SOFTWARE REQUIREMENTS SPECIFICATION

for

Bongo Congo [Code Name]

Version 1.0

Prepared by Adam Cox, Adam Worwood, Andra-Cristina Tudoran, Jacqueline Henes, Joseph Tuffin, Lewis Relph

Knightlore

February 1, 2019

Contents

1 Introduction

1.1 Purpose

The purpose of this document is to identify the requirements of our game. We have also included a detailed Action Plan which specifies requirements and deadlines so we can ensure we are making adequate progress.

1.2 Brief

Our game is an isometric platform/race game where players compete to reach the end of the level in the fastest time. Players race whilst dodging hazards and enemies and the winner of a game calculated based on their total time remaining for individual levels.

2 Game

2.1 Single Player

In Single Player mode, the player races against the clock, with their aim being to get to the end of a level as quickly as possible. There will be three levels, and each level will have a given time limit that the player must try to beat. If the player can't beat the timer, they are given a score of zero and automatically moved to the next level. A player will receive a higher score if they finish a level with more time to spare. The player will also have three lives, for each level, that can be lost by colliding with enemies and obstacles, or by falling off the level. If a player loses a life, they are sent back to the beginning of the level and if they run out of lives, they are given a score of zero for that level and automatically moved to the next one.

2.2 Multiplayer

In Multiplayer mode, two players race against each other to see who can get to the end quicker. The winner of the two is decided by whoever has the highest score after the end of the third level. If one player loses all their lives, they will still be given a score of zero for that level, but the level will continue until the other player has completed the level, or also run out of lives.

3 User Interface

3.1 Main Menu

There must be a main menu which will be the first screen the player sees when they start the game. This main menu will include a button to start a single-player game which will take the player straight into the game. There must be a multiplayer button which moves the player to the multiplayer screen. There must be an options buttons which takes the user to the options menu.

3.2 Multiplayer Menu

There must be a screen for connecting to multiplayer hosts. This screen will list the available games to join. This menu must also include a button to create a game which will move the player to the waiting screen, where they will updated with information on whether someone has joined their game or not. There will also be buttons to join the selected game, a button to refresh the listed games, and finally a button to return to the main menu.

3.3 Options Menu

There must be an options menu to adjust the game settings. This will be where the user is able to change settings such as toggling sound effects and music on and off as well as adjusting volume levels of each. There also must be a button to return to the main menu.

3.4 In-Game

There must be a Head-Up Display during the game. The display must show the count-down clock which shows the duration of the level left. The HUD must show the overall scores for each player across all levels (starting at zero). This score will be updated at the end of each level. The HUD must show the number of lives left for each player. The number of lives starts at three and goes down by 1 if the player dies. If the player runs out of lives then the game will display a death message telling the player that they have run out of lives. The HUD must display the winner at the end of the final level and include a button to exit the game.

4 Map

4.1 Design

The map will be a layered isometric map, because of the limited view, the map will generally be a hill climb with the finish line at the top of the hill.

4.2 Tiles

Tiles act as the individual blocks that build up the map. Some tiles have certain functionality, including Spawn tiles on which the player is initially placed at the start of the game, for multiplayer 2 tiles must be used. Floor tiles have collision, allowing for a player or entity to walk on top of them. Climbable tiles will allow players who approach the side to "climb" up them, elevating the player position. River tiles are hazardous, sweeping the player off the map if stepped on. A finish tile must also be located on the map, which players will race to in order to win. Enemy spawn tiles will also simply pick which tiles enemies spawn on.

5 Player

The player should have 3 lives. The player must be able to move in 8 directions; up, down, left, right and along the isometric grid. Another functionality the player must display is "rolling", where rolling is defined as a period of invulnerability where the player quickly moves in the direction they're facing. In this period the player's sprite will be animated to show them enacting a forward roll.

Finally the player must be able to climb, where climbing acts as the ability to move up one layer of the map in the Y direction.

6 AI

Each enemy placed on the game map will have a preset AI behaviour designated upon game initialisation. Enemies will have this AI for the entire duration of that specific game.

All enemy AI will only be able to walk on floor tiles, and only stay on the level that they were spawned on. If any enemy, during their walk cycle, walks into a wall, they will stop, reverse direction, and walk in that direction. Otherwise, there are the following differing AI behaviours:

- Walker: Walks along a set direction on one plane.
- Charger: Idles at their spawn point until a player enters their activation range. Once a player is within range, the Charger starts moving towards that player. The range is static, so if the player leaves the activation area the enemy stops chasing the player
- Circler: Moves around in a circle of a set size
- Randomer: Walks in a direction for a set distance, then randomly chooses a new direction to walk in before continuing

7 Physics

The game will have collision physics to do with the entities and tiles in the game. A collision event happens when either an entity moves onto a tile that is not a floor tile, or when an entity moves onto the same tile as another entity.

The following distinct entity collision events can happen:

- A player collides with an enemy - Player loses a life
- A player collides with an enemy while rolling -Player does not lose a life, passes through the enemy
- A player colliding with a wall tile - Player is prevented from moving onto the tile
- A player colliding with a climbable tile either from one level higher or one level below - The player either moves up one layer or moves down one layer
- A player colliding with an air tile - Player falls to a lower layer of the map
- A player colliding with two air tiles in a row - Player dies
- A player colliding with the finish tile - Player wins, and the current level finishes
- Two human-controlled players colliding - Nothing happens
- Enemies colliding with wall tiles - Enemies reverse direction and continue
- An enemy colliding with another enemy - Nothing happens

Additionally, entities (including the player) cannot fall off the boundaries of the map.The edges of the map are essentially treated as wall tiles.

8 Audio

8.1 Music

There must be different, distinct music tracks for both the menu screens and in the game, i.e. the music playing on the menu screen must be different to the music playing during the main game. The music will loop indefinitely. The music must also have adjustable volume, and players must be able to completely disable (and re-enable) it. Both of these actions can be performed by accessing the Options Menu.

8.2 Sound Effects

There must be short sound effects for the important events that can take place in the game, such as rolling, climbing and dying, as well as the actions players perform outside of the game like selecting menu items. Sound effects will only play when a certain condition is met, upon which the effect should only play once, until the condition is met again. Sound effects should have adjustable volume and players must be able to completely disable (and re-enable) them. Both of these actions can be performed by accessing the Options Menu.

8.3 Jingles

There must be short musical jingles that play when significant events occur in the game (winning and losing). Jingles will not loop, and should be able to be disabled and have adjustable volume, both of which can be accessed through the Options Menu.

9 Networking

A system must be in place to allow for the following functionality before the game itself starts; registering the game with a authentication server to validate its authenticity (DRM). The system must also allow for creating new games, listing the current games waiting to start and both joining and starting the online game.

The system should also be able to send the initial game data to clients so they are synced.

When the game begins, the system should be responsible for sending various information about player movement, including the entity being moved, the new location and the heading. The games across clients should be constantly in sync, so information such as time remaining and lives should also be handled. Other specific events that may occur include deaths, game ending, and who won the game.

When levels are completed, the client should wait to receive the next level object from the server.

Finally, this must also be implemented in a way which the model being used is independent from the transmission implementation, allowing for simply using different controllers for single/multiplayer games.

10 Action Plan

10.1 Week 6 Prototype

By week 6 we plan to have the following features implemented into the product. A basic model of the game (map, tiles, entities etc). A rendering system that is based on the model, and reflects changes in it. A working HUD should also be implemented displaying player lives and scores. Audio should be completely implemented. The initial networking should also be complete, including authentication, creating and listing games, joining games and starting games. Player movement over the network should also be implemented.

10.2 Week 9 Prototype

By week 9 we plan to have enemies and physics introduced into the game, along with the main menu and multiplayer screens implemented. Networking should also be ready to accommodate all other remaining aspects, including sending new level data, detecting deaths etc. At this point we also should have rolling implemented, along with complete support for multiple levels.

10.3 Final Build

Having finished all of the above we want to take the time to add a bit of polish to our game. This includes finishing up the art style of the game, creating/acquiring assets which fit a desired theme. The levels will also have to be worked on, making sure they are a fun and fair experience to play on. The fairness comes into account for multiplayer as we don't want specific player spawns having an unfair advantage over another. Finally, tuning network requirements for the server and the clients. This includes adjusting server tick rate (how often the server sends updates) and how the clients handle these updates. While this may be worked out throughout, we will want to get this nailed down by the final build, so that it is a smooth experience to play multiplayer without noticeable networking issues.