CHAPTER 12

The GANNO Procedure

Overview

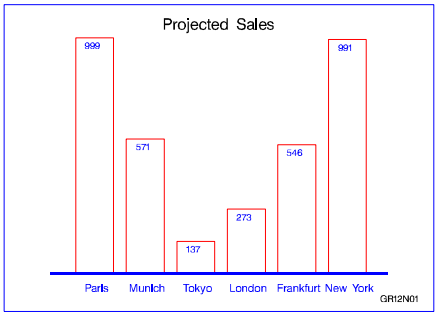
The GANNO procedure displays graphs created by Annotate data sets. It ignores all currently defined titles and footnotes and some graphics s, including BORDER. Use it when you want to display only output from an Annotate data set. See Chapter 10, “The Annotate Data Set,” on page 403 for details.

If you want to include titles, footnotes, and graphics options in the output, use the GSLIDE procedure instead of GANNO to display the graph created by the Annotate data set. See Chapter 27, “The GSLIDE Procedure,” on page 959 for an example of graphics output created with Annotate data sets and displayed with the GSLIDE procedure.

By default, both the GANNO and GSLIDE procedures scale graphics output from the data set to fill the entire graphics area. However, if you are using a data coordinate system and the data values are so large that some of the graphics elements do not fit in the graphics output area and are not displayed, you can use the GANNO procedure with the DATASYS option. This will cause the procedure to scale the output to fit the available space. The GSLIDE procedure does not have this capability.

Figure 12.1 on page 504 displays output from an Annotate data set.

Figure 12.1 Displaying Annotate Graphics with the GANNO Procedure (GR12N01)



The program for this graph is in Example 1 on page 507.

Procedure Syntax

Requirements: An input Annotate data set is required.

Supports: Output Delivery System (ODS)

PROC GANNO ANNOTATE=*Annotate-data-set*
<DATASYS>
<DESCRIPTION='*entry-description*'>
<GOUT=<*libref.*>*output-catalog*>
<IMAGEMAP=*output-data-set*>
<NAME='*entry-name*' | *variable-name*>;

PROC GANNO Statement

Identifies the Annotate data set and draws the graphics output defined by that data set. Optionally, it scales the output to accommodate data-dependent coordinate values and specifies an output catalog.

Syntax

PROC GANNO ANNOTATE=*Annotate-data-set*
<DATASYS>
<DESCRIPTION='*entry-description*'>
<GOUT=<*libref.*>*output-catalog*>
<IMAGEMAP=*output-data-set*>
<NAME='*entry-name*' | *variable-name*>;

Required Arguments

ANNOTATE=*Annotate-data-set*
ANNO=*Annotate-data-set*
specifies a data set that includes Annotate variables that identify graphics commands and parameters.

See also: Chapter 10, "The Annotate Data Set," on page 403

Options

Options in the GANNO statement affect all graphs produced by that statement. You can specify as many options as you want and list them in any order.

DATASYS
indicates that absolute or relative data-dependent coordinates occur in the Annotate data set and scales the coordinates to fit the graphics output area. Use DATASYS only with Annotate data sets in which the coordinate system variables XSYS, YSYS, and HSYS specify the values 1, 2, 7, or 8.

Use the DATASYS option when graphics elements that were created with data-dependent variables do not fit in the graphics output area. This happens when the coordinate values generated by the data exceed a range of 0 to 100.

If you omit DATASYS, GANNO attempts to draw each graphics element according to the data values assigned to it without scaling the values. If the range of data values is too large, some graphics elements do not display.

See also: "Using the DATASYS Option to Scale Graphs" on page 506

Featured in: Example 1 on page 507

DESCRIPTION='*entry-description*'
DES='*entry-description*'
specifies the description of the catalog entry for the chart. The maximum length for *entry-description* is 40 characters. The description does not appear on the chart. By default, the GANNO procedure assigns the description OUTPUT FROM PROC GANNO.

Featured in: Example 2 on page 509

GOUT=<*libref.*>*output-catalog*
specifies the SAS catalog in which to save the graphics output produced by the GANNO procedure. If you omit the libref, SAS/GRAPH looks for the catalog in the temporary library called WORK and creates the catalog if it does not exist.

See also: "Storing Graphics Output in SAS Catalogs" on page 49

Featured in: Example 2 on page 509

IMAGEMAP=*output-data-set*
creates a SAS data set that contains information about the graph and about areas in the graph. This information includes the shape and coordinates of the areas, and is used to build an HTML file that links the graph areas to other files or images. This linking provides drill-down functionality on the graph. The Imagemap data set also contains the information that is stored in the HTML variable in the Annotate data set. Therefore, in order to use IMAGEMAP= to create an HTML file, you must also use the HTML variable in the Annotate data set.

See also: "Customizing Web Pages for Drill-down Graphs" on page 100 and "HTML Variable" on page 462

Featured in: Example 4 on page 514

NAME='*entry-name*' | *variable-name*
specifies one of the following:

- the name of the catalog entry for the graph
- a variable name for each value for which a separate graph is produced.

If the value you assign to the NAME= option is enclosed in quotation marks, the procedure interprets it as a catalog entry name; if the value is not enclosed in quotes, the procedure interprets it as a variable name.

The value *entry-name* specifies the name of the catalog entry for the graph. The maximum length for *entry-name* is 8 characters. The default name is GANNO. If the specified name duplicates the name of an existing entry, SAS/GRAPH software adds a number to the duplicate name to create a unique entry, for example, GANNO1.

If you specify *variable-name*, the GANNO procedure produces a separate graph for each different value of that variable. In addition, when you specify NAME=*variable-name*, each value of the variable is used as the name of the catalog entry for that graph. A value that is longer than 8 characters is truncated. For example, if the value is **`Frankfurt`**, it is truncated to **`Frankfur`**. A second catalog entry would be **`Frankfu1`**. Consequently, you cannot use NAME='*entry-name*' at the same time.

Note: Specifying NAME=*variable-name* in the PROC GANNO statement produces results similar to those produced by the BY statement in a procedure that supports BY-group processing. See "BY Statement" on page 177 for details. △

Featured in: Example 2 on page 509Example 3 on page 511

Using the DATASYS Option to Scale Graphs

If your Annotate data set specifies a coordinate system that is based on data values (that is, XSYS, YSYS, and HSYS are assigned the values 1, 2, 7, or 8), the data values determine the size and location of the graphics elements on the output.

If the procedure that specifies the annotation generates axes (such as GPLOT or GCHART), by default the axes are scaled to accommodate the full range of data values and to fit in the procedure output area. Because all values are included in the axes, the graph displays all the Annotate output that is dependent on data values.

However, if the annotation displays with the GSLIDE or GANNO procedure, which do not generate axes, the data values may generate coordinate values that exceed the limits of the graphics output area.

In this case, you can use the DATASYS option to tell the procedure that the Annotate data set contains data-dependent coordinates and to scale the output accordingly. For an illustration of this process, see Example 1 on page 507.

When you use the DATASYS option, the GANNO procedure reads the entire input data set before drawing the graph and creates an output environment that is data dependent; that is, the environment is based on the minimum and maximum values that are contained in the data set. It then scales the data to fit this environment so that all graphics elements can be drawn.

Although the DATASYS option enables you to generate graphs using one of the data-dependent coordinate systems, it requires that the procedure scan the entire data set to determine the minimum and maximum data values. You can save this extra pass of the data set by using data-dependent values only in procedures that generate axes. Annotate coordinate system 5 (percent of the procedure output area) is recommended for use with the GANNO procedure. This coordinate system works equally well with the GSLIDE procedure if you decide to display the annotation with titles and footnotes.

Examples

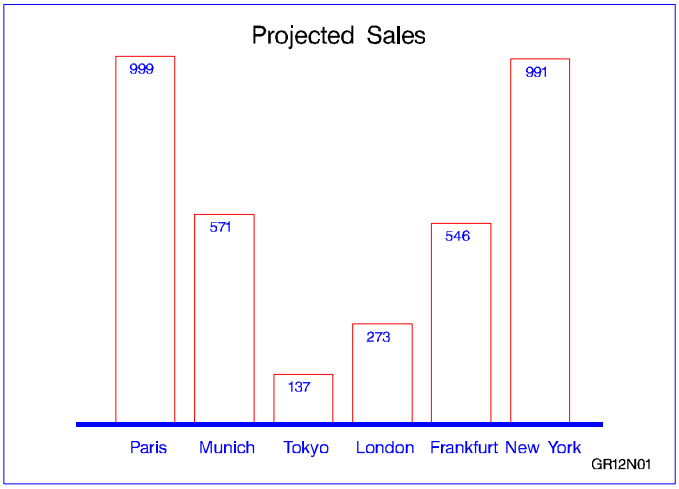
Example 1: Scaling Data-Dependent Output

Procedure features:

PROC GANNO statement options:

ANNOTATE=
DATASYS

Sample library member: GR12N01



This example uses an Annotate data set to scale data-dependent output with the DATASYS option and create a vertical bar chart of sales for each of six sites. The values that determine the height of each bar range from 137 to 999. The range of values is so large that the GANNO procedure cannot fit all of the bars in the output area without scaling the output. This program uses the DATASYS option to scale the data values so that the bars fit in the graphics output area.

Set the graphics environment.

```
goptions reset=global gunit=pct cback=white
         colors=(black blue green red);
```

Create the data set WRLDTOTL. WRLDTOTL contains sales data for six sites. SITENAME contains the names of the sites. MEAN contains the average sales for each site.

```
data wrldtotl;
   length sitename $ 10;
```

```
   input sitename $ 1-10 mean 12-15;
   datalines;
Paris      999
Munich     571
Tokyo      137
London     273
Frankfurt  546
New York   991
;
run;
```

Create the Annotate data set, WRLDANNO. XSYS and YSYS specify coordinate system 2 (absolute data values) for X and Y. HSYS specifies coordinate system 3 (percent of the graphics output area) for SIZE. The SET statement processes every observation in WRLDTOTL.

```
data wrldanno;
   length function color $ 8 text $ 20;
   retain line 0 xsys ysys '2' hsys '3' x 8;
   set wrldtotl end=end;
```

Draw the bars. The MOVE function defines the lower left corner of the bar. The BAR function draws the bar. Bar height (Y) is controlled by MEAN.

```
   function='move'; x=x+8; y=20; output;
   function='bar'; y=y+(mean); x=x+9;
      style='empty'; color='red'; output;
```

Label the bar with the name of site.

```
   function='label'; y=0; x=x-4; size=3.5;
      position='E'; style='swiss';
      color='blue'; text=sitename; output;
```

Move to the top of the bar and write the value of MEAN.

```
   function='move'; y=y+(mean)-3; output;
   function='label'; x=x-1; text=left(put(mean,3.));
      position='5'; style='swiss'; size=3; output;
```

After all the observations are processed, add an axis line, title, footnote, and frame. The MOVE and DRAW functions draw the axis line. The LABEL function writes the title and the footnote. The FRAME function draws a border around the output.

```
   if end then do;
      function='move'; x=10; y=20; output;
      function='draw'; x=90; y=20; line=1;
         size=.5; color='blue'; output;
      function='label'; x=50; y=95; text='Projected Sales';
```

```
         xsys='3'; ysys='3'; position='5'; style='swissb';
         size=5; color=' '; output;
         x=92; y=5; size=3; style='swiss'; text='GR12N01 '; output;
      function='frame'; color='blue'; when='b';
         style='empty'; output;
   end;
run;
```

Display the annotate graphics. The ANNOTATE= identifies the data set that contains the graphics commands. DATASYS tells the procedure to use the maximum and minimum data values to construct the output environment. In addition, the values of X and Y are scaled to fit the environment and all of the bars display on the graph.

```
proc ganno annotate=wrldanno
   datasys;
run;
quit;
```

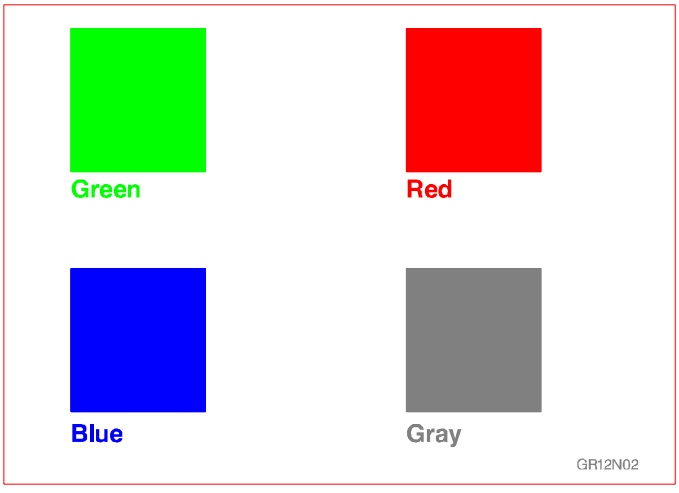
Example 2: Storing Annotate Graphics

Procedure features:

PROC GANNO statement options:

DESCRIPTION=
GOUT=
NAME=

Sample library member: GR12N02



This example creates an Annotate data set that draws four colored squares, displays the data set as a single graphics output, and stores the output as a catalog entry in a

permanent catalog. In this example, the NAME= option specifies a text string that identifies the name that is stored with the graphics output in the catalog.

Assign the libref and set the graphics environment.

```
libname reflib 'SAS-data-library';
goptions reset=global gunit=pct cback=white
   colors=(black blue green red);
```

Create the Annotate data set, SQUARES. XSYS and YSYS specify coordinate system 3 (percent of the graphics output area) for X and Y.

```
data reflib.squares;
   length function style color $ 8 text $ 15;
   xsys='3'; ysys='3';
```

Draw the first square. The COLOR variable assigns the color for the square. The FUNCTION variable selects the operation to be performed by the Annotate facility. The X and Y variables contain coordinate values. The BAR function draws the square. When the STYLE variable is used with the BAR function, it selects the fill pattern for the bar.

```
   color='green';
   function='move'; x=10; y=65; output;
   function='bar';  x=30; y=95; style='solid'; output;
```

Label the first square. The LABEL function creates the label. The POSITION value of 6 left-justifies the text with respect to X and Y. The TEXT variable specifies the text string to be written.

```
   function='label'; x=10; y=63; position='6';
      style='swissb'; size=2; text='Green'; output;
```

Draw and label the second square.

```
   color='red';
   function='move'; x=60; y=65; output;
   function='bar';  x=80; y=95; output;
   function='label'; x=60; y=63; position='6';
      style='swissb'; size=2; text='Red'; output;
```

Draw and label the third square.

```
   color='blue';
   function='move'; x=10; y=15; output;
   function='bar';  x=30; y=45; output;
   function='label'; x=10; y=12; position='6';
      style='swissb'; size=2; text='Blue'; output;
```

Draw and label the fourth square.

```
    color='gray';
    function='move'; x=60; y=15; output;
    function='bar';  x=80; y=45; output;
    function='label'; x=60; y=12; position='6';
       style='swissb'; size=2; text='Gray'; output;
```

Add a footnote.

```
    x=88; y=5; position='5'; size=1.5; style='swiss';
       text='GR12N02'; output;
```

Draw a red frame.

```
    function='frame'; color='red'; when='b';
       style='empty'; output;
run;
```

Display the annotate graphics. GOUT= assigns the catalog in which the graphics output is stored. NAME= assigns a name to the entry stored in the REFLIB.EXCAT catalog. DESCRIPTION= assigns a description to the catalog entry.

```
proc ganno annotate=reflib.squares
   gout=reflib.excat
   name='gr12n02'
   description='Four squares';
run;
quit;
```

Example 3: Using the NAME= Option to Produce Multiple Graphs

Procedure features:

PROC GANNO statement option:

NAME=

Data set: REFLIB.SQUARES on page 510

Sample library member: GR12N03

In this example, the GANNO procedure uses the NAME= option to generate multiple graphs from one Annotate data set. Since NAME= is assigned the variable COLOR, the GANNO procedure generates separate graphics output for each value of the COLOR, as shown in Figure 12.2 on page 512, Figure 12.3 on page 513, Figure 12.2 on page 512 and Figure 12.4 on page 513.

Each output is stored as a separate entry in the output catalog REFLIB.EXCAT. The entries are named according to the values of COLOR: **BLUE, GRAY, GREEN**, and **RED**.

Note that the output for **GRAY** includes the footnote shown in Example 2 on page 509. The output for **RED** shows the frame that is generated by the Annotate data set. The black borders in the other outputs are not generated by the code.

Assign the libref and set the graphics environment.

```
libname reflib 'SAS-data-library';
goptions reset=global gunit=pct cback=white
   colors=(black blue green red);
```

Generate the annotate graphics, separating graphs by color. NAME= identifies the variable whose values PROC GANNO uses to generate the output. GANNO produces separate output for each value of COLOR. The COLOR value is the name of the catalog entry.

```
proc ganno annotate=reflib.squares
           name=color
           gout=reflib.excat
           description='Individual squares';
run;
```

Figure 12.2 Output for COLOR Value BLUE (REFLIB.EXCAT.BLUE.GRSEG)

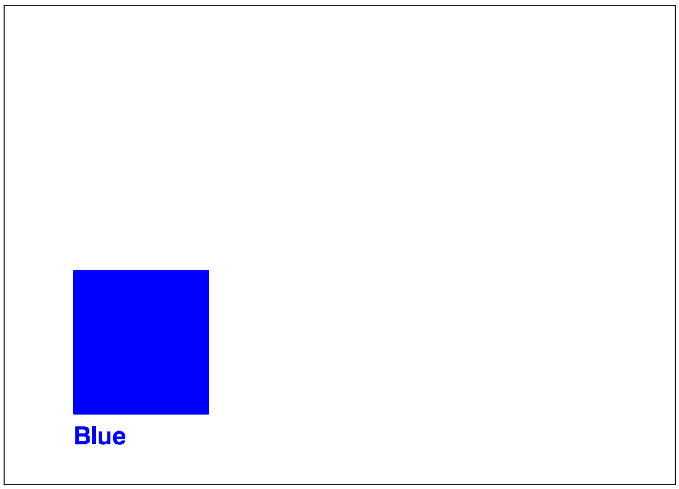


Figure 12.3 Output for COLOR Value GRAY (REFLIB.EXCAT.GRAY.GRSEG)

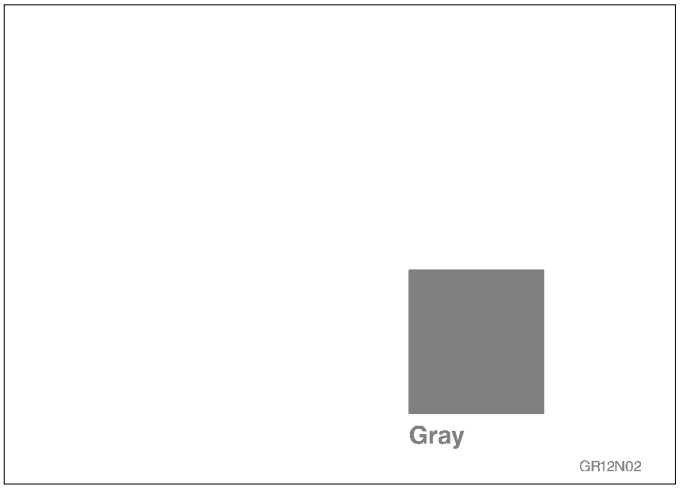


Figure 12.4 Output for COLOR Value GREEN (REFLIB.EXCAT.GREEN.GRSEG)

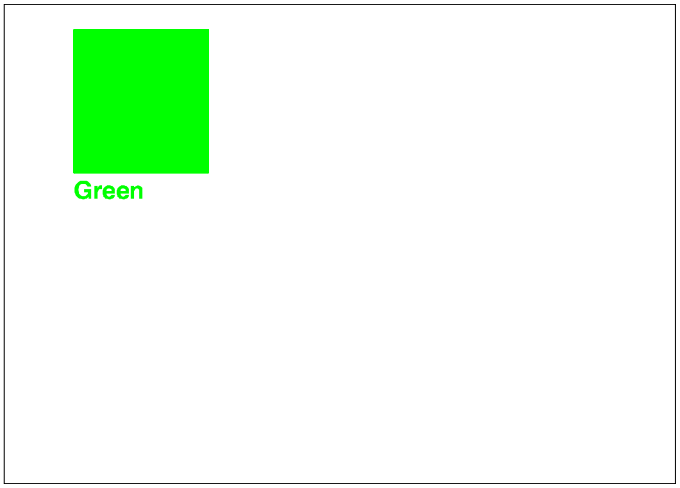
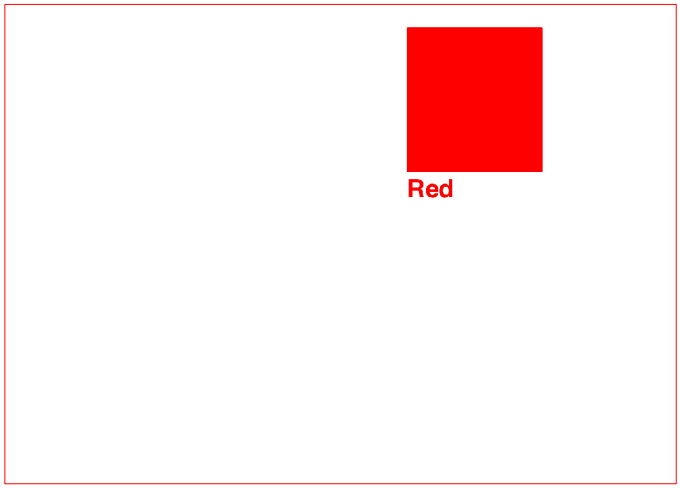


Figure 12.5 Output for COLOR Value RED (REFLIB.EXCAT.RED.GRSEG)



Example 4: Using Annotate Graphics in a Drill-down Graph

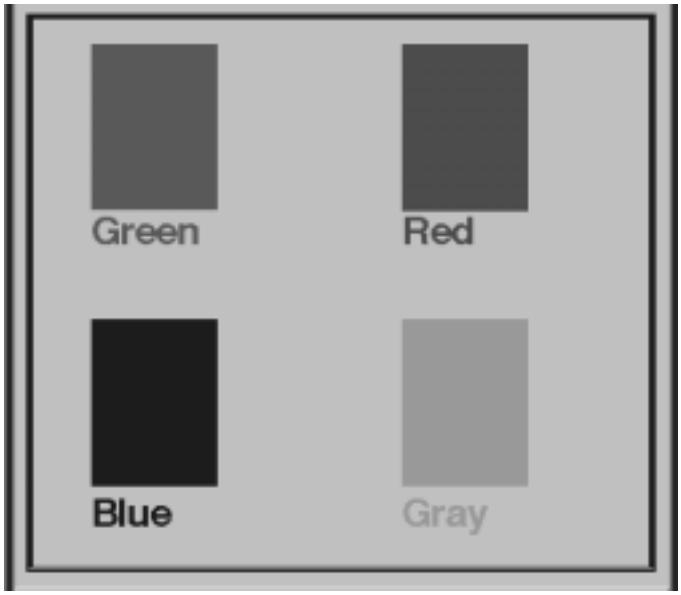
Procedure features:

PROC GANNO statement option:

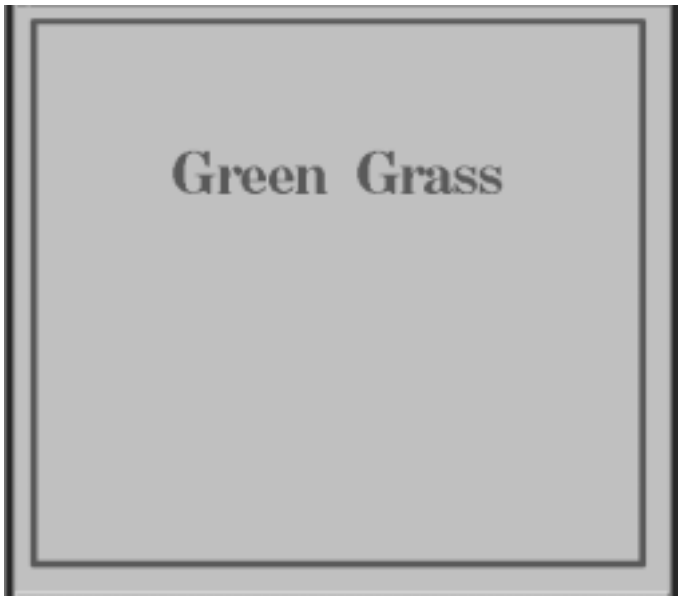
IMAGEMAP=

Sample library member: GR12N04

This example creates essentially the same Annotate data set used in Example 2 on page 509. It draws four colored squares and displays the data set as a single graphics output.



However, this time the example shows you how to use Annotate graphics to generate a drill-down graph (see "About Drill-down Graphs" on page 90). The example uses the HTML variable in the Annotate data set to specify linking information that defines each of the four squares as a hot zone. When the graph is viewed in a browser, you can click on a square to drill down to a related graph. For example, if you click on the green square, it drills down to a graph that confirms that you selected the green square.



The example uses the HTML device driver to generate the drill-down graph. To implement the drill-down capability, the Annotate data set uses the HTML variable to provide the linking information (see "HTML Variable" on page 462), and the GANNO procedure uses the IMAGEMAP= option to create an Imagemap data set. The presence of the HTML variable in the Annotate data set and the IMAGEMAP= option on the GANNO procedure causes the HTML device driver to generate an image map for the graph. It writes the image map to the file index.html, which the HTML device driver creates for displaying Web output (see "Displaying Graphs on One Web Page (DEV=HTML)" on page 77).

To prevent the HTML device driver from writing over the contents of index.html after the drill-down graph has been generated, the example switches to the GIF device driver. It then runs four GSLIDE procedures to generate the target output. Each GSLIDE procedure uses the NAME= option to name the graph it produces, ensuring that the GIF driver creates files named green.gif, blue.gif, red.gif, and gray.gif. These are the files that are referenced as targets by the strings that are specified for the Annotate data set's HTML variable.

Allocate a storage location for all the output files, and set the graphics environment. The HTML device driver generates output that includes both HTML and GIF files, so the libref must point to an aggregate storage location. It cannot point to a file.

```
/* define the output location */
filename webout 'path-to-Web-server';
   /* set the graphics environment */
goptions reset=global gunit=pct
         colors=(black blue green red);
```

Create the Annotate data set. The HTML variable is used to define the linking information for each square. Because the GSLIDE procedures that generate the target output use NAME= to ensure the output files are named green.gif, red.gif, blue.gif, and gray.gif, strings that reference those names are assigned to the HTML variable for the appropriate observation in the data. For the final observation, the HTML variable's value is set to a null string; otherwise it would retain the last assigned value, which is **href=gray.gif**. In that case, the graph's background area would be defined as a hot zone that links to file gray.gif. For a description of the other functions and variables used in the Annotate data set, see Example 2 on page 509.

```
    /* create Annotate data set */
data squares;
     length function style color $ 8
            html text $ 15;
     xsys='3'; ysys='3';

        /* draw the green square */
     color='green';
     function='move'; x=10; y=65; output;
     function='bar';  x=30; y=95; style='solid';
        html='href=green.gif'; output;

        /* label green square */
     function='label'; x=10; y=63; position='6';
        style='swissb'; size=2; text='Green'; output;

        /* draw the red square */
     color='red';
     function='move'; x=60; y=65; output;
     function='bar';  x=80; y=95;
        html='href=red.gif'; output;

        /* label red square */
     function='label'; x=60; y=63; position='6';
        style='swissb'; size=2; text='Red'; output;

        /* draw the blue square */
     color='blue';
     function='move'; x=10; y=15; output;
     function='bar';  x=30; y=45;
        html='href=blue.gif'; output;

        /* label blue square */
     function='label'; x=10; y=12; position='6';
        style='swissb'; size=2; text='Blue'; output;

        /* draw the gray square */
     color='gray';
     function='move'; x=60; y=15; output;
     function='bar';  x=80; y=45;
        html='href=gray.gif'; output;

        /* label gray square and add a footnote */
     function='label'; x=60; y=12; position='6';
```

```
        style='swissb'; size=2; text='Gray'; output;

        /* draw a blue frame */
     function='frame'; color='blue'; style='empty';
        /* set null link for background area in frame */
        html=''; output;
run;
```

Set the graphics environment for the Web page. DEV= specifies the HTML device driver, which will create the HTML and GIF files needed for the Web page. GSFNAME= specifies the libref that points to the storage location that was allocated for the Web output. XPIXELS= and YPIXELS= define a size in pixels for the graphics area. TRANSPARENCY specifies that the background areas in all generated graphs should appear to be transparent when the images are displayed in a browser.

```
/* set the graphics options for the web page */
goptions dev=html gsfname=webout
         xpixels=450 ypixels=400
         transparency;
```

Generate the drill-down graph. IMAGEMAP= specifies ANNOMAP as the name for the Imagemap data set.

```
/* generate annotate graphics */
proc ganno annotate=squares
    imagemap=annomap
    description='Four squares';
run;
```

Change to the GIF driver and generate the target output. DEV= changes the device driver to GIF so that the target output files will be generated as GIF files. FTEXT= and CTEXT= specify a font and color for the text in graphics output. PROC GSLIDE is then run four times to generate the four graphs that will serve as target output for the links that are defined in the drill-down graph.

```
/* change to gif driver for target output */
goptions dev=gif ftext=centb ctext=green;

/* generate the target output */
proc gslide wframe=4
   cframe=green name='green';
   note height=20;
   note height=10
        justify=center
       'Green Grass';
run;

goptions ctext=blue;
proc gslide wframe=4
   cframe=blue name='blue';
   note height=20;
```

```
   note height=10
        justify=center
       'Blue Sky';
run;

goptions ctext=red;
proc gslide wframe=4
   cframe=red name='red';
   note height=20;
   note height=10
        justify=center
       'Red Wine';
run;

goptions ctext=gray;
proc gslide wframe=4
   cframe=gray name='gray';
   note height=20;
   note height=10
        justify=center
       'Gray Mare';
run;
quit;
```

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/GRAPH® Software: Reference, Version 8*, Cary, NC: SAS Institute Inc., 1999.

SAS/GRAPH® Software: Reference, Version 8

ISBN 1-58025-525-6

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, October 1999