Storyboard Documentation

Documentation for version 1.0 (2009)

Updated 29 June 2009

Contents

Overview

A storyboard in M.U.G.E.N is composed of a combination of animation, text, music and/or sound, in the form of a cutscene. Storyboards can be used to make cutscenes such as the game introduction, character endings, credits and more.

Terminology

cutscene
: What you actually see (and hear).

event
: What triggers a cutscene to be played back.

storyboard
: The .def file that defines what you see during a cutscene.

For example, during the *ending event*, the *ending cutscene* will be played back. The *ending cutscene* uses the ending.def *storyboard*.

Getting Started

Cutscenes Events

There are several global events when cutscenes may be played. The storyboards for these cutscenes are defined in system.def, and applies to all characters within the game. The cutscenes in this set are called the "system cutscenes".

Game Logo
: Played back once when you start M.U.G.E.N.

Game Intro
: Played back after the Game Logo.

Default Ending
: Played back after you beat the game with a character that does not have a user-defined ending.

Credits
: Played back after the ending cutscene.

Game Over
: Played back if you choose "No" at the continue screen.

Another set of cutscenes is specific to each character, and is defined in each character's .def file. These are known as the "character cutscenes". For example, Kung Fu Man's character cutscenes are defined in chars/kfm/kfm.def.

Character Intro
: Played back once after character selection in arcade mode.

Character Ending
: Played back after beating the game in arcade mode. The default ending, where a generic congratulatory message is shown, will not be played back if this cutscene is defined.

Trying It Out

At the command line, type:

```
mugen -r kfm
```

This will start the "KFM" motif (for more information on motifs, please read readme.txt). Right away, you will see the game logo, followed by the game intro. After the game intro ends, you will be at the title screen. If you start Arcade mode and choose Kung Fu Man, his character intro will be played, and then the fight begins. If you lose the fight and choose "No" at the continue screen, the game over cutscene is played and you are returned to the title screen. If you win, Kung Fu Man's character ending will be played, followed by the ending credits.

These are what each of the storyboard files are named in the KFM motif.

Game Logo	data/kfm/logo.def
Game Intro	data/kfm/intro.def
Default Ending	(none)
Credits	data/kfm/credits.def
Game Over	data/kfm/gameover.def
Character Intro	chars/kfm/intro.def
Character Ending	chars/kfm/ending.def

The storyboard filenames for the system cutscenes are specified in data/kfm/system.def.

Viewing Storyboards

To play back a storyboard file, use M.U.G.E.N's -storyboard command-line option. The syntax is:

```
mugen -storyboard <storyboard filename>
```

For example, to play back data/kfm/intro.def, you would type:

```
mugen -storyboard data/kfm/intro.def
```

The storyboard search directory defaults to data/, so you can omit the "data/" part if you like. The following will also play back data/kfm/intro.def:

mugen -storyboard kfm/intro.def

This feature can be useful for quickly testing your storyboards as you build them.

Storyboard Basics

Here is an example of a simple storyboard that shows just one picture for 5 seconds. If you are not already familiar with sprites and animations, please consult the spr and air docs before continuing.

```
;-----------------------------------------------------------
[Info]
localcoord = 640,480   ;This storyboard is made for a 640x480 size screen

[SceneDef]
spr = sprite.sff       ;Filename of sprite file to use

[Scene 0]
; Layers
layerall.pos = 319,239 ;Default position for all layers
layer0.anim =  0       ;Anim action number to show
; Total time
end.time = 300         ;300 ticks = 5 seconds

;Animation to use
[Begin Action 0]
0,0, 0,0, -1
;-----------------------------------------------------------
```

A storyboard must always contain a [SceneDef] group. Within this group you must specify the name of the sprite file to use. This sprite file must exist in the same directory as the storyboard file. The sprite file should contain all the sprites you need for your storyboard.

A storyboard is broken up into scenes. How you choose to divide up your storyboard is up to you. In the example above, there is only one scene, the [Scene 0] group. All scenes must begin with a group that looks like [Scene ?], where the question mark (?) can be any identifier string. Typically, scenes are labeled in numerical order starting from 0.

Scenes are composed of up to 10 animation objects. Each of these objects exists on its own layer, and the layers are overlaid on top of each other. Layers are numbered from 0 to 9, with 0 being at the bottom, and 9 at the top. For instance, an animation on layer 0 is underneath another on layer 1, and the animation on layer 1 is under that of layer 2, and so on. Each animation object must make reference to an animation action using a "layerX.anim" parameter, where X is the layer number. Each action must be defined within the same storyboard file.

The simple example above has only one layer. Its drawing position is determined by the value of the "layerall.pos" parameter, and the action number of the animation is the value of the "layer0.anim" parameter. In this case, the action number is 0. Action 0 is defined right after the [Scene 0] group.

One important parameter is "end.time". This is the time to end the scene, measured in ticks. One tick is equivalent to 1/60th of a second. When one scene ends, it goes on to show the next. If there are no more scenes, the storyboard playback ends. In the example, "end.time" is 300 ticks, which is the same as 5 seconds. Since there is only one scene, the storyboard ends at the same time.

The next example will show a slideshow of pictures, along with text overlaid on top. Again, this only has one scene.

```
;-----------------------------------------------------------
[Info]
localcoord = 640,480   ;This storyboard is made for a 640x480 size screen

[SceneDef]
spr = sprite.sff       ;Filename of sprite file to use
font0 = myfont.def     ;Filename of a font file to use for the text
```

```
[Scene 0]
; Layers
layerall.pos = 319,239 ;Default position for all layers
layer0.anim =  0       ;Anim action number for slideshow pictures
layer1.text = "Hello"  ;Text to show
layer1.font = 0        ;Use font0
; Total time
end.time = 600         ;600 ticks = 10 seconds

;Animation to use for pictures
[Begin Action 0]
0,0, 0,0, 120
0,1, 0,0, 120
0,2, 0,0, 120
0,3, 0,0, 120
0,4, 0,0, 120
;-------------------------------------------------------------
```

Note that there are 2 layers now. Layer 0 is used for the slideshow of pictures, and layer 1 is used for the overlay title. If you imagine the slideshow to be pictures of your vacation trip, the title might be an image with text that reads, "My summer vacation".

The new "fadein.time" and "fadeout.time" parameters specify fade-in and fade-out times respectively, measured in ticks.

This third example shows multiple scenes, with screen fades between each scene. Additionally, it plays music and a sound effect.

```
;-------------------------------------------------------------
[Info]
localcoord = 640,480

[SceneDef]
spr = sprite.sff
snd = sound.snd

[Scene 0]
; Fade parameters
fadein.time = 30       ;30 ticks (0.5 seconds) for fade-in
fadeout.time = 30      ;30 ticks for fade-out
; Layers
layerall.pos = 319,239
layer0.anim =  0
; Music
bgm = mysong.s3m       ;Filename for music
; Total time
end.time = 120

;Animation to use for scene 0
[Begin Action 0]
0,0, 0,0, -1

[Scene 1]
; Fade parameters
fadein.time = 30       ;30 ticks (0.5 seconds) for fade-in
fadeout.time = 30      ;30 ticks for fade-out
; Layers
layerall.pos = 160,120
layer0.anim =  10
; Total time
end.time = 120

;Animation to use for scene 1
[Begin Action 10]
10,0, 0,0, -1

[Scene 2]
; Fade parameters
fadein.time = 30       ;30 ticks (0.5 seconds) for fade-in
fadeout.time = 30      ;30 ticks for fade-out
; Layers
layerall.pos = 160,120
```

```
layer0.anim =  20
; Sounds
sound0.value = 0,1      ;Plays sound 0,1
sound0.starttime = 60   ;Starts playing the sound 60 ticks after the start of the scene
; Total time
end.time = 120

;Animation to use for scene 2
[Begin Action 20]
20,0, 0,0, -1
;-----------------------------------------------------------
```

There is a "bgm" parameter in the first scene. This plays back a music file "mysong.s3m" at the start of the scene, and that music will continue playing until the end of the storyboard. In this case, mysong.s3m should be placed in the same directory as the storyboard file.

For full details on the parameters for Scenes and SceneDefs, see the sections titled "SceneDef parameter reference" and "Scene parameter reference" below.

Testing Your Storyboard

This section covers some tips for testing your storyboard.

You may find pausing and framestepping useful when you play back your storyboard. The pause and framestep buttons are Pause and Scroll Lock on your keyboard respectively. These key are enabled only if M.U.G.E.N is running in debug mode. You are not advised to use the pause function if your storyboard has animations synchronized to the music, as pausing may disrupt synchronization.

When you write a long storyboard with many scenes, you may like to skip over completed scenes to view your newer ones. To do this, add a line to your SceneDef group:

```
[SceneDef]
spr = sprite.sff        ;Filename of sprite file to use
startscene = 0          ;<-- line added
```

Change the 0 to whatever scene number you would like to start your storyboard at. For example, if you are working on the 10th scene, and you would like to skip the first 9 scenes during testing, enter the number 10. Note that skipping a scene will cause the music in that scene not to be played back.

Parameter Reference

Info Parameter Reference

Optional parameters:

localcoord = *width*, *height* (int, int)
: This specifies the drawing canvas dimensions in pixels. If the canvas dimensions do not match the window resolution, it will be scaled to fit. Defaults to 320,240 if omitted.

SceneDef Parameter Reference

Required parameters:

spr = *filename* (string)
: This is the filename of the sprite (.sff) file to use in the storyboard.

Optional parameters:

snd = *filename* (string)
This is the name of the sound (.snd) file to use in the storyboard.
fontX = *filename* (string)
This set of parameters loads up to 10 fonts for use in the storyboard. You can specify multiple font files, using *X* as the font number. *X* is valid for values from 0 to 9. Font files are first searched for within the same directory as the storyboard def file. If the font named *filename* cannot be loaded from that directory, the engine will attempt to load it from the font/ directory instead.
startscene = *scene_number* (int)
This parameter is used mainly for testing purposes. If specified, the first *scene_number* scenes will be skipped. Valid values are from 0 to the total number of scenes minus 1. If omitted, defaults to 0.

Scene Parameter Reference

Please note that time is measured in ticks, where 60 ticks is equal to one second. Take note that some optional parameters have default values that depend on the scene number. For example, if the "clearcolor" parameter is omitted, it will have a different default value on the first scene, compared to if it was omitted on following scenes.

Required parameters:

end.time = *t* (int)
t is the time to end the scene, measured in ticks relative to the starting time of the scene. If there is another scene after the current one, it will start when the current scene ends. Otherwise, the whole storyboard will end.

Basic optional parameters:

fadein.time = *duration* (int)
duration is the length of time (measured in ticks) to fade the screen in, at the start of the scene. Note that fadein.time does not affect the ending time of the scene.
fadein.col = *r*, *g*, *b* (int, int, int)
This parameter is paired with fadein.time. *r*, *g*, *b* represents the R,G and B values of the starting fade color. Valid values for duration are from 0 (no fade) to the value of end.time. Valid values for the *r*, *g*, *b* triplet are 0,0,0 (fade from black) and 255,255,255 (fade from white) only. Other fade colors are currently not supported. If omitted, duration defaults to 0 (no fade) and r,g,b to 0,0,0 (fade from black).
fadeout.time = duration (int)
Similar to the fadein parameters, except this fades the scene out.
fadeout.col = *r*, *g*, *b* (int, int, int)
Similar to the fadein parameters, except this fades the scene out.
clearcolor = *r*, *g*, *b* (int, int, int)
This is the color to clear the screen with before drawing each tick. If r is set to -1, the screen will not be cleared. If omitted on the first scene, r defaults to -1. If omitted on successive scenes, *r*, *g*, *b* defaults the previous scene's values. For instance, if you set clearcolor to 0,0,0 in the first scene, you do not need to specify the parameter for successive scenes if you would like them all to use the same color values.
layerall.pos = *x*, *y* (int, int)
This is the default position to draw all layers to. If omitted on the first scene, defaults to 0,0. If omitted on successive scenes, the value defaults to that of the previous scene's. Note that this parameter does not affect background objects.
layerX.anim = *actionno* (int)

The layerX.* parameters control animations and texts to be drawn to the screen. layerX.anim specifies an animation. The value of *X* controls the drawing priority of the layer. For example, an layer with *X* = 0 will be drawn below another with *X* = 1. Valid values for *X* are from 0 to 99, for a total of 100 layers.

actionno specifies that an the animation with action number *actionno* will be shown. If this parameter is omitted, no animation will be drawn.

layerX.text = *text* (string)

This parameter specifies the text that will be drawn to the screen. If you need to have a line break in *text*, enclose it in quotes and use \n to specify the line break escape sequence, e.g.
`layerX.text = "This is one one line.\nThis is on a second line."`

A layerX.text parameter must be paired with a layerX.font parameter.

You cannot use text and anim on the same layer.

layerX.font = *font_no*, *bank*, *align*, *r*, *g*, *b* (int, int, int, int, int, int)

This parameter specifies the font to use for rendering the text of the layer.

font_no is the number of the font from the [SceneDef].

bank is the color bank of the font (bitmap fonts only).

align is the text alignment: 1 for left-aligned text, 0 for center, and -1 for right-aligned text.

r, *g*, *b* specifies the color of the font. Values are from 0 to 255.

layerX.textdelay = *delay* (int)
This parameter controls how quickly each letter of text is rendered to the screen. Text is rendered one letter at a time to the screen, waiting *delay* ticks between each letter. *delay* defaults to 0 (all text is rendered immediately).
layerX.offset = *offx*, *offy* (int, int)
offx, *offy* is the x,y position offset to draw the layer to. The values of this parameter are added to those of the "layerall.pos" parameter to determine the drawing position. This defaults to 0,0 if omitted.
layerX.starttime = *t* (int, int)
t is the time to start drawing the layer. If omitted, it defaults to 0.
layerX.endtime = *t* (int, int)
t is the time to stop drawing the layer. If omitted, it defaults to the end of the scene.
soundX.value = *group_no*, *sound_no* (int, int)
This parameter specifies a sound to play back during the scene. *group_no* and *sound_no* correspond to the identifying pair that you assigned each sound in the player's snd file. Valid values for *X* are from 0 to 99 (up to 100 sounds per scene).
soundX.starttime = *t* (int)
Controls when the sound is played. *t* is the time to start playing the sound. *t* defaults to 0.
soundX.volumescale = *volume_scale* (double)
volume_scale is 100 for 100% volume, 50 for 50% volume, and so on. Defaults to 100. Currently, there is no support to make a sound louder.
soundX.pan = *p* (int)
This is the positional offset of the sound. 0 plays the sound in the center. A negative value plays it towards the left. A positive value plays it towards the right. Defaults to 0. Valid values for *p* are -127 to 127.
bgm = *filename* (string)
If specified, the music file named *filename* will begin to play at the start of the current scene. If omitted on the first scene, no BGM will play. If omitted on successive scenes, the BGM from the previous scene will continue to play. If *filename* is an empty string, the BGM currently being played will be stopped. The BGM file should be placed in the same directory as the storyboard file.
bgm.loop = *loop* (boolean)
Set *loop* to 1 to make the background music loop, 0 to prevent looping. The default value is 0.

Advanced optional parameters:

window = *x1*, *y1*, *x2*, *y2* (int, int, int, int)
This defines the drawing window of the storyboard. *x1*, *y1* are the x,y coordinates of the upper left of the window, while *x2*, *y2* represents the lower right. Anything drawn outside this window will not be seen. Note that this parameter does not affect the drawing window of background-type objects. If omitted on the first scene, the values will default to the full size of

the screen. If omitted on successive scenes, the values will default to the those of the previous scene's.

bg.name = *bgname* (string)

If this parameter is specified, you can make use of a background object. *bgname* is a string prepended to your background definition groups. For example, if *bgname* is "Scene04bg", then your background definition group must be named "Scene04bgDef", and the following background elements must begin with "Scene04bg". data/kfm/intro.def and data/kfm/credits.def are examples that use this parameter. Note that background objects are not affected by the "window" parameter. All elements of a background objects are drawn underneath all "layerX" animation objects, with the exception of elements with the "layer" parameter set at 1.