Software

Requirements

Specification

Low Rez Studios

Version 1.0

**Prepared by:**

Joseph Lipinski

Sarah Kushner

Zachary Lopez

Matthew Freihofer

Patrick Nwanah

Cory Cellucci

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Date | Comment | Version |
| All | January 15, 2016 | Initial Version | 1.0 |

Table of Contents

[1. Introduction](#h.5cydk1ow3xg6)

[1.1 Overview and Scope](#h.ixlsz9f7ajzd)

[2. Project Description](#h.aou1jb3tnrq0)

[2.1 User Profile](#h.fgtwvjo2ah92)

[2.2 Assumptions and Dependencies](#h.w10ixo3s1t0s)

[3. Requirements](#h.krdp4x60ddu6)

[3.1 Requirements Apportioning](#h.6oj09i9qfdu9)

[3.2 Functional Requirements](#h.2cu3fhlibz5l)

[3.3 Non-Functional Requirements](#h.uzcdp3rtnsle)

[4. Use Cases](#h.gq28usj1n027)

[4.1 Use Case Flow](#h.b7zhi0rn16s1)

[4.1.1 Start Screen](#h.z6cu878m55i4)

[4.1.2 Start Game](#h.nxy013v66qjl)

[4.1.3 Play Game](#h.e6hxgsxdktdv)

[4.1.4 Defeating an AI enemy](#h.rbu8q46grdsn)

[4.1.5 Losing to an AI enemy](#h.agfrfibe0tan)

[4.1.6 Help](#h.lapdoq7wa3ig)

[5. System Evolution](#h.dnbkyy89j1c5)

[6. Appendix](#h.xujnpyallcig)

[6.1 Glossary](#h.bnv7i0m7a8g0)

# 1. Introduction

## **1.1 Overview and Scope**

Low Rez studios has come together in order to produce a game centered around a robot fighting for his freedom. Armed with lasers and a jetpack, you control this robot as he escapes from a laboratory and fights off other creations like him in order to be free. This game is a 2D sidescroller with both shooting and platforming elements. We have created five levels with hidden areas to find powerups as well as enemies with intelligent tracking of the player, and pathing of the terrain around them. The game will also have a physics engine coded by us as opposed to using a library. We will be using Pygame as our game engine in order to give us more control over the aspects of the game.

# 2. Project Description

## **2.1 User Profile**

There is only one type of user for the software, gamers. Users will play each level of the game in order to win.

## **2.2 Assumptions and Dependencies**

Players must have Pygame 3.2 and Python 3.2 installed to run the game on their personal systems.

# 

# 

# 3. Requirements

## **3.1 Requirements Apportioning**

|  |  |
| --- | --- |
| **Priority Level** | **Description** |
| 1 | These are requirements which are critical to the core functionality of the program. These must be rigorously tested before the software is released. |
| 2 | These are non-vital requirements but are desirable to enhance the user experience. These are expected to be satisfied provided there is extra time to do so. |
| 3 | These are requirements which are outside the scope of the project for this term. These are appropriate for continuing work on the software in the future.. |

## **3.2 Functional Requirements**

R1 Player

* R1.1 Movement
  + R1.1.1 The player's movement will be controlled by input from the keyboard or from a controller in order to move about the terrain and fire at enemies. **Priority 1**
* R1.2 Actions
  + R1.2.1 The player will be able to fire at enemies. **Priority 1**
  + R1.2.2 The player will be able to jump around the terrain using the jetpack. **Priority 2**
  + R1.2.3 The player will also gain powerups that allow the user to do more actions (please see the powerups section for more details) **Priority 2**
* R1.3 Animation
  + R1.3.1 The players movements from turning left and right will be animated **Priority 1**
  + R1.3.2 The jump movement will be animated. **Priority 1**
  + R1.3.3 The player firing will be animated. **Priority 1**
  + R1.3.4 The player dying will be animated. **Priority 1**

R2 Physics Engine

* R2.1 Gravity
  + R2.1.1 Gravity will be coded so that enemies and the player will fall after jumping in the air. **Priority 1**
  + R2.1.2 Enemies will also fall and not float after they walk off a block or off the terrain.**Priority 1**
* R2.2 Collisions
  + R2.2.1 When the player fires at the enemies or the enemies for back, collision will be detected and then health deduction or death will occur. **Priority 1**
  + R2.2.2 Collision with the terrain will also be detected, and dependent on the block; terrain destruction will occur. **Priority 2**
* R2.3 Motion Detection
  + R2.3.1 The game will be able to detect the movements of both the player, enemies, and the projectiles fired by either. **Priority 1**
* R2.4 Partial Terrain Destruction
  + R2.4.1 After collision has occurred with the player’s projectiles, parts of the terrain will be destroyed and the image will be replaced with that of a destroyed block. **Priority 2**
* R2.5 Universal Applicability
  + R2.5.1 The Physics Engine should be implementable on all objects in the game. The player, AI, and terrain are all examples of this. We will have one Physics class/file that will be used to generate realistic game physics for all objects. **Priority 1**

R3 Enemies

* R3.1 Pathfinding
  + R3.1.1 Enemies will be coded so that they can detect blocks in front of them blocking their path, and navigate around them. In addition enemies will have an aggro-range which will dictate that if a player moves into this range the enemy’s behavior toward the player will change. **Priority 1**
* R3.2 Movement in Relation to Player
  + R3.2.1 Enemies will wait until player is within a certain range of them and then begin to chase and attack the player until they are dead or have killed the player. **Priority 1**
* R3.3 Player Reward
  + R3.3.1 Each enemy that is defeated will add on to the player’s total score. **Priority 1**

R4 Level Design

* R4.1 Terrain
  + R4.1.1 Levels will designed in such a way that platforming will be as challenging of an element as fighting an enemy. The player’s jump functionally has a range that will vary with the initial velocity (this will be done using our physics engine). Platforms will be placed so that the player will sometimes have to take running jumps in order to progress farther. **Priority 1**
* R4.2 Procedural Generation
  + R4.2.1 Levels will be procedurally generated so that each time you start a new level it will be different from the last! **Priority 1**
* R4.3 Hidden areas
  + R4.3.1 Exploration of the level will be rewarded, as there are hidden areas around the level that contain power ups for the player to use. Hidden areas may or may not be accessible by killing certain mobs for certain powerups. **Priority 2**

R5 User Interface

* R5.1 Health
  + R5.1.1 The player’s health will represented by a bar located on the far right of the screen. **Priority 1**
  + R5.1.2 The bar will deplete as the player gets hit by enemies. **Priority 1**
* R5.2 Jetpack Fuel
  + R5.2.1 The jetpack fuel will be represented by a bar located on the right side of the screen on the left side of the health bar. **Priority 2**
  + R5.2.2 It will deplete as the jump key is held projecting the player into the air. **Priority 2**
* R5.3 Time
  + R5.3.1 A time counter will be located towards the far right of the screen that will be constantly incrementing as each second goes by. **Priority 1**
* R5.4 Score
  + R5.4.1 The user’s score will be centered towards the top of the screen. **Priority 1**
  + R5.4.2 The user's total score will be calculated using an algorithm taking into consideration these attributes: health, enemies defeated, powerups gathered, and time to complete the level. **Priority 1**
* R5.5 Help
  + R5.5.1 The user can at any time throughout the game click and view the rules and controls of the game. **Priority 1**

R6 Art Assets

* R6.1 Player assets
  + R6.1.1 The player is a robot and will be designed in an 8-bit fashion reminiscent of the first Super Mario game. **Priority 1**
  + R6.1.2 Player animation and design will be generated using Krita and saved in the .png format to be used in Pygame. **Priority 1**
* R6.2 Enemy assets
  + R6.2.1 Enemies are also robots built to oppose the player. **Priority 1**
  + R6.2.2 There are variations of enemies such as robot dogs and final bosses. **Priority 1**
  + R6.2.3 Enemies will be designed and imported into Pygame. **Priority 1**
* R6.3 Background assets
  + R6.3.1 In order to make levels more exciting, levels will be cycled through five different backgrounds. **Priority 1**
  + R6.3.2 Backgrounds will include: forest, city, and more to be developed. **Priority 1**
  + R6.3.3 Backgrounds will be designed and imported into Pygame. **Priority 1**
* R6.4 User Interface assets
  + R6.4.1 Both health bar and jetpack fuel bar will be created and imported into Pygame. **Priority 1**
  + R6.4.2 The score and timer will be generated by Pygame and shown on all levels. **Priority 1**

## 

## 

## **3.3 Non-Functional Requirements**

R7 Extensibility

* The software will be defined in such a way that *new* functionality can be added in the future with minimal impact. **Priority 2**

R8 Maintainability

* Design will be clean and understandable to facilitate efficient monitoring and upkeep of quality. **Priority 1**

R9 Modularity

* Related to Maintainability, the software will be designed in clear, distinct modules as much as possible. These modules will have the ability to be developed and tested more or less independently of each other, to improve development efficiency and make testing more robust. **Priority 1**

R10 Scalability

* Related to extensibility, the software will be designed so that *additional* functionality can be added to expand upon existing functionality.
* R10.1 Scalability of AI
  + R10.1.1 The AI will be easily built upon to extend with more types of AI characters. We can guarantee the presence of a single smart AI, and then build any amount of new AI types using the base AI. **Priority 2**

# 4. Use Cases

## **4.1 Use Case Flow**

### **4.1.1 Start Screen**

**Preconditions**

* The user has opened the application

**Main Flow**

1. The user clicks the “Start Game” button.

**Post-conditions**

* The user proceeds to the first level

### **4.1.2 Start Game**

**Preconditions**

* The user selected "Start Game"

**Main Flow**

1. The first level (tutorial) is displayed.
2. The user has access to the movement controls of the player.
3. The user begins to play the level and can progress to subsequent levels.

**Post-conditions**

* The user continues playing and either loses (dies), or progresses to the second level

### **4.1.3 Play Game**

**Preconditions**

* The user is playing a level of the game

**Main Flow**

1. The user runs into AI enemies and other obstacles.

**Post-conditions**

* The user must utilize controls, powerups, and weapons to finish the level

### **4.1.4 Defeating an AI enemy**

**Preconditions**

* The user has control of the player inside a level of the game

**Main Flow**

1. The user shoots to attack the enemy.
2. The user moves to avoid taking damage by the enemy.
3. Repeat step 1 and 2 until the enemy's health is down to 0.

**Post-conditions**

* The enemy is defeated and disappears

### **4.1.5 Losing to an AI enemy**

**Preconditions**

* The user has control of the player inside a level of the game

**Main Flow**

1. The player is shot by an enemy..
2. The player takes damage from the enemy.
3. The player's health goes down to 0.

**Post-conditions**

* The player dies and the user loses the level of the game
* The user starts the level over

### **4.1.6 Help**

**Preconditions**

* The user pressed the "Help" button

**Main Flow**

1. The user reads the rules
2. The user clicks the exit button

**Post-conditions**

* The user is now a master of the game
* The user understands the rules and controls

**4.1.7 Winning the Game**

**Preconditions**

* The user plays the last level of the game

**Main Flow**

1. The user defeats all enemies.

**Post-conditions**

* The user wins, the game ends, and the user is brought back to the Start screen

# 5. System Evolution

At this time, the game is meant to be a short 2D platformer that will run on personal computers. Given more time and resources the game would be expanded by adding more content; new enemies, more levels, power ups, etc. The game will most likely not be ported over to a console system or a mobile device.

# 6. Appendix

## **6.1 Glossary**

**Player** - The main character in the game.

**Pathfinding** - Algorithms used by A.I to determine the best route to take to get to the player without killing themselves in the process.

**Aggro Range** - A “detection range” used by A.I to determine how far the Player is away from them. The closer the player gets, the more likely the A.I is to attack.

**Powerup** - A “bonus” or “boost” picked up by the player. This might be a health pack, to regain lost health, or a boost to the Players laser/gun to do more damage to enemy A.I

**A.I -** Artificial Intelligence; also referred to as NPC.

**NPC -** Non-Player Character; also referred to as A.I