IMAT3451\_1819\_520 Final Year Project

Student name: Ming-Hei Lui (Felix)

Student number: P15245731

Supervisor: Conor Fahy

Date: 05/04/2019

Word count: 8007

# Abstract

This project will talk about Unity and Unreal. It shows which game engine is suitable for the beginner. It will definition for RPG game and explain differences between 2D, 2.5D, 3D game. In this project use the Unity to create the 2D RPG game. The game has the requirement to make the game. It will show this project of the game element. It will have the testing part of the final game.

# Review of literature

## Unity VS Unreal

Unity and Unreal are cross-platform game engines like Android, Windows and PlayStation. Those game engine can create the games for a different platform. This two-game engine have their advantage for different facets such as graphics, coding and price.

Unity and Unreal have different graphics display. Base on Sunday Sundae (2018). The Unreal is more suitable for highly graphical and realism. Also, it has better material editor to make the lighting more accurate and smooth compared to Unity. However, the editor to make good textures or mapping it more complex. Unity has the material editor it limited for adjustment of textures and mapping. It can see in the Unity can make a simple game with textures and the good of the beginner. Because of it easy to use.

The big difference between Unity and Unreal is the main programming language, Unity are C# and JavaScript to build the game. But Unreal has used C++ to program the game physic. C# is the high-level program league. C++ will need the time to learn and code in the game without getting any error.

Unity and Unreal have different charging method to keep the company running. Although Unreal is free to use and create games in playing the game, Unreal will charge the developer like buying the tools of the game or microtransaction, based on Tim, S. (2014) “Anyone can ship a commercial product with UE4 by paying 5% of gross revenue resulting from sales to users.”. If the game being popular and make the income, it will take 5% income of the product.

Unity has three options of the price, Personal, Plus and, Pro. It for the beginner, hobbyists and team. It is free for the beginner to create and learn the game. Unity did not have to pay when the game gets any income.

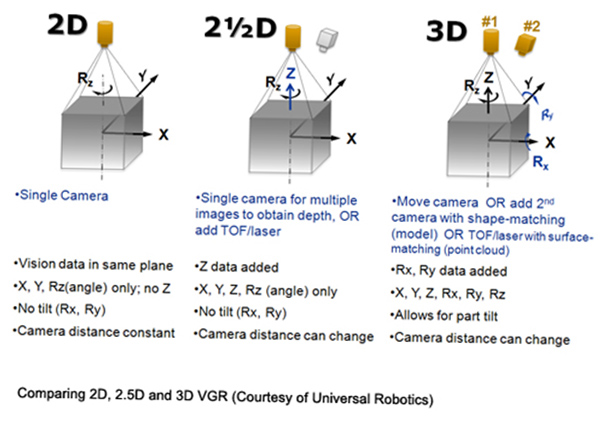
The Unity is more suitable for the 2Dgame or in the Android platform, its uses C# can easy to create the game of the beginner. Unreal engine has better graphics and building in the 3D content. But it uses C++ to program the code it very takes time to learn for the beginner.

## 2.5 games and 3D games

2.5D and 3D can use the 2D or 3D platform to design a game. According to Unity3D (N/A) “Some 2D games use 3D geometry for the environment and characters but restrict the gameplay to 2 dimensions”. Also, there are many ways to show 2.5D games, one of them is orthographic 3D, using the birds-eye view to the play like overcooked and Warcraft III: Reign of Chaos. The orthographic 3D games always use in resource allocation and strategic. The other one is using the 2D gameplay with the 3D graphics. It makes the play in 3D mode but 2D gameplay like Crash Bandicoot and LIMBO. Also, both games are 3D modelling for the character. 2.5D is using the parallax mapping to look like a 3D model. 2.5D games are between 2D and 3D. 2.5D can use inside scroller game and Top-down perspective to show the more 3D model detail in the game and make the characters, map and item more realism and detail. The presentation of shadow, charter move, lighting…. etc. is more real than the 2D games.

In the early days of designing game, the computer did not have the configuration to run the 3D games in the PC platform. The game was making pseudo-3D in the game to project multiple facets of the object. According to Russell, K. (2018) “Around the late '90s and early 2000s, all of the most treasured video game franchises has transitioned over into 3D”. it shows the PC platform and the 3D technology more and more mature. The 3D game uses more resource to run, the platform needs to high efficacy to run. Because 3D game needs to run three coordinates x, y, z of the basic data, it can make the three dimensions of height width and depth in the game.

The big difference between 2.5D games and 3D games. According to tabletgames.tech/ (2018), the 2.5D game camera only moves x and y position to see the main character. Also, the game object will not tilt. The 3D game camera is using three coordinates to manufacture the stereoscopy of the player. However, the 2.5D game can make by 3D model or fake 3D perspective with 2D graphics.



## 2D

The 2D platform game is using 2 dimensions’ gameplay, there has a lot of 2D gameplay like the side-scrolling game and top-down view. In the ’90s has many games using the side-scrolling game and top-down view to playing. Because the platform has restricted, it cannot make the 3D game at that time. However, the game developer still produced famous game in their known technology, for examples: Grand Theft Auto (GTA) and Metal Gear 2: Solid Snake. Now-a-day these games have improved in many different versions and stories. According to Mahinder, V. (2019), the animation always uses 2D and 3D animation. The 2D game of the development is good for the beginner to create the game. Also, 2D game development is easy to develop because 2D need less technology than 3D. Moreover, the 2D animation will take a short time to create. It can create the 2D game in a short time and low technique.

## RPG

RPG is Role-playing game. The video game RPGs is the player will set the roles of characters in the virtual world. The player will control one main character or more to do some quests. In the game world, the player can role the character to save the world or have fun this game world. Based on Techopedia (N/A), the RPG game need to have a challenging enemy or map. The element of the RPG have the levels system, fight with the enemy and have the storyline of the quest. The RPG has many forms to play such as Action/RPG, online/RPG and Adventure/RPG…etc. The RPG game has an interesting story of the player. It can easy to make the player focus on the story and the game. The game world will have the rule needs the player follows. The player has to explore the game world and do the quests to get some reward. It can more easy to overcome the big boss.

The player character may need the loot like such as weapon, armour and potion. The player needs to upgrade those loot or get the better item to be strong to the enemy. Also, the player has the level system to upgrade the basis value to fight easily.

### Benefit of RPG

RPG game can give the player many benefits, it can make the player thinks about how to overcome this level or enemy. Base on Patrick, A. (2015), RPG game can increase the player problem-solving skill because the game world has the rule, problem and maze. The player needs to solve the problem get to the next level. It makes the player think more the problem. If the player playing the RPG chess game. The player will see problems from multiple perspectives.

On the other head, RPG game can improve social skill. The player needs to communicate with these team or other player to get the information. It can make more friends or discuss the game. Also, training communication and social skill in the game.

## UI design

In the UI design, different art style design can combine to the games and make the player more immersion in the games. Also, the UI style is important to the game such as Dishonored about the Science fiction it is using like rocking UI and Diablo 2 have the ancient UI in the game. The UI can make the player to see the same style in the game, determine game era. The game art style and the UI design need consistency in the game. According to Medium (2017) “The User Interface is the space where interactions between your users and the machine occur.” The good user interface can communicate to the user and the game. The user interface needs to look good because the game will have a lot of time to show the UI and the game information to the player. In different UI design, it gives the different experience for the player.

In addition, a good UI design can let the player easy to get the information. Also, the UI does not have too many spaces to display inventory. Base on Margaree, P. (2015), the UI designer put the UI in a different place can affect the player’s experience and commitment to the character. In the paper show, the HUD (head-up display) can make the player enjoyment and effective to see the information about the game.

## Level design

Level design is one of the main conditions to make the game more challenging and interesting. According to Gillian, S. (2018) “Levels are the space where a player explores the rules and mechanics of a game”. The level can use the experience or skill to overcome the problem, it can build up the player’s problem-solving skills. The game level can interact with the player or character.

The Good level design can let the player know the story and remember deeply about the story or the level scenes. The scenes have the suitable for arrangement can make the player be absorbed in the game. According to Dan, T. (2013), the level design has a clear vision for the player to follow the main path to play the game. A lot of RPG game the player is free to move and explore the game world. It is easy to lose the purpose of the game and the player not knowing where to go next. So, the clear vision for the player is to follow the path is necessary. Moreover, the level design is the guide to the player how to play the game. Also, the game story can be explained in the level design and different scenes. Each level can upgrade the character or give new ability to keep the player playing the game. The player can feel the character stronger to solve some problems step by step in the level like trap and strong enemies. Furthermore, the new ability can increase the player how to play the game and create the playing style in the game. It can add new things or some rewards to encourage the player to keep on playing the game and more understand about the game. The player will more enjoy the game from the good level design.

## Target age

The target age is important for games because different ages will have different understandings and problem-solving skill. The content of the game has ratings for target age such as ESRB (Entertainment Software Rating Board) and PEGI (Pan European Game Information). According to ESRB (N/A), “Content Descriptors indicate content that may have triggered a particular rating and/or may be of interest or concern”. It shows the content depends on the rating board to set the content such as Animated Blood and fantasy violence.

The average age of gamers is youth, In order to have more people play and know about the games, the target age set to 20-30 years. According to Liza, B. (2018), the highest of games players by age between 21-35 have 35%. The second has 28% of game players between 36-50 years ago. The third is the 10-20 years teenagers have 22% is the games players. It can see 21-50 age of gamer over half of the age has 63%. Hence, this age of the player can afford to buy games or microtransaction (income job). Moreover, this group of gamers are playing video games more than teenagers.

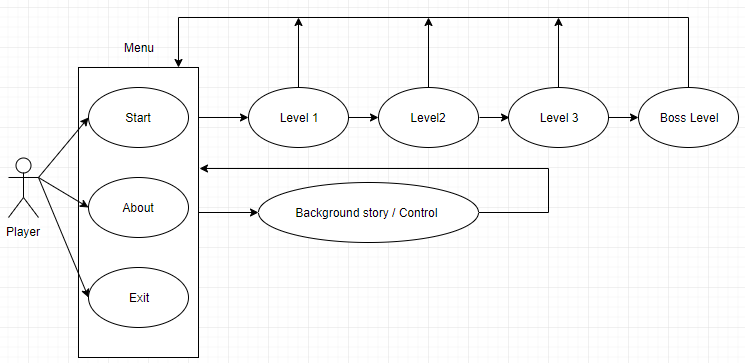
# System requirements

1. Game must make in the Unity
2. Game is the 2D creation RPG in Unity.
3. Game is playing in PC platform.
4. Game is suitable to kids and adults.
5. Game have four maps to play.
6. Game have the trap and different type of enemy.
7. Game has UI design.
8. Game has NPC show the information.
9. Game have the trophy system.
10. Game have the Coin system.
11. Game have five audios for sound effect or background music.
12. Game will save the data about game detail, item and skill.
13. The player needs to step by step to go through each level.
14. The player can move the main characters.
15. The player can use the weapon to shoot to enemies.
16. The player has the health bar and Level bar.
17. The player can buy some item in the game.
18. The player can upgrade and unlock the skills.
19. The player can add the element in the bullet.
20. The player can change the weapon.
21. Give the story about the game.
22. Each level has the exit the door to move to other levels.
23. The enemy have the AI to do some addition.

# Use requirements

1. The player can use the key board to control the characters.
2. The player can use the mouse the aim and shoot to the enemies.
3. The player can use key skill to cast.
4. The player can collect the coin and trophy.
5. The player can buy potion in the shop area.
6. UI design to illustrate the characters detail.
7. Menu design have option for the player.

This is the case diagram about the game flow. It shows all level and the menu for the game relationship. Also, can see the option for the player to choose.



(Use case diagram)

# Implementation

## Introduction

This project is using Unity to create an RPG game in the PC platform. The game is about the tank to save the world. The name of the game is “Tank hero”. The player can control the tank to kill the enemy tank. The enemy will find the player and attack. The player needs to go through all the level. In this project, using Unity to create the RPG game. The game coding part is using C# and some Unity online resources to create and build the game system. In addition, Unity can make a simple physical in the game. The project chooses Unity, because Unity is the game engine and it is easy for the beginner to learn and can create the game in a short time. In contrast to the C++, SFML and android studio, Unity can simulations multiple platforms. This project is Use unity 5.4.0f3 (64-bit) version to run.

Unity is a popular engine of the developer in the world. Unity is the easy engine to create games and using C# and Java to coding the game is much easier than other programming languages. In the other game engine like unreal it using the C++. It will take a long time to create the game. In the world there have many game engines can create the games, for example, Unreal, Unity and Game Maker Studio. I project chooses Unity because I have the experience of using Unity to make game, it give some base idea about Unity. I used the Unity to make a game in the first year with a group project. It can easy to get started to create the game by myself. This project has spent 5 months to design the game and create a game product.

There have many excellent 2D games in the ’90s and ’20s. The point to excitation for this project is the PC platform is mature and I played those excellent 2D game when I was a teenager. So, it makes me create the RPG game in the PC platform. Nowadays, the player can easy to get in the PC platform game to play games. The PC game is still taking the part of the video games. This RPG game project is worth to do. I wants to make the game like the legend of Zelda and Diablo. The project can study by myself, it shows what I learned in this whole course. Furthermore, this project allows me to learn how to design a good game and create the game physically. The project can increase the knowledge about C# coding and use Unity. It shows the work about this final year work in this project.

Game idea

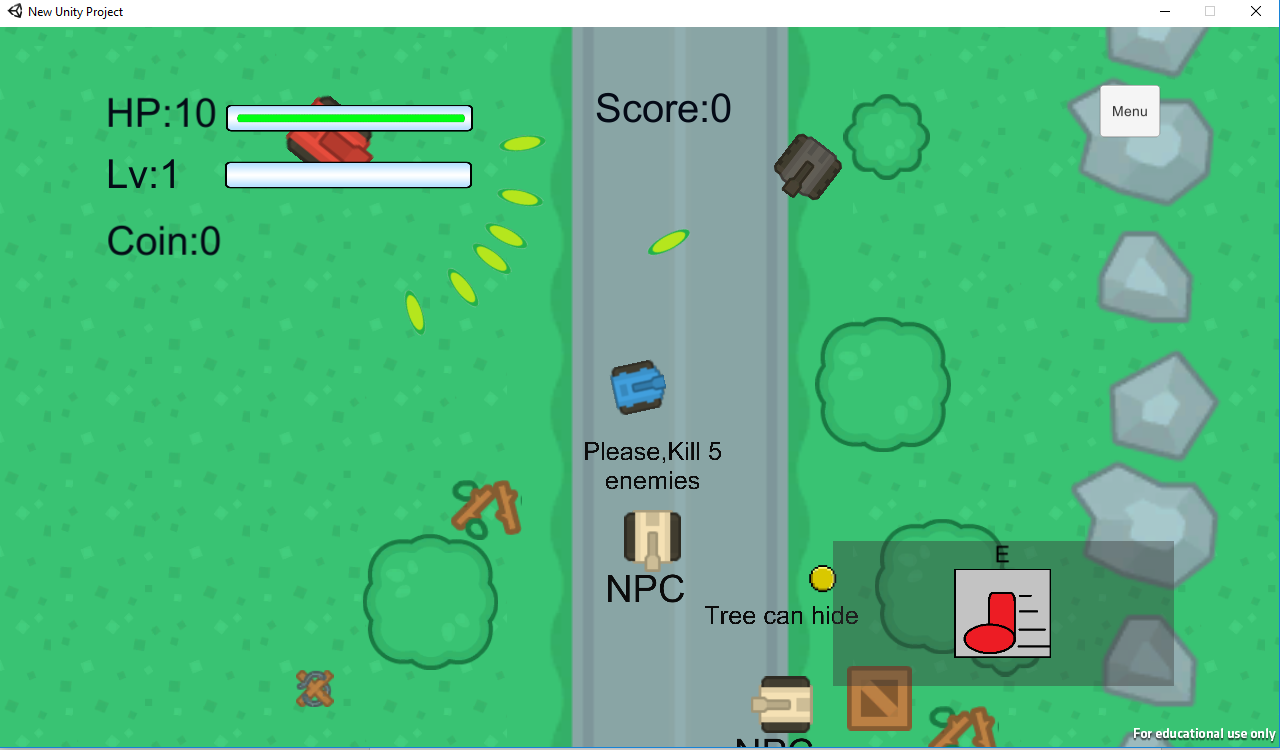
This game idea comes from the binding of Isaac and the legend of Zelda. Those game are the 2D games and has the same control. The “Tank hero” has made reference to those game, the fighting system and the level design. The game uses the mouse to aim, this idea wants the player can shoot all the angle.

Gameplay

The game is in the 2D platform. The player uses the keyboard WASD to move, QER to use the skill and use the mouse to aim the enemy and left click to shoot the bullet. The game is of 16:9 display to play, it can show the correct UI of the game. In the “Tank hero” game, the player is the blue tank need to kill the enemies and go through the maze. Which is full of different traps and enemies. Also, the map has the trophy need the player to find and collate them. This game has three levels and one Boss level. The Boss tank is stronger than the other normal enemy. Furthermore, on the map, it has the NPC to give information about the game and communicate with the player.

The player will need the shooting skill and the ability to overcome the enemy. Moreover, the player needs to follow the road and look for the exit level or something else. The coin, trophy and shop will show on the map. The player finds the way out needs the kill a certain amount of enemies. The player has the skill to dodge or the fast attacking to the enemies. In the game, the player can use the item in the map to fight with the enemy. The map has the trap, road and tree to give benefit and advantage to the player.

When the player goes to the other level, the player data will be saved. It saves five items the player HP, Level, coin, skill and score. In RPG game those data use into another level, the player can continue to play. When the player health=0 it will game over, it shows two option to the player one is the play again the other is going back to the main menu.



Main body

First, the player will play the role of the blue tank and control tank. The player tank can move, shoot and using the skill to kill the enemy tank. It gives three skill to the player. At first, the player only has one skill. The player needs to kill the enemy to get the experience and level up the blue tank. Level up the player tank can get the skill point to unlock the other two skills. The player tank is in the middle of the camera. Moreover, the camera will follow the player tank.



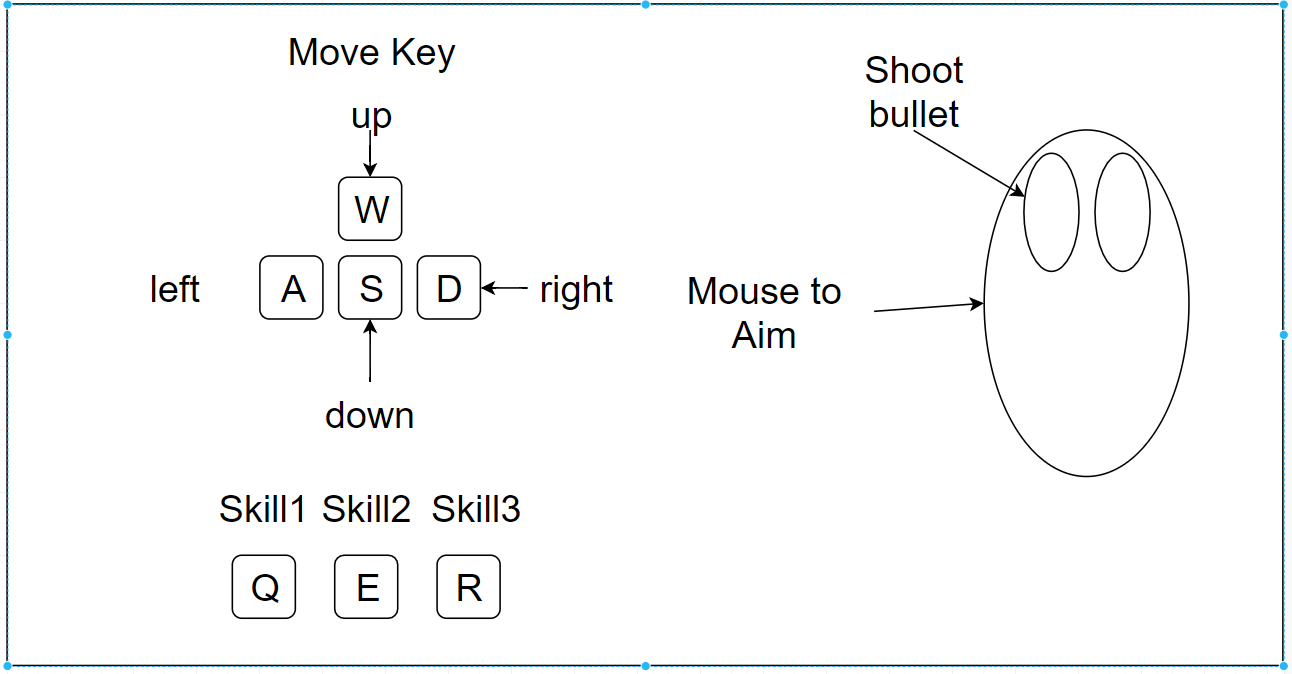
(The player tank)

In game art, the art sets the style and keeps the player to play. The game art most use the free online resource, this game is using the Kenney.nl for this game. In Kenney have the many cute arts for many different types of the game. Also, the art is free to use, it chooses Top-down Tanks Redux is the suitable art for this game idea. This game uses a lot of Kenney Top-down Tanks Redux of the tank, background and explosion…etc.

## Control

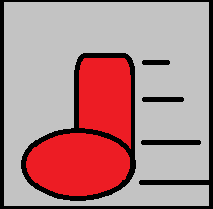
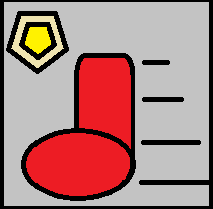
In the player control part, this game is in the PC platform will use keyboard key and mouse to control the main character. The character can use WASD to move and mouse left click to shoot the bullet.

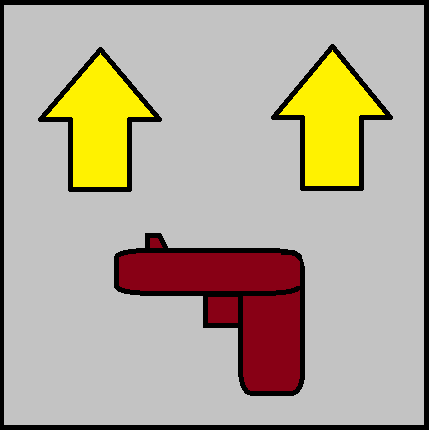
The player tank will follow the mouse cursor to aim because the mouse is shooting front sight for the player tank. Also, the cursor was designed the grey crosshair to the default mouse cursor. It can make the player shoot more precise and realistic. Furthermore, if the grey crosshair detects the enemy, it will call the function OnMouseEnter(), it will change the grey crosshair to red crosshair and can much easy to aim the enemy object.

The Skill button use Q, E and R to activate the skill. Set Q, E and R Key to skill because it near the WASD key, it is more convenience for the player to use the skill in the game. The skill key can activate different skill in the game like move and shoot faster. 

## Skill

The skill system has the Ability base class to create the skills. The ability1, 2, 3 are inheritance Ability base class. The three skill buttons have the cold down timer and lock the skill. Each skill button has the 3 seconds of cold down. Also, the skill has unlimited use. The player needs to level up to get the skill point. In the game menu, it shows the skill button to open the skill panel and use the skill point to unlock skill. If it is unlocked, the skill will show the image in the Interface. The Skill can activate when a mouse clicks the skill image button or skill key to use the skill. The skill image is the paint software creates. Furthermore, the image was drawn what the skill is used.



(Skill Q) (Skill E) (Skill R)

For the first time to design unlock the skill system, it was to use the skill point item to upgrade the skill point. Skill point item will somewhere in the map, the player needs to find it and get more skill to fight with the enemies. But in the end, this idea was changed to the player level up and get the skill point, because of the level up and get the new skill will more like RPG game.

Skill E key

In the game, it has three skill for the player use. First skill gives to the player to use, the skill key is E. If the player pressing the E key will activate the skill to move faster, this skill can easy to dodge the enemy bullets. The speed was faster than the player on the road.

Skill Q key

The skill Q is the shooting speed up, unlock this key gives the player’s shooting speed up in 3 seconds. With this kill, the player can shoot a lot of bullet to the enemies, therefore he can do many damages and easy to kill the enemies. Designed this skill because in the game there are many enemies and the player’s shooting speed cannot kill the enemies in a very short time.

Skill R key

The skill R can create the yellow shield in 5 seconds the shield can withstand the enemies’ bullet. At the same time, the player can shoot through the shield and do the damage to the enemies.

## Enemy

In the game, Enemy was divided into three different types the first type is the black tank enemy. The second is the red tank and the last one is the Boss enemy. Each enemy has their special shooting style, it makes the game more varieties and the player will get more challenging in playing. All enemy tank will shoot the bullet continuously. The black enemy tank is the simple tank; it only shoots one bullet to the player. Next is the red enemy tank, shooting style like a short gun, it has six fire point to shoot the bullet at the same time in different directions. In order not to let the Boss enemy tank to easily get kill and have more difficulty in level up, the Boss enemy tank has been designed to have more HP, bigger and four fire point shooting to the same direction. Also, it has the attack pattern: shoot four bullets with two middle turrets follow the shooting.

(Black tank) (Red tank) (Boss tank)

The explosion is the animation for the enemy, when the enemy health = 0, the enemy die position will show the explosion animation. The animation can be improved the player’s experience. Unity can make five-explosion image to work for one animation for the game object. Each image of the animation timeline takes 0:07 then move on to other images. This animation will loop 1 second.



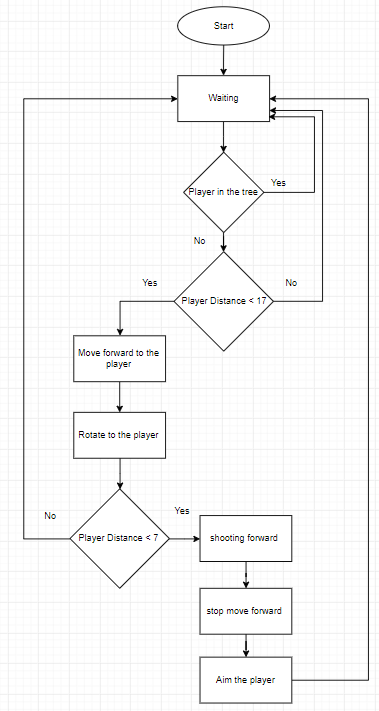
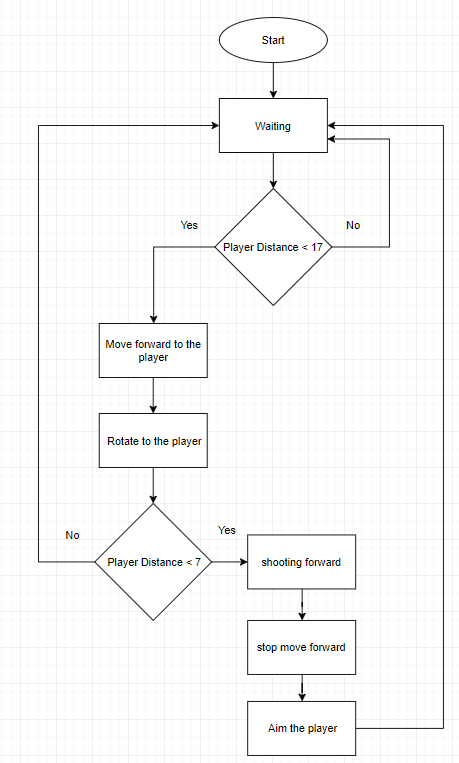
(Explosion image)

## AI

The enemy was designed by AI, the tank will focus on the player tank and attack. Also, the enemy tank will detect the player tank’s position and rotate the fire point face to the player tank. The AI sets the fire range for the enemy tank. If the player tank is in the fire range, it will keep aim the player to shoot and stop move the position. The AI design needs to step by step to test and develop a system to make the enemy more interact with the player. Also, by adding more functions to enable the enemy more flexible to knock down the player, it will increase the difficulty of the game and make more challenging when the player playing it.

The first decision is to check the tree because the player tank will hide in the tree. The AI will make the enemy look like not seeing the player tank and keep detects the player tank. The second decision is designed the type of approaching because the enemy needs to find the position of the player then move and rotation. If the enemy ready to know the player’s position, it will like a zombie follow the player to move. After detected and pinpointed the player, it will keep facing and moving forward to the player. The third decision is to set the distance range, how close to the player to do the shooting bullet, stop the move and keep aim the player. Because if it does not set the shooting distance, it will keep move to the player tank’s position and will be discovered by the player tank. Also, the enemy tank is using the raged weapon shooting the bullet to the player, it set the distance range for shooting will have a firework frame and makes the game more exciting.

On the other hand, the boss tank having the same AI in the game. But it does not have the first decision for the player in the tree, because in the game the boss tank was designed that he can see and keep shooting to the player. The player will get the same experience of the player.

(Enemy AI diagram) (Boss AI diagram)

## NPC

In the game, NPC is set to the brown tank. Each level has some NPC to show some text to the player when the player coming close to the NPC. Also, the NPC do not attack the player and move, it only shows some information to the player. NPC in a way to have to interact with the player. The player will know more about the game, the NPC is an essential part of the RPG game world.

C:\Users\P15245731\AppData\Local\Microsoft\Windows\INetCache\Content.Word\tank_sand.png

(NPC tank)

(NPC talk to the player)

## Level block

The level needs to kill a certain amount of the enemies to unlock the fence. The fence is blocked the next level door. The player cannot go through the fence until killing a certain amount of enemies. The fence has one NPC to tell the player how many enemies need to kill to unlock the fence. Make this decision because it can force the player to kill the enemies. The player cannot rush the level to finish the game.



## Audio

In the Audio part, there have many resources of the sound effects and background music in the online. Unity can support the common formats such as .mp3 and .wav, they can easily add in the Unity. In the game, it has four sound effects and one background music using the .wav formats. Audio gives the game more fun and vibrant. The background music of this game is making by beepbox.co. This website is an online tool that can easy to create the original songs of the user. Also, on the website, there are many other users uploaded their original songs. With the variety of choices, this game background music and sound effects are created by those webs. When the game starts, the background music will run and keep the loop in the game. Until the game ends quit/end/finished.

Besides the above, other sound effects are created by the online website tool called bfxr.net. It is like beepbox.co because those websites are for the game audio. But bfxr.net is focused on the sound effects for the computer games. The sound effect gives the player a more real-life in the game world. This website got Coin, shoot and jump sound effects. Also, can synth the sound to make multiple changes for sound effects. It does the sound effects for the enemy and player shooting, the player gets hit and pick up the coin in the game. It adds those sound effects in the Unity and in the coding make the trigger to run those audios.

## Game menu

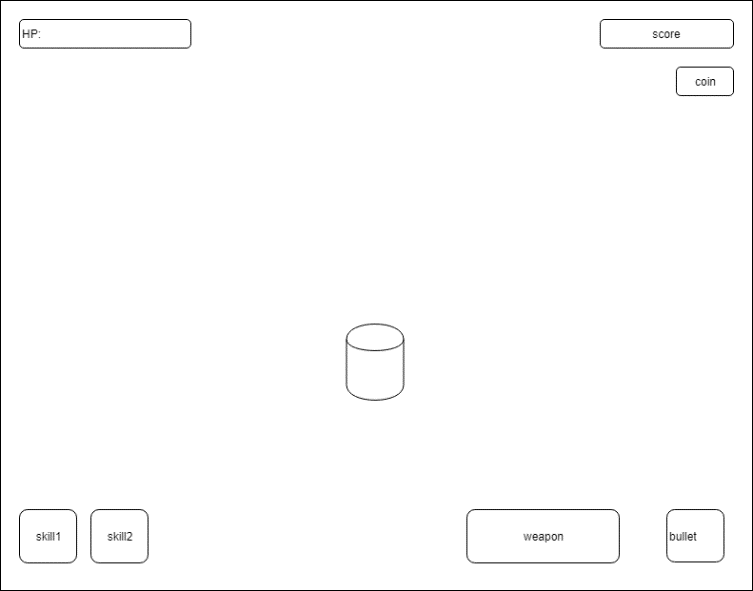
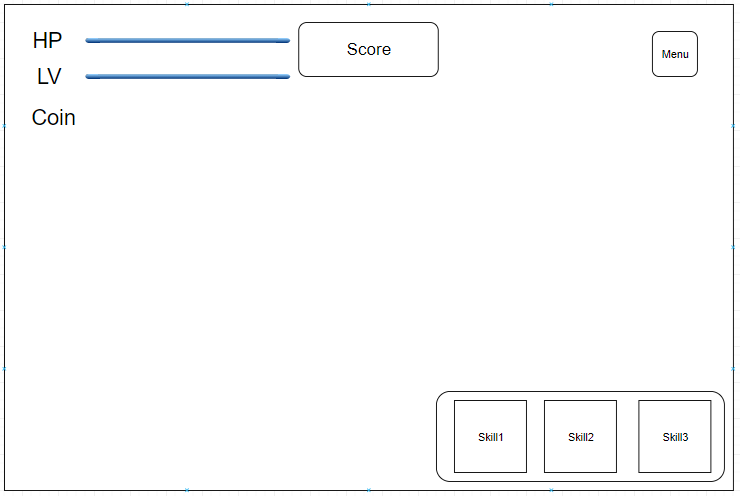
The menu has three option for the player. It has Start, Help and Exit button options. The first one is the Start, press it enters the game and player. The game has three level and one boss level for the player to play. In each level right head side has the menu button. It can go back to the main menu.

The second is the Help button, it goes to the help scene. The scene shows the game story and the instruction about how to control the character. Also, the scene has the back button back to the main menu.

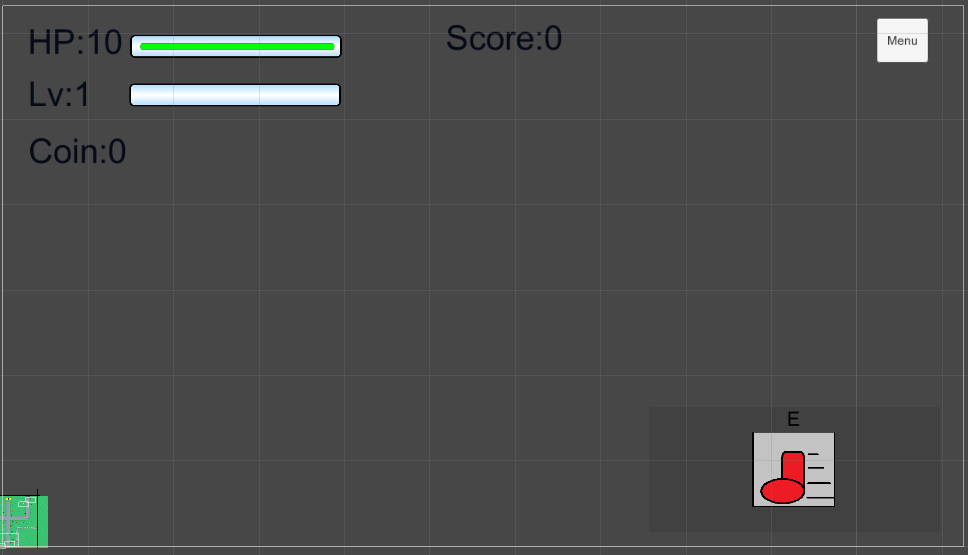
The last one is the Exit button, it can close the game. The scene was designed that only in the main menu has the button to execute close the window.

## UI design

The first UI design for the game. It gives the game a basic idea about the game and needs to think like the HP, score, coin and skill…etc. In the final game, the UI designed to have some changes in the game. It can see the difference between the draft and the final design. It adds the Lv bar, removes the UI not need to use and some UI position can be changed so that the player easily to see. The good UI design needs to show information more clearly. The player will easy to get that information and keep concentrate in the game.



(clear final UI design) (first UI design draft)



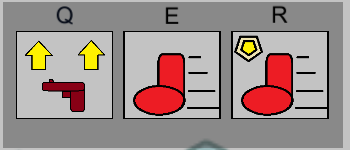
(Final UI in the game)

1. Menu

In the menu button, it has the three buttons of the player to click. First one is a skill tree, the skill tree panel is on the left. It shows the player skill point and the other two skills. This panel can unlock skills. The second button is the Trophy button is show and save the trophy when the player gets the trophies in the game. The last button is quit, this button will the go back to the game menu scene. This three panel can display at the same time. Click one more time the menu button will continue the game.

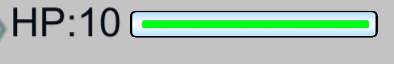
1. Skill

The skill panel have three skill inside, each skill has the key top of the skill to show the player what key can use. Furthermore, when unlock the skill it will shot in this panel and the player can use the mouse to click the skill button to switch on the skills.



1. HP

The UI is showing the player health bar and the number to the player. If the player health is low than 3, the health bar will change green to red. It can let the player know the health is low and have the danger of attack by the enemies.





1. Level bar

The Level bar shows the player level in the game, the player kills the enemy will increase the blue bar. If kill a certain amount of the enemy blue bar will full, the player will level up to get heal and skill point.



1. Score

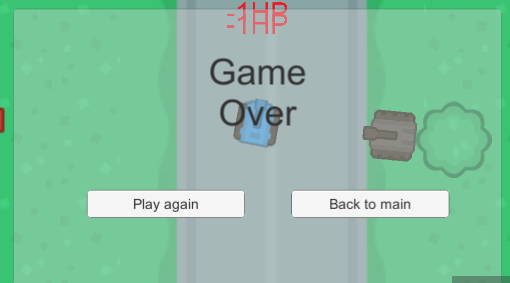
The score is in the middle of the top. The score is the UI text to display the player how to get the score from the killing enemies.

1. Coin

The coin is in the left side and down of the level text. The coin is the UI text to show how many coins did the player got.

1. Game over

The player health = 0, it will display the Game over panel. The panel shows two buttons for the player. One button is play again, it will back to the level the player dead and two is to the go back to the main menu. The player needs to choose one option to continue the game.



## Level design

In the game have 3 level and 1 boss level, each level did not have not the same level design. The first three level like the maze, the player must find the way out and killing enough enemies to past the level. In addition, those level have some particular item to help and challenge the player to play. The game has the NPC to give the clear mission to do and move on to the next level. Moreover, the player can explore the map to find the trophy or the shop to buy some item. In the map design in the appendices, it show each maps of the game. It can see the shop, enemy and NPC position…etc.

Designed to create the boss level because it needs to test the players know about the control and can the player fluently to control the characters to kill the enemies.

In the level design, the item creates the game object to set the game rules and challenge for the player. In the level can see the road, tree, and potion…etc. In the game has collation system for the player to use. Collation system can make the player look around the map. In the appendices will show the map design of the game.

Coin

In the game, the player can see the coin and get it. Also, the coin system can cooperate with the shop, the coin is easy to see and hide in the map.

When designing the game, resource management is one of the main RPG game elements. The player can control the resource once he gets into gets in the game. The coin system in this project store up to another scene, it means another scene will have the shop of the player. In RPG game



Shop

In the game will see the shop, the shop is a sale the item to healing the characters. The item is the potion, it can heal some player health in the game. The shop will take some coins from the player and let the player have the potion. Design the shop it makes the coin usable in the game. The player can decide to buy the item if necessary.

Potion

The potion in the game can heal the player health. The potion is the only item the player can buy in the shop. The potion was designed to need 100 coins to buy the potion. But in the final game the player a lot of coins and nowhere to spend. The potion will increase to price to 500 and heal more of the player, it will heal 1 player health to up 3.

****

Trophy

The trophy has the collation system, it has three trophies in the game and the player need to find it out. The player gets the trophies will save in the data and show in the Trophy button. The trophy was hiding somewhere in the first three levels. The player did not find the trophy would not effect of the levels or the enemy.

Road

The Road is one of the special designs in the game. The player on the road can enhance the moving speed. It creates good fighting spaces of the player, it can speed faster to dodge the enemy bullets. This design because in the real world the tank moves faster on the road.

Tree

The tree is another special design in the game. The tree can hide the player to avoid the enemy, the enemy cannot detect the player. The player can shoot inside the tree and kill the enemy. This design can cover up the player and kill the enemy because there are too many enemies need to kill and will easily get around up by the enemies. The tree will not be born on the road.

Box and Trap

The trap was can to the damage to the player and enemies. One special design of the trap is the player shooting cannot do the damage or destroy the trap, but the enemies can shoot through the trap. If the trap collided the player tank or enemy, it will destroy the trap and do the damage of character. Having this trap setting because it wants the player to avoid the trap and lure the enemies to collide the trap to take damage or destroy the trap. The Box can block the enemy bullet but the player cannot go through the box.

barricadeWood **C:\Users\P15245731\AppData\Local\Microsoft\Windows\INetCache\Content.Word\crateWood.png**

## Save data

Saving is an important part of the RPG game. It records the processes of the player in the game and marks down the record of achievements gained by the player. Unity has many ways to save the player data, RPG games have many data need to be saved in the games such as player health, coin, source and skills. As the Unity scene cannot share the data to another scene, Unity usually uses JSON and PlayerPrefs method to save the data into Unity. So, in this project, it chooses the PlayerPrefs to save the player data in Unity. Using PlayerPrefs in the coding because it is much easier to use than JSON method and it will save data such as the player Health, Source and skill. The game will continue to run while the PlayerPrefs saves the date for the game. When playing the game, the player goes to the next level, it will save five things the health, source, player Level, coin and skill, all the data will use in the other level. If the player quits and restates the game, those data will be set to the default values.

In the skills saving system, it has the SaveSkilled script to set the skill are active. It is using the array list to put the skills in the PlayerPrefs. Each skill needs to have those script to active the skill button.

## System testing

In the testing, the game project is use the Unity to find the error. The Unity have the debug tool to debug. Moreover, in the coding can use print function to find out the error or the debug. It test the base function such as user input, player tank and enemy test. The system test plan is in the appendices. It can see more the detail about the game testing.

The test part, it find the some Bugs In the executable file. After the test try to fix the problem, it need a lot of time. In the end of the fix the game, it still has few bug need more time the find out the solution.

## Conclusions

### Critical evaluation

In the first time playing the game demo, I want to make a mini-map into the game but in the final game did not make it in the UI. Because I did not have enough time to make it without any error. Therefore, it makes the map smaller for the player do not need the mini-map.

In the project, most of the requirements are completed, only system requirements 19 and 20 did not have the time to create it in the product.

In the project and the requirements, the game did not have the element of the bullet such as the fire and cold. Because it does not have enough time to implement the element system. Element system will need more script to change the enemy speed, health and the player shoot system.

Another system did not add in the game is the player can change the weapon and choosing the weapon. The player can only change weapon the normal gun and shotgun in the UI. This idea did not do it in the final game. Adding the change weapon system need more time to do the creation. It needs to change the player shooting and UI script.

If I have more time for this project, I can build more systems in the game and make the game more playful. I will add the element of the player bullets, it can shoot the fire or ice image bullet and the enemy will have the combust and cold damage. Moreover, the player tank can hold two different weapons and change it anytime. It can response distinct situation in the game. In the game still have many the things need to improve each as coding, more level and add more skill. Those improve things can make the game more fun and playful. Those requirements will do in future development.

In the time management, in Christmas and after the New Year time has two the project for this cause need to work on. It the origin plan need to delays because the other project is taking more time than imagined. The origin plan has to change, the take more time to work on the project in March. It makes the new plan still can make the game and the report can finish on the time.

### Challenge and learn

In the game, there have the good thing about the project. They are very profitable, I learn many pieces of knowledge in using Unity engine, C# coding and how to design to a good RPG game. Unity is the easy game engine for the beginner, it can create the demo game in a short time. In addition, the C# high-level programming language is easy to use and understands to code. I know more about design a good and the coding in the game physic, a good RPG game needs many the elements of the game to ground the rules playful of the game. In the project, it was added some elements like the collation system and coin. In the Unity has the online resource of the beginner like health bar is from Unity. In the start of the project, it have to take the time learn the using Unity.

The level design is the part of the design that taking a long time. In the game, each level needs to decide the map and the process on the player to get the item and guide the player to the right way to the next level. The flow of the level, the player needs the kill a certain amount of enemies to go to the next level.

In the level design, it needs the set position of the enemy, coin and the saving point. It is an important part of the level design. It sets how often the player will fight with the enemy. It is also the fun part of the game development.

The one difficulty is the game art because in the online resource of the game art it did not match the original game idea. But luckily find the Kenney.nl have the Top-down art for my game. I create some art in my game like the skills image and the potion. It is difficult to find the right skill image of the game in the online resource.

The other difficulties are how to create the level the player like. The level design is taking a long time to create. Because the good level design the player need to think about how to play the game or overcome some differently.

For now, Unity and C# skill are adept for me. I take any time to learn and create the game. I can create the other game in the Unity or C# very quickly to build the base system or game physic. The good game need time and people to create it, the game has many things included such as the game physic, audio, fix bug and game design.

### Bugs and problem

In the final game, it still has four bugs need to fix. It needs more time to fix the bugs. The first is the trap bugs, it did not destroy the object by the enemy collide but it still hit the enemy. This problem in the Unity engine is working fine, the enemy can destroy the trap. The problem needs more time to fix and find out which part broken the code.

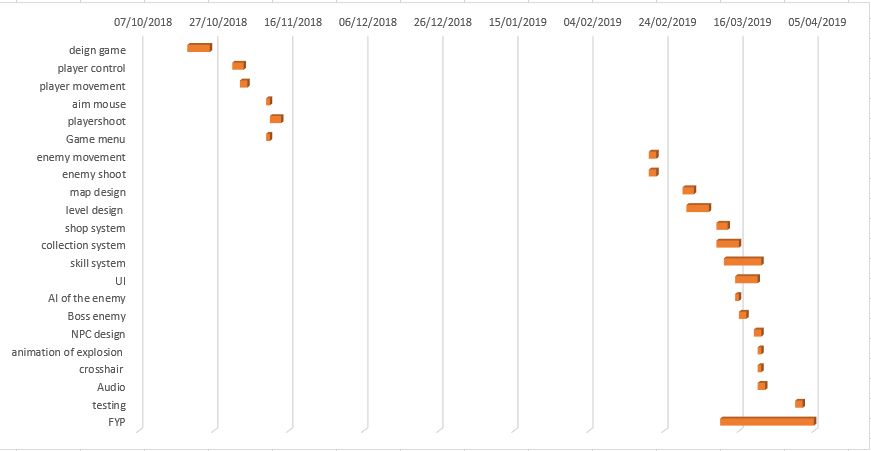
The second is the coin, it sometimes broke the game when the player gets the coin. This problem is the player tank take the coin and destroy the coin audio, it makes the other coin cannot use the audio to make the player cannot get the coin. In the problem, it tries two way to solves that out but it still not solve the problem.

Next is the cursor bug, the bug is the mouse detect on the enemy it changes to the red cursor. But sometimes it cannot change back to the grey cursor. The bug needs detect on other enemies will fix the problem.

The last is the tree, the player hiding in the tree but the enemy still can see the player and attack. This problem is tree box collides close to the road box collides because the player only enters one box collides. If the player enter collides at the same time, it makes the tree box collides not active. The enemy will shoot when the player inside the tree. Those bugs need to take more time to fix, it will do it in future development.

# Project Management

## Gantt chart



## Project details

The project did not following the origin plan to go, but it still finish the report and the game can play. The project almost finish the requirements. In the December and January were working in the other project. It did not have the time to implement to requirements. The project need to plan again to make the project can submit on the time. In March need to do requirements and finish the report. This is the new Gantt chart, I follow this chat to finish the project.

In the software development is use the waterfall method to set the project plan. The project did not completely follow this method, because it has many things in the Verification part need to redesign. The waterfall method has five points of software development.

1. Requirements are the starting point of the project, it set the project title and aim. The Requirement can set the clear aim and specific the software for the developer to create the project. Also, it can have a Gantt chart for time management.
2. The design is to decide the programming language and software to create the system. Also, it needs to think about the design which system the project needs.
3. Implementation includes the codling and build the project. In this part, the developer need the follow the requirements to create the system.
4. Verification is part of testing the system. It testing the system functional and find the error or bugs in the project. The error and bugs need to be fixed.
5. Maintenance is the whole system of the project, it needs to keep system functional and keep to do the future development or support.



Douglas,H. (2009)

# Reference

Allan, P. (2015). *The Surprising Benefits of Role-Playing Games (and How to Get Started)*. [online] Lifehacker.com. Available at: <https://lifehacker.com/the-surprising-benefits-of-role-playing-games-and-how-1684582789> [Accessed 14 May 2019].

Brown, L. (2019). *Digital Video Game Trends and Stats for 2019*. [online] Wondershare. Available at: <https://filmora.wondershare.com/infographic/video-game-trends-and-stats.html> [Accessed 16 Nov. 2018].

Esrb.org. (n.d.). *Rating categories, content descriptors, and interactive elements from ESRB*. [online] Available at: <https://www.esrb.org/ratings/ratings_guide.aspx> [Accessed 14 Nov. 2018].

Gillian Smith (2018)

Smith, G., Cha, M. and Whitehead, J. (2008). *A framework for analysis of 2D platformer levels*. Ph.D. Northeastern University.

Hughey, D. (2009). *Comparing Traditional Systems Analysis and Design with Agile Methodologies*. [image] Available at: http://waterfall.html [Accessed 2 Apr. 2019].

Medium. (2017). *THE ONLY ADVICE YOU WILL NEED TO MAKE A GREAT GAME UI/UX*. [online] Available at: <https://medium.com/ironequal/the-only-advice-you-will-need-to-make-a-great-game-ui-ux-74a0db8de642> [Accessed 14 Mar. 2019].

Peacocke, M., Teather, R., Carette, J. and MacKenzie, I. (2015). *Evaluating the effectiveness of HUDs and diegetic ammo displays in first-person shooter games - IEEE Conference Publication*. [online] Ieeexplore.ieee.org. Available at: <https://ieeexplore.ieee.org/abstract/document/7377211/figures#figures> [Accessed 18 Nov. 2018].

Russell, K. (2018). *The future of 2D gaming*. [online] GamesIndustry.biz. Available at: <https://www.gamesindustry.biz/articles/2018-11-27-the-future-of-2d-gaming> [Accessed 3 Mar. 2019].

Sunday Sundae. (2018). *Unity VS Unreal - Which Engine Should You Choose? - Sunday Sundae*. [online] Available at: <https://sundaysundae.co/unity-vs-unreal/> [Accessed 10 Nov. 2018].

Sweeney, T. (2014). *Welcome to Unreal Engine 4*. [online] Unreal Engine. Available at: <https://www.unrealengine.com/en-US/blog/welcome-to-unreal-engine-4> [Accessed 14 Nov. 2018].

Taylor, D. (2013). *Ten Principles of Good Level Design (Part 2)*. [online] Gamasutra.com. Available at: <http://www.gamasutra.com/blogs/DanTaylor/20131006/197209/Ten_Principles_of_Good_Level_Design_Part_2.php> [Accessed 10 Mar. 2019].

tabletgames.tech/. (2018). *2d, 2.5d and 3d mobile game development | part2 | tabletgames.tech/*. [online] Available at: <http://tabletgames.tech/2d-2-5d-and-3d-mobile-game-development-part2/> [Accessed 4 Mar. 2019].

Techopedia.com. (n.d.). *What is Role-Playing Game (RPG)? - Definition from Techopedia*. [online] Available at: <https://www.techopedia.com/definition/27052/role-playing-game-rpg> [Accessed 15 Mar. 2019].

Unity. (n.d.). *Create 2D and 3D games with Unity - Unity*. [online] Available at: <https://Unity3d.com/difference-between-2d-and-3d-games> [Accessed 20 Feb. 2019].

Vandhanaa, M. (2019). *2D vs 3D Animation — Pros and Cons*. [online] Medium. Available at: <https://medium.com/@vandhana.appiness/2d-vs-3d-animation-pros-and-cons-51daba16a4ec> [Accessed 9 Mar. 2019].

Game art:

PiñaSinPizza (2017). *Simple Coin (Editable)*. [image] Available at: <https://opengameart.org/content/simple-coin-editable> [Accessed 6 May 2019].

kenney (n.d.). *Top-down Tanks Redux*. [image] Available at: <https://www.kenney.nl/assets/topdown-tanks-redux> [Accessed 23 Feb. 2019].

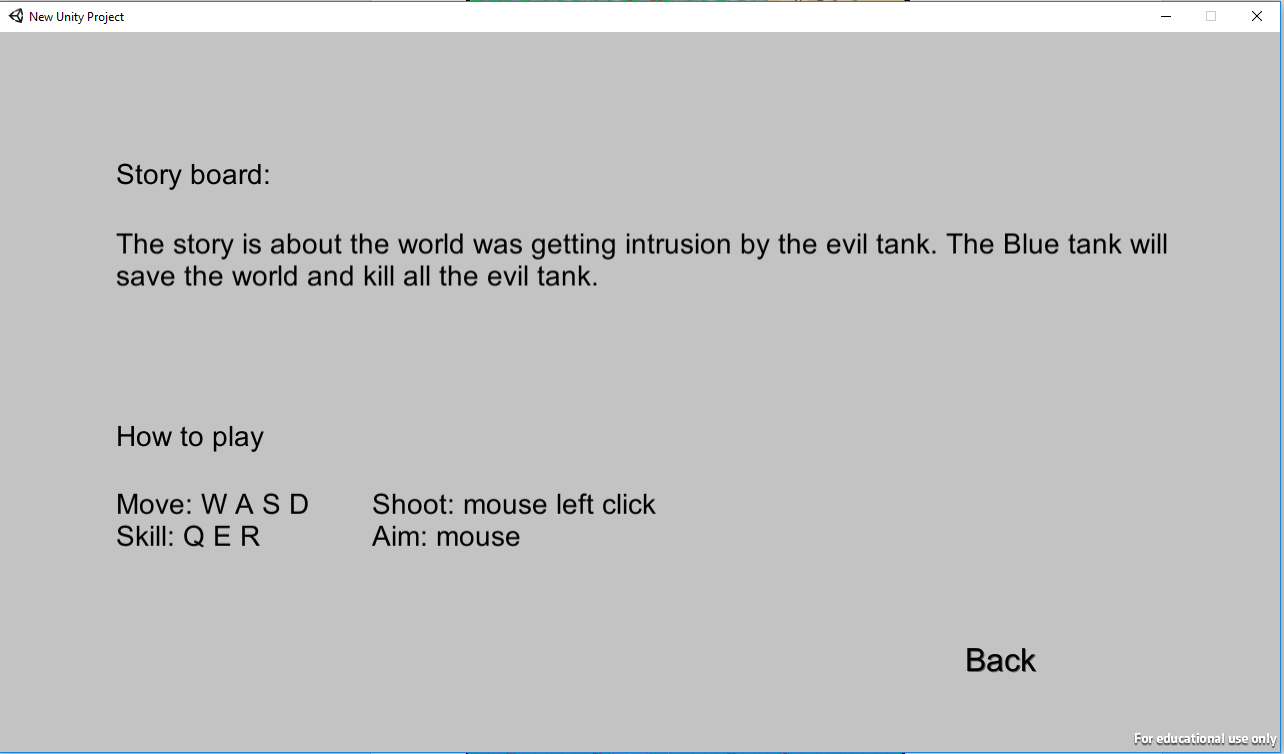
# The Appendices

## The game design

This is screen shot of the game. It show the more the detail about the game. This main game page. It has three option of the player.



This the scene of the Help button, it show the game story and the player control.



This is the mouse cursor, it change to this image. The gray cursor is the normal mouse, if the cursor detect the enemy will change to red cursor.

This the NPC take or give the information to the player.

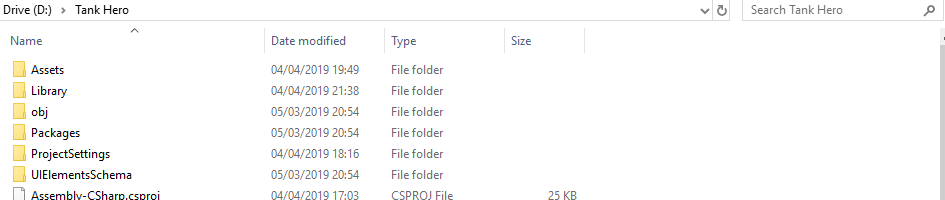


The NPC give the mission to the player. The mission is about unblock the next level exit.

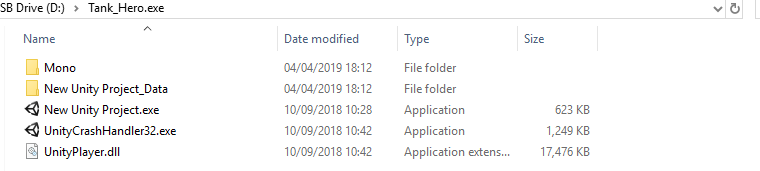


## File

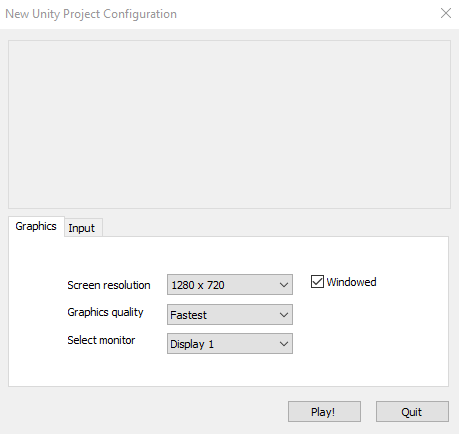
There has two files of the game. One is the game code, art and project setting, it can use the Unity to open the project and edit the game detail.



The other is the executable file and data file. The user can open the executable file to play the game.

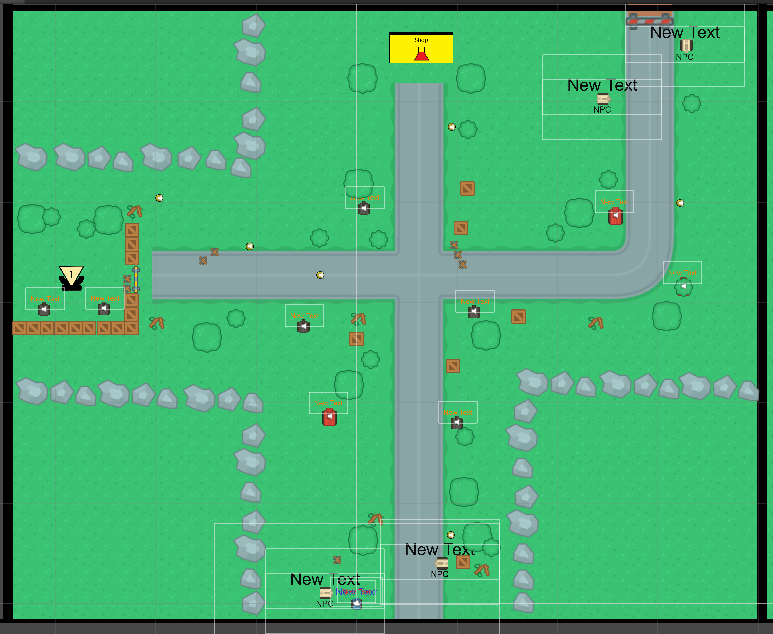


I will recommend screen resolution is 1280 x 720 and windowed. Click play to open the game.

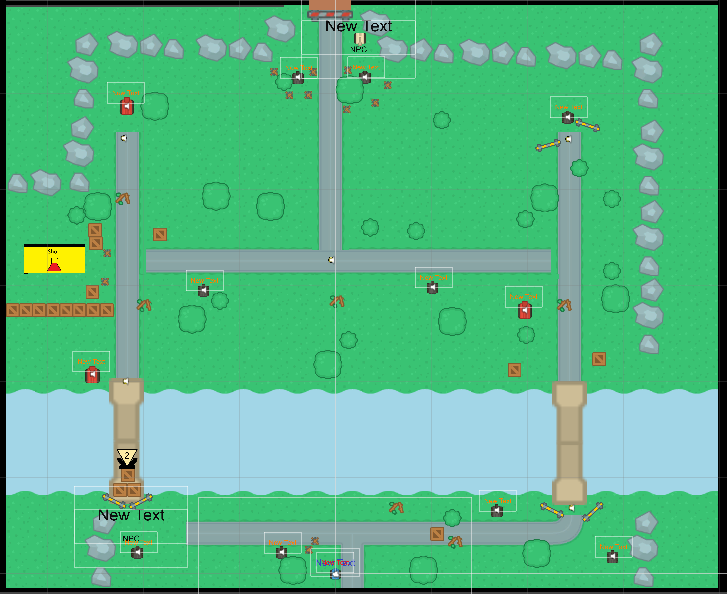


## Map design

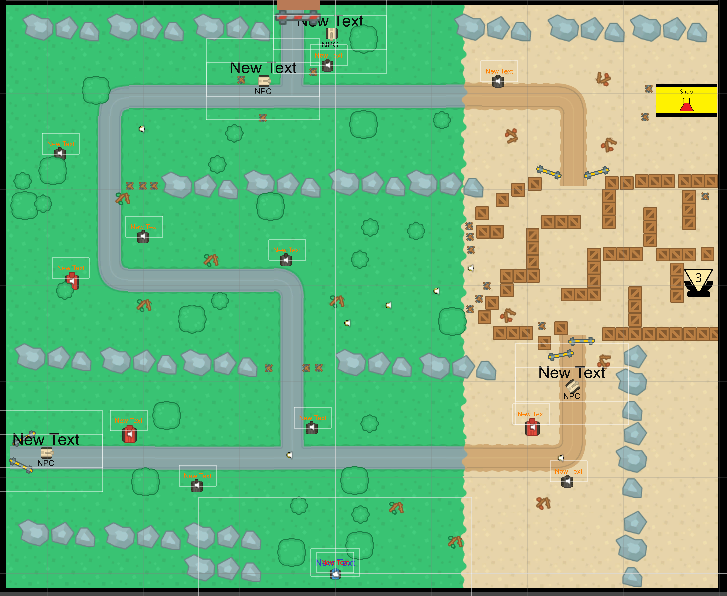
This is the map design of the game. It have 3 level and 1 Boss level. It show the next level exit, shop, enemy position, trophy and trap



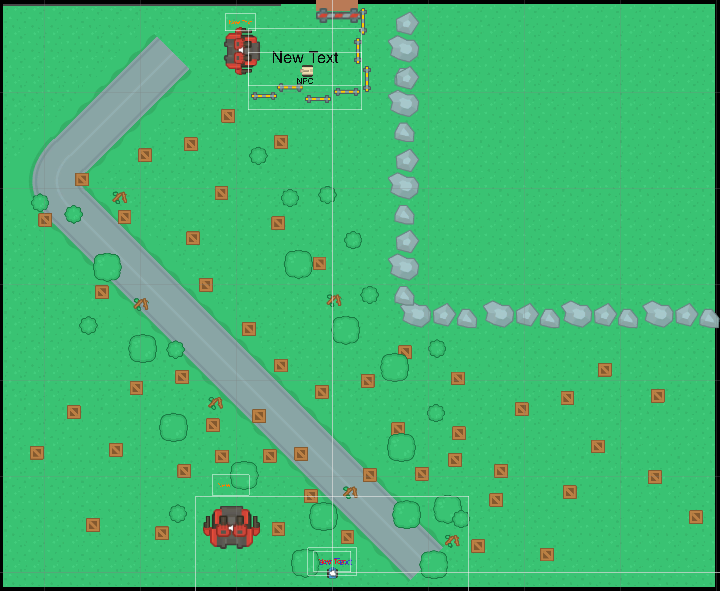
(The first level)



(The second level)



(The third level)



(Boss level)

## Test plan

User input

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test number | Description | The user input/ game test | Expected Outcome | actual outcome |
| 0.01 | Blue tank move up | User pressed W | Blue tank move up | As expected |
| 0.02 | Blue Tank move down | User pressed S | Blue tank move down | As expected |
| 0.03 | Blue Tank move left | User pressed A | Blue tank move left | As expected |
| 0.04 | Blue Tank move right | User pressed D | Blue tank move right | As expected |
| 0.05 | Blue Tank used skill1 to increase the attack speed | User pressed Q | Blue tank shooting faster | As expected |
| 0.06 | Blue Tank used skill2 to increase to move speed | User pressed E | Blue tank move faster | As expected |
| 0.07 | Blue Tank used skill3 to create the shield | User pressed R | Create the shield infant of the blue tank | As expected |
| 0.08 | Blue Tank shoot bullet follow the mouse | User pressed mouse left | Blue tank shoot red bullet | As expected |
| 0.09 | Audio the player shoot | The player shooting | Have the audio for shoot sound | As expected |

Enemy test

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test number | Description | The user input/ game test | Expected Outcome | actual outcome |
| 0.01 | Enemy can shoot the bullet | Enemy find the player tank | Enemy tank can shoot the green bullet | As expected |
| 0.02 | Shotgun shooting | Enemy find the player tank | Enemy tank can shoot six the green bullets in same time and different direction | As expected |
| 0.03 | AI of the enemy | The enemy rotate, follow and find to the player and shoot. | The enemy will follow the player and shoot to the player | As expected |
| 0.04 | Enemy movement | The enemy can move and follow the player | The enemy can move to the player | As expected |
| 0.05 | Enemy get hit | The enemy take the red bullet | It show the hit text -1, The enemy health -1, it have player the enemy hit audio. | As expected |
| 0.06 | Enemy explosion animation | The player kill the enemy show the explosion animation | The enemy die show the explosion | As expected |

Player tank

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test number | Description | The user input/ game test | Expected Outcome | actual outcome |
| 0.01 | The player level up have the skill point, heal and Level text | The player kill the enemy to level up | The player HP +1, in the panel get 1 skill point and shot the Level up text | As expected |
| 0.02 | The player get hit will show the Hp-1 text. | The player take the enemy bullet. | The player health will -1 and show the text HP-1 | As expected |
| 0.03 | The player get hit play the audio | The player take the enemy bullet. | The player will play hit player audio | As expected |
| 0.04 | The player Health = 0 show the game over panel | The player take damage health = 0 | The canvas show the game over panel | As expected |
| 0.05 | The player get the coin and play audio | The player get the coin in the map | The player have the 100 coin and have the coin audio | Some the coin cannot get and no audio |
| 0.06 | The player kill the enemy to get score | The player kill the enemy. | The player get 100 score | As expected |
| 0.07 | The fire point follow the mouse to move | The player shooting in different angle | The red bullet will shoot in angle of the mouse | As expected |

Scene

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test number | Description | The user input/ game test | Expected Outcome | actual outcome |
| 0.01 | Start the game and go to scene level1 | The user click start button in the main menu | The user enter the game. | As expected |
| 0.02 | Go the Help scene | The user click help button in the main menu | The user enter the help scene. | As expected |
| 0.03 | Close the game window | The user click Quit button in the main menu | The user close window and exit the game. | As expected |
| 0.04 | Test the play Again and back the menu function. | The player needs the die in each level and click the option of the game over panel. | The player can play again in each level and the back to main function is work. | As expected |

UI

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test number | Description | The user input/ game test | Expected Outcome | actual outcome |
| 0.01 | show the HP text of the player | Can see on the left corner. | It show the number of the health | As expected |
| 0.02 | show the Coin text of the player | Can see on the left corner. | It show the Coin when the player get the coin | As expected |
| 0.03 | show the level text of the player | Can see on the left corner. | It show the level when the player get level up | As expected |
| 0.04 | Show the HP bar and HP >3 change to red | the player take damage > 3 | The HP bar get low and low when the player take damage | As expected |
| 0.05 | Show the Level bar and full the bar will reset | The player kill enemy and get level up. | The level bar get increase by the player kill enemy. | As expected |
| 0.06 | The menu button can open the panel about skill tree and trophy. Open the menu panel can pause the game | The user click the menu button in the game. The menu button is in the right corner. | Click the menu open the panel and pause the game. | As expected |
| 0.07 | Skill tree panel can actives the skill | The player need to have the skill point, open skill tree and click the skill button. | The click the button in the skill tree panel can actives the skill and able to use | As expected |
| 0.08 | Trophy panel to show the trophy | The player get trophy in the map | The player gets the trophy, it can open the panel see the trophy the player got | As expected |

Camera

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test number | Description | The user input/ game test | Expected Outcome | actual outcome |
| 0.01 | The main camera follow the player tank. | The player move | The player tank will always in the middle of the camera. | As expected |
| 0.02 | Make the mouse to crosshair | The player can see the mouse was change. | The mouse change to the crosshair | Sometime cannot change it back to the grey cursor |

Save data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test number | Description | The user input/ game test | Expected Outcome | actual outcome |
| 0.01 | The next level object can save the data | The player go through to other level | The data was success save | As expected |
| 0.02 | Reset the player data when it quit. | The player quit game and play again. | The data can be reset | As expected |

NPC

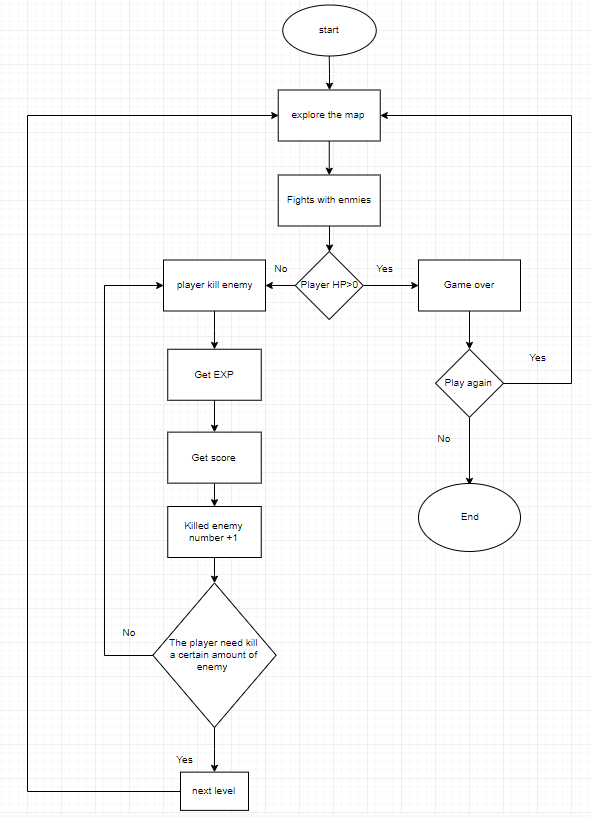
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test number | Description | The user input/ game test | Expected Outcome | actual outcome |
| 0.01 | Detects the player show the text/information | The player get close to NPC | The NPC show the text to the player. | As expected |

2D Box collide

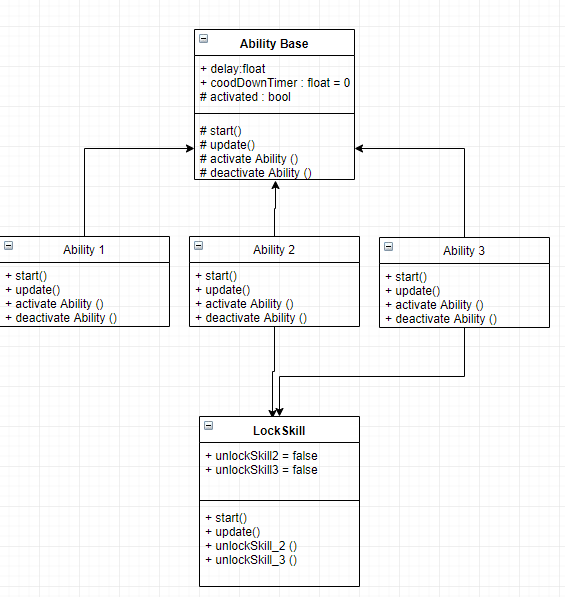
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test number | Description | The user input/ game test | Expected Outcome | actual outcome |
| 0.01 | The tree can hide the player and the enemy cannot detect. | The player hide in the tree, wait the enemy shoot, move and rotate. | The enemy do not attack the player in the tree | The tree near the road some time not working. |
| 0.02 | The player on the road can move faster | The player walk on the road and grass. | The player walk fast on the road. | As expected |
| 0.03 | The trap collide the player and enemy health -1. The player cannot shoot over the trap | The player or enemy collide the trap | The player and enemy can destroy the trap, health -1 | The player and enemy can health-1. But the enemy cannot destroy the trap. |
| 0.04 | The potion need the 500 coin to get 5 health | The player have enough coin to buy potion. | The player can have +3 heal, but lose 500 coin. | As expected |
| 0.05 | Fence Red/ Yellow can block the player. | The player go through fence red/ yellow | The player cannot go through fence red/ yellow | As expected |

## Diagram

This is the main game loop, it is the method of the player go to the level. The player need to kill a certain amount of enemies. If the player HP = 0, it will show the game over.



This diagram is about the ability class to active the skill and the base set of the skill. The Ability base is set the class of other skill to inherit.



## Historical status information

### Project Contract

**MAT3451\_1819\_520 Final Year Project**

**Student Name: Ming Hei LUI**

**P-number: P15245731**

**Programme: Computer games programming**

**Email address: luiminghei@hotmail.com**

**Project Title: 2D RPG game**

**Project Proposer: Self**

**Supervisor**

Supervisor name: Conor Fahy

Email: [conorfahy@dmu.ac.uk](mailto:conorfahy@dmu.ac.uk)

Affiliation: IMAT2608\_1718\_520 Mobile Games lab tutor

**Introduction**

In this Project will be make a 2D RPG game in unity. The game is about the character needs to escape the maze and find the exit. In the maze, the player needs use weapon or item to survive and fight with the enemies.

**Project Background**

In this game will make a unity game. This game will use unity, C# and java to program the game physics. The game is 2D action RPG game. This game will use PC platform. The target for the game is kids to adults. The play has five different chapters and map to play. The player has to step by step go through the chapters.

The game story is about the characters how to escape the maze. In the maze there are many different enemies and trap. To survive, the player can use different weapons to shot enemies. Also the player has health bar, level bar and buying system can buy some items to upgrade, heal and unlock more weapon.

In the 2D RPG planform have many different games. Those games are stimulate to this project make the new one. .The special of the game is the player can add the element each as fire, ice and lightning for the weapon. It can keep hurt, slow down and pause the enemies.

Choose RPG game can the make player step by step to know about the game world. The player can easier to enjoy playing the game, also the player can use the problem-solving skills to overcome the puzzle.

**Aim/****Objectives/****Deliverables**

Aims:

The project can be played 8-10min and has at least 4-5 maps.

Objectives:

Software Development project include the objectives as

1. To design and implement the game interface
2. To make a test and debug plan
3. To implement achievement by collection system
4. To implement the upgrade system by collecting the coin
5. To design the audio e.g. background music, hurt sound
6. To implement different chapters/map
7. To implement the character and enemies.

Deliverables:

|  |  |
| --- | --- |
|  | **Development Projects** |
| **Final Submission**  These are some examples: each project will need a complete set of objectives/deliverables  Week 27 | * Project contract * Ethics form * Project Plan * Global Checklist * Aim/Objectives * Use UML Diagrams/ Flow chart/ OO design * Story boards/ UI design * Audio * Game Art/ texture * Game design idea * Test/debug Plan * Appendices (e.g. Reference) * Final Year Project Report word count: >10.000 |
| **Viva examination:** attended by the supervisor and the 2nd marker  Weeks 31-33 | * Presentation(PPT) |

**Resources and Constraints**

1. Software: Unity, notepad++, visual studio, MonoDevelop
2. Hardware: windows 7 and above, 4GB RAM, storage 4GB available space

Constraints

1. Time, the project have six month make the job done.
2. Availability of the unity resources, it can help to program the game.

**Sources of Information**

1. Internet sources

YouTube <https://www.youtube.com/>

Unity3D <https://unity3d.com/learn/tutorials/s/2d-game-creation>

OpenGameArt <https://opengameart.org/>

Game Art 2D <https://www.gameart2d.com/>

1. De Montfort university blackboard:

IMAT2608\_1718 Mobile Games

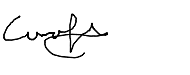
IMAT1606\_1617\_520 Game Architecture Design & Develpt

**Risk Analysis**

The risk of the project is time management, it will not finish the objectives on the project plan. Also in the end the game may not be run as expected. In the develop process risk is programming problem, the program cannot merge and work together. It will get error and crash in the game. Also the program needs to always backup avoid lose data.**Schedule of Activities**

**Student\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date\_\_25/10/2018\_\_\_\_**

**Proposer** (if other from the student and/or the supervisor)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **Date\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Supervisor\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Date\_\_\_25/10/2018\_\_\_\_\_\_\_\_\_\_\_\_**

Keep the signed copy somewhere safe: include it with your initial submission. Your supervisor will require a copy as well.

### Ethical Review

**IMAT3451 FINAL YEAR PROJECT - ETHICAL REVIEW FORM**

The University requires all undergraduate final year projects to undergo an ethical review and, where human research ethical issues are identified, to ensure that these issues are addressed.

For the majority of Computing Final Year Projects, the outcome will be either ‘No ethical issues’ or ‘Minor/Major ethical issues which have been addressed’; in these cases approval can be given by the supervisor. In the unlikely event that the outcome is ‘Ethical issues that have not been addressed’, the completed form will need to be forwarded to the Faculty Research Ethics Committee.

**Student Name** **Programme**

Ming Hei LUI

Computer games programming

**Project Title**

2D RPG game

**Brief description of proposed activity and its objectives:**

In this Project will be a 2D RPG game in unity. The game is about the characters how to escape the maze. The player needs to fight and escape the map.

The objectives are to implement the game included audio, test, player, map and game systems.

**Ethical Issues Identified: How these will be addressed:**

(see overleaf)

The information and the data will be preservation and confidentiality in project

Gathering information about human being through to improve the game.

The game may save the data for the player.

**Checklist**

Has the project proposal identified any of the following research procedures?

1. Gathering information about human beings through: Interviewing, Surveying,

Questionnaires, Observation of human behaviour Yes / ~~No~~

2. Using archived data in which individuals are identifiable Yes / ~~No~~

3. Researching into illegal activities, activities at the margins of the law or

activities that have a risk of personal injury ~~Yes~~ / No

4. Supporting innovation that might impact on human behaviour

e.g. Behavioural Studies ~~Yes~~ / No

**If ‘Yes’ to any of 1-4 above: have you considered the following?**

🞏 Providing participants with full details of the objectives of the research

🞏 Providing information appropriate for those whose first language is not English

🞏 Voluntary participation with informed consent

🞏 Written description of involvement

🞏 Freedom to withdraw

☑ Keeping appropriate records

🞏 Signed acknowledgement and understanding by participants

🞏 Consideration of relevant codes of conduct/guidelines

**Ethical Review Outcome**

🞏 1. No ethical issues

☑ 2. Minor ethical issues which have been addressed and concerns resolved

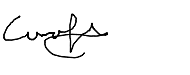
🞏 3. Major ethical issues which have been addressed and concerns resolved

🞏 4. Ethical issues that have not been resolved/addressed

**Authorisation**

*If the outcome is no. 3 or 4 above, this form should be forwarded to the Faculty Research Ethics Committee.*

Signature of student \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_**25/10/2018**\_\_\_

Signature of supervisor ****\_\_\_\_\_\_\_\_\_ Date 25/10/2018\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### Global Checklist

**IMAT3451 FINAL YEAR PROJECT - Global Checklist**

The University requires all undergraduate final year projects students to undertake a global review of their project. Here is an International Impact Checklist for you to complete, which can be done in consultation with the project supervisor.

**Student Name** **Programme**

Ming Hei LUI

Computer games programming

**Project Title**

2D RPG game

**Please indicate which of these possible attributes is addressed by your undertaking of this project.**

|  |  |
| --- | --- |
| **Possible Global Experience** | **Addressed by Project** |
| Ability to work collaboratively: teams from a range of backgrounds and countries | None |
| Excellent communication skills with a sensitivity to speaking with and listening to non-native English speakers | ✓ |
| An ability to embrace multiple perspectives and challenge thinking in a range of cultural context | None |
| A capacity to develop new skills and behaviours according to role requirements | ✓ |
| An ability to negotiate and influence clients across the globe from different cultures | ✓ |
| An ability to form professional, global networks | None |
| An openness to/respect of a range of perspectives from around the world | None |
| Multi-cultural learning agility (i.e. able to learn in any culture or environment) | None |

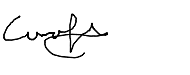
**Brief description of how the ticked attributes have been addressed:**

The project will communicate the non-native English speakers get more idea to improve the game.

The project can improve the knowledge of unity and game programming.

The project will have other cultures in the game. It makes the player from another county can have aroused.

Signature of student \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_**25/10/2018**\_\_\_\_

Signature of supervisor \_\_\_\_\_\_\_**** Date \_\_\_\_\_\_25/10/2018\_\_\_\_\_\_\_\_\_\_

## Source coding

### Ability

This is the code about the Ability, it has Ability1, 2, 3 and the base.

#### Ability 1

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Ability1 : AbilityBase {

    private playerMovement PlayerMovement;

    //this is the Ability 1 skill

    //The skill can change the speed of the player tank.

    void Start ()

    {

        // set the default Move Speed

        PlayerMovement = GameObject.FindGameObjectWithTag("Player").GetComponent<playerMovement>();

        PlayerMovement.defaultMoveSpeed = 3;

    }

    // Update is called once per frame

    void Update () {

        base.Update();

            // Key press E can activate the skill

            if (Input.GetKeyDown(KeyCode.E))

    {

                activated = true;

    print("space key E was pressed");

    }

    }

    // This is the skill speed

    public override void activateAbility()

    {

            PlayerMovement.moveSpeed =10;

    }

    // after using skill change to the defaultMoveSpeed

    public override void deactivateAbility()

    {

            PlayerMovement.moveSpeed =PlayerMovement.defaultMoveSpeed;

    }

}

#### Ability 2

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Ability2 : AbilityBase {

private playerShooting PlayerShooting;

    private LockSkill unlockNewSkill;

    void Start () {

        PlayerShooting = GameObject.FindGameObjectWithTag("Player").GetComponent<playerShooting>();

        unlockNewSkill = GetComponent<LockSkill>();

    }

    void Update () {

        base.Update();

            //check unlock the skill

            if(unlockNewSkill.unlockSkill2 == true)

            {

                //key press Q to use the skill

                if (Input.GetKeyDown(KeyCode.Q))

        {

                    activated = true;

        print("space key Q was pressed");

        }

            }

        //}

    }

    //The chabge the fire delay time to shoot faster

    public override void activateAbility()

    {

PlayerShooting.fireDelay =0.2f;

         print("skill 2 used");

    }

    // set the default shooting speed

    public override void deactivateAbility()

    {

        PlayerShooting.fireDelay =0.5f;

    }

}

#### Ability 3

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Ability3 : AbilityBase {

    private playerMovement PlayerMovement;

    private SkillTreeButton playerSkillTreeButton;

    //public int getSkill3 = 0;

    private LockSkill unlockNewSkill;

    // Use this for initialization

    void Start () {

        //PlayerMovement = GameObject.FindGameObjectWithTag("Player").GetComponent<playerMovement>();

    //playerSkillTreeButton = GameObject.FindGameObjectWithTag("UI").GetComponent<SkillTreeButton>();

        unlockNewSkill = GetComponent<LockSkill>();

    }

    // Update is called once per frame

    void Update () {

        base.Update();

        //if(lockSkill = unlockNewSkill.unlockSkill)

        // the skill key to use the skill

        if(unlockNewSkill.unlockSkill3 == true)

            {

            if (Input.GetKeyDown(KeyCode.R))

    {

                activated = true;

    print("space key Q was pressed");

    }

        }

    }

    // active the shield

    public override void activateAbility()

    {

                bool isActive = playerShield.activeSelf;

                playerShield.SetActive(true);

                print("SetActive(true)");

    }

    // not active the shield

    public override void deactivateAbility()

    {

        playerShield.SetActive(false);

    }

}

#### Ability Base

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Ability3 : AbilityBase {

    private playerMovement PlayerMovement;

    private SkillTreeButton playerSkillTreeButton;

    //public int getSkill3 = 0;

    private LockSkill unlockNewSkill;

    // Use this for initialization

    void Start () {

        //PlayerMovement = GameObject.FindGameObjectWithTag("Player").GetComponent<playerMovement>();

    //playerSkillTreeButton = GameObject.FindGameObjectWithTag("UI").GetComponent<SkillTreeButton>();

        unlockNewSkill = GetComponent<LockSkill>();

    }

    // Update is called once per frame

    void Update () {

        base.Update();

        //if(lockSkill = unlockNewSkill.unlockSkill)

        // the skill key to use the skill

        if(unlockNewSkill.unlockSkill3 == true)

            {

            if (Input.GetKeyDown(KeyCode.R))

    {

                activated = true;

    print("space key Q was pressed");

    }

        }

    }

    // active the shield

    public override void activateAbility()

    {

                bool isActive = playerShield.activeSelf;

                playerShield.SetActive(true);

                print("SetActive(true)");

    }

    // not active the shield

    public override void deactivateAbility()

    {

        playerShield.SetActive(false);

    }

}

### Barricade

This is the barricade code to make the trap

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Barricade : MonoBehaviour {

    //public GameObject DestroyBarricade;

    private DamageHandler damageHandler;

    // Use this for initialization

    void Start () {

        damageHandler = GameObject.FindGameObjectWithTag("Enemy").GetComponent<DamageHandler>();

    }

    // Update is called once per frame

    void Update () {

    }

    void OnTriggerEnter2D(Collider2D other) {

        if (other.gameObject.tag == "Enemy")

        {

            FloatTextEnemy floatTextEnemy = other.gameObject.GetComponent<FloatTextEnemy>();

            floatTextEnemy.SpawnText ();

            damageHandler.health -=1;

            Destroy(gameObject);

        }

    }

}

#### Barricade of player

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class BarricadePlayer : MonoBehaviour {

    private PlayerHealth playerHealth;

    // Use this for initialization

    void Start () {

        playerHealth = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerHealth>();

    }

    // Update is called once per frame

    void Update () {

    }

    void OnTriggerEnter2D(Collider2D other) {

        if (other.gameObject.tag == "Player")

        {

            ClickMe clickMe = other.gameObject.GetComponent<ClickMe>();

            clickMe.SpawnText ();

            playerHealth.health -=1;

            Destroy(gameObject);

        }

    }

}

### Scene

This is the game change the scene.

#### Change level

using UnityEngine;

using UnityEngine.SceneManagement;

using System.Collections;

public class ToGame2 : MonoBehaviour {

    // This script is to the other scene call level1

    public GameObject Player;

    void OnCollisionEnter2D (Collision2D collision)

    {

        if(collision.gameObject.tag == "Player")        //If collides with Player

        {

            //Player.gameObject.SetActive(false);           //Make it invisible

            SceneManager.LoadScene("level1");           //Load scene "game02"

        }

        if(PlayerPrefs.GetInt("skill1")==1)

        {

            PlayerPrefs.SetInt("skill1", 0);

        }

        if(PlayerPrefs.GetInt("skill0")==1)

        {

            PlayerPrefs.SetInt("skill0", 0);

        }

    }

}

##### Change level2

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class ToGame3 : MonoBehaviour {

    // This script is to the other scene call level2

    public GameObject Player;

    void OnCollisionEnter2D (Collision2D collision)

    {

        if(collision.gameObject.tag == "Player")        //If collides with Player

        {

            //Player.gameObject.SetActive(false);           //Make it invisible

            SceneManager.LoadScene("level2");           //Load scene "Win02"

        }

    }

}

##### Change level3

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class ToGame4 : MonoBehaviour {

    // This script is to the other scene call level3

    // Use this for initialization

    void Start () {

    }

    // Update is called once per frame

    void Update () {

    }

    void OnCollisionEnter2D (Collision2D collision)

    {

        if(collision.gameObject.tag == "Player")        //If collides with Player

        {

            //Player.gameObject.SetActive(false);           //Make it invisible

            SceneManager.LoadScene("Level3");           //Load scene "Win02"

        }

    }

}

##### Change level4

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class ToGame5 : MonoBehaviour {

    // This script is to the other scene call Boss level

    void Start () {

    }

    // Update is called once per frame

    void Update () {

    }

    void OnCollisionEnter2D (Collision2D collision)

    {

        if(collision.gameObject.tag == "Player")        //If collides with Player

        {

            //Player.gameObject.SetActive(false);           //Make it invisible

            SceneManager.LoadScene("BossLevel");            //Load scene "Win02"

        }

    }

}

#### HelpToMeun

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

using UnityEngine.UI;

public class HelpToMenu : MonoBehaviour {

    //This script is in the help scene and go back to the main menu

    void Start () {

    }

    public void BackToMenu () {

        /\* if(BackButton == !null)

        {

SceneManager.LoadScene("Menu");

        }\*/

        SceneManager.LoadScene("Menu");

    }

    // Update is called once per frame

    void Update () {

    }

}

#### MainStart

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

using UnityEngine.UI;

public class MainStart : MonoBehaviour {

//This script is about the main menu change scene

public Button startButton;

public Button HelpButton;

public Button exitButton;

public void Start()

{

startButton = startButton.GetComponent<Button>();

//HelpButton = HelpButton.GetComponent<Button>();

exitButton = exitButton.GetComponent<Button>();

}

// Use this for initialization

public void StartLevel () {

SceneManager.LoadScene("Game");

//PlayerPrefs.DeleteAll ();

    }

public void HelpScene () {

SceneManager.LoadScene("Help");

    }

public void ExitGame()

{

Application.Quit();

}

}

#### GameOver1

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

using UnityEngine.UI;

public class GameOver1 : MonoBehaviour {

    //This script is player game over to the scnce, resset the HP score and LV

    private PlayerCoin playerCoin;

    private ScoreScipt playerScore;

    private PlayerHealth playerHealth;

    private PlayerLevel playerLevelClass;

    private SkillTreeButton skillTreeButton;

    //private PlayerLevel playerLevelClass;

    //public GameObject gameOverPanel;

    // Use this for initialization

    void Start () {

        playerCoin = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerCoin>();

        playerScore = GameObject.FindGameObjectWithTag("UI").GetComponent<ScoreScipt>();

        playerHealth = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerHealth>();

        playerLevelClass = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerLevel>();

        skillTreeButton = GameObject.FindGameObjectWithTag("Player").GetComponent<SkillTreeButton>();

    }

    // Update is called once per frame

    void Update () {

    }

    public void playAgainLevel\_1()

    {

        SceneManager.LoadScene("level1");

        //gameOverPanel.SetActive(false);

    }

    public void playAgainLevel\_2()

    {

        SceneManager.LoadScene("level2");

        //gameOverPanel.SetActive(false);

    }

    public void playAgainLevel\_3()

    {

        SceneManager.LoadScene("level3");

        //gameOverPanel.SetActive(false);

    }

    public void playAgainBossLevel\_1()

    {

        SceneManager.LoadScene("BossLevel");

        //gameOverPanel.SetActive(false);

    }

    public void BackToMain()

    {

        SceneManager.LoadScene("Menu");

        //PlayerPrefs.DeleteAll ();

        //gameOverPanel.SetActive(false);

    }

    public void resetPlayerCoinData ()

    {

        playerCoin.currentPlayerCoinc = 0;

        PlayerPrefs.SetInt ("PlayerCoin", playerCoin.currentPlayerCoinc);

        PlayerPrefs.DeleteAll();

    }

    public void resetPlayerScoreData ()

    {

        ScoreScipt.scoreValue = 0;

        PlayerPrefs.SetInt ("PlayerScore", ScoreScipt.scoreValue);

    }

    public void resetPlayerHealthData ()

    {

        playerHealth.health = 10;

        PlayerPrefs.SetInt ("PlayerHP", playerHealth.health);

    }

    public void resetPlayerLevelData ()

    {

        playerLevelClass.playerLevel = 1;

        PlayerPrefs.SetInt ("PlayerLevel", playerLevelClass.playerLevel);

    }

    public void resetPlayerSkill ()

    {

        PlayerPrefs.SetInt ("Skill", 0);

    }

    public void resetPlayerExpData ()

    {

        playerLevelClass.playerExp = 0;

        PlayerPrefs.SetInt ("PlayerExp", playerLevelClass.playerExp);

    }

}

#### CamaeraaMove

using UnityEngine;

using System.Collections;

public class CameraaMove : MonoBehaviour {

    // set the camera follow the player.

    public GameObject player;

    private Vector3 offset;

    // Use this for initialization

    void Start () {

        offset = transform.position - player.transform.position;

    }

    // Update is called once per frame

    void LateUpdate ()

    {

        // Set the position of the camera's transform to be the same as the player's, but offset by the calculated offset distance.

        transform.position = player.transform.position + offset;

    }

}

#### Back Main scene

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

using UnityEngine.UI;

public class ToMain : MonoBehaviour {

    // This script is reset the coin and score when the player back to main menu

    private PlayerCoin playerCoin;

    private ScoreScipt playerScore;

    //private PlayerHealth playerHealth;

    //private PlayerLevel playerLevelClass;

    //private SkillTreeButton skillTreeButton;

    // Use this for initialization

    void Start () {

        playerCoin = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerCoin>();

        playerScore = GameObject.FindGameObjectWithTag("UI").GetComponent<ScoreScipt>();

        //playerHealth = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerHealth>();

        //playerLevelClass = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerLevel>();

        //skillTreeButton = GameObject.FindGameObjectWithTag("Player").GetComponent<SkillTreeButton>();

    }

    // Update is called once per frame

    void Update () {

    }

    void OnCollisionEnter2D (Collision2D collision)

    {

        if(collision.gameObject.tag == "Player")        //If collides with Player

        {

            //Player.gameObject.SetActive(false);           //Make it invisible

            SceneManager.LoadScene("Menu");         //Load scene "Win02"

            playerCoin.currentPlayerCoinc = 0;

            PlayerPrefs.SetInt ("PlayerCoin", playerCoin.currentPlayerCoinc);

            ScoreScipt.scoreValue = 0;

            PlayerPrefs.SetInt ("PlayerScore", ScoreScipt.scoreValue);

        }

    }

}

#### GameMenu

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

using UnityEngine.UI;

public class GameMenu : MonoBehaviour {

    //This script is game menu panel to active the trophy and skilltree panel system

    public GameObject panel;

    public GameObject trophyPanel;

    public GameObject skillTreeButtonPanel;

    //public GameObject backMenu;

        void Start () {

/\*      trophyPanel = GameObject.FindGameObjectWithTag("TrophyPanel");

        skillTreeButtonPanel = GameObject.FindGameObjectWithTag("SkillTreeButtonPanel"); \*/

        }

    // Use this for initialization

    public void OpenMenu () {

        if(panel != null)

        {

//////////////////////////

//pause time

//////////////////////////

            if(Time.timeScale == 1)

            Time.timeScale = 0;

            else

            Time.timeScale = 1;

//////////////////////////////////////////

//////////////////////////////////////////

            bool isActive = panel.activeSelf;

            panel.SetActive(!isActive);

            trophyPanel.SetActive(false);

            skillTreeButtonPanel.SetActive(false);

            //print("menu");

        }

    }

    public void backToMain()

    {

        /\* if(panel != null)

        {

        SceneManager.LoadScene("Menu");

        }

        panel.SetActive(false);\*/

        if(Time.timeScale == 1)

        Time.timeScale = 0;

        else

        Time.timeScale = 1;

        SceneManager.LoadScene("Menu");

    }

}

### Save data

#### SaveData

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class SaveData : MonoBehaviour {

// This script is about saving the player data, health, coin, level, trophy

private PlayerHealth playerHealth;

private PlayerCoin playerCoin;

private Torphy1 playerTrophy;

private Trophy2 playerTrophy2;

private Trophy3 playerTrophy3;

private PlayerLevel playerLevelClass;

//private AbilityBase[] abilityBases;

//private TrophyBase[] trophys;

    // Use this for initialization

    void Start () {

        playerHealth = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerHealth>();

        playerCoin = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerCoin>();

        playerLevelClass = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerLevel>();

        playerTrophy = GameObject.FindGameObjectWithTag("Trophy").GetComponent<Torphy1>();

        playerTrophy2 = GameObject.FindGameObjectWithTag("Trophy2").GetComponent<Trophy2>();

        playerTrophy3 = GameObject.FindGameObjectWithTag("Trophy3").GetComponent<Trophy3>();

        //playerScore = GameObject.FindGameObjectWithTag("UI").GetComponent<ScoreScipt>();

    }

    // Update is called once per frame

    void Update ()

    {

    }

    void OnCollisionEnter2D (Collision2D collision)

    {

        //PlayerHealth playerHealth = gameObject.GetComponent<PlayerHealth>();

        //PlayerCoin playerCoin = gameObject.GetComponent<PlayerCoin>();

        if(collision.gameObject.tag == "Player")        //If collides with Player

        {

            PlayerPrefs.SetInt ("PlayerLevel", playerLevelClass.playerLevel);

            PlayerPrefs.SetInt ("PlayerExp", playerLevelClass.playerExp);

            PlayerPrefs.SetInt ("PlayerHP", playerHealth.health);

            PlayerPrefs.SetInt ("PlayerCoin", playerCoin.currentPlayerCoinc);

            PlayerPrefs.SetInt ("PlayerScore", ScoreScipt.scoreValue);

            PlayerPrefs.SetInt ("Trophy", playerTrophy.getTrophy1);

            PlayerPrefs.SetInt ("Trophy2", playerTrophy2.getTrophy2);

            PlayerPrefs.SetInt ("Trophy3", playerTrophy3.getTrophy3);

            /\*for(int i = 0;i<trophys.Lenght;i++)

            {

                PlayerPrefs.SetInt ("Trophy"+ i.ToString(), trophys[i]);

            }\*/

        /\* for(int i = 0;i<trophys.Lenght;i++)

            {

                PlayerPrefs.SetInt ("Trophy"+ i.ToString(), trophys[i]);

            }

        \*/

        }

    }

}

#### LoadData

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class LoadData : MonoBehaviour {

//This script is load the data from playerPrets

private PlayerHealth playerHealth;

private PlayerCoin playerCoin;

private ScoreScipt playerScore;

private Torphy1 playerTrophy;

private Trophy2 playerTrophy2;

private Trophy3 playerTrophy3;

private PlayerLevel playerLevelClass;

public GameObject[] skills;

public SkillTreeButton skillTreeButton;

//skillTreeButton. GameObject[] skills

/\* private bool[3] Torphy; \*/

    // Use this for initialization

    void Start () {

        playerHealth = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerHealth>();

        playerCoin = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerCoin>();

        playerLevelClass = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerLevel>();

        playerTrophy = GameObject.FindGameObjectWithTag("Trophy").GetComponent<Torphy1>();

        playerTrophy2 = GameObject.FindGameObjectWithTag("Trophy2").GetComponent<Trophy2>();

        playerTrophy3 = GameObject.FindGameObjectWithTag("Trophy3").GetComponent<Trophy3>();

        //playerScore = GameObject.FindGameObjectWithTag("Player").GetComponent<ScoreScipt>();

        playerCoin.currentPlayerCoinc = PlayerPrefs.GetInt("PlayerCoin");

        playerHealth.health = PlayerPrefs.GetInt("PlayerHP");

        ScoreScipt.scoreValue = PlayerPrefs.GetInt("PlayerScore");

        playerLevelClass.playerLevel = PlayerPrefs.GetInt("PlayerLevel");

        playerLevelClass.playerExp = PlayerPrefs.GetInt("PlayerExp");

        playerTrophy.getTrophy1 = PlayerPrefs.GetInt("Trophy");

        playerTrophy2.getTrophy2 = PlayerPrefs.GetInt("Trophy2");

        playerTrophy3.getTrophy3 = PlayerPrefs.GetInt("Trophy3");

        for(int i = 0; i< skills.Length; i++)

        {

            print(PlayerPrefs.GetInt ("skill"+i.ToString()));

        }

    }

    // Update is called once per frame

    void Update () {

    }

}

### Trophy

This is the code about the trophy, it has trophy1, 2, 3 and the base.

#### Trophy1

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Torphy1 : TrophyBase {

    // This script is save the trophy and active in the trophy panel

    public int getTrophy1 = 0;

    public void TrophyGet()

    {

    }

    void Start ()

    {

    }

    // Update is called once per frame

    void Update ()

    {

        if (getTrophy1 == 1)

        {

            trophyMenu.SetActive(true);

            //print(getTrophy1);

        }

    }

    public override void OnTriggerEnter2D(Collider2D other)

    {

        if (other.gameObject.tag == "Player")

        {

            Destroy(gameObject);

            print("Get Trophy");

            if (base.trophyMenu != null)

            {

            bool isActive = trophyMenu.activeSelf;

            trophyMenu.SetActive(!isActive);

            getTrophy1 =1;

            print(getTrophy1);

            }

        }

    }

}

#### Trophy2

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Trophy2 : TrophyBase {

    // This script is save the trophy and active in the trophy panel

    public int getTrophy2 = 0;

    public void TrophyGet()

    {

    }

    void Start ()

    {

    }

    // Update is called once per frame

    void Update ()

    {

        if (getTrophy2 == 1)

        {

            trophyMenu.SetActive(true);

            print(getTrophy2);

        }

    }

    public override void OnTriggerEnter2D(Collider2D other)

    {

        if (other.gameObject.tag == "Player")

        {

            Destroy(gameObject);

            print("Get Trophy");

            if (base.trophyMenu != null)

            {

            bool isActive = trophyMenu.activeSelf;

            trophyMenu.SetActive(!isActive);

            getTrophy2 =1;

            print(getTrophy2);

            }

        }

    }

}

#### Trophy3

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Trophy3 : TrophyBase {

    // This script is save the trophy and active in the trophy panel

    public int getTrophy3 = 0;

    public void TrophyGet()

    {

    }

    void Start ()

    {

    }

    // Update is called once per frame

    void Update ()

    {

        if (getTrophy3 == 1)

        {

            trophyMenu.SetActive(true);

            print(getTrophy3);

        }

    }

    public override void OnTriggerEnter2D(Collider2D other)

    {

        if (other.gameObject.tag == "Player")

        {

            Destroy(gameObject);

            print("Get Trophy");

            if (base.trophyMenu != null)

            {

            bool isActive = trophyMenu.activeSelf;

            trophyMenu.SetActive(!isActive);

            getTrophy3 =1;

            print(getTrophy3);

            }

        }

    }

}

#### Trophy base

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class TrophyBase : MonoBehaviour {

    [SerializeField] protected GameObject trophy;

    [SerializeField] protected GameObject trophyMenu;

    public virtual void OnTriggerEnter2D(Collider2D other)

    {

    }

    // Use this for initialization

    void Start () {

    }

    // Update is called once per frame

    void Update () {

        /\* if (getTrophy1 == 1)

        {

            trophyMenu.SetActive(true);

            print(getTrophy1);

        }

        if (getTrophy2 == 1)

        {

            trophyMenu.SetActive(true);

            print(getTrophy2);

        }

        if (getTrophy3 == 1)

        {

            trophyMenu.SetActive(true);

            print(getTrophy3);

        }

\*/

    }

}

### Skill

#### SkillPointText

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class SkillPointText : MonoBehaviour {

    // This script is show the skill point text in the skill tree panel

    private int newSkillPoint;

    private PlayerSkillPoint playerSkillPoint;

    private Text playerSkillPointText;

    // Use this for initialization

    void Start () {

        playerSkillPointText = GetComponent<Text>();

        playerSkillPoint = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerSkillPoint>();

        newSkillPoint = playerSkillPoint.skillPoint;

    }

    // Update is called once per frame

    void Update () {

        newSkillPoint = playerSkillPoint.skillPoint;

        playerSkillPointText.text = "Skill Point:" + newSkillPoint;

    }

}

#### SkillAbility

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

using UnityEngine.UI;

public class SkillAbility : MonoBehaviour {

public Button skillSpeedUpButton;

public Button skillSpeedUp2Button;

public Button skillShootSpeedUp;

public float skill1Delay = 5.0f;

public float coolDownTimer = 0;

public float shootingSkillDelay = 3.0f;

public float shootingCoolDownTimer = 0;

public float speedUpLv2Delay = 10.0f;

public float speedUpLv2CoolDownTimer = 0;

public GameObject shield;

    private playerMovement PlayerMovement;

private playerShooting PlayerShooting;

    //private float speedUp;

public void Start()

{

skillSpeedUpButton = skillSpeedUpButton.GetComponent<Button>();

//PlayerMovement = GameObject.FindGameObjectWithTag("Player").GetComponent<playerMovement>();

        //PlayerShooting = GameObject.FindGameObjectWithTag("Player").GetComponent<playerShooting>();

//skill2Button = skill2Button.GetComponent<Button>();

}

void Update() {

coolDownTimer -= Time.deltaTime;

shootingCoolDownTimer-= Time.deltaTime;

speedUpLv2CoolDownTimer-= Time.deltaTime;

}

}

#### SkilltreeButton

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class SkillTreeButton : MonoBehaviour {

    // This script is about the player use the skill point to upgrade skill and using array list save skill

    // Abilitu Base using array list to create the skill.

    public GameObject skillsPanel;

    public GameObject skill2;

    public GameObject skillShootUp;

    public GameObject[] skills;

    //public int savedSkill2;

    private PlayerSkillPoint playerSkillPoint;

    private AbilityBase[] abilityBases;

    // Use this for initialization

    public void OpenSkillsMenu () {

        if(skillsPanel != null)

        {

            bool isActive = skillsPanel.activeSelf;

            skillsPanel.SetActive(!isActive);

            //print("skill tree Button");

        }

    }

    public void skillLevelUp(int skillIndex)

    {

        abilityBases = GameObject.FindGameObjectWithTag("UI").GetComponents<Ability2>();

        if(skills[skillIndex] !=null)

        {

            playerSkillPoint = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerSkillPoint>();

            if(playerSkillPoint.skillPoint >=1)

            {

            playerSkillPoint.skillPoint-=1;

            skills[skillIndex].SetActive(true);

            PlayerPrefs.SetInt ("skill"+skillIndex.ToString(), 1);

            //print(PlayerPrefs.SetInt ("skill"));

            //savedSkill2 = skills[skillIndex];

            //print(abilityBases.Length);

            //abilityBases[skillIndex].getSkill =1;

            }

        }

    }

    void Update ()

    {

    }

}

#### Skill2Saved

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Skill2Saved : AbilityBase {

    public GameObject Skill2;

    private LockSkill unlockNewSkill;

    // This script is about the load the skill to other scene and unlock the skill for the player

    void Start () {

        unlockNewSkill = GetComponent<LockSkill>();

        if(PlayerPrefs.GetInt("skill1")==1)

        {

            unlockNewSkill.unlockSkill\_2();

            Skill2.SetActive(true);

            //print(PlayerPrefs.GetInt ("skill"+i.ToString()));

        }

        else

        {

            Skill2.SetActive(false);

        }

    }

    // Update is called once per frame

    void Update () {

    //print(PlayerPrefs.GetInt ("skill2"());

    }

}

#### Skill3Saved

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Skill3Saved : AbilityBase {

    public GameObject Skill3;

    private LockSkill unlockNewSkill;

    // Use this for initialization

    void Start () {

        unlockNewSkill = GetComponent<LockSkill>();

        if(PlayerPrefs.GetInt("skill0")==1)

        {

            unlockNewSkill.unlockSkill\_3();

            Skill3.SetActive(true);

            //print(PlayerPrefs.GetInt ("skill"+i.ToString()));

        }

        else

        {

            Skill3.SetActive(false);

        }

    }

    // Update is called once per frame

    void Update () {

    }

}

#### LockSkill

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class LockSkill : AbilityBase {

//public class LockSkill : MonoBehaviour {

    //public AbilityBase abilityBase; print(abilityBase.lockSkill);

    //This script is locking skill

    public bool unlockSkill2 = false;

    public bool unlockSkill3 = false;

    //public bool unlockSkill = false;

    // Use this for initialization

    void Start () {

        //PlayerShooting = GameObject.FindGameObjectWithTag("Player").GetComponent<playerShooting>();

        }

    // Update is called once per frame

    void Update () {

        //print(lockSkill);

    }

    public void unlockSkill\_2()

    {

        //lockSkill = unlockSkill;

        unlockSkill2 = true;

    }

    public void unlockSkill\_3()

    {

        //lockSkill = unlockSkill;

        unlockSkill3 = true;

    }

}

### Text

#### NPCText

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class NPCText : MonoBehaviour {

    //This script is about the NPC show the text to the player

    public int PointsToGivePlayer;

    public string TextToShow;

    void OnTriggerEnter2D(Collider2D other) {

        if (other.gameObject.tag == "Player")

        {

            SpawnText();

        }

    }

    public void SpawnText()

    {

        GameObject PointsText = Instantiate(Resources.Load("Prefabs/NPCtalk")) as GameObject;

        if(PointsText.GetComponent<TextOnSpotScript>() != null) {

            var givePointsText = PointsText.GetComponent<TextOnSpotScript>();

            givePointsText.DisplayPoints = PointsToGivePlayer;

            givePointsText.DisplayText = TextToShow;

            //print("Hit number");

        }

        PointsText.transform.position = gameObject.transform.position;

    }

}

#### ScoreScipt

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class ScoreScipt : MonoBehaviour {

    // This script is about score text in the UI

    public static int scoreValue = 0;

    public Text score;

    // Use this for initialization

    void Start () {

        score = GetComponent<Text> ();

    }

    // Update is called once per frame

    void Update () {

        score.text = "Score:" + scoreValue;

    }

}

#### PlayerLevelText

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class PlayerLevelText : MonoBehaviour {

    // This script is show the level of the player text

    private int playerLv;

    private PlayerLevel playerLevelClass;

    private Text playerLevelText;

    // Use this for initialization

    void Start () {

        playerLevelText = GetComponent<Text>();

        playerLevelClass = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerLevel>();

        playerLv = playerLevelClass.playerLevel;

    }

    // Update is called once per frame

    void Update () {

        playerLv = playerLevelClass.playerLevel;

        print(playerLv);

        playerLevelText.text = "Lv:" + playerLv;

    }

}

#### PlayerHPText

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class PlayerHPText : MonoBehaviour {

    //This script is the UI Player HP Text, it show the current HP

    private int playerHP;

    private PlayerHealth playerHealth;

    private Text healthText;

    // Use this for initialization

    void Start () {

        healthText = GetComponent<Text>();

        playerHealth = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerHealth>();

        playerHP = playerHealth.health;

    }

    // Update is called once per frame

    void Update () {

        playerHP = playerHealth.health;

        //print(playerHP);

        healthText.text = "HP:" + playerHP;

    }

}

#### textOnSpotScript

using UnityEngine;

using System.Collections;

using UnityEngine.UI;

public class TextOnSpotScript : MonoBehaviour {

    // This script is about the flowing text

    public string DisplayText;

    public int DisplayPoints;

    public Text TextPrefab;

    public float Speed;

    public float DestroyAfter;

    private float Timer;

    // Use this for initialization

    void Start () {

        Timer = DestroyAfter/2;

        TextPrefab = GetComponentInChildren<Text>();

    }

    // Update is called once per frame

    void Update () {

        Timer -= Time.deltaTime;

        if(Timer < 0) {

            Destroy(gameObject);

        }

        if(DisplayPoints > 0) {

            TextPrefab.text = "+" + DisplayPoints + "!";

        } else if(DisplayText != null) {

            TextPrefab.text = DisplayText;

        }

        if(Speed > 0) {

            transform.Translate(Vector3.up \* Speed \* Time.deltaTime, Space.World);

        }

    }

}

#### FloatTextEnemy

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class FloatTextEnemy : MonoBehaviour {

    public int PointsToGivePlayer;

    public string TextToShow;

    // Use this for initialization

    void Start () {

    }

    // Update is called once per frame

    void Update () {

    }

    public void SpawnText()

    {

        GameObject PointsText = Instantiate(Resources.Load("Prefabs/TextOnSpot2")) as GameObject;

        if(PointsText.GetComponent<TextOnSpotScript>() != null) {

            var givePointsText = PointsText.GetComponent<TextOnSpotScript>();

            givePointsText.DisplayPoints = PointsToGivePlayer;

            givePointsText.DisplayText = TextToShow;

            //print("Hit number");

        }

        PointsText.transform.position = gameObject.transform.position;

    }

}

#### CoinText

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class CoinText : MonoBehaviour {

    private int newplayerCoin;

    private PlayerCoin playerCoin;

    private Text coinText;

    // Use this for initialization

    void Start () {

        coinText = GetComponent<Text>();

        playerCoin = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerCoin>();

        newplayerCoin = playerCoin.currentPlayerCoinc;

    }

    // Update is called once per frame

    void Update () {

        newplayerCoin = playerCoin.currentPlayerCoinc;

        coinText.text = "Coin:" + newplayerCoin;

    }

}

#### ClickMe

using UnityEngine;

using System.Collections;

public class ClickMe : MonoBehaviour {

    public int PointsToGivePlayer;

    public string TextToShow;

    public void SpawnText()

    {

        GameObject PointsText = Instantiate(Resources.Load("Prefabs/TextOnSpot2")) as GameObject;

        if(PointsText.GetComponent<TextOnSpotScript>() != null) {

            var givePointsText = PointsText.GetComponent<TextOnSpotScript>();

            givePointsText.DisplayPoints = PointsToGivePlayer;

            givePointsText.DisplayText = TextToShow;

            //print("Hit number");

        }

        PointsText.transform.position = gameObject.transform.position;

    }

}

#### LevelUpText

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class LevelUpText : MonoBehaviour {

    //This script is show the levelup text

    public int PointsToGivePlayer;

    public string TextToShow;

    /\* void OnTriggerEnter2D(Collider2D other) {

        if (other.gameObject.tag == "bulletEnemy")

        {

            SpawnText ();

            print("Hit number");

        }

    }\*/

    /\* public void OnMouseDown ()

    {

        SpawnText ();

        print("Hit number");

    }\*/

    public void SpawnText()

    {

        GameObject PointsText = Instantiate(Resources.Load("Prefabs/LevelUpText")) as GameObject;

        if(PointsText.GetComponent<TextOnSpotScript>() != null) {

            var givePointsText = PointsText.GetComponent<TextOnSpotScript>();

            givePointsText.DisplayPoints = PointsToGivePlayer;

            givePointsText.DisplayText = TextToShow;

            //print("Hit number");

        }

        PointsText.transform.position = gameObject.transform.position;

    }

}

### Player

#### GetPotion

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class GetPotion : MonoBehaviour {

    //This script is the player buy the potion and lose coin to get health

    //public GameObject PotionPanel;

    //private int playerHP;

    //private PlayerHealth playerHealth;

    // Use this for initialization

    void Start () {

        /\* playerHealth = GetComponent<PlayerHealth>();

        playerHP = playerHealth.health;

        print("get PlayerHealth");\*/

    }

    // Update is called once per frame

    void Update () {

    }

    void OnTriggerEnter2D(Collider2D other)

    {

        if (other.gameObject.tag == "Player")

        {

            PlayerCoin playerCoin = other.gameObject.GetComponent<PlayerCoin>();

            if(playerCoin.currentPlayerCoinc >=500)

            {

                playerCoin.currentPlayerCoinc-=500;

                print("Coin-100");

                PlayerHealth playerHealth = other.gameObject.GetComponent<PlayerHealth>();

                print("get potion");

                if(playerHealth.health < 10)

                {

                    playerHealth.health+= 3;

                }

                Destroy(gameObject);

            }

        }

    }

/\*

    public void GoPotionPanel()

    {

         if(PotionPanel != null)

        {

//pause Time

            if(Time.timeScale == 1)

            Time.timeScale = 0;

            else

            Time.timeScale = 1;

print("GoPotionPanel");

            PotionPanel.SetActive(true);

        }

    }\*/

}

#### Hitplayer

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Hitplayer : MonoBehaviour {

    // Use this for initialization

    public AudioSource playerHitSound;

    //This script is play the audio of player hit and lose HP

    void Start () {

        playerHitSound = GetComponent<AudioSource>();

    }

    // Update is called once per frame

    void Update () {

    }

void OnTriggerEnter2D(Collider2D other)

{

        if(other.gameObject.tag == "Player")

{

            PlayerHealth playerHealth = other.gameObject.GetComponent<PlayerHealth>();

            ClickMe clickMe = other.gameObject.GetComponent<ClickMe>();

            clickMe.SpawnText ();

            //print("hit get");

playerHealth.health--;

            //print("hit");

            //playerHitSound.Play();

}

}

}

#### GetCoin

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class GetCoin : MonoBehaviour {

    public AudioSource playerGetCoinSound;

    private PlayerCoin plyerClass;

    //This script is play the audio of coin

void Start () {

        playerGetCoinSound = GetComponent<AudioSource>();

        plyerClass = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerCoin>();

    }

    // Update is called once per frame

    void Update () {

    }

    void OnTriggerEnter2D(Collider2D other)

    {

        if (other.gameObject.tag == "Player")

        {

            //CoinSound();

        }

    }

    public void CoinSound()

    {

        playerGetCoinSound.Play();

        print("Sound Coin");

    }

}

#### PlayerShooting

using UnityEngine;

using System.Collections;

public class playerShooting : MonoBehaviour {

    // Use this for initialization

    public Vector3 bulletOffset = new Vector3(0, 0f, 0);

    public GameObject bulletPrefab;

    int bulletLayer;

    public AudioSource playerShootSound;

    // set the shoot system, leftkey to shoot the bulletGO

    public float fireDelay = 0.25f;

    float cooldownTimer = 0;

    void Start() {

        bulletLayer = gameObject.layer;

        playerShootSound = GetComponent<AudioSource>();

    }

    // Update is called once per frame

    void Update () {

        cooldownTimer -= Time.deltaTime;

        if( Input.GetButton("Fire1") && cooldownTimer <= 0 ) {

            // SHOOT!

            Debug.Log("Enter leftkey");

            cooldownTimer = fireDelay;

            playerShootSound.Play();

            Vector3 offset = transform.rotation \* bulletOffset;

            GameObject bulletGO = (GameObject)Instantiate(bulletPrefab, transform.position + offset, transform.rotation);

            bulletGO.layer = bulletLayer;

        }

    }

}

#### PlayerSkillPoint

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerSkillPoint : MonoBehaviour {

    // This script is about play and the skill point, the collider skill item (it is the testing item)

    public int skillPoint = 0;

    // Use this for initialization

    void Start () {

    }

    // Update is called once per frame

    void Update () {

    }

    void OnTriggerEnter2D(Collider2D other)

    {

        if (other.gameObject.tag == "Skill")

        {

            Destroy(other.gameObject);

            print("Get Coin");

            skillPoint +=1;

        }

    }

}

#### PlayerMovement

using UnityEngine;

using System.Collections;

public class playerMovement : MonoBehaviour {

// This script is about the palyer movespeed

public float defaultMoveSpeed;

    private float movex;

    private float movey;

    //Ability1

    public float moveSpeed = 5;

    // Update is called nce per frame

    void FixedUpdate () {

        movex = Input.GetAxis ("Horizontal");

        movey = Input.GetAxis ("Vertical");

        GetComponent<Rigidbody2D> ().velocity = new Vector2 (movex \* moveSpeed, movey \* moveSpeed);

        //print (moveSpeed);

    }

}

#### PlayerInTree

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerInTree : MonoBehaviour {

    public bool playerInTree;

    //This script is about the player can hide in the tree, the enemy AI need to check the playerInTree = false;

    void Start () {

    }

    // Update is called once per frame

    void Update () {

        //print(playerInTree);

    }

    void OnTriggerEnter2D(Collider2D other)

    {

        if (other.gameObject.tag == "Tree")

        {

            playerInTree = true;

        }

    }

    void OnTriggerExit2D(Collider2D other)

{

playerInTree = false;

}

}

#### PlayerLevel

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerLevel : MonoBehaviour {

    // set the player level and need exp to levelUp

    public int playerLevel = 1;

    public int playerExp = 0;

    public int LevelUp = 10;

    private int resetLevel = 0;

    private PlayerHealth playerHealth;

    private PlayerSkillPoint playerSkillPoint;

    private LevelUpText levelUpText;

    private ProgressBar progressBar;

    // Use this for initialization

    void Start () {

        playerHealth = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerHealth>();

        playerSkillPoint = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerSkillPoint>();

        progressBar = GameObject.FindGameObjectWithTag("LevelBar").GetComponent<ProgressBar>();

    }

    // Update is called once per frame

    void Update () {

       progressBar.BarValue = playerExp;

        if(playerExp >= LevelUp)

        {

            levelUpText = GameObject.FindGameObjectWithTag("Player").GetComponent<LevelUpText>();

            levelUpText.SpawnText();

            playerLevel++;

            playerSkillPoint.skillPoint +=1;

            if(playerHealth.health < 10){

            playerHealth.health +=1;

            }

            playerExp = resetLevel;

        }

    }

}

#### PlayerCoin

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerCoin : MonoBehaviour {

    //This script is the player collider the coin and add the coin

    public int currentPlayerCoinc = 0;

    //public GetCoin getCoin;

    public AudioSource playerGetCoinSound;

    // Use this for initialization

    void Start () {

        //getCoin = GameObject.FindGameObjectWithTag("Coin").GetComponent<GetCoin>();

        playerGetCoinSound = GetComponent<AudioSource>();

    }

    // Update is called once per frame

    void Update () {

    }

    // Collider to coin add 100 coin to the player and destroy object

    void OnTriggerEnter2D(Collider2D other)

    {

        if (other.gameObject.tag == "Coin")

        {

            //getCoin.CoinSound();

            playerGetCoinSound.Play();

            currentPlayerCoinc +=100;

            Destroy(other.gameObject);

            print("Get Coin");

            //currentPlayerCoinc +=100;

        }

    }

}

#### PlayerHealth

using UnityEngine;

using System.Collections;

using UnityEngine.SceneManagement;

public class PlayerHealth : MonoBehaviour {

//This script is about the player HP, it set 10 if HP 0 it show the gameover panel

    public int health = 10;

public GameObject GameOverPanel;

private ProgressBar progressBar;

    void Start () {

progressBar = GameObject.FindGameObjectWithTag("HealthBar").GetComponent<ProgressBar>();

    }

    // Update is called once per frame

    void Update () {

progressBar.BarValue = health;

    }

void OnTriggerEnter2D()

{

if (health <= 0)

{

Die();

}

}

void Die()

{

if(GameOverPanel != null)

        {

//pause Time

            /\* if(Time.timeScale == 1)

            Time.timeScale = 0;

            else

            Time.timeScale = 1;\*/

print("player die");

            //bool isActive = GameOverPanel.activeSelf;

            GameOverPanel.SetActive(true);

}

}

}

### Bullet

#### DestructEnemyButtlets

using UnityEngine;

using System.Collections;

public class DestructEnemyButtlet : MonoBehaviour {

    // This script is the enemy bullet collider wall and box

    public float timer;

    void Update () {

        timer -= Time.deltaTime;

        if(timer <= 0) {

            Destroy(gameObject);

        }

    }

    void OnTriggerEnter2D(Collider2D other) {

        if (other.gameObject.tag == "Wall")

        {

            Destroy(gameObject);

        }

        if (other.gameObject.tag == "Box")

        {

            Destroy(gameObject);

        }

    }

}

#### DestructButtet

using UnityEngine;

using System.Collections;

public class DestructButtlet : MonoBehaviour {

    // Use this for initialization

    public float timer;

    void Update () {

        timer -= Time.deltaTime;

        if(timer <= 0) {

            Destroy(gameObject);

        }

    }

/\*  void OnTriggerEnter2D(Collider2D other) {

        if (other.gameObject.tag == "Wall")

        {

            Destroy(gameObject);

        }

    } \*/

}

#### FollowMouse

using UnityEngine;

using System.Collections.Generic;

using System.Collections;

public class FollowMouse : MonoBehaviour {

    public float offset = 0.0f;

    void Start () {

    }

    // Update is called once per frame

    void Update () {

        Vector3 difference = Camera.main.ScreenToWorldPoint(Input.mousePosition) - transform.position;

        difference.Normalize();

        float rotation\_z = Mathf.Atan2(difference.y, difference.x) \* Mathf.Rad2Deg;

        transform.rotation = Quaternion.Euler(0f, 0f, rotation\_z + offset);

    }

}

#### MoveForword

using UnityEngine;

using System.Collections;

public class MoveForword : MonoBehaviour {

    public float maxSpeed;

    // Update is called once per frame

    void Update () {

        Vector3 pos = transform.position;

        Vector3 velocity = new Vector3(0, maxSpeed \* Time.deltaTime, 0);

        pos += transform.rotation \* velocity;

        transform.position = pos;

    }

    void OnTriggerEnter2D(Collider2D other)

    {

        if (other.gameObject.tag == "Enemy")

        {

            Destroy(gameObject);

        }

    }

}

#### MouseCorsor

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class MouseCorsor : MonoBehaviour {

    //This script is about the crosshair image

public Texture2D cursorTexture;

public CursorMode cursorMode = CursorMode.Auto;

public Vector2 hotSpot = Vector2.zero;

    public Texture2D enterEnemyTexture;

    // Use this for initialization

    void Start () {

        Cursor.SetCursor(cursorTexture, hotSpot, cursorMode);

    }

    // Update is called once per frame

    void Update () {

        //Cursor.SetCursor(cursorTexture, hotSpot, cursorMode);

    }

    public void changeMouseCursor()

    {

        Cursor.SetCursor(enterEnemyTexture, hotSpot, cursorMode);

    }

    public void defaultMouseCursor()

    {

        Cursor.SetCursor(cursorTexture, hotSpot, cursorMode);

    }

}

### Enemy

#### BossShooting

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class BossShooting : MonoBehaviour {

/// This script is the boss enemy shooting and AI system

public Vector3 bulletOffset = new Vector3(0, 0f, 0);

public GameObject bulletPrefab;

public GameObject targetPlayer;

public GameObject FirePoint;

    public GameObject FirePoint1;

    public GameObject FirePoint2;

    public GameObject FirePoint3;

public PlayerInTree InTree;

public AudioSource enemyShootSound;

int bulletLayer;

public float fireDelay = 4f;

    public float fireDelay2 = 2f;

    float cooldownTimer2 = 0;

float cooldownTimer = 0;

public float moveSpeed;

public float rotationSpeed;

void Start()

{

bulletLayer = gameObject.layer;

enemyShootSound = GetComponent<AudioSource>();

InTree = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerInTree>();

}

// Update is called once per frame

void Update()

{

if(InTree.playerInTree == false)

{

Vector3 vectorToTarget = targetPlayer.transform.position - transform.position;

float angle = Mathf.Atan2(vectorToTarget.y, vectorToTarget.x) \* Mathf.Rad2Deg;

Quaternion qt = Quaternion.AngleAxis(angle, Vector3.forward);

transform.rotation = Quaternion.RotateTowards(transform.rotation, qt, Time.deltaTime \* rotationSpeed);

if(Vector2.Distance(transform.position, targetPlayer.transform.position) > 13)

{

transform.position = Vector3.MoveTowards(transform.position, targetPlayer.transform.position, moveSpeed \* Time.deltaTime);

transform.position = Vector3.MoveTowards(transform.position, targetPlayer.transform.position, moveSpeed \* Time.deltaTime);

/\* Vector3 vectorToTarget = targetPlayer.transform.position - transform.position;

float angle = Mathf.Atan2(vectorToTarget.y, vectorToTarget.x) \* Mathf.Rad2Deg;

Quaternion qt = Quaternion.AngleAxis(angle, Vector3.forward);

transform.rotation = Quaternion.RotateTowards(transform.rotation, qt, Time.deltaTime \* rotationSpeed);

\*/

}

if(Vector2.Distance(transform.position, targetPlayer.transform.position) < 13)

{

/\* Vector3 vectorToTarget = targetPlayer.transform.position - transform.position;

float angle = Mathf.Atan2(vectorToTarget.y, vectorToTarget.x) \* Mathf.Rad2Deg;

Quaternion qt = Quaternion.AngleAxis(angle, Vector3.forward);

transform.rotation = Quaternion.RotateTowards(transform.rotation, qt, Time.deltaTime \* rotationSpeed);

\*/

cooldownTimer -= Time.deltaTime;

         cooldownTimer2 -= Time.deltaTime;

if (cooldownTimer <= 0)

{

cooldownTimer = fireDelay;

enemyShootSound.Play();

Vector3 offset1 = transform.rotation \* bulletOffset;

GameObject bulletGO1 = (GameObject)Instantiate(bulletPrefab, FirePoint1.transform.position + offset1, FirePoint1.transform.rotation);

bulletGO1.layer = bulletLayer;

             Vector3 offset = transform.rotation \* bulletOffset;

GameObject bulletGO = (GameObject)Instantiate(bulletPrefab, FirePoint.transform.position + offset, FirePoint.transform.rotation);

bulletGO.layer = bulletLayer;

}

if (cooldownTimer2 <= 0)

{

cooldownTimer2 = fireDelay2;

enemyShootSound.Play();

             Vector3 offset2 = transform.rotation \* bulletOffset;

GameObject bulletGO2 = (GameObject)Instantiate(bulletPrefab, FirePoint2.transform.position + offset2, FirePoint2.transform.rotation);

bulletGO2.layer = bulletLayer;

             Vector3 offset3 = transform.rotation \* bulletOffset;

GameObject bulletGO3 = (GameObject)Instantiate(bulletPrefab, FirePoint3.transform.position + offset3, FirePoint3.transform.rotation);

bulletGO3.layer = bulletLayer;

         }

}

}

}

}

#### DestroyDamageNumber

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class DestroyDamageNumber : MonoBehaviour {

    public float timeToDestroy;

    // Use this for initialization

    void Start () {

    }

    // Update is called once per frame

    void Update () {

        timeToDestroy -= Time.deltaTime;

        if(timeToDestroy <=0)

        {

            Destroy (gameObject);

        }

    }

}

#### EnemyBox

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class EnemyBox : MonoBehaviour {

    // Use this for initialization

    void Start () {

    }

    // Update is called once per frame

    void Update () {

    }

    void OnTriggerEnter2D(Collider2D other)

    {

        if (other.gameObject.tag == "bulletEnemy")

        {

            Destroy(gameObject);

        }

    }

    /\* void OnTriggerExit2D(Collider2D other)

{

}\*/

}

#### DamageHandler

using UnityEngine;

using System.Collections;

public class DamageHandler : MonoBehaviour {

    // set the enemy collider the bullet, get the player exp and score

    public int health;

    public float invulnPeriod = 0;

    public GameObject DestoryObject;

    public GameObject Explosion;

    //public float ExplosionTimer = 1;

    private PlayerLevel playerLevelClass;

    private EnemyCount enemyCount;

    float invulnTimer = 0;

    int correctLayer;

    SpriteRenderer spriteRend;

    void Start() {

        correctLayer = gameObject.layer;

        // NOTE! This only get the renderer on the parent object.

        // In other words, it doesn't work for children. I.E. "enemy01"

        playerLevelClass = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerLevel>();

        enemyCount = GameObject.FindGameObjectWithTag("LevelBlock").GetComponent<EnemyCount>();

        spriteRend = GetComponent<SpriteRenderer>();

        if(spriteRend == null) {

            spriteRend = transform.GetComponentInChildren<SpriteRenderer>();

            if(spriteRend==null) {

                Debug.LogError("Object '"+gameObject.name+"' has no sprite renderer.");

            }

        }

    }

    void OnTriggerEnter2D(Collider2D other) {

        if (other.gameObject.tag == "bullet2")

        {

            //print("Enemy Hit");

            health-=1;

            FloatTextEnemy floatTextEnemy = gameObject.GetComponent<FloatTextEnemy>();

            floatTextEnemy.SpawnText ();

            //print(health);

        }

        if(invulnPeriod > 0) {

            invulnTimer = invulnPeriod;

            gameObject.layer = 10;

        }

    }

    void Update() {

        if(health <= 0) {

            //print("<=0");

            // player get score

            //ExplosionTimer -= Time.deltaTime;

            enemyCount.enemyKilled +=1;

            ScoreScipt.scoreValue += 100;

            playerLevelClass.playerExp +=2;

            print(playerLevelClass.playerExp);

            Instantiate(Explosion, transform.position, transform.rotation);

            Die();

        }

        if(invulnTimer > 0) {

            invulnTimer -= Time.deltaTime;

            if(invulnTimer <= 0) {

                gameObject.layer = correctLayer;

                if(spriteRend != null) {

                    spriteRend.enabled = true;

                }

            }

            else {

                if(spriteRend != null) {

                    spriteRend.enabled = !spriteRend.enabled;

                }

            }

        }

    }

    void Die()

    {

            Destroy(DestoryObject);

        //Destroy(DestoryObject);

    }

}

#### Enemy

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Enemy : MonoBehaviour {

    private MouseCorsor mouse;

    // Use this for initialization

    //this script is the change the crosshair

    void Start () {

        mouse = GameObject.FindGameObjectWithTag("Mouse").GetComponent<MouseCorsor>();

    }

    // Update is called once per frame

    void Update () {

    }

    void OnMouseEnter()

    {

        mouse.changeMouseCursor();

    }

    void OnMouseExit()

    {

        mouse.defaultMouseCursor();

    }

}

#### EnemyCount

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class EnemyCount : MonoBehaviour {

public int enemyKilled = 0;

public int endLevel;

public GameObject RedBlock;

    // Use this for initialization

    void Start () {

    }

    // Update is called once per frame

    void Update () {

        if(enemyKilled >= endLevel)

        {

            RedBlock.SetActive(false);

        }

    }

}

#### EnemyShooting

using UnityEngine;

using System.Collections;

using System.Collections.Generic;

public class enemyShooting : MonoBehaviour

{

//This script is the enemy shooting and AI system

public Vector3 bulletOffset = new Vector3(0, 0f, 0);

public GameObject bulletPrefab;

public GameObject targetPlayer;

public GameObject FirePoint;

public AudioSource enemyShootSound;

public PlayerInTree InTree;

int bulletLayer;

public float fireDelay = 4f;

float cooldownTimer = 0;

public float moveSpeed;

public float rotationSpeed;

void Start()

{

bulletLayer = gameObject.layer;

InTree = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerInTree>();

enemyShootSound = GetComponent<AudioSource>();

}

// Update is called once per frame

void Update()

{

if(InTree.playerInTree == false)

{

// https://www.youtube.com/watch?v=rhoQd6IAtDo

//discovery the player

if(Vector2.Distance(transform.position, targetPlayer.transform.position) <17)

{

if(Vector2.Distance(transform.position, targetPlayer.transform.position) > 7)

{

transform.position = Vector3.MoveTowards(transform.position, targetPlayer.transform.position, moveSpeed \* Time.deltaTime);

transform.position = Vector3.MoveTowards(transform.position, targetPlayer.transform.position, moveSpeed \* Time.deltaTime);

Vector3 vectorToTarget = targetPlayer.transform.position - transform.position;

float angle = Mathf.Atan2(vectorToTarget.y, vectorToTarget.x) \* Mathf.Rad2Deg;

Quaternion qt = Quaternion.AngleAxis(angle, Vector3.forward);

transform.rotation = Quaternion.RotateTowards(transform.rotation, qt, Time.deltaTime \* rotationSpeed);

}

if(Vector2.Distance(transform.position, targetPlayer.transform.position) < 7)

{

Vector3 vectorToTarget = targetPlayer.transform.position - transform.position;

float angle = Mathf.Atan2(vectorToTarget.y, vectorToTarget.x) \* Mathf.Rad2Deg;

Quaternion qt = Quaternion.AngleAxis(angle, Vector3.forward);

transform.rotation = Quaternion.RotateTowards(transform.rotation, qt, Time.deltaTime \* rotationSpeed);

cooldownTimer -= Time.deltaTime;

if (cooldownTimer <= 0)

{

cooldownTimer = fireDelay;

enemyShootSound.Play();

Vector3 offset = transform.rotation \* bulletOffset;

GameObject bulletGO = (GameObject)Instantiate(bulletPrefab, FirePoint.transform.position + offset, FirePoint.transform.rotation);

bulletGO.layer = bulletLayer;

}

}

}

}

}

}

#### ExplosionTimer

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class ExplosionTimer : MonoBehaviour {

    //This script is the how long explosion

    public float Timer = 1;

    // Use this for initialization

    void Start ()

    {

    }

    // Update is called once per frame

    void Update ()

    {

        Timer -= Time.deltaTime;

        if(Timer < 0)

        {

            Destroy(gameObject);

        }

    }

}

#### enemyBulletMoveForword

using UnityEngine;

using System.Collections;

public class enemyBulletMoveForword : MonoBehaviour {

    public float maxSpeed;

    // / This script is the enemy bullet move and collider player will destroy

    void Update () {

        Vector3 pos = transform.position;

        Vector3 velocity = new Vector3(0, maxSpeed \* Time.deltaTime, 0);

        pos += transform.rotation \* velocity;

        transform.position = pos;

    }

    void OnTriggerEnter2D(Collider2D other)

    {

        if (other.gameObject.tag == "Player")

        {

            Destroy(gameObject);

        }

    }

}

#### EnemyShortGunShooting

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class EnemyShortGunShooting : MonoBehaviour {

    //This script is the shortgun enemy shooting and AI system

public Vector3 bulletOffset = new Vector3(0, 0f, 0);

public GameObject bulletPrefab;

public GameObject targetPlayer;

public GameObject FirePoint1;

    public GameObject FirePoint2;

    public GameObject FirePoint3;

    public GameObject FirePoint4;

    public GameObject FirePoint5;

    public GameObject FirePoint6;

public AudioSource enemyShootSound;

public PlayerInTree InTree;

int bulletLayer;

public float fireDelay = 4f;

float cooldownTimer = 0;

public float moveSpeed;

public float rotationSpeed;

void Start()

{

bulletLayer = gameObject.layer;

enemyShootSound = GetComponent<AudioSource>();

InTree = GameObject.FindGameObjectWithTag("Player").GetComponent<PlayerInTree>();

}

// Update is called once per frame

void Update()

{

// https://www.youtube.com/watch?v=rhoQd6IAtDo

//discovery the player

if(InTree.playerInTree == false)

{

if(Vector2.Distance(transform.position, targetPlayer.transform.position) <17)

{

if(Vector2.Distance(transform.position, targetPlayer.transform.position) > 10)

{

transform.position = Vector3.MoveTowards(transform.position, targetPlayer.transform.position, moveSpeed \* Time.deltaTime);

transform.position = Vector3.MoveTowards(transform.position, targetPlayer.transform.position, moveSpeed \* Time.deltaTime);

Vector3 vectorToTarget = targetPlayer.transform.position - transform.position;

float angle = Mathf.Atan2(vectorToTarget.y, vectorToTarget.x) \* Mathf.Rad2Deg;

Quaternion qt = Quaternion.AngleAxis(angle, Vector3.forward);

transform.rotation = Quaternion.RotateTowards(transform.rotation, qt, Time.deltaTime \* rotationSpeed);

}

if(Vector2.Distance(transform.position, targetPlayer.transform.position) <10)

{

Vector3 vectorToTarget = targetPlayer.transform.position - transform.position;

float angle = Mathf.Atan2(vectorToTarget.y, vectorToTarget.x) \* Mathf.Rad2Deg;

Quaternion qt = Quaternion.AngleAxis(angle, Vector3.forward);

transform.rotation = Quaternion.RotateTowards(transform.rotation, qt, Time.deltaTime \* rotationSpeed);

cooldownTimer -= Time.deltaTime;

if (cooldownTimer <= 0)

{

cooldownTimer = fireDelay;

enemyShootSound.Play();

                Vector3 offset2 = transform.rotation \* bulletOffset;

     GameObject bulletGO2 = (GameObject)Instantiate(bulletPrefab, FirePoint2.transform.position + offset2, FirePoint2.transform.rotation);

    bulletGO2.layer = bulletLayer;

                Vector3 offset1 = transform.rotation \* bulletOffset;

    GameObject bulletGO1 = (GameObject)Instantiate(bulletPrefab, FirePoint1.transform.position + offset1, FirePoint1.transform.rotation);

    bulletGO1.layer = bulletLayer;

                Vector3 offset3 = transform.rotation \* bulletOffset;

    GameObject bulletGO3 = (GameObject)Instantiate(bulletPrefab, FirePoint3.transform.position + offset3, FirePoint3.transform.rotation);

    bulletGO3.layer = bulletLayer;

                Vector3 offset4 = transform.rotation \* bulletOffset;

    GameObject bulletGO4 = (GameObject)Instantiate(bulletPrefab, FirePoint4.transform.position + offset4, FirePoint4.transform.rotation);

    bulletGO4.layer = bulletLayer;

                Vector3 offset5 = transform.rotation \* bulletOffset;

    GameObject bulletGO5 = (GameObject)Instantiate(bulletPrefab, FirePoint5.transform.position + offset5, FirePoint5.transform.rotation);

    bulletGO5.layer = bulletLayer;

                Vector3 offset6 = transform.rotation \* bulletOffset;

    GameObject bulletGO6 = (GameObject)Instantiate(bulletPrefab, FirePoint6.transform.position + offset6, FirePoint6.transform.rotation);

    bulletGO6.layer = bulletLayer;

                }

}

}

}

}

}

### PauseScript

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class pauseScript : MonoBehaviour {

    // Use this for initialization

    void Start () {

    }

    // Update is called once per frame

    void Update () {

        if (Input.GetKeyDown (KeyCode.Space))

        {

            if(Time.timeScale == 1)

            Time.timeScale = 0;

            else

            Time.timeScale = 1;

        }

    }

}

### WalkRoad

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class WalkRoad : MonoBehaviour {

    public bool playerOnRoad;

    private playerMovement PlayerMovement;

    // This script is the player on the road can move faster.

    void Start () {

    }

    // Update is called once per frame

    void Update () {

    }

    void OnTriggerEnter2D(Collider2D other)

    {

        if (other.gameObject.tag == "Road")

        {

            PlayerMovement = GetComponent<playerMovement>();

            PlayerMovement.defaultMoveSpeed =6;

            playerOnRoad = true;

            print(PlayerMovement.moveSpeed);

        }

    }

    void OnTriggerExit2D(Collider2D other)

    {

        if (other.gameObject.tag == "Road")

        {

            PlayerMovement.defaultMoveSpeed =3;

            playerOnRoad = false;

            print(PlayerMovement.moveSpeed);

        }

    }

}