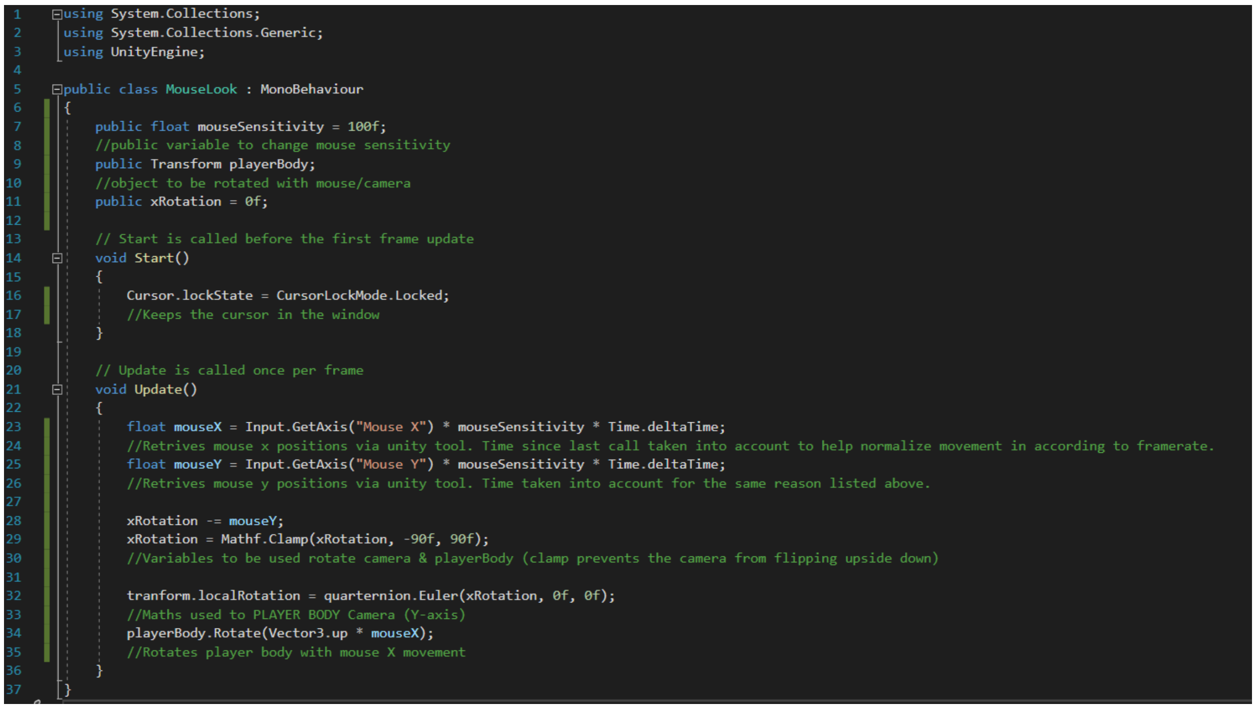
BEFORE STARTING

Create a new (3D) project with a sample scene with an interactable environment. This should include a floor and terrain like objects for our character to interact with. It is also advisable to have rudimentary lighting set up.

1: PLAYER OBJECT MODELING AND CAMERA MOVEMENT

Create an empty object and name it something memorable like “First Person Player”. Add a Character Controller component. Enabling Gizmos will enable you to see an outline of the controller in the developer view to see if, later on, that the character model is successfully rotating. Edit the Radius and Height appropriate to the scene you are working with. Create a cylinder under your First Person Player with its X and Y values being double of the empty object’s radius and half the height. This will act as your collision detection as well.

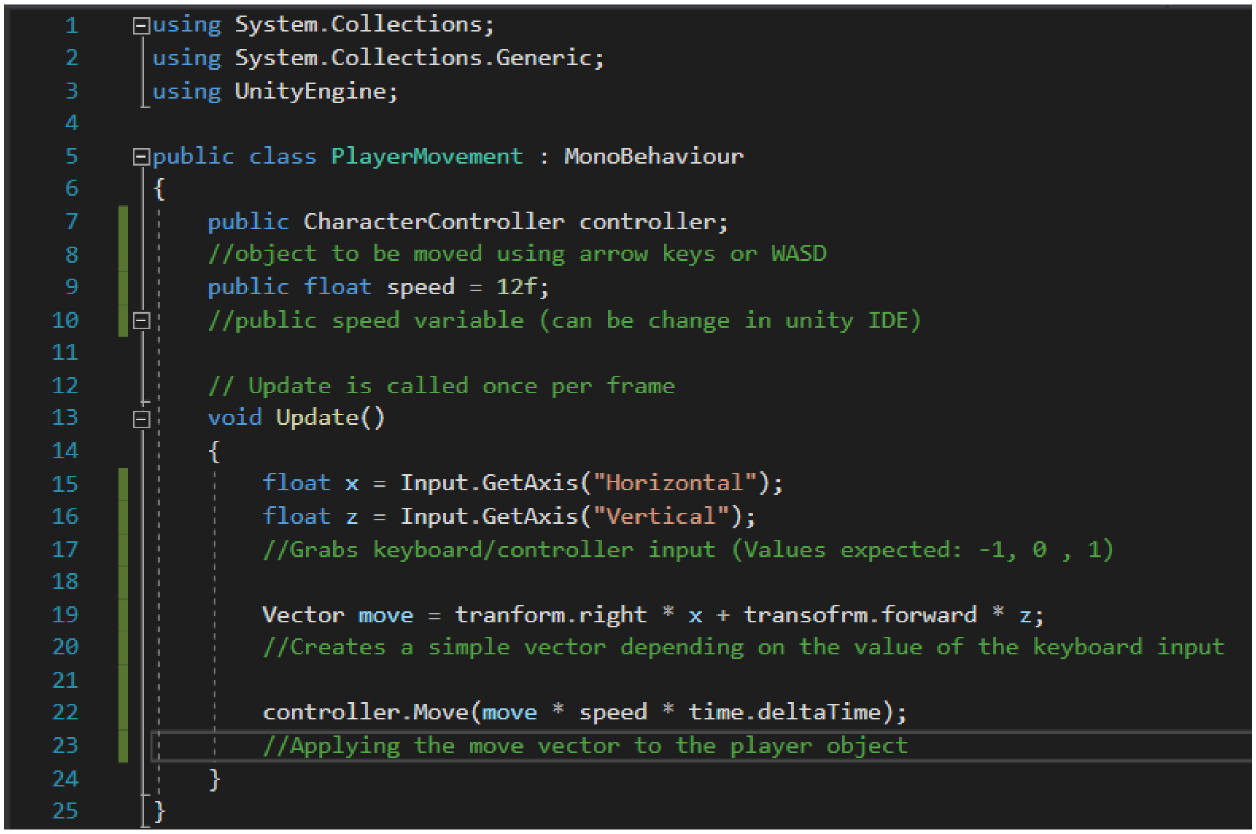
Move the camera under the First Person Player object. Reset its’ transform and move it vertically to just under the top of our cylinder, leaving it clipped inside the visible object. Add a new component script (MouseLook in this example). Code is provided below:



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MouseLook : MonoBehaviour
{
    public float mouseSensitivity = 100f;
    //public variable to change mouse sensitivity
    public Transform playerBody;
    //object to be rotated with mouse/camera
    public xRotation = 0f;

    // Start is called before the first frame update
    void Start()
    {
        Cursor.lockState = CursorLockMode.Locked;
        //Keeps the cursor in the window
    }

    // Update is called once per frame
    void Update()
    {
        float mouseX = Input.GetAxis("Mouse X") * mouseSensitivity * Time.deltaTime;
        //Retrives mouse x positions via unity tool. Time since last call taken into account to help normalize movement in according to framerate.
        float mouseY = Input.GetAxis("Mouse Y") * mouseSensitivity * Time.deltaTime;
        //Retrives mouse y positions via unity tool. Time taken into account for the same reason listed above.

        xRotation -= mouseY;
        xRotation = Mathf.Clamp(xRotation, -90f, 90f);
        //Variables to be used rotate camera & playerBody (clamp prevents the camera from flipping upside down)

        tranform.localRotation = quarternion.Euler(xRotation, 0f, 0f);
        //Maths used to PLAYER BODY Camera (Y-axis)
        playerBody.Rotate(Vector3.up * mouseX);
        //Rotates player body with mouse X movement
    }
}
```

2: PLAYER OBJECT MOVEMENT

Under our player object, add a new component script (PlayerMovement in this example). Code is provided below:



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlayerMovement : MonoBehaviour
{
    public CharacterController controller;
    //object to be moved using arrow keys or WASD
    public float speed = 12f;
    //public speed variable (can be change in unity IDE)

    // Update is called once per frame
    void Update()
    {
        float x = Input.GetAxis("Horizontal");
        float z = Input.GetAxis("Vertical");
        //Grabs keyboard/controller input (Values expected: -1, 0 , 1)

        Vector move = tranform.right * x + transofrm.forward * z;
        //Creates a simple vector depending on the value of the keyboard input

        controller.Move(move * speed * time.deltaTime);
        //Applying the move vector to the player object
    }
}
```

At this point, the player movement and camera should be functional. Mess around with Character Controller component values in order to better interact with the scene.