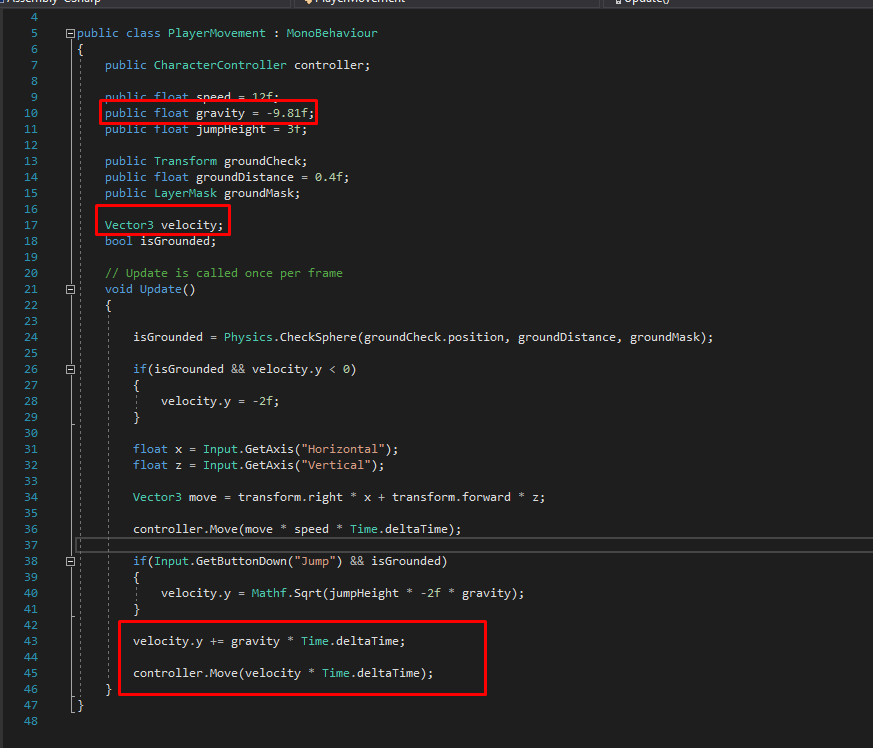
**Tutorial 3 – Gravity and Jumping**

**Gravity and Ground Checks**

* Since gravity is a force that is applied over time, the speed of the fall will increase depending on the time you have fallen so far. To simulate this, we need to add a velocity variable. To do that, we also need to be able to check when the player is grounded, by adding a “ground check”
* Open up the “PlayerMovement” script from before and create a Vector3 public class which will store our velocity. You then want to create a script in the update method that applies gravity from the Y axis.

To be able to adjust the gravity to your liking within Unity, you can create a public float that allows you to change the gravity strength. To the add the velocity to the player, we want to use the controller.Move script again, but this time based on velocity.



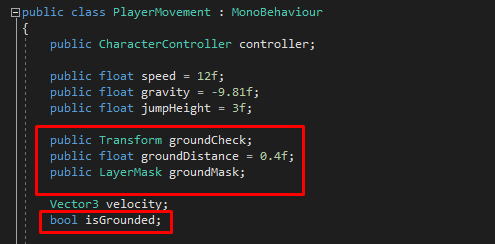
* To make sure the velocity resets once the player lands and does not constantly keeps increasing, we need a ground check using the “checkSphere” method, which tells the script when the player has landed, so the velocity can reset.

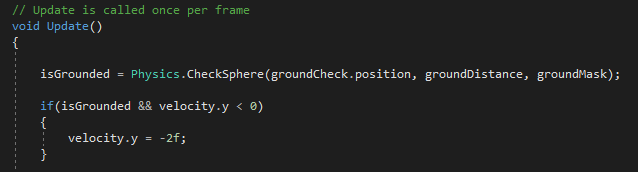
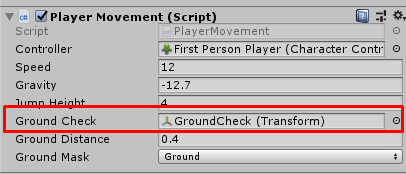
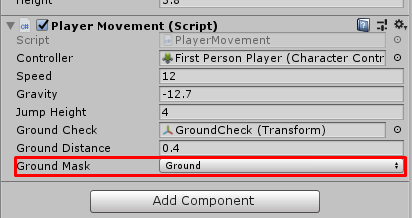
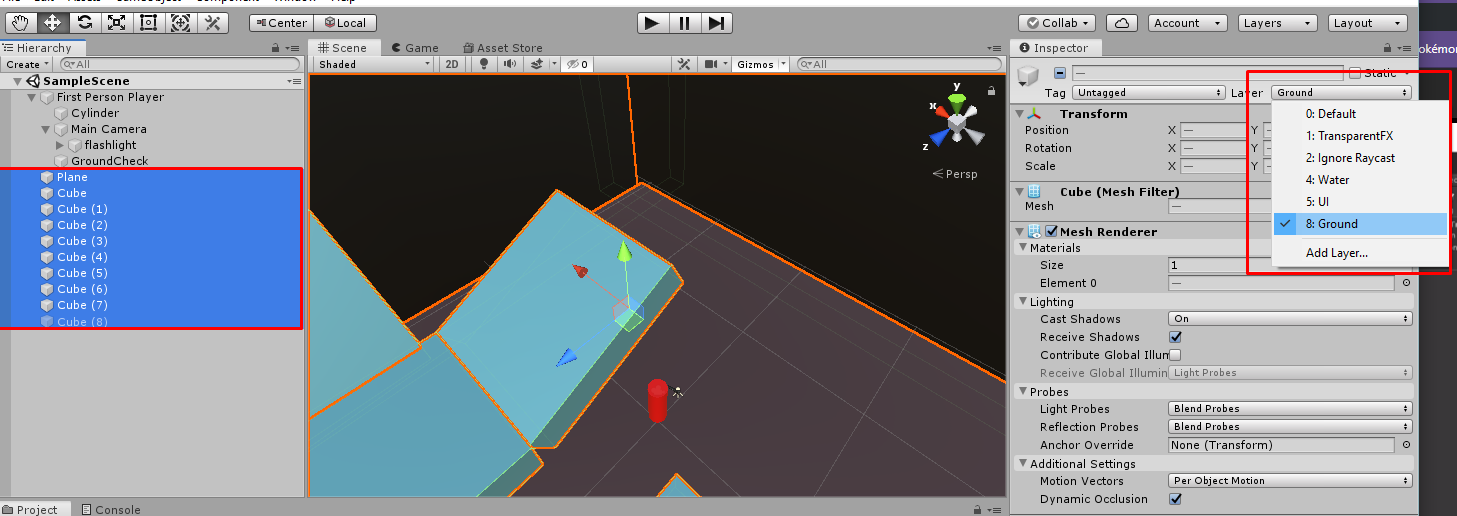
Create a GameObject in the First Person Player and move it to the very bottom of the cylinder and rename it to “GroundCheck”.

* To make this work, we start by creating a public transform in the “PlayerMovement” script and calling it “groundCheck”, as well as creating a public float deciding the distance it has to be to the ground, for the velocity to reset. Call it “groundDistance.

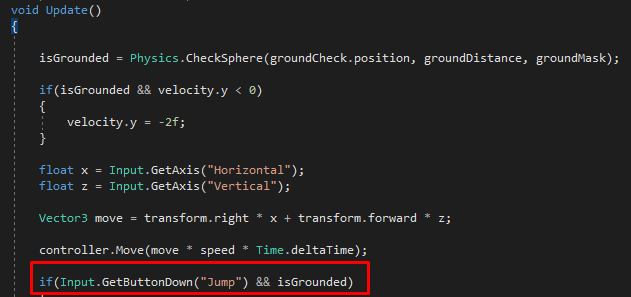
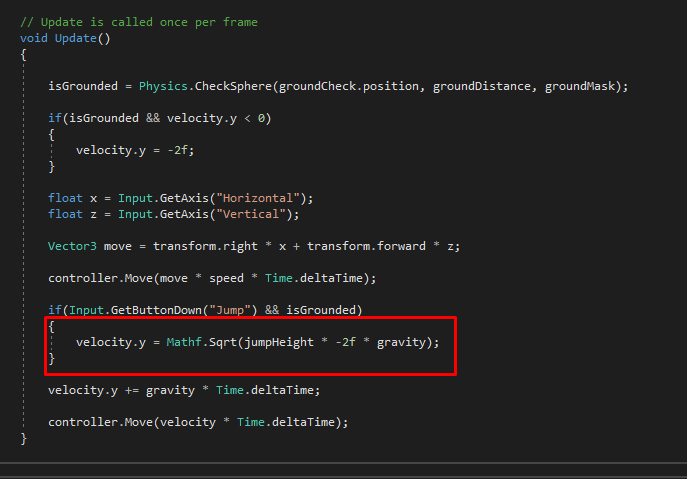
At last, create a public LayerMask and call it “groundMask” so we can control what objects the sphere should check for.

And to check if the player is grounded or not, we create a simple bool that checks that for us.



* Now in the update function we want to check if we are grounded with a physics check, as well as make it reset the velocity if the statement is true.
* Make sure to drag the GroundCheck component into the PlayerMovement script in Unity.
* Create a new Layer by clicking the layer drop-down box on the top right, and name it “Ground”, and then select that layer in the Ground Mask section of the script, so the script know what tagged objects to look for, to reset the velocity upon collision.
* Make sure all the objects in your environment are tagged with the same Layer as stated in the script, to make sure all the objects the player can land on, reset the velocity. You can do that by highlight all the objects that are from your environment in your Hierarchy and tag all of the as Layer “Ground” at the same time.

**Jumping**

* Since gravity and a ground check is implemented, making the character jump is really easy, by simply checking if the player is currently grounded and when the player presses the “jump” button, creating a realistic jump with velocity, using physics.
* Start by going back into your “PlayerMovement” script and adding an if-statement in the void update section that records if the player has pressed the jump button, as well as if the player is currently grounded, to make sure the player cannot jump again mid-air.
* Now create a public float which decides your jump hight and then add a bit of code in the update script, that tracks your jump height and uses maths and velocity to create a realistic jump.