



CREATING AN INTERACTIVE APPLICATION

**YANIKA ZERAFA
347696M**

**HND INTERACTIVE MEDIA YEAR 2
COMPUTER DESIGN INTERFACE PRINCIPLES**

Table of Contents

Task 1 – P 1.1	5
Task 2 – P 1.2	6
ScreenToViewPoint.....	7
Task 3 - (P2.1).....	8
Task 4 - (P2.2).....	9
Task 5 - (P3.1).....	10
Task 6 – (P3.2)	11
Task 7 – (P4.1)	12
Start()	12
Update()	12
StartCoroutine(“coroutinename”)	12
Task 8 – (P 4.2)	14
How to play	17
Task 9 – (P 4.3)	18
Task 10 – (M1.1).....	19
3D models with animation in the game:.....	19
Full use of sound with at least 5 different sound effects:	20
Background Music.....	21
Task 12 – (M3.1).....	22
Task 12 – (M3.1).....	23
Task 14 – (D1.1).....	25
Task 15 – (D2.1).....	26
Commit 1: Change in Camera	27
Commit 2: UI Score Text	27
Commit 3: UI Scaler.....	28
Commit 4: Background Music Level 3	28
Commit 5: Win Scene sound.....	28
Commit 6: Power up Spawn Time.....	28
Commit 7: Game Title	29
Commit 8: Environment Spawn position	29

Creating an Interactive Application

Task 16 – (D3.1).....	30
Storyboard Design.....	30
Visual Experience Designers	30
3D Modelers.....	30
Game Production	30

Task 1 – P 1.1

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.

When searching for an appropriate search engine, one must keep in mind the following criteria.

What genre is the game?

First of all, one must have a clear idea of what the game is going to look like so as to be able to determine the following facts. Some game engines are built so as to accommodate the needs of some genres more than others. For example Unreal Engine accommodates perfectly the needs of first-person shooters.

What Platform?

Not all game engines can export to all platforms. Therefore this is one of the most crucial points. If you want your game to be exported as an android app, an iOS app and or any other platform, you have to make sure that the game engine supports this feature.

2D or 3D?

With unity you can work both in 2D and 3D, however, not all game engines are able to do so. Some are 3D exclusive while others, like Construct 2, supports 2D only.

Are you going to sell the game? Licensing

If you want to develop a game which is later going to be up for sale, you want to make sure that you choose the best game engine that has an appropriate licensing method for you.

High quality graphics?

Even though Unity can support 3D, high-end type of games with heavy graphics can't be created in this game engine. There are other game engines that are appropriate for these types of graphics such as CryEngine.

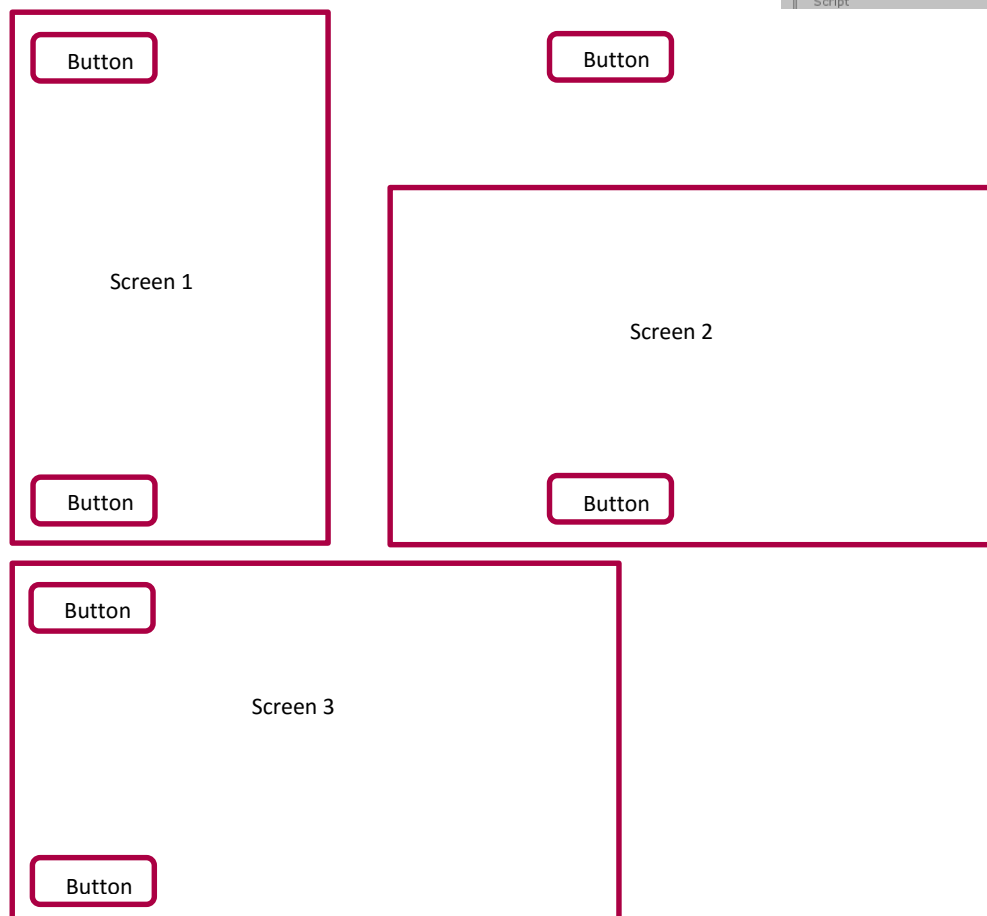
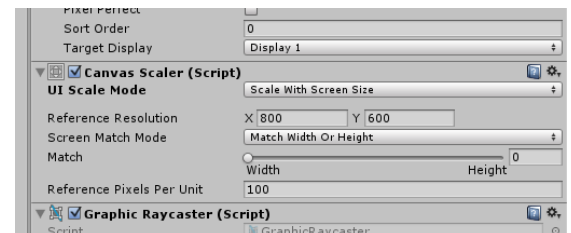
Task 2 – P 1.2

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. Screenshot of the ScreenToViewportPoint method.

The best way to set up UI items for different resolutions is to use anchors. Anchors enable the programmer to have the items and buttons set up according to their position relative to the screen size and not the canvas size.

Without anchors, buttons on screen 2 keep the same position as on canvas and not on screen. Anchoring the buttons to Top left and bottom left would in turn give the result as shown on Screen 3.

Also, when exporting for other devices, to have the items keep an appropriate size, it is important that the canvas scaler is set to 'scale with screen size' mode.



ScreenToViewPoint

Screen to view point is a function in Unity that translates the position of the items on screen into from screen space (pixels) to viewport space (relative to camera). These two basically represent the same area in the game, however, the difference between the two is the system used to calculate them. The calculation for screen to view point is the following: $X / \text{Screen width}$ & $Y / \text{Screen height}$.

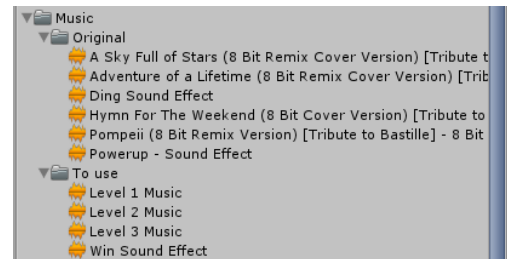
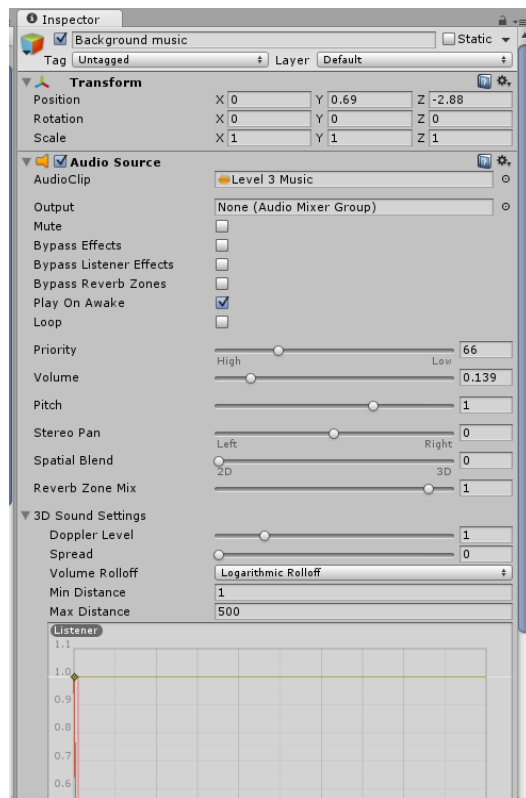
```
using UnityEngine;
using System.Collections;

public class ExampleClass : MonoBehaviour {
    void Update() {
        transform.position = Camera.main.ScreenToWorldPoint(new Vector3(Input.mousePosition.x, Input.mousePosition.y, 10));
    }
}
```

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.

To start, I researched the appropriate sound for my game. When I had the files, I imported them in my Unity project by pasting them in the Assets folder.



For the background music, that requires no particular triggers or conditions to play, I simply created an empty game object and added an 'Audio Source' component. The sound had to be played once the scene was loaded, so it is crucial to tick the 'Play on Awake' function.

Through Unity, I could modify some properties to my sound. For example, I lowered the volume of the background music so that the other sound effects can be heard clearly through the gameplay.

On the other hand, sounds that were to play on/after an event, for example the 'cling' effect when the player collects a can, is implemented differently. An audio source is attached to the player and the desired Audio clip is chosen. Since I didn't want the audio to play on the loading of the scene, I unticked 'Play on Awake'. In the player's script, when the player hits the can, I scripted the audio as shown in the screenshot.

```
if (other.tag == "can") {  
    score += 1;  
    AudioSource audio = GetComponent<AudioSource>();  
    audio.Play();  
    Destroy (other.gameObject);  
}
```

To edit the sounds I implemented in the game, I used the 'Audacity' software. This software is free for everyone, is fast, easy to download and install and offers a variety of features.

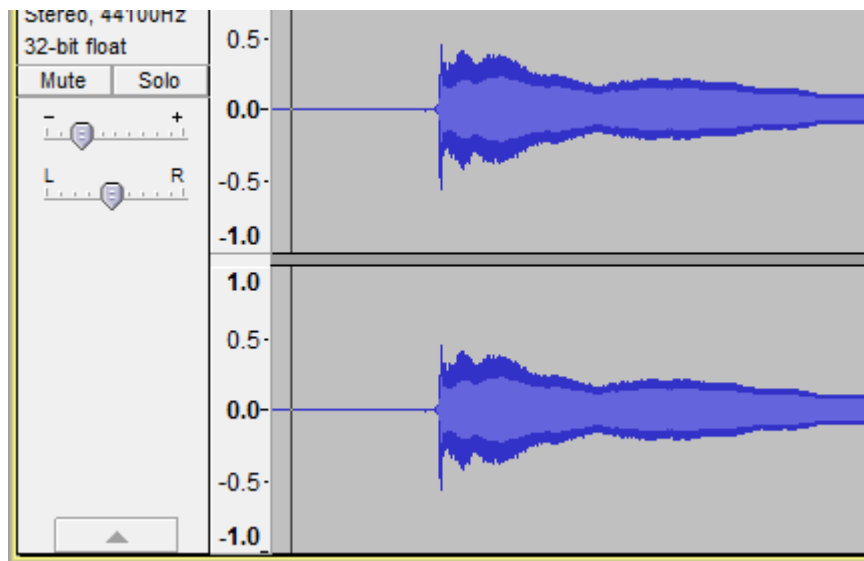
Task 4 - (P2.2)

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.

Since the recorded sound effect was too loud and noisy, I decided to lower the gain as shown below.



The original and edited sound mentioned can be found in the Assets folder in 'Music'. The original sound is in the folder 'Original' (named as 'cling sound effect original') and the edited sound is in the 'To use' folder.

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.tag == "palm1") {
        SceneManager.LoadScene("gameoverlost");
    }
    if (other.tag == "car") {
        SceneManager.LoadScene("gameoverlost");
    }
    if (other.tag == "chest") {
        SceneManager.LoadScene("gameoverlost");
    }
    if (other.tag == "tree") {
        SceneManager.LoadScene("gameoverlost");
    }
    if (other.tag == "bush") {
        SceneManager.LoadScene("gameoverlost");
    }

    if (other.tag == "floor") {
        isFalling = false;
    }

    if (other.tag == "can") {
        score += 1;
        AudioSource audio = GetComponent();
        audio.Play();
        Destroy (other.gameObject);
        if (score >= 20) {
            if (Application.loadedLevelName == "1") {
                SceneManager.LoadScene ("2");
            }
        }
        if (score >= 40) {
            if (Application.loadedLevelName == "2") {
                SceneManager.LoadScene ("3");
            }
        }
        if (score >= 60) {
```

The following image contains a snippet from the code applied to the 'player'. The prefabs and the player have been assigned specific tags. Through the use of these tags, when the player hits a particular object, another action is triggered. For example, when the player hits the object that has a tag of 'palm1', the Game over scene is loaded as the player has lost the game. Every tag can have a different action applied to it.

Both the player and all the prefabs have a collider (box collider for example). The collider are the boundaries to the objects. The void on Trigger Enter function starts when the player hits something else, which can be calculated through colliders.

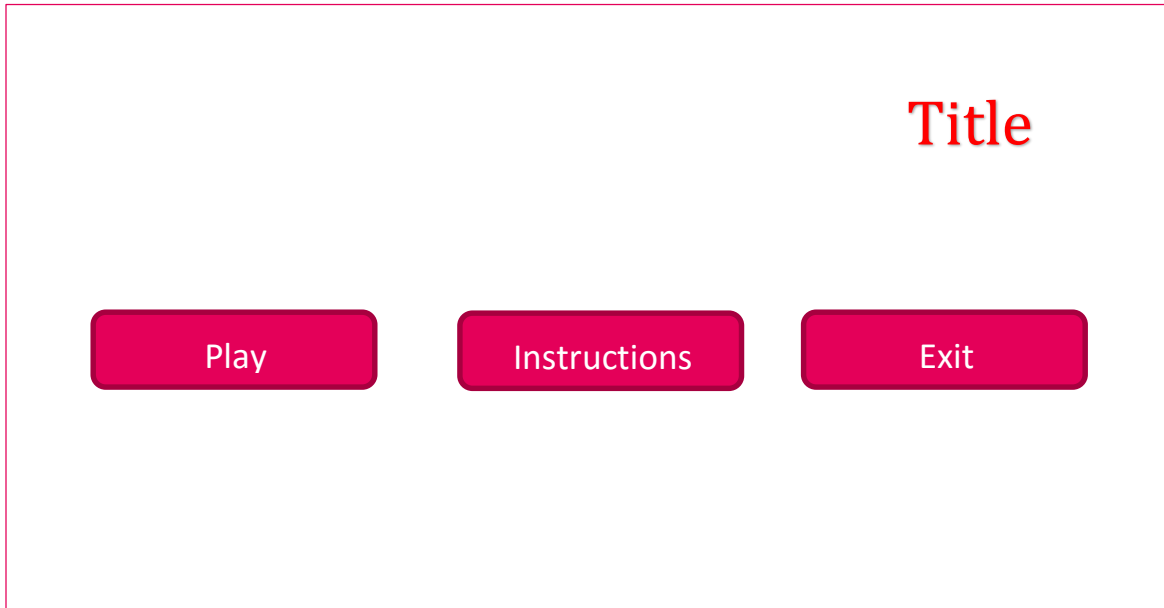
Task 6 - (P3.2)

Draw a sketch of two screens of your game:

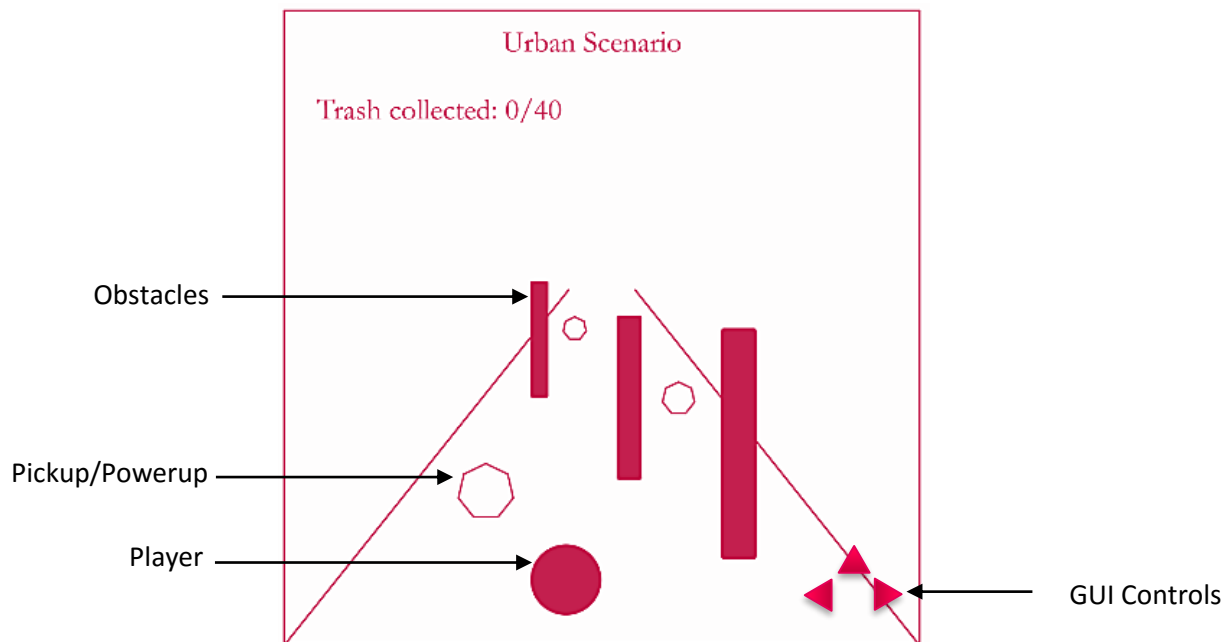
- Main menu
- Game screen

The sketches will show the positions of your GUI elements. The sketches must reflect the layout of the game.

Main Menu



Game Screen:



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.

Start()

This code is executed once as the scene loads. In the image below, there is an example of how this method can be used. When the game is loaded, the palm1Generator, canGenerator and chestGenerator are started as well. Since the prefabs are generated with a timer, I also generated them once in the start method so that when the game loads, the player doesn't have to wait a long time for the prefabs to generate.

```
void Start() {  
    palm1Generator();  
    canGenerator();  
    chestGenerator();  
}
```

Update()

The Update() method is executed once every frame. I used the update method to generate my prefabs. The timer is increased every frame since the method is run every frame. Once the timer reaches the spawnTimePalm, the palm prefab is generated.

```
void Update () {  
    timer++;  
    if (timer >= spawnTimePalm) {  
        palm1Generator ();  
        timer = 5;  
    }  
}
```

StartCoroutine("coroutinename")

StartCoroutine method calls an IEnumerator, that can have a yield function. Meaning that it can be paused for a specified amount of time. I used this function for my powerups, then I used StartCoroutine("") in the OnTriggerEnter method when the player hits a powerup.

```
IEnumerator Smallsize (){
    smallsize = true;
    if (smallsize== true) {
        transform.localScale += new Vector3 (-0.3F, -0.3F, -0.3F);
        yield return new WaitForSeconds (6);
        transform.localScale += new Vector3 (0.3F, 0.3F, 0.3F);
        smallsize = false;
    }
}
```

The code above shows the IEnumerator function in my code. This one in particular is for the small size powerup. The boolean is set to true once the code is initiated and the size of the player is minimized. After 6 seconds (hence yield wait for seconds), when the code is paused, the size is increased back to its normal state.

```
void OnTriggerEnter (Collider other){

    if (other.tag == "smallsize") {
        mini.SetTrigger("mini");
        Destroy (other.gameObject);
        StartCoroutine (Smallsize ());
    }
}
```

StartCoroutine is executed in the OnTriggerEnter when the player hits a prefab with the 'smallsize' tag as shown in the above image.

Task 8 - (P 4.2)

Main Menu:



Level 1:



Creating an Interactive Application

Level 2:

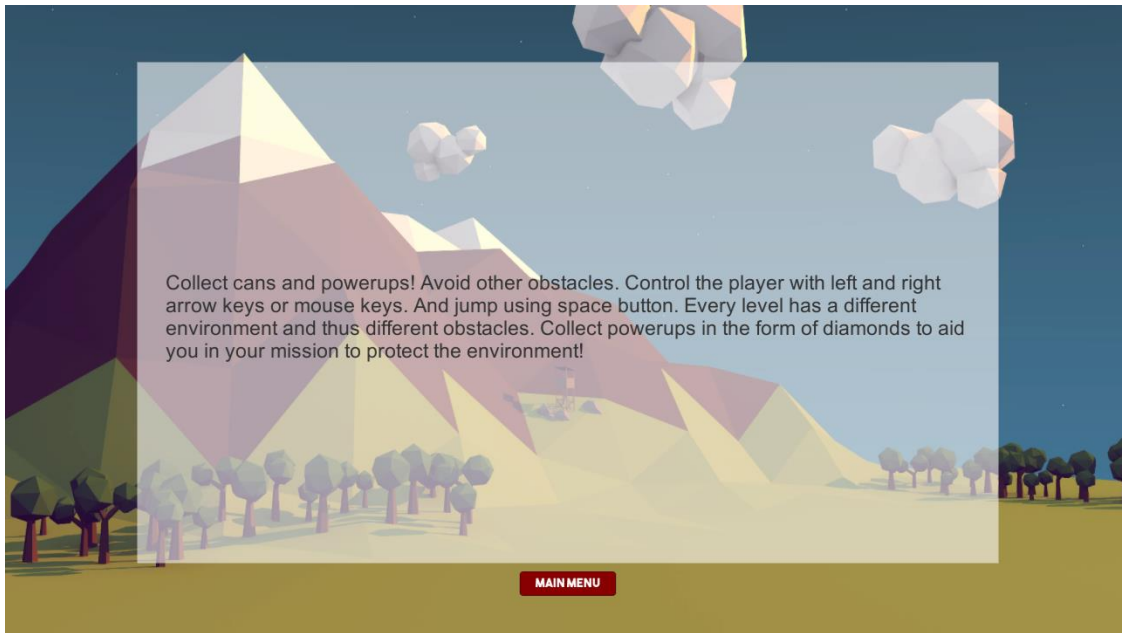


Level 3:



Creating an Interactive Application

Instructions screen:



Gameover (lost):



Gameover (win):



How to play

The purpose of the game is to collect trash (cans) so as to clean the environment. Use the arrow keys on the keyboard or the ones on screen to control the magnet. It can move left, right and jump. You can also collect modifiers: supersize, smallsize and superjump. The supersize is not very helpful as it increases the chance that you hit an obstacle and thus lose the game. Every level has an environment of its own. The first level takes you to the beach, the second to the city and the third to the countryside. In every level, there are different obstacles to avoid. In each level you have to collect a specific amount of cans to proceed to the next one. For the first level you have to collect 20 cans, for the second level you need to collect 40 and for the final level, 60.

Task 9 – (P 4.3)

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

There are a number of things that I would have liked to improve in my game. First of all are the 3d Models. In an ideal world, I would have implemented 3D models that I would have created myself using a 3D software like Maya or 3Ds Max. This would have helped me created the desired type of graphics and keep the same style throughout the game. To have a more realistic feel, having a floor generator would have helped. This would have improved the feeling that the player is moving forward rather than just everything moving towards the player. Something I would have liked to improve are the UI during the gameplay. For example, the score would have a box around it and there could have also been a pause and or exit button. A more in depth example is shown below.



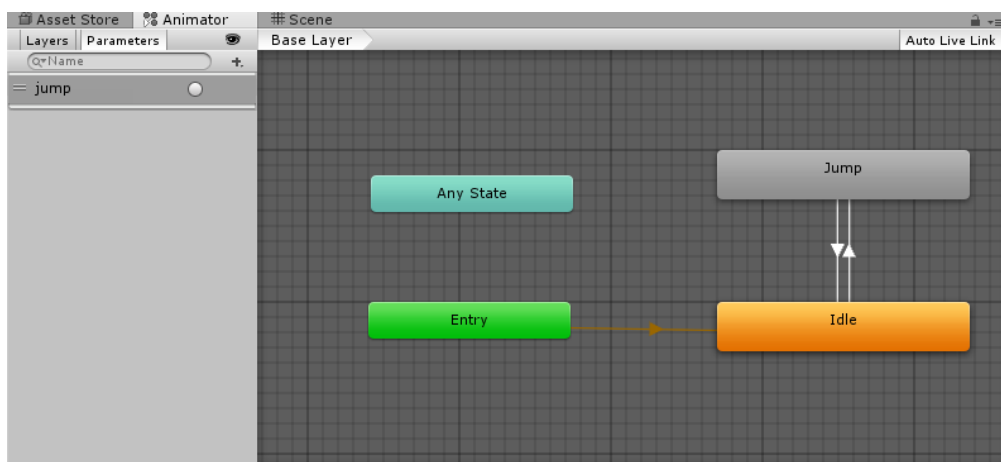
Task 10 – (M1.1)

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

3D models with animation in the game:

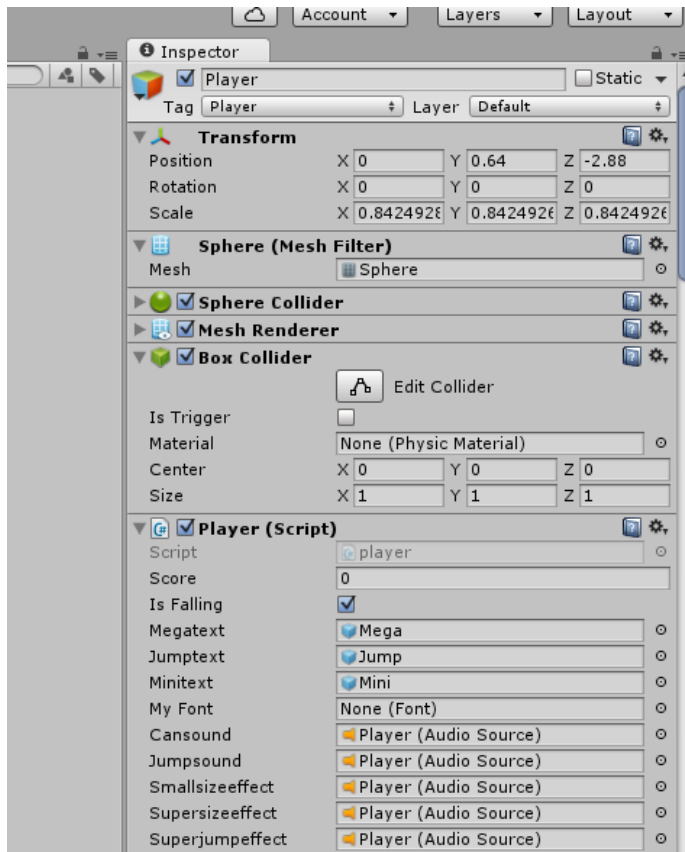
When the player collects a power up, a 3D model (text) is animated and moves towards the center of the screen from above. This is done for all 3 of the power ups using 3 different models.



Creating an Interactive Application



Full use of sound with at least 5 different sound effects:

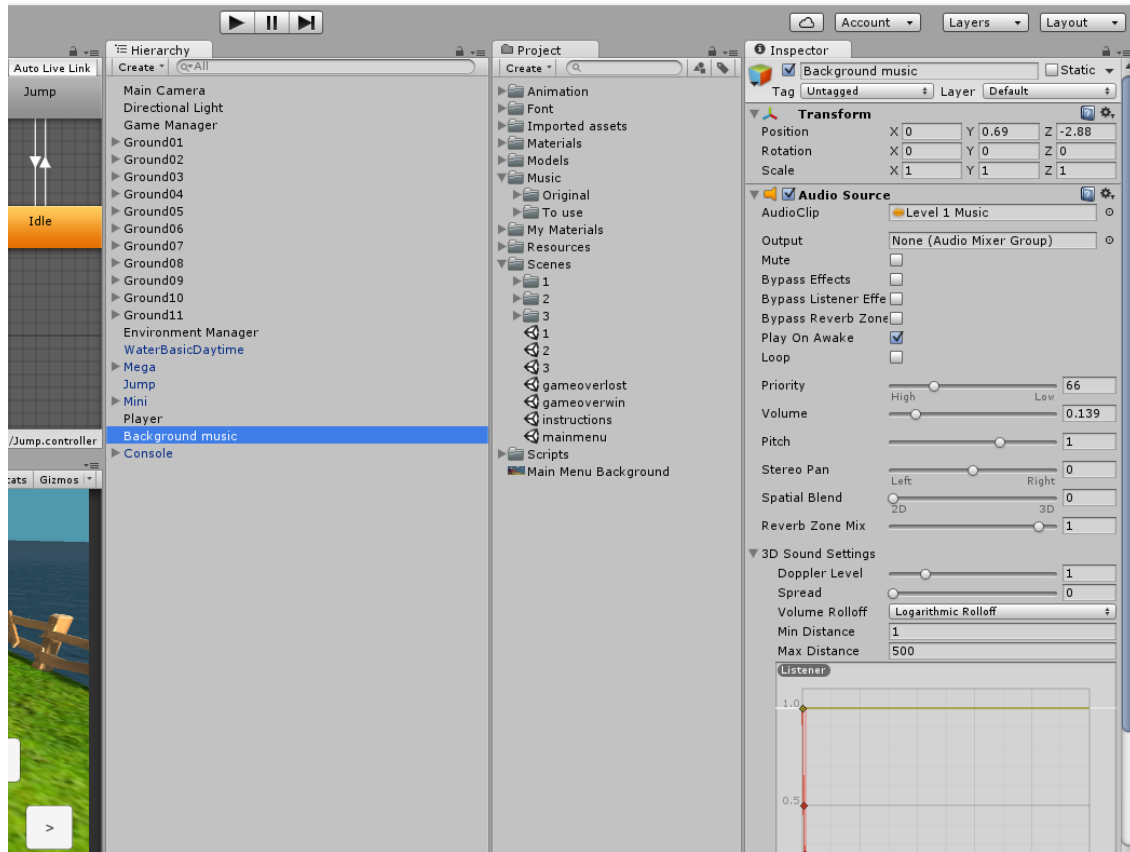


In my game, I have implemented 5 sound effects that happen on event. Three of them happen when the player picks up a power up, another one happens when the player collects a trashcan and another one happens when the player jumps.

Creating an Interactive Application

Background Music

I have implemented different background music throughout the game. The main menu and the levels have popular pop songs turned 8-bit as background music. They all have been implemented using an empty game object and adding an audio source to it. Then the Play on Awake option is ticked.



Task 12 – (M3.1)

Use the appropriate structure and approach to creating appropriate visual cues in interactive contexts by adding animated elements to the game.

Animated elements in my game can be found when the player gets a power up. The corresponding 3D model in the form of text, is animated when the player collides with the power up in the game. It moves downwards into the screen, pauses, and then goes back up. The animation is played once every time the player hits a power up and every power up has its own 3D model text.

How the animation was made: each model has its own animation and animator. The animation is done by trigger and doesn't loop. These were referenced in the player script and in the On Trigger Enter method, the trigger is set.

```
if (other.tag == "superjump") {  
    superjumpeffect.Play ();  
    jump.SetTrigger("jump");  
    Destroy (other.gameObject);  
    StartCoroutine (Superjump ());  
}
```

Line 1: If player hits object with the tag 'superjump' execute the following code.

Line 2: Play sound attached.

Line 3: Set animation trigger to true.

Line 4: Destroy game object.

Line 5: Execute Coroutine.

Task 12 – (M3.1)

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 I have created is a runner. This means that the player has a path, generally in front of him and he has to avoid incoming obstacles and collect items. This genre is quite popular among smartphone applications which is why I took multiple examples of games as inspiration. Mainly Minion Rush and Temple Run as they are my two personal favourites.



The images above show the popular game 'Minion Rush' during two different times of gameplay. One is set at the beach and the other is set in the town. Similarly, I was inspired by this and the following screenshots portray two levels of the game, one from the beach level (level 1) and the other from the city level (level 2).



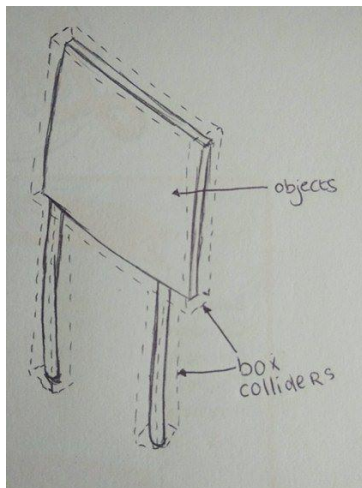
Creating an Interactive Application



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.

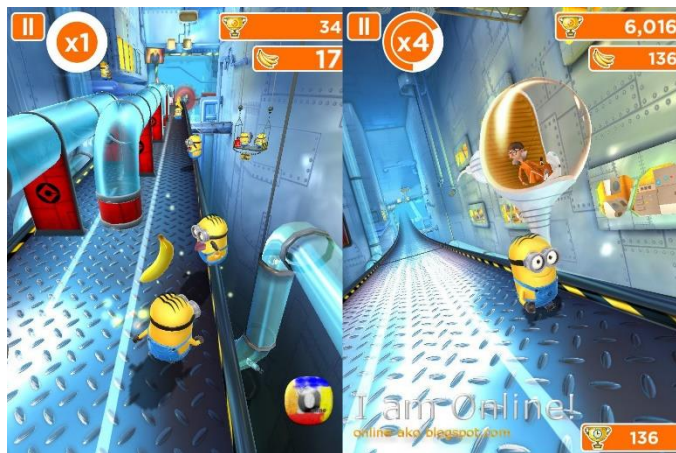
Both of the games mentioned in the previous task, have a very detailed storyline. This enabled them to create all the appropriate environments, appropriate obstacles, power ups and ultimately characters. Defining my story and making it more interesting would in turn help me create the appropriate 3D models to use, as well as add a character to the game. To do so, I have to use a 3D software, for example Maya. This would also give me the possibility to have everything of the same style too.



All of the environment, including the floor, is generated randomly. Both temple run and minion rush also enable the character to crouch to avoid high obstacles. To do this in my game, I would have to create another obstacle. Regarding this object's collider, I would probably have to create multiple box colliders and modify their shapes as shown in the image on the left.

Another major difference is the user interface during the gameplay. I couldn't help but noticing the perfect design used in Minion Rush. UI design would be involved to create such assets. This includes logo, colour scheme and user interface design.

The levels are generated randomly and the spawn point is very far away. This means that when the items are spawned, they don't just pop on the screen and they can't be seen. However, the major difference is the generation of the floors. It gives a better movement feeling. This could be created in a similar way as the environment and the obstacles were generated.



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.

<https://github.com/YanikaZerafa/CIDPAssignmentDistinction2016.git>




List of Commits:

Branch: **master** ▾

Commits on May 9, 2016

 Level 3 Environment spawn point ...	YanikaZerafa committed 2 minutes ago	 303f1d7	
 Main Menu game Title ...	YanikaZerafa committed 8 minutes ago	 8d53f7c	
 Fix in powerup spawntime ...	YanikaZerafa committed 14 minutes ago	 d92c21c	
 Game over Win Sound effect ...	YanikaZerafa committed 20 minutes ago	 7c6ed9e	
 Fixes: Level 3 Music, adding gameoverwin scene to build. ...	YanikaZerafa committed 29 minutes ago	 fc6b683	
 Canvas scales with screen size ...	YanikaZerafa committed an hour ago	 9a26345	
 GUI Text Position ...	YanikaZerafa committed an hour ago	 3b760ed	
 Change in Camera ...	YanikaZerafa committed an hour ago	 02e4e2b	

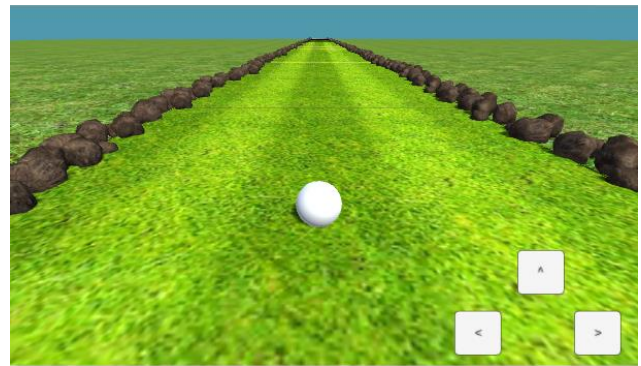
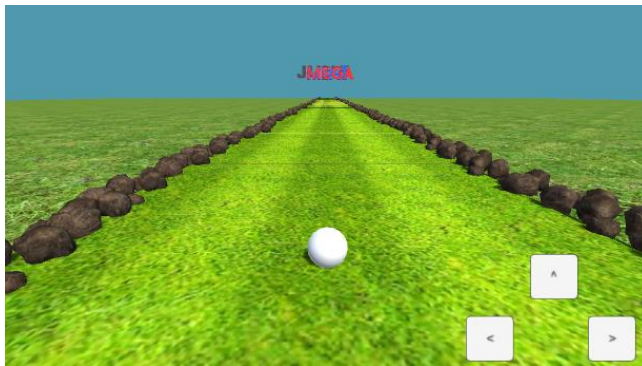
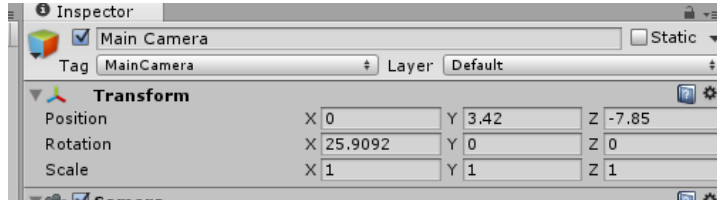
Commits on May 8, 2016

 Commit n. 1 ...	YanikaZerafa committed 16 hours ago	 c5c74e6	
--	-------------------------------------	---	---

Creating an Interactive Application

Commit 1: Change in Camera

The change done stands in the position of the camera. The camera was previously set to a good angle, however the 3D models for the power ups were showing even when they were not animated and they were not meant to. The position and rotation angle were slightly modified to the values shown in the image below.



Before and after images.

Commit 2: UI Score Text

The score text that appears in the top left part of the screen, was touching the border of the screen. Having some space would give it a nicer effect. To do so, I changed the co-ordinates of the code of the GUI.

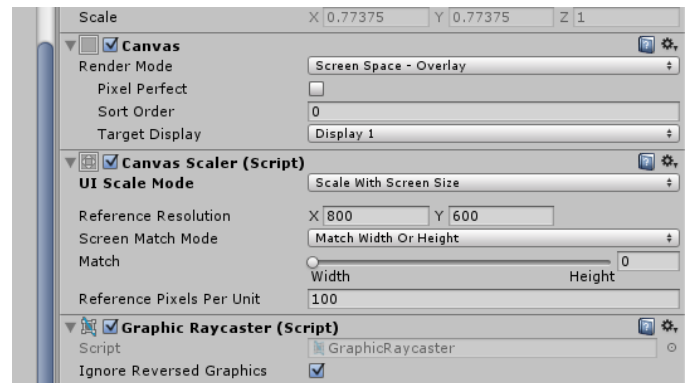
```
void OnGUI(){
    GUI.color = Color.black;
    GUI.Label (new Rect(0, 0, 100, 25), "Score: "+score, mystyle);
}

void OnGUI(){
    GUI.color = Color.black;
    GUI.Label (new Rect(50, 30, 100, 25), "Score: "+score, mystyle);
}
```

Creating an Interactive Application

Commit 3: UI Scaler

When I installed the apk file on my smartphone, the size of the elements of the UI was too small. I had to fix the Canvas Scaler and set it to Scale with screen size.

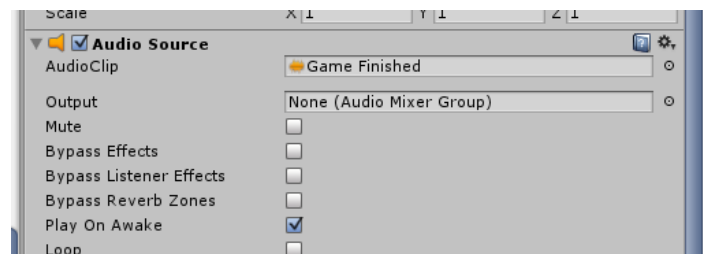


Commit 4: Background Music Level 3

The background music for level 3 was the same one as level two by mistake. The sound clip was edited and changed.

Commit 5: Win Scene sound

I had no sound in the Win Game over Scene. I added the sound, in a similar way as is in the Game over lost scene by adding an audio source that plays on awake.



Commit 6: Power up Spawn Time

The power up spawn time in every level was fixed. Previously, there were spawning at a time that I had given them randomly. To have the game make more sense, I fixed the spawn time of every level. The higher the level, the higher the spawn times.

Spawn Time Supersize	609
Spawn Time Minisize	550
Spawn Time Superjump	679

Creating an Interactive Application

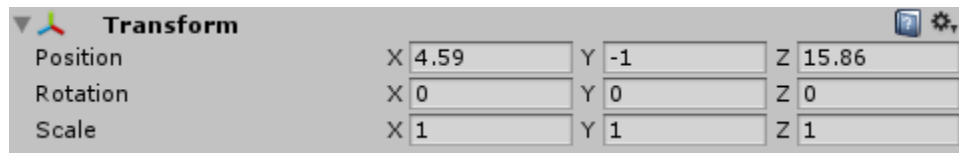
Commit 7: Game Title

The title for the game in the main menu was barely visible due to the color it had. Also it was enlarged for better understanding.



Commit 8: Environment Spawn position

The spawn points for the environment trees (not the game obstacles) in level 3 had the Y Axis too high, making the trees spawn in the air. Setting the spawn point to -1 made the trees spawn in the correct position.



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

Storyboard Design

The team members of this section would have the responsibility of creating the ideas and set the structure. They would determine what environments to have, what power ups, what obstacles and the story behind everything. This part is crucial as everything is taking a rough form upon which the rest of the teams can start building and perfecting at a later stage.

Visual Experience Designers

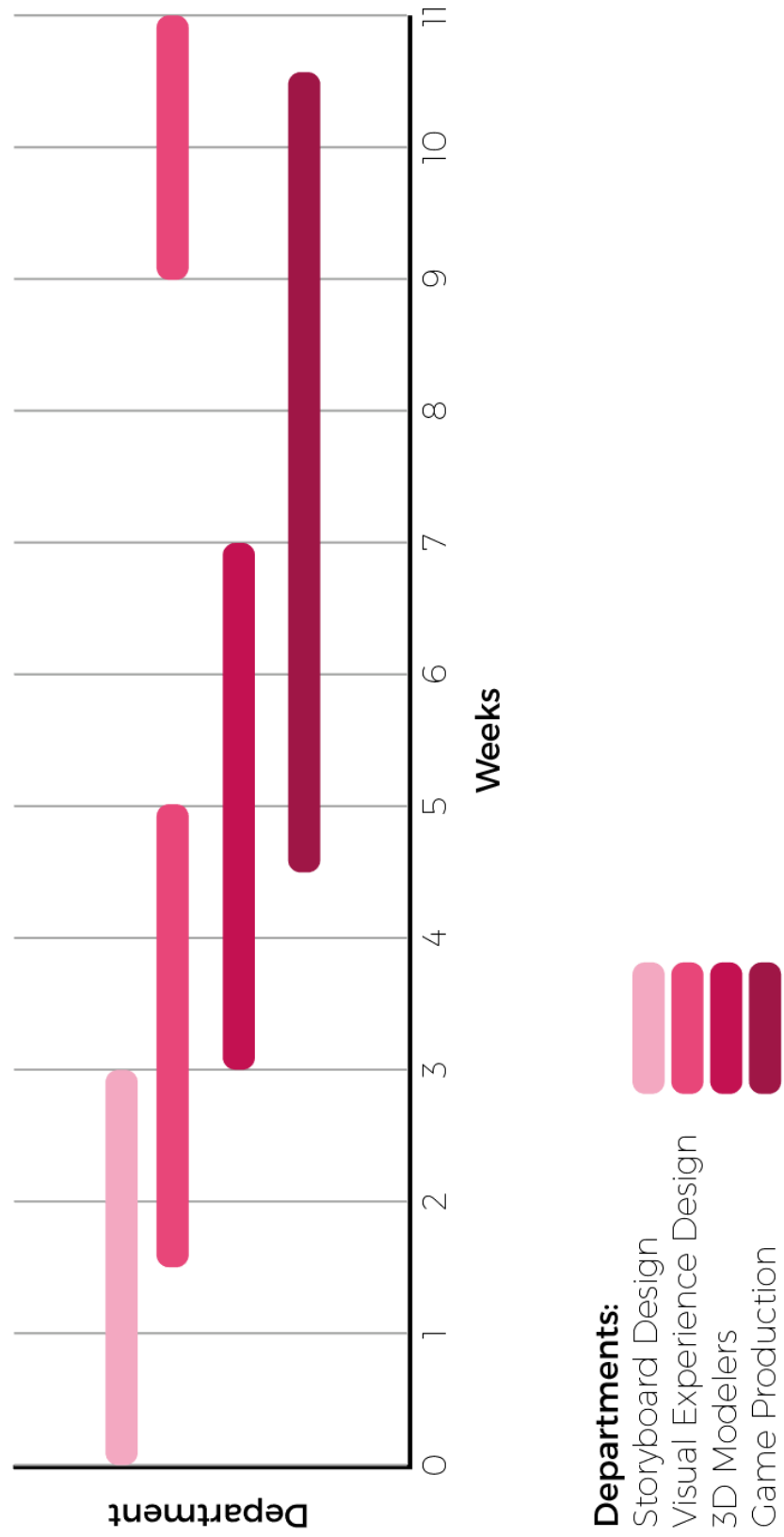
This team would be responsible of logo design, user interface design, colour schemes and the overall general style to use for the game. This is a long process and the team's work doesn't limit itself to the game only, but also to promotional items once the game is released. This team is basically the team that makes the game look aesthetically pleasing or appropriate.

3D Modelers

The role of the team members would be split into modeling, texturing and animation. They have to create 3D models through 3D software like Maya, according to the style chosen by the previously mentioned team (for example, low poly style). Once ready, the models are to be handed to the following team to implement them in the game.

Game Production

Game production means that they are responsible of coding and creating the game mechanics. They are in charge of gathering all the results produced by the three other teams into the game so that it takes shape according to what they all agreed.



Bibliography

Websites:

Digital-Tutors Blog. (2015). *Choosing the Right Game Engine | Unity, Source 2, Unreal Engine 4 or CryENGINE*. [online] Available at: <http://blog.digitaltutors.com/unity-udk-cryengine-game-engine-choose/> [Accessed 14 Apr. 2016].

reddit. (2011). *Comparison of free AAA game engines (CryENGINE, Source, UDK, Unity) • /r/gamedev*. [online] Available at: https://www.reddit.com/r/gamedev/comments/jmwf0/comparison_of_free_aaa_game_engines_cryengine [Accessed 14 Apr. 2016].

Gamesparks.com. (2016). *Game Engine Analysis and Comparison | Game Sparks*. [online] Available at: <http://www.gamesparks.com/blog/game-engine-analysis-and-comparison/> [Accessed 14 Apr. 2016].

Despicable Me Wiki. (2016). *Despicable Me: Minion Rush*. [online] Available at: http://despicableme.wikia.com/wiki/Despicable_Me:_Minion_Rush [Accessed 5 May 2016].

Wikipedia. (2016). *Temple Run*. [online] Available at: https://en.wikipedia.org/wiki/Temple_Run [Accessed 4 May 2016].

Wikipedia. (2016). *Platform game*. [online] Available at: https://en.wikipedia.org/wiki/Platform_game#Endless_running_games [Accessed 5 May 2016].

Answers.unity3d.com. (2016). *Screen to view point? - Unity Answers*. [online] Available at: <http://answers.unity3d.com/questions/45798/screen-to-view-point.html> [Accessed 3 May 2016].

Answers.unity3d.com. (2016). *Screen VS Viewport What is the difference - Unity Answers*. [online] Available at: <http://answers.unity3d.com/questions/168156/screen-vs-viewport-what-is-the-difference.html> [Accessed 10 May 2016].

Technologies, U. (2016). *Unity - Scripting API: Camera.ScreenToViewportPoint*. [online] Docs.unity3d.com. Available at: <http://docs.unity3d.com/ScriptReference/Camera.ScreenToViewportPoint.html> [Accessed 7 May 2016].