Scalable Vector Graphics in SAS 9.2

Sven Greiner
Accovion GmbH
Helfmann-Park 10
65760 Eschborn
sven.greiner@accovion.com

Nicola Tambascia
Accovion GmbH
Helfmann-Park 10
65760 Eschborn
nicola.tambascia@accovion.com

Zusammenfassung

Seit gut zehn Jahren ist SVG (Scalable Vector Graphics) ein anerkannter Internetstandard zur Darstellung von vektorbasierten Grafiken. Mit SAS 9.2 ist es jetzt erstmals möglich auch SAS zur Erstellung von Grafiken im SVG-Format zu nutzen. Aus diesem Grund soll der SVG-Standard mit seinen Eigenschaften als Vektorgrafikformat kurz vorgestellt werden. Dazu wird zunächst ein Vergleich zwischen Vektor- und Rastergrafiken angestellt, der die Vor- und Nachteile beider Grafiktypen gegenüberstellt und gleichzeitig die grundlegenden Eigenschaften von SVG darlegt. Auf dieser Grundlage werden die Geschichte und die technische Umsetzung von SVG vorgestellt und anhand von Beispielen erklärt. Schließlich soll mittels einiger Programmbeispiele die tatsächliche Anwendung von SVG in SAS demonstriert werden. Besondere Aufmerksamkeit wird dabei auf den SVG-spezifischen Anwendungsmöglichkeiten liegen, wie dem Einbinden von Tooltips und Links in Grafiken oder dem Erstellen von mehrseitigen Grafiken mit Inhaltsverzeichnis und Steuerungselementen. Abschließend werden einige der nicht von SAS unterstützten Funktionen vorgestellt sowie die Kinderkrankheiten des SVG-Standards besprochen.

Schlüsselwörter: SVG, Scalable Vector Graphics, Grafik, Vektorgrafik

1 Vektor- und Rastergrafiken

Bei der Darstellung einer Grafik am Computer kann zwischen zwei Typen von Grafiken unterschieden werden: Vektorgrafiken und Rastergrafiken. Noch ehe man sich für ein Grafikformat entscheidet, sollte die Vorauswahl auf einen dieser beiden Typen gefallen sein. Da es sich bei „Scalable Vector Graphics“ (SVG) um ein Grafikformat der Vektorgrafiken handelt, sollen an dieser Stelle kurz die Unterschiede zwischen Vektor- und Rastergrafiken erklärt und die daraus resultierenden Vor- und Nachteile besprochen werden.

Rastergrafiken, auch Pixelgrafiken genannt, bestehen aus rasterartig angeordneten Bildpunkten (Pixel). Jedem Bildpunkt wird eine eindeutige Farbe zugeordnet, so dass die Rastergrafik einem Mosaik ähnelt. Die Bildgröße einer Rastergrafik, also die Breite und Höhe in Pixel, bezeichnet man als Auflösung, während man die Differenzierbarkeit der Farbwerte jedes einzelnen Pixels als Farbtiefe bezeichnet. Gängige Formate zur Darstellung von Rastergrafiken am Computer sind u.a. JPEG, Bitmap und GIF.

Im Gegensatz zu Rastergrafiken werden **Vektorgrafiken** nicht in Form von Pixel, sondern als grafische Primitive abgelegt. Vektorgrafiken sind objektorientiert und diese grafischen Primitive (üblicherweise u.a. Linien, Kreise, Polygone und Kurven) werden als Objekte behandelt und entsprechende Attribute (beispielsweise x- und y-Koordinaten, Radius, Farbe, Linienstärke) zugeordnet. Um einen Kreis als Vektorgrafik abzulegen, benötigt man also im einfachsten Fall nur den Mittelpunkt und den Radius des Kreises.
Da die Bildinformationen von Vektorgrafiken in Form von grafischen Primitiven und den dazugehörigen Attributen abgelegt werden, ist zum Anzeigen der Grafiken ein Interpreter notwendig. Diese Aufgabe erfüllt üblicherweise ein Browser oder ein Grafikbearbeitungsprogramm. Weit verbreitete Vektorgrafikformate sind u.a. EPS (Encapsulated PostScript), SWF (Macromedia Flash) und SVG.

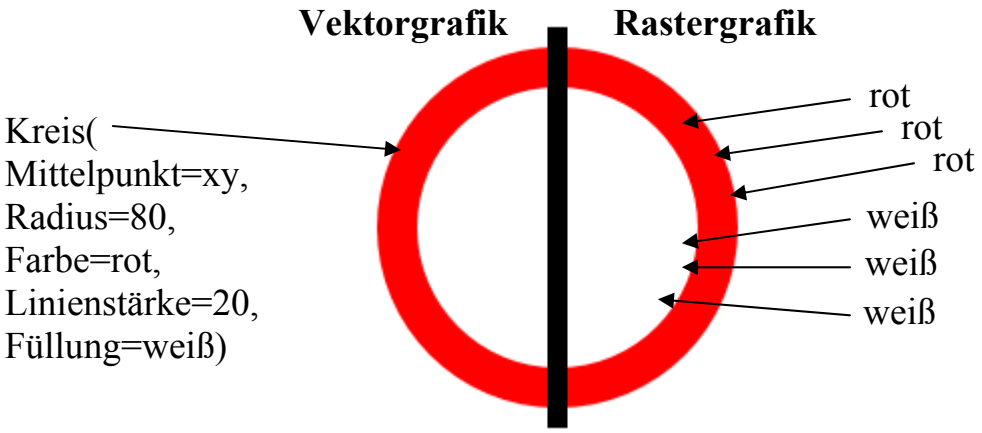

Abbildung 1.1: Darstellung eines Kreises als Vektor- und Rastergrafik

Durch die **Unterschiede zwischen den beiden Grafiktypen** ergeben sich für verschiedene Anwendungsgebiete Vor- und Nachteile. Ist die darzustellende Grafik einfach aus grafischen Primitiven zu erzeugen, ist die Vektorgrafik im Allgemeinen der Rastergrafik vorzuziehen. Wie in Abbildung 1.1 zu sehen, kann ein Kreis als Vektorgrafik mit sehr kleinem Speicherplatzverbrauch und sehr hoher Qualität abgelegt werden. Als Rastergrafik muss trotz der Einfachheit der Grafik für jeden Pixel die Farbinformation einzeln abgelegt werden. Gerade bei großen Bildern kann dies schnell zu einem erheblichen Verbrauch an Speicherplatz führen.

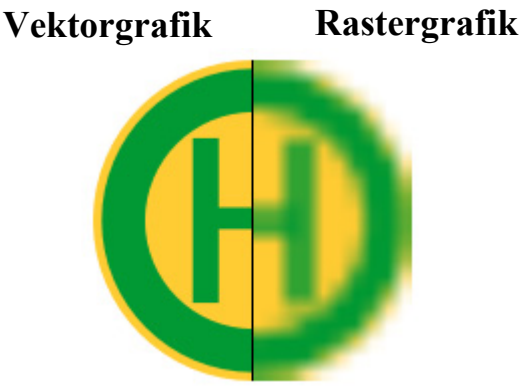

Abbildung 1.2: Hochskalierung einer Vektor- und Rastergrafik im Vergleich [1]
(Vektor- und Rastergrafik vergrößert von 20x20 auf 160x160 Pixel)

Ein weiterer Vorteil von Vektorgrafiken gegenüber Rastergrafiken ist ihre bessere **Skalierbarkeit**. Ist es notwendig die Größe einer Vektorgrafik zu ändern, wird weder beim Vergrößern, noch beim Verkleinern, die Qualität der Darstellung reduziert oder der Speicherplatzverbrauch erhöht. Die grafischen Primitive werden einfach in der entsprechenden Größe neu erzeugt.
Die Skalierung von Rastergrafiken dagegen ist nicht ohne Verlust in der Qualität der Grafik möglich. Wird eine Rastergrafik vergrößert werden neue Pixel erzeugt, die die Farbinformationen der benachbarten Pixel erhalten. Die Bildinformationen werden verfälscht und das Bild wirkt „verwaschen“ (siehe Abbildung 1.2). Außerdem nimmt der Speicherplatzbedarf erheblich zu.
Wird eine Rastergrafik stattdessen verkleinert, werden Pixel gelöscht und Bildinformationen gehen verloren. Teilweise entstehen beim Verkleinern völlig neue Farbtöne, wenn die Farbinformationen aus zwei (oder mehreren) Pixel in einem Pixel zusammengelegt werden, um das zweite (oder alle betreffenden) Pixel löschen zu können.

Die Stärke von Rastergrafiken liegt in der Darstellung von **komplexen Farbverläufen**. Diese findet man meist in Bildern, die mit einer Digitalkamera aufgenommen oder mit einem Scanner digitalisiert wurden. Hier ist es nicht oder nur sehr schwer möglich, die Formen und Farben der Grafik anhand von grafischen Primitiven zu beschreiben. Versucht man dennoch komplexe Farbverläufe als Vektorgrafik darzustellen, wirkt das Ergebnis meist künstlich und unansehnlich.
Aufgrund der Vor- und Nachteile beider Grafiktypen haben sich in unterschiedlichen **Anwendungsgebieten** jeweils andere Grafiktypen durchgesetzt. Prinzipiell werden photorealistische Bilder als Rastergrafik beschrieben, während man überall, wo dies nicht notwendig ist, auch auf Vektorgrafiken zurückgreift. Das betrifft u.a. Grafikanwendungen (vektorbasierte Zeichenprogramme wie Corel-Draw oder Inkscape), Seitenbeschreibungen und Schriften am Computer (PostScript, PDF, TrueType) und Internetanwendungen (Macromedia Flash, SVG).

2 Scalable Vector Graphics (SVG)

Scalable Vector Graphic (SVG) ist der vom World Wide Web Consortium (W3C) empfohlene Standard zur Beschreibung zweidimensionaler Vektorgrafiken im Internet. Entwickelt wurde SVG von Adobe Systems aus der hauseigenen Sprache Precision Graphics Markup Language (PGML) und Microsofts Vector Markup Language (VML). Sowohl PGML als auch VML war zuvor vom W3C abgelehnt worden, ehe SVG als Weiterentwicklung aus beiden Sprachen im Jahr 1999 angenommen wurde. Die momentan empfohlene Version von SVG ist die seit 2001 aktuelle Version 1.1. Version 1.2 ist in Entwicklung.

Adobe Systems stellt SVG als neuen Web-Grafikstandard vor

Unterschleißheim, 18. Februar 1999 – Adobe Systems hat dem World Wide Web Consortium (W3C) einen neuen offenen Standard für Web-Grafiken vorgeschlagen: Mit den hochauflösenden Scalable Vector Graphics (SVG) läßt sich die grafische Qualität von Web-Angeboten entscheidend verbessern. Ein wichtiger Anwendungsbereich sind eCommerce-Websites, denn mit SVG kann die von gedruckten Katalogen, Magazinen und Anzeigen gewohnte Qualität jetzt auch im Web realisiert werden. SVG verleiht Web-Grafiken darüber hinaus dynamische und interaktive Dimensionen. Die Grafiken sind deutlich schneller herunterzuladen, entlasten damit die Bandbreite und optimieren die Browser-Performance. SVG wird im Laufe des Jahres für Endkunden, Web-Publisher und Entwickler verfügbar sein. Mit der Spezifikation der Precision Graphics Markup Language (PGML) hatte Adobe dem W3C bereits einen Web-Standard vorgeschlagen und damit seine zentrale Rolle bei der technologischen Entwicklung im Web-Bereich unterstrichen.

Abbildung 2.1: Offizielle Pressemitteilung von Adobe Systems zur Vorstellung von SVG [2]

Die technische Umsetzung von SVG erfolgt durch verschiedene **Elemente**, die in Form von XML-Code (Extensible Markup Language) abgelegt werden. SVG unterteilt diese Elemente wiederum in drei Gruppen von Grafikobjekten:

- Vektorgrafikelemente
- Rastergrafiken
- Text

Vektorgrafikelemente bezeichnet die Umsetzung der verschiedenen grafischen Primitive in SVG. Zur Beschreibung von Vektorgrafiken stehen folgende Elemente zur Verfügung: Pfad <PATH>, Rechteck <RECT>, Kreis <CIRCLE>, Ellipse <ELLIPSE>, Linie <LINE>, Linienzug <POLYLINE> und Polygon <POLYGON>.
Neben den Vektorgrafikelementen ist es auch möglich **Rastergrafiken** in SVG zu verwenden. In diesem Fall muss die Rastergrafik extern zur Verfügung stehen und kann dann beispielsweise als Hintergrundbild für die Vektorgrafik eingebunden werden. Dies leistet das Element <IMAGE>.
Mit dem Element <TEXT> wird **Text** in die Grafik eingebettet.

Da SVG-Grafiken in XML definiert werden, können diese mit jedem beliebigen Texteditor gelesen und bearbeitet werden. Folgender Code zeigt den Inhalt einer SVG-Grafikdatei mit einigen simplen Text- und Vektorgrafikelementen. Abbildung 2.2 zeigt die aus dem Code erzeugt Grafik.

```
<svg
  xmlns="http://www.w3.org/2000/svg"  version="1.1"
  width="500" height="450">
 <title>Text- und Grafikelemente von 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">
    KSFE 2011
 </text>
</svg>
```

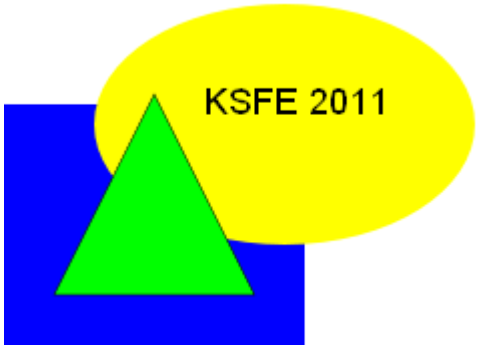

Abbildung 2.2: Text- und Grafikelemente von SVG

Der Code beginnt mit dem **Wurzelelement <SVG>**, das den gesamten Definitionsbereich der SVG-Grafik umschließt. Die Attribute XMLNS (XML Name Space) und VERSION spezifizieren, welcher Standard zur Interpretation des folgenden Codes genutzt wird und in welcher Version. In diesem Fall wird der SVG-Standard des W3C in der Version 1.1 verwendet. Die Attribute WIDTH und HEIGHT legen die Größe des Ausgabebereichs fest, in dem die Grafik angezeigt werden kann.
Es folgt das Element <TITLE>, mit dem der Text festgelegt wird, der als Überschrift im Fenster mit der Grafik erscheint.
Die Elemente <LINE>, <RECT>, <CIRCLE> und <TEXT> definieren mit ihren diversen Attributen die in Abbildung 2.2 sichtbaren Formen und Text.
Natürlich sind die in Abbildung 2.2 dargestellten Formen nur ein sehr simpler Anwendungsfall für SVG. SVG unterstützt auch **interaktive und dynamische** Anwendungen, d.h. es kann auch auf Benutzereingaben reagieren und Animationen darstellen. Wegen dieser Flexibilität und der gleichzeitig effizienten Arbeitsweise hat sich SVG auf vielen tragbaren Geräten etabliert. Hier werden üblicherweise reduzierte Versionen von SVG wie SVG Basic oder SVG Tiny genutzt.

SVG wurde in erster Linie für die Darstellung von Grafiken im Internet entworfen. Allerdings ist die **Browser-Unterstützung** für SVG nicht so weit fortgeschritten, dass man ohne weiteres von einer korrekten Darstellung im eigenen Browser ausgehen kann. Zu beachten ist an diesem Punkt vor allem die fehlende Unterstützung des Internet Ex-

plorers für SVG. Erst in der noch nicht finalen Version 9 des Internet Explorers wird eine Unterstützung implementiert sein. Tabelle 2.1 bietet eine Übersicht über die Unterstützung von SVG in verschiedenen Browsern:

Tabelle 2.1: Browser-Unterstützung für SVG 1.1 [3]

Browser	SVG 1.1 Unterstützung
Opera 10.61	95,26 %
Safari 5.0	82,48 %
Google Chrome 4.0	82,12 %
Mozilla Firefox 4.0 Nightly	78,83 %
Mozilla Firefox 3.6	61,50 %
Internet Explorer 9 Preview 4	58,00 %
Internet Explorer bis Version 8	keine

Neben der nativen SVG-Unterstützung durch den Browser wurde von Adobe Systems der „Adobe SVG Viewer“ entwickelt, der SVG-Unterstützung auch für den Internet Explorer und andere Browser ermöglicht. Weitere Informationen zum „Adobe SVG Viewer“ können auf der Webseite von Adobe Systems gefunden werden: *http://www.adobe.com/svg/viewer/install/*.

3 SVG in SAS 9.2

Mit SAS 9.2 ist es jetzt zum ersten Mal möglich, aus SAS heraus Grafiken zu erstellen, die dem SVG Standard entsprechen. Die Änderungen im Vergleich zu anderen Grafikformaten sind dabei marginal: Nur die GOPTION DEVICE und gegebenenfalls die ODS Destination müssen geändert werden, um aus einem bestehenden SAS-Grafikprogramm eine SVG-Grafik zu erzeugen.

Für diesen Zweck stehen seit SAS 9.2 fünf neue **Devices** zur Erstellung von SVG-Grafiken zur Verfügung. Welche davon das optimale Resultat liefert, hängt von der Art der Anwendung ab. Hier eine kurze Übersicht über die neuen Devices:

- SVG: Erzeugt SVG 1.1 Grafiken
- SVGT: Erzeugt SVG 1.1 Grafiken mit transparentem Hintergrund
- SVGZ: Erzeugt komprimierte SVG 1.1 Grafiken
- SVGVIEW: Erzeugt SVG 1.1 Grafiken mit Steuerungselementen (mehrseitige Grafiken)
- SVGnotip: Erzeugt SVG 1.1 Grafiken ohne Tooltip

Die Ausgabe der SVG-Grafiken hängt aber nicht alleine vom benutzten Device ab. Je nach verwendeter ODS Destination unterscheiden sich die Merkmale der von SAS erzeugten Grafikdateien. Die für SVG gültigen ODS Destinations sind: ODS Listing, ODS HTML, ODS PDF und ODS Printer.

Die Ausgabe von ODS Listing und ODS Printer ist in jedem Fall eine einzelne SVG-Datei. Diese alleinstehende SVG-Grafik kann auch über mehrere Seiten gehen.
ODS HTML dagegen generiert eine SVG-Datei und zusätzlich eine HTML-Datei, die die SVG-Datei in HTML einbettet. Geht die SVG-Grafik über mehrere Seiten wird für jede Seite eine einzelne SVG-Datei erzeugt.
ODS PDF produziert eine einzelne PDF-Datei, in der die SVG-Grafik enthalten ist. Eine SVG-Datei wird mit ODS PDF nicht erzeugt.

Um einige der gerade beschriebenen Anwendungsmöglichkeiten besser zu verdeutlichen und das Thema SVG in SAS etwas weiter zu vertiefen, folgen jetzt einige Beispiele zu SVG in SAS. Der Code wurde in vielen Fällen auf die grundlegenden Befehle reduziert und ist in dieser Form nicht zwangsläufig lauffähig.

Beispiel 1 – Mehrseitige Grafik mit Steuerungselementen

```
FILENAME fname "shoes.svg";
GOPTIONS 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;
```

Die ersten beiden Zeilen des Codes legen den Device und den Ausgabenamen der Grafikdatei fest. Weil in diesem Beispiel keine andere ODS Destination festgelegt ist, wird für die Ausgabe ODS Listing und damit eine einzelne SVG-Datei verwendet. DEVICE=SVGVIEW aktiviert die Steuerungselemente für mehrseitige SVG-Grafiken.

PROC GCHART erzeugt die eigentliche Grafik. Pro Stadt werden Verkäufe über die Zeit, aufgeschlüsselt nach Schuhtyp, dargestellt. Da die Daten von drei Städten (London, Paris und Madrid) betrachtet werden, ist das Ergebnis eine dreiseitige Grafik.

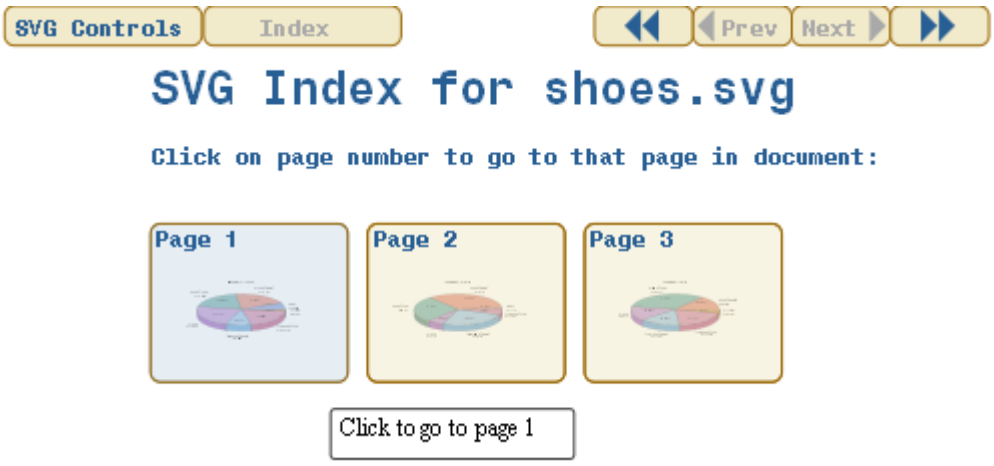

Abbildung 3.1: Index einer dreiseitigen Grafik mit Steuerungselementen

In Abbildung 3.1 wird der Index der dreiseitigen SVG-Grafik in einem Browserfenster gezeigt. Durch das Klicken auf eine der drei Seiten im Index erhält man die entsprechende Seite in Großansicht. Die **Steuerungselemente** im oberen Teil des Bildes können durch ein Klicken auf „SVG Controls“ ausgeblendet werden. Das Steuerungselement „Index“ führt von jeder Einzelseite der Grafik wieder zurück zu dieser Indexansicht. Die Pfeilsteuerungselemente im rechten Teil des Bildes lassen den Anwender eine Seite nach der anderen, bzw. die erste und letzte Seite der Grafik ansteuern.
Aktiviert werden kann die Funktionalität von Index und Steuerungselementen entweder durch die Device SVGView oder das Setzen der SAS-Systemoption SVGCONTROLBUTTONS mit einem anderen SVG-Device.

Beispiel 2 - Transparenz

```
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;
```

Abbildung 3.2: Deutschlandkarte mit transparentem Balkendiagramm

In Beispiel 2 werden zwei Grafiken erzeugt, die mit Hilfe von SVG **Transparenz**-Device und ODS Printer übereinander gelegt werden sollen. Zu diesem Zweck wird mit PRINTERPATH=SVGT der Transparenz-Device als Ausgabe-Printer gesetzt und ODS Printer geöffnet. Die Funktionalität von ODS Printer erlaubt es, mehrere Grafiken übereinander zu legen.
Die von PROC GMAP gezeichnete Karte von Deutschland und das Balkendiagramm aus PROC GCHART werden übereinander gelegt. Wie in Abbildung 3.2 zu sehen ist, bildet die Deutschlandkarte den Hintergrund zum Balkendiagramm. Durch die Transparenz kann man durch den Hintergrund des Balkendiagramms auf die Deutschlandkarte sehen.

Beispiel 3 – SVG in HTML und 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;
```

Abbildung 3.3: SVG-Grafik in HTML (links) und PDF (rechts)

Bis zu diesem Zeitpunkt haben wir uns auf bei der Ausgabe von Grafiken auf ODS Listing beschränkt. Beispiel 3 zeigt wie SVG-Grafiken auch mit ODS PDF und ODS HTML ausgegeben werden können. Die Änderungen zu ODS Listing sind geringfügig: lediglich die entsprechende ODS Destination mit Dateiname muss angegeben werden.
Das von ODS PDF generierte PDF-Dokument besteht aus einer einzelnen Seite, die die von PROC GCHART gezeichnete Grafik beinhaltet.
ODS HTML dagegen erzeugt zwei Dateien: eine Datei namens „stocks.svg“ und eine namens „stocks.html“. Die erste Datei „stocks.svg“ enthält die aus PROC GCHART erzeugte SVG-Grafik, die auch ohne die HTML-Datei vollständig und lauffähig ist. Ihren Namen erhält die Datei vom Parameter NAME=“stocks“ im VBAR Statement. Die zweite Datei „stocks.html“ beinhaltet von SAS erzeugte Formatierungsanweisungen und bindet „stocks.svg“ über das Element <EMBED> ein. Das Ergebnis ist die in Abbildung 3.3 dargestellte HTML-Datei mit Balkendiagramm und grauem Hintergrund.

Beispiel 4 – Grafik mit Link und 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;
```

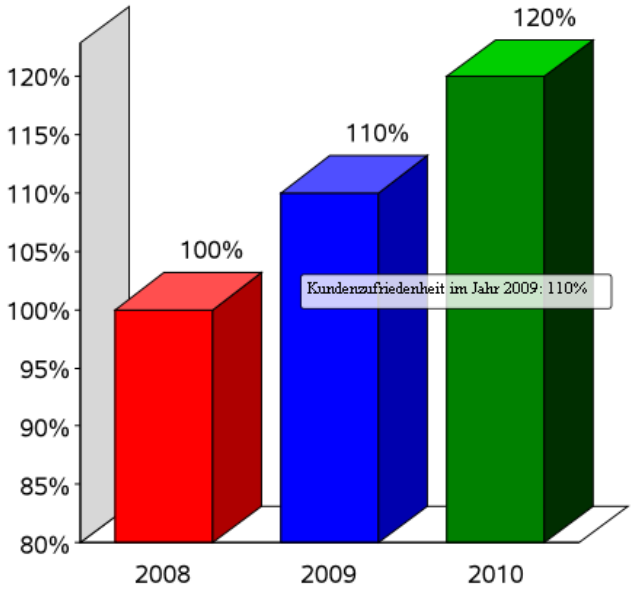

Abbildung 3.4: Balkendiagramm mit Tooltip und Links

Der angegebene Code erzeugt eine Grafik wie in Abbildung 3.4 zu sehen. Die entscheidenden Merkmale dieser Grafik sind das Anzeigen eines **Tooltips** (Beschreibungstext zu einem Balken) über den Balkenflächen und das **Verlinken** zu anderen Internetseiten (bzw. Dokumenten) beim Klicken auf die Balkenflächen.
Wie in den ersten Zeilen des Codes angegeben, wird für dieses Beispiel als Device SVG mit ODS HTML verwendet. Ohne ODS HTML kann die erzeugte Grafik weder einen Tooltip anzeigen, noch auf ein anderes Ziel verlinken. Der kritische Parameter für diesen Zweck ist HTML=html_var, der auf eine Variable im Eingangsdatensatz verweist (in diesem Fall html_var). Diese Variable enthält die Informationen zu Tooltip und Links.

Obs	year	result	html_var
1	2008	100%	title="Kundenzufriedenheit im Jahr 2008: 100%"href="http://www.accovion.com"
2	2009	110%	title="Kundenzufriedenheit im Jahr 2009: 110%"href="http://www.accovion.com"
3	2010	120%	title="Kundenzufriedenheit im Jahr 2010: 120%"href="textdokument.pdf"

Abbildung 3.5: Eingangsdatensatz zu Abbildung 3.4

Die in Abbildung 3.5 zu sehenden Beobachtungen enthalten die Daten zum Balkendiagramm in Abbildung 3.4. Höhe und Aussage der Balken werden durch die Variablen year und result bestimmt. Die Variable html_var dagegen beinhaltet mit den Parametern **TITLE und HREF** beschreibende Informationen. TITLE wird ein Text übergeben, der als Tooltip über der Balkenfläche angezeigt wird. HREF übergibt dem Balken einen Link, dem beim Klicken auf die Balkenoberfläche gefolgt wird. Natürlich ist hier nicht nur das Verlinken von Webseiten möglich. Auch Dokumente mit ins Detail gehenden Statistiken können für jeden einzelnen Balken eingebunden werden.

Beispiel 5 – Kompression

```
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;
```

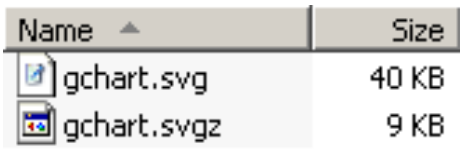

Abbildung 3.6: SVG-Grafik, komprimiert (SVGZ) und unkomprimiert (SVG)

Beispiel 5 zeigt den Effekt der Kompression auf die Dateigröße einer SVG-Datei. Wir erzeugen zweimal dieselbe Grafik mit ODS Listing. Anstelle von Device SVG wird beim zweiten Lauf mit Device SVGZ eine komprimierte SVGZ-Datei ausgegeben. Das Ergebnis ist in Abbildung 3.6 zu sehen: Die Größe der komprimierten Datei ist um mehr als 75% geringer als die der unkomprimierten. Allerdings ist es nicht mehr möglich, die komprimierte Datei in einem Texteditor sinnvoll zu lesen oder zu editieren.

4 Ausblick

Wie im vorigen Kapitel zu sehen, ist die in SAS umgesetzte Funktionalität von SVG bisher auf statische Grafiken beschränkt. Grafiken, die **Animation oder Benutzereingabe** erfordern, werden von SAS nicht unterstützt.

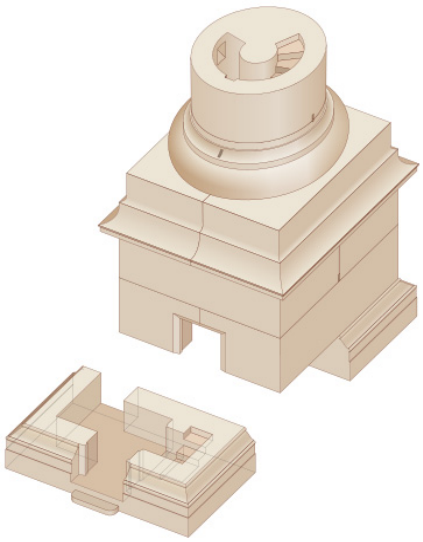

Abbildung 4.1: Unterer Teil der Trajanssäule (Rom); per Mausklick zerlegbar [4]

Abbildung 4.1 zeigt den unteren Teil der Trajanssäule in Rom als SVG-Grafik. Durch Klicken auf die einzelnen Teile der Säule kann der Nutzer die Säule zerlegen und so Gänge und Treppen freilegen. Ob eine solche Anwendung jemals aus SAS heraus erzeugt werden kann, bleibt abzuwarten.

Ein weiteres Problem für die Nutzung von SVG mit SAS ist die mangelnde Integrierbarkeit in gängige **Textverarbeitungsprogramme**. Dieses Problem betrifft weniger SAS selbst, als dass es die Nützlichkeit der mit SAS erzeugten Grafiken stark einschränkt. So können SVG-Grafiken in MS Word nur über das Umwandeln der Grafik in ein unterstütztes Grafikformat eingebunden werden. Auch in LaTeX kann nur der Umweg über eine Umwandlung in ein PDF gewählt werden. In Open Office dagegen ist das Einbinden der SVG-Grafiken einfach und reibungslos möglich. Um SVG nicht nur als Standard für das Internet zu etablieren, sondern auch zu einer ernsthaften Alternative in der Dokumentenerstellung zu machen, müsste SVG an diesem Punkt mit anderen Grafikformaten gleichziehen.

Literatur

[1] http://de.wikipedia.org/w/index.php?title=Datei:Zeichen_224.svg

[2] http://www.adobe.com/de/aboutadobe/pressroom/pr/feb99/adbsvg.pdf

[3] Die Quelle ist http://www.codedread.com/svg-support.php. Hier ist auch eine Interpretation der Zahlen zu finden.

[4] http://de.wikipedia.org/w/index.php?title=Datei:Trajans-Column-lower-animated.svg; Urheber: Hk kng