BILKENT UNIVERSITY

DEPARTMENT OF COMPUTER ENGINEERING

CS 319 - Object Oriented Software Engineering

Term Project Iteration 2

FIGHT OR FLIGHT

Analysis Report

03/04/2018

Group 1D

Mihri Nur Ceren

Adahan Yalçınkaya

Berk Erzin

Emre Sülün

TABLE OF CONTENTS

1. Introduction

1.1. Purpose of the system

The purpose is creating an entertaining and competitive game that allows both singleplayer and multiplayer.

1.2. Scope of the system

The system works on any desktop computer on which Java is installed. Minimum version required for Java is 9.0.1. Multiplayer functions supports 2 players on the same computer. So, network connection is not required.

1.3. Objectives and success criteria of the project

Objective of this process is to create a working game that combines two different genres and provides an entertaining experience all the while learning to work on larger projects as a group, utilizing what we learned through the course.

Our success criterias are,

- Successful implementation of “fighting” state.
- Successful implementation of “running” state.
- Successful implementation of multiplayer and singleplayer
- Successful implementation of different weapons and attacks.
- Successful implementation of different enemy kinds.
- Modular and easy to understand source code.

1.4. References

- Bruegge, B, & Dutoit, A 2014, Object-Oriented Software Engineering : Using UML, Patterns, And Java, n.p.: Boston : Prentice Hall, 2014., Bilkent University Library Catalog (BULC), EBSCOhost, viewed 16 February 2018.

2. Overview

Fight or Flight is a hybrid game which consists of both runner game elements and 2-D shooter/RPG game elements. The main aim is to survive as long as you can while getting the most amount of points, like most runner games our game ends with the player eventually dying as well.

In most runner games, the player runs while the speed of the character and the amount of points the player gains increases, the speed increasing to a level so high that the player eventually dies thus the game ends where it is the most exciting. Our game has got these runner game elements described above with one more additional feature. Instead of the game ending with the player dying at some point while running, in “Fight or Flight” the player has the option to stop running and take the game to the “Fight” state, ending the “Flight” state.

In fight state, the area in which the player character can move is limited and the player has to confront waves of orcs, undead and bosses that comes in waves. After the end of each wave, it is up to the player to transition back into “flight” state or keep fighting waves to get more points. Defeating consecutive waves gains player a larger point multiplier but with each wave player also has to face tougher and more numerous enemy groups.

Overall, the point of this game to survive as long as possibly can while accumulating points. You can try to survive on your own or cooperate with a friend since the game can be played together with up to 2 characters.

2.1. Gameplay

Singleplayer: The game begins with the “Flight State” with player character starting his run through an obstacle course like level. In this state player can only control their character’s vertical movement by pressing “W” key for up and “S” key for down. Throughout the level, there will be obstacles to hamper the player. By pressing “W” or “S” the player will change the character’s vertical position to avoid the obstacles or grab items. The player can press “TAB” to switch between states.

In the “Fight” state the player gains full control over his character’s movements. “W” to go up, “A” to go left, “S” to go down and “D” to go right on the screen. Adding to that, in this state player will have to fight against waves of enemies.

“Left Mouse Button” is used for standard attacks and the “Right Mouse Button” is used for special attacks. The player is able to pause the game anytime by pressing “ESC”.

Multiplayer: Multiplayer mode allows another player to take control of a second character. Player 2 uses the arrow keys to move his character instead of “W”, “A”, “S”, “D”. Melee attack is designated to “O” and ranged attack is “P” for Player 2. Switching between states and pausing the game are the same key for both players.

2.2. States

The game has 2 states which player can switch between during a playthrough.

Flight state: The Game starts with this state. Random obstacles appear as the player automatically runs through an ever ending map. Armor and ammo will spawn along the way randomly. The rewards will not spawn entirely random, the faster the character gets, spawn chance of rewards increase. Player only has one health in this states. Failing to avoid an obstacle would result in death. This state ends either by death or by the player switching the state.

Fight state: This state always appears after a Flight state. The automatic running of the player comes to an end with the player now gaining full control over the characters movement. There will be waves of enemies from wave 1 to wave n, n being the number where the player decides to switch the state or die. Player will initially encounter small number of basic enemies. Number of enemies and the difficulty to beat these enemies will increase with each wave with special kind of enemies spawning in further waves. Armor and ammo will randomly spawn in this state as well with spawn chance increasing with each wave. This state ends either by death or by the player switching the state.

2.3. Armor, Power-ups, Enemies, Skills and Characters

This section begins by describing collectible in-game items which directly affect the player character’s attributes. Armor and power-ups are stationary items spawning randomly across the map; increasing the health, mana or damage of the character if they are collected. Following sections include,

The enemies that the player will encounter in the fight stage,

The skills that our player character possess,

The game has 2 different player characters. Brief information about the characters.

- **Paladin Armor**
    - **Judgement Armor:** This is the first possible armor upgrade which increases health by 50 and also increases the Holy Light special abilities damage by 10.
    - **Lightsworn Armor:** This is the second possible armor upgrade which increases health by 75. Melee deals 25 additional damage. Holy Light deals 25 additional damage. It can only be worn by Arathras the Pure.
    - **Ashbringer:** A legendary paladin sword that increases melee damage by 25. It cannot be wielded without Paladin armor. It cannot be wielded by Death Knights.

- **Death Knight Armor**
    - **Betrayer of the Light!:** Harnessing dark energy, the character “Darion” becomes fully equipped with Death knight armor. Holy Light is replaced with Death Coil. Melee deals 50 additional damage.
    - **Scourgeborne Armor:** Strong death knight armor. Melee deals 25 additional damage. Death Coil deals 25 additional damage. It cannot be picked up before “Betrayer of the Light!”.

- **Power ups**
    - **Lesser Healing Potion:** Increases health by 25.
    - **Ancient Health Potion:** Increases health by 50.
    - **Valorous Health Potion:** Increases health by 100.
    - **Lesser Mana Potion:** Increases health by 25.
    - **Ancient Mana Potion:** Increases health by 50.
    - **Valorous Mana Potion:** Increases health by 100.
    - **Altar of the Lightbringer:** Gives the Paladin Max Health and Mana.
    - **Altar of Darkness:** Gives the Death Knight Max Health and Mana.

- **Enemies**
    - **Undead:** Basic enemy. Low health and damage. It can only deal melee damage.

- **Orc Warrior:** Skilled fighter. Moderate health and damage. It can only deal melee damage.
- **Abomination:** Slow moving monstrosity with very high damage and health. It can only deal melee damage.
- **Gargoyle:** A flying creature with low damage and health. It can only deal ranged damage.
- **Boss:** An elite enemy with extremely high damage and health. Has different skills over regular mobs.

- **Paladin Skills**
  - **Melee:** The paladin deals damage with his main weapon.
  - **Holy Light:** A ranged attack that sends a bolt of holy energy to the target. Effective against undead enemies.
  - **Divine Shield:** The paladin becomes invulnerable for 5 seconds.

- **Death Knight Skills**
  - **Melee:** The death knight deals damage with his main weapon.
  - **Death Coil:** A ranged attack that sends a blast of unholy energy to the target. Effective against orcs.
  - **Bone Shield:** The death knight becomes immune to all damage for 5 seconds.

- **Characters**
  - **Arathras:** An experienced paladin of the Silver Hand, mentor of Darion. Stats of the character are below(without any armor or power-ups):
    - Health: 100
    - Mana: 100
    - Melee Damage: 10
    - Ranged Damage: 10
  - **Darion:** A young paladin of the Silver Hand, student of Arathras. Initially a paladin, can later become a death knight. Stats of the character are below(without any armor or power-ups):
    - Health: 100
    - Mana: 100

    - Melee Damage: 10
    - Ranged Damage: 10

3. Functional requirements

- Players should be able to view highscores.
- Players should be able play game with either multiplayer or singleplayer.
- Players should be able to switch between “fight” and “flight” states.
- Players should be able to use weapons and special powers.
- Difficulty level should increase over time with “fight” state containing more enemies and the flight state becoming faster.
- System should spawn obstacles through flight state randomly without ever closing all possible escape routes of the player.
- Players character should be able to activate power ups by walking over them.
- Players should be able to switch between abilities.
- Players should be able to change the background music of the game.
- Players should be able to change control buttons on keyboard.
- Players should be able to see a manual containing information about the game on the “help” screen.
- The enemy characters should have attack functions with appropriate cooldown timers.
- In flight stage, running into an obstacle should be an automatic game-over.
- Players should be able to pause the game to either continue playing, changing settings, viewing help, or exiting the game.

4. Nonfunctional requirements

4.1. Usability

- The system should have a simple interface so that any user who knows basic English can play easily.
- The system should show keyboard controls and allow users to change them.
- The game does not need installment.

4.2. Reliability

- The system should prevent collision of game elements.
- The user's data should be kept in a text file inside the user's computer if a crash were to occur.

4.3. Performance

- The system should provide at least 30 fps for a satisfying experience.
- The system should support 2 concurrent users.
- The response time of all of the movements and attacks should be short.
- Since the fight state has infinite waves with each wave containing more enemies, the game should be able to handle large quantities of objects.
- Since the background changes very rapidly in the "run" state, it should change smoothly without any delays.

4.4. Supportability

- The system should be able to maintain itself if new updates and new game mechanics were to be implemented.

4.5. Implementation

- Users should have a monitor that supports minimum 800x600 resolution.

4.6. Interface

- Users should interact with the system via a keyboard and a mouse
- The interface should be user-friendly, visually satisfactory and intelligible

4.7. Packaging

- Game should be runned via a jar file.
- No installation is required because running the jar file is sufficient.

4.8. Legal

- The system is open source and hosted on GitHub
- The system is licensed under the Apache License 2.0.

5. System models

This section includes static models such as use case model and object model. Also, dynamic models such as sequence diagrams, activity diagram and state transition diagram are given. Finally, mockups are provided.

5.1. Use case model

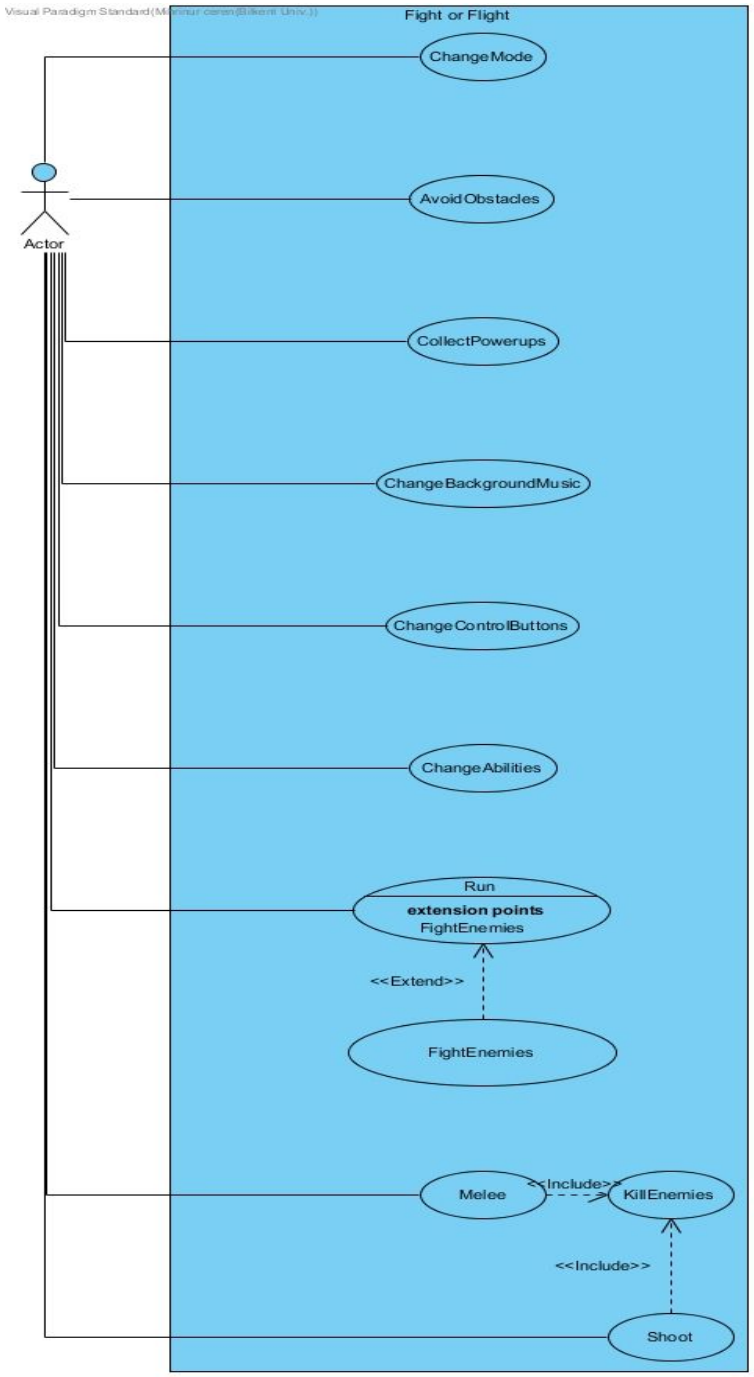

Figure 5.1.1 Use Case Diagram Of The System

5.1.1 **ChangeMode**

Use case name: ChangeMode

Participating actor: Player

Entry condition: Player is playing the game.

Exit condition: The game is over.

Main flow of events:

1. Player is on the flight stage.
2. Player presses the “Change Mode” button to switch the state.
3. Player starts to play the other state(fight state).
4. Player clears a wave of enemies.
5. Player presses the “Change Mode” button to switch the state back to the flight state.
6. Player collides with an obstacle and dies in the flight state. The game is over.

5.1.2 **AvoidObstacles**

Use case name: AvoidObstacles

Participating actor: Player

Entry condition: Player is playing the game.

Exit condition: The game is over.

Main flow of events:

1. Player is playing the flight state.
2. Player moves upwards to avoid a obstacle.
3. Player fails to avoid the next obstacle and dies.

5.1.3 **CollectPowerups**

Use case name: CollectPowerups

Participating actor: Player

Entry condition: Player is playing the game.

Exit condition: The game is over.

Main flow of events:

1. Player is moves and avoid obstacles.
2. Player runs over a powerup.

3. By running over the powerup, player collects it and his health is or mana is increased depending on the type of the powerup.

5.1.4 **ChangeBackgroundMusic**

Use case name: ChangeBackgroundMusic

Participating actor: Player

Entry condition: Prior to playing the game.

Exit condition: Player starts to play the game.

Main flow of events:

1. Before the player starts to play the game, the player can change the background music.
2. Player starts the game.

5.1.5 **ChangeControlButtons**

Use case name: ChangeControlButtons

Participating actor: Player

Entry condition: Prior to playing the game.

Exit condition: Playing the game.

Main flow of events:

1. Before the player starts to play the game, the control buttons can be changed by the player.
2. Player starts to play the game.

5.1.6 **ChangeAbilities**

Use case name: ChangeAbilities

Participating actor: Player

Entry condition: Player is in the game.

Exit condition: Game is over.

Main flow of events:

1. When the player is playing the game, the changeability button is pressed to switch between abilities.
2. Player uses his new ability.

.

5.1.7 **Run**

Use case name: Run

Participating actor: Player

Entry condition: Player is in the game.

Exit condition: Player dies, OR

Player switches the level.

Main flow of events:

1. Player automatically starts running.
2. Player's speed automatically increases.
3. Player is able to change his vertical position by moving upwards or downwards.
4. Player presses the switch level button and gains full control over his movement.

5.1.8 **FightEnemies**

Use case name: FightEnemies

Participating actor: Player

Entry condition: Player does not die in flight state.

Exit condition: Player dies.

Main flow of events:

1. Player moves toward enemies.
2. Player melees the enemies.
3. Player moves away from the enemies.
4. Player shoots at enemies.
5. Player is killed by the enemies, the game is over.

5.1.9 **Shoot**

Use case name: Shoot

Participating actor: Player

Entry condition: Player is in the game.

Exit condition: Player dies.

Main flow of events:

1. When the player confronts enemies, player shoots at them.
2. If the player dies, the game is over.

5.1.10 **Melee**

Use case name: Melee

Participating actor: Player

Entry condition: Player is in the game.

Exit condition: Player dies.

Main flow of events:

1. When the player encounters enemies, player is able to melee them.
2. If the player dies, the game is over.

5.1.11 **KillEnemies**

Use case name: KillEnemies

Participating actor: Player

Entry condition: Player is in the game.

Exit condition: Player dies.

Main flow of events:

1. When the player encounters enemies, player is able to kill enemies by shooting or melee.
2. If the player dies, the game is over.

5.2. Object model

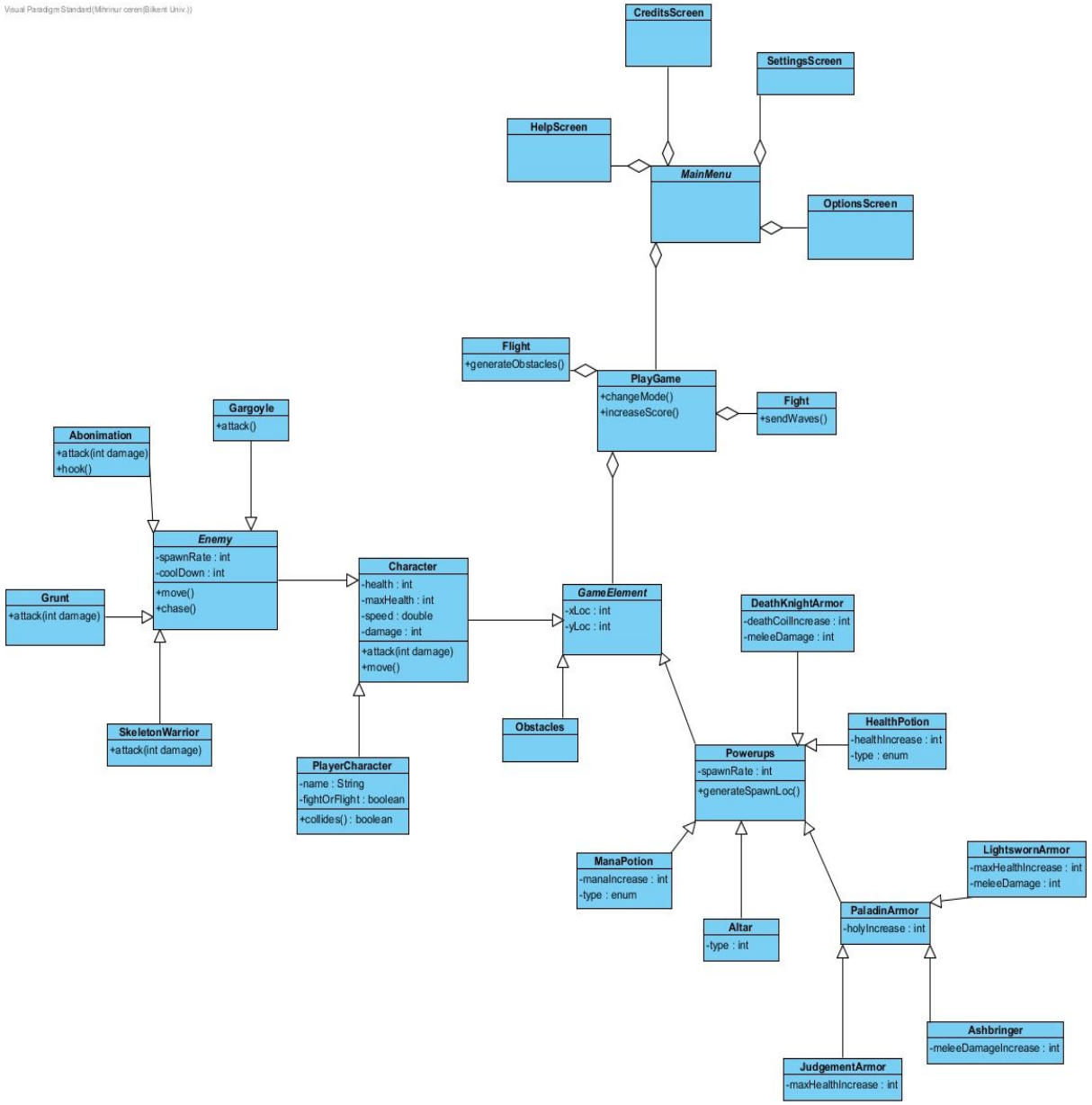

Figure 5.2.1 Object Diagram

Detailed description of object diagram is given below.

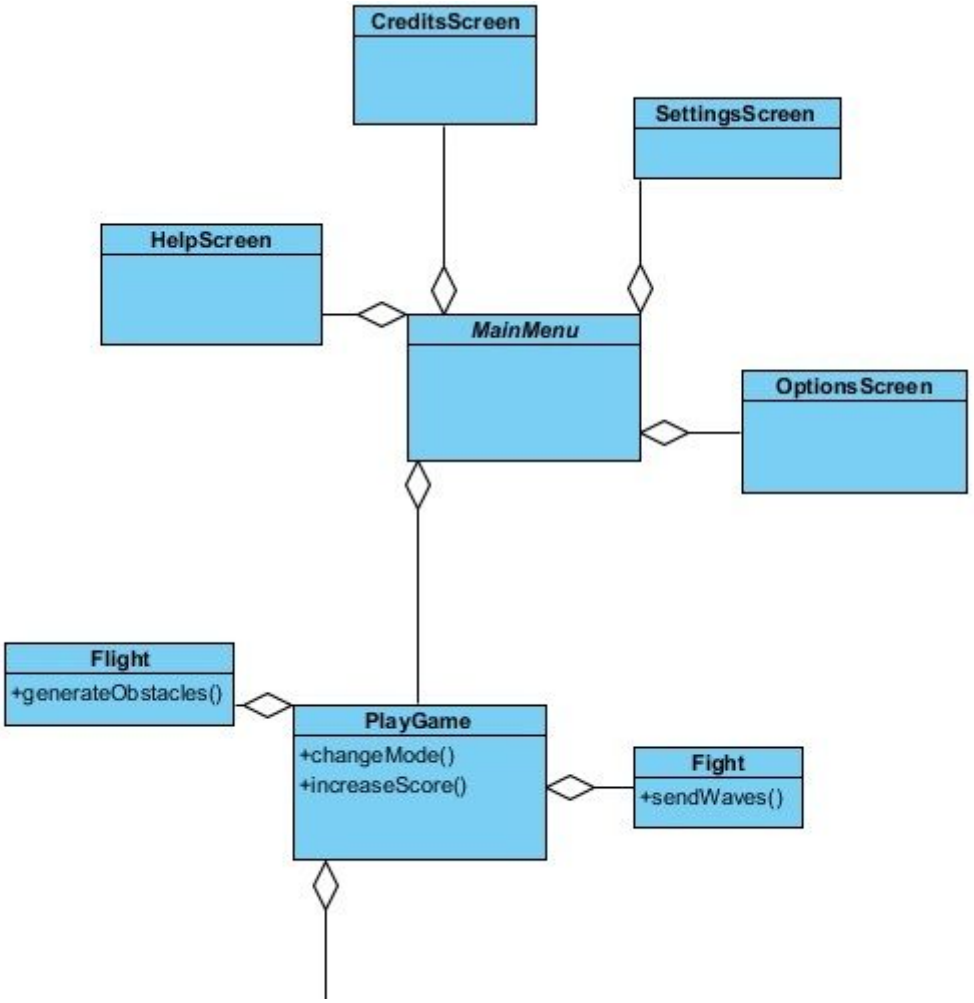

Figure 5.2.2 Beginning of the object diagram

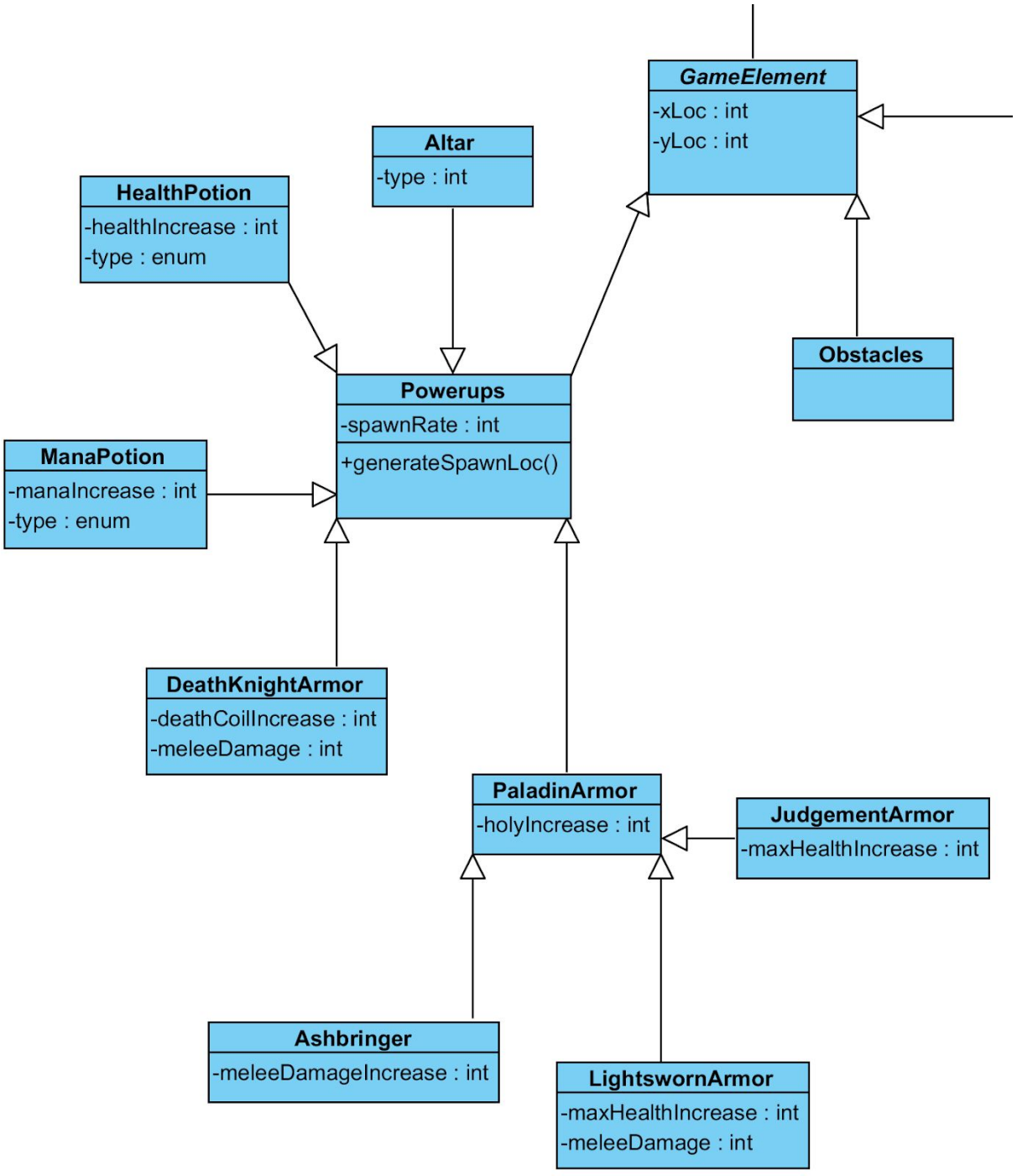

Figure 5.2.3 GameElement aggregates PlayGame

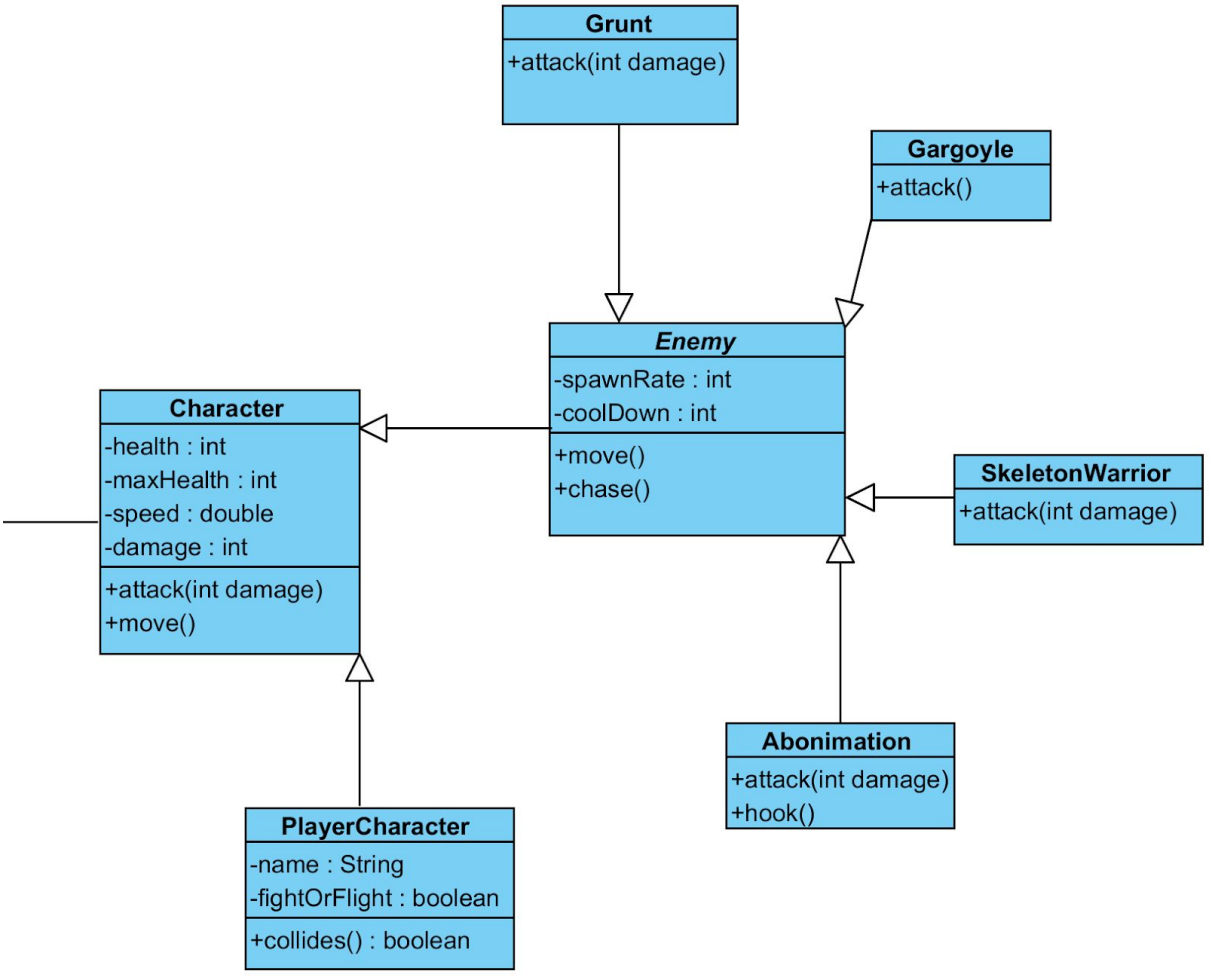

Figure 5.2.4 Character extends GameElement

5.3. Dynamic models

This section describes the flow of events in the game. It includes three sequence diagrams, one activity diagram and one state transition diagram.

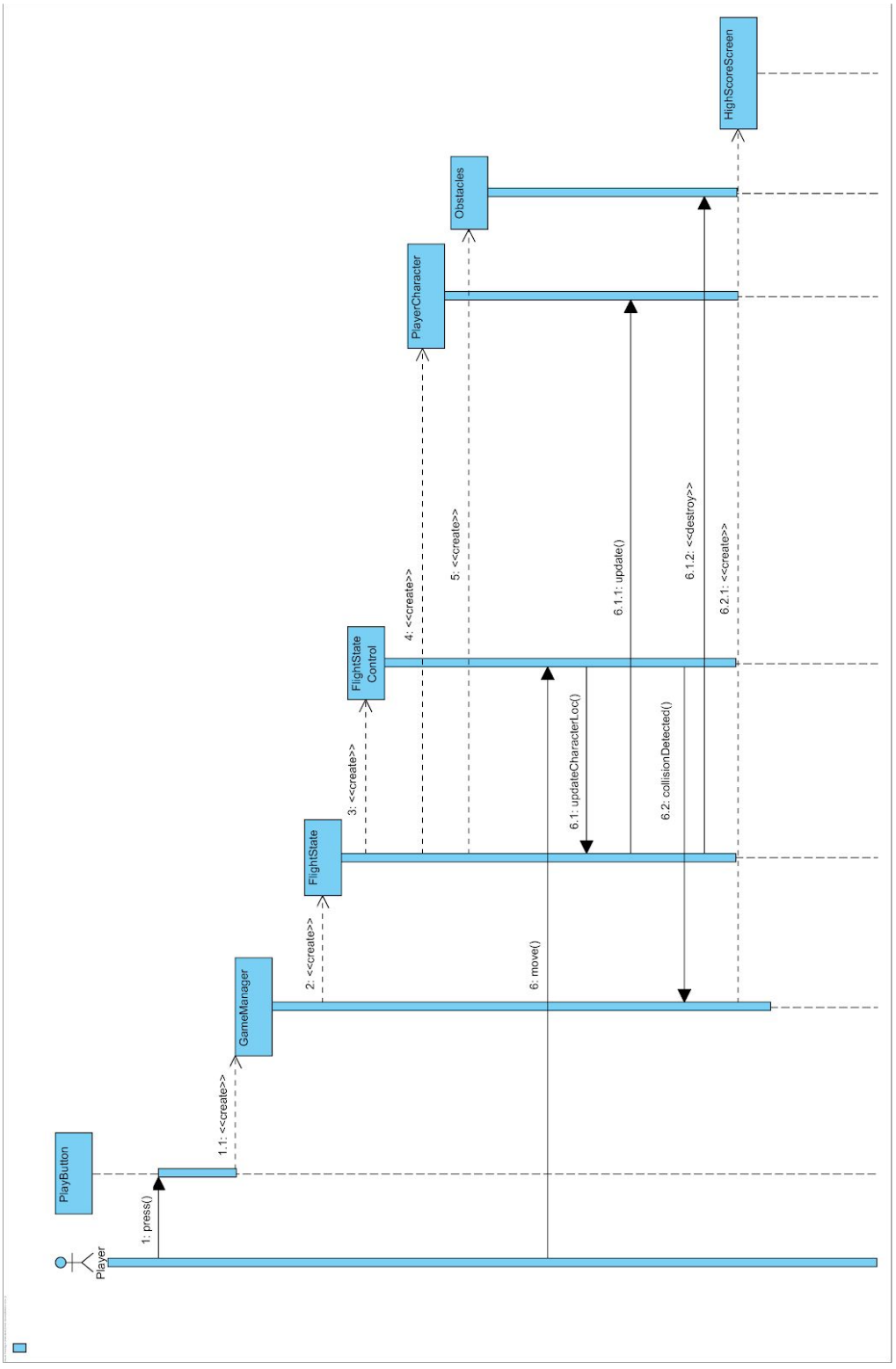

Figure 5.3.1 Illustrates the scenario for Flight Stage

Scenario 1: Playing the Flight Stage

Player presses the play button. The Game is initialized with obstacles and the player character. The player automatically starts running. The player avoids obstacles by moving upwards or downwards. The player collides with an obstacle and dies. Then, high scores screen is shown.

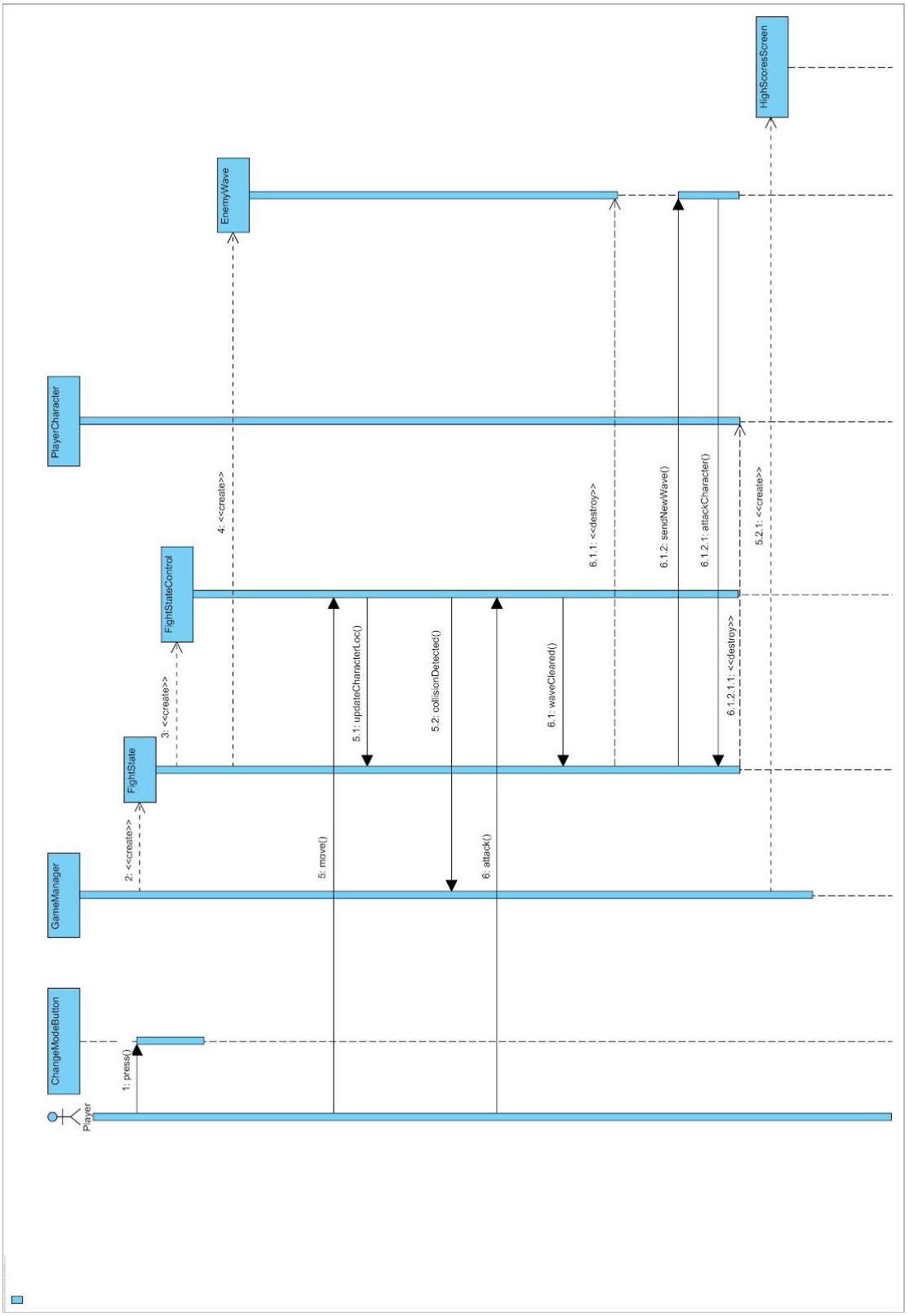

Figure 5.3.2 Illustrates the scenario for Fight Stage

Scenario 2: Playing the Fight Stage

Player is on the flight state. Player presses the Change Mode Button. Fight stage is initialized with the first wave of enemies (enemy waves being a certain number of enemies coming at a time). The player attacks the enemies and wipes out the first wave of enemies. After the first wave is cleared by the player, a second enemy wave containing more enemies is sent. Enemies attack the player. The player runs out of health and dies. Finally, high scores screen is shown.

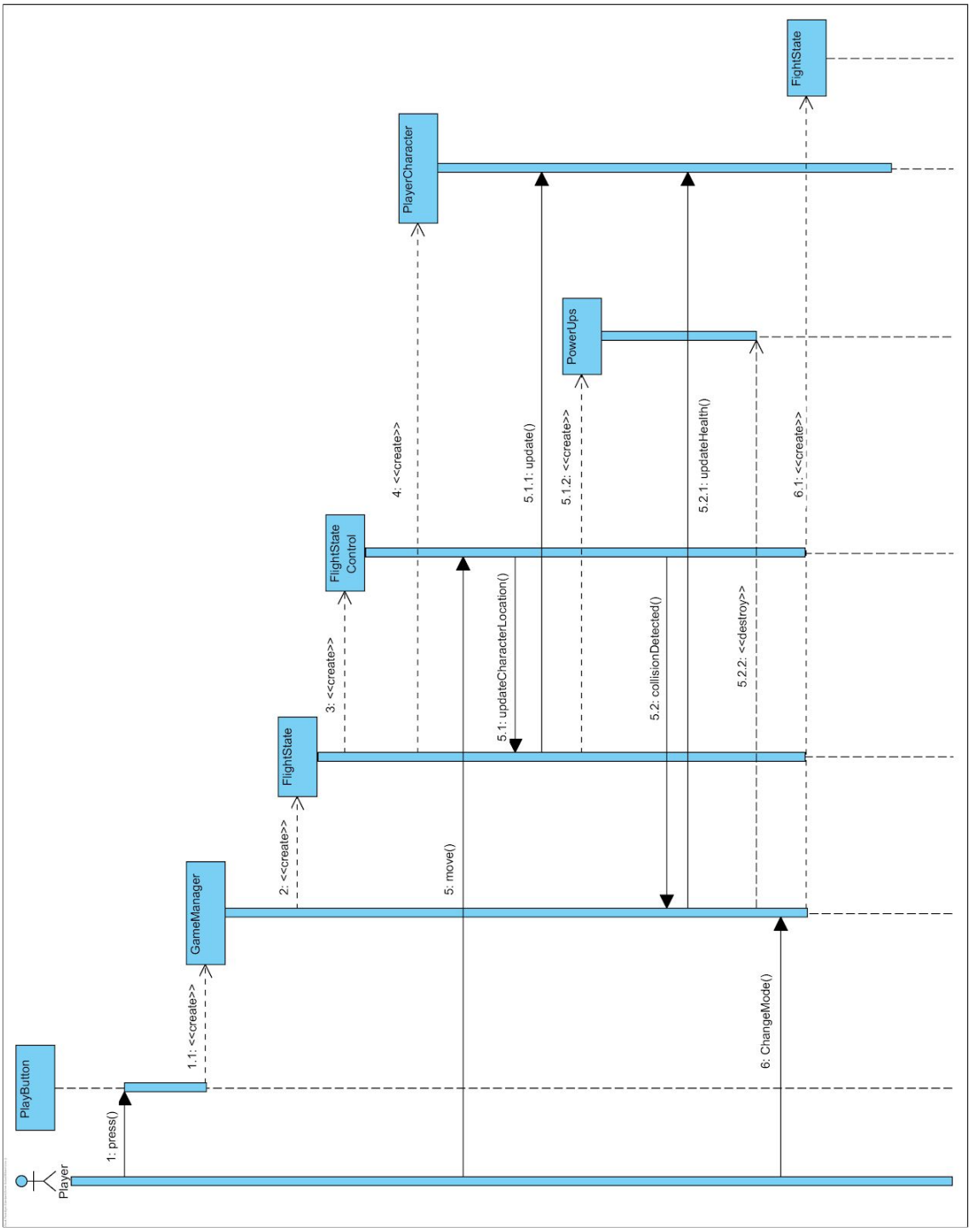

Figure 5.3.3 Illustrates the scenario for collecting powerups and changing game state

Scenario 3: Collecting Powerups and Changing the Mode

Player presses the play button. The Game is initialized with obstacles and the player character. The player automatically starts running. The player moves upwards or downwards to collect power-ups. The player collides with a powerup. The player's health is increased. The player presses the Change Mode Button. Fight stage is initialized with the first wave of enemies.

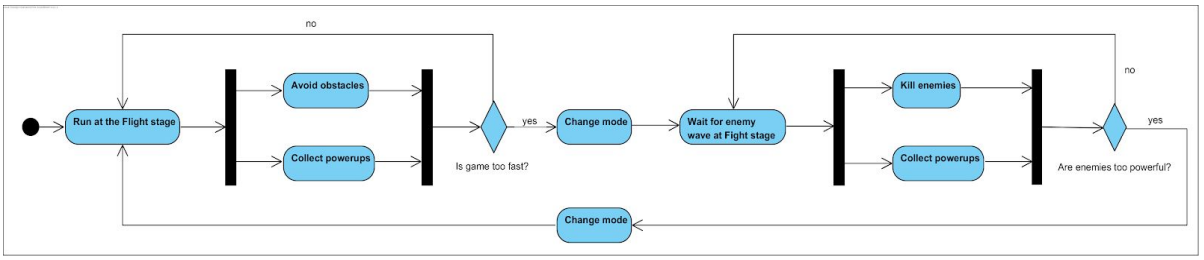

Figure 5.3.4 Activity diagram

This diagram shows the activities and flow of a successful game (i.e. without hitting obstacles or killed by enemies).

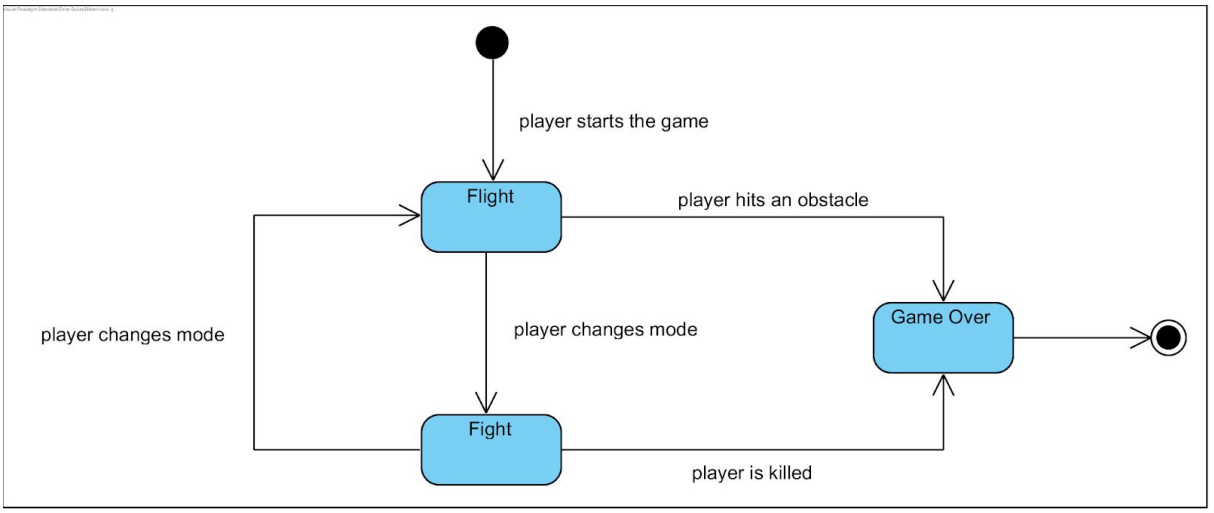

Figure 5.3.5 State transition diagram

This diagram shows the three different states and transition conditions of the game object.

5.4. User interface

This sections includes example mockups from the game.

Main Menu: When the game is executed, this is the first menu that the player encounters. There will be 4 buttons to press: Play, High Scores, Help, Exit.

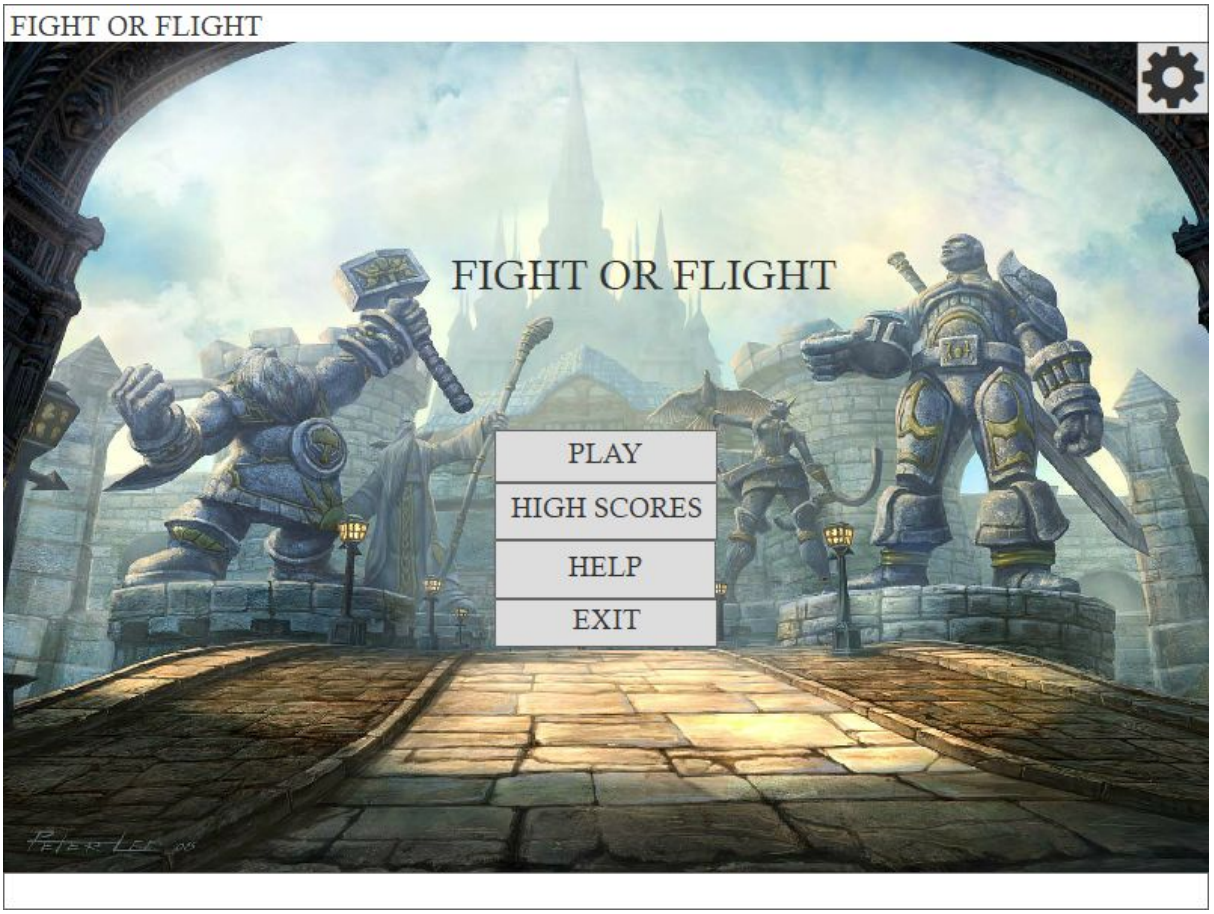

Figure 5.4.1.1: Mockup of the main menu

Single or Multiplayer Selection Screen: The player encounters this frame right after pressing the “Play” button from the main menu. By selecting “Single” or “Multi” the game is initialized with default single player settings or default multiplayer settings.

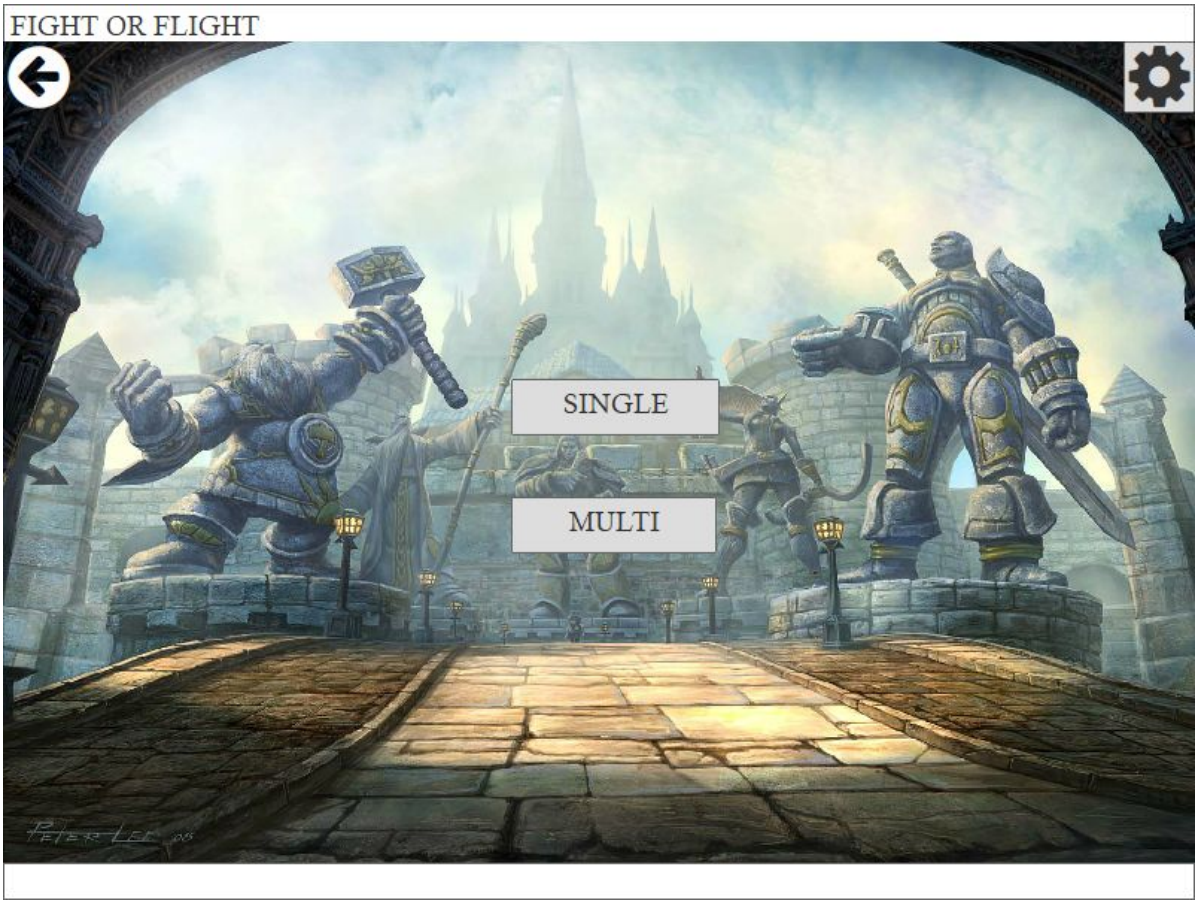

Figure 5.4.1.2: Mockup of the Single or Multiplayer Selection Screen

High Scores: This frame is displayed after the player presses the High Scores button from the main menu or after the game is over. The highest scores are displayed from top to bottom. The top 10 highest scores will be displayed. If the player's score is between this interval, the score will be displayed within the appropriate place.

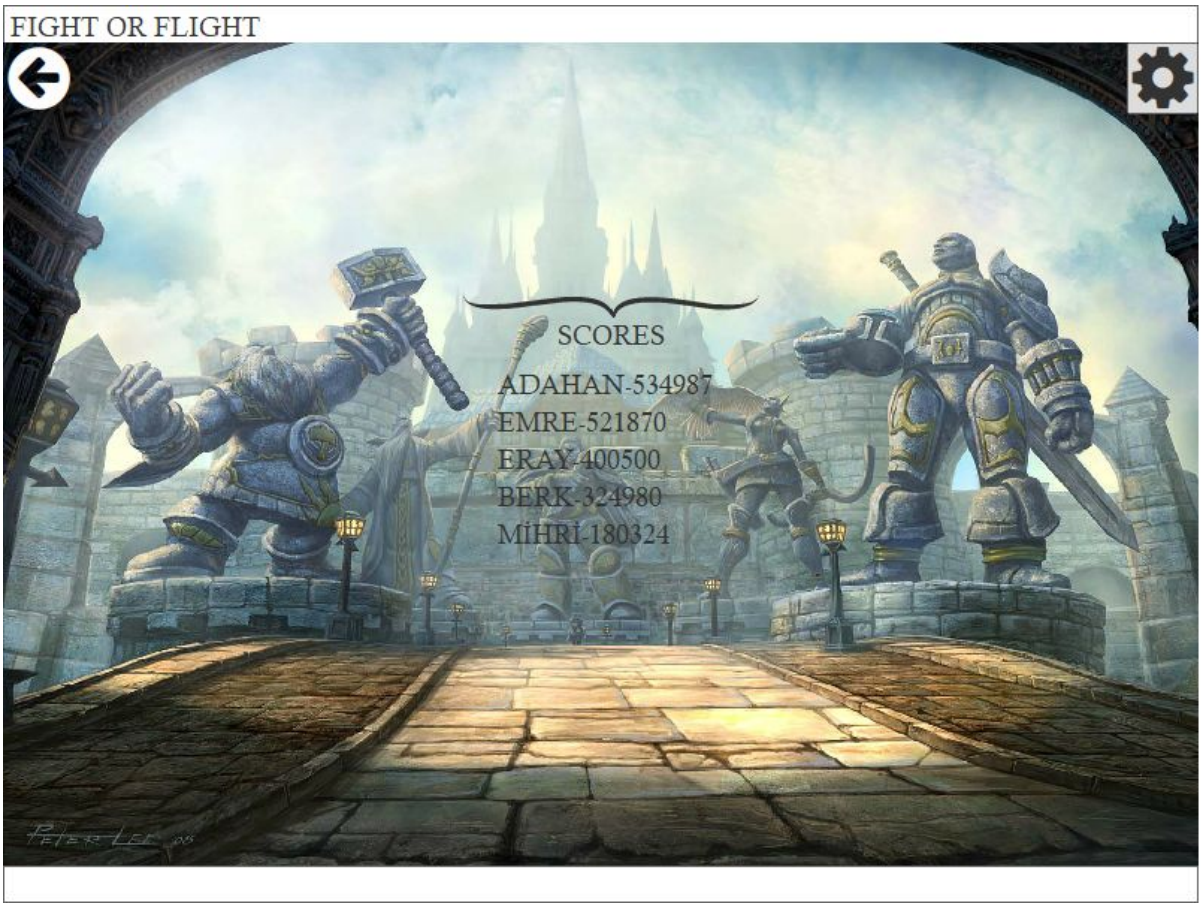

Figure 5.4.1.3: Mockup of the High Scores Screen

Help: The help screen can be reached either by pressing the help button from the main menu or the pause menu. A small manual of the game appears at the center of the screen containing all the essential information about the game. The player can scroll up and down to read this manual.

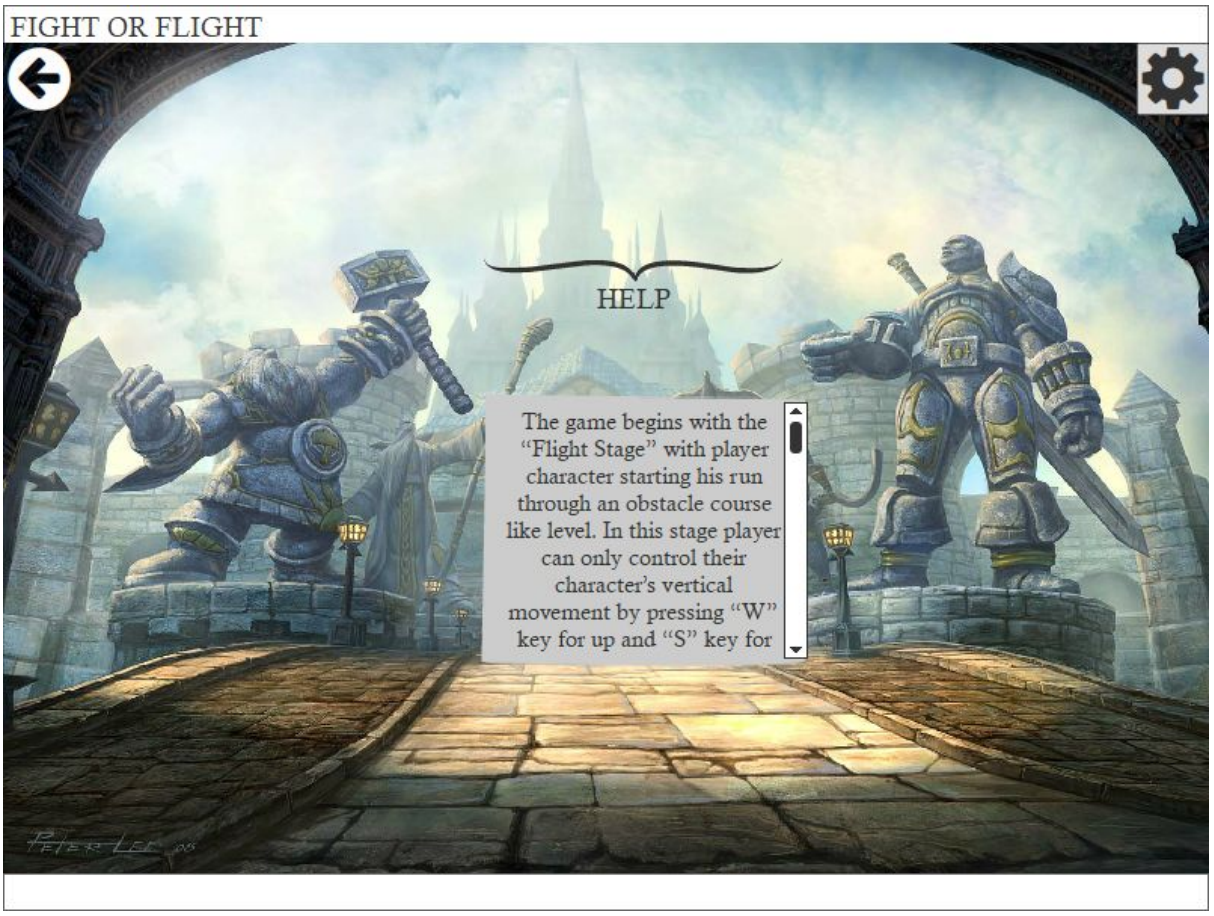

Figure 5.4.1.4: Mockup of the Help screen

Fight State: The mockup below represents the fight state. The section above the game screen shows the attributes of the current level. The character's health is initially 100, when it reaches 0 the game is over. Wave denotes the current difficulty of the game. As the wave number increases, more enemies will appear. Wave also denotes the amount of points gained, if the wave number is high more points are gained. State denotes the current level of the game (whether it is "Fight" or "Flight") and it can be switched only when a wave is over. Score displays the current summation of the points gained throughout the game.

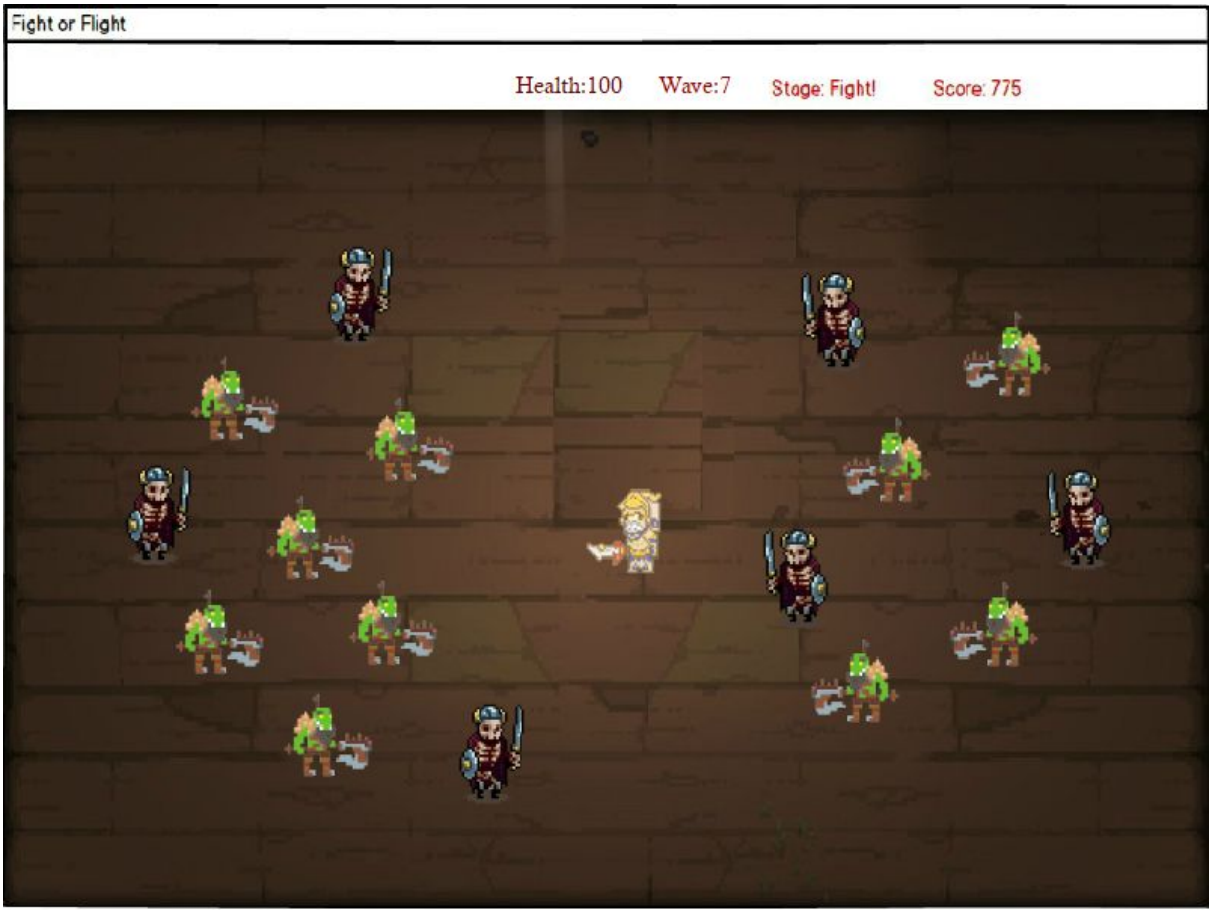

Figure 5.4.1.4: Mockup of the game screen(Fight level)

Flight State: Flight state takes place in between fight states. In this state, goal is to dodge obstacles as long as possible. The longer you stay in flight state, faster the game becomes, making obstacles harder to avoid. Yet a longer flight state also means a larger score multiplier and might also provide bonuses for the fight state, rewarding player for reaching higher difficulties.

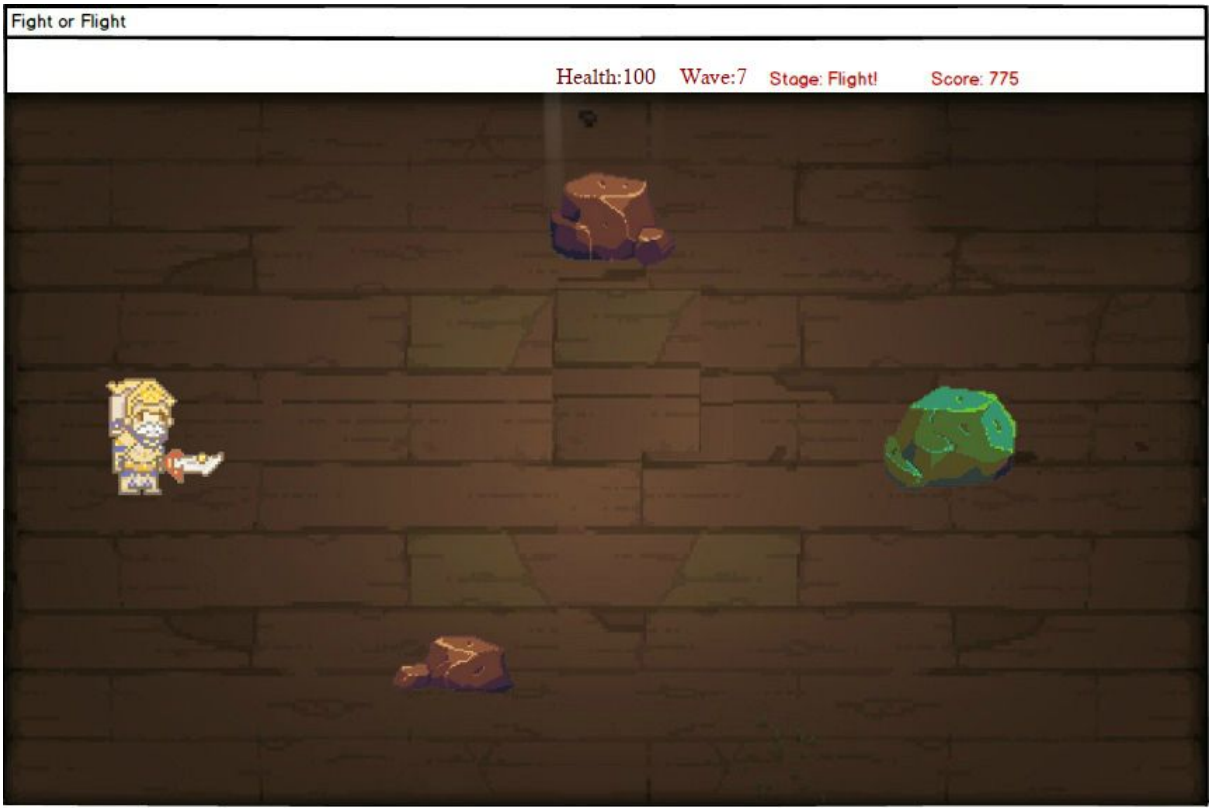

Figure 5.4.1.5: Mockup of the game screen(Flight state)

6. Glossary

- **RPG:** Abbreviation of role playing game. RPG's are games in which players assume the role of a character in a fictional setting.
- **2-D:** Stands for two dimensional, meaning everything is confined to a single plane.
- **Shooter game:** In context of games, shooters are games which employ the act of shooting as a main game mechanic.