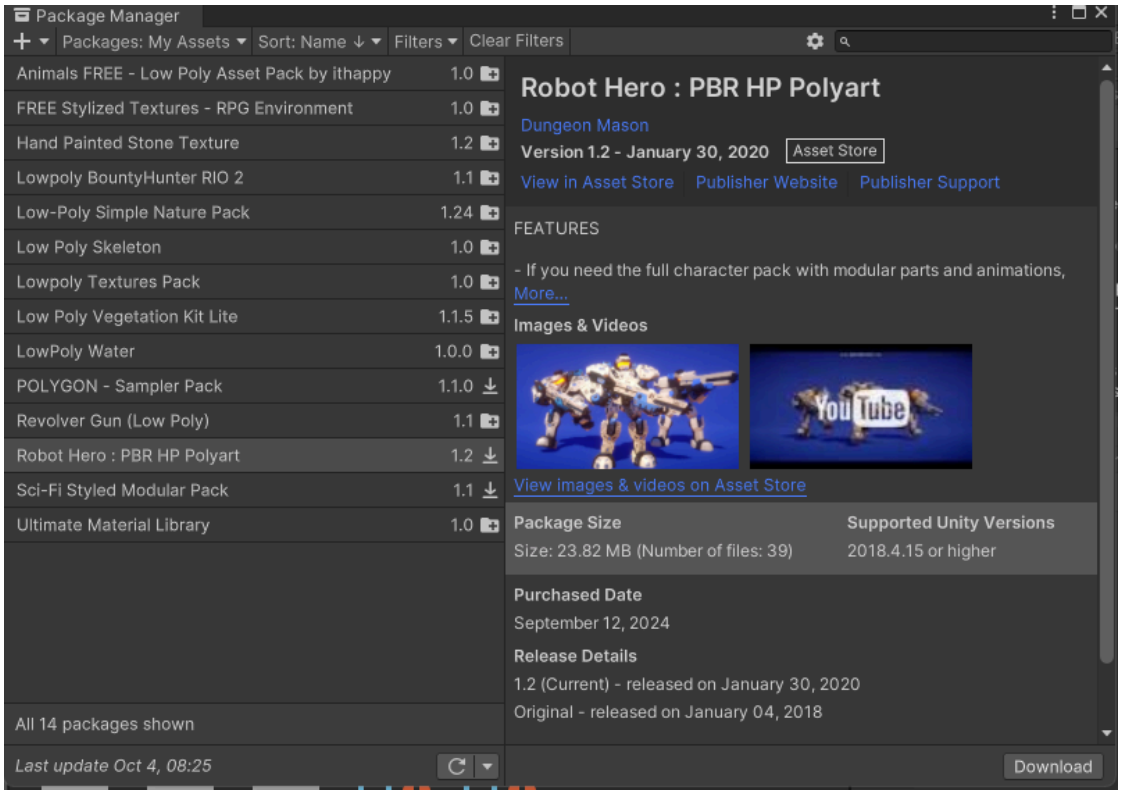
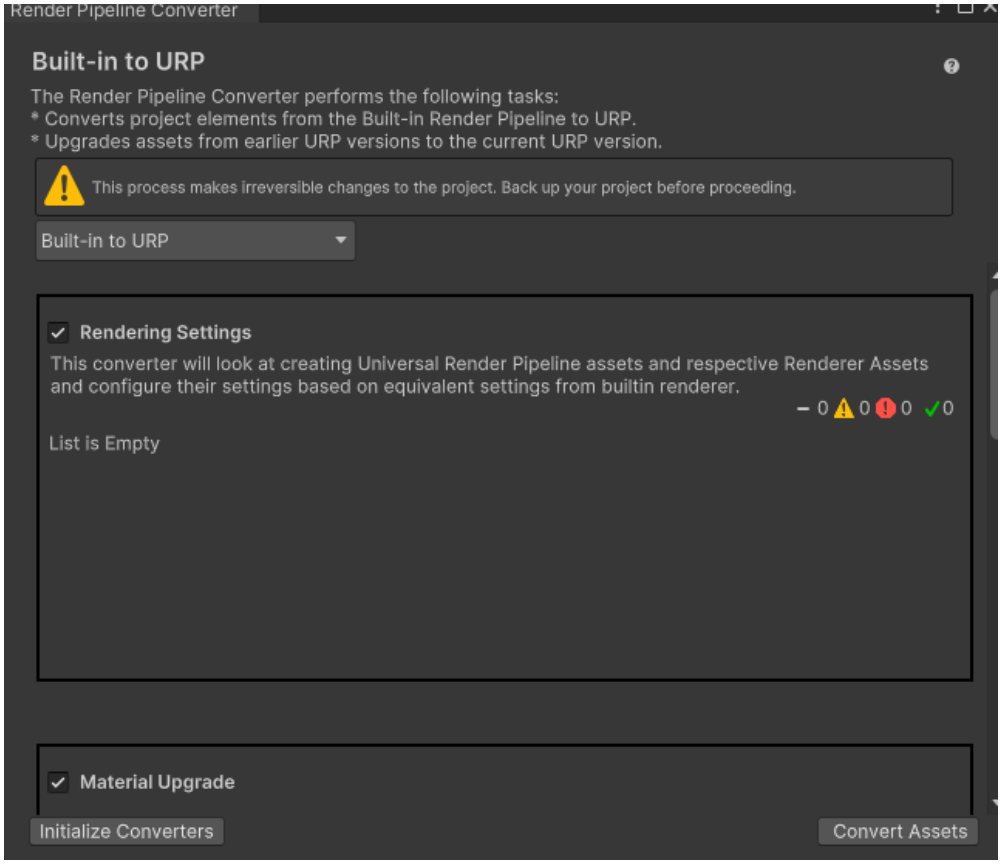
Challenge - 04 Bullet Time

For this challenge we'll be shooting a gameobject from an asset.

To start first we import our assets into unity using the package manager.



Then we convert the project elements to URP utilizing the Render Pipeline Converter to eliminate purple textures.

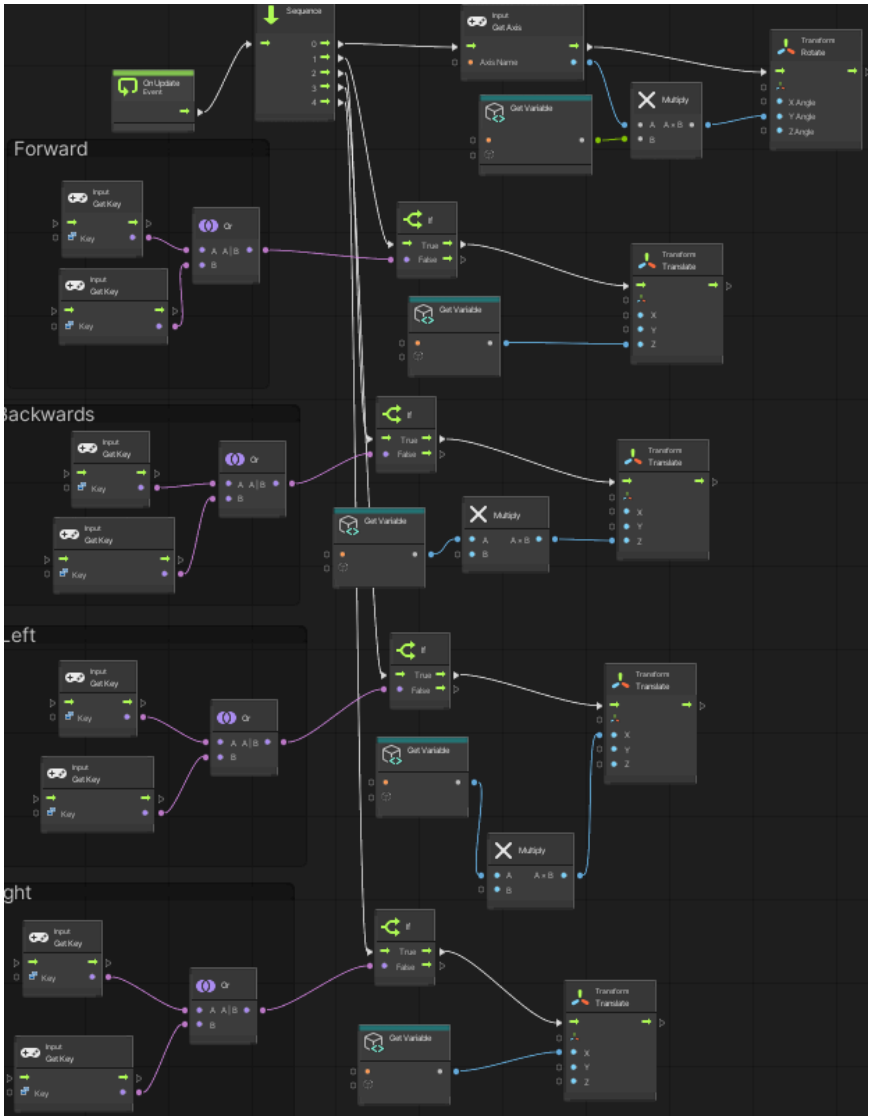


We put the first asset into the scene.

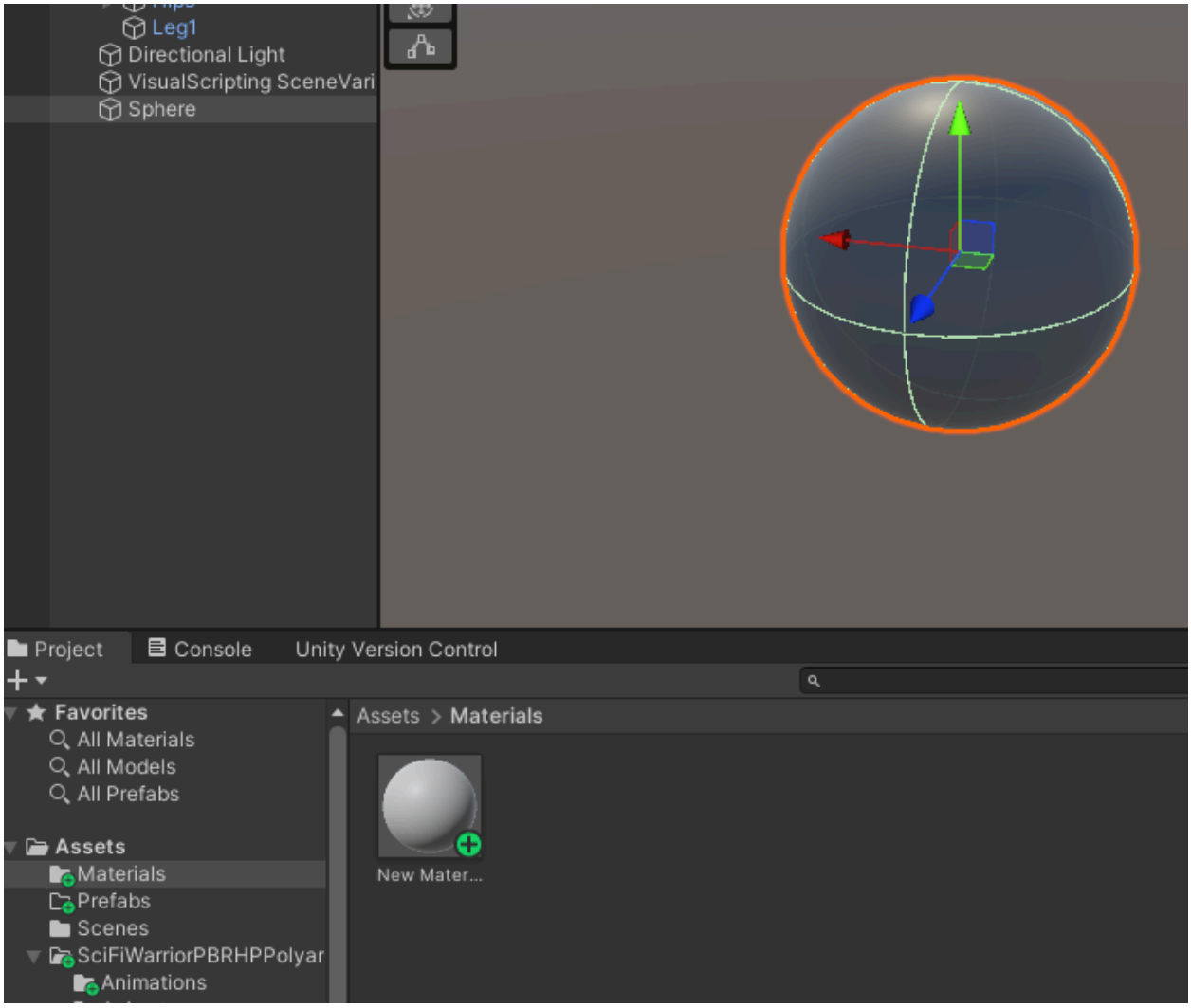
After that we add basic movement and rotation scripts for the asset.

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

public class PlayerMovement : MonoBehaviour
{
    public float speed = 5.0f; // Set a default speed value greater than 0 for movement

    public float rotationSpeed = 1.0f;

    // Update is called once per frame
    void Update()
    {
        float mouseX = Input.GetAxis("Mouse X");

        transform.Rotate(0, mouseX * rotationSpeed, 0);

        // Forward movement with W or Up Arrow
        if (Input.GetKey(KeyCode.W) || Input.GetKey(KeyCode.UpArrow))
        {
            transform.Translate(0, 0, speed * Time.deltaTime);
        }
        // Backward movement with S or Down Arrow
        if (Input.GetKey(KeyCode.S) || Input.GetKey(KeyCode.DownArrow))
        {
            transform.Translate(0, 0, -speed * Time.deltaTime);
        }
        // Left movement with A or Left Arrow
        if (Input.GetKey(KeyCode.A) || Input.GetKey(KeyCode.LeftArrow))
        {
            transform.Translate(-speed * Time.deltaTime, 0, 0);
        }
        // Right movement with D or Right Arrow
        if (Input.GetKey(KeyCode.D) || Input.GetKey(KeyCode.RightArrow))
        {
            transform.Translate(speed * Time.deltaTime, 0, 0);
        }
    }
}
```

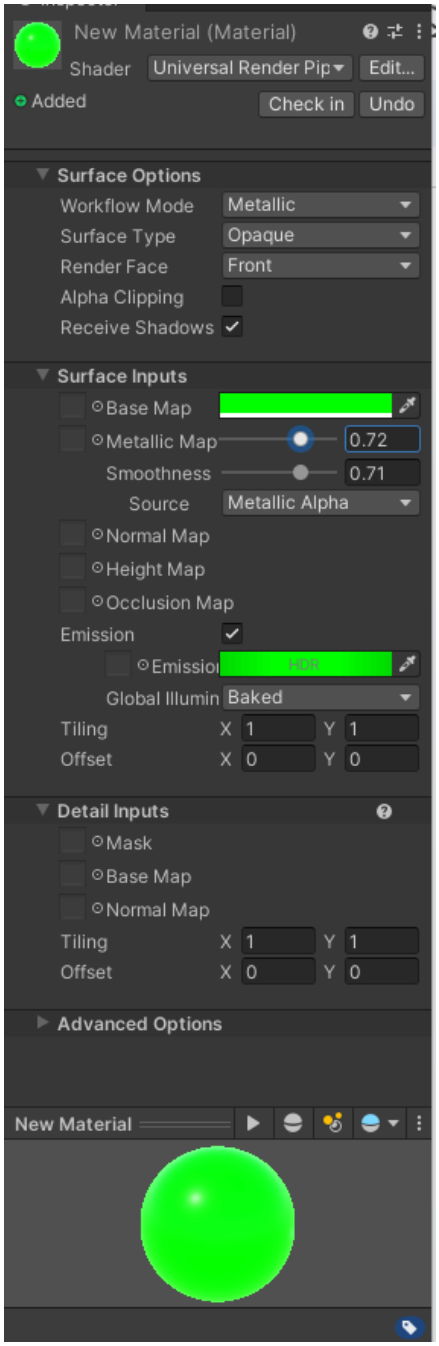
We then do the same script but in a graph.



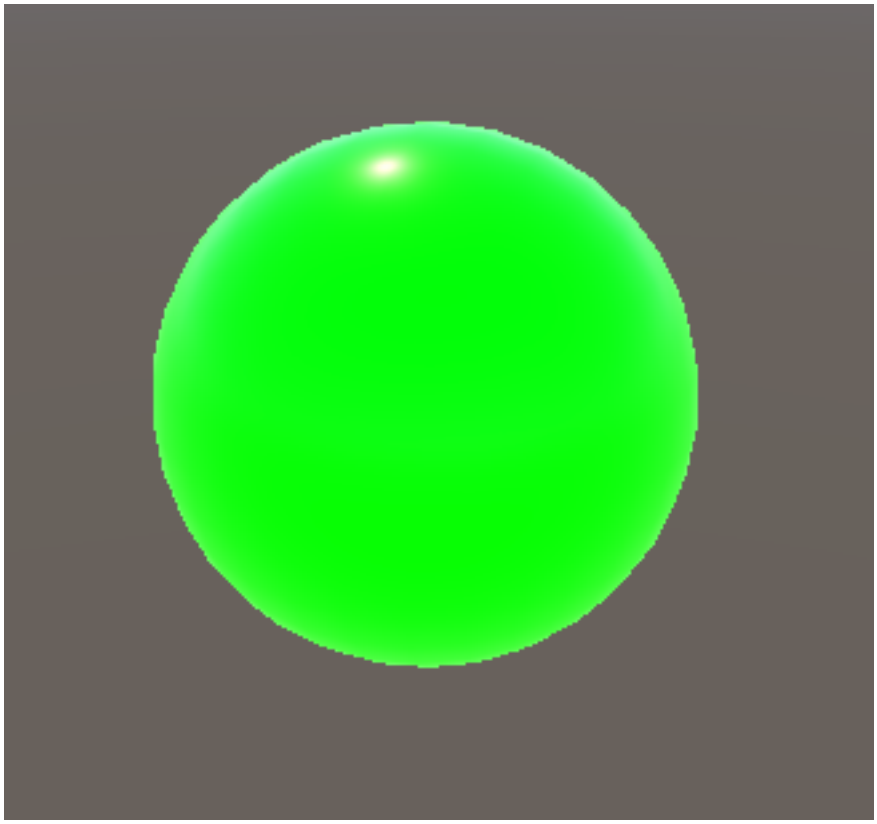
Now to create a bullet we make a new material and insert a sphere into the scene.



We edit our material to our liking.



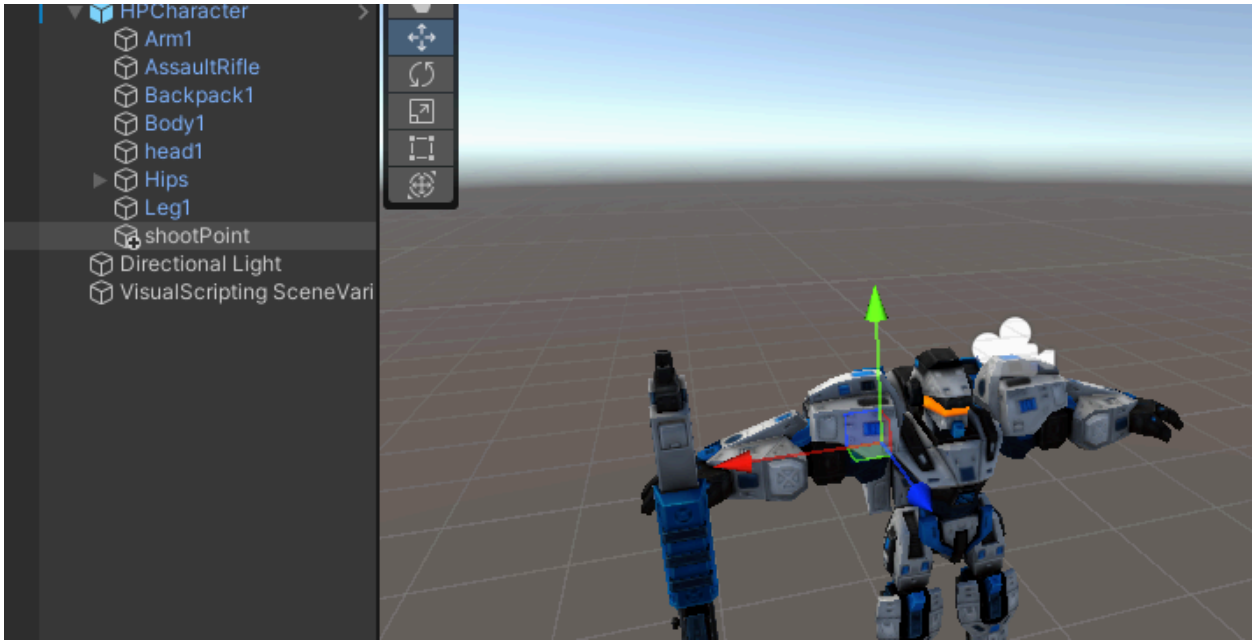
Here is the sphere after the material has been applied.



We write a script to make the bullet travel forward.

```
7       public float speed;
8
9       // Updated is called onece per frame
10      void Update()
11      {
12          transform.Translate(0, 0, speed * Time.deltaTime);
13      }
14  }
```

We create an empty object and attach it to the asset as the point the asset will be shooting from.

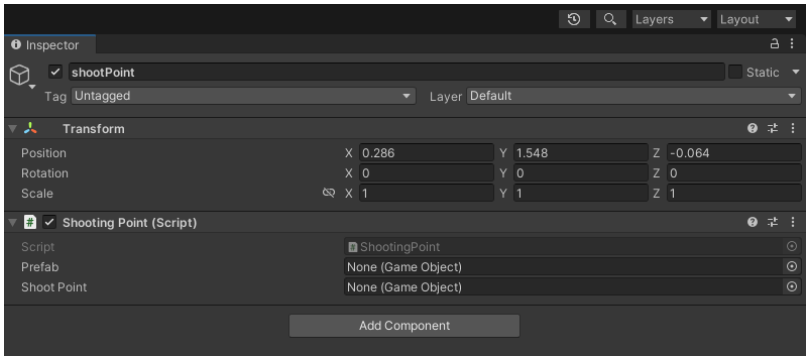


Now that we have our shooting point we add a script to it. This script will create the bullet, from the prefab, every time we push left click.

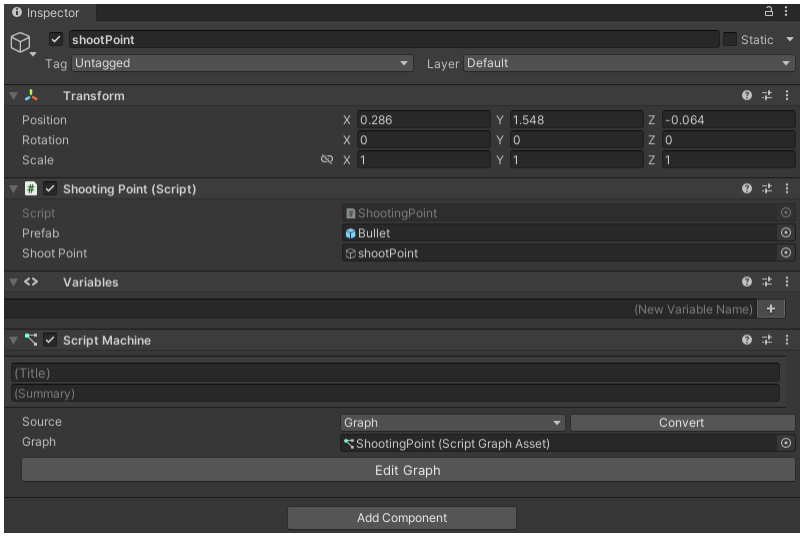
ShootingPoint.cs

```
Assets > ShootingPoint.cs > ShootingPoint
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4
    0 references
5   public class ShootingPoint : MonoBehaviour
6   {
        1 reference
7       public GameObject prefab;
        2 references
8       public GameObject shootPoint;
9
10      // Update is called once per frame
        0 references
11      void Update()
12      {
13          if (Input.GetKeyDown(KeyCode.Mouse0))
14          {
15              GameObject clone = Instantiate(prefab);
16              clone.transform.SetPositionAndRotation(shootPoint.transform.position, shootPoint.transform.rotation);
17
18          }
19      }
20  }
```

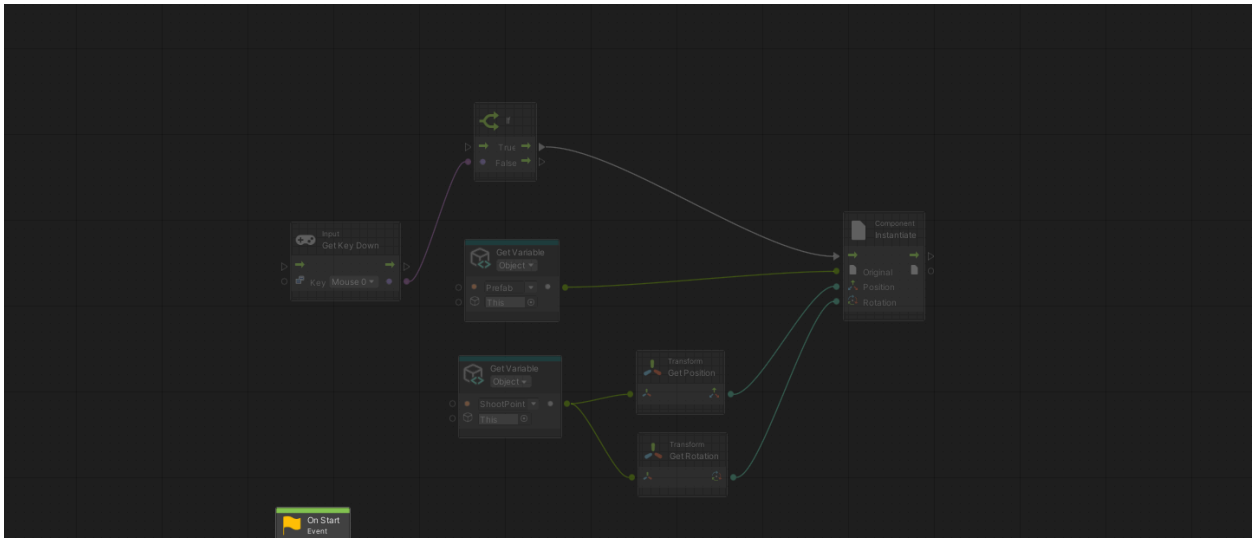
When we go back to our Unity window, we can see that we need to give it the prefab and position from where the script will work from. To do this we would simply drag and drop the prefab a shooting point object.



When we have correctly added them, it would look like this.



Now we proceed with creating the visual script that would complete the same action



Finally, we added a terrain to set our character in