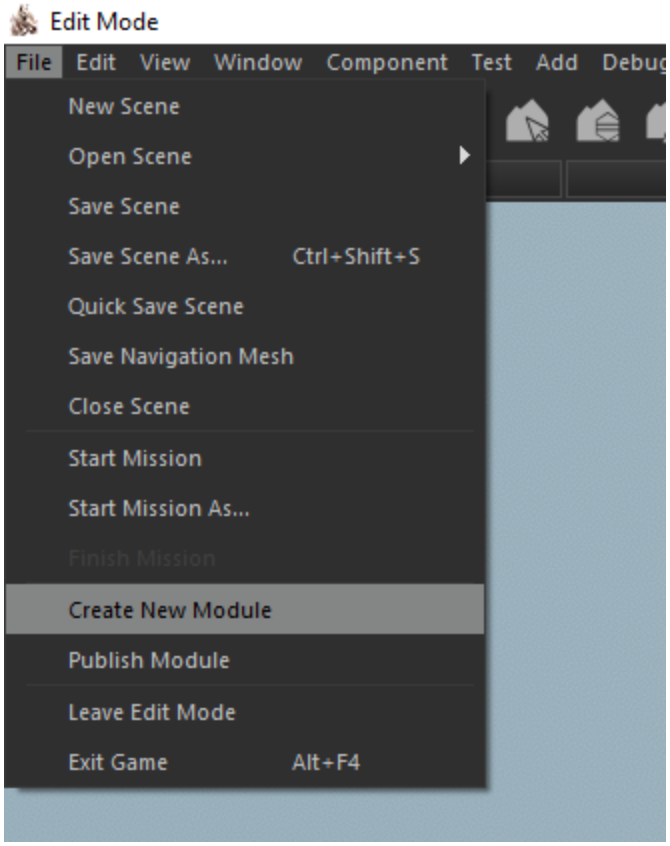
Bannerlord 3D Asset Workflow

Quick Guide to Importing Assets

1. You cannot make changes to Native modules, you must create your own, so add a folder in Mount & Blade II Bannerlord\Modules. Within your new module you can set up the basic module folders:

You simply need to add these empty folders, some are necessary for importing assets, some allow for future expansion of the mod.

Alternatively you can now create a module with a simple click of the button in the editor:



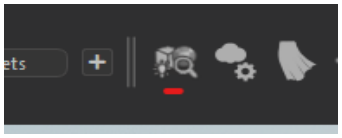
2. Next step is to add a SubModule .xml into your module folder, here is a template. Relevant information:

- `Name` This value is what will show up as your module name in the launcher.
- `ID` This value **must** be set to the name of your module folder.

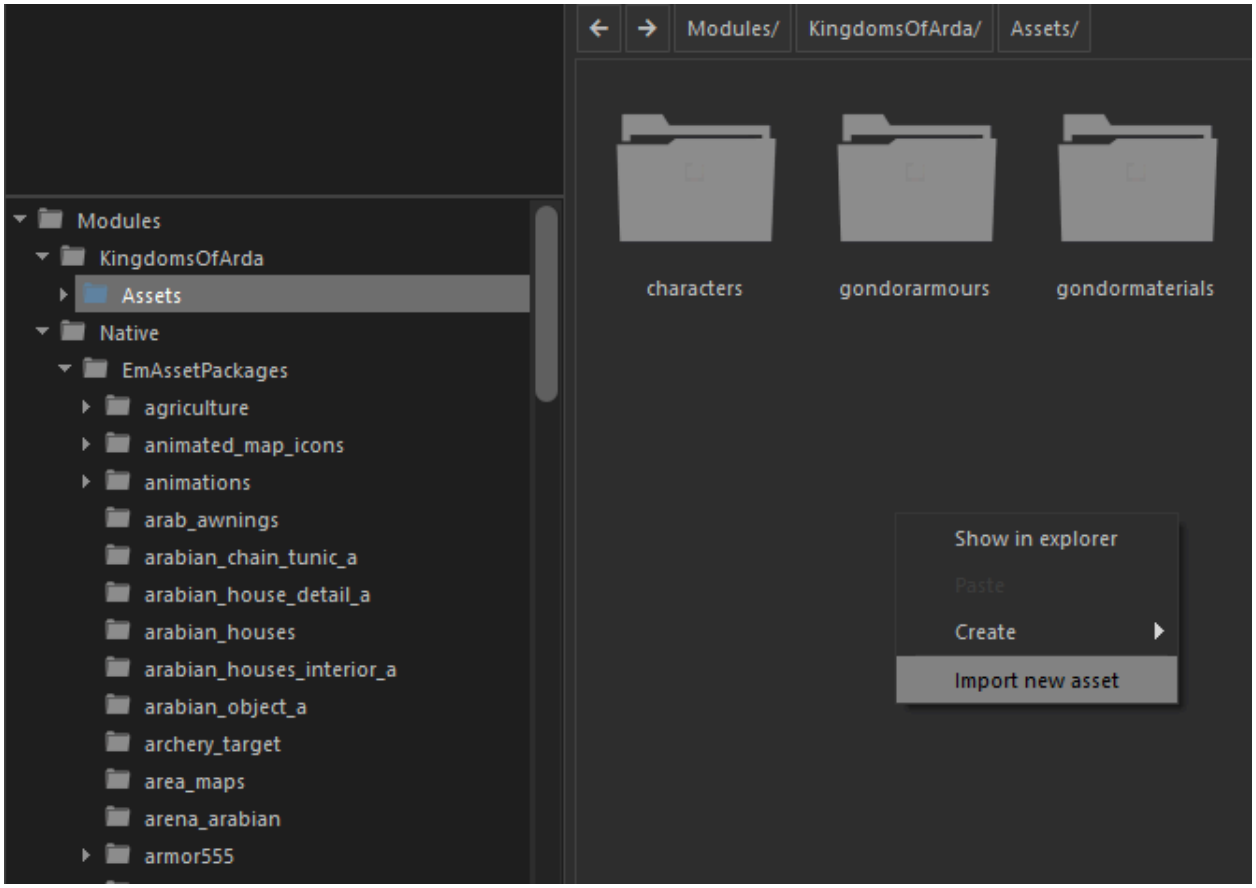
The SubModule file also has structure for you to add items and npcs into your module through `ModuleData`.

3. Launch the Modding Kit and make sure to enable your module in the launcher.

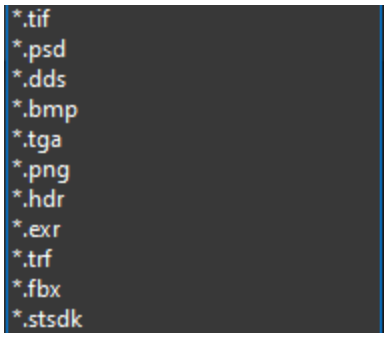
4. Using Ctrl + E on the Singleplayer main menu, you will enter Edit Mode. Select the Resource Browser icon on the top of the screen.



5. Within the Resource Browser enter your module on the left and open the Assets folder inside. You can set up further folder structures within this, for organization and controlling how .tpacs are generated. Right click anywhere to import assets.



Accepted formats to import:



Meshes

es

Polycount

Note, poly only ever refers to triangles, never faces.

An analysis of the poly counts within Bannerlord, made to give modders an idea of what their goals should be when making their own models.

Asset Type	Min	Average	Max	Notes
HeadArmor	544	1893	8362	*(some special items more(outliers)empire_crown has 31k polys)Don't base any normal helm on this*
BodyArmor	2357	8808	19844	
Cape	300	2556	7296	
HandArmor	744	2983	6844	
LegArmor	384	1297	3292	
Shield	336	2390	12276	*Most shields can definitely afford the lower end of triangles*
Bow	1	2	3	
Quiver	1	2	3	
Arrow	1	2	3	
Sword Blade	94	477	1111	
Sword Guard	288	1154	2600	
Sword Handle	80	713	2284	
Sword Pommel	224	747	2284	
More weapon parts.	1	2	3	
Character Model	1	2	3	
Hair Style	1	2	3	
Mount	1	2	3	
Mount Armour	1	2	3	
Campaign Map Rock	40	300	1122	Wide Range depending on type. Low end has no LODS, high end had up to LOD3

Here is a link to the above data in a spreadsheet.
This doesn't cover the poly count of scene assets, as their variety is very large and could be enough data for its own table.

Rigging

Skeletons for reference and rigging, human, horse, etc.

Clean vertices/normalize weights expand

4 bone rigging - verify

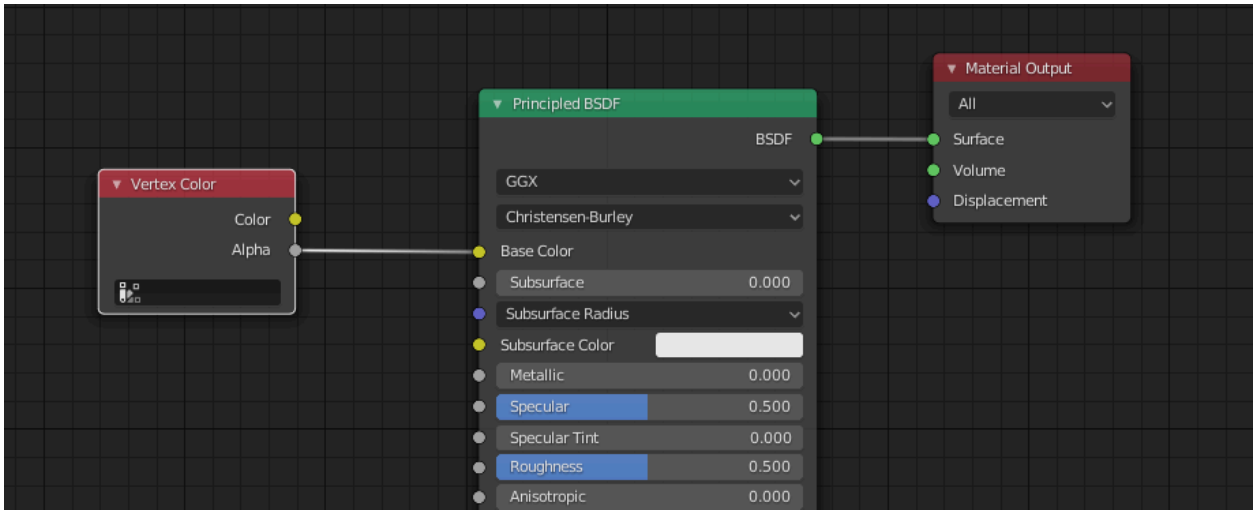
After

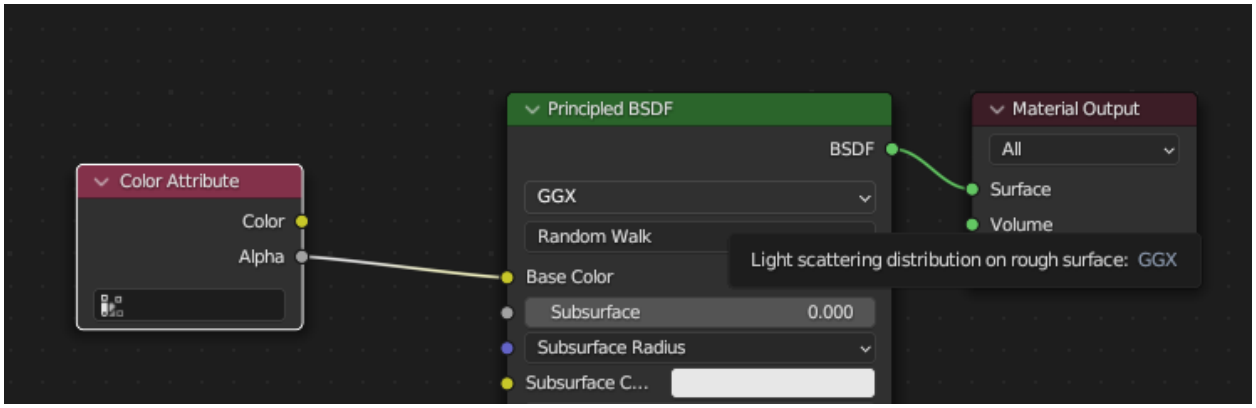
Cloth Physics

To tell Bannerlord which parts of the object is cloth, vertex alphas need to be painted onto meshes, this must be done within a 3D modeling program where white (value of 1) means the mesh is entirely affected by cloth physics and black (value of 0) means that mesh is unaffected by cloth physics. A value between 1 and 0 will be partially affected by cloth physics based on how close to 1 or 0 it is.

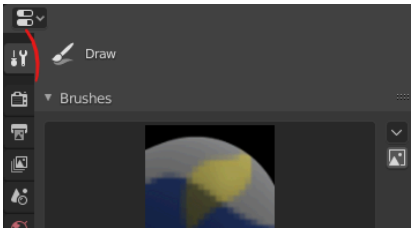
Note: Painting a black/white vertex alpha is different from painting black/white vertex colour.

Blender users (Updated for 2.8.3):
Blender has the ability to paint vertex alphas, but it is not noob friendly. Firstly the material of the mesh needs to be opened in the node editor. A “Vertex Color” or “Color Attribute”, depending on Blender version, node needs to be added, which is found under ‘Add’ then ‘Input’. Attach the Alpha output into the Base Colour input for the material.





(“Color Attribute” on newer versions of Blender)



Changing your Shading to Material Preview will show the vertex alpha on your mesh.

To edit the vertex alpha, go into Vertex Paint mode on the mesh, under Tool Settings change the Blend mode to Erase Alpha to paint black or Add Alpha to paint white.

Suggestion: Under advanced turn off Front Faces Only, this helps quickly painting in large regions that you want to be black or white.

Other Users:
Other artists using different programs will have to provide relevant information.

Levels of Detail

After rigging we will need 6 levels of detail for each mesh to optimize performance. If LOD2 is omitted, the engine will continue to use a sub-optimal LOD1 at LOD2 distances. You can make lods manually, or use automatic generation. Taleworlds uses Simplygon, available here: https://www.simplygon.com/
Here is a list of the LODs and target triangle percentages:

Level of Detail	**% of Triangles**	**Distance Active (Meters) Environment Assets**	**Distance Active (Meters) Character Assets**
LoD 0	100%	15	15
LoD 1	80%	22.5	22.5
LoD 2	50%	30	30
LoD 3	20%	50	48
LoD 4	10%	70	66
LoD 5	5%	130	106
LoD 6 (rare)	?	?	166

Distance needs confirmation by Taleworlds, they state LOD distances are “15, 22.5, 30, 50, 70, 130, 210 meters.“ yet I have only seen examples of 6 LODs, not 7.

When exporting your .fbx file, you need to include the main mesh and all of the LoDs with their names ending in .lod# as they are packed together for use by TW’s engine.
For example:

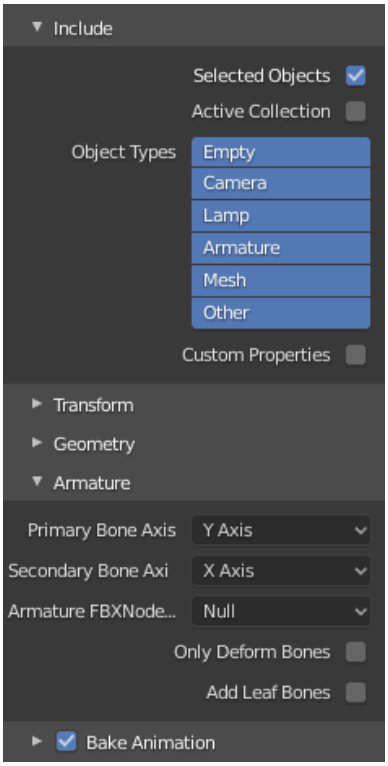
This is what should be in the gondor plate armour .fbx file, where the main model is first, and the following LoDs are named as such.
It's important to maintain the silhouette for the LoDs, if the model will be completely decimated, it's okay to have less than 5 levels. For example a sword blade that has 16 triangles on LoD 2, the other 3 are not necessary.

Exporting/Importing

human_skeleton_notused

1. How to package tpacs efficiently, same concept as texture space/materials?

Blender Users:
When exporting meshes, two options need to be changed from default:



Make sure **Selected Objects** under Include is *Enabled.*

Add Leaf Bones under Armature needs to be *Disabled.* This is only relevant for rigged meshes, but will not hurt if done for other types of meshes.

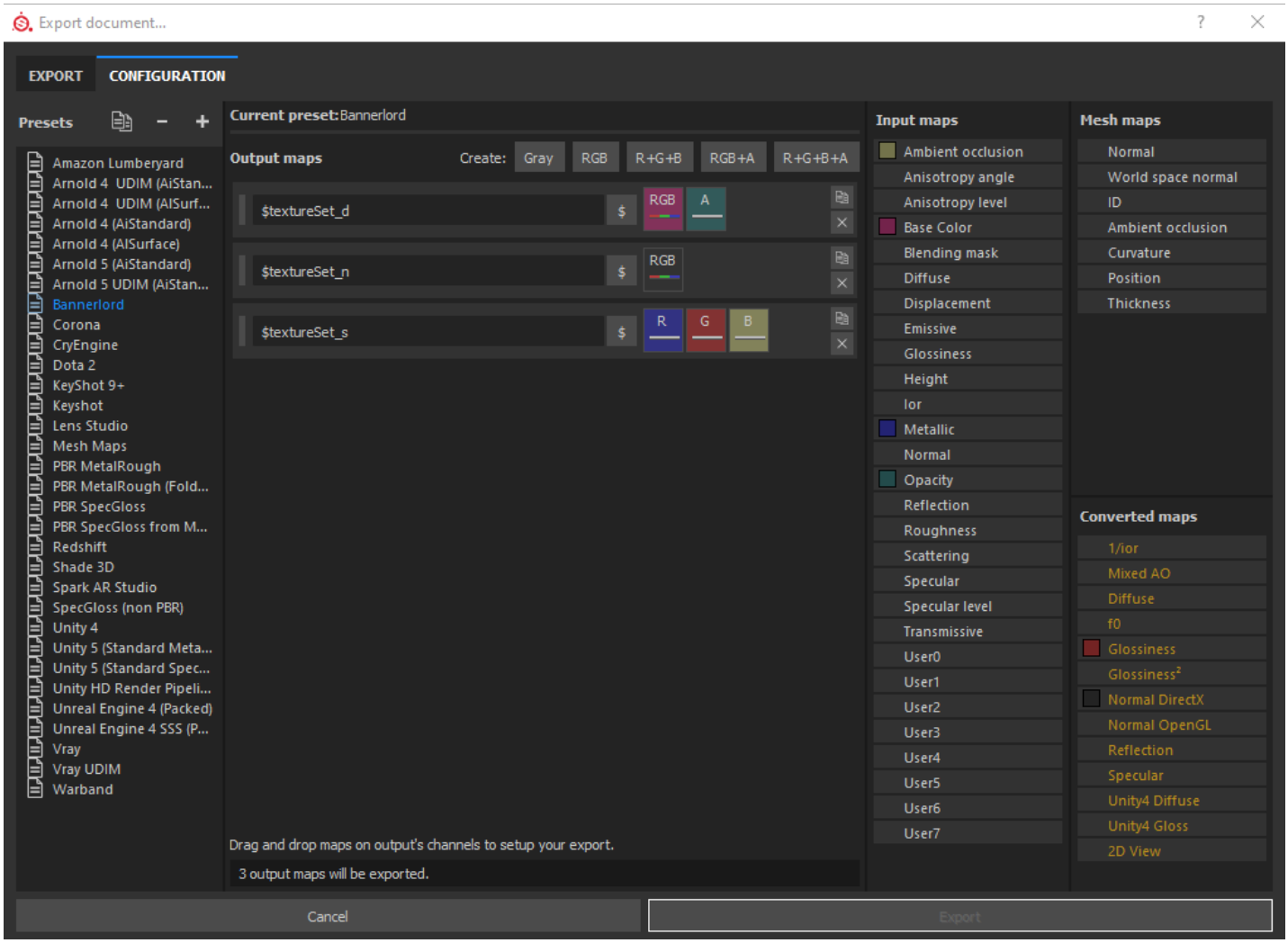
Other Users:
Unsure if the default export settings are compatible with BL.

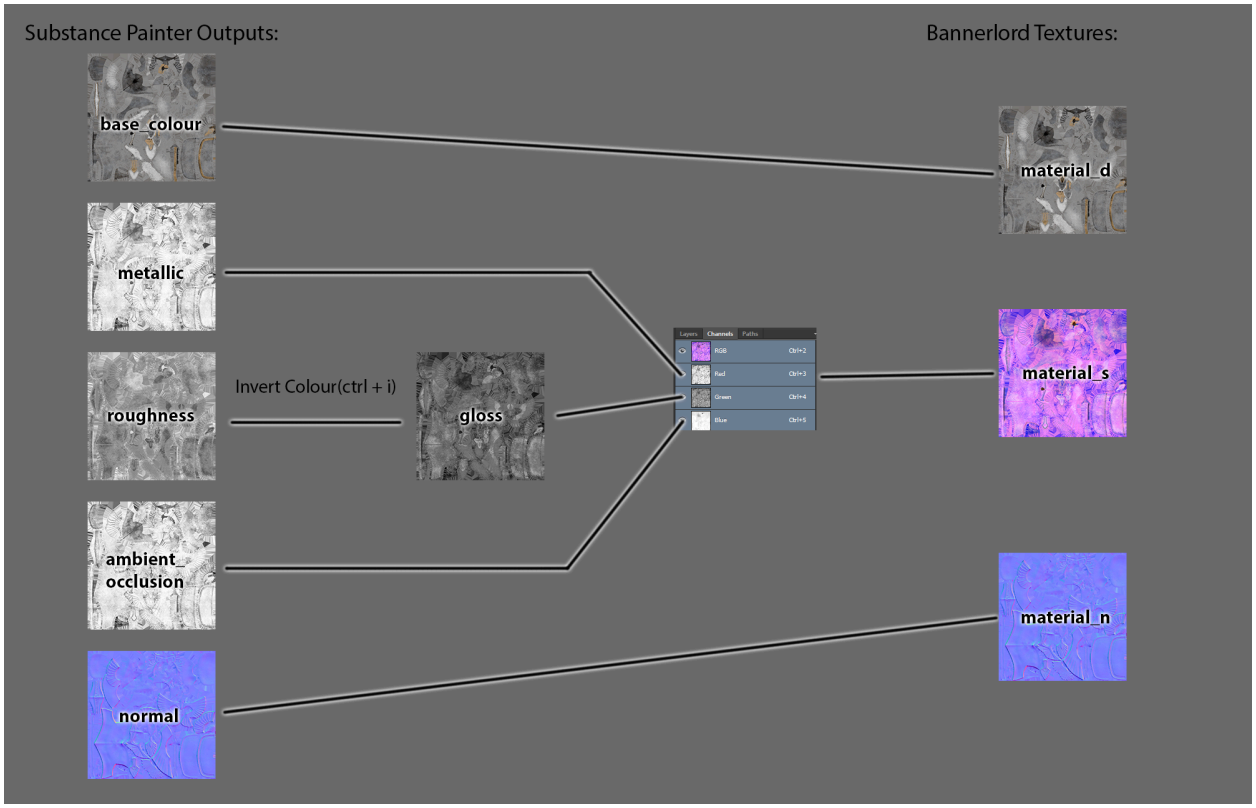
Resolution

After

Textures

Explain texture workflow





Opacity

To add opacity to a material, the information needs to be included in the diffuse.png. The parts that should be transparent need to be transparent within the .png, this will give alpha information to the engine. The transparent spots can be manually erased in a program like photoshop, or an opacity map can be used to select and delete the transparent spots.

Once the textures are imported and material made, the proper flags can be turned on. Firstly, enable **alpha_test** under Material Shader Flags. After this, hit **Compile Shader** which will allow transparency. Lastly, under Transparency, increase **Alpha Test** to a satisfactory amount(Value 0 to 255).

If your mesh has cloth physics(meaning vertices have black alphas) you will want to enable **disable_vertex_color_alpha** or else those vertices with black alphas will become invisible.

Material Settings

After

Asset Implementation

Armour

Reference models for characters and armours needed.

For armours there are `modifier_group` and `material_type` tags. The former controlling the type of modifiers available for the item and the latter controlling the sound effects created from this item.
modifier_group: “cloth_unarmoured” “leather” “plate” “chain” “cloth”
material_type: “Cloth” “Leather” “Plate” “Chainmail”

Does the appearance value affect visuals? Is this simply how often it appears in shops?

subtype vs Type?

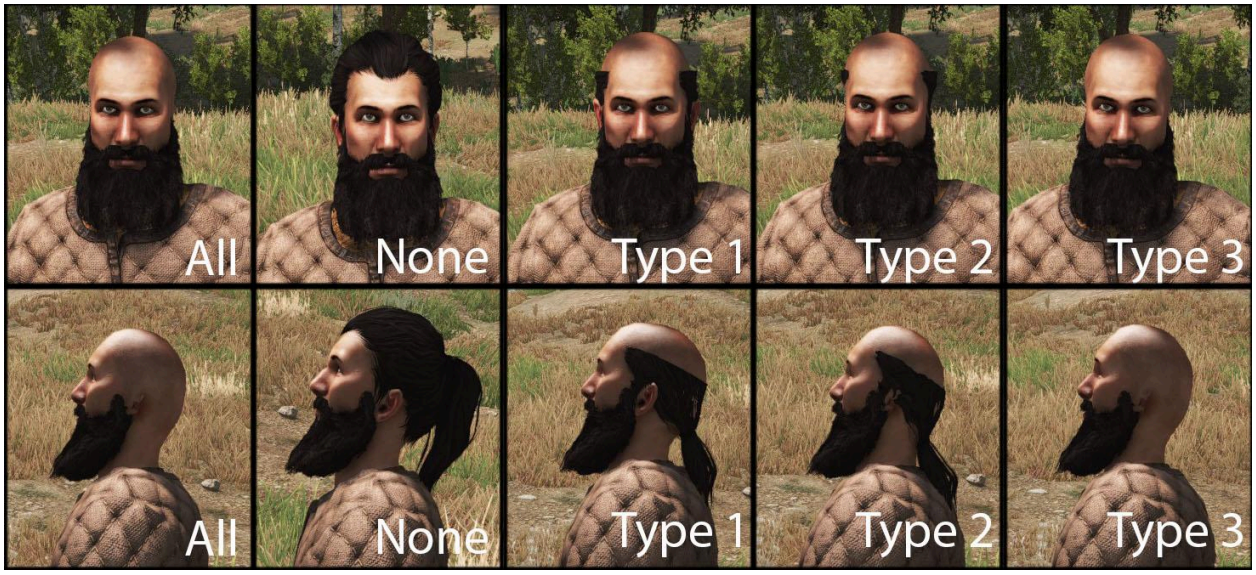
Armour meshes can cover the character’s meshes with the following tags:
covers_hands="true" covers_legs="true" covers_body="true" covers_head="true"

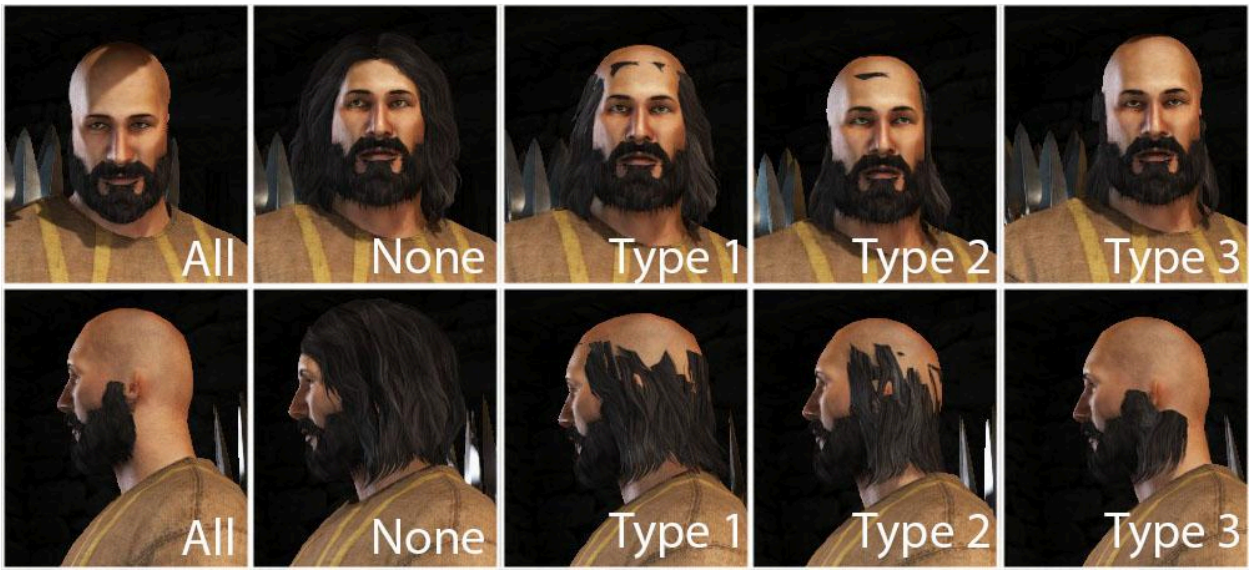
has_gender_variations
Each armour piece has the option to have gender_variations. To add an armour variation for female characters, there needs to be an additional mesh created with the suffix of `_converted`. This mesh should automatically be used if the character is a woman.
*Note, ARMOUR_converted will need its own LODs.
Taleworlds has automated this process, ideally they can share this method with the community as well.
Hasn't been experimented with, surface level knowledge.

Helmets

Helmets have options to cover or morph hair and beards that are edited in the sp_items .xmls. Options include covering all, none, or morphs of type 1, 2, or 3.

Hair Options





All types remove hair off the top of the head. Types 1 and 2 tighten the hair around the head to prevent clipping with head armour, type 2 having tighter hair and flattened ears. Type 3 removes any hair at the back of the head along with flattening the ears. Some female hairs appear to be long enough to contradict the back hair length on type 3.

Investigating more hair types will help get a better idea of how each type should be used.

Beard Options

Looking at the current selections for beards, the type system feels inconsistent. This requires either another look into a larger selection of beards, or TW has not fully implemented this feature as of yet.

Ideally the community can make(or Taleworlds can provide) a reference mesh for modders to make helmets around with the confidence that their helmet mesh will work with a hair or beard type.

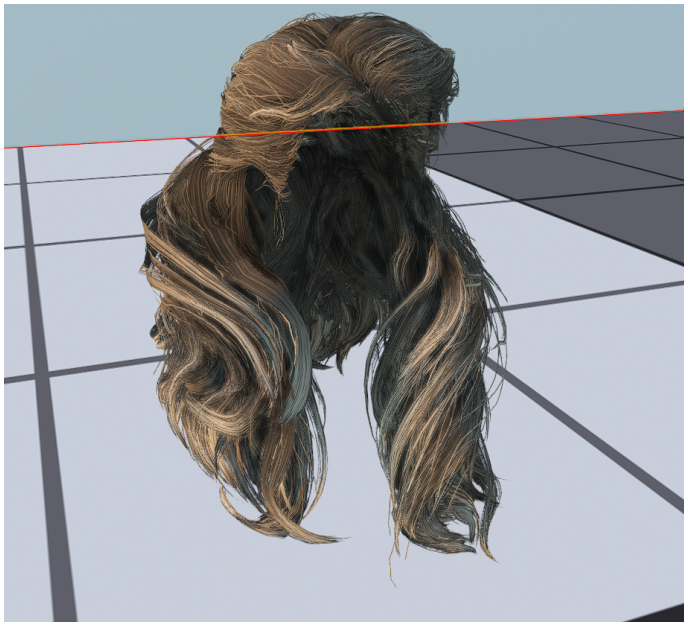
Hair Styles

Bannerlord hair meshes utilize vertex colours. Red seems to be default, blue seems to darken the hair(likely through ambient occlusion) and green adds a sheen.

Full Blue Vertex Colour: Blue adds AO(darkness)

Full Green Vertex Colour: Green adds shininess(glossiness or metallic?), exaggerates highlights and darkens the opposite

Full Red Vertex Colour: Makes the hair very rough and dry look, removes shininess and shows more of the diffuse colour.



Resources:
https://www.artstation.com/artwork/yb1b89
https://www.artstation.com/artwork/39K6Wm

Hand Armour

Straight forward implementation.

Some meshes have closed fists while others have open hands, are there shape keys for hand armours? And if so, does Taleworlds have a way to automate this?

Feet Armour, Shoulder Armour

Straight forward implementation.

Body Armour

Modders can include additional meshes with the suffix of "_slim" with the purpose of inserting a new mesh into the game when a character wears the body armour with hand armour. The inserted mesh is smaller where the hand armour is, reducing or removing clipping.

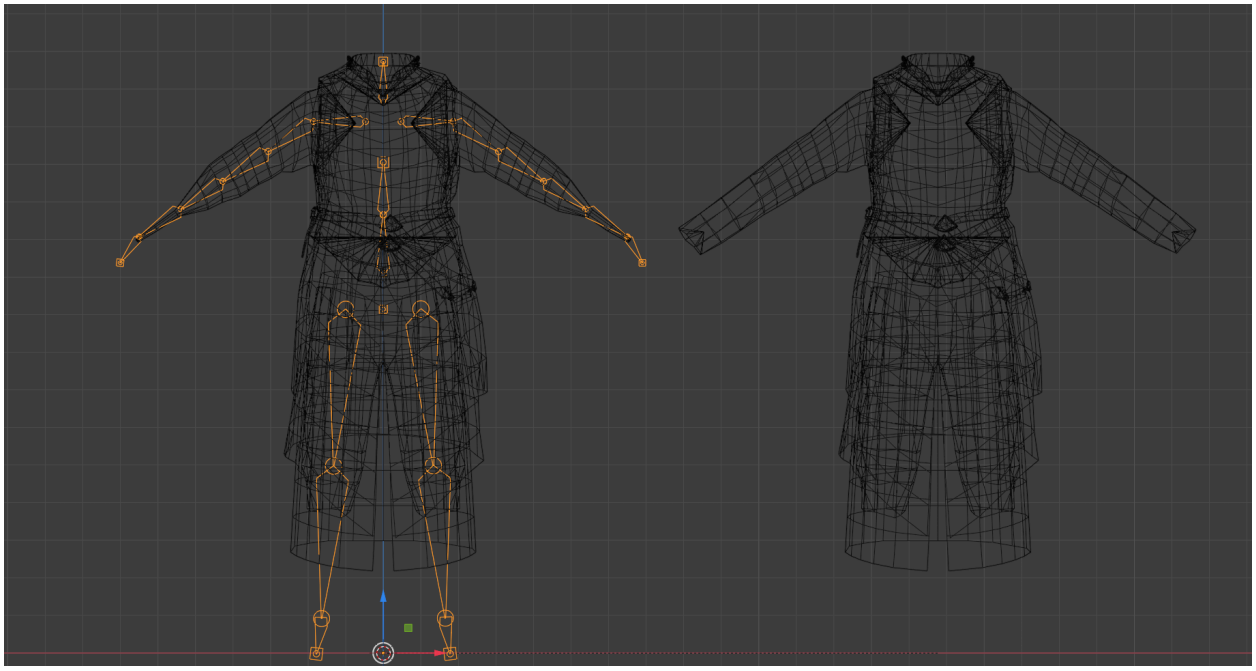
The slim version of armours should only be added when all work on the original mesh is complete, elsewise you will need to remake the slim version to include your updated work.

Taleworlds has a way to automate this, ideally they share their method with the modding community.

*Note, ARMOUR_slim will need its own LODs.

This compounds with the gender_variations mentioned above. To make a female version of an armour use the slim function, a mesh will need to be added with the suffix of “_converted_slim”.

And yes, ARMOUR_converted_slim will need its own LODs.



Example of the slimmed vs. the standard armour.

Shields

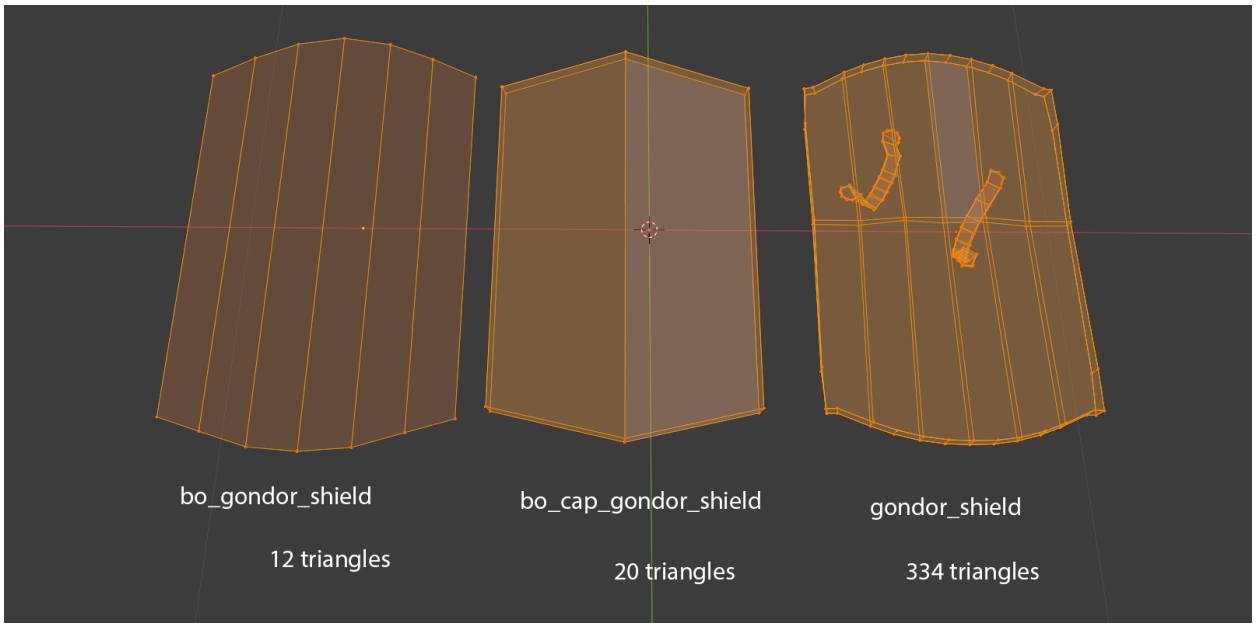
Shields require 3 models for full implementation. The mesh for visuals, and two physics meshes.

The mesh is the in-game asset, the low poly and model that is going to be seen by the player. Position within the 3d space of your modeling program is important, here is a file of my shield that will show positioning and examples of the hitboxes:
https://drive.google.com/file/d/1VTBo4NxbngNePlgiJ_ndJMaRSLcEbMdN/view?usp=sharing
It’s best to use a shield close to what you want to implement. and make sure to copy the position, center line, and hand grip.

The other models you will need are physics bodies: body_name and shield_body_name. body_name requires the prefix of `bo_cap_` on your shield name and shield_body_name requires the prefix of `bo_`. For example, your .fbx export should look like the following:

These physics meshes **require** their material to be named in specific ways. bo_cap_SHIELD needs its material to be called “wood” and bo_SHIELD needs a material dubbed “wood_shield”. Renaming is done in your 3D program, when imported to the Editor, it’ll automatically assign the material you have named.(Perhaps different naming conventions offer different material physics?)

Here is a preview of a modded shield with the physics meshes.

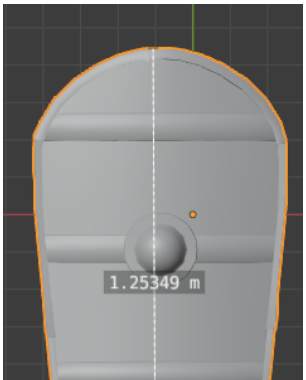


Note: they are moved to the left and right for a preview, when putting into the game they should be aligned in the same space.

Here is a link to the above assets ready to implement to Bannerlord. They are meant for learning, but use them however you wish.

Triangle count is important for the physics calculations, more polygons means a lot more calculation by the engine, so low is key. bo_SHIELD likely is the hitbox for projectiles and melee. bo_cap_SHIELD is likely the ragdoll hitbox for the shield. *conjecture, needs confirmation

*Note: Do not enable skinning in the material settings of your shield mesh.



.xml tags

Make sure to reference your physics meshes in sp_items, `body_name` and `shield_body_name`

recalculate_body:
Every Taleworlds shield has false, perhaps there is a hidden function here.

item_usage: “hand_shield” “shield”
There are two types of options for the tag, I am unsure of the difference.

weapon_length:
Each shield has a weapon_length, `kite_shield_e` here for an example, has a length of 118. When measured in Blender, this number comes to roughly 125, a clear discrepancy. I cannot extract the physics mesh to measure, so perhaps this is the length of that. Or there is another purpose behind the length, and an easy way to calculate it.

weapon_class: “LargeShield”
The only option for `weapon_class`. Perhaps this is an unfinished feature, as the prefix of ‘Large’ suggests other shield types within the weapon_class. Or perhaps this is only a way to differentiate shields from other equipment types.

physics_material: “wood_shield” “metal_shield”
Changes the sound effects used by the shield.

item_usage: “shield” “hand_shield”
Unsure about the differences between the two, perhaps animation.

Crafted Weapons

Melee weapons in Bannerlord have crafting_templates and crafting pieces. To add a weapon in Bannerlord really means to add weapon parts into the game and using those pieces in a crafting_template to make a CraftedItem.

Reference meshes for the different crafting pieces.

crafting_template	***Crafting Pieces***

OneHandedSword	*Blade*	*Guard*	*Handle*	*Pommel*
OneHandedAxe	*Blade*		*Handle*	
Mace	*Blade*		*Handle*	
TwoHandedAxe	*Blade*		*Handle*	
TwoHandedPolearm	*Blade*	***Guard***	*Handle*	***Pommel***
TwoHandedSword	*Blade*	*Guard*	*Handle*	*Pommel*
Pike	*Blade*		*Handle*	
Dagger	*Blade*	*Guard*	*Handle*	*Pommel*
Javelin	*Blade*	*Guard*	*Handle*	*Pommel*
ThrowingAxe	*Blade*		*Handle*	
ThrowingKnife	*Blade*	*Guard*	*Handle*	*Pommel*

These are the piece combinations Taleworlds uses on their weapons, **bolded** pieces being optional. It could be that every piece is optional when putting together a weapon, but this may affect the weapon physics/balance.

.xml editing

To add a crafted weapon into Bannerlord you need to edit 3 .xmls: `weapons`, `crafting_pieces`, and `crafting_templates`.

1. weapons:

Within this file you create a CraftedItem from crafting pieces based on the crafting_template.

2. crafting_pieces:

This file is where you add the meshes into the game.

physics_material

body_name

holster_mesh

holster_body_name

holster_mesh_length

length

1.What is the purpose of length on a guard?

previous_piece_offset, next_piece_offset, and piece offset

distance_to_next_piece, and distance_to_previous_piece

blade_length
AXES: vertical length of cutting part of blade.

blade_width
AXES: distance from cutting part of blade to center of handle.

*With so many different options for each piece, many differing based on the type of weapon, it's best to find pieces that you want your model to look like and copy their information and tweak numbers where necessary. Ideally we could get a visual within our 3D programs to measure the needed values or some way to automatically calculate them.

3. crafting_templates:

The game needs to recognize your parts, so in this file, you must add your crafting pieces to the CraftingTemplate that corresponds to what crafting_template the piece uses in the 'weapons' file. For example, I added a line under UsablePieces in the OneHandedSword CraftingTemplate because this blade is referenced within the weapons and crafting_pieces files.

```
<UsablePiece piece_id="battania_blade_4"/>
<UsablePiece piece_id="battania_blade_5"/>
<UsablePiece piece_id="battania_blade_6"/>
<UsablePiece piece_id="gondor_sword_blade"/>
<UsablePiece piece_id="gondor_sword_blade_citadel"/>
```

4. weapon_descriptions

Copy paste like in crafting templates 🙂

Ranged Weapons

Bows

Bow mesh for placement reference.

The bending of the bow is controlled by the green channel of the vertex colour. Paint the parts that don't move a full value of green and anywhere that is meant to bend, like the ends of the arms on the bow and the string, less or no green.

To have bows animated, you need to change the bow's shader to **longbow_deformer** or **bow_deformer**, the difference being the bend amount. **Skinning** does **not** need to be enabled.

Note With different parts of the bow set using different shaders, you can't have the arrow, quiver, nor bow mesh use the same material/texturespace.*

1.XML information to do...
2.Don't know how the holstered bows works, it appears the bows in the game have a secondary model with the suffix of '_holstered', but I cannot get this working.
3. Is the bow size limited? I changed the size of my bow several times, but it did not change size in game. Based on physics mesh?

Quivers and Arrows

Quiver and arrow meshes for placement reference.
The placement of the quiver on the character can be changed in the weapon file, simply rearrange(or add) the holster list under **item_holsters.** The quiver changes position, but the draw animation does not account for this.
Quiver: 100 Alpha(white)
Arrow 1: 80
Arrow 2: 71
Arrow 3: 64
Arrow 4: 55
Arrow 5: 44
Arrow 6: 32
Arrow 7: 20
Arrow 8: 09

Crossbows

Crossbow mesh for placement reference.

Mount Equipment

Has not been explored

Characters

Has not been explored

Mounts

Has not been explored

Scene Assets

Has not been explored

Flora

Has not been explored

```Questions```
1.* Hand armour has morphs/animations baked in, how does Taleworlds achieve this? Could the method be shared? If we get an inside to how these morphs are added, then implementing character meshes will be more easily done as well.
Hand armour and faces use this.
Murat - However, blender sometimes have issues with exporting the keyframes.

2*. The tpacs are said to be packed in a way to increase performance, could we get the method behind how to optimize this? Is it the same idea has optimizing texture space/number of textures?

3.* A breakdown and explanation of the physics objects for things like shields would be helpful, I think I have this relatively correct, but I'd like confirmation.

4. *If I made 3 materials, all using the same textures, does this harm performance? It seems like I will need a different material for each deformer for the bow(bow, arrow, quiver), but right now my textures have all of these on the same sheet.

5. *How do you implement a holstered bow mesh? I cant get the holstered version of a bow to appear, despite copying all the info I could from the other bows.

6. *How do you get the quiver to lose arrows as you approach empty? I assume it needs the quiver_deformer, but I don't know how to make individual arrows to go.

7. *How do you get arrow meshes to appear in flight?

8. *Is the bow size limited? I increased the size of my bow several times, but it did not change size in game. Is it based on physics mesh?

9.* Is there a way to change the draw animation depending on the holster? I changed it so it appears on the back first, but the character still draws arrows from the hip.

10. *Shields have a weapon_length, why? Does the physics objects not perform that purpose?

11. *Likewise, guards have a length, what is the purpose? It doesnt seem to be the same type of length as the other crafting pieces(pushing the next piece by that amount)
```

12. *Vertex animations

```Suggestions/Requests```
1. *Currently, none of the native module is editable meaning that we cannot export and view Bannerlord's assets to learn from without something like TpacTool(which is broken in newer releases and gives little information). It makes sense that a company will not want to make all of their assets readily available publicly; a few assets of most types should be good enough reference for people. These assets include: Skeletons for Humans, Horses, and others. The human and horse bodies to model around. Each of the basic armour types. Bows, crossbows, quivers, arrows, bolts. Crafting pieces for weapons. Physics objects for equipment like the shield.

2.* Documentation for different programs. I've only documented stuff for Blender, but there were a decent amount of hurdles I came across and I assume there will be plenty more for other programs too.

3. *Apparently there is an automatic system for the modelling programs to create _converted and _slim models. Sharing this with the community would be very helpful.

4. Making weapons is quite a hassle, I think Taleworlds should share more of their workflow with implementing this. One can get through by making tiny tweaks and a lot of measuring, but it is very time consuming, I assume the team has some sort of method that speeds up this workflow.

5. Does Taleworlds have a formula to calculate weapon stats based on the stats inputted? Making high or low quality weapons is hard when constantly tweaking weights or sizes, the only stat that feels like there is control over is the damage.

7. Reference meshes that show where hair and beard meshes are morphed to will help people design helmets with less clipping.*
```