

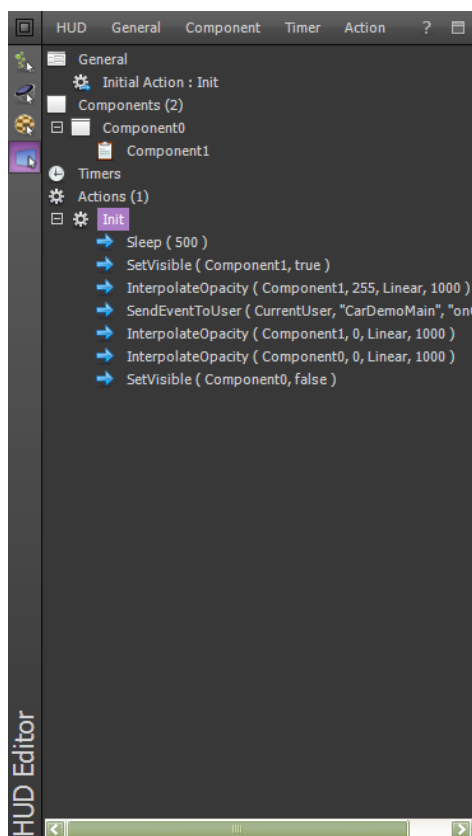
Heads Up!

In this Chapter I'm going to be showing you how to create a HUD using the all-new (in v1.7) WYSIWYG Editor in ShiVa. This is a great new addition to the ShiVa toolset, and will make it much easier to design and create HUDs.

A HUD (Heads Up Display), according to Wikipedia, is:

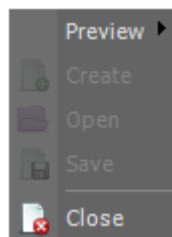
“A template used in data processing to indicate a model of design of software or presentation of data.”

It is the 2D “cover” over the 3D world that displays data such as health, ammo, speed etc. In ShiVa, HUD templates are created using a tree-like structure (see Overview below), where each element (component, timer, action etc.) is displayed in the “tree”. Components from the HUD can be saved, and used multiple times in your applications. Imagine, for example, making a template “Button” in which the behaviour of a generic button is defined. This can then be duplicated and managed easily (via StoneScript) from within any application. So, basically, the HUD Editor allows you to define a GUI (Graphical User Interface), using Containers, Sliders, Labels, Buttons etc.

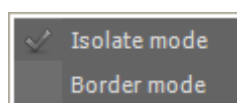


HUD Menu

The HUD menu allows the creation, opening, saving, closing, and previewing of your HUD:

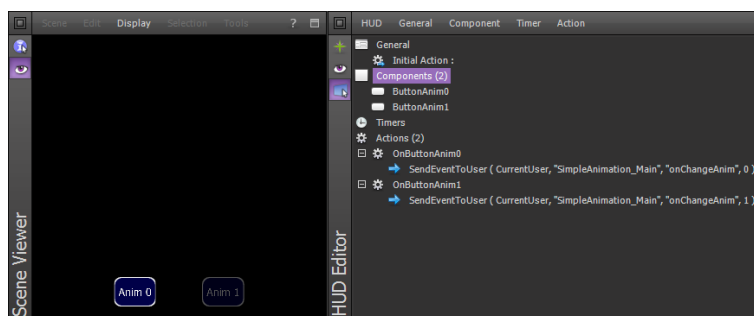


Preview: This option opens the following menu, which allows the setting of the options for the preview of the HUD in the Scene Viewer:



NOTE: the  next to the mode in the menu indicates that the particular mode is set.

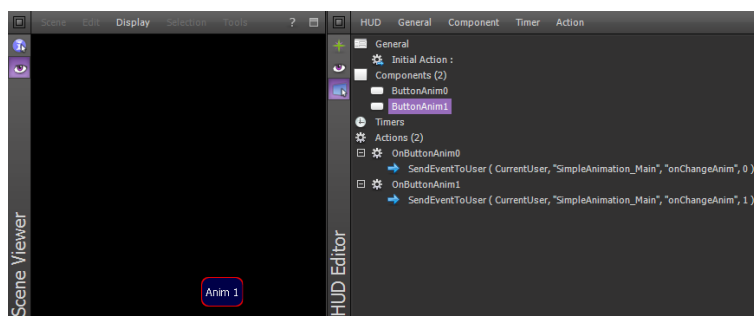
Isolate mode: This option sets, or unsets, the isolate mode of the preview. This mode displays the Components (when the “Components” heading has been selected to display all Components), with the last selected Component being the one with the focus:



ButtonAnim0 was the last selected Component.

NOTE: if “Isolate mode” is unset, ButtonAnim1 would be shown with the same brightness as ButtonAnim0.

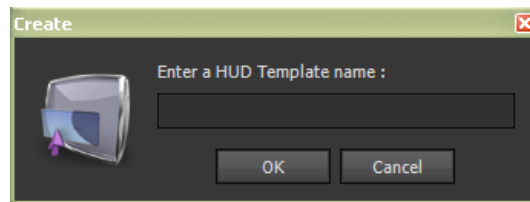
Border mode: This option sets, or unsets, the border mode of the preview. This mode displays a red border around the selected Component:



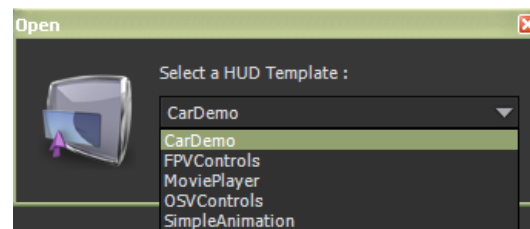
ButtonAnim1 is the selected Component.

NOTE: Border mode only displays the red border when a single Component is selected.

Create: This option will create a new HUD template. A dialog will appear asking for the name of the new HUD:



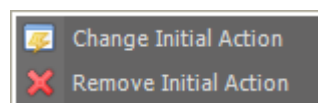
Open: This option will open any HUD template that is available in the current project. A dialog will appear that has a drop-down selection of all of the HUD templates available in the current project:



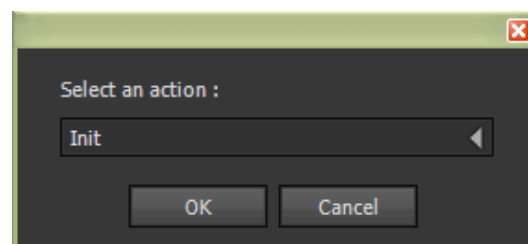
Save: This option will save the current HUD template.

Close: This option will close the current HUD template.

General Menu

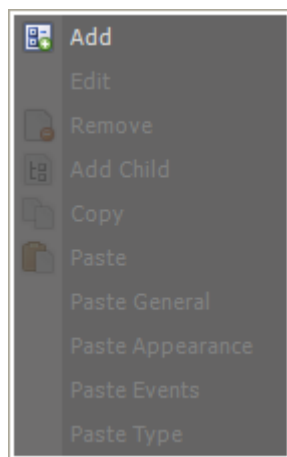


Change Initial Action: This option will define an action that will be called automatically with the instantiation of the HUD template. A dialog will appear enabling the selection of all available actions:



Remove Initial Action: This option will clear the initial action of the HUD template.

Component Menu



Add: This option will add a new Component to the current HUD template, immediately after the last one in the current template. This will open the “Component” dialog, which is described later.

Edit: This option will allow the editing of a selected Component. This will open the “Component” dialog.

Remove: This option will delete a selected Component and its children (if it has any).

NOTE: you will be asked if you want to proceed with the deletion.

Add Child: This option will add a child to a selected Component. This will open the “Component” dialog.

Copy: This option will copy a Component and place it in the clipboard.

NOTE: this will only copy the root Component, not its children.

Paste: This option will paste a copied Component from the clipboard.

Paste General: This option will paste a Copied component from the clipboard.

NOTE: this will only paste the general properties of the copied Component into the selected Component.

Paste Appearance: This option will paste a copied Component from the clipboard.

NOTE: this will only paste the appearance properties of the copied Component into the selected Component.

Paste Events: This option will paste a copied Component from the clipboard.

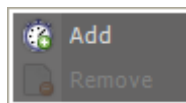
NOTE: this will only paste the events properties of the copied Component into the selected Component.

Paste Type: This option will paste a copied Component from the clipboard.

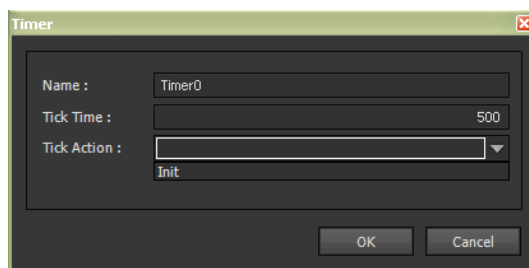
NOTE: this will only paste the type properties of the copied Component into the selected Component.

Timer Menu

HUD Timers are similar to stopwatches, and can be started, paused and stopped at will. They are generally used to make HUD Actions occur at the correct time.



Add: This option will add a new Timer to the current HUD template, immediately after the last one in the current template. This will open the following dialog:



Name: This option will set the name of the new Timer.

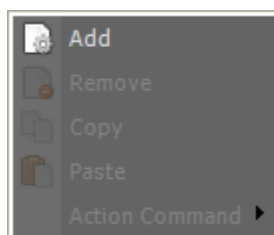
Tick Time: This option will set the time between ticks for the new Timer (in milliseconds).

Tick Action: This option will set the action to be performed on each tick, via a drop-down list of all available actions.

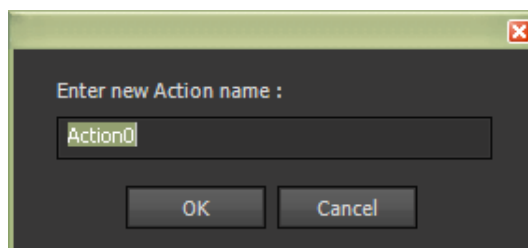
Remove: This option will delete the selected Timer.

NOTE: you will be asked if you want to proceed with the deletion.

Action Menu



Add: This option will add a new Action to the current HUD template, immediately after the last one in the current template. A dialog will appear asking for the name of the new Action:



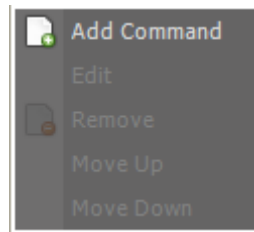
Remove: This option will delete a selected Action and its children (if there are any).

NOTE: you will be asked if you want to proceed with the deletion.

Copy: This option will copy a selected Action, and its children, to the clipboard.

Paste: This will paste a previously copied Action, and its children, from the clipboard.

Action command: This option will open the following menu:



Add Command: Adds a new Command to the selected Action.

Edit: Edits the selected Action Command.

Remove: Removes the selected Action Command.

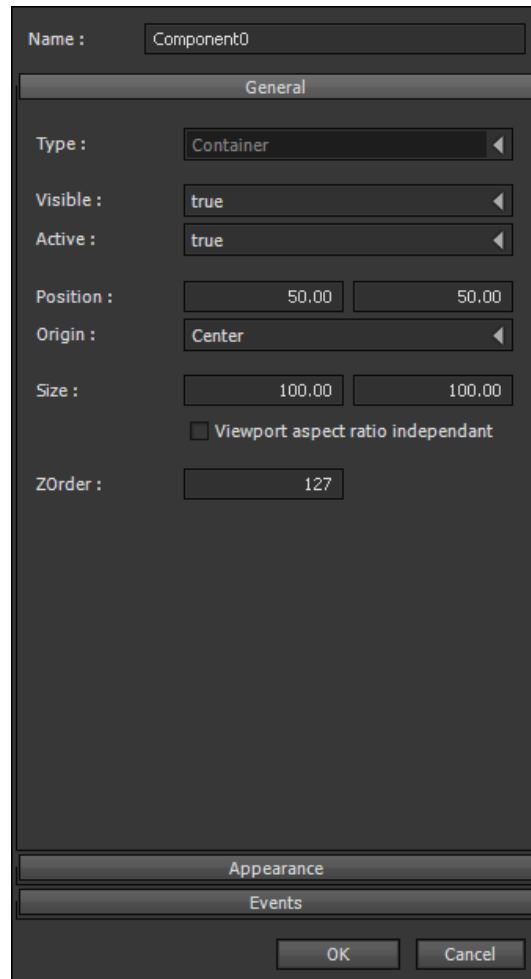
Move Up: Move the selected Action Command UP the tree.

Move Down: Moves the selected Action Command DOWN the tree.

Components

When you select either the “New” or the “Edit” option on the “Component” menu, the HUD Editor changes to look as follows:

NOTE: use the “Esc” key to switch back without saving any changes

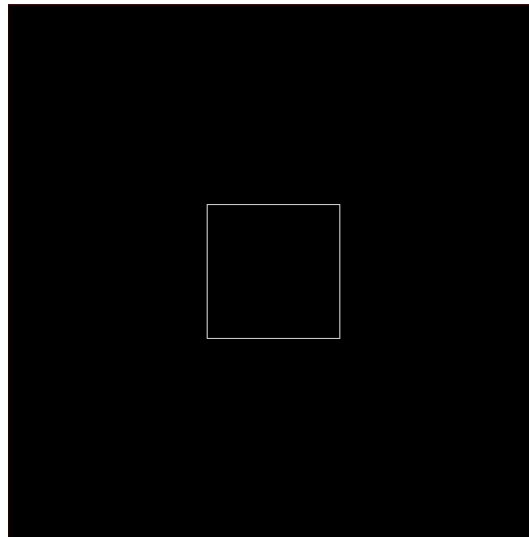


General:

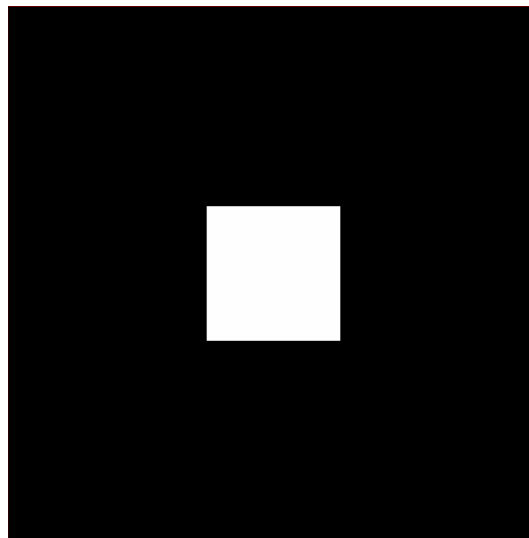
Type: The component type, selectable from a drop-down list of all of the available types:

- Container
- Label
- Edit
- Button
- Progress
- DVInput
- Movie
- List
- Slider
- RenderMap

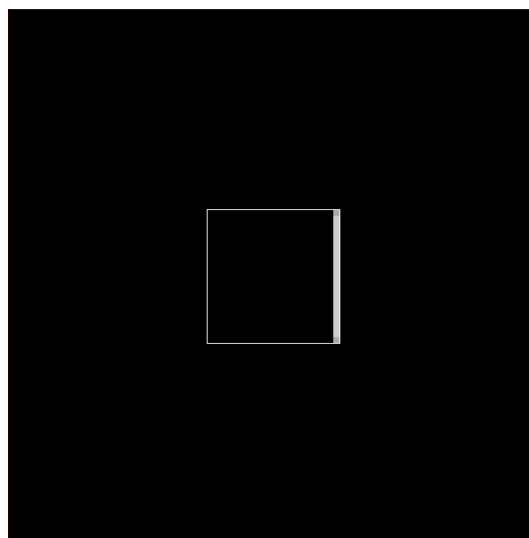
When each type is selected, you will see it display in the Scene Viewer, as shown below:



Container / Label / Edit / Button / Progress / DVInput



Movie / Slider / RenderMap



List

As you can see, there's not much difference between the types as they all have the same size etc. at the moment.....

Visible: Defines if the Component is visible, or not, at instantiation of the HUD.

Active: Defines if the Component is active, or not, at the instantiation of the HUD.

Position: Defines the x and y positions of the Component relative to the bottom left window corner, in percentages.

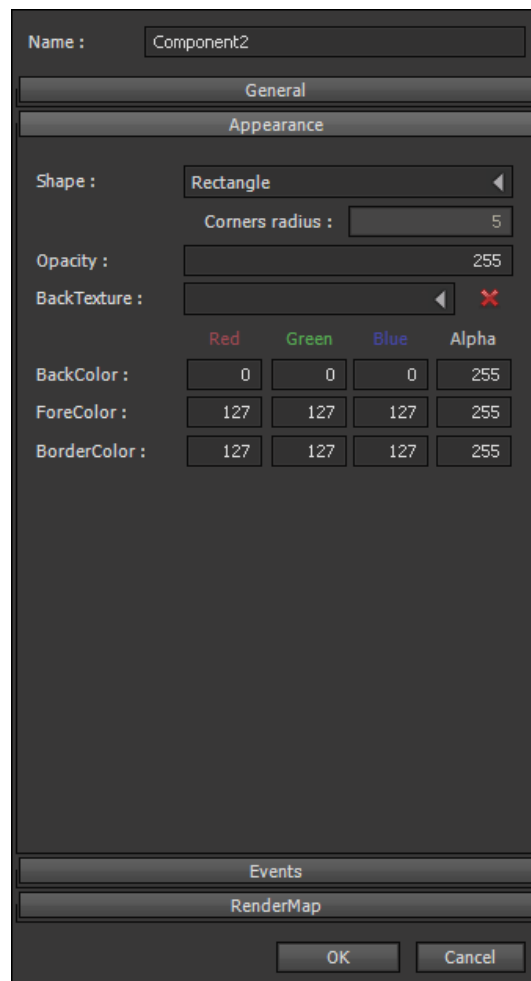
Origin: This is the origin of the Component, like a pivot.

Size: This is the size of the Component, in percentage of the window size.

Viewport aspect ratio independent: If selected, HUD dimensions don't depend on the viewport ratio (1:1), and will stay the same size irrespective of the Screen resolution.

ZOrder: This is the position of the HUD template relative to depth [0-255].

Appearance:



Name : Component2

General

Appearance

Shape : Rectangle

Corners radius : 5

Opacity : 255

BackTexture : [Empty] X

	Red	Green	Blue	Alpha
BackColor :	0	0	0	255
ForeColor :	127	127	127	255
BorderColor :	127	127	127	255

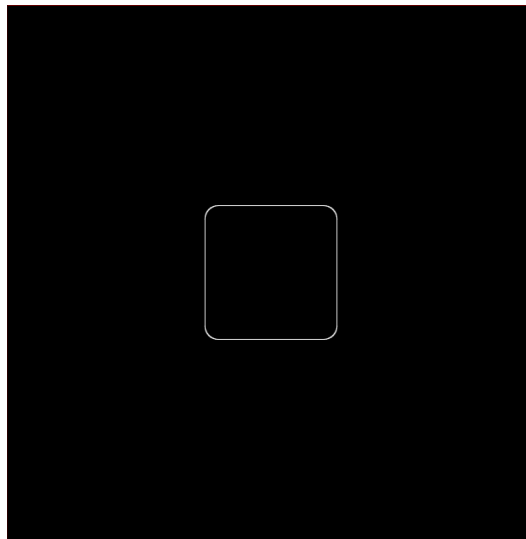
Events

RenderMap

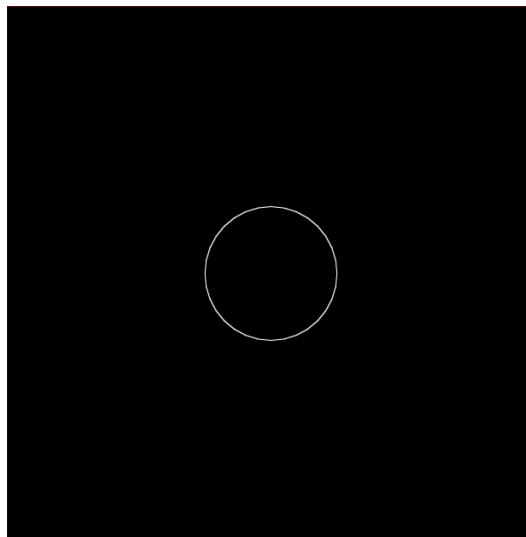
OK Cancel

These options alter the appearance of the Component.

Shape: Rectangle (as shown above), Round Rectangle or ellipsoid.



Round Rectangle



Ellipsoid

Corners radius: For “Round Rectangle” only, defines the corner size.

NOTE: a Round Rectangle with “Corners radius” of “50” is a circle, and with a “Corners radius” of “0” is a square.

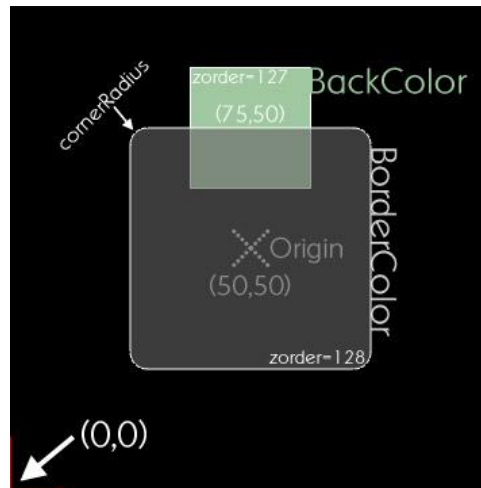
Opacity: Defines this Component’s opacity (0 to 255).

Back texture: Enables the use of a Texture instead of a background colour (computes the alpha channel).

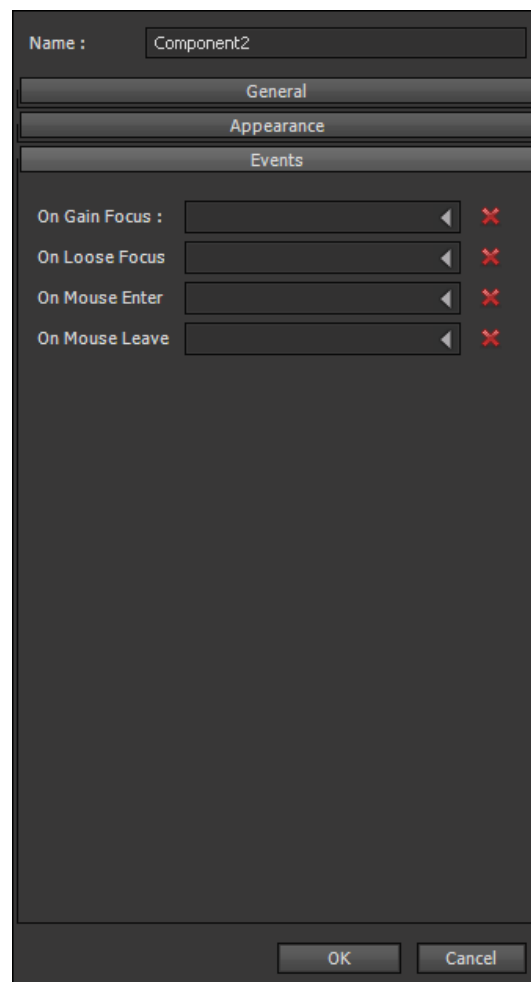
Back Color: The background colour of the Component (normal colour range 0 to 127).

Fore Color: The foreground colour of the Component (normal colour range 0 to 127).

Border Color: The border colour of the Component (normal colour range 0 to 127).



Events:



On Gain Focus: This option allows the selection of an Action to be performed when the Component gets the focus. The Action can be selected from the drop-down list containing all currently available Actions.

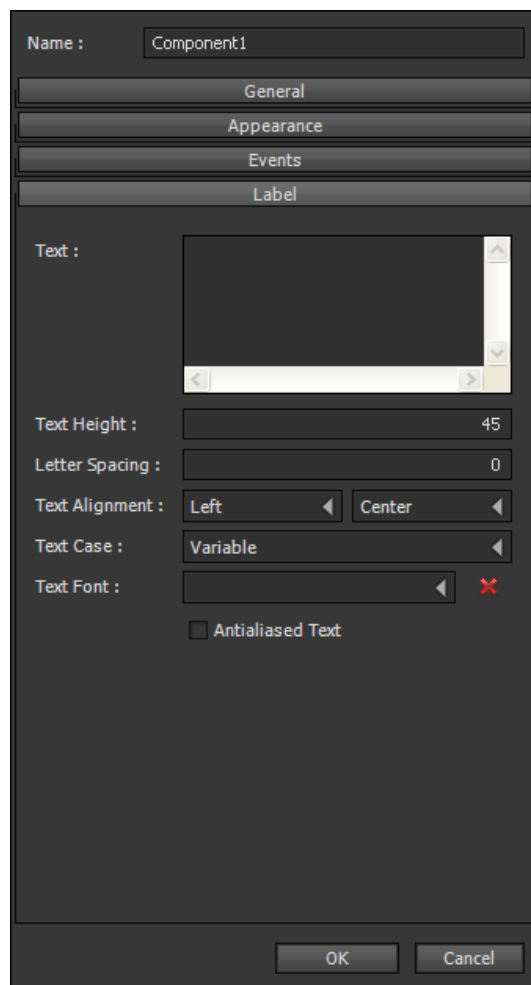
On Loose Focus: This option allows the selection of an Action to be performed when the Component loses the focus. The Action can be selected from the drop-down list containing all currently available Actions.

On Mouse Enter: This option allows the selection of an Action to be performed when the Mouse cursor enters the Component. The Action can be selected from the drop-down list containing all currently available Actions.

On Mouse Leave: This option allows the selection of an Action to be performed when the Mouse cursor leaves the Component. The Action can be selected from the drop-down list containing all currently available Actions.

Now we will move on to the Component types themselves:

Label:



This Component is used to display Text in the HUD.

NOTE: **Fore Color** is the colour of the Text.

Text: This input box is where the Text to be displayed in the Component is entered.

Text Height: This option sets the size of the Font, as a percentage of the height of the Component [0 to 100].

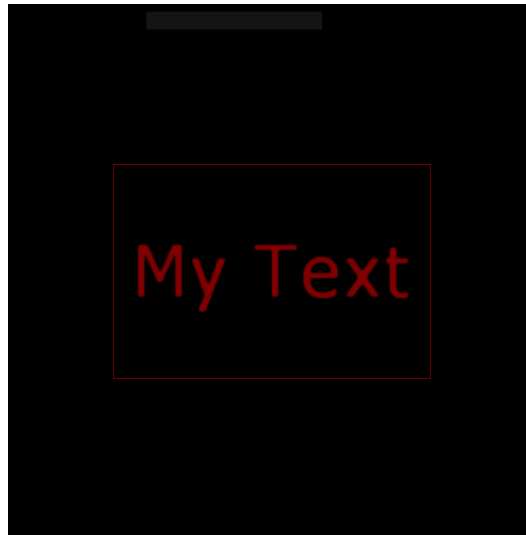
Letter Spacing: This option sets the spacing between the letters of the Text [-100 to 100].

Text Alignment: These options set the alignment of the Text in the Component. The first drop-down lists the options for the horizontal alignment (Centre, Left, Right & Justify), and the second lists the options for the vertical alignment (Centre & Top).

Text Case: This option sets the spacing between the letters, either fixed or variable.

Text Font: This option sets the font Texture to be used. The drop-down lists all available Fonts currently loaded into the application.

AntiAliased Text: This option sets whether the Text is antialiased, or not, using mipmaps.



Edit:

Name :

General

Appearance

Events

Edit

Text :

Text Height :

Letter Spacing :

Text Alignment :

Text Case :

Text Font :

☒ Antialiased Text

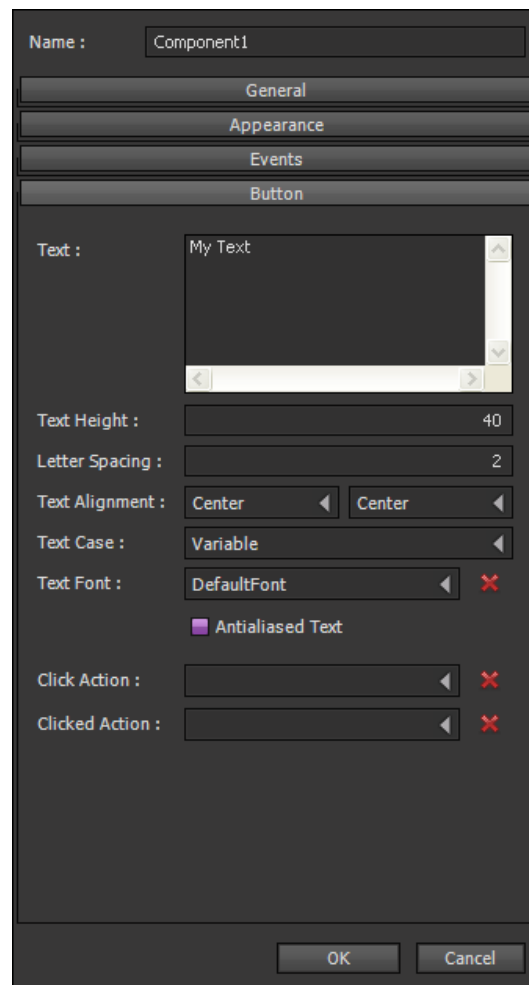
Changed Action :

This Component is used this to display user editable Text.

The majority of the Parameters are identical to Label, except for “Changed Action”:

Changed Action: This option allows the setting of an Action that will be called when the Text has been changed. The drop-down lists all Actions currently defined in the HUD.

Button:



This Component is used to allow the creation of a Button, which is similar to a Label, but has the capability of responding to Mouse clicks.

The majority of the Parameters are identical to Label, with the exception of “Click Action”, and “Clicked Action”:

Click Action: This option allows the setting of an Action that will be called when the Mouse button is first pressed whilst the Cursor is over the Button.

Clicked Action: This option allows the setting of an Action that will be called when the Mouse button is released.

Progress Bar:

Name :

General

Appearance

Events

Progress

Value :

Type :

OK Cancel

This Component is used to display a progress bar.

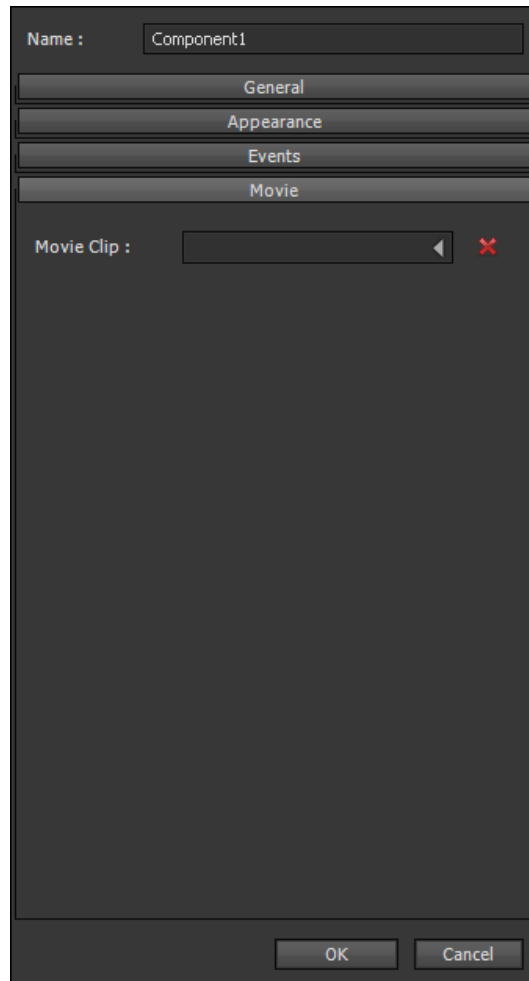
For this Component type, **Back Color** is the progress bar colour, and **Fore Color** becomes the background colour. Also a Texture can be added to this Component, if required.

Value: This option sets the initial progress value [0 to 255].

Type: This option allows the selection of the type of Progress Bar (“Left to Right”, “Right to Left”, “Bottom to Top” or “Top to Bottom”).

DVInput:

This Component displays the output from a video camera in the HUD, and does not have its own roll-up.

Movie:

This Component allows the playing of a Movie Clip in the HUD.

Movie Clip: This option allows the selection of a Movie Clip to be displayed in the HUD. The drop-down lists all of the Movie Clips available in the current application.

List:

Name : Component1

General

Appearance

Events

List

Items Height : 10

Text Height : 50

Letter Spacing : 2

Text Margins : 0 0

Text Case : Variable

Text Font : DefaultFont

☒ Antialiased Text

Selection Action

☐ Use single selection

☐ Use smooth scrolling

☐ User can select items

List Items

List Scrollbars

OK Cancel

The options in this roll-up allow the setting of various attributes for the items in the List:

Items Height: This option allows the setting of the height (as a percentage of the total height for the row) of each of the items in the List [0 to 100].

Text Height: This option sets the size of the Font, as a percentage of the height of the Component [0 to 100].

Letter Spacing: This option sets the spacing between the letters of the Text [-100 to 100].

Text Alignment: These options set the alignment of the Text in the Component. The first drop-down lists the options for the horizontal alignment (Centre, Left, Right & Justify), and the second lists the options for the vertical alignment (Centre & Top).

Text Case: This option sets the spacing between the letters, either fixed or variable.

Text Font: This option sets the font Texture to be used. The drop-down lists all available Fonts currently loaded into the application.

AntiAliased Text: This option sets whether the Text is antialiased, or not, using mipmaps.

Selection Action: This option allows the selection of the Action to be performed when an item in the List is selected. The drop-down lists all of the Actions available to the current HUD.

Use Single Selection: This option allows the selection of items in the List to be restricted to a single item (if checked), or multiple items (if unchecked).

Use Smooth Scrolling: This option allows the use of “Smooth Scrolling” (if checked), or normal scrolling (if unchecked).

User Can Select Items: This option allows the setting of whether the User can select items in the List (if checked), or if it is just for display purposes only (if unchecked).

List Items:

Name : Component1

General

Appearance

Events

List

List Items

BackImage :

BackImage Sel :

	Red	Green	Blue	Alpha
BackColor Even	68	68	68	204
BackColor Odd	68	68	68	204
BackColor Sel	68	68	68	255
ForeColor Sel	255	255	0	255

List Scrollbars

OK Cancel

The options in this roll-up allows for the setting of various attributes for the List.

BackImage: This option allows the setting of a Texture as the background image for an item in the List. The drop-down lists the Textures that are available in the current application.

BackImageSel: This option allows the setting of a Texture as the background image for an item in the List when it has been selected. The drop-down lists the Textures that are available in the current application.

BackColorEven: This option allows the setting of the background colour for even rows in the List.

BackColorOdd: This option allows the setting of the background colour for odd rows in the List.

BackColorSel: This option allows the setting of the background colour for an item in the List when it is selected.

ForeColorSel: This option allows the setting of the foreground colour of an item in the List when it is selected.

List Scrollbars:

	Red	Green	Blue	Alpha
Arrow :	85	85	85	255
Back Color :	0	0	0	255
Fore Color :	102	102	102	255

Arrow Top :		✖
Arrow Bottom :		✖
Back Top :		✖
Back Middle :		✖
Back Bottom :		✖
Fore Top :		✖
Fore Middle :		✖
Fore Bottom :		✖

OK Cancel

This roll-up allows the setting of various attributes of the Scrollbars for the List:

Arrow: This option allows the setting of the colour of the Arrow to be used on the Scrollbars.

Back Color: This option allows the setting of the background colour of the Scrollbars.

Fore Color: This option allows the setting of the foreground colour of the Scrollbars.

Arrow Top: This option allows the selection of the Texture to be used as the top Arrow. The drop-down lists all Textures available in the current application.

Arrow Bottom: This option allows the selection of the Texture to be used as the bottom Arrow. The drop-down lists all Textures available in the current application.

Back Top: This option allows the selection of the Texture to be used as the background image for the top of the Scrollbar. The drop-down lists all Textures available in the current application.

Back Middle: This option allows the selection of the Texture to be used as the background image for the middle of the Scrollbar. The drop-down lists all Textures available in the current application.

Back Bottom: This option allows the selection of the Texture to be used as the background image for the bottom of the Scrollbar. The drop-down lists all Textures available in the current application.

Fore Top: This option allows the selection of the Texture to be used as the foreground image of the top of the Scrollbar. The drop-down lists all Textures available in the current application.

Fore Middle: This option allows the selection of the Texture to be used as the foreground image for the middle of the Scrollbar. The drop-down lists all Textures available in the current application.

Fore Bottom: This option allows the selection of the Texture to be used as the foreground image for the bottom of the Scrollbar. The drop-down lists all Textures available in the current application.

Slider:

Name : Component1

General

Appearance

Events

Slider

Value : 0

Type : Left to Right

Range Min : 0

Range Max : 255

Thumb Texture :

Changed Action :

OK Cancel

The options in this roll-up allow the setting of various attributes for the Slider Component:

Value: This option allows the setting of the value of the Slider on initialisation [0 to 255].

Type: This option allows the selection of the type of the Slider (“Left to Right”, “Right to Left”, “Bottom to Top” or “Top to Bottom”).

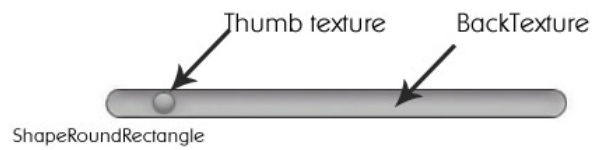
Range Min: This option allows the setting of the minimum value of the Slider [-1000 to 1000].

Range Max: This option allows the setting of the maximum value of the Slider [-1000 to 1000].

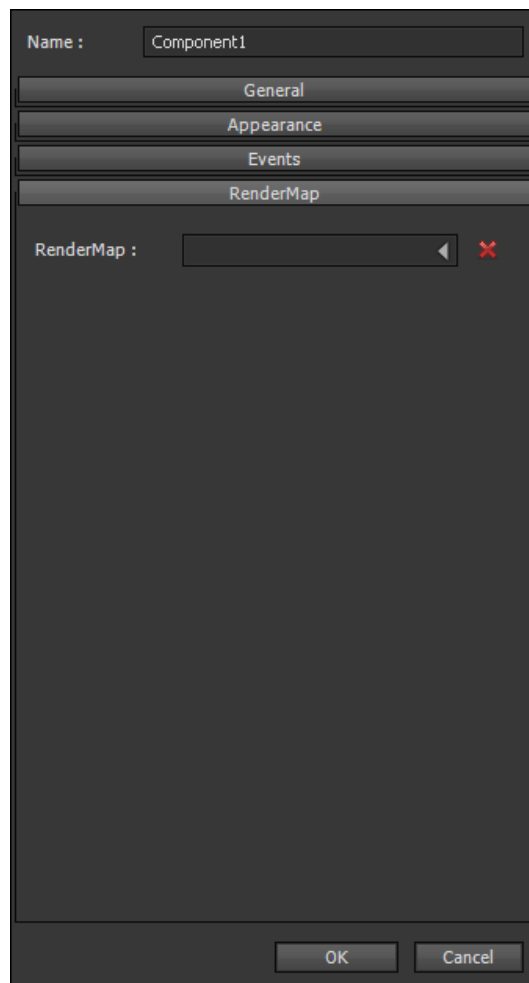
Changed Action: This option allows the setting of the Action to be performed when the Slider is moved. The drop-down lists all available Actions in the current HUD.

Thumb Texture: This option allows the selection of the Texture to be used as the “Thumb” image for the Slider. The drop-down lists all Textures available in the current application.

Back Texture: This option allows the selection of the Texture to be used as the background image for the Slider. The drop-down lists all Textures available in the current application.



Render Map:



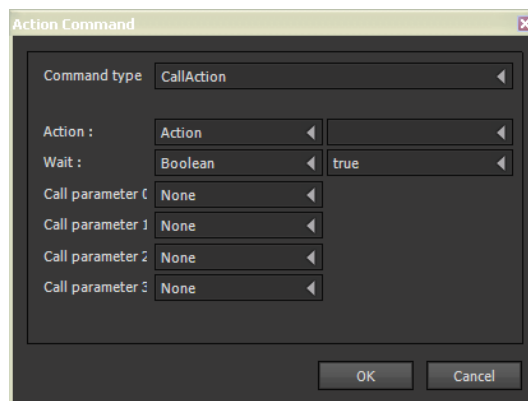
This roll-up allows you to select the Render Map to be applied to the RenderMap Component:

RenderMap: This option allows the selection of the Render Map to be used for this Component. The drop-down lists all Render Maps that are available in the current application.

Actions

Each Action is made up of several commands, which are carried out sequentially. Actions bring the HUD to life and add User interactivity.

CallAction:



This Action allows the calling of another Action that has been defined within the HUD.

Action: This option allows the selection of the Action to be called. The first drop-down contains only one value “Action”, and the second contains a list of all available Actions.

Wait: This option allows the setting of the Parameter to be used as the value on which the current Action will continue after the call to the “Called” Action. Note that the value of the second drop-down will change dependant on the value of the first drop-down:

Drop-down1 Options:

Boolean

RuntimeValue

Drop-down2 Options:

True

False

CallArgument0

CallArgument1

CallArgument2

CallArgument3

NOTE: To make this Chapter smaller, I will not be repeating the definitions of drop-downs where they appear in other Actions.

CallParameter0: This option allows the setting of Parameters to be passed to the “Called” Action. Note that the values allowed in the second box (which appears when a selection is made in the first) will change dependant on the value of the first drop-down:

Drop-down1 Options:

None

Boolean

Number

String

RuntimeValue

Box2 Options:

Not Applicable

True

False

-1000000 to 1000000

Any String of valid characters

CallArgument0

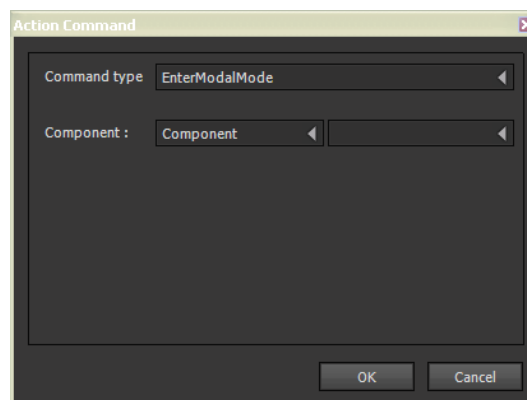
CallArgument1

CallArgument2

CallArgument3

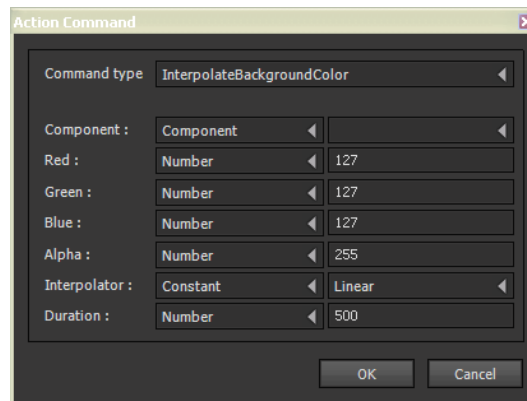
NOTE: CallParameter1, CallParameter2 and CallParameter3 are identical to CallParameter0.

EnterModalMode:



This Action sets the selected Component, that has been defined in the HUD, to be in “Modal” mode.

Component: This option allows the selection of the Component. The first drop-down contains only one value “Component”, and the second contains a list of all available Components.

InterpolateBackgroundColor:

This Action allows the setting of the Interpolator to be used on the background colour of the selected Component.

Red: This option allows the setting of the “Red” value of the colour. Note that the values allowed in the second box will change dependant on the value of the first drop-down:

Drop-down1 Options:

Number

RuntimeValue

Box2 Options:

0 to 255

CallArgument0

CallArgument1

CallArgument2

CallArgument3

NOTE: Green, Blue and Alpha are identical to Red, and set the “Green”, “Blue” and “Alpha” components of the colour accordingly.

Interpolator: This option allows the setting of the “Interpolator” type to be used. The first drop-down contains one value “Constant”, and the second contains the following options (I won’t go into these in detail, as they are beyond the scope of this book):

Linear:
 Power2
 Power3
 Power4
 Root2
 Root3
 Root4
 Spring1
 Spring2
 Spring3
 Spring4
 Spring5
 Spring6

Duration: This option allows the setting of the length of the interpolation (in milliseconds). Note that the values allowed in the second box will change dependant on the value of the first drop-down:

Drop-down1 Options:

Number

RuntimeValue

Box2 Options:

0 to 100000000

CallArgument0

CallArgument1

CallArgument2

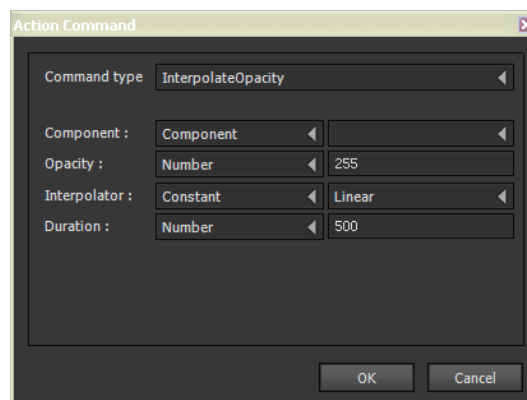
CallArgument3

InterpolateBorderColor:

This Action is identical to “InterpolateBackgroundColor”, but interpolates the border colour of the Component.

InterpolateForegroundColor:

This Action is identical to “InterpolateBackgroundColor”, but interpolates the foreground colour of the Component.

InterpolateOpacity:

This Action is similar to the “InterpolateBackgroundColor”, but interpolates the Opacity of the Component.

Opacity: This option allows the setting of the “Opacity” value of the Component. Note that the values allowed in the second box will change dependant on the value of the first drop-down:

Drop-down1 Options:

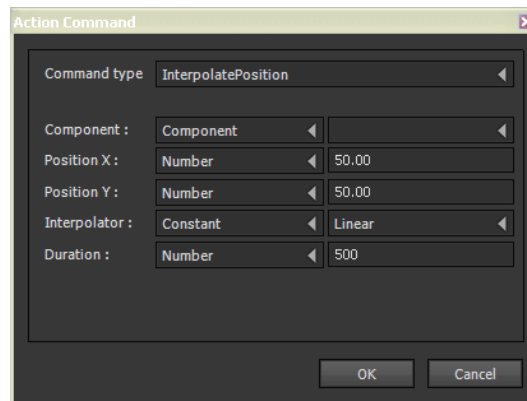
Number

RuntimeValue

Box2 Options:

0 to 255

CallArgument0..3

InterpolatePosition:

This Action is again similar to the other “Interpolate” Actions, but this time allows the setting of the values required to interpolate the actual position of the Component.

Position X: This option allows the setting of the “X” position value of the Component. Note that the values allowed in the second box will change dependant on the value of the first drop-down:

Drop-down1 Options:

Number

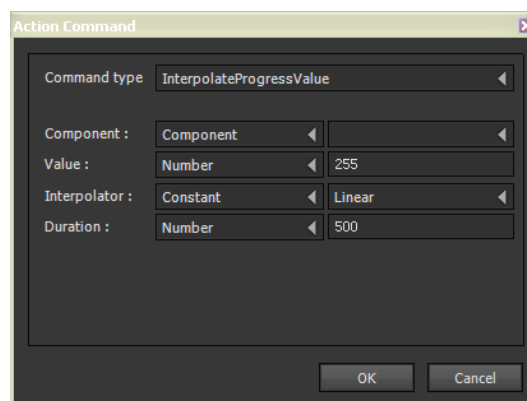
RuntimeValue

Box2 Options:

-1000 to 1000

CallArgument0..3

NOTE: Position Y is identical to Position X, but sets the “Y” position value of the Component.

InterpolateProgressValue:

This Action is again similar to the other “Interpolate” Actions, but this time allows the setting of the values required to interpolate the progress value of the Component.

Value: This option allows the setting of the “Progress” value of the Component. Note that the values allowed in the second box will change dependant on the value of the first drop-down:

Drop-down1 Options:

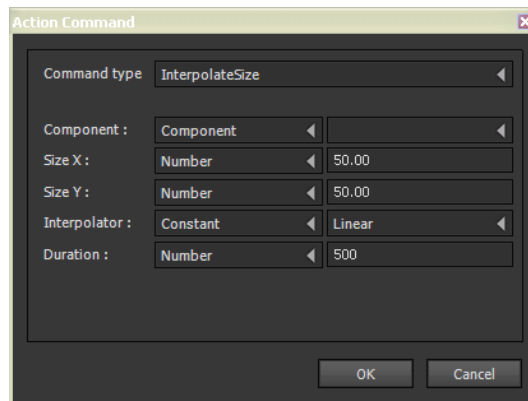
Number

RuntimeValue

Box2 Options:

0 to 255

CallArgument0..3

InterpolateSize:

This Action is again similar to the other “Interpolate” Actions, but this time allows the setting of the values required to interpolate the actual size of the Component.

Size X: This option allows the setting of the “X” size value of the Component. Note that the values allowed in the second box will change dependant on the value of the first drop-down:

Drop-down1 Options:

Number

RuntimeValue

Box2 Options:

0 to 1000

CallArgument0..3

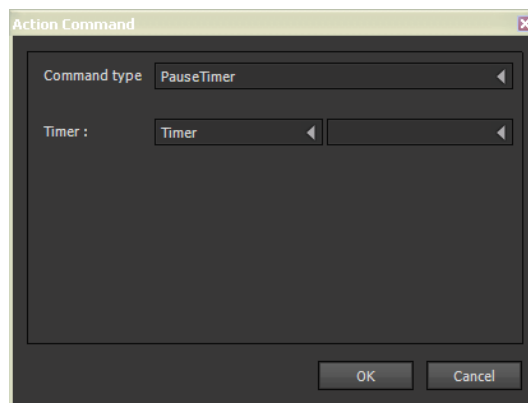
NOTE: Size Y is identical to Size X, but sets the “Y” size value of the Component.

LeaveModalMode:

This Action is identical to “EnterModalMode”, but makes the selected Component leave “Modal” mode.

PauseMovie:

This Action is identical to “EnterModalMode”, but pauses any Movie currently playing in the selected Component.

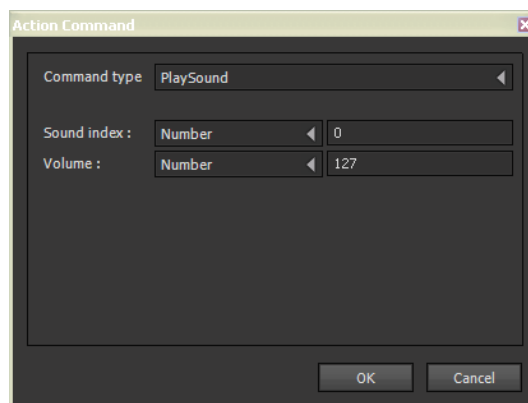
PauseTimer:

This Action is almost identical to “EnterModalMode”, but causes the selected Timer to pause.

Timer: This option allows the selection of the Timer to be paused. The first drop-down contains one value “Timer”, and the second contains a list of all available Timers (that have been defined in the HUD).

PlayMovie:

This Action is identical to “EnterModalMode”, but plays any Movie that has been attached to the selected Component.

PlaySound:

This Action allows the playing of a Sound.

Sound index: This option allows the setting of the Index of the Sound to be played. Note that the values allowed in the second box will change dependant on the value of the first drop-down:

Drop-down1 Options:

Number

RuntimeValue

Box2 Options:

0 to 64

CallArgument0..3

Volume: This option allows the setting of the volume of the Sound to be played. Note that the values allowed in the second box will change dependant on the value of the first drop-down:

Drop-down1 Options:

Number

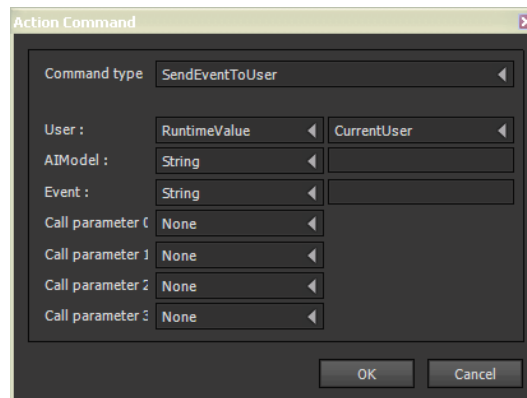
RuntimeValue

Box2 Options:

0 to 255

CallArgument0..3

SendEventToUser:



This Action allows the sending of an Event to a User AI.

User: This option allows the selection of the User. Currently, the only options available are “RuntimeValue” for the first drop-down, and “CurrentUser” for the second.

AIModel: This option allows the selection of the AI Model to send the Event to. Note that the values allowed in the second box will change dependant on the value of the first drop-down:

Drop-down1 Options:

String

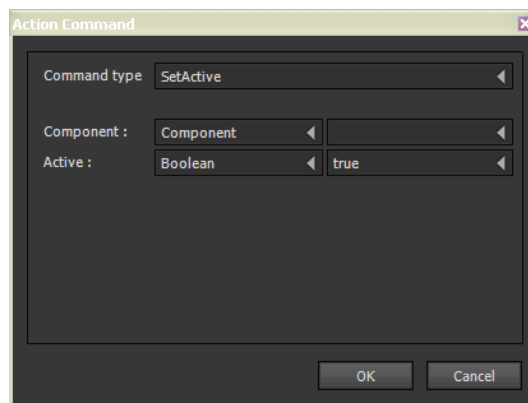
RuntimeValue

Box2 Options:

Any String of valid characters

CallArgument0..3

Event: This option allows the selection of the Event to be sent, and has identical options to “AIModel”.

SetActive:

This Action sets the selected Component to be “Active”.

Active: This option allows the selection of the activation condition. Note that the values allowed in the second box will change dependant on the value of the first drop-down:

Drop-down1 Options:

Boolean

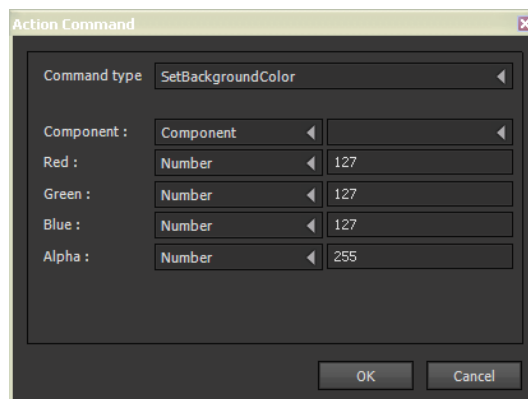
RuntimeValue

Box2 Options:

True

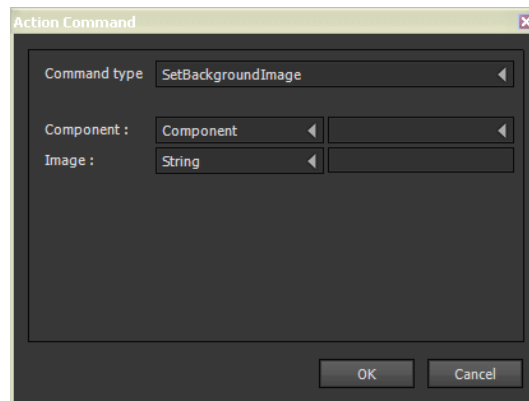
False

CallArgument0..3

SetBackgroundColor:

This Action sets the background colour of the Component.

The options are identical to “InterpolateBackgroundColor”, except that there are no options for “Interpolator” or “Duration”.

SetBackgroundImage:

This Action sets the background image of the selected Component.

Image: This option allows the selection of the image to be used. Note that the values allowed in the second box will change dependant on the value of the first drop-down:

Drop-down1 Options:

String

RuntimeValue

Box2 Options:

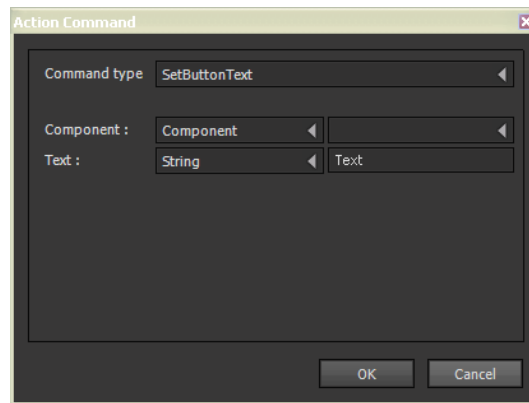
A valid String of characters

CallArgument0..3

SetBorderColor:

This Action sets the border colour of the Component.

The options are identical to “InterpolateBackgroundColor”, except that there are no options for “Interpolator” or “Duration”.

SetButtonText:

This Action sets the Text to be used on a Button Component.

Text: This option allows the input of the Text to be used. Note that the values allowed in the second box will change dependant on the value of the first drop-down:

Drop-down1 Options:

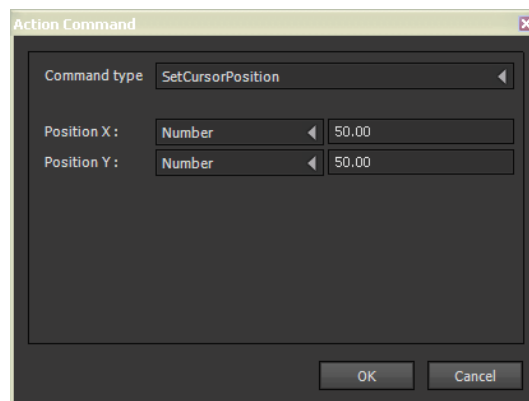
String

RuntimeValue

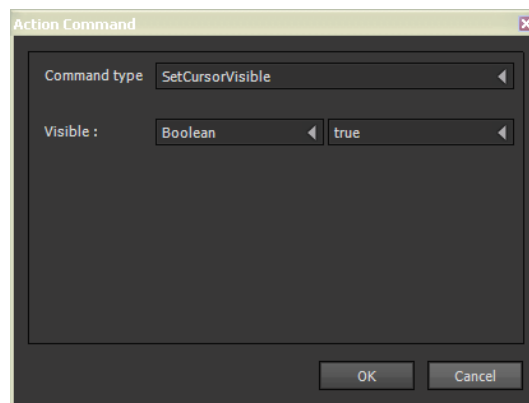
Box2 Options:

A valid String of characters

CallArgument0..3

SetCursorPosition:

This Action sets the position of the Cursor.

SetCursorVisible:

This Action sets whether the Cursor is visible or not.

Visible: This option allows the setting of the visibility of the Cursor. Note that the value of the second drop-down will change dependant on the value of the first drop-down:

Drop-down1 Options:

Boolean

RuntimeValue

Drop-down2 Options:

True

False

CallArgument0..3

SetEditText:

This Action sets the Text to be used on an Edit Component, and is identical to “SetButtonText”.

SetFocus:

This Action sets whether the selected Component has the current focus, and is identical to “EnterModalMode”.

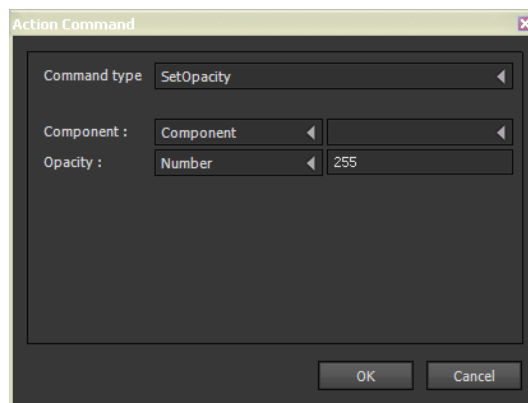
SetForegroundColor:

This Action sets the foreground colour of the selected Component, and is identical to “SetBackgroundColor”.

SetLabelText:

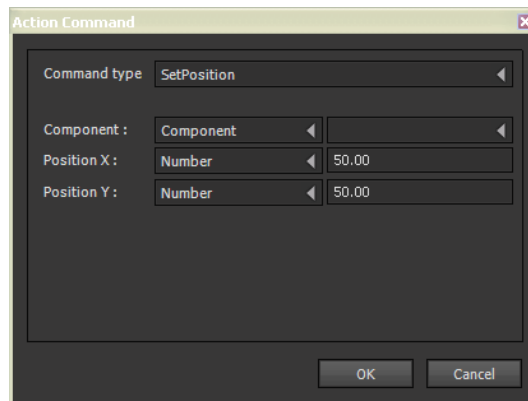
This Action sets the Text to be used on a Label Component, and is identical to “SetButtonText”.

SetOpacity:



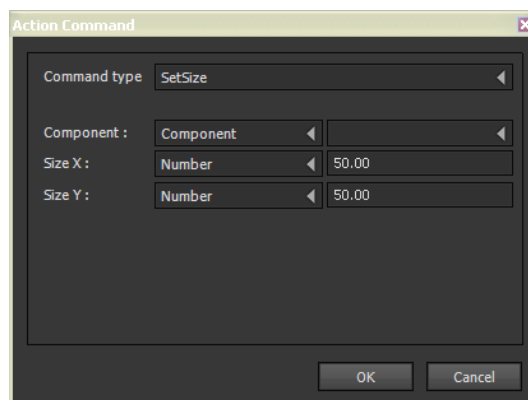
This Action sets the Opacity of the selected Component.

SetPosition:



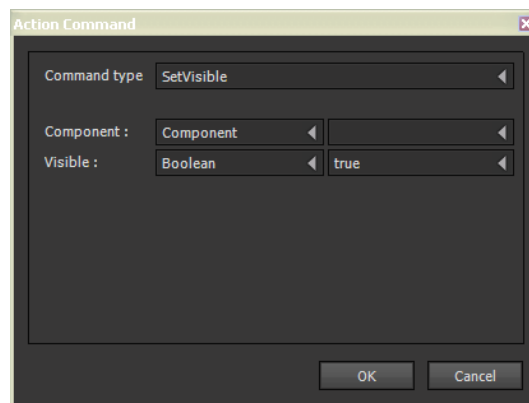
This Action sets the Screen position of the selected Component.

SetSize:



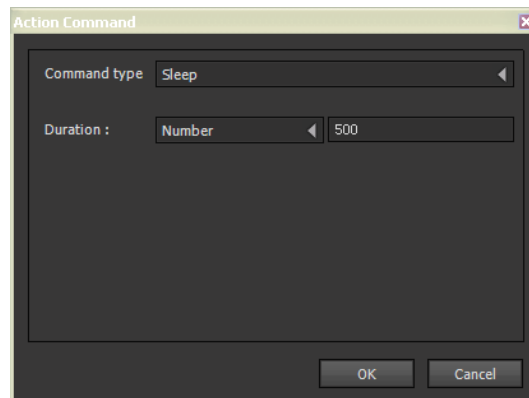
This Action sets the size of the selected Component.

SetVisible:



This Action sets whether the selected Component is visible, or not.

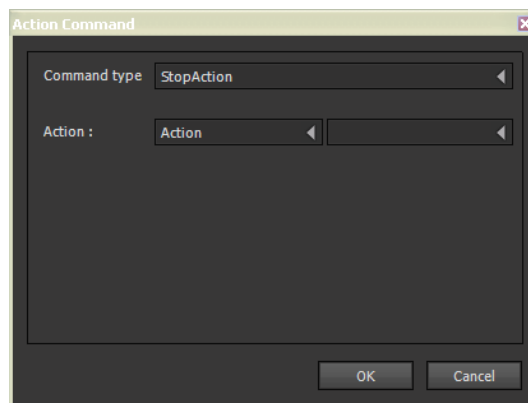
Sleep:



This Action stops all Action processing for the period specified (in milliseconds).

StartTimer:

This Action starts the specified Timer, and is identical to "PauseTimer".

StopAction:

This Action stops the specified Action from running.

Action: This option allows the selection of the Action to be stopped. The first drop-down only has one possible value “Action”, and the second lists all Actions defined in the current HUD.

StopMovie:

This Action stops a Movie from playing in the selected Component, and is identical to “EnterModalMode”.

StopTimer:

This Action stops the specified Timer, and is identical to “PauseTimer”.

As you can tell from the previous pages, the ShiVa HUD Editor contains many features that are commonly found in GUIs. For this Chapter, I’ll go through another ShiVa demo app that shows you how to create a little movie player.

The demo for this is called “MoviePlayer”, and can be found in the standard ShiVa demos (“Installation_Samples.zip”).

So, create a new project in ShiVa, and we’ll get started.

This demo consists of the following files (in addition to those automatically created by ShiVa when you create a new project):

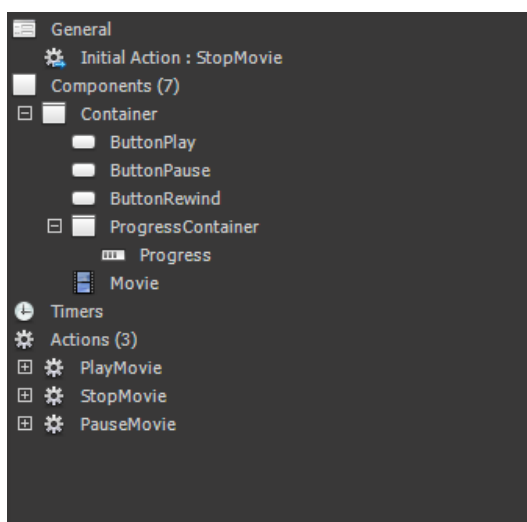
Games	-	MoviePlayer
AIModels	-	MoviePlayer_Main

HUD	-	MoviePlayer
Movies	-	madness
Scripts	-	MoviePlayer_Main_Handler_onEnterFrame MoviePlayer_Main_Handler_onInit

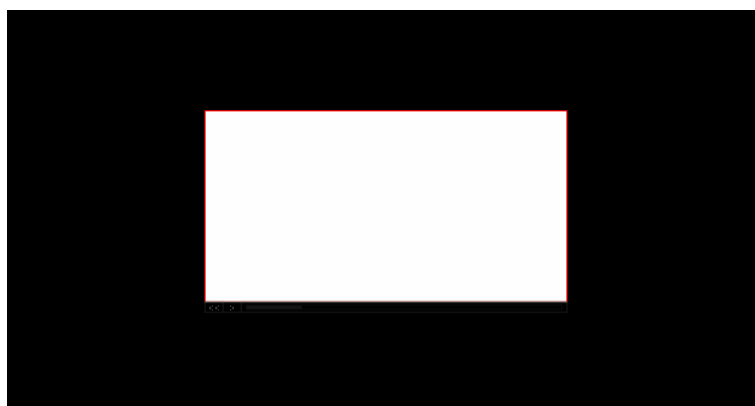
Again, the ones highlighted in Red will be the files that I'll be dissecting here. Since this Chapter is about HUDs, we'll start with the HUD:

So, open up the HUD Editor, and we'll get started.

If you click on the "HUD" menu, and then "Open", you can select the "MoviePlayer" HUD, and you should see this:

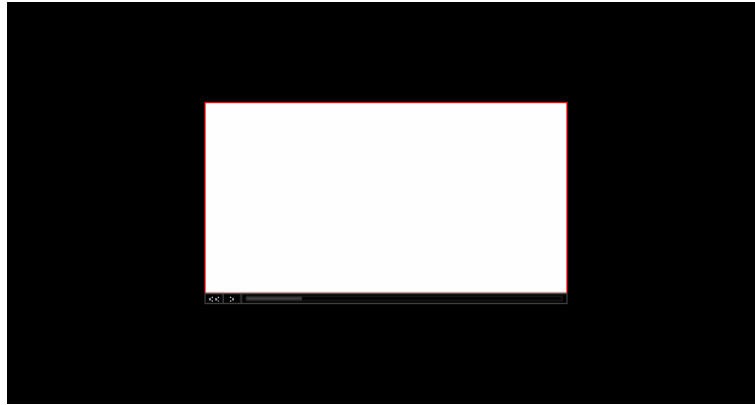


As you can see, this HUD contains 7 Components, and 3 Actions. To work with the WYSIWYG editor, open the Scene Viewer. All you'll see at the moment is a blank screen. However, if you click on one of the Components in the HUD Editor tree (such as "Movie"), you'll see something similar to the following (depending on which Component you clicked on of course!):

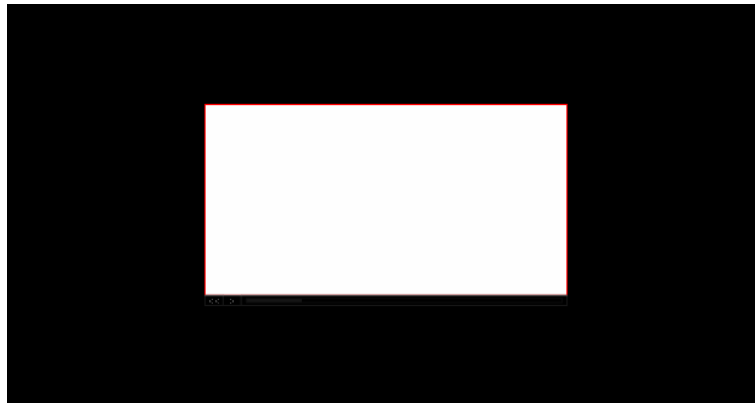


Notice how the rectangle has a red border. This is to show the currently selected Component.

If you now click on several of the other Components, you'll eventually see something like this:



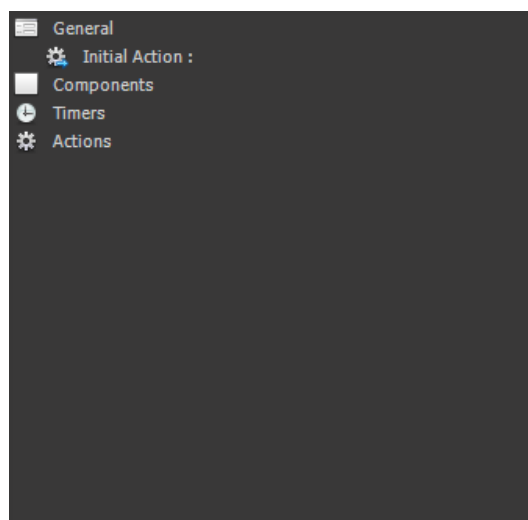
The above screenshot is showing what the final movie player will look like. By playing with the options in the “HUD” menu, under the “Preview” options, you’ll be able to see for yourself the way ShiVa handles both “Border”, and “Isolate” mode. The above screenshot looks like this in “Isolate” mode:



Not too much difference, but the buttons, and progress bar are now “greyed” out.

So, how was this HUD created?

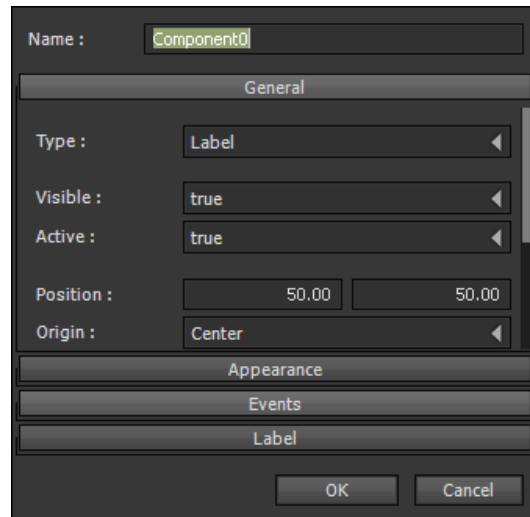
Firstly, you will need to create a new HUD by closing the demo one, and clicking on “HUD”, and then “Create”. Give your HUD a name, and you should see the following in the HUD Editor:



HEADS UP!

Firstly, we'll create the graphics of the HUD, and then the Actions.

To begin, right-click on the “Components” part of the tree, and select the “Add Component” option (or click on the “Component” menu, and select “Add”). This will generate a new Component, as shown below:



The first Component we need to add is a Container, with the following settings:

General

Name:	Container	
Type:	Container	
Visible:	true	
Active:	true	
Position:	50	50
Origin:	Center	
Size:	90	50
Viewport aspect ratio independent	checked	
ZOrder:	127	

Appearance

Shape:	Rectangle			
Corners radius:	5 (cannot be changed for a Rectangle)			
Opacity:	255			
Back Texture:	“blank”			
BackColor:	0	0	0	255
ForeColor:	127	127	127	255
BorderColor:	0	0	0	255

Events

Leave all “blank”.

These settings create a rectangular Container, of size 90 by 50, in the centre of the screen.

Now, we need to add some children into the Container. So, right-click on the Container in the tree, and select the “Add Child” option. You will be presented with the same default Component as shown above.

The first child that we will be adding is the “ButtonPlay” Component.

This has the following settings:

General

Name:	ButtonPlay
Type:	Button
Visible:	true
Active:	true
Position:	7.5 2.5
Origin:	Center
Size:	5 5
Viewport aspect ratio independent	unchecked
ZOrder:	127

Appearance

Shape:	Rectangle
Corners radius:	5 (cannot be changed for a Rectangle)
Opacity:	255
Back Texture:	“blank”
BackColor:	0 0 0 255
ForeColor:	127 127 127 255
BorderColor:	32 32 32 255

Events

Leave all “blank”.

Button

Text:	>
Text Height:	100
Letter Spacing:	0
Text Alignment:	Center Center
Text Case:	Variable
Text Font:	“blank”
Antialiased Text:	unchecked
Click Action:	“blank”
Clicked Action:	“blank” (for now, this will eventually contain the “PlayMovie” Action)

The above will create a small button towards the bottom left of the Container, with “>” displayed on it.

Now, we need to create two more buttons (“ButtonPause” and “ButtonRewind”).

“ButtonPause” is identical to “ButtonPlay” (since it replaces “ButtonPlay” whilst the movie is playing), except for the following:

General

Name: ButtonPause
Visible: false

Button

Text: II
Clicked Action: “blank” (for now, this will eventually contain the “PausedMovie” Action)

“ButtonRewind” is also mostly the same as “ButtonPlay”, with the following differences:

General

Name: ButtonRewind
Position: 2.5 2.5

Button

Text: <<
Clicked Action: “blank” (for now, this will eventually contain the “StopMovie” Action)

Now that you have three buttons, the next step is to add the “Progress Bar” Component. This is done in two stages:

1) Add a new Container

This is done in the same way as the main Container described above, but with the following settings:

General

Name: ProgressContainer
Type: Container
Visible: true
Active: true
Position: 55 2.5
Origin: Center
Size: 90 5
Viewport aspect ratio independent: unchecked
ZOrder: 127

Appearance

Shape: Rectangle
Corners radius: 5 (cannot be changed for a Rectangle)
Opacity: 255
Back Texture: “blank”
BackColor: 0 0 0 255
ForeColor: 127 127 127 255
BorderColor: 32 32 32 255

Events

Leave all “blank”.

The above will generate a long, thin Container to the right of the buttons at the bottom of the main Container (see screenshot below – the new Container is highlighted in red):



2) Add a Progress Bar

This is added in much the same way that the buttons were added to the main Container. Firstly, you need to right-click on the Container created above, and select the “Add Child” option. Then you need to input the following settings:

General

Name:	Progress		
Type:	Progress		
Visible:	true		
Active:	true		
Position:	50	50	
Origin:	Center		
Size:	97	40	
Viewport aspect ratio independent	unchecked		
ZOrder:	127		

Appearance

Shape:	Rectangle			
Corners radius:	5 (cannot be changed for a Rectangle)			
Opacity:	255			
Back Texture:	“blank”			
BackColor:	32	32	32	255
ForeColor:	127	127	127	255
BorderColor:	16	16	16	255

Events

Leave all “blank”.

Progress

Value: 45
Type: Left to Right

The above will generate a Progress Bar in the Container you have just created. Notice that the colouring is slightly different for the Progress Bar, this is so that it stands out against the Container.

As an aside, what I found with designing HUDs, or any other layout for that matter, is that half of the battle is getting everything to look right (colour, size etc.). So, I would suggest that you do a quick drawing of the HUD you want to achieve (graph paper makes it much easier when working out sizes!), and maybe even use some coloured pencils to colour some of it in. This way, you don't spend hours in front of the screen! As I've told you before, I'm a pretty atrocious artist, but I am not too bad at “Technical Drawing” (so long as it doesn't involve curves or circles!), so whenever possible, I grab some of my son's graph paper (and coloured pencils!), and sketch out my design. This method has saved me hours of “screen squinting”!

The final piece to this particular jigsaw is adding a Component to show the movie itself. This is again pretty straightforward, and can be achieved by adding a new Component (to the MAIN Container), and entering the following settings:

General

Name:	Movie
Type:	Movie
Visible:	true
Active:	true
Position:	50 52.5
Origin:	Center
Size:	100 95
Viewport aspect ratio independent	unchecked
ZOrder:	127

Appearance

Shape:	Rectangle
Corners radius:	5 (cannot be changed for a Rectangle)
Opacity:	255
Back Texture:	“blank”
BackColor:	0 0 0 255
ForeColor:	127 127 127 255
BorderColor:	32 32 32 255

Events

Leave all “blank”.

Movie

Movie Clip:	“blank” (this is populated from Script, so doesn't need to be referenced here)
-------------	--

Well, that's it for the actual visible part of the HUD! Now, we need to turn our attention to the "Actions" that need to be set up to control how the HUD operates.

For this HUD there are 3 Actions:

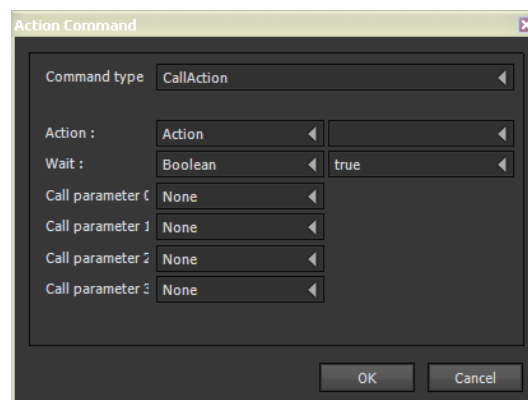
- 1) PlayMovie
- 2) PauseMovie
- 3) StopMovie

I'll run through each in turn:

PlayMovie

This Action is created as follows:

- 1) Click on the "Action" menu, and then on "Add Action" (or right-click on the "Actions" Heading, and select "Add Action"). This will prompt you for a name for the Action, and generate an empty Action in the Tree.
- 2) Click on the newly created Action, and then on the "Action" menu. You will now see that the "Action Command" option is available. If you select this option, you will see a new menu appear with the option to "Add Command". If you select this, you will be presented with the standard Command pop-up:



NOTE: you can also get to this pop-up by right-clicking on the newly created Action, and clicking on "Add Command".

- 3) Enter the settings for the Command.

The Commands, and settings for "PlayMovie" are as follows:

PlayMovie	Component	-	Movie	
SetVisible	Component	-	ButtonPause	
	Visible	-	Boolean	true
SetVisible	Component	-	ButtonPlay	
	Visible	-	Boolean	false

Basically, this Action starts the movie playing, and then makes the "Pause" Button visible, and the "Play" Button, invisible.

The Commands, and settings for “StopMovie” are as follows:

StopMovie	Component	-	Movie		
SetVisible	Component	-	ButtonPause		
	Visible	-	Boolean	-	false
SetVisible	Component	-	ButtonPlay		
	Visible	-	Boolean	-	true

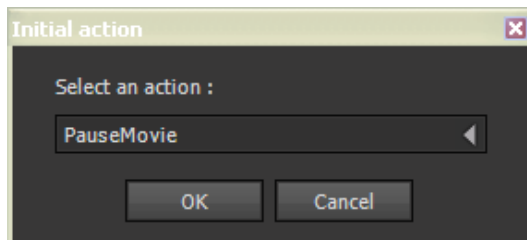
Basically, this Action stops the movie playing, and then makes the “Pause” Button invisible, and the “Play” Button, visible.

The Commands, and settings for “PauseMovie” are as follows:

PauseMovie	Component	-	Movie		
SetVisible	Component	-	ButtonPause		
	Visible	-	Boolean	-	false
SetVisible	Component	-	ButtonPlay		
	Visible	-	Boolean	-	true

Basically, this Action is identical to “StopMovie”, except that it pauses the movie, and then makes the “Pause” Button invisible, and the “Play” Button, visible.

The only thing left to do now is to set the “Initial Action” under the “General” heading in the tree. To do this, right-click on “Initial Action”, and select the “Change Initial Action” option. This will open the following dialog, allowing the selection of one of the Actions created above:



So, all you need to do is select the “StopMovie” Action, and we’re almost done.

Before we can move on to the AIModel, however, we need to populate the “Clicked Action” events that we left “blank” earlier. In the ButtonPlay Component, set the “Clicked Action” to “PlayMovie”. In the ButtonPause Component, set the “Clicked Action” to “PauseMovie”, and in the ButtonRewind Component, set the “Clicked Action” to “StopMovie”.

And that, my friends, is the end of the HUD design for the Movie Player. Next, we’ll quickly run through the Scripts for this Game (there’s only two of them, and they’re not very big).

MoviePlayer_Ball

This is the only AIModel for this demo, and consists of the following:

Handlers

onEnterFrame ()
onInit ()

onEnterFrame ()

```
local hMovie = hud.getComponent ( this.getUser (), "MoviePlayer00.Movie" )  
local hProgress = hud.getComponent ( this.getUser (), "MoviePlayer00.Progress" )
```

OK, all we are doing here is getting Handles to the HUD Components “Movie” and “Progress”. Hang on though, why is the HUD called “MoviePlayer00”? This is because in the “onInit” Handler, we will be creating an instance of the HUD that we have just created specifically for the current User.

```
if ( hMovie ~= nil and hProgress ~= nil )  
then  
    hud.setProgressValue ( hProgress, hud.getMoviePlaybackProgress ( hMovie ) )  
end
```

Next, we check that our Handles have been successfully created and, if so, we set the current playback value of “hMovie”, and update the Progress Component (“hProgress”) accordingly.

That’s it for this Handler.

onInit ()

```
hud.newTemplateInstance ( this.getUser (), "MoviePlayer", "MoviePlayer00" )
```

This line does all the hard work! It creates a new instance of the “MoviePlayer” HUD for the current User, and names it “MoviePlayer00”.

```
local hMovie = hud.getComponent ( this.getUser (), "MoviePlayer00.Movie" )
```

```
if ( hMovie ~= nil )  
then  
    hud.setMovieClip ( "madness" )  
end
```

This last bit creates a Handle to the “Movie” Component of our HUD, checks if it is created and, if so, sets the Movie to be played (“madness”). This could have been set in the Movie Component itself, but, doing it this way shows how easy it is to set it via Script! Note that the selected Movie MUST be available to your project, or you will get an AI error.

Well, that brings to an end my coverage of the HUD Editor. In the next Chapter, I’ll introduce the Attributes Editor, and what it can do for you.