**Double and Triple Jump Tutorial**

**Step 1: Level Layout**

Add a white square sprite (you can make it or import one online)

After that make 4 platforms which will be rectangular to simulate the jumping area.

Lastly create one square with a darker colour, which will be our PC

**Step 2: Player Controller**

Create a new C# Script and call it PlayerController.

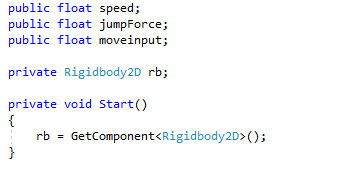
Delete the start and update function and then type a public float function called “speed”, this will dictate how fast the player will move.

Afterwards add another public float and call it “jumpForce” this one will dictate how high your character jumps.

Then add a private float called “moveInput” which will dictate if the player is pressing down the keys to move.

Lastly add another private variable of type rigidbody2D and call it rb, this will define the term rigidbody.

And finally create a void start option and inside it call the rigidbody variable by setting rb to be equal to get component rigidbody; this will allow us to control the rigidbody of our character.

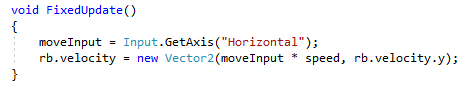


**Step 3: Moving the character left and right**

Create a new void FixedUpdate function. This function is used to manage all physics related aspects of your game.

Inside the FixedUpdate function add the moveInput we created previously and set it equal to Input.GetAxis (“Horizontal”) what this does is it calls your horizontal axis allowing your player to move.

Now to get the character actually moving you require speed so below the previous line add rb.velocity and set it equal to new vector2. Set the X axis to moveInput times speed and since we don’t want the y Axis to be affected we will set it as “rb.velocity.y”.



**Step 4: Unity Movement**

Return to Unity and add the “PlayerController” to the player, as well as adding a 2D boxcollider and a rigidbody2D, make sure both of them are 2D.

In addition to that, add 2D Boxcollider to the platforms as well so the player does not fall through them.

**Step 5: Detecting Ground**

To start we have to create several variables.

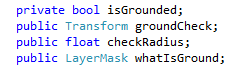
First we create a private bool called “isGrounded” which will detect if the player is touching the ground.

To check if the player is touching the ground we need to make a small circle via script that will indicate if we are touching the ground or not.

We create a public transform function called “groundCheck”

We also create a float called “checkRadius”

And lastly we create a LayerMask called whatIsGround.



Now inside the FixedUpdate function create a new variable. Set isGrounded equal to Physics2D.OverlapCircle, afterwards inside parenthesis state a position for the circle so generate it by the player’s feet so at the “groundCheck” position.

Afterwards separate with a coma and state a radius for the circle so checkRadius and lastly add the layer mask with whatIsGround.



Afterwards we go back to Unity and create an Empty Game Object and call it “Ground Check” we place it on the player’s feet, afterwards select the player again and drag and drop the ground check in the ground check slot on the script.

Also remember to type in a value for check Radius (like 0.5).

Furthermore create a new layer on the top right and name it Ground, after creating this layer specify the layer on the player character in the whatIsGround area and also set every platform as a ground layer.

**Step 6: Jumping**

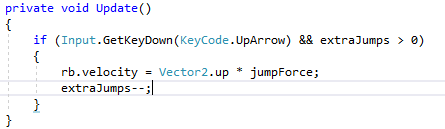
To make the player start jumping create a Public Int variable called “extraJumps”.

Afterwards create an update function and create an if statement that will check if the player hits the up arrow key.

But the function will only work if the up arrow is pressed and if extra jumps is greater than 0.

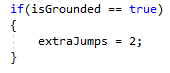
If both those conditions are met then we set rb.velocity and we make it equal to Vector2.up time’s jumpForce.

Now to prevent our player from having unlimited jumps we need to set the jumps to decrease each time by typing extraJumps—



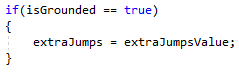
Now to make sure that the jumps get reset when the player lands on the floor we need to create a new if statement, check with the if statement if isGrounded is equal to true.

If it is then set extraJumps to be equal to any number you want, in this case it will be 2.



Now a thing you can do in order to not directly affect the extraJumps value, you can set extraJumps to a private int and create a new public Int variable called extraJumpsValue.

Then just replace the 2 with extraJumps.

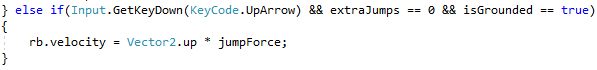


**Step 6: Limiting Jumping**

Now to limit the players ability of being able to jump you need to create an else if statement under the jump variable.

Test if the player is pressing down the jump key, has no more jumps available and also make sure that the player is touching the floor, if this is true then set the same rb.velocity line of code as before.

If this line of code is not done then the player will be able to jump infinitely.



And there you go, that’s how you make a player jump as many times as you want in unity

Extra Step: Colliding

The player will get stuck to the walls or the platforms, there is a very simple fix to this issue and that’s by adding within unity a 2D Physics material, give it the name “PlayerMat” then modify the friction to 2.

Lastly drag and drop it in the rigid body physics material and there you go!