Data Visualization Using Java and VRML

Lingxiao Li, Art Barnes, SAS Institute Inc., Cary, NC

ABSTRACT

Java and VRML (Virtual Reality Modeling Language) are tools with tremendous potential for creating two- and three-dimensional visual representations of data and sharing those representations over a network. Visualization software written in Java can run on any platform where the JVM (Java Virtual Machine) is supported, either as a stand-alone application or as an applet embedded in a Web page. By using VRML as the output format for data visualization, the results can be easily shared over the web and have access to the rich graphical capabilities that VRML provides.

This presentation will outline ways you can use the new Java charting components in your custom data visualization applications. It will also describe a new batch procedure that uses VRML technology for web-enabled data visualization.

INTRODUCTION

Data visualization software is used to create visual representations of data that can help you reveal patterns and trends in data. As more and more companies start using Internet as their information delivery channel, there is an increasing demand for web-enabled data visualization tools. Because of its network-centric and platform-neutral features, Java has become the programming language of choice for dealing with Internet applications. SAS Institute has developed some new Java-based, thin client solutions to address the needs for data visualization in the networked environment.

As part of the efforts to provide Java client access to the full power and functionality of SAS® software, we have developed new Java charting components to visualize data accessible from a SAS server. These charting components are highly interactive and flexible. They can be used to view multiple dependent variables and up to three independent variables in the data at a time. Independent variables support sorting operations and dependent variables support basic statistics such as mean, sum, and frequency. These components provide probe, drill down, and some other interactive capabilities. You can display each chart in 2D view or 2.5D view (2D visualization with a 3D look). The input to these components is a Java interface to two-dimensional data that is commonly referred to as rectangular or tabular data. Rectangular data is composed of columns and rows. The intersection of a column with a row is known as a cell. Examples of rectangular data include SAS data sets, ODBC and JDBC databases, and spreadsheets. The interface includes methods for retrieving the number and type of columns, the number and type of rows, and the type and value of cells. In the subsequent discussions, we will follow terminology of SAS data set when we describe the input data.

Another data visualization solution that SAS Institute has developed is a set of web publishing tools based on the VRML technology. VRML is a modeling language that can be used to distribute 3D interactive graphics on the World Wide Web. A new GGRAF procedure is included in the SAS/INTRNET™ software that enables you to generate 3D VRML charts on the SAS server. You can view these charts with a VRML browser plug-in in your web browser. SAS Institute has also developed a VRML-2.0 Browser component that is written in 100% Pure Java and can be used on any platform with a JVM.

In this presentation, we will introduce the features of the Java charting components, Java VRML Browser, and PROC GGRAF. We will also demonstrate how these tools can be used to build data visualization applications.

JAVA CHARTING COMPONENTS

In this section we will outline the features of the Java charting components currently being developed by SAS Institute. We will also demonstrate how to use these components with sample Java code.

Bar Chart

The Bar Chart component allows you to build a visual representation of one or two independent (classifier) variables and one or two dependent (response) variables

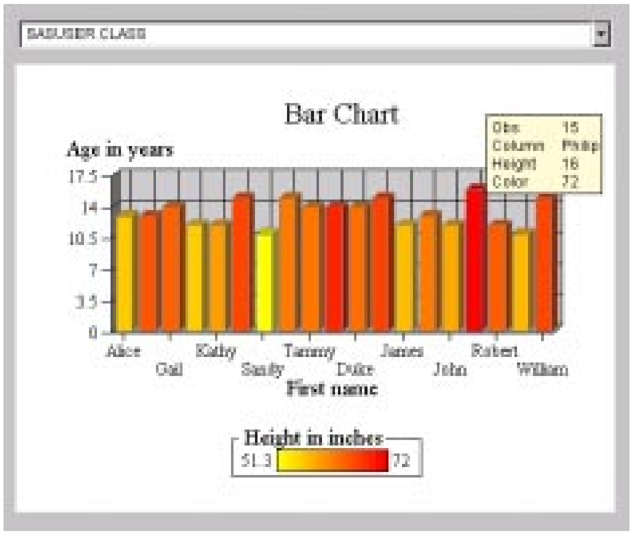


Figure 1. Bar Chart.

in a data set. Each unique observation in a data set corresponds to a bar in the chart. The independent variables are Column and Group that control the location and grouping of the bars. Column or Group can only be assigned to categorical data. The dependent variables are Height and Color. They control the height and color of the bars and must be assigned to numerical data. You can choose to display the bar chart in a vertical or horizontal orientation. You can also select a variety of geometry shapes for the bars.

Figure 1 shows an example of a Bar Chart. In this case, the Column variable is attached to “First Name” and Height variable is attached to “Age in years." The Color variable is attached to “Height in inches." The GROUP variable is undefined. Therefore, the bars are not grouped. The legend of the color range is shown at the bottom. The figure also shows the probe information displayed when the mouse is over one of the bars.

Segmented Bar Chart

The Segmented Bar Chart (Bar Chart with Subgroups) component supports Column, Segment, and Group as independent variables. The dependent variable in this chart is Height which controls the height of each segment in the stacked bar. As in the case of the bar chart component, the independent variables can only be assigned to categorical data and the dependent variable must be assigned to numerical data. You can choose to display the chart in a vertical or horizontal orientation, and use different geometry shapes for the bars.

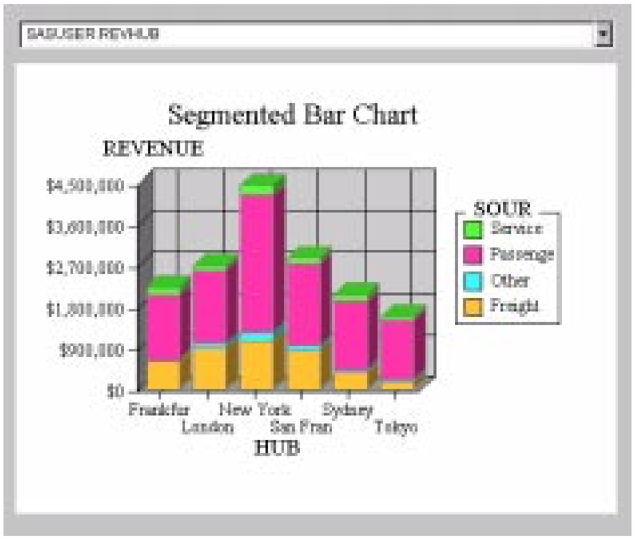


Figure 2. Segmented Bar Chart.

The chart in Figure 2 is an example of a Segmented Bar Chart. In this case, the Column variable is attached to “HUB” and the Height variable is attached to “REVENUE”. The Segment variable is defined to be “SOURCE”. The legend of Segment variable is shown on the right of the chart. You can drag the mouse button to scroll the legend values in case there is not enough room to fit all values in the user-specified legend area.

Scatter Chart

The Scatter Chart component allows you to build a scatter plot of observations using Column as the independent variable and Height, Color as the dependent variables. Unlike the two charts described above, Column variable can be assigned to either categorical or numerical data. You can choose different geometry shapes for the markers. You can also connect markers with joins.

Figure 3 shows an example of a Scatter Chart. The Column variable is assigned to be “DATE”. The Height variable is assigned to be “Retail Sales”. The chart also shows “YEAR” as the color of each marker.

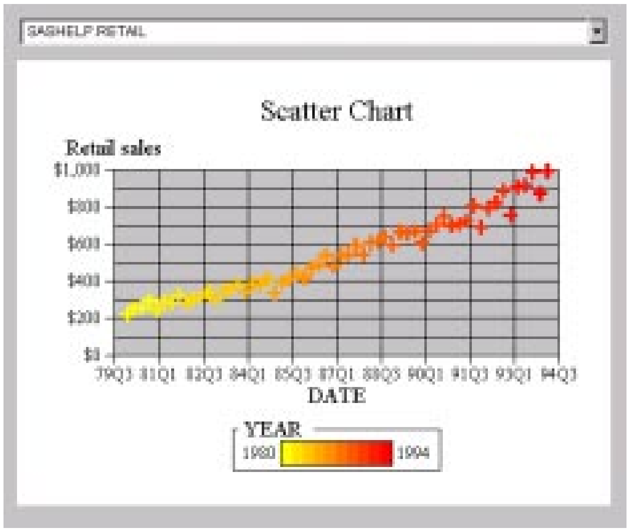


Figure 3. Scatter Chart.

Overlay Chart

The Overlay Chart component allows you to build multiple plots of the data using Column as the independent variable and Height as an array of dependent variables. It is ideal for comparing several patterns in the same data source. Each variable of the Height variable array is attached to a user-specified column in a data set. Therefore, multiple columns in a data set can be shown simultaneously as overlaying plots with different identifying colors. As in the case of the Scatter Chart, the Column variable can be either categorical or numerical. You can choose to display the plots as lines or scattered markers.

An example of an Overlay Chart is shown in Figure 4. In this case, the Column variable is defined as “First name” and the Height variable array is assigned to the columns labeled as “Age in years”, “Height in inches”, and “Weight in pounds” in the data.

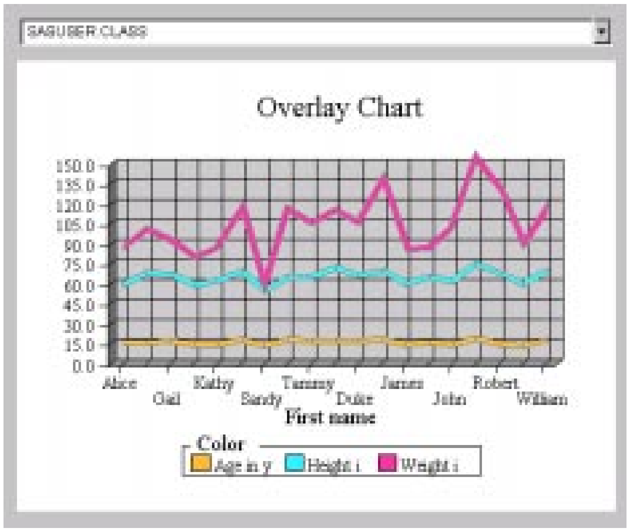


Figure 4. Overlay Chart.

Pie Chart

The Pie Chart component supports Sector as independent variable and Weight as dependent variable. Each category of the Sector variable is displayed as an instance of a pie slice and the value of the Weight variable controls the angular size of the pie slice. You can specify the maximum number of slices to be displayed so that any entry beyond that will fall into the "Other" category.

Figure 5 shows an example of a Pie Chart. In the example, the Sector variable is attached to “STYLE” and the Weight variable is attached to “SQFEET”. The Mean is taken for the Weight variable.

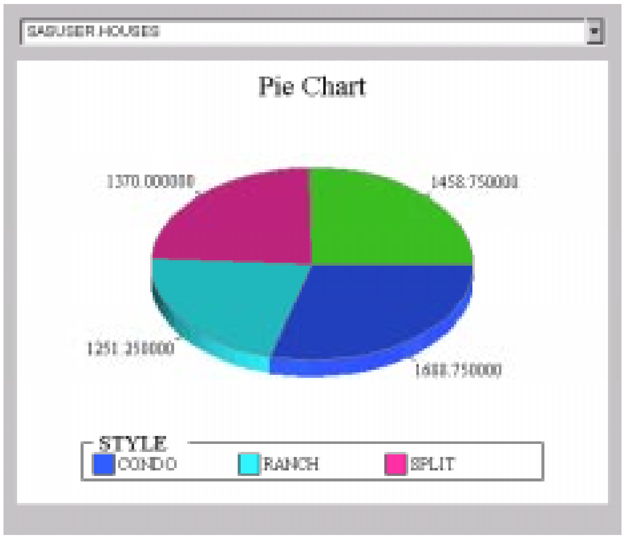
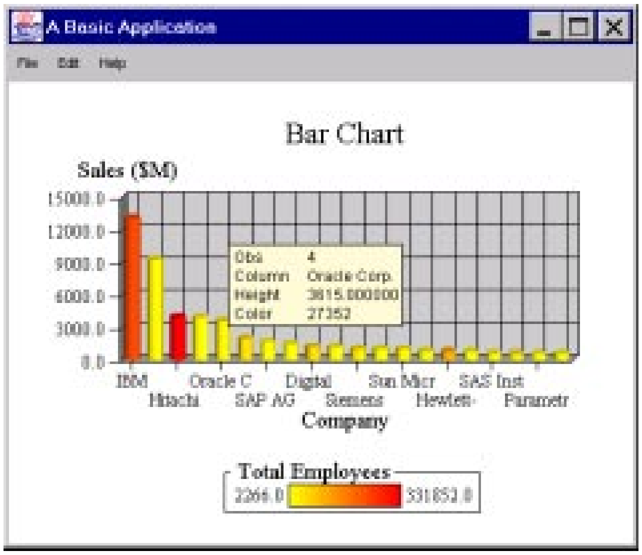


Figure 5. Pie Chart.

How to Use Java Charting Components

The Java charting components are light-weight visual components implemented as JavaBeans. They support JavaBeans property editors, customizers, and serialization. They provide a rich set of properties and methods that allow you to choose the desired appearance and behavior. The components have been integrated into SAS Institute Inc.’s Java Integrated Development Environment (IDE). If you are building your application with the IDE software, you can use drag-and-drop to create these charts and set their properties, such as how the variables are attached to the data, through the property editors and customizers.

You can also directly use these Java classes along with other SAS Java tools to build your Java application. The Java package that contains the chart classes is *com.sas.graphics.chart*. The input to these charts is a rectangular data that implements the required interface *StaticTableInterface.* SAS Institute has developed Java tools to deliver rectangular data from a SAS server that implements *StaticTableInterface* interface. In the following example, we demonstrate how to create a chart in a Java applet or application. The example is for illustration purpose only. For detailed information on the charting components and the Java classes that supply the required interfaces to SAS data, please refer to the online documentation for the IDE software.



Obs 4
Column Oracle Corp
Height 3815.000000
Color 27352

Figure 6. "SoftwareSale" Bar Chart.

Suppose you want to create a Bar Chart from a data source, "SoftwareSales", that implements the required interfaces and contains three columns: "Company", "Sales ($M)", and "Total Employees". The data is made up from Software Magazine’s software industry ranking (“The 1997 Software 500”). In the chart, you want "Company" assigned to the Column variable, "Sales ($M)" assigned to the Height variable, and "Total Employees" assigned to the Color variable. You also want to use cylinder as bar shape. In your Java applet or application, you can use the following code segments:

Step 1: Construct a Bar Chart Object

```
Bar bar = new Bar();
```

Step 2: Assign Data to Variables

```
// Attach the column labeled as "Company" to the Column
// variable.
IndependentVariable iv1 = new IndependentVariable();
iv1.setName("Company");
bar.setColumnVariable(iv1);

// Attach the column labeled as "Sales ($M)" to the Height
// variable.
DependentVariable dv1 = new DependentVariable();
dv1.setName ("Sales ($M)");
bar.setHeightVariable(dv1);

// Attach the column labeled as "Total Employees" to the
// Color variable.
DependentVariable dv2 = new DependentVariable();
dv2.setName ("Total Employees");
bar.setColorVariable(dv2);
```

Step 3: Set Desired Properties

```
// Use cylinder as the geometry shapes of bars.
bar.setGeometryType(Bar.CYLINDER_GEOMETRY_TYPE);
```

Step 4: Attach Data to the Chart

```
// "SoftwareSales" is an instance ofStaticTableInterface
bar.attachModel(SoftwareSales);
```

The chart created by the above code is shown in Figure 6.

WEB PUBLISHING WITH VRML

In this section, we will introduce new web publishing tools that use VRML technology to distribute data visualization graphics on the web.

Java VRML Browser

We have developed a Java VRML browser component that can be embedded in a Java application or applet to view VRML contents such as the graphical output generated from PROC GGRAF. The browser is written in 100% Pure Java and supports the VRML 2.0 specification. There are two implementations of the browser that support JDK 1.02 and JDK 1.1.

The JViewer application that ships with SAS/INTRNET uses PROC GGRAF to create VRML charts on the remote SAS server. The application then downloads the VRML file by remotely submitting PROC DOWNLOAD of SAS/CONNECT® software using JCONNECT of SAS/INTRNET software. The byte stream returned from the download is then fed into the browser component for display. Once the graphics are displayed, you can examine the data from different angles, zoom-in and zoom-out, and get detailed information about specific items. You can perform various drill down tasks with a mouse click. The browser component also provides a set of properties and methods that allow you to select navigation type, light intensity, projection type, etc.

As a Java applet the VRML browser can be embedded into an HTML page. Currently two browser applets are provided, one with user controllable settings (Figure 7) and one without.

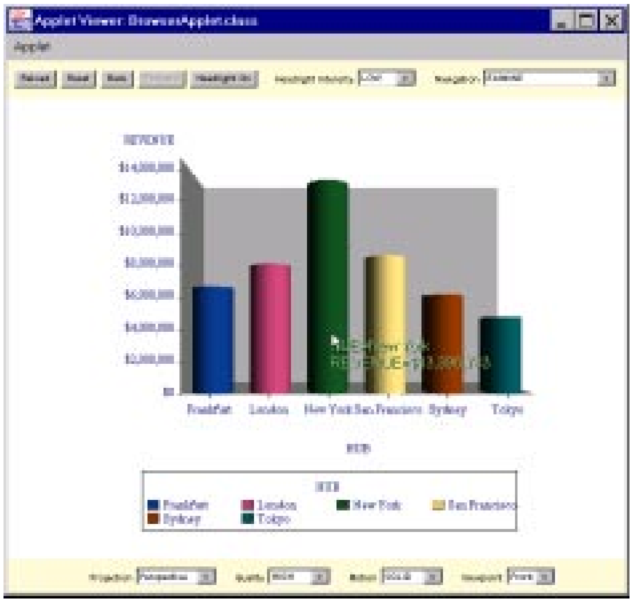


Figure 7. PROC GGRAF output.

Figure 7 shows the output of a PROC GGRAF chart displayed in the VRML applet with user controllable settings.

PROC GGRAF

PROC GGRAF is an experimental procedural interface to SAS/GRAPH® software that enables SAS programmers to create 3D clickable charts in VRML. Using VRML as the output format, the charts can be easily distributed on the web. You can also take full advantage of the rich graphical capabilities that VRML offers when you view these charts. Some of the features that make this PROC unique for web enabled graphics are detailed below. See the reference section for the Web page that contains the full usage documentation for PROC GGRAF.

Clickable Charts

The GGRAF procedure can dynamically set the URL information inside the VRML file based on data you provide. This feature is supported by a SAS macro variable and is used to describe the data substitution that occurs during the PROC execution.

Syntax:

```
%let BASEURL=http//your-server-name/%nrstr({&name | &value |
   [n] | [f]}).html
```

BASEURL
The macro variable that PROC GGRAF uses to define the URL for each chart element.

%nstr
This prevents the macro facility from replacing the name value pairs inside the parentheses.

&name

The variable name from the current chart data and can be one of the following values.

G_DEP	- dependent name
G_INDEP	- independent name
G_SUBGR	- subgroup name
G_GROUP	- group name

&value

The variable value from the current chart data and can be one of the following values.

G_DEPV	- dependent value
G_INDEPV	- independent value
G_SUBGRV	- subgroup value
G_GROUPV	- group value

n (optional)

This prevents the name value pair from being URL encoded. If n is omitted, then encoding occurs. The encoding allows certain characters to be transmitted correctly by the http protocol.

f (optional)

This instructs that the formatted value be inserted if the data variable has a format associated with it.

Examples:

1. One possible usage is to allow the BASEURL macro to describe a link setting of a static web page.

```
 %let BASEURL=http://webserver/%nrstr({&G_INDEPV}).html

    /* generate the VRML graph in Figure 7. */
proc ggraf data=sashelp.revhub;
   vcylin hub / dep=revenue;
run;
```

This, in turn, will generate the URL string for each independent variable in the VRML output file. Selecting the bar for “New York” in Figure 7 would bring up the following link.

```
http://webserver/New+York.html
```

2. Another feature is the ability to exploit the data and compute services of the SAS/INTRNET product. The SAS/INTRNET htmSQL data service allows you to incorporate data into a Web page using SQL. To process an htmSQL query based on a currently selected independent value, the following BASEURL would be defined.

```
%let BASEURL =
http://webserver/bin/htmSQL/sales/where.hsql?%ntstr(where={
&G_INDEP}%3D%27{&G_INDEPV}%27)

   /* generate the VRML graph in Figure 7. */
proc ggraf data=sashelp.revhub;
   vcylin hub / dep=revenue;
run;
```

If the currently highlighted element in Figure 7 was selected, the htmSQL query could run to provide a sales report for the New York hub.

```
http://webserver/bin/htmSQL/sales/where.hsql?
where=HUB%3D%27New+York%27
```

Visual Control

PROC GGRAF supports options that control the visual behavior of the graph during display and will allow you to exploit specific features of the VRML specifications.

PROC GGRAF *options-list;*

options list -

URL | NOURL

Indicates that a URL macro variable, BASEURL, is provided which contains the URL that is to be included in the VRML output file. The default is URL.

INFO | NOINFO

Specifies whether to provide data feedback information for the graph components. When you specify a value of INFO, you enable the data value to display when your cursor passes over a bar or pie slice. The default is INFO. An example of the data information is shown in Figure 7. (HUB=New York REVENUE=$13,396,783)

LAUNCH | NOLAUNCH

Specifies whether to launch a new browser or to update the existing browser with each click. The default value is NOLAUNCH

CONCLUSION

The Java-based thin client and VRML output solutions described in this presentation are powerful and effective tools for developing web-enabled data visualization applications. Java-based tools can help you create interactive models that have an improved capability for revealing patterns and trends in data. In the near future, we plan to add more chart types to the collection of Java charting components. We also plan to enhance the visualization models of these charts with three-dimensional graphics. For example, you can define a Row variable in the Bar Chart to make it a 3D Bar Chart. This will also allow you to view the charts from different perspectives.

The Java charting components will be shipped with SAS Institute Inc.’s Java IDE software. The Java VRML Browser component and PROC GGRAF have shipped with SAS/INTRNET software release 1.1. They are demonstrated in the sample applet, JViewer, of SAS/INTRNET software, and web publishing tools.

REFERENCES

”The 1997 Software 500,” (1997), *www.sentrytech.com/index500.htm*, The Sentry Group, MA, USA.

SAS/INTRNET software, *www.sas.com/rnd/web/* - Use this URL to review the VRML and PROC GGRAF usage documentation and examples.

ACKNOWLEDGMENTS

The authors would like to thank Jyoti Chaudhry, Jim Begley, and Dana Marnell for their contributions to the projects presented in this paper.

Lingxiao Li
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27512
Phone: 919-677-8000 ext. 5858
Email: saslol@sas.com

Art Barnes
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27512
Phone: 919-677-8000 ext. 6863
Email: sasafb@sas.com