Versions and how to install them

The following parts of this document clarify the installation process of the model_to_svg plugin in Blender 2.90. The full version of this plugin is using the Shapely Python package, therefore it is necessary to install it into the Python version that is bundled with Blender. This process can be complicated because Blender (as of the version 2.90) doesn't offer any easy method of installing modules into its version of Python. Because of that there are 2 available versions of this plugin:

1. **`model_to_svg_full.py`** – standard version of the plugin which requires Shapely Python package to be installed in Blender's Python and is more difficult to install

2. **`model_to_svg_lite.py`** – lite version of the plugin that contains only 1 method for cutting conflicting polygons but doesn't require any dependencies to be installed

If you want to install the **full plugin**, follow 1) Installing the dependencies and then 2) Installing the plugin (does not work whenever Blender is installed in read-only folders, for example when Blender is installed using Snap on Linux or when it is located in a folder which the user doesn't have permissions to modify).

If the **lite version** is enough, skip to 2) Installing the plugin.

If Blender implements an easier method of installing modules into its Python version in the future (or if you know about such a method), feel free to use it to install the Shapely Python package and skip to the 2) Installing the plugin section and install the full plugin.

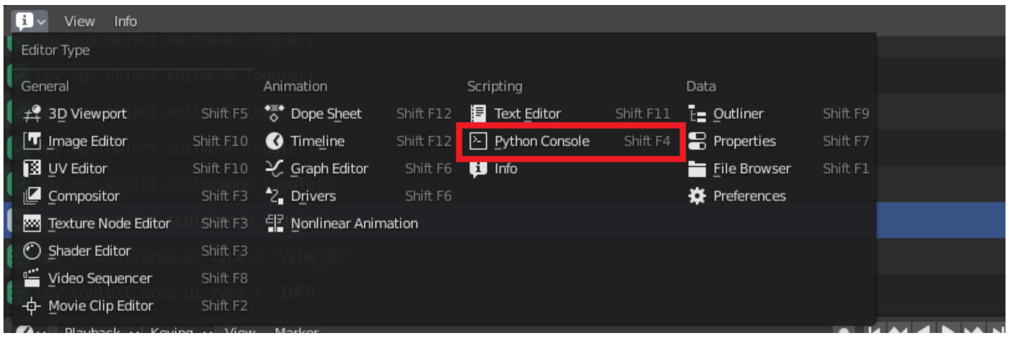
1) Installing the dependencies

Depending on the machine configuration and whether a previous version of Shapely has already been installed, the **a) Console Shapely install steps might not be able to install Shapely correctly. If that is the case, try following the b) Manual Shapely install section**.

a) Console Shapely install

In order to install the Shapely package via Python console (or any other Python package for that matter), follow these steps:

1. Run Blender (preferably with elevated privileges, for example as an administrator on Windows).
2. Open a window with "Python Console" Editor Type (Shift+F4).



3. Type/Paste the following commands into the console:

   ```
   import subprocess
   ```
   (Imports subprocess module)

   ```
   PY_EXE = bpy.app.binary_path_python
   ```
   (Gets the Blender Python path, ignore the warning on Linux)

   ```
   subprocess.call([str(PY_EXE), "-m", "ensurepip", "--user"])
   ```
   (Ensures pip is installed)

   ```
   subprocess.call([str(PY_EXE), "-m", "pip", "install", "--upgrade",
   "pip"])
   ```
   (Ensures pip is up to date)

   ```
   subprocess.call([str(PY_EXE),"-m", "pip", "install", "shapely"])
   ```
   (Installs Shapely)

b) Manual Shapely install

In order to install the Shapely package manually, follow these steps:

1. Download the respective Shapely source files **`shapely_windows.zip`** or **`shapely_linux.zip`** for your operating system bundled with the plugin from the https://github.com/Craszh/BlenderModelToSVG repository.
2. Unzip the archive and copy the source folders (**`shapely`** and **`Shapely-1.7.1.dist-info`**) into the Blender Python packages folder ("2.90" changes based on the installed version of Blender):
   <pathToBlender>\Blender 2.90\2.90\python\lib\site-packages
   (note that the location and folder structure might vary among different installations and operating systems, what's important is to find the site-packages folder of Blender's Python)

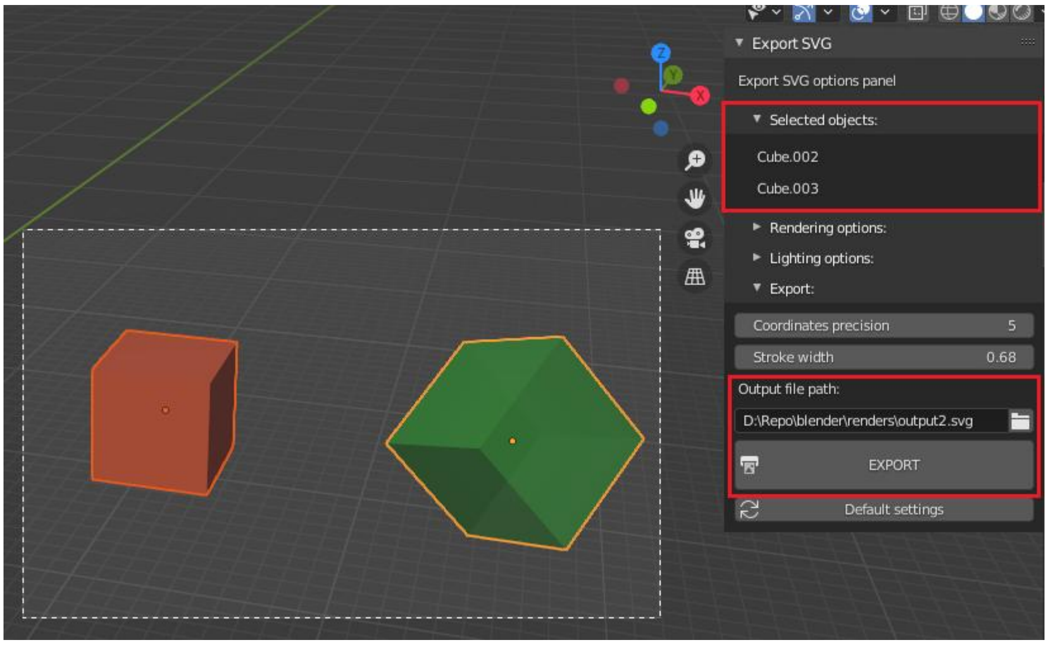
2) Installing the plugin

1. Download the **`model_to_svg_full.py`** or **`model_to_svg_lite.py`** file from the https://github.com/Craszh/BlenderModelToSVG repository.
2. Open Blender and go to Edit -> Preferences -> Add-ons.
3. Press install, select the downloaded Python file and activate the newly displayed addon (when installing the full version: if Shapely has been installed correctly, the addon will activate, otherwise an exception will be displayed saying that module named shapely was not found).

How to use

After the plugin is activated, a new panel (named Export SVG) will be displayed on the right side of the 3D Viewport window.

In order to convert objects in the scene, select them in the viewport and click on Export in the panel after selecting the output path. These objects must be of type “MESH”. If the export was successful, a success message will be displayed and the SVG file will be generated on the selected file path:



Export SVG

Export SVG options panel

Selected objects:

Cube.002

Cube.003

Rendering options:

Lighting options:

Export:

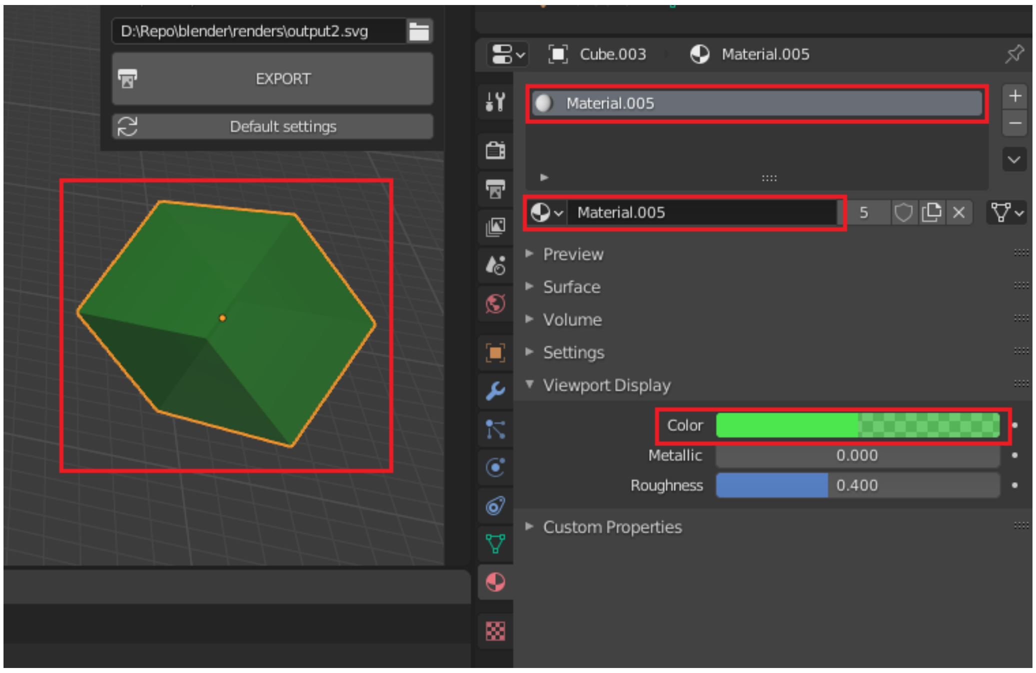
Coordinates precision 5

Stroke width 0.68

Output file path:

D:\Repo\blender\renders\output2.svg

To define the base color of a material used during the conversion, edit the Viewport Display Color property of the selected material (both the RGB color and the opacity/alpha):



D:\Repo\blender\renders\output2.svg

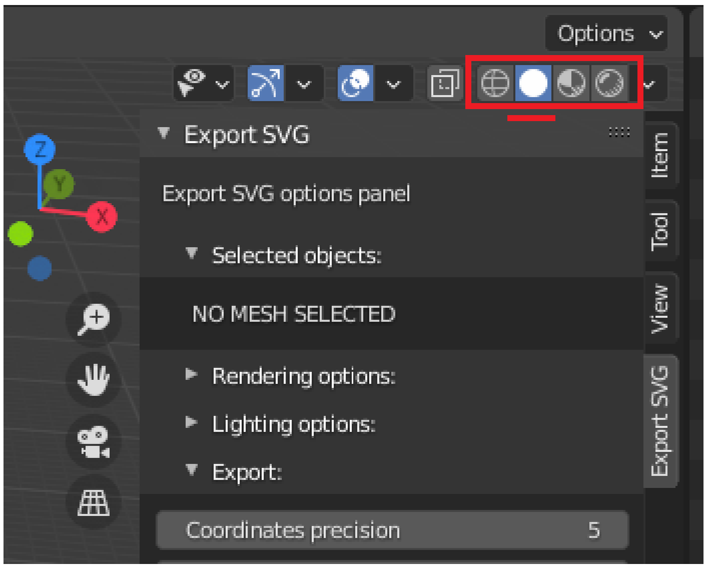
Color

Metallic 0.000

Roughness 0.400

The resulting SVG file will have the same height and width as the 3D Viewport window at the time of the conversion, essentially becoming a vector “photo” of the viewport.

It is recommended to view the scene in the Solid Viewport Shading mode to get a better idea of how the final image is going to look like.



Export SVG

Export SVG options panel

Selected objects:

NO MESH SELECTED

Rendering options:

Lighting options:

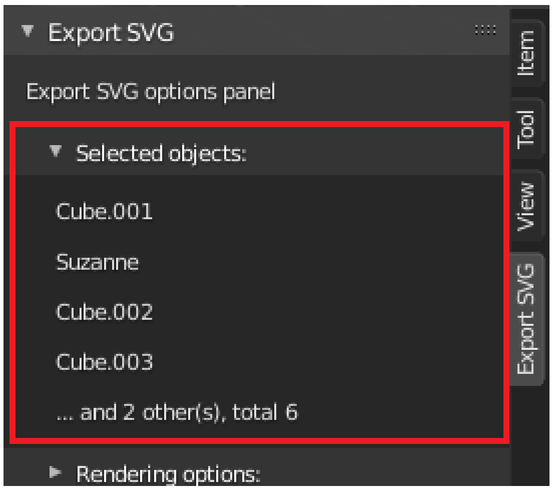
Export:

Coordinates precision 5

Settings explanation

The following section contains descriptions of all individual plugin settings. It is also possible to hover over the desired UI element in order to display its description while inside Blender.

Selected objects section



Export SVG

Export SVG options panel

Selected objects:

Cube.001

Suzanne

Cube.002

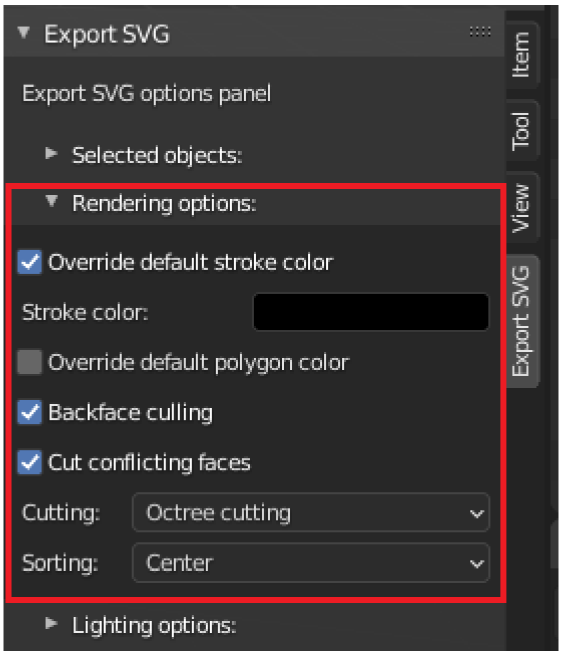
Cube.003

... and 2 other(s), total 6

Rendering options:

This section displays all currently selected “MESH” type objects that are supposed to be converted. In order to not get too large, the number of displayed objects is limited to 4. After that only the total number of selected objects will be displayed. If no “MESH” object is selected, a message “NO MESH SELECTED” is displayed instead.

Rendering options section


Export SVG

Export SVG options panel

Selected objects:

Rendering options:

✓ Override default stroke color

Stroke color: ■

Override default polygon color

✓ Backface culling

✓ Cut conflicting faces

Cutting: Octree cutting

Sorting: Center

Lighting options:

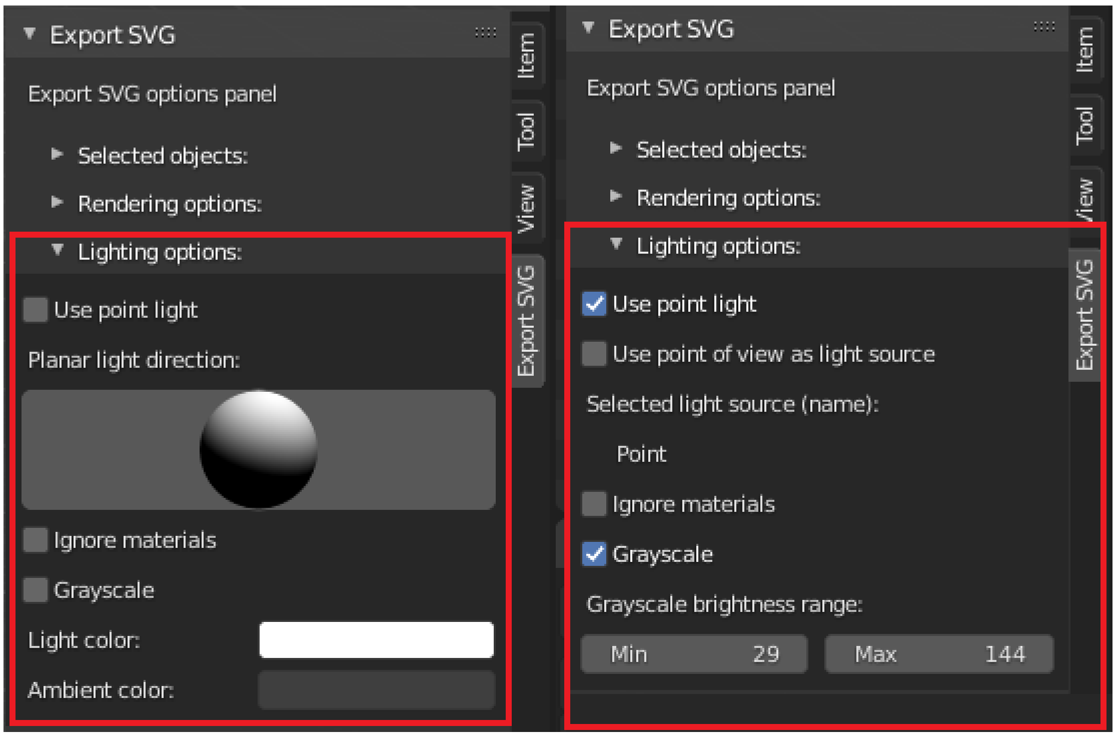
This section contains several rendering options:

- **Override default stroke color** – by default all polygon strokes are of the same color as the polygon and therefore are not visible in the final image. This can be overridden using this option which changes all polygon strokes to the desired color. The selection of this color will only appear after ticking this option.
- **Override default polygon color** – by default all polygon fill colors are calculated based on the selected lighting options and the color of the face's material. This can be overridden using this option which changes all polygon fill colors to the desired color. The selection of this color will only appear after ticking this option. (It is also recommended to override the stroke color when overriding the fill color, otherwise the entire image will only be of one color.)
- **Backface culling** – by default all faces of a model are converted, including those not facing the camera. By ticking this option all faces not facing the camera will be ignored.
- **Cut conflicting faces** – by default all faces are converted as they are laid out in the scene. This option allows the user to select one of the three methods for dealing with colliding or overlapping polygons by cutting them into smaller fragments. After ticking this box, the selection of the **cutting** method will appear, allowing the user to choose one of the following:
    - **Octree cutting** – Uses an octree for detecting and cutting polygons, default method.
    - **(EXPERIMENTAL Newell cutting)** – Detects and cuts conflicting polygons based on Newell's algorithm. This method is not fully functional and can cause infinite cutting loops in certain configurations and therefore crash Blender. However it can also sometimes produce more precise results than the octree cutting method. When trying this method, it is advised to have the console window opened in case a keyboard interrupt (Ctrl+c) is required to forcibly stop the looped cutting process.
    - **BSP Tree** – Uses a BSP tree to partition and sort the scene. Produces the most precise results. However because of the SVG format in which all polygon cuts are visible, it can create many visible gaps or overlaps especially when converting transparent models.
        - After selecting this option, the Sorting option will disappear and instead the **BSP cycles limit** option will be available. This option defines the maximum number of

space partitions. More complex and larger scenes require more cycles, however more cycles require much more memory and time. The default number (500) should be enough to convert all simple scenes up to a few thousand polygons.

- **Sorting** – defines the rule used for back-to-front depth sorting of all polygons in the scene:
    - **Center** – sorts all polygons based on the center of their Z coordinates, default rule.
    - **Closes vertex** – sorts all polygons based on the Z coords of their closest vertex.
    - **Furthest vertex** – sorts all polygons based on the Z coords of their furthest vertex.
    - **Weighted center** – sorts all polygons based the weighted center of their Z coords.

Lighting options section



Export SVG

Export SVG options panel

▸ Selected objects:

▸ Rendering options:

▾ Lighting options:

☐ Use point light

Planar light direction:

☐ Ignore materials

☐ Grayscale

Light color:

Ambient color:

Export SVG

Export SVG options panel

▸ Selected objects:

▸ Rendering options:

▾ Lighting options:

☑ Use point light

☐ Use point of view as light source

Selected light source (name):

Point

☐ Ignore materials

☑ Grayscale

Grayscale brightness range:

Min 29 Max 144

This section contains several lighting options:

- **Use point light** – by default, a planar light source is used. Ticking this option switches to point light source instead. When planar light is used, the **Planar light direction** option is available. This allows the user to specify from which direction the objects in the scene should be highlighted. When point light is used instead, the light source options become available:
    - **Use point of view as light source** – ticking this option means that the current camera location will be used as the light source location. If this option is not selected, the **Selected light source (name)** element is visible.
    - **Selected light source (name)** – much like the Selected objects section, this element displays the name of the selected object in the scene that will be used as the light source location. This object must be of type “LIGHT” and only one can be used. (If more “LIGHT” objects are selected, only one of them will be considered.) If no “LIGHT” object is selected and the point of view is not used as a light source, this element will display the “NO LIGHT SELECTED” message instead.

- **Ignore materials** – by default, the base color of models is defined by the Viewport Display Color property of their material. By ticking this option the material will be ignored and the resulting color will be calculated as if the material was white and opaque.
- **Grayscale** – by default, the resulting image is colored based on the base colors of materials and lighting options. By ticking this option the resulting image will be in grayscale.

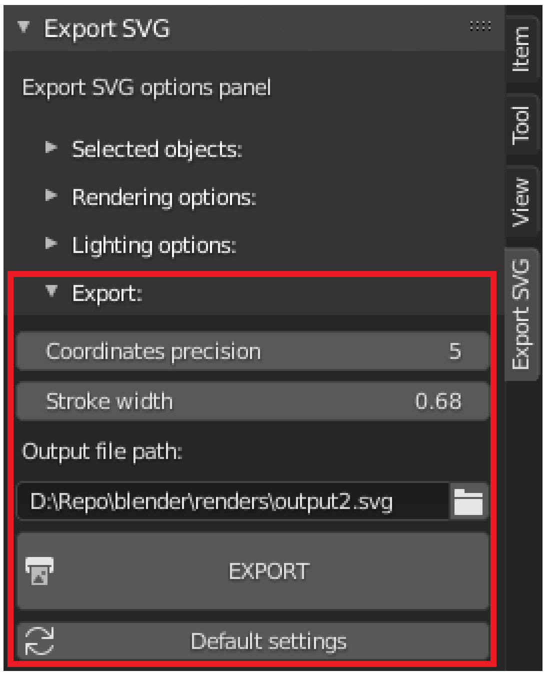
If grayscale is not selected, the light color options become available:

- **Light color** – defines the color of the light emitted by the light source.
- **Ambient color** – defines the color of the ambient light in the scene in order to reduce the contrast of shadows and highlights.

If grayscale is selected, the brightness range option becomes available:

- **Grayscale brightness range** – sets the minimum and maximum brightness values of the resulting polygons in order to reduce the contrast of shadows and highlights.

Export section



Coordinates precision 5
Stroke width 0.68
Output file path:
D:\Repo\blender\renders\output2.svg

This section contains several export-related elements:

- **Coordinates precision** – defines the number of decimals used for the polygon vertex coordinates in the file. Smaller precision results in smaller SVG files and vice versa.
- **Stroke width** – defines the width of the polygon strokes in the SVG file. Thinner strokes might lead to visible gaps between polygons while thicker strokes might create artifacts near model vertices and edges.
- **Output file path** – defines the path on which the SVG file will be generated. If the filename is missing the .svg extension, it will be appended. If a .svg file already exists, it will be overwritten.
- **EXPORT** – clicking this button starts the conversion process, after which the .svg file will be generated based on the options and selected objects.
- **Default settings** – clicking this button resets all options except the output file path to their default values (this action asks for confirmation first).