**Assignment 02**

**Requirement Document**

Braeden Timlin

101005580

IMD 3006A

Revision History

|  |  |  |
| --- | --- | --- |
| Version | Description | Approval |
| 1 | First Draft |  |
| 2 | Revision 1 |  |
| 3 | Final |  |

Scope of Document

This document describes the objectives and requirements of the visual based game I created.

Description of the Program

The game allows the player to move using the arrow keys. They must pick up all the power ups on the screen without getting hit by the enemies that are moving randomly on screen. If the user gets hit by an enemy they lose, but if they collect all the power ups then they win.

Due to limited skills with JavaScript programming, the development style of the code became very different than originally intended in Revision 1 of this report.

Objectives of the Project

The objectives of this project are to demonstrate my abilities to use Object Oriented programming methodologies to create a simple graphics-based game. The game will consist of a user-controlled player object, 10 enemies that move in random directions, and 15 collectible items that the user needs to pick up to win.

Functional Requirements

1. User Input:
   * The user will use the keyboard arrow keys to control the player entity in the game
2. Game Objects
   * Player
     + Will be using a single sprite of Link from the original Legend of Zelda. Typical users would associate this character with the player character.
     + User keyboard input will move the character around the screen.
   * Enemies
     + Will use a single sprite of an Octorok from the original Legend of Zelda. This image is associated with enemies typically.
     + Getting hit by an enemy will cause the user to lose the game.
     + These entities will move at about the same speed as the player in straight lines, changing directions to a different random direction of UP, DOWN, LEFT, or Right every 5 seconds.
   * Pick-Ups
     + The pick-ups will all use the sprite of a Rupee from the original Legend of Zelda. This is associated with currency and points to most users.
     + Collecting all of these will win the game.
3. Data Structures (classes & functions)
   * **Char** class
     + Integer iX variable
       - X Position
     + Integer iY variable
       - Y Position
     + Image Sprite variable
       - The image that the game will render for the entity
     + Integer iSpd variable
       - The number of pixels that the entity will translate each frame
     + Integer direction variable
       - 0-3 corresponds to the directions up, left, down, and right respectively
         * 0: up
         * 1: left
         * 2: down
         * 3: right
     + Boolean draw
       - Dictates whether the object is active or not
   * enemyMove function
     + Checks if the gameRun variable is true, does not run function if it is false
     + Runs through all enemies and checks each of their iDir values, then increments/decrements their iX and iY values corresponding to which direction they are going
     + The function also changes the enemy’s direction if they hit the edges of the screen
     + Runs collision function between player and all enemies, then runs gameCheck function to check game end parameters
   * Collision function
     + Checks the distance between 2 entities and returns true or false based on whether the distance between them is smaller or larger than 25 pixels using Pythagorean theorem
   * Move function
     + Increments or decrements the player’s iX and iY values based on the key input and player’s speed
     + Checks iX and iY position plus projected movement to ensure player does not move out of the viewable frame
   * setDir function
     + Sets the dir variable to a random Number between UP and RIGHT constants (0, 3)
   * RandInt function
     + Returns a random integer between the provided min and max values
   * Init function
     + Initializes objects using constructor functions
       - Player
       - Array of 10 enemies
       - Array of 15 pick ups
     + Initializes canvas
     + Set render framerate
     + Set interval for enemy changing direction
     + Set interval for enemy movement, set to same rate as game framerate
     + Create event listener for keypress, call KeyInput when event occurs
   * Draw function
     + Clears canvas so sprites do not create a trail
     + Draw all the active pickups
     + Draw the player
     + Draw all enemies
   * Update function
     + Check collision between the player and all pick ups
     + If the pickup is active and the collision returned true
       - Set the current pickup’s draw value to false, deactivating it
       - Increment pnts by 1
     + Set the content of the points div in the HTML to be the number in the pnts variable after the points increment so when the user picks up all the rupees it will show 15 points
   * gameCheck function, take parameter (hit)
     + Checks if hit variable is true
       - If true
         * Sets gameRun to false
         * Sets gameWin to false
         * Turn div of id gameStatus to “You Lose” message
       - Checks if pnts is greater than or equal to NUM\_PICKUPS
         * If true, the game ends with a “You Win” message

Sets gameRun to false

Sets gameWin to true

* + - * + If false, the game continues

1. Data Modules (Variables and Constants)
   * Constant NUM\_PICKUPS set to 15
     + Total number of pick-ups in the game
   * Constant NUM\_ENEMIES to 10
     + Total number of enemies in the game
   * Constant UP to 0
     + Used for enemy movement to make which direction more clear
   * Constant LEFT to 1
     + Used for enemy movement to make which direction more clear
   * Constant DOWN to 2
     + Used for enemy movement to make which direction more clear
   * Constant RIGHT to 3
     + Used for enemy movement to make which direction more clear
   * Variable c
     + Used for canvas
   * Variable ctx
     + Used for canvas
   * Variable player
     + Uninitialized player object
   * Array enemies
     + Uninitialized enemies array of object
   * Array pickUps
     + Uninitialized pickUps array of objects
   * Variable pnts
     + Used to keep track of points in game
   * Var gameRun
     + Used to indicate that the game is active
     + Default at true
   * Var gameWin
     + Used to determine whether the player has won or lost
     + Used in tandom with gameRun
     + Defaults to false
2. Multimedia Components
   * The program will support .PNG type images
   * There will be no audio