User Modules

Overview

What follows is a list of all the premade user modules that come with the game. All of these have helper functions that can be accessed by looking for functions with the prefix displayed for each module.

Interactive camera animation

Helper function prefix: CameraAnimation

General

The camera animation system is used to move the player camera along a specific path while still allowing some degree of interaction.

Instructions

To use the system, place CameraAnimation nodes (Area → CameraAnimation) in the scene and give each of them the name of the animation followed by an underscore and their placement within the animation (such as WakeUpAnimation_1, WakeUpAnimation_2 etc). Note that if you duplicate (Ctrl-D) a node, the duplicate will automatically get the next available number as its suffix.
Rotate the nodes so that their Z-axes point in the direction you want the camera to look at that part of the animation.
In script, call the helper function CameraAnimation_Begin() to start the animation.

Node properties

MaxYaw/MaxPitch
Dictates how far the player is allowed to rotate the camera away from the target forward rotation. This value is interpolated between nodes.

LookSpeedMultiplier
How fast the player can rotate the camera.

ForwardTime
The time it takes to move from this node to the next at full speed forward.

BackwardTime
The time it takes to move from the next node to this one at full reverse speed.

AutoMovement

Dictates how the animation behaves when no movement input is given, or when movement input is disabled. 0 stops movement, 1 is as if given full forward speed (animation will finish after [ForwardTime] seconds), -1 is as if given full reverse speed.

LookMoveDistance
The length between this node and the next the player can move just by looking toward the next node, given as a value between 0 and 1.

LookMoveMaxAngle
The maximum angle away from the next node the player can look and still move forward. Only valid if LookMoveDistance > 0.

InteractiveMovement
If checked, the player can control the movement between this node and the next. If unchecked, the AutoMovement property has complete control over the movement and should not be set to 0.

CrouchOnExit
If checked, the player will automatically crouch if the animation is ended at this node.

InitToCamera
If checked, the node is given the orientation of the camera when the animation starts. This is very useful when you want the animation to have a smooth beginning, or you want to make sure the player is returned to where they were when the end node is reached.

CallbackFunc
Function in the level script that should be called when reaching this node. The syntax is bool FuncName(const tString &in asEntity, int alDirection), where asEntity is the name of the node and alDirection is 1 if moving forward and -1 if moving backward.

Attack Meter

Helper function prefix: AttackMeter_

The attack meter module is used to tell the player that they are in danger and, if they don't escape the danger in time, applies damage to them, knocks them down and kills them. It's used by AI agents, and also for when (for example) the player jumps off a cliff.

Credits

Helper function prefix: Credits_

The credits handler simply displays a scrolling list of credits - see `config/Credits.cfg` for the format used.

Datamine

Helper function prefix: Datamine_

Datamining is what happens when the player in SOMA touches a dead person, intercom or other piece of technology and hears the last few moments of data buffer as an audio clip. Most of this is handled automatically by `Prop_Datamine`, but there are a few helper functions available.

Description

Helper function prefix: Description_

The description handler is an old module pre-SOMA; it's for adding descriptive text on-screen. SOMA uses Readables and ZoomAreas instead. Description may not be full up-to-date with SOMA's code.

Distortion Effects

Helper function prefix: DistortionEffect_

This module implements the distortion effects that happen to the player's vision as they get close to enemy creatures such as the Flesher. It's called automatically from some AI, but it's also possible to explicitly add additional distortion effects e.g. to trigger effects in a particular area.

Emotion

Helper function prefix: Emotion_

This module implements heavy breathing and heartbeats effects, allowing you to programmatically change volume and rate of both.

Game Over

Helper function prefix: GameOver_

This module implements the death screen, and the weird sequence of images you get just before death. It's automatically called by the `AttackHandler`.

Highlight Effect

This module manages the highlighting of particular interactive objects in an area near to the player.

Hint

Helper function prefix: Hint_

This module handles the textual hints (with embedded button symbols) that pop up at the top of the screen.

Inventory

Helper function prefix: Inventory_

The inventory module handles the display of the player's inventory (the management of it is done in `PlayerToolHandler`).

Light Flash

Helper function prefix: LightFlash_

The light flash module handles the display of a flash effect of bright light at the particular positions in the 3D world.

Map Effects

Helper function prefix: Map_

The Map Effects module handles a variety of different helpers for the map's effects, including fog and environment particles. It also manages collections of environmental presets to fade between, so you can transition a map's appearance from one state to another.

Menu

Helper function prefix: MainMenu_, PauseMenu_

The MenuHandler module implements all of SOMA's front-end menus.

Player Energy

Helper function prefix: PlayerEnergy_

The Player Energy module implements the recharging of the player's health (through use of 'Wau Flowers') in SOMA. 'Wau Flowers' themselves are of type `Prop_EnergySource`.

Player Hands

Helper function prefix: PlayerHands_

The Player Hands module implements all the animations that the first-person player hands take in game. It also allows binding the camera to a socket on the player hands animation for custom 'cut-scene'-like moments, such as climbing out of the dome in Upsilon.

Player Pickup

This is an old module no longer used in SOMA.

Player Tool

Helper function prefix: Player_

This module handles the player's use of `Prop_Tool` objects that they pick up, including maintaining an inventory list, deciding if a tool is equipped (shown in the player's hands) and whether tools can be used on particular entities or areas. In SOMA, this covers things like the Omnitool or scan chips.

Terrain Particles

This is an old module no longer used in SOMA.

Wake Handler

Helper function prefix: Wake_

This implements a very simple wakeup effect, as if blinking eyelids (see the start of the Theta labyrinth after Akers has embedded you in coral).