

Scalable Vector Graphics in SAS 9.2

Nicola Tambascia, Accovion GmbH, Eschborn, Germany
Sven Greiner, Accovion GmbH, Eschborn, Germany

ABSTRACT

For more than ten years, Scalable Vector Graphics (SVG) has been a standard for displaying vector graphics on the internet. For the first time, the SAS 9.2 software supports the creation of graphics in SVG format. This document provides a brief introduction to the XML-based SVG standard and to the creation of SVG graphs using SAS/GRAPH® software.

Firstly, the concept of vector graphics and its technical implementation as scalable vector graphics are explained. Secondly, a number of examples are presented to illustrate the specific features of SVG using the SAS software. Thirdly, the advantages and disadvantages of SVG as a graphics format in the analysis and reporting of clinical trial data are discussed.

VECTOR AND RASTER GRAPHICS

When displaying an image on the computer you can distinguish between two types of graphics: vector graphics and raster graphics. Even before choosing the graphic format you should have decided for one of these two types. Since "Scalable Vector Graphics" (SVG) is a graphics format of vector graphics, in the following paragraphs, the differences between vector and raster graphics are explained and the advantages and disadvantages of each format are discussed.

Raster graphics, also called pixel graphics, consists of grid-like arrangement of picture points (pixels). Each pixel has a unique color assigned to it, so that the raster graphics resembles a mosaic. The image size of a raster graphics, i.e. the width and height in pixels is known as resolution and the differentiability of the known color values of each pixel as color depth. Common formats for displaying raster graphics on the computer are JPEG, bitmap, and GIF among others.

Vector graphics on the other hand are not stored in the form of pixels, but as graphical primitives. Vector graphics are object-oriented, and these graphic primitives (usually among others lines, circles, polygons, and curves) are treated as objects and corresponding attributes (for example, x and y coordinates, radius, color, line width) are assigned. So to store a circle as a vector graphic only the centre and the radius of the circle are needed in the simplest case.

Since the image information of vector graphics is stored in the form of graphic primitives and associated attributes, an interpreter is needed to display the graphics. Usually a browser or a graphics editing program fulfills this task. Widely used vector graphics formats are EPS (Encapsulated PostScript), SWF (Macromedia Flash) and SVG.

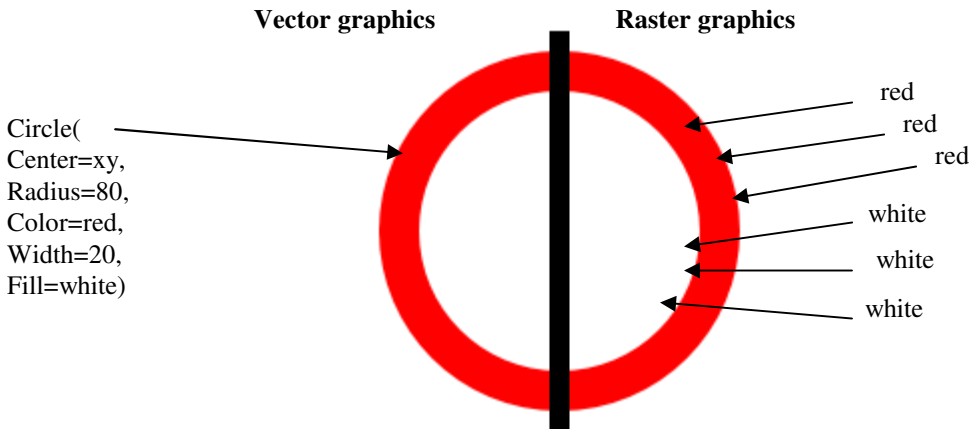
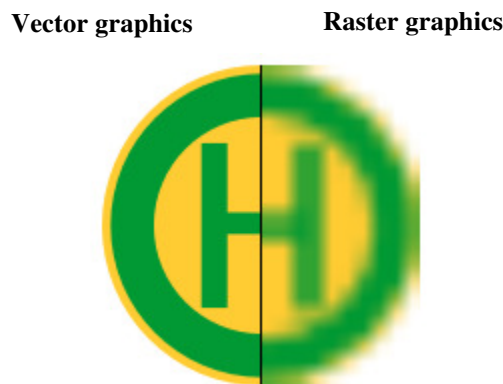


Figure 1.1: Representation of a circle as vector and raster graphics

Because of their inherent advantages and disadvantages, both types of graphics are suitable for different applications. When a graphic can be easily represented by graphical primitives, vector graphics is generally preferred over raster graphics. As shown in Figure 1.1, a circle can be stored as vector graphics with very little memory consumption and very high quality. As a raster image the color information for each pixel must be stored separately, despite the simplicity of the image. Especially with large images, this can quickly lead to a significant consumption of memory.



**Figure 1.2: Up scaling of a vector and raster graphics in comparison [1]
(Vector and raster graphics increased from 20x20 to 160x160 pixels)**

Another advantage of vector graphics over raster graphics is its scalability. When it is necessary to change the size of a vector graphic, neither when you zoom in, even when you zoomed out, reduces the quality of the representation or the consumption of the memory space. The graphical primitives are simply recreated in the appropriate size. However, scaling of raster graphics is not possible without losing quality. When a raster graphics is enlarged, new pixels are created and they receive the color information from neighboring pixels. The image information is distorted and the picture looks "washed out" (see Figure 1.2). In addition, the consumption of memory increases significantly. On the other hand, if the size of a raster graphics is decreased, pixels are deleted and image information is lost. Partly completely new colors arise when the color information from two (or more) pixels are merged into one in order to be able to delete the second (or all relevant) pixel.

The strength of raster graphics is the **representation of complex gradients**. These are usually found in images taken with a digital camera or digitized with a scanner. In this case it is not possible or very difficult to describe the shapes and colors of the graphic using graphical primitives. If you still try to represent complex gradients using vector graphics, the result usually looks artificial and blurry.

Due to the advantages and disadvantages of both graphic types they are used in different application areas. Usually, photo-realistic images are described as a raster graphics, while, wherever this is not necessary, vector graphics are used. Vector graphics are used in graphic applications (vector-based drawing programs like Corel Draw or

PhUSE 2011

Inkscape), page descriptions and writings on the computer (PostScript, PDF, TrueType) and internet applications (Macromedia Flash, SVG).

SCALABLE VECTOR GRAPHICS (SVG)

Scalable Vector Graphic (SVG) is the recommended standard to describe two-dimensional vector graphics on the Internet by the World Wide Web Consortium (W3C).

SVG was developed by Adobe Systems from the in-house language Precision Graphics Markup Language (PGML) and Microsoft's Vector Markup Language (VML). PGML and VML were previously rejected by the W3C before SVG, as a further development of both languages, was accepted in 1999. SVG version 1.1 is the currently recommended version since 2001. Version 1.2 is in development.

The technical implementation of SVG is formed by various elements that are stored in the form of XML (Extensible Markup Language). SVG classifies these elements into three groups of graphic objects:

- Vector graphics elements
- Raster graphics
- Text

Vector graphics elements describe the implementation of the various graphic primitives in SVG. The following elements are available to describe vector graphics: path <PATH>, rectangle <RECT>, circle <CIRCLE>, ellipse <ELLIPSE>, line <LINE>, polyline <POLYLINE> and polygon <POLYGON>.

In addition to the vector graphics elements, it is also possible to use **raster graphics** in SVG. In this case, the raster graphics must be available externally and can then be integrated, for example, as a background image for the vector graphics. The element <IMAGE> makes this possible. With the element <TEXT> **text** is embedded into the vector graphics.

Since SVG graphics are defined in XML, they can be read and edited with any text editor. The following code shows the content of an SVG file with some simple text and vector graphics elements. Figure 2.2 shows the generated image from the code.

```
<svg
  xmlns="http://www.w3.org/2000/svg"  version="1.1"
  width="500" height="450">
  <title>Text and graphic elements of SVG</title>

  <rect width="300" height="240" x="0" y="210" fill="blue" />
  <ellipse cx="280" cy="230" rx="190" ry="120" fill="yellow"/>
  <path d="M150 200 L50 400 L250 400 Z" stroke="black" fill="lime"
/>
  <text x="200" y="220" font-family="Arial" font-size="35">
    PhUSE 2011
  </text>
</svg>
```

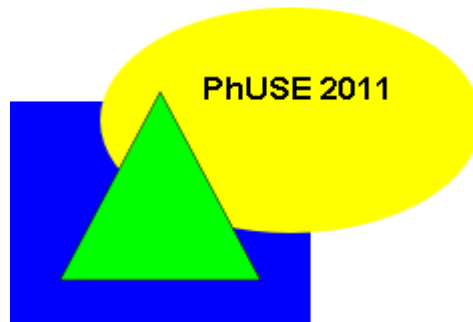


Figure 2.2: Text and graphic elements of SVG

The code begins with the **root element** <SVG> that encloses the entire domain of the SVG graphics. The attributes XMLNS (XML namespace) and VERSION specify the standard and version used to interpret the code. In this case,

PhUSE 2011

the W3C SVG standard version 1.1 is used. The WIDTH and HEIGHT attributes specify the size of the output area in which the image can be displayed.

It follows the <TITLE> element with which sets the text, that appears as a heading in the window of the graphic. The elements <LINE>, <RECT>, <CIRCLE> and <TEXT> and their various attributes define the visible shapes and text in Figure 2.2.

Of course, the forms shown in Figure 2.2 are very simple use cases for SVG. SVG also supports interactive and dynamic applications, i.e. it can also respond to user input and display animations. SVG is found in many portable devices because of its flexibility and its efficient way of working. Usually reduced versions of SVG such as SVG Basic and SVG Tiny are used in portable devices.

SVG was primarily designed to display graphics on the Internet. However, the browser support for SVG is not so far advanced that you can assume a proper representation in your browser. It should be noted at this point, that the Internet Explorer supports SVG only in the new version 9. Table 2.1 provides an overview of browser support for SVG:

Table 2.1: Browser-support for SVG 1.1 [2]

Browser	SVG 1.1 support
Opera 11.01	95,44 %
Google Chrome 10	89,23 %
Safari 5.0	82,48 %
Mozilla Firefox 4.0	82,30 %
Mozilla Firefox 3.6	61,50 %
Internet Explorer 9	59,64 %
Internet Explorer 8 and older	none

In addition to the native SVG support provided by the browser Adobe Systems developed the "Adobe SVG Viewer" to allow the SVG support for the Internet Explorer 8 and older, as well as in other browsers. More information about the "Adobe SVG Viewer" can be found on the website of Adobe Systems: <http://www.adobe.com/svg/viewer/install/>.

SVG IN SAS 9.2

With SAS 9.2, it is now possible for the first time to create graphics from SAS in the SVG standard. The changes are marginal in comparison to other graphics formats. Only GOPTION DEVICE and, if applicable, the ODS destination must be modified to generate an SVG graphics from an existing SAS program.

For this purpose, five new devices are available in for SAS 9.2 Which one provides the bests results will depend on the type of application. Here is a brief overview of the new devices:

- SVG: Generates SVG 1.1 graphics
- SVGT: Generates SVG 1.1 graphics with transparent background
- SVGZ: Generates compressed SVG 1.1 graphics
- SVGViewer: Generates SVG 1.1 graphics with control elements (multi-page graphics)
- SVGnotip: Generates SVG 1.1 graphics without Tooltip

However, the output of the SVG graphics depends not only on the device used, but also on the ODS destination. The ODS destination affects the features of the graphics files generated by SAS. The valid ODS destinations for SVG are: ODS Listing, ODS HTML, ODS PDF and ODS Printer.

The output of ODS Listing and ODS Printer is in each case a single SVG file. This stand-alone SVG graphics can go over multiple pages.

ODS HTML, in contrast, generates an SVG file and an additional HTML file that embeds the SVG file into HTML. If the SVG graphics goes over several pages a single SVG file is generated for each page.

ODS PDF produces a single PDF file including the SVG graphics. An SVG file is not generated with ODS PDF.

The following examples aim to illustrate some of the possible applications of SVG in SAS.

The code has been reduced, in many cases, to the basic commands and is not necessarily executable in this form.

PhUSE 2011

EXAMPLE 1 - MULTIPLE-PAGE GRAPHICS WITH CONTROL ELEMENTS

```
FILENAME fname "shoes.svg";
OPTIONS GSFMODE=REPLACE GSFNAME=fname DEVICE=SVGVIEW;

PROC GCHART DATA=sashelp.shoes;
  PIE3D product / TYPE=SUM SUMVAR=sales GROUP=subsidiary;
  WHERE subsidiary IN ('London' 'Paris' 'Madrid');
RUN; QUIT;
```

The first two lines of code assign the device and the output filename of the image. Because no other ODS destination is assigned, the ODS listing is used for the output, and therefore uses a single SVG file. DEVICE = SVGVIEW activates the control elements for multi-page SVG graphics.

PROC GCHART generates the actual graphic. Sales are represented per city and split over time and by type of shoe. Since the data from three cities are analyzed (London, Paris and Madrid), the result is a three-sided graphic.

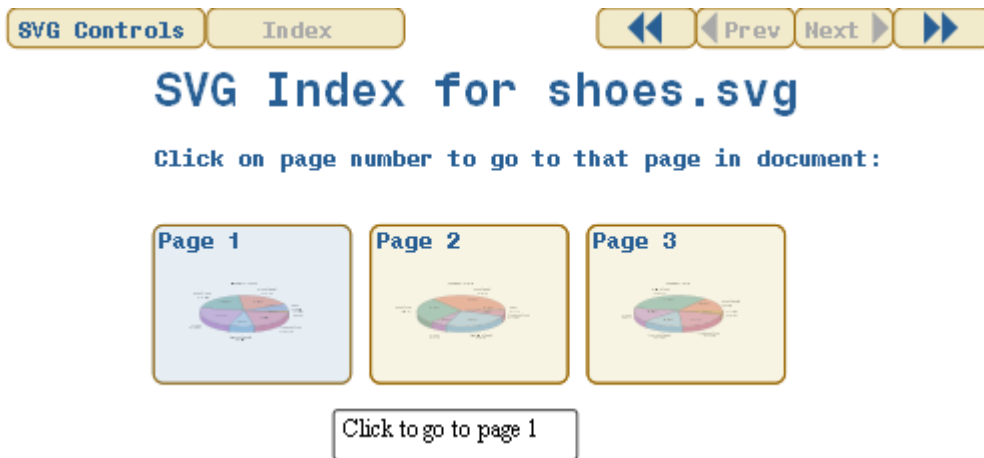


Figure 3.1: Index of a three-sided graphic with control elements

In Figure 3.1, the index of the tripartite SVG graphics is shown in a browser window. By clicking on one of the three pages in the index, the corresponding page will be displayed in full view. The **control elements** in the upper part of the image can be hidden by clicking on "SVG Controls". The control element "index" takes you back to the index view from every single page of the graphic. The arrow controls in the upper right part of the image allow the user to turn over one page at a time or go back to the first and last pages of the graphic. The functionality of the index and control elements can be activated by either using the device SVGVIEW or setting the SAS system option SVGCONTROLBUTTONS with another SVG device.

EXAMPLE 2 - TRANSPARENCY

```

OPTIONS PRINTERPATH=SVGT;
ODS PRINTER FILE="germany.svg";

PROC GMAP DATA=maps.germany MAP=maps.germany;
  ID id;
  CHORO id / COUTLINE=YELLOW DES="";
RUN; QUIT;

PROC GCHART DATA = sashelp.demographics;
  VBAR name / TYPE=SUM SUMVAR=pop DES="";
  WHERE name IN ("GERMANY" "FRANCE");
RUN; QUIT;

ODS PRINTER CLOSE;

```

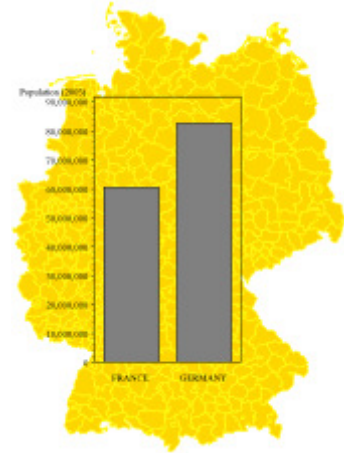


Figure 3.2: Germany map with transparent bar chart

In Example 2, the two images generated will be overlaid with the help of the SVG **transparency** device and the ODS printer. For this purpose, the transparency device is set with PRINTERPATH = SVGT as the output printer, and ODS printer is opened. The functionality of the ODS Printer allows you to create multiple images overlaying each other. The map of Germany drawn with PROC GMAP and the bar chart graph from PROC GCHART is overlapped. As shown in Figure 3.2, the map of Germany is the background to the bar graph. You can see the map of Germany through the transparent background of the bar chart.

EXAMPLE 3 – SVG IN HTML AND PDF

```

GOPTIONS DEVICE=SVG;

ODS LISTING CLOSE;
ODS PDF FILE="stocks.pdf";
ODS HTML FILE="stocks.html";

PROC GCHART DATA = sashelp.stocks;
  VBAR date /DISCRETE SUMVAR = close
  NAME="stocks";
RUN; QUIT;

ODS HTML CLOSE;
ODS PDF CLOSE;
ODS LISTING;

```



Figure 3.3: SVG graphic in HTML (left) and PDF (right)

Up to this point we have limited ourselves to the issue of graphics using ODS listing. Example 3 shows how SVG graphics can be outputted with ODS PDF and ODS HTML. The changes are minor in comparison to ODS LISTING: only an ODS destination with the appropriate file name must be specified. The PDF document generated with ODS PDF is a single page that contains the graphic drawn with PROC GCHART.

However, ODS HTML creates two files a file called "stocks.svg" and another one called "stocks.html". Stocks.svg contains the generated SVG graphic from PROC GCHART, and is complete and executable without the HTML file. The file receives its name from the parameter NAME = "stocks" in the VBAR statement. The second file "stocks.html" generated by SAS contains formatting instructions and embeds "stocks.svg" with the element <EMBED>. The HTML file with a bar graph and a gray background shown in Figure 3.3 is the result.

PhUSE 2011

EXAMPLE 4 - GRAPHICS WITH LINK AND TOOLTIP

```
GOPTIONS DEVICE=SVG;
```

```
ODS LISTING CLOSE;
```

```
ODS HTML FILE="zufriedenheit.htm";
```

```
PROC GCHART DATA = svg1;  
    VBAR3D year / DISCRETE TYPE=SUM SUMVAR=result  
                HTML=html_var DES=" ";
```

```
RUN;
```

```
QUIT;
```

```
ODS HTML CLOSE;
```

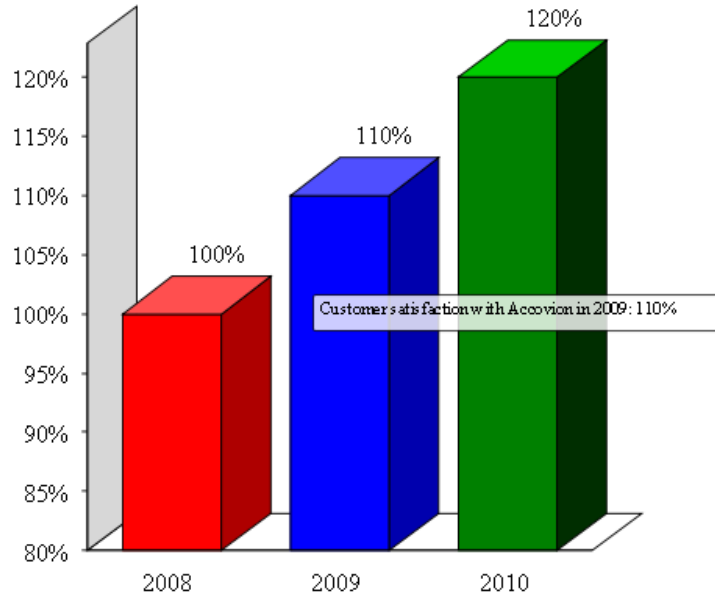


Figure 3.4: Bar chart with tooltip and links

The code presented above will generate a graphic as seen in figure 3.4. The key features of this graphic are displaying a tooltip (text description to a bar) over a bar and linking to other web pages (or documents) when clicking on the bars.

As indicated in the first few lines of code for this example the device SVG with ODS HTML is used in this example. Without ODS HTML, the generated graphic would be unable to display a tooltip or to link. HTML = html_var is the critical parameter for this purpose, that points to a variable in the input data set (html_var in this case). This variable contains the information for the tooltips and links.

year	result	html_var
2008	100%	title="Customer satisfaction with Accovion in 2008: 100%"href="http://www.accovion.com"
2009	110%	title="Customer satisfaction with Accovion in 2009: 110%"href="http://www.accovion.com"
2010	120%	title="Customer satisfaction with Accovion in 2010: 120%"href="document.pdf"

Figure 3.5: Input data set for figure 3.4

The observations shown in figure 3.5 include the data for the bar chart in figure 3.4. The height and the significance of the bars are determined by the variables year and result. The variable html_var includes descriptive information with the parameters **HREF** and **TITLE**. The text assigned to TITLE is displayed as a tooltip over the bar area. HREF contains the link which will be followed when clicking on the bar. Of course, not only a link to Web sites is possible. Links to documents with more detailed statistics for each bar are possible with this application.

EXAMPLE 5 - COMPRESSION

```
GOPTIONS DEVICE=SVG;
PROC GCHART DATA = sashelp.class;
    VBAR name / SUMVAR = height;
RUN; QUIT;

GOPTIONS DEVICE=SVGZ;
PROC GCHART DATA = sashelp.class;
    VBAR name / SUMVAR = height;
RUN; QUIT;
```

Name	Size
gchart.svg	40 KB
gchart.svgz	9 KB

Figure 3.6: SVG graphic, compressed (SVGZ) and uncompressed (SVG)

Example 5 shows the effect of compression on the file size of an SVG file. We produce two equal graphics with ODS listing. The device SVGZ outputs a compressed SVGZ file in the second round. The result is shown in figure 3.6: The size of the compressed file is more than 75% smaller than the uncompressed one. However, it is no longer possible to read or edit the compressed file in a text editor.

OUTLOOK

As seen in the previous chapter, the functionality of SVG implemented in SAS is limited to static graphics. Graphics **with animation or user input** are not supported by SAS.

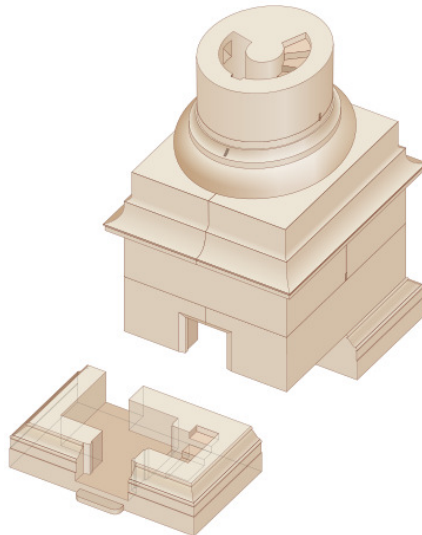


Figure 4.1: Lower portion of Trajan's Column (Rome); dismantled with a mouse click [3]

Figure 4.1 shows the lower part of Trajan's Column in Rome as an SVG graphic. By clicking on the different parts of the column, the user can disassemble the column and free corridors and stairways. It remains to be seen if such an application can ever be created out of SAS.

Another problem for the use of SVG with SAS is the lack of integration in popular **word processing programs**. This problem affects less SAS itself, but it greatly limits the utility of the graphics generated with SAS. The inclusion of SVG graphics in MS Word is only possible by converting the image into a supported image format. To use it in LaTeX you have to convert the graphic to PDF. In Open Office, however, the inclusion of SVG graphics is possible. In order to establish SVG not only as a standard for the Internet, but also to make it a serious alternative in the preparation of documents, SVG would need to catch up with other graphics formats in this point.

REFERENCES

- [1] http://de.wikipedia.org/w/index.php?title=Datei:Zeichen_224.svg
- [2] The source (in German) is <http://www.codedread.com/svg-support.php>. Here also an interpretation of the numbers can be found.
- [3] <http://en.wikipedia.org/wiki/File:Trajans-Column-lower-animated.svg>; Author: Hk kng

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Nicola Tambascia
Accovion GmbH
Helfmann-Park 10
Eschborn / 65760
Work Phone: +49 (6196) 7709 283
Fax: +49 (6196) 7709 115
Email: nicola.tambascia@accovion.com
Web: <http://accovion.com>

Sven Greiner
Accovion GmbH
Helfmann-Park 10
Eschborn / 65760
Work Phone: +49 (6196) 7709 288
Fax: +49 (6196) 7709 115
Email: sven.greiner@accovion.com
Web: <http://accovion.com>

Brand and product names are trademarks of their respective companies.