CHAPTER

5

Bringing SAS/GRAPH Output to the Web

About SAS/GRAPH Software and the Web

You can bring SAS/GRAPH output to the Web in several ways:

□ You can use SAS/GRAPH Web drivers to produce simple Web pages (see “Using SAS/GRAPH Web Drivers to Create Web Output” on page 77). The Web drivers are the GIF, HTML, and WEBFRAME drivers. The GIF driver creates GIF format files for your graphs, and you have to create the HTML files to reference those graphs. The HTML and WEBFRAME drivers generate both the HTML and GIF files for you and fully automate the process. All you have to do is specify either the HTML or WEBFRAME device driver, and SAS/GRAPH does the rest. If you specify the WEBFRAME driver, SAS/GRAPH generates thumbnail-size graphs that you can click to see the full-size graph; no other approach to generating Web output produces these thumbnail graphs. With the HTML and WEBFRAME drivers, you can also create drill-down graphs that contain hot zones that link to other output, such as another graph or a report (see “About Drill-down Graphs” on page 90).

□ You can use the Output Delivery System (ODS) to gain more control over the Web pages that you produce (see “Using the Output Delivery System (ODS) with SAS/GRAPH Software” on page 81). This approach can be used with the GIF, JAVA, or ACTIVEX device drivers.

When used with the GIF driver, ODS automatically generates the HTML and GIF files needed to display the output. When used with the JAVA or ACTIVEX drivers, ODS automatically generates the HTML file needed to display the graph, which is defined within the HTML file. (As discussed in “Creating Java Applets and ActiveX Controls” on page 104, the JAVA and ACTIVEX drivers can only be used with the GCHART, GCONTOUR, GMAP, GPLOT, and G3D procedures.)

With ODS, you name the HTML files and have more influence over the content and presentation of that output than you have with the SAS/GRAPH Web drivers. For example, you can combine graphics and non-graphics output, and generate a Table of Contents that automatically links to each piece of output. You can also use SAS/GRAPH options with ODS to create drill-down graphs that contain hot zones that link to other output, such as another graph or a report (see “About Drill-down Graphs” on page 90).

□ You can customize Web-page design for drill-down graphs by creating your own HTML files to reference graphics output (see “Customizing Web Pages for Drill-down Graphs” on page 100). This approach is for users who have extensive

HTML knowledge who want drill-down graphs but who also want to design the Web pages. With this approach, you use SAS/GRAPH procedure options and a SAS/GRAPH macro to create the image map needed to implement drill-down graphs. Then you create the HTML files and write the HTML tags needed to display the output, associate the image map with that output, and resolve the image-map links.

□ You can combine multiple graphs into a single GIF image that displays the graphs at timed intervals, thus creating an animation sequence (see “Animating GIF Files (DEV=GIFANIM)” on page 105). The output is a single GIF file, which you can then incorporate into a Web page. This approach is for users who have experience in animation. You can generate animated sequences of graphs to display in reports that are published on the Web.

About the Output Files Generated for the Web

If you use SAS/GRAPH to generate Web output, then the output must include at least one HTML file. If you use the GIF device driver for the graphs, then the output also includes at least one GIF file. An HTML file contains the HTML language elements required to view the output in a Web browser, and a GIF file contains graphics output.

An HTML file that is generated by SAS/GRAPH always has the elements needed to reference each graph that is produced by the SAS program. Table 5.1 on page 73 shows the HTML tags that SAS/GRAPH creates as needed to reference graphs.

Table 5.1 HTML Tags Used to Reference Graphs

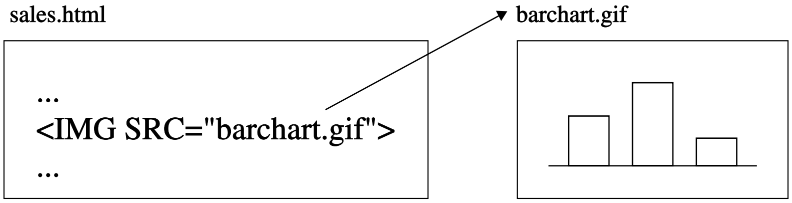
Tag	References ...
<IMG>	the location and name of a GIF file
<APPLET>	the location and name of a Java applet
<OBJECT>	the location and name of an ActiveX control

For example, this tag references an image named barchart.gif, which resides in the same location as the HTML file (because no other location is indicated):

```
<IMG SRC="barchart.gif">
```

Figure 5.1 on page 73 shows the relationship between an HTML file’s <IMG> tag and the GIF file.

Figure 5.1 How an HTML File References a GIF File



To view Web output, use a Web browser to view the HTML file, which automatically displays the graph that is referenced by the <IMG>, <APPLET>, or <OBJECT> tag.

If the output files are located in your file system, then users with access to that file system can directly open the HTML file. If the output files are located on a Web server, then users with Internet access to the World Wide Web can access the HTML file through its *uniform resource locator* (URL).

Specifying Output Locations for HTML and GIF Files

When you use the HTML or WEBFRAME device drivers, or when you use SAS/GRAPH with ODS and specify the GIF device driver, SAS/GRAPH creates all of the HTML and GIF files you need to view the output. In these cases, you must specify a location in your file system where you want the output files to be stored. This location must be an aggregate file storage location (for example, a directory or PDS); it cannot be a file.

The following alternatives are available for specifying the output location for the HTML and GIF files:

□ Let SAS/GRAPH create all files in the default location. This is the easiest alternative and might be the best choice if you plan to move the files after they are created. For example, you might move them from your local file system to a Web server. The location that SAS/GRAPH uses as a default depends on the approach you use to generate the Web output:

Output from HTML or WEBFRAME Driver	Output from ODS
The driver creates an aggregate file storage location named *sasgraph*. The *sasgraph* location is created under the storage location where the SAS session was started.	In most operating environments, the default location is the storage location where the SAS session was started.

When you use the default output location, be sure to start the SAS session from the location where you want to store the HTML and GIF files.

□ Use a FILENAME statement to define a fileref that points to the storage location where you want to create the files. The advantage of using a FILENAME statement is that you can specify it at the top of your program, and then assign the fileref in multiple locations in the program. To redirect all the output that goes to that fileref, you only have to redefine the location on the FILENAME statement.

A fileref that defines the location of Web output must point to an aggregate file storage location. The following FILENAME statement defines the fileref outfiles:

```
filename outfiles 'path-to-Web-server-space';
```

After defining the fileref, you can use it as needed to direct the output files. The place to specify it depends on the approach you use to generate the Web output:

Output from HTML or WEBFRAME Driver	Output from ODS
You can specify the fileref on the GSFNAME= graphics option: **goptions gsfname=outfiles;**	Specify the fileref on the ODS HTML statement's PATH= or GPATH= options. To direct all files to the same location, use only PATH=. To direct the HTML files to one location and the GIF files to another, use both PATH= and GPATH=. The following specification directs all ODS output to the same location: **ods html body='myfile.html' path=outfiles;**

□ Specify the full location (not a fileref) directly in the ODS HTML statement's PATH= or GPATH= options (available only when using ODS). This approach might be best for directing to different locations the output from multiple procedures. The following shows the syntax:

```
ods html body='myfile.html'
         path='path-to-Web-server-space';
```

Naming Conventions Used for HTML Files

The names that are assigned to HTML files depend on the approach you use to generate the Web output:

Output from HTML or WEBFRAME Driver	Output from ODS
You cannot specify a name for the HTML files. Both drivers always create a file named index.html, which is the file that should be viewed in the browser. With the HTML driver, index.html is the only HTML file that is created. With the WEBFRAME driver, additional HTML files are created, which the driver also names. The HTML files always replace the HTML files from previous procedure runs, unless the output is directed to a different location.	You assign filenames for the HTML files you want created. You assign the names on the BODY=, CONTENTS=, PAGE=, and FRAME= options. While the HTML destination is open, each specified HTML file remains open and additional output is appended to the file, until you open an alternative file for the output type or close the HTML destination. If you run the program again at another time without changing the filenames specified, the output from the current program run replaces the output from the previous program run.

Naming Conventions Used for GIF Files

When you use SAS/GRAPH to generate Web output, the graphs produced in the program are always saved in two formats: GRSEG catalog entries and GIF files. In all cases, the GIF filenames are always derived from the catalog entry names of the corresponding graphs.

For example, you can use a procedure's NAME= option to assign a name of up to eight characters to the catalog entry. If you assign the name MYGRAPH to the catalog entry, SAS/GRAPH names the GIF file mygraph.gif. If you do not use NAME=, SAS/GRAPH names the entry with the first eight characters of the procedure name (for example, GCHART), in which case SAS/GRAPH names the GIF file gchart.gif. (For more information on catalog entry names, see “Names and Descriptions of Catalog Entries” on page 51.)

SAS/GRAPH does not replace existing GRSEG entries when a procedure creates a new entry of the same name. Rather, it adds a number to the duplicate name to make it unique. For example, if you use a procedure's NAME= option to name an entry MYGRAPH and an entry named MYGRAPH already exists in the output catalog, SAS/GRAPH names the new entry MYGRAPH1, and then names the GIF file mygraph1.gif. Catalog entry names are limited to eight characters, so if the duplicate name has eight characters, SAS/GRAPH replaces the final character with the added number.

To replace an existing catalog entry, your program must first use the GREPLAY procedure to delete the existing catalog. For example, assume that the output catalog is the default, WORK.GSEG, and assume that you use BY-group processing on the GCHART procedure to run a program that generates three catalog entries that are named GCHART, GCHART1, and GCHART by default. To run the same program and ensure that the catalog entries receive the same names, you must first run the following GREPLAY procedure to delete the three existing catalog entries before generating the new entries that have the same names:

```
proc greplay igout=work.gseg nofs;
   delete gchart gchart1 gchart2;
```

About the Size of Graphs and Text in GIF Images

When a SAS/GRAPH program generates a GIF image, many factors in your operating environment can influence the size of the graphs and text in that image. This usually isn't an issue for you. However, if you run a SAS/GRAPH program in your local SAS session, and then run that same program from a Web server, the GIF images generated by the two program runs might use different sizes for the graphs and text. A program that is run from your local SAS session is likely to generate smaller graphs with larger text than what is generated from the very same program that is run from the Web server. The explanation for this can be quite complicated, but the way to prevent it from happening is simple.

To ensure that the size of graphs and text is consistent in the GIF images generated by a SAS/GRAPH program, regardless of where and how that program is run, do either of the following:

- □ specify the NOCHARACTERS option on a GOPTIONS statement, as in

  ```
  goptions nocharacters;
  ```

- □ set CHARACTERS=N in the device entry for the specified device driver (GIF, HTML, or WEBFRAME).

Use one of these techniques only for SAS/GRAPH programs that are run in different circumstances (for example, locally and from a Web server), and only if you want the sizing to be consistent across all runs. For example, you would want the sizing to be consistent if you were designing the graph in your local SAS session, and then you intended to copy your SAS program to a Web server so that it could be run from that server.

Specifying the NOCHARACTERS graphics option or setting CHARACTERS=N in the device entry does not explicitly specify sizing information, it simply determines that hardware fonts are not used for rendering the graphs and text. You should therefore specify in your SAS/GRAPH code any sizing information you want.

Using SAS/GRAPH Web Drivers to Create Web Output

SAS/GRAPH provides three device drivers that you can use to create graphics output for the Web: GIF, HTML, and WEBFRAME. The GIF driver creates GIF format files, and you have to create the HTML files to reference those files. The HTML and WEBFRAME drivers automatically write graphics output to one or more GIF files, and both create the HTML file(s) needed to display the GIF files. What distinguishes the HTML and WEBFRAME drivers is the layout they use to display the output:

HTML driver	WEBFRAME driver
Generates one HTML file that displays all the graphics output on a single page. To see the output, you can scroll to any graph on the page. *Note:* To get more control over the output, you can use ODS to generate Web output with this same layout.	Uses two frames to display graphs and, therefore, requires a browser that supports HTML frames. The left frame displays thumbnail-size versions of all the graphics output. To see a full-size version of the graph, select a thumbnail, and the full-size graph displays in the right frame. *Note:* The WEBFRAME driver is the only SAS/GRAPH driver that creates thumbnail-size graphs; you cannot get this same layout using ODS.

Producing GIF Files (DEV=GIF)

The GIF driver produces a GIF file for each graph that is generated when the GIF device is specified. The GIF files are created from the GRSEG entries generated by SAS/GRAPH procedure output. To see the output, you have to create the HTML file(s) needed to view the GIF files in a Web browser.

To direct the GIF output files, you can let SAS/GRAPH use the default output location, or you can use a FILENAME statement and the GSFNAME= graphics option to specify a location as follows:

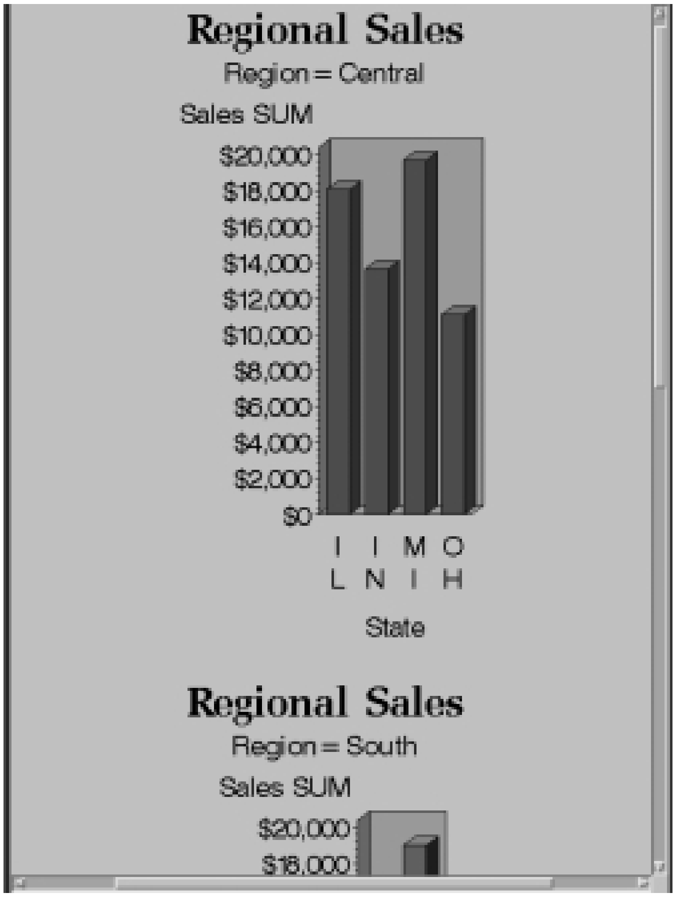
```
filename outfiles 'path-to-Web-server';
goptions gsfname=outfiles dev=gif;
```

For more details, see “Specifying Output Locations for HTML and GIF Files” on page 74. After a graphics procedure is run, the resulting GIF file can be used in any Web page.

Displaying Graphs on One Web Page (DEV=HTML)

The HTML device driver generates one HTML file and one or more GIF files. The GIF files are created from the GRSEG entries generated by SAS/GRAPH procedure output. To see the output, you use a browser to view the HTML file, which references the GIF file(s) created by the last graphics procedure that is run in the SAS program. If that procedure creates multiple GIF files, as would be the case with BY-group processing, the files are referenced consecutively so that they are displayed in sequence on a single Web page, as shown in Display 5.1 on page 78.

Display 5.1 Graphs Generated with DEVICE=HTML



To direct the HTML and GIF output files, you can let SAS/GRAPH use the default output location, or you can use a FILENAME statement and the GSFNAME= graphics option to specify a location as follows:

```
filename outfiles 'path-to-Web-server-space';
goptions gsfname=outfiles dev=html;
```

For more details, see “Specifying Output Locations for HTML and GIF Files” on page 74.

Each time a graphics procedure is run, the HTML device driver creates an HTML file named index.html. This is the file you view in a browser. You cannot change this name.

Note: To control the name of the output HTML files, use ODS to generate the Web output. For information on ODS, see “Using the Output Delivery System (ODS) with SAS/GRAPH Software” on page 81. △

The GIF files are always named by appending a .GIF extension to the name of the corresponding GRSEG entry. Thus, if you use a procedure’s NAME= option to name a GRSEG MYGRAPH, then the GIF file is assigned the name mygraph.gif. If you do not use the NAME= option, then the GIF file is assigned the GRSEG’s default name, such as gchart.gif. For more details, see “Naming Conventions Used for GIF Files” on page 75.

Note: Because SAS/GRAPH does not replace existing GRSEG entries when a procedure creates a new entry of the same name, you may want to use a GREPLAY procedure to delete existing GRSEG entries before generating the Web output. For details, see “Naming Conventions Used for GIF Files” on page 75. △

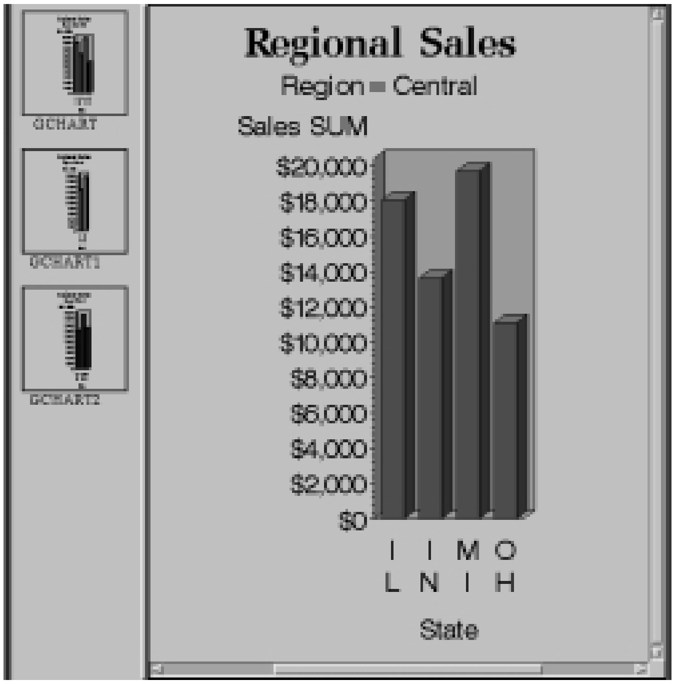
With DEV=HTML, the output from each run of a graphics procedure overwrites the existing index.html file; this is true even when you use RUN-group processing to generate the procedure runs. Thus, when you view index.html, only output from the last run of a graphics procedure is displayed. To see additional output, use the GREPLAY procedure as discussed in “Replaying Multiple Graphs for Web Output” on page 80.

For an example, see “Example 1: Using the HTML and WEBFRAME Device Drivers” on page 107. For an example on using the Web drivers to create drill-down graphs, see “Example 3: Using a Web Driver to Generate a Drill-down Graph” on page 111.

Linking to Graphs with Thumbnail-Size Images (DEV=WEBFRAME)

The WEBFRAME device driver generates multiple HTML files and multiple GIF files. The GIF files are created from the GRSEG entries that are generated by SAS/GRAPH procedure output. The HTML file that you view in a browser defines two frames for displaying the output, as shown in Display 5.2 on page 79. Thus, to display the output from the WEBFRAME device driver, you must have a browser that supports HTML frames.

Display 5.2 Graphs Generated with DEVICE=WEBFRAME



To direct the HTML and GIF output files, you can let SAS/GRAPH use the default output location, or you can use a FILENAME statement and the GSFNAME= graphics option to specify a location as follows:

```
filename outfiles 'path-to-Web-server-space';
goptions gsfname=outfiles dev=webframe;
```

For more details, see “Specifying Output Locations for HTML and GIF Files” on page 74.

The WEBFRAME device is designed for displaying multiple graphs. It creates these files:

index.html
: This is the file that you view in a browser or reference from another HTML page. It defines two frames for displaying procedure output. The left frame displays file sasthumb.html, and the right frame displays the graph selected in the left frame.

sasthumb.html
: This file references small, thumbnail versions of the GIF images that are created by the WEBFRAME device driver. Also, it links each thumbnail to its corresponding full-size GIF image. When file index.html is displayed in a browser, the thumbnails appear one under the other in the left frame. The name of the corresponding full-size image appears just below the thumbnail. When you click on a thumbnail, the full-size GIF image is displayed in the right frame.

GIF Files
: The WEBFRAME device driver creates a thumbnail/full-size image pair for each graph that is produced by the procedure output. The filename for the full-size image corresponds to the name in the GRSEG entry. The thumbnail image file is assigned the same name, but with the prefix "f". For information on the naming conventions that SAS/GRAPH uses for GRSEG entries, see “Naming Conventions Used for GIF Files” on page 75.

graphname.html
: Each full-size image has the name *graphname*.gif, where *graphname* corresponds to the name in the GRSEG entry. For each file *graphname*.gif, there is a corresponding HTML file named *graphname*.html, which simply references the GIF file. When a thumbnail image is selected in the left frame, the corresponding HTML file is loaded into the browser’s right frame, displaying the GIF image.

Note: Referencing the GIF files from an HTML file rather than displaying the GIF files directly gives you the capability of using the WEBFRAME driver to generate drill-down graphs (see “About Drill-down Graphs” on page 90). △

With DEV=WEBFRAME, the output from each run of a graphics procedure overwrites the existing index.html and sasthumb.html files; this is true even when you use run-group processing to generate the procedure runs. Thus, when you view index.html, only output from the last run of a graphics procedure is displayed. To see additional output, use the GREPLAY procedure as discussed in “Replaying Multiple Graphs for Web Output” on page 80.

For a complete example, see “Example 1: Using the HTML and WEBFRAME Device Drivers” on page 107. For an example on using the Web drivers to create drill-down graphs, see “Example 3: Using a Web Driver to Generate a Drill-down Graph” on page 111.

Replaying Multiple Graphs for Web Output

With DEV=HTML or DEV=WEBFRAME, the output from each run of a graphics procedure overwrites the existing index.html file (with DEV=WEBFRAME, the driver also overwrites the existing sasthumb.html). Thus, when you view index.html, only output from the last run of a graphics procedure is displayed.

Note: If the last run generates multiple graphs, as would be the case with BY-group processing, then all of the output from that run is referenced in the final write to index.html. △

To view output from multiple procedure runs, do both of the following:

1 specify PROC GREPLAY as the last graphics procedure run with DEV=HTML or DEV=WEBFRAME

2 use the REPLAY statement with PROC GREPLAY to replay all the output you want to view.

For example, assume a program has three separate procedure runs that generate catalog entries named CENTRAL, SOUTH, and WEST, in that order. Without a PROC GREPLAY, only the file west.gif is displayed when you view index.html. However, assuming that the entries are stored in the default output catalog, WORK.GSEG, if you run the following GREPLAY procedure, then all three GIF files will be displayed when you view index.html:

```
proc greplay igout=work.gseg nofs;
   replay central south west;
```

Note: You can replay any output from the catalog; you are not limited to replaying entries that were created in the current program. △

Using the Output Delivery System (ODS) with SAS/GRAPH Software

Using ODS with SAS/GRAPH has numerous advantages over generating Web output with the HTML or WEBFRAME device drivers. With ODS, you can

□ name the body file(s) for storing the ODS output (see "Specifying Body Files for Displaying Graphs" on page 83)

□ determine whether titles and footnotes are written as part of the graphs or as part of the HTML files (see "Controlling Titles and Footnotes with ODS Output" on page 85)

□ combine graphics and non-graphics output in your Web page (see "Adding Non-graphics Output to a Web Page" on page 85)

□ generate a Table of Contents to link to the output (see "Linking to Output through a Table of Contents" on page 86)

□ generate a Table of Pages to link to the output (see "Linking to Output through a Table of Pages" on page 87)

□ use HTML frames to display the Table of Contents or Table of Pages (see "Using Frames to Display ODS Output" on page 89)

□ create drill-down graphs (see "About Drill-down Graphs" on page 90)

□ create Java applets or ActiveX controls (see "Creating Java Applets and ActiveX Controls" on page 104).

These sections only briefly discuss ODS as it is used with SAS/GRAPH. For a more detailed discussion of ODS, see *The Complete Guide to the SAS Output Delivery System*.

At a minimum, to use ODS with SAS/GRAPH, you must do all of the following:

1 use a GOPTIONS statement to specify that DEVICE=GIF, DEVICE=JAVA, or DEVICE=ACTIVEX. For each GRSEG generated in the SAS program, the GIF device driver creates a GIF file, the JAVA device driver creates a Java applet, and the ACTIVEX device driver creates an ActiveX control.

   Note: The JAVA and ACTIVEX device drivers can only be used with the GCHART, GCONTOUR, GMAP, GPLOT, and G3D procedures. △

2 use the ODS HTML statement to specify the ODS options you want to use. At a minimum, you must use the BODY= (alias FILE=) option to specify a body file.

3 run a graphics procedure.

4 close the HTML destination by specifying ODS HTML CLOSE.

Operating Environment Information: In mainframe environments, you must also use the PATH= option to direct output files. In CMS, you must use the URL= suboption when you use the CONTENTS=, FRAME=, or PAGE= options. The URL= suboption specifies a valid URL for linking to other Web pages (CMS file specifications do not form valid URLs). △

Managing Links

With ODS, you can create output files within your file system, or create them directly on your Web server (assuming you can write files to that server). Either way, ODS builds the links for you. In this chapter, the examples are based on the assumption that you are creating all your output files in the same location within your file system. To deliver the output through a Web server, you can copy the files from your file system to the Web server.

To create the output files directly on the Web server for distribution across the Web, ODS needs to build valid URLs for the linking information. In this case, several options on the ODS HTML statement affect how ODS constructs the links and references. For details, see “How ODS Constructs Links and References” on page 210.

Managing ODS Destinations

ODS supports multiple destinations for procedure output. When using ODS with SAS/GRAPH, you manage two of those locations: the *listing* destination, which is the destination that receives traditional SAS output, and the *HTML* destination, which receives the HTML and GIF files needed for Web output.

ODS destinations can be open or closed. When a destination is open, ODS can send output to it, and when a destination is closed, ODS cannot send output to it. An open destination always uses system resources.

By default, the Listing destination is open, and the HTML destination is closed. Thus, when you do not specify the ODS HTML statement, procedure output is written to the Listing destination, and the output is rendered the same way it has been rendered in all earlier releases of the SAS System. Specifying the ODS HTML statement opens the HTML destination, but it has no effect on the Listing destination, which remains open unless you explicitly close it.

When using SAS/GRAPH procedures with ODS, you should conserve system resources by closing the Listing destination before issuing the ODS HTML statement. After generating ODS output, you must close the HTML destination before you can view that output. A typical ODS session with SAS/GRAPH should be structured like this:

```
/* specify the output location */
filename odsout 'path-to-Web-server';
goptions device=gif;

/* close the listing destination */
ods listing close;

/* open the html destination */
ods html path=odsout body='myfile.html';

/* SAS/GRAPH program code */
/* close the html destination */
ods html close;

/* open the listing destination */
ods listing;
```

If both the Listing and the HTML destinations are open when you use the GIF driver to generate graphics for the Web, the following output is generated:

HTML destination	Listing destination
The GIF driver creates a GIF file for each graph. Each file's name corresponds to the name of the GRSEG entry for the same graph.	The GIF driver creates a GIF file named *sasgraph.gif* and writes or appends to it (depending on the GSFMODE= setting) each time it generates a graph.
All the HTML files specified on the ODS HTML statement are created. Only the GIF files that are created in the HTML destination are referenced in the HTML files.	

Output in the Listing destination is superfluous for Web use for the following reasons:

- □ The file sasgraph.gif is not referenced in an HTML file.
- □ Previewing GRSEGs in the GRAPH window is not a reliable way to proofread the graphs for use with the Web because the GRAPH window and a Web browser render graphs differently.

Note: For more information on ODS destinations, see *The Complete Guide to the SAS Output Delivery System*. △

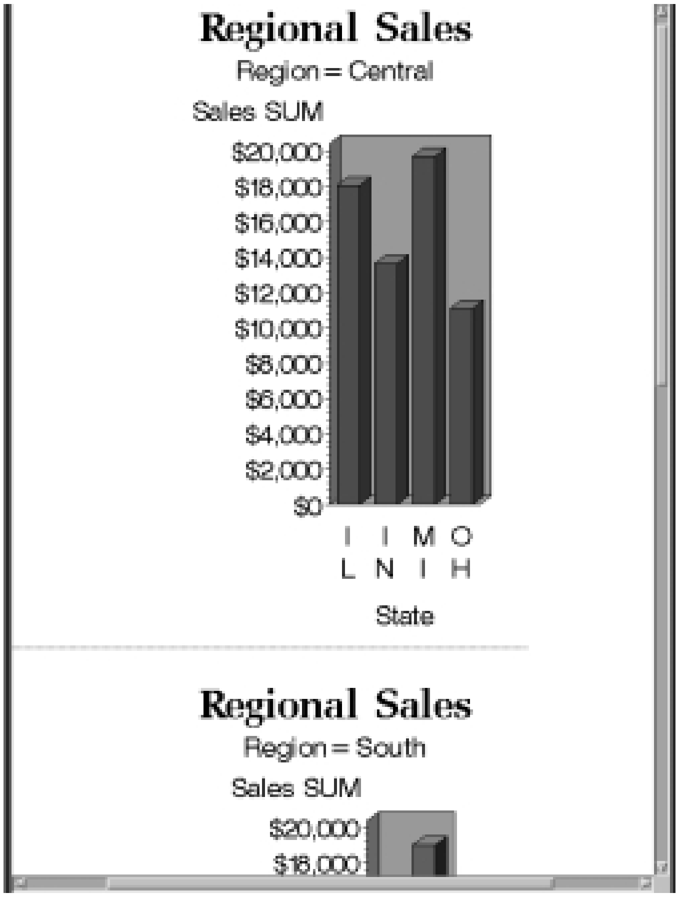
ODS and Procedures that Support RUN-Group Processing

When you use ODS, it is wise to specify a QUIT statement at the end of every procedure that supports RUN-group processing. If you end every procedure step explicitly, rather than waiting for the next PROC or DATA step to end it for you, the QUIT statement will cause the selection list to clear, and you will be less likely to encounter unexpected results.

Specifying Body Files for Displaying Graphs

When you use ODS with SAS/GRAPH, you specify a body file to reference the graphics output. A body file is simply an HTML file that is created by ODS to contain non-graphics output and to reference graphics output so that it displays as if it were part of the HTML file. You can use many body files during the SAS session, although only one at a time can be open. Display 5.3 on page 84 shows a body file that references charts that were generated by PROC GCHART.

Display 5.3 Displaying a Body File in a Browser



To open a body file, use the ODS HTML statement's BODY= option. The following code creates a body file named sales.html, which is created in the output location specified on the FILENAME statement (see "Specifying Output Locations for HTML and GIF Files" on page 74). A more complete ODS example is shown in "Example 2: Using ODS with SAS/GRAPH Software" on page 109.

```
filename odsout 'path-to-Web-server';
goptions device=gif;
ods html body='sales.html' path=odsout;
```

The body file remains open and all graphics and non-graphics output is written to it until the HTML destination is closed or another body file is opened.

To direct output to multiple body files, use an ODS HTML statement with the BODY= option each time you want to close the current body file and open another:

```
filename odsout 'path-to-Web-server';
goptions device=gif;

ods html body='sales.html' path=odsout;

/* code whose output goes to sales.html */

ods html body='costs.html' path=odsout;

/* code whose output goes to costs.html */
```

```
ods html close;
```

If you use BY-group processing on a graphics procedure, a separate graph is generated for each value of the BY variable. In that case, all the graphs will be referenced in the same body file, unless you use the ODS HTML statement's NEWFILE= option. For example, you might use NEWFILE=OUTPUT:

```
 /* use newfile= to open a new */
 /* body file for each graph */
filename odsout 'path-to-Web-server';
goptions device=gif;
ods html body='sales.html' path=odsout
    newfile=output;
```

NEWFILE=OUTPUT opens a new body file for each new graph that is generated, whether the graphs are generated with BY-group processing or by multiple procedure runs. The new body files are named by appending consecutive numbers to the name you specify in the BODY= option. In the example above, the initial body file is named sales.html, and the additional body files that are created will be named sales1.html, sales2.html, and so on.

To determine the appearance of output on the Web page, the body file uses table definitions and style definitions. This document shows output with the default definitions. Other definitions are available with the STYLE= option. You can also create your own style definitions. For information on table definitions and style definitions, see *The Complete Guide to the SAS Output Delivery System*.

Controlling Titles and Footnotes with ODS Output

With ODS, if DEVICE=JAVA or DEVICE=ACTIVEX, titles and footnotes are always stored as part of the HTML file. However, if DEVICE=GIF, titles and footnotes are stored by default as part of the image in the GIF file.

With DEVICE=GIF, you can direct titles and footnotes to the body file with the NOGTITLE and NOGFOOTNOTE options in the ODS HTML statement. When titles and/or footnotes are directed to the body file, they do not appear in the GIF file image.

```
/* direct titles and footnotes */
/* to the html file            */
filename odsout 'path-to-Web-server';
goptions device=gif;
ods html body='sales.html' path=odsout
    nogtitle nogfootnote;
```

Titles and footnotes in the body file are rendered as separate HTML tables near the top and bottom of each page of HTML output.

Note: With NOGTITLE and NOGFOOTNOTE, if either titles or footnotes take up much more vertical space than the other, the text on a vertical axis may not be centered properly relative to the axis tick marks. If this happens, you may not want to specify the option. △

Adding Non-graphics Output to a Web Page

When you open a body file in ODS, all graphics and non-graphics output is referenced in that body file until the HTML destination is closed or another body file is opened. Thus, you do not have to do anything special to combine graphics and non-graphics output. Simply open a body file and run the procedures whose output you want to combine:

```
filename odsout 'path-to-Web-server';
goptions device=gif;

/* close the listing destination */
ods listing close;

/* open html destination and a body file */
ods html body='sales.html' path=odsout;

/* generate graphics and */
/* non-graphics output   */
/* using the current data set */
proc gchart;
     vbar3d state / sumvar=sales;
run;
quit;

proc print noobs;
run;

/* close the html destination,  */
ods html close;
/* open the listing destination */
ods listing;
```

In this example, the GCHART procedure output is referenced in the body file, and then the PRINT procedure output is written below it. The two outputs appear together on the same Web page when file sales.html is viewed in a browser.

For a more complete example, see “Example 9. Combining Graphs and Reports in a Web Page” on page 287.

Linking to Output through a Table of Contents

With ODS, you can create a contents file to link to the graphics and non-graphics output generated during a SAS session. A contents file is simply a file that uses a Table of Contents to link to the output. You can use multiple contents files during the SAS session, although only one at a time can be open.

Display 5.4 Displaying a Contents File in a Browser

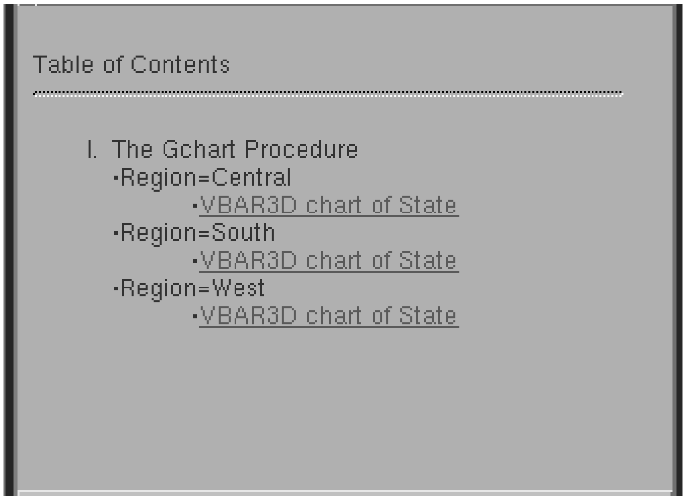


Table of Contents

I. The Gchart Procedure
- Region=Central
  - VBAR3D chart of State
- Region=South
  - VBAR3D chart of State
- Region=West
  - VBAR3D chart of State

To create a contents file, specify a name for the file in the ODS HTML statement's CONTENTS= option.

The following code creates a contents file named salesCon.html, which is created in the output location specified in the FILENAME statement (see "Specifying Output Locations for HTML and GIF Files" on page 74). A more complete ODS example is shown in "Example 2: Using ODS with SAS/GRAPH Software" on page 109.

```
filename odsout 'path-to-Web-server';
goptions device=gif;
ods html body='sales.html' path=odsout
    contents='salesCon.html';
```

The contents file remains open and links are written to it for all graphics and non-graphics output that is generated by the SAS program until one of the following occurs:

□ The HTML destination is closed.

□ Another contents file is opened. To open a new contents file, specify another ODS HTML statement and use CONTENTS= to specify the new file's name.

For graphics procedures, use the DESCRIPTION= option to specify the text to be displayed for the links to that procedure's output. If you don't use the DESCRIPTION= option, the procedure's default description text is used.

To use the Table of Contents, view the contents file in the browser. When you select a link from the Table of Contents, the browser goes to the target output referenced by that link. To use the Table of Contents again, you must use the browser's Back button or some other mechanism to return to the contents page. If your browser supports HTML frames, you can keep the Table of Contents visible and its links accessible at all times by displaying the contents page in a frame (see "Using Frames to Display ODS Output" on page 89).

Linking to Output through a Table of Pages

With ODS, you can create a page file to link to the graphics and non-graphics output generated during a SAS session. A page file is simply a file that uses a Table of Pages (page references) to link to output. You can use multiple page files during the SAS session, although only one at a time can be open.

Display 5.5 Displaying a Page File in a Browser

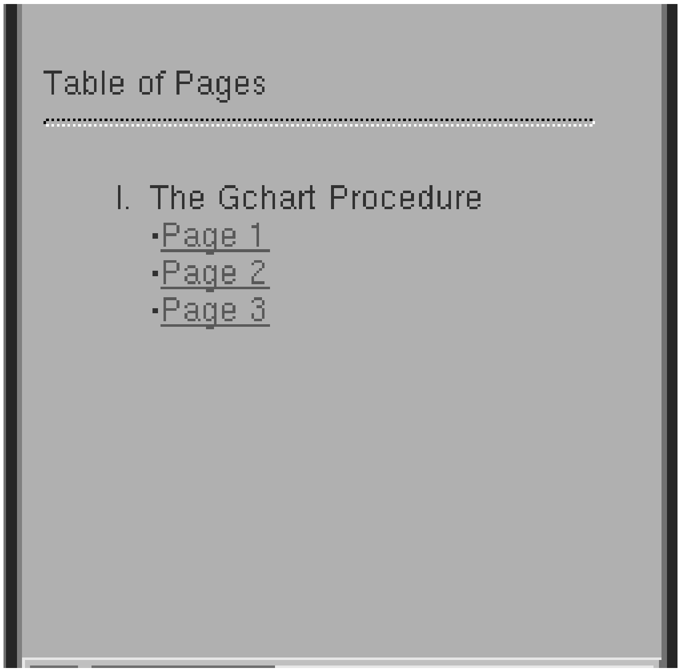


Table of Pages

I. The Gchart Procedure
- Page 1
- Page 2
- Page 3

To create a page file, specify a name for the file in the ODS HTML statement's PAGE= option.

The following code creates a page file named salePage.html, which is created in the output location specified in the FILENAME statement (see "Specifying Output Locations for HTML and GIF Files" on page 74). For a more complete ODS example, see "Example 2: Using ODS with SAS/GRAPH Software" on page 109.

```
filename odsout 'path-to-Web-server';
goptions device=gif;
ods html body='sales.html' path=odsout
    page='salePage.html';
```

The page file remains open and links are written to it for all graphics and non-graphics output that is generated by the SAS program until one of the following occurs:

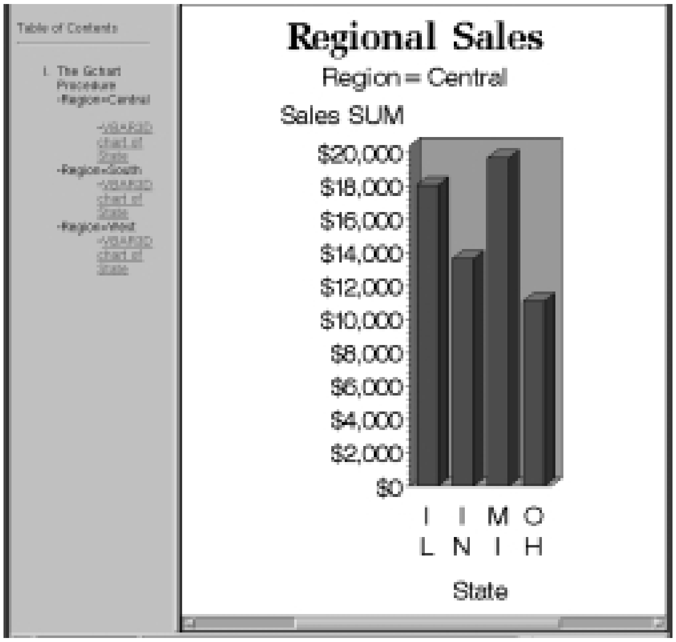
- □ The HTML destination is closed.
- □ Another page file is opened. To open a new page file, specify another ODS HTML statement and use PAGE= to specify the new file's name.

To use the Table of Pages, view the page file in the browser. When you select a link from the Table of Pages, the browser goes to the target output referenced by that link. To use the Table of Pages again, you must use the browser's Back button or some other mechanism to return to the page file. If your browser supports HTML frames, you can keep the Table of Pages visible and its links accessible at all times by displaying the page file in a frame.

Using Frames to Display ODS Output

With ODS, you can create a frame file to display a Table of Contents, a Table of Pages, or both. A frame file is simply a file that uses two frames: one to reference a contents file, a page file, or both and a second to display output that is selected from the table of contents or pages. To use a frame file, your browser must support HTML frames. Display 5.6 on page 89 shows a frame file that is displaying a Table of Contents.

Display 5.6 Displaying a Frame File in a Browser



To create a frame file, specify a name for the file on the ODS HTML statement's FRAME= option. You must also use the options CONTENTS=, PAGE=, or both to provide a table to display in the left frame.

The following code creates a frame file named saleFram.html, which is created in the output location specified in the FILENAME statement (see "Specifying Output Locations for HTML and GIF Files" on page 74). A more complete ODS example is shown in "Example 2: Using ODS with SAS/GRAPH Software" on page 109.

```
filename odsout 'path-to-Web-server';
goptions device=gif;
ods html body='sales.html' path=odsout
    contents='saleCon.html'
    frame='saleFram.html';
```

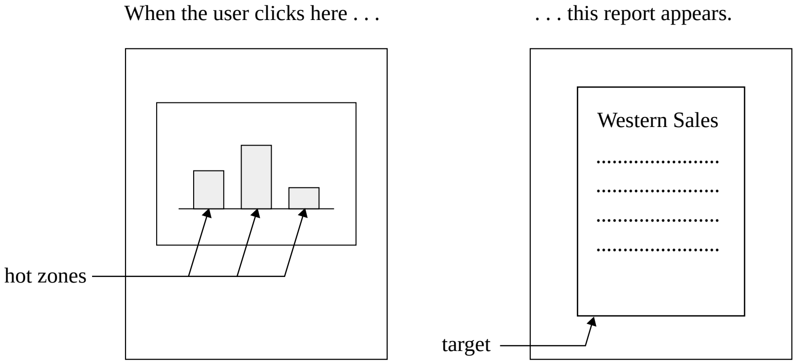
To use the frame file, view the frame file in the browser. When you select a link from the Table of Contents or Table of Pages, the content of the right frame changes to display the output that is the target of the selected link, but the links from the contents or pages remain accessible in the left frame.

About Drill-down Graphs

A drill-down graph has designated areas or *hot zones* that are linked to a target output, such as other graphs or reports. The target output can be in a different location on the same Web page, or it can be on a different Web page. If the target output contains another graph, that graph can also have hot zones that link to additional targets. In this way, you can establish a series of links that drill down to more and more detailed information.

Figure 5.2 on page 90 shows a bar chart with three bars. Assume that the bars represent a company's regional sales, and that the third bar represents the western region. If the chart is a drill-down graph, with each bar linking to a report on the corresponding region's sales, then selecting the third bar drills down to a report on sales for the western region.

Figure 5.2 Drill-down on a Bar Chart



The hot zones in a drill-down graph can be the graphic areas (for example, the bars in a bar chart or slices in a pie chart) or the pattern boxes that identify legend entries. SAS/GRAPH defines the hot zones for you, based on the specifications you make on the graphics procedure that is used to produce the graph. You use the HTML= option to make hot zones in the graphic areas and the HTML_LEGEND= option to make hot zones in the legend. You provide the link information by storing the paths to target output in a special HTML variable, which is usually created in the data set that is used to generate the graph.

Only procedures that support the IMAGEMAP= option, or the HTML= or HTML_LEGEND= options, can be used for drill-down graphs, although you can also generate drill-down graphs using the DATA Step Graphics Interface (DSGI). Table 5.2 on page 91 lists those procedures and shows where the HTML variable that stores the linking information is specified for each of them.

Table 5.2 Procedures that Support Drill-down Graphs

Procedure	Location where HTML variable is supported
PROC GANNO	Supports the HTML variable from the Annotate data set that is used to generate the graph.
PROC GCHART	The following statements support the HTML= or HTML_LEGEND= option: □ BLOCK statement □ DONUT statement □ HBAR, HBAR3D statements □ PIE, PIE3D statements □ STAR statement □ VBAR, VBAR3D statements.
PROC GMAP	The following statements support the HTML= or HTML_LEGEND= option: □ BLOCK statement □ CHORO statement □ PRISM statement.
PROC GPLOT	The following statements support the HTML= or HTML_LEGEND= option: □ PLOT statement □ PLOT2 statement.
PROC GREPLAY	Replays a drill-down graph from the graphics catalog, so the HTML variable must be specified on the HTML= or HTML_LEGEND= option on the procedure that stores the original graph in the catalog.
PROC GSLIDE	Supports the HTML variable from the Annotate data set that is used to generate the graph.

Ways You Can Generate a Drill-down Graph

You can generate a drill-down graph using the SAS/GRAPH Web drivers or using ODS. The approach you should use depends on your Web-page or site design.

Using Web Drivers to Generate Drill-down Graphs

To link to individual graphs, it is easiest to use the Web drivers to generate the drill-down graphs. The Web drivers create all the files you need and require the fewest specifications in your SAS code. For an example, see “Example 3: Using a Web Driver to Generate a Drill-down Graph” on page 111.

The Web drivers do not assign anchor names to the output they generate. Thus, they are not useful for linking to multiple pieces of output that are stored or referenced in one HTML file. Also, you cannot use the Web drivers to generate drill-down graphs in a SAS program that also generates ODS output; they are designed as an alternative to using ODS.

Using ODS to Generate Drill-down Graphs

To link to both graphics and non-graphics output, or to link to multiple graphs that are stored or referenced in a single file, it is easiest to use ODS to generate the drill-down graphs. For an example, see “Example 4: Using ODS to Generate a Drill-down Graph” on page 115. Also, if you want to design the attributes of your Web page, ODS enables you to use style definitions. For example, you can set the fonts and color for text. For information on style sheets, see *The Complete Guide to the SAS Output Delivery System*.

To generate a drill-down graph with ODS, your main responsibility is to create the required HTML variables and assign to those variables the appropriate linking information in the form "*URL#anchor-name*".

URL The location and name of the HTML file that contains the target. When you use ODS, the URL is composed of information you specified on the ODS HTML statement options BASE=, PATH= or GPATH=, and BODY= (for more information, see “How ODS Constructs Links and References” on page 210).

anchor name The target’s location within the HTML file. An anchor name must be unique within an HTML file. You can let ODS assign default anchor names, or you can specify anchor names in the ODS HTML statement’s ANCHOR= option. If a target is located at the top of the HTML file that contains it, the anchor name is optional on the HREF value because, by default, a Web browser displays information that is at the top of the HTML file.

ODS always assigns anchor names to the output it generates; that way, the output can be used as the target to a link, no matter where the output is located. It does not matter whether the output is all written to the same HTML file or to separate HTML files. The default system for assigning anchor names always starts with IDX for the first piece of output that is generated after the HTML destination is opened and increments that name (IDX1, IDX2, ...) for each piece of output that is written until the HTML destination is closed. If the HTML destination is opened again during the current SAS session, the default naming scheme starts over again, beginning with IDX.

The value that you assign to an HTML variable must be the HREF value that is needed for the HTML link. Therefore, to define a link for a drill-down graph, you must know what URL is needed to access the target output and what anchor name ODS assigns to that output.

While planning your Web page, you have to decide whether to let ODS assign default anchor names, or whether to control anchor names yourself using the ODS HTML statement’s ANCHOR= option. Generally, the default anchor names will suffice for simple drill-down graphs or when each link target contains only a single piece of output in its own HTML file. You may want to control anchor names when you use multiple procedures to generate the target output because it will be easier to keep track of the anchor names. You may also want to control anchor names when you open a body file to append output to it; in this case, controlling anchor names prevents ODS from using the same default anchor names as were used when the file was first created.

Customizing a Web Page for a Drill-down Graph

To get complete control over Web page design, you can customize a Web page for use with a drill-down graph by writing your own HTML files, letting SAS/GRAPH generate the drill-down graph. To customize a Web page, you must know how to

□ write the HTML tags needed to design the Web page

□ implement HTML links

□ use an image map to implement a drill-down graph.

However, you do not have to know how to create an image map, because SAS/GRAPH does that for you. For details, see "Customizing Web Pages for Drill-down Graphs" on page 100. For an example, see "Example 5: Customizing a Web Page with a Drill-Down Graph" on page 119.

General Requirements for Generating a Drill-down Graph

Generally, to create a drill-down graph, you provide the link information needed to connect the graph to its related output. Your SAS program must do the following:

□ create an HTML variable to hold HTML link locations. Typically, for each drill-down level, you create one HTML variable to store the links for the graphic areas; to create links from the legend, you can use that same variable or a separate variable to store links for the legend values. For example, to implement the graph in Figure 5.2 on page 90, you need only one HTML variable for the bars in the chart.

□ assign values to the HTML variable(s). These values are the link locations that connect the graph to its target output.

□ run a SAS/GRAPH procedure to create the drill-down graph(s). This procedure must use an HTML= option, an HTML_LEGEND= option, or both to associate the HTML variable(s) with the graph. Or, if the drill-down capability is being implemented from an Annotate data set, the Annotate data set's HTML variable must specify the link locations (for an example, see Example 4 on page 514).

  If you are using the HTML or WEBFRAME device drivers, or if you are writing your own HTML files, you must also use the IMAGEMAP= option to create a data set that can be used to generate an image map for the drill-down capability.

□ run the procedures that create the target output. The target output must be directed to the locations identified by the HTML variable.

Because an HTML variable stores the HTML link locations for target output, you must understand HTML links in order to assign values to it. You should also know what an image map is, because SAS/GRAPH uses image maps to attach HTML links to drill-down graphs (see "Image Maps in Drill-down Graphs" on page 94).

HTML Links in Drill-down Graphs

In HTML files, links are specified as HREF attributes on tags that support linking. The HREF value must point to the location of the link target, which is identified by a URL or a URL and an anchor name in the form

```
HREF="URL#anchor-name"
```

URL
: The location and name of the HTML or GIF file that contains the target.

anchor name
: The target's location within the HTML file. An anchor name must be unique within an HTML file. If a target is located at the top of the HTML file that contains it, the anchor name is optional on the HREF value because, by default, a Web browser displays information that is at the top of the HTML file.

In Figure 5.2 on page 90, the bars represent regional sales, and the third bar represents the western region. If the third bar drills down to a report, and if that report is stored in your file system at the top of a file named west.html, then the following HREF points to the report:

```
 HREF="west.html"
```

Here, only a filename is given. The file west.html is, therefore, assumed to be in the same storage location as the HTML file where the link is initiated. The anchor name is omitted because the report is at the top of file west.html, so the anchor name isn't needed. However, if the report is not at the top of the file, the HREF must specify the anchor name.

In an HTML file, anchor names are assigned on the NAME attribute of an <A> (anchor) tag. In the following example, the anchor name **west** is assigned to the table that follows it:

```
<A NAME="west" ></A>
<P>
<TABLE><TR><TD>
Western Sales
</TD></TR></TABLE>
```

If the report is stored in a file named reports.html, as is the case for Figure 5.2 on page 90, then the following HREF points to the report at the west location:

```
 HREF="reports.html#west"
```

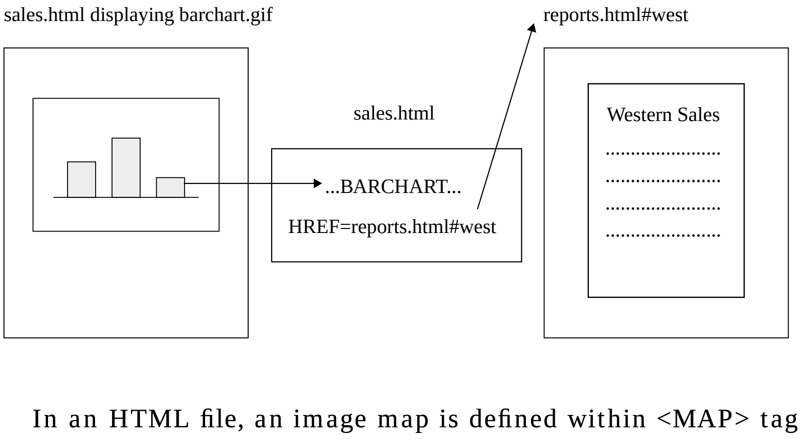
More HREF= values are shown in "Assigning Values to HTML Variables" on page 97.

For information on how ODS constructs HTML links and references, see "How ODS Constructs Links and References" on page 210.

Image Maps in Drill-down Graphs

Image maps are defined in HTML files and are used to implement drill-down capability for GIF images (Java applets and ActiveX controls use their own internal mechanisms). An image map defines the graphic areas that you can select to link to other locations. For example, in Figure 5.3 on page 94, the third bar in the image barchart.gif links to a report that is stored in the file reports.html, which is located at anchor name west. To implement the drill-down capability, an image map must define that third bar as a hot zone by identifying its coordinates and linking those coordinates to the report's location.

Figure 5.3 Links in Drill-down Graphs



In an HTML file, an image map is defined within <MAP> tags, which are used to encompass <AREA> tags that define the areas that will serve as hot zones for the links. Each <AREA> tag has the following attributes:

- □ a SHAPE attribute to identify the area either as a rectangle (RECT) (for example, any of the bars in Figure 5.3 on page 94) or a polygon (POLY) (for example, a three-dimensional bar from a bar chart)
- □ a COORDS attribute to identify the coordinates of the hot zone
- □ an HREF attribute to define the link location of the target output.

For example, the chart shown in Figure 5.3 on page 94 is stored in file barchart.gif, which is referenced in an HTML file named sales.html. To give barchart.gif drill-down capability, sales.html must contain HTML tags to define an image map that is assigned to barchart.gif. The image map must have <AREA> tags that identify the coordinates of each of the three bars in the chart, and that define a link target for each bar. The image map is assigned to the graph on the USEMAP attribute of the <IMG> tag that references the graph's GIF file.

In the following image map, the HREF attributes link each bar to an HTML file named reports.html. The target output for each bar is directed to a different anchor location; the anchor names for the target output are central, south, and west. File reports.html is in the same location as the file that contains the image map, because no other location is identified.

```
<MAP NAME="BARCHART">
<AREA SHAPE=RECT COORDS="424, 143, 470, 256"
      HREF="reports.html#central">
<AREA SHAPE=RECT COORDS="366, 175, 412, 256"
      HREF="reports.html#south">
<AREA SHAPE=RECT COORDS="308, 106, 354, 256"
      HREF="reports.html#west">
</MAP>
<P>
<IMG SRC="barchart.gif" USEMAP="#BARCHART">
```

When you select a bar in the graph, the browser uses the image map to find the correct report to link to, as shown in Figure 5.3 on page 94.

The image map for a particular graph must be defined in the same HTML file that contains the <IMG> tag that references that graph. However, the map can be located anywhere in the file. For example, the image map can be defined at the very end of the HTML file, even though the <IMG> tag for the graph that uses the map is at the top of the HTML file.

SAS/GRAPH automatically generates image maps; you do not have to know how to create an image map, and you do not have to supply values for its SHAPE or COORDS attributes. When you use ODS, SAS/GRAPH writes the image map directly in the HTML file that contains the reference to the drill-down graph. When you use the Web drivers to generate the drill-down graph, SAS/GRAPH stores the image map information in an Imagemap data set (see "About the Imagemap Data Set" on page 95).

Whether you use ODS or the Web drivers to implement a drill-down graph, you must supply HREF values that identify the locations of the target output. You provide these values by assigning them to an HTML variable and specifying that variable on the graphics procedure that produces the drill-down graph. For details, see "Creating HTML Variables" on page 97.

About the Imagemap Data Set

To create a drill-down graph using the SAS/GRAPH Web drivers, you specify the IMAGEMAP= option on the PROC statement of the procedure that generates the drill-down graph. The following graphics procedures have the IMAGEMAP= option:

GANNO

GCHART

GMAP

GPLOT

GREPLAY

GSLIDE

Note: With ODS, you do not use the IMAGEMAP= option, and SAS/GRAPH does not create an Imagemap data set. △

The name of the data set can be any valid SAS data set name. The code that specifies it should resemble the following:

```
proc gchart data=regsales imagemap=salemap;
  vbar3d region / sumvar=sales
   patternid=midpoint
   html=links name='htmldril';
```

Here, the IMAGEMAP= option names the data set SALEMAP. The HTML= and/or HTML_LEGEND= option must also be used on the procedure to identify the HTML variable that contains the linking information (see "Creating HTML Variables" on page 97).

The IMAGEMAP= option creates an Imagemap data set, which contains information about the hot zones in the graph. If you generate your output with the HTML or WEBFRAME device drivers, the driver automatically uses the data set to implement the drill-down capability. If you customize your own Web pages (see "Customizing Web Pages for Drill-down Graphs" on page 100), you specify the Imagemap data set as an argument on the IMAGEMAP macro that writes the image map to your HTML file:

```
%imagemap(salemap, vbar);
```

This call to the macro assumes that VBAR is the fileref for the HTML file that references the drill-down graph. For more information on the IMAGEMAP macro, see "About the IMAGEMAP Macro" on page 102.

The Imagemap data set contains the following variables:

GRAPH
: a character variable (length 8) that contains the name of the graph. The name of the graph is the name of the catalog entry.

LENGTH
: a numeric variable (length 4) that contains the length of the LINK variable.

LINK
: a character variable (maximum length 1024) that contains the values from the input data set variables specified by the HTML= and HTML_LEGEND= options. This information specifies the action taken when the corresponding polygon or legend entry is selected. It may be an actual HREF value or other information that the application can use to associate the area with the desired reference.

NXY
: a numeric variable (length 4) that contains the number of coordinates (*x*,*y* pairs) in the area. If SHAPE='RECT', then NXY has a value of 2; the first pair of coordinates are the lower- left corner and the second pair of coordinates are the upper- right corner. If SHAPE='POLY', NXY contains the actual number of points in the polygon. The POLY shape is a polygon with a maximum of 100 vertices.

SHAPE
: a character variable (length 4) that describes the shape of the areas in the graph. The value of SHAPE is either 'POLY' or 'RECT'. For example, for a bar chart or a legend, the value of SHAPE is RECT; for a pie chart or map, the value of SHAPE is POLY.

X1...X100 numeric variables (length 4) that contain the *x* coordinates of the shape.

Y1...Y100 numeric variables (length 4) that contain the *y* coordinates of the shape.

Creating HTML Variables

HTML variables are used with ODS to create graphs with drill-down capability. The HTML variable is a character variable that stores the HTML link locations for target output. You must create the HTML variables by adding them to the data set that is used to produce the drill-down graph. Typically, you create the HTML variables in a DATA step.

The following code fragment creates the HTML variable RPT and assigns it a length of 40 characters so that it can store a long string:

```
 /* create data set REGSALES */
data regsales;
  input Region State Sales;
/* create the HTML variable */
  length rpt $40
```

Assigning Values to HTML Variables

Drill-down graphs use image maps to define hot zones in GIF graphs and to identify each hot zone's link target. A hot zone is defined on an <AREA> tag, and the zone's link target is specified as a value on the <AREA> tag's HREF attribute (see "Image Maps in Drill-down Graphs" on page 94). SAS/GRAPH automatically creates the image maps for you, but you must provide the HREF values by first assigning those values to an HTML variable and then specifying that variable on an HTML= or HTML_LEGEND= option on the procedure that produces the drill-down graph.

The HTML variable must be a character variable, so the value you assign to it must be a string. Always begin the string with **href=** followed by the link location in the form "*URL#anchor-name*". The following code assigns a valid value to an HTML variable named RPT:

```
RPT=’href="reports.html#west"’;
```

Note: To form valid HTML, the URL and anchor name must be enclosed in double quotation marks, as shown in the example. △

Using the RPT variable's value in an image map, SAS/GRAPH can build an <AREA> tag that resembles this:

```
<AREA SHAPE="RECT" COORDS="424,143,470,256"
      HREF="reports.html#west">
```

To specify an HREF value correctly, you must know in advance the correct location to specify for the link target. This requires you to plan your SAS program. You need to decide such issues as

- □ whether to write the link-target output to one or to multiple files.
- □ whether your users will access the output through the Web or through your file system. To provide access through the Web, you need to provide complete URLs for the links. To provide access through your file system, you only need to provide file specifications.

Table 5.3 on page 98 shows the forms you can use to assign values to an HTML variable.

Table 5.3 Valid Forms for the HTML Variable's Values

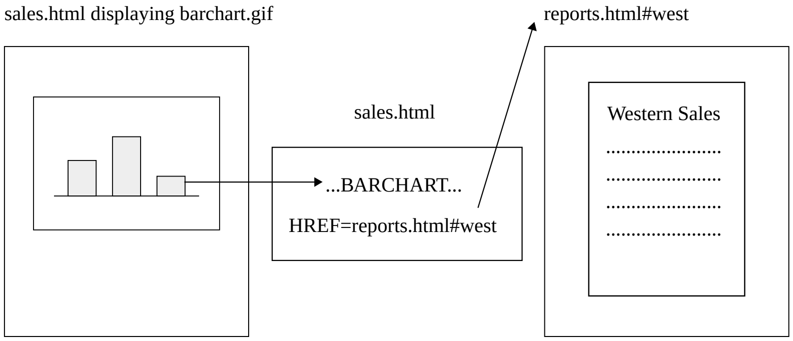
Value Assigned to an HTML Variable	Implications
'HREF="<*path*>reports.html"' 'HREF="<*path*>reports.html#west"'	Tells the browser to look in the file reports.html. If <*path*> is not provided, the file must be in the same location as the HTML file that initiates the link. For the value that has #west, go to the output with the anchor name west. Users must have access to your file system in order to access the link target.
'HREF="http://www.company.com/web/reports.html"' 'HREF="http://www.company.com/web/reports.html#west"'	Tells the browser to go to the Web site address http://www.company.com and look in the file web/reports.html. For the value that has #west, go to the output with the anchor name west. Users must have access to the Web.
'HREF="#west"'	Go to the output with the anchor name west. The target output must be referenced or contained in the same HTML file as the drill-down graph that contains the link. For example, if the drill-down graph is in file sales.html, then the target output must be referenced or contained in sales.html. Because this target output is in the same file as the drill-down graph, this link will work whether the output is viewed within your file system or across the Web.

Typically a drill-down graph has multiple hot zones, each of which links to different output. This means your HTML variable needs different values, based on the criteria used to produce the graph. For example, if the bars in a bar graph represent sales regions, then the values of the HTML variable will be based on those sales regions.

To assign values to the HTML variable, you can include the variable in the DATA step's INPUT statement and specify the values in the DATALINES statement. Or you can specify the values programmatically, perhaps in a conditional block such as an IF/THEN statement.

In Figure 5.4 on page 99, assume that the three bars represent regional sales for a company's central, southern, and western regions.

Figure 5.4 Links in Drill-down Graphs

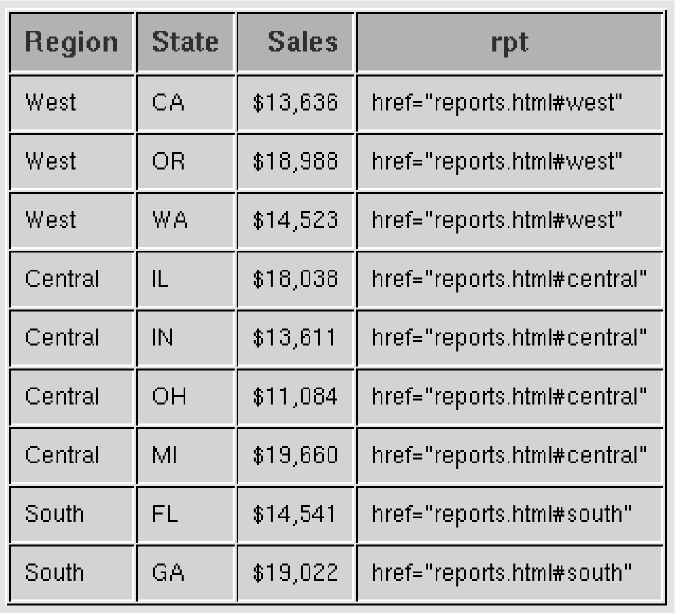


To link each bar to a report on that region's sales figures, you need one HTML variable to store the HREF links. The HTML variable needs to point to a different location for each of the three regions. If you decide to write all the reports to a file named reports.html that your users will access through your file system, and if the reports have anchor names central, south, and west, you might write a DATA step that resembles the following:

```
/* create data set REGSALES */
data regsales;
   length Region State $ 8;
   format Sales dollar8.;
   input Region State Sales;
   length rpt $40; /* the HTML variable */

/* assign HREF values to HTML variable */
if Region='Central' then
     rpt='HREF="reports.html#central"';
  else if Region='South' then
     rpt='HREF="reports.html#south"';
  else if Region='West' then
     rpt='HREF="reports.html#west"';

  datalines;
West CA 13636
West OR 18988
West WA 14523
Central IL 18038
Central IN 13611
Central OH 11084
Central MI 19660
South FL 14541
South GA 19022
;
```

Display 5.7 on page 100 shows the values in the data set REGSALES.

Display 5.7 Values in the REGSALES Data Set



Region	State	Sales	rpt
West	CA	$13,636	href="reports.html#west"
West	OR	$18,988	href="reports.html#west"
West	WA	$14,523	href="reports.html#west"
Central	IL	$18,038	href="reports.html#central"
Central	IN	$13,611	href="reports.html#central"
Central	OH	$11,084	href="reports.html#central"
Central	MI	$19,660	href="reports.html#central"
South	FL	$14,541	href="reports.html#south"
South	GA	$19,022	href="reports.html#south"

To use the HTML variable to create hot zones on the bars of the drill-down chart, you must specify it in the HTML= option of the graphics procedure that produces the graph.

See also:

□ "Example 3: Using a Web Driver to Generate a Drill-down Graph" on page 111

□ "Example 4: Using ODS to Generate a Drill-down Graph" on page 115

□ "Customizing Web Pages for Drill-down Graphs" on page 100 and "Example 5: Customizing a Web Page with a Drill-Down Graph" on page 119.

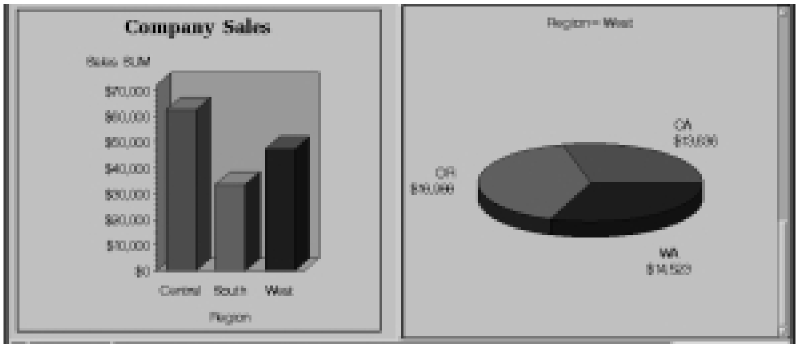
Customizing Web Pages for Drill-down Graphs

If you use the HTML or WEBFRAME driver to generate a drill-down graph, you have no control over the page design. The HTML driver always displays graphs on a single Web page and requires you to scroll the page if multiple graphs are produced by the last graphics procedure that is run. The WEBFRAME driver always displays thumbnail graphs in a left frame and uses the right frame to display the output that corresponds to a thumbnail that is selected in the left frame.

If you use ODS to generate a drill-down graph, you can use style sheets to control some of the page attributes, but you are limited to the layouts that ODS offers. Although you can use a frame to display a Table of Contents, a Table of Pages, or both, you must display the drill-down graph on a body page. When you drill down to target output, the target output will display in the body page, replacing the drill-down graph.

If you have a good working knowledge of HTML, and you understand how to generate drill-down graphs in SAS/GRAPH, you can customize the layout of Web pages that implement the drill-down graphs. For example, Display 5.8 on page 101 shows a Web page that cannot be produced with the HTML or WEBFRAME drivers or with ODS; however, you can customize the page with help from SAS/GRAPH. On this Web page, the left frame displays a drill-down bar chart showing a company's regional sales. When one of the bars in the chart is selected, the right frame displays a pie chart showing the total sales for the states in the corresponding region.

Display 5.8 Customized Web Page with Drill-down Graph



For a customized Web page, SAS/GRAPH produces the GIF files and an image map for you. You provide the linking information needed to point to the target output, and you create the HTML files and write the HTML tags needed to display the output and use the image map.

Thus, customizing the Web page differs in the following ways from using another method to generate drill-down graphs:

Using the HTML driver, the WEBFRAME driver, or ODS	Customizing the Web Page
The HTML files are created for you, and the HTML tags needed to implement the drill-down graph are automatically generated.	You create the HTML files and write the HTML tags needed to display your output and use the image map.
You have limited control over the structure of the HTML that is generated.	You have complete control over the Web-page layout.

For a SAS/GRAPH Web driver to generate an image map, the graphics procedure that produces the graph must use the IMAGEMAP= option and also the HTML= or HTML_LEGEND= option. IMAGEMAP= causes the procedure to generate an *Imagemap data set* that contains the following information:

- □ the shape of each area on the graph
- □ the coordinates of each area on the graph
- □ the link information, which is derived from the values of the HTML variable that is specified on HTML= or HTML_LEGEND=.

After the procedure generates the Imagemap data set, you specify that data set as an argument on a call to the IMAGEMAP macro (delivered with SAS/GRAPH). The macro writes the HTML tags needed to define an image map for the graph. You then create the HTML files needed to use that image map and implement the drill-down graph.

An image map must be defined and used in the same HTML file that references the drill-down graph. To create the HTML files, you can use a text editor or an HTML editor. Or, you can use DATA step processing in your SAS program. You can even use a SAS/GRAPH Web driver to generate the initial files, and then edit those files as needed to customize the page. For example, you could use the WEBFRAME driver to generate HTML and GIF files, and then edit the resulting HTML files as needed to customize the page layout and use an image map. This saves you the trouble of writing all your HTML files from scratch.

The technique to use for creating the HTML files is entirely up to you and depends on your needs. For example, if you are generating your graphs in batch runs and do not want to update the HTML files manually every time the image map changes, you may want to generate the HTML files in your SAS program, as shown in "Example 5: Customizing a Web Page with a Drill-Down Graph" on page 119.

About the IMAGEMAP Macro

The IMAGEMAP macro is delivered with SAS/GRAPH. It reads an Imagemap data set that is generated by a graphics procedure and writes the HTML tags needed to create an image map for the graph produced by that procedure. You only use the IMAGEMAP macro when you create a customized Web page. In that case, you use the GIF driver (without ODS) to generate the GIF files. Because the GIF driver does not generate HTML files, you have to create the HTML files you need for your Web page, and you use the IMAGEMAP macro to write the image map to the appropriate HTML file. The IMAGEMAP macro writes the image map for you, but you must write the <IMG> tag that references the image that will use the map, and you must specify the USEMAP attribute on that <IMG> tag to associate the image map with the GIF image.

Note: When you use ODS, or when you use the HTML or WEBFRAME device drivers, SAS/GRAPH creates the HTML files needed for your output, and it implements the drill-down graph for you. In those cases, you do not need to use the IMAGEMAP macro. △

The macro has the following syntax:
%IMAGEMAP(*imagemap-data-set, outfile*)

imagemap-data-set
The name of the data set that contains the image-map information. This is the data set specified on the IMAGEMAP= option on the graphics procedure that produces the graph (see "About the Imagemap Data Set" on page 95).

outfile
The file in which to write the HTML tags that define the image map. This can be any file. However, the image map must eventually be defined in the HTML file that references the drill-down graph's GIF file. Typically, it is easiest to specify the HTML file as the outfile, although you can write the image map to a different file, and later paste it into the appropriate HTML file.

The IMAGEMAP macro is delivered as one of the Annotate macros that come with SAS/GRAPH. To use an Annotate macro, you must provide your program with access to the data set that contains the macros, and you must compile the macros. Some sites automatically define a fileref for the data set that contains the Annotate macros.

If the fileref is set automatically at your site, you can compile the Annotate macros and make them available by simply submitting the ANNOMAC macro:

```
%annomac;
```

If the fileref is not set automatically, find out from your SAS Support Consultant where the Annotate macros are stored, and allocate a fileref that points to the data set:

```
filename fileref 'external-file';
```

Then include the Annotate macros in your session:

```
%include fileref (annomac);
```

To call the IMAGEMAP macro, you must specify both the name of the Imagemap data set to be used as input and the name of the output file where you want the image map defined. The following code assumes that a graphics procedure has already specified IMAGEMAP=MAPDATA to create the Imagemap data set named MAPDATA:

```
/* compile the annomac macros */
%annomac;
/* allocate a file for custom html file */
filename vbar  'external-file';
/* generate image map for drill-down */
%imagemap(mapdata, vbar);
```

Here, the macro reads the information in the data set MAPDATA and uses it to write the image map to the file that is referenced by the fileref VBAR. If the file does not exist, it is created. If the file does exist, the HTML tags are appended to the end of the file. The resulting HTML tags resemble the following:

```
<MAP NAME="BARCHART">
<AREA SHAPE=RECT COORDS="424, 143, 470, 256"
      href="reports.html#central">
<AREA SHAPE=RECT COORDS="366, 175, 412, 256"
      href="reports.html#south">
<AREA SHAPE=RECT COORDS="308, 106, 354, 256"
      href="reports.html#west">
</MAP>
<P>
<IMG SRC="barchart.gif" USEMAP="#BARCHART">
```

The image map is assigned the same name as the GRSEG that is produced by the procedure. For example, if the GRSEG is named BARCHART, then the image-map name will be BARCHART.

The image-map name is written in all uppercase letters. Because the Web browsers in some operating environments use case-sensitive image-map names, the USEMAP attribute on the <IMG> tag that uses the map should reference the map's anchor name in all uppercase letters, as shown in the HTML tags above.

Creating an HTML File in a SAS Program

For a customized Web page, you must create all the HTML files you need to display your output. You can create the files any way you like. This section shows you how to create an HTML file in a SAS program, using PUT statements in a DATA step.

To implement a drill-down graph, you need an HTML file that references the graph. That same HTML file must also contain the image map that defines the graph's hot zones and the location of the graph's link targets. When you call the IMAGEMAP macro to create the image map, you can direct its output to the HTML file that references the graph. To use the IMAGEMAP macro, you must first compile the Annotate macros that are delivered with SAS/GRAPH (see "About the IMAGEMAP Macro" on page 102).

The following code creates a file that is referenced by the fileref VBAR. The code assumes the following:

□ A fileref already exists to point to the location of the Annotate macros.

□ The Imagemap data set MAPDATA has already been created.

□ An HTML file with two frames will be created to display the output. The left frame will display the drill-down graph, and the right frame will display the target output for the link selected in the graph. The frame that displays the target output will be assigned the name view_pies.

```
/* specification for the html file */
filename vbar  'external-file';
/* generate html file to display */
/* drill-down graph */
data _null_;
  file vbar;
  put '<HTML>';
  put '<BODY>';
  put '<BASE TARGET=view_pies>';
  put '<IMG SRC="barchart.gif" '@;
  put ' USEMAP="#BARCHART">';
  put '</BODY>';
  put '</HTML>';
/* writes image map to file */
%imagemap(mapdata, vbar);
```

Here, the _NULL_ keyword is used in the DATA step to suppress the creation of a SAS data set. Instead, output is directed to file VBAR.

The PUT statements write the HTML tags to create a body page and to reference the chart that is stored in the file barchart.gif. The USEMAP attribute on the <IMG> tag indicates that an image map named BARCHART defines hot zones and links for the chart. The image map does not yet exist, but it will be appended to the file by the IMAGEMAP macro, which is called on the last line in the code that is shown above. The <BASE> tag indicates that the target output for the links will be displayed in a frame named view_pies.

After all the HTML tags have been completed for file VBAR, the code calls the IMAGEMAP macro, which specifies file VBAR as the output file. Thus, the macro appends to the end of file VBAR the HTML tags that are needed to define the image map. The image map is assigned the name BARCHART because the GCHART procedure that created file barchart.gif used NAME=BARCHART to name the resulting graph.

To place the image map in the middle of the file (for example, above the <IMG> tag that uses it), you would have to start a new DATA step after the macro call to continue writing HTML tags to the file. The new DATA step would have to open the HTML file in “modify” mode, using an operating-environment specific option on the FILE statement. For example, some environments use the option MOD. For more information on DATA step programming, see *SAS Language Reference*.

For a complete example that shows how to create all the HTML files needed to implement a custom Web page, see “Example 5: Customizing a Web Page with a Drill-Down Graph” on page 119.

Creating Java Applets and ActiveX Controls

When used with ODS, procedures GCHART, GCONTOUR, GMAP, GPLOT, and G3D enable you to specify DEVICE=JAVA on a GOPTIONS statement to generate graphs that are Java applets. These same procedures enable you to use DEVICE=ACTIVEX to generate graphs that are ActiveX controls. The DEVICE=JAVA or DEVICE=ACTIVEX settings are used instead of specifying DEVICE=GIF; otherwise, the ODS capabilities are the same as those discussed in “Using the Output Delivery System (ODS) with SAS/GRAPH Software” on page 81.

The JAVA and ACTIVEX device drivers can be used with the GCHART, GCONTOUR, GMAP, GPLOT, and G3D procedures. They cannot be used with any other SAS/GRAPH procedures.

Both the JAVA and ACTIVEX drivers create an HTML file that defines the graphs that are generated in the SAS program. The JAVA driver references each graph within an <APPLET> tag; the graphs can only be displayed in web browsers that support Java applets. The ACTIVEX driver references each graph within an <OBJECT> tag; the graphs can only be displayed in Web browsers that run in a Windows operating environment and that support ActiveX controls.

Graphs that are saved as GIF files are static graphs when viewed in a Web browser. However, Java or ActiveX graphs can be modified interactively during a browser session. For example, you can change the graph type from a bar chart to a pie chart, and then change the colors of the pie slices.

To take advantage of the features available in Java applets and ActiveX controls, SAS/GRAPH produces graphs that suit these device types. The resulting Java or ActiveX graphs may differ visually from their corresponding GRSEGs, but they show the same data relationships as the GRSEG shows: the visual differences are aesthetic differences.

Because SAS/GRAPH produces graphs that take advantage of the Java and ActiveX technologies, not every SAS/GRAPH option that you set in your SAS program is applied to the Java or ActiveX graph. Again, the graph essentially shows the same data relationships as the corresponding GRSEG, but the graph's visual characteristics may be different.

For more information about the Java applets and ActiveX controls, consult the Version 8 online documentation that is listed on the Publication pages of the SAS external Web site (http://www.sas.com), or contact SAS Institute Technical Support.

Animating GIF Files (DEV=GIFANIM)

The GIFANIM driver provides a mechanism for combining GIF images created with SAS/GRAPH procedures that allow you to create GIF animations for reports you publish on the Web. The behavior of the driver is controlled by graphics options that enable you to set such things as delay time, iteration count, transparency, and disposal methods.

Before using the GIFANIM device driver, you should become familiar with the animation process, the controls available, and certain limitations. The driver counts on you to control the process so that the animated sequence will be constructed properly.

For an example, see "Example 6: Creating a GIF Animation File" on page 123.

GIF Animation Process

The process involved with creating an animated GIF with the GIFANIM driver requires that you, the animator, take control of the job sequence and ensure that the resulting data stream is constructed properly. The GIFANIM data stream has three parts: Header, Body, and Trailer. Each portion of the data stream is equally important and must be present. Otherwise, an incomplete or unreadable animation sequence will result.

Preparing the Header

When creating a new animated GIF data stream, you must issue GOPTIONS GSFMODE=REPLACE prior to the invocation of the first SAS/GRAPH procedure in the

SAS job. The driver will then construct a new data stream by writing a valid GIF header and graphic data from the first procedure.

Preparing the Body

After the first SAS/GRAPH procedure has been executed, you must construct the body of the GIF animation. You can think of the Body as all the graphic images between the first and last image. Set GOPTIONS GSFMODE=APPEND to signal the GIFANIM driver to suppress the header information and to begin appending graphic data to the current data stream. The GOPTIONS GSFMODE=APPEND statement must appear somewhere between the first and second SAS/GRAPH procedures.

Note: If you use BY-group processing on the first graphics procedure to generate multiple graphs, they are automatically appended to the same GIF file. Thus, you do not need GSFMODE=APPEND for that first procedure. If you do not use a second graphics procedure to append additional graphs to the GIF file, you do not need this Body section in your program. △

Preparing the Trailer

The final step is to mark the end of the animation by appending a GIF trailer ('3B'x) to the data stream. The way to do this depends on whether the last procedure uses BY-group processing.

□ Without BY-group processing, set GOPTIONS GEPILOG='3B'X before the last SAS/GRAPH procedure.

□ With BY-group processing, do not assign a value to GEPILOG; otherwise your GIF animation sequence will be incomplete. Because a GEPILOG is written after each graph in a BY-group, the GIF decoder will interpret the first **'3B'x** as the end of the animation. Instead, you should use a DATA step to add the trailer to the data stream:

```
data _null_;
   file out recfm=n mod;
   put '3B'x;
run;
```

After the animation is complete, issue a GOPTIONS RESET=ALL statement to prepare for succeeding SAS jobs.

GIFANIM Device Driver Controls

You can control the GIFANIM device driver with these GOPTIONS settings:

ITERATION=*iteration-count*
specifies the number of times to repeat the animation loop, or whether to loop infinitely. An iteration of 0 specifies an infinite loop.

GSFMODE=REPLACE | APPEND
specifies whether the graphics output should replace the contents of an existing file or be appended to it. In addition, the GIFANIM driver uses the value of GSFMODE to determine when to write the GIF header.

DELAY=*delay-time*
controls the amount of time between graphs in the animation sequence. A delay time of 1 specifies a delay of .01 seconds.

DISPOSAL = NONE | BACKGROUND | PREVIOUS | UNSPECIFIED
specifies what happens to the graphic after it is displayed.

USERINPUT | NOUSERINPUT
: enables or disables user input during image animation if supported by the browser displaying the animation.

TRANSPARENCY | NOTRANSPARENCY
: specifies whether the background of the image should appear to be transparent when the image is displayed in the browser.

For a complete description of these options, see Chapter 9, “Graphics Options and Device Parameters Dictionary,” on page 301.

Note: Beginning with Version 7, some of the options that control the GIFANIM driver’s behavior are different from earlier releases. △

Changing the Size of the Output

If the default size of the GIFANIM device is not suitable for your needs, you can modify GOPTIONS HSIZE= and VSIZE= accordingly. The following SAS macro has been provided so that you can set HSIZE= and VSIZE= indirectly in pixel units:

```
%macro IMGSIZE(w=1280, h=1024, dpi=100,
               rows=43, cols=83);

   %if &dpi <=0 %then
      %put DPI must be greater than zero.;
   %else %do;
      goptions hsize=%sysevalf(&w/&dpi)in
               vsize=%sysevalf(&h/&dpi)in
               hpos=&cols vpos=&rows
   %end;

%mend IMGSIZE;
```

For example, to create an image that is 373 pixels wide and 280 pixels high, submit this macro call:

```
%imgsize(w=373, h=280);
```

Here are some key points to remember when using the %IMGSIZE macro:

□ You may set HSIZE and VSIZE up to 12.8 in. and 10.24 in., respectively. The device entry will need to be redefined with PROC GDEVICE if a larger size is required.

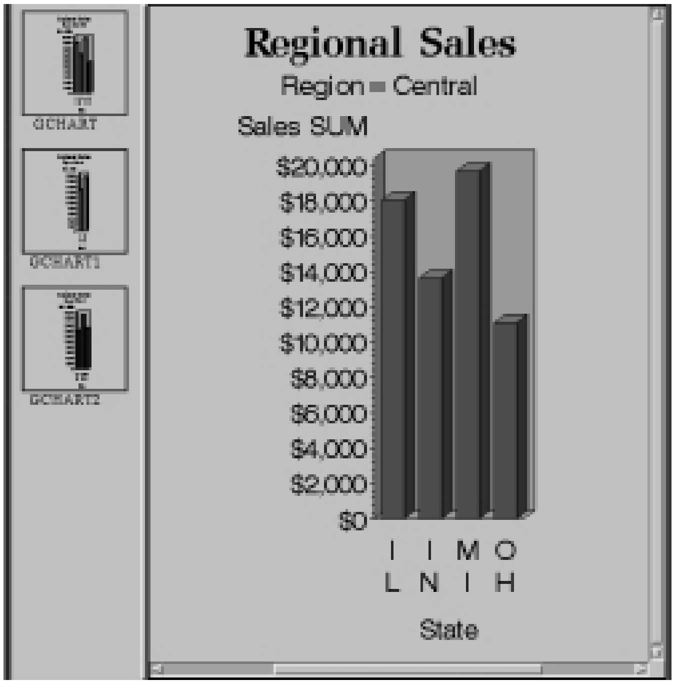
□ The dots per inch (dpi) should match that of the device driver being used. The resolution of the GIFANIM device driver is 100 dpi. Because this is the default value for the %IMGSIZE macro, you should not have to specify the dpi value unless you use PROC GDEVICE to modify GIFANIM. The ROWS and COLS parameters were added to the %IMGSIZE macro to facilitate setting GOPTIONS HPOS= and VPOS=, if you wish to do so.

Example 1: Using the HTML and WEBFRAME Device Drivers

This example shows how to use the SAS/GRAPH device drivers HTML and WEBFRAME. The example specifies DEVICE=WEBFRAME, but you would simply change that to DEVICE=HTML to use the HTML device driver.

Because the example uses BY-group processing, it generates a separate graph for each value of the BY variable, in this case REGION. Display 5.9 on page 108 shows how the output looks when file index.html is viewed in a Web browser.

Display 5.9 Graph Generated with DEVICE=WEBFRAME

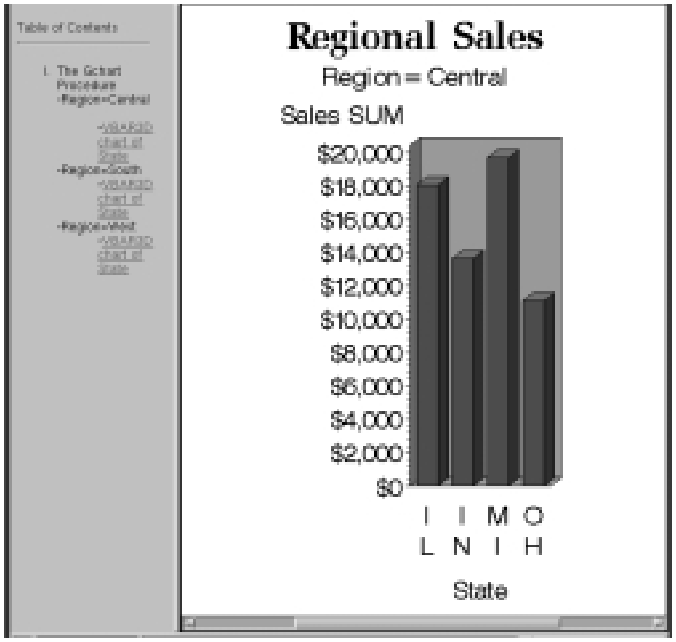


```
/* This example uses the WEBFRAME driver */
/* You can modify it to use the HTML driver */

/* This is the only line you have to change */
/* to run the program. Specify */
/* a location in your file system. */
filename webdoc 'path-to-Web-server';

/* Set the general graphics environment */
goptions reset=global gunit=pct
         htitle=6 htext=4
         ftitle=zapfb ftext=swiss;

/* Specify the webframe device driver */
/* To use the HTML driver, use device=html */
goptions device=webframe gsfname=webdoc
         transparency noborder;

/* Create data set REGSALES */
data regsales;
   length Region State $ 8;
   format Sales dollar8.;
   input Region State Sales;
   datalines;
West CA 13636
West OR 18988
West WA 14523
Central IL 18038
Central IN 13611
Central OH 11084
Central MI 19660
South FL 14541
South GA 19022
;

/* Sort data by the BY variable */
proc sort data=regsales;
by region;
run;

/* Run the GCHART procedure */
title1 'Regional Sales';
proc gchart data=regsales;
   vbar3d state / sumvar=sales
   patternid=by;
   by region;
run;
quit;
```

Example 2: Using ODS with SAS/GRAPH Software

This example uses ODS to generate Web output. It produces a bar chart that shows sales figures for three sales regions. The GCHART procedure uses BY-group processing to generate a separate chart for each region.

The ODS HTML statement creates a body file and contents file, and it uses a frame file to display the output. The NEWFILE= option is used to direct each output graph to its own GIF file. To see each graph, you select the reference to it in the Table of Contents, which is displayed in the left frame when you view file saleFram.html in a browser.



In this example, the data and the output are kept simple so that you can focus on the code used in ODS processing. For more detailed examples, see the following:

"Example 8. Creating a Simple Web Page with the ODS HTML Statement" on page 284

"Example 9. Combining Graphs and Reports in a Web Page" on page 287

"Example 10. Creating a Bar Chart with Drill-down for the Web" on page 294

```
/* This program uses ODS to create html */
/* and gif output. This is the only line you */
/* have to change to run the program. Specify */
/* a location in your file system.    */
filename odsout 'path-to-Web-server';

/* close the listing destination */
ods listing close;

/* set the graphics environment */
goptions reset=global gunit=pct
         htitle=6 htext=4 ctext=black
         ftitle=zapfb ftext=swiss;

/* create data set REGSALES */
data regsales;
    length Region State $ 8;
    format Sales dollar8.;
    input Region State Sales;
    datalines;
West CA 13636
West OR 18988
West WA 14523
Central IL 18038
Central IN 13611
Central OH 11084
Central MI 19660
South FL 14541
South GA 19022
;

/* Sort data by the BY variable */
proc sort data=regsales;
by region;
run;

/* assign graphics options for ODS output */
goptions device=gif transparency noborder;

/* open html destination for ODS output */
ods html body='sales.html'
         contents='saleCont.html'
         frame='saleFram.html'
         path=odsout
         newfile=output;

/* Run the GCHART procedure */
title1 'Regional Sales';
proc gchart data=regsales;
   vbar3d state / sumvar=sales
   patternid=by;
by region;
run;
quit;

/* close the html destination */
ods html close;

/* open the listing destination */
ods listing;
```

Example 3: Using a Web Driver to Generate a Drill-down Graph

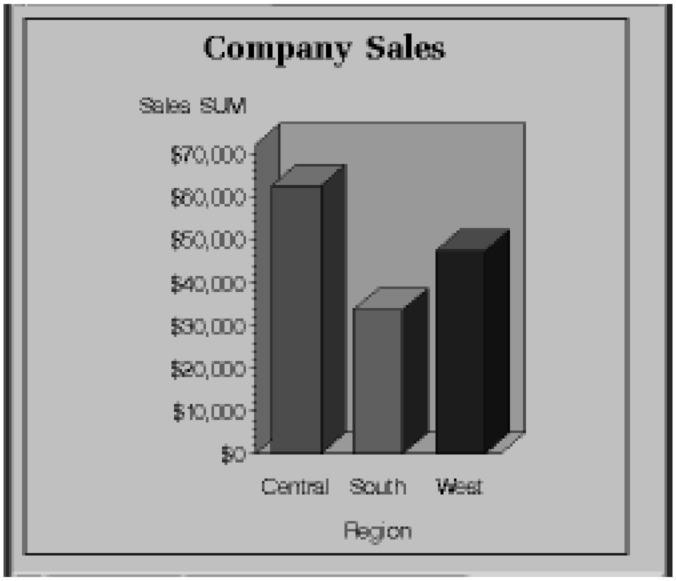
This example shows how to use a SAS/GRAPH Web driver to generate a drill-down graph (see "About Drill-down Graphs" on page 90). The example uses the HTML driver, but the principles would be the same for using the WEBFRAME driver.

Note: The SAS/GRAPH Web drivers do not assign anchor names to the output that they generate. Thus, the Web drivers are best used to generate drill-down graphs that link to GIF files that are stored or referenced in separate files. To link to HTML files that store multiple pieces of output, or to link to both graphics and non-graphics output, it is easier to use ODS to generate drill-down graphs. △

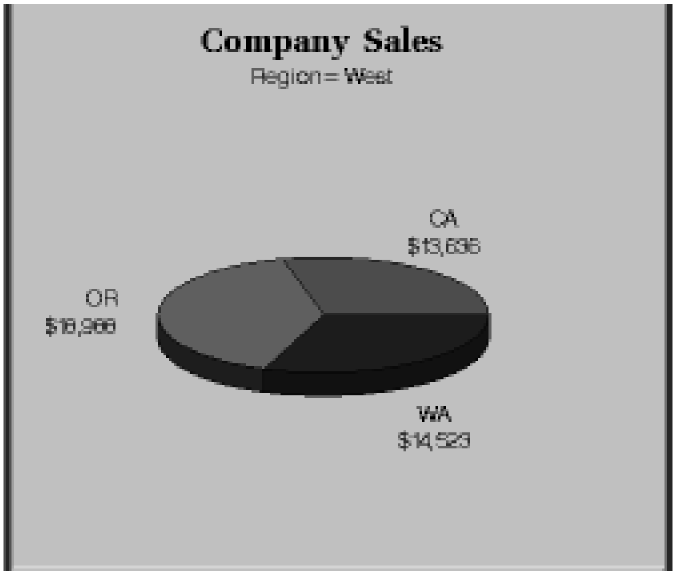
To generate a drill-down graph with the HTML or WEBFRAME device driver, all you have to do is specify the IMAGEMAP= option on the graphics procedure that generates the graph and supply the linking information that identifies the location of the target output. SAS/GRAPH produces the GIF files and the image map needed for the drill-down graph and creates the HTML files needed to display the output and implement the drill-down capability.

Planning the Web Page

To plan the Web page, you must determine what output you need, which Web driver you will use, and what links you need to implement the drill-down capability. This example generates a simple drill-down bar chart showing the regional sales figures for three sales regions.



When the bar chart is displayed in a browser, you can select any one of the bars to drill down to a pie chart that shows the sales figures for the corresponding region. For example, you can click on the bar that represents sales for the Western region to drill down to a pie chart that shows the Western region's sales figures.



To create this drill-down graph, you need the following:

output
: The output requires a 3-D vertical bar chart to show the drill-down chart. The output also requires three pie charts to show the state sales figures for each region.

drivers
: The drill-down chart is a single bar chart, so the HTML driver is preferable to the WEBFRAME driver, which is designed for linking to multiple graphs. For the pie charts, the GIF driver can generate the required GIF files.

For naming the GIF files, you can let SAS/GRAPH assign default names, or you can use the NAME= option to control the names. This example will use the NAME= option to ensure that the GIF files are named salereg1.gif, salereg2.gif, and salereg3.gif.

links
: Each bar in the bar chart requires a link to the corresponding region's pie chart. Because the example will direct each pie chart to its own GIF file, the links do not need anchor names.

The drill-down capability for this design requires only one drill-down level: from the bar chart to the pie charts. There is no legend and, therefore, there are no links from the legend. Thus, only one HTML variable is required to store link information for the chart. This example will create a variable named LINKS.

Each pie chart will be directed to its own GIF file. Thus, the links do not need anchor names. The example assumes that users will access all output through a file system, so a file specification will suffice for the linking information.

Output Needed for the Drill-down Graph

To generate this example drill-down bar chart, you must run the GCHART procedure with the VBAR3D statement. The PROC GCHART statement must use the IMAGEMAP= option to create an Imagemap data set, and the VBAR3D statement must use the HTML= option to specify the HTML variable that contains the linking information—in this case, the variable LINKS. To generate the pie charts, you can use PROC GCHART to generate all three pie charts in a single procedure run by using BY-group processing (see "BY Statement" on page 177) .

In this example code, note the following:

- The graphics options specify the HTML device driver and set the display area to 450x400 pixels. The HTML driver always creates a file named index.html to reference the graphics output. File index.html is the file you must view in the browser.
- The DATA step creates the variable LINKS and assigns to it the values needed for the links in the drill-down graph.
- The PROC GCHART statement uses the IMAGEMAP= option to create an Imagemap data set named SALEMAP.
- The VBAR3D statement uses the HTML= option to specify the variable LINKS, thus associating the linking information with the bar chart.
- The presence of both IMAGEMAP= and HTML= causes the HTML device driver to write the image map information to the file index.html, which contains the <IMG> tag that references the bar chart. In the image map, the linking information is derived from the values of the variable LINKS.
- Because the HTML driver writes over the contents of file index.html each time a graphics procedure is run, the example switches to the GIF driver to generate new files for the pie charts.

□ For the pie charts, the example specifies a second GCHART procedure, which does not use the IMAGEMAP= option. This second GCHART uses the PIE3D statement and uses BY-group processing to generate a separate pie chart for each region in the data. On the PIE3D statement, the NAME= option specifies the name SALEREG1 for the first pie chart. SAS/GRAPH automatically increments that name for the next two pie charts, which are assigned the names SALEREG2 and SALEREG3.

Code for the Example

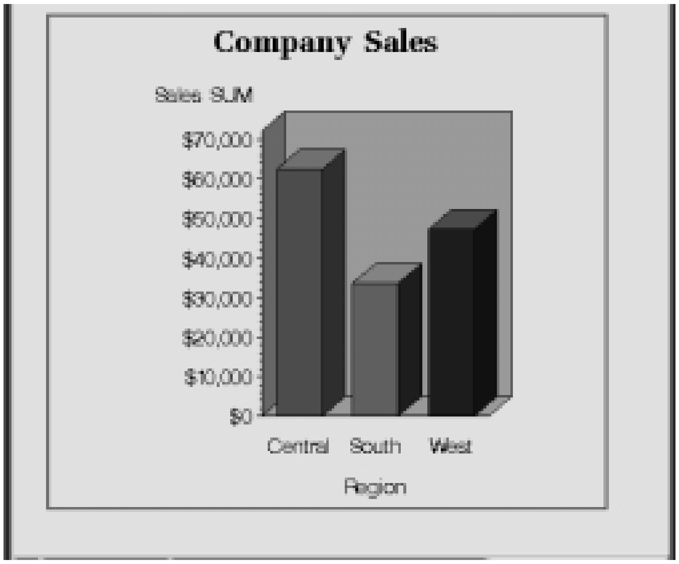
```
/* This is the only line you have to change to run the */
/* program. Specify a location in your file system. */
filename webout ’path-to-Web-server’;

/* set general graphic options */
goptions reset=global gunit=pct
         htitle=6 htext=4
         ftitle=zapfb ftext=swiss border;

/* assign graphics options for ODS output */
goptions transparency noborder
  xpixels=450 ypixels=400
  gsfname=webout device=html;

/* create data set REGSALES */
data regsales;
    length Region State $ 8;
    format Sales dollar8.;
    input Region State Sales;
    length links $40; /* the HTML variable */

/* add the HTML variable and assign its values */
if Region=’Central’ then links=’href="salereg1.gif"’;
  else if Region=’South’ then links=’href="salereg2.gif"’;
  else if Region=’West’ then links=’href="salereg3.gif"’;

datalines;
 West    CA  13636
 West    OR  18988
 West    WA  14523
 Central IL  18038
 Central IN  13611
 Central OH  11084
 Central MI  19660
 South   FL  14541
 South   GA  19022
 ;

title1 ’Company Sales’;
proc gchart data=regsales imagemap=salemap;
 vbar3d region / sumvar=sales
  patternid=midpoint
  html=links name='htmldril';
run;

/* change to GIF driver for pie charts */
goptions dev=gif;

proc sort data=regsales out=regsales;
by region;

/* Create three charts that use the HTML variable */
proc gchart data=regsales;
  pie3d state / sumvar=sales
   noheading name='salereg1';
by region;
run;
quit;
```

Example 4: Using ODS to Generate a Drill-down Graph

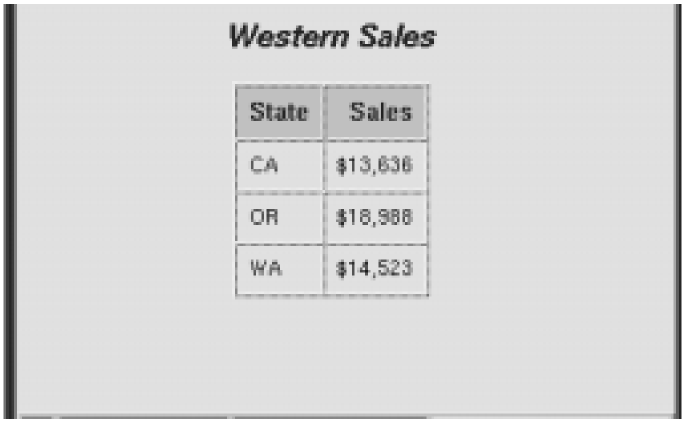
You can use ODS to generate a drill-down graph (see "About Drill-down Graphs" on page 90). All you have to do is supply the linking information that identifies the location of the target output. SAS/GRAPH produces the GIF files and the image map needed for the drill-down graph, and ODS creates the HTML files to display the output and implement the drill-down capability.

Planning the Web Page

To plan a Web page when using ODS to generate a drill-down graph, you must determine what output you need, what HTML files you will name to store that output, and what links you need to implement the drill-down capability. This example generates a simple drill-down bar chart that shows the regional sales figures for three sales regions.



When the bar chart is displayed in a browser, you can select any one of the bars to drill down to a report that shows the sales figures for the corresponding region. For example, you can click on the bar that represents sales for the Western region to drill down to a report that shows the Western region’s sales figures.



Western Sales

State	Sales
CA	$13,636
OR	$18,988
WA	$14,523

To create this drill-down graph, you need the following:

output
The output requires a 3-D vertical bar chart to show the drill-down chart. The output also requires three reports to show the state sales figures for each region.

HTML files
You can write all of the output to a single body file, or distribute the output across multiple body files. This example will direct each piece of output to its own body file.

You can name the HTML files with any legal file name in your operating environment. This example will use the following names: sales.html for the file that references the drill-down chart, and central.html, south.html, and west.html for the reports on regional sales figures.

links
Each bar in the bar chart requires a link to the corresponding region’s report. Because the example will direct each report to its own HTML file, the links do not need anchor names. The example assumes that users will access all output through a file system, so a file specification will suffice for the linking information.

The drill-down capability for this design requires only one drill-down level: from the bar chart to the reports. There is no legend and, therefore, there are no links from the legend. Thus, only one HTML variable is required to store link information for the chart. This example will create a variable named RPT.

To provide the drill-down capability, the example creates an HTML variable named RPT. For each sales region, RPT stores the HREF value that links to the target output. The example then specifies RPT as the HTML variable for the GCHART procedure that generates the drill-down graph. To generate the target reports, the example uses PROC PRINT. The example assumes that users will access all output through a file system.

In this example, the data, the bar chart, and the reports are simple so that you can focus on the code needed to generate a drill-down graph. For more realistic drill-down examples, see Example 7 on page 596 and Example 5 on page 779.

Producing the Output for the Drill-down Graph

To generate the drill-down bar chart for this example, you must run the GCHART procedure with the VBAR3D statement. The VBAR3D statement must use the HTML= option to specify the HTML variable that contains the linking information – in this case the variable RPT. To generate the reports, you can run the PRINT procedure. To direct each report to its own body file, a separate ODS HTML statement must precede each PRINT procedure.

In this example code, note the following:

□ The DATA step creates the variable RPT and assigns to it the values needed for the links in the drill-down graph.

□ The first ODS HTML statement uses BODY= to direct the ODS output to file sales.html. Therefore, the GCHART procedure output will be referenced in that file.

□ The GCHART procedure uses HTML=RPT to provide SAS/GRAPH with the linking information for the drill-down graph.

□ After running the GCHART procedure, the example specifies another ODS HTML statement to direct output to file central.html. Therefore, output from the PRINT procedure that follows is written to central.html.

For more information about ODS and how it constructs HTML links and references, see “How ODS Constructs Links and References” on page 210.

Code for the Example

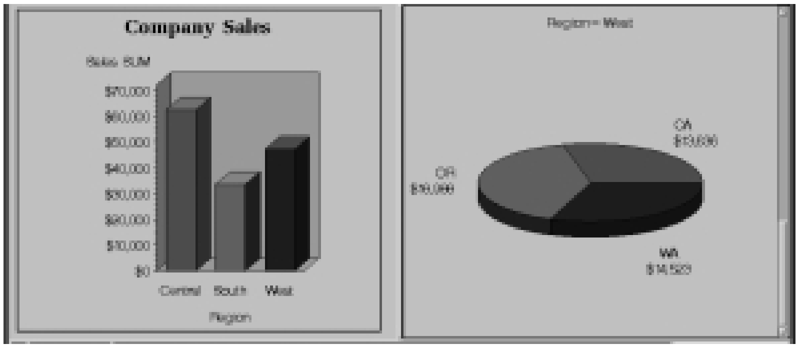
```
/* This is the only line you have to  */
/* change to run the program. Specify */
/* a location in your file system.    */
filename odsout 'path-to-Web-server';

/* close the listing destination */
ods listing close;

/* set general graphic options */
goptions reset=global gunit=pct
   htitle=6 htext=4
   ftitle=zapfb ftext=swiss;

/* create data set REGSALES */
data regsales;
   length Region State $ 8;
   format Sales dollar8.;
   input Region State Sales;
/* the HTML variable */
   length rpt $40;

/* assign values to HTML variable */
if Region='Central' then
     rpt='href="central.html"';
   else if Region='South' then
     rpt='href="south.html"';
   else if Region=’West’ then
     rpt=’href="west.html"’;

   datalines;
West CA 13636
West OR 18988
West WA 14523
Central IL 18038
Central IN 13611
Central OH 11084
Central MI 19660
South FL 14541
South GA 19022
;

/* assign graphics options for ODS output */
goptions device=gif transparency noborder
         xpixels=450 ypixels=400;

/* open the html destination for ODS output */
ods html body=’sales.html’ path=odsout;

/* Create a chart that uses HTML variable */
title1 ’Company Sales’;
proc gchart data=regsales;
   vbar3d region / sumvar=sales
   patternid=midpoint
   html=rpt;
run;
quit;

/* open a body file for report */
/* on central sales            */
ods html body=’central.html’ path=odsout;
title1 ’Central Sales’;
proc print data=regsales noobs;
   var state sales;
   where region=’Central’;
run;

/* open a body file for report */
/* on southern sales           */
title1 ’Southern Sales’;
ods html body=’south.html’ path=odsout;
proc print data=regsales noobs;
   var state sales;
   where region=’South’;
run;

/* open a body file for report */
/* on western sales            */
title1 ’Western Sales’;
ods html body=’west.html’ path=odsout;
proc print data=regsales noobs;
```

```
      var state sales;
      where region='West';
   run;
   quit;

   /* close the html destination */
   ods html close;

   /* open the listing destination */
   ods listing;
```

Example 5: Customizing a Web Page with a Drill-Down Graph

To plan a customized Web page for a drill-down graph, you must determine what the Web page will look like, what output you need, what HTML files you need to display the output in the layout you want, and what links you need to implement the drill-down capability. This example creates the Web page shown in Display 5.10 on page 119. On this Web page, the left frame displays a drill-down bar chart that shows the regional sales for three sales regions. When you select one of the bars in the bar chart, the right frame displays a pie chart that shows total sales for the states in the corresponding region.

Display 5.10 Customized Web Page with Drill-down Graph



To create this Web page, you need the following:

output The output requires a 3-D vertical bar chart to show the drill-down chart. The output also requires three 3-D pie charts to show the state sales figures for each region. To produce the graphs, you can use the GIF device driver to write each chart to its own GIF file.

HTML files Requires one HTML file to reference the drill-down chart, and a second to define the two HTML frames needed for the Web-page layout.

For the pie charts, you don't need an HTML file to display them, you can simply display each GIF file as needed. However, you can also create one HTML file to reference all three pie charts. This example will create an HTML file to reference all three pie charts, which will demonstrate how to use anchor names in a customized Web page.

You can name the HTML files with any legal file name in your operating environment. This example will use the following names: vbar.html for the bar chart, pies.html for the pie charts, and sales.html for the file that defines the frames. To view the custom Web page, you must open sales.html in a browser.

links
: Each bar in the bar chart requires a link to the corresponding region's pie chart. Because the example will reference all three pie charts in a single HTML file, the links need anchor names to locate each pie chart within the file. You can choose any anchor names you want. Because the pie charts represent the three sales regions, this example will use the anchor names central, south, and west.

The drill-down capability for this design requires only one drill-down level: from the bar chart to the pie charts. There is no legend and, therefore, there are no links from the legend. Thus, only one HTML variable is required to store link information for the chart. This example will create a variable named LINKS.

Output Needed for the Drill-down Graph

To generate the drill-down bar chart for this example, you need to run the GCHART procedure with the VBAR3D statement. The PROC GCHART statement must use the IMAGEMAP= option to specify a name for the Imagemap data set. The VBAR3D statement must use the HTML= option to specify the HTML variable that contains the linking information—in this case, the variable LINKS.

To generate the pie charts, you need to run GCHART with the PIE3D statement. You can use BY-group processing "BY Statement" on page 177 to generate all three pie charts on a single procedure run.

In this example code, note the following:

- □ The graphics options specify the GIF device driver and set the display area to 450 x 400 pixels.
- □ A FILENAME statement allocates a storage location for all the GIF files. The HTML files that are required for the output will be created with DATA step processing, so additional FILENAME statements allocate files for the HTML files.
- □ On the GCHART procedure:
  - □ The PROC GCHART statement uses IMAGEMAP= to create an Imagemap data set named MAPDATA.
  - □ The VBAR3D statement assigns the name BARCHART to the bar chart's GRSEG. Thus, the resulting GIF file is named barchart.gif.
  - □ The PIE3D statement assigns the name SALEREG1 to the first pie chart's GRSEG. Thus, the GIF files for the pie charts are named salereg1.gif, salereg2.gif, and salereg3.gif. For more information on the naming conventions for the output GIF files, see "Naming Conventions Used for GIF Files" on page 75.
- □ Several DATA steps create the required HTML files.
- □ The IMAGEMAP macro uses the data set MAPDATA to create an image map, which it writes to the HTML file that references the drill-down bar chart. Because the GRSEG for the chart was assigned the name BARCHART, the image map is also named BARCHART.

Code for the Example

This example uses the following names for the output HTML files:

- vbar.html to reference the bar chart.
- pies.html to reference the three pie charts, which are assigned the anchor names central, south, and west.
- sales.html to define the frames needed for the Web-page layout. The right frame is assigned the name view_pies.

The code assumes the following:

- A fileref already exists to point to the location of the Annotate macros (see “About the IMAGEMAP Macro” on page 102).
- Users will access the output through a file system.

```
/* These FILENAME statements are the only       */
/* lines you have to change to run the program. */
/* Specify locations in your file system.       */

/* aggregate location for all gif files */
filename webout ’path-to-Web-server’;

/* filename for this file must be vbar.html */
filename vbar  ’external-file’;

/* filename for this file must be pies.html */
filename pies  ’external-file’;

/* filename for this file must be sales.html */
filename frame ’external-file’;

/* compile the annomac macros */
%annomac;

/* set general graphic options */
goptions reset=all gunit=pct
         htitle=6 htext=4
         ftitle=zapfb ftext=swiss;

/* assign graphics options for Web output */
goptions device=gif transparency
         gsfname=webout gsfmode=replace
         xpixels=450 ypixels=400;

/* create data set REGSALES */
data regsales;
     length Region State $ 8;
     format Sales dollar8.;
     input Region State Sales;
/* the HTML variable */
     length links $40;

/* add HTML variable, assign its values */
if Region=’Central’ then
     links=’href="pies.html#central"’;
  else if Region=’South’ then
     links=’href="pies.html#south"’;
  else if Region='West' then
     links='href="pies.html#west"';

     datalines;
West    CA  13636
West    OR  18988
West    WA  14523
Central IL  18038
Central IN  13611
Central OH  11084
Central MI  19660
South   FL  14541
South   GA  19022
;

/* sort data by region for pie charts */
proc sort data=regsales out=regsales;
by region;

/* Create chart for drill-down graph */
title1 'Company Sales';
proc gchart data=regsales
    imagemap=mapdata;
  vbar3d region / sumvar=sales
    patternid=midpoint
    html=links
    name='barchart';
run;

/* create pie charts for regional sales */
title1;
  pie3d state / sumvar=sales
    noheading name='salereg1';
  by region;
run;
quit;

/* generate html file for drill-down graph */
data _null_;
  file vbar;
  put '<HTML>';
  put '<BODY>';
  put '<BASE TARGET=view_pies>';
  put '<IMG SRC="barchart.gif" '@;
  put ' USEMAP="#BARCHART">';
  put '</BODY>';
  put '</HTML>';
/* write image map to file */
%imagemap(mapdata, vbar);

/* generate html file to display pie charts */
data _null_;
  file pies;
  put '<HTML>';
  put '<BODY>';
  put '<A NAME="central"></A>'@;
  put '<IMG SRC="salereg1.gif">';
  put '<A NAME="south"></A>'@;
  put '<IMG SRC="salereg2.gif">';
  put '<A NAME="west"></A>'@;
  put '<IMG SRC="salereg3.gif">';
  put '</BODY>';
  put '</HTML>';

/* generate html file to display frames */
data _null_;
  file frame;
  put '<HTML>';
  put '<FRAMESET COLS="50%, *">';
  put '<FRAME SRC="vbar.html">';
  put '<FRAME SRC="pies.html"'@;
  put ' NAME="view_pies">';
  put '</FRAMESET>';
  put '</HTML>';
run;
```

Example 6: Creating a GIF Animation File

This example creates a GIF animation using regional sales data. BY-group processing is used to generate three bar charts, one chart each to represent sales figures for three sales regions. When the output GIF file is viewed in a browser, the charts are displayed in a timed sequence. To speed up the animation, decrease the setting for the DELAY= graphics option. To slow down the animation, increase the setting for the DELAY= graphics option.

```
/* assign the destination for the output */
filename out 'external-file.gif';

/* set the graphics environment */
goptions reset=global gunit=pct border
   cback=ligr ctext=black
   colors=(blue green red) ftext=swissb
   ftitle=swissb htitle=6 htext=5;

/* assign graphics options for the animation */
goptions display
         dev=gifanim
         gsfname=out
         gsfmode=replace
         iteration=0
         delay=200;

/* create data set REGSALES */
data regsales;
  length Region State $ 8;
  format Sales dollar8.;
  input Region State Sales;
  datalines;
West CA 13636
West OR 18988
West WA 14523
Central IL 18038
Central IN 13611
Central OH 11084
Central MI 19660
South FL 14541
South GA 19022
;

/* sort the data set */
proc sort data=regsales out=regsales;
by region;
run;

/* generate the charts */
title1 'Company Sales';
proc gchart data=regsales;
   vbar state / sumvar=sales
   patternid=by;
by region;
run;
quit;

/* end the animation */
data _null_;
   file out recfm=n mod;
   put '3B'x;
run;

goptions reset=all;
```

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

The Institute is a private company devoted to the support and further development of its software and related services.