This plugin is intended to be used to import and export Star Wars Galaxies MGN assets into and out of Blender v2.79b. It can also be used to Export custom assets (new stuff!) from Blender into the SWG MGN format.

A content warning:

I recommend that you do not use this plugin with copyrighted material from other games. As was the case with Poem Studio's "Apeiron" and its reboot of the KOTOR game, you're likely to get into legal trouble with Lucas film, or any other license holder, for using assets from other games in SWG. Especially if your SWG server is public. SWG is in and of itself on shaky legal grounds with respect to public servers, and almost certainly illegal out right for anyone making money from it.

I did not write this plugin, so people could bring illegal content into SWG. Rather, I built it, so the existing content could be modified and updated (like belts that look like hulahoops). I also hope that creative folk will start making new items for SWG using this plugin. However, if you use the plugin to bring copyrighted and unlicensed IP into SWG, you do so at your own risk. You have been warned.

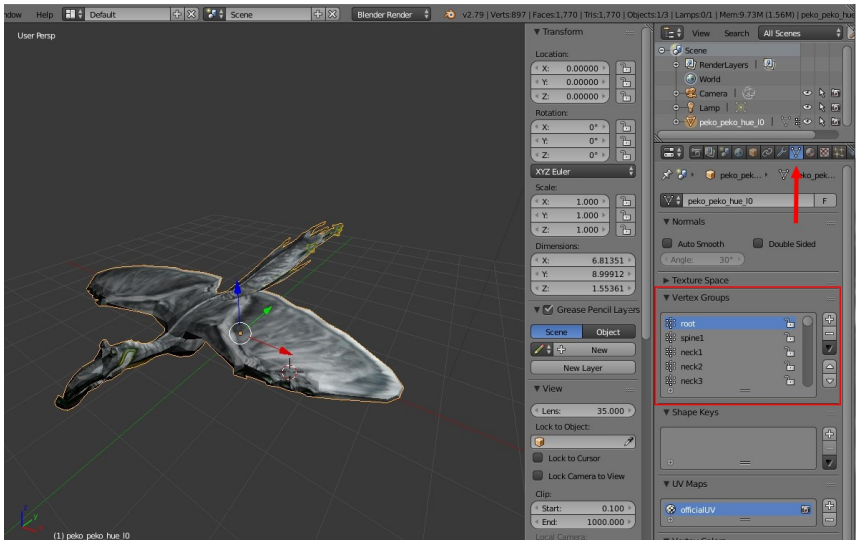
****

Importer notes:

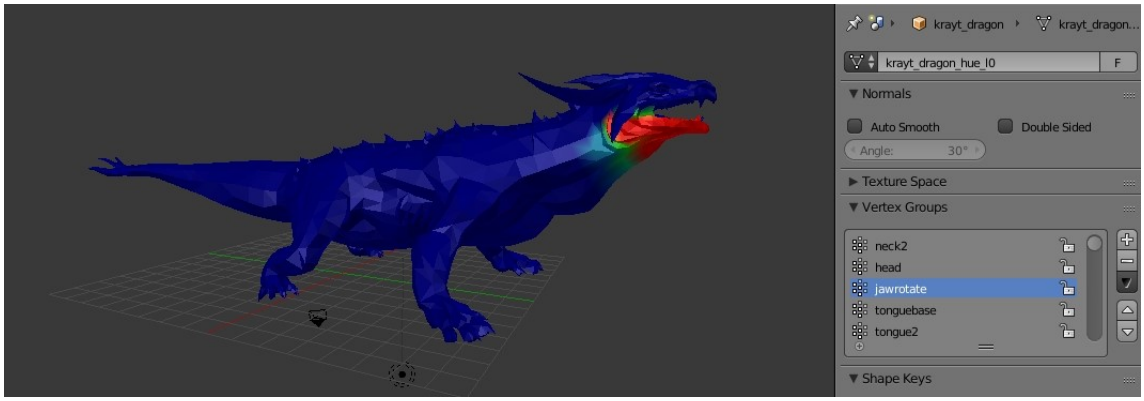
This plugin will properly import a Star Wars Galaxies MGN file into blender. Items imported include the base mesh, UV, Shader Name, Bone names, Vertex weights, Blends, occlusion zones, and skeleton name as follows:

The mesh is obviously the active imported object.

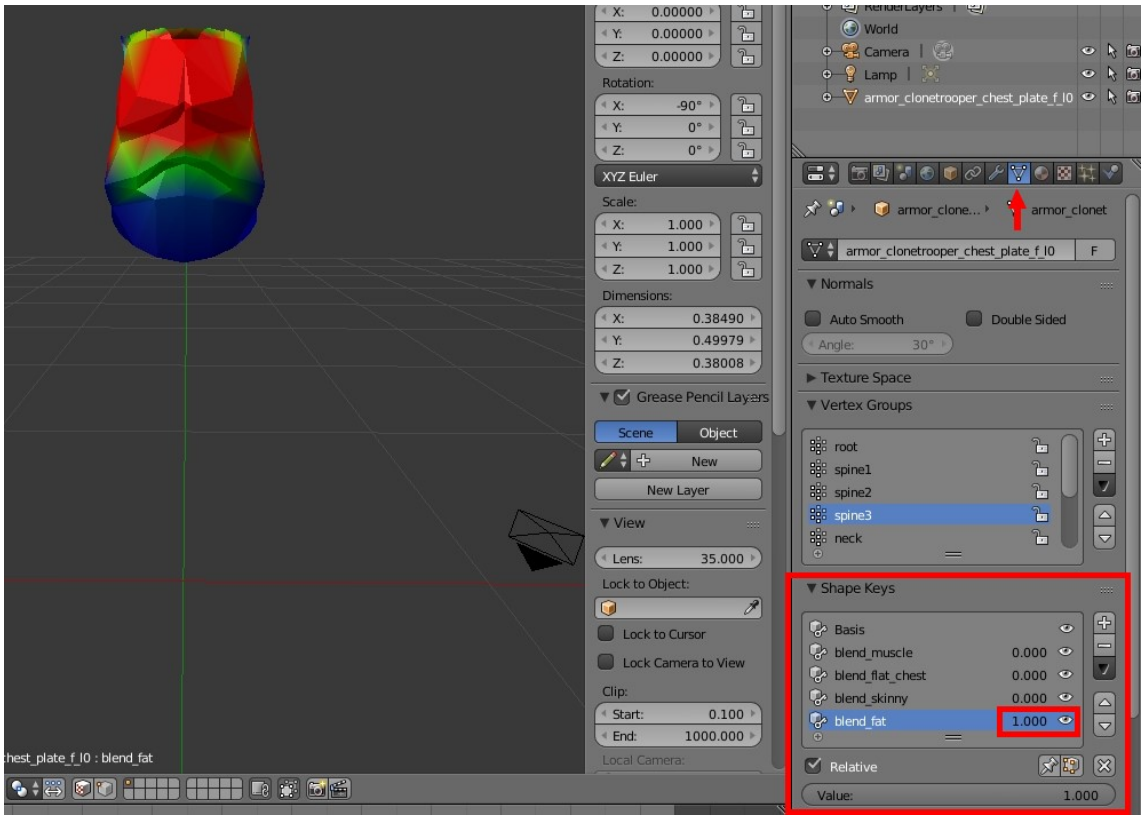
Bone names are Imported as vertex groups.



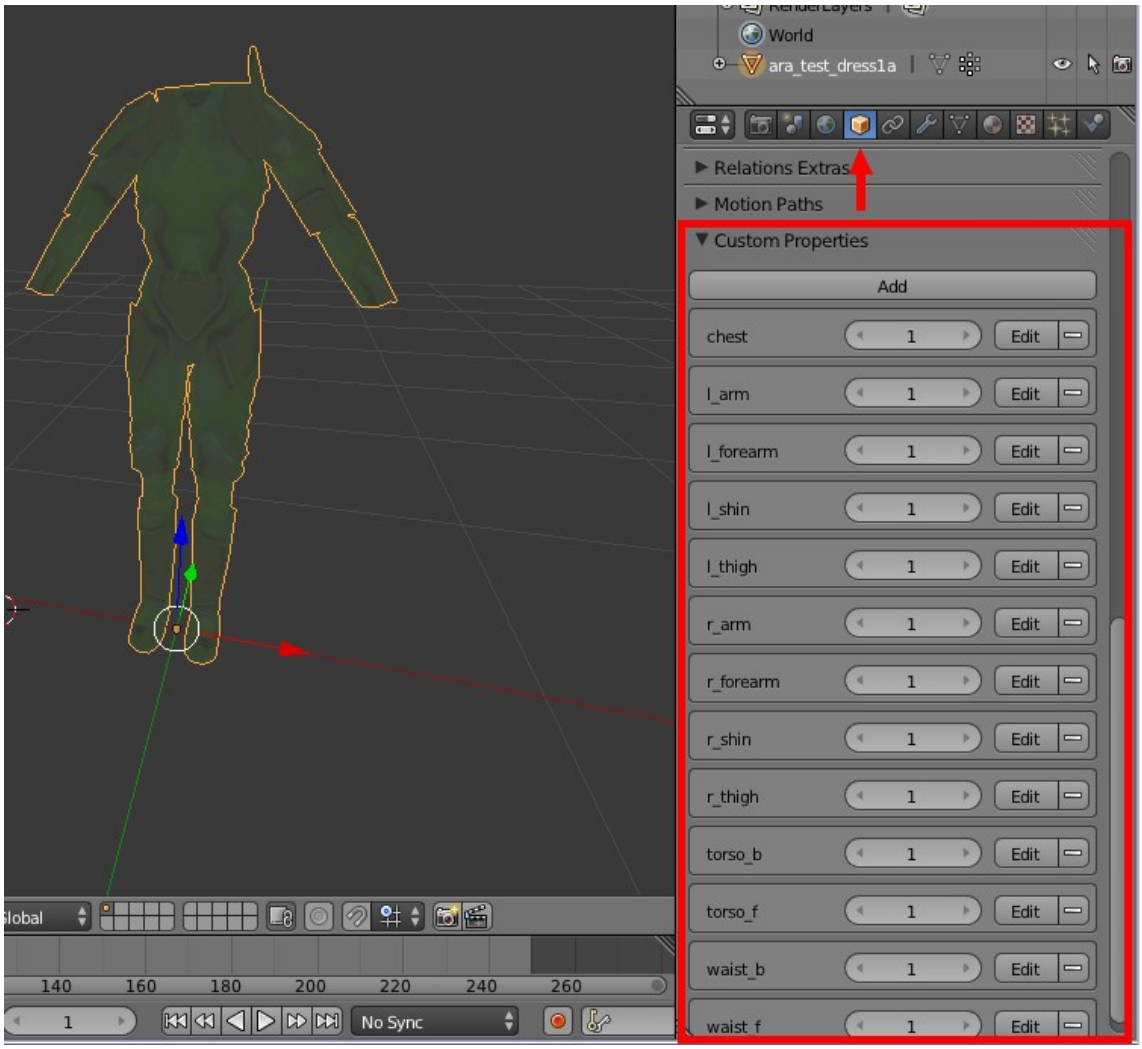
Vertex weights are imported and assigned relative to the vertex groups they belong to.



Blends are imported as shape keys.



Occlusions are imported as custom properties.

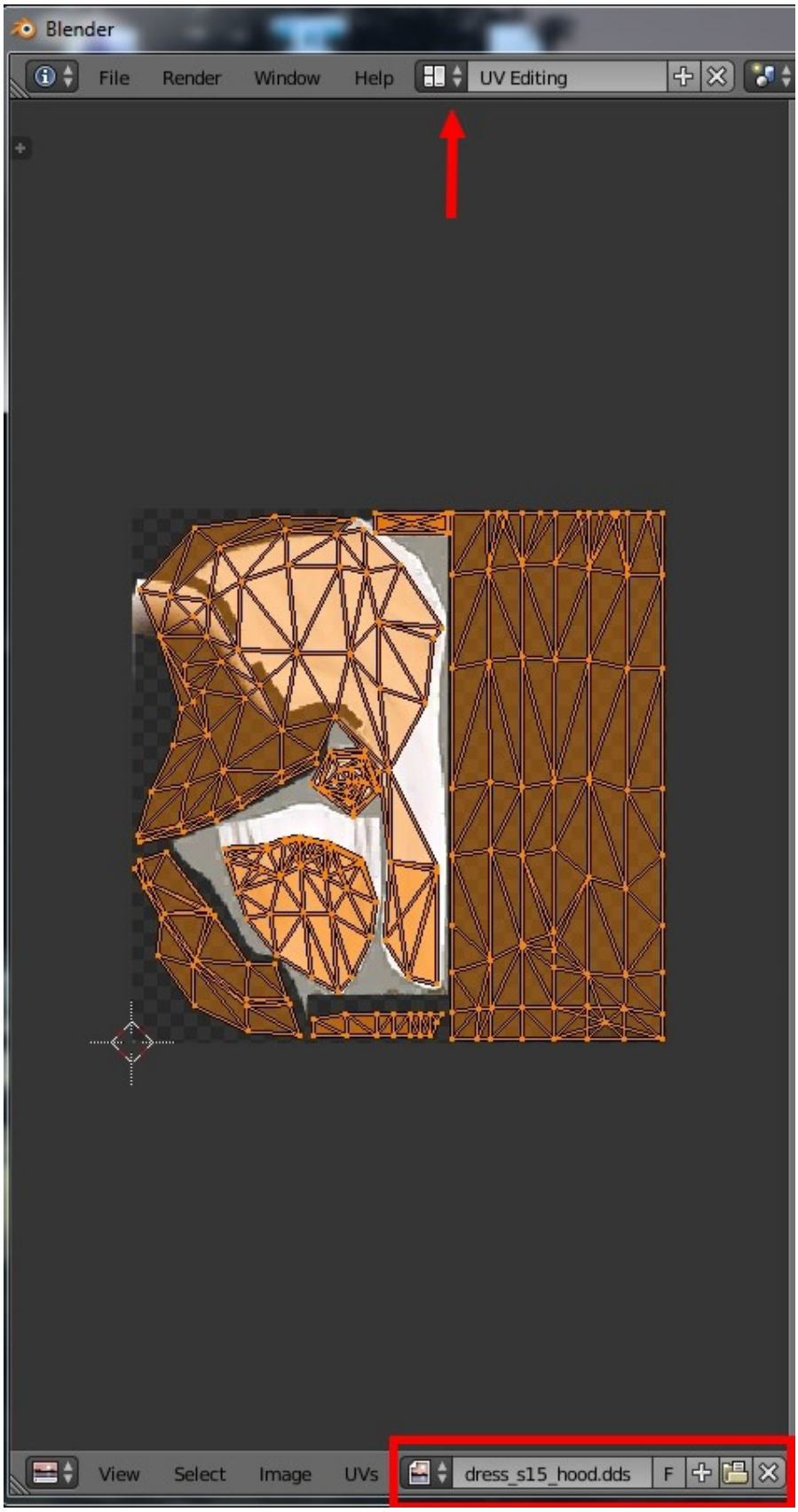


Skeleton name is imported as a custom property.

Shader name is imported as a material, in cases where there are multiple shaders, each shader is added as a new material. Also, each polygon in the mesh is properly assigned to each material. However, each created material while having the proper shader name, will still only be a default blank material, without textures, shading, etc… You can, however, load any textures associated with the SWG shader into blender, and they will map properly onto the mesh. But you have to do this manually, the importer will not do this for you.

UV's are imported for each shader, and stored in a single UV file within blender. Again, the UV's are assigned properly to each Poly and material that gets created. This allows you to import any and all textures from the SWG

shader files into blender, and they will map properly. Please be aware that SWG UV's are written to the MGN files Upside-Down. Meaning they have to be flipped upright on import for them to work properly in blender.



Export Notes:

This plugin will export a single object from blender into the MGN file format for SWG. Items exported are the mesh, UV, Shader names, Bone names, bone weights, Blends, Occlusions and skeleton name.

Each item works the same as has already been described above for the importer. This exporter will fail if multiple objects are selected for export. Or rather, it will probably fail, I have not added any logic to catch errors, or prevent you from exporting multiple objects at once. But it should fail pretty badly if you do.

The exporter will also flip the UV Upside down (mirror on the Y axis), on export, so you don't need to manually flip the UV.

Materials get written to the PDST chunks in the order in which they appear in blender. I would not change this order for imported MGN's, and for custom items, if you find the materials and shaders getting mixed up in the client, I'd adjust the listing order to compensate. This shouldn't be a problem, but it has on occasion been a bit fickle.

_____

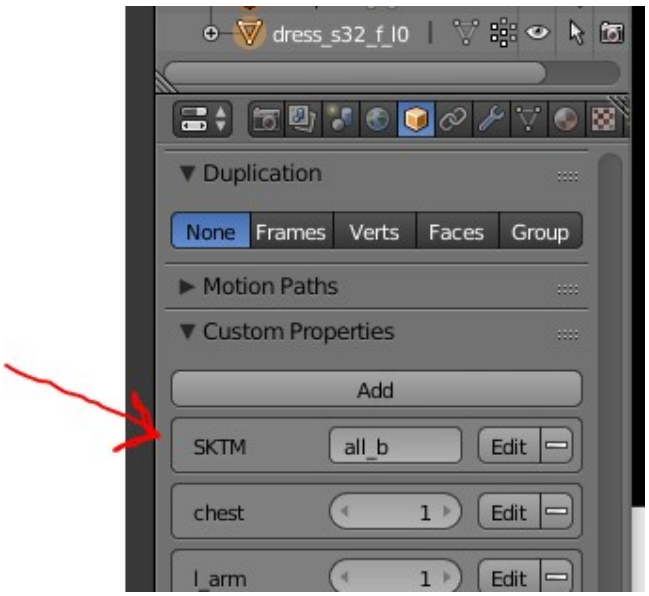
Concerning the export dialogue Options:

Checking the "Calculate Tangents" option will tell the exporter to recalculate the tangential normals for each vertex. In some cases, this HAS to be done anyway in order to have the data to start with. I suggest you leave it checked.

the Skeleton option is a text box that is intended to contain the name of the skeleton to be used for the MGN. The vertex groups (bone names) you use in your mesh should correspond to the bones in this skeleton file.

If you create a new original mesh/object, you'll first need to choose a skeleton file that your mesh should use. From that skeleton file, you'll want to use the bone names in the file for your vertex group names in Blender. Then you can assign vertex weights as necessary. When finished, make sure that the skeleton file name is typed into the export option box before export. Also, you'll need to omit the file extension, example: most clothing uses the all_b.skt file, you'd enter "all_b" into the box.

If you import an existing MGN, the vertex groups will be named properly from the start. The skeleton file to be used will also be added as a custom property to the mesh. In this case this option in the export menu is unused, it will instead use the custom property if it exists.



That last bit is important, instead of using the export option, you can add a custom property named "SKTM" and give it the skeleton file name without the extension, example: "all_b". The custom property will override anything typed into the export option.

The "Include OZN & ZTO?" option is for exporting the occlusion data. To understand what this is doing you probably need some additional information on how it works.

The SWG client has a list of zones that can be made invisible for humanoid type objects. Most creatures do not use occlusions, and any extra layers of clothing or items are made to fit exactly without clipping. For humanoids that can use extra layers of clothing and items, SWG uses occlusions to avoid clipping with lower level layers. So using a human as an example, it loads with a default skin as layer 1. If you make a shirt for the human to wear, the shirt will occupy layer 2, and without any occlusions the layer 1 body can clip through the shirt during movement in some circumstances. To avoid this clipping, you can use occlusions, which will make segments of the layer 1 skin invisible. a long sleeved shirt will occlude the chest, torso_f, torso_b, L_arm, R_arm, and maybe the formarms, and maybe even the waist zones…

It's important to understand that you are not occluding zones on the object you're working with, but rather that you are occluding zones on the base skin mesh you want to make invisible.

So, to include occlusions as part of the export, I did so by making each exclusion zone a custom property for the blender mesh/object. The easiest way to see this in action is to import an existing clothing item, dresses or robes are probably the best examples to get to know the system. Then in blender, look at the custom properties, and you'll see a listing for every occlusion zone that clothing item has been set to use. For the properties: 1 = occluded (invisible), 0 = not occluded. All zones import as occluded "1" by default, so you'll want to make sure that you've switched the zones you want to see to "0".

If you export your clothing item, and load up the SWG client, and don't see a body part you were expecting to see, example: hands, or feet, or face, etc…, then come back to blender and set those zones to zero. Then make sure the checkbox for "Include OZN & ZTO?" is checked on export.

Last, the "Use OITL?" checkbox. The MGN format has in it the ability to occlude not only by zone, but by polygon as well. However, this is very poorly understood at present. The OITL chunk contains an occlusion index, as well as the 3 coordinates for each triangle polygon. While I have included this option, it sets all occlusion indexes to zero in addition to enumerating each polygon. So it doesn't really do anything yet. We still have to figure out how the indexing works, and if this works by itself, or is to be used with the OZC chunk. Again, very poorly understood. I would not use this option until we figure out how to use it properly. It's only included because I figured out how to build the Chunk should we need it later.

****

Some additional function information.

Blends / Shape keys:

Blends are the basic deformations of the base mesh that define how the object deforms along with the "body shape" sliders within the SWG client. There are 4 main Blends: flat_chest, Skinny, Fat, and Muscle. There are two other blends I've found for Chin size. The base mesh, is also the Basis shape key. Search google, research, and learn for yourself how to properly use and save shape keys within blender.

However, some things you won't find on google. You can import the basic human / species skins into blender, and the shape keys will import with them. You can then use those to see how the basic skins deform with the body sliders within the SWG client. You could then copy each fully deformed shape key as a new mesh, and use that mesh to size your custom object appropriately. There may even be other plugins to help you deform your object to match without dragging each vertex one at a time. *shrug*

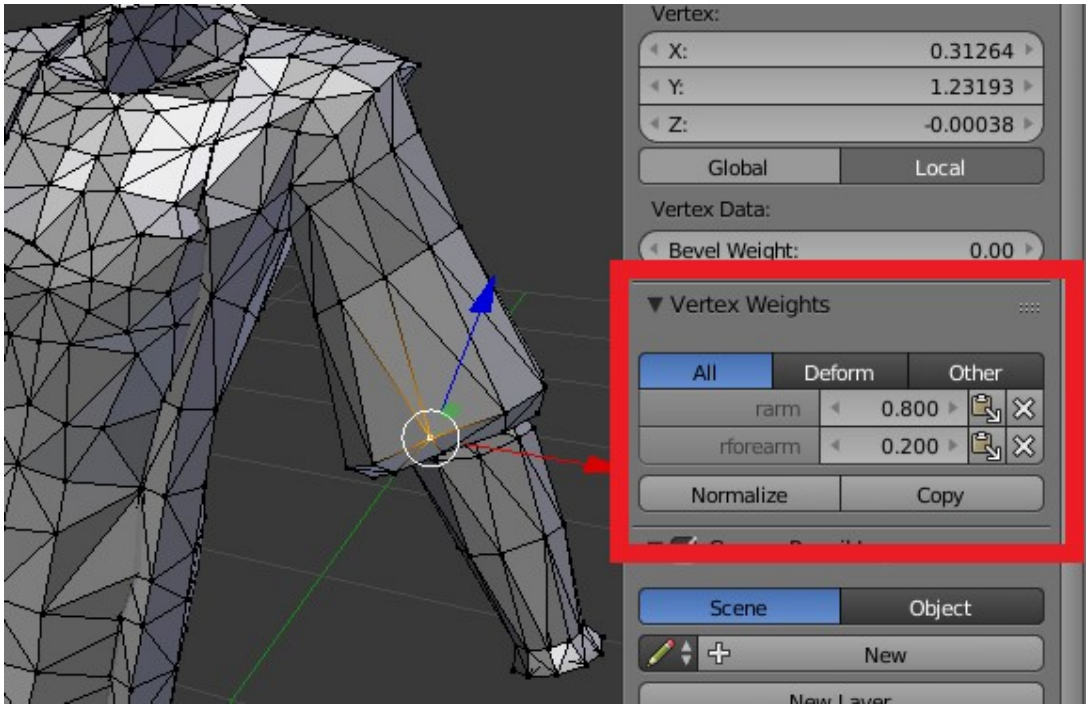
Once the shapes have been deformed, and the keys saved, then the export will calc the delta's and your item should size properly within the client.

Vertex Weights and Vertex Groups:

MGN files are Animated Meshes, meaning that not only do they deform as necessary with body shapes, but they also move along with the bones. This movement is tied to the Vertex groups and the weighting of each vertex in each group. I'd like to explain a little about that to help you out.

If you have pants, it's going to use the following bones, usually: spine1, spine2, root, l&r thigh, l&r shin. The most basic weighting here would be to select all the vertices that would use say, the left thigh, and assign them to the left thigh vertex group. Same with the right thigh, and each shin. Everything above the thighs should be assigned to the root (hips) group. Spine 1 and 2 are probably irrelevant.

At this point the pants should move with the bones when you walk, and you shouldn't see many issues. However, there might be some extreme stretching in the Knee and Hip area's. In these cases you might want to share vertices between groups to lessen that stretch. So assign the vertices at the bottom of the thigh group to the shin group, and the very top vertices of the shin group to the thigh group. When you do this you'll need to normalize their weights, as a vertex cannot have a weight of 1 in each group, it would have to be .5 in each group. Or .333 if it's 3 groups, or .25 if it's 4, etc…



Every vertex has to belong to a group, and every vertex cannot have a total weight greater than 1. I mean, you can assign them a higher weight, but the SWG client will balk. If you load your client and see wild deformations, then you probably have your weights wrong. Try to locate one of the badly placed vertices and check it's weight for each group it's part of. Also check which groups are assigned, it wouldn't make sense to have it in the shin group and root group.

Flowing clothing like robes, dresses, dusters… While they use the exact same process, they get very complicated cause any given vertex might interact with several different bones, and it might not be evenly distributed (i.e. 1, .5, .333, .25, etc..). you might actually want it to favor the thigh but also move with the twist of the spine too, in that case you can weight it in any combo that adds to 1 total for the vertex… again, it gets very complicated, but the process and restraints are still the same.