Intermediate Guide: Material2 & Custom Normals

Editing .decl Files

Required Tools:

- **Notepad++** - **Download**
  You will need this in order to edit decl files properly.

- **EternalResourceExtractor by PowerBall253 - Download**
  If you haven't already done so, it is essential that you use EternalResourceExtractor to extract all the game resources. This will give you access to *all* decl files of all kinds, and allow you to see any and every file path to get a greater understanding of where everything is located, which will save you hours of time down the road.

Continuing on from the previous guide, we're now going to talk about editing decl files, why you would do it, what to modify, and what not to modify. We're going to go through each set of files you can edit for your texture mods one by one, starting off with the most basic and ending with the most specialized. So here we go!

What are .decl files and what do they do?

Decl files **declare** definitions for the game to utilize in establishing functionality on boot. They do everything from telling the game how to make the Slayer animate properly down to simply what things are named in menus. They control particle colors and appearance, what models are assigned to what entities; basically how everything in the game functions is dictated by these files. They're extremely important, and, as a result, in addition to purely cosmetic purposes, can also be used to cheat.

.decl files and the new Mod Injector

As of Mod Injector 6.0, modifying certain .decl files that can be used to cheat in the game will disable Battlemode. It is up to you whether to use mods that alter those files or not and the Eternal Mod Manager will let you know if your mods are safe for multiplayer or not. Now, let us get to the most important .decl files you can use to improve and enhance your projects.

Material2 .decl files

Material2 files, at base, declare what textures are assigned to a given part of a model. They exist for all skins of all kinds (Slayer, guns, and demons). These can be modified to assign *any texture* to *any model* and will match the names of the textures being assigned to that part of the model by default. However, assigning textures to a model not created for that model will result in stretching and wrapping in a way that doesn't match the shape at all, so instead, we're going to discuss proper usage for

positive results.

Slayer files

As we discussed in the previous guide, Slayer skins have textures for every part of the model that they are assigned to. For the sake of simplicity, we're going to use the Maykr skin as an example. The parts of the models for texture assignment are:

```
doomslayer_arm_left
doomslayer_arm_right
doomslayer_helmet
doomslayer_knife
doomslayer_launcher
doomslayer_legs
doomslayer_torso
```

So where do you find these decl files? Regardless of what resource library you're looking in, all material2 decl files will be located in:

`generated/decls/material2`

Now, it's very clear what each of these model parts are and the only difference between the HQ and Photo Mode textures is the _hq suffix in the file name. If you were to replace the Fallen Angel (set41), your textures would be located in:

`warehouse/customization/characters/doomslayer/set41`
`warehouse/customization/characters/doomslayer_1p/set41`

Since this is set41, the textures will be named doomslayer_arm_left_set41_hq and so on. These are the same names as the material2 decl files and the file path for the textures will also be the same inside the material2 folder:

`warehouse/generated/decls/material2/models/customization/characters/doomslayer/set41`
`warehouse/generated/decls/material2/models/customization/characters/doomslayer_1p/set41`

If you'd like, you can take some time to locate these files, open them up, take a look at them so you can follow the next sections, and then continue on.

gameresources or warehouse?

Depending on the skin you have chosen to replace, those textures may be in gameresources, gameresources_patch1, gameresources_patch2, or warehouse. If the texture files are in any of the

gameresources libraries, it is very likely that the corresponding material2 decl files will be in more than one of them. If this is the case, only the copies in the **highest priority .resources file** will be loaded by the game and actually affect the given skin. **Check this spreadsheet for current resource loading priority**.

For example, the material2 decl files for the default skins of every gun are in gameresources, gameresources_patch1, *and* gameresources_patch2. The only ones you want to modify and replace are the copies in **gameresoures_patch1**, because it is the highest on the resource priority list. If you have chosen a skin in warehouse, the corresponding material2 files will be there or in warehouse_patch1.

We will now discuss both what you can modify and what you *should* modify to get the job done.

material2 modification

The text of these decl will look similar to this:

```
{
inherit = "template/pbr_gun";
edit = {
	RenderLayers = {
		item[0] = {
			parms = {
				heightmap = {
					filePath = "textures/system/constant_color/grey_md.tga";
					options = {
						type = "TT_2D";
						filter = "TF_DEFAULT";
						repeat = "TR_REPEAT";
						format = "FMT_BC1";
						atlasPadding = 0;
						minMip = 0;
						fullScaleBias = false;
						noMips = false;
						fftBloom = false;
					}
				}
				smoothness = {
					filePath = "art/weapons/combatshotgun/skins/hotrod/combatshotgun_barrel_g.tga";
				}
				normal = {
```

```
        filePath = "art/weapons/combatshotgun/skins/hotrod/combatshotgun_barrel_n.tga";
    }
    specular = {
        filePath = "art/weapons/combatshotgun/skins/hotrod/combatshotgun_barrel_s.tga";
    }
    albedo = {
        filePath = "art/weapons/combatshotgun/skins/hotrod/combatshotgun_barrel.tga";
    }
  }
 }
 Parms = {
  wetnessdroptiling = 4;
 }
 SurfaceType = "SURFTYPE_METAL";
}
}
```

This is the combatshotgun_barrel material2 file for the Hot Rod Shotgun skin. The primary section to focus on first is the “parms” at the top, specifically the fields titled heightmap, smoothness, normal, specular, and albedo. You will notice that the primary purpose of this file is to assign the textures of this skin by *declaring* the file paths for each of them. These are, typically, what you will modify and what we will cover first.

Modifying File Paths

It is always recommended that you replace weapon skins in warehouse.resources. There are two reasons for this:

1. If you are declaring different file paths, it is pointless to do so for textures replaced in the gameresources libraries.
2. The combined file size of mods injected into warehouse.resources is far greater than that of the gameresources libraries. This is because all three libraries are actually all part of the same ‘mapresources’ library, common.mapresources. This means that common.mapresources has a far greater initial load and is far less tolerant of increased size. In contrast, warehouse.resources is the only resources library in the warehouse.mapresources library and will tolerate significantly more.

So why would you want to modify file paths? Well, the majority of skins located in warehouse are from event series and, thus, many players who recently purchased the game could very easily not have access to the skin you are replacing. As a result, you will want to change the file paths for a base

campaign skin such as Crimson, Midnight, or even Default so that it will use the textures for said event skin and all players who have not unlocked that skin will be able to use your mod. (NOTE: If you are editing the Skullface skin, one of the Maykr skins, or any other skin that has a completely different model, you cannot reassign it to a campaign skin using this method. The method for doing so will be covered in the advanced guide if you want to skip this method and continue there. For all default-style skins, however, that method will typically not be necessary and you may use either based on your personal preference).

So, how is this done? Well, let's return to the Hot Rod Shotgun skin and use it here as well. If you return to the example, you will notice that the file paths declared always start with the names of folders within the resources libraries and do not begin with the name of the resource library itself. This is because if a folder path is not located in the resource library, it will search the others to locate it in order of load priority. **Check this spreadsheet for current resource loading priority**.

This load priority will be relevant in the advanced guide, but for now, you can ignore it. However, each given resource library will always search *itself* first, as you would expect. The Hot Rod skin is located in warehouse and, thus, the corresponding decl files will search warehouse, find the textures in the declared file paths, and assign them to the model. However, if you were to modify a decl file in gameresources_patch2 for a Shotgun skin located in common.mapresources to the file path for the Hot Rod skin, it would fail to find it in gameresources_patch2, eventually search warehouse, and load those textures. For the purposes of this guide, we will use the default Shotgun skin. The declared file paths of the default combatshotgun_barrel file (combatshotgun_barrel.decl located in gameresources_patch2/generated/decls/material2/art/weapons/combatshotgun) will look like this:

```
smoothness = {
	filePath = "art/weapons/combatshotgun/combatshotgun_barrel_g.tga";
}
normal = {
	filePath = "art/weapons/combatshotgun/combatshotgun_barrel_n.tga";
}
specular = {
	filePath = "art/weapons/combatshotgun/combatshotgun_barrel_s.tga";
}
albedo = {
	filePath = "art/weapons/combatshotgun/combatshotgun_barrel.tga";
}
```

"Smoothness" will correspond to your "glossiness" layer in Substance Painter if you are using it for your textures (_g). Normal and Specular are self-explanatory, but "albedo" refers to your Diffuse layer. So, we will now modify the parms to use the Hot Rod textures instead, which will make them identical to the parms of the Hot Rod decl file:

```
smoothness = {
	filePath = "art/weapons/combatshotgun/skins/hotrod/combatshotgun_barrel_g.tga";
}
```

```
normal = {
	filePath = "art/weapons/combatshotgun/skins/hotrod/combatshotgun_barrel_n.tga";
}
specular = {
	filePath = "art/weapons/combatshotgun/skins/hotrod/combatshotgun_barrel_s.tga";
}
albedo = {
	filePath = "art/weapons/combatshotgun/skins/hotrod/combatshotgun_barrel.tga";
}
```

The main drawback to this method is that your textures will replace *both* the textures for Hot Rod *and* Default. However, since you are doing this for the benefit of those that do not have the Hot Rod skin, it can be entirely forgiven. To inject these decl files, you will create the file path gameresources_patch2/generated/decls/material2/art/weapons/combatshotgun, place the files in this new combatshotgun folder, add the gameresources_patch2 and inject! Your skin will now replace the default skin and be usable for everyone! For most skins, this is all that will be required. For this method, there is no need whatsoever to ever inject custom normal maps. The vanilla normals will work just fine for your purposes. However, if you are using Substance Painter and you have added additional detail from the skin you have created, you may want to use your own custom normal maps. For those of you to whom this applies, continue on to the second half of this guide.

Custom Normal Maps

Required Tools:

- **SAMUEL by SamPT - Download**
  To ensure the best possible results, you will want to use SAMUEL to specifically extract Normal Maps. This is due to encoding that will be explained briefly. For all other textures, VEGA will do the trick, but you may elect to use SAMUEL for all your textures and simplify the process.

- **VEGA by DTZxPorter (optional) - Download**
  See above.

- **GameImageUtil by Scobalula - Download**
  You will use this program on normal maps extracted by SAMUEL to make them usable in Substance Painter.

- **Substance Painter/Photoshop**
  You will use either one of these programs or both to create the modified Normal maps.

- **Auto Heckin' Texture Converter by PowerBall253 - Download**
  You will use this to convert your Normal maps from Substance Painter to work properly in

game.

Before we get started, I will give a brief history of modifying normal maps for Doom Eternal, as it will help you better understand file formats, which is helpful information for best understanding the whole process. At the beginning of texture modding, we found that textures (aside from Normal Maps), need to be encoded as "BC1a" DDS files before using Divinity Machine (now a part of the Texture Converter) to make the TGA files for injection. As a result, VEGA extracted all image files (including Normal Maps) as BC1a and Divinity Machine encoded them as BC1a for injection as well. Normal Maps produced by this process looked visually identical to the Normal Maps from the game in terms of the dark red and green hues, but they would break in game, resulting in terrible, inconsistent lighting.

To compensate for this undesired outcome, a time-consuming, complicated process was devised involving Blender and Photoshop to brighten the hues of red and green, which actually caused them to work in game. So why was this the case?

The red and green in Normal Maps are not processed as colors. Instead, a complex algorithm converts them to information that is used by the game to simulate lighting and reflections. This is why the encoding format and hues of the Normal Map are critical. The process we used for the majority of a year, however, created a problem of its own: Compressed Normal Maps that produced pixelation and loss of information, making shiny surfaces look choppy and wavy; a complete mess. We spent months trying to fix this issue to no avail. However, a solution was eventually discovered and, as it turned out, that solution was impossible for us to find where we were looking because it happened at the beginning before our process of extraction even began.

As mentioned at the beginning, VEGA was extracting Normal Maps as BC1a because it was deduced that they must be BC1a because all other textures required BC1a encoding. This was, however, completely wrong. The normal maps, in fact, required BC5 encoding to work properly and, as of now, SAMUEL is the only app that will give this proper encoding. With it, there is no more damage to the Normals or loss of information at the outset and, using the process which I will now outline, will not be lost in the end product you see after injection.

Creating Your Custom Normal Maps

1. Use SAMUEL to extract your desired Normal Maps. You will notice that these normal maps are in only red and green, which is what Doom Eternal uses. However, they will also require blue in order for Substance Painter to use them properly and, later, export them in a usable state (to understand how to properly utilize and export normal maps in Substance Painter, make sure to watch the tutorial video also posted on this wiki).

2. Run GameImageUtil. In the drop down at the top, select the last option: “XY.” Drag and drop your DDS BC5 Normal Map onto the app window and it will export a BC5 PNG with blue added in the same folder as your DDS image.

3. Run Substance Painter. Import your Normal Maps, create your skin, and then export your customized Normal Maps (PNG).

4. Open your PNG file in Photoshop and save it with DDS encoding. Whether you are using a simple Intel plugin or the NVIDIA encoder, select “8 bit BC5” from your list of options and save the file with the appropriate name (e.g., combatshotgun_barrel_n.dds for the example in the previous section).
5. Drag and drop your DDS file onto the Auto Heckin’ Texture Conerter .bat file and it will convert it to a BC5 .tga file ready for injection. Congratulations! You’ve just created your own custom Normal Map and, when paired with the material2 replacement method, it’s now available for everyone to use!

Coming up in the Advanced Guide, we will discuss a more complex method for assigning event textures to campaign skins and taken on the most complicated process of all: Adding new custom assets. See you there!

Revision #7
Created 19 September 2021 14:31:04 by Will Ryan
Updated 14 March 2022 14:15:09 by SamPT