Bilkent University

Department of Computer Engineering

Object Oriented Software Engineering Project

CS319 Project Group 1G: Super Mario Bros.

Analysis Report

Ahmet Çandıroğlu, Mert Sezer, Unas Sikandar Butt, Salih Zeki Okur

Course Instructor: Uğur Doğrusöz
Course Assistant: Hasan Balcı

Progress/ Iteration 1
Oct 7 2017

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfilment of the requirements of the Object Oriented Programming course CS319/1

Contents

1 Introduction

Super Mario, first released in 1985 is a game developed by Nintendo. This project will aim to create a game that is very similar to the original Super Mario game, but with some differences.

The game can be categorized as a level based adventure game, with Mario being the main character. The main objective of the game would be to overcome obstacles and reach the end point.

This game will be a desktop application and will be developed using the Java programming language.

2 Game Overview

The objective of the game is to clear as many levels as one can with a limited number of lives. Each level will consist of obstacles, monsters and traps. There will also be bonuses like coins and power-ups. The game will also have a net score of the player. The score will be dependent upon the number of coins collected, the number of monsters killed and the number of lives remaining.

The game will be controlled by just the keyboard. Additionally playing the game is very simple as the user only needs to use the arrow keys and the jump key (spacebar).

2.1 List of Prizes

- **Coin:** Player collects coin to gain points. Coins can be located in *Surprise Bricks* and some of the *Ordinary Bricks*.

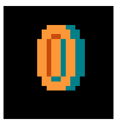

Figure 1: Coin

- **Super Mushroom:** This boost item turns Mario into Super form. Mario can break breakable bricks in this form. Super Mushrooms can be located in *Surprise Brick*s.

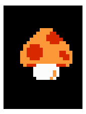
Figure 2: Super Mushroom

- **1-up Mushroom:** This boost item gives 1 extra life to Mario. It can be located in *Surprise Brick*s.

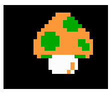
Figure 3: 1-up Mushroom

- **Fire Flower:** This item gives ability to throw fireballs which can kill enemies to Mario. It can be located in *Surprise Brick*s and *Invisible Brick*s which are basically invisible *Surprise Brick*s.

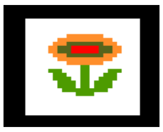
Figure 4: Fire Flower

- **Super Star:** This boost item makes Mario invincible for a short period of time. This item can be located in *Surprise Brick*s and some of the *Ordinary Brick*s.

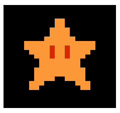
Figure 5: Super Star

2.2 List of Brick Types

- **Surprise Brick:** This bricks always contain a prize inside. Turns into empty unbreakable brick when hit.

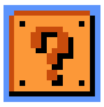
Figure 6: Surprise Brick

- **Ordinary Brick:** This bricks may contain prize but it is uncommon. They are breakable according to Mario's state. It cannot be broken when Mario is in small state.

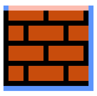
Figure 7: Ordinary Brick

- **Empty Unbreakable Brick:** This bricks do not contain price and cannot be broken.

Figure 8: Empty Unbreakable Brick

2.3 List of Mario's Forms

- **Small Mario:** Mario's initial state. Dies when hits enemy and cannot break bricks.

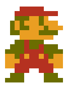
Figure 9: Small Mario

- **Super Mario:** Mario can break in this form. It reverts to *Small Mario* when he hits to enemy.

Figure 10: Super Mario

- **Fire Mario:** Mario can throw fireballs in this form. It reverts to *Small Mario* when he hits to enemy.

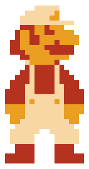
Figure 11: Fire Mario

- **Invincible Mario:** Mario is invincible in this form and he can kill enemies just by touching to them. Only dies when time is over or he falls into pit.

Figure 12: Invincible Mario

2.4 List of Enemies

- **Goompa:** They can move right and left. Dies when stomped on.

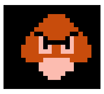
Figure 13: Goompa

- **Koopa Troopa:** They can move right and left. Dies when stomped on and leaves its shell which can be thrown to other enemies to kill by Mario.

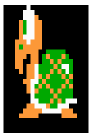
Figure 14: Koopa Troopa

3 Requirements

3.1 Functional Requirements

- User will be able to control Mario:
  - By speeding him up with a constant acceleration to a constant velocity.
  - By making him jump.
  - By making him duck.
- User will be able to break some bricks on the maps by making Mario jump to them.
- User will be able to proceed to right on the map by going right when Mario is at the one thirds of the game screen.
- User will be able to kill monsters.
- User will be able to collect coins:
  - By going over them.
  - By jumping under coin boxes.
- User will have three lives in the beginning.
- Game will be over when user spends all lives
- User will get one life when he/she collects 100 coins.
- User will start a new level when he/she reaches to end of a level.
- User will be able to pause/resume the game.

3.2 Non-Functional Requirements

- Mario icon will have simulations of walking, jumping and ducking.
- Bricks will have breaking simulations.
- Movement of Mario will be under some physics rules which will increase difficulty of game.

3.3 Pseudo Functional Requirements

- Games code will be written in java.

4 System Models

4.1 Use Case Model

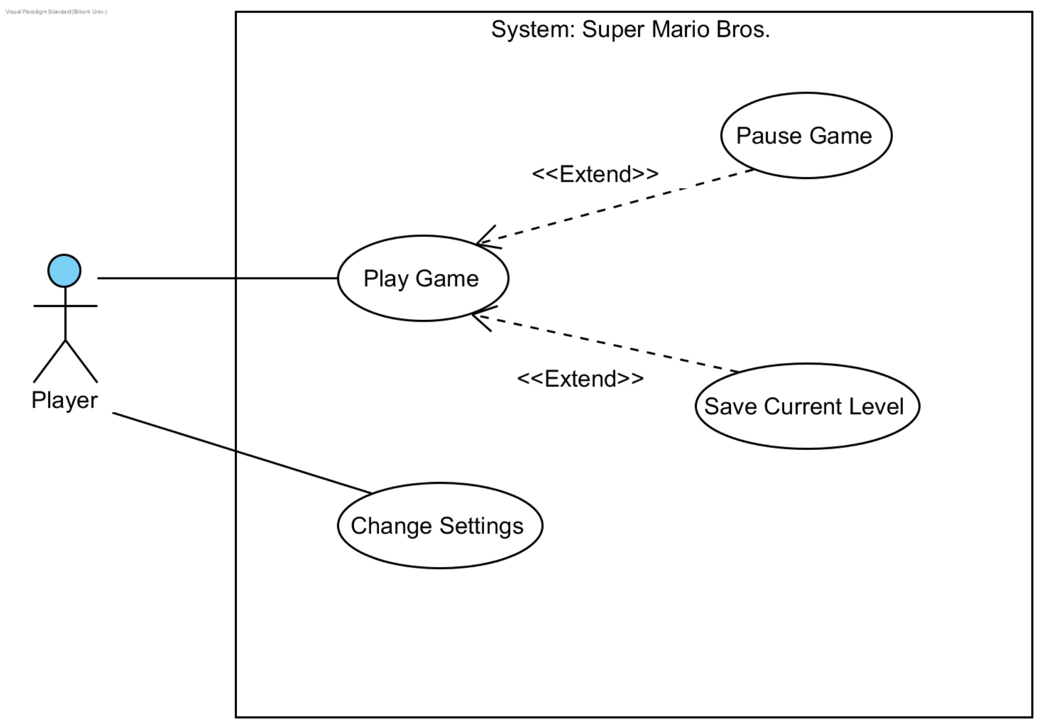

Figure 15: Use case diagram of the game

4.1.1 Use Case Descriptions

Use Case #1

Use Case Name: Play Game

Participating Actors: Player

Entry Condition: Player has opened the game and clicked "Play Game"

Exit Condition:

- Player closed the game
- Player lost all his/her lives
- Player completed all the levels of game

Main Flow of Events:

1. Start game
2. Engine constructs level
3. Player beats the level
4. If all levels are not beaten go to (1.) else display "game over"
5. Display score of player
6. Go to Main Menu

Alternative Flows of Events:

- If player loses all his lives go to (4.)

4.2 Dynamic Models

Scenario Name: Start Game

Scenario: Ahmet clicks on game icon and see main menu in the game. Then he decides to play without any configuration. He clicks on play game and starts to play new level of the game.

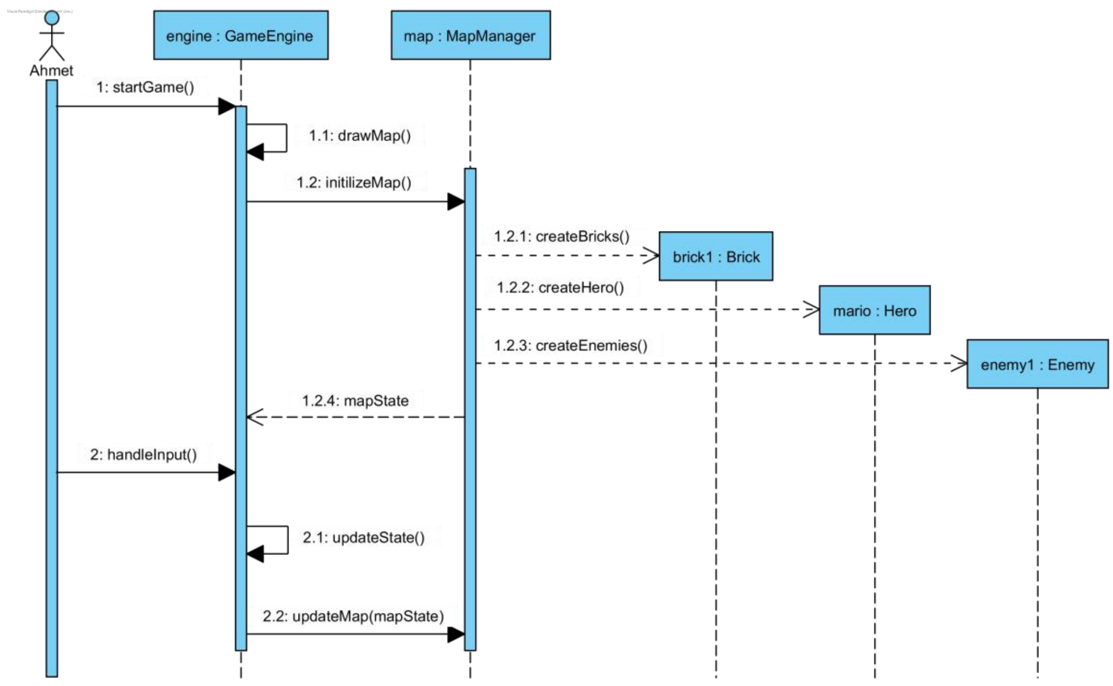

Figure 16: Sequence diagram for initial start of

Description: Ahmet wants to play the game so he starts the game. This process initialized by *startGame()* method of *GameEngine*. Then, *GameEngine*, draws the map, not visually, and calls *initilizeMap(mapState)* method of MapManager to create required classes. *mapState* is the information of bricks, hero, enemies etc. which is gathered from *drawMap()* method. After that, MapManager creates asked components then return *mapState* again to *GameEngine.* From now on Ahmet gives input to *GameEngine*, *GameEngine* updates the state with *updateState()* method and then calls *updateMap(mapState)* method which changes view.

Scenario Name: Eating Super Mushroom

Scenario: Ahmet plays Super Mario Bros. with great joy. He is trying to get coins and mushrooms by hitting to *SurpriseBrick*s. He opens a brick which contains *SuperMushroom*. He eats it and his Mario becomes super.

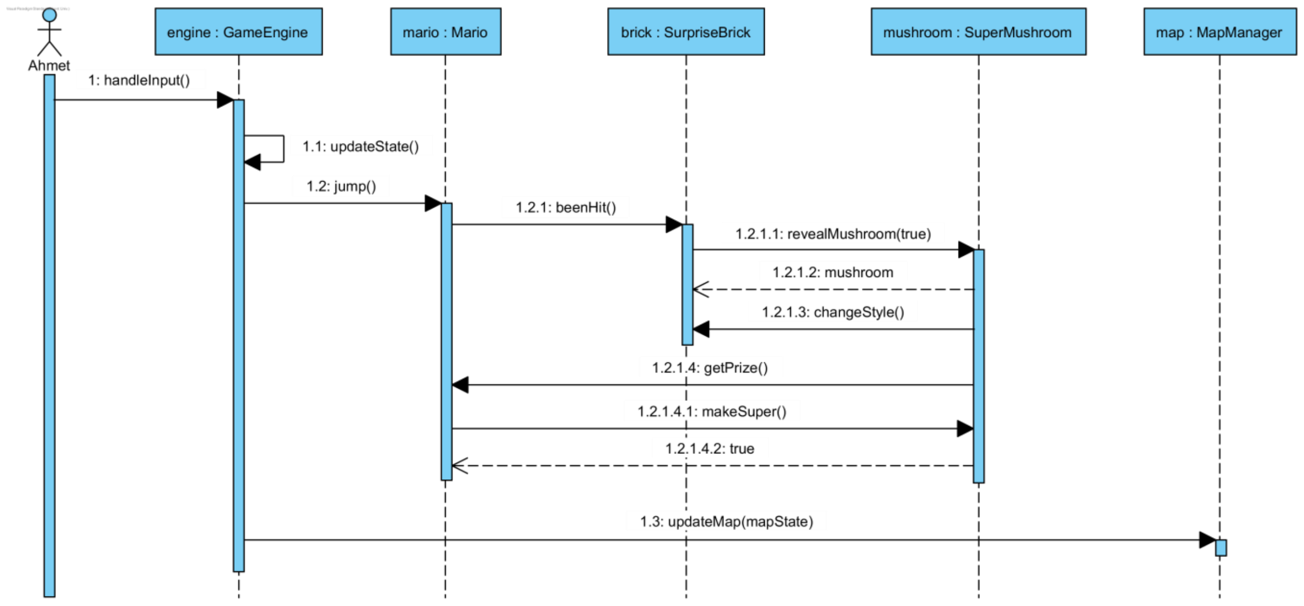

Figure 17: Sequence diagram of eating super mushroom

Description: Ahmet jumps under a *SurpriseBrick* which contains a *SuperMushroom*. *GameEngine* updates the state after getting input and then calls *jump()* method of Mario. Mario calls *beenHit()* function of *SurpriseBrick* which leads to mushroom to be revealed. Then, Mario calls *makeSuper()* function of mushroom by eating it. After all happened finally MapManager is updated and so view.

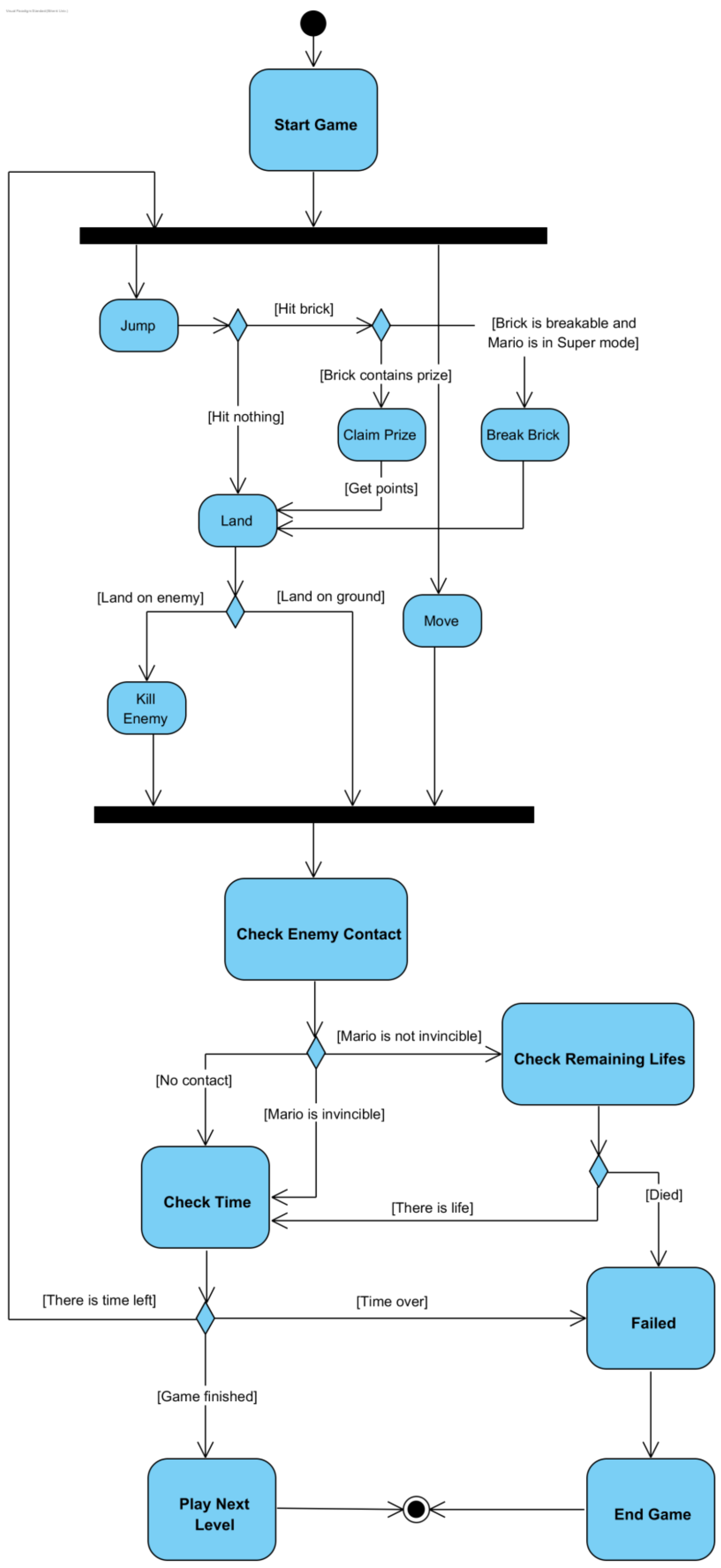

Figure 18: Activity diagram of the game

As it is shown in activity diagram there are lots of checking and decision making during gameplay. After each move or jump of the player, *GameEngine* needs to check several conditions. Player needs to reach end of the map within given time so after each move *GameEngine* checks time. Also, there are enemies on the ground so *GameEngine* also check for possible enemy contact and changes state accordingly. If player reaches final checkpoint without wasting every life within given time, then he/she can go to next level which is a different map or exit the game. Points will be gained by killing enemies and obtaining prizes.

4.3 Object and Class Model

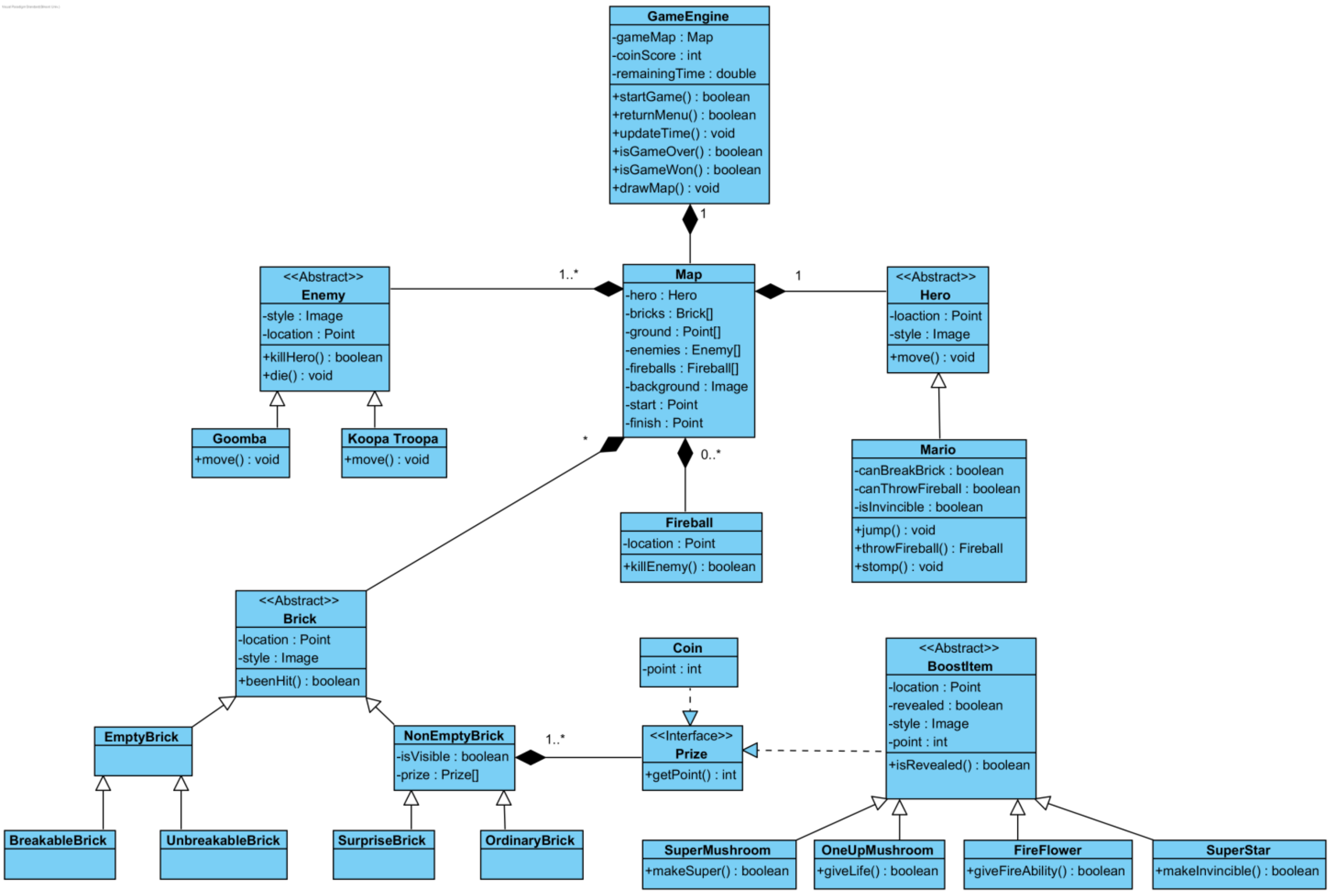

Figure 19: UML class diagram of the game

GameEngine is the main class which controls the game according to state of the actors in map and inputs received from player. *Map* class contains everything related to map and draws to the screen. It is only a visual container, it does not control anything. Hero class represents the possible characters player can control during the game. *BoostItem* changes state of the Hero, it makes Hero stronger. *Prize*, *Brick*, *Hero* classes are model classes where as *GameEngine* is controller and *Map* is view class.

4.4 User Interface and Screen Mock-ups

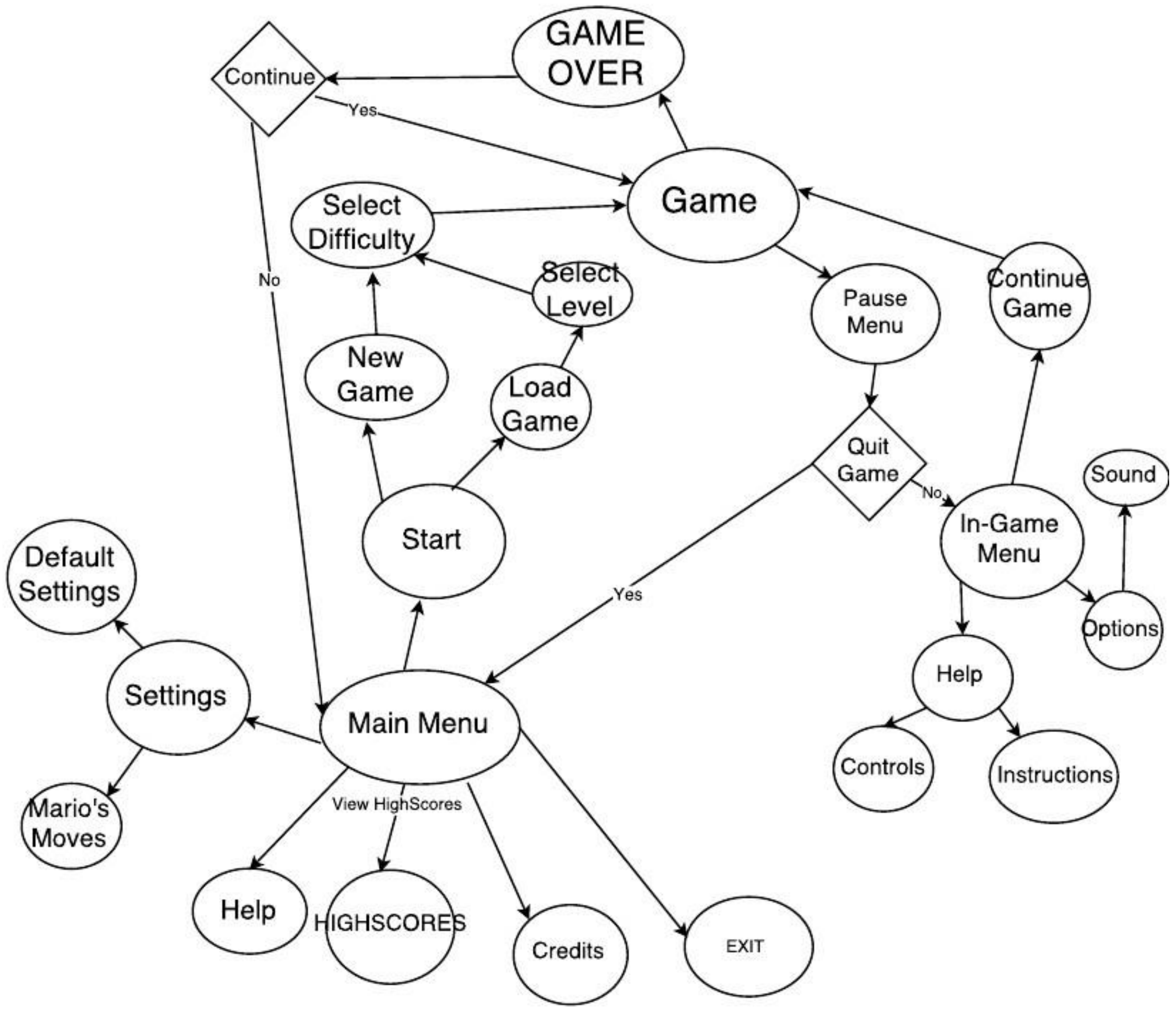

Figure 20: Navigational path diagram

Load Game: After player selects “Start Game” from the main menu, if player does not select “New Game” and selects “Load Game”, player can select any game from the previous levels; which are shown in red buttons. Player will not be able to play a game from the levels; which are shown in black buttons; because these are the levels; which the player hasn’t accomplished to pass before.

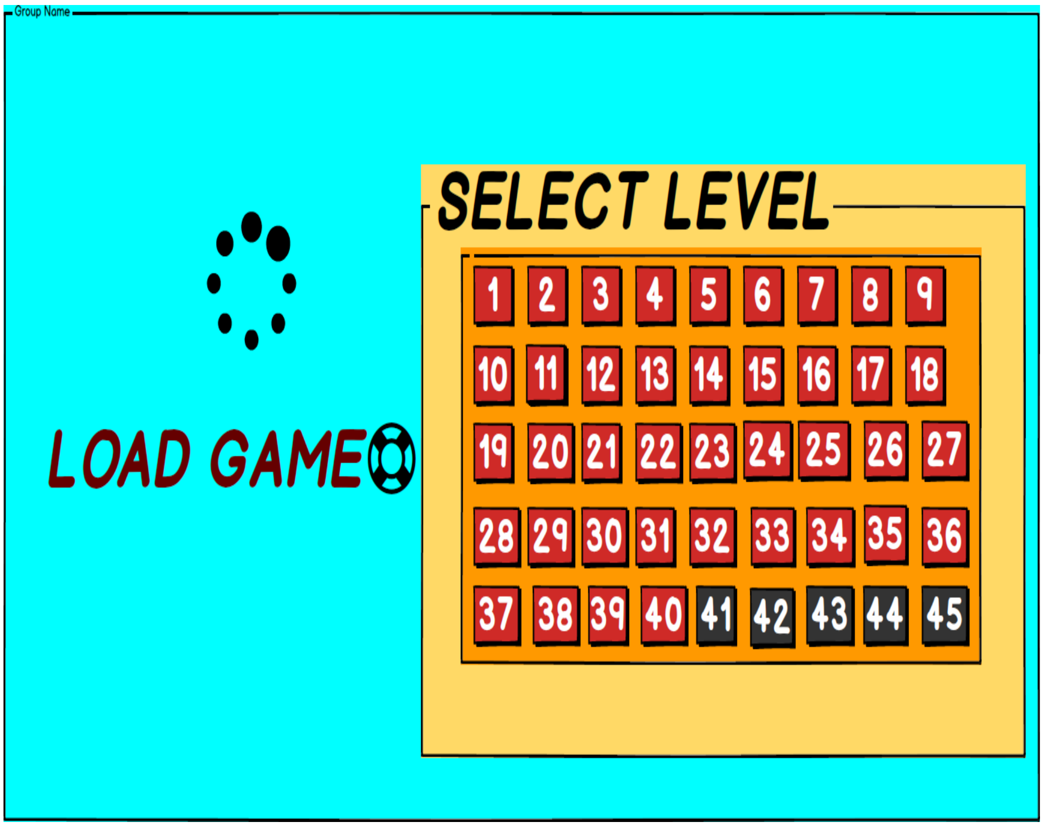

Figure 21: Load game interface

Pause Menu: If the player press the pause button; which is in the top right corner of the gameplay screen, this player encounters with the pause menu; which has 2 options.

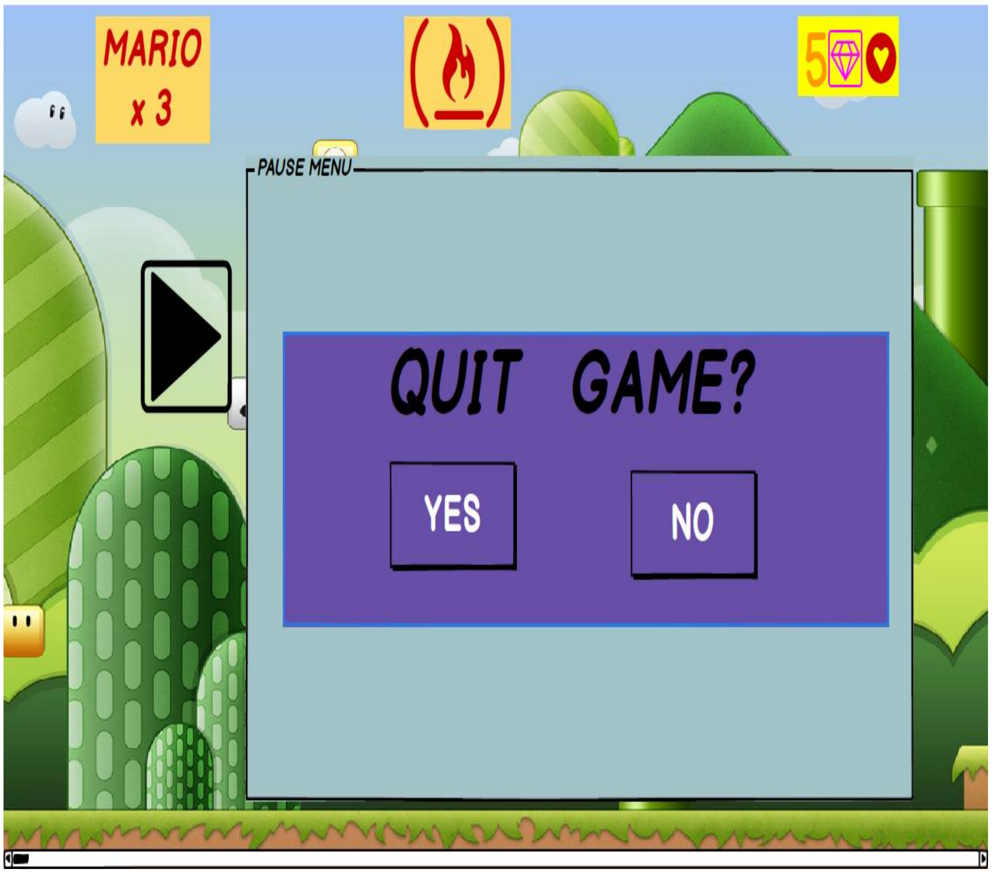

Figure 22: Pause menu interface

In-Game Menu: If the player press the “NO” button in the pause menu; this player encounters with 3 options.

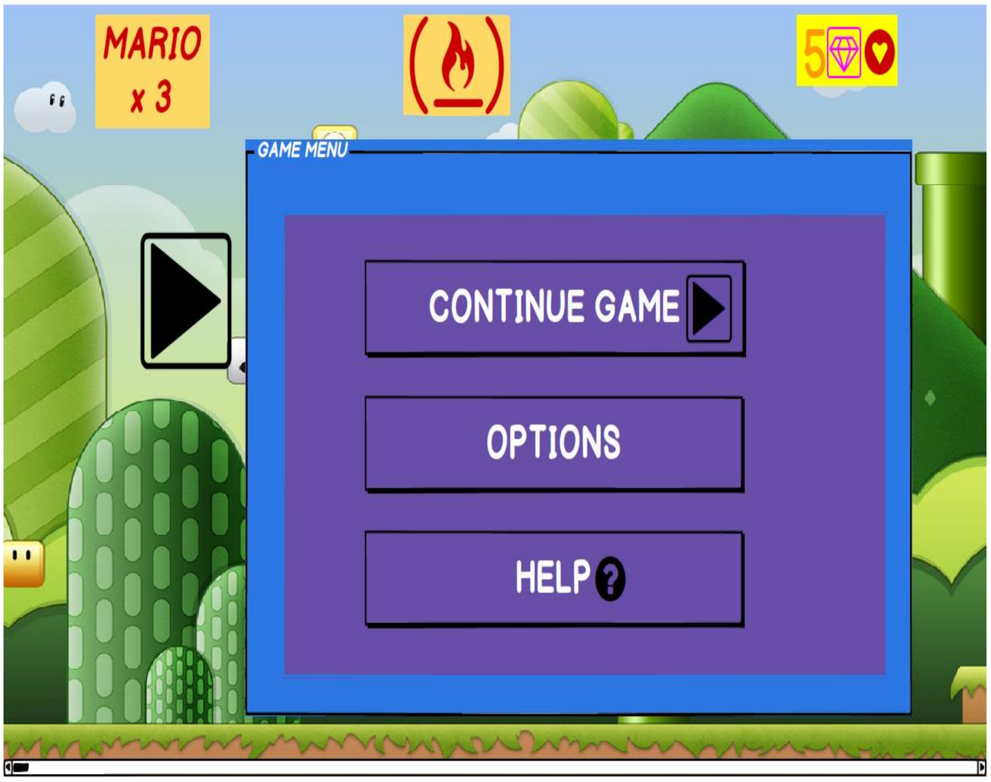

Figure 23: In-game menu interface

Help Menu

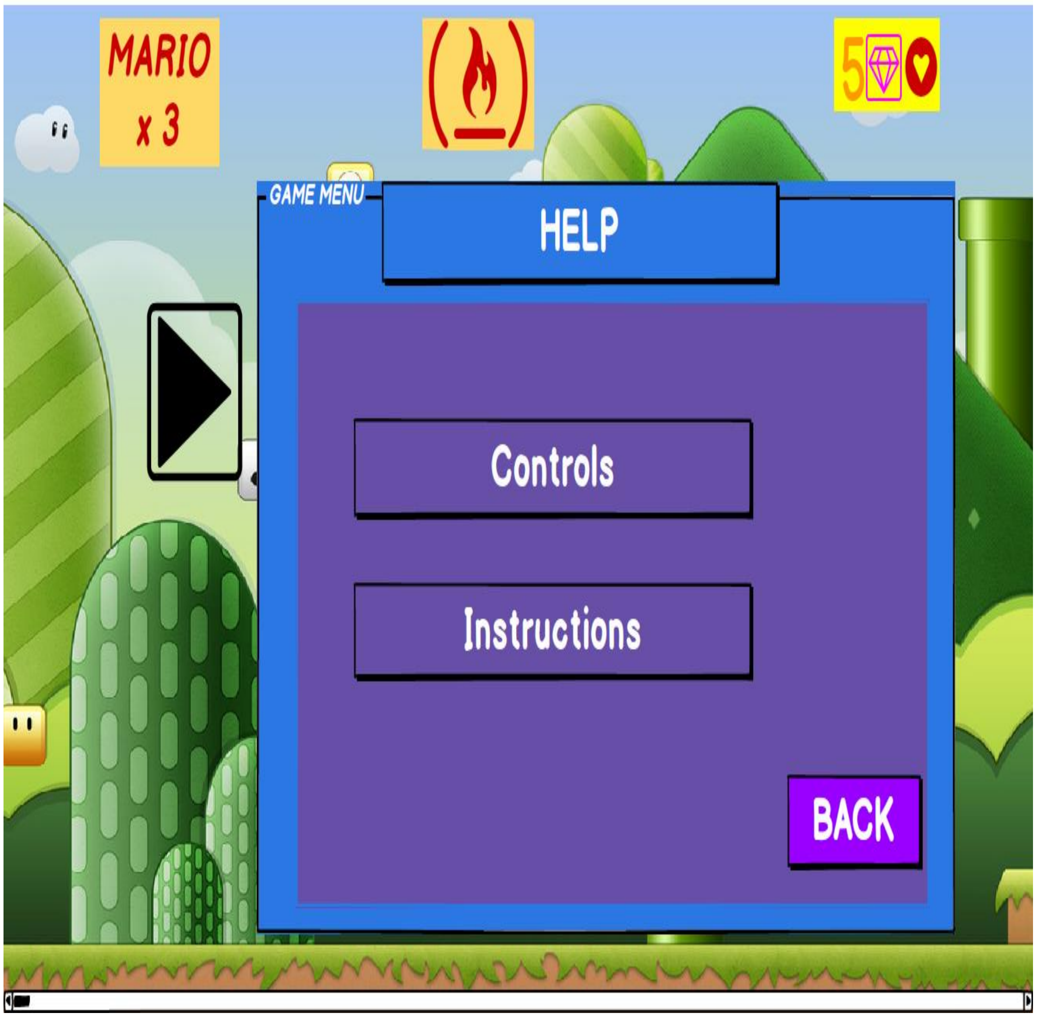

Figure 24: Help menu interface

Control Menu: If the player selects "Controls" button in the help menu, he/she can see which keyboard keys should be used to move, run, jump.

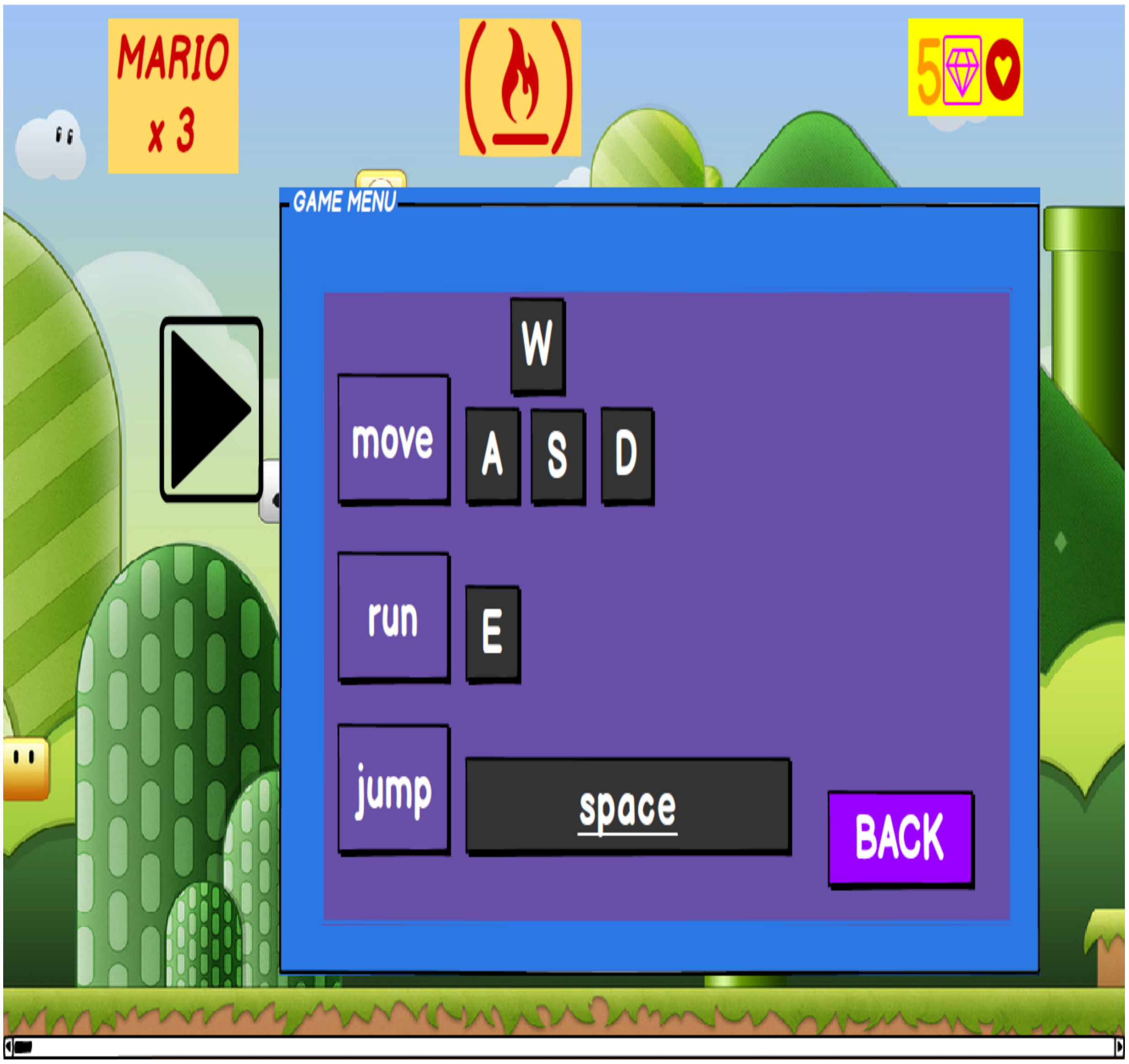

Figure 25: Control menu interface

5 References

https://www.mariowiki.com/Super_Mario_Bros.

https://www.mariowiki.com/Coin

https://www.mariowiki.com/Super_Mushroom

https://www.mariowiki.com/Fire_Flower

https://www.mariowiki.com/Super_Star

https://www.mariowiki.com/1-Up_Mushroom

https://www.mariowiki.com/Small_Mario

https://www.mariowiki.com/Super_Mario

https://www.mariowiki.com/Fire_Mario

https://www.mariowiki.com/Invincible_Mario

http://www.mariouniverse.com/maps/nes/smb