

A Dynamic Imagemap Generator

Carol Martell, UNC Highway Safety Research Center, Chapel Hill, NC

ABSTRACT

The SAS[®] Output Delivery System enables the creation of an image map using SAS/GRAPH[®]. Perhaps no one could appreciate this more than the Web designer assigned the task of converting an image of North Carolina to an image map by manually outlining all 100 counties. Ouch! SAS came to the rescue. After fulfilling repeated requests for image maps of varying sizes, I wrote an application to automate the process through the Web using SAS/IntrNet[®]. Parameters for size, color, and link attributes are provided through a Web form. The image map is accessed in a browser window and the user can 'save as' and 'save image as' to get both HTML and GIF files for placement elsewhere. The application is generalized to create an image map of any state.

INTRODUCTION

An image map, once created, cannot be correctly resized using today's GUI-based HTML editors. One can resize the GIF image, but the map coordinates in the HTML file do not change accordingly. Consequently, the hotspots no longer line up with the map image. The application makes image map recreation a breeze. This paper steps through the process of moving from creating a simple map in an interactive SAS session to creating an image map through a Web browser. At each new step, additional code is shown in bold typeface. Since the intention is that the output generated be moved elsewhere, the map is purposefully devoid of legend, title, footnotes, etc. This paper assumes familiarity with running Application Dispatcher.

PREPARATION

The first step is to prepare the two components of input data for PROC GMAP: a map data set and the data to be mapped. SAS provides various map data sets, one of which is Counties. The state and county variable values in that table are represented by FIPS codes. SAS has several functions for manipulating FIPS state codes. The STFIPS function converts postal codes to FIPS codes. We create a new map data set by subsetting for North Carolina records:

```
DATA two;
SET maps.counties;
IF state=STFIPS("NC");
RUN;
```

Since Counties is unprojected, we run PROC GPROJECT to ensure the map will be undistorted.

```
PROC GPROJECT DATA=two OUT=one;
ID county;
RUN;
```

The SAS table containing data to be mapped must contain an ID variable to link with the map table ID variable. In this case ID is county. PROC GMAP also requires a variable whose value is to be reflected on the map. We assign the variable the name mapv. For simplicity in this example we will use mapv=1 for each county, so every county in the data set will map as equal. To be mapped, every county must have an observation in the table. The easiest way to create a table with an observation for every county code is to subset the map data set, because FIPS county code assignments are not intuitive.

```
PROC SQL;
CREATE TABLE dummy AS
SELECT DISTINCT county,
1 AS mapv
FROM one;
QUIT;
```

We reset the graphics options and then specify the GIF device driver, choosing a white background for the image. Pattern statements assign colors for map variable levels. Since there is only one value for mapv, we need only one pattern.

```
GOPTIONS RESET;
GOPTIONS DEVICE=GIF CBACK=white;
PATTERN1 COLOR=blue;
```

GENERATING A MAP INTERACTIVELY

We can produce the map in an interactive session as follows.

```
PROC GMAP MAP=one
DATA=dummy;
ID county;
CHORO mapv/ COUTLINE=black NOLEGEND
NAME="mymap";
RUN;
QUIT;
```

Figure 1 shows the resulting map.

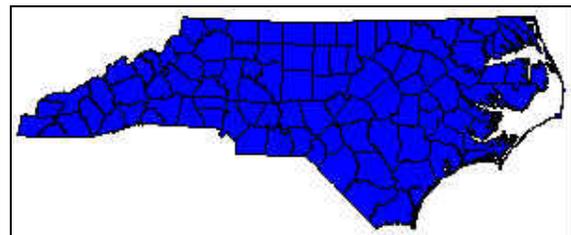


Figure 1

GENERATING A MAP INTERACTIVELY USING ODS

The following additions use the Output Delivery System to create an HTML file treating the map as a simple GIF, not an image map. The ODS path parameter gives the hard path for writing both the HTML and the GIF file. If you use a defined FILEREF in the PATH= option, ODS will write both the HTML and GIF files in the directory specified in the FILEREF, and the reference to the GIF in the html will be relative (i.e., not be prefaced with any pathing.) This will make it possible to place the two files wherever we want later without having to manually change references in the HTML file.

```
FILENAME o 'directory/path/for/files';
```

```
ODS HTML
```

```
  PATH=o
```

```
  BODY='pagename.html';
```

```
PROC GMAP MAP=one
```

```
  DATA=dummy;
```

```
  ID county;
```

```
  CHORO mapv/ COUTLINE=black NOLEGEND
```

```
  NAME="mymap";
```

```
RUN;
```

```
QUIT;
```

```
ODS HTML CLOSE;
```

Browsing to pagename.html, we again see the map of North Carolina. Viewing the HTML source code reveals the following reference.

```
<IMG SRC="mymap.gif" border="0">.
```

GENERATING A MAP FROM THE WEB

Leaving the interactive session behind, we place our SAS code in a program library defined to the Application Dispatcher. We then generate the map using either a URL or a Web form. With Application Dispatcher, one usually supplies an ODS HTML BODY value of _WEBOUT (DYNAMIC) to return output to the browser window. In this instance, however, when we change the BODY parameter to _WEBOUT, we find that the reference to the GIF file (in the current directory) no longer works. The solution is to use a BASE parameter. The BASE parameter provides the URL prefix to correctly locate the GIF file from an HTML page. A complication introduced by using the code below is that the reference to the image would need to be manually edited in the HTML file if we wanted to place the two files elsewhere.

```
ODS LISTING CLOSE;
```

```
FILENAME o 'directory/path/for/files';
```

```
ODS HTML
```

```
  PATH=o
```

```
  BODY=_WEBOUT (DYNAMIC)
```

```
  BASE="complete-url-to-directory-o/";
```

```
PROC GMAP MAP=one
```

```
  DATA=dummy;
```

```
  ID county;
```

```
    CHORO mapv/ COUTLINE=black NOLEGEND
```

```
    NAME="mymap";
```

```
RUN;
```

```
QUIT;
```

```
ODS HTML CLOSE;
```

Whenever running graphics using Application Dispatcher, it is important to first close the ODS listing destination. Otherwise, SAS will attempt to write the listing output where the broker application resides, hence the first statement above.

There is another complication involving browser caching when using _WEBOUT. Even if browser preferences are set to always obtain files from the server, the GIF may still be pulled from

cache. The first submission would show the new GIF. Subsequent submissions would show the original GIF rather than new versions. Manually refreshing the browser window reposts the data (reruns the program), and the browser again uses the cached GIF. Aaughh! As a workaround, we can 'open image in new window' and then successfully refresh. Another workaround is to write the html to a file rather than to _WEBOUT. A simple data step can, using PUT statements, write an intermediate page to _WEBOUT containing a link to the map files. The user can click that link to bring the map page into the browser. Since the page showing the GIF is no longer _WEBOUT, it can be refreshed without resubmitting the job. The BASE parameter is not needed in this workaround. Instead, we revert to using a path defined in a FILENAME statement, causing the browser to look for the GIF in the current directory.

```
ODS LISTING CLOSE;
```

```
FILENAME o 'directory/path/for/files';
```

```
ODS HTML
```

```
  PATH=o
```

```
  BODY="myfile.html";
```

```
PROC GMAP MAP=one
```

```
  DATA=dummy;
```

```
  ID county;
```

```
    CHORO mapv/ COUTLINE=black NOLEGEND
```

```
    NAME="mymap";
```

```
RUN;
```

```
QUIT;
```

```
ODS HTML CLOSE;
```

```
DATA _NULL_;
```

```
FILE _WEBOUT;
```

```
PUT 'Content-type: text/html';
```

```
PUT;
```

```
PUT 'Click below to see your map';
```

```
PUT
```

```
'<a href="url-to-myfile.html">map</a>';
```

```
RUN;
```

SPECIFYING THAT THE MAP BE AN IMAGE MAP

For the examples thus far, the original PROC GMAP input data was sufficient. For an image map, however, SAS requires a third variable, a link variable, in the data to be mapped. This variable should hold the HTML link value for each observation, to cause each county to link to a distinct URL. To keep our example simple, we assume there exists an HTML page for each county named FIPScode.html. In other words, for a FIPS county code of 25, the county page is named 25.html so the variable value we need is a href="25.html". The HTML parameter on the CHORO statement causes the output to contain an image map.

```
PROC SQL;
```

```
CREATE TABLE dummy AS
```

```
  SELECT DISTINCT county,
```

```
  1 AS mapv,
```

```
  'a href=""||
```

```
    TRIM(LEFT(PUT(i,3.)))||
```

```
  '.html"' AS linkvar
```

```
FROM two;
```

```
QUIT;
```

```
ODS LISTING CLOSE;
```

```
FILENAME o 'directory/path/for/files';
```

```
ODS HTML
```

```
  PATH=o
```

```
  BODY="myfile.html";
```

```

PROC GMAP MAP=one
DATA=dummy;
ID county;
CHORO mapv/ COUTLINE=black NOLEGEND
NAME="mymap" HTML=linkvar;
RUN;
QUIT;
ODS HTML CLOSE;

DATA _NULL_;
FILE _WEBOUT;
PUT 'Content-type: text/html';
PUT;
PUT 'Click below to see your map';
PUT
'<a href="url-to-myfile.html">map</a>';
RUN;
    
```

Browsing to myfile.html reveals a map looking exactly like those already created. This time, however, when the mouse is passed over the state, links are visible and each county links to a different page.

LETTING THE USER CUSTOMIZE THE MAP

NAME	DESCRIPTION
St	State to map
yp	Image height in pixels
xp	Image width in pixels
hsqname	htmSQL filename. If mybase is entered, the link for the county with a FIPS code of 25 will be: mybase.hsql?co=25
progrname	Application Dispatcher program name. If a.b.sas is entered, the link will be: url?_program=a.b.sas&_service=default&co=25
mapcolor	Map color
co	County outline color
transp	Specifies a transparent background
cb	Image background color

Table 1

The fields listed in **Table 1** are parameters the user can customize through the Web form shown in **Figure 2**.

The screenshot shows a web form titled "Generate a state imagemap". It is divided into three main sections: "General Options", "Link Options", and "Color Options".

- General Options:** Includes a dropdown menu for "State" (set to "North Carolina"), and two input fields for "Optional ypixel size (try 300)" and "Optional xpixel size (try 300)".
- Link Options:** Contains explanatory text: "Three choices: htmSQL, application server, or html. County parameter gets appended to htmSQL or application server links. No entry defaults to html link (25.html where fips county code is 25)." It has an input field for "optional htmSQL page name (enter mymap to get mymap.hsql?co=25)", followed by "OR" and an input field for "optional application server program name (mylib.mypgm.sas)".
- Color Options:** Includes input fields for "Optional map color.", "Optional county outline color.", and "Optional background color.". There is also a checkbox for "Transparent background." and "OR" text.

At the bottom right of the form are two buttons: "Generate Map" and "Reset".

FIGURE 2

When using Application Dispatcher, name/value parameters from a form or URL are used in a SAS program as macro variables. Placing the program code inside a macro enables the use of macro programming language. The following code makes use of all the options listed in the form.

```

%GLOBAL st transp xp yp cb co hsqname progrname
mapcolor;

%MACRO amap;

DATA two;
SET maps.counties;
IF state=STFIPS("&st");
RUN;

PROC GPROJECT DATA=two OUT=one;
ID county;
RUN;

PROC SQL;
CREATE TABLE dummy AS SELECT
DISTINCT county,
1 AS mapv,
'a href=""||
%IF &hsqname NE %THEN
"%TRIM(&hsqname)"||
    
```

```

'.hsq!?'co=||
LEFT(PUT(county,3.))||
""
;
%ELSE %IF &programe NE %THEN
"%TRIM(&_URL)"||
'?service=_default&_program=||
"%TRIM(&programe)"||
'&co=||
LEFT(PUT(county,3.))||
""
;
;
%ELSE
TRIM(LEFT(PUT(county,3.)))||
'.html"'
;
AS linkvar
FROM two;
QUIT;

GOPTIONS RESET;
ODS LISTING CLOSE;
FILENAME o '/mypath';
ODS HTML BODY='amap.html' PATH=o ;
GOPTIONS DEVICE=GIF
%IF "&transp"="yes" %THEN TRANSPARENCY;
%IF &cb NE %THEN CBACK=&cb;
%IF &xp NE %THEN XPIXELS=&xp;
%IF &yp NE %THEN YPIXELS=&yp;
;
%IF &pattern1 NE %THEN
PATTERN1 COLOR=&mapcolor;;
PROC GMAP MAP=one
DATA=dummy;
ID county;
CHORO mapv/
COUTLINE=
%IF &co EQ %THEN black;
%ELSE &co;
HTML=linkvar NAME='forweb' NOLEGEND;
RUN;
ODL HTML CLOSE;

DATA _NULL_;
FILE _WEBOUT;
PUT 'Content-type: text/html';
PUT;
PUT '<title>Image Map Results</title>';
PUT '<font size=+3 align=center>'
'Click the link below to see your map.'
'</font><p>';
PUT '<font size=+2 align=center>'
'You must perform'
'<font color=red>two</font>'
'saves to the same directory.</font><p>';
PUT '<table><font align=center>'
'<tr><td>'
'"save as" for the html'
'</td><td>'
'<font size=+2 color=orange>and</font>'
'</td><td>'
'"save image as" for the gif file.'
'</td></tr><tr><tr>'
PUT '<tr><td>'
'<font size=+2 color=green>ok</font>'
'to rename html</td><td>'
'<font size=+2 color=orange>but</font></td>'

```

```

'<td><font size=+2 color=red>do not</font>'
'rename the gif file';
PUT '</td></tr><tr><td>';
PUT
'<a href="myurlpath/amap.html">'
'<font size=+3>map</font></a>'
'</font></td></tr></table>';
PUT '<font size=+3 color=orange>NOTE:</font>'
'Make sure your browser refreshes both html '
'and image for repetitive map generations.';
RUN;
%MEND amap;

%AMAP

```

A CUSTOMIZED EXAMPLE

The user requests a map of North Carolina 200 pixels tall by 300 pixels wide. Links are to be htmSQL links. The htmSQL page name is hpage.hsq. The map should be gray with counties outlined in white on a black background. **Figure 3** shows this request. **Figure 4** shows the message returned to the browser window. The message prompts the user concerning refresh issues and provides instructions regarding moving the files to another location. The user clicks on the word 'map' to reveal **Figure 5**.

Generate a state imagemap

General Options

*State

Optional ypixel size (try 300)

Optional xpixel size (try 300)

Link Options

Three choices: htmSQL, application server, or html.
County parameter gets appended to htmSQL or application server links.
No entry defaults to html link (25.html where fips county code is 25).

optional htmSQL page name
(enter mymap to get mymap.hsq!?'co=25)

OR

optional application server
program name (mylib.mypgm.sas)

Color Options

Optional map color.

Optional county outline color.

Transparent background.

OR

Optional background color.

Figure 3

Click the link below to see your map.

You must perform **two** saves to the same directory.

"save as" for the html **and** "save image as" for the gif file.

ok to rename html **but do not** rename the gif file

[map](#)

NOTE: Make sure your browser refreshes both html and image for repetitive map generations.

Figure 4

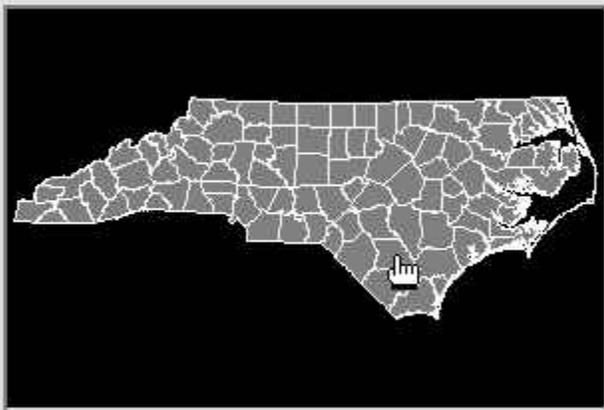


Figure 5

With the mouse is positioned over the state, and the URL appears as in **Figure 6**.

<http://statweb.unc.edu/hpage.hsql?co=17>

Figure 6

SUMMARY

This application uses the Version 8.0 SAS Output Delivery System and SAS/GRAPH with Version 2.0 Application Dispatcher to process graph customizations from a Web form. The resulting graph is an image map of any state in the US with the associated links taking one of three forms: Application Dispatcher calls, htmSQL calls, or html page references. Size and colors are to the user's specifications. Using an intermediate page for output, which links to the requested image map, circumvents problems related to browser caching.

CONTACT INFORMATION

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

Carol Martell
 UNC Highway Safety Research Center
 730 Airport Rd, CB# 3430
 Chapel Hill, NC 27599-3430
 Work Phone: 919-962-8713
 Fax: 919-962-8710
 Email: carol_martell@unc.edu

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

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