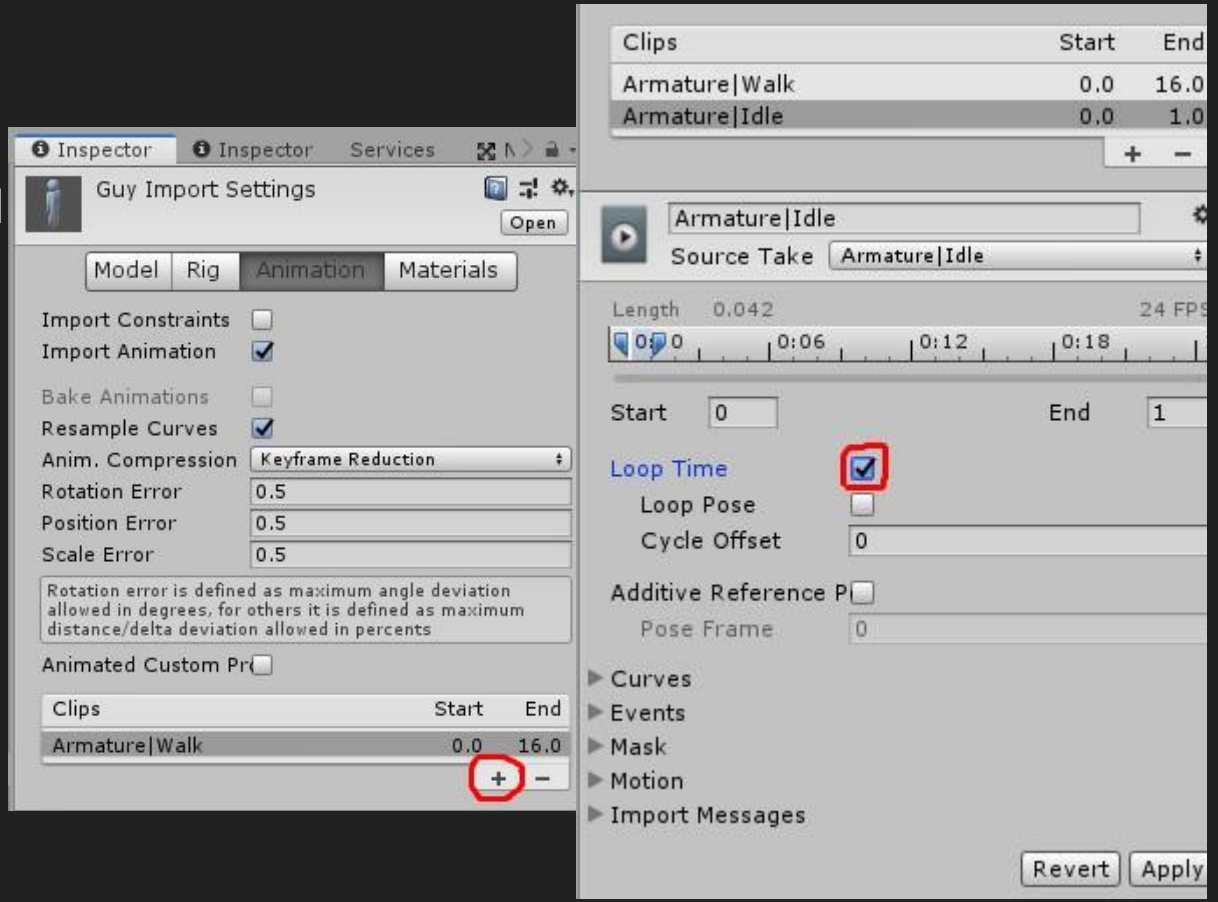
Lesson 5

Getting Unity to Import New Animations

- Find the imported blender file in your assets.

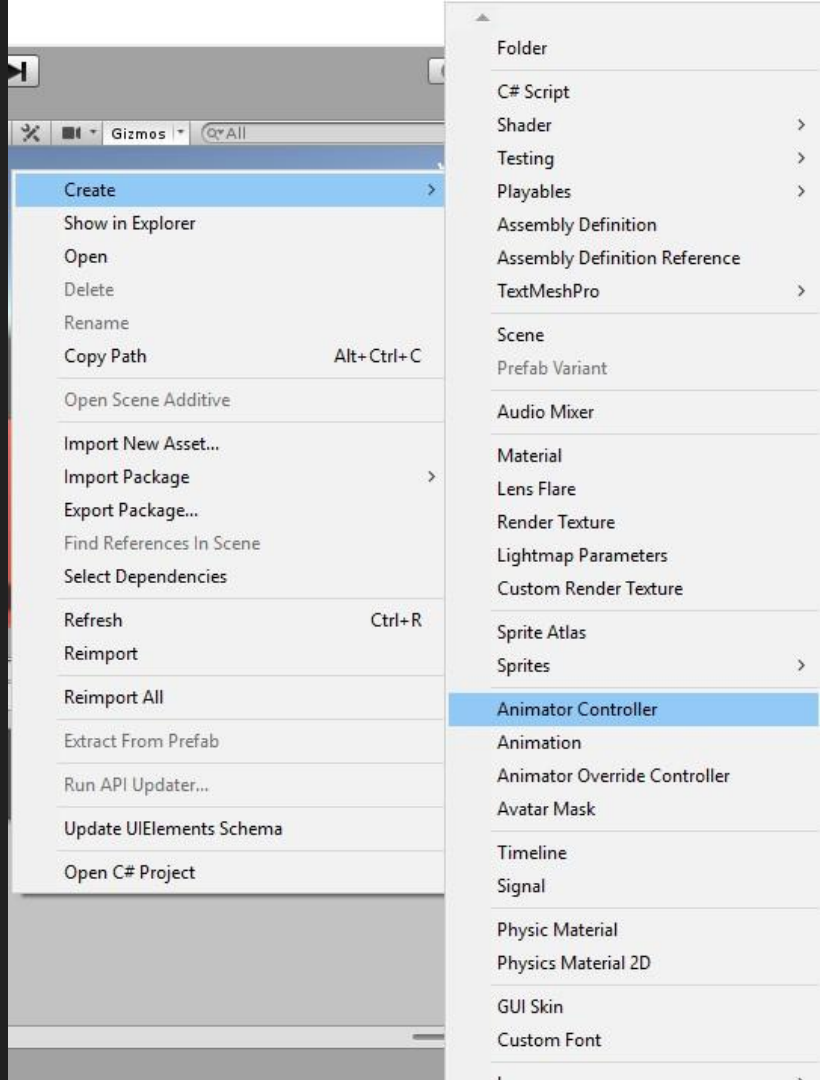
- Go the the animations tab and of the imported file and hit the plus button if an animation is missing from the list.
- Click on the new animation and make sure check the ‘Loop Time’ box.



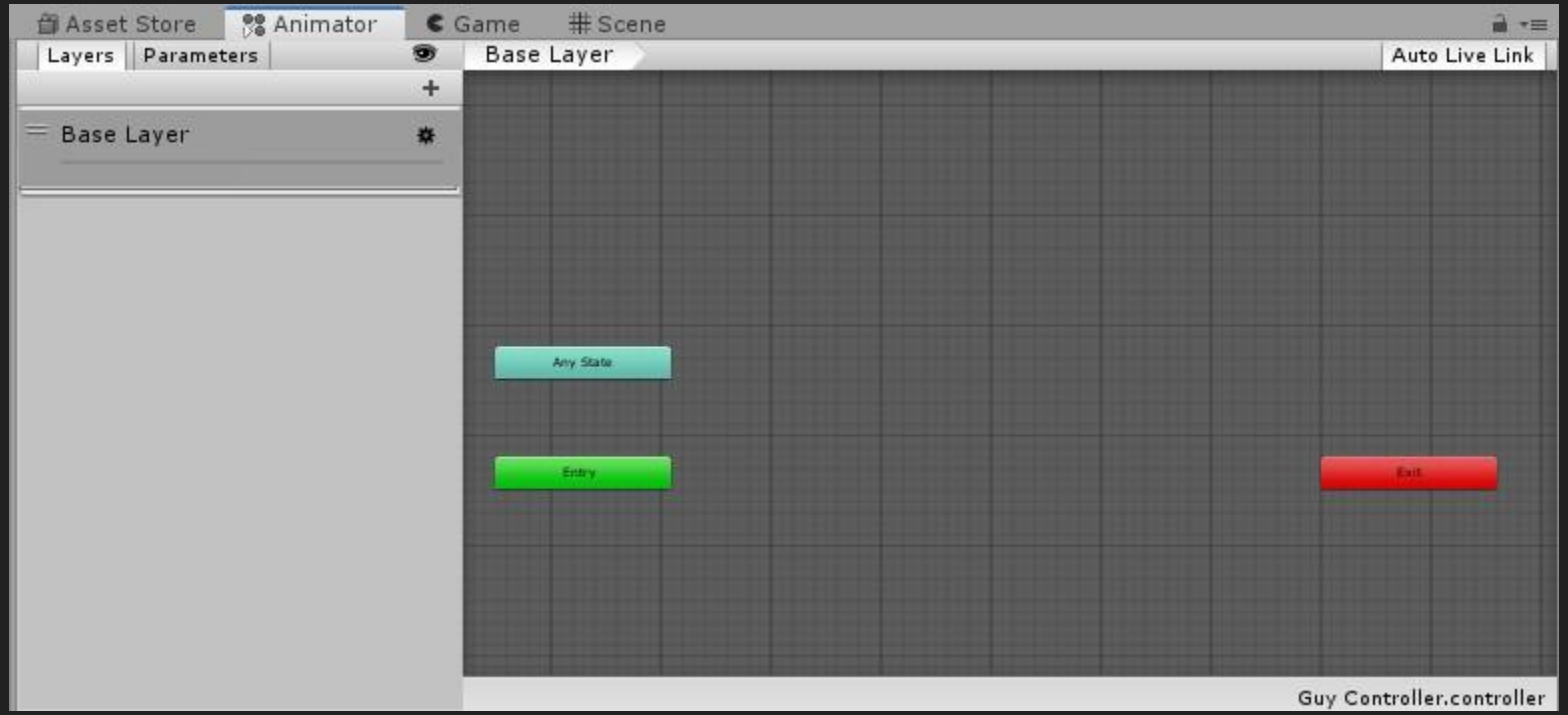
A Review of the Animator Controller

- Create an ‘Animator Controller’ asset

- Double click on the new asset or change the viewer to the ‘Animator’ tab to open the animator controller view

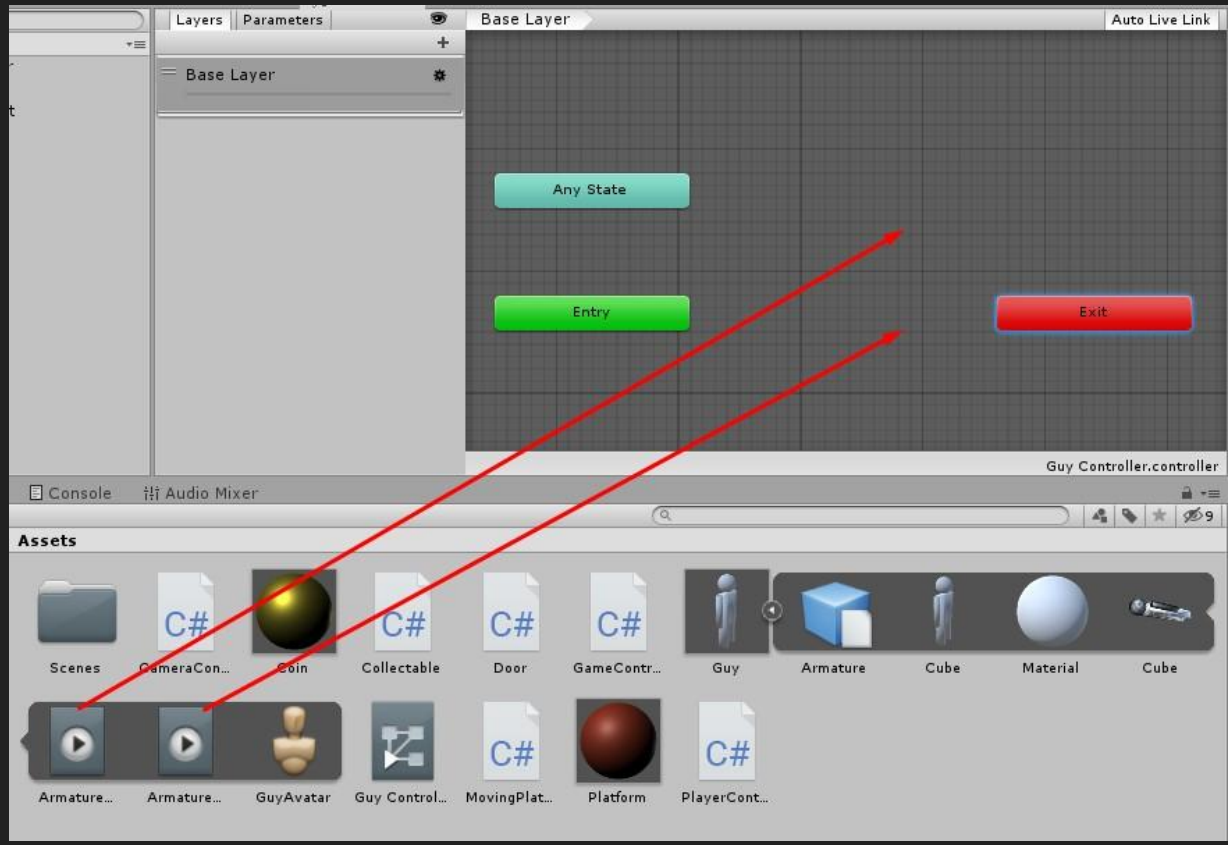


The Animator Controller View



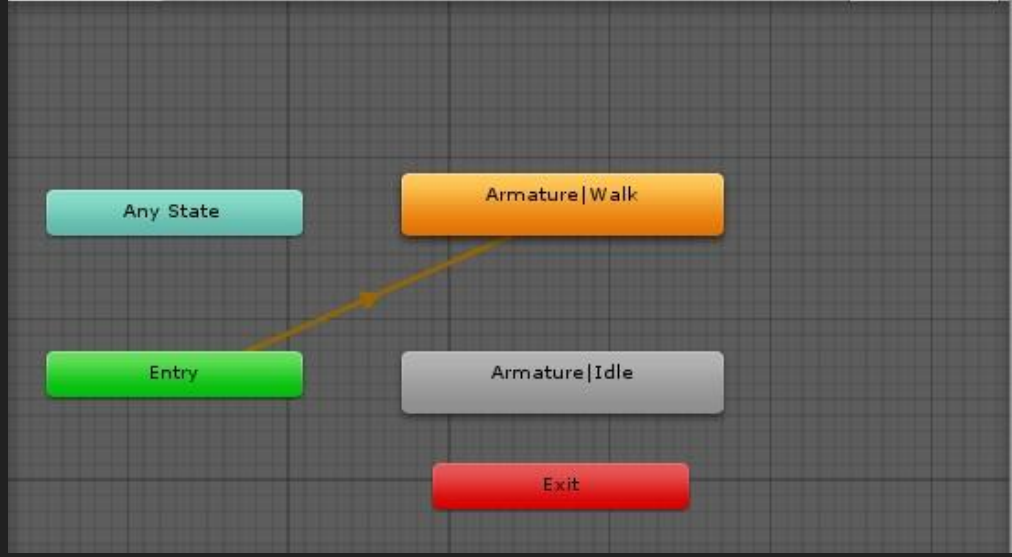
- Click the expand button on the imported character

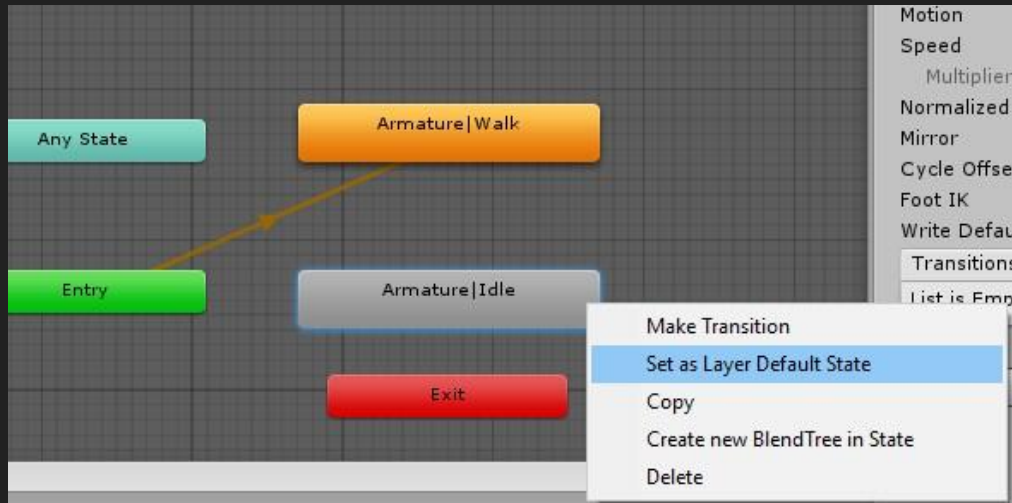
- Drag and drop the animations from the asset window into the animator window



- Depending on the order you add animations one will be set as the default animation state

- We want the Idle animation to be the default, do this by right clicking on the ‘Idle’ block and hit the ‘Set Layer Default State’ option

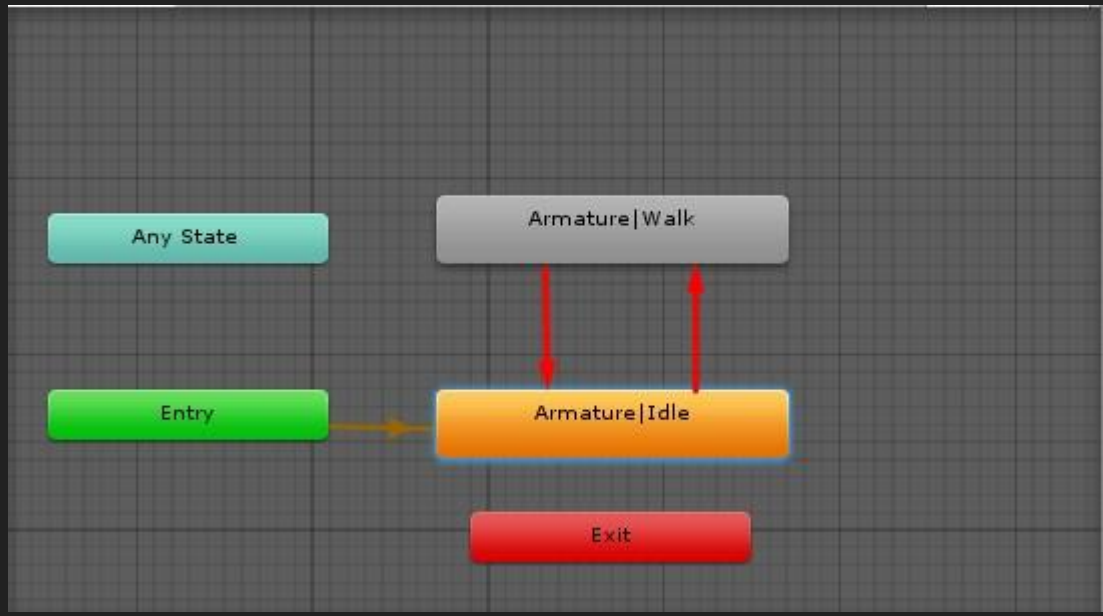




Our Animation Logic

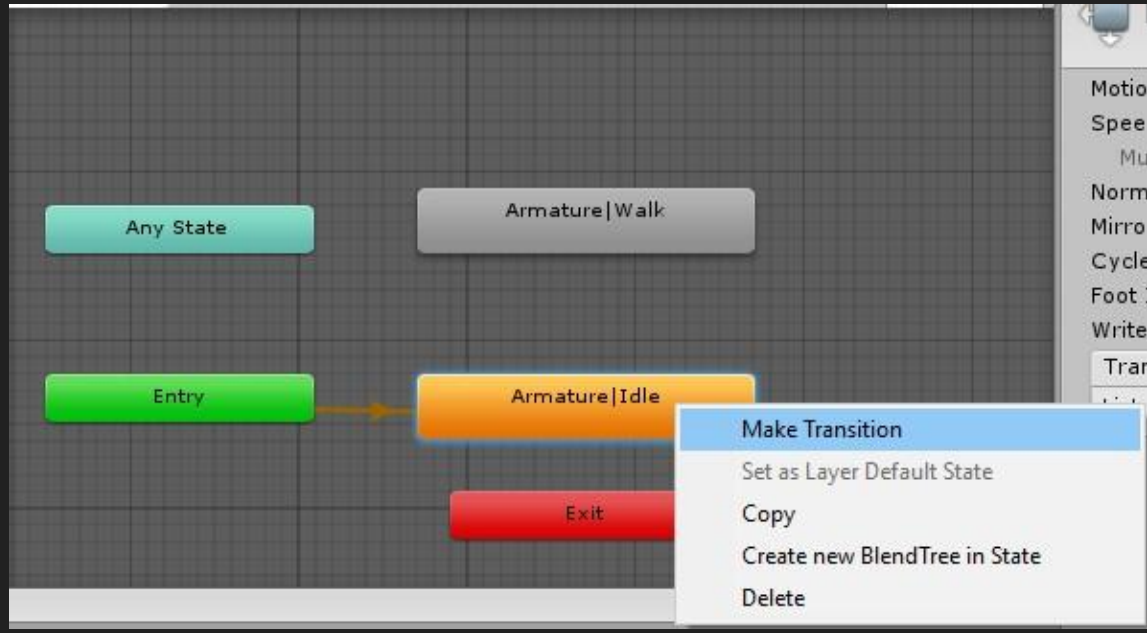
- We want our character to do the walk animation when we are moving but also stop when we are not moving

- So we need to be able to transition between walking and idling both ways



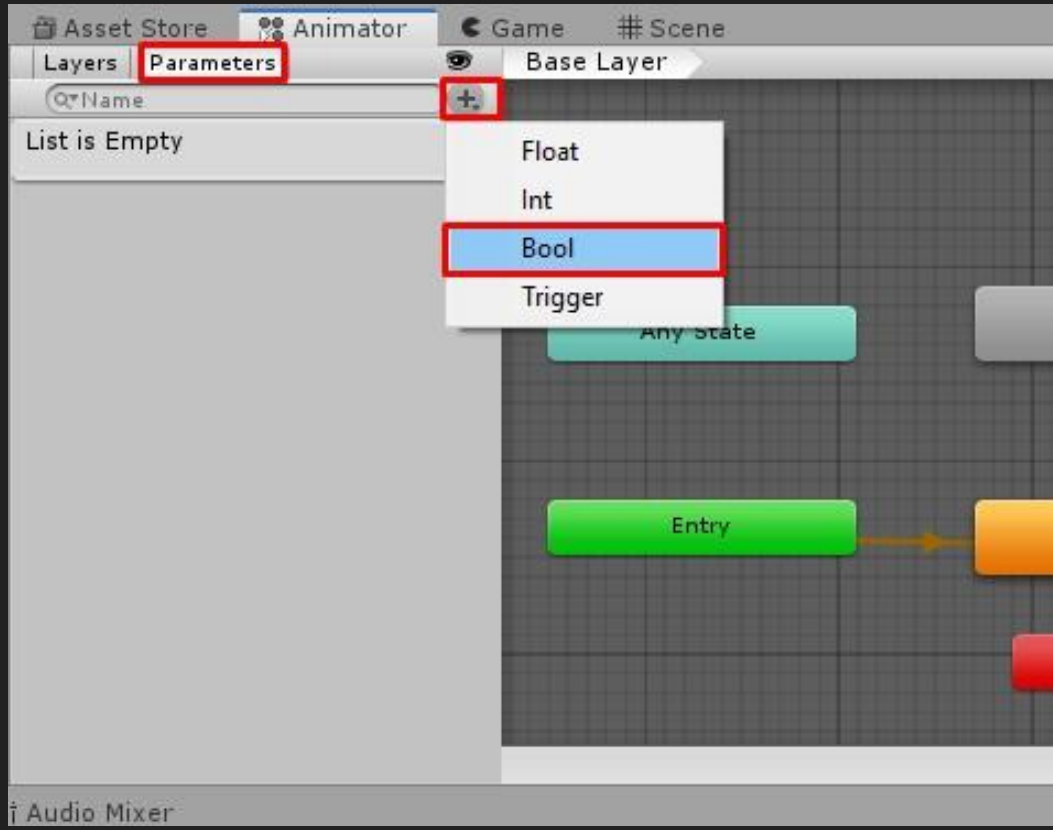
- Create a transition from ‘Idle’ to ‘Walk’

- Right click on ‘Idle’ block and hit the ‘Make Transition’ option then select the ‘Walk’ block
- Repeat to make a transition from ‘Walk’ to ‘Idle’



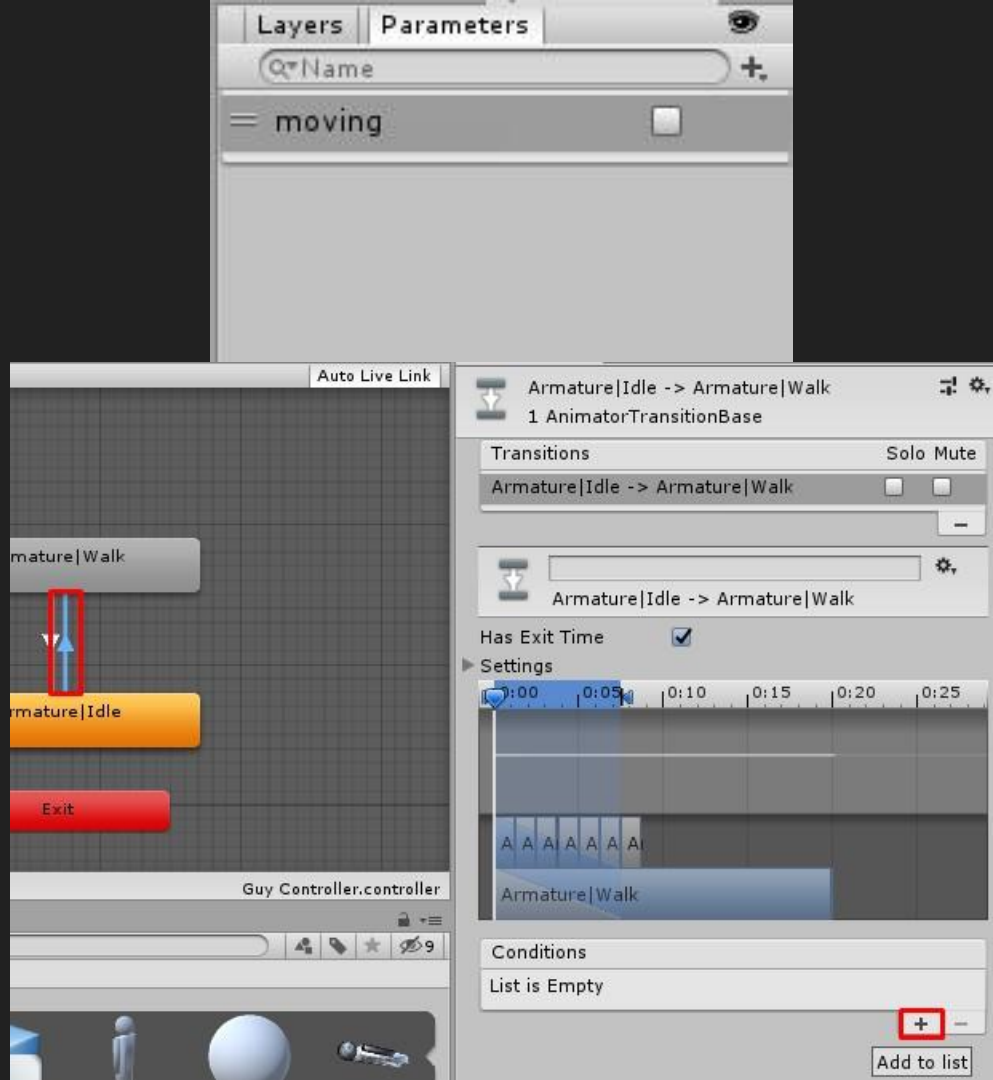
- We need a variable within the animator to control the transitions with

- Create a boolean variable in the animator window. We can then control this variable through scripts

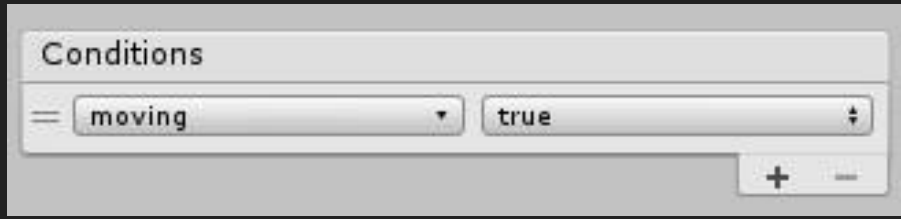


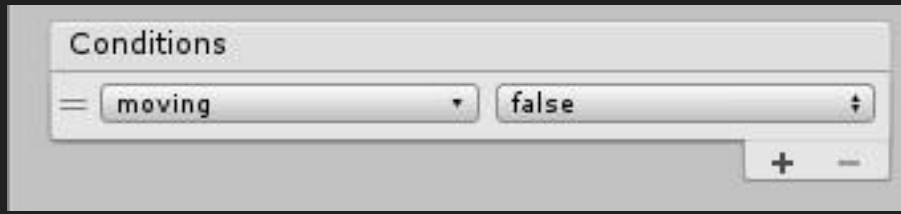
- Give this variable a good name

- Click on a transition arrow and find the ‘Conditions’ list in the inspector. Hit the plus button to create a new condition

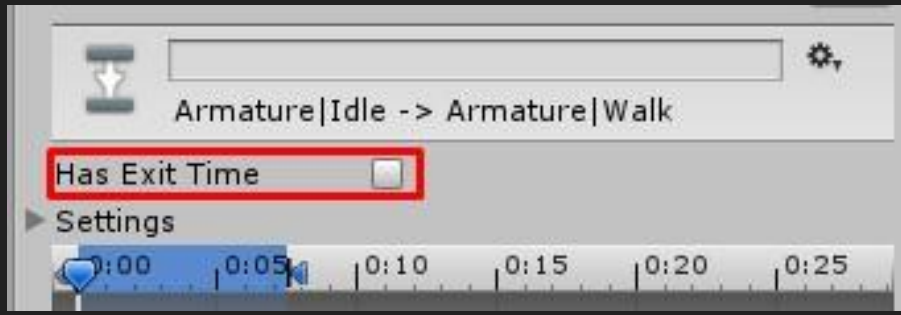


- For the ‘Idle’ --> ‘Moving’ transition set the variable to ‘TRUE’.
- For the ‘Moving’ --> ‘Idle’ transition set the variable to ‘FALSE’.





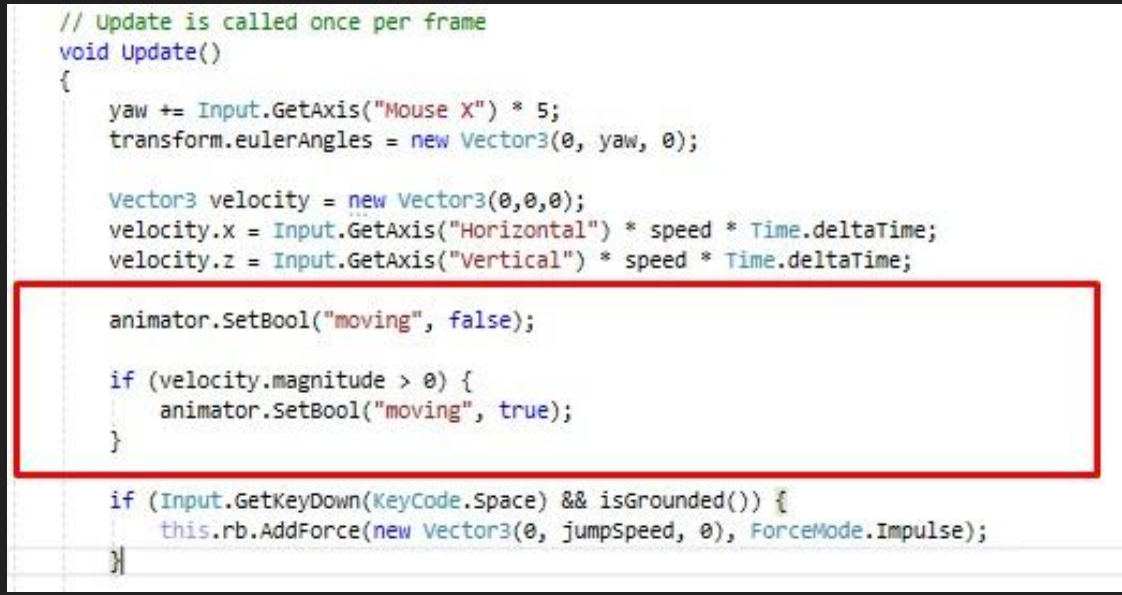
- Set the ‘Has Exit Time’ checkbox on both of the transitions to ‘FALSE’



Scripting the Animations

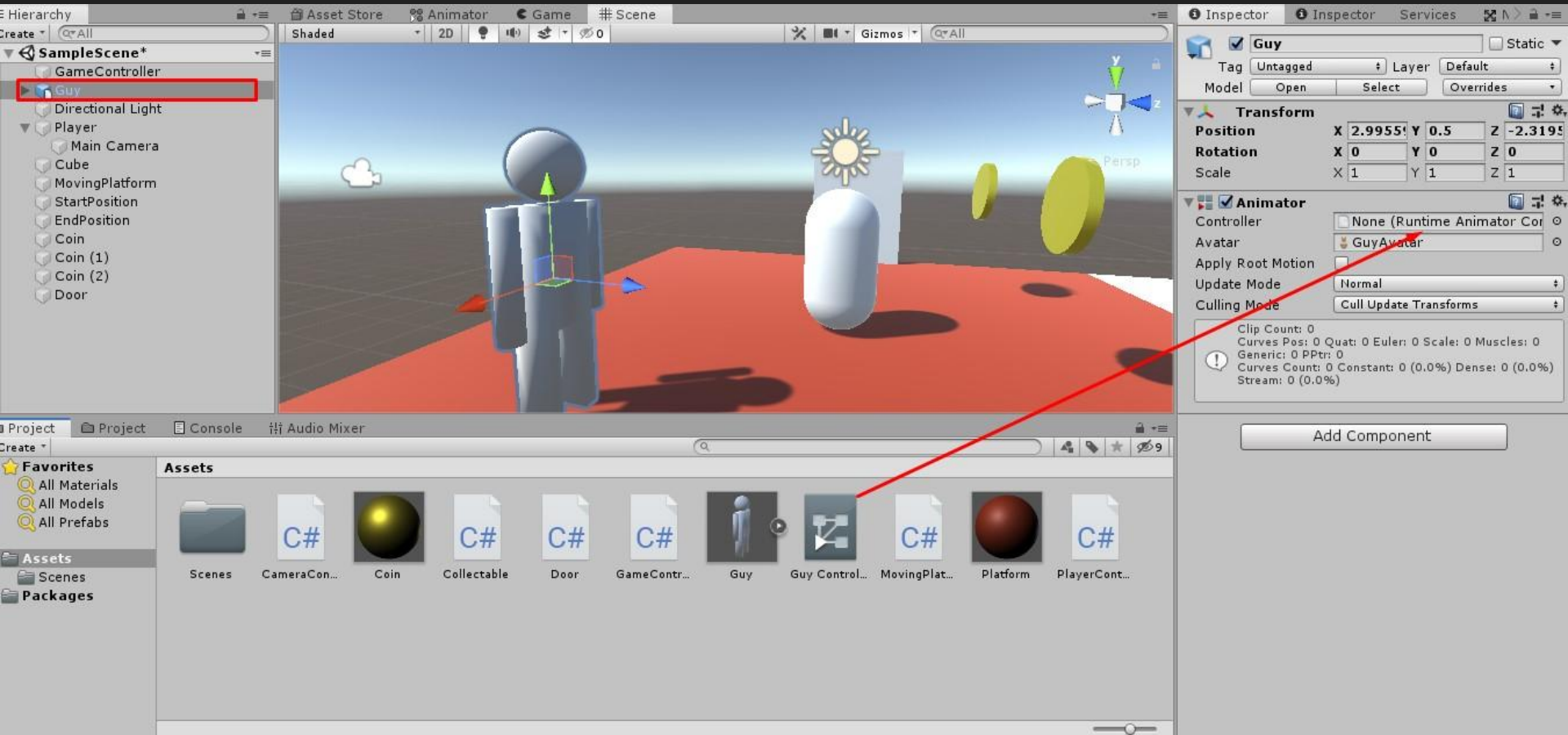
- In the ‘PlayerController’ Script add a public reference to an Animator component

- In Update() of PlayerController add logic to control the boolean variable of the animator



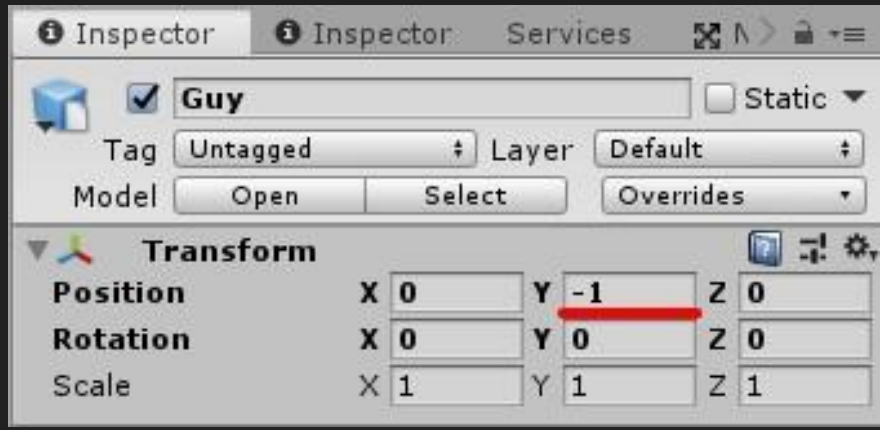
```
// Update is called once per frame
void Update()
{
    yaw += Input.GetAxis("Mouse X") * 5;
    transform.eulerAngles = new Vector3(0, yaw, 0);

    Vector3 velocity = new Vector3(0,0,0);
    velocity.x = Input.GetAxis("Horizontal") * speed * Time.deltaTime;
    velocity.z = Input.GetAxis("Vertical") * speed * Time.deltaTime;

    animator.SetBool("moving", false);

    if (velocity.magnitude > 0) {
        animator.SetBool("moving", true);
    }

    if (Input.GetKeyDown(KeyCode.Space) && isGrounded()) {
        this.rb.AddForce(new Vector3(0, jumpSpeed, 0), ForceMode.Impulse);
    }
```

- In the scene drag the animator into the controller reference for our character

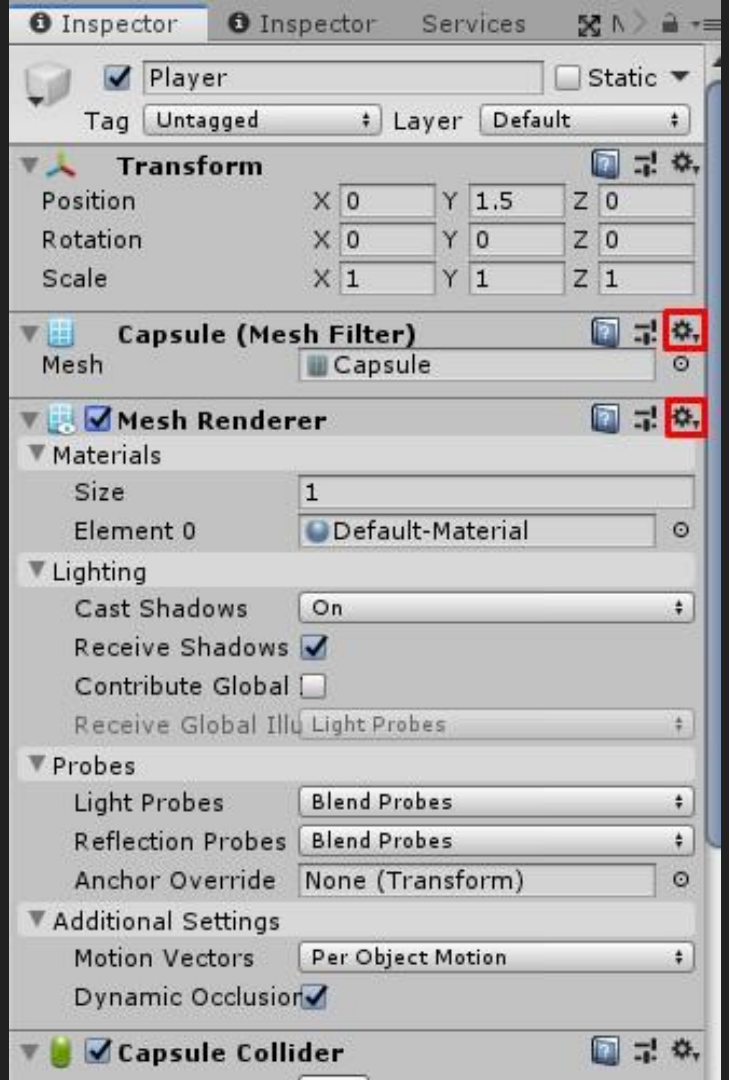


- Make the character a child of our existing Player object.

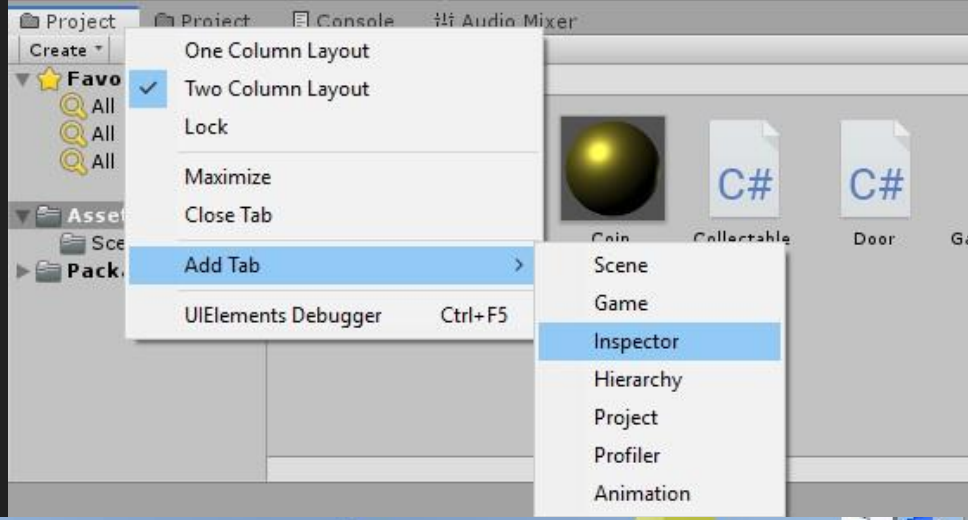
- Reset the transform of your character (now child of the player), then set the y position to -1.

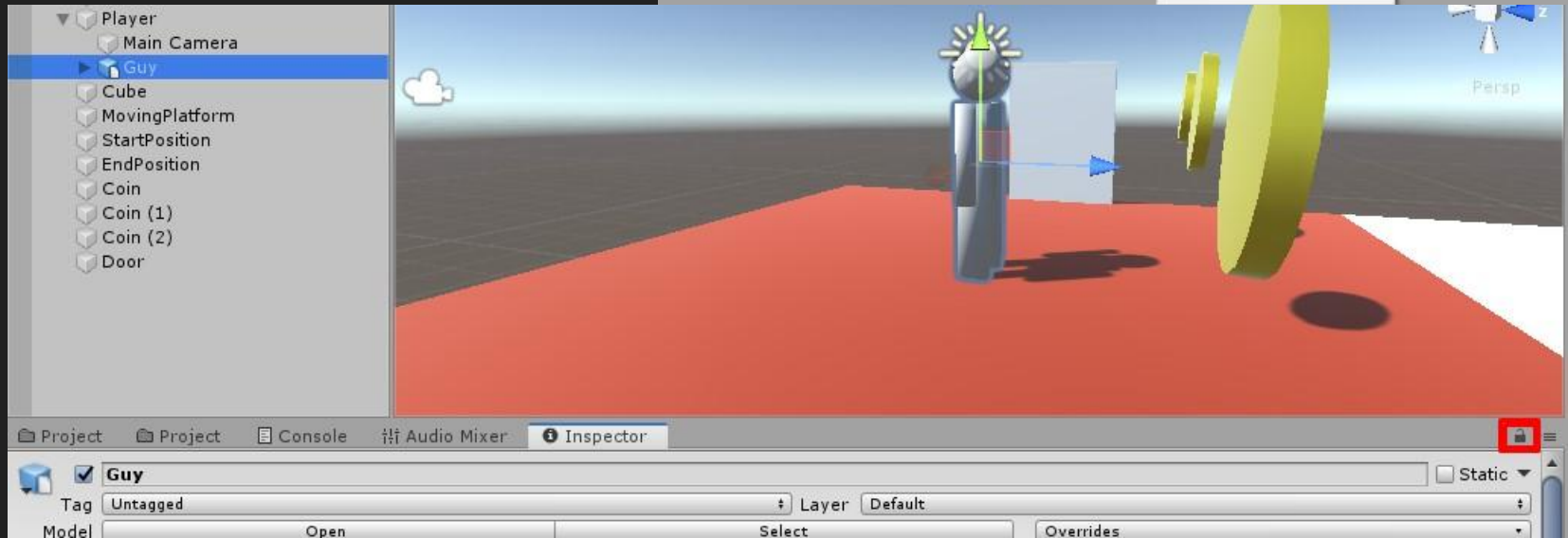


- Remove the old capsule mesh from the Player object
  - Remove Capsule (Mesh Filter)
  - Remove Mesh Renderer



- In the bottom tab add another inspector tab, then click on your character object, then lock the bottom tab.





- Click on the Player then drag the Animator component from your character to the Animator reference in the PlayerController script on the Player object.