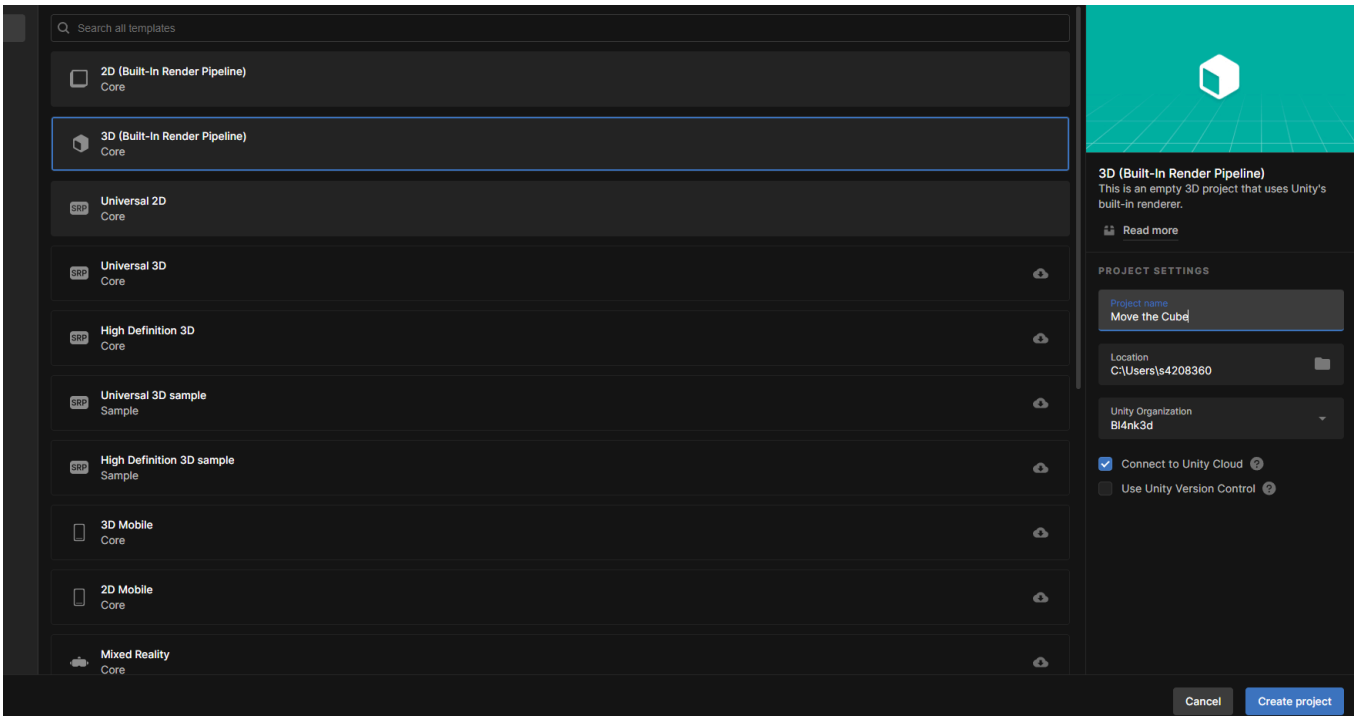
In this tutorial I will be presenting how to code simple 3D movement within a space

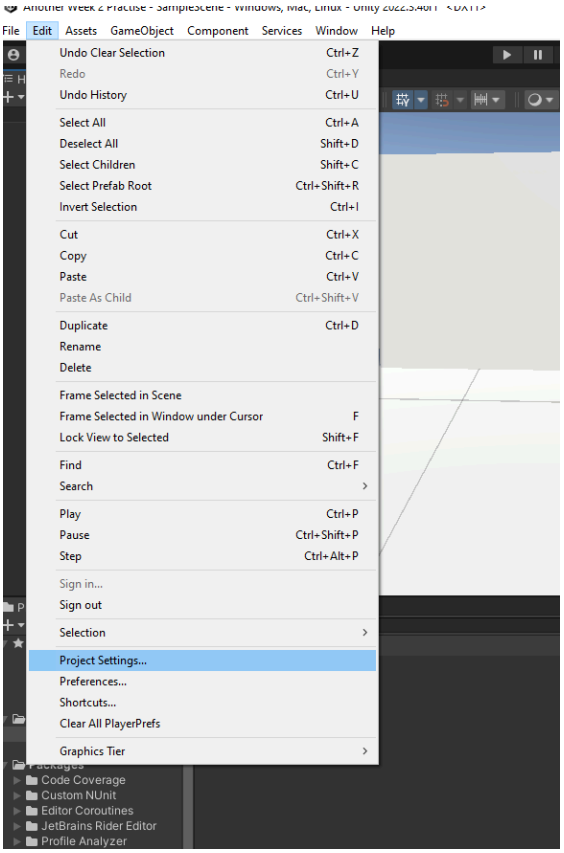
Beginning with this tutorial does not require much previous knowledge of coding essentials and can be followed regardless of previous coding backgrounds.

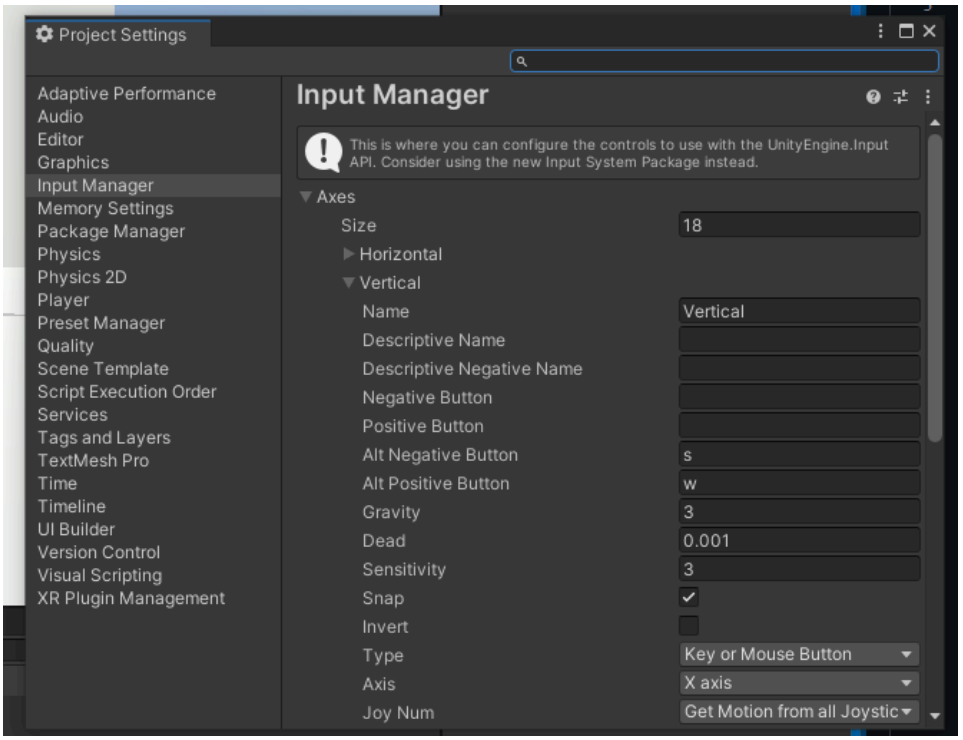
For the first step we will be setting up our project so that the basic rules and functions of the scene will play out how we want.

First create a new project as a 3D Project and name it Move the Cube



After the project has been opened head to the Edit tab at the top left of the screen and go to Project Settings. Here we want to edit what keys trigger movement within the code. Within the Project Settings open the sub tab Vertical and remove the primary buttons associated with it. This will ensure that our code does not clash with pre-set buttons.

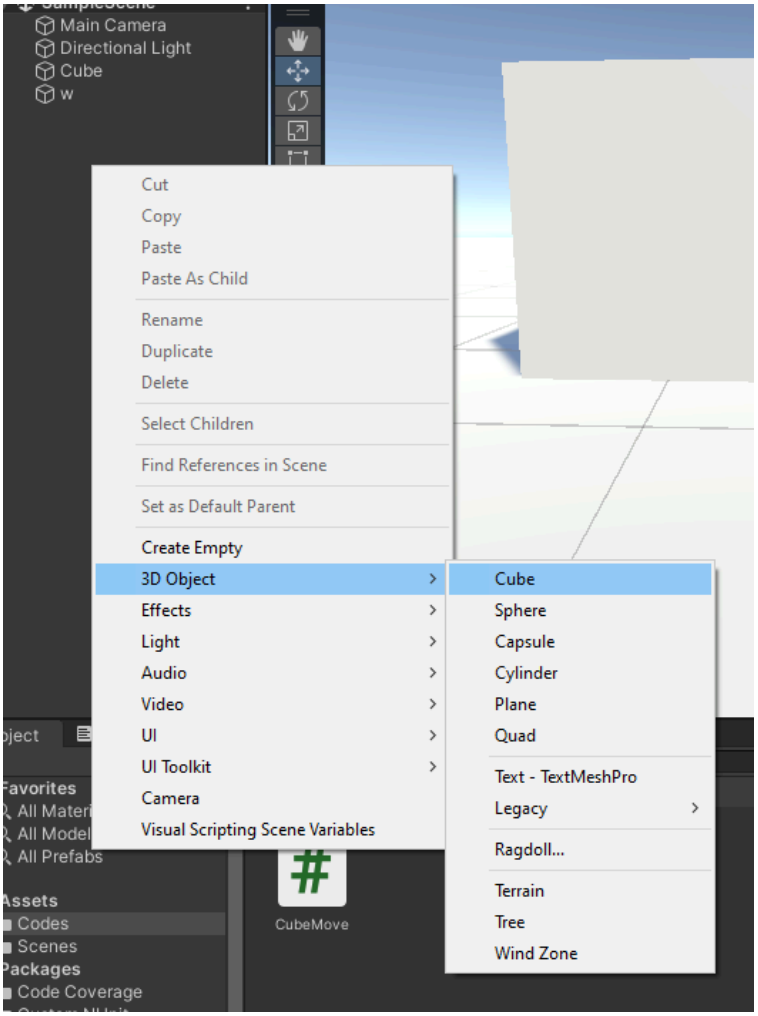


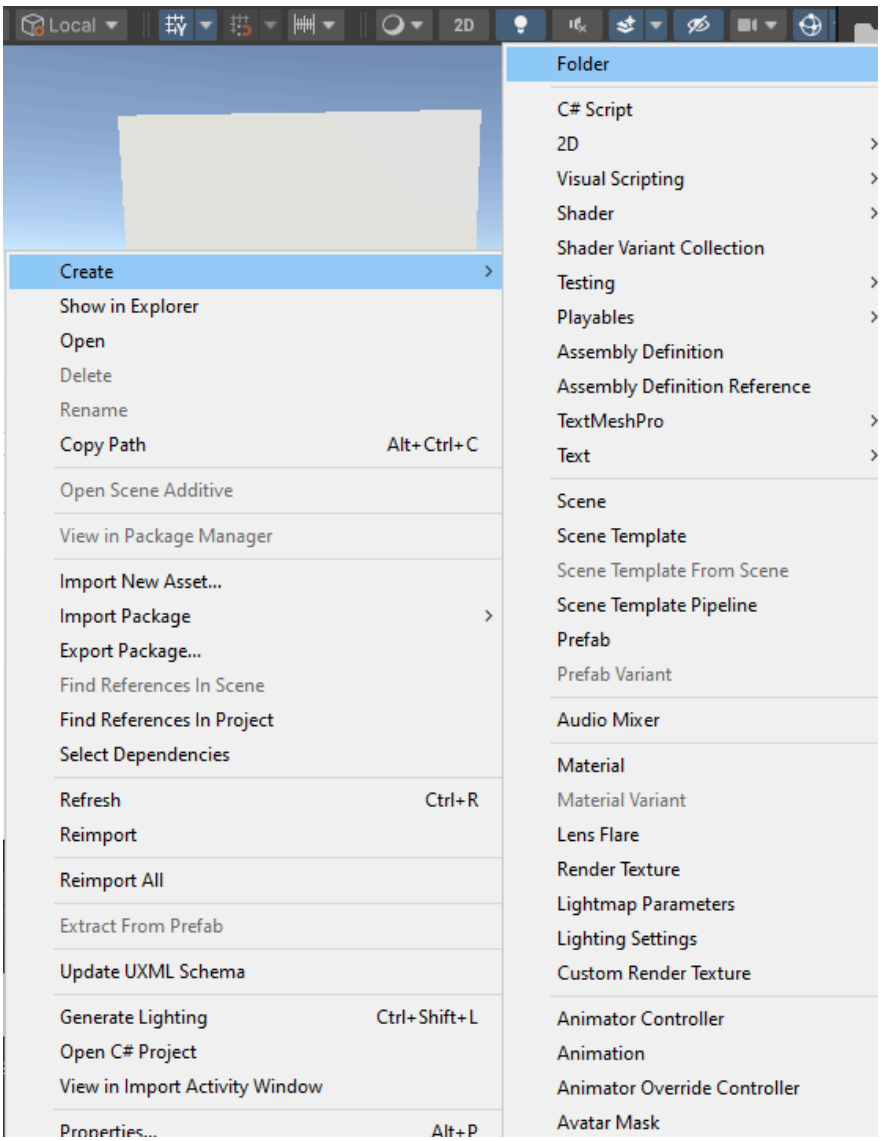


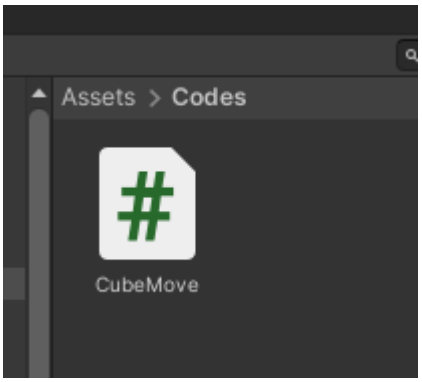
Close the tab and we can begin making our mover.

//For the next step we will be looking at how to add, create and edit 3D items and assets in our project.

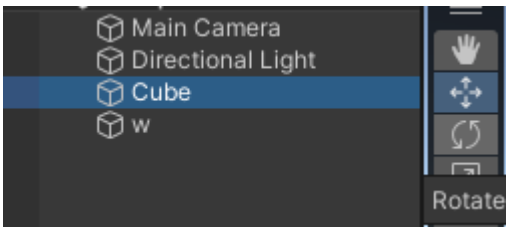
In the Scene box right click and click 3D Object - Cube to place a cube within our scene. Repeat this with a Plane and place it below our cube. This will be so we can clearly see the ways our cube moves. Next go to our Folders and create a new Folder called Scripts, this is where we will keep our code that determines how our cube works. Using the same create menu, make a new script within our new folder and call it CubeMove and open it.



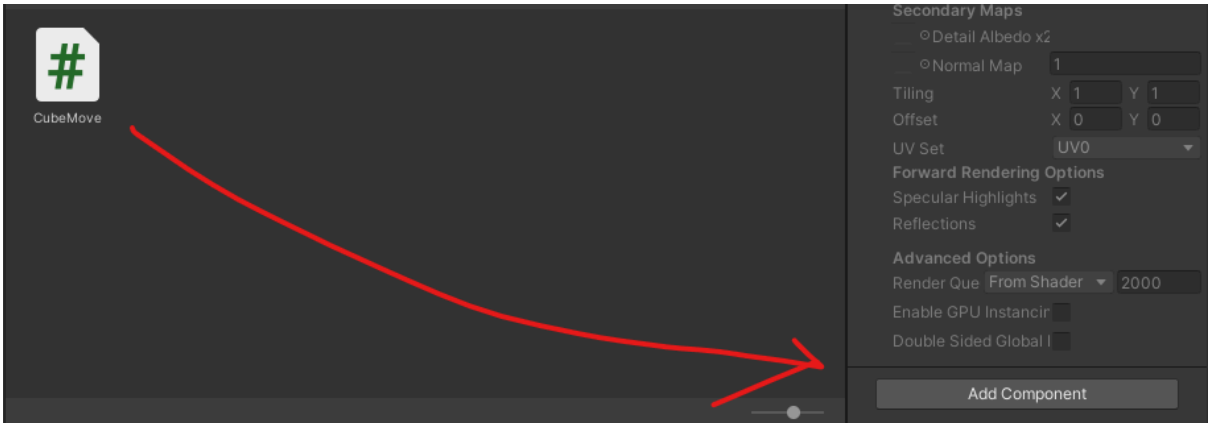




Before we begin to edit what is in the script we should first assign it to the cube we wish to move. Select the cube in the hierarchy or just click on it within the scene and find Add Components.



Next drag the script file over the Add component button to assign it directly to the object



//The next and final step of the process is putting together the code that will perform the task that we wish.

Now we can start coding. Go back to the open script and within the void update insert the following code. This code will determine that when an Input relating to a dimension is pushed its position along that respective axis will be changed, moving it around as the player desires. Make sure to add the public float above the start line.

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

0 references
public class CubeMove : MonoBehaviour
{
    public float speed = 5f;
    // Start is called before the first frame update
    0 references
    void Start()
    {

    }

    // Update is called once per frame
    0 references
    void Update()
    {
        float inputX = Input.GetAxis("Horizontal");
        Debug.Log(inputX);
        transform.Translate(inputX * Time.deltaTime * speed * Vector3.right);

        float inputY = Input.GetAxis("Vertical");
        Debug.Log(inputY);
        transform.Translate(inputY * Time.deltaTime * speed * Vector3.up);
```

You will notice with this code however that only horizontal movement is available with the arrow keys and this is because the default keys to move along that axis were removed earlier. However it should work when using W and S.

Underneath the previous code insert the following code as seen.

```
    if (Input.GetKey(KeyCode.DownArrow) == true)
    {
        transform.Translate(new Vector3(0,0,-1) * Time.deltaTime * speed);
    }

    if (Input.GetKey(KeyCode.UpArrow) == true)
    {
        transform.Translate(new Vector3(0, 0, 1) * Time.deltaTime * speed);
    }
}
```

This code will be what allows our cube to move along the plane in a way that previously could not and this movement will be assigned to the up and down arrows. Make sure that the code reads GetKey and not GetKeyDown as this will mean the movement only triggers when the press occurs and not when it is held down.

If you wish to change any of the button layouts for the cube simply either change the keycodes within the script or use the Project Manager to change the keys assigned to movement.

After this save the code, play the scene and enjoy your flying cube.