The Little Plumber

Project – Release 1

CIS 350

By Quinn Gairson, Michael Gairson

07/19/2021

**Project Description**

This project represents our interpretation of the original Mario Bros gam released in 1983. A spin off the original Mario Bros., was created by Nintendo for each major platform release. The Super Mario games are typically set in the fictional world of mushrooms with Mario as the character in play. This project utilized the Mario concept with the player in control of a screen character that advances through a map facing challenges that can assist or hinder the play.

**Features Implemented**

* Creation of additional levels
* Further refine level design and layout
* Multiple levels linked together in secession
* Ability to control characters movement (forward, back and up)
* Create special abilities for main character
* Abilities based upon the environment of the level
* Have unique attack talents against opponents
* Create multiple opponents are varying skills
* Create coins that can be collected by the player that effects score
* Timer/Count down for game and abilities
* A visual representation of player’s score
* Background music based upon level and skill awards

**Sample Screenshots**

**Movement Right**

Timeline

Description automatically generated with medium confidence

**Movement Left**

Timeline

Description automatically generated with low confidence

**Movement Up**

Timeline

Description automatically generated

**Collision**

Timeline

Description automatically generated with medium confidence

**Collision**

Timeline

Description automatically generated with medium confidence

**Pipe Collision**

**Use Case Diagram**

Diagram

Description automatically generated

**Use Case Descriptions**

|  |  |
| --- | --- |
| **Name** | Movement |
| **ID** | UC2 |
| **Brief Description** | Allow the player to direct the character |
| **Actors (primary and supporting/secondary)** | Current Player interacts assuming the role of the character |
| **Triggers** | The player uses the appropriate arrow keys (Left – Right) to direct the character movement left or right. The space bar is used to direct the character to jump. |
| **Preconditions** | The game must be started and in play |
| **Primary Flow** | 1. Left arrow is pressed causing the character to move left.  2. Right arrow is pressed causing the character to move right  3. Space bar is pressed causing the character to jump. |
| **Alternate Flows** | The player doesn’t press action keys and stays stationary   1. Possible collision with evil mushroom causing a loss in end of game. 2. Player remains in stationary |
| **Minimal Guarantees** | Player will have successfully moved in a given direction of their choosing |
| **Success Guarantees** | Playing will successfully navigate level |

|  |  |
| --- | --- |
| **Name** | Collision |
| **ID** | UC3 |
| **Brief Description** | Allows player to interact with level objects |
| **Actors (primary and supporting/secondary)** | Current player that is in control of the character. |
| **Triggers** | The player either runs into or is otherwise involved in a collision |
| **Preconditions** | The game level must first be started |
| **Primary Flow** | 1. The player collides on top on of an enemy which results in the elimination of the enemy.  2. The player collides with a “pipe” whereby causing a change in level |
| **Alternate Flows** | The player is involved in a collision however the collision was not invoked by the player   1. A enemy collides with the player causing end of game. |
| **Minimal Guarantees** | Player will have a collision that results in the end of game |
| **Success Guarantees** | Player will initiate a collision resulting in the elimination of an enemy or the change in a level. |

|  |  |
| --- | --- |
| **Name** | Create Game |
| **ID** | UC1 |
| **Brief Description** | Allows player to interact with level objects |
| **Actors (primary and supporting/secondary)** | Current player controlling the character |
| **Triggers** | The player either runs into or is ran into |
| **Preconditions** | The game must be started |
| **Primary Flow** | Welcome screen – game start, single player |
| **Alternate Flows** | Player ends game – game fails to start |
| **Minimal Guarantees** | Game start |
| **Success Guarantees** | Game starts player has character movement, collision involving enemy players and transitional pipe |

|  |  |
| --- | --- |
| **Name** | Network |
| **ID** | UC4 |
| **Brief Description** | Allows connection to remote service that user can access web application from |
| **Actors (primary and supporting/secondary)** | Server |
| **Triggers** | The user starts game |
| **Preconditions** | The game must be available for play |
| **Primary Flow** | Player has network availability to the game |
| **Alternate Flows** | Player must use a static or local copy of the game |
| **Minimal Guarantees** | Remote access available for start of game |
| **Success Guarantees** | Player has full control of game |

**UML Diagram**

**Coding Standards Report/ Static Code Analyzer (Both generated using Eslint)**

The warnings “const”, and “arrow” that appear are not due to the code having errors. Instead, they just represent coding constraints with respect to what version of JavaScript is being utilized. With the “gameLevel” error, gameLevel is used just to start the game and is not required to be called again. Exceeding the 80 lines of length helps with coding and the readably of the code.

Text

Description automatically generated

**Static Code Analyzer (Google V8)**

Code coverage was only 75.6% usage due to not fully utilizing the implemented library.

User generated code was fully utilized.

Graphical user interface, text, application

Description automatically generated

**GitHub Repository**

https://github.com/DirtyWhiskers/CPS350.git

**Roles and Responsibilities**

Quinn: Responsible for setting up the server and creating collision among the player and environment. Also worked on Eslint for code analyzer and google V8 for code coverage.

Michael: Responsible for creating environment and character movement. Also worked on Eslint for code analyzer.

**Self-reflection**

Quinn: This was the first time that I’ve made a game using JavaScript. This project required a lot of trial and error since I not very familiar with this particular programming language. The use of the kaboom library was a great aid in creating levels, character design and key mapping, but also presented its fair share of problems. Due to the how new the kaboom library is there is very little documentation or examples available for reference. Finding a static code analyzer that was easy to use and did not require multiple plugins was also a challenge. Overall, this was a great learning experience that built upon many different areas of software design.

Michael: