

# Isometric Tile Toolkit for Blender

by Screaming Brain Studios

The **Isometric Tile Kit for Blender** is a series of Blender scenes each containing an Isometric Camera and a 3D Mesh, as well as a Custom Properties panel for making adjustments. There are no lights in the scenes, and all objects are using Emissive material, which allows for rendered tiles to have exactly the same brightness and lighting as the original textures. This toolkit was made using Blender 3.2.0, and may not work correctly with older versions of Blender. There isn't anything too complicated going on behind the scenes, so there shouldn't be many issues using older versions, but it is recommended you use the latest version of Blender just in case.

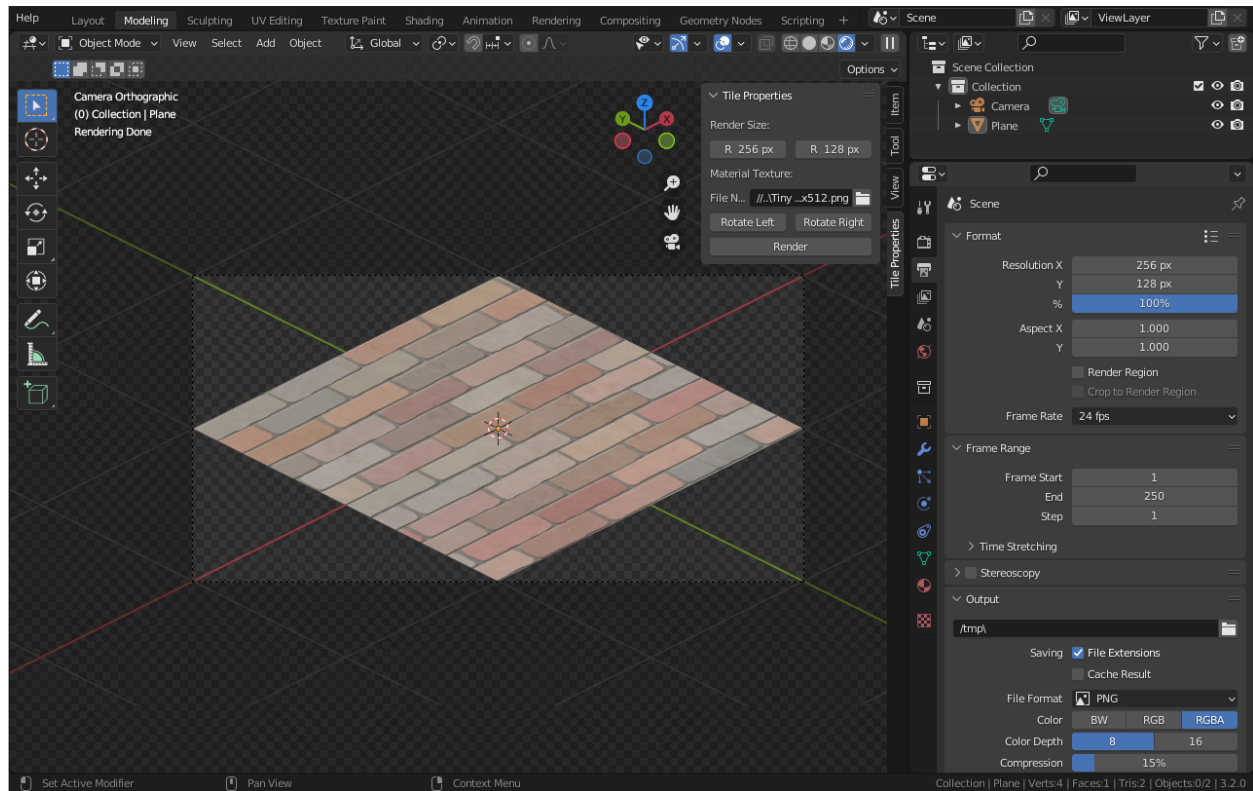
Each scene has been set up for a specific type of Isometric tile: Floor Tiles, Wall Tiles, and Cube Tiles. By default, each scene has a demo texture from the Tiny Texture Pack loaded and ready for you to experiment with. As stated above, there are no scene lights, allowing for renders to retain the exact color and brightness of the textures being used, though you can certainly add your own lighting to the scenes. To handle lighting in Screaming Brain Studios fashion, a series of Selection Masks and Shading Masks have been included with the toolkit for applying post processing or adjustments in an image editing tool like **GIMP**.

When opening each Blender scene, a small script will be run to create the Tile Properties panel. By default, the script should run when you open each scene. An alert box may pop up warning about running Python scripts, so click *Allow Execution* so that the scene can load properly. In some rare cases, the script that creates the Tile Properties panel may not actually execute. In this case, simply click on the Scripting tab at the top of the Blender workspace, make sure the *Properties Panel* script is selected (Which by default, it should be) and press the Run Script button to create the Tile Properties panel. If for some reason you cannot see the Tile Properties panel when viewing the 3D Viewport, press Ctrl+N and then select the Tile Properties tab on the right side of the Viewport to bring up the Tile Properties panel.

- *Isometric Floor Maker.blend*
- *Isometric Wall Maker.blend*
- *Isometric Cube Maker.blend*

Using the Blender scenes in combination with the Shading Masks, you can create just about any kind of Isometric Tile you might need for your project!

# Isometric Floor Maker

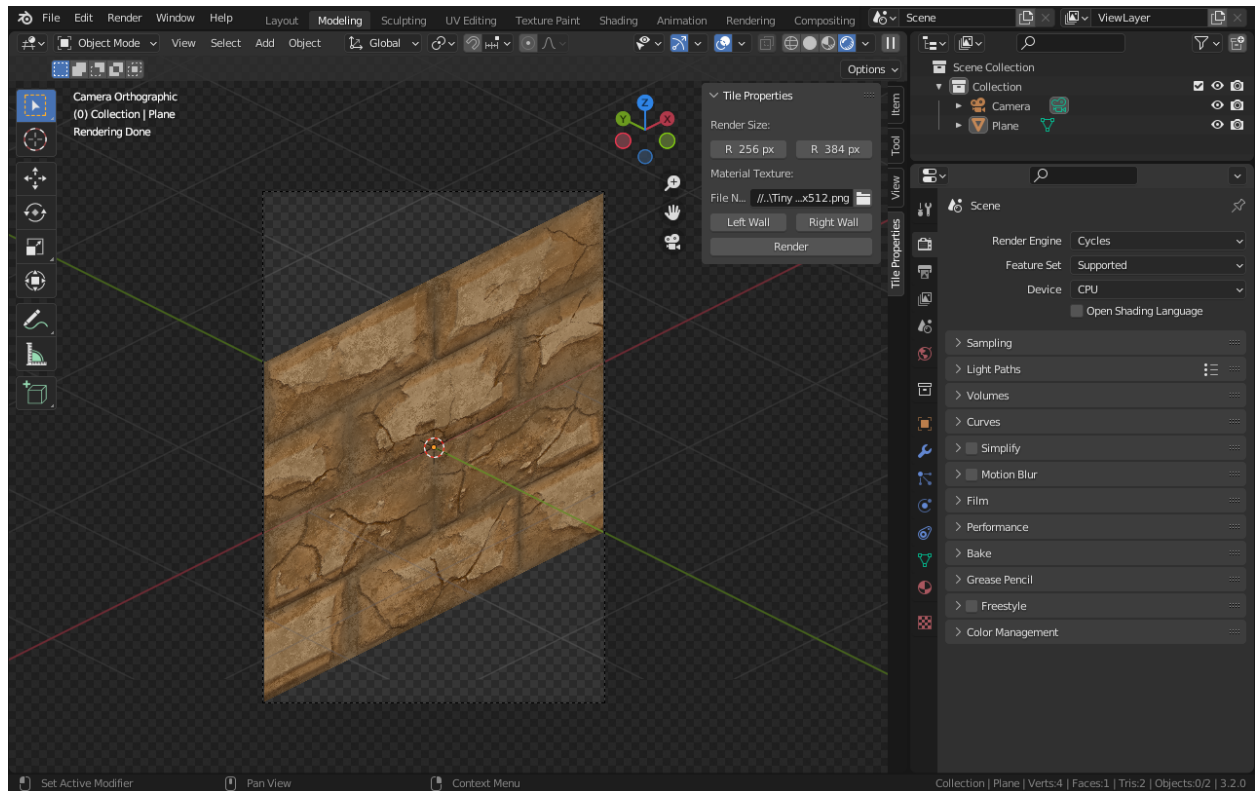


This Blender scene is set up to render Isometric Floor Tiles using the texture loaded into the Image Texture node of the 3D Plane's material shader. The Render Settings and Camera Angle have all been set up to produce a perfect 2:1 tile using any texture you choose. You can set the size of your final rendered tile by changing the values of the X and Y fields in the Tile Properties panel. It is recommended that you use 2:1 values that are multiples of two (512x256, 256x128, 128x64, 64x32, etc.) to maintain the proper render frame around the Tile.

1. Set your desired tile size using the X and Y fields in the Tile Properties panel
2. Click the Folder icon to the right of "File Name" to select a texture
3. Use the Rotate Right and Rotate Left buttons to rotate the tile
4. Press the Render button to render and export your Isometric Tile

The Isometric Floor Maker scene can rotate the texture plane around the Z axis allowing you to render the texture from 4 different directions, should the need arise.

# Isometric Wall Maker

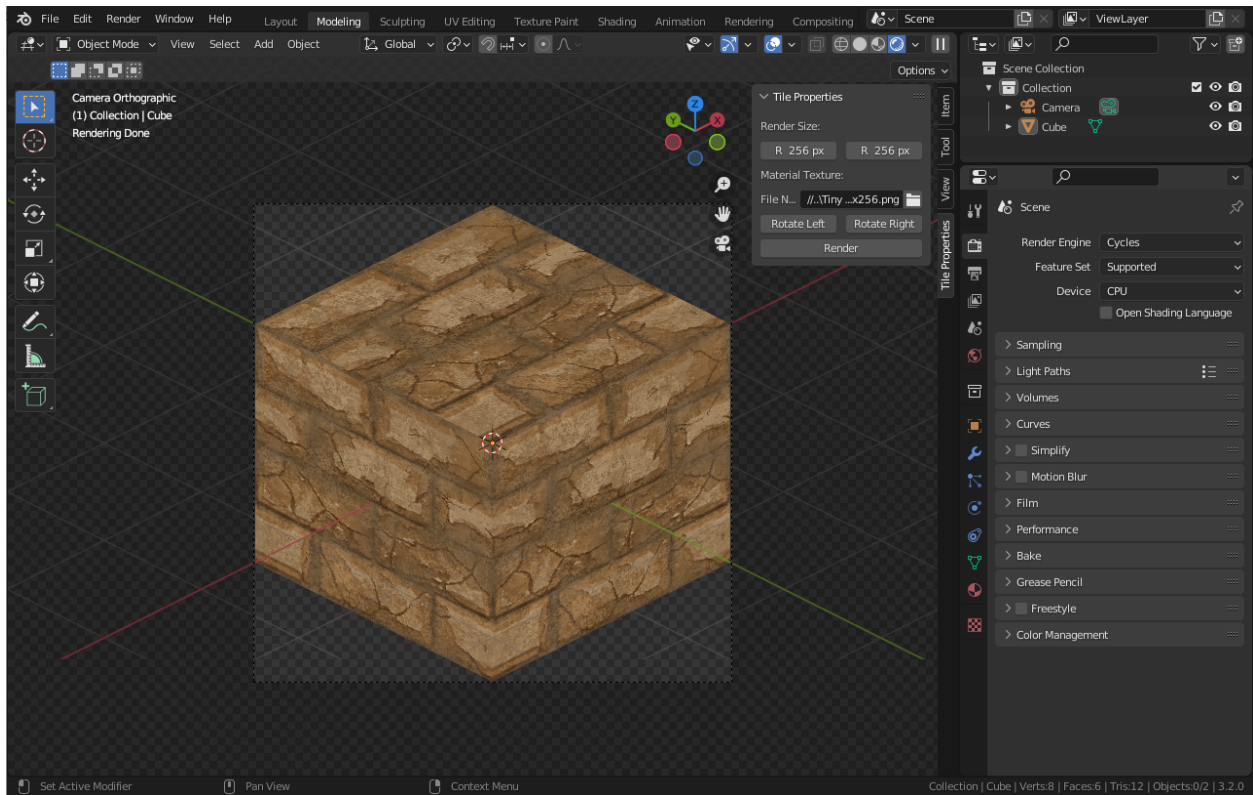


This Blender scene is set up to render Isometric Wall tiles using the texture loaded into the Image Texture node of the 3D Plane's material shader. The Render Settings and Camera Angle have all been set up to produce a perfect 2:1 tile using any texture you choose. You can set the size of the final rendered tile by changing the values of the X and Y fields in the Tile Properties panel. It is recommended that you use the following 2:1 resolutions for Walls (512x768, 256x384, 128x192, 64x96, etc.) to maintain the proper render frame around the Tile.

1. Set your desired tile size using the X and Y fields in the Tile Properties panel
2. Click the Folder icon to the right of "File Name" to select a texture
3. Use the Left Wall and Right Wall buttons to flip the view
4. Press the Render button to render and export your Isometric Tile

The Isometric Wall Maker scene can flip the texture plane on the Z axis allowing you to render both Left or Right Walls.

# Isometric Cube Maker



This Blender scene is set up to render Isometric Cube tiles using the texture loaded into the Image Texture node of the 3D Cube's material shader. The Render Settings and Camera Angle have all been set up to produce a perfect 2:1 cubic tile using any texture you choose. You can set the size of the final rendered tile by changing the values of the X and Y fields in the Tile Properties panel. It is recommended that you use resolutions that are multiples of two (512x256, 256x128, 128x64, 64x32, etc.) to maintain the proper render frame around the Tile.

1. Set your desired tile size using the X and Y fields in the Tile Properties panel
2. Click the Folder icon to the right of "File Name" to select a texture
3. Use the Rotate Right and Rotate Left buttons to rotate the view
4. Press the Render button to render and export your Isometric Cube

The Isometric Cube Maker scene can rotate the texture cube on the Z axis allowing you to adjust the top face of the cube.

**NOTE:** In most cases, using a texture that is larger than your desired render size will produce the best results. If you do not have a texture that is large enough, try using textures that are one half the size of your desired render size. For example, use a 32x32 texture if you wish to render a 64x64 cube, or a 64x64 texture if you are rendering a 128x128 cube.

The tiles produced using the **Isometric Tile Maker for Blender** scenes do not use the same Isometric Corner Style as the other Screaming Brain Studios isometric asset packs, however due to the similarities, they will line up perfectly with any existing tiles from the Isometric Floor Pack, Isometric Wall Pack, or Isometric Overworld Pack with no issues.

## Isometric Camera Scripts

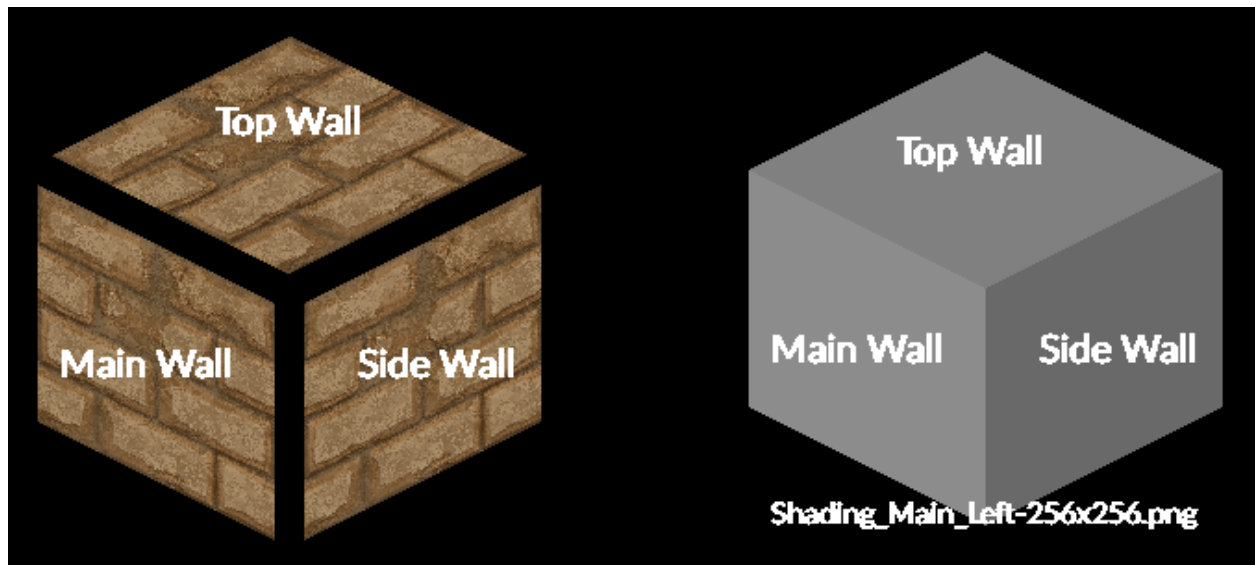
In addition to Blender scenes, there are also several Python (\*.py) scripts included with the toolkit. Each script is prepared to generate a scene, including a camera and a 3D mesh, render settings, and a custom properties panel. You can dissect the scripts or modify them to suit your needs, or you can just import and run them to generate a new Tile Maker scene in Blender. Be aware, each script will select and clear any other objects that are already in the scene, which makes it useful for placing in brand new Blender project files since it saves you from having to delete or modify the default camera and cube every time you create a new scene.

- *Isometric Floor.py*
- *Isometric Wall.py*
- *Isometric Cube.py*

To use these scripts, simply create a new Blender scene and switch to the Scripting Editor. You can access the Text/Scripting Editor by selecting the Tab at the top of the workspace or by pressing Shift+F11. Press the “New Text” icon to create an empty script to copy/paste into or select the “Open Text” button to manually select the script and import it. Lastly, click the “Run Script” button (the one that looks like a Play button) to run the script. Each script can be saved with the Scene to be loaded and used again later.

# Shading and Selection Masks

In order to apply lighting or shading to your tiles, you normally have to place a light in your scene. However, due to the way Walls and Floors are rendered separately from one another, it can sometimes be tricky to maintain uniform lighting between different Blender scenes. In order to remedy this, a series of Shading Masks has been included with the toolkit that allow you to quickly apply preset shading values to any of your tiles. There are also multiple angles of preset shading masks, allowing you to select from Side-lit or Top-lit cubes, depending on your needs. These masks are mainly intended to be used with Cube tile renders, though you can edit them and use them for Floor or Wall tiles should the need arise.



The Masks have all been named using a specific naming convention. For example, let's take a look at the mask named "Shading\_Main\_Left-256x256.png" to get an idea how the files have been named. The first part of the filename, "Shading," indicates the type of Mask, which in this case would be a Shading Mask. The second part of the filename, "Main," indicates the particular Face of the cube that will be *unaffected* by the shading mask. The third part of the filename, "Left," refers to the *direction* the pseudo-lighting will be coming from. For this particular mask, the lighting tag is "Left," meaning the simulated lighting or shading effect will appear to be coming from the 'left' side of the canvas. The final part of the filename refers to the Dimensions of the Mask itself, which in this case is "256x256." Be sure to select the Mask that corresponds to the size of the Tile you are working on!

**NOTE:** The *unaffected* side of the shading mask uses an RGB value of (128,128,128) so that it applies absolutely no shading to that side of the cube. Any value above (128,128,128) will lighten the cube face, while any value below (128,128,128) will darken it. Masks named "Shading\_Main" will leave either the left or right side walls unaffected, while the masks named "Shading\_Top" will leave the top face of the cube unaffected.

In order to utilize the Shading Masks correctly, you will need a photo editing tool like **GIMP** or **Photoshop**, but for the sake of keeping everything FREE and accessible, we will stick with using **GIMP**.

1. Open your rendered Tile in GIMP
2. Add a New Layer above the Tile layer
3. Import or paste the Shading Mask into the new layer
4. Change the Blend Mode to “Grain Merge,” “Overlay,” or “Soft Light” for the best results

In addition to the Shading Masks, also included with the toolkit is a series of Selection Masks that can be used to work on individual faces of Cube tile renders. For example, if you wish to only apply lighting, filters, or any sort of effect to one side of a Cube but not the others, the Selection Masks allow you to isolate each side from the rest. These are also useful for combining multiple Cube tile renders into a single cube. For example, if you were to render a Brick Wall texture, your cube will have the same texture applied to the Top, Side, and Main walls of the cube. If you wish to have a different texture on the Top face of the cube, you would use a Selection Mask to isolate the Top face and replace it with a different Cube tile texture.

## Tile Sheet Maker

To make use of your newly rendered tiles, the toolkit also comes with a handy script for **GIMP** that will automatically create Tile Sheets for you to use with the game engine or map editor of your choice! Once you install the script to your Plug-ins folder, you will be able to use the Tile Sheet Maker from the Filters menu.

1. Locate the Plug-ins folder for your **GIMP** installation (C:\Program Files\GIMP 2\lib\gimp\2.0\plug-ins or C:\Users\YourUsername\AppData\Roaming\GIMP\2.10\plug-ins)
2. Copy the *Tile Sheet Maker.py* file to the Plug-ins folder
3. Open **GIMP** and find the *Tile Sheet Maker* in the *Filters* menu
4. Select the folder containing your rendered tiles
5. Set the Tile Size to correspond with the size of your rendered tiles
6. Set the number of columns you want in your tile sheet (This will also determine how many rows are created)
7. Generate the Tile Sheet

This script should produce tile-sheets with up to 8 tiles across (referred to as columns) and will determine how many rows of tiles there should be based on the total number of tiles divided by 8. Tiles will be arranged one after another based on their filenames, so if you want certain tiles to line up in a specific order, it is recommended that you use a 3-digit numbering system. When using a 2-digit numbering system, it is possible for tiles to be organized in such a way that they are placed in different orders in the tile-sheet. Try using names like “Filename\_001.png” to maintain the correct ordering of your tiles.

**NOTE:** This toolkit is a work in progress, and will likely be updated or modified with additional content, scripts, or blender scenes.