**Movement**

**Objective:** In this lesson we’re going to write code to move the Player Ship to the left/right

If we click on Player Rig we see its coordinates are the GLOBAL coordinates in the game. If you click on the Player Ship you’ll see that these are the LOCAL coordinates. Its within the system OF the Player Rig.

In order to move the player we need to add 2 things

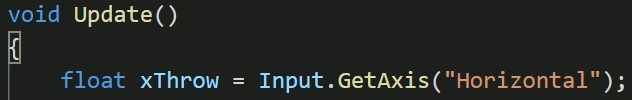
* + Where the player is currently located

+

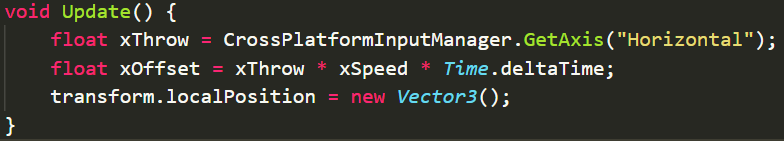
* + How far we’d like to move this frame

After calculating new position we move the Player Ship to said position

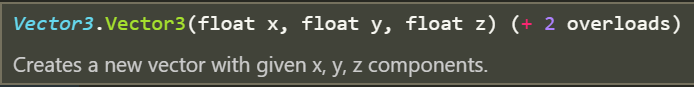
1. Open **PlayerController.cs**
   * The only thing in this file should be **Update** method with **xThrow**



1. After xThrow lets type



Now Vector3 takes 3 parameters.



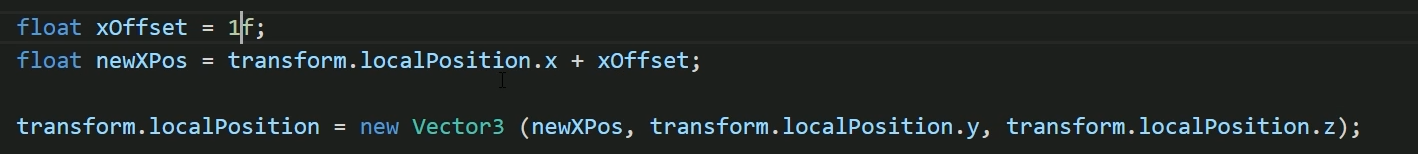
1. Inside of vector 3 lets maintain our current position



If we went to Unity to test our changes our Ship should not move.

So lets replace transform.localPosition.x with something to make it move to the right.

1. After the **xThrow** and before the **transform.localPosition variable**
   * Create a new float variable called **xOffset** and set it to **.1f**
   * Create new float variable called **newXPos** and have this value be the ships current position plus the change (xOffset)
   * Finally replace the x parameter in Vector3 with **newXPos**



1. If you go to Unity and press Play the ship should gradually move to the right.

­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Now we don’t always want the ship to move to the right. So lets calculate the xOffset based on the xThrow the user inputs and remember to account for frame rate.



**Reminder:** Time.deltaTime is the elapsed time since the previous frame.

1. Lets go to Unity and test this out.

You should have noticed that the ship moves very slowly.

1. Lets make a **[SerializeField]** float variable called **controlSpeed**



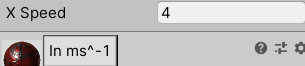
1. So lets recalculate the **xOffset** and increase it by a factor **controlSpeed**.



1. There's a new attribute here for you in square brackets called **Tooltip**. And what we do is, we give it some brackets, and then we say, in meters per second.



1. Lets go back to **Unity**. Click on the **Spaceship** and in the **Inspector** window notice when you hover over **controlSpeed** you get a hint at the units.



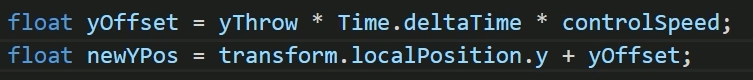
**Control Speed**

1. Lets go to Unity and test our changes. Move the ship left and right. Adjust **Control Speed** field accordingly if you feel the ship should move faster/slower.
2. Apply overrides to the Player Ship prefab

**Challenge:** Apply what you’ve learned to move your Player Ship vertically

Solution:

1. Copy and pasted xOffset and newXpos lines to rework for Y movement



1. Change the transform.localPosition to include this newYpos in the Vector3 function

