

Animation Lab

Table of Contents

Table of Contents	0
Overview	1
Making Sprites	2
Frame-based Method	3
Making the Sprites	3
Importing Sprites	4
Creating Animations	6
Modular-based Method	8
Making the Sprites	8
Importing Sprites	10
Sprite Editor	10
Creating Animations	12
Supplementary Lecture	16
Checkoff Requirement	16
Challenge	16

Overview

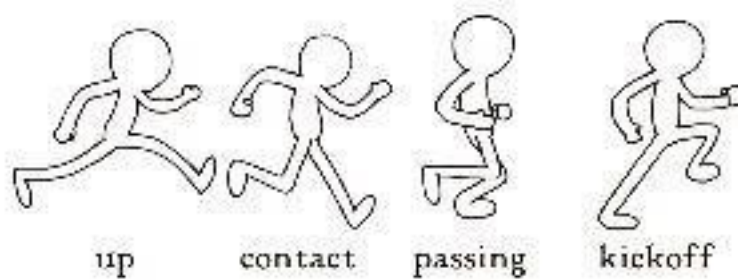
In this lab, you will be getting a preview of the game animation pipeline, from a Photoshop sprite to a finished Unity animation. For this lab, we will be focusing on a puppet-based method of animation that uses many of the tools that 3D animations use. We will also be covering some animation techniques for creating sprite-based animations; however, we won't go too in depth about how to animate it in Unity, since there is another lab (Animator and Blend Trees) that covers it in more detail, so check that out!

The **Modular** or Puppet method has more overhead, and is harder to make look realistic, but it is very easily scalable if you have lots of animations. It's not well-suited to low-resolution or pixel animations, although there are ways of making it work. **[Warning:** This method may take you longer than the sprite-based method, depending on the complexity of your model. It's only recommended if you are comfortable with sprite-based animations.]

The **Frame-Based** method generally takes more time per animation frame but gives you much more freedom in your sprites, allowing for more convincing animations. Since we're only making one animation, both methods will take a similar amount of time, so choose the one you're more interested in, or do both!

Making Sprites

Open a canvas in Photoshop or your preferred software. In this lab, we'll be making a walk animation. You may either choose to draw an original/existing character, or use our sprites provided in the file. Artists, we recommend you try drawing your sprites.



We'll be doing the **passing and up** poses. This is just **four frames** in total (two for each step for the left and right), although for your actual game animations you can (and should!) add frames in between to make the animation more fluid.

We'll first go over the frame-based method.


Frame-based Method

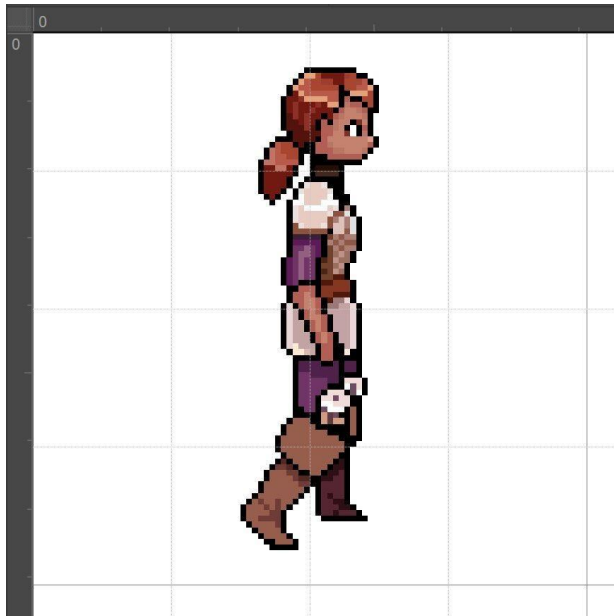
Making the Sprites

1. **(Photoshop Specific) Set up a grid.** (If you don't have Photoshop, see if your drawing software has a grid tool. Otherwise, do your best to space your drawings out correctly). In photoshop, go to **View > Show** and make sure **Grid** is checked. Then go to **Edit >**

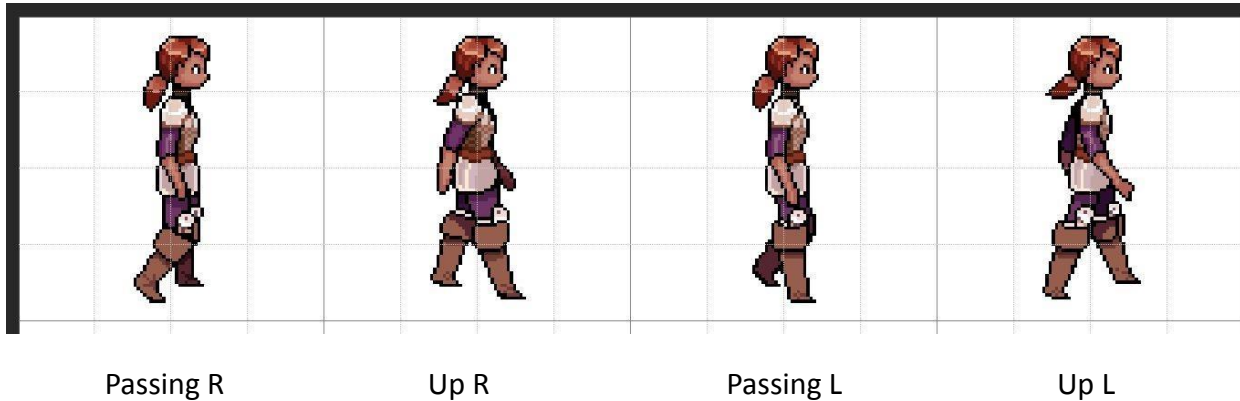
Preferences > Guides, Grid & Slices and change **Gridline Every...** to the right size. Each grid box will contain one sprite. If you're doing pixel art, **64** or **128** pixels usually works. If you're drawing regular art, you'll need something bigger, like 500 or 1000 pixels, depending on the size of your canvas.

2. **Draw one keyframe.** Use the grid to center the drawing inside the **top left grid square**.

With **View > Snap** enabled, you can do this by selecting the **Move tool**  and holding **Shift** as you click and drag the layer vertically or horizontally. We'll be making adjustments to this frame for later frames to reduce the amount of work necessary.



3. **Draw the other three frames on separate layers.** You can do this by copying your one frame three times into adjacent grid squares, then changing parts that need to move. This can save you having to redraw parts like the head and face, which don't change as much.



Keep in mind the 12 principles of animation while you animate to make your animations bouncier, and more believable! Also, don't be afraid to look up frames online by googling images of *insert action* animation frames (ie walking animation frames) so that you can get an idea of how many frames you need and what each frame should roughly look like.

Importing Sprites

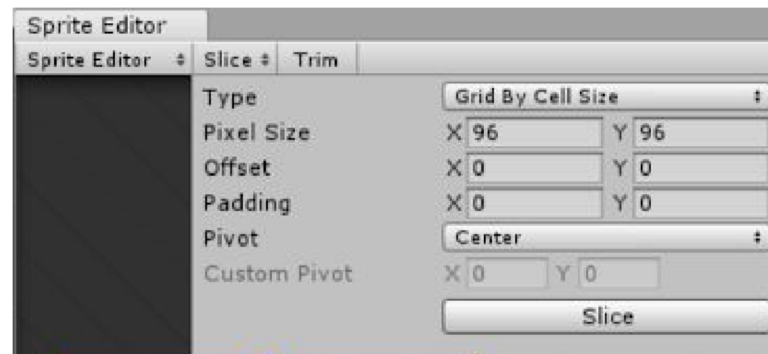
1. **Create a new 2D Unity project.**
2. **Create a Sprites Folder inside Assets**
3. **Import**
 - a. Inside the folder, right click the background and select **Import New Asset**. Locate and import your sprite sheet.
4. **Change Settings**

- a. In the import settings window in the Inspector, change **Sprite Mode** to **Multiple**. If you are doing pixel art, change **Filter Mode** to **Point** (this will prevent your sprites from being blurry). Then click **Apply**.

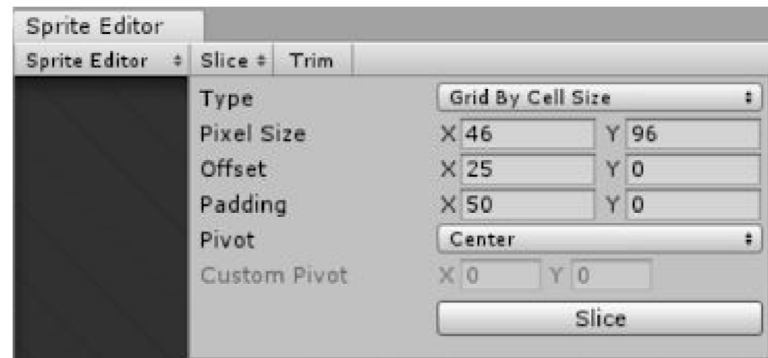
5. Sprite Editor

a. Frame Based:

- i. In the window that opens, click **Slice**. Then, in **Type**, select **Grid By Cell Size**. Choose the size that you used to make your grid in Photoshop.



- ii. If your sprites are taller than they are wide or vice-versa, like many character sprites, you can use **Offset** and **Padding** to change the cell dimensions. Make sure the x and y **Padding + Pixel Size** sums are equal and that **Offset = Padding / 2**, like below:



Click **Slice**.

Creating Animations

This part will go by very quickly. If you want a more in depth explanation and learn how to animate a sprite to walk in other directions as well, check out the Animator and Blend Trees lab.

1. Right click in the project **Hierarchy** and click **Create Empty**. Select the object and name it **Player**. Then, click **Window > Animation > Animation** to open the **Animation** window. Then, select **Player** and click Create. This will create a New Clip. Name this clip **walking.anim** and save. Notice that if you click **Player**, an **Animator** component is automatically added. For future reference, to create more animations, select the box that has your animation name (in this case **walking.anim**) and click **Create New Clip**.

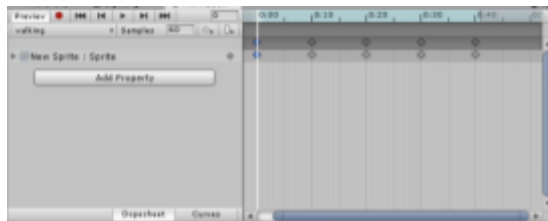


2. Frame Based

- A. In the **Sprites** folder, click on the arrow next to your sprite sheet to expand it.



- B. Click and drag each sprite frame from the **Sprites** folder to the **Animation** timeline, at **0:10 intervals**. **Add the last frame twice, 0:10 apart**, to prevent a jitter as the animation loops. You should have a total of 5 frame markers. Click the play button in the **Animation** window to see the animation. Feel free to adjust the **Samples value** and the frames' spacing if the animation is too slow or fast.



Congrats, you're done!

Next, we'll cover the modular based animation.

Modular-based Method

Making the Sprites

Similar to the sprite based method, you may either choose to draw an original/existing character, or use our sprites provided in the file. Artists, we recommend you try drawing your sprites.

1. **Draw your character.** You'll be separating all their limbs into a big weird body-part sheet later, so pick a pose where the arms, legs, and head are all distinct and make a strong silhouette. Make sure the character is facing either right or left (¾ view can work as well).



2. **Modularize.** Using the lasso tool (or something similar in the drawing program you use if you don't use photoshop) select each significant body part, cut them out of the character, and paste them somewhere on the canvas. At a **minimum**, you'll need these parts:

- Head + Torso (Can be one part)
- Right:
 - Arm
 - Thigh
 - Calf + Foot (Can be one part)
- Left:
 - Arm
 - Thigh
 - Calf + Foot (Can be one part)

For some parts, you'll need to go back and **manually extend the lassoed drawing** to allow for a smooth range of motion later on (see rounded-off limb ends below). Feel free to be as detailed as you want with the parts, but **make sure to leave a lot of white space in between each one!**



Importing Sprites

1. **Create a new 2D Unity project.**
2. **Create a Sprites Folder inside Assets**
3. **Import**
 - a. Inside the folder, right click the background and select **Import New Asset**. Locate and import your sprite sheet.
4. **Change Settings**
 - a. In the import settings window in the Inspector, change **Sprite Mode** to **Multiple**. If you are doing pixel art, change **Filter Mode** to **Point** (this will prevent your sprites from being blurry). Then click **Apply**.

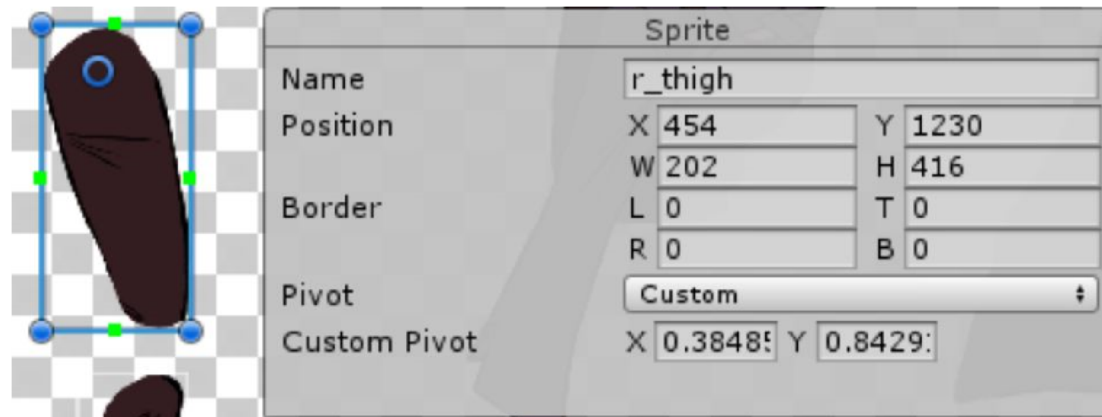
Sprite Editor

In the window that opens, click **Slice**, make sure the settings are **Automatic, Center, and Delete Existing**, then click **Slice** again. This will automatically separate the sprite sheet into individual sprites.



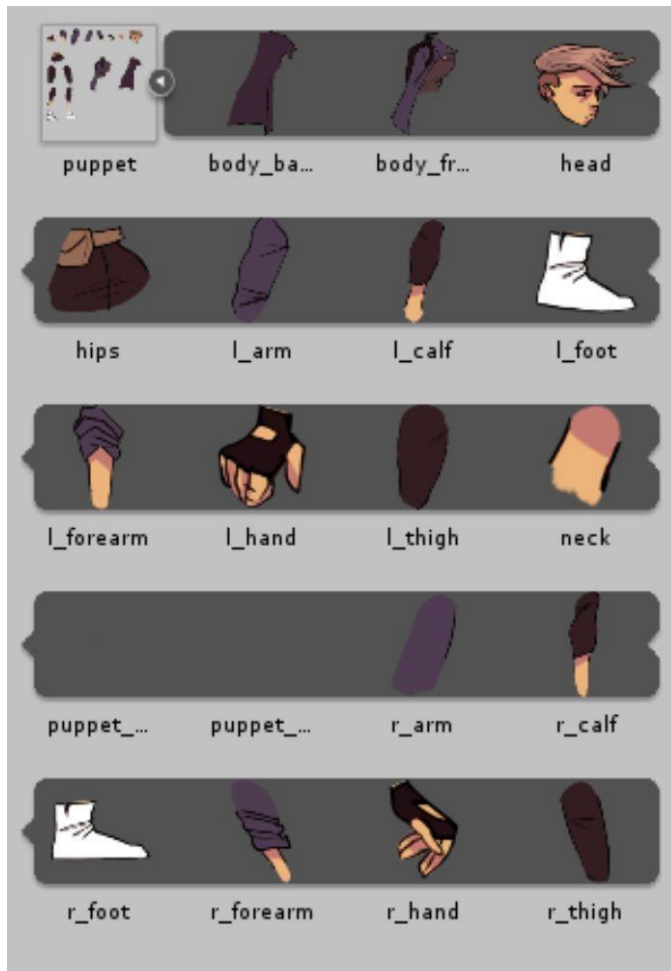
Now, to edit the pivot points so that they resemble joints, click on each part and move the blue circle to its correct pivot location. For example, thighs pivot around the hips, calves pivot around the knee, and arms pivot around the shoulder. You can also set pivots values manually in the **Sprite** window by entering values in the **Custom Pivot** box.

Now is also a good time to name the body parts. As you set pivots, rename each of the body parts with a R or L label and the part name. This will make organization much easier later on.

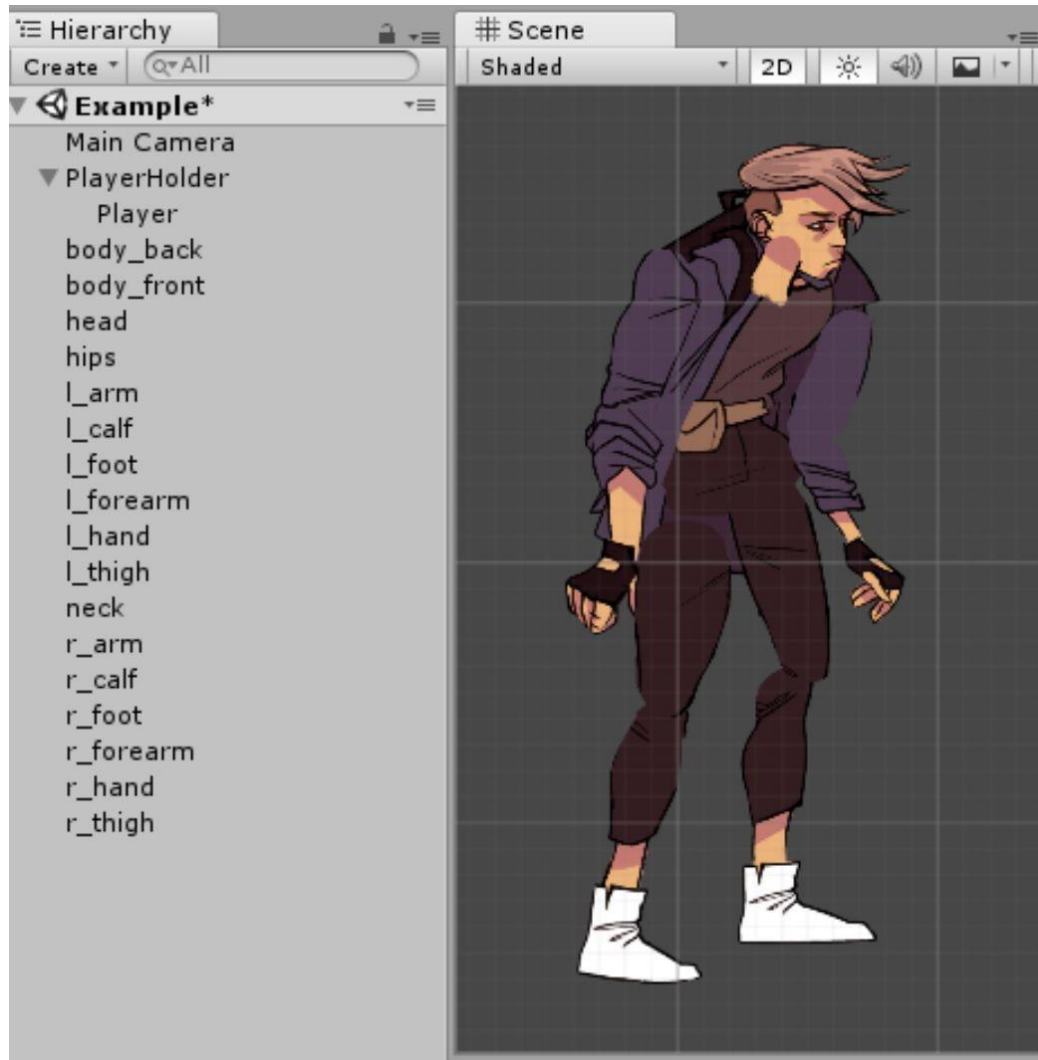


Creating Animations

1. Right click in the **Hierarchy** and select **Create Empty**. Name this empty object **PlayerHolder**. Create another empty game object, name it **Player**, and drag it onto PlayerHolder so that it becomes a child. This is so that translations and rotations we make to the character while animating it won't affect its position in the world space.
2. Expand your sprite sheet in the **Sprites** folder by clicking on the gray arrow.

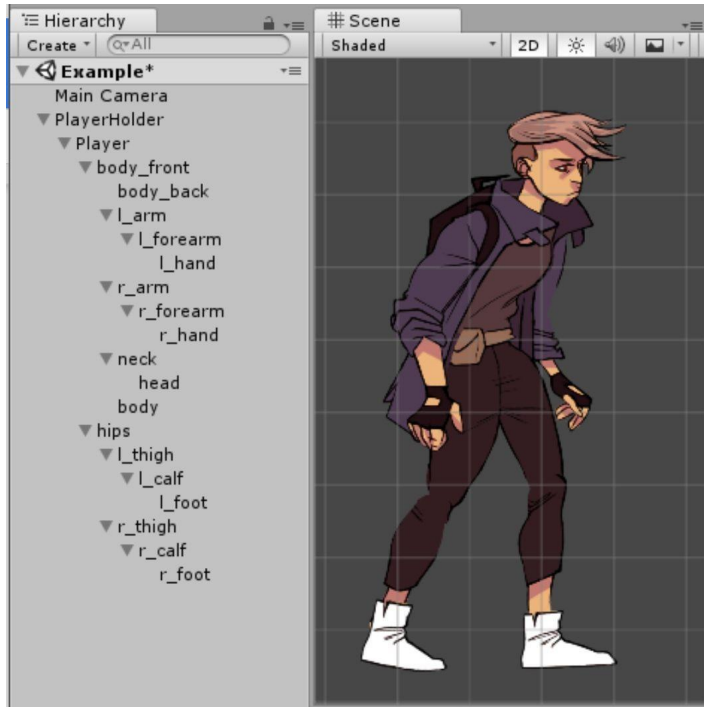


3. Click and drag each body part onto the **Scene** view to its approximate location.



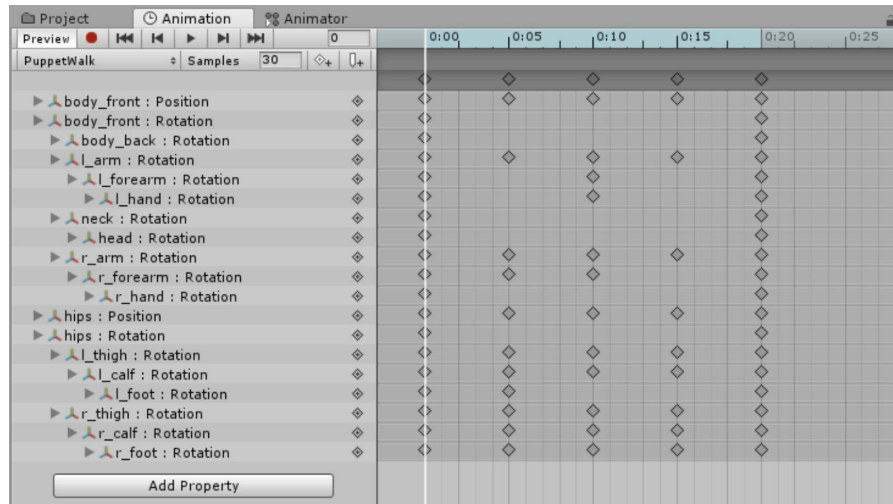
Then rearrange the objects in the **Hierarchy** with parent-child relationships that roughly follow the organization of a skeleton. For example, **Hips > Left Thigh > Left Calf > Left Foot**. This will make it much easier to animate later, since moving the hips would move everything attached to them at the same time.

4. The layer ordering will be messed up, so go to **Edit > Project Settings > Tags & Layers** and create a new sorting layer called **Player** using the + button. Then, select all body parts and in the **Sprite Renderer** in the **Inspector**, and change the sorting layer to **Player**. Now, change body parts' **Order in Layer** values to fix the ordering (like below) - objects with a higher **Order in Layer** value will be displayed above others. Take this time to also adjust the positioning of all body parts so that your character looks right. On the toolbar left of the **Play Button**, make sure that **Toggle Tool Handle Position** is set to **Pivot**. This will allow you to rotate your body parts according to your custom pivot points.

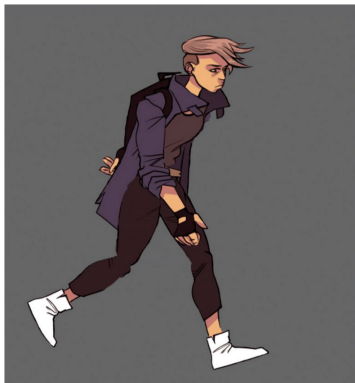


5. Click the **Recording Mode (red circle)** button in the **Animation** window. This will save all your object rotations and translations as frames of animation. Move the white animation line marker to the **0:05** mark. Then, use the rotation and translation tools to move each body part **in the scene** to make a walking keyframe. Go to the **0:10** mark and move everything to the next keyframe. **Once you have made four keyframes, copy the first keyframe and paste it 0:05 after your last keyframe.** (You can do this by clicking on the top gray diamond of the last keyframe so that all the diamonds turn blue, doing Ctrl+C, then moving the white animation line marker to the right time and doing Ctrl+V).

6. When you're done, click the **Recording Mode** button again to exit recording mode. Notice that if you scrub through the top blue animation timeline, Unity interpolates your keyframes to create a smooth animation! At this point, your Animation window should look something like this:



7. Click the play button in the **Animation** window to see the animation. Feel free to adjust the **Samples value** and the frames' spacing if the animation is too slow or fast.



Congrats, you're done!

Supplementary Lecture

This supplementary lecture demonstrates how to improve your animations in the Animator using the 12 principles of animation.

It is linked here: <https://youtu.be/l8FbrXuJu9A>

Checkoff Requirement

Show your animation window and play your finished animation.

Challenge

1. Try the other type of animation, if you didn't do one of the types.
2. If you haven't drawn your own characters to animate, do so! On either of the animation types. If you don't have a drawing tablet/aren't able to draw on your computer, design some frames on paper.