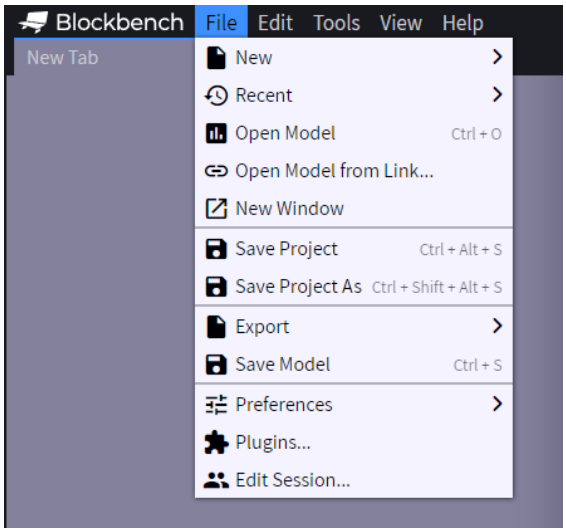
Important Links

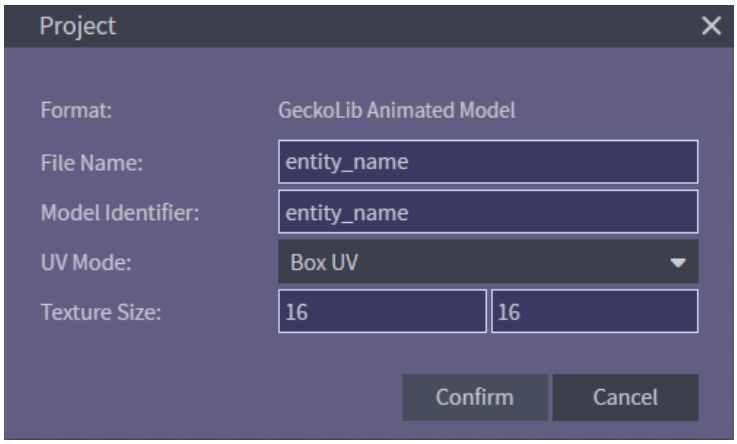
MCreator Version 20022.3: https://mcreator.net/download/eap-2022-3-44708
MCreator GeckoLib Plugin v 2.4:
https://mcreator.net/plugin/91484/nerdys-geckolib-plugin-forge-1182-1192
GeckoLib Changelog: https://mcreator.net/forum/91485/tutorial-how-use-nerdys-geckolib-plugin
BlockBench Download: https://www.blockbench.net/downloads
BlockBench Plugin: go to File\Plugins and search Gecko, should be first option



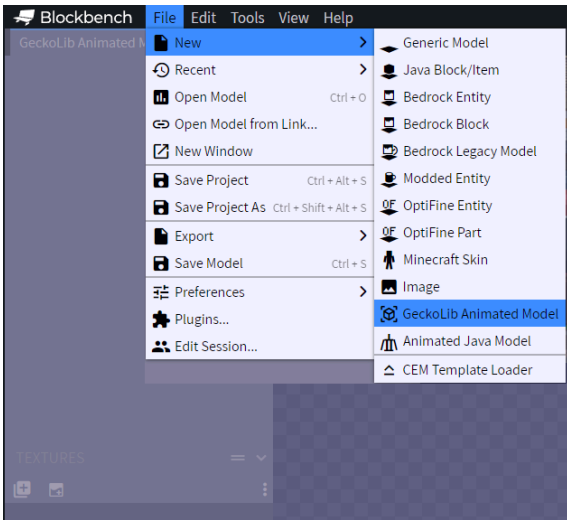
How to Create GeckoLib Model

Once you've installed the version of BlockBench above (any semi recent version should be fine) and the BlockBench GeckoLib plugin, the first step is to create the correct model file.

Go to File\GeckoLib Animated Model and create a new project.

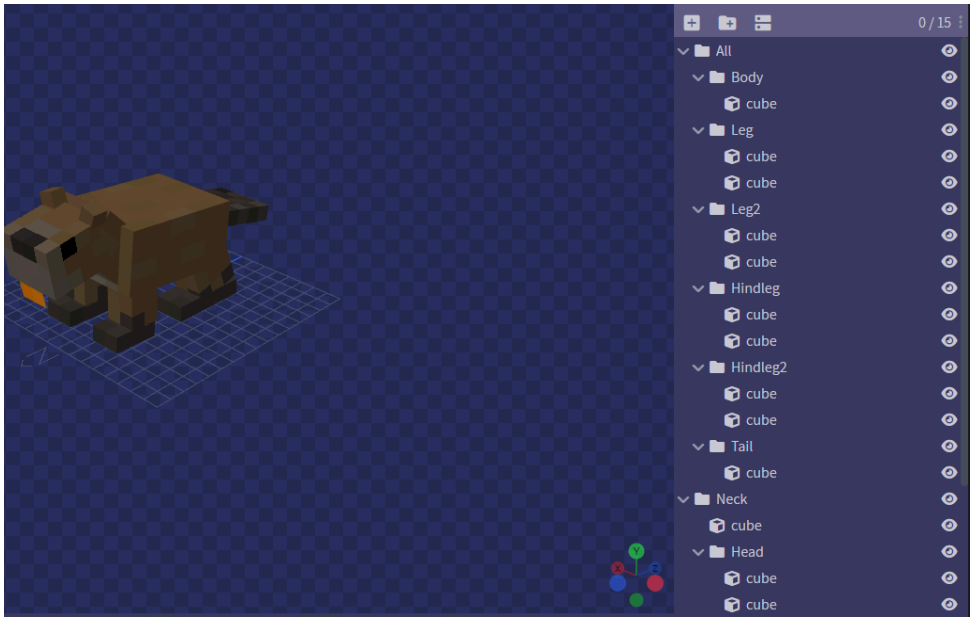


File

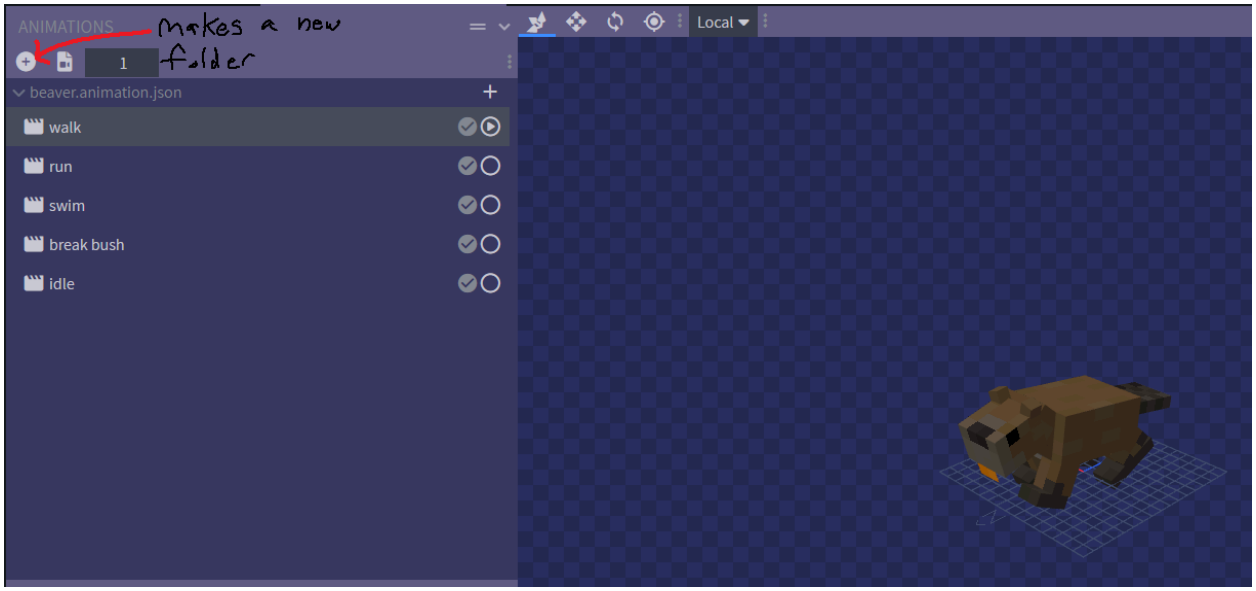


Name: this is the name of your entity.

The entity you create and reference in MCreator should match the File Name exactly, so for simplicity's sake they should have no capital letters or spaces, use _ instead.
Model Identifier: this is the name of the model itself, which usually cannot contain capital letters or spaces. Again, use the same name. This also makes it easier when referencing the models and entities in MCreator to ensure the correct name is used



While making the model using BlockBench, the cubes making it up need to be sorted into folders in order for them to be animated. You can only animate the folders as they make up the "bones" of the entity. The folders and cubes inside them can be named anything with any capitalization, although the folders can't have the same name.



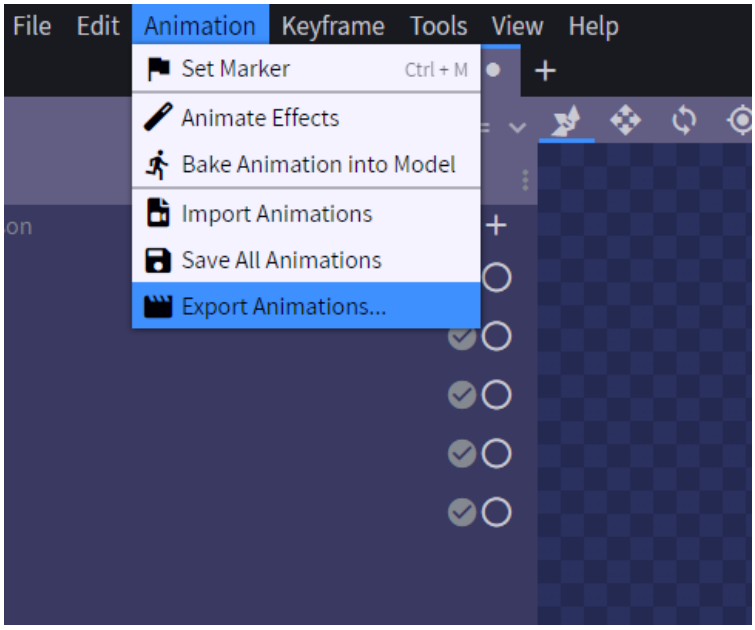
The animations themselves are slightly more complicated. Click the plus button at the top of the page to make a new folder to hold the animations. This folder should be named “entity_name”.animation.json, in the case above the beaver is the name of the entity.

The animations you create can be automatically used by the GeckoLib plugin in MCreator by naming them certain things. If you use one of the terms below as the name of an animation, once the model is established in MCreator, that is all that has to be done for the animation to work in game when the entity does whatever the word is.

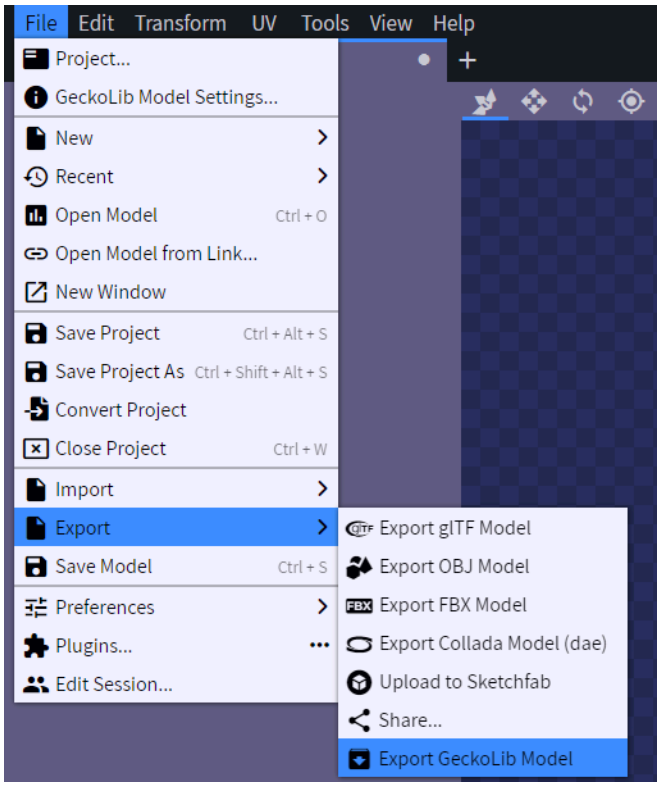
<u>Animation Plugin Code Names (Case Sensitive):</u>

- walk
- death
- swim
- sneak
- sprint
- idle
- attack

For custom animations like “break bush” in the example above, they will need to be called in a procedure in MCreator.



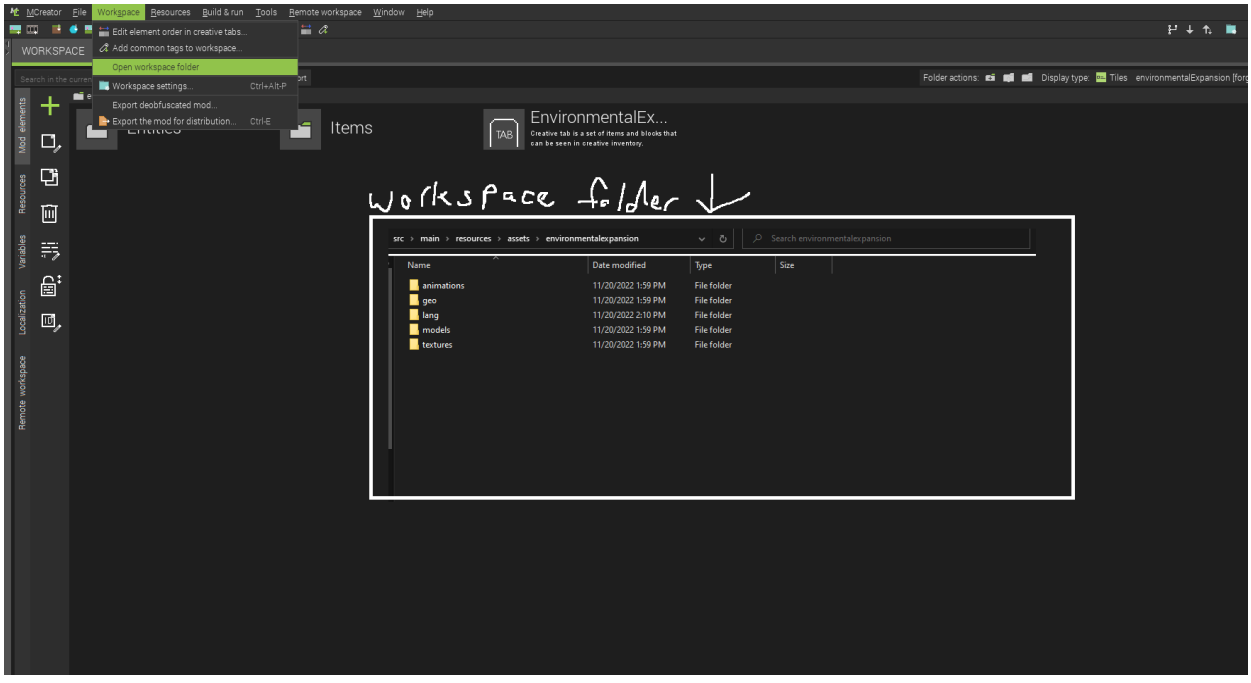
Finally, the animations can be exported in one file that will be used in MCreator. Click Export Animations under the Animation tab and choose all of the animations to include in the file. If the file was named “entity_name”.animation.json at the start it will be named correctly, otherwise that can be fixed when you export the animations.



The same can be done to the model itself, go to File/Export/Export GeckoLib Model. It will be named “entity_name”.geo.json, but “entity_name” can be corrected here if it wasn’t named properly earlier. Both this file and the animations file are case sensitive and need to be named after the entity in order to be used in MCreator.

Adding Files to MCreator:

To add the animation and model files to MCreator, open the workspace folder as demonstrated below. Navigate to the assets folder by opening src/main/resources/assets/name_of_mod. name_of_mod will be whatever the MCreator workspace was called.



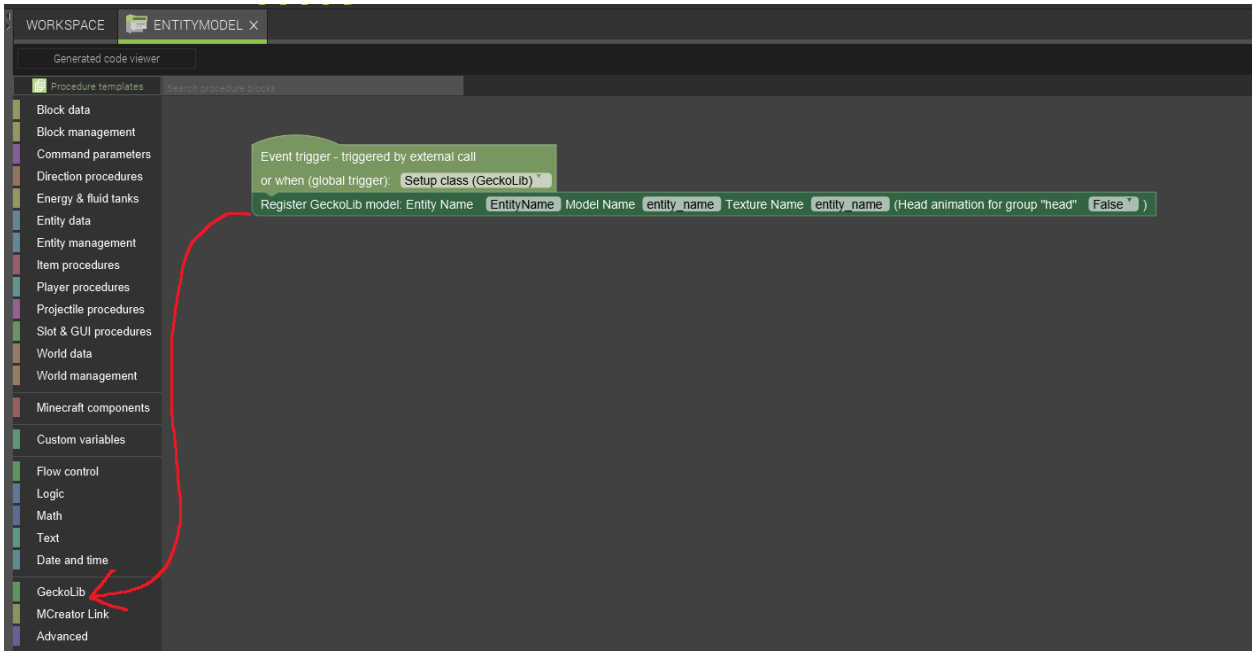
The animation file exported from BlockBench goes in the animations folder, the geo.json file goes into the models folder.

Creating the Entity

By adding the files in this way, MCreator will not need to have a separate json model imported in the Resources tab, nor will you need to select a custom model when creating an entity. The only thing you'll need to upload in the Resources tab is a texture file. It can be named whatever you want, but you will need the name of the texture when registering the model. Choose that texture to represent it and change its bounding box as necessary. For model type, you can select the one with the most similar bounding box to make it easier to edit.

Registering the Model:

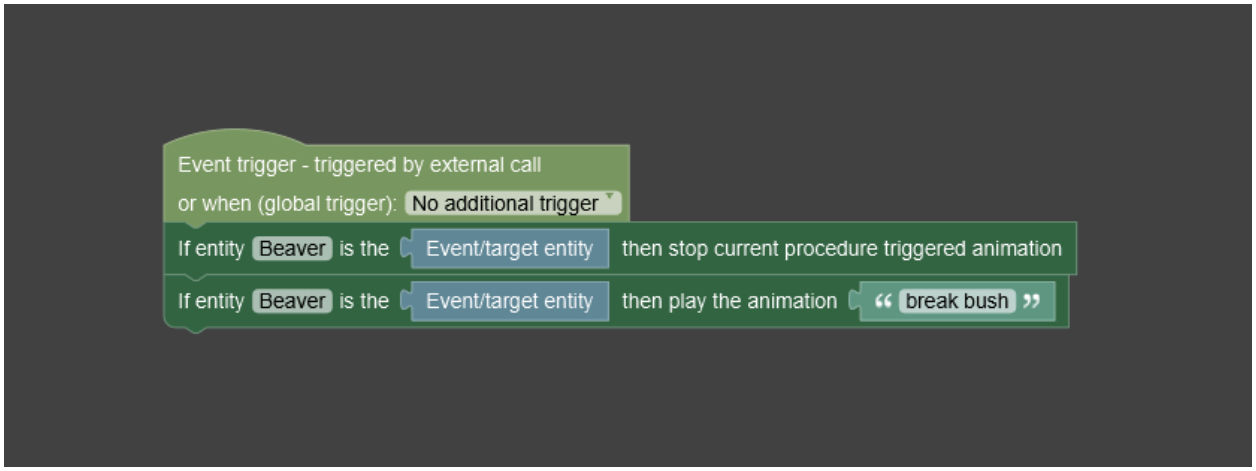
Only one procedure is needed to attach the built in animations to the model. Create a procedure and choose the trigger "Setup class (GeckoLib), then add the Register GeckoLib model block from the GeckoLib tab. They first box should have the EntityName as it was created in MCreator (so for example if you have an entity with model entity_name.geo.json and entity_name.animation.json, the name of the entity you make in MCreator would be EntityName. EntityName is what you put in the first box). The second box Model Name takes the name in front of the geo.json file (for model entity_name.geo.json, put entity_name in the second box). Finally, the third box Texture Name takes the name of the texture file you uploaded to MCreator through the Resources tab.



I haven't messed with the Head animation thing much, I assume it chooses the head bone if your model has that and applies the minecraft default head bobbing when idle and stuff, but I'm not sure.

Adding Custom Animations:

Adding custom animations can be somewhat tricky because the built in animations (idle, walk, attack, etc) seem to take precedence over an animation you call through a procedure. To add a new animation, create a procedure without an additional trigger. This procedure can be added in the Triggers tab of the entity model, triggered by right clicking, when the entity dies, or it can be a constantly running procedure with an if statement to check when it should run.



As shown above, the first block can be used to cancel the current animation, and the second is used to call the animation according to its name in the entity_name.animation.json file (the name of the animation can include spaces but is case sensitive. Open the animation.json file to check what it's called to make sure it's correct).

The first block is supposed to cancel the current animation (like if an entity walked towards something and then ran a custom animation, the first block should cancel the walk animation to run the custom animation). However I've had the issue where the custom animation doesn't want to run sometimes. It could be a buggy thing due to MCreator or the plugin, or there might be an optimized way to do it that I'm not privy to, shrug.

But anyway that's most of what I know. Check out videos on Youtube or MCreator's help forum on their website to troubleshoot. o/