

Turn Base Game Using Unity

Project Report

DT265

Higher Diploma in Computing

**Devin Forder**

**Brian Gillespie**

School of Computing

Dublin Institute of Technology

**12/01/2017**



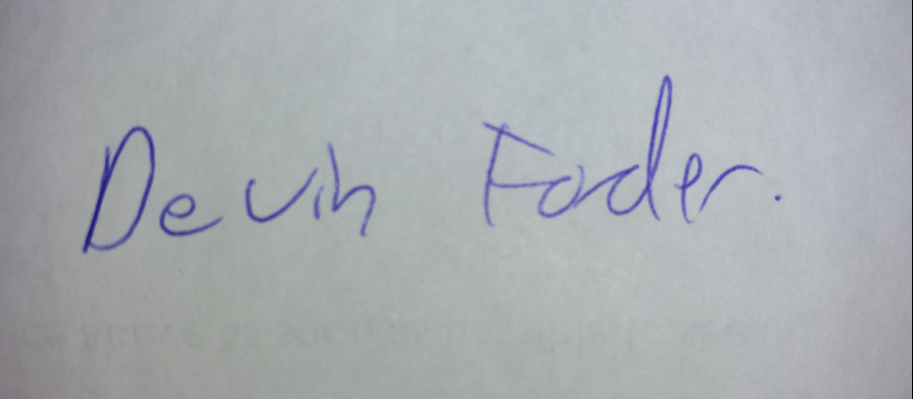
Abstract

The turn-based game, called Bandits and Knights, was created using the Unity Game Engine. It was developed to show an understanding of the Unity system, and basic game mechanics. The game is simple to use and has 2D graphics which are a combination of tile sets and sprite creations. The aim of this project was to make a working game that displayed features of a game such as player choices, character interactions and goals.

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Devin Forder

12/01/2017

Acknowledgements

First, I would like to thank Debbie Forder and Martin Buckley for helping me to complete this course, as without them I wouldn’t have been able to even think about doing this. Thanks to Ruth NicGinneá for helping me to proof read this document and giving some advice on the aesthetics of the game. Thanks to my supervisor Brian Gillespie for helping me think about the best way to go about the project, and for giving me a new way to approach game design.

Contents

[1. Introduction 6](#_Toc472011220)

[1.1 Overview 6](#_Toc472011221)

[1.2 Project Objectives 7](#_Toc472011222)

[1.3 Project Challenges 10](#_Toc472011223)

[2. Background Research 11](#_Toc472011224)

[2.2 Unity 11](#_Toc472011225)

[2.2 Tiled & Sprites 13](#_Toc472011226)

[2.3 Similar Games 15](#_Toc472011227)

[3. Software Design 16](#_Toc472011228)

[3.1 Game Design 16](#_Toc472011229)

[3.1.1 User Interface and Layout 16](#_Toc472011230)

[3.1.2 Code 18](#_Toc472011231)

[3.2 Requirements 19](#_Toc472011232)

[3.3 UML Diagrams 20](#_Toc472011233)

[4. Evaluation 23](#_Toc472011234)

[5. Project Plan 25](#_Toc472011235)

[6. Conclusions 26](#_Toc472011236)

[References 27](#_Toc472011237)

### Introduction

### Overview

This report discusses the development of a turn-based game using the Unity Engine and outlines what I have learned during the course of its development. The original concept was to develop a resource management/turn-based strategy game, based on the Multiplayer Online Battle Area genre, however, due to certain constraints, the scope of my project has changed to the production of a purely turn-based game.

As outlined above, my original concept was to develop a game in which the player managed a team that he/she owned, by buying and selling players. Each player was to have characteristics which affected how well they performed in the turn-based portion of the game. The objective was to make a game where the player felt they had an impact in all aspects of the game. As I began the development, it came to my attention that I would not be able to complete the game as I had initially planned, as my technical skills would not allow for it in the allotted time allowed for completion.

Another challenge I faced was my lack of experience in programming, my first formal introduction into which began at the outset of this course. Using both Unity and C# have presented challenges, which I have worked hard throughout the duration of the project to overcome. Given the constraints presented both in the amount of time I had to complete a game as complex as I had initially planned, and my inexperience in the area, I decided that a purely turn-based game, of medium complexity was the best approach.

The game I have developed is a two player game, with both players choosing a type of weapon that their character will use. This choice determines if a character takes damage, inflicts damage or if neither character takes/inflicts any damage. The game concept, in its most simplified form, can be likened to the game of rock, paper and scissors, in that the choice the players make determines the outcome of the game. The graphics of the game are basic 2D, as my aim was to focus on the technical development side.

### Project Objectives

There are two objective groups within this project, the first involving the objectives of the game and the other involving the development of the game.

Game Objectives

The objectives of the game, which I have named “Bandits and Knights”, are relatively simple, in that a player can either lose or win based on the weapon choices they make for their characters.

On opening the application, Bandits and Knights.exe, players select resolution 1366 \* 768 and the game begins to load. The Welcome Screen opens and players are advised of the aim of the game and the characters are introduced. Once players are ready to play, there is a “Start Game” button in the centre of the screen to commence the game.

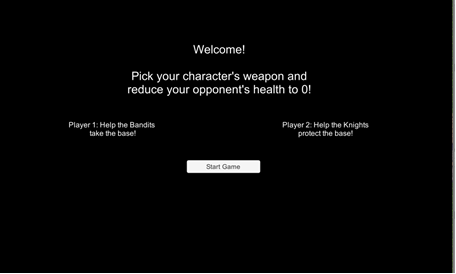


Figure 1 Welcome Screen

Game play starts with three characters on each side of the screen, The Bandits attacking from the left and The Knights defending from the right. Each character begins the game with full health, which is portrayed in a green bar close to the character’s head. During each turn, the players pick a weapon for the active character (which is shown by a green spot appearing by the character) using the respective “Weapons…” dropdown menu for each character. These choices determine the weapon used by the character and will result in a character taking damage, inflicting damage or a draw, in which neither character takes nor inflicts damage. Once a player reduces the health of two or more of his opponents’ characters to 0, the game will end.

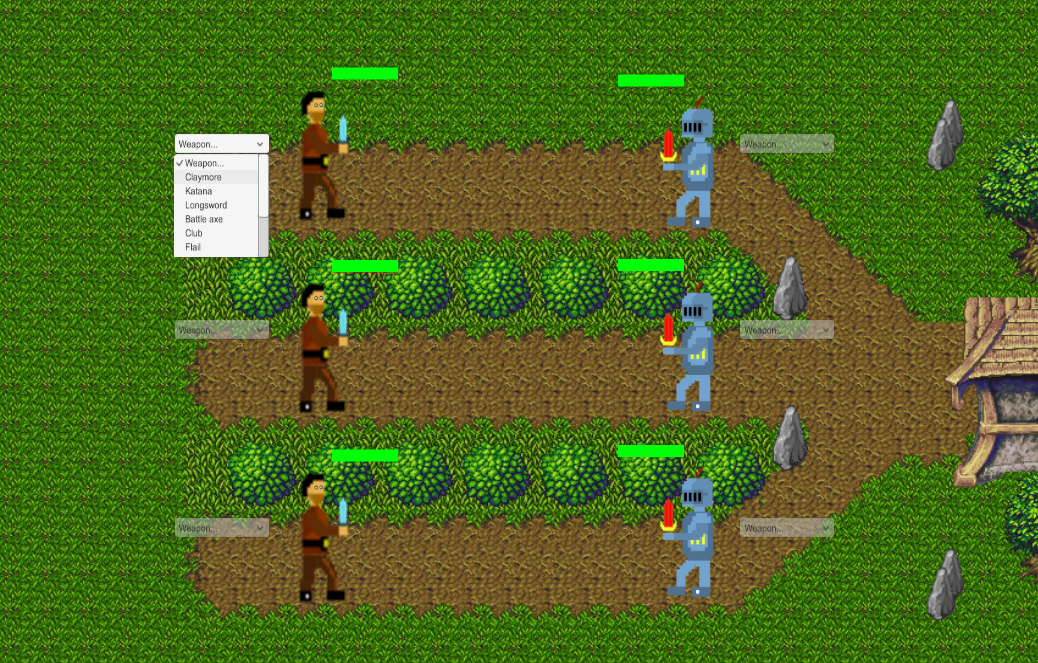


Figure 2 Game Play Screen

The game finishes with a screen advising that the game is over and which character is the winner. This screen also invites players to play again.



Figure 3 Game Over Screen

Development Objectives

The objectives of the development of the game included develop a game which can be played and in doing so, lay down the foundation knowledge for future game development. Throughout the project, I completed extensive research by watching a plethora of development tutorials and by reading a multitude of manuals to get an understanding of how games are developed. I also used my meetings with my supervisor to gain further insight on the workings of certain aspects of gaming and broader programme development. During the course of the development of Bandits and Knights, I have gained experience using the Unity Engine, which has become a widely-used development tool for mobile device application development and Massively Multiplayer Online games, such as Crowfall.

My main goal as part of this project was to complete the process of completing a medium complexity turn-based game in its entirety. I felt this would give me insight into the game development process and the time and effort required to create a game from start to finish. I believe that I have achieved the goals I set out for myself and am eager to use my new skills to continue working on Bandits and Knights to develop it further and bring it more in line with my initial idea for this project, a resource management/turn-based strategy game.

### 1.3 Project Challenges

I faced a number of challenges while completing this project, the biggest of which was the scale and complexity of my initial idea, the development of a resource management/turn-based strategy game. To combat this, I completed online tutorials and supplementary Unity and C# research. As I moved from the planning to the development phase of the game, my inexperience in the game engine and the development language posed too big a challenge for me to overcome in the time allotted for the completion of my project. In consultation with my supervisor, I made the decision to limit the scope of the project, which increased the likelihood of success and allowed for more focused work on the remaining aspect of development.

Effective game development planning was another challenged I faced throughout the lifetime of my project. As I progressed, I found it challenging to familiarise myself with the idea of the game as a collection of classes which had to be attached to gameObjects in Unity. The design of class diagrams also posed me some difficulty. This in turn made it difficult for me to utilise the S.O.L.I.D design principles and how they could be used to effectively and efficiently make a well thought out and well planned game. These difficulties, and my initial confusion on the interaction between gameObjects and the scripts attached to them to each other, made the flow of the game very difficult to achieve, i.e. the turn-phase from one character to another and the lowering of the health bar depending on the weapon selection.

### Background Research

### 2.2 Unity

As part of my research for this project, I drew on my past experience of playing games which used various game engines. This allowed me to experience what could and could not be achieved using the various game engines available. The number of basic, online, flash games I encountered that used the Unity game engine led to my deciding that it was the best engine for the development of my game. I completed some extensive research on the engine and what it had to offer, which included comprehensive learning and tutorial materials. Unity also offers its services without charge, which also factored into my decision making process.

Typically, the Unity engine is used to create 3D games, but in 2013 Unity added functionality to create a 2D game using their engine. In the game engine, you can run a game as it is being developed on the same interface, this function helped me develop the game, as it provided a constant view of the affect of changes that I made and it allowed me to visibly monitor my overall progress. There are also numerous functions provided by the engine itself, for instance the ability to create UI elements, such as buttons, text, etc. on the Unity interface and position them with relative ease. The Networking and Multiplayer functions also factored into my decision as it allows for the functionality required for my initial project idea of the game being an online resource management/turn-based strategy game. Unity also has an Assets Store offering a variety of packages which can be downloaded and used in projects.

As I familiarised myself with C# in the early stages of this course, I decided that it would be best for me to use this language in the development of my project. Unity can be used with C# and you can also set Unity up so that a separate code editor can be used. This fitted in with Microsoft Visual Studio, so was also factored into my decision to use Unity. As well as the functionality included in the engine, the Unity Manual [1] provides a lot of supplementary knowledge on the Unity Engine. It is from this manual that I gained much of insight I now have into the functionality required to develop the game. I also used other sites to find out the benefits of using the Unity Engine [2].

I had to research how to use the Unity Engine so that I could learn the basic functionality that I would need to complete my project. I completed tutorials which used the Unity game engine, some were provided by Unity themselves while others were from other external sites. I completed one which taught me the basics on the multiplayer networking [3], one which discussed the use of UI elements, which are pictured below [4] and one which I found on Pixelnest which discussed the development of a scrolling 2D game, where I learned skills that I would not need to use in the making of my game, but I used it as an introduction to the Unity interface and with it I learned the basics of Unity [5]. This had been updated to use a more recent Unity Version.

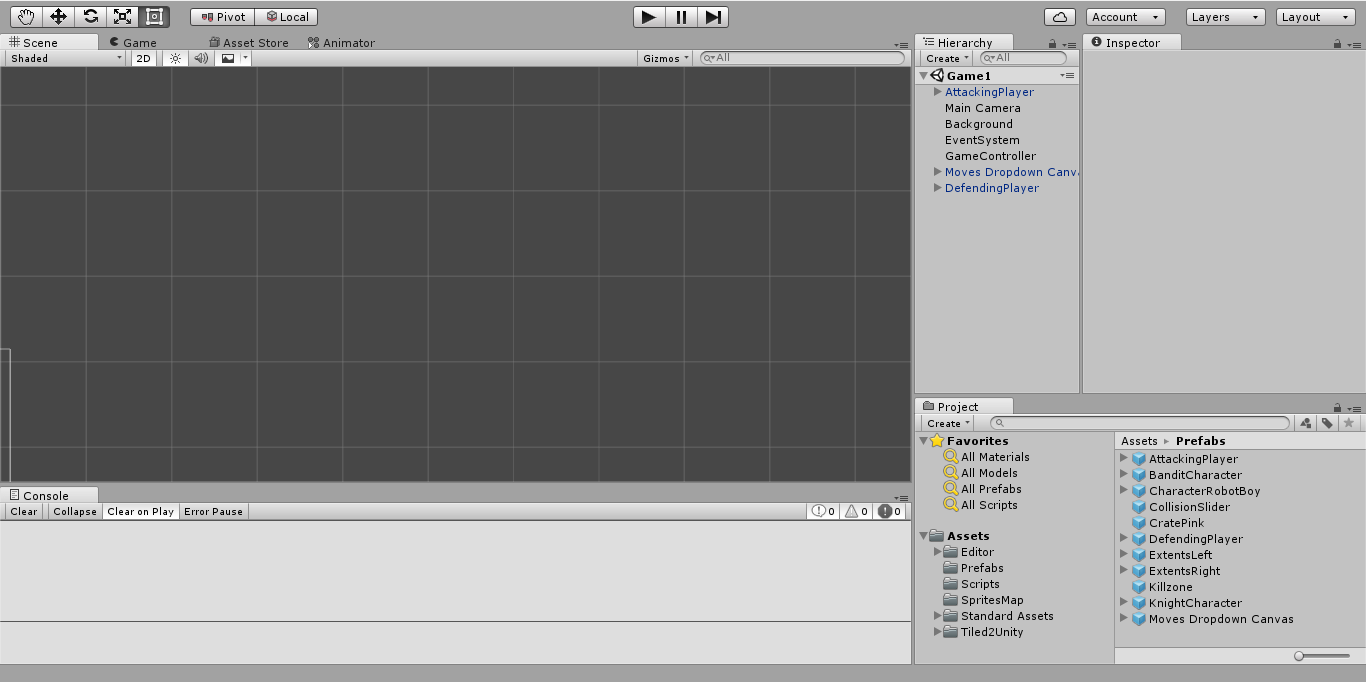


Figure 4Unity User-Interface

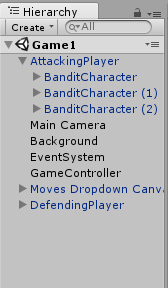
 As for the layout of game elements in Unity I studied how the tutorials laid out the various gameObjects/managers and my supervisor shed some light onto some game functions that are usually implemented. From this I discovered that there can be certain game managers that are included, i.e. a gameController, which would be looking at the end game conditions, or else keep the tick of the game going along where it would be checking the state of the game. This can be seen in the hierarchy tree provided in the Unity User Interface(UI).

Figure 5 Unity UI Hierarchy

### 2.2 Tiled & Sprites

The background of the game was made using Tiled Map Editor and two different tile sets. I used two to try get different looks of the map, the tile sets are called 2D lost Garden tile set transition to Jetrel's Wood Tile set [6] and 2D Lost Garden Zelda style tiles resized to 32x32 with additions [7]. Tiled splits up the images contained within the tile set and allows the user to select the part of the image that they want to make the map out of. I decided to have 3 lanes that lead to a house, which is the base of the defenders.

As with Unity there were some tutorials that I completed to gain the base understanding of the program. The UI of Tiled is relatively simple, but the biggest issue I had was that Tiled does not support exporting Tiled Map Editor files to Unity. Fortunately, there is a free online extension, which can be found here <http://www.seanba.com/tiled2unity>, which provides support for exporting these files to Unity.

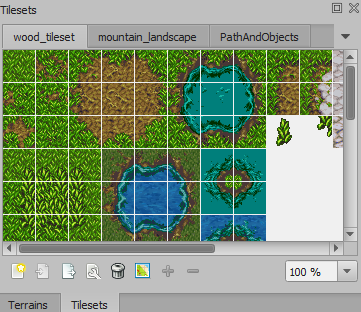


Figure 6 Part of Tiled UI, with tile set.

For my sprite creation, I found two online resources which allowed me to create simple pixel sprites for the Bandits and Knights included in the game. The websites allowed the sprites to be downloaded in PNG format and to be imported into Unity with ease. I used two different online resources, because after creating the Bandit character using <http://www.piskelapp.com/>, I realised that due to the size of the grid that is used, I couldn’t create a defined character. The online resource, <http://pixelartmaker.com/> offers a much larger grid with smaller pixels so it was easier to create shading effects, shapes etc.

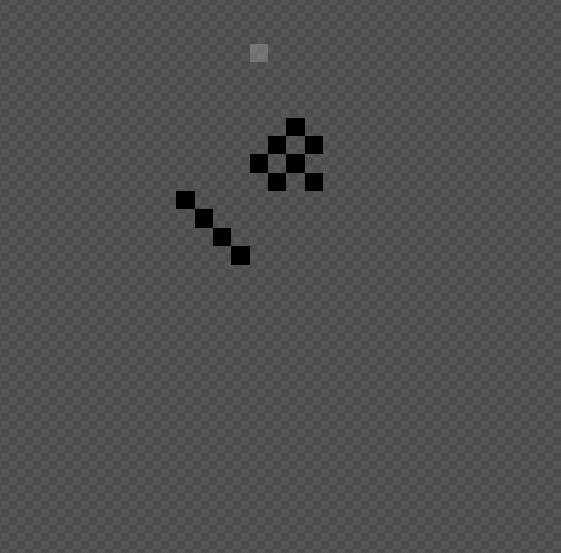
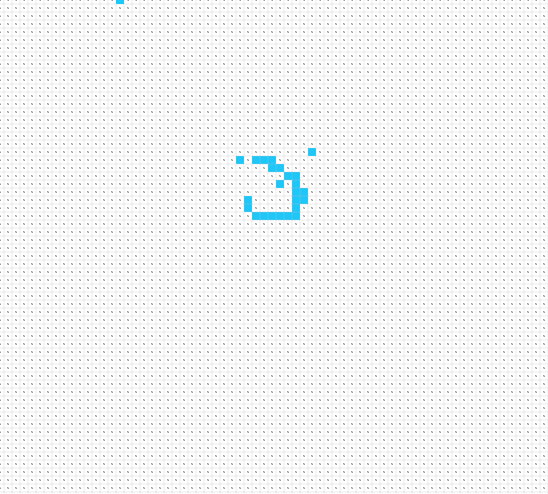


Figure 7 Pixelartmaker Grid

Figure 8 Piskelapp grid

### 2.3 Similar Games

At the beginning of the project, I studied a number of games in both the turn-based genre and the resource management genre, but I was unable to find one that merged the two genres. The closest game that I found that would be in a similar style was Pro Game Manager, which can be found on Steam, (<http://store.steampowered.com/app/408740/>). However, this game offered only the management side of the game I wanted to make and shared similarities with other manager games such as Football Manager.

For the turn-based part of the game, there are many different types of games that offer this feature, but for the purposes of the game I wanted to create, I based my thinking around its design on the idea behind the game of chess.

### Software Design

### Game Design

### 3.1.1 User Interface and Layout

The game is designed to be played by two players, who will pick a weapon for each of their characters to use. The goal is to reduce your opponent’s health to 0 and once three characters have been defeated (regardless of which teams they are on) the game will end, declaring a winner. A player can win by defeating two or three of the opposing characters. The health of each character is represented by a green bar which will change in size once damage has been inflicted. The damage is constant for each weapon so it will always take three successful hits to reduce a character’s health to 0. I excluded the online aspect of the game so that it would be completed on time, so the game is played on a single screen. The characters’ weapons are stored in a dropdown menu that is present on the UI and the intractability of these will change depending on which player’s turn it is.

The interaction of the weapons is based on the idea of rock, paper and scissors. I created a list of moves which stores 13 integers, this correlates to the index of the weapons in the dropdown. When a player picks a weapon, its index is stored and the relevant dropdown is set to inactive. The next dropdown will become active, this allows the second player to select a weapon and once they have done this the weapons will be compared to see which, if any character wins or there is a draw.

When the game is first loaded, the players are presented with a Game Info screen that explains the goal of the game and gives the general layout of each players’ team. Once a user clicks on start the info screen will disappear and the first dropdown will become interactable. Once the win conditions have been met, the game over screen will appear declaring game over, the winner and the option to restart the game or exit.

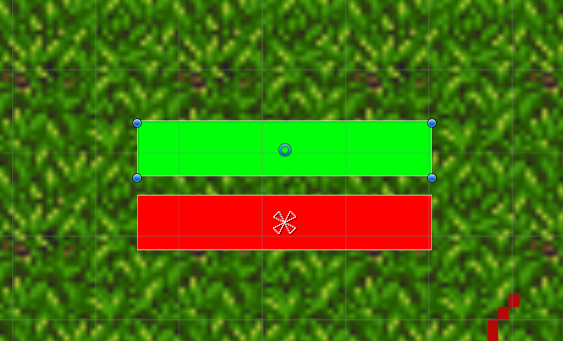
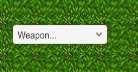


Figure 9 Game Over Screen



Figure 10 Game Start Screen

The map has 3 lanes, with indicates which characters are fighting each other. Each character is made up of various parts created in the sprite editor, which are positioned as children elements in the relevant character’s gameObject. Included in the gameObject is the health bar canvas, this contains the background and foreground of the health bar. The foreground is the element which changes size as damage is inflicted. The dropdown menus are in their own canvas, with contains 6 menus for each of the characters present.



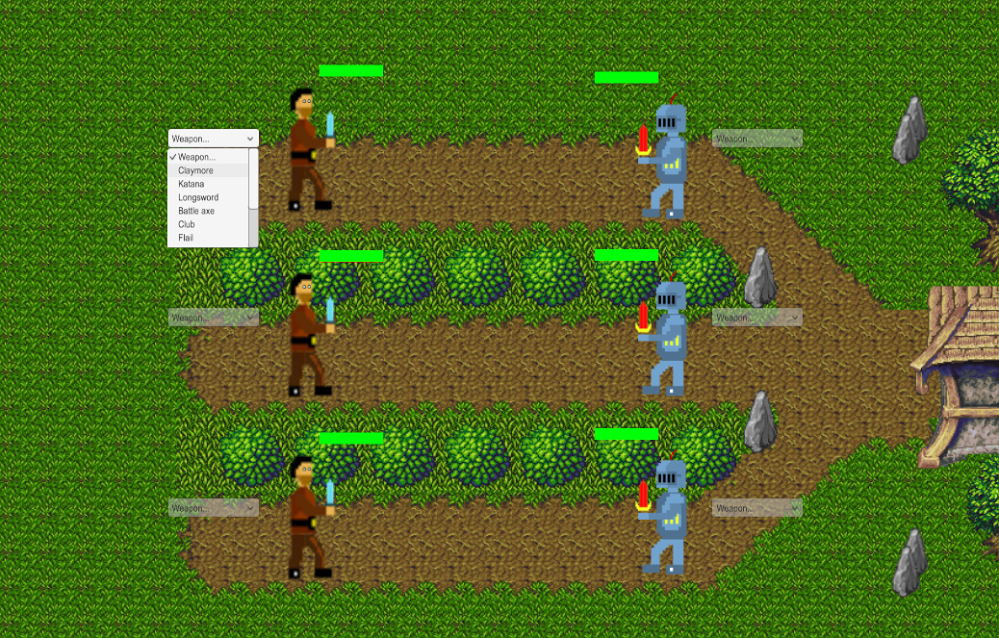


Figure 11 Components of the Health bar

Figure 12 Dropdown Menu

Figure 13 Layout at start of game

### 3.1.2 Code

One of the features of Unity is that when declaring variables in the code you can include a reference to the variable from the Unity Interface. I made use of this as I needed to interact with various interface objects.

Each character has three scripts attached to it which are used during the game. These are PlayerScript, Moves and Health Script. The Health Script controls the change of the health bar and current health. The methods included here are takeDamage(), onChangeHealth() and resetHealth(). In order for takeDamage() to call onChangeHealth() health it checks to see if the damage being received is greater than 0, which would indicate that the character has lost, if this check is true, then onChangeHealth() is called. onChangeHealth() reduces the health bar by changing the width for the foreground health bar, this reference was passed in using the Unity Interface. The resetHealth method uses the background health bar reference to change the size of the foreground health bar to its original size.

The moves script contains all the logic for creating the list of integers for comparison against the dropdown weapon index. It creates four lists in total, one with all possible weapon indexes and three which contain indexes for the weapons that will win, lose or draw. The method for comparing these indexes accepts two parameters which are the indexes of the weapons. From there I used a switch statement, which will go to the relevant case of the weapon, check the three lists to see if the index is present in the win, draw or lose list. It will then return damage if it is in the lose list, or else return 0 damage. This damage is returned to the takeDamage method located in the Health Script.

The Player Script contains a reference to these two scripts by calling the Get Component method and stores them in the relevant variables declared. The only method in Player Script is called attack, which combines the comparing method of the moves script and the take damage method in the Health Script.

The GameController object has the GameController script attached to it, this essentially runs the game by hiding or showing elements and checking win conditions. It contains numerous methods, like attackPhase, which passes the stored weapon numbers input from the dropdown menu into the relevant character’s attack method in the Player Script.

For the turn phase the controller looks to see if both characters are active in a lane, if one character has been defeated on either side then it will check the next lane. If both characters are active it will activate the Bandits dropdown to begin the turn phase for the lane. I did it this way because if either character in a lane has been defeated then it can skip the single character in the lane as it has already been won.

The win conditions that the controller looks for to end the game is if either three characters on one side have been defeated, or if two out of three characters have been defeated. This way there can be no draw in the game. In this method, it checks to see if one of the character objects has 0 health and if this is true, then it will disable the character.

### 3.2 Requirements

From the design above, the functional requirements of the game are:

* The user must be able to pick a move for a character,
* The game must end when the win conditions are reached,
* The game must offer the option to start when loaded,
* When the game is over the user must be presented with the option to restart or exit,
* The user needs to be able to know the outcome of a turn by the updating of the health bar.

Non-functional requirements based on design above:

* Each character must begin a new game with full health, at the beginning of a game or after a restart,
* When a character’s health reaches 0 they must become inactive, and the turn phase needs to skip them,
* The turn phase needs to continue after each selection of a weapon.

### 3.3 UML Diagrams

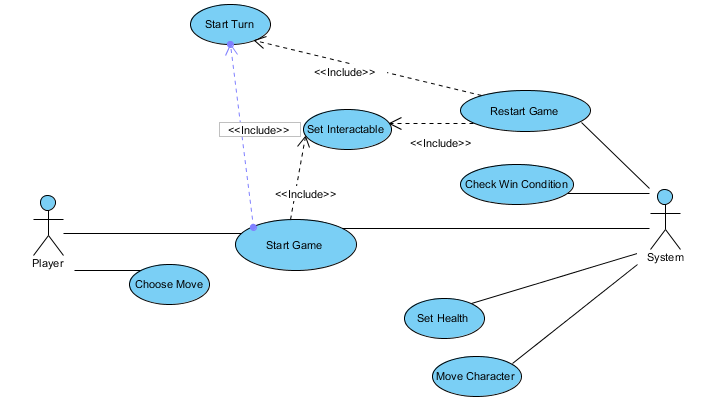


Figure 14 Updated Use Case Diagram

Updated Use Case Diagram, to reflect my choice to exclude the networking functionality.

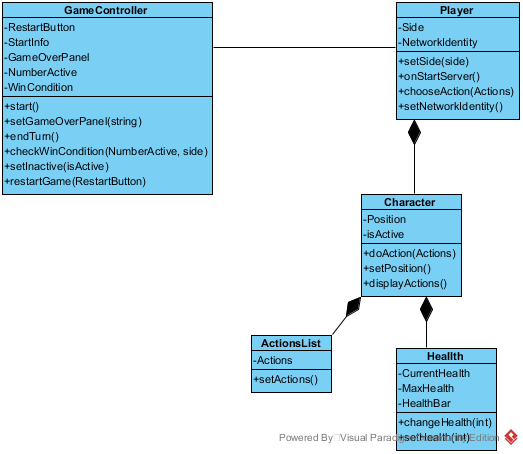


Figure 15 The first Class Diagram made

The above was the first class diagram that I made for the game. I expected that there would be more attributes and methods contained within the Game Controller class, which is reflected in the new class diagram.

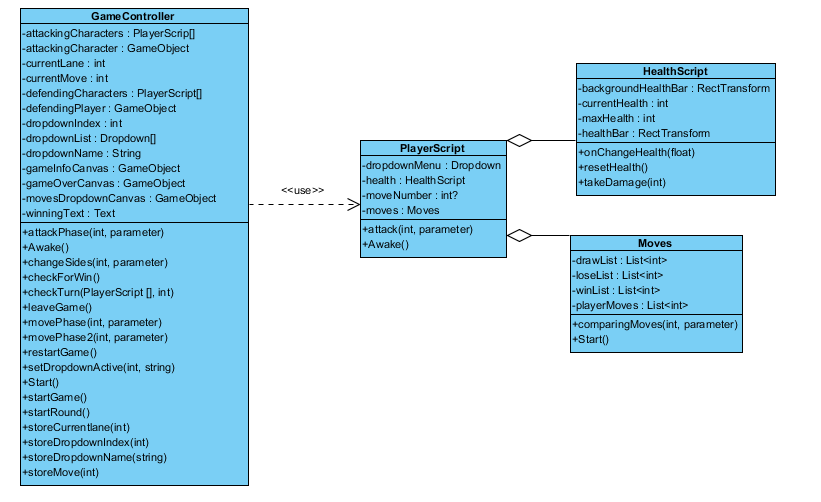


Figure 16 Updated Class Diagram

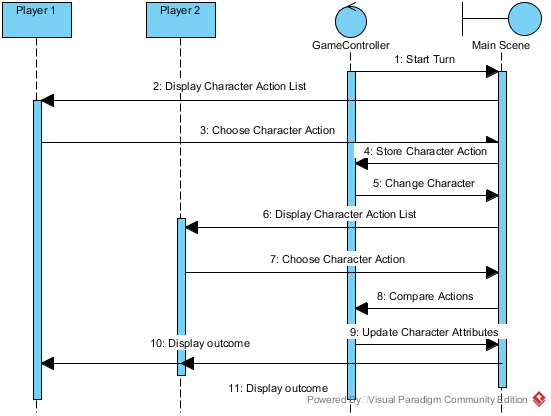


Figure 17 Turn Phase between players

This sequence diagram is how the turn phase to go between the two players flows. Each player chooses an action for their character in the first position, which is the top lane. Once Player 1 has picked a weapon, the gameController then moves to the opposing character on the defending side for Player 2 to pick an action. These are then compared to see what the outcome of the interaction was and the scene will be updated if any damage has been dealt to either character. This is repeated for all characters until the game is over.

### Evaluation

For an evaluation of the game I uploaded the zip file of the game with the current build in it to Dropbox, and then I sent the link to multiple people/groups to get their feedback. The ages of the people that I sent the link to are all between the ages of 23 – 30 years old. The vary in the amount experience they have playing games, from rarely playing games to playing games multiple times a week. Once they had played the game they then filled out a questionnaire that I had made. The overall results were good and the feedback that I got was helpful. I implemented some small changes that I could, such as changing the scroll speed on the dropdowns, as this slowed down the flow of the game. Unfortunately, my dataset of those who answered the survey is small, but it has still given me an idea of stuff that was good/bad.

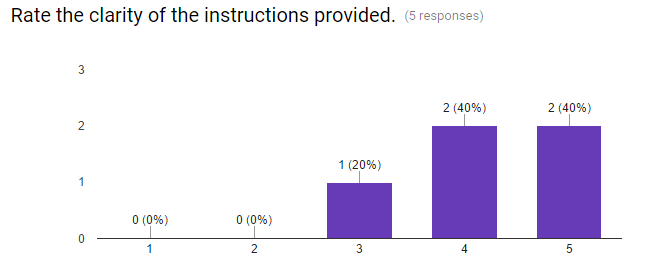


Figure 18 Results from the survey

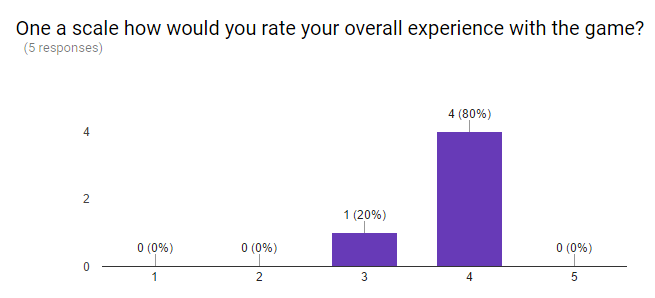


Figure 19 Results from the survey

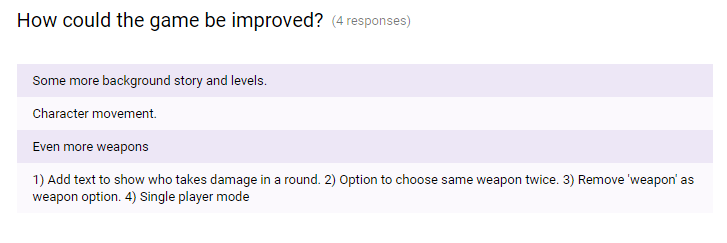


Figure 20 Results from Survey

The survey helped to find some issue with the actual game, like users not being able to choose the same weapon twice and being able to select weapon as a choice for the turn. Users also gave some feedback on how to improve the game to make it more fun, like character movement. People seemed to have no trouble understanding how the game worked or what the goal was, so the flow of the game seemed simple enough that people could just start the game and play. A good thing is that people seemed to have a good experience with the game, so this is a step in the right direction for me to learn how to make a game.

### Project Plan

Before I started the development of the game I wanted to make sure that I had a basic understanding of Unity and the other resources that I decided to use. This meant that I had to spend some time reading manuals and completing tutorials. This was an essential first step of creating this project. As the development began and progressed, there were other resources that I decided I would need to use, and this meant again I would have to take time away from developing to expand my knowledge.

The way that I approached the development of the game was to split it up into different parts that could be developed independently of each other such as the sprites and map. This way it gave me a place to start the game, where I could start making progress and give me an idea of the end result. I continued this process until the last thing that I needed to do was write the code so that the game could run.

As the project went on and I figured out how I could implement the functions outlined in the original proposal. When doing, this I came across various challenges that would determine if a function would be left in the game or removed. A good example of this was when I tried to implement an AI for a user to play against. I research various frameworks to see if I could include one, but most of the ones I came across were pathfinding AIs, which I couldn’t use. After a while of trying to figure out a way to include this function, I decided that it was taking up too much time and it was starting to negatively affect the progress of the game, which led me to decide that the positives no longer outweighed the negatives so I dropped it.

Another feature that I excluded was the online multiplayer. I decided that to make sure that I handed up a project which someone could use, that I would first get the base game running. I feel that I have designed the game that in the future I could add the multiplayer aspect. I wanted to include this in the game so that when each player took a turn that the other player wouldn’t be able to see their turn.

In the proposal, I thought that I would take the Agile approach to develop the game, but now I see that the Iterative and Incremental approach suited the games development better. The whole project was a series of iterations where I would come across a challenge, try to resolve it, and depending on the outcome I would Include or exclude a feature. This seemed to be the way that the project was developed and it suited the way that I had split up the project into different smaller parts. The way I saw it was that if I could manage all the individual smaller parts, I could link them all together to make one complex project.

I think in general most of the challenges I came across were down to inexperience, but I feel that if I were to undertake this project again I would have a more intense planning phase.

### Conclusions

The biggest element of this project was to lay down a good base knowledge of what it takes to make a game. By doing this project I have added some bit of a foundation of the knowledge it will take to make a game, and it has given me a good insight into what exactly can go on during the development/planning of a project. It has given me some experience working with a game engine which is being used to make some very complex games. After beginning this project, my aim switched from trying to make my original game which is more of a personal goal, to understanding that it will take some learning before I will be able to make the final game that I have in my head.

I have many ideas of how to take this project further in the future, as some of my original ideas were removed, and some have been provided from the questionnaire that people have filled out. Now that I have some bit of an understanding of what needs to be done I can move forward and plan for a better game. Some features that could be included are and AI so that it can be a single player game. Some form of character movement so that perhaps it would make the game more fun to play and it will add another dimension to the gameplay. One user also suggested a back story and more levels, which don’t exactly fit into my idea for this game, but it could be another project.

I feel that I have learnt a lot from this experience and it has been really eye opening as at the beginning I felt that it wouldn’t be completed. I think that I have made a game that is simple to use, has some bit of variety, but it also doesn’t have a very long game time, and after a while the users would be able to figure out what weapons they need to pick. Therefore, I think that implementing online multiplayer will be beneficial to the games experience.

### References

[1] Technologies, U. (2016). Unity - Manual: Unity Manual. [online] Docs.unity3d.com. Available at: https://docs.unity3d.com/Manual/index.html [Accessed 14 Nov. 2016].

[2] Blog.udemy.com. (2016). Advantages of the Unity Game Engine – The Ultimate Tool for Game Development. [online] Available at: https://blog.udemy.com/unity-game-engine/ [Accessed 14 Nov. 2016].

[3] Unity. (2016). Unity - Introduction to a Simple Multiplayer Example. [online] Available at: https://unity3d.com/learn/tutorials/topics/multiplayer-networking/introduction-simple-multiplayer-example?playlist=29690 [Accessed 14 Nov. 2016].

[4] Unity. (2016). Unity - Introduction and setting-up the project. [online] Available at: https://unity3d.com/learn/tutorials/tic-tac-toe/introduction-and-setting-project?playlist=17111 [Accessed 14 Nov. 2016].

[5] Mayance, D. and Oger, M. (2013). Pixelnest Studio. [online] Pixelnest Studio. Available at: http://pixelnest.io/tutorials/2d-game-unity/ [Accessed 14 Nov. 2016].

[6] <http://opengameart.org/content/2d-lost-garden-tileset-transition-to-jetrels-wood-tileset>, Credit goes to Jetrel, Daniel Cook, Bertram and Zabin.

[7] <http://opengameart.org/content/2d-lost-garden-zelda-style-tiles-resized-to-32x32-with-additions> Credit goes to Daniel Cook's 2d Circle Graphic Archive, Jetrel's mockups resized 32x32, Bertram's improvements, Zabin's modification and additions, Saphy (TMW) tall grass.