**Programming Learning Journal** Anish Pun

For my game prototype, I will be attempting to create a simple 3D action/platformer game. I am learning Unity and will be writing anything that went wrong and how I resolved the issue.

-I created a 3D Scene of primitives and implemented player movement. I was unaware of the drag properties that needed to be calculated. Player movement was accelerated when keys were held down and felt very slippery. I followed a tutorial that used a Raycast below the player to determine if they were ‘grounded’ based on the character’s height; this grounded state would initiate the rigid body drag in Unity(rb.drag).

-Jump feature added. The bool for readyToJump was private and therefore my player was unable to jump. Bool was made public, allowing me to check readyToJump– the player was able to jump. I later realised that I never set the base state of the bool to be true, and changed this in the script.

-The jump had low gravity, I changed the mass but this affected the player's movement as well. Going to Edit > Project Settings > Physics > Gravity and changing Gravity from 9.81 to 25 did the trick. If I was using a character controller for movement instead of a rigid body, I wouldn’t have to alter the gravity of the project. I could have just changed the mass. I could also lower the jump force but this would result in a floaty jump due to the lower gravity.

-Drag(rb.drag) continues to act in the air. Solved by creating a public float groundDrag and assigning it to the player if in a ‘grounded’ state. In the air, drag is 0. This makes the jump feel smoother and the player is able to slightly adjust jump trajectory..

-Made a layer for platforms but assigning the layer to primitives made them disappear. I duplicated the primitive as a child object and then assigned a visible layer to that platform. After doing all this, I noticed that the ground layer was not visible (crossed out). I realised that all those child objects were pointless as I could simply assign them to that new layer which was now visible. While the layer is far more efficient, I understand that the same result can be achieved in two different ways.

-I noticed that my character was sticking to walls in the environment when not intended. This happened as the movement input still produced a velocity towards the walls. To fix this, I created a new Physics material with friction multiplied by zero. I applied this to a capsule within the player object that was shorter and wider. The player no longer stuck to walls but the friction from the ground still worked on the player. Later, I simply applied the frictionless material to the player object and removed the separate collider. The separate collider would work for a character controller movement system, my rigid body system already calculated drag on the ground so the material applied to the player object will not affect general movement friction.

-I learned how to use the canvas system in Unity, creating UI such as HP Bars as well as menus and the buttons on those menus. Transitions between scenes are made using ‘UnityEngine.SceneManagement’, forgot to include that at the top. I learned that these are called libraries and access specific aspects of the Unity engine.

-Transitioning between scenes would crash the Unity player, I fixed this by going into build settings and including the new scenes as part of the build. I learned the unity player views the game view as if the game was built into a playable application. This is why I had to add those scenes to the build settings.

-Projectiles instantiated would linger after being thrown by the enemy capsule, solved by destroying gameObject after 2 seconds. This was a temporary fix, I later made the projectile disappear if it collided with the player. I realised that the projectile lingering would be very bothersome in tight spaces and the environment would be difficult to traverse. As a result, I made the projectile disappear on every collision.

-The cursor was locked and hidden at the start of the game. When making the pause menu, I didn’t consider this fact and the menu buttons were not selectable. Resolved by adding the cursor lock and hide lines when the game is paused and un-paused.

-When loading from the main menu to the level, the game would hide the pause UI but the game would be paused. Solved this by adding the un-pause code into void start.

-I realised I had redundant code as I had a button that called a subject that called another subject when I could just have removed the middle subject and directly connected the button and the subject.

-I learned that it was difficult to gauge the attack and sight range of the enemy turrets that I had created. I used Gizmos.DrawWireSphere to create an indicator for the range of their sight and attack.

-When creating scripts from the “Add New Component” section, it will not be created in the scripts folder, this created an issue where I had to rewrite the code and change the name of the script for the game to work. I found the stray script in the assets folder.

- I learned how to make simple animations in unity by setting up keyframes and manipulating the object according to those frames. Animations can be called by the script at certain points.

- I was instantiating a projectile from the player object but forgot that I had to aim the projectile according to the camera. I created a public Transform cam and added it to the instantiate origin, this made the projectile fire from where the camera was facing.

- I was creating a pickup system where the player would touch the object and destroy that object while checking a bool/ increasing an integer for that object. For the throwing knife in my game, that became an issue as the knife would instantiate inside the player and then immediately destroy itself. This problem could be fixed by disabling the knife’s collider for half a second but I would rather change the pickup system.

- I iterated on the pickup system by implementing a ray-cast check in front of the character that would determine what I picked up. This was activated by the push of a button. While it was working, smaller objects such as the throwing knife’s colliders were too low to the ground for the ray to detect. I fixed this by increasing the height of the collider but the item would act as a wall. due to the large collider.

- I reiterated the pickup system by using a physics overlap sphere that destroyed other colliders tagged “item” when I pressed E. This would both pick up the item and check the bool I needed. This final iteration was ideal as it would check for tags and I could use this for numerous future pickups.

- I ran into issues when instantiating the throwing knife from the camera, this would knock the camera back and shoot only in the direction the camera was facing in the beginning. This was solved by creating a separate empty object as the spawn point for the knife.

- The knife bounces off enemies when it strikes them, I would like for the knife to lose all momentum on contact with the enemy. There is also no crosshair, so the knife is difficult to aim. I added a simple Indicator to the HUD on the same canvas as the health bar as the crosshair. For the knife, I set it so the velocity of the knife would be set to 0 on collision, allowing the player to retrieve the knife where the enemy was.

- Added a UI element to indicate when the player had the knife throwable and had difficulty toggling the UI on and off when placed in the same canvas as the health. Canvas child objects are not detected by public Canvas, so I created a new canvas specifically for the knife UI. The knife UI would toggle while checking for the bool hasKnife from the same script.

- In the code for the pick-up system, I realised that the interaction only references the knife’s interactions as an “Item”. I had to separate the knife and the item tags for other items to function and not just be destroyed.

- On collision, my knife prefab would do a spin and add a new 5f velocity to the y axis, this would add a visual flair for the collision. Unfortunately, when colliding with a vertical wall, this would make the knife collide multiple times and it would start climbing the wall. Fixed it by having the collision check for the “Enemy” and “Level” tag.

\*The added velocity had strange effects at times and had unnatural motion, I changed the velocity to 0 so the knife would still stop on impact. I added a line to the script so the object would always face the direction of its velocity. This still created a flip as the knife would face the floor as it would fall. I also added a feature to where the knife would freeze all rigid body constraints on collision with the ground tag. This created an effect as if the knife was stuck into the floor.

- I noticed there was a white circle around the player that was visible when playing the game. The scene view did not have this white circle however. I changed the Physics interactions in project settings, unchecking the interactions between items and the player. This removed the white lines in the game view.

- When setting up the checkpoint system, I realised that I had to change the Gameover scene by bringing it into the main level scene. The Gameover scene had many interactions with the Level Scene, it was more efficient to have them both in the same scene. I moved the canvas from the Gameover scene and also changed the button interactions to accommodate for the change.

-The checkpoint system would not register checkpoints, respawning the player back at the start. Checkpoint collider had to be a trigger and had to change the effects of death. Time scale had to be zero now that Gameovers were on the same scene, making the death screen an alternate pause screen.

-Added a material change to the checkpoint so the player would know when the checkpoint is activated. Wasn’t working initially as the default state was the Activated state, simply switched the materials around. Also added a separate box collider to the checkpoint as the initial one was a trigger and players would phase through it.