Animation

Today’s Agenda

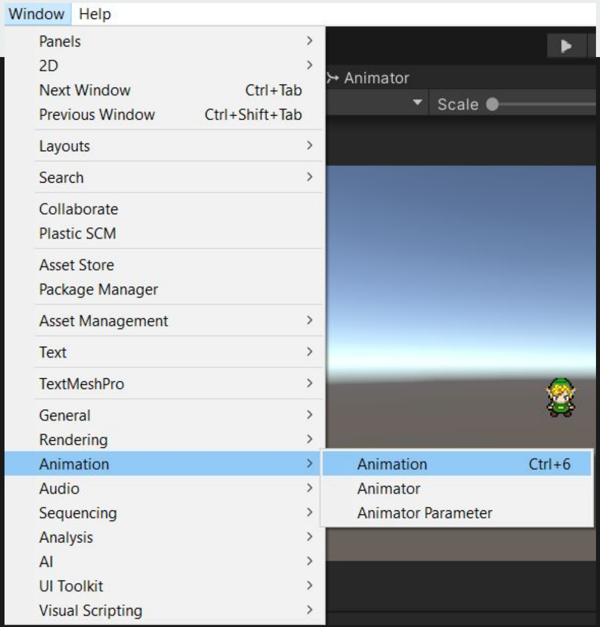
We’re going to be going to be continuing with Chapter 17: Animation, Chapter 18: Animators

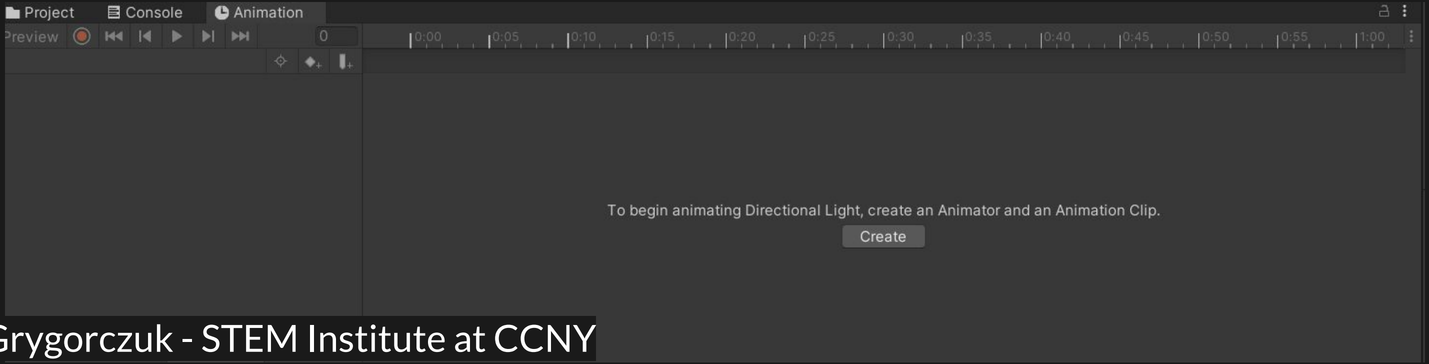
- How to animate any game object
- How to use Sprite Sheets and Rigging to animate characters
- How to use The Animator to switch between animations on the fly

Animation

To Start animating we will open up the Animation View which can be found Windows -> Animation -> Animation

I like to put this window on the same area as project and console but it’s up to you where you palace it.



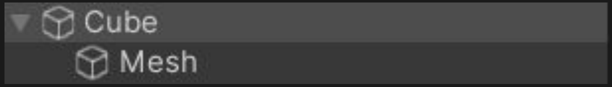


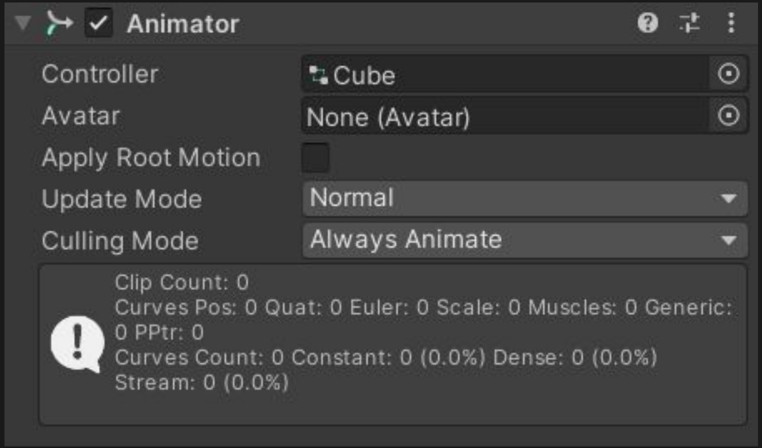
To begin animating Directional Light, create an Animator and an Animation Clip.

Animator Component

When you select a game object such as Cube you want to have a 3D Mesh or 2D model as a child so that editing the transform won't affect the control of the game object.

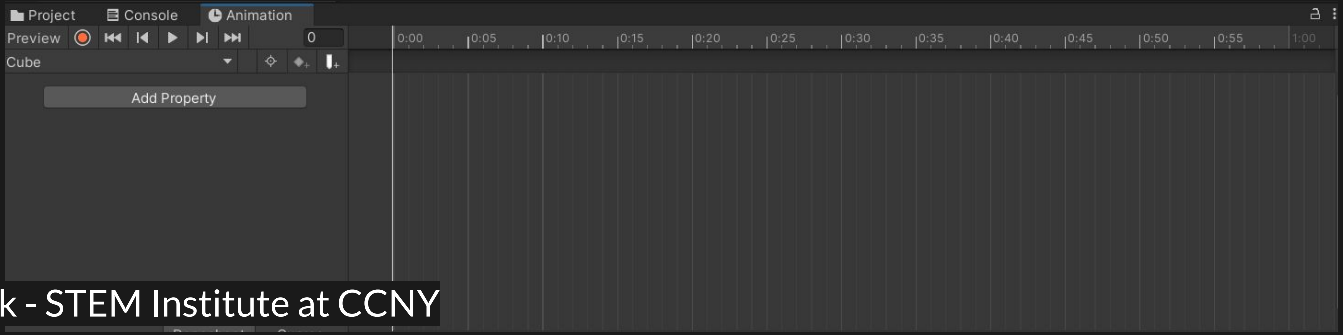
Once you've selected the parent you can click create this will create two assets. A controller and an animation clip.




Animator

Controller: Cube
Avatar: None (Avatar)
Apply Root Motion
Update Mode: Normal
Culling Mode: Always Animate

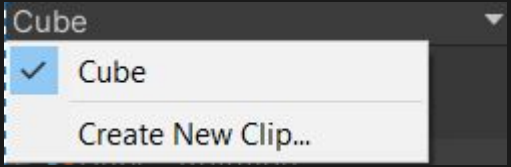
```
Clip Count: 0
Curves Pos: 0 Quat: 0 Euler: 0 Scale: 0 Muscles: 0 Generic:
0 PPtr: 0
Curves Count: 0 Constant: 0 (0.0%) Dense: 0 (0.0%)
Stream: 0 (0.0%)
```

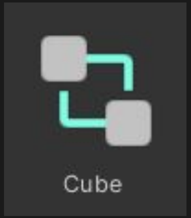


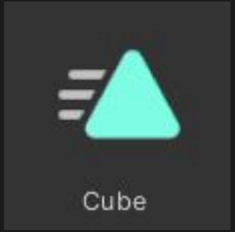
Animation Assets

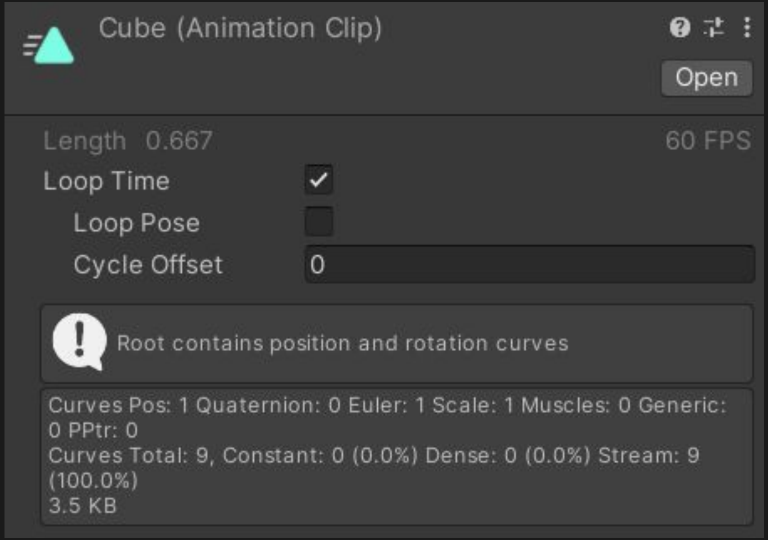
When you create your assets you will have two things, the three squares is the controller, it holds all of the animations and later we'll see how it allows us to switch between animations.

The Triangle is the animation, if you click on it you will see that you can control so of the data of it such as if it loops or if it starts few seconds into the animation.









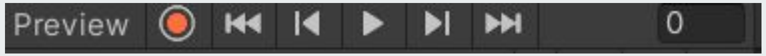
Length 0.667 60 FPS
Loop Time ✓
Loop Pose
Cycle Offset 0

Root contains position and rotation curves

Curves Pos: 1 Quaternion: 0 Euler: 1 Scale: 1 Muscles: 0 Generic: 0 PPtr: 0
Curves Total: 9, Constant: 0 (0.0%) Dense: 0 (0.0%) Stream: 9 (100.0%)
3.5 KB

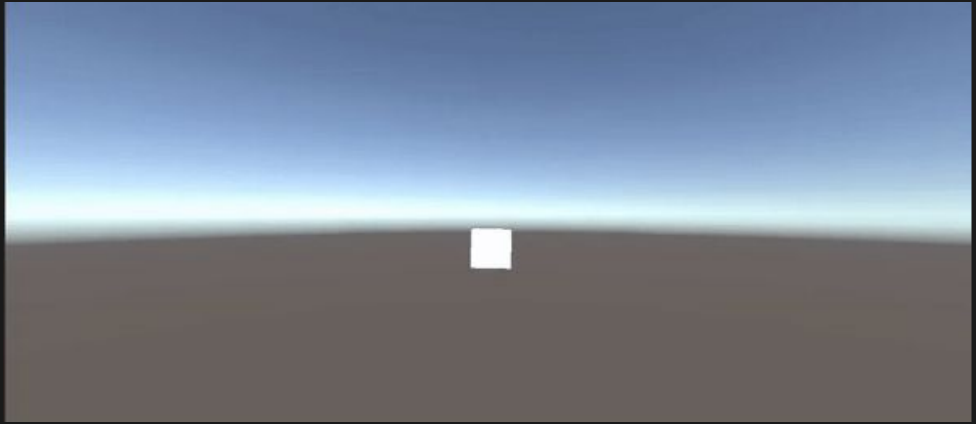
Sebastian Grygorczuk - STEM Institute at CCNY

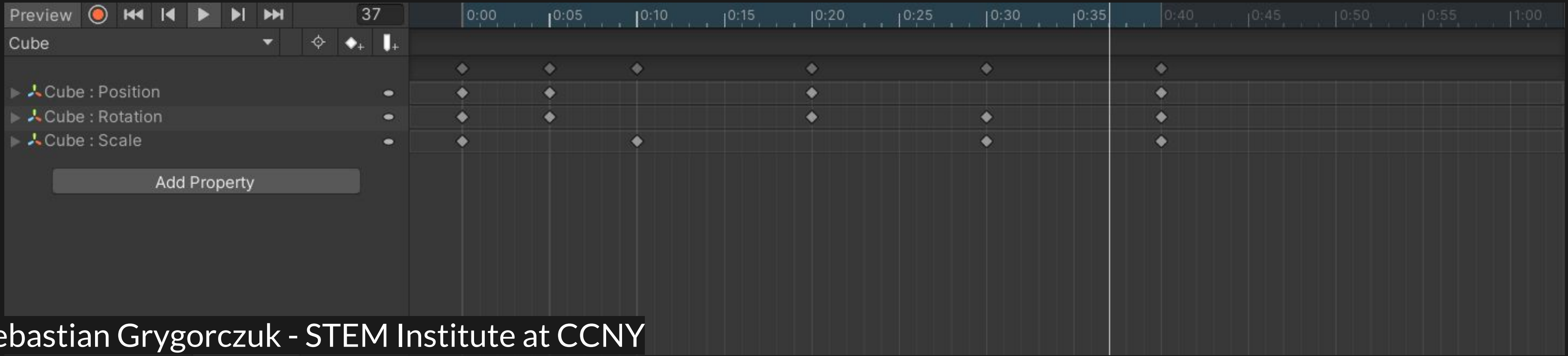
Dope Sheet



The Section below is called a Dope Sheet and it lets you make nodes at different times.

You can only modify the nodes while you are in record mode, which will make your time red. You can affect anything in here, as long as it's the game object or it's children.

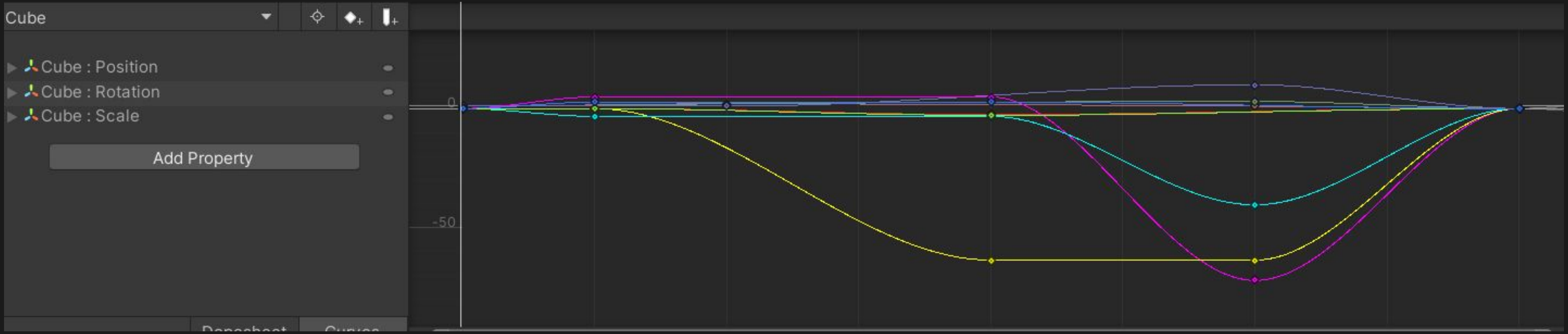




Sebastian Grygorczuk - STEM Institute at CCNY

Curves

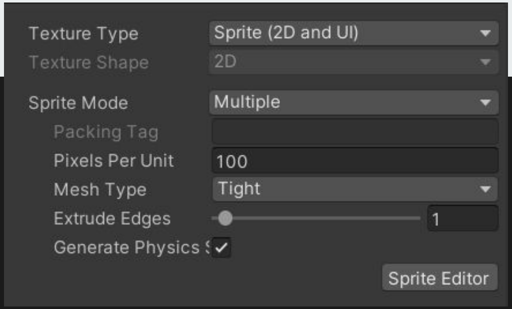
When you are controlling something’s transitions, such as transforms Unity calculator the gradual steps between where you start and where you end. The Curves Menus allows you to see and modify that transition using the curves. You can make something happened instantly, or make it very slow and speed up at the end or vice versa.



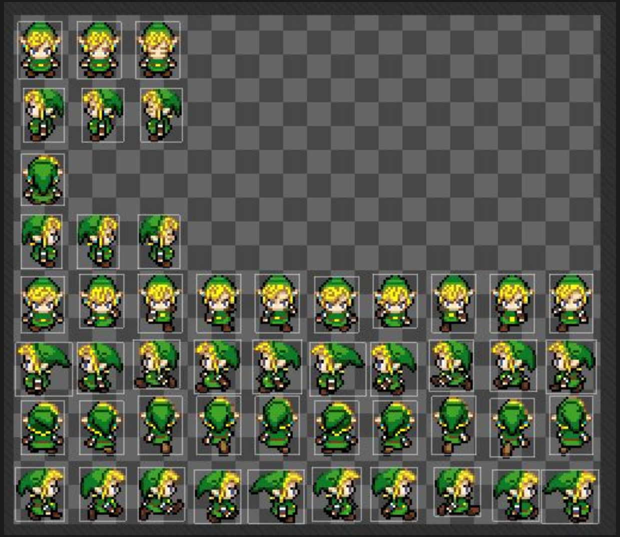
Sprite Sheet

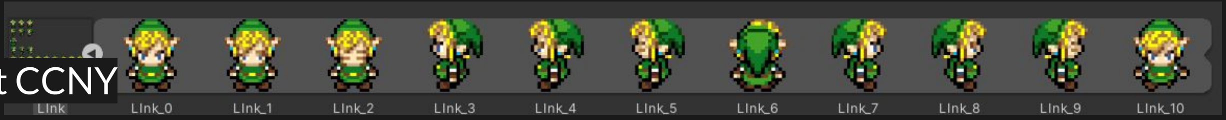
Sprite Sheets are very useful tool for animation. They hold all of the frames of one action and you can just switch between them.

Make sure your Texture Type is set to Sprite (2D and UI), Sprite Mode is set to Multiple and go into Sprite Editor to slice up the sheet automatically or by cell.



Texture Type: Sprite (2D and UI)
Texture Shape: 2D
Sprite Mode: Multiple
Packing Tag:
Pixels Per Unit: 100
Mesh Type: Tight
Extrude Edges: 1
Generate Physics S ✓



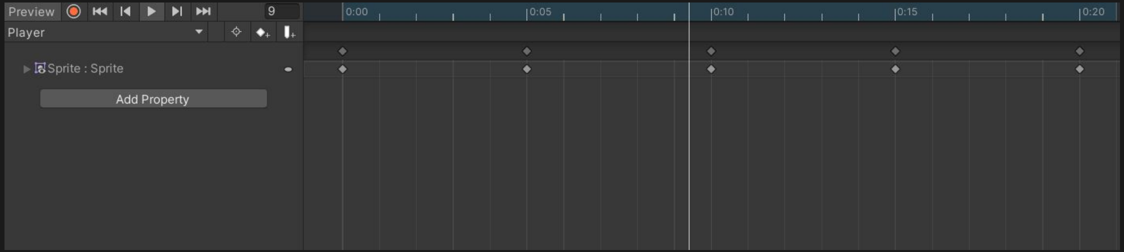


Sebastian Grygorczuk - STEM Institute at CCNY

Sprite Animation

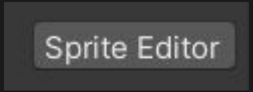
To Create a Sprite based animation you just replace the sprite render at x time intervals.

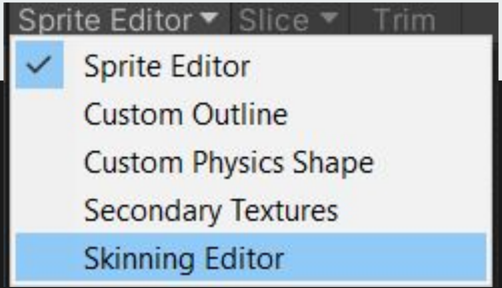
Just make sure once you reached your final frame you put the start frame at the end so you get that last frames time



Sebastian Grygorczuk - STEM Institute at CCNY

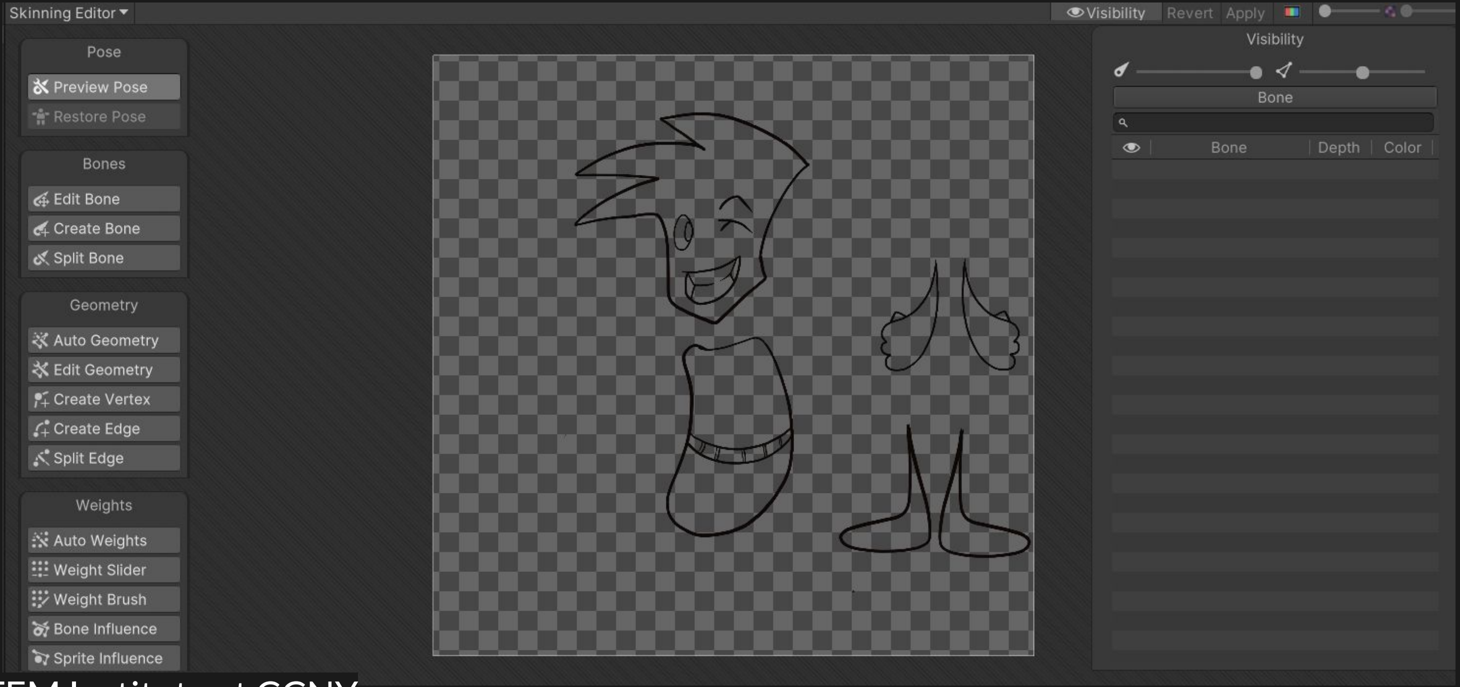
Rigging



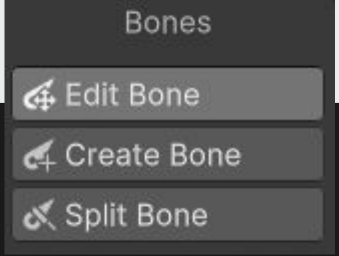


Rigging is the process of giving a skeleton to a image or 3D model that can be used to animate it.

To create bones we will go into an image Import settings, the go to Sprite Editor and switch the editor to Skinning Editor



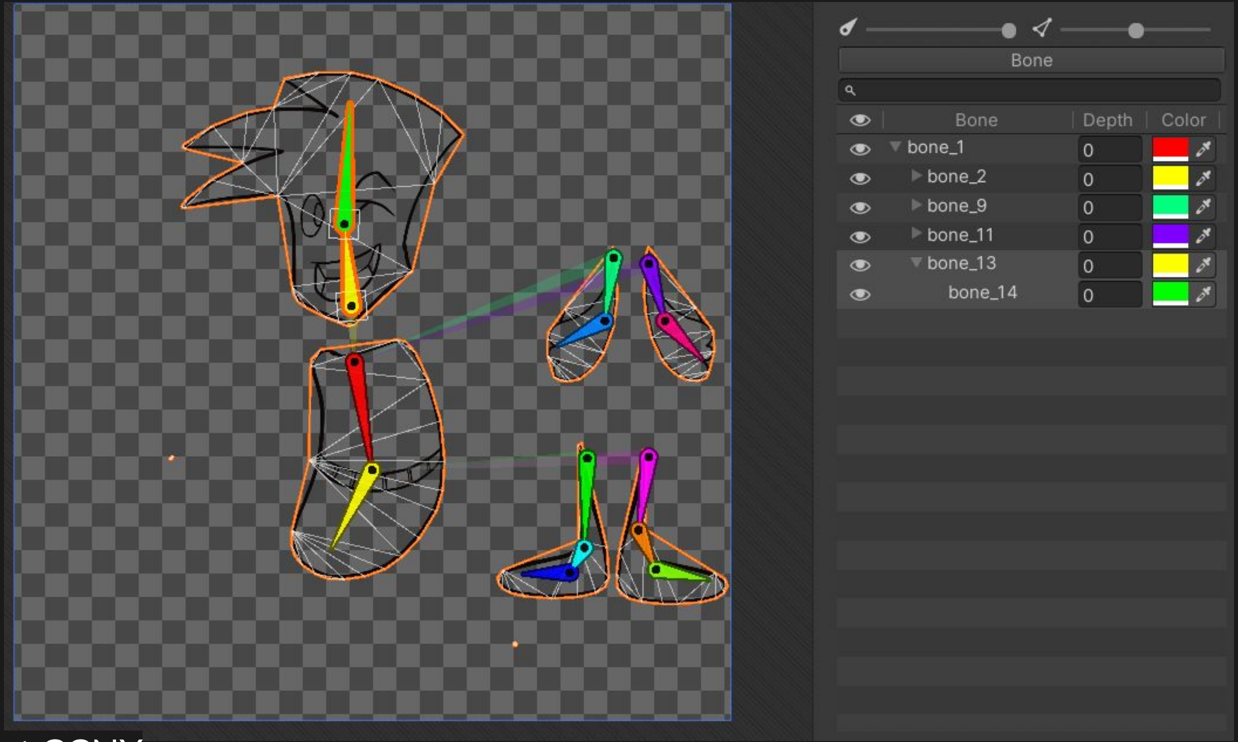
Adding Bones



Doubleclick and the select all of the objects. It will glow orange when it's selected.

Go to the bones and click Create Bone, then head to the body. Create a few in the body, once you're done right click to exit.

If you want it to connect to other part click on the circle in the and move it to another orange area, it will be transparent meaning it's connected.



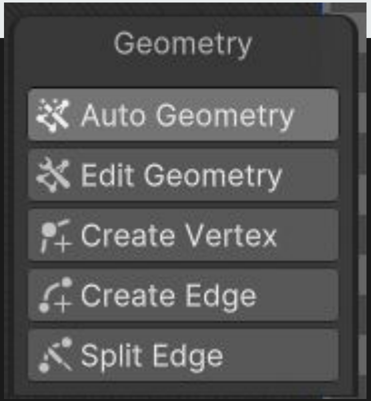
Bone	Depth	Color
bone_1	0	🟥
bone_2	0	🟨
bone_9	0	🟩
bone_11	0	🟪
bone_13	0	🟨
bone_14	0	🟩

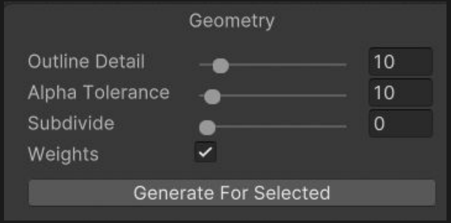
Geometry

White Lines is the geometry of the character, it's how the image is divided up internally.

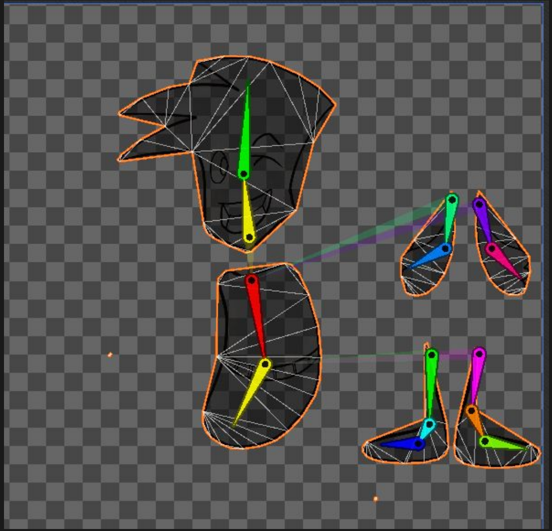
The color is the weight it shows how much the area in the geometry is affected by the bone near it.

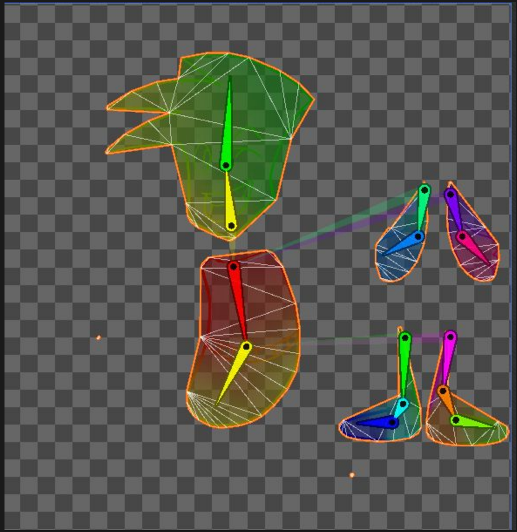
When you click on Auto Geometry the Geometry field will show up in the visibility. Click Generate for Selected and it will auto fill all of it.




Geometry

Outline Detail 10
Alpha Tolerance 10
Subdivide 0
Weights ✓



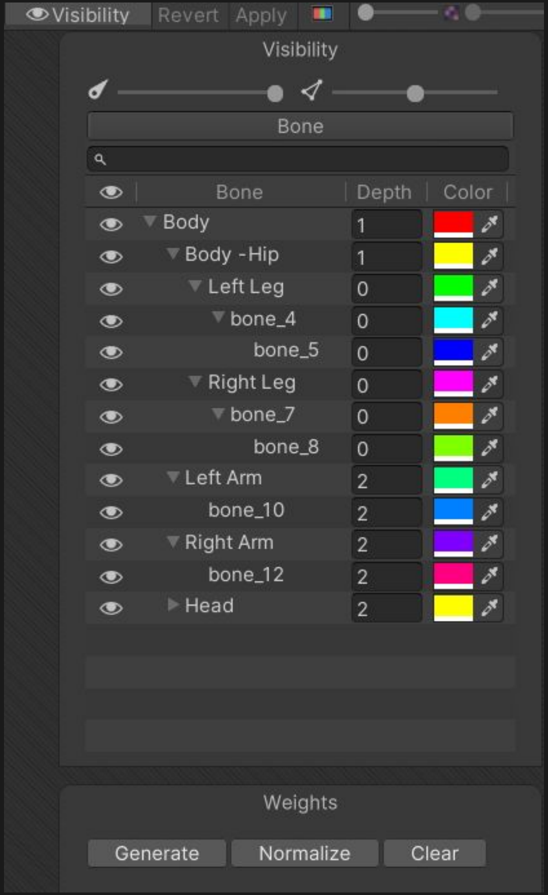


Bone Depth

Bone Depth is the order which will bones will show up above others.

It works like Sprite Layer Order. Depth being larger makes it show closer to the camera and lower further back.

In visibility you can also rename your bones. This will help you animate.



Bone	Depth	Color
Body	1	🟥
Body -Hip	1	🟨
Left Leg	0	🟩
bone_4	0	🟦
bone_5	0	🟦
Right Leg	0	🟪
bone_7	0	🟧
bone_8	0	🟩
Left Arm	2	🟩
bone_10	2	🟦
Right Arm	2	🟪
bone_12	2	🟥
Head	2	🟨

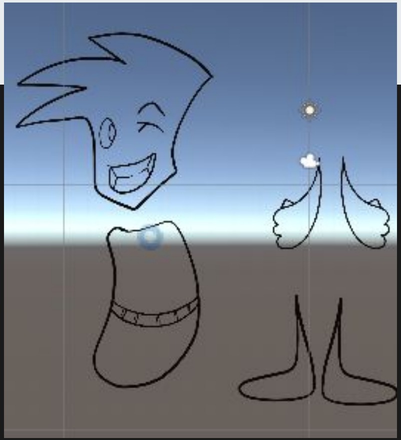
Sprite Skin

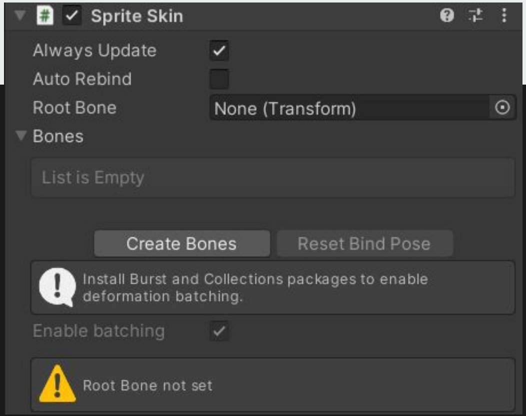
Now all of that is saved we can bring our sprite into the Scene.

You will notice that it's still broken apart. To remedy that we will add a Sprite Skin and click create bones which will create the bones and children game objects that correspond them.

Once that happens you will be able to move the limbs in their correct palace.

Sebastian Grygorczuk - STEM Institute at CCNY





Sprite Skin

Always Update	✓
Auto Rebind	
Root Bone	None (Transform)

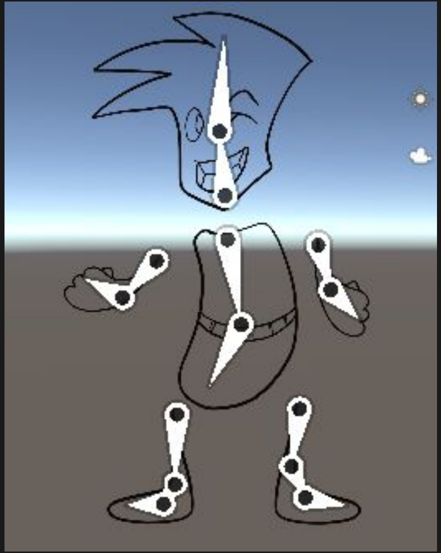
Bones

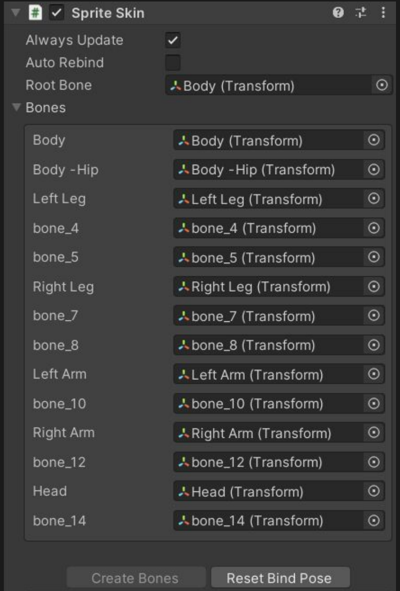
List is Empty

Install Burst and Collections packages to enable deformation batching.

Enable batching ✓

Root Bone not set





Sprite Skin

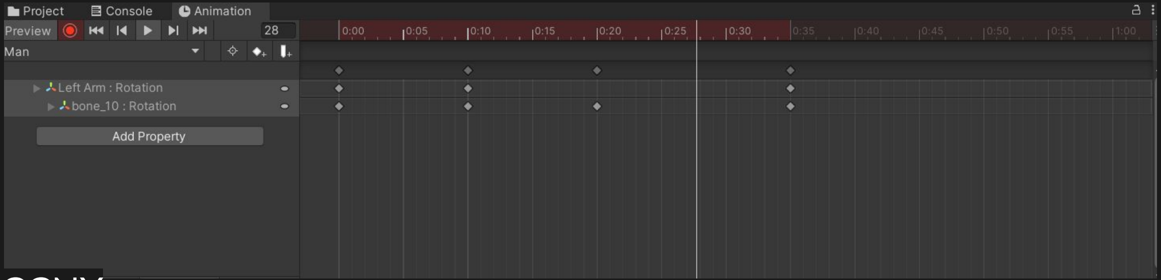
Always Update	✓
Auto Rebind	
Root Bone	Body (Transform)

Bones

Body	Body (Transform)
Body -Hip	Body -Hip (Transform)
Left Leg	Left Leg (Transform)
bone_4	bone_4 (Transform)
bone_5	bone_5 (Transform)
Right Leg	Right Leg (Transform)
bone_7	bone_7 (Transform)
bone_8	bone_8 (Transform)
Left Arm	Left Arm (Transform)
bone_10	bone_10 (Transform)
Right Arm	Right Arm (Transform)
bone_12	bone_12 (Transform)
Head	Head (Transform)
bone_14	bone_14 (Transform)

Animate

Now that we have that we can create an animation controller and use the bones to animate our character the same way we did with everything before.

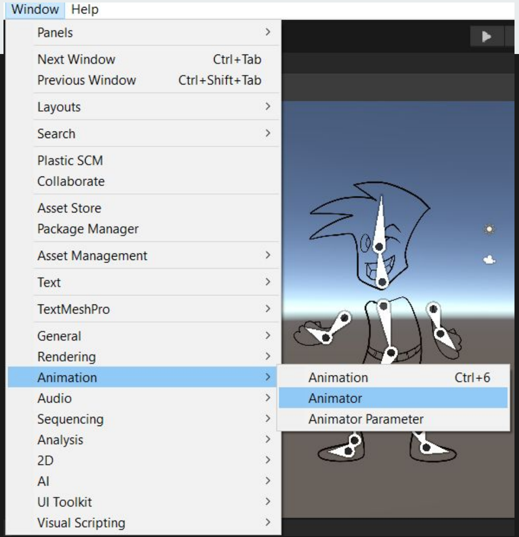


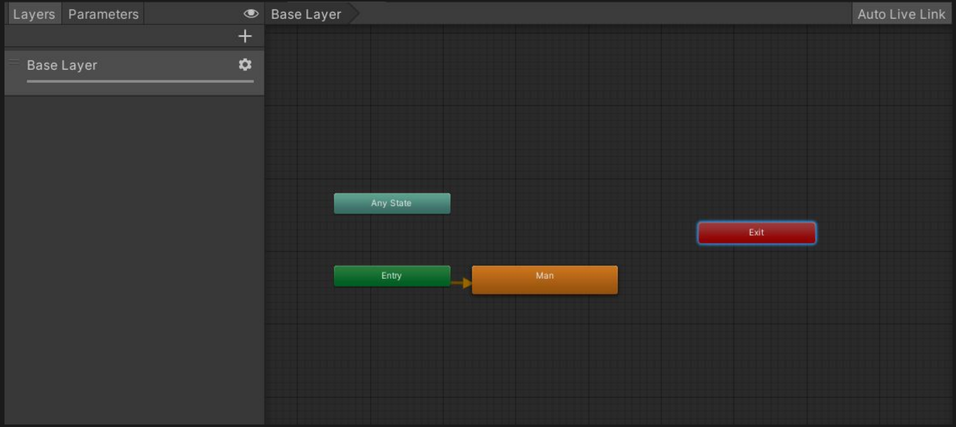
Animator View

Now that we have a couple animation set up we can use the Animator View to link them and set Transition between them.

Click Window -> Animation -> Animator

In here we will create a State Machine we'll have the animation alternate between several differentiate state based on player's actions





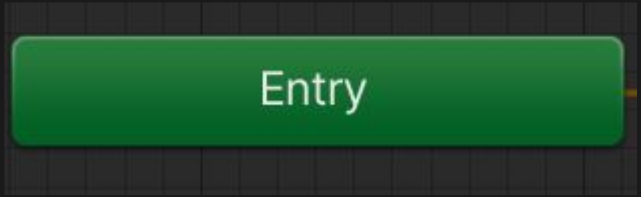
Base States

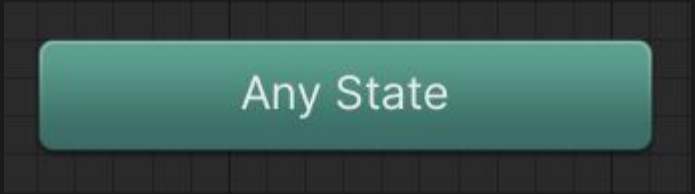
Every animator will have these three connections.

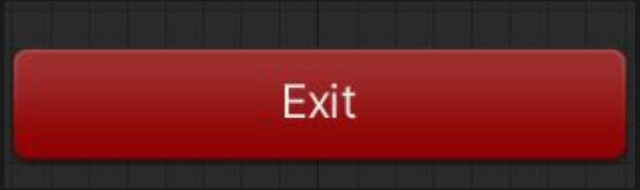
Entry is where the animation starts, it will have an arrow pointing to a Animation clips, that will be the first animation to play when the game object is in playmode.

Any State means that no matter where in the machine you State Machine you are you if the parameters permit will move to the specified animation.

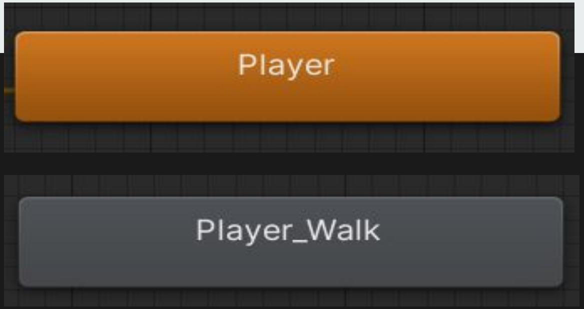
Exit lets you end an animation and go back to Entry.







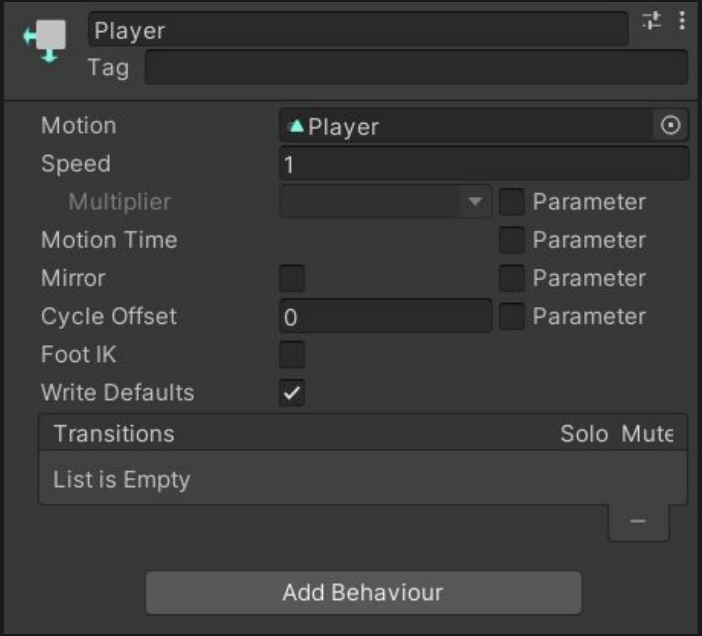
Animation Clips



Animation clips will be shown in either Orange or Gray.

Orange Animation Clip is the initial clip that will play when the game object beings to animate while the gray ones will be states that you can enter.

Currently this only has the two states we created in the Animation View but we can add additional clips by dragging them from Project View.



Player		
Tag		
Motion	Player	
Speed	1	
Multiplier		Parameter
Motion Time		Parameter
Mirror		Parameter
Cycle Offset	0	Parameter
Foot IK		
Write Defaults	✓	

Transitions Solo Mute

List is Empty

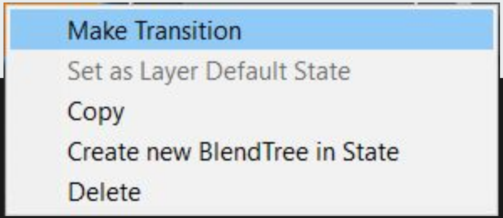
Transitions

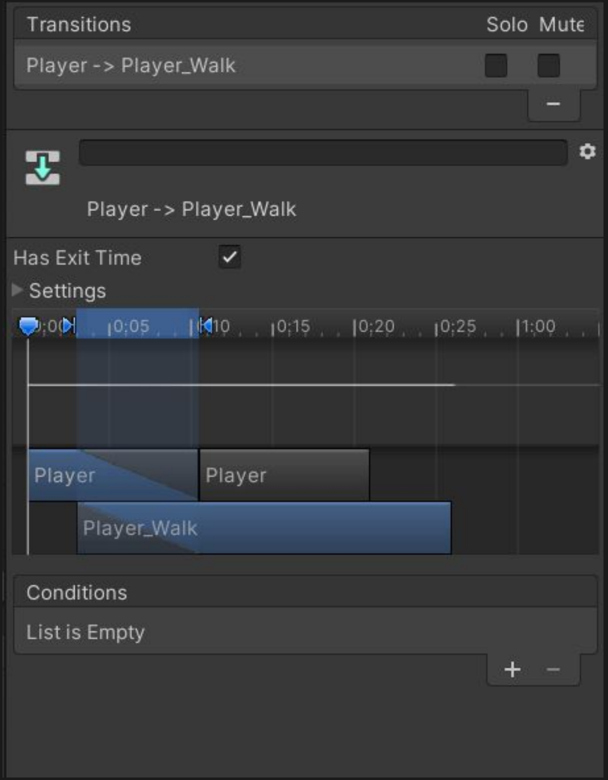
The arrows going between each animation are the transitions, when you click on the their settings.

The blue area is how long it will take to move between one animation and another.

Usually you will want it to be almost zero for things that the player controls such as movement, otherwise they feel input lag.

It can vary for other animations.



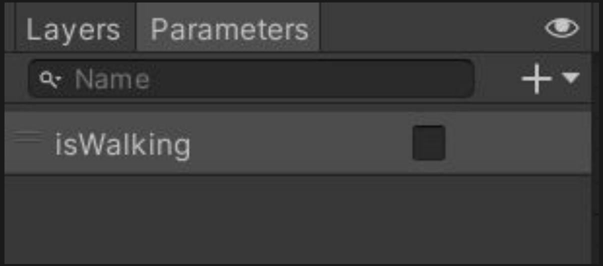


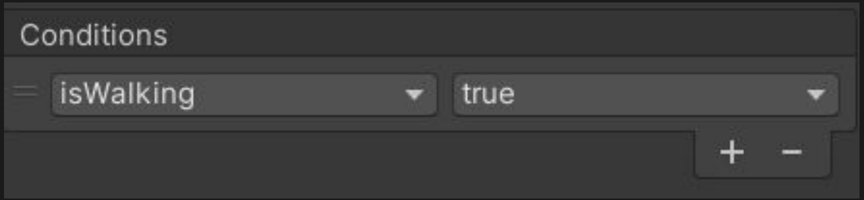
Parameters && Conditions

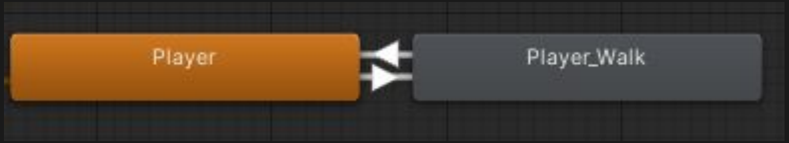
Now to have something transition between different states you will need to create parameters. In this case we will create a bool call is walking.

Then we'll click on the two trasnton, add a condition and the set one to be true and the other false.

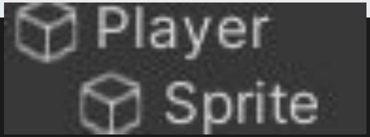
But to actually have them move between these animation we'll have to use code.

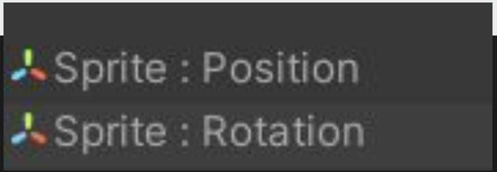




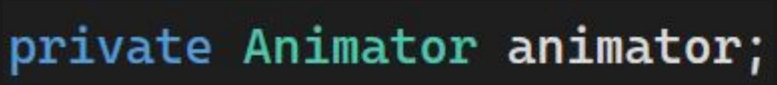


Script - Call A Single Animation

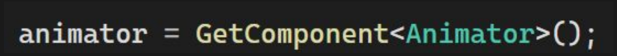




Before we update the parmenaters make sure you have a Animator class, and connect that class to the component that's on the body.

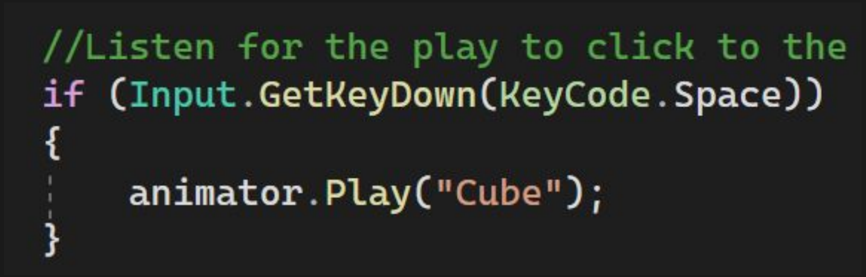


```
private Animator animator;
```



```
animator = GetComponent<Animator>();
```

You can call individual Animation by using Play(name of the animation);

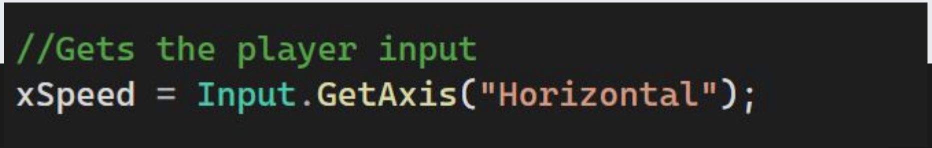


```
//Listen for the play to click to the
if (Input.GetKeyDown(KeyCode.Space))
{
    animator.Play("Cube");
}
```

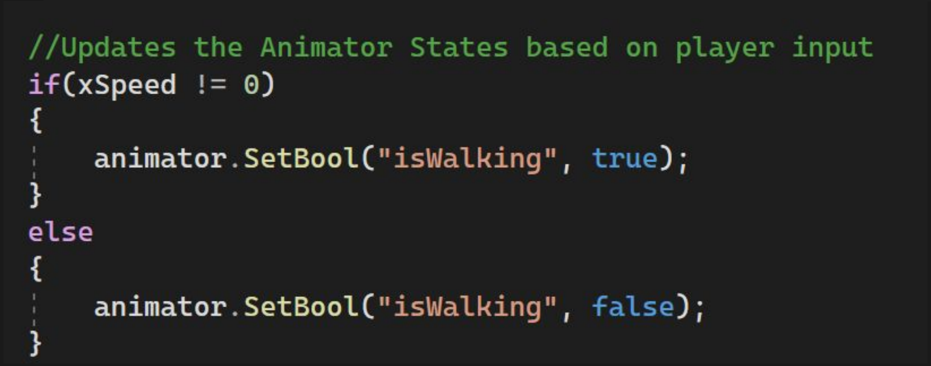
Warning: Make sure that the game objects that you are animating are not the same you will use to program, animation locks up the variables used in the component and you won't be able to change them using code.

Sebastian Grygorczuk - STEM Institute at CCNY

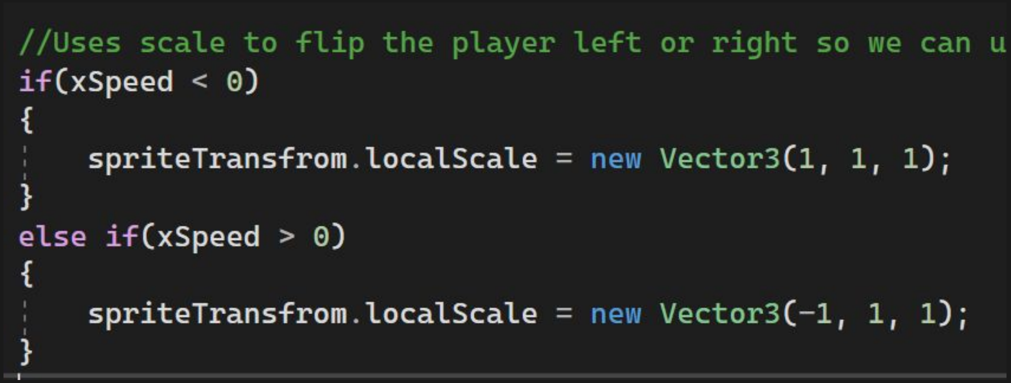
Updating Parameters



```
//Gets the player input
xSpeed = Input.GetAxis("Horizontal");
```



```
//Updates the Animator States based on player input
if(xSpeed != 0)
{
    animator.SetBool("isWalking", true);
}
else
{
    animator.SetBool("isWalking", false);
}
```



```
//Uses scale to flip the player left or right so we can u
if(xSpeed < 0)
{
    spriteTransfrom.localScale = new Vector3(1, 1, 1);
}
else if(xSpeed > 0)
{
    spriteTransfrom.localScale = new Vector3(-1, 1, 1);
}
```

Here we have two snips of code that let us animate our character properly.

We take in input from the Horizontal keys and check if the player is moving, if they are we set the isWalkingParamter to true otherwise false.

We can also update the scale, by setting the x axis to be negative when the speed is 0 we will have animation look like it's moving in the opposite direct saving us the trouble of animating several walk cycles.