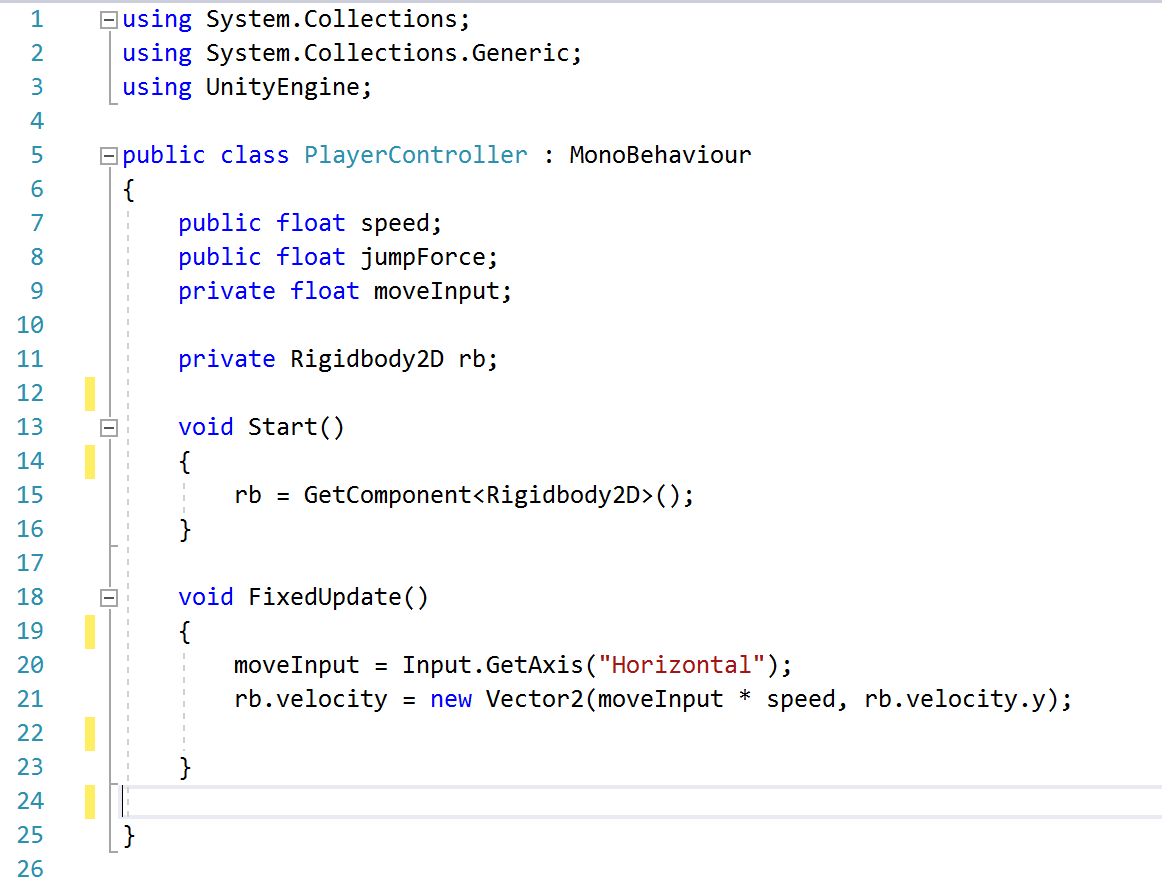
Unity Tutorial: 2D Movement and Jumping

1. Create a Scene

Start by adding 2D object sprites named platforms and place three or more around the scene. Also, add another 2D object sprite named player. Create an empty object and labelled it Ground Check and a new tag called Ground and tag it on both the player and the platforms.

1. Create a new C# script

Next, create a C# script called PlayerController and open it on Visual Studios. First, it needs a public speed variable that will dictate how fast the character will move in the scene. Also, it needs a public float variable called jump force to alter the jump height. Then create a private float variable named moveInput which detect if the player has both left and right keys pressed. And finally, add a Rigidbody2D (rb) so the player is set up for the players when collided.

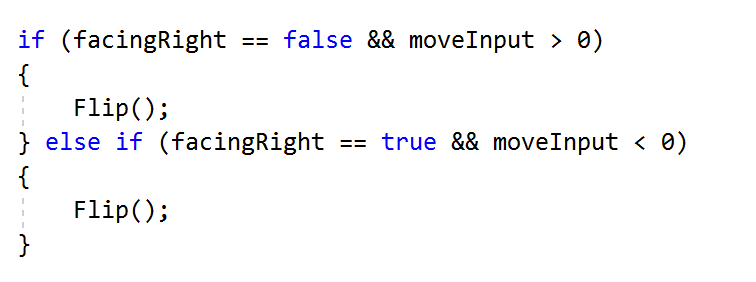
Then write up the rb with GetComponent with Rigidbody2D in void Start. After that write the moveInput in the FixedUpdate in order to get the variable to activate with the velocity of the player’s sprite in the scene with the speed and through its axis. The code below is how it should be formatted.

1. Making the player move

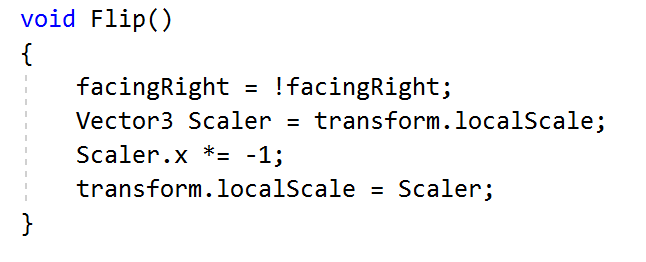
After the code is written, go to Unity and add the PlayerMovement script to the Player object and add some value for speed and make sure that the player object has Rigidbody2D on it and freeze the rotation Z. Add box colliders on the platforms so the player doesn’t fall through it and now it should be on the platform without falling through it.

1. Flipping the player on left side

In order to make the player flip on the left side without adding too many sprites, go back to the script and private bool facingRight = true; in above then add the code below in the FixedUpdate()

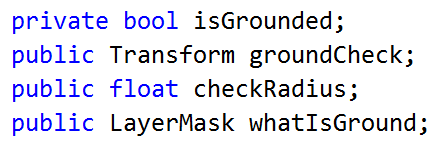


After that make an add a separate void Flip() and add the code below. Once it’s working the player should turn natural in both sides.



1. Adding Jumping Mechanic

Now the player needs a jump mechanic to navigate in the scene, add the code below into the script above so it can help make the variables function. Then in the FixedUpdate() add the overlap circle code.

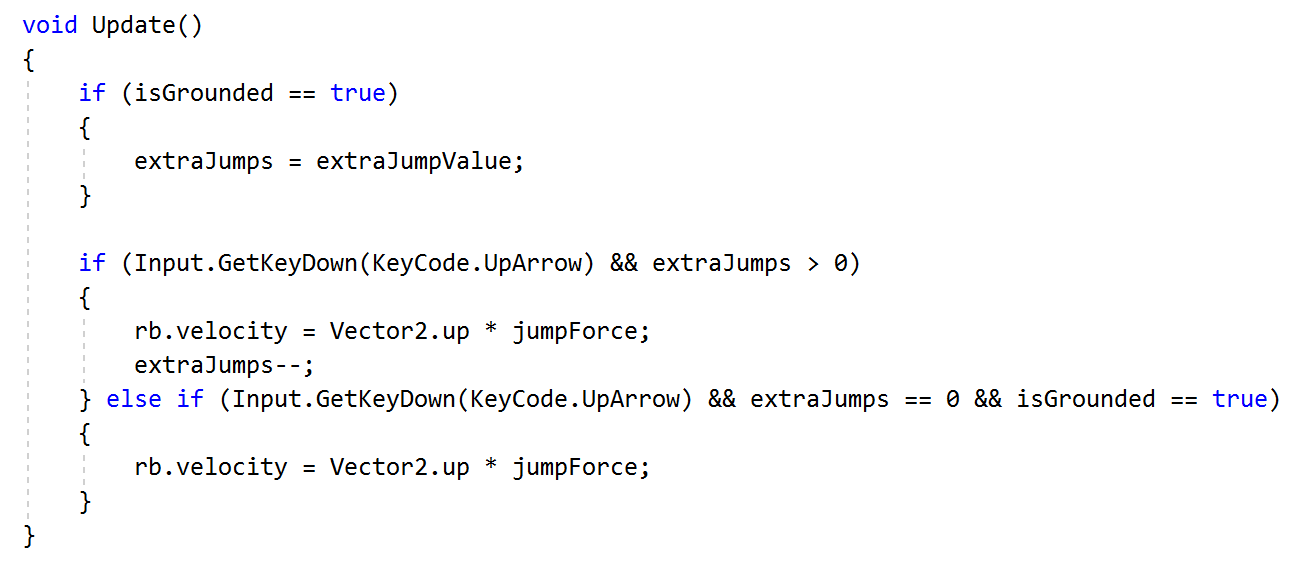




After that, add an empty game object called Ground Check and place it on the player’s feet. Then drag it to the ground check section in the player script in Unity, check if it has the ‘Ground’ tag and adds a check radius decimal.

1. Adding extra jumps

Go back to the script and add [public int extraJumps;] and [public int extraJumps;] above the page. Then in the Void Start() add [extraJumps = extraJumpsValue;] and add the code below into the Void Update() section.



Finally, add the values in jump force and it should be able to jump in the scene. Also to prevent the player from getting stuck on edges on the platform, create a Physics Material 2D and labelled it PlayerMat and set friction to 0. Now it should be able to move and jump in a 2D setting.