



# **DrawSVG user's guide**

**Free online drawing applications  
for designers and developers .**

**Joseph LIARD**



# Draw SVG

Joseph LIARD

release 12-3

Publication date 2025 April

Copyright © 2013 drawsvg.org

## Abstract

This document introduced the use of "Draw SVG", free online drawing applications for designers and developers **drawsvg.org**

In addition to this document, the showcases tool can gives more details and explanations with features interactive demonstrations.

This application uses the SVG drawing format, standard defined by the **W3C** .

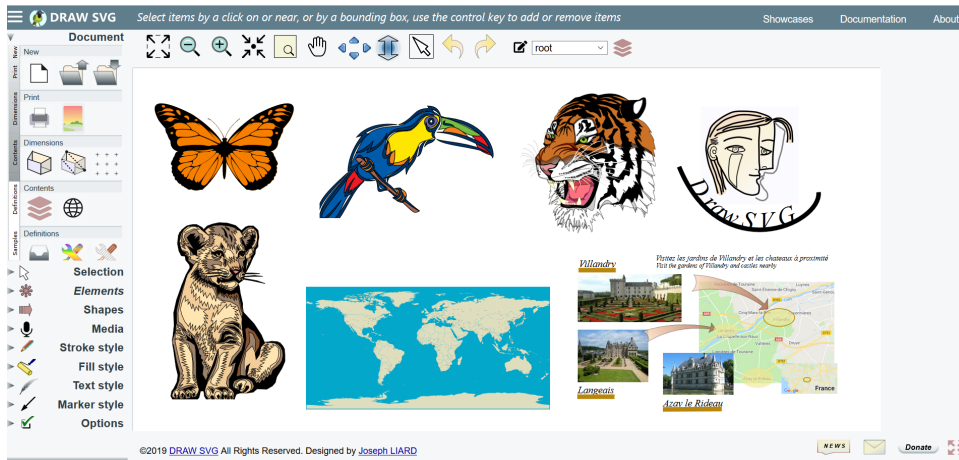
Draw freely, save and print your documents.

Know the full potential of vector graphics with SVG.

You can perform a wide variety of designs, art as the icon of DRAW-SVG, mapping, technical, presentation slides, etc. ...



Some examples of well-known drawings and other from DRAW-SVG:



This documentation is also available **online** at the following address: [draw-svg.html doc.](#)



---

1. User profile .....	1
2. Quick Start .....	5
2.1. Introducing GUI DRAW-SVG .....	5
2.1.1. Main menu .....	5
2.1.2. Actions bar .....	9
2.1.3. Selection floating menu .....	10
2.1.4. Dialog panels .....	11
2.2. Create your document .....	11
2.3. Drawing a line .....	13
2.4. Editing a line .....	13
2.5. Drawing a rectangle .....	14
2.6. Editing a rectangle .....	15
2.7. Drawing a text .....	16
2.8. Editing a text .....	16
2.9. Using patterns .....	17
2.10. Using gradients .....	19
2.11. Using markers .....	21
3. Manage documents .....	23
3.1. Document menu .....	23
3.2. Create new document .....	24
3.3. Open a document .....	25
3.3.1. Loading a SVG file .....	26
3.3.2. Loading an image file .....	26
3.3.3. Loading an image from the WEB .....	27
3.3.4. Loading an URL .....	28
3.3.5. Loading a svg from freesvg gallery .....	29
3.3.6. Load a SVG code .....	30
3.4. Save the document .....	31
3.5. Print the document .....	32
3.6. Export the document as PNG .....	33
3.7. Define document dimensions .....	34
3.8. Resize document .....	34
3.9. Layers management .....	35
3.10. Grid definition .....	36
3.11. Import definitions .....	36
3.12. Gradient definitions .....	37
3.13. Pattern definitions .....	38
3.14. Filter definitions .....	38
3.15. Marker definitions .....	39
4. Editing selection .....	40
4.1. Selection menu .....	40
4.2. Select elements .....	41
4.3. Apply style .....	41
4.4. Select background rectangle .....	42
4.5. Select drawn elements .....	42
4.6. Order front .....	42
4.7. Order back .....	43
4.8. Align .....	43



---

4.9. Space equally .....	44
4.10. Marker definition .....	44
4.11. Pattern definition .....	45
4.12. Editors .....	46
4.13. Editor panels .....	50
4.13.1. Map editor dialog panel .....	50
4.13.2. Filter points panel .....	51
4.13.3. Smooth vertices panel .....	52
4.13.4. TextPath offset panel .....	52
4.13.5. Transform gradient panel .....	53
4.13.6. Transform pattern panel .....	54
4.13.7. Image file load panel .....	54
4.13.8. Image search WEB panel .....	55
4.13.9. Image url load panel .....	55
4.13.10. WEB Image load panel .....	56
4.13.11. Properties dialog panel .....	56
5. Drawing elements .....	60
5.1. Drawing menu .....	60
5.2. Drawing repetitive mode .....	61
5.3. Drawing tasks .....	61
6. Drawing shapes .....	67
7. Style properties .....	69
7.1. Stroke style properties .....	69
7.2. Fill style properties .....	71
7.3. Text style properties .....	72
7.4. Marker style properties .....	73
7.5. Style panels .....	75
7.5.1. Color chooser panel .....	75
7.5.2. Gradient dialog panel .....	76
7.5.3. Marker dialog panel .....	78
7.5.4. Pattern dialog panel .....	79
7.5.5. Filter dialog panel .....	81
8. Apply constraints .....	83
8.1. Constraints introduction .....	83
8.2. Connect points constraint .....	83
8.3. Bounding box constraint .....	84
8.4. Clipping constraint .....	85
9. Animations .....	87
9.1. Introduction .....	87
9.2. The animations menu .....	87
9.3. The animations schedule .....	88
9.4. Animation timing properties .....	88
9.5. Animation interpolation properties .....	92
9.6. Animate attribute .....	94
9.7. Animate path .....	96
9.8. Animate transform .....	97
9.9. Animate motion .....	101
9.10. Set attribute .....	105



---

9.11. Animate gradients .....	108
9.12. Animate patterns .....	109
10. Options .....	111
11. Tools .....	112
11.1. Photo to drawing .....	112
11.2. Base64 Image Encoder .....	116
11.3. SVG to PNG Converter .....	116
11.4. Optimize SVG .....	117
12. Integration .....	119
12.1. Integration by URL .....	119
12.2. Integration by API .....	119
12.2.1. Introduction .....	119
12.2.2. Function setDocumentMenu .....	121
12.2.3. Function loadStringSVG .....	122
12.2.4. Function loadUrlSVG .....	124
12.2.5. Callback saveService .....	126
12.2.6. Function getSVG .....	127
12.2.7. Function getSVGObject .....	127
12.2.8. Function addLayer .....	128
12.2.9. Function setInputLayerId .....	129
12.2.10. Function setCustomShapeCatalog .....	130
13. Customization .....	131
13.1. Define a custom shape catalog .....	131
13.2. Customize Edrawsvg color theme .....	132
13.3. Customize Edrawsvg shape catalogs .....	132
13.4. Customize Edrawsvg parameters .....	133
13.5. Customize DRAW SVG UI .....	134
13.5.1. Introduction .....	134
13.5.2. The MVP pattern .....	134
13.5.3. The SVG viewing area .....	135
13.5.4. The application .....	135
13.5.5. The DRAWSVG engine .....	136
13.5.6. The drwapp sample application .....	136
13.5.7. Website integration .....	137



---

# Chapter 1. User profile

The functions of DrawSVG editor are classified into two user profiles:

- A **basic** profile comprising all the common functions for editing an SVG document
- An **expert** profile with additional advanced features intended for users specializing in SVG

## Differences between basic and expert profile

The basic profile is **free** to use with no identification required.

Expert profile needs a drawsvg account which can be created with a:

- **Patreon** account (see drawsvg patreon account [2])
- **Google** account (see drawsvg google account [3])
- **Email** account (see drawsvg email account [4])

The expert profile is :

- **Free** for students and teachers (see drawsvg education account [4])
- Granted by **Drawsvg Patreon** subscription with supports (see drawsvg patreon account [2])
- Can be evaluated for a period of 5 days of use by logging in with your drawsvg account

Each profile has tasks organized into three categories:

- Tasks to manage documents (opening, saving, etc.)
- Tasks to draw elements
- Tasks to modify the selected elements


To see the difference between basic and expert profile, visit the links below with the showcases associated with each task:

**Table 1.1. User profile tasks**

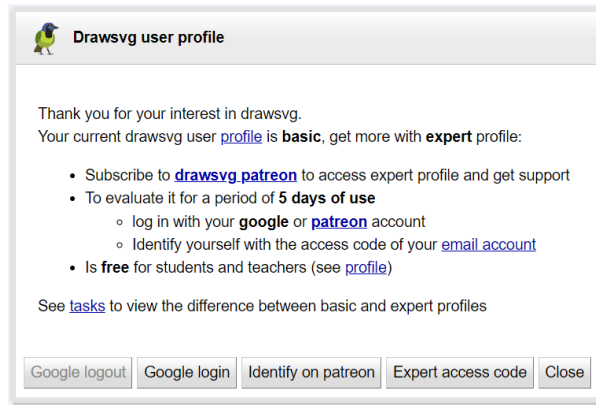
Categories	Basic profile	Expert profile	All
Document	basic document tasks	expert document tasks	all document tasks
Drawing	basic element drawing tasks	expert element drawing tasks	all element drawing tasks
Editing	basic element editing tasks	expert element editing tasks	all element editing tasks

## The user profile dialog

User profile dialog box appears at startup.

To view it during a session click on the user profile icon  in the menu bar.





**Figure 1.1. User profile**

Then identify you with your **Patreon** [2] or your **Google** [3] account, otherwise enter the access code provided with your **email** [4] account.

An access code is required for **node-red** users (see node-red users [2]).

The expert profile remains activated the next time you access drawsvg with the same browser (with cookies enabled).

## Using the expert profile for node-red users

Users of **node-red** with its **floor plan** use the built-in drawsvg editor through its integration API [119].

Unfortunately, due to node-red's architecture based on Node JS, the Google login service and Patreon login service cannot work on it.

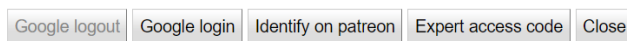
To overcome this problem, Node-Red users can use an **access code** to obtain the expert profile.

An access code is automatically provided with email accounts [4].

For other users, they should get it from the user profile dialog [1]:

- Authenticate with your Google [3] or Patreon [2] account.
- Then request an access code as shown by clicking on the link:

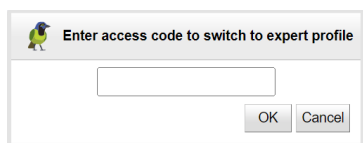
[Node-red users](#) needs an access code to obtain the expert profile.  
Just click [here](#) to request an access code.



- The access code will be transmitted immediately by drawsvg server and displayed in a popup. The access code provided is attached to the account and must remain confidential.

If you have forgotten your access code, you can request it again from the dialog box.

Once obtained, click the "**Expert access code**" button and enter it to get expert profile of your account :



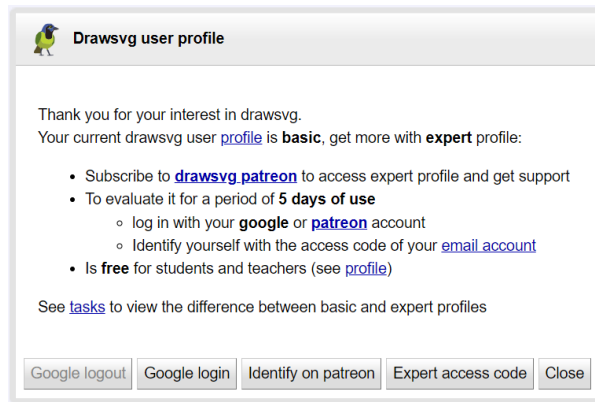
## Drawsvg patreon account

Drawsvg is registered on the **Patreon** platform as a creator, see **drawsvg patreon page**.



The Patreon platform offers WEB services to connect creators with users in their community. To create a drawsvg patreon account, you need first a **Patreon** account.

Once your patreon account is created, click the "**Identify on patreon**" button to connect your drawsvg account (creation is done at the first request) with it:



The connection is established with the standard **oauth2** protocol to ensure your data is securely protected. Next, you need to authorize drawsvg to connect to your Patreon account. Patreon does not provide your personal data, drawsvg will only obtain your user ID and drawsvg subscription status.

Drawsvg expert profile is granted by **Drawsvg Patreon** subscription with supports.

The subscription may include a trial period. Outside of subscription, the expert profile can be evaluated over a period of 5 days of use.

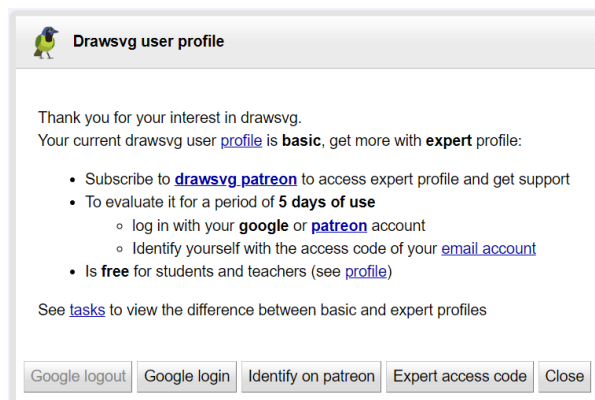
Using node-red with your drawsvg patreon account requires an **access code** to obtain the expert profile (see Node-red users [2]).

## Drawsvg Google account

Drawsvg is hosted by the Google platform, so it can identify its users using the Google login service.

Your personal data is securely protected, drawsvg only receives your Google email address.

Click the "**Google login**" button to connect your drawsvg account (creation is done at the first login) with your Google account:



With your Drawsvg Google account, the expert profile can be evaluated over a period of 5 days of use, beyond that you must subscribe to a Patreon subscription.

Drawsvg Google user account can have education status (see education account [4]).





Using node-red with your drawsvg google account requires an **access code** to obtain the expert profile (see Node-red users [2]).

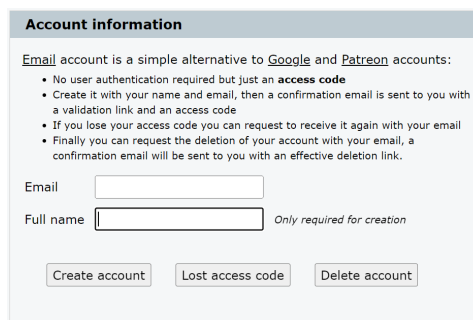
## Drawsvg email account

You can create an account with your email address when using a Google or Patreon account is not appropriate. This is the case for an address of a company or institution when the email domain is not provided by Google.

Email accounts are simple to create:

1. They just require a name and an email address
2. They are activated using a link provided in the welcome email
3. No user authentication required but just an **access code** sent in the welcome email

To create one, use the email account management tool:



The screenshot shows a web form titled "Account information". It contains the following text and elements:

- Section header: **Account information**
- Text: "Email account is a simple alternative to [Google](#) and [Patreon](#) accounts:"
- Bullet points:
  - No user authentication required but just an **access code**
  - Create it with your name and email, then a confirmation email is sent to you with a validation link and an access code
  - If you lose your access code you can request to receive it again with your email
  - Finally you can request the deletion of your account with your email, a confirmation email will be sent to you with an effective deletion link.
- Form fields:
  - Email:
  - Full name:  Only required for creation
- Buttons: "Create account", "Lost access code", "Delete account"

With your Drawsvg email account, the expert profile can be evaluated over a period of 5 days of use, beyond that you must subscribe to a Patreon subscription.

Drawsvg email user account can have education status (see education account [4]).

For node-red users, the access code provided allows use of the expert profile.

## Drawsvg education account

The drawsvg expert profile is **free** for education account.

Students and teachers who have a school email address can create a drawsvg education account.

It is advisable to first register the **email domain** of the institution by sending an email request to drawsvg (see contact). The request should be sent by a student or professor using **their institution's email address** and include the institution's **website URL**. Drawsvg will confirm the email domain registration.

Then, users of the educational institution :

1. Can create a Google account [3] only if the institution's email domain is provided by Google.
2. Or create an email account [4] with their institution email address.
3. Then the account is automatically marked as educational by identifying the institution's email domain in the user's email address.

Note that when registering an educational domain, all existing accounts associated with the institution are marked as educational. The process can therefore be reversed.

Using node-red with your drawsvg education account requires an **access code** to obtain the expert profile (see Node-red users [2]).



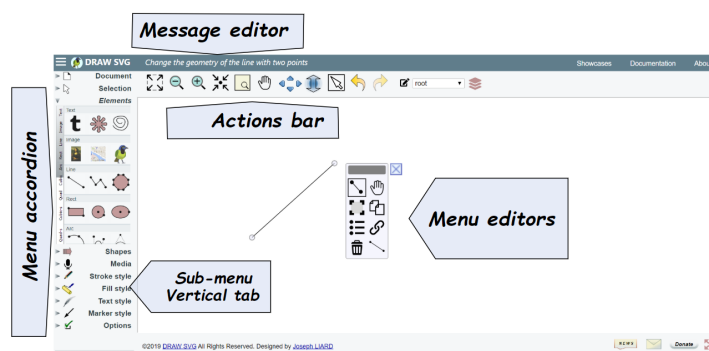
---

# Chapter 2. Quick Start

## 2.1. Introducing GUI DRAW-SVG

The GUI DRAW-SVG consists of :

- an accordion-style main menu,
- an action bar,
- A floating menu graphic editing the selection,
- dialog panels in modal mode (ex Google map dialog panel [50] ),
- svg editor panels which interacts with the drawing (ex filter points panel [51] )



**Figure 2.1. Introduction GUI**

Then in addition, buttons and links:

- documentation, view this documentation,
- about, gives informations about DRAW SVG such as release notes,
- news, new features of the current release,
- letter, send a message to us,
- links to additional sites and partners

### 2.1.1. Main menu

The main menu is composed of sub menus (Document, Selection, Elements, Shapes, Media, Stroke style, Fill Style, Text Style, Marker style, Animations, options).

Each sub-menu has functions in the form of icons.



**Table 2.1. Main menu map**


**Table 2.2. Sub-menus features**

<p><b>Document</b> sub-menus:</p> <ul style="list-style-type: none"> <li>• <b>New</b> : create, open, save document.</li> <li>• <b>Print</b> : print, png export document.</li> </ul>	<p><b>Selection</b> sub-menus:</p>
---	------------------------------------



<ul style="list-style-type: none"> <li>• <b>Dimensions</b> ; edit, resize dimensions.</li> <li>• <b>Contents</b> : browse layers, define multi-language support.</li> <li>• <b>Definitions</b> : import from another document, define gradients, patterns filters, markers, css styles.</li> <li>• <b>Samples</b> : loading samples</li> </ul>	<ul style="list-style-type: none"> <li>• <b>Select</b> : select elements, apply style of selected element to another, select background rectangle.</li> <li>• <b>Draw</b> : select elements from drawn list.</li> <li>• <b>Order</b> ; move selected elements to top of its layer, to bottom of its layer, forward one, back one.</li> <li>• <b>Align</b> : align selected elements to top border, middle, bottom, left, center, right.</li> <li>• <b>Distribute equally</b> : distribute space between selected element horizontal, vertical.</li> <li>• <b>Definitions</b> : use selected elements to define markers and patterns</li> </ul>
<p><b>Elements</b> sub-menus, draw:</p> <ul style="list-style-type: none"> <li>• <b>Text</b> : draw text, path, free path.</li> <li>• <b>Draw</b> : select elements from drawn list.</li> <li>• <b>Image</b> ; insert an image, insert a map, insert an svg document.</li> <li>• <b>Line</b> : draw line, polyline, polygon.</li> <li>• <b>Rect</b> : draw rectangle, circle, ellipse.</li> <li>• <b>Arcs</b> : draw circle arc, cubic arc, quadratic arc.</li> <li>• <b>Cubic</b> : draw cubic curve, closed curve.</li> <li>• <b>Quadratic</b> : draw quadratic curve, closed curve.</li> <li>• <b>Cubic smooth</b> : draw cubic smooth curve, closed curve.</li> <li>• <b>Quadratic smooth</b> : draw quadratic smooth curve, closed curve.</li> </ul>	<p><b>Shapes</b> sub-menus, insert:</p> <ul style="list-style-type: none"> <li>• <b>Draw object from</b> : draw shapes by corner, center.</li> <li>• <b>Catalog</b> : select and insert shape from drawsvg catalog, emojis icones, Font Awesome icones.</li> <li>• <b>Arrows</b> ; select and insert shape from arrows list.</li> <li>• <b>Flowcharts</b> ; select and insert shape from flowcharts list.</li> <li>• <b>Symbols</b> ; select and insert shape from symbols list.</li> </ul>
<p><b>Media</b> sub-menus, insert:</p> <ul style="list-style-type: none"> <li>• <b>Controls</b> : insert HTML audio, video control, HTML rich text, multi-language text, YouTube control</li> </ul>	<p><b>Stroke style</b> sub-menus, set selected element stroke style</p> <ul style="list-style-type: none"> <li>• <b>Transparent</b> : set stroke style to transparent.</li> <li>• <b>Color</b> : set stroke color.</li> <li>• <b>Width</b> ; set stroke width.</li> <li>• <b>Dash</b> ; set stroke dash.</li> <li>• <b>Opacity</b> ; set stroke opacity.</li> <li>• <b>Gradient</b> ; set stroke gradient.</li> <li>• <b>Cap</b> ; set stroke line cap.</li> <li>• <b>Join</b> ; set stroke line join.</li> <li>• <b>Pattern</b> ; set stroke pattern.</li> </ul>



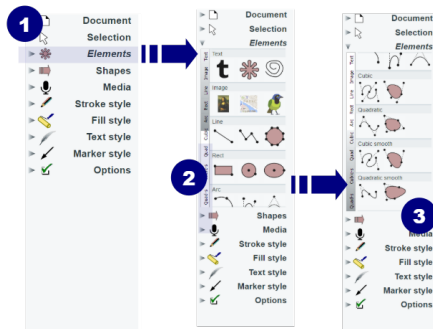
<p><b>Fill style</b> sub-menus, set selected element fill style</p> <ul style="list-style-type: none"> <li>• <b>Transparent</b> : set fill style to transparent.</li> <li>• <b>Color</b> : set fill color.</li> <li>• <b>Opacity</b> ; set fill opacity.</li> <li>• <b>Gradient</b> ; set fill gradient.</li> <li>• <b>Pattern</b> ; set fill pattern.</li> <li>• <b>Rule</b> ; set fill rule to evenodd or nonzero.</li> </ul>	<ul style="list-style-type: none"> <li>• <b>Effect</b> ; set vector effect as scaling or not.</li> </ul> <p><b>Text style</b> sub-menus, set selected text style</p> <ul style="list-style-type: none"> <li>• <b>Family</b> : set font family.</li> <li>• <b>Size</b> : set font size.</li> <li>• <b>Style</b> ; set font style to normal, italic, oblique.</li> <li>• <b>Weight</b> ; set font weight.</li> <li>• <b>Anchor</b> ; set text anchor to start, middle, end.</li> <li>• <b>Decoration</b> ; set text decoration to underline, overline, line through, blink.</li> </ul>
<p><b>Marker style</b> sub-menus, set selected line marker style</p> <ul style="list-style-type: none"> <li>• <b>Samples</b> : apply one of default marker style samples.</li> <li>• <b>Start</b> : set start marker from the list.</li> <li>• <b>Mid</b> ; set middle marker from the list.</li> <li>• <b>End</b> ; set end marker from the list.</li> </ul>	<p><b>Animations</b> sub-menus,</p> <ul style="list-style-type: none"> <li>• <b>Pause</b> : Pause all animations.</li> <li>• <b>Play</b> : Unpause all animations.</li> <li>• <b>Scheduler</b> : View animations planning.</li> <li>• <b>Attribute</b> : Animate attribute, add an animate element on the selected object, edit all animate elements of the document.</li> <li>• <b>Transform</b> : Coming soon.</li> <li>• <b>Motion</b> : Move a referenced element along a motion path.</li> <li>• <b>Set</b> : Provides a simple means of just setting the value of an attribute for a specified duration. It is useful to fix the state of an element during animations.</li> <li>• <b>Gradients</b> : Animate a gradient properties and its stop color elements. Transform the gradient (scale, rotate, translate).</li> <li>• <b>Patterns</b> : Animate a pattern properties and its content elements. Transform the pattern (scale, rotate, translate).</li> </ul>
<p><b>Options</b> sub-menus,</p> <ul style="list-style-type: none"> <li>• <b>Animate menus</b> :</li> </ul>	



Animates the menu when opening a sub-menu with SMIL animations.

To open a sub-menu :

1. Click on the title of the sub-menu
2. A sub-menu is divided into vertical tabs. To access the content of a tab, click its title in the vertical bar.
3. Click on the sub-menu item.



**Figure 2.2. Open a menu**



**Note**

Scrolling in the main menu is possible with the mouse wheel :

- on the left side of the main menu, closing the current sub-menu and opening the next or previous one,
- inside a sub-menu, scrolling down or up inside it's area












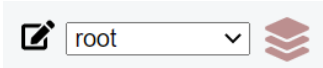
**2.1.2. Actions bar**

The Action Bar provides functions on the view (Original view, zoom out, zoom in, zoom on point on Zoom rectangle Scroll down left top right), selecting drawn elements, undo and redo functions.

**Table 2.3. Actions bar**

icon	function
	Return to the original view
	Zoom out
	Zoom in



icon	function
	Zoom on the clicked point
	Zoom on a rectangle defined by its top left and bottom right corners
	Move view
	Scroll left
	Scroll right
	Scroll up
	Scroll down
	Enable/Disable zoom mouse wheel
	Select items by a click on or near, or by a bounding box, use the control key to add or remove item
	Undo
	Redo
	Layer chooser 

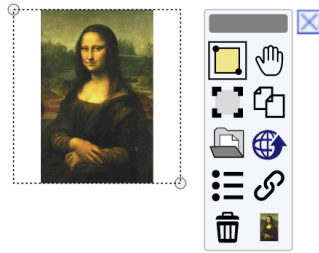
### 2.1.3. Selection floating menu

Each selected shape has a floating menu that provides editing tools for selection. When an editing tool is activated, it displays a help message at the top right of the document.

The selection menu can be moved with its bar, stopped with its quit button.

When the mouse is over the icon of a function, its title appears on the right side of the icon.

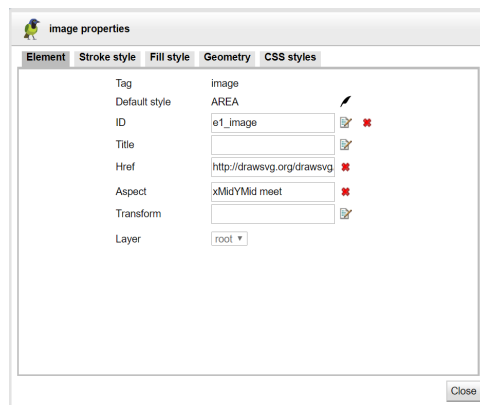
Next, the floating menu of an image element.



**Figure 2.3. Image floating menu**

### 2.1.4. Dialog panels

Some functions use modal dialog panels, such as " edit properties" [56] from the floating menu.



**Figure 2.4. Image properties panel**



#### Note

The properties panel shows all attributes of the selected element.

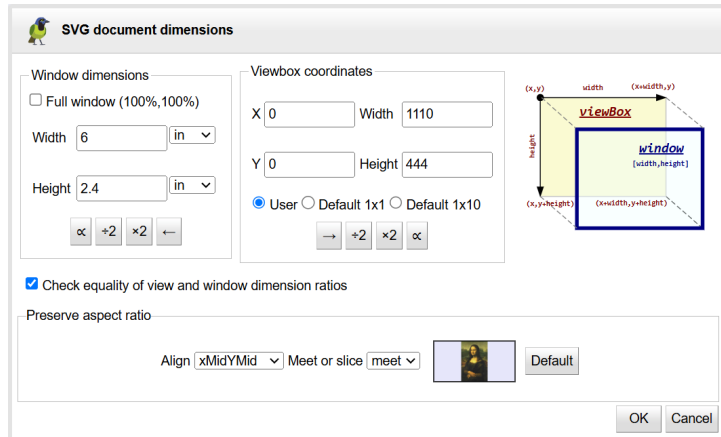
## 2.2. Create your document

Draw SVG opens with a blank document the size by default. You can draw in this document or create a new one with the desired dimensions.

Open the "Document" menu by clicking its title, then click on the item "New".

The dialog panel "SVG document dimensions" opens and you can specify the document dimensions.

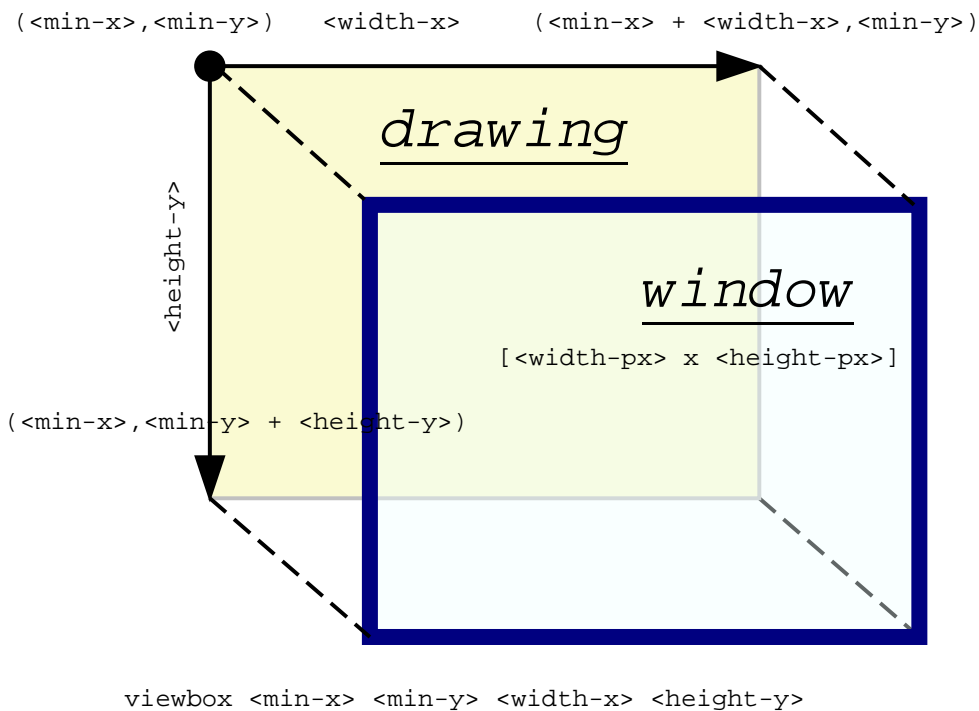




**Figure 2.5. Create document**

The panel allows you to define the dimensions of the document display and its coordinate system.

The "Full window" option allows you to define a document whose size occupies the entire display of the document window. Click "ok" to create the document.



**Figure 2.6. document dimensions**

This figure shows the two dimensions of the document

- The drawing space and its coordinate system defined by the "viewBox" attribute of the SVG document
- The physical dimensions with its units, width and height of the window or the paper size.



- The projection method of the drawing space defined by the attribute "preserveAspectRatio". Then you can draw basic graphical items "Draw elements" or predefined shapes "Draw shapes".

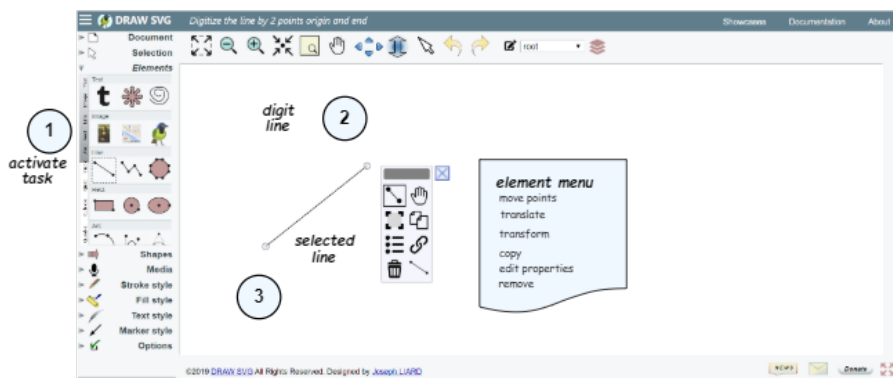
## 2.3. Drawing a line

The "Draw elements" menu allows you to draw all known elementary forms of SVG.

We start by drawing a line.

To draw a line follow these steps:

1. Open the "Elements" menu and its "line" sub-menu. Click on the icon "Line" to activate the function. The icon is framed by a dotted line to show that it is activated. The function displays the message "Digitize the line by 2 points origin and end."
2. Draw the line with both ends, you can point each item separately or continuously as with a pencil.
3. Drawn line is selected and the floating menu shows available editors to apply on it, you can re-draw a line from the floating menu.



**Figure 2.7. Drawing a line**

See also draw line showcase for an interactive demonstration of how to draw a line.



### Note

The method of drawing a line is applicable to all basic forms with two points (rectangle, circle, ellipse, image, map).

## 2.4. Editing a line

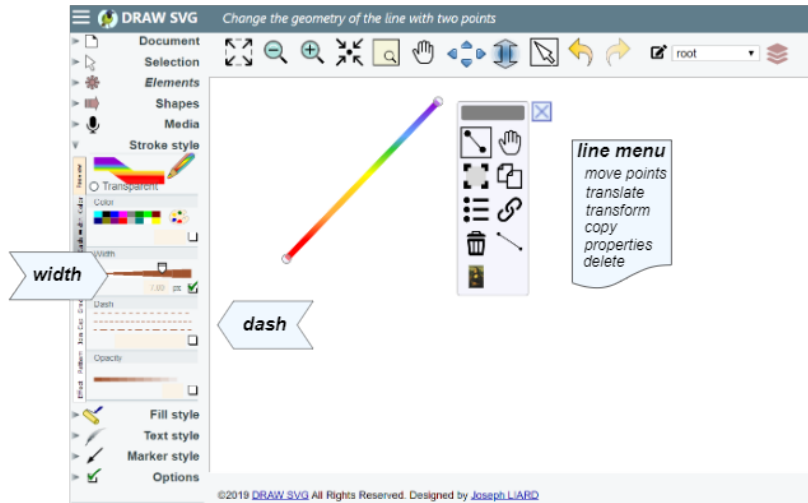
The style of the selected line can be personalized with menus (stroke, fill, text, marker). These menus adapt to the element and display mode wysiwig the current values of its style properties.

The menus that are not applicable (eg to fill a line) are disabled.

Each style property has a "widget" to enter its value and illustrate and a check box to activate or cancel.

The style of a line can be personalized with the properties of stroke style menus [69] and marker style [73].





**Figure 2.8. Styling a line**

You can change the style properties of the line and activate an editing tool of the floating menu :

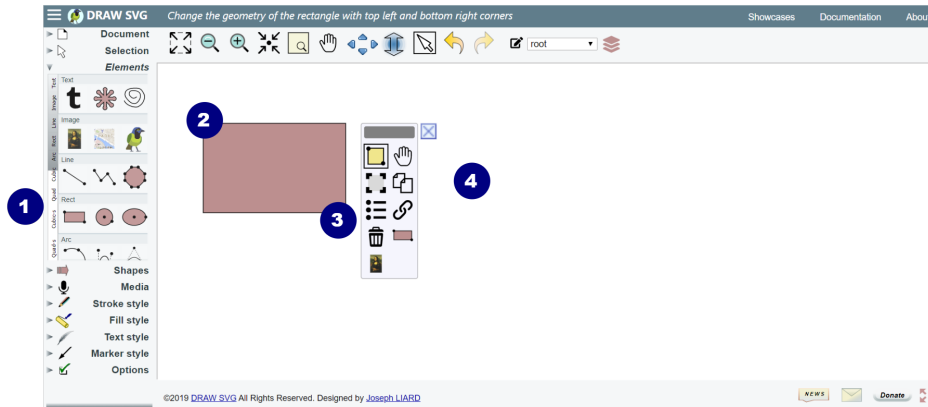
- move points
- translate the line
- transform the line
- copy the line
- edit line properties
- delete the line

## 2.5. Drawing a rectangle

To draw a rectangle follow these steps:

1. Open the "Elements" menu and its "rect" sub-menu. Click on the icon "Rectangle" to activate the function. The icon is framed by a dotted line to show that it is activated. The function displays the message " *Digitize the rectangle by top left and bottom right corners.* "
2. Click the first point
3. Click the second point, you can point each point separately or continuously as with a pencil.
4. Drawn rect is selected and the floating menu shows available editors to apply on it.





**Figure 2.9. Drawing a rectangle**

See also draw rectangle showcase for an interactive demonstration of how to draw a rectangle.

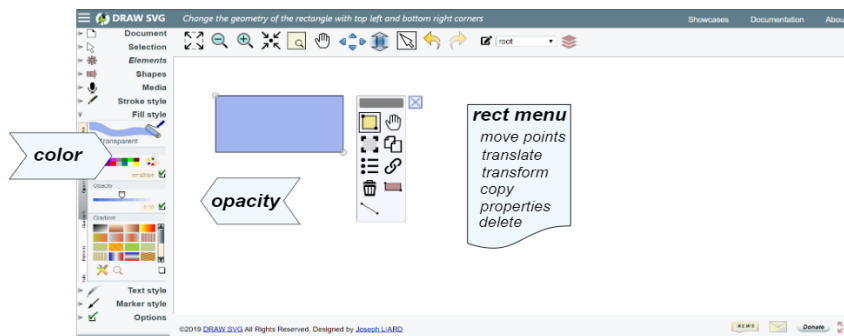
## 2.6. Editing a rectangle

The style of the selected rectangle can be personalized with menus stroke and fill. These menus adapt to the element and display mode wysiwig the current values of its style properties.

The menus that are not applicable (eg text style and marker style) are disabled.

Each style property has a "widget" to enter its value and illustrate and a check box to activate or cancel.

The style of a rectangle can be personalized with the properties of stroke style menus [69] and fill style [71] .



**Figure 2.10. Styling a rectangle**

You can change the style properties of the rectangle and activate an editing tool of the floating menu :

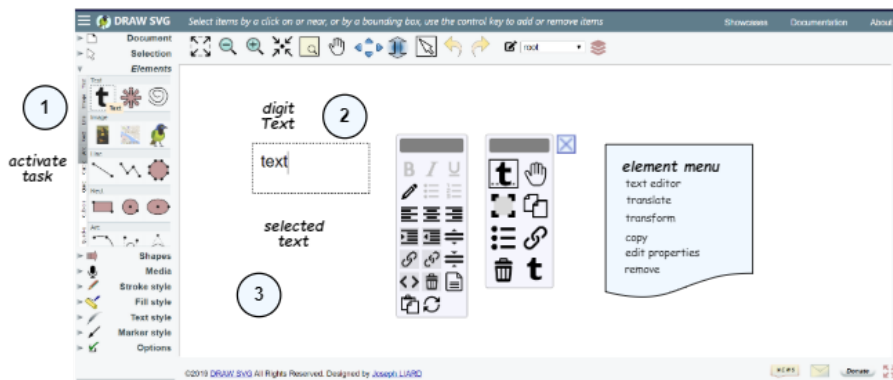
- move points
- translate the rectangle
- transform the rectangle
- copy the rectangle
- edit rectangle properties
- delete the rectangle



## 2.7. Drawing a text

To draw a text follow these steps:

1. Open the "Draw elements" menu and its "text" sub-menu. Click on the icon "Text" to activate the function. The icon is framed by a dotted line to show that it is activated. The function displays the message " *Click the anchor text point* ".
2. Click the text point
3. The item is selected and the text editor is enabled. You can type the text. The input cursor can be moved with the arrow keys or by clicking on the target character.



**Figure 2.11. Drawing a text**

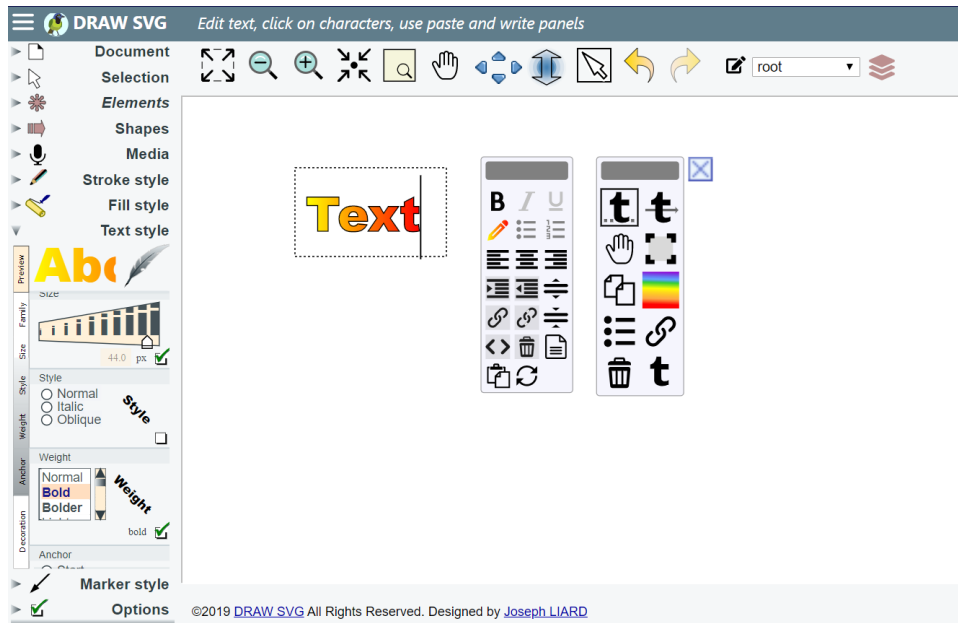
See also draw simple text , multiline text showcases for an interactive demonstration of how to draw and edit a text.

## 2.8. Editing a text

The style of the selected text can be personalized with the properties of 3 menus ( stroke style [69] , fill style [71] , text style [72] ).

Each style property has a "widget" to enter its value and illustrate and a check box to activate or cancel.





**Figure 2.12. Styling a text**

You can change the style properties of the text and activate an editing tool of the floating menu :

- edit text string and move anchor point
- translate the text
- transform the text
- copy the text
- edit text properties
- delete the text
- re-draw a text

## 2.9. Using patterns

The ' pattern ' element is a member of a 'defs' section and defines the graphic that is to be used to fill a shape by replicating an object.

It performs two key functions: hatching and wallpaper.

DrawSVG has a default palette of patterns.

See creating , using patterns showcases for interactive demonstrations.

To use pattern :


- Apply a pattern of the default palette to the selected shape
- Customize the pattern for the selected shape with a transformation
- Edit patterns definitions
- Draw and define a new one


How to apply a pattern to a shape

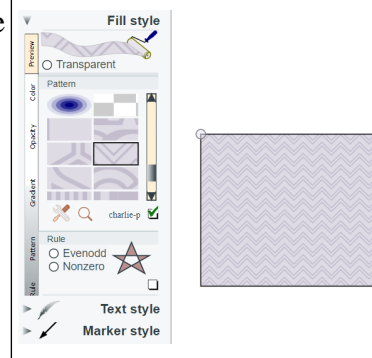


To **apply** a pattern, select the shape and choose the pattern from the fill style menu.

To choose the pattern, two other tools can be used :

The pattern definition [79]  tool


Or the pattern chooser [81]  tool



**Note**

A pattern can be applied also to a line with the stroke style menu.

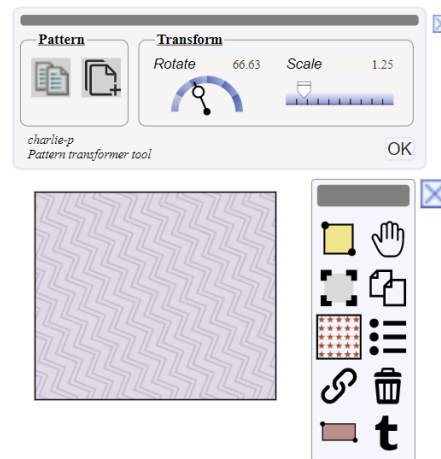
How to customize the pattern of a shape with a transformation

To **customize** the pattern of a shape, use the transform pattern [54]  editor from the floating menu :


- To prevent modification of the shared pattern definition, make a copy or an extension of the pattern,
- Then define the transformation with the rotation angle and the scale factor

Apply the transformation or close the panel without modification.

Customization by extension has the advantage of minimizing the volume and continue to benefit from changes made to the basic pattern.

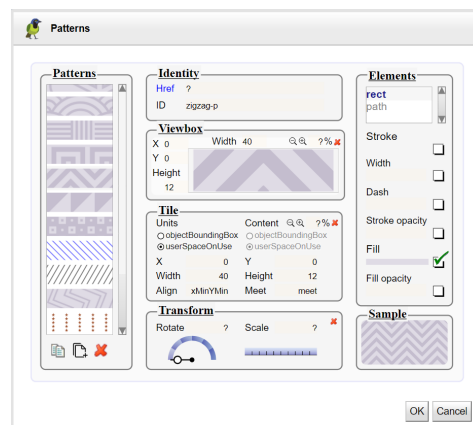


How to edit the pattern of a shape

To **edit** the pattern definition of a shape, select the shape and use the pattern definition [79] tool  from the fill style menu.


The tool allows to edit/copy/extends/delete a pattern definition,

When opened, the panel is showing the pattern of the selected shape. You can change its properties and apply them, or select another pattern and apply it to the shape.



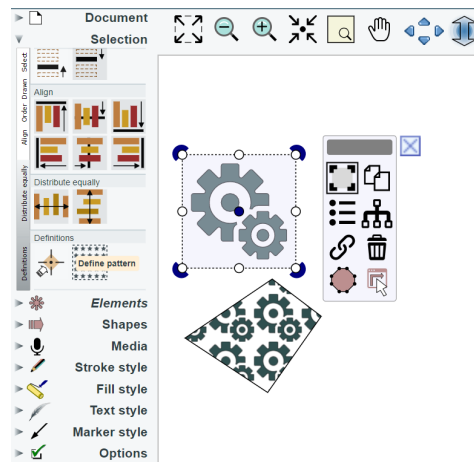
How to define a new pattern



To **define** a new pattern, draw elements of it's graphic, select them and use the pattern definition [79] tool  from the selection menu.

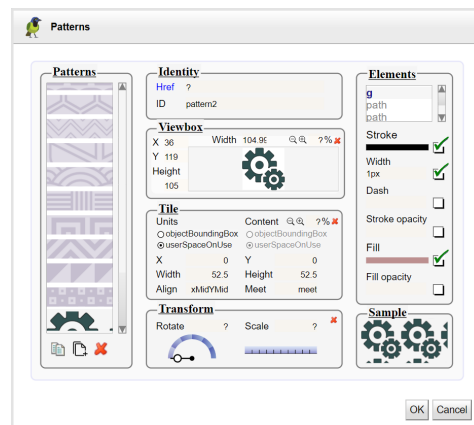
The pattern is defined with the selection :

- The elements of the pattern are defined by a copy of the selection.
- The viewBox is set with the bounding box of the selection
- The units of the tile is set as userSpaceOnUse and it's size equals to viewBox size



Then you can change each properties :

- **Identity** : the pattern's id.
- **ViewBox** : Use text fields (x,y,width, height), icons (zoom/in zoom/out) and text field ratio for extending or reducing the area. To add a margin of 20% type 120<enter> in the ratio field.
- **Tile** : Define the dimensions and units of the filling tile. Use text fields (x,y,width, height), icons (zoom/in zoom/out) and text field ratio for extending or reducing the size of the tile.
- **Transform** : Rotating and scaling the tile.
- **Elements** : List of all the graphic elements of the pattern with their style editable properties (stroke, stroke-width, stroke-opacity, stroke-dash, fill, fill-opacity).



## 2.10. Using gradients

A gradient is a smooth transition from one color to another. In addition, several color transitions can be applied to the same element.

The gradient element is a member of a 'defs' section.

There are two types of gradients : linear and radial .

DrawSVG has a default palette of gradients.

See creating , using gradients showcases for interactive demonstrations.

To use gradient :

- Apply a gradient of the default palette to the selected shape
- Customize the gradient for the selected shape with a transformation
- Edit gradient definitions


How to apply a gradient to a shape




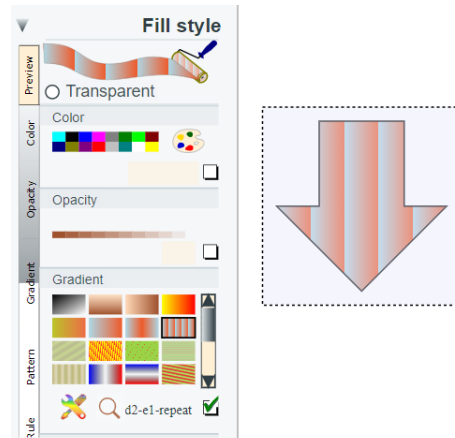


To **apply** a gradient, select the shape and choose the gradient from the fill style menu.

To choose the gradient, two other tools can be used :

The gradient definition [76]  tool


Or the gradient chooser [77]  tool



**Note**

A gradient can be applied also to a line with the stroke style menu.

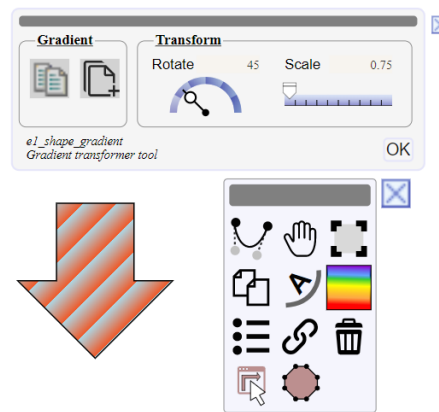
How to customize the gradient of a shape with a transformation

To **customize** the gradient of a shape, use the transform gradient [53]  editor from the floating menu :


- To prevent modification of the shared gradient definition, make a copy or an extension of the gradient,
- Then define the transformation with the rotation angle and the scale factor

Apply the transformation or close the panel without modification.

Customization by extension has the advantage of minimizing the volume and continue to benefit from changes made to the basic gradient.

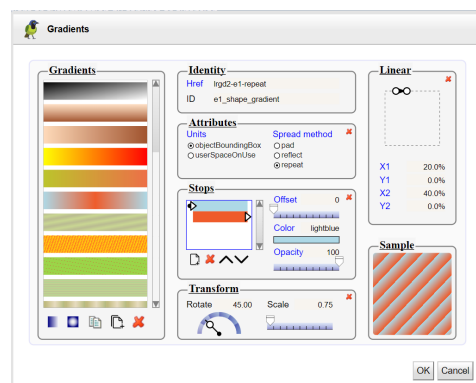


How to edit the gradient of a shape

To **edit** the gradient definition of a shape, select the shape and use the gradient definition [76] tool  from the fill style menu.

The tool allows to edit/copy/extends/delete a gradient definition,

When opened, the panel is showing the gradient of the selected shape. You can change



it's properties and apply then, or select another gradient and apply it to the shape.

## 2.11. Using markers

The ' marker ' element is a member of a 'defs' section and defines the graphics that is to be used for drawing arrowheads or polymarkers on a given 'path', 'line', 'polyline' or 'polygon' element.

A linear form can have three different markers :

- start, on the first point
- end, on the last point
- mid, on each intermediate points

DrawSVG has a default palette of markers.

See creating , using markers showcases for interactive demonstrations.


To use markers :


- Apply a marker of the default palette to the selected linear form and the style property (end, mid, start)
- Edit markers definitions
- Draw and define a new one

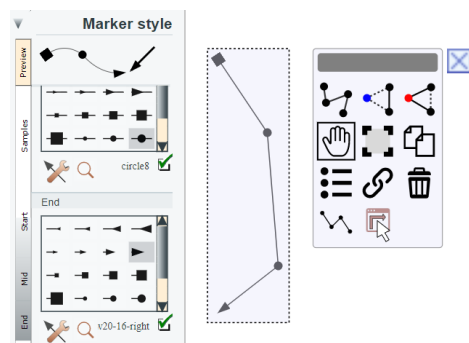
### How to apply a marker to a linear form

To **apply** a marker, select the shape and choose the marker from the marker style menu and the property (end,mid, start).


To choose the marker, two other tools can be used :

The marker definition [78]  tool

Or the marker chooser [79]  tool

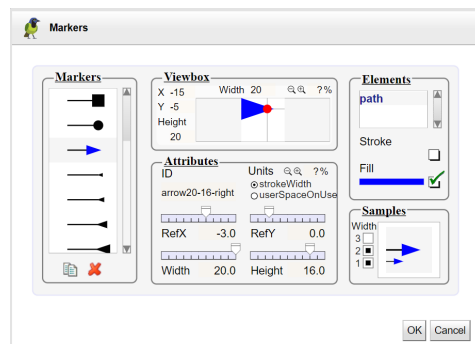


### How to edit and customize the marker of a shape

To **edit** the marker definition of a shape, select the shape and use the marker definition [78] tool  from the marker style menu and the property (end, mid, start).

The tool allows to edit/copy/delete a marker definition,

When opened, the panel is showing the marker of the selected shape. You can change it's properties and apply then, or select another marker and apply it to the shape.




You can change each properties :

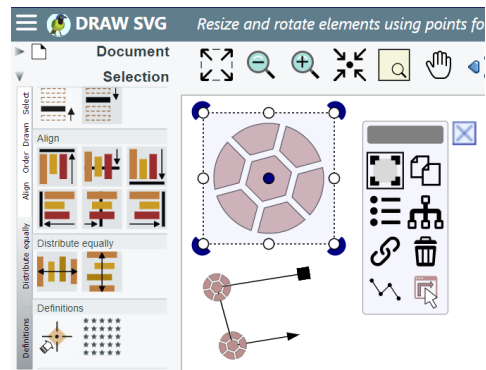
- **ViewBox** : Use text fields ( **x** , **y** , **width** , **height** ), icons (zoom/in zoom/out) and text field ratio for extending or reducing the area. To add a margin of 20% type 120<enter> in the ratio field.
- **Id** : the marker's id.
- **Size** : Define the dimensions and units of the marker. Use text fields ( **width** , **height** ), icons (zoom/in zoom/out) and text field ratio for extending or reducing the size of the tile.
- **Ref point** : change the coordinates of the reference point, move it's red circle or use **refX** / **refY** sliders and fields.
- **Elements** : List of all the graphic elements of the marker with their stroke and fill colors.

To customize the marker of the selected shape, make a copy, modify properties and apply. You can change for example dimensions and colors.

How to define a new marker

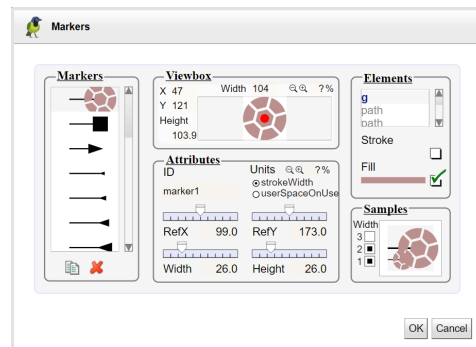
To **define** a new marker, draw elements of it's graphic, select them and use the marker

definition [78] tool  from the selection menu.



The marker is defined with the selection :

- The elements of the pattern are defined by a copy of the selection.
- The viewBox is set with the bounding box of the selection
- The units of the marker is set as strokeWidth and it's size equals to viewBox size



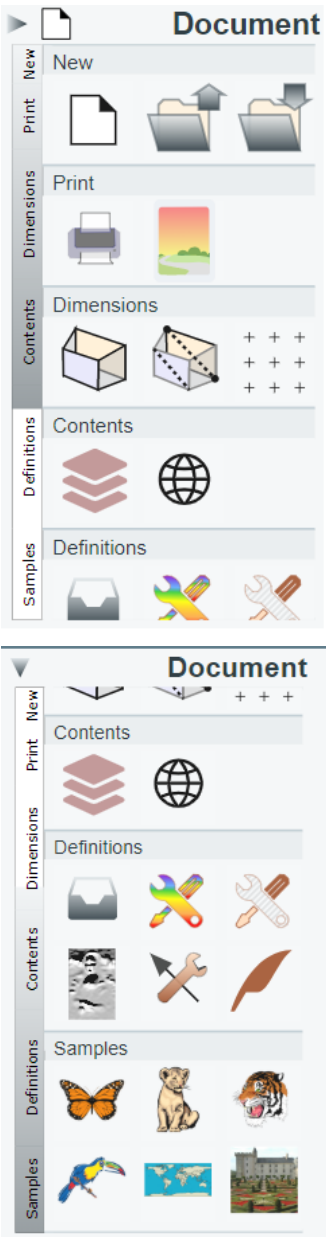
# Chapter 3. Manage documents

This chapter describes the features of document management Draw SVG.

## 3.1. Document menu

The functions of document management are grouped in this menu.

**Table 3.1. Document menu**

Menu	Sub-menus	
	<b>New</b>	Create a new document
		Open a document
		Save the document
	<b>Print</b>	Print the document
		Export the document as PNG image
	<b>Dimensions</b>	Define document dimensions
		Resizes the document by its bounding box
		Grid definition
	<b>Contents</b>	Browse layers
		Define multi-language support
	<b>Definitions</b>	Import definitions from a SVG file
		Edit gradient definitions
		Edit pattern definitions
		Edit filter definitions
		Edit marker definitions
	<b>Samples</b>	Butterfly
		Lion
		Tiger
		Toucan
		World map
	French Vilandry castle garden	




### Note

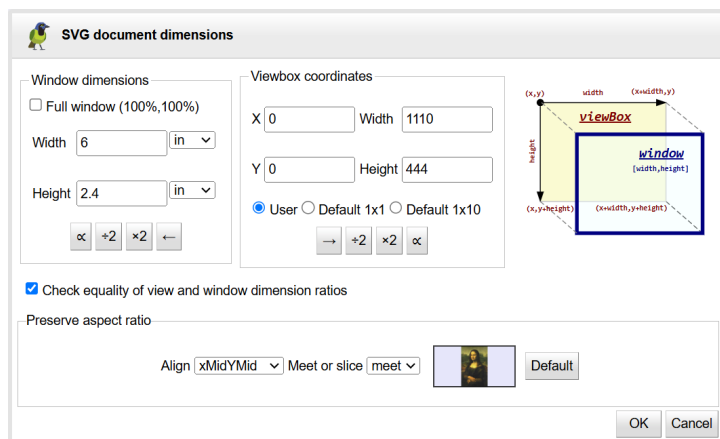
Since version 3, the functions (new, open, save, print, png export) are processed client side. The browser must comply HTML5.



## 3.2. Create new document

This function is available from the document menu [23] .

Item	Description
	This function creates a new SVG document by specifying its dimensions: the "viewBox" attribute of the SVG document and the window display size.



**Figure 3.1. New SVG**

The document size is fully characterized by :

- The drawing space and it's coordinate system defined by the "viewBox" attribute of the SVG document
- The physical dimensions with it's units, width and height of the window or the paper size.
- The projection method of the drawing space defined by the attribute "preserveAspectRatio" .

The display dimensions can be defined as:

- Full window (100%,100%) to occupy the entire window or the insert element of the SVG document.

This is the common mode for a WEB publication in an HTML page.

The preserveAspectRatio property defines how the document is displayed in the window.

- With fixed dimensions with a unit of measurement.

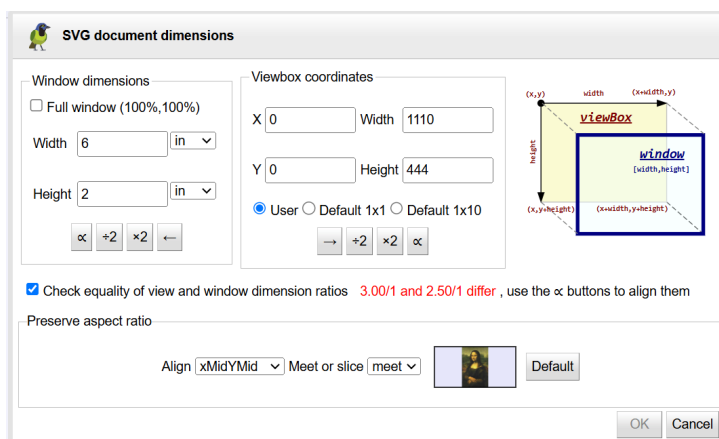
This is the usual mode for printing.

When the window dimensions are fixed, it is advisable to check the equality of the proportions of the display dimensions and the viewBox dimensions to ensure correct publication of the document.

If not equals, the panel indicates the differences between the two ratios and the OK button is deactivated.

The control can be removed to force the change.





In most cases the dimensions of the viewBox are set according to the content of the document and then the display dimensions are defined accordingly.

The display dimensions can thus be defined from those of the viewBox with the tools bar:



- Proportionally to viewBox dimensions, keeping the largest value (width or height) and its unit.

With one decimal if the value is less than 100.


- Equals to viewBox dimensions divided by 2
- Equals to viewBox dimensions twice
- Equal to the dimensions of viewBox in pixel units

Otherwise if the dimensions of the viewBox are fixed from those of the display, The viewbox coordinates can be defined :

- from the window size with one pixel equals one xy unit (default 1x1)
- from the window size with one pixel equals one ten xy units (default 1x10)
- With specific values (user)
- Dimensions equals to window dimensions divided by 2
- Dimensions equals to window dimensions twice
- Dimensions proportionally to window dimensions, keeping the largest value (width or height)

## 3.3. Open a document

This function is available from the document menu [23] .

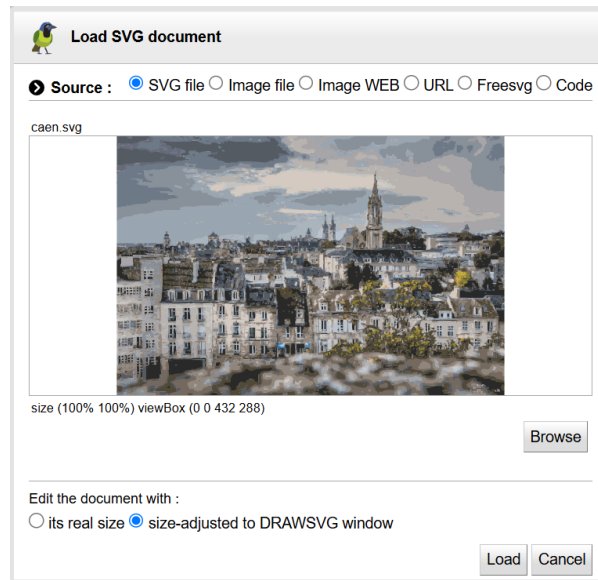
Item	Description
	<p>With this function you can load a SVG document from different sources :</p> <ul style="list-style-type: none"> <li>• from a svg file on your hard drive</li> <li>• from a image file on your hard drive</li> <li>• from a image found on the WEB</li> <li>• from a web URL</li> <li>• from the freesvg gallery</li> </ul>



Item	Description
	<ul style="list-style-type: none"> <li>By entering the SVG code by copy and paste</li> </ul>

### 3.3.1. Loading a SVG file

Check the **SVG file** source radio button. Select the svg file with the browse button and load it.



**Figure 3.2. Loading a SVG file**

If the document does not contains html audio/video controls, and if its size is different than (100%,100%), then the document can be edited :

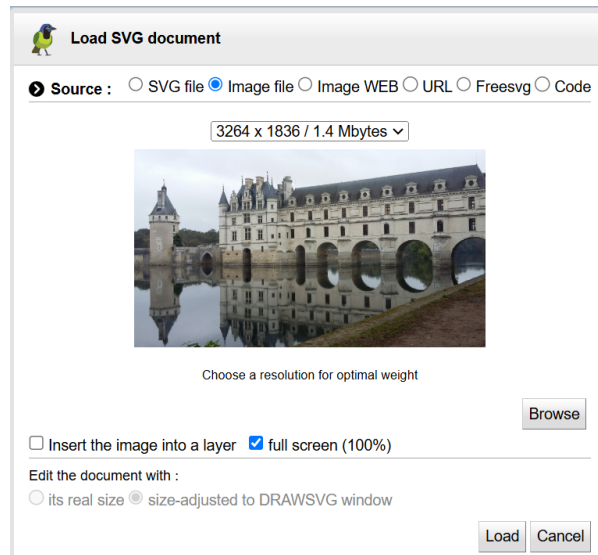
- with its real size, and scrollbars if needed
- or with an adjusted size to the drawsvg window.

Otherwise, the document is edited with its defined size and this option is disabled.

### 3.3.2. Loading an image file

This mode allows to set an SVG from an image file. The dimensions and the coordinate system of the document are identical to the image.

Check the **image file** source radio button. Select the image file with the browse button and load it.



**Figure 3.3. Loading an image file**

The image can be embedded in a separate layer or as a background image. The document dimensions are set by default to (100%,100%) or otherwise the same as the image in pixels.

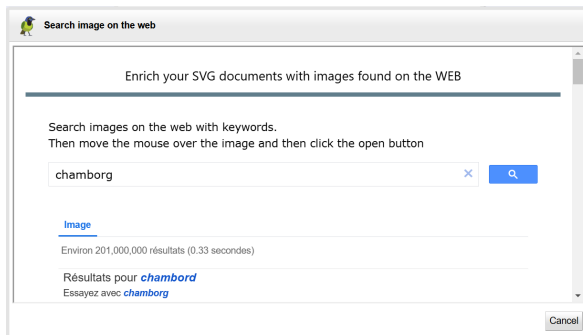
### 3.3.3. Loading an image from the WEB

This source allows to create an SVG document from an image found on the web.

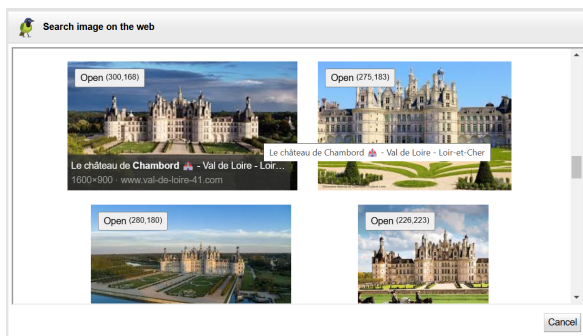
Check the **image WEB** source radio button and click the browse WEB button.

Then search for an image on the WEB using keywords with the image search WEB [55] dialog panel:

- Enter your keywords and click on the search button :



- Select the desired image then click on the open button :

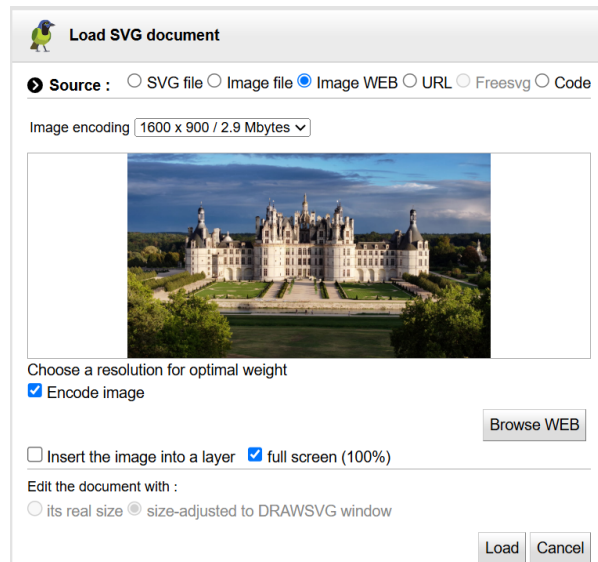


Then set the different options to create the document:





- The image can be encoded or referenced by its URL
- If encoded, choose the resolution to optimize its rendering and its weight
- The image can be embedded in a separate layer or as a background image
- The document dimensions are set by default to (100%,100%) or otherwise the same as the uploaded image in pixels



**Figure 3.4. Loading an image from the WEB**



### Note

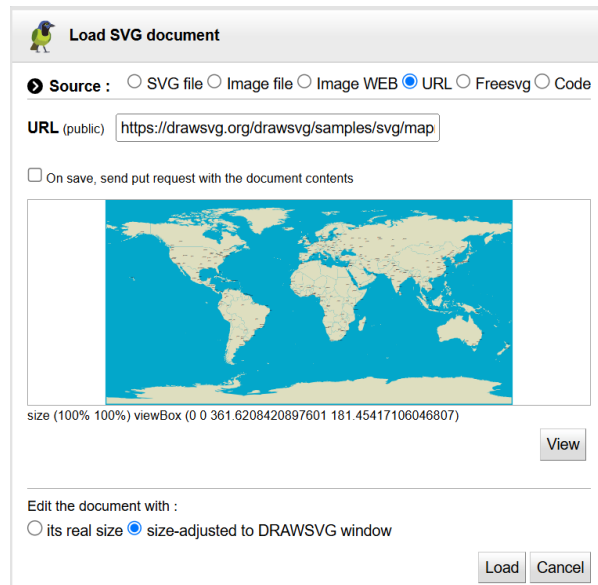
Creating a document from an image found on the WEB can be done directly with the drawWEB tool without opening the drawsvg editor.

Watch this example YouTube video .

### 3.3.4. Loading an URL

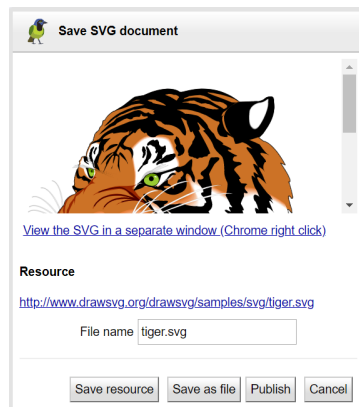
Check the **url** source radio button. Copy the URL of an SVG document from the browser to the edit field. Then view the URL and load it.

For example, to load <https://drawsvg.org/drawsvg/samples/svg/mapmonde.svg>



**Figure 3.5. Loading a SVG URL**

If the option save is checked, when saving, DRAWSVG will suggest to update the resource with an HTTP PUT Request with the document contents.



**Figure 3.6. Saving a SVG URL**

### 3.3.5. Loading a svg from freesvg gallery

FreeSVG is a well known WEB site with a lot of SVG files free to use. You can search documents in the gallery with tags, author names.

DRAWSG has a navigation GUI in the gallery of FreeSVG that lets you edit it's documents directly.

The source is only available with the expert [1] profile.

Check the **freesvg** source radio button. Query the gallery, browse documents, look them and load the selected one.





**Figure 3.7. Loading a SVG from Freesvg gallery**

See showcase load freesvg document



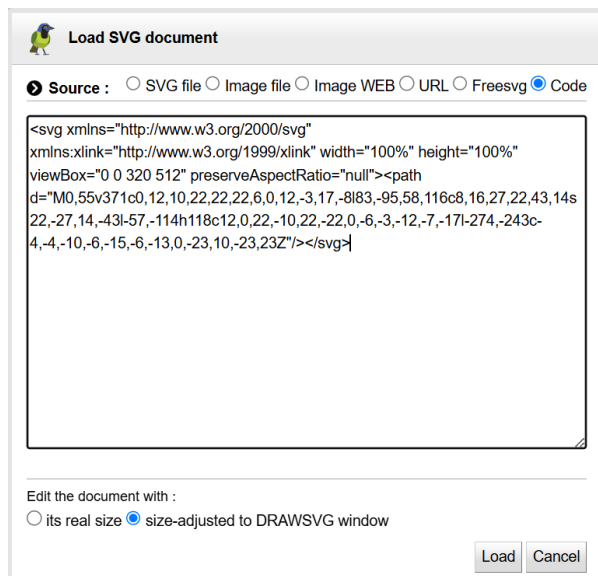
**Note**

This feature is not currently supported by the freesvg server.

**3.3.6. Load a SVG code**

This mode allows you to enter the code of an svg document.

Check the **code** source radio button. You can enter the code by copying and pasting from your editor it.



**Figure 3.8. Loading SVG code**

SVG Code Example

```
<svg xmlns="https://www.w3.org/2000/svg" xmlns:xlink="https://www.w3.org/1999/xlink" width="100%" height="100%"
```




```

viewBox="0 0 320 512" preserveAspectRatio="xMidYMid meet">
<path
d="M0,55v371c0,12,10,22,22,22,6,0,12,-3,17,-8183,-95,58,116c8,16,27,22,43,14s22,-27,14,-43l-57,-
h118c12,0,22,-10,22,-22,0,-6,-3,-12,-7,-17l-274,-243c-4,-4,-10,-6,-15,-6,-13,0,-23,10,-23,23Z"/
>
</svg>

```

### 3.4. Save the document

This function is available from the document menu [23] .

Item	Description
	<p>This function allows the user to save the svg document with differents ways :</p> <ul style="list-style-type: none"> <li>• Saving the document as a file on the disk of the computer</li> <li>• Updating the loaded resource by its URL if marked as available for update</li> <li>• Calling the jsChannel save service (see chapter integration [119] )</li> <li>• Publish on drawing board network</li> </ul> <p>The document can be viewed in a separate window.</p>



**Figure 3.9. Save dialog box**

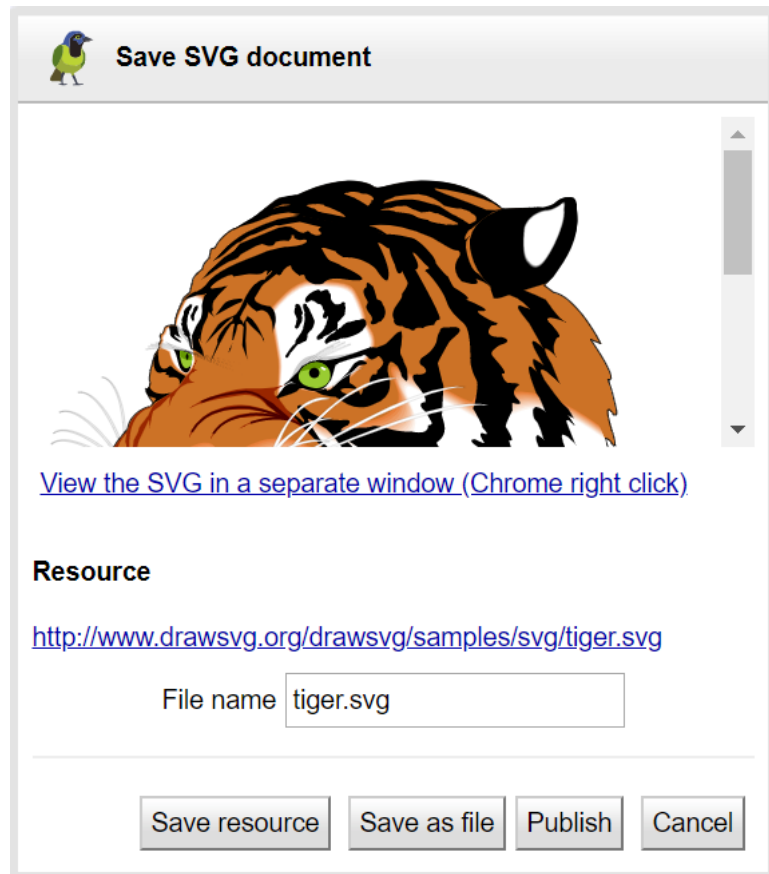
The dialog shows the document in a preview window, with a link to view it in a separate window and a button to save it as a file.



**Note**

The document is saved with the original view.

If the document has been loaded from an URL marked to be updated, the dialog show an other button "save resource"



**Figure 3.10. Saving a SVG URL**


If the document has been loaded from a jsChannel client application with a save service, the dialog shows an other button with the service name to call it (see chapter integration [119] ).

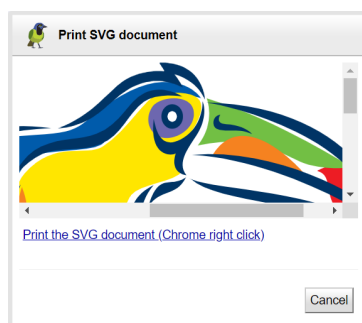


**Figure 3.11. Saving with a jsChannel service**

### 3.5. Print the document

This function is available from the document menu [23] .

Item	Description
	This function allows you to print the svg document.



**Figure 3.12. Print dialog box**

The dialog shows the document in a preview window, with a link to print it (on chrome browser you should right click on the link). The document is printed from a separated window.




**Note**

The document is printed with the current view.

### 3.6. Export the document as PNG

This function is available from the document menu [23] .

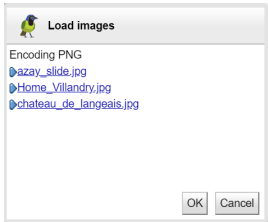

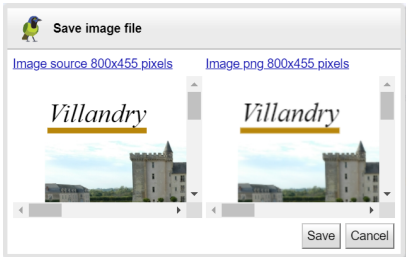
Item	Description
	This function allows to export the document as a PNG image and to save it on your disk.

This feature allows you to export the current view image in PNG format.

Export is performed in two or three steps:

- Loading images (or svg) referenced by HTTP URL by elements SVGImage for conversion png if the document contains.
- Definition of the size of the image
- Image preview and backup on the hard drive


**Table 3.2. Export PNG**

Load images	Size definition	Preview and save
		



### 3.7. Define document dimensions

This function is available from the document menu [23] .

Item	Description
	This function allows you to edit the document dimensions with the same dialog box than the creation function [24] .

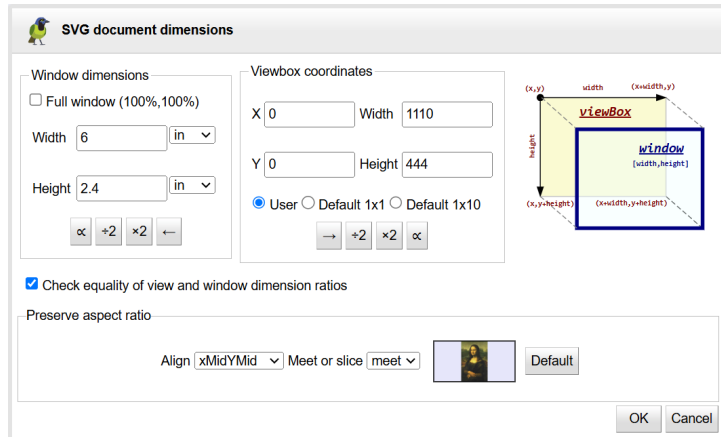



Figure 3.13. Dimensions dialog box

### 3.8. Resize document

This function is available from the document menu [23] .

Item	Description
	This function allows you to resize the document by a drawn rectangle : <ol style="list-style-type: none"> <li>1. Digitize the rectangle viewBox document by top left and bottom right corners,</li> <li>2. then apply the dimensions</li> </ol>

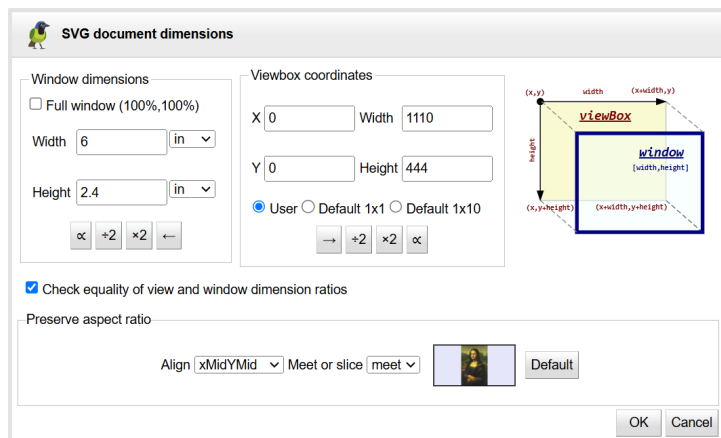


Figure 3.14. Resize document



This function uses the same dialog box than the creation function [24]

### 3.9. Layers management

#### Layer definition

A layer is a group (g element) of elements identified by an ID  
Layers are useful to organize large drawing by grouping elements by theme.  
They are used traditionally in cartographic maps to separate roads, buildings, utilities networks.

#### Layer hierarchy

Layers are defined as children of the document root element.  
Layers can also be defined as children of layers as needed.

#### Default layer

The document root element is identified as the default layer (named root).

#### Input layer

One layer is defined as the input layer to object creation, selection and edition.

The input layer can be selected from the menu bar :



#### Layers functions

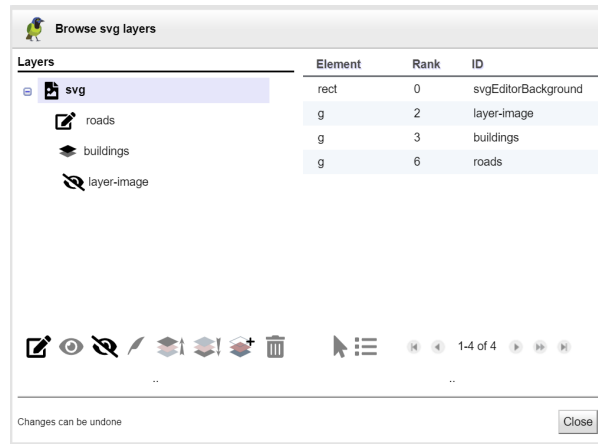
Layers functions are available from their dialog panel identified by it's icon 

The layers dialog panel offers full features:

- Select input layer
- Show/hide layers
- Edit layers style properties
- Change layers ID
- Change layers order
- Adding a layer
- Removing a layer
- Browse layer's elements



- Select layer's elements



**Figure 3.15. Layers dialog panel**

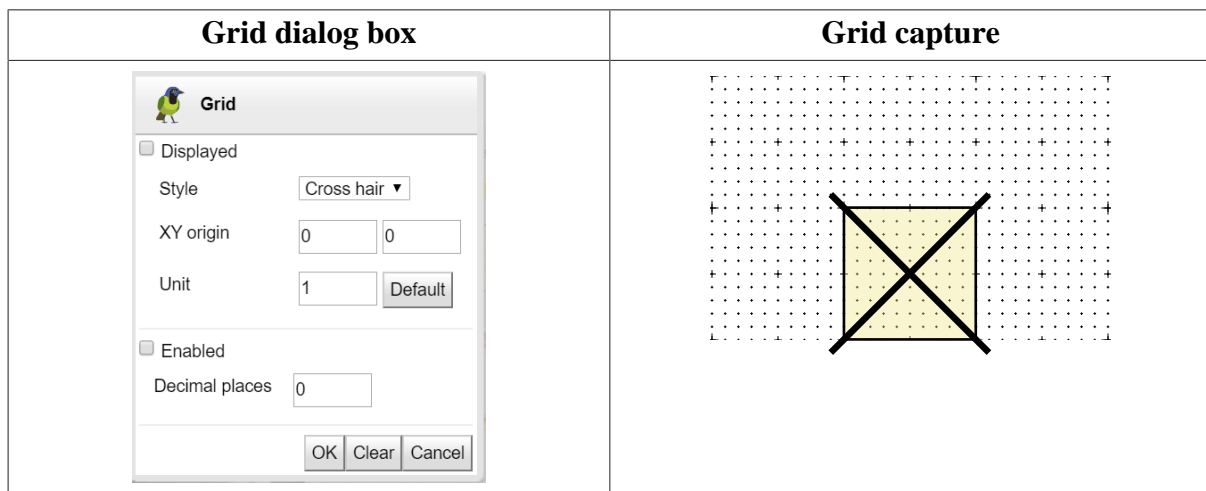
See showcases manage layers , style layers and change object layer for a full features demonstration.

### 3.10. Grid definition

This function is available from the document menu [23] .

Item	Description
⋮	This function allows you to define a grid described in the coordinate system of the SVG document. The grid can be displayed and enabled to capture points of elements.

**Table 3.3. Grid definition**




See grid showcase.

### 3.11. Import definitions

This function is available from the document menu [23] .



Item	Description
	<p>This function allows you to import definitions declared in another svg file. Choose File, the panel displays two lists: the element types defined in the file and their components. Select the types of elements (tags) and the definitions to import.</p> <p>The panel does not show the elements already defined in the document.</p> <p>This function allows you to maintain a catalog of definitions (gradients, patterns, markers, filters) reusable.</p>

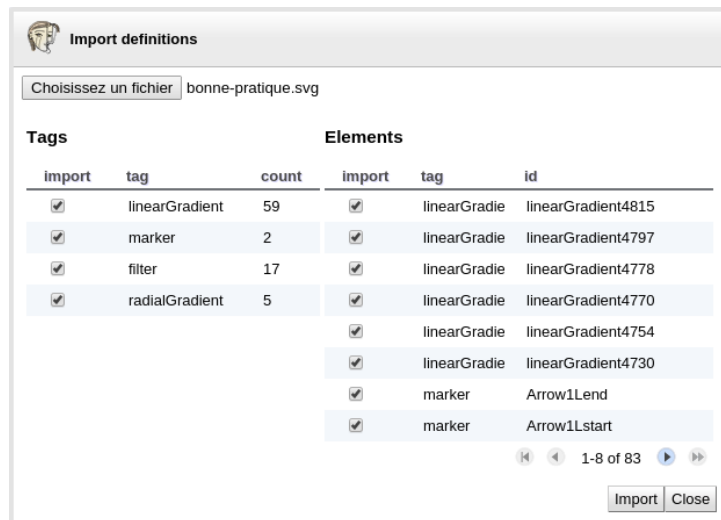

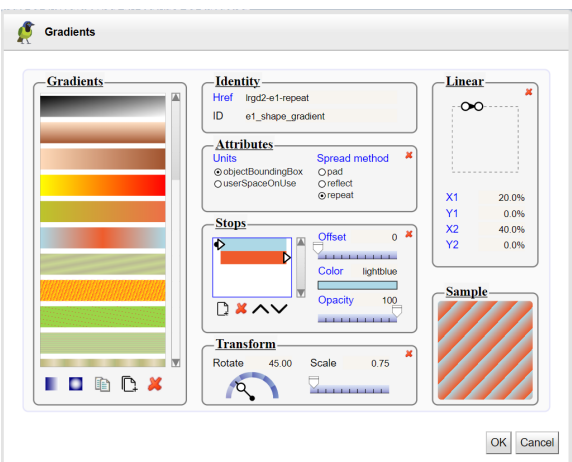


Figure 3.16. Import definitions panel

### 3.12. Gradient definitions

This function is available from the document menu [23] , the stroke style properties menu [69] , the fill style properties menu [71] and the properties dialog panel [56] .


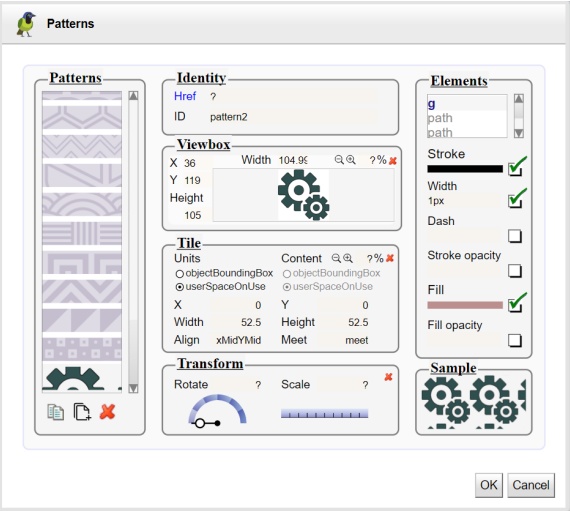
Item	Description
	<p>This action launches the gradient definition [76] tool.</p> 



See creating gradients showcase.

### 3.13. Pattern definitions


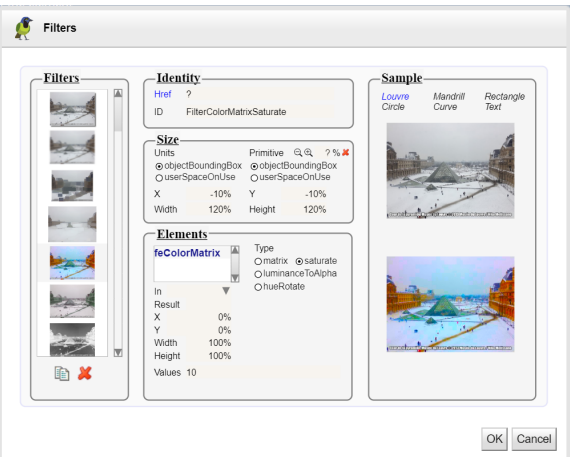
This function is available from the document menu [23] , the stroke style properties menu [69] , the fill style properties menu [71] and the properties dialog panel [56] .

Item	Description
	<p>This action launches the pattern definition [79] tool.</p> 

See creating patterns showcase.

### 3.14. Filter definitions

This function is available from the document menu [23] and the properties dialog panel [56] .

Item	Description
	<p>This action launches the filter definition [81] tool.</p> 


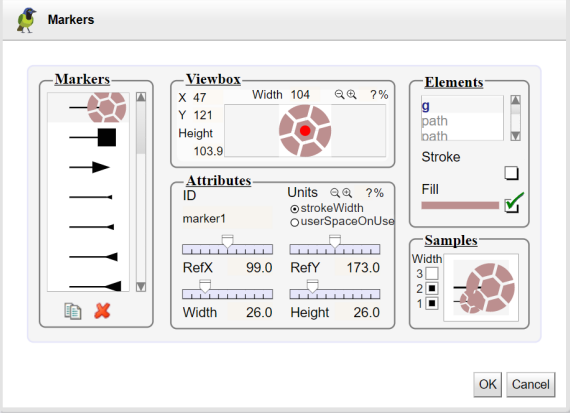
See using filters showcase.



### 3.15. Marker definitions

This function is available from the document menu [23] , the marker style menu [73] and the properties dialog panel [56] .

See creating , using markers showcases for interactive demonstrations.

Item	Description
	<p>This action launches the marker definition [78] tool.</p> 

# Chapter 4. Editing selection

This chapter describes the interactive editing of graphical elements in a document.

The elements of the current selection can be changed from the Selection menu or from the floating menu displayed near the selection.

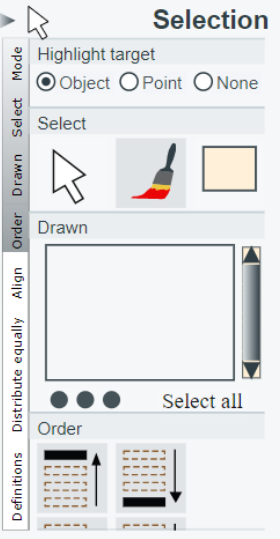
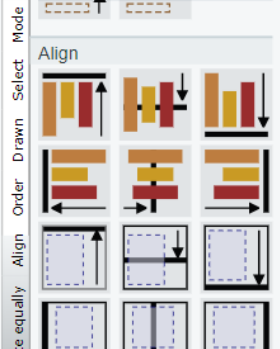
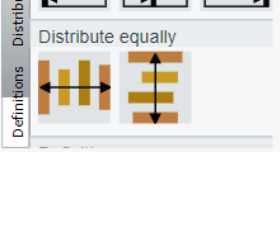
The selection can be extended to the group membership of selected items with the group select function [49] .

To explore all the drawing tasks and editing tasks available on selected objects, see the drawing tasks and editors pages with showcase links on each task.

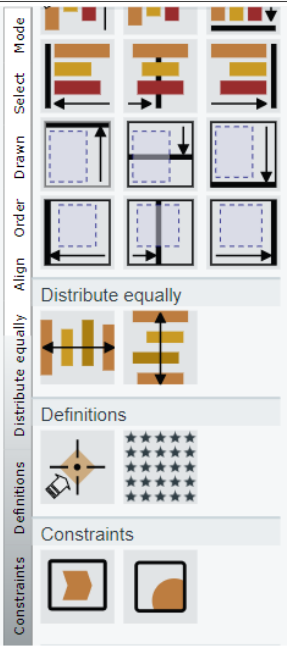
## 4.1. Selection menu

This menu contains the functions applicable to the current selection.

**Table 4.1. Selection menu**


Menu	Sub-menus	
	Mode	Enable/disable target highlighter mode (Object, Point, None) If enabled, when stopping mouse move, the object/point under or near the pointer is highlighted to easy selecting. See showcases: Select with target highlighting Select point
	Select	Select graphical items Apply the style of the selected item to another item. Select the background rectangle.
	Drawn	List of drawn elements, show selection. Select all elements See showcase Select all elements .
	Order	Moves the elements of the selection in front Moves the elements of the selection in back Moves the elements of the selection forward one Moves the elements of the selection back one
	Align	Align the elements of selection on their upper edge.
		Align the elements of selection on their middle edge.
		Align the elements of selection on their bottom edge.



Menu	Sub-menus	
	Align the elements of selection on their left edge.	
	Align the elements of selection on their center edge.	
	Align the elements of selection on their right edge.	
	Align to frame	Align the elements of selection on the frame upper edge.
		Align the elements of selection on the frame middle edge.
		Align the elements of selection on the frame bottom edge.
		Align the elements of selection on the frame left edge.
		Align the elements of selection on the frame center edge.
		Align the elements of selection on the frame right edge.
	Definitions	Use the elements of the selection to create a marker with the marker definition [78] tool.
		Use the elements of the selection to create a pattern with the pattern definition [79] tool.
	Constraints	Add bounding box constraint [84] .
		Add clipping constraint [85] .

## 4.2. Select elements


This function is available from the selection menu [40] and from the actions bar [9] .

Item	Description
	<p>This function allows you to select elements by graphic selection.</p> <ul style="list-style-type: none"> <li>• Select one item by clicking on the graphic or nearby. The function selects the pointed element or the closest.</li> <li>• To select multiple items, draw the bounding rectangle.</li> </ul> <p>Enable target highlighter to easy selection. See showcase Select with target highlighting .</p>

## 4.3. Apply style

This function is available from the selection menu [40] .



Item	Description
	<p>This function allows you to apply the style properties of the selected item to a new selection.</p> <p>Click on a drawn element to apply the style of the current selected one, the clicked element becomes the new selection.</p>


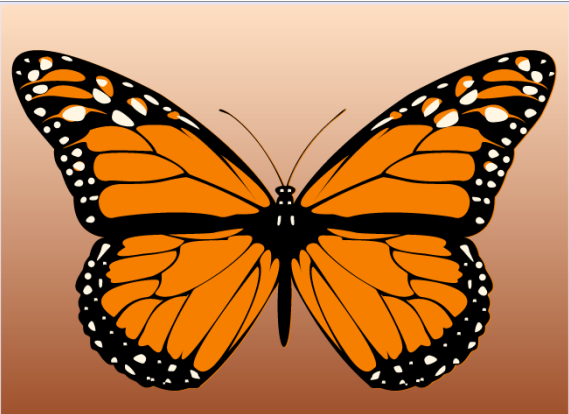


**Note**

The function is disabled if the selection is empty or has more than one element.


## 4.4. Select background rectangle

This function is available from the selection menu [40] .

Item	Description
	<p>This function selects the background rectangle.</p> <p>By default the background rectangle is transparent. After selected, its style can be customized with the properties panel [56] or with the fill style menu. [71]</p>
	
<p>Background rectangle with fill gradient</p>	

## 4.5. Select drawn elements


This function is available from the selection menu [40] .

Item	Description
	<p>This component list identifies drawn elements and allows you to edit the selection. The &lt;ctrl&gt; key (add or remove from the selection) and &lt;shift&gt; key (range select) are supported.</p>

## 4.6. Order front

This function is available from the selection menu [40] .



Item	Description
	This function moves the elements of the selection above drawing.



**Note**


The function is deactivated if the following conditions are not met..

- Selection should not be empty
- The elements of the selection must be son of the same parent

This function reorders the elements inside a single node which can be either the <svg> document root node or a grouping element <g>.

## 4.7. Order back

This function is available from the selection menu [40] .

Item	Description
	This function moves the elements of the selection below other.



**Note**





The function is deactivated if the following conditions are not met..

- Selection should not be empty
- The elements of the selection must be son of the same parent

This function reorders the elements inside a single node which can be either the <svg> document root node or a grouping element <g>.



## 4.8. Align

This functions are available from the selection menu [40] .

Item	Description
	This function move selected items to align their top edge.
	This function move selected items to align their median horizontal axis.
	This function move selected items to align their bottom edge.
	This function move selected items to align their left edge.





Item	Description
	This function move selected items to align their median vertical axis.
	This function move selected items to align their right edge.





**Note**

This functions area activated if two or more items are selected.

## 4.9. Space equally

This functions area available from the selection menu [40] .

Item	Description
	This function move the selected items to distribute equally space between them along the horizontal axis. First and last items are not moved.
	This function move the selected items to distribute equally space between them along the vertical axis. First and last items are not moved.




**Note**

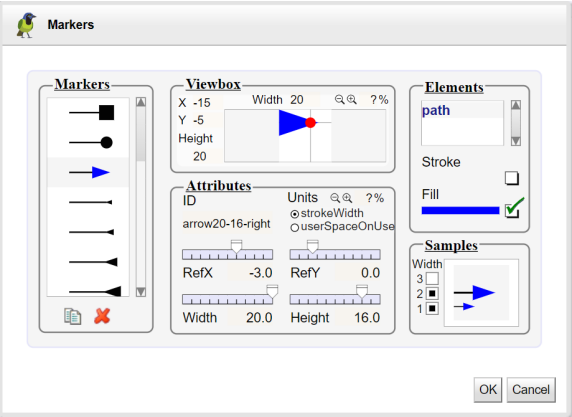
This functions area activated if three or more items are selected.

## 4.10. Marker definition

This function is available from the selection menu [40] .

See creating , using markers showcases for interactive demonstrations.

Item	Description
	<p>This function allows you to set a marker with selected items.</p> <p>The marker is defined with the selection :</p> <ul style="list-style-type: none"> <li>• The elements of the pattern are defined by a copy of the selection.</li> <li>• The viewBox is set with the bounding box of the selection</li> <li>• The units of the marker is set as strokeWidth and it's size equals to viewBox size</li> </ul> <p>The new marker is edited with the marker dialog panel [78] .</p>

Item	Description
	



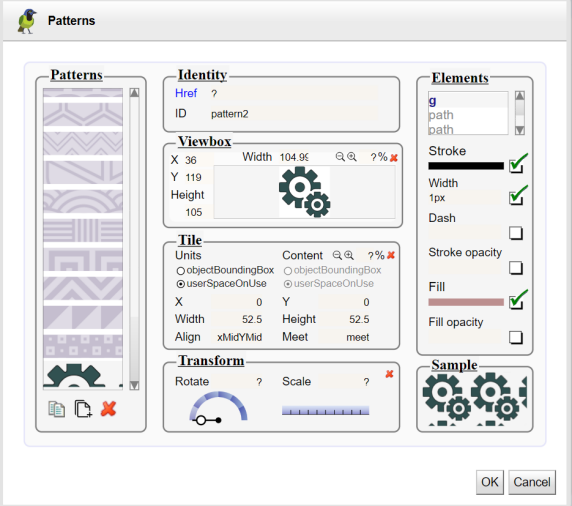
**Note**

The function is disabled if the selection is empty.

## 4.11. Pattern definition

This function is available from the selection menu [40] .

See creating , using patterns showcases for interactive demonstrations.

Item	Description
<p>★★★★★</p> <p>★★★★★</p> <p>★★★★★</p> <p>★★★★★</p> <p>★★★★★</p>	<p>This function allows you to set a pattern with selected items.</p> <p>The pattern is defined with the selection :</p> <ul style="list-style-type: none"> <li>• The elements of the pattern are defined by a copy of the selection.</li> <li>• The viewbox is set with the bounding box of the selection</li> <li>• The units of the tile is set as userSpaceOnUse and it's size equals to viewbox size</li> </ul> <p>The new pattern is edited with the pattern dialog panel [79] .</p> 



**Note**

The function is disabled if the selection is empty.



## 4.12. Editors

Editors available for the current selection are displayed in the floating menu.








All changes can be canceled and replayed with undo [10] and redo [10] action.

Each editor is characterized by:





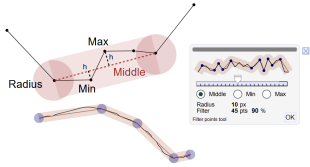

- It's icon
- types of editable elements,
- The number of elements simultaneously edited (one or many).

This list is not exhaustive, see the full editor list also showcase tool for interactive demonstrations.

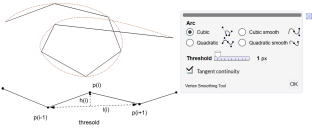


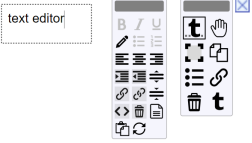



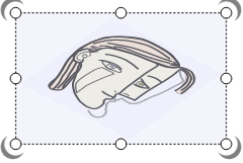
**Table 4.2. Editors**










Editor	Selection	Cardinality	Description
Edit size 	Image, rectangle, svg	one	Edit position and size of the element by two control points. See also draw rectangle showcase for an interactive demonstration of how to draw a rectangle.
Edit circle 	Circle	one	Edit center and radius of the circle by two control points.
Edit ellipse 	Ellipse	one	Edit center and radius of the ellipse by two control points.
Edit map 	Google map image	one	Edit the map location. Display the Google map dialog panel [50] for searching a place and browsing the map. See also draw map showcase for an interactive demonstration of how to insert and edit a Google Map in your SVG drawing..
Edit line 	Line	one	Move each point of the line. See also draw line showcase for an interactive demonstration of how to draw a line.
Insert points 	Polyline, polygon	one	Insert point. A blue dot is drawn in the middle of each segment. Move the points to be inserted.
Delete points 	Polyline, polygon	one	Delete point. A red dot is drawn on each point. Click the points to be deleted.



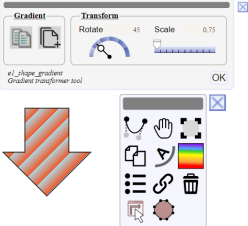

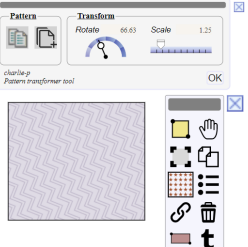



Editor	Selection	Cardinality	Description
<p>Move points</p> 	<p>Polyline, polygon</p>	<p>one</p>	<p>Move each point of the selected element.</p>
<p>Edit circle arc</p> 	<p>Circle arc</p>	<p>one</p>	<p>Edit the geometry of a circle arc by three control points. A circle arc is a path with an elliptical arc curve commands</p>
<p>Edit path</p> 	<p>Path</p>	<p>one</p>	<p>Move each point and tangent point of the path.</p>
<p>Filter points</p> 	<p>Polyline, polygon, path</p>	<p>one</p>	<p>Filter the points of a geometric element (polyline, polygon or path) by removing unnecessary points. The filter is defined by a corridor of a given radius to fix the points of the chord and delete those whose distance to the chord is less than the radius. The curve segments of the path elements are replaced by straight line segments See filter points showcase. Display the filter points panel [51] near the selected element.</p> 
<p>Smooth vertices</p> 	<p>Polyline, polygon, path</p>	<p>one</p>	<p>Smoothing the vertices of a polyline, polygon or path. Smoothing goes through the points of the curve with arc of 4 types (cubic, cubic smooth, quadratic, quadratic smooth). It is advisable to first remove redundant points using the filter points [47] function. Polylines and polygons are replaced by paths. See smooth vertices showcase. Display the svg Smooth vertices panel [52] near the curve.</p>



Editor	Selection	Cardinality	Description
			
<p>Close path</p> 	Path	one	<p>Close a path. Add a closepath command to the path.</p>
<p>Edit text</p> 	Text	one	<p>Edit text.</p>  <p>This function interacts directly with the drawn text. Click on the text to insert characters by typing on the keyboard. Use the icons to move cursor or delete characters.</p> <p>See also draw simple text , multiline text , text editor showcases for an interactive demonstration of how to draw and edit a text.</p>
<p>Edit textPath offset</p> 	TextPath	one	<p>Edit the offset attribute of a textPath .</p> <p>Display the svg textPath offset panel [52] near the text.</p> <p>See also drawing text on a path showcase for an interactive demonstration.</p>
<p>Translate xy</p> 	Circle, ellipse, polyline, polygon, rectangle, path, image, svg, text, tspan, use	many	<p>Translate xy coordinates of elements.</p> <p>This editor does not set the transform attribute.</p>
<p>Transform</p> 	all except tspan and textPath	many	<p>Apply a transform matrix to elements.</p>  <p>The transformation matrix is defined using eight control points and four half moons rotation.</p>

Editor	Selection	Cardinality	Description
Copy 	Circle, ellipse, polyline, polygon, rectangle, path, image, svg, text, tspan, use	many	Duplicate elements.
Load image file 	Image	one	Load an image file, then encode it's content as a dataURI to set the href attribute of the image with the image file load [54] dialog panel.
Download image from the WEB 	Image	one	Search for an image on the WEB using keywords with the image search WEB [55] dialog panel. Then encode it's content as a dataURI to set the href attribute of the selected image element with the WEB image load [56] dialog panel.
Load image URL 	Image	one	Load the HTTP URL referenced by the image, then encode it's content as a dataURI to set the href attribute of the image with the image url load [55] dialog panel.
Load svg 	svg	one	Set the content of the included svg [63] by loading a svg document from a file, a HTTP URL or from the FreeSVG gallery. The editor uses the same dialog than open document [25] .
Add textPath 	Path	one	Add a textPath on the selected path. See also drawing text on a path showcase for an interactive demonstration.
Edit properties 	all	one	Edit properties of the selected element with the properties [56] dialog panel.
Select group 	all	many	Select parent group.
Group elements 	all	many	Create a group and move selected items inside the group. You can assign style properties to the group. Rendering a member of the group is made with the values of style properties set on the element or its group.

Editor	Selection	Cardinality	Description
			To force the rendering with the group's style, we must unset the properties on group elements.
Ungroup elements 	g	one	Delete the selected group and move all its children to the group's parent item.
Transform gradient 	all	one	<p>Customize the gradient of the selected shape by copying or extension and transformation of it.</p>  <p>see transform gradient svg panel [53] See also creating , using gradients showcases for interactive demonstrations.</p>
Transform pattern 	all	one	<p>Customize the pattern of the selected shape by copying or extension and transformation of it.</p>  <p>see transform pattern svg panel [54] See also creating , using patterns showcases for interactive demonstrations.</p>
Delete 	all	many	Delete selection.

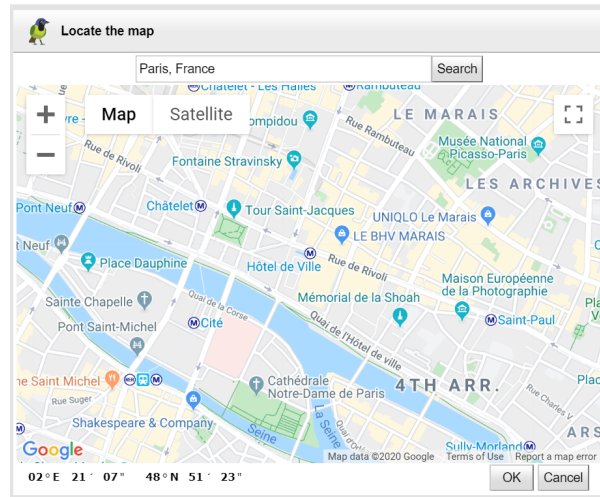
## 4.13. Editor panels

### 4.13.1. Map editor dialog panel

This tool is used to edit an image which shows a google map.

The map editor display a Google map dialog panel for searching a place and browsing the map.





**Figure 4.1. Map editor panel**

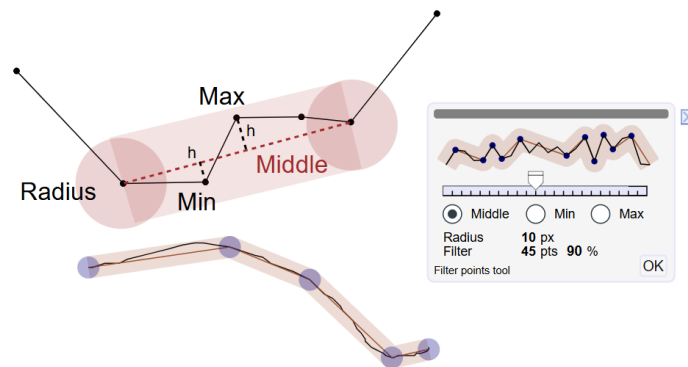
See also draw map showcase for an interactive demonstration of how to insert and edit a Google Map in your SVG drawing..

### 4.13.2. Filter points panel



The filter points editor can be applied to polylines, polygons and paths to remove unnecessary points of the selected element.

The tool is displayed as a svg panel inside the document.



**Figure 4.2. Filter points panel**

The filter is defined by a corridor of a given radius to fix the points of the rope line and delete those whose distance to the rope line is less than the radius.

The panel has a slider to set the filter radius in pixels and a radio button to set the position of the rope line when the corridor includes one or more points in three ways (Middle, Min, Max):

- **Middle** position (default). The rope line connects each end of the corridor and excludes all included points.
- **Min** position. The rope line passes through the point of lowest height  $h$ , this reduces the area of the polygon if it is described clockwise.





- **Max** position. The rope passes through the point of greatest height  $h$ , this increases the area of the polygon if described clockwise.

To make the filter logic more visual and understandable:

- The filter is drawn in transparency with a circle on each point of the rope line and its corridor to show the filtered points.
- The rope line is drawn in brown color.
- The panel displays the number of points removed and its percentage.

The curve segments of the path elements are replaced by straight line segments

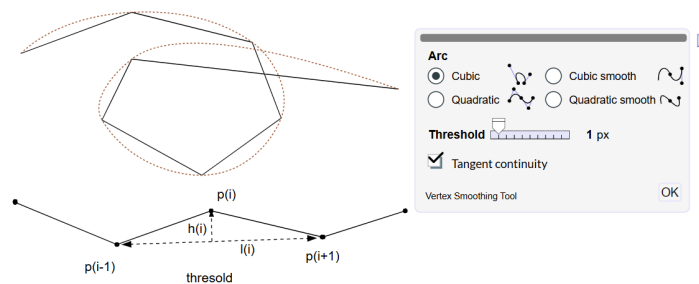
See filter points showcase.

### 4.13.3. Smooth vertices panel



Smoothing the vertices of a polyline, polygon or path.

The tool is displayed as a svg panel inside the document.



**Figure 4.3. Smooth vertices panel**

The curve is smoothed with Bézier arcs of degree 3 or 2 (cubic, smooth cubic, quadratic, smooth quadratic).

The panel consists of:

- A radio button to select the arc type (cubic, cubic smooth, quadratic, quadratic smooth)
- A slider to define the threshold distance
- A checkbox for continuity of tangents of cubic arcs

The threshold distance sets the two smoothing conditions for a vertex  $p(i)$  as follows:

- the height  $h(i)$  must be greater than the threshold
- the length  $l(i)$  of the line  $(i-1,i+1)$  must be greater than twice the threshold

The smoothed curve is plotted in brown dotted lines. It is advisable to first remove redundant points using the filter points [47] function.

Polylines and polygons are replaced by paths.

See smooth vertices showcase.

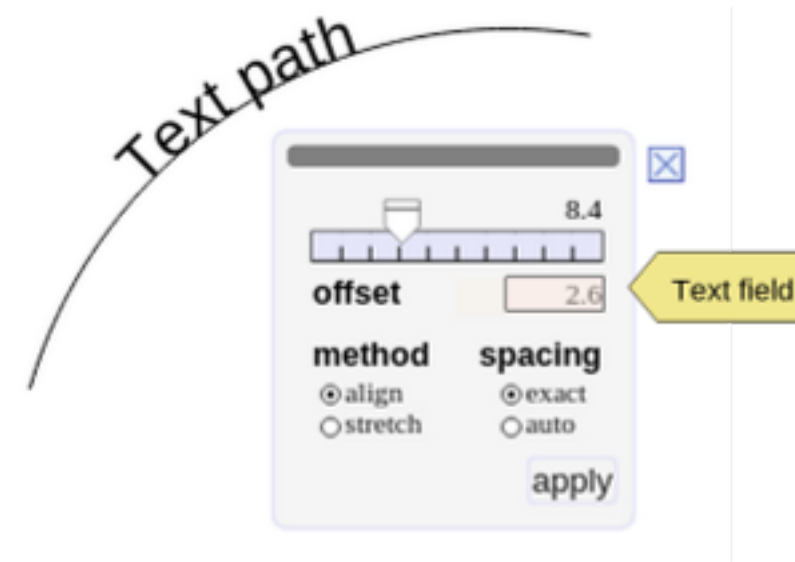
### 4.13.4. TextPath offset panel

The offset attribute of a textPath defines the curvilinear abscissa of the text on the associated path .



The offset can be adjusted with the slider of the editor's svg panel, or entered directly into the field.

The tool is displayed as a svg panel inside the document.



**Figure 4.4. TextPath offset panel**

See also drawing text on a path showcase for an interactive demonstration.

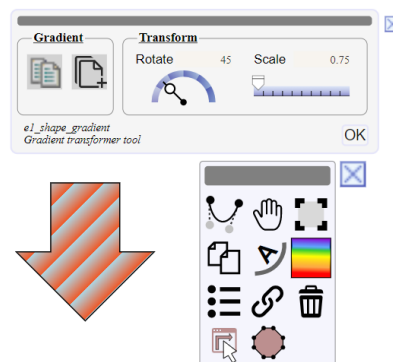
#### 4.13.5. Transform gradient panel

This tool is used to customize the gradient of a shape

- To prevent modification of the shared gradient definition, make a copy or an extension of the gradient,
- Then define the transformation with the rotation angle and the scale factor

Apply the transformation or close the panel without modification.

Customization by extension has the advantage of minimizing the volume and continue to benefit from changes made to the basic gradient.



**Figure 4.5. Transform gradient panel**

See also creating , using gradients showcases for interactive demonstrations.

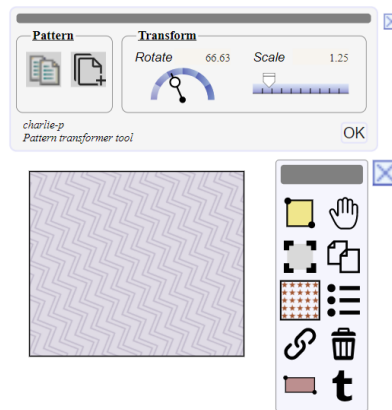
### 4.13.6. Transform pattern panel

This tool is used to customize the pattern of a shape

- To prevent modification of the shared pattern definition, make a copy or an extension of the pattern,
- Then define the transformation with the rotation angle and the scale factor

Apply the transformation or close the panel without modification.

Customization by extension has the advantage of minimizing the volume and continue to benefit from changes made to the basic pattern.



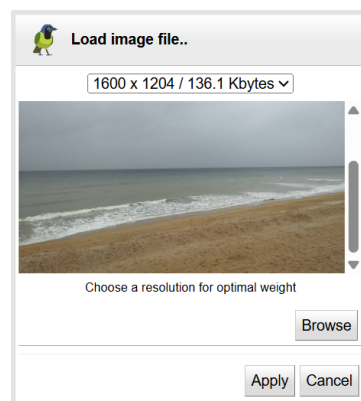
**Figure 4.6. Transform pattern panel**

See creating , using patterns showcases for interactive demonstrations.

### 4.13.7. Image file load panel

This panel loads an image file, then encode it's content as a dataURI to set the href attribute of the selected image element :

- Select the image resolution to optimize the encoded size based on the expected rendering quality.
- Click the Apply button to set the href attribute of the selected image element.



**Figure 4.7. Image file load panel**

See also draw image showcase for an interactive demonstration of how to draw an image.



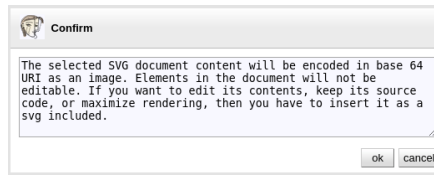
### Note

It's also possible to load a svg file by this way.

But in this case the most suitable method is to use the element `svg` included for the following reasons:

- The vector display renders the document with its original resolution and quality.
- The content of the included `svg` is editable.

The editor asks for confirmation to avoid confusion and to learn about the alternative use.



### 4.13.8. Image search WEB panel

This panel integrates Google's image search function on the WEB.

Enter your keywords, then click the search button to view matching images:

- For a mouse-enabled screen, hover over the images to display the upload button, then click the upload button for the desired image.
- For a touch screen without a mouse, click on the images to bring up the load button, then click the load button of the desired image.

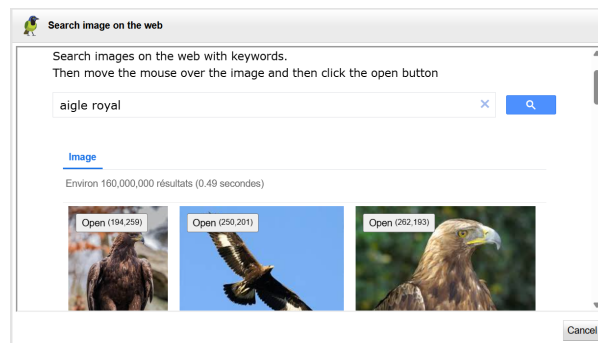


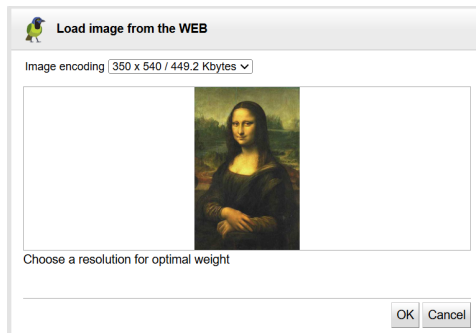
Figure 4.8. Image search WEB panel

### 4.13.9. Image url load panel

This panel loads the HTTP URL referenced by the selected image element, then encode it's content as a `dataURI` to set its `href` attribute :

- Select the image resolution to optimize the encoded size based on the expected rendering quality.
- Click the OK button to set the `href` attribute of the selected image element.





**Figure 4.9. Image URL load panel**



**Note**

It's also possible to view a svg URL by this way.

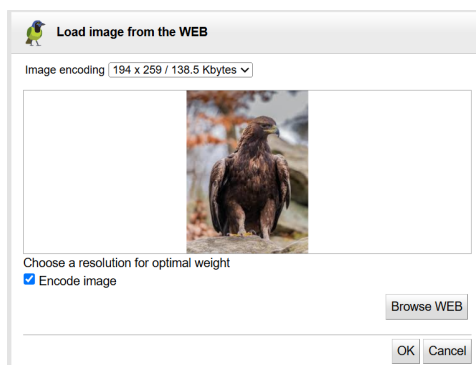
For the same reasons detailed previously in image file load [54] , the most suitable method is to use the element `svg` included.

The URL is downloaded via a proxy for safety reasons. The URL must be accessible from the server.

**4.13.10. WEB Image load panel**

This panel loads an image searched on the WEB, encodes its content as a dataURI to set the href attribute of the selected image element :

- Select the image resolution to optimize the encoded size based on the expected rendering quality.
- Uncheck the encoding option to reference the image by its URL.
- Click the Browse button to search for another image.
- Click the OK button to set the href attribute of the selected image element.



**Figure 4.10. WEB image load panel**

**4.13.11. Properties dialog panel**



This editor allows you to edit all properties of the selected item which are supported by DRAW-SVG.

Properties are organized into tabs :

- Element
- Stroke style

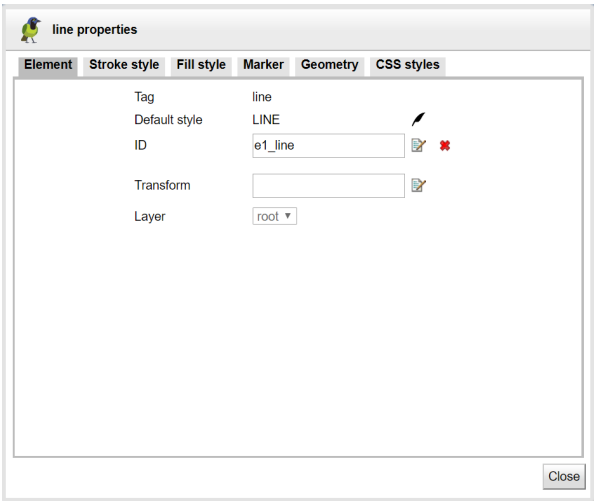


- Fill style
- Marker style
- Geometry
- CSS styles

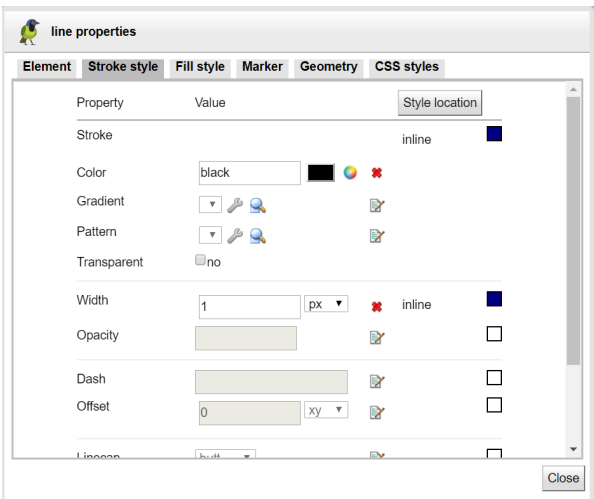
Each property is followed by an icon to edit its value  or to delete it .

See also showcase using css styles

**Table 4.3. Properties tab element panel**

Tab element	Properties
	<p>Element's attributes</p> <ul style="list-style-type: none"> <li>• Tag, element type</li> <li>• id, element id</li> <li>• transform, the transformation matrix</li> </ul>

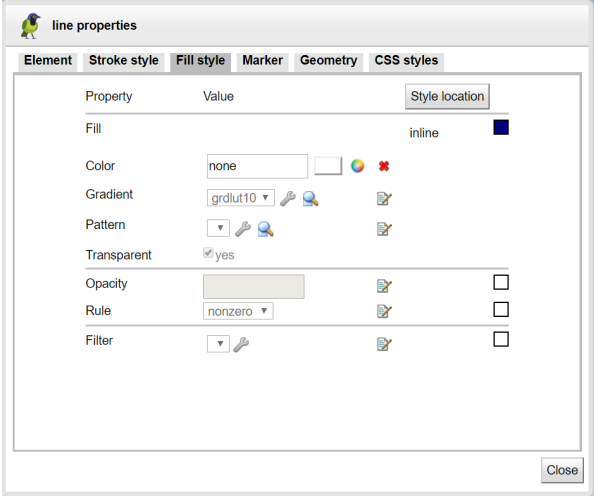
**Table 4.4. Properties tab stroke style panel**

Tab stroke style	Properties
	<p>Element's stroke style properties</p> <ul style="list-style-type: none"> <li>• Color, stroke color with color chooser [75] button</li> <li>• effect, scaling stroke (SVG 1.2)</li> <li>• gradient, stroke gradient with customize [76] and chooser [77] tools buttons</li> <li>• pattern, stroke pattern with customize [79] and chooser [81] tools buttons</li> <li>• width, stroke width value and unit</li> <li>• dash, stroke dash array</li> <li>• offset, stroke dash offset</li> <li>• stroke line cap (butt, round, square)</li> </ul>

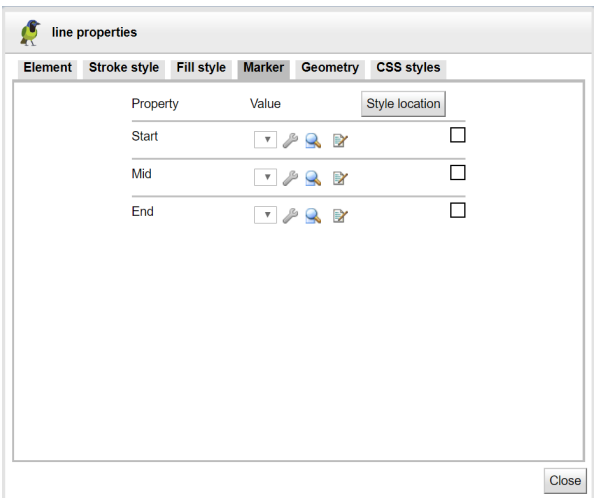


Tab stroke style	Properties
	<ul style="list-style-type: none"> <li>stroke line join (miter, round, bevel)</li> <li>stroke opacity, float value between 0 and 1</li> <li>stroke miter-limit</li> </ul>

**Table 4.5. Properties tab fill style panel**

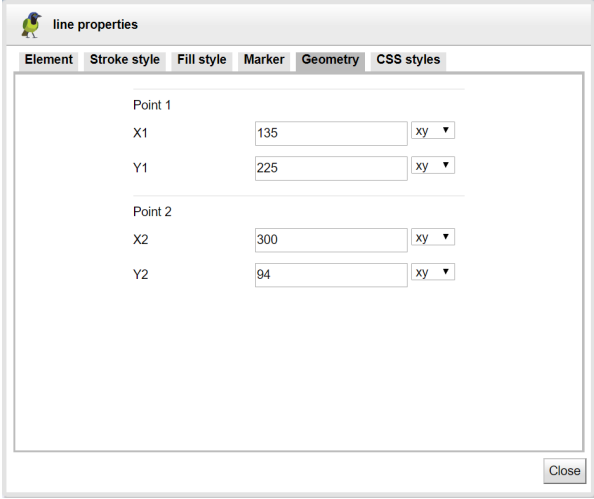
Tab fill style	Properties
	<p>Element's fill style properties</p> <ul style="list-style-type: none"> <li>Color, fill color with color chooser [75] button</li> <li>gradient, fill gradient with customize [76] and chooser [77] tools buttons</li> <li>pattern, fill pattern with customize [79] and chooser [81] tools buttons</li> <li>fill opacity, float value between 0 and 1</li> <li>filter with customize [81] tool</li> <li>fill rule</li> </ul>

**Table 4.6. Properties tab marker style panel**

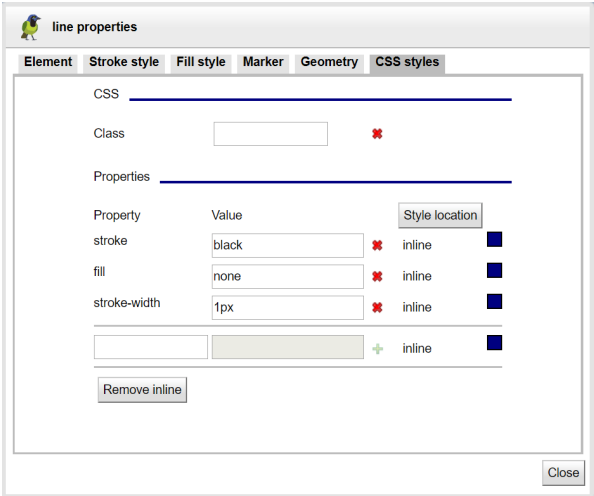
Tab marker style	Properties
	<p>Element's marker style properties</p> <ul style="list-style-type: none"> <li>marker start with customize [78] and chooser [79] tools buttons</li> <li>marker mid with customize [78] and chooser [79] tools buttons</li> <li>marker end with customize [78] and chooser [79] tools buttons</li> </ul>



**Table 4.7. Properties tab geometry panel**

Tab geometry	Properties
	<p>Definition of the geometry of the element. For a path element, shows each command with arguments.</p> <p>See showcase edit geometry</p>

**Table 4.8. Properties CSS styles panel**

Tab CSS	Properties
	<p>Define CSS classes, move properties from inline location to classes.</p> <p>See showcase using css styles</p>



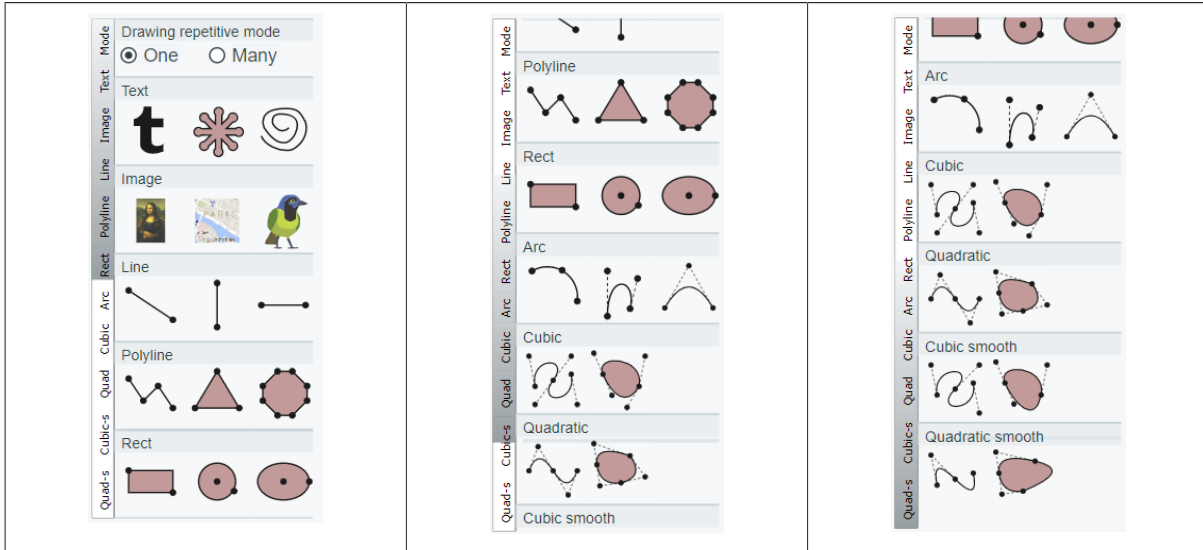
# Chapter 5. Drawing elements

This chapter describes the interactive drawing of graphical elements in a document.

## 5.1. Drawing menu

The menu offers more than 20 drawing tasks classified into 10 tabs.

**Table 5.1. Menu drawing elements**



## 5.2. Drawing repetitive mode

By default, drawing an object follows on its selection and choosing an editor to modify its properties.

With repetitive mode enabled, you can draw many object with the same task.

When a repetitive drawing task is started, it is not stopped after each drawn object until you run another task .

Not all tasks are repetitive, only these marked with a start are (path, free path, image, lines, polyline, polygon, triangle, rectangle, circle, ellipse, circle arc).

This mode is useful when drawing a lot of objects.

See showcase drawing repetitive mode

## 5.3. Drawing tasks

Each drawing task is characterized by:


- Its icon
- The drawn type element
- The number of entered points (1,2,3,..)

Two successive points can be entered in two ways (excepted free path):


- Click the first point keeping the mouse button pressed, move the mouse to the next point and release the button (sequence is mouse down, mouse move and mouse up).
- Click the first point by releasing the mouse button and click on the next point (sequence is two mouse clicks).

Drawing tasks with many points have a floating menu with the two actions remove last point and next to finish the drawing element.


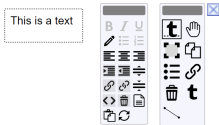

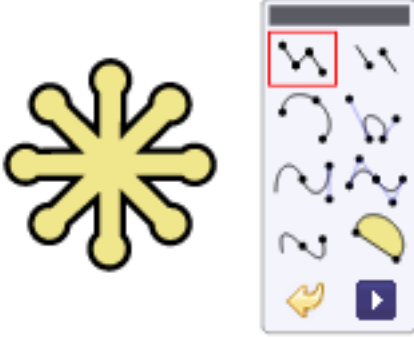

**Table 5.2. floating drawing task menu**

Icon	Action
	remove last point







Icon	Action
	finish drawing element






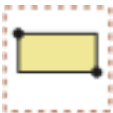
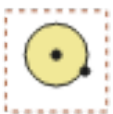



**Table 5.3. Drawing tasks**

Drawing task	Element	Points	Description
Drawing a text 	Text	1	Click the text position and type its content.  See draw simple text , multiline text showcases for an interactive demonstration of how to draw and edit a text.
Drawing a path 	Path	many	Drawing a path with basic commands  The floating drawing task menu is showing the path commands. Select command and enter points then click next to finish the path. See draw path with curves , path with hole showcases for an interactive demonstration of how to draw path.
Drawing a free path 	Path	many	The shape can be drawn in two ways: <ul style="list-style-type: none"> <li>• Click the starting point keeping the mouse button pressed, move the mouse and release the button on the last point.</li> <li>• Click the starting point by releasing the mouse button, move the mouse and click on the last point.</li> </ul> The path can then be simplified by filtering the points using the filter points editor [51] See draw free path showcase for an interactive demonstration.
Drawing an image	Image	2	Draw the image position by its top left and bottom right corners.









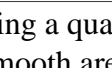


Drawing task	Element	Points	Description
			<p>The svg image element can represent a picture by referencing its HTTP-URL or locally by storing it's base64 encoded data. The default element draws the image of the Mona Lisa by HTTP reference.</p> <p>To change the default content :</p> <ul style="list-style-type: none"> <li>• Replace the href attribute value by another http link by using the properties editor [49]</li> <li>• Replace it by the base64 encoded data of a selected picture file by using the image file load editor [49]</li> <li>• Replace it by the inline base64 encoded data of it's URL by using the load image editor [49]</li> </ul> <p>See draw image showcase for an interactive demonstration of how to draw an image.</p>
<p>Drawing a map</p> 	Image	2	<p>Draw the map position by its top left and bottom right corners.</p> <p>The element is an image with a Google Map image link.</p> <p>The default map location is centered on Paris. To change it use the map editor [46] .</p> <p>The map element offers also the editors of an image element.</p> <p>See draw map showcase for an interactive demonstration of how to insert and edit a Google Map in your SVG drawing.</p>
<p>Drawing an included svg</p> 	Svg	2	<p>Draw the svg position by its top left and bottom right corners.</p> <p>The default content can be replaced by another document loaded from a file or an url .</p> <p>See include svg showcase for an interactive demonstration of how to insert a svg document.</p>
<p>Drawing a line</p> 	Line	2	<p>Draw the line by its start and end points.</p> <p>See draw line showcase for an interactive demonstration of how to draw a line.</p>
<p>Drawing a vertical line</p>	Vertical line	2	<p>Draw the line by its start and end points.</p> <p>See showcase draw horizontal and vertical lines .</p>




Drawing task	Element	Points	Description
			
Drawing a horizontal line 	Horizontal line	2	Draw the line by its start and end points. See showcase draw horizontal and vertical lines .
Drawing a polyline 	Polyline	many	Enter each point of the polyline then click next in the floating menu to finish the polyline.
Drawing a triangle 	Triangle	3	Draw the triangle by its three points. See showcase drawing a triangle .
Drawing a polygon 	Polygon	many	Enter each point of the polygon then click next in the floating menu to finish the polygon.
Drawing a rectangle 	Rectangle	2	Draw the rectangle by its top left and bottom right corners. See draw rectangle showcase for an interactive demonstration of how to draw a rectangle.
Drawing a circle 	Circle	2	Draw the circle by its center and radius points.
Drawing an ellipse 	Ellipse	2	Draw the ellipse by its center and size points.
Drawing a circle arc 	Path	3	Draw the circle arc by its first point, end point and middle point. The middle point defines the radius.
Drawing a cubic arc 	Path	4	Draw the cubic arc by its first point, end point and the two tangents points.



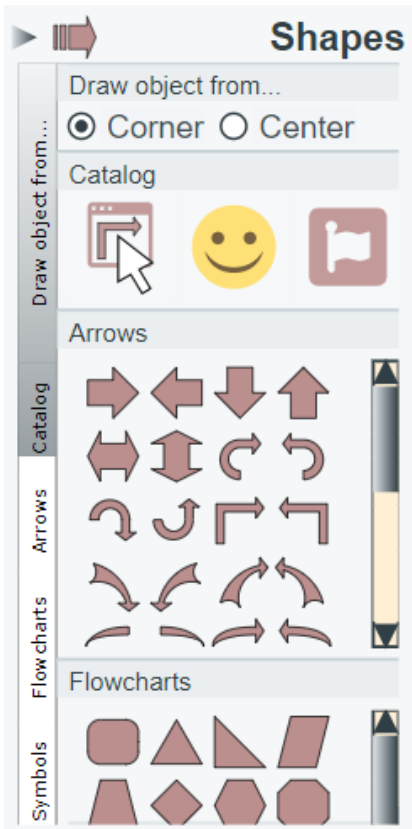
Drawing task	Element	Points	Description
Drawing a quadratic arc 	Path	3	Draw the quadratic arc by its first point, end point and the tangent point.
Drawing a cubic curve 	Path	many	Enter each vertex of the curve and then click Next in the floating menu. The tasks create tangents default then be adjusted with the edit function.
Drawing a cubic area 	Path	many	Drawing a cubic area Enter each vertex of the area and then click Next in the floating menu. The tasks create tangents default then be adjusted with the edit function.
Drawing a quadratic curve 	Path	many	Drawing a quadratic curve Enter each vertex of the curve and then click Next in the floating menu. The tasks create tangents default then be adjusted with the edit function.
Drawing a quadratic area 	Path	many	Drawing a quadratic area Enter each vertex of the area and then click Next in the floating menu. The tasks create tangents default then be adjusted with the edit function.
Drawing a cubic smooth curve 	Path	many	Drawing a cubic smooth curve Enter each vertex of the curve and then click Next in the floating menu. The tasks create tangents default then be adjusted with the edit function.
Drawing a cubic smooth area 	Path	many	Drawing a cubic smooth area Enter each vertex of the area and then click Next in the floating menu. The tasks create tangents default then be adjusted with the edit function.
Drawing a quadratic smooth curve 	Path	many	Drawing a quadratic smooth curve Enter each vertex of the curve and then click Next in the floating menu. The tasks create tangents default then be adjusted with the edit function.
Drawing a quadratic smooth area 	Path	many	Drawing a quadratic smooth area



<b>Drawing task</b>	<b>Element</b>	<b>Points</b>	<b>Description</b>
			Enter each vertex of the area and then click Next in the floating menu. The tasks create tangents default then be adjusted with the edit function.

# Chapter 6. Drawing shapes

## Drawing shapes menu



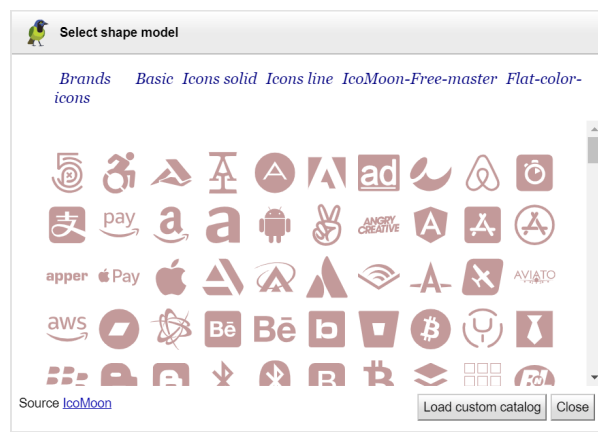
The menu "Drawing shapes" provides ready-made shapes.

Each shape is a path element or a group of paths.

A shape can be defined by its center or a corner (upper left/right or bottom left/right). Its size can preserve the model's ratio.

The shape model can be selected from the three sets (arrows, flowcharts, symbols) of the menu or from the drawsvg catalog, emoji, font Awesome icon choosers.

See draw shapes , emoji , font Awesome icon showcases for an interactive demonstrations.



**Figure 6.1. Shape full catalog**

The shape catalog provide a lot of models from IcoMoon .and fontawesome Custom catalog can be added with custom shape catalog tool [131] .

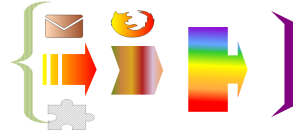
## How to draw a shape





### To draw a **shape**

- set drawing mode (corner or center, keep ratio or not)
- select it's model
- draw it with two points
- then defines its style properties



# Chapter 7. Style properties

This chapter describes the style properties supported by Draw SVG.

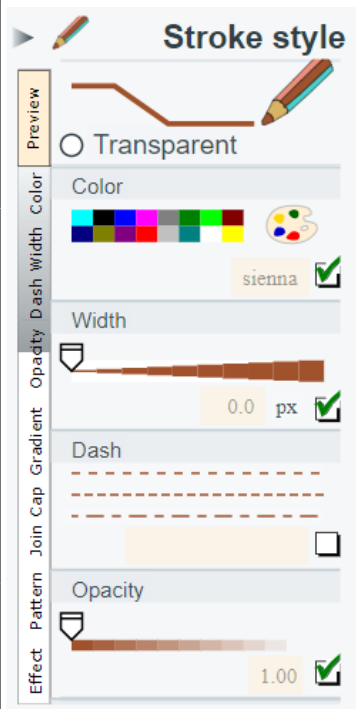

Style properties are classified into four categories (stroke, fill, text, markers) with a dedicated menu.

Each menu shows the rendering properties with an illustration.

## 7.1. Stroke style properties

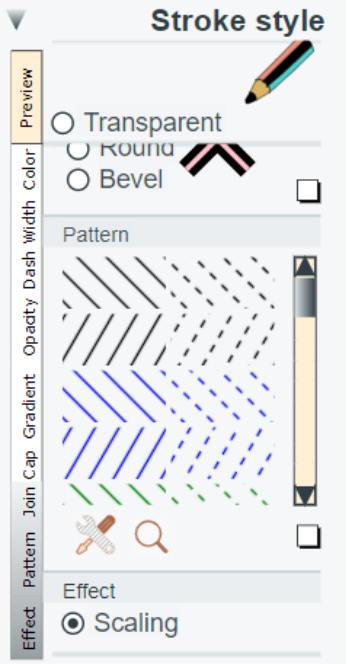


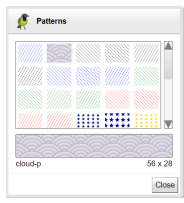
These properties are applicable to all geometric shapes as well as text.

**Table 7.1. Stroke style properties**

Stroke style menu	Property	Description
 <p>transparent, color, scaling, width, dash, opacity</p>	transparent	Controls the appearance of lines, visible or transparent. Sets the stroke property to none ( <code>stroke="none"</code> ). <a href="https://www.w3.org/TR/SVG/painting.html#StrokeProperty">https://www.w3.org/TR/SVG/painting.html#StrokeProperty</a>
	color	Line solid color. Sets the stroke property with a color ( <code>stroke="color"</code> ). The color can be selected from the basic palette, entered in the text field, or set with the color chooser svg panel [75] available from its icon  . <a href="https://www.w3.org/TR/SVG/painting.html#StrokeProperty">https://www.w3.org/TR/SVG/painting.html#StrokeProperty</a>
	scaling	Controls the appearance of the line thickness. If this option is enabled, the drawing of line thicknesses is sensitive to changes in the view as defined in the SVG1.1. Off, "vector-effect" SVG1.2 property is set to "non-scaling-stroke". Drawing the line thickness is constant regardless of the zoom level. <a href="https://www.w3.org/TR/SVGTiny12/painting.html#NonScalingStroke">https://www.w3.org/TR/SVGTiny12/painting.html#NonScalingStroke</a>
	width	Sets the line thickness with his unit. <ul style="list-style-type: none"> <li>• xy The line width is defined in the svg coordinate space</li> <li>• css units px, pt, pc, in, cm, mm</li> </ul> <a href="https://www.w3.org/TR/SVG/painting.html#StrokeWidthProperty">https://www.w3.org/TR/SVG/painting.html#StrokeWidthProperty</a>
	dash	Specifies the pattern of dashes.



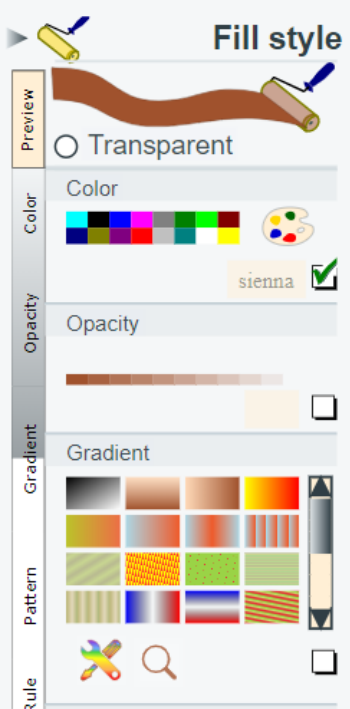

Stroke style menu	Property	Description
		<p>A list of lengths of alternating dashes and gaps. If an odd number of values is provided, then the list of values is repeated to yield an even number of values. Thus, 5,3,2 is equivalent to 5,3,2,5,3,2.</p> <p><a href="https://www.w3.org/TR/SVG/painting.html#StrokeDasharrayProperty">https://www.w3.org/TR/SVG/painting.html#StrokeDasharrayProperty</a></p>
	opacity	<p>Line opacity between 0 (transparent) and 1 (solid).</p> <p><a href="https://www.w3.org/TR/SVG/painting.html#StrokeOpacityProperty">https://www.w3.org/TR/SVG/painting.html#StrokeOpacityProperty</a></p>
 <p>gradient, cap, join</p>	gradient	<p>Gradient applied to the fill line in place of the solid color. Sets the stroke property with a gradient id ( <code>stroke="url(#MyGradient)"</code> ). The gradient can be customized with the gradient definition [76] tool  .</p> <p>The gradient can be selected from the drawn list or with the chooser tool  .</p>  <p><a href="https://www.w3.org/TR/SVG/pservers.html#Gradients">https://www.w3.org/TR/SVG/pservers.html#Gradients</a></p>
	cap	<p>Defines how the ends of an line is rendered.</p> <p><a href="https://www.w3.org/TR/SVG/painting.html#StrokeLinecapProperty">https://www.w3.org/TR/SVG/painting.html#StrokeLinecapProperty</a></p>
	join	<p>Defines how the join between two lines in a shape is rendered.</p> <p><a href="https://www.w3.org/TR/SVG/painting.html#StrokeLinejoinProperty">https://www.w3.org/TR/SVG/painting.html#StrokeLinejoinProperty</a></p>

Stroke style menu	Property	Description
 <p>pattern</p>	<p>pattern</p> <p>Pattern applied to the fill line in place of the solid color. Sets the stroke property with a pattern id ( <code>stroke="url(#MyPattern)"</code> ).</p> <p>The pattern can be customized with the pattern definition [79] tool  .</p> <p>The pattern can be selected from the drawn list or with the chooser tool  .</p>  <p><a href="https://www.w3.org/TR/SVG/pservers.html#Patterns">https://www.w3.org/TR/SVG/pservers.html#Patterns</a></p>	





## 7.2. Fill style properties

These properties are applicable to closed geometric shapes as well as text.

**Table 7.2. Fill style properties**

Menu	Property	Description
	<p>transparent Controls the appearance of filling the form. Sets the fill property to none ( <code>fill="none"</code> ). <a href="https://www.w3.org/TR/SVG/painting.html#FillProperty">https://www.w3.org/TR/SVG/painting.html#FillProperty</a></p> <p>color Fill solid color. Sets the fill property with a color ( <code>fill="color"</code> )..</p> <p>The color can be selected from the basic palette, entered in the text field, or set with the color chooser svg panel [75] available from its icon  .</p> <p><a href="https://www.w3.org/TR/SVG/painting.html#FillProperty">https://www.w3.org/TR/SVG/painting.html#FillProperty</a></p> <p>Opacity Fill form opacity between 0 (transparent) and 1 (solid).</p> <p><a href="https://www.w3.org/TR/SVG/painting.html#FillOpacityProperty">https://www.w3.org/TR/SVG/painting.html#FillOpacityProperty</a></p>	



Menu	Property	Description
transparent, color, opacity, gradient.	gradient	<p>Gradient applied to the fill form in place of the solid color. Sets the fill property with a gradient id ( <code>fill="url(#MyGradient)"</code> ).</p> <p>The gradient can be customized and selected with the gradient definition [76] tool  .</p> <p>The gradient can be selected with the chooser tool  .</p> <div data-bbox="1007 645 1187 837" data-label="Image"> </div> <p>There are two types of gradients: linear and radial .</p>
<div data-bbox="193 958 549 1630" data-label="Image"> </div> <p>rule</p>	pattern	<p>Pattern applied to the fill form in place of the solid color. Sets the fill property with a pattern id ( <code>fill="url(#MyPattern)"</code> ).</p> <p>The pattern can be customized and selected with the pattern definition [79] tool  .</p> <p>The pattern can be selected from the drawn list or with the chooser tool  .</p> <div data-bbox="1007 1312 1187 1505" data-label="Image"> </div>
	rule	<p>Defines the filling algorithm.</p> <p>The method <b>evenodd</b> is useful for surfaces with holes.</p> <p><a href="https://www.w3.org/TR/SVG/painting.html#FillRuleProperty">https://www.w3.org/TR/SVG/painting.html#FillRuleProperty</a></p>

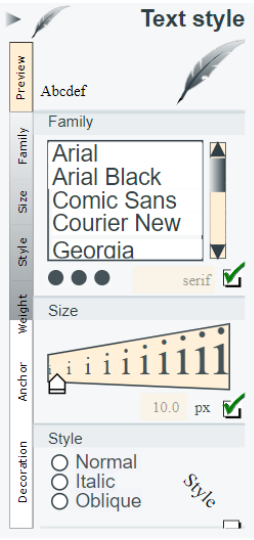
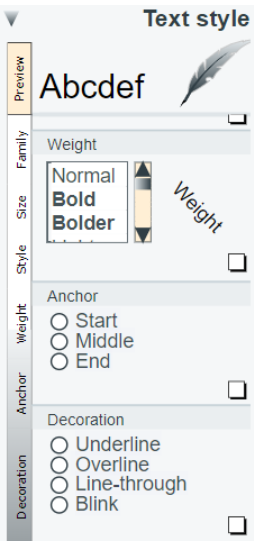
### 7.3. Text style properties

These properties are only applicable to texts.

See draw simple text , multiline text , text editor showcases for interactive demonstrations.



**Table 7.3. Text style properties**

Menu	Property	Description
 <p>family, size, style</p>	family	Select font family. <a href="https://www.w3.org/TR/SVG/text.html#FontFamilyProperty">https://www.w3.org/TR/SVG/text.html#FontFamilyProperty</a>
	size	Defines font size. <a href="https://www.w3.org/TR/SVG/text.html#FontSizeProperty">https://www.w3.org/TR/SVG/text.html#FontSizeProperty</a>
	style	Defines font style. <a href="https://www.w3.org/TR/SVG/text.html#FontStyleProperty">https://www.w3.org/TR/SVG/text.html#FontStyleProperty</a>
	weight	Defines font weight. <a href="https://www.w3.org/TR/SVG/text.html#FontWeightProperty">https://www.w3.org/TR/SVG/text.html#FontWeightProperty</a>
	anchor	Defines the text anchor. <a href="https://www.w3.org/TR/SVG/text.html#TextAnchorProperty">https://www.w3.org/TR/SVG/text.html#TextAnchorProperty</a>
	decoration	This property describes decorations that are added to the text of an element. <a href="https://www.w3.org/TR/SVG/text.html#TextDecorationProperty">https://www.w3.org/TR/SVG/text.html#TextDecorationProperty</a>
 <p>weight, anchor, decoration</p>		



**Note**

Fill and stroke properties are also applicable to text.

**7.4. Marker style properties**

These properties are only applicable to forms with support points (line, polyline, polygon, path). The ‘ marker ’ element defines the graphics that is to be used for drawing arrowheads or polymarkers on a given ‘path’, ‘line’, ‘polyline’ or ‘polygon’ element.

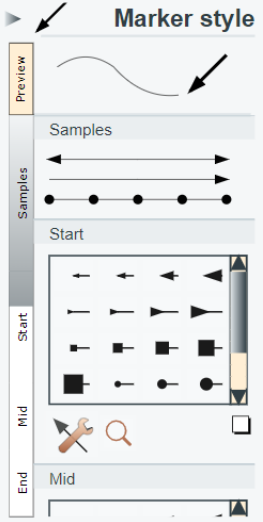


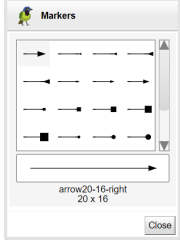
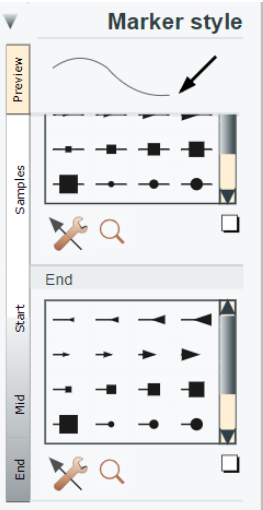


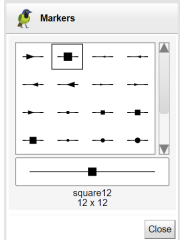

Each marker can be customized with the maker dialog panel [78] launched from it's icon




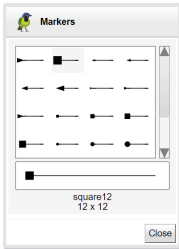
See creating , using markers showcases for interactive demonstrations.



**Table 7.4. Marker style properties**


Menu	Property	Description
 <p>samples, end.</p>	<p>samples Offers the most useful markers.</p> <p>end Selects the marker to represent the end of the curve.                      The gradient can be customized and selected with the marker definition [78] tool  .                      The marker can be selected from the drawn list or with the chooser tool  .</p>  <p><a href="https://www.w3.org/TR/SVG/painting.html#MarkerEndProperty">https://www.w3.org/TR/SVG/painting.html#MarkerEndProperty</a></p>	
 <p>mid, start</p>	<p>mid Selects the marker to represent each intermediate point of the curve.                      The gradient can be customized and selected with the marker definition [78] tool  .                      The marker can be selected from the drawn list or with the chooser tool  .</p>  <p><a href="https://www.w3.org/TR/SVG/painting.html#MarkerMidProperty">https://www.w3.org/TR/SVG/painting.html#MarkerMidProperty</a></p>	
	<p>start Selects the marker to represent the origin of the curve.                      The gradient can be customized and selected with the marker definition [78] tool  .</p>	



Menu	Property	Description
		<p>The marker can be selected from the drawn list or with the chooser tool .</p>  <p><a href="https://www.w3.org/TR/SVG/painting.html#MarkerStartProperty">https://www.w3.org/TR/SVG/painting.html#MarkerStartProperty</a></p>

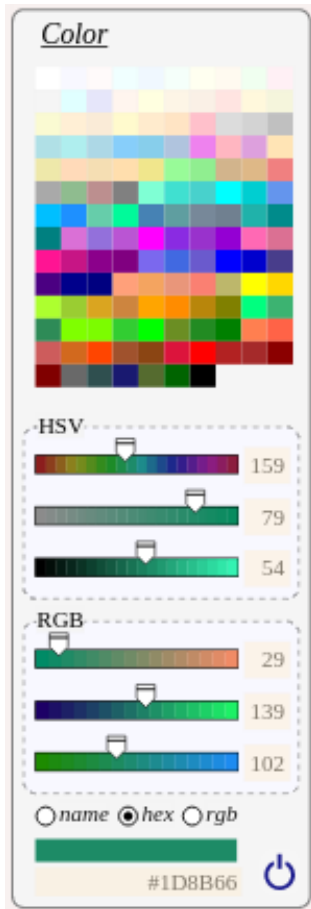
## 7.5. Style panels

### 7.5.1. Color chooser panel

The color chooser svg panel is launched from it's icon  in the menu and is displayed above. It can be also displayed from other tools such as marker definition. [78]  
See showcase using color chooser



**Table 7.5. Color chooser panel**



the color chooser panel.

The color can be selected from the extended palette or defined by its HSV/RGB components.

Each HSV/RGB component can be defined with the slider or entered within the text field.

The color value has 3 available formats :

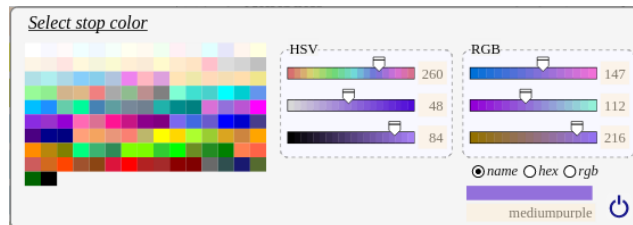
- the name from the palette
- hexa format
- rgb format

To select the value, click on the renderer rectangle.


The value can be defined with it's text field.

To cancel the action, click on the close icon.

The color chooser can be displayed with a horizontal alignment.



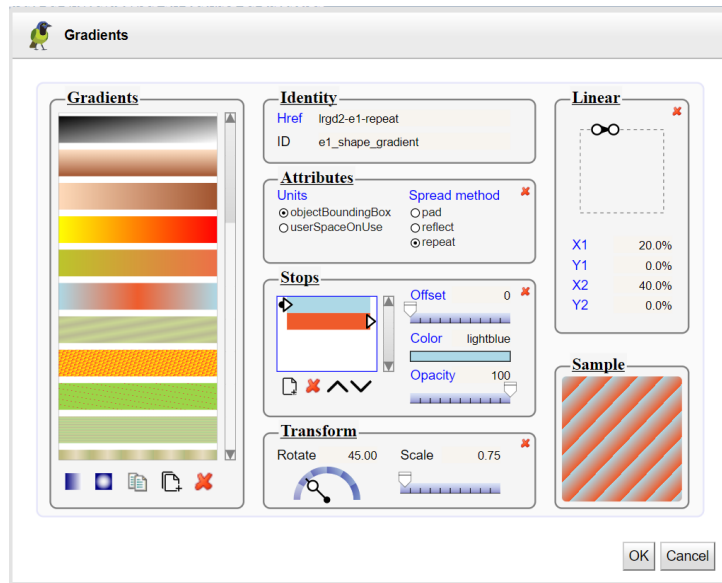
## 7.5.2. Gradient dialog panel

The gradient dialog panel is launched from it's icon  in the menu and is displayed in grab mode.

Gradients consist of continuously smooth color transitions along a vector from one color to another, possibly followed by additional transitions along the same vector to other colors.

There are two types of gradients : linear and radial .

See creating , using gradients showcases for interactive demonstrations.



**Figure 7.1. Gradient dialog panel**


The panel allows to edit a gradient definition, add a new one radial or linear, copy an existing, or extends another to customize it.

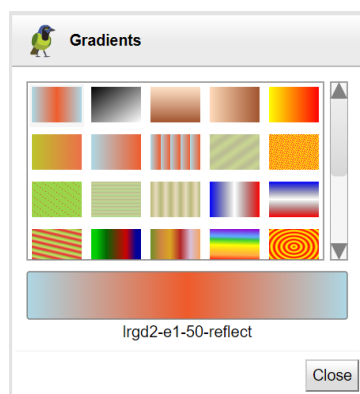
Select a gradient in the list, then change properties :

- Units, if you select userSpaceOnUse then the drawing of control points will be not available because these must be expressed as drawing coordinates and not as a percentage.
- Spread mode with the toggle group,
- Linear or Radial definition with the control points,
- Add stop elements, remove or change order,
- Move stop offsets with the sliders,
- Choose stop color and change its opacity with the slider.
- Apply a transformation to the gradient

The sample box show the gradient with the current scale of the drawing.

All changes are applied with the **apply** button and the current gradient is applied to the style property (stroke or fill) of the selected object.

To directly apply a gradient, use the selection tool .




**Figure 7.2. Gradient chooser**

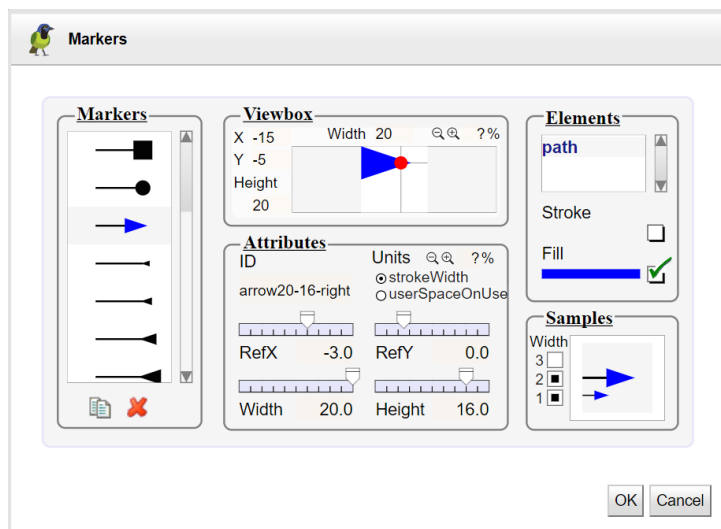
The selected gradient is drawn with the current scale of the drawing.

### 7.5.3. Marker dialog panel

The 'marker' element defines the graphics that is to be used for drawing arrowheads or polymarkers on a given 'path', 'line', 'polyline' or 'polygon' element.

The marker dialog panel is launched from its icon  in the menu and is displayed in grab mode.

See creating , using markers showcases for interactive demonstrations.



**Figure 7.3. Markers tool**

The panel allows to edit/copy/delete a marker definition,

To add a new one, drawn it, select elements and use the action marker definition [44] from the selection menu.

Select a marker in the list, then change attributes :


- **ViewBox** : Use text fields ( **x** , **y** , **width** , **height** ), icons (zoom/in zoom/out) and text field ratio for extending or reducing the area. To add a margin of 20% type 120<enter> in the ratio field.
- **Id** : the marker's id.
- **Units** : Defines the coordinate system for the size (width,height) and the contents of the 'marker'. If markerUnits equals " **strokeWidth** ", the size and the contents of the 'marker' represent values in a coordinate system which has a single unit equal the size in user units of the current stroke width in place for the graphic object referencing the marker. If markerUnits=" **userSpaceOnUse** ", the size and the contents of the 'marker' represent values in the current user coordinate system in place for the graphic object referencing the marker. The default units is strokeWidth'.
- **Size** : Define the dimensions of the marker relative to specified units. Use text fields ( **width** , **height** ), icons (zoom/in zoom/out) and text field ratio for extending or reducing the size of the tile.
- **Ref point** : change the coordinates of the reference point, move it's red circle or use **refX** / **refY** sliders and fields.

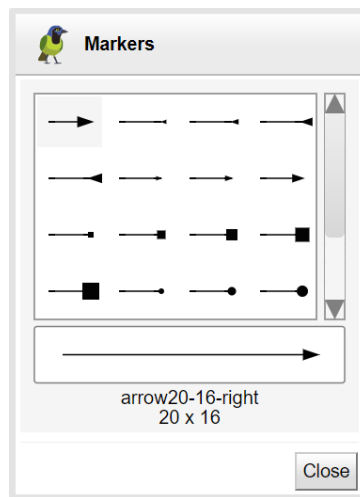


- **Elements** : List of all the graphic elements of the marker with their stroke and fill colors.

To change the colors (stroke or fill) of an element of the marker, select it in the list and click on the renderer rectangle of the color to change it with the color chooser [75] or remove it with the check-box. .

Finally the **samples** box show the renderer of the selected marker with different stroke widths. All changes are applied with the **apply** button and the current marker is applied to the style property (marker-start, marker-mid, marker-end) of the selected object.

To directly apply a marker, use the selection tool .



**Figure 7.4. Marker chooser**

The selected marker is drawn with the current scale of the drawing and style property (marker-start, marker-mid, marker-end).

### 7.5.4. Pattern dialog panel

The ' pattern ' element defines the graphic that is to be used to fill a shape by replicating an object.

It performs two key functions: hatching and wallpaper.

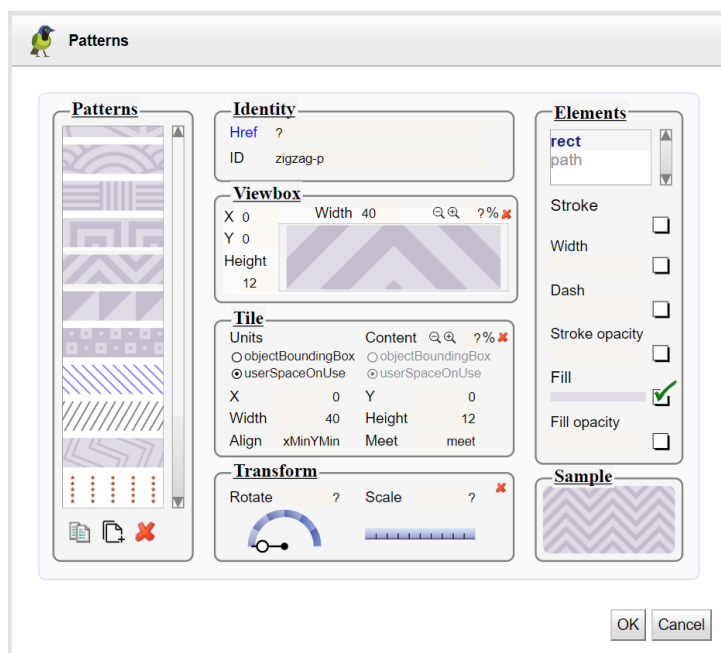
These function uses two different units ( **userSpaceOnUse** , **objectBoundingBox** ).

A pattern is defined either in the coordinate system of the drawing (userSpaceOnUse) or in proportion to the bounding box of the object to fill (objectBoundingBox).

A pattern may be defined by **extension** of another. The most frequent case is the definition of a pattern by applying a transformation to another basic pattern.

The pattern dialog panel is launched from it's icon  in the fill menu and is displayed in grab mode.

See creating , using patterns showcases for interactive demonstrations.



**Figure 7.5. Patterns tool**

The panel allows to edit/copy/extends/delete a pattern definition,

To add a new one, draw it's elements, select them and use the action pattern definition [45] from the selection menu.


Select a pattern in the list, then change attributes :

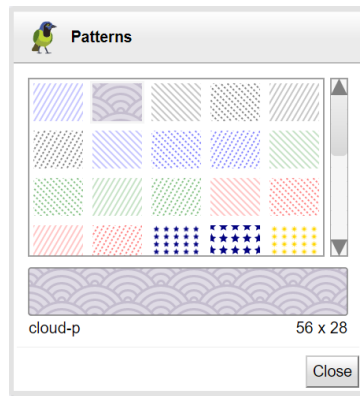
- **Identity** : the inherited pattern's id (href) and the pattern's id. The Inherited properties are drawn in blue color.
- **ViewBox** : Use text fields (x,y,width, height), icons (zoom/in zoom/out) and text field ratio for extending or reducing the area. To add a margin of 20% type 120<enter> in the ratio field.
- **Tile** : Define the dimensions and units of the filling tile. Use text fields (x,y,width, height), icons (zoom/in zoom/out) and text field ratio for extending or reducing the size of the tile.
- **Transform** : Rotating and scaling the tile.
- **Elements** : List of all the graphic elements of the pattern with their style editable properties (stroke, stroke-width, stroke-opacity, stroke-dash, fill, fill-opacity).

To change the colors of an element inside a pattern, click on the element inside the viewBox or select it from the list and click on the render rectangle of the color to change it with the color chooser [75] or remove it with the check-box.

Defining a pattern by **transformation** of another. Select the basic pattern and click on the extension icon. Set it's id and apply the desired transformation.

Finally the **Sample** box show the renderer of the selected pattern with the scale of the drawing. All changes are applied with the **apply** button and the current pattern is applied to the style property (stroke or fill) of the selected object.

To directly apply a gradient, use the selection tool .



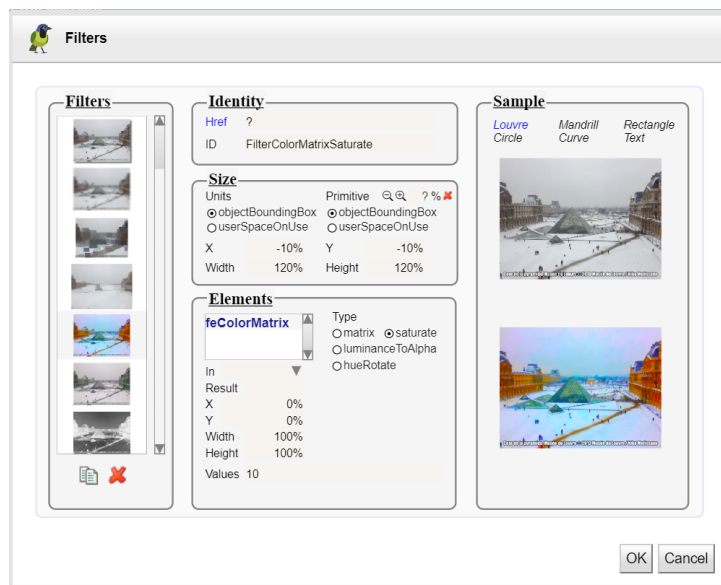
**Figure 7.6. Pattern chooser**

The selected pattern is drawn with the current scale of the drawing.

### 7.5.5. Filter dialog panel

A 'filter' consists of a series of graphics operations that are applied to a given source graphic to produce a modified graphical result.

See showcase using filters



**Figure 7.7. Filter tool**

The panel allows to edit/copy/delete a filter definition,






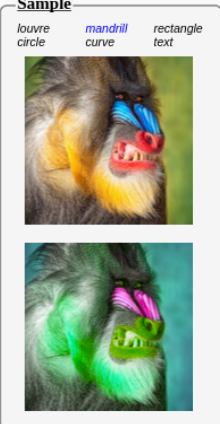
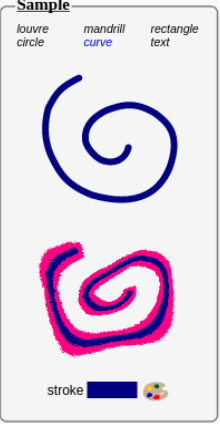


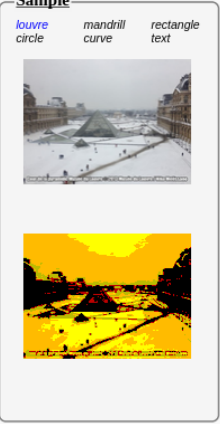
Select a filter in the list, then change attributes :

- **Size** : Use text fields ( **x** , **y** , **width** , **height** ), icons (zoom/in zoom/out) and text field ratio for extending or reducing the area. To add a margin of 20% type 120<enter> in the ratio field.
- **Id** : the filter's id.
- **Units** : Defines the coordinate system for the size (width,height) and the contents of the 'filter'.
- **Elements** : List of all operations of the filter.
- **Sample** : The rendering of the filter applied to 6 different graphical objects (picture of the pyramid of the Louvre, Mandrill monkey, rectangle, circle, curve, text). .



All changes are applied with the **apply** button and the current filter is applied to the filter property of the selected object.

**Table 7.6. Examples of filters**

Relief	ColorMatrix	Turbulence	Transfert
<p><b>Sample</b></p> <p>louvre mandrill rectangle circle curve text</p>  <p>stroke  </p> <p>fill  </p>	<p><b>Sample</b></p> <p>louvre mandrill rectangle circle curve text</p> 	<p><b>Sample</b></p> <p>louvre mandrill rectangle circle curve text</p>  <p>stroke  </p>	<p><b>Sample</b></p> <p>louvre mandrill rectangle circle curve text</p> 

# Chapter 8. Apply constraints

## 8.1. Constraints introduction

DRAWSVG includes a constraints manager.

Constraints are defined in the document (with defs elements) and applied to the elements (with drawsvg:constraints attribute).

```
<svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" width="880px" height="450px" viewBox="0 0 880 450"
  preserveAspectRatio="xMidYMid meet" xmlns:drawsvg="http://www.drawsvg.org">
  <defs id="drawsvgConstraints">
    <defs id="constraint1" drawsvg:constraint="{ 'type':'connectPoints', 'ids':['e2_pathH','e7_line'], 'priority':0, 'index-p1':2, 'index-p2':0}"/>
    <defs id="constraint2" drawsvg:constraint="{ 'type':'connectPoints', 'ids':['e4_pathH','e7_line'], 'priority':0, 'index-p1':0, 'index-p2':1}"/>
  </defs>
  <path d="M105,210l61-75l42,79Z" style="fill:rosybrown;stroke:black;stroke-width:1px;" id="e2_pathH"
    drawsvg:constraints="['constraint1]" transform="matrix(0.707107 -0.707107 0.707107 0.707107 -77.5523 161.772)"/>
  <path d="M327,165l72,78l49,-117Z" style="fill:rosybrown;stroke:black;stroke-width:1px;" id="e4_pathH"
    drawsvg:constraints="['constraint2]"/>
  <line id="e7_line" x1="220.84671708797583" y1="166.01471862576142" x2="327" y2="165" style="stroke:black;fill:none;stroke-width:1px;"
    drawsvg:constraints="['constraint1','constraint2]"/>
</svg>
```

**Figure 8.1. Constraints definition**

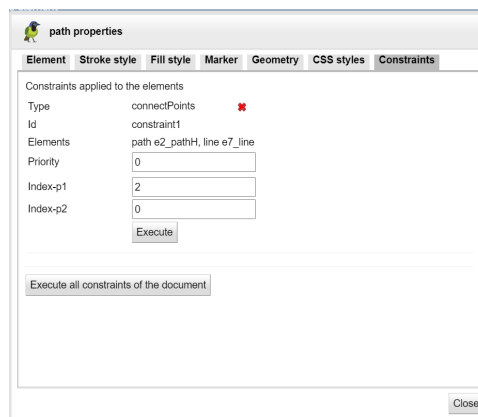
The constraints are executed automatically on modification of the associated elements and propagated on the impacted elements.

The management of constraints is transparent for the user.

Constraints are typed and have modifiable properties.

Constraints properties can be edited and removed with the selected element properties dialog and its constraints tab panel.

The execution of constraints can be started explicitly.



**Figure 8.2. Constraints tab panel**

Constraints have a priority property that can be specified to control the order of execution on an element.

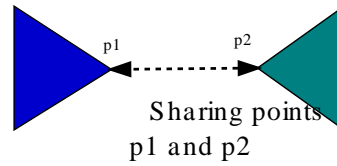
## 8.2. Connect points constraint

This type of constraints allows to share points between elements.

When transforming an object that share points with this type of constraint, the transformation applied to the points will be propagated to the associated elements.







**Figure 8.3. Connect points**

A connect points constraint refers two elements and the indices (index-p1,index-p2) of the shared point for each element.

```
<defs id="drawsvgConstraints">
<defs id="constraint1" drawsvg:constraint="{ 'type':'connectPoints', 'ids':
['e2_pathH','e7_line'], 'priority':0, 'index-p1':2, 'index-p2':0}"/>
<defs id="constraint2" drawsvg:constraint="{ 'type':'connectPoints', 'ids':
['e4_pathH','e7_line'], 'priority':0, 'index-p1':0, 'index-p2':1}"/>
</defs>
```

The constraint is defined when drawing elements with point capture, or when moving and merging points.

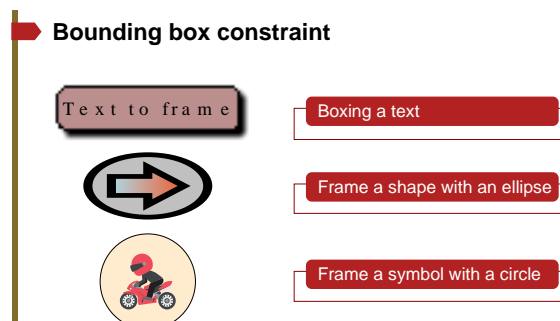
See showcases constraint connected points , merging point , draw with point capture ,

### 8.3. Bounding box constraint

This type of constraint apply the bounding box of an element to another of type (rectangle, ellipse, circle).

When transforming the element its linked bounding box element is updated.

Conversely, the modification of the bounding box element is carried over to the linked element so as to keep the center and force the dimensions to ensure the inclusion of the element.



**Figure 8.4. Bounding box constraint**



A bounding box constraint refers two elements (the bounding box element and the target), a margin property can be set.

```
<defs id="drawsvgConstraints">
  <defs id="constraint1" drawsvg:constraint="{ 'type':'alignRectToBBox', 'ids':
['rect1','el_texte'], 'priority':0, 'margin':20}"/>
</defs>
```

Creating the bounding box of a text is available from the floating menu with "add bounding box" action.

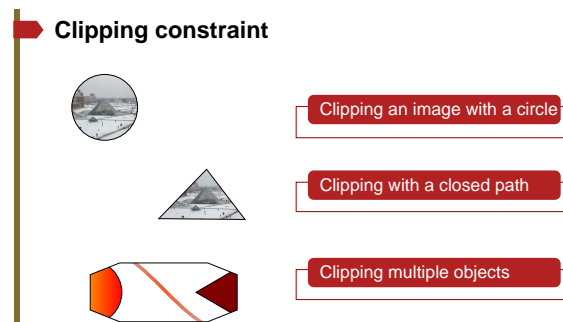
Otherwise the bounding box element (rectangle, ellipse, circle) must first be created then selected with the target element to apply the constraint with the action of the selection / constraints menu.

See showcase bounding box constraint .

## 8.4. Clipping constraint

This type of constraint define a clip path with an element (rectangle, ellipse, circle, polygon, closed path) and apply it to linked elements.

When transforming the element the definition of the clip path is updated.



**Figure 8.5. Clipping constraint**

The clipping constraint has one defs element to define the clip path and one defs element for each clipped element to apply it.

```
<defs id="drawsvgConstraints">
  <defs id="constraint1" drawsvg:constraint="{ 'type':'clipPathDef', 'ids':
['e2_circle'], 'priority':0}">
    <clipPath id="clipPath-constraint1" drawsvg:constraints="['constraint2']">
      <circle cx="174.655" cy="185.337" r="55.9534"/>
    </clipPath>
  </defs>
```

```
<defs id="constraint2" drawsvg:constraint="{ 'type':'clipPathApply', 'ids':  
['clipPath-constraint1','el_image'], 'priority':0}"/>  
</defs>
```

To define a clipping constraint: select first the clip definition element (rectangle, ellipse, circle, polygon, closed path) then the target element to apply the constraint with the action of the selection / constraints menu.

See showcase [Clipping constraint](#) .



---

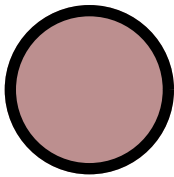
# Chapter 9. Animations

## 9.1. Introduction

After some debate, standard **SMIL** animations are now fully supported by common browsers (Chrome, Edge, FireFox, Safari, Opera).

SMIL animations have the advantage of being able to cover **SVG DOM attributes** like style properties whereas CSS animations are limited to style properties.

The sample below animates the circle radius and its stroke-dashoffset style property.



See the online sample (for the pdf version of the document that don't show the animation in action).

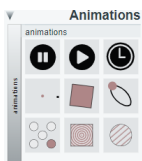
SVG SMIL animations are available as 4 common elements:

- The **animate** element used to animate a single attribute or property over time.
- The **animateMotion** element to move a referenced element along a motion path.
- The **animateTransform** element to transform a target element, thereby allowing animations to control translation, scaling, rotation and/or skewing.
- The **set** element provides a simple means of just setting the value of an attribute for a specified duration. It is useful to fix the state of an element during animations.

SMIL animations are integrated within drawsvg editor since release **10.3** (2022 november).

## 9.2. The animations menu

The animations menu has 7 functions (available with expert profile [1])



- **Pause:**  
Pause all animations.
- **Play:**  
Unpause all animations.
- **Scheduler** [88]:  
View animations planning.
- **Attribute** [94]:  
Animate attribute, add an animate element on the selected object, edit all animate elements of the document.
- **Transform** [97] :  
Animate an element by applying a transformation (translate, scale, rotate,..) to it.
- **Motion** [101]:  
Move an element along a path.



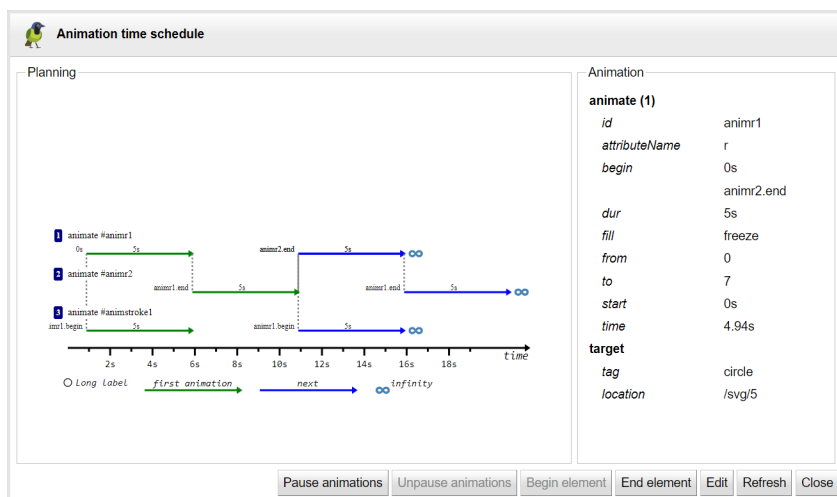
- **Set** [105]:  
Set the value of an attribute for a specified duration without transition.
- **Gradients** [108]:  
Animate a gradient properties and its stop color elements. Transform the gradient (scale, rotate, translate).
- **Patterns** [109]:  
Animate a pattern properties and its content elements. Transform the pattern (scale, rotate, translate).

### 9.3. The animations schedule

Task icon



- This feature shows an overview dialog of all animations with two parts:
- A drawing of each animation in time with animated time intervals and sequences of events.
  - Each animation line is selectable on its label.
  - A panel with the description of the selected animation element.



The selected animation can be started, stopped and edited.  
See animations schedule showcase for an interactive demonstration.

### 9.4. Animation timing properties

They are common to all animation elements and control the timing of the animation, including what causes the animation to start and end, whether the animation runs repeatedly, and whether to retain the end state the animation once the animation ends.

Each property has a dedicated panel that explains its behavior with links to standard specifications, and presents a simple and intuitive user interface for entering the property's value (begin [88], duration [89], end [89], min [90], max [90], restart [90], repeat count [91], repeat duration [91], fill [91]).

#### The animate begin property

The begin property define when the animate element should start.



**Begin**

Defines when the element should begin (i.e. become active).  
Set the begin property as:

- 0s** , starts at the beginning of the document (default)
- indefinite**, the begin of the animation will be determined by a "beginElement()" method call, or a hyperlink targeted to the element. see [indefinite value hyperlinks and timing](#)
- list** , one or more elements of :
  - **offset** defines a time count from the time begin of the document. see [offset value](#)
  - **syncbase** defines a time count from another animation. see [syncbase value](#)
  - **event** defines a time count from an event. see [event value](#)
  - **repeat** defines a time count from a raised animation repeat event. see [repeat value](#)

event        ms ✖

Note 'accessKey' is ignored because not supported on Chrome, Edge and FireFox navigators

Define it as default (0s starts on document load) or indefinite, or as a list of :

- time **offset** from the document load.
- **syncbase** item to begin the animation when another ends or begins.  
Select the animate element, the event type begin or end and define the time offset.

syncbase        ms ✖

- **repeat** item to begin the animation when another repeats.  
Select the animate element, define the repeat count and the time offset.
- **event** item to begin the animation when the user click on an element.  
Select the element from the drawn list (by default the target element), the event type (click,...) and define the time offset.

event        ms ✖

The synchronization can be controlled with the animations schedule [88] dialog.  
See synchronization by events showcase for an interactive demonstration.  
See specifications.

## The animate duration property

Specifies the simple duration of the animation (indefinite, a time count with unit, a full or partial clock duration).

**Duration**

Specifies the simple duration of the animation as :

- indefinite** (default)  
Note that interpolation will not work if the simple duration is indefinite
- a time count with unit and fraction**  
   ms
- a full** clock duration in hours, minutes, seconds, milliseconds
- a partial** clock duration in minutes, seconds, milliseconds.

see [duration details clock value syntax](#)

See specifications.

## The animate end property

Similar to the begin property, the end property define when the animate element should stop.



## End

Defines an end value for the animation that can constrain the active duration.

- none** (default)
- indefinite**, the end of the animation will be determined by a "endElement()" method call.
- list**, one or more elements of:
  - **offset** defines a time count from the time begin of the document. see [offset value](#)
  - **syncbase** defines a time count from another animation. see [syncbase value](#)
  - **event** defines a time count from an event. see [event value](#)
  - **repeat** defines a time count from a raised animation repeat event. see [repeat value](#)

*If both ('end', 'repeatCount', 'repeatDur') are specified, the active duration is defined as the minimum of them.*

see [details computing active duration](#)

See specifications.

## The animate min property

Similar to the duration property, the min property specify the minimum value of the active duration (including repetitions).

### Min duration

Specifies the length of the minimum of the active duration, measured in local time as :

- 0s**, this does not constrain the active duration at all (default).

a time **count** with unit and fraction

a **full** clock duration in hours, minutes, seconds, milliseconds

a **partial** clock duration in minutes, seconds, milliseconds.

see [min attribute details clock value syntax](#)

See specifications.

## The animate max property

Similar to the duration property, the max property specify the maximum value of the active duration (including repetitions).

### Max duration

Specifies the length of the maximum of the active duration, measured in local time as :

- none**, this does not constrain the active duration at all (default).

a time **count** with unit and fraction

a **full** clock duration in hours, minutes, seconds, milliseconds

a **partial** clock duration in minutes, seconds, milliseconds.

see [min attribute details clock value syntax](#)

See specifications.

## The animate restart property

Specify the restart mode of the animation.



## Restart

Specifies the restart mode for the animation.

**always**

The animation can be restarted at any time. This is the default value.

**whenNotActive**

The animation can only be restarted when it is not active (i.e. after the active end).

Attempts to restart the animation during its active duration are ignored.

**never**

The element cannot be restarted for the remainder of the current simple duration of the parent time container.

see [details](#)

See specifications.

## The animate repeat count property

Specify the number of repetitions of the animation.

### Number of repeating

Specifies the number of repetitions ('repeatCount') of the animation as:

**none**, the number of repetitions is undefined (default)

**indefinite**, the animation is defined to repeat indefinitely (i.e. until the document ends)

**number**

This is a numeric value that specifies the number of repetitions.

It can include partial repetitions expressed as fraction values.

*The animation can be repetitive by specify either how many times to repeat, using 'repeatCount', and/or how long to repeat using 'repeatDur'. If both are specified, the active duration is defined as the minimum of them.*

see [repeat count details](#) [computing active duration](#)

See specifications.

## The animate repeat duration property

Specify the total duration for repetitions of the animation.

### Repeat duration

Specifies the total duration ('repeatDur') for repeat as :

**none**, the total duration is not set (default)

**indefinite**, the animation is defined to repeat indefinitely (i.e. until the document ends)

a time **count** with unit and fraction

a **full** clock duration in hours, minutes, seconds, milliseconds

a **partial** clock duration in minutes, seconds, milliseconds.

*The animation can be repetitive by specify either how many times to repeat, using 'repeatCount', and/or how long to repeat using 'repeatDur'. If both are specified, the active duration is defined as the minimum of them.*

see [repeat duration details](#) [computing active duration](#)

See specifications.

## The animate fill property

Specify the visual effect when the animation ends.





## Fill mode

Specifies the visual effect when animation ends as:

**remove**

The animation effect is removed (no longer applied) when the active duration of the animation is over.

After the active end of the animation, the animation no longer affects the target.

This is the default.

**freeze**

The animation effect  $F(t)$  is defined to freeze the effect value at the last value of the active duration.

The animation effect is "frozen" for the remainder of the document duration.

see [details](#)

See specifications.

## 9.5. Animation interpolation properties

They are common to animate ( attribute [94], transform [97], motion [101]) elements and control the method to compute the value along the simple duration of the animation.

Each property has a dedicated panel that explains its behavior with links to standard specifications, and presents a simple and intuitive user interface for entering the property's value ( calcMode [92], keyTimes [92], keySplines [93] ).

### The animate calcMode property

The calcMode property define the interpolation function between :

- values [95] of the animated attribute [95] of a animate [94] element.
- values [95] of the transformation [99] of a animate transform [97] element.
- positions of the target element along the the motion path [103] of a animate motion [101] element.

## CalcMode

Specifies the interpolation mode for the animation as :

**discrete**

This specifies that the animation function will jump from one value to the next without any interpolation.

**linear**

Simple linear interpolation between values is used to calculate the animation function.

This is the default mode if the attribute supports linear interpolation (not string).

**paced**

Defines interpolation to produce an even pace of change across the animation.

This is only supported for the data types for which there is an appropriate distance function defined.

**spline**

Interpolates from one value in the 'values' list to the next according to a time function defined by a cubic Bézier spline

see [details](#)

See specifications.

### The animate keyTimes property

The keyTimes property divides the simple duration into intervals to compute the variations over each :

- The value [95] variations of the animated attribute [95] of a animate [94] element.

The attribute value must be defined for each interval at its origin and extremity (lists of values and keyTimes are the same size).



- The transformation matrix [95] variations of a animate transform [97] element.  
The transformation matrix must be defined for each interval at its origin and extremity (lists of transformation values and keyTimes are the same size).
- The target element position variations along the motion path [103] of a animate motion [101] element.  
The position along the motion path must be defined for each interval at its origin and extremity with the keyPoints [104] property (lists of keyPoints and keyTimes are the same size).

The keyTimes property can be used to control the transition speed between values.

Each keyTimes value is a percent of the duration. The first is zero and the last is 100. The input is done with an integer value between 0 and 100 to be more convenient, finally the value is converted as a real number between 0 and 1.

**KeyTimes**

Defines for each value when it should be used in the animation function.  
Each *keytime* value is input as a percentage of the simple animation duration.

none (default)

**keyTimes values** between (0,100)

from

to

If the interpolation mode is 'paced', the 'keyTimes' attribute is ignored.  
If the simple duration is indefinite, any 'keyTimes' specification will be ignored.

See animate using keyTimes showcase for an interactive demonstration.

See specifications.

## The animate keySplines property

With calMode [92] set as **spline**, the keySplines property define the cubic bezier curve of the interpolation between two values that controls interval pacing.

If the animation has (**n**) values, keySplines has (**n-1**) curves. Each curve is defined with the coordinates of the 2 tangents origin and extremity (x1,y1,x2,y2).

Each coordinate is input as a positive integer value between 0 and 100 to be more convenient, then the value is converted as a real number between 0 and 1 by the task.

**KeySplines**

A set of Bézier control points associated with the 'keyTimes' list, defining a cubic Bézier function that controls interval pacing.

none (default)

**keySplines coordinates** between (0,100)


Define the interpolation cubic spline for each time interval.  
Enter control points coordinates as percent or use the drawing to choose a spline sample and move each control point. The animation on the right shows the difference between linear and spline interpolation applied to the green circle ordinate as an example.

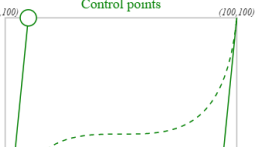
Time interval [0%, 25%]

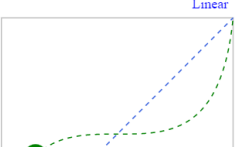
x1

y1

x2

Samples 

Control points 

Linear 



The task draw the cubic curve with an interactive SVG document.  
 Each tangent point can be moved, and the curve can be selected from one drawn sample.  
 See animate using keySplines showcase for an interactive demonstration.  
 See specifications.

## 9.6. Animate attribute

This task create or edit **animate** element. The task is enabled if one object is selected to animate it, or if the document contains animate elements.

The context of the task can be relative to the selection or to the document.

Task icon



This feature can be used :

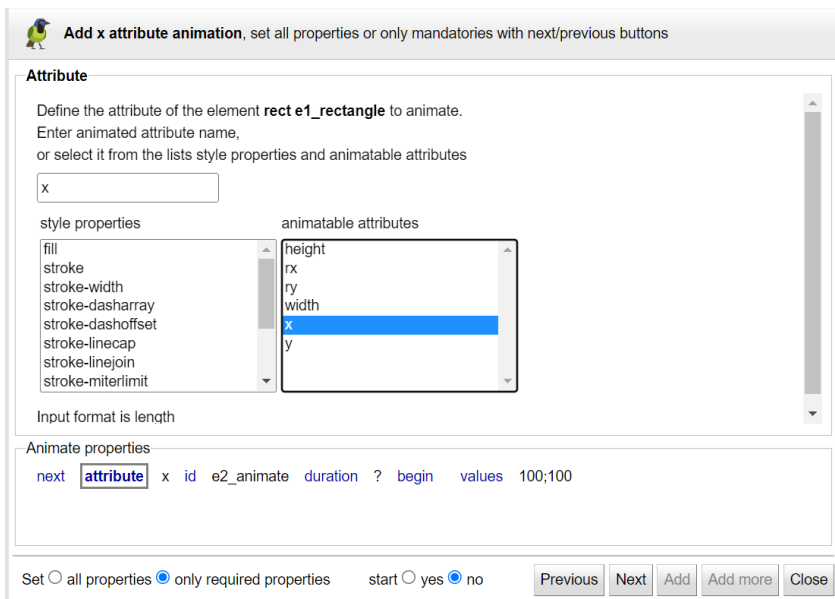
- To animate an attribute of the selected object that does not have yet one with the animate dialog [94].  
 Select the object then click on the task icon.
- To view the animate elements of the selection, then editing one or adding another one to the target of the selected animation with the animate elements dialog [96].  
 Select the objects then click on the task icon.
- To view all animate elements of the document, then editing one or adding another one to the target of the selected animation with the animate elements dialog [96].  
 Click on the task icon if enabled.

See animate attributes showcase for an interactive demonstration.

## The animate dialog

Create an animate element or update one.

The dialog is composed of an active pane, a pane links area and a bar of actions buttons.



This dialog is an assistant to create or edit an animate element

- To easily create an animation:
  - start by setting only the required properties ( attribute [95], id, duration [89], begin [88], values [95])
  - then edit the animation and set the other properties ( repeat count [91], repeat duration [91], fill [91], end [89], min [90], max [90], restart [90], calcMode [92], keyTimes [92], keySplines [93]).
- Use the next link and button to open the next property pane, or click on its link.
- Each property pane explains its property's behavior with links to standard specifications, and presents a simple and intuitive user interface for entering the property's value.
- By default start the created animation or restart the updated one.

## The animate attribute property

Define the attribute to animate, enter or select it from the DOM list or style properties list.

See specifications.

## The animate values property

This panel allow to specify the animation values of the attribute with one of the **5** possible syntaxes (from-to, from-by, by, to, values).

Each value is input with its required format (color, length, opacity, number..) using a dedicated UI.

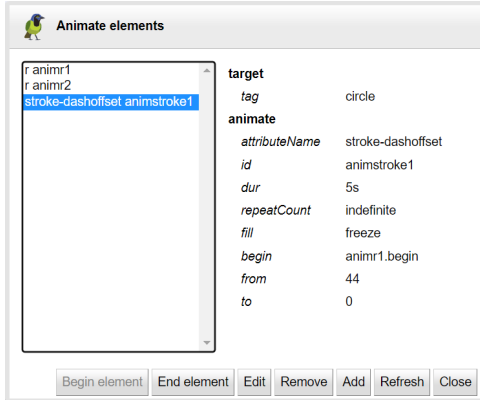
See syntax specifications ( values, from, to, by)



## The animate elements dialog

The animate attribute task [94] show this dialog when:

- the selection is empty and the document contains animate elements.
- the selected objects have animate elements.



The dialog shows:

- The list of the animate elements identified by the target attribute and the animation id.
- A panel with the description of the selected animation element.

The selected animation can be started, stopped and edited.

Use the add button to create another animate element on the target element of the selected animation.

## 9.7. Animate path

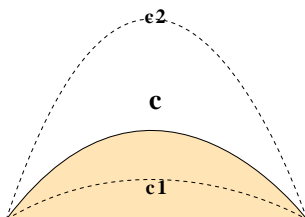
The animation of a path can be carried out with the animate [94] function by specifying the values of the attribute *d* describing the geometry of the path.

The drawsvg editor makes it easier to define the animation of a path by replacing the textual entry of the path description with the selection of a drawn element.

Each state of the path must be drawn in the form of an element path with the same structure (identical number and type of segments).

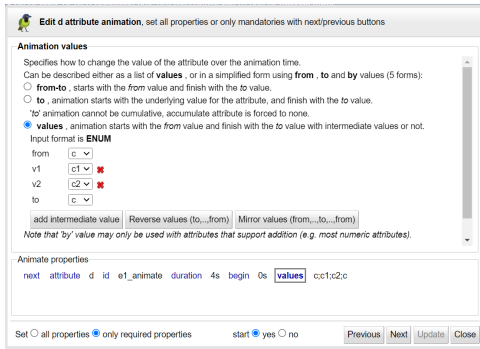
### Animate a path

Draw each state of the path as an element with the same structure.  
Use the grid and make a copy of the path to create each curve.

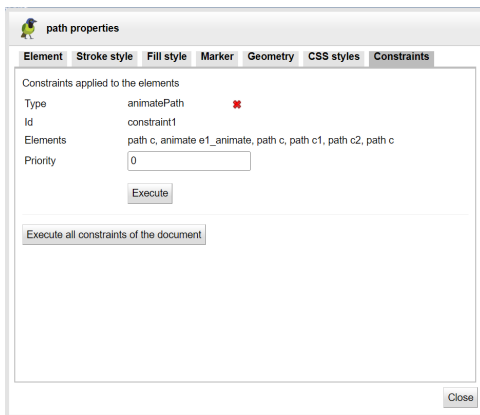


In the 'values' tab, select the path definition elements with the from-to, to and values syntaxes.

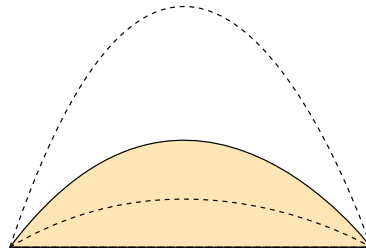




Modifications made subsequently to the control points of the drawn elements are carried over to the definition of the animation by a constraint.



## Animate a path



**Figure 9.1. Animate path quadratic sample**

See the online [sample](#) .

See showcases [animate path](#), [path with keyTimes](#) and [wave sample](#) for interactive demonstrations.

## 9.8. Animate transform

This task create or edit **animateTransform** element.

The task is enabled if one object is selected to animate it with an animateTransform element, or if the document contains animateTransform elements.



The context of the task can be relative to the selection or to the document.

Task icon



This feature can be used :

- To animate by a transformation the selected object that has not yet such animation with the animate transform dialog [98].  
Select the object then click on the task icon.
- To view the animateTransform elements of the selection, then editing one or adding another one to the target of the selected animation with the animateTransform elements dialog [100].  
Select the objects then click on the task icon.
- To view all animateTransform elements of the document, then editing one or adding another one to the target of the selected animation with the animateTransform elements dialog [100].  
Click on the task icon if enabled.

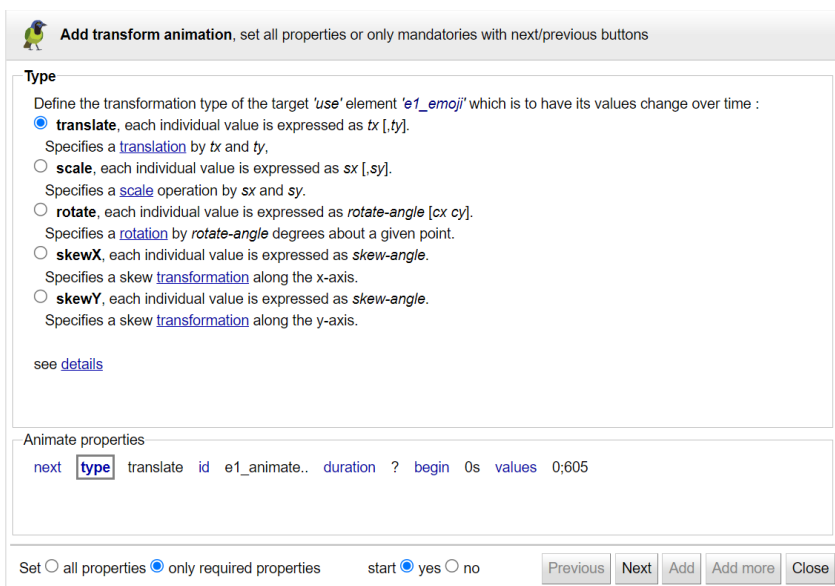
See animate-transform showcase for an interactive demonstration.

## The animate transform dialog

Create an animateTransform element (translate, scale, rotate, skewX, skewY) or update one.

The dialog is similar to the animate [94] dialog.

The dialog is composed of an active pane, a pane links area and a bar of actions buttons.



This dialog is an assistant to create or edit an animateTransform element

- To easily create an animation:  
start by setting only the required properties ( type [99], id, rotate and scale origin [99], duration [89], begin [88], values [99])  
then edit the animation and set the other properties ( repeat count [91], repeat duration [91], fill [91], end [89], min [90], max [90], restart [90], calcMode [92], keyTimes [92], keySplines [93]).
- The dialog has a pane for each property (16 panes).



- Use the next link and button to open the next pane, or click on its link.
- Each pane explains its property's behavior with links to standard specifications, and presents a simple and intuitive user interface for entering the property's value.
- By default start the created animation or restart the updated one.

## The animate transform type property

Select the type of the transformation from (translate, rotate, scale, skewX, skewY).

Selecting rotate or scale type will show the origin pane if the target element is not scaled (with a transform matrix that is not a translation).

**Type**

Define the transformation type of the element **use e1\_emoji** which is to have its values change over time :

- translate**, each individual value is expressed as *tx* [*ty*].  
Specifies a [translation](#) by *tx* and *ty*.
- scale**, each individual value is expressed as *sx* [*sy*].  
Specifies a [scale](#) operation by *sx* and *sy*.
- rotate**, each individual value is expressed as *rotate-angle* [*cx cy*].  
Specifies a [rotation](#) by *rotate-angle* degrees about a given point.
- skewX**, each individual value is expressed as *skew-angle*.  
Specifies a skew [transformation](#) along the x-axis.
- skewY**, each individual value is expressed as *skew-angle*.  
Specifies a skew [transformation](#) along the y-axis.

see [details](#)

## The animate transform origin property

This pane is enabled for rotate and scale animation applied to a target element which is not scaled (with a transform matrix that is not a translation).

The rotate animation can define its origin with the (cx,cy) parameters, unfortunately the scale animation can't. So to solve this problem, in addition to the animation parameters, it's recommended to define the origin with the CSS transform-origin properties. This solution has the advantage of sharing the origin between rotate/scale transformations applied to the same target element.

If the animation applied to the target element is only defined as a rotation transformation, the definition can be omitted/removed to use the (cx,cy) parameters of the rotate.

**Origin**

Define the transformation **origin** of the element using the CSS properties [transform-box](#) and [transform-origin](#).  
This definition is unique to the transformed element and applies to its transformations other than translate.  
It is essential when combine several transformations on the same element such as **rotation** and **scaling**.  
Select action to do:

- Define  Pass the transformation origin definition with CSS properties *transform-box* and *transform-origin*
- Modify**  Remove

- transform-box** property defines the *layout box* to which the transform relate.  
Set the *layout box* as
- transform-origin** property sets the *origin* for an element's transformations.  
Set the origin as  a position (*x-axis offset*,*y-axis offset*)  initial (50%,50%)  inherit

x-axis

y-axis

## The animate transform values property

This pane allow to specify the transform parameter values with one of the 4 possible syntaxes (from-to, from-by, by, values). The 'to' syntax (from animate element) has no meaning with animateTransform element.





Each value is input with its required format (translate, rotate, scale, skewX, skewY) using a dedicated UI.

The **translate** values pane with (tx,ty) parameters.

**Animation values**

Specifies how to change the value of the attribute over the animation time.  
 Can be described either as a list of **values** , or in a simplified form using **from** , **to** and **by** values (5 forms):

- from-to** , starts with the *from* value and finish with the *to* value.
- from-by** , starts with the *from* value and adds to it the *by* value over the simple duration.
- by** , starts with the underlying value for the attribute, and adds to it the *by* value over the simple duration.
- to** , animation starts with the underlying value for the attribute, and finish with the *to* value.

'to' animation cannot be cumulative, accumulate attribute is forced to none.

- values** , animation starts with the *from* value and finish with the *to* value with intermediate values or not.

Input format is **translate tx[,ty]**  
**translation** by *tx* and *ty*, if *ty* is not provided, it is assumed to be zero.

from *tx* \* 0   [*ty* *ty*  ]

to *tx* \* 250   [*ty* *ty*  ]

Reverse values (to,...,from)    Mirror values (from,...,to,...,from)

The **rotate** values pane with angle parameter and origin coordinates (cx,cy) in the case it is not defined with CSS transform-origin properties.

**Animation values**

Specifies how to change the value of the attribute over the animation time.  
 Can be described either as a list of **values** , or in a simplified form using **from** , **to** and **by** values (5 forms):

- from-to** , starts with the *from* value and finish with the *to* value.
- from-by** , starts with the *from* value and adds to it the *by* value over the simple duration.
- by** , starts with the underlying value for the attribute, and adds to it the *by* value over the simple duration.
- to** , animation starts with the underlying value for the attribute, and finish with the *to* value.

'to' animation cannot be cumulative, accumulate attribute is forced to none.

- values** , animation starts with the *from* value and finish with the *to* value with intermediate values or not.

Input format is **rotate angle**  
**rotation** by *angle* degrees about the **transform-origin** defined on the target, optional point [cx,cy] is ignored.

from *angle* \* 0   [*cx* *cx*  *cy* *cy*  target  ]

to *angle* \* 360   [*cx* *cx*  *cy* *cy*  target  ]

Reverse values (to,...,from)    Mirror values (from,...,to,...,from)

The **scale** values pane with (sx,sy) parameters.

**Animation values**

Specifies how to change the value of the attribute over the animation time.  
 Can be described either as a list of **values** , or in a simplified form using **from** , **to** and **by** values (5 forms):

- from-to** , starts with the *from* value and finish with the *to* value.
- from-by** , starts with the *from* value and adds to it the *by* value over the simple duration.
- by** , starts with the underlying value for the attribute, and adds to it the *by* value over the simple duration.
- to** , animation starts with the underlying value for the attribute, and finish with the *to* value.

'to' animation cannot be cumulative, accumulate attribute is forced to none.

- values** , animation starts with the *from* value and finish with the *to* value with intermediate values or not.

Input format is **scale sx[,sy]**  
**scale** by *sx* and *sy*, if *sy* is not provided, it is assumed to be equal to *sx*.

from *sx* \* 0   [*sy* *sy*  ]

to *sx* \* 2   [*sy* *sy*  ]

Reverse values (to,...,from)    Mirror values (from,...,to,...,from)

See animate transform showcase.

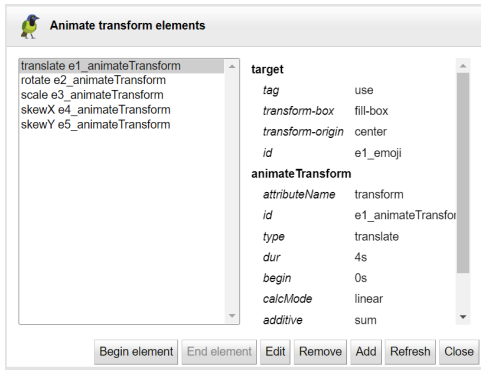
See syntax specifications ( values, from, to, by)

## The animate transform elements dialog

The animateTransform task [97] show this dialog when:

- the selection is empty and the document contains animateTransform elements.
- the selected objects have animateTransform elements.





The dialog shows:

- The list of the animateTransform elements identified by the animation type and id.
- A panel with the description of the selected animation element.

The selected animation can be started, stopped and edited.

Use the add button to create another animateTransform element on the target element of the selected animation.

## 9.9. Animate motion

This task create or edit **animateMotion** element.

The task is enabled if one object is selected to animate it with an animateMotion element, or if the document contains animateMotion elements.

The context of the task can be relative to the selection or to the document.

Task icon



This feature can be used :

- To move along a path the selected object that has not yet such animation with the animate motion [102] dialog.  
Select the object (and the motion path element) then click on the task icon.
- To view the animateMotion elements of the selection, then editing one or adding another one to the target of the selected animation with the animateMotion elements [105] dialog.  
Select the objects then click on the task icon.
- To view all animateMotion elements of the document, then editing one or adding another one to the target of the selected animation with the animateMotion elements [105] dialog.  
Click on the task icon if enabled.



### Note

The standard animateMotion element has **restrictions** that limits its applications :

- The motion is achieved by a transformation (translation+rotation) of origin **(0,0)** calculated on the path and applied to the target element.  
The target element must not be **transformed**.
- The motion path is defined by **referring** to a path element or **describing** its geometry.



The coordinates of the path must be defined in the same user system that the target element.

Drawsvg lifts these restrictions with **extensions** that make it easier to define motion animations:

- The motion **origin** is defined as a point relative to the target element bounding box. The motion is achieved by a transformation (translation+rotation) of origin **(0,0)** calculated on the path and applied to the target element.

Then this origin is applied to the target element by a **translate transformation** to be the (0,0) motion origin, or by a **transform animation** than begins before the motion.

If the target is transformed then the motion is applied to a **group** empacking the target element.

- The motion path can also be defined by retrieving its description from a **geometric** element (circle, ellipse, rectangle, polyline, polygon).

The coordinates of the resulting path are transformed into the same user system as the target element.

- All these extensions are automated by drawsvg **constraints**. For example, the modification of the geometry of the path definition element is automatically indicated in the description of the animation.

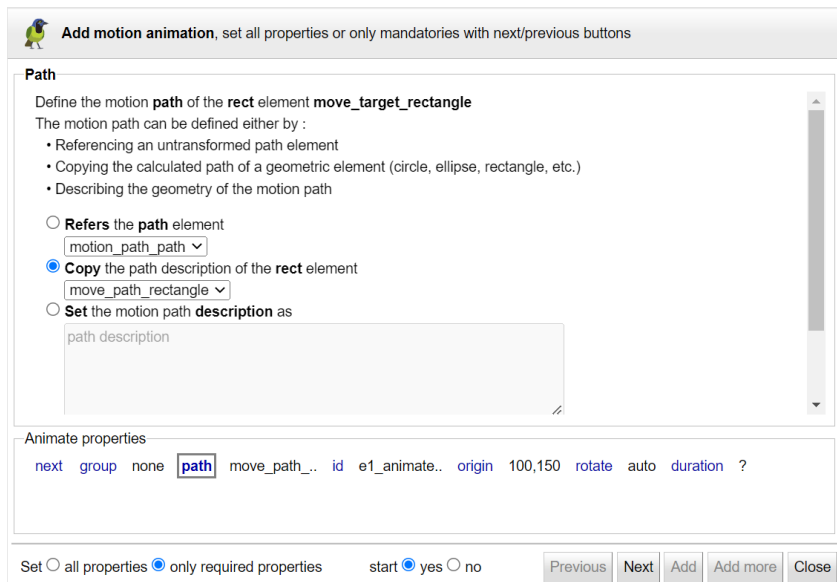
See motion path showcase for an interactive demonstration.

## The animate motion dialog

Create an animateMotion element or update one.

The dialog is similar to the animate [94] dialog.

The dialog is composed of an active pane, a pane links area and a bar of actions buttons.



This dialog is an assistant to create or edit an animateMotion element

- To easily create an animation: start by setting only the required properties ( group [103], id, path [103], origin [104], rotate [104], duration [89], begin [88])



then edit the animation and set the other properties ( repeat count [91], repeat duration [91], fill [91], end [89], min [90], max [90], restart [90], calcMode [92], keyTimes [92], keySplines [93], keyPoints [104]).

- The dialog has a pane for each property.
- Use the next link and button to open the next pane, or click on its link.
- Each pane explains its property's behavior with links to standard specifications, and presents a simple and intuitive user interface for entering the property's value.
- By default start the created animation or restart the updated one.

## The animate motion group property

This panel is only shown on animate motion creation.

Its controls the way the animation is created: under a group inclosing the target element or directly under the target.

Note that if the target element is transformed other than a translate, the creation within a group is necessary to preserve its transformation.

Select create (default if the target element is transformed) or none.

**Group**

Apply the animation to a **group** enclosing the target 'rect' element 'move\_target\_rectangle'.

**create** the group  
The animation is applied to a created group (default if animate motion on transformed target).

**none**  
The animation is applied to the target element (default if animate transform).

See motion group showcase.

## The animate motion path property

Select the motion path definition mode (Reference, copy, description) and its element or path description.

**Path**

Define the motion **path** of the **rect** element **move\_target\_rectangle**

The motion path can be defined either by :

- Referencing an untransformed path element
- Copying the calculated path of a geometric element (circle, ellipse, rectangle, etc.)
- Describing the geometry of the motion path

**Refers** the **path** element

**Copy** the path description of the **rect** element

**Set** the motion path **description** as

Depending on the selected mode:

- the wizard creates an mpath element (Reference)
- or populates the path attribute (Copy or describe) of the animateMotion element.





See motion path showcase.

## The animate motion origin panel

Define the motion origin point (using cx cy fields and target point listbox) and select the applied method transform or animate.

**Origin**

- **Define** the coordinates (cx,cy) of the motion origin (0,0) on the target element aligned on its bounding box. Select the target point or set coordinates (cx,cy).  
 cx | 100 | cy | 150 | target | center | point  
 If (cx,cy) not equals to (0,0) then define how to move the target.
- **Move** the target to the motion origin (0,0) using one of these two methods:
  - Transform** the target element to **set** its origin to (0,0).
    -  Forces the *fill* parameter to **freeze** and *begin* to **0s**.  
Associated default setting (1): *repeat count* parameter to **indefinite**.  
*Use this method if the animation starts from the beginning of the document with an indefinite duration.*
  - Animate**, start by a **translation** to (0,0) nested by the **motion** then ended by an inverse **translation** (2).
    -  Forces the *fill* parameter to **remove** and *begin* to target **click** event.

See motion origin showcase.

## The animate motion rotate property

Select the rotate mode.

**Rotate**

Defines a **rotation** applied after the translation transformation that is computed on the motion path as:

- auto**  
Indicates that the object is rotated over time by the angle of the direction (i.e., directional tangent vector) of the motion path.
- auto-reverse**  
Indicates that the object is rotated over time by the angle of the direction (i.e., directional tangent vector) of the motion path plus 180 degrees.
- angle**  
Indicates that the target element has a constant rotation transformation applied to it, where the rotation angle is the specified number of degrees.

see [details](#)

See specifications

## The animate motion keyPoints property

Set keyPoints values (one value for each value of keyTimes).

Each keyPoints value identifies a point on the motion path positioned by its curvilinear abscissa expressed as a percentage of the length of the path.

**KeyPoints**

Sets how far along the motion path the object should move at the time specified by the corresponding *keyTimes* value. Each *keypoint* value is entered as a percentage of the motion path length.

- none** (default)
- keyPoints values** between (0,100)

from   
  
 to

*There must be exactly as many values in the 'keyPoints' list as in the 'keyTimes' list.*  
*If the interpolation mode is 'paced', the 'keyTimes' attribute is ignored and therefore 'keyPoints'.*  
*If the simple duration is indefinite, any 'keyTimes' and 'keyPoints' specification will be ignored.*

see [keyPoints](#), [keyTimes](#) details

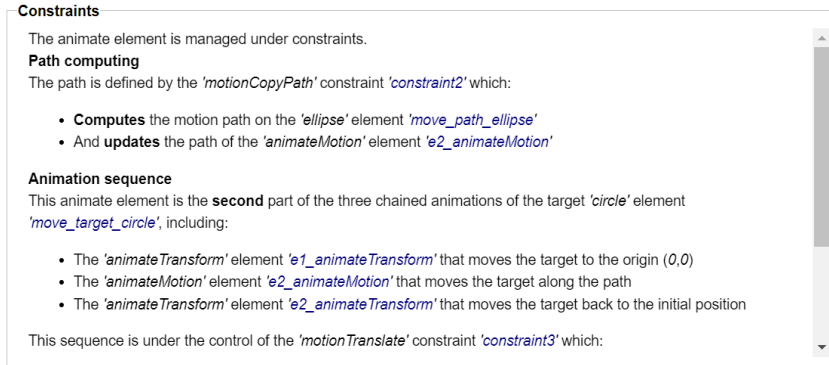


See motion keySplines showcase.

See specifications

## The animate motion constraints panel

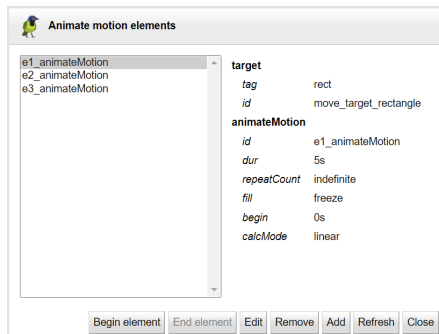
This panels describes the constraints defined on the edited animateMotion element.



## The animate motion elements dialog

The animateMotion task [101] show this dialog when:

- the selection is empty and the document contains animateMotion elements.
- the selected objects have animateMotion elements.



The dialog shows:

- The list of the animateMotion elements identified by the animation id.
- A panel with the description of the selected animation element.

The selected animation can be started, stopped and edited.

Use the add button to create another animateMotion element on the target element of the selected animation.

## 9.10. Set attribute

The set command assign a value to an attribute of a target element for a specified duration without transition. It is a restriction of the animate command:

- accepts only the 'to' syntax with one value
- calcMode, keyTimes and keySplines are ignored



Use this command to fix the state of an element during animations.

This task create or edit **set** element. The task is enabled if one object is selected to animate it, or if the document contains set elements.

The context of the task can be relative to the selection or to the document.

Task icon



This feature can be used :

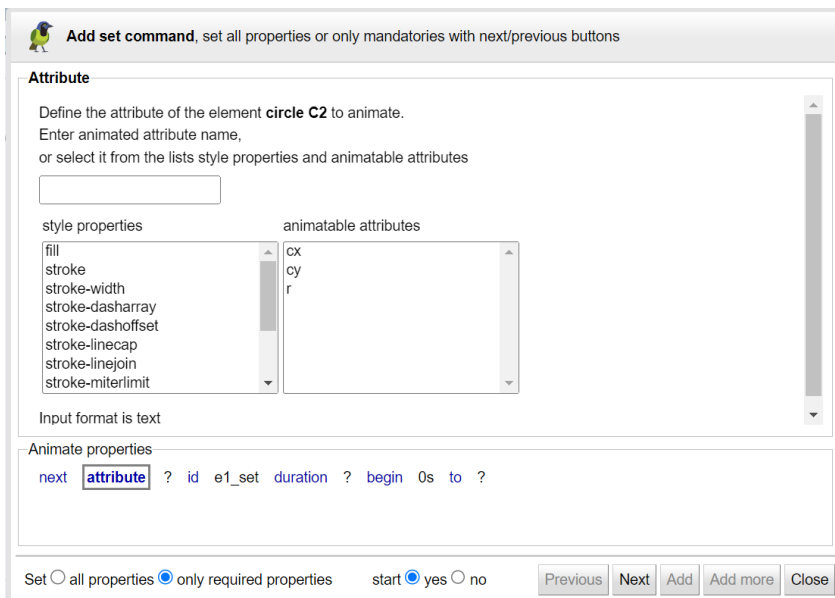
- To set an attribute of the selected object that does not have yet one with the set dialog [106].  
Select the object then click on the task icon.
- To view the set elements of the selection, then editing one or adding another one to the target of the selected animation with the set elements dialog [107].  
Select the objects then click on the task icon.
- To view all set elements of the document, then editing one or adding another one to the target of the selected animation with the set elements dialog [107].  
Click on the task icon if enabled.

See set attributes showcase for an interactive demonstration.

## The set command dialog

Create a set element or update one.

The dialog is composed of an active pane, a pane links area and a bar of actions buttons.



This dialog is an assistant to create or edit a set element

- To easily create a set element:  
start by setting only the required properties ( attribute [107], id, duration [89], begin [88], to [107])



then edit the set element and define the other properties ( repeat count [91], repeat duration [91], fill [91], end [89], min [90], max [90], restart [90]).

- Use the next link and button to open the next property pane, or click on its link.
- Each property pane explains its property's behavior with links to standard specifications, and presents a simple and intuitive user interface for entering the property's value.
- By default start the created set element or restart the updated one.

## The attribute property

Define the attribute to set, enter or select it from the DOM list or style properties list.

**Attribute**

Define the attribute of the element **circle e1\_circle** to animate.  
 Enter animated attribute name,  
 or select it from the lists style properties and animatable attributes

<p>style properties</p> <ul style="list-style-type: none"> <li>fill</li> <li>stroke-width</li> <li>stroke</li> <li>stroke-dasharray</li> <li>stroke-dashoffset</li> <li>stroke-linecap</li> <li>stroke-linejoin</li> <li>stroke-miterlimit</li> </ul>	<p>animatable attributes</p> <ul style="list-style-type: none"> <li>cx</li> <li>cy</li> <li>r</li> </ul>
---	--

Input format is length

See specifications.

## The to property

Specifies the value for the attribute during the duration of the set element.

The value is input with its required format (color, length, opacity, number..) using a dedicated UI.

**To value**

Specifies the value for the attribute during the duration of the 'set' element.  
 Input format is **text**

xy ▾

see [details](#)

See syntax specification to.

## The set elements dialog

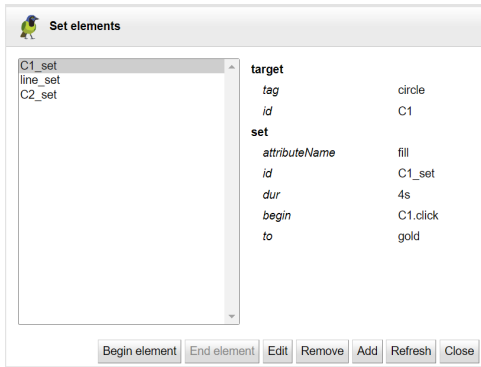
The set attribute task [105] show this dialog when:

- the selection is empty and the document contains set elements.





- the selected objects have set elements.



The dialog shows:

- The list of the set elements identified by their id.
- A panel with the description of the selected set element.

The selected set element can be started, stopped and edited.

Use the add button to create another set element on the target element of the selected one.

## 9.11. Animate gradients

This task animates linear and radial gradient style [72] elements.

This task create or edit **animate** [94] and **animateTransform** [97]. The task is enabled if one object is selected with a gradient applied as a fill [71] or stroke [69] property.

Task icon



This feature can be used :

- To **animate** an attribute of the stroke or fill gradient of an object with the animate dialog [94]:
  1. Select the object then click on the task icon
  2. Choose from the selection dialog [109] the element to animate: the gradient element or one of its stop elements
  3. Then click on the animate button
- To **animate transform** the stroke or fill gradient of an object with the animate transform dialog [98]:
  1. Select the object then click on the task icon
  2. Choose from the selection dialog [109] the gradient to animate
  3. Then click on the animate transform button
- To **edit** (begin, end, remove) an animation applied to the stroke or fill gradient of an object:
  1. Select the object then click on the task icon
  2. Choose from the selection dialog [109] the animation to edit (begin, end, remove)
  3. Then click on the edit (begin, end, remove) button

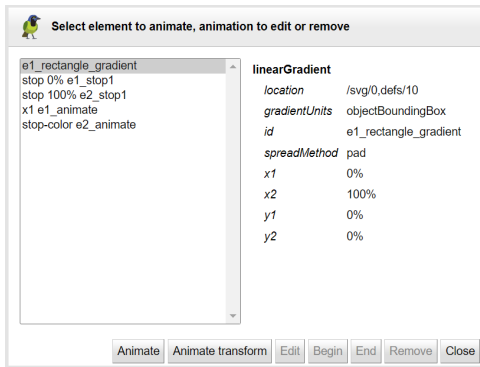
See animate gradients showcase for an interactive demonstration.



## The animate gradient selection dialog box

The context of the task consists of the gradients applied to the selected object, their stopping elements and the animations applied to all these elements.

The task's selection dialog box show context elements and actions that can be performed (animate, transform, modify, start, finish, delete) on them:



The dialog box show the selected element details.

## 9.12. Animate patterns

This task animates pattern style [72] elements.

This task create or edit **animate** [94] and **animateTransform** [97]. The task is enabled if one object is selected with a pattern applied as a fill [71] or stroke [69] property.

Task icon



This feature can be used :

- To **animate** an attribute of the stroke or fill pattern of an object with the animate dialog [94]:
  1. Select the object then click on the task icon
  2. Choose from the selection dialog [110] the element to animate: the pattern element or one of its stop elements
  3. Then click on the animate button
- To **animate transform** the stroke or fill pattern of an object with the animate transform dialog [98]:
  1. Select the object then click on the task icon
  2. Choose from the selection dialog [110] the pattern to animate
  3. Then click on the animate transform button
- To **edit** (begin, end, remove) an animation applied to the stroke or fill pattern of an object:
  1. Select the object then click on the task icon
  2. Choose from the selection dialog [110] the animation to edit (begin, end, remove)
  3. Then click on the edit (begin, end, remove) button

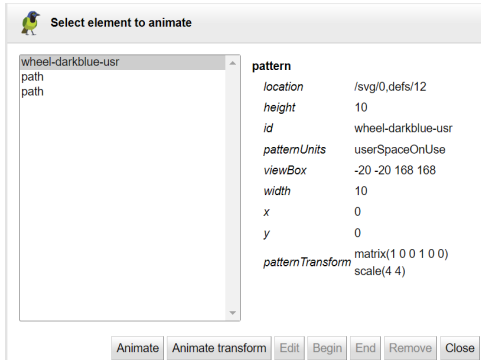
See animate patterns showcase for an interactive demonstration.



## The animate pattern selection dialog box

The context of the task consists of the patterns applied to the selected object, their content elements and the animations applied to all these elements.

The task's selection dialog box show context elements and actions that can be performed (animate, transform, modify, start, finish, delete) on them:



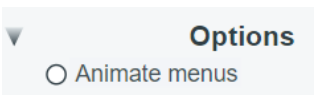
In the example the pattern consists of two path elements which can also be animated. The dialog box show the selected element details.

---

# Chapter 10. Options

This chapter describes the options of Draw SVG.

**Table 10.1. Options menu**

Menu	Options	
	Animate	Animate menus (Chrome and FireFox only)



# Chapter 11. Tools

This chapter describes the additional tools associated with the editor.

## 11.1. Photo to drawing

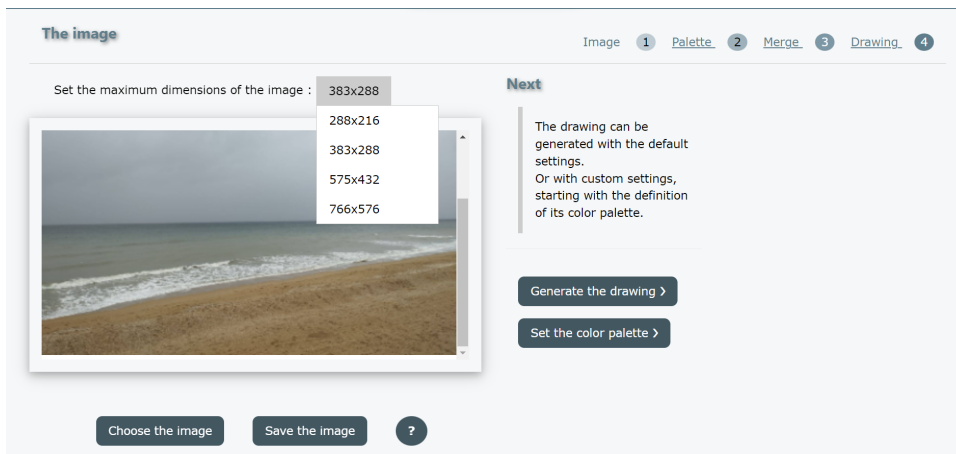
With this tool load an image file (png, jpg, ...) and generate a SVG drawing. The generated drawing may look like an Impressionist painting.

The generation process follows four steps:

1. Select the image file and reduce its size
2. Choosing the color palette
3. Fusion of nearby pixels
4. Extraction of contours with filtering and smoothing vertices.

### Step 1 the image

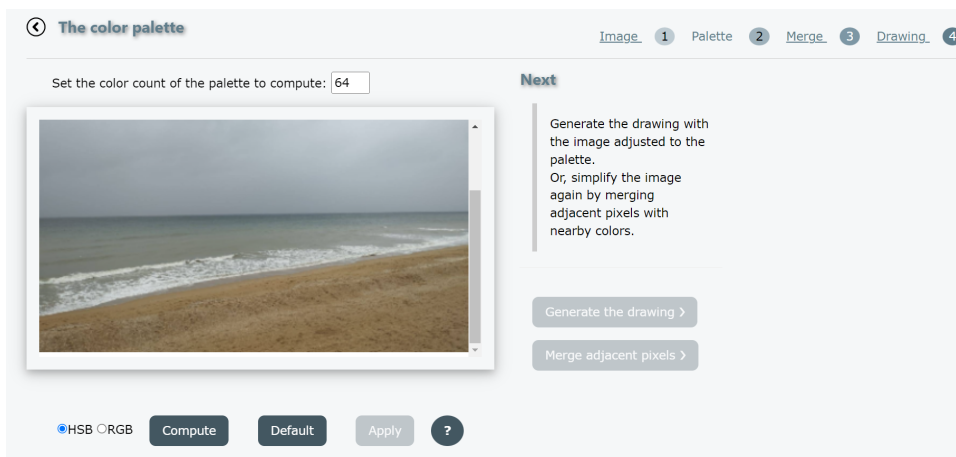
Choose the image file and select the dimensions.



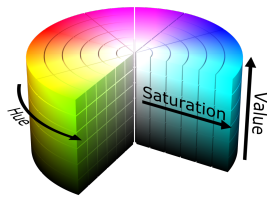
### Step 2 the color palette

The goal of this step is to reduce the number of colors in the image, which can be very large, to a fixed number by calculating a color palette using a quantification algorithm.

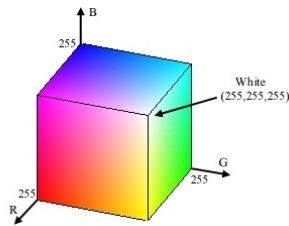
The image is then transformed by applying the color closest to the palette to each pixel.



The quantization algorithm calculates the distances between image colors in a three-dimensional space that can be:



The **HSV** (Hue Saturation Value) cylinder



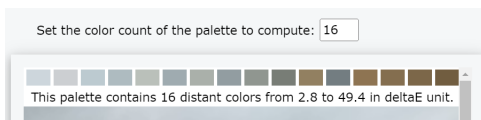
Or the **RGB** (Red Green Blue) cube

Depends of the image HSV space can delivers better results.

The default palette provides a minimum distance between colors **5** in deltaE unit.

- **Select** the color space, then
- **Calculate** the default palette
- Or **enter** the desired number of colors and **calculate** it

The palette is drawn at the top of the image:



**Apply** the color palette, the color of each pixel is assigned to the closest color in the palette :



The image before



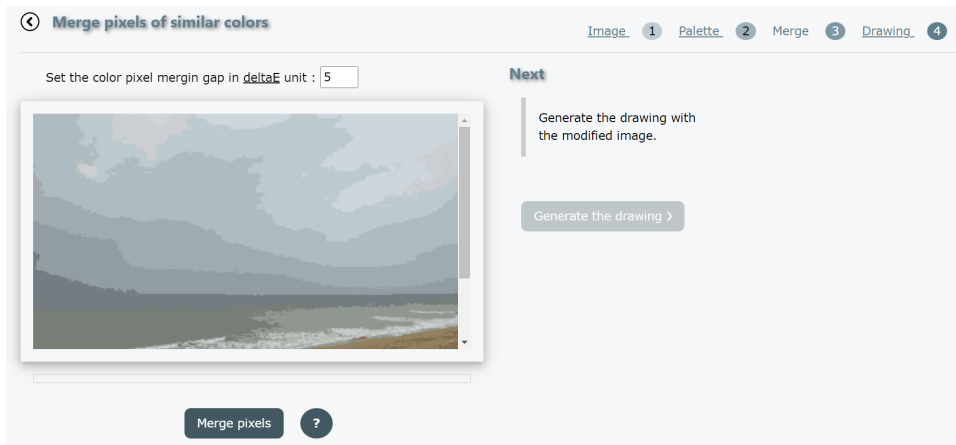
The image after

The step can be replayed until the correct palette is found. Don't forget that the number of colors chosen affects the size of the SVG file.

### Step 3 merge pixels of similar colors

This step affects the color of a pixel to its adjacent one, if the distance of its color from that of the adjacent one is less than the chosen distance in deltaE unit.





**Enter** the gap in deltaE units, then **merge** the pixels.

The operation reduces the number of contours:

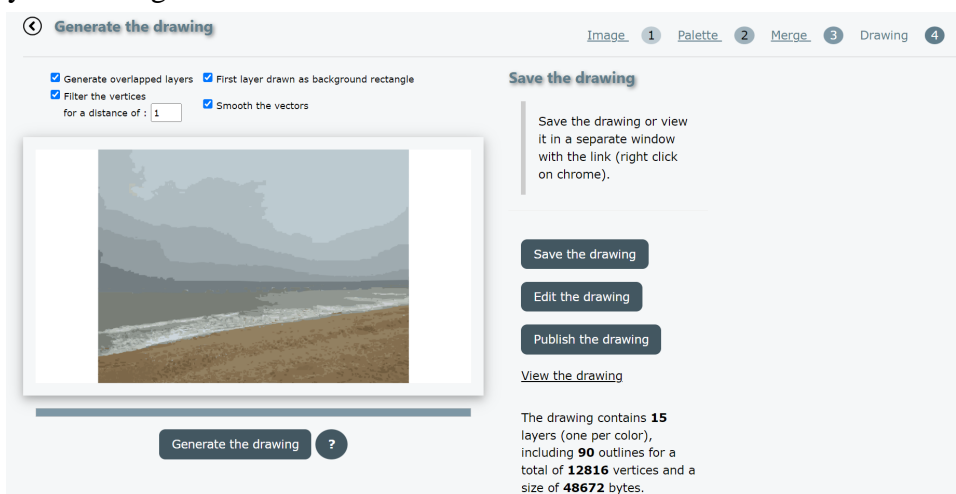


The image before merging pixels

The image after merging pixels with a gap of 5 deltaE units, the number of modified pixels is displayed below the image

## Step 4 generate the drawing

The drawing is generated by extracting contours of identical colors, then filtering the vertices and finally smoothing the vectors.



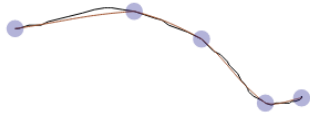
The drawing is structured with one layer per color. Layers are generated in descending order of pixel count with a few options:

- Generate **overlapped** layers



Each layer is extended to cover the following ones, otherwise the layers are disjointed and adjacent by their contours. This generally simplifies outlines by removing holes and removes stroke effects at layer boundaries.

- First layer drawn as background **rectangle**  
This option reduces the size of the file produced as the dominant color covers the background.
- **Filter** vertices with specified distance  
This option simplifies the contours by removing unnecessary points (point with a distance to the chord less than the threshold) and removes staircase effects.



- **Smooth** the vectors  
The contours are smoothed by substitution of the vectors with cubic or quadratic arcs depending on the curvature. If the layers are generated disjointed, only the concave portions of the contours are smoothed.



1 the transformed photo to be vectorized



2 generated drawing, contours without filters or smoothing



3 filtered contours by a distance of 4



4 smoothed contours

The generated drawing is described with its size, number of layers, contours and vertices.

Finally you can:

- **Save** the drawing SVG file
- **Edit** it with drawsvg editor





- **Publish** it with drawsvg board tool
- **View** it in new window

## 11.2. Base64 Image Encoder

With this tool load an image file (png, jpg, svg,..) and encode it into base64 data:URL to use it in your SVG, CSS and XHTML pages.

There is no transformation of the image content. The data:URL computed retains the type of the source image such as:

- image/png for a png file
- image/jpeg for a jpeg file
- image/svg+xml for a svg file
- ..



Figure 11.1. Base64 image Encoder



### Note

To copy text to the clipboard data:URL or HTML img code click on it and type ctrl + c. Then ctrl + v to paste it into your editor.

## 11.3. SVG to PNG Converter

With this tool load a SVG file and convert it into PNG image to use it in office documents or others.

The conversion is performed by the browser using **HTML5** features like canvas.

SVG **image** elements can reference **HTTP** resources (png, jpeg, .., svg) **external** to the document. It is thus possible to display in a SVG document dynamic images such as **Google maps** (as in the case of example **Villandry** ). But these resources are ignored by HTML img elements and canvas for **security** reasons. So the document in this state can not be converted to PNG format.

This tool **solves** this problem by loading these resources to encode them as data URI and make stand-alone document. The document thus transformed can be converted to PNG format. Resource loading is handled by the component "load http images."



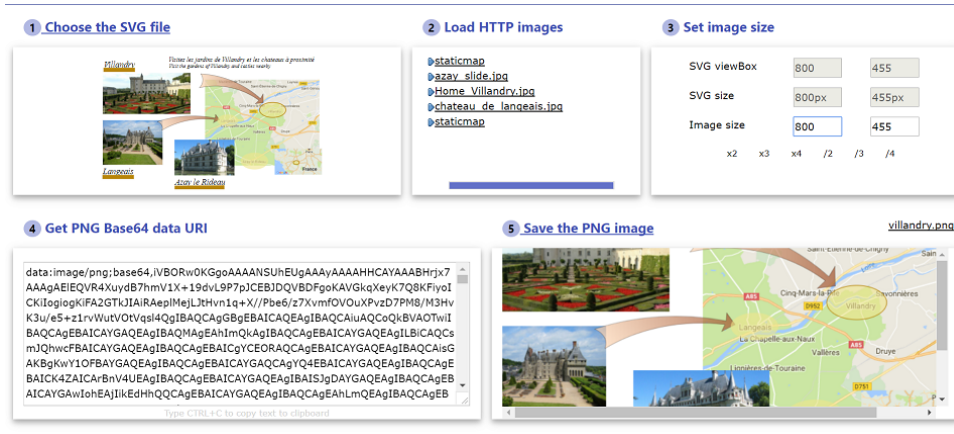


Figure 11.2. SVG to PNG Converter

## 11.4. Optimize SVG

Use this tool to optimize the size of your svg files before publishing them.

The svg file size can be reduced by optimizing geometry, style and text.

- Geometry optimizations are achieved by :

First by using path relative commands instead of absolute commands

- the argument separator optimization option allows you to reduce the number of commas (120,-10 becomes 120-10)
- The option of grouping successive operators allows the syntax to be further reduced (120,10,110.5 becomes 120,10,10,5)

Converting geometric objects to path (line,polyline,polygon)

Consolidation of transformation matrices (translate(..)scale(..) becomes matrix(..))

Reducing the number of decimal places

The size reduction can be significant.

- Style optimizations group properties assigned by attributes or inline (eg stroke=".." or style="stroke:...") into CSS classes.

CSS classes are computed to factorize properties between more than one element.

The last option, moving attribute properties to inline, is more a syntax convenient than a size optimization.

- Text optimizations are achieved by removing empty texts between items and also comments. These are useful when the SVG is hand written.



The screenshot shows the 'Optimize SVG' web application interface. It is divided into three main sections:

- 1 Choose the SVG file:** Displays a butterfly image. Below the image, it shows 'view box 0 0 415 300' and 'size 31,196 bytes, optimized 17,638 56%'. A link 'View the drawing (use right click on chrome)' is present.
- 2 Geometry metrics:** A table with columns 'Element type', 'Count', 'Data', and 'Decimals'.

Element type	Count	Data	Decimals
Rect			
Circle			
Ellipse			
Image			
Line			
Path (absolute)			
Path (relative)	3	18640	1
Polygon			
Polyline			
Group			
$\Sigma$	3	18640	
- 3 Geometry optimizations:** A list of checkboxes for optimization options:
  - Transform absolute to relative paths
  - Convert polygon to path
  - Convert polyline to path
  - Convert line to path
  - Consolidate transform matrix
  - Limit decimals count toBelow the list, it shows 'result 13,554 bytes, 43 %' and a help icon. Three buttons are at the bottom: 'Optimize geometry', 'Save', and 'Next'.

©2020 DRAW SVG All Rights Reserved.

Figure 11.3. Optimize SVG tool

---

# Chapter 12. Integration

DRAW-SVG can be used to edit svg documents within a web application.

There is two ways to integrate DRAW-SVG :

1. By URL and REST services
2. By API with jsChannel

DRAW-SVG website has several entry points to be called from another site.

Each entry point can have parameters. The general syntax of the URL for calling an entry point is :

```
https://drawsvg.org/drawsvg.html#  
entry  
:  
parameter1=value&parameter2=value
```

## 12.1. Integration by URL

This solution can only be used if your svg documents can be edited and updated by URL with a service (REST) to read (HTTP GET) documents and update them (HTTP PUT).

This use case is done with the ' **svgrurl** ' entry with two parameters:

1. the **url** parameter to specify the encoded url of the document to be edited.
2. the **save** parameter with true or false value.

If **true** , when saving the document, DRAWSVG will send a put request to the url with the document contents to save it (see save document [31] ).

3. the **fullWindow** parameter with true (default) or false value.

If **true** the document is loaded in full screen mode.

The general syntax is :

```
https://drawsvg.org/drawsvg.html#svgrurl:?url=encoded_url&save=true
```

To call DRAW-SVG with the svgrurl entry, use an HTML 'a' element or an iframe element :

```
<a href='https://drawsvg.org/drawsvg.html#svgrurl:?url=encoded_url&save=true'>edit  
with drawsvg</a>
```

Try this sample of loading the document jay.svg from freesvg.org .

```
https://drawsvg.org/drawsvg.html#svgrurl:?url=https%3A%2F%2Ffreesvg.org%2Fstorage  
%2Fzip%2Fblog%2Fjay.svg
```

## 12.2. Integration by API

### 12.2.1. Introduction

This is the most powerfull integration mode. It requires the use of an iframe. It uses mozilla jsChannel tool.

See the demo of this mode.

Choose your integration mode, **Online** or **Embedded** with **Edrawsvg** distribution.

Drawsvg can be integrated in the same way both online mode and embedded mode with **Edrawsvg** .

Which mode to choose ?



1. The **Online** mode has the advantages of benefiting from the latest version without updating with more services.
2. The **Embedded** mode with Edrawsvg allows you to group drawsvg editor and your application within your **WEB container** and runs on a private network without internet connexion.

**Edrawsvg** is a distribution of Draw SVG editor, intended to developers to be embedded into their websites. Edrawsvg is a free software released under the GNU LGPL.

Edrawsvg is a pure ajax web application, it's need a javascript engine. It's cannot be run directly from files.

The **war** distribution can be deployed on an apache tomcat server, **zip** distribution with Node.js engine.

### Request an API Key

To activate your drawsvg integration, and download Edrawsvg if you plan to use it, you should request a registered API Key .

### How to

Insert drawsvg within an iframe with the ' **jsChannel** ' entry and key parameter like this:

```
<iframe id="drawsvg" src="https://drawsvg.org/drawsvg.html#jsChannel:?key=yourKey"></iframe>
```

For the offline embedded release **edrawsvg** :

```
<iframe id="drawsvg" src="edrawsvg.html#jsChannel:?key=yourKey"></iframe>
```

The integration sequence is :

1. Establish a communication channel with drawsvg iframe.
2. Wait drawsvg ready notification
3. Call drawsvg functions when ready

To establish a communication channel:

```
// create drawsvg channel
var chan = Channel.build({
  debugOutput: true,
  window: document.getElementById("drawsvg").contentWindow,
  origin: "*",
  scope: "drawsvg"
});
```

To wait the drawsvg ready notification :

1. Define a function to receive notification
2. bind it as " **onDrawSVGReady** " to be called by drawsvg

```
    // drawsvg ready callback

    function
      onDrawSVGReady(trans,params) {
  // now you can communicate with drawsvg
  log("got drawsvg ready notification");
  return "connected";
};
// bind drawsvg ready callback
chan.bind("
  onDrawSVGReady
  ", onDrawSVGReady);
```



There are functions to control the document menu, edit and save svg documents.

More functions will be added to customize drawsvg under user requests.

**User profile**

Only features of the basic user profile are enabled with Edrawsvg (see chapter user profile [1]).

With jsChannel online integration, the expert profile remains activated if it was validated on the last access to drawsvg online on the same browser.

**12.2.2. Function setDocumentMenu**

This function is used to control the document menu.

Parameter	Type / description
enableSamples	boolean Enable (true) or disable (false) the samples sub-menu.
disableTasks	string Disable a list of tasks from: <ul style="list-style-type: none"> <li>• new, create a new document</li> <li>• open, open a existing document</li> <li>• save, save the document</li> <li>• print, print the document</li> <li>• exportpng, export the document as a png image</li> <li>• dimensions, set the document dimensions</li> <li>• resize, set the document viewBox with a rectangle</li> <li>• grid, set the grid tool parameters</li> <li>• importdef, import definitions from another svg document</li> <li>• gradient, open the gradient tool editor</li> <li>• pattern, open the pattern tool editor</li> <li>• filter, open the filter tool editor</li> <li>• marker, open the marker tool editor</li> </ul>

This example disables the new and open tasks, the user can only edit the loaded document :

```

// drawsvg ready callback

function
  onDrawSVGReady(trans,params) {
// now you can communicate with drawsvg
log("got drawsvg ready notification");
// enable buttons
document.getElementById("svg1btn").disabled=false;
document.getElementById("svg2btn").disabled=false;
// setting document menu
// call 'setDocumentMenu' method
// with params {enableSamples, disableTasks}

```



```

chan.call
({

method:
  "setDocumentMenu",

params:
  {
// disable samples sub-menu
'enableSamples' : false,
// disable taks new and open
'disableTasks' : 'new open'
},
// jsChannel callbacks

error:
  function(error, message) {
log( "
error:
  " + error + " (" + message + ")");
},

success:
  function(v) {
log("setting document menu done");
}
});
return "connected";
};

```

### 12.2.3. Function loadStringSVG

This function is used to edit a svg document when its contents is stored in a javascript string variable.

Parameter	Type / description
stringSVG	string The document contents
backgroundImageURL	string The background image URL, can be: <ul style="list-style-type: none"> <li>• A valid global URL (https: or https:)</li> <li>• A relative image file path from your domain starting with a slash character (ex: /home/images/..).</li> </ul> <p>The loadStringSVG function will add your domain adress to build a valid URL.</p> <ul style="list-style-type: none"> <li>• A encoded 64 base URI.</li> </ul> <p>The image dimensions should have the same ratio width/height than the svg viewBox.</p> <p>This will create an image element with the URL and geometry (x,y,width,height) to cover the svg viewBox.</p>



Parameter	Type / description
nameSVG	string The svg name
saveService	string The name of the bind function to be called to save the document when the user click on the save button of drawsvg.
showSaveDialog	boolean false: the dialog will not be displayed on save (default is true).
fullWindow	boolean Change (true) the dimensions of the document to the full window (100%) or (false) view the document with it's size.
saveButtonLabel	boolean The label of the save button.
onLoad	string The function called by drawsvg when the document is loaded.
onError	string The function called by drawsvg when the document cannot be loaded.



**Note**

This function is asynchronous

Example:

```

chan.call
({
  method:
    "loadStringSVG",

  params:
    {
// string svg contents
'stringSVG' : stringSVG1,
// svg name
'nameSVG' : 'svg1',
// The name of the service which to be called
// when user clicks on the save button of drawsvg
'saveService' : 'onSaveSVG',
// don't show save dialog
'showSaveDialog' : false,
// Change the dimensions to the full window (100%)
'fullWindow' : true,
// svg loading callbacks
'onLoad' : function() {
  log("got svg1 onLoad notification");
}
}
}

```





```

    },
    'onError': function(err) {
      log("ERROR while loading svg1: "+err);
    }
  },
  // jsChannel callbacks

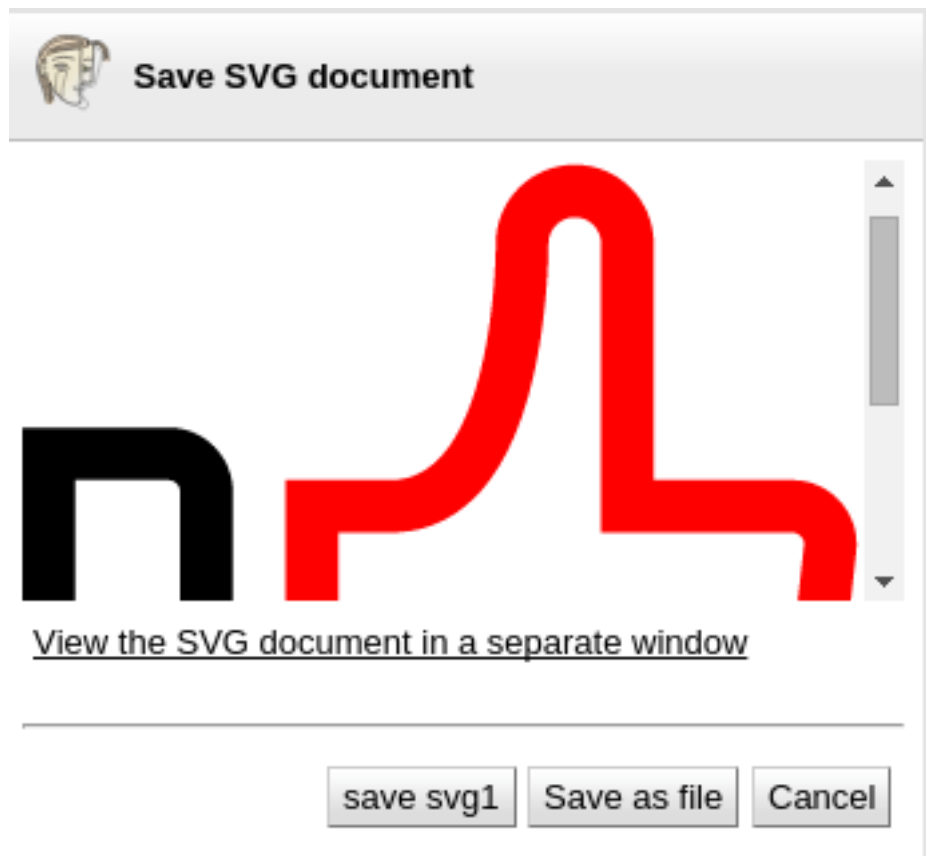
  error:
    function(error, message) {
      log( "
        error:
          " + error + " (" + message + ")");
    },

  success:
    function(v) {}
}
);

```

When the user clicks on the "save" icon of the document menu, then DRAW-SVG shows a dialog with a button to call the "saveService" function.

The label of the 'saveService' button is set with the parameter 'saveButtonLabel' value.



**Figure 12.1. Save dialog with save service button**

#### 12.2.4. Function loadUrlSVG

This function is used to edit a svg document identified by its URL.

Parameter	Type / Description
urlSVG	string The document URL
nameSVG	string The svg name
useEmbed	string Selection of the container type within drawsvg 1. true (default) , the svg is edited with an embed element 2. false, the svg is edited with an svg inline element.
saveService	string The name of the bind function to be called to save the document when the user click on the save button of drawsvg.
showSaveDialog	boolean false: the dialog will not be displayed on save (default is true).
fullWindow	boolean Change (true) the dimensions of the document to the full window (100%) or (false) view the document with it's size.
saveButtonLabel	string The label of the save button.
onLoad	string The function called by drawsvg when the document is loaded.
onError	string The function called by drawsvg when the document cannot be loaded.



**Note**

This function is asynchronous

The embed mode has the advantages to delegate the loading to the element and to isolate the svg contents from the drawsvg page. If the URL is not in the same domain, drawsvg will use its proxy service.

In the other case (useEmbed:false) drawsvg will load the document contents with an HTTP Request.

Example:

```
chan.call
({
method:
```



```

        "loadUrlSVG",

        params:
        {
// svg url
'urlSVG' : urlSVG2,
'nameSVG' : 'svg2',
// use inline svg instead of embed
'useEmbed' : true,
// The name of the save service
'saveService' : 'onSaveSVG',
// save button label
'saveButtonLabel' : 'save svg2',
// svg loading callbacks
'onLoad' : function() {
log("got svg2 onLoad notification");
},
'onError': function(err) {
log("ERROR while loading svg2: "+err);
}
},

        error:
        function(error, message) {
log( "
        error:
        " + error + " (" + message + ")");
},

        success:
        function(v) {}
    });

```

### 12.2.5. Callback saveService

To save the document with a dedicated function when the user click on the "save" button of drawsvg :

1. Declare the function with arguments (trans,params)
2. Bind it with a dedicated name
3. Set the 'saveService' parameter value with this name

The 'params' argument contains the name and the new contents of the document to save it

Parameter	Type	Description
stringSVG	string	The document content
nameSVG	string	The svg name
modified	boolean	Indicates whether the document has been modified.

Example:

```

// save svg service

function
onSaveSVG(trans,params) {

```



```

log("onSaveSVG nameSVG="+params['nameSVG']);
log("onSaveSVG modified="+params['modified']);
log("onSaveSVG stringSVG="+params['stringSVG']);
// save svg with stringSVG
// .... write your code
return "save done";
}

// bind save callback
chan.bind("onSaveSVG", onSaveSVG);

```

### 12.2.6. Function getSVG

This function is used to get the current SVG document returned as String.

This function has only one parameter.

Parameter	Type / Description
unloadBackgroundImage	boolean If true, the background image is removed from the output.

Example:

```

chan.call
({
  method:
    "getSVG",

  params:
    {},

  error:
    function(error, message) {
log( "
  error:
    " + error + " (" + message + ")");
},

  success:
    function(v) {
log(v);
}
});

```

### 12.2.7. Function getSVGObject

This function is used to get the current SVG document returned as object with properties.

It has only one parameter.

Parameter	Type / Description
unloadBackgroundImage	boolean



Parameter	Type / Description
	If true, the background image is removed from the output.

The current SVG document is returned as object with properties.

Property	Type / Description
modified	boolean Indicates whether the document has been modified.
stringSVG	string The SVG document

This function is asynchronous and does not have parameters.

Example:

```

chan.call
({
  method:
    "getSVG",
  params:
    {},
  error:
    function(error, message) {
log( "
  error:
    " + error + " (" + message + ")");
},
  success:
    function(v) {
log("modified="+v['modified']);
log("stringSVG="+v['stringSVG']);
}
});

```

### 12.2.8. Function addLayer

This function creates a layer (a g element child of the document root).

Parameter	Type / Description
layerId	string The ID of the layer

The layer can be used after receiving the success method.

Example:

```

chan.call

```



```

    ({
      method:
        "addLayer",

      params:
        {
          'layerId' : 'myLayer',
        }
    },
    // jsChannel callbacks

    error:
      function(error, message) {
log( "
      error:
        " + error + " (" + message + ")");
    },

    success:
      function(v) {}
  });

```

### 12.2.9. Function setInputLayerId

This function limits the operations of creating and modifying the elements of the documents to those included in the identified layer.

The layer must exist and be a 'g' element.

Parameter	Type / Description
layerId	string The ID of the layer

Example:

```

chan.call
  ({
    method:
      "setInputLayerId",

    params:
      {
        'layerId' : 'myLayer',
      }
  },
  // jsChannel callbacks

  error:
    function(error, message) {
log( "
    error:
      " + error + " (" + message + ")");
  },

  success:
    function(v) {}
  });

```



## 12.2.10. Function setCustomShapeCatalog

This function can be used to define a custom shape catalog with the generated file by the tool custom shape catalog tool . (see chapter customization [131] )

Parameter	Type / Description
index	int The index of the catalog (from 1 to 4)
title	string The title of the catalog
stringSVG	string The generated svg document of the catalog
store	boolean Option to save the catalog in the local storage space of the browser.

Example:

```

        chan.call
        ({
            method:
                "setCustomShapeCatalog",

            params:
                {
                    'index' : 1,
                    'title': 'brands',
                    'stringSVG': doc,
                    'store':true
                }
        },
        // jsChannel callbacks

        error:
            function(error, message) {
log( "
                error:
                    " + error + " (" + message + ")";
            },

            success:
                function(v) {}
        });

```



# Chapter 13. Customization

## 13.1. Define a custom shape catalog

Shape catalogs are generated by merging svg files into one file and exported to drawsvg editor with the custom shape catalog tool .

Shape catalog are saved in the local storage of the browser then reloaded from the local storage on each session.

Drawing's inserted shapes are free from the catalog with no link or trace.

There are two methods for creating a catalog:

- Each svg document is append in the catalog as a symbol .

In this case drawsvg will append the symbol from the catalog to the drawing and the shape instance will be created with a use element.

The initial contents of the svg can not be modified, only style and transform can be applied to it (as it is with the Font Awesome catalog).

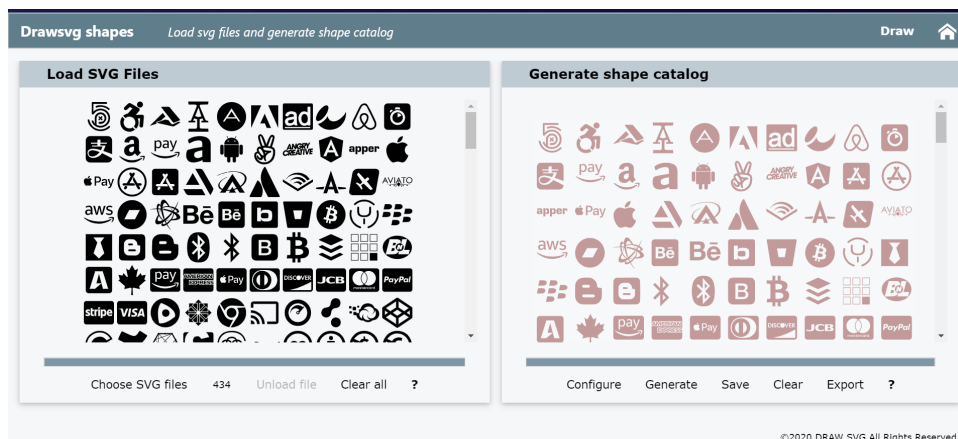
This is the simplest method for complex shapes such as emoji and when it has no sense to change them in the drawing.

- Each svg document contents is append in the catalog with a group (g element) ,

For this case, drawsvg will simply append the group and apply a transform to it.

Each element of the initial contents can be modified (as it is with the arrows catalog).

This method can be used to basic shapes.



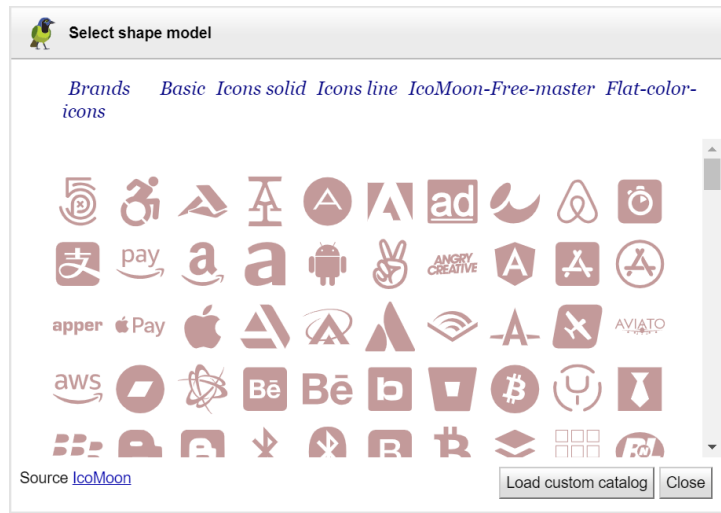
SVG files should contains only drawing elements, they should not contains defs or style elements to prevent from style conflicts.

- Select SVG files to build the catalog.
- Configure parameters of the shape catalog generator,
- Generate the catalog by merging svg files in one file.
- Shape catalogs are svg files that you can save.
- Clear your browser local storage to remove existing shape catalogs.
- Export your catalog to drawsvg editor for each browser and device you use.
- Catalog can be defined also by program with the JsChannel API.

Custom shape catalogs are stored inside the local storage of the browser to be re-used for each drawsvg editor session with the shape chooser dialog.







Custom catalogs are shown first in the top bar. User can define from one to four catalogs. Catalog can be defined also by program with the JsChannel API. (see function `setCustomShapeCatalog [130]` )  
The generated svg files can also be loaded from the dialog.

## 13.2. Customize Edrawsvg color theme

Edrawsvg is using the w3-theme-blue-grey W3.CSS color theme.

To change it, select a color theme.

Then replace stylesheet import in `edrawsvg.html` file:

```
<link type="text/css" rel="stylesheet" href="drwapp/w3-theme-blue-grey.css"/>
```

By the selected color theme, for example:

```
<link rel="stylesheet" href="https://www.w3schools.com/lib/w3-theme-deep-orange.css"/>
```

The `edrawsvg.html` file can copied to do this changes.

## 13.3. Customize Edrawsvg shape catalogs

Put in the directory `edrawsvg/config/customshapes` your custom shape catalog files generated with `drawsvg` tool .

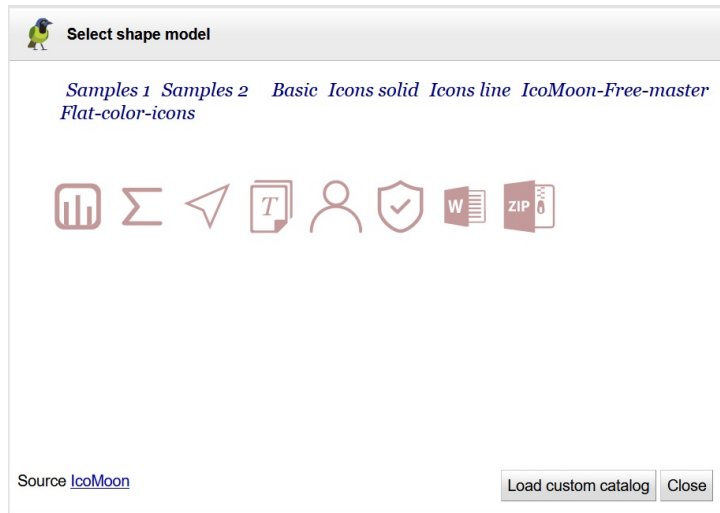
Declare them in the `index.json` file like this:

```
[
  {"index":1,"title":"Samples 1","file":"shape-samples1.svg"},
  {"index":2,"title":"Samples 2","file":"shape-samples2.svg"}
]
```

The file is in JSON format.



You can define up to 4 catalogs, index must be between 1 and 4.



## 13.4. Customize Edrawsvg parameters

To customize drawsvg engine behaviour change parameters definition in parameters.json file from edrawsvg/config/engine directory.

Parameter	Default value	Description
imageDefaultURL	samples/png/joconde.png	Define the default URL used by the image drawing task. The value must begin with / to define an URL relative to the domain.
defaultFillColor	rosybrown	define the default fill color of drawn shapes.
defaultShapeStrokeWidth	1px	define the default stroke width of SHAPE style category.
defaultAreaStrokeWidth	1px	define the default stroke width of AREA style category.
defaultContourStrokeWidth	1px	define the default stroke width of CONTOUR style category.

Sample

```
{
  "imageDefaultURL": "samples/png/joconde.png",
  "defaultFillColor": "rosybrown"
}
```

The file is in JSON format.



## 13.5. Customize DRAW SVG UI

### 13.5.1. Introduction

This chapter is intended for developers to create a personalized application for the DRAW SVG editor.

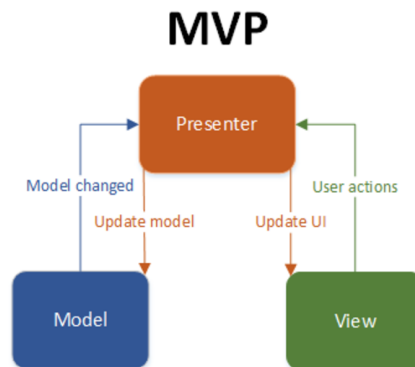
This development requires the Edrawsvg [120] distribution

To download Edrawsvg, you should request a registered API Key .

See this sample of customized application.

### 13.5.2. The MVP pattern

DRAW SVG editor is designed on the MVP (Model View Presenter) .



The application is structured with logical views that interact with the SVG document through presenters.

Each view has a given presenter and is responsible to update its UI objects. Presenters does not seen UI objects.

The structure of views and UI objects (buttons, inputs, labels,..) is full free of choices.

Drawsvg has **12** views and corresponding presenters

**Table 13.1. DRAW SVG views and presenters**

View	Responsibility	Presenter
Document view	Shows and run document management tasks	Presenter
Navigation view	Shows and run viewing actions (zoom in/out, pan,..)	Presenter
Selection view	Shows selection and run available presenter's tasks on selected elements	Presenter
Edit view	Shows and run (select, undo, redo) tasks	Presenter
Stoke view	Shows and modify the stroke style properties of the selected element.	Presenter
Fill view	Shows and modify the fill style properties of the selected element.	Presenter
Text style view	Shows and modify the text style properties of the selected element.	Presenter



View	Responsibility	Presenter
Marker style view	Shows and modify the marker style properties of the selected element.	Presenter
Elements view	Shows and run tasks to draw basic elements.	Presenter
Shapes view	Shows and run shape drawing tasks.	Presenter
Controls view	Shows and run controls drawing tasks.	Presenter
Layer view	Shows and run layers tasks.	Presenter

See the full API

By default UI objects of actions that depends on the selection should be **disabled** . Presenters send instructions to enable or disable UI when depends of the selection.

Implementation of views by the application can be **fully** , **partially** or **not exist** .



### Note

Each view and presenter are specified by a java interface because DRAW SVG kernel is written in java based on GWT framework.

### 13.5.3. The SVG viewing area

The SVG viewing area is defined by the application and is specified by the SVG view interface.

This view has no presenter and is responsible to:

- give the div element where to insert the svg engine window,
- indicates whether this area is scrollable, if not the SVG document is scaled to fully covers the area, otherwise it is displayed with its real size.
- give the maximun SVG viewing area size.

Implementation of the SVG viewing area must be **fully** .

### 13.5.4. The application

The application is implemented as a javascript object that gives the views as specified by its ISVGEngineApp interface.

The object application make the glue between the UI objects and views.

The application object is informed by the method `onSVGEngineLoad` when the application is ready to use.

The application javascript object must be registered as a global variable of the window under the name `svgEngineApp` to be identified by the engine.

Implementation of the application can be **fully** or **partially** .

```

// The SVG Engine application, implements ISVGEngineApp
var drwApp = {};

// Notify SVG engine ready
drwApp.onSVGEngineLoad =
    function
    (engine) {
    console.log("drwApp.onSVGEngineLoad engine="+engine);
    drwApp.engine=engine;
    };

```



```
// SVG view implements ISVGView (required)
drwApp.svgView = {
  // the svg window div element
  getSVGWindowContainer :
    function
      () {
        return document.getElementById("svgwindow");
      },
  // The maximum svg size
  getSVGMaxPixelSize :
    function
      () {
        console.log("drwApp.svgView.getSVGMaxPixelSize ");
        let w = document.getElementById("svgwindow");
        console.log("drwApp.svgView.getSVGMaxPixelSize "+w.clientWidth);
        return {width:w.clientWidth ,height:460};
      },
  // Setting the svg area size
  setSVGPixelSize :
    function
      (sz) {
    },
  // Indicates whether the visualization container has scrollbars.
  hasScrollbars :
    function
      () {
        return false;
      }
};
drwApp.getSVGView =
  function
    () {return drwApp.svgView;};

// implements views ....

// Register drwApp as svgEngineApp
window['svgEngineApp']=drwApp;
```

### 13.5.5. The DRAWSVG engine

The DRAWSVG engine is the controller, it detains view presenters and is visible from the application through its client API once ready.

The engine is automatically loaded on website startup:

- It load presenters,
- It find the object application,
- Inserts the SVG window widget with a default empty SVG document inside the div element given by the application,
- Calls the application `onSVGEngineLoad` when ready to use.

### 13.5.6. The drwapp sample application

The drwapp sample application must be understand as a quick start to explain how to develop a DRAWSVG custom application.

Its UI should be replaced by a custom one based on different choices.

Its javascript file can be duplicated to be adapted. It contents all the necessary objects and methods to implements.



Not desired implemented methods or views can be removed or leave empty.

### 13.5.7. Website integration

To integration the application inside a website and to interact with it use the JsChannel API in the same way for edrawsvg within a iframe (see chapter integration [119] ).

Insert the application within an iframe with the '**jsChannel**' entry and key parameter like this:

```
<iframe id="drawsvg" src="drwapp.html#jsChannel:key=yourKey"></iframe>
```

See demo of jsChannel drwapp integration

