

Systems Design Laboratory

Designing Graphical Interfaces with SVG

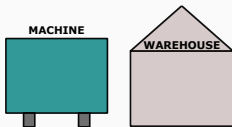
Matteo Zavatteri

¹Department of Mathematics, University of Padova, ITALY

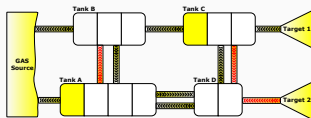
²Department of Computer Science, University of Verona, ITALY

Graphical Interfaces - Examples

Machine Warehouse



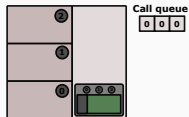
Gas Tanks



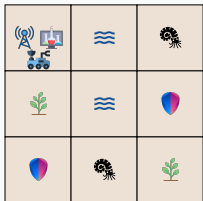
Traffic Lights



Elevator



Rover Sample Collection



Sample collection:

0

0

0

0

Sample analysis

Wolf-Goat-Cabbage



Scalable Vector Graphics (SVG) - Concepts Behind the Name



- ***Scalable*** = increasing or decreasing the graphics uniformly
- ***Vector*** = geometric objects (no “pixel information”)
- ***Graphics*** = a rich, structured description of vector and mixed vector/raster graphics

In vector graphics, a mathematical description of a drawing is given by means of coordinates, vectors, objects, etc. In order to display it on a computer screen, the vector description is converted to pixels to achieve maximum sharpness for all possible display resolutions. SVG is the international World Wide Web (W3C) Consortium standard for 2D.

<https://www.w3.org/TR/SVG11/>

SVG - Main Features



- **SVG is a language for describing two-dimensional graphics in XML**
- **SVG allows for 3 types of graphic objects:**
 - 1) **vector graphic shapes (e.g., paths consisting of straight lines and curves)**
 - 2) **images**
 - 3) **text**
- **Graphical objects can be grouped, styled, transformed and composited into previously rendered objects**
- **SVG drawings can be interactive and dynamic**



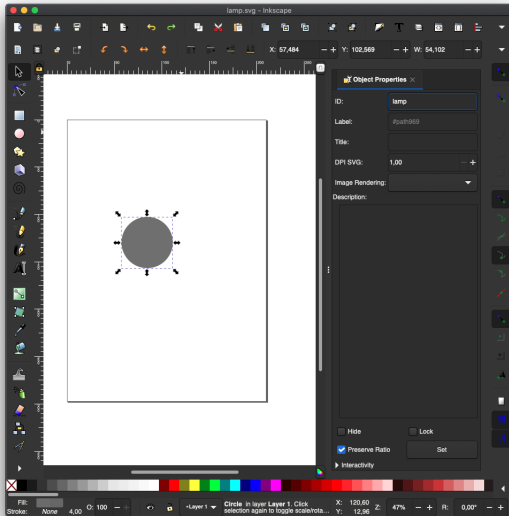
INKSCAPE

Draw Freely.

<https://inkscape.org>

- Inkscape is a free and open source vector graphics editor for GNU/Linux, Windows and MacOS
- Inkscape uses the standardized SVG file format
- It can import and export various file formats, including SVG, AI, EPS, PDF, PS and PNG
- It has a simple interface
- It has a growing international user community, and many learning materials exist to help get you started with your creations <https://inkscape.org/learn/>
- Inkscape is a member of the Software Freedom Conservancy, a US 501(c)(3) non-profit organization. Contributions to Inkscape are tax deductible in the United States.

SVG - Inkscape

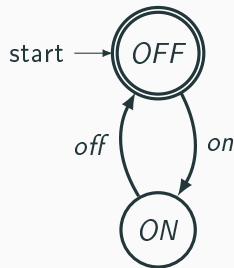


- Create a circle in Inkscape
- Edit Object Properties
- ID = "lamp"
- Save it as `plant.svg`

SVG - ESCET - A Lamp Example

A Plant Automaton
with an intuitive “on/off
semantics”

```
plant lamp:  
  controllable on;  
  controllable off;  
  
  location OFF:  
    initial; marked;  
    edge on goto ON;  
  
  location ON:  
    edge off goto OFF;  
end
```



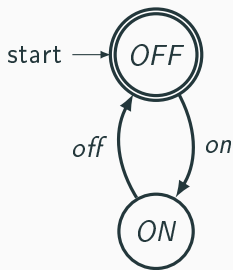
$\Sigma := \{on, off\}$

We would like to “connect” the CIF specification to the `plant.svg` file so that:

- when the lamp is on the circle becomes yellow and
- when the lamp is off the circle becomes gray

SVG - CIF - Filling Objects

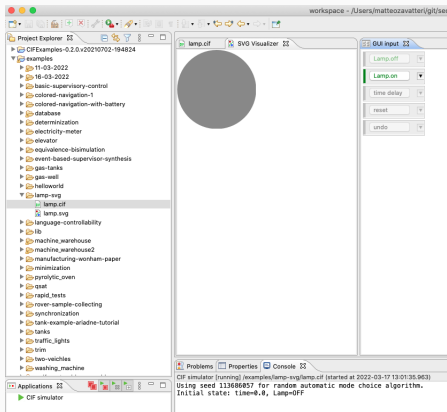
```
plant Lamp:  
  controllable on;  
  controllable off;  
  
  location OFF:  
    initial; marked;  
    edge on goto ON;  
  
  location ON:  
    edge off goto OFF;  
end
```



$\Sigma := \{on, off\}$

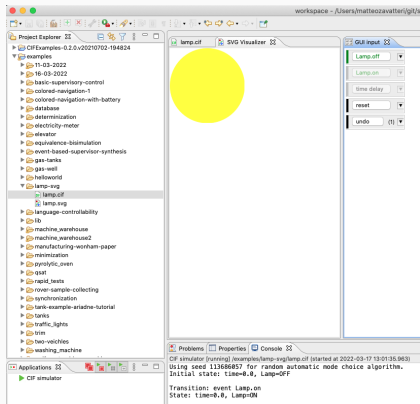
```
svgfile "lamp.svg";  
svgout id "lamp" attr "fill" value if Lamp.ON : "yellow"  
else "gray"  
end;
```


SVG - CIF - Filling Objects



The screenshot shows the CIF simulator interface. The Project Explorer on the left lists various examples, with 'lamp-cif' selected under the 'lamp-svg' folder. The main window is split into three panes: 'lamp.cif' on the left showing a grey circle, 'SVG Visualizer' in the center showing the same grey circle, and 'GUI Input' on the right with a dropdown menu set to 'Lamp.off'. The console at the bottom displays the following text:

```
CIF simulator [running] /examples/lamp-svg/lamp.cif (started at 2022-03-17 13:01:35.963)
Using seed 113686857 for random automatic mode choice algorithm.
Initial state: time=0.0, Lamp=OFF
```



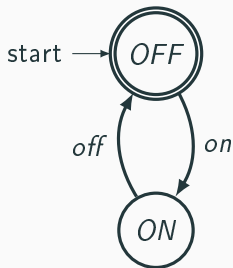
The screenshot shows the CIF simulator interface after a transition. The Project Explorer on the left is the same as in the previous screenshot. The main window is split into three panes: 'lamp.cif' on the left showing a yellow circle, 'SVG Visualizer' in the center showing the same yellow circle, and 'GUI Input' on the right with a dropdown menu set to 'Lamp.on'. The console at the bottom displays the following text:

```
CIF simulator [running] /examples/lamp-svg/lamp.cif (started at 2022-03-17 13:01:35.963)
Using seed 113686857 for random automatic mode choice algorithm.
Transition: event Lamp.on
State: time=0.0, Lamp=ON
```

fill: changes the color of an object

SVG - CIF - Moving Objects

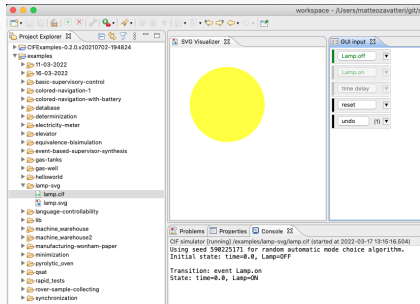
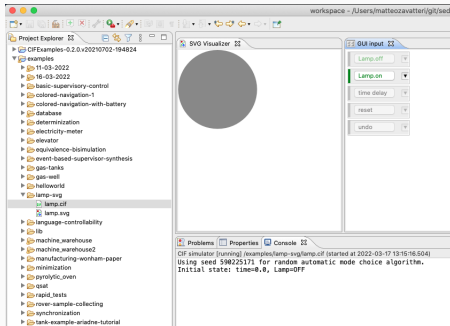
```
plant Lamp:  
  controllable on, off;  
  
  location OFF: initial; marked;  
    edge on goto ON;  
  
  location ON:  
    edge off goto OFF;  
end
```



```
svgfile "lamp.svg";  
svgout id "lamp" attr "fill" value if Lamp.ON : "yellow"  
  else "gray"  
  end;  
svgout id "lamp" attr "transform" value if Lamp.ON : "translate(10,10)"  
  else "translate(0,0)"  
  end;
```

$\Sigma := \{on, off\}$

SVG - CIF - Moving Objects



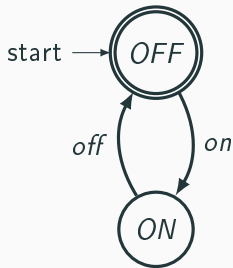
$\text{translate}(x,y)$: shifts an object of x and y

Remember to “group” complex svg objects if you want to move them altogether

See also <https://jenkov.com/tutorials/svg/svg-transformation.html> for more transformations

SVG - CIF - Displaying Objects

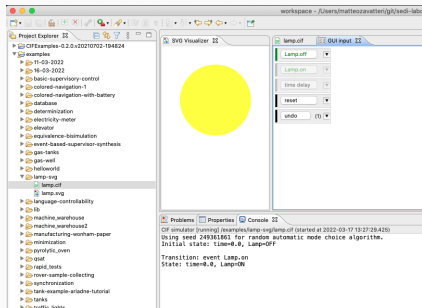
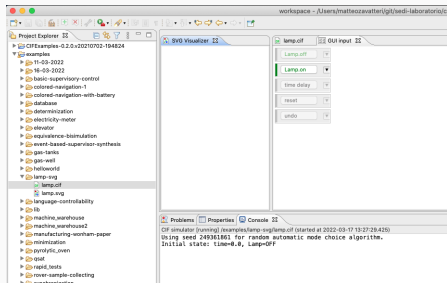
```
plant Lamp:  
  controllable on, off;  
  
  location OFF: initial; marked;  
    edge on goto ON;  
  
  location ON:  
    edge off goto OFF;  
end
```



```
svgfile "lamp.svg";  
svgout id "lamp" attr "fill" value if Lamp.ON : "yellow"  
  else "gray"  
  end;  
svgout id "lamp" attr "transform" value if Lamp.ON : "translate(10,10)"  
  else "translate(0,0)"  
  end;  
svgout id "lamp" attr "display" value if Lamp.ON : "inline"  
  else "none"  
  end;
```

$\Sigma := \{on, off\}$

SVG - CIF - Displaying Objects



display: **hides/shows an object**