Creating an Interactive Application

Lee George Gauci - HND in Interactive media Yr2

Contents

TASK 1 – (P1.1)

Identify interactive media systems by answering the following question: Which are the most important criteria when it comes to choosing a game engine? List five criteria and justify each criterion with a sentence.

Understand the Genre

First off you need to understand everything about your game not just the basic story line ,is it going to be a first person shooter or a simple side scroller. What kind of mood is the game going to produce and the graphic quality is it going to have? How comfortable are users going to be playing your game. These questions have to be well thought out and defined before you can even start about choosing a game engine.

Figure Out the Dimension

Is it going to be in 2D or 3D? This two are completely different working factors and choosing an engine dedicated your game dimension will help greatly. There are engines that work in both dimensions but still they favor their own dimension proving to be more effective in one of the two.

Determine the Platform

What platform is the game going to be played on, or is it going to be played on more than one, there are many platforms in the market (PC, Mac, PlayStation, Xbox, ETC) and certain game engines will not allow you to publish your games on some of them. This is a key component that you will need to figure out as it could create a game crushing problem in the end.

Choose a language

What is your preferred language? It makes sense to pick a language native to your program such as C# on Unity. This isn't completely necessary but it will surly help in the long run. There are many game engines with many languages so one must be sure to choose an engine that caters for ones preferences.

Identify Your Game Plan

Now it's time to think about what happens after the game is created. What is your end game? Are you creating a free HTML5 Game or are you going to publish it on one of the major platforms in the market? Many game engines require a you to pay royalties from games which are producing an income and if the project is not planned well you can fall under quite fast.

http://mightyfingers.com/blog/11-steps-for-choosing-the-best-game-engine/

TASK 2 – (P1.2)

Evaluate constraints affecting interface design by completing the following task: Explain how different screen sizes and aspect ratios are catered for in unity. How does the game engine cater for different screen sizes / resolution? Write a description and show one screenshot of how the screen limits are calculated based on the camera viewport, with reference to the camera. ScreentoviewportPoint method.

Catering for different screen sizes

The most effective way is to use anchors which are built in Unity. Anchors set the default position of objects/assets on screen and hold them there regardless of the canvas size and dimensions. This means that if I were to anchor a button to the bottom left of the canvas at an aspect ratio of 3:5 and later change the aspect ratio to 4:7 the button will still be found at the bottom left. This helps when building games for multiple screen sizes Ex Mobile Games.

ScreentoViewportPoint

The reason this line of code exists is to convert the screen coordinate system (Screen Space) which is measured in pixels into Viewport coordinate system (Viewport space) which is relative to the camera. Screen space is calculated by getting the bottom left coordinate which is (0, 0) and the top right is (pixelWidth, pixelHeight) and the z position is the world units from the camera. On the other hand Viewport space is calculated by getting the bottom left of the camera which is (0, 0) and the top right which is (1, 1) the z position is also world units from the camera.

```
using UnityEngine;
using System.Collections;

public class ExampleClass : MonoBehaviour {
    void Update() {
        transform.position = Camera.main.ScreenToViewportPoin
    }
}
```

TASK 3 – (P2.1)

Evaluate different sound editing tools for implementing a sound effect in your game. Show how you implemented a sound effect in your game with a sound editing tool of your choice and indicate the reason why you chose that particular sound editing tool for your game sound. You must also include a description of how sound effects are imported and implemented in Unity, with reference to a unity project where sounds have been imported.

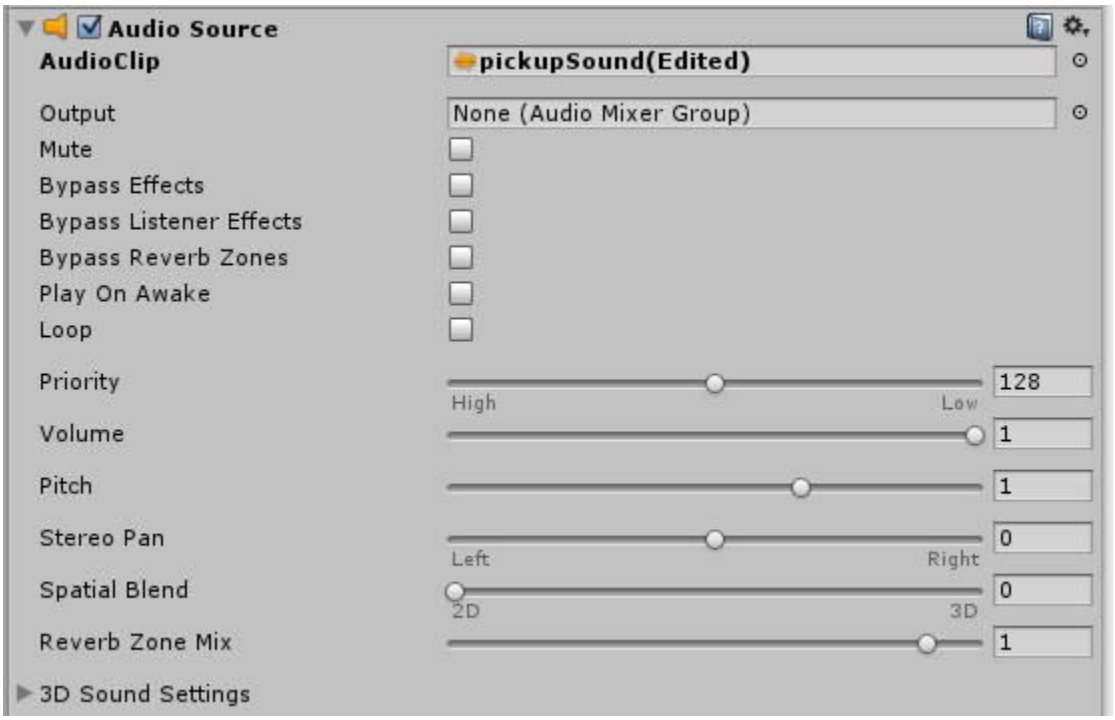
Sounds are first imported into unity by simple drag and drop or they can be pasted manually in the assets folder in your game folder.

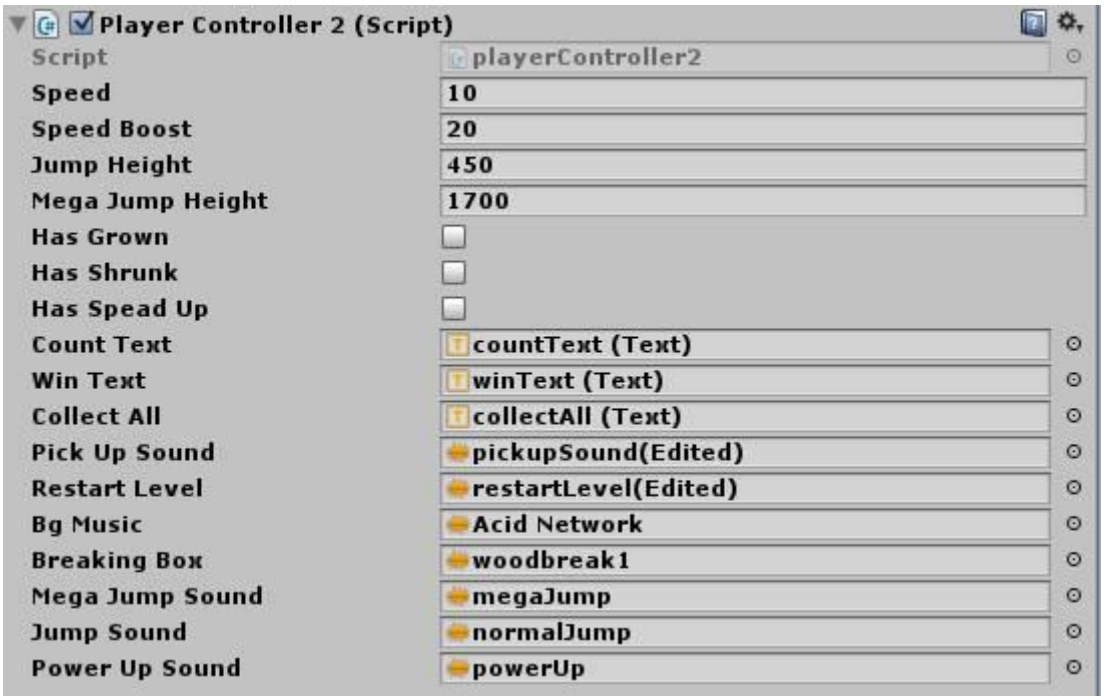
```
private AudioSource source = GetComponent<AudioSource>();
public AudioClip pickUpSound;
```

In your code you must then declare 2 variables: The first one pulls the AudioSource Component from your game object this script is attached to and is named to source and the second one allows you to place the desired audio clip in the unity inspector.

```
if (other.gameObject.CompareTag ("pickup"))
{
    source.PickUpSound(pickUpSound);
    other.gameObject.SetActive (false); //hide other object
    count = count + 1; // 1 point to the score
    SetCountText (); //run the SetCountText Function
}
```

You must then set when you want the sound to be played in my case when you pick up an object with the tag pickup. Source refares to the GetComponent<AudioSource>() which we declared earlier. The brackets contain the name of the audio clip we have set in the unity inspector.





Lastly you must add an audio source component to your game object and simply drag the sound from your asset folder to the audio clip tab and untick play on awake as to not have it play when the game starts, also drag and drop the sound in your player controller script from the inspector.

Why did I Choose Audacity

Audacity provides quick and easy sound modification whilst also provides loads of fast conversion from different formats. Most of all it is free to the public to use as you wish.

TASK 4 – (P2.2)

Record and generate an effective sound for your game, using any online sound generator or the built in audio editing tools in Unity. To do this you need to present the following three .mp3 format sounds:

- ***The original sound recorded.***
- ***A description of the effects applied to the sound.***
- ***The final modified sound.***

Indicate the folder where the files are located in your documentation.

Changes:

I set the pitch from B(3) to A#/Bb (1)

I changed the frequency from 253.448 to 59.806

Location:

In the Game Folder –

CIDPAssignmentDistinction2016\Lee George Gauci\EnviromentGame

TASK 5 – (P3.1)

Explain how to raise events in-game, with reference to the collision/trigger functions in Unity. To achieve this task, paste a sample collision function from your code in your documentation and explain the sequence of events and the use of tags in the collision function you have implemented. You must also explain the difference between a collision and a trigger

```
void OnTriggerEnter(Collider other)
{
    if (other.gameObject.CompareTag ("breakablebox")) {
        Destroy (other.gameObject, 1.2f);

        Vector3 jump = new Vector3 (0.0f, jumpHeight, 0.0f);
        GetComponent<Rigidbody> ().AddForce (jump);

        isFalling = false;
        canMove = false;
    }
```

Void On TriggerEnter(collider other) is only run when objects with a collider hit each other and triggers any sequence of code only if specified with the trigger tick box in the collider component. The if function states that if the collider of the object attached to this script (ball) touches another object with a collider under the tag “breakable” The following code will be run.

1. If function checks if the other game objects has the breakable box tag before continuing.
2. Line 1 – The game object will be destroyed.
3. Line 2/3 – The ball is going to be pushed up using the new Vector 3 method.
4. Line 4/5 – Setting Booleans to false.

A collision is the Bounding Box located around the objects. This can be set through “Add Component” function in Unity, It will not allow two objects to occupy the same space at any given time sort of like giving them mass there are several types of collision boxes which can be useful in different situations. The trigger is a tick box found in the Inspector when the Collision box is selected. On the other hand if the collider with trigger selected touches another collider it can trigger events which are set through code.

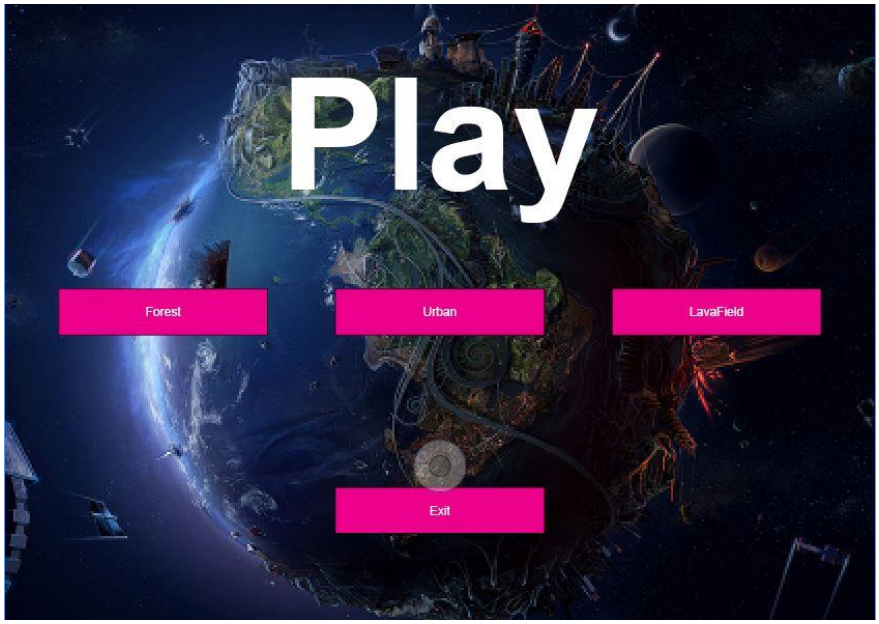
TASK 6 – (P3.2)

Draw a sketch of two screens of your game:

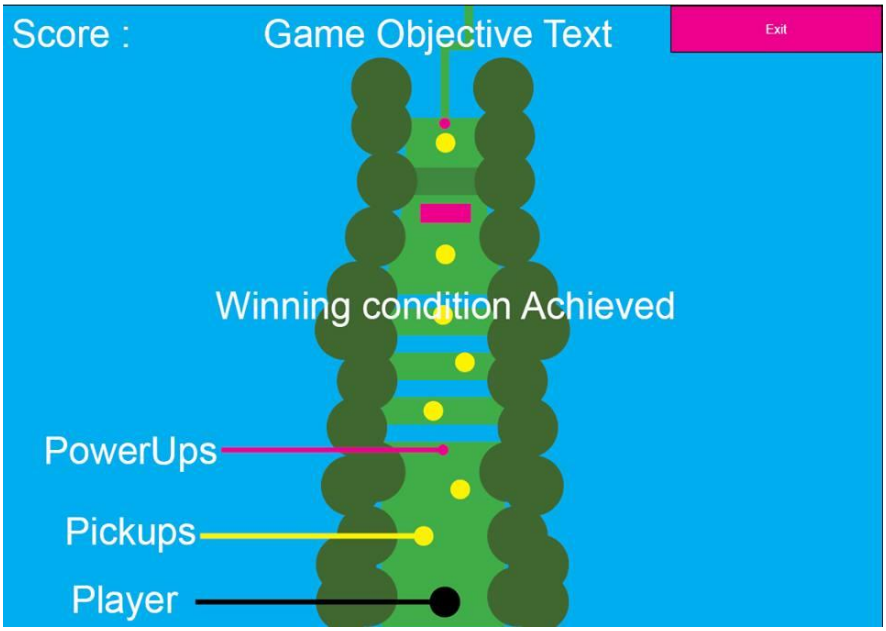
• Main menu • Game screen The sketches will show the positions of your GUI elements.

The sketches must reflect the layout of the game.

Main Menu



Game



TASK 7 – (P4.1)

Conceptualize an interactive system by explaining the reason for the following 3 methods: Start(), Update(), StartCoroutine("coroutinename") Explain the role and timing of each of these methods by using an example from one of your own scripts.

Void Start

```
void Start ()
{
    SetCountText ();

    count = 0;

    winText.text = "";
    collectAll.text = "";

    isFalling = false;
    hasGrown = false;
    megaJump = false;
    canMove = true;

    rb = GetComponent<Rigidbody>();
    source = GetComponent<AudioSource>();
}
```

Void start holds the code that is immediately run when the game is booted up.

Void update

Void update holds the code which is continuously referred to throughout the game, for example if your players has a jump function, the jump function reeds to be readily available at all times. If the user pressed the jump key void update takes care that the input is being continuously monitored.

```
void Update ()
{
    CanMove ();

    if (Input.GetMouseButtonDown(0) && isFalling == false && megaJump == false)
    {
        source.PlayOneShot(JumpSound);
        Vector3 jump = new Vector3 (0.0f, jumpHeight, 0.0f);
        GetComponent<Rigidbody>().AddForce (jump);
        isFalling = true;
        canMove = false;
```

StartCoroutine("coroutinename")

```
if (other.gameObject.CompareTag ("grow"))
{
    source.PlayOneShot(powerUpSound);
    hasGrown = true;
    other.gameObject.SetActive (false); //hide other object
    StartCoroutine(powerUps()); // run the IEnumerator
}

if (other.gameObject.CompareTag ("shrink"))

{
    source.PlayOneShot(powerUpSound);
    hasShrunk = true;
    other.gameObject.SetActive (false); //hide other object
    StartCoroutine(powerUps()); // run the IEnumerator
}
```

This function has the ability to run code over a period of time using the ienumerator. This is only run when called for when it is stated in your code.

```
IEnumerator powerUps()
{

    if (hasGrown == true)
    {
        GetComponent<Animator>().SetTrigger("grow");
        yield return new WaitForSeconds (3);
        GetComponent<Animator>().SetTrigger("returnToNormal2");
        yield return new WaitForSeconds (1);
        GetComponent<Animator>().SetTrigger("Idle");
        hasGrown = false;
    }

    if (hasShrunk == true)
    {
        GetComponent<Animator>().SetTrigger("shrink");
        yield return new WaitForSeconds (4);
        GetComponent<Animator>().SetTrigger("returnToNormal");
        yield return new WaitForSeconds (1);
        GetComponent<Animator>().SetTrigger("Idle");
        hasShrunk = false;
    }

    if (hasSpeadUp == true)
    {
        speed = speed + speedBoost;
        yield return new WaitForSeconds (3);
        speed = speed - speedBoost;
        hasSpeadUp = false;
    }

}
```

This can be explained better with the example to the left. In the IEnumerator the if function states that if hasGrown = true the following will happen. It will trigger the animation wait for 3 seconds then trigger another animation wait 1 second and trigger the last one.

TASK 8 – (P4.2)

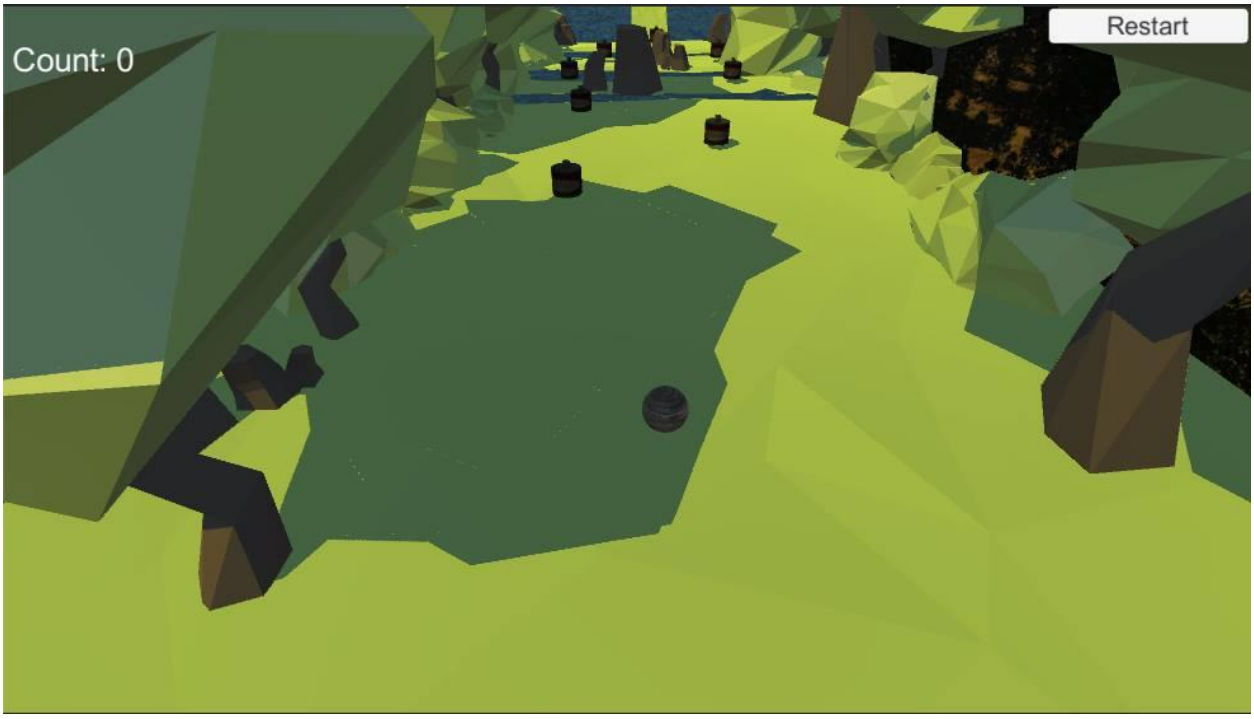
Present your game. The game must implement the following functionalities: • Player object with score • Player must implement at least 3 actions • 3 different game scenes with different background images • A fully featured menu with a new game/exit button using the Unity 5 GUI Canvas • 3 different player modifiers which can be picked up • A specified winning condition • A clearly documented educational element Your game must be presented as an html5 web player game. Credentials will be provided for the game upload. Include a screenshot of each game screen in your documentation, as well as simple instructions on how to play the game.

How it is played

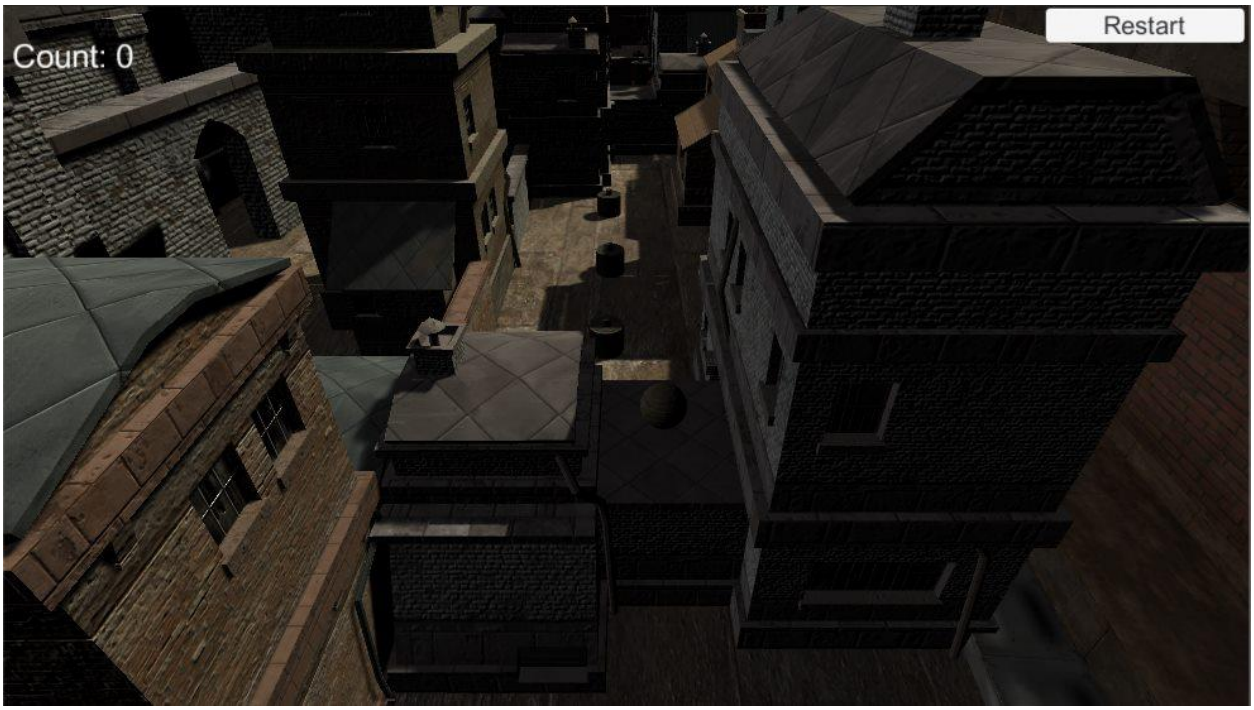
The game is fairly simple to play, the player must collect all the oilcans before he canpick up the final one which will boost him to the next level. The player must tavel from Point A to Point B using the joystick on mobile devices and the wasd keys on pc's to move and the jump button/ right mouse button to jump. Along the way the player will find many obstacles some of which have to have a power up active or find a platform to jump higher.

Main Menu

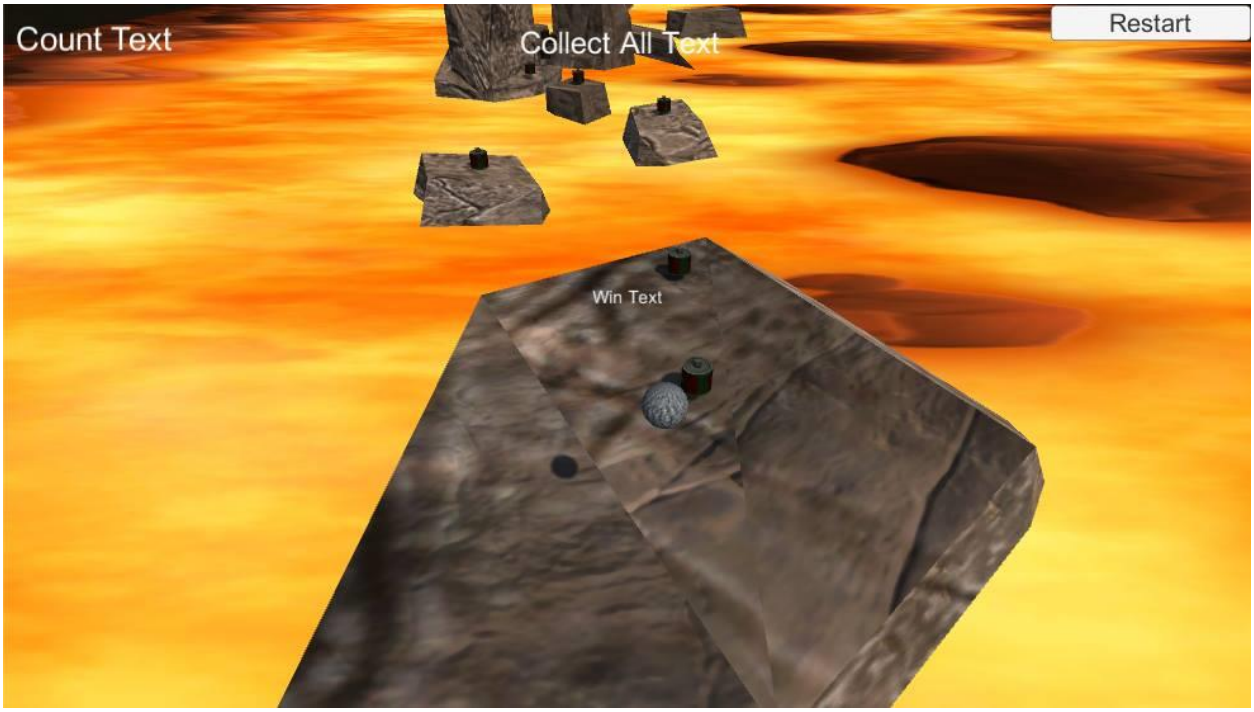
Level1 (Forest)



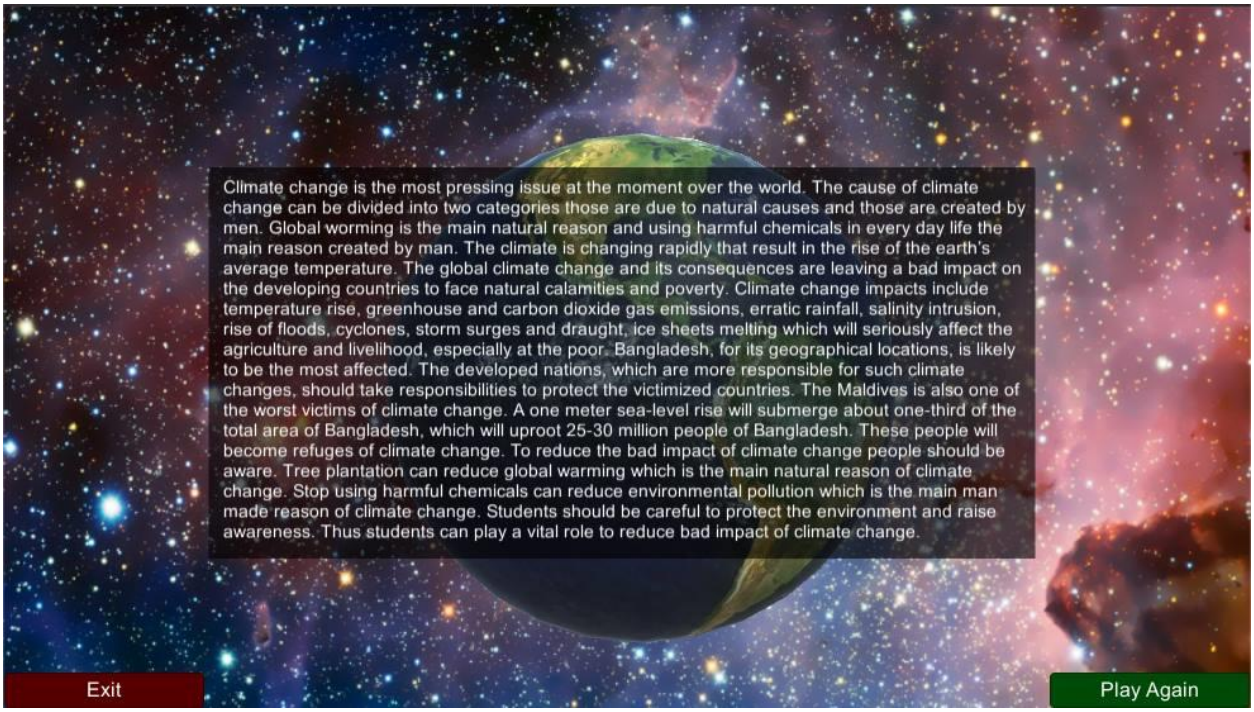
Level2 (Urban)



Level3 (Lava Field)



End Game Screen



Educational Element

The game is based on the climate change theme, As the user plays the game he has to pick up oil cans (referring to the negative products we use that harm the environment), with that being said the objective of the user is to harm the environment (something that we do unknowingly.) as the game progresses and the user travels to each level he will notice the drastic change in environments. This is to show what effects we as humans have on the environment, the forest level shows where we started from, The urban level shows where we are now and if we keep going the way we are going the last level (slightly exaggerated) is showing that we will ultimately be the destruction of our own planet.

TASK 9 – (P4.3)

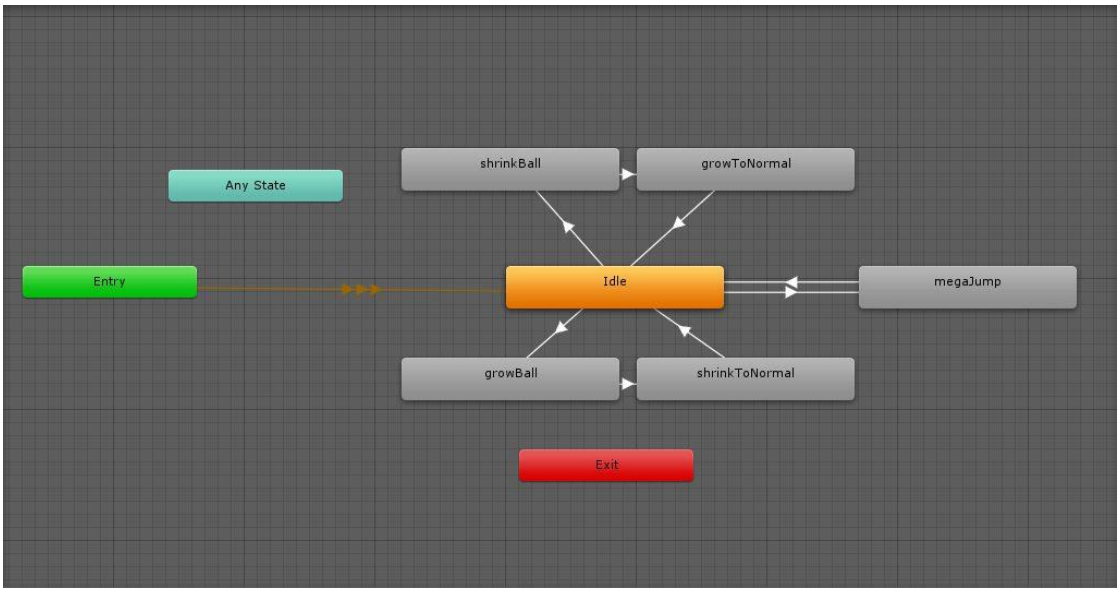
Write a short paragraph explaining what improvements you would have implemented in the game if you had more time.

The game as it is, is a basic version of an rpg game. I would have created custom 3d models and animations through the whole game to keep it consistent. I would take a lot more time planning more thoughtful layouts of the maps and better ways to incorporate power ups. Spend more time creating/finding/manipulating better music and sounds to keep consistency through the game and also give more feeling as it is being played. I would also take time to manage my assets folder better and clean up my code a bit more.

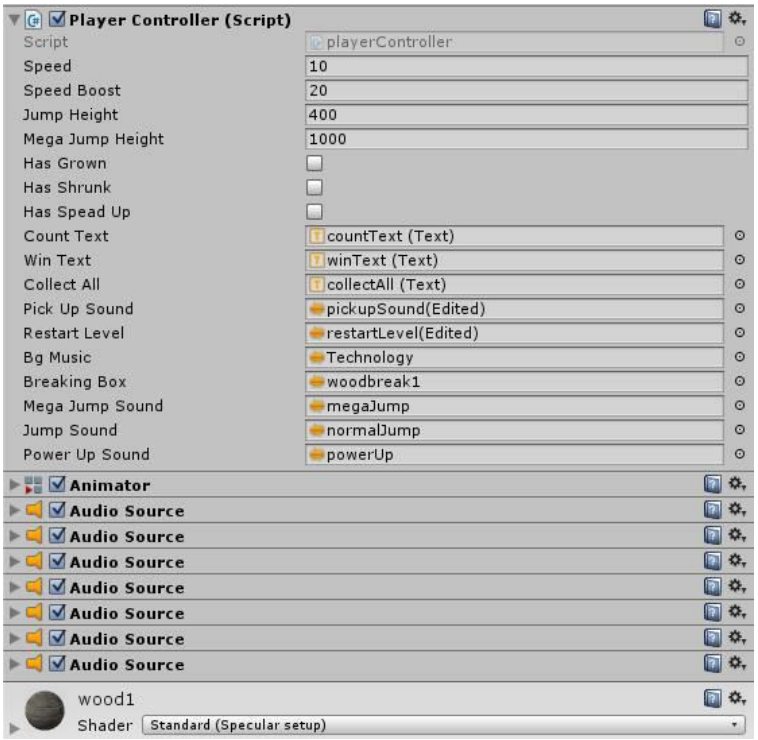
Task 10 – (M1.1) / (Task 12 – (M3.1)

Show that complex problems with more than one variable have been explored in the context of presenting an interactive application by adding the following functionality to your game: 3D model with animation and animator states Full use of sound with at least 5 different sound effects Background music

Evidence of animations:



Evidence of Sounds/Background Music:



Task 13 – (M2.1)

Apply relevant theories and techniques to creating an interactive interface project by writing a short paragraph explaining specifically what genre of game you have created. Include references to similar games which have been created. Show similarities using 2 screenshots. (4 in total, 2 of your game, 2 of the game you used as a reference)

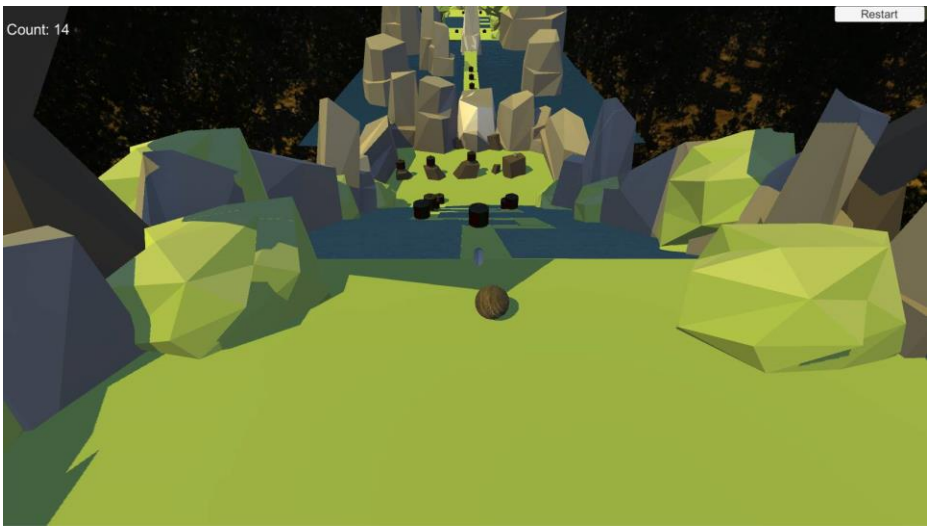
The game is a basic rpg / platform game as the player has to traverse the world and overcome obstacles in its way to reach its goal. Such an example would be crash bandicoot beaing able to traverse his world and also be able to break boxes both of which can be found in my game.

Crash Bandicoot

Lvl2 (Urban)

Spyro

Level1(forest



Spyro is more obstacle oriented but still holds the same key factors. Go from point A to point B and to collect items.

Task 14 – (D1.1)

Show that realistic improvements have been proposed against defined characteristics for success by explaining how you would update your game to reflect one of the finished titles you used as your inspiration in Task 13. What features are you missing and how would you go about implementing them? To achieve this task, identify the missing features and explain how these missing features could be implemented in your game.

From crash bandicoot I noticed that I would need a breaking animation that the user can initiate to break boxes, also refine the obstacles and breakable objects to be more coherent with the game and animations themselves. To fix this ideally I would need a more complex model for the breaking animation or at least a more complex animation. As for the breakable objects I would create them from scratch and implement the animations through other 3d software such as Maya or 3ds max. That way I could have much more control on what is happening and when.

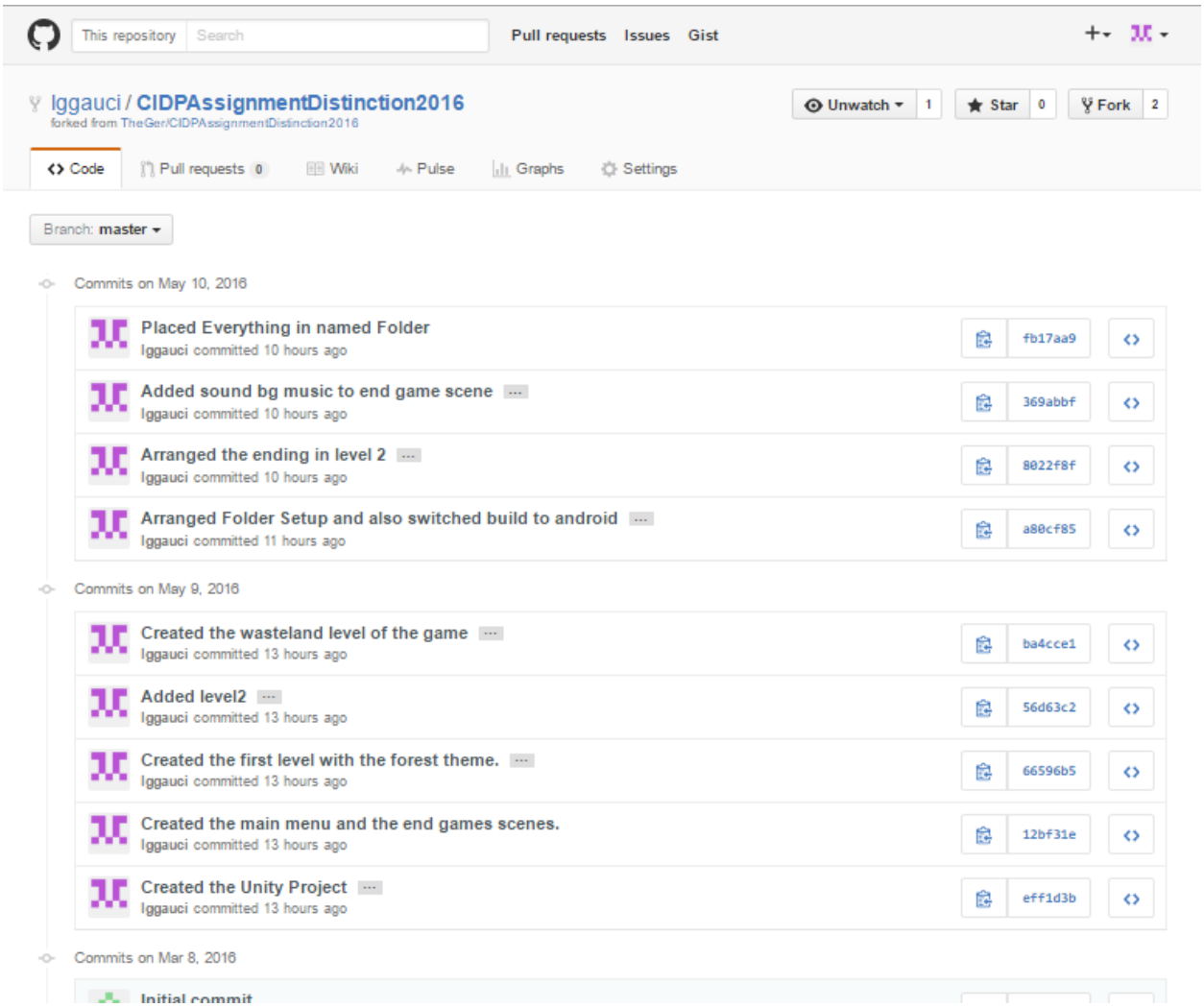
As for Spyro I would create a camera and script that would not follow the ball only from the back but rather give the ball a forward facing direction so that the camera would follow the direction that I am looking at, that way the game could become tremendously more interesting and could open up tons of new layouts which are more feasible to the user. To solve this I would research or create my own code and implement it into the game and make sure all functionalities work as they should. I also noticed the Open world aspect of the game which caught my eye again, if the camera would look in the direction of the player these layouts would be possible

Task 15 – (D2.1)

Show activities have been managed by forking the following Github project. https://github.com/TheGer/CIDPAssignmentDistinction2016
all commits must be FULLY documented with the exact features that were implemented in that commit. The sequence of commits must CLEARLY show how the game was implemented over a realistic time span. To achieve this task, include a link to the project on Github that you have been using as part of your project under this task heading, and include a screenshot of the list of commits you have carried out as part of your assignment. You must have at least 8 fully documented commits with a sequence of commit dates to achieve this criterion. Your submission must be a valid Pull request to the original project.

Link: https://github.com/lggauci/CIDPAssignmentDistinction2016

Screen Shot of Commits.



Commit 1
The Unity project was build and also many import for assets where added, mostly from the asset store. These are going to be used to build up the game. IE 3 Playable Levels, main menu and also the Endgame.

Commit 2
I created the main Menu and End Game Using simple ui buttons and creating the text. Also added the same background to each.

Commit 3
This is where I worked on the first level of the game, still not 100% still may need some bug fixes here and there but the level is playable both as web and also on mobile devices. I used the assets imported earlier. Still need to export apk and web gl.

Commit 4
I used the same methods and scripts to build up the second level. Only using assets that where based on the urban environment. The models can prove to be a little tricky as there mesh colliders don't always work so editing some box colliders in is a must.

Commit 5
Used some of the assets from the first to build the terrain apart from the lava floor which I implemented later, The level works, but still need some tweaking here and there. This is the wasteland level of the game.

Commit 6
Some of the assets were out of the assets folder and thus not being loaded, Replaced all items in their respective folders. All worked well, The GUI joystick was not working until I noticed that I was using unity on a different build setting, when switched to android it was visible and worked correctly.

Commit 7
The ending in level 2 was not allowing the player to move when landing on the ledge applied a tag to the buildings so that he can move and drop down.

Commit 8
The end game had no sound so I added an audio source with the same music as the main menu.

Commit 9
As this was going to be placed on the teacher's repo I placed the whole contents of my work in a named folder.

Task 16 – (D3.1)

Demonstrate convergent/lateral/creative thinking by performing the following task: Show how professional game development may be achieved by explaining how a professional team of different specialists would have worked on your game. Identify a list of job descriptions in the computer game development context and map them to specific parts of your game development. To achieve this task, you must identify 4 different job titles and map them to 4 different elements in your game. Once you have identified these jobs, create a timeline showing how the different developers are interrelated and which files and file-types need to be passed along between these developers to achieve a finished product. The timeline must be clearly presented and of professional quality and it must clearly show the work being carried out by the various specializations

Concept art

These artists would create a clear picture of what the game is going to look like before it has actually been put into development. These people would eliminate (not completely) the “off the top of your head” assets or gameplay which would arise during the development. In my case this team would have created the looks of the environments, the player, and the pickups and other material such as the breakable items and also the mega jump platforms. These people would then create the necessary .jpegs, png and .tiff files to be handed over to the next team.

Asset Design

This team is in charge of creating the actual assets in game which were previously created by the concept artists. The team would 3d model/texture and animate the 3d elements which are found in game(complex animations are best not done in unity) When the assets are ready they would need to export them as .obj or.mb files which later can be used by the game design team.

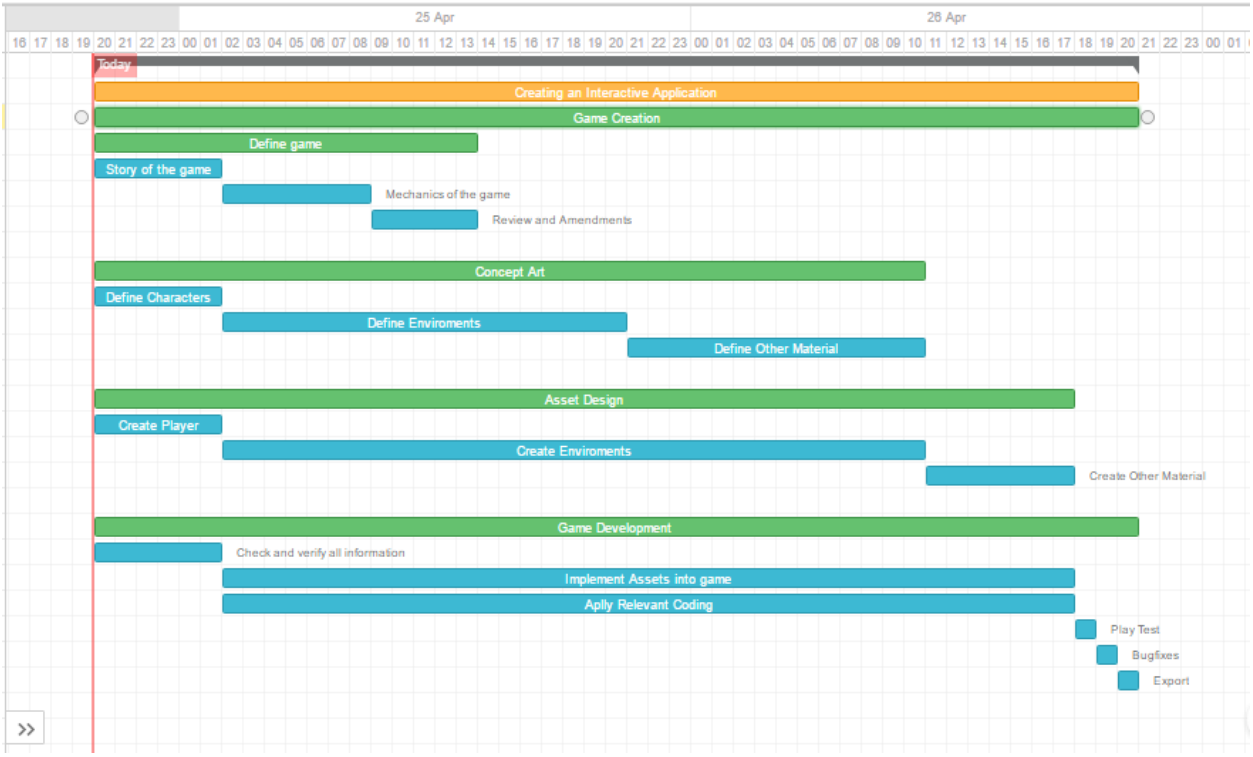
UI Design

The UI Team would create all of the buttons/backgrounds and displayed information which are found in game. In my case they would have designed the main menu. The buttons and displayed information throughout the game (Score, Win Text, Objective Text) and also the End Game screen and buttons. These could then be exported as images or font sheets so that the game system designers would know what the game will look like.

Game system Design

These people are in charge of creating the mechanics of the game. These would need to implement everything in unity and the necessary code along with it, these would have created the scenes in unity and layed out everything which was previously specified by the concept artists and also the UI Team.

Time Frame:



**Numbers are Days*