Render Clonk Addon – Manual

An addon for Blender 3.0+ to produce graphic assets for the game LegacyClonk and Clonk Rage. It can import meshes/animations of Clonks and it contains a spritesheet renderer which can render things like Clonks, buildings, vehicles (etc.) and export them directly to these games.

This manual should provide enough information for beginners to get started and it should be proficient to guide advanced Blender users to create new spritesheets with this addon.
If something is unclear or unintuitive, don't hesitate to ask in the forums or on Discord.

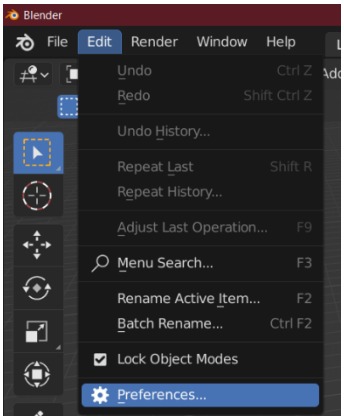
Contents

Robin Hohnsbeen (Ryou/RoboClonk) 2022

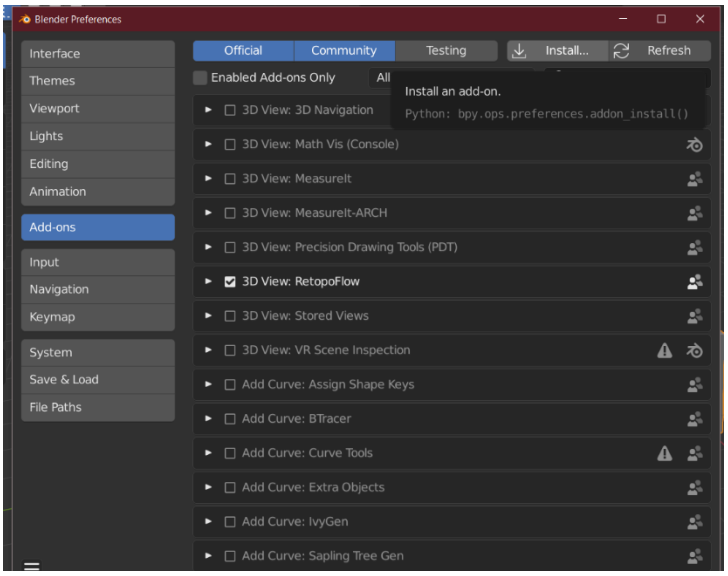
1. Getting Started

a) Installation

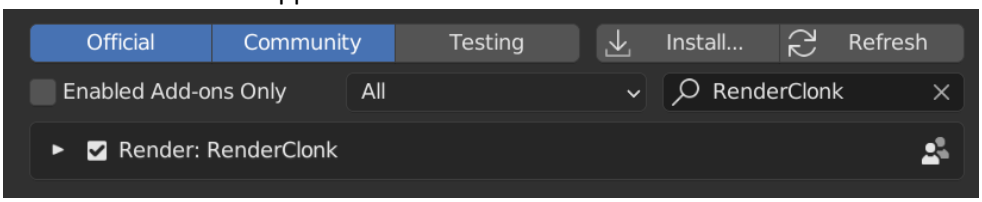
1. Go to the releases to get the latest stable version of the addon or download the repository directly as zip-file. You don't need to unzip the addon after you downloaded it
2. Start Blender and navigate to 'Edit->Preferences...' then Select the tab 'Add-ons' in the new Window



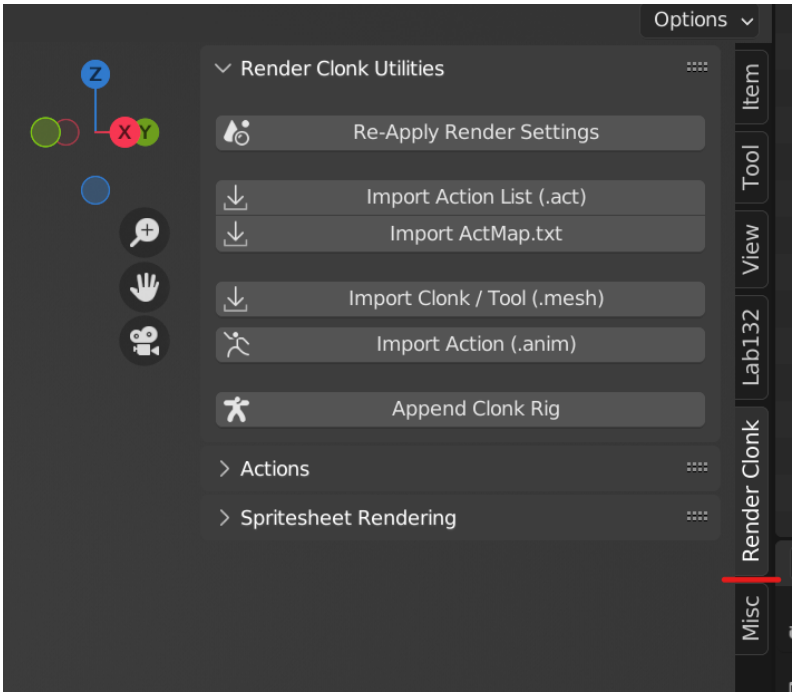
3. Click on 'Install...' at the top right corner and browse to the zip-file via the filebrowser of Blender



4. Make sure the addon appears in the list and the checkbox on the left is checked.



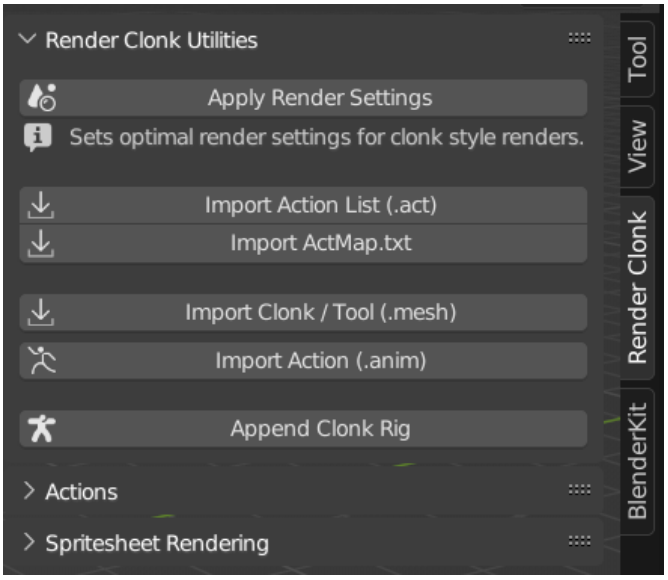
5. The addon can be found in the 3D View inside the properties panel (Open with 'N'). There is a Tab called Render Clonk



b) Making new Clonks or other Clonk Graphics

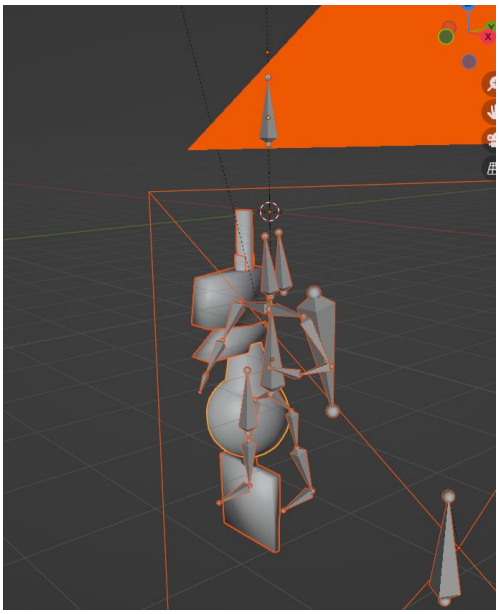
If you want to have a quick start, the repository contains some example files, that you can use. There is a Clonk, Ozark or Hut that contain everything the addon needs to generate a spritesheet out of. If you want to create your own Clonk or import old Clonk files, modify them to create something new, follow along. 😊

1. To render a Clonk from old Clonk Files you first need to download the content files from the repository. I structured the folders in a way, so they can be used by the addon directly. If you have your own (.mesh, .act, .anim) files you should structure them as stated on the repository page.
2. Open a new Blend-File and delete the light, camera and cube, since they are not needed. (press 'A', then 'X', 'Enter', while your mouse in on the 3D Viewport)
3. For the next step, open the Render Clonk Panels, if they are not already open ('N' in the 3D Viewport)

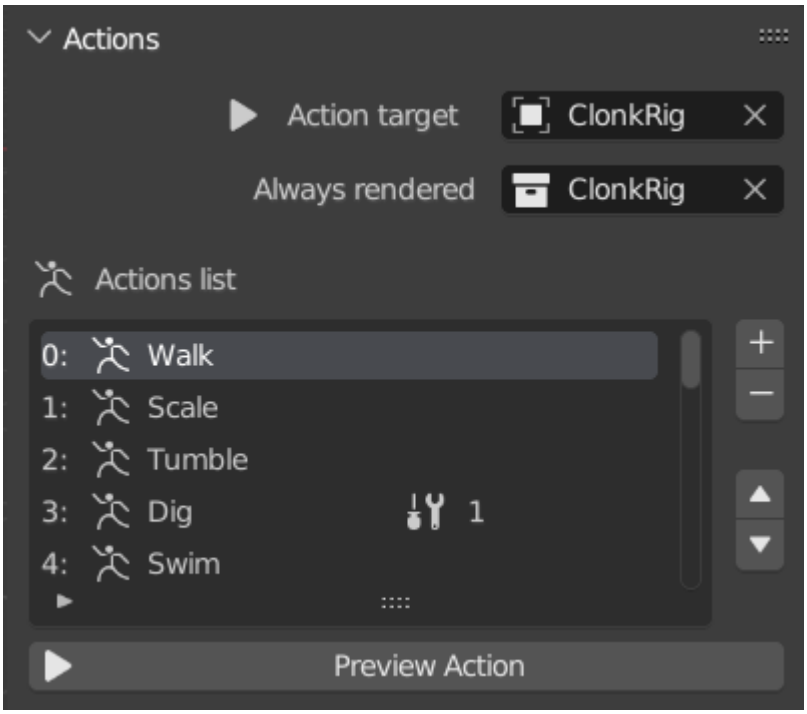


4. The addon consist of three panels. Usually, you work from top panel to bottom panel when you start from a new blend file. The first ('Render Clonk Utilities') is used for importing things and the first setup.
   Click on 'Apply Render Settings', before you import anything. This will change many render settings including the default resolution and the color handling. So your renders will look like the renders of the original clonks (or at least very similar). The full list of changes that are made can be found here: > Clonk Render Settings <
5. Next, we will import the act-file. This contains a list of all action names of a specific Clonk. Click on 'Import Action List' and load an act-file of your liking from the Clonk content files. This will load all actions, the clonk rig, and all tools that are needed by the actions.

6. Your scene should now look somewhat like this (I chose Aquaclonk.act):

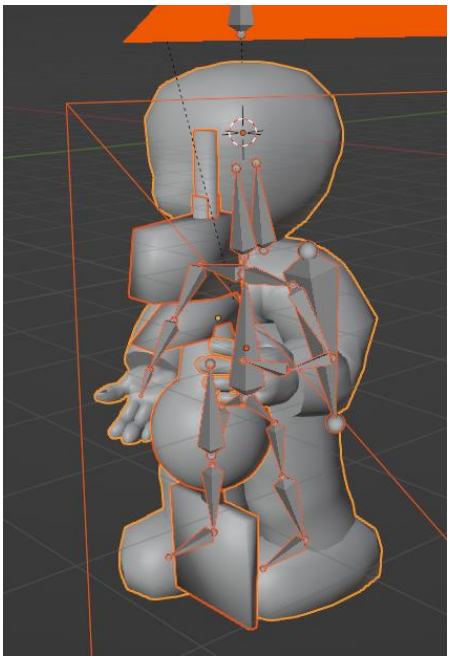


And the actions panel should be filled with actions:

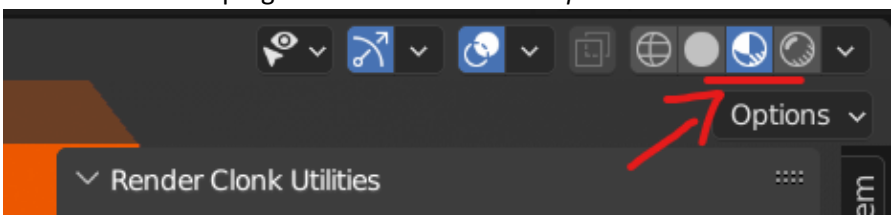


7. There is still no Clonk visible, except its rig. We need to load it separately via:

(You can load the Aquaclonk.mesh or any other Clonk mesh)



8. If you now see everything in gray like me, you have to set the viewport rendering to ‘Material Preview’ on the top right corner of the *3D Viewport*:



9. You can click on ‘Preview Action’ below the Actions list to see him walking.

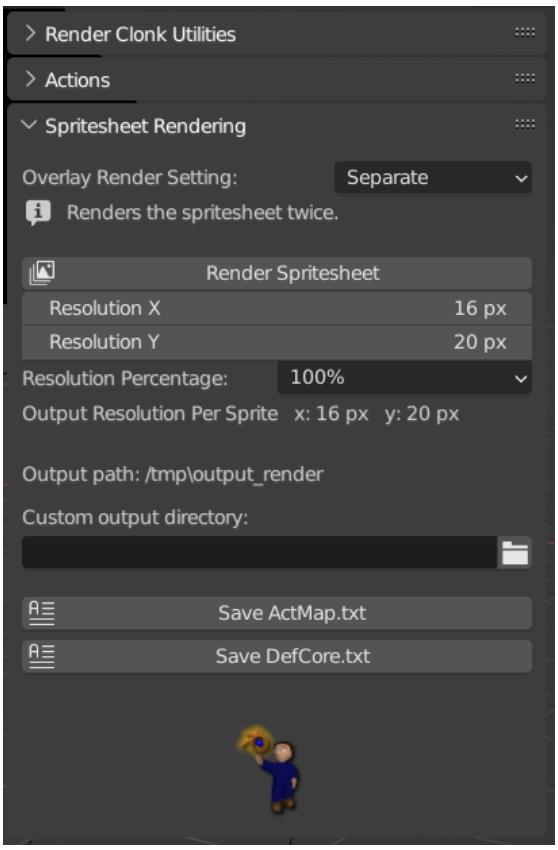
(The face of your clonk will probably look a bit different. I placed different face textures inside the clonk folder to be loaded. If you want to use them as well, look in my example files. There is a folder called textures that contains them. You don’t need to import the clonk again, you can exchange the texture later, as explained here: > Change the Clonks face texture <

And if you want to know how to set the skin tones of hands and feet like the head, look here:
> Make the skin tones of the Clonk consistent <

10. The next step will be to render the spritesheet.

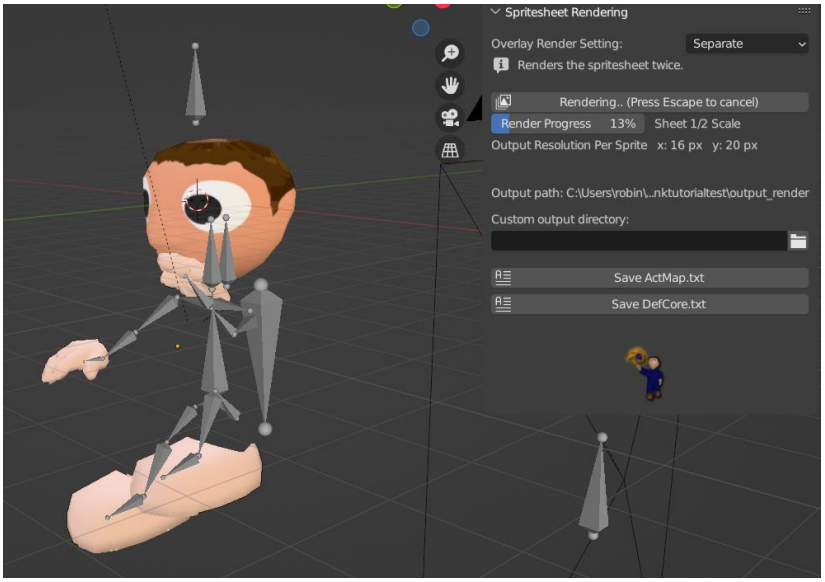
c) Rendering the Spritesheet

The clonk is now ready to be rendered. Inside the addons panel scroll down to ‘Spritesheet Rendering’. You can collapse the other tabs by clicking on the titles.

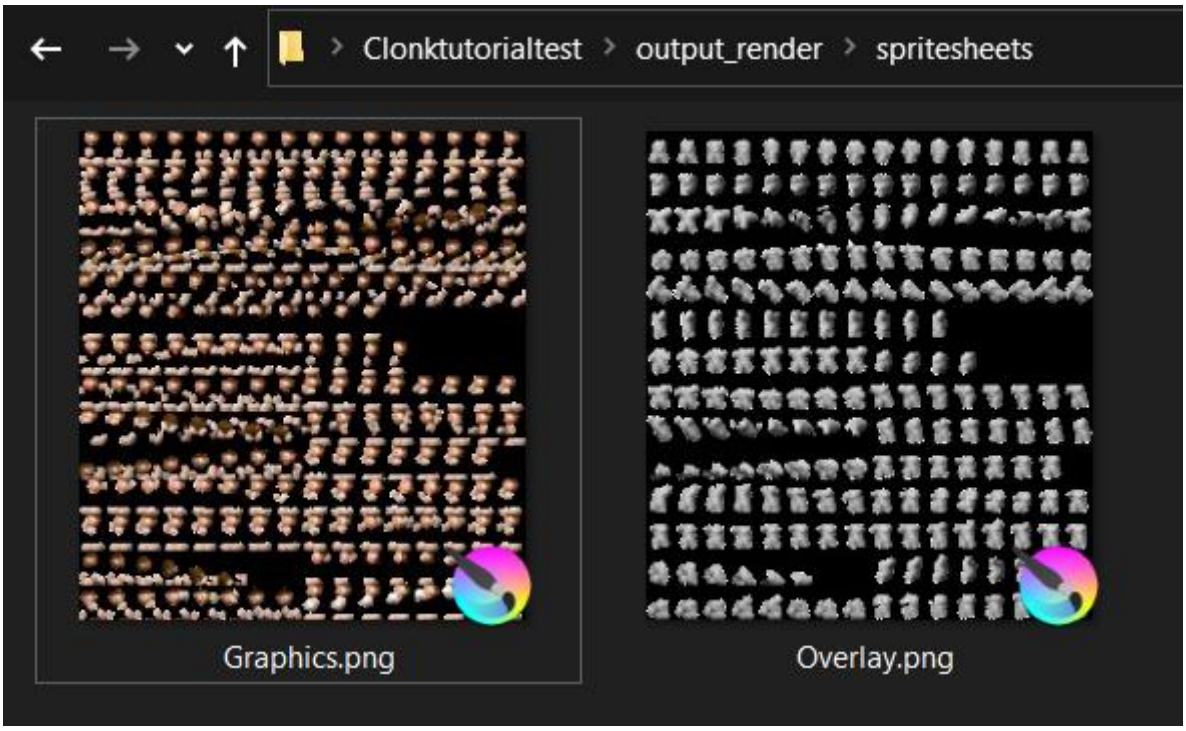


All you need to do is to click on the ‘Render Spritesheet’ button. I recommend that you save your blend-file first, since the spritesheets will be stored right next to it, in a separate folder. Otherwise, they will be stored in a temporary folder somewhere.
After you’ve done that. The path will update inside the panel, like here:

Now, click on render and watch the Clonk animate. 😊



Et voilà!

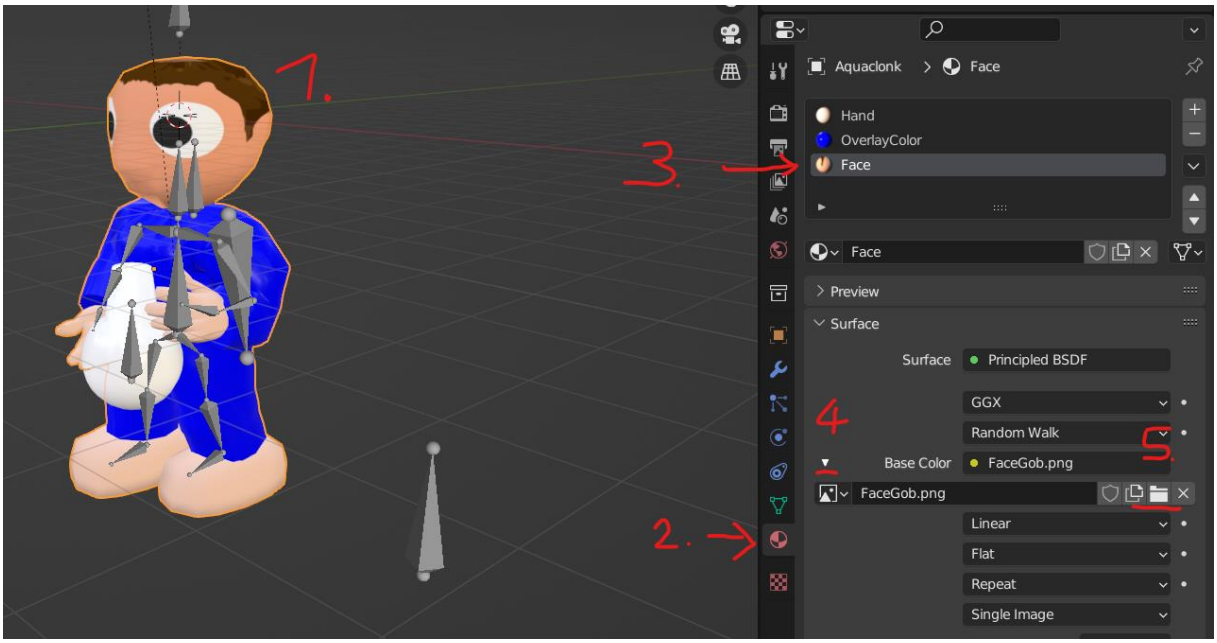


Keep in mind that you only have to import the Clonk and its animations once. All the information will be saved inside the blend-file.

2. Make it your own

a) Change the Clonks face texture

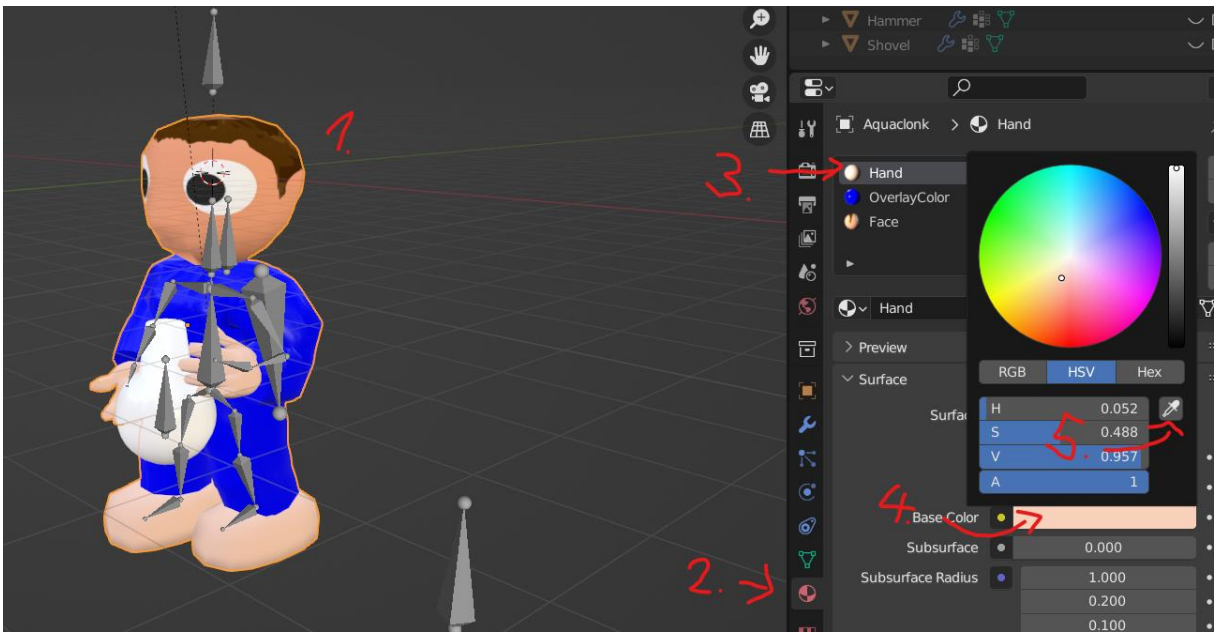
Once you've loaded a Clonk and want to change its textures you have to click on the model to select it (1.), then follow these steps:



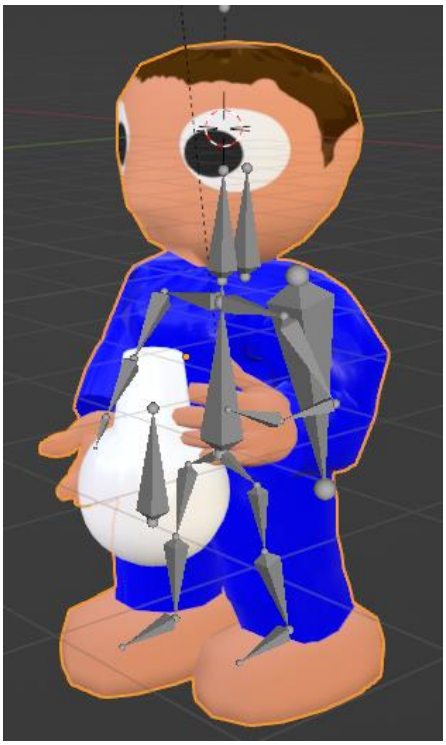
On step five, select another face texture using the filebrowser.

b) Make the skin tones of the Clonk consistent

The hands (or feet) of the Clonk will probably not be the same color as the head. Follow these steps to copy the color from the head:



Step five will enable the eye dropper tool on the color. Click somewhere on the head and it will copy the color from there.

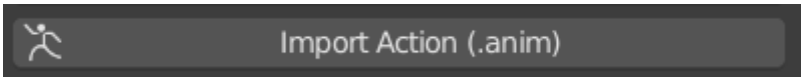


c) Include a title image

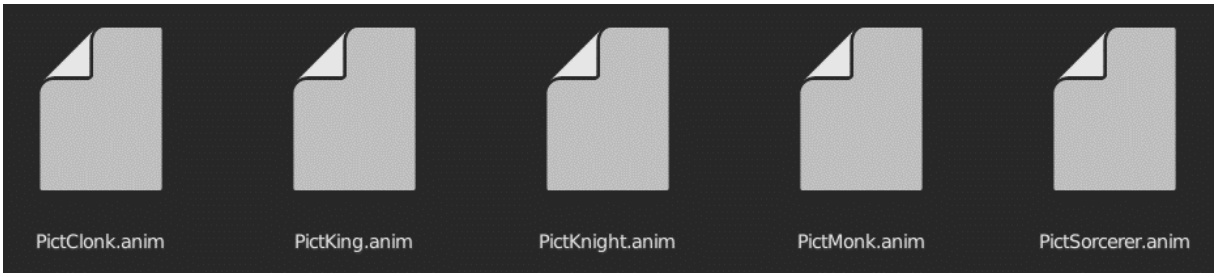
Clonk graphics need title images so they can be shown inside the inventory or the content of other objects within the game.

If you don't specify a title image the DefCore exporter will use the first sprite of the spritesheet as your title image. If you want to know how the DefCore exporter works, look here: ActMap.txt and DefCore.txt.

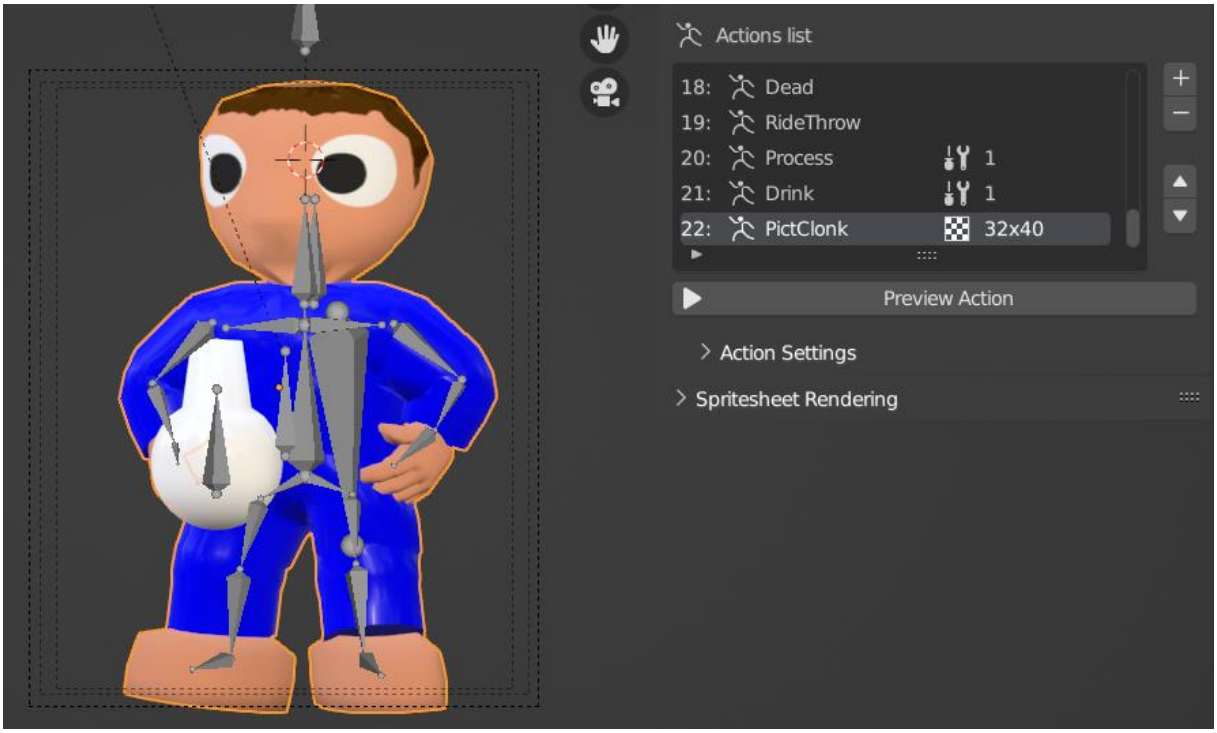
The title image is basically set like an action. It uses an animation to pose the Clonk, but it will only use one sprite that is placed onto the spritesheet.
You don't need to create your own 'Picture Pose'. The Clonk content contains pre-made actions. Click on 'Import Action' inside the Render Clonk Utilities panel:



Then select a picture action. I found these:



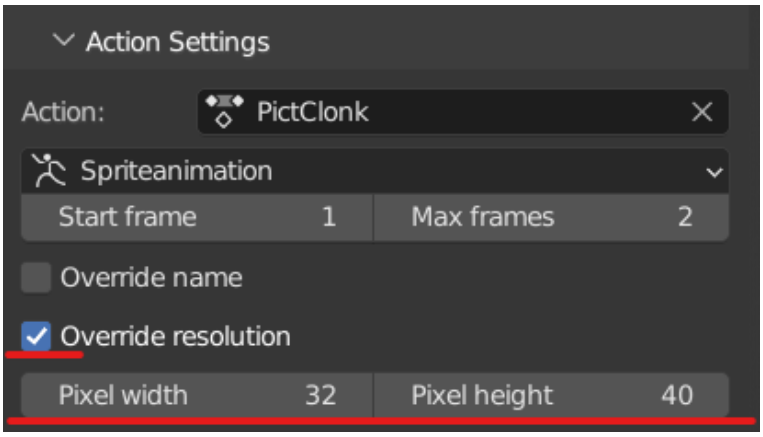
This will load the action and create a new entry inside the Actions list. So it looks like this:



(You might have to scroll all the way down in the Actions list to see it)

Notice the 32x40 next to the entry. This stands for the dimensions of the sprite. It is shown here, because it differs from the default rendering resolution set by "Apply Render Settings".

You can see this if you unfold the Action Settings below the list:



This sprite will be larger than the other sprites, since title image are usually in a higher resolution than other sprites. Which makes sense, they are usually shown larger in the game.

There are two settings, that you should make:

- Change the action from 'Spriteanimation' to 'Picture':

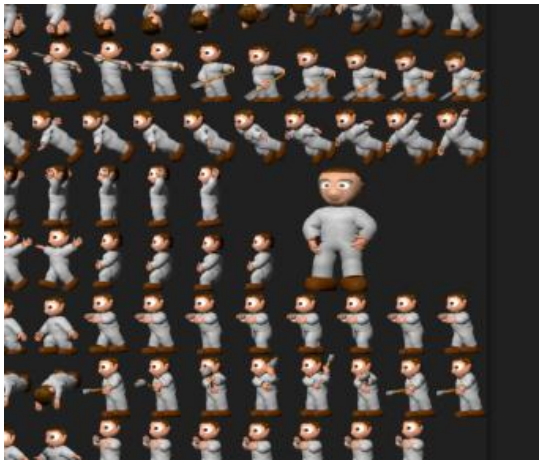
  

  This will make sure that only one sprite is being rendered *and* it will set the sprite position of this action to the picture position inside the DefCore.txt (Once you hit 'Save DefCore.txt').

- Uncheck 

  at the bottom of the Action Settings. This is a bit advanced, but it will put the sprite where it fits in the spritesheet, instead of the end. This can make the whole spritesheet image smaller.

  This picture should illustrate that:

  But you only should use this for title image actions. Not for regular actions.
  Note: There is an exception for buildings. You might want to check it there an put the title image action at the top of the action list, so it will be the first in the spritesheet. Otherwise, it might not show correctly inside the game.

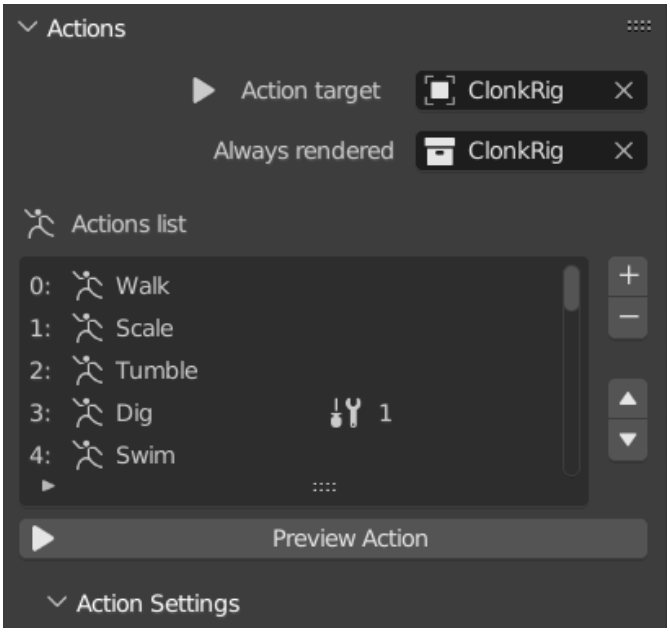
3. Details, details, details..

Upon importing a Clonk, all important settings will automatically be filled in. But in case you want to make your own Clonk, Vehicle, Animal, etc. you might want to understand how to use the addon fully.
If so, I assume you have a basic understanding of Blender, since I can't explain everything from scratch here.

Most fields of the addon have a tooltip that says what the field is for, but I'd like to go into a little bit more detail about how they function.

a) Actions Panel

The actions panel lets you control exactly how the addon will render the spritesheet. What objects are displayed, what is animated, etc.



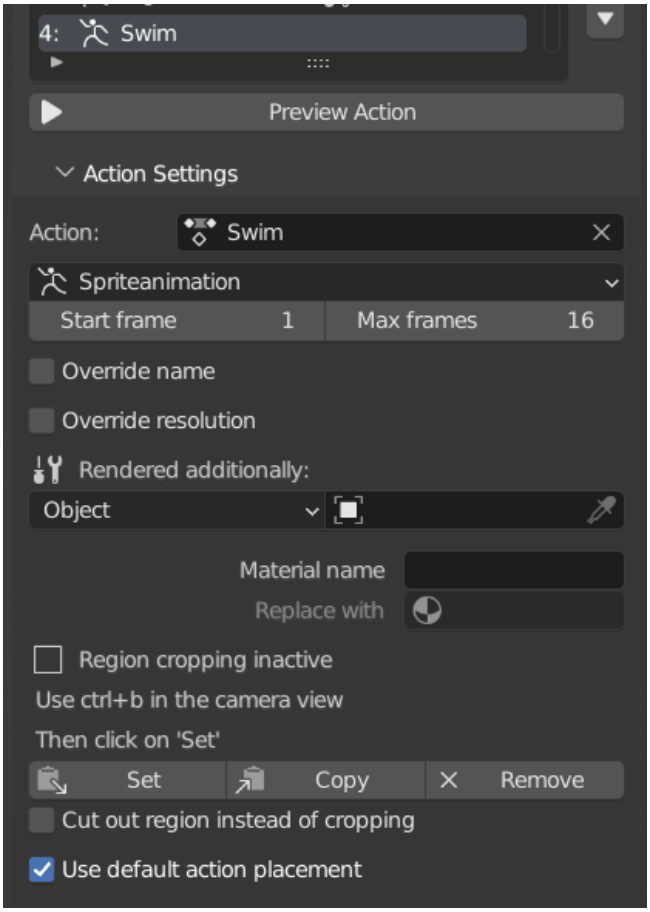
Starting from the top:

- **Action Target**: Here you can reference an object from the scene. Although it is recommended to use an armature, it is not necessary. This is the object that will 'receive' the animation data. Every action in the Actions list, will be applied one by one upon rendering to this object.
  An action in Blender terms is an animation with a name, that can be applied to almost any type of object. If you want to learn more about it, the Blender Manual has detailed information about actions themselves and the Action Editor where you can see what data is actually inside these actions.
  Note: The reason why armatures are recommended is, that you have way more control about what changes and is animated inside one action. The clonk rig for example animates the camera as well, not only the pose of the Clonk!
- **Always rendered**: This is a collection that will be visible in every sprite rendering. It usually contains a Clonk model, but no tools, since they should only be visible in some actions. You can see from the list directly, which action has a tool assigned (and how many): 

- **Actions list**: The list itself should be relatively self-explanatory. You can create new actions in the list, delete one or move them around. You can rename an action by double clicking onto its name.
- **Preview Action**: Previewing the action. It will set the frame range temporarily to match the actions length. It will also replace a material if one is set. But more on that feature later. Once you started the action, you have to click somewhere to stop it. Stopping it will reset the frame range and the material.

b) Action settings

The action settings visualize the data of the action that is currently selected inside the Actions list.



There is quite a few things that can be changed. Let's go through them one by one:

- **Action**: This references the Blender action that contains the animation data.
- **Spriteanimation**: This defines that the action has more than one frame. You can change it to 'Picture' that is used for title images. You can read more on that here: Include a title image
- **Start frame** and **Max frames**: The *beginning frame* of the animation and the *length*.
- **Override name**: Can be checked for an input box. This is useful, if you want to have two actions like 'SwordFight' and 'AxeFight', that both use the same Blender Action 'Fight'. The tool can be changed per action entry and you don't have to copy an animation, since it is basically the same movement.
- **Override resolution**: Can be checked for two input boxes containing the new resolution. This is used for sprites that need to be larger than the standard sprites. This is used in the throw spear action of the knight for instance.

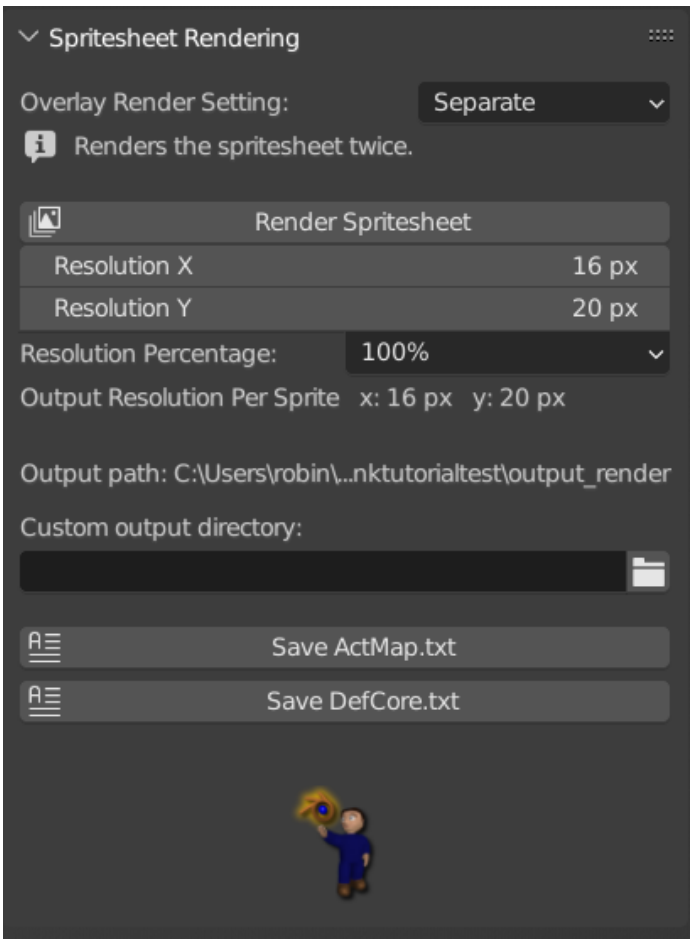
- **Rendered additionally**: This is the counterpart to 'Always rendered'. You can reference an object or a whole collection. These objects will be visible *only* for that action. It is used mainly for tools (axe, shovel, sword, bow,..) but can be used for anything. You could for example show some particles in one action. I am thinking of the mage casting a spell with flashes between his hands.
  Note: You can test if your setup works when you preview the action. The objects will be hidden/shown in the viewport as well.
- **Material name** and **Replace with**: You can exchange materials inside one action. 'Material name' should take the exact name of the Material, that is going to be replaced, for example 'Face'. 'Replace with' is another material that the other one is replaced with. It could be another face material with a different face expression for a Clonk. I used this for the example Clonk. The tumble action exchanges the normal face with a screaming face.
  Originally the feature itself was implemented for title images where Clonks get higher resolution faces.
  Note: If you preview the action, you can see if your setup works.
- **Region cropping**: This feature will crop the sprites to the rectangle that you specify. It has little use for Clonks, since it is mainly meant for buildings and their doors. You crop the part where the entrance is (that is animated). Once you export the ActMap.txt it will automatically calculate the offset of the door, so it is on the right spot in the game.
  It is used like this: Go into camera view ('Numpad 0', mouse must be inside the 3D Viewport), press ctrl+b then drag and drop a rectangle inside the camera. Then click on 'set' inside the Action settings. Notice that the rectangle will move a little bit once you click set. This is because I set them pixel perfect so they will work when the graphic is rendered in higher resolutions as this will avoid rounding errors.
  You can even cut the region instead of cropping it. This is useful if the space behind an animated region should be transparent. Hazard uses this for example for the big generator. The motor is turning (animated region) and the main graphic has a cutout so the animation wouldn't overlap with it:

  (Of course, they didn't use this addon, but I've seen this and though this feature might be useful to some people)
  If you want to recreate something like this, you need to adjust two actions. Crop the animated action and make a cutout inside the title image action.
- **Use default action placement**: This should be checked for almost all actions. Exception are title images as explained here: Include a title image. The 'default placement' relates to the placement on the spritesheet. Here is an explanation on how the placement works:
  For packing the sprites onto the sheet, it is determined which action has the most sprites and is the widest overall. This sets the maximum width of the spritesheet. Now, each spriteanimation will be placed next to the previous one on the right. If there is no space left (because one row became wider than the maximum), it will be put in a new row below it. This is done for all sprites and is *default*. The order of placement is defined by the order

inside the Actions list, so you can control it, if you wish.
Non default placement means, it will be placed after all other sprites (that use default placement) have been placed. It will look for empty space right next to the rows where it fits. If it fits nowhere, it will be put at the bottom.

c) Spritesheet rendering



- **Overlay Render Setting**: Many objects in clonk use an extra graphic as overlay, which is dyed in the player's color. The dropdown lets you decide whether you want to render an overlay graphic as well as the normal graphic ("Separate") or a Combined version, hence only one spritesheet.
  The Combined option will render everything like you can see it inside the 3D Viewport.
  The Separate mode will render in two passes:
  1. On the first pass, it will exchange every material with 'overlay' in its name with a material that is transparent and render a spritesheet called 'Graphics.png'.
  2. On the second pass, it will exchange every material that has 'overlay' in its name with a white material upon render and everything else with a material, that is transparent. The output will be called 'Overlay.png'.
- **Render Spritesheet**: Starts the rendering process.
- **Resolution X/Y**: The default rendering resolution. Will be set by 'Apply Render Settings'. Can be changed afterwards, of course.
- **Resolution Percentage**: Choose between 100%, 200% or 300%. Everything above 100% will make the rendering bigger. This value will be set inside the DefCore.txt, once 'Save

DefCore.txt' is pressed, as Scale=x. This can only be used in LegacyClonk, since it supports graphics with higher pixel density.

- **Custom output directory**: Here you can set an alternative output folder of the spritesheets and the ActMap.txt and DefCore.txt. The default is the folder right next to your blend file or a temp folder, if the blend file, wasn't saved yet.
  It is quite handy to set this to a Clonk folder (.c4d) that is unpacked. You need to set writing permissions for that folder though!

d) ActMap.txt and DefCore.txt

- **Save ActMap.txt:**
  This will write all actions and their sprite positions on the spritesheet into a text file called ActMap.txt. If you specify an output folder that already contains an AxtMap.txt, this will try to update all the values inside it. For example, if the ActMap contains an action called 'Walk' and the addon contains it as well, it will update the Facet and Length. It will even recognize actions that use the same sprites but with a different name. 'Ride' and 'RideStill' for example. Both use the same sprites but with different length of the animation.
- **Save DefCore.txt**:
  This works the same as 'Save AxtMap.txt'. It will output the information that is relevant for the DefCore from the addon and writes in into a file called DefCore.txt. If there is a DefCore.txt at the output path, the addon will update its data.
  The data that is changed is: Width, Height, Offset, Picture and Scale

4. Reference

a) Clonk Render Settings

These settings will be used when ‘Apply Render Settings’ is pressed

```
bpy.context.scene.render.engine = "CYCLES"
bpy.context.scene.cycles.device = "GPU"
bpy.context.scene.render.film_transparent = True
bpy.context.scene.cycles.pixel_filter_type = "GAUSSIAN"
bpy.context.scene.cycles.filter_width = 1.5
bpy.context.scene.display_settings.display_device = "sRGB"
bpy.context.scene.view_settings.view_transform = "Standard"
if bpy.context.scene.render.resolution_x == 1920:
     bpy.context.scene.render.resolution_x = 16
     bpy.context.scene.render.resolution_y = 20

bpy.context.scene.render.image_settings.compression = 0
bpy.context.scene.cycles.use_denoising = False
bpy.context.scene.cycles.caustics_reflective = False
bpy.context.scene.cycles.caustics_refractive = False
bpy.context.scene.cycles.max_bounces = 1
bpy.context.scene.cycles.diffuse_bounces = 0
bpy.context.scene.cycles.glossy_bounces = 1
bpy.context.scene.render.use_persistent_data = True
bpy.context.scene.cycles.samples = 1024
bpy.context.scene.render.fps = 15
```