GML and GameMaker Scripts Tutorial
Alex Schneider (aschneid@ucsc.edu)

Introduction

The following tutorial is a brief introduction to using GameMaker Language (GML) and scripts in GameMaker. If you are familiar with programming (have taken at least an intro class like CMPS 5J or CMPS 12A) we would recommend at least giving GML and scripting a look as it might make more sense to you. This tutorial will be focusing on effective ways to take advantage of scripting and GML and NOT in-depth details about the language's syntax and capabilities (however we will very briefly discuss some syntax stuff). If you only want to know more about the functions and syntax of GML, follow this link: http://wiki.yoyogames.com/index.php/Main_Page. This is the GameMaker wiki and it contains detailed documentation of various aspects of GameMaker including GML.

Syntax

A couple of basic things to note about syntax:

• You must end any statement with a semi-colon;
• Define a variable with the “var” keyword.
  ○ EX: var currentHealth;
• “if” statements and loops (for and while) work the same as they do with other languages

“Execute a piece of code”

While modifying an object in GameMaker, you can select the “execute a piece of code” action under the “control” tab to the right (See Figure 1).

Figure 1: The action highlighted in red is the “execute piece of code action.”
This allows you to write a bit of GML that will be executed during the selected event in the same way the drag and drop elements are executed. The code seen in Figure 2 will cause the object to lose 10 health upon colliding with an enemy using GML code rather than the drag and drop interface.

Figure 2: A simple use of the “execute piece of code” action.

Let's take a look at a piece of code in both GML and the drag and drop interface that perform the same actions. Figure 3 and 4 both show a snippet of code that decrements the object's health upon colliding with an enemy and then checking if the health is less than 0, destroying the object if so.

Figure 3: Damage and death checking for an object using code.
Through Figure 3 and 4, you should be able to see how the drag and drop visually represents the GML code. Keeping this in mind, you can use GML to perform all of the function seen in the list of actions and more. Those of you with prior coding experience are most likely used to seeing code like that seen in Figure 3 and may have an easier time using that. GML also offers a wider range of functions that could allow you to create a feature that might be difficult or impossible to create using only the drag and drop interface.

**Scripts**

Scripts are another aspect of GameMaker that further increases its capabilities and can ultimately lead to you creating a better game. Scripts can be used in a variety of ways in GameMaker, one of which is to use them in a similar fashion to function calls in a programming language. We could easily take our code snippet from Figure 3 and put it in a script. Doing this would allow us to call that script from any object in the game that needs to check if it is dead and prevents us from having to constantly rewrite code for checking if an object has been destroyed. Figure 5 and 6 show how to create a script.
Figure 6: Enter your code into the box that appears when creating a script and the name of the script in the top right corner.

Now that we have created a script, we can now call it with an action within an object. This will be demonstrated in Figure 7.
During this object’s step event, we are now going to be calling the “Check Health” script created in Figure 6. This will cause this object to run this script every time the step event occurs.

This script we created can now be used for any object that needs to run this code. These scripts can also take in variables through the argument values seen in Figure 7. These work similar to passing in arguments to a function in a programming language. You can use these arguments in the script by using the variables called argument0 through argument4. An example of this can be seen in Figure 8.
Figure 8: The box highlighted in red shows argument0 taking in the value of 100 upon executing this script. We can see in the box highlighted in green that we assign this object's health to the value passed into argument0.

Conclusion

That just about covers it for basic GML and script uses. If you have any sort of coding background, a lot of this will make a lot more sense. No matter what your coding background is, don't feel obligated to use anything mentioned in this tutorial – it isn't required to create a game good enough to do well in this class. Good luck with your games and feel free to e-mail us if you have any questions about this tutorial or about GML and scripting in GameMaker.