

Sage Code Reference

1.6.0

Generated by Doxygen 1.7.6.1

Thu Sep 13 2012 21:41:01

Contents

1	Class Index	1
1.1	Class List	1
2	Class Documentation	3
2.1	SageController Class Reference	3
2.1.1	Detailed Description	6
2.1.2	Member Function Documentation	6
2.1.2.1	Awake	6
2.1.2.2	ForceState	6
2.1.2.3	ForceState	6
2.1.2.4	ForceState	7
2.1.2.5	GetFloat	7
2.1.2.6	GetGlobalSpeed	7
2.1.2.7	GetStateMachineCurrentState	8
2.1.2.8	IsStateMachineActive	8
2.1.2.9	IsSuspended	8
2.1.2.10	OnDestroy	8
2.1.2.11	RecacheAnimations	8
2.1.2.12	RecreateLibrary	9
2.1.2.13	Reset	9
2.1.2.14	Resume	9
2.1.2.15	SetApplyMovementDelegate	9
2.1.2.16	SetFloat	9
2.1.2.17	SetFloat	10
2.1.2.18	SetGlobalSpeed	10
2.1.2.19	SetStateEnteredDelegate	10

2.1.2.20	SetStateExitedDelegate	11
2.1.2.21	SetStateUpdatedDelegate	11
2.1.2.22	StartStateMachine	11
2.1.2.23	StartStateMachine	12
2.1.2.24	StartStateMachine	12
2.1.2.25	StopAllStateMachines	12
2.1.2.26	StopAllStateMachines	13
2.1.2.27	StopStateMachine	13
2.1.2.28	StopStateMachine	13
2.1.2.29	Suspend	13
2.1.2.30	SuspendAndPlayAnimation	14
2.1.2.31	SuspendAndPlayAnimations	14
2.1.2.32	TransistionToState	15
2.1.2.33	TransistionToState	15
2.1.2.34	Update	15
2.1.3	Member Data Documentation	15
2.1.3.1	animationTarget	16
2.1.3.2	displayRuntimeInfo	16
2.1.3.3	libraryAsset	16
2.1.3.4	movementTarget	16
2.1.4	Property Documentation	16
2.1.4.1	LibraryRuntime	16
2.2	SageTestingGUI Class Reference	16
2.2.1	Detailed Description	17
2.2.2	Member Data Documentation	17
2.2.2.1	depthOffset	17
2.2.2.2	horizontalOffset	17
2.2.2.3	rotation	17
2.2.2.4	targetController	17
2.2.2.5	verticalOffset	17
2.2.3	Property Documentation	17
2.2.3.1	Instance	17

Chapter 1

Class Index

1.1 Class List

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

[SageController](#)

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 3

[SageTestingGUI](#)

This class is used to test a Sage Library in game. Only one instance of this class can be active at a time 16

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 initialState)
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 allowState-Restart)

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, SageState-EnteredDelegate enteredDelegate)

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

- bool [SetStateUpdatedDelegate](#) (string stateMachine, string state, SageState-UpdatedDelegate updatedDelegate)

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

- bool [SetStateExitedDelegate](#) (string stateMachine, string state, SageStateExited-Delegate exitedDelegate)

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

- bool [SetApplyMovementDelegate](#) (SageApplyMovementDelegate apply-MovementDelegate)

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

<i>state-Machine</i>	The state machine to access.
<i>newState</i>	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

<i>state-Machine</i>	The state machine to access.
<i>newState</i>	The new state to force on the state machine.
<i>transition-Time</i>	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

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

<i>variable-Name</i>	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

<i>state-Machine</i>	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

<i>state-Machine</i>	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

<i>inTime</i>	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

<i>apply-Movement-Delegate</i>	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

<i>variable-Name</i>	The name of the variable to be set.
<i>value</i>	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

<i>variable-Name</i>	The name of the variable to be set.
<i>value</i>	The value to set the variable to.
<i>instant</i>	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

<i>speed</i>	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

<i>state-Machine</i>	The state machine to find the state in.
<i>state</i>	The state to set the entered delegate on.
<i>entered-Delegate</i>	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

<i>state-Machine</i>	The state machine to find the state in.
<i>state</i>	The state to set the exited delegate on.
<i>exited-Delegate</i>	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

<i>state-Machine</i>	The state machine to find the state in.
<i>state</i>	The state to set the updated delegate on.
<i>updated-Delegate</i>	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

<i>state-Machine</i>	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

<i>state-Machine</i>	The name of the state machine to start.
<i>transition-Time</i>	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

<i>state-Machine</i>	The name of the state machine to start.
<i>transition-Time</i>	This is how long (in seconds) it should take to fully start the state machine.
<i>initialState</i>	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

<i>transition-Time</i>	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

<i>state-Machine</i>	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

<i>state-Machine</i>	The name of the state machine to stop.
<i>transition-Time</i>	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

<i>outTime</i>	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

<i>animation</i>	The animation to play.
<i>speed</i>	The speed to playback the animation at.
<i>outTime</i>	This is how long (in seconds) it should take to suspend and blend into this animation.
<i>inTime</i>	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

<i>animations</i>	The list of animations to play, in order.
<i>speed</i>	The speed to playback the animations at.
<i>outTime</i>	This is how long (in seconds) it should take to suspend and blend into these animations.
<i>crossBlendTime</i>	This is how long (in seconds) it should take to blend between animations in the list.
<i>inTime</i>	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

<i>state-Machine</i>	The state machine to access.
<i>newState</i>	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

<i>state-Machine</i>	The state machine to access.
<i>newState</i>	The new state to transistion to on the state machine.
<i>allowState-Restart</i>	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 component 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

- Awake
 - SageController, [6](#)
- ForceState
 - SageController, [6](#), [7](#)
- GetFloat
 - SageController, [7](#)
- GetGlobalSpeed
 - SageController, [7](#)
- GetStateMachineCurrentState
 - SageController, [7](#)
- Instance
 - SageTestingGUI, [17](#)
- IsStateMachineActive
 - SageController, [8](#)
- IsSuspended
 - SageController, [8](#)
- LibraryRuntime
 - SageController, [16](#)
- OnDestroy
 - SageController, [8](#)
- RecacheAnimations
 - SageController, [8](#)
- RecreateLibrary
 - SageController, [8](#)
- Reset
 - SageController, [9](#)
- Resume
 - SageController, [9](#)
- SageController, [3](#)
 - Awake, [6](#)
 - ForceState, [6](#), [7](#)
 - GetFloat, [7](#)
 - GetGlobalSpeed, [7](#)
 - GetStateMachineCurrentState, [7](#)
 - IsStateMachineActive, [8](#)
 - IsSuspended, [8](#)
 - LibraryRuntime, [16](#)
 - OnDestroy, [8](#)
 - RecacheAnimations, [8](#)
 - RecreateLibrary, [8](#)
 - Reset, [9](#)
 - Resume, [9](#)
 - SetApplyMovementDelegate, [9](#)
 - SetFloat, [9](#), [10](#)
 - SetGlobalSpeed, [10](#)
 - SetStateEnteredDelegate, [10](#)
 - SetStateExitedDelegate, [11](#)
 - SetStateUpdatedDelegate, [11](#)
 - StartStateMachine, [11](#), [12](#)
 - StopAllStateMachines, [12](#)
 - StopStateMachine, [13](#)
 - Suspend, [13](#)
 - SuspendAndPlayAnimation, [14](#)
 - SuspendAndPlayAnimations, [14](#)
 - TransistionToState, [15](#)
 - Update, [15](#)
 - animationTarget, [15](#)
 - displayRuntimeInfo, [16](#)
 - libraryAsset, [16](#)
 - movementTarget, [16](#)
- SageTestingGUI, [16](#)
 - Instance, [17](#)
 - depthOffset, [17](#)
 - horizontalOffset, [17](#)
 - rotation, [17](#)
 - targetController, [17](#)
 - verticalOffset, [17](#)
- SetApplyMovementDelegate
 - SageController, [9](#)
- SetFloat
 - SageController, [9](#), [10](#)
- SetGlobalSpeed
 - SageController, [10](#)
- SetStateEnteredDelegate
 - SageController, [10](#)
- SetStateExitedDelegate
 - SageController, [11](#)
- SetStateUpdatedDelegate
 - SageController, [11](#)
- StartStateMachine
 - SageController, [11](#), [12](#)
- StopAllStateMachines

- SageController, [12](#)
- StopStateMachine
 - SageController, [13](#)
- Suspend
 - SageController, [13](#)
- SuspendAndPlayAnimation
 - SageController, [14](#)
- SuspendAndPlayAnimations
 - SageController, [14](#)
- TransistionToState
 - SageController, [15](#)
- Update
 - SageController, [15](#)
- animationTarget
 - SageController, [15](#)
- depthOffset
 - SageTestingGUI, [17](#)
- displayRuntimeInfo
 - SageController, [16](#)
- horizontalOffset
 - SageTestingGUI, [17](#)
- libraryAsset
 - SageController, [16](#)
- movementTarget
 - SageController, [16](#)
- rotation
 - SageTestingGUI, [17](#)
- targetController
 - SageTestingGUI, [17](#)
- verticalOffset
 - SageTestingGUI, [17](#)