|  |
| --- |
| [Yoshi Moshi Platform] |
| Project Proposal |
| [Yoshi Moshi] |

|  |
| --- |
| 12/6/2013 |

# Purpose

* A 2D puzzle platform game which highlights:
  + Gravity Changing
  + Jumping Scenarios
  + Lever Platforms
  + Hunger Levels
  + Score & High Score System
  + Multiple Levels
* The Player must jump across platforms, as well as turn them on and off, and have to maintain his player's hunger, as well as how fast he manages to beat the level.
* This will demonstrate my grasp of fundamental programming skills with a platform game.
* It will teach me new skills like reading in Tiles, and creating a map.
* Will test my programming and problem solving skills and capablities.

# Statement

I will be creating a Yoshi Puzzle Game, where a Yoshi character tries to get to a finish line, as fast as possible, and without dying. The player must complete a set of puzzles and jumping tests in order to progress in the level. He must also strategize as to how he will do so, as different levers open different platforms, opening wrong ones, can lead to wasting time. The player will receive a score based on how fast he completes the level, and how many stars he receives. Although, Yoshi tends to get hungry fast, so feeding him will be key for the player to complete the level. But beware, if you feed him you also have to wait for him to finish eating. Which will ultimately result in a lower score. Once you finish the level, your score will be calculated, and you will continue to play in the next level. When the player, or Yoshi, runs out of lives, or beats the game, and if the player's total score will be higher than any of the Top Ten Scores, then their score will be updated along with their name onto the leader boards, and the game will be finished.

# Objectives

**Group Name:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Yoshi Moshi Platforms\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1: **Objective**: Title: Level Loading

Description:

The program has to read in a picture (or text file) indicating every tile in the map.

This should be done before the game even draws, and should spawn in the tiles in a certain spot.

2: **Objective**: Title: Collision Detection:

Description:

The program/game must make sure that the player stays within the boundary of each tile, as well as, above certain tiles. This should be done as soon as the game draws itself

3: **Objective**: Title: 2D Camera:

Description:

The program must be able to show only a portion of the map, and be able to move the tiles toward the player appropriately.

4: **Objective**: Title: Movable Objects:

Description:

The player should be able to move certain blocks and should be able to jump onto these blocks in order to complete a certain objective

5: **Objective**: Title: Multiple Levels:

Description:

The game should have a finish line, where once the player reaches it, the next level will be loaded. There will be a total of 3 levels.

6: **Objective**: Title: Interactive Items:

Description:

Certain tiles will kill the player, while others will give him extra life. Also a star tile will raise his score, which will reward him with a higher chance of making the High Score. These stars will be strategically placed in the map, and be difficult to obtain. Although completing a level does not require these stars. In addition, an apple tree or just an apple will be another interactive tile, which will let the player restore his Hunger, but will deduct his score.

7: **Objective**: Title: Frame Animation:

Description:

The game will have different frames which will loop and animate the character at a smooth speed.

8: **Objective**: Title: Background Music & Sound Effects

Description:

Once the game loads, a background song will continuously loop, and won’t stop until then player clicks the M button which will mute the audio. Also a sound effect whenever the player completes the level, one when he picks up stars, when he jumps, when he eats, and one where he dies.

9: **Objective**: Title: Score System:

Description:

The game will have a time based score system. The longer it takes the player to complete the level, the smaller his score will be. Once the player loses 3 lives, or beats the game, and he surpasses one of the top ten scores, then his score and name will be saved on the scoreboard and in a separate Text File. Throughout the game, anyone can view the top 10 scores (if there are 10 scores) by pressing L.

10: **Objective**: Title: Hunger:

Description:

The player has a hunger limit, when his hunger bar becomes empty he starts to starve. After 30 seconds of starvation, the player will die and lose a life. At certain checkpoints, the player will have a chance to replenish his hunger, but not his health

# Technical Outline

1: **Objective**: Title: Level Loading

Technical Details:

Find a program which takes each picture, and goes through every pixel in the picture, and then stores its red value (RGBA), and the player's location in a separate file. And then have a program which reads in this red value and stores it as the id, and the location of this tile in the map. Then it redraws the map, and if player interacts with his environment, based on the tile ID that he interacts with, different events will happen. Or have a subprogram which reads in a Text file, and based on what characters it reads, it knows what Picture it has to draw, and where.

2: **Objective**: Title: Collision Detection

Technical Details:

Adds the player's X value and his width, and the same thing with the Y value and its Height, then checks this with the Each Tile to make sure that their X's and Y's don't Intersects. Using the Intersects Method to make sure that player doesn't intersect with each tile. And making sure that the player can't get out of the Window or even map.

3: **Objective**: Title: 2D Camera

Technical Details:

Shows a certain view of the entire map, and as the player moves, the camera moves with him.

4: **Objective**: Title: Movable Objects:

Technical Details:

Every single tile will have a boolean assigned to it, which will evaluate to true or false based on movable or not. If the object is movable, and the player collides with it, and continues to move, then the tile will move in that direction with it.

5: **Objective**: Title: Multiple Levels

Technical Details:

Once the player passes a level, or lands on the finish line, the next level will load. This will be done with an integer that is set to the level needed to be loaded, , and when the player finishes the level the integer increments by one and loads the next level based on the integer value.

6: **Objective**: Title: Interactive Items:

Technical Details:

If the player lands on or collides with any tile that might kill him, give him an extra life, feed him, raise his score, or even lower his score, appropriate data will be changed. For example if the player lands on an apple tile, then the hunger variable will be reset to its original value (100).

7: **Objective**: Title: Frame Animation:

Technical Details:

The program will use an image, and given the frame width and height, it will draw the first frame in the image, and then the next, until reaches the end of the image, to which it will start again at the beginning of the image, on the first frame. It will go through all the frames in X amount of time. The X value depends on the smoothness of the animation.

8: **Objective**: Title: Background Music and Sound Effects

Technical Details:

I will integrate the audio with the program, by adding the song to the program, and whenever a certain event happens, then it will play the sound. For example, the Background music will start playing as soon as the game opens up, and will loop until the game is closed. Or when the player dies, eats an apple, gets a star, or an extra life, sound effects will be played.

9: **Objective**: Title: Score System

Technical Details:

A stopwatch starts at the beginning of the game, and when continues until the player either wins the level, or loses all his lives. If he beats the level then if his time is greater than, a certain time, then he gets a certain amount of score. If he gets less than the certain time which I set, then he gets more even more score. Events like eating an apple or gaining a star will lower his score(apple) or raise his score(star). If the player's score by the end of the game or when he loses all his lives, is greater than any of the scores in the Top Ten Score board, then his score and name is added to the leader board, and is stored in an external file.

10: **Objective**: Title: Hunger

Technical Details:

There will be a variable that is set to 100 (percent), named something along the lines of hunger. After every 5 seconds, hunger goes down by 5 (percent). At certain checkpoints, there will be an option for Yoshi to eat. If he does eat, then his Hunger goes back to 100 (percent), but he gets a 5 second time increase, for eating. If Yoshi's hunger gets down to 0, then slowly the player will start to lose Health, at a rate of a percent for every second, until the player dies and loses one life. At this point the game is reset, and the player starts from the beginning.

# Design(s)

## Hunger Objective (Part of it):

SET health to 100

SET hunger to 100

SET timePassed to 0

SET time to 0

DISPLAY timePassed

DISPLAY hunger

DISPLAY health

IF 5 seconds passed in time THEN

COMPUTE hunger as hunger - 5

SET time to 0

ENDIF

IF hunger equals to 0 THEN

IF one second passes THEN

DECREMENT health

IF health reaches 0 THEN

CALL ResetGame with map, health, hunger, lives, score, and timePassed

ENDIF

IF player eats apple THEN

SET hunger as 100

COMPUTE timePassed as timePassed + 5

ENDIF

ENDIF

ENDIF

# Milestones

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task #** | **Description** | **Worker** | **Due Date** | **Dependent Task #s** |
| 1 | Create a basic map to test with. | Ori | 12/11/13 |  |
| 2 | Get all artwork needed. | Ori | 12/13/13 |  |
| 3 | Reading in the map data (Level Loading Objective). | Ori | 12/21/13 | 1,2 |
| 4 | Work out the 2D camera view objective. | Ori | 12/23/13 | 3 |
| 5 | (On vacation from 24-27)Make sure that player can't go through most tiles (Collision Detection). | Ori | 12/28/13 | 3 |
| 6 | Be able to move certain objects. | Ori | 12/29/13 | 5 |
| 7 | Make the levers and the bridges. | Ori | 12/31/13 | 3 |
| 8 | Make the stars and the apples appear on the screen. | Ori | 12/30/13 | 3 |
| 9 | Make sure that the player can go through the levers, stars, and the apples, as well as collide with them and interact with them. Make it so that the apples disappear after eating, if it is not an apple tree, and that the same happens with the stars. | Ori | 12/31/13 | 5 |
| 10 | Make sure that all incentives give their power up's or downs. | Ori | 01/01/14 | 9 |
| 11 | Create or Download Audio files, and integrate them into the game were needed (i.e. Background Music to loop the entire time, Music to play when the user completes the level, music to play when he dies, music to play when he gets a star... etc.). | Ori | 01/04/14 |  |
| 12 | Integrate the score system into the program. | Ori | 01/05/14 |  |
| 13 | Create a minimum of 3 Levels. | Ori | 01/07/14 |  |
| 14 | Create High Score System, with System IO. | Ori | 01/06/14 | 12 |
| 15 | Make sure that the apples (or even apple tree) work with the character and that the character has the option whether or not to eat the apple. Make sure that it restores Hunger, and that it increases the Stopwatch by 5 seconds. | Ori | 01/07/14 | 9 |
| 16 | Make sure that every 5 seconds the player loses 5% hunger and when he reaches 0% he starts losing 10% health for every 10 seconds or 1% per second. | Ori | 01/08/14 |  |
| 17 | Make sure that when the player dies the map resets, and that his hunger and health become full. But his time continues. | Ori | 01/09/14 |  |
| 18 | Program that when the player completes the level, he can move onto the next. Until there are no more levels created. | Ori | 01/09/14 | 13 |
| 19 | Work on Frame Animation, try to find a Sprite Sheet and integrate Frame Animation. | Ori | 01/15/14 |  |
| 20 | Finalize Project, and work on any final details. Polish up the project. | Ori | 01/19/14 |  |

**Declaration:**

**I have read the statements regarding cheating in the assignment description. I affirm with my signature that I have worked out my own solution to this assignment, and the code I am handing in is my own.**

**Signatures: (print and sign name)**