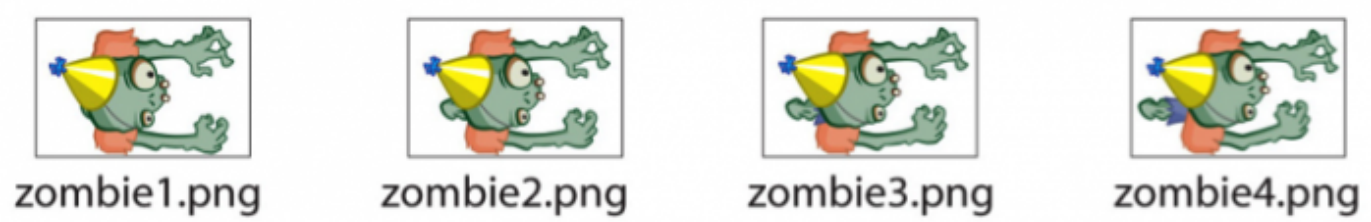
raywenderlich.com

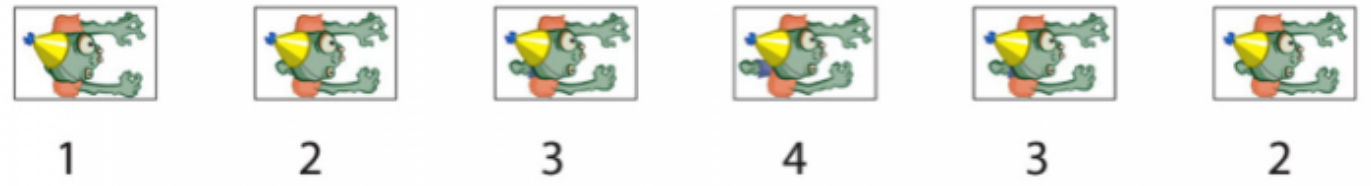
2D Skeletal Animation with Spine Tutorial

Greg Pugh on December 17, 2013

If you've ever made a 2D game and needed to animate your sprites, you likely asked your artist to create separate images for each frame of the animation, like this example from iOS Games by Tutorials:



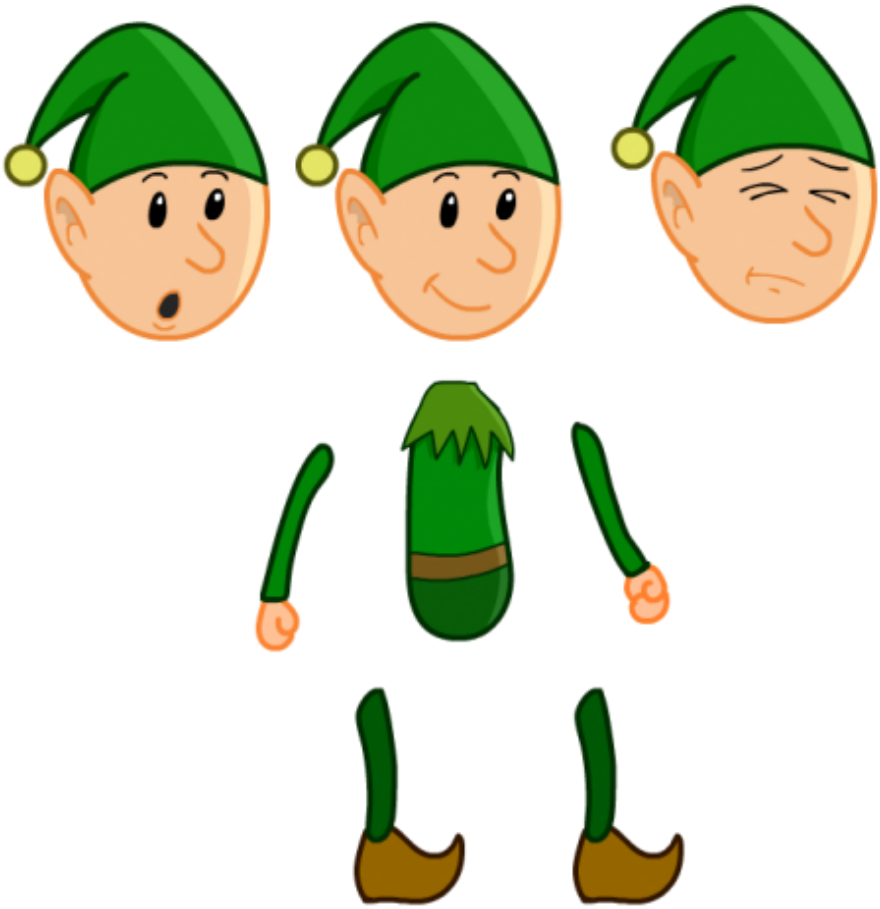
You then probably wrote some code to play through the list of frames quickly, to give the illusion of movement, like you see here:



This method is simple and it works, but it has a number of big disadvantages:

- **High memory and storage requirements.** Because you have to make a separate image for each frame of animation, you are using a lot of memory and storage for your textures. The bigger the sprites are that you are animating, and the more sprites you have, the bigger a problem this becomes. This is a particularly big problem on mobile devices, which only have a limited amount of memory and texture memory.
- **The animations are expensive to make.** Drawing individual animation frames like this is time consuming for your artist. Also, making changes to the animations after they have been completed is very time-consuming.
- **You (probably) cannot make the animations yourself.** Since each frame animation needs to be hand-drawn, if you are a developer this is probably something you need to rely on your artist to do – even if there's a particular effect you're going after.

The way to solve these problems is to integrate something called a **2D Skeletal Animation** system into your games. The idea is instead of saving out each and every frame of animation, instead you save out individual body parts like this:



Then you create a small file that describes how to move the body parts around in order to perform the animation you want, such as walking, running, or jumping. You also add some code into your game to read this animation file, create sprites for each body part, and move them around according to the instructions in the file.

Of course, creating a 2D skeletal animation system by hand is a crazy amount of work. Luckily, the folks at Esoteric Software have created a great tool to help you out called Spine.

Spine is a graphical interface that allows you to create a skeleton out of each pieces of your sprite, and move it around in order to create animations you can use in your game.

Spine also comes with a huge list of pre-made Spine runtimes, which is a fancy way of saying “code you can add into your game to read Spine files, and create animated sprites from them.” Runtimes include Unity, Sprite Kit, cocos2d-iPhone, and much more.

In this tutorial, you’ll use Spine to animate a clumsy elf so that it walks and trips. Along the way, you’ll learn how to:

- Import artwork into Spine.
- Build a skeleton for the elf.
- Create two different animations.
- Save and export your work.

Note that this tutorial does not cover integrating the resulting animations into a game; that will be a separate tutorial. Instead, the focus of this tutorial is using Spine itself, which will be useful no matter what game framework you may be using.

If you’re ready to take your first steps with Spine, let’s get started!

Getting Started

First things first: you need to download and install Spine.

Spine is available for Windows, Mac and Linux. There are five versions of Spine from which you can choose.

- **Trial** *(Free)*: Includes all features, but you cannot save, import or export projects. This version is great for learning the software, but you won’t be able to export your animations into your app.
- **Essential** *($60 – $75 USD)*: Contains the most important features with the ability to save, import and export projects. This version does not include some current features, such as auto-keying, dopesheets and ghosting. It also does not include support for new releases.
- **Professional** *($249 – $299 USD)*: Contains every feature, as well as all future-release features of Spine.

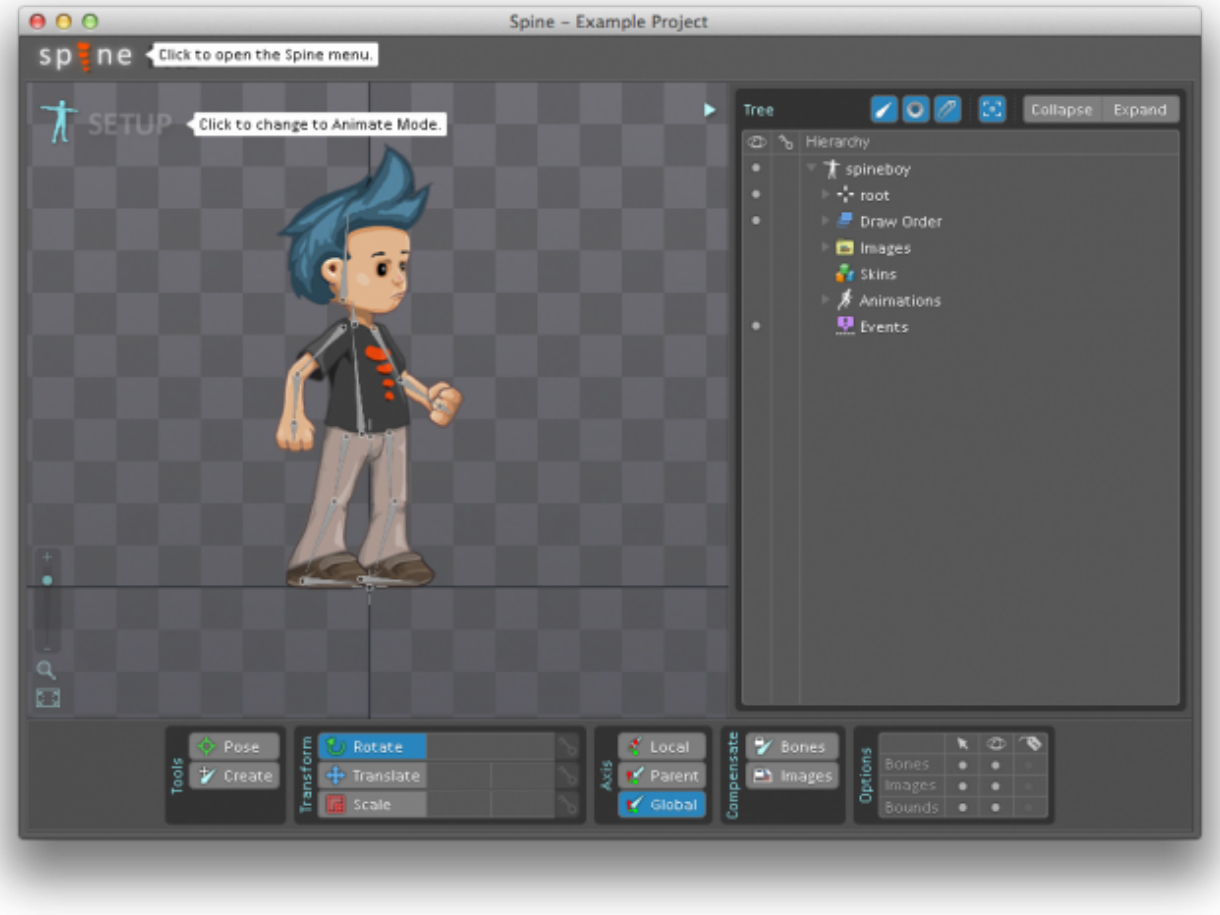
- **Enterprise** *(Base price $2200 USD)*: The same as Professional, but for businesses with $500,000+ of annual revenue.
- **Education** *($610 – $8217 USD + 10% enrollment fee)*: The same as Professional, but for schools and educational institutions. The price of the license depends upon the number of computers supported.

For the purposes of this tutorial, you can do almost everything with the trial version. However, at the end of this tutorial, you'll find an optional section on exporting your animations, which you cannot do with the trial version. If you complete the rest of the tutorial and are eager to see your animations running in your apps, you should consider purchasing an Essential or Professional license so that you have the ability to save and export your work.

So – choose a version of Spine and download, install, and run it. If you are running on the Mac, you may get the following message when you try to run Spine:

Click Continue and you will be directed to an Apple support page. On this page, click the http://xquartz.macosforge.org link in the first paragraph. This will take you the X Quartz download page. Download and install X11, then run Spine again and it should launch with no problems.

Once you successfully run Spine, you'll be greeted with a sample project.



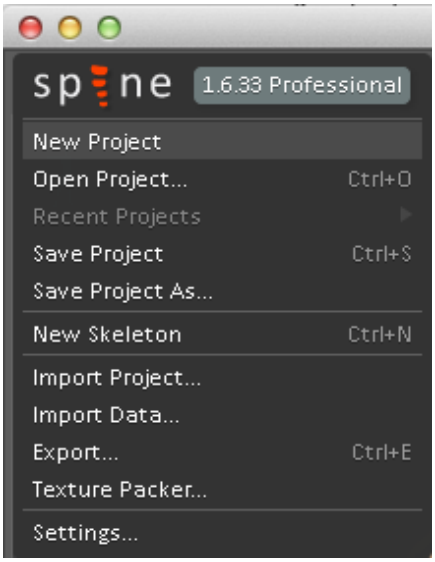
Feel free to peek around the sample project if you'd like. When you're ready, read on to learn how to create your own animation!

Importing Artwork Into Spine

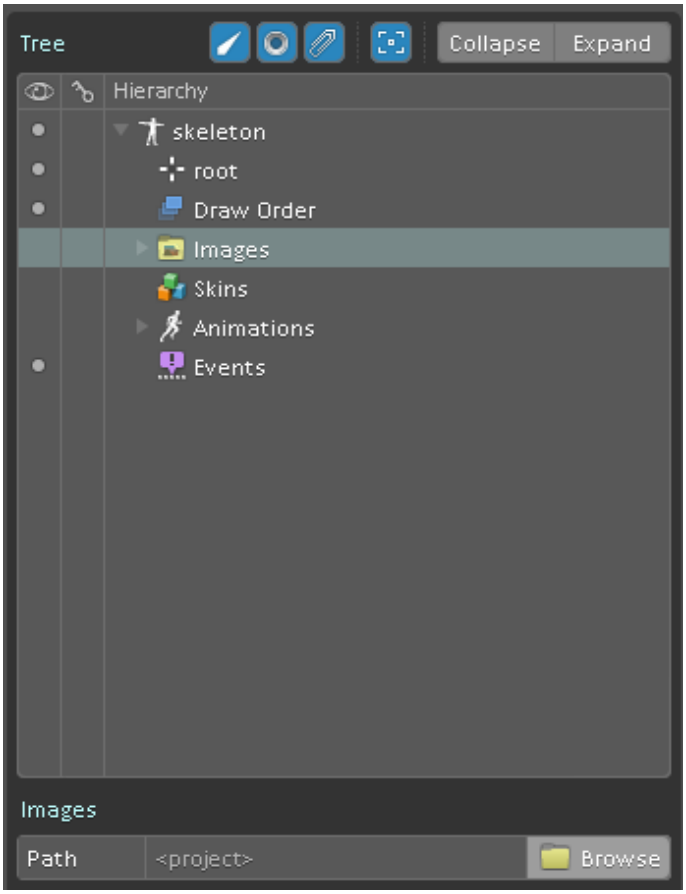
So that you can focus on learning how to use Spine, I've created some artwork for you to create an animated elf.

Download the art here, uncompress the folder and drag it to your Desktop. This will make it easier to find it in Spine.

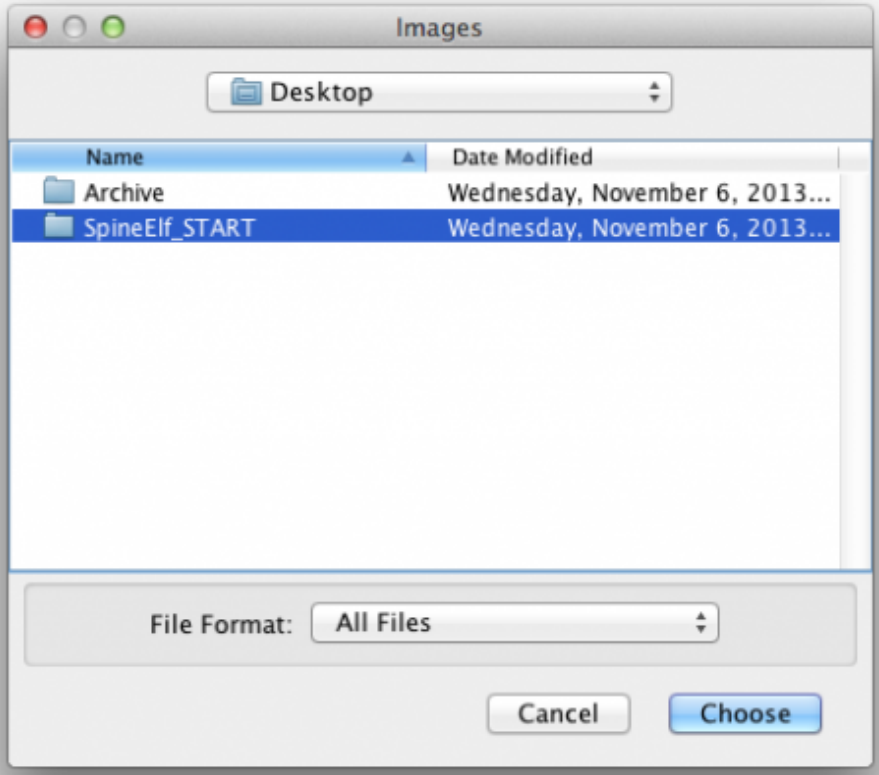
Click on the **Spine logo** in the upper-left corner and select **New Project**.



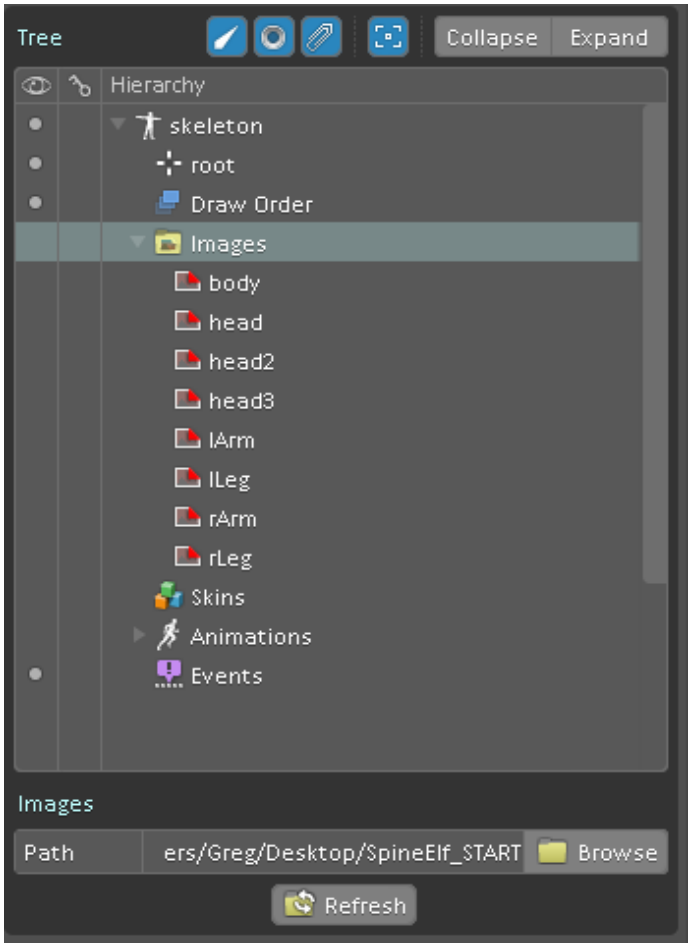
In the **Tree** panel on the right, select the **Images** folder and then click **Browse** under the Images listing.



Browse for the **SpineElf_START** folder on your Desktop, select it and click **Choose**.



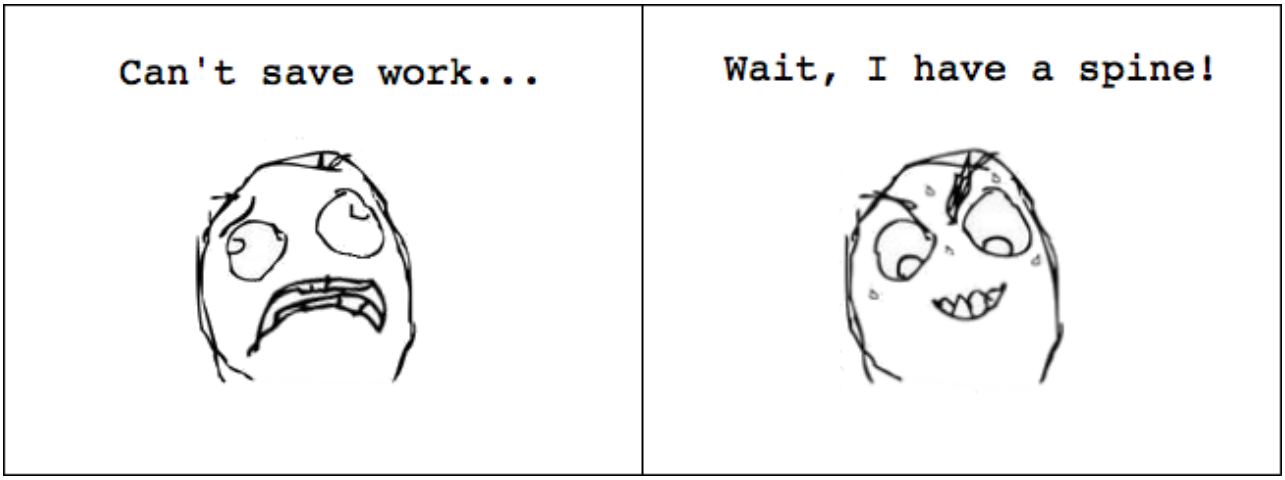
Now your Tree window contains all of the art for your elf in the Images folder.



At this point, you would normally save your project. After all, the Number 1 rule in developing is to save often.

Unfortunately, if you're using the trial version of Spine, you won't be able to save. However, if you've upgraded to the Essential or Professional version, you can **Ctrl+S** or **Cmd+S** now and save your project in the **SpineElf_START** folder.

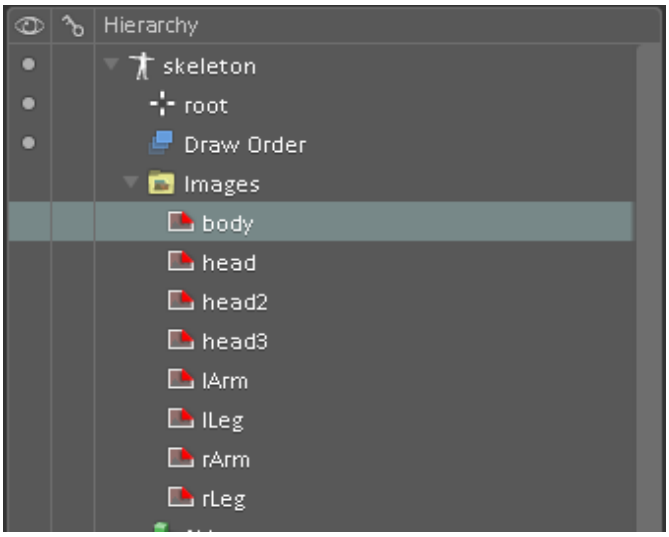
If you're using the trial version, don't fret. You're a developer making your own animations, which means you are bold and adventurous! That Cmd+S hotkey is for the faint of heart, which you certainly are not!



Assembling Your Character

To create your character, you're going to need to enroll in some anatomy and fine art classes at your local university. Just kidding! Since this tutorial provides the art for you, all you need to do is drag and drop the images onto the stage.

Select the **body** label in the **Images** folder and then drag it onto the stage.



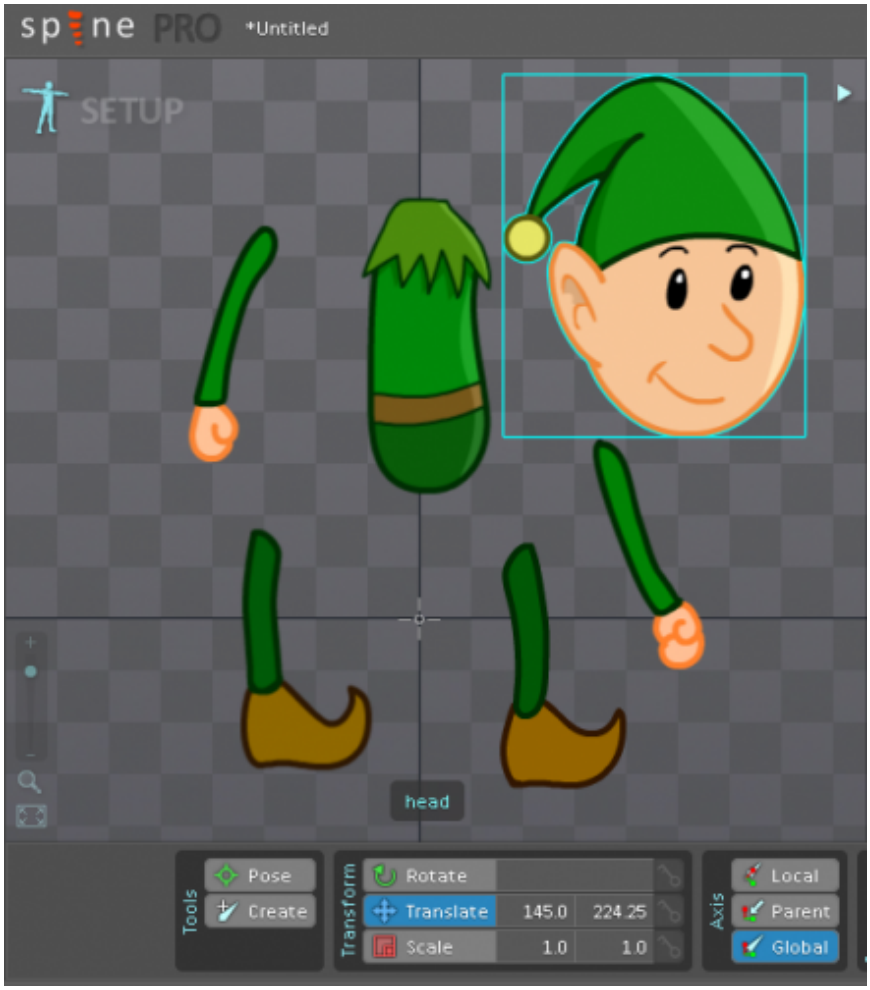
As far as I can tell, there's no way to drag the canvas itself around. So to get the view of the canvas where you want it, you have to zoom out (using the mouse scroll wheel), and then zoom in to the portion of the canvas you want to look at. If anyone finds a better way to do this, let me know. **Update:** @mig_akira pointed out that you can move the canvas by right-clicking somewhere and moving the mouse. Thanks!

Now drag the **head** label onto the stage.

Drag the **lArm**, **lLeg**, **rArm** and **rLeg** labels onto the stage, but *not* **head2** or **head3**.

If you accidentally dragged head2 and/or head3 onto the stage, don't worry. Ctrl+Z or Cmd+Z will undo any mistakes you make. Although you may be bold enough to work without saving, even the toughest of the tough still use the undo hotkey!

Now you need to assemble your elf. You can build him better, stronger and faster. Select the **Translate tool** in the **Transform** toolbar and then select the elf's head.



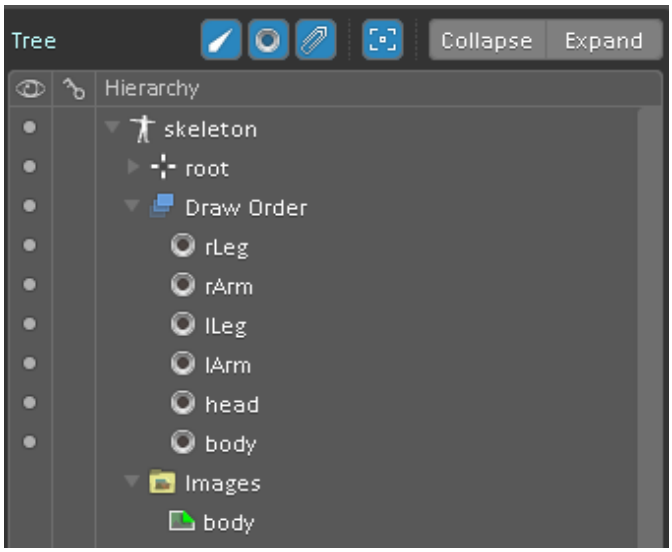
Drag the elf's head to the top of his body. If your stage isn't big enough, you can either use your mouse scroll wheel to zoom out or use the zoom tool on the left side of Spine.

Using the same **Translate tool**, drag the elf's arms and legs to their appropriate positions.

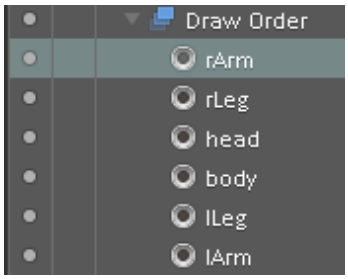
Wait a second… why are his left arm and leg on top of his torso instead of behind it? It looks like you need to adjust the order of body parts.

Changing the Draw Order

Above the **Images** folder, you'll see a listing called **Draw Order**. If you're familiar with Adobe Photoshop or Sketchbook Pro, think of the draw order as layers. The artwork on the top of the list appears on top of the artwork below it.



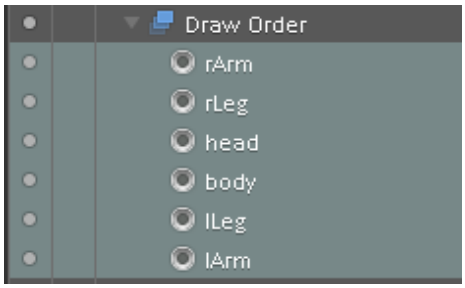
To rearrange the draw order, simply drag and drop a label up or down the list. Rearrange the order from top to bottom to be like this: *rArm, rLeg, head, body, lLeg and lArm*.



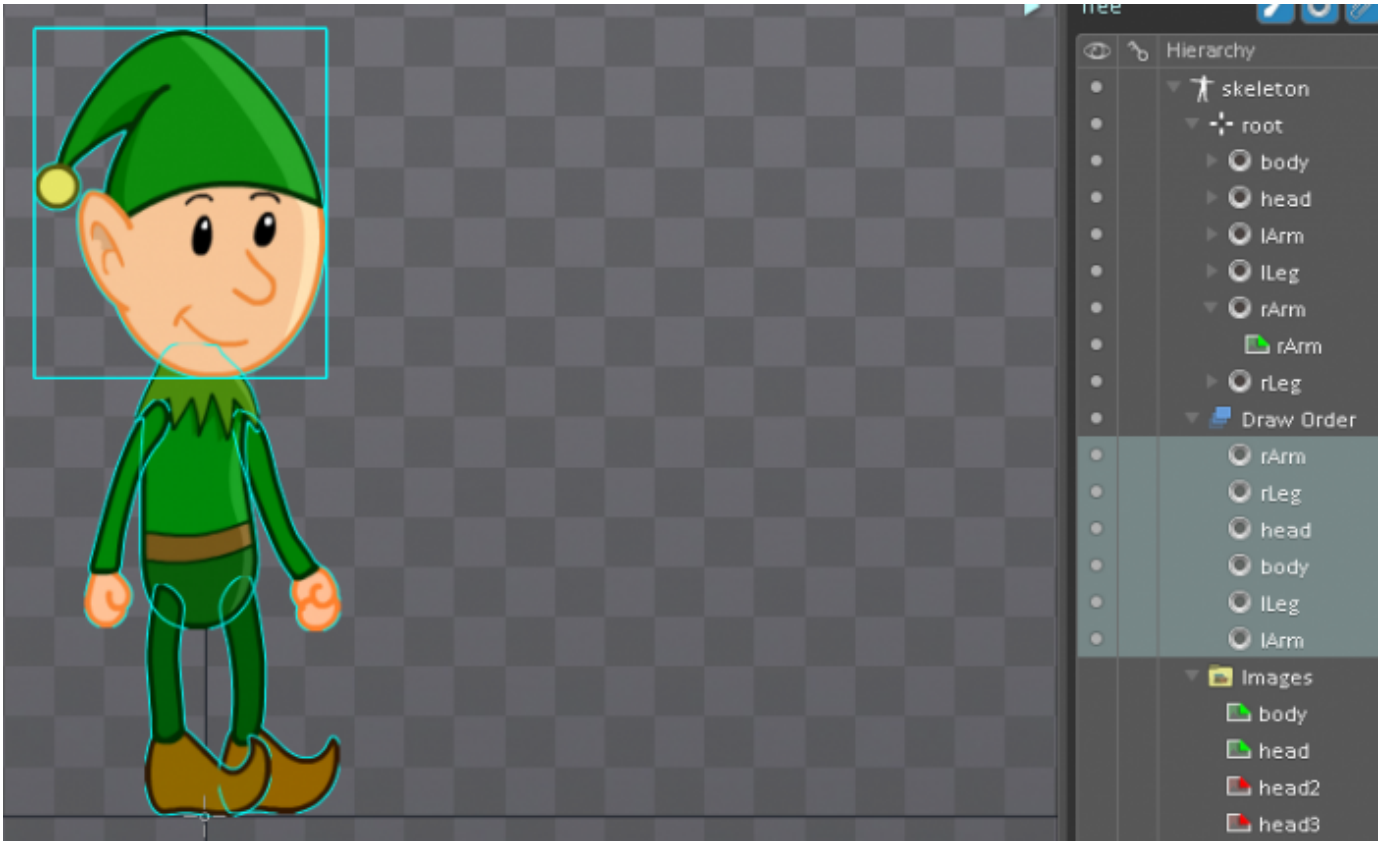
Your elf should now look like this:

Now that's a good-looking elf! The final step in setting up your elf is to align his feet with the horizon line in Spine. You can do this by moving each body part one-by-one—*or* you can select everything and do it in one swoop, which is much easier.

Select all of the elf's body parts in the **Draw Order** folder by **Shift+clicking**.



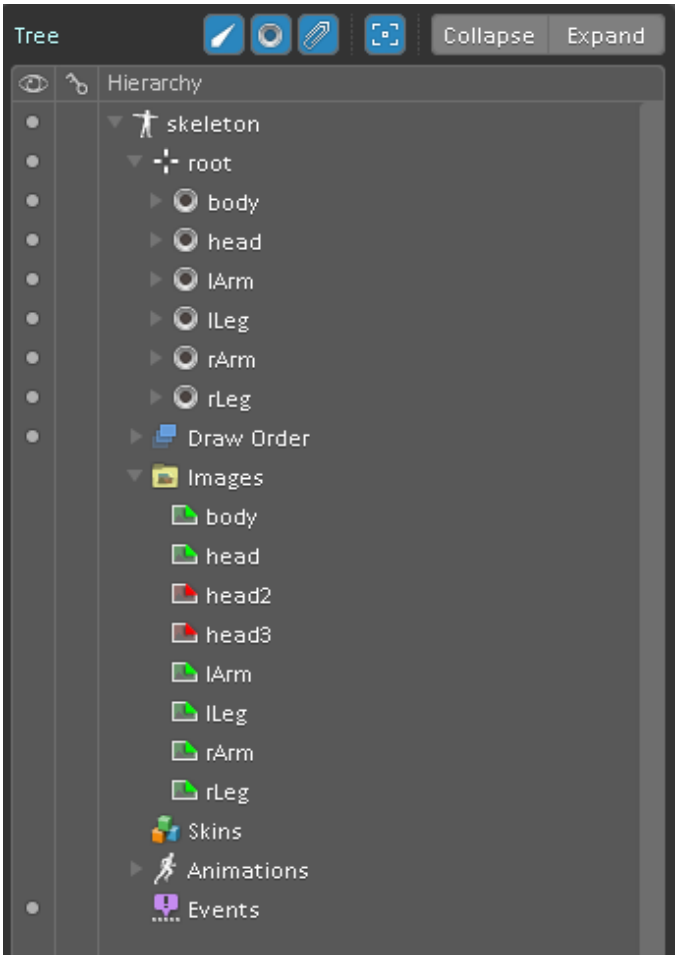
While still using the **Translate tool**, drag the elf so that he's standing right on the horizon line.



You might be wondering what you're supposed to do with those other two head images. After all, **head2** and **head3** have just been sitting there, patiently waiting for you to use them.

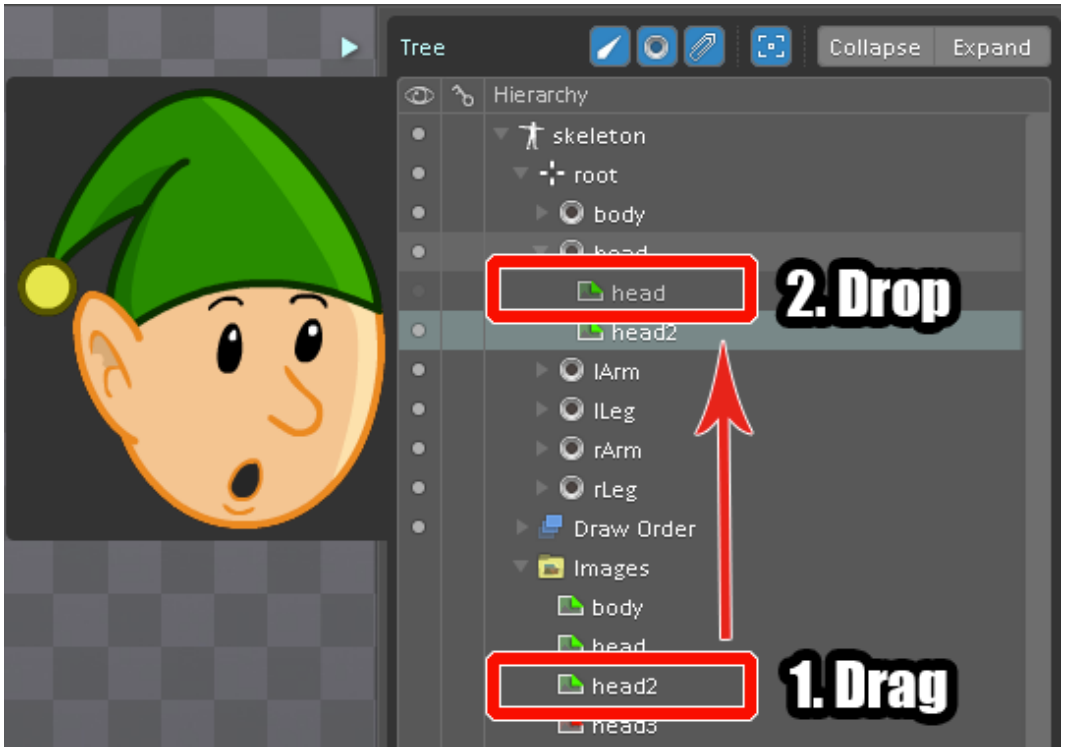
Multiple Images for One Body Part

Above the **Draw Order** folder, there is a listing for **root**. Click the drop-down arrow next to **root** and you'll see all of the body parts listed.



Click on the drop-down arrow next to **head** and you'll see the image of the head that is attached to this body part.

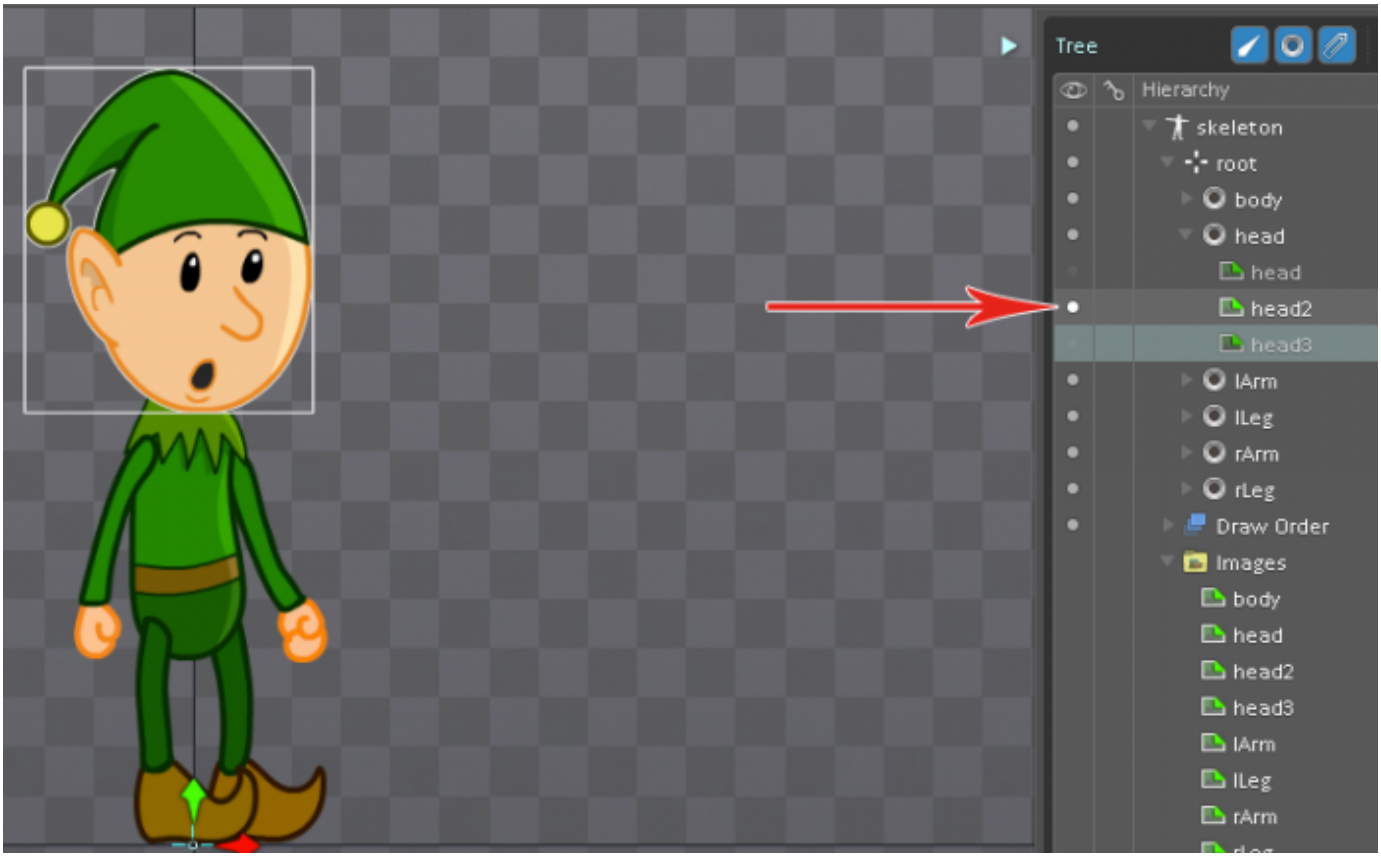
You can add multiple images to each body part and switch between them to animate your character. Drag **head2** from the images folder and drop it under **head** in the root listing.



Note that when you drag **head2** on to the canvas, it might default to the origin. If that happens, just move the head back to where it belongs.

Do the same for **head3**.

If you want to toggle between the elf's different faces, click the dots under the eye in the Tree panel.

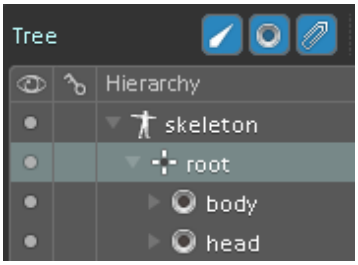


Now that you're using all of the artwork, you can build your elf's bones!

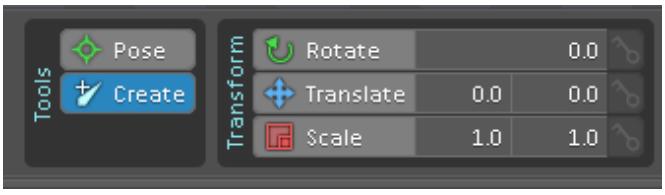
Bone Up!: Adding a Skeleton

It's time to give your elf some bones. How else is he going to move if he doesn't have a skeleton?

In the Tree window, select the **root** listing.



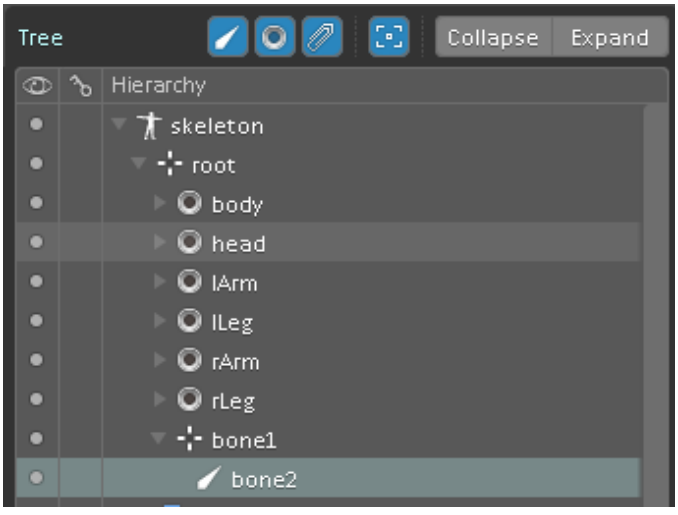
Then select the **Create tool** from the Tools window at the bottom of Spine.



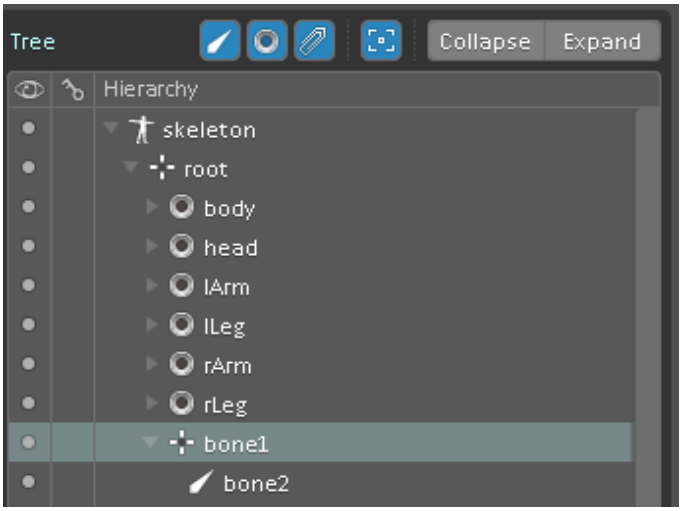
Click on the middle of the elf's chest. This creates a new bone called **bone1** (or maybe just **bone**).

Now click and drag from the bottom of the elf's head to his hat. This creates a new joint where his neck would be.

The attached bone is called **bone2** and appears under **bone1** in the Tree window because **bone2** is a child of **bone1**. That means if you were to move bone1, bone2 and any other children of bone1 would also move.

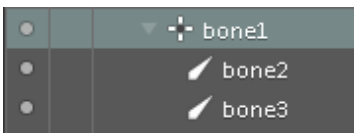


In the Tree window, select **bone1**. This will make the next bone you create also a child of bone1.



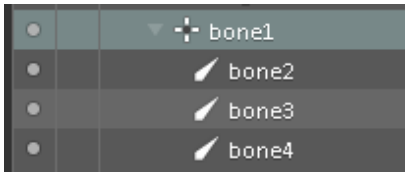
Click and drag from the point where the elf's left arm meets his torso down to his elbow to create **bone3**.

Repeat the same procedure for the elf's right arm and both legs. First, click **bone1** in the Tree window.

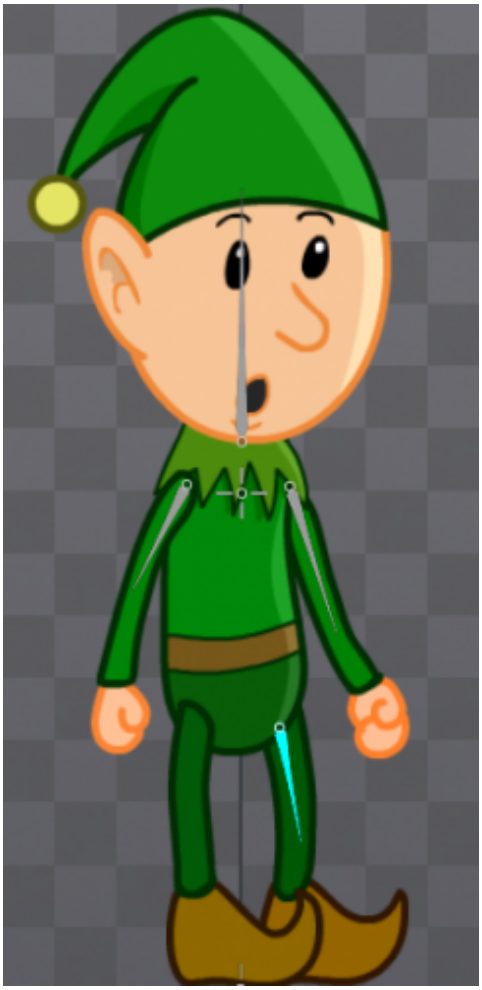


Then click the point where the elf's right arm meets his torso and drag down to his right elbow.

Go back and click **bone1** in the Tree window again.



Next, click the point where his left leg meets his body and drag down to his knee.



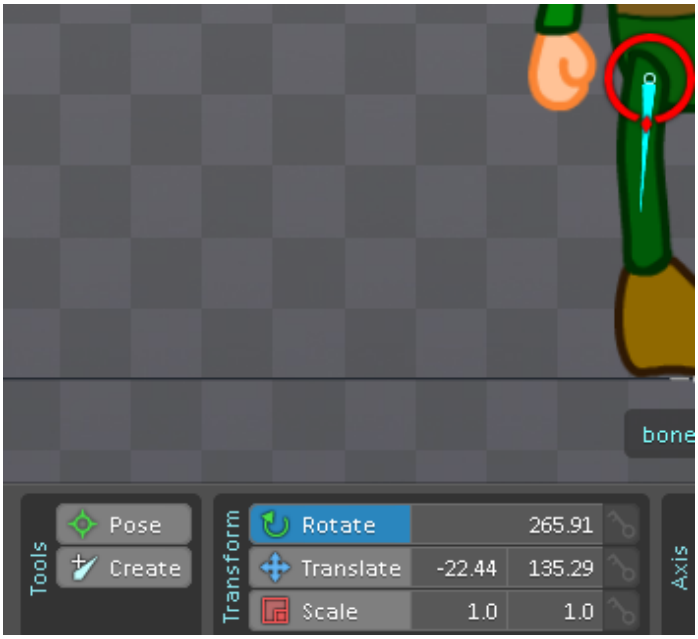
Do the same for the elf's right leg.

The elf's skeleton is now complete. The head bone's connected to the… body bone! The arm bone's connected to the… body bone! The leg bone's connected to the… body bone! And that's the way it goes.

> **Note:** You can create more complex skeletons depending on your character's needs. You can have bones for shoulders, elbows, wrists, ankles, tails and even clothing. If you were to add an upper arm and forearm, you'd want to parent the forearm to the upper arm and the upper arm to the torso. That way, all pieces of the arm would be tied together.

Attaching the Bones to the Body

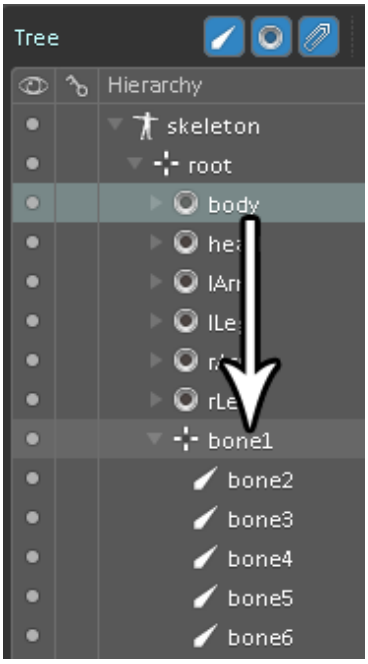
Now you've got images that are pieced together to look like an elf and a skeleton that can fit inside the elf, but they're not actually attached to each other. You don't believe me? Select the **Rotate tool** and then click on any of the skeleton's bones.



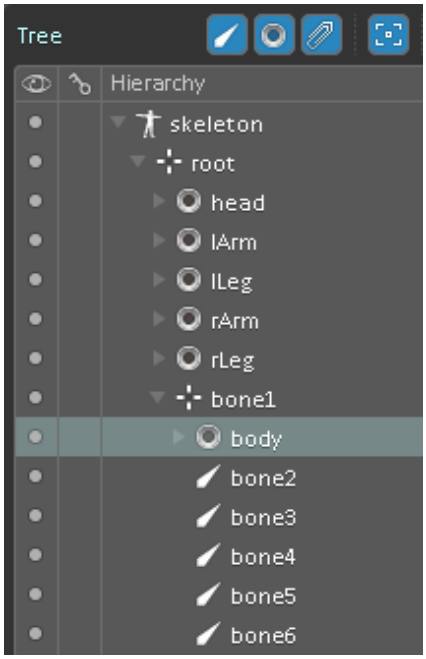
Click anywhere on the stage and drag. The bones rotate, but the elf doesn't move. D'oh!

Hit Ctrl+Z or Cmd+Z to undo the bone rotation and look at the Tree window. You'll see that the images and bones are in different lists—that's why they're not paired.

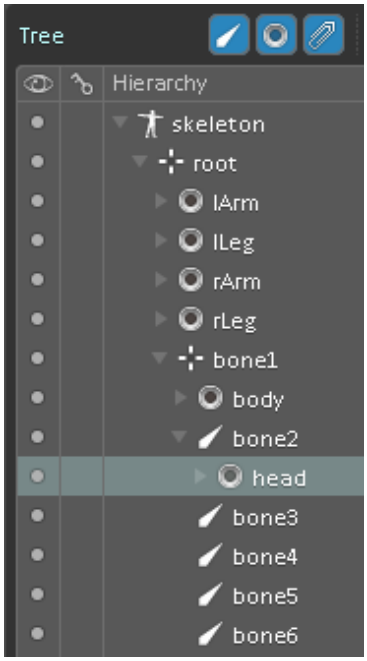
To pair them, you'll have to—you guessed it: drag and drop! Click on the **body** image in the Tree window and drag it down to **bone1**.



Notice how **body** is now listed under **bone1**? The body bone and the body image are now married and can function as one. Awww!



Drag the **head** image down to **bone2** to attach the head bone to the elf's head.



If you're ever confused by the hierarchy structure in the Tree window, an easy way to tell if the body parts have been properly attached is to test them. Select the rotate tool like you did before and then select the skeleton bone. Click and drag on the stage to see if the image moves when you move the skeleton. You can always undo any mistakes by hitting Ctrl+Z or Cmd+Z.

Drag and drop the following:

- **lArm** to **bone3**
- **rArm** to **bone4**
- **lLeg** to **bone5**
- **rLeg** to **bone6**

Your elf now has a fully functioning skeleton! And just think—all you've done so far is drag and drop. Next you're going to move onto animating your elf. This simply requires more dragging and dropping. Who would have guessed?

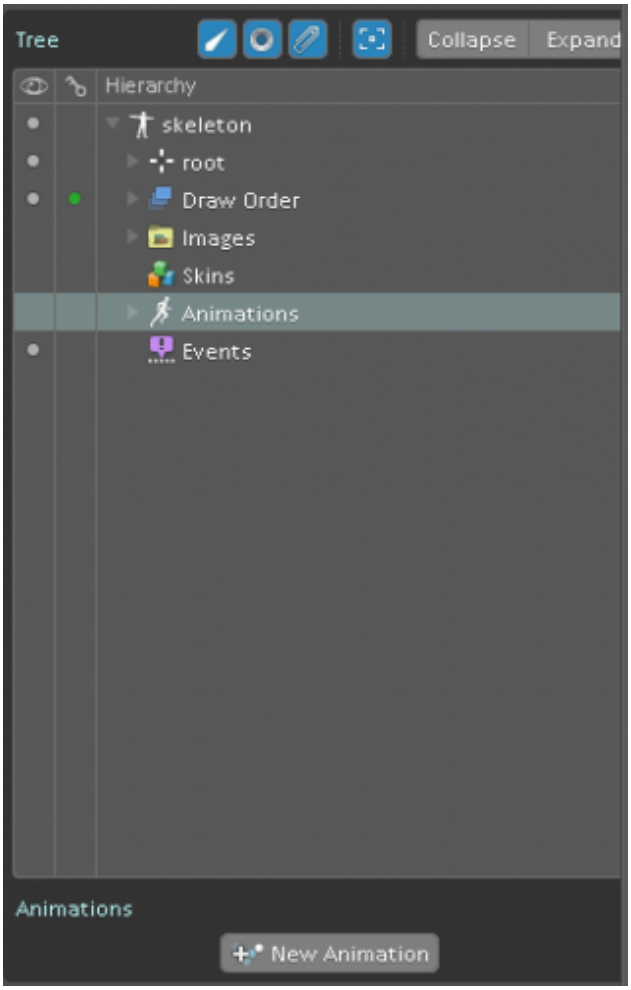
A Standing Animation

The first animation you'll create is one of the elf standing. You might be asking yourself, "Isn't he already standing? That doesn't require animation!"

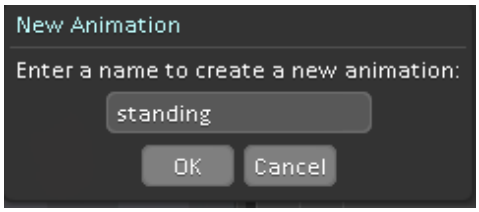
True, Santa's little helper is already standing, but he's not doing anything. That makes him a pretty boring subject, but you can give him some subtle movements while he's standing in place. That will make for a more interesting game.

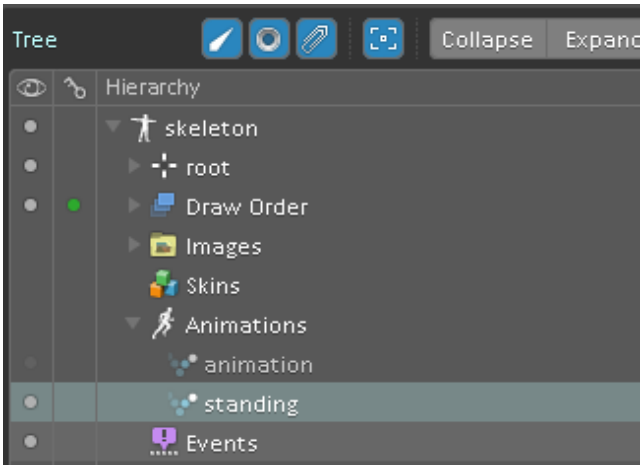
To switch to the Animate mode, click the word **SETUP** in the upper-left of Spine. This brings up a timeline at the bottom of the screen.

In the Tree window, click on **Animations** and then on **New Animation**.



Name the new animation **standing**.





Assuming you're using the trial version of Spine, you have access to advanced features that are available in the Professional version that aren't included in the Essential version—including the Dopesheet and Auto Keying.

Using the Dopesheet and Auto Keying

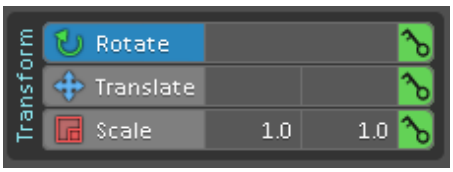
Think of the Dopesheet as a more advanced timeline on which your animation will play. And Auto Key lets Spine set the **keyframes** for you when you animate your character. But what are keyframes, you ask?

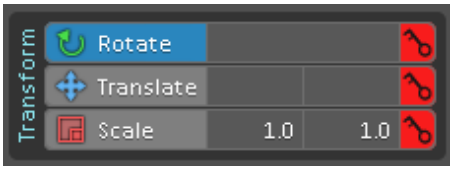
Keyframes are an animation's most important frames. If you wanted to animate a ball rolling from the left to the right, you'd need one keyframe for the ball on the left and one keyframe for the ball on the right. The frames between the keyframes are called **in-betweens**, also referred to as "tweens". Spine creates the in-betweens for you and Auto Key will help you set the keyframes. Pretty sweet!

Click on the **Dopesheet** and **Auto Key** buttons at the bottom of Spine.

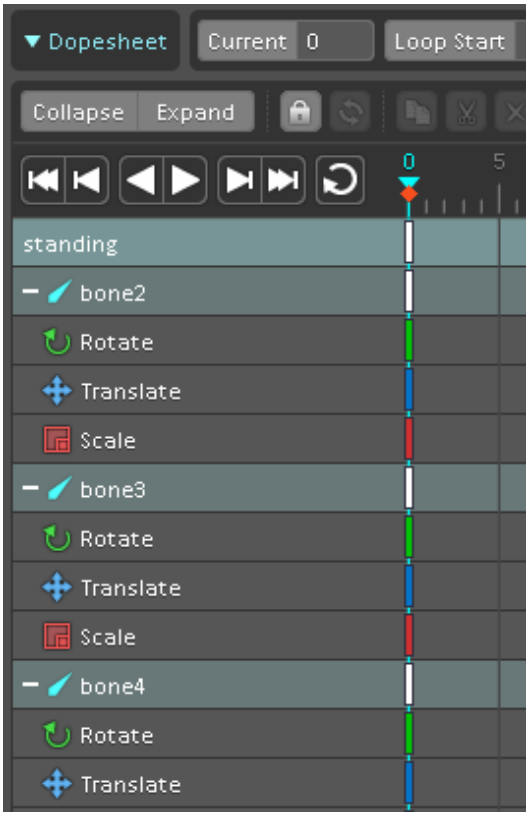
Hold down the **Cmd** or **Ctrl** key and click on the **left arm**, **right arm** and **head** bones of the elf's skeleton.

In the **Transform** window, there are three green **key icons**. Click on each key once to turn it red.



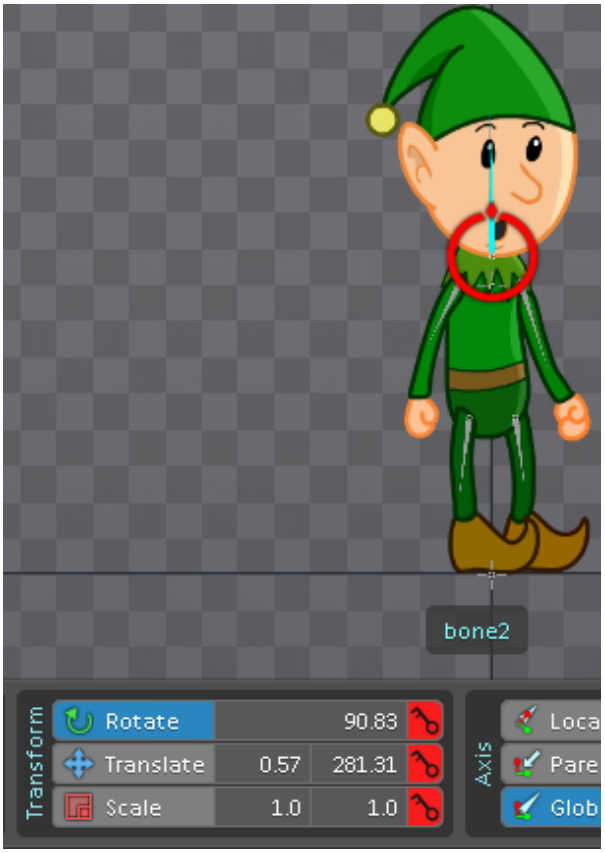


This simply sets the initial keyframes for the elf's arms and head, which you will see in the Dopesheet.

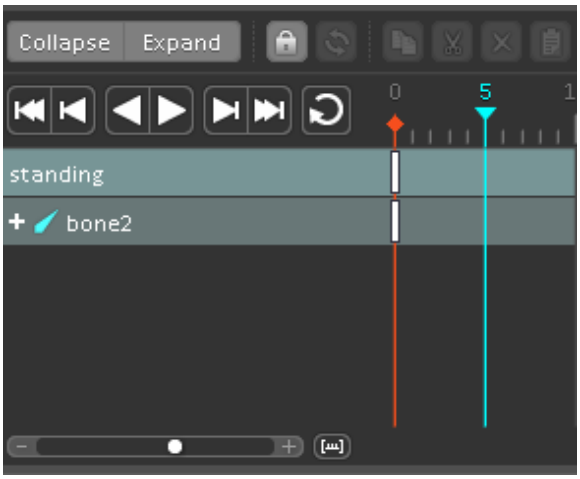


You won't need to set keyframes on the elf's legs in this animation, since he'll be standing still. Also, since you've enabled Auto Key, that was the last time you'll have to click on the key icons. Spine will do it automatically for the rest of the standing animation.

Select the **Rotate tool** if it's not already selected, and then click on the elf's **head bone** in the skeleton.

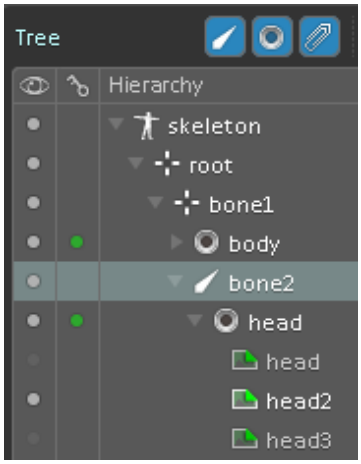


On the timeline in the Dopesheet, click on the mark for **frame 5**. To keep things simple, you'll animate everything by increments of 5.

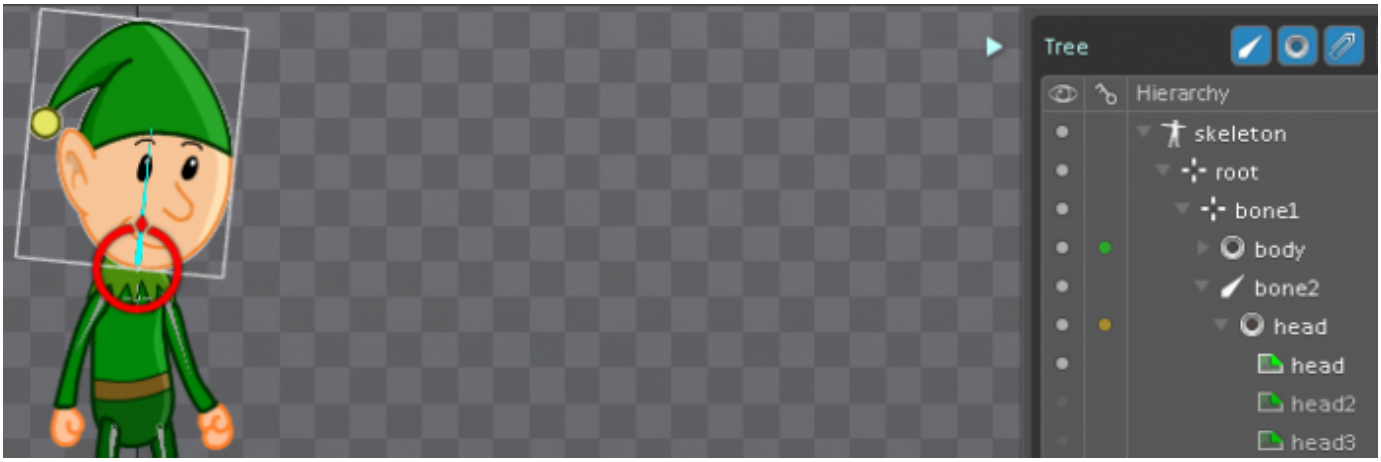


Now click and drag on the stage to move the elf's head forward *slightly*. Subtlety is key here, unless you want him to look very cartoony. Since you've enabled Auto Key, Spine makes a new keyframe for you on the 5th frame.

You can also change his facial expression here. In the Tree window, navigate to the **head** image under **bone2** and expand the list by clicking the corresponding arrow icon.



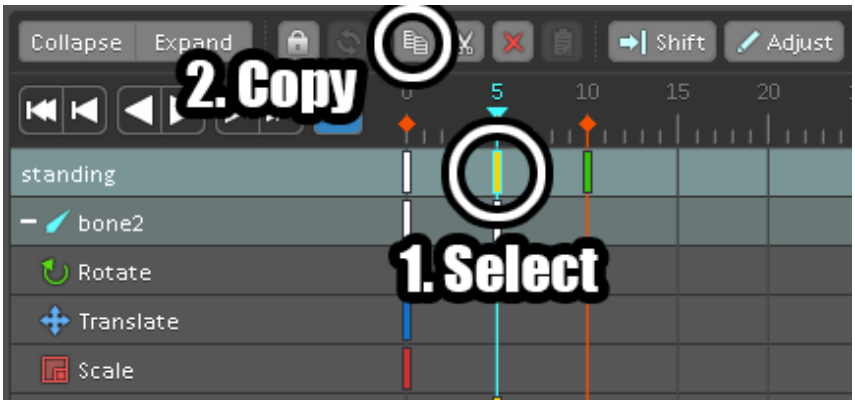
Click on the dot underneath the eye icon next to **head** to display the image of the elf smiling.



If you see a red dot next to the head listing under the key icon, you're good to go. But if you see a yellow dot instead, this is to show you that you've made an uncommitted change. Click the yellow dot to turn it red, which sets a keyframe for the image swap.

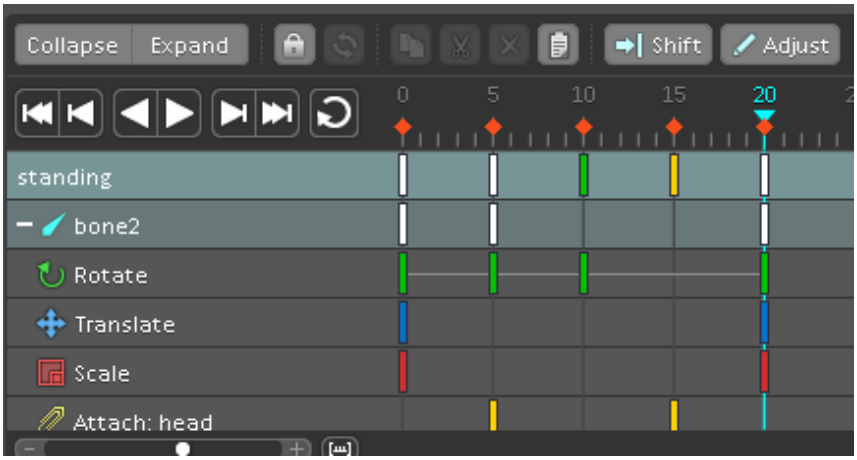
Click on **frame 10** in the Dopesheet timeline and move the elf's head slightly forward again by clicking and dragging on the stage.

To speed up the animating process, you can also copy and paste keyframes. With the head bone still selected, look in the timeline and click on the **white rectangle on frame 5** in the **standing** row. Then click the **copy** button.



Click on **frame 15** and then click the **paste** button.

Now select **frame 0**, click **copy**, select **frame 20** and then click **paste**.



In the playback controls, click the **loop** button and then **play**. Your elf is now bobbing his head back and forth.

Note: If you want to experiment further, try changing the elf's head on different keyframes. Remember to select the frame you want, pick the different head image in the Tree window and then click the yellow dot to turn it red to set the keyframe.

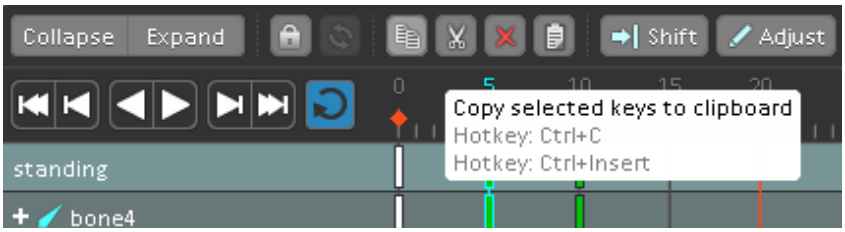
Completing the Animation

Now onto the arms! Select **frame 0** and then select the elf's **right arm bone**. Then, simply follow the same steps that you used to animate his head.

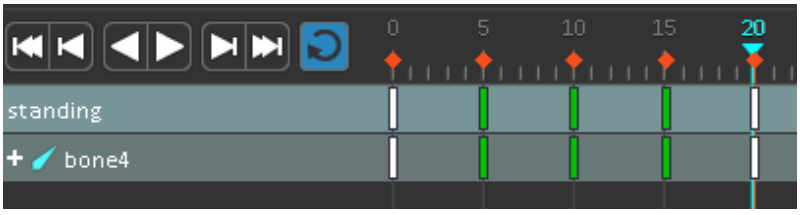
Select **frame 5** and rotate his right arm slightly outward.

Select **frame 10** and move it slightly outward again.

Click the **white rectangle on frame 5** in the **standing row** and then click **copy**. **Paste** it on **frame 15**.



Click the **white rectangle on frame 0** in the **standing row**, click **copy** and paste it on **frame 20**.



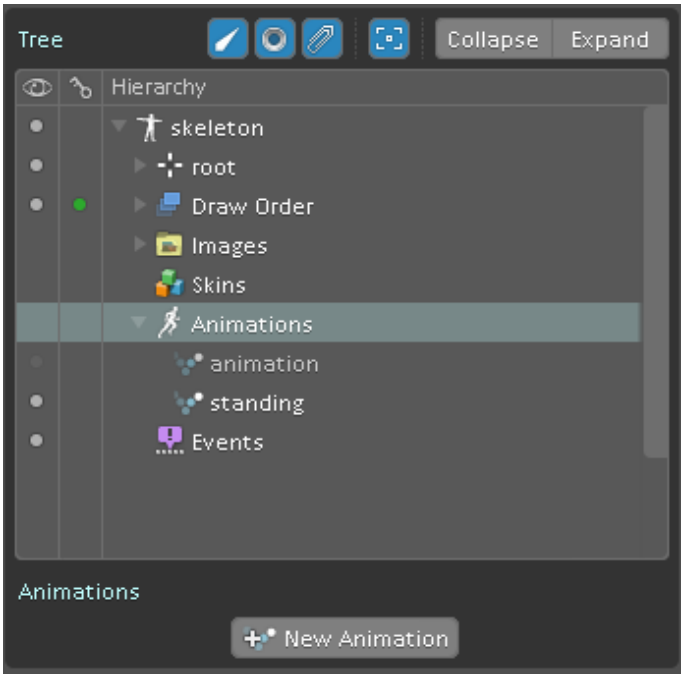
Repeat the same steps for the **left arm** and then click **play** to see the results. It's a fully animated elf!

A Walking and Tripping Animation

If you're new to animation, what you just did may have seemed like a lot of work. In actuality, all it took to animate the elf was to select a frame, move a body part, select a frame, move a body part and then copy and paste. In the traditional days of animation, what you just did could have taken at least a day to complete.

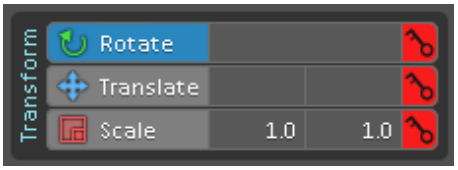
Now you'll create a new animation where the elf will take a couple of steps and fall to the ground. Since you're well on your way to becoming a professional animator, these steps will seem familiar and go quickly.

In the Tree window, click on **Animations**, then on **New Animation** and name it **walking**.



You've created a brand new animation file, so Spine has reset the elf to his default position. Select **frame 0** in the timeline and then in the Tree window, **Shift+select** all of the bones.

Click the green **key icons** in the **Transform** window to turn them red. This sets the initial keyframe.



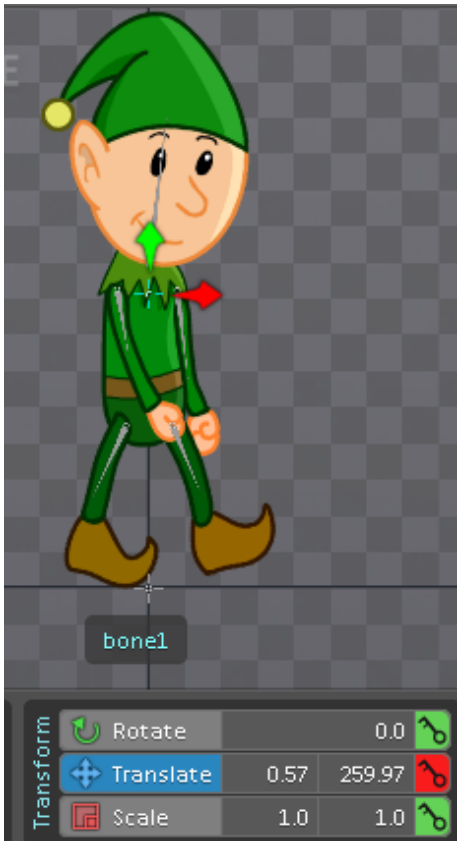
First He Walks...

Select **frame 5**, then select the elf's **left leg bone** and rotate it forward slightly. Then select his **right arm bone** and rotate that forward slightly. When humans (and elves) walk, they alternate opposing arm and leg movement, so make sure you alternate opposing arms and legs.

Select **frame 10** in the timeline and then rotate both the elf's **left leg** and **right arm** forward a bit more. Rotate the elf's **right leg** and **left arm** backward slightly and his head forward.

Select **frame 15** and begin slowly reversing the animation by rotating his left leg backward, right leg forward and so forth.

If you're having trouble keeping the elf's feet level with the horizon line, select his **body bone** and then select the **Translate tool** to move his entire body. This is why you made the torso the parent for all of the other bones earlier in the tutorial.



...Then He Trips!

Start the tripping motion on **frame 20**. When someone trips, their feet get caught up behind them, their arms go forward and their head leans backwards. Begin to simulate that movement with your elf.

Now is also the time to swap out the head image for **head2**. Remember to bring up **head2** in the Tree window, and then click the yellow dot to turn it red next to **head**.

On **frame 25**, use the **Translate tool** to select the **body bone** to raise the elf off the ground. Switch to the **Rotate tool** and rotate his entire body to exaggerate the tripping motion. Continue with the rotation of the arms, legs and neck.

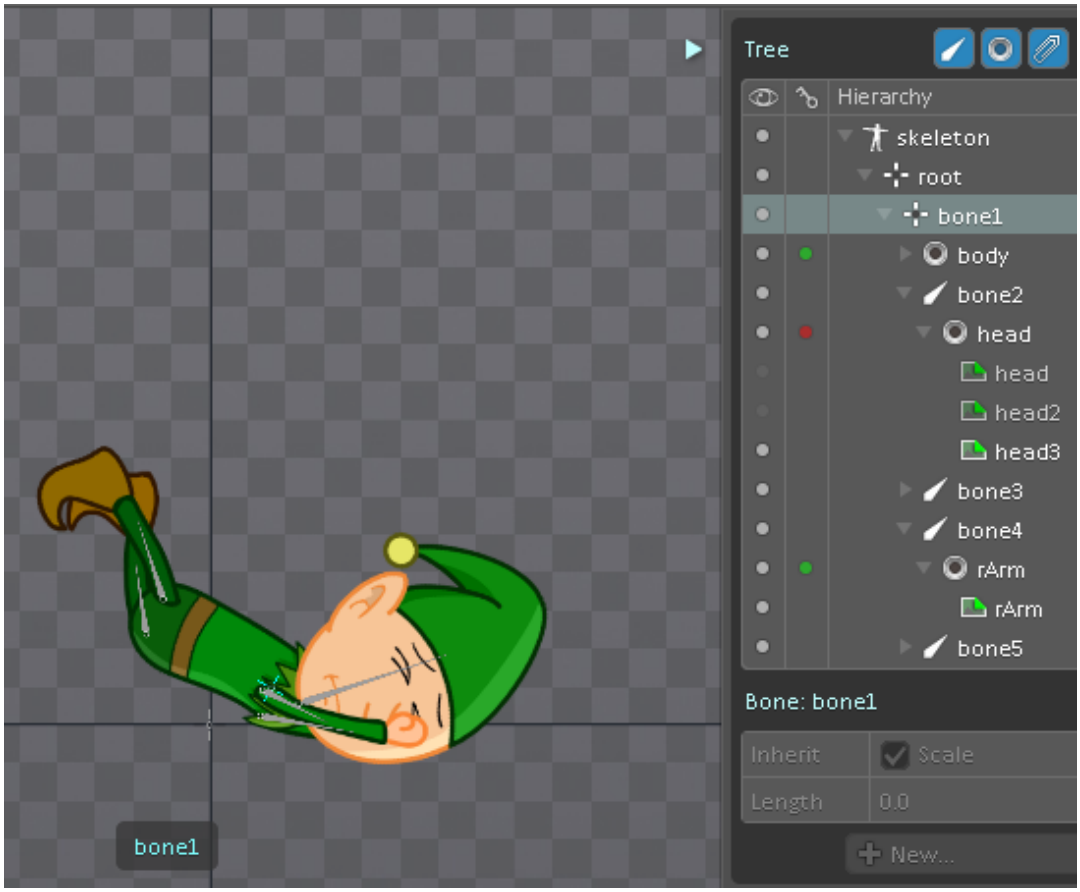
If at any point you notice a limb starting to pop out, use the **Translate tool** to shift it back behind the body.

By **frame 30**, you can really start to get the elf airborne and flying like Superman.

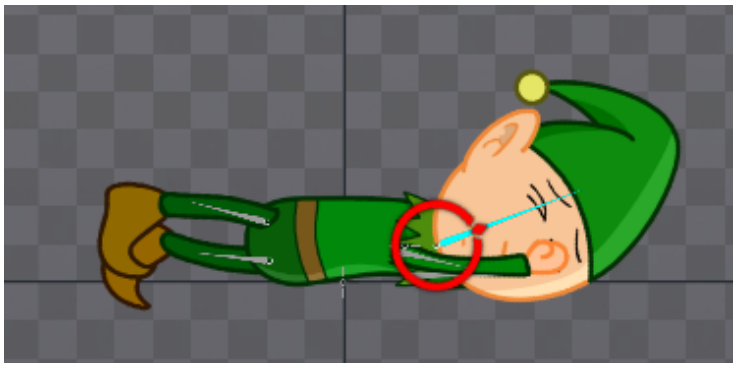
On **frame 35**, begin the downward motion of the elf falling back to the ground.

On **frame 40**, make the elf begin his initial impact with the ground.

Change the elf's head to **head3** via the Tree window on **frame 45** to really sell that ground impact.



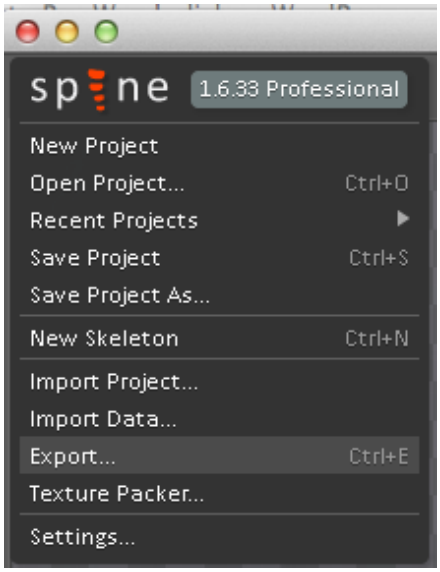
On **frame 50**, make your elf lie face-first on the ground. Now you have the chance to add some fine details to enhance the animation's effect. When someone hits their face on the ground, their head bounces slightly. Animate this by going to **frame 51** and rotating the **head bone** up slightly, and then moving it back down by **frame 53**.



And there you have it! You've created an animation of an elf standing and an animation of an elf doing a face plant. If at anytime you want to switch between the animations, simply click on the circle under the eye icon in the **Animations** listing in the Tree window.

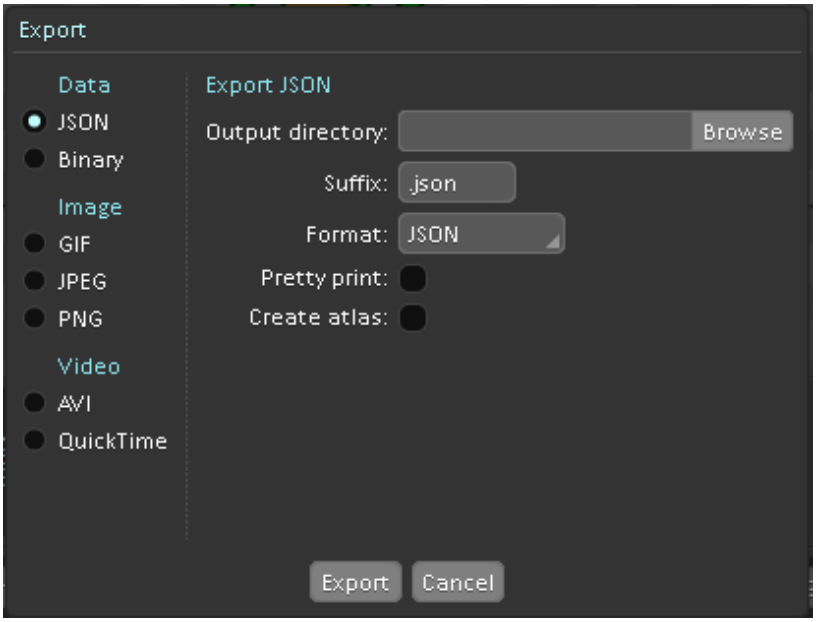
Optional: Exporting Your Work

If you've decided to upgrade your Spine license from the trial version, you have the ability to export your animations. To do this, first click on the **Spine** logo in the upper-left and choose **Export**.



Here you probably want the JSON option – this creates a nice file that describes the animation concisely that the Spine runtimes know how to read. Save the file as elf.json, in the same SpineElf_START folder on your desktop.

If you're unsure how to implement the animations in your app, have a look at the runtimes you can use.



Where to Go From Here?

This tutorial is a very basic example of what you can do with Spine. Please experiment with adding keyframes in other increments to change the timing, adding different artwork, building more complex skeletons, animating logos and anything else you can think of.

I've created a slightly more complex version of the elf tripping, which you can download here. If you don't have an Essential or Professional license, you won't be able to open the .spine project file, but you will be able to use the JSON file I've included to import it into your game.

If you're interested in animation, it is definitely worth checking out the book *The Animator's Survival Kit* by Richard Williams. Every animator I know has a copy of this book in his or her studio, as it's the go-to guide for animation. If you're not one for paperback books, they've also converted it into an iPad app.

Finally, if you enjoyed this tutorial, stay tuned for an upcoming tutorial by our own Ray Wenderlich on how to integrate your animations into a Sprite Kit game!

If you have any questions, comments or suggestions, feel free to join the discussion below!

Greg Pugh

Greg is a mobile app developer, animator, and Flash developer for his company GP Animations. He also writes and illustrates children's books both in print and for apps. When Greg isn't writing children's books, he's playing guitar, watching hockey, and enjoying a dram of scotch with a cigar.