Rush N' Battle

Abstract - Rush N' Battle is a Unity-based First Person Shooter (FPS) game designed and developed by Raymond Do, Kevin Huang, and Peter Nguyen as a semester-long group project to enhance their proficiency in C Sharp (C#) programming. The primary objective of the project was to create an interactive application that facilitates user engagement. The developers honed their skills in Graphical User Interfaces (GUI), which provide players with visual elements to manipulate and connect with in-game.

I. Introduction

Before working on the game development process, two essential programs, Unity and Plastic SCM, were downloaded and installed. Plastic SCM was utilized in place of Unity Collaborate, and a workspace and repository were created to facilitate code-sharing among team members. While live collaboration on the same scene was not possible with Plastic SCM, it provided the added benefit of allowing users to revert changes in case of errors. The completed game features several key elements, including scripts, enemies, player movement, audio, animations, and a GUI with menus and interactables. The team's plan was to create an immersive first-person shooter game, featuring a variety of challenges and activities to keep players engaged. The main focus was on player functions and levels, so the team split the tasks to tackle those first.

II. Materials

To add in-game objects or materials for the project, the team utilized the Unity Asset Store. This is a marketplace where developers who are using the Unity Game Engine, can purchase or download tools and assets from others. The team took advantage of the free assets posted by other developers to provide inspiration for their own project. The assets used in Rush N' Battle from the asset store includes:

- *"Fantasy Kindom - Building Pack Lite"* by xiaolianhuastudio
- *"3D Low-Poly Chest"* by Tridentcorp
- *"CITY package"* by 255 pixel studios
- *"Environment Pack: Free Forest Sample"* by Supercyan
- *"Meshtint Free Polygonal Metalon"* by Meshtint Studio
- *"3D Monster Bomb!!"* by JKTimmons
- *"Potions, Coin And Box of Pandora Pack"* by Alexander Kotov
- *"Party Monster Duo Polyart PBR"* by Dungeon Mason
- *"RPG Monster Duo PBR Polyart"* by Dungeon Mason

These assets include decorations as well as enemies and animations for the team to incorporate into the game. Other assets were provided by Google Images and Youtube tutorials or created by hand. Through the Unity UI toolkit, the group were able to create a terrain that had volume and a path that directs the player in a general direction. AI navigation was also used to bake the terrain, so that the mobs and player can move around on the terrain.

III. Methods

Player - For the player, the team created multiple scripts that would be utilized. To be brief, these scripts allow the player to move, look, and to interact with items in the world. A camera was attached to the player which acted as the field of vision. It also includes a player UI, which consists of a health bar, damage overlay, ammo counter, and a crosshair.

Enemies - To make sentient enemies for the player to come across, the team created three different scripts for each movement state (chase, attack, idle). The scripts were connected through the animator function of unity. Each enemy had a separate script that included variables for their total health and how much damage they would deal.

Animations - Using Unity animator function the team connected each script with their own animation to create a more game-like scene for the players. Parameters were added to act as bool functions for each of the actions. Transitions were picked to complete the enemy's ai movements.

GUI - Regarding the GUI of the game, the team created multiple canvases. Each canvas had varying amounts of buttons and text-boxes, that when clicked, would run a script based on which button was pressed. Not only could the GUI be accessed with the mouse, but the team also allowed keyboard inputs to initialize GUI components.

Interactables - Interactable objects were created with the usage of animations as well as certain player scripts. Each new interactable would inherit from a parent class. These entities would possess the capability to either trigger prompts for the player or utilize a function from one of our player scripts.

Audio - Background music and sound effects were created with audio clips and modified in Unity's built-in audio mixer. This is mainly a quality-of-life addition that really allows the player to be immersed in the game.

IV. Result

After countless hours of trials and tribulations. After weeks of smashing the desk and ripping out hair follicles. Rush N' Battle was complete.

V. Conclusion

Rush N' Battle was an ambitious project by the team with many many **many** ideas. The team faced several challenges in order to piece everything together. These obstacles range from learning a new language (C#), using Unity and its built-in libraries, as well as other academic responsibilities each member had to focus on. As the deadline inched closer and closer, the team realized that certain ideas they had for the project would have to be scrapped due to the underlying difficulties. Despite the challenges faced by the team, they were able to showcase their dedication and hardwork with a working game for everyone to enjoy.