National College of Ireland

BSc in Computing

2014/2015

Kelly McGuinness

x12503953

kelly.mcguinness@student.ncirl.ie

There's No Place Like Home...

Technical Report

National College of Ireland

Table of Contents

Executive Summary

No Place Like Home is a First Person Shooter game developed in Unity 3D. The objective of this game is to move through the 3d environment and shoot the enemy while trying to stay alive. I had initially intended on my game being the typical FPS style game where the player is a soldier. I needed to come up with an unique aspect to set my game aside from all the FPS's already on the market.

My imagination grew the more I worked on the game and I decided to use an interesting back story to my advantage. They say you shouldn't base on a game on a story, but rather develop the game and the story will come. Every game also needs a hero type character that can kill hundreds without coming across like the bad guy. Making the enemies fantasy based can make this more socially acceptable. I decided to use skeleton like creatures and a God of the underworld for my main boss character. Based on what I already had in my game I thought of a suitable back story that would be interesting and unique enough. I also wanted to incorporate some new technology so after some research I came across RAIN AI. I felt this would be something unique I could add to my project as it's a relatively new technology and not many people are using it in their games. Not being a gamer myself I felt like I found a hidden gem inside the Unity game engine.

Back Story

"Ten years ago you made a deal with a demon. The day has finally come and he's come to your home town to collect what's rightfully his. You pay your debt but regretful & terrified, you look for a way to revoke this binding contract. You find a book on mythology which speaks of a loophole in demon contracts. It states that if you have the courage and the power to defeat the necromancer - the demonic god of the underworld and his army, the contract will be unbound. You take the only weapon you have and pray it’s enough to take down this soul taker. Fail in your only chance and you’ll be taken to hell, and you’ll realise, there really is No Place Like Home."
After already giving my game a title at the start I wanted to be able to tie it in with characters and the haunted town scene I was using. Given these circumstances I feel I did a good job of creating an interesting storyline.

Introduction

Background

The games industry has had much growth in recent years. It is a great industry to get involved in as it allows creativity, innovation and freedom for developers and hobbyists. They get a chance to experiment with all forms of media including sound design, environment design and programming.
 As there has been an explosion of the new 'Indie' market of games, the gaming industry now allows smaller-scale games to be developed and released more freely than in the past. Today it has become so easy to create a game. A large studio is no longer needed and the freely available tools make it so simple to create an idea from the comfort of your own home. Indie games show more innovation and developers are willing to take risks on their games. 'Indie' games also have a large market base with many of the games being released on PC, Android etc.

Currently there is a massive increase in the popularity of not just indie games, but particularly innovative indie horror games. I put a lot of research into trying to find out what users want from a game. I had to decide between a first person shooter or a third person shooter, and what type of environment would be best to build my game. I finally decided to go with a FPS in a horror style environment. I decided to go with a 3D game over a 2D game as I think it makes the game that extra bit challenging and interesting. There are also benefits to the style of game I chose "While one widely held view maintains that playing video games is intellectually lazy, such play actually may strengthen a range of cognitive skills such as spatial navigation, reasoning, memory and perception, according to several studies reviewed in the article. This is particularly true for shooter video games, which are often violent, the authors found. A 2013 meta-analysis found that playing shooter video games improved a player's capacity to think about objects in three dimensions just as well as academic courses designed to enhance these same skills, according to the study." (Apa.org, 2016)

I chose the Unity 3D game engine to develop my game as it has a large community which provides learning support from the Unity website and forums. There are many tutorials also available on YouTube. You can code your game in either C# or JavaScript which made it ideal, as both languages had been covered in modules in the past years of college. The game will run on a Windows PC. This gives me the option of publishing my game to the popular PC online gaming market Steam. This would make my game available to many users online.

As indie games allow developers to take risks, I wanted to change it up from the usual FPS game and do something different by combining two different game genres which included the shooting style game and the fantasy role playing game (RPG). I also wanted to add interactive aspects to the game so I incorporated some interactable objects to add more realistic senses. The player can interact with certain objects in the game by using the E key. I wanted conditions in the game for example, the door won't unlock without the key. This adds a greater challenge and depth to the game as it gives the player an extra task to accomplish. By including some RPG aspects, it adds creativity and problem solving skills to a typical shooting style game.

The game engine used was Unity3D. The game coding was written mainly in JavaScript with some other parts in C#. MonoDevelop, the main IDE which is used with Unity was used to edit scripts. A-Star pathfinding algorithm was implemented as well as the new RAIN AI technology. Assets in my game were sourced from the Unity asset store and sound clips were found from soundbible.com.

Aims

The aim of this project was to create a fully functional FPS 3D game. The overall aim of my game was to create a tense and unnerving experience for the player. I chose a creepy environment setup as atmosphere is essential for the game. As I have pathfinding on my enemies, they will not attack until the player is in a certain range. They might not always be visible to the player as the scene has many hiding places. This adds a level of fear in the game as you will not know how close the next enemy is or when they will attack. I also included audio and sound effects to support the atmosphere of the game.

The purpose of this project is to develop a first person shooter game with good graphics, audio and animations made with basic technologies and a low budget. I wanted to make a game that was easy to install and show what can be done with just a bit of time and effort. I have already inspired one of my friends to create a game as I had no experience prior and they were impressed with what I could accomplish in a short amount of time.

The game is intended to be used on a PC running windows with a keyboard and a mouse.

Technologies

Unity

Unity is a cross platform game engine with its own built in IDE which allows for development of either 2D or 3D games. Unity is primarily used to create games for a variety of platforms such as web, desktop, consoles and most recently, mobile platforms.

Unity is the primary tool that was used to create this project. It was chosen primarily because of the fact that it is free to use and has a vast amount of support available from the community including an asset store from which some free assets can be added to a project in order to enhance it as well as saving time and money.
Unity features a main scene view which displays the visual elements of the project such as the terrains, models, etc. Any object that is used in Unity, whether it's a terrain, a shape, a model, is referred to as a GameObject. These game objects can be modified using the Inspector view. With the Inspector view you can see the transform, rotation and scale information of the selected GameObject. You can also see the scripts that the object is using and the components attached to it.
Unity offers support for assets that are created in a wide variety of file formats, such as fbx. Assets created in other programs can simply be added to a game project in Unity by choosing them from the saved location on your computer. Unity makes adding imported art assets to the game scene as simple as drag and drop in the desired location.

Unity's scripting is based primarily on MonoDevelop which is an open source implementation within the .Net framework. When working with Unity, you are given the option of using JavaScript or C#. Unity makes use of MonoDevelop, though developers are free to use whatever scripting tool they desire.
Unity supports project deployment on multiple platforms. When a project is finished, developers can choose which platform they would like their project to run on, eg. PlayStation 4, Xbox One, Window, Linux, Android, IOS and many others. I am currently using Unity version 5.3.4 in the development of my game.

Unity Asset Store

The asset store is a resource that is located within the Unity editor and is also available online. The store currently consists of thousands of packages which are divided into sections such as environments, characters, textures/materials etc. that are available to developers to use within their projects. They can be free or paid, depending on the level of work involved in the asset. Without the asset store, all the objects in the game would have to be created manually in a program such as blender and would become extremely time consuming. The asset store was used for all the models in my game ranging from the gun to the enemies to the scene.

MonoDevelop

MonoDevelop is an IDE which is installed with Unity 3D. It allows you to create and edit C# and JavaScript files used in the game. It differs from the standard IDE because it has in-built functions and auto-completion with coding specifically relating to Unity. MonoDevelop is able to interact with the GameObjects used in Unity. It can access their co-ordinates, scripts attached to them, their tags, etc. These are very useful in creating powerful scripts. It also allows variables to be edited from Unity so changes do not need to be made in the scripts every time. This made it really easy to test different things in my game as I could easily change variables such as distance or try different images or soundclips. Errors that appear within the IDE are also shown in the Unity Debugging Window which recompiles regularly. Double clicking the error will bring you to the line of code in the script that is causing the problem.

C#

C# is a modern programming language which allows developers to create applications which will then run on the .NET framework. I had been introduced to C# in one of my modules in college which is a reason Unity was chosen. As I had been introduced to C# in college I was familiar with it and it made it that bit easier when trying to fix problems in scripts.

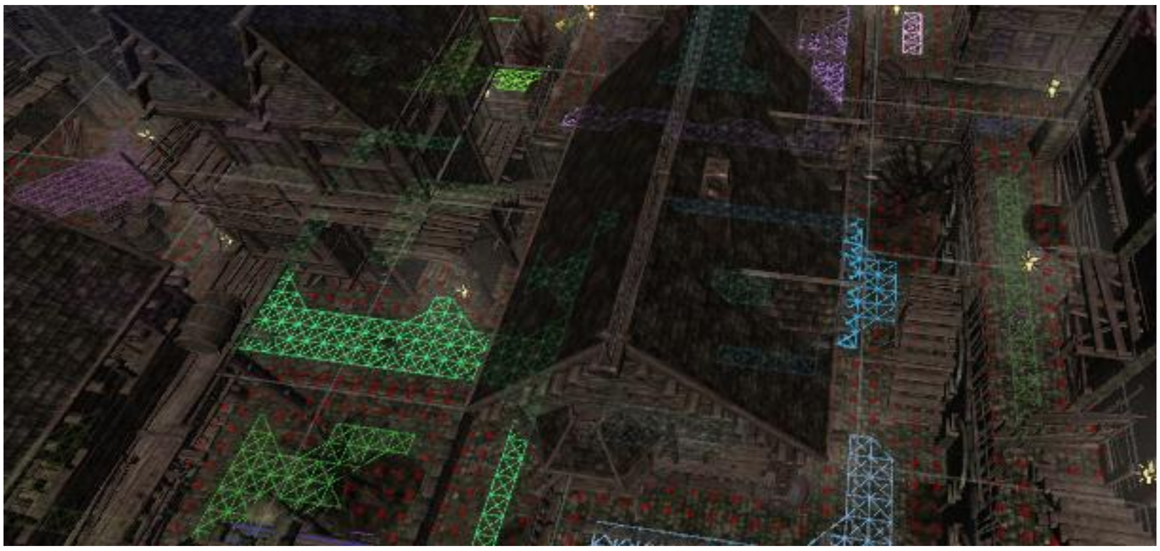
JavaScript

JavaScript is a multi platform, object-oriented scripting language. JavaScript can be connected to the objects of its environment to allow programmatic control over them. I enjoyed working with Javascript more than C# in my project as it made it easy to do things like play an animation for a set amount of time using yield WaitForSeconds (5).

A-Star Pathfinding Algorithm

This is a computer algorithm that is widely used in pathfinding and graph traversal. A-Star tries to look for a better path by using heuristic functions to optimize the search which gives priority to nodes. A-Star is typically used in games as it is useful when you are searching for a path on a map where you can guess the distance to the target from a given graph node. Grid graphs were used which are useful for ease of use and fast updates. These are built by arranging nodes in a grid like pattern around the world. They allow for fast updates due to their regular structure and are useful in games that require frequent updates to the graphs. (Arongranberg.com, 2016)

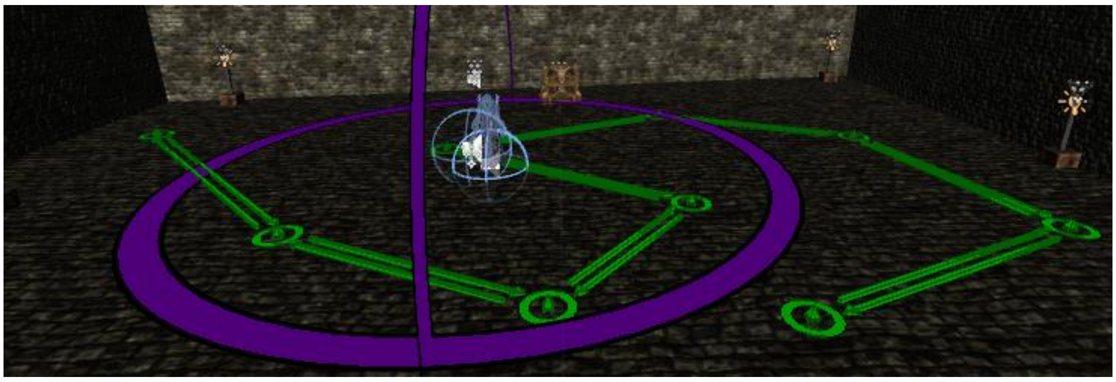
A* was beneficial in my game as I could easily set up a grid graph that showed the ground layer and the obstacles layer. I could assign objects in my game to a specific layer and decide if the enemy could walk on them or not. This made it so the enemy couldn't walk through walls or objects and they would have to go around them. Walkable areas are in colour and non walkable areas are identified by red dots (See image below). The pathfinding also computed the best path to take to reach the specified destination (the player) while avoiding the obstacles and doing it in the least amount of time possible.


Generated grid graph with raycast hitting terrain and allowing walkable nodes in ground layer mask

Rain AI

RAIN by Rival Theory has been specifically created to reduce the challenge of creating interactive characters. RAIN is actually a number of different AI products in one. Included in the package are support for creating pathfinding algorithms, behaviour trees and a sensory system. RAIN's movement, behaviour and pathfinding systems work together to control character motion. Waypoints can be added to create paths the character will follow. Behaviour trees can help to tie movement behaviour to actions such as patrol, attack, chase etc. RAIN's behaviour system can help bring a character to life by adding realistic movement, animation and senses. RAIN allows a character to become aware of external elements in their environment through digital sensations. When equipped, the character gains the ability to understand or notice something easily like footsteps as a character approaches. (Rival{Theory}, 2016)

I wanted to increase the complexity of my project even further by incorporating something unique. I brought an enemy into my game and set up a waypath using behaviour trees which will allow my enemy to patrol an area and detect the player if they come too close. They will then transition to chase and attack mode and will take health from the player until they deteriorate.



Structure

Section One: The summary, background story, aims, concepts and ideas behind the project will be discussed. I will also include the technology used to create the project and how it helped to benefit my game.

Section Two: The functional and non-functional requirements of the project will be discussed as well as the structure and classes of the project . Then the implementation of the project will be described with details of the testing process it undertook.

Section Three: This will outline the conclusions and what I have learned from implementing and completing the project. I will include the advantages and disadvantages in developing the project.

Section Four: This will explain the ways I could expand the project in the future and the future development of the project.

Section Five: This will show the references of the project.

Section Six: This will contain the appendix of previously submitted documents including project proposal, requirements specification and monthly journals.

System / Requirements

The requirements in this section have expanded from the previous version. As the game evolved so did the requirements and I felt they were necessary to include.

Functional Requirements

This section shows the basic needs for the game in order to have the game up and running in the time that is required.

Main Menu: The game should have a main menu where the player can decide what option they would like to invoke.

Terrain: The game should have a terrain where the characters can exist. It should have realistic physics including gravity, wind and borders.

Characters: The game must have a set of characters who will either act as the Artificial intelligence or the player character

Weapon: The player must be provided with a weapon in order to kill and damage an enemy.

Play Game: The player should be able to select the play game option from the main menu. The play game button will give a short background story (which the player can skip if they wish) before the game will start.

Controls: The player should be able to select the controls option from the main menu which will show them the buttons used to control the player.

Volume: The player should be able to adjust the volume of the background audio. A slider is presented on the main menu screen.

Interactable Objects: The player should be able interact with a few objects in the game. The player can interact with an object by using the E key.

Game Completed: The player should be able to complete the game and receive recognition from the game that it is over. The player should complete the game once they have collected their soul after defeating the boss. When the player walks through the door and into the soul it will load the game over screen which will then bring them back to the main menu where they can decide to play again or quit.

Pause Game: The player should be able to pause the game. The player will press the ESC button or press the pause button in the corner of the screen which will pause the game and give the player the option to continue, return to main menu or quit the game.

Quit Game: The player should be able to quit the game and stop playing. To prevent accidental exit, the game will ask is the user sure. They can click yes to exit the game or no to continue.

Fight/Defeat Enemies: The player should be able to shoot and kill enemies in the game. These enemies will make the game more challenging as the player will need to pass them. The player can fight them or attempt to run away from them.

Health System: The player must be able to lose health when attacked by enemies. When the player has run out of health they will have to respawn which will bring them back to the start of the game. Health will be restored to max health and any enemies they have already killed will remain in that state.

Player Movement: The player should be able to move the controllable character. The character can be moved using the movement keys and they can jump, crouch and shoot.

Aim And Sight: The player should be able to aim the gun and look down the sight of the gun. This is achieved by pressing the right mouse click.

Intelligent Enemy AI:

The game requires multiple enemies which attack the player. When the user has defeated these enemies, a boss character will be introduced which will be stronger and harder to defeat. This boss character will also be more intelligent and it will provide the player with a challenge.

Graphics, animation and sound engineering

The system will constantly render textures and models. Graphics will need to be realistic to provide the user with a believable world. The game should also have suitable animations to compliment the realism of the graphics including walking, running, attacking etc. I feel that audio will help to complete this 3D environment. Background music to set the scene, footsteps, and gun shots should be incorporated.

Data requirements

All data is passed in the background. These are variables that monitor the positions of both the player and the enemy and keep track of the players health. Information is only saved during a game. No information is saved when the application is closed.

User requirements

The user requirements are used to describe what the user will need in order to run and play the game. The user must have a computer or laptop capable of running a modern version of the Windows operating system (must be Windows XP or higher) and the graphics card requirements will vary depending on the type of game. As this project consists of simple graphics, the standard graphics cards in the users' PC should be able to run the game.
Internet access will also be required for the user to be able to download the application. When the game is compiled, it is built into an executable file which is separate from Unity which will benefit the user as they will not be required to download and install Unity if they want to play the game. The game can also be compiled to run on web browsers. Users will need to install Unity Web Player which is supported by all web browsers. The user must also have a computer, key board, mouse and speakers/headphones to play the game.

Environmental requirements

In order to play the game, the users will need a Windows, Linux or Mac operating system environment but the game was designed to use on the Windows platform.

This application will run on a computer or laptop capable of running a modern version of the Windows operating system (must be Windows XP or higher)

Usability requirements

The menu for the game should be easily accessible and visible, and clearly laid out to provide a clear understanding of the game menu. It should be easy for the user to accomplish a task and navigate through the menu. The game should not overwhelm the users and the UI should be clearly laid out and visually appealing.

Non-Functional Requirements

Performance/Response time requirement

It is important to ensure that the game does not suffer any graphical or performance problems as this can lead to a frustrating game play. The game needs to run at a consistent frame rate to ensure that the graphics and animations remain smooth. The response time of the game should be immediate so that when the player initiates an action in the game, the results will be immediate. The player will need precision for jumping and attacking, so the response time for input methods must be very quick.

Availability requirement

Once the user has downloaded and installed the game onto their machine it should be available to play whenever they want. The game will be built for Windows which will make the game easily available.

Security requirement

The player will need to ensure that they have the proper security privileges on their machine to allow them to install and run the game.

Reliability requirement

Once the user has downloaded and installed the game onto their machine it should be available to play whenever they want.

Maintainability requirement

Games can be easily maintained after their release through online patches. These patches can be used to fix any problems in the game. If there are several major bugs then a patch will be released with the intention of addressing and fixing these. Players also have the ability to report any bugs to developers allowing them to fix the problem.

Portability requirement

Users can play the game through their PC or laptop. They can also copy it onto a storage device such as a memory stick and take the game with them if they wish. As I am using Unity3D, the game can be built to run using the Unity web player. The web player would allow users to play the game over the internet so they could play on any device connected to the internet.

Extendibility requirement

The option to expand the game with additional content, even after it has been released is always there. Development for the game does not need to end and this will be released in the form of an expansion which will aim to improve the overall game play experience.

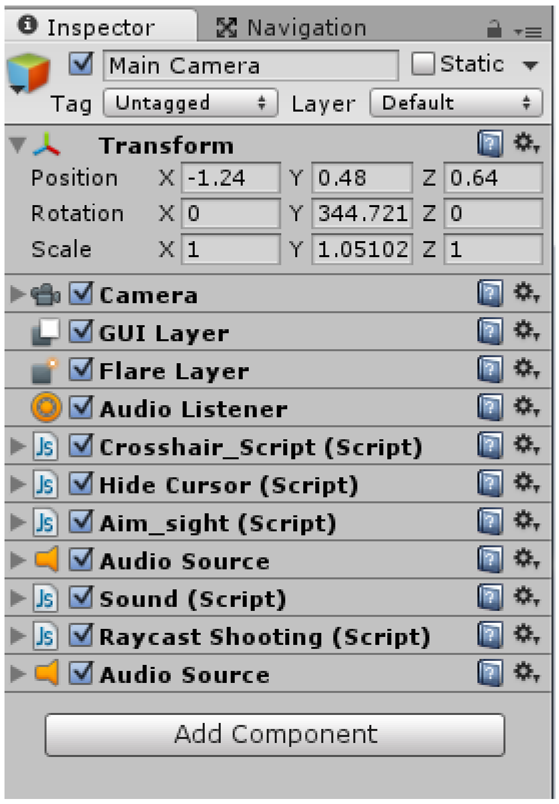
Reusability requirement

If I were to create another FPS / horror or fantasy game I could reuse most of the code. I could also reuse the graphics or assets used in my game.

Design and Architecture

The application was built using the Unity 3D game engine. The functions to build and animate objects, characters and scenes are supported in Unity. These are known as Game Objects. Unity supports JavaScript and C# but Unity differs from other programs as it doesn't just consist of classes and interfaces. It users the hierarchy, where Game Objects are arranged and can be parented to other objects and an inspector view where you can scale, rotate and move objects and assets. You can also assign materials, colliders and scripts to an object in the inspector view to give the object interactivity.

These are the scripts attached to my main camera which control my shooting, audio and visuals.

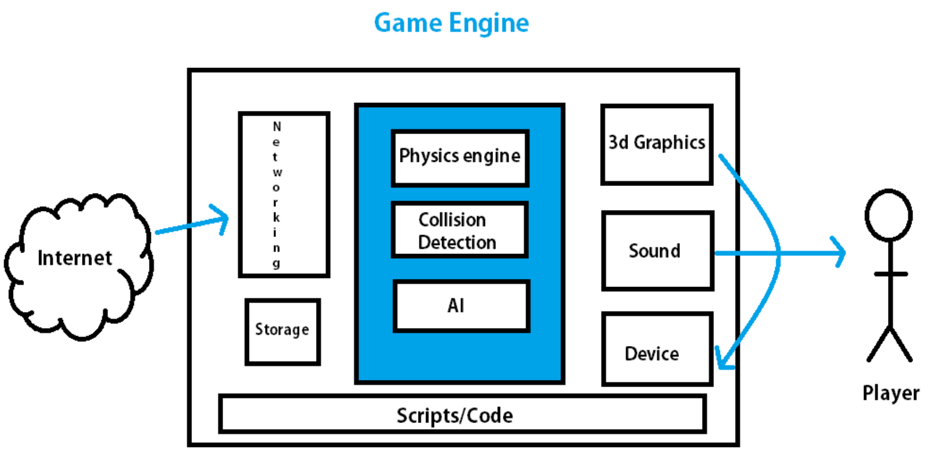


Main Camera — Static
Tag: Untagged — Layer: Default

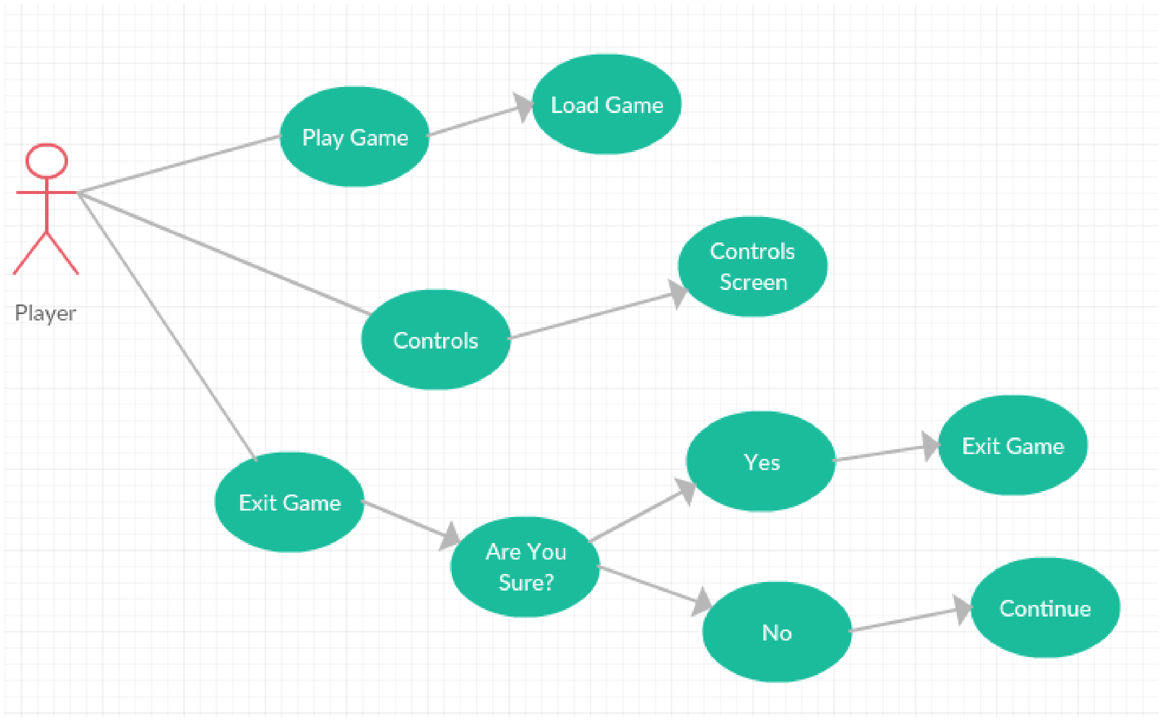
Transform
Position X -1.24 Y 0.48 Z 0.64
Rotation X 0 Y 344.721 Z 0
Scale X 1 Y 1.05102 Z 1

Camera
GUI Layer
Flare Layer
Audio Listener
Crosshair_Script (Script)
Hide Cursor (Script)
Aim_sight (Script)
Audio Source
Sound (Script)
Raycast Shooting (Script)
Audio Source

System Architecture

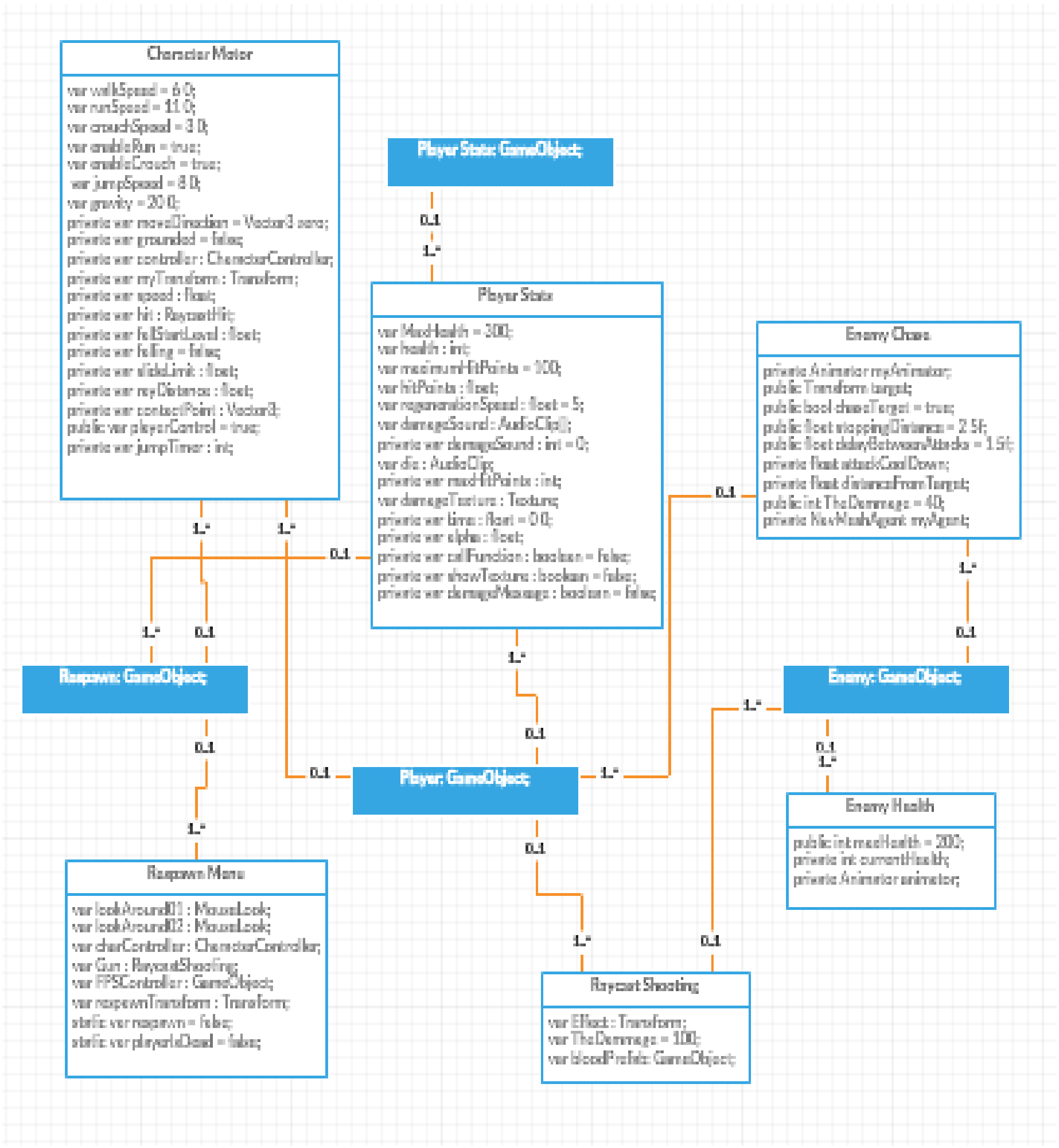


Use Case Diagram



When the player starts up the game the main menu will be displayed. The player will be presented with 3 options to choose from. The ‘Play Game’ option will bring the player to play the game. The ‘Controls' option will bring the player to a screen that will show the controls for the game and the ‘Exit Game’ option will allow the player to quit the game and exit the application. A pop-up menu will appear asking the player are they sure. If the user selects yes, the application will close, if they select no, they can continue as normal.

Class Diagram



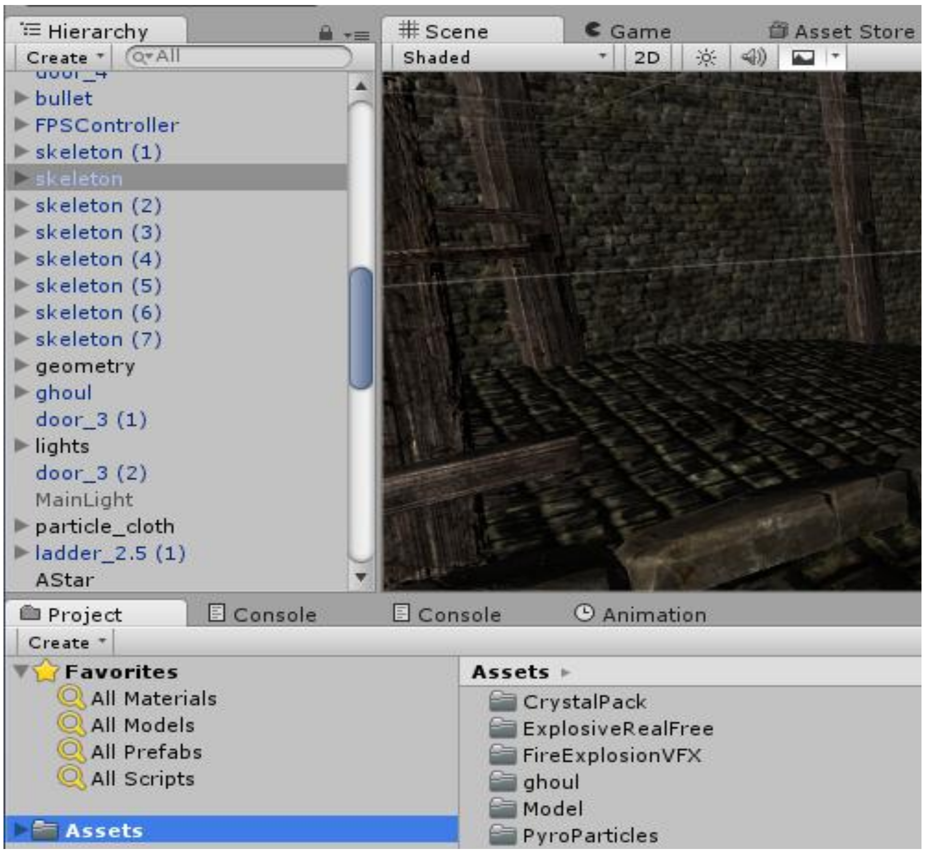
Character Motor
var walkSpeed = 6.0;
var runSpeed = 11.0;
var crouchSpeed = 3.0;
var enableRun = true;
var enableCrouch = true;
var jumpSpeed = 8.0;
var gravity = 20.0;
private var moveDirection = Vector3.zero;
private var grounded = false;
private var controller : CharacterController;
private var myTransform : Transform;
private var speed : float;
private var hit : RaycastHit;
private var fallStartLevel : float;
private var falling = false;
private var slideLimit : float;
private var rayDistance : float;
private var contactPoint : Vector3;
public var playerControl = true;
private var jumpTimer : int;
Player Stats: GameObject;
Player Stats
var MaxHealth = 300;
var health : int;
var maximumHitPoints = 100;
var hitPoints : float;
var regenerationSpeed : float = 5;
var damageSound : AudioClip[];
private var damageSound : int = 0;
var die : AudioClip;
private var maxHitPoints : int;
var damageTexture : Texture;
private var time : float = 0.0;
private var alpha : float;
private var callFunction : boolean = false;
private var showTexture : boolean = false;
private var damageMessage : boolean = false;
Enemy Chase
private Animator myAnimator;
public Transform target;
public bool chaseTarget = true;
public float stoppingDistance = 2.5f;
public float delayBetweenAttacks = 1.5f;
private float attackCoolDown;
private float distanceFromTarget;
public int TheDamage = 40;
private NavMeshAgent myAgent;
Respawn: GameObject;
Enemy: GameObject;
Player: GameObject;
Enemy Health
public int maxHealth = 200;
private int currentHealth;
private Animator animator;
Respawn Menu
var lookAround01 : MouseLook;
var lookAround02 : MouseLook;
var charController : CharacterController;
var Gun : RaycastShooting;
var FPSController : GameObject;
var respawnTransform : Transform;
static var respawn = false;
static var playerIsDead = false;
Raycast Shooting
var Effect : Transform;
var TheDammage = 100;
var bloodPrefab: GameObject;

I have chosen this architecture as the player and the enemy both interact with the game. The Character Motor and Player Stats are scripts attached to the player. They control movement and health in the game. When health reaches 0, the player will respawn to a location set in the game. The Respawn script controls this. The enemy game object has two scripts which include the Enemy Chase and Enemy Health. Same as the player, this controls movement and health for the enemy. When the health reaches 0, the enemy game object is destroyed. As the player is the target for the enemy, the two game objects interact with eachother.

Implementation

Creating The Project

When Unity is first launched you have the option to create a new project. You give the project a name and you are given a list of in-built Unity packages which you can import and use in your project. You can choose where you want the project to be saved. Once the desired directory has been chosen click Create Project. Unity will restart and the user will be presented with a blank scene in which the game is to be developed. There are two view windows within Unity. One is the Scene view where all the design and the development happen. The other is the Game view. This is what it actually appears like while playing the game. The Hierarchy is a column on the left hand side and this is where game objects in the scene are stored. The Project section stores all the assets in the game including scripts, scenes, images, audio, materials and textures. The Inspector column shows what scripts are attached to the game object. You can download and import packages from the asset store which include standard assets such as skyboxes or terrain materials to add realistic features to your project.



```
door_4
bullet
FPSController
skeleton (1)
skeleton
skeleton (2)
skeleton (3)
skeleton (4)
skeleton (5)
skeleton (6)
skeleton (7)
geometry
ghoul
door_3 (1)
lights
door_3 (2)
MainLight
particle_cloth
ladder_2.5 (1)
AStar
```

```
CrystalPack
ExplosiveRealFree
FireExplosionVFX
ghoul
Model
PyroParticles
```

Scripts are essential to create function and behaviour in a game. This is where you can tell an object what you want it to do, what animation to play or to perform different actions on different conditions.

CharacterMotorV2.js

This script controls the players movement. This allows the player to move and control the player character in the game. The character is moved along the x, y, and z axis depending on what directional keys are pressed on the keyboard. This script also controls movements such as walking, running and crouching.

```
// If running is enabled, change to run speed when left shift is pressed:
if (Input.GetKey(KeyCode.LeftShift) && enableRun == true)
{
    speed = runSpeed;
}
// If crouching is enabled, change to crouch speed when "c" is pressed:
else if (Input.GetKey("c") && enableCrouch == true)
{
    speed = crouchSpeed;
}
// If nothing is pressed, use walkSpeed:
else
{
    speed = walkSpeed;
}
```

PlayerStatsV2.js

This script keeps record of the health of the player. When the player is attacked by an enemy, health is taken away from the player. The player also makes a hurt sound when it is attacked. If the health is less or equal to zero, it calls the dead function which shows the respawn menu.

```
function ApplyDammage (TheDammage : int)
{
    health -= TheDammage;
        playSoundsGO.GetComponent.<AudioSource>().clip = damageSound[Random.Range(0, damageSound.length)];
        playSoundsGO.GetComponent.<AudioSource>().volume = 0.4;
        playSoundsGO.GetComponent.<AudioSource>().Play();
        time += TheDammage/7;

        if(health <= 0)
    {
        Dead();
        }
        HealthBar();

}

function Dead()
{
    RespawnMenuV2.playerIsDead = true;

    Debug.Log("Player Died");
}
```

GhoulHealth.cs

This script is similar to the player health script. It removes health from the enemy when they are hit with the bullet but instead of being able to respawn, they play a death animation and then the game object is destroyed.

```
// Update is called once per frame
public void ApplyDammage (int ApplyDammage) {

    currentHealth -= ApplyDammage;
    animator.SetTrigger ("isHit");
    if (currentHealth <= 0){
        //call die method
        Die();
    }

}

void Die () {
    animator.SetTrigger ("isDead");
    StartCoroutine ("waitThreeSeconds");

}

IEnumerator waitThreeSeconds(){
    yield return new WaitForSeconds (3);
    Destroy (gameObject);

}
```

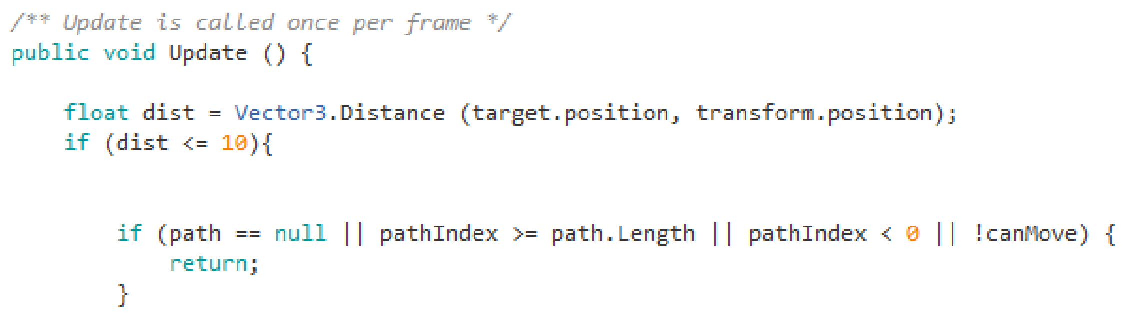
RaycastShooting.js

You would think you shoot from the gun but this script is actually attached to the main camera. It sends a ray out and whatever game object it hits gets damage applied to them (if the health script is attached to them). Eg, shooting at a wall won't affect the wall but shooting at an enemy will take health. A blood effect and sparks also present when the gun is shot to create a realistic effect when killing enemies.

```
var hit : RaycastHit;
var ray : Ray = Camera.main.ScreenPointToRay(Vector3(Screen.width*0.5, Screen.height*0.5, 0));


if (Input.GetMouseButtonDown(0))
{
    if (Physics.Raycast (ray, hit, 100))
    {
        var particleClone = Instantiate(Effect, hit.point, Quaternion.LookRotation(hit.normal));
        Destroy(particleClone.gameObject, 2);
        Instantiate(bloodPrefab, hit.point, transform.rotation);
        hit.transform.SendMessage("ApplyDammage", TheDammage, SendMessageOptions.DontRequireReceiver);
    }
}
```

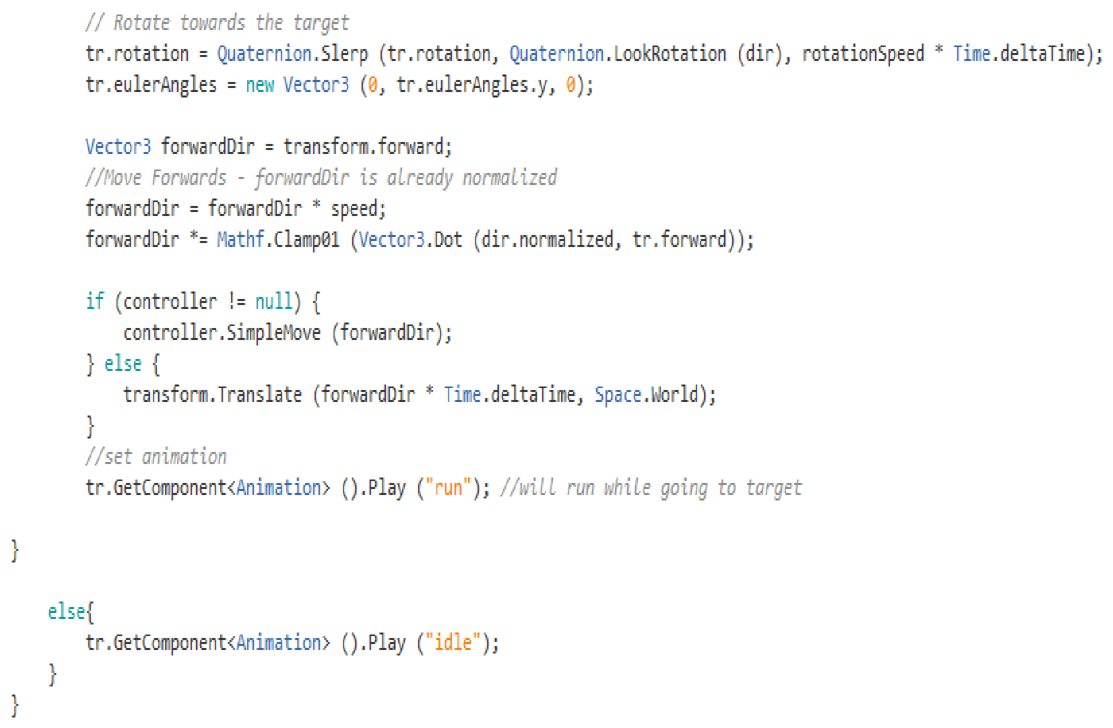
AIFollow.cs

This script is used for the pathfinding algorithm. The pathfinding algorithm path updates as the character moves around the map. An AI Follow script and Seeker script was added to the enemy to allow him to follow the character. A trigger will go off if you get to close to the enemy, which will allow him to identify the character player. An if statement is used to tell the enemy what it should do. If the player is 10 units or less away from the enemy, it will start it's pathfinding to get to the target.



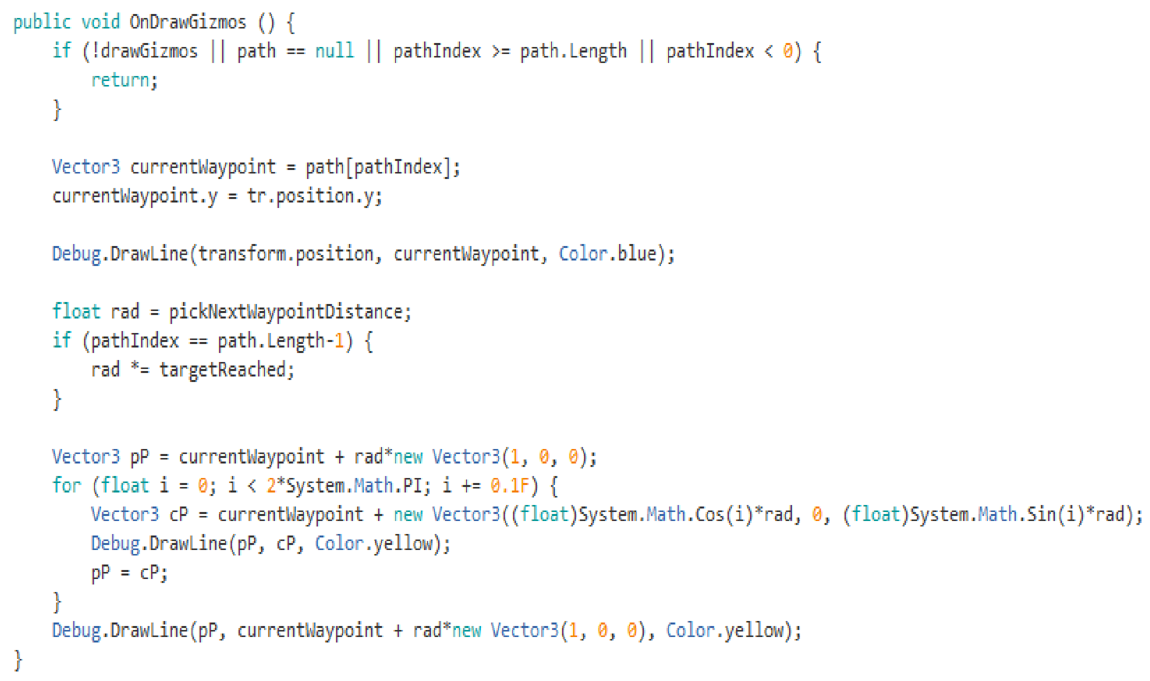
```
/** Update is called once per frame */
public void Update () {

    float dist = Vector3.Distance (target.position, transform.position);
    if (dist <= 10){


        if (path == null || pathIndex >= path.Length || pathIndex < 0 || !canMove) {
            return;
        }
```

While the enemy is going towards the target, it will rotate towards it and will play the running animation. If the player is more than 10 units away from the enemy, it will just remain in an idle position until the player is closer.



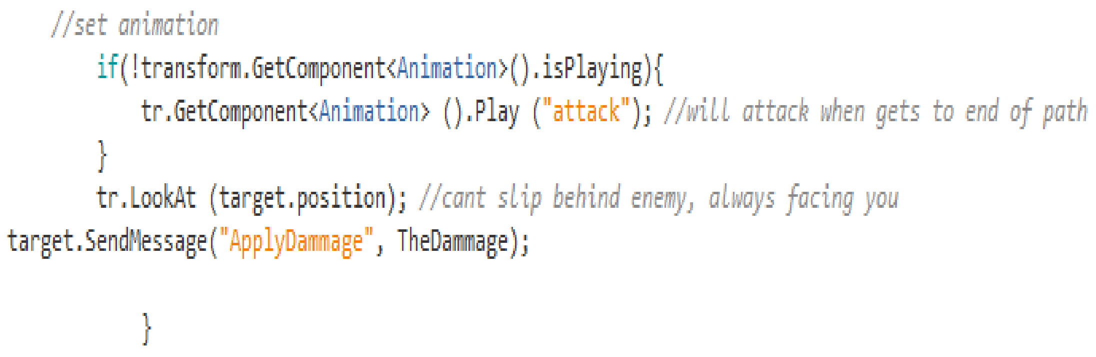
```
            // Rotate towards the target
            tr.rotation = Quaternion.Slerp (tr.rotation, Quaternion.LookRotation (dir), rotationSpeed * Time.deltaTime);
            tr.eulerAngles = new Vector3 (0, tr.eulerAngles.y, 0);

            Vector3 forwardDir = transform.forward;
            //Move Forwards - forwardDir is already normalized
            forwardDir = forwardDir * speed;
            forwardDir *= Mathf.Clamp01 (Vector3.Dot (dir.normalized, tr.forward));

            if (controller != null) {
                controller.SimpleMove (forwardDir);
            } else {
                transform.Translate (forwardDir * Time.deltaTime, Space.World);
            }
            //set animation
            tr.GetComponent<Animation> ().Play ("run"); //will run while going to target

    }

        else{
            tr.GetComponent<Animation> ().Play ("idle");
        }
    }
```

The script draws a circle around the target, which is the player, with the size showing how close the AI needs to get to it for it to count a 'reached'



```
public void OnDrawGizmos () {
    if (!drawGizmos || path == null || pathIndex >= path.Length || pathIndex < 0) {
        return;
    }

    Vector3 currentWaypoint = path[pathIndex];
    currentWaypoint.y = tr.position.y;

    Debug.DrawLine(transform.position, currentWaypoint, Color.blue);

    float rad = pickNextWaypointDistance;
    if (pathIndex == path.Length-1) {
        rad *= targetReached;
    }

    Vector3 pP = currentWaypoint + rad*new Vector3(1, 0, 0);
    for (float i = 0; i < 2*System.Math.PI; i += 0.1F) {
        Vector3 cP = currentWaypoint + new Vector3((float)System.Math.Cos(i)*rad, 0, (float)System.Math.Sin(i)*rad);
        Debug.DrawLine(pP, cP, Color.yellow);
        pP = cP;
    }
    Debug.DrawLine(pP, currentWaypoint + rad*new Vector3(1, 0, 0), Color.yellow);
}
```

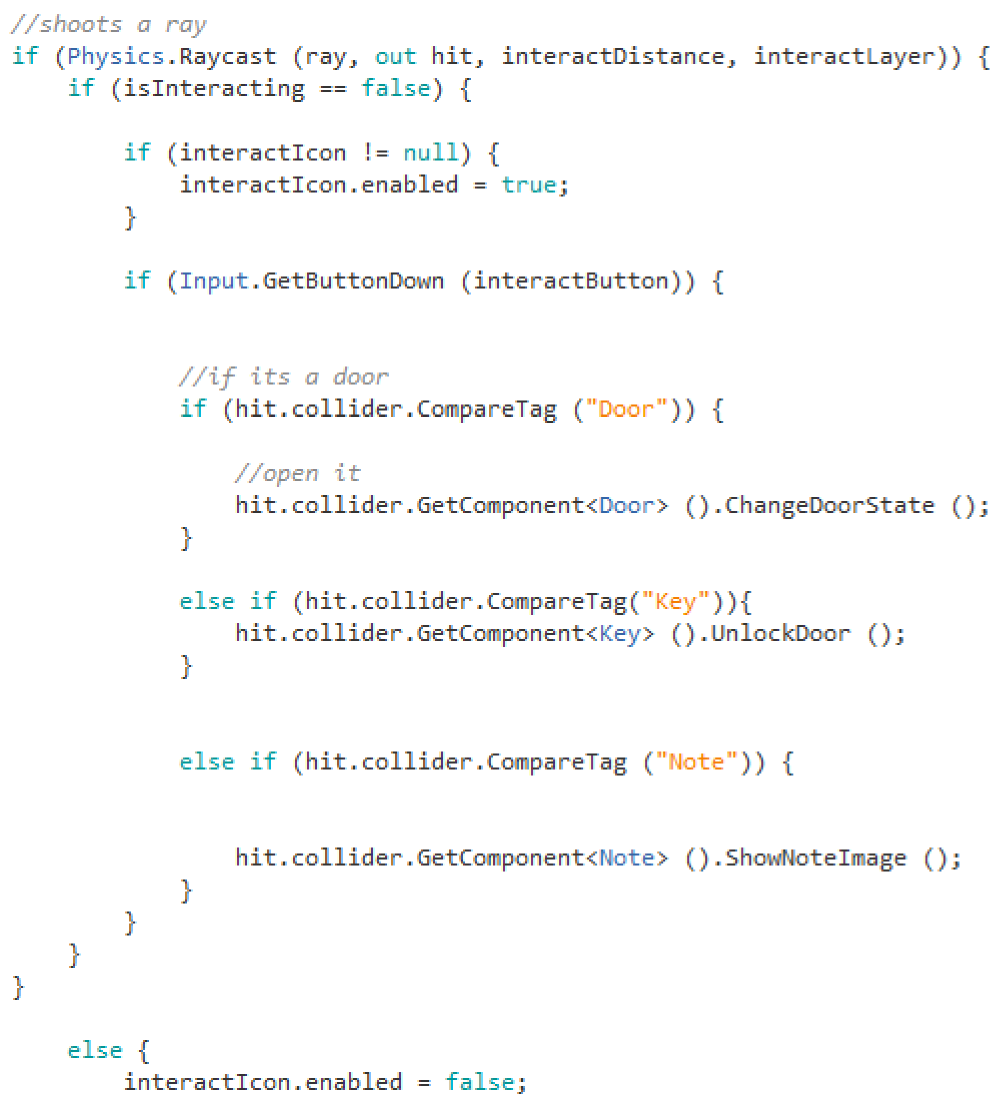
The animations will change to attack when the enemy reaches the end of the path. I also added in code so the enemy always looks at target. This prevents the character moving in behind the enemy. The enemy will apply damage to the player if it attacks.



```
    //set animation
        if(!transform.GetComponent<Animation>().isPlaying){
            tr.GetComponent<Animation> ().Play ("attack"); //will attack when gets to end of path
        }
        tr.LookAt (target.position); //cant slip behind enemy, always facing you
target.SendMessage("ApplyDammage", TheDammage);

            }
```

Interact.cs

This script is attached to a game object to make it interactable. It also sends a raycast like the shooting script, and if it hits an interactable object, an interact icon will appear to let the user know. The game objects with this script are given tags so the script can determine what to do for each object. In the code below we can see that if the object has the tag 'Door' it will open it. I wanted to make it a bit harder so the door will not unlock unless an object with the tag 'Key' has been picked up. If the tag is 'Note' then the note object will show on the screen. If the ray does not hit an interactable object, the interact icon will not show.



```
//shoots a ray
if (Physics.Raycast (ray, out hit, interactDistance, interactLayer)) {
    if (isInteracting == false) {

        if (interactIcon != null) {
            interactIcon.enabled = true;
        }

        if (Input.GetButtonDown (interactButton)) {


            //if its a door
            if (hit.collider.CompareTag ("Door")) {

                //open it
                hit.collider.GetComponent<Door> ().ChangeDoorState ();
            }

            else if (hit.collider.CompareTag("Key")){
                hit.collider.GetComponent<Key> ().UnlockDoor ();
            }


            else if (hit.collider.CompareTag ("Note")) {


                hit.collider.GetComponent<Note> ().ShowNoteImage ();
            }
        }
    }
}

    else {
        interactIcon.enabled = false;
```

Aim_sight.js

This script allows the player to aim and sight down the gun. This is useful for focusing on objects. When the game starts the aim camera is set to false. If the player right clicks the mouse, the aim camera is set to true. If the right mouse button is unclicked, the camera returns back to the normal view.

```
function Start (){
Aim = false;

if(Aim == false){
Cam.active = false;
}
}

function Update () {

if(Input.GetMouseButtonDown(1)){
Aim = true;

if(Aim == true){
Cam.active = true;
}
}

if(Input.GetMouseButtonUp(1)){

Aim = true;
if(Aim){
Cam.active = false;
}
}
}
```

Door.cs

This script is attached to interactable doors. If the door state is not locked, the door will open and play the opening sound clip. Otherwise it will play the locked door sound. If the door is set to open, it will change the rotation of the door by 90 degrees (doorOpenAngle) on the Y axis and the door will transition from a closed to open state.

```
public void ChangeDoorState () {

    if (isLocked != true) {

        open = !open;
        if (audioSource != null) {

            audioSource.PlayOneShot (openingSound);
        }


    } else {

        PlayLockedDoorSound ();
    }
}

void PlayLockedDoorSound(){
    audioSource.PlayOneShot (lockedDoorSound);
}

// Update is called once per frame
void Update () {


    if (open) {

        if (front && hasOpenedCompletely ==false){


        //open the door
        Quaternion targetRotationOpen = Quaternion.Euler (0, doorOpenAngle, 0);
        transform.localRotation = Quaternion.Slerp (transform.localRotation, targetRotationOpen, smooth * Time.deltaTime);
```

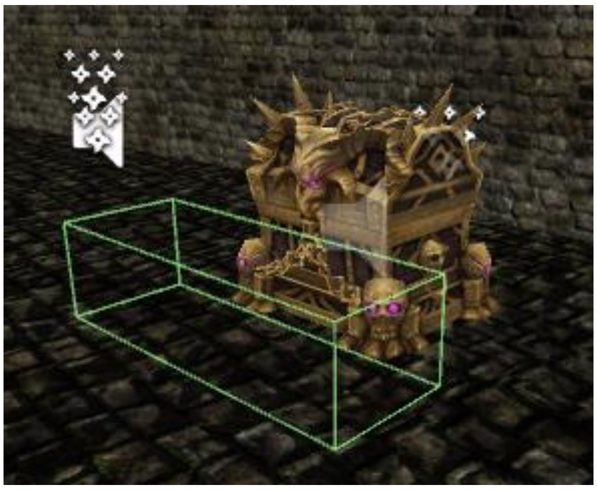
Splash.js

This script is used for changing scenes in the game. After a certain amount of seconds, the specified scene will load. If the user is impatient, they can use the space bar to go straight to the next scene and skip the waiting time.

```
function Start () {
yield WaitForSeconds (10);
Application.LoadLevel ("main menu");
}


function Update () {
if(Input.GetKeyDown(KeyCode.Space))
Application.LoadLevel ("main menu");
}
```

ChestOpen.js

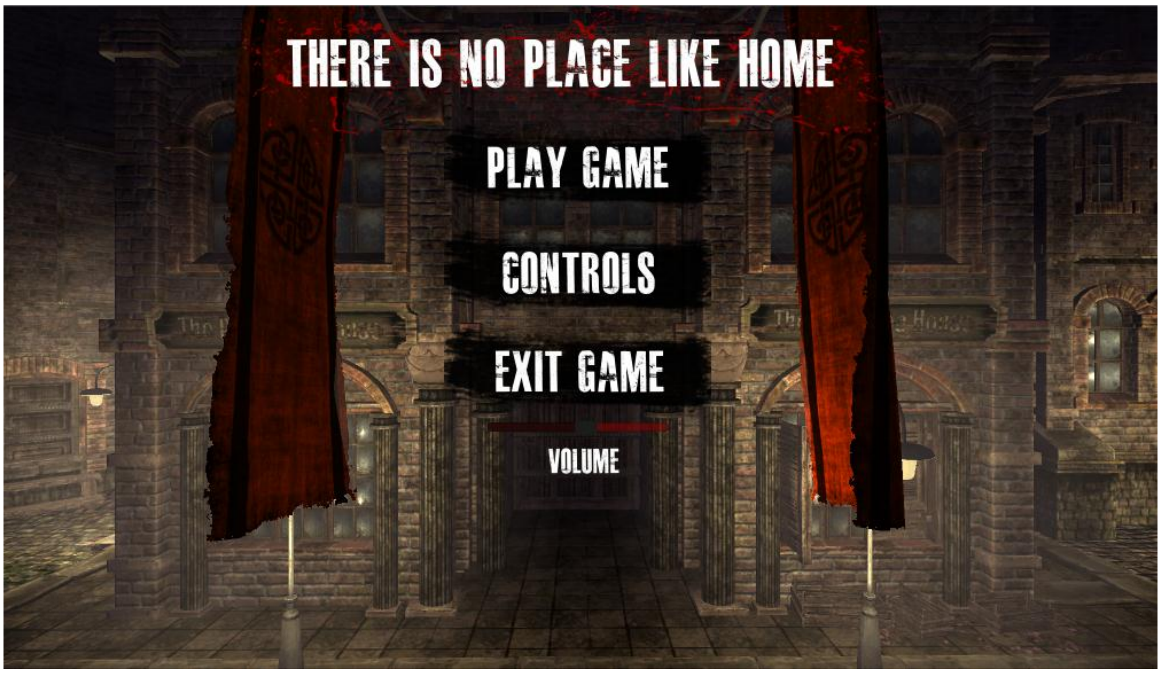
Certain game objects are given triggers. A trigger can be made by using a box collider. When the player enters this trigger, an animation, sound clip or a scene switch can occur. On my chest object I have created a trigger that will change the chest animation from closed to open and also play a sound clip to make the animation more realistic.



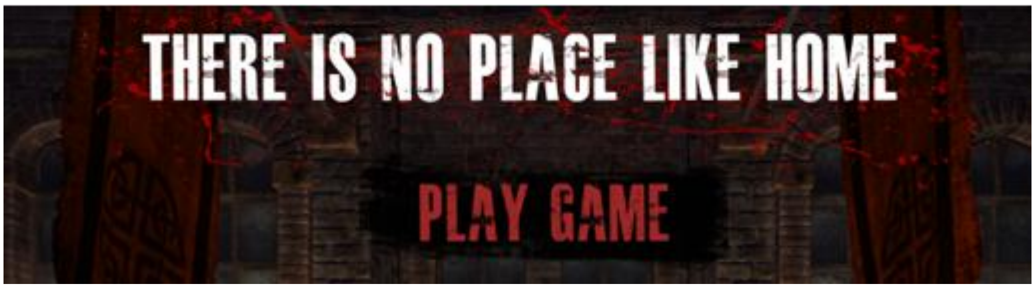
```
function OnTriggerEnter (col : Collider) {

if(col.gameObject.tag == "Player"){
AudioSource.PlayClipAtPoint(ChestSound, transform.position);
Chest.GetComponent.<Animation>().Play();
Destroy (gameObject);
}
}
```

Graphical User Interface (GUI) Layout

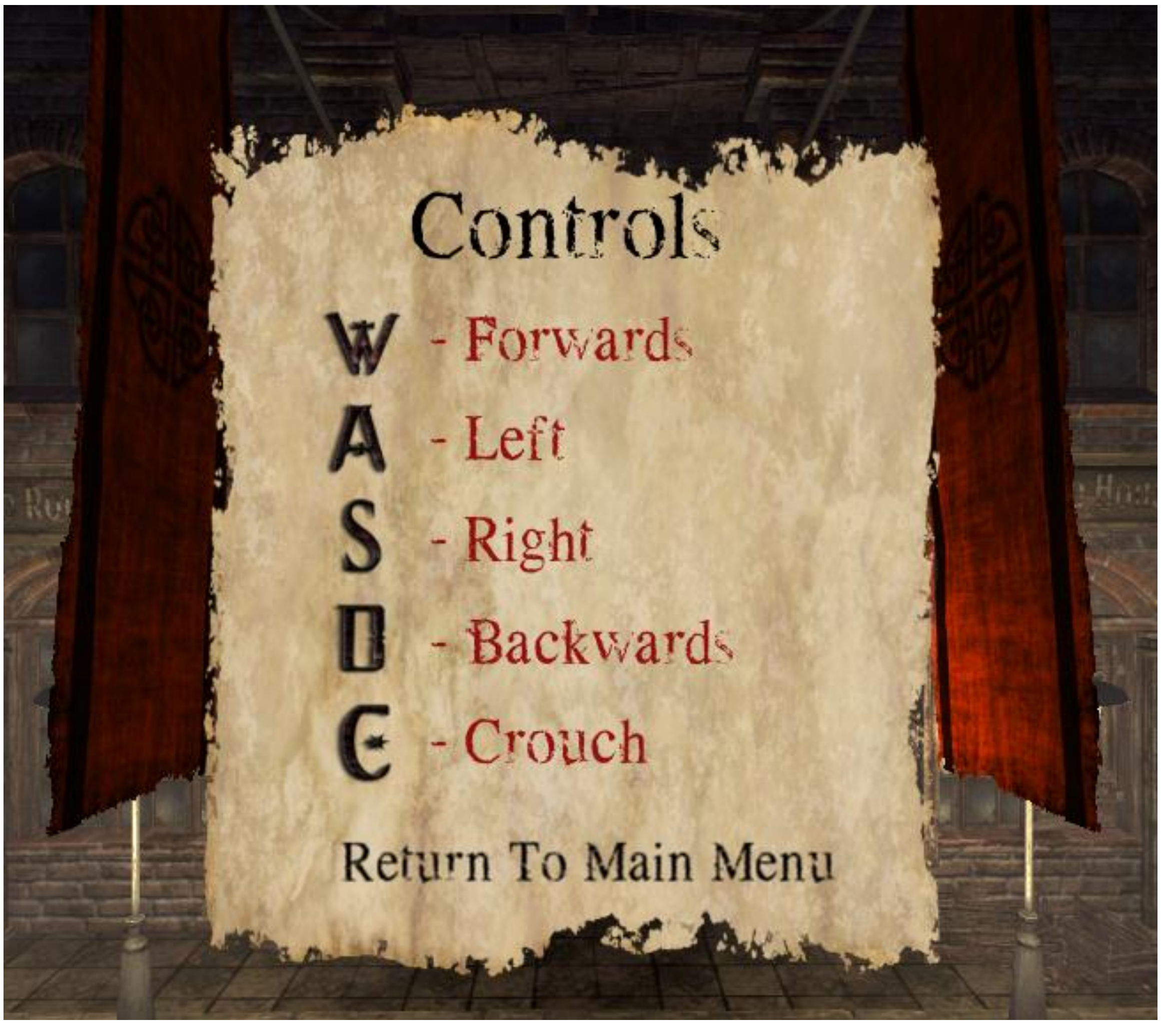
For my main menu I wanted to capture the background story of my game. I felt a simple main menu would capture the essence of the game. The main menu features the game title and 3 button options to choose from. There are options to play the game, view controls or exit the game.



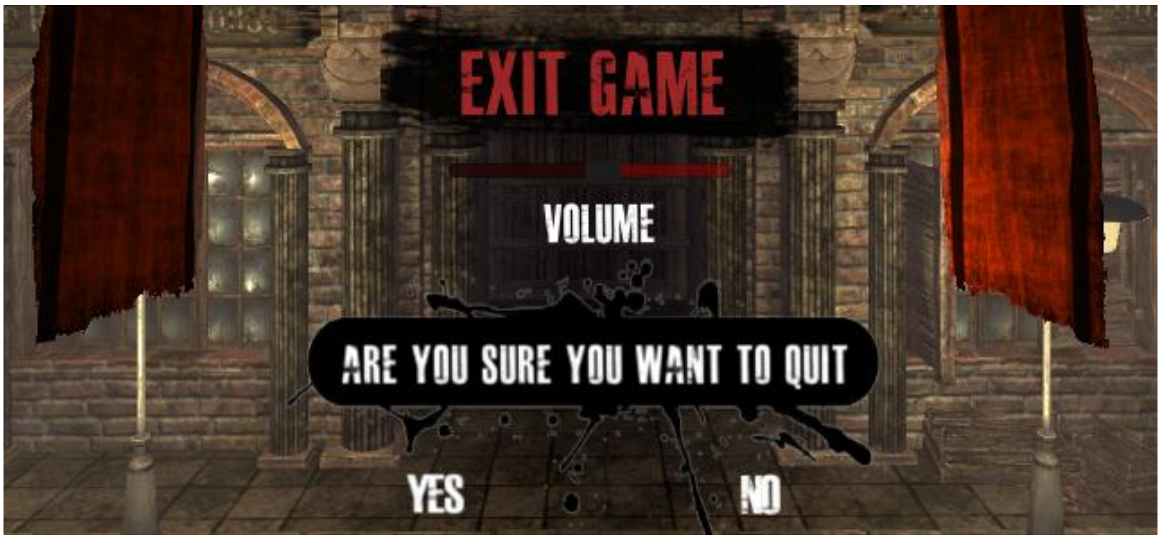
Buttons change colour on hover so the user knows they are interacting with the button and it is actually doing something. This prevents the user being confused as to whether they are pressing the button or not.



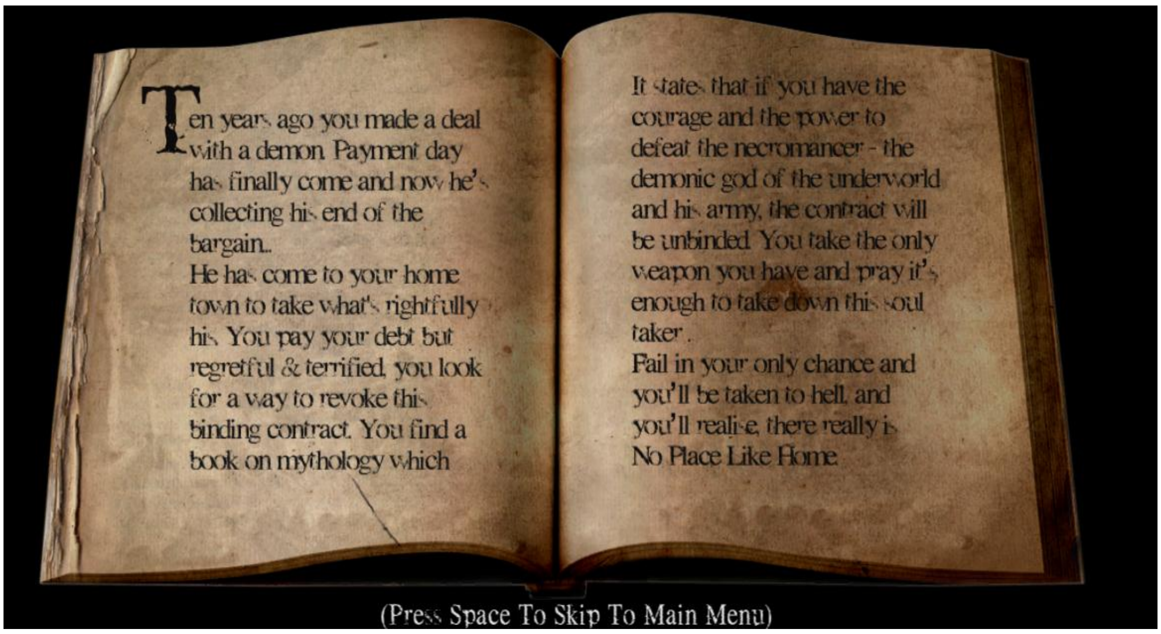
The controls button opens the controls screen which tells the user the keys for player movement. They can use the button below to return to the main menu.



The exit game button brings up a sub menu from which the user can select an option from the two. This prevents the game from closing accidently.



A short backstory is also played before the main menu which explains the story of the game. Once again, the user has the option to skip and go right to the main menu.

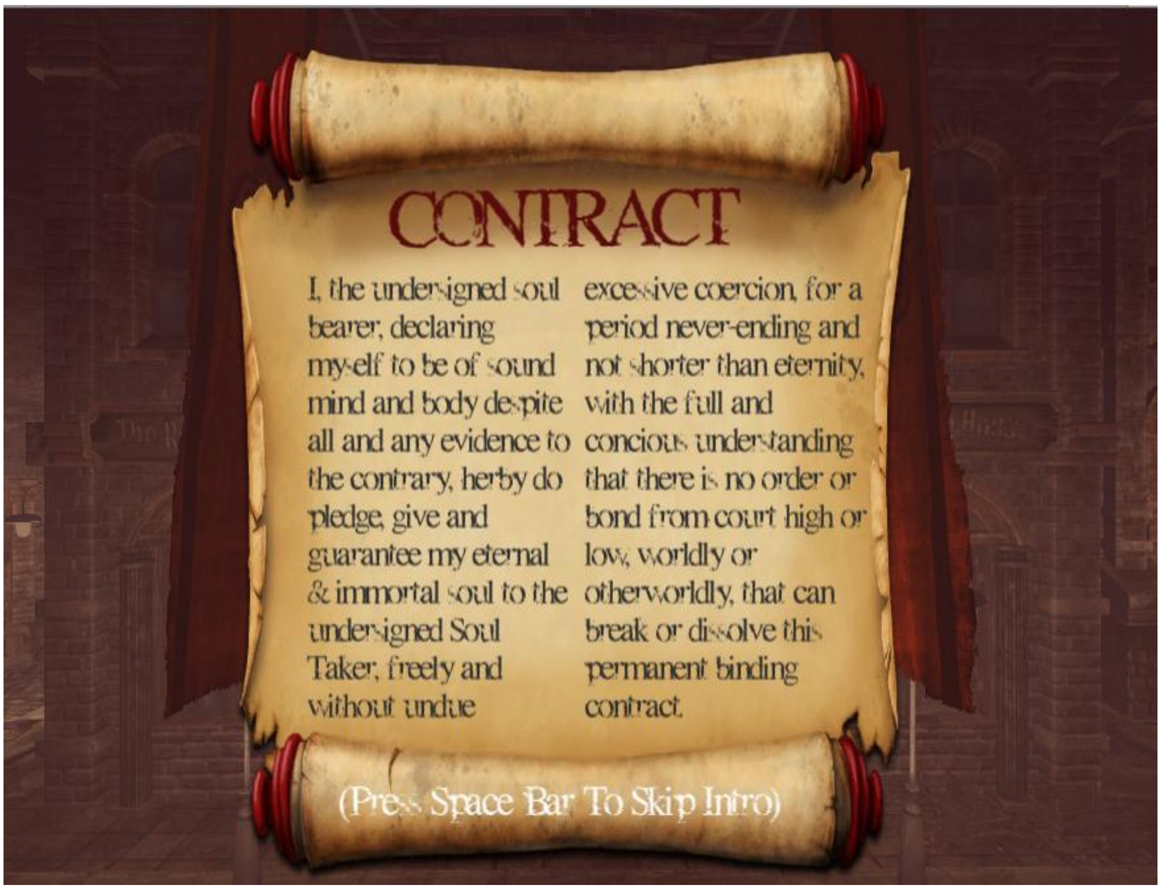


Ten years ago you made a deal with a demon. Payment day has finally come and now he's collecting his end of the bargain..
He has come to your home town to take what's rightfully his. You pay your debt but regretful & terrified, you look for a way to revoke this binding contract. You find a book on mythology which

It states that if you have the courage and the power to defeat the necromancer - the demonic god of the underworld and his army, the contract will be unbinded. You take the only weapon you have and pray it's enough to take down this soul taker.
Fail in your only chance and you'll be taken to hell, and you'll realise, there really is No Place Like Home

(Press Space To Skip To Main Menu)

When the play game button is pressed, a piece of background story is given to the character. It is set on a timer and after the amount of seconds specified in code is up it will change the scene to play the game. The user has the option to skip this by pressing the space bar which is essential to have for users that are familiar with the game.

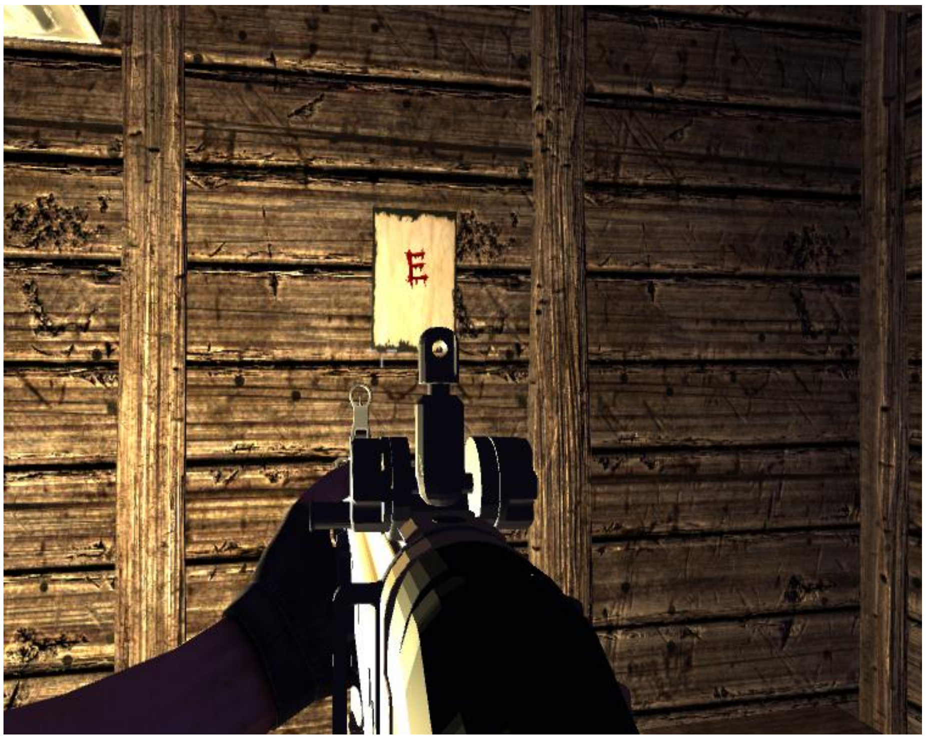


CONTRACT

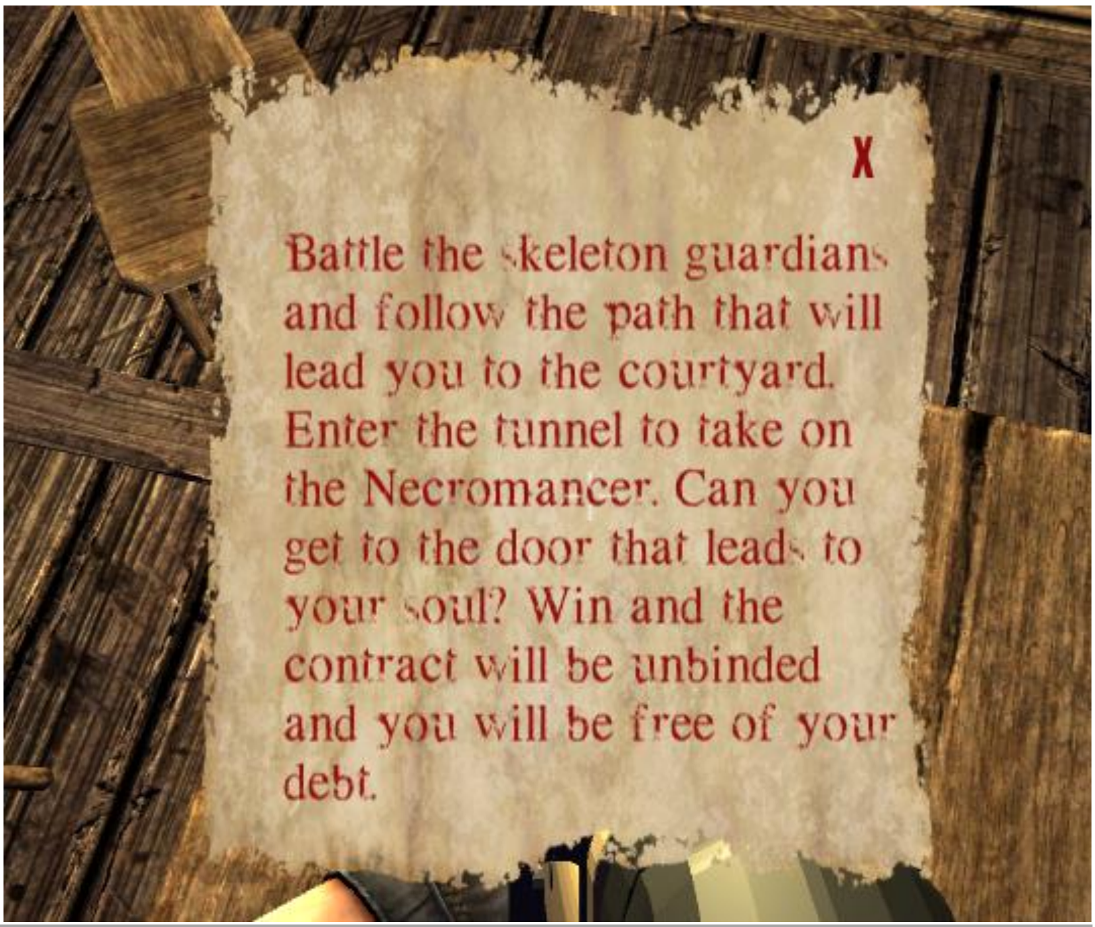
I, the undersigned soul bearer, declaring myself to be of sound mind and body despite all and any evidence to the contrary, herby do pledge, give and guarantee my eternal & immortal soul to the undersigned Soul Taker, freely and without undue excessive coercion, for a period never-ending and not shorter than eternity, with the full and concious understanding that there is no order or bond from court high or low, worldly or otherworldly, that can break or dissolve this permanent binding contract.

(Press Space Bar To Skip Intro)

Interactable objects in the game have an interact icon which shows the user which key to press to interact with the object.

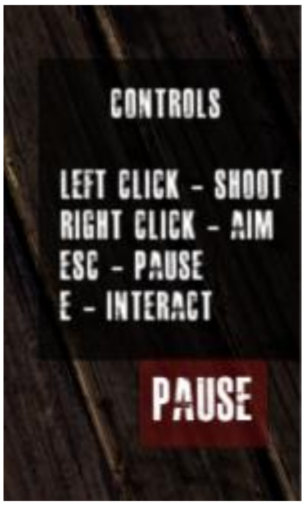


When the E key is pressed, a larger version of the note appears on screen which can be removed by pressing the X

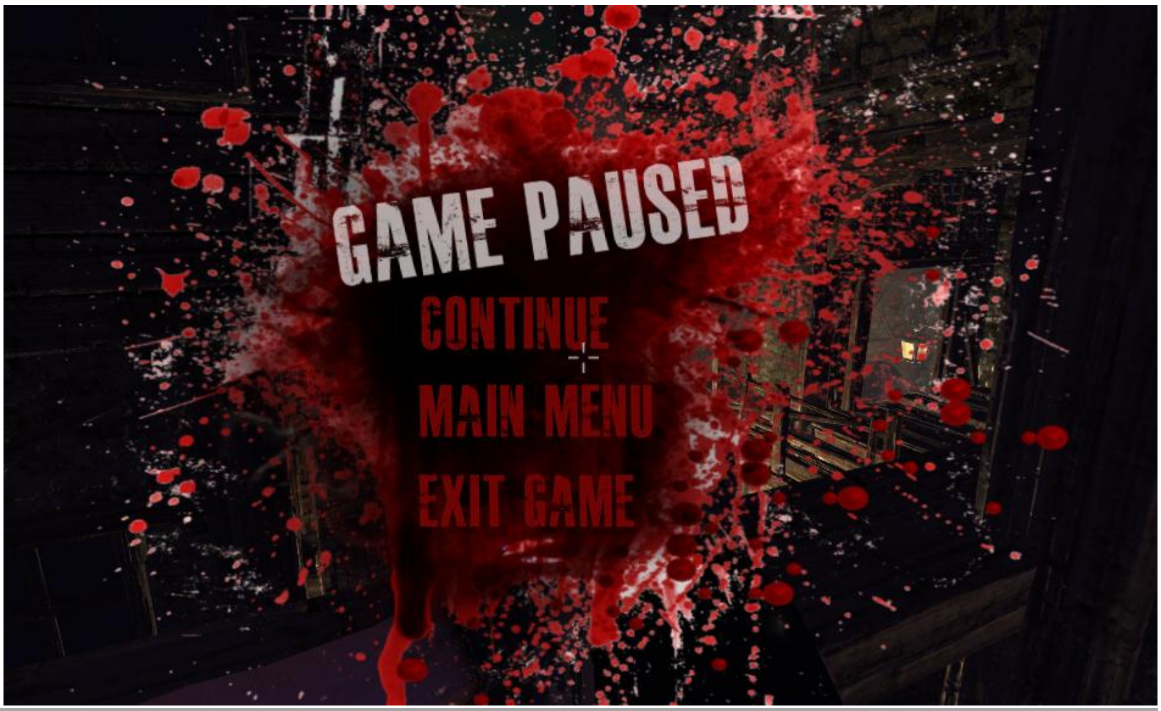

X

Battle the skeleton guardians and follow the path that will lead you to the courtyard. Enter the tunnel to take on the Necromancer. Can you get to the door that leads to your soul? Win and the contract will be unbinded and you will be free of your debt.

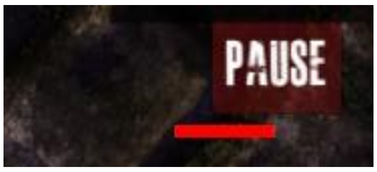
A small controls panel and a pause button are in the bottom right hand corner of the screen to provide the user with more options.

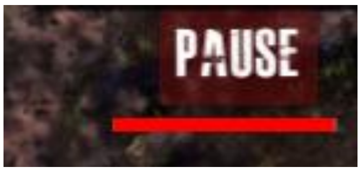


The pause menu includes three options the user can choose from. They include continuing the game, returning to the main menu and exiting the game. I am still working on removing the cross hairs from this pause menu.



A health bar also appears when the player is attacked for the first time. The more damage done to the player, the more it goes down. When the player respawns, the health bar fills to max again.





Respawn Button brings a player back to the respawn location when health gets to zero.

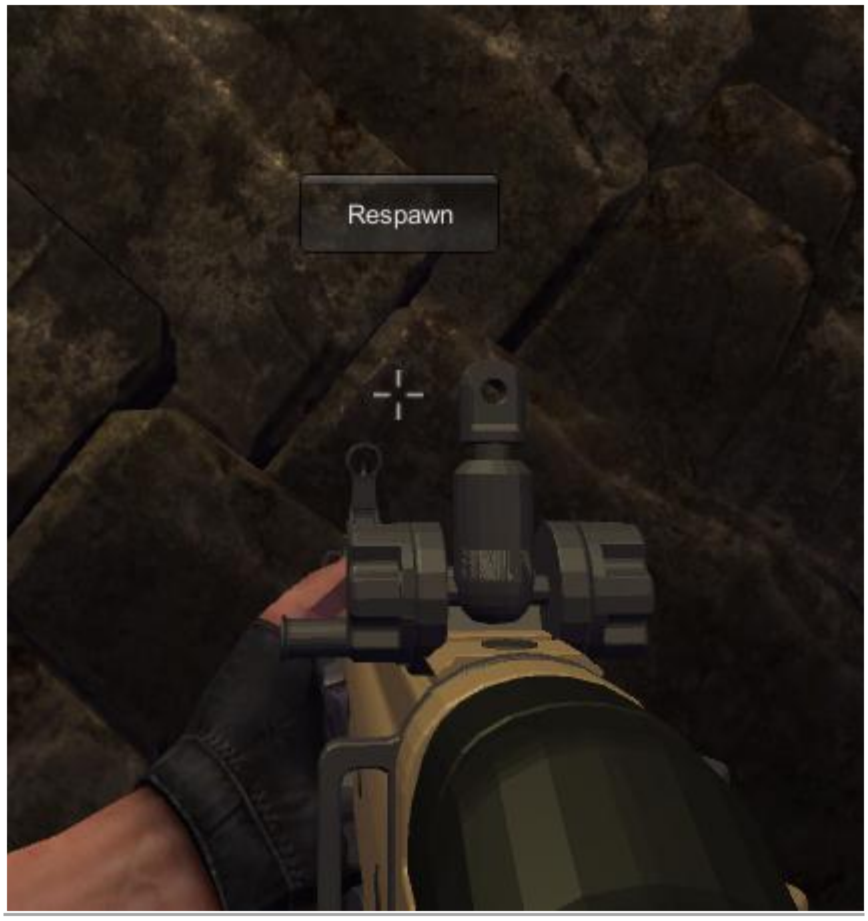
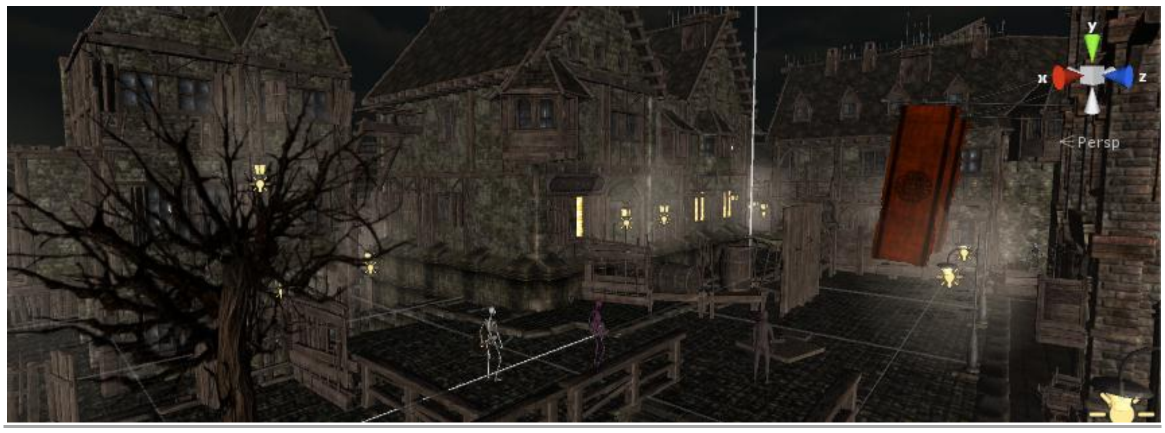
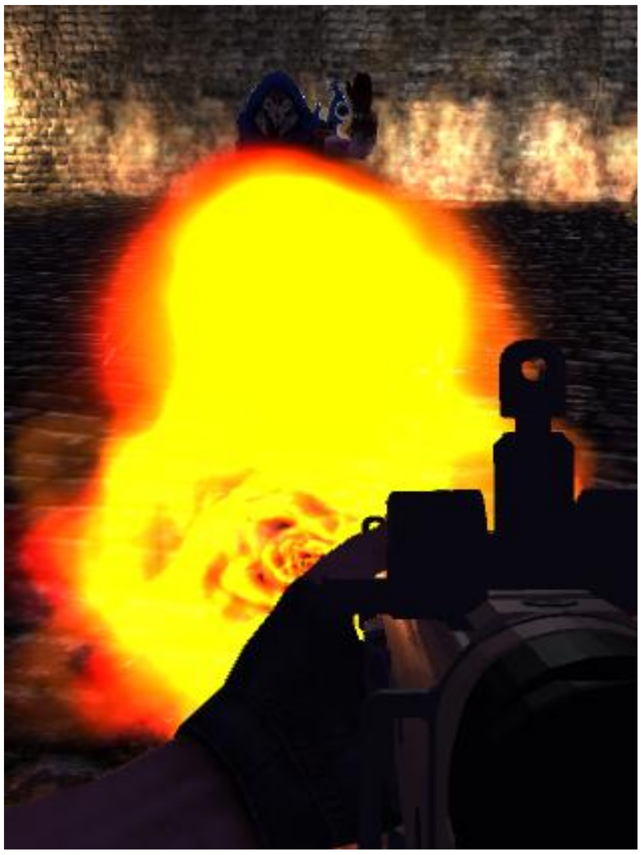


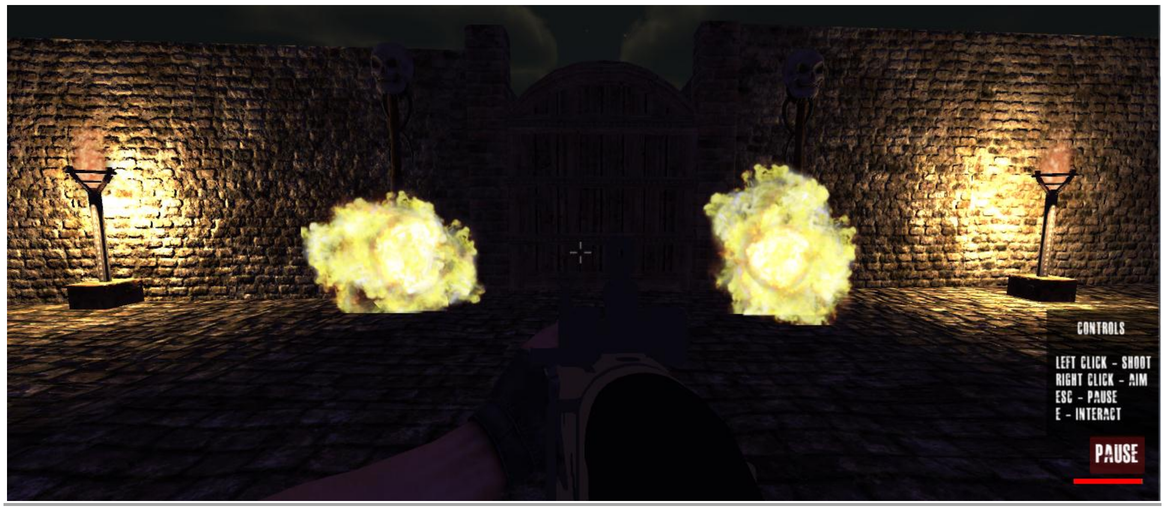
Image of the town in scene mode



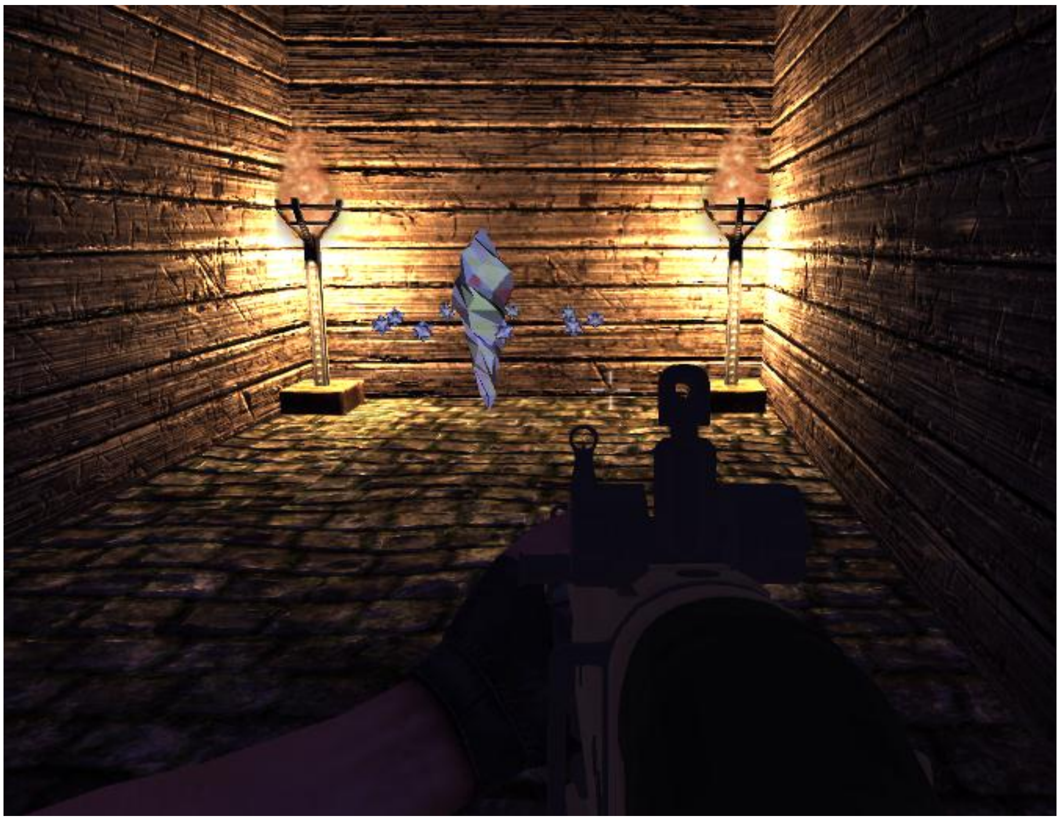
Necromancer that uses RAIN AI with mystical powers



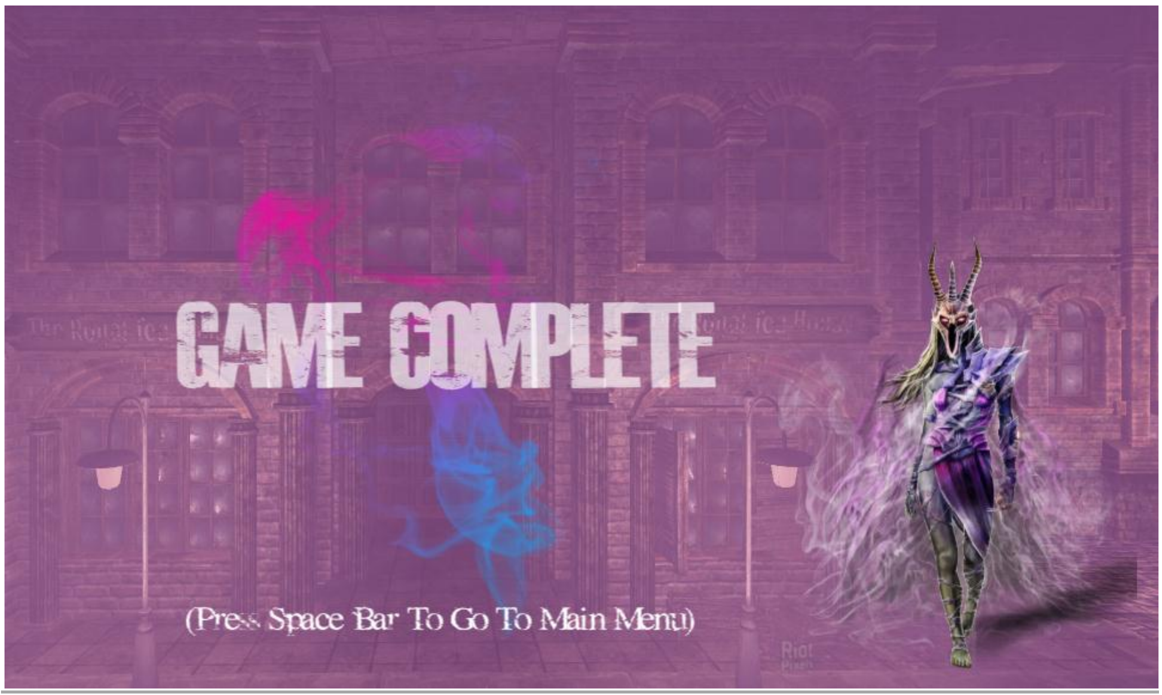
Door the player must enter to complete the game. Will only unlock with the key.



When the player collects the soul it will bring them to the game over screen



On game complete, the users is presented with this scene. I changed the colour scheme to brighter colours to show achievement and that darkness has been defeated. The main menu loads after a specified amount of seconds



Testing

Game testing is a software testing process for quality control of games. Once code is generated, software must be tested to uncover as many errors as possible before delivery to the customer. Testing was very important for this project and many different forms of testing have been carried out in order to ensure that the software acts as expected in all situations as well as ensuring that the requirements have all been successfully met.

Pathfinding Testing

The grid graph was scanned to show all walkable areas. I checked that the areas I wanted to be un-walkable were marked as obstacles and I tested this by bringing my player into objects marked obstacles and seeing if the enemy entered these areas.

Usability Testing

Usability testing was used to identify how players would interact with the overall look of the game. I gave the game to a few of my friends and asked them what they thought of my game menu and scenes. Most of the criticism I received was about the choice of font. As the font was quite hard to read in a small size I decided to only use it on areas where I could have large text and it would be easier to read. Another small problem that was identified was buttons not changing colour on hover and players didn't know if anything was happening. I had set the colour to change but then realised it didn't work on button UI's for some reason. Instead I changed it to text and added a button component. The highlight colour on hover then changed. Lastly some users had too much wait time after they had finished reading the introduction and were left waiting for a scene to load. I added in a press space bar to skip so they could change the scene to the next when they were ready.
As my terrain was so large players didn't really know where to go and ended up avoiding most of the enemies in the game. I started to wonder what stops a player from not exploring the map to its potential and just going straight for the main boss character. I decided to make a path that goes around my map. I used objects to block out shortcuts so you can't steer of course. This involved a lot of testing as I constantly had to test the paths and make sure they couldn't be avoided.

White-Box Testing

I used white box testing in order to test my logical paths through my game using specific sets of conditions and loops. I started my game from the very first scene and played it through to end to test if scenes connected the way they should. Scenes are connected by conditions that are met such as selecting a button on a menu, to waiting a certain amount of seconds or to winning the game. The goal of the white box testing was to ensure that every possible condition that could have been met was.

Integration testing

Integration testing was next after Unit testing. It was unrealistic just to test individual aspects of the game. Multiple processes within the game needed to be tested working in conjunction with eachother. I tested to see if all my different code and game objects worked seamlessly together and didn't overlap.

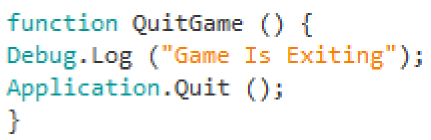
Enemy Testing

Objects in the game were disabled so I could test one thing at a time. The skeletons and the ghoul in my game use different scripts so I turned the skeletons off so I could concentrate solely on the ghoul. I tested the enemy chase script by making sure it stayed in an idle position until I got within a certain distance. I made sure the animation changed to run when he was coming after me. I checked that the enemy actually followed me and attacked when it was in range and lastly I checked that it removed player health when it attacked. I tested the enemy health script by shooting the enemy. The hit animation played and after his health went to 0, the death animation played and the ghoul disappeared.

Unit Testing

Unity has a debugger and console which can be used to monitor what the game is doing with information and functions. The console shows errors and warnings in scripts and double clicking will bring the user to the line of code that has the error.

The Debug.Log() function and Debug.Print() function were also used to check if code was actually working. An example of this would be when the exit game button is pressed. The application won't close when you are developing the game so you can use Debug.Log to send text to the console to check if the code is executing.



```
function QuitGame () {
Debug.Log ("Game Is Exiting");
Application.Quit ();
}
```

This is a form of Black Box testing as it allows you to input information into the game and monitor the output to see if its returning what is expected. The system is a ‘black-box’ whose behaviour can only be determined by studying it’s inputs and outputs. Black box testing identifies errors, shows incorrect or missing functions and behaviour and performance errors

If the text is not appearing in the console you know that the code is not being executed. You can then step through the program using the console and use these debugger functions to find the parts of the code which are causing the problem. I made sure the Debug.Log functions were working before moving on to the next part of the code. Unit testing helped ensure that individual aspects of the code worked as intended.

User Testing

As I am the developer of this game sometimes it might be easy for me to miss something so simple. It is a good idea to get someone else to play the game and give feedback. I asked a few family members and friends to test the beta version of my game. All had a different skill level, some were excellent gamers, some had programming experience and some didn't even know how to turn on a computer. The idea of this was to evaluate how the game performed when an average user who wasn't the developer tried to interact with the software. Out of the five participants, 3 were female and 2 were male. I also used different age ranges to get more accurate feedback. My little sisters aged 7 and 10 found smaller problems like a gap in the ground, while my novice mother had difficulty with the mechanics of the game and complained about the camera moving too fast. I had my 26 year old expert gaming boyfriend play the game who did everything in his power to try break it He tested the jumping mechanism and complimented how realistic it was that you couldn't jump over extremely high or large objects. The last tester was a college friend who already had a firm grasp over the control scheme and what was expected in a game. He found a small bug where an enemy was able to attack you through a wall. These are small things that I may not have identified if testing had not taken place.

Overall the feedback was positive but there were a few small problems such a lag. This is a problem in gaming but as my project was so large and many users had old or not top of the range model laptops there was very little I could do about this.

This feedback showed that novice gamers needed more support in controls. I added a small control panel to the game screen so if any of the buttons were forgotten, they were easily visible without having to go back to the main menu. I made this panel small so it would provide help but not distract more experienced gamers.

Customer Testing

I think the key to great and simple feedback is a survey. I wanted to make a list of information that I wanted to find out from the play testers and prioritize them. I wanted to keep the questions specific and detailed as they are easier to answer than broad or open ended questions. I wanted to get an indication of people's true reaction to the game. I chose the first question in my survey because I wanted their initial gut reaction. I wanted to identify potential problems in my game so I made these questions priority over questions like 'how can this game be improved?'

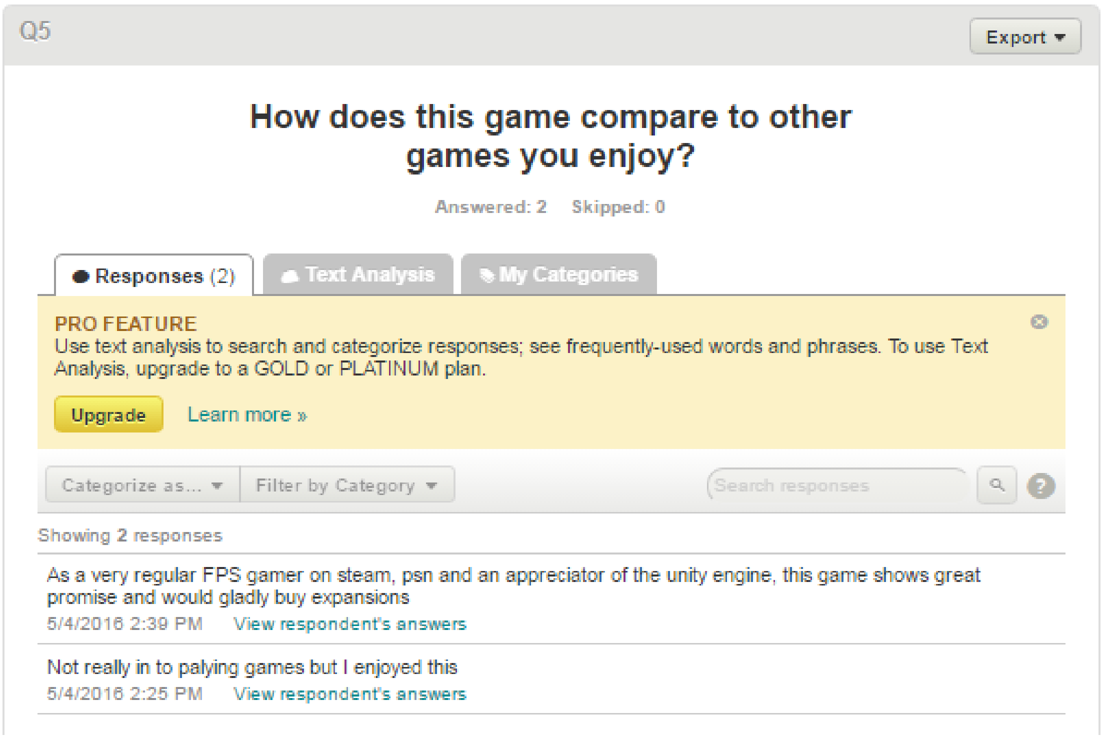
The questions asked were:

- On a scale of 1-5, 5 being best, how would you rate the quality of your experience with this game overall?
- About how long did it take to set up the game?
- On a scale of 1-5, 5 being best, how clear were the explanations for the controls of the game?
- Were there any phases of the game or elements of game play which stood out as engaging?
- How does this game compare to other games you enjoy?
- How would those similarities and differences affect your likelihood, either positively or negatively of buying this game?
- About how long did each play take?
- What parts did you like the best/least?
- Is this a game you see yourself playing frequently? If not, why?
- Did the game feel like it was missing a crucial element?
- How many times did you play the game?
- (If more than once) was the game more enjoyable, less enjoyable or equally enjoyable during subsequent plays.

In order to publish my survey I had to remove two questions. The survey can be found at this link https://www.surveymonkey.com/r/3LCJBLK

I received some useful feedback from the survey. This player enjoyed the fact that the enemies only came after you when you were within a certain distance. This gives players a chance to enjoy the game as they are not being attacked throughout the whole game play.

I also got feedback from a user not into gaming but they would recommend it to others. The honesty in beneficial in surveys as you can see the users who are not interested but they are giving positive feedback.



How does this game compare to other games you enjoy?

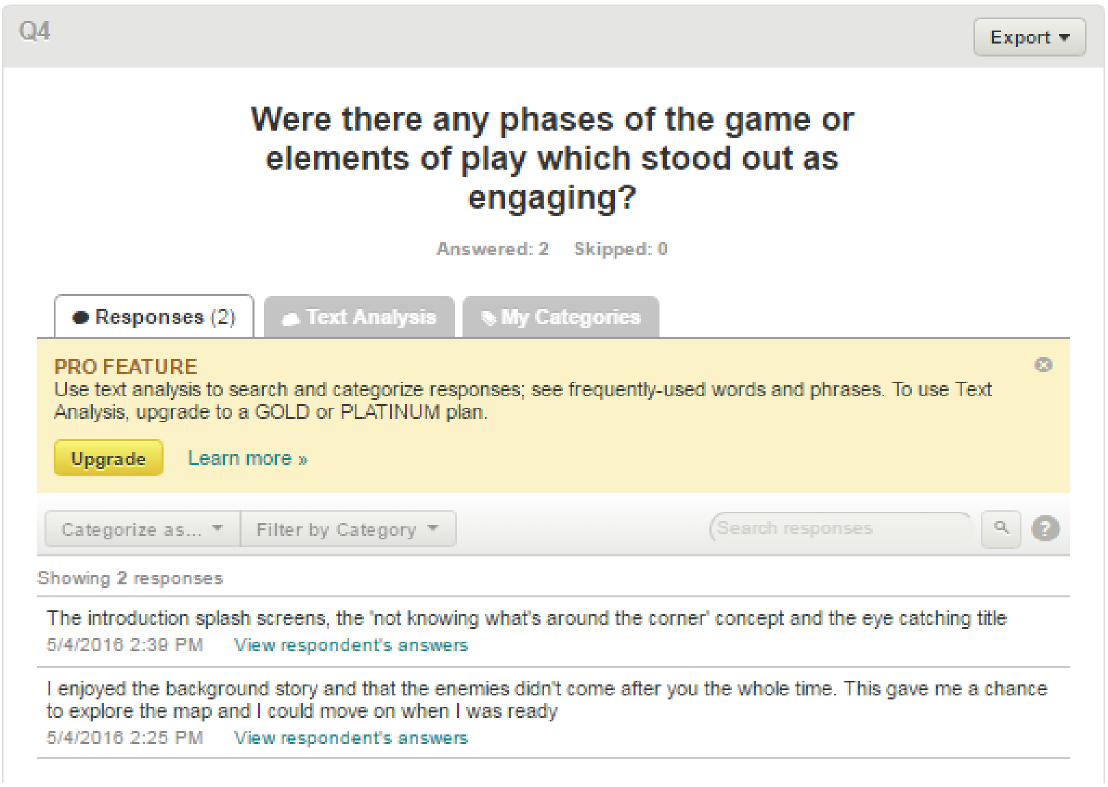
Answered: 2 Skipped: 0

PRO FEATURE
Use text analysis to search and categorize responses; see frequently-used words and phrases. To use Text Analysis, upgrade to a GOLD or PLATINUM plan.

Showing 2 responses

As a very regular FPS gamer on steam, psn and an appreciator of the unity engine, this game shows great promise and would gladly buy expansions
5/4/2016 2:39 PM View respondent's answers

Not really in to palying games but I enjoyed this
5/4/2016 2:25 PM View respondent's answers



Were there any phases of the game or elements of play which stood out as engaging?

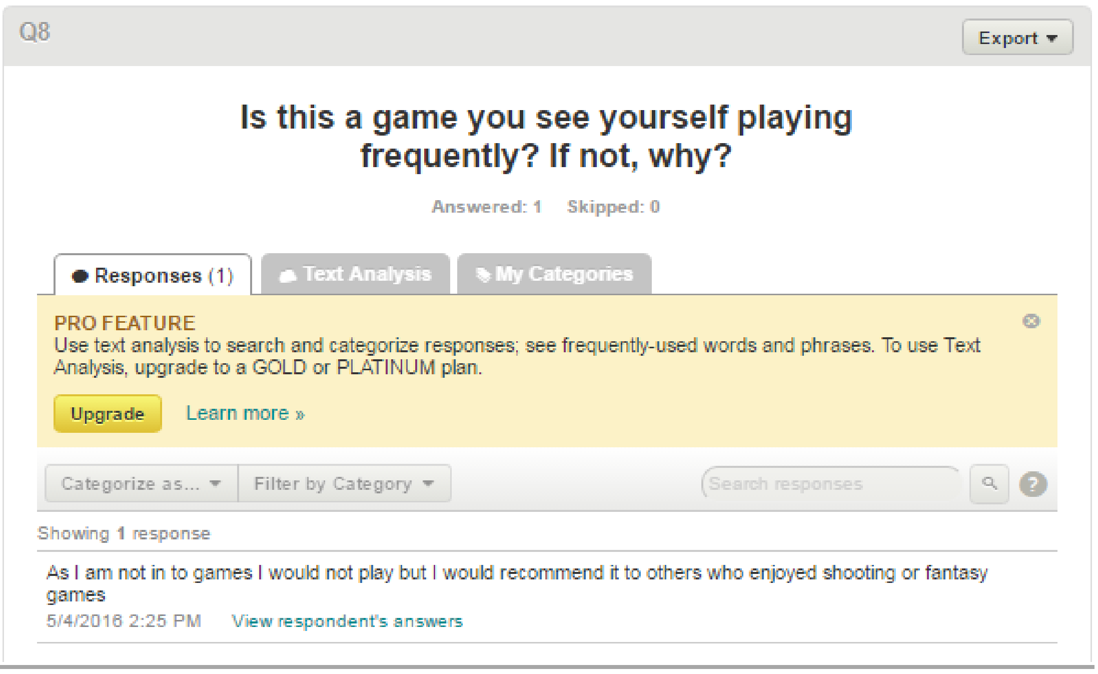
Answered: 2 Skipped: 0

PRO FEATURE
Use text analysis to search and categorize responses; see frequently-used words and phrases. To use Text Analysis, upgrade to a GOLD or PLATINUM plan.

Showing 2 responses

The introduction splash screens, the 'not knowing what's around the corner' concept and the eye catching title
5/4/2016 2:39 PM View respondent's answers

I enjoyed the background story and that the enemies didn't come after you the whole time. This gave me a chance to explore the map and I could move on when I was ready
5/4/2016 2:25 PM View respondent's answers


Q8

Export

Is this a game you see yourself playing frequently? If not, why?

Answered: 1 Skipped: 0

PRO FEATURE
Use text analysis to search and categorize responses; see frequently-used words and phrases. To use Text Analysis, upgrade to a GOLD or PLATINUM plan.

Showing 1 response

As I am not in to games I would not play but I would recommend it to others who enjoyed shooting or fantasy games
5/4/2016 2:25 PM View respondent's answers

Evaluation

So far in my evaluation I have continuously checked the console for errors to make sure the game is running smoothly.

Myself and testers repeatedly played my game to try cheat it. Some of the objects I used to block paths were easy to jump through and this made it easy to skip half of the game. I changed these objects into larger objects such as fences or doors. After receiving feedback from the final product I noticed that most people said they enjoyed the game and would recommend it to others.

Conclusions

Advantages

In my opinion there are many advantages to creating this type of project. It is a chance to do something different, step outside my comfort zone and learn how games are created. I liked that I had full freedom with the creativity of the game and I could implement it with my vision in mind. Another advantage was the experience of coding in Javascript and C# in previous college modules. This was a benefit to me as these are the two languages used to code in Unity. I was pleasantly surprised with how much coding experience I managed to pick up. As someone who always struggled a bit with coding I tried to challenge myself by fixing errors without online help. I was able to fix these errors and after a while I was able to write whole scripts without help. This is something I did not expect to happen and it has made me more confident and eager to create more games in the future. I also got to play around with new technologies including RAIN AI and pathfinding algorithms.

Lots of FPS tutorials were available online and this made it easy to get inspiration and get my project off the ground. The genre of game that I chose did not limit me and I was free to make it as realistic/simple as I desired. I incorporated elements that were not planned like interactable objects, locked doors and object animation. The asset store was also a huge help in getting realistic models into my game. It also gave me more time as I did not have to make these models from scratch. I really enjoyed creating the scene and making the GUI for the project. It was a chance to be creative and test out different ideas.

Disadvantages

There are some disadvantages associated with the development of a FPS game. It was evident during development that this type of game is extremely large and complex to develop thus requiring a substantial amount of time to even get it to a playable state. As I have no previous experience in game development, or Unity, it meant I had to learn how to use and navigate around a new program. I spent a lot of my initial time watching tutorials and getting myself familiar with Unity. I had basic C# skills but this was not enough so research was needed for this.

This lack of knowledge also proved to be a challenge again when it came to the requirements specification aspect of the project. I found it difficult to create the class diagrams and use cases as I did not know how to compare it to past projects. The biggest disadvantage for me was not realizing the effort it takes for a laptop with a basic graphics card to run a project of this size.

My laptop constantly crashed when I try to run Unity. This has made the creation of my game very difficult. I am also limited in college as only one classroom has Unity installed and it is not always available.

Summary

Overall I have really enjoyed developing this project. I have had a few set-backs including the screen of my laptop falling off its hinges and having to find a replacement in a short amount of time but I managed to create a finished product and I am extremely proud and happy of what I have accomplished. I did not expect my project to turn out as well as it did and with even more time I could have improved it further. I plan to continue development of this project in the future as I would love to see how far I could take it. I have got experience with a new program and it has aspired me to create more games. I really enjoyed the environment design aspect of the game. I thoroughly enjoyed creating the GUI and the menus for the game also. The game was a challenge for me but I am happy I took it on. I have also gained experience in time management and goals as this is the longest time frame I have had a project spread over. I have learned the benefits of starting early as I am usually someone who leaves it till nearer the time and works better under pressure. I feel this experience will stand to me and I can apply what I have learned to other problems I will face after I leave college.

There are many ways that this project can be expanded in the future. I could create a new world with new characters and have the storyline continue on. I could make the game more challenging by adding in a more difficult enemy to defeat, adding more levels or by adding in obstacles to overcome like timers, missions etc. I could expand on weapons and have the option to switch weapon while playing the game. As I did not have time to incorporate a save feature, this is something I would definitely implement in the future.

A multiplayer option could also be introduced which would expand the game. It would have to be published to a server where users can play as a team and help eachother in the battle. A difficulty option could also be put in so users stay challenged and interested.

A website could also be created to compliment the game. News and updates would be available on this. Users could write a short review of the game and give a rating. This feedback would be helpful in providing a better game play experience.

I feel that I have not limited my options with the style of game I have chosen. The possibilities are endless and the story can be developed. I can re-use the environment in as many levels as I wish.

The completed game could also be submitted to Steam's Greenlight service. With Steam Greenlight, games are submitted with their descriptions, videos, screenshots, etc. The users of Steam vote for the game to be sold through the Steam store if they think it would be worth buying. When the game has received enough votes it will then be added to the Steam marketplace where users can purchase and play your game.

References

Apa.org. (2016). [online] Available at: http://www.apa.org/monitor/2014/02/video-game.aspx [Accessed 4 May 2016].

Rival{Theory}. (2016). *Features - Rival{Theory}*. [online] Available at: http://rivaltheory.com/rain/features/ [Accessed 5 May 2016].

star, D. (2016). *Difference and advantages between dijkstra & A star*. [online] Stackoverflow.com. Available at: http://stackoverflow.com/questions/13031462/difference-and-advantages-between-dijkstra-a-star [Accessed 5 May 2016].

YouTube Tutorials

YouTube. (2016). *SadSmile Games*. [online] Available at: https://www.youtube.com/channel/UCU9oBhCHSC8xWQ4_tSkR1fQ [Accessed 2016].

YouTube. (2016). *CodersExpo*. [online] Available at: https://www.youtube.com/user/CodersExpo [Accessed 2016].

YouTube. (2016). *MisterNinjaBoy*. [online] Available at: https://www.youtube.com/user/misterninjaboy [Accessed 2016].

Pathfinding

Arongranberg.com. (2016). *A* Pathfinding Project*. [online] Available at: http://arongranberg.com/astar/ [Accessed 12 Mar. 2016].

Audio

Koenig, M. (2016). *Free Sound Clips | SoundBible.com*. [online] Soundbible.com. Available at: http://soundbible.com/ [Accessed May 2016].

Appendix

Project Proposal

Objectives

My main objective of my final year project is to create a 3D gaming application within the Game Engine Unity. I am using version 5 of the Unity software. My main goal is to make a fully functional game with different levels of difficulty.

My idea for this project is to create a FPS (First Person Shooter) game where the player will move through the 3D environment. This is centred on gun and projectile weapon-based combat through a first-person perspective, that is, the player experiences the action through the eyes of the protagonist.

The main idea of the game is to have a wave based system. In the first wave there will be a small number of enemies to defeat. When all enemies are defeated, wave 2 will begin where more enemies are introduced to the game. Waves will become successfully more difficult. In between waves a stronger boss character may be introduced and you will not be able to continue until he is defeated. My main objective of this game is for the player to defeat the given amount of enemies without letting his/her health deteriorate which will end in game over.

I aim to make this game with the use of 3D graphics as I feel this will make the game more appealing for players. It will also enable me to create a more realistic environment. I plan to make use of the Unity asset store and use some freely available assets (skybox for example) while also building my own terrain including mountainous land, a beach scene and a forest environment.

I am thinking I will use a mixture of C# and Javascript to program my project as I have some experience using this from my modules in the past few years.
I already have some knowledge of C# from my previous years at College and this will help me with creating the fundamental concepts that will underline the beginning stages of creating this project.

Background

For my final year specialization I decided to choose Gaming and Multimedia. I chose this as I had a multimedia module in first year and I found it was something I was good at and I enjoyed. As I wanted my final year project to be in this specialization area I decided to create a game. This will be a challenge for me as I have no experience with game engines or game making overall. I have chosen to do this as I want to step outside my comfort zone and create something new. I am excited to learn how to create a project in the gaming area and I feel if successfully completed, will add variety to my CV. I have experience with mainly websites and I would like to expand on that. It is also an opportunity to step into the career path of multimedia and gaming as there are many career opportunities in the gaming sector and I feel it would be an interesting and creative area to work in.

Although I am not a gamer I have played a few games throughout the years, my favourites being FPS games like Call of Duty and racing games like Need for Speed. As I have little experience with games I had a hard time choosing what genre of game to create but I knew these were the two main topics I was interested in. I had no knowledge of what was involved in creating these kinds of games so I asked for advice from a colleague and after getting insight from someone with more experience than I had, I decided to choose the shooting style game as there is more room for expansion and levels of difficulty. This style of game is very popular among gamers. In the 21st century, the first-person shooter is the most commercially viable video game genre, and has more market share of any other genre in the gaming industry with best sellers like Battlefield, Bioshock and GTA.

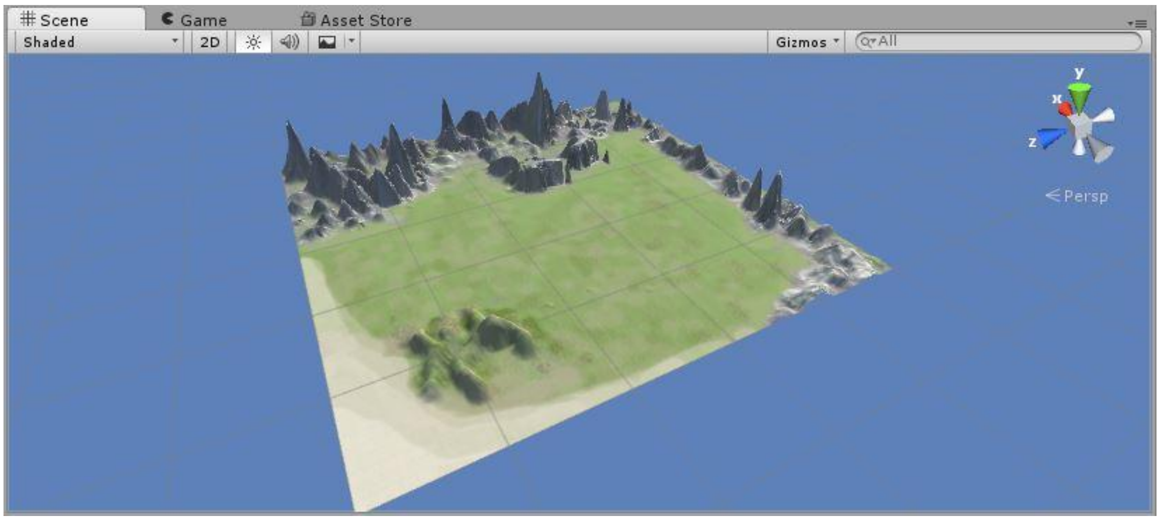
I chose Unity as the game engine to create this game as there is the option to use the game engine for free, with the free version being feature complete. It is multi-platform which means it can be published on many different types of operating systems. Unity is fairly easy to use but if you get stuck there is a great deal of resources available online for beginners such as tutorials and forums. There is also the asset store which allows you incorporate models into your game without making them yourself.

Technical Approach

Research:

Before I started on my project I made sure to fully research what is involved in creating a game. As I have never created a game or used the Unity platform, I had to make sure I was making the right decision for my final year project. I wanted to make sure I could create a project I could complete and feel proud of. I watched many YouTube tutorials on how to use Unity, on different types of games and what was involved in creating them. These tutorials helped me feel more comfortable and confident in my decision. I felt that I would not have any limits with this project topic and there would also be help online to get my project goals accomplished. As I mentioned there was 2 game genres I was interested in. I made my decision to go with a first person shooting game. I watched many YouTube tutorials which showed me many different ways this style of game can be executed. I have the option to go big or small with this genre and I was given the confidence that I would be able to undertake and complete this game in the college year.

After I decided on the platform and game genre I started to decide on my terrain layout. I chose to go with a mountainous and grassy terrain as I don't want the levels overwhelmed with landscape. As I have chosen to go with a wave style game I can re-use the same background if I wish. After I finish my terrain I will start working on the functionality of the player and the enemies that will attack.



Work started on my terrain

From the tutorials and research online I hope to implement the gameplay elements that I need for the game. These elements would include attacking, a health bar and the amount of enemies left to kill.

Special resources required

FOR DEVELOPMENT:

OS: Windows XP SP2+, 7 SP1+, 8, 10; Mac OS X 10.8+.

GPU: Graphics card with DX9 (shader model 2.0) capabilities. Anything made since 2004 should work.

Additional platform development requirements:

- iOS: Mac computer running minimum OS X 10.9.4 version and Xcode 6.x.
- Android: Android SDK and Java Development Kit (JDK).
- Windows 8/8.1 Store Apps / Windows Phone 8/8.1: 64 bit Windows 8.1 Pro and Visual Studio 2013 Update 2+.
- WebGL: Mac OS X 10.8+ or Windows 7 SP1+ (64-bit editor only)

I don’t feel I will have the need for any special resources to complete this project. My laptop meets the system requirements in order for Unity development.

Technical Details

The language I am planning to use for my project is C#. This is a standard language that the Unity Game Engine uses for the creation of games.
I plan to make use of the Unity asset store for most of the models in my game but as I am learning how to use Unreal Engine this year in my multimedia module I might attempt to make a model and import it into my game.

Evaluation

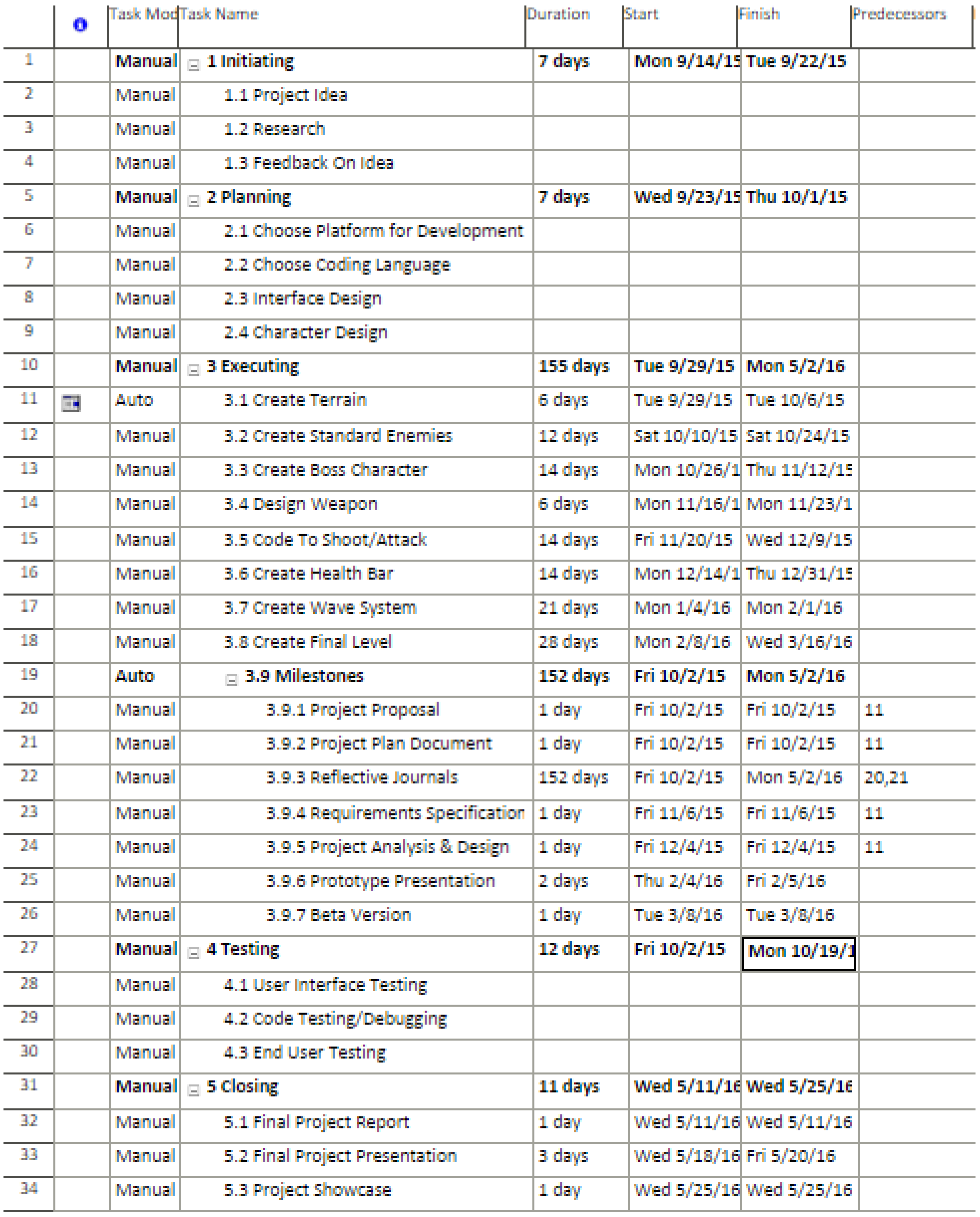
My plan for the evaluation of my project is to test my game by playing it. My aim is to have a fully functioning health bar for my character which I can test by letting enemies attack. If the health bar drains, the game will be over. I also want to be able to attack the enemies using the gun so I will test that by shooting at them and seeing if their heath reacts the way I want. I will test my wave system by killing the amount of enemies numbered on the screen and seeing if it moves on to the next round. I will continue to test this and see if I can make it to the final round and win the game.

When I have finished testing every possible outcome of the game I will let an end user play the game. I think it's important to get different eyes and hands looking and playing your game, trying new things. This will be beneficial as someone else might play the game a different way to the way I play the game so they might find flaws or errors that I would not have found. I also think it's important to get feedback off users as I want to know if my game is easy to use, friendly, and enjoyable.

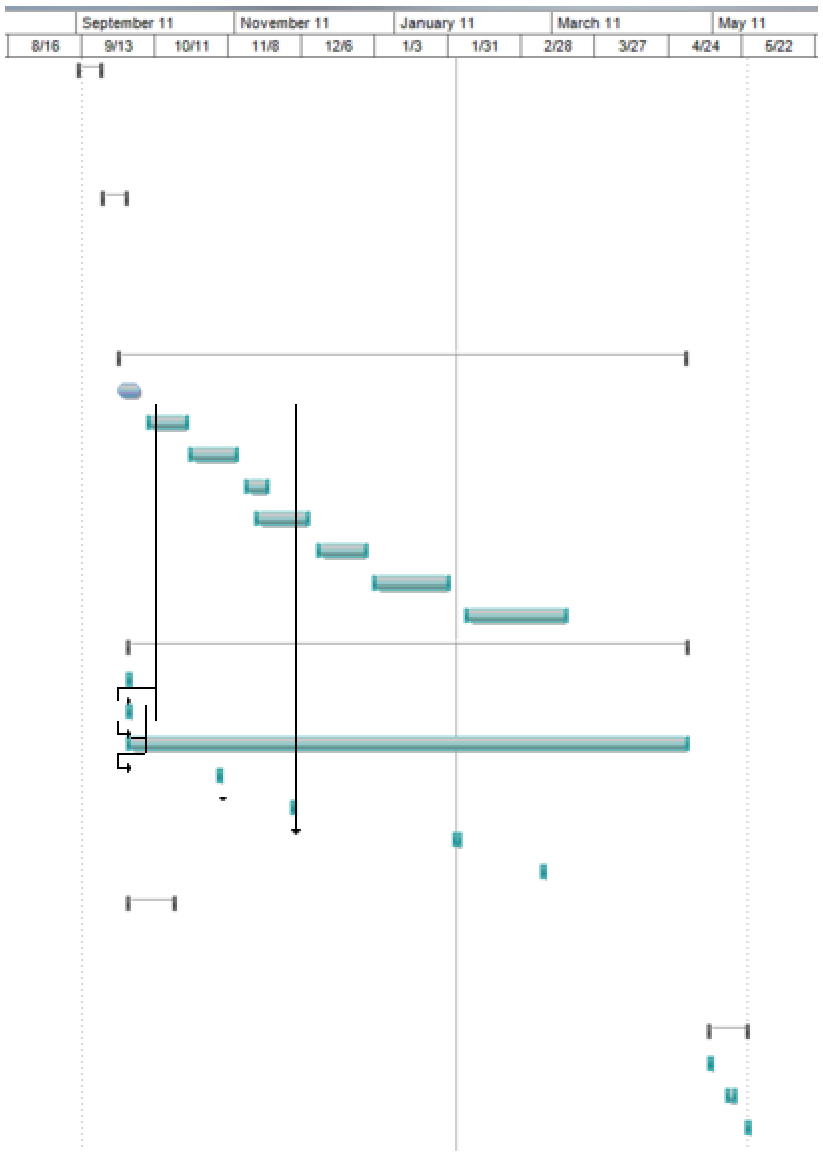
Kelly McGuinness 23.09.2015

Signature of student and date

Project Plan



	ℹ	Task Mod	Task Name	Duration	Start	Finish	Predecessors
1		**Manual**	**⊟ 1 Initiating**	**7 days**	**Mon 9/14/15**	**Tue 9/22/15**	
2		Manual	1.1 Project Idea				
3		Manual	1.2 Research				
4		Manual	1.3 Feedback On Idea				
5		**Manual**	**⊟ 2 Planning**	**7 days**	**Wed 9/23/15**	**Thu 10/1/15**	
6		Manual	2.1 Choose Platform for Development				
7		Manual	2.2 Choose Coding Language				
8		Manual	2.3 Interface Design				
9		Manual	2.4 Character Design				
10		**Manual**	**⊟ 3 Executing**	**155 days**	**Tue 9/29/15**	**Mon 5/2/16**	
11	▦	Auto	3.1 Create Terrain	6 days	Tue 9/29/15	Tue 10/6/15	
12		Manual	3.2 Create Standard Enemies	12 days	Sat 10/10/15	Sat 10/24/15	
13		Manual	3.3 Create Boss Character	14 days	Mon 10/26/1	Thu 11/12/15	
14		Manual	3.4 Design Weapon	6 days	Mon 11/16/1	Mon 11/23/1	
15		Manual	3.5 Code To Shoot/Attack	14 days	Fri 11/20/15	Wed 12/9/15	
16		Manual	3.6 Create Health Bar	14 days	Mon 12/14/1	Thu 12/31/15	
17		Manual	3.7 Create Wave System	21 days	Mon 1/4/16	Mon 2/1/16	
18		Manual	3.8 Create Final Level	28 days	Mon 2/8/16	Wed 3/16/16	
19		**Auto**	**⊟ 3.9 Milestones**	**152 days**	**Fri 10/2/15**	**Mon 5/2/16**	
20		Manual	3.9.1 Project Proposal	1 day	Fri 10/2/15	Fri 10/2/15	11
21		Manual	3.9.2 Project Plan Document	1 day	Fri 10/2/15	Fri 10/2/15	11
22		Manual	3.9.3 Reflective Journals	152 days	Fri 10/2/15	Mon 5/2/16	20,21
23		Manual	3.9.4 Requirements Specification	1 day	Fri 11/6/15	Fri 11/6/15	11
24		Manual	3.9.5 Project Analysis & Design	1 day	Fri 12/4/15	Fri 12/4/15	11
25		Manual	3.9.6 Prototype Presentation	2 days	Thu 2/4/16	Fri 2/5/16	
26		Manual	3.9.7 Beta Version	1 day	Tue 3/8/16	Tue 3/8/16	
27		**Manual**	**⊟ 4 Testing**	**12 days**	**Fri 10/2/15**	**Mon 10/19/1**	
28		Manual	4.1 User Interface Testing				
29		Manual	4.2 Code Testing/Debugging				
30		Manual	4.3 End User Testing				
31		**Manual**	**⊟ 5 Closing**	**11 days**	**Wed 5/11/16**	**Wed 5/25/16**	
32		Manual	5.1 Final Project Report	1 day	Wed 5/11/16	Wed 5/11/16	
33		Manual	5.2 Final Project Presentation	3 days	Wed 5/18/16	Fri 5/20/16	
34		Manual	5.3 Project Showcase	1 day	Wed 5/25/16	Wed 5/25/16	


September 11
November 11
January 11
March 11
May 11
8/16
9/13
10/11
11/8
12/6
1/3
1/31
2/28
3/27
4/24
5/22

Month: September

My Achievements

During the month of September I spent a lot of time deciding on what the topic of my 4th year final project would be. After much consideration I decided to go outside my comfort zone and try something new. I decided that I would create a game and I am very excited about my decision. I did a lot of research on the type of game I was going to create and after speaking with colleagues and watching online tutorials I decided to choose a first person shooter game. I decided to jump right in and familiarize myself with the Unity game platform which I would be using to implement my game. I have never used Unity before so I was eager to get started. I worked on my terrain and created a mountain scene with hills and a beach area. I feel I have made a great start on this. I worked on my project proposal which outlines all the steps I took and will take to complete my project successfully and I created a Gantt Chart to give myself an overview of the time I have estimated to reach my project goals and deadlines. I feel my project now has a good structure and is more organised. I have also completed my first reflective journal

My Reflection

Overall I am glad I put a lot of research into my project. I am happy with the genre of game I have chosen as I feel I would have run into problems if I had chosen the car racing style game. There are many more tutorials and help online for the style I have chosen. Completing the project proposal helped me outline my ideas and goals. I have become familiar with Unity so I am happy I have started using my selected program. I have started working on my terrain and I am happy I did not procrastinate in my first month back at college. I have decided I am going to work on this until I am fully satisfied with it before I move onto characters and coding. I feel prepared for the work that is to come. However, I was not successful in fully committing to the allotted hours for my project. Although I am happy with my progress I feel I could have achieved more if I worked on it for more hours at a time. I also find it very hard to concentrate with others around me as not every student works on their project.

Intended Changes

During the next month I hope to get started on my characters and weapons. I will spend some time watching tutorials and then put the knowledge I have learned into my project. I am happy with what I have done so far but I want to get more work done in the next month as I do not want to be overwhelmed with the amount of work I will have. I am going to have to find a practical way to get a good amount of my project done while trying to complete my other modules. I need to be well prepared for all my deadlines as I do not want to rush my work. This will help remain calm and help me feel more organised.

Month: October

My Achievements

During the month of October I completed the requirements specification as part of the project. I created a Use Case Diagram that shows the important steps of the game system. I also created a class diagram that shows the relationship between the classes and how they will interact with each other. I had my first meeting with my project supervisor and he went through the marking scheme with me. I have started working on a document for him which will outline the development process and the innovation in my project. He advised me to do some research on how games are developed, the architecture in games and to have a unique selling point for my project. I researched the types of requirements in developing a game. I have also completed my second reflective journal.

My Reflection

I am glad I worked on the requirements specification document as it enabled me to gain an insight into my project and define the scope of the project. I felt the meeting with my project supervisor was beneficial as he led me on the right track. He advised me to focus on the marking scheme in order to achieve a good grade. He provided valuable feedback in regards to game development and assured me that I could go to him with any problem I had. He set me some initial tasks which helped to understand the overall process. However, I was not successful in achieving all my goals this month. I have found the workload overwhelming and I have to learn to prioritize my work better. I have not yet started the tutorials that I had found on the internet.

I have all of them ready to start but I was preoccupied with completing the Requirement Specification hand-up for the class among assignments for other modules as well.

Intended Changes

Next month, I will try to work on the coding aspect of my project. I also hope to have a menu created for the game.
I realized that I need to get started on tutorials and spend even more time on the project so I don't fall behind.

Month: November

My Achievements

This month, I decided to re-start my project as I was not happy with the decision I had initially intended to go with. I am much happier and I feel I have a much clearer picture in my head.
My contributions to the projects included re-creating my terrain and looking into the creation of code. I also worked on my project analysis and design document.

My Reflection

I felt, it was a wise decision to re-start my project. I now know what I want so it will be easier and faster to do it the second time around. Once again, I have spent all my time focusing on the environment and look of the game that I have neglected the programing aspect. During the Christmas break l plan to work extensively on my project and focus on coding so that I will have functionality in my game for the mid-point presentation.

Intended Changes

Next month, I will try to keep to the plan that I have set myself for the Christmas break. I will work on the main game functions and create a game level. This level will be a good starter point for the game and hopefully it will make it easier for me to get the project finished. I also need to practice my coding in Javascript and C#.

Month: December

My Achievements

This month, I watched many tutorials and start adding code for shooting a gun to my project for the mid-point presentation

My Reflection

I felt it was difficult to balance the work for my main project and projects due for my other modules. I also had interviews for graduate programs so I had to prioritize my time accordingly.

Intended Changes

Next month, I will try to work on my project even further while also studying for the Christmas exams.

Month: January

My Achievements

This month, I finalized my environment for my game. I am very pleased with the environment as it turned out exactly how I imagined it. I added in the gun model weapon and hand model for my first person shooter to make the gun more realistic. I parented the gun and hand to the first person to make the gun follow the camera. The gun movement was still static at this stage so I added in a gun movement java script to make the gun more realistic. I also added in a cross hair for my gun. I added in animations to make the walking of my character look more natural, which included a head bob. I will continue to work on these animations to add movement like running, crouching etc.

I also completed my Technical Report and started working on my power point presentation for my mid-point presentation.

My Reflection

There was a lot of trial and error trying to get the gun model and hand model positioned correctly. I had problems with this initially, which included the character falling into the gun when I started the game but I eventually fixed this by lowering the clipping planes and the gun and hand finally displayed and worked how I wanted them. I had problems trying to get the cross hair to display exactly where I wanted it on the screen so I decided to take this out for the moment. I might return to it at the end when I am finalizing my project.

Intended Changes

Next month, I hope to add in my enemies and work on shooting and killing. I realized that I need to work harder on implementing the functional gameplay aspects of the project.

Month: February

My Achievements

This month I worked on implementing intelligent enemy AI that will use pathfinding to detect and track a player around the game. I feel this will make the game more realistic and exciting. I also got familiar with a new AI engine called RAIN which is used inside of Unity. It is a new standard in digital entertainment which is used by top studios.

I wanted to increase the complexity of my project even further by incorporating something unique. I brought an enemy into my game and set up a waypath using behaviour trees which will allow my enemy to patrol an area and detect the player if they come too close.

I added gunshot sounds and player hit sounds and also worked on my 30-40 word profile. I found it quite hard to show the complexity of my project in so little words.

My Reflection

This month I am finding it very difficult to get characters that are suitable and actually work in my game. The number of free assets in the asset store are limited which means I might have to purchase assets in order to complete my game how I would like. This could be a big problem for my game.

Ideas that I initially started working on before I came across RAIN might be irrelevant now so that is some time wasted.

A lot of research and tutorials are needed in order to familiarise myself with this new engine so hopefully it will be worth it for the final product.

I am finding it difficult to tie my background story into the game right now but I am looking for ideas that I could use. I was thinking I could use notes or clues around my game in order for the player to find the target.

Intended Changes

Overall I am happy with the progress I am making in my game but I still have a lot to do to get the finished game how I want. Next month I will focus on getting my enemies finished so I can start on the boss character.

Month: March

My Achievements

This month I added AI into my project. I knew I wanted to incorporate A Star Algorithm. I wanted to use this as A Star tries to look for a better path by using heuristic functions to optimize the search which gives priority to nodes. A Star is typically used in games as it is useful when you are searching for a path on a map where you can guess the distance to the target from a given graph node.

I used grid graphs which are useful for ease of use and fast updates. These are built by arranging nodes in a grid like pattern around the world. They allow for fast updates due to their regular structure and are useful in games that require frequent updates to the graphs. The nodes which are blue show the walkable area. I assigned objects in my game to different layers and selected an obstacles layer as my collision testing which my skeleton AI will avoid.

The pathfinding algorithm path updates as the character moves around the map. An AI Follow script and Seeker was added to the enemy to allow him to follow the character. A trigger will go off if you get to close to the enemy, which will allow him to identify the character player. The enemy will chase the player and animations will change to attack when he reaches the end of the path.

The enemy will only attack if you're invading his space. If the distance is more than 10 the enemy will stop attacking and remain in an idle animation. I also added in code so the enemy always looks at target. This prevents the character moving in behind the enemy.

I also added in an empty game object called Sensor and added a SensorAI script with a box collider. This Sensor has a smell sensor which prevents the player sneaking up on the enemy. The enemy has a raycast field of view of 68f. When the player enters this sensor, the enemy will come after the player. I also wrote up the 200 word profile for the project showcase.

My Reflection

This month I am really happy with the pathfinding I implemented into my project. I got a suitable enemy for my game with animations and came up with a better background story to tie everything together.

Intended Changes

This month will be hard to finish the project as I have exams so I will have to get it finished whenever I get free time. I will finish up the enemies to start on the main boss character.