This is a guide for creating a super simple custom VRChat avatar from scratch. The focus of this guide will be for creating avatars that work in both Oculus Quest and for PC, with a focus on Quest (since Quest avatars work on PC, but not vice versa).

This guide assumes that you (like me) are a complete noob to Blender and Unity. So, I've written this to try to be as newbie-friendly as possible.

Prereqs:

To get started, you'll need to install the following software:

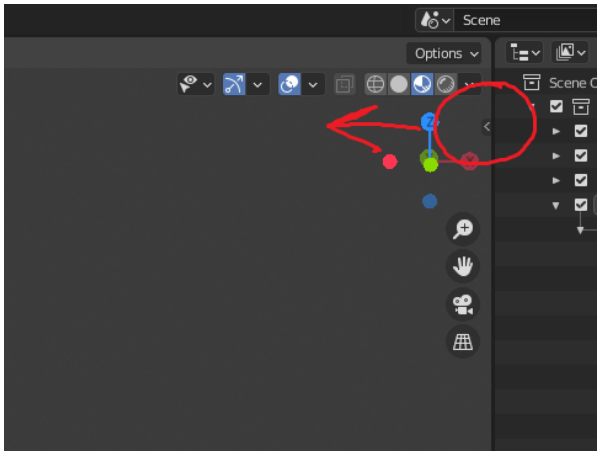
- Blender - https://www.blender.org/ (3D Modelling software. See **Setting up Blender**)
- Blender CATS plugin - https://github.com/GiveMeAllYourCats/cats-blender-plugin (Provides some shortcuts for exporting models to VRChat. Don't Unzip it!)
- Unity + Unity Hub - https://unity3d.com/get-unity/download (VRChat is based on Unity. Unity Hub is a package manager for Unity that makes it easy to make sure you have all the right libraries and versions installed. See **Installing Unity**)
- VRChat SDK for Unity - https://vrchat.com/home/login (Go with v3.0)

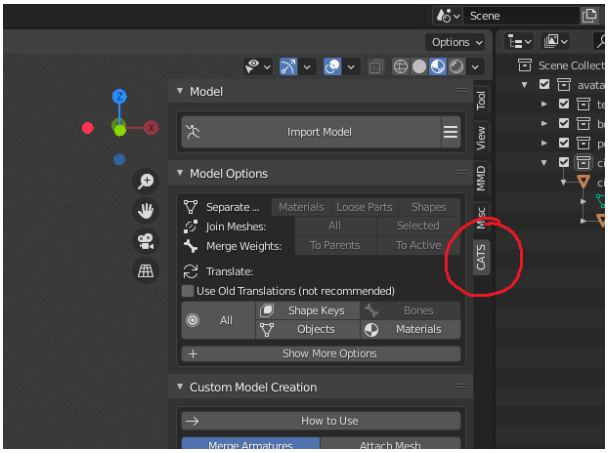
Setting up Blender

Before you start using blender, you'll need to install the **CATS** add-on. Here's how you do that (w/ Blender 2.9):

1. From the top menu bar, select **Edit** > **Preferences** > **Add-Ons** > **Install**
2. Find the zip file for your CATS plugin, and click Install Add-on.
3. When the add-on finishes installing, be sure to click the checkbox next to it to enable it.

You can access the CATS plugin from the 3D viewport's sidebar. It was not obvious from the GitHub README for CATS how to do this at first (because they were using an earlier version of Blender), so here's some screenshots:





Certain camera controls in Blender rely on you to have a mouse with a middle click button. If you don't have this, you should turn on emulation for the middle mouse button. Here's how you do that:

1. From the top menu bar, select Edit > Preferences > Input
2. Check the box next to "**Emulate 3 Button Mouse**".

Now you can rotate the camera around the scene using **Alt + Left Mouse Button** and you can pan around the scene using **Alt + Shift + Left Mouse Button**.

Installing Unity

You'll want to get Unity set up for creating VRChat avatars for **Oculus Quest**. Specifically, VRChat requires Unity version **2018.4.20f1**. Please do not try to use another version of Unity, even a newer one!

To install Unity, you'll first need to install **Unity Hub**, which acts as a package and version manager for Unity. Once you've got Unity Hub installed, you might notice that version 2018.4.20f1 is not available from the list of available install packages. Don't worry! Just use this link (copypasta it to your browser and go) and it'll install the version VRChat needs through Unity Hub:

unityhub://2018.4.20f1/008688490035

When you are installing Unity through Hub, it'll ask you if you want to install any additional modules. Since we are going to be making avatars for Quest, be sure to check the boxes next to the **Android SDK** modules.

When you first start up Unity, it'll ask you to make an account and select a license. The free "Personal" license is fine.

Getting started w/ Blender

Before we start creating avatars from scratch or from modifying base models (and using CATS to make them VRChat-conformant), let's first learn the basics of using Blender. An in-depth manual for using Blender is available at https://docs.blender.org/manual/en/latest/index.html, but that is more of a reference guide and not so useful as a tutorial for when you're just getting started. It will be important that you learn several hotkeys for Blender to work as efficiently with it as possible.

In this guide when I reference mouse controls, I will use the following abbreviations:

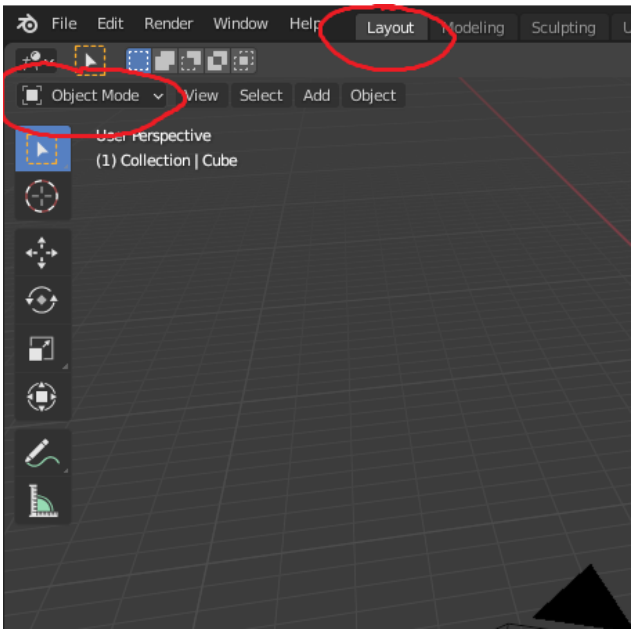
LMB - Left Mouse Button

MMB - Middle Mouse Button

RMB - Right Mouse Button

Starting up

When you first start up Blender, you should see a scene in the 3D viewport with a 2m cube, a camera, and a light. You don't need these. Delete them by pressing **A** (select All) and then **Delete**.



Note also that the 3D viewport is started up in the **Layout** workspace and **Object** mode.

Go ahead and switch to the **Modelling** workspace. As far as I can tell so far, Layout workspace just has some extra tools for animation, which we won't need.

The only other workspace you'll probably use is the **UV Editing** workspace for setting up your textures.

Camera controls

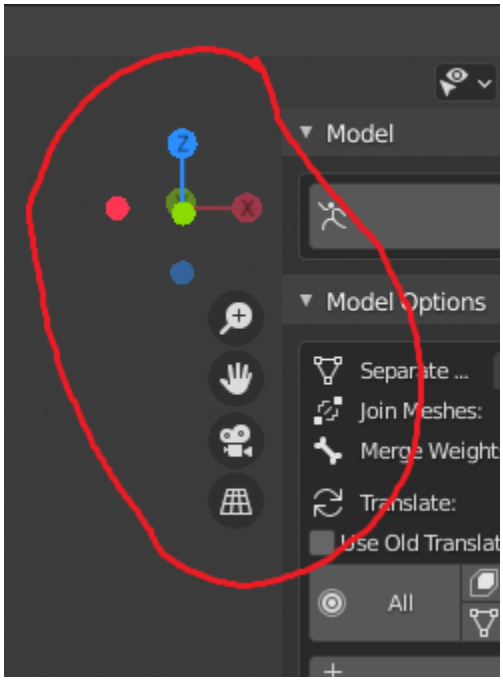
The following camera controls will be useful regardless of whether you're making an avatar from scratch or just exporting a pre-made model to VRChat.

Zoom: Use your mouse wheel to zoom in and out of the scene.

Rotate: Hold **MMB** to rotate around the scene. If you're emulating the middle mouse button, instead use **Alt - LMB**.

Pan: Hold **Shift - MMB** to pan around the scene. If you're emulating the middle mouse button, instead use **Shift - Alt - LMB**.

There are also buttons in the upper-right corner of the 3D viewport for controlling the camera. The axes buttons will be the most useful buttons here since they can be used to reorient the camera along specific axes.



Clicking the blue Z knob will orient the camera so it is looking at your scene in the direction of the negative Z axis (top).

Clicking the red X knob will orient the camera so it is looking at your scene in the direction of the negative X axis (right).

Clicking the green Y knob will orient the camera so it is looking at your scene in the direction of the negative Y axis (front).

There are corresponding knobs for looking from the bottom, left, and back as well.

You can also press ` (back tick) to bring up a pie menu to switch between these camera orientations.

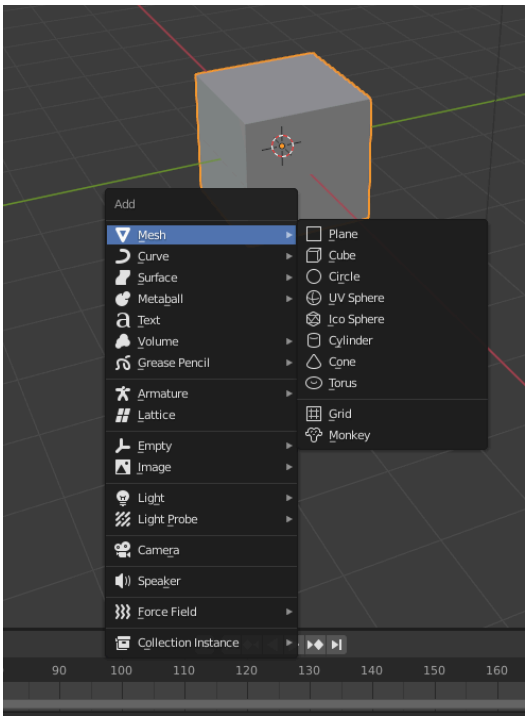
Creating an avatar from scratch in Blender

The high-level workflow we'll follow for creating our VRChat avatar in blender is as follows:

1. Modelling (geometry)
2. Materials (colors and textures)
3. Bones (armature and skinning)
4. CATS plugin
    a. Visemes
    b. Eye Tracking
    c. Optimization
    d. Texture Atlasing
5. Export for Unity

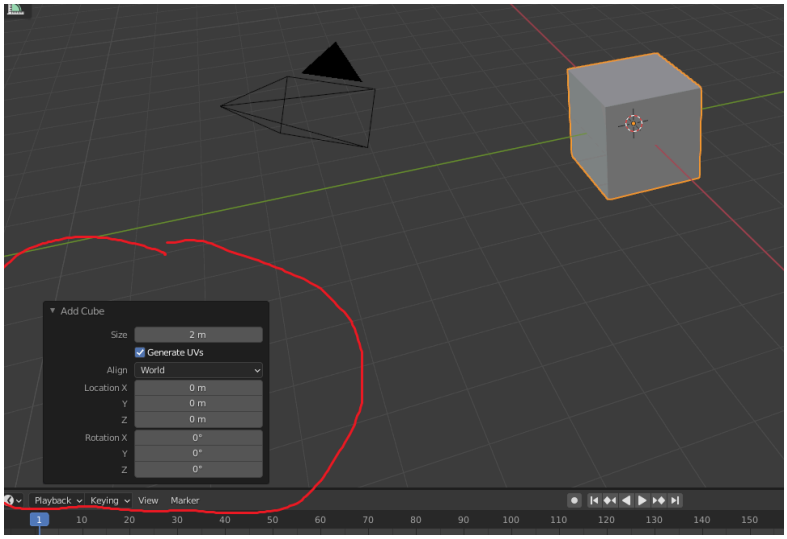
Modelling

Creating Objects

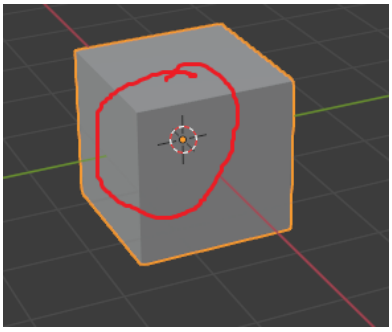


To create a new object, press **Shift - A**. From the menu that appears, you can create basic types of geometries and other 3D objects for creating and modifying your avatar. You'll most often use the shapes listed under **Mesh** for modelling your avatar. Later, we'll also rig up our object for VRChat animations by also creating bone objects for it.

When you create an object, a menu will appear in the lower-left corner of the 3D viewport to set its initial properties.



The 3D Cursor



Note that your new object will always be created at the 3D cursor - the little thing in the 3D viewport that looks like a really thin lifesaver ring.

You can move the 3D cursor with **Shift - RMB**. While moving it, you can lock it along one of the 3 axes by pressing **X**, **Y**, or **Z**.

You can recenter it at the origin of the scene with **Shift - C**.

Remember this 3D cursor. It will be very useful for many modelling operations.

Selecting stuff

The simplest way to select things in the 3D viewport is just to left-click them or drag **LMB** to select everything in a rectangle.

You can select everything in the scene by pressing **A**.

You can unselect everything by pressing **Alt - A**.

You can invert your current selection by pressing **Ctrl - I**.

In Edit mode, you can press **Ctrl - L** to select the entire mesh connected to the currently selected element.

Use **Shift - LMB** to select additional objects after the initial selection.

Use **Ctrl- RMB** to draw a freehand lasso to select objects.

Moving, Rotating, and Scaling stuff

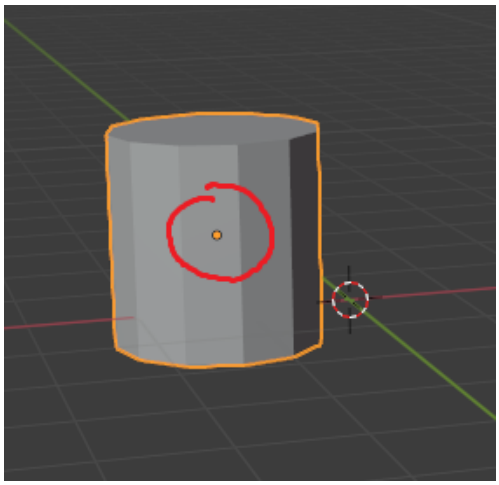
To move an object, select it and press **G**. Left click to stop moving it.

To rotate an object around the axis the camera is currently facing, select the object and press **R**. Left click to stop rotating it.

To scale (resize) an object, select it and press **S**. Left click to stop scaling it.

To can lock any of these operations along the X, Y, or Z axis by pressing (but not holding) **X**, **Y**, or **Z** respectively.

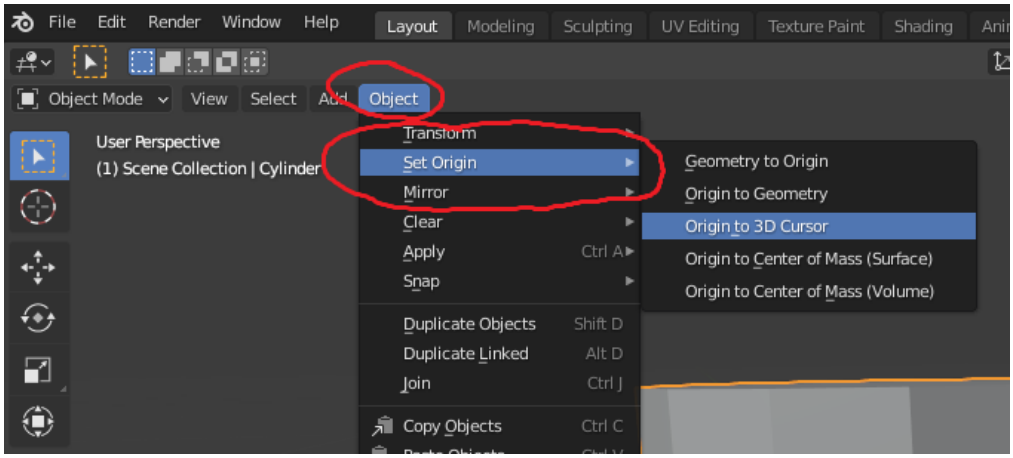
Object origins



When you rotate or scale an object in Object mode, it is transformed relative to its **origin**, represented by an orange dot when the object is selected. In other tutorials, this is also referred to as the object's "pivot point".

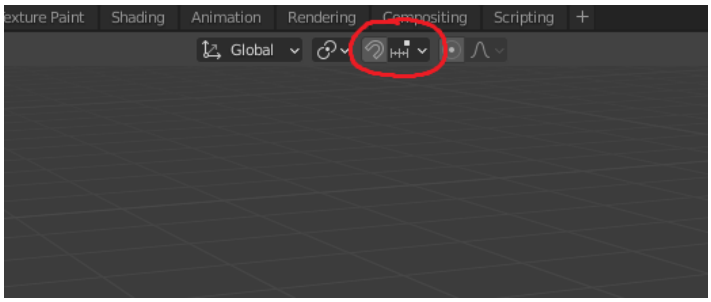
You can adjust the origin of the selected object from the 3D viewport's Header bar by going to **Object** > **Set Origin**.

It is often useful to move the 3D cursor first, set the object's origin to the 3D cursor, and then move/rotate/scale it around that.



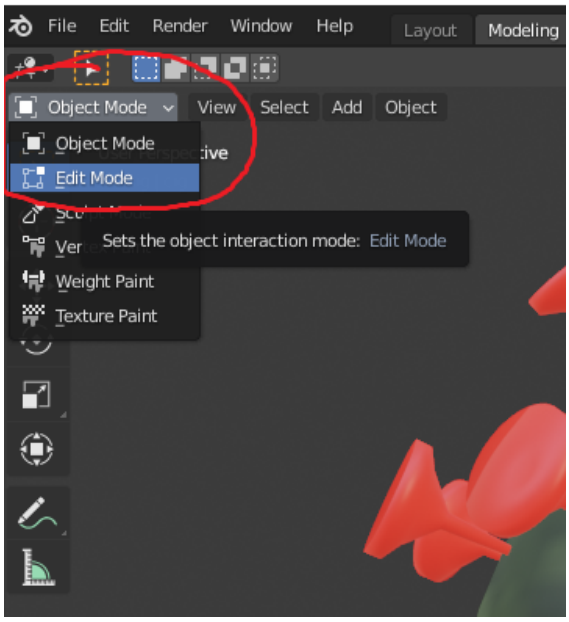
Snapping

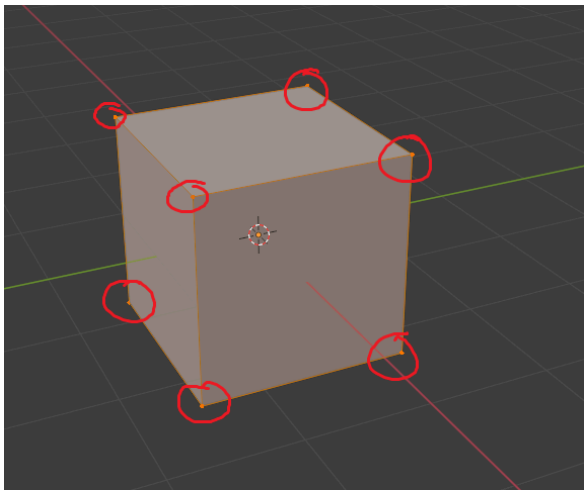
While moving, rotating, and scaling stuff, you can snap objects to the grid or relative to other objects using options in the Header bar of the 3D Viewport. Snapping can be enabled by clicking the magnet-looking button or by pressing **Shift - Tab**. While moving, rotating, or scaling an object, you can also have it snap to the grid while you're doing the transform by holding **ctrl**.



Editing Geometry

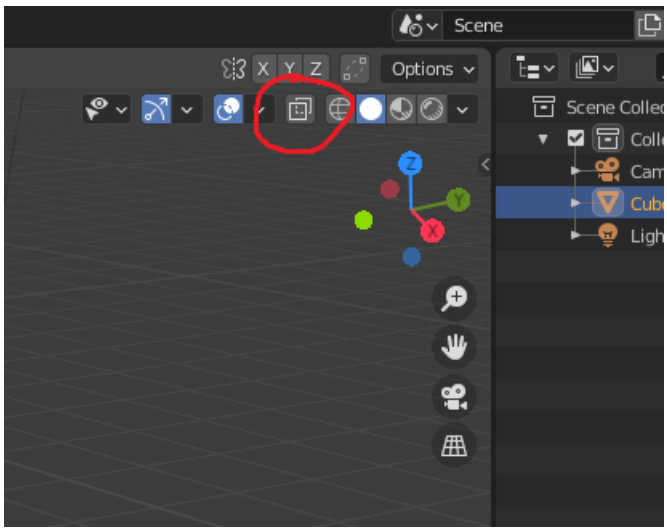
In **Object** mode, you'll only be able to interact with the high-level objects in your scene. If you want to edit and sculpt their meshes, you'll need to switch to **Edit** mode. You can do this by pressing **Tab** (pressing Tab again will switch back to Object mode) or by selecting "Edit Mode" from the Mode drop-down menu.

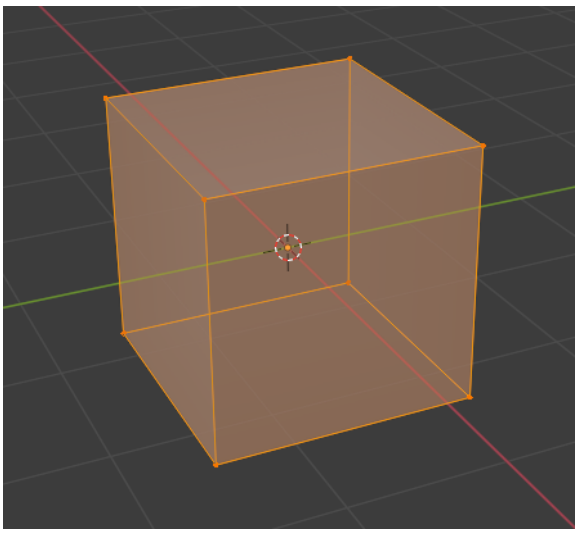




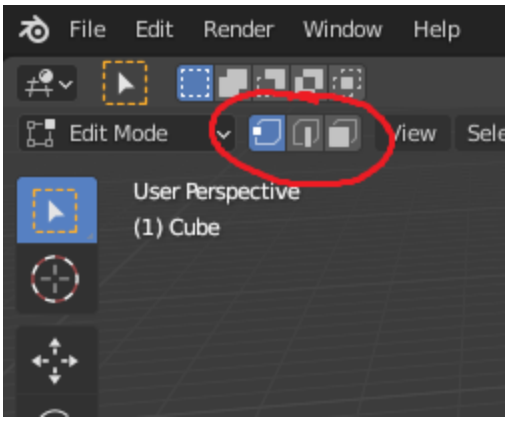
When you enter Edit mode, you'll be able to see and manipulate the individual vertices, edges, and faces of the currently selected objects.

By default, you'll only be able to select and manipulate front-facing vertices and faces. To also be able to select and mess around with the back-facing stuff, you'll need to switch on **X-Ray** mode with **Alt - Z**, or by pressing its button next to the rendering modes.





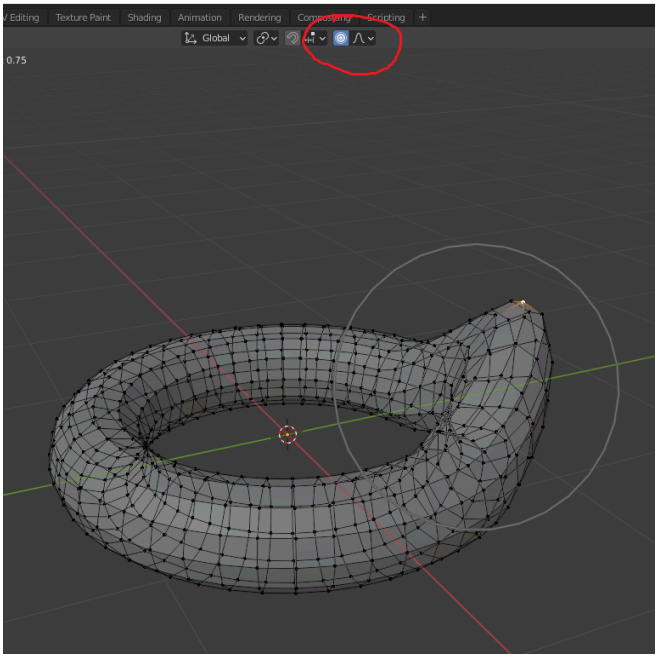
Vertices, Edges, and Faces



Vertices (singular “vertex”) are the most basic elements that define individual points in a geometry. Edges are the lines between two connected vertices. Faces are the triangles or quads formed by 3 or 4 connected vertices. You can switch between selecting vertices, edges, or faces from Header Menu in the 3D viewport.

You can use the usual Move (G), Rotate (R), and Scale (S) functions to manipulate your meshes here.

Proportional Editing



If you want to manipulate several elements of a mesh at once, it might be useful to turn on Proportional Editing. With this turned on, you’ll see a circle around the elements you are editing. Elements within that circle will also be modified, proportional to how close they are to the focal elements.

You can use the mouse wheel to change the size of the circle for proportional editing.

See in the screenshot how one vertex is moved to also move the vertices near it!

Duplicating Objects

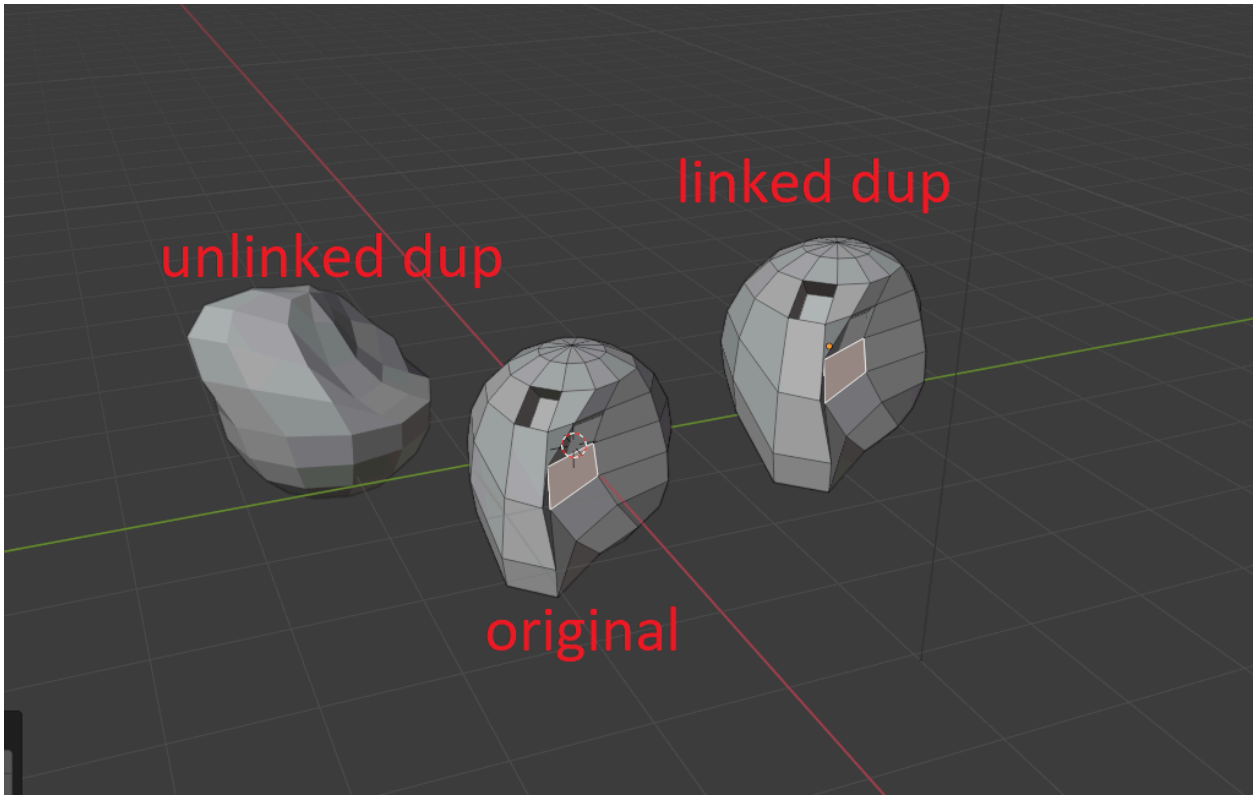
There are two ways to duplicate an object, and it is important to know the difference!

Unlinked duplicates

The first way is to create an **unlinked duplicate** of the object by selecting the object and pressing **Shift - D**. This clone is entirely independent of the original object. You can change the meshes, materials, and everything else about these objects independently of each other.

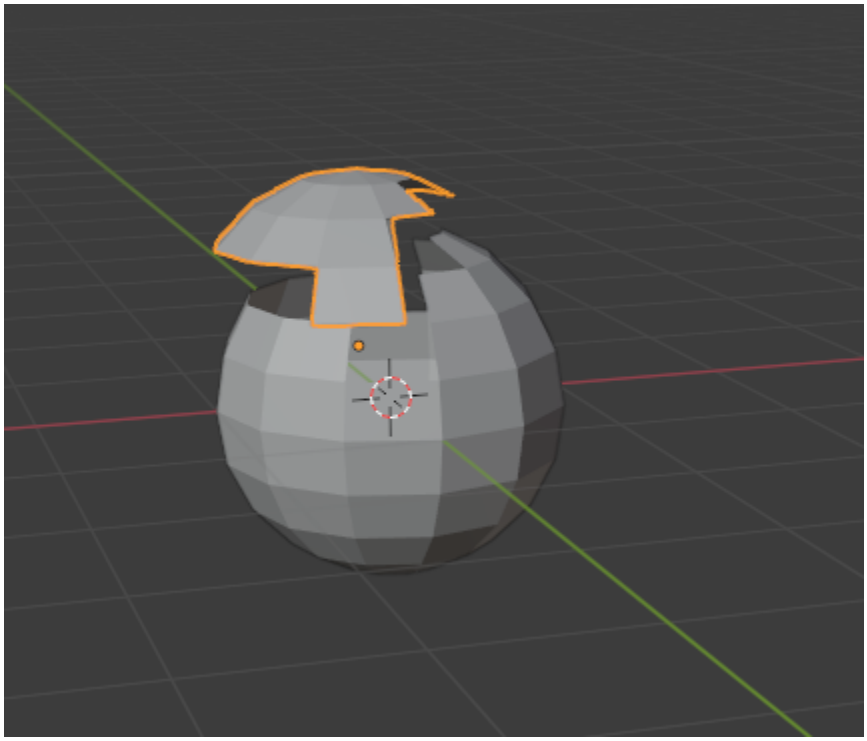
Linked duplicates

The second way is to create a linked duplicate. This is done by selecting the object and pressing **Alt - D**. Any changes to the mesh of the linked duplicate, or to its materials, or most other properties, will affect the original object as well, and vice versa. You can have multiple linked duplicates of a single object!



Separating parts of a mesh

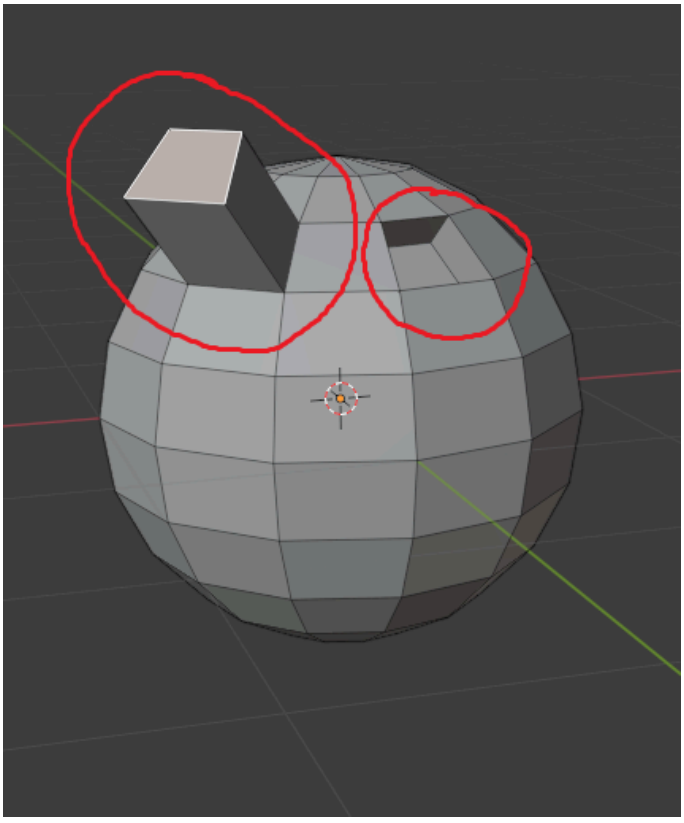
You can separate the selected elements of a mesh into a new object by pressing P in Edit mode. These elements will become disjoined from the original object and will become a new object.



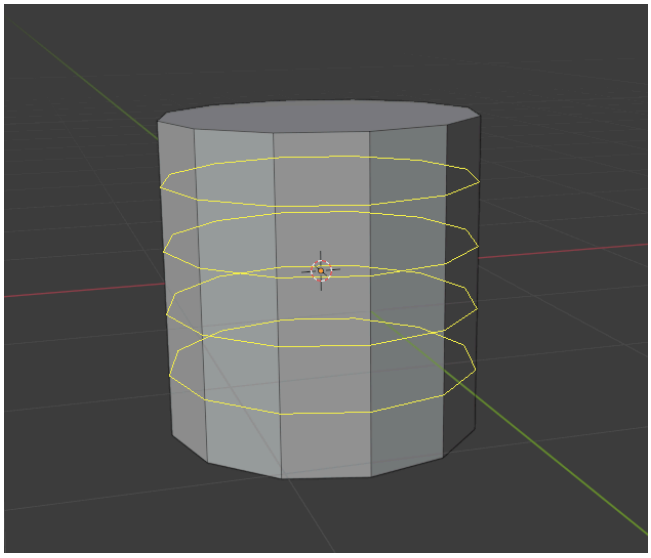
Extruding part of a mesh

You can extrude the selected elements of a mesh in Edit mode by pressing E. This will duplicate the selected elements and automatically connect them to the rest of the mesh. This allows you to do cool things like raise or lower a face straight out or into the object! This works best with faces or with a selection of 3 or more vertices.

It is strongly advised to move, rotate, or scale the extruded stuff immediately after extruding it. Otherwise you risk it getting lost and overlapped into the original parts of the mesh.



Loop cuts

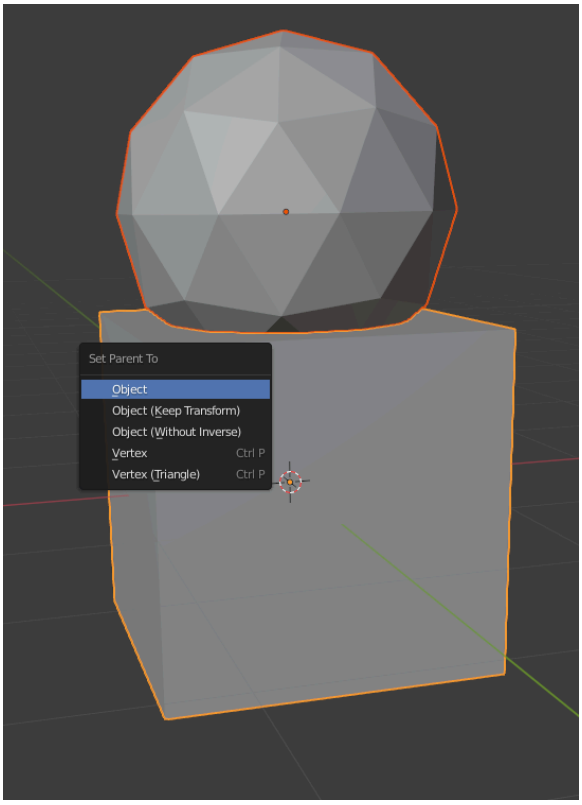


Another way to add more geometry to a mesh in Edit mode by loop cutting it. This only works with meshes that have manifold geometry (their faces use quads instead of triangles). So you can use this on objects such as spheres, cubes, and cylinders, but not cones.

To start creating a loop cut for the object currently being edited in Edit mode, press Ctrl - R. You can adjust the number of loop cuts being made using the mouse wheel. You can move the mouse to also change whether the loop cuts are being made horizontally or vertically. Press LMB when you have the desired number of loops. Adjust the loop cuts by moving the mouse once more and then press LMB again to finalize the loop cuts.

Parenting an object

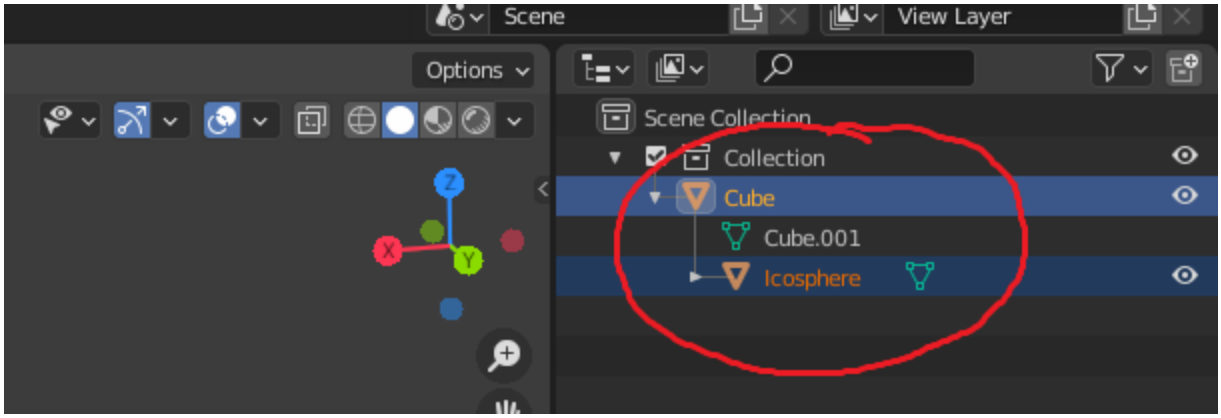
You can set one object as the parent of another object by selecting two objects, with the parent object selected last, and pressing **Ctrl - P**.



Whenever you move, rotate, or scale the parent object, the child object will be moved, rotated, or scaled along with it! This is very useful for moving entire groups of objects around easily. It will also be important for bones used later for rigging your model.

One parent object can have multiple child objects. When selecting the objects, the children elements are highlighted in orange and the parent in gold.

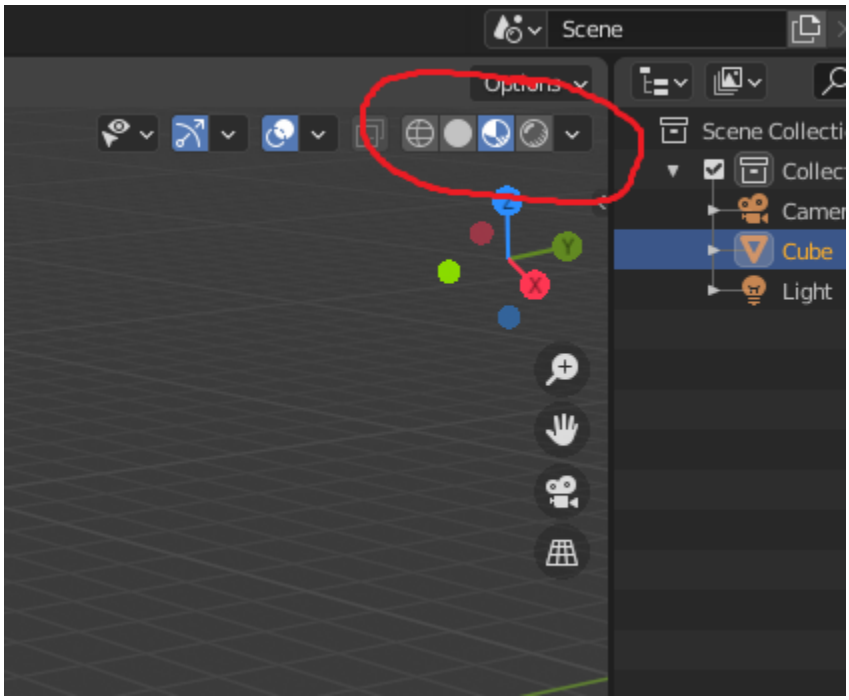
In Outliner panel in the upper-right side of the UI, the child objects will appear in the hierarchy inside their parent object.



Materials (colors and textures)

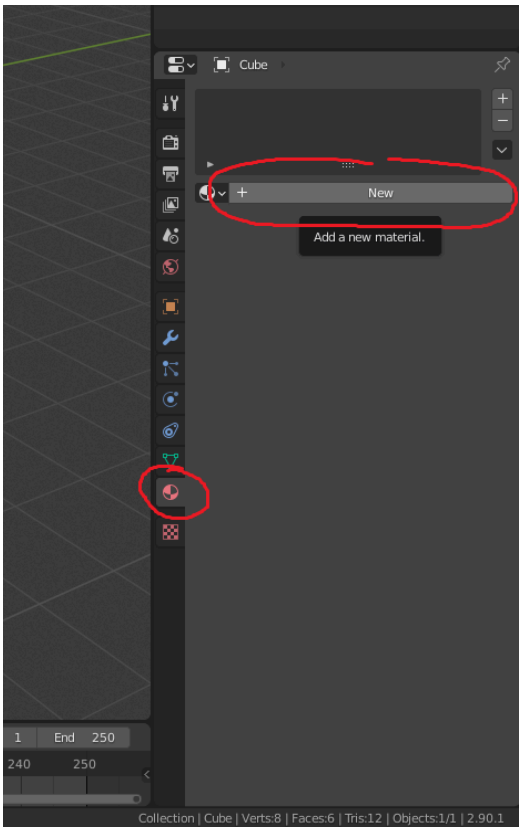
With our model done, we are ready to apply **materials** to its objects in the form of either solid colors or image textures. In this part, we will go over both kinds of materials.

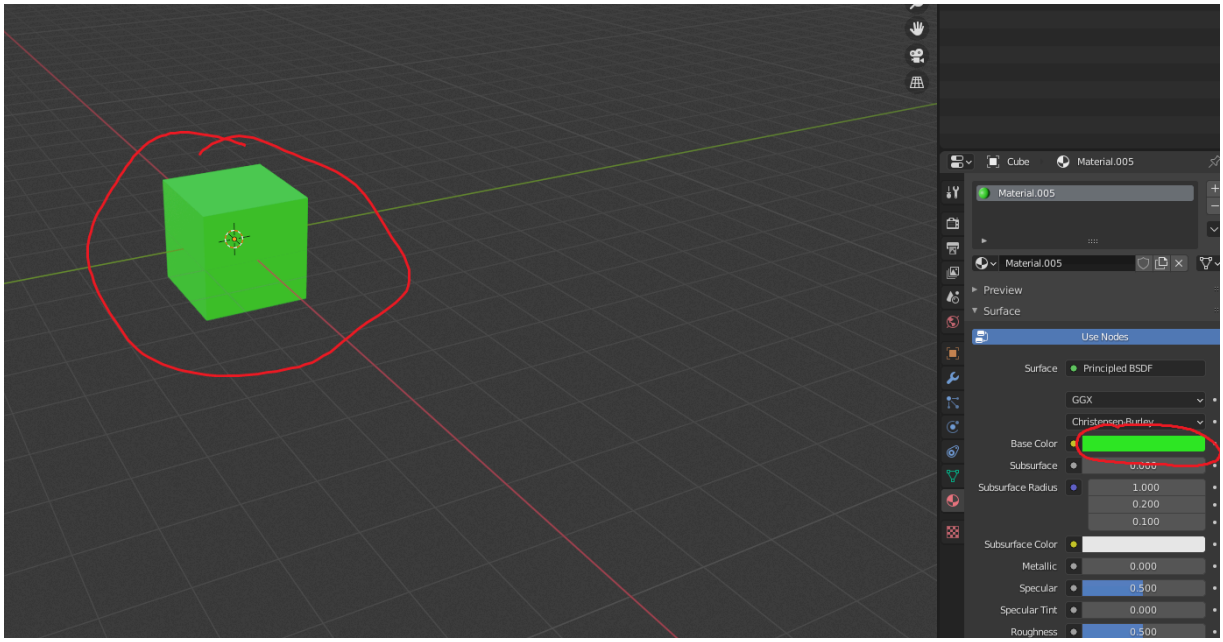
Before we start adding textures and colors to our models, it will be useful to switch the render mode to **Material Preview** mode so that we can actually see the materials applied to our model. (The other modes are **Wireframe**, **Solid** (the default mode), and **Render Preview**).



Solid colors

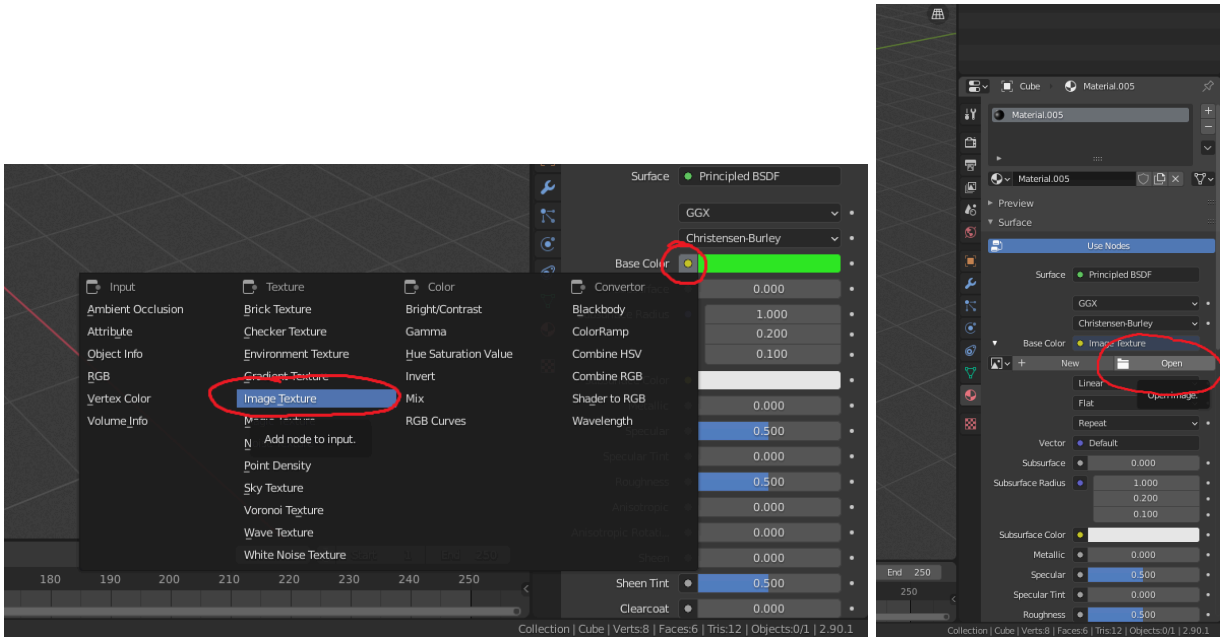
The most basic kind of material you can apply to an object is a solid color. To apply a new color to an object, select the object while in **Object** mode and click the **Material Properties** tab in the **Properties** workspace. Then click **New**. A new material will be created for the object that is initially just a solid white color. You can change the color by setting the **Base Color**.

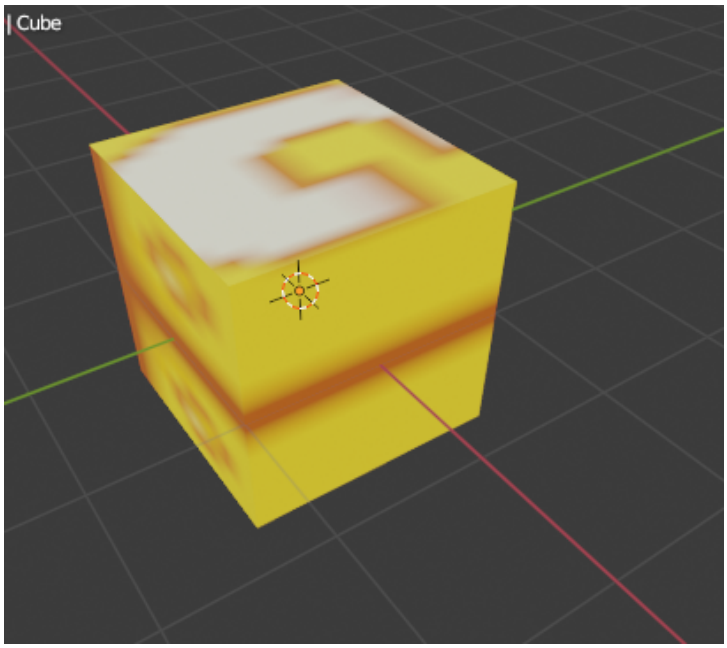




Textures

You can apply an image to the surface of your object by adding an **Image Texture** material to it.To do this, add a new material to your object (just like you did for solid colors), and click the little dot next to **Base Color**. A menu will appear. Select **Image Texture** and then click **Open** just under **Base Color** to select the image to use from a file.

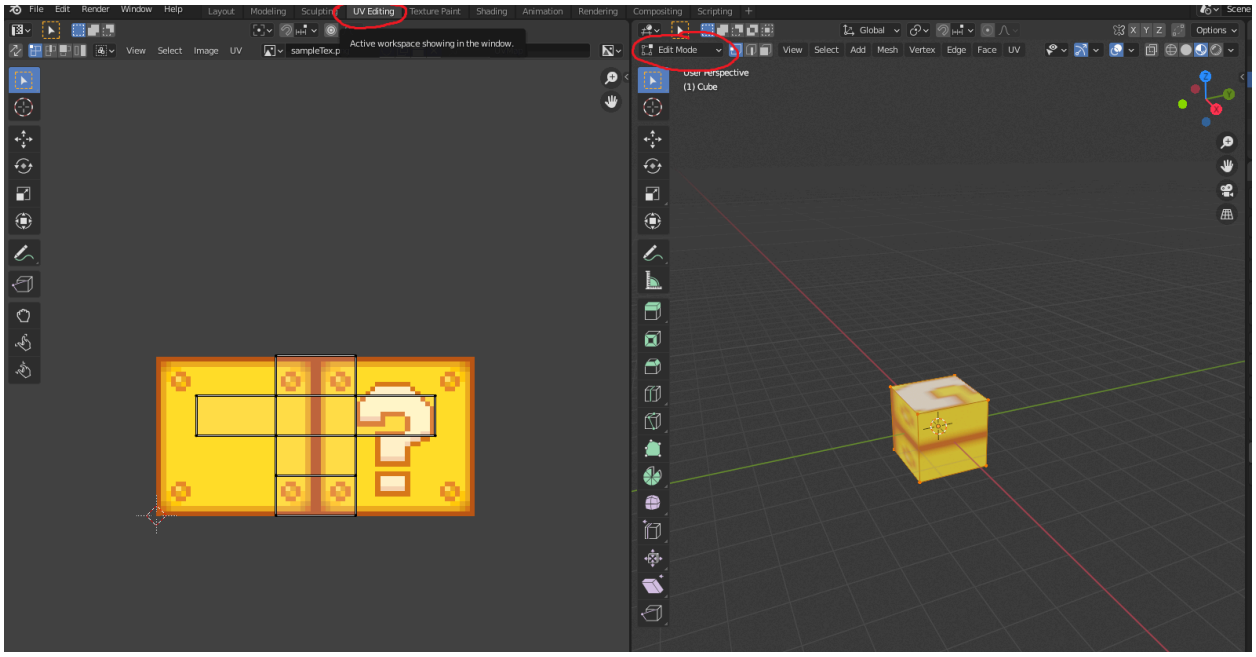




You'll probably notice that the parts of the image are probably not in the place you want them to be. We'll fix that next with UV editing!

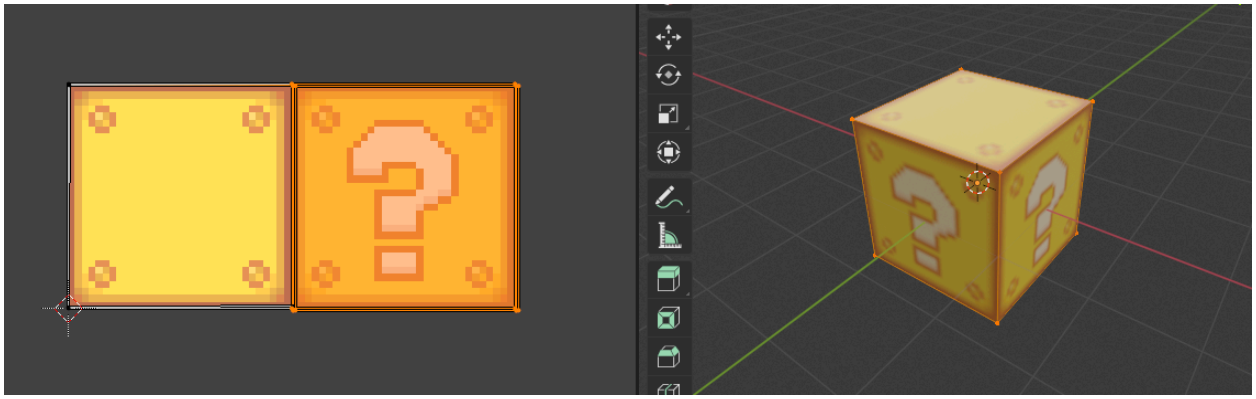
UV Editing

Each vertex in your model has not just positional coordinates, but also UV coordinates used to tell what part of an image maps up to that vertex on the model's surface. To change these coordinates, switch to the **UV Editing** workspace, select your textured object, switch to **Edit** mode, and select the vertices you want to remap on the texture.



The lefthand screen is the UV map. You can select the vertices overlaid on the texture and move, rotate, and scale them until you've got them looking good on your object in the 3D Viewport on the righthand screen. While mapping the vertices, it is often useful to enable snapping (or hold **ctrl** for snapping).

It might also be more useful to only map part of your model at a time instead of all at once. In this example, I mapped each face of the cube one at a time. Eventually, you end up with your fully mapped texture like this:



Finalizing your model

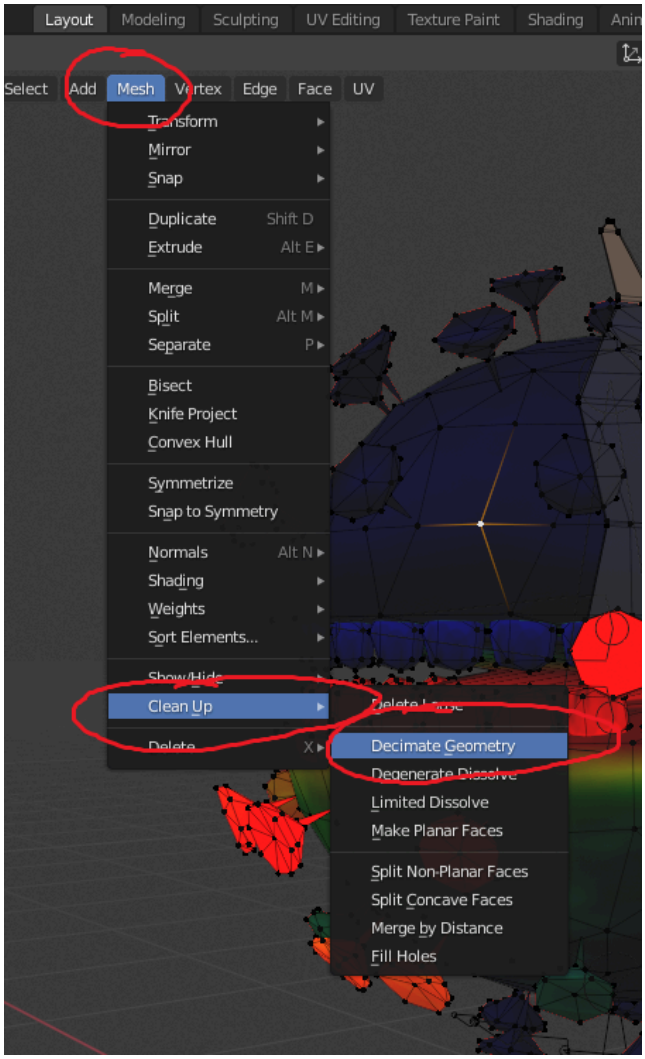
By now, you should have your avatar modeled and textured. It should also be standing in a T-pose. This will make it easier for you to create the skeleton for your model in the next section.

Before we move onto rigging your model with a skeleton, we have one more step that will make the rigging process easier and will optimize your model for VRChat. You'll need to join all the objects of your avatar into one mesh. Unity has to make a "Draw Call" for every mesh in your scene, so the fewer meshes you have, the faster your model will render.

Do this by selecting all the objects for it and press **Ctrl - J**. Now we are ready to make a spooky, scary, skeleton for your model!

Also, if you plan to make your avatar usable for Quest (which you should), also make sure that your model has 10,000 or fewer triangles. For PC, this limit is 70,000 triangles. If you need to cut down the number of tris in your model, you can do so easily by decimating its mesh. This will reduce the number of vertices in your mesh while trying to keep its general shape preserved.

To do this, go to **Edit Mode**, then select the vertices for the part of your mesh you want to decimate. You can easily select all the vertices connected to part of your mesh by selecting one or more vertices in and pressing **Ctrl - L**. From the 3D Viewport' header, select **Mesh** > **Clean Up** > **Decimate Geometry**. Set the ratio to something lower than 1.0 until you've got a good balance between the shape you want and the number of triangles you need to get rid of.



Bones (rigging)

In the process of rigging, we'll create a skeleton for our model called an **armature** and skin it to objects and vertex groups of our model. In pose mode, we can then move the bones around and the parts of the model they are skinned to will move with them! Once the model is skinned to our armature, we will then need to create shape points for eye tracking and visemes (mouth movement).

Skeleton Requirements

For VRChat's humanoid rig, you'll need the following bones in your armature. The names are case sensitive! ">" means the right part should be parented to the left part.

Hips > Spine > Chest > Neck > Head

Hips > Left leg > Left knee > Left ankle

Hips > Right leg > Right knee > Right ankle

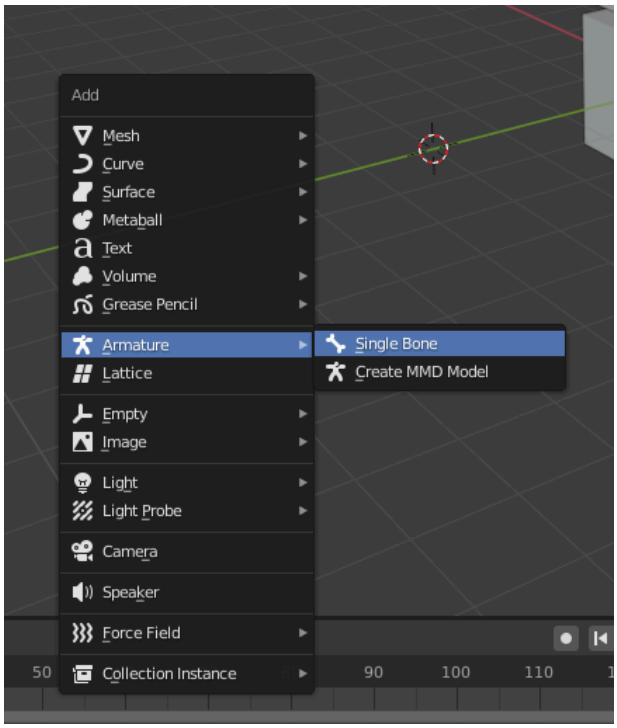
Chest > Left shoulder > Left arm > Left elbow > Left wrist

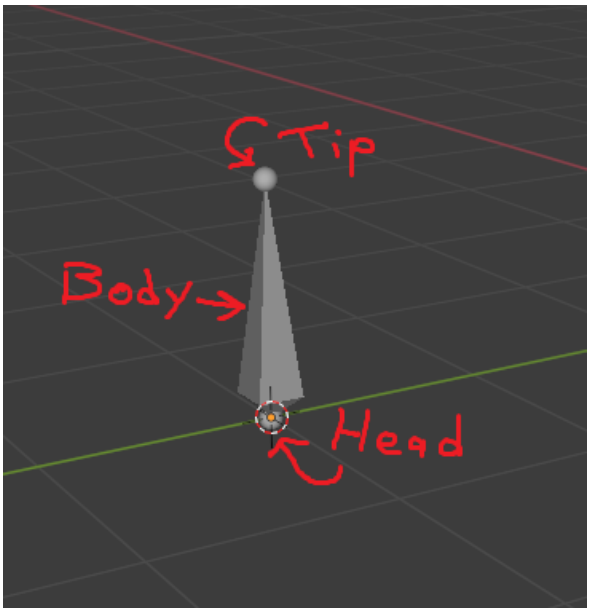
Chest > Right shoulder > Right arm > Right elbow > Right wrist

Other bones, such as for eye tracking, visemes, or fingers, are optional.

Creating the skeleton

We create our skeleton by creating bones and then parenting them to each other. You can create bones with **Shift - A**, then go to **Armature** > **Single Bone**.



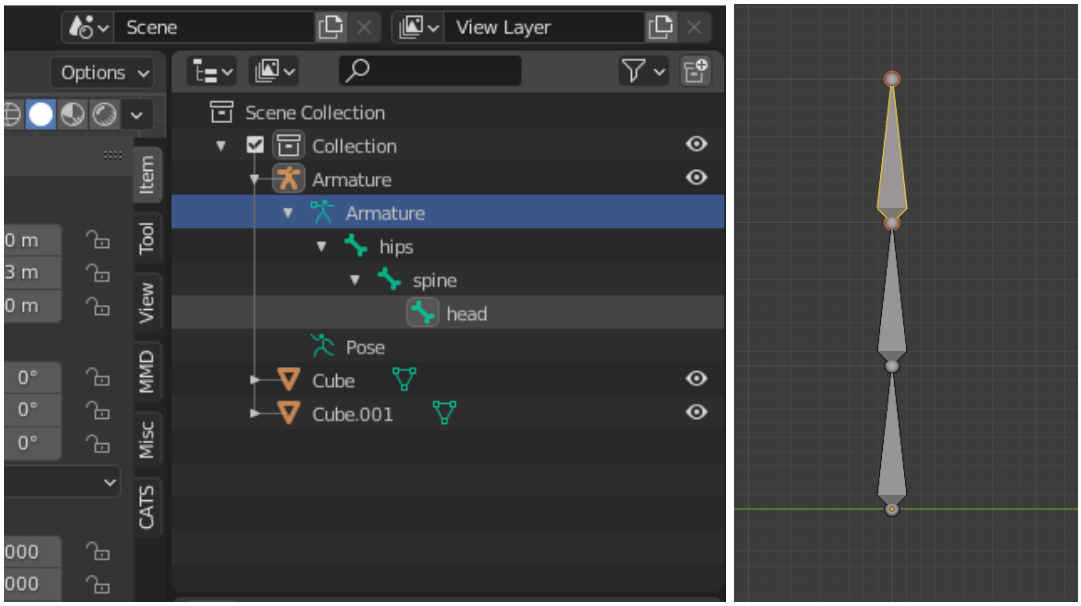


Each bone has a **head**, a **body**, and a **tip**. The head functions as the pivot/anchor point for the bone. The indicates the length and direction of the bone. The body of the bone defines the bone's volume and its roll (the rotation of the bone along the axis between its head and tip, aka its quaternion).

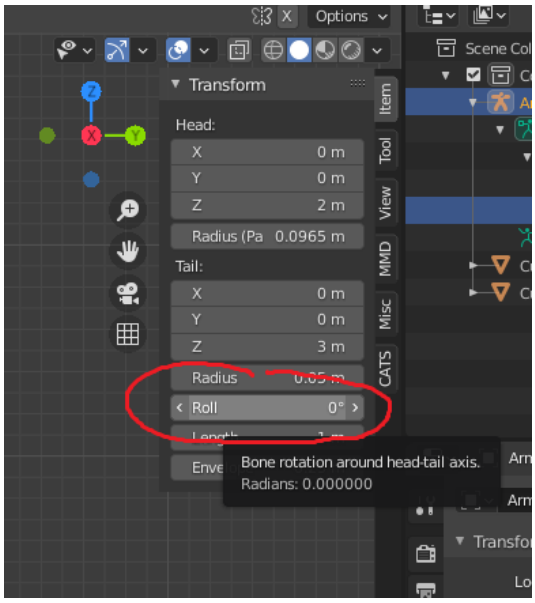
When you parent a bone to another bone, the child bone's head becomes attached to the tip of the parent bone.

In **Edit** mode, you can continue to build your existing skeleton. By selecting the tip of a bone and then pressing **E** or **Ctrl - RMB**, you can create a new bone extruded from that tip. The new bone will be parented to the tip's bone.

As you build up your skeleton, the bones will be organized into an object called an **armature**. This represents the skeleton or rigging as a whole.

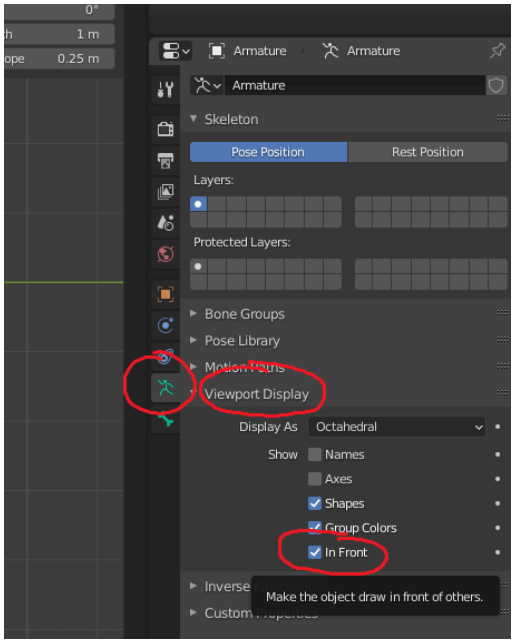


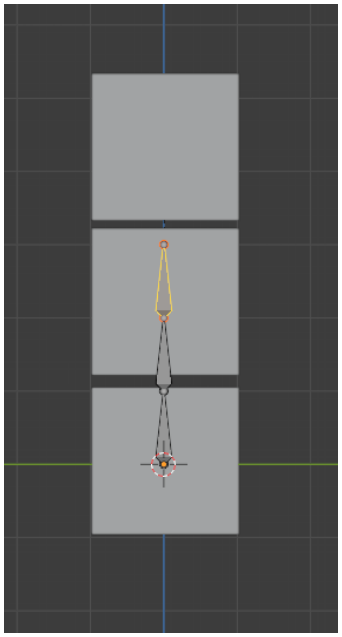
Like most objects, you can move, rotate, and scale bones to get them in the position you want them. From the transform menu of the sidebar, you can also set the **roll** of the bone.



Skeleton visibility

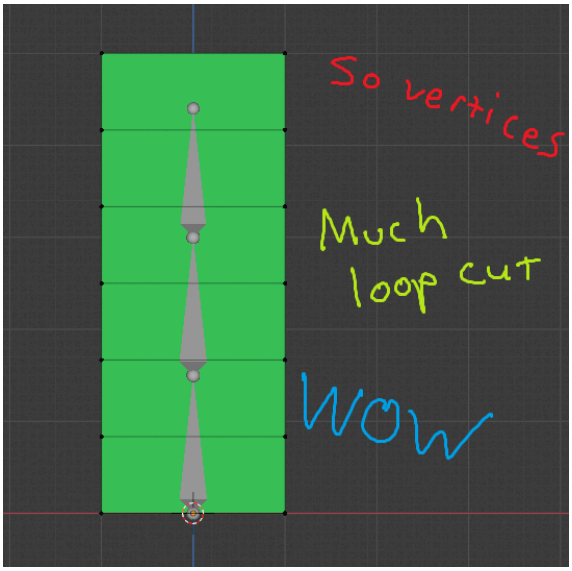
Before we start rigging, it will be useful to be able to see the skeleton for your model without having to be in wireframe mode all the time. To make it so that the skeleton will always be visible ontop of your model, select your skeleton's armature, then in the **Properties** editor, go to the **Armature** tab (the one that looks like a tiny green running guy), and expand the **Viewport Display** group. In here, check **Show** > **In Front**.





Rigging your model

The process of assigning parts of your model to the bones of your armature is called **rigging** or skinning. This is done by assigning **vertex groups** in your model to **bones** in your **armature** and **weighting** them to those bones. Each vertex group will correspond to a specific bone in your armature. Once your



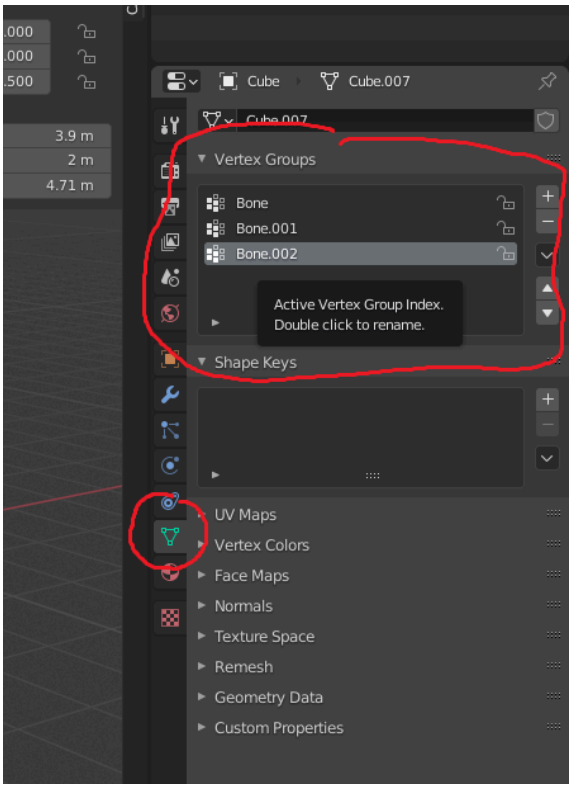
The positioning of your bones relative to your model matters a lot. When you pose your bones, the parts of the model they control will be moved, scaled, and rotated relative to the heads of these bones.

Vertex Weights

When vertices are assigned to a vertex group, they can be weighted with a number between 0 and 1, inclusive. This decides how much those vertices are affected by the bone when it moves. 0 means it won't be moved at all, and 1 means that it will be moved the exact same amount as the bone. So, a weight of 0.5 would mean that when the bone moves, the vertices will be moved only half as much as the bone.

Armature Deform

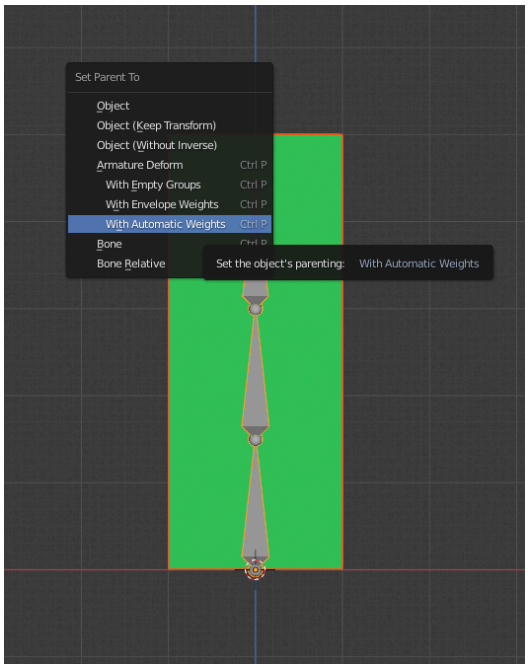
Anyways, there are a few ways to do this, depending on how much control you want over how your object's vertices are weighted to bones in the armature. All of these ways involve parenting your object to the armature using **Armature Deform**.

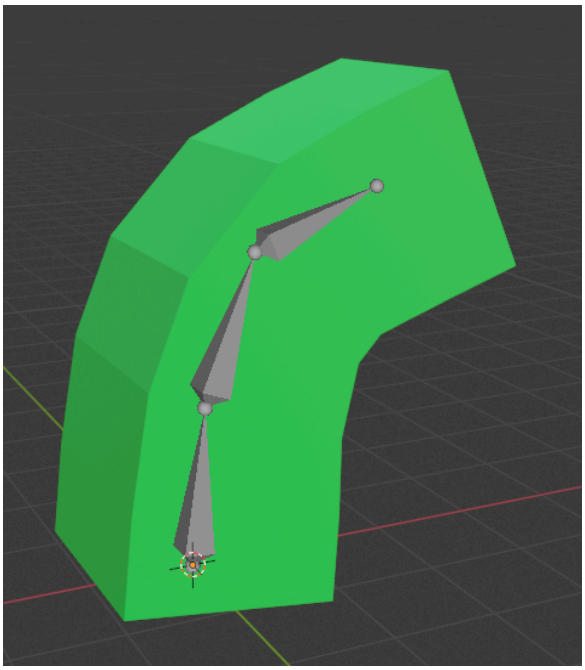


Regardless of which Armature Deform method you use, it will create vertex groups in your object named after all the bones in your armature. You can see this for yourself by selecting the object and switching to **Edit** mode, then go to the **Properties** editor and the **Object Data Properties** tab (the one that looks like a green triangle of vertices), and open the **Vertex Groups** group.

Automatic Weights

Our first and easiest option is to use Armature Deform **With Automatic Weights**. This is quick, but won't give us any fine control over how the vertices are weighted to the bones in the armature.





Envelope Weights

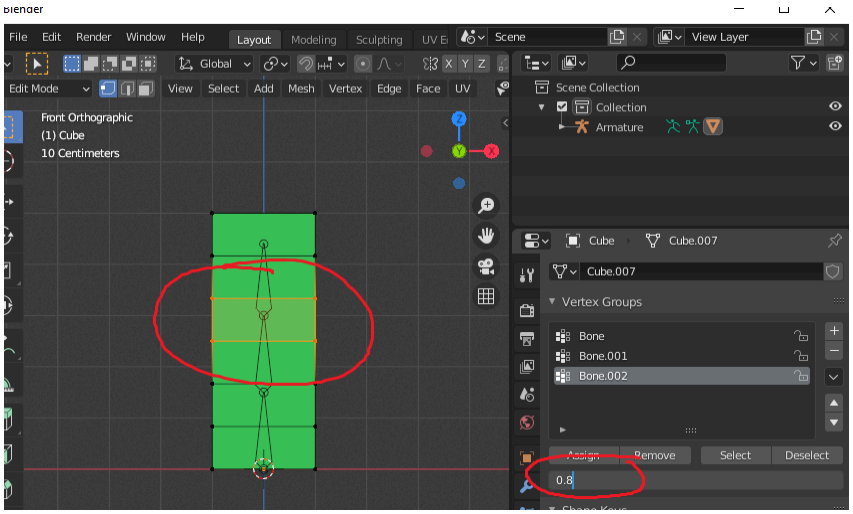
The second option is Armature Deform **With Envelope Weights**. This is like automatic weights, but the weights are proportional to the volume assigned to the bones. Thicker bones yield thicker animations. This will require you to set volume to the bones beforehand. I haven't used this, so we'll skip over it.

Empty Groups

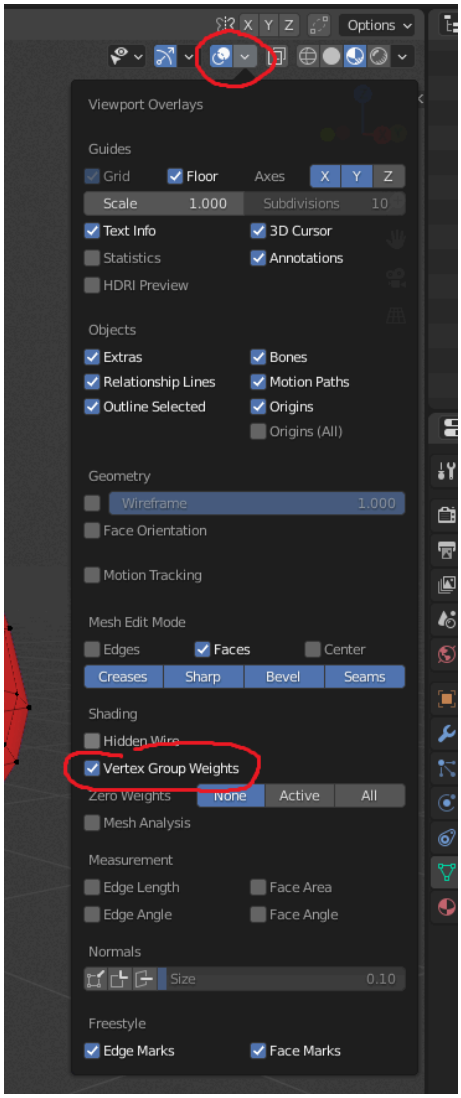
The third and most flexible option is Armature Deform **With Empty Groups**. This takes the most amount of work because it creates the vertex groups for all of your bones. Initially there are no vertices assigned to these groups. You'll have to assign the vertices and their weights to these groups yourself! However, this gives you the most amount of control over how your bones influence your vertices.

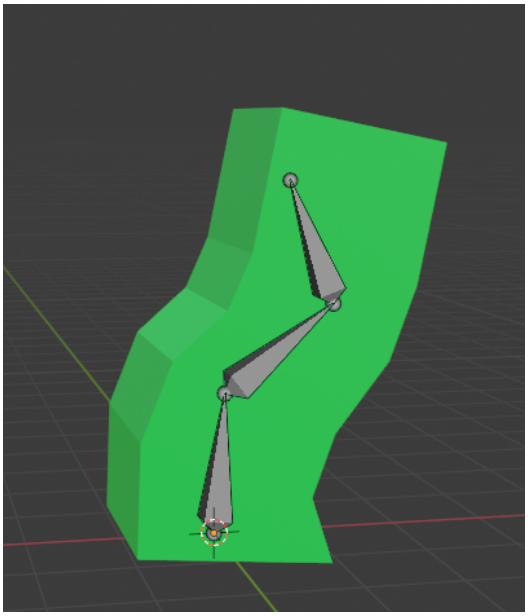
The general workflow for using this technique is as follows:

Select your object and switch to Edit mode. Go to your **Vertex Groups**, and select the vertex group corresponding to the bone you want to map vertices to. Select the vertices in your model (check first to make sure that X-Ray mode is enabled if you need it!). Then specify a weight for the vertices and click **Assign**.



With this technique, it might also be useful to make vertex weight groups visible. You can do this by going to the Viewport Overlays drop-down menu in your 3D Viewport and checking the checkbox next to Vertex Group Weights.



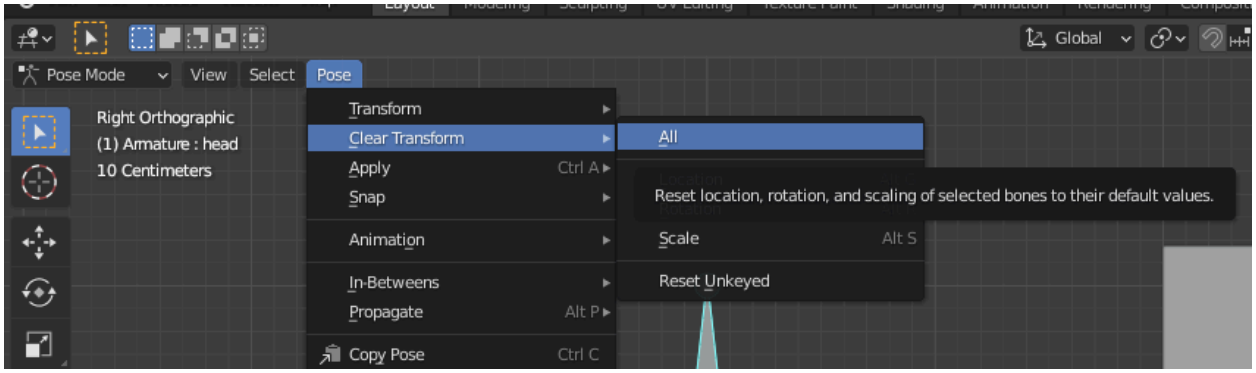


With this visible, your model will be colored according to the weights of your vertices. Vertices colored dark blue have a weight of 0, and vertices colored red have a weight of 1.

Posing your model

Once you've rigged part or all of your model, you can select the armature and switch to **Pose Mode** to test out the movement of your skeleton. All the usual bone transforms can be applied to your armature in Pose Mode - moving, rotating, and scaling. Transforms done in Pose mode will have no effect on the structure of the skeleton itself.

You can clear the pose back to the skeleton's original state by going to **Pose** > **Clear Transform** > **All** in the 3D Viewport header (be sure to select the whole armature first!).



A mistake of weights

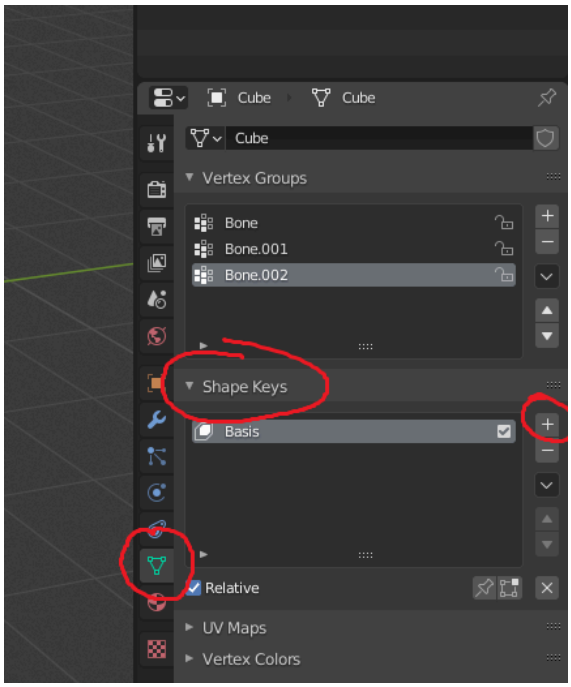
Sometimes after assigning weights to the vertices of your model, you might wind up with your bones deforming your model in very weird ways when you start posing it. This is because Blender expects the sums of all the weights on each vertex to always equal 1.

If you've, say, assigned some group of vertices a weight of 1 to multiple vertex groups, then Blender will sort of take the average of those weights among the vertex groups. That can cause things to get weird!

You can actually see that I made this mistake in the screenshot above with the jiggling cube example, as the top bone doesn't quite line up with the top of the cube.

If you run into this issue, check the weights in your vertex groups to see if you have any red areas overlapping (make sure to have visuals for vertex group weights turned on!). When you're first getting started, it might help to only have weights of either 1 or 0 on your vertices, with vertices closest to a bone being 1 and other vertex being 0 for that bone's vertex group. As you get more experienced with rigging your model, you might try using weights between 0 and 1 across multiple vertex groups.

Shape Keys



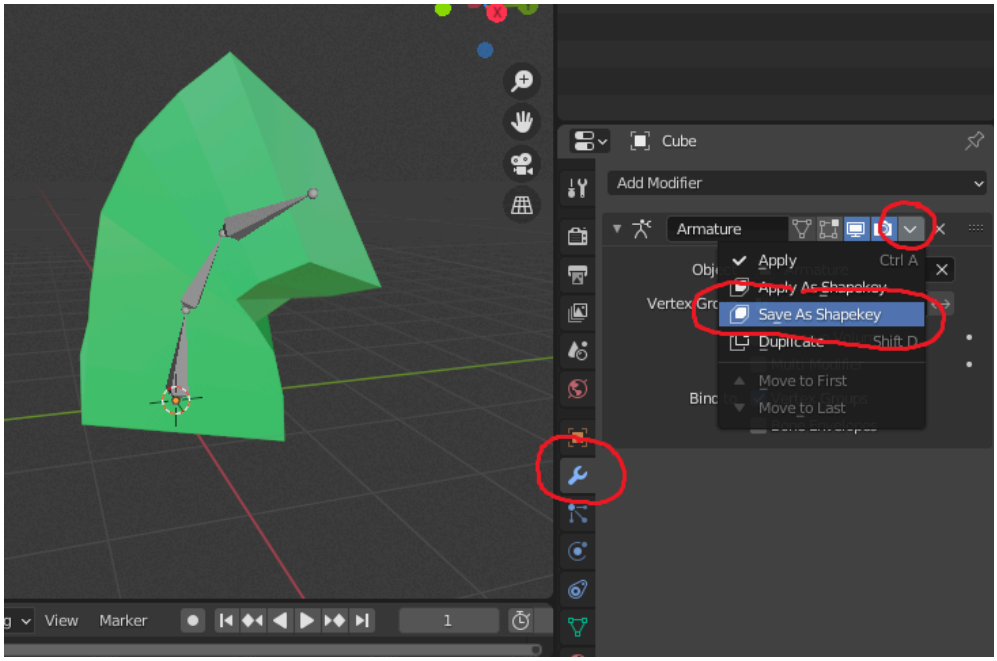
Once you've got your model rigged, you can create **shape keys**, which function as animation key frames for eye tracking and visemes. We'll talk more about those subjects once we get into using the CATS plugin later.

To create a shape key, select your mesh in **Object mode**. In the **Properties Editor**, go to the **Object Data Properties** tab (the one that looks like 3 green vertices connected in an upside down triangle) and expand the **Shape Keys** group. Click the **+** button to create a new shape key. The very first shape key you create will be the "**Basis**" shape key. This is the shape key for your model in its default pose.

To make additional shape keys, we will pose the armature for our model and then transform those poses into shape keys. Here's how!

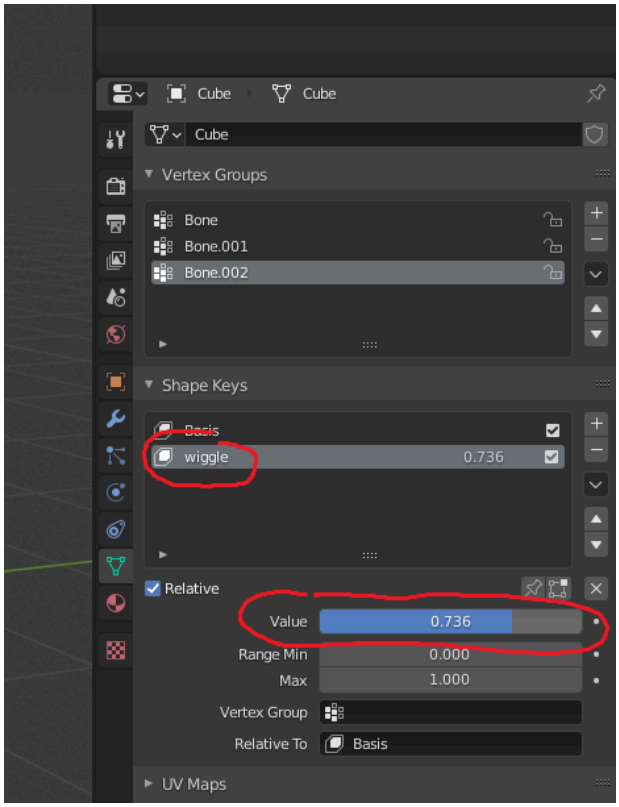
Select your armature and switch to **Pose Mode**. Pose your armature into the final position for your shape key.

Switch back to Object Mode and select your model. In the **Properties Editor**, click the **Modifiers** tab (the one that looks like a blue wrench). There should be an **Armature** modifier on your model. Click the drop-down arrow to the right of the modifier's name and select **Save as Shapekey**. This will create a new shape key from your armature's current pose!



Clear the transform of your armature to reset it to its default pose.

Now, if you go back to your model's **Shape Keys**, you should see a new shape key there! Name it to whatever you want. You can test out your shape key by selecting it and adjusting the Value slider bar. Setting this to 0 will pose your model at its initial position. Setting this to 1 will pose your model at the final position for the shape key.

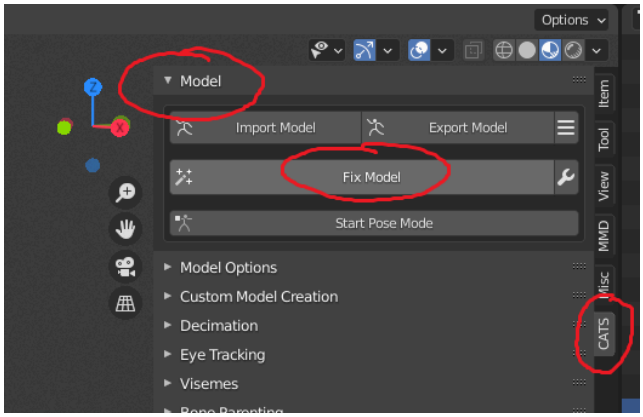


You can also create shape keys without bones. After creating your Basis shape key, you can manually transform your mesh into the pose you want, then create a new shape key from that. In some cases, this technique may be easier to achieve the pose you want rather than using bones for it.

Prepping your Avatar with CATS

Now that we've got your avatar all rigged, we'll get it ready to be imported to Unity with the help of some of the tools provided with the CATS Blender plugin.

Fix Model

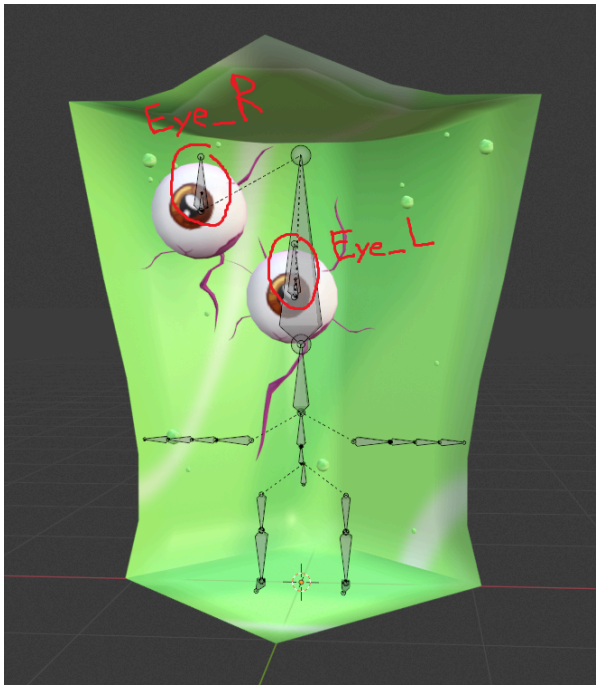


The first thing you'll want to do with CATS is use its **Fix Model** tool. This will do a variety of things to prepare your model for Unity, optimize it, and make sure it is conformant for the VRChat Avatar SDK. Do this by opening the **CATS** tab in the sidebar, opening the **Model** group, and click **Fix Model**.

Eye Tracking

Eye tracking allows the eyes of your model to follow the direction you're looking with your actual eyes with your Quest headset. With CATS, setting this up is super simple.

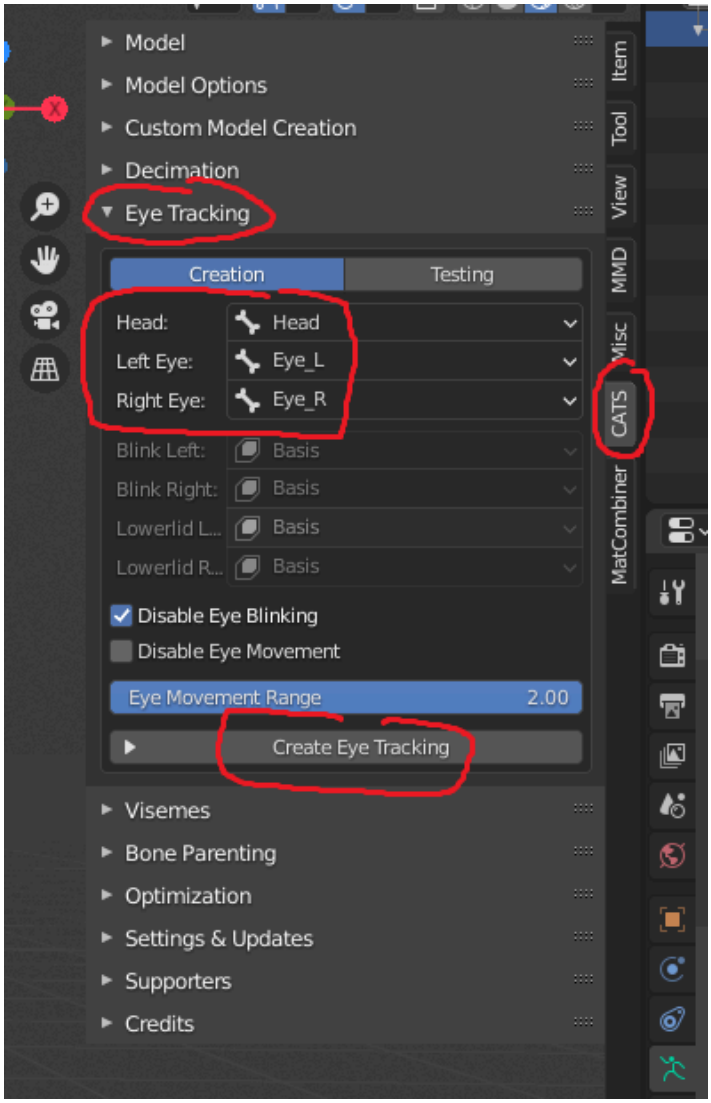
As a prerequisite, you'll need to have two extra bones for your armature parented to your **Head** bone: **Eye_L** and **Eye_R**. Put the head of each of these bones in the center of their corresponding eye and put their tip towards the top of the eye, so that the bone is pointing directly upwards.



Optionally, you can also set up shape keys for blinking if you want your avatar's eyelids to blink. For this, you should create **Blink left**, **Blink right**, **Lowerlid left**, and **Lowerlid right** shape keys which move your avatar's eyelids.

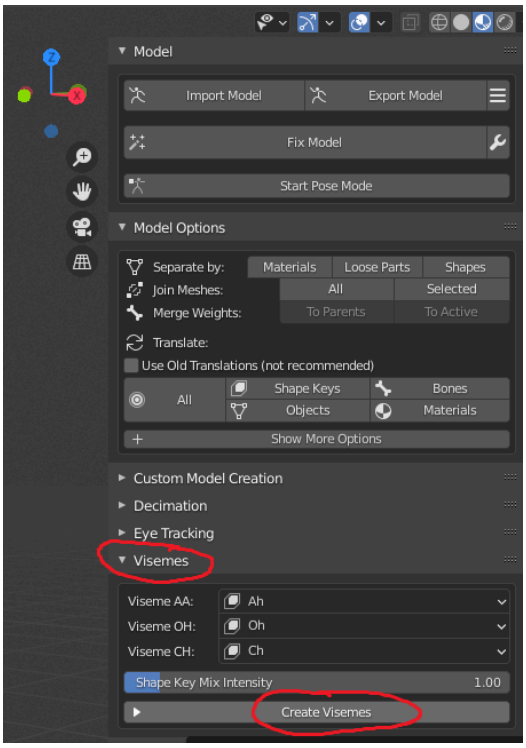
Once you've got your eye bones and (optionally) blinking shapekeys created, you're ready to use the CATS tool to create eye tracking for your model. All you have to do is open the CATS tab in the sidebar and open the Eye Tracking group. Make sure that the **Head**, **Eye_L**, and **Eye_R** bones are set to **Head**, **Left Eye**, and **Right Eye**, respectively. If you didn't set up shape keys for blinking, check the checkbox next to **Disable Eye Blinking**. Adjust the **Eye Movement Range** if you want the eyes' movement to be more or less sensitive. Then click **Create Eye Tracking**.

To test things out, you can go over to the **Testing** tab and use the sliders to move the eyes left and right, up and down. If you set up blinking, you can also test it out here as well!



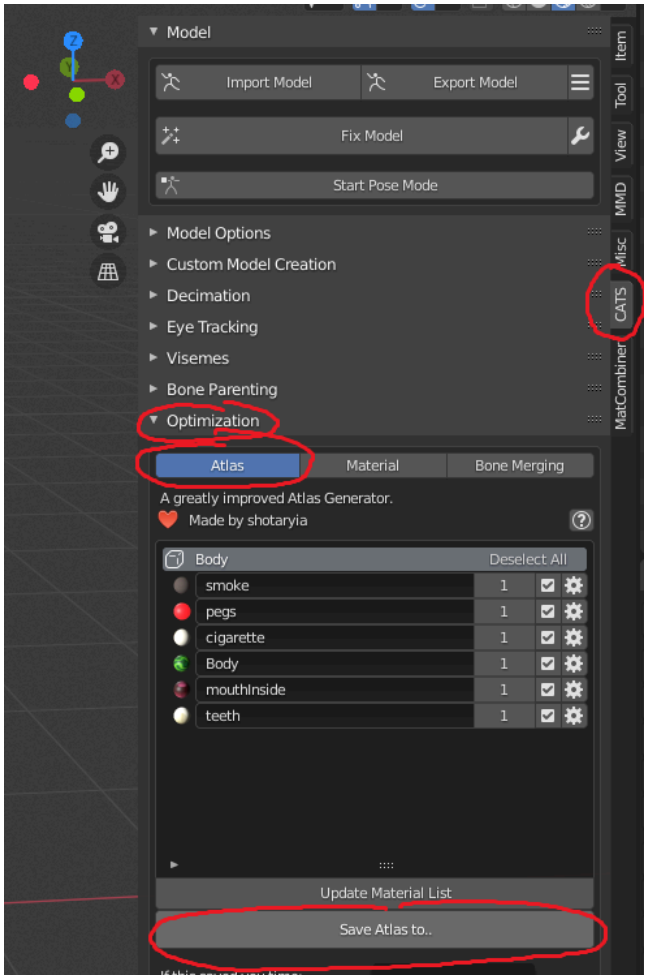
Visemes

Visemes are used to map your speech into mouth movement animations for your avatar. Normally, you would have to create several shape keys for the mouth positions of many different sounds to effectively mimic human speech. CATS makes this easy by only requiring you to make shape keys for 3 sounds: **Ah**, **Oh**, and **Ch**. It will create the shape keys for all the other visemes from them when you set the Ah, Oh, and Ch shape keys and click **Create Visemes**. When this is done, you'll notice in your model's Shape Keys, that new shape keys have been created for those other sounds.



Texture Atlasing

VRChat can support at most 4 materials for your avatar. Since you're most likely using lots of textures for different parts of your avatar, you can use CATS to transform all of your textures into just a single texture called an atlas. To do this, open the **CATS** plugin in the sidebar, open the **Optimization** group, and click the **Atlas** tab. In this tab, you should see a list of all the textures used in your model. Make sure all of the textures' checkboxes are checked, then click **Save Atlas to..**. Use the file dialog that appears to save your atlas to a new png file.



Also, if you have a lot of solid color materials, you should probably use textures of those colors instead so that they'll be included in the atlas.

Exporting for Unity

Once your model, textures, rigging, eye tracking, and visemes are all done, you are ready to export your final model so that it can be imported into Unity! You can do this by just exporting it as an .fbx file from the **File** > **Export** menu.

Basics of Unity

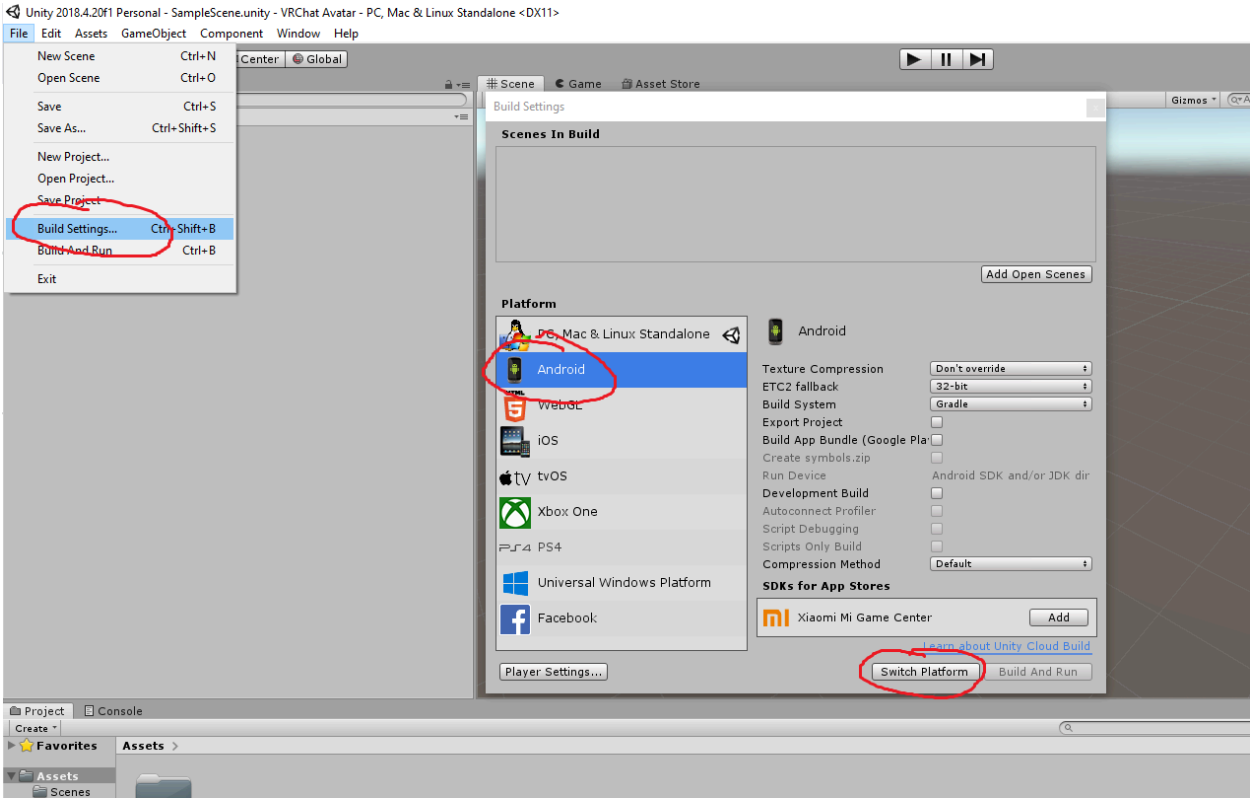
The topic of how to actually use Unity is beyond the scope of this document. You won't be doing a whole lot with it, but you should at least take a little time to become familiar with the general interface of Unity. I recommend going through the Using the Unity Interface tutorial on Unity's website: https://learn.unity.com/tutorial/using-the-unity-interface

It may also be helpful to check out the Essential Unity Concepts tutorial: https://learn.unity.com/tutorial/essential-unity-concepts.

Creating a Unity project for your avatar

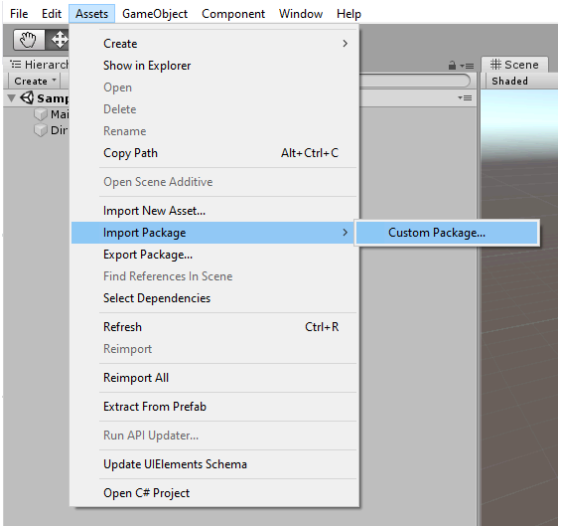
Create a new 3D project in Unity. Then, from Windows Explorer (or the equivalent of Explorer for your OS), open the VRChat SDK for Avatars that you downloaded. This will import the VRChat stuff into your unity project so that you can turn your avatar model into an actual VRChat avatar!

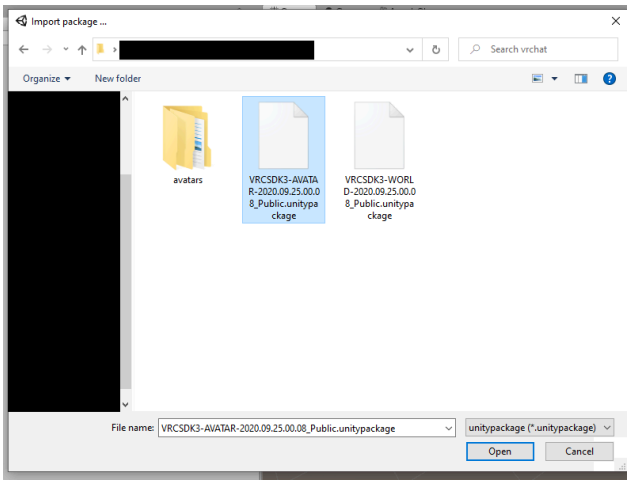
To get started, create a brand new project in Unity. Since we want our avatar to be available in Quest, we'll also want to switch our platform to Android. To do this, go to **File** > **Build Settings**. Then choose **Android** as your platform and click **Switch**.



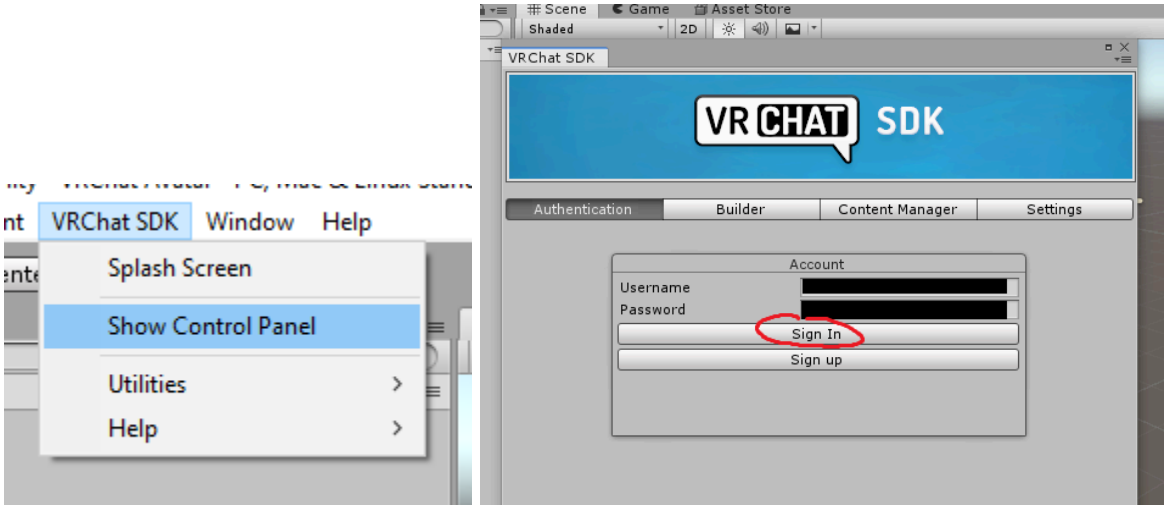
Setup the VRChat SDK

Next, you'll need to import the **VRChat Avatars 3.0 SDK**. Do this by going to **Assets** > **Import Package** > **Custom Package**. Select the VRChat Avatars 3.0 SDK package from the open file dialog. In the pop-up window that appears, make sure all the SDK components are selected and click **Import**. and wait for it to import. When it's done, you should see a new **VRChat SDK** menu in the top menu bar of Unity.

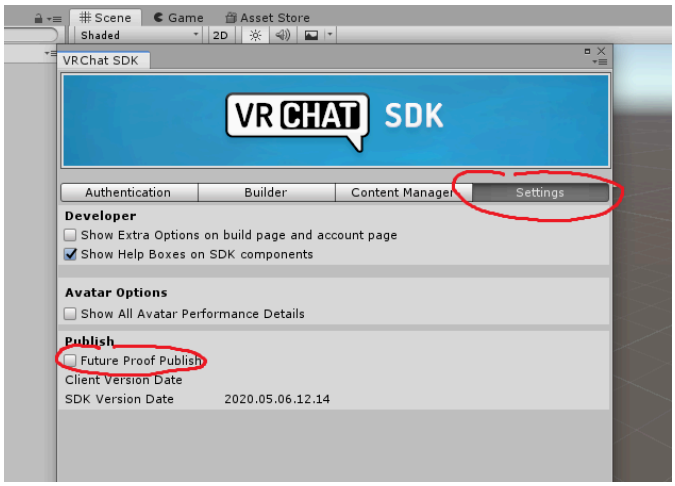




Now, let's take care of some initial setting up for the SDK. Open up the VRChat SDK by going to **VRChat SDK** (top menu) > **Show Control Panel**. A separate window will pop up and ask you to log in using your VRChat user/pass. Do that.



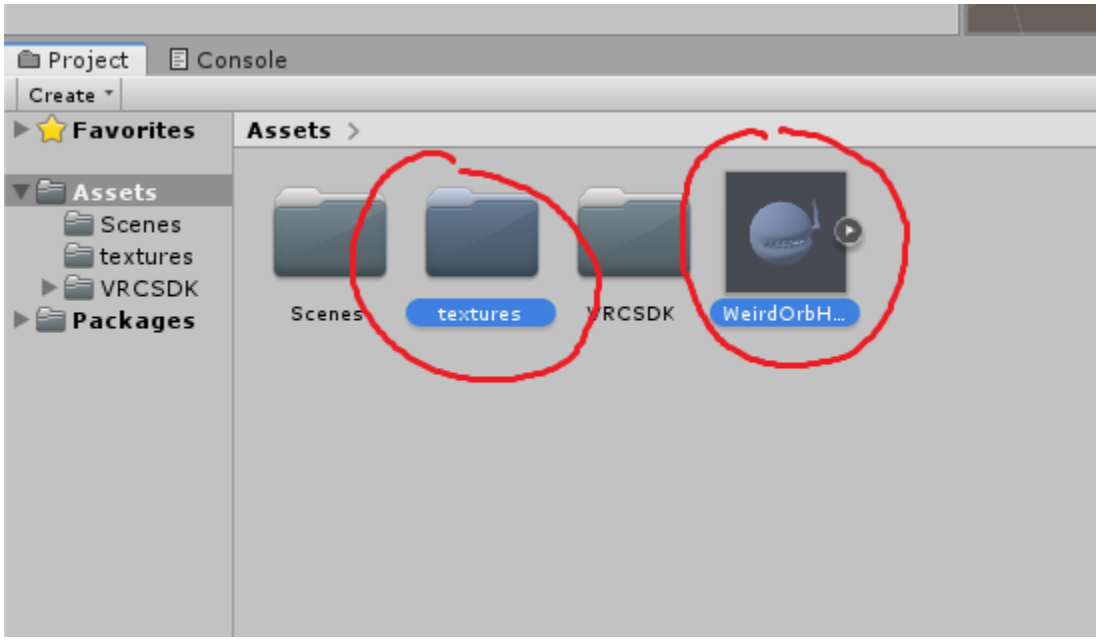
Now go to the **Settings** tab of the control panel and uncheck **Future Proof Publish**.



Now we're ready to get your model and its assets into Unity!

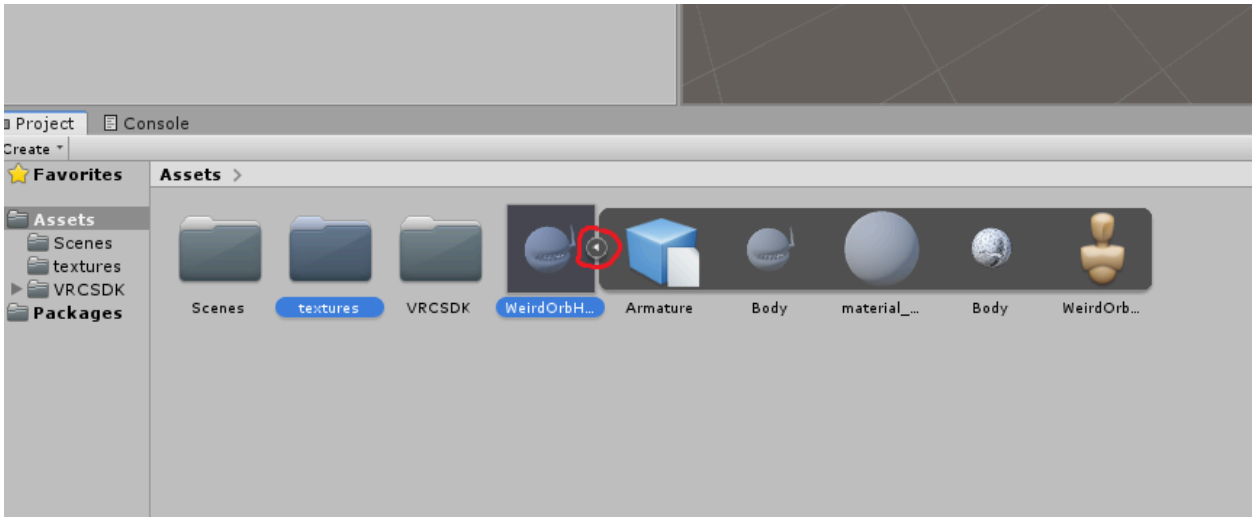
Importing your model

To import your model, just drag and drop the .fbx file for your avatar's model into your Unity project's **Assets**. Also drag and drop a directory containing your model's **textures** into your Assets. You'll really only need the atlas texture from that though.

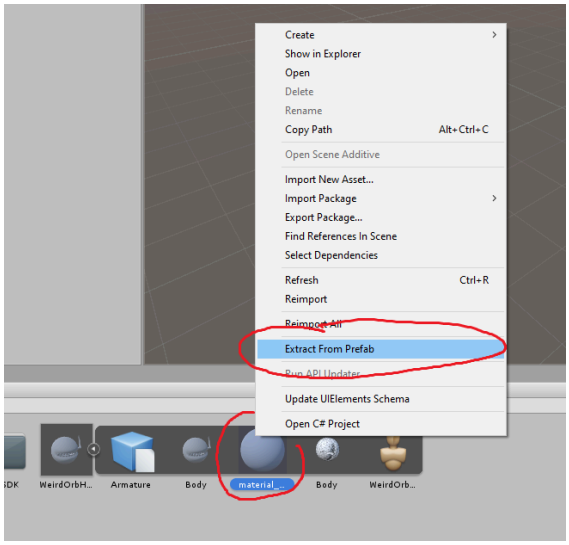


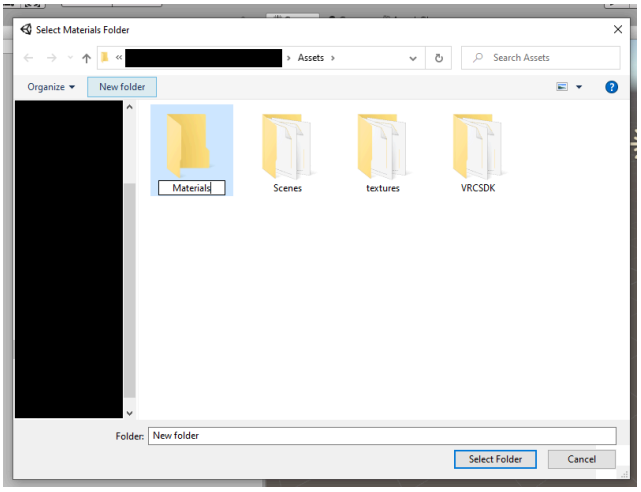
Materials and Shaders

At this point you might notice that your fbx model is devoid of color in the Assets panel. To fix this, we'll need to set up its materials and shaders in Unity. Start this by clicking the little right arrow next to the icon for your model's .fbx file. It'll expand to show a bunch of the model's components.



We'll need to extract the materials from the model in order to apply a shader to them and retexture them. Do this by selecting your material(s), right click them, then select **Extract from Prefab**. A file dialog will prompt you to choose a directory to put your materials in. Just put these in a new Materials folder inside your Assets folder.

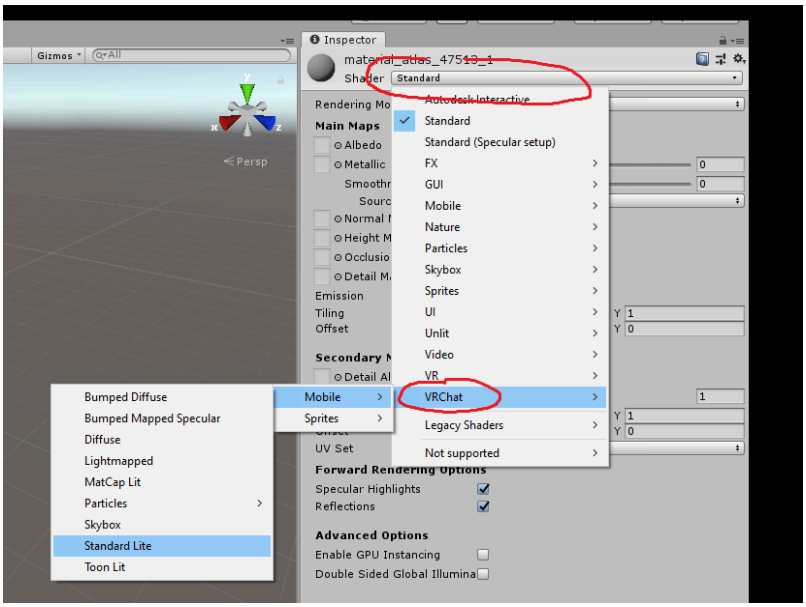




A new Materials folder will appear in your Assets containing the material objects from your model. Let's assign them shaders and retexture them now!

Shaders

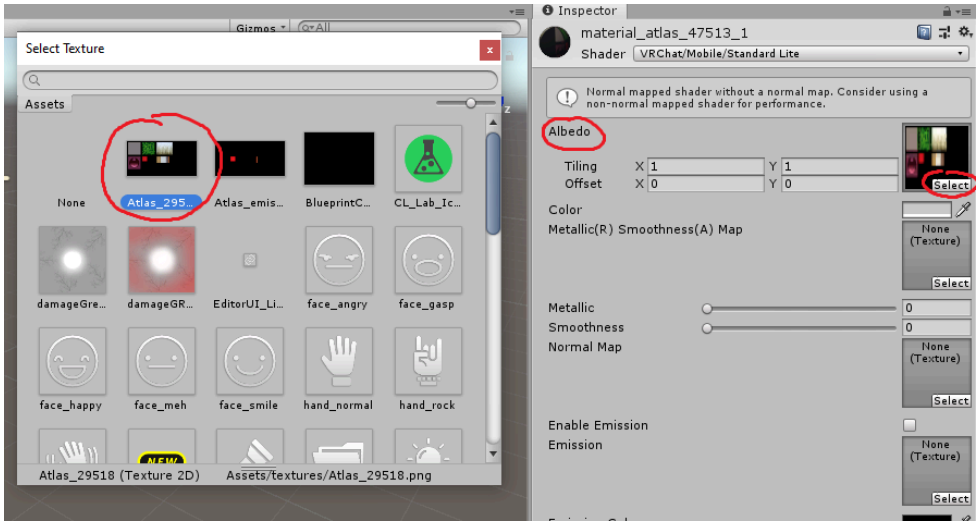
When you click a material in your Assets, the Inspector panel will show a Shader component for the material. By default, the material will have the Standard shader. For your avatar to be Quest-compatible, you should only use one of the VRChat shaders that comes with the SDK.



The Inspector contents will change to show options for your selected shader on the material.

Retexturing your material

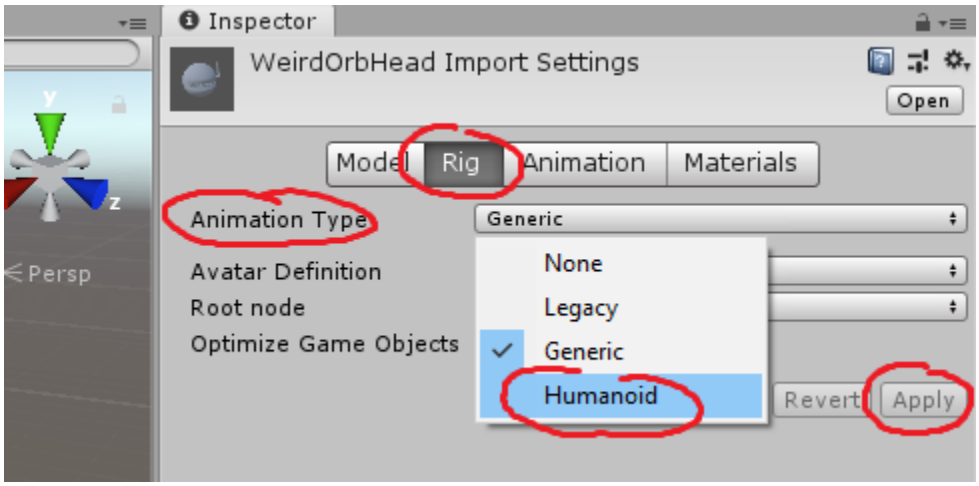
Now that you've selected the shader for your material, we need to retexture it. This is super simple. For the **Albedo** property of the shader, just select your model's atlas texture.



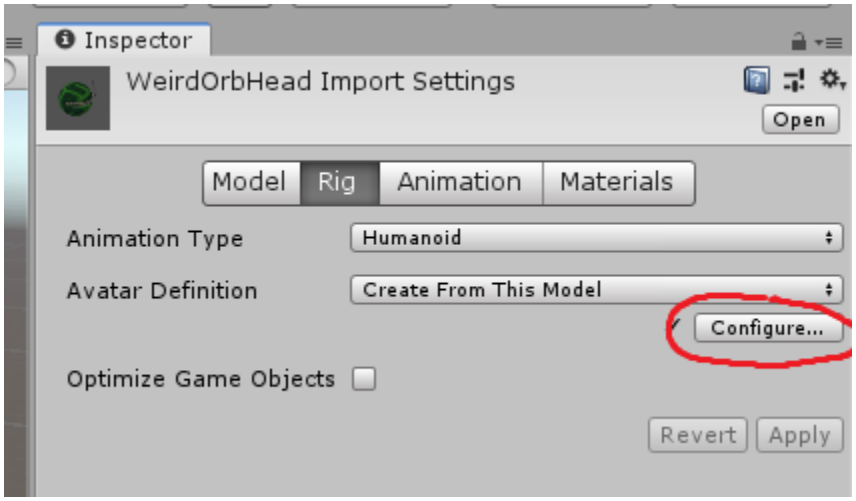
In your Assets, you should notice that the preview icon for your model appears textured again. Yay! There are some other shader options you might like to try playing with, but those are beyond the scope of this document.

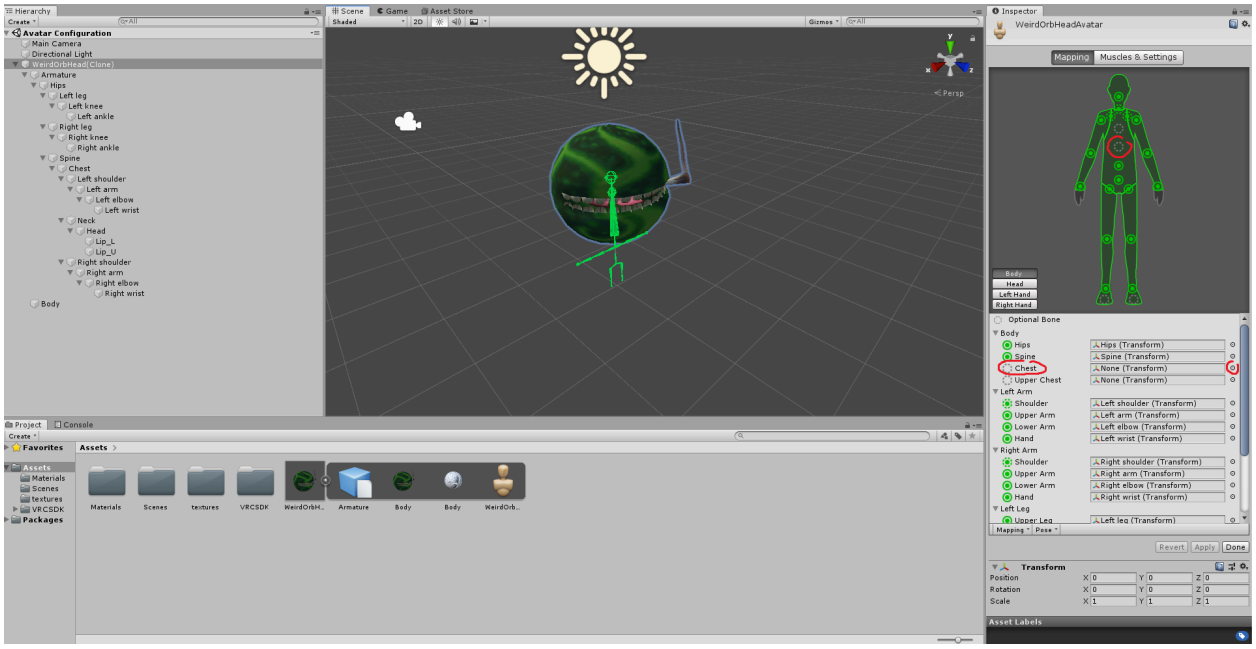
Rigging your model (again)

You've got your model skinned to its armature in Blender. Now you'll need to rig it to VRChat's humanoid skeleton so that it can be animated in game when you move, look around, wave your arms, etc. To do this, first select your model's .fbx file in your **Assets**. In the **Inspector** panel, some settings should appear for the model, including a **Rig** tab! Go to the Rig tab, and for the **Animation Type** option, choose **Humanoid**. Then click Apply.



Next, let's make sure that our skeleton is rigged up correctly. Under the **Avatar Definition** property, click **Configure**. A new screen will appear showing the mapping of your model's armature to the VRChat humanoid skeleton!

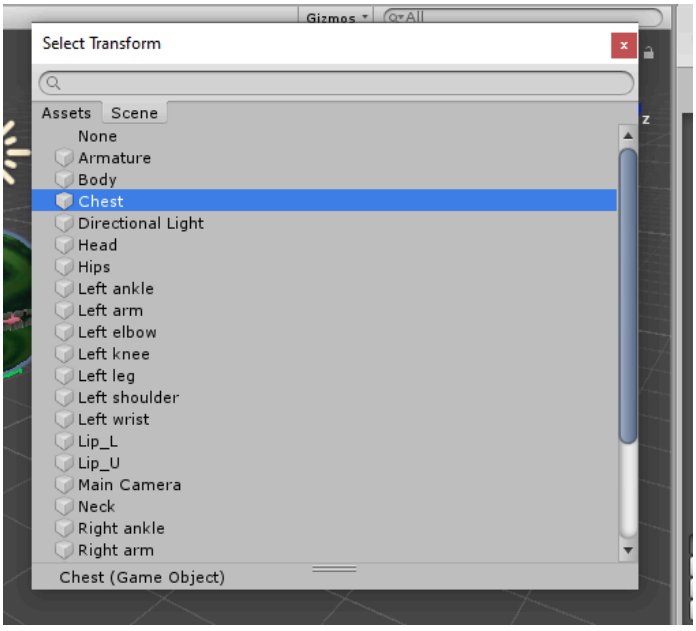




The **Hierarchy** shows the armature for your model. The **Scene** viewport shows the Humanoid skeleton applied to your Armature on your model. The **Inspector** shows the actual mapping of the Humanoid skeleton bones to bones in your armature.

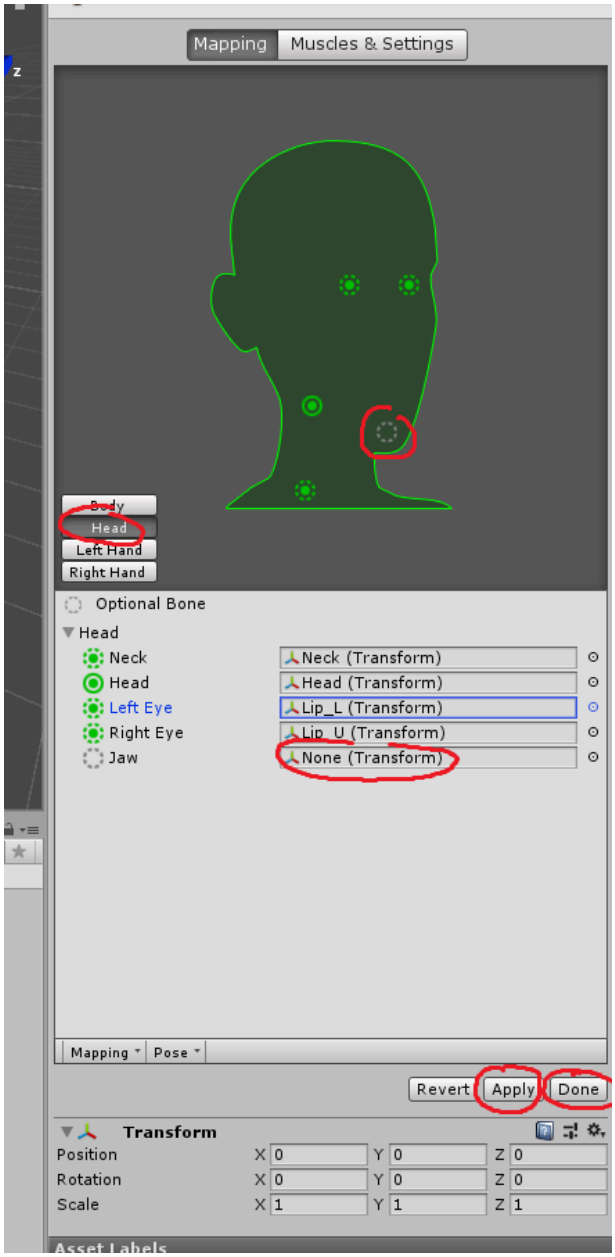
Missing Chest Bone

One common thing you might need to correct is the **Chest** bone. If it's grayed out on the little human-shaped diagram in the inspector, then you need to fix it. To do this, click the dot to the right of **Chest** and **None (Transform)** in the **Inspector**. From the Select Transform dialog that appears, select your armature's Chest bone. Click X in the dialog to save your changes to this.



Unset the Jaw bone

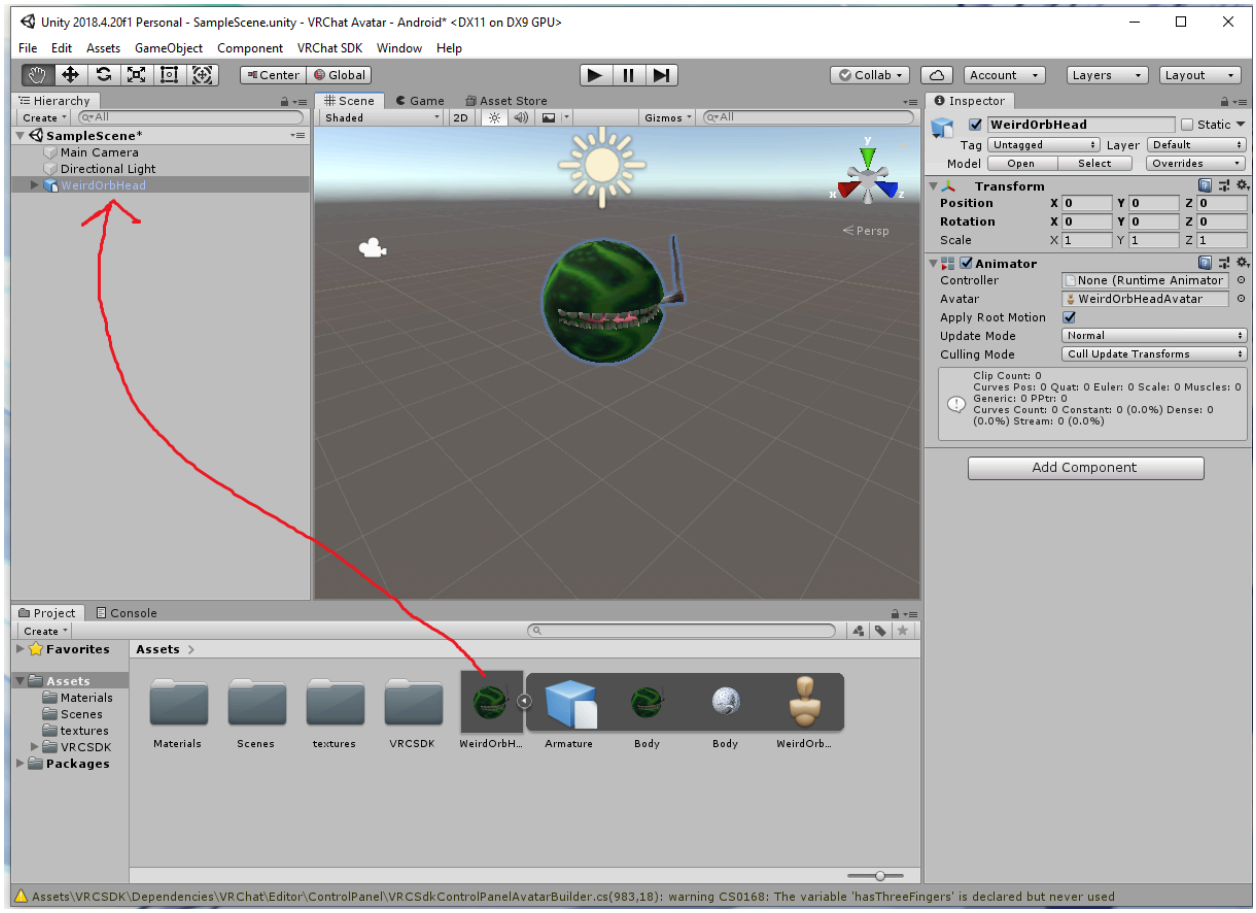
Last, we need to make sure that the jaw bone is not set. If it is, it could mess with our visemes' mouth animations. In the Inspector's diagram, click Head. Make sure that the Jaw bone is grayed out and set to None (Transform).



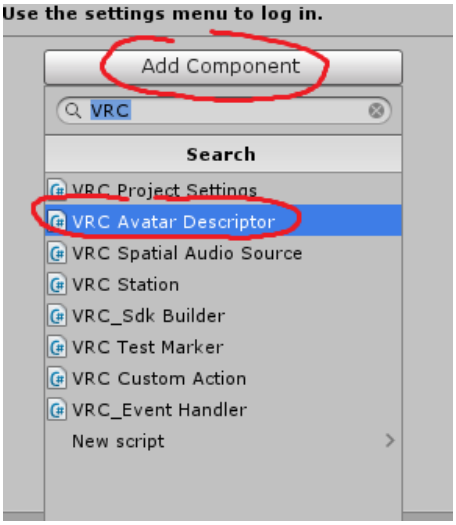
That should do it for rigging your model! If you want, you can click the **Muscles and Settings** tab in the Inspector to test out the rigging's animations. Click **Apply** and then **Done** in the Inspector to save your changes to the model's rigging when you are done.

Set up your Avatar Descriptor

Now we are ready to add your model to your project's scene and establish it as a VRChat avatar object. Start by dragging your model's .fbx file from your **Assets** into your **Scene** in the **Hierarchy** panel. In the Scene viewport, your model should appear at the origin of your scene!



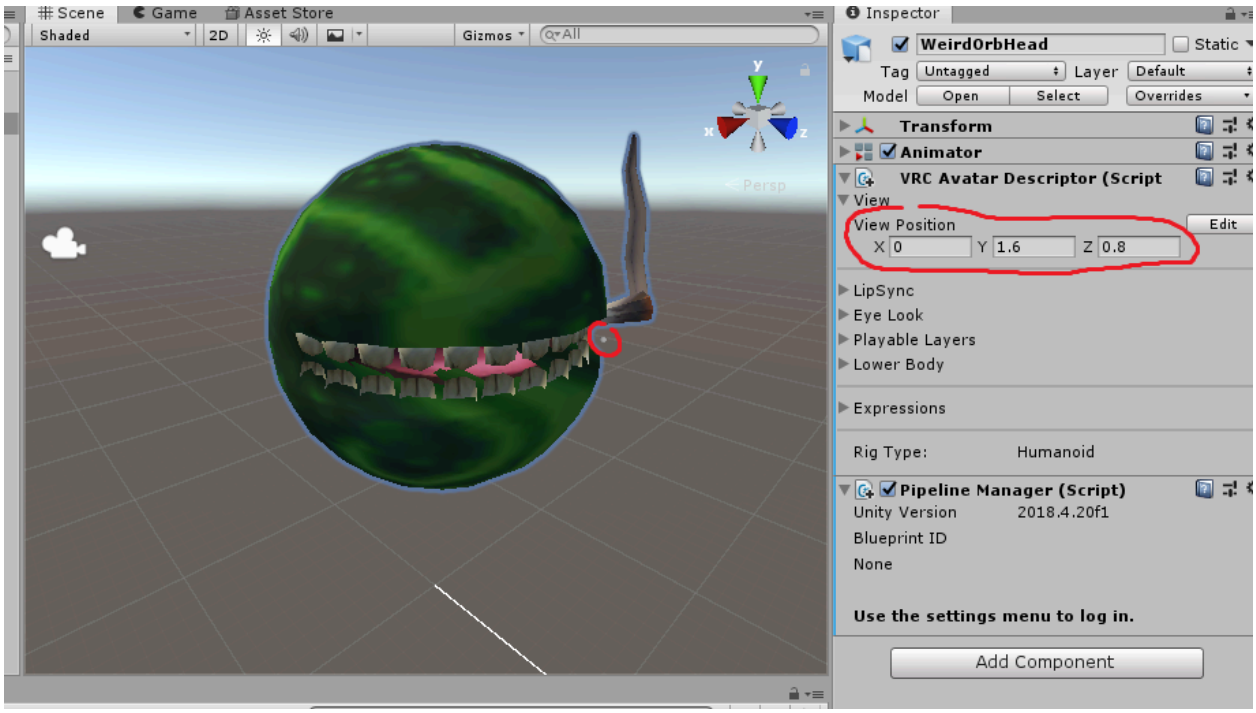
Notice that in the Inspector, your model has two Components in it: Transform and Animator. You don't need to touch either of those. Instead, let's add a third component to set your model as a VRChat avatar. Click Add Component and choose VRChat Avatar Descriptor.



This component has a bunch of interesting settings for setting up your avatar.

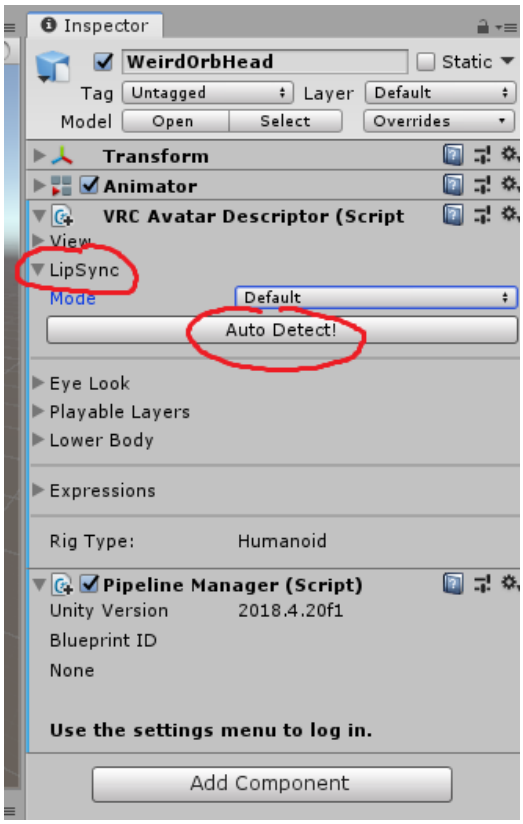
View Position

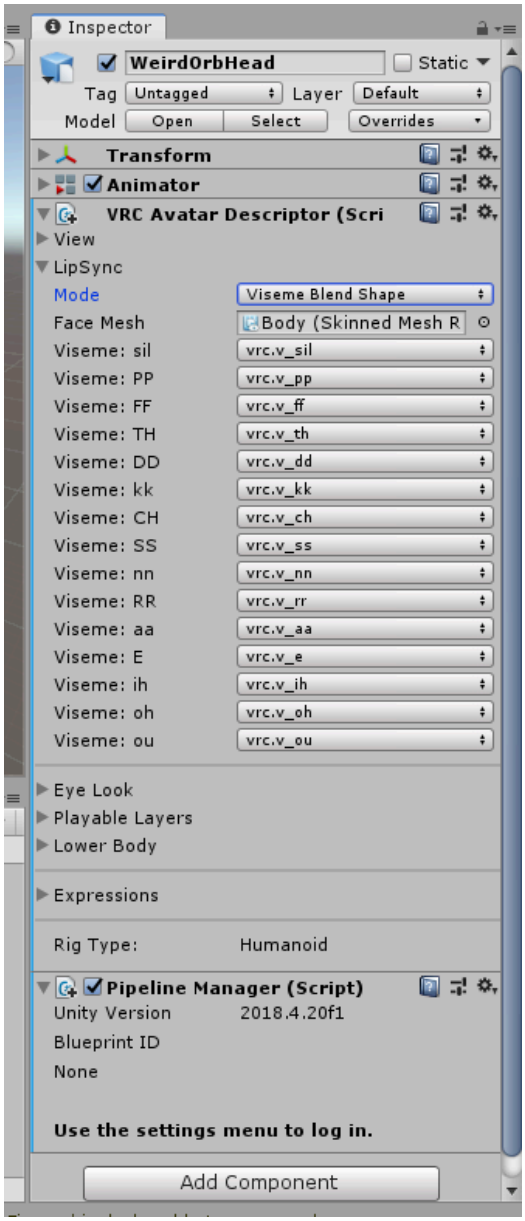
Let's start with the **View Position** under the **View** group. This is used to set where your eyes are on the model (your eyes as in YOU, in real life). In the scene, you'll see this point represented as a tiny gray dot. Adjust its position until it is between the eyes of your model, just above their skin.



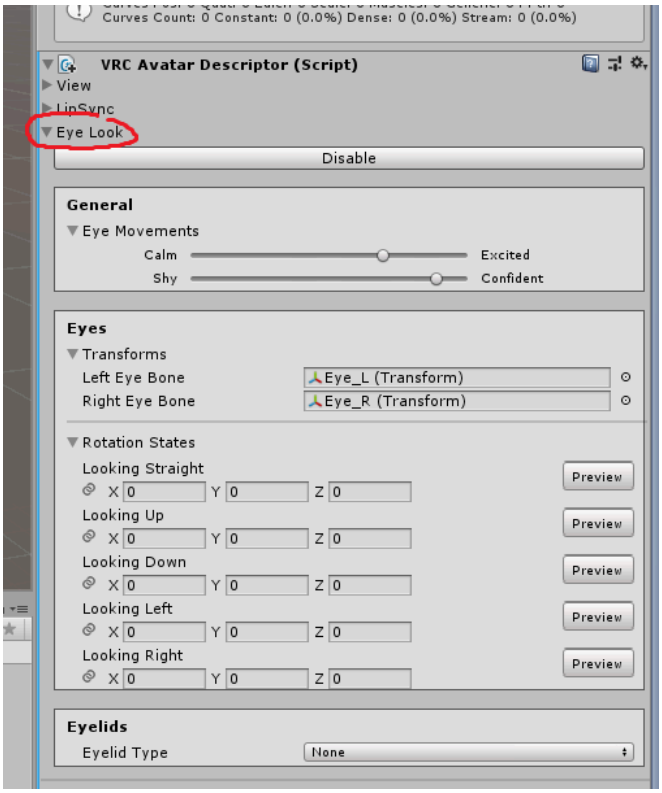
Lip Sync

If you've got visemes set up on your model for mouth movements, you can set these up on your avatar here. It's super easy if you set them up correctly in Blender. Just click **Auto Detect!** and it'll plug in all the visemes you already created with CATS!





Eye Look



If you set up Eye Tracking with the **CATS** tool earlier, you can use **Eye Look** part of the VRC Avatar Descriptor to give your avatar finely controlled simulated eye movement.

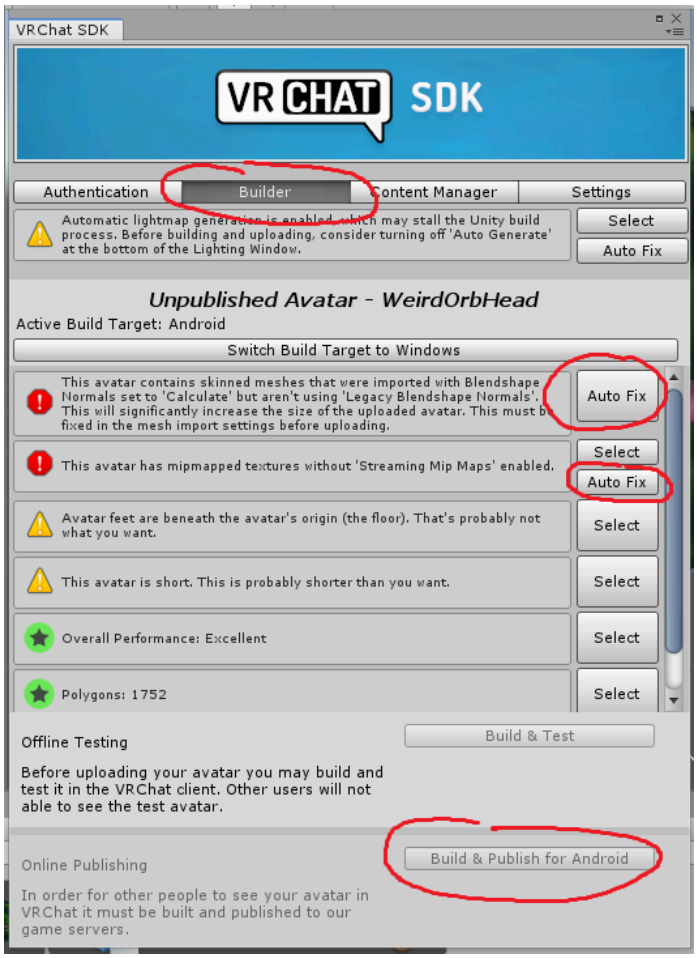
When Eye Look is **enabled**, this will allow them to have their eyes look around on their own, and you can control the degree of movement they have while doing this (The **Eyes** > **Transforms** tools), how often they move (The **Calm/Excited** slider), and how long they stay fixated on other players' faces (The **Shy/Confident** slider). Be sure to set the bones under **Transforms** to the eye bones in your armature and also set the rotation states for looking up, down, left, and right.

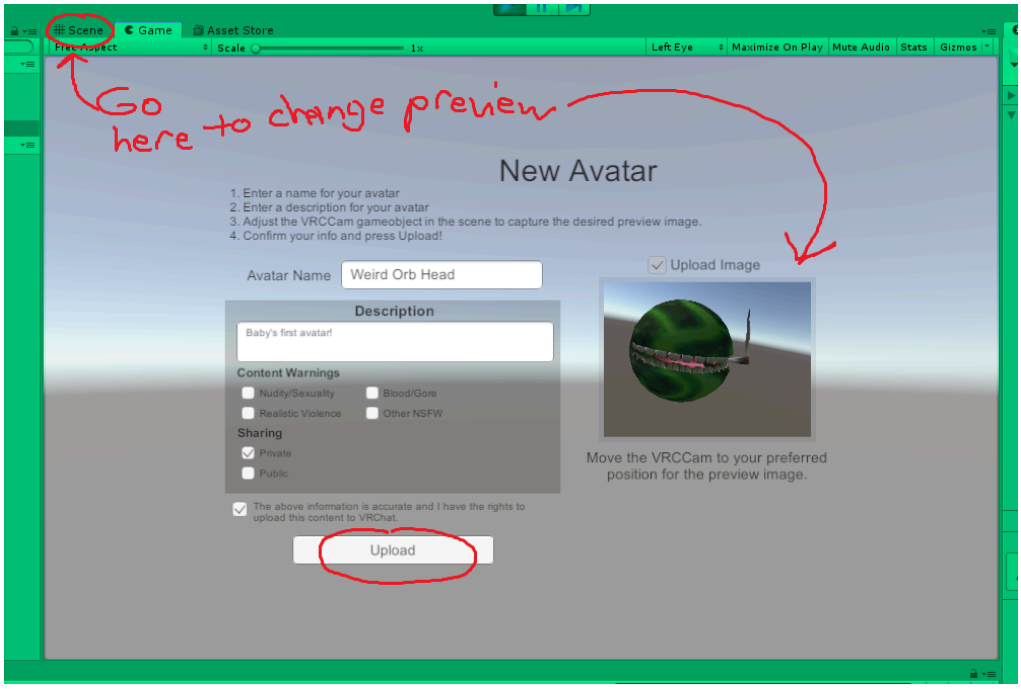
There are also options for fine control over **Eyelids** movement which can be configured using either bones or key shapes.

(Note: In the VRChat docs, it mentions that the Eye Look menu is optional for eyetracking, and that if you don't use it then 2.0 eyetracking will be used by default. However, when I tried that, it didn't work at all. So, I recommend disregarding that bit in the docs and just use the new and improved 3.0 Eye Look menu for setting up both eye tracking and blinking.)

Uploading your finished avatar to VRChat

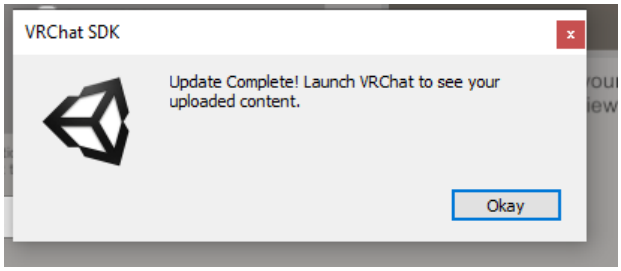
Your avatar should be ready to upload to VRChat now! When you're ready, go to the **VRChat SDK** top menu > **Show Control Panel**. Go to the Builder tab. In this tab, you should see some things that might need fixing before you can upload your avatar. Click **Auto Fix** for any of these errors. This will also tell you the performance rating of your avatar. Once these errors are fixed, click **Build & Publish for Android**.





A progress bar will appear as it builds your final avatar. Wait for this, and eventually the screen will change in the Scene viewport to **Game** mode, asking you to enter a **name** and **description** for your avatar, as well as **Content Warnings** and **Sharing** settings. You can also adjust the camera in Scene mode to set the preview image for your avatar. Be sure to also check the box next to "*The above information is accurate and I have the rights to upload this content to VRChat.*"

When you are done with all this, click **Upload**. You'll see some more progress bars and eventually you'll be informed that your avatar has been uploaded!



Congratulations!

You made your first VRChat avatar! Go test it out for real in VRChat!

If it didn't work and you're stuck, try asking for help on the VRChat Discord. The community there is very active and helpful. There's usually someone on who can help you out.

Addendum: Uploading your avatar for PC use

Now that you've got your avatar uploaded and working in VRChat on Quest, you can also port it for use on PC very easily. Simply change your Unity project's **platform** to **PC** (through **File** > **Build Settings**) and then upload your avatar again.

Now your avatar will work on both Quest and PC! Thanks to **Tupper** for pointing this out on the VRChat forums!