MOUSE TUBER 1.4.1+

Guide

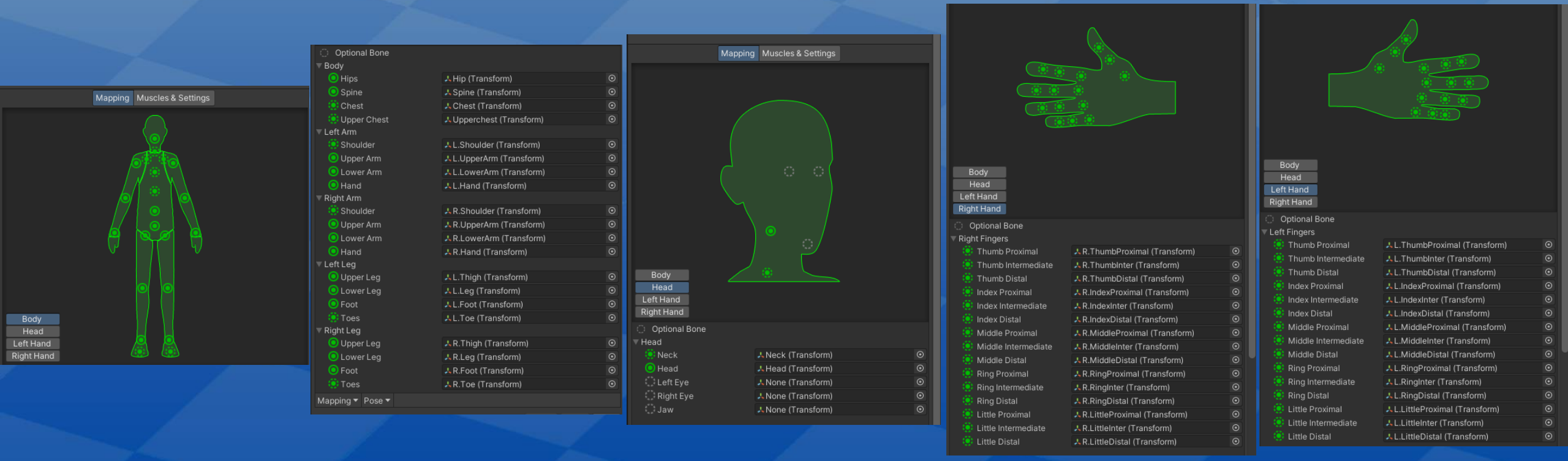
Made in Power Point

By: the Creator of Mouse Tuber

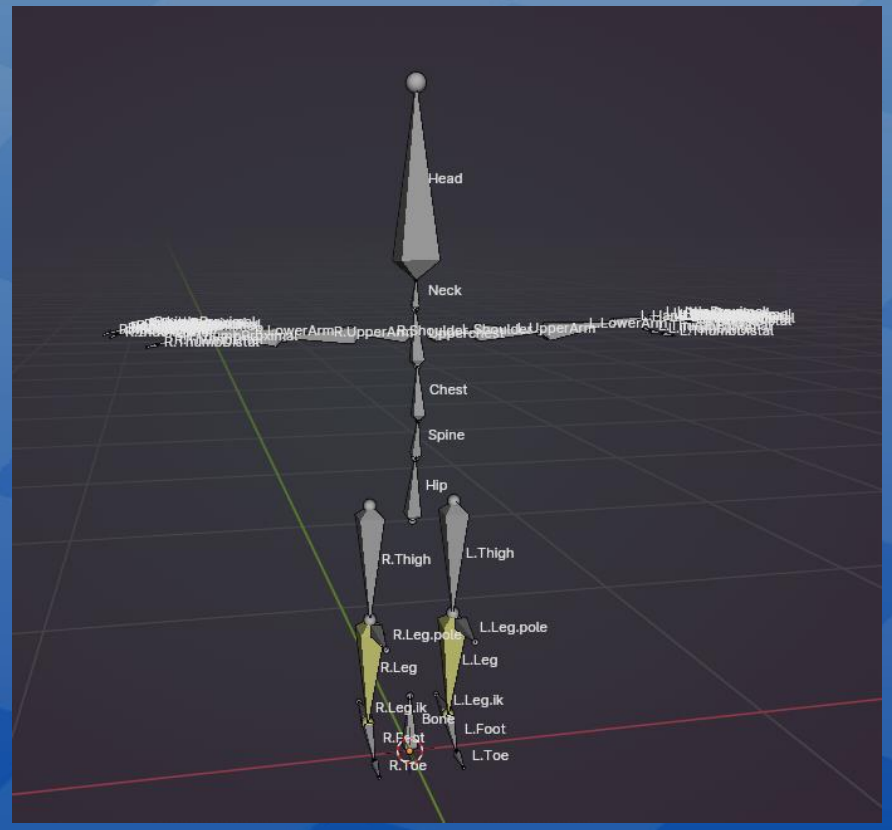
Mouse Tuber by Klausbdl (itch.io)

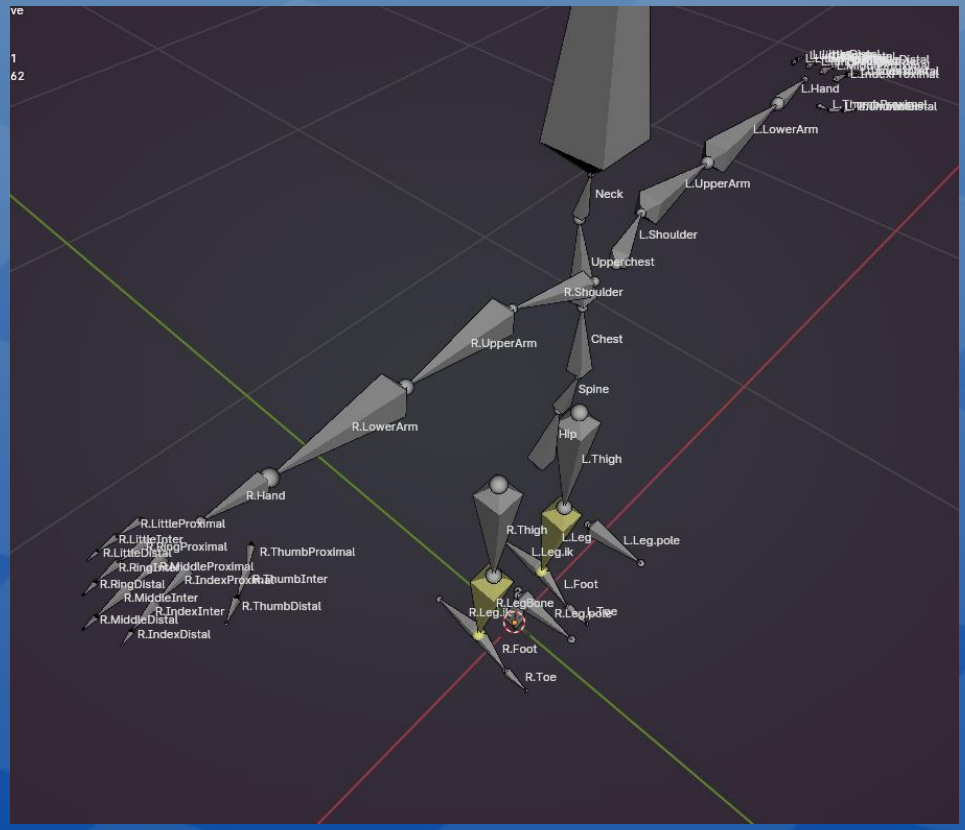
1. 3D MODEL

- Your 3D model needs to be humanoid: at least use a humanoid rig supported by Unity Engine.
- This is the humanoid rig definition:

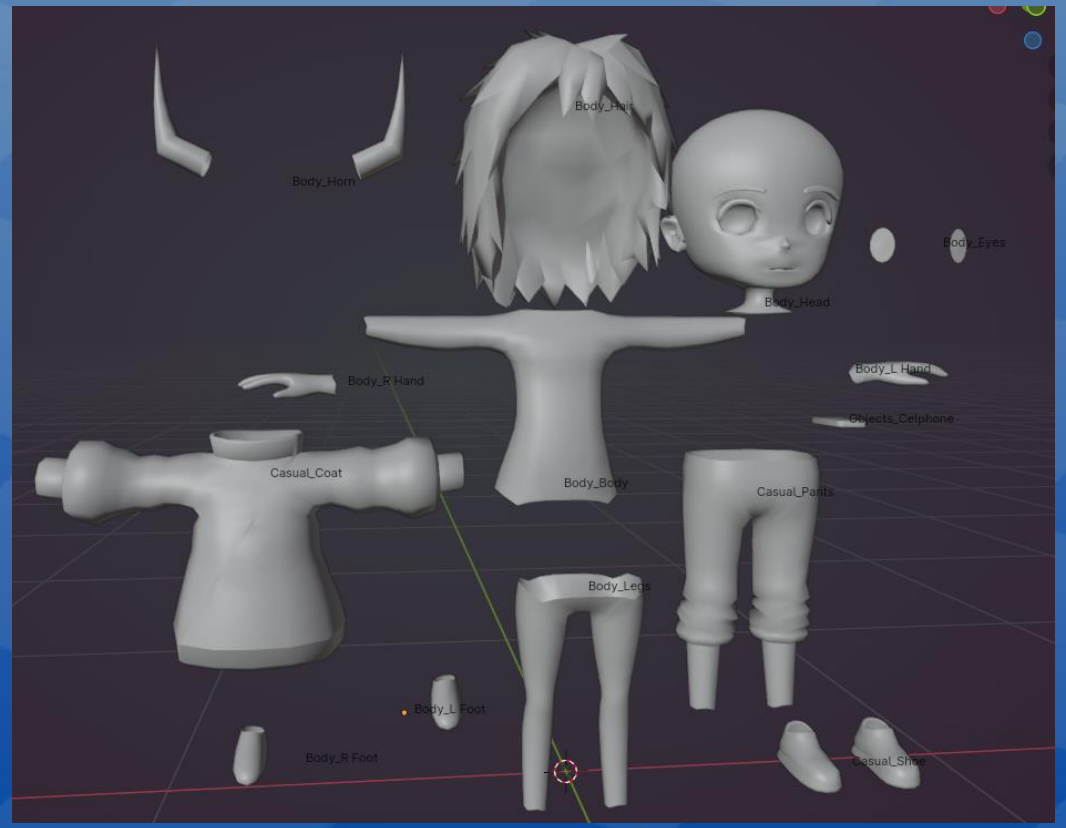


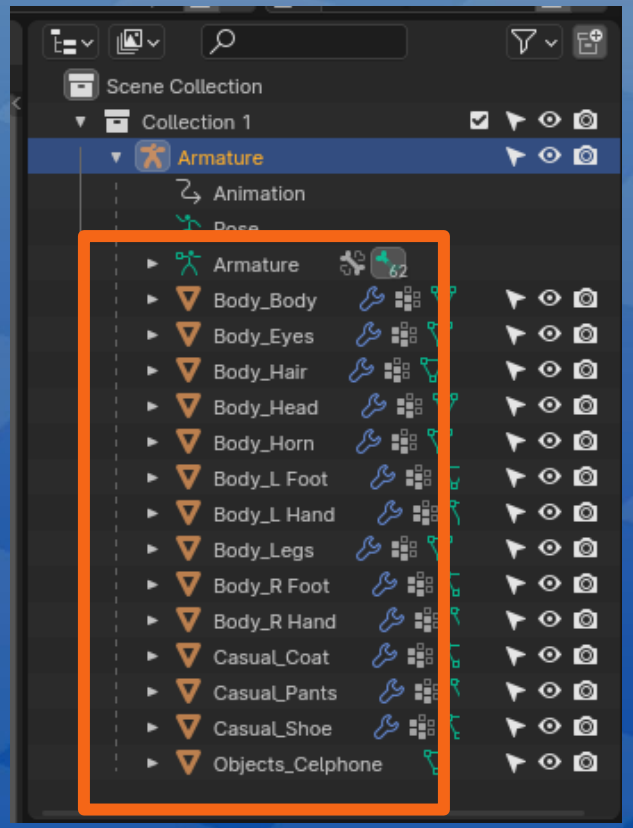
- In this Unity project, you can get a rig from the Klausbdl blender file to use it on your own avatar. It's in the Assets/Models folder.
- If you have your model ready, use the Append feature to import only the Armature to your file and resize the rig to fit your avatar.



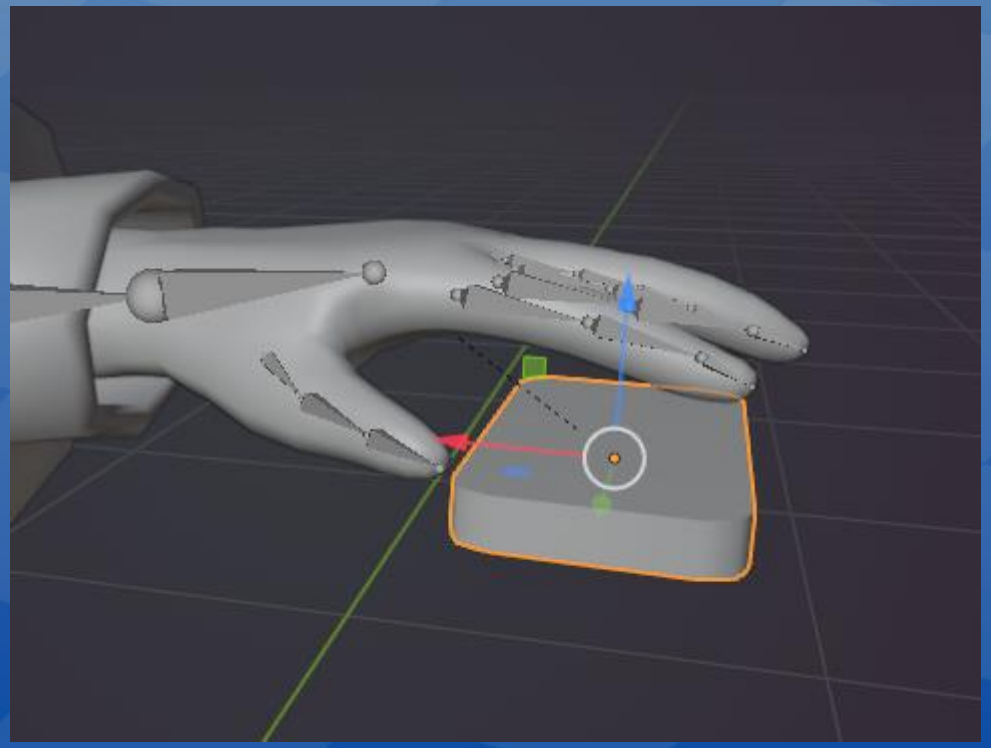


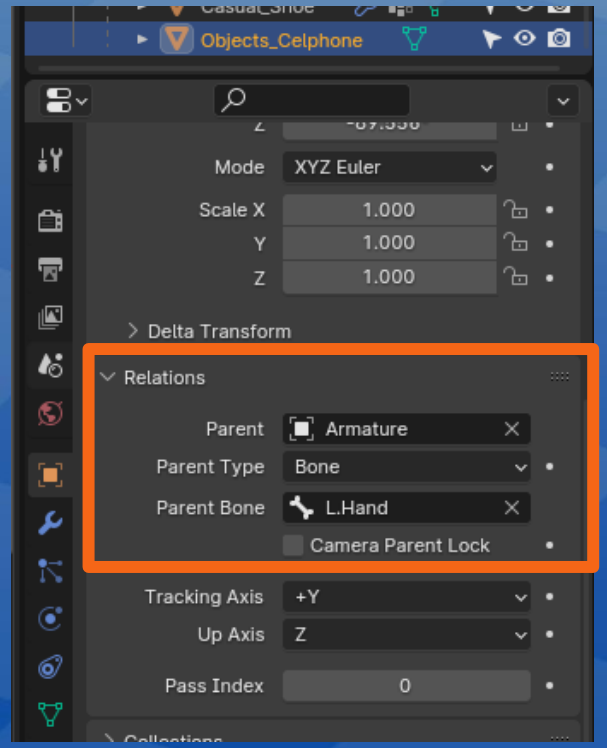
- For better usage of the Mouse Tuber app, you must separate your model into parts. Not just the body parts though, but clothing as well.
- You should follow the naming convention so it appears nicely in the Outfits panel:
- Category_Part Name





- If you have togglable Objects, such as glasses, hats, a sword… anything that does not require weight painting to the rig, you should name it “Objects_name”.
- However, so it follows the bones, you need to parent it to a bone in the Relations section. You can always make adjustment later in Unity. More on that later.
- Example with a cellphone object parented to the left hand bone:





End of 3D MODEL

2. SHAPE KEYS

- So your avatar can move their eyes and mouth, you need to setup shape keys. There are shape keys for the eyes looking in the four directions and blinking, and shape keys for the mouth open and closed.
- These are the shape keys, where “name” can be any name:

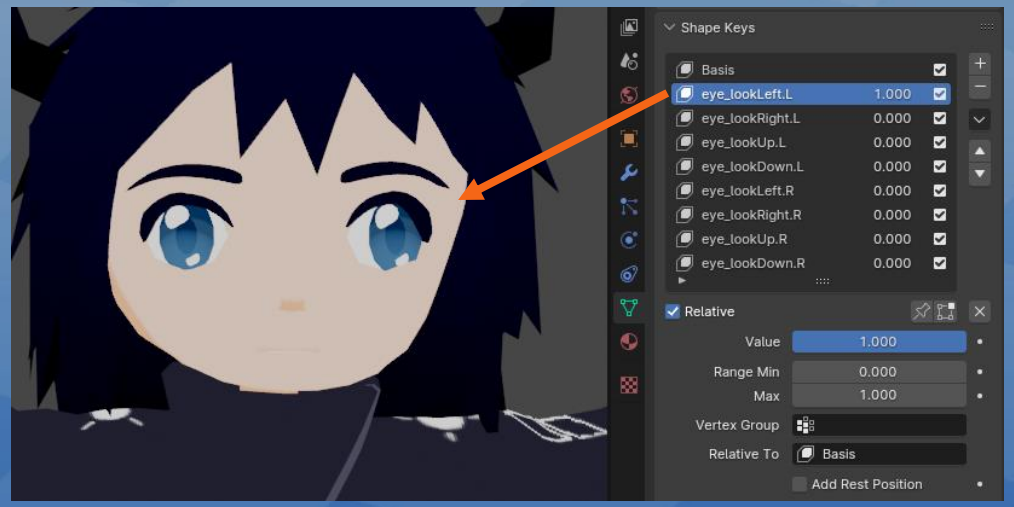
name_lookLeft.R
name_lookRight.R
name_lookUp.R
name_lookDown.R

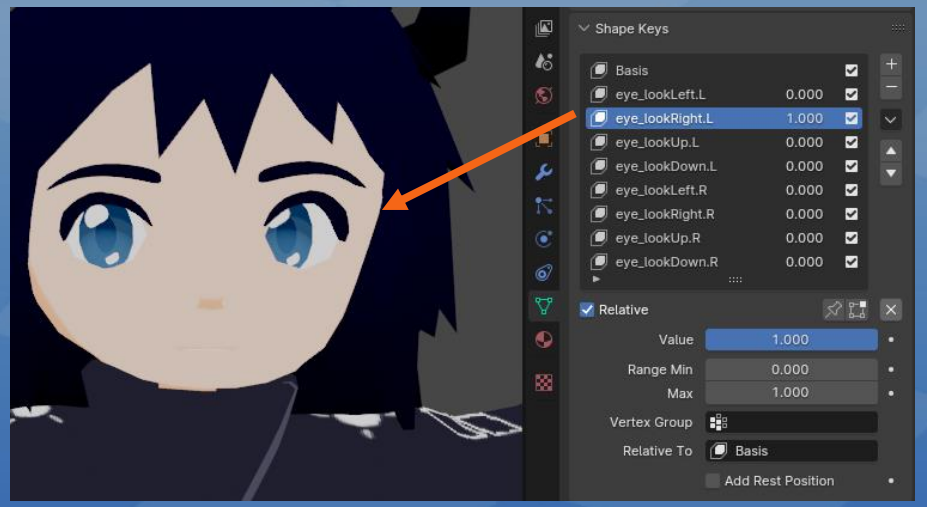
name_lookLeft.L
name_lookRight.L
name_lookUp.L
name_lookDown.L

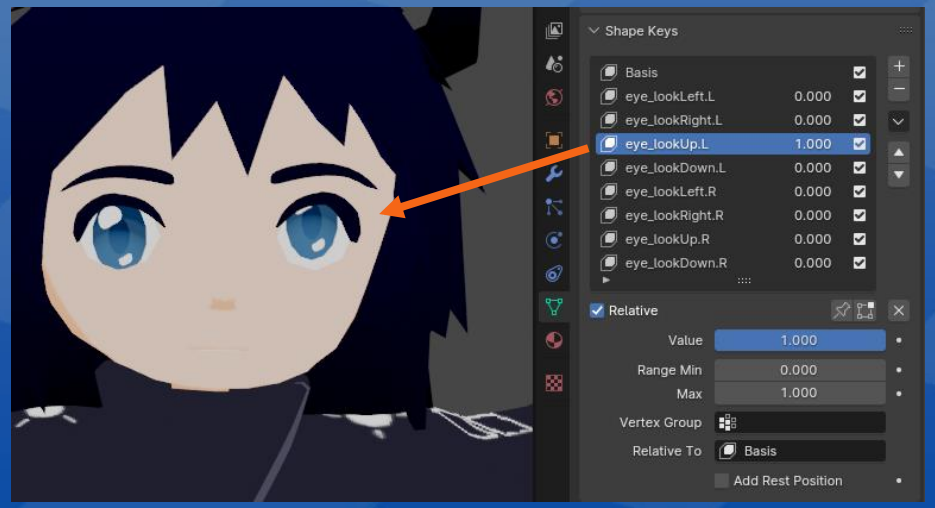
name_blink.L
name_blink.R

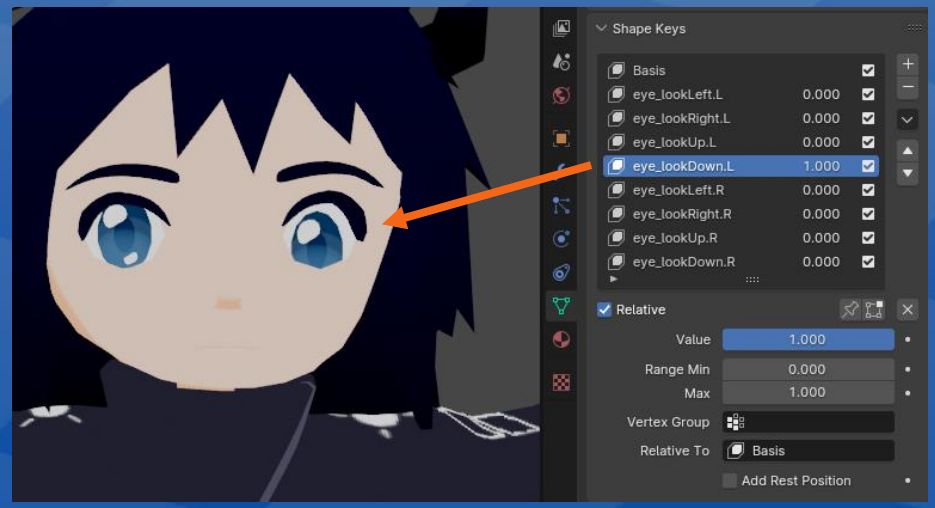
name_mouthClosed
name_mouthOpened

- name_lookLeft.L | name_lookRight.L | name_lookUp.L | name_lookDown.L
- Do the same for the right eye, but replace the L with R in the shape key name

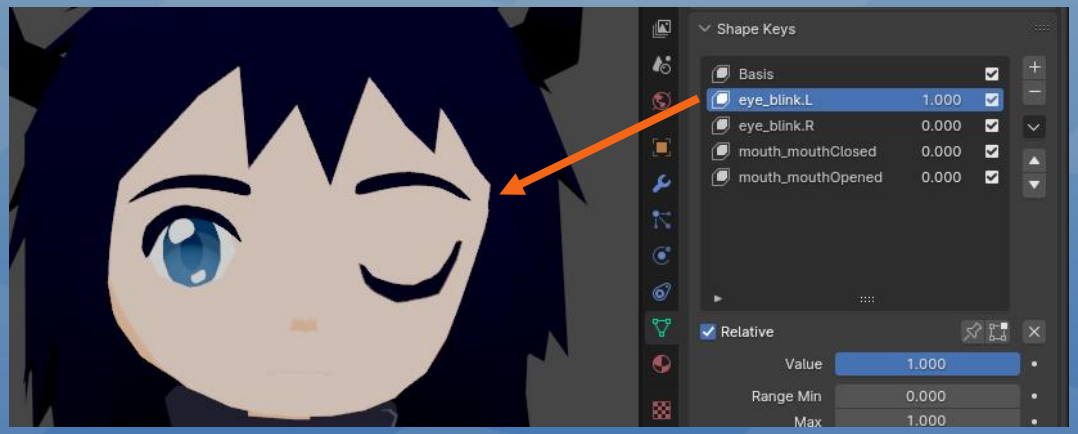


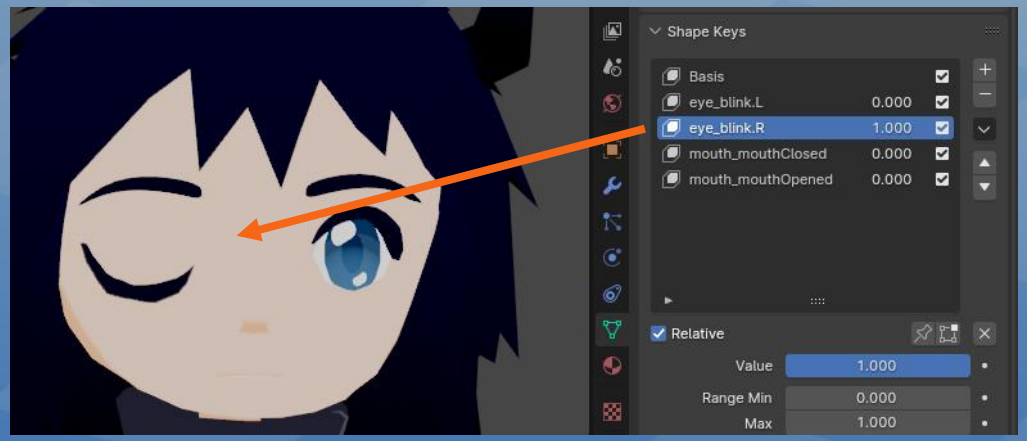




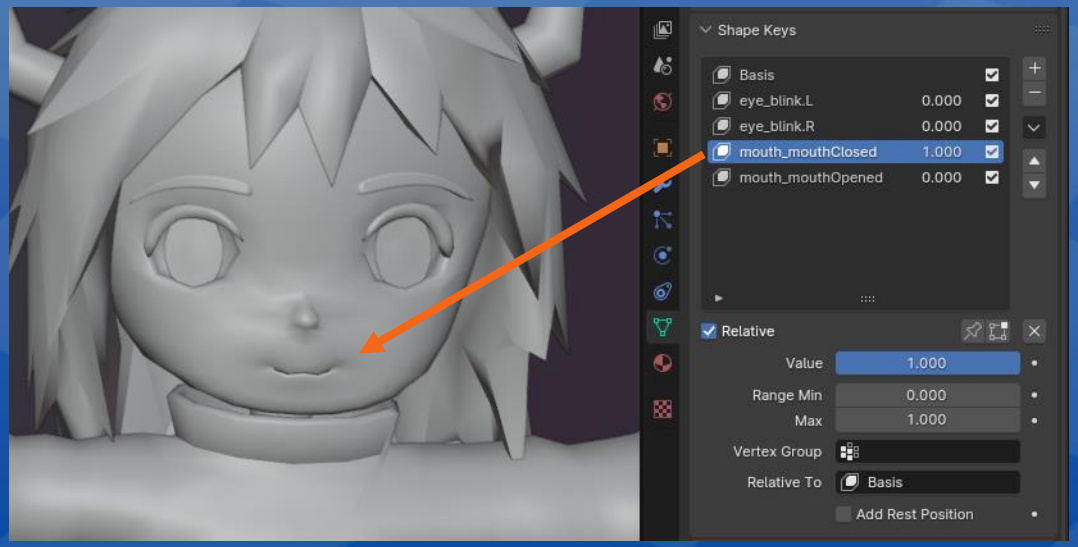


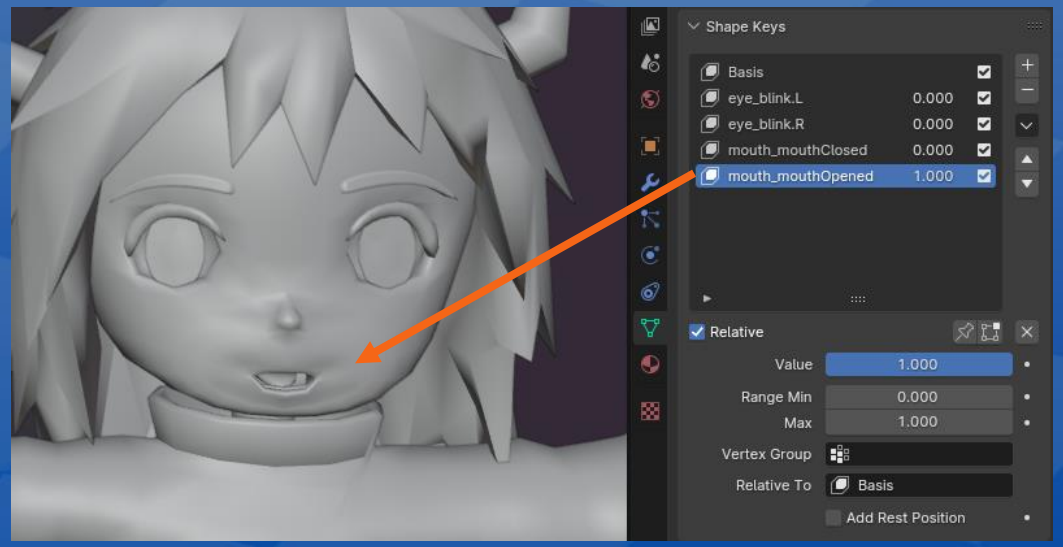
- name_blink.L | name_blink.R



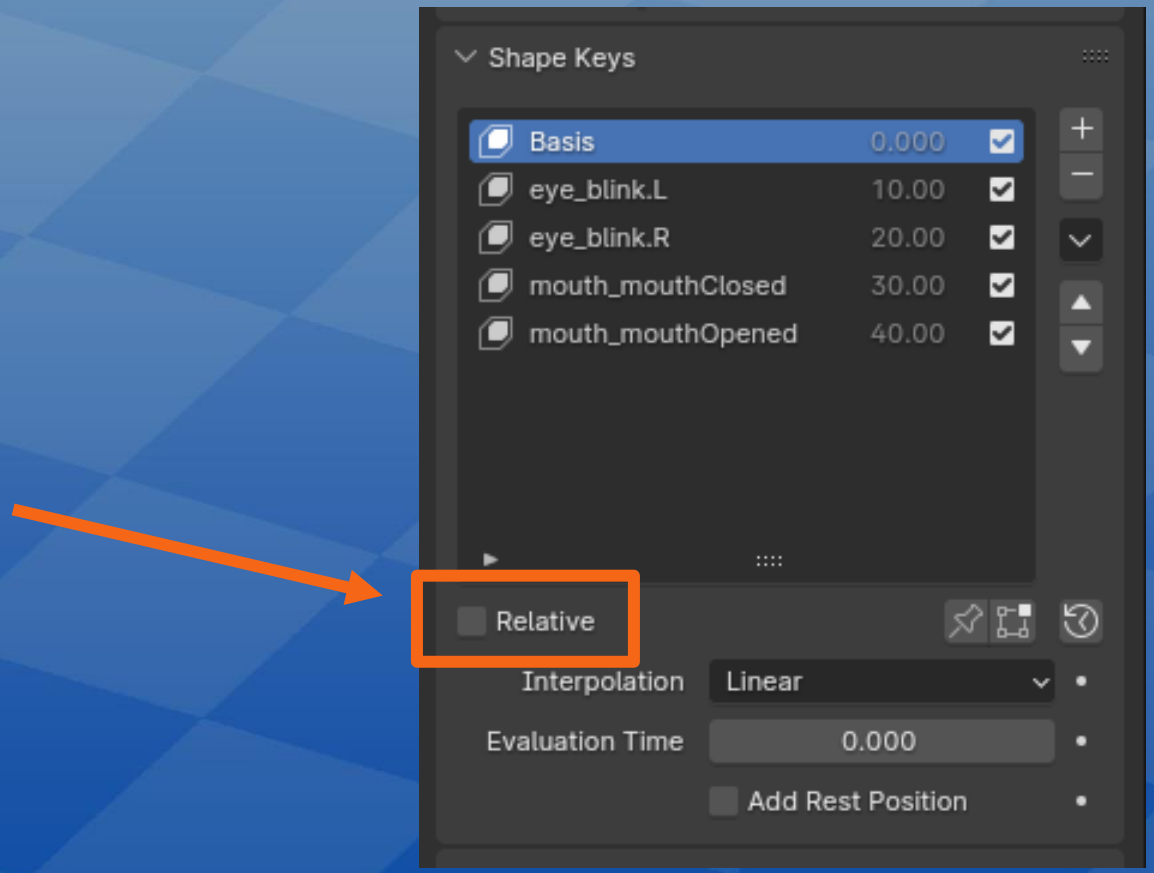


- name_mouthClosed | name_mouthOpened





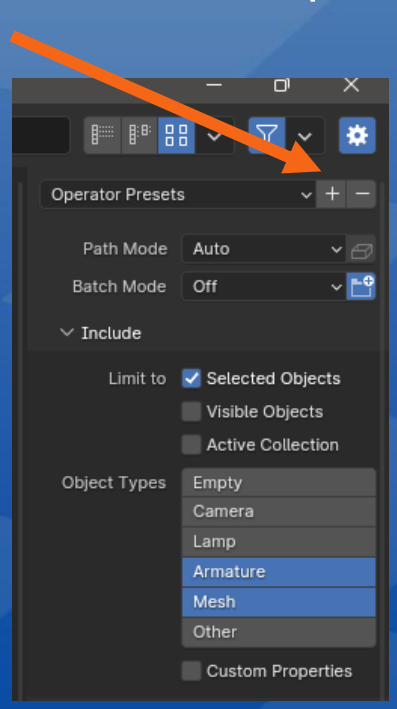
- **IMPORTANT!**
- After editing your shape keys, make sure to **UNCHECK** "Relative" for every mesh with shape keys. This is necessary for Unity to recognize them.

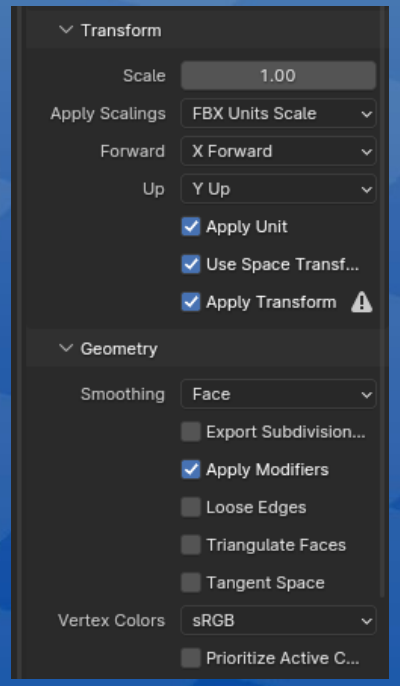


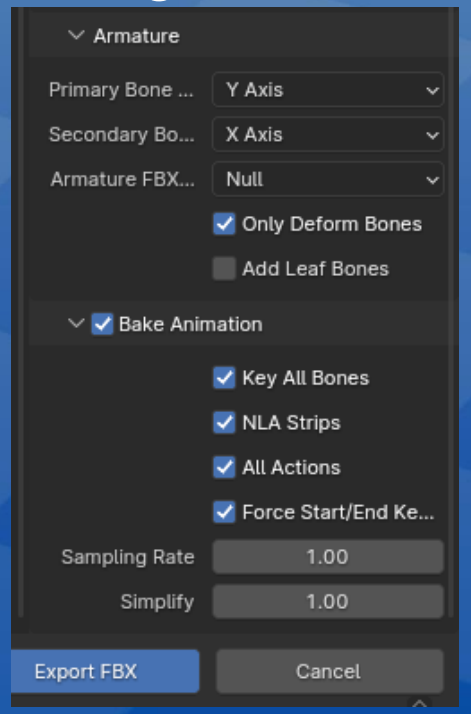
End of SHAPE KEYS

3. EXPORTING

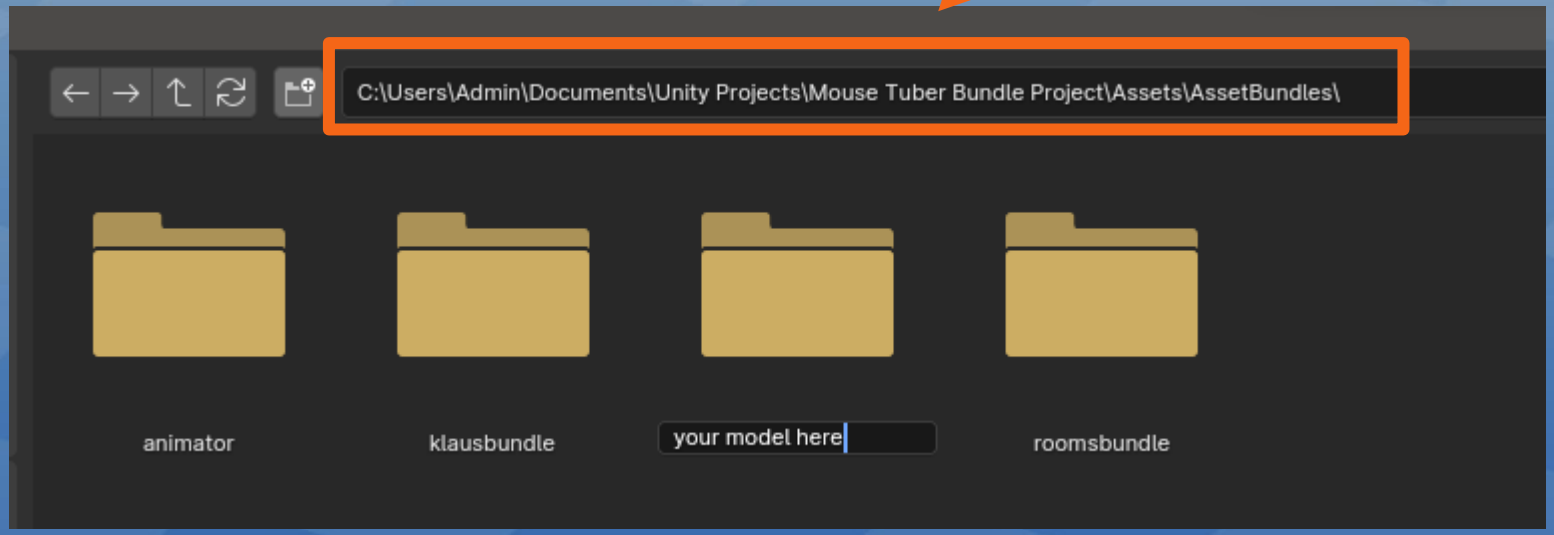
- When you're done, you are ready to export it as FBX.
- Select the rig and all the meshes, go to File > Export > FBX.
- Use the following settings:
- PS: you can save it as a preset for faster exports in the future by clicking on the + button and giving it a name.







- Export the model to a folder inside the AssetsBundle folder inside the Unity project's folder

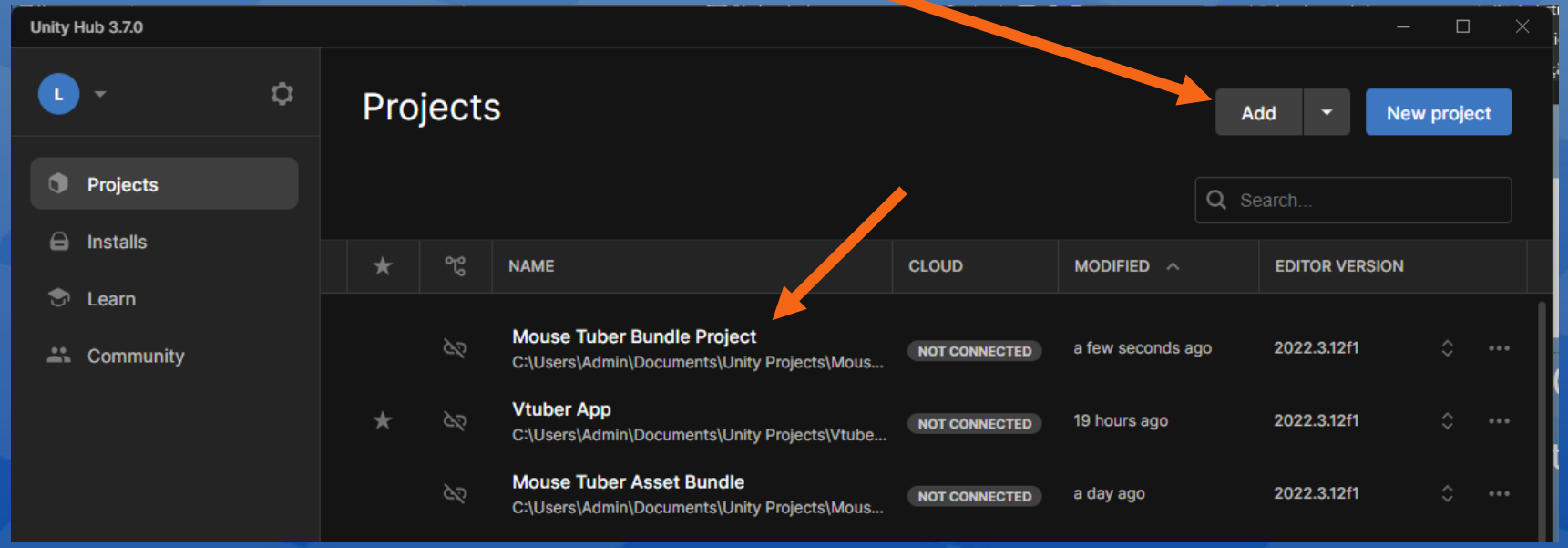


- Name it anything BUT ending in "_model". This naming is reserved for actual models to be loaded by the app.
- I recommend naming it "avatar name_obj".

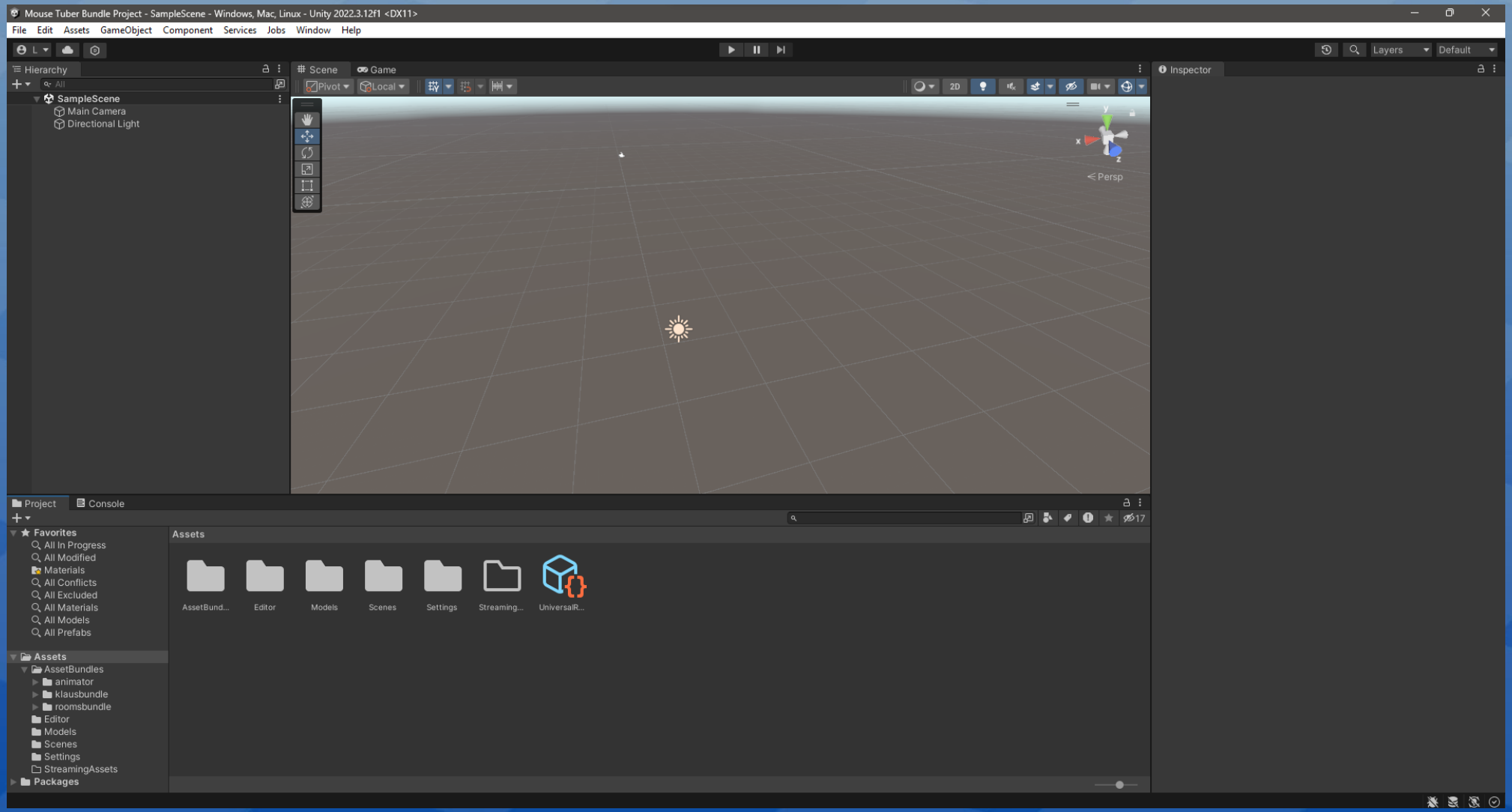
End of EXPORTING

4. IMPORTING TO UNITY

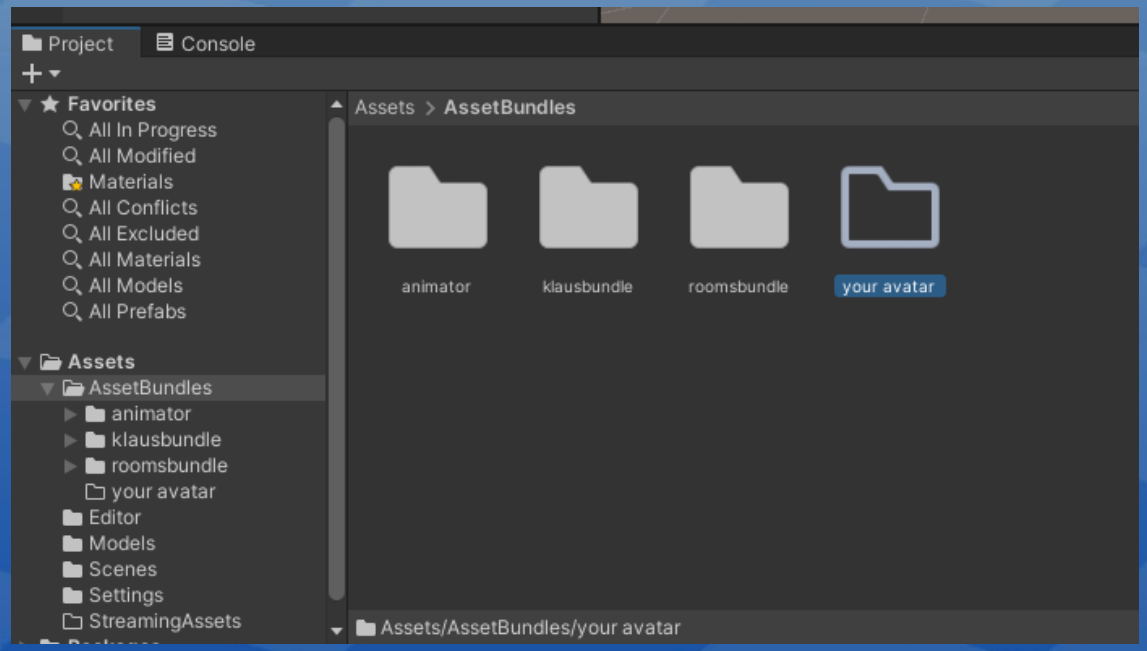
- Download and install the Unity Hub, and then install Unity version 2022.3.12f1.
- Click on the Add button on top and select the Bundle Project folder to open the project.

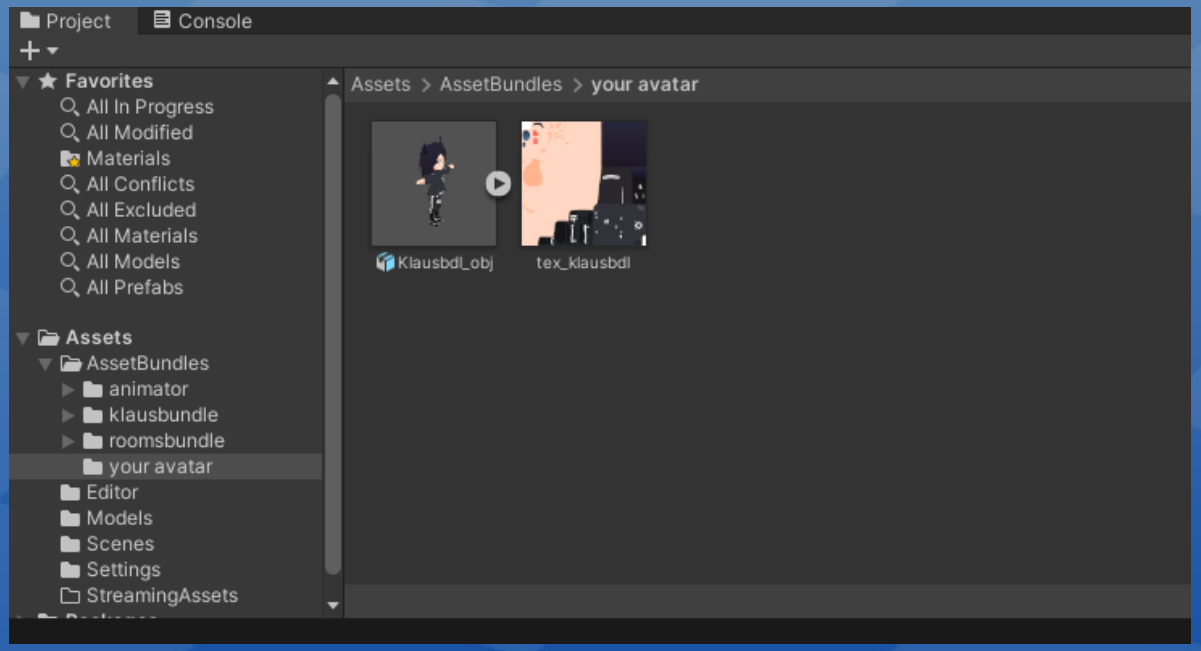


- You should see something like this:

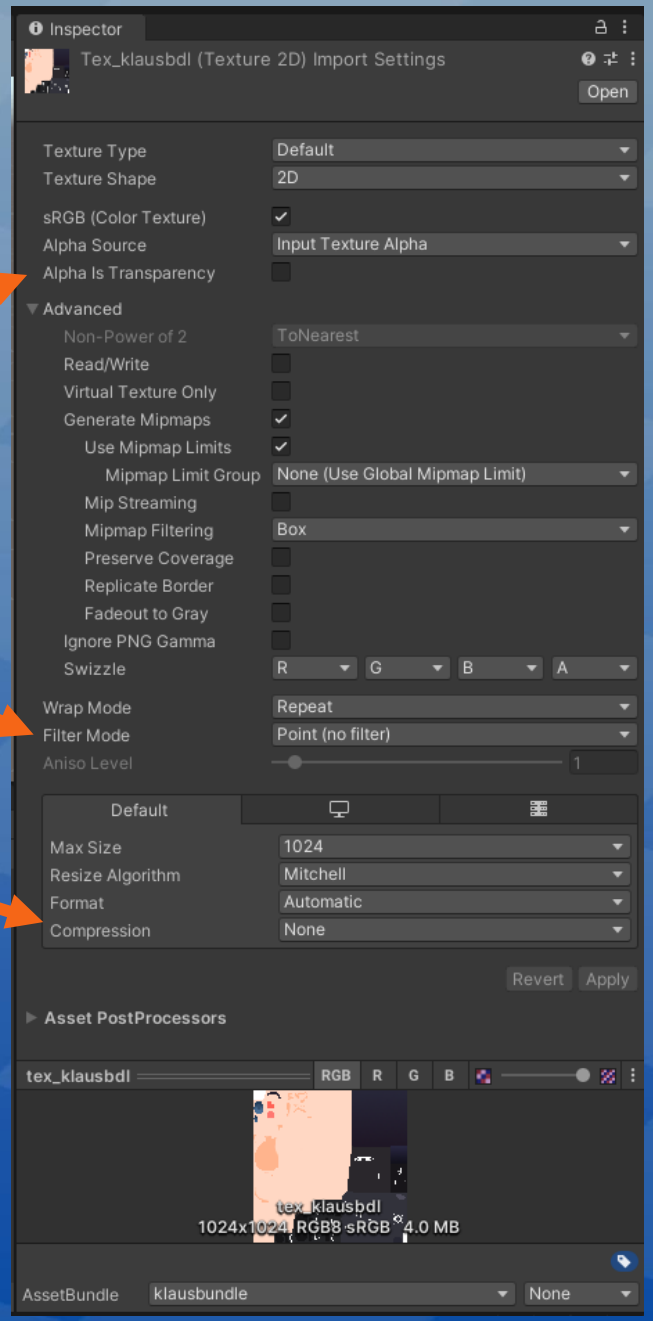


- Navigate to the AssetBundles folder, where you exported your FBX.
- Go ahead and drag in any textures you may have.
- I recommend renaming it to “tex_name” where name is anything you want. See the example below on the right.

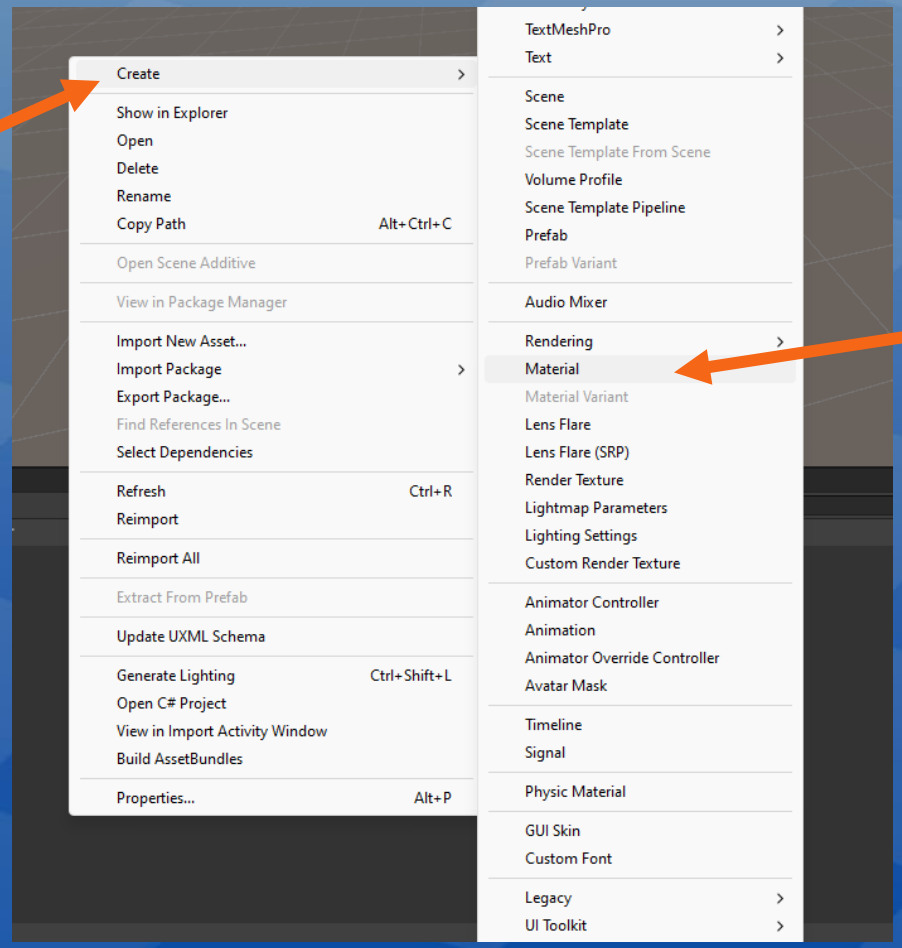




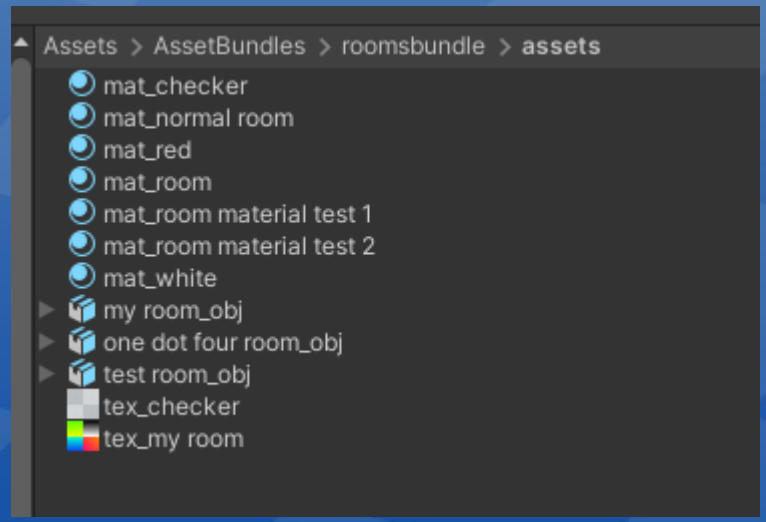
- First of all, lets prepare the texture. Click on it once to load the import settings on the Inspector window on the right.
- If your texture has transparency, check THIS option.
- If your texture is pixel art, change THIS to “Point (no filter)”.
- Finally, change the Compression to none so its not low quality.



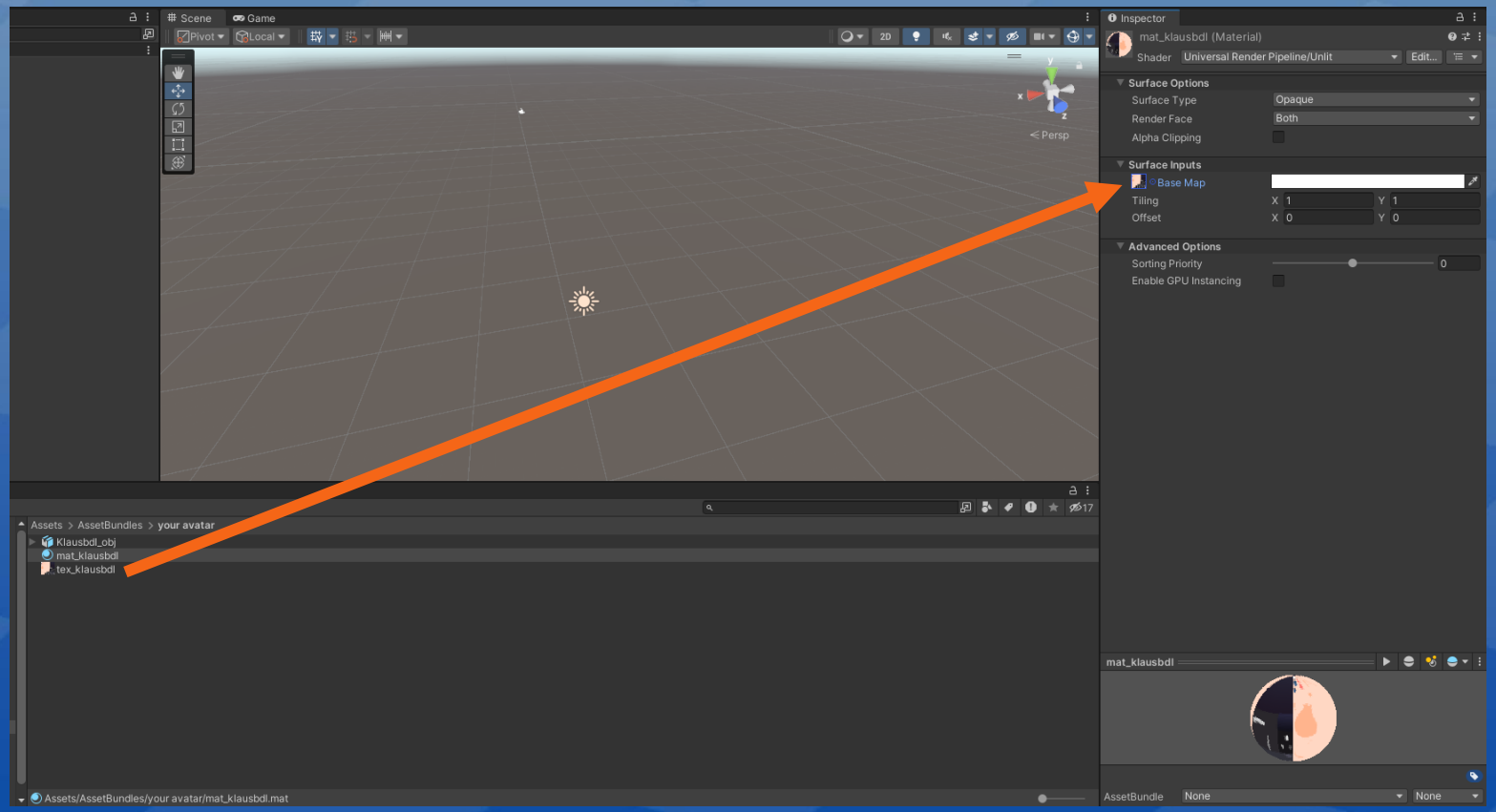
- Now, in the Project window (where you dragged in the textures before), right click > go to Create > Material.
- Give it consistent names, such as “mat_name”.

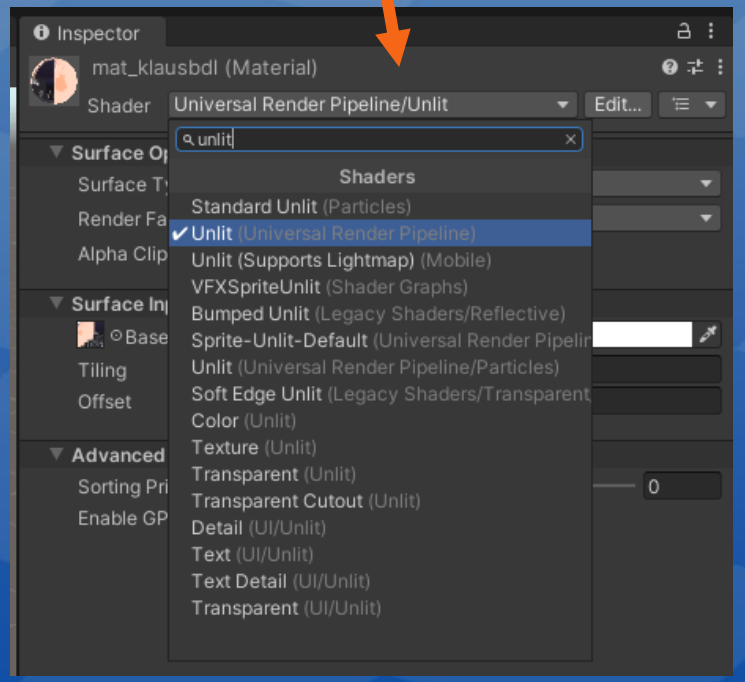


- Example from the rooms bundle inside the project:



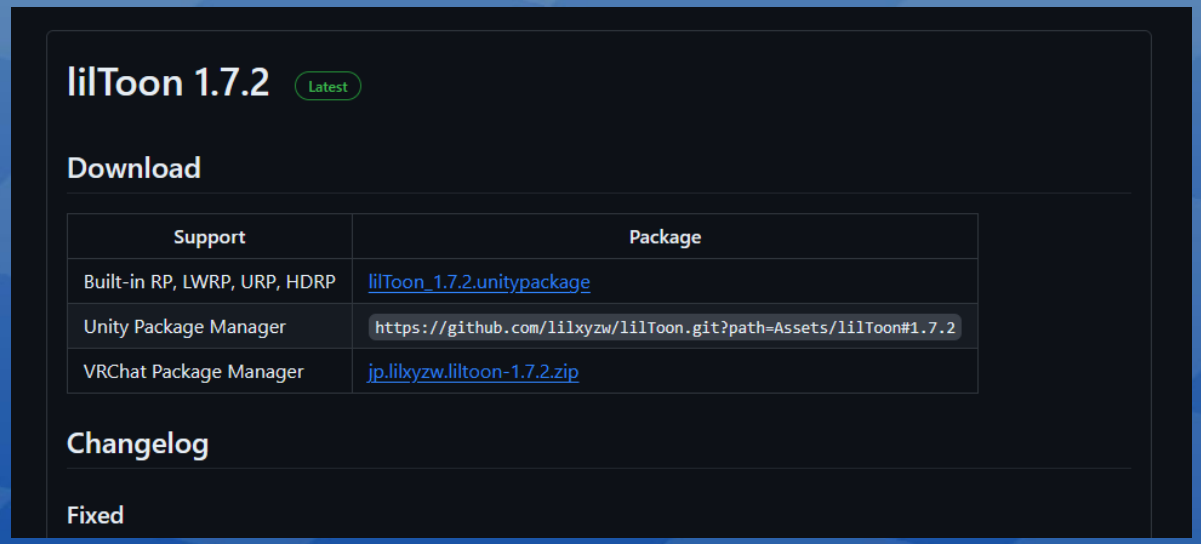
- Set the Base Map of the material to your texture by dragging it to the little square.
- So you model can have an anime shading, you can set the material's shader to Unlit.

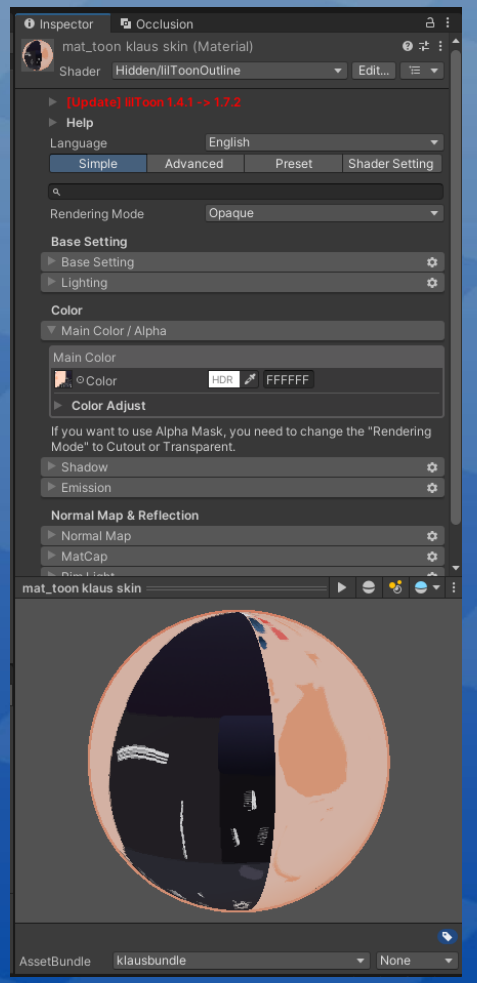




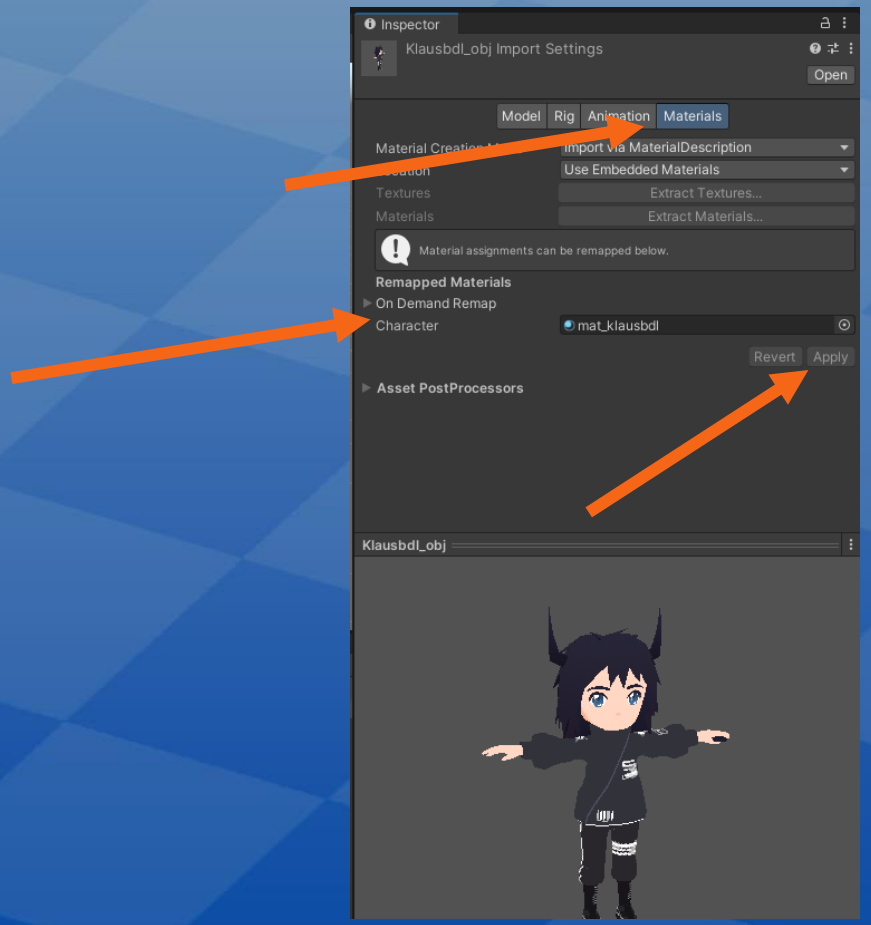
- You can also download the lilToon shader and import it to the Bundle Project. With its shader, you can have cool shadows, outlines and more effects. You can also just create your own shader, if you know how to do it!

Releases · lilxyzw/lilToon (github.com)

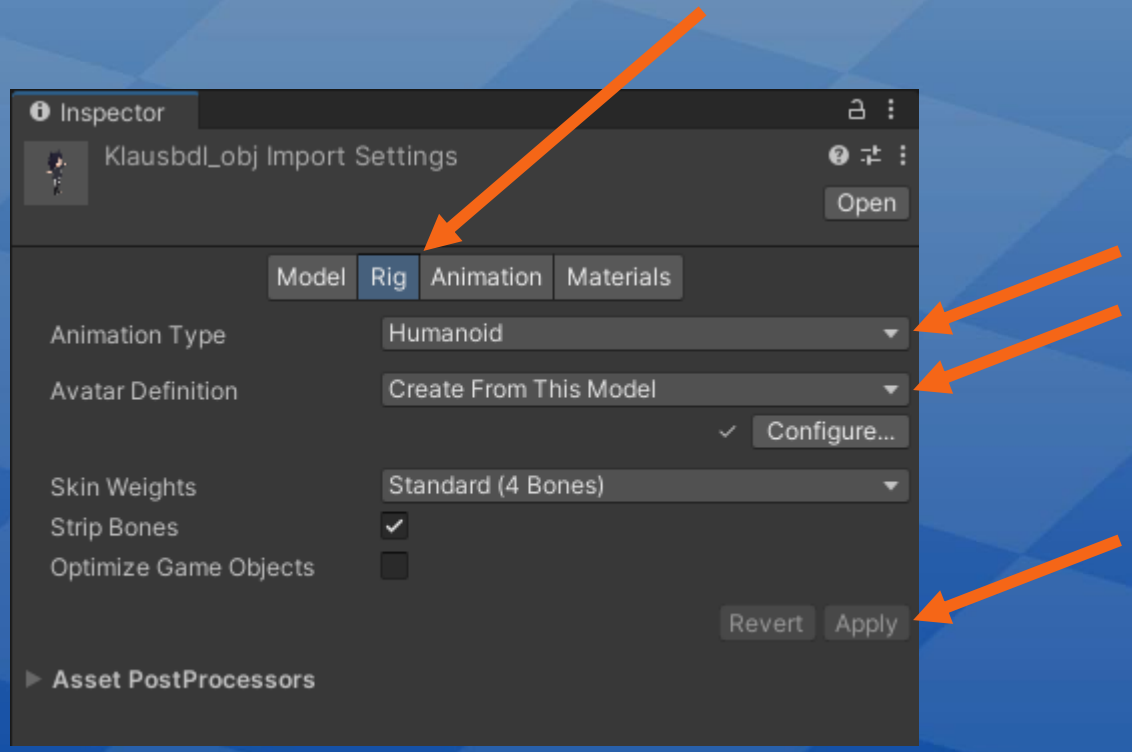




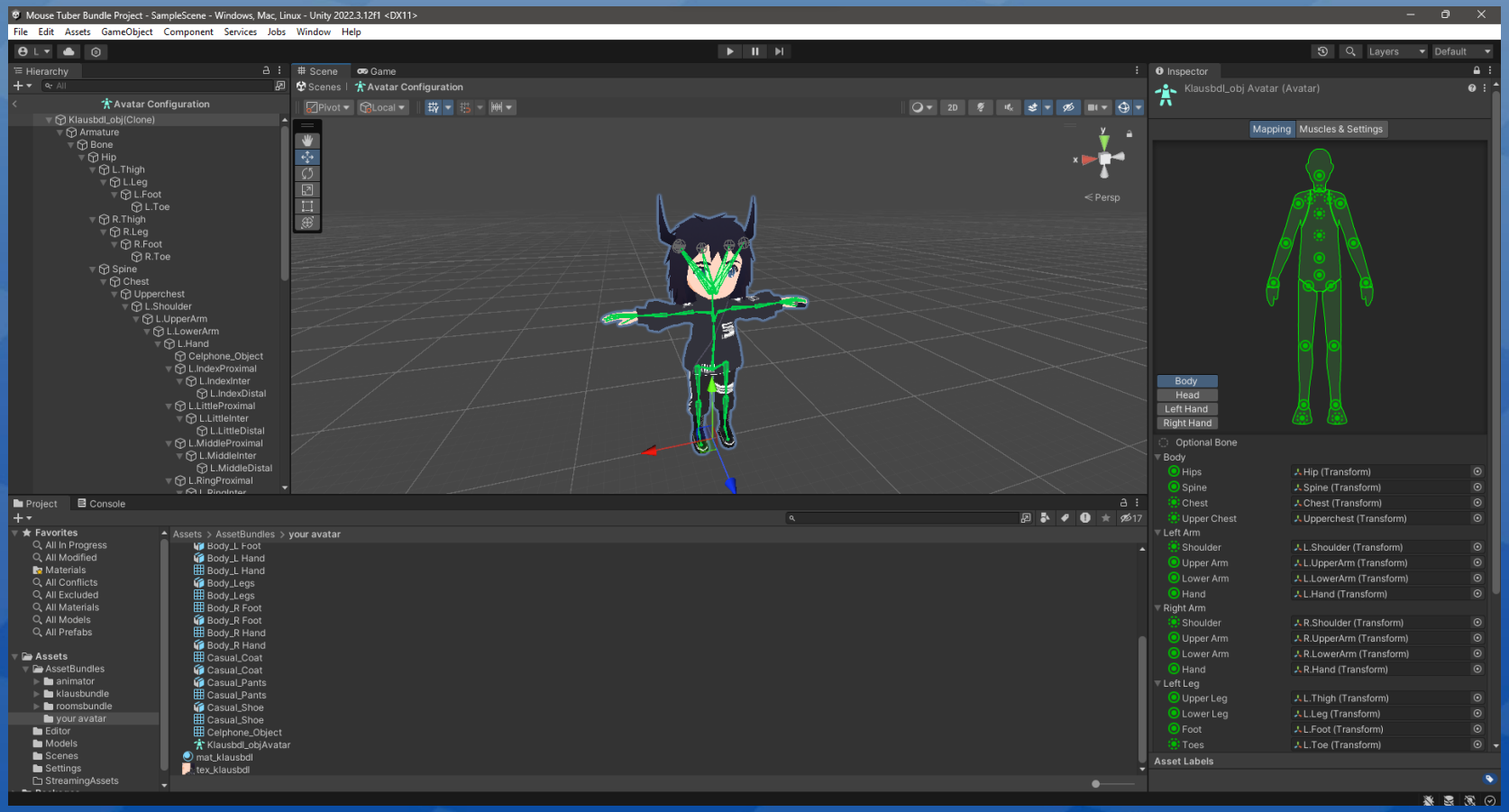
- Next, lets change the import settings of the avatar model. Click on it once to load in the Inspector window.
- First, go to the Materials tab and assign the correct materials to the slots. Then, click on Apply.



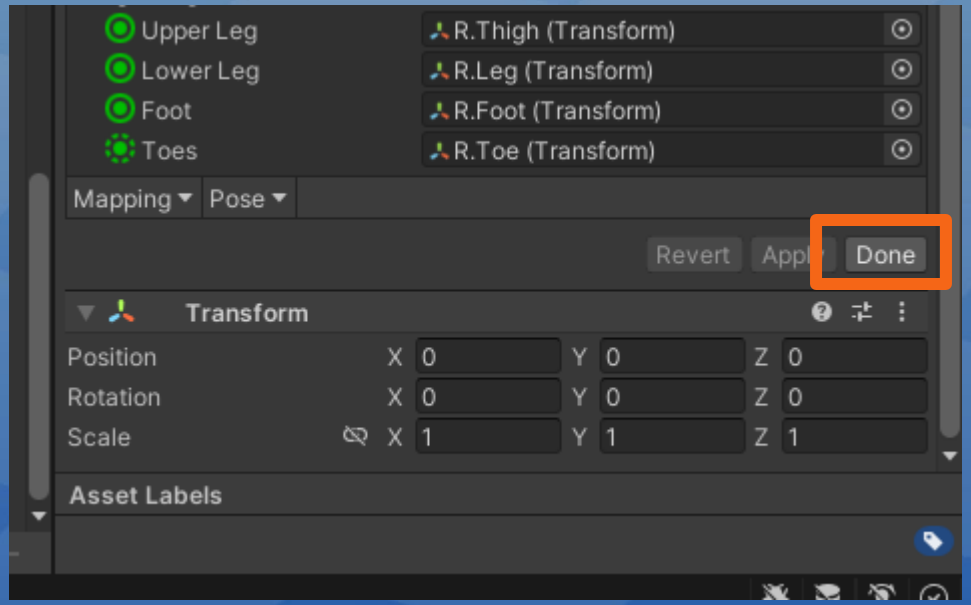
- Next, go to the Rig tab.
- Change “Animation Type” to Humanoid and “Avatar Definition” to Create From This Model.
- Click Apply.



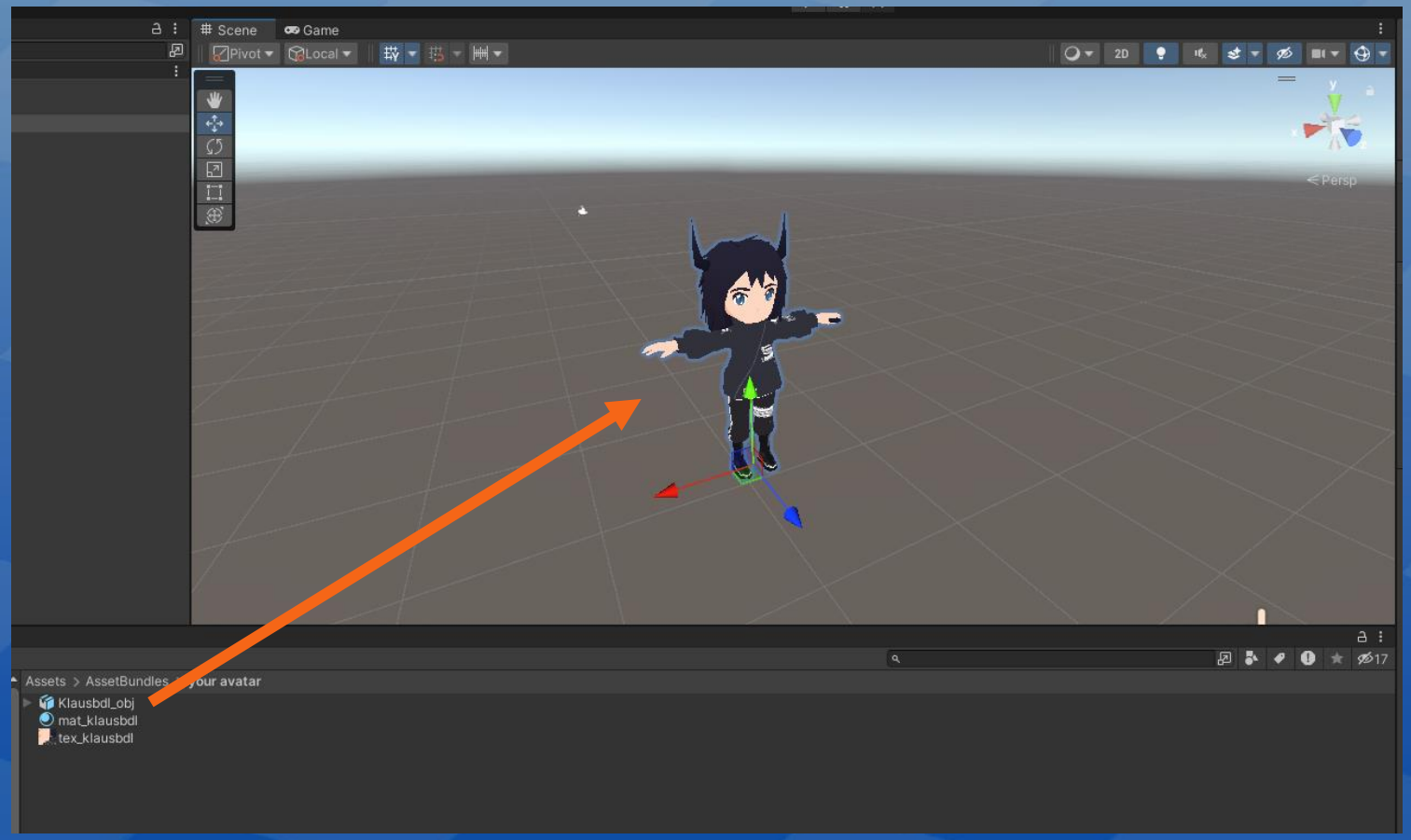
- Now, click on the Configure button. A new viewport will load so you can check the bones. Everything should be green, like on the screenshot below.
- If it's not green, you can try to fix the bones in the Inspector window. By clicking on the circle button next to the bones, you can pick a new one from the list.



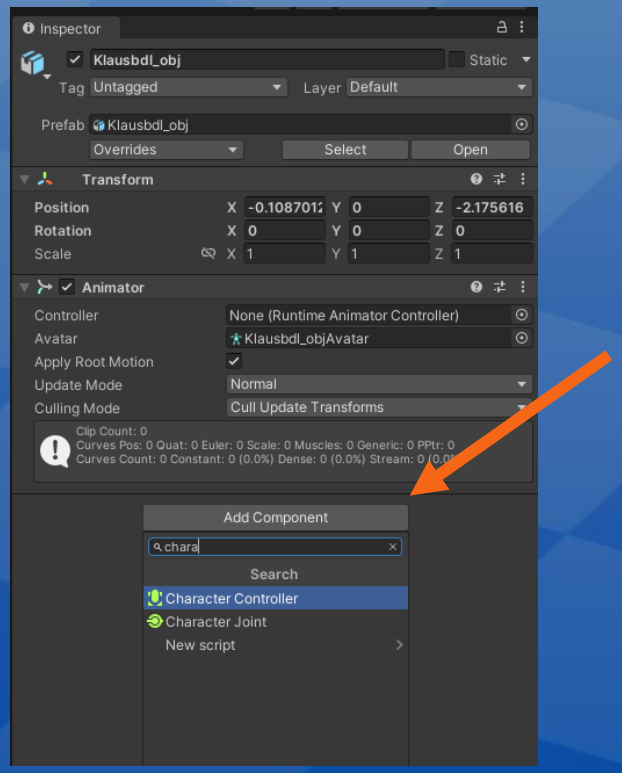
- If everything is OK, scroll down and click Done.

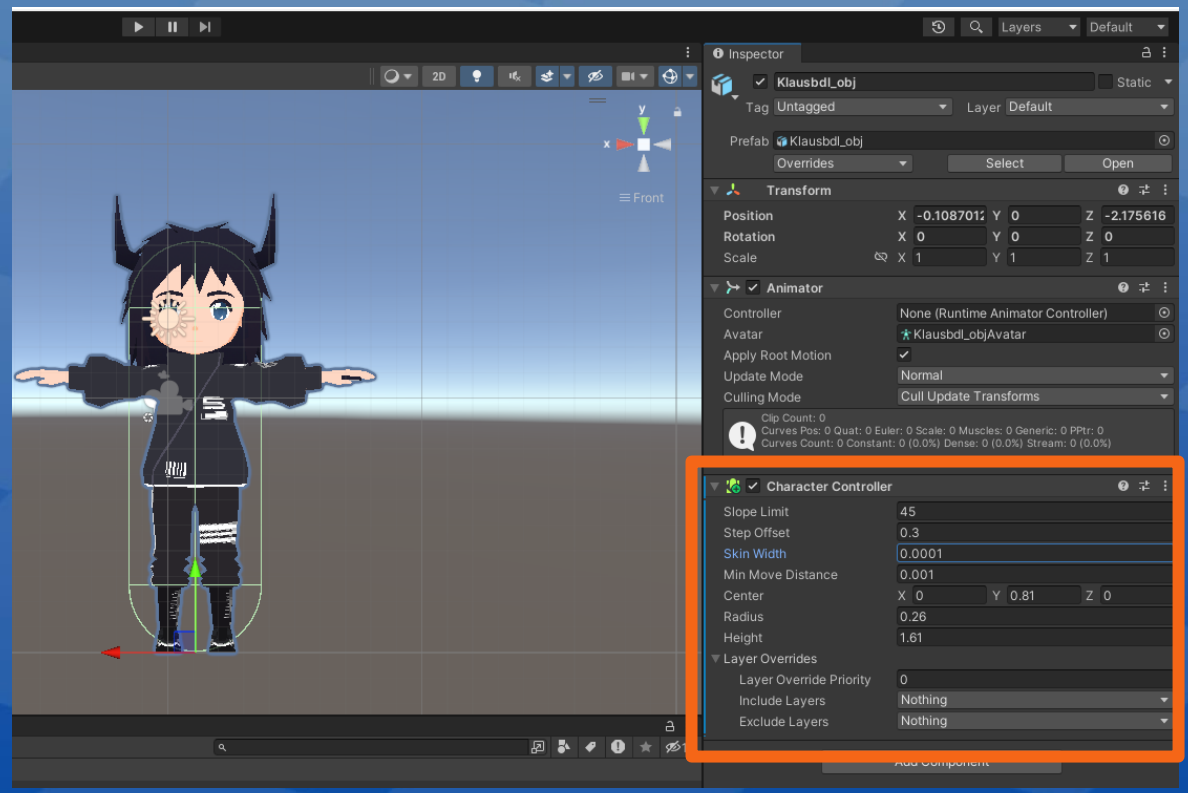


- Almost there (if you don’t have hair physics)!
- Now, drag the avatar_obj to the scene. We will prepare it for the app.

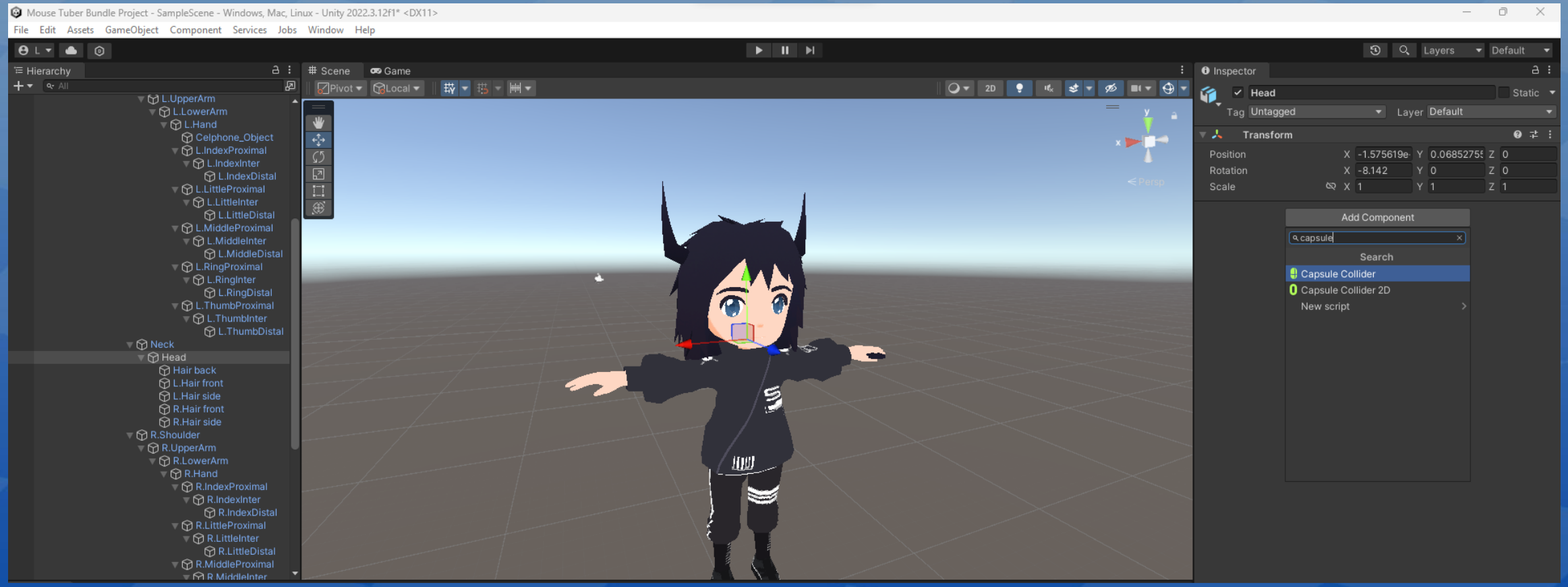


- With it selected, in the Inspector window, add a new component of the type Character Controller.
- Then, adjust the Center, Radius and Heigh of the capsule to fit your avatar. It may take some minutes. Doesn't need to be perfect, but make sure to align the feet!
- Change the Skin Width to the lowest value possible.

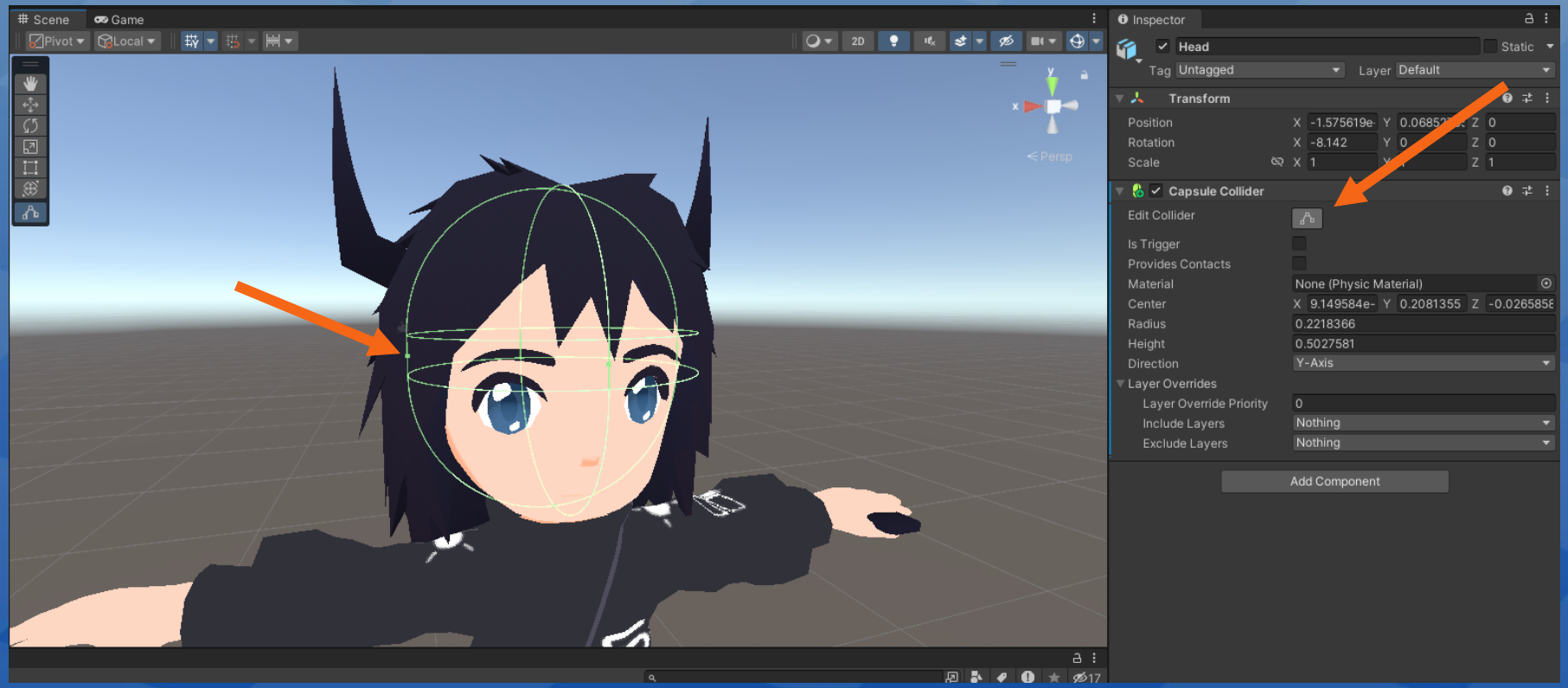




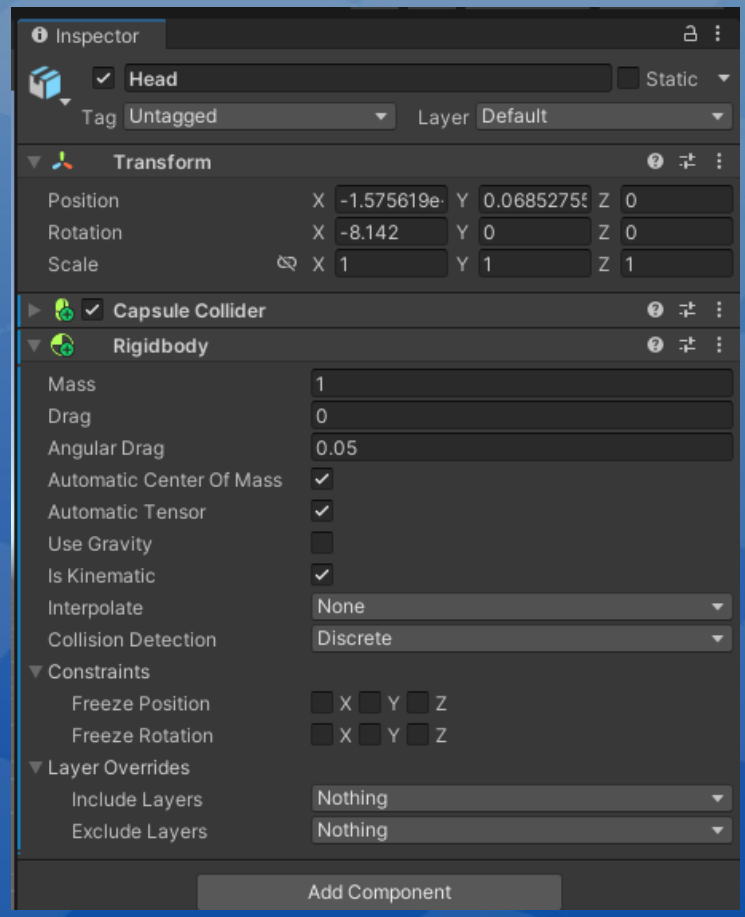
- Next, let's add collision to the head bone. Expand the model in the Hierarchy window. You can hold Alt and click on the arrow to expand everything.
- Select the head bone and add a Capsule Collider component.



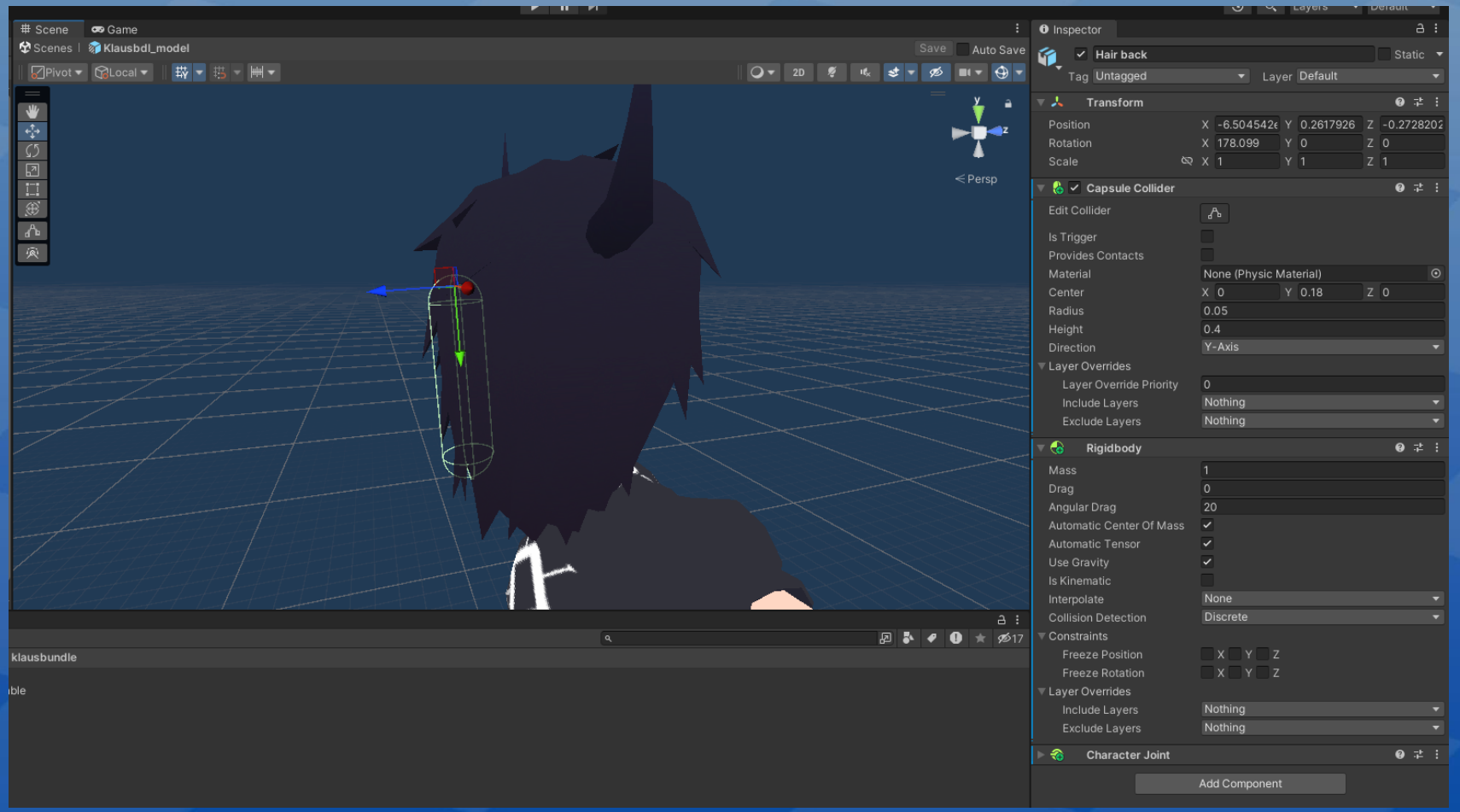
- To adjust the size, click on the Edit Collider button and use the little green dots to fit the collider to the head.
- You can hold Shift for proportional scaling, and Alt for symmetrical scaling.



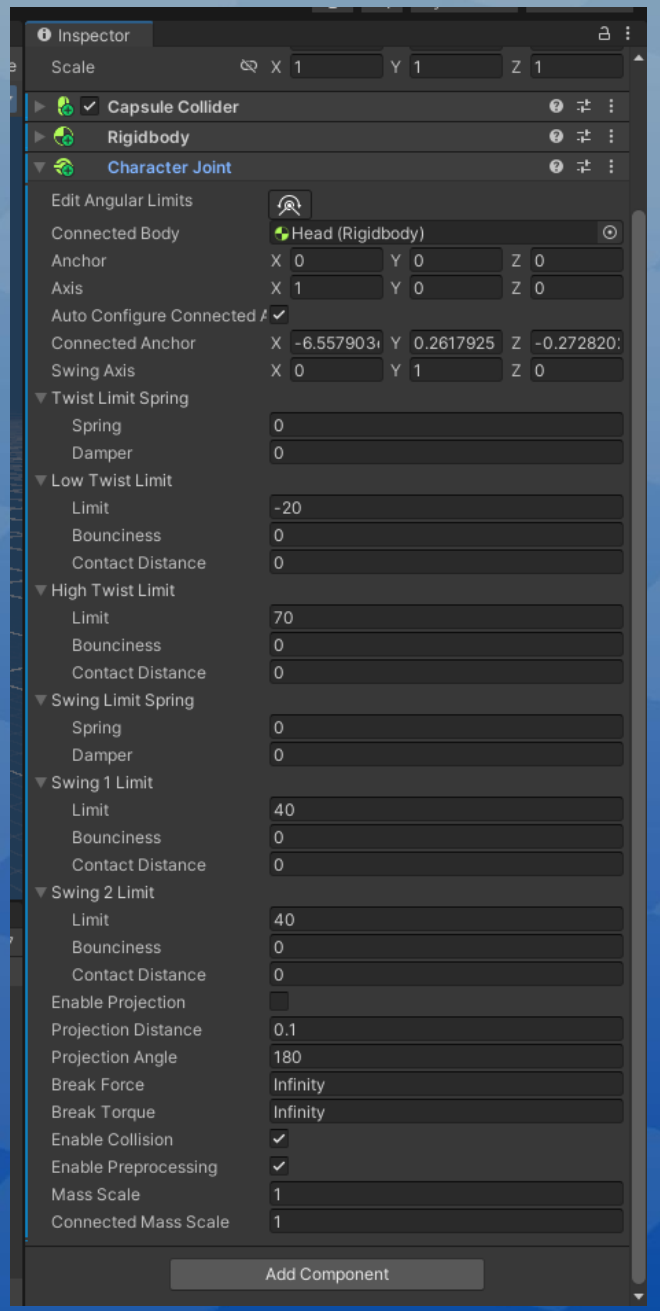
- Next, for hair physics.
- If you have hair physics, go ahead and also add a Rigidbody component to the head bone.



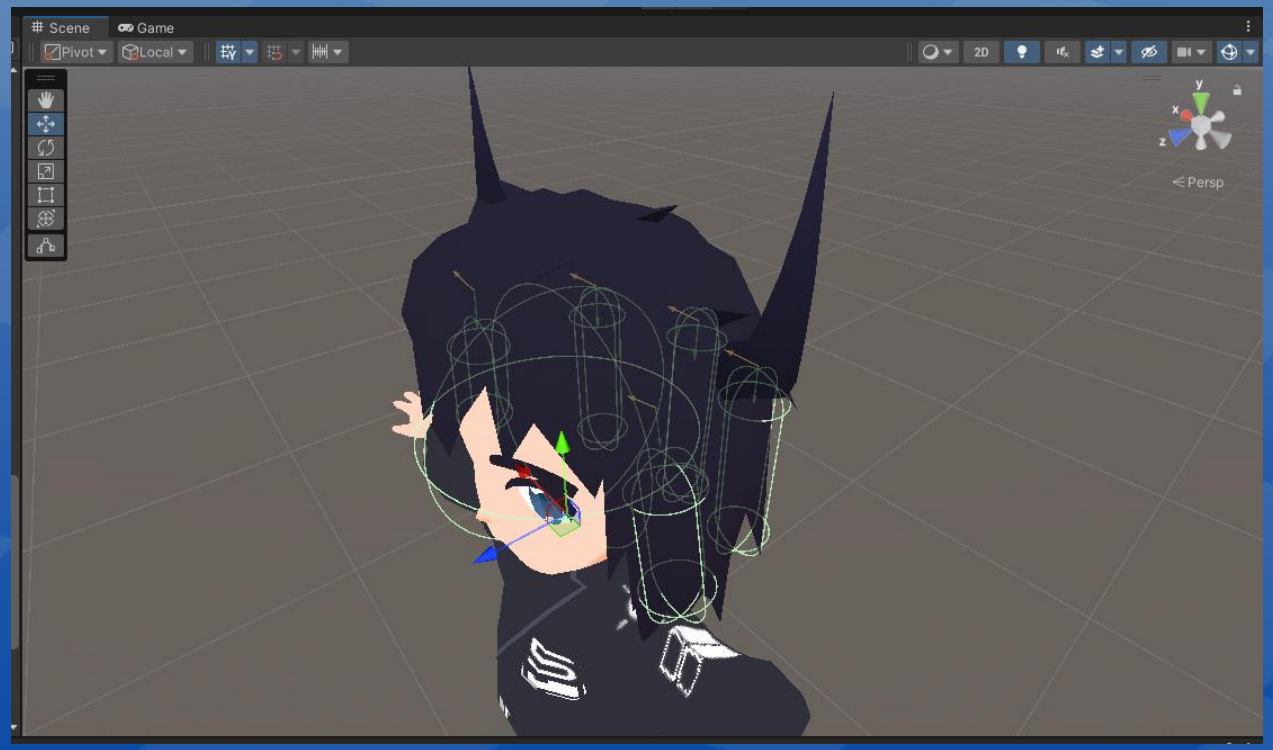
- Select your hair bones and add three components: Rigidbody, Capsule Collider and Character Joint.
- Like the head, try to match the Capsule Collider to the hair bone.

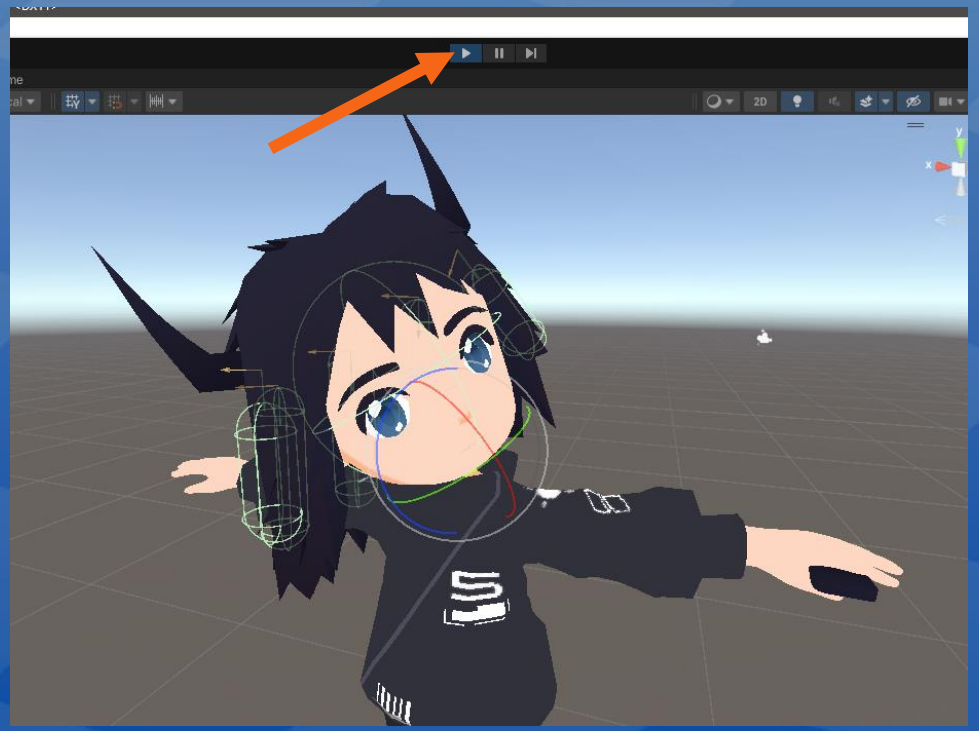


- For the Character Joint component, try to match the settings in the screenshot:
- (You don't need to match the Connected Anchor, its automatic)
- You may need to edit the Angular Limits by dragging the handles in the viewport. Do that when testing the gravity (next page).
- **IMPORTANT!**
  - Where it says "Connected Body", the second setting you see, you need to pick the parent bone of that hair.
  - In my case, I don't have a hair chain, but you might do, so each bone will have its parent as the Connected Body.



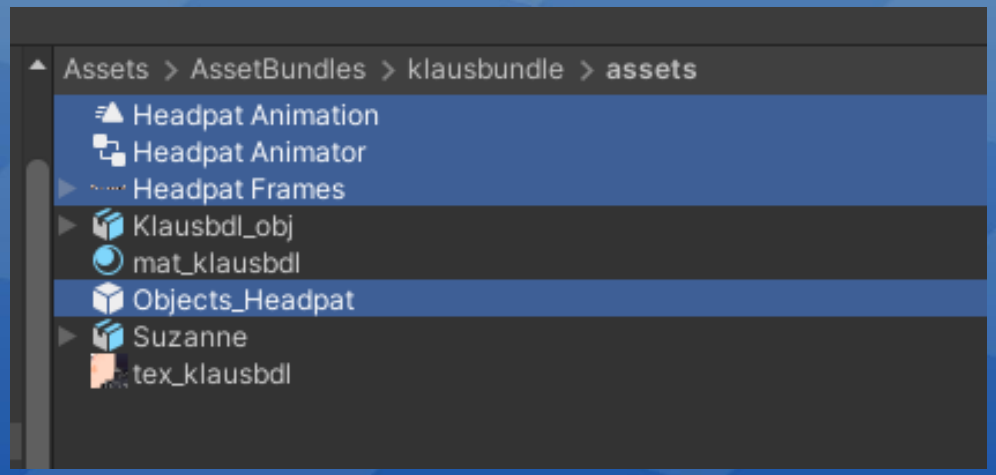
- By the end you should have something like this, when selecting the head bone:
- You can test it by clicking the Play button on the top and rotating the head bone to see the hair falling.
- Here you can adjust the Angular Limits, but be aware that changes made in Play Mode do not save. So exit Play Mode, make your changes, and test it again.

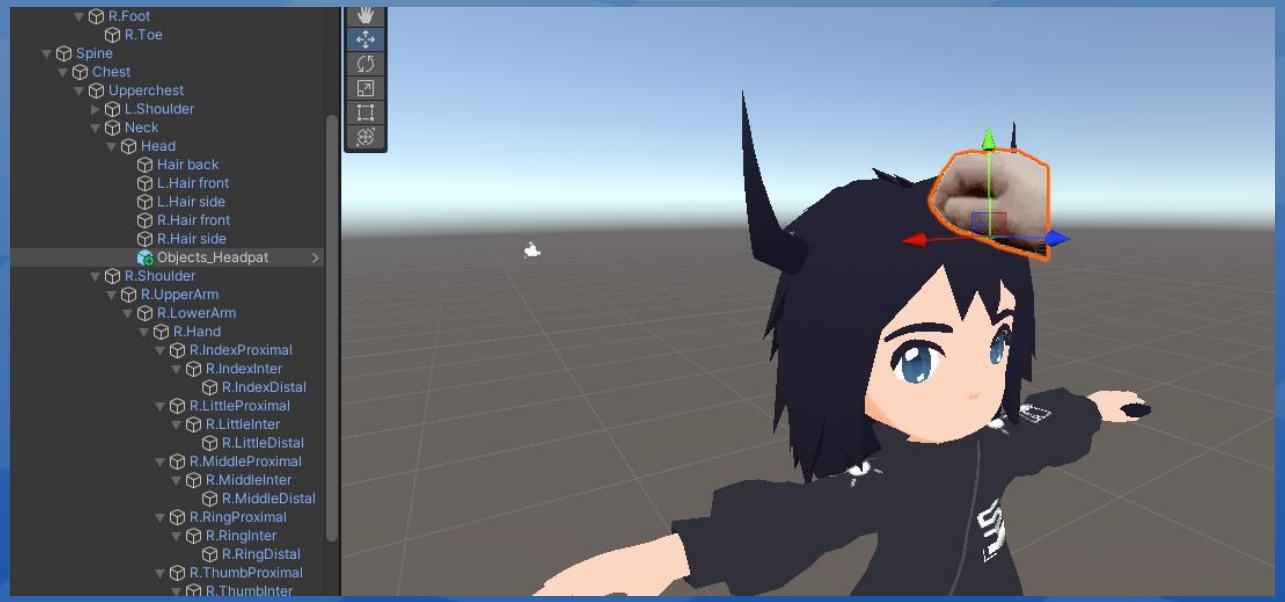




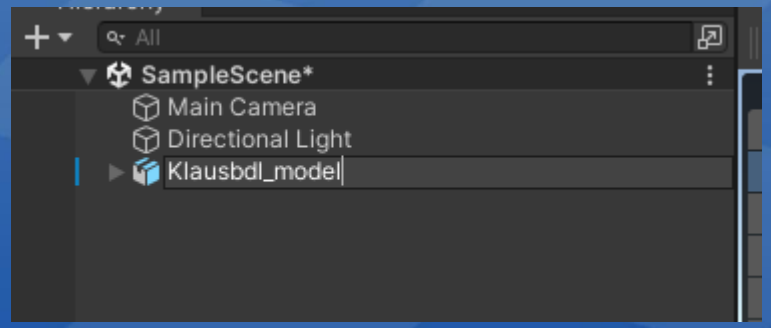
- For the object toggles:
- If you expand the armature, you may find the objects you parented to the bones before. For example, my cellphone object.
- But you can also just import any model and parent it here, in Unity. Just make sure to rename it “Objects_name”!

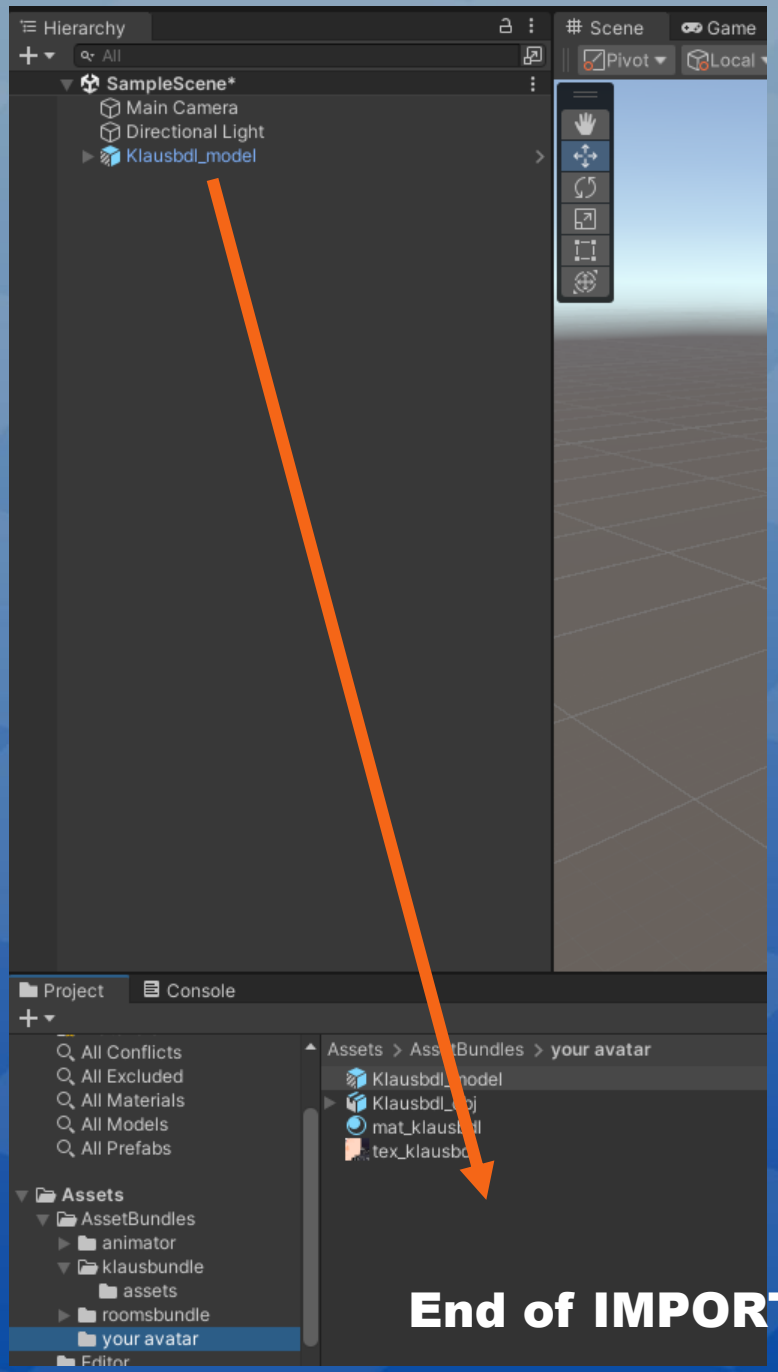
- In the Bundle Project, under AssetBundles/klausbundle/assets you can find a “headpat” object. Copy these 4 assets and paste them in your folder.
- Then, you can drag the “Objects_Headpat” prefab to your head bone and fix its position and scale.





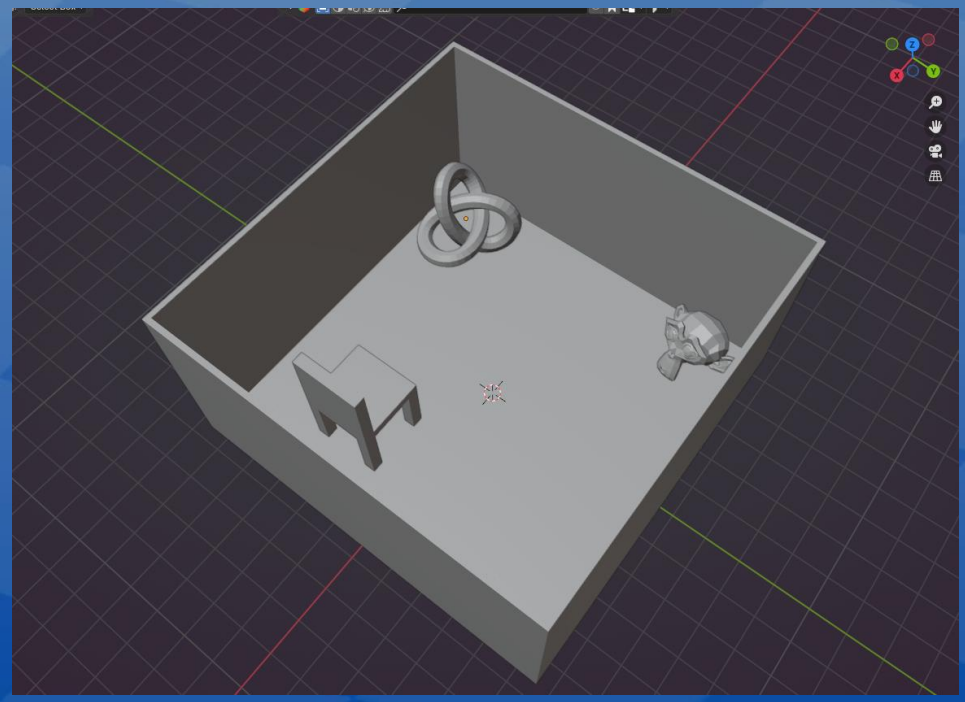
- Hard part is done!!!! Finally!!!! I can’t take this anymore
- Select your avatar in the Hierarchy window and rename it to “name_model”.
- Then, drag it from there to your folder, this will make it a prefab.



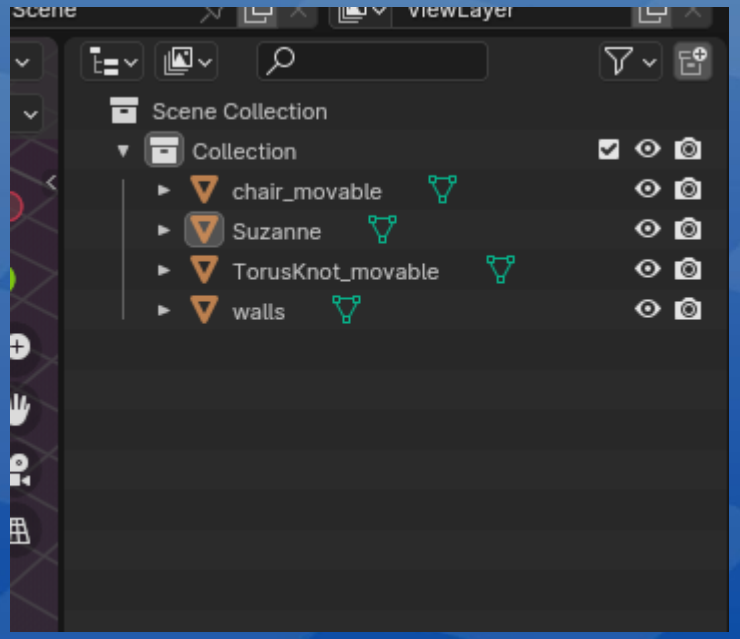


5. MAKING A ROOM

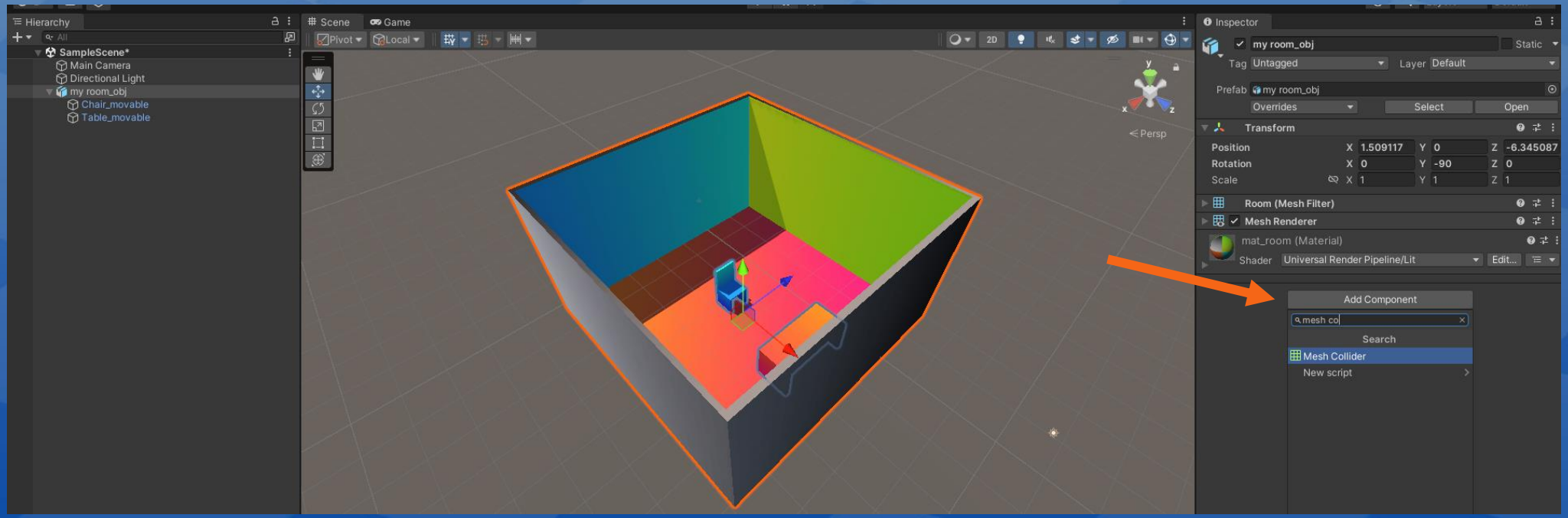
- You are as free as the wind to create a room the way you want. There are no constraints to poly count, number of objects etc…
- You can get models from the asset store and assembly a room in Unity as well.
- Example room with random objects:



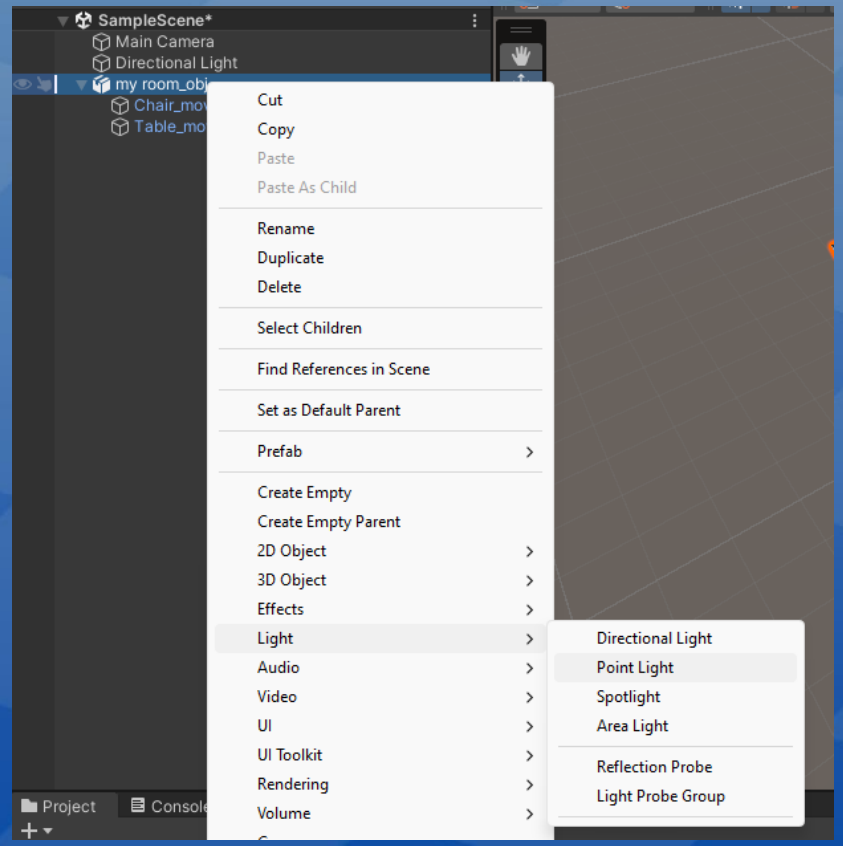
- If you want an object in your room to be movable, you need to rename it like so: “object_movable”. You can rename it in Blender or later in Unity.
- After your room is done, you can export it following the same rules as seen in the 3rd chapter of this guide.
- You can save it in the same folder as your avatar model, or in a new folder dedicated to it.

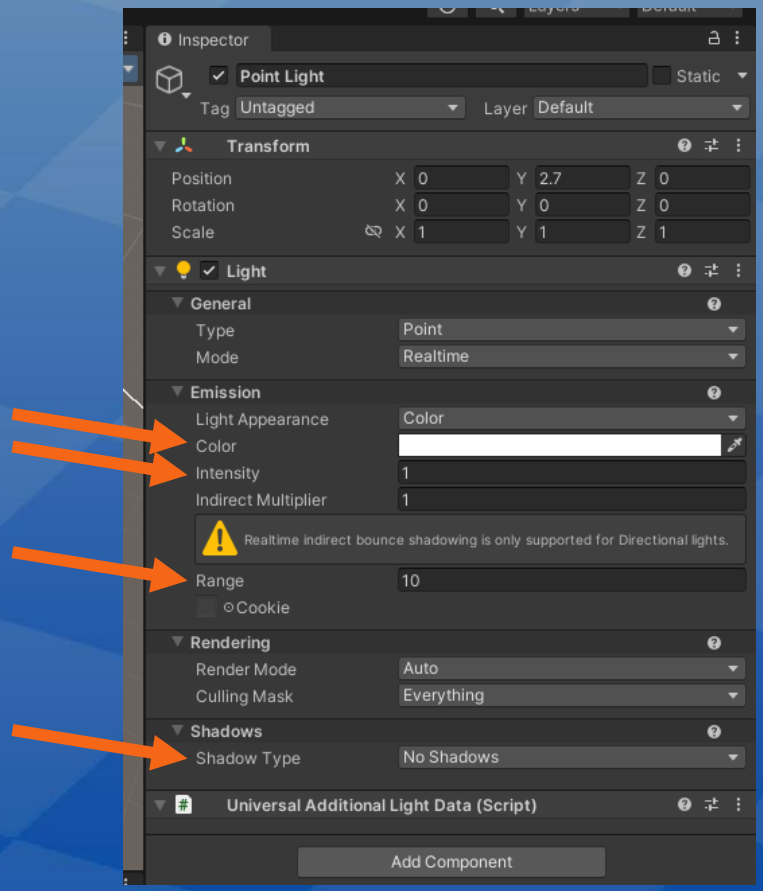


- In Unity, you need to add collision to your room so your avatar doesn't fall in the void in Play Mode.
- To do so, first drag from the Project window the room FBX to the scene.
- Select the room and add a Mesh Collider component.
- Do the same for the objects inside. You can also use Box, Sphere and Capsule Colliders.

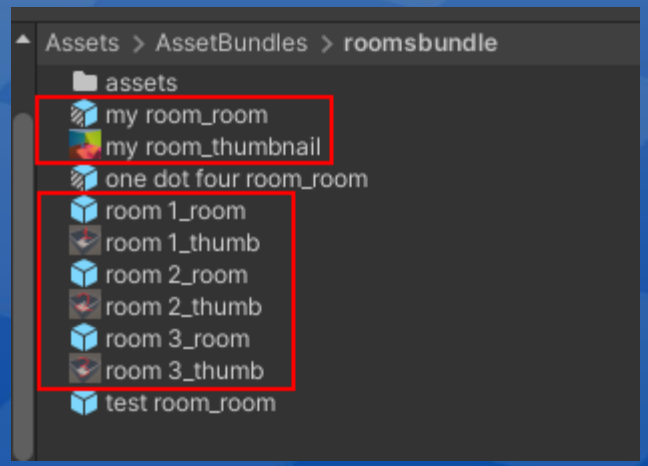


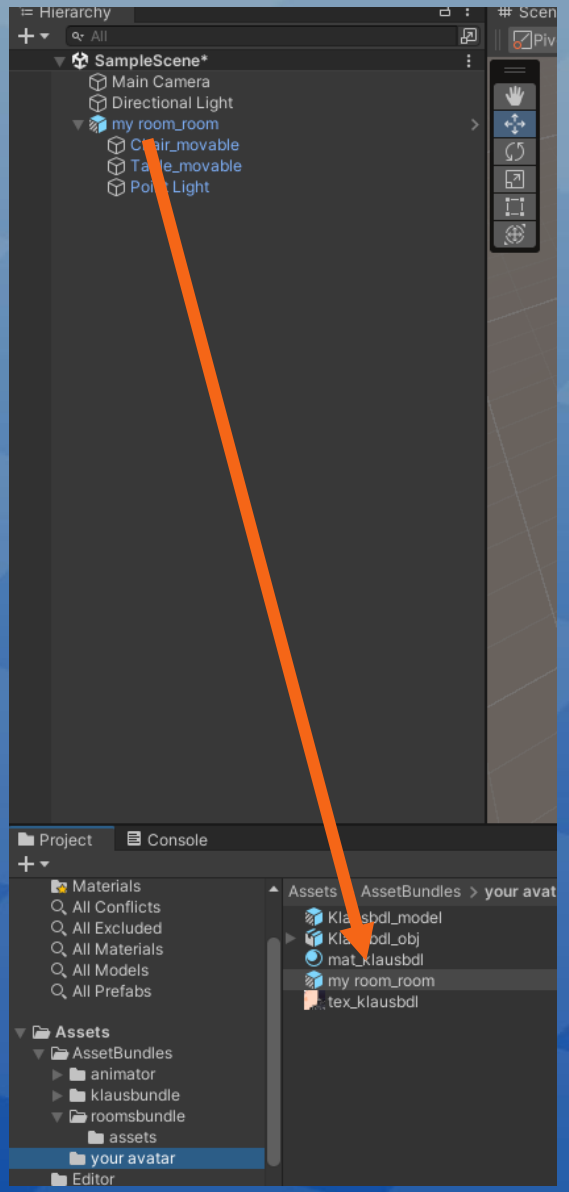
- While at it, you can also add point and spot lights to your room!
- To do so, right click on the room object in the Hierarchy window > go to Light > add any light you want. You can change its properties in the Inspector window.





- When you finish editing your room, rename the object using the naming convention “name_room”.
- Then, drag from the Hierarchy window to the Project window to create a prefab of it.
- EXTRA:
  - You can add a thumbnail so its easier to choose the room in the app.
  - Just import an image to the same folder and give it the same name as the room, but ending in “_thumb” or “_thumbnail”.

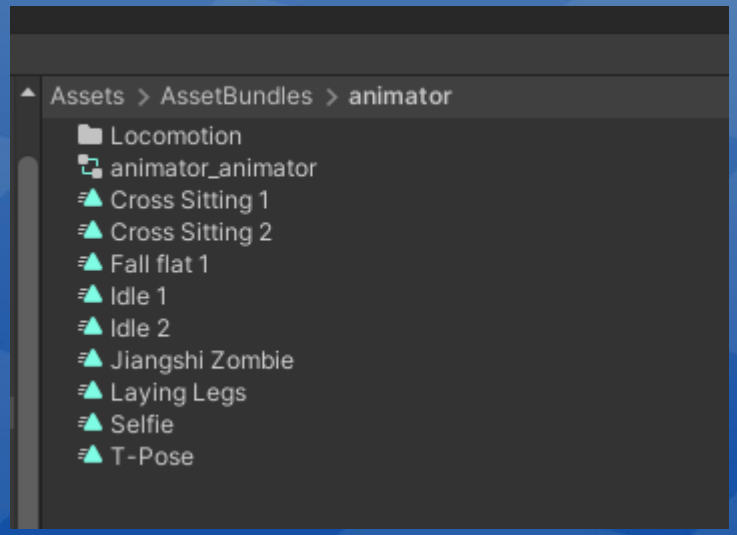




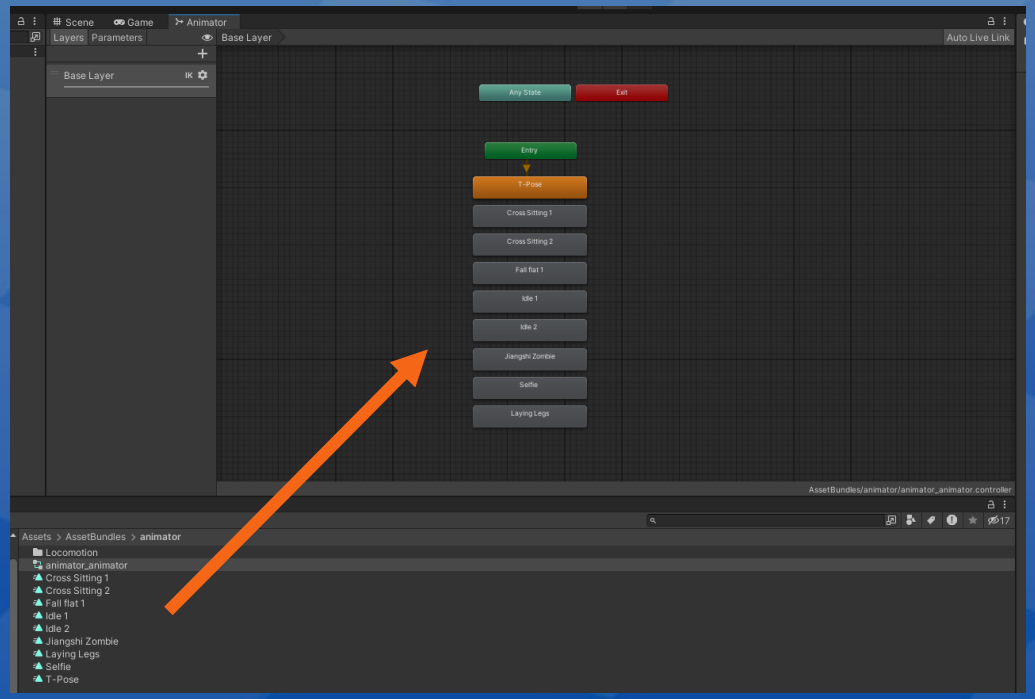
End of ROOM

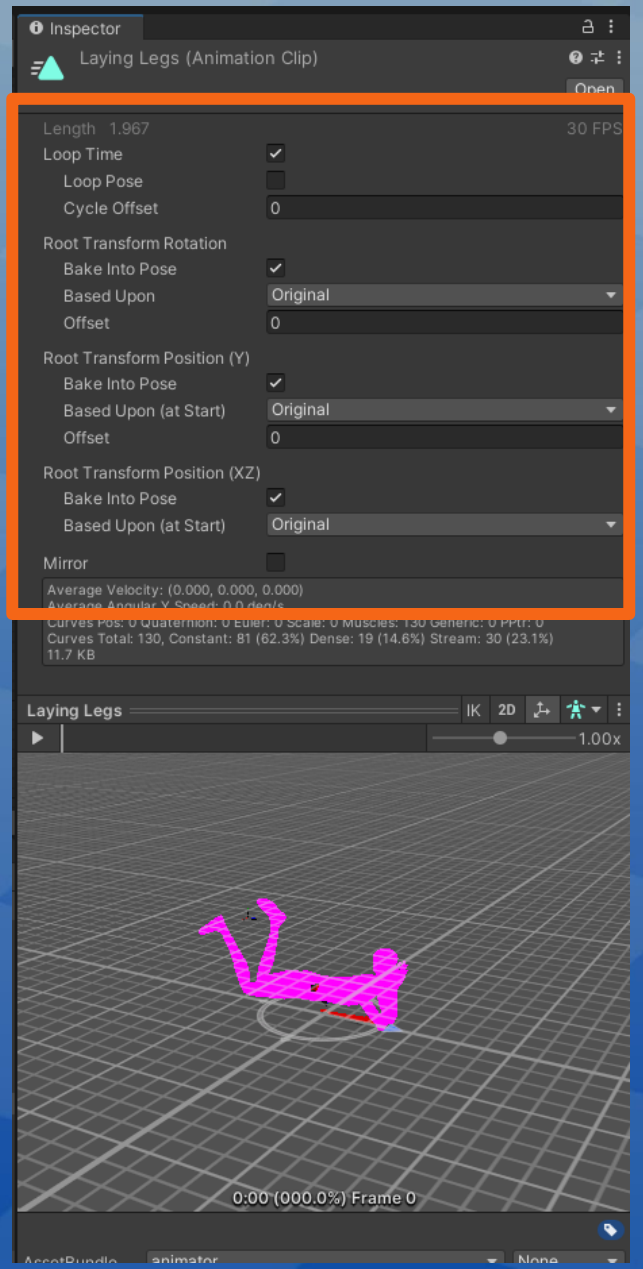
6. ANIMATIONS

- Inside the AssetBundles folder, there is one folder called “animator”. That folder is extremely important. It has all the animations the avatar can make in the app, which means you can add your own Humanoid animations from Mixamo or vrcmods.
- More on that in the next pages.
- Mixamo
- VRCMods

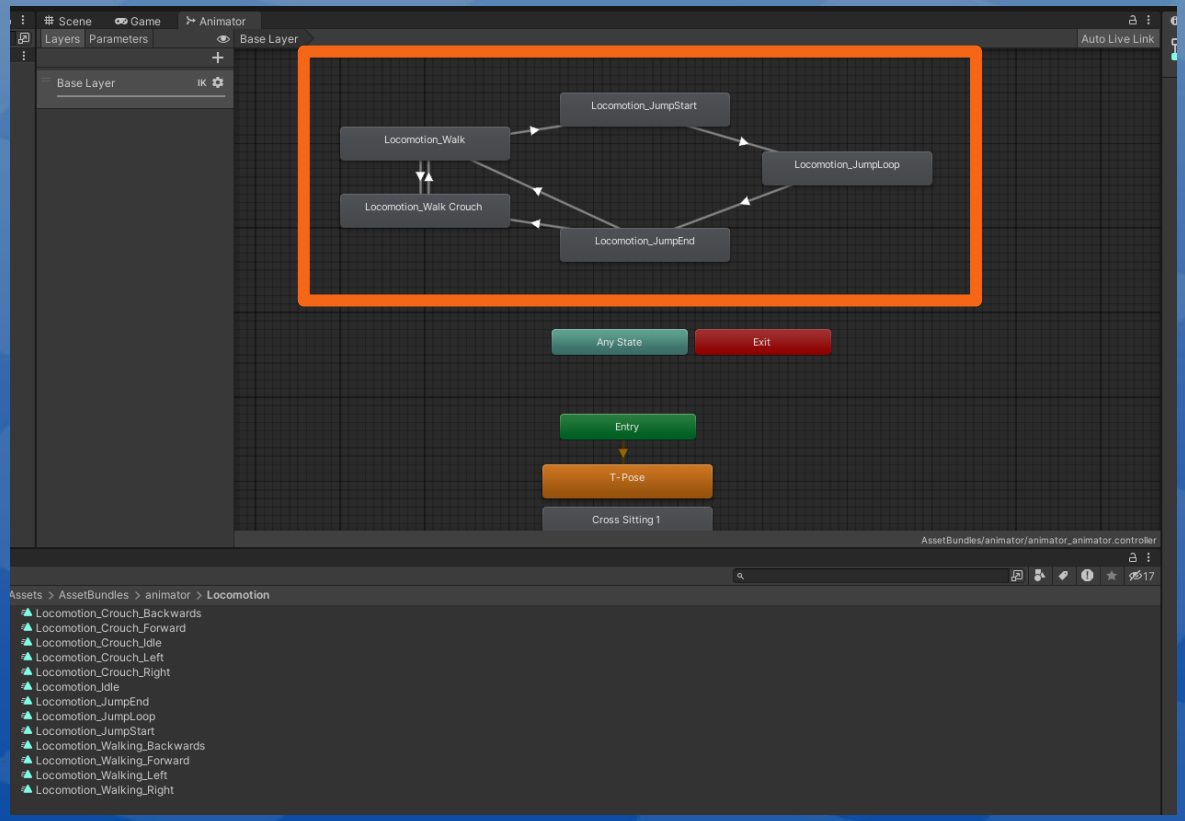


- You can double click the “animator_animator” file to see all the animations inside. Here, you can drag in your new animations and leave it anywhere in the area.
- Make sure your animations are all inside the animators folder, so they are loaded together.
- When importing new animations, make sure to have these settings:





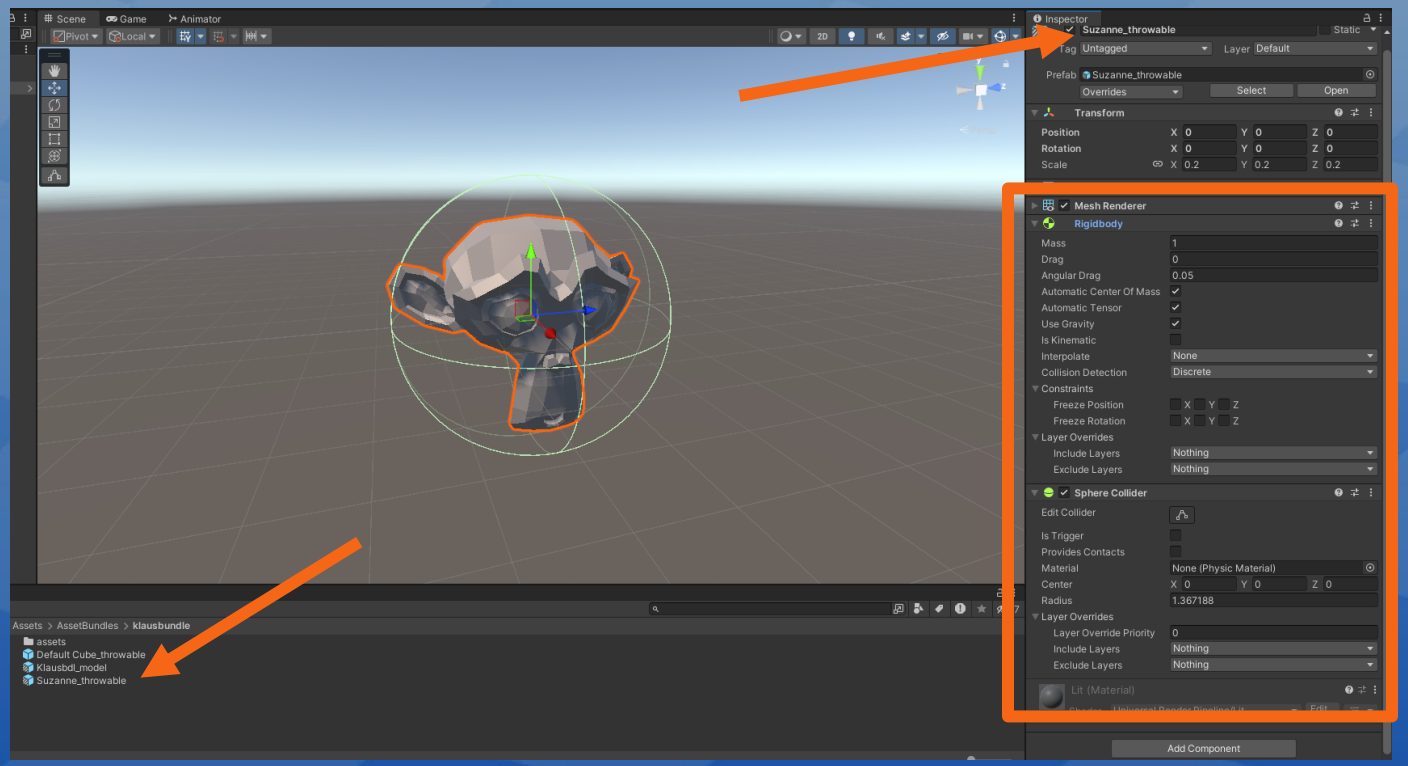
- You can also see the Locomotion folder. There are the animations used in Play Mode; animations of walking, jumping etc.
- If you know what you are doing, you can edit them as you wish! Just replace them on the setup above the other animations.



End of ANIMATIONS

7. THROWABLE OBJECTS

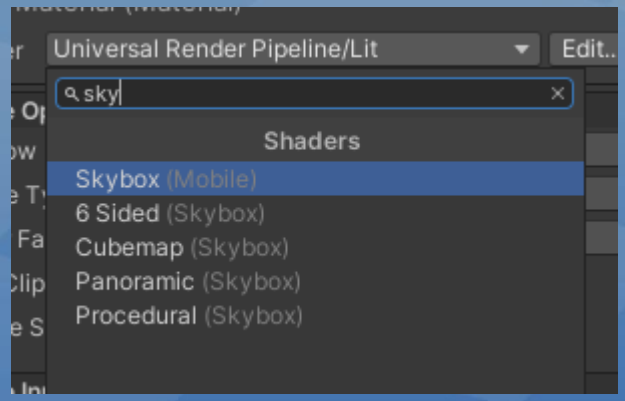
- To put it simple: throwable objects are just prefabs named “name_throwable”.
- First, import any model and drag it into the scene.
- Give it a Rigidbody and a collider component.
- Tip: You can drag in your avatar model to the scene and check their scales.
- Then, just drag from the Hierarchy window to the Project window to create a prefab of it.

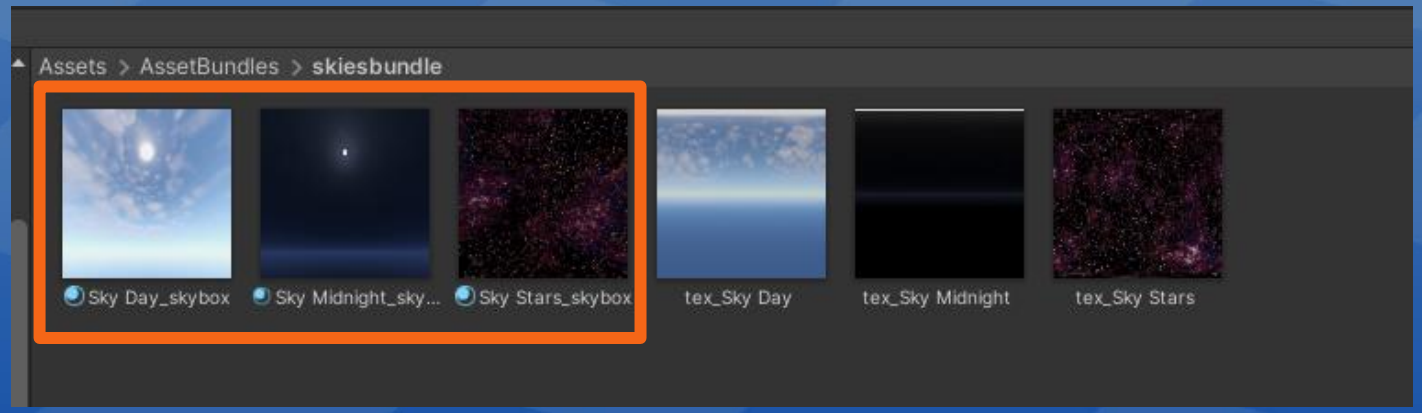


End of THROWABLES

8. SKYBOXES

- Skyboxes can be made using HDRs or 6-sided textures.
- Create a new material and select the desired shader to be used and add the textures.
- Save the material ending in "_skybox" and add it to your bundle, both the material and the textures.





End of SKYBOXES

9. ASSET LIBRARY

- The Asset Library will display any assets in your bundle that has its name ending in "_asset".
- It has to be a prefab (we've already learned how to create those) and they need a Rigidbody component and a Collider. Just like a Throwable.
- On the right, an example of a cube with the components. Its got the same material as my avatar for no specific reason.

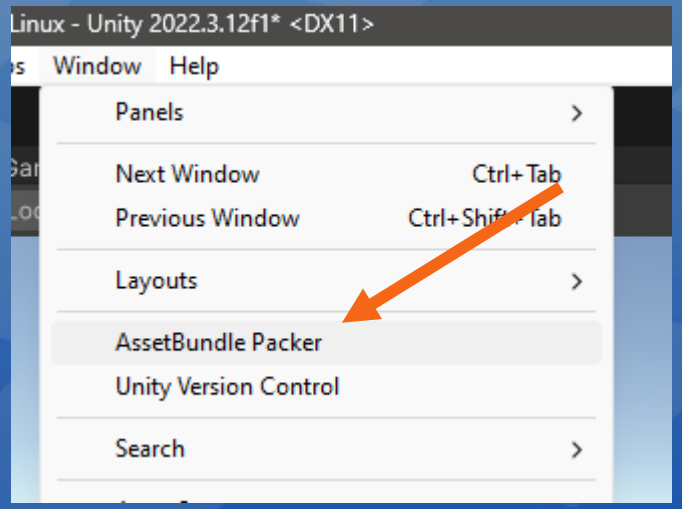
End of Asset Library

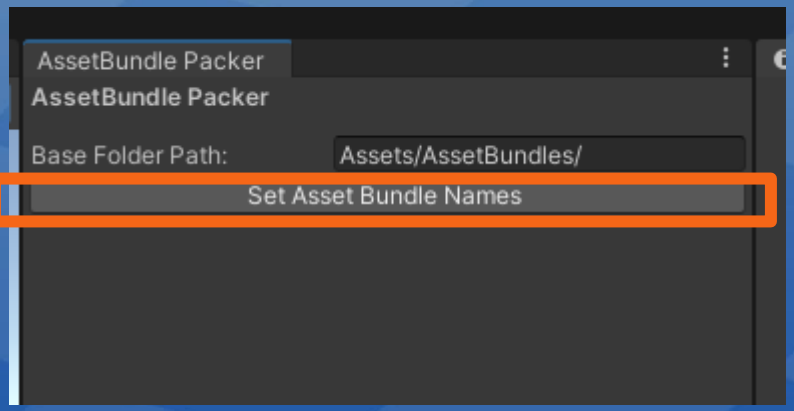
10. BUILDING THE ASSET BUNDLES (FINAL)

- Do a last check up:
  - The avatar models should be named "name_model".
    - Put at least a collider on the head bone
  - The room models should be named "name_room".
    - Add colliders to the objects in the room.
    - Room thumbnails should have the same name as their room, but ending in "_thumb" or "thumbnail".
  - The throwable models should be named "name_throwable".

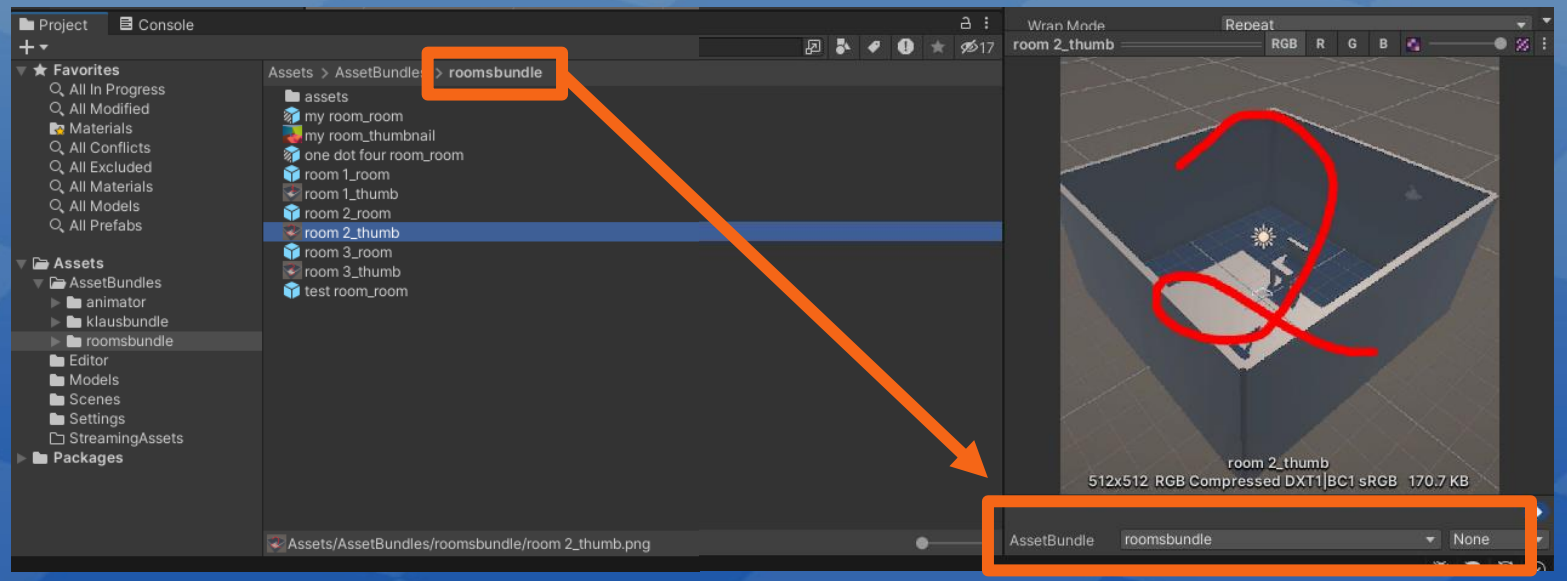
- I recommend doing the same as I did on the samples that comes with the Bundle Project: anything that is a material or a texture, put it inside an "assets" folder, and leave the important files separated from them. Just for organization.

- Next, you will go to Window > AssetBundle Packer to open the new window.
- Just click the button!
- This will automatically set every file in the AssetBundle folder to its appropriate bundle in the engine.

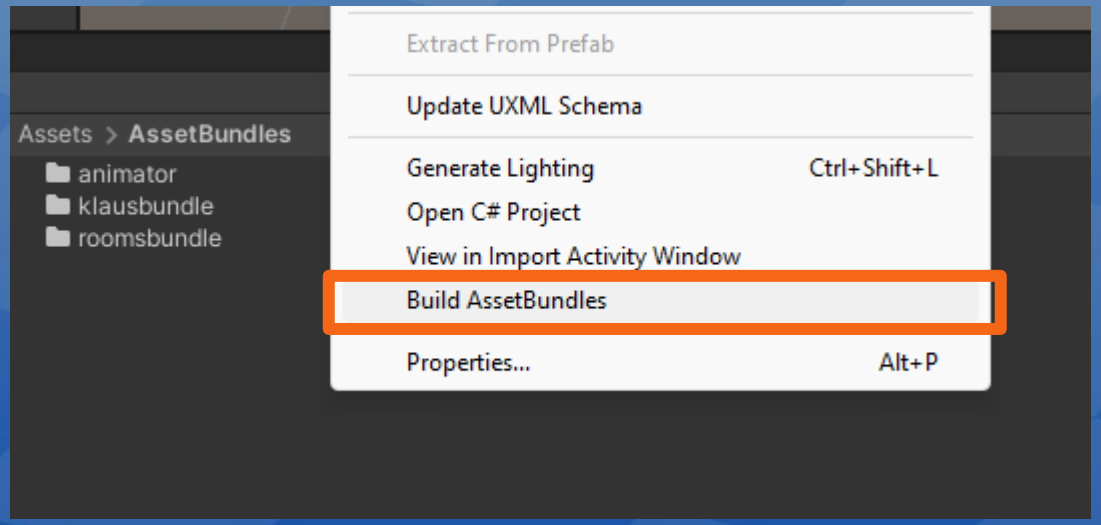


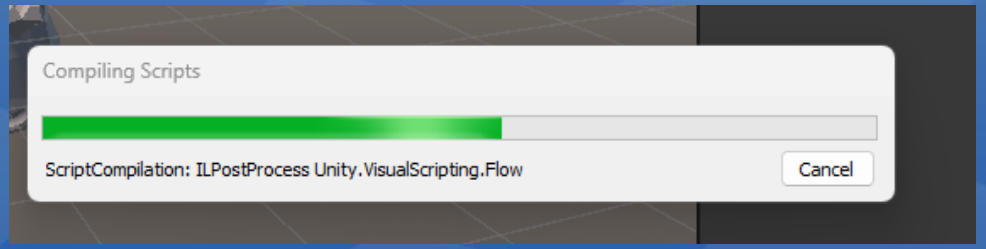


- Double check it. Open your folder and check if all the files belong to the asset bundle with the same name as the folder.

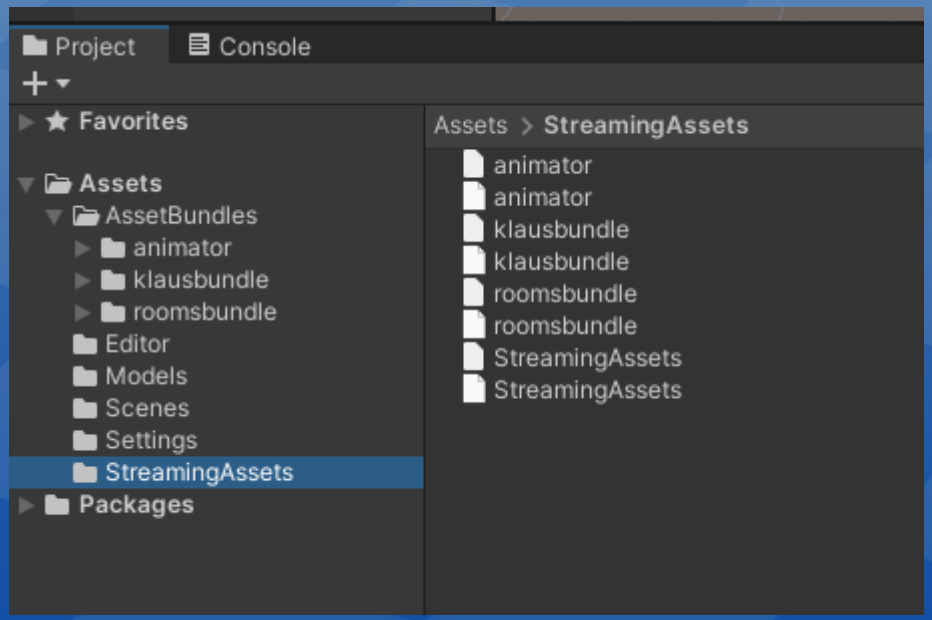


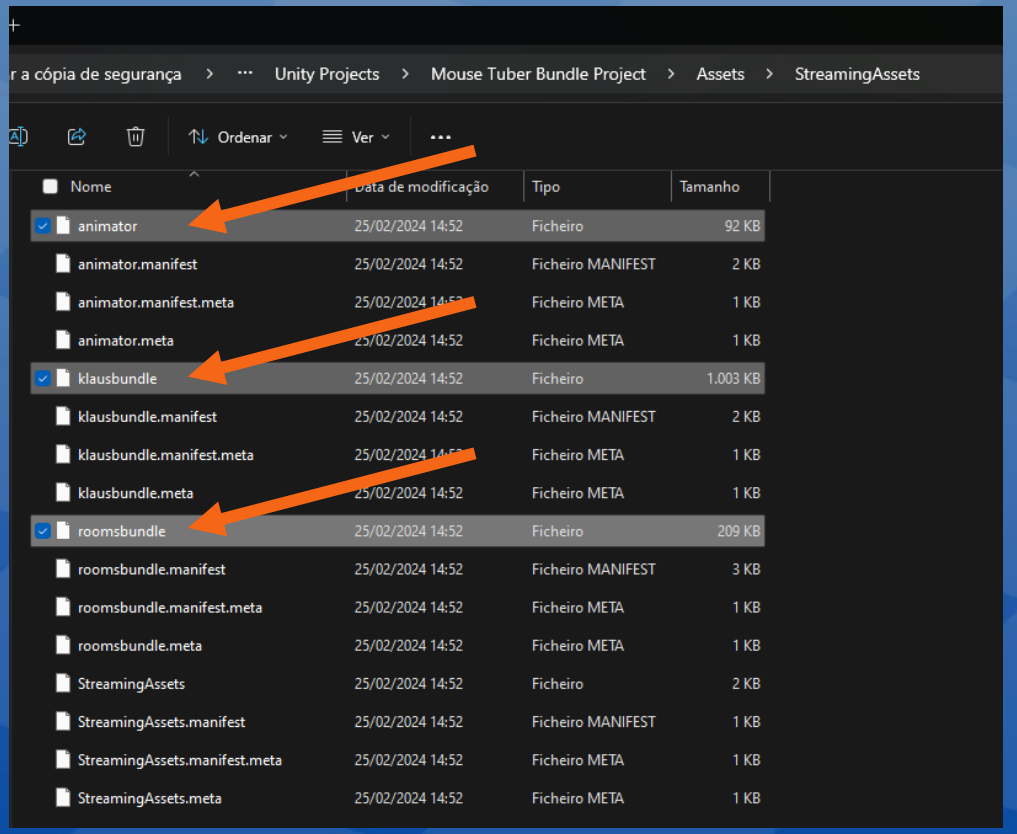
- Finally, you just need to right click anywhere empty on the Project window and click “Build AssetBundles”.
- Wait for it to finish building it.





- After it finishes, the /StreamingAssets folder should be filled with new files. Those are the files you need to place inside the app folder.
- You only need the “animator” and your other bundles. You can ignore the ones that end in “.manifest” and “.meta”. You can also ignore the “StreamingAssets” file.





- Inside the Mouse Tuber app folder, navigate to Mouse Tuber_Data/StreamingAssets and paste them there.
- You're done! You can now open the app and all the models will be loaded properly.