Bilkent University

Department of Computer Engineering

CS 319 Term Project

Group 1H

War of Domination

Final Report

Project Group Members:

Akant Atılgan

Unas Sikandar Butt

Kazım Ayberk Tecimer

Supervisor: Eray Tüzün

Contents

Introduction

The implementation of our game progressed as it was planned. During the second iteration the main focus of the development was the design of the project and adding extra functionalities to the game. As planned, 2 completely new maps were added with the third party software called Tiled Map. Various different obstacles (Tree , house etc.) and water were also added to these maps. We also added a round system to the game which is designed to add some complexity as players can get more chances defeat their opponent with multiple rounds. Additionally a score system was implemented which will keep track of various data related to the game that can be viewed on a separate screen (Damage dealt, ammo used, shots fired etc.). Different character and weapon choices are added so that users can choose them according to their wishes. We also wrote test classes to test the accuracy of our calculations and logic.

To sum up, we implemented almost all of the requirements that we planned yet, some of extra functionalities (brightness and full screen options in the game settings) were not implemented due to time and cost constraints, and preference was given to some more important features over them (round system, different characters, different weapons).

Design Changes

Added Parts

-To add pickup objects to our system we added **Pickup Manager** class and related it with **Game Manager** class.

-To add different weapons in our game we divided the role of our **Weapon Manager** class into **Knife Manager** class and **Bullet Manager** class. **Weapon Manager** class acts like a parent class to these classes.

-To support multiple controllable characters on the screen we added **Moving** , **Enemy** & **Character** classes. **Enemy** and **Character** classes extends moving since both of them are moving objects.

-To store different screens we have created a **Screen View** subsystem. We send one of the screens to the newly added class **Screen Container** to display it.

-To allow users to choose different weapons and characters we added **CharacterViewChoice** class.

-Following Subsystems are added to increase low coupling & high cohesion.

*GameModel

    *personmodel

    *weaponmodel

    *utilitiesmodel

*GameView

    *screenview

    *mapview

*GameControl

    *personcontrol

*weaponcontrol

We added 3 design patterns to our system as explained in the introduction. (Singleton, Façade, Observer) The reason behind these design patterns was to increase the re-usability of our previous implementations. When a change is needed in our system we can easily handle them because of such design patterns.

Singleton Pattern

We choose to use Singleton pattern as it leads us to create a **RoundData** class which can be accessed by every class. This was used in creating different rounds and holding different scores for each round.

Façade Pattern

We choose Façade pattern in our main controller class (**Game Manager**) in order to increase low coupling & high cohesion. Since we had a lot of helper controller classes such as **Weapon Manager, Pickup Manager, Input Manager** etc. Calling single methods in the Façade class that would call all of the necessary methods from other classes was much easier to implement and allowed us to have low coupling.

Observer Pattern

We added observable pattern because in our system we have one-to-many relationship between objects that is if one object is modified, its dependent objects must also be modified. To do so, we added observer **interface** our **character** and **map view** classes to notify them whenever a class in our game extending **java's Observable** class gets modified.

Removed Parts

Removed **ActionListener**, **MouseListener** and **KeyboardListener** interfaces from the UI subsystem because their functionality was already provided in the Slick library.

Removed **Bush**, **Road** and **Water** classes from the Model subsystem and converted them into a single class. (TiledMap)

In first iteration, our map and models (characters, weapons) was too simple and we removed them to add better versions.

We removed previous **Bullet** class and replaced them with proper model classes. (**Weapon**,**Knife**,**Bullet**.)

We removed previous **Map Manager** class and added it's functionalities to the new **Game Manager** class. The reason behind this change was to reduce the weight of Map Manager class because it was doing all collision checks in previous iteration.

Lesson Learnt

This project has been very educational for the team. As computers scientists and engineers we have experienced firsthand the importance of project management.

- This undertaking required time management in order to meet deadlines. The difficulty here was to setup meeting times where all members of the team are available. We realized mid-way that for a meeting to be efficient the meeting tasks should be preplanned.

- The project experience highlighted the importance of communication in the group, as sometimes the information provided by one member was misunderstood by the other members.

- UML diagrams were a big help in the project, as we realized that a properly made diagram made the task of implementation easy. The UMLs also aided in communication.

- We also learned how to document the project, with analysis, design and implementation reports. These reports were very helpful as they were referred to whenever a complication arose, for example: what is the association between two classes?

User Guide

System Requirements

War of Domination is a Java based game. For this reason, Java Run Environment should be present and installed to the computer in order to start the game. Java SE Development Kit 8 must be (JDK 8) installed, which is compatible with Java 8. Thus users must have Java 8 as minimum version. More, Windows 8.1 is the least version (that we tested), 256 MB of Ram is required whereas 1 GB ram and Windows 10 is highly recommended to have smooth game pleasure.

Installation

The game requires less amount of installation. The user is able to run game via .jar file (which is available on github). To run the .jar users first must install JDK8 and after installing it users can run the .jar file by clicking on it twice times. This file can be copied and distributed. This .jar file is created using the eclipse IDE export project function.

How to Play

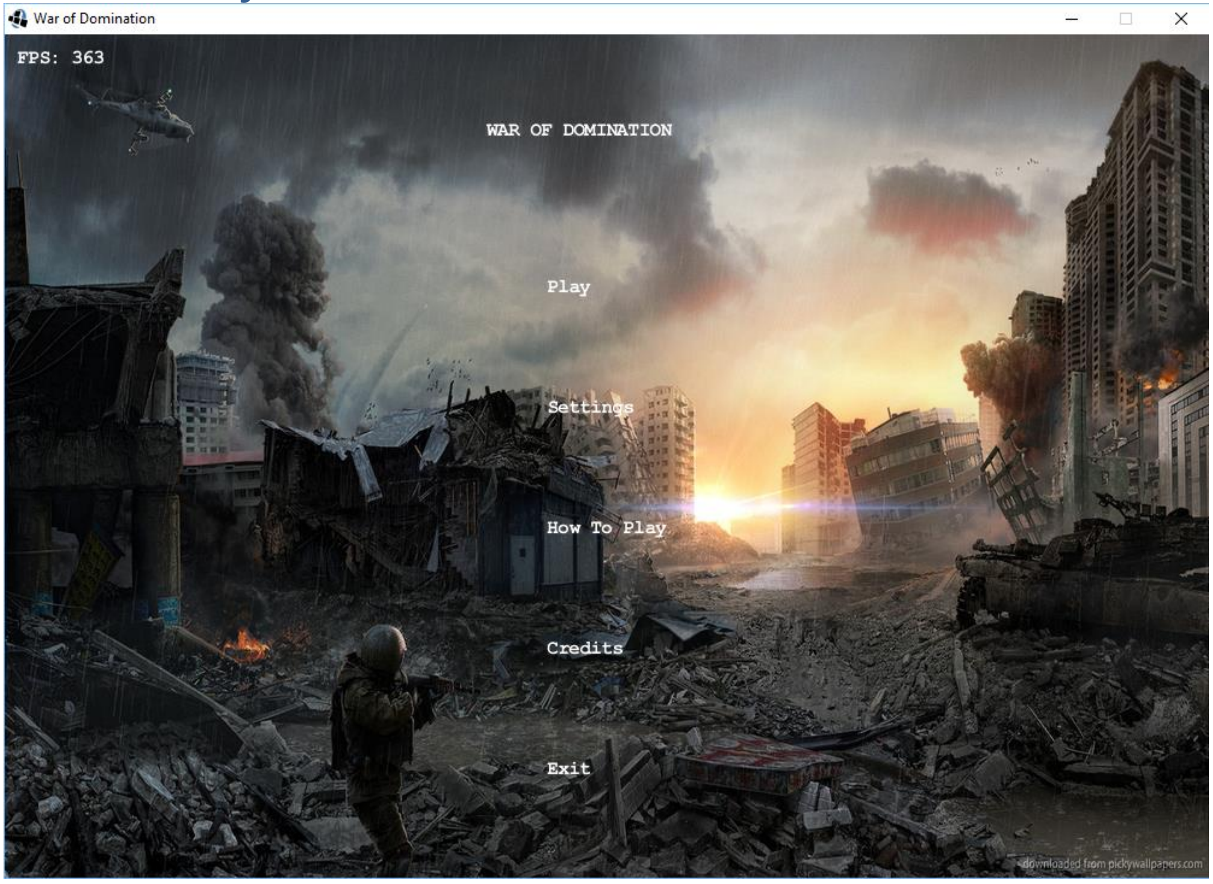


Figure 1: Main Menu

As the Game is started, the first screen that the User is brought to is the Main Menu screen. This screen contains the game title and provides the User five options to choose from. 1. Play, 2. Settings, 3. How to Play, 4. Credits, 5. Exit. The User can then select any of these options by dragging the mouse to the button and left-clicking.

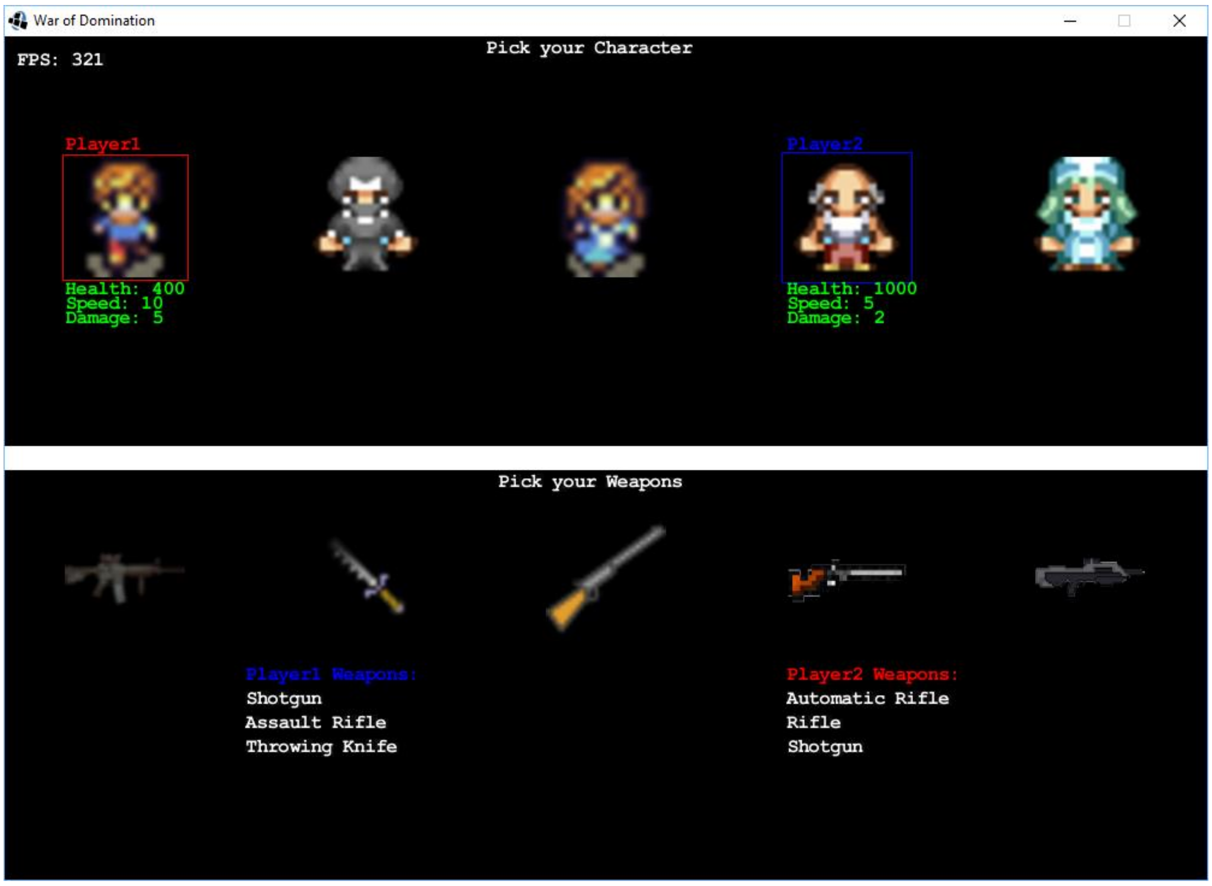


Figure 2: Character and Weapon selection

When the User selects the Play option from the Main Menu, the next screen presented is the character and weapons selection screen. Here the Player-1 can select a character using the left-click button on the mouse on the image of the character desired. Similarly Player-2 can select a character using the right-click button on the mouse. Each and every character comes with its advantages and disadvantages. When the character is selected, its attributes are displayed right underneath it. For the weapons a similar scheme is applied. Player-1 selects weapons using the left-click and Player-2 selects weapons using the right-click on the mouse. Each player can select upto 3 weapons.

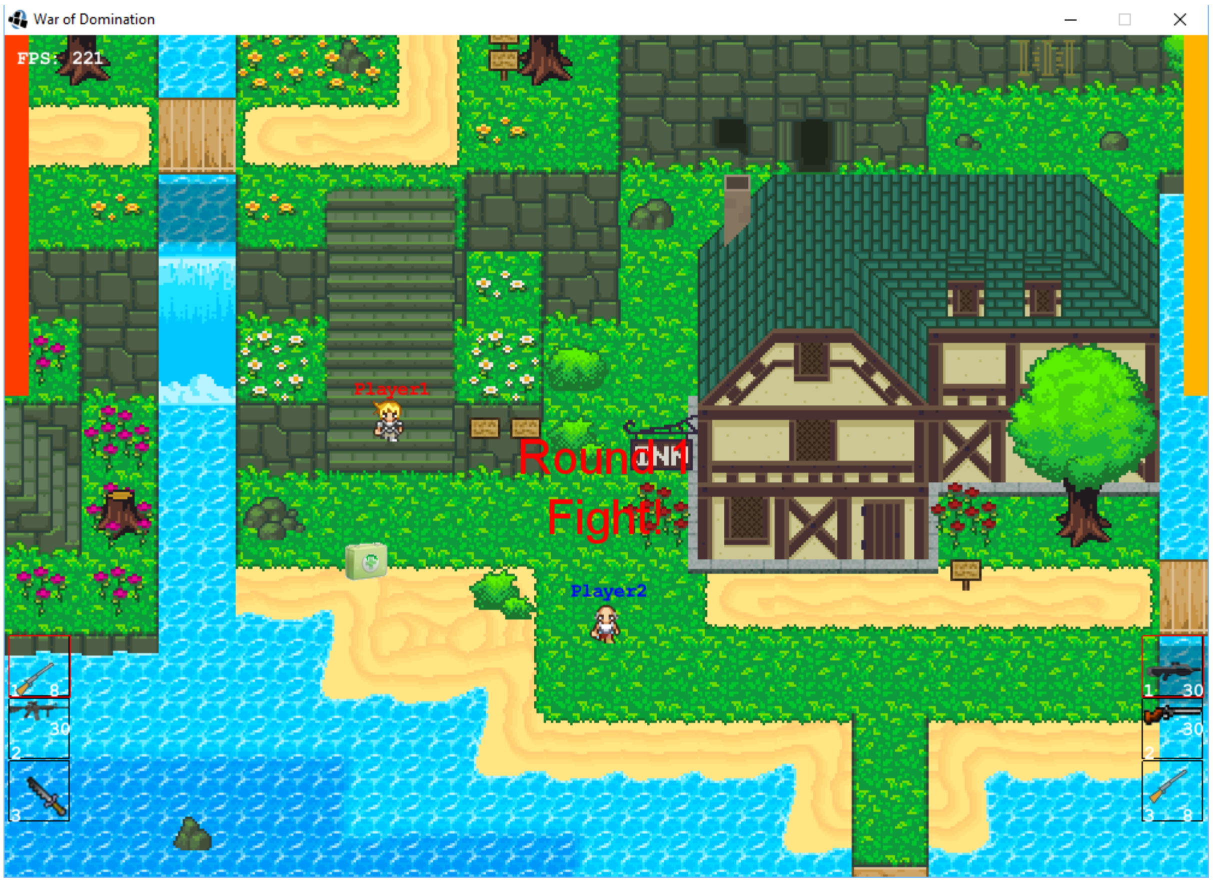


Figure 3: Round 1 Map

After selecting characters and weapons the game enters Round 1. Each round has its own different map. Both players' characters are spawned on the map with the objective of eliminating the other. The map consists of obstacles which hinder player movement and can also provide the player with a reliable cover against enemy attacks. The map also consists of randomly placed power-ups like ammo boxes and health boxes. These provide players crucial sustenance to outlast their opponent. Player-1's Health Bar is displayed as a red bar on the top-left corner of the screen and the inventory on the bottom-left. Similarly Player-2's Health Bar and Inventory is displayed on the top-right and bottom-right respectively.

Arrow keys are assigned to be Player-1's movement keys and Player-2's movement keys are designated as {W, A, S, D} which correspond to {Up, Down, Left, Right} respectively.

Player-1 will be using the left-mouse-button to shoot the selected weapon and the right-ctrl-key to reload the weapon. For Player-2 shooting and reloading are done using the left-ctrl and r-key.

At any given time during the round the Players can press the p-key for pausing the game, o-key to display the options or esc-key to return to the Main Menu.

The round is finished when one of the Players' health reaches 0.

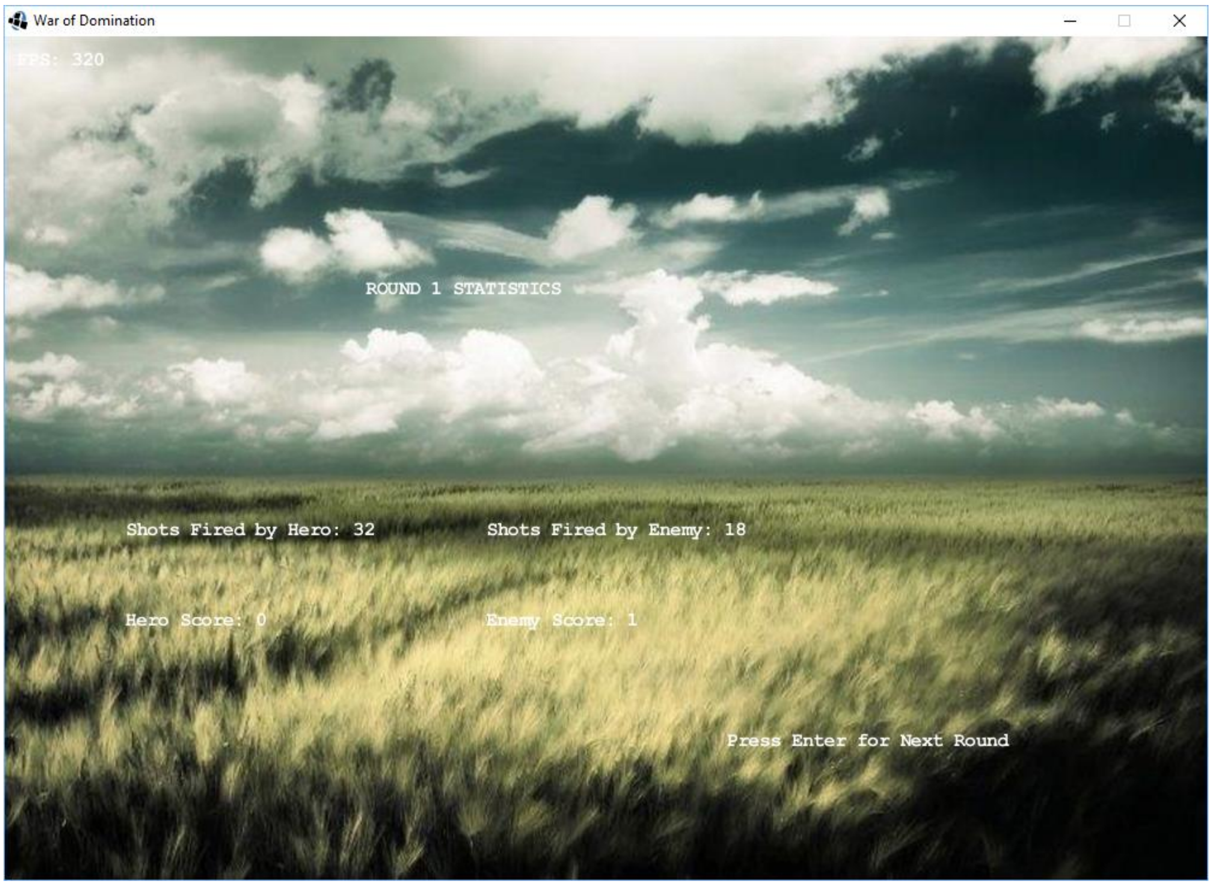


Figure 4: Round Statistics

After the round ends, the Players are presented with the Round Statistics screen which displays the score of the Players and the shots fired by each player. The Players can use the Enter key to continue to the next round.

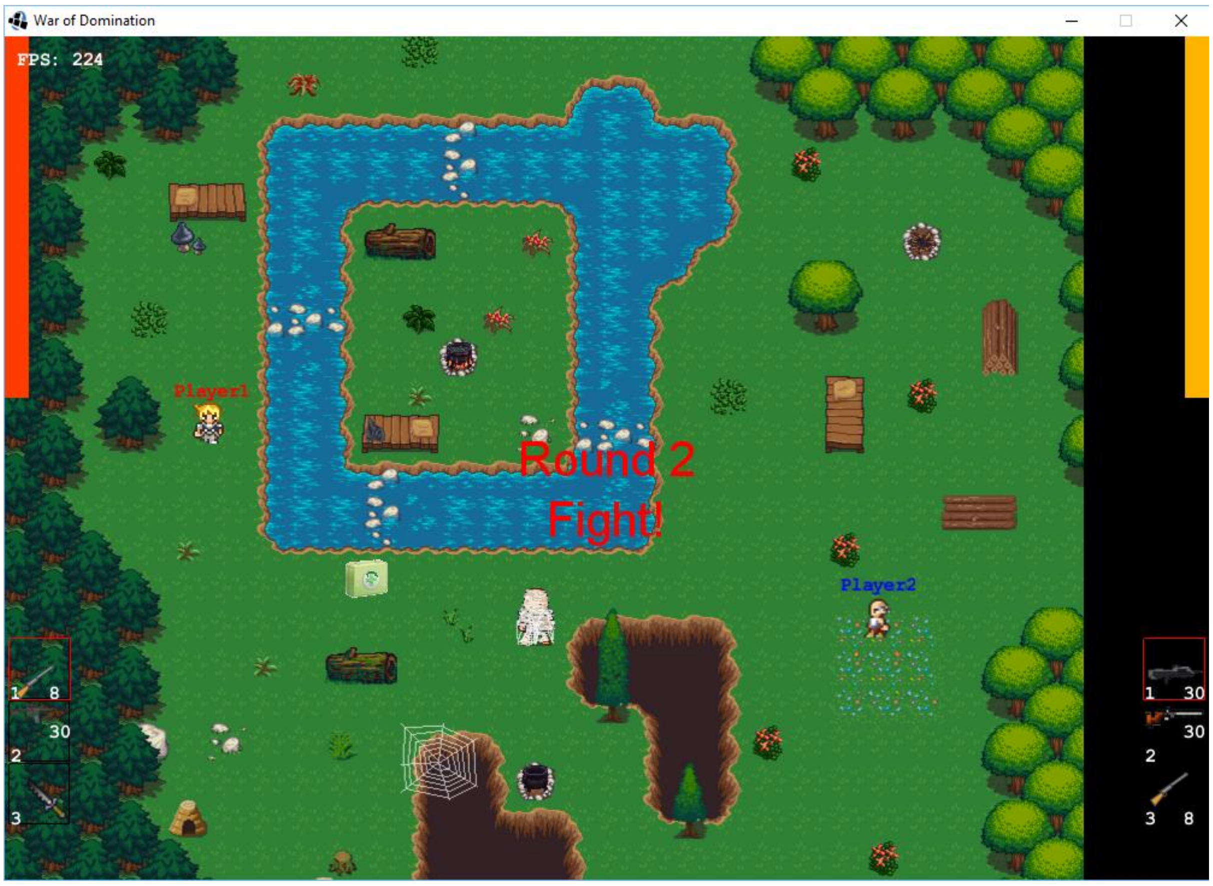


Figure 5: Round 2 Map

Continuing onward from the Round 1 statistics screen the Round 2 map is displayed and both players spawn with full health and full ammo. The rules are the same for Round 2 as well. The Player wins when s/he eliminates the opponent. Even though the rules stay unchanged the map is changed. The map contains different obstacles and terrain. After finishing this round the Players are again showed the round statistics screen as shown above.

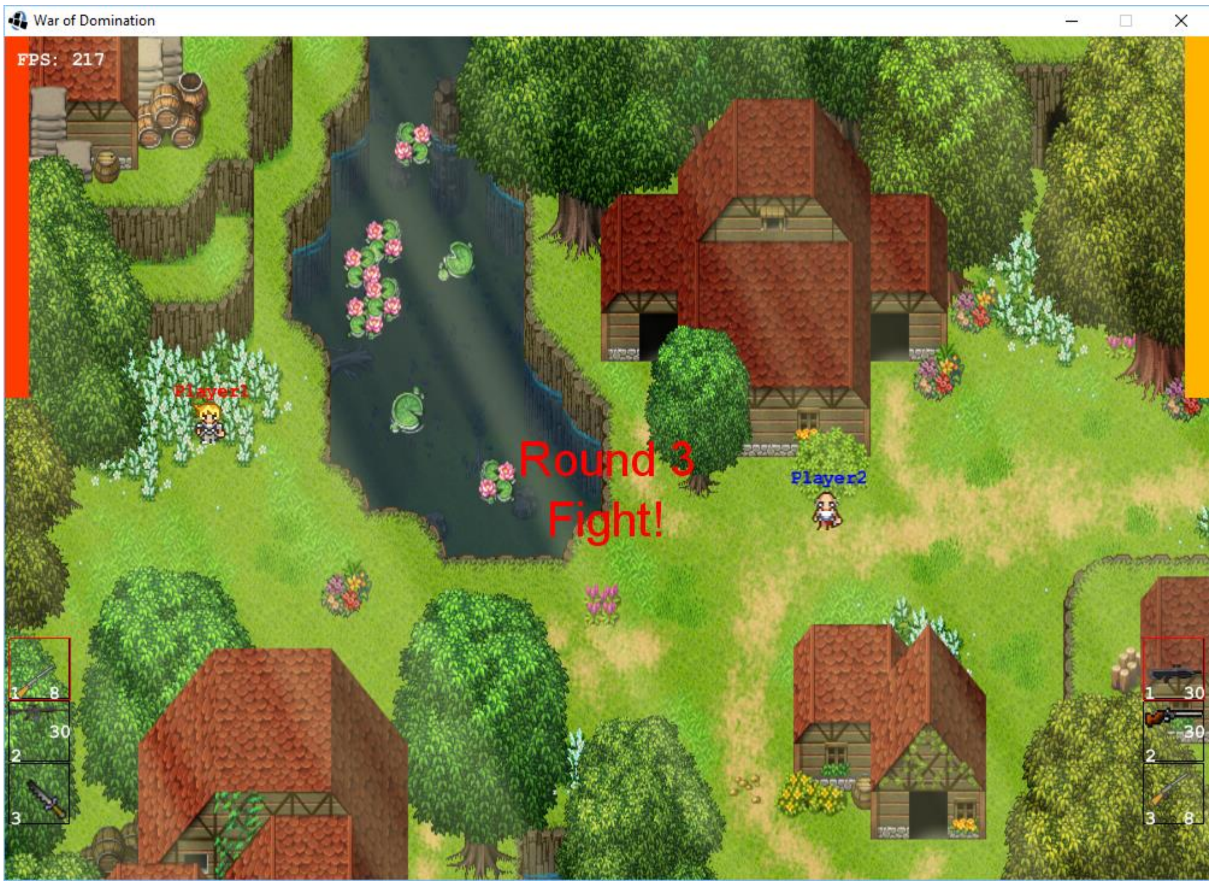
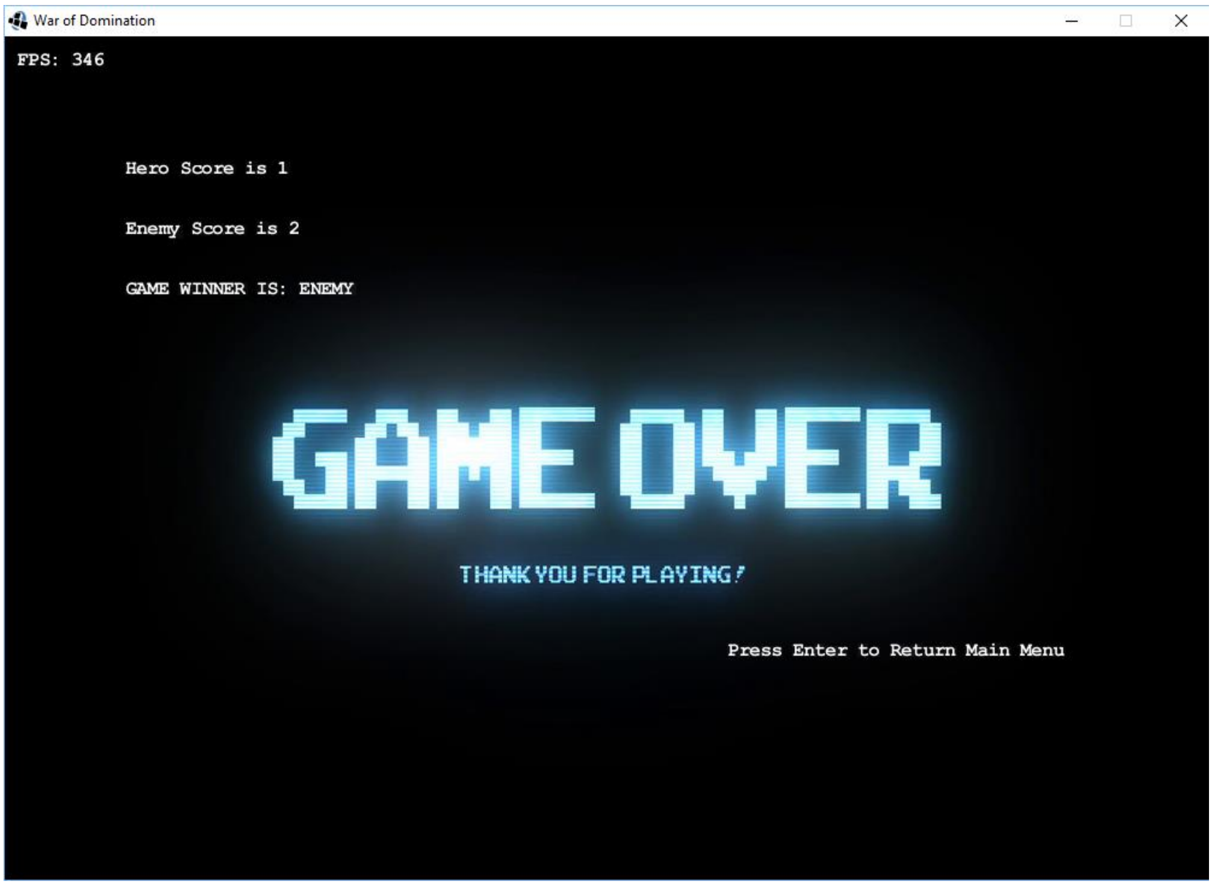


Figure 6: Round 3 Map

Round 3 brings with it a new map and again the obstacles and terrain are changed. This is the final round of the game.



FPS: 346

Hero Score is 1

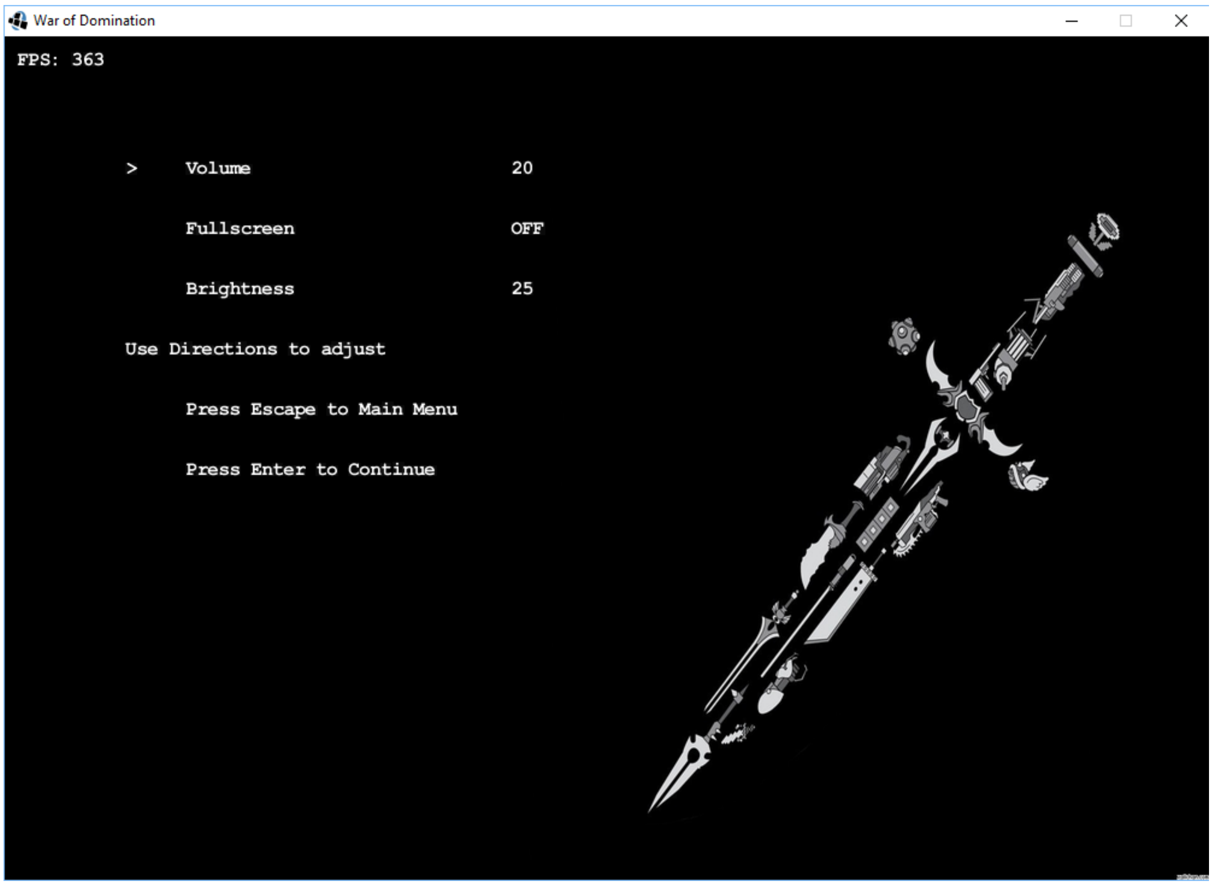
Enemy Score is 2

GAME WINNER IS: ENEMY

Press Enter to Return Main Menu

Figure 7: Game Over

After finishing round 3 the Game Over screen is displayed. Here the Players are presented with the final tally of their scores. Player-1’s score is represented by Hero Score and Player-2’s score is represented by Enemy score. The winner of the game is also announced. The Player can now press the Enter key to return to the Main Menu.



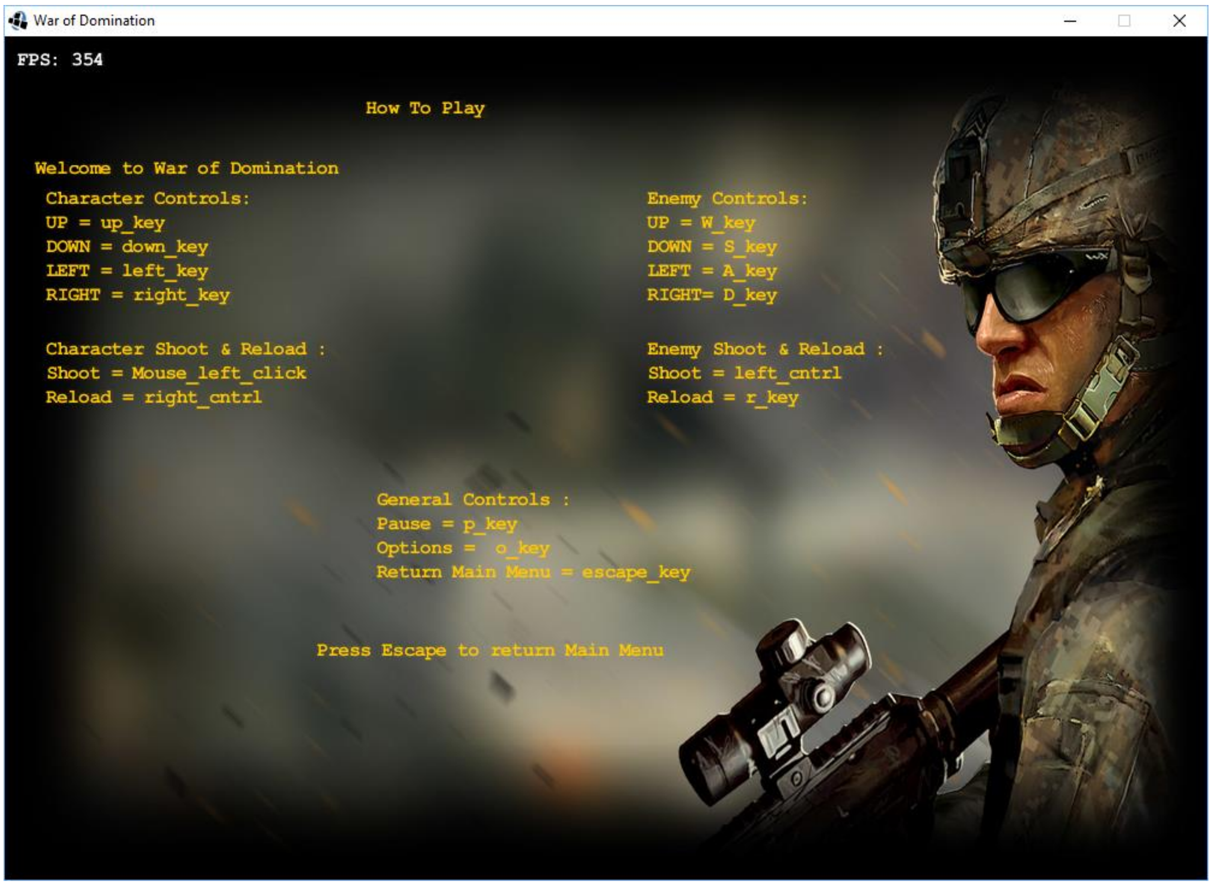
War of Domination

FPS: 363

```
>    Volume                        20

     Fullscreen                    OFF

     Brightness                    25

Use Directions to adjust

     Press Escape to Main Menu

     Press Enter to Continue
```

Figure 8: Settings

If the Player chooses the settings option from the Main Menu, the Settings Menu screen is displayed. Here the Player can change Volume, Brightness and Fullscreen settings. To change the setting the Player must use the up and down direction keys to select the particular setting to change, and the use the left and right direction keys to change the value. After changing the settings the Player can press the esc-key to return to the Main Menu or press the enter-key to continue to play the game.


War of Domination

FPS: 354

How To Play

Welcome to War of Domination

Character Controls:
UP = up_key
DOWN = down_key
LEFT = left_key
RIGHT = right_key

Character Shoot & Reload :
Shoot = Mouse_left_click
Reload = right_cntrl

Enemy Controls:
UP = W_key
DOWN = S_key
LEFT = A_key
RIGHT= D_key

Enemy Shoot & Reload :
Shoot = left_cntrl
Reload = r_key

General Controls :
Pause = p_key
Options = o_key
Return Main Menu = escape_key

Press Escape to return Main Menu

Figure 9: How to Play

The Player can visit the How to Play screen from the Main Menu. This screen displays all the information that the Players need to know to play the game and control their characters.

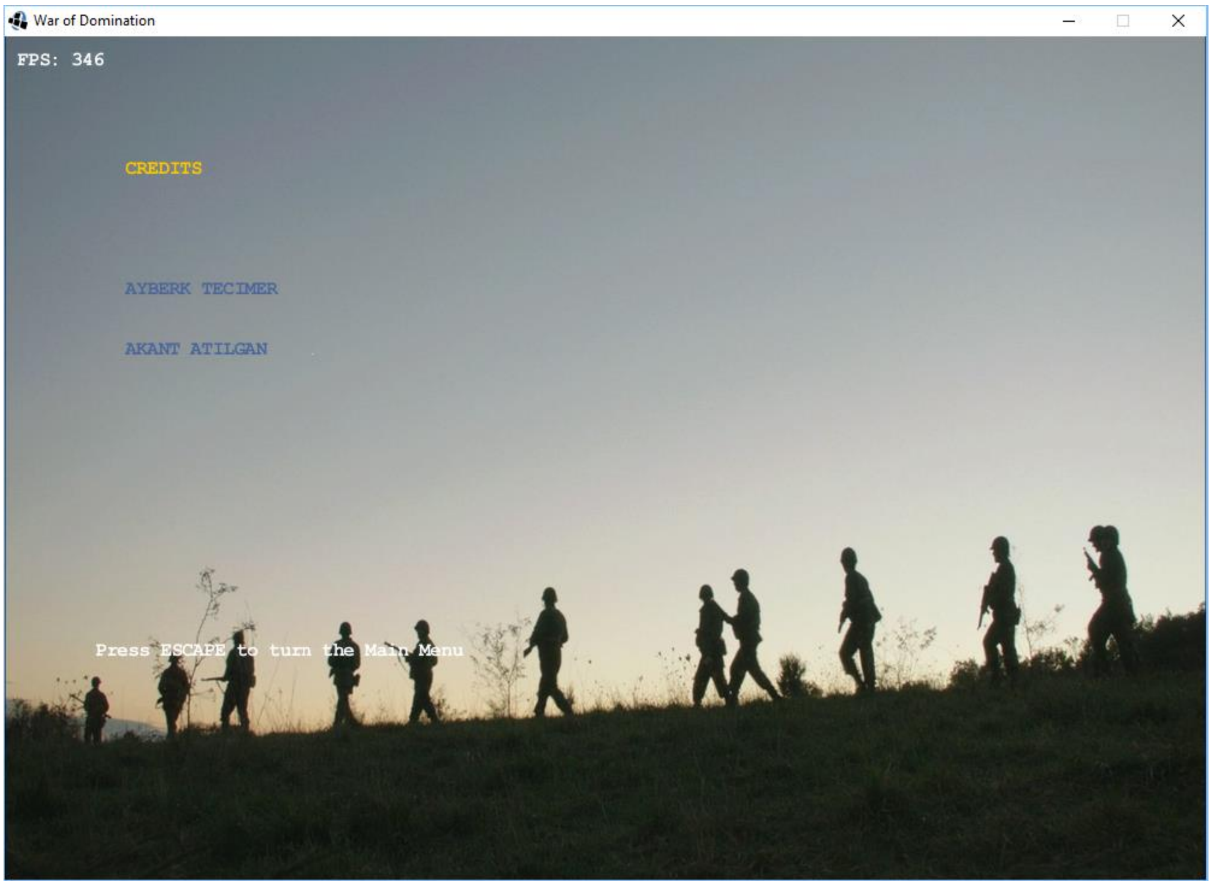


Figure 10: Credits

The User can visit the Credits from the Main Menu. Here the user can get information about the game developers.

References

Tiled Map editor: http://www.mapeditor.org/

Slick library documentation: http://slick.ninjacave.com/