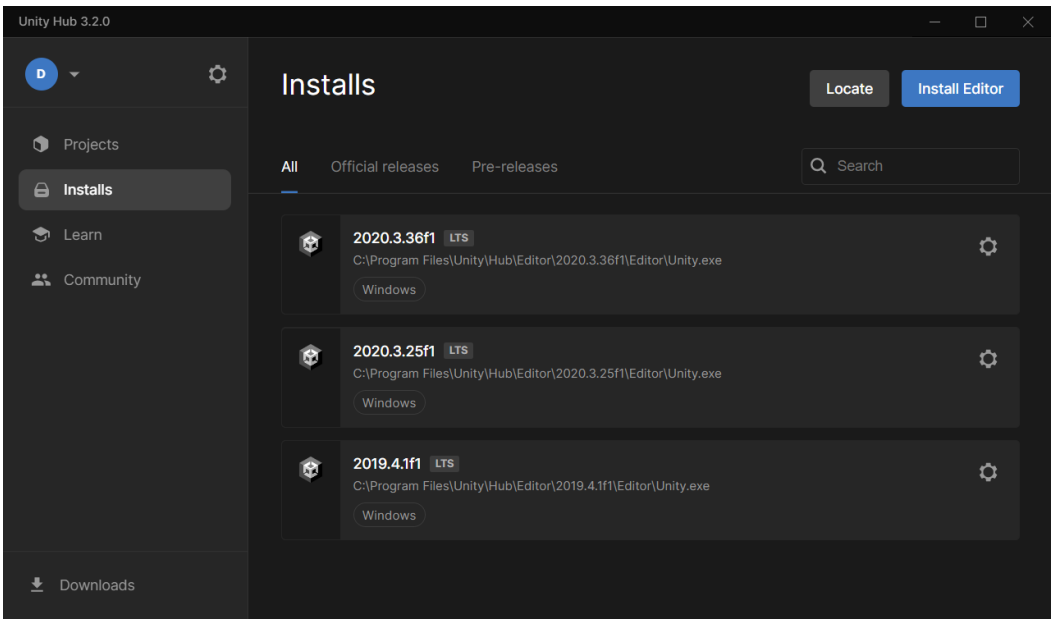
Old World Graphics Modding Guide

Installing Unity

Old World uses the Unity game engine so it is necessary to install the correct version of Unity in order to modify the graphics of the game. To do this:

1) Download Unity Hub from here and install it.
2) Next we need to check the current version of Unity being used by Old World since the version is updated from time to time. Locate the log file ../Documents/My Games/OldWorld/Logs/output.txt (this file should exist assuming you have run the game at least once!). The Unity version can be seen a few lines from the top of the file e.g. `Unity Version: 2020.3.25f1`
3) Now install the relevant Unity version via the Unity Hub.



Extracting Unity Assets

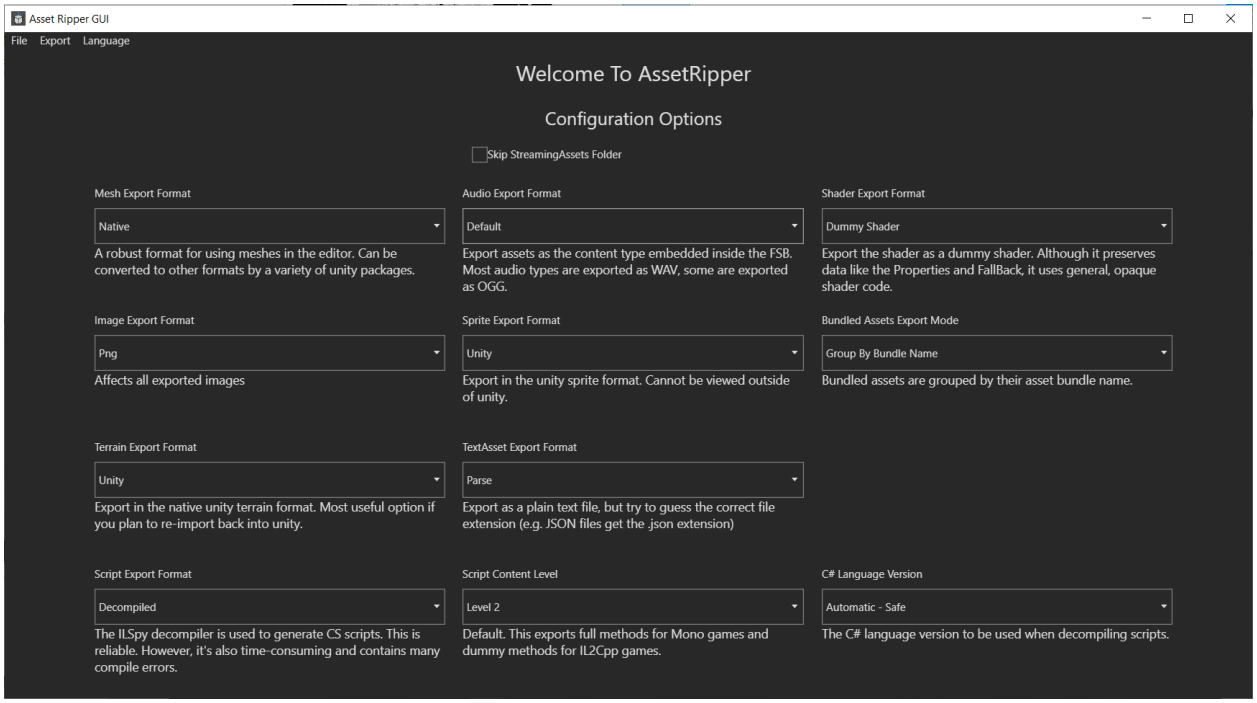
Old World's graphical assets are built into Unity bundles which cannot be inspected in their raw form. To be able to inspect the graphics and understand how they are put together we first need to extract them from the bundles. There are a number of utilities that aim to extract Unity assets from bundles, but the most useful (at time of writing) are Asset Ripper and Asset Studio.

Asset Ripper attempts to reconstruct the entire Unity project that was used to build the game. Asset Studio is more limited in the range of assets that can be extracted but still has its uses especially when we get to 3D model extraction.

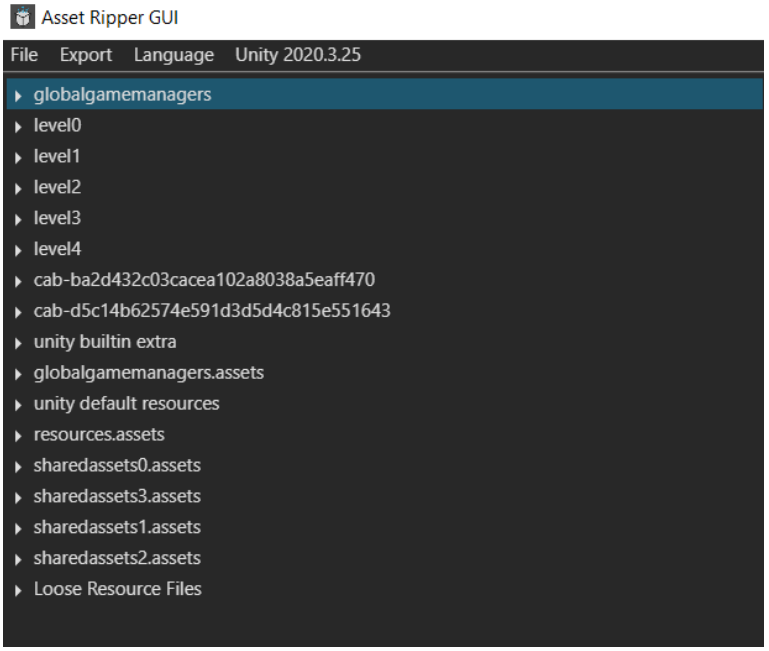
Using Asset Ripper to create an Old World asset library project

The following steps describe how to use Asset Ripper to extract the game assets from the Unity bundles and create a Unity project so they can be viewed and modified.

1. Download AssetRipperGUI from the latest release and unzip the files.
2. Run AssetRipper.exe.
3. You can leave all options as the defaults on the config page:

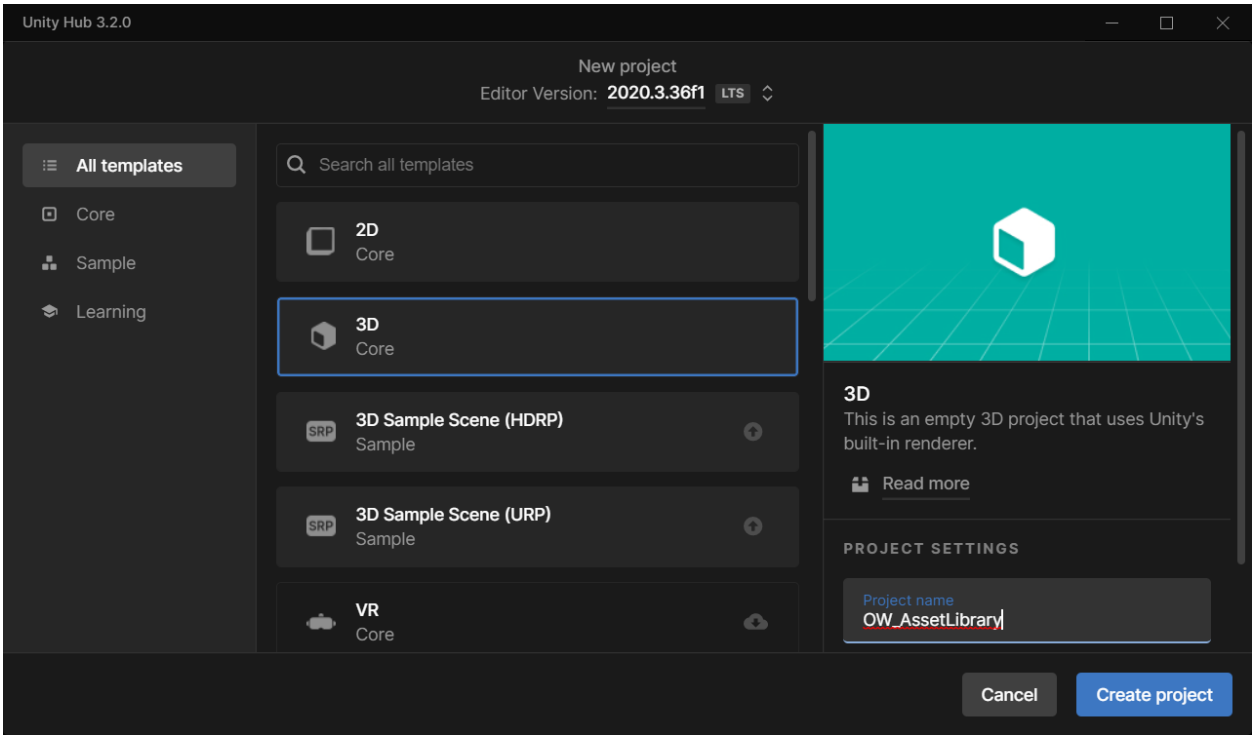


4. From the menu select **File > Open Folder** and select the folder \Old World\OldWorld_Data (this can be found under C:\Program Files (x86)\Steam\steamapps\common if you have the Steam version).
5. After a few moments you should see this top level list of components:



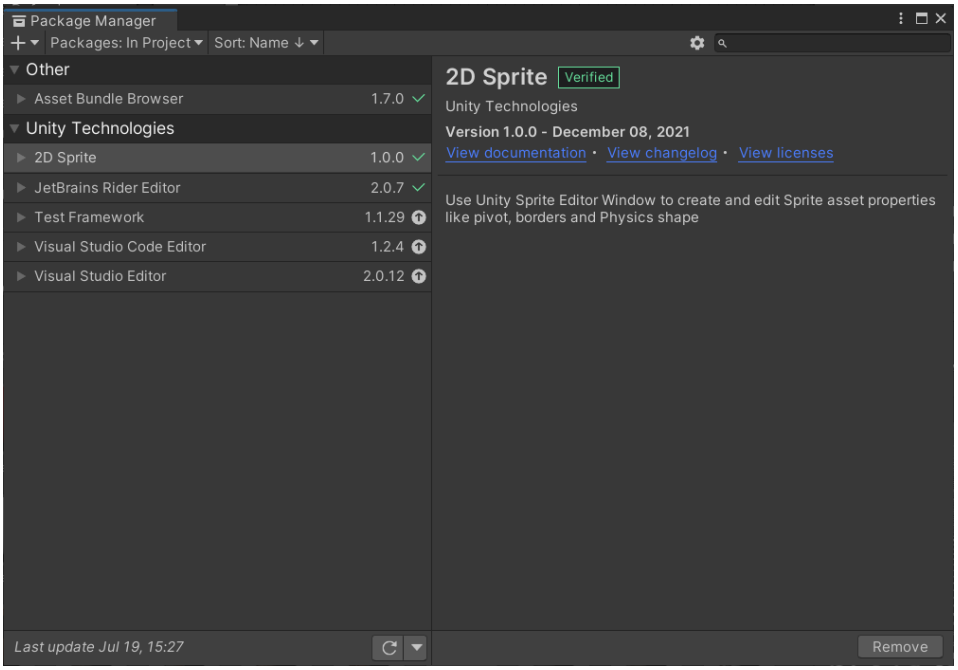
6. From the menu select **Export > Export all Files** and navigate to an appropriate directory where you want to extract all the Unity assets to. The extraction takes quite a long time to complete so be patient. You can track progress in the AssetRipper.exe terminal window.

7. Next we create a new Unity project that will allow us to view and modify the extracted assets. Create a new project using the correct Unity version and the 3D Core template. You can call it **OldWorld_AssetLibrary** or something similar.



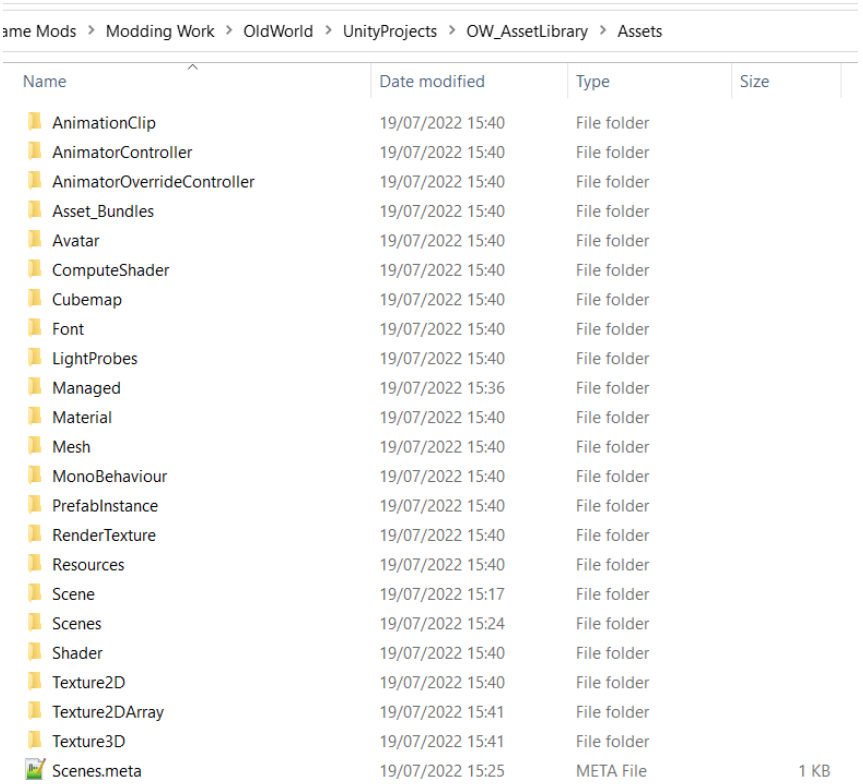
8. When your new project opens go to **Window > Package Manager** in the menu. Remove the packages TextMeshPro, Timeline, Unity UI and Version Control. Switch the **Packages: In Project** dropdown to **Unity Registry**. Install the package 2D Sprite. Finally, using the **+** dropdown in the top left, select **Add package from git URL** and enter the URL `https://github.com/Unity-Technologies/AssetBundles-Browser.git` to install the AssetBundle Browser.

Your installed packages should look like this when you are done:

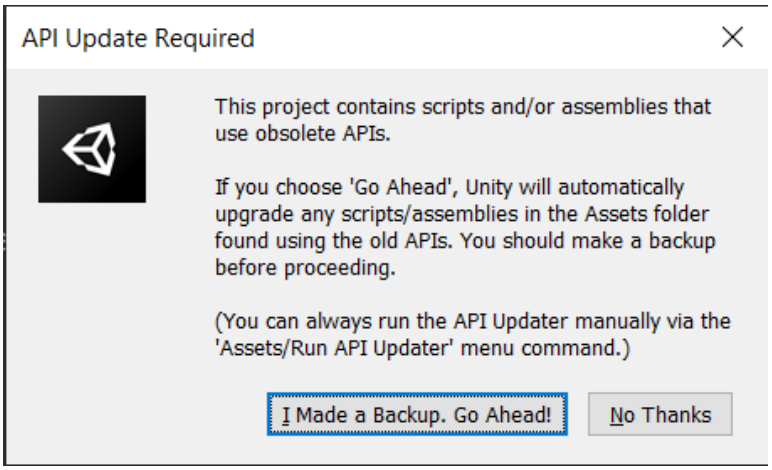


9. Close the Unity project and locate the folder **../Old World/OldWorld_Data/Managed**. Copy this folder to the Assets subdirectory of your Unity project. Delete **mscorlib.dll** from this copied Managed folder.

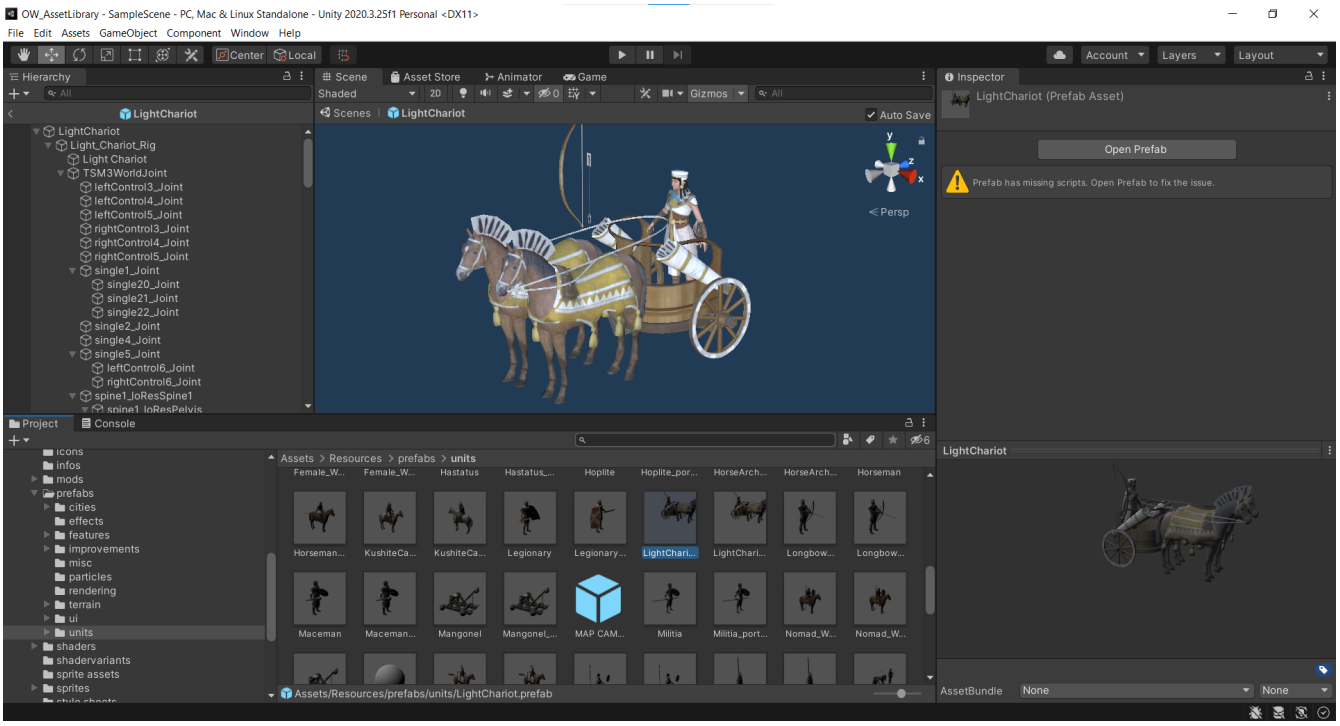
10. Navigate to your Asset Ripper output directory. Copy all the subdirectories under Old World/Exported Project/Assets except for **MonoScript** to within the Assets folder of your Unity project. Your project Assets folder should look something like this after this step:



11. Now re-open the Unity project. Just say **No Thanks** if you get this popup:



12. Unity will take several minutes to import all the assets so go and make another cup of tea. Once the project is open you can start to browse and inspect the assets. The Unit prefabs (which assemble the elements of each 3D Unit) can be viewed under Assets > Resources > prefabs > units for example.



Limitations of Asset Ripper process

While creating an Asset Library project using Asset Ripper is highly useful for reference purposes there are some significant limitations to the process.

1) **MonoScripts** Many of the Old World assets rely on C# scripts for their creation. Asset Ripper attempts to decompile all the C# MonoScripts and dependencies but it does not do so accurately - it is an extremely complex task due to the complexity of the code and the number of first and third party libraries involved. This means that some assets cannot be modified at present. For example, the graphical Terrain

tiles do not render in Unity and cannot be modified currently. Animated Unit graphics are possible (see later chapter) but cannot be linked to visual FX or sound. Static 3D Improvements, Wonders and Resources can be added but they cannot modify the terrain heightmap or block clutter because those things require MonoScripts to be in place and correctly compiled.

2) **AnimatorControllers** These are not accurately extracted by Asset Ripper so the originals are needed to get Unit animation (for example) working correctly.

2D Graphics: Adding Single Image Assets

This section will walk through adding a single 2D image Sprite to Old World. In this example we'll add a character select background image.

1. Create a Unity project (if you have not done so already).

2. Make sure the **2D Sprite** and **AssetBundle Browser** packages are added in Window > Package Manager. If 2D Sprite is not yet added switch the **Packages: In Project** dropdown to **Unity Registry**. Install the package 2D Sprite. You can use the **+** dropdown in the top left, select **Add package from git URL** and enter the URL `https://github.com/Unity-Technologies/AssetBundles-Browser.git` to install the AssetBundle Browser.

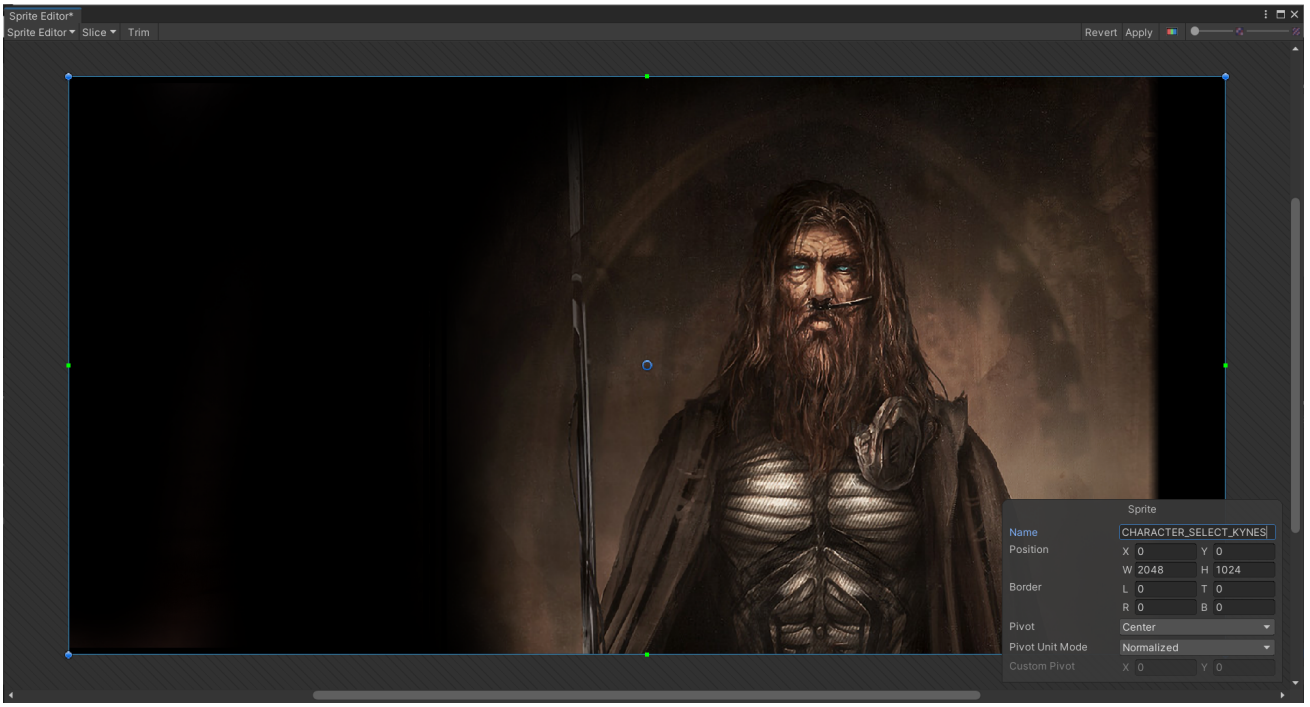
3. In Edit > Project Settings > Editor locate the Sprite Packer Mode setting and switch it to **Sprite Atlas V1 - Always Enabled**.

4. Now create your image using whatever image editing software you prefer and **save it in PNG forma**t. In our example we'll use this image which is 2048 x 1024 like the base game character select portraits. (Artwork by Mark Molnar)

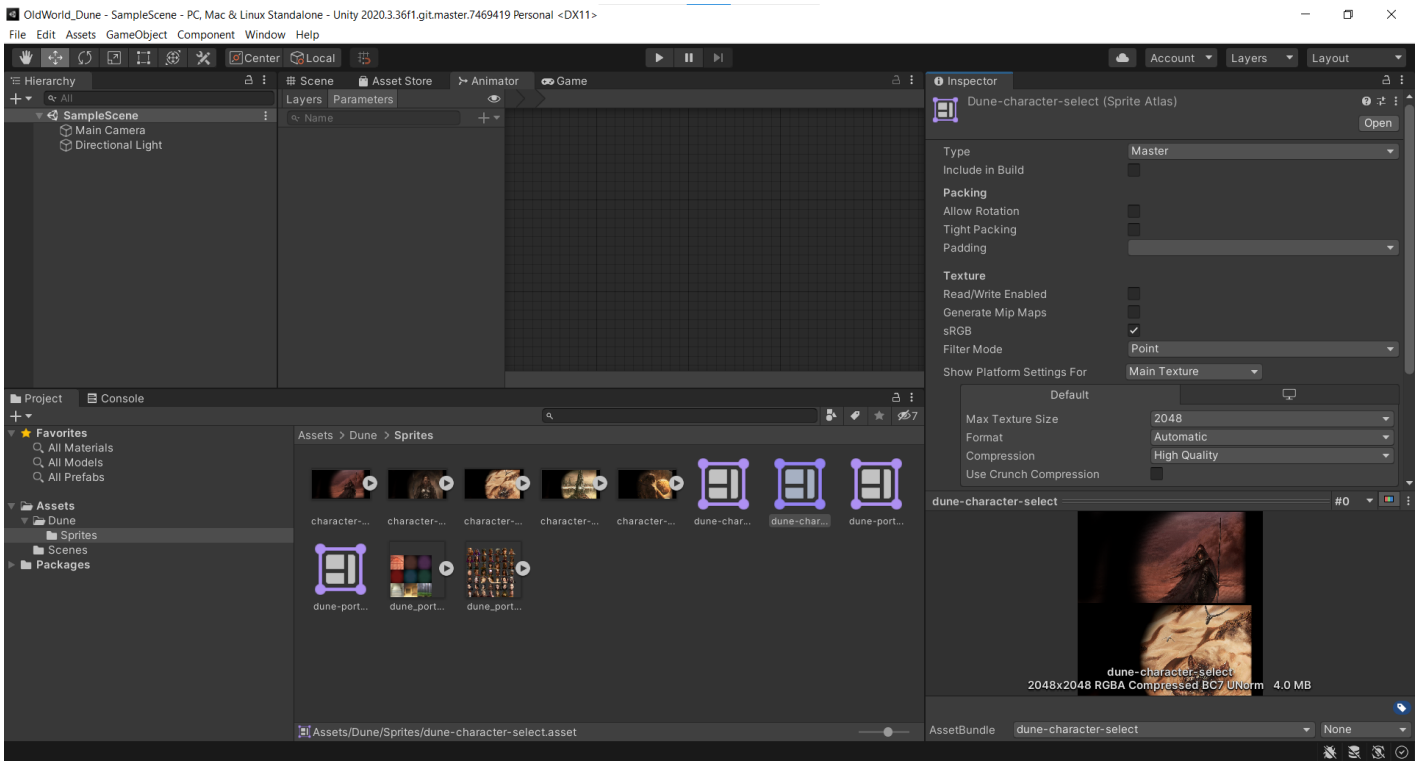
5. Drag and drop your image into the central bottom panel of your Unity project. Alternatively you can use **Assets > Import New Asset…** and then select your PNG file to import. *Tip: you can create a folder structure however you like under Assets to keep things organized.*

6. Change the following settings on your image asset: **Setting Type** to **Sprite (2D and UI)**, **Sprite Mode** to **Multiple** (yes, use Multiple even for a single image), **Compression** to **None**.

7. Open the Sprite Editor and drag a select box over the entire image then name the Sprite with the reference we want to use in our XML file. In this example, I've used CHARACTER_SELECT_KYNES:

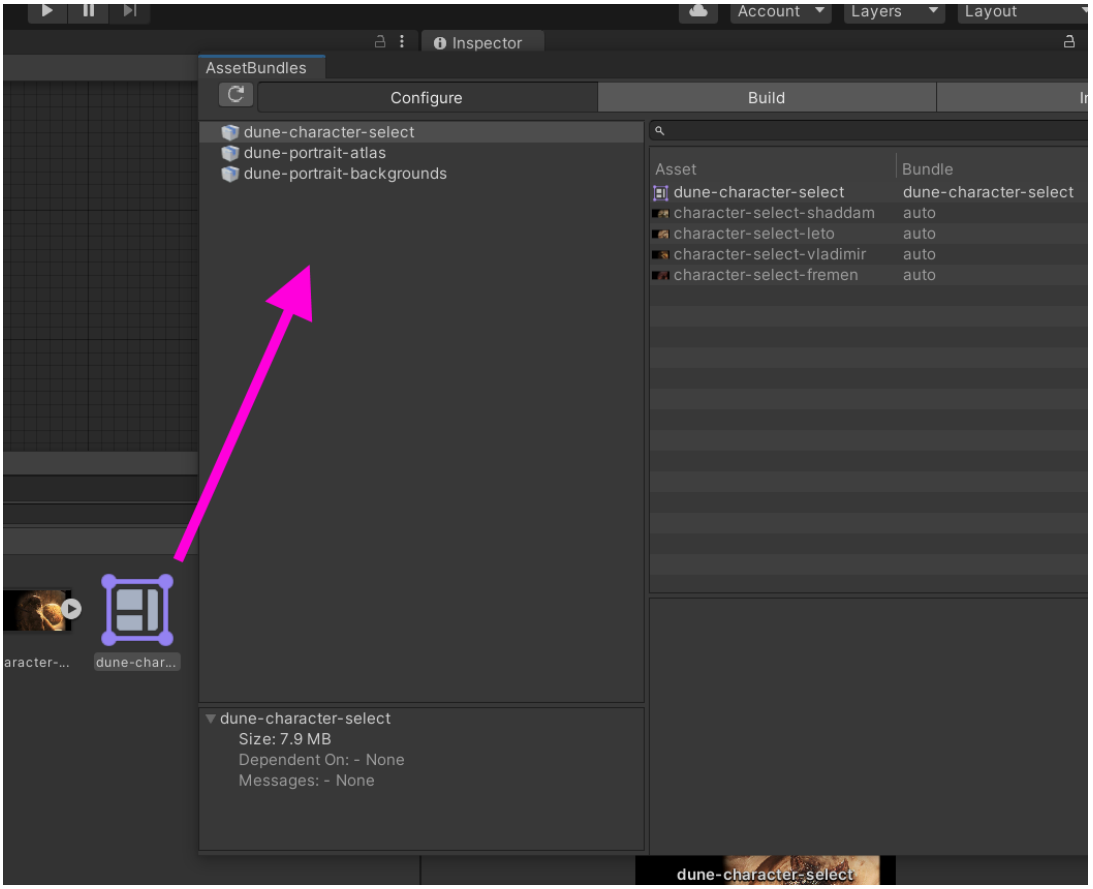


8. Now we are ready to add our Sprite to a Sprite Atlas. The best way to create a Sprite Atlas is to duplicate an existing Sprite Atlas either from the extracted Asset Library project or you can use the [Dune example Unity project here on Github] for example taking the dune-character-select.asset and dune-character-select.asset.meta files and copying them into your Unity project outside of the Unity application. If you have an existing Sprite Atlas in your project you can use Ctrl-D to Duplicate it and then rename it to the name you want it to have. *(The reason for copying an existing Sprite Atlas rather than just creating a new one in Unity is that there are some settings seem to be different with the extracted Old World Sprite Libraries to what can be created through the user interface - for example Padding is blank which cannot be achieved through the UI - if Padding is not blank the Sprite Atlases do not work in game.)*



9. To add our Sprite to the Sprite Atlas scroll down the panel on the right until you can see **Objects for Packing**. Click the **+** and select the Sprite you created in step 7. You can click Pack Preview to Preview how the Sprite Atlas is going to pack your Sprite (This becomes more relevant when adding multiple Sprites to an Sprite Atlas.)

10. Next open Window > Asset Bundle Browser and drag and drop the Sprite Atlas into the left hand panel.



This takes care of creating an Assets Bundle and adding your Sprite Atlas to it.

11. Switch to the Build tab and select a suitable Output Path. If you have a mod created then you can build directly to the Assets subfolder of your mod. Check the Clear Folders and Force Rebuild checkboxes to clear out any existing files and make sure everything is recreated on each build. Finally go ahead and click Build to build out your Asset Bundle.

12. Now we're ready to make the Infos XML changes to integrate our sprite. First we need to add the reference to our Sprite Atlas asset in our asset-add.xml:

```
<Root>
  <Entry>
     <zType>ASSET_SPRITE_SHEET_DUNE_CHARACTER_SELECT</zType>
     <zAsset>dune-character-select/Dune/Sprites/dune-character-select</zAsset>
  </Entry>
</Root>
```

To determine the zAsset path we can look at the Asset Bundle manifest file created by the build process, in this example dune-character-select.manifest. We can check the path to our Asset towards the bottom:

```
Assets:
- Assets/Dune/Sprites/dune-character-select.asset
```

If we substitube the Asset Bundle name for "Assets" and remove the .asset file extension we get the zAsset path we need for asset-add.xml.

13. Next we need to append our Sprite Atlas to the relevant Sprite Group which can be discovered by looking through Old World/Reference/XML/Infos/spriteGroup.xml. We need to add the following to spriteGroup-append.xml:

```
<Root>
  <Entry>
        <zType>SPRITE_GROUP_CHARACTER_SELECT_BACKGROUNDS</zType>
        <aeSpriteSheets>
              <zValue>ASSET_SPRITE_SHEET_DUNE_CHARACTER_SELECT</zValue>
        </aeSpriteSheets>
  </Entry>
</Root>
```

14. We can reference our sprite image from character-add.xml using the name we gave it in step 7 like this:

```
<Entry>
  <zType>CHARACTER_KYNES</zType>
  <Gender>GENDER_MALE</Gender>
  <FirstName>NAME_KYNES</FirstName>
  <PreferredPortrait>CHARACTER_PORTRAIT_KYNES</PreferredPortrait>
  <Title>TITLE_LIET</Title>
  <PlayerDynasty>DYNASTY_FREMEN</PlayerDynasty>
  <Father>CHARACTER_PARDOT</Father>
  <Spouse>CHARACTER_FAROULA</Spouse>
  <CharacterSelectPortrait>CHARACTER_SELECT_KYNES</CharacterSelectPortrait>
  <iAge>35</iAge>
  <aeTraits>
     <zValue>TRAIT_SCHEMER_ARCHETYPE</zValue>
     <zValue>TRAIT_RUTHLESS</zValue>
  </aeTraits>
</Entry>
```

15. Now we can start Old World, activate our mod and test in game. All being well you're image should appear like this:

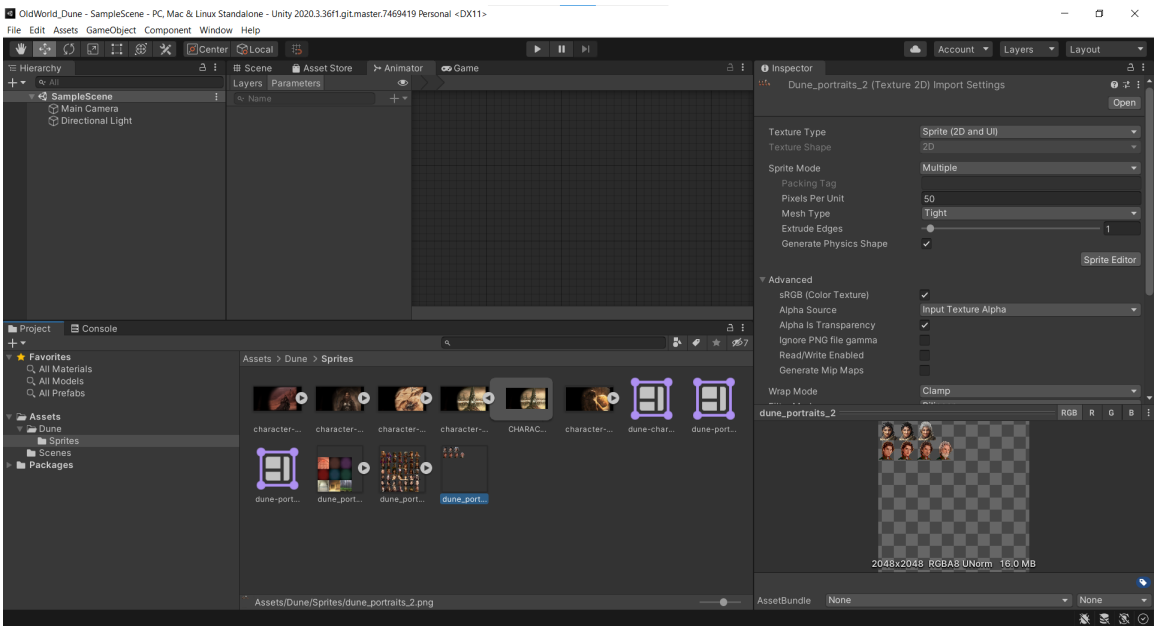
If you want to refer to a complete example you can find the work-in-progress Dynasties of Dune mod and its Unity project here on Github.

2D Graphics: Adding Multi-Image Sprites

We have introduced most of the information needed to create multi-image sprites in the previous section. In this section we will walk through adding more character portraits to the game using multi-image Sprites.

1. Create your multi character portrait image. The cleanest way to do this is to create a 2048 x 2048 image to be your portrait atlas and lay out your portraits in a 7 x 7 grid of 260 x 260 pixel individual portraits. When we mark out the bounding boxes for our character portrait sprites they will be 256 x 256 pixels so we are creating them with a 2 pixel buffer.

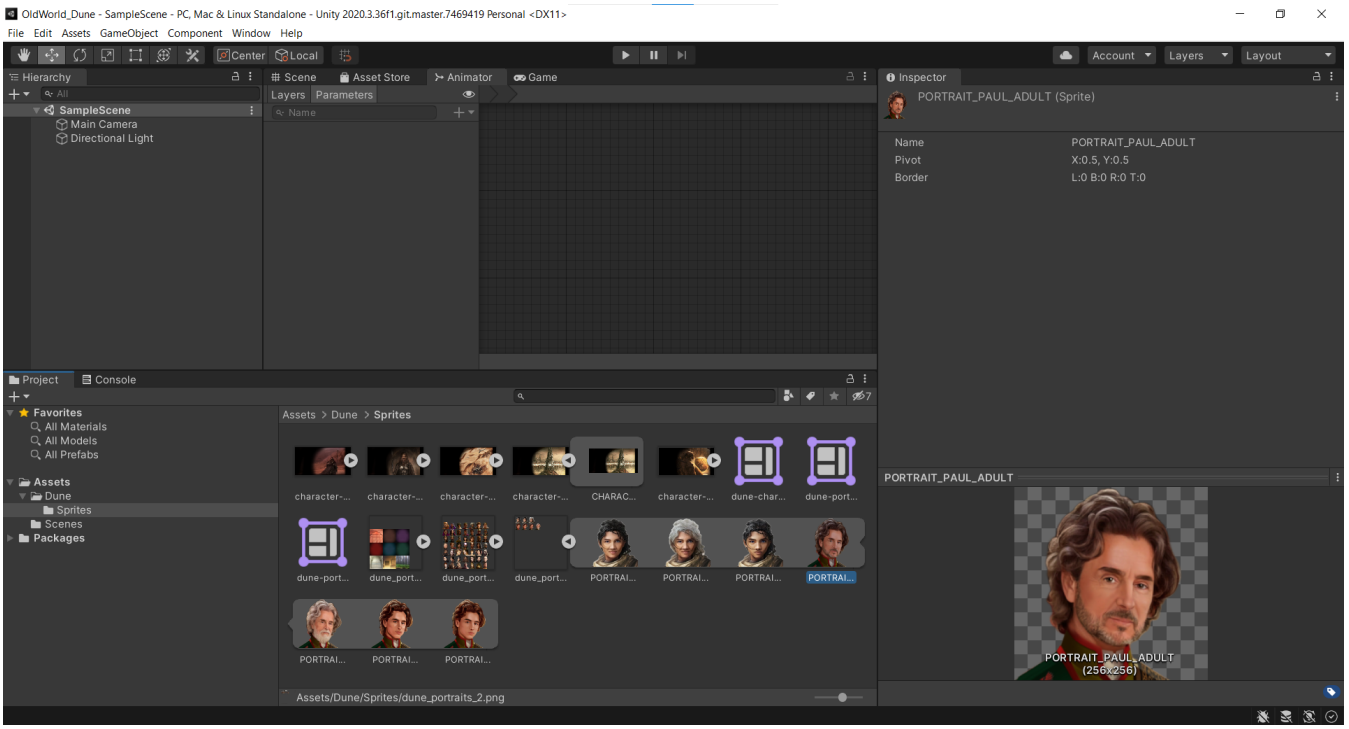
2. Add the Sprite Image to your Unity project as in the previous section. The only different to the character select portraits is that the Pixels Per Unit should be set to 50 rather than 100 to be consistent with the base game assets.



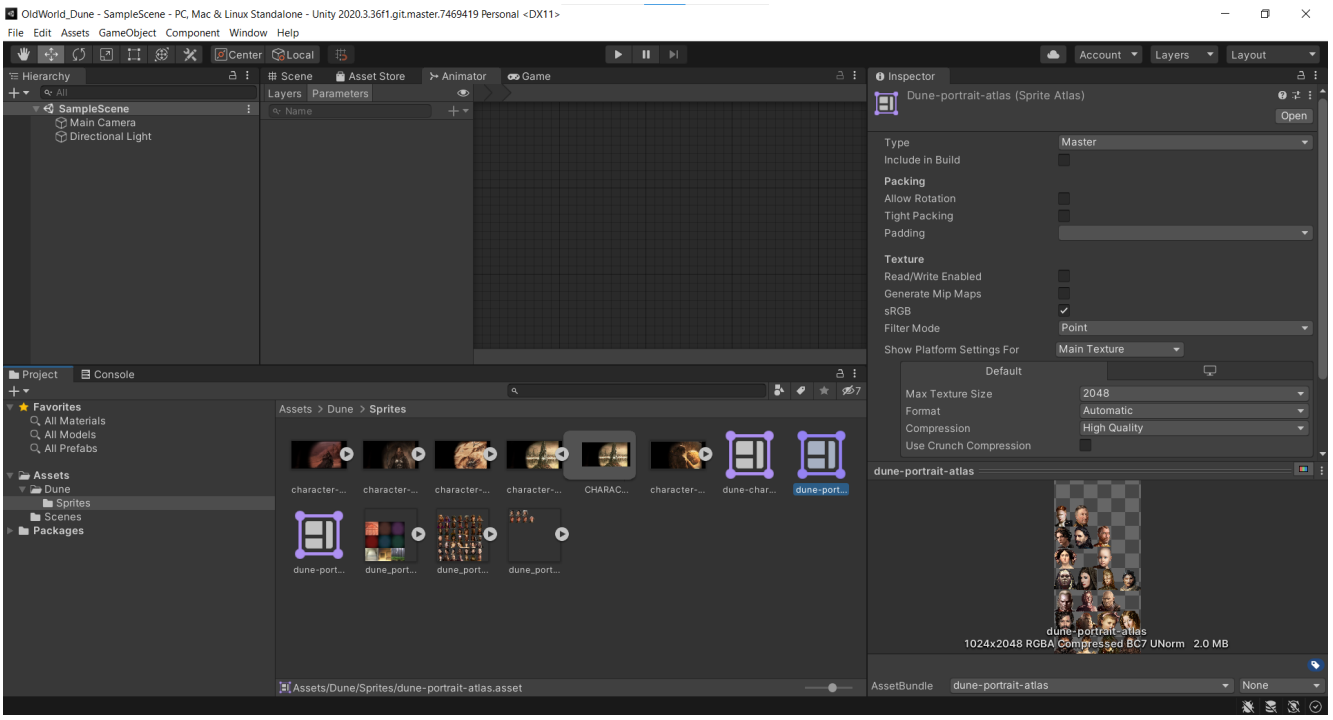
3. Open the Sprite Editor using the button on the left and use LMB-Drag to drag an initial bounding box over one of your portraits. You can use the text box to set both W and H to 256 and then position the box so it is 2 pixels from the edge of the 260 x 260 portrait. You can use the mouse scroll wheel to zoom in and out and MMB-Drag to pan. Finally give the Sprite a name you will use to reference it from characterPortrait-add.xml.

4. Next use CTRL-D to duplicate the first bounding box and position over the next portrait, name it and repeat until you have created bounding boxes and named all your sprites. Once you are all done click **Apply** in the top right.

5. Back in the main Unity window you can expand the arrow on your multi sprite image to view the individual sprites you have marked out and check them.



6. Create a Sprite Atlas, add it to an Asset Bundle and build it out as described in steps 8 to 10 in the previous section. You can just add the multi sprite image to the **Objects for Packing** in the Sprite Atlas and all of the sprites it contains will be included.



7. The Infos XML changes to integrate your character portraits are as follows:

asset-add.xml

```
<Root>
  <Entry>
     <zType>ASSET_SPRITE_SHEET_DUNE_CHARACTER_PORTRAITS</zType>
     <zAsset>dune-portrait-atlas/Dune/Sprites/dune-portrait-atlas</zAsset>
  </Entry>
</Root>
```

spriteGroup-append.xml

```
<Root>
  <Entry>
        <zType>SPRITE_GROUP_CHARACTER_PORTRAITS</zType>
        <aeSpriteSheets>
              <zValue>ASSET_SPRITE_SHEET_DUNE_CHARACTER_PORTRAITS</zValue>
        </aeSpriteSheets>
  </Entry>
</Root>
```

characterPortrait-add.xml

```
<Root>
<Entry>
 <zType>CHARACTER_PORTRAIT_PAUL</zType>
 <Gender>GENDER_MALE</Gender>
 <azAgeGroupSpriteNames>
   <Pair>
     <zIndex>CHARACTER_AGE_GROUP_BABY</zIndex>
     <zValue>GENERIC_BABY_07</zValue>
   </Pair>
   <Pair>
     <zIndex>CHARACTER_AGE_GROUP_YOUTH</zIndex>
     <zValue>PORTRAIT_PAUL_YOUTH</zValue>
   </Pair>
   <Pair>
     <zIndex>CHARACTER_AGE_GROUP_TEEN</zIndex>
     <zValue>PORTRAIT_PAUL_TEEN</zValue>
   </Pair>
   <Pair>
     <zIndex>CHARACTER_AGE_GROUP_ADULT</zIndex>
     <zValue>PORTRAIT_PAUL_ADULT</zValue>
   </Pair>
   <Pair>
     <zIndex>CHARACTER_AGE_GROUP_SENIOR</zIndex>
     <zValue>PORTRAIT_PAUL_SENIOR</zValue>
   </Pair>
 </azAgeGroupSpriteNames>
</Entry>
</Root>
```

character-add.xml

```
<Entry>
  <zType>CHARACTER_PAUL</zType>
  <Gender>GENDER_MALE</Gender>
  <FirstName>NAME_PAUL</FirstName>
  <PreferredPortrait>CHARACTER_PORTRAIT_PAUL</PreferredPortrait>
  <PlayerDynasty>DYNASTY_ATREIDES</PlayerDynasty>
</Entry>
```

8. Once all the data is updated and the asset bundle is in place in your mod, your character portraits should now show up in the game.

3D Graphics: Adding a Static Improvement

3D Graphics: Adding an Animated Unit

updated by Deliverator 2nd August 2022