Player Movement tutorial:

This tutorial is for adding basic movement mechanic to a player in a 2D game using unity.

**Step 1: New scene**

* Start by creating a new scene, you can call it whatever, but for now we’ll call it “SimpleGameMechanicDemo”.
* Click the + icon under the hierarchy tab then select the following: 2D objects>sprites>square
* Do this twice, making sure to make one of them long into a platform by scaling its X scale in the inspector on the right-hand side.
* Make the original square, the player, a different color by clicking on the 3 dots next to the sprite render in the inspector.

**Step 2: Player and ground setup**

Rename the ground and the player objects respectively by right clicking on them and selecting rename in the hierarchy.

Next, add a BoxCollider2D to the player and the ground by clicking on it, then add component in the inspector, search “box” and boxcollider2d will show up, do the same but search “rigid” and add a rigidbody2d but only to the player.

Now that the components are added, in the rigid body settings, make sure that there is a new physics material 2d is applied, do this by right clicking in the assets area then going to create>2d> physics 2d.

Once this is done, drag it into the field called material in the players rigidbody 2d, doing this enables the player to have gravity, friction and drag.

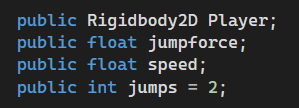
Finally, tick the Z box for freeze rotations under the constraints sub-tab to ensure the player does not rotate when not on the ground.

**Step 3: Preparing movement variables withing the player script**

Add a component for the player and type “Movement”, then new script and then create and add.

Double click on the script in the assets area and it will open visual studios.

Above void start types the following:



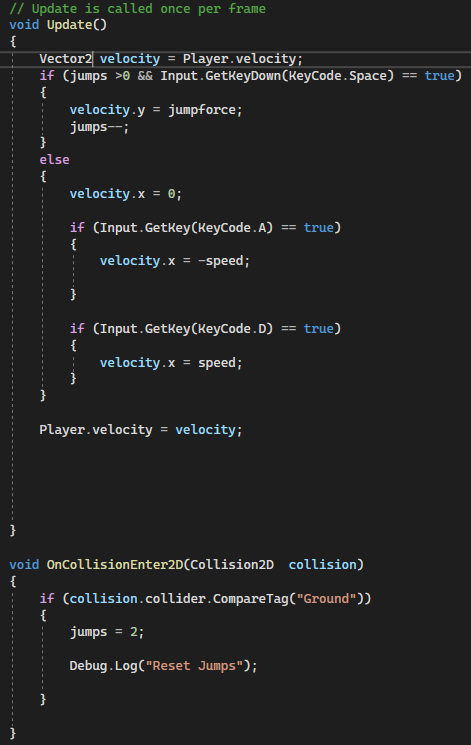
Public means that that variable will be able to be accessed directly within unity which is useful for tweaking things such as the speed of the player for example, if it were private, you would have to open and change the script every time to change something. The first line gives a refence as to what the script is affecting, in this case it’s the 2d body of the player, the second line is how much force is in each jump, the speed is how fast the player can move, and the jumps is the base amount of jumps the player has.

Using floats means that the adjustment can be finer as float variables can go into decimal numbers, but with integer “int” it can only be whole numbers, you can't exactly have 2.5 jumps.

**Step 4: Scripting the movement**

Inside the void Update we must add all our code as it is called once every frame, so every frame can be an opportunity to move.

Insert the following code:



The “Vector2 velocity = Player.Velocity;” creates a vector2 variable called velocity and then gives it to the player object, its vector2 as is a 2d game and therefore the game objects can only move on the X and Y axis.

“if (jumps >0 && Input.GetKeyDown(KeyCode.Space) == true)” checks for 2 things, if the jumps are greater than 0 and if the “keycode.space” (jump button) is being pressed at that moment.

Withing this condition is code for if the previous conditions are met (true). If it is, it will execute “velocity = jumpforce;” causing the y component of the velocity vector to have the value of the “jumpforce” giving the player vertical velocity, enabling a jump.

“jumps--;” takes away a jump, which shows up in unity’s inspector during play.

That covers jumping, but to cover movement, the code in the “else” code is executed.

Essentially, the if statements are the same except with opposing keys, being A and D, and opposing velocities, to move the player in the 2 directions, left and right. It states that if the A or D key is down, the “speed” wich is just the velocity of the player along the X axis, will change in the respctive direction.

The “Player.velocity = velocity” applies the modified velocity to the player object which holds a Vector2 value.

Now, saving the code and going back into unity and pressing play, we should see that using A,D and space, we can jump and move at the same time, while also being able to adjust the speed of the movement, as well as the force of the jumps.