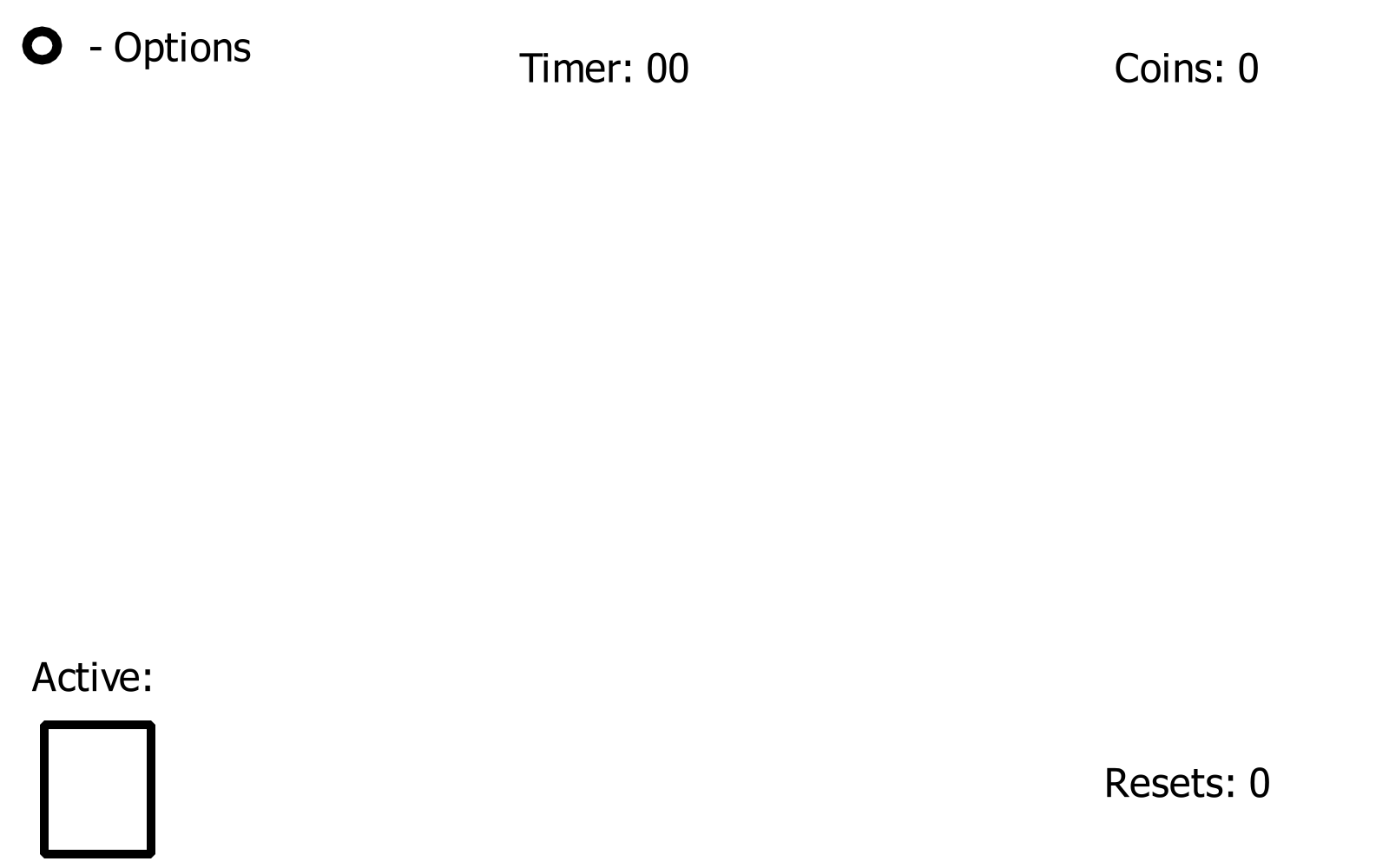
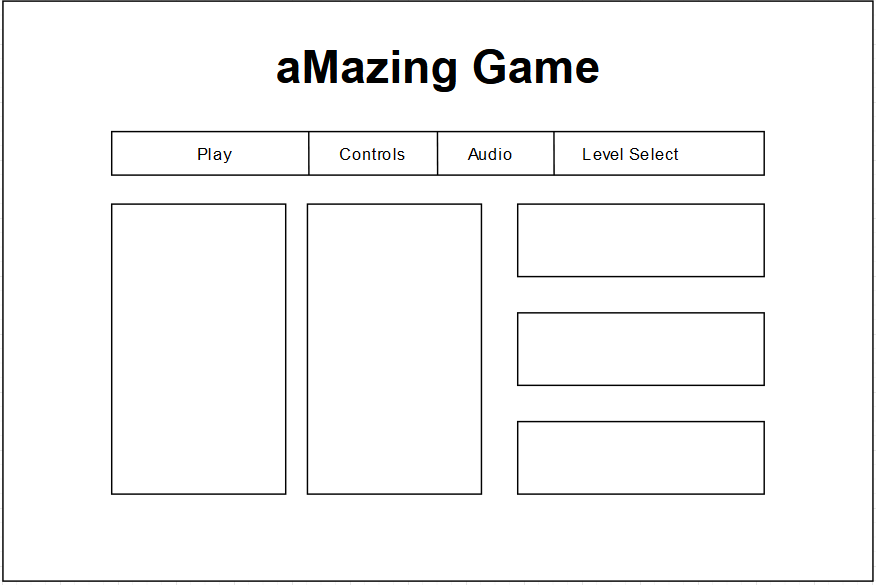
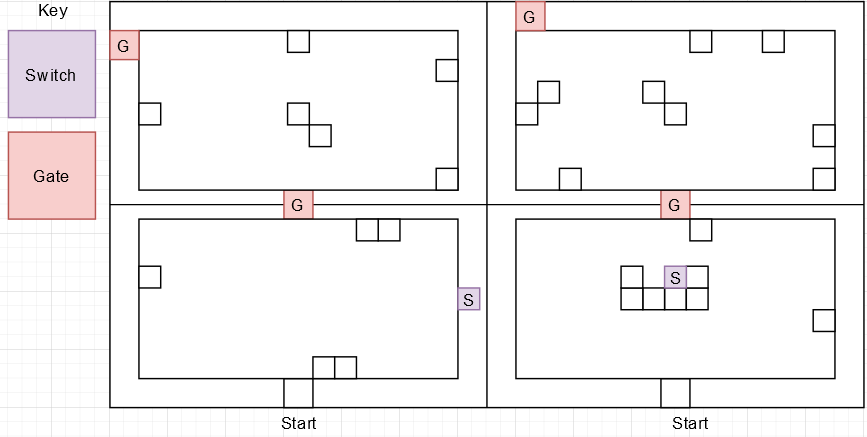
**Maze Game**

**Basic description of the Game:**

Simplistic maze game with two moveable characters, when one character is moved the other will move in the opposite direction. With the goal to have both characters escape the maze.

**UI example:**



**Level Example:** 

**Software and pre-built Libraries used:**

Unity 3D  
Visual Studio 2022  
Standard Assets Library  
Text Mesh Pro  
POLYGON - 3D Asset Series (by Synty Studios)

**Game mechanics:**

Movement:

Movement will be calculated using Unity’s built in keywords “Horizontal” and “Vertical” that returns a float depending on the directional input. A raycast will be sent from the player character in the direction until it hits a wall.

PC controls: The PC controls will use the arrow keys or WASD to set the direction the characters will move.

Mobile controls: The mobile controls will use raycasts on screen space to detect player input and calculate a direction based on that.

Switches/Buttons:

Within the game will be switches and buttons. These are going to be used to open gates and paths for each character. Some buttons may interact with the opposite character's map.

Timer:

Displayed at the top of the screen will be a timer that shows the player how long it has taken them to complete the level, adding replayability to the game in hopes of accomplishing a faster time.

Coins:

As an additional method to measure how well the player did, coins will be placed around the map for the player to collect and will be displayed in the top right of the screen.

Mirroring:

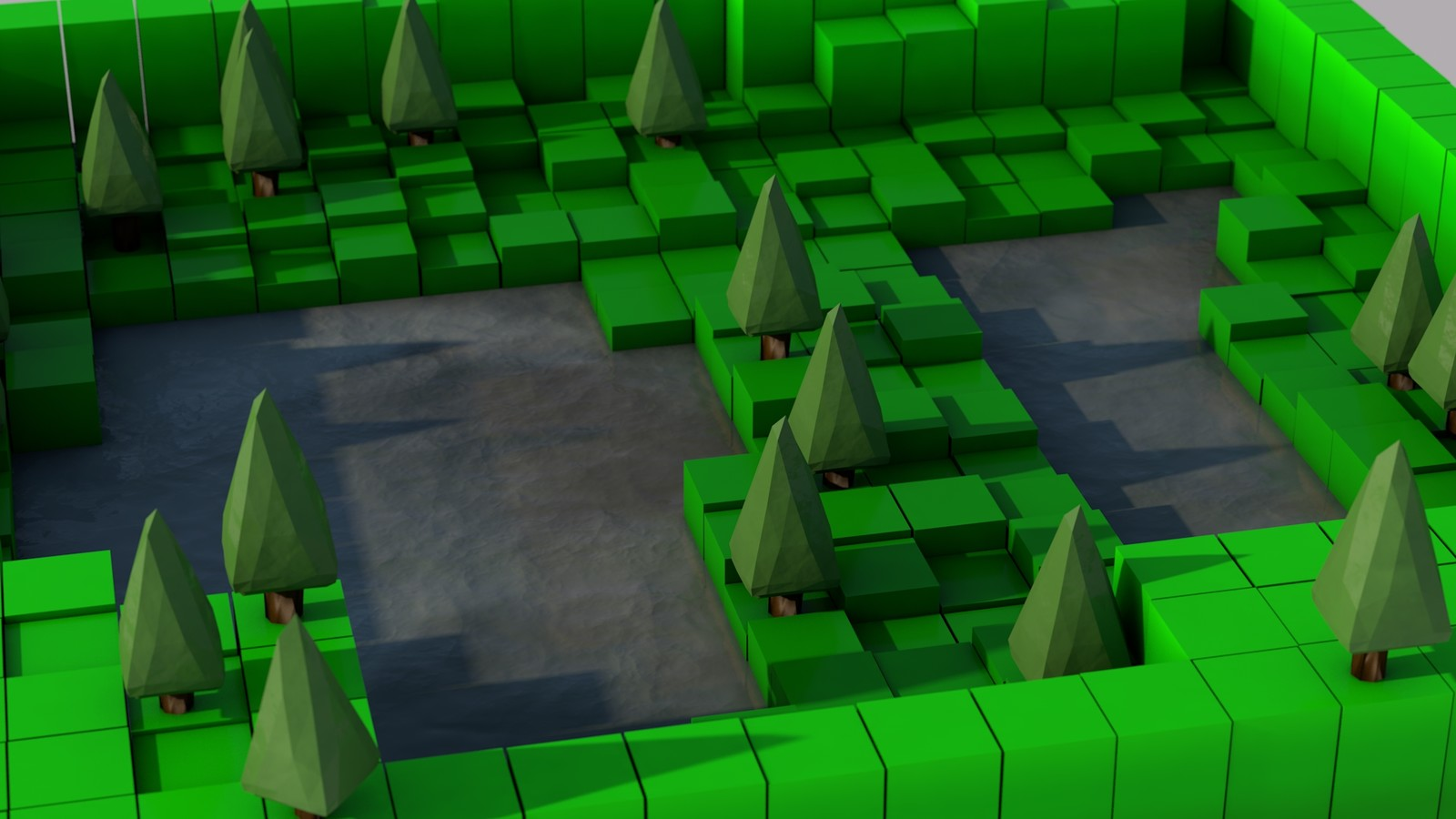
When controlling the highlighted character, the other character will move in the opposite direction horizontally at the same time.

Resets:

If the player ever gets stuck, they are able to start again using a dedicated reset button. Using this will set the characters positions back to where they started and reset the buttons and switches.

**Art:**

low poly. Forest with a maze in the middle. Using self made assets and assets from POLYGON - 3D Asset Series (by Synty Studios)

Example of forest:

# 

**Task 3:**

**The project requirements:**

A Playable game that can run smoothly on three different platforms.

**A technical description of the project you will make:**

The game will revolve around controlling 2 characters using a single input system that will move them in the same direction vertically but in the opposite direction horizontally.   
  
In order to detect how far the characters move, each of them will send a raycast in the input direction that will collide with the first wall hit. From there the characters designate that as the target location and will rotate in that direction and move until they are a specified distance from the wall, where they will wait for further input from the player. While moving the player will be unable to input any directions until the character reaches their destination, ensuring well thought out moves.

The player is required to get both characters to the exits in their level. This may require switches and buttons to be activated on both ends.

**A list of assets required (this includes code assets, as well as art assets):**

Standard Assets Pack.  
Text Mesh Pro.  
POLYGON - 3D Asset Series (by Synty Studios)

**The formats of assets and any issues of asset integration:**

The assets that we will use from the Standard Assets Pack, includes player models and animations. Integrating models from an outside source runs the risk of the models not being an appropriate scale for the project and may be difficult to properly scale down.

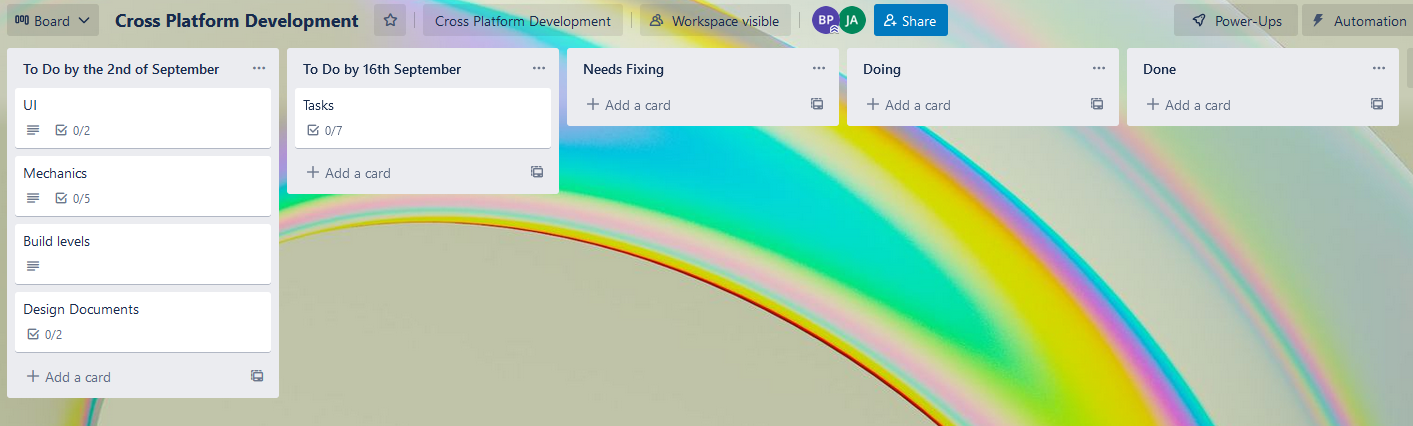
Text Mesh Pro will be used to style the UI and menus for the game by giving us access to fonts and additional designs for text,images and buttons.

**System Requirements:**PC: OS- Windows 7+ (Windows), High Sierra 10.13+(Mac), Ubuntu 16.04+(Linux)  
CPU- X64 architecture  
Graphics API- DX10+

Mobile:   
Android 4.4 (API 19)+, iOS 11+  
1 GB RAM

Web: HTML 5 standards compliant browser (e.g. Chrome, Firefox, Safari).

**A schedule for production and testing, including milestones:**

[**https://trello.com/invite/b/4nYnEg4E/b5fb1885a92db4203539be8ac8736b3d/cross-platform-development**](https://trello.com/invite/b/4nYnEg4E/b5fb1885a92db4203539be8ac8736b3d/cross-platform-development) ****

**Defined developer roles for each team member:**

Jason - Programmer,

Blake - Programmer, Level Designer

Luke - Programmer, UI Designer and programmer

**Strategies for monitoring production progress**

Using websites such as trello to have a visible plan to set and follow, allowing the group to decide what should be completed in production after each week. This will allow for everyone to see the total progress of the project in all aspects.

**Task 4:**

**Implement a 3D environment (combining a 3D environment with 2D mechanics is acceptable).**

**Demonstrate visual effects.**

**Contain source code that includes comments.**

**Contain gameplay elements that have been created and checked according to creative and technical requirements.**

**Use 3D, audio and physics libraries (or related features of the selected game engine).**

**Contain animated 3D objects required by gameplay.**

**Use exception handling techniques.**

**Contain a GUI interface.**

**Task 5:**

**Development testing**

* Test for faults, documenting findings
* Performance profiling and optimisation
* Using log files for errors and events
* Use debugging techniques to remove faults
* Formally evaluate your prototype against the design requirements, discuss and agree on required changes

**Playtesting**

* Run playtesting sessions and evaluate feedback from user trials

**Present**

* Run your prototype as a presentation. Your presentation will be 3-5 minutes in length.
* Provide an overview of your game and elicit feedback.
* Record any feedback received. You will use this feedback to refine your Technical Design Document in Task 6.

**Task 6:**

* Apply changes as required from testing and end-user trials
* Integrate all game elements, as needed for the initial brief
* Create and maintain internal code documentation using a third-party tool like *Visual Studio XML Comments* or *Doxygen*

**How to Build Information:**

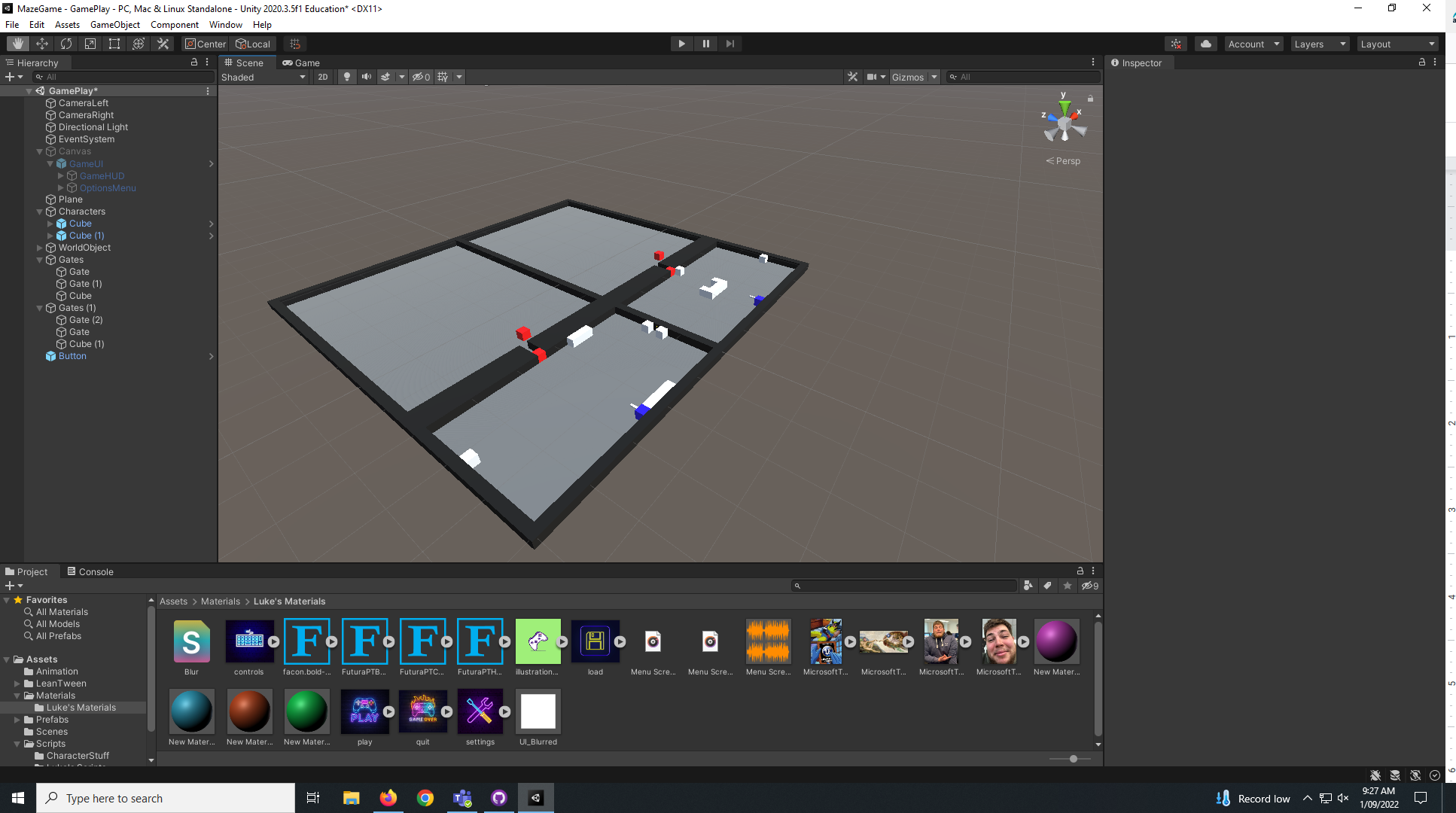
<https://docs.unity3d.com/Manual/android-BuildProcess.html> - Android Build

<https://docs.unity3d.com/Manual/webgl-building.html> - Web build

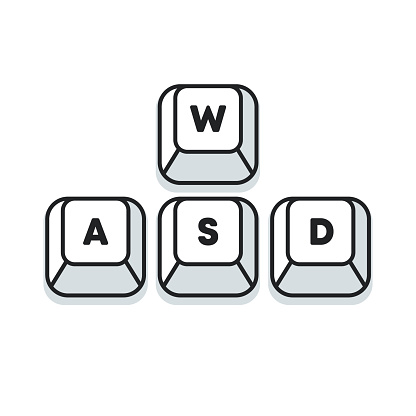
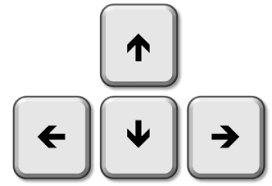
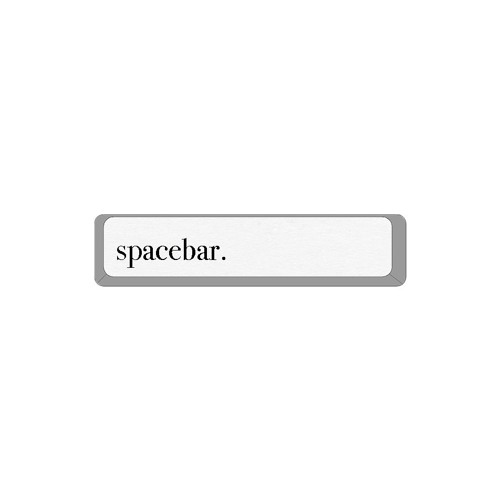
<https://docs.unity3d.com/Manual/Windows.html> - Pc build

First camera z location -22.5

Second camera z location 12



Controls:

PC:



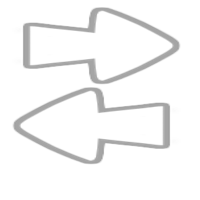
Wasd and arrow keys move the characters up, down, left and right. Spacebar will swap the active character, E is used to interact with buttons/switches. R is used for resetting to the start of the current level.

Mobile:



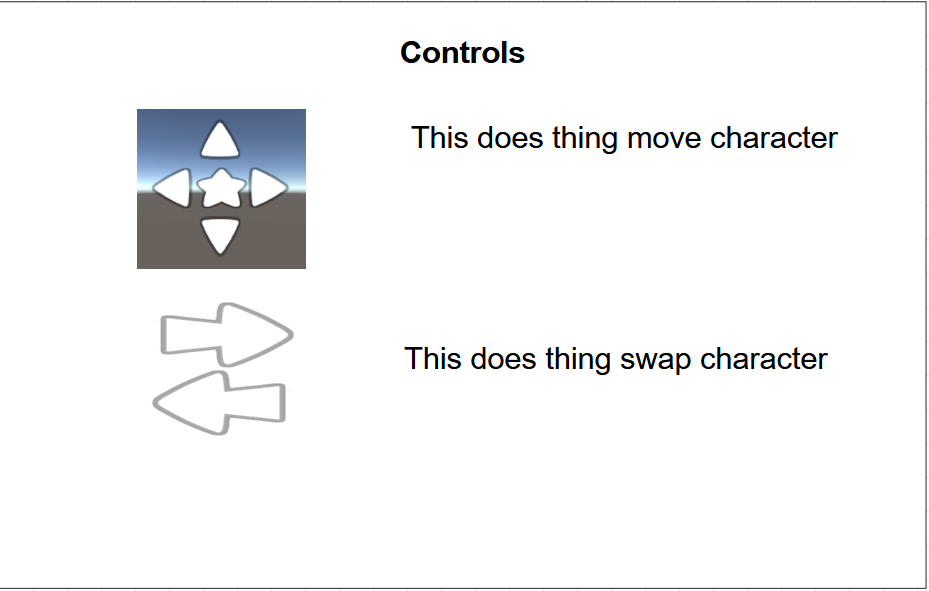
Arrows are used for up, down, left and right.

Star is interactable, this activates buttons/switches.



Swaps the active character

Pop up UI for mobile controls before game starts but after the play button is selected



**List on Known problems:**

Cameras need to be adjusted to fit the level

No art assets

Hard to see UI (black text on black objects)

T posing characters

Main menu UI slightly different to in game UI

| Name: | FeedBack: |
| --- | --- |
| Thomas | The controls of possessing both characters at the same time are not necessarily intuitive. The fact the inactive player goes opposite for the left and right is unorthodox and jarring. Possessing 1 character at a time or not inverting the controls of only one axis may benefit the game. |
| Millie | The main menu looks nice, but the controls were incorrect. Make more levels and add terrain to the scene to make it look more eye-catching.  Change character model. |
|  |  |
|  |  |
|  |  |
| Class | Add grid to the ground. Add indicator under the character |

