Sage Code Reference

1.6.0

Generated by Doxygen 1.7.6.1

Thu Sep 13 2012 21:41:01

Contents

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

Class Documentation

2.1 SageController Class Reference

This is the main class that controls a Sage Library. This is attached to any animated GameObject, and using the Sage Library, it will control all of the animations on the GameObject.

Public Member Functions

- **SageController** (ISageLibrary library)
- virtual void Reset ()

  This function is called when this object is reset in the editor.
- void Awake ()

  This function is called when this script is awoken.
- void OnDestroy ()

  This function is called when this script is destroyed.
- void RecreateLibrary ()

  This will stop all current state machines, remove the current library, and then recreate everything.
- void RecacheAnimations ()

  This will recache all animations that have been cached. This needs to be called if and when the Animation object has it's animation clips changed dynamically.
- virtual void Update ()

  This function is called when this script is updated.
- bool SetGlobalSpeed (float speed)

  This function sets the global speed of this SageController.
- float GetGlobalSpeed ()

  This function gets the global speed of this SageController.
- bool SetFloat (string variableName, float value)

  This function sets a float variable on this SageController.

- bool SetFloat (string variableName, float value, bool instant)

  This function sets a float variable on this SageController.
- float GetFloat (string variableName)

  This function gets a float variable on this SageController.
- bool StartStateMachine (string stateMachine)

  This function starts a state machine on this SageController.
- bool StartStateMachine (string stateMachine, float transitionTime)

  This function starts a state machine on this SageController.
- bool StartStateMachine (string stateMachine, float transitionTime, string initial-State)

  This function starts a state machine on this SageController.
- bool StopStateMachine (string stateMachine)

  This function stops a state machine on this SageController.
- bool StopStateMachine (string stateMachine, float transitionTime)

  This function stops a state machine on this SageController.
- bool StopAllStateMachines ()

  This function stops all state machines on this SageController.
- bool StopAllStateMachines (float transitionTime)

  This function stops all state machines on this SageController.
- bool IsSuspended ()

  This function checks if this SageController is currently suspended.
- bool Suspend (float outTime)

  This function suspends this SageController, which will pause the SageController and preserve all current states and active state machines.
- bool SuspendAndPlayAnimation (string animation, float speed, float outTime, float inTime)

  This function suspends this SageController, plays an animation, and then will automatically resume the SageController.
- bool SuspendAndPlayAnimations (List$<$ string $>$ animations, float speed, float outTime, float crossBlendTime, float inTime)

  This function suspends this SageController, plays a list of animations, and then will automatically resume the SageController.
- bool Resume (float inTime)

  This function resumes this SageController, if suspended, reactivating all previous activate states and state machines.
- bool IsStateMachineActive (string stateMachine)

  This function checks if a state machine is currently activate on this SageController.
- string GetStateMachineCurrentState (string stateMachine)

  This function gets the current state of an activate state machine.
- bool ForceState (string stateMachine, string newState)

  This functions forces a state on a state machine, ignoring normal transistion paths.
- bool ForceState (string stateMachine, string newState, float transitionTime)

  This functions forces a state on a state machine, ignoring normal transistion paths.
- bool ForceState (string stateMachine, string newState, float transitionTime, bool allowStateRestart)

This functions forces a state on a state machine, ignoring normal transistion paths.

- bool TransistionToState (string stateMachine, string newState)

  This function causes a state machine to transition from it's current state to a new state, using only normal transition paths.
- bool TransistionToState (string stateMachine, string newState, bool allowStateRestart)

  This function causes a state machine to transition from it's current state to a new state, using only normal transition paths.
- bool SetStateEnteredDelegate (string stateMachine, string state, SageStateEnteredDelegate enteredDelegate)

  This function sets the callback for entering a state in a specific state machine.
- bool SetStateUpdatedDelegate (string stateMachine, string state, SageStateUpdatedDelegate updatedDelegate)

  This function sets the callback for updating a state in a specific state machine.
- bool SetStateExitedDelegate (string stateMachine, string state, SageStateExitedDelegate exitedDelegate)

  This function sets the callback for exiting a state in a specific state machine.
- bool SetApplyMovementDelegate (SageApplyMovementDelegate applyMovementDelegate)

  This sets a delegate function that should be called when movement should be applied.
- void **OnGUI** ()

Public Attributes

- SageLibraryAsset libraryAsset = null

  This is the Sage Library Asset that this Sage Controller is using.
- Animation animationTarget = null

  This is the Animation compontent that is the target for animation for this Sage - Controller. If this is null, it will default to looking for an Animation component on the same game object as the SageController component.
- GameObject movementTarget = null

  This is the GameObject that is the target for world movement for this Sage Controller. If this is null, it will default to use the GameObject that this SageController component is attached to.
- bool displayRuntimeInfo = false

  This will display the runtime info of this Sage Controller. Such as what graphs are active and animations are playing etc.

Properties

- ISageLibrary LibraryRuntime `[get]`

  This gets the library runtime that is assigned to this Sage Controller. NOTE: USING THIS DIRECTLY IS AN ADVANCED OPTION, AND NOT FULLY SUPPORTED!

2.1.1 Detailed Description

This is the main class that controls a Sage Library. This is attached to any animated GameObject, and using the Sage Library, it will control all of the animations on the GameObject.

2.1.2 Member Function Documentation

2.1.2.1 void **SageController.Awake** ( )

This function is called when this script is awoken.

2.1.2.2 bool **SageController.ForceState** (string *stateMachine,* string *newState*)

This functions forces a state on a state machine, ignoring normal transistion paths.

Parameters

state-Machine	The state machine to access.
newState	The new state to force on the state machine.

Returns

Returns whether or not this function was executed successfully.

2.1.2.3 bool **SageController.ForceState** (string *stateMachine,* string *newState,* float *transitionTime*)

This functions forces a state on a state machine, ignoring normal transistion paths.

Parameters

state-Machine	The state machine to access.
newState	The new state to force on the state machine.
transition-Time	This is the amount of time (in seconds) it should take to transistion into this state.

Returns

Returns whether or not this function was executed successfully.

2.1.2.4 bool SageController.ForceState (string *stateMachine*, string *newState*, float *transitionTime*, bool *allowStateRestart*)

This functions forces a state on a state machine, ignoring normal transistion paths.

Parameters

stateMachine	The state machine to access.
newState	The new state to force on the state machine.
transitionTime	This is the amount of time (in seconds) it should take to transistion into this state.
allowStateRestart	This indicates if when trying to transition into the same state, that the state should be allowed to start over from the beginning.

Returns

Returns whether or not this function was executed successfully.

2.1.2.5 float SageController.GetFloat (string *variableName*)

This function gets a float variable on this SageController.

Parameters

variableName	The name of the variable to get.

Returns

Returns the float value of the variable.

2.1.2.6 float SageController.GetGlobalSpeed ( )

This function gets the global speed of this SageController.

Returns

Returns the global speed.

2.1.2.7 string SageController.GetStateMachineCurrentState (string *stateMachine*)

This function gets the current state of an activate state machine.

Parameters

state-Machine	The state machine to get the state of.

Returns

Returns the current state of the state machine.

2.1.2.8 bool SageController.IsStateMachineActive (string *stateMachine*)

This function checks if a state machine is currently activate on this SageController.

Parameters

state-Machine	The state machine to check.

Returns

Returns if the requested state machine is currently active.

2.1.2.9 bool SageController.IsSuspended ( )

This function checks if this SageController is currently suspended.

Returns

Returns if this SageController is suspended.

2.1.2.10 void SageController.OnDestroy ( )

This function is called when this script is destroyed.

2.1.2.11 void SageController.RecacheAnimations ( )

This will recache all animations that have been cached. This needs to be called if and when the Animation object has it's animation clips changed dynamically.

2.1.2.12 void SageController.RecreateLibrary ( )

This will stop all current state machines, remove the current library, and then recreate everything.

2.1.2.13 virtual void SageController.Reset ( ) `[virtual]`

This function is called when this object is reset in the editor.

2.1.2.14 bool SageController.Resume (float *inTime*)

This function resumes this SageController, if suspended, reactivating all previous activate states and state machines.

Parameters

inTime	This is how long (in seconds) it should take to resume this SageController.

Returns

Returns whether or not this function was executed successfully.

2.1.2.15 bool SageController.SetApplyMovementDelegate (SageApplyMovementDelegate *applyMovementDelegate*)

This sets a delegate function that should be called when movement should be applied.

Parameters

applyMovementDelegate	The function to call.

Returns

Whether or not the delegate function was set properly.

2.1.2.16 bool SageController.SetFloat (string *variableName,* float *value*)

This function sets a float variable on this SageController.

Parameters

variableName	The name of the variable to be set.
value	The value to set the variable to.

Returns

Returns whether or not this function was executed successfully.

2.1.2.17 bool SageController.SetFloat (string *variableName,* float *value,* bool *instant*)

This function sets a float variable on this SageController.

Parameters

variable-Name	The name of the variable to be set.
value	The value to set the variable to.
instant	This indicates if this value should be set instantly, or if it should enforce any change speed restrictions present on the variable.

Returns

Returns whether or not this function was executed successfully.

2.1.2.18 bool SageController.SetGlobalSpeed (float *speed*)

This function sets the global speed of this SageController.

Parameters

speed	The speed to set it to.

Returns

Returns whether or not this function was executed successfully.

2.1.2.19 bool SageController.SetStateEnteredDelegate (string *stateMachine,* string *state,* SageStateEnteredDelegate *enteredDelegate*)

This function sets the callback for entering a state in a specific state machine.

Parameters

state-Machine	The state machine to find the state in.
state	The state to set the entered delegate on.
entered-Delegate	The entered delegate to set. Passing in null clears any existing delegate.

Returns

Whether or not the entered delegate was set successfully.

2.1.2.20 bool SageController.SetStateExitedDelegate (string *stateMachine,* string *state,* SageStateExitedDelegate *exitedDelegate*)

This function sets the callback for exiting a state in a specific state machine.

Parameters

state-Machine	The state machine to find the state in.
state	The state to set the exited delegate on.
exited-Delegate	The exited delegate to set. Passing in null clears any existing delegate.

Returns

Whether or not the exited delegate was set successfully.

2.1.2.21 bool SageController.SetStateUpdatedDelegate (string *stateMachine,* string *state,* SageStateUpdatedDelegate *updatedDelegate*)

This function sets the callback for updating a state in a specific state machine.

Parameters

state-Machine	The state machine to find the state in.
state	The state to set the updated delegate on.
updated-Delegate	The updated delegate to set. Passing in null clears any existing delegate.

Returns

Whether or not the updated delegate was set successfully.

2.1.2.22 bool SageController.StartStateMachine (string *stateMachine*)

This function starts a state machine on this SageController.

Parameters

state-Machine	The name of the state machine to start.

Returns

Returns whether or not this function was executed successfully.

2.1.2.23 bool SageController.StartStateMachine (string *stateMachine*, float *transitionTime*)

This function starts a state machine on this SageController.

Parameters

stateMachine	The name of the state machine to start.
transitionTime	This is how long it should take to fully start the state machine.

Returns

Returns whether or not this function was executed successfully.

2.1.2.24 bool SageController.StartStateMachine (string *stateMachine*, float *transitionTime*, string *initialState*)

This function starts a state machine on this SageController.

Parameters

stateMachine	The name of the state machine to start.
transitionTime	This is how long (in seconds) it should take to fully start the state machine.
initialState	The name of the initial state to start this state machine in.

Returns

Returns whether or not this function was executed successfully.

2.1.2.25 bool SageController.StopAllStateMachines ( )

This function stops all state machines on this SageController.

Returns

Returns whether or not this function was executed successfully.

2.1.2.26 bool SageController.StopAllStateMachines (float *transitionTime*)

This function stops all state machines on this SageController.

Parameters

transition-Time	This is how long (in seconds) it should take to fully stop all of the state machine.

Returns

Returns whether or not this function was executed successfully.

2.1.2.27 bool SageController.StopStateMachine (string *stateMachine*)

This function stops a state machine on this SageController.

Parameters

state-Machine	The name of the state machine to stop.

Returns

Returns whether or not this function was executed successfully.

2.1.2.28 bool SageController.StopStateMachine (string *stateMachine,* float *transitionTime*)

This function stops a state machine on this SageController.

Parameters

state-Machine	The name of the state machine to stop.
transition-Time	This is how long (in seconds) it should take to fully stop the state machine.

Returns

Returns whether or not this function was executed successfully.

2.1.2.29 bool SageController.Suspend (float *outTime*)

This function suspends this SageController, which will pause the SageController and preserve all current states and active state machines.

Parameters

outTime	This is how long (in seconds) it should take to suspend this SageController.

Returns

Returns whether or not this function was executed successfully.

2.1.2.30 bool SageController.SuspendAndPlayAnimation (string *animation,* float *speed,* float *outTime,* float *inTime*)

This function suspends this SageController, plays an animation, and then will automatically resume the SageController.

Parameters

animation	The animation to play.
speed	The speed to playback the animation at.
outTime	This is how long (in seconds) it should take to suspend and blend into this animation.
inTime	This is how long (in seconds) it should take to resume and blend out of this animation.

Returns

Returns whether or not this function was executed successfully.

2.1.2.31 bool SageController.SuspendAndPlayAnimations (List< string > *animations,* float *speed,* float *outTime,* float *crossBlendTime,* float *inTime*)

This function suspends this SageController, plays a list of animations, and then will automatically resume the SageController.

Parameters

animations	The list of animations to play, in order.
speed	The speed to playback the animations at.
outTime	This is how long (in seconds) it should take to suspend and blend into these animations.
crossBlend-Time	This is how long (in seconds) it should take to blend between animations in the list.
inTime	This is how long (in seconds) it should take to resume and blend out of these animations.

Returns

Returns whether or not this function was executed successfully.

2.1.2.32 bool SageController.TransistionToState (string *stateMachine,* string *newState*)

This function causes a state machine to transition from it's current state to a new state, using only normal transition paths.

Parameters

state-Machine	The state machine to access.
newState	The new state to transistion to on the state machine.

Returns

Returns whether or not this function was executed successfully.

2.1.2.33 bool SageController.TransistionToState (string *stateMachine,* string *newState,* bool *allowStateRestart*)

This function causes a state machine to transition from it's current state to a new state, using only normal transition paths.

Parameters

state-Machine	The state machine to access.
newState	The new state to transistion to on the state machine.
allowState-Restart	This indicates if when trying to transition into the same state, that the state should be allowed to start over from the beginning.

Returns

Returns whether or not this function was executed successfully.

2.1.2.34 virtual void SageController.Update ( ) `[virtual]`

This function is called when this script is updated.

2.1.3 Member Data Documentation

2.1.3.1 Animation **SageController.animationTarget** = null

This is the Animation compontent that is the target for animation for this Sage Controller. If this is null, it will default to looking for an Animation component on the same game object as the SageController component.

2.1.3.2 bool **SageController.displayRuntimeInfo** = false

This will display the runtime info of this Sage Controller. Such as what graphs are active and animations are playing etc.

2.1.3.3 SageLibraryAsset **SageController.libraryAsset** = null

This is the Sage Library Asset that this Sage Controller is using.

2.1.3.4 GameObject **SageController.movementTarget** = null

This is the GameObject that is the target for world movement for this Sage Controller. If this is null, it will default to use the GameObject that this SageController component is attached to.

2.1.4 Property Documentation

2.1.4.1 ISageLibrary **SageController.LibraryRuntime** `[get]`

This gets the library runtime that is assigned to this Sage Controller. NOTE: USING THIS DIRECTLY IS AN ADVANCED OPTION, AND NOT FULLY SUPPORTED!

2.2 SageTestingGUI Class Reference

This class is used to test a Sage Library in game. Only one instance of this class can be active at a time.

Public Attributes

- SageController targetController = null

  This is the target controller that this Sage Testing GUI should be testing.
- float depthOffset = 2.0f

  This is how much to offset the target controller away from the main camera.
- float horizontalOffset = 0.0f

  This is how much to offset the target controller to the right of the main camera.
- float verticalOffset = 0.75f

This is how much to offset the target controller vertically from the main camera.

- float rotation = 270.0f

  This is the number of degrees to rotate the main camera around the target controller.

Properties

- static SageTestingGUI Instance `[get]`

  This returns the current instance of the Sage Testing GUI.

2.2.1 Detailed Description

This class is used to test a Sage Library in game. Only one instance of this class can be active at a time.

2.2.2 Member Data Documentation

2.2.2.1 float **SageTestingGUI.depthOffset** = 2.0f

This is how much to offset the target controller away from the main camera.

2.2.2.2 float **SageTestingGUI.horizontalOffset** = 0.0f

This is how much to offset the target controller to the right of the main camera.

2.2.2.3 float **SageTestingGUI.rotation** = 270.0f

This is the number of degrees to rotate the main camera around the target controller.

2.2.2.4 **SageController SageTestingGUI.targetController** = null

This is the target controller that this Sage Testing GUI should be testing.

2.2.2.5 float **SageTestingGUI.verticalOffset** = 0.75f

This is how much to offset the target controller vertically from the main camera.

2.2.3 Property Documentation

2.2.3.1 **SageTestingGUI SageTestingGUI.Instance** `[static, get]`

This returns the current instance of the Sage Testing GUI.

Index