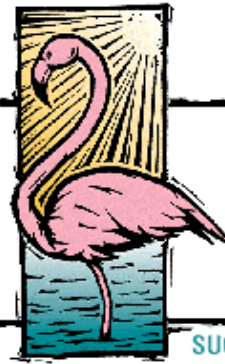

Using SAS/GRAPH® Software to Create Graphs on the Web

Himesh Patel, David Caira



ABSTRACT

This paper highlights some ways of customizing SAS/GRAPH® device drivers to produce graphics that are suitable for presentation on the Web. It also highlights new Version 7 enhancements that allow you to create HTML image maps in order to produce drill-down graphs on the Web.

INTRODUCTION

You can use SAS/GRAPH software in three different ways to produce output on the Web:

1. SAS/GRAPH Device Drivers

This method is designed for batch jobs. It creates HTML files for you and uses the GOPTIONS statement to direct procedure output to GIF files that are referenced in the HTML files.

2. Output Delivery System (ODS)

This method also creates HTML files for you. The advantage of using this method over device drivers is that it lets you combine output from different procedures to define custom layouts.

3. Output Data Set of SAS/GRAPH

This method is for users with extensive HTML knowledge who want to write their own Web pages that include SAS/GRAPH output with drill-down capabilities. With this method, you use the SAS language to generate your own HTML files. Thus, your HTML page design is not limited to designs the first two methods generate for you; you can design the Web pages any way you like.

Using SAS/GRAPH Device Drivers

You may prefer this method if you have experience with device drivers and GOPTIONS statements. SAS/GRAPH software provides three device drivers to create HTML output for the Web: GIF, HTML and WEBFRAME. The GIF driver creates a GIF format file that can be referenced in your HTML file for viewing. The HTML driver creates one HTML file, and also one GIF file for each graph produced by SAS/GRAPH procedures. All the GIF files are referenced in the HTML file. The WEBFRAME driver is similar to the HTML driver, but it creates multiple HTML files that establish a set of frames for displaying the GIF images as thumbnail links. Clicking on these links will cause the full-size version of the thumbnail to be displayed.

GIF DEVICE DRIVER

This is the most basic method to get the SAS/GRAPH output in your HTML files. All other methods utilize GIF device drivers to create the SAS/GRAPH output in GIF format to be referenced in HTML files.

HTML DEVICE DRIVERS

You can use the HTML device driver to create a simple scrollable graph that can be viewed on any viewer. The HTML device driver creates one or more GIF files and one HTML file that references the GIF images created. To create a series of graphs use GOPTIONS DEVICE=HTML. This creates a default HTML file called index.html that can be viewed on any browser. It also creates a GIF file for every graph that is produced by the procedure. *Output 1* and *2* show the first two graphs in a series of five graphs produced by the GCHART procedure with the HTML device driver.

This is the program that creates the graphs shown in *Output 1* and *2*.

Example 1:

```
data totals;
  length dept $ 7 site $ 8;
  input dept site quarter sales;
  datalines;
Parts Sydney 1 4043.97
Parts Atlanta 1 6225.26
Parts Paris 1 3543.97
Repairs Sydney 1 5592.82
Repairs Atlanta 1 9210.21
Repairs Paris 1 8591.98
Tools Sydney 1 1775.74
Tools Atlanta 1 2424.19
Tools Paris 1 5914.25
Parts Sydney 2 3723.44
Parts Atlanta 2 11595.07
Parts Paris 2 9558.29
Repairs Sydney 2 5505.31
Repairs Atlanta 2 4589.59
Repairs Paris 2 7538.56
Tools Sydney 2 2945.17
Tools Atlanta 2 1903.99
Tools Paris 2 7868.34
Parts Sydney 3 8437.96
Parts Atlanta 3 6847.91
Parts Paris 3 6789.85
;
run;

/* define location for output files */
filename out '/tmp';

/* Specify device related options */
goptions reset=all
  nodisplay
  device=html
  gsfname=out
  gsfmode=replace;

/* define axis characteristics */
axis1 value=none label=none width=2;
axis2 label=none minor=none;

/* define legend characteristics */
legend1 cborder=black label=none;

/* Create a vertical bar chart */
proc gchart data=totals;
  title 'Quarterly Sales by Site';
  format quarter roman.;
  format sales dollar8.;
```

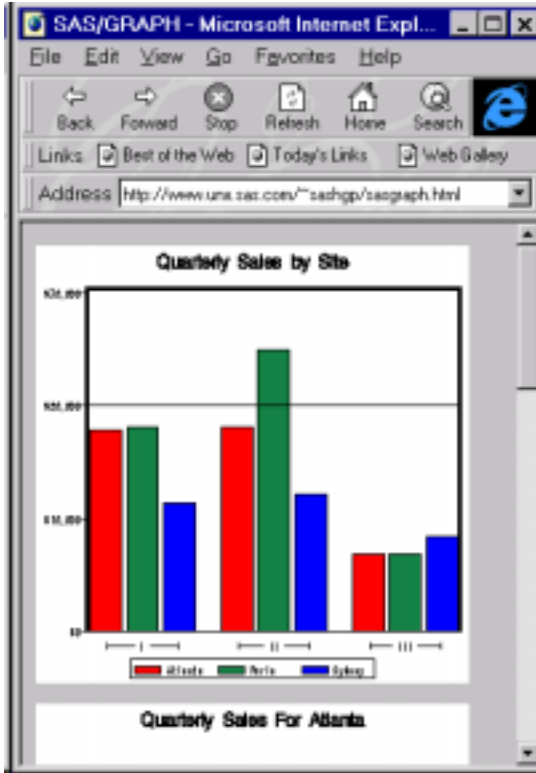
```
vbar site / sumvar=sales
  subgroup=site
  group=quarter
  gspace=4
  space=1
  ref=20000
  maxis=axis1
  raxis=axis2
  gaxis=axis2
  caxis=black
  legend=legend1
  name='Sales';
run;

/* Create a 3d pie chart for Atlanta */
title 'Quarterly Sales For Atlanta';
where site="Atlanta";
pie3d dept / noheading
  sumvar=sales
  name='Atlanta';
run;

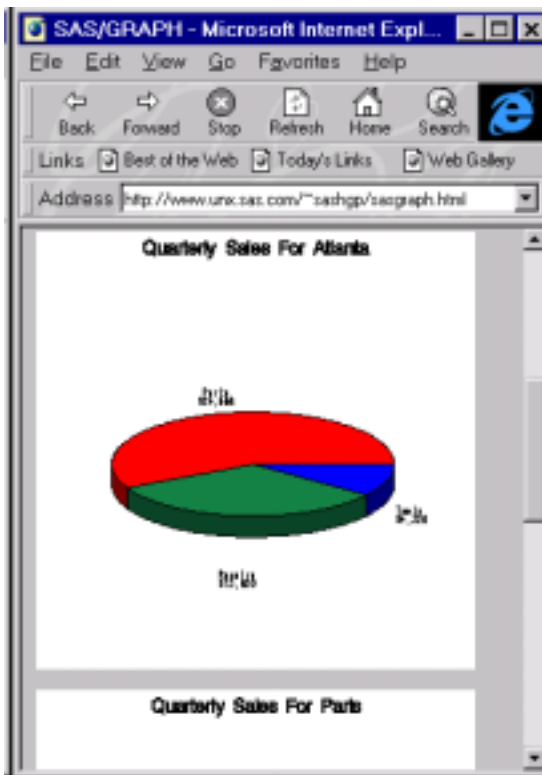
/* Create a 3dpie chart for Paris */
title 'Quarterly Sales For Paris';
where site="Paris";
pie3d dept / noheading
  sumvar=sales
  name='Paris';
run;

/* Create a 3dpie chart for Sydney */
title 'Quarterly Sales For Sydney';
where site="Sydney";
pie3d dept / noheading
  sumvar=sales
  name='Sydney';
run;
quit;

goptions display dev=html;
proc greplay igout=work.gseg nofs;
  replay _all_;
  run;
quit;
```



Output 1



Output 2

Example 1 creates an index.html file and four GIF files, one for each graph. The driver formats the HTML file so that each GIF image displays in the view window of an HTML browser in sequence.

The GSFNAME field in the HTML device entry points to the default location where the files are stored. On most hosts this defaults to your current directory. To redirect the graphics output elsewhere you can change the GSFNAME value by modifying the device entry or by using a GOPTIONS statement.

As the charts are being generated, GOPTIONS NODISPLAY is set. When the PROC GCHART has completed, GOPTIONS DISPLAY is set, and the previously generated charts are replayed into the index.html file. This prevents each chart from overwriting the previous index.html file as it is generated.

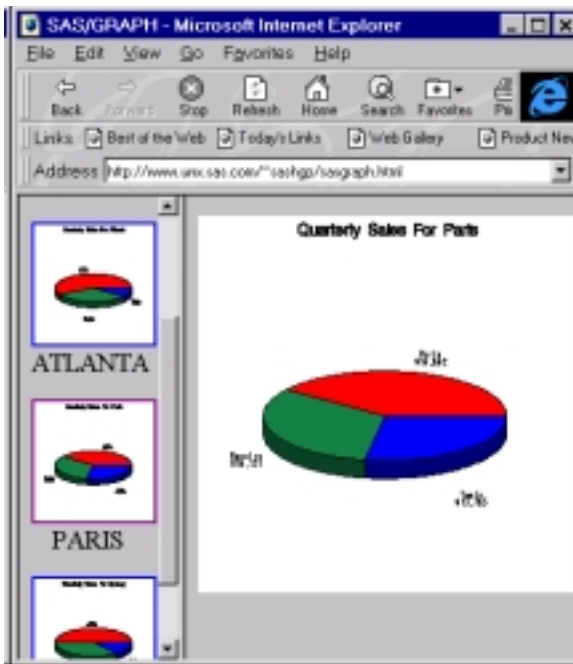
WEBFRAME DRIVER

The WEBFRAME device works like the HTML device with a more advanced layout. WEBFRAME creates one or more GIF files and several HTML files that display the GIF images with thumbnail links. By default the layout is broken down in two views. The left side contains the thumbnail size graphs and the right side contains the current full size graph. You can select a thumbnail graph to view the corresponding full size image on the right side of the page. *Output 3* is created by running the code from *Example 1*, replacing GOPTIONS DEVICE=HTML with GOPTIONS DEVICE=WEBFRAME.



Output 3

In *Output 3* when you click on the thumbnail for Paris the graph for Paris appears on the right frame as shown in *Output 4*. Each full-size graph has a separate HTML file.



Output 4

The files generated by the WEBFRAME device are:

- ***index.html***: This is the layout document and the most important of the group, as this file is the one which is intended to be viewed by a browser or referenced from another HTML page.
- ***sasthumb.html***: This file displays thumbnails of each full size GIF image created by the driver in a frame on the left hand side of the browser window. It also links the thumbnails to the full size GIF images. When the thumbnails are clicked, the full size image is displayed in the target frame. The name of each image appears just below each thumbnail and corresponds to the name that is found in the GRSEG entry.
- ***<graphname>.html***: This HTML file simply displays the full size image indicated, where *<graphname>* is the base name associated with the full size image. There is one of these HTML files created for each graph produced by the procedure.
- ***<GRSEG entry name>.gif***: The WEBFRAME device creates a thumbnail and a full size image pair for each graph produced by the procedure. The full size image filename corresponds to the name in the GRSEG entry. The thumbnail size image filename begins with "f". The remaining characters are the same as those that are used for the full size image filename.

By default, the GSFNAME field of the WEBFRAME device entry points to the location where the files are created. On most hosts this defaults to your current directory. To redirect your graphics output elsewhere you may take the same approach as with the HTML device. This device will create several HTML files and can potentially create many GIF files per procedure. Like the HTML device, the WEBFRAME device is intended to be used in batch jobs to create HTML pages on the fly.

Using Output Delivery System (ODS)

Like the device drivers, the ODS HTML statement allows you to create SAS/GRAPH output for the Web with drill-down capability. In addition, ODS allows you to combine output from different procedures and define custom layouts. This method requires some knowledge of the HTML language. You should also be familiar with the ODS HTML statement and how the graphs are created from the data sets.

Creating SAS/GRAPH Output With ODS

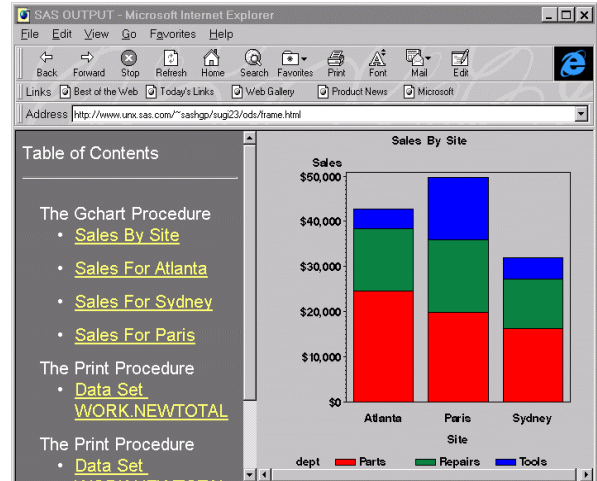
ODS HTML output always creates a “body” file that the viewer displays as a single Web page. ODS HTML can also create a “frame” file, which allows you to divide the Web page into areas. You can use the frame to display the body file, a table of contents, and other frames. For example, the following output displays a table of contents and a graph in a single frame.

In this example you can also drill down on the bars. If you click on the bar for Atlanta, it displays a 3D pie chart for the site Atlanta. Similarly, you can click on Paris or Sydney to get the 3D pie chart for each city. Clicking LEGEND for Parts, Repairs or Tools displays a quarterly report that was produced using PROC PRINT. In addition, clicking on the text links in the Table of Contents will display the appropriate graphs or reports.

To generate this graph with drill-down capability, you must define the links that connect each clickable area with the corresponding pie chart or report. These links are HTML commands that are stored in the HTML body file and point to the file that is the target of the drill-down.

To define these links, you store the HTML commands in a new variable of the input data set. When the ODS HTML statement creates the body file, it includes these commands in the file.

First, look at the current data and decide what action needs to happen when you click on the bars. Then assign this action to a new variable that defines the HTML links from those bars. A new variable is added to the data set from *Example 1*, which defines the drill-down areas. These variables are Bardrills and Legenddrills. A value for Bardrills will be set based on “Atlanta”, “Paris”, and “Sydney” bars, so the new values will be assigned based on the site variable of the new variable Bardrills. Legends can also have a similar drill-down capability. To define drill-down action for the legend “Parts”, “Repairs”, and “Tools”, you will assign new values based on the dept variable for the new variable Legenddrills.



Output 5

The code below is used to generate what is displayed in *Output 5*. The data set, newtotal, is based on the totals data set created in *Example 1*.

Example 2:

```
data newtotal;
set totals;
length Bardrills $ 40 Legenddrills $ 40;
Bardrills='HREF=#'||trim(site);
Legenddrills='HREF=#'||trim(dept);
run;
```

Note that the values of both the new variables is of the form ‘HREF=#<anchor-name>’. This is required syntax for HTML. <Anchor-name> can be any unique name, but it must match exactly the name of the corresponding output. To control the name of the output, use the ANCHOR option in the ODS HTML statement before the procedure.

Once the data set is setup, you need to stop the output from going to the graph and output window. All the graphs are displayed in the GIF format. GIF files can be created using the GIF device driver. Both of these statements are required to get your HTML output.

```
/* Start sending output to html */
ods listing close;
options reset=all
device=gif;
```

ODS has several predefined layouts which can be selected by the STYLE= option. *Output 5* displays the layout of the main frame, containing two sub frames, table of contents and the output frame. The FILE option specifies the HTML body that is the

destination for the graphics output displayed on the right hand side; this is a required option. The PATH= option specifies the destination for the graphics output that is generated by the ODS HTML statement. The CONTENTS= option creates the file that contains the HTML information regarding the table that is displayed on the left column. The FRAME= option creates the file that is the main file that is viewed once the graphs are created. The frame file contains the location for the contents and the body created for the graphs.

```
/* define files to generate html & gif files */
ods html file='chart.html'
      style=styles.d3d
      path='mydir'
      contents='contents.html'
      frame='frame.html';
```

By default the graph has a white background. To give the graph a transparency effect specify:

```
goptions transparency;
```

If you have an image map in the background then you will be able to use that as the background for your graph

Once the setup for the files and devices is done, all that remains are the graphs and reports. The GCHART procedure has two VBAR statement options that control the drill-down feature: HTML= variable and HTML_LEGEND= variable. The HTML= variable option is associated with the graph area such as rectangles and polygons that are used to draw the graph itself. In the data set NEWTOTAL, a new variable, BARDRILLS, is defined to allow the drill-down action for each site.

The HTML_LEGEND= allows you to control the drill-down action for each legend value in the graph. In the data set NEWTOTAL, a new variable LEGENDDRILLS is defined to allow the drill-down action for each department.

```
/* Create vertical bar with drill-down action */
proc gchart;
format sales dollar8.;
title 'Sales by Site';
vbar site / sumvar=sales
      subgroup=dept
      html=Bardrills
      html_legend=Legenddrills
      /* Displayed in Table of contents */
```

```
des='Sales By Site';
```

```
run;
```

The graph that will be displayed when the drill-down is selected on the bar will depend on the unique anchor name given to that graph. Anchor name should be defined before each graph is produced. If no anchor name is defined then ODS will give a unique name starting with IDX<index>, where <index> starts with 0. In the *Output 5*, a 3d pie for selected site is displayed when the drill-down action is performed on the vertical bar.

To define a drill-down action for Atlanta an anchor name should be assigned. The anchor name has to match the one in the data set. Since the next chart uses a WHERE clause to get the information only about Atlanta, you need to assign the ODS HTML ANCHOR option to Atlanta. The anchor name for Sydney and Paris also needs to be assigned appropriately before the PIE3D statement.

```
/* define the anchor point in the html file */
ods html anchor='Atlanta';
where site ? 'Atlanta';
title 'Sales For Atlanta';
pie3d dept / noheading
      sumvar=sales
      des='Sales For Atlanta'
      name='Atlanta';
run;
```

```
ods html anchor='Sydney';
where site ? 'Sydney';
title 'Sales For Sydney';
pie3d dept / noheading
      sumvar=sales
      des='Sales For Sydney'
      name='Sydney';
run;
```

```
ods html anchor='Paris';
where site ? 'Paris';
title 'Sales For Paris';
pie3d dept / noheading
      sumvar=sales
      des='Sales For Paris'
      name='Paris';
run; quit;
```

The Legend drill-down action needs to be defined based on the anchor value that is used in the LEGENDDRILLS variable. Before each PROC PRINT statement an ODS HTML ANCHOR option should be used to define the unique anchor for each department. Anchor name should match the name that

is used in the LEGENDDRILLS variable for each department. PROC SORT sorts the data by site rather than by department.

```
/* sort data for proc print output */
proc sort data=newtotal; by site; run;

/* generate a report for each department */
/* oda anchor= defines the unique html anchor */
title 'Parts Sold';
ods html anchor='Parts';
proc print data=newtotal
    (where = (dept ? 'Parts')) noobs;
format sales dollar8.;
sum sales;
var site quarter sales;
run;

title 'Repairs Done';
ods html anchor='Repairs';
proc print data=newtotal
    (where = (dept ? 'Repairs')) noobs;
format sales dollar8.;
sum sales;
var site quarter sales;
run;

title 'Tools Sold';
ods html anchor='Tools';
proc print data=newtotal
    (where = (dept ? 'Tools')) noobs;
format sales dollar8.;
sum sales;
var site quarter sales;
run;
```

The CLOSE option for the ODS HTML statement closes all files opened by the FILE=, FRAME=, and CONTENTS= options and stops generating HTML output. These files remain open until you close them with the CLOSE option or you specify a different file. To redirect future output to its default window an ODS LISTING statement is required.

```
ods html close;
ods listing;
```

You can use the ODS HTML statement with any SAS/GRAPH procedures. The HTML and HTML_LEGEND options that allow the drill-down ability are only supported under the following procedure statements:

PROC GCHART -

- VBAR, HBAR

- BLOCK
- PIE, PIE3D
- VBAR3D, HBAR3D
- STAR,
- DONUT

PROC GPLOT -

- AREAS
- POINTS

PROC GMAP-

- CHORO
- BLOCK
- PRISM

Using Output Data Set of SAS/GRAPH

If you have extensive knowledge about HTML and want to write your own Web pages that include output from SAS/GRAPH software with drill-down options, use the output data set method.

This method combines procedure syntax with SAS DATA steps and macros to generate the HTML output. To generate the GCHART, GPLOT or GMAP output, you use the procedure's IMAGEMAP= option to specify a data set that will store information that is used to generate the graphs produced by the procedure. In addition to the IMAGEMAP= option, use the procedure's HTML= or HTML_LEGEND= option to identify the variable whose values create the HTML drill-down links. To generate the HTML code, you use SAS DATA steps to write the HTML tags that you need to create and design your frames, and the SAS macro language to dynamically construct the Web pages, based on the output from your SAS program.

Using IMAGEMAP =<data set name> will create the output data set that contains the outline information about the shapes used in the graph. The HTML or HTML_LEGEND procedure option is necessary for the IMAGEMAP option to create a SAS data set. The output data set has the following variables:

graph variable: This is an eight character variable. By default it has the name of the graph. In case of PROC GCHART it will have a value of GCHART. The naming conventions are the same as those for GRSEGS and can be modified by using the NAME option in the procedure.

Length variable: This is a numeric variable that contains the character length of the Link variable.

Link variable: This is a character variable that contains the information that you assigned for the HTML variable in the data set. It has a maximum length of 1024 characters, but its current length is determined from the value in the Length variable. From *Example2*, this variable will contain the same information as Bardrills or Legenddrills.

Shape variable: This is a four character name of the shape that is contained in the graph and can be either RECT or POLY. For a BAR chart it will have a value of RECT and other chart types will have a value of POLY.

Nxy variable: This is a numeric variable that designates the number of points (xy pairs) for the shape. In the case of RECT it will have the lower left as the first pair and upper right as the second pair.

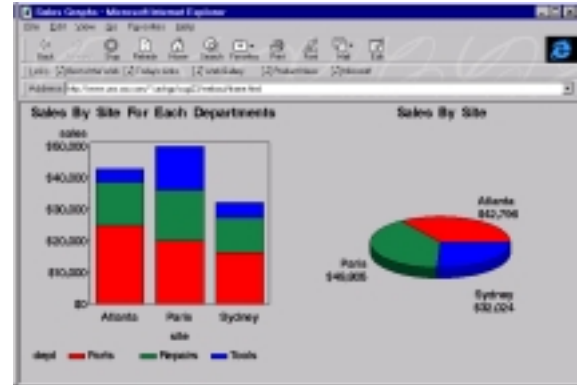
X1 to X100: These are numeric variables for the x coordinates of the shape.

Y1 to Y100: These are numeric variables for the y coordinates of the shape.

With the data set information you can use the SAS MACRO facility to create the HTML file with the appropriate image maps. *Output 6* shows the Web page that was created using the output data set from SAS/GRAPH using the IMAGEMAP= and HTML= options. The display is divided into two separate frames. The left side of the frame displays the BAR chart with drill-down regions and the right displays the output displayed based on the drill-down action.

The IMAGEMAP=<data set name> option is specified as the procedure statement. By default it will create the SAS data set in the work directory.

Example 3 produces the html file displayed in *Output 6*. A VBAR chart similar to *Example 2* can be created with the IMAGEMAP= option, followed by a PIE3D chart to display in the initial content window (on the right side), and PIE3D charts for each site, to display when the corresponding bar is clicked. *Example 3* starts with the newtotal data set created in *Example 2*.



Output 6

Example 3:

```

/* Location to store all the files created */
%let dirpath=<storage directory path>;
filename frame '&dirpath'
filename html '&dirpath'
filename salespie '&dirpath'
filename image '&dirpath';
libname ex '&dirpath';

options reset=all dev=gif
        gsfname=image gsfmode=replace
        xpixels=400 ypixels=400
        ftext=swissb transparency htext=4 pct;

axis1 label=('Site');
axis2 label=('Sales');
legend label=('Dept. ');
Title h=5 pct 'Sales By Site For Each
Departments';

libname ex 'sugi24';

/* Generate picture for contents frame */
proc gchart gout=ex.exp2
    imagemap=htmlmat
    data=newtotal;
format sales dollar8.;
vbar site / subgroup=dept
    sumvar=sales
    width=10
    maxis=axis1
    raxis=axis2
    legend=legend1
    html=bardrill
    des='sales by site'
    name='salesbar';

run;
quit;
/* Generate initial picture for body frame */

```

```

title h=5 pct 'Sales By Site';
proc gchart gout=ex.exp2 data=newtotal;
format sales dollar8.;
pie3d site / noheading
    sumvar=sales
    des='Sales by site'
    name='salespie';
run;

/* Generate picture for each site */
Title h=5 pct 'Sales For Atlanta';
where site ? 'Atlanta';
pie3d dept / noheading
    sumvar=sales
    des='sales for atlanta'
    name='atlanta';
run;

```

```

Title h=5 pct 'Sales For Sydney';
where site ? 'Sydney';
pie3d dept / noheading
    sumvar=sales
    des='sales for sydney'
    name='sydney';
run;

```

```

Title h=5 pct 'Sales For Paris';
where site ? 'Paris';
pie3d dept / noheading
    sumvar=sales
    des='sales for paris'
    name='paris';
run;
quit;

```

To generate the main frame window that contains salesbar.html and salespie.html we will use a simple SAS DATA step with PUT statements that will write HTML information to an external file. The main frame window is divided into two subframe windows. The left frame is referenced as *contents*, containing the output from *salesbar.html*, and the right is referenced as *body*, containing the output from *salespie.html*. Note this type of layout is defined automatically for you if you select the ODS HTML method.

```

/* create the main frame file */
filename frame '&dirpath/frame.html';
data _null_;
    file frame;
    put '<HTML>';
    put '<HEAD>';
    put '<TITLE>Sales Graphs</TITLE>';
    put '</HEAD>';

```

```

    put '<FRAMESET FRAMEBORDER=YES'@;
    put ' FRAMESPACING=0 COLS="52%,*">';
    put '<FRAME MARGINWIDTH="0"'@;
    put ' MARGINHEIGHT="0"'@;
    put ' SRC="salesbar.html" NAME="contents">';
    put '<FRAME MARGINWIDTH="0"'@;
    put ' MARGINHEIGHT="0"'@;
    put ' SRC="salespie.html" NAME="body">';
    put '</FRAMESET>';
    put '</HTML>';
    put '</FONT>';
    put '</BODY>';
    put '</HTML>';
run;

```

The next step is to read in the output data set that was created by the IMAGEMAP option, and generate an HTML file with appropriate image maps. Creation of this HTML file can be viewed in three sections:

- Initializing the necessary HTML tags.
- Generating the appropriate image maps from the IMAGEMAP data set.
- Closing the necessary HTML tags.

To make the example simple, only the necessary HTML tags have been used. After initializing HTML tags, you can customize the statement based on the shape variable, and output to the link, and coordinates for the image maps.

```

/* Generate html file with drill-down information */
data _null_;
    set htmdl dat end=last;
    file html;

```

```

/* Initialize the HTML file with required          */
/* tags and information for the first graph.        */
if _n_=1 then do;
    put '<HTML>';
    put '<HEAD>';
    put '<TITLE>Sales Chart</TITLE>';
    put '</HEAD>';
    put '<BODY>';
    put '<IMG SRC="salesbar.gif" '@;
    put 'USEMAP="#barchart_map">';
    put '<MAP NAME="barchart_map">';
end;

```

```

array x{100} x1-x100;
array y{100} y1-y100;

```

```

/* Add image map points based on shape */
if shape='RECT' then do;

```

```

put '<AREA SHAPE="RECT" '@;
put link $varying. length @;
put 'TARGET="body"' @;
put 'COORDS="" '@;
put 'x1 4. ','y1 4. ','x2 4. ','y2 4. ' ">';
end;
else
if shape='POLY' then do;
put '<AREA SHAPE="POLY" '@;
put link $varying. length @;
put 'TARGET="body"' @;
put 'COORDS="" '@;
do i=1 to nxy-1;
put x{i} 4. ','y{i} 4. ',' @;
end;
put x{nxy} 4. ','y{nxy} 4. @;
put ' ">';
end;

/* End the HTML file */
if last then do;
put '</MAP>';
put '</BODY>';
put '</HTML>';
end;
run;

```

When you drill down on the bar, the *body* frame gets updated with the appropriate PIE3D chart. Each of the 3D pies is displayed from a separate HTML file. In *Output 6* there are three 3D pies, one for each site. It is appropriate to write a simple macro that accepts the site name as a parameter and creates the HTML file based on this name.

```

/* Create the link file in a separate HTML file */
%macro linkfile(htmlfl);
%let flname=&dirpath\&htmlfl..html;
data _null_;
file "&flname";
put '<HTML>';
put '<HEAD>';
put '<TITLE>' "&htmlfl" '</TITLE>';
put '</HEAD>';
put '<BODY>';
put '<MAP NAME=""&htmlfl">';
put '<IMG SRC=""&htmlfl"'.gif>';
put '</BODY>';
put '</HTML>';
run;
%mend;

/* Create the html file to be shown initially */
%linkfile(salespie);

```

```

/* Create the html files to use when drilling down */
%linkfile(atlanta);
%linkfile(sydney);
%linkfile(paris);

```

SAS will provide some simple macros that will generate a basic HTML file from the data set that was created by the IMAGEMAP option. These macros will be located in the ANNOMAC library. The IMAGEMAP option is currently supported for the following procedures:

- GCHART
- GPLOT
- GMAP

Note that the IMAGEMAP option can be used only with the HTML= and/or HTML_LEGEND= options, supported under the SAS/GRAPH procedures.

CONCLUSION

SAS/GRAPH software provides three methods to produce Web output, each with its own advantages. If you're looking for a simple Web output format with a predefined layout, use the device driver method. Device drivers can include the output only from SAS/GRAPH procedures. If you want to combine procedures, define your layout, and have data-driven, drill-down abilities, use the ODS method. If you have extensive HTML knowledge and specific formatting requirements, use the IMAGEMAP method.

CONTACT INFORMATION

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

David Caira (Dave.Caira@sas.com)
SAS Campus Drive
Cary, North Carolina 27513

SAS and SAS/GRAPH are registered trademark or trademarks of SAS Institute Inc in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.