Tilegen: Tileset Generator & Automatic Rule Tiles v0.9

By Igor Hatakeyama

If you have any questions or suggestions please contact me at igor.hatake@gmail.com

If you like the asset please leave a review on the asset store!

What does this asset do, exactly?

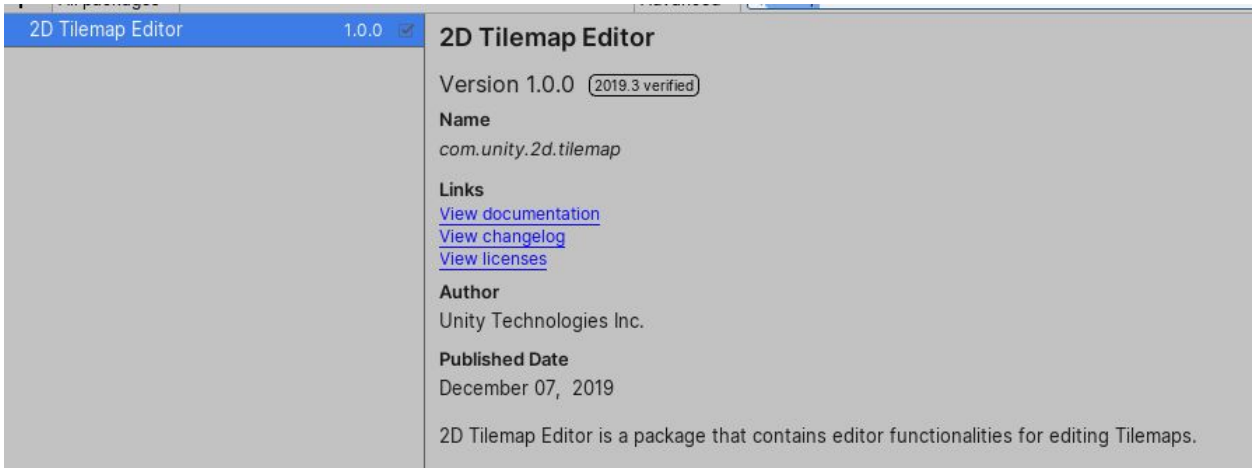
The asset works in two parts. First, it takes a base tileset (containing 13 tiles) as an input and compiles it into a bigger tileset containing many variations of tiles that the user would otherwise have to draw by hand. It does that by taking information already present in the base tileset and creating new tiles using that information.

After the tileset is done, the algorithm automatically creates a rule tile using the tiles from your brand new tileset which can then be used to paint the tile freely on a grid tilemap.

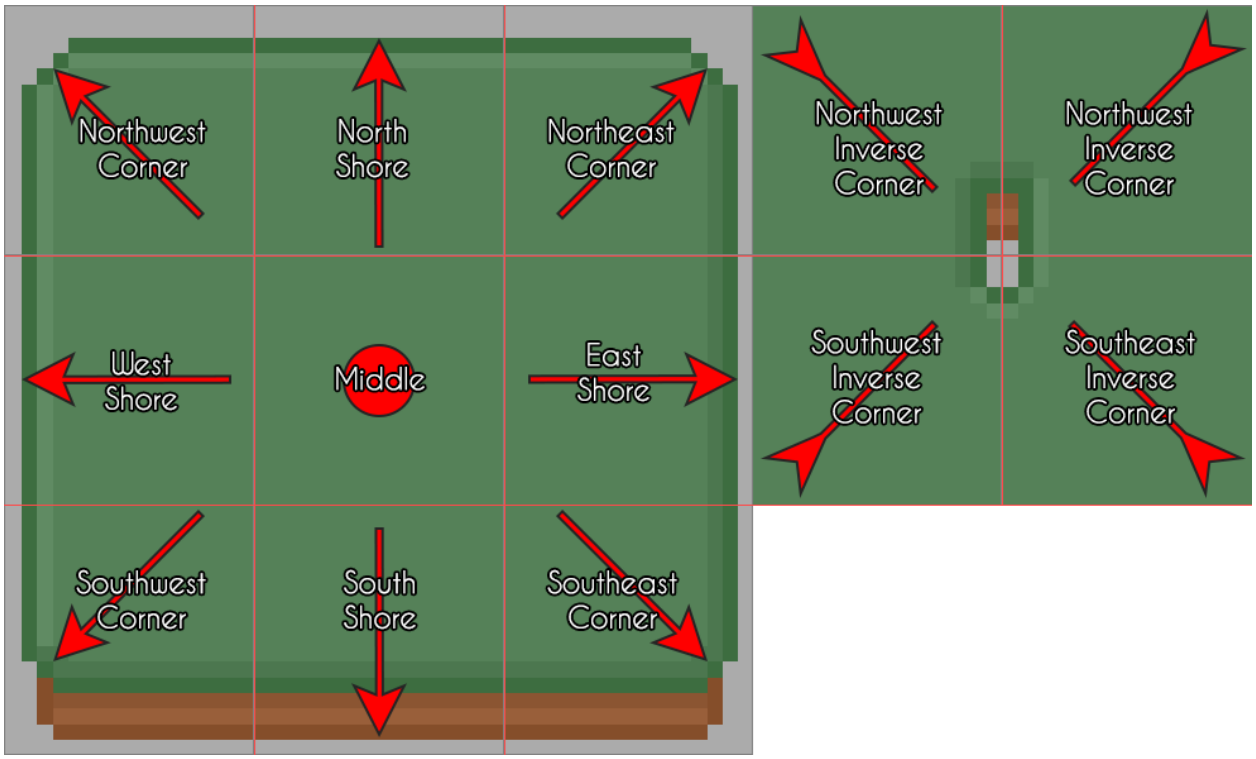
As this tool uses rule tiles, you need to have the **2D extras, V1.5.0-preview** by Unity Technologies installed in your project, which can be acquired for free in the following link: https://github.com/Unity-Technologies/2d-extras

The asset may or may not work with future versions of 2D extras. If you get errors, make sure to test it with V1.5.0-preview.

You also need to download the tilemap package by going to Window > Package Manager and looking for tilemap. The asset has been confirmed to work with Tilemap 1.0.0 version



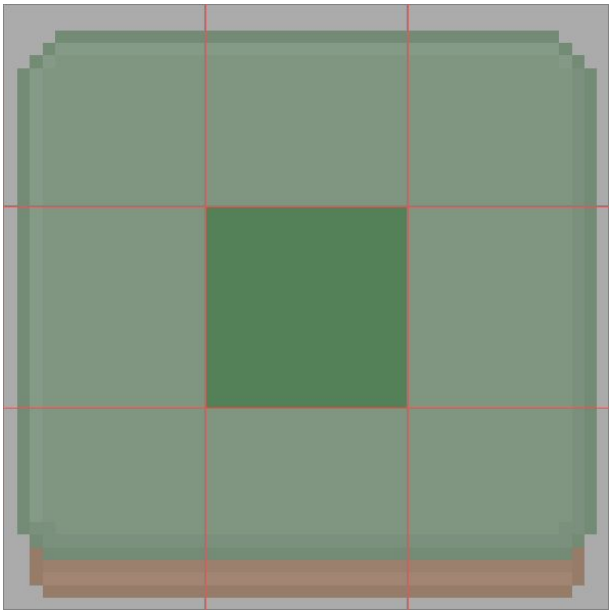
Necessary tiles



This is an example tileset made out of 16x16 tiles. For it to be a valid tileset that works with this tool, we'll need a total of 13 tiles:

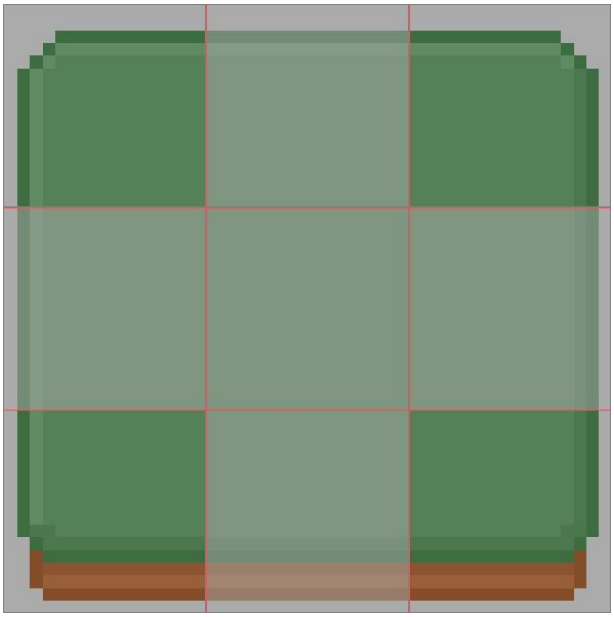
- 1 MiddleTile
- 4 Corner tiles
- 4 Shore Tiles
- 4 Inverse Corner Tiles

Middle/Center/Filling Tile

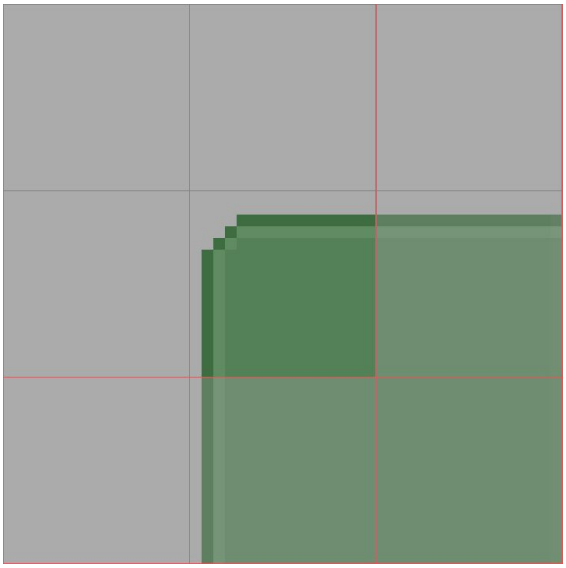


The middle tile is the “filling” tile of any tileset. It is always surrounded by tiles on all of its sides, even diagonally.

Corner Tiles

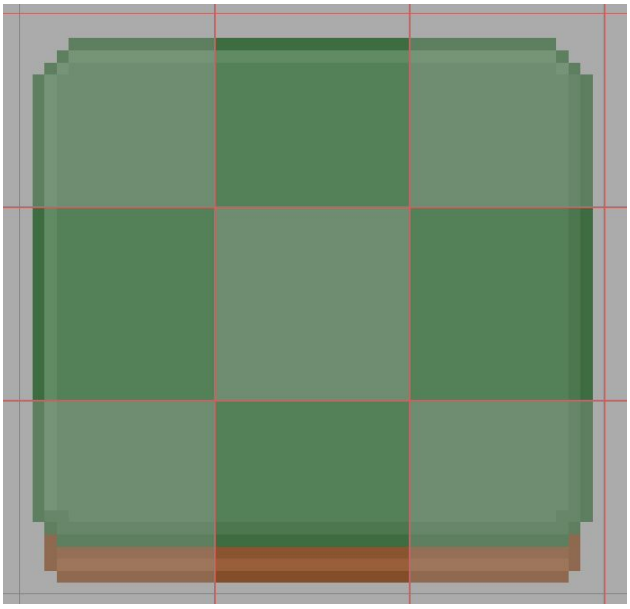


The four corner tiles of the tileset used to make convex corners. Those are the northwest corner, northeast corner, southwest corner and southeast corner.

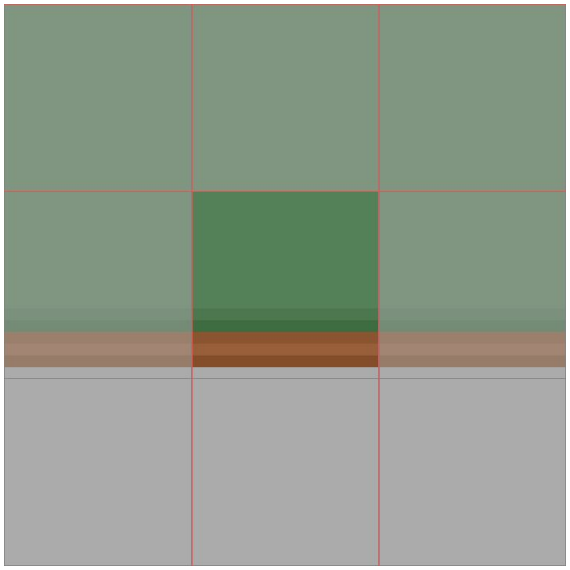


Usage example of a northwest corner tile

Shore/Edge/Side Tiles

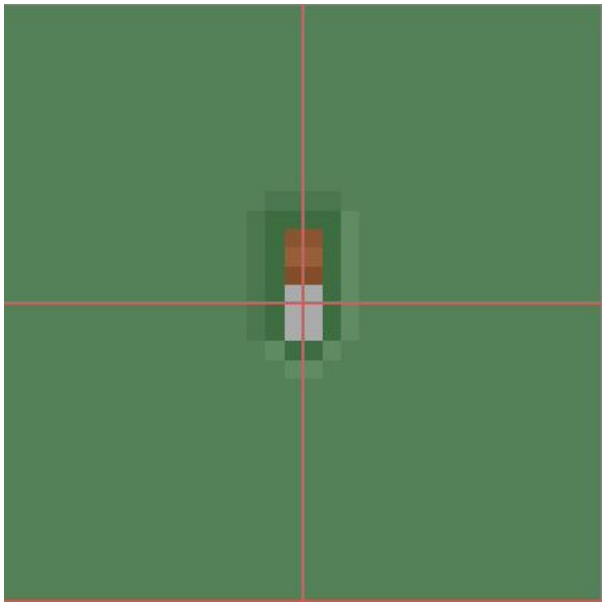


Shore or edge tiles are the four tiles forming the sides of the tileset. Those are the north shore, east shore, south shore and west shore.

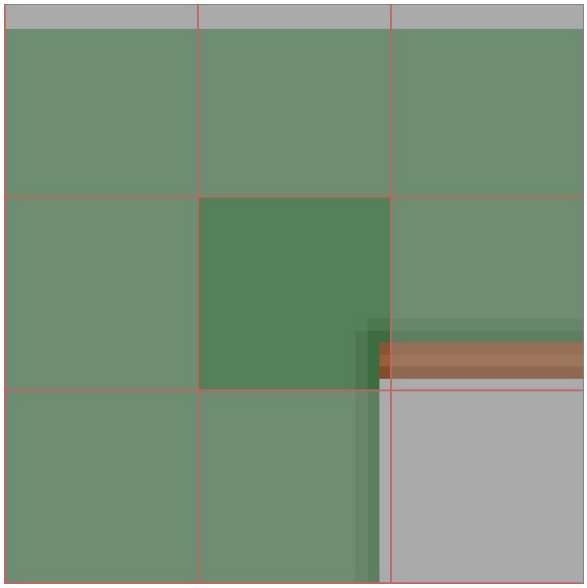


South shore tile usage example

Inverse Corner Tiles



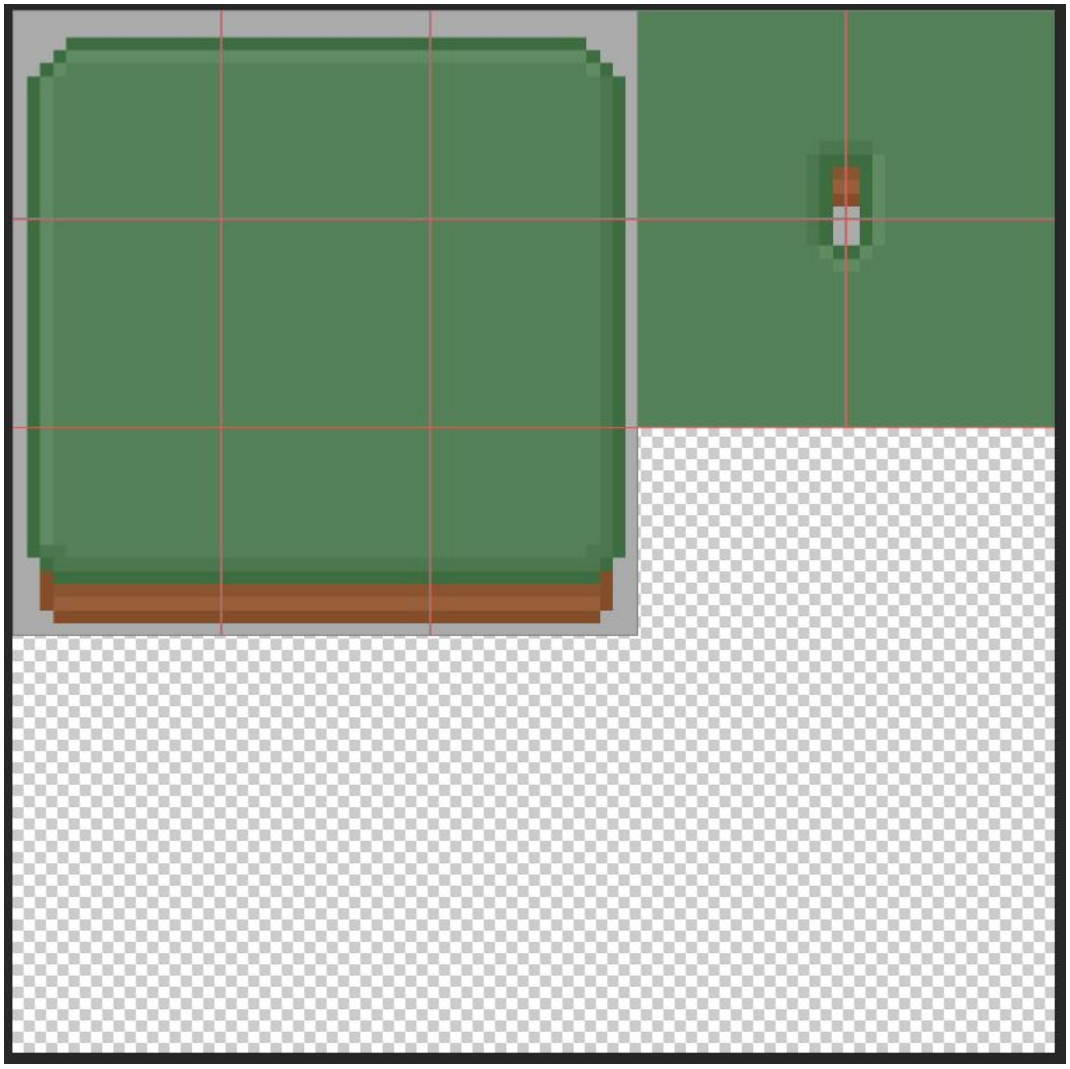
Inverse corner tiles are used to make concave turns. Those are the northwest inverse corner, northeast inverse corner, southeast inverse corner and southwest inverse corner.



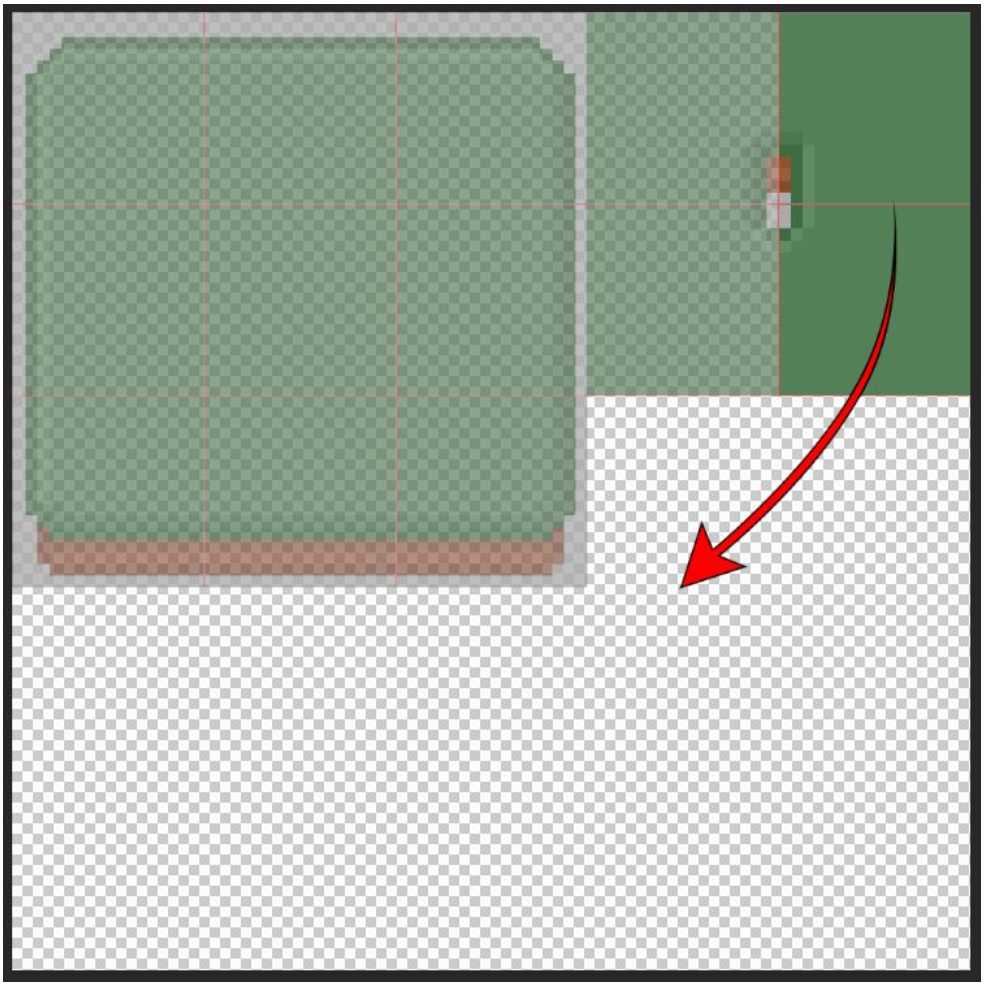
Usage example of the northwest inverse corner.

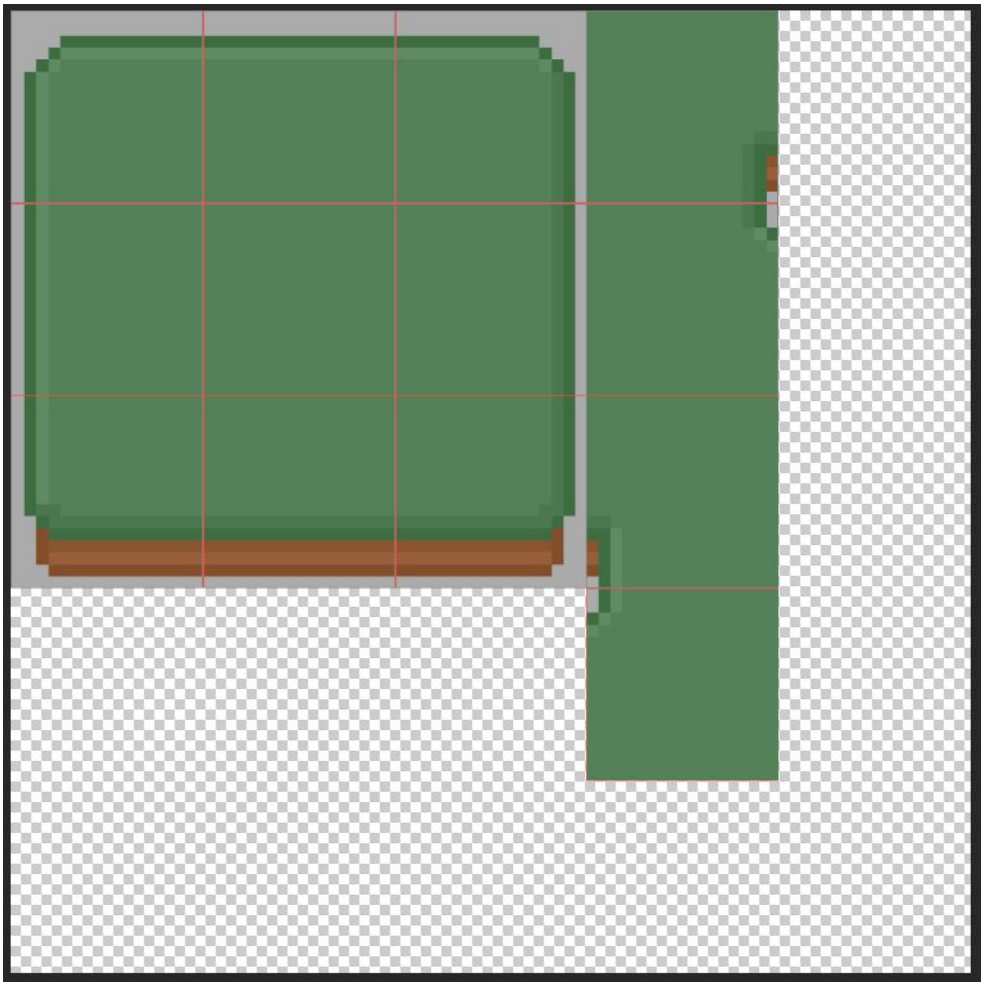
Setting up your base tileset

First, draw your base tileset and assemble as shown. It doesn't have to be the same resolution as the one below. You can use tilesets where the individual resolution of each tile is 8x8, 16x16, 32x32, 64x64, 128x128. Check the limitation section for more details.

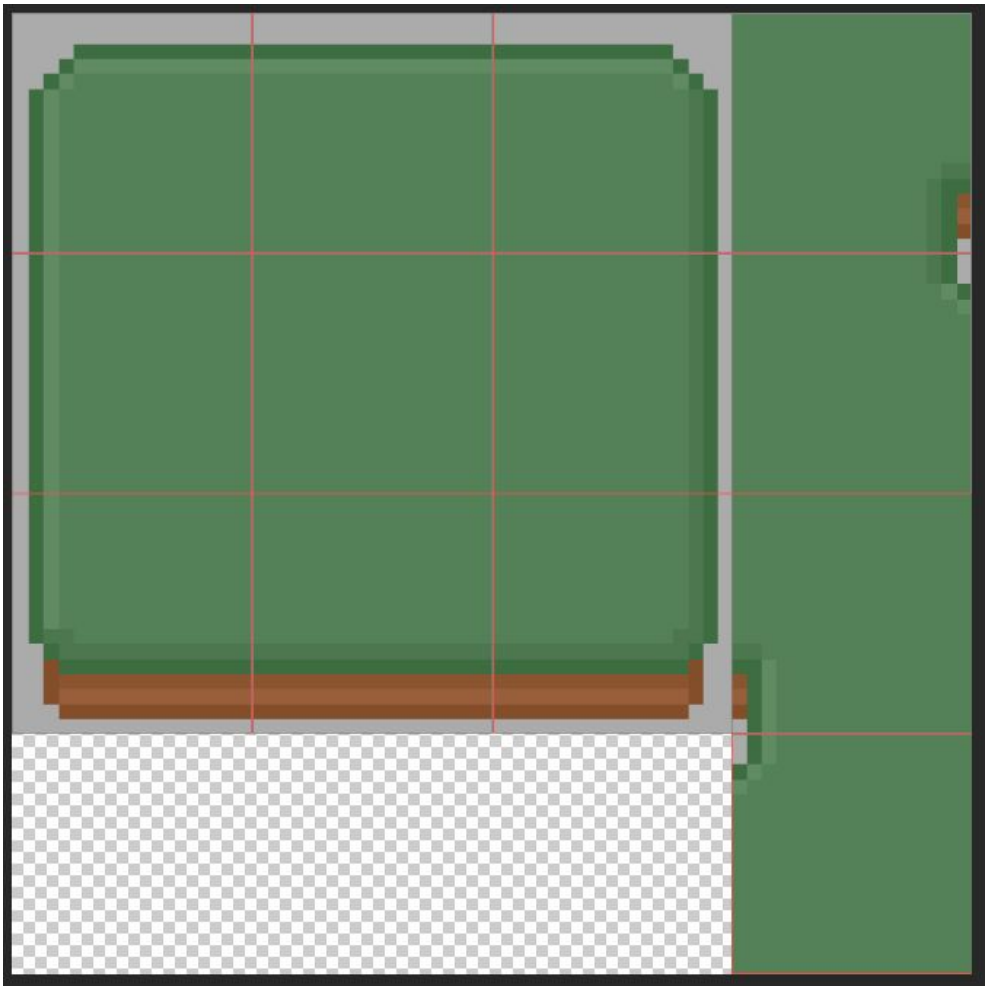


Take the Northeast Inverse Corner and Southeast Inverse Corner tiles and move them below the Northwest Inverse Corner and Southwest Inverse Corner tiles.





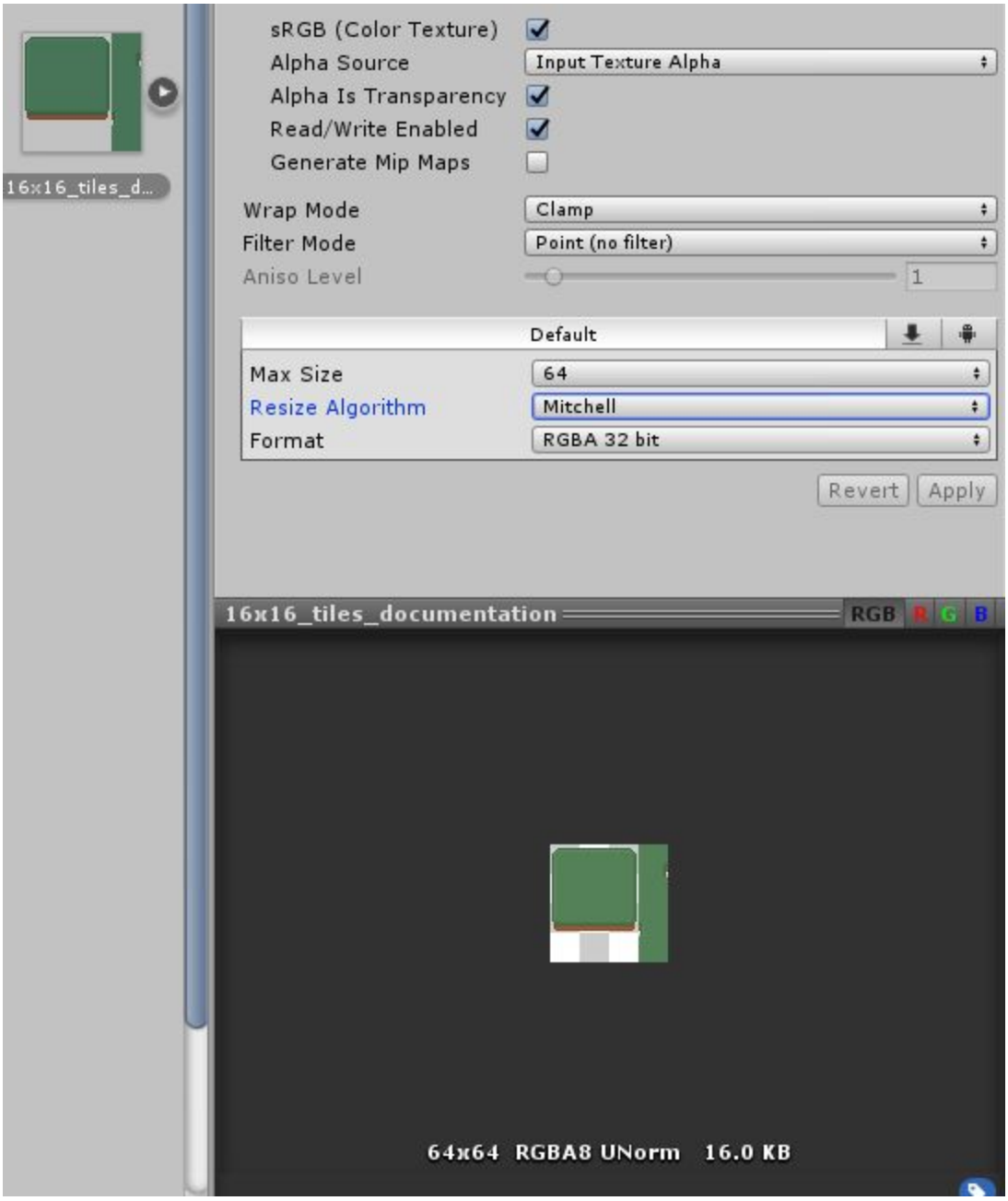
Now crop the image so it’s a square.



And you’re set, that’s the layout required for the tool to work. Your base tile should always be a 4x4 tile image containing the tiles setup in this way.
Note that if your input base tileset is not using this layout, your output tileset will not be generated properly.

Importing your base tileset to Unity

Save the base tileset that you drew inside any of the folders within the assets folder of your project. Then navigate to it on your project tab to access the image’s import settings.



Change the following settings:

Read/Write Enabled: Check that option. It is very important that it is checked, or else the script won't be able to access the pixel information on the image.

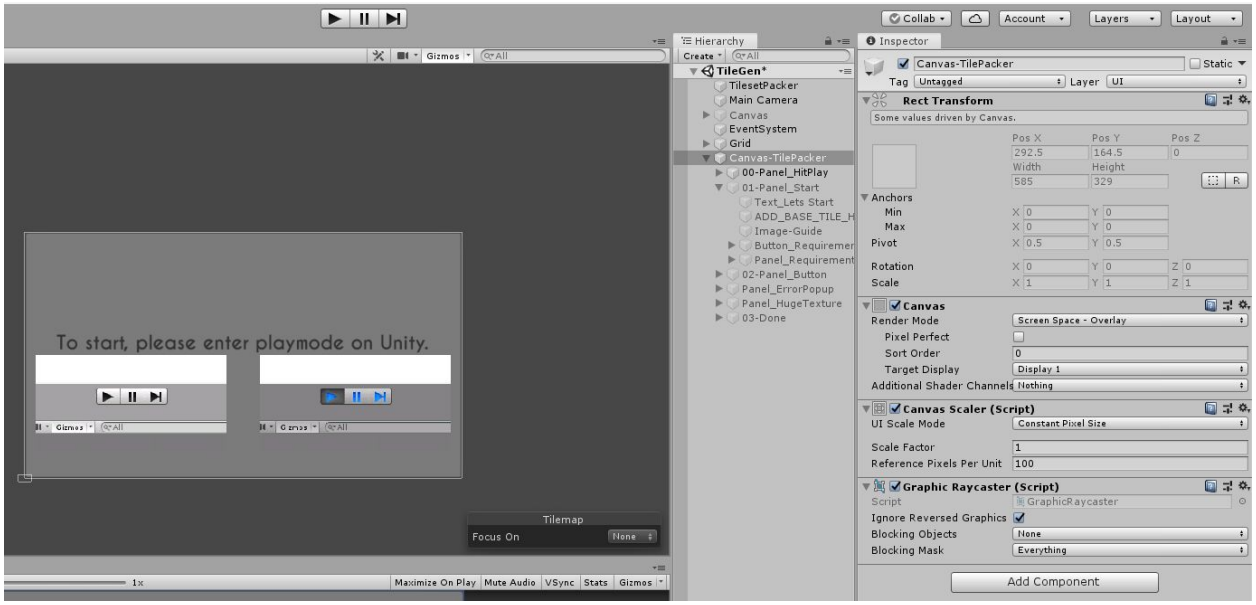
Filter Mode: If your want the image to be "crisp" (that's the case with most pixel art) you should set it to Point (no filter). If not, set it to bilinear.

Max Size: Set it to the equivalent of the resolution of your base tileset. On the example above, the tileset is 64x64 (as shown in the preview at the bottom), so I set it to 64.

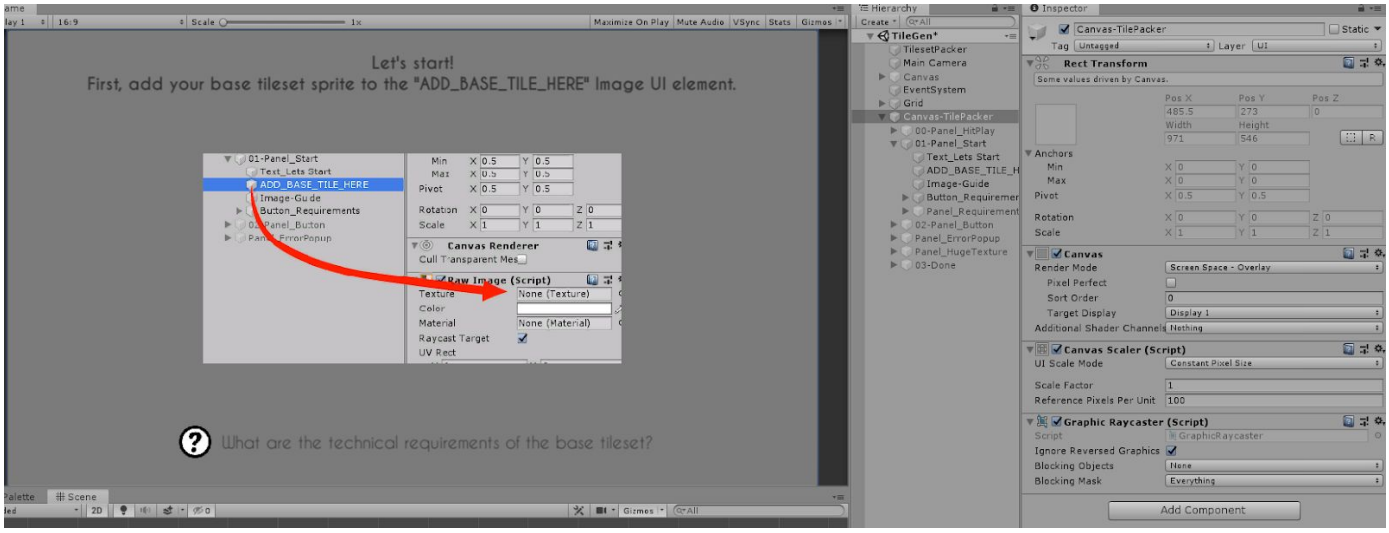
Format: RGBA 32 bit

After that, open the scene “TileGen” included in the project. It’s an easy process, the tool should guide you on how to do it.

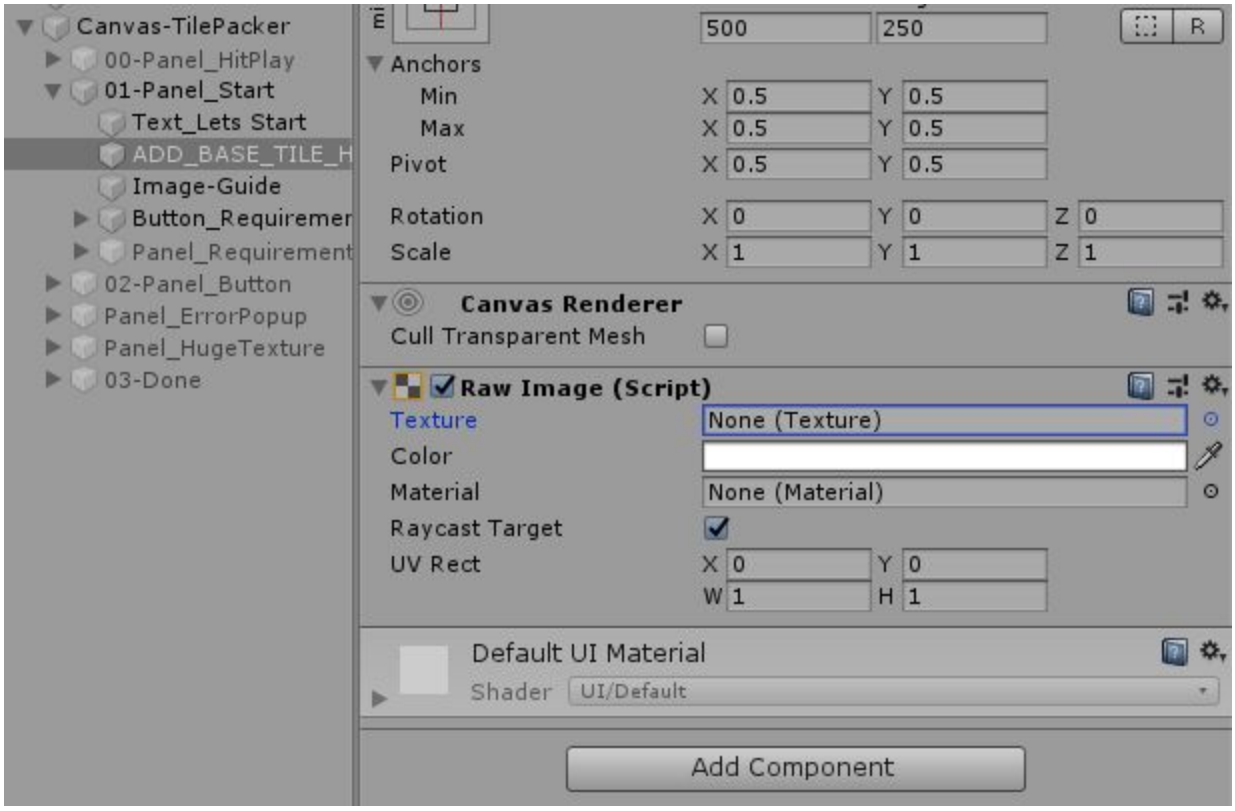
Enter playmode.



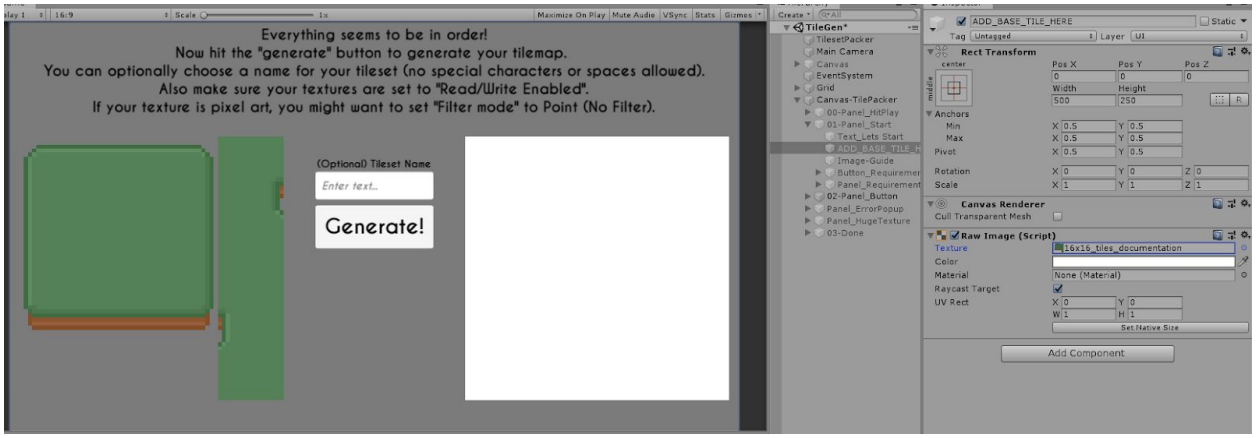
As soon as you enter playmode, the tool will ask you to add your base input tileset under the “ADD_BASE_TILE_HERE” raw image component. On your hierarchy, it is located under Canvas-TileGen > 01-Panel_Start > ADD_BASE_TILE_HERE.
Access that GameObject and add your input tileset to the texture field of the Raw Image component.

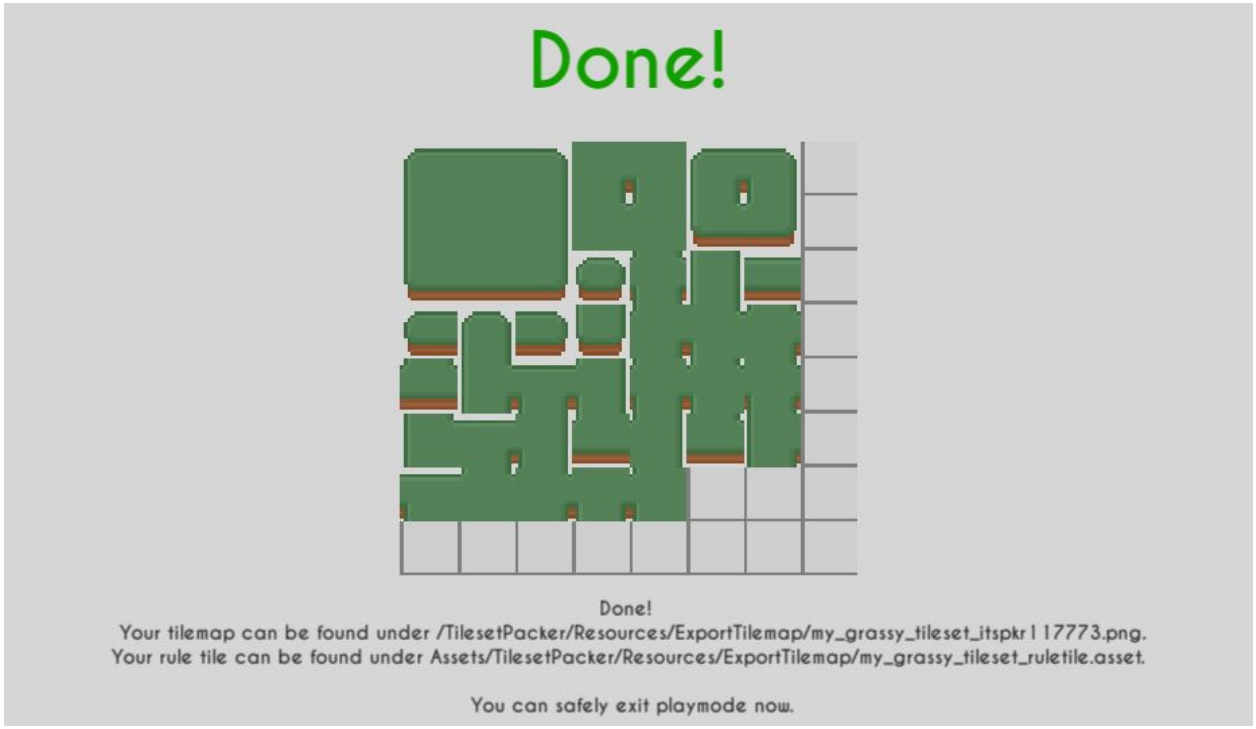


Navigate to the desired base tileset image. Be careful as the tool will choose the first image you click on. If you choose the wrong image accidentally, just exit and enter playmode again.

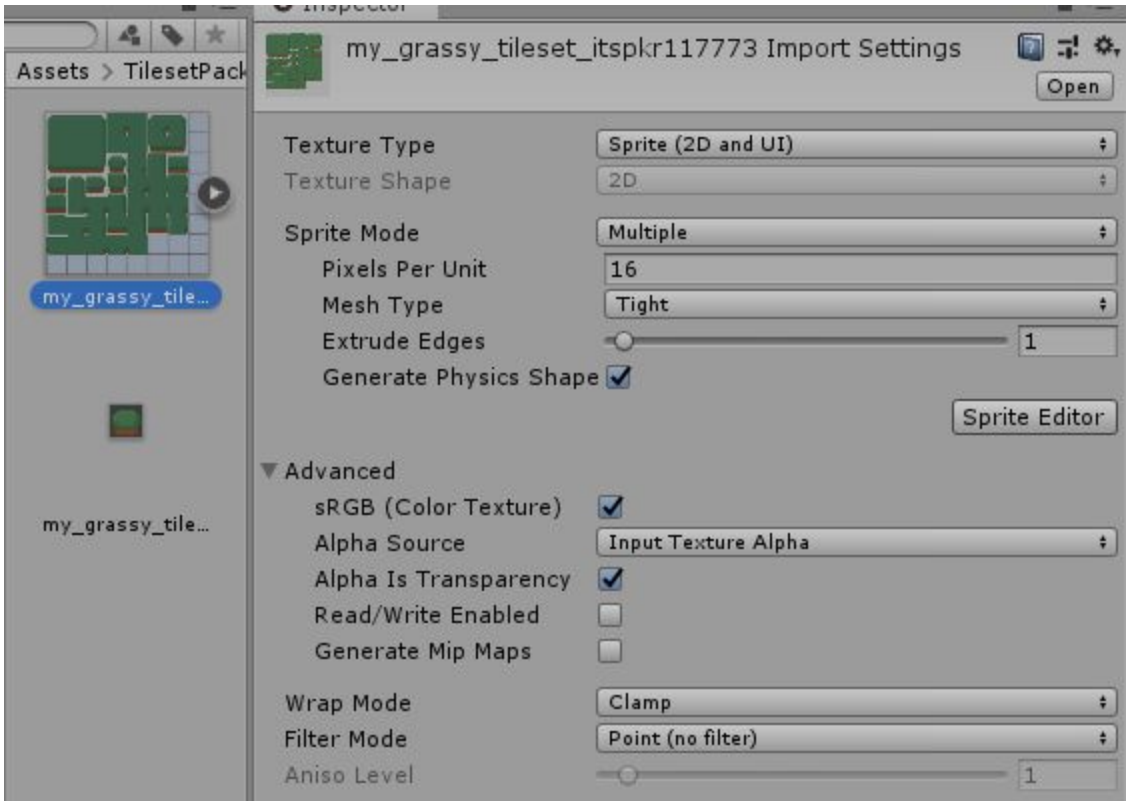


The tool will evaluate your image and see if it's within the limitations. If you did everything correctly, it should be. If not, please check the limitations section on this document.
You can optionally give the image a name and click on generate. The process is quite quick on smaller images, but can take quite a while on larger images.





After the processing is done, the tool will export your final tileset file and the Rule Tile. It will also give you the export directory.

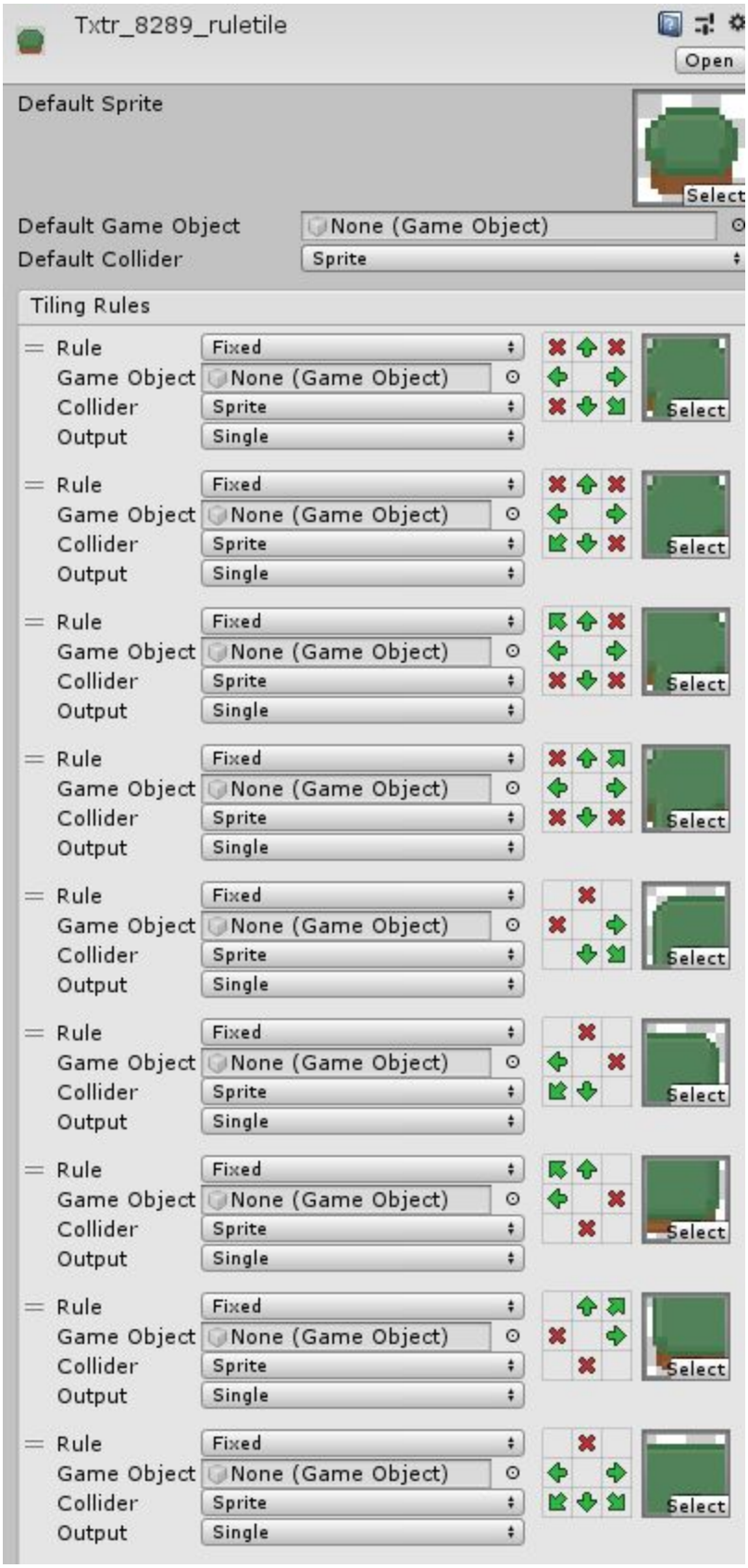


The Rule Tile

Unity's rule tile is an impressive asset that works perfectly with the complex tileset that TileGen creates. It works by letting the user define rules that the tile will follow. If a certain rule is met, a certain tile will be displayed.

Rules are based on neighboring tiles - you can set a rule to be either:

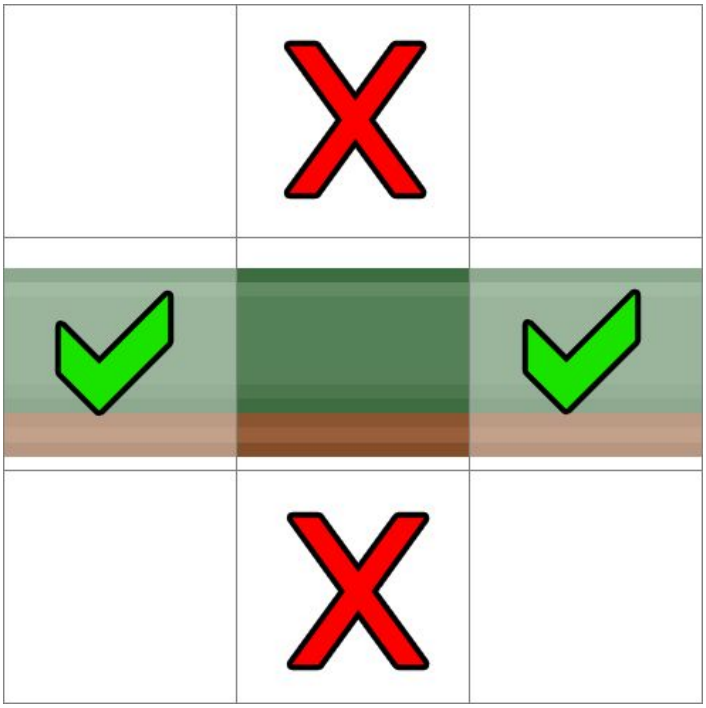
- "There needs to be a tile here" (**green** arrow)
- "There cannot be a tile here" (**red** X)
- "Irrelevant" (**nothing**).


Txtr_8289_ruletile
Open
Default Sprite
Select
Default Game Object
None (Game Object)
Default Collider
Sprite
Tiling Rules
Rule
Fixed
Game Object
None (Game Object)
Collider
Sprite
Output
Single
Select

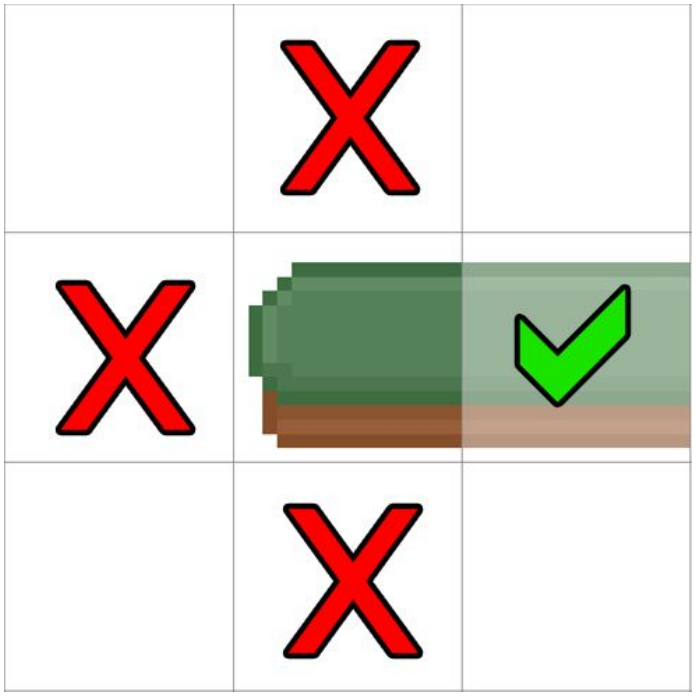
Let’s take a closer look at the west/east bridge tile and it’s rules.

The neighboring tile rules translate to:
There **cannot** be any tile north and south of this tile
There **needs** to be tiles to the east and west tile.
It’s **irrelevant** whether there is a tile or not on the rest of the neighboring positions.

If we put this on a grid, we will see that it makes sense:



If we, for instance, make it so there is no tile on the left neighboring position, we create a west-facing end tile.



And if we go back to Unity and check the rule for the west-facing end tile, we'll see that it looks like this:

Which is exactly what we expect.

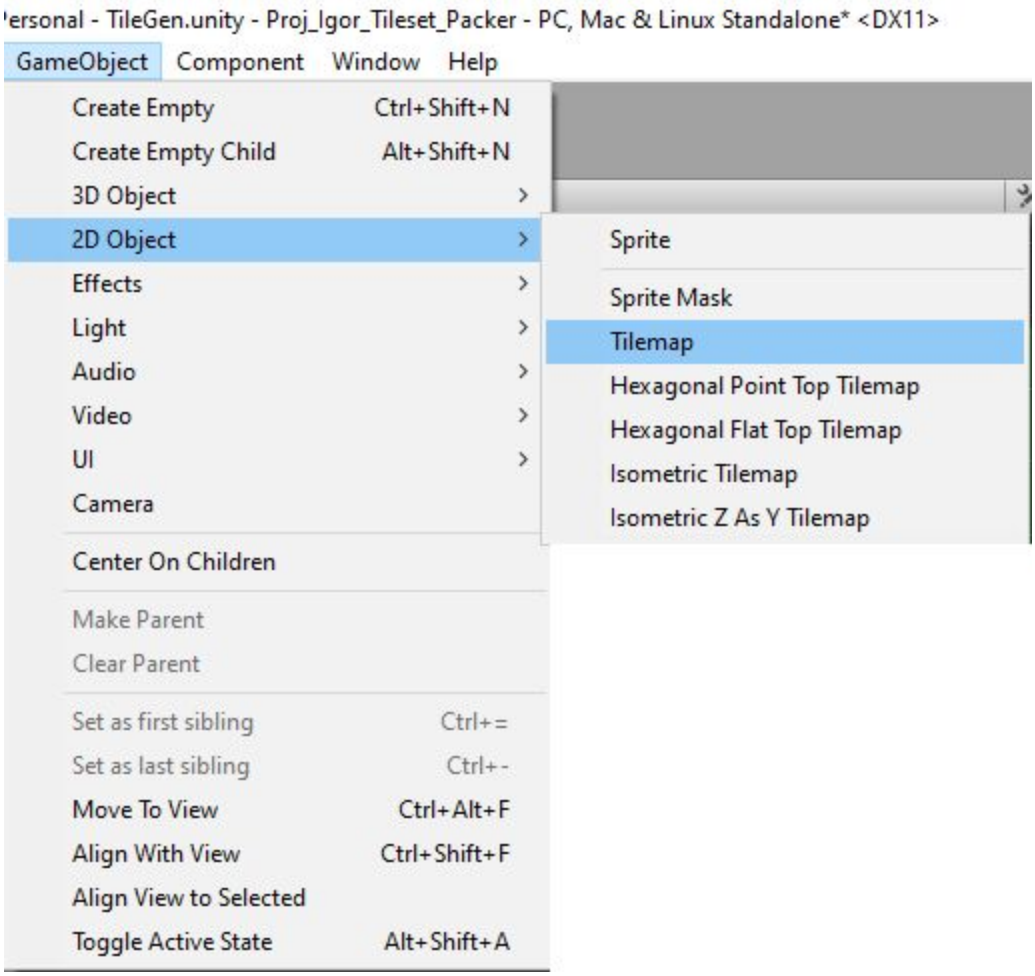
TileGen provides all of the complicated rules, set up in a way that some rules are prioritized over others, to avoid visual glitches while painting with the rule tile.

How can I paint using the Rule Tile?

First, you need a Grid and a TileMap.

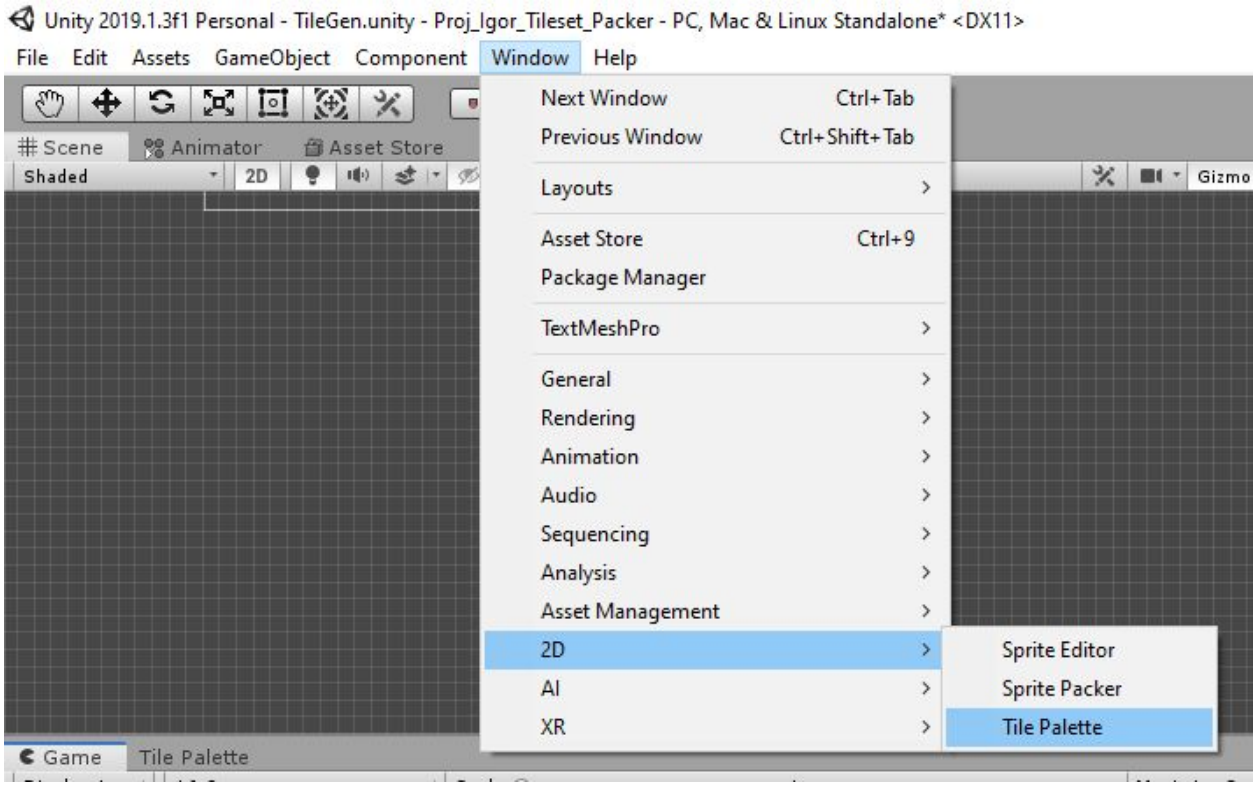
The former is the grid of tiles used as a canvas upon which the user can draw their environments. The tilemap acts as a layer of tiles - You can have multiple tilemaps within a grid. For instance, the bottom tilemap could be the water tilemap, used to paint water tiles. The middle tilemap could be a ground tilemap, used to paint dirt and grass. The top tilemap could be a structure tilemap used to paint buildings and objects, and so on.

To create a grid and a tilemap, go to GameObject > 2D Object > TileMap

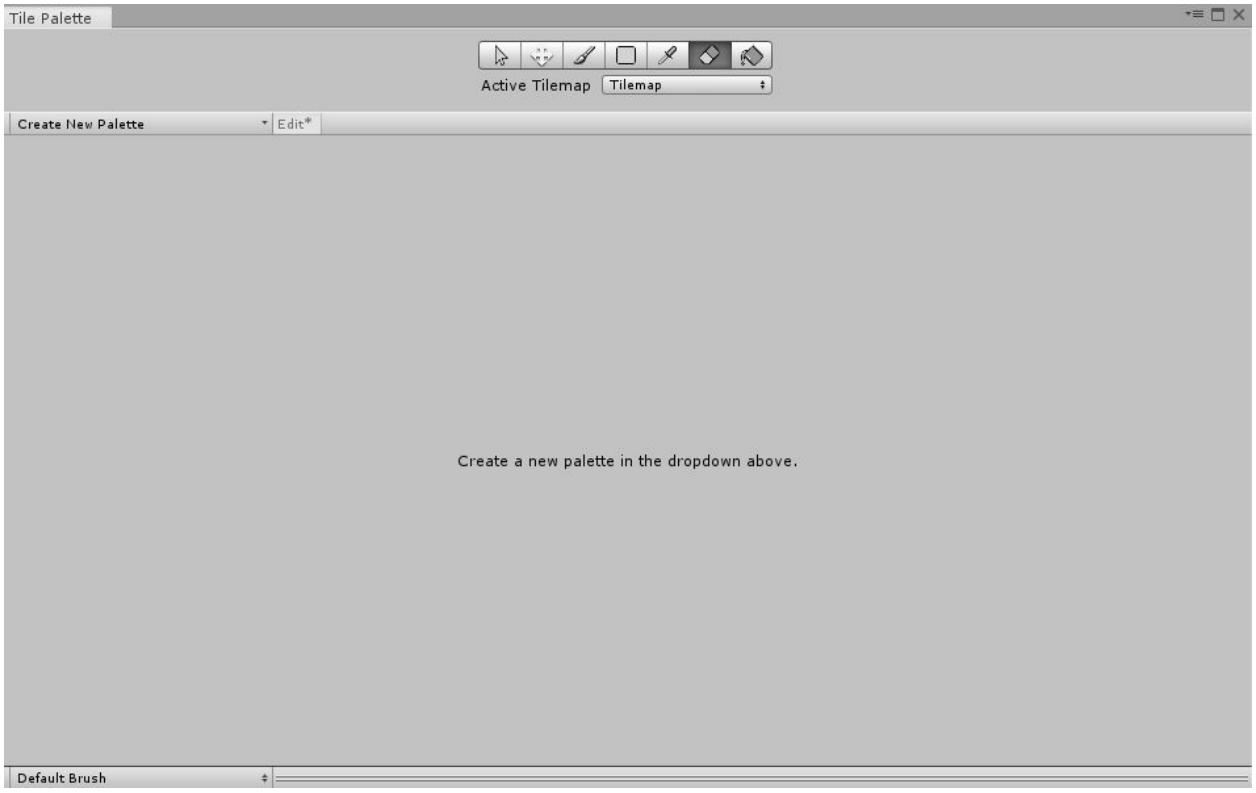


Unity will create a grid on your scene automatically, and a tilemap as a child of the grid. Doing this again will create a second tilemap within the grid, and so on.

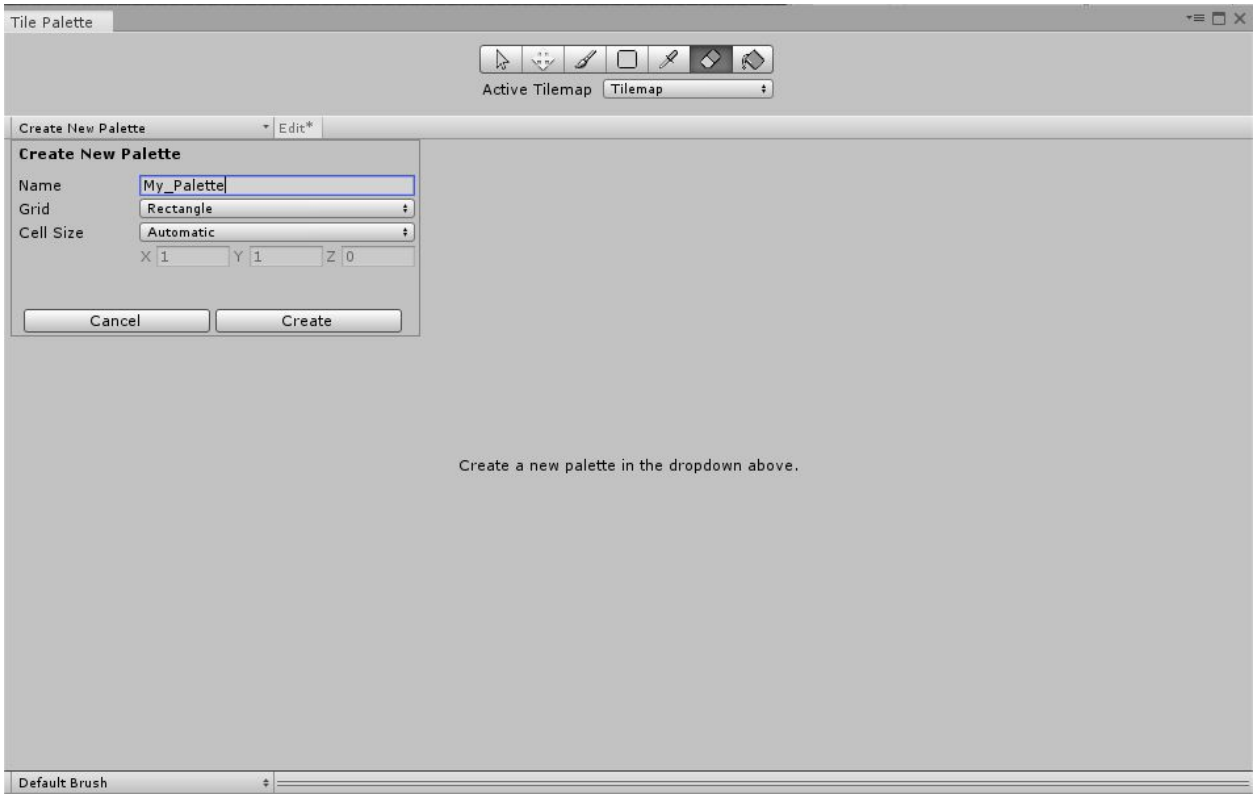
Now that the grid is set, you'll need a tile palette. A Tile Palette is the collection of tiles you have, that can be used to paint upon the grid.
To create one, go to Window > 2D > Tile Palette



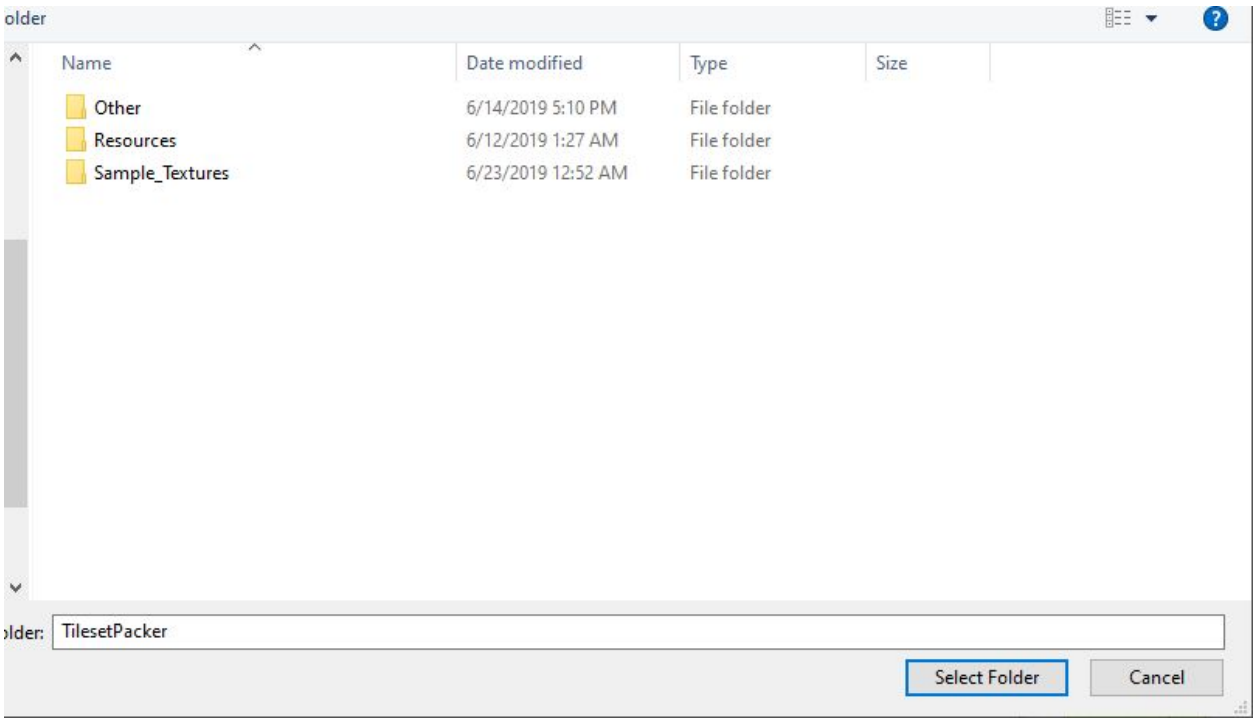
And you will be met with this window:



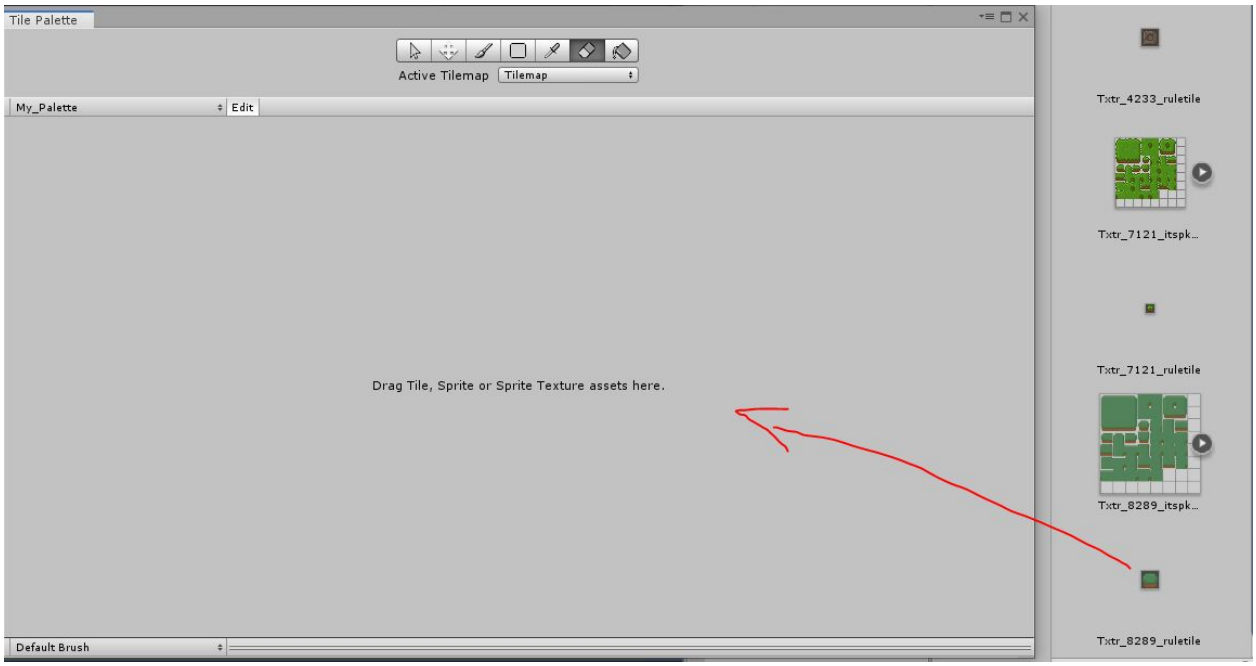
Click on “Create New Palette” on the top left



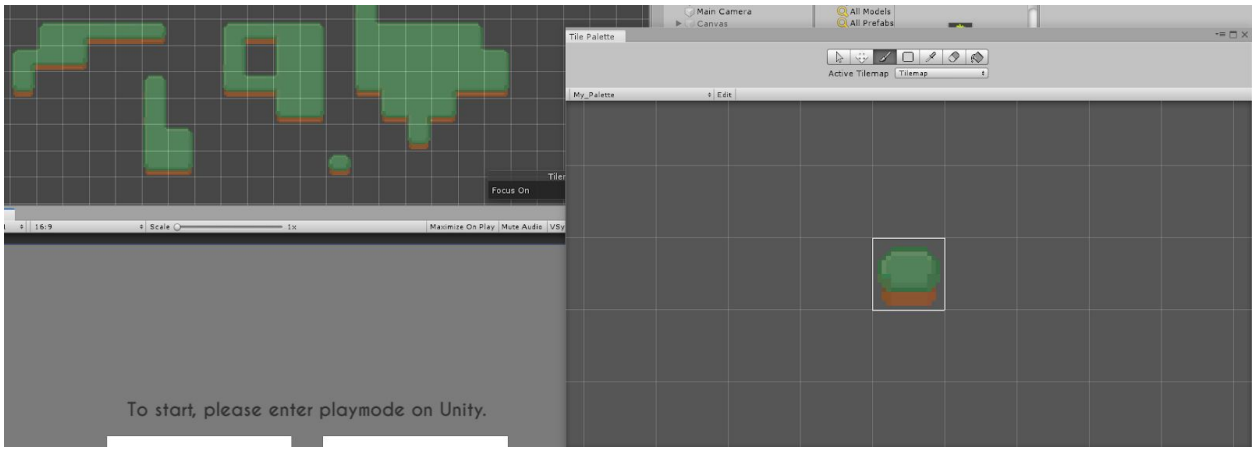
You will be met with a small window. Choose the name of your palette, and it's directory



After that your Tile Palette will be made. All you need to do is to drag the Rule Tile that TileGen created for you into the Tile Palette.



Note that you can also drag other individual tiles to the palette as well.



Click on your newly imported Rule Tile within the Tile Palette, choose the brush tool, and paint your tile over the grid, on the scene view.

What are the limitations?

While Tilegen is a really useful tool, it does not come without its limitations.

- Your base square tileset cannot have a resolution greater than 512x512. The output tileset is 4 times bigger than the base tileset, so if, for instance, your base tileset is 256x256, the output tileset is going to be 1024x1024. That is a lot of pixels to redraw.
    - Technically the algorithm should work with any dimensions given enough time, but testing showed that it gets very slow when the canvas it has to draw on has more than 1024x1024 pixels.

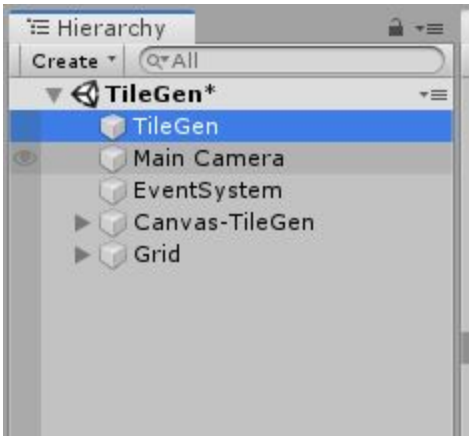
        - There is, though, a safe mode you can disable to try and render very large tilesets. Unity will probably freeze, so do it at your own risk!
- You are also limited on how you draw the corner tiles. The way the algorithm works with the corner tiles is that it divides the corner tile by 4 and copies the corner quadrant out of that division.
    - So for instance, the Northeast corner will have its northeasternmost quadrant copied and used for generating other tiles.
    - So essentially, the drawn corners on the corner tiles must be within the equivalent corner quadrant of those tles. As an example, this is how the algorithm builds the North End tile.
    - If the rounded/sloped part of the corner tile is outside of its proper quadrant, this is how it will generate it:
        - As this is a somewhat complex limitation, please watch the tutorial on the asset's main page on the Unity Asset Store for more information.

I already have a full tileset. Can I just create a rule tile with it?

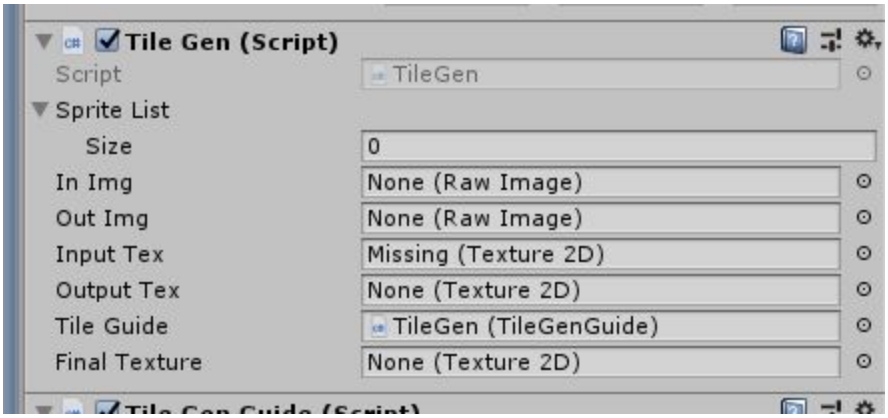
Yes! As long as your full tileset follows the template used by the system.

You don't even have to enter playmode to do this. Do the following:

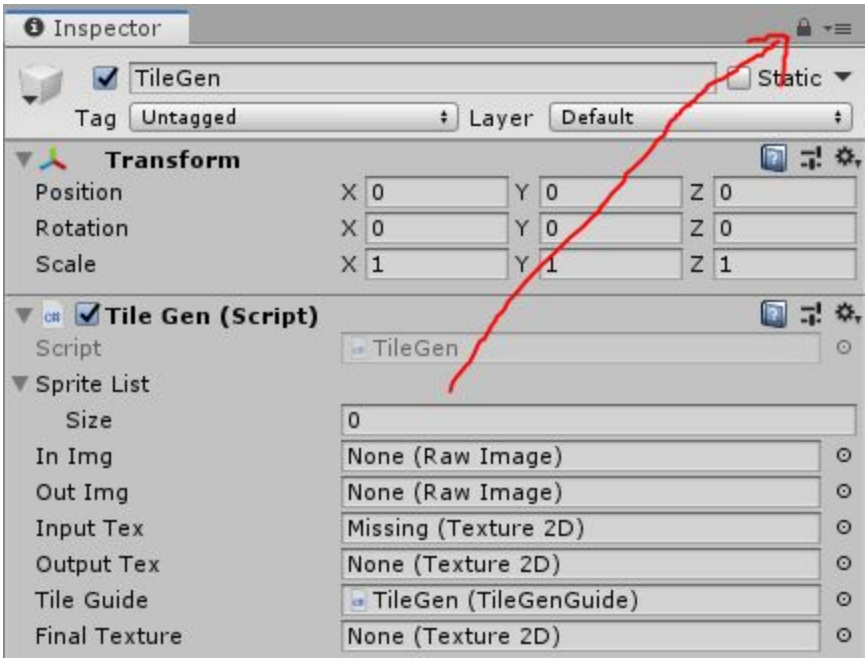
Access the "TileGen" gameobject on the "TileGen" scene provided with this tool.



Access the "TileGen.cs" Script contained within this GameObject.

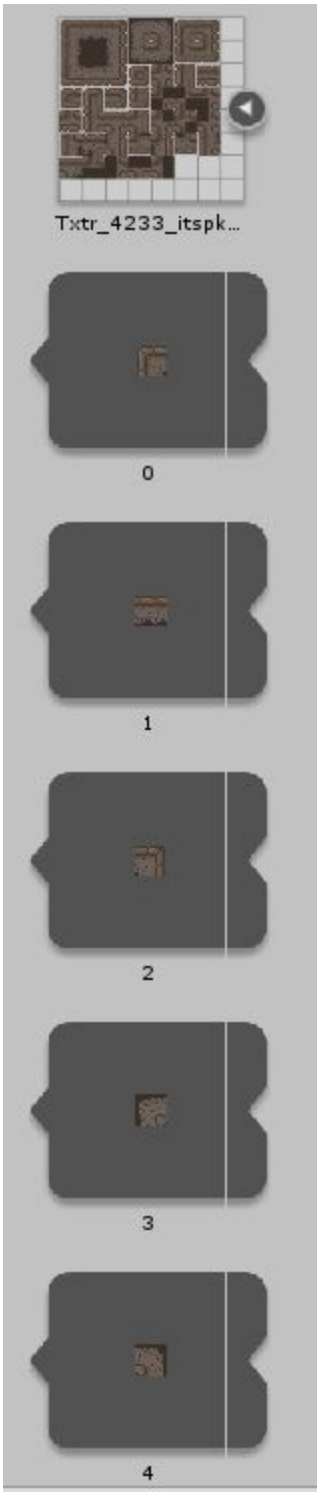


Now, lock the inspector so it doesn't change tabs when you click on something else.

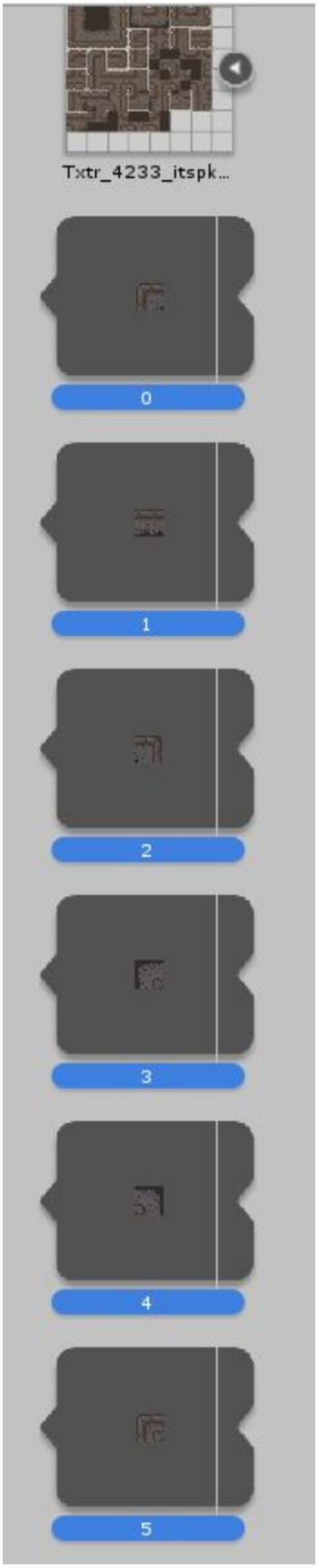


Find your tileset. It should already be divided into multiple Tiles, and also it needs to be setup according to the standard tileset layout of TileGen.

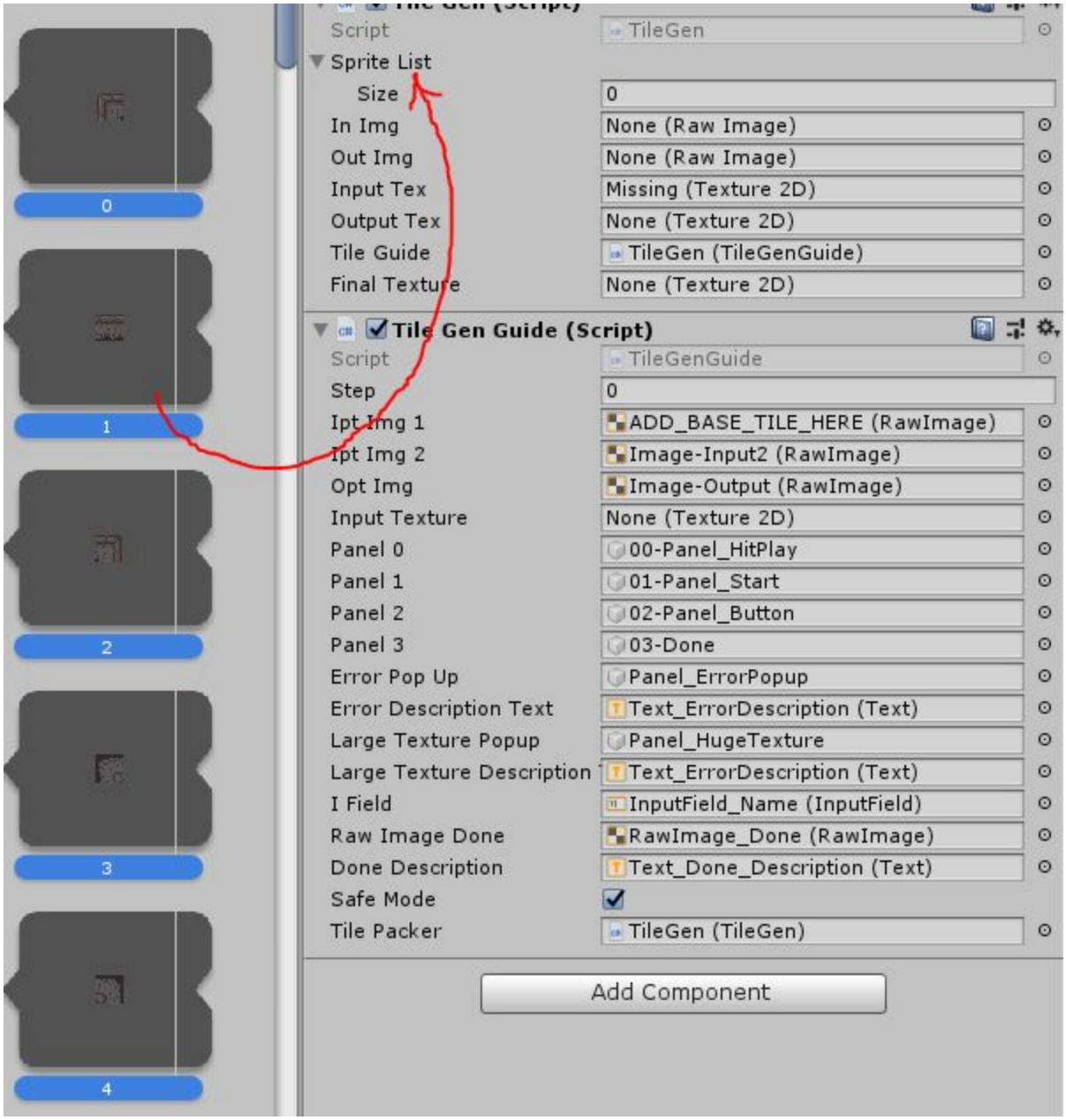
Expand the main tileset so you can see the individual tiles within



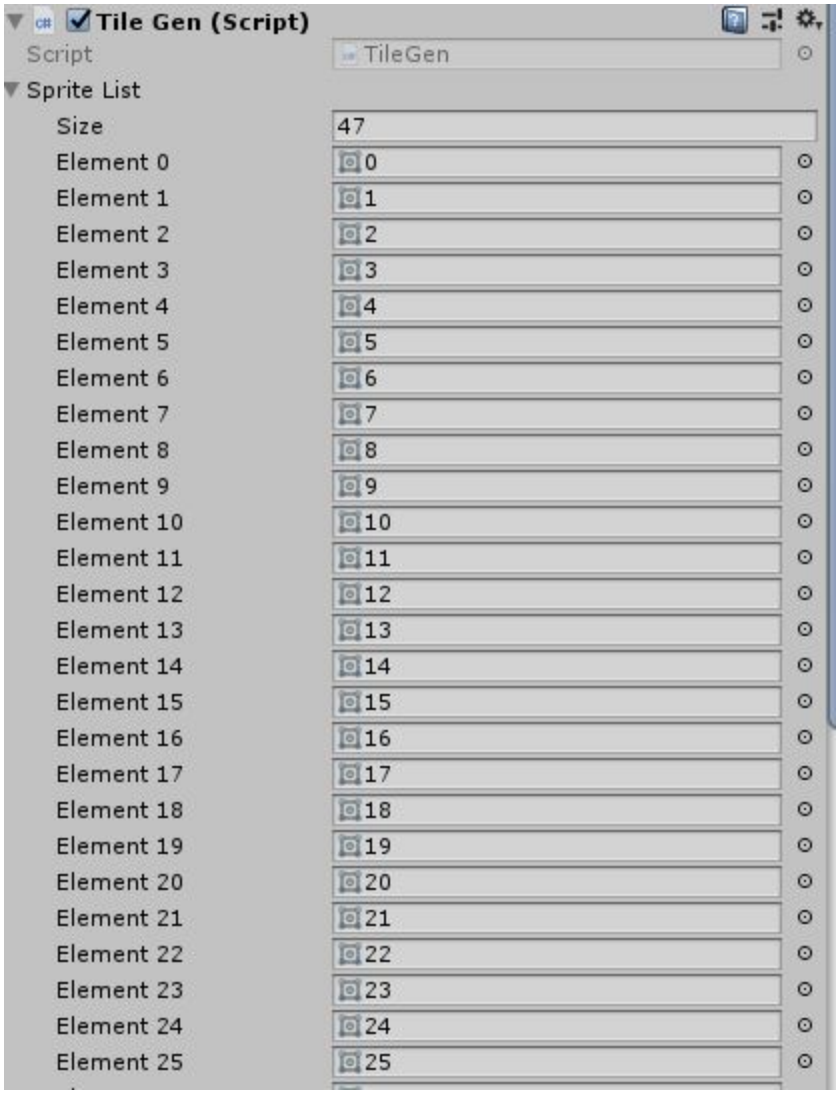
Now select all tiles except for the main tiles. If your configuration of tiles is proper, they should range from 0 to 46.


Txtr_4233_itspk...
0
1
2
3
4
5

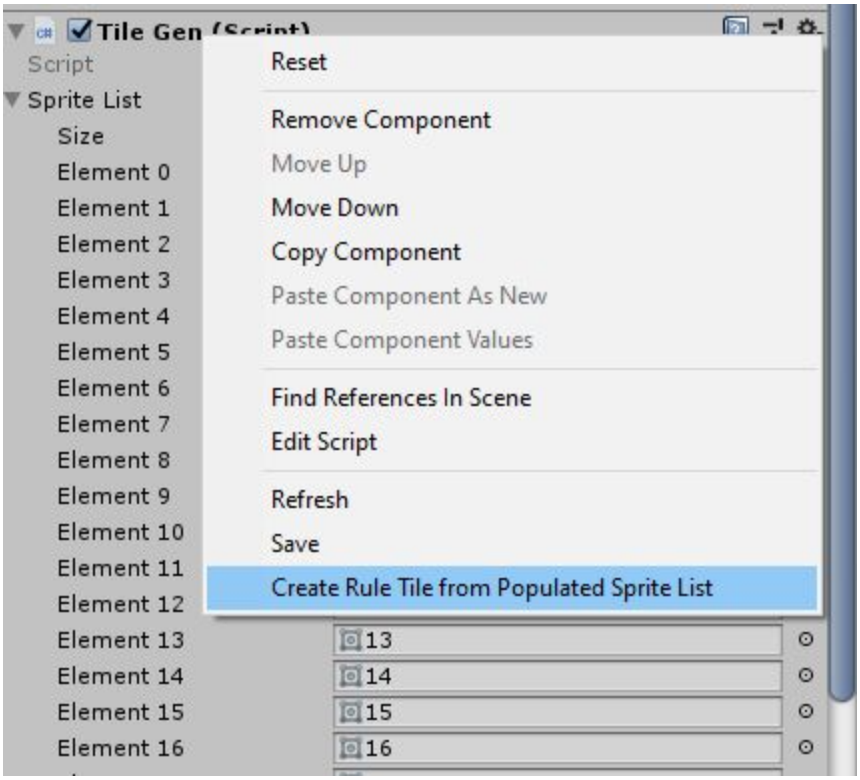
That all the tiles (from 0 to 45) into the “Sprite List” variable on TileGen (you should still be able to see it, if you locked the inspector tab properly). Do not drag the tiles to the “Size” text or the “0”, Drag it to the name “Sprite List”.



Your Sprite List should be now populated with all the tiles.

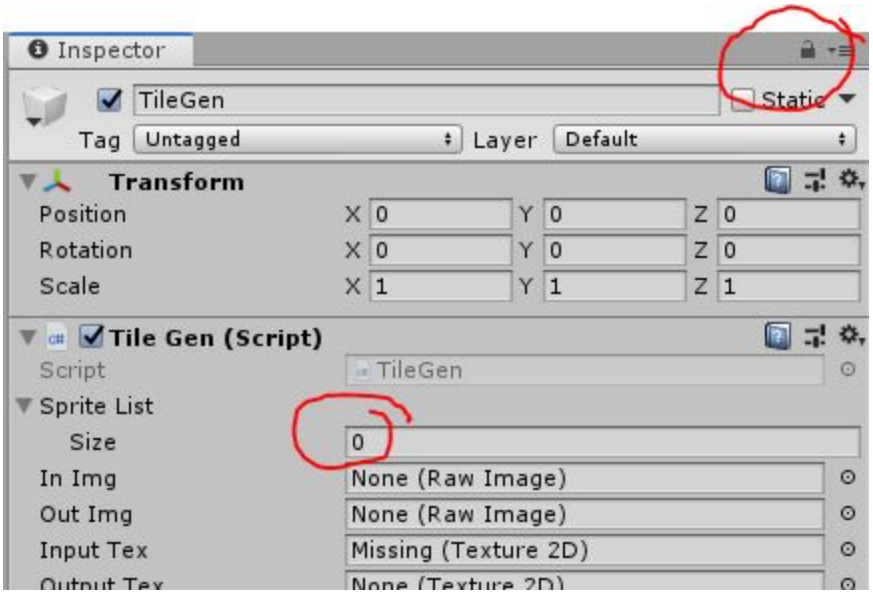


Now, right click on “Tile Gen (Script) and select the “Create Rule Tile from Populated Sprite List” option.



Your tule tile should be created.

Don’t forget to reset the Sprite List (set it to 0) and Unlock the inspector.



If you forget to reset the Sprite List, it will malfunction the next time you try to create a Rule Tile.