**Project Summaries & Learnings**

**Learning the Basics (Projects 1 & 2):**

Tutorial used: <https://www.youtube.com/watch?v=gB1F9G0JXOo>

**Project #1: Learning Unity**

Project Description: This was a basic 2d project in which I messed around with different aspects of Unity without any overarching goal (ex. Game idea, Application, etc.)

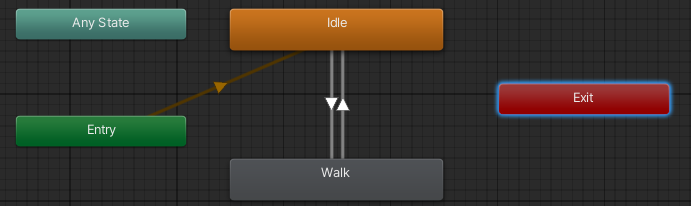
* Sprites and Sprite Sheets



* Sprite Sheets help save memory by simply putting all the images in one single file instead of many different ones.
* Programming
  + Classes (In Object-Based Programming)
    - Accessing classes and their methods in other programs (Super/Subclasses)
  + Basic Movement
  + Components
    - Common components and how to utilize them.
      * Rigid bodies, Animators, Colliders, Attaching Scripts, Audio, etc.
    - Accessing Components in Code (Integration with Unity)
  + Getters and Setters (I honestly forgot what this was)

**Project #2: Monster Chase**

Project Description: This is a 2d game where endless waves of enemies spawn from either side of the map, and the player must avoid them for as long as possible.

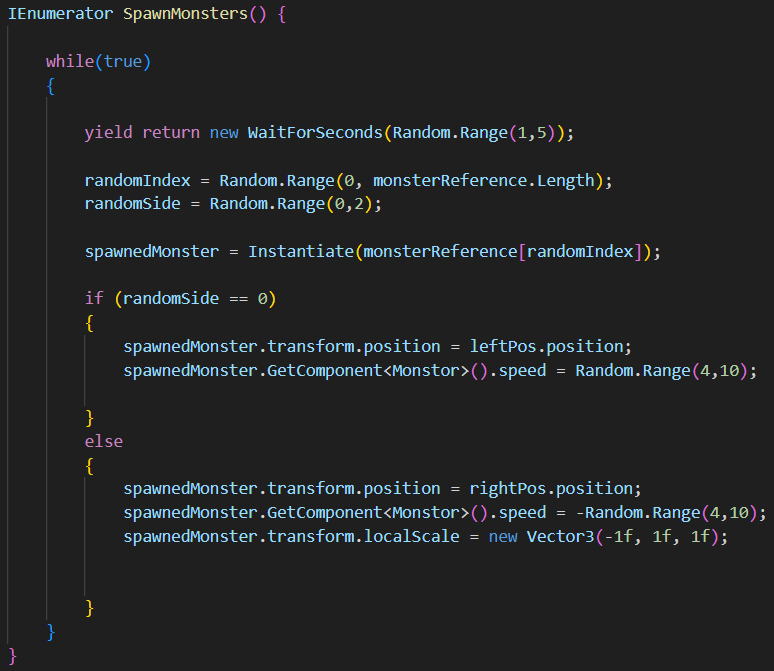
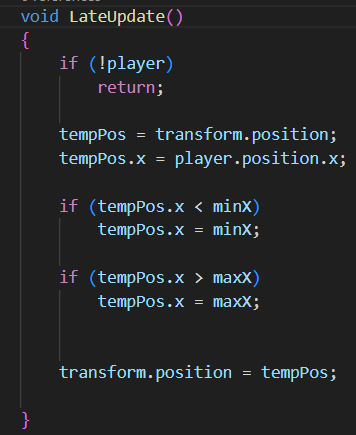
* Prefabs
  + Used to save objects for use.
* Animations
  + Helped animate the various characters in the game to give them life.
* Scenes:
  + Use of multiple scenes to transition between parts of the game.
  + In this scenario, it is used to transition from the Main Menu to actual gameplay.
* UI
  + Buttons that can be accessed in code.
  + Completely different scene to create a main menu.



* Programming
* A camera follows that is smooth and doesn’t stutter.

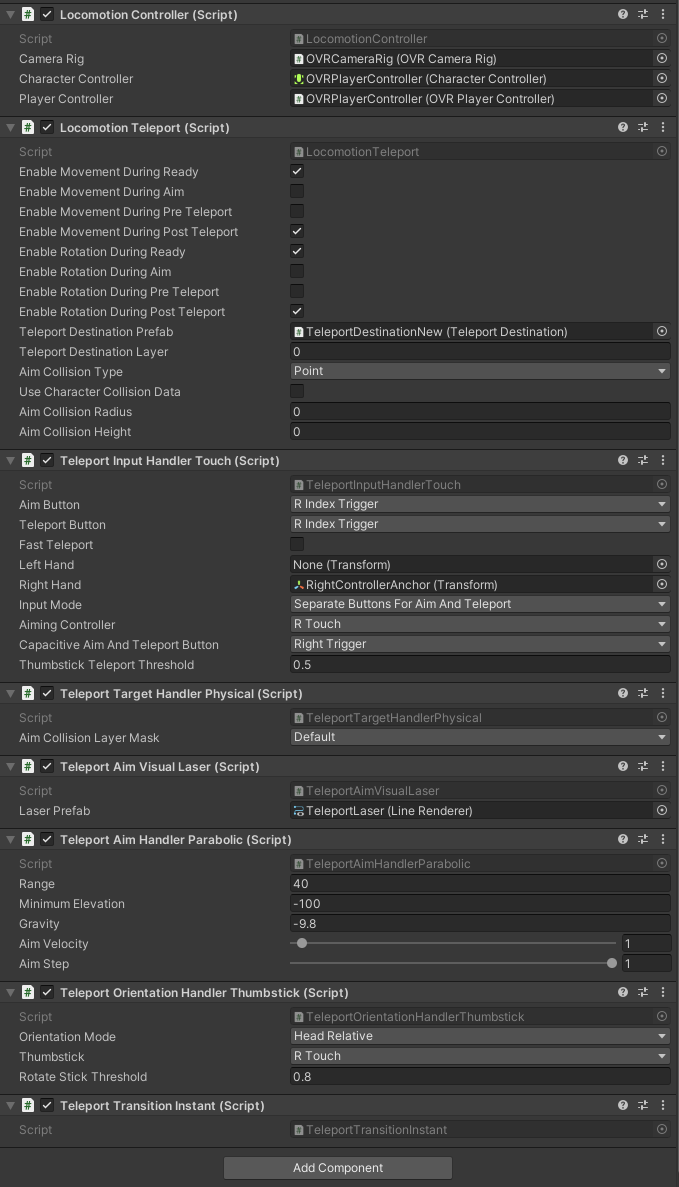
A tag-based collision system to validate certain actions and to act as a test for avoiding enemies.

Spawning Script that can spawn enemies at variable times.



**Main Project (Project 3):**

**Project #3: Flight Project**

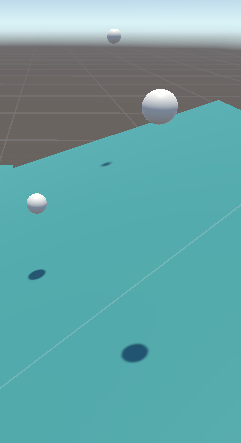
Project Description: This is a plane hangar simulation where the user can teleport and explore the hanger in VR. This project mostly focused on being able to create this application in Unity over scripting everything as that would take many months.

* Objects
  + Parenting Objects to each other allows for the organization of each of the various components.
* Components
  + I learned how to link different scripts/components to make them reference and access each other to make this experience work.
* Scripts
  + Separating each script into single purposes makes each part of the code easy to debug, access, and read.
  + This approach doesn’t come with any disadvantages either because you can link and reference other scripts in the Inspector panel.

\*Add a picture of the game in VR

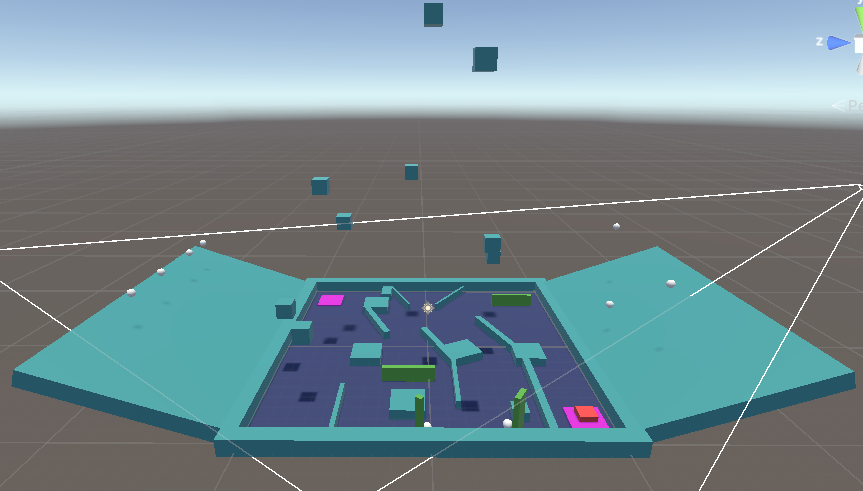
**Side Projects (Projects 4, 5, & 6):**

Learning Tool: <https://www.udemy.com/course/unitycourse2/?couponCode=ST4MT73124>

**Project #4: Obstacle Course**

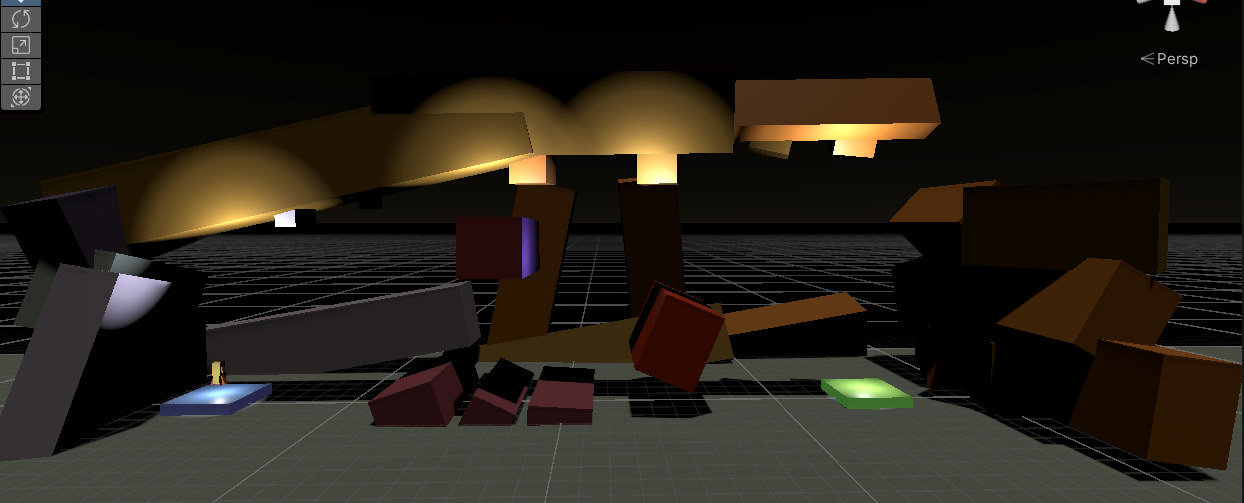
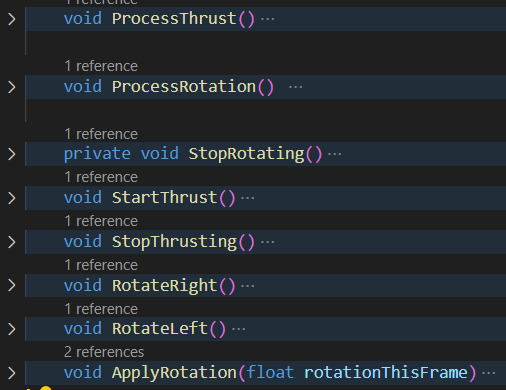
Project Description: This is a 3d game in which the player needs to navigate an obstacle course that uses physics to simulate many objects and to make the course a more enjoyable experience.

* Physics
  + Worked with Unity’s built-in physics engine.
  + Created a system of falling objects to hinder the player in this obstacle course.
* Collisions
  + Used a Tag-Based system for collisions.
* Game Building
  + Focused on building a course with different components to make it unique.



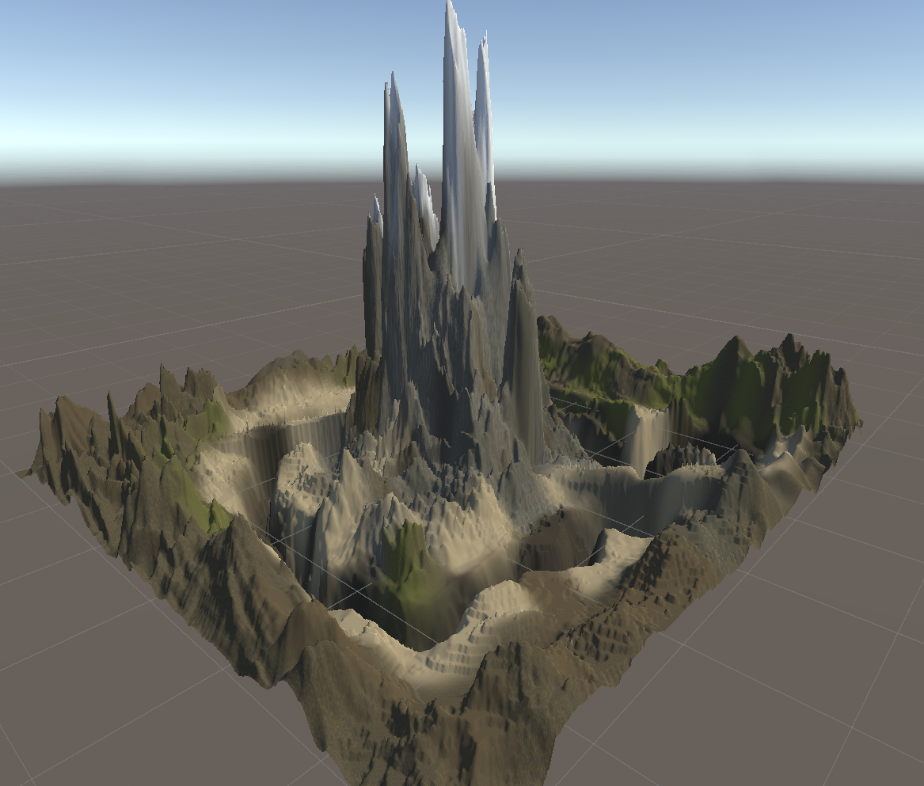
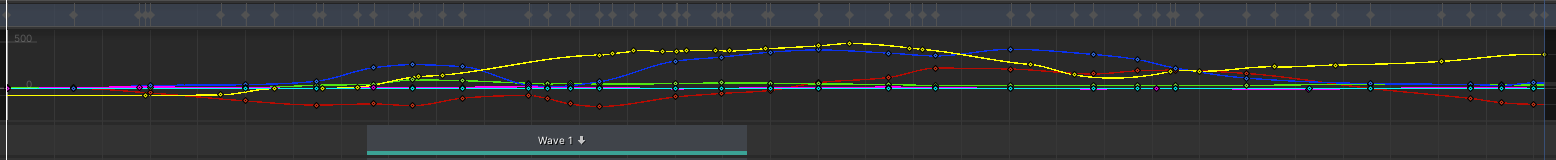
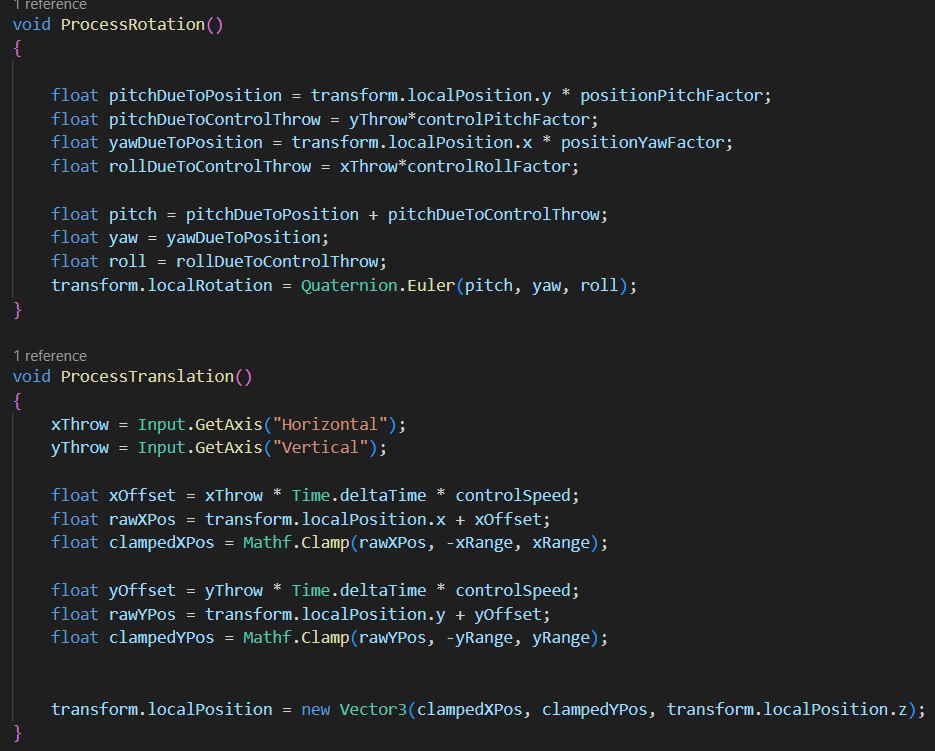
**Project #5: Project Boost (I should probably rename these to make them unique)**

Project Description: This is a 3d game in which players must control a spaceship through multiple unique levels. The player can only control the thrust and the direction the ship is going to make gameplay more interesting.

* Levels
  + Learned to make it so that the Player learns something new every single level instead of forcing it onto them simultaneously.
  + Make each level have similar components but unique in some way.
  + Made triggers to transition from one level to another.
* Lighting
  + Used very minimal bright lighting to keep the environment moody.
  + Used Point Lights to illuminate only portions of the level keeping part of the level dark.
* Unique Control System
  + The player could only control the thrust and the movement direction of their rocket.
* Audio
  + Sounds for any kind of action (ex. thrusting, crashing, and finishing the level)
* Particles
  + Used Unity’s particle system.
  + Built into code to activate particles on certain triggers (ex. Rocket thrusting)
* Code
  + Splitting up each action into multiple functions to keep code readable.

**Project #6: Argon Assault (I should probably rename these to make them unique)**

Project Description: This is a 3d game in which the player controls a ship that navigates a map in a predetermined path avoids terrain and destroys enemies to progress across the whole map.

* Terrain Building
  + Created a custom environment with Unity’s terrain-building tools.
* Timelines & Animations
  + Gave each moving object a set path in this game that looks dynamically interesting.
* Custom Assets
  + Used assets from the Unity Asset Store to have assets to use.
  + This is done to save time on aspects of an application so that you have more time for other things.
* Player Controller
  + Custom program that allows the character to move within the camera area while still flying along the set path.
  + In addition, the character is rotated using pitch, yaw, and roll to make the gameplay more interesting.