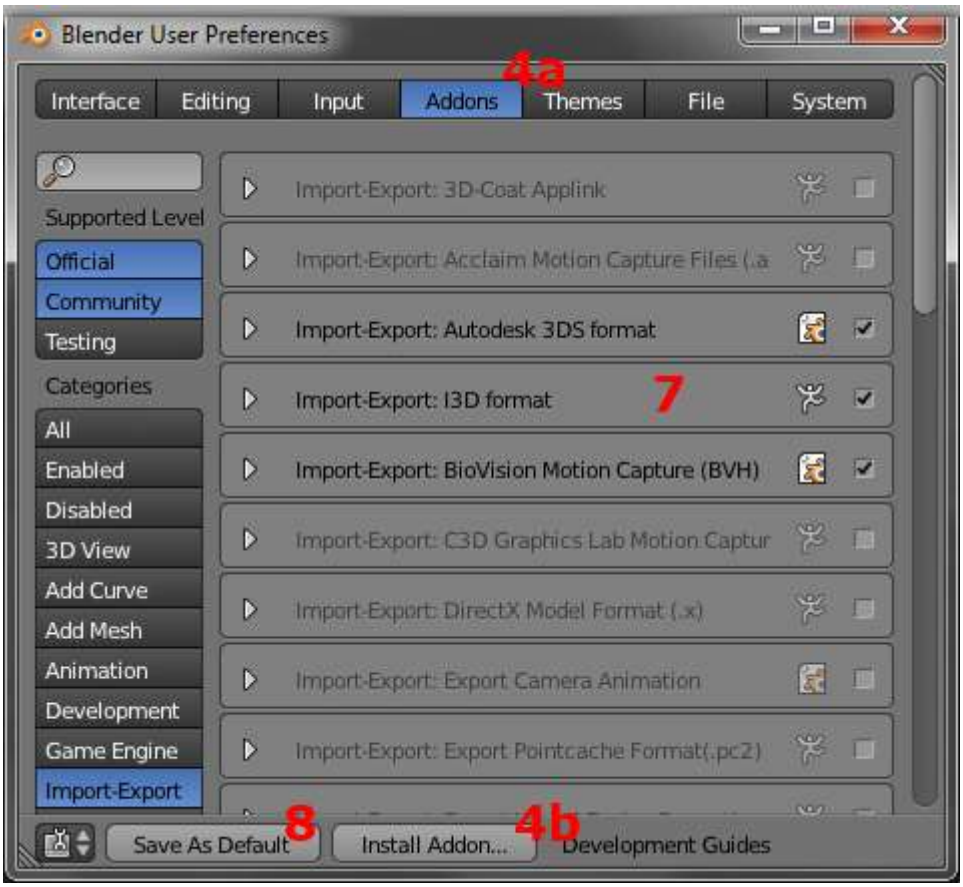
- Maya don't like backslashes so replace them with slashes.
- Replace <*USERNAME*> with your username.
- The file path can be different depending on the language of your operating system (eg. C:/Documents and Settings/<USERNAME>/My Documents/maya/scripts/I3DExporter2014-x64.mll)

Step 5
Copy the I3D_icon.bmp into the folder "C:/Documents and Settings/<USERNAME>/My Documents/maya/2014/prefs/icons". Edit shelf to replace the default icon with the one enclosed.

Blender I3D Exporter

Install I3D Exporter Plugin Manually (Windows)

1. Install Blender 2.6.2 32 or 64bit or higher here: http://www.blender.org/download/get-blender/
2. Extract the io_scene_i3d folder from the exporter zip archive and place it in your Blender addons folder. Example addons folder path: C:\Program Files\Blender Foundation\Blender\2.62\scripts\addons
3. Launch Blender and go to "File -> User Preferences...".
4. Click on "Install Addon..." in the "Addons" section.
5. Browse to the io_scene_i3d folder previously extracted. Example path: C:\Program Files\Blender Foundation\Blender\2.62\scripts\addons\io_scene_i3d
6. Select the file __init__.py and click "Install Addon...".
7. Select and enable the "Import-Export: I3D format" addon in the right list. Hint: Apply the "Community" and "Import-Export" filters on the left for less items to browse.
8. Click on the "Save as Default" button to automatically load the addon each time you launch Blender.
9. Now, you can export with "File -> Export" to "GIANTS (.i3d)".



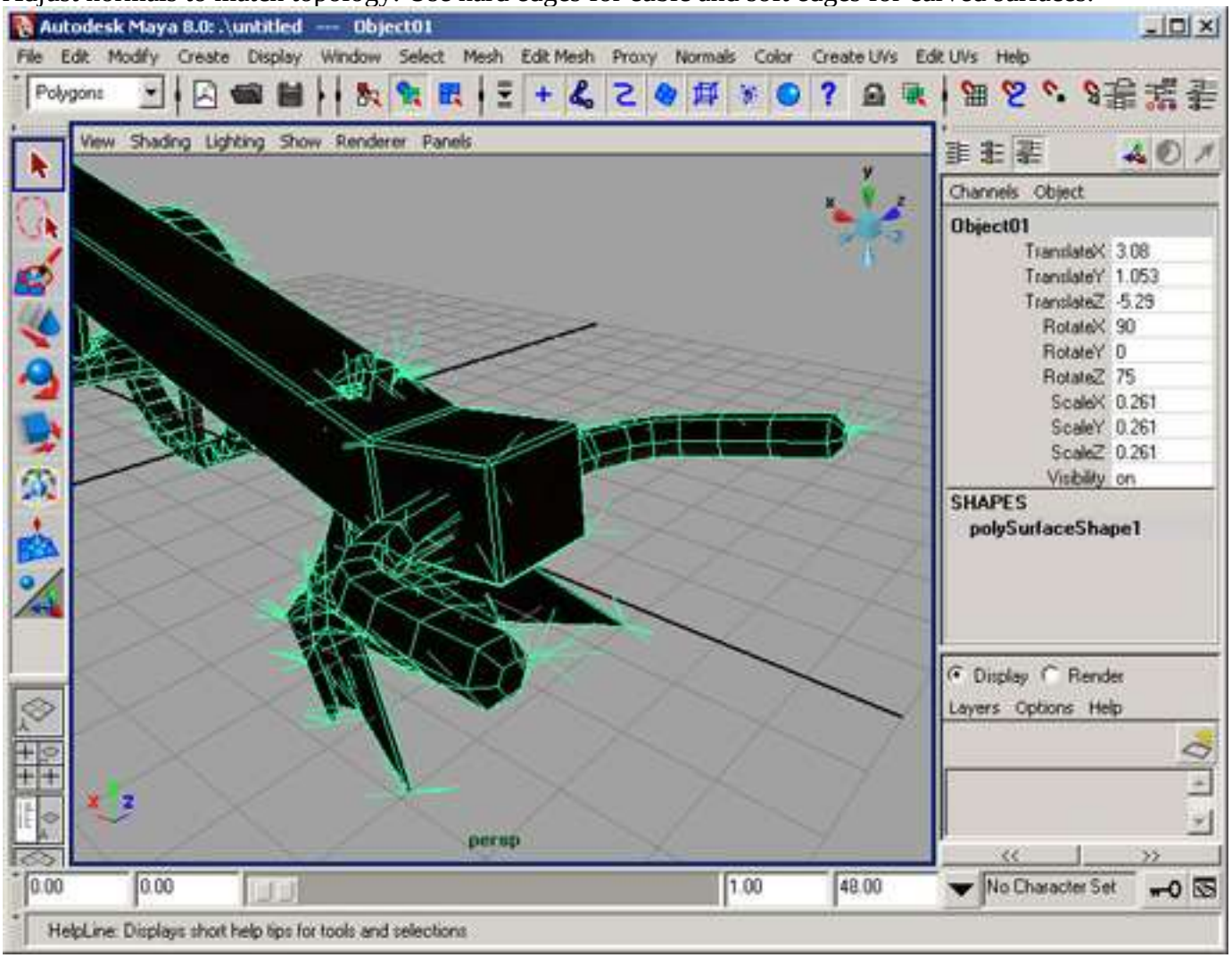
Autodesk 3ds MAX and Autodesk Maya are registered trademarks of the Autodesk Corp.

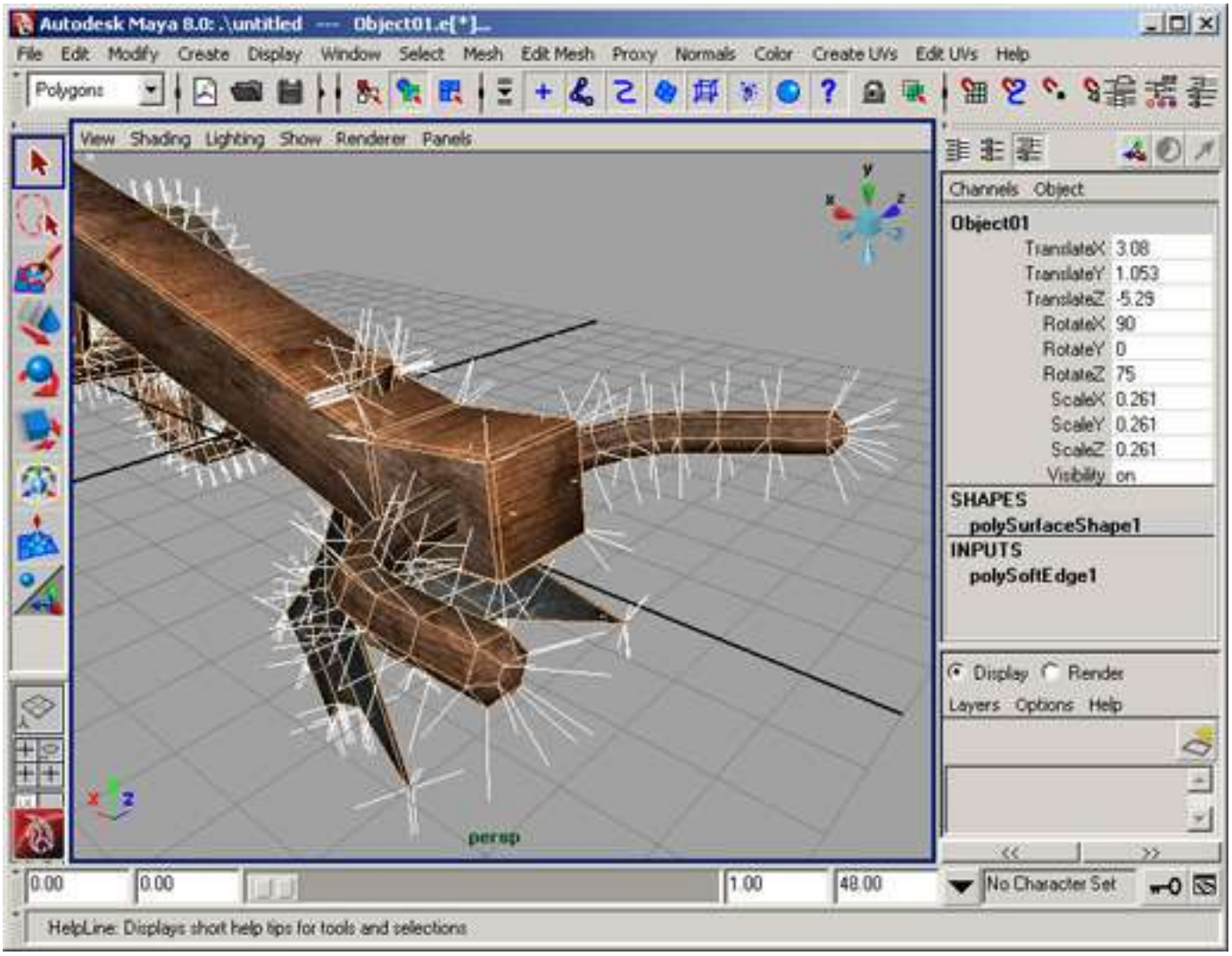
Content Creation - Artwork Guide

Autodesk Maya

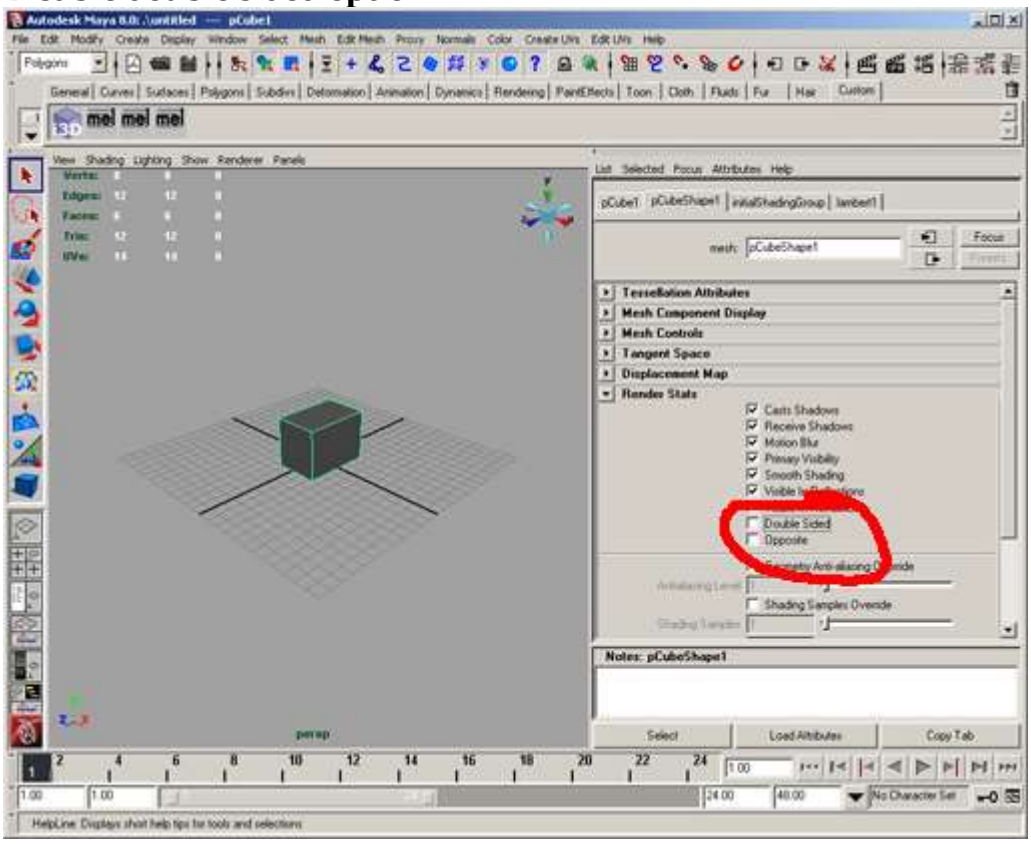
Set correct normals

Adjust normals to match topology. Use hard edges for cubic and soft edges for curved surfaces.

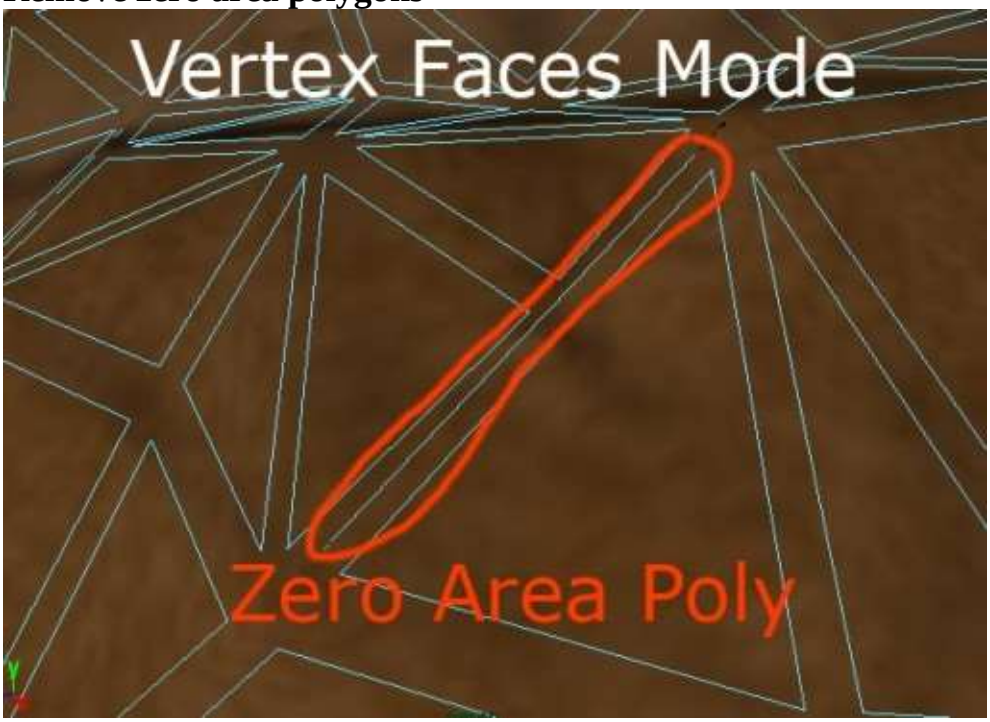




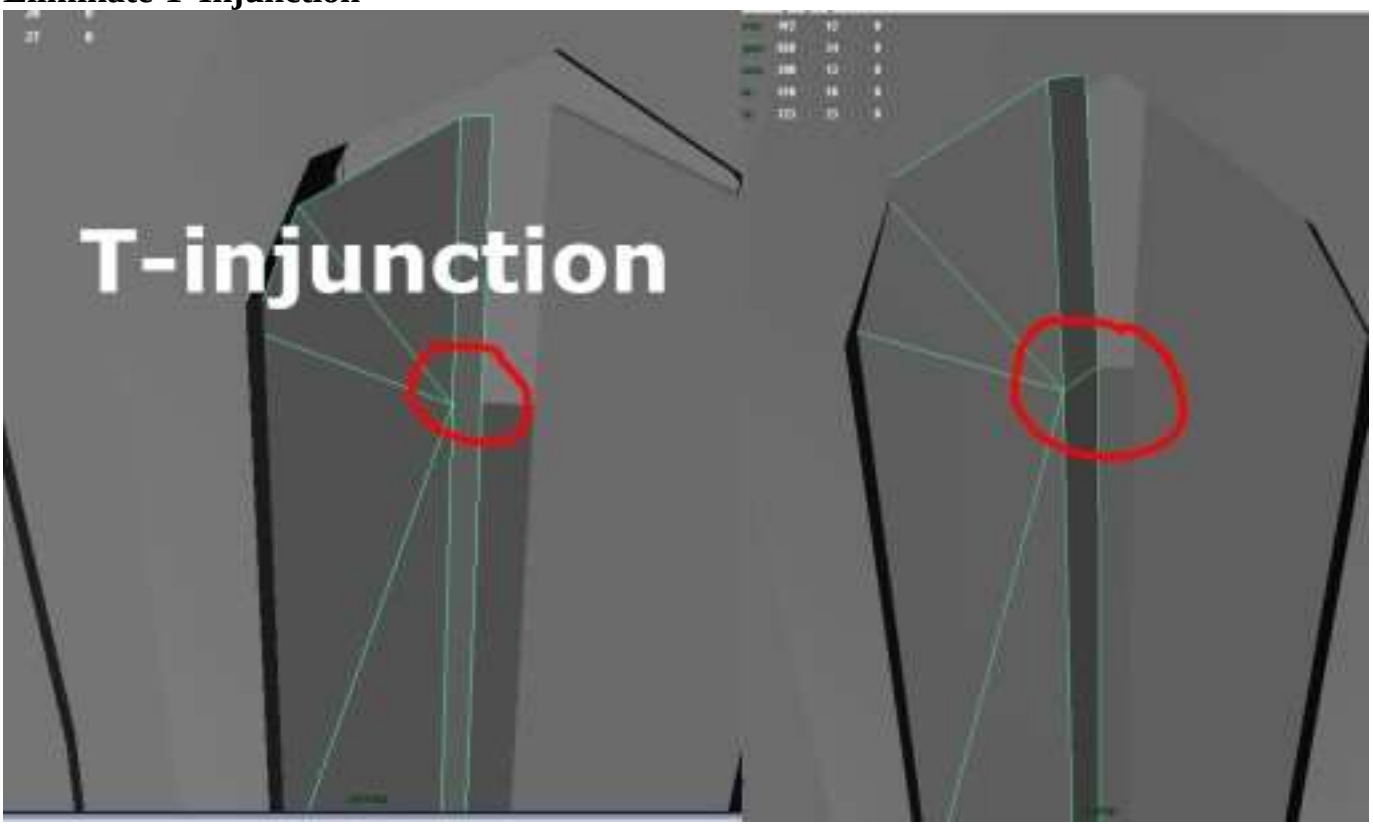
Disable double sided option



Remove zero area polygons



Eliminate T-Injunction



Texturing

JPEG compression artifacts

Never use the JPG format to store textures. Each time a jpg file is stored the quality of the image decreases. Even if the compression quality is set to 100%.

Saved once as png from orignal texture (left), saved multiple times as jpg (right)

Non-Power-of-Two textures

Use power-of-two textures but you can use non-squared textures. As a general rule you should make your textures as big as they are in pixels when beeing projected onto the screen in the game.
Eg. if you want to texture a fullscreen squad, you probably want to use a 1024x1024px to 2048x1024px sized texture.
Normalmap textures should be at least as big as the diffuse textures or even twice as big.

Texture size

Texture size must be 2048x2048 pixels or smaller. Bigger textures are not supported on older graphics cards (eg. GeforceFX, Radeon9xxx, Radeon8xx and Intel onboard graphic chips).

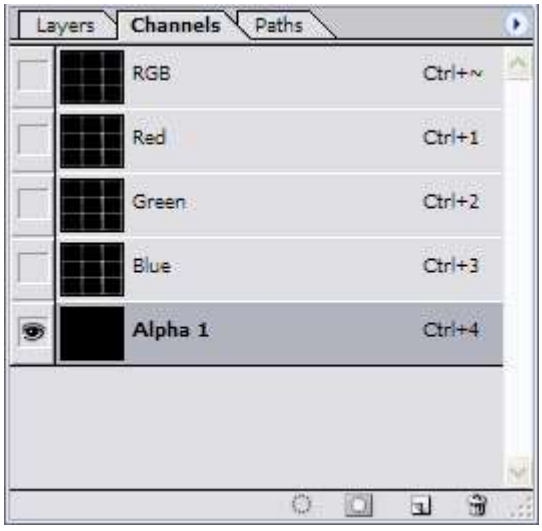
Generate maps for parallaxmapping

Requirements

The Nvidia Photoshop Normal Map Filter plug-in from Nvidia. This plug-in can be downloaded from developer.nvidia.com

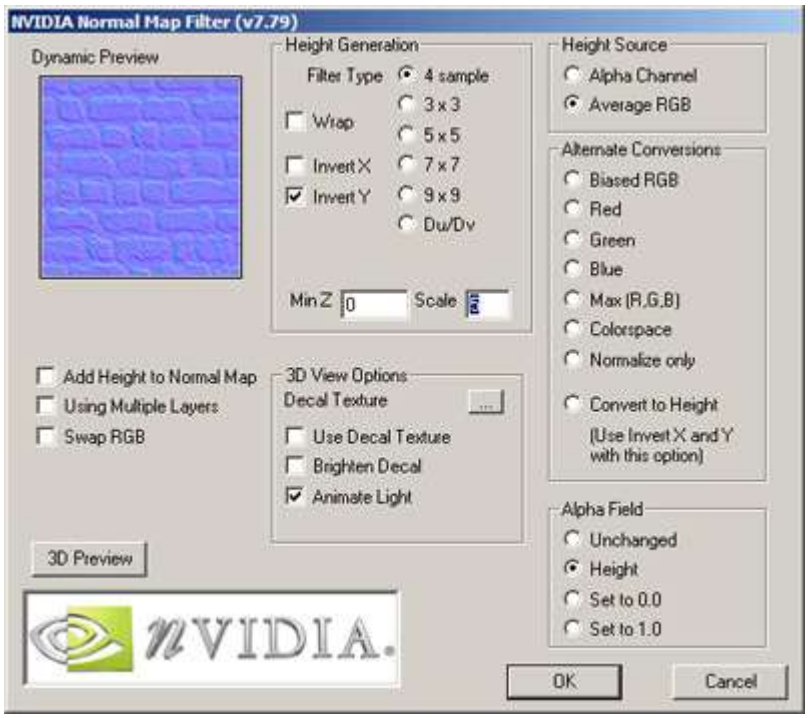
Description

- First, you need a greyscale image or a bump map as the base for the Parallax Map. Please note that most of the time, you will have to convert this greyscale image to a RGB image. You can do this with select Image > Mode > RGB Color (menu bar).
- Go to the "Window" tab and select "Show Channels". In the channel window, click the 'Create New Channel' icon on the bottom right, just left of the Trash Icon. A new channel "Alpha 1" will appear.



- Click one of the Color channels, press CTRL-A then CTRL-C to copy the channel informations to the clipboard for later use.

- In the channel window select the RGB channel with shift and the left mouse button. All channel tabs should now be marked or accordingly selected.
- Go now to the "Filter" menu and select "nvTools/NormalMapFilter...". In the Height Generation section, select InvertY. To create a Parallax Map you also have to select Height in the Alpha Field Section. With the "Scale" setting you can alter the intensity (depth) of the normal map. The "Filter Type" setting, defines how precise the normal map has to be. "4 sample" means precise, whereas 9x9 means rough. Click "OK" to proceed.



- You should now have a normal map, combined with a bump map in the Alpha Channel.
- Now you have to insert the previous saved channel information to the "Alpha 1" channel. Select the alpha channel with your left mouse button and then press CTRL-V to paste the Heightmap image into the channel.
- To increase the quality of your Parallax Map you can try also to apply a low gaussian filter to the alpha channel. Further you can reduce the contrast to get better results.
- Save the Image as png with transparency (RGBA 32 bit).

I3D Format

Introduction

I3D is an eXtensible Markup Language (XML) file format.

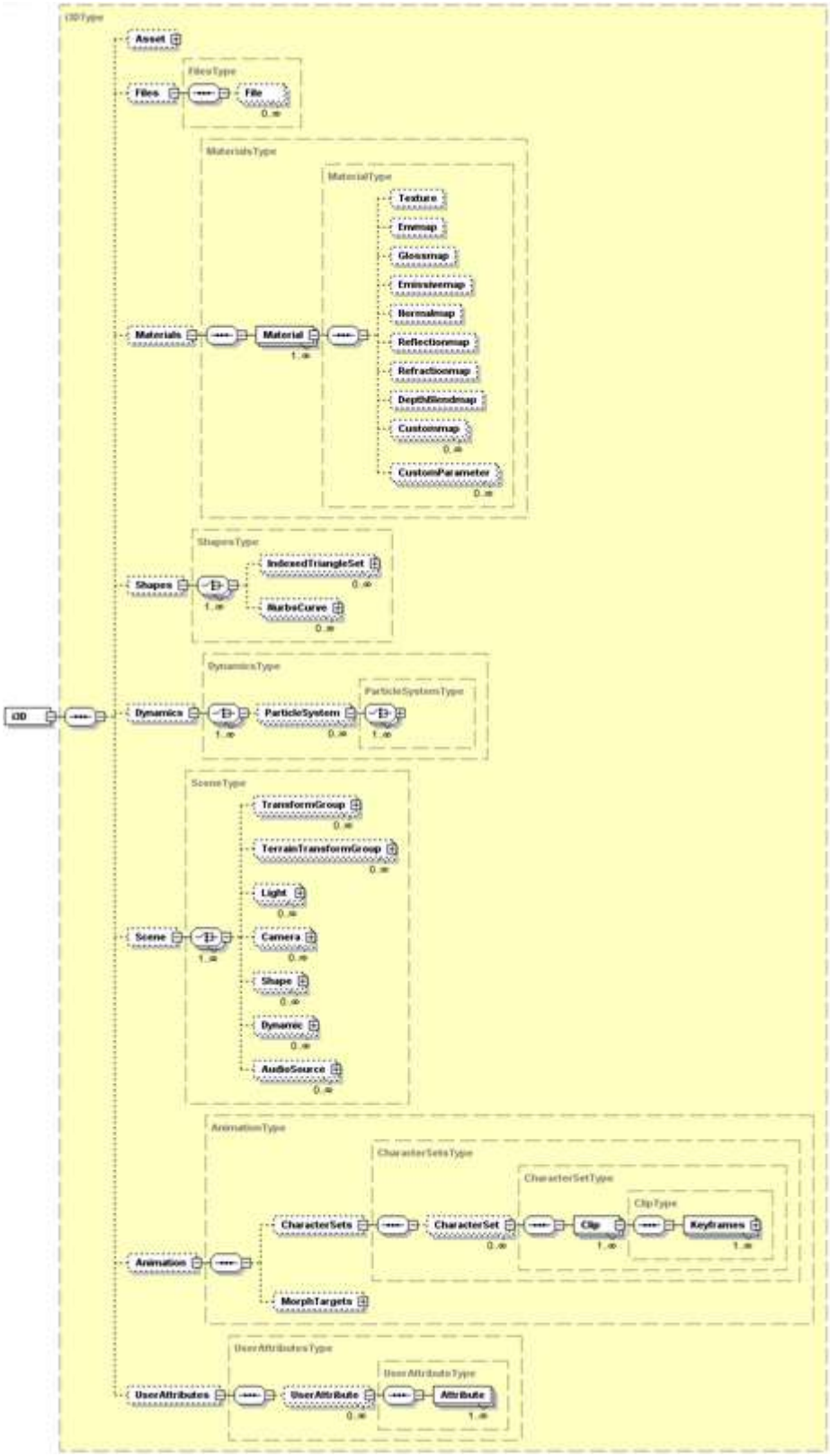
A vaild I3D file has up to seven parts: Textures, Materials, Shapes, Dynamics, Scene graph, Animation and Userdata. Dependent on application each part can be omitted.

The XML Schema language is used to describe the I3D feature set. Download: I3D 1.6 Schema

Features

- Scenegraph hierarchy with
  - TransformGroups/Bones: translation, rotation, scale and visibility
  - Lights: ambient, point, directional and spot lights
  - Cameras: field of view, near and far clipping plane
  - Shapes: meshes and nurbs curves (supports geometry instancing)
  - Particle Systems
- Animation
  - Animation sets
  - Clips
  - Keyframes: translation, rotation, scale and visibility
  - Morph Targets
- User Attributes
- Dynamics
  - Particle Systems (Sprite based)
    - Surface Emitter
    - Gravity Fields
    - Air Fields
- Meshes
  - Vertices
  - Normals
  - Vertex colors
  - Skin weights
  - Texture coordinates (including multiple uv sets)
  - Faces (vertex color, normals and texture coordinates per face)
- Curves
  - Nurbscurves
- Shader/Materials
  - Multitextures (arbitrary layered textures)
  - Lightmaps
  - Bumpmaps/Normalmaps
  - Environmentmaps
  - Specularmaps

Overview



Specification

General Layout

```
<?xml version="1.0" encoding="iso-8859-1"?>
<i3D name="MyScene" version="1.6" xsi:noNamespaceSchemaLocation="http://i3d.giants.ch/schema/i3d-1.6.xsd">
  <Files>..</Files>
  <Materials>..</Materials>
```

```
  <Shapes>..</Shapes>
  <Dynamics>..</Dynamics>
  <Scene>..</Scene>
  <Animation>..</Animation>
  <UserAttributes>..</UserAttributes>
</i3D>
```

Coordinate Systems

I3D uses a right-handed coordinate system.

Files

All used file references are defined here. File handles are mapped to the corresponding filenames.

```
<Files>
  <File fileId="1" filename="myTexture1.png" relativePath="true"/>
  ...
</Files>
```

Materials

Materials used in the materials attribute of Shape nodes within the scenegraph section are defined in this section.

```
<Materials>
  <Material name="myShader_001" materialId="12">
    <Texture fileId="1"/>
  </Material>
  ...
</Materials>
```

Shapes

Shapes are defined here and referenced from the scene graph section. This mechanism allows multiple instancing of shapes.

Dynamics

Dynamic objects are defined in this part and referenced from the Scene graph section.

```
<Dynamics>
  <ParticleSystem name="emitter1" dynamicId="1" type="sprite" rate="0.004" lifespanInfinite="false"
  speed="0.01" speedRandom="0" tangentSpeed="0" normalSpeed="1" lifespan="10000"
maxCount="1000"
  spriteScaleX="50" spriteScaleY="50" depthSort="false" emitterShape="pCubeShape1"
shader="lambert2">
    <Gravity force="0 -3 0"/>
    <Air velocity="2 0 0"/>
  </ParticleSystem>
</Dynamics>
```

Scenegraph

The Scene graph hierarchy (transformations, bones, joints, shapes, lights, cameras and particle systems) are stored in this section. Shapes, materials and particle systems are stored as references.

```
<Scene>
  <TransformGroup name="myGroup1" translation="1.25 0 -9" nodeId="47">
```

```
    <Shape name="mySphere" materialIds="1" shapeId="1" nodeId="48"/>
    <TransformGroup name="myGroup2" rotation="90 0 270" nodeId="49">
      <Shape name="mySphere2" materialIds="1" shapeId="2" nodeId="50"/>
    </TransformGroup>
    ...
  </TransformGroup>
  ...
</Scene>
```

Scene graph node types

TransformGroup

Attribute	Description	Type	Optional
name	Name of TransformGroup	string	no
translation	Translation	complex	yes
rotation	Rotation, euler order ZY'X"	complex	yes
scale	Scale	complex	yes
visibility	Visibility	boolean	yes
kinematic	Kinematic object	boolean	yes
dynamic	Dynamic object	boolean	yes
static	Static object	boolean	yes
compound	Compound parent	boolean	yes
compoundChild	Compound child	boolean	yes
collision	Enable collision	boolean	yes
ccd	Continuous collision detection	boolean	yes
trigger	Trigger	boolean	yes
cloth	Cloth	boolean	yes
restitution	Restitution	float	yes
staticFriction	Static friction	float	yes
dynamicFriction	Dynamic friction	float	yes
skinWidth	Skin width	float	yes
density	Density	float	yes
collisionMask	Collision mask	integer	yes
joint	Joint	boolean	yes
breakableJoint	Breakable joint	boolean	yes
jointBreakForce	Joint break force	double	yes
jointBreakTorque	Joint break torque	double	yes
projection	Enable joint projection	boolean	yes
xAxisDrive	Enable x axis drive	boolean	yes
yAxisDrive	Enable y axis drive	boolean	yes
zAxisDrive	Enable z axis drive	boolean	yes
drivePos	Enable drive position	boolean	yes
projDistance	Projection distance	double	yes
driveForceLimit	Drive force limit	double	yes
driveSpring	Drive spring	double	yes
driveSpring	Drive spring	double	yes

solverIterationCount	Solver iteration count	integer	yes
rotMinX	Rotation minimum x	double	yes
rotMinY	Rotation minimum y	double	yes
rotMinZ	Rotation minimum z	double	yes
rotMaxX	Rotation maximum x	double	yes
rotMaxY	Rotation maximum y	double	yes
rotMaxZ	Rotation maximum z	double	yes
transMinX	Translation minimum x	double	yes
transMinY	Translation minimum y	double	yes
transMinZ	Translation minimum z	double	yes
transMaxX	Translation maximum x	double	yes
transMaxY	Translation maximum y	double	yes
transMaxZ	Translation maximum z	double	yes
objectMask	Object mask	integer	yes
nodeId	Node reference id, used for Animation and UserAttributes section	integer	no

Element	Description	Type	Cardinality
TransformGroup	Transform group node	complex	0..N
Shape	Shape node	complex	0..N
Camera	Camera node	complex	0..N
Light	Light node	complex	0..N
Dynamic	Dynamic node (eg. particle system)	complex	0..N

Shape

Attribute	Description	Type	Optional
shapeId	Shape reference id (reference to shapes section)	string	no
materialIds	List of used material ids (reference to materials section)	string	no
skinBindNodeIds	Skin bind node ids	string	yes
castsShadows	Casts shadows	boolean	yes
receiveShadows	Receive shadows	boolean	yes
clipDistance	Clip distance	double	yes
nonRenderable	Non renderable	boolean	yes

Camera

Attribute	Description	Type	Optional
fov	Field of view	double	no
nearClip	Near clipping plane	double	no
farClip	Far clipping plane	double	no

Light

Attribute	Description	Type	Optional
type	Light type	string	no

diffuseColor	Diffuse color	complex	yes
emitDiffuse	Emit diffuse	string	yes
specularColor	Specular color	complex	yes
emitSpecular	Emit specular	string	yes
castShadowMap	Cast shadow Map	boolean	yes
depthMapBias	Depth map bias	double	yes
depthMapResolution	Depth map resolution	integer	yes
shadowFarDistance	Shadow far fistance	double	yes
shadowTextureOffset	Shadow texture offset	double	yes
shadowExtrusionDistance	Shadow extrusion distance	double	yes
decayRate	Decay rate	integer	yes
coneAngle	coneAngle	double	yes
dropOff	dropOff	integer	yes
projTexture	Projective texture name	string	yes
range	Range	double	no

5 Dynamic

Attribute	Description	Type	Optional
dynamicId	Dynamic id (reference to dynamics section)	string	no

Example

```
<Scene>
  <Camera name="camera1" translation="0 2.00 6.47" rotation="-13.07 0 0" fov="54.43" nearClip="0.01"
farClip="1000" nodeId="34"/>
  <Light name="pointLight1" translation="6.56 5.76 4.04" type="point" diffuseColor="1 1 1" range="10"
nodeId="35"/>
  <TransformGroup name="group1" translation="1.35 0.96 0.81" nodeId="36">
     <Shape name="pCubeShape1" rotation="-23.94 6.49 14.29" materialIds="1 2" shapeId="1"
nodeId="37"/>
  </TransformGroup>
</Scene>
```

Animation

Motions are defined in this part. Clips are the basic building block and allow Non-Linear Animation by composing multiple clips.

```
  <Animation>
    <AnimationSets>
      <AnimationSet name="walk_crouched">
        <Clip name="walk_crouched1Source" duration="1000">
          <Keyframes nodeId="Hips">
            <Keyframe time="0" translation="-0.467 13.504 39.842"/>
            <Keyframe time="333" translation="-0.559 12.915 39.370"/>
            <Keyframe time="1000" translation="-2.610 11.917 35.462"/>
          </Keyframes>
          <Keyframes node="Chest">
            <Keyframe time="0" rotation="-1.013 -4.465 0.890"/>
            <Keyframe time="666" rotation="10.009 -6.667 4.381"/>
```

```
        <Keyframe time="1000" rotation="11.638 -3.906 4.115"/>
       </Keyframes>
      </Clip>
    </AnimationSet>
  </AnimationSets>
</Animation>
```