3D Items Part 2 – Accessory Guide

Wogrim's Epic Guide to Creating Accessories with Blender and KK Modding Tools

Before Reading This Guide

Read the Studio Item guide first (and follow along). This guide is focused on how making Accessories is different from making Studio Items, because a lot of the process is the same.

What Are We Making?

We will make the following types of Accessories:

- unskinned Accessory
- skinned Accessory - dynamic bones
- two-piece Accessory

Accessory Hair will be covered in the Hair guide (3D Items Part 3).

Accessory Clothes will be covered in the Clothes Guide (3D Items Part 4).

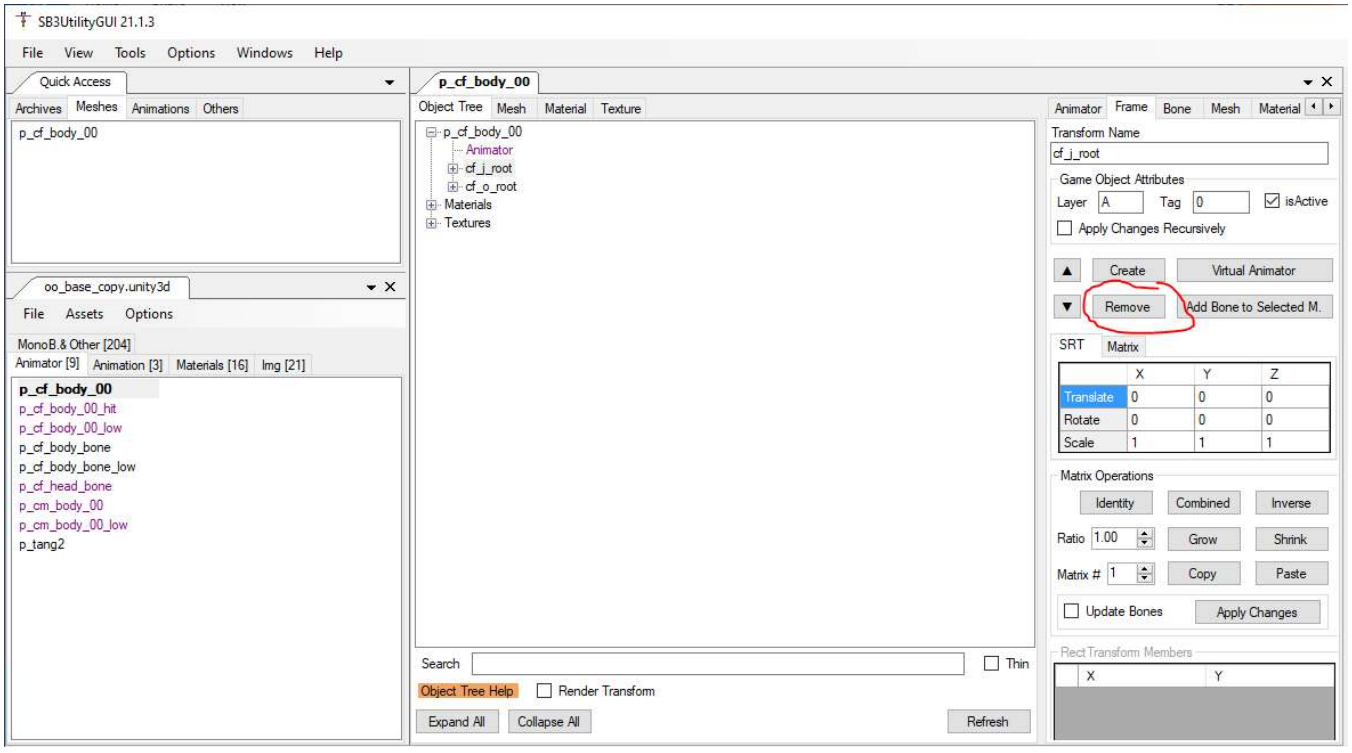
If you want to study the game's Accessories, they are found in **abdata/chara** and the ABs all start with **ao_**.

Preparing Blender: Accessory Version

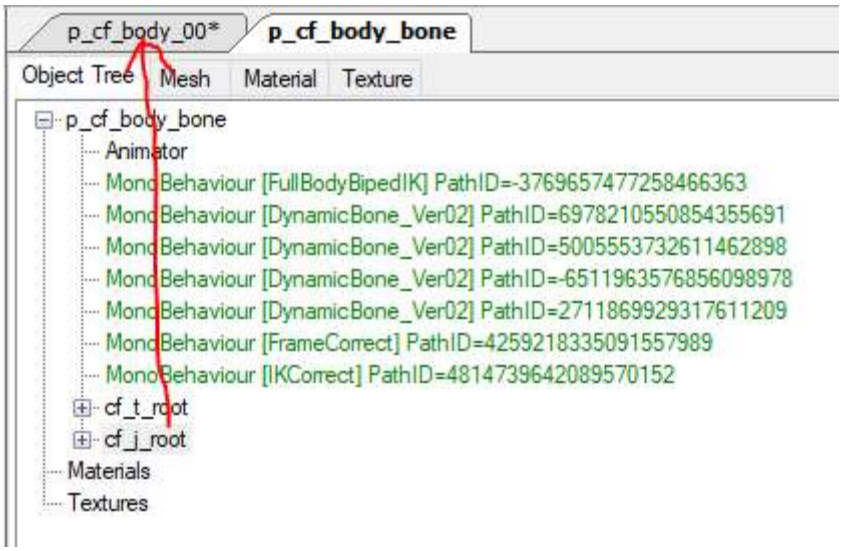
The main difference between Accessories and Studio Items is that Accessories are attached to the character, which means there is extra work to make the Accessory show up with the intended location and rotation (yes they can be moved later, but that is not user-friendly). We will bring the body and head to Blender so we can build the Accessory exactly where it will be in game, but the process is a little different from what you may have seen before.

The problem with the "normal" way of bringing the body to Blender is that the Accessory attach points (parents) are not included with default SB3U export options, or if you do include them ("All Frames" checkbox), some of them have a different rotation than what you see in-game. How does this happen? The body (**p_cf_body_00**) does not have the actual skeleton that gets used for putting the character together; some of the accessory parents are incorrect. The skeleton we need is **p_cf_body_bone** which is located in the same AB (**oo_base.unity3d**). We will have to do something a bit unusual, so make a copy of oo_base.unity3d and do the steps on that instead of your actual game file just in case.

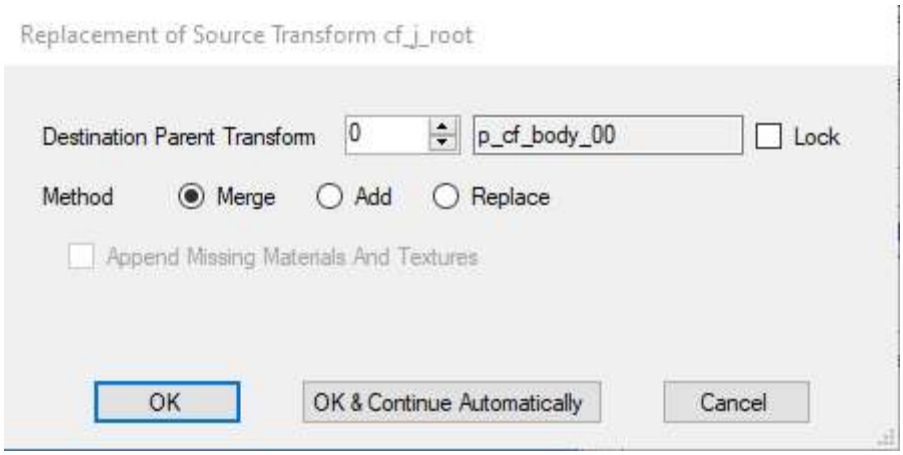
You can't just export the p_cf_body_bone armature (the FBX export button is on the mesh tab and there is no mesh with p_cf_body_bone). You could probably add a mesh to it and then export it (and the body mesh), and in Blender unbind the body mesh from its own armature and bind it to the p_cf_body_bone armature. But instead we'll replace the body's skeleton in SB3U before export. Open p_cf_body_00 and select cf_j_root. Hit **Remove** on the Frame tab to remove the whole skeleton structure.



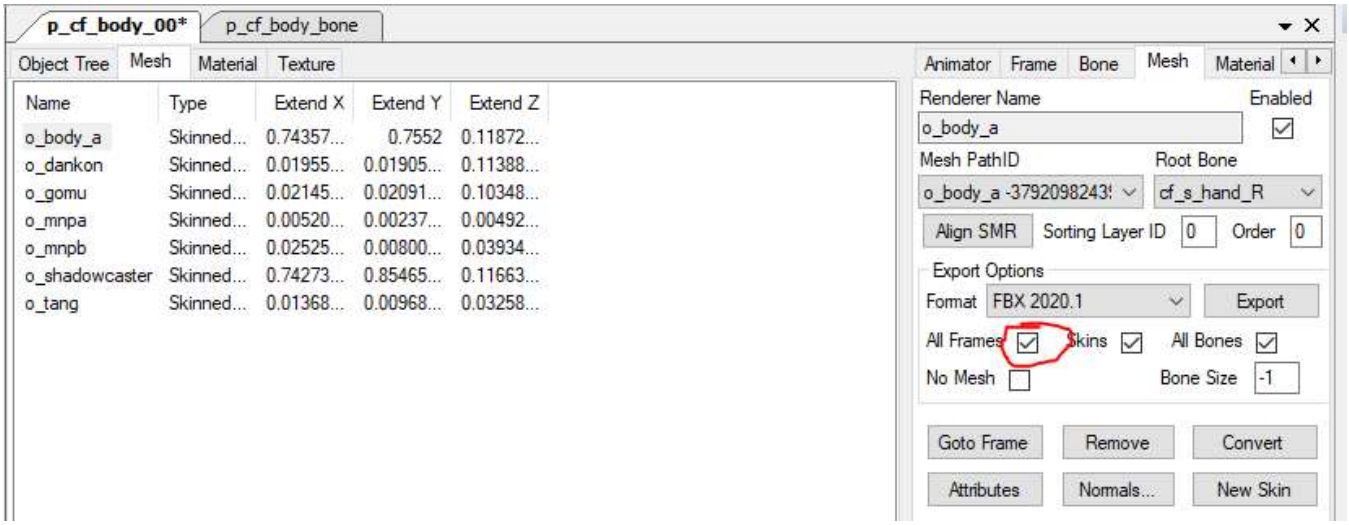
Now open p_cf_body_bone next to it, and drag cf_j_root up to the p_cf_body_00 tab, and drop it on p_cf_body_00 at the root of the Object Tree.



A little window should pop up asking how you want the stuff copied over.

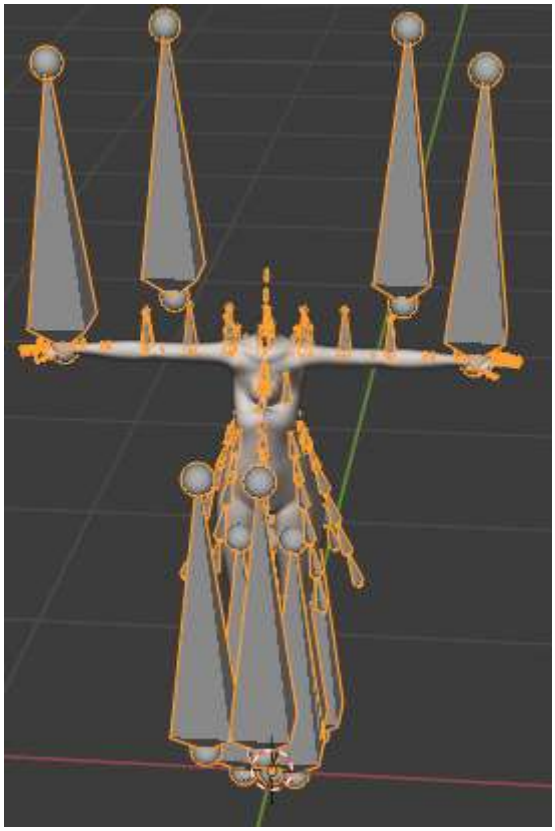


The parent should remain p_cf_body_00. I don't know if the other options matter since we removed cf_j_root earlier, but I just leave it on Merge. So now that the body has the new skeleton, we can export the body mesh (**o_body_a**) with All Frames.



I thought we would similarly need p_cf_head_bone for the head attach points, instead of the skeleton that comes with the head (**p_cf_head_00** located in **bo_head_00.unity3d**). But from what I've seen with ponytail and earring rotations, we actually don't want to do that, so just export the head mesh (**cf_O_face**) with All Frames.

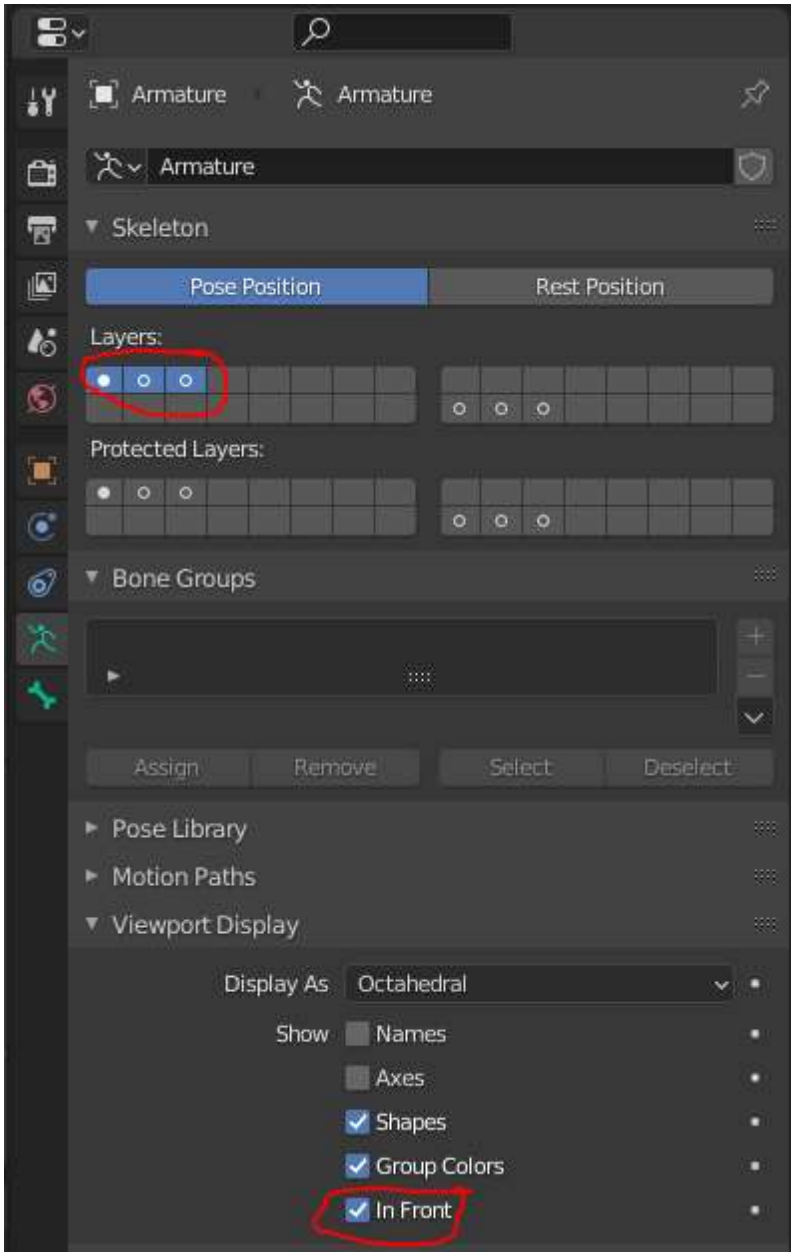
So now import the body FBX into Blender. Bring the scale up to 1,1,1 and the first thing you'll probably notice is there's a bunch of big bones blocking our view. We're going to put bones in different layers so that we can show and hide different sets of bones.



So the first thing I want to get rid of is these big cf_pv bones. I don't know what they're for, but they're useless to us. Select the armature, tab into edit mode, and in the menu go to Select->Select Pattern. Uncheck Extend, type in "cf_pv*", and it'll select all the bones that start with cf_pv. Then go to Armature->Change Bone Layers and pick some out-of-the-way layer to throw all these bones in.

Now that those are out of the way, I'll do the same for all the ones that start with k_f. These aren't as big and obnoxious, but we might as well get rid of them because we have an easy way to hide them all. I put them in a different layer from the cf_pv bones just in case. And do the same for cf_hit bones, which seem to just be positions for colliders. Then do the cf_d and cf_s bones, but keep these nearby.

Now we wanna hide all the "skirt" bones but some of them went away with the cf_d group, and maybe cf_s also. So go to the armature properties and shift click your cf_d and cf_s layers so they are shown as well. Also, Viewport Display -> In Front so you can see the bones hiding in the middle of the body.

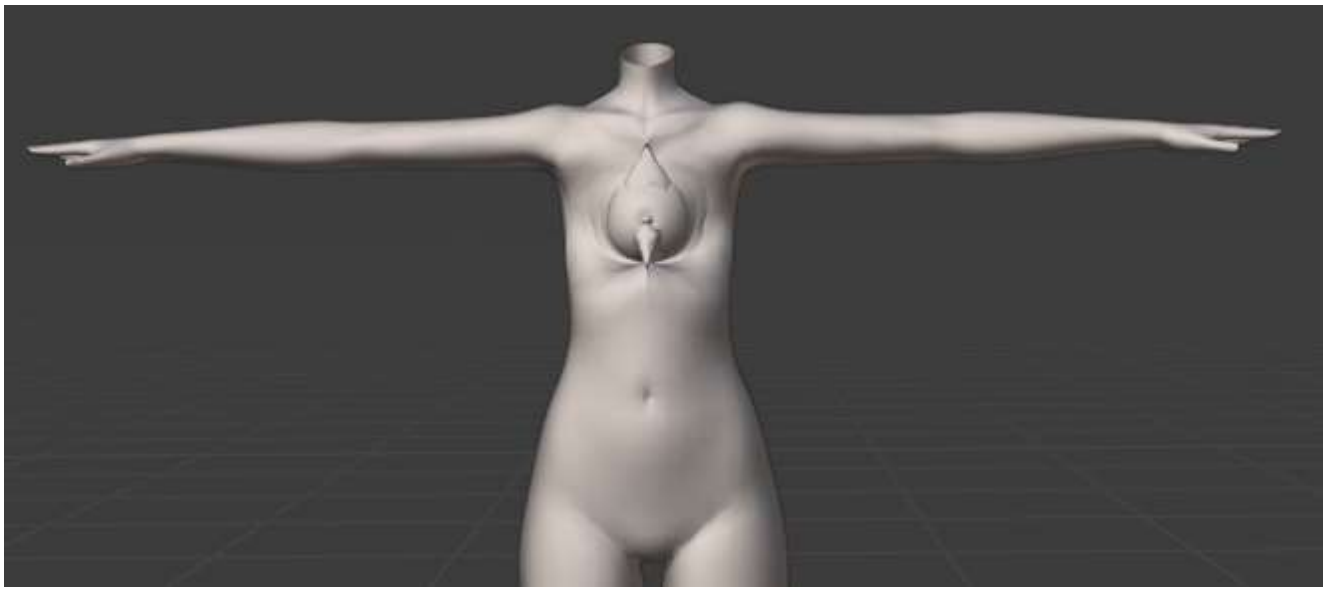


So now do another pattern select for "*sk*" to get all the skirt bones, and then put them in another out-of-the-way layer. Now, we want to easily be able to look at all the Accessory parents, which start with **a_n**. So do a pattern search to find all of those and put them in an easy-to-reach layer.

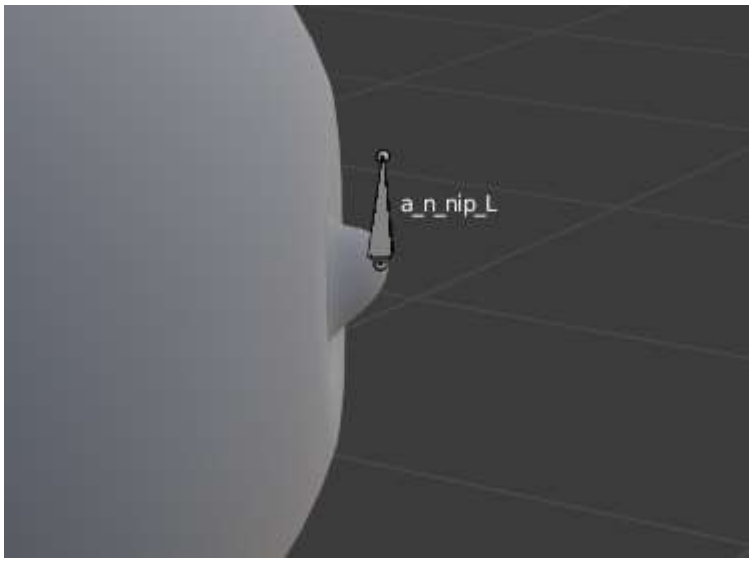
And the last thing we'll get rid of is cf_o_root and all of its children. So select that and Select->Similar->Children to select all those, and throw it in a bone layer. We won't be needing it.

So the default layer pretty much just has cf_j bones which you might rotate later in pose mode to test stuff, and the layer below it has the accessory attach points, which makes it easy to see where you're attaching your Accessory to.

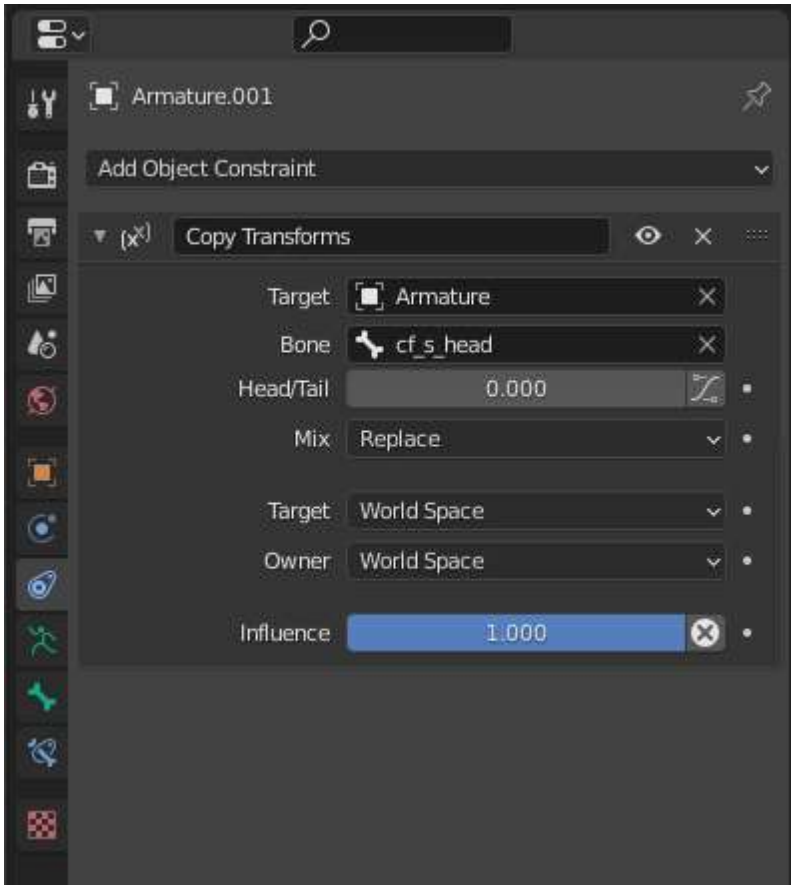
Now if you go back to object mode you may notice there's a weird single-boob thing going on. This is not a problem with the mesh; for some reason the armature is in a pose.



So go into pose mode, show all the armature layers, and select all the bones. Then do Pose->Clear Transform->All and the mesh should look normal. BUT some of the boob bones are in the wrong place (important for us, the nipple attach points are not in the right spot). I don't know of a way to fix this, so you'll have to also bring in the body mesh with its normal skeleton if you want to properly place nipple accessories (the attach points seem to be right). Don't do the bnip025 pre-export SB3U scaling thing you may have seen, you want to be able to see the nipple bumps for placing your accessory.



Bringing in the head is pretty much the same, but there's less clutter. Bring the scale to 1,1,1 and clear the pose in case there is one (doesn't seem like it, but doesn't hurt). Then move all the accessory parents (a_n) to the same layer spot as the ones on the body. Then add an object constraint to the head armature which matches its transform to the cf_s_head bone on the body's armature.



This is sort of an unofficial way to attach the head to the body, and the same method will be used to attach our Accessories to their parents in Blender to get perfect placement. Rename the body and head armatures to something easy to identify. So if you're looking at the body and head with only the accessory parent layers showing, it should look like this.

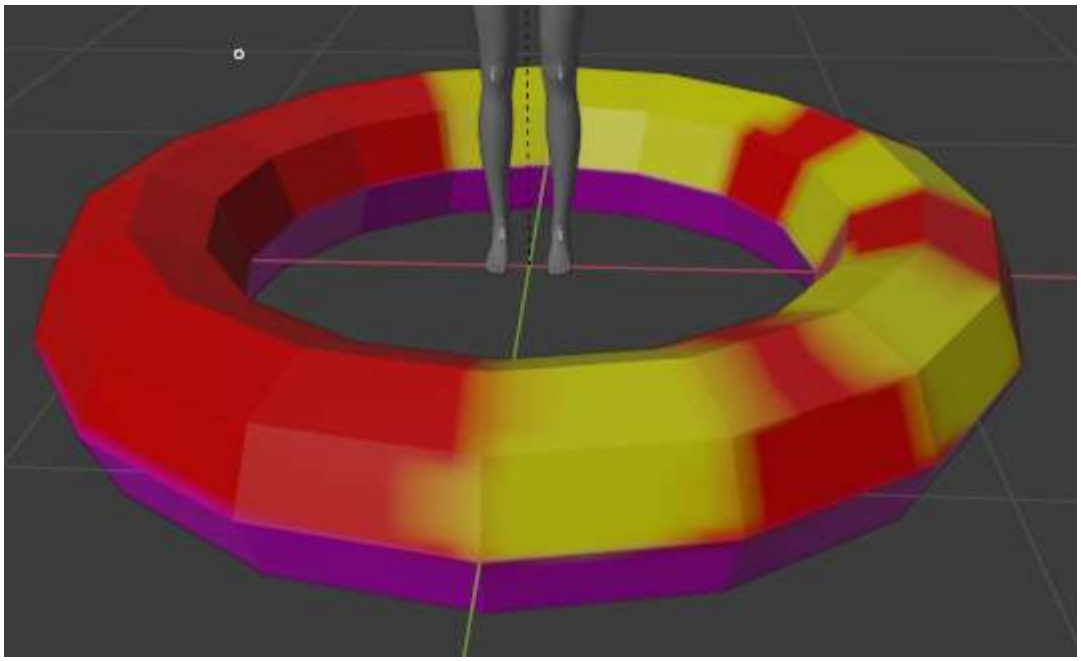
Preparing KK Modding Tools: Accessory Version

Similar to making Studio Items, you duplicate the Accessory Example folder and put the copy in the Mods folder to be the folder for your mod. Rename it, delete everything but the folders, manifest, and list file. Fill out the manifest.

Assuming you set things up already to follow the Studio Items guide, there isn't anything else you need to do. Since Accessories get attached to bones which have rotations (and are children of other bones that have rotations) you generally can't intuitively figure out if your item has the correct rotation or position. (I really should modify my preview scene to work better with Accessories because they can be partially or completely in the floor). But it is still good for making sure the item has the right scale and for the material preview.

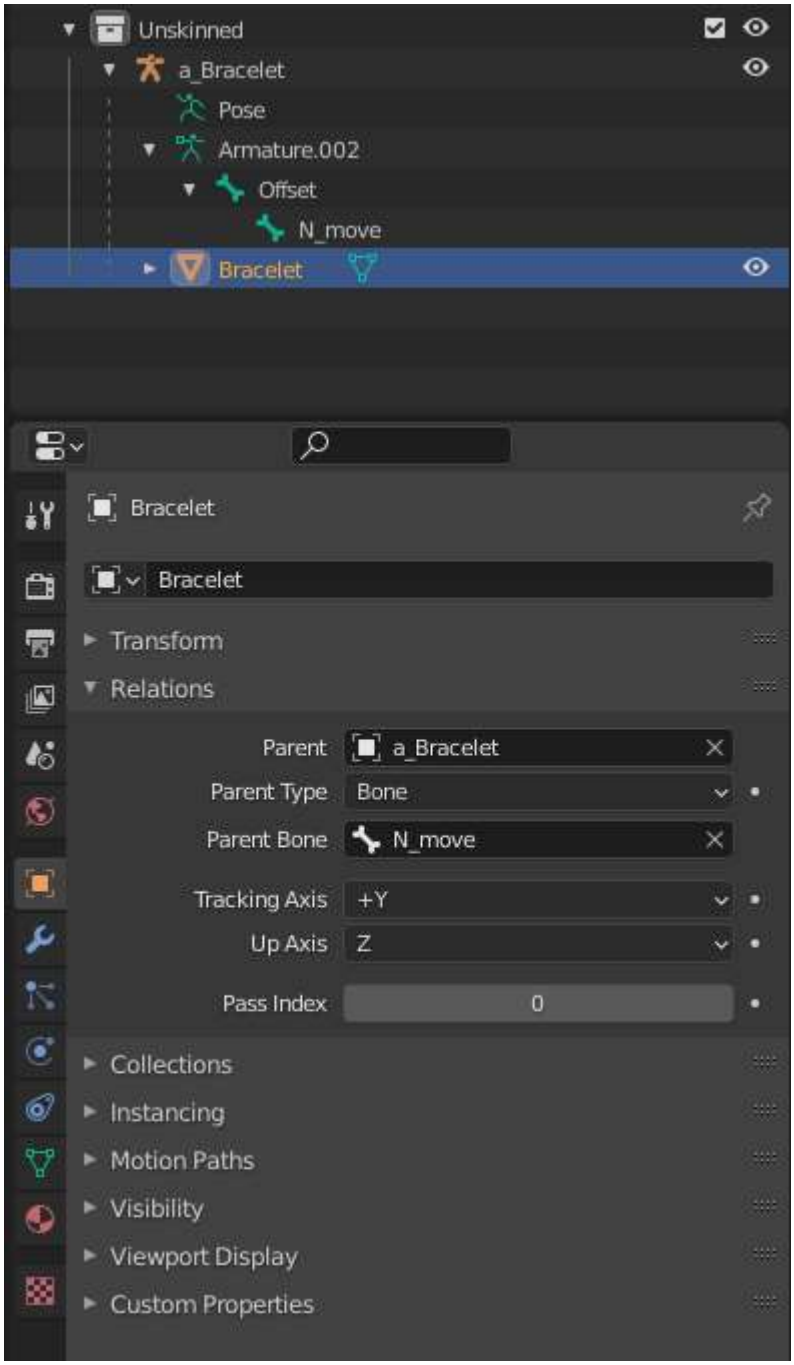
Unskinned Accessory: Blender

Make or import an unskinned mesh that will be your Accessory. With the UV mapping and textures. Make sure you start with something that isn't too simple to tell if it's rotated the right direction when you see it in game; we went through a lot of effort so we can get the right rotations. So here's a quick bracelet I made, just a mesh so far; no armature. Don't worry about size or rotation yet. Hide the body for now if it's in your way.

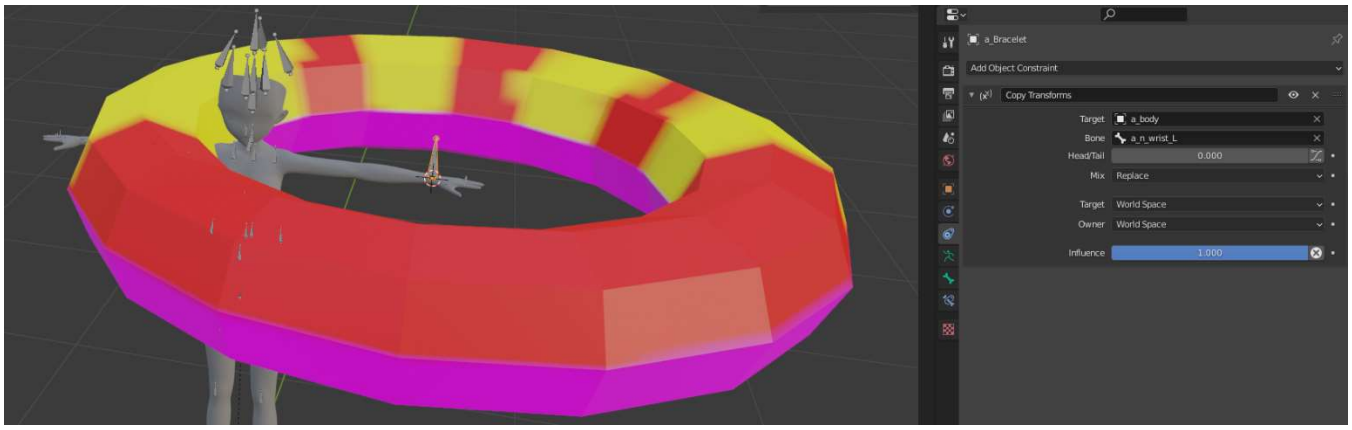


The hierarchy you usually make for Accessories is a little different because it must have a transform called **N_move** (exact spelling) for the Accessory to be moved around in Maker with the controls. The simplest thing you'd do in Blender would be to have an armature with 1 bone called N_move, and make that bone the parent of your mesh(es). But we will add another bone to get the Accessory to the desired location and rotation by default. You could technically do this on the mesh, but it is less user-friendly when N_move isn't at a good rotation point for the item.

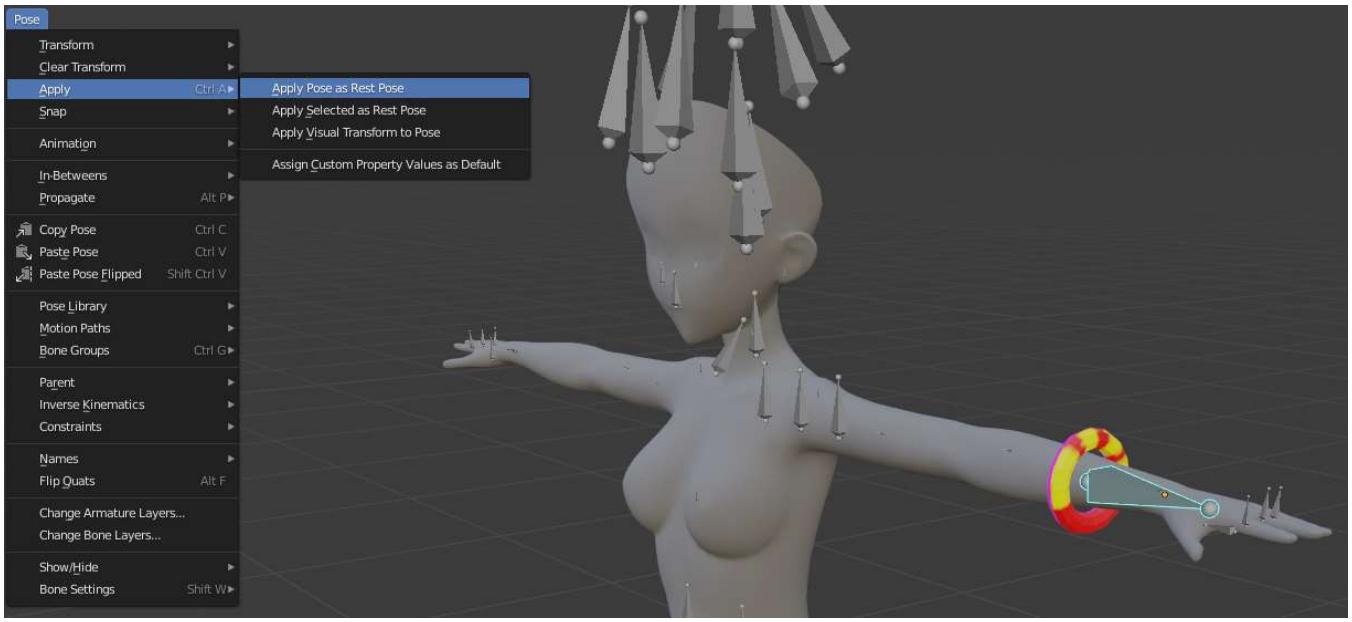
So make yourself an armature, and rename appropriately for your item. Bring the bone's tail straight down so it's not so big. Duplicate it (Shift + D) so you have 2 bones in the same spot. Rename the bones Offset and N_move, and make Offset the parent of N_move (keep offset, not connected). Make the N_move bone the parent of your mesh, just like how I made the Meshes bone the parent of the mesh in the Studio Items guide. Your item's hierarchy should now look like this. The armature, mesh, and bones have not been moved, rotated, or scaled yet.



Now, we're gonna show the body and get our Accessory's position, rotation, and size correct. If you haven't figured out what your item's parent will be, do that. I will be attaching my item to a_n_wrist_L. On the armature's object constraints, Copy Transforms of the bone that will be the parent. It should get moved and probably rotated.

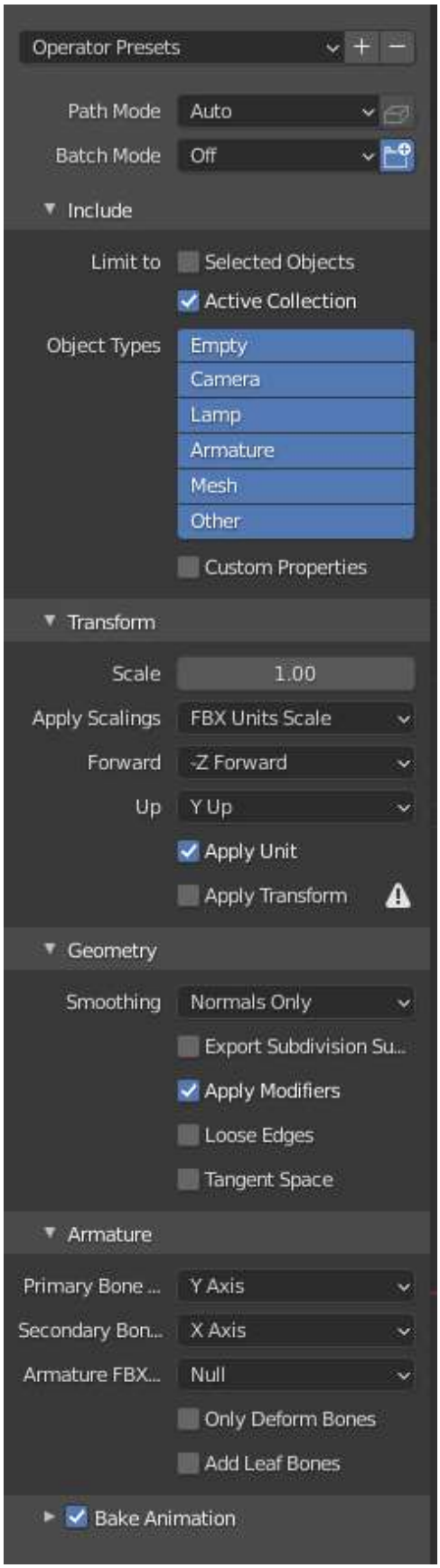


Now scale your mesh to the right size (or as close as you can figure without it being in the right location/rotation). Then take your item's armature into Pose Mode and get the location and rotation correct ONLY by changing the Offset bone (do some of both so you can make sure you get the whole process right). You can hide the N_move bone on a different layer if it's getting in the way. If you need to tweak the scale some more, do it on the mesh. When you are happy with it, apply the scaling on the mesh. Go back to the armature and into Pose mode, and go to Pose->Apply->Apply Pose as Rest Pose.



Now we're pretty much ready to export. Disable the armature's Copy Transforms constraint.

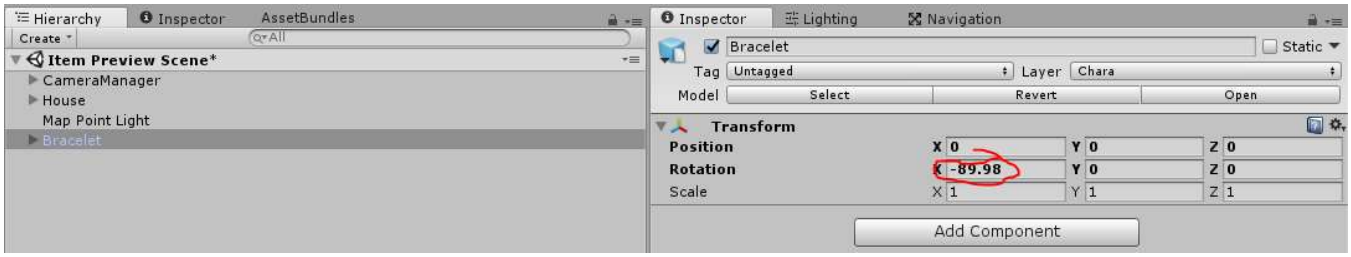
Export of an Accessory is a little different from a Studio Item. A Studio Item gets placed in the world with no rotation, so we did a 90 degree rotation thing before export to counter-act the 90 degree rotation caused by the Z-up Blender to Y-up Unity axis system. The Accessory just gets attached to its parent, so we just want to keep the relative rotation we've done all this work to set up. So we DO NOT do the 90 degree pre-export rotation thing for Accessories. It's also good that we don't do any extra stuff so that we can verify the position and rotation in-game are same as in Blender, just by re-enabling the Copy Transforms constraint (it just needs to be disabled during the export). There might be a way to change export settings so the item isn't rotated in Unity, but I'm just going to use the same export settings as Studio Items and get rid of the rotation in Unity.


Operator Presets
Path Mode
Auto
Batch Mode
Off
Include
Limit to
Selected Objects
Active Collection
Object Types
Empty
Camera
Lamp
Armature
Mesh
Other
Custom Properties
Transform
Scale
1.00
Apply Scalings
FBX Units Scale
Forward
-Z Forward
Up
Y Up
Apply Unit
Apply Transform
Geometry
Smoothing
Normals Only
Export Subdivision Su...
Apply Modifiers
Loose Edges
Tangent Space
Armature
Primary Bone ...
Y Axis
Secondary Bon...
X Axis
Armature FBX...
Null
Only Deform Bones
Add Leaf Bones
Bake Animation

Unskinned Accessory: Unity

Same as Studio Items; import your FBX and textures into Unity by dragging them from Explorer to the appropriate folders in your mod. Nothing new to worry about for import settings.

So drag the FBX thing into the Hierarchy like with Studio Items, but this time it should come with a rotation.



Zero that out.

The rest is pretty generic stuff that should be familiar to you:

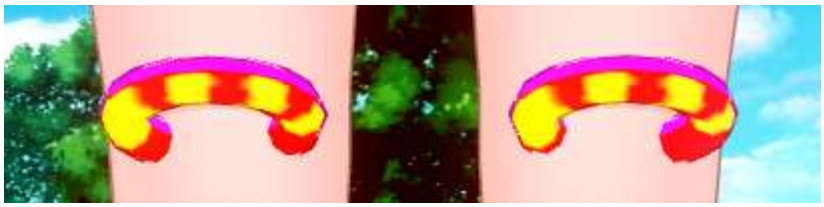
- verify item is on correct layer (probably always Chara layer)
- create a material for your item with appropriate shader (typically **Shader Forge/main_item**)
- fill in material textures and variables
- drag material onto your item's GameObject(s) with Mesh Renderer
- add appropriate MB to item (**ChaAccessoryComponent**) and fill it out, generally:
    - set size of RendNormal and drag in appropriate GameObjects with Mesh Renderers
    - select which color pickers will be used
    - choose default color picker colors
- make prefab
- make prefab instance to check preview
- assign prefab to AB (naming: **chara/author/modname.unity3d**)
- assign thumbnail to AB (can be same or different) if you already have it
- rename and fill out list file, note:
    - get category number and column titles from **abdata/list/characustom/00.unity3d**
    - MainAB is the AB you assigned the prefab to
    - MainData is the name of the prefab
- build ABs
- build zipmod
- test item in-game

Unskinned Accessory: Result

It came out in the same position and rotation as I made it in Blender, just how I wanted it. I like it so much I added another one to the character and set the attach point to the right wrist. But something funny happens.

(I had set the default colors to red/yellow/magenta for easy comparison to my color mask version in Blender). So let's look in Blender at how this happened. I duplicated the item and changed its copy transform constraint to a_n_wrist_R. I circled the wrist attach points.

This happened because for whatever reason the wrist parents are facing the same direction. On the other hand, many of the accessory parents with a left and right version have a "mirrored" rotation which will generally put the item how you want it when you switch sides. This doesn't mirror the item itself, just rotates it for a mirrored "this way up, this way forward" type of thing. Here's upper arms and knees.

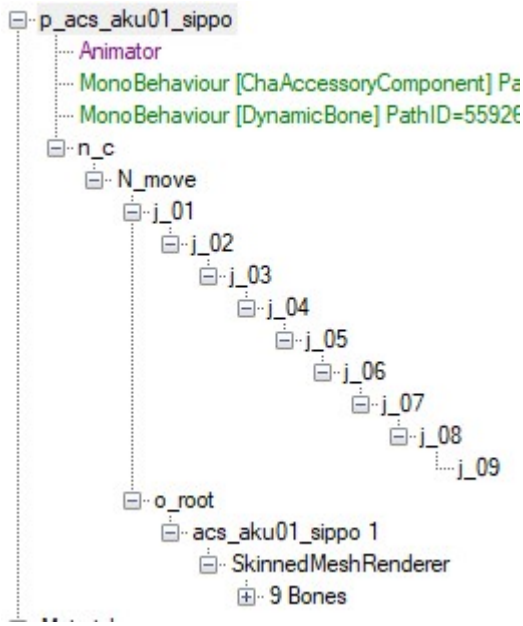


So for some attach points you may be happy with only one version of the accessory. Or you might need to make a second item which is a mirrored version of the accessory for the other side. And/Or you may need to give the other-side version a different position/rotation.

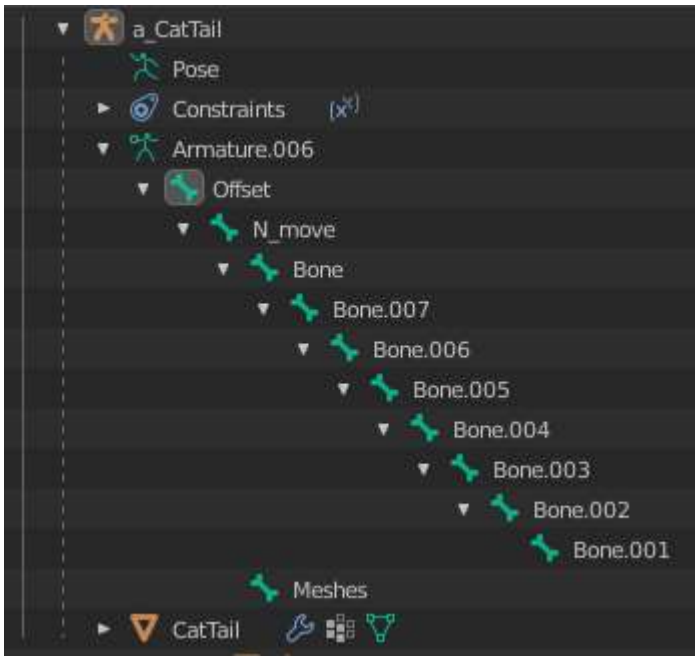
Anyways, make sure to verify functionality (color pickers, being able to move/rotate/resize accessory) before moving on to something more complicated.

Skinned Accessory: Blender

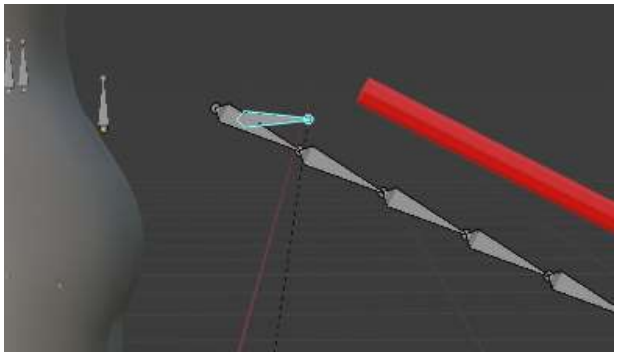
So you might make a skinned Accessory with a hierarchy like an unskinned Accessory, but it has extra bones under N_move that the mesh is bound to. Or the other way to look at it, it's like a skinned Studio Item but the skeleton and mesh have N_move as their parent (and an Offset above that). And that is how you'll see the hierarchy if you look at game items, and that is how you will see me do it in this guide.



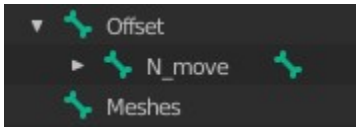
But if you do this in Blender and have everything ready (making a cat tail btw)…



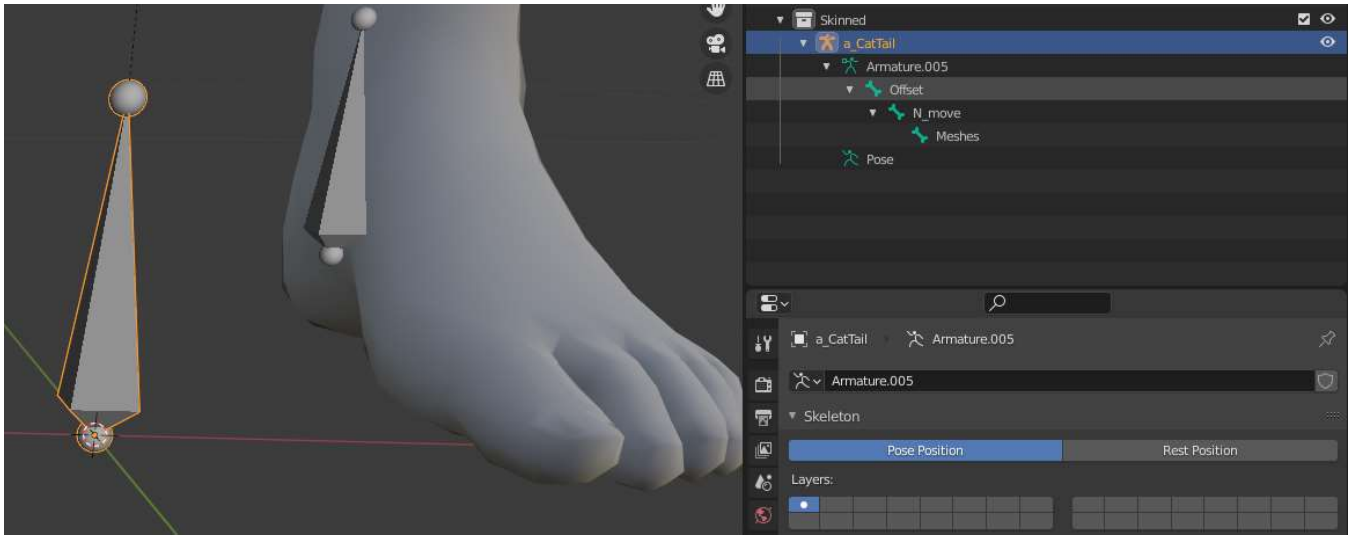
And then decide you want to change the placement so you move the offset bone in Pose mode, you will see the mesh becomes “disconnected” from the bones. This seems to be because moving the offset moves both the skeleton and the mesh parent so it’s getting extra movement. And same problem with rotation/scaling.



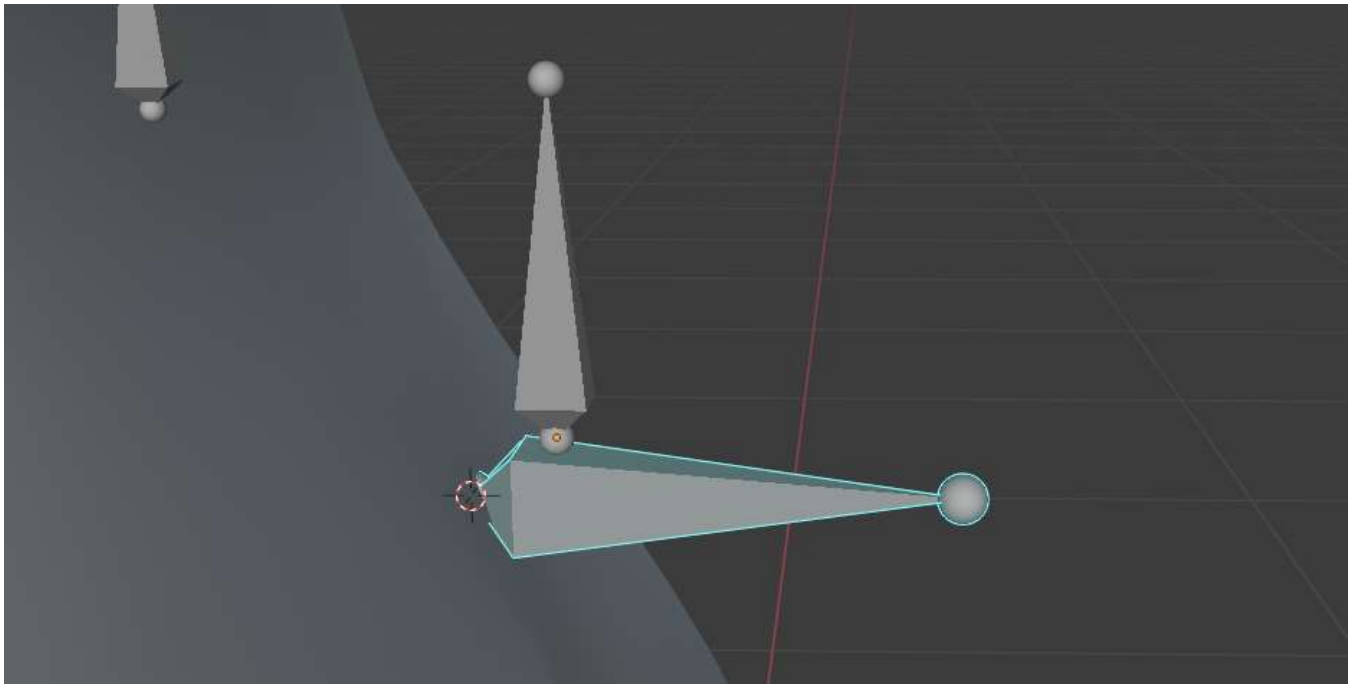
If you make your item without changing Offset, it will come out fine. But with this in mind, I consider it better to instead make Meshes a sibling of Offset, and you won’t have this problem (this is how I do it for Accessory Hair in the Hair guide).



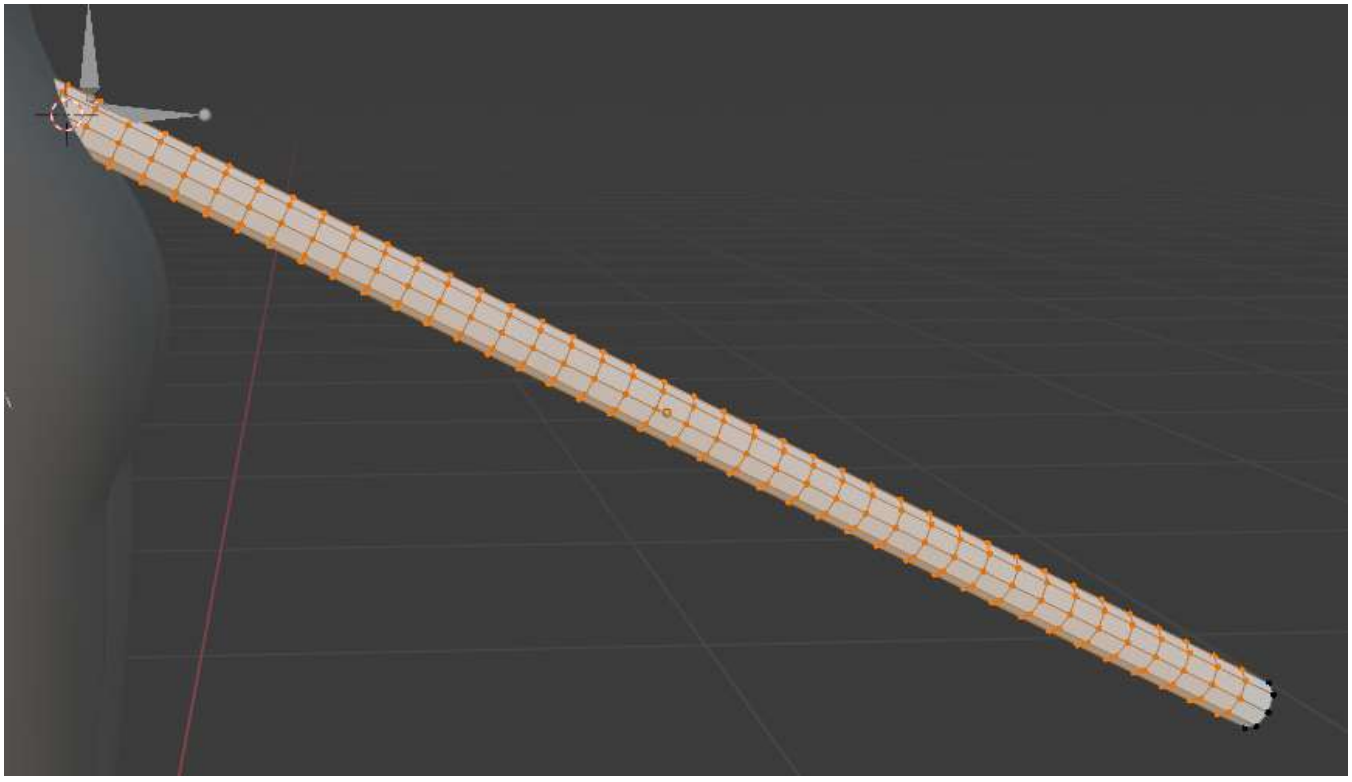
Anyways back to making stuff. Last time I made the Accessory, attached it and then moved/rotated/resized it to get to the right location. This time I will create the hierarchy first, attach it, and model the Accessory on-location (for Science). So I'll start with creating the armature, without the bones the item will be skinned to.



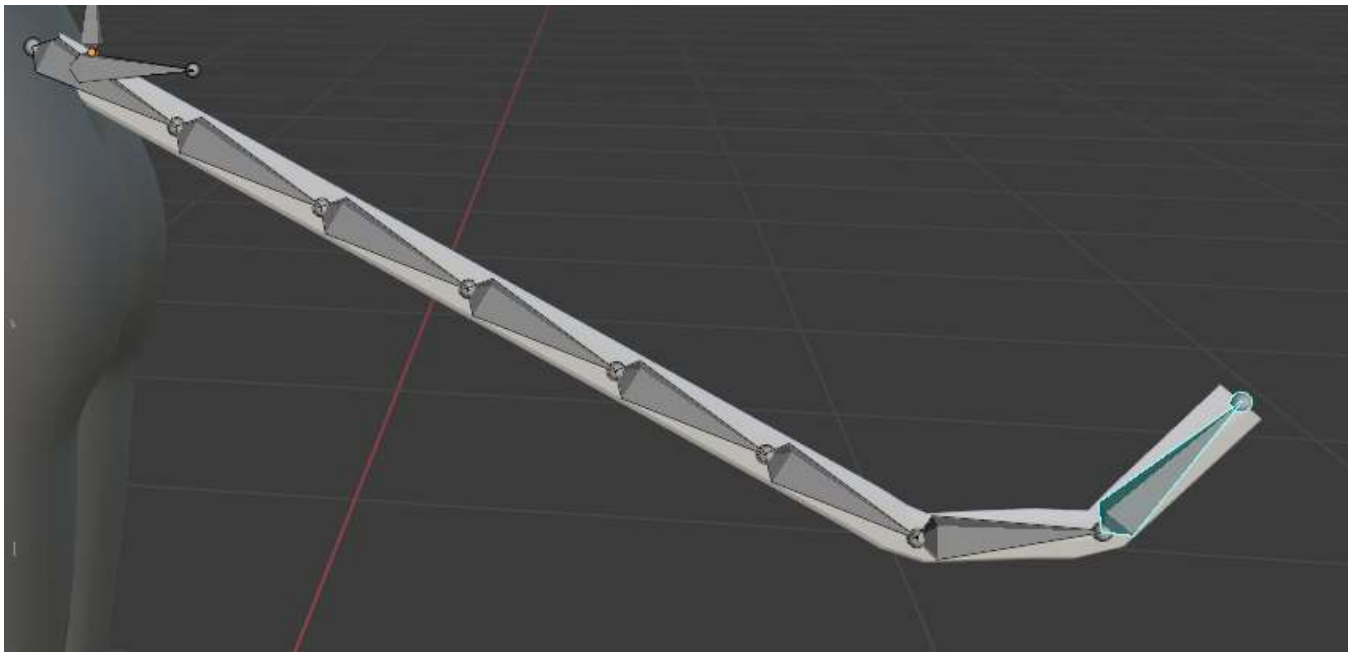
Then I put Offset, N_move, and Meshes in different bone layers so I can get to them easier when I need them, since they're all right on top of each other. I put the Copy Transforms on the armature and set it to a_n_waist_b since I'm making a tail. In Pose Mode I move the Offset bone to where I want N_move to be; at the base of the tail (well, where it will be). This is just because it is the most intuitive place for someone to move/rotate the tail from if they don't like my placement. Bring cursor to selection for when you start modeling (soon).



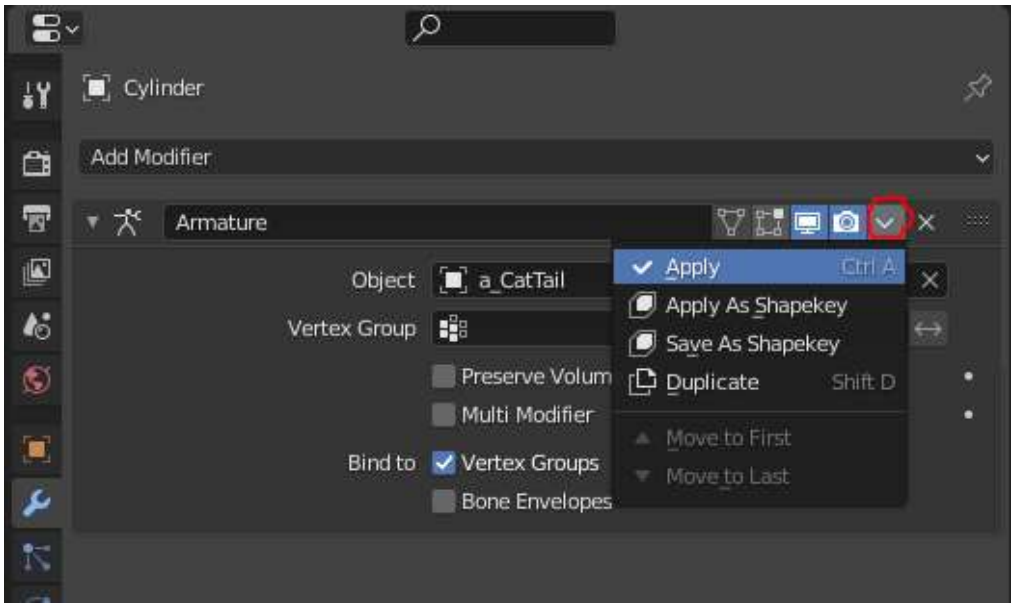
So apply the pose as rest pose, and then create the model on-location.



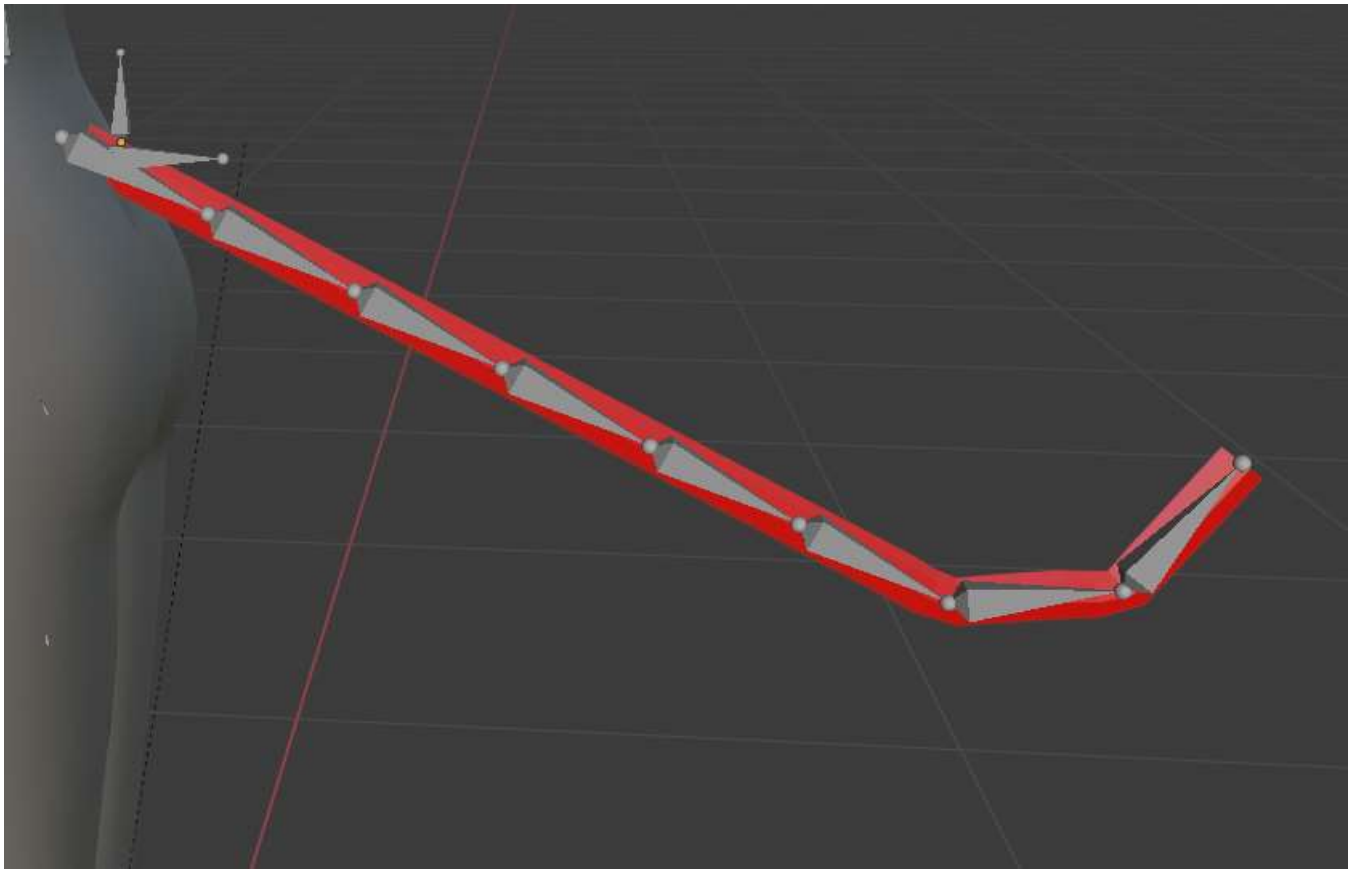
I made a nice and simple cylindrical tail. I want a curve at the end of the tail, but it is easier to do that after skinning than modeling the curve. You could UV unwrap it and make textures for it at this point, but I'm just going to have it as a 1-color item. Then I create the bones that the mesh will be skinned to. Remember, these go under N_move. I apply the transform I still had on the mesh, and I'm ready to skin the mesh to the bones. I didn't bother renaming them all. Do not skin the mesh to Offset, N_move, or Meshes. This is just a simple job for Automatic Weights.



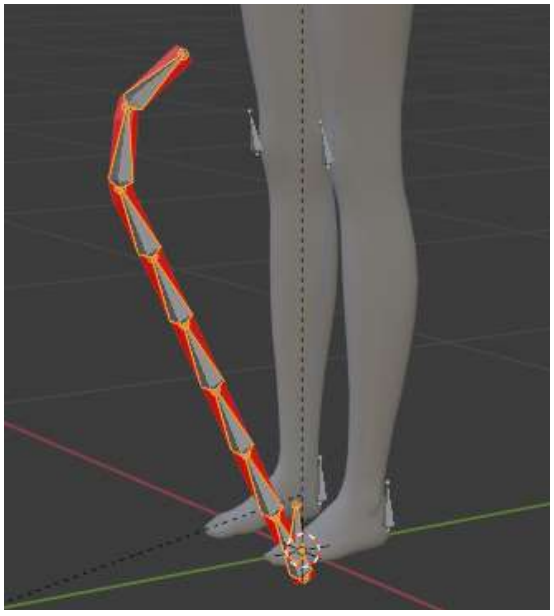
So now it's skinned, and I've posed some curve into the tail. Remember, these will be dynamic bones so they're going to move a bit, depending on how stiff you make them. I want to apply this pose as the rest pose to "bake" this curve into the tail, but if I do that the bones stay bent but the mesh straightens back out. So I have to apply the mesh's Armature modifier first to "bake" the curve into the geometry.



Then apply the armature's pose as rest pose, then put an Armature modifier back on the mesh, and then make the Meshes bone the parent of the mesh. I made my textures also.



So I'm done making the item in Blender, just have to disable the Copy Transforms constraint on the armature and export the FBX and textures. Of course, this puts the item in a weird position, but don't worry about it.



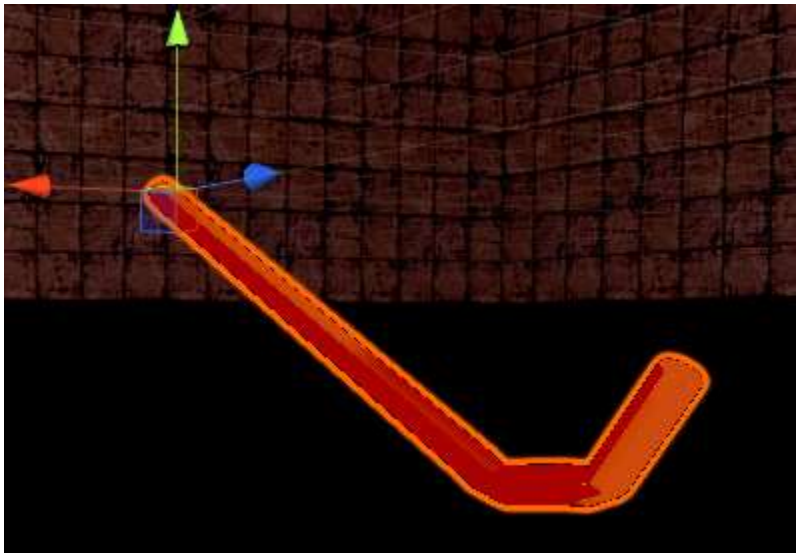
Skinned Accessory: Unity

So we're back to Unity. You should be able to figure out what you need to do from previous items, but here's a checklist:

- import FBX and textures, change import settings as needed (all 3D items)
- drag FBX thing into Hierarchy (all 3D items)
- zero out the rotation (Accessories)
- check item is on the right layer (all 3D items)
- create material with appropriate shader for item type, fill it out (all 3D items)
- drag material onto GameObjects with MeshRenderer (all 3D items)
- put appropriate MB on item and fill it out (all 3D items)
- put dynamic bone MB on item for each dynamic bone chain, fill it out (appropriate 3D items)
- make prefab (all 3D items)
- make prefab instance to check it looks okay (all 3D items)
- assign prefab to AB (all 3D items)
- assign thumbnail to AB if you already have one (all but Studio Items)
- fill out list file appropriately (all items)
- build ABs
- build zipmod
- test in-game

A couple notes about my CatTail item

- I don't want the base of the tail to move, so dynamic bone starts on the second bone
- I changed some dynamic bone settings so it should hopefully behave more like a tail
- it will be in the waist category, so it is in a separate list file from the Bracelet
- my preview scene isn't great for this, it is underground



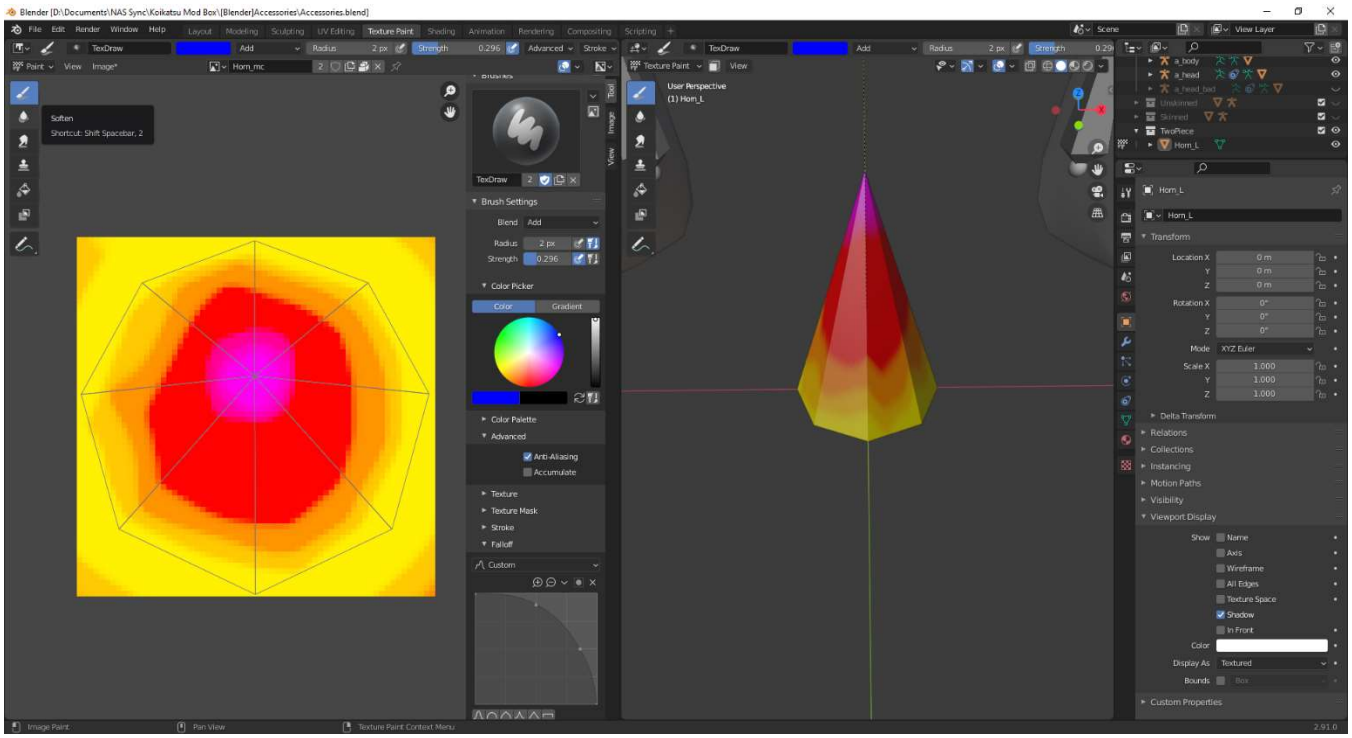
Skinned Accessory: Result

It is functional, but it looks pretty janky. Or should I say, kinky?. You might want more bones in the tail if you make one. You can also add geometry and make smoother bone weights so the bends don't look as sharp. If you want to test Dynamic Bones, Maker is not a great place because of limited character movement. Save a character with your item and go into Studio.

Two-Piece Accessory: Blender

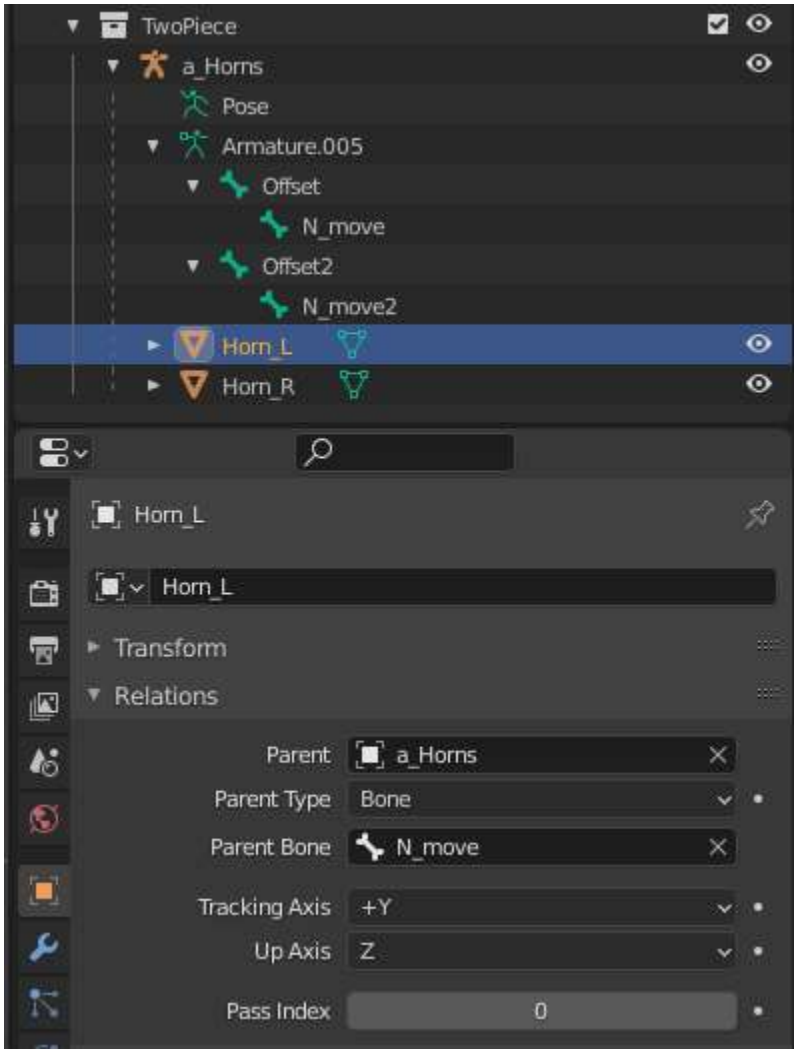
A two-piece accessory gives the player control over two parts of the accessory: N_move and **N_move2**. You can have N_move2 be a child of N_move, so that N_move moves the whole item and N_move2 moves part of it, or you can have them be parallel hierarchies so you're basically just putting two accessories in one slot. But they are both still attached to the same parent, so it's not quite the same. You normally see it for things like horns; they are both attached to the head, but in different locations, and you want people to move them separately so they can move them closer together or farther apart if they want, or angle them different directions. In these situations you may also consider making separate left and right accessories in case someone only wants one side.

This time I will just be making simple horns. You can probably do this as a single skinned mesh where each horn is skinned to a different side (under N_move and N_move2), but I'm just going to do them as two separate unskinned meshes. So I started with making a mesh for the left horn, and UV unwrapped it and made a color mask for different color at the base and tip. This is one of those situations where making the color mask in GIMP (radial gradient) before UV unwrap might look better, but oh well.

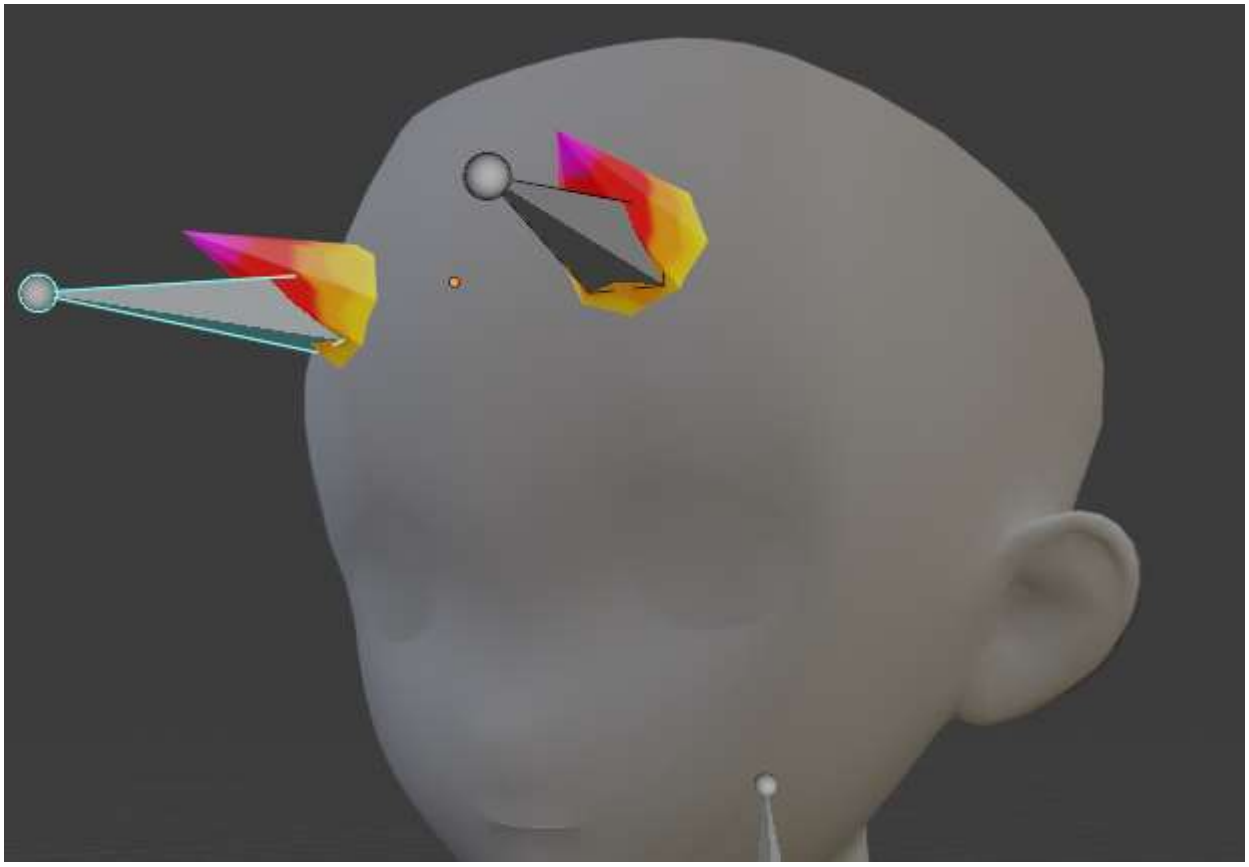


So to make the other horn I duplicated this horn and mirrored it on the X axis, which gives it a scale of -1 on X. I apply the scale to make sure it doesn't give problems later. At this point I have two horns right on top of each other, which are mirrored. The mirroring keeps the UVs in the same place so the texture basically ends up looking mirrored. If you don't want this perfect symmetry, you would UV them separately. The mirroring also flipped my normals inside out, so I had to flip them back for that second horn.

Next I create the armature and set the parents of the meshes to N_move and N_move2.



And then put the Copy Transforms on the armature, set to a_n_head. Pose each Offset bone like I did with the bracelet, and apply pose as rest pose.



Then you're ready to disable the armature's Copy Transforms and export, same as other accessories.

Two-Piece Accessory: Unity

There is nothing special to do in Unity with a two-piece accessory. Just do the same steps as other accessories.

Two-Piece Accessory: Result

Make sure to test being able to move both parts. There should be no surprises unless you messed up the hierarchy.