|  |  |
| --- | --- |
| ITB logo portrait B&W | INSTITUTE OF TECHNOLOGY BLANCHARDSTOWN |

A Taster of Computing

[[VERSION – Unity 2D – C# language]]

Gravity Guy 2D (2015) - a little computer game...

Part 5 – Game Over and Welcome screens (and buttons)



Welcome to “Gravity Guy”. In this multimedia programming exercise you will create a little 2D computer game.

CONTENTS

1 Aims of this part of the tutorial 2

2 Add a ‘Game Over’ scene to our project 2

3 Add code in Player to goto Game Over when last life lost 6

4 Add a ‘Welcome’ scene to our project 9

# Aims of this part of the tutorial

## New features / skills to be learned in this part of the tutorial

In this part of the tutorial you will add the following features to our game:

* Add a Game Over screen to the game, when last life is lost
* Add a Welcome screen to the game, and a button to PLAY

# Add a ‘Game Over’ scene to our project

## Get organised – create a plan for all the ‘scenes’ in our game

Interactive Multimedia games and applications tend to break up naturally into different screens, or levels, or ‘scenes’ as they are know in Unity, such as:

* Welcome to the game
* Main Menu
* High Scores
* Credits
* Instructions
* Level 1 playing
* Level 1 complete (with button to start level 2)
* Leve 1 lost / Game Over
* Level 2 playing etc.

Let’s think ahead, and play the scenes our game will need, let’s assume we are going to add a second level to our ‘Gravity Guy’ game, we’ll need the following scenes in the end:

* Scene 0 – Welcome screen
* Scene 1 – Game Over
* Scene 2 – level 1 playing
* Scene 3 – level 1 Complete
* Scene 4 – level 2 playing
* Scene 5 – Game Complete

Currently we have 1 scene, which will become our “Scene 2 – Level 1 playing”. We’ll add the Welcome and Game Over scenes in this part of the tutorial.

**Note**

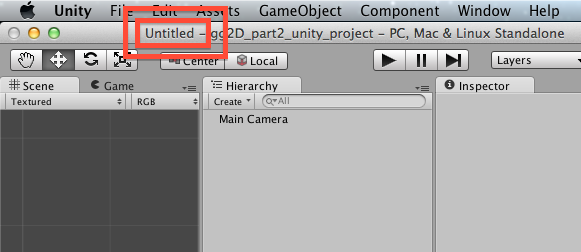
In reality the sequence of the scenes in Unity does not matter EXCEPT FOR THE FIRST SCENE (scene 0) – since that is the scene that runs when the final deployed game is started up ….

## Rename current scene “scene2\_level1Playing

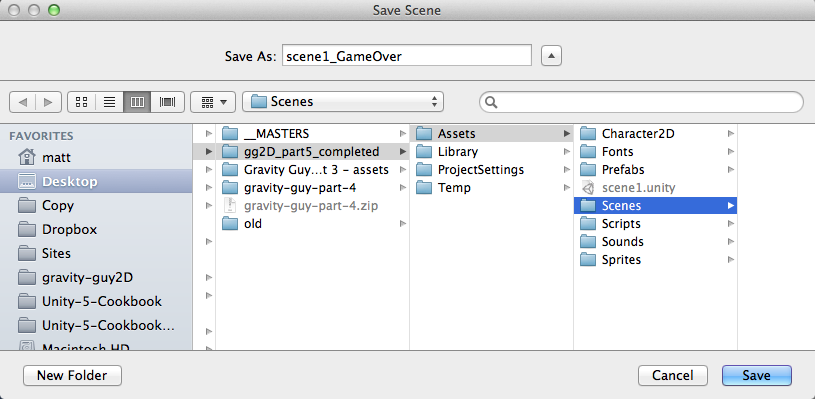
Let’s tidy things up, by creating a folder for scenes, creating a new scene in our new folder, and moving and renaming our original scene containing the hero and platforms etc.:

1. Save the current scene (currently named Scene1).
2. Create a new folder in the Project panel named “Scenes”
3. Create a new scene via menu: File | New Scene

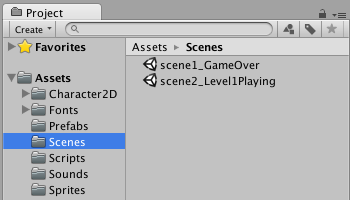
* You should now see that all game objects in the Scene and Hierarchy panels have gone: Except for default **Main Camera** that is always added to a new scene
* You can also see that the NAME of the scene you are now working on (in the Unity application window title bar) has changed from “scene1” to “untitled”



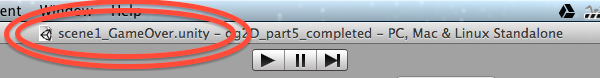
1. Save this new scene into your “Scenes” folder naming it “scene1\_GameOver



1. In the project panel, rename your original scene to be “scene2\_Level1Playing”, and drag this scene into your folder “Scenes”



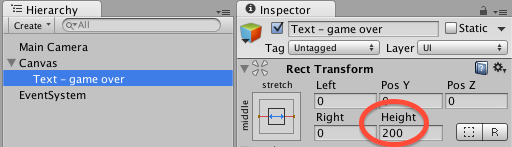
You should now see that you are editing scene **gameOver:**



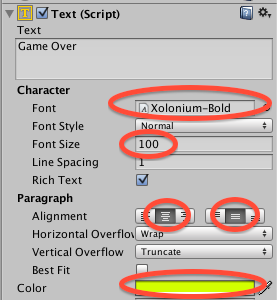
## Add a UI text message stating that the game is over

Add a big text message to the user using a UI Text object

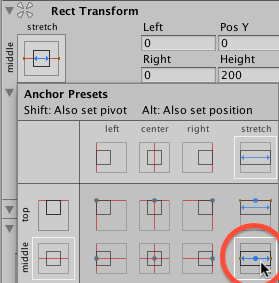
* From the “Create” menu of the **Hierarchy**, choose UI | Text
* Rename this item “Text – game over”
* In Inspector component Rect Transform set its height to 200 pixels



* In Inspector Component Text (Script) set
  + Text to: Game Over
  + Font to: Xolonium-Bold
  + Font Size: 100
  + Pargraph Alignment: center horizontally and vertically
  + Color: yellow



Finally let's position this UI Text object to be vertically centered and stretch to the whole Game panel width, via the presets of the Rect Transform component. With **Text – game over** selected in the **Hierarchy**, and hold SHIFT and ALT choose **middle-stretch** in the popup presents:



Now SAVE YOUR NEW SCENE’s content !

You should now see a new Canvas > Text\_-\_game\_over gameObject in the Hierarchy and Scene/Game panels:



# Add code in Player to goto Game Over when last life lost

## Edit Player code, to load scene “scene1\_GameOver” when lives less than zero

We now need to edit our Player script class, so that when the last life is lost (lives < 0) we make Unity change to our game over scene.

Let’s back to scene2\_Level1Playing, and then edit the Player script class …

* Ensure the game over scene has been saved
* load **scene2\_Level1Playing**
  + Double click **scene2\_Level1Playing i**n the **Assets/Scenes** folder of the **Project** panel
* Double click the **Player** script class file to load it into the **Monodevelop** editor
* Edit the code as follows, so that we tell the Unity **Application** to load the “**scene1\_GameOver”** scene when the last life is lose
  + We’ll test the value of variables “**lives**” after we reduce it by 1 in the **LoseLife()** method:

private void LoseLife(){

lives--;

if(lives < 0){

Application.LoadLevel("scene1\_GameOver");

}

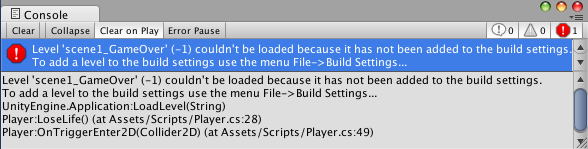
playerDisplay.UpdateLivesText(lives);

MoveToStartPosition();

}

## Playtest your game

Starting with the editor in **sene2\_Level1Playing**, run your game. Then keep falling off platforms to make the number of lives less than zero …. What happens?



The game did NOT load our **gameOver** level ?? Why not?? And what is that error message:

**Level 'scene1\_GameOver' (-1) couldn't be loaded because it has not been added to the build settings.**

**To add a level to the build settings use the menu File->Build Settings...**

This is what has happened …

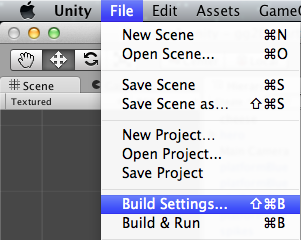
* Unity will only let you load a scene if the scene you are trying to load is included in the list of scenes that will be packaged up into the final ‘built’ application (whether a stand along desktop application, or web player game, or mobile phone app, or console game etc..)
* Sometimes you will create scenes just to TEST things out
  + You’ll never include these in your final build
* Unity does not want you to be able to write in code the loading of a scene that will cause an error when the final application is built for deployment or selling etc.

The SOLUTION …. You must now start telling Unity which scenes to include in your build.

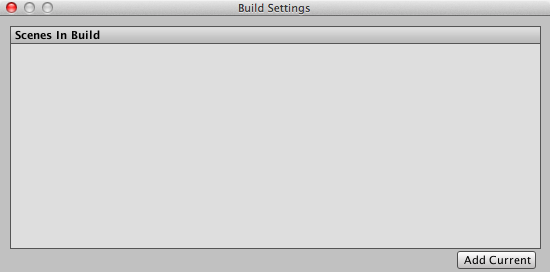
## Add ‘scene1’ and ‘gameOver’ scenes to your build list

Let’s add ‘scene1’ to the build:

* When editing **scene1** choose menu: File | Build Settings …



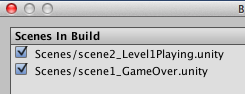
* Delete any scenes that might be listed already, then click the button “Add Current”:



* You should now see **scene2\_Level1Playing** added to the list of Scenes in Build:

Macintosh HD:Users:matt:Copy:2014_TEACHING:comp2_IMMedia:gravity-guy2D:figures:gg-part-5:gg_pt5_12_playing_added.png

* Now load scene **scene1\_GameOver**
  + You can either close the Build Setting dialog window, and re-open it after loading **scene1\_GameOver**, or just move this window to one site
* With scene **scene1\_GameOver** loaded, go back to the Build Settings dialog, and click “Add Current” again
  + You should now see **scene2\_Level1Playing** and **scene1\_GameOver** listed as scenes in the build
  + (note the scene list also have corresponding numbers, starting at 0 – Application.LoadLevel() also accepts the integer ‘index’ of the scene to load, as an alternative to the text name of the scene file …)



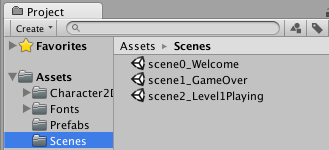
## Playtest your game

Start on the Welcome scene and click the button to play the game. When all lives are lost you should see the Game Over screen – unfortunately at present you are stuck there (we’ll learn about buttons later in this part of the tutorial)

# Add a ‘Welcome’ scene to our project

## Create new empty scene for first welcome to game

Ensure the scene you are working on is saved, and then create a new scene, naming it scene0\_welcome, saving this new scene into your Projects/Scenes folder:

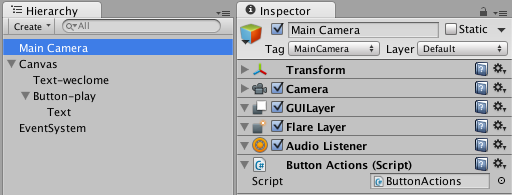


## Add Welcome in big text for the user to see

Create a new UI Text object named Text-welcome. As with Game Over, make it big yellow text in the center of the screen.



NOTE: If you were really clever, you would have DUPLICATED and rename your gameover scene to become your welcome scene, and then just changed the text being displayed…



Save your new scene.

## Add a button inviting the user to PLAY your game

Create a new UI Button object named Button-play.

With SHIFT and ALT held down, position this button at the **bottom-center** of the screen

* Clicking the appropriate box in the **Rect Transform** presets for anchors and positions

Select the Text child of button Button-play, and change its Text (Script) text property to be the word “Play”.



## Playtest your game

When you run this scene, you’ll notice that the button changes color slightly when mouse over or when clicked

* but NOTHING HAPPENS when the button is clicked

we need to create an EVENT HANDLER – a method in a class that can be executed when the button detects a “Click” event.

## Create class with methods for button clicks

Our game at this stage is going to have a button on the first screen to play the game (make Unity goto scene2\_Level1Playing), and also on the Game Over screen it will have a button making Unity return to the Welcome scene. So let’s write a special class to declare methods for these actions.

In the Scripts folder of the Project panel, create a new C# file named ButtonActions, containing the following code:

using UnityEngine;

using System.Collections;

public class ButtonActions : MonoBehaviour {

public void BUTTON\_LOAD\_SCENE\_WELCOME(){

Application.LoadLevel("scene0\_Welcome");

}

public void BUTTON\_LOAD\_SCENE\_LEVEL1\_PLAYING(){

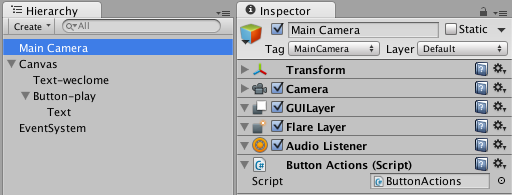
Application.LoadLevel("scene2\_Level1Playing");

}

}

Note, these are PUBLIC and VOID methods – which is required for them to be assigned as UI Event handlers.

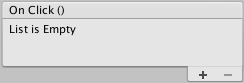
Next add an instance of this class to the Main Camera (we have to put this somewhere – if in doubt, put instances of classes in the Main Camera). So select the Main Camera in the Hierarchy, and then drag script ButtonActions into the Inspector. You should now see an instance of ButtonActions as a component of the Main Camera.



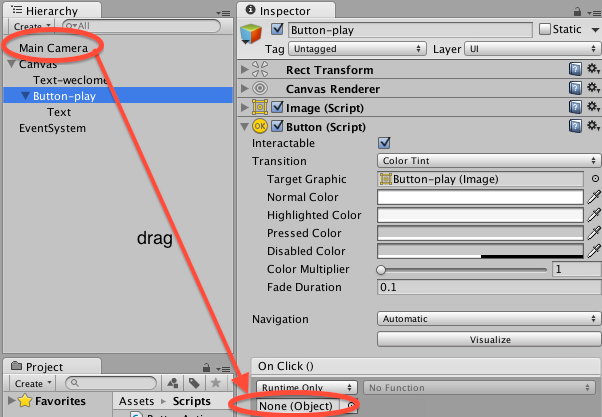
Save your scene.

## Define method BUTTON\_LOAD\_SCENE\_LEVEL1\_PLAYING() as action for Play button OnClick event

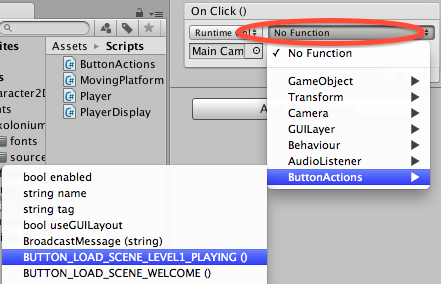
Select UI Button-play in the Inspector, and create a new OnClick event handler by clicking the plus sign + at the bottom of the Button (Script) component in the Inspector:



Underneath a dropdown (leave it at default “Runtime Only”) you’ll see a property slot to link to the gameObject that contains the scripted component where the OnClick method is located. Since we stored our event methods in an instance of ButtonActions in the Main Camera, then we need to drag the Main Camera into this object slot:



Now we can choose a “function” (method!) to be executed when this button receives an OnClick event. From the drop-down menu choose ButtonsActions and then BUTTON\_LOAD\_SCENE\_LEVEL1\_PLAYING:



Now when you run the scene, you should be able to click on the Play button and your level 1 playing scene should begin!.

## Add a “Welcome scene” button to your Game Over scene

Ensure your welcome scene is saved.

Load (double click) your game over scene, and add a button. Do the following:

* name the Button-load-welcome
* change the text to “Welcome scene”
* add an instance of ButtonActions to the Main Camera in this scene
* add an OnClick handler to the button, that will call method BUTTON\_LOAD\_SCENE\_WELCOME from the instance of ButtonActions in Main Camera
* save the scene

Finally, you need to add the Welcome scene to your projects “build” list.

## Playtest your game

Start on the Welcome scene and click the button to play the game. When all lives are lost you should see the Game Over screen, and when you click the welcome screen button you’ll be brought back to the welcome / main menu screen.

**Congratulations**

**You have now created part 5 of the tutorial !**