**Abstract – 150 Words**

The aim of this project is to use the 2D functionality within Unity to create a simple platforming game. It will show off a range of skills such as animation, game design and software engineering.

This is a pixel art game which involves a human adventurer travelling through a magical forest. It includes several obstacles which test the user’s skills and adds an aspect of fun. It also uses time travel so you can replay the last 20 seconds.

**Introduction – 300 Words**

*Project and its aims.*

This is a 2D game which is inspired by Braid – taking the main game mechanic and adapting it into this project. One of the main game mechanics is to have the ability to travel your character back 20 seconds. I designed two different levels for the game – the game ends after the second level has been completed.

The main character is an adventurer, who is fully animated – with the ability to jump, climb, and run their way through levels. I did not create the original assets; sources are listed below. I meant this project to be a display of my skills within Unity and game creation.

The aim is to create a similar game in Unity, using 2D physics and tile maps.

**Game Design – 1 page**

*What will they get out of this? Describe the primary mechanic.*

*What makes this project a game?*

This project is game because it has a win and lose state. The win state is when the player reaches the flag – there is one per level. There are several obstacles such as falling rocks and spikes – they cause the player to die – ending the game. It also has a special feature which allows the player to go back in time.

*What sort of game is it?*

It is a side scrolling, platformer game. It revolves around an adventurer who must traverse ladders and platforms to reach the end goal. The camera follows the player - keeping it centred. This means that the focus is on the player and allows for a complex and bigger screen.

The game also has a graphic user interface, in which you can restart and quit a level. The initial design of that is shown below.

*What will motivate someone to play this game?*

The game is challenging as the player must traverse platforms to reach the goal. People will play this game because it has a nice aesthetic, challenging levels and a variety of obstacles.

*How the mechanics fit together and make the user play?*

I went through a few stages of designing levels – the basic level design is here though it needed a lot more testing when during the implementation stage. This is due to changes in player implementation which affected the jump height and distance for example.

**Software Design – 4 pages**

*Describe the principle components of the design.*

The main components of the game are:

* Platforms
* Spikes
* Falling Rocks
* Time Travel
* Ladders
* Goal

*Describe how these components contribute to fulfilling the specification.*

All assets are third-party – they are referenced at the end of the document.

Platforms:

Platforms within the levels have varied sprites to give the game more flair and variety. They are situated on a grid layer which is used for player collision. The player changes state to GROUND when it has collided with the platform. This is important as without it, the player would not be able to trigger the jump method.

Spikes:

Touching the spikes results in a game over state for the player. The sprites are located on the “GAMEOVER” grid layer because the player contains a script which checks for any collision via the layers. This makes adding any spike components easier to manage and add as they are encapsulated within the grid.

Ladders:

The ladders are on a grid layer as the methods are managed by the player controller script. The player automatically changes to the CLIMB state. The player can then traverse up and down, with appropriate physics. The sprite for this is a vine because it fits the theme.

Goal:

The goal is on a separate grid level and is

Level Limits:

I also used a similar layer to limit the players movement going off the screen. I chose sprite blocks that blend in with the environment. I did this because it fits the aesthetic, and it does not distract from the immersion of the game.

Time Travel

Time travel is a key aspect in this game as it is loosely based on Braid – see introduction for more details. It tracks the last 20 positions the player has been in and gives the user the ability to take the player back. Unfortunately, it is not implemented into the game due to time constraints.

*Describe how the components fit together and contribute to the whole.*

There are two challenging levels for the player to beat – which combine all the components specified in the brief.

*Use UML diagrams to explain key points.*

An overall class diagram of the scripts is below:

The player has a script called “playerController” attached to it and that controls the states/actions of the player. It observes the different interactions between the features and the player. It is also linked to the “EnvironmentController” as it relays the players win state and shows the appropriate graphical user interface.

*What alternative designs do you consider (or try) and what are the pros and cons of these different choices?*

I took a modular approach to designing levels, starting by designing small components that could fit together creating the level. Examples of these are below. I got a small test group to rate these components in fun and difficulty – I took those ratings and drafted levels from there. They then got play tested and tweaked – to make sure the levels were beatable and fun.

As I used third party graphics, I also had to bear in mind the sizing of these sprites as this affected the hit boxes and the positioning for example.

**Testing – 1 page**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task | Description | Expected Outcome | Outcome | Pass/Fail |
| Player moves right on the horizontal axis. | Player moves when ‘D’ and right arrow are pressed. | Player moves right on key press. |  | Pass |
| Player moves left on the horizontal axis. | Player moves when ‘A’ and left arrow are pressed. | Player moves left on key press. |  | Pass |
| Right running animation plays when moved on Horizonal Axis | Player animation plays when ‘D’ and right arrow are pressed. | Right running animation plays when key pressed down. |  | Pass |
| Left running animation plays when moved on Horizonal Axis | Player animation plays when ‘A’ and left arrow are pressed. | Left running animation plays when key pressed down. |  |  |
| Player jumps correctly on the Y axis. | Player triggers the JUMP state and moves on the Y axis when the space bar is pressed. | Player moves on the Y axis appropriately and triggers JUMP state. |  | Pass |
| Jump animation plays when Player jumps. | Player jump animation plays when space bar is pressed. | Player changes to jump animation when space key pressed. |  | Pass |
| Player dies when it touches spikes. | Player triggers GAMEOVER state and ends game when colliding with spikes. | Player triggers GAMEOVER state and ends game when colliding with spikes. |  |  |
| Game over animation plays when Player dies. | Player game over animation plays when GAMOVER state triggered. | Player changes to game over animation when GAMEOVER state triggered. |  | Pass |
| Player dies when it touches rocks. | Player triggers GAMEOVER state and ends game when colliding with rocks. | Player triggers GAMEOVER state and ends game when colliding with rocks. |  | Fail |
| Player wins the level when touching the flag. | Player triggers WIN state and ends level when colliding with flag. | Player triggers WIN state and ends game when colliding with the flag. |  | Fail |
| Player goes to next level when touching the flag. | New scene opens and Player starts new level if not currently at the last level. | New scene opens and Player starts new level if not currently at the last level. |  | Fail |
| Climb animation plays when Player climbs. | Player climb animation plays when either ‘W’ or Up arrow is pressed. | Player climb animation when correct input detected. |  | Fail |
| Player climbs when it touches ladder. | Player triggers CLIMB state and allows climb movement. | Player triggers CLIMB state and allows climb movement when colliding with ladder. |  | Fail |

**Discussion and Reflection – 1 page**

*This section should answer the following questions: What are the primary strengths of your project? What are its weakness? What have you learned during this project? What would you do differently next time? If during self reflection you have identified an issue, e.g. time management, what actual steps could you take to address this issue?*

The main strengths of the project are the element abstraction as I spent a lot of time organising all the elements in the project. This makes it easier to develop further and debug if any issues arrise.

References:

* [*https://rvros.itch.io/animated-pixel-hero?download*](https://rvros.itch.io/animated-pixel-hero?download)
* *https://trixelized.itch.io/starstring-fields*