Immersive Environments Tutorial

Goals:

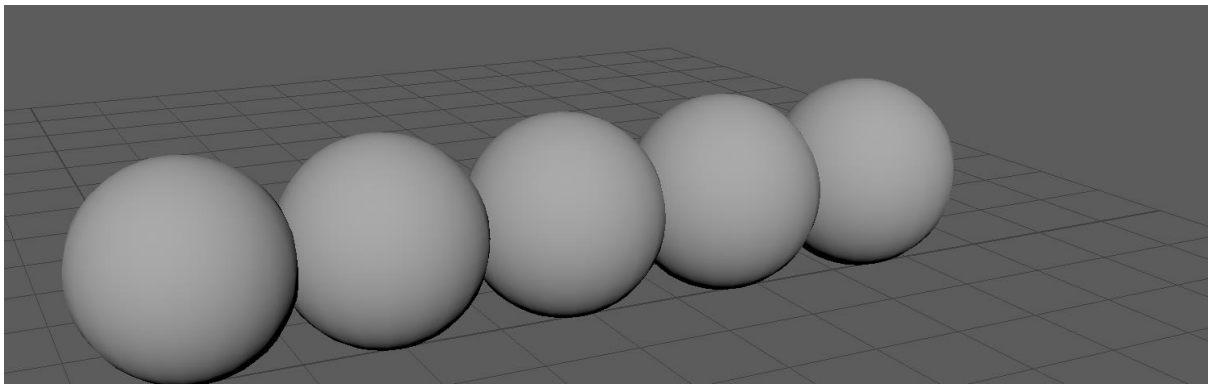
- Rigging
- Ik Rigging
- Basic Animation

Load up Maya

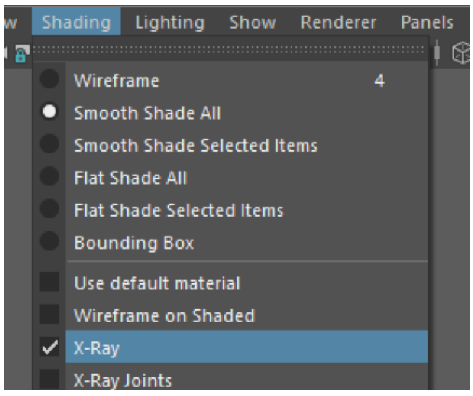
Build Object: Manual Rigging

Aim: Create a few differing shapes, rig with a skeleton structure

To being with, create 5 spheres and arrange them in a line. Like the following image:

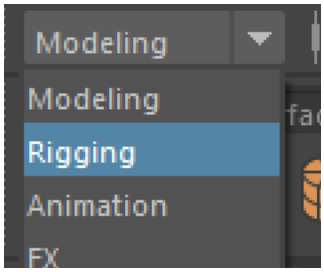


For this example, we don't need to worry about changing the objects name. Next turn on X-ray through the view's shading menu.

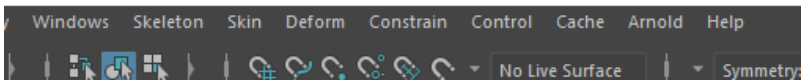


We need X-ray turned on so when we link the joints up, we can see what is occurring.

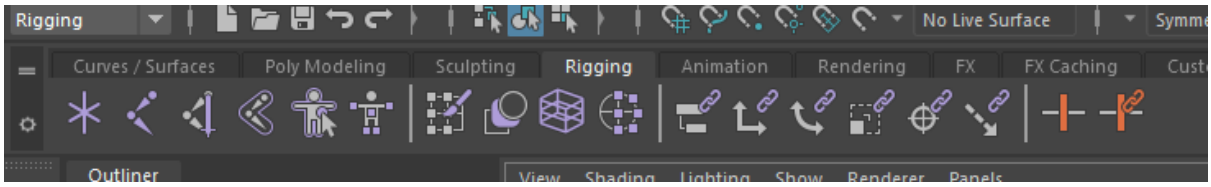
Now, to get to the rigging side of Maya we need to use the drop down menu and change from Modelling to Rigging.



This allows access to the normal menu system for rigging elements:

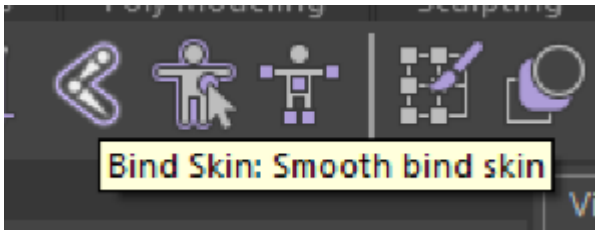


We will also want to change the Maya Shelf to the Rigging shelf.

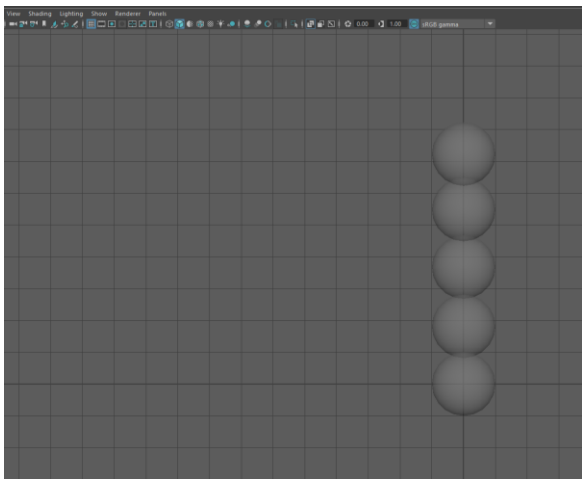


The second and forth icons are the ones we will be working with on this object, the second icon is the Create Joints ability.

With the fourth being Bind Skin, which combines the skeleton made of joints and the object it is within.

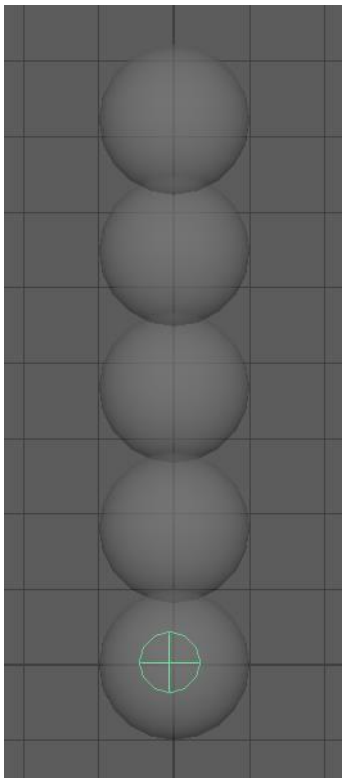


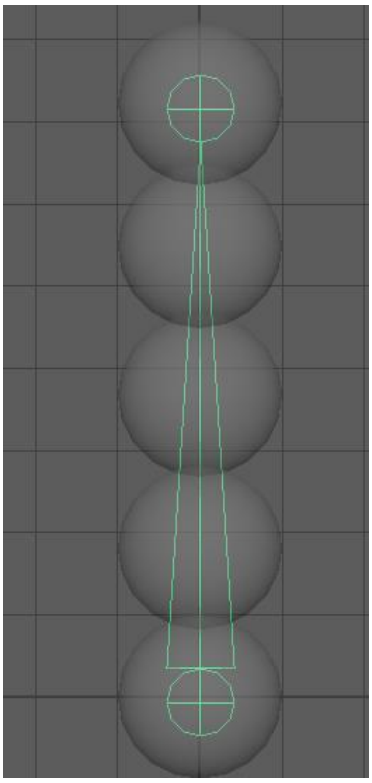
To make this adding a skeleton easier, you will find that you will bounce between top, side and front view. In this case, we should be okay with just top and side. Ensure that X-Ray is turn on in those view, then go to the top view.



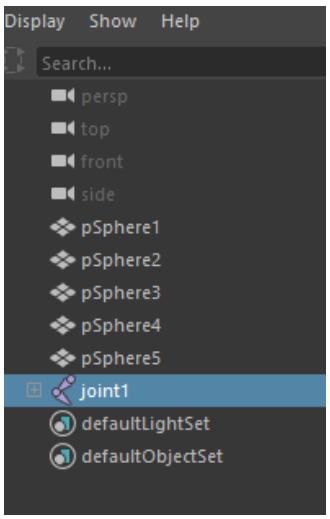
From here, click on the create joints icon. 

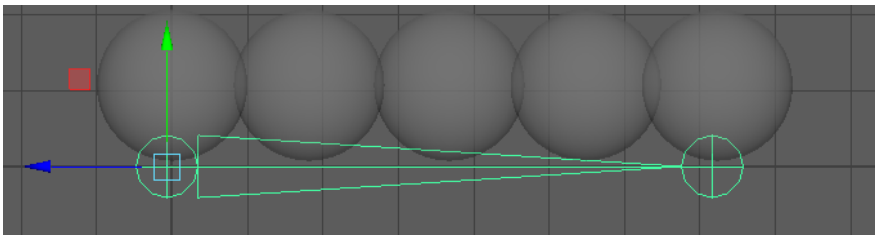
Once click, select the first point of the joint, in this case I will be starting from the bottom one. This first click creates the parent node of the joint. Then finish making the joint by clicking in the middle of the 5th sphere. To stop making joints, push Q or click on the arrow icon in the toolbar.

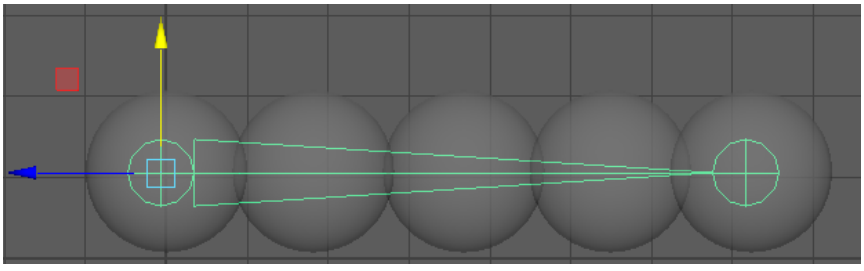




This creates a singular skeletal joint. Now Swap to side view and position the joint in the middle of the spheres. You can move the joint, as you would an object, you might find it easier to highlight the joint using the outliner.

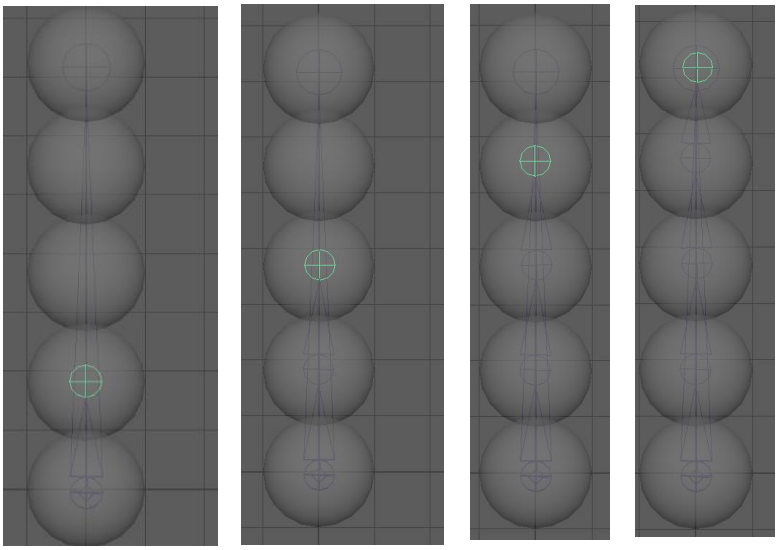






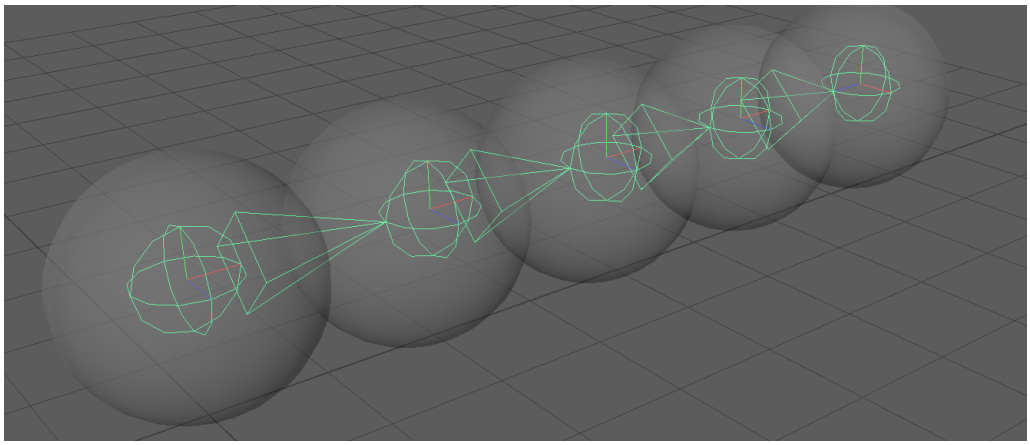
Now that it is all centred, we can add additional joints, one for each sphere. To do this, go to top view.

Select from the menu Skeleton-> Insert Joints. Then click on the same starting point again, then click in the centre of each sphere. This will add the joints to the primary long joint we created.

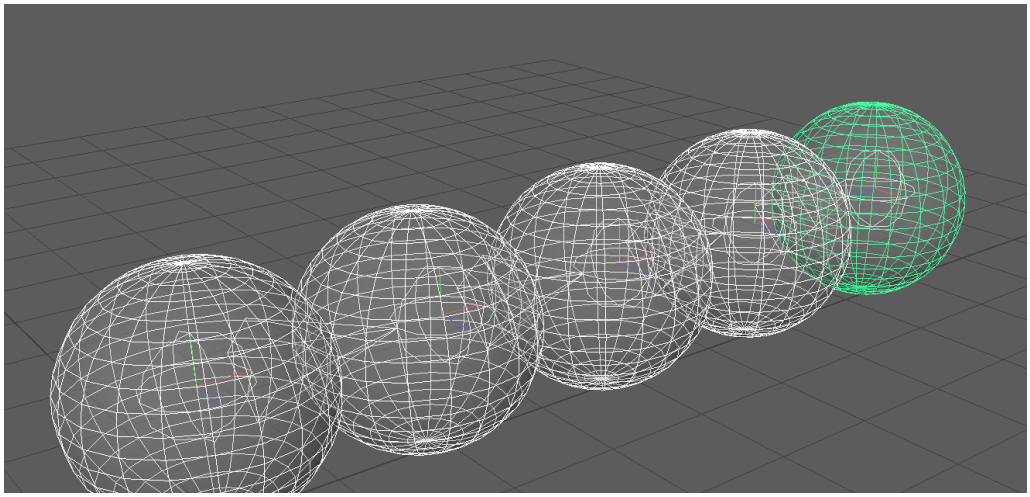


As before, to stop making joints, push q to swap to select mode or click on the arrow icon in the toolbar

Swap to perspective view.

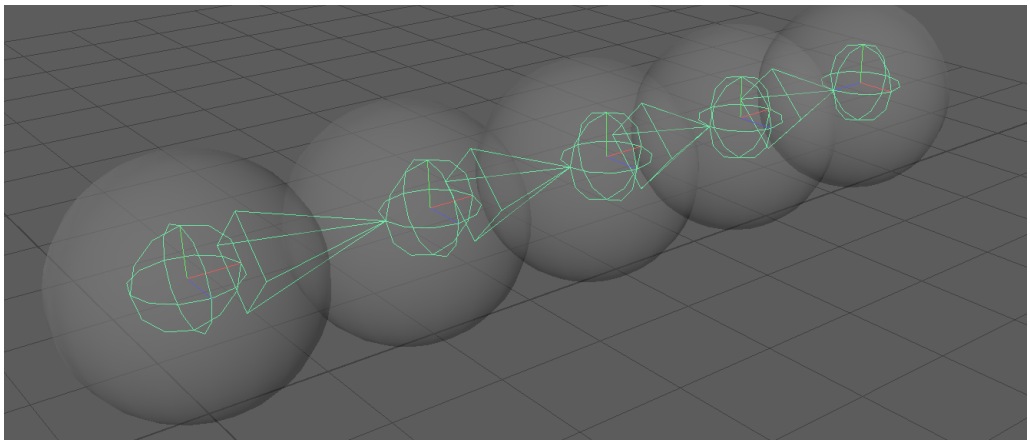


Hold down shift and select all Objects.

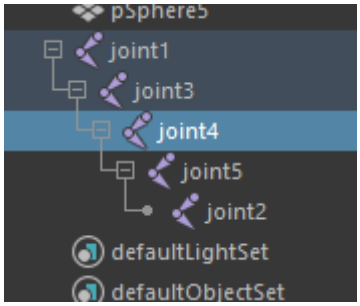


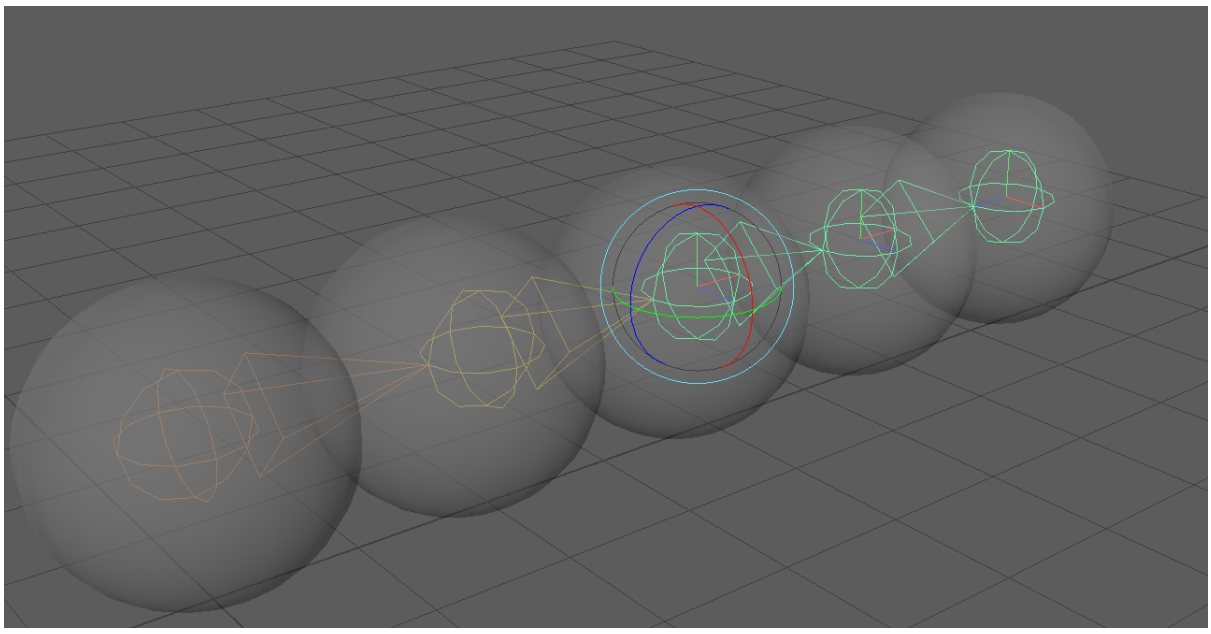
Next, we have to bind the objects together, with all items selected, click on the forth icon, Bind skin.

This will swap back to the normal view.

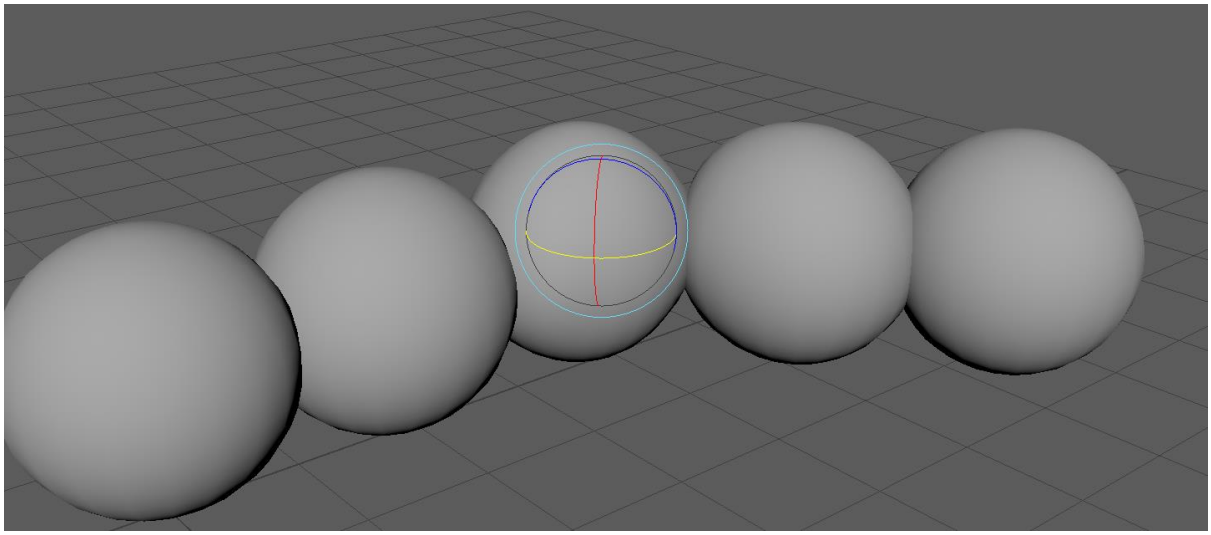


Now, change to the rotate tool and click on a joint. This can be done through the outliner. You can also rename the joints to make them easy to locate on complicated models.

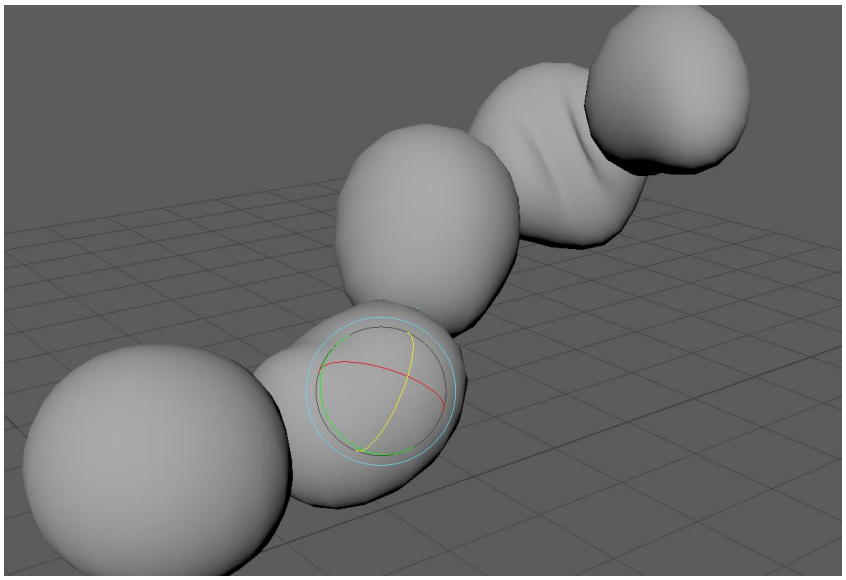




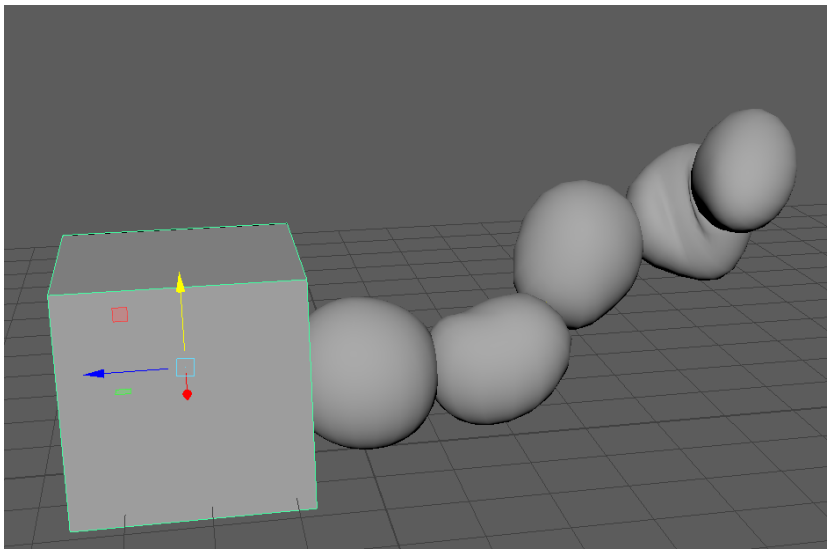
Apply a rotation.



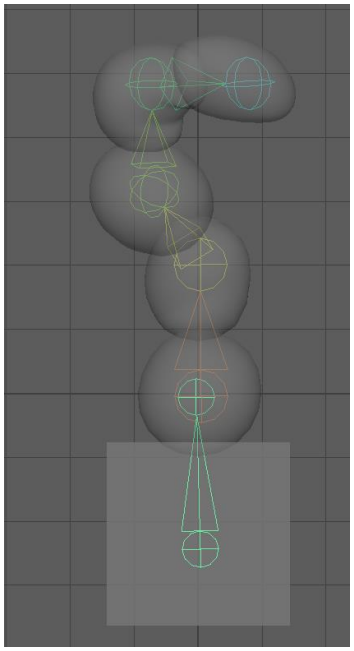
Repeat on as many spheres as you want.



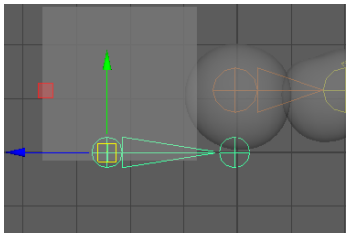
Now, that we have the basics down, we will add an addition element to the scene. Create a cube and move to the side of the first sphere

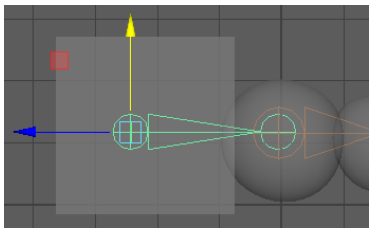


From here go back to the top view and add a new Joint from Cube to the first Sphere.



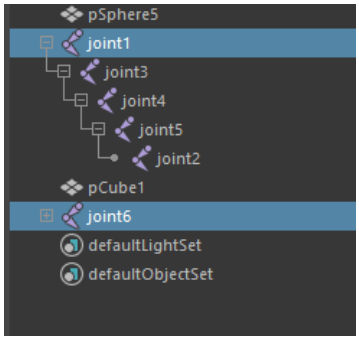
Remember to go to the side view to position the new joint at the right height of the cube.



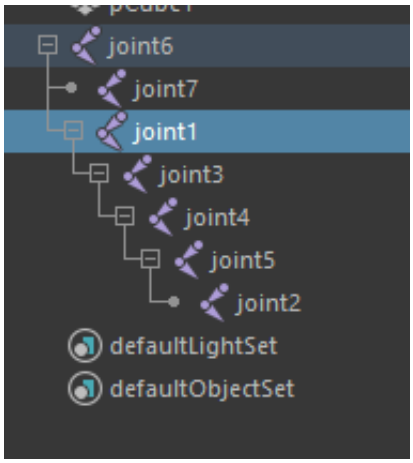


Even though we've position the end of the new joint on the start of the original joint, they are not connected, so to fix this we have to link them together and turn the cube joint into the Parent point. To make this happen, highlight the Sphere joint, then holding SHIFT, click on the cube joint. And then push P.

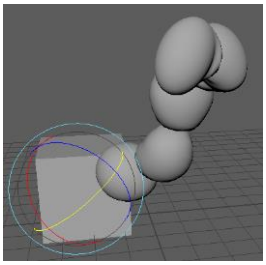
To Ensure you only have the two main points, check the Outliner.

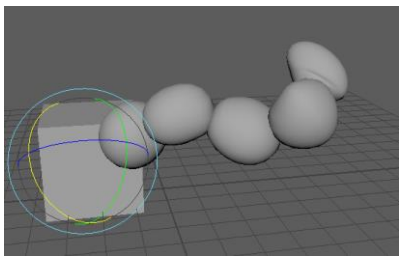


After you have pushed P, you should see the outliner show the following.



Notice that Joint 6 is now the Parent of Joint 1. From here, if you rotate the joint on the cube, you should see all of the spheres move with the rotation.





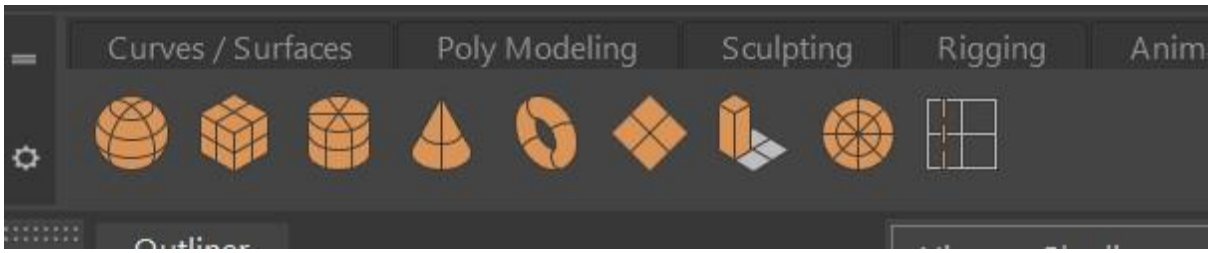
Build Object: IK Rigging

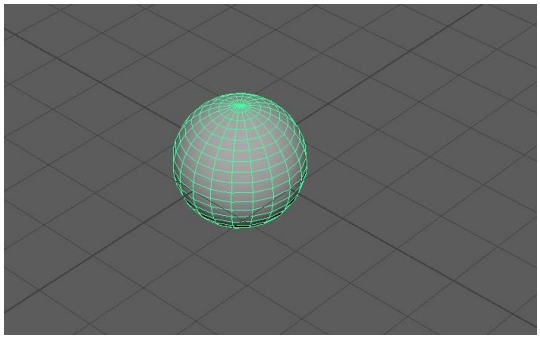
Aim: Create an arm, add joints and ik handles.

IK Rigging, Inverse Kinematics, is a method of controlling models that have joints to allow for multiple joint movement to correlate with a singular movement. So, instead of moving each joint separately, the movement of an end joint effects the positioning and rotation of all joints from parent through to child.

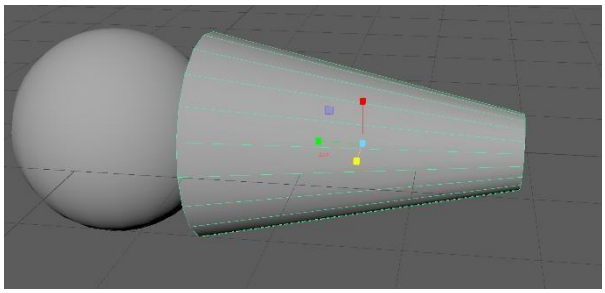
To start with, let's create a simple robot arm. Start a new Scene.

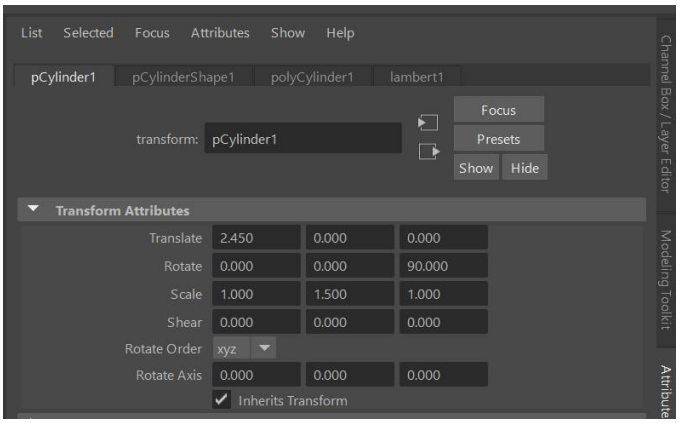
Create a sphere, from the custom shelf, click sphere



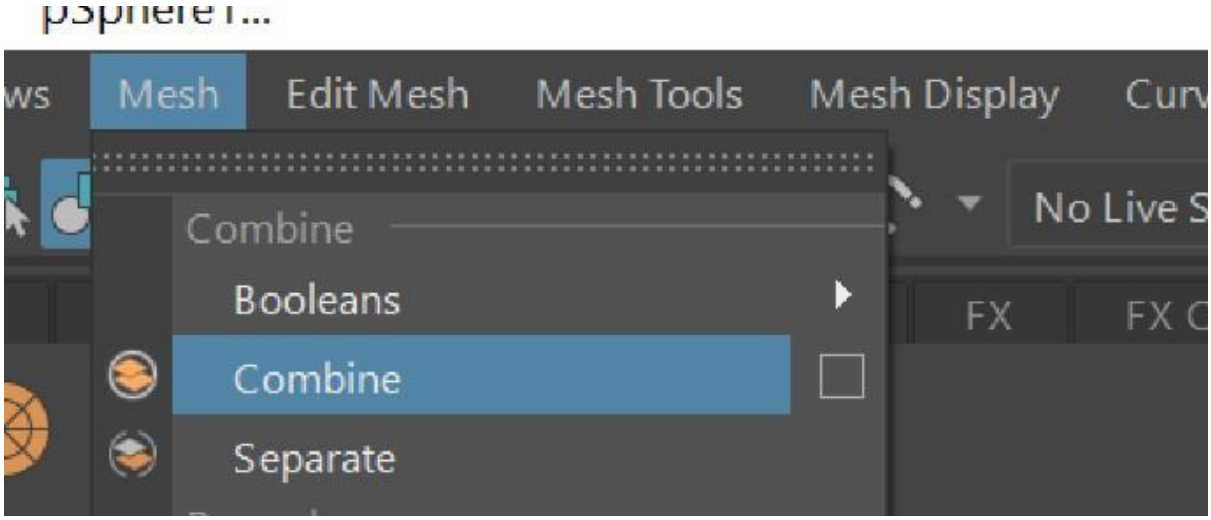


Next create a cylinder and position it to the sphere, once the cylinder has been created, you can make the cylinder in the correct position by changing the rotation which can be located in the attribute editor





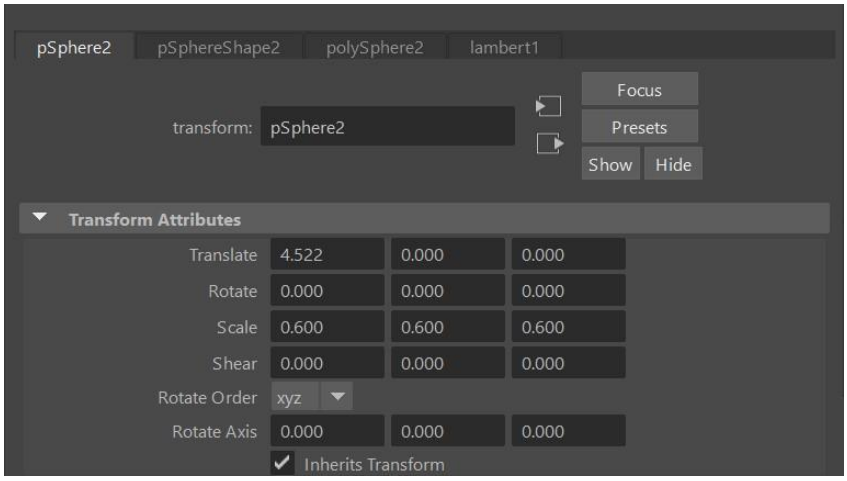
Once you have that shape, combine them into the one object, use Mesh combine.



Create a new ball and cylinder, scale them down a little bit and move them to the end of the first cylinder.

Sphere scaled down to 0.6

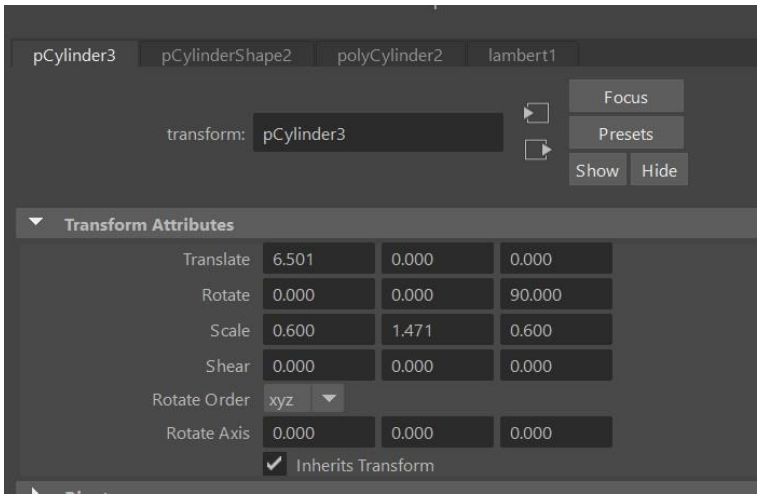
Sphere translate is 4.522



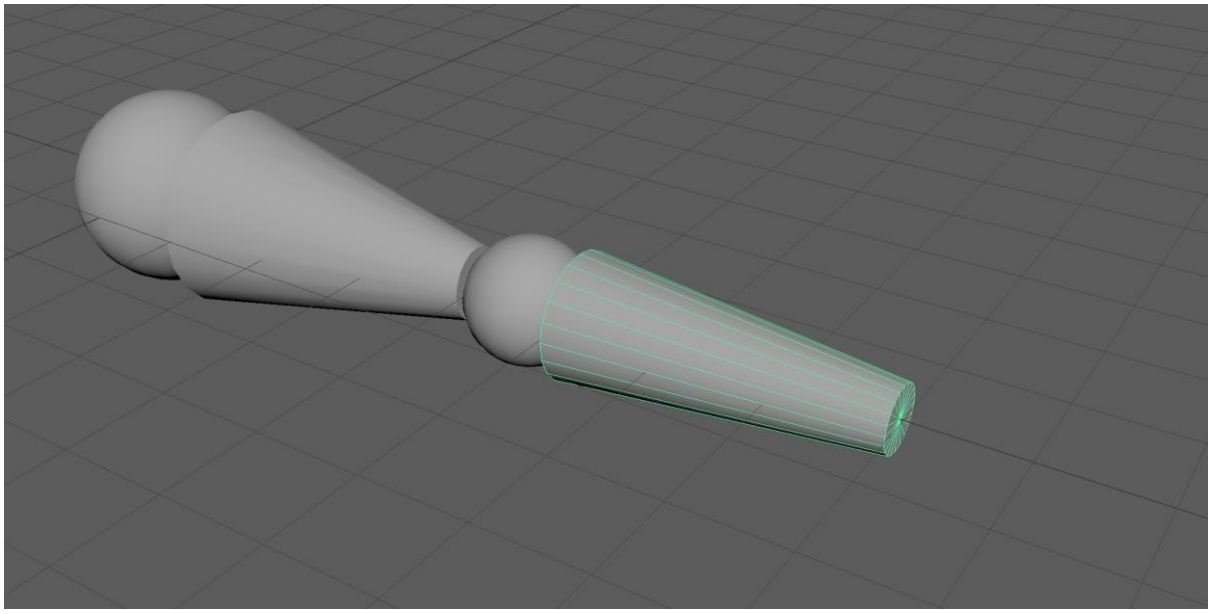
With the cylinder

Scale is 0.6/1.471/0.6

Translate 6.501



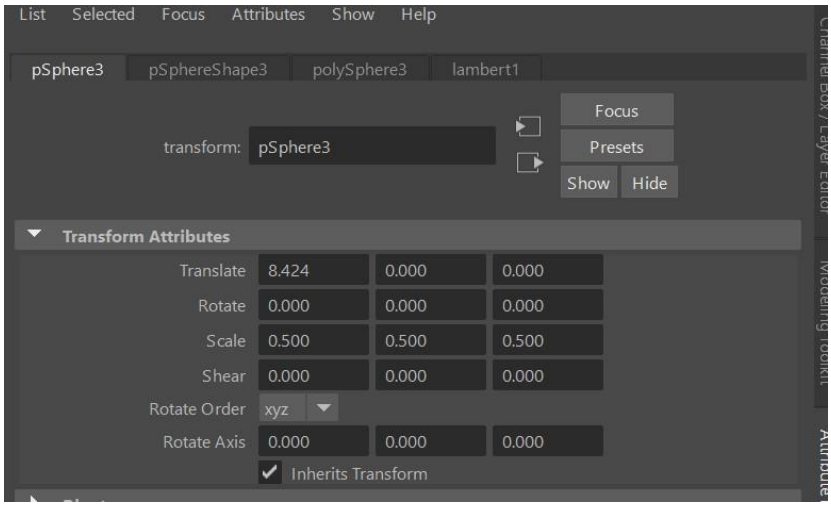
This should give you the following.



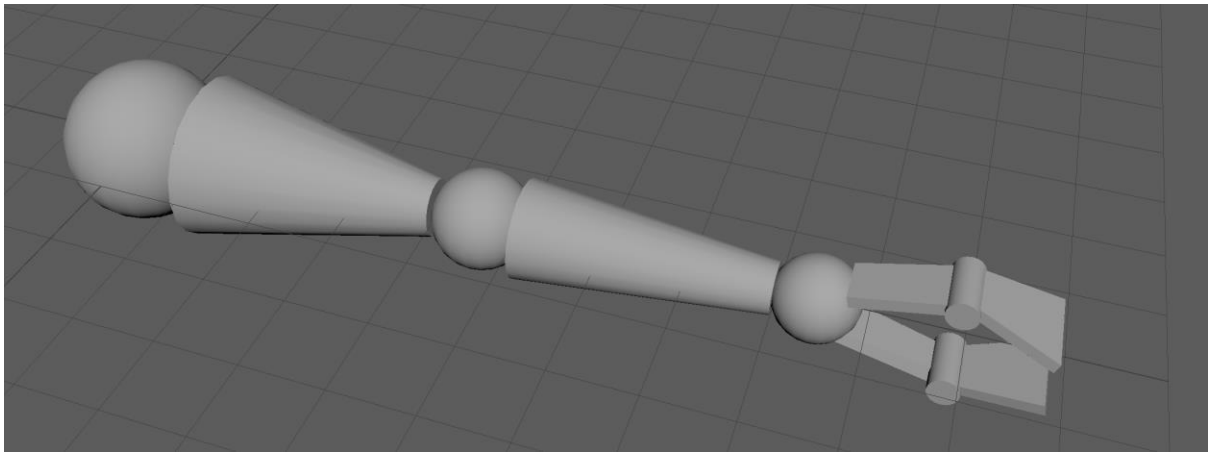
Next, we will add the wrist ball and two pincers to the ball.

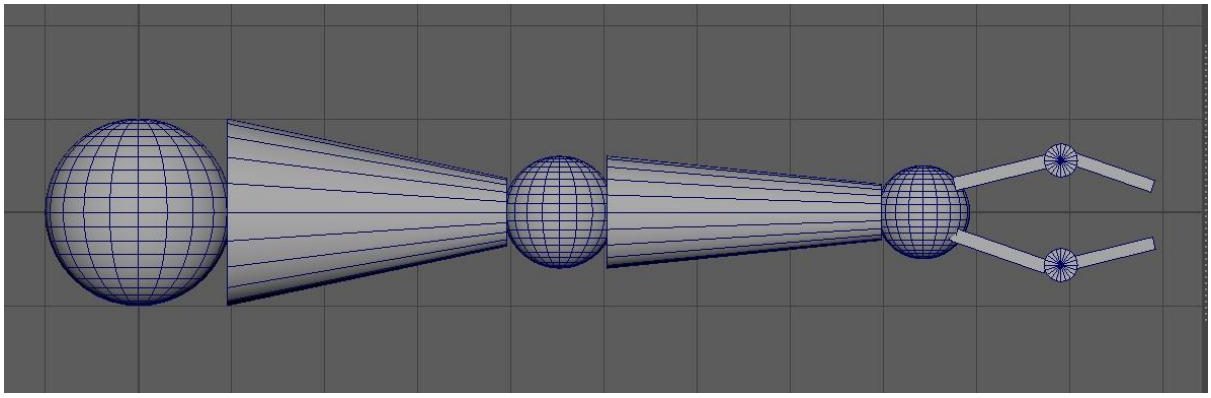
The wrist sphere is scaled to 0.5

Translate of 8.424



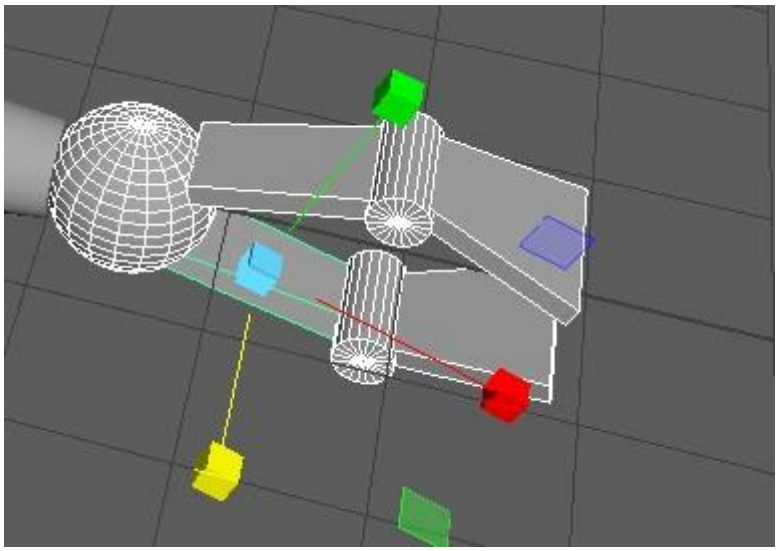
The pincers are composed of flattened cubes with a cylinder connecting as a joint. You can ‘eye-ball’ these positions if desired, but make the final product look like this:



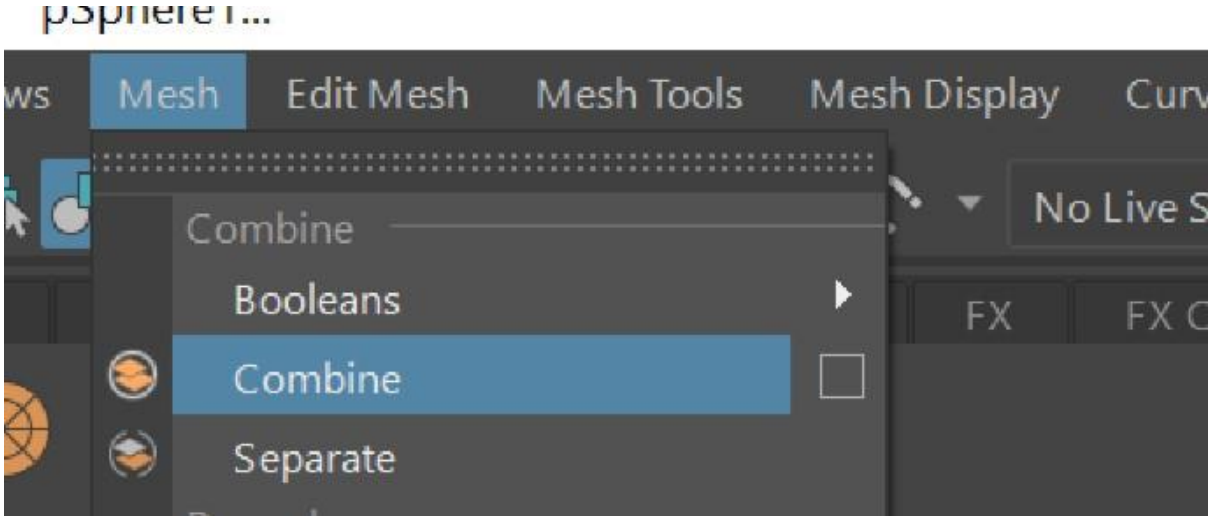


Next, we want to apply a joint structure to the arm, basically we want the parent to be the large sphere with child joints down. To ensure that the pincers move with the ball, combine the meshes as we did with the first sphere and cylinder.

Highlight the sphere, cubes and cylinders

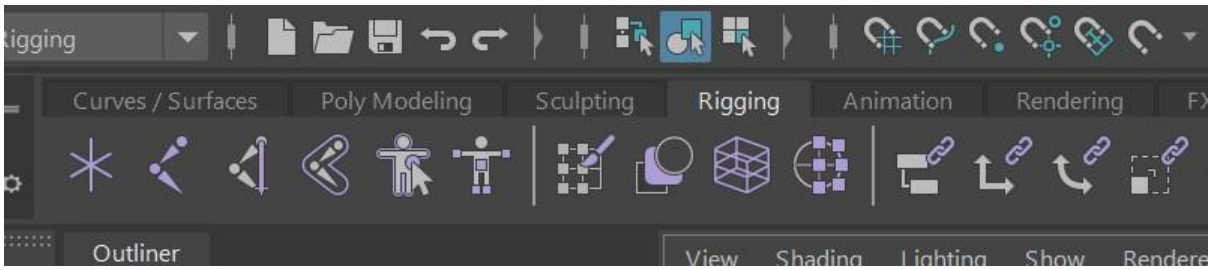


Then, mesh combine:

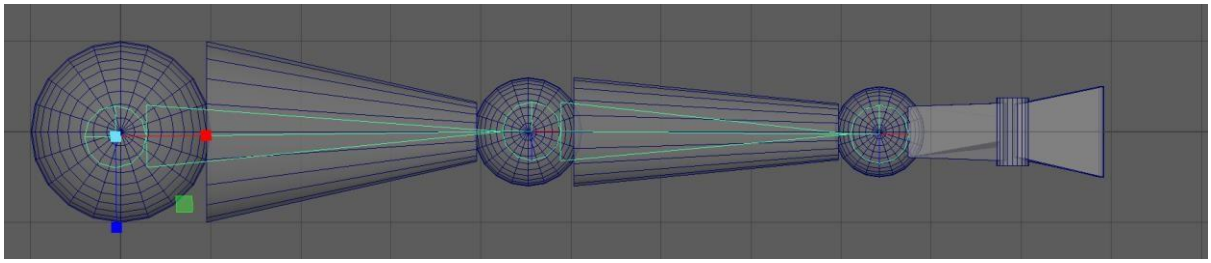


If you wanted to be more precise in the movements, then you would link the pincers separately.

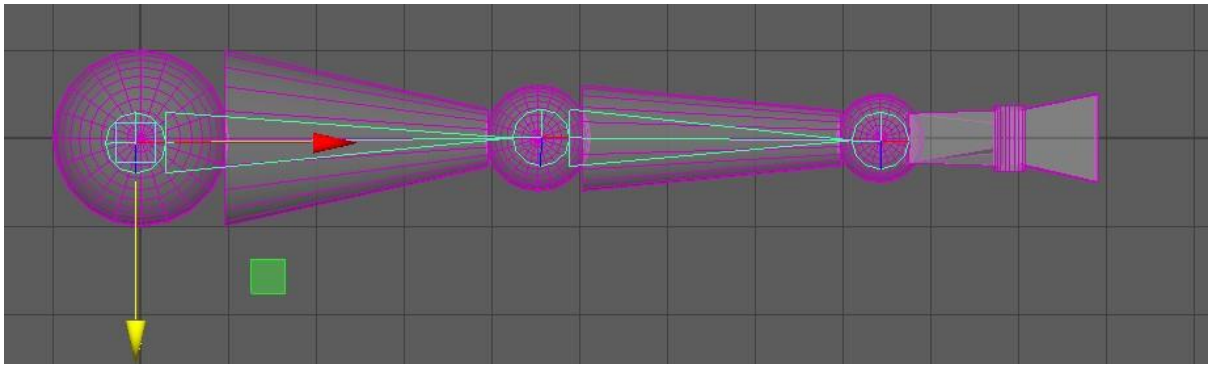
From here switch to top view and change to the rigging shelf.



Create the joints to each main sphere.



Bind the skeleton to the mesh we've created. Highlight the joints and mesh then use the Bind skin icon.



Quickly test each joint to ensure the binding worked, i.e. add rotation, rotate and then undo the rotation.

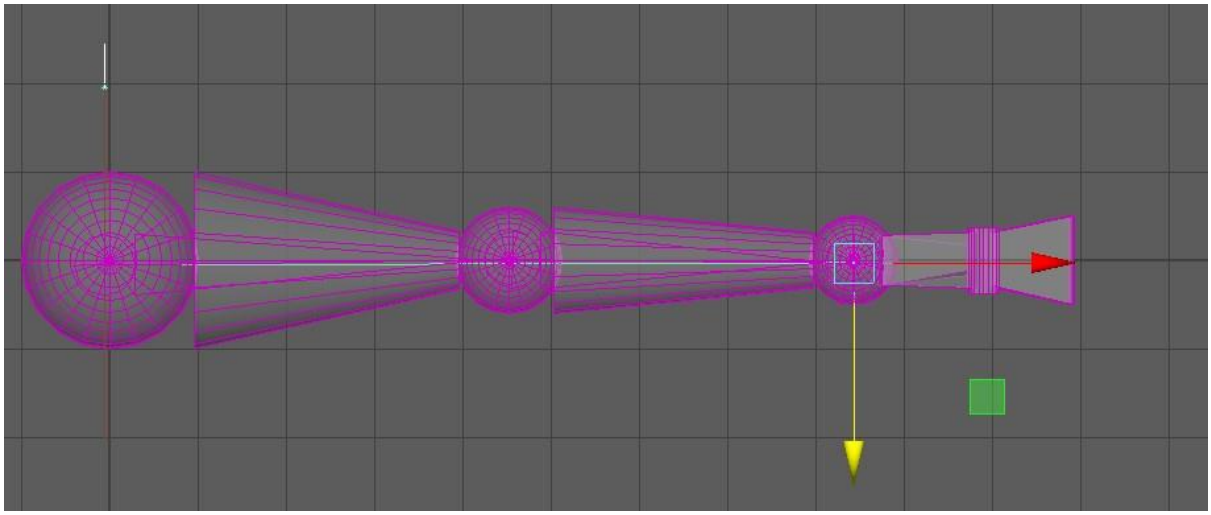
Next, we can add the IK handles to it.

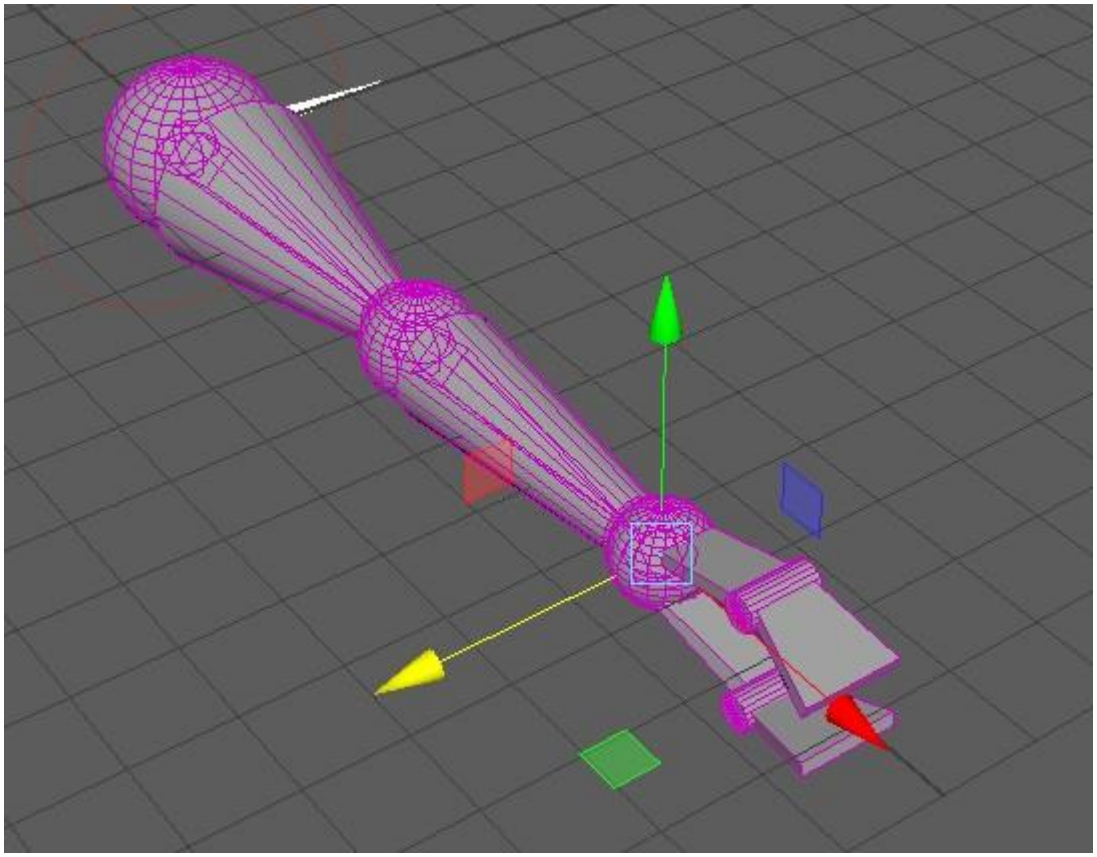
Use the following icon.

From the top view, click on the shoulder joint in the centre and then click on the wrist joint, in the centre.

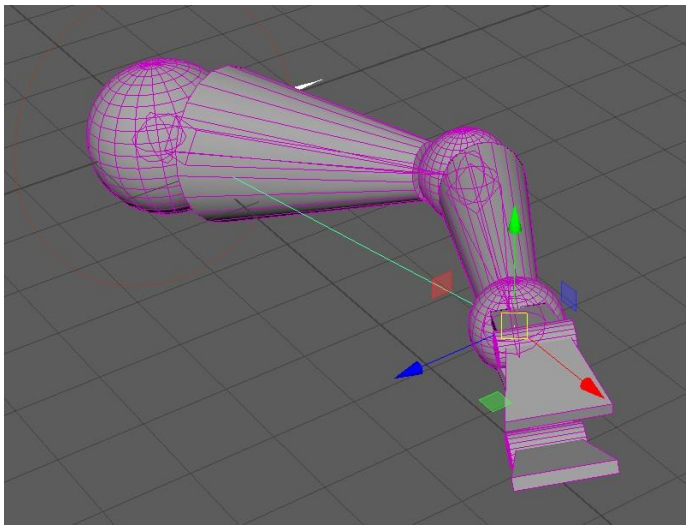
Once you have clicked, you can push the enter key to finish the joint.

If you look at the top view you will see a line appear above the parent joint, this is also visible in the perspective view.





Next, test the joint, go to perspective view, use the move tool to move the ‘hand’ section.



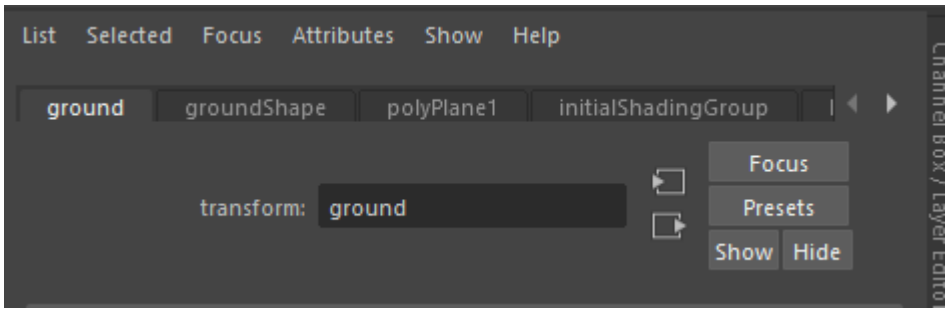
As you can see, the middle joint has moved due to the IK rigging. This will produce more natural movement in your models.

Build Object: Basic Animation

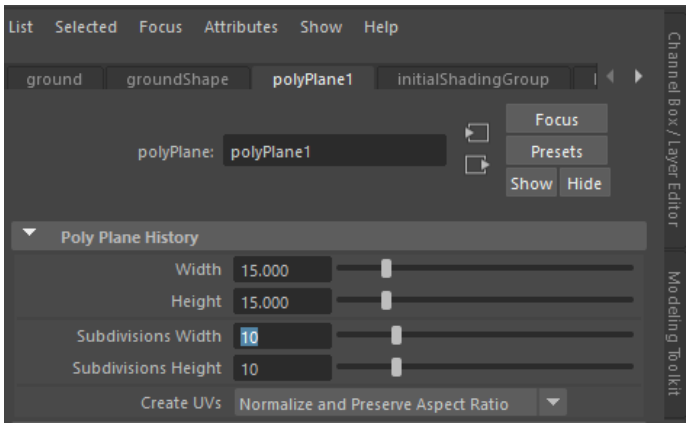
Aim: Create some objects and animate them to move on the screen.

Animation comes in many differing ways in Maya, there is key frame, graph editor animation and the use of dynamic/physics animation. To start with, we will look at key animation.

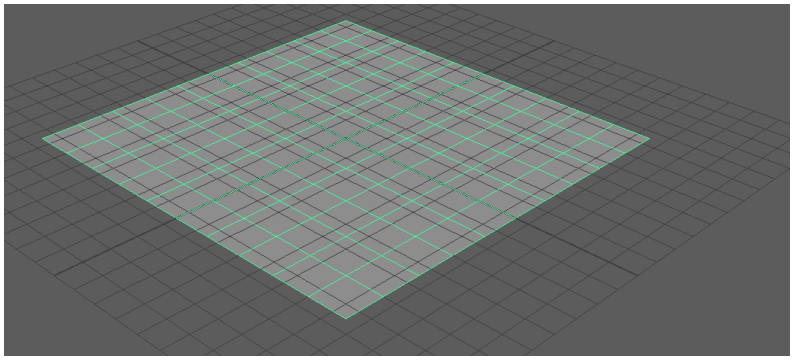
In Maya, draw a plane and a sphere. Name them ground and ball respectively. Move he sphere to the edge of the plane.



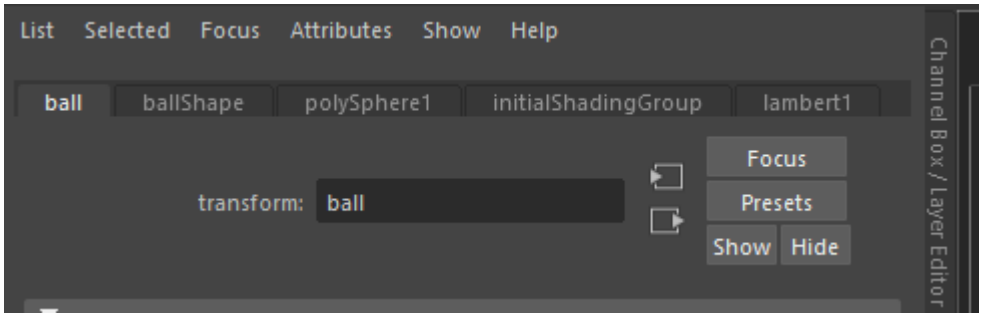
Scale the plane up from the attribute editor.

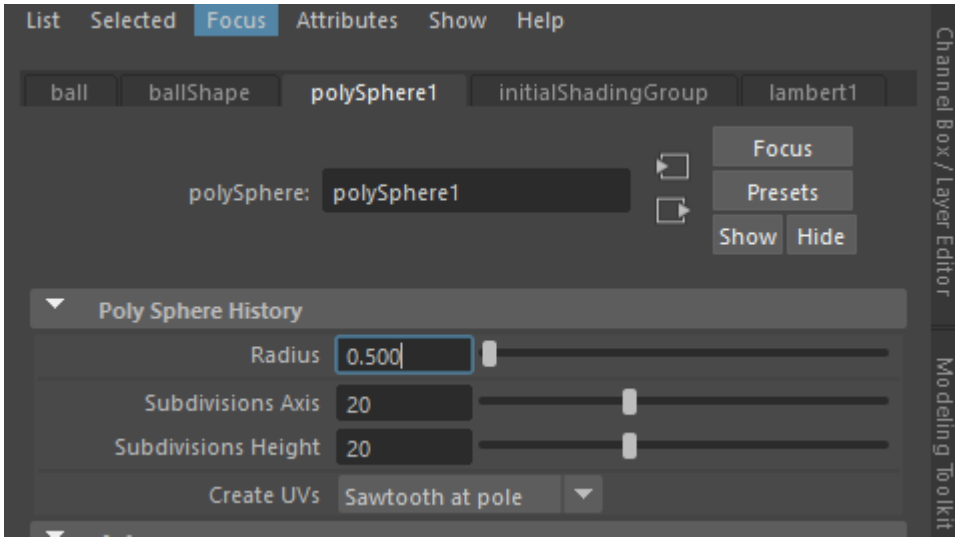


This should give the following scene in perspective view.

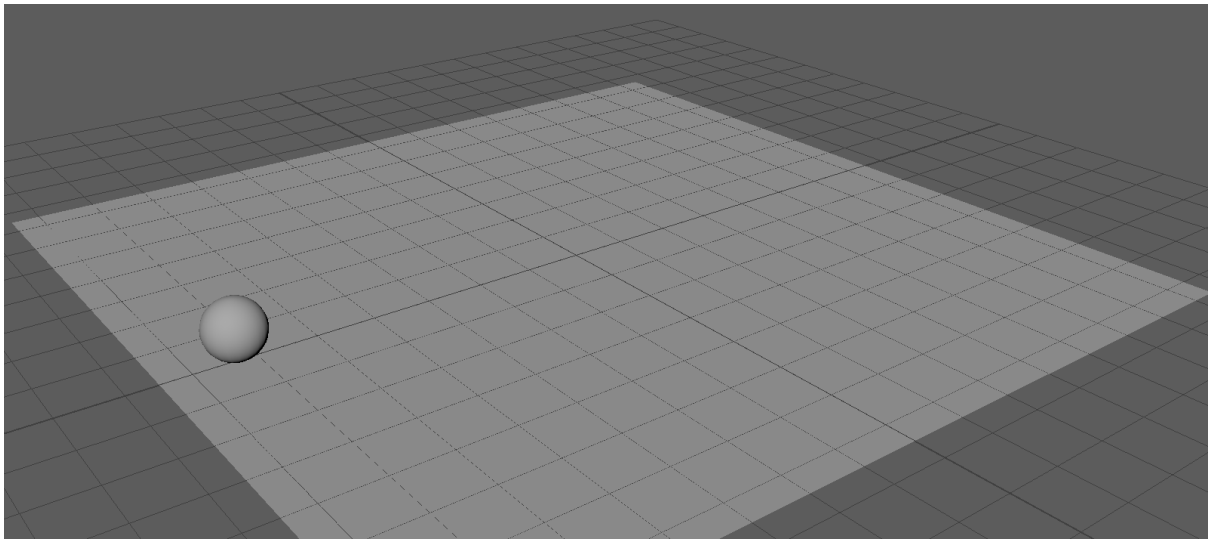


Now add the Sphere, name, scale and move the sphere.



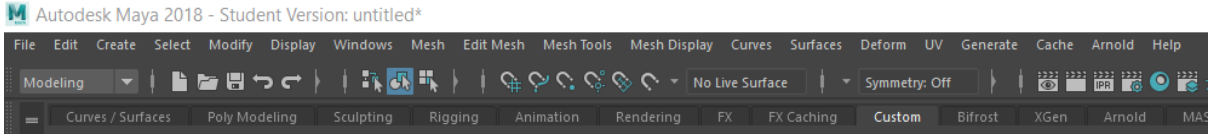


This should give the following in perspective view

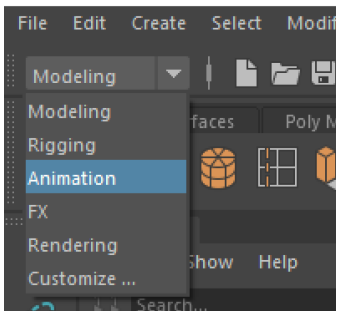


Now that we have the basic layout of the screen, let's examine the UI of Maya to determine the points that we need.

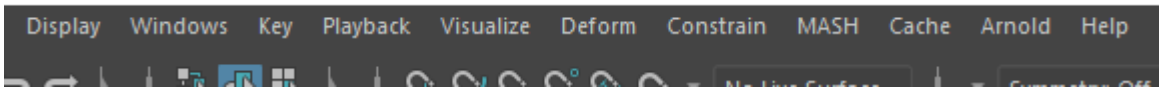
To start with, the basic menu system is set up like this:



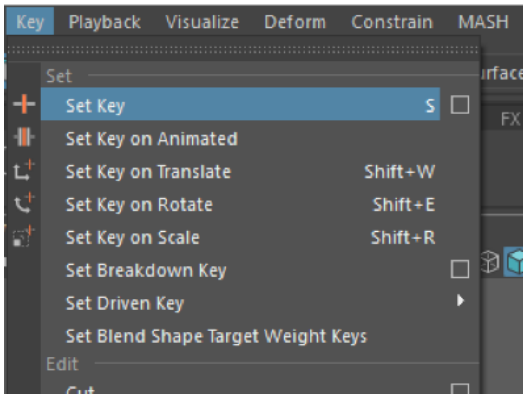
Notice how we have modelling in the drop down, we will change this to Animation.



This change will also affect the menu options that are being offered. As you can see, there is a new menu option called Key, which we use for key frame animation.



The Set key option is the one we will use the most for this basic set of animation.



Now we need to look at another aspect of the UI of Maya. And this is the frame counter and playback areas at the bottom of the screen.

The top bar is where you can set key frames, in the above image it is constrained as I shrunk the Maya application allow that much detail to fit on the screen. Below the frame counter, there is a scroll bar, which allows you to modify the number of viewable frames you see, it's default is about 120.

The two input boxes 120/200 are the end of playback frames and end of animation frames. If you increase the playback, ensure your animation frames exist long enough, i.e. playback becomes 300 frames boost animation to 300 as well.

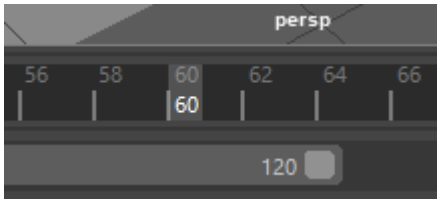
In the top right is the play/stop controls you would expect to view animation.

Key Framing

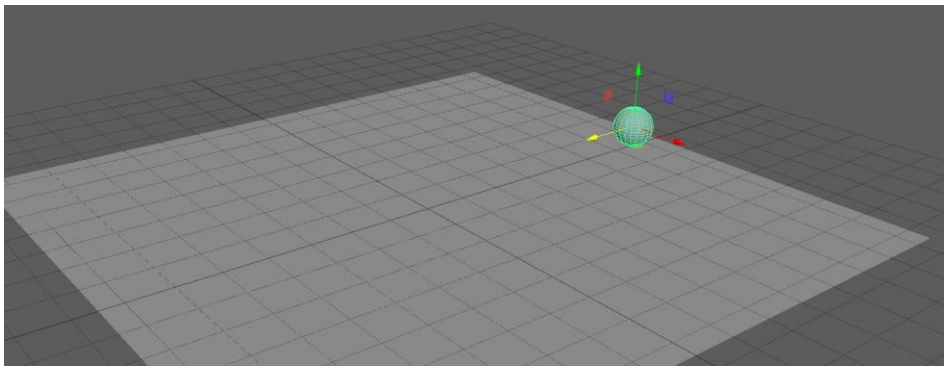
Now, let's animate the sphere.

Click on the sphere. Push the S key or go to the menu and select key->Set Key. This will set the first animation frame. Look at the playback timeline and you will see a red line.

Next, move the selected frame to 60

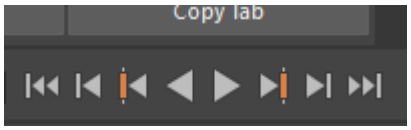


The Move the sphere to the other edge of the plane

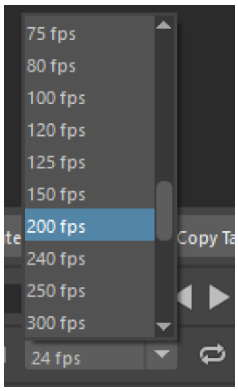


Set a new Key for this position (S or Menu key->Set Key), check the playback line and there should now be a red line in frame 60.

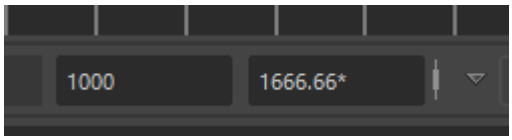
Next, push play and test the animation.



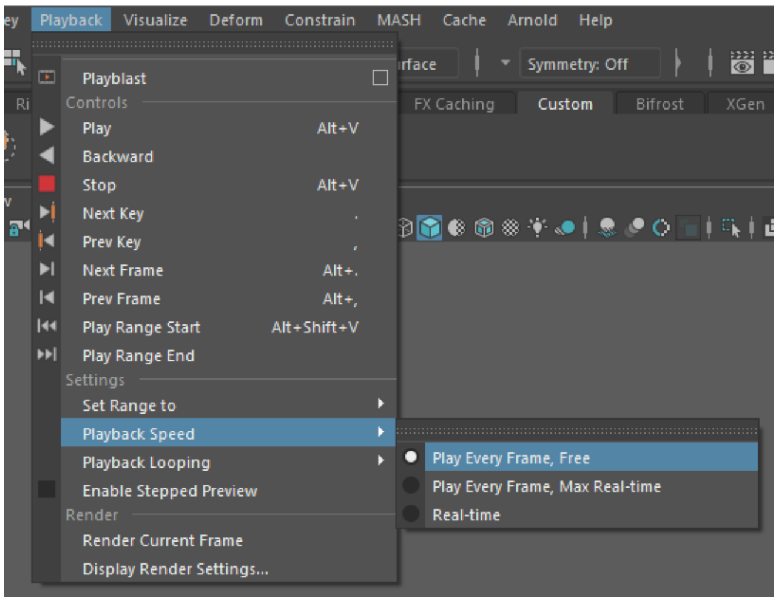
You should see the ball moving between the edges quickly. This move quickly as the base animation speed is 24 frames per second. To view the sphere moving at a non-super speed level, change the frames per second to a larger value.



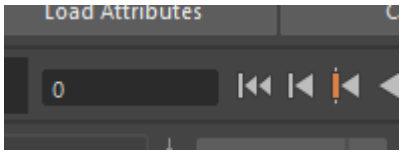
This change should allow you to view the animation occurring. Notice this change also affects the length of the playback and animation frames.

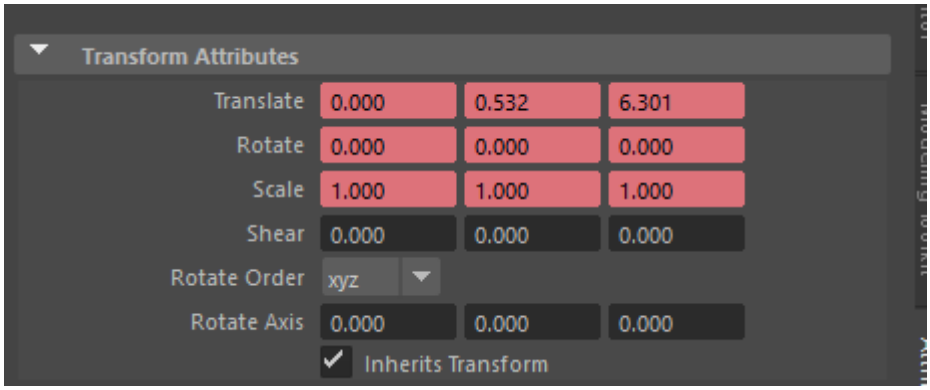


You can also change the way animation speeds occur in the playback menu by changing the playback speed.

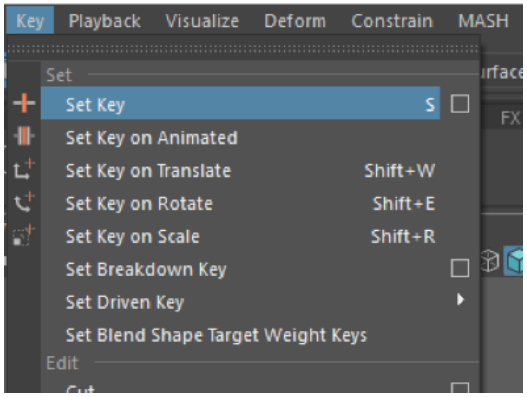


Now, we can manipulate more than just the position, if you reset the sphere back to frame 1, and then go look at the attribute editor of the sphere, you should see something like this:





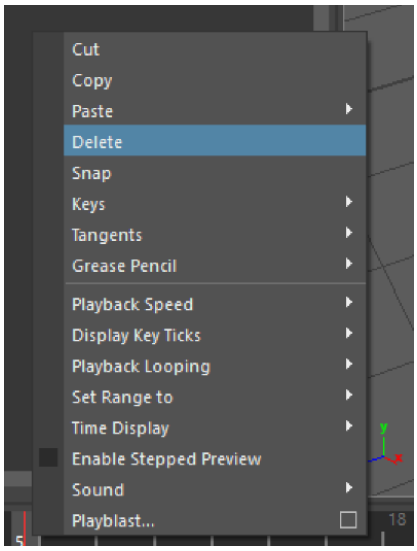
We can change the other attributes such as rotation and scale. From the Key menu, you can see the shortcut keys to this:



Knowing this, we can add some differing elements to the ball moving, we will scale it up on frame 15, and lower the scale on frame 45.

Once you have done, test it and see how it looks.

One other important aspect of key framing, is the ability to remove a key frame, this is done by selecting the frame in which there is a set key, then right click on the playback line and select delete.

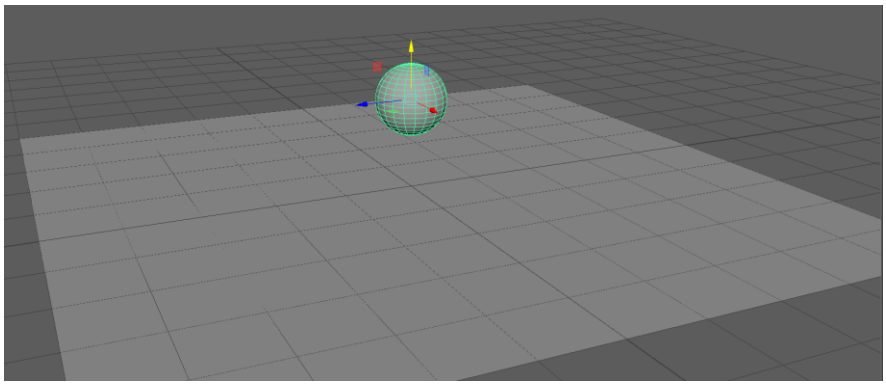


You can apply key framing to any object in Maya.

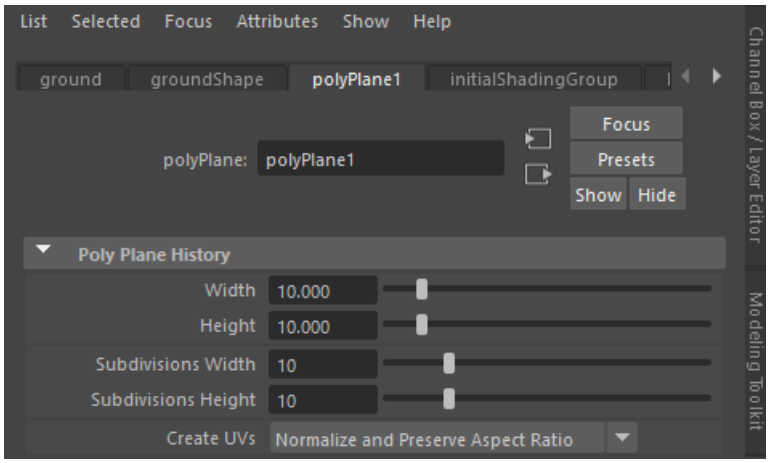
Graph Editor

The graph editor is another way of instigating animation.

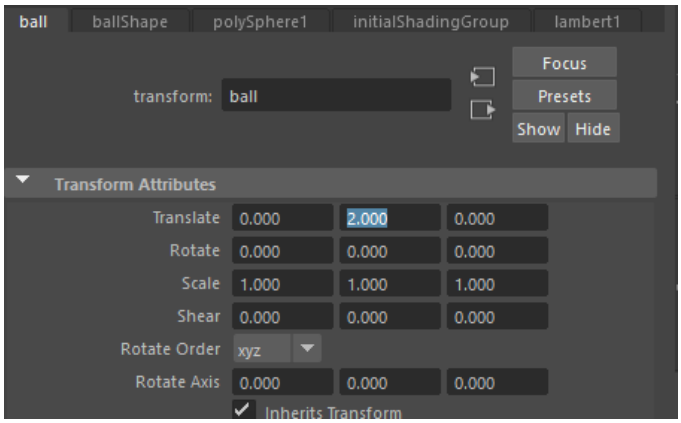
Create a new scene with a sphere (ball) and plane(ground). You should have this in the perspective view.



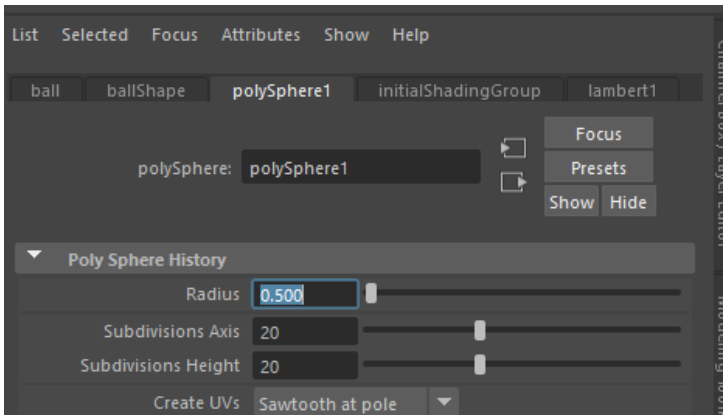
With the plane, increase the height and width to 10/10



Start with the ball translate options as such 0/2/0:



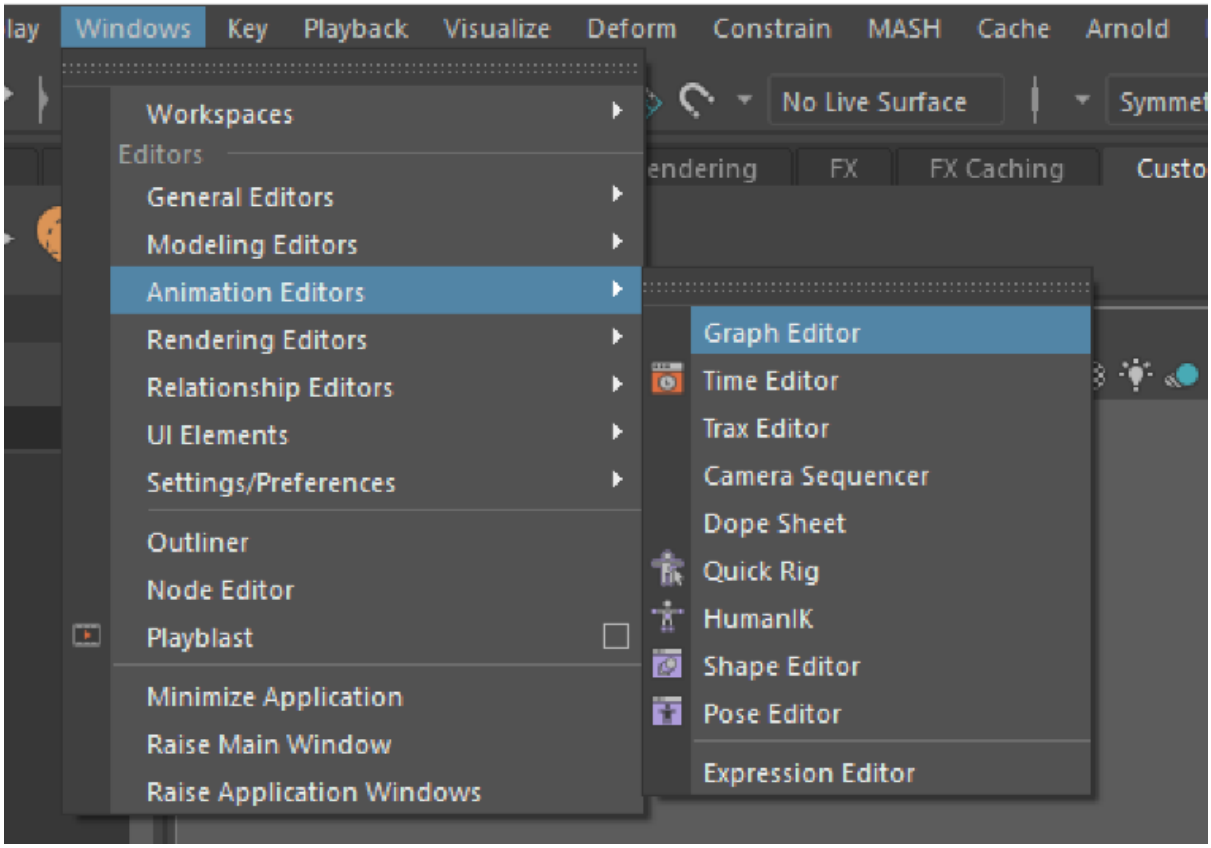
Also drop the radius (0.5) as well



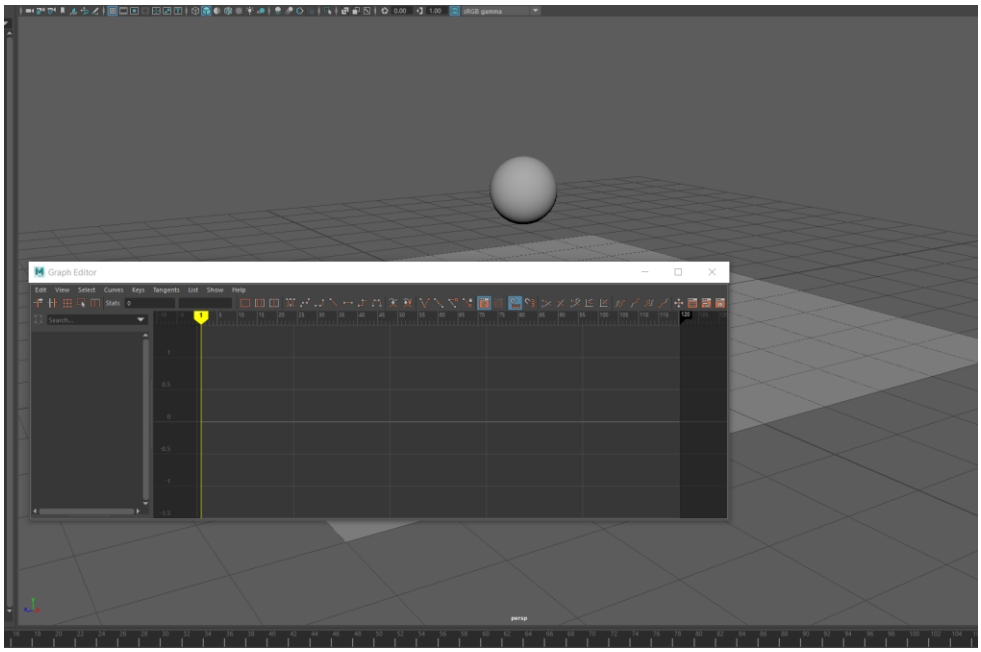
This will give us a decent sized ball on our plane.

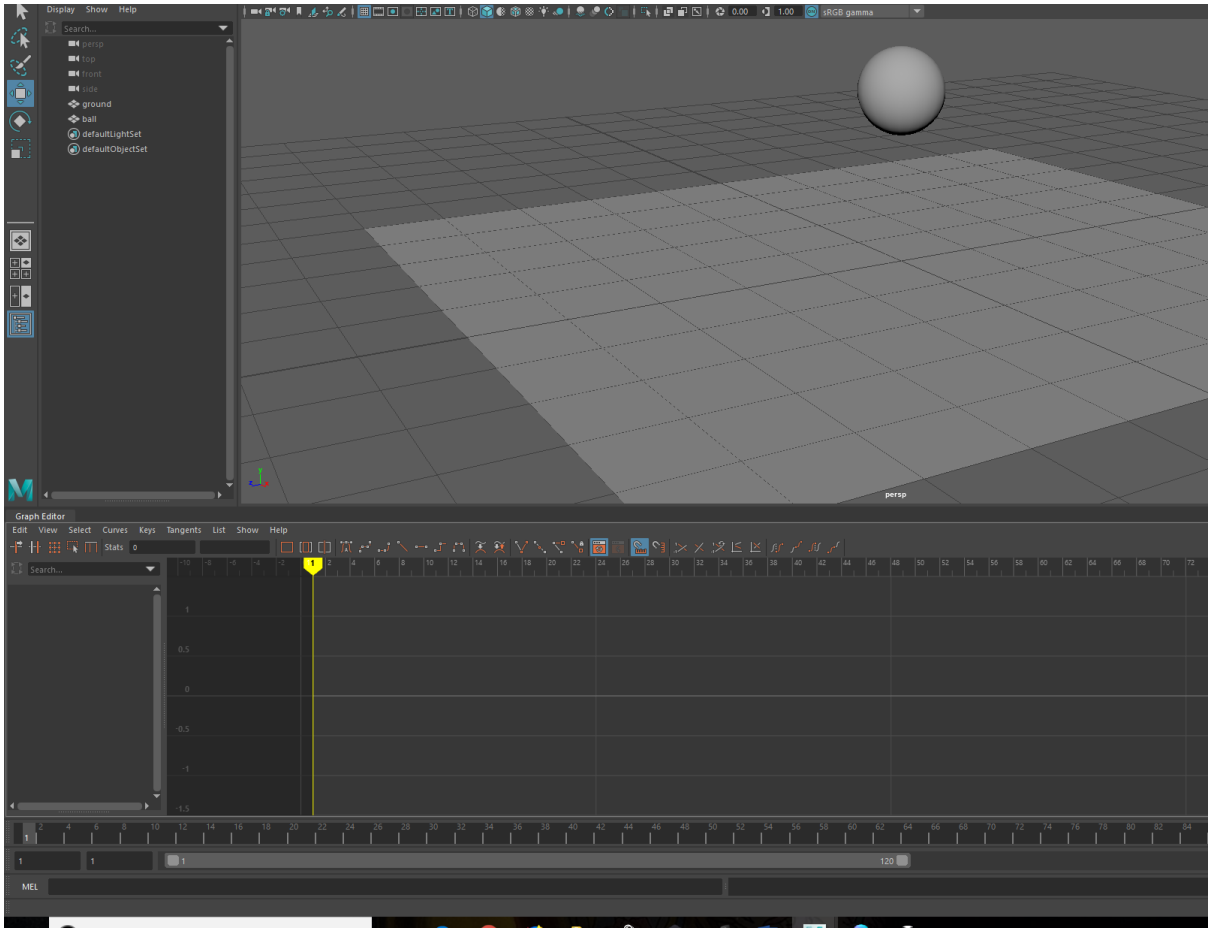
Now we open the Graph editor. The graph editor allows us to use key frames, but we can be more dynamic with the easing and transition of the animation.

Windows->Animation Editors->Graph Editor



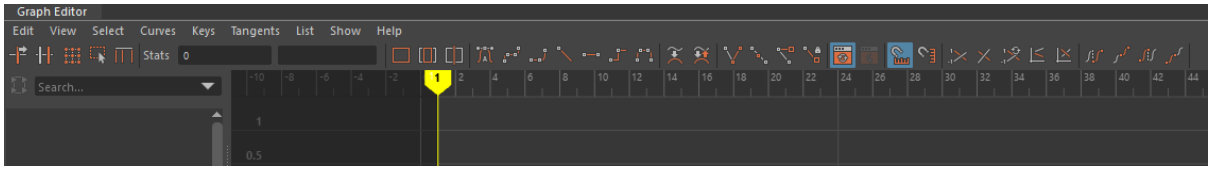
If the graph editor appears above the window, you can drag it to the timeline to snap it into the frame, then expand the window to see the editor.



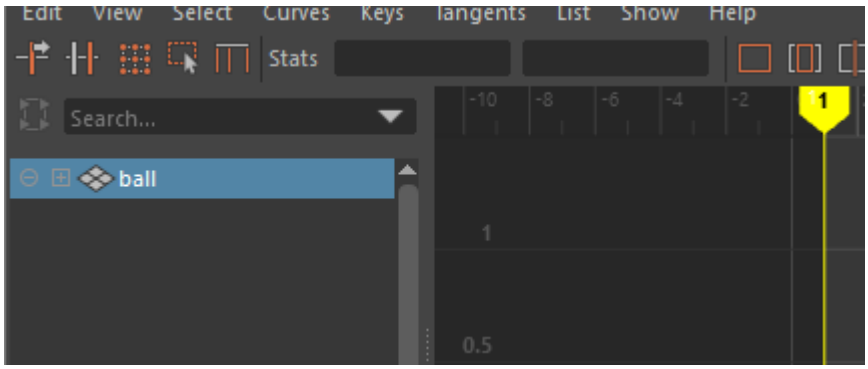


By having this window snapped into place, it allows for it to be out of the way and fully accessible.

The graph editor has its own menu system which can be used to assist with animating objects on the screen.

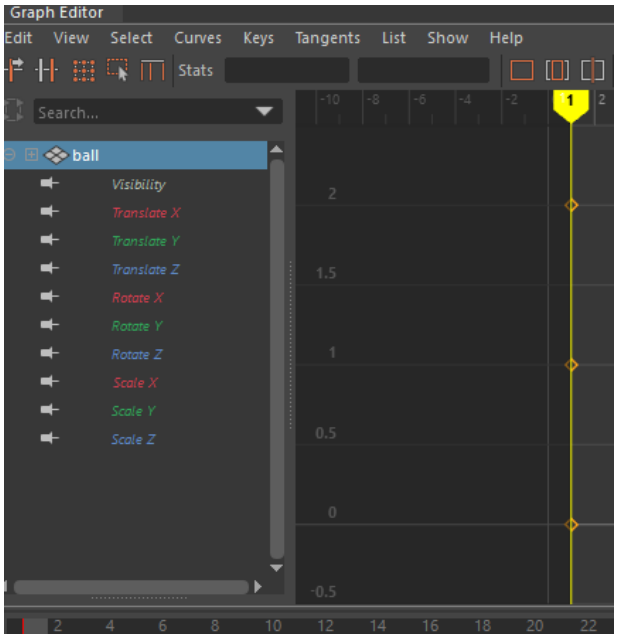


Click on the ball, you should see the following in the graph editor



This is the default position, as we have no animation set at the moment. So from this point, as the ball is in the starting position to drop, we will set a default key here, this is done by hitting the S key or using menu ->Key->Set Key.

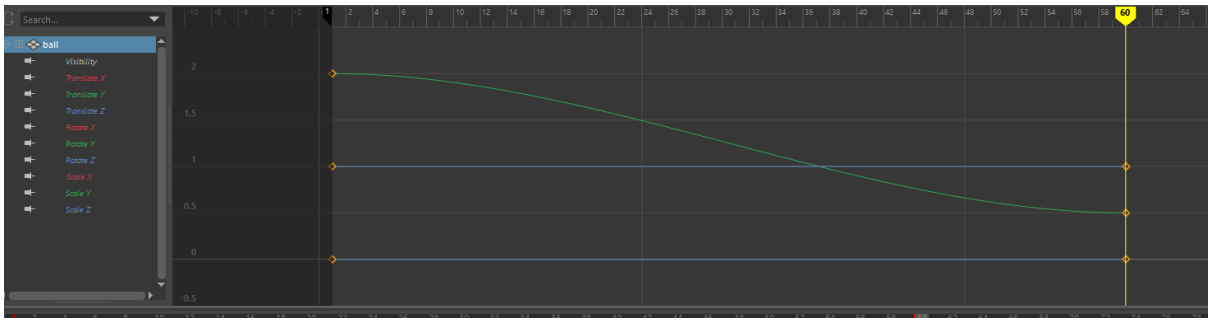
Once you have done this, the graph editor recognises that there is a set of elements of the ball that can be manipulated.



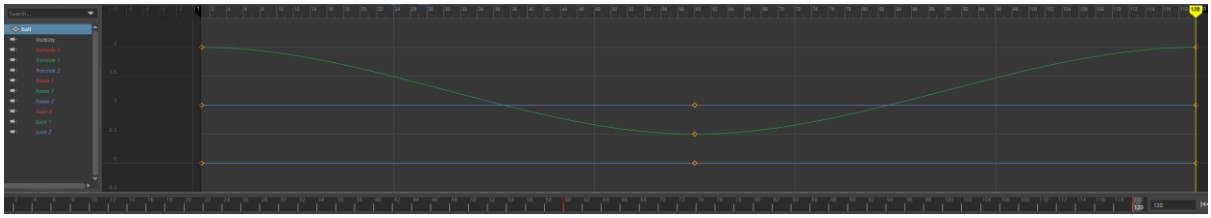
Now that we have a default position, let's move the frame to 60, and position the ball on the ground.

Remember, once you have done all of the movement elements, you need to set another key for the frame.

You should end up with the following in the graph editor.



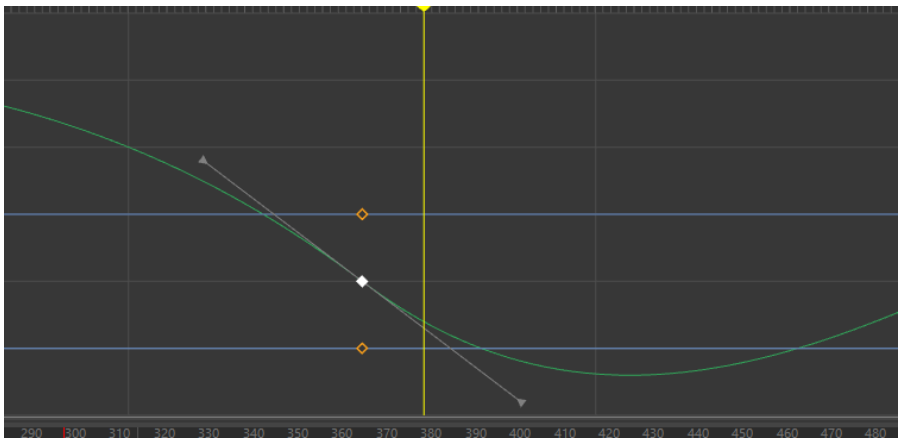
Notice how the graph editor has now shown us the ease in of the drop, move the frame to 120 and the ball back to a Y of 2. Set another frame.



So, we are now shown a nice curve from 2 to 0.5 and back to 2 in 120 frames. Push play to see the animation happening. As before the animation is moving really fast, now if you swap the fps from 24 to 120fps and push play, you should see a reasonably smooth bounce for the ball.

The main advantage to the graph editor is that we can grab and manipulate the frames of the curve.

Grab the bottom bounce point, it's on the green line, frame 300. When you click on the point you will get handles off it in which you can manipulate the way the curve works.



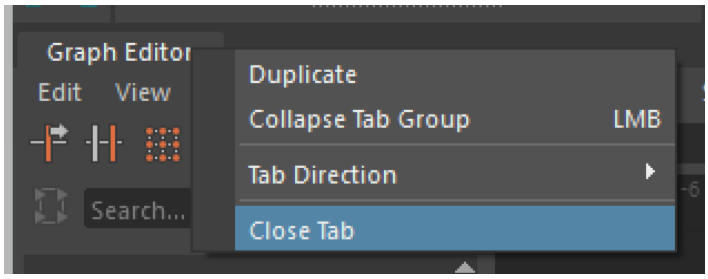
This form of curvature drops the ball fast and bounces high slowly, till it reaches the apex and drops again.

As you can see, we have just played with the translate of the object, you can manipulate the rotation and scale of the object as well.

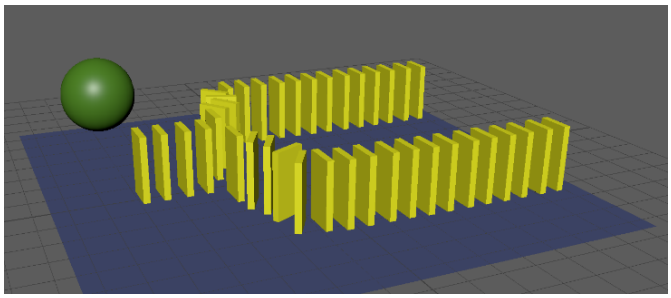
This is normal keyframing for an object, but you are given far more control over the way the transition occurs.

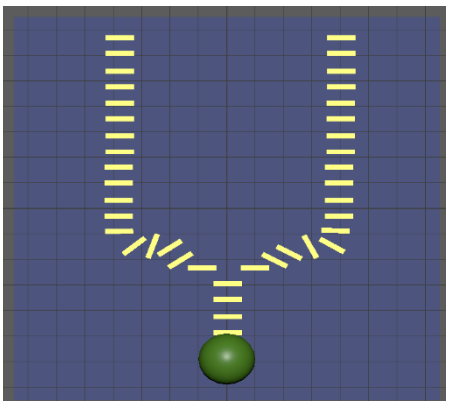
Physics Animation

Physics animation is where we can apply forces to the objects on the scene. To start with, we will start a new project. Then close the graph tab, right click and close tab.

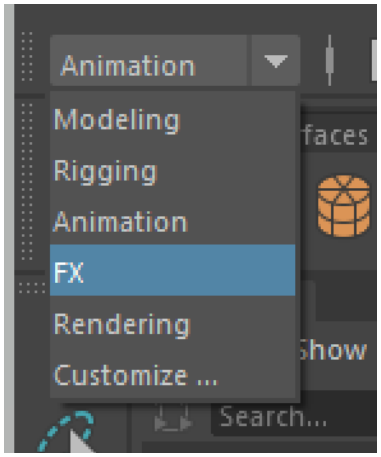


Create the following use a plane, sphere and modified cubes





The goal of this animation is to get Maya to do all the work for us, so now that we have built the environment we need to apply some effects to the objects that we have. To start with, need to swap the menu system of Maya over. For this we go to the FX menu selection

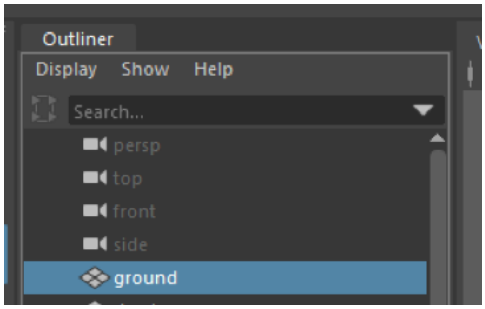


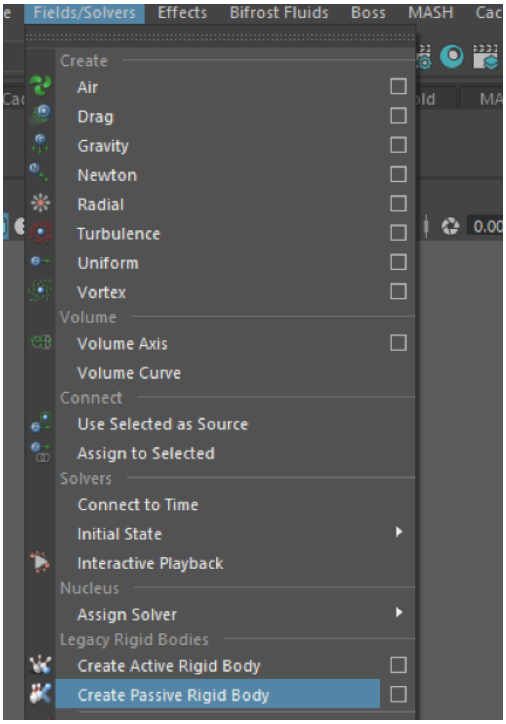
This gives us access to the fields/Solvers menu system in which we need to access 3 elements;

- Create Active Rigidbody
- Create Passive Rigidbody
- Gravity.

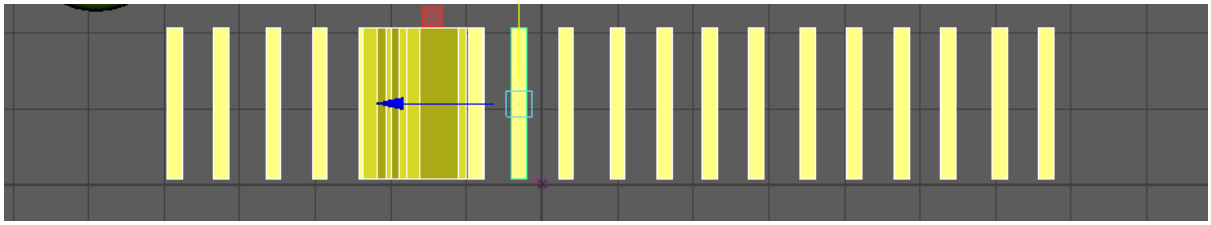
The elements will allow things to be manipulated by the system.

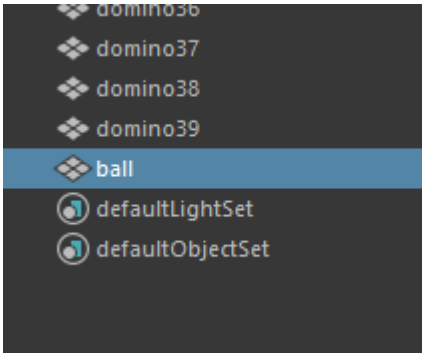
Select the ground (plane) and apply a passive rigid body to it.

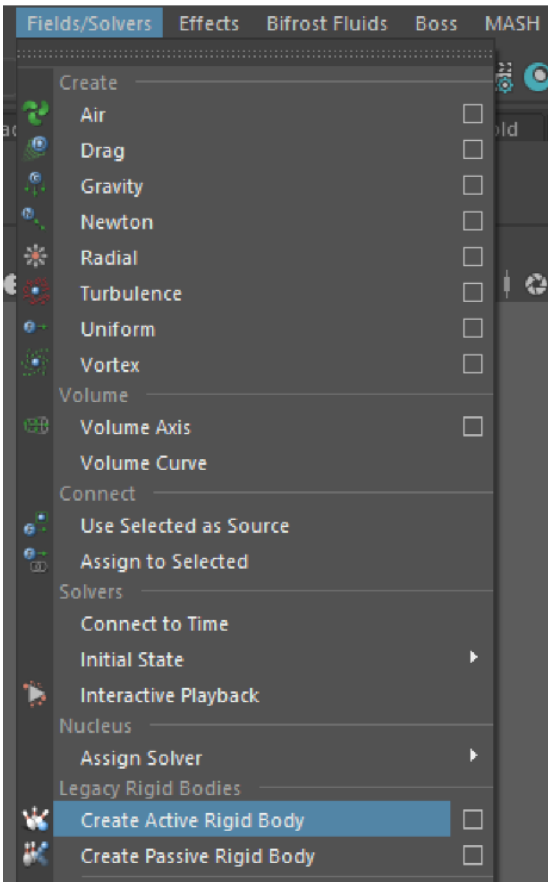




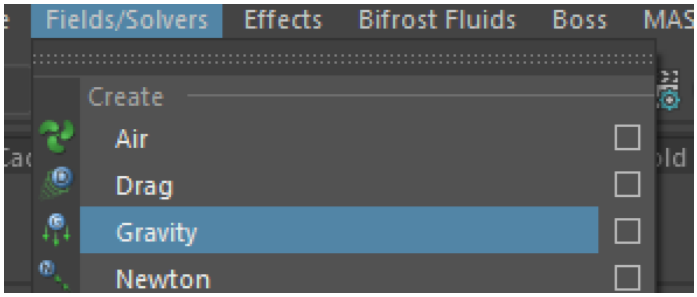
In the hierarchy, highlight all the dominos and group them. Then individually, select the ball and apply an Active rigidbody. Repeat for the dominos. Easiest method for apply to the dominos is to go to the side view and select them in all.



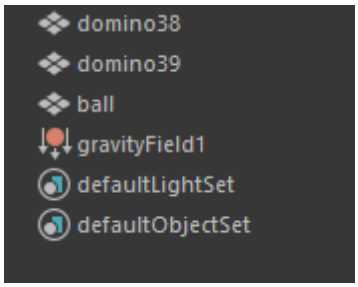




From here, apply gravity to the ball and dominos. In this manner, there will be a force applied to these objects. Select both elements from the side view, then create the gravity.



If you notice, we now have a new gravity object in the outliner and the inspector gives us properties that can be manipulated to affect the way gravity works.

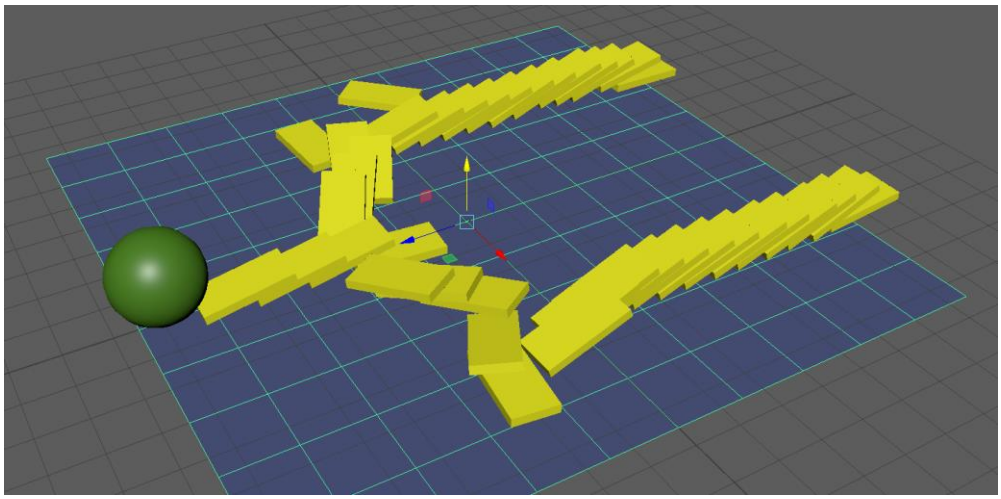


As you can see, the base magnitude of gravity is earth normal, 9.8. The direction is -1 on the Y axis. As you can gather, if we play with

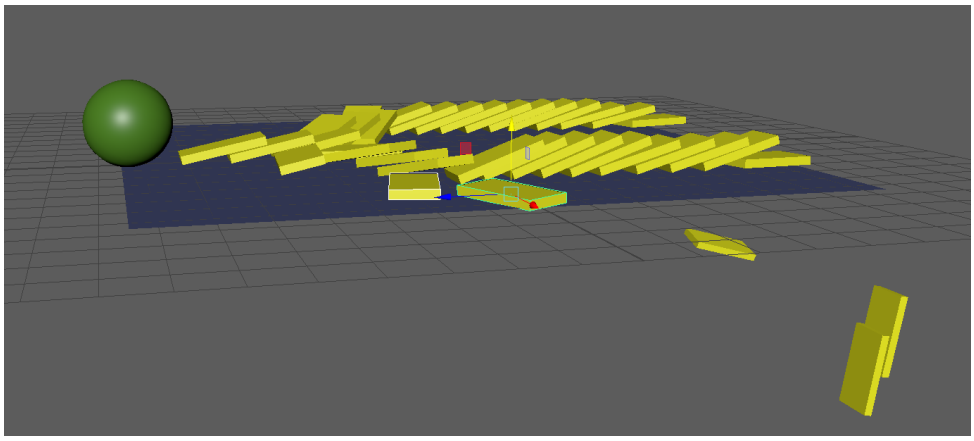
these attributes the effects of gravity will change and the animation that is created will differ as well.

Now that we have a base, run the animation, the same way you would with a key frame.

You should see the following:



If you want to see it all happen slower, drop the level of gravity from 9.8 to 1, but this introduces an issue in that you need to increase the frames for the animation. If you boost the playback frames to 500, it should be viewable, and you will some of the dominos fall of the plane.



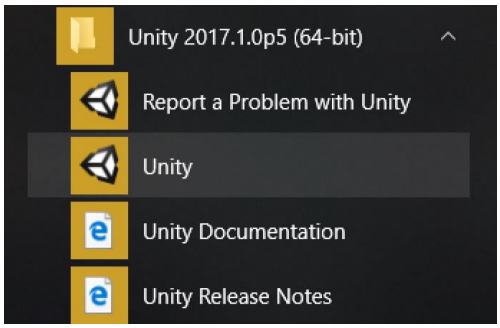
Unity

Goals:

- Animating in Maya and moving to unity.
- Implementing Animation of characters in Unity

Load up Maya

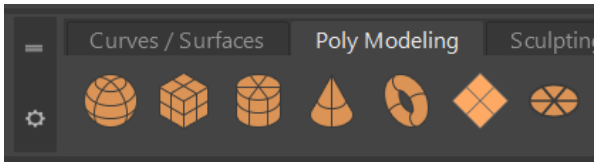
Load up unity

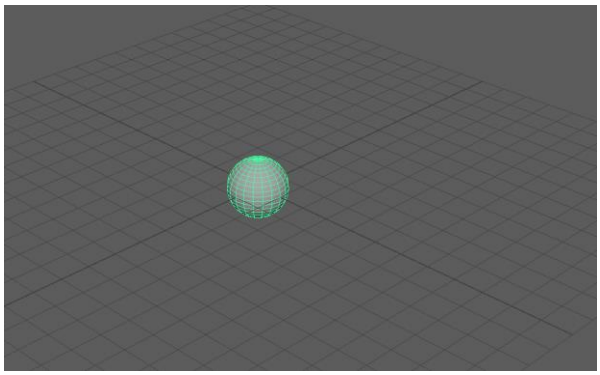


Build Object: Basic Maya Animation to Unity

Aim: build a very simple key frame animation in maya, move it to unity and loop the animation.

To start with, let's create a sphere



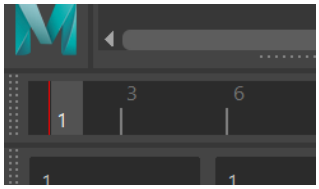


Now we are going to set a bunch of keyframes and move the sphere around the grid area.

As you saw earlier, you can either push S to set a keyframe or in the animation menu, you can go key→Set Key.

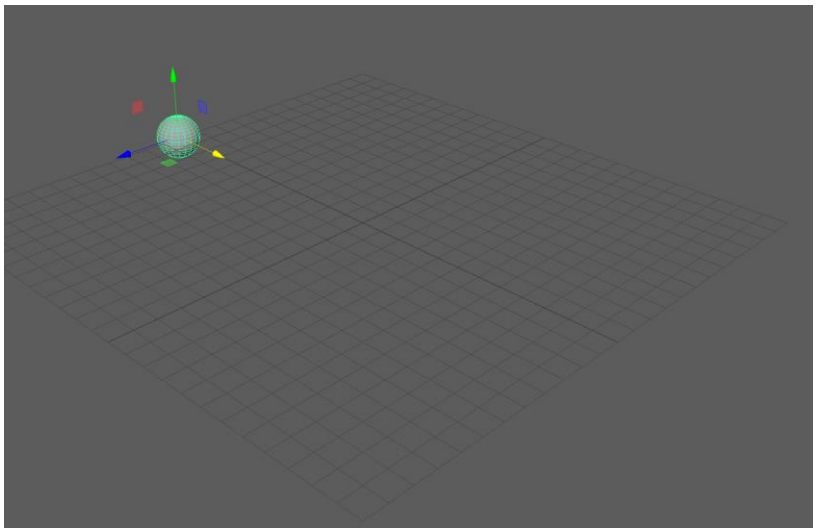
To begin, we'll set a key in its default position.

Remember to check the timeline for the red bar, which indicates the keyframe has been set.

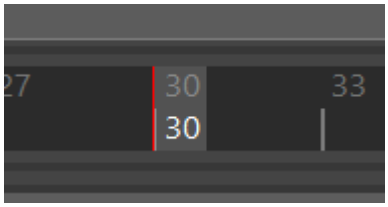


Now, select about keyframe 30

And drag the sphere to the edge of the grid.

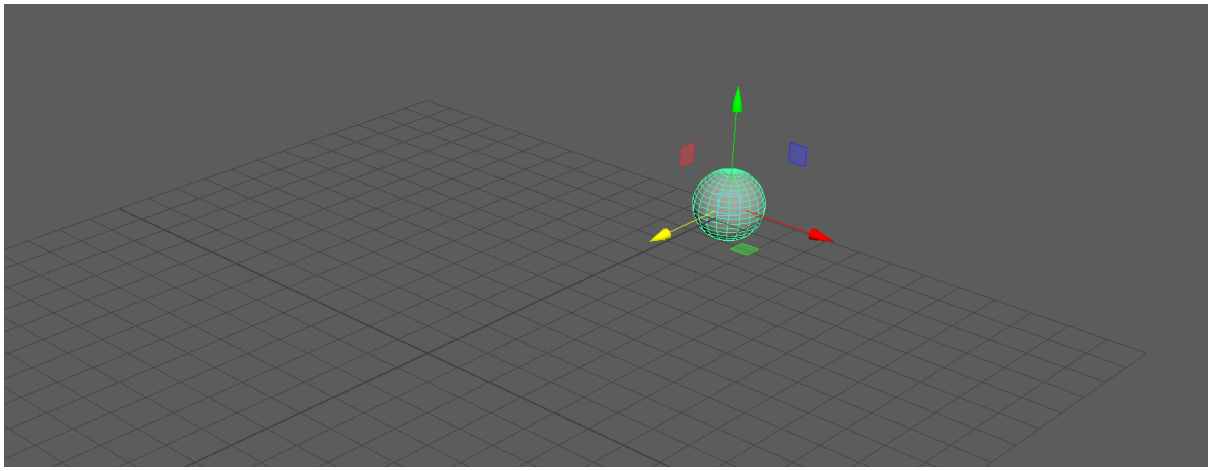


Push S and confirm the setting of the key frame.

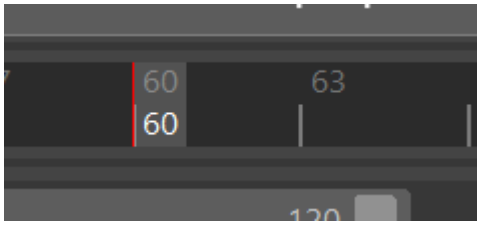


Move the keyframe to 60

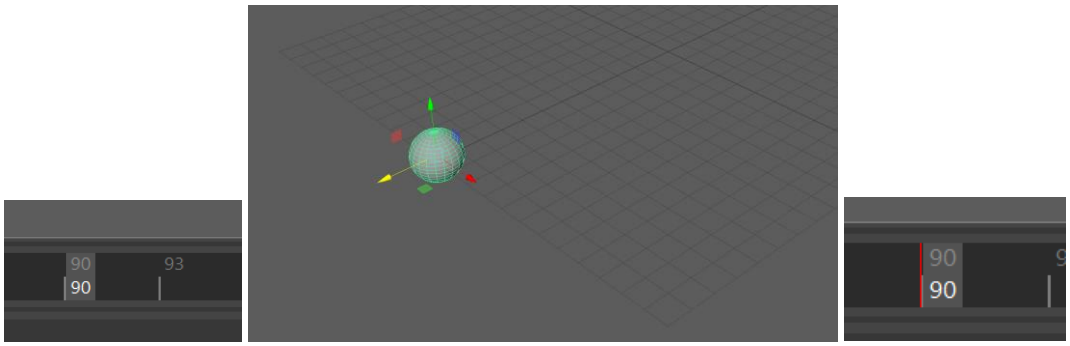
Then move the sphere to the other edge of the grid.



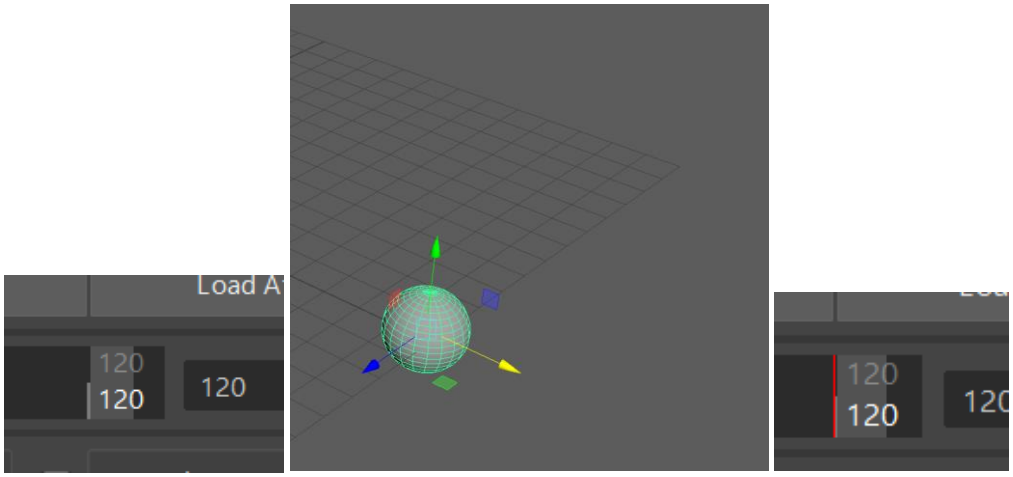
Push S and set the keyframe



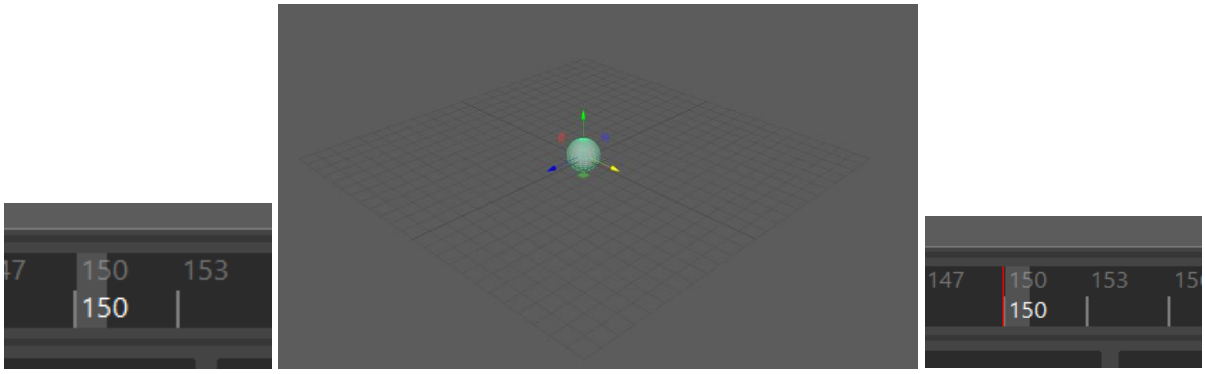
Select 90 on the timeline, move the sphere to the opposite edge and set the keyframe



Then select 120 on the keyframe, move the sphere to the final edge and set the keyframe

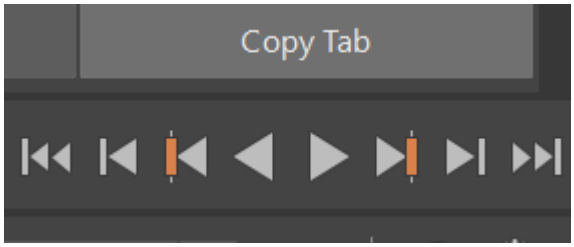


Lastly, move the keyframe to 150, move the sphere back to the center and set the keyframe

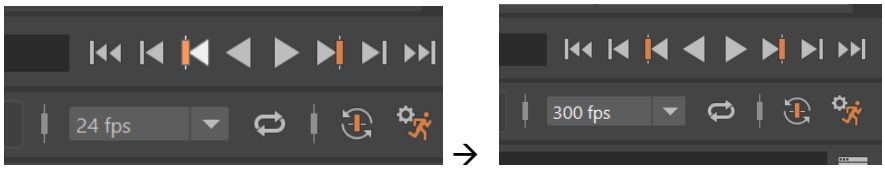


So, this is our very simple animation, just moving the sphere around the grid.

Push play and confirm that you have an animation.

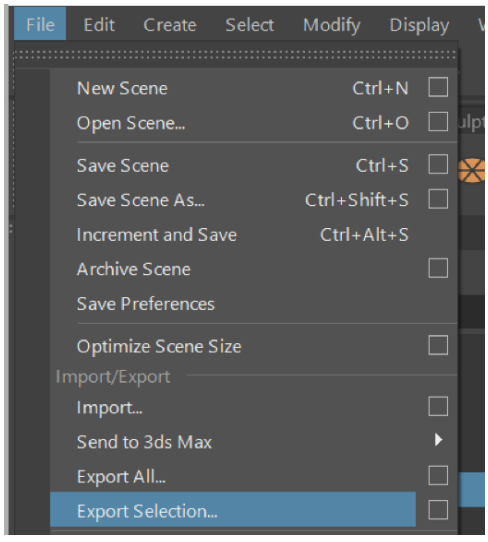


If it's moving too fast, change the FPS from 24 to 300 fps



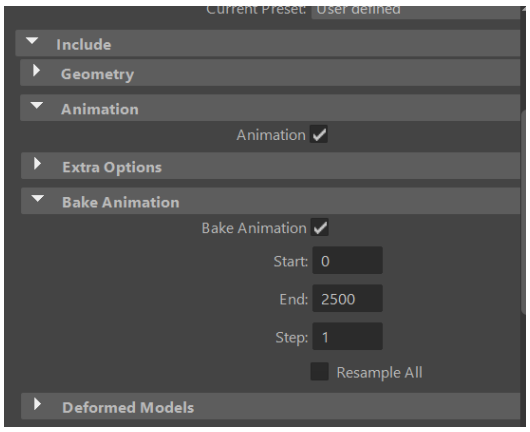
Now export out for our unity project.

With the sphere selected go, File→ Export Selection

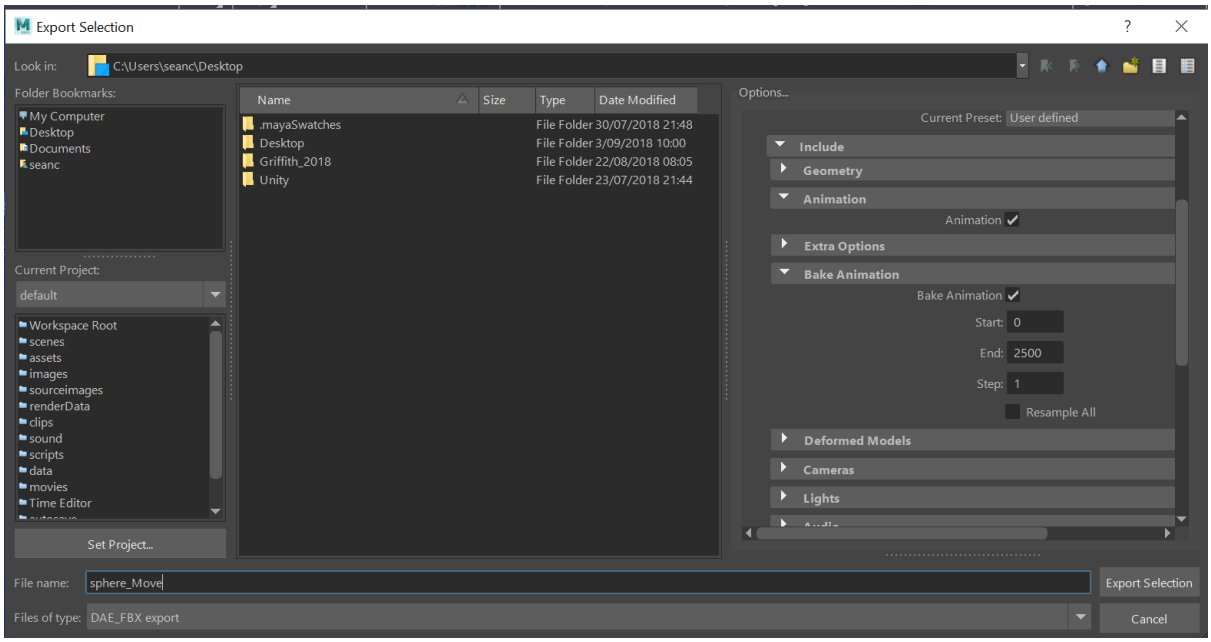


This will bring up the export pop up. From here, change the file type to DAE_FBX export

This modifies the options that are available to the object. Down the menu section ensure that the animation export section is checked and bake animation is checked as well.

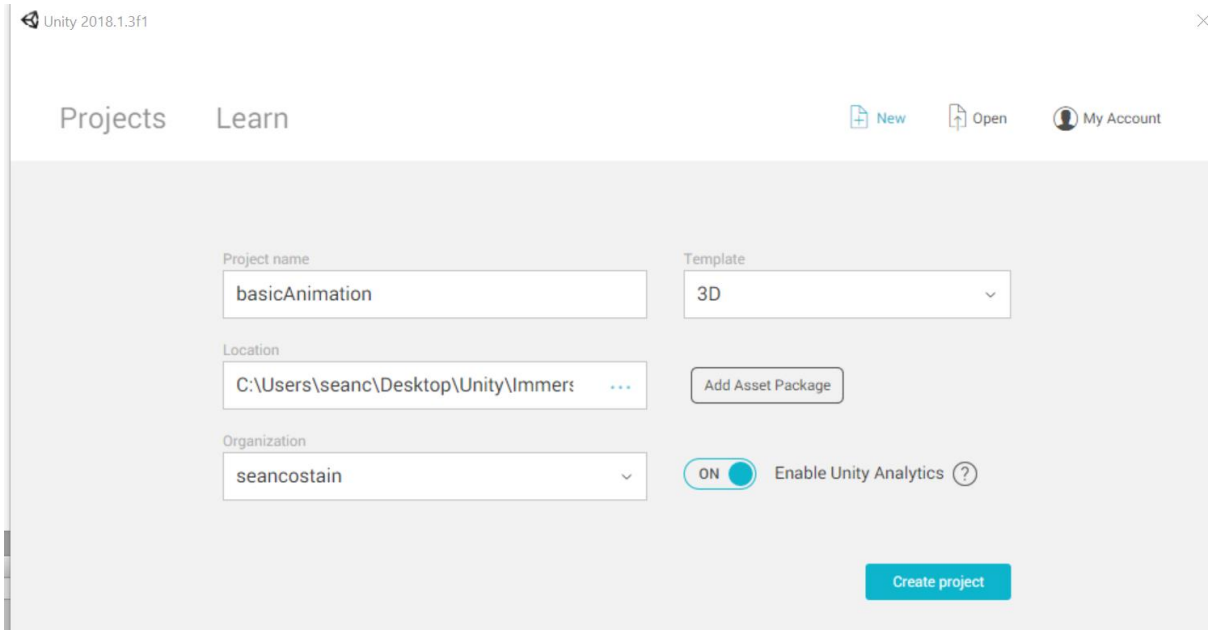


Name your file and save it to a location you can find, in my case I saved it to the desktop.

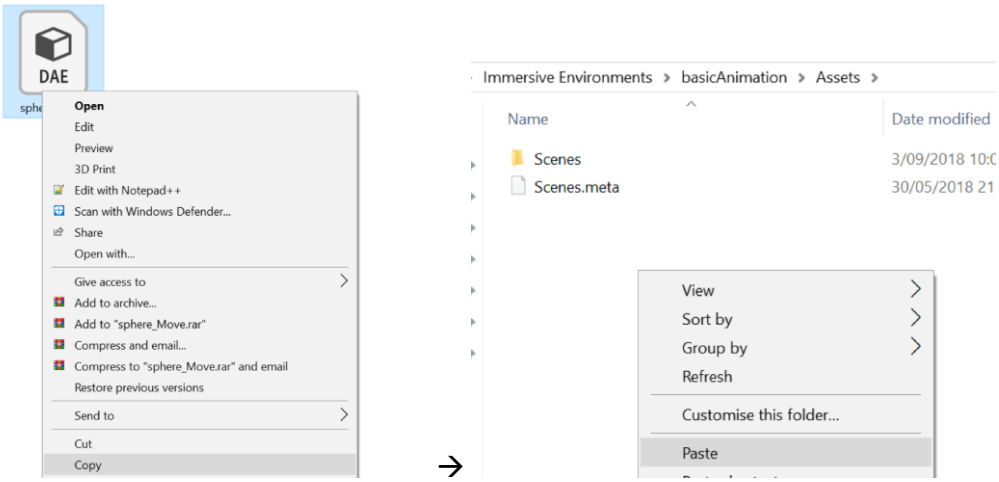


Export the selection, then go to unity.

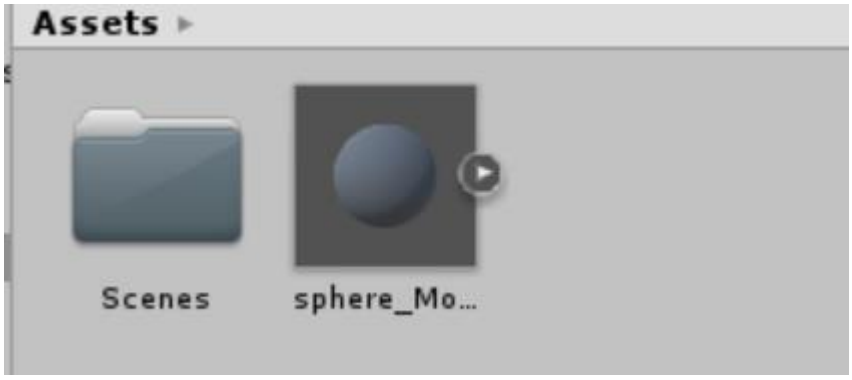
Create your new Project



From here, go to the file explorer and copy the maya export into the assets fodler in unity.

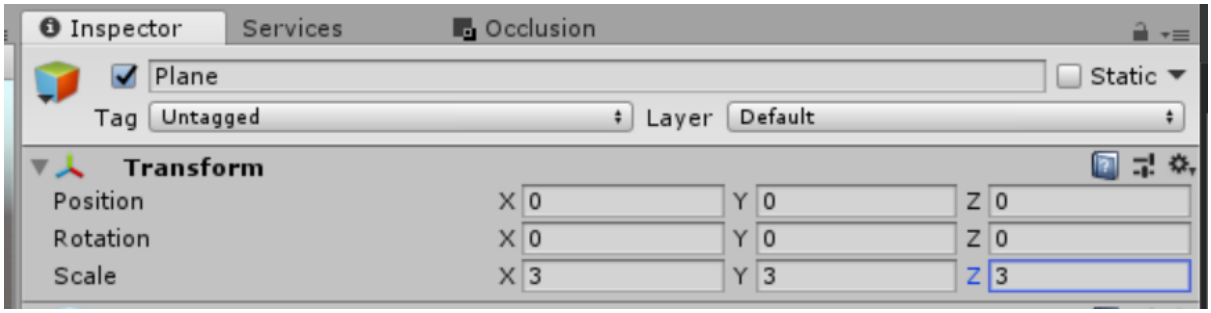


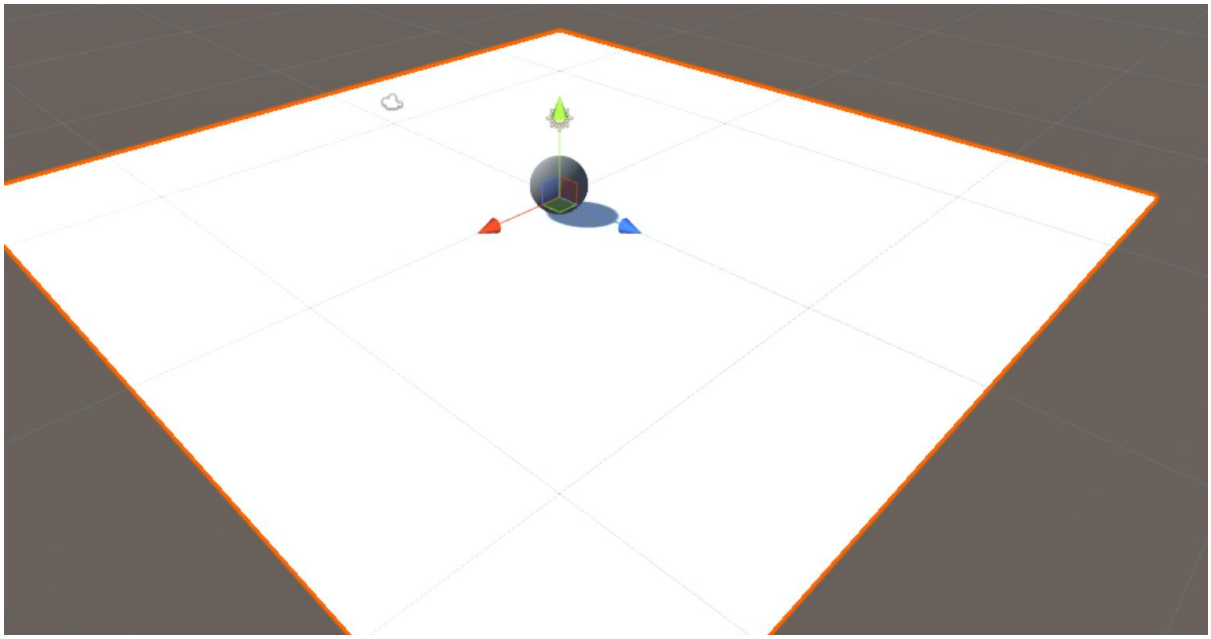
Switch back to unity, the assets folder should look like the following:



Next, place a plane in the scene and drag the sphere onto the plane.

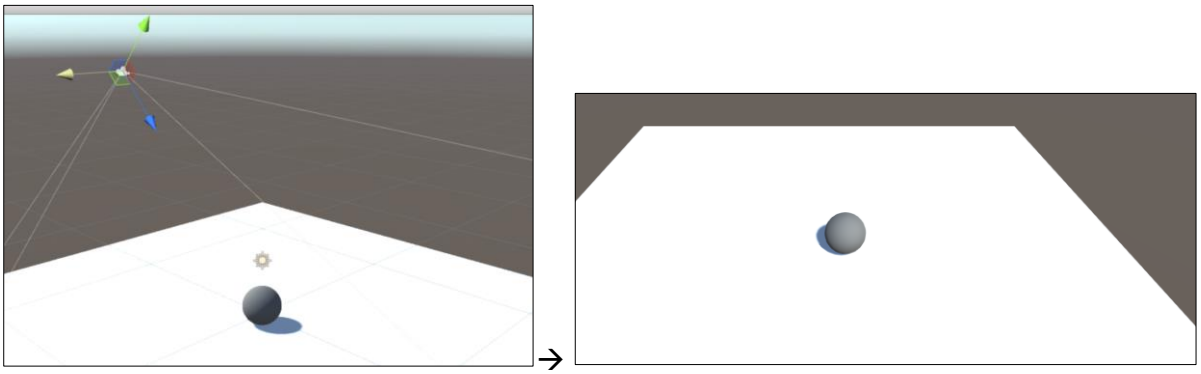
There is a distinct difference in the size, as such, scale up the plane so it looks like the following



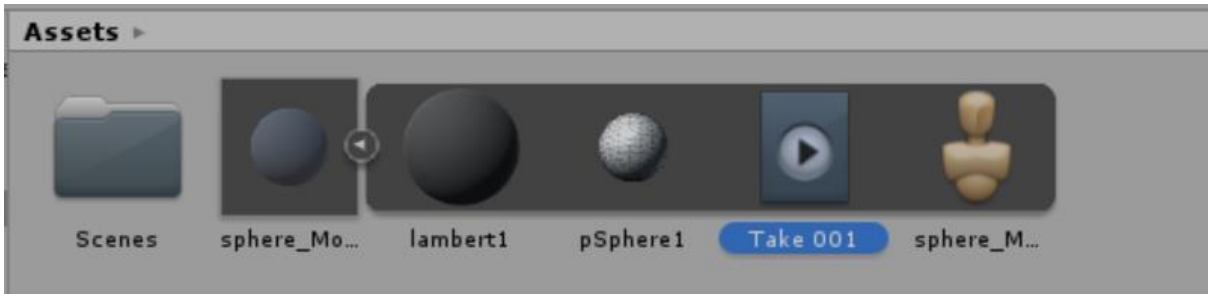


Now that we have the basics setup, if you push play and test, you will see that the sphere doesn't move at all. As such, we now must add an animator controller to enable the animation to be accessible.

Also, now is a good time to move the camera to where you will be able to view the entire scene.

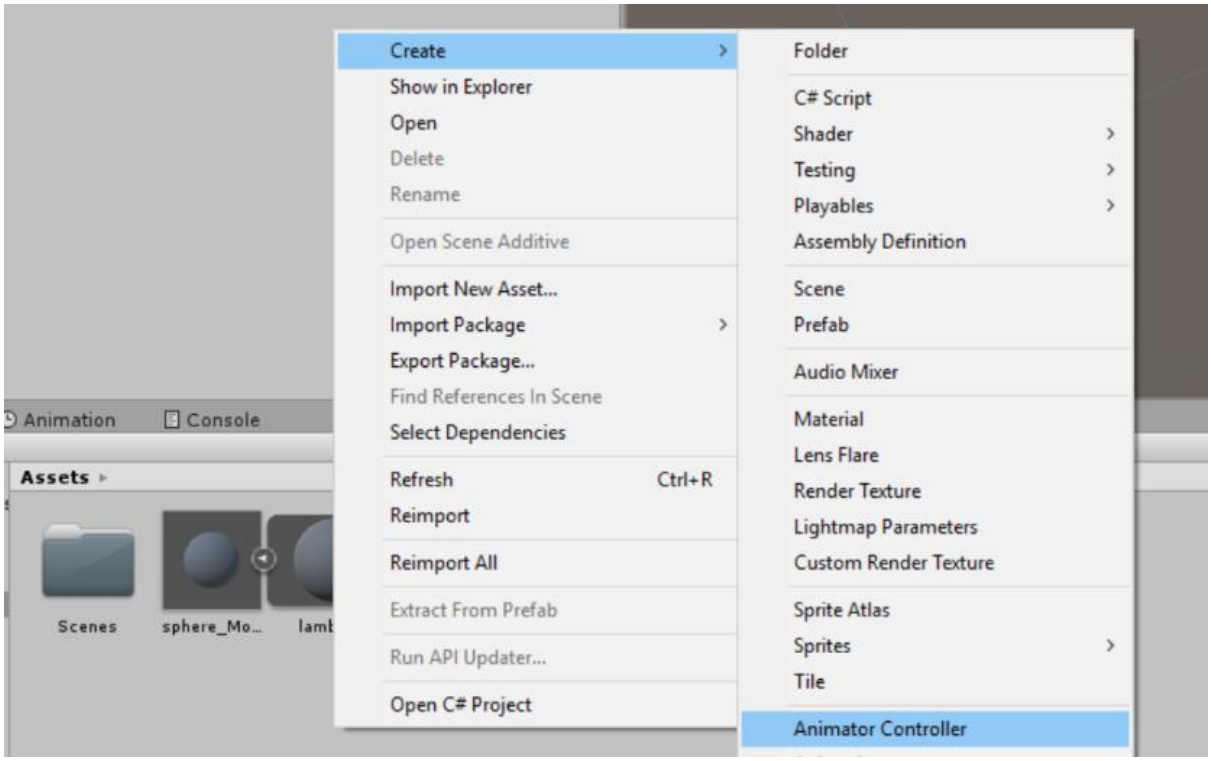


Back into the assets view, if you expand the sphere, you should see the following.



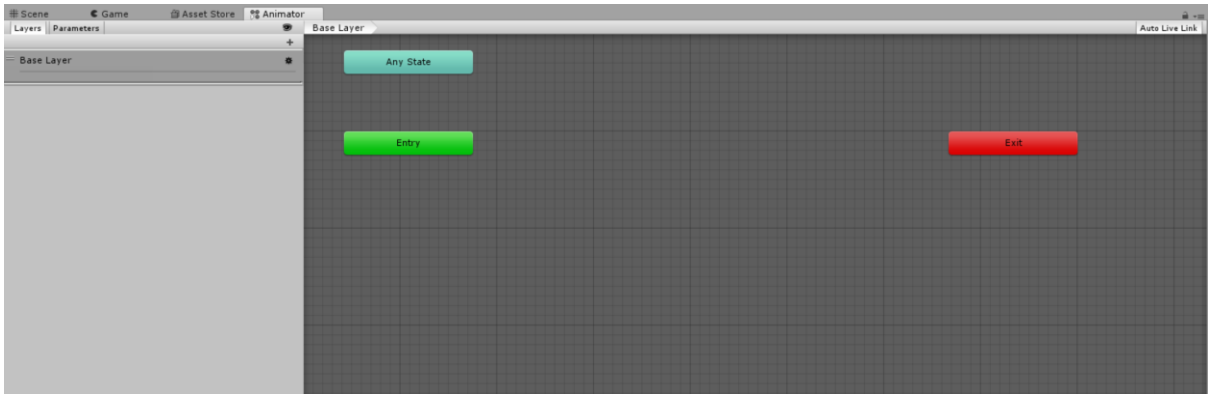
If this isn't there, then go check how you exported the object.

Next, we need to add the animator controller. This can be done by right clicking in the assets view and then selecting Animation Controller

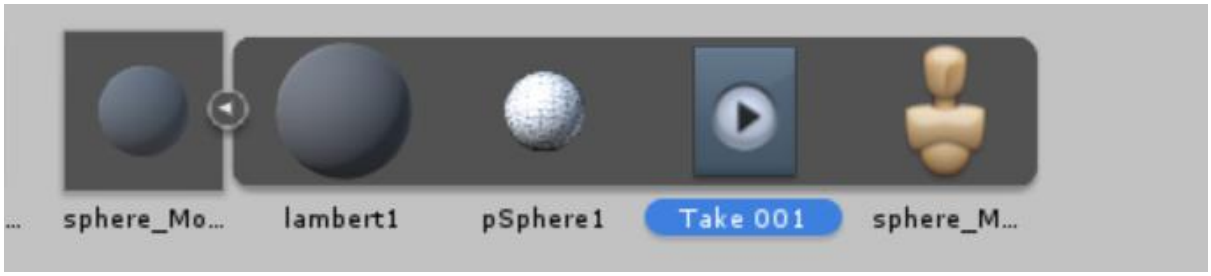


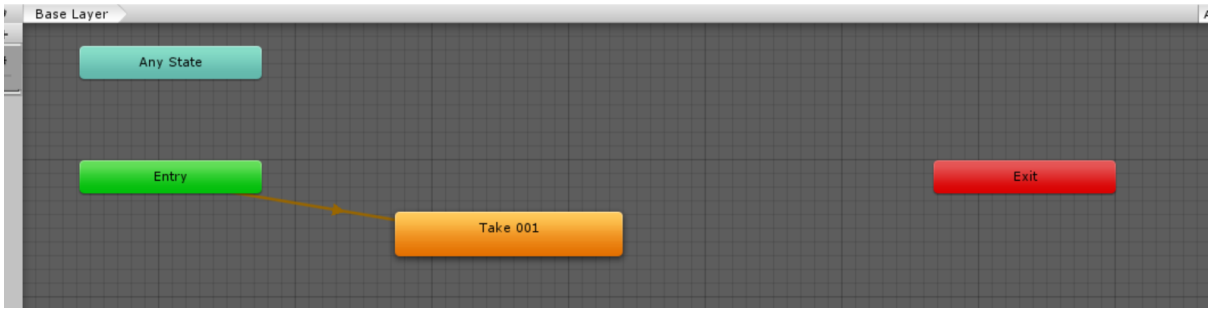
Name it something you recognise, in this case I called it sphere_Animation.

Double click the animation controller. This will open up the Animator window.



From here, drag the animation from the sphere onto the animator window



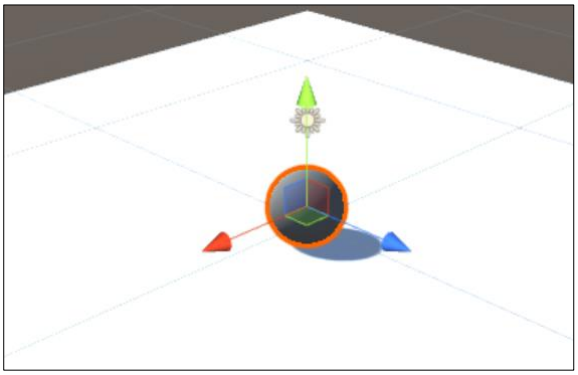


Notice how 'Take 001'is now automatically linked to the Entry button and has turned orange. If an animation turns orange, then it is the default animation. In this manner you can add additional animations to the one controller and link them up, for example a player character could have an idle as it's default animation, a walk, run, jump and so forth each triggered by a key press. In this case, we'll just do the one animation.

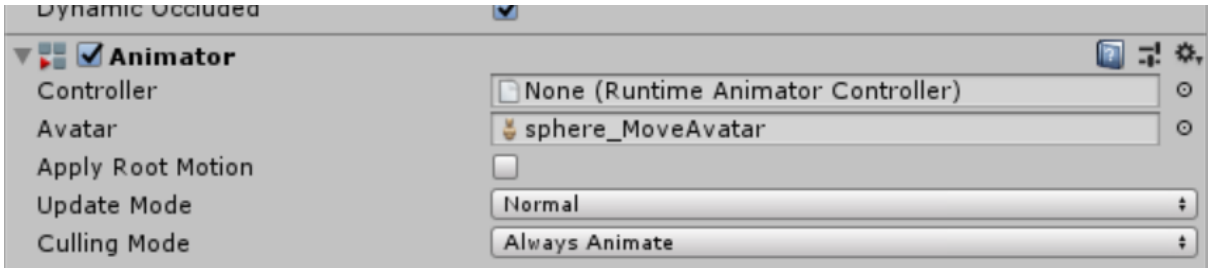
Now, click on play and see what happens.

Yep, nothing.

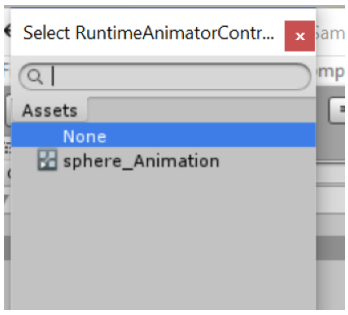
Go back to the scene and click on the sphere.



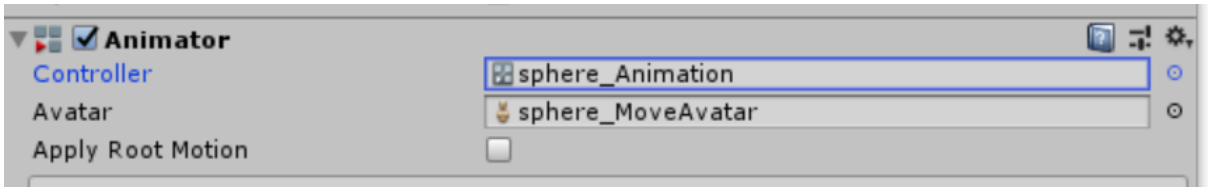
If you examine the Inspector you will see that the Animator on the sphere doesn't have a controller, we need to attach the sphere_Animation to this object.



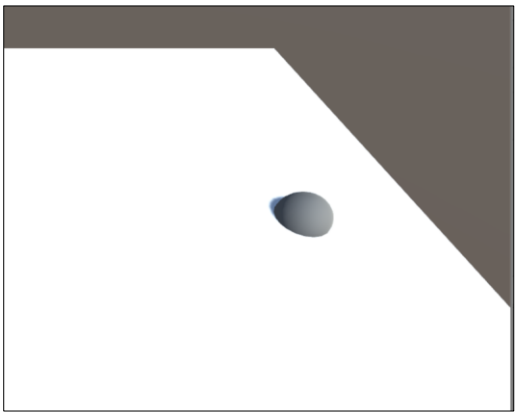
Click on the ‘target’ icon next to the None(Runtime Animator Controller), this will open the following pop-up box



Double click on sphere_Animation, this should then give you the following

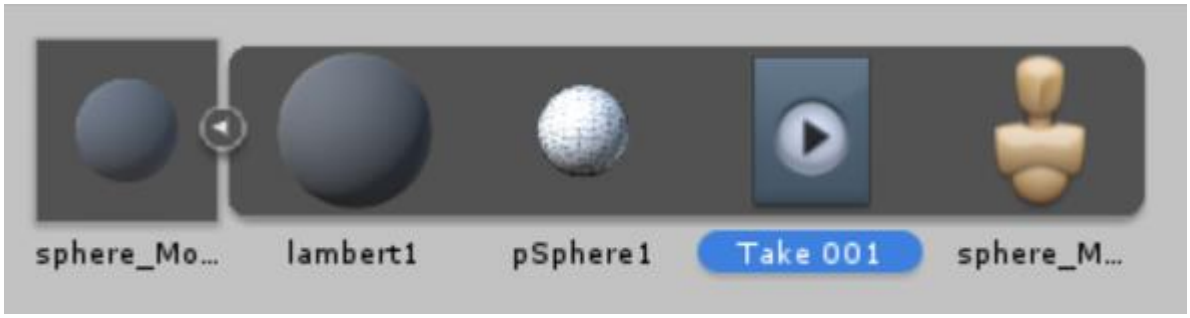


Now, push play and test.

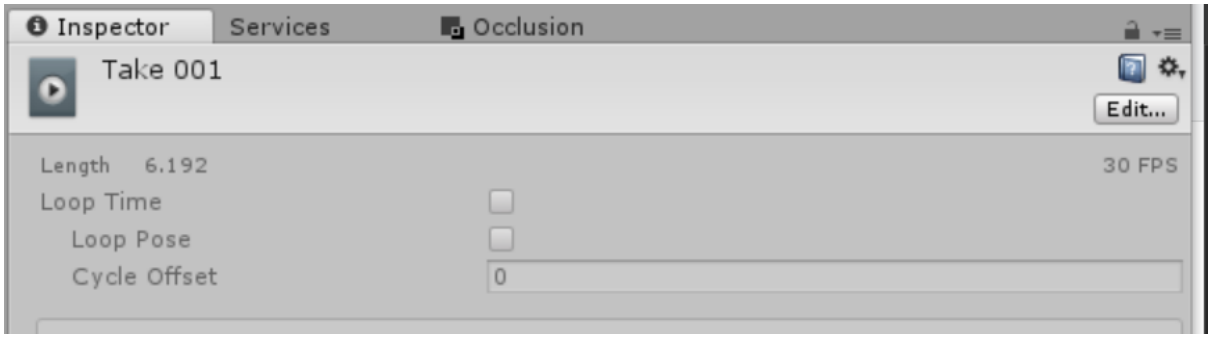


You should have the sphere moving around the plane, but it only does this once. If we want to have an animation loop, for example and animation in the background to give more life to the environment, we need to loop the animation.

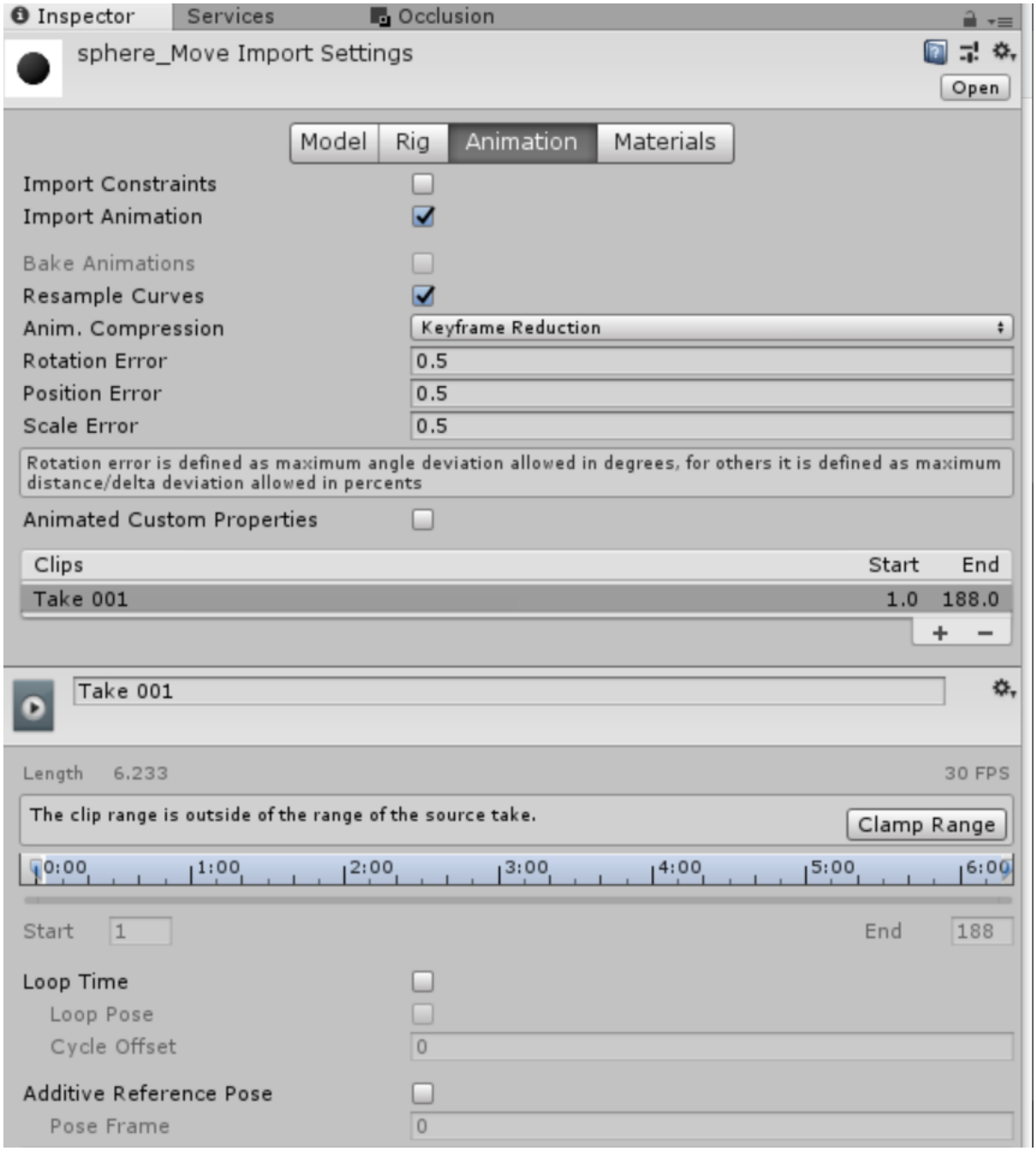
To do this, expand the sphere_move object and select the Take 001 animation.



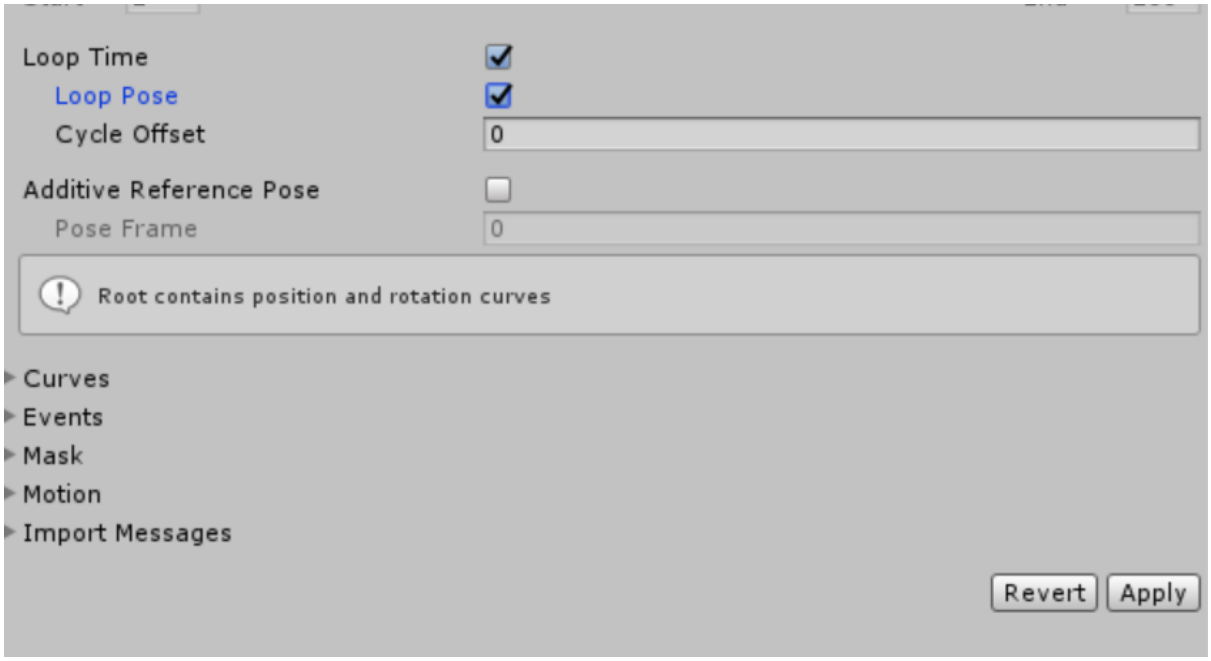
If you examine the inspector, you will see an edit button. Click this.



Once you have clicked edit, you will be presented with a lot more information, in the animation section, look for Loop Time.



Check loop time and loop pose, once done click on apply.



Now test and run, if done correctly, your object will loop through the animation continuously.

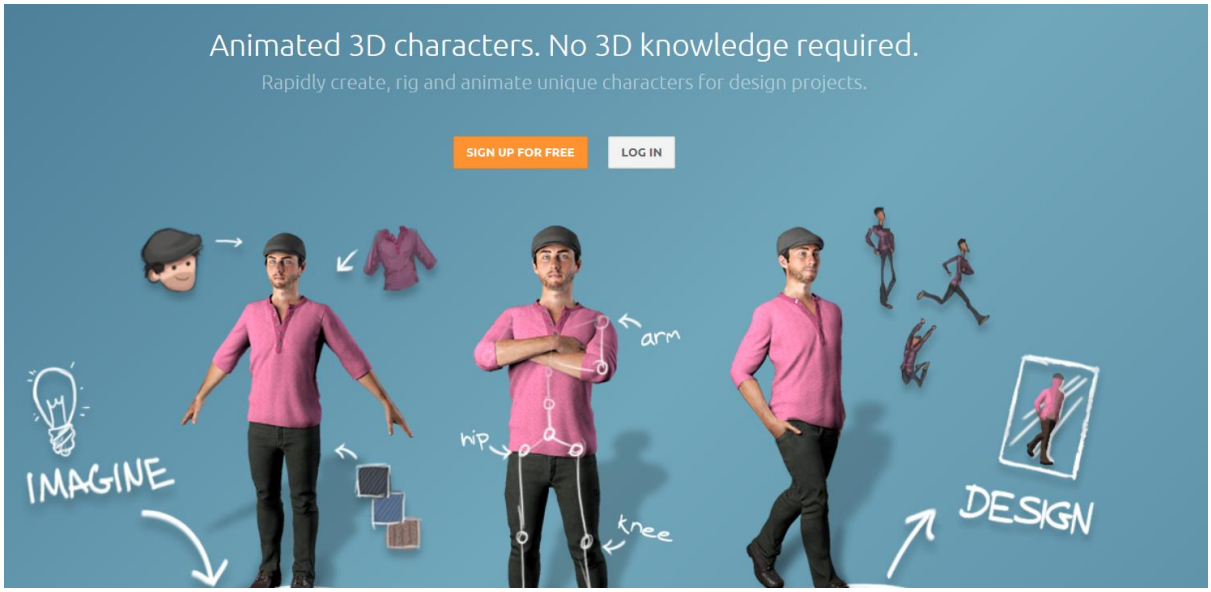
Build Object: Unity and Mixamo

Aim: Use Mixamo to get a character and animations into a unity scene.

To start with, go to the mixamo website.

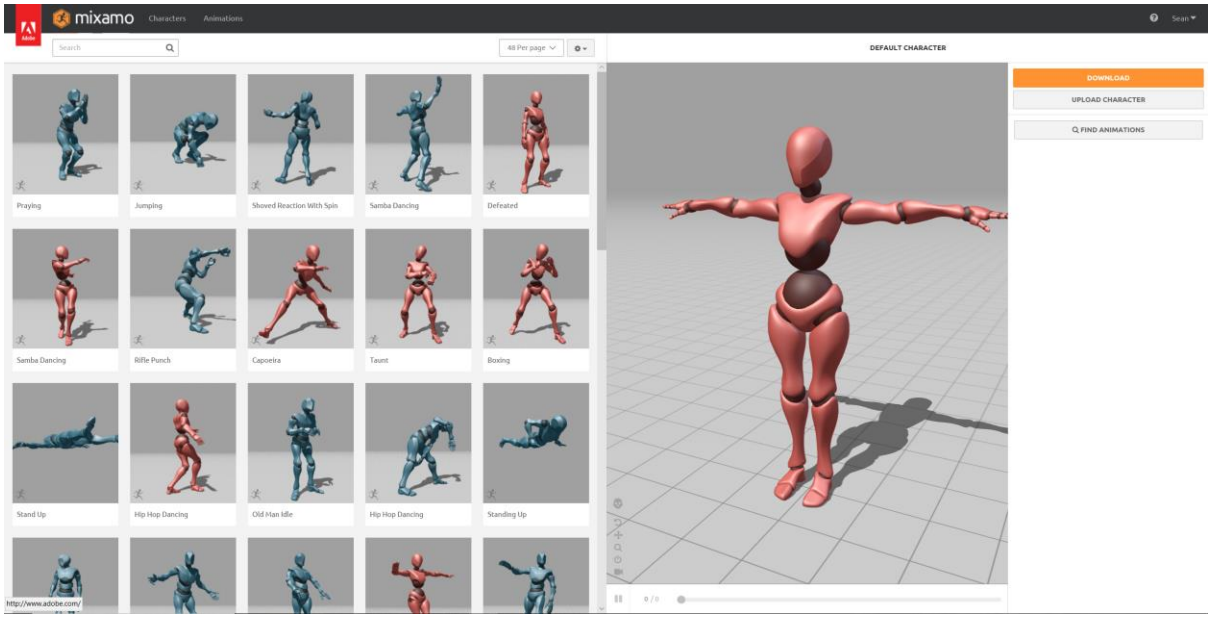
https://www.mixamo.com/#/

Sign up for a free account.



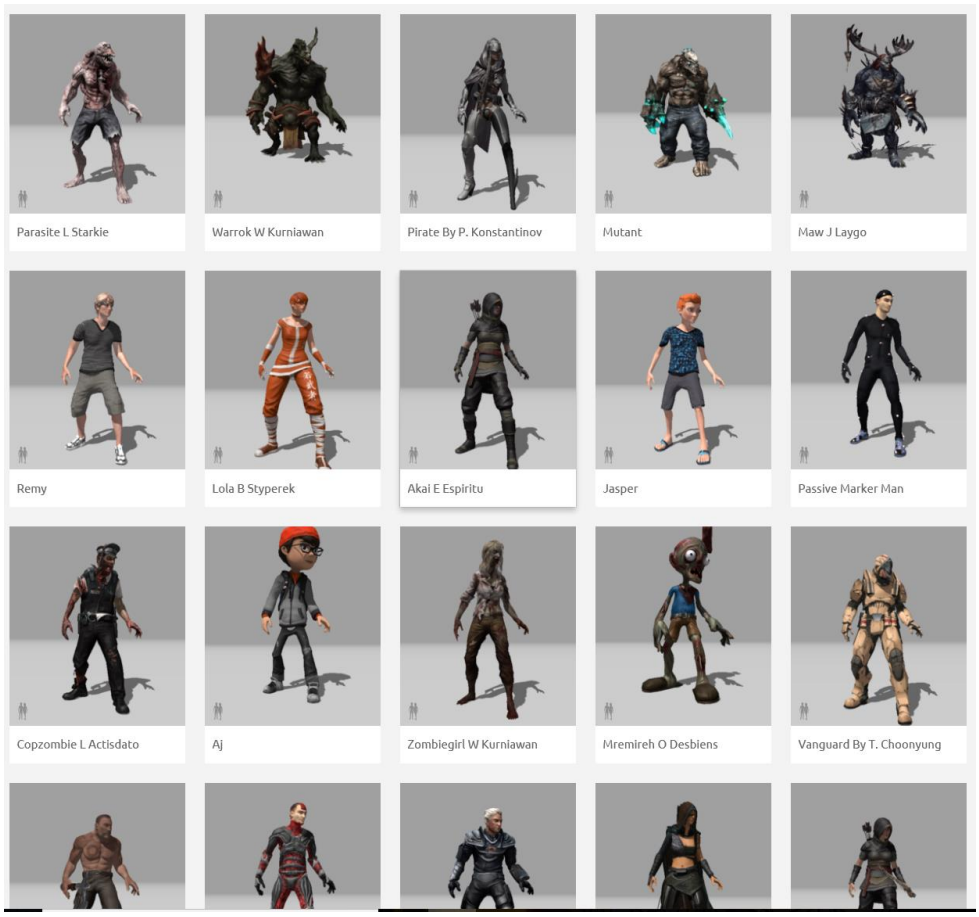
This is linked to Adobe, so if you already have an adobe account, you can use that to sign in to the website.

The first screen presented is:



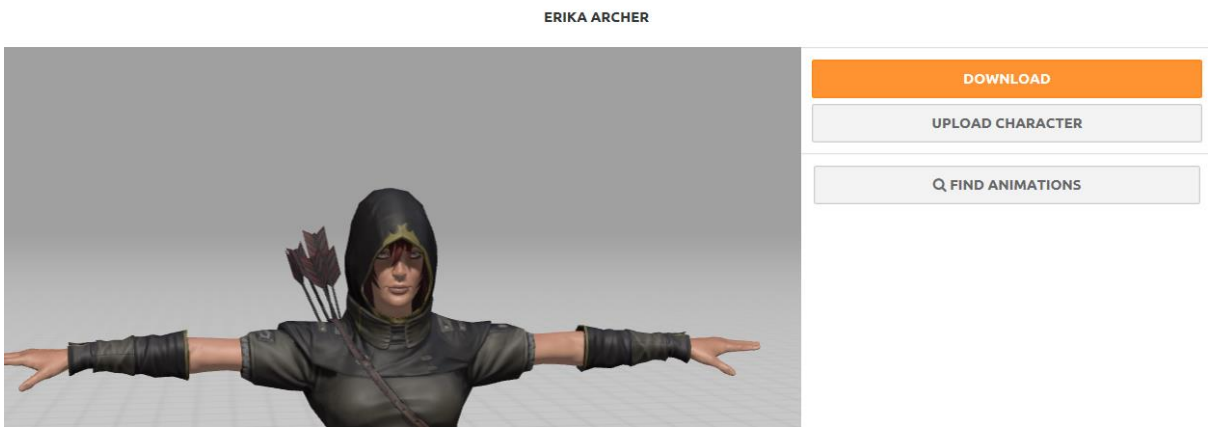
Mixamo allows you to upload your own characters and apply animations, but as you can see from the first page, we have a default character on the right, with a list of animations we can apply to it.

If you click on characters, you'll see a lot of pre-made characters.

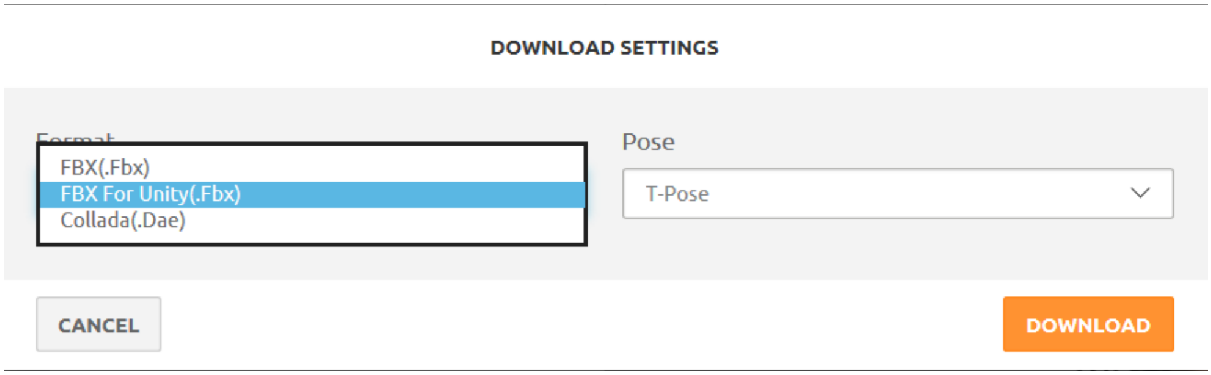


If you select a character from the left, it will replace the default character on the right. Select a character that you like. Once you have chosen, click back on the animation section of the site.

In this case, I've selected Erika Archer. From here, we want to download the model.



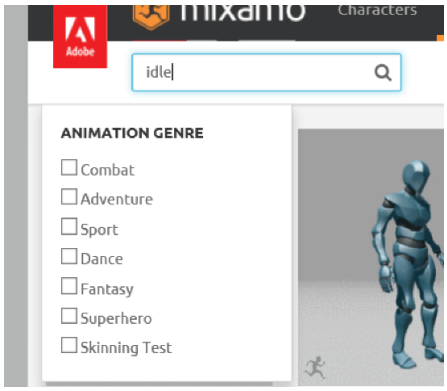
When you start to save, do a save fbx for unity in the format, T-Pose is fine.



Once saved, we will now move to animations.

From the animations page, we will be looking for a couple of animations. There are a lot of animations, 52 pages at 48 per page. So, we can use the search function locate the animations we are after. In this case, we will look for an idle animation, a run animation and a punch animation.

Search for idle



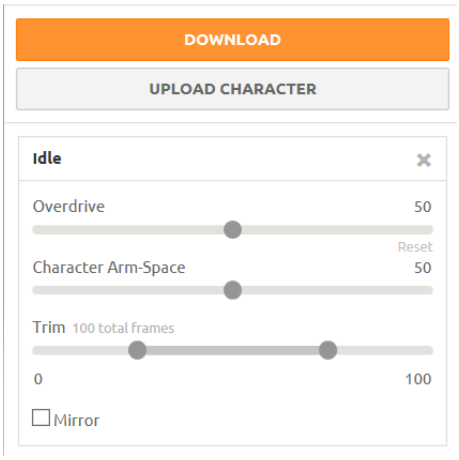
Notice the genre's that a relisted, this will assist in locating animations.

The search has revealed there is 11 pages of idle animations, with any good game, if a character is standing around too much, we want them to do something, this adds a level of realism to the character.

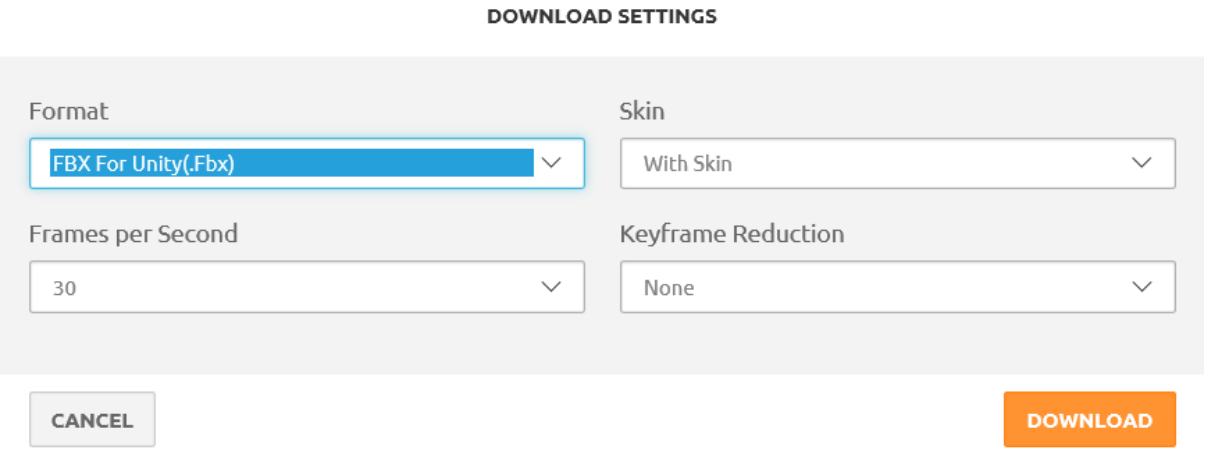
In this case, I'm going to add the idle that has minimal movement.

The animation is a slight movement with breathing and shoulders moving. Once you have clicked on the left panel, then you will see the animation applied to the character we have chosen.

There are additional aspects that can be applied to customise the animation even more.



In this case, I needed to drag the arms out to64, to ensure they wouldn't fall through the model. Check this on the model you have selected. From here, download the model.

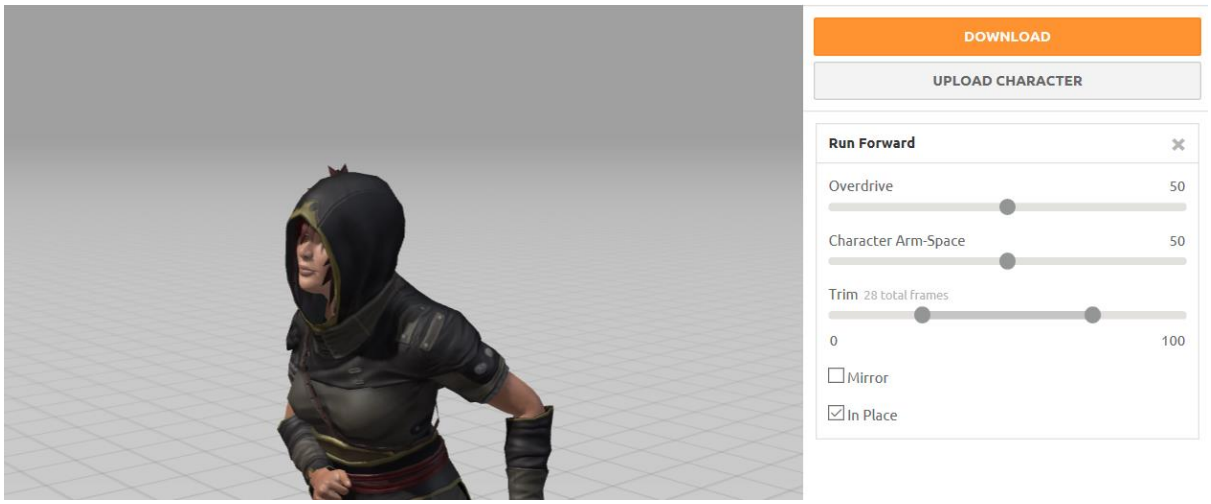


Now that we have the idle, let's search for a run. I'm selecting a run forward model.

In this case, the model is running off the screen, which is not quite what is needed. We need the character to run in the one place as we will be making the character run through code.

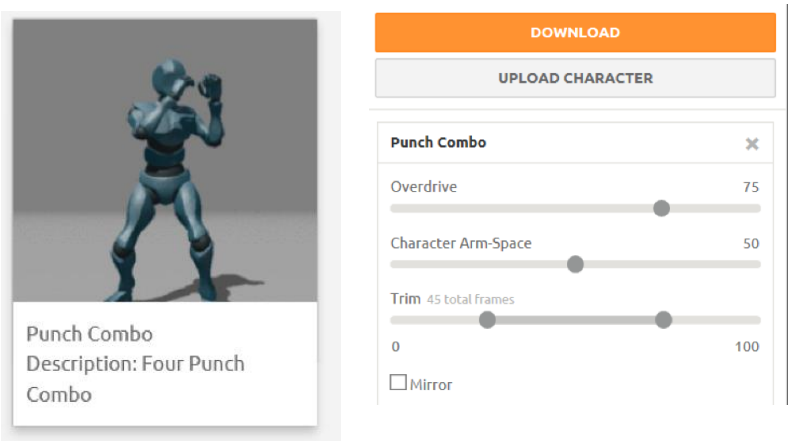
On the right-hand side, click the checkbox, In Place. Once you are happy with the model, download it.

RUN FORWARD ON ERIKA ARCHER

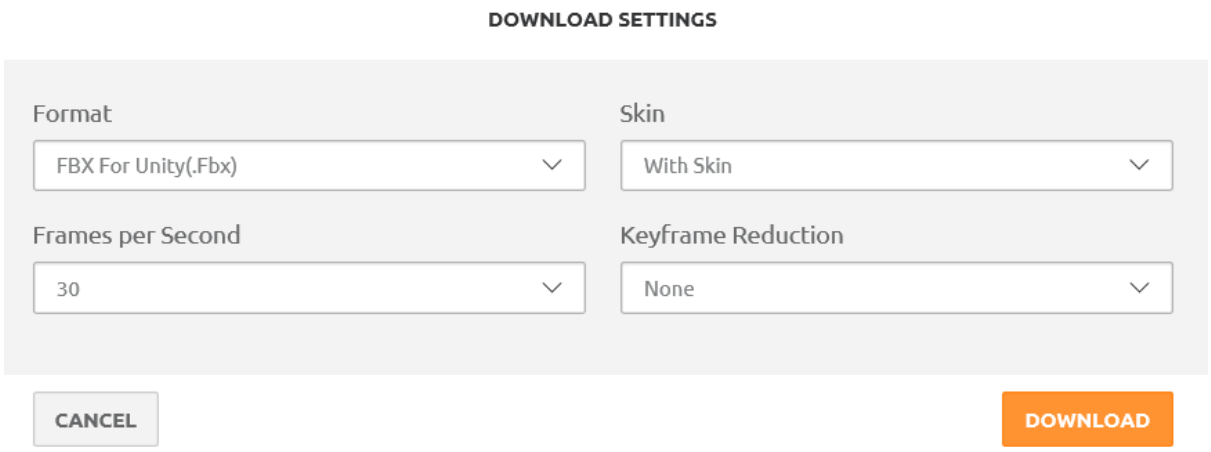


Next, search for a punch animation to add to the model.

I found this 4-punch combo that I'm going to use, I also increased the overdrive to 75.



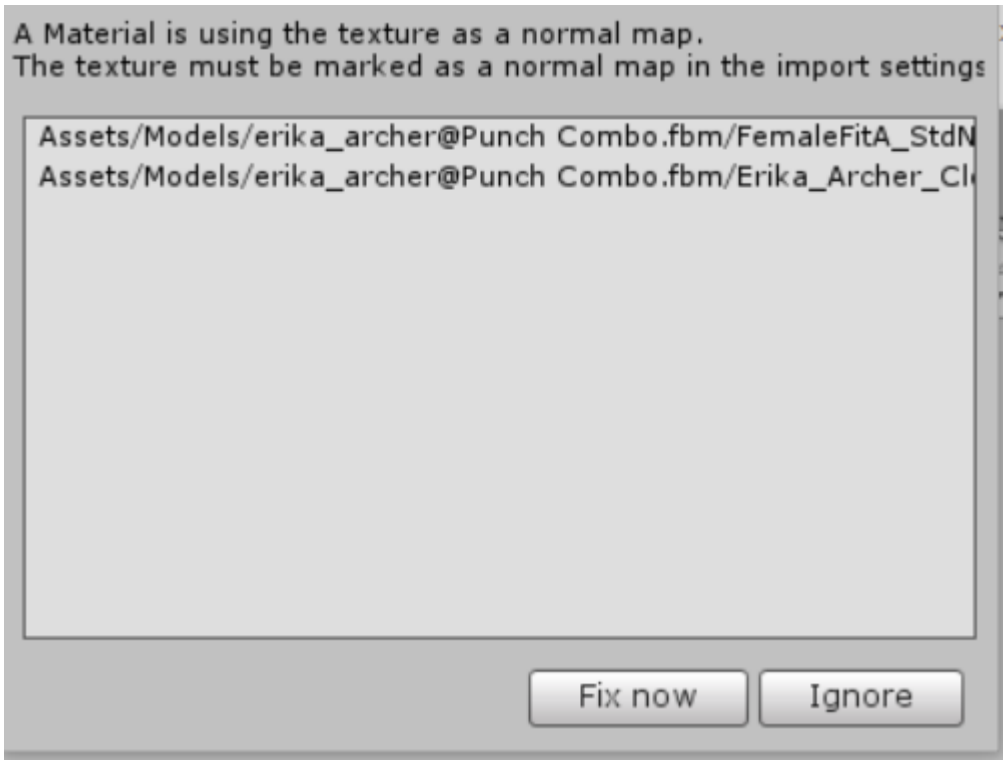
Download the file



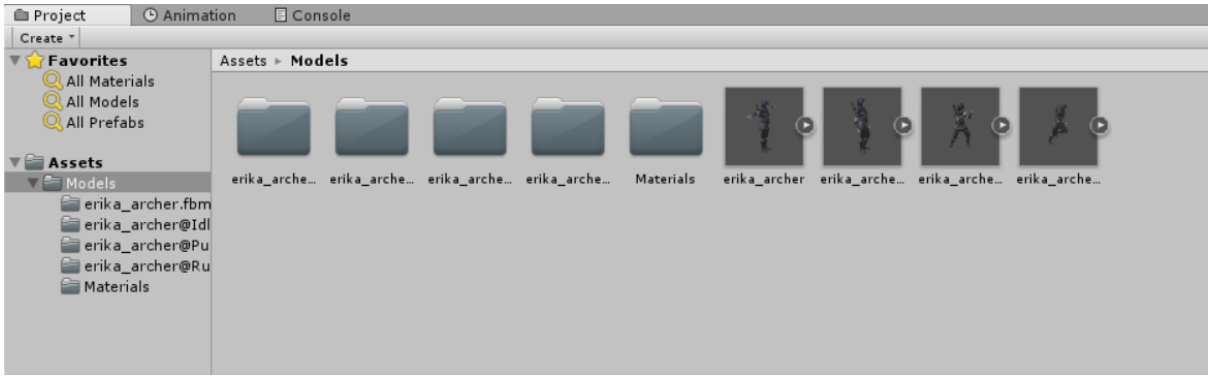
Now that we have all of our assets downloaded, go into Unity.

Create a Models folder in the Assets Panel and drag in all of the downloaded files from Mixamo.

If this occurs, click Fix Now.



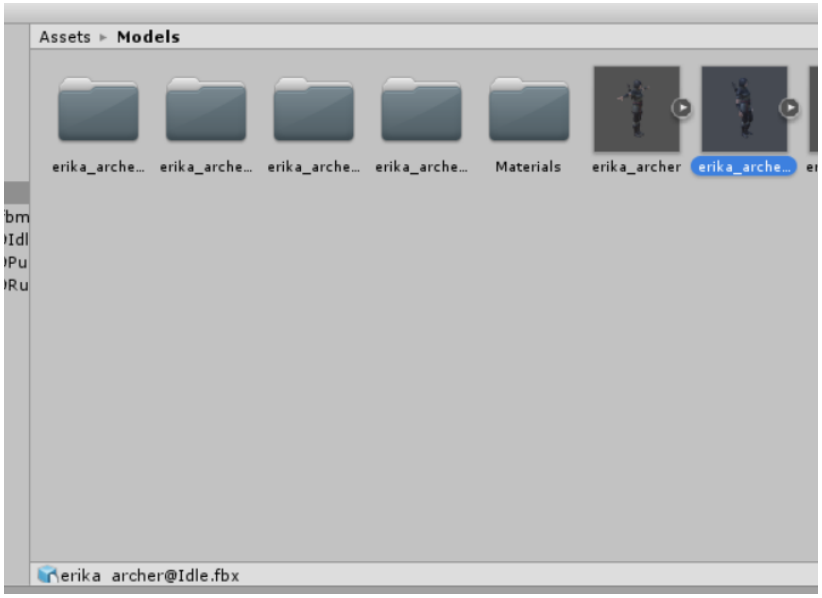
You should end up with the following (or something similar):



Next in the Assets Folder, create a new folder called Animation

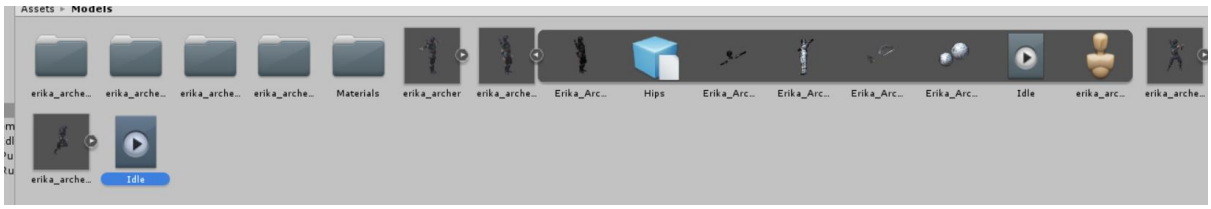
Now what we are going to do is to duplicate the animation and then drag out the animation from the models folder for each idle, run and punch models and place them into the Animation folder.

We will start with the idle model.

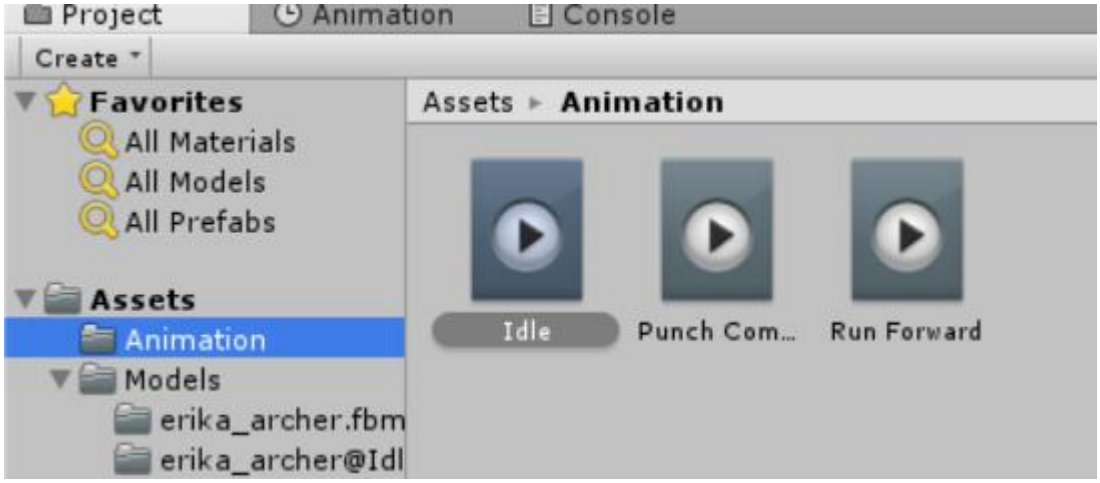


Notice that the name of the file is listed at the bottom of the panel. This makes it far easier to locate the correct file. Once this has been done, then click on the arrow located on the model, this will expand the model and we can see all of its components.

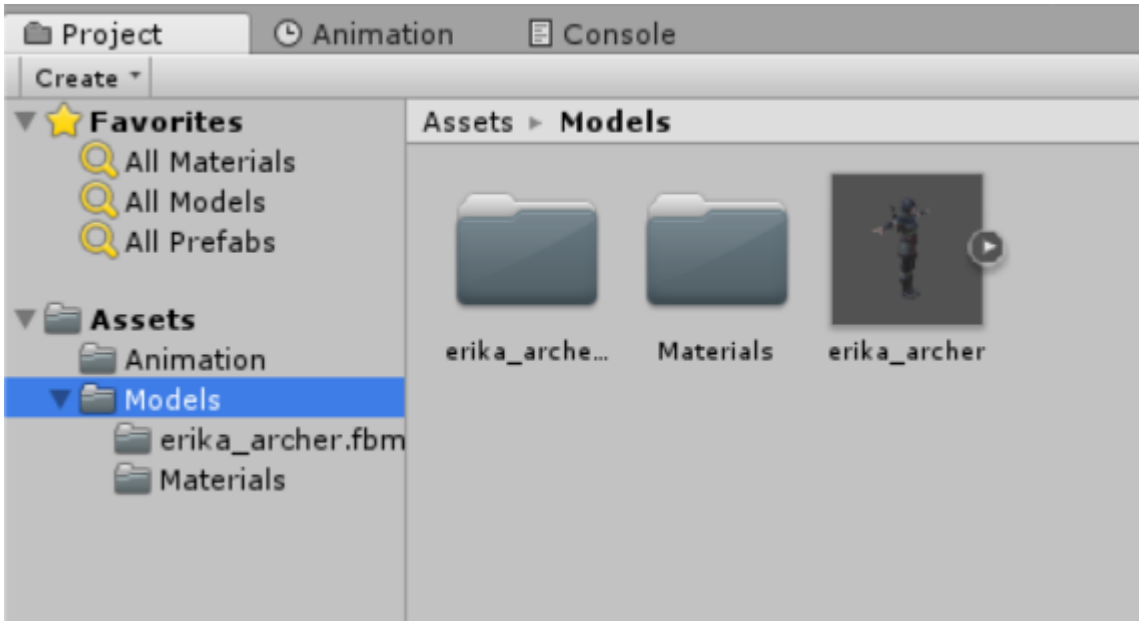
As you can see there is an Idle animation. Click on it and then do Ctrl+D this will produce the following.



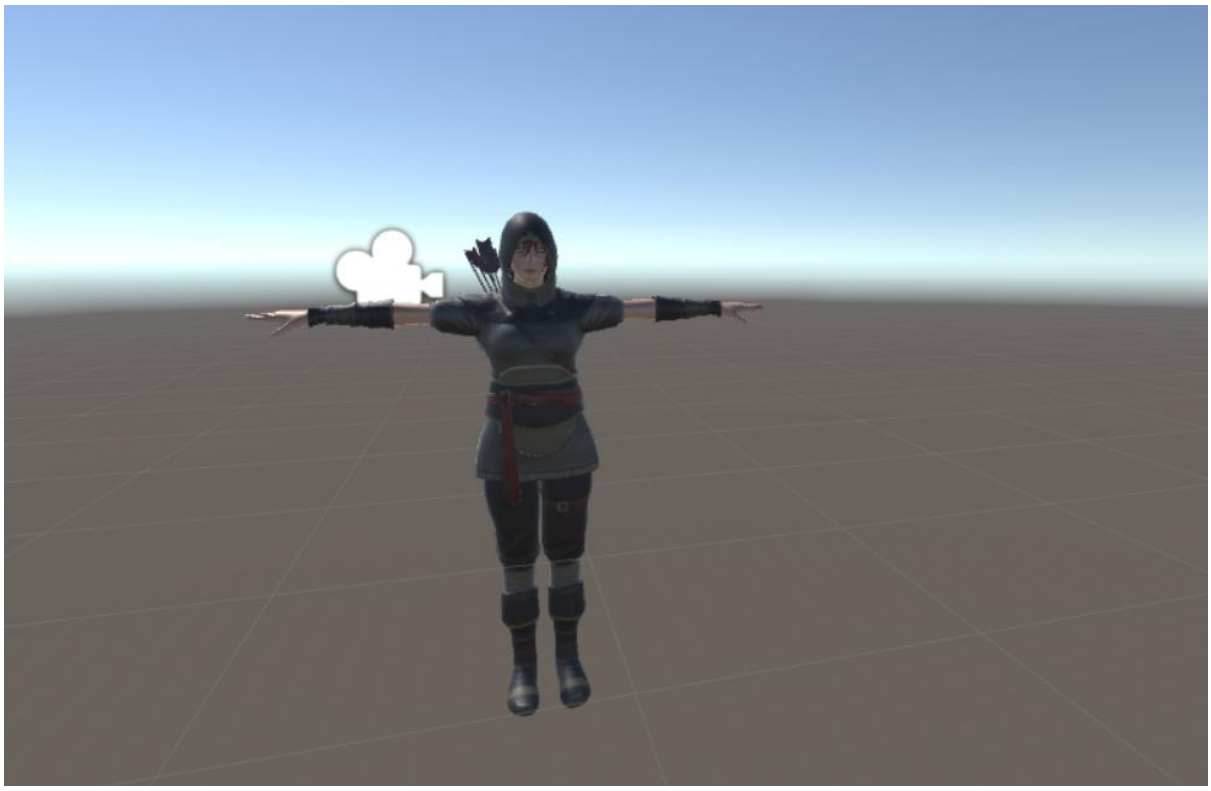
We have just copied the animation out of the model, from here we can then drag it into the Animation folder. Do this for the run and punch models as well.



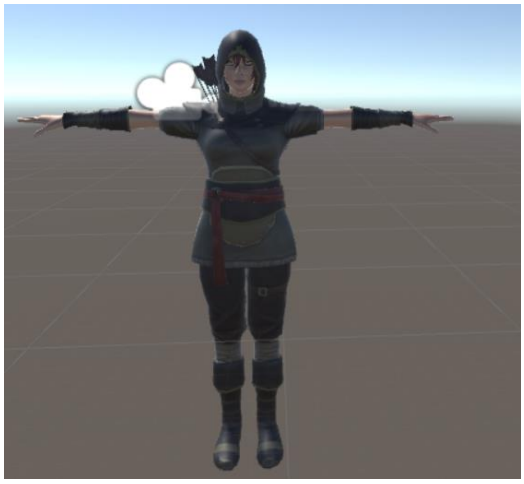
Once done, you can delete the extraneous models, leaving only the main model.



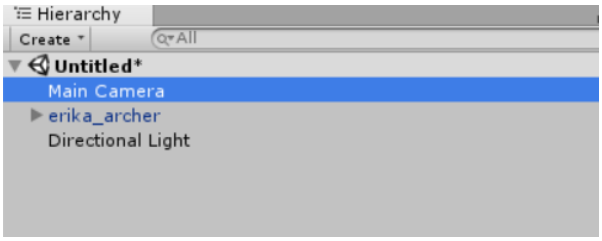
Now drag the model into the scene view



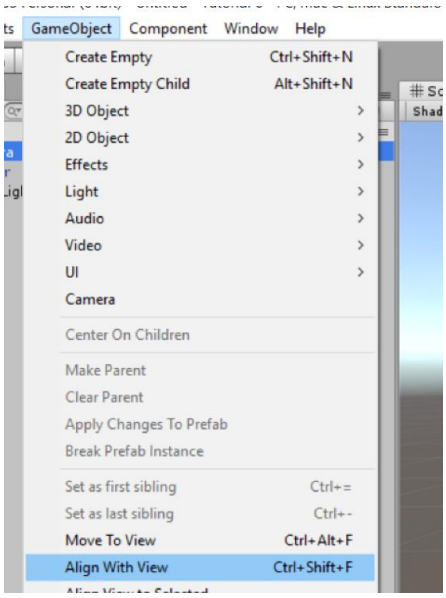
From here, if you view the main camera, we will only see the models back, so we will flip the camera around. The shortcut to doing this is to align the scene view on the model.



From here, select the main camera from the hierarchy.



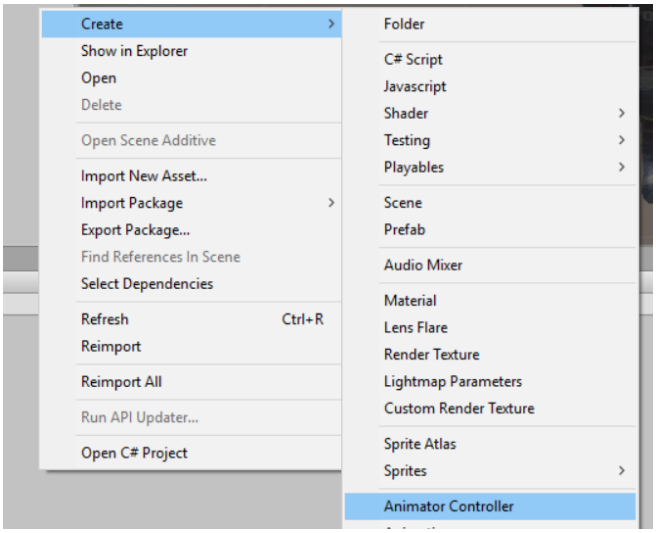
Next, we go to the menu and select GameObject -> Align with view.



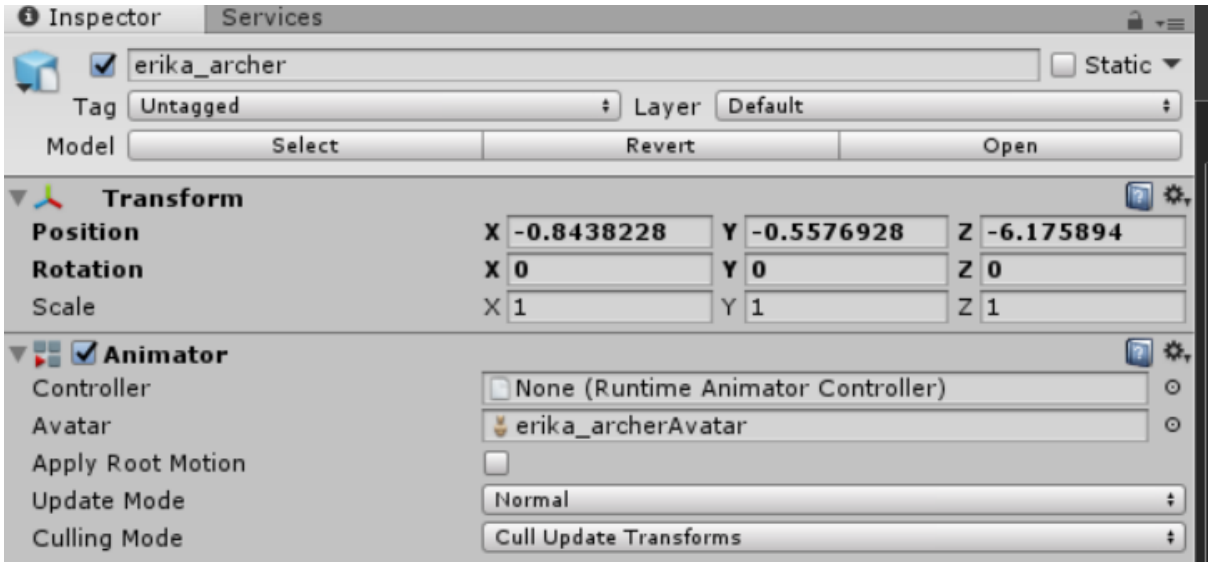
This will bring the main camera around the model, this will allow us to view what happens when we run the game.

Now that we have things set up, we want to start applying the animations to the model. To start with, we need to create an animator controller. This is done by right-clicking in the assets folder and selecting Create-> Animator Controller.

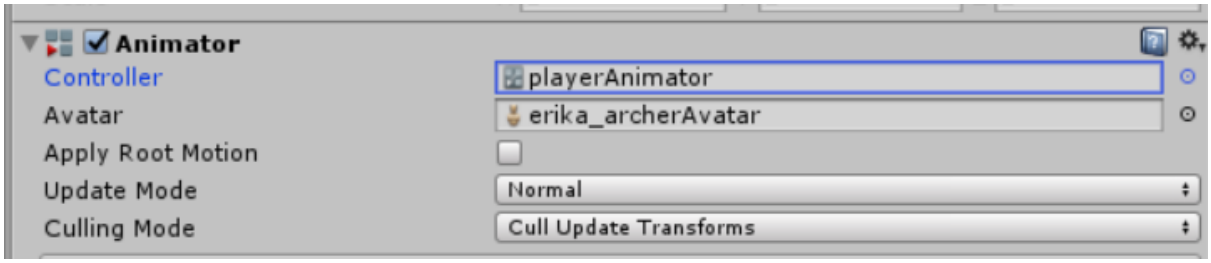
In this case it will be called playerAnimator. Put it in the animation folder.



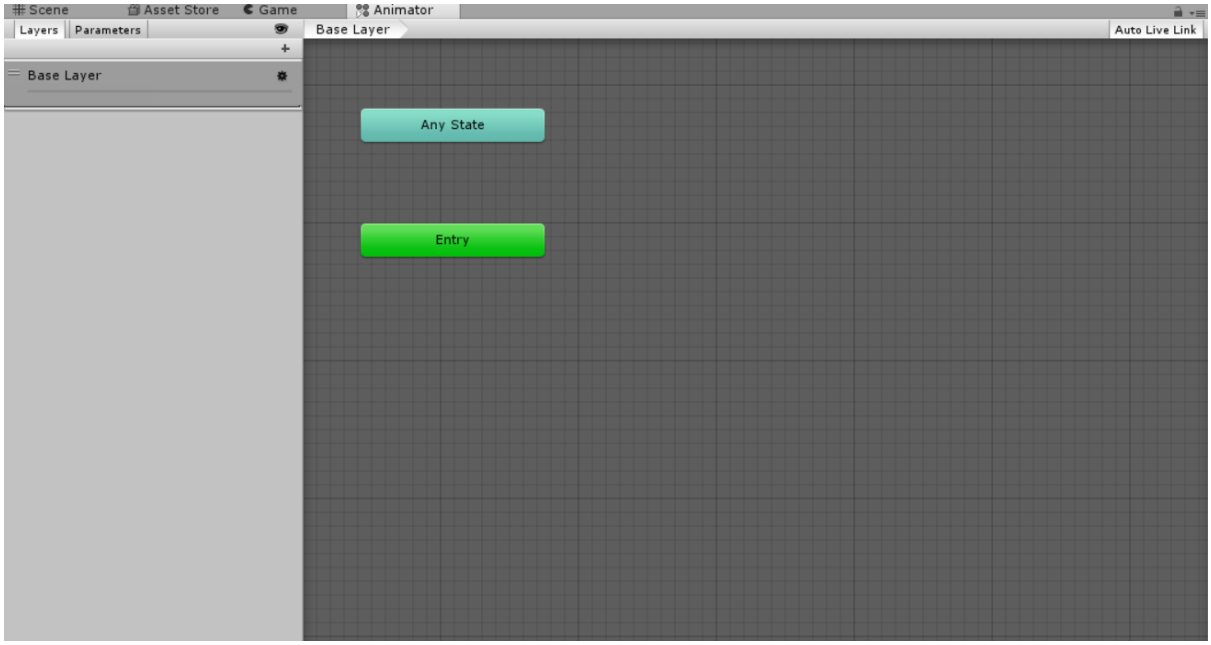
From here, we need to attach it to the Animator on the model. Select Erika and then look at the inspector panel.



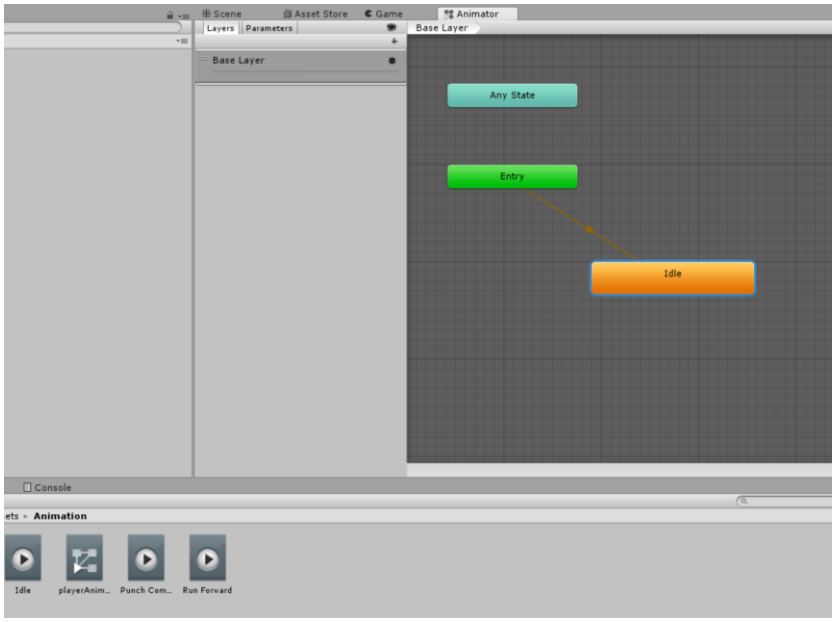
The model has an Animator component but no controller, that's the file we just created, use the target to locate the Animation Controller we created.



Once this is done, double click the playerAnimator, this will then open the following screen.



Click and drag the Idle animation from the assets folder into the Base Layer window.

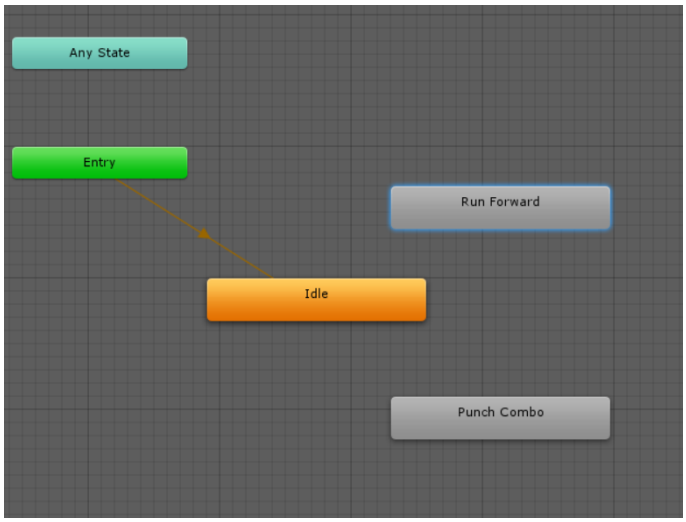


Idle has turn orange which means this is the default action of the character. If you run the game now, you will see the model standing doing the idle animation.

To ensure that the idle animation loops, select the animation from the assets panel and then examine the Inspector.

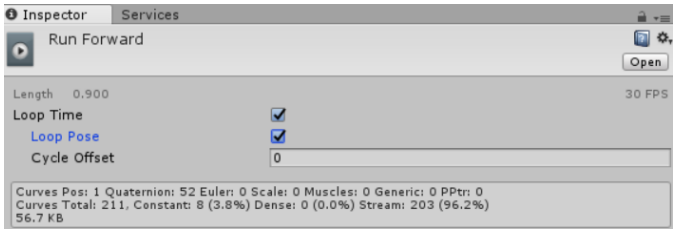
You will see that there is a loop time and loop pose that is unchecked. Check both boxes and then test it, the animation should continue to loop.

From here, drag the other animations into the animator view.

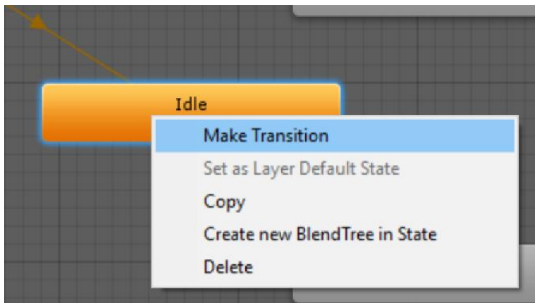


Notice that the new animations are not classified as the default action, this is why they are gray not orange. What we will do now, is make it so that the character will run forward when the player interacts, i.e. WASD or cursor keys.

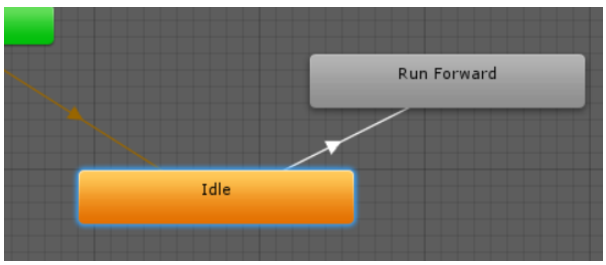
Looking at the actions we want, running will need to be a looped animation whereas the punch combo doesn't need to. So, click on the running animation and click the two check boxes for loop time and loop pose.



Now that the animations are done, let's make the transition occur, right click on idle and select Make Transition

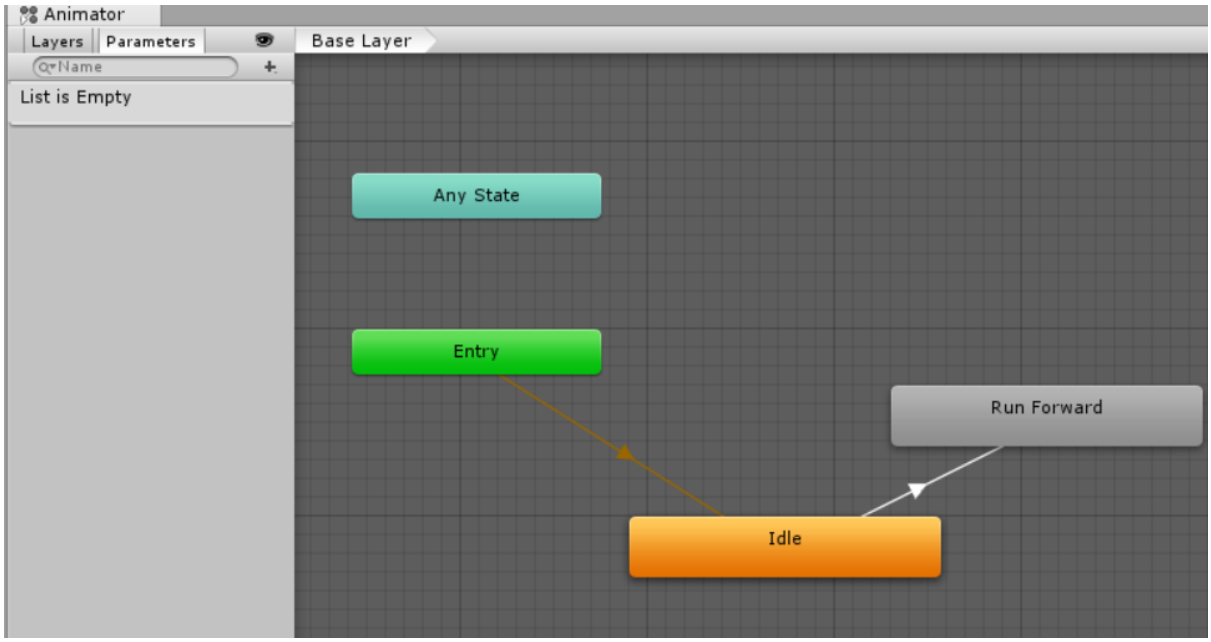


From here, drag the arrow on to the run forward animation.

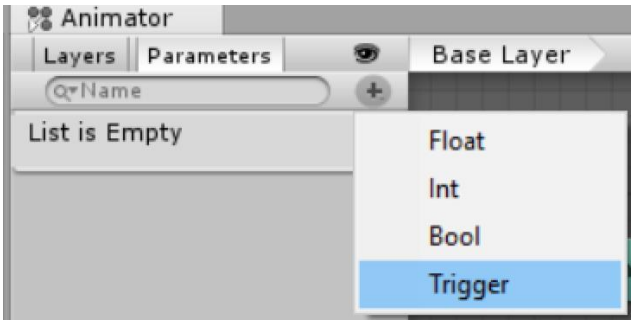


If you run the game now, you will see that the player stands in idle mode until the full animation is done and then breaks into a run, of which she continues to loop through. Though this is what we want we need it to be activated when keys are pressed, namely the W or up arrow. To do this we need to make it so that the transition state only occurs when a certain effect is happening. As such we will create a trigger value to apply to the transition.

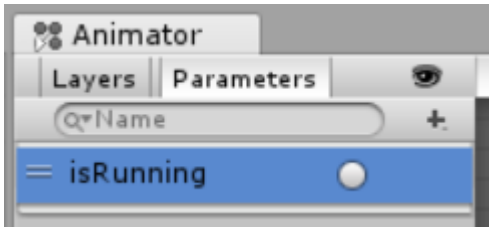
We need to create a trigger for this to occur, to do this we change to the Parameters section of the animator.



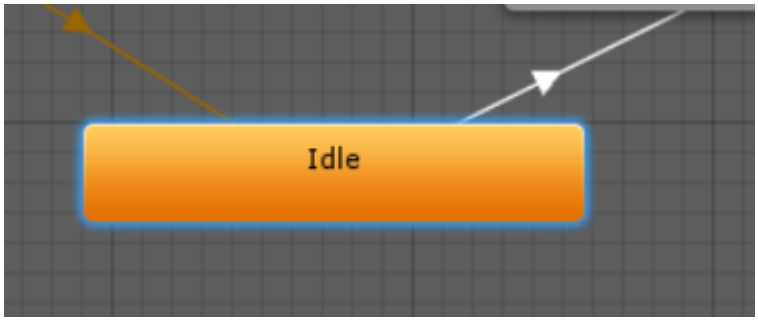
Underneath the Parameters label, is a plus sign, click on the plus sign and add a trigger value.

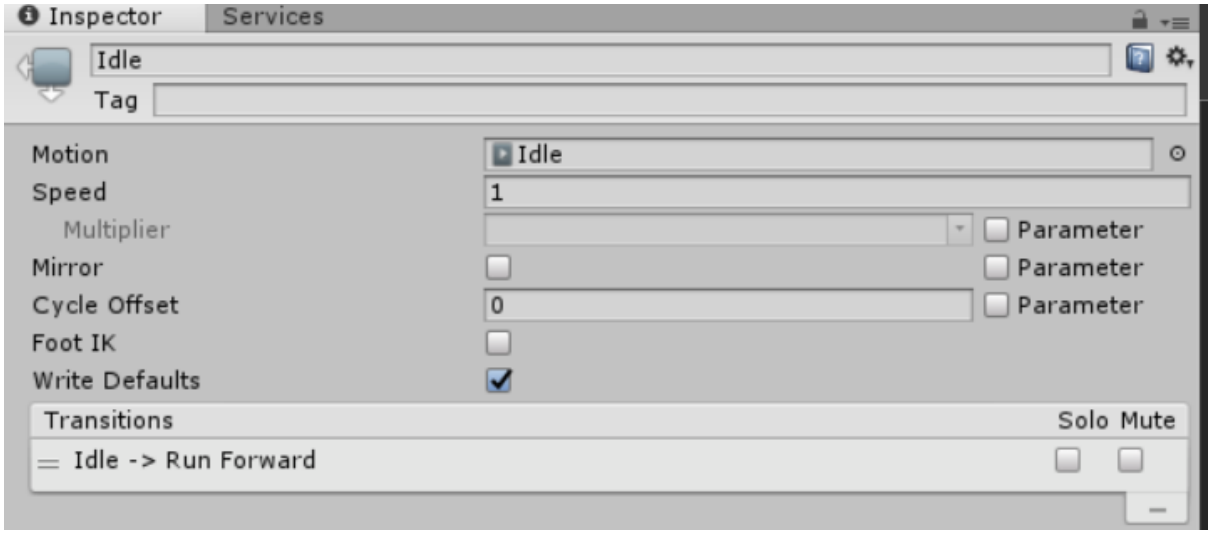


Call the trigger isRunning. You should end up with:

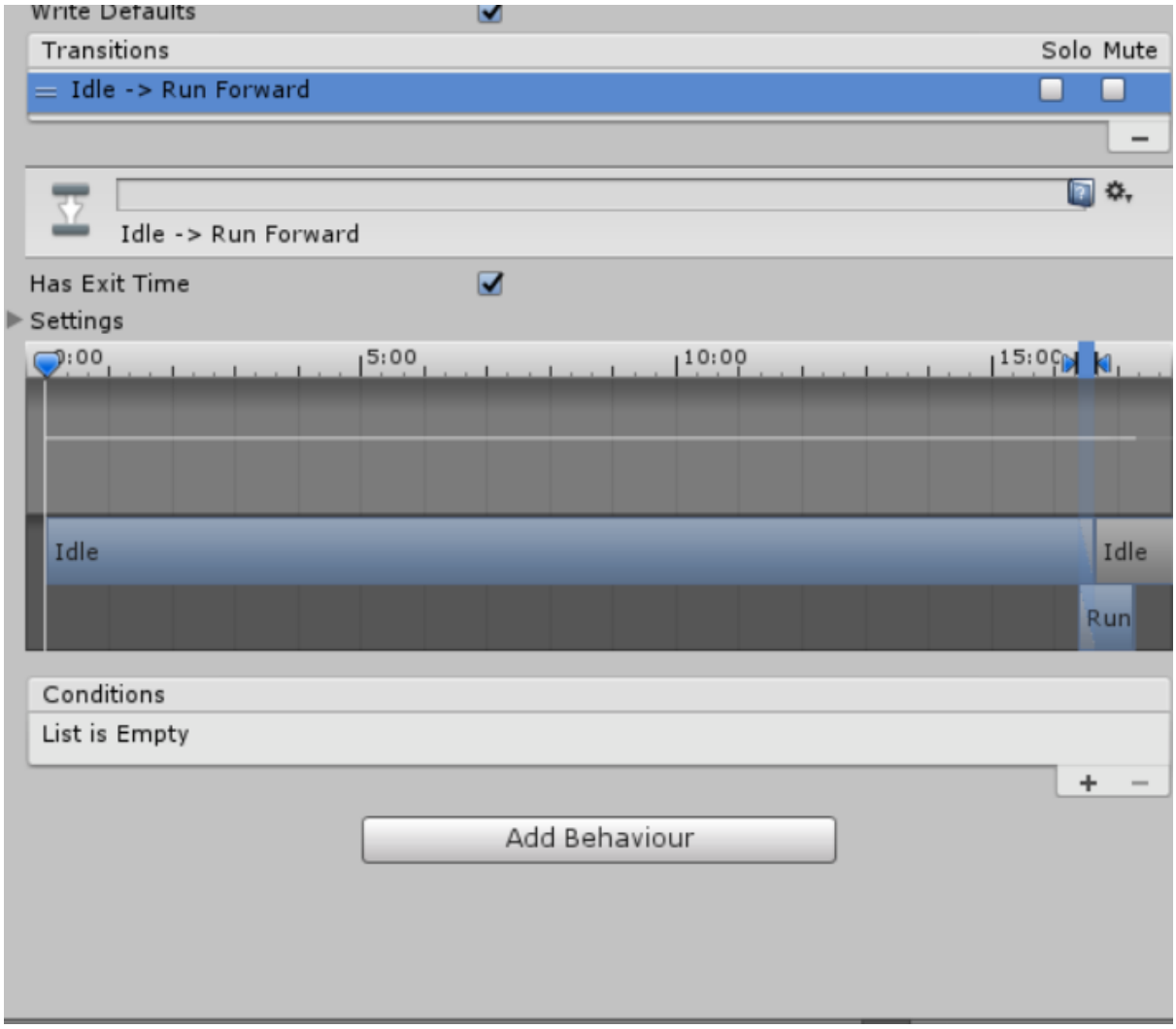


Once done click on the Idle button, then go to the inspector.

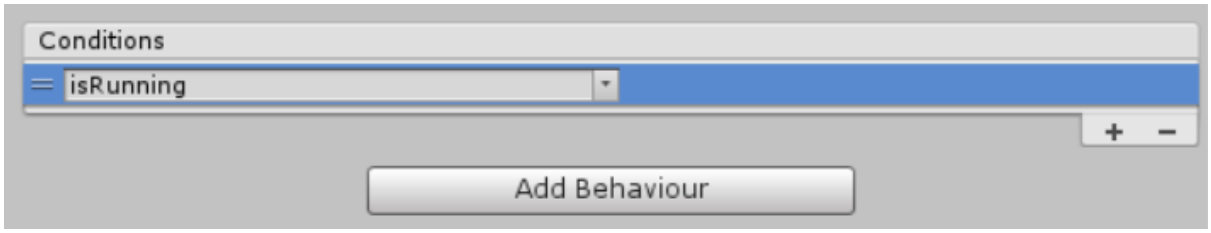




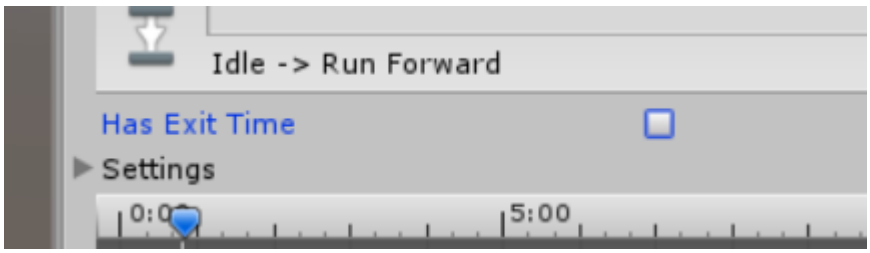
Next click on the Idle -> Run Forward This will open up the additional items on the Panel.



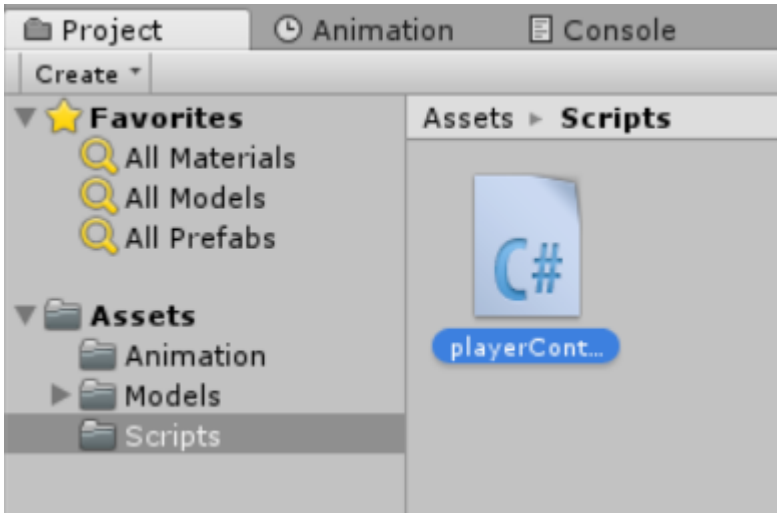
As the conditions section is empty, we need to add the isJumping condition. Click the + sign.



Before we move on, notice that there is a check box called has exit time. What this does is that it allows for an animation to complete before it moves onto the next, which is not the behaviour we want. If the player hits the W or up cursor we want them to start running straight away, not wait for the idle to finish. Uncheck Has exit time.



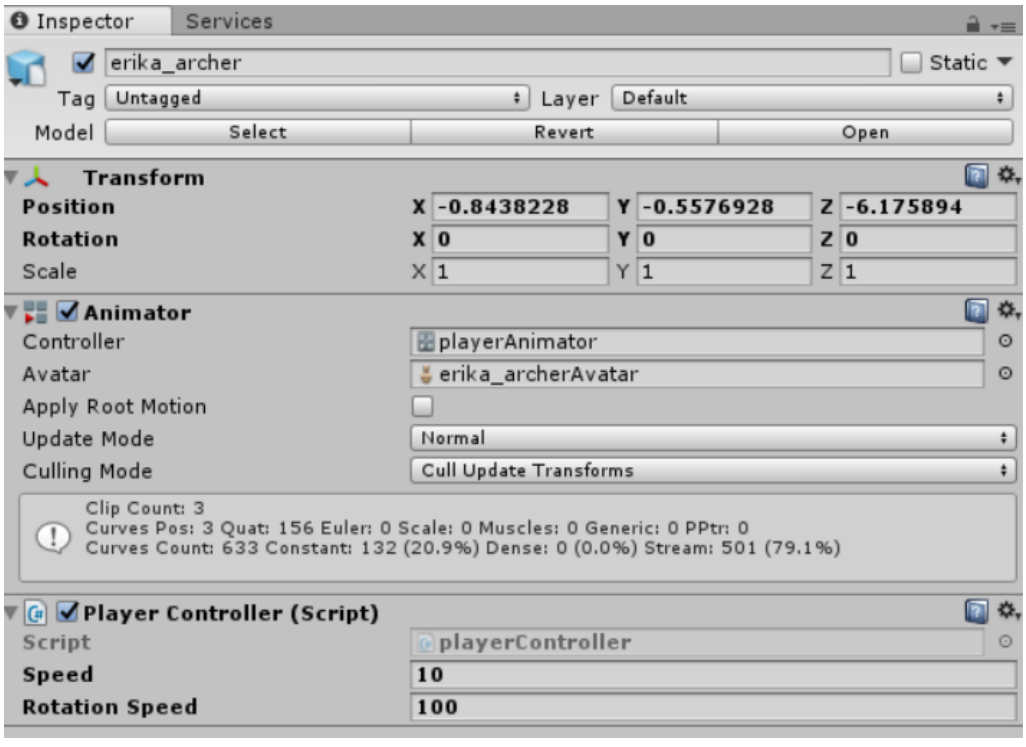
Now we need to make a scrip to activate the trigger. In the assets folder create a Scripts folder and inside of it, create a playerController script. Then open up the file.



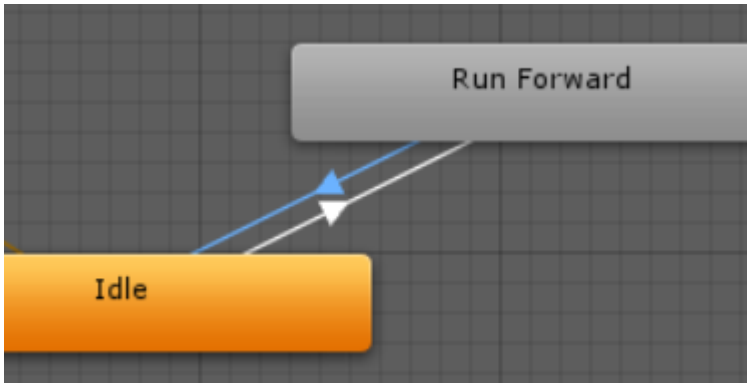
Write the following code

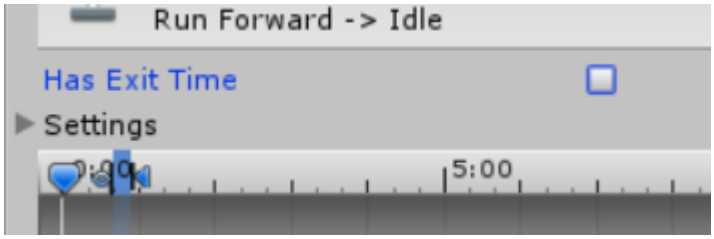
```
5   public class playerController : MonoBehaviour {
6       static Animator anim;
7       public float speed = 10.0f;
8       public float rotationSpeed = 100.0f;
9
10      // Use this for initialization
11      void Start () {
12          anim = GetComponent<Animator>();
13      }
14
15      // Update is called once per frame
16      void Update () {
17          float translation = Input.GetAxis("Vertical") * speed;
18          float rotation = Input.GetAxis("Horizontal") * rotationSpeed;
19          translation *= Time.deltaTime;
20          rotation *= Time.deltaTime;
21          transform.Translate(0, 0, translation);
22          transform.Rotate(0, rotation, 0);
23
24          if(translation != 0)
25          {
26              anim.SetBool("isRunning", true);
27          }
28          else
29          {
30              anim.SetBool("isRunning", false);
31          }
32      }
33  }
```

Once this is done, drag it onto the Erika inspector panel.

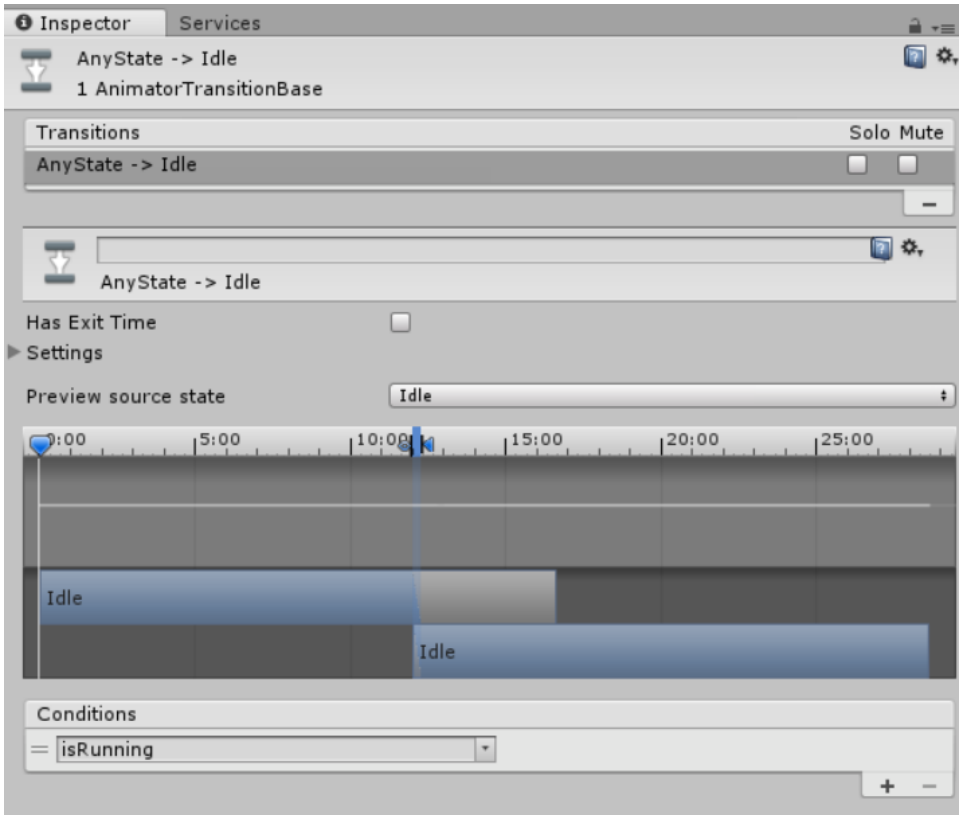


If you test it, we have the player running around. But, if we stop holding down the keys to move forward, we still have the animation for running working. To fix this, we need to add in a transition state from run forward to Idle. Click on the transition and uncheck the has exit time.





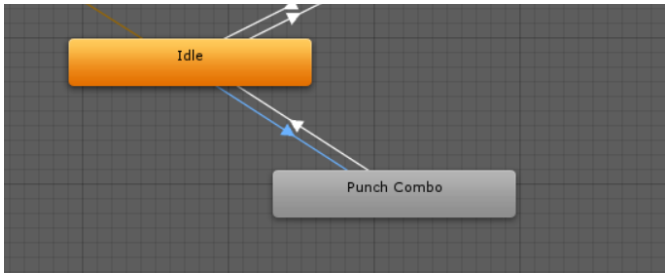
Ensure that the state of AnyState->idle has isRunning as a condition and the character will come to a stop.



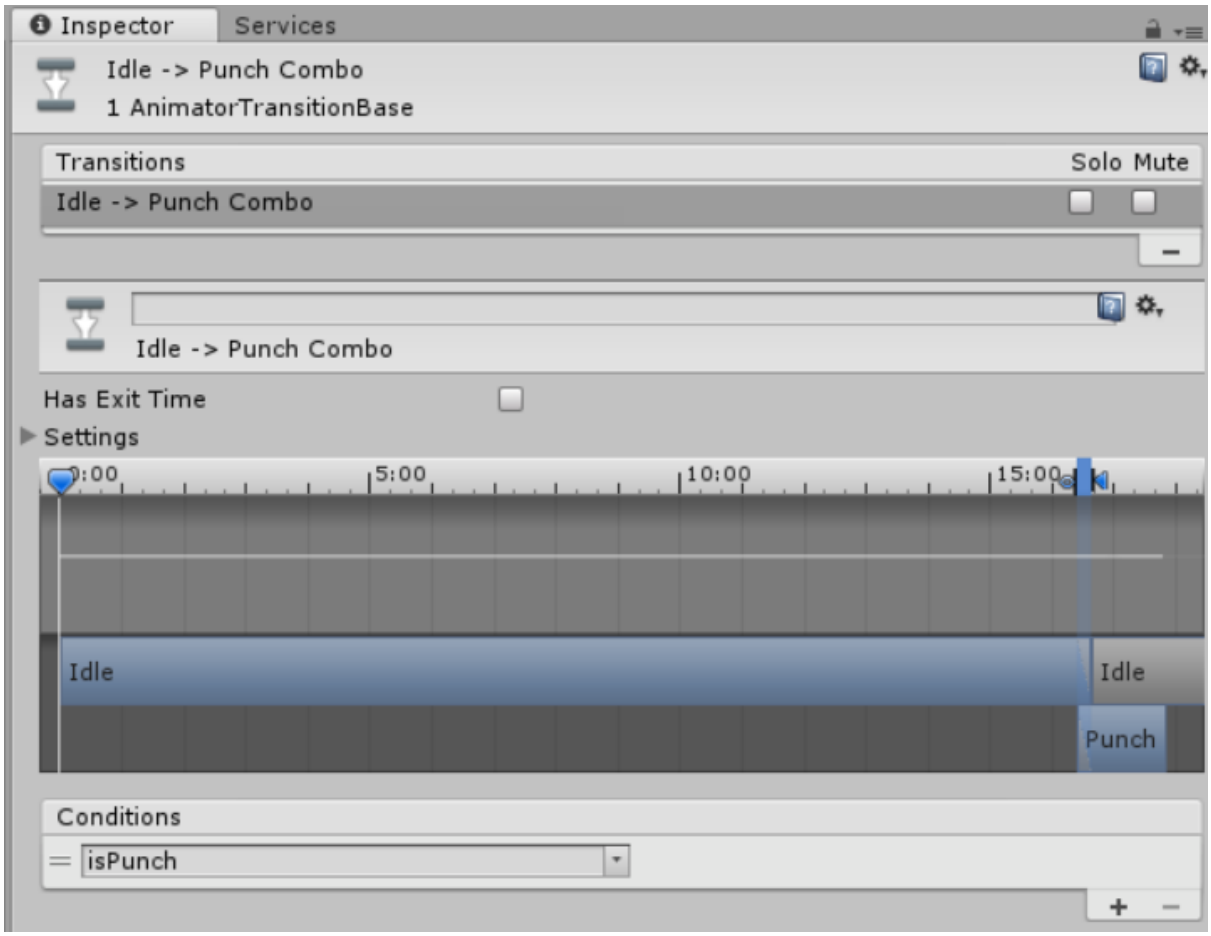
Repeat the same process for the punch animation.

Create a trigger:

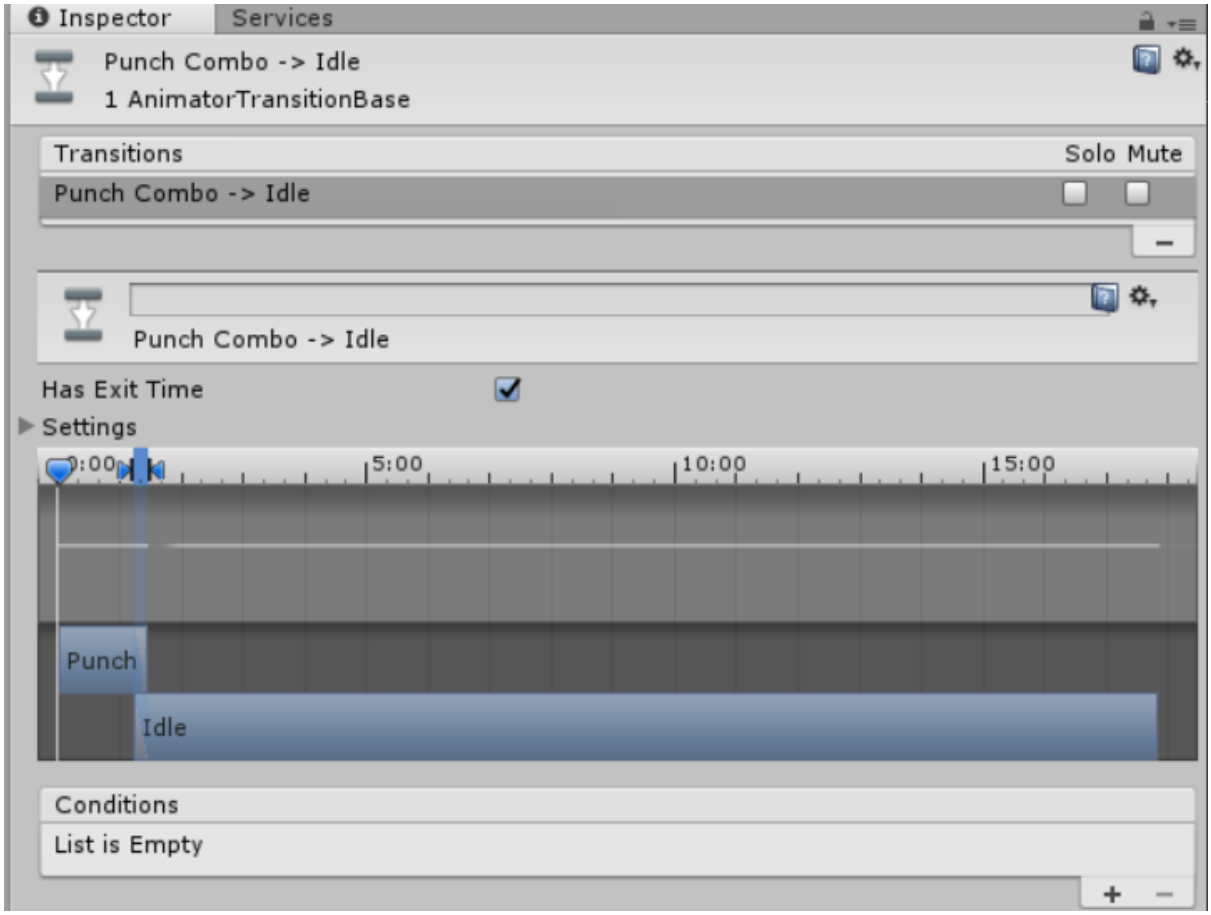
Link the animations



Inspector for the spacebar being pushed (Referenced in code as Jump.



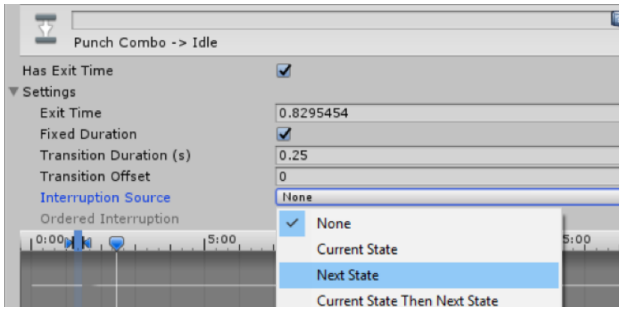
Inspector for return from Punch Combo to Idle



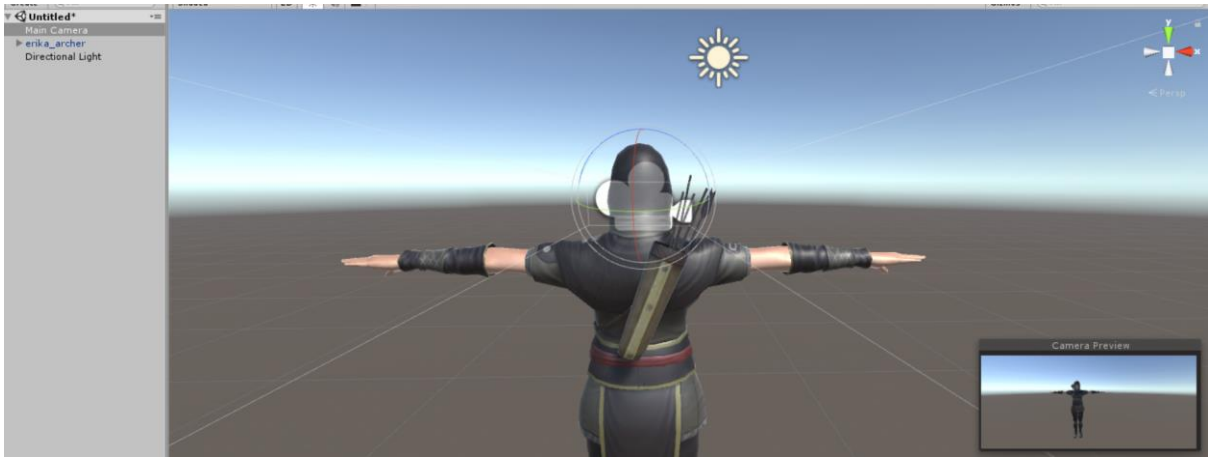
New Code applied to Player Controller

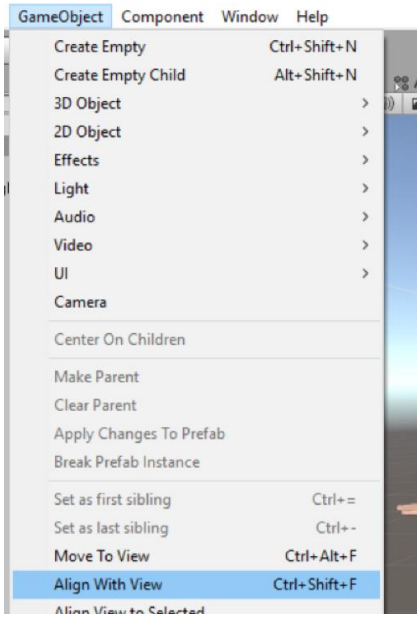
```
        if(translation != 0)
        {
            anim.SetBool("isRunning", true);
        }
        else
        {
            anim.SetBool("isRunning", false);
        }
        if (Input.GetButtonDown("Jump"))
            anim.SetTrigger("isPunch");

    }
}
```

Once this is done, test it. You should have the player punching. To tidy up the transition between animations, go into the inspector panel on the punch combo -> idle, go to settings and change the interruption source from none to next state.



From here, rotate and position the camera to the back of the model and then set the main camera to follow the character.





Now when you run the game, the camera will be behind the character.