Zipmod Basics with Unity

Wogrim's Epic Guide to Making a Zipmod Using Unity

Is There Another Way to Make a Zipmod?

Yes, the other way involves making your asset bundles with SB3UGS by copying existing ones and editing them.

Why Does This Guide Not Use KK Modding Tools?

I want this guide to be equivalent to the SB3UGS version. KK Modding Tools makes some things easier, but in a way that is not good for teaching the basics you need to know to be able to troubleshoot any problems you may run into.

Why Make an Empty Zipmod?

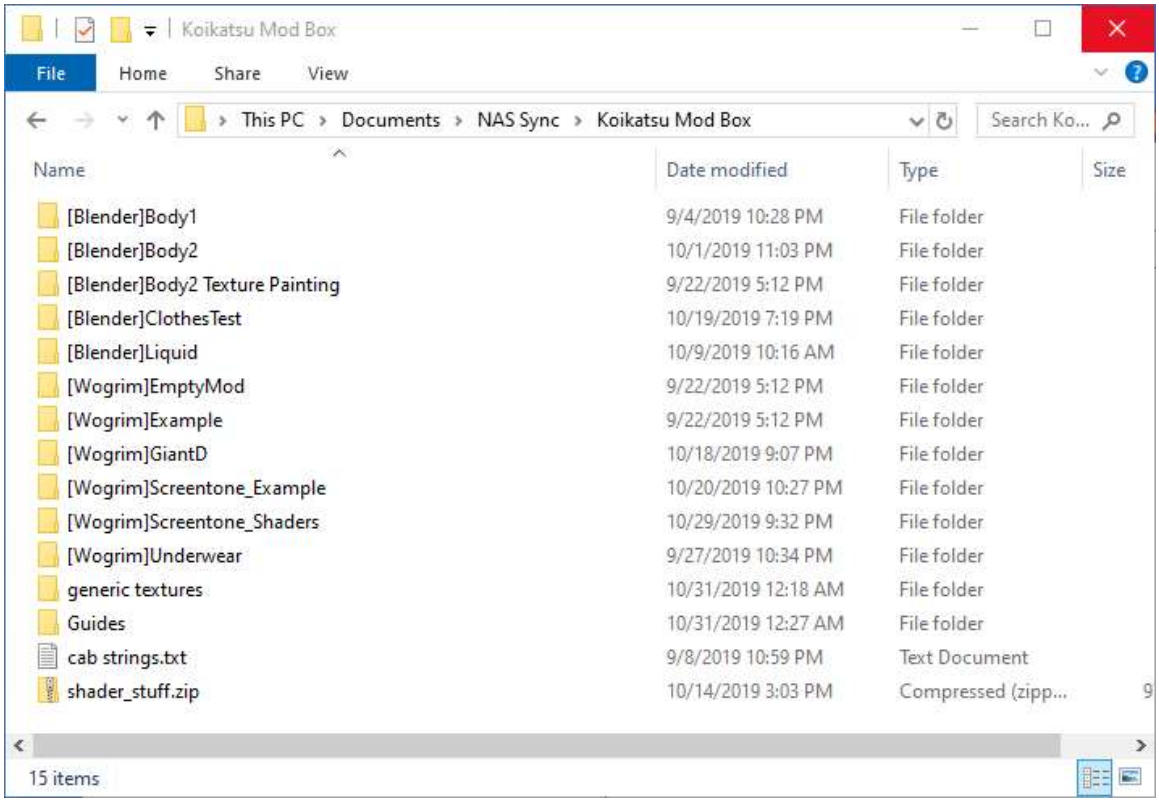
Before we make a working Zipmod, we will make an empty one. This will give you a folder that you just duplicate and modify whenever you want to make a new mod. It is faster than remaking everything from the beginning. It is a little different from the EmptyMod made in the SB3UGS workflow; we will not put asset bundles in it because we will be using Unity to create a new asset bundle per mod instead of using SB3UGS to modify existing asset bundles.

What Software is Needed To Follow This Guide?

- **Sideloader** - This is the plugin that loads zipmods. It is included in the HF Patch. I haven't looked into how to install it otherwise.
- **SB3UGS** - Get this from enimaroah's ~~github~~ gitea page. You need this to look at game files to see how your mod should work.
- **7-Zip** - This is used at the end to "zip" your mod's files with no compression. Other software can probably do this.
- **Text Editing Software** - Notepad is fine.
- **Table Editing Software** - Optional. I use Excel. It is to edit the list file, which is .csv format. Notepad can open this, but then it isn't organized into proper columns.
- **Image Editing Software** - Something to make thumbnails and textures. GIMP is free and works fine.
- **Unity 5.6.2f1** - this is the version that the game runs; you need to create your asset bundles with the same version

So I've Got the Software, What Now?

Make a folder somewhere on your computer for working on mods. I call mine "Koikatsu Mod Box". Inside of it, make a folder called "[YourName]EmptyMod". The naming doesn't really matter, but it is strongly recommended to be organized. Here is what my mod box currently looks like.



We will first create the manifest file, called **manifest.xml**. Create it and open with text editor. It can't be hiding in any subfolders or Sideloader can't find it. The only required field is "guid", which must be different from all other mods. For this reason, it is recommended that you put "YourName.ModName" to avoid conflicts. You should be able to copy-paste this and then change it.

```
<manifest schema-ver="1">
	<guid>Wogrim.EmptyMod</guid>
	<name>Empty Mod</name>
	<version>1.0</version>
	<author>Wogrim</author>
	<description>empty</description>
	<website></website>
	<game>Koikatsu</game>
</manifest>
```

Next we will need to make some folders, which are to emulate the folder structure of the game files. For this reason, I will explain important folders in the game files first.

How Are the Game Files Organized?

Go to your game install folder. There are 3 folders in here that are good to know about:

- **abdata** - Contains all the asset bundles (ABs), which are files that contain assets. Assets are basically any kind of textures, models, sounds, animations, or anything else the game needs to load. These are what we will make copies of and modify.
- **mods** - This is where the zipmod goes when we are done making it.
- **UserData** - This isn't important for this guide, but your screenshots, MaterialEditor exports, character cards, and game saves end up here.

Go into the **abdata** folder. There are 3 important folders in here:

- **chara** - Contains all of the game's models and textures for your characters. All skin textures, clothes, and accessories.
- **list** - Contains the lists that tell the game what types of things are in the other folders. We have to make a list file for any items we make to show up in the game.
- **studio** - I don't like studio because you don't get to interact with the girls, but this is where studio items are.

Go into the **chara** folder. All of the .unity3d files in here are asset bundles. The only folder in here is **thumb**, which contains assetbundles for thumbnails (which are used for selecting items in Maker). There are a couple extra things I will not explain, but for the most part these are organized as follows:

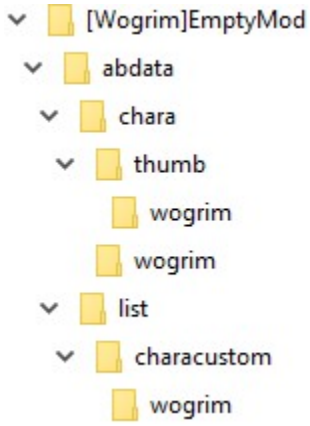
- starts with **ao_** - These are accessories.
- starts with **bo_** - These are hair types.
- starts with **co_** - These are clothes.
- starts with **mt_** - Some of these are texture-only items like eye textures, and some are textures shared by multiple items.
- ends with **_00** - These are the base game files; if it ends in a different number, it is part of a DLC. This number matches a list file, so it can help you find the right file.

Go into the **thumb** folder. If you are trying to find an in-game item, this is a place to find its name, because the in-game name is different from the item's name in the asset bundle.

Go back up to the **abdata** folder and go into the **list** folder. From there, go into the **characustom** folder. These asset bundles contain all the lists, numbered like the asset bundles in the **chara** and **thumb** folders; when you need to see how an item type's list file should look, check out 00.unity3d.

Okay, Back to Making EmptyMod

For now, we will recreate the folder structure in our EmptyMod folder, starting with **abdata**. I add my name as a subfolder in all the folders because I don't want to worry about my mods somehow getting mixed up with someone else's. You don't have to do this, but if you do, you need to make the folder names all lower-case because Untiy makes the asset bundle paths all-lowercase when you specify them later on.



The List File

A zipmod normally has a list file, which tells the game what types of items are in the asset bundles. We have no items in our EmptyMod, but we will make a list file as an example. The format for a list file depends on the type of item it is. If you are making a mod with multiple item types, you have to make a list file for each type. I will make a nipple texture list file for example, but it doesn't matter what type of list file you put in your EmptyMod. What matters is that you understand the process of how to create it.

Make a file in your EmptyMod's **abdata/list/characustom** folder (or subfolder) called **EmptyMod_Nipples.csv**. You can open with a text editor or a table editor. To find what a nipple texture list file should look like, we will look at the game's list files.

Open SB3UGS. There are some important areas:

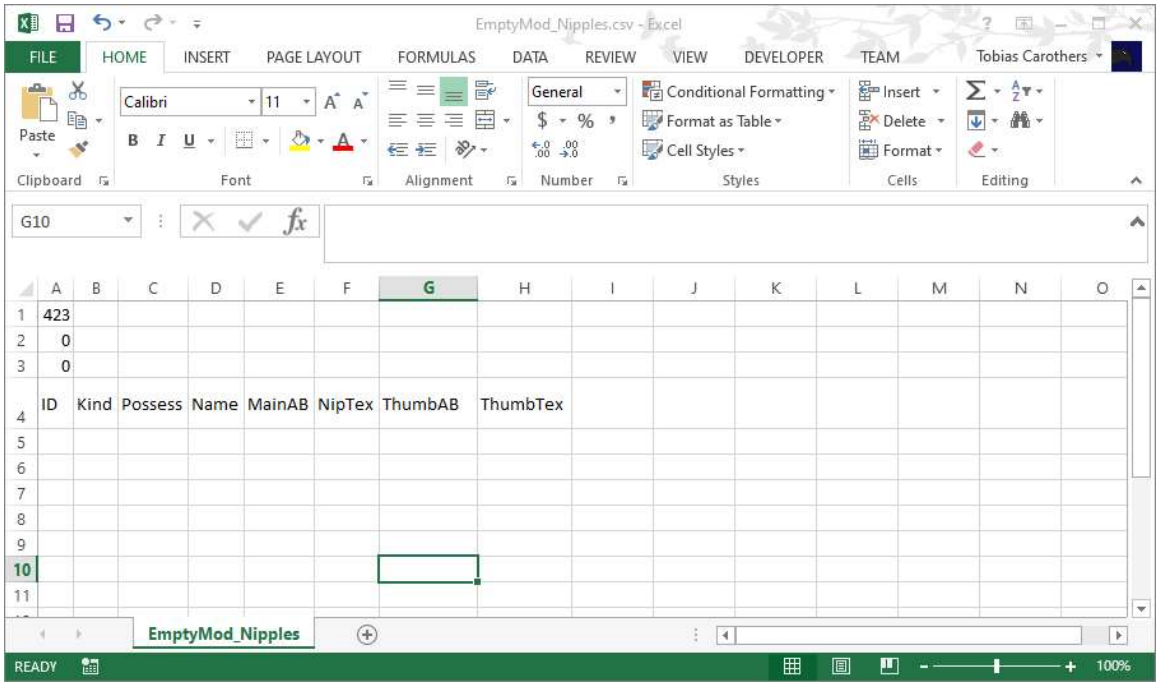
- **Files** - Any asset bundles that are open show up as tabs here. Drag your **EmptyMod.unity3d** into this area.
- **Editors** - If you double click on an asset it will generally show up here. Sometimes you will edit things, sometimes you will replace things, sometimes you will export things.
- **Renderer** - If you select one or more meshes you will get a preview here. It will change to an **Image** window if you have a Texture2D asset selected.
- **Log** and **Script** - Useful messages sometimes show up here.

Go the game's **abdata/list/characustom** folder and open **00.unity3d** in SB3UGS. Double click on the **mt_nip_00** asset and the Editor section will look like this.

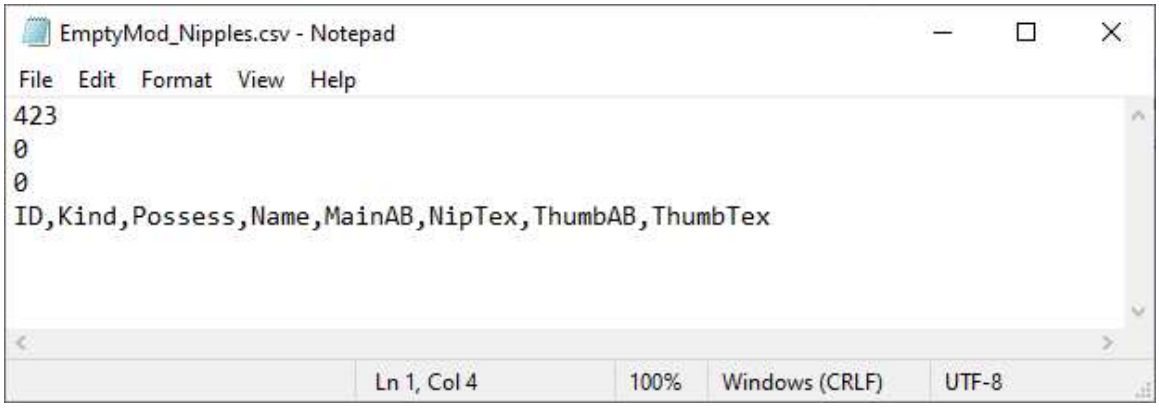
mt_nip_00 [423]

ID	Kind	Possess	Name	MainAB	NipTex	ThumbAB	ThumbTex
0	0	1	タイプ1	chara/mt_nip_00.unity3d	cw_t_nip_000	chara/thumb/mt_nip_00.unity3d	thumb_nip_000
1	0	1	タイプ2	chara/mt_nip_00.unity3d	cw_t_nip_001	chara/thumb/mt_nip_00.unity3d	thumb_nip_001
2	0	1	タイプ3	chara/mt_nip_00.unity3d	cw_t_nip_002	chara/thumb/mt_nip_00.unity3d	thumb_nip_002
3	0	1	タイプ4	chara/mt_nip_00.unity3d	cw_t_nip_003	chara/thumb/mt_nip_00.unity3d	thumb_nip_003
4	0	1	タイプ5	chara/mt_nip_00.unity3d	cw_t_nip_004	chara/thumb/mt_nip_00.unity3d	thumb_nip_004
5	0	1	タイプ6	chara/mt_nip_00.unity3d	cw_t_nip_005	chara/thumb/mt_nip_00.unity3d	thumb_nip_005
6	0	1	タイプ7	chara/mt_nip_00.unity3d	cw_t_nip_006	chara/thumb/mt_nip_00.unity3d	thumb_nip_006

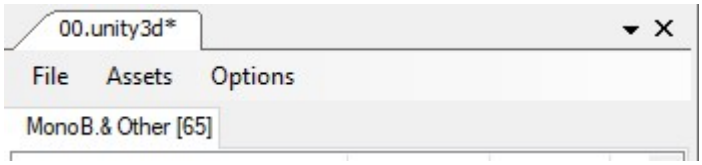
I will describe how to fill in the list file as Excel, but I will show a picture after how it looks in a text editor. On the tab at the top, the number in brackets is the type of item. That goes in A1 in our list file. Just put 0 in A2. Just put 0 in A3. The row of A4 is where all column titles of the list file goes, so copy them from the game's list file. If our EmptyMod actually had nipple textures, we would fill in one row for each of them, but we will leave it empty because our asset bundles are empty. Here is the result.



Unfortunately Excel sometimes acts weird, so you should open it in a text editor when you're done to make sure there is no strange formatting. A newline is a new row, and commas separate columns. So we want it to look like this picture.

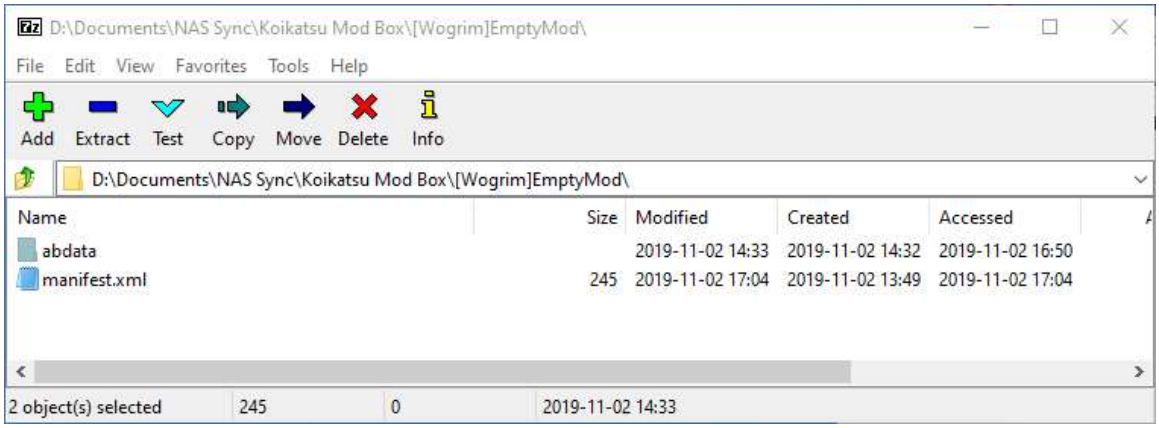


If you do something that modifies an AB, there will be an asterisk by the file name. If it's a game AB make sure to close it without saving so you don't damage your game install.

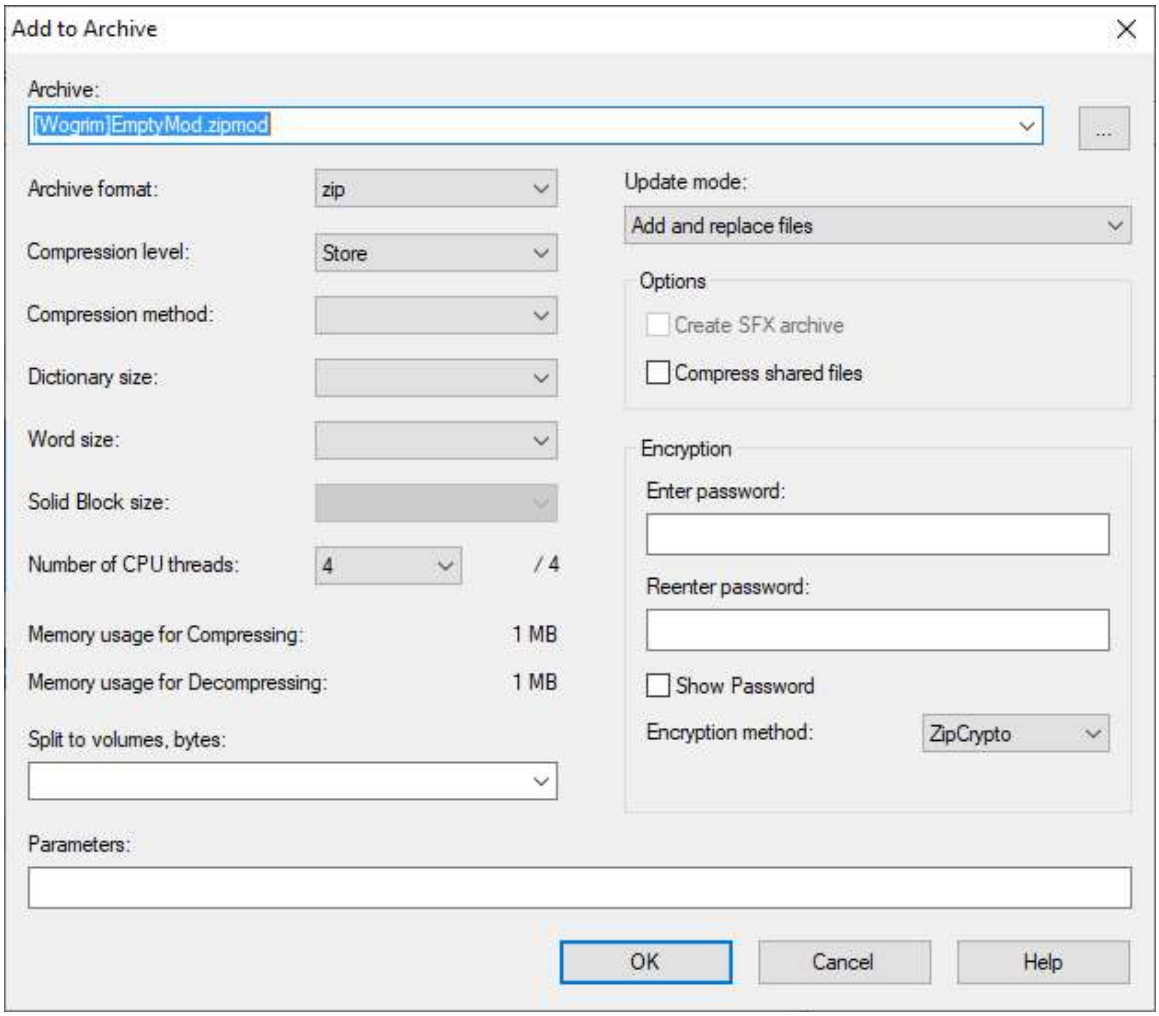


How to Make It a Zipmod?

Now we can zip our EmptyMod. Not that it does us any good, but I'm just showing the technique here so I don't have to explain it again later. Open 7-Zip and navigate to your EmptyMod's root folder. Select the **abdata** folder and the **manifest.xml** file and click "Add".



Make sure Archive format is "zip", Compression level is "Store", then change the extension to **.zipmod** and press OK.



Congratulations! You now have a Zipmod that doesn't do anything! I know it seems like a lot of work to do nothing, but making mods from the EmptyMod is easy.

You can delete the .zipmod, it is just a waste of space.

How to Make the Empty Zipmod Into an Actual Texture Mod?

Duplicate your whole EmptyMod folder. I'm going to change things up and make an Eye (Iris) mod, which I will be calling RobotEyes. Start by changing all of these:

- mod folder name
- manifest guid, mod name, etc.
- list file name (add more list files if the mod will have more than one item type)

If you don't know what assets your item type needs, check the list file. In this case, an Eye has a texture and a thumbnail texture.

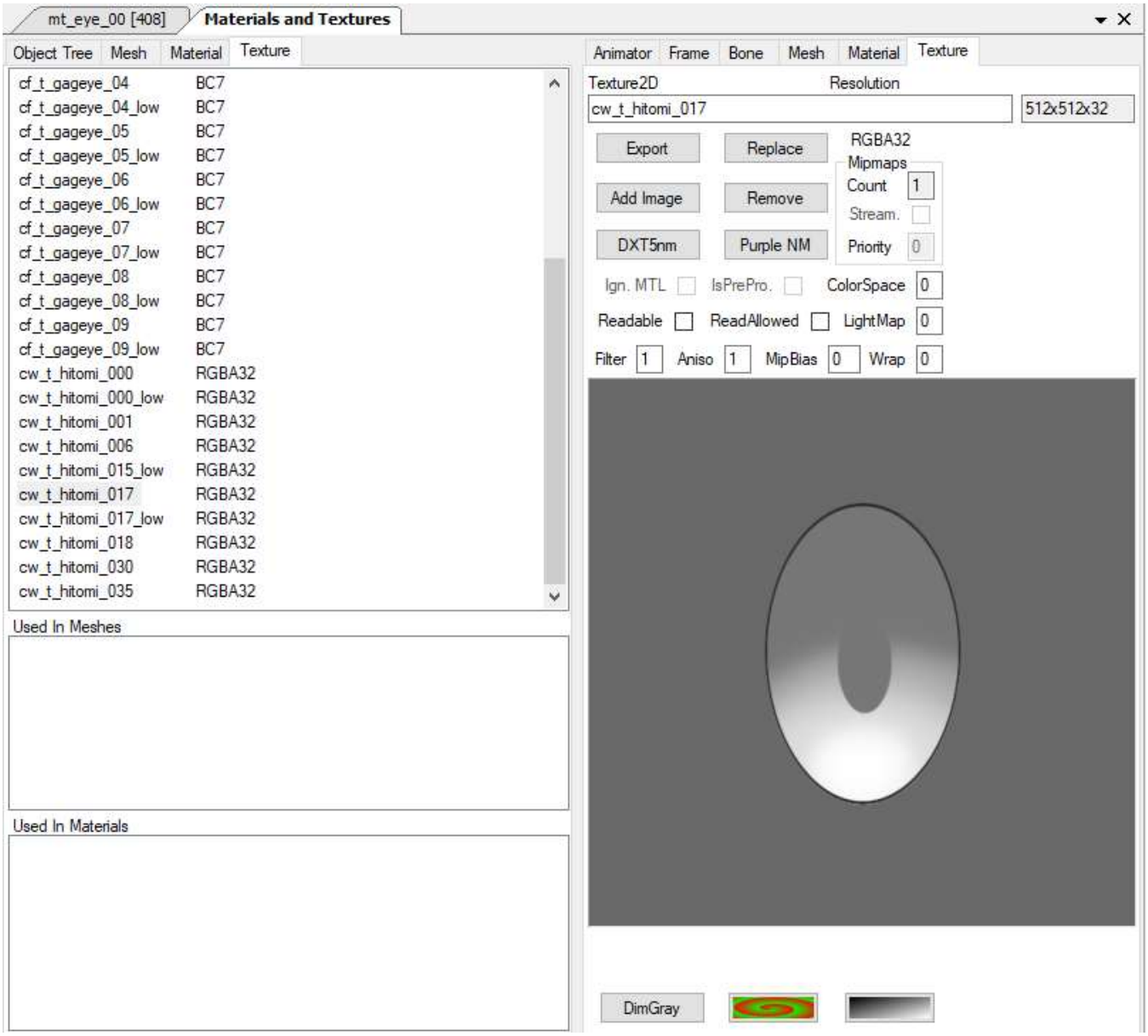
mt_eye_00 [408]

ID	Kind	Possess	Name	MainAB	EyeTex	ThumbAB	ThumbTex
0	0	1	タイプ1	chara/mt_eye_00.unity3d	cw_t_hitomi_000	chara/thumb/mt_eye_00.unity3d	thumb_hitomi_000
1	0	1	タイプ2	chara/mt_eye_00.unity3d	cw_t_hitomi_001	chara/thumb/mt_eye_00.unity3d	thumb_hitomi_001
2	0	1	タイプ3	chara/mt_eye_00.unity3d	cw_t_hitomi_002	chara/thumb/mt_eye_00.unity3d	thumb_hitomi_002
3	0	1	タイプ4	chara/mt_eye_00.unity3d	cw_t_hitomi_003	chara/thumb/mt_eye_00.unity3d	thumb_hitomi_003

While we are looking at the list file, we might as well copy the necessary info to our own list file.

For non-texture items, there will usually be more columns, and some of the items in those columns can reference extra things that are not part of the list file. So as good practice we'll go look at one of these eyes.

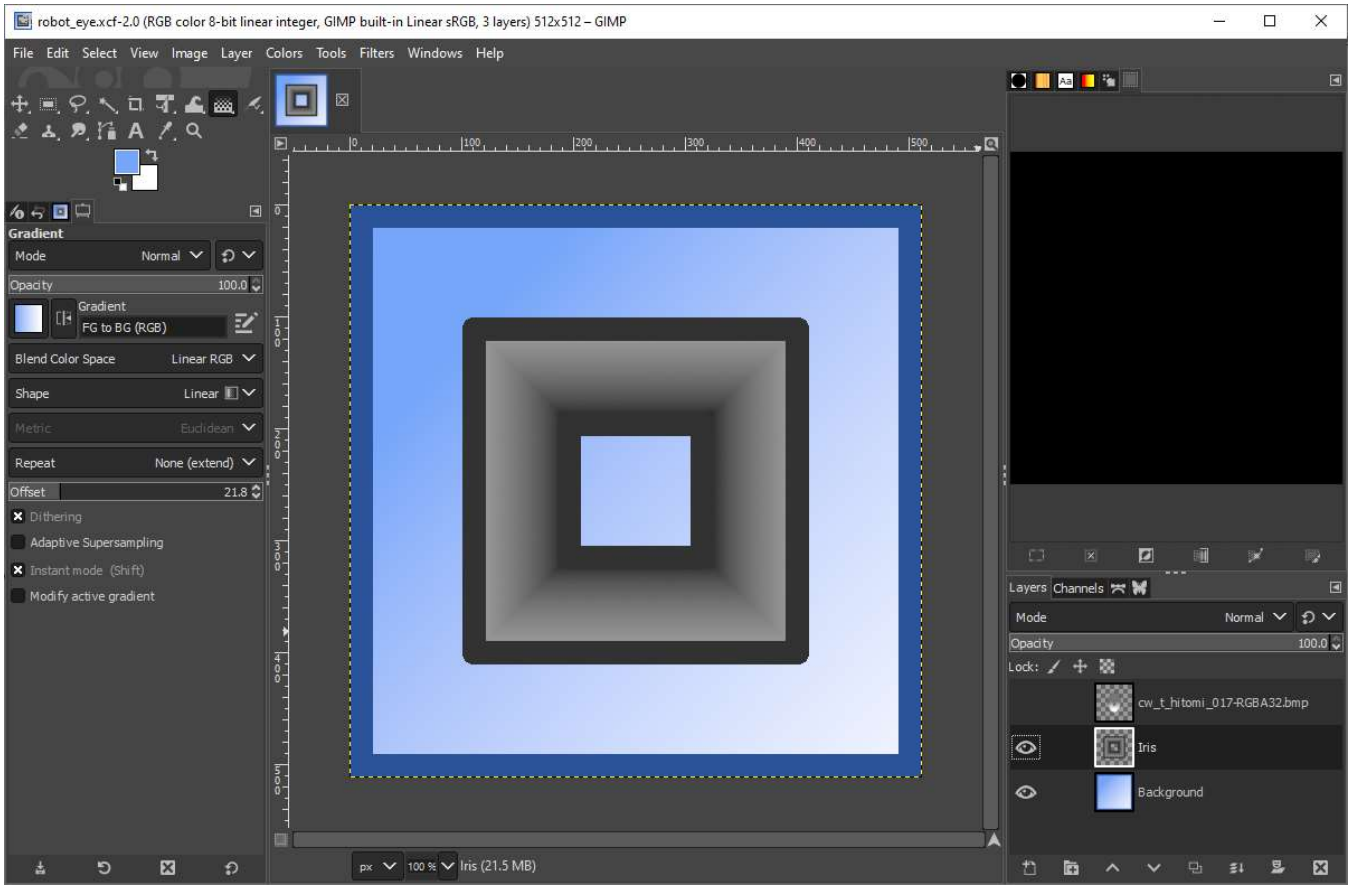
Open the game's asset bundle for the eyes, **abdata/chara/mt_eye_00.unity3d** (as noted in the MainAB column). You should notice that there is a "_low" version of each of them, which is used when roaming in story mode. We must make a _low version of our asset so that our mod works for that mode. For textures, this just means cut the resolution in half (height and width). We will use an existing game asset as reference.



We generally want to match compression, texture resolution, color space, and mipmaps for what we make. We will go with BC7 compression, which is high quality. We will go with 512x512 for texture resolution and 256x256 for the low version. In this case, the texture is in Linear color space and has no mip maps because of the way the game mixes it with the iris colors and gradient. I export the texture to use as reference for making my own texture (note: SB3UGS does not export the color space). This puts it in the game folder, so move it somewhere else so you're not making a mess of your game folder.

As for how exactly the texture works, it can vary greatly from item to item. You can ask around or experiment to find out. In this case, I'm pretty sure it just determines shape and darkness of the iris and pupil.

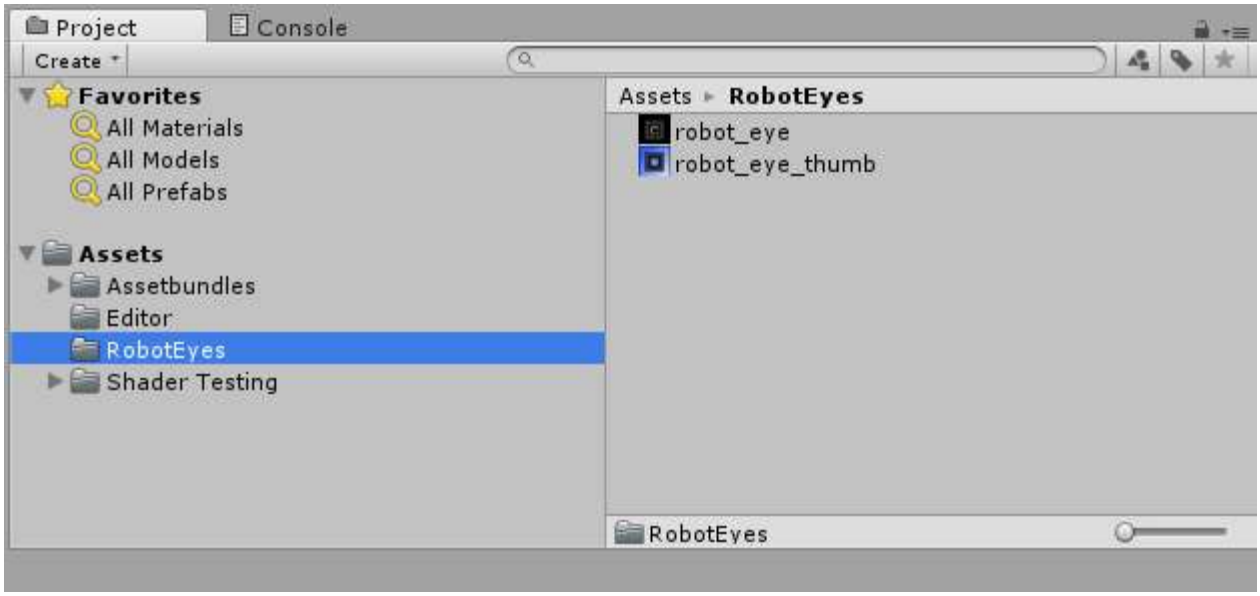
Start up GIMP and make a new 512x512 image in Linear (under Advanced Options => Gamma => Linear light), filled with Transparency. I've made myself a nice weird square eye shape with no pupil in the middle. I've also made a quick background similar to the game one, to use for thumbnail. For some items a screenshot is better.



So I'm exporting a PNG of this with and without the background. Choose 0 compression, because we will compress it in Unity.

Finally the Unity Part

Make yourself a Unity 5.6.2f1 project for modding if you haven't already, and open it. In your Assets folder, create a folder for your mod, and then drag the images you made into it from Explorer. Your Project window should look like this but probably less folders.



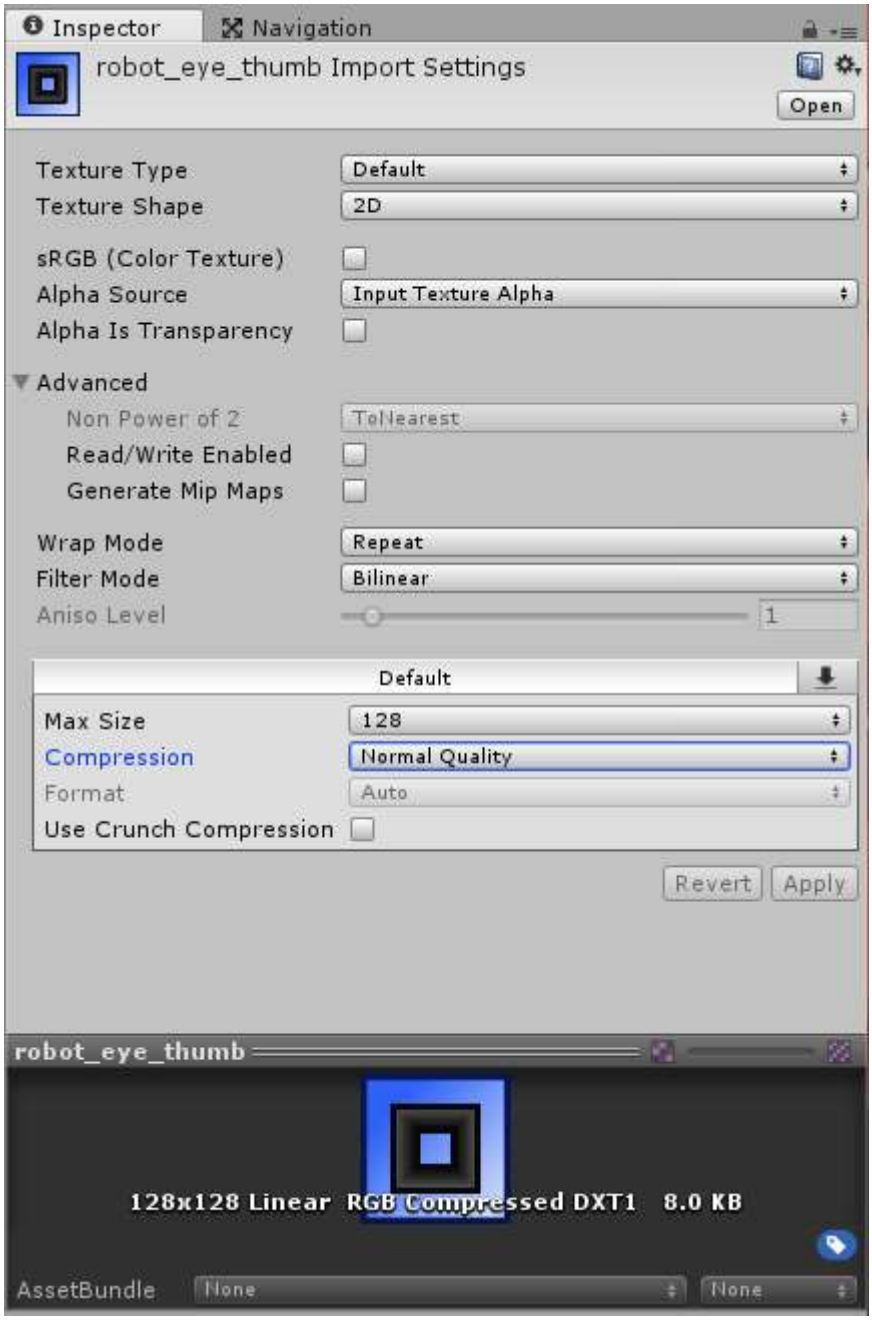
Select the regular one, and do **Ctrl+D** to duplicate it, and rename to be the _low version (robot_eye_low for me). Now for each of these we will look at import settings, but first I need to see how the thumbnail should be, so I look in **abdata/chara/thumb/mt_eye_00.unity3d** to see it's size 128x128. Thumbnails are always no mip maps. Color space is 1 (Gamma) but since I made the image in Linear, an import as Gamma would darken it so I will leave my thumbnail as linear also. If you separately make the thumbnail, Gamma makes more sense. Thumbnails are usually more compressed (less quality) because you don't look at them much.

So when you click on your image in the Project window, the import settings show up in the Inspector window.

For my robot_eye I uncheck "sRGB (Color Texture)" to tell Unity it is Linear. I uncheck "Generate Mip Maps" because we don't need those for this item type. And I change Compression to "High Quality" which will give BC7 compression. Make sure to hit Apply for these changes to take effect.

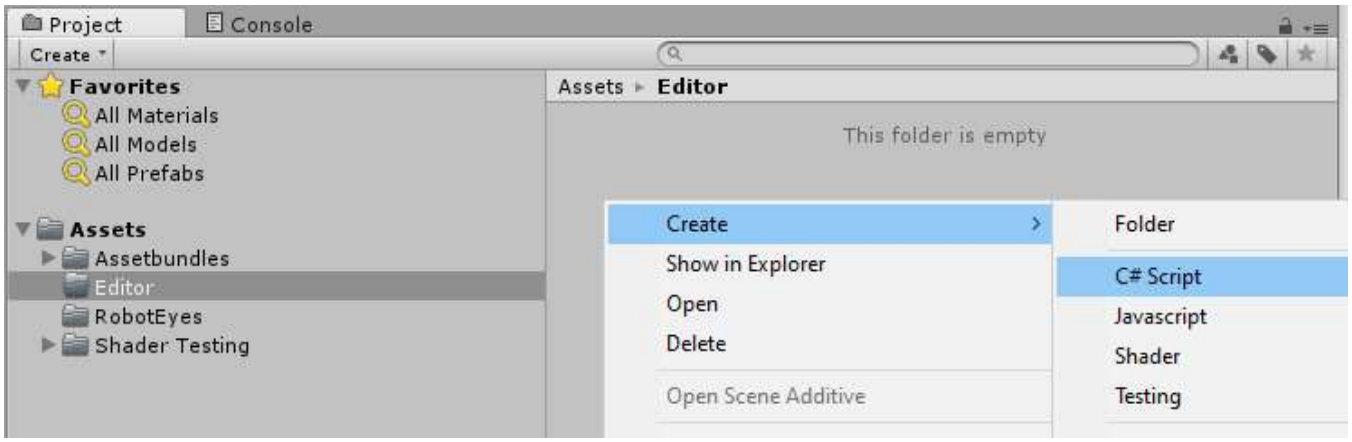
For my robot_eye_low I do all the same things, but also change "Max Size" to 256 so that it is half as big.

For my robot_eye_thumb I do the same things except "Max Size" is 128 and Compression stays on "Normal Quality" which you will see on the preview says DXT1.



You will also see at the bottom it says **AssetBundle**; that is where the asset will be saved when your Unity project is built. So what we're going to do is "create" an AB for thumbnails and an AB for items in this mod. With your thumbnail asset still selected, click where it says "None" (bottom center) and click "New" from the dropdown. You name your thumbnail AB here, but you want it to match the folder structure that you've created for where your thumbnail AB will go, starting after abdata. So in my case, I type in
chara/thumb/wogrim/roboteyes.unity3d
If you use upper case letters it will automatically change to lower case. Now on my "main" texture I create
chara/wogrim/roboteyes.unity3d
and on the _low version I can just navigate to it from the dropdown.

So now that all the textures are assigned to ABs we just have to build them, which is kind of a pain to set up but you only have to do the setup once if you keep using the same Unity Project for all your mods. In your Unity project's Assets folder (in the Project window), you need to create a folder called "Editor" and inside that right click and create a C# script called "CreateAssetBundles" (name doesn't actually matter).

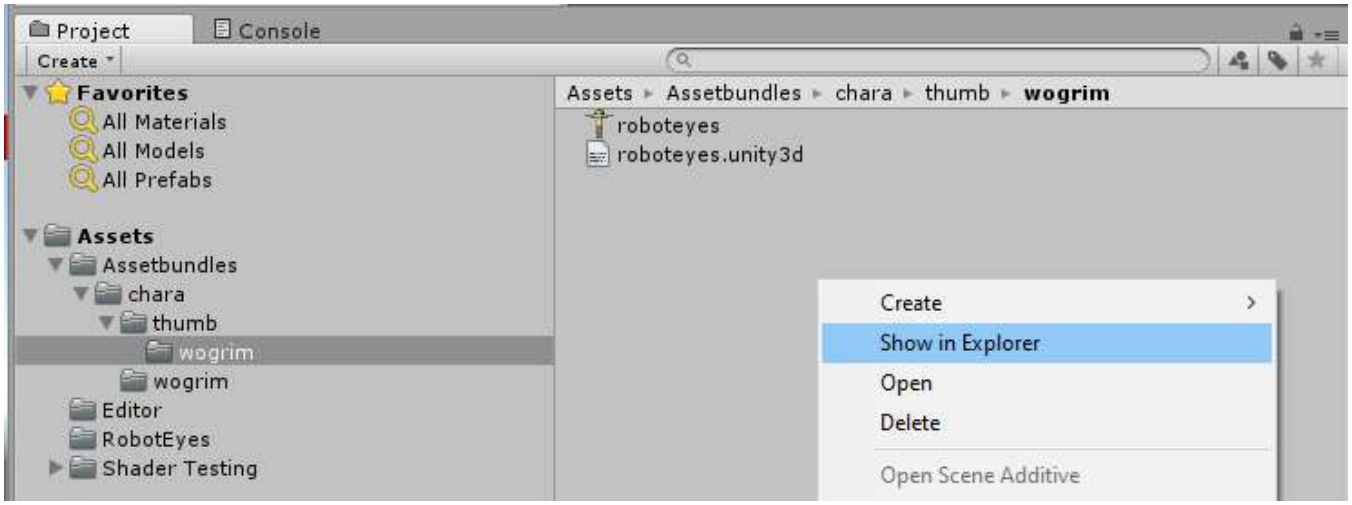


You need to open this up (navigate to it and open with Notepad is fine) and paste some code in there (delete the default code), which originally comes from [Unity - Manual: AssetBundle Workflow (unity3d.com)] but if you copy-paste their version it won't work. So here it is:

```
using System.IO;
using UnityEditor;

public class CreateAssetBundles
{
    [MenuItem("Assets/Build AssetBundles")]
    static void BuildAllAssetBundles()
    {
        string assetBundleDirectory = "Assets/AssetBundles";
        if(!Directory.Exists(assetBundleDirectory))
        {
            Directory.CreateDirectory(assetBundleDirectory);
        }
        BuildPipeline.BuildAssetBundles(assetBundleDirectory,
BuildAssetBundleOptions.ChunkBasedCompression, BuildTarget.StandaloneWindows64);
    }
}
```

(I'm not sure if this will give problems if you're not running 64 bit Windows). This gives you a new Unity menu option: "Assets => Build AssetBundles" which will create the ABs in appropriate subfolders of a new "AssetBundles" folder. Navigate to these in Unity and right click -> "Show in Explorer" and copy them to the corresponding folders of your mod. JUST the ABs, not the extra files that get generated.

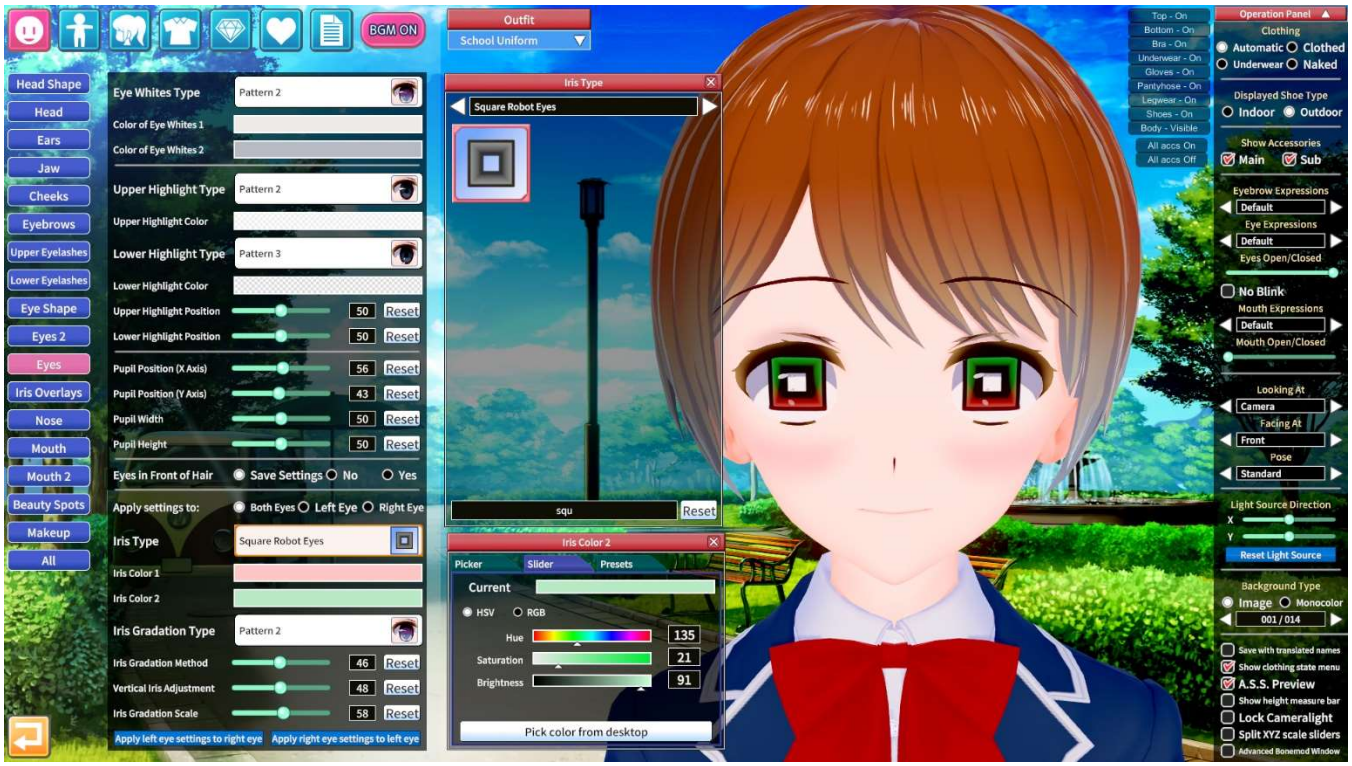


Finishing Up

So, back to the list file. Use the game's list file to help you fill in the columns, just remember to put your file names in the right spots. For each item in the list file, ID must be different, so start with 1 and go up if you have more items. The AB names are the same as in Unity when assigning assets to an asset bundle. If you're having trouble lining up the columns in a text editor, you can temporarily reformat it for editing by replacing commas with newlines so that it's in a vertical format, then switch it back when you are done.

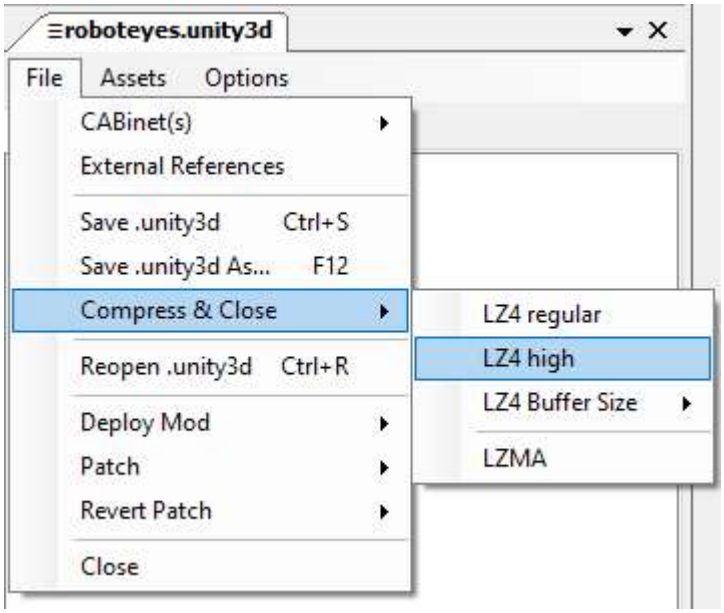
	A	B	C	D	E	F	G	H
1	408							
2	0							
3	0							
4	ID	Kind	Possess	Name	MainAB	EyeTex	ThumbAB	ThumbTex
5	1	0	1	Square Robot Eyes	chara/wogrim/roboteyes.unity3d	robot_eye	chara/thumb/wogrim/roboteyes.unity3d	robot_eye_thumb

Now use 7-Zip to make the zipmod like we did with the EmptyMod, and put it in your mods folder. You should have results, but if you don't, check the troubleshooting steps at the end of this guide. The eyes came out a little big and significantly darker than I wanted, but look okay with some sliders changed.



Compressing the Asset Bundles

If you make a mod you want to release, you want to make sure your ABs are compressed (before the zip step) to make the zipmod smaller. The script that builds the asset bundles in Unity does an LZ4 compression so you should be good to go, but if you use SB3UGS to edit your asset bundle, it gets uncompressed and you have to do this to recompress: open the AB in SB3UGS and go to "File => Compress & Close => LZ4 high".



How to Make Something More Than a Texture Item?

There are guides for some item types, and a lot of knowledge if you ask in **#mod-modelling**. Start with easier items and work your way up, you will learn important things so you can be successful with harder items.

How to Solve Common Zipmod Problems?

I can't find any of my items to select!

- There may be a conflict with the GUID in your manifest (or you messed up the formatting part)
- You may have zipped your mod wrong (if you extract it, manifest should be in root folder)
- You may have put your list file(s) in the wrong location (must be in "abdata/list/characustom" or subfolder)
- Your list file is super messed up (check it very carefully)

I can select my item but nothing shows up!

- You may have mistyped an asset bundle path or asset name in the list file
- There may be some kind of problem with your textures such as they’re not in the AB
- 3D items: it may be so small or so large you can't see it, or something with the shader