**Kurdistan Region**

**Salahaddin University Erbil**

**College of Science**

**Computer Science & IT Department**

# SWAT SHOOTER GAME

A Project Submitted to the Computer Science & IT Department
University of Salahaddin-Erbil

In the Partial Fulfillment of the Requirement for the Degree of Bachelor of Science in Computer Science & IT

**Prepare By:**

**Safa Muhamad**

**Araz Ibrahim**

**Mohamad Tariq**

**Supervisor:**

**M. Sajda Hadi Baker**

**(2020 - 2021)**

## Dedicated to

We would like to firstly thank God, Our parents who have learned us the way of live, brothers, sisters. And our supervisor M.Sajda with all other teachers and our dear friends that help us in preparing this project, and those who want to learn.

## Certification

I certify that the project titled (Swat Shooter Game) was done under my supervision at the Computer Science and IT Department, College of Science - Salahddin University -Erbil. In the partial fulfillment of the requirement for the degree of Bachelor of computer Science and IT.

Supervisor Signature :

Name: M.Sajda Bakr

Date: 25 / 4 / 2021

# TABLE OF CONTENTS

# LIST OF GRAPHICS AND SHAPES

# ABSTRACT

Video games and other new media artifacts are an important part of our cultural and economic landscape and collecting institutions have the responsibility to collect and preserve these materials for future access.

In this research, we have created and developed a game which we named “Swat Shooter” by using Unity engine program which is a standard application for creating games and by default it supports C# as a main programming language for developing games. Adobe Photoshop and Adobe Illustrator for designing a game user interface.

The game that we have create is working on the android system and the all devices that work on this operating system and in the future we have plan to develop the game for the IOS system and the personal computer.

And the game consist three levels and each level have it is own number of zombies that level name are (barzan map , sangasar map , maxmor map).

# Chapter One

## 1.1 Introduction of Game Development

Game development is the art of game making and describes the design, development and release of the game.

## 1.2 Introduction of Game Unity

Let's go over the steps needed for unity to work in your machine. We will then look at creating workflow to organize your files and recommended steps to avoid further problems in growth. Unity is a Cross Platform game engine developed by Unity Technologies. It first appeared and was released in June 2005 at the Apple Inc. International Conference of Engineers as a special game engine for Mac OS X Since 2018, the engine has been expanded to support more than +25 platforms.

## 1.3 Aim & Objective

Like any other game our game is for fun enjoyment and makes the player lives for a while in the digital world. This year we have seen that there is no other game development project so we have decided to make this project necessary so that we can develop it in the future. It is mainly used to improve video games and simulation of computers, consoles and mobile devices. Unity is an all-purpose gaming engine that supports (2D) and (3D) graphics.

## 1.4 Deliver incredible possibilities

Build once, deploy anywhere to reach the largest possible audience across 25+ leading platforms and technologies.

Figure 1.1 platforms

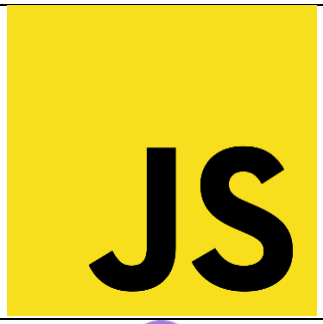
## 1.5 How to code a Unity Game?

Unity implements the mono compiler.
Script can be written in scripts:

- JavaScript
- C #

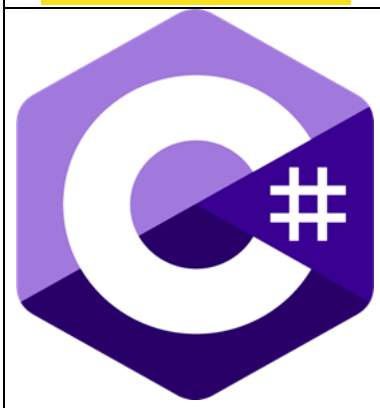
Moreover, scripts can be used. Create a graphical effect to control Physical behavior of objects.

Also run a compatible AI system For the characters in the game.



```
var explosion : Transform;

function OnCollisionEnter()
{
Destroy(gameObject);
}
```



```
using UnityEngine;
using System.Collections;
public class Example : MonoBehaviour
{
public Transform explosion;

 void OnCollisionEnter()
        {
        Destroy(gameObject);
        }
}
```

## 1.6 Scripts

Scripts contain four main types of items: variables, functions, statistics, and to comment. The variable captures values that can be anything from numbers to text.

Tasks do something, usually with diversity and equality. Comments are ignored when files for code is generated, allowing the user to make notes about what the code is or should be temporarily creating or disabling code.

When a user creates a new script with C#, a new script file will be created in local assets> Script which is a folder that the user can call. If the user double-clicks this file, then a new text will be opened with MonoDevelop or Visual Studio as new C # editing file. MonoDevelop or Visual Studio adds the required files to the title and sections automatically, as shown in Figure 1.2 .

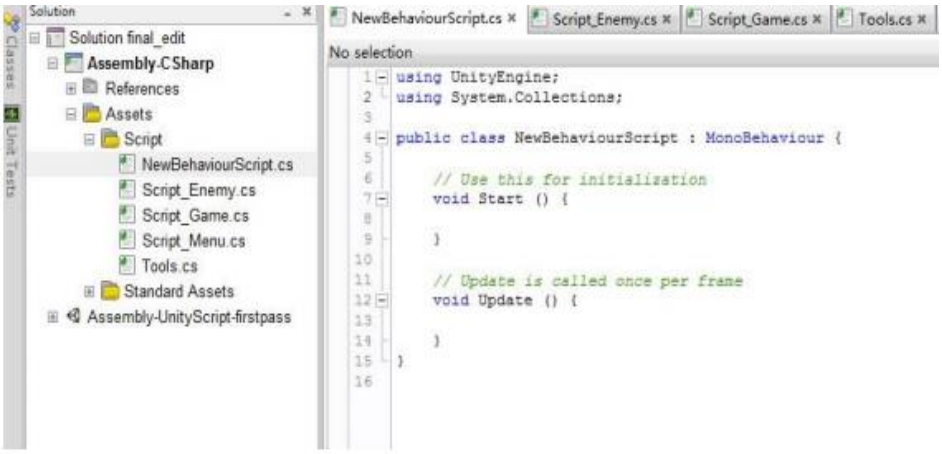



Figure 1.2 the new script

A function called Update is one of the most important functions in any game engine.

This is one of the tasks of the program to be tested in at least a whole time during operation to see if anything needs to be done. For example, it can use animation when encountering a specific situation by testing.

The written system is based on Mono and Visual Studio an open version of .NET source. Like .NET,

Mono and Visual Studio supports multiple programming languages, but Unity only supports C# and JavaScript. Each language has its advantages and disadvantages. C# has the advantage of having official commentary and many textbooks and online tutorials, are the closest to the main language in. and open source software. But C# is the most preferred course of action. In this game project, the text language was C#. Figure 1.3 same example of scripts.

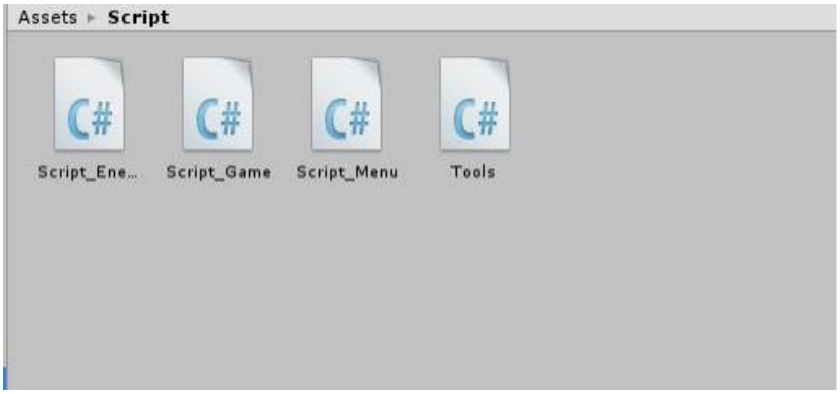



Figure 1.3 same example of scripts

Scenes viewing is a playful place, because it is a visual space, and it supports very simple builders. All views and windows plays the role of tools that can decorate the game. Parameters for everything it should be properly placed. Game launcher is a script. Always successful the script must be bound with an object or prefab, in order for the object to work.

## 1.7 C# or JavaScript?

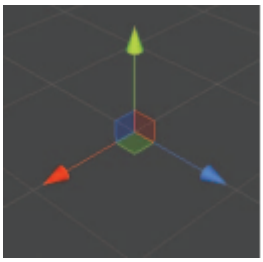
C# because:

- Pre You can use any pre-compiled dll.
- There are many libraries available in C#.
- Build short construction time.
- Editors (Visual Studio or Mono) can provide good advice with C#. Other features of VS such as creating graphs from code, finding references, etc. do not work with JavaScript.
- Examples Many examples, models, and plugins use C#.
- C# more readable and manageable with code C#.
- New All new Unity Framework documentation and tutorials are in C#, improved overall support from MSDN and core library docs, better library support. Used outside of unity.

## Transform Tools in Unity

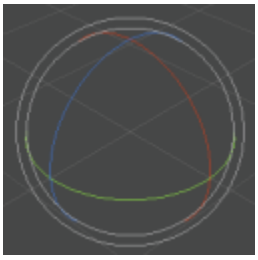
In order to put down our Game Objects, we need to know how to deliver them all around in unity. We need to be able to place, move, and measure our belongings and location where we need them. We also need to know the main differences between performance in 2D and 3D and how we can get deeper into our game.

- Translate:



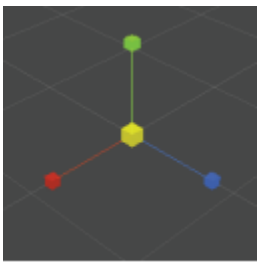
You will make the most of the translation tool when setting up your Game Object. In the left side figure you can see the X, Y, and Z-axes as the Scene Gizmo has. In fact, in 3D mode, you will see that Translate Gizmo and Scene Gizmo are the same. This for in the space of the earth we shall walk with these axes. The Translate button can be found next to the hand tool in the Transform toolbar. See in the left side figure if you do not remember this. You can also access the Translate tool with by tapping the W hotkey on the keyboard. Just click left and drag on one of the colored arrows, and your item will move on that axis. Navigating to the arrow points will move the object to good correction, and opposite opposition.

- Rotate:



Touching the E key will bring up the rotate tool. This will allow you to rotate the object near its surrounding point. Colored circles also show axes, but something will do rotate the axis. This tool is widely used to set an angle.in the left side figure shows the Rotate Gizmo.

- Scale:



The final tool is the Scale tool (In the left side figure). Measurement means to increase or decrease in size something in terms of its actual size. Using axis handles will measure the object only in that one axis. This can be used if you want to adjust the appearance of something, such as making a cube rectangular or a circle.

## 1.8 2D or 3D Projects

### Full 3D: -

- 3D games often use three-dimensional geometry, with Materials and drawings.
- Games 3D games tend to give Scene using perspective, therefore items appear large on the screen.

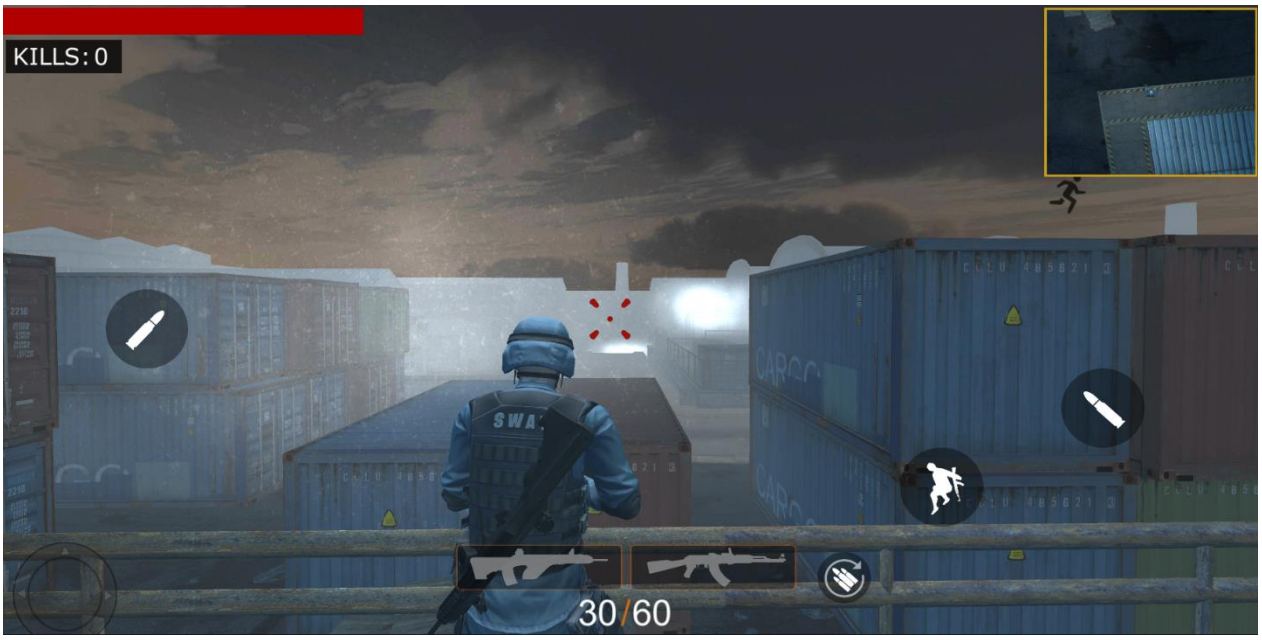



Figure 1.4 Full 3D Game

### Orthographic 3D:-

- Sometimes games use 3D geometry, but use orthographic camera instead of viewing.

- The standard procedure used in the games you give bird view of the action, and sometimes called "2.5D".

Figure 1.5 Orthographic 3D

## • Full 2D :-

- Games Many 2D games use flat graphics, sometimes called sprites, with no three-dimensional geometry at all.

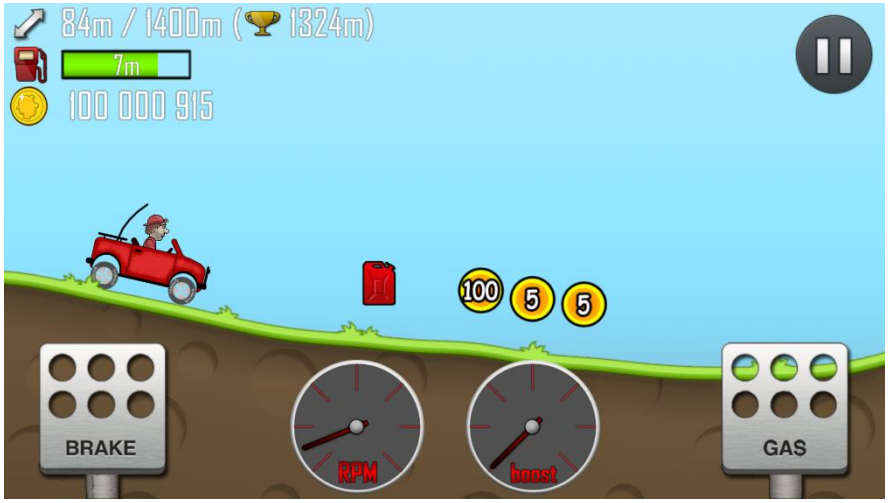



Figure 1.6 Full 2D

# Chapter Tow

## 2.1 What is Adobe Illustrator

Adobe Illustrator is a software program for creating graphics, graphics, and graphics using a Windows or MacOS computer. Illustrator was first released in 1987 and continues to be updated from time to time, and is now included as part of the Adobe Creative Cloud. Illustrator is widely used by artists, web designers, visual artists, and professional artists around the world to create high-quality art work. Illustrator includes many drawing tools that can reduce image creation time.

## 2.2 Adobe Illustrator History

Version 1 of Illustrator was first released in 1987 by Apple Macintosh. At the time, Adobe was focused on font building and providing computer-assisted language for office printers, known as PostScript. Illustrator also supported efforts to improve Adobe font and served as a compatible product for Photoshop, which Adobe did not start developing but distributed and purchased. The original Illustrator version did not have a preview mode, and users had to open a second window to preview their work.

## 2.3 What is Photoshop?

Photoshop is a photo editing and raster graphic design software which let you to users to create, edit, and manipulate various graphics as well as digital art. It also let you to create and edit raster images with multiple layers and import the images in various file formats. Photoshop is improved by Adobe Systems for both Windows and MacOS.

## 2.4 Adobe Photoshop History

Adobe Photoshop Version History Photoshop was created in 1988 by Thomas and John Knoll and the official distribution license of the program was owned by Adobe Systems. Then, so many Photoshop versions have been released.

## 2.5 Aims to Use That Applications

This program used to design something  just like the interface of the game, and it was very important for the design and the creation of the logo and the icon of the game

Also adobe illustrator was very important for the creation of the buttons that we have in the interface like ( start , help , about game , select level , replay , menu , etc ).

## 2.6 PlayMaker

PlayMaker is the Unity menu below that lets you create games without coding.

Created by Hutong Games, PlayMaker uses state-of-the-art machine tools (FSM) to add physics, graphics, interactive features and flexibility to the scene.

Their developers have already created scripts for you, which can greatly reduce your game development time.

## 2.7 The Playmaker Difference

- Easy to Understand

Some visual typing tools simply display Unity API (written for programmers) and C# language (editing) as thousands of nodes. If you already have the code you can create what you need, but a great learning curve for non-programmers!

PlayMaker takes a high-level approach, offering an intuitive structure with American, Actions and Events to quickly build character. This actually empowers non-editors and program editors to create much faster and more efficient.

- Production Ready

PlayMaker has been used in a wide range of products posted including Hearthstone, INSIDE, Hollow Knight, First Tree, Dreamfall Chapters, Firewatch and more. We have worked with AAA studios for solo indie developers to ensure that PlayMaker meets its needs across all platforms.

- Performance

Some written writing solutions use visitors or telephones to show all the same categories in the graph. Even simple behavior requires a lot of space. This can quickly add to the hard work!

PlayMaker Actions are C# text just like any other text in your project. Calling them is quick and effective. You also need a few actions to perform the same task as multiple nodes in other text editing tools.

- Ecosystem

PlayMaker has an active community that works with managers and power users online to answer your questions. Ecosystem Browser provides easy access to a growing collection of custom actions, tutorials, samples, templates etc.

## 2.8 Playmaker Interface

It's time to dump her and move on. In the main menu  go to >> PlayMaker | PlayMaker Editor. A welcome window with different Playmaker options and a PlayMaker panel should appear. Remove the welcome window, and paste the playMaker panel into the same Editor area where you have your Console panel as shown in the figure 2.7


Figure 2.1 PlayMaker Interface

This is where you will organize your finite state machines and create state nodes and transitions between them, which we will discuss in more detail in the next

chapter. In the meantime, follow the instructions provided in the lower-right corner of the FSM view:

1. Select the Wall game item in the Hierarchy panel or in the scene view by clicking on it.

2. Right-click anywhere on FSM view and select Insert FSM from the context menu that appears to add the finite state machine to the Wall game item. When you do that, notice the red icon that appears next to the Wall in the Governance panel. This icon means that the item has a Playmaker FSM component. Also note the Playmaker FSM section from the Inspector panel.

3. You can now trick FSM by assigning various options to Wall. Read all the tips shown on the playMaker panel carefully.

To the right of the play Maker panel, there are tabs that show and allow you to change various types of information, including FSM as a whole, and its scenarios, events, and variations. Click on each tab and check out the gray rectangular tips that describe the content of each tab. When you're done with the tips, you can press the Hint toggle button at the bottom of panel to disable them. You can also do this by pressing F1. Before you do that, click the favorite button next to it and read the tips there.

Under FSM view you should see controls such as pausing / playing in the toolbar. These are actually the same buttons, and they just exist to make it much easier for you to control the game when working with Playmaker.

The second panel most closely related to the Playable Player is Action Browser. You can open it by selecting PlayMaker | Editor's Window | Action Browser from the main menu. Part of the Action panel is shown in the figure 2.8

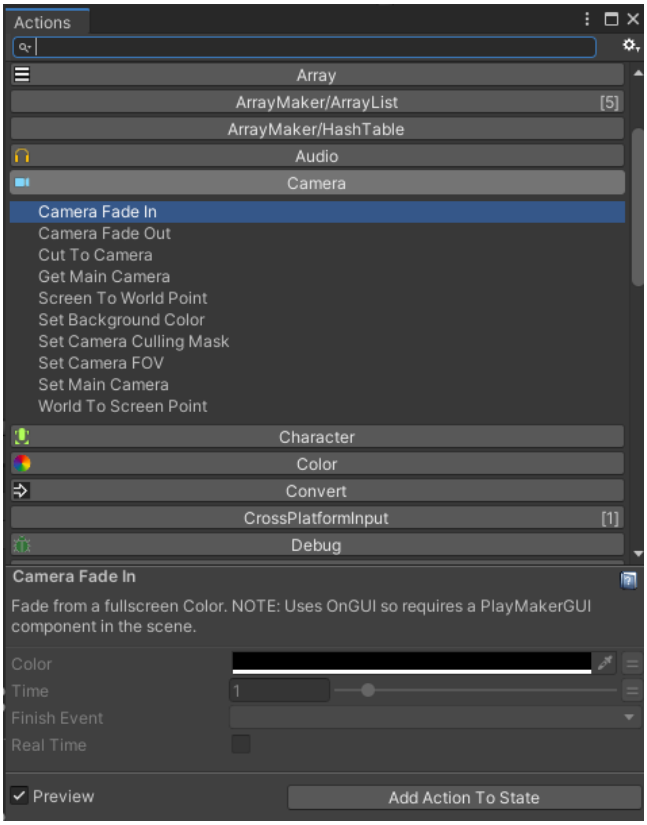



Figure 2.2 Action Browser

Drag its title to the right of the Editor until it enters the same area of the screen as the Inspector. This panel shows the categories of actions you have in your Playmaker library. Clicking on one of the categories, for example, Camera, will display actions. There is also a search bar near the up of the panel that let you to access the actions you need quickly. If you select an action, for example, Camera | Screen To World Point, a preview of its borders appears at the bottom of the panel. Indicates how the selected action will look in the Kingdom tab of the play Maker panel. If you have a Playmaker enabled item and have status in the selected FSM view, you can add action to it by clicking the Add Action To State button in the lower-right corner of the Actions panel.

You can press the play button (or use the shortcut + P or Ctrl + P Windows command) to make sure there are no errors and everything works fine. When you're done, press play again and save location.

# Chapter Three

## 3.1 Introduction to the Third-Person-Shooter (TPS)

Third-person shooter (TPS) video games, at its core, are console games in the shooter genre where the camera during gameplay is completely focused in a third-person viewpoint. Third-person is a viewpoint where the player controls a character totally.

## 3.2 Swat Shooter

The game that we have created it is name is Swat Shooter that is a third person console video game , ad right now we will introduce the first thing that the user can see when enter the game form mobile in the figure 3.1


Figure 3.1 Main Menu

- **HELP:**
  This tab is help the user to guide to how play the game and introduce the levels.
- **ABOUT GAME:**
  This tab have all the information about the game creation.
- **HOME:**
  This tab take the user to the home to start the game.
- **START:**
  This tab is take the user to select the level to start the game.

And after the user click the start button the figure 3.2 is shown

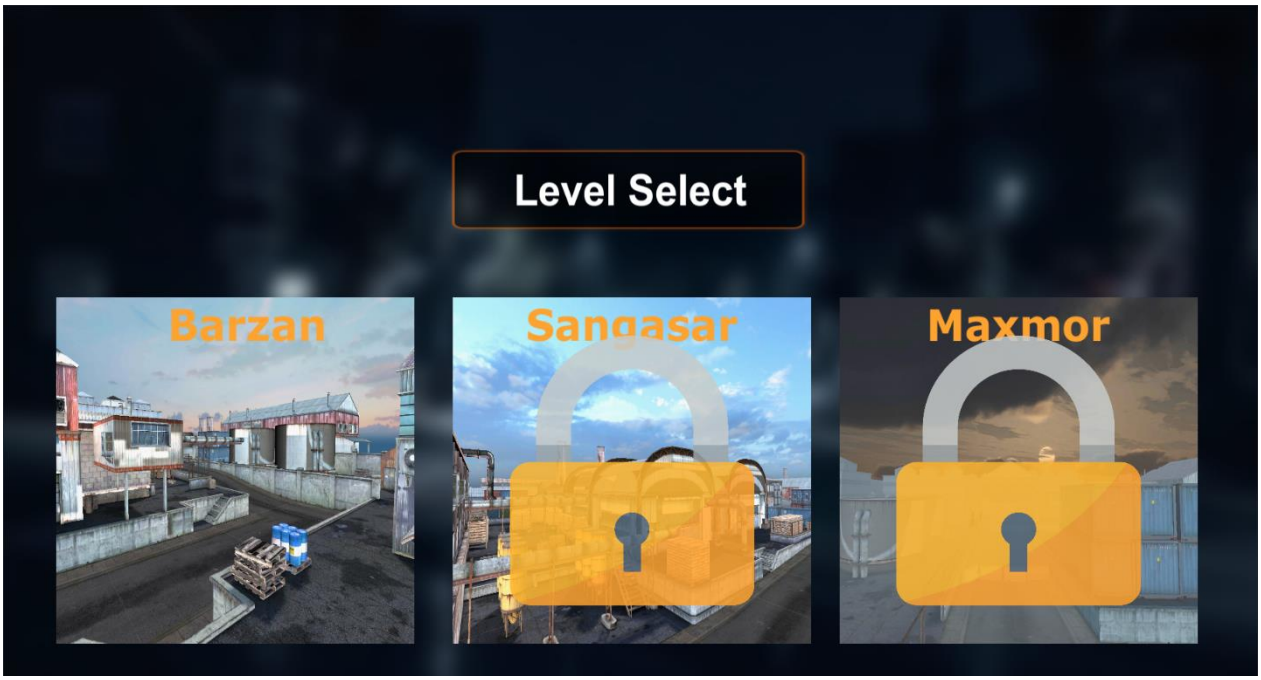



Figure 3.2 level select

In this figure we have three level that the user only start with the first level when he start the game, the user must finish the first level to unlock the next level.

- Barzan map is the start level when the user enter the game.
- Sangasar map is the second level and it is a little harder than the first level.
- Maxmor map is the final and the bigger and the harder level among them.

## 3.3 swat shooter game interface

This figure below 3.3 is show the user when play the game in the first level


Figure 3.3 Game Interface

The left side of the interface show so many things in like the health bar that consider the life span of the user to stay alive, and another thing it is count the number of the kills that user makes.

The top right of the interface show a live mini map of the level.

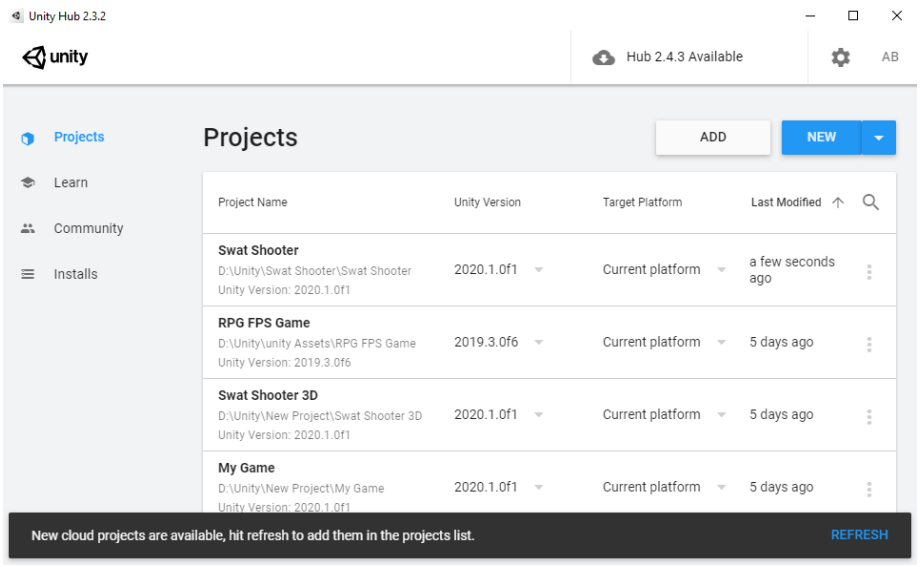
The bottom right of the interface show the bullet to shoot the zombies and jumping of the user and refill the ammo.

The bottom middle of the interface show the weapons and the user can choose among two weapon and carry another weapon if there existed and the number of the ammo.

When the user complete a level the figure 3.4 is shown

Figure 3.4 level up

- **Menu:** This button is to open the main menu.
- **Replay:** This button is to play the game again.
- **Next level:** This button is to move the user to the next level.




When the user failed and got killed by the zombies, the figure 3.5 is shown


Figure 3.5 failed

# Chapter Four

## 4.1 Project Creation

The first steps to click the New button as shown figure 4.1, then Select the template for our game then write the name and select the location as shown in figure 4.2,then click the CREAT.

Figure 4.1 Creating New Game

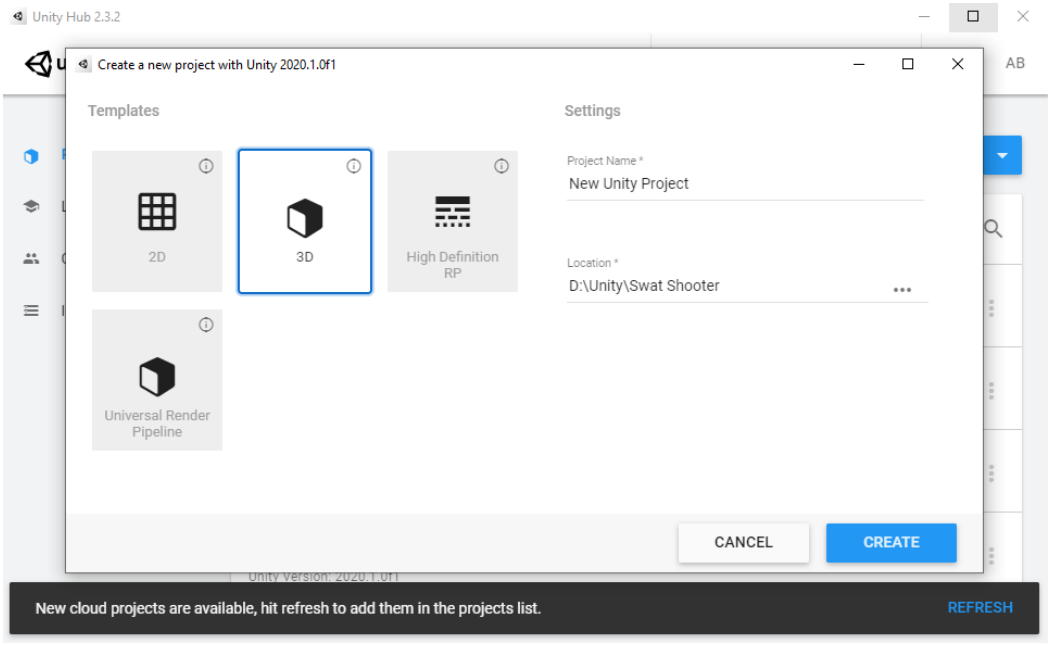



Figure 4.2 Select and Create Project

## 4.2 PlayMaker's FSM Codes

**Movement system:** it is the move, jump and the vault of the player.

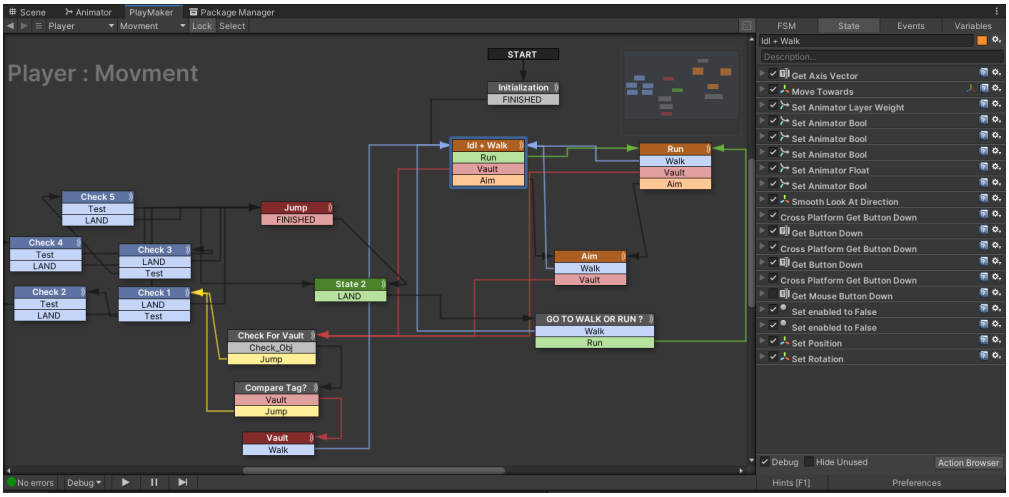



Figure 4.3 movement

**Shooting system:** it is the shoot and firing the bullets by the player to kill the zombie's enemy.

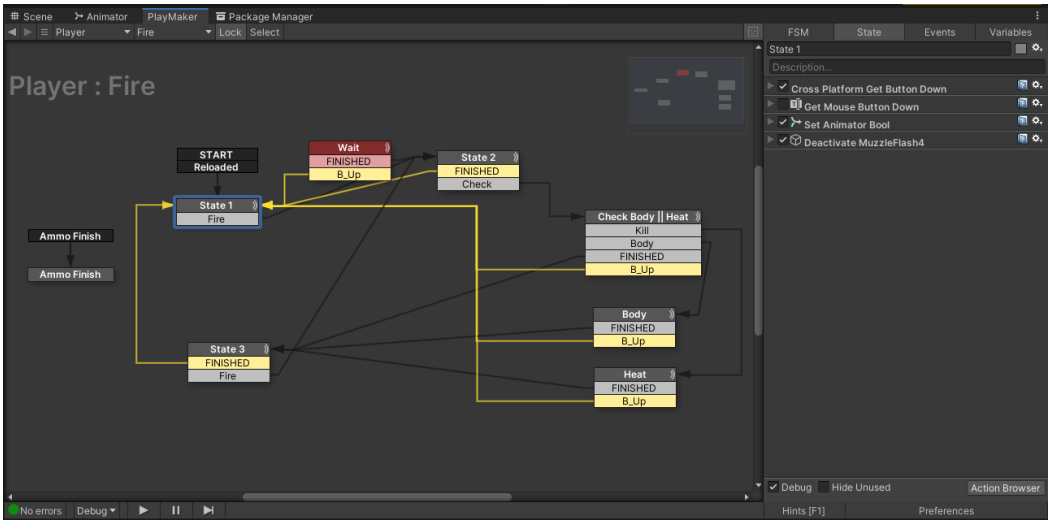



Figure 4.4 shooting system

**Pick up system**: it is the system that allows the player to pick up and put the weapons down.

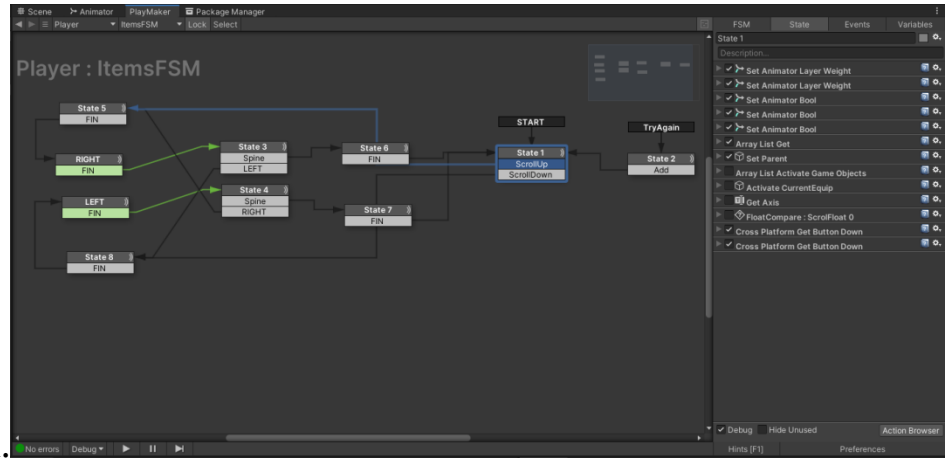



Figure 4.5 pick up

**Bullet system:** it allows the player to know the number of the bullet and reload the weapons that the players have.

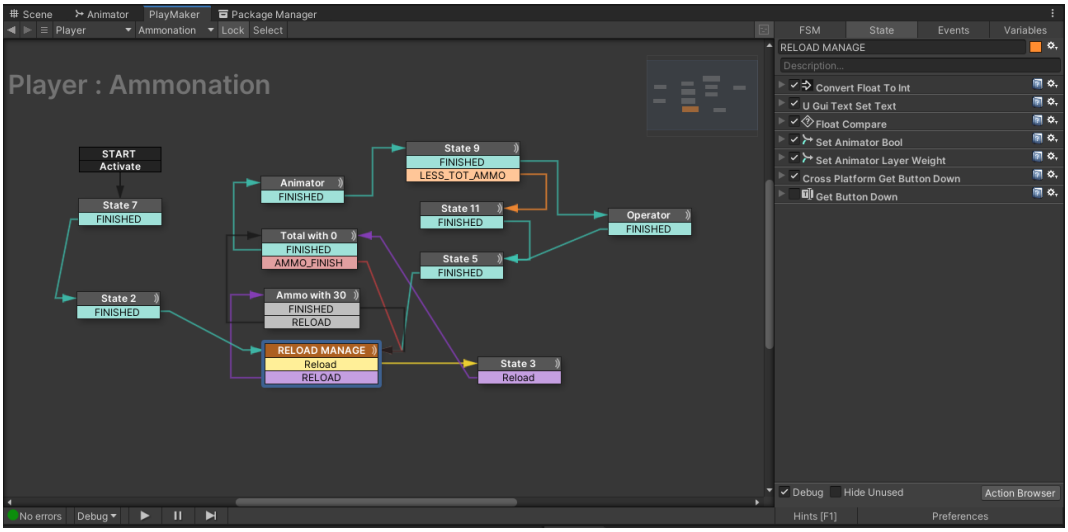



Figure 4.6 bullet system

**Player health:** it allows the player to know his life span and how many time the player have died.

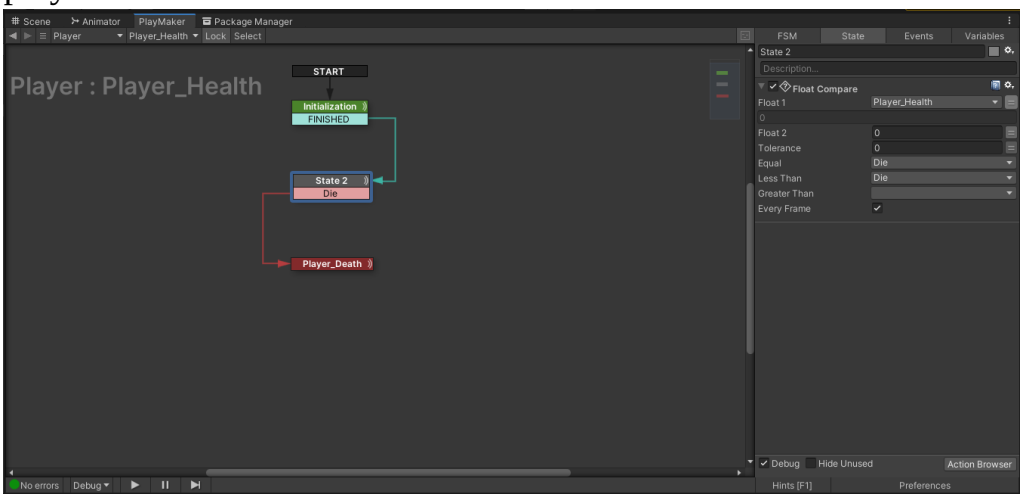



Figure 4.7 player health

## 4.3 C# Codes

In this game we used C# code with playmaker frame to create the finite state machines, the UNITY software depends on the C#.

one of the most important things that we use C# codes to stop the camera go through the walls in the game to not see the other side of the wall and when the camera beat the wall the camera make a zoom.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine; //Unity Library
public class CameraCollision : MonoBehaviour {
        public float minDistance = 1.0f;
        public float maxDistance = 4.0f;
        public float smooth = 10.0f;
        Vector3 dollyDirection;
        public Vector3 dollyDirAdjusted;
        public float _distance;
   public float dis_ray;

        // Use this for initialization
        void Awake () {
                dollyDirection = transform.localPosition.normalized;
                _distance = transform.localPosition.magnitude;
        }

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

                Vector3 desiredCameraPos = transform.parent.TransformPoint (dollyDirection *
maxDistance);
                RaycastHit hit;

                if (Physics.Linecast (transform.parent.position, desiredCameraPos, out hit)) {
                        _distance = Mathf.Clamp ((hit.distance * dis_ray), minDistance,
maxDistance);

                                } else {
                                _distance = maxDistance;
```

```
                }

                transform.localPosition = Vector3.Lerp (transform.localPosition,
dollyDirection * _distance, Time.deltaTime * smooth);
        }
}
```

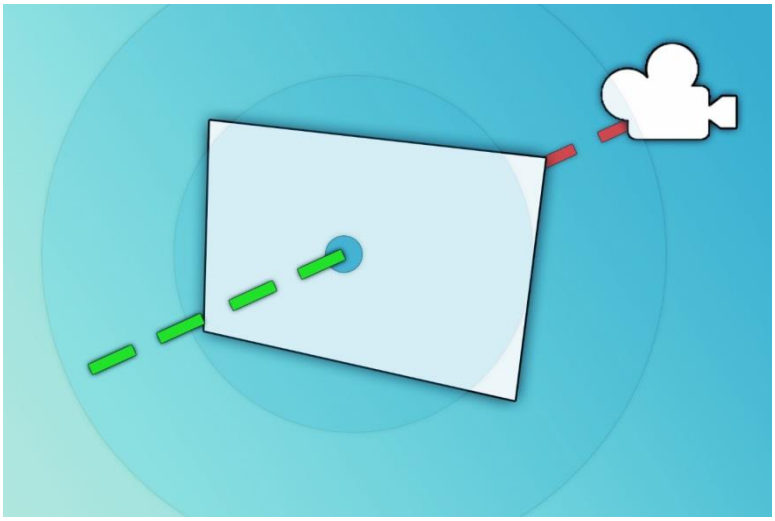


Figure 4.8 Camera Collision

## 4.4 Conclusion and Future Work:

The aim of this document is to describe how the problem was analyzed, development methods, development processes carried out for creating a working product and provides information on what went right and wrong about the project and the lessons learned from experiences that have been gained during a 6 months project period and our discussions.

In the future, we will try to design and advance the project if we have enough time We also want to add an online server and make it a Multy-Player online Game, and add more options to Choose from the game different maps and weapons, and build our beauty Character's and enemy's Character's with high graphics, and then we will upload in the Play Store.

# References


- Sergey, M. (2013). Practical Game Design with Unity and Playmaker. Available a[https://oiipdf.com/practical-game-design-with-unity-and-playmaker] (Accessed at : 27th April 2021)

- Matthew, J. and James A. Learning 2D Game Development with Unity. Available[https://ptgmedia.pearsoncmg.com/images/9780321957726/samplepages/9780321957726.pdf] (Accessed at : 27th April 2021)

- David, G. (2013). Game Development with Unity3D. Available A[http://seriousgamesnet.eu/mod/elgg_segan_framework/static/conferences/2013/presentations/10_Game_Development_Using_Unity_David_Gouveia.pdf] (Accessed at : 27th April 2021)

- Nikhil B. Unity 3D Game Engine Seminar. Available A[https://www.slideshare.net/NikhilThorat15/unity-3d-game-engine-seminar] (Accessed at : 27th April 2021)

- Peng X. (2014). Unity 3D Game Engine Seminar. Available A[3D Game Development with Unity A Case Study: A First-Person Shooter (FPS) Game] (Accessed at : 27th April 2021)


## پۆختە


**لەم توێژینەوەیەدا , هەستاوین بە دروست کردن و پەرەپێدانی یارییەك کە ناومان ناوە (سوات شوتەر) بە بەکارهێنانی پروگرامی (یونیتی) کە بەرنامەیەکی تایبەتە بە دروستکردنی یاریە ئەلیکترۆکنیەکان و پشتگیری لە هەریەکە لە زمانی سی شارپ و جاڤاسکریپت دەکات.**

**وە فوتوشوپ و ئیلوستڕەیتور مان بەکار هێناوە بۆ دیزاینکردنی ڕوکاری سەرەکی یاریەکە.**

**لە ئێستادا یاریەکەمان تەنیا لەسەر سیستەمەکانی ئەندروید ئیشدەکات بەڵام لە داهاتودا بۆ هەریەکە لە سیستەکانی ئای ئو ئێس و پیسی بەردەست دەکەین**

**یاریەکەمان لە ئێستادا لە سێ ماپ پێکهاتوە بە ناوەکانی (بارزان, سەنگەسەر, مەخمۆر).**