Underground Escape – Report

# Introduction

At the beginning of this project, the group had a team of 5 and thoroughly discussed which of the 5 design documents should be chosen to be implemented into a game prototype. The group then chose at the time that Ruya’s game would be the best option to carry out.

Other options considered was Akash’s game called “What Little Remains” which was a turn-based strategy game similar and inspired by Fire Emblem and Marks game called “The Charlatan” which was a 2D Adventure game. The group decided not to go for those design documents due to the complication of the games and given the time frame when the project was first being created the group needed a simpler and more easy game to implement but still had the fun and challenge that was sought for.

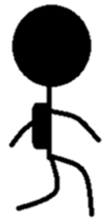
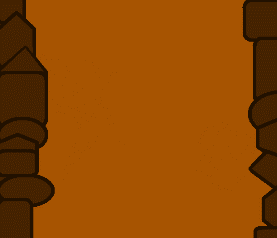
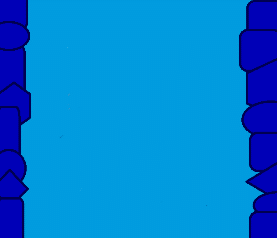
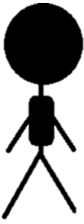
Which is why the game chosen was “Underground Escape” by Ruya, it is a 2D Platform game with vertical parallax scrolling, with varying difficulty each time the game is played with random placed platforms, enemies, and hazards throughout the screen. This game also had multiple worlds so we could showcase at least 2 of them for the prototype.

# Project Management

At the beginning of the project the group were 5 team members, but due to unforeseen circumstances the group became 3 team members of Ruya, Akash and Mark, and had to rearrange roles and planning in July 2020. In our first meeting as a group of 3 on the 6th July 2020 the group had decided to go with the Scrum Methodology, this meant the game could be built in small parts and tested along the way during development and also give each person specific roles to do within the game in order to maximise efficiency and time in creating this prototype. As Ruya was the one who created the design document being used for the prototype project she was appointed as the Product Owner and the SCRUM Master and Akash and Mark were appointed as The Team. Each main features of the game were done by the product owner and additional parts were added in from the team to further development. The group would have meetings frequently and contacted each other timely using Discord to communicate and work on the project and this allowed for easy management of what needed to be done and by who, and to communicate when issues arose or when certain things had been completed. The group also used GitHub Desktop to easily access the project folder and make changes easily from one person to another virtually, seeing who made what changes and in cases where problems arose the group could revert changes and go back to previous versions of the project.

# Implementation

The implementation of the game was a rather linear, most important and needed task first and then moving on to the next part of the game creation process. The first thing that the group developed was the sprites and designs needed for the prototype, sticking nicely to the design document in the sprites and designs likeness, the same drawings made from paint were turned into sprites such as the backgrounds and platforms, the platforms were created by making small blocks that could be turned into prefabs so the group could make multiple platforms of different sizes in width. The player sprites were made in paint also with different movements in different sprites so that they could become sprites for animations later in development. All sprites and prefabs were created by the group.

Example of some Sprites:



These sprites created were ones that were developed for testing and building of the game, with hope of changing the sprites later in order to create a more visually appealing game, however a member of the team in charge of sprites did not create these sprites in time and by the time they were made there was no longer much time left in development and so much had already been created so the sprites were unable to be changed, therefore the game was not as visually appealing as we had hoped, however it did mean the designs stayed true to the designs in the game design document.

Once the sprites were developed the group then created the unity project and began on building the scene and creating prefabs along with colliders that would be needed on the prefab platforms, enemy sprites such as the lava and icicle and also the right and left borders of the background sprites. For the background, two child game objects were made to be the right and left border collisions that would also act as triggers for kill player and the game over scene if the player collided with either border. This was then saved as a prefab so that the background and borders could be reused easily in the game.

The group then moved on to player control and creating the PlayerMovement script that would allow the player to move left and right using the left and right arrow keys on the keyboard and also press the space bar in order to jump, as stated in the design document character and controls section. The player sprite in unity itself was then added with a RigidBody2D and a polygon collider 2D, which was later changed to a box collider due to the polygon collider being easy to climb up platforms and also easily get stuck since the sprite was not a perfect shape. The PlayerMovement script was then added to the player “Jake” in unity, and after some quick testing, the player was able to move as intended. At first the controls were a little out of place as sorts, as the player would slide along the platforms and ground and it was difficult to control, but as the group continued to work on the script that issue was solved and many other things have been added to that script, for example double jump was a feature updated in order to improve game play and animations were also added to make the game feel less rigid and robust, and more like an actual prototype game. The final thing that was added to the player control and movement script was the use of a power up in the game, a small gem sprite was placed in random places in the game and if the player interacts with the power up gem then their speed and jump height was doubled for 3 seconds only.

Next on the to do list for the group, was platforms and levels. The first thing was making a prefab for one of the types of platforms as a starting ground in development. A polygon collider was added to the platforms to fit the shape of the platforms, which was why a box collider was not what the group chose. The platforms prefabs were then randomly placed on the scene so that they could be tested with the player.

After creating the solid and breakable platforms, we needed a way to trigger the breakable platforms to be destroyed as a gameplay mechanic. This was done through a script called “PlatformFall” which was written is such a way that if the player collided with the platform, the script would invoke a fall command after a set amount of time.

Following on from player movement, we needed to have a functional camera script so that the camera could track the players position within the game. The camera started off always following the player, however this created various issues such as the player not dying when falling as well as the issue of the player being soft-locked due to the platform destruction script being implemented at the same time. This resulted in a complete overhaul in how the camera functioned, with the new method placing the camera at the highest point the player has reached, allowing for game objects to be introduced as children of the camera to enable the player to die if they fall off screen. This was then further worked on later in the project when we added the effect of the camera shake to the game.

When coming to adding the final touches, we looked to add a camera shake to make the game more immersive and convincing. This was done through a script which made it so that the camera moved around in a random manner for some duration that could be altered within the script. After creating the script, we needed a way to keep it from interfering with the camera position when following the player. The solution we arrived at was creating an empty game object with the cameras original position with the camera follow script applied onto it and making the main camera a child of this empty game object we called “camera,” following on from this, the following script on the main camera was replaced by the new camera shake script. with the camera shake script seeming functional, we tested the script out by introducing an update to the script an invoked the shake when the button “T” was pressed, this test was successful. Following on from this we conducted a more practical test of the script by making the script function at the end of a timer as this was our idea for its final implementation within the game.

**Prototype Walkthrough**

**Play Testing**

**Discussion**

**Conclusion**