User Guide: Scene To SVG 2.0 Add-on

Hello and thank you for checking out the **Scene To SVG 2.0** add-on for **Blender 3.2+**. This is an upgraded version of my previous add-on (originally named Model To SVG as it was only used for exporting meshes) with many newly added features that increased its complexity.

Because of that, the functionality of some parts might be hard to understand just from the Blender menu itself and its descriptions. This guide should contain more detailed explanations of every setting and feature in case you get lost.

The guide itself is split into 2 parts:

1. A **Quick Start** section which serves as an introduction for new users. This part explains how to install the add-on, how it works, what are its main features, and simple explanations of the basic workflow.
2. A **Settings Explanation** section which provides more detailed explanations of every available setting of the add-on, divided into sections:
    1. MESH Options
    2. MESH Lighting
    3. CURVE/GPENCIL Options
    4. FONT Options
    5. Cameras
    6. Export
    7. Material > Export SVG Properties
    8. Material > Export SVG CSS Animation

1) Quick Start

How to install scene_to_svg.py into Blender

Note: The add-on comes in two versions, the basic one (**scene_to_svg.py**) and the special one (**scene_to_svg_extra_cutting.py**). The basic one contains 99 % of all the add-on's features. The only additional feature of the special version are 2 extra methods for cutting overlapping/clipping polygons of conflicting models/meshes, however this version requires an additional external library (shapely) installed into Blender Python. This section will explain installation of only the basic version as installing an external library in Blender Python is a bit tricky. If you really need the additional feature, you can find the install guide of the special version at the end of this document.

1. **Download** the source file of the Scene To SVG add-on (**scene_to_svg.py**) 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 add-on.

How the add-on works

The add-on allows you to select objects of a certain type in a scene and export them to SVG by generating a new .svg file in the specified directory. It does so in a form **similar to a screenshot** – the object in the SVG image will appear to be drawn from the same angle as from which you viewed it when you pressed export (also supports both perspective and orthographic views).

On top of that the add-on provides additional features such as styling the exported objects or even animating some of their visual properties by generating CSS animations (see individual settings for all the available features).

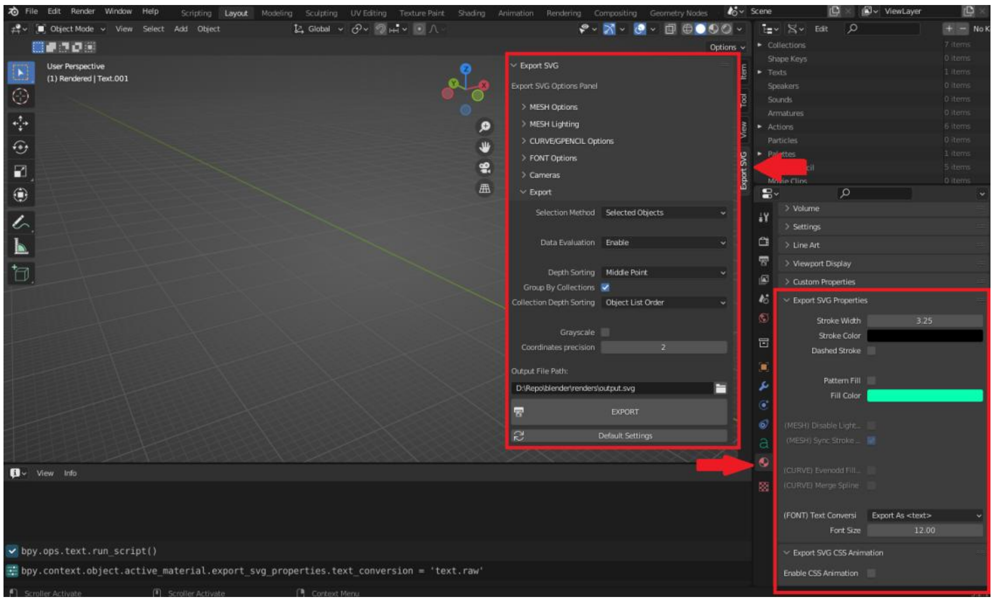
The add-on can export the following types of objects into SVG:

Blender type	Converts to
MESH	<polygon>
CURVE (Bezier only)	<path>
GPENCIL	<path>
FONT	<polygon>, <path>, <text>

When exporting MESH types, it takes each face of the mesh and converts it to a <polygon> element. When exporting CURVE/GPENCIL types, it takes each spline/stroke and converts it to a <path> element (while retaining the order of layers for gpencil). When exporting FONT types, you can select whether to convert it into a series of <polygon> or <path> to keep the shape or a very simple <text>.

Where is the add-on UI located

The UI of the add-on consists of **2 main panels**, one in the **3D view** containing global settings and one in the **material properties** section containing material specific settings.


Export SVG

Export SVG Options Panel

- MESH Options
- MESH Lighting
- CURVE/GPENCIL Options
- FONT Options
- Cameras
- Export

Selection Method: Selected Objects
Data Evaluation: Enable
Depth Sorting: Middle Point
Group By Collections: ✓
Collection Depth Sorting: Object List Order
Grayscale: ☐
Coordinates precision: 2
Output File Path: D:\Repo\blender\renders\output.svg
EXPORT
Default Settings

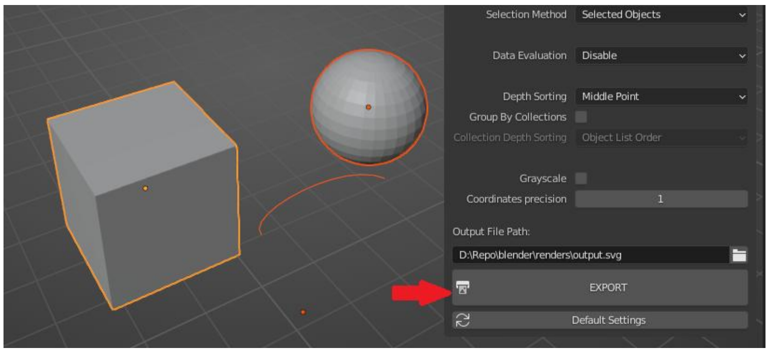
Export SVG Properties
Stroke Width: 3.25
Stroke Color: ⬛
Dashed Stroke: ☐
Pattern Fill: ☐
Fill Color: 🟩
(MESH) Disable Light...: ☐
(MESH) Sync Stroke ...: ✓
(CURVE) Evenodd Fill...: ☐
(CURVE) Merge Spline: ☐
(FONT) Text Conversi: Export As <text>
Font Size: 12.00

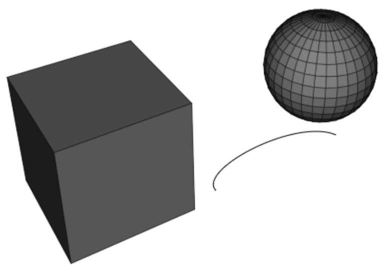
Export SVG CSS Animation
Enable CSS Animation: ☐

```
bpy.ops.text.run_script()
bpy.context.object.active_material.export_svg_properties.text_conversion = 'text.raw'
```

How to do a simple export with default settings

With **default settings**, you can simply **select the exported objects** and **click the EXPORT button** at the bottom of the panel. Objects will be converted from the angle you are viewing them from into a new .svg file with the **path and filename you specify** right above the button (overwrites existing files).

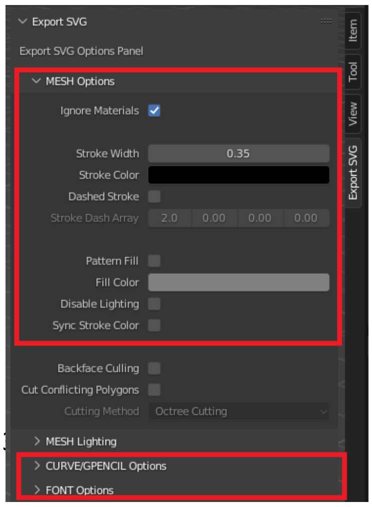

Selection Method: Selected Objects
Data Evaluation: Disable
Depth Sorting: Middle Point
Group By Collections: ☐
Collection Depth Sorting: Object List Order
Grayscale: ☐
Coordinates precision: 1
Output File Path: D:\Repo\blender\renders\output.svg
EXPORT
Default Settings



How to style/change the appearance of objects

The add-on uses SVG/CSS attributes to change visual aspects of exported polygons, curves and text. There are **two ways** you can change an object's appearance depending on whether the object has a material assigned to it: the **global settings** or the **material settings.**

1) Global settings


Export SVG

Export SVG Options Panel

MESH Options
Ignore Materials: ✓
Stroke Width: 0.35
Stroke Color: ⬛
Dashed Stroke: ☐
Stroke Dash Array: 2.0 | 0.00 | 0.00 | 0.00
Pattern Fill: ☐
Fill Color: ▨
Disable Lighting: ☐
Sync Stroke Color: ☐
Backface Culling: ☐
Cut Conflicting Polygons: ☐
Cutting Method: Octree Cutting

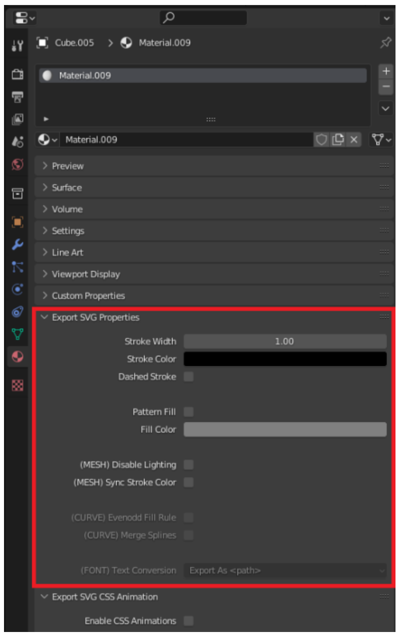
MESH Lighting
CURVE/GPENCIL Options
FONT Options

Global settings are settings contained in the main 3D view sidepanel split by type into MESH/CURVE+GPENCIL/FONT sections.

The attributes you specify in these sections **will be applied globally to all objects of that type that do NOT have any material assigned**.

If you check the Ignore Materials option, the settings will be applied to every object of that type even if it has a material assigned.

2) Individual material settings



Stroke Width	1.00
Stroke Color	■
Dashed Stroke	☐
Pattern Fill	☐
Fill Color	▨
(MESH) Disable Lighting	☐
(MESH) Sync Stroke Color	☐
(CURVE) Evenodd Fill Rule	☐
(CURVE) Merge Splines	☐
(FONT) Text Conversion	Export As <path>
Enable CSS Animations	☐

Individual material settings are settings contained in the Export SVG Properties subpanel of Material Properties.

The attributes you specify in this section will be applied to the material you specify them for, therefore they will **affect the appearance of every object/stroke/face/… that this material is assigned to**.

Remember that checking the Ignore Materials option will result in these settings being ignored for that specified type.

Note however that not all global options are available in this panel; settings like Backface Culling or Cutting are only available globally.

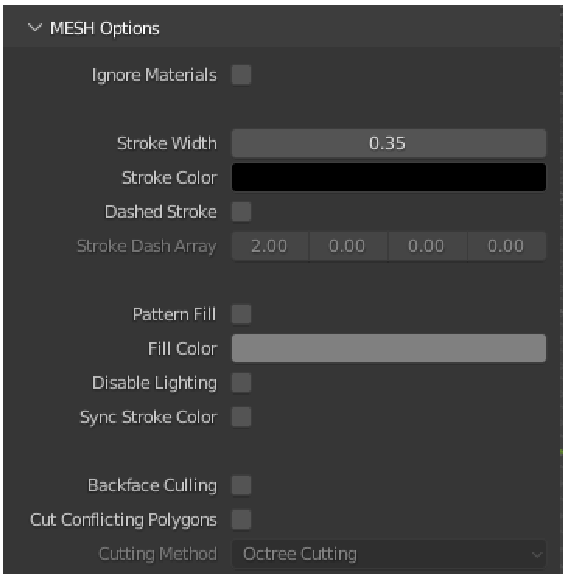
Where to find more information

The explanations above should give you a basic idea of how to use the add-on. If you wish to know more and make better use of all the features the add-on offers, feel free to play around with all the available settings. **Hovering over any setting** of the add-on menus will give you a **more detailed description** of what it does. If the explanation isn't clear enough, try **looking up the setting in the following section** of this user guide containing more detailed explanations.

2) Settings Explanation

The following section provides more detailed explanations regarding all the add-on's more complicated settings. All settings are listed in the same order as they appear in the add-on's UI, split into subsections based on the add-on's subpanels:

MESH Options



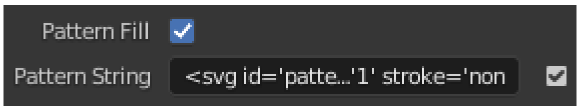
MESH Options

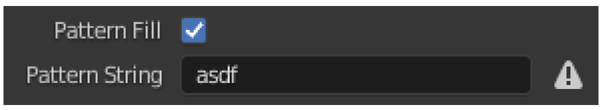
Ignore Materials	☐
Stroke Width	0.35
Stroke Color	■
Dashed Stroke	☐
Stroke Dash Array	2.00 · 0.00 · 0.00 · 0.00
Pattern Fill	☐
Fill Color	▨
Disable Lighting	☐
Sync Stroke Color	☐
Backface Culling	☐
Cut Conflicting Polygons	☐
Cutting Method	Octree Cutting

- **Ignore Materials** – Disables individual material settings of all MESH type objects which means the settings below will be applied to all meshes regardless of whether they have any material assigned
- **Stroke Width, Stroke Color, Dashed Stroke, Stroke Dash Array** – These options change stroke attributes of meshes, which means the outline of every individual polygon (note that you can also change **Alpha** of Stroke Color, which affects the stroke's **opacity**)
- **Pattern Fill** – If unchecked, allows setting **Fill Color**, if checked, allows you to set a pattern as the polygon's fill and show the setting **Pattern String**
    - **Fill Color** – This option changes the fill color of individual polygons (note that you can also change **Alpha** of Fill Color, which affects the fill's **opacity**)
    - **Pattern String** – This option allows you to paste in an SVG formatted string containing <pattern> definition (generated by an online tool[1] for example). This

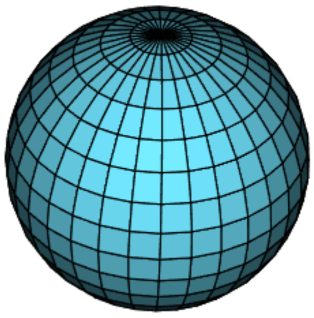
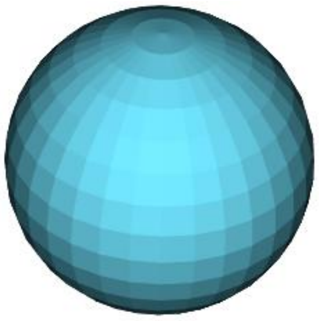
[1] For example https://pattern.monster/, https://10015.io/tools/svg-pattern-generator, https://fffuel.co/ooorganize/ and others allow you to create a pattern and copy the SVG string to clipboard

<pattern> will then be applied as the polygon’s fill. A different **icon will be displayed** based on whether the pasted string has a **valid <pattern>** or not. WARNING: Patterns work when displaying the svg in a web browser but cannot be displayed or exported when opening the file in a 2D editor like Inkscape.


Pattern Fill ✓
Pattern String <svg id='patte...'1' stroke='non ✓


Pattern Fill ✓
Pattern String asdf ⚠

- **Disable Lighting** – By default, meshes (more specifically the fill colors of polygons) are affected by primitive lighting. If this option is **checked**, all polygons will have the **same Fill Color** unaffected by light or angle. If this option is **not checked**, the Fill Color option will only be used as the **base/diffuse color** when calculating the final color affected by light.
- **Sync Stroke Color** – This option is only available when lighting is NOT disabled. If this option is checked, the color of strokes will not be determined by **Stroke Color** but will instead be set individually for every polygon so that it corresponds to that individual polygon’s fill color.

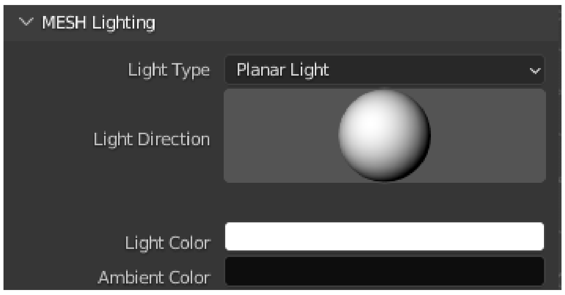



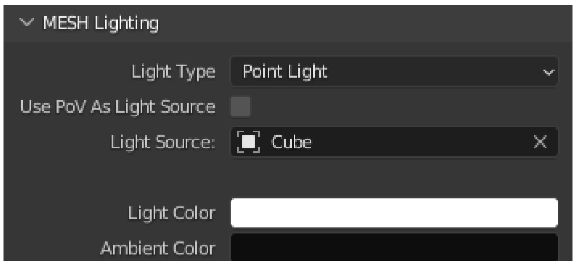
- **Backface Culling** – If checked, polygons that are not facing the camera are not exported.
- **Cut Conflicting Polygons** – If checked, polygons that are colliding and clipping through one another in the scene are detected and cut accordingly and sorted so that no conflicts remain. This should result in a more accurate SVG image that resembles the original scene more closely. This option however doesn’t work perfectly and is not always precise, especially when it comes to sorting polygons. It is also the most expensive option in the entire add-on and significantly increases the time it takes to convert the scene. Only use when you are absolutely sure you need to use it and understand what it does. Toggling this option allows you to choose which algorithm to use for cutting. The basic version of the add-on contains only one option (**BSP Tree**).
  - **Octree Cutting (only available in the special vision)** – Uses an octree for detecting and cutting polygons, default method.
  - **(EXPERIMENTAL) Newell Cutting (only available in the special vision)** – 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 **Depth Sorting** option will disappear and instead the **Partition Cycle 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 value (500) should be enough to convert all simple scenes up to a few thousand polygons.

- **Depth Sorting** – Only available when cutting is enabled and Octree Cutting or Newell Cutting is selected. Allows you to choose how to depth sort all the cut polygons.

MESH Lighting


MESH Lighting
Light Type: Planar Light
Light Direction
Light Color
Ambient Color

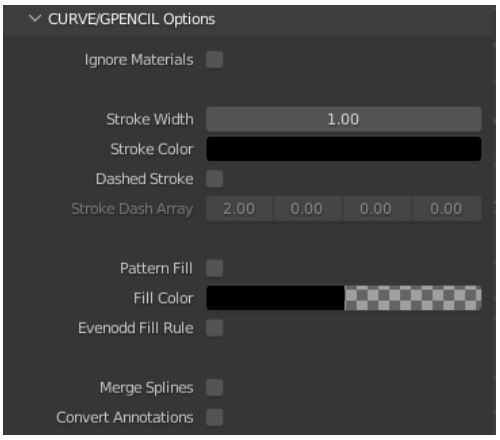

MESH Lighting
Light Type: Point Light
Use PoV As Light Source
Light Source: Cube
Light Color
Ambient Color

- **Light Type** – Allows you to select whether to use a **Planar Light** or **Point Light**.
- **Light Direction** – Only available when **Planar Light** is selected. Sets the angle under which light reaches the surface of meshes.
- **Use PoV As Light Source** – Only available when **Point Light** is selected. If checked, the current position of the **view of the scene** will be considered the **position of the point light source**. If not checked, allows the selection of **Light Source**.
    - **Light Source** – Allows you to set an **object** as the point light source. Its position will then be considered the position from which the light in the scene originates.
- **Light Color** – Sets the color of the light emitted by the light source.
- **Ambient Color** – Sets the color of the ambient light in the scene.

CURVE/GPENCIL Options

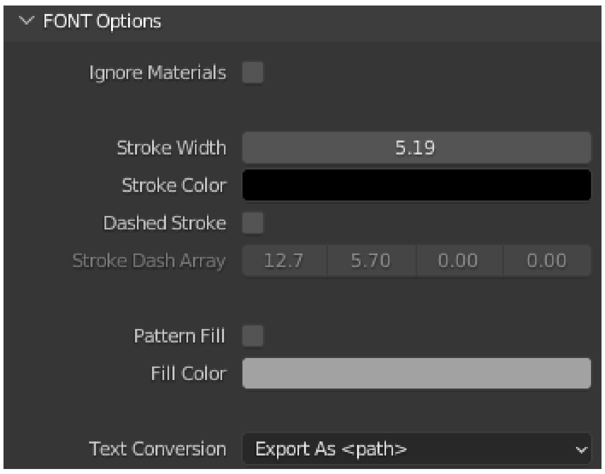
Note regarding **Curves**: Only **Bezier** type curves are supported.

Note regarding **Grease Pencils**: Grease Pencils can contain different strokes for every frame. The **frame** that will be **exported** is the **currently active frame** of the scene (which you can set on the timeline). Grease Pencils can also contain **different layers** that control which strokes are in front and which ones are behind. These **layers are exported in the same order, preserving the way their strokes overlap.**


CURVE/GPENCIL Options
Ignore Materials
Stroke Width: 1.00
Stroke Color
Dashed Stroke
Stroke Dash Array: 2.00 0.00 0.00 0.00
Pattern Fill
Fill Color
Evenodd Fill Rule
Merge Splines
Convert Annotations

- **Ignore Materials, Stroke Width, Stroke Color, Dashed Stroke, Stroke Dash Array, Pattern Fill, Fill Color and Pattern String** – These options work the same as their MESH counterparts but affect only CURVE and GPENCIL objects. Check the previous section for their explanation.
- **Evenodd Fill Rule** - If checked, sets the fill-rule attribute to "evenodd". If not checked, default SVG fill-rule is used ("nonzero"). Check https://en.wikipedia.org/wiki/Even%E2%80%93odd_rule for further explanation of this rule. Should be used only when **Merge Splines** is active.
- **Merge Splines** – If checked, **all splines** of a curve object are converted into a **single <path>** element. If not checked, all splines of a curve object are treated as separate curves and **each spline** is converted into its **own <path>** element.
- **Convert Annotations** – If checked, **all visible annotations** will also be exported as if they were a Grease Pencil object. The visual attributes of annotations cannot be set using the plugin's UI, instead their **stroke width, color and opacity is set to the same values as it is in Blender**. The annotation layer visibility and option to display them in front of other objects is also considered when exporting them. **Both 3D annotations and annotations sticked to the view are supported**.

FONT Options


FONT Options

Ignore Materials ☐

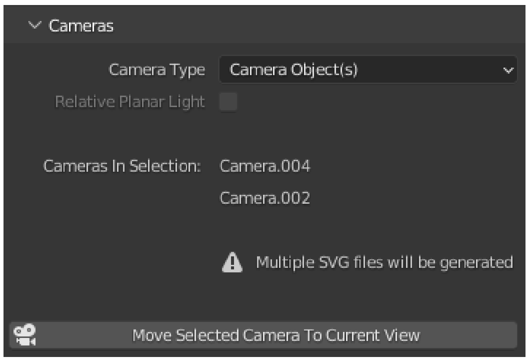
Stroke Width 5.19
Stroke Color ■
Dashed Stroke ☐
Stroke Dash Array 12.7 5.70 0.00 0.00

Pattern Fill ☐
Fill Color ▨

Text Conversion Export As <path>

- **Ignore Materials, Stroke Width, Stroke Color, Dashed Stroke, Stroke Dash Array, Pattern Fill, Fill Color and Pattern String** – These options work the same as their MESH counterparts but affect only FONT objects. Check the previous section for their explanation.
- **Text Conversion** – Defines how the text object is supposed to be exported.
  - When **exporting as <text>**, a simple <text> element with the specified **Font Size** will be generated at the text object's location. This does not retain the original text object shape.
  - When **exporting as <path>**, the text object will be **converted to a curve object** and then exported as a single <path> element. This retains the shape of the original text object.
  - When **exporting as <polygon>**, the text object will be **converted to a mesh object** and then exported as multiple <polygon> elements. This retains the shape of the original text object but result in a **very large number of polygons** even for short texts.
  - **Exporting as <path> and <polygon>** also includes the option to **rotate the text object so that it faces the camera** during the conversion.

Cameras



Cameras

Camera Type: Camera Object(s)

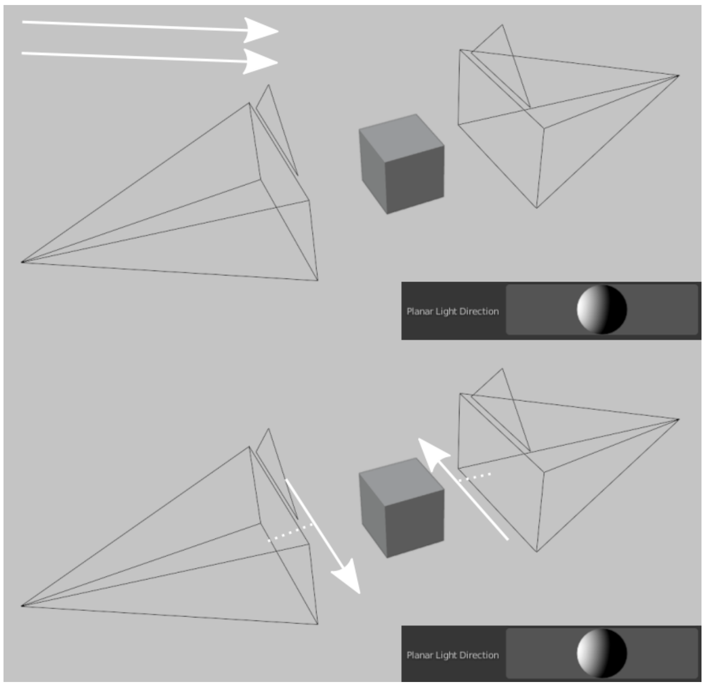
Relative Planar Light

Cameras In Selection: Camera.004
Camera.002

Multiple SVG files will be generated

Move Selected Camera To Current View

- **Camera Type** – Allows you to select whether to use the **Viewport Camera** or **Camera Object(s)** for exporting the view of the scene. If **Viewport Camera** is selected, the add-on works exactly like described in the quick start section and exports from the **active 3D view** of the scene. If **Camera Object(s)** is selected, CAMERA type objects are used instead.
    - Several Blender camera settings are supported – perspective and orthographic views, resolution (or its aspect ratio), focal length, XY shift and sensor size.
- **Relative Planar Light** – Only available when **Planar Light** is selected. Easier to explain using an image.

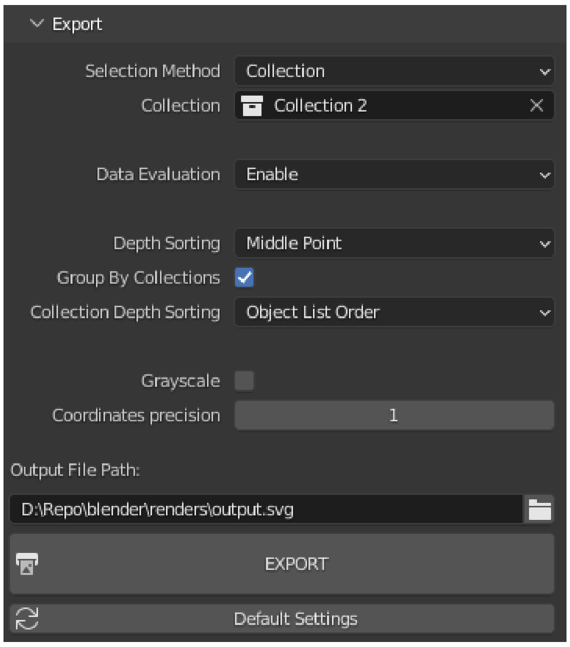


Relative Planar Light **disabled** – the Planar Light Direction for every camera **is absolute** and is determined by the **setting** and **current 3D View angle**.

Relative Planar Light **enabled** – the Planar Light Direction for every camera **is relative** and is determined by the **setting** and **current rotation of each camera**.

- **Cameras In Selection** – Only displayed if **Camera Object(s)** is selected. Shows all the cameras detected in the current selection that will have their view exported to an SVG file. **Note that you can select multiple cameras and create multiple images with one click of the Export button**. The resulting files will be named based on the original path with the camera name appended to the filename (if the camera name contains invalid characters, the file will be renamed).
- **Move Selected Camera To Current View** – Allows easier positioning of camera objects. Moves the first selected camera to the current 3D view so that the view of the camera is the same.

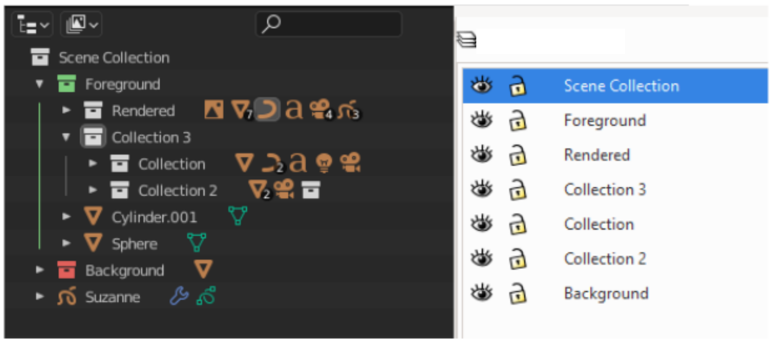
Export



Selection Method: Collection
Collection: Collection 2
Data Evaluation: Enable
Depth Sorting: Middle Point
Group By Collections: ✓
Collection Depth Sorting: Object List Order
Grayscale:
Coordinates precision: 1
Output File Path:
D:\Repo\blender\renders\output.svg

- **Selection Method** – Defines what method will be used for selecting objects (and cameras if using camera objects) for export.
  - **Selected Objects** – Only objects in the active selection will be exported.
  - **Collection** – Only objects (and subcollections) in the selected collection will be exported.
  - **Collection Visible Objects** – Only objects (and subcollections) in the selected collection that are not hidden will be exported.
    - When one of the above 2 options is selected, a new option **Collection** will appear, which allows you to select the collection which should be exported.
  - **All Objects** – All objects in the scene will be exported. Be careful when using this with complex scenes.
  - **All Visible Objects** – All objects in the scene that are not hidden will be exported. Be careful when using this with complex scenes.
- **Data Evaluation** – If enabled, objects will be **evaluated** before export, which means that some operations such as modifiers will be applied. Currently only works with types MESH and GPENCIL.
- **Depth Sorting** – Defines what rule is used for **depth sorting elements** among each other (deciding which element should be in front and which one behind). Does not have significant impact unless many objects are clustered closely on top of each other. Does not affect how polygons are sorted among each other, check MESH options for that. Allows you to select whether to depth sort objects based on the **closest/middle/furthest point of its bounding box**.
- **Group By Collections** – If checked, **every collection** of the scene and its objects will be converted into a **separate <g> element**. Objects in Blender can be part of multiple collections, but only the first one the object was assigned to will be considered its parent collection by the add-on. If not checked, all objects are part of one <g> element. If a collection does not contain any objects selected for export, it is ignored.

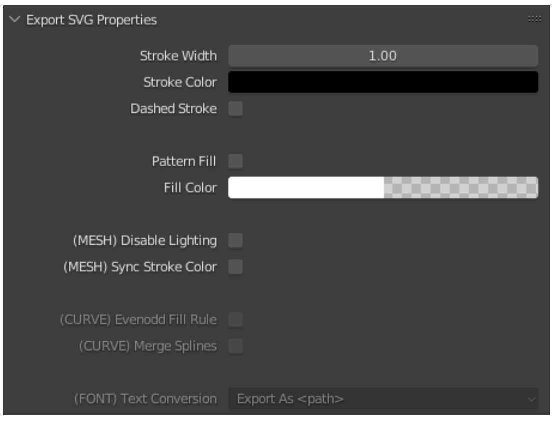
- **Collection Depth Sorting** – Only available if **Group By Collections** is enabled. Same as previous **Depth Sorting** but for sorting collections. Bounding boxes of collections are bounding boxes that encapsulate all its converted objects.
    - Compared to Depth Sorting, offers one additional option for sorting collections – **Object List Order**. If this option is selected, collections are written into the SVG file in the **same order** as they appear in the **View Layer** which allows them to be used in the same way as layers in 2D editor software as shown in the following example which compares the same layer structure in Blender, Inkscape and SVG file.



```
<svg>
<g id="Background"> ...   </g>
<g id="Collection 2"> ...  </g>
<g id="Collection"> ...  </g>
<g id="Collection 3"> ...  </g>
<g id="Rendered"> ...  </g>
<g id="Foreground"> ...  </g>
<g id="Scene Collection"> ...  </g>
</svg>
```

- **Grayscale** – If checked generates a **grayscale SVG filter** and applies it to every element in the file. WARNING: Grayscale filters work when displaying the SVG in a web browser but cannot be displayed in a 2D editor like Inkscape.
- **Coordinates Precision** – Specifies the **number of decimals** all coordinates should be rounded to in the generated file.
- **Output File Path** – Defines **path and filename** of the generated file. Automatically **appends .svg** if missing. If a file already exists on this path, it is **rewritten**.
- **EXPORT** – Starts the export when clicked.
- **Default Settings** – Resets all global options to their default values, except for Output File Path

Material > Export SVG Properties



Export SVG Properties

Stroke Width 1.00
Stroke Color
Dashed Stroke

Pattern Fill
Fill Color

(MESH) Disable Lighting
(MESH) Sync Stroke Color

(CURVE) Evenodd Fill Rule
(CURVE) Merge Splines

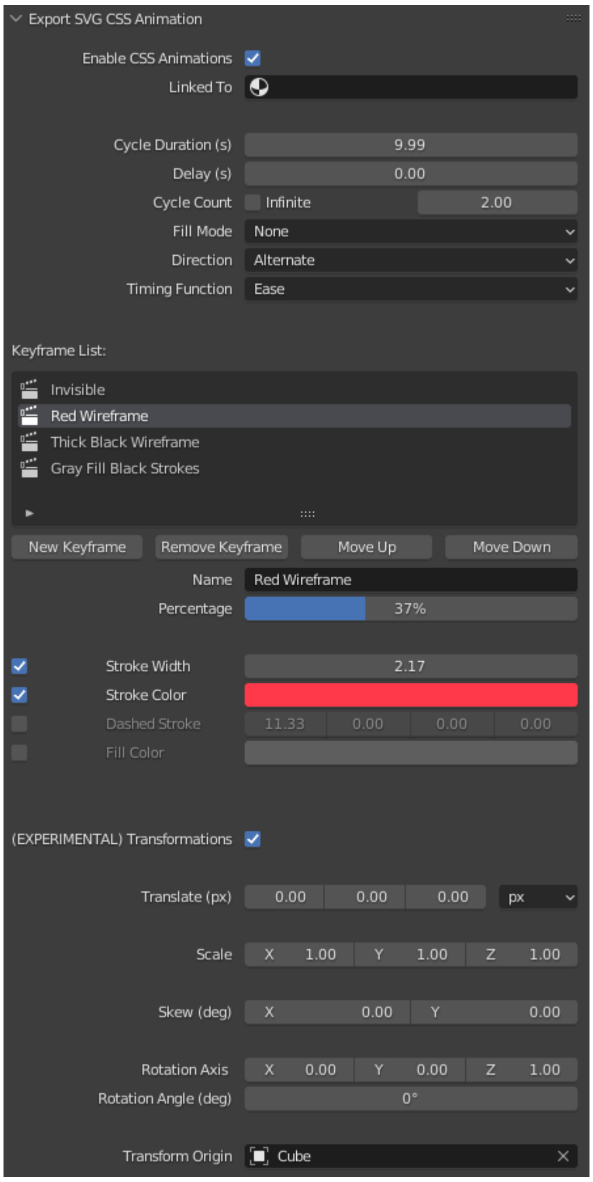
(FONT) Text Conversion Export As <path>

- All options in this section work the same way as they do in Global Settings. Some of them are disabled based on which type of object is currently selected, for clarity. Check the previous corresponding sections for their explanation: MESH, CURVE, FONT.

Material > Export SVG CSS Animation

The animation options of this add-on are just an extra feature that was tacked on while finishing it; therefore they work in a very primitive way but may still be useful in some cases. The animations themselves are tied to materials (you can create one animation per material).

The following sections allow you to setup a CSS animation sequence. This guide assumes you know how CSS animations work as the options themselves basically just copy CSS attributes from Blender menu into the SVG file. If you've never used CSS animations, it is advised to check a basic guide before trying these settings, for example https://www.w3schools.com/css/css3_animations.asp.


Export SVG CSS Animation

Enable CSS Animations ✓
Linked To

Cycle Duration (s) 9.99
Delay (s) 0.00
Cycle Count ☐ Infinite 2.00
Fill Mode None
Direction Alternate
Timing Function Ease

Keyframe List:

- Invisible
- Red Wireframe
- Thick Black Wireframe
- Gray Fill Black Strokes

New Keyframe | Remove Keyframe | Move Up | Move Down

Name Red Wireframe
Percentage 37%

✓ Stroke Width 2.17
✓ Stroke Color 🟥
☐ Dashed Stroke 11.33 0.00 0.00 0.00
☐ Fill Color

(EXPERIMENTAL) Transformations ✓

Translate (px) 0.00 0.00 0.00 px
Scale X 1.00 Y 1.00 Z 1.00
Skew (deg) X 0.00 Y 0.00
Rotation Axis X 0.00 Y 0.00 Z 1.00
Rotation Angle (deg) 0°
Transform Origin Cube

- **Enable CSS animations** – If checked, allows the customization of CSS animations and generates animation attribute and keyframes for this material in the file. WARNING: Animations work when displaying the SVG in a web browser but cannot be displayed in a 2D editor like Inkscape.

- **Linked To** – Allows **linking the animations of this material to another material**. If a material is linked to another, all other settings are disabled and all **animations properties will be copied** from the linked material. This allows you to create animations for one material and easily copy them to others.
- **Cycle Duration (s), Delay (s), Cycle Count, Fill Mode, Direction** and **Timing Function** only copy their values into the CSS animation definition. You can find their explanation in the previously linked guide or in their description.

- **Keyframe List** – Displays all keyframes defined for this material's animation. Their **order and name** in this list **do not matter**; keyframes are sorted based on their percentages when written to the file.

The following options are individual for every keyframe:

- **Name** – Name of the keyframe usable for visual clarity.
- **Percentage** – Works the same as percentages explained in the linked guide (also it is good to know that if a keyframe with **0% or 100% is not defined**, the **original appearance** of the object/material is **assumed to be the 0% or 100% keyframe respectively**).
- **Stroke Width, Stroke Color, Dashed Stroke** and **Fill Color** work the same as in the previous sections. This time they define visual attributes of elements for that specific material animation keyframe. You can specifically select which of these will be animated in this keyframe using the checkboxes on the left. These are the attributes which the animation interpolates in between keyframes.
- **(EXPERIMENTAL) Transformations** – If enabled, allows you to set up a **transformation** for the current keyframe. Supported transformations are **Translation3D, Scale3D, Rotation3D and Skew** applied in this order. This is more of an experimental feature that is a bit more difficult to control and set up properly and intuitively. Also lags quite a bit if used on scenes/objects with many elements such as complicated meshes.
  - **Translate (unit)** – Translates the elements this material is tied to based on the selected values and units.
  - **Scale** – Scales the elements
  - **Skew (deg)** – Skews the elements by specified degrees.
  - **Rotation Axis** – Defines the axis around which to rotate. You can use (1, 0, 0) for rotation along X-axis; (0, 1, 0) for Y-axis or (0, 0, 1) for Z-axis (which is the same as a 2D rotation).
  - **Rotation Angle (deg)** – Rotates elements around the specified axis by the specified amount of degrees.
- **Transform Origin** – By **default** transformations such as rotation are applied around the **top left corner** of the canvas. This setting allows you to **choose an object** in a scene; the **2D location of this object in the view will then become the transform origin** for the defined transformations. Difficult to use, but might help someone get better results when experimenting with transformations. For example by creating a special small object to use as a sort of pivot in the scene for all the transformations.

3) How to install the special version

In order to run the special version (**scene_to_svg_extra_cutting.py**), you need to install Python library “Shapely” into your version of Blender’s Python. Once you get it installed, the process is the same as for the basic version.

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, sys
   ```
   (Imports subprocess and sys modules)

   ```
   PY_EXE = sys.executable
   ```
   (Gets the Blender Python executable path)

   ```
   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 (“3.2” changes based on the installed version of Blender):
   <pathToBlender>\Blender 3.2\3.2\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)

After this, Shapely should be correctly installed into Blender’s version of Python. Once that is complete, you can follow the same steps as when installing the basic version.