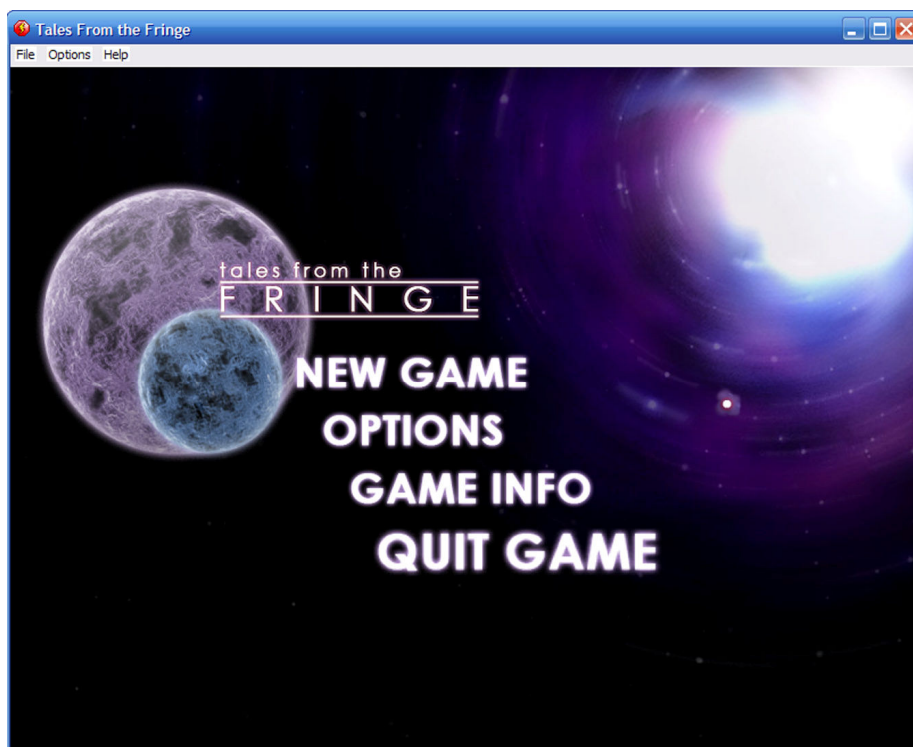


HOW TO CREATE  
A GREAT LOOKING

# MENU

...AND LIVE TO TELL THE TALE.

## PART TWO MAIN MENU & OPTIONS



You may not use this tutorial for any other purpose than learning, working or having fun... In other words: You can use this PDF tutorial for anything you'd like, as long as it doesn't involve both a hammer and a squirrel.

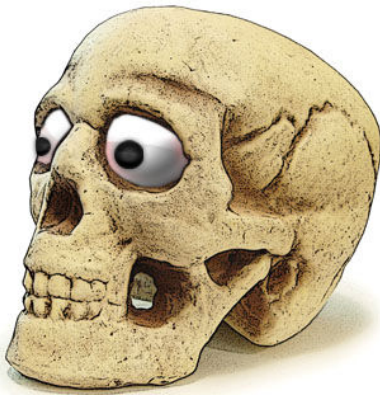
Koobare  
marchewkowy@gmail.com

**Welcome** to part two of “How to create a great looking menu and live to tell the tale”, another one of Koobare’s little tutorials, teaching you – as always – how to effectively and efficiently use the best multimedia authoring tool ever – [Multimedia Fusion 2](#) by Clickteam!

This tutorial is meant for people who have already finished “How to create a great looking menu and live to tell the tale – Part One”, and have their title screens for “**Tales from the Fringe**” up and ready. If you haven’t done that yet – just turn around, get back to Clickteam’s website, download part one of this here tutorial and we’ll see each other once you’re done with the first chapter. You wouldn’t start reading a book from the middle section, would you?

Previously on “*How to create a great looking menu and live to tell the tale*”...

**NOT ALL OF US  
MADE IT...**



**(BUT DON'T WORRY ABOUT  
THIS GUY - THIS IS SPENCER  
THE SKULL AND HE  
ALWAYS LOOKED LIKE THIS)**

**It was a long and harsh journey of epic proportions**, but we made it somehow. We were deep behind enemy lines, out of bullets, out of hope, with blisters in our toes and skies pouring on top of our heads... Yet, somehow, we prevailed and managed to get that title screen working.

After months and months of violent strife and inhuman sacrifices, we finally were able to raise our heads and gaze upon the **mighty zooming title, the pulsing “press Enter to continue” sign, the weirdly fascinating two planets orbiting in the background...** It was beautiful. Simple, but nonetheless mesmerizing.

We thought that this was it... That this was the end, nothing more to accomplish, no more blood and tears, no more turmoil and chaos... We stood there – in front of the might title screen – thinking that we have managed to achieve everything achievable to mortal men. **But we were wrong.** Oh, we undeniably were wrong, brave menu builder. That wasn’t the end. That wasn’t even the beginning of the end... **That was just the end of the beginning...**

Epic (and, yet, somehow creepy) speeches aside: you've previously completed the first part of this here tutorial, **now it's time for part deux**. We've got a nice-looking title screen ready (take a look at the list to the right for a reminder of what we've done so far), but there are still two menu screens that we'll have to create for our game – the main menu and options screens.

As you've most probably noticed during part one of this tutorial, the **two main keywords** behind our menu screens are "**space**" and "**rotation**". That's why we're going to have even more **orbiting** in our main menu – all the menu options will be revolving around a selected point. Additionally, we'll have more of that purple smoke, too – this time it will be spawned around the mouse cursor.



Anyways, prepare yourself, because we're just about to launch! Get into that spacesuit once again, cadet! "**Tales from the Fringe**" – here we come!

- ☐ If you have any problems with this tutorial, or notice that there are some mistakes present, please, contact me and I'll do my best to help you and replace all the errors with correct information.

Contact me at: [marchewkowy@gmail.com](mailto:marchewkowy@gmail.com)

- ☐ **Note:** I've been receiving some reports that not all e-mails get to me for some reason. Seems that some of them (quite a lot) end up in my spambox or are **blocked out by the server**. I dunno why this is happening, so if you're experiencing any difficulties with delivering me a message or haven't received a reply in quite some time, please, send me another e-mail at [marchewkowy@wp.pl](mailto:marchewkowy@wp.pl). I'll do my best to check both these e-mails regularly.



## Part I: Here we go again...

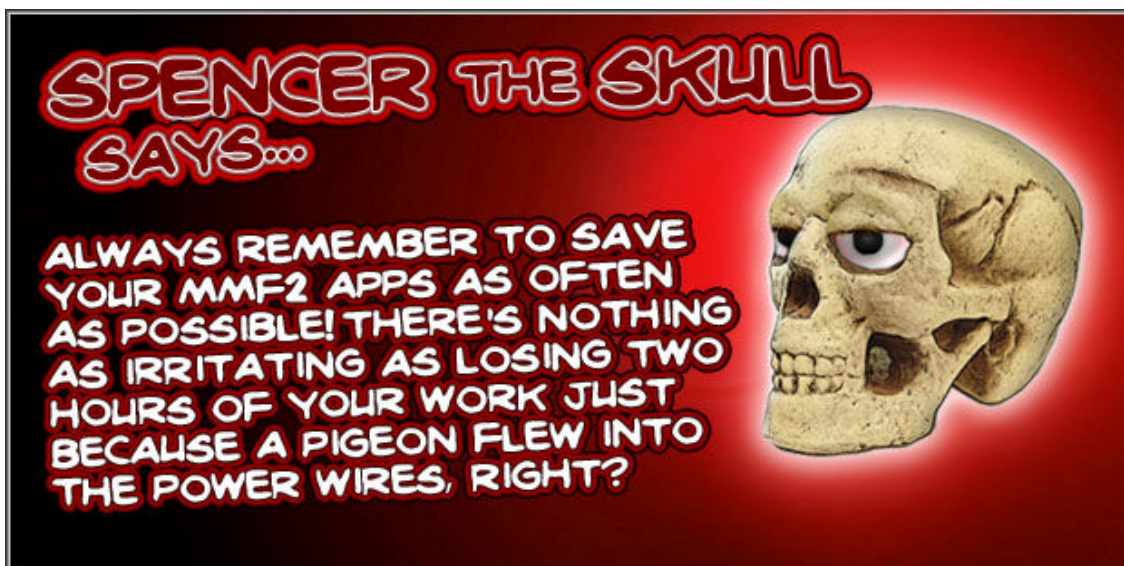
---

Open Multimedia Fusion 2 and open up the **menu-tutorial-part-two.mfa** file (should be in the same directory as this tutorial, since they were packed together into the same archive). If you wish, you can import the title screen we've created earlier, copy-and-pasting it to the front of this new app, so that all the screens can be found in one application. This isn't necessary, though, you can skip this if you wish.



**Please note:** most of the objects use alpha channels, a feature that is unavailable in Games Factory 2. TGF2 users should use basic library objects or create their own graphics instead (sorry, no easy way out of this if you want high quality objects) – I generally would advise you to upgrade to MMF2 as soon as possible, since you're missing out on some really good stuff, a lot of quite impressive features.

Now, once we've opened **menu-tutorial-part-two.mfa**, open up the frame entitled "Main Menu" and take a look around. As you can see, we have used some of these objects in part one of this tutorial, while others are completely new. We'll know them all in a jiffy.



## Part II: The objects.

---

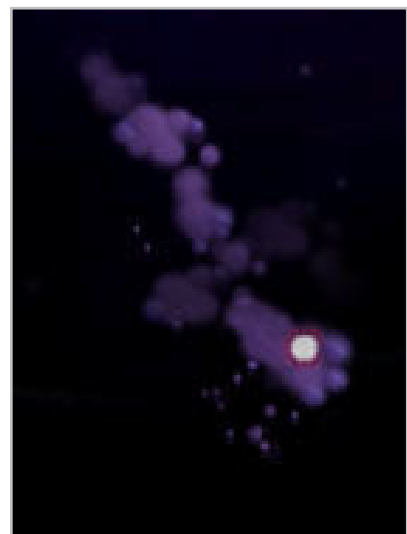
Time to learn a thing or two about the weird gizmos that are laying around in our frame. Below you can find an alphabetical list of all the objects – both the old ones, and the new ones:

Object's name:	So... What is it?
<b>game.logo</b>	<p><u>Just a rotating active object presenting the game's logo.</u></p> <p>MOVEMENT</p> <p>As with most objects in this frame, I've set this active up with a <b>Simple Ellipse</b> movement (Angular Velocity at 55, Radius X at 28, Radius Y at 108, Centre X at 190, Centre Y set at 150, Starting Angle and Offset Angle both set at 0), so it rotates frantically, joining the four menu options that are spinning beneath..</p>
<b>hidden.cursor</b>	<p><u>A small active always following the player's cursor.</u></p> <p>POSITION</p> <p>As you'll see later on in the Event Editor, this object will always stay in a very close proximity to the <b>player's cursor</b>. It will be used to spawn that purple smoke of ours.</p>
<b>menu.background</b>	<p><u>The same active object that was used in our title screen.</u></p> <p>MOVEMENT</p> <p><b>Simple Ellipse</b> movement: Angular Velocity at 180, Radius X and Radius Y both set at 15, Centre X and Centre Y both set at -15, Starting Angle and Offset Angle both set at 0.</p>
<b>menu.counter</b>	<p><u>A hidden counter helping us to control the menu with a keyboard.</u></p> <p>PROPERTIES</p> <p>Initial value set to 1, Minimum value set to 1, Maximum value set to 4. This counter will help us to control our menu (you'll see why later on).</p>
<b>menu.info</b>	<p><u>One of the menu selection buttons – "Game info".</u></p> <p>MOVEMENT</p> <p>I've set this active up with a <b>Simple Ellipse</b> movement: Angular Velocity at 55, Radius X at 85, Radius Y at 115, Centre X at 220, Centre Y set at 300, Starting Angle and Offset Angle both set to 0.</p>
<b>menu.newgame</b>	<p><u>One of the menu selection buttons – "New game".</u></p> <p>MOVEMENT</p> <p><b>Simple Ellipse</b>: Angular Velocity and Radius X at 55, Radius Y at 110, Centre X at 200, Centre Y set at 200, Starting and Offset Angle set to 0.</p>

<b>menu.options</b>	<p><u>One of the menu selection buttons – “Options”.</u></p> <p>MOVEMENT</p> <p>Another active set up with a <b>Simple Ellipse</b> movement: Angular Velocity at 55, Radius X at 70, Radius Y at 112, Centre X at 210, Centre Y set at 250, Starting Angle and Offset Angle both set to 0.</p>
<b>menu.quit</b>	<p><u>One of the menu selection buttons – “Quit game”.</u></p> <p>MOVEMENT</p> <p><b>Simple Ellipse:</b> Angular Velocity at 55, Radius X at 100, Radius Y at 119, Centre X at 230, Centre Y set at 350, Starting and Offset Angle set to 0.</p>
<b>planet.bigger and planet.smaller</b>	<p><u>You know why they're here...</u> Two planets spinning in the background, with the same setting as in the Title Screen part of this tutorial...</p>
<b>smokey.1 and smokey.2 and smokey.3</b>	<p><u>Three different actives used to create a nice purple smoke effect.</u></p> <p>WHY IT'S HERE?</p> <p>Once again – the purpose of these objects haven't changed since part one of this tutorial. The only difference is that they are now created around the “hidden.cursor” object, which makes them trail the player's cursor.</p>
<b>pointing.dot</b>	<p><u>A visible cursor that the player operates with.</u></p> <p>WHY IT'S HERE?</p> <p>It's just a visual cursor that our player moves with his mouse. It is always repositioned to the X and Y coordinates of the Windows mouse cursor.</p>

## Rockets flying, spaceships lifting off... More than just purple smoke.

The simple technique used in our tutorial to generate the **purple smoke** around the player's cursor can also be easily applied to different visual effects as well – for example as exhaustion smoke from a starship's massive engines, or as a trail of burnt gas left after an incoming rocket... **Possibilities are endless** – just replace the “smokey” thingies with bubbles and you can use it for a submarine or a torpedo. Replace it with stars and you have a cool effect for a fantasy game... Just use your imagination and play with the settings (change the transition, delay the destruction of the spawned objects, etc.) and you'll always have a nice effect to pull out of your sleeve.



### Part III: Programmer's delight.

---

Once we're done with examining all those objects – it's time for my favorite part of every tutorial! Save your project and go into the **Event Editor**. If you're new to my tutorials, let me introduce you to my little event-recording system. If you know it already – just skip this frame below and quickly move on to the coding part:

#### *Koobare's MMF-to-paper coding system*

IF (Condition): **[Object for the condition] > Condition group > Condition**  
THEN (Action): **[Object for the action] > Action group > Action**

Seems simple, right? Well, that's just because IT IS simple. All the conditions are marked in red, while actions are written in fancy blue.

Object names are always put in [square brackets]. The final condition/action is always in *Italic*. If we'll have a multi-condition event, then it'll be like this:

IF (Condition 1): **[Object for condition 1] > Condition group 1 > Condition 1**  
IF (Condition 2): **[Object for condition 2] > Condition group 2 > Condition 2**  
THEN (Action): **[Object for the action] > Action group > Action**

Whereas a multi-action event looks like this:

IF (Condition): **[Object for condition] > Condition group > Condition**  
THEN (Action 1): **[Object for the action 1] > Action group 1 > Action 1**  
THEN (Action 2): **[Object for the action 2] > Action group 2 > Action 2**

If you'll have to input anything by keyboard, it will be indicated by coloring the text green and using < angle brackets >, like this:

**< Set the Global Value A to 32 >**

Additional comments, instructions and info will be put in  
<< double angle brackets >>, using a different color:

<< Select any wave sound from the MMF2's sound library >>

From time to time I'll also use this style to throw in some extra tips and tricks about MMF2 and more advanced coding techniques.

All you have to do is to go step-by-step through all the listed events and keep one eye on your Event Editor, and the second one on this tutorial...

Let's make our main menu work...

1) Usually I start off with a "**Start of frame**" event, but this time... **We'll start with two.** One of these events will be activated when the music has been switched off in the options menu, the other one will be triggered if the player still wants the music to be played in the background...

IF: [Storyboard Controls] > *Start of frame*

IF: [Special Object] > *Compare to a global value*

<< Choose value *Music* >>

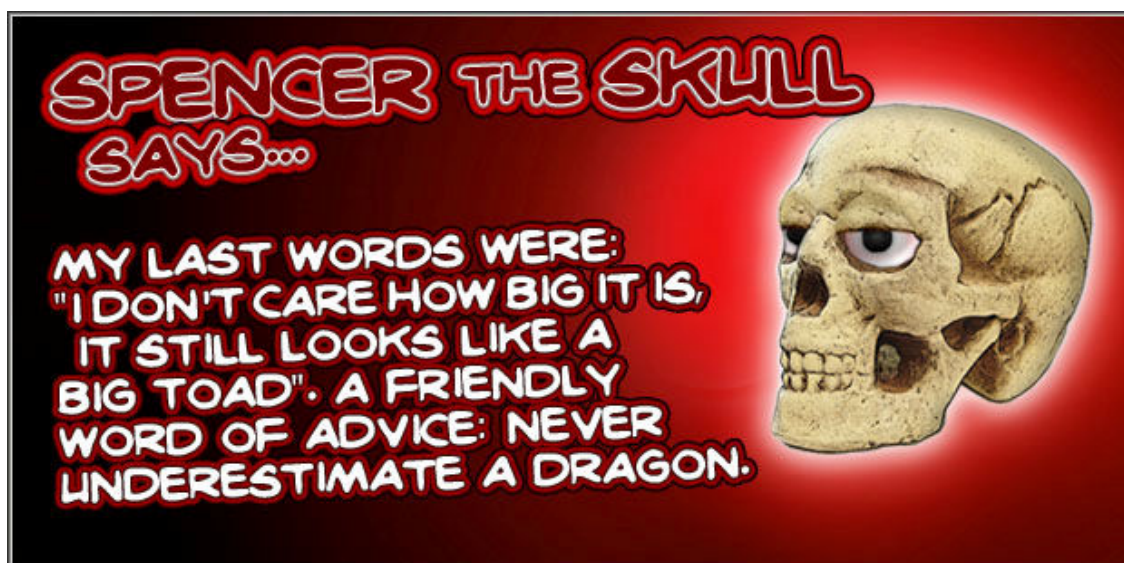
< compare if it is *Equal to 1* >

THEN: [hidden.cursor] > *Visibility* > *Make invisible*

THEN: [The Mouse Pointer and Keyboard] > *Hide Windows Mouse Pointer*

THEN: [Sound Object] > *Samples* > *Play and loop sample*

<< Choose the "London 2098.wav" sound, loop it 0 times, which means infinitely >>





As you can see, this event is triggered when – **at the very start of the frame – the “Music” Global Value equals 1**. Once activated, this event hides the Windows mouse pointer, makes the “**hidden.cursor**” object invisible and starts playing our menu music (the “**London 2098.wav**” from MMF2’s Bonus disc seems to fit pretty well with the overall “lost in deep space, exploring unknown worlds” atmosphere). Now, let’s see what happens when the “Music” Global Value equals 0 (meaning that music has been turned off in the options screen)... **Let’s create our second event:**

**IF: [Storyboard Controls] > Start of frame**

**IF: [Special Object] > Compare to a global value**

**<< Choose value Music >>**

**< compare if it is Equal to 0 >**

**THEN: [hidden.cursor] > Visibility > Make invisible**

**THEN: [The Mouse Pointer and Keyboard] > Hide Windows Mouse Pointer**

Well, as you can see here, it basically triggers the same actions, but this time without the action playing music in the background.

**2) Time to create two Event Groups** that we’ll need in a moment. Right-click on a blank event number, choose “Insert” and then “**A group of events**” from the context menu. Name the first group “**Menu control**” and make sure that its “**Active when frame starts**” option is on. Once that’s done, create the second group and name it “**Pointer smoke**” – once again, the “Active when frame starts” option should be on.



This is what we should have by now (don’t panic if it doesn’t look 100% the same)...

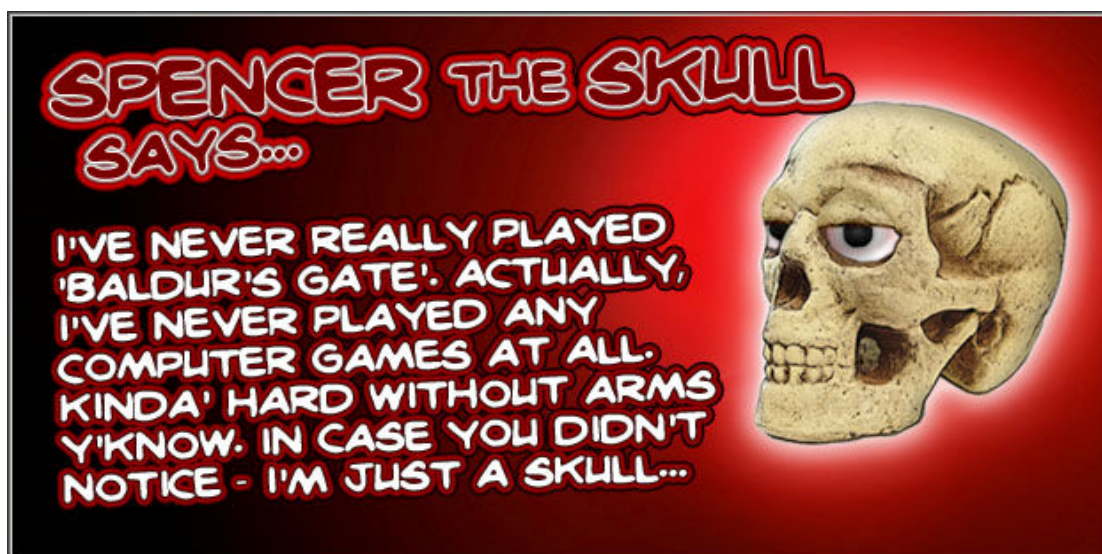
All the events All the objects																	
1	<ul style="list-style-type: none"> <li>Start of Frame</li> <li>Music = 1</li> </ul>		✓				✓										✓
2	<ul style="list-style-type: none"> <li>Start of Frame</li> <li>Music = 0</li> </ul>					✓											✓
3	Menu control																
4	Pointer smoke																

**3) In the options screen**, which we’ll take a look at a bit later on, our player will not only have the possibility to **turn the music off**, but he’ll be able to do the same for our “**menu animation**” (meaning that he will have the possibility to stop the rotation of our background, those two

planets and all the other thingies orbiting around our frame). This event helps to control this, stopping all animations and destroying all no longer needed objects when the “**Menu Animation**” Global Value is equal to 0:

```
IF: [Special Object] > Compare to a global value
<< Choose value Menu Animation >>
< compare if it is Equal to 0 >
IF: [Special Object] > Limit conditions > Run this event once
THEN: [planet.bigger] > Movement > Stop
THEN: [planet.smaller] > Movement > Stop
THEN: [game.logo] > Movement > Stop
THEN: [menu.quit] > Movement > Stop
THEN: [menu.info] > Movement > Stop
THEN: [menu.options] > Movement > Stop
THEN: [menu.newgame] > Movement > Stop
THEN: [smokey.1] > Destroy
THEN: [smokey.2] > Destroy
THEN: [smokey.3] > Destroy
THEN: [hidden.cursor] > Destroy
THEN: [menu.background] > Movement > Stop
THEN: [menu.options] > Movement > Stop
THEN: [Special Object] > Group of events > Deactivate
<< Select the Pointer smoke group >>
```


Notice that the event above deactivates the “Pointer smoke” Event Group that we created a bit earlier – this group will be responsible for spawning all the “smokey” objects.



4) Now it's time to add another event – this time based on the “**Always**” condition. This little thingie is needed to reposition both the cursors (the visible one and the hidden one), to make sure that the visible cursor is always on top of the “purple smoke” and to destroy all the “smokey” objects with every loop of our event list:

**IF: [Special Object] > Always**  
**THEN: [pointing.dot] > Position > Set X Coordinate**  
 << Type in: *XMouse* >>  
**THEN: [pointing.dot] > Position > Set Y Coordinate**  
 << Type in: *YMouse* >>  
**THEN: [smokey.1] > Destroy**  
**THEN: [smokey.2] > Destroy**  
**THEN: [smokey.3] > Destroy**  
**THEN: [hidden.cursor] > Position > Set X Coordinate**  
 << Type in: *XMouse+Random(3)-Random(3)* >>  
**THEN: [hidden.cursor] > Position > Set Y Coordinate**  
 << Type in: *YMouse+Random(3)-Random(3)* >>  
**THEN: [pointing.dot] > Order > Bring to front**

5) Time for a bit of event-moving... **Drag & drop both the groups lower in the events list**, so that they are below all of the previously created events, like this:

All the events All the objects																					
1	<ul style="list-style-type: none"><li>Start of Frame</li><li>Music = 1</li></ul>		✓				✓										✓				
2	<ul style="list-style-type: none"><li>Start of Frame</li><li>Music = 0</li></ul>						✓										✓				
3	<ul style="list-style-type: none"><li>Menu Animation = 0</li><li>Run this event once</li></ul>	✓							✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓
4	<ul style="list-style-type: none"><li>Always</li></ul>																✓	✓	✓	✓	✓
5	Menu control																				
6	Pointer smoke																				
7	<ul style="list-style-type: none"><li>New condition</li></ul>																				

6) Time to create a new line inside the first group, “**Menu control**”. We’ll start off with a **comment**, a very useful feature of MMF2, that saved me hours of work a couple of times before. Believe me: comments are truly underused and unfairly ignored.

How come? Well, commenting your code can be extremely helpful, particularly when you’re developing something a bit bigger... And especially if your memory is as bad as mine. Some time ago I returned to a big project after two months of working on something else and... I was lost. I would have been literally forced to retake my steps and create large portions of code

from the very beginning (and, believe me, there were tons of code in that app, hundreds and hundreds of lines), if it wasn't for my own comments, messages that I left hanging in event groups, explaining in details where the heck did all of these equations come from.

Anyways, right click on the blank line number inside our first Event Group, select "Insert" and then "**A comment**". Input this text (but spare yourself the brackets):

*<< This group contains all the events needed to control the main menu >>*

Once that's done, we'll have something like this in our event list:

5	Menu control
6	This group contains all the events needed to control the main menu.

7) Aaaand here we go with more lines. Create these two events inside the first Event Group, they will play a specific sample when someone presses "Enter" or clicks with the Left Mouse Button (you can find this sample on your MMF2 Bonus disc):

**IF: [Keyboard & Mouse Object] > The Keyboard > Upon pressing a key**

*<< press ENTER on your keyboard >>*

**THEN: [Sound Object] > Samples > Play sample**

*<< choose the "PULSE03.wav" sound >>*

**IF: [Keyboard & Mouse Object] > The Mouse > User clicks**

*<< select: LEFT BUTTON, SINGLE CLICK >>*

**THEN: [Sound Object] > Samples > Play sample**

*<< choose the "PULSE03.wav" sound >>*

8) These four events are used to control our main menu with the use of a mouse:

**IF: [Keyboard & Mouse Object] > The Mouse > Check for mouse pointer over an object**

*<< choose the [menu.newgame] object >>*

**THEN: [menu.counter] > Set Counter**

*<< input: 1 >>*

**IF: [Keyboard & Mouse Object] > The Mouse > Check for mouse pointer over an object**

*<< choose the [menu.options] object >>*

**THEN: [menu.counter] > Set Counter**

*<< input: 2 >>*



IF: [Keyboard & Mouse Object] > The Mouse > *Check for mouse pointer over an object*

<< choose the [menu.info] object >>

THEN: [menu.counter] > *Set Counter*

<< input: 3 >>

IF: [Keyboard & Mouse Object] > The Mouse > *Check for mouse pointer over an object*

<< choose the [menu.quit] object >>

THEN: [menu.counter] > *Set Counter*

<< input: 4 >>

9) And now its time for two little events that will enable the player to control our menu with his keyboard. As you can see – our application lets the player choose how he wants to navigate the menu, which is a pretty good feature in any game:

IF: [Keyboard & Mouse Object] > The Keyboard > *Upon pressing a key*

<< press *UP* on your keyboard >>

THEN: [menu.counter] > *Subtract from Counter*

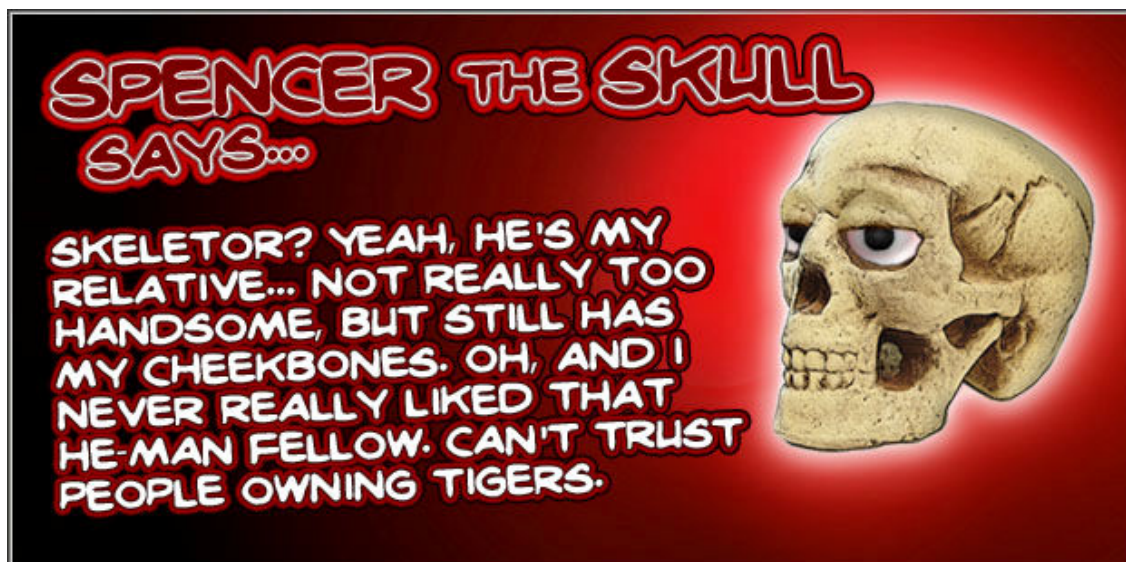
<< input: 1 >>

IF: [Keyboard & Mouse Object] > The Keyboard > *Upon pressing a key*

<< press *DOWN* on your keyboard >>

THEN: [menu.counter] > *Add to Counter*

<< input: 1 >>



10) Here are four events that make all of this possible... Also, note how we use the scaling feature to show which menu option has been currently chosen:

IF: [menu.counter] > Compare the counter to a value  
 < compare whether it is *Equal 1* >  
 IF: [Special Object] > Limit conditions > Only one action when event loops  
 THEN: [menu.newgame] > Scale / Angle > Set scale  
 < set scale to 1.2 >  
 < set quality to 1 >  
 THEN: [menu.options] > Scale / Angle > Set scale  
 < set scale to 1.0 >  
 < set quality to 1 >  
 THEN: [menu.info] > Scale / Angle > Set scale  
 < set scale to 1.0 >  
 < set quality to 1 >  
 THEN: [menu.quit] > Scale / Angle > Set scale  
 < set scale to 1.0 >  
 < set quality to 1 >  
 THEN: [Sound Object] > Samples > Play sample  
 << choose the "PULSE02.wav" sound >>



Aaaand here goes the second one... Notice that it's the "menu.options" that gets scaled bigger this time, whereas "menu.newgame" is scaled to 1.0 once again:

IF: [menu.counter] > Compare the counter to a value  
 < compare whether it is *Equal 2* >  
 IF: [Special Object] > Limit conditions > Only one action when event loops  
 THEN: [menu.newgame] > Scale / Angle > Set scale  
 < set scale to 1.0 >  
 < set quality to 1 >  
 THEN: [menu.options] > Scale / Angle > Set scale  
 < set scale to 1.2 >  
 < set quality to 1 >  
 THEN: [menu.info] > Scale / Angle > Set scale  
 < set scale to 1.0 >  
 < set quality to 1 >  
 THEN: [menu.quit] > Scale / Angle > Set scale  
 < set scale to 1.0 >  
 < set quality to 1 >  
 THEN: [Sound Object] > Samples > Play sample  
 << choose the "PULSE02.wav" sound >>

Here's the third one – once again, notice the subtle changes in the code:

```
IF: [menu.counter] > Compare the counter to a value
< compare whether it is Equal 3 >
IF: [Special Object] > Limit conditions > Only one action when event loops
THEN: [menu.newgame] > Scale / Angle > Set scale
< set scale to 1.0 >
< set quality to 1 >
THEN: [menu.options] > Scale / Angle > Set scale
< set scale to 1.0 >
< set quality to 1 >
THEN: [menu.info] > Scale / Angle > Set scale
< set scale to 1.2 >
< set quality to 1 >
THEN: [menu.quit] > Scale / Angle > Set scale
< set scale to 1.0 >
< set quality to 1 >
THEN: [Sound Object] > Samples > Play sample
<< choose the "PULSE02.wav" sound >>
```

And here's the fourth one, the last one in this batch of similar events... This time it's the "menu.quit" object that gets enlarged...

```
IF: [menu.counter] > Compare the counter to a value
< compare whether it is Equal 4 >
IF: [Special Object] > Limit conditions > Only one action when event loops
THEN: [menu.newgame] > Scale / Angle > Set scale
< set scale to 1.0 >
< set quality to 1 >
THEN: [menu.options] > Scale / Angle > Set scale
< set scale to 1.0 >
< set quality to 1 >
THEN: [menu.info] > Scale / Angle > Set scale
< set scale to 1.0 >
< set quality to 1 >
THEN: [menu.quit] > Scale / Angle > Set scale
< set scale to 1.2 >
< set quality to 1 >
THEN: [Sound Object] > Samples > Play sample
<< choose the "PULSE02.wav" sound >>
```

Got it? Great! Take a look at what I've got in my "Menu control" Event Group... As always: don't panic if something looks a tad different in your Events list, just try to examine where's the problem (if there – indeed – even is a problem):

5	Menu control																			
6	This group contains all the events needed to control the main menu.																			
7	• Upon pressing "Enter"	✓																		
8	• User clicks with left button	✓																		
9	• Mouse pointer is over																			✓
10	• Mouse pointer is over																			✓
11	• Mouse pointer is over																			✓
12	• Mouse pointer is over																			✓
13	• Upon pressing "Up Arrow"																			✓
14	• Upon pressing "Down Arrow"																			✓
15	•  = 1 • Only one action when event loops	✓										✓	✓	✓	✓					
16	•  = 2 • Only one action when event loops	✓										✓	✓	✓	✓					
17	•  = 3 • Only one action when event loops	✓										✓	✓	✓	✓					
18	•  = 4 • Only one action when event loops	✓										✓	✓	✓	✓					

11) There's still work to do, so let's continue... These two events control where does the game take us once a menu option is chosen. Notice that we're only having two selections active right now – the "Quit game" and "Options" ones. You'll have to create the "New game" and "Game info" reactions yourself or wait for the full "Tales from the Fringe" game tutorial... Anyways, all these comments aside, create these two events inside the "Menu control" group:

IF: [Keyboard & Mouse Object] > The Mouse > User clicks on an object

<< select: LEFT BUTTON, SINGLE CLICK >>

<< choose the [menu.options] object >>

<< OR operator (filtered) >>

IF: [Keyboard & Mouse Object] > The Keyboard > Upon pressing a key

<< press ENTER on your keyboard >>

IF: [menu.counter] > Compare the counter to a value

< compare whether it is Equal 2 >

THEN: [Storyboard Controls] > Next frame

Hope that you remember how to create the "OR" operator from the first part of this tutorial?

19	• User clicks with left button on																			
	OR																			
	• Upon pressing "Enter"																			
	•  = 2																			



And here's the second, very similar event:

IF: **[Keyboard & Mouse Object] > The Mouse > User clicks on an object**

<< select: **LEFT BUTTON, SINGLE CLICK** >>

<< choose the **[menu.quit]** object >>

<< OR operator (filtered) >>

IF: **[Keyboard & Mouse Object] > The Keyboard > Upon pressing a key**

<< press **ENTER** on your keyboard >>

IF: **[menu.counter] > Compare the counter to a value**

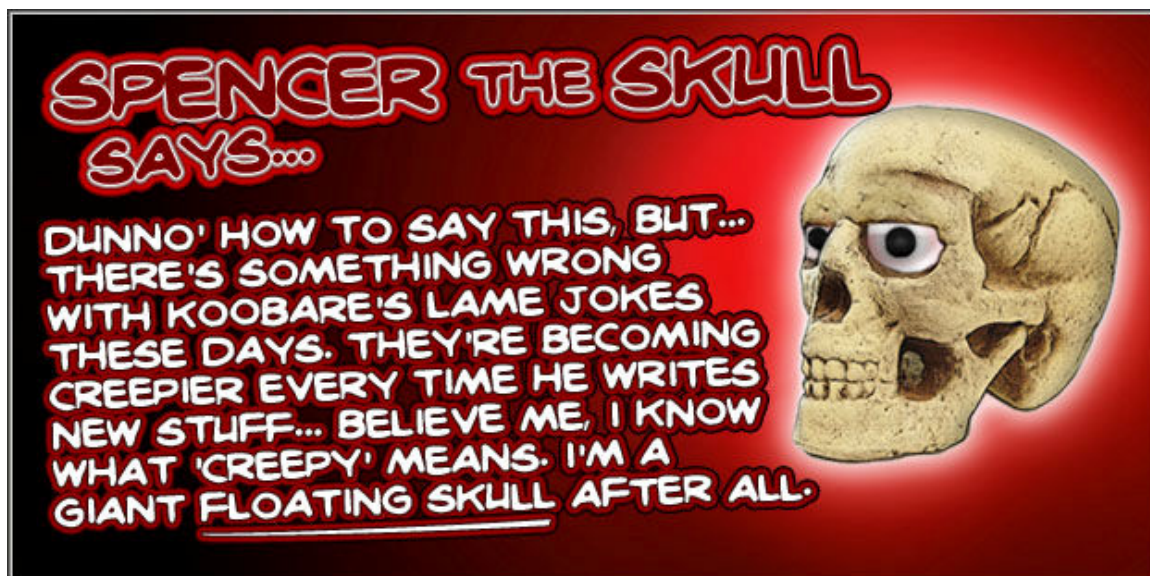
< compare whether it is **Equal 4** >

THEN: **[Storyboard Controls] > End the application**

Got it? Great! That means we're done with the events from the "Menu control" group!

12) It's now time to open up the second Event Group, named "**Pointer smoke**". After opening it up, create another comment inside, this time stating such a thing:

<< *Thanks to the events below we can see that cool smokey effect following the cursor.* >>



13) Once that's done, we have to create a whole sequence of similar events, spawning the "smokey" objects all around the "hidden.cursor" object. I'll show you how it's done, and then you'll have to fill all the blanks yourself, using the "Every X of a second" and "X chances out of Y" commands, up until the moment where those sporadic bursts of "smokeys" create a really nice-looking continuous effect. We need quite a lot of these events, creating "smokey.1", "smokey.2" and "smokey.3" at different time intervals to make it look random.

Here's the example... Notice that I always bring the "pointing.dot" object to front at the end:

```

IF: [The Timer Object] > Every
<< Set the timer to 0.07 of a second >>
THEN: [Create New Objects] > Create Object
<< Select the [smokey.1] object >>
<< Set the coordinates to x=0, y=1, relative to [hidden.cursor] object >>
THEN: [pointing.dot] > Order > Bring to front

```

Here's another example, this time using "smokey.2" instead of "smokey.1":

```

IF: [The Timer Object] > Every
<< Set the timer to 0.03 of a second >>
THEN: [Create New Objects] > Create Object
<< Select the [smokey.2] object >>
<< Set the coordinates to x=-2, y=0, relative to [hidden.cursor] object >>
THEN: [pointing.dot] > Order > Bring to front

```

Now, your job is to copy these examples and create all the three smokey objects at **different time intervals, in a distance of X and Y from the "hidden.cursor" object**, where X and Y are numbers ranging from 3 to -3 (actually, you can even use bigger or smaller numbers, such as -5 or 4, just make it sporadically). Here's how it looks in my Event Editor, you can copy it off if you want to, or you can do it for yourself, by experimentation:

22	Pointer smoke																			
23	Thanks to the events below we can see that cool smokey effect following the cursor.																			
24	• Every 00"-07					✓												✓		
25	• Every 00"-03					✓												✓		
26	• Every 00"-02					✓												✓		
27	• Every 00"-17					✓												✓		
28	• Every 00"-19					✓												✓		
29	• Every 00"-14					✓												✓		
30	• Every 00"-15					✓												✓		
31	• Every 00"-18					✓												✓		
32	• Every 00"-21					✓												✓		
33	• Every 00"-25					✓												✓		
34	• Every 00"-23					✓												✓		
35	• Every 00"-26					✓												✓		
36	• Every 00"-27					✓												✓		
37	• 1 chances out of 5 at random					✓												✓		
38	• 1 chances out of 6 at random					✓												✓		
39	• 1 chances out of 7 at random					✓												✓		
40	• 1 chances out of 8 at random					✓												✓		

Notice that I've used some "X chances out of Y at random" events at the bottom of the list...

This is a very interesting and powerful feature of MMF2 – you can create things truly randomly at chances specified by yourself, thanks to this little command:

**IF: [Special Object] > X chances out of Y at random**

Have fun with that, but be sure to save your application once you're done. Experiment with the time intervals, check how they influence the performance of your game and the overall look of our "purple smoke" effect. Once you're done, go straight for Section IV of this here tutorial...

#### **Part IV: Examining The Options Menu.**

---

Open up the next frame in the Event Editor, the one containing **Options Menu**. It's filled with events, from top till bottom, as I've decided to spare you the annoyance of creating the same events twice. Let's just take a look at what are the main differences between the events that can be found here, and those from the previous frame... It's up to you to check everything yourself, to reverse-engineer it up to the point in which you'll understand how it works. It isn't really that hard, I assure you, the code is very simple. Everything that could prove problematic is explained below, so I'm pretty sure that you won't have any problems with any of this.

One of the main dissimilarities between the two frames is that **you can actually switch both the music and animation on and off here**. This created a need for an event triggered by the "(Global Value) Menu Animation is equal to 1" condition, which results in actions bringing the menu to life once again. You can observe it in the fourth event from the top ("Menu Animation=1"). Also, notice that event number 3 no longer destroys the "hidden.cursor" object, as it did in the previous frame.

You should also take a look at the "**Music**" and "**Animation**" Event Groups, responsible for switching the soundtrack and background animation options on and off. Everything from event number 20 to event number 30 is worth checking out... Especially a "safety trick" I had to use, utilizing the previously created (in part one of this tutorial) "**Safety**" Global Value.

"What is this *safety trick*, what is he talking about?", you may ask. Basically it works like this: if you're using one of the objects as a flag of sorts (pressing Enter while it's selected can turn an option on or off, depending whether it was turned on or off before – like the "Music: ON", "Music: OFF" option in our game) there is a probability of a mix-up in the events list, which will lead to the incapability of switching the flag between the two possible options. You have to use

a “safety switch” to build your events around this, another value that is set to “1” when an option is changed and set back to “0” somewhere lower on the events list (take a look at event number 28 in our current frame). Thanks to this “safety lever”, which is incorporated in our flag-switching event, you can safely create events such as the ones marked as 21, 22, 25 and 26 on our event list.


See? I told you that none of this is really that hard... Try to reverse-engineer this frame a little bit more, to take out all of its secrets to the light. Remember that thanks to using Global Values we were able to transport the player’s option selections between the frames: switching the music off in the Options menu will also switch it off in the main menu screen. Check it out for yourself. Anyways, play as much with this frame as you wish. Once you’re done, then... Well...

**You’ve successfully completed another tutorial! You’re done here!**

Congratulations, dear menu builder! You’ve made it, and you’ve got some really great looking menus to prove it! Try to expand these effects and options a bit, add some new preferences, create a “Game Info” screen, add some sparkling new animations here and there... Just do what you will, young cadet, as it is truly so that practice makes perfect! We’ll most probably see each other soon enough, in another tutorial. Oh, and I’ll most probably take Spencer with me. He’s a really great companion...

*Thanks for your time and see you again soon!*

***Cheers!***

The logo consists of the word "Koobare" in a white, sans-serif font, centered within a black rectangular box with a thin white border.

*If you have any questions, suggestions or just need help –  
mail me at [marchewkowy@gmail.com](mailto:marchewkowy@gmail.com)*



You have been reading...

HOW TO CREATE  
A GREAT LOOKING

# MENU

...AND LIVE TO TELL THE TALE.

## PART TWO MAIN MENU & OPTIONS

brought to you by  
**Koobare**

Clickteam's funkiest  
~~mercenary~~  
*menu builder*



Created for Multimedia Fusion 2 & Multimedia Fusion 2: Developer

Always be sure to have your MMF2 up-to-date!