National College of Ireland
BSc in Computing
2015/2016

Ricardo O'Hara Camones
x12425828
x12425828@student.ncirl.ie

Blood Moon

Technical Report

Declaration Cover Sheet for Project Submission

SECTION 1 *Student to complete*

Name: **Ricardo O’Hara Camones**
Student ID: **X12425828**
Supervisor: **Anu Sahni**

SECTION 2 Confirmation of Authorship
The acceptance of your work is subject to your signature on the following declaration:
I confirm that I have read the College statement on plagiarism (summarised overleaf and printed in full in the Student Handbook) and that the work I have submitted for assessment is entirely my own work.

Signature: Ricardo O’Hara Camones
Date:10/5/2016

NB. If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the College’s Disciplinary Committee. Should the Committee be satisfied that plagiarism has occurred this is likely to lead to your failing the module and possibly to your being suspended or expelled from college.

Complete the sections above and attach it to the front of one of the copies of your assignment,

What constitutes plagiarism or cheating?
The following is extracted from the college's formal statement on plagiarism as quoted in the Student Handbooks. References to "assignments" should be taken to include any piece of work submitted for assessment.

Paraphrasing refers to taking the ideas, words or work of another, putting it into your own words and crediting the source. This is acceptable academic practice provided you ensure that credit is given to the author. Plagiarism refers to copying the ideas and work of another and misrepresenting it as your own. This is completely unacceptable and is prohibited in all academic institutions. It is a serious offence and may result in a fail grade and/or disciplinary action. All sources that you use in your writing must be acknowledged and included in the reference or bibliography section. If a particular piece of writing proves difficult to paraphrase, or you want to include it in its original form, it must be enclosed in quotation marks
and credit given to the author.

When referring to the work of another author within the text of your project you must give the author's surname and the date the work was published. Full details for each source must then be given in the bibliography at the end of the project

Penalties for Plagiarism
If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the college's Disciplinary Committee. Where the Disciplinary Committee makes a finding that there has been plagiarism, the Disciplinary Committee may recommend

- that a student's marks shall be reduced
- that the student be deemed not to have passed the assignment
- that other forms of assessment undertaken in that academic year by the same student be declared void
- that other examinations sat by the same student at the same sitting be declared void

Further penalties are also possible including

- suspending a student college for a specified time,
- expelling a student from college,
- prohibiting a student from sitting any examination or assessment.,
- the imposition of a fine and
- the requirement that a student to attend additional or other lectures or courses or undertake additional academic work.

Table of Content

Executive Summary

Blood Moon is made with Unity 5 and is a challenging 2D action platforming game, with adventure and light RPG (Role Playing Game) elements set in a pixel Gothic world, inspired by classic games such as Castlevania, along with modern games such as Dark Souls and Bloodborne.

The game is following the player character, “The Druid”, where you’ll explore a dark fantasy world scattered with varying types of creatures created with Photoshop CC and Aseprite. These enemies will have their own AI using unique behaviour trees with C# to perform attacks and defences to add variety and challenge to gameplay. The player will increase their health, magic and damage levels by gaining XP (experience points) to level up through tasks such as collecting rare items, and defeating enemies. This is essential to progress further to face tougher enemies and bosses, along with gaining access to new areas.

The combat involves using weapons (blades), blocking (shield) and abilities (magic) along with “The Druid’s” mobility, such as double jumping to manoeuvre around the world and enemies. The goal is to make the game more about strategic, thought-out play, but still rely on perfect twitch reflex or muscle memory from retro games.

1. Introduction

1.1. Background

1.1.1. Genre

Studying various genres my research has lead me to designing a 2D platforming game due to its ease of entry level for the consumer and its resurgence in the gaming industry by independent (indie) developers and to the ease of access achieved by the downloadable marketplace. Along with the knowledge that it doesn't take a team of hundreds to create a successful game, with freely and easily accessible tools such as Unity, anyone with a passion and desire to be creative can be successful in creating a game. A traditional platforming game involves guiding a player character to jump between suspended platforms, floors, ledges, stairs over obstacles or a mixture to advance in the game, these challenges are commonly referred to as jumping puzzles.

1.1.2. Gameplay

The new elements that will be added to make Blood Moon standout from traditional platforming games that I have seen not commonly done in my research of the marketplace, is an RPG levelling system that will increase health and magic set boundaries. This will make the player more invested in the game and they have their own personal progression which will also be essential for defeating later enemies, bosses and accessing new areas.

The movement of the character will be achieved in various ways such as jumping, double jumping, climbing but will also require the player to use these for combat as well and attacking at exact moments to reach areas that would not be achieved by simple and traditional platforming movements.

1.1.3. Art

Due to the constraints and resources that come with being a solo or a small time developer, many focus on a unique art style. The reasons being to save time and money on developing high end graphics and also increase financial return on the project. It's easy to underestimate the importance of artwork to a game and even easier to underestimate the importance of having a unique style of artwork. The result is that there are many generic-looking games fail to capture people's attention. Best examples I found of games using a unique art style and being successful in the marketplace are, Minecraft, which uses simple blocky pixel art, Apotheon, using Greek pottery art as its inspiration and most famously Braid, using "painterly art" and also seen as a keystone title in the growth of indie game development. Even big studios see the value in this and make small games occasionally, as it's considered low risk, high reward, an examples of this is Ubisoft's Valiant Hearts, a 2D puzzle adventure and Insomniac Game's recently announced Song of the Deep, a 2D action-adventure. Hence why my research led me to decide on not simply buying and downloading generic 3D models or 2D assets and set myself the challenge of creating all the unique assets and animations myself seen in Blood Moon, reflecting my imagination and innovation, creating this world and characters that'll bring life to it. This ultimately would adhere to people's nostalgia of old games that have inspired my artwork and will not limit the platforms that I can release my game on. This also adds to the commercial value of the project, as licensing and royalty fees of using assets would detract from the financial return.

This decision will also not deter from the game itself, as Unity creates the game in a 3D environment and simply adjusts the camera to be of a 2D perspective. So the C# scripts I have made are as complicated as a 3D game and can mostly be used for such a 3D game, with minimal adjustments to certain scripts.

1.1.4. Story

The story follows the player character called "The Druid" who is a hunter of beasts and legends. The Druid uses metal and magic to combat these enemies to keep the world safe. Unfortunately, a blood man eclipse has begun, giving increase power to users of blood magic such as The Druid and his enemies. The Druid must defend the cities while the blood moon engulfs the world and darkness reigns.

1.2. Aims

The aim of this project is to create a fully functional and commercially viable 2D platforming game aimed primarily for Windows or Mac OS X but viable for various other platforms such as PlayStation, Android, iOS etc.

The game should look unique compared to other games by using Photoshop and Aseprite for designing the artwork and animating characters respectively made by myself and not purchased from the Unity Asset Store or Unreal Marketplace.

The game will be a mixture of two game genres, 2D platformer and RPG which an uncommon mixture seen in the gaming industry to make my game more unique. The RPG elements should encourage the player to invest time in the game by levelling their character up and be able to approach more difficult challenges and access new areas.

The story of the game will be told through environmental aspects as to not distract from gameplay. This way if there is a story the player wants to know they'll see it through the world and not affect players who do not care for stories in games.

1.3. Technologies

1.3.1. Unity

Is a cross-platform game engine used to develop video games for PC, consoles, mobile devices and websites. Unity is used commonly in video game development in many organisations. Unity will be my primary tool in development of my project as many jobs prefer experience with this game engine over other freely available game engines. I am currently using the latest version Unity 5.2.1.

1.3.2. MonoDevelop (cross platform IDE)

Is an open source multi-platform IDE (Integrated Development Environment) used in Mac OS X, Windows and Linux. Similar to other IDEs, such as Netbeans or Microsoft Visual Studio having features such as auto complete code, a GUI and support a wide variety of programming languages, such as C#.

I choose MonoDevelop over other IDEs as a customised version of MonoDevelop ships with Unity and runs on Mac OS X, my primary development environment.

1.3.3. C#

The standard language that is used in the creation of games developed with Unity. C# has been relatively easy to learn thus far. It is very similar to other object-oriented programming languages such as Java. It only took a short amount of time to become familiar with C#. This represents a primary reason why Unity was chosen for the project over Unreal Engine which commonly uses Blueprint Visual Scripting, which is a node based interface.

1.3.4. InControl

InControl is an input manager for Unity3D that standardizes input mappings across platforms for common gamepad controllers. It is written in C# and strives to make it easy to add cross-platform controller support to your game. While Unity does have its own input manager but due to my development environment this was rendered redundant as Mac OS X does not support PlayStation and Xbox controllers without external inputs. This was the best option available and is used professionally by several successful game developers.

1.3.5. Adobe Photoshop CC

An easy to use imaging and design editor developed by Adobe Systems for Windows and OS X that'll assist me in creating the assets for my project such as characters, environment etc. to achieve the look and style I want.

1.3.6. Aseprite

An open source program to create animated sprites and pixel art. Sprites are little images that can be used in a game or a website. An essential piece of software in creating animations from the designs created in Photoshop.

2. System

2.1. Requirements

2.1.1. Functional requirements

New Game

The player should be able to select the New Game option and deleting all previously stored data, such as character level. This requirement has not changed from the original requirements specification document.

Save Game

The player should be able to save their game at several save points throughout the game. This requirement changed from the original requirements specification document, previously it was the player who controlled the save game but now it’s automatically saved at the save points. This was due to my research leading me to see that this is a more common, functional, and practical for the player.

Continue Game

The player should be able to continue on with their previously saved data. This requirement has changed from the original requirements specification document due to changing the layout.

Pause Game

The player should be able to pause the game at any moment of gameplay and access the control scheme or exit back to the main menu. This requirement has not changed from the original requirements specification document.

Quit Game

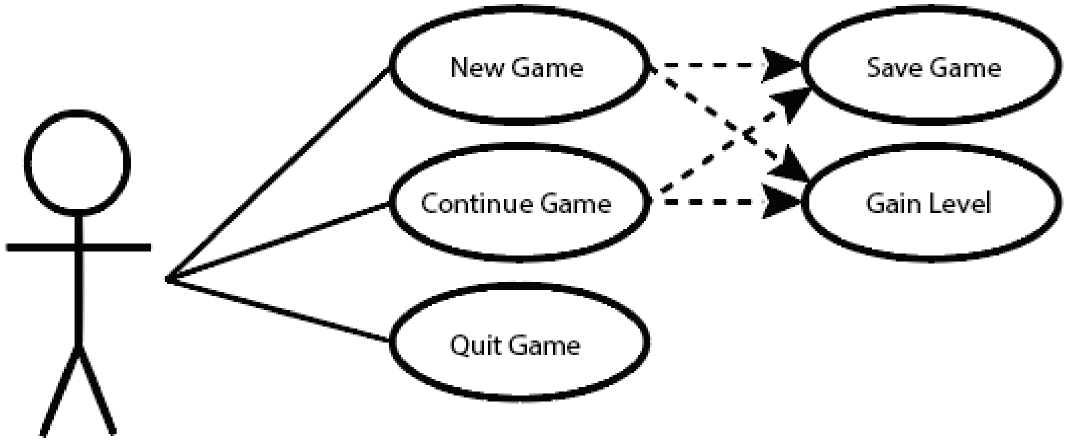
The player should be able to exit the game and close the application while returning to their desktop environment. This requirement has not changed from the original requirements specification document.

Gain Level

This requirement enables the player to gain a level. Levels are gained through the accumulation of enough experience points (XP). XP is awarded for the completion of objectives within the game, killing enemies and collecting rare items hidden within each level. This requirement has not changed from the original requirements specification document.

2.1.2. Use Case Diagram

The Use Case Diagram provides an overview of all functional requirements.



2.1.3. Requirement 1 <New Game>

Description & Priority

The player should be able to select New Game from the main menu if they're new to the game or wish to start the game over again. This requirement is essential in getting the player to start the game.

Use Case

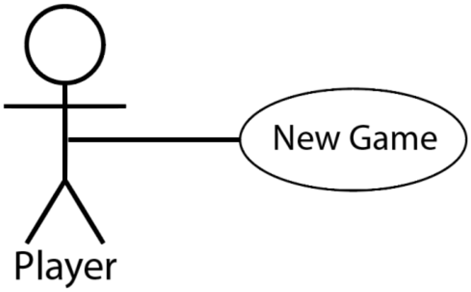
Scope

The scope of this use case is to allow the player to start a new game.

Description

This use case describes the process by which a player starts a new game.

Use Case Diagram



Flow Description

Precondition

The system is in initialisation mode.

Activation

This use case starts when an player starts a new game.

Main flow

1. The system identifies the player.
2. The system creates a save file.
3. The player starts a new game.
4. The system loads up the first level.

Termination

The system will bring the next scene to the user.

Post condition

The system goes into a wait state.

2.1.3. Requirement 2 <Save Game>

Description & Priority

The player saves their game at certain save points in the game. This requirement is important to give incentive to the player to invest in the game and continue on from where they finished.

Use Case

Scope

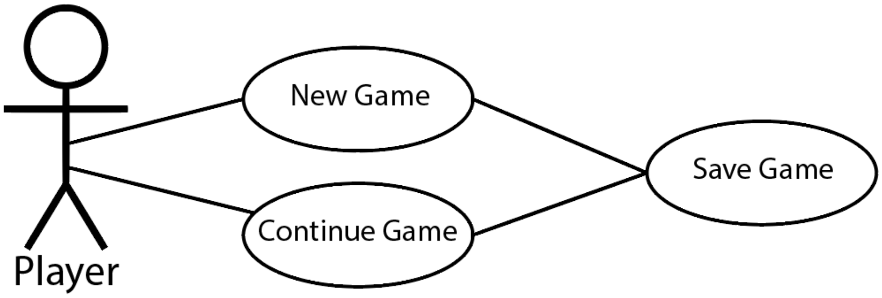
The scope of this use case is to allow the player to save the game.

Description

This use case describes the the process by which a player saves the game.

Use Case Diagram

Flow Description



Precondition

The system is in initialisation mode.

Activation

This use case starts when an player is at a save point.

Main flow

1. The system identifies the player at save point.
2. The player saves the game.
3. The system stores the saved game data.

Termination

The system presents the previous scene to the user.

Post condition

The system goes in to a wait state.

2.1.3. Requirement 3 <Continue Game>

Description & Priority

The player should be able to select Continue Game from the main menu. This requirement is essential in getting the player to continue on from their progress made in previous game sessions.

Use Case

Scope

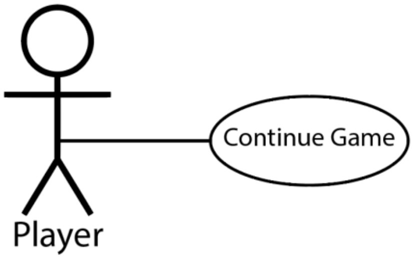
The scope of this use case is to is to allow the player to continue their game.

Description

This use case describes the process by which a player continues a game from their stored saved game data.

Use Case Diagram

Flow Description



Precondition

The player has played the game before and has a saved file.

Activation

This use case starts when the system has stored saved game data.

Main flow

1. The player selects Continue Game
2. The system will load stored game data.
3. The system will spawn character from saved location.

Termination

The system presents the loaded scene.

Post condition

The system goes into a wait state.

2.1.3. Requirement 4 <Gain Level>

Description & Priority

The player should be able to gain experience points and level up to increase max health and magic limits.

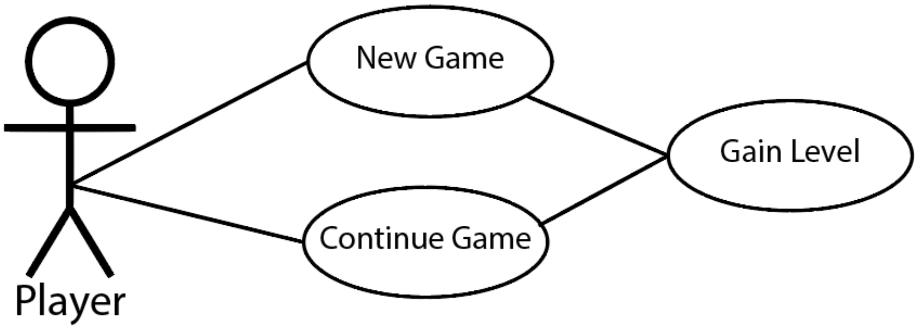
Use Case

Scope

The scope of this use case is to allow the player to increase level of their character.

Description

This use case describes the process by which a player gains a level.



Use Case Diagram

Flow Description

Precondition

The system is in initialisation mode.

Activation

This use case starts when the player starts a new game or continues game from a previous save.

Main flow

1. The player attacks enemies or collects rare items over time.
2. After the limit is reached for the level the player upgrades.

Termination

n/a

Post condition

The system goes into a wait state.

2.1.3. Requirement 5 <Quit Game>

Description & Priority

The player quits the game at any point but is warned that any unsaved data will be lost to confirm the player wants to quit game. This requirement is essential in ending a game session.

Use Case

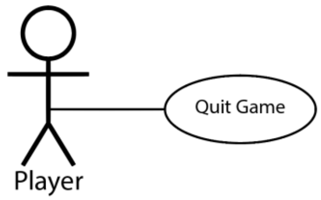
Scope

The scope of this use case is to quit the game.

Description

This use case describes the process by which the player can exit the game.

Use Case Diagram



Flow Description

Precondition

The system is in initialisation mode.

Activation

This use case starts when the player wants to quit the game.

Main flow

1. The player selects quit game.<A1>
2. The player pauses the game.
3. The player quits the game.
4. The system exits the game.

Alternative flow

1. The player starts the game.
2. The player pauses the game.
3. The player quits the game.
4. The system exits the game.

Termination

The system closes the game.

Post condition

The system is off.

2.1.4. Data requirements

The player should have their progress in each level saved by a checkpoint. These are activated by walking past them and are easily visible as campfires. This is where the player will respawn when killed by enemies or by environmental hazards. This information will be lost if the level is not completed in the same sitting. This was to add challenge and encourage the player to complete the level.

The game will auto save and store the information to a predefined folder permanently at the end of each level, this is to encourage more of a challenge to the player and an assentive to complete the level. This is a feature seen in older games but I have made the adjustment for today's current generation of gamers used to less challenging games with checkpoints in place and infinite lives as oppose to no checkpoint and a small number of lives, usually three. The levels have been adjusted so they are smaller so if not completed in a sitting the user will not feel annoyance towards the game for having to begin the level from the start in the next session. The player will need a small amount of free space on their internal hard drive to store this saved data associated with the game.

This is all achieved through Unity's PlayerPrefs script which stores and accesses the saved data between play sessions. The predefined folder that stores this data differs on each platform. On Mac OS X PlayerPrefs are stored in ~/Library/Preferences folder, in a file named unity.[company name].[product name].plist,. On Windows, PlayerPrefs are stored in the registry under HKCU\Software\[company name]\[product name] key. The company and product names are the names set up in Project Settings for both platforms.

2.1.5. User requirements

After the game is complete it’ll be built in Unity and be able to play without the use of Unity as is currently needed to play the game. The user will require a computer running either Windows OS or Mac OS X to run the software.

2.1.6. Environmental requirements

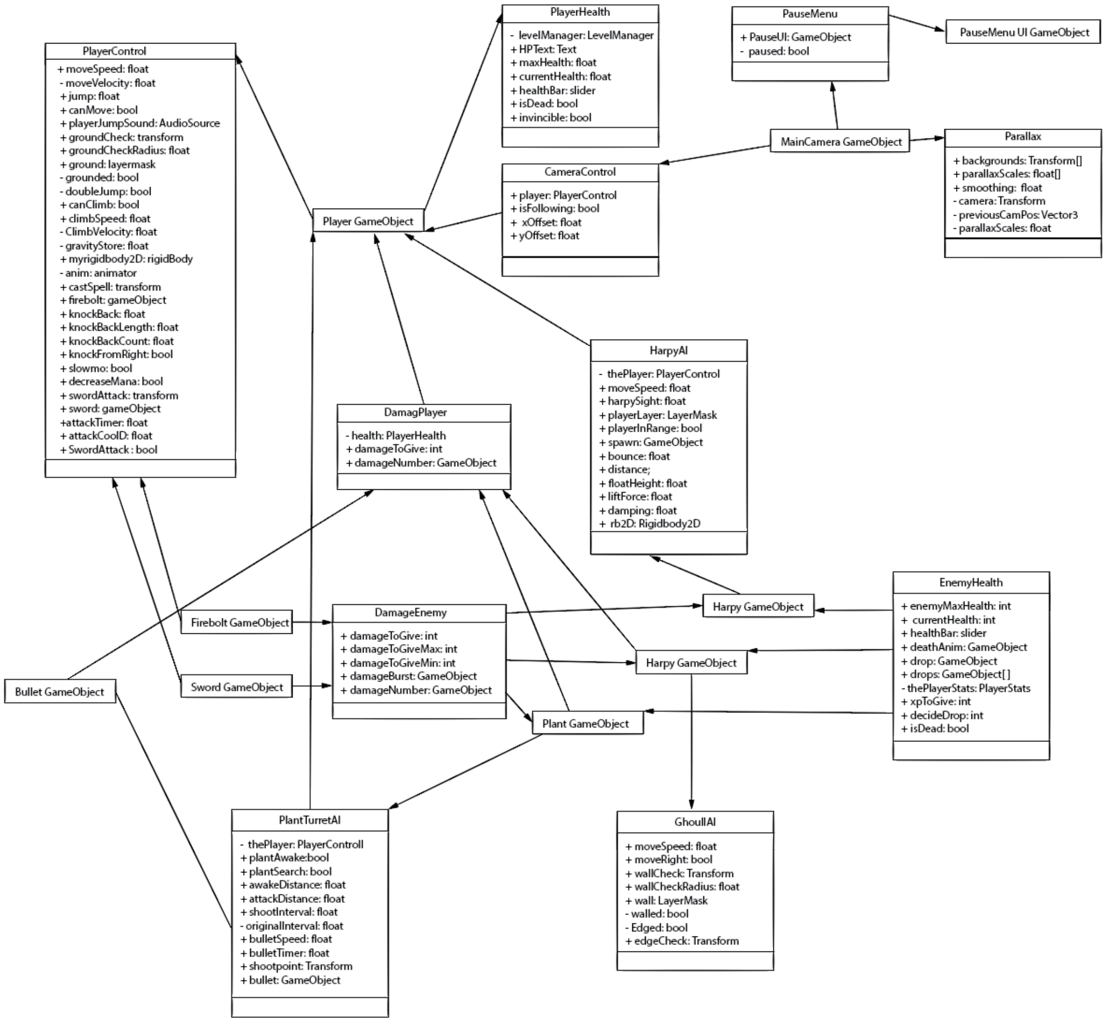
The game must run in a stable working environment for maximum usability with the basic computer hardware such as a keyboard, monitor and mouse. The game currently runs flawlessly on Mac OS X and Windows with rapid response and loading times. Performance on other platforms such as PlayStation and Xbox could not be performed but suspect to react the same as Unity is the lead in multiplatform development.

2.1.7. Usability requirements

The game’s menus should be easily understandable and simple to navigate to accomplish the intended task such as beginning new game or quitting the application.

2.2. *Design and Architecture*

2.2.1. Class Diagram



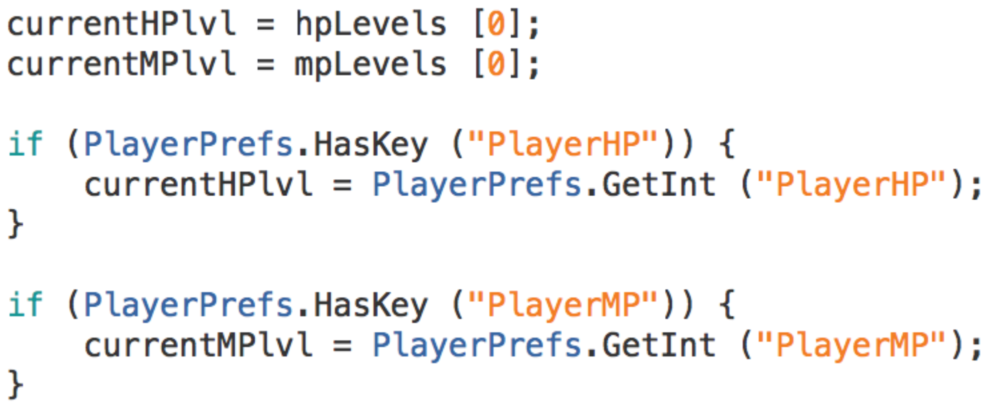
2.3. *Implementation*

The scripts mentioned below are not all the scripts involved and created for the project but are the most important and complicated ones.

2.3.1. PlayerStats.cs

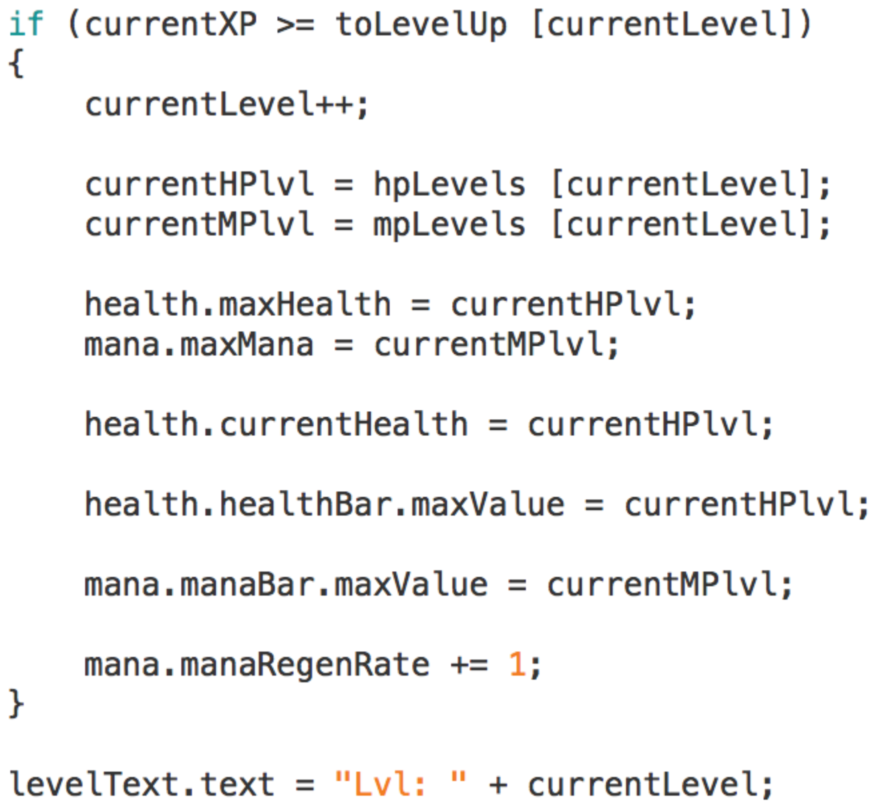
This script is responsible for out Gain Level requirement and controls our leveling up our player character throughout the game and adjusting predefined settings in other scripts such as maxHealth and maxMana in the PlayerHealth and PlayerMagic scripts respectively to better suit the level of the player, i.e. the increase to your level increases the max values of health and magic your player can have.

We use arrays to achieve this, setting how much XP the player has currently with and how much more he will need to increase their current level, and then what the respective maxHealth and maxMana is now and also increasing their health to this new level. We start by making sure the player has no previously stored save data using PlayerPrefs and if so we match his health and magic to these levels. This is an extra precaution as this is primarily done in the levelManager script but with extensive testing found that this would not always work. If the player does not have any stored data, we set the values to first int in the array.



```
currentHPlvl = hpLevels [0];
currentMPlvl = mpLevels [0];

if (PlayerPrefs.HasKey ("PlayerHP")) {
    currentHPlvl = PlayerPrefs.GetInt ("PlayerHP");
}

if (PlayerPrefs.HasKey ("PlayerMP")) {
    currentMPlvl = PlayerPrefs.GetInt ("PlayerMP");
}
```

Using an if statement in the void Update function to constantly check that the currentXP that the player has is a larger value than the XP needed for each level defined in the toLevelUp array is met or surpassed which will then begin the process of leveling up the player. So currentLevel value is increased by one so it knows not to repeat this statement and the next amount of XP needed to level up is put in place. After this we update the levels of the maxHealth and maxMana along with updating the UI so the health/mana bars know what their new max values are and the text of the player’s current level is updated. While doing this we make sure to increase the currentHealth of the player is set to the max incase leveling up is happening during combat and the health boost might increase the chances of victory. Finally, we increase the player’s magic regeneration rate by 1.

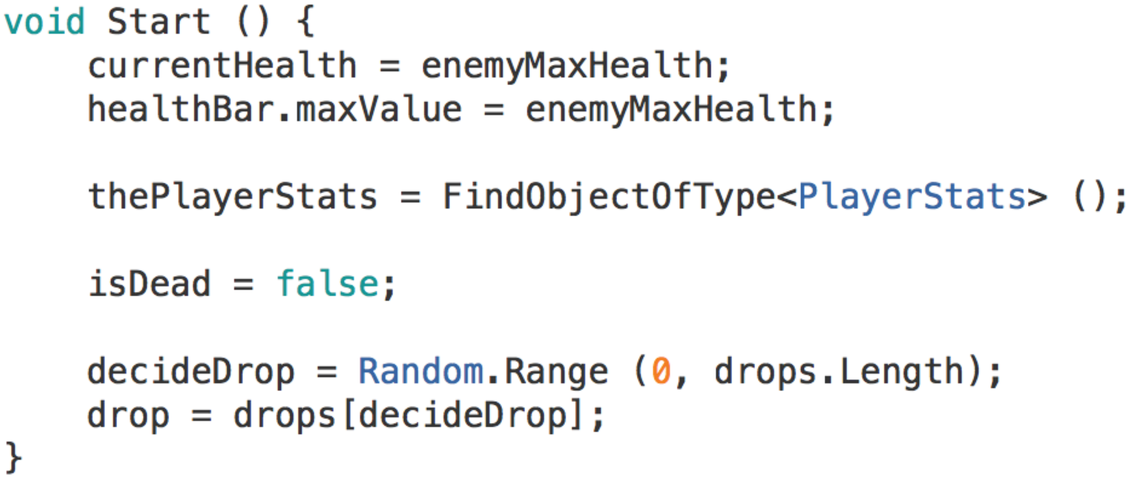


```
if (currentXP >= toLevelUp [currentLevel])
{
    currentLevel++;

    currentHPlvl = hpLevels [currentLevel];
    currentMPlvl = mpLevels [currentLevel];

    health.maxHealth = currentHPlvl;
    mana.maxMana = currentMPlvl;

    health.currentHealth = currentHPlvl;

    health.healthBar.maxValue = currentHPlvl;

    mana.manaBar.maxValue = currentMPlvl;

    mana.manaRegenRate += 1;
}

levelText.text = "Lvl: " + currentLevel;
```

2.3.2. EnemyHealth.cs and PlayerHealth.cs

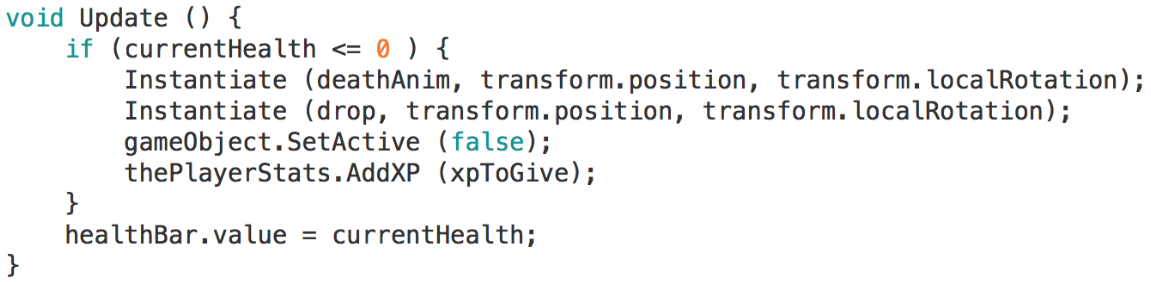
EnemyHealth defines the the enemy health, the XP to give to the player on death, and choosing the item for the player to drop on death, while having the responsibility of representing the enemy's health with an enemy health bar using a slider. It was created separately from the enemy AI scripts to be more efficient and prevent repetition of code. Having public variables such as enemyMaxHealth and xpToGive so they can be set within Unity and differ on each enemy.

We start with getting the currentHealth to equal the enemyMaxHealth which we have set in Unity and then we set our healthBar.maxValue to equal enemyMaxHealth so the UI EnemyHealth is represented to the player correctly. We also decide on the item that'll drop for the player as soon as the game starts to prevent any issues of the item not being chosen before the enemy dies. We use an array to store all the possible items called drops and then we use decideDrop to go through the array and choose a random item. Finally, the drop GameObject is chosen.



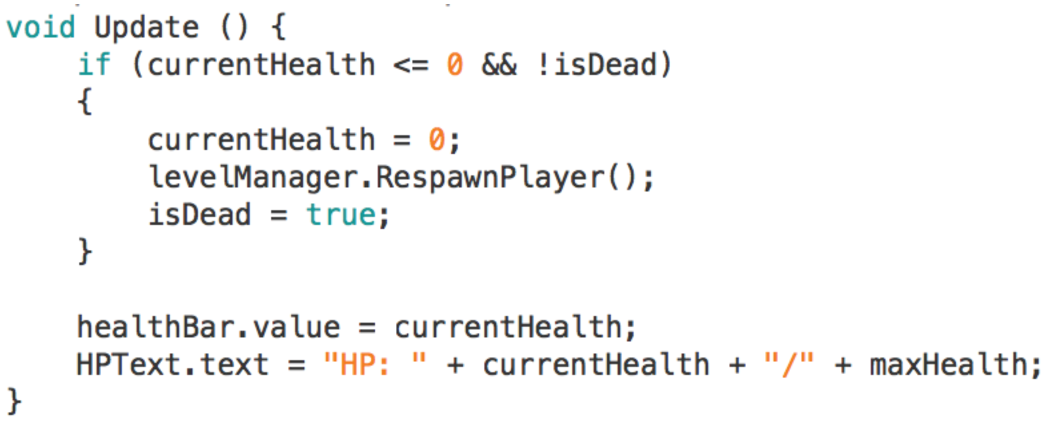
```
void Start () {
    currentHealth = enemyMaxHealth;
    healthBar.maxValue = enemyMaxHealth;

    thePlayerStats = FindObjectOfType<PlayerStats> ();

    isDead = false;

    decideDrop = Random.Range (0, drops.Length);
    drop = drops[decideDrop];
}
```

We constantly check to update the health bar value to the current health of enemy and to see if the enemy health is below zero, in which case we use Insatiate to create the death animation using practical effects on the enemy's last position and also drop loot for the player to pick up. The enemy is set to false but not destroyed as destroying the gameObject would remove it from the game world and not possible to respawn on player death. Finally, the player receives the XP to the PlayerStats script.



```
void Update () {
    if (currentHealth <= 0 ) {
        Instantiate (deathAnim, transform.position, transform.localRotation);
        Instantiate (drop, transform.position, transform.localRotation);
        gameObject.SetActive (false);
        thePlayerStats.AddXP (xpToGive);
    }
    healthBar.value = currentHealth;
}
```

PlayerHealth fundamentally works the same as EnemyHealth having a currentHealth value and a maxHealth value. These function the exact same and are represented with healthBar in the top left corner of the screen. The main difference here is that over the bar there is UI text element that constantly updates to visually to give a more detailed representation of the currentHealth and maxHealth.

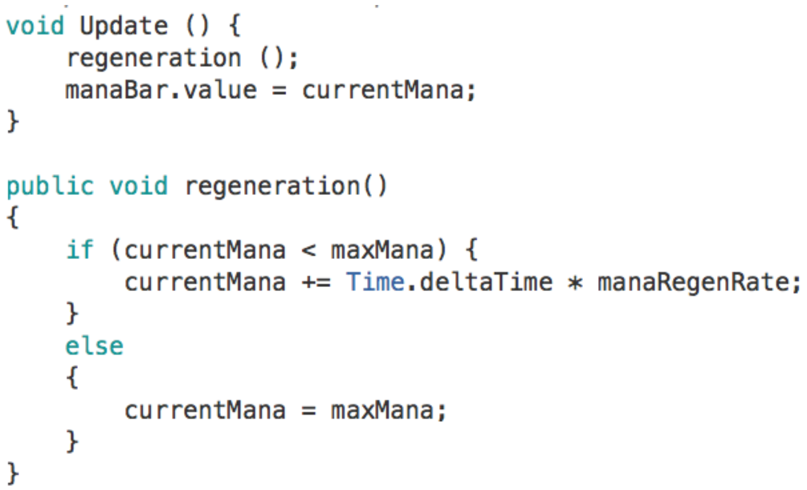


```
void Update () {
    if (currentHealth <= 0 && !isDead)
    {
        currentHealth = 0;
        levelManager.RespawnPlayer();
        isDead = true;
    }

    healthBar.value = currentHealth;
    HPText.text = "HP: " + currentHealth + "/" + maxHealth;
}
```

The playerHealth does not have values for XP to give or items to drop on death for enemies to pick up.

2.3.3. PlayerMagic.cs

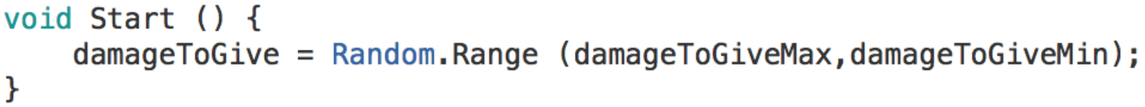
This script functions the same as the PlayerHealth but for the player’s magic. The key difference is that the magic has a regeneration rate which is controlled by a float variable called manaRegenRate which is then combined with Time.deltaTime to regenerate over time.



```
void Update () {
    regeneration ();
    manaBar.value = currentMana;
}

public void regeneration()
{
    if (currentMana < maxMana) {
        currentMana += Time.deltaTime * manaRegenRate;
    }
    else
    {
        currentMana = maxMana;
    }
}
```

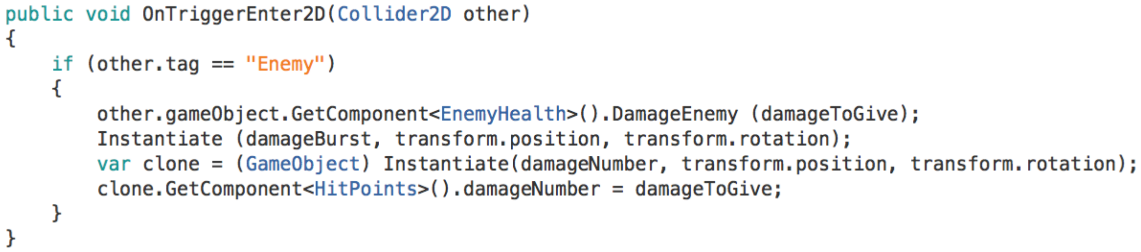
2.3.4. DamageEnemy.cs and DamagePlayer.cs

DamageEnemy defines the damage the player can do for different offenses, such as fire bolt and sword attack. We have two int variables that decide the damage to do to the enemy, damageToGiveMin and damageToGiveMax. So to decide the damageToGive to the enemy we use a random range so damage is always between these two variables being different each time, a feature seen commonly in RPG games.



```
void Start () {
    damageToGive = Random.Range (damageToGiveMax,damageToGiveMin);
}
```

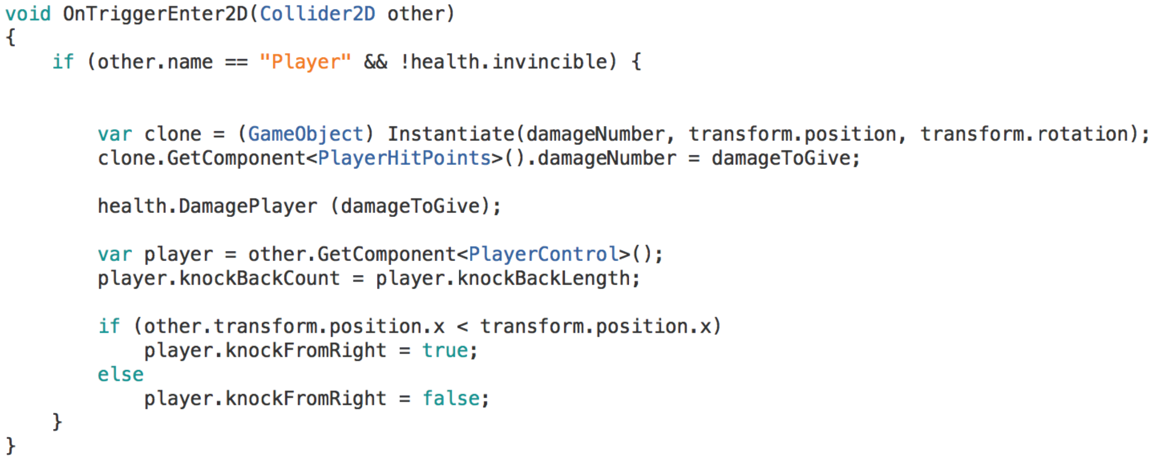
To trigger the damage to the enemy's health we use a OnTriggerEnter2D
So when the collider box attached to these attacks are touched by a gameobject that has been tagged as enemy it'll begin searching for that enemy's health and take away the damageToGive from the enemy's currentHealth as seen below.



```
public void OnTriggerEnter2D(Collider2D other)
{
    if (other.tag == "Enemy")
    {
        other.gameObject.GetComponent<EnemyHealth>().DamageEnemy (damageToGive);
        Instantiate (damageBurst, transform.position, transform.rotation);
        var clone = (GameObject) Instantiate(damageNumber, transform.position, transform.rotation);
        clone.GetComponent<HitPoints>().damageNumber = damageToGive;
    }
}
```

Also being activated in the same moment is a scripted UI element to show the player the damage that has been given to the enemy called HitPoints float off the enemy. A feature seen commonly in RPG games.
DamagePlayer works similarly to DamageEnemy with a few differences. Defining the damage the enemy or environment can do to the player with a singular int value so the player always knows the damage he'll take from each enemy or hazard. This value is then being taken away from the player's health using OnTriggerEnter2D so when the player comes in to contact with the enemy. Simultaneously it'll show the damage taken by the player by a scripted UI element similar to HitPoints in the DamageEnemy script and also hurtle the player away disabling controls and from taking any further damage for a short time in the same

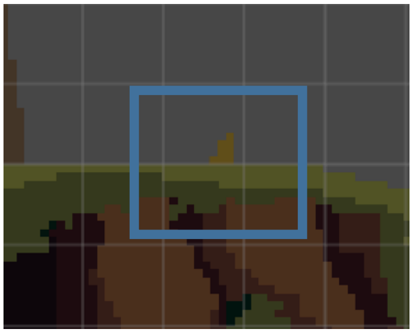
direction he was attacked in to prevent the player taking multiple hits in one attack and also to add some visual flair to the scene.



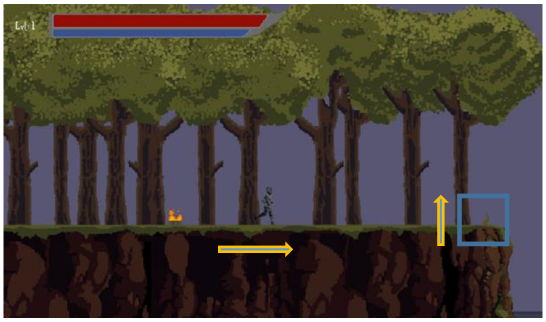
```
void OnTriggerEnter2D(Collider2D other)
{
    if (other.name == "Player" && !health.invincible) {


        var clone = (GameObject) Instantiate(damageNumber, transform.position, transform.rotation);
        clone.GetComponent<PlayerHitPoints>().damageNumber = damageToGive;

        health.DamagePlayer (damageToGive);

        var player = other.GetComponent<PlayerControl>();
        player.knockBackCount = player.knockBackLength;

        if (other.transform.position.x < transform.position.x)
            player.knockFromRight = true;
        else
            player.knockFromRight = false;
    }
}
```

2.3.5. PlantTurretAI .cs

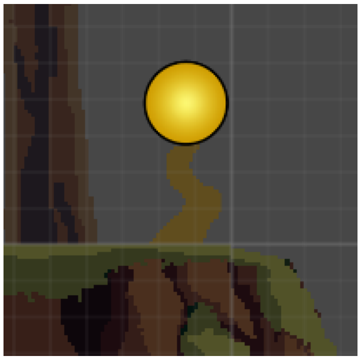
In the PlantTurret's idle position it is hidden underground. When the player approaches it'll rise to so it can begin searching for the player. The player will be discovered when it reaches a certain distance and the PlantTurret will begin firing. If the player manages to stay in the search parameters of the PlantTurretAI, it'll increase its fire rate to a certain point. This is achieved by setting a timer to shootInterval, the timer is reset to less time each time the enemy shoots and the enemy is still within its search parameters. This will then reset once it leaves its search area but not it's awaken area by retrieving the originalInterval. How the PlantTurret AI is aware of the distance of the player is through using Physics2D.OverlapCircle.



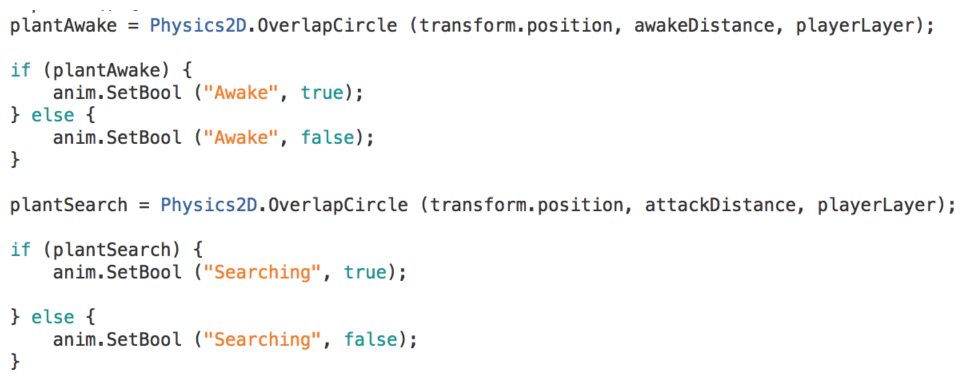
In idle position



As the player approaches the turret rises

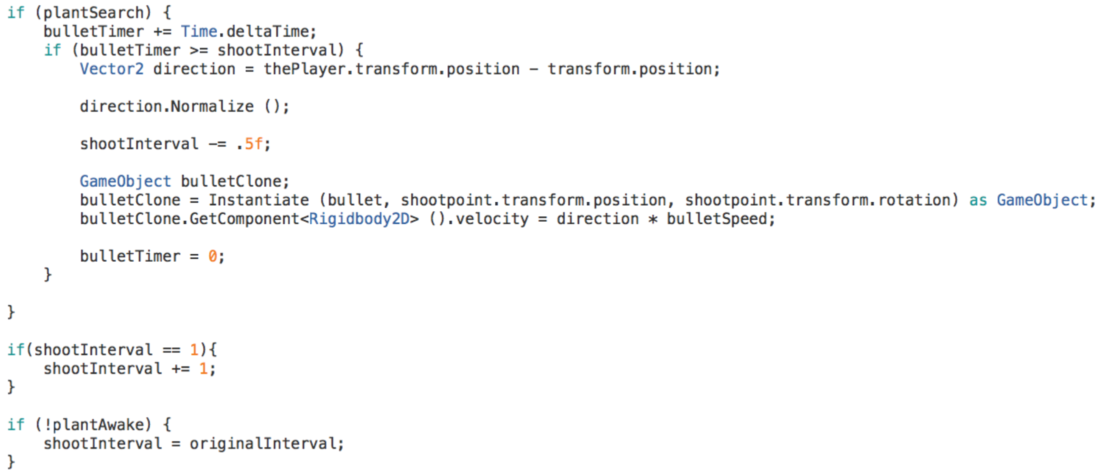


The yellow Gameobject set at the top of the plant, this is where it fires projectiles from



```
plantAwake = Physics2D.OverlapCircle (transform.position, awakeDistance, playerLayer);

if (plantAwake) {
    anim.SetBool ("Awake", true);
} else {
    anim.SetBool ("Awake", false);
}

plantSearch = Physics2D.OverlapCircle (transform.position, attackDistance, playerLayer);

if (plantSearch) {
    anim.SetBool ("Searching", true);

} else {
    anim.SetBool ("Searching", false);
}
```

Here we can see in my C# script how I connected to the Animator in Unity that uses state machine using true or false statements to know to transition to another state of animation.



```
if (plantSearch) {
    bulletTimer += Time.deltaTime;
    if (bulletTimer >= shootInterval) {
        Vector2 direction = thePlayer.transform.position - transform.position;

        direction.Normalize ();

        shootInterval -= .5f;

        GameObject bulletClone;
        bulletClone = Instantiate (bullet, shootpoint.transform.position, shootpoint.transform.rotation) as GameObject;
        bulletClone.GetComponent<Rigidbody2D> ().velocity = direction * bulletSpeed;

        bulletTimer = 0;
    }

}

if(shootInterval == 1){
    shootInterval += 1;
}

if (!plantAwake) {
    shootInterval = originalInterval;
}
```

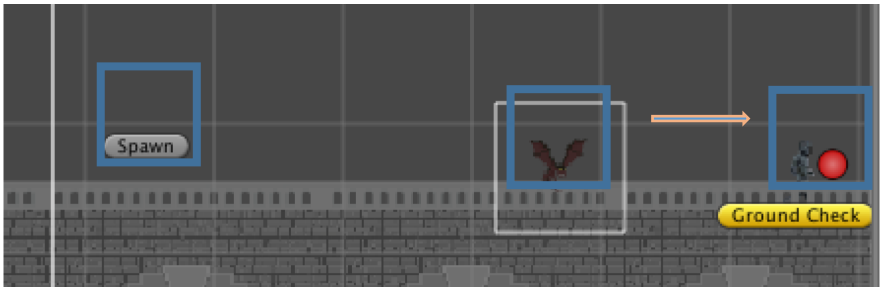
The core of the AI is seen here. When plantSearch is activated bulletTimer begins, which counts down to when the turret fires its first shot. When the bulletTimer is more than the shootInterval the shoot is fired which finds the location of the player so it knows where to fire. The time of the shootInterval is then decreased by .5 of a second. The gameobject bullet clone is a prefab of the bullet, recreating it each time. Finally, the bulletTimer is reset back to 0 and will continue to count back up to the shootInterval if the player is still in the plantSearch area.

Below the shootInterval is prevented from being able to less a bullet per second. This is to prevent the shootInterval going in to the negative and causing errors.

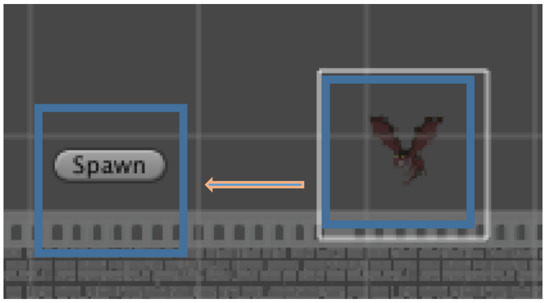
We see the shootInterval is returned back to its originalInterval once the player leaves a certain area.

2.3.6. HarpyAI.cs

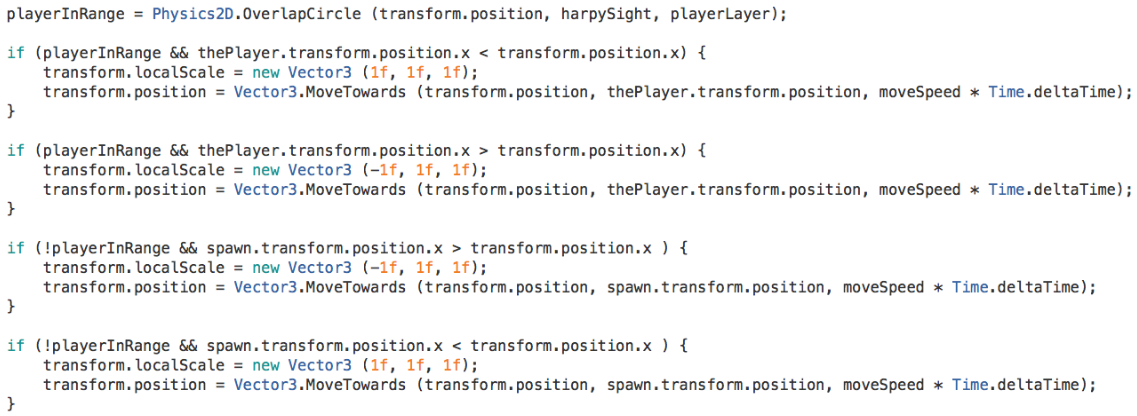
The Harpy AI reacts to the player as it approaches its Spawn area. It'll follow the player for as long as it's in within a certain distance of itself. If the player outruns the Harpy, it'll return back to its spawn location waiting yet again to chase and attack the player.



Can see the harpy AI has left its spawn location to chase the player character and attack it.



The player has left the area of the HarpyAI created using the Physics2D.OverlapCircle and has returned to its spawn location.

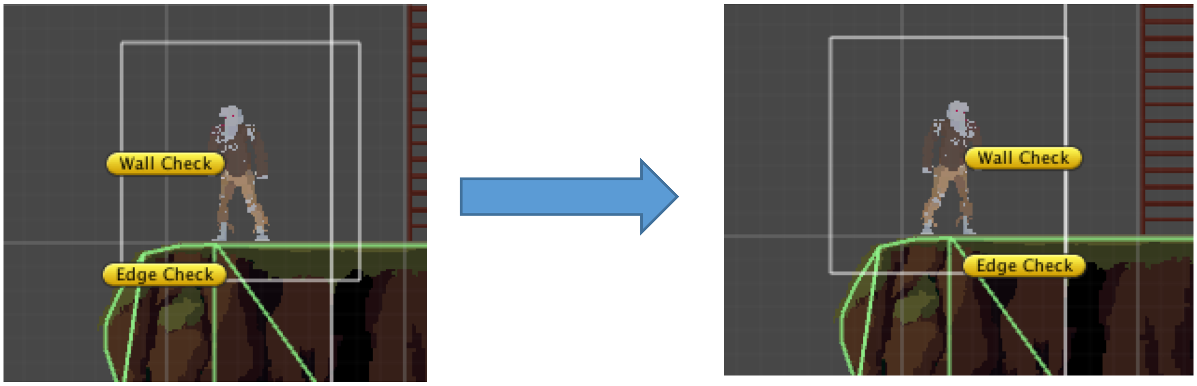


```
playerInRange = Physics2D.OverlapCircle (transform.position, harpySight, playerLayer);

if (playerInRange && thePlayer.transform.position.x < transform.position.x) {
    transform.localScale = new Vector3 (1f, 1f, 1f);
    transform.position = Vector3.MoveTowards (transform.position, thePlayer.transform.position, moveSpeed * Time.deltaTime);
}

if (playerInRange && thePlayer.transform.position.x > transform.position.x) {
    transform.localScale = new Vector3 (-1f, 1f, 1f);
    transform.position = Vector3.MoveTowards (transform.position, thePlayer.transform.position, moveSpeed * Time.deltaTime);
}

if (!playerInRange && spawn.transform.position.x > transform.position.x ) {
    transform.localScale = new Vector3 (-1f, 1f, 1f);
    transform.position = Vector3.MoveTowards (transform.position, spawn.transform.position, moveSpeed * Time.deltaTime);
}

if (!playerInRange && spawn.transform.position.x < transform.position.x ) {
    transform.localScale = new Vector3 (1f, 1f, 1f);
    transform.position = Vector3.MoveTowards (transform.position, spawn.transform.position, moveSpeed * Time.deltaTime);
}
```

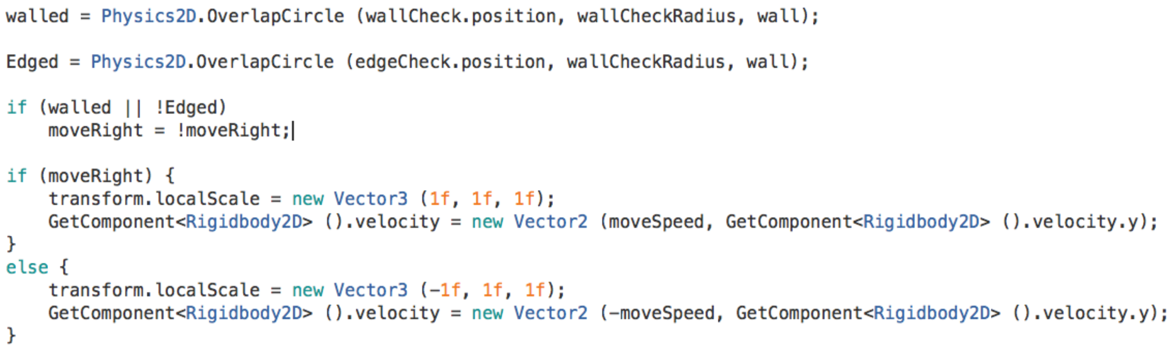
Here is the code I used that makes the the harpy react when the player is near. I had to set up two separate parameters for when the player is near for the animation, thePlayer.transform.position.x > transform.position.x
And
thePlayer.transform.position.x < transform.position.x
retrieving the players positing to know how to be facing the character.
transform.localScale flips the image of the character around to face the player.

2.3.7. GhoulAI.cs

The Ghoul AI is using mostly A* pathfinding to bring it to life. The Ghoul patrols the ground where it’s placed and then with the additions of two gameobjects called Wall Check and Edge Check. Using Physics2D.OverlapCircle on these gameobjects I could command the AI when the Edge Check gameobject is not contacting an object with the ground layer it’ll know that the ground is no longer there and turn around. The same is for the Wall Check but it’ll turn around when the gameobject is contacting an object with the ground layer set to it.



Can see the gameobject edge about to lose contact with the collider of the ground seen in green outline. Once this happens the character turns around.

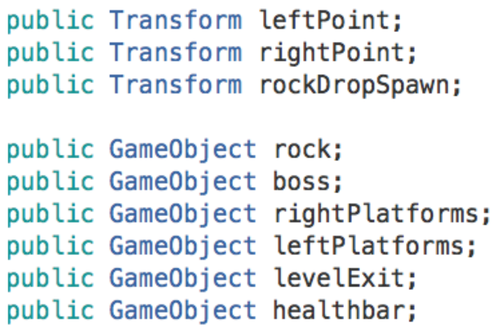


```
walled = Physics2D.OverlapCircle (wallCheck.position, wallCheckRadius, wall);

Edged = Physics2D.OverlapCircle (edgeCheck.position, wallCheckRadius, wall);

if (walled || !Edged)
    moveRight = !moveRight;

if (moveRight) {
    transform.localScale = new Vector3 (1f, 1f, 1f);
    GetComponent<Rigidbody2D> ().velocity = new Vector2 (moveSpeed, GetComponent<Rigidbody2D> ().velocity.y);
}
else {
    transform.localScale = new Vector3 (-1f, 1f, 1f);
    GetComponent<Rigidbody2D> ().velocity = new Vector2 (-moveSpeed, GetComponent<Rigidbody2D> ().velocity.y);
}
```

The code here shows how the AI for enemy is created. The enemy will always begin moving in one direction, which is to the left, hence the –moveSpeed. It’s only when walled or !edged that the moveSpeed is turned positive making the object go the other direction. While also doing this the image of the enemy is flipped using transform.localScale again so it’s traversing in the right direction and not walking backwards.

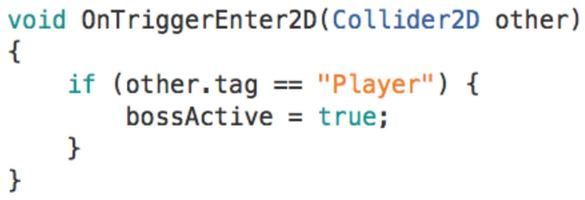
2.3.8. BossBattle.cs

This script controls the climax of our game which is the boss battle. The boss battle is unique in that it is connected to several game objects, the most within the game which were all outlined as public variables so they could be connected together.



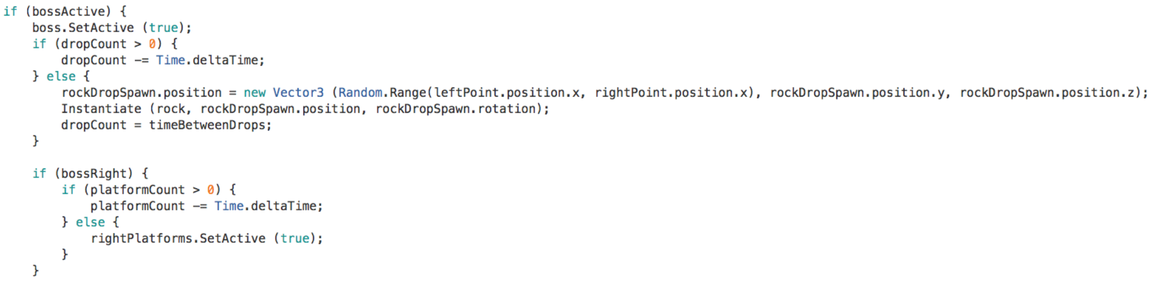
```
public Transform leftPoint;
public Transform rightPoint;
public Transform rockDropSpawn;

public GameObject rock;
public GameObject boss;
public GameObject rightPlatforms;
public GameObject leftPlatforms;
public GameObject levelExit;
public GameObject healthbar;
```

The boss battle is active with the box collider when triggered by the player and setting the boss battle to begin by setting the bossActive Boolean to true.



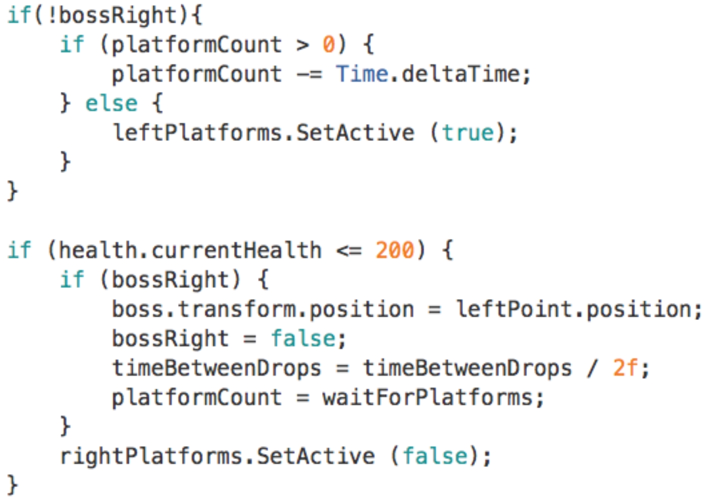
```
void OnTriggerEnter2D(Collider2D other)
{
    if (other.tag == "Player") {
        bossActive = true;
    }
}
```

Once the bossActive is set to true we turn the boss game object set to true so the boss can be seen and interacted with. Starting the dropCount to countdown to spawning the rocks the enemies throws down on player which is spawned between a random range between two game objects. There is also a countdown for the platforms to spawn which is needed to attack the boss as his height is a challenge to attack his head which is his weak spot.



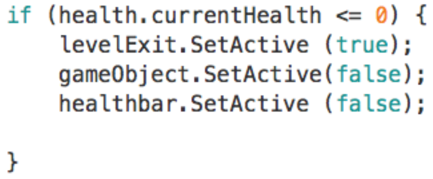
```
if (bossActive) {
    boss.SetActive (true);
    if (dropCount > 0) {
        dropCount -= Time.deltaTime;
    } else {
        rockDropSpawn.position = new Vector3 (Random.Range(leftPoint.position.x, rightPoint.position.x), rockDropSpawn.position.y, rockDropSpawn.position.z);
        Instantiate (rock, rockDropSpawn.position, rockDropSpawn.rotation);
        dropCount = timeBetweenDrops;
    }

    if (bossRight) {
        if (platformCount > 0) {
            platformCount -= Time.deltaTime;
        } else {
            rightPlatforms.SetActive (true);
        }
    }
```

Once the player reaches the halfway point of the boss's health his location changes, his timeBetweenDrops for throwing rocks is decreased increasing the intensity of the battle. The platforms also disappear making the player wait again until they reappear as they dodge rocks.



```
if(!bossRight){
    if (platformCount > 0) {
        platformCount -= Time.deltaTime;
    } else {
        leftPlatforms.SetActive (true);
    }
}

if (health.currentHealth <= 200) {
    if (bossRight) {
        boss.transform.position = leftPoint.position;
        bossRight = false;
        timeBetweenDrops = timeBetweenDrops / 2f;
        platformCount = waitForPlatforms;
    }
    rightPlatforms.SetActive (false);
}
```

Once the player wins the battle all the objects are set to false as to disappear and the levelExit where the player can complete the boss battle level and return to the level select area.

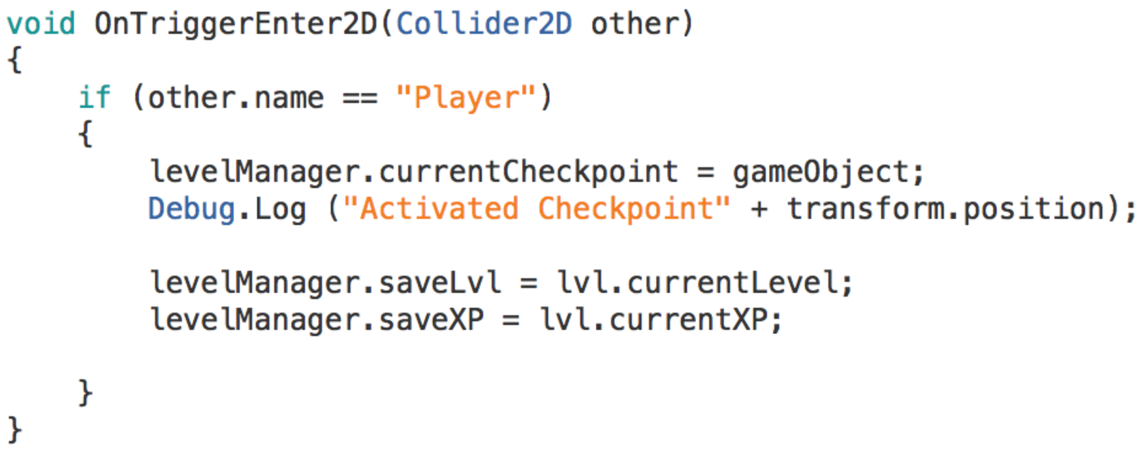


```
if (health.currentHealth <= 0) {
    levelExit.SetActive (true);
    gameObject.SetActive(false);
    healthbar.SetActive (false);

}
```

2.3.9. LevelManager.cs, Checkpoint.cs and ResetOnRespawn.cs

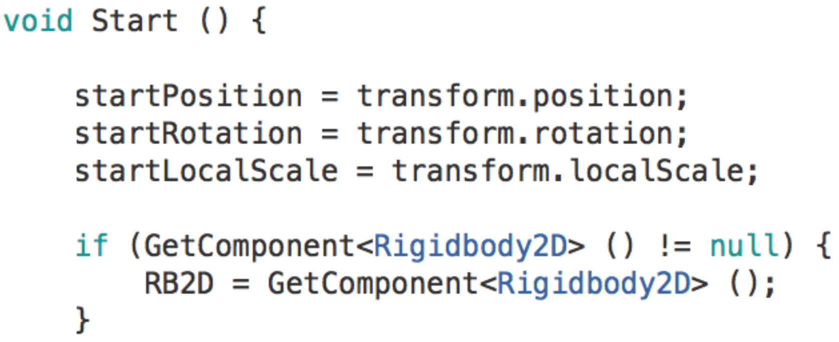
These three scripts work closely together to control the player and enemies after death to make sure the player is respawned on their latest checkpoint reached along with their level when the checkpoint is activated, and the enemies respawn on their predefined spawn location respectively.

The LevelManager script is the core of this implementation, while Checkpoint and ResetOnRespawn sends the data needed to levelManager. Checkpoint collects the data when the checkpoint is activated with a OnTriggerEnter2D function setting the currentCheckpoint to the one just activated and the stats of the player is saved temporarily until the end of the game session of completion of the level.



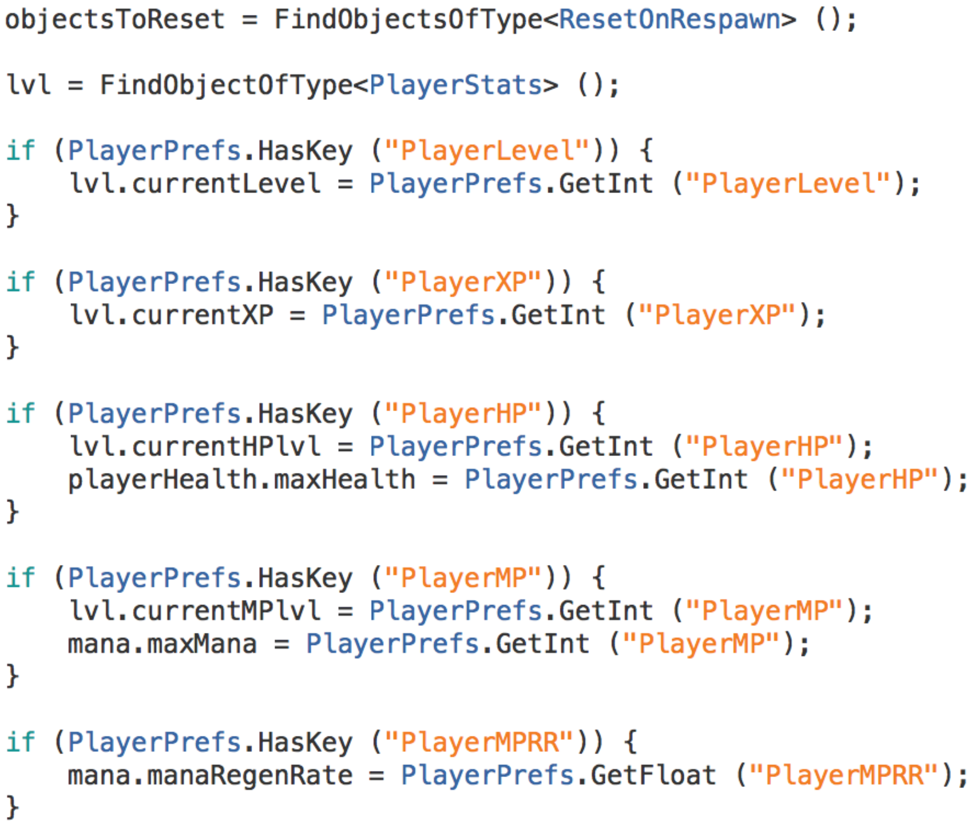
```
void OnTriggerEnter2D(Collider2D other)
{
    if (other.name == "Player")
    {
        levelManager.currentCheckpoint = gameObject;
        Debug.Log ("Activated Checkpoint" + transform.position);

        levelManager.saveLvl = lvl.currentLevel;
        levelManager.saveXP = lvl.currentXP;

    }
}
```

ResetOnRespawn script is attached to all enemy type gameObejcts and when the game starts saves the information of its x,y and z axis position with startPosition. Also other variables such as its rotation(startRotation) and scale(startLocalScale). If there is a Rigidbody2D which was primarly planned to be used for an enemy type that wasn’t fully developed in time.



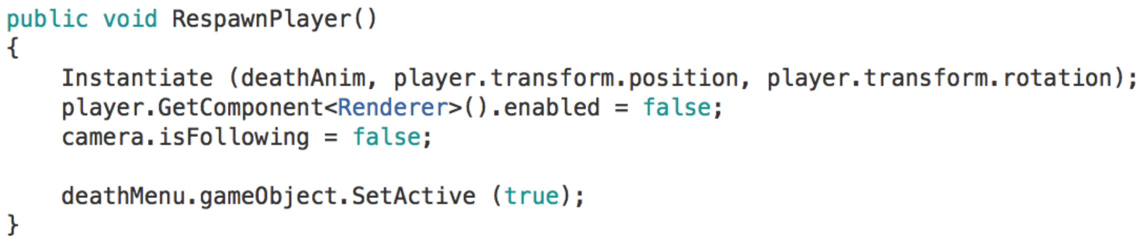
```
void Start () {

    startPosition = transform.position;
    startRotation = transform.rotation;
    startLocalScale = transform.localScale;

    if (GetComponent<Rigidbody2D> () != null) {
        RB2D = GetComponent<Rigidbody2D> ();
    }
```

This all comes together in the LevelManager script. Where we start by searching for all the data variables such as player level and finding the objectsToReset and storing them in an array.



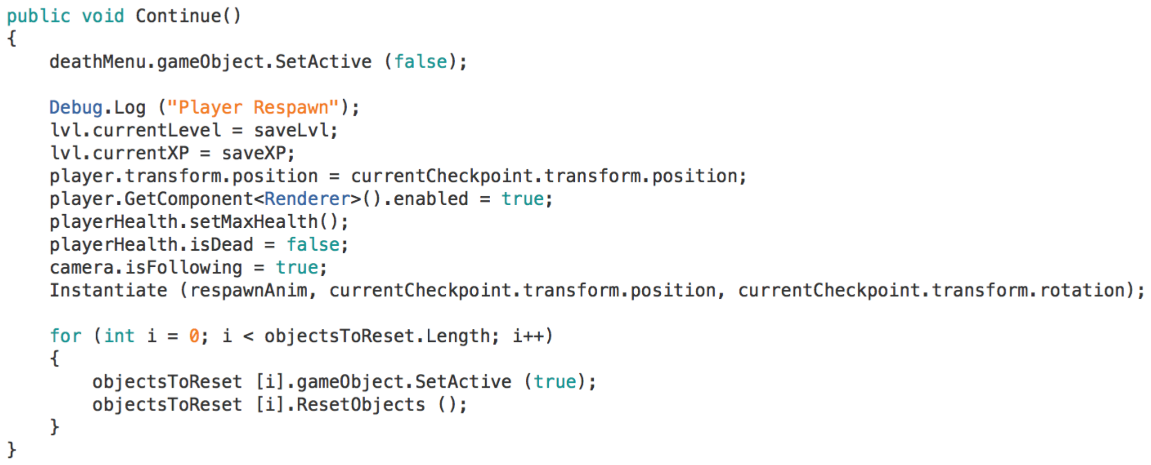
```
objectsToReset = FindObjectsOfType<ResetOnRespawn> ();

lvl = FindObjectOfType<PlayerStats> ();

if (PlayerPrefs.HasKey ("PlayerLevel")) {
    lvl.currentLevel = PlayerPrefs.GetInt ("PlayerLevel");
}

if (PlayerPrefs.HasKey ("PlayerXP")) {
    lvl.currentXP = PlayerPrefs.GetInt ("PlayerXP");
}

if (PlayerPrefs.HasKey ("PlayerHP")) {
    lvl.currentHPlvl = PlayerPrefs.GetInt ("PlayerHP");
    playerHealth.maxHealth = PlayerPrefs.GetInt ("PlayerHP");
}

if (PlayerPrefs.HasKey ("PlayerMP")) {
    lvl.currentMPlvl = PlayerPrefs.GetInt ("PlayerMP");
    mana.maxMana = PlayerPrefs.GetInt ("PlayerMP");
}

if (PlayerPrefs.HasKey ("PlayerMPRR")) {
    mana.manaRegenRate = PlayerPrefs.GetFloat ("PlayerMPRR");
}
```

Using two made functions called RespawnPlayer and Continue we can achieve the goal of resetting enemies to their spawn location will full health and bringing back the player to his last checkpoint with all the information that was true when passing the checkpoint such as level, max health, max magic, etc. In RespawnPlayer we give the death animation called deathAnim using particle effects and disabling controls and camera movement. Finally, we open up the death screen menu by setting it to true and that will give the player the option to continue, restart, and main menu.

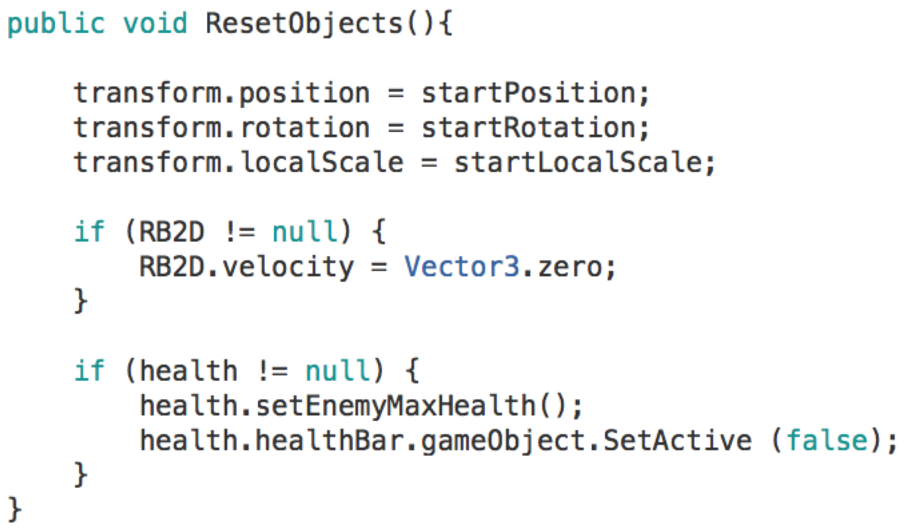


```
public void RespawnPlayer()
{
    Instantiate (deathAnim, player.transform.position, player.transform.rotation);
    player.GetComponent<Renderer>().enabled = false;
    camera.isFollowing = false;

    deathMenu.gameObject.SetActive (true);
}
```

When the player has selected continue we begin resetting the enemy gameobjects and player to the checkpoint. Firstly, turning the deathMenu to false finding the information that was set in the Checkpoint script to set the player to them values and resetting the player’s health to its max value and allowing the camera to follow the player again. Also have a small respawn animation using particle effects. Finally using a for loop we go through the objectsToReset array and set them to active and calling in the information from the ResetOnRespawn script with the function called ResetObjects.



```
public void Continue()
{
    deathMenu.gameObject.SetActive (false);

    Debug.Log ("Player Respawn");
    lvl.currentLevel = saveLvl;
    lvl.currentXP = saveXP;
    player.transform.position = currentCheckpoint.transform.position;
    player.GetComponent<Renderer>().enabled = true;
    playerHealth.setMaxHealth();
    playerHealth.isDead = false;
    camera.isFollowing = true;
    Instantiate (respawnAnim, currentCheckpoint.transform.position, currentCheckpoint.transform.rotation);

    for (int i = 0; i < objectsToReset.Length; i++)
    {
        objectsToReset [i].gameObject.SetActive (true);
        objectsToReset [i].ResetObjects ();
    }
}
```

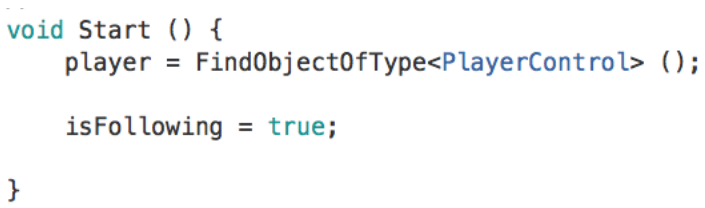


```
public void ResetObjects(){

    transform.position = startPosition;
    transform.rotation = startRotation;
    transform.localScale = startLocalScale;

    if (RB2D != null) {
        RB2D.velocity = Vector3.zero;
    }

    if (health != null) {
        health.setEnemyMaxHealth();
        health.healthBar.gameObject.SetActive (false);
    }
}
```

2.3.10. CameraControl.cs

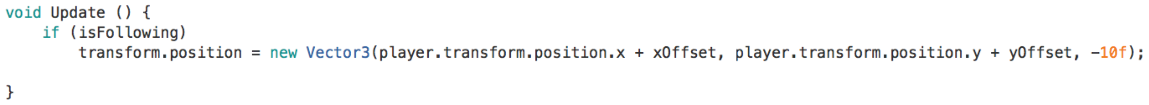
This script controls the camera to follow the player in gameplay. While quite a simple script it is vital for the game or else the player would not be able to see his actions beyond passing the initial starting positon.

First we need to find the player in the game so we FIndObjectOfType playerControl scripts as there is only one instance of this which is attached the player object. We also have a bool for deciding if the camera is following the player which is used in the leveEnd script to create a simple cut scene.



```
void Start () {
    player = FindObjectOfType<PlayerControl> ();

    isFollowing = true;

}
```

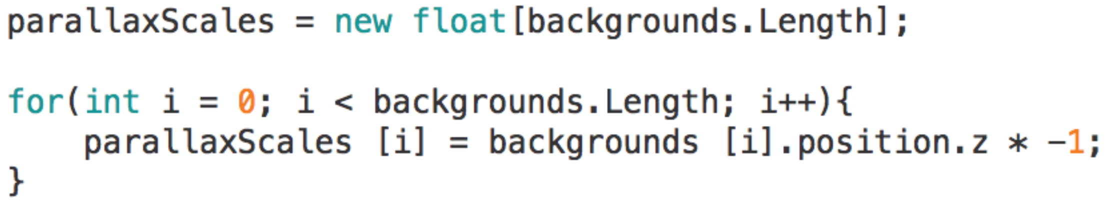
Finally, if the camera is following the player we get the player's x and y axis position which we can then change the position with our xOffset and yOffset variables to get the optimal position.



```
void Update () {
    if (isFollowing)
        transform.position = new Vector3(player.transform.position.x + xOffset, player.transform.position.y + yOffset, -10f);

}
```

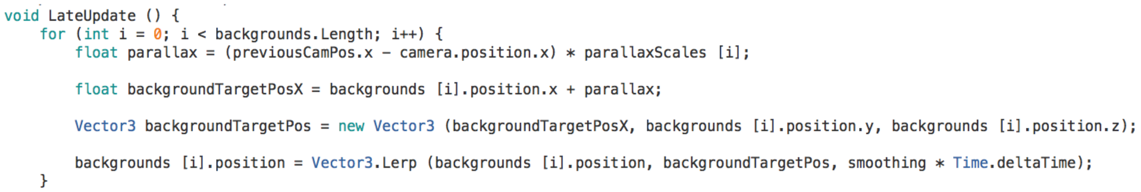
2.3.11. Parallax.cs

This script controls the backgrounds relevant to the player, moving it at a different pace to the player to give the illusion of depth. Essentially getting the the background object’s z axis position relevant to the player. Even though this is a 2D game, it’s created within a 3D development environment. We accomplish this effect through several ways, firstly by using two arrays, backgrounds and parallaxScales, which stores the gameObject’s z position and decides how much the gameobject should move relevant to the player’s positon. Smoothing is used for deciding how fast we want the background to move. Next two items are related to the camera which is following the player, getting the current position and its previous starting positon, knowing the difference between the two so it’ll move the background by the that much. We start by setting up the parallaxScales to have the same array length as backgrounds and then by using a for loop we fill up the list equal to background’s z position.



```
parallaxScales = new float[backgrounds.Length];

for(int i = 0; i < backgrounds.Length; i++){
    parallaxScales [i] = backgrounds [i].position.z * -1;
}
```

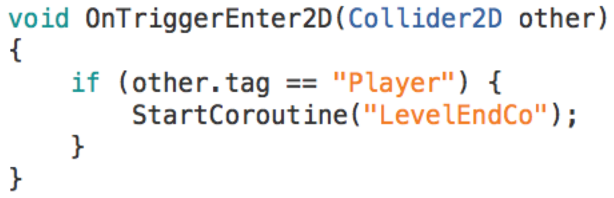
We use void LateUpdate to make this script runs after the camera. Then we are using another for loop to making sure every background gameobject in the array is being moved accordingly. We use parallax to figure out the difference between the current and previous camera position to then find out what its new positon will be with backgroundTargetPosX. Finally, we have a new positon we want to move the background to so we use Vector3.Lerp which will smooth the transition prevent any jumping, using our smoothing value we will have control over how much the background moves when the player moves with the camera.



```
void LateUpdate () {
    for (int i = 0; i < backgrounds.Length; i++) {
        float parallax = (previousCamPos.x - camera.position.x) * parallaxScales [i];

        float backgroundTargetPosX = backgrounds [i].position.x + parallax;

        Vector3 backgroundTargetPos = new Vector3 (backgroundTargetPosX, backgrounds [i].position.y, backgrounds [i].position.z);

        backgrounds [i].position = Vector3.Lerp (backgrounds [i].position, backgroundTargetPos, smoothing * Time.deltaTime);
    }
```

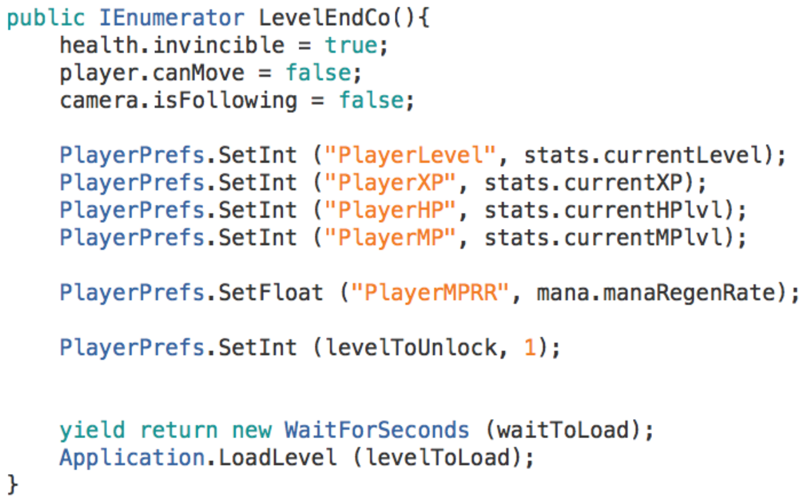
2.3.12. LevelEnd.cs

This script automatically saves all the current data in PlayerStats script and also opening up access to the next level and saving this activation so when the player stops playing the game no important data will be lost, such as player level, magic regeneration rate, etc. This is all done with the use of PlayerPrefs storing the data in a preset folder depending on the device the game is running on. A StartCorutine is activated called LevelEndCo with an OnTriggerEnter2D and that the item colliding with the collider is the player and not anything else.



```
void OnTriggerEnter2D(Collider2D other)
{
    if (other.tag == "Player") {
        StartCoroutine("LevelEndCo");
    }
}
```

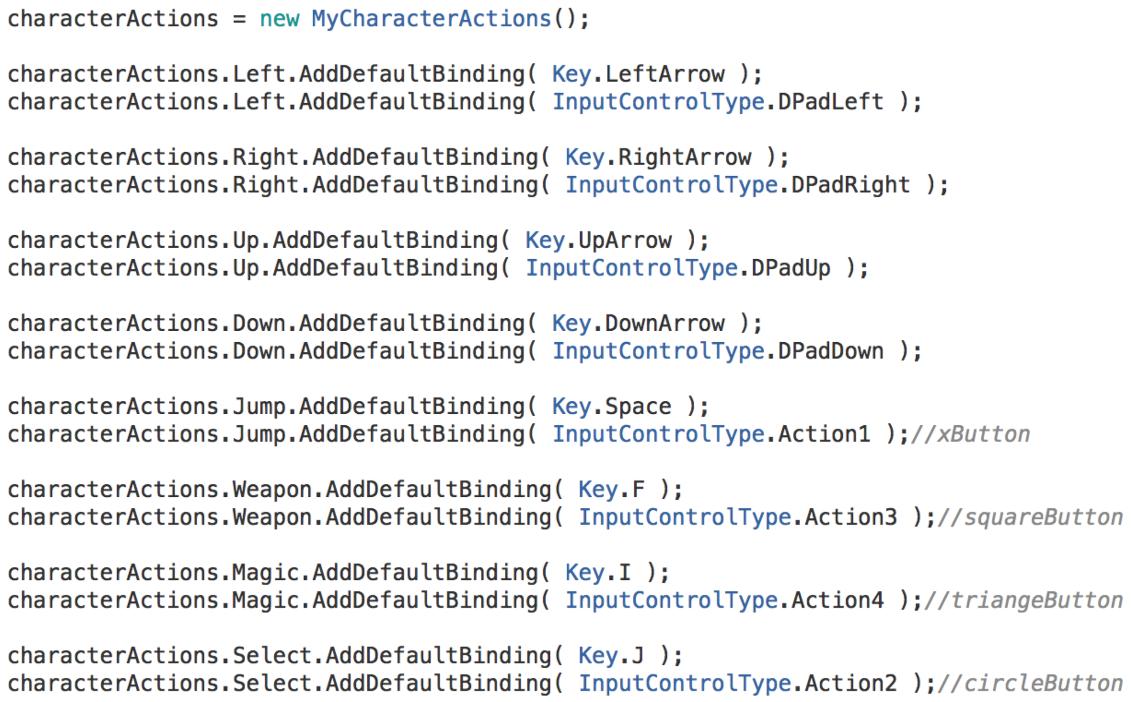
IEnumerator is setting the order of in which the data is saved and setting precautions so as making the player invincible so nothing in the game world can interfere with the process. For the the next level to unlock I use a binary system of 0 and 1, being locked and unlocked respectively. This is defined within the LevelSelect script along with a script setting how the player can enter a new level. So we get the PlayerPrefs to change and save the default for the next level to 1.



```
public IEnumerator LevelEndCo(){
    health.invincible = true;
    player.canMove = false;
    camera.isFollowing = false;

    PlayerPrefs.SetInt ("PlayerLevel", stats.currentLevel);
    PlayerPrefs.SetInt ("PlayerXP", stats.currentXP);
    PlayerPrefs.SetInt ("PlayerHP", stats.currentHPlvl);
    PlayerPrefs.SetInt ("PlayerMP", stats.currentMPlvl);

    PlayerPrefs.SetFloat ("PlayerMPRR", mana.manaRegenRate);

    PlayerPrefs.SetInt (levelToUnlock, 1);


    yield return new WaitForSeconds (waitToLoad);
    Application.LoadLevel (levelToLoad);
}
```

2.3.13. PlayerControl.cs

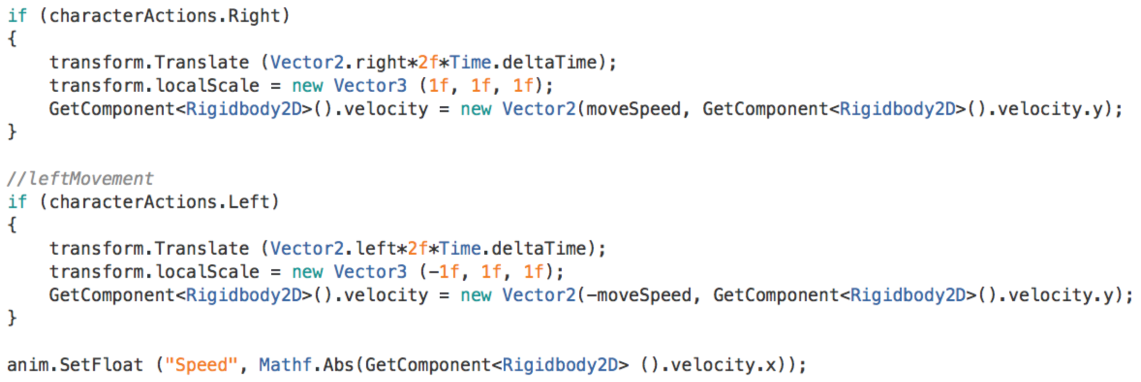
This script is essential for player to play the game as it controls the character in the game world and connects with many other objects through the player gameObject. I begin with connecting the MyCharacterActions script so that the control scheme for the game is set by binding actions to controls. Bindings can include mouse input, keyboard input and any of the standard or non-standard controls for supported devices. First we import the InControl namespace by adding using InControl.



```
characterActions = new MyCharacterActions();

characterActions.Left.AddDefaultBinding( Key.LeftArrow );
characterActions.Left.AddDefaultBinding( InputControlType.DPadLeft );

characterActions.Right.AddDefaultBinding( Key.RightArrow );
characterActions.Right.AddDefaultBinding( InputControlType.DPadRight );

characterActions.Up.AddDefaultBinding( Key.UpArrow );
characterActions.Up.AddDefaultBinding( InputControlType.DPadUp );

characterActions.Down.AddDefaultBinding( Key.DownArrow );
characterActions.Down.AddDefaultBinding( InputControlType.DPadDown );

characterActions.Jump.AddDefaultBinding( Key.Space );
characterActions.Jump.AddDefaultBinding( InputControlType.Action1 );//xButton

characterActions.Weapon.AddDefaultBinding( Key.F );
characterActions.Weapon.AddDefaultBinding( InputControlType.Action3 );//squareButton

characterActions.Magic.AddDefaultBinding( Key.I );
characterActions.Magic.AddDefaultBinding( InputControlType.Action4 );//triangeButton

characterActions.Select.AddDefaultBinding( Key.J );
characterActions.Select.AddDefaultBinding( InputControlType.Action2 );//circleButton
```

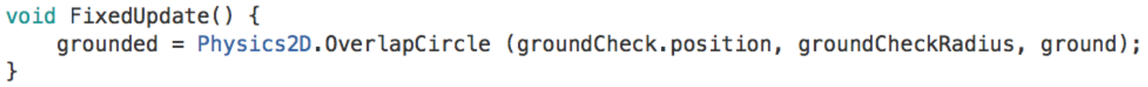
In the image above we can see the actions such as moving, jumping, attacking etc. defined in MyCharacterActions being binded to gamepad controls and keyboard controls. This can be further extended to touch controls with a future port of the game.

The character moves along the x axis depending on the inputs from the player and changes the animation of the character from idle to running by using the animator within Unity and seeing if the set variables match, which in this case the variables are that the Speed of the character.



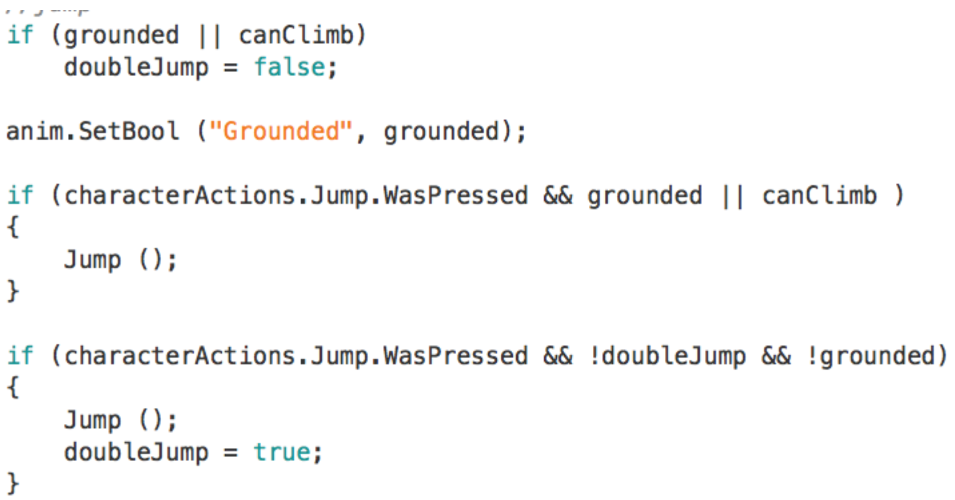
```
if (characterActions.Right)
{
    transform.Translate (Vector2.right*2f*Time.deltaTime);
    transform.localScale = new Vector3 (1f, 1f, 1f);
    GetComponent<Rigidbody2D>().velocity = new Vector2(moveSpeed, GetComponent<Rigidbody2D>().velocity.y);
}

//leftMovement
if (characterActions.Left)
{
    transform.Translate (Vector2.left*2f*Time.deltaTime);
    transform.localScale = new Vector3 (-1f, 1f, 1f);
    GetComponent<Rigidbody2D>().velocity = new Vector2(-moveSpeed, GetComponent<Rigidbody2D>().velocity.y);
}

anim.SetFloat ("Speed", Mathf.Abs(GetComponent<Rigidbody2D> ().velocity.x));
```

To perform a jump, we must first check that the player's GroundCheck game object must be in contact with a collider that has its layer set to ground. We use a Physics2D.OverlapCircle to check the position of the groundCheck object, then setting the radius of the OverlapCircle using groundCheckRadius and finally set the layer to be looking for which is ground. The reason this is done is to prevent the player from constantly being able to jump upwards without any limits which would cause extensive problems with the player easily able to escape the confounds of the level map.

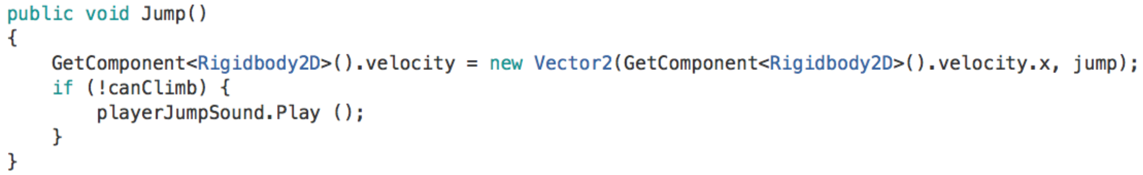


```
void FixedUpdate() {
    grounded = Physics2D.OverlapCircle (groundCheck.position, groundCheckRadius, ground);
}
```

So we have prevented the player from jumping multiple times constantly without touching the ground but we need to be able to allow the player to double jump as this is a common feature seen in platforming games. So we use a bool called doubleJump and if the player is in the air this value is set to true but then set immediately back to false to prevent any further jumps and thus creating a double jump feature. We have put the Jump function separate simply to prevent repetition of code which would be used for the double jump.

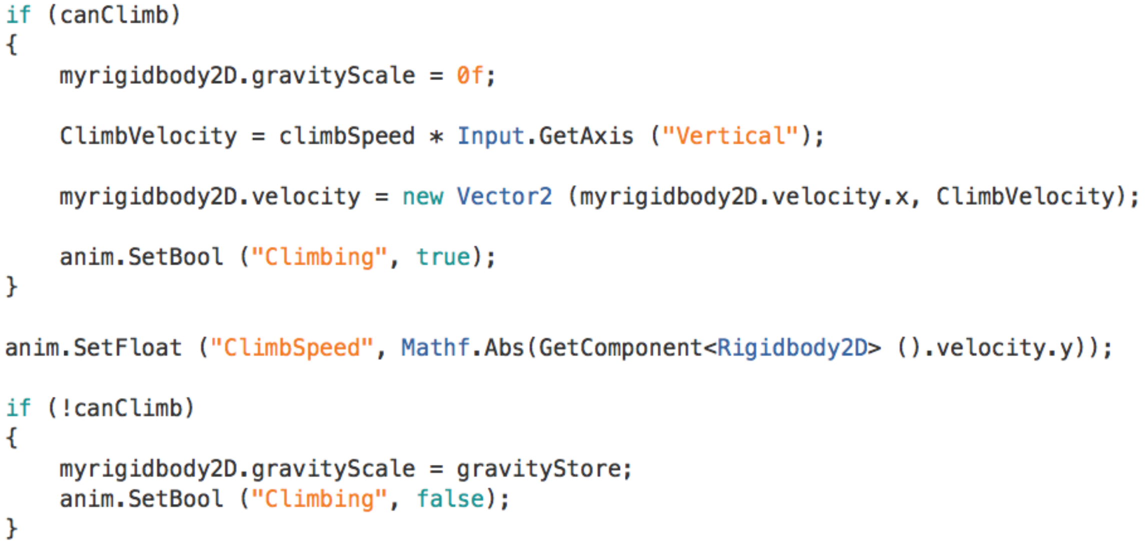


```
//Jump
if (grounded || canClimb)
    doubleJump = false;

anim.SetBool ("Grounded", grounded);

if (characterActions.Jump.WasPressed && grounded || canClimb )
{
    Jump ();
}

if (characterActions.Jump.WasPressed && !doubleJump && !grounded)
{
    Jump ();
    doubleJump = true;
}
```



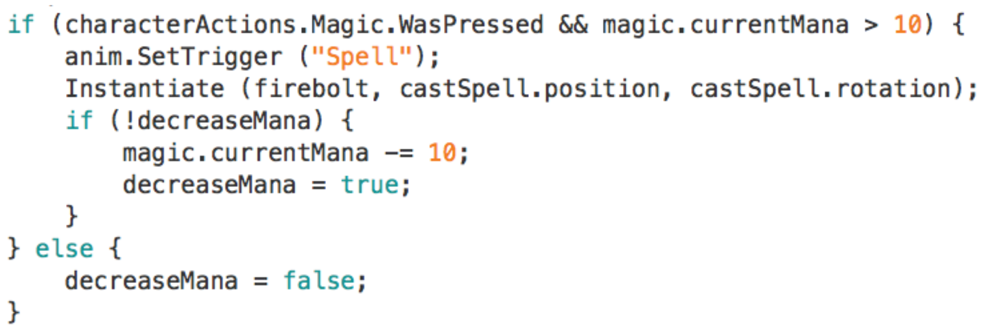
```
public void Jump()
{
    GetComponent<Rigidbody2D>().velocity = new Vector2(GetComponent<Rigidbody2D>().velocity.x, jump);
    if (!canClimb) {
        playerJumpSound.Play ();
    }
}
```

Another movement was climbing up ladders which we had to turn off the gravity for the character for him to give the illusion he's on ladder moving up and down the x axis which the player inputs his commands. We use climbVelocity to set the speed in which the player can climb up and down. We use a bool for the animator to state whether the character is in contact with the ladder and to get the speed in which the character is going along the x axis to show the animation of the player climbing. Finally, if the player is not climbing we restore the gravity to the character.

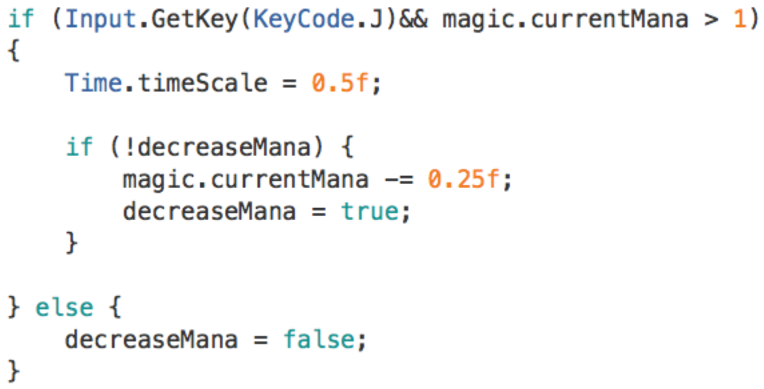


```
if (canClimb)
{
    myrigidbody2D.gravityScale = 0f;

    ClimbVelocity = climbSpeed * Input.GetAxis ("Vertical");

    myrigidbody2D.velocity = new Vector2 (myrigidbody2D.velocity.x, ClimbVelocity);

    anim.SetBool ("Climbing", true);
}

anim.SetFloat ("ClimbSpeed", Mathf.Abs(GetComponent<Rigidbody2D> ().velocity.y));

if (!canClimb)
{
    myrigidbody2D.gravityScale = gravityStore;
    anim.SetBool ("Climbing", false);
}
```

Our attacks using the sword and magic are created similarly but the magic attacks need to decrease the player’s current magic amount. We have to make sure the player has the equivalent amount of magic needed to cause the spell to prevent the values going in to the minus values. Then to stop the further decrease of magic we use an else statement.



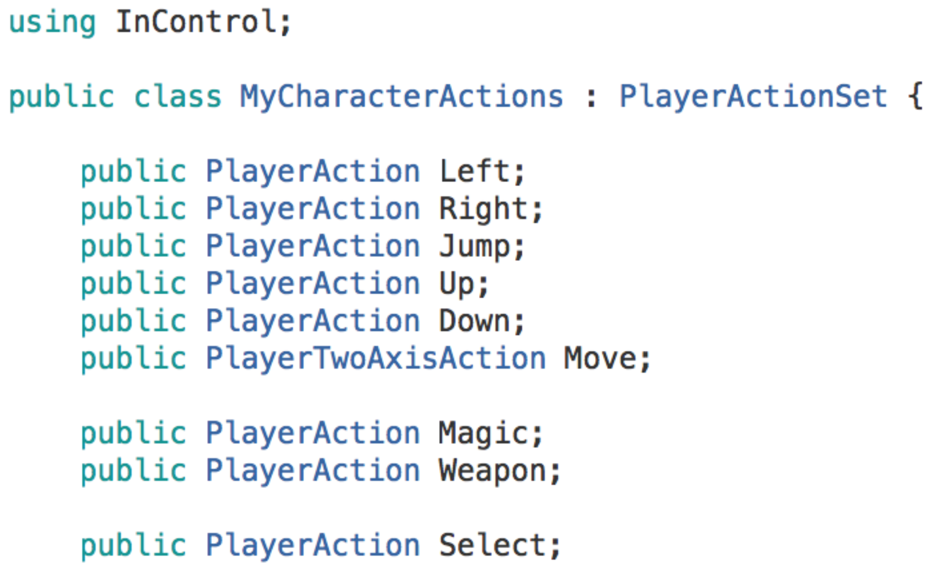
```
if (characterActions.Magic.WasPressed && magic.currentMana > 10) {
    anim.SetTrigger ("Spell");
    Instantiate (firebolt, castSpell.position, castSpell.rotation);
    if (!decreaseMana) {
        magic.currentMana -= 10;
        decreaseMana = true;
    }
} else {
    decreaseMana = false;
}
```



```
if (Input.GetKey(KeyCode.J)&& magic.currentMana > 1)
{
    Time.timeScale = 0.5f;

    if (!decreaseMana) {
        magic.currentMana -= 0.25f;
        decreaseMana = true;
    }

} else {
    decreaseMana = false;
}
```

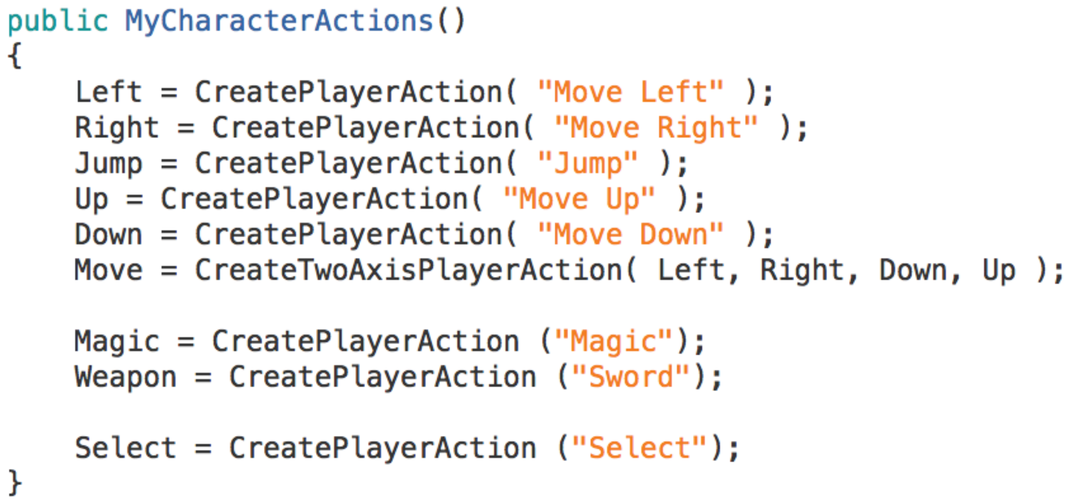
2.3.14. MyCharacterActions.cs

This is where I define the character actions which will later be binded in the PlayerControl script.



```
using InControl;

public class MyCharacterActions : PlayerActionSet {

    public PlayerAction Left;
    public PlayerAction Right;
    public PlayerAction Jump;
    public PlayerAction Up;
    public PlayerAction Down;
    public PlayerTwoAxisAction Move;

    public PlayerAction Magic;
    public PlayerAction Weapon;

    public PlayerAction Select;
```

Next we need to create a subclass of PlayerActionSet called MyCharacterActions that defines these actions.



```
public MyCharacterActions()
{
    Left = CreatePlayerAction( "Move Left" );
    Right = CreatePlayerAction( "Move Right" );
    Jump = CreatePlayerAction( "Jump" );
    Up = CreatePlayerAction( "Move Up" );
    Down = CreatePlayerAction( "Move Down" );
    Move = CreateTwoAxisPlayerAction( Left, Right, Down, Up );

    Magic = CreatePlayerAction ("Magic");
    Weapon = CreatePlayerAction ("Sword");

    Select = CreatePlayerAction ("Select");
}
```

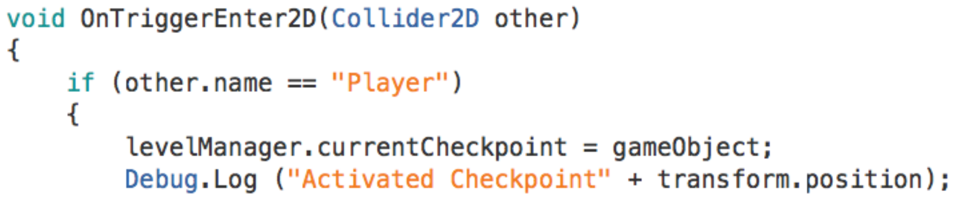
2.4. Testing

Testing was vital in quality assurance of the game to make sure that the product functions as expected to various situations such as the environment and the inputs from the player can not cause any errors. The testing that was performed were unit testing, integration testing and system testing. This was all done with Unity as it has a debugger and console which monitors the game's information and functions and also within Unity's UI inspector of gameObjects. To activate the debugger option to monitor the results We used the function Debug.Log() and Debug.Print().

2.4.1. Unit Testing

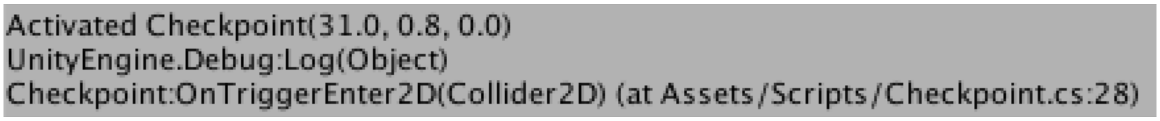
Unit testing was used at the beginning of each script in order to test the code separately from other scripts to make sure the script was functioning properly independently and if errors were to persist it would be from another unknown variable. This helped in ensuring that what I was creating worked as intended before connecting it to the main project which could cause more harm than good.

Best examples are within the Checkpoint script where we have the Debug.Log function programmed to print out the player's activated checkpoint like so.



```
void OnTriggerEnter2D(Collider2D other)
{
    if (other.name == "Player")
    {
        levelManager.currentCheckpoint = gameObject;
        Debug.Log ("Activated Checkpoint" + transform.position);
```

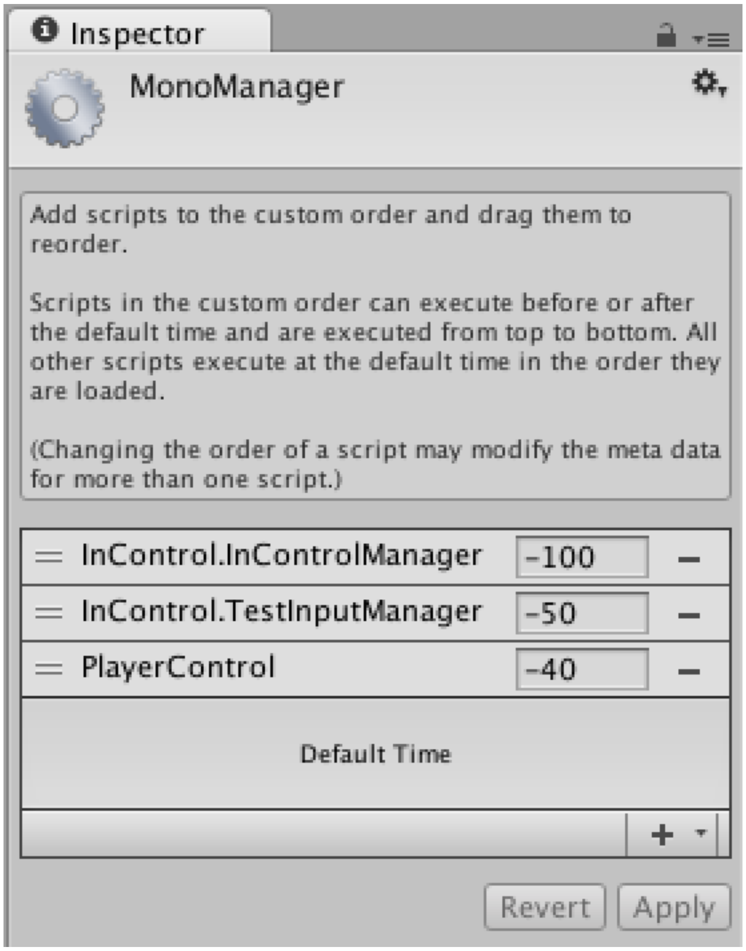
This is then displayed in Unity's debug console as follows.



```
Activated Checkpoint(31.0, 0.8, 0.0)
UnityEngine.Debug:Log(Object)
Checkpoint:OnTriggerEnter2D(Collider2D) (at Assets/Scripts/Checkpoint.cs:28)
```

2.4.2. Integration Testing

Integration testing follows unit testing in this project. This is testing the combination of the separate units and see if they interact correctly. This was important to test as the individual units working would not make a final product and had to be combined with others to make a fully functional game. I found several problems when integrating the units together but was easily solved with Unity’s debugger options seeing that occasionally some scripts were being called before others causing problems but was easily fixed by using Unity’s script execution setting scripts before the default time by using minus integers and after the default time by using positive integers.



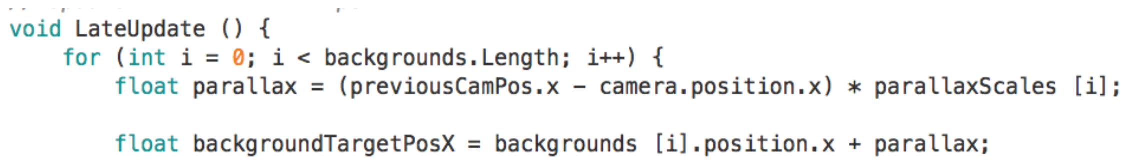
Add scripts to the custom order and drag them to reorder.

Scripts in the custom order can execute before or after the default time and are executed from top to bottom. All other scripts execute at the default time in the order they are loaded.

(Changing the order of a script may modify the meta data for more than one script.)

InControl.InControlManager	-100
InControl.TestInputManager	-50
PlayerControl	-40

Also using Unity’s function of LateUpdate as oppose to Update can achieve the same effect but not as a precise order as can be achieved in Unity’s script execution order. This was only used in one script so this was not a major issue where the execution order of the camera would vary between the CameraControl script and Parallax script.



```
void LateUpdate () {
    for (int i = 0; i < backgrounds.Length; i++) {
        float parallax = (previousCamPos.x - camera.position.x) * parallaxScales [i];

        float backgroundTargetPosX = backgrounds [i].position.x + parallax;
```

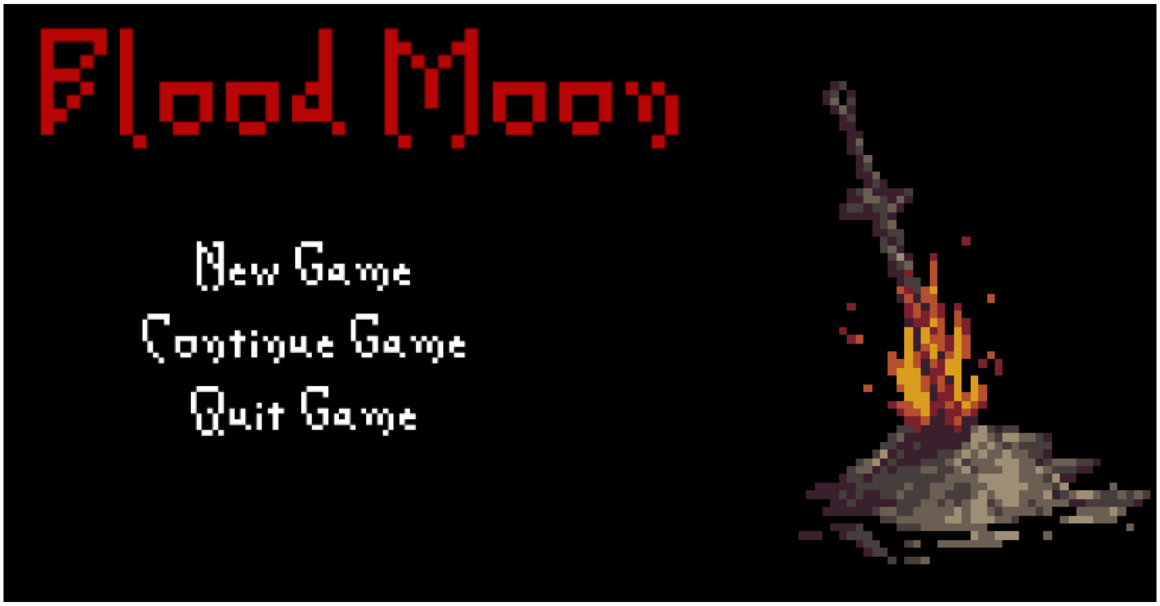
2.4.3. System Testing

This was the final testing of the game, making sure Blood Moon runs smoothly and as intended on various systems intended for the first wave of releases. The systems that I tested were computer operating systems I tested were Mac OSX and Windows. I also tested the software to work with various game controllers such as a PlayStation and Xbox controller. I intended to test these on the console themselves but do not have a development kits of these consoles as they are provided to developers by Sony and Microsoft themselves. As of writing though Sony and Microsoft are updating their respective consoles to allow consumers to turn their console in to a development kit. There is the possibility that the software could run on a Linux OS but due to lack of resources this could not be confirmed. All this worked as intended on the first round of testing and did not combat any errors.

2.5. Graphical User Interface (GUI) Layout

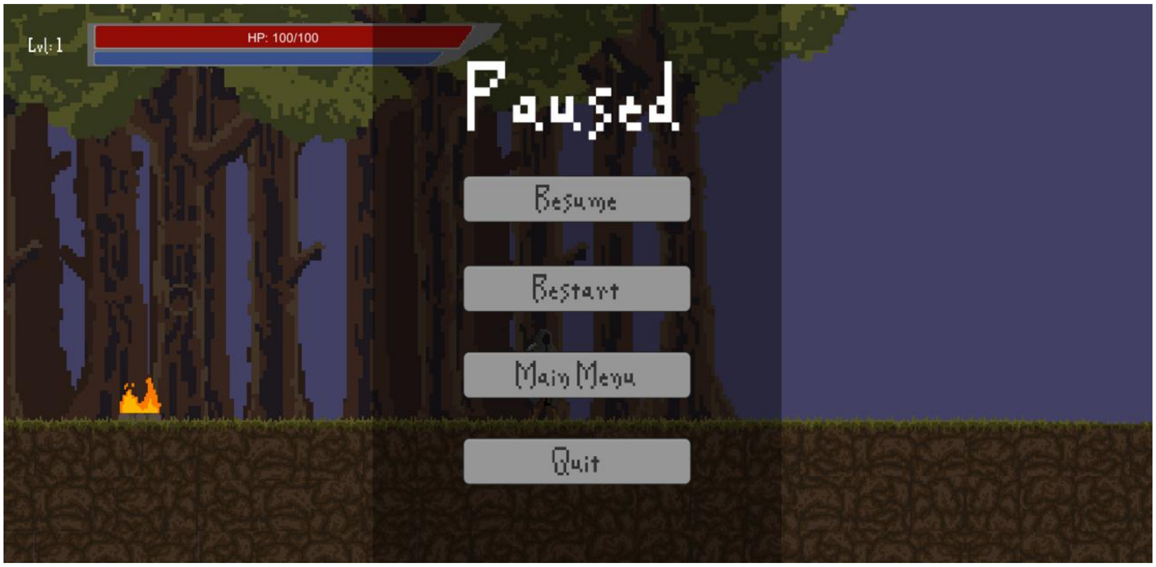
2.5.1. Menus

2.5.1.1. Main Menu



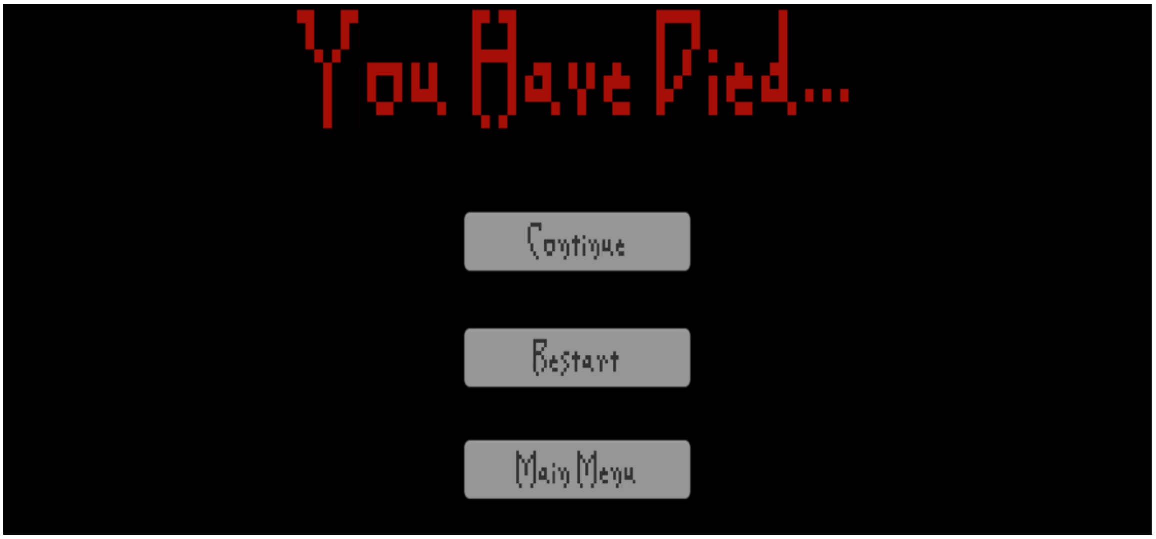
This is the title menu screen; it is what you see first when opening up Blood Moon and provides the player with several options that they can choose from. The first option is New Game, which will allow the player to start a new game and bring you to the first area in the game. The second option Continue Game will bring you to the area select if you have played the game before and have made a save. You'll have access in area select to previous areas and areas recently unlocked. The third option is Quit game, this closes the application and returns the player to their desktop environment. The design is dark and represents the game's intended theme.

2.5.1.2. Pause Menu



This is the pause menu screen; it is what you see when the player pauses the game during gameplay. The player is given several options, some of which are seen within the Main Menu screen. The first option Resume, resumes the game and continues on with gameplay. The second option Restart, restarts the current level and deletes any progress made so far within the level, such as activated checkpoints and XP gained. The third option returns the player back to the Main Menu screen. The fourth Quit function the same as in the Main Menu screen respectively. The design is plain and simple, a trait I found common in many pause menu screens in games.

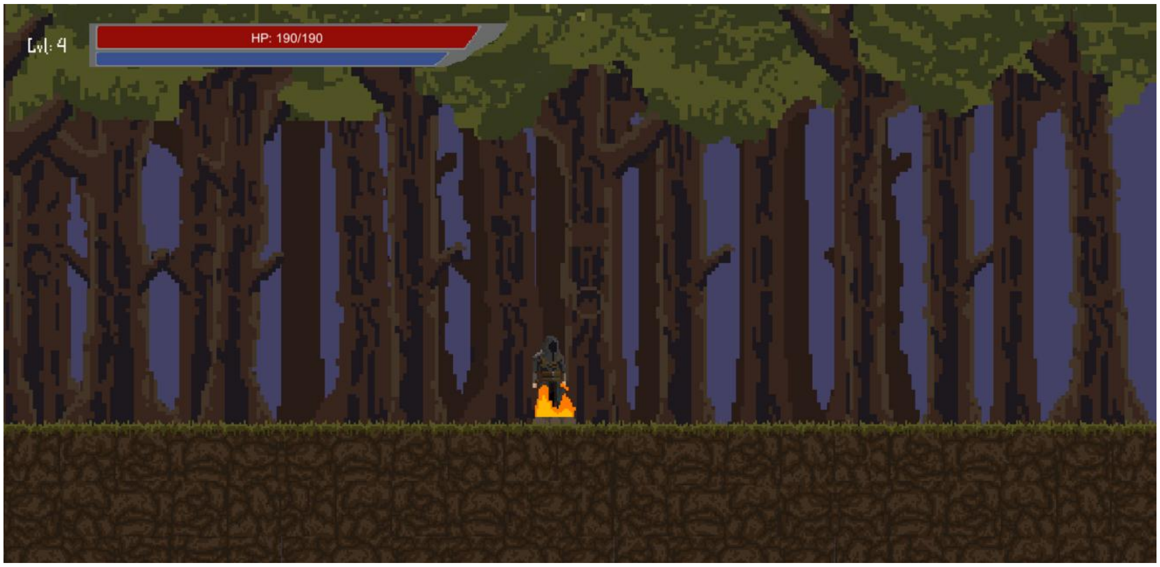
2.5.1.3. Death Screen Menu



This is the death screen menu; it appears when the player dies providing him with three options. The first option is Continue which activates the respawn of the player and enemies saved between the Checkpoint and ResetOnRespawn scripts. The second option simply restarts the level along with deleting any data stored that hasn't been saved by completing the level. The third option brings the player back to the main menu screen.

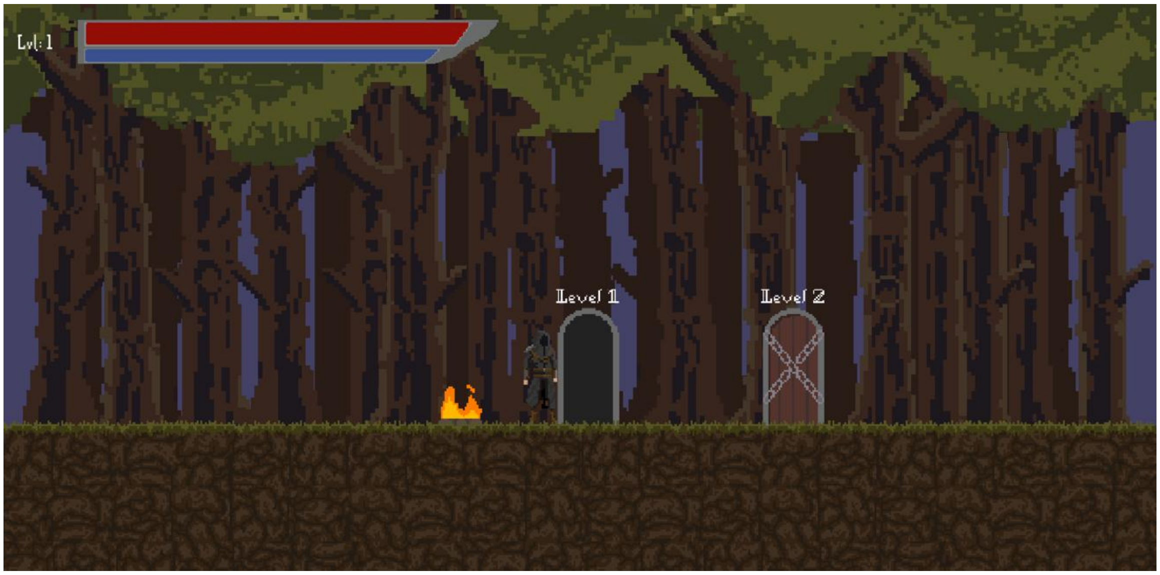
2.5.2. In Game

2.5.2.1. Heads Up Display (HUD)



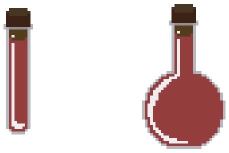
This screenshot represents the main UI for the player in the game world. Starting at the top left corner, the player’s health value is represented with a red health bar, created using a slider with an image placed over to give it a more stylised looking fitting of the world I am creating. On top of the slider is the player’s current health and max health represented with text which was added after customer testing who felt the effects of the levelling up system needed to represented better. Directly underneath the player’s magic value represented with a blue bar and created similarly as the health bar. To the immediate left of both of these UI items is the player’s current level. The design is clean and out of the way in the corner as to not take away focus from the gameplay.

2.5.2.2. Level Select



This is the level select screen; it is the area that you sent to after completion of a level or when you continue you game. The doors represent their respected level and you can see that the door can be in two states, locked (chained door) and unlocked (no door), to unlock the door to the next level the player will have to complete the previous level. Once the player approaches the door, he'll have to press a key/button to enter an unlocked level. This is so the player can freely walk between the road signs and not be accidently sent to the wrong level. The design was inspired by retro level select areas, making the process more interactive and enjoyable.

2.5.2.3. In Game Items



Health pickups with varying values that are dropped at random on the enemy’s death to heal the damage player.

Loot pickups with varying values that are dropped at random on the enemy’s death and scattered around levels hidden to increase XP.

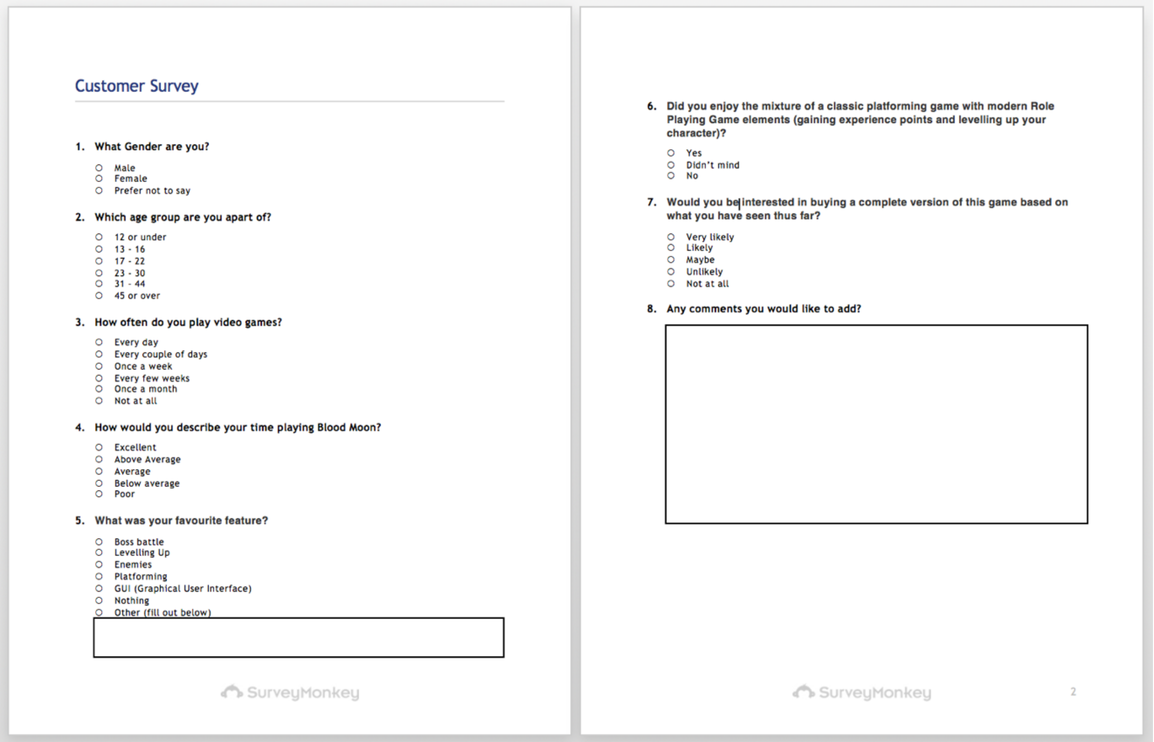
Road sign seen at the end of the level signify the end and completion of the level when reached.

Fireplace signifies the checkpoint location which is activated when passing by.

These are the main game objects in the game that interact with the player throughout the game with various functions. The design of these items were kept in line with the game world being corrected as to blend in with the world while standing out to the player visually.

2.6. Customer testing

Using a customer survey template, I was able to gather a substantial volume of data quickly on the set of questions I outlined. The survey looked as follows:



Customer Survey

1. What Gender are you?

   ○ Male
   ○ Female
   ○ Prefer not to say

2. Which age group are you apart of?

   ○ 12 or under
   ○ 13 - 16
   ○ 17 - 22
   ○ 23 - 30
   ○ 31 - 44
   ○ 45 or over

3. How often do you play video games?

   ○ Every day
   ○ Every couple of days
   ○ Once a week
   ○ Every few weeks
   ○ Once a month
   ○ Not at all

4. How would you describe your time playing Blood Moon?

   ○ Excellent
   ○ Above Average
   ○ Average
   ○ Below average
   ○ Poor

5. What was your favourite feature?

   ○ Boss battle
   ○ Levelling Up
   ○ Enemies
   ○ Platforming
   ○ GUI (Graphical User Interface)
   ○ Nothing
   ○ Other (fill out below)

6. Did you enjoy the mixture of a classic platforming game with modern Role Playing Game elements (gaining experience points and levelling up your character)?

   ○ Yes
   ○ Didn't mind
   ○ No

7. Would you be interested in buying a complete version of this game based on what you have seen thus far?

   ○ Very likely
   ○ Likely
   ○ Maybe
   ○ Unlikely
   ○ Not at all

8. Any comments you would like to add?

I had to keep in mind to not make the survey too long as people would be less inclined to fill out the form. These questions I set covered all the data I needed to receive from a customer's point of view of the game.

Questions 1 to 3 give us the information on the demographic answering our survey so we better understand the potential market.

Question 4 to 6 are the core of the survey getting the customer's opinions on Blood Moon. Given options to express their experience with their time, the feature they most enjoyed and finally if they enjoyed the mixture of old genre of video games with elements of a new genre.

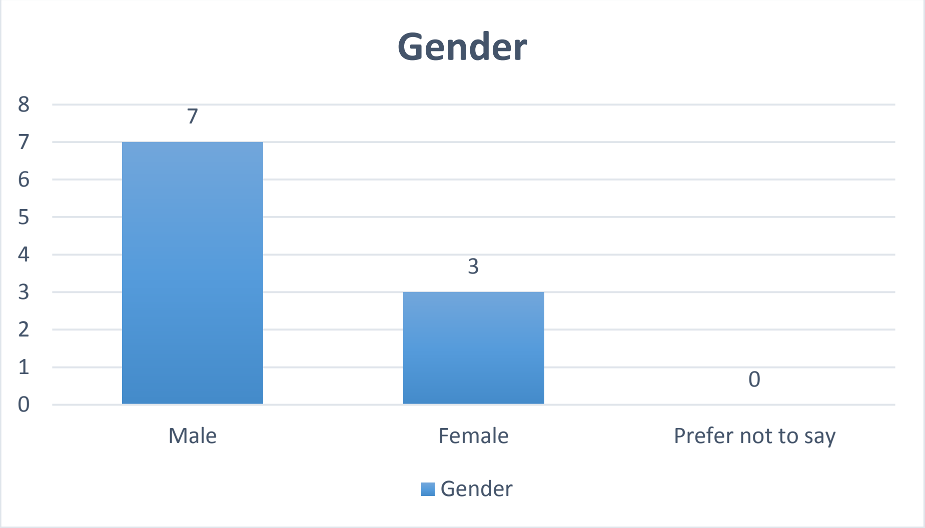
Question 7 sees if we have compelled the customer in buying Blood Moon based on what they have seen so far which will tell me if the project would be a commercial success.

Question 8 where the customer is asked to add any optional comments would cover anything that I could not in this survey.

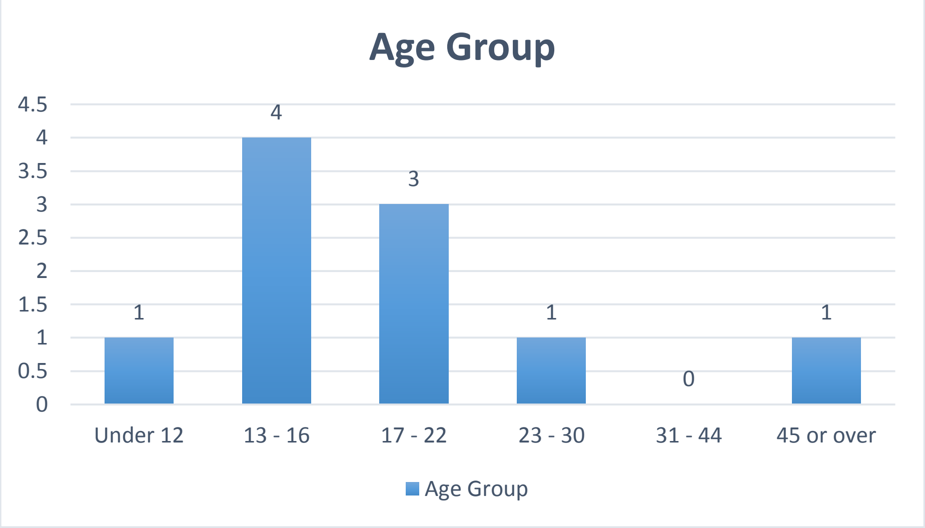
2.7. Evaluation

Here are the results of the customer testing which was carried out on ten people from as varied demographics as possible. The Other Materials used stores all ten surveys in further detail.

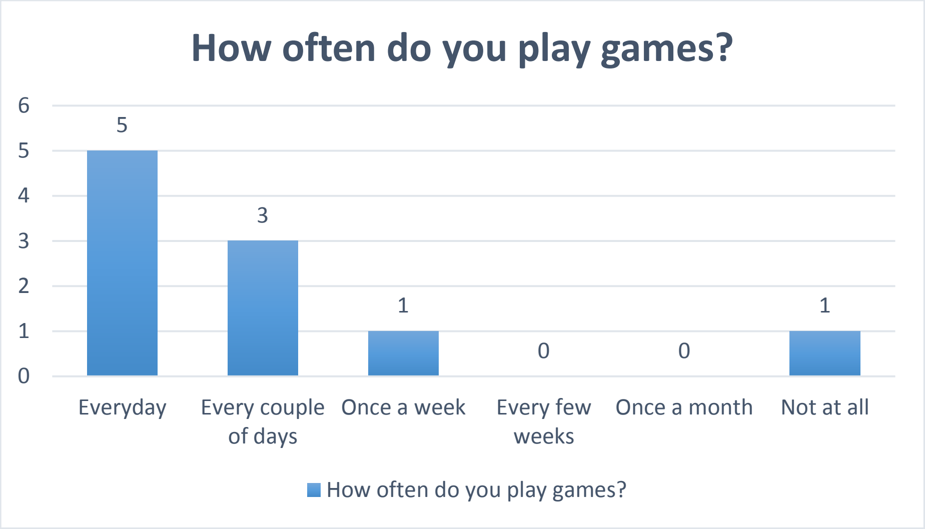
What Gender are you?



Which age group are you apart of?

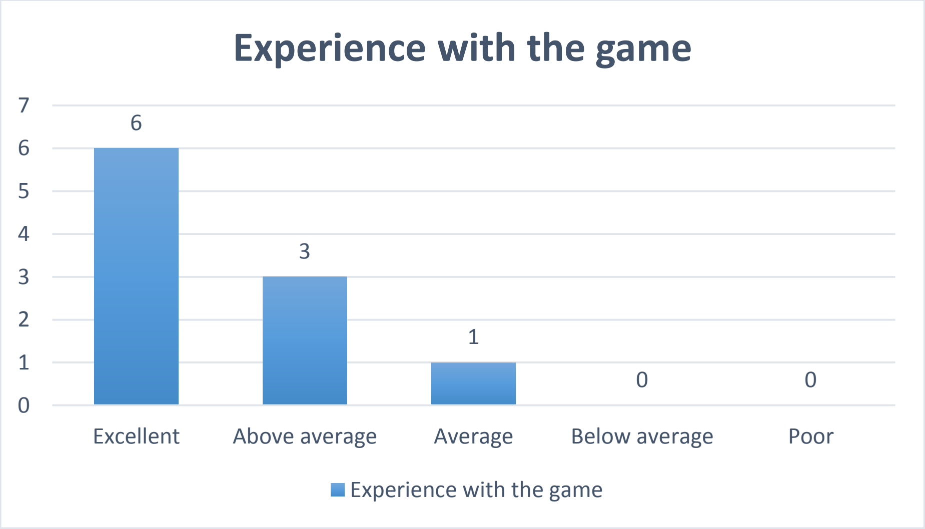


How often do you play video games?

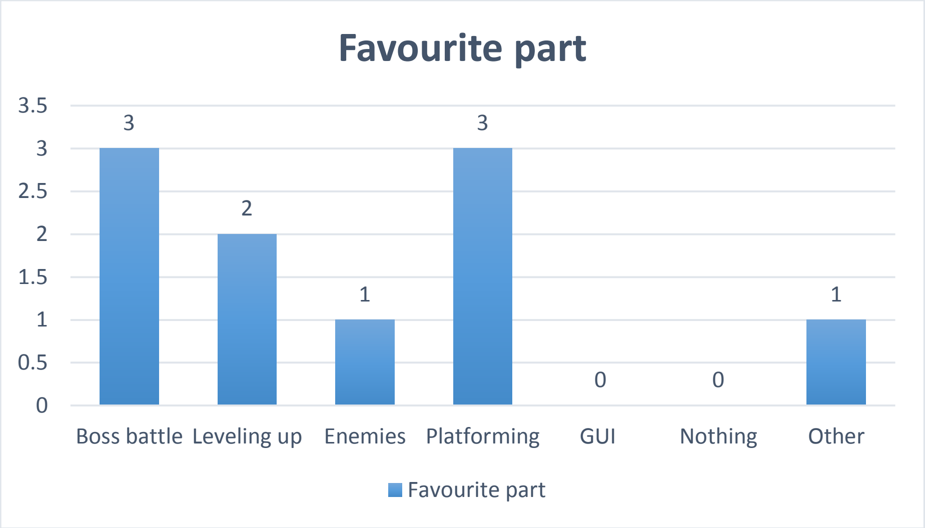


We can see from these three questions that are demographic is quite varied while leaning more towards the younger gaming audience which is important as this is the key market to target on release hence why the game appealed to them more to test. There weren't many people that I could find who did not play games frequently as the growth of the mobile app market with the Apple App Store and Google Play Store gives people access to free games that can be played in short bursts aligns with my previous statements.

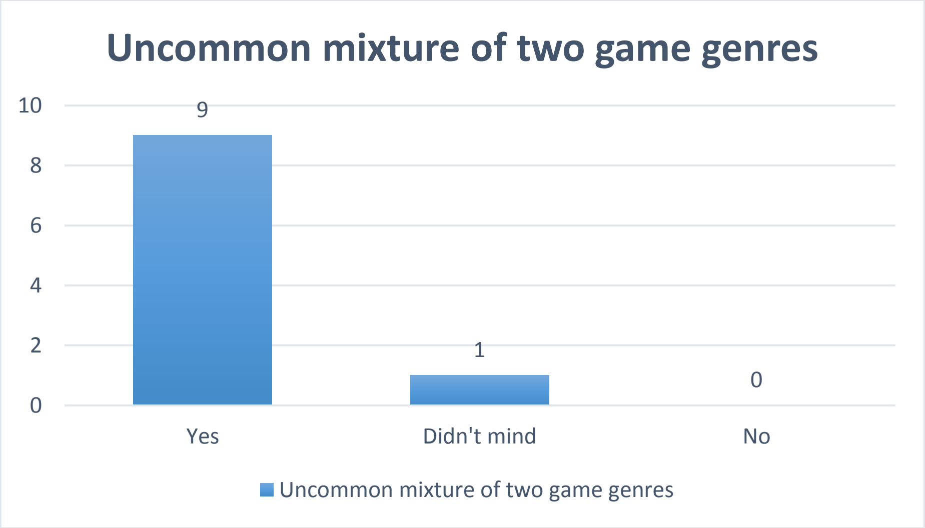
How would you describe your time playing Blood Moon?



What was your favourite part?

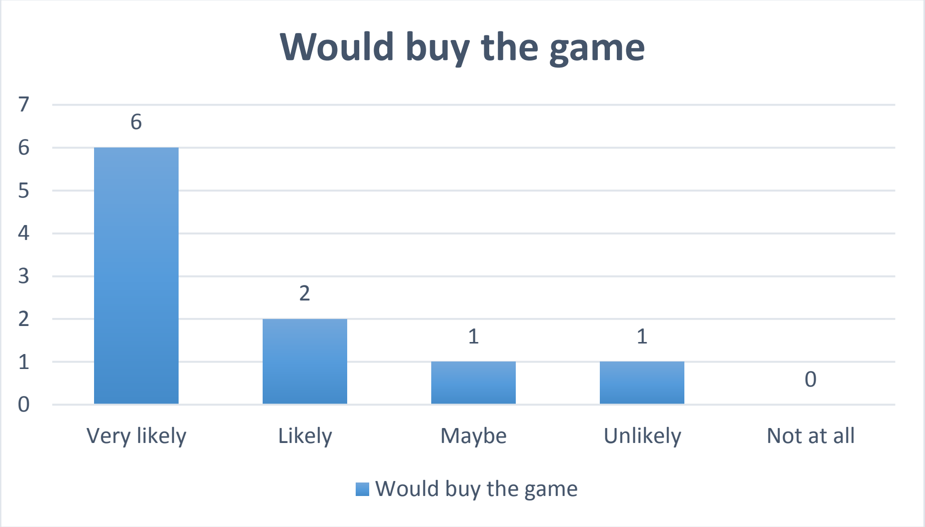


Did you enjoy the mixture of a classic platforming game with modern Role Playing Game elements (gaining experience points and levelling up your character)?



We see the response to the game was very positive with customers enjoying a wide variety of features with no stand out. This could be interpreted that the game was average across the board but also that due to the mixture of the two game genres that a certain demographic leaned to gameplay elements of a certain game. Which in the surveys in the appendix show that this is true, the female audience from varying age groups enjoyed the platforming of the game seen in older games and the male audience preferred boss battles and the levelling up features. There was also one survey that resulted in the customer enjoying an unnamed feature which was that the player would repeat the previous level and find access to new areas adding replay ability to the game. The exact comment was “Like that the first level had a locked door that could only be opened after beating the boss”.

Would you be interested in buying a complete version of this game based on what you have seen thus far?



This was a very positive result to see from the customers but unfortunately have one that would not buy the game. But on further investigation we see this person does not play games and this game did not persuade him begin doing so. A lot of the people enjoyed the game and would buy it but further surveys would need to be taken to see which platform they would prefer to see this product released on such as consoles; PlayStation 4, Xbox. Computers, mobile etc. As I currently plan on focusing on the computer and console market charging a higher cost which would be more acceptable by customers in the gaming market and then to further develop for mobile devices at a lower cost or a free to play model which is more commonly seen in the mobile market.

Any comments you would like to add?

“Enjoyed that I didn’t rush through the first level and killed all the enemies so defeating the boss was easier.”

“Liked the idea of adding the RPG elements to the a 2D platformer but would’ve like some more visual feedback that I was levelling up and my health and magic was increasing”

“Would like a timer on the levels with a high score table to make it more competitive with my friends”

“Would prefer more jumping challenges”

“Liked that it looked like terraria”

“Even though my favorurite part was the boss battle I feel it wasn’t as polished as the first level but overall great enjoyement”

“Didn’t really understand what I was suppose to do. Needed to be more user friendly for people like me that don’t really play games. Was thankful for the controller support which was another feature I really enjoyed as it made it simpler for me to get to grips with how to control the game.”

The comments were very useful on providing feedback on features that would like to be added and applauding the features they liked. Unfortunately, these features could not be all implemented in time for completion but will hopefully do so in the future. The pixel art was received well from a younger audience which was a concern but due to games like Terraria and Minecraft, pixel art in 2D and 3D is commonly seen in the gaming industry.

3. Conclusions

This project set out to accomplish in bringing an old game style of 2D platformer with elements in more common games seen in today's market; Role Playing Games. While the project did accomplish in having these elements integrated within the game successfully, the features could've been more intertwined together but within the schedule of the project there wasn't allowances to delve deeper in to this and get a perfect balance between the two without sacrificing other aspects of the game that were enjoyed by customers such as the boss battle.

There were some disadvantages of deciding to be more creative and innovative with the art of the game not using generic assets. This was that the animations could not be completed to a fuller extent on enemies but this was more so a time issue, as during this development time I had to learn about animation and feel confident in the animation I have shown mainly with the main character that I could complete the rest with ease.

Using Unity and learning C# has been a great experience throughout development and has giving me skills that would increase my chances of placing a job within the gaming industry that is currently growing in Ireland.

While I'm confident in what I have created is commercially viable in the market place and profitable as the development cost was minimal, I would like to continue building on the framework I have set up to include features mentioned in the feedback from the customer.

4. Further development or research

Blood Moon with further development and resources could lead to incorporating many elements mentioned from the feedback given in the customer survey, like a level timer which proved to be quite popular. This would then be developed further with an online element to compare your score with other players around the world. This would encourage players to build a social element around the game discussing the best routes to take to avoid enemies, platforming hazards, etc.

Creating more artwork and animations could be added to increase the visuals of the pixel art style I have chosen. Also multiple characters could be created with different abilities to add variety of gameplay. Have different attributes such as faster or slower, weaker or stronger etc. and mix these attributes around with various player characters to make a unique gameplay experience with each. This is a feature seen in multiple games to add replay value to the game.

Blood Moon would also be further developed for mobile devices, adding touch controls which would involve changing the GUI elements to better suit a touch interface. This would require further research in to design patterns to learn the best gestures to use for player inputs. The pay scheme could become freemium an option seen in my games where we could monetize levelling up your character.

A 3D version could be released with more art development time spending time learning 3D modelling and many of the scripts would function the same so changes programming would be minimal. Bring this one step further Virtual reality could be looked in to, it’s a new market and would need to be investigated further to see it there is a potential market.

5. References

Bibliography: Technologies, U. (2016) *Unity - learn - modules*. Available at: http://unity3d.com/learn/tutorials (Accessed: 12 October 2015). **In-line Citation:**(Technologies, 2015)

Bibliography: Technologies, U. (2016) *Unity - Scripting API:* Available at: http://docs.unity3d.com/ScriptReference/index.html (Accessed: 20 October 2015).**In-line Citation:**(Technologies, 2016)

Bibliography:*Unity community* (2016) Available at: http://forum.unity3d.com/ (Accessed: 20 October 2015).**In-line Citation:**(*Unity community*, 2016)

Bibliography:*GucioDevs* (no date) Available at: https://www.youtube.com/playlist?list=PLq3pyCh4J1B2va_ftIthSpUaQH0LycRA- (Accessed: 3 November 2015).**In-line Citation:**(*GucioDevs*, no date)

Bibliography:*Achebit* (no date) Available at: https://www.youtube.com/channel/UCGOJ92rTGadpUha61KrqrLQ (Accessed: 10 November 2015).**In-line Citation:**(*Achebit*, no date)

Bibliography:*C sharp accent tutorials* (no date) Available at: https://www.youtube.com/playlist?list=PL1bPKmY0c-wl6LKOiySAZhVoF9pDevFQv (Accessed: 22 February 2016).**In-line Citation:**(*C sharp accent tutorials*, no date)

Bibliography:gamesplusjames (2015) *Unity RPG Tutorial #1 - setting up the basics*. Available at: https://www.youtube.com/watch?v=Pk3GCgaNVTY&list=PLiyfvmtjWC_X6e0EYLPczO9tNCkm2dzkm (Accessed: 4 March 2016).**In-line Citation:**(gamesplusjames, 2015)

Bibliography: Astrauk (2010) *Example of using player Prefs to save and get a variable*. Available at: http://forum.unity3d.com/threads/example-of-using-player-prefs-to-save-and-get-a-variable.42398/ (Accessed: 29 April 2016).**In-line Citation:**(Astrauk, 2010)

Bibliography: Scout, H. (2013) *10 essential tactics for creating valuable customer surveys*. Available at: https://www.helpscout.net/blog/customer-survey/ (Accessed: 1 May 2016).**In-line Citation:**(Scout, 2013)

6. Appendix

6.1. Project Proposal

Objectives

The objectives of this project is to develop a fully functional 2D action platform game. The game will contain most features that are expected of a platform game but focus mostly on combat. The game will be easy to learn and play, with simple controls and functions.

The game will begin with an optional tutorial before they start playing allowing the player to become familiar with the controls and the GUI. This will take place in a safe environment and can be accessed at any time for the player in the menu if they need to refresh their memory of the controls or GUI. The aim of this is mainly to show the player what differences they are from the standard platform and what they share in common.

The game will encourage players to play and complete the game in a main story mode but upon completion, the game will have a level select screen allowing players to choose whatever level they want to play with the option of speed challenge to complete the level as fast as possible.

Each level of the game in the story will have a unique function in the environment such as particular environment hazards to keep each level distinct and refreshing to the user but also using the same controls to prevent a steep learning curve. Each of these bosses will have a unique boss at the end that will beatable by using attacks along with the unique function of the environment.

The game will have a mini games section where it'll have challenges that will use the same assets from the main campaign to save time and use them in different ways to add more content and variety for the player.

I hope to have the game available on as many platforms as possible but my main focus will be to have it working on PC.

Background

The gaming industry has had a renaissance of such due to the rise of the indie developers. These developers are providing a variety of old school classics in the vein of Castlevania and Metroid to innovative original concepts that would be considered too risky finically for big studios. For example, No Man Sky, where players are free to explore the entirety of a procedurally generated open universe, which includes over 18 quintillion planets each with their own set of creatures and locations to discover. These days it doesn't take a team of thousands to create a successful game, with freely and easily accessible tools such as Unity, anyone with a passion and desire to be creative can be successful in creating the game.

A traditional platforming game involves guiding an avatar to jump between suspended platforms, floors, ledges, stairs over obstacles or a mixture to advance in the game, these challenges are commonly referred to as jumping puzzles. These games became widely popular and dominated in the early years of the game industry due to games like Super Mario and Sonic. These games evolved as technology advanced along with graphics and 3D platformers became common place but over the last few years the genre has seen a significant resurgence in the traditional 2D side scrolling sub-genre, due to the downloadable marketplace.

Being one of the earliest, and accessible genre of games it was very interesting to investigate how these types of games work on a technical level, as well as exploring new and innovative ideas developers have created to keep the genre fresh, and create sub genres of the game type but also keeping the key elements in what makes these games so successful and accessible to any consumer.

Technical Approach

Research:

The research and study that went into the pre-development of this game was extensive. It begun looking at games that I found had the most enjoyable gameplay and interesting visual style. The games I mainly looked at for inspiration and investigated were the following:

Guacamelee! - a 2D side-scroller where the combat system consists of using basic attacks and can perform various grapple attacks when they are stunned, dealing more damage or throwing them towards other enemies thus creating combos.

Shovel Knight - 2D side-scrolling platform game presented in 8-bit graphics. The player can purchase secondary items that can be used with a limited supply of magic. These include long range projectiles, gloves that can punch through dirt blocks, and the ability to make you invincible for a brief period.

I then looked online using Google, YouTube and Pluralsight for how to approach what I wanted in my game and see what is achievable within my time line. I then decided on using Unity as it seems the most user friendly software available and a vast library of tools that support the project and ensure its completed. This research helped me decide on all the functions I want in my game and how I can successfully carry them out within the time frame.

Implementation:

The implementation of this project will be carried out with the use of C# and within the Unity Game Engine. I aim to have several stages of development over the year.

The art and animations of the game by the end of the first semester will not be finalised as they will be developed along with other parts of the game. I will start by coding the basic logic for the game and implementing the basic features that I want to appear within the game such as jumping and combat. I expect to have this done by the end of semester one and then test the game myself or with the assistance of alpha testers to find bugs and optimise gameplay.

In the second semester I will begin focusing on the environment, including artwork and how their mechanics. This will also be worked on along with bosses. Having completed this, I will beta test all of the created content for bugs or errors and begin polishing off the game. I expect to have completed this stage midway through semester two.

Close to the end of semester two I will have the game tested by final testers to fine tune anything that may need it.

Special resources required

If applicable, e.g., books, hardware, etc.
Keyboard
Mouse
Graphics tablet
PS4 controller

Project Plan

Please see attached file 'Project Plan.mpp'.

Technical Details

- Unity Game Engine
  - Is a cross-platform game engine used to develop video games for PC, consoles, mobile devices and websites. Unity will be my primary tool in development of my project. I am currently using the latest version Unity 5.2.1.
- MonoDevelop (cross platform IDE)
  - Is a free GNOME IDE primarily designed for C#. A customised version of MonoDevelop ships with Unity.
- C#
  - The standard language that is used in the creation of games developed with Unity. C# has been relatively easy to learn thus far. It is very similar to other object-oriented programming languages such as Java. It only took a short amount of time to become familiar with C#. This represents a primary reason why Unity was chosen for the project over Unreal Engine.
- Adobe Illustrator
  - An easy to use vector art creator that'll assist me in creating the assets for my project such as characters, environment etc. to achieve the look and style I want.

Evaluation

To evaluate the game, I will have 4 stages, two main testing stages followed by a survey and finally a comparison.

Alpha Test (stage one)

- The first test will happen at the beginning of December, exactly one month before midpoint presentation. I will need outside testers such as college colleagues to receive their input and feedback on what I have created so far. I'll compare what feedback from the individuals and see what area needs most focus and fine tuning or if a gameplay mechanic is not achieving the reaction I was expecting it will be improved or edited to a new idea if need be.

Beta Test (stage two)

- Showing the changes that have been made between the alpha and beta test to the same testers. I'll hopefully extend my testers list and ask them individually to specifically focus on certain areas that I found they had the most problems in the alpha test. This is oppose to the previous test were I allowed them to play alpha build entirely. I expect only minor changes to be made from user feedback and input.

Survey (stage three)

- I will create a survey with Survey Monkey and send them out using my NCI student email and ask about what they think of my project as a whole and how they compare with games they enjoy.

Comparison (stage four)

- My final version of my project will be compared to games that have inspired me and also to games that are currently out in the market. I will see where I have improved on what has come before and see if there is anything similar in terms of visuals or mechanics to what I hope to have achieved.

Ricardo O'Hara Camones   2/10/2015

Signature of student and date

6.2. Project Plan

Please see attached file 'Project Plan.mpp'.

6.3. *Monthly Journals*

6.3.1. September

Reflective Journal

Student name: Ricardo O'Hara Camones Student Number: x12425828

Programme: BSc in Computing

Month: September

My Achievements

This month, I was able to begin the required research needed for the pre-development of my game.

It begun looking at vast library of games that I found had the most enjoyable gameplay and interesting visual style. Then I moved on to looking at many aspects of game production for the genre of the game I wish to make such as, game AI, design, engines, programming, UI and level design.

My contributions to the projects included deciding on what genre I want my game to be in, game mechanics and the visual style I wish to adhere to.

My Reflection

I felt, it worked well to make me understand what I can accomplish within my time limit and make my expectations realistic.

However, I was not successful in diving in to my actual project as much as I hoped as documentation for the project took up quite a lot of unnecessary amount of my time. Some aspects of the documentation were important in making decisions for my project but other aspects seem to be filler content and had no purpose in my opinion.

Intended Changes

Next month, I will try to delve deeper in to the development of my project as documentation will not interfere as much in the following month than previously. I'll begin with working on level design so I can visually picture what the final product will look like. From there my main focus will move to where the majority of work will be for the several months, graphics and programming.

I realised that I need to add more dedicated time to my originally planned scheduled to make up for the lack of work I did not accomplish this month as I had hopped. I will also remember for further documentation that it may take up more time than estimated and best to finish it far earlier than the due date.

Supervisor Meetings

My supervisor has not been assigned to me yet, I hope for one that is enthusiastic as I am about gaming. I look forward to discussing my project with them and showing them my progress over the following months.

6.3.2. October

Reflective Journal

Student name: Ricardo O'Hara Camones Student Number: x12425828

Programme: BSc in Computing

Month: October

My Achievements

This month, I was able to begin on creating some of the assets for my project such as the the protagonist and the environment. Before any of these begun I started with level design. I spent several days on perfecting my first level that also function as a tutorial level. I made sure all the basic mechanics would be covered by providing scenarios where only certain function has to be used to pass on to the next section.

My Reflection

I was not successful in diving in to my actual project as much as I hoped I would in comparison to last month as assignments and CAs took place throughout the month. I can see that a lot of this project will be done during my time of during the Christmas break as I am required a to complete a lot of assignments from others modules.

Intended Changes

Next month, I will try to delve deeper in to the development of my project as documentation will not interfere as much in the following month than previously. I'll begin with working on character movement and fully create the first level using some of the assets I have created already and what I will continue to do so through the month.

I realised that I need to add more dedicated time to my rescheduled planned schedule to make up for the lack of work I did not accomplish the past two months as I had hopped.

Supervisor Meetings

I had my first meeting with with my supervisor Anu Sahni where we discussed what I intend my game to achieve functionally and visually. This expanded in going in to specify the unique environmental mechanics that will be in each level.

6.3.3. November

Reflective Journal

Student name: Ricardo O'Hara Camones Student Number: x12425828

Programme: BSc in Computing

Month: November

My Achievements

This month, I was able to begin creating character controls for my protagonist. Using the sprite sheet I had created previously I was able to create an animation of running using Unity and then along with MonoDevelop I was able to add in controls to move my character around. I then continued to add in 2D physics to the character, such as Rigidbody. This would allow the character to collide and falloff environmental assets.

My Reflection

I still feel that I was not successful in progressing in to my actual project as much as I hoped. I can see that a lot of this project will be done during my time of during the Christmas break as I am required a to complete a lot of assignments from others modules.

Intended Changes

Next month, I will have finished semester one and be free from assignments and CA for a while before I have to study for my exams. During this time I will dive deeper in to the development of my project as other time consuming modules will no longer interfere. I'll continue on complete the controls for my main character, such as including jumping and attacking. I will have the first level design completed and created in Unity with placeholder assets as I feel it's too early on in development to decide on a final design and would only end up on me spending time redesigning assets to a finalised stage and then changing my mind.

Supervisor Meetings

I had my first meeting with with my supervisor Anu Sahni last month where we discussed what I intend my game to achieve and what I hoped to accomplish in time for our next supervisor meeting but I did not feel confident in my progress that I reached an important milestone so I didn't schedule another meeting.

6.3.4. December

Reflective Journal

Student name: Ricardo O'Hara Camones Student Number: x12425828

Programme: BSc in Computing

Month: December

My Achievements

This month, I was able to begin creating several scripts in MonoDevelop such as PlayerControl, InstaDeath and Checkpoint which give the user control over the protagonist to perform platforming and movement such as double jump and running, killing a character instantly for being in contact with a hazardous asset, such as fire and poisonous gas and creating respawn locations for when the protagonist dies, respectively. I've also accomplished a temporary camera control system that follows the protagonist along to keep them in the screen.

My Reflection

As expected within a short amount of time I have made successful progress in my project without the distraction of documentation that has plagued me during the semester. As my exams are quickly approaching I will need to pause production until my exams are completed and only then can I continue working on my project and have a prototype available for demonstration come February.

Intended Changes

I intend to make several advances in my project between the end of my exams and my February prototype demonstration. These advances will consist of various elements such as enemies/enemy AI, advance camera controls, health system, sound effects, pickups, scoring, attacking and defending.

I'll also continue on finalising assets such as environment, enemies, protagonist, bosses etc. as per usual.

Supervisor Meetings

Had an unsuccessful meeting with my supervisor Anu Sahni as I had not succeeded in what I intended to accomplish and I could see that she was questioning whether my idea would be successful in any sense of the word.

We’ve scheduled a meeting a week before returning to college to see my progress and I’m currently on track to what I said I would be able to do and then some. I hope this will diminish any concerns she had about my project.

6.3.5. January

Reflective Journal

Student name: Ricardo O'Hara Camones Student Number: x12425828

Programme: BSc in Computing

Month: January

My Achievements

This month, I was able to accomplish more than I have in previous months. I achieved most of what I intended to such as enemies/ enemy AI, where the "Ghoul" patrols an area and when he comes in to contact with a wall and will turn around the same will if the edge of a platform is detected or not detected as that is how I have set up the script he will turn around. The "Ghoul" also has a health system and once is health reaches zero is destroyed and activates his death animation. Advance camera controls are in place from previous camera controls were I simply made the camera a child of the player, but problems occurred when the player turned back which also flipped the camera's z axis. Health system was accomplished along with a health bar using a slider in Unity and an image placed over it to give it a more unique design. This also saved me from creating multiple images of the health bar at various stages of health. Attacking enemies with a fireball that has been animated by myself, which has a collision detection to explode on contact using particle effects and take health away from an enemy.

I also accomplished others things that I did not plan to originally, but were connected to what I was working on and was convenient to work on, such as a damage system that was connected to my fireball to damage enemies on impact and also for when the "Ghoul" contacts the player. Death and respawn animations were created using particles in Unity, that work for both enemies and the player character and a lot of artwork for the game was accomplished.

I discovered an issue that has plagued me from the beginning as to why my assets were blurry in the game, but this was due to the default setting in Unity, which I was lucky to find a solution to as many online forums I discovered suggested drastic measures to remove the blur which would've been severely time consuming.

My Reflection

As before within a short amount of time, this past month I have accomplished more than I have in the several months before. Unfortunately, in the past week my progress has slowed down due to college which is acceptable but when I am free I have to work on more documentation for this project which I find is important but is not needed until towards the end. I'm writing on my game in its current stage but as I continue my plans change due to learning more and understanding more about the game engine than before. So then I have to rewrite what I have already done which accomplishes nothing but wasting my time and hindering the potential of my project and from my understanding talking to my fellow class mates I am not the only one that feels this way.

Intended Changes

I intend to make several advances in my project in the next month as before. These advances will consist of various elements such as flying enemies/AI which will add more diversity to the gameplay and will make the player consider hazards just not from the same level or below but above in the skies. A knock back effect for when a player takes damage cause currently when he takes damage there is no effect or indication visually that the player has taken damage apart from the health bar. It'll make the player also take in to account that an enemy could attack them and push them off the edge to their death.

More interactive environments such as claimable ladders and movable platforms. Hopefully have destructible crates that could potentially explode and do damage to enemies and the player themselves and make the player more cautious when approaching and attacking an enemy.

New attacks and defences will be added such as sword attacks, blocking and rolling to evade enemy attack and also add another dimension to manoeuvres.

As always I continue to work on designing and creating assets such as environment and enemies, along with the animations with them to bring these characters to life.

Supervisor Meetings

Had a short meeting with my supervisor Anu Sahni discussing my midpoint presentation and showing her my progress so far. She guided me on what to prepare and what to mention in my presentation, to impress as much as I can and receive a high grade, such as showing stuff that are in development but not completed which I would've previously thought against doing.

6.3.6. February

Reflective Journal

Student name: Ricardo O'Hara Camones Student Number: x12425828

Programme: BSc in Computing

Month: February

My Achievements

This month I was able to accomplish most of what I intended to, such as a flying enemy, a knock back affect to stagger the enemy after being damaged, interactive environments that include, climbable ladders, moving platforms.

What I managed to accomplish that I did not initially intend to do was a leveling system, where the player gains XP (experience points) for killing enemies. A mana bar (magic usage system) that will decrease when magic is used, such as fire bolt and slowing down time. The mana bar also regenerates over time.

As always I continued to work on designing and creating assets such as environment and enemies, along with the animations with them to bring these characters to life.

My Reflection

Unfortunately, I did not accomplish all I intended to this month, such as adding more attacks and defenses to the player. I know how to accomplish these with code but have yet to begin the animations for them. Once these are done though all my character movements will be done.

Intended Changes

I intend to make several advances in my project in the next month as before. These advances will consist of programming more enemy AI, along with the boss AI, which I have begun designing and coding. To save time the boss will be a head and just two arms, saving me from creating several animation frames.

These will be completed after project upload, as I know animations will not be key to receiving a high grade in my project. I continue my wish on releasing my game on Steam like many independent developers before me.

Supervisor Meetings

Had a short meeting with my supervisor Anu Sahni discussing my midpoint presentation and where I can improve on my 72%.

6.3.7. March

Reflective Journal

Student name: Ricardo O'Hara Camones Student Number: x12425828

Programme: BSc in Computing

Month: March

My Achievements

This month I was able to accomplish most of what I intended to, such as another enemy type with a unique AI, along with improving the AI of my previously created enemies.

As always I continued to work on designing and creating assets such as environment and enemies, along with the animations with them to bring these characters to life.

My Reflection

Unfortunately, I did not accomplish all I intended to this month, such as beginning work on my boss AI but through learning more about AI and creating different types to add more challenge to the game I have a better knowledge on how creating the boss AI I want.

Intended Changes

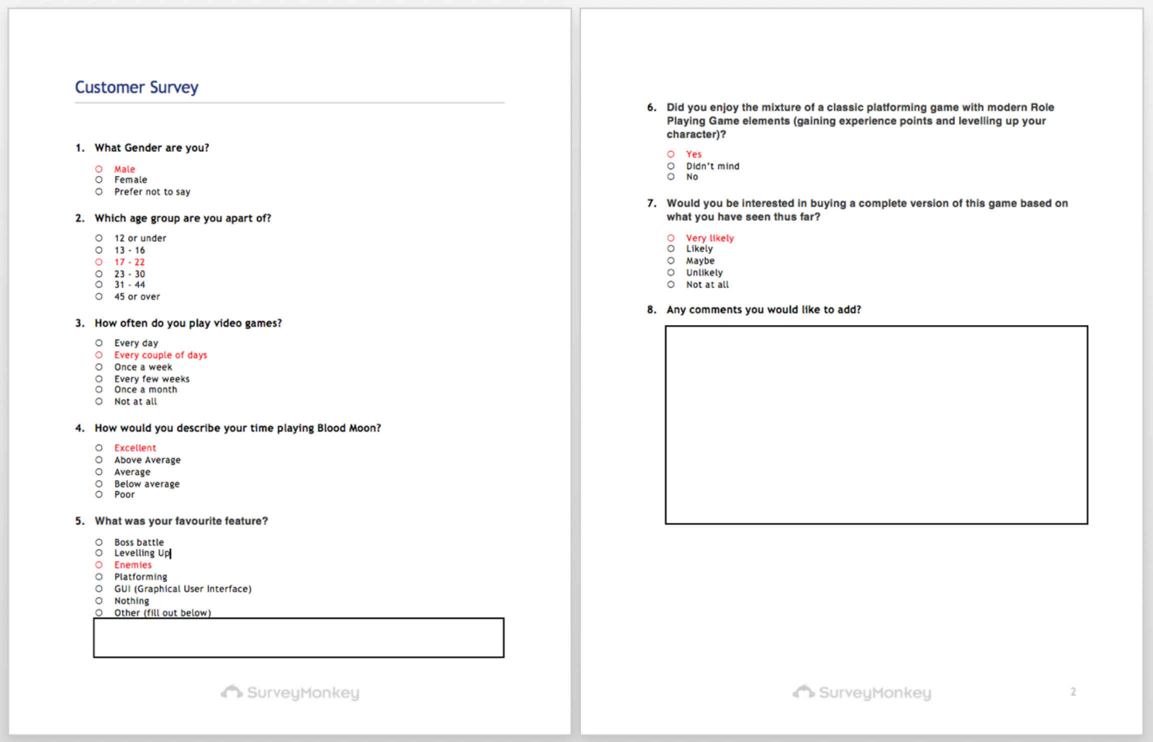
I intend to make several advances in my project in the next month, but these will likely be the final additions to my project. These advances will consist of programming more enemy AI, along with the boss AI. Complete the first level and be playable. Have a save and load system working, this is vital to have in my finished game.

Supervisor Meetings

Did not have any meeting with my supervisor this month.

6.4. Other Material Used

6.5. Customer Surveys



Customer Survey

1. What Gender are you?

○ [✓ Male]
○ Female
○ Prefer not to say

2. Which age group are you apart of?

○ 12 or under
○ 13 - 16
○ [✓ 17 - 22]
○ 23 - 30
○ 31 - 44
○ 45 or over

3. How often do you play video games?

○ Every day
○ [✓ Every couple of days]
○ Once a week
○ Every few weeks
○ Once a month
○ Not at all

4. How would you describe your time playing Blood Moon?

○ [✓ Excellent]
○ Above Average
○ Average
○ Below average
○ Poor

5. What was your favourite feature?

○ Boss battle
○ Levelling Up
○ [✓ Enemies]
○ Platforming
○ GUI (Graphical User Interface)
○ Nothing
○ Other (fill out below)

6. Did you enjoy the mixture of a classic platforming game with modern Role Playing Game elements (gaining experience points and levelling up your character)?

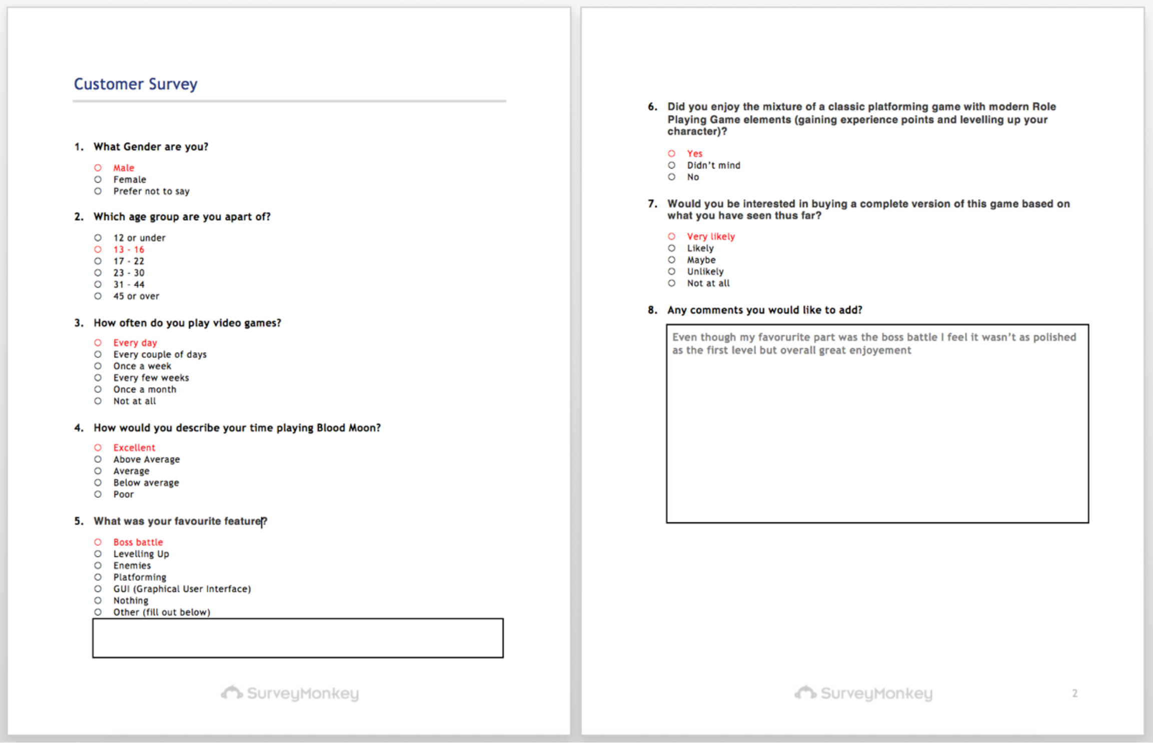
○ [✓ Yes]
○ Didn't mind
○ No

7. Would you be interested in buying a complete version of this game based on what you have seen thus far?

○ [✓ Very likely]
○ Likely
○ Maybe
○ Unlikely
○ Not at all

8. Any comments you would like to add?

2



Customer Survey

1. What Gender are you?

○ [✓ Male]
○ Female
○ Prefer not to say

2. Which age group are you apart of?

○ 12 or under
○ [✓ 13 - 16]
○ 17 - 22
○ 23 - 30
○ 31 - 44
○ 45 or over

3. How often do you play video games?

○ [✓ Every day]
○ Every couple of days
○ Once a week
○ Every few weeks
○ Once a month
○ Not at all

4. How would you describe your time playing Blood Moon?

○ [✓ Excellent]
○ Above Average
○ Average
○ Below average
○ Poor

5. What was your favourite feature?

○ [✓ Boss battle]
○ Levelling Up
○ Enemies
○ Platforming
○ GUI (Graphical User Interface)
○ Nothing
○ Other (fill out below)

6. Did you enjoy the mixture of a classic platforming game with modern Role Playing Game elements (gaining experience points and levelling up your character)?

○ [✓ Yes]
○ Didn't mind
○ No

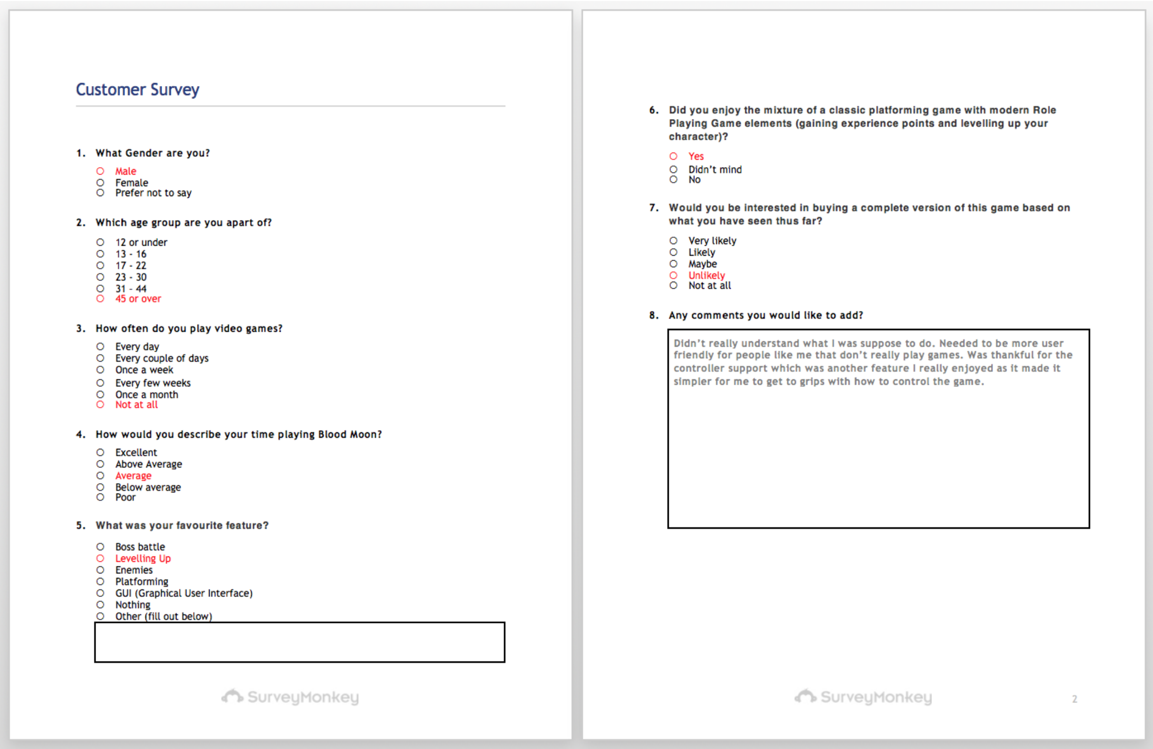
7. Would you be interested in buying a complete version of this game based on what you have seen thus far?

○ [✓ Very likely]
○ Likely
○ Maybe
○ Unlikely
○ Not at all

8. Any comments you would like to add?

Even though my favorurite part was the boss battle I feel it wasn't as polished as the first level but overall great enjoyement

2



Customer Survey

1. **What Gender are you?**
   - ○ [✓ Male]
   - ○ Female
   - ○ Prefer not to say

2. **Which age group are you apart of?**
   - ○ 12 or under
   - ○ 13 - 16
   - ○ 17 - 22
   - ○ 23 - 30
   - ○ 31 - 44
   - ○ [✓ 45 or over]

3. **How often do you play video games?**
   - ○ Every day
   - ○ Every couple of days
   - ○ Once a week
   - ○ Every few weeks
   - ○ Once a month
   - ○ [✓ Not at all]

4. **How would you describe your time playing Blood Moon?**
   - ○ Excellent
   - ○ Above Average
   - ○ [✓ Average]
   - ○ Below average
   - ○ Poor

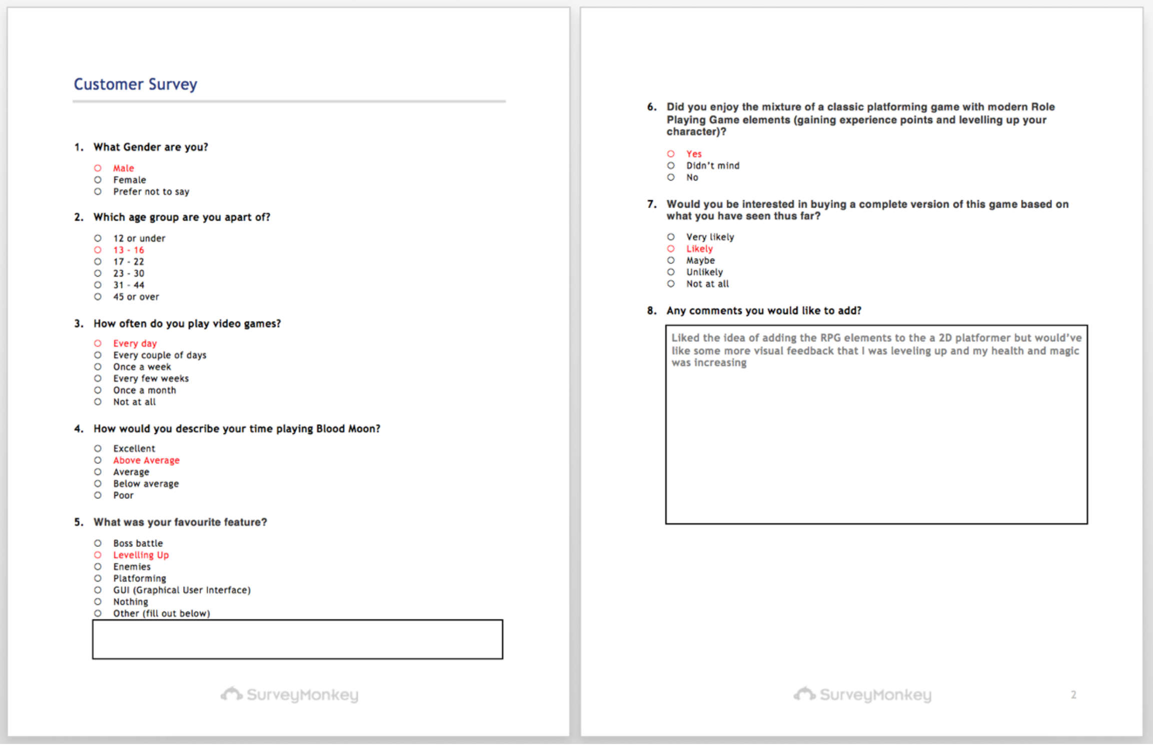
5. **What was your favourite feature?**
   - ○ Boss battle
   - ○ [✓ Levelling Up]
   - ○ Enemies
   - ○ Platforming
   - ○ GUI (Graphical User Interface)
   - ○ Nothing
   - ○ Other (fill out below)

6. **Did you enjoy the mixture of a classic platforming game with modern Role Playing Game elements (gaining experience points and levelling up your character)?**
   - ○ [✓ Yes]
   - ○ Didn't mind
   - ○ No

7. **Would you be interested in buying a complete version of this game based on what you have seen thus far?**
   - ○ Very likely
   - ○ Likely
   - ○ Maybe
   - ○ [✓ Unlikely]
   - ○ Not at all

8. **Any comments you would like to add?**

Didn't really understand what I was suppose to do. Needed to be more user friendly for people like me that don't really play games. Was thankful for the controller support which was another feature I really enjoyed as it made it simpler for me to get to grips with how to control the game.



Customer Survey

1. **What Gender are you?**
   - ○ [✓ Male]
   - ○ Female
   - ○ Prefer not to say

2. **Which age group are you apart of?**
   - ○ 12 or under
   - ○ [✓ 13 - 16]
   - ○ 17 - 22
   - ○ 23 - 30
   - ○ 31 - 44
   - ○ 45 or over

3. **How often do you play video games?**
   - ○ [✓ Every day]
   - ○ Every couple of days
   - ○ Once a week
   - ○ Every few weeks
   - ○ Once a month
   - ○ Not at all

4. **How would you describe your time playing Blood Moon?**
   - ○ Excellent
   - ○ [✓ Above Average]
   - ○ Average
   - ○ Below average
   - ○ Poor

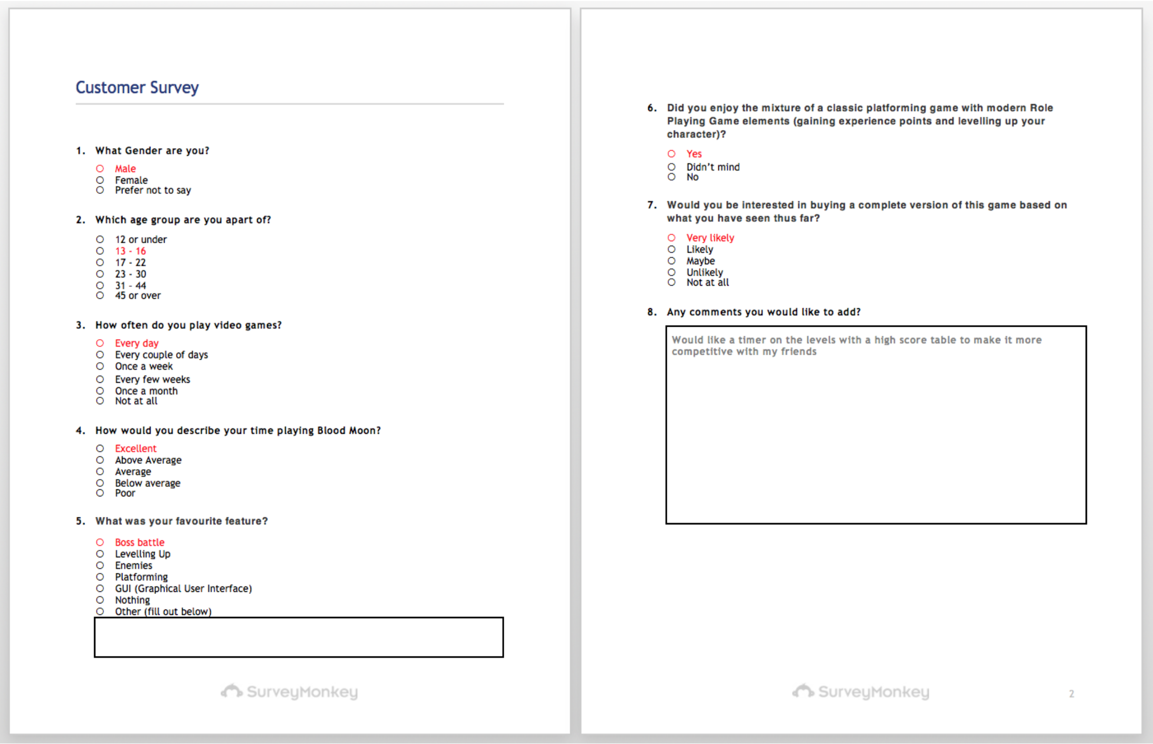
5. **What was your favourite feature?**
   - ○ Boss battle
   - ○ [✓ Levelling Up]
   - ○ Enemies
   - ○ Platforming
   - ○ GUI (Graphical User Interface)
   - ○ Nothing
   - ○ Other (fill out below)

6. **Did you enjoy the mixture of a classic platforming game with modern Role Playing Game elements (gaining experience points and levelling up your character)?**
   - ○ [✓ Yes]
   - ○ Didn't mind
   - ○ No

7. **Would you be interested in buying a complete version of this game based on what you have seen thus far?**
   - ○ Very likely
   - ○ [✓ Likely]
   - ○ Maybe
   - ○ Unlikely
   - ○ Not at all

8. **Any comments you would like to add?**

Liked the idea of adding the RPG elements to the a 2D platformer but would've like some more visual feedback that I was leveling up and my health and magic was increasing



Customer Survey

1. **What Gender are you?**

○ [✓ Male]
○ Female
○ Prefer not to say

2. **Which age group are you apart of?**

○ 12 or under
○ [✓ 13 - 16]
○ 17 - 22
○ 23 - 30
○ 31 - 44
○ 45 or over

3. **How often do you play video games?**

○ [✓ Every day]
○ Every couple of days
○ Once a week
○ Every few weeks
○ Once a month
○ Not at all

4. **How would you describe your time playing Blood Moon?**

○ [✓ Excellent]
○ Above Average
○ Average
○ Below average
○ Poor

5. **What was your favourite feature?**

○ [✓ Boss battle]
○ Levelling Up
○ Enemies
○ Platforming
○ GUI (Graphical User Interface)
○ Nothing
○ Other (fill out below)

6. **Did you enjoy the mixture of a classic platforming game with modern Role Playing Game elements (gaining experience points and levelling up your character)?**

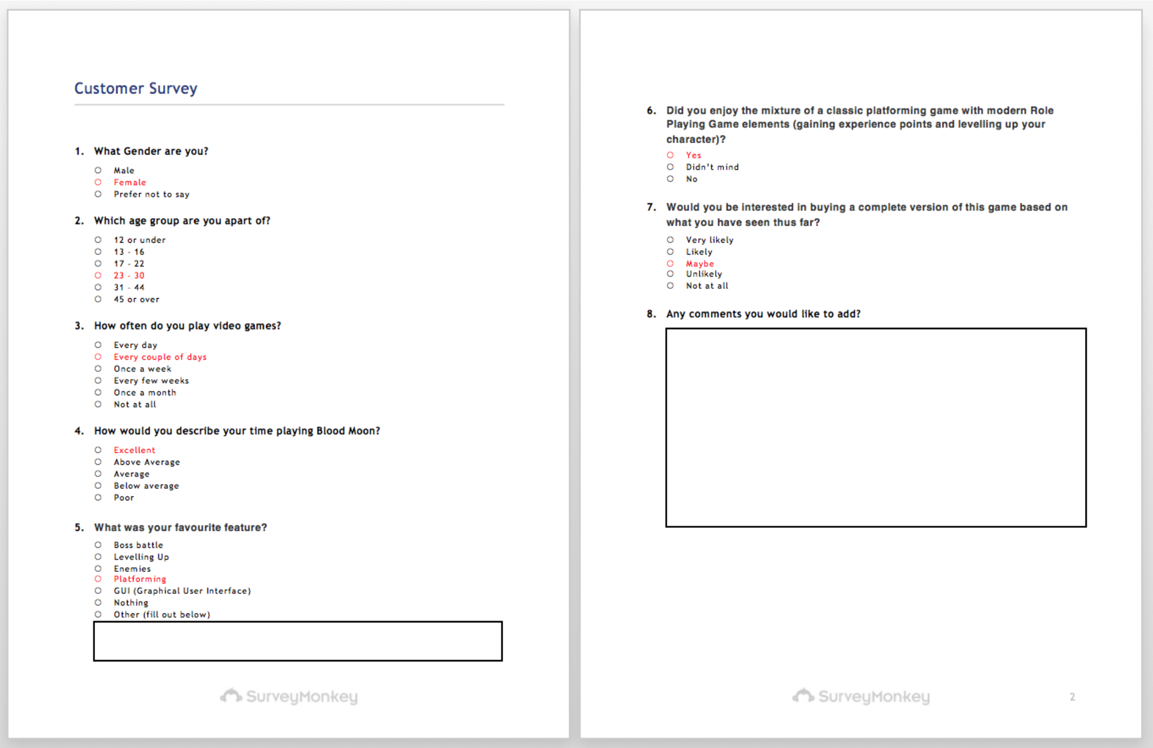
○ [✓ Yes]
○ Didn't mind
○ No

7. **Would you be interested in buying a complete version of this game based on what you have seen thus far?**

○ [✓ Very likely]
○ Likely
○ Maybe
○ Unlikely
○ Not at all

8. **Any comments you would like to add?**

Would like a timer on the levels with a high score table to make it more competitive with my friends



Customer Survey

1. **What Gender are you?**

○ Male
○ [✓ Female]
○ Prefer not to say

2. **Which age group are you apart of?**

○ 12 or under
○ 13 - 16
○ 17 - 22
○ [✓ 23 - 30]
○ 31 - 44
○ 45 or over

3. **How often do you play video games?**

○ Every day
○ [✓ Every couple of days]
○ Once a week
○ Every few weeks
○ Once a month
○ Not at all

4. **How would you describe your time playing Blood Moon?**

○ [✓ Excellent]
○ Above Average
○ Average
○ Below average
○ Poor

5. **What was your favourite feature?**

○ Boss battle
○ Levelling Up
○ Enemies
○ [✓ Platforming]
○ GUI (Graphical User Interface)
○ Nothing
○ Other (fill out below)

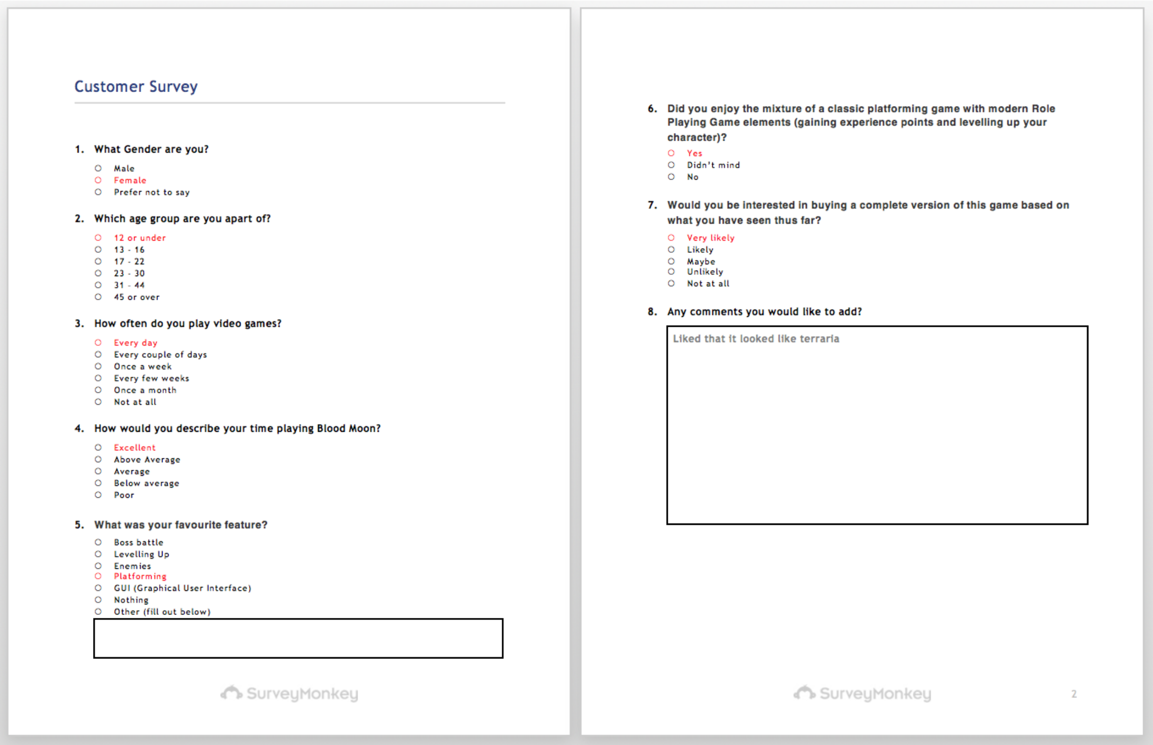
6. **Did you enjoy the mixture of a classic platforming game with modern Role Playing Game elements (gaining experience points and levelling up your character)?**

○ [✓ Yes]
○ Didn't mind
○ No

7. **Would you be interested in buying a complete version of this game based on what you have seen thus far?**

○ Very likely
○ Likely
○ [✓ Maybe]
○ Unlikely
○ Not at all

8. **Any comments you would like to add?**

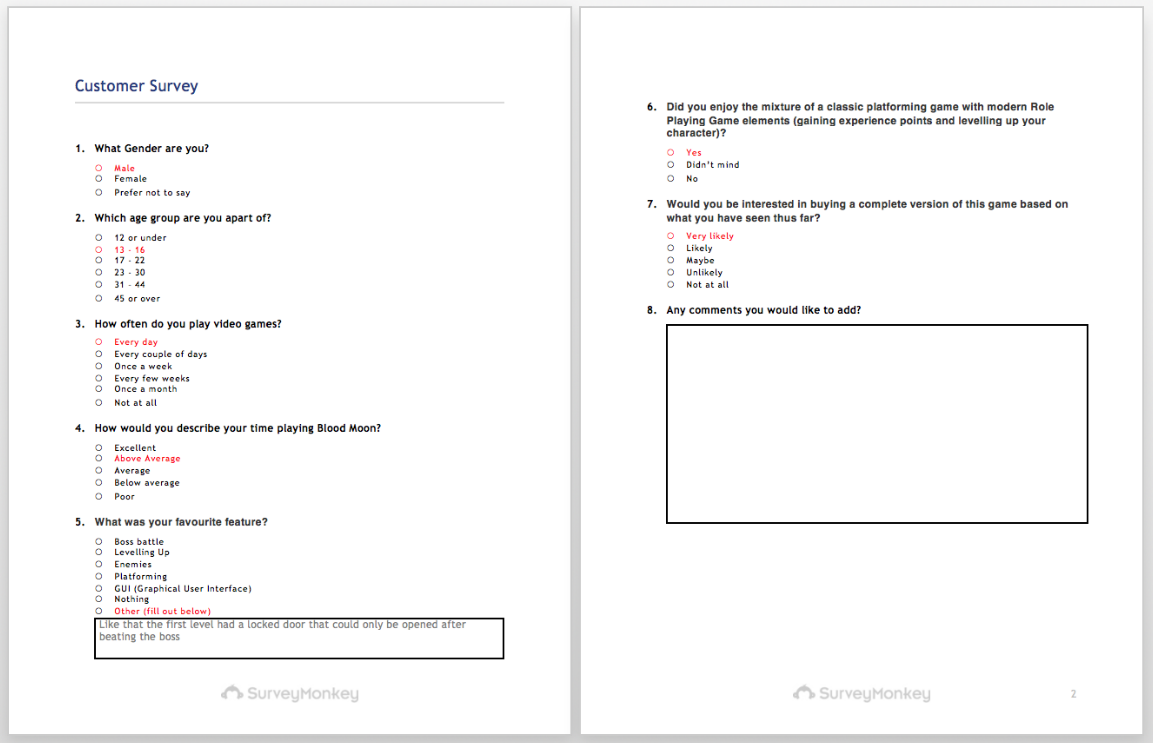


Customer Survey

1. **What Gender are you?**
   - ○ Male
   - ○ [✓ Female]
   - ○ Prefer not to say
2. **Which age group are you apart of?**
   - ○ [✓ 12 or under]
   - ○ 13 - 16
   - ○ 17 - 22
   - ○ 23 - 30
   - ○ 31 - 44
   - ○ 45 or over
3. **How often do you play video games?**
   - ○ [✓ Every day]
   - ○ Every couple of days
   - ○ Once a week
   - ○ Every few weeks
   - ○ Once a month
   - ○ Not at all
4. **How would you describe your time playing Blood Moon?**
   - ○ [✓ Excellent]
   - ○ Above Average
   - ○ Average
   - ○ Below average
   - ○ Poor
5. **What was your favourite feature?**
   - ○ Boss battle
   - ○ Levelling Up
   - ○ Enemies
   - ○ [✓ Platforming]
   - ○ GUI (Graphical User Interface)
   - ○ Nothing
   - ○ Other (fill out below)
6. **Did you enjoy the mixture of a classic platforming game with modern Role Playing Game elements (gaining experience points and levelling up your character)?**
   - ○ [✓ Yes]
   - ○ Didn't mind
   - ○ No
7. **Would you be interested in buying a complete version of this game based on what you have seen thus far?**
   - ○ [✓ Very likely]
   - ○ Likely
   - ○ Maybe
   - ○ Unlikely
   - ○ Not at all
8. **Any comments you would like to add?**

Liked that it looked like terraria

2



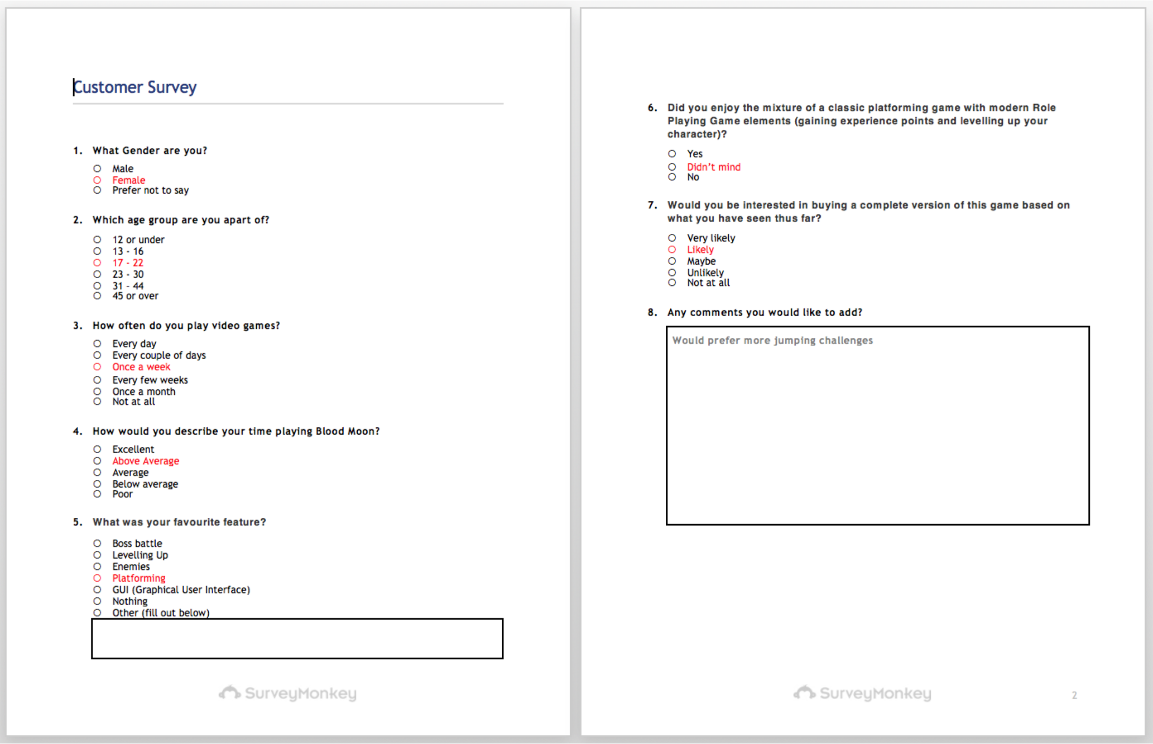
Customer Survey

1. **What Gender are you?**
   - ○ [✓ Male]
   - ○ Female
   - ○ Prefer not to say
2. **Which age group are you apart of?**
   - ○ 12 or under
   - ○ [✓ 13 - 16]
   - ○ 17 - 22
   - ○ 23 - 30
   - ○ 31 - 44
   - ○ 45 or over
3. **How often do you play video games?**
   - ○ [✓ Every day]
   - ○ Every couple of days
   - ○ Once a week
   - ○ Every few weeks
   - ○ Once a month
   - ○ Not at all
4. **How would you describe your time playing Blood Moon?**
   - ○ Excellent
   - ○ [✓ Above Average]
   - ○ Average
   - ○ Below average
   - ○ Poor
5. **What was your favourite feature?**
   - ○ Boss battle
   - ○ Levelling Up
   - ○ Enemies
   - ○ Platforming
   - ○ GUI (Graphical User Interface)
   - ○ Nothing
   - ○ [✓ Other (fill out below)]

Like that the first level had a locked door that could only be opened after beating the boss

6. **Did you enjoy the mixture of a classic platforming game with modern Role Playing Game elements (gaining experience points and levelling up your character)?**
   - ○ [✓ Yes]
   - ○ Didn't mind
   - ○ No
7. **Would you be interested in buying a complete version of this game based on what you have seen thus far?**
   - ○ [✓ Very likely]
   - ○ Likely
   - ○ Maybe
   - ○ Unlikely
   - ○ Not at all
8. **Any comments you would like to add?**

2



Customer Survey

1. **What Gender are you?**
   - ○ Male
   - ○ [✓ Female]
   - ○ Prefer not to say

2. **Which age group are you apart of?**
   - ○ 12 or under
   - ○ 13 - 16
   - ○ [✓ 17 - 22]
   - ○ 23 - 30
   - ○ 31 - 44
   - ○ 45 or over

3. **How often do you play video games?**
   - ○ Every day
   - ○ Every couple of days
   - ○ [✓ Once a week]
   - ○ Every few weeks
   - ○ Once a month
   - ○ Not at all

4. **How would you describe your time playing Blood Moon?**
   - ○ Excellent
   - ○ [✓ Above Average]
   - ○ Average
   - ○ Below average
   - ○ Poor

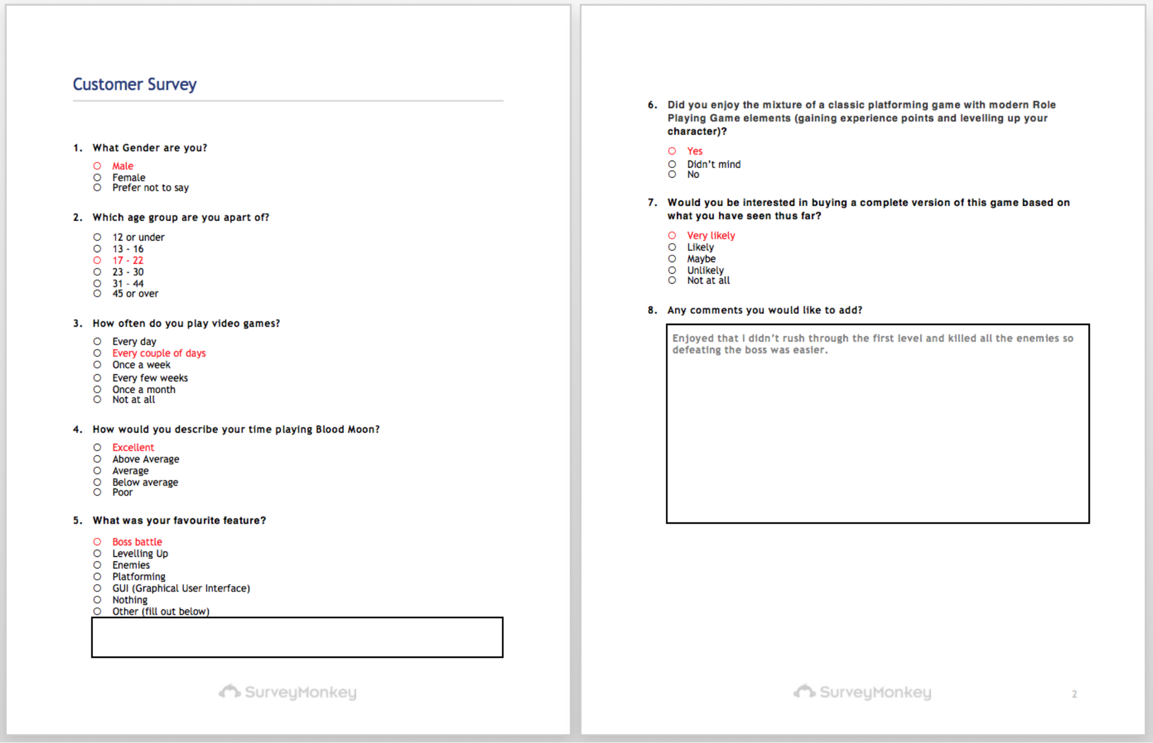
5. **What was your favourite feature?**
   - ○ Boss battle
   - ○ Levelling Up
   - ○ Enemies
   - ○ [✓ Platforming]
   - ○ GUI (Graphical User Interface)
   - ○ Nothing
   - ○ Other (fill out below)

6. **Did you enjoy the mixture of a classic platforming game with modern Role Playing Game elements (gaining experience points and levelling up your character)?**
   - ○ Yes
   - ○ [✓ Didn't mind]
   - ○ No

7. **Would you be interested in buying a complete version of this game based on what you have seen thus far?**
   - ○ Very likely
   - ○ [✓ Likely]
   - ○ Maybe
   - ○ Unlikely
   - ○ Not at all

8. **Any comments you would like to add?**

Would prefer more jumping challenges



Customer Survey

1. **What Gender are you?**
   - ○ [✓ Male]
   - ○ Female
   - ○ Prefer not to say

2. **Which age group are you apart of?**
   - ○ 12 or under
   - ○ 13 - 16
   - ○ [✓ 17 - 22]
   - ○ 23 - 30
   - ○ 31 - 44
   - ○ 45 or over

3. **How often do you play video games?**
   - ○ Every day
   - ○ [✓ Every couple of days]
   - ○ Once a week
   - ○ Every few weeks
   - ○ Once a month
   - ○ Not at all

4. **How would you describe your time playing Blood Moon?**
   - ○ [✓ Excellent]
   - ○ Above Average
   - ○ Average
   - ○ Below average
   - ○ Poor

5. **What was your favourite feature?**
   - ○ [✓ Boss battle]
   - ○ Levelling Up
   - ○ Enemies
   - ○ Platforming
   - ○ GUI (Graphical User Interface)
   - ○ Nothing
   - ○ Other (fill out below)

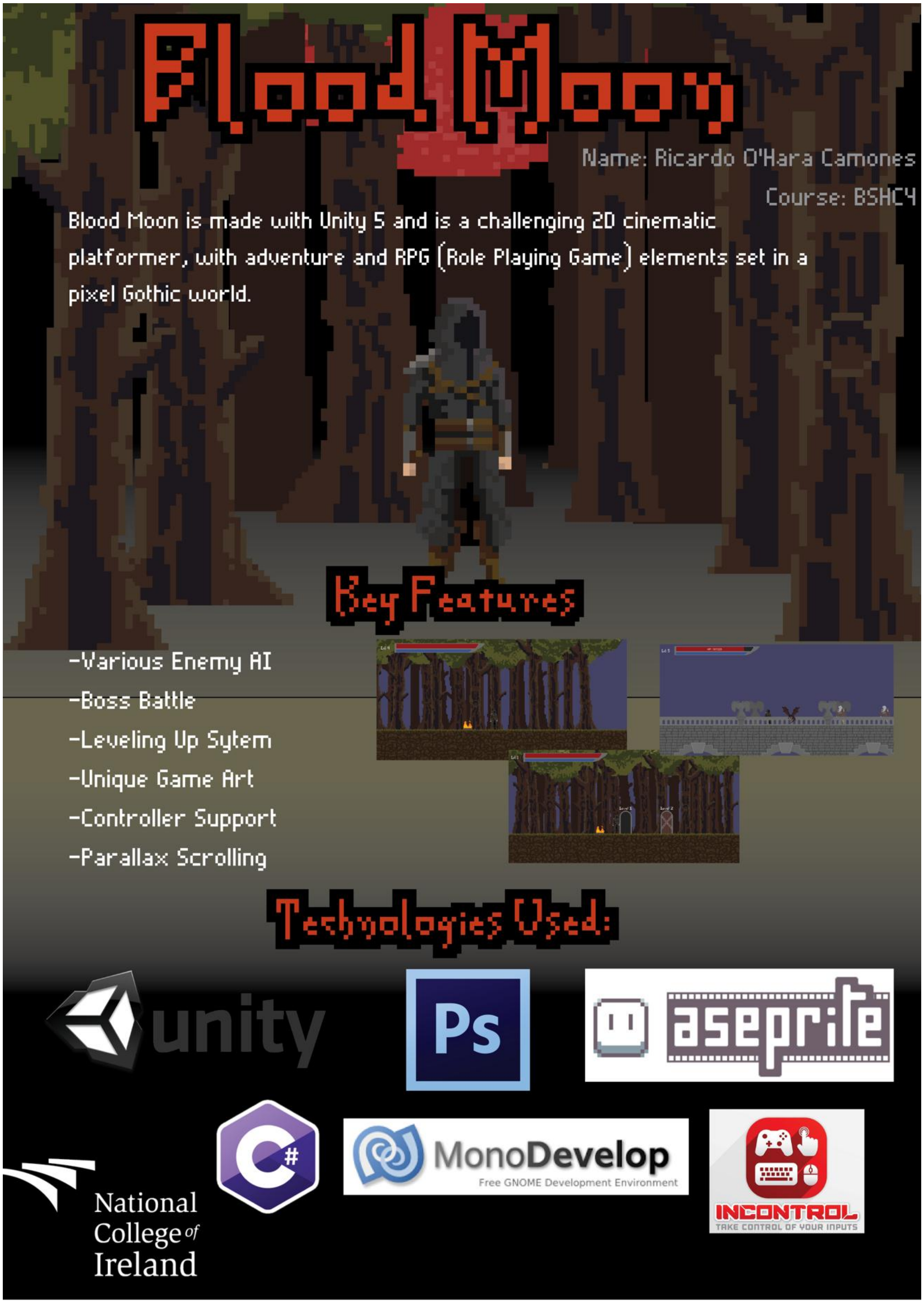
6. **Did you enjoy the mixture of a classic platforming game with modern Role Playing Game elements (gaining experience points and levelling up your character)?**
   - ○ [✓ Yes]
   - ○ Didn't mind
   - ○ No

7. **Would you be interested in buying a complete version of this game based on what you have seen thus far?**
   - ○ [✓ Very likely]
   - ○ Likely
   - ○ Maybe
   - ○ Unlikely
   - ○ Not at all

8. **Any comments you would like to add?**

Enjoyed that I didn't rush through the first level and killed all the enemies so defeating the boss was easier.

6.6. Showcase Poster



Blood Moon

Name: Ricardo O'Hara Camones

Course: BSHC4

Blood Moon is made with Unity 5 and is a challenging 2D cinematic platformer, with adventure and RPG (Role Playing Game) elements set in a pixel Gothic world.

Key Features

- -Various Enemy AI
- -Boss Battle
- -Leveling Up Sytem
- -Unique Game Art
- -Controller Support
- -Parallax Scrolling

Technologies Used:

unity
Ps
aseprite
MonoDevelop
Free GNOME Development Environment
C#
INCONTROL
TAKE CONTROL OF YOUR INPUTS
National College of Ireland