Paper 13-28

The Power of Pictures and Paint: Using Image Files and Color with ODS, SAS®, and SAS/GRAPH®

LeRoy Bessler, Bessler Consulting & Research, Fox Point, Milwaukee, Wisconsin, USA

Abstract

You might want to use a logo. You can create images with SAS/GRAPH, other software, or a digital camera. Hardcopy photos or illustrations, or even a 3D object, can be converted to image files with a scanner. There are stock photos and clip art, free or for fee, on the web or bundled with software. This tutorial shows how to create image files with SAS/GRAPH, and how to imbed an image in a graph, but emphasis is on numerous ways to imbed image files, from any source, in web pages (and other output), using the SAS Output Delivery System. Besides imbedding images in tables, graphs, web pages, etc., you can import images into Microsoft Word or PowerPoint, for use as is, or after adjustments there (clipping, resizing, and other manipulation) without fancy expensive image editing tools. You will learn a trick to export image files for use elsewhere, after editing them in Word. The tutorial compares characteristics, advantages, and disadvantages of image file types (GIF vs. JPEG), and explains how to minimize web download wait. It also discusses color systems, SAS colors, crucial color concerns dependent on output destination (especially the web), and design for effective visual communication (not decoration) with color. SAS 8.2 was used on Windows 98 Second Edition. ODS (and SAS/GRAPH) experience is assumed.

Introduction

This is a tutorial about effective visual communication of information, using images and using color, taught with examples, code, and guidelines.

My scope is limited to what fits in the page and time constraints.

Slides for the stand-up presentation, and code used here and there, as well as code omitted in this paper (its omission is usually shown here as " . . ."), are available via email in a zip file. If you have comments or questions, or suggestions on what to cover in a future edition, please send them.

The Power of Simplicity

The Number One Usability Concern of Web Users = download time.

Simple web pages download faster. So, avoid excessive use of images, if there only for decoration. Unneeded complexity can distort, distract, and delay communication. A sparse web page focuses attention. The Power of Simplicity—Shows Them What's Important.

Using Images for Communication

Image File Formats for the Web

To reduce download time, all image file formats for the web use compression. This is the removal of "extra" data from the file that was the original image. GIF and PNG use lossless compression. JPG uses lossy compression, but the result usually may not be noticeable.

GIF (Graphic Interchange Format) is used primarily for graphs, logos, simple cartoon-like illustrations, icons, and other images with areas or segments of solid color (also called "flat color"). Its advantages include: (a) small file size—quick download; (b) support for transparency; and (c) support for animation. Transparency permits the web page background to show through your image—if you wish that to happen. It eliminates the effect of a box around each of your graphic images. Though animation is not usually relevant for business graphics, it can be used to show evolution over time of measurements that have to be displayed via a map (e.g., population change over time). GIF animation is easier and cheaper than any other animation alternative. GIF's main "theoretical" disadvantage is the fact that it supports only 256 colors (8-bit color). This is not necessarily a serious limitation. There are only 216 "browser-safe" (a.k.a. "web-safe") colors—more about them later.

If you create an image with SAS/GRAPH, you cannot use more than 255 colors. It is unlikely that you would design a graph with anywhere near 255 colors, much less want to use more than the 256-color GIF limit. GIF is the oldest format used for web images, and the most popular choice, except when dealing with photographs. It is universally supported: all browsers, all versions of browsers. All web page screen images in this paper were captured with Version 7.04 of Jasc Paint Shop Pro, and saved as GIF files. Except for two photos and the logo, all images in my web pages were created as GIFs with SAS/GRAPH.

JPG (a.k.a. JPEG, for Joint Photographic Experts Group) is used for photographs, which are continuous tone images. Its main advantage is its support for 24-bit color (16.7 million colors). JPG uses a "lossy" compression. Some of the data is thrown away and not recoverable. You may not notice the difference, however. Despite compression, JPG files are usually larger and slower to download, because photographs require a lot of data to digitally define them. Also decompression by the web browser takes extra time. JPG has no support for transparency or animation. The JPG compression algorithm works well for continuous tone color, but areas of "flat" (or solid) color, areas with sharp edges, lines, and text are adversely affected. Though JPG supports 16.7 million colors, if it is displayed on a 256-color display (there are a surprisingly large number of these out there), the colors are remapped to the 216-color browser-safe palette and "dithered". Dithering is a way to try to simulate the unsupported colors.

> Tip: Before you edit a JPG image, be sure to make a duplicate of the original. When you save the edited image, it gets compressed further. And if you edit that result and save it, it gets recompressed yet further. After enough progressive compression, the result will be unacceptable.

> Tip: You can mitigate the effect of long download times for a JPG image (or any other image type). Define a tiny box (1 pixel by 1 pixel) at the top of a web page, and "load" the JPG file into it. E.g., you can do this with your home page. You can load multiple such boxes with images. If a linked web page, which the viewer gets to later, references that image file at full size, the image file will be retrieved from a place on the viewer's PC disk where it has been stored, rather than being downloaded again over the network. This creates the illusion of instant download. Of course, enough time must have elapsed since opening the initial web page. Later discussion will explain how to set up these boxes.

PNG (Portable Network Graphic) is a lossless compression format, which results in files that are larger than JPG for the same image content. Besides the 8-bit color supported by GIF, and 24-bit color supported by JPG, 16-bit gray scale (very useful for medical images) and 48-bit color are supported by PNG. There are unique features of PNG, including gamma correction (see below) and variable levels of transparency, but they are not supported by all creation and browsing tools. The main disadvantage of PNG is that, despite its benefits, it does not enjoy the same universality of support as GIF and JPG. Older web browsers and some PCs might not be able to display it. All illustrations for my discussion of color in this paper are 300 dpi PNG files.

> **Gamma Correction.** Displays vary in their performance. Images created on a Mac look dark on a PC or Unix. Images created on a PC or Unix look washed out on a Mac. Gamma Correction enables a PNG image to display with its intended brightness regardless of platform.

Reducing Image Size without File Compression

Case 1. Creating an image file with SAS/GRAPH. For how to create a minimalist image that is communication-effective (in fact, to see how minimalism enhances communication effectiveness), please see my Beginning Tutorial (Reference 1). Simpler images are smaller and download faster.

Case 2. Working with a pre-existing image. This could be a stock photo, a digital photo made with your own camera, your output from scanning, etc. You could use a fancy image-editing tool. If you don't have one, and don't want to get one, all hope is not lost. Let's assume that your only need is to crop the image (i.e., to trim off areas from one or more of the edges of the image), and possibly re-size it (you could reduce it, or enlarge it). Here is my solution. Open MS Word to create a new document. Define your page and/or column size any way you like. Click on Insert. Click on Picture. Click on From File. Find the .GIF or .JPG (or other file type) you want. Highlight the file, and click on Insert. Regardless of the "native" (i.e., natural) size of the image, it will be imported to the space in your page and column. If it is too big, it will shrink to fit. If it is too small, it will occupy only part of the space. Right click on the image, and then Format Picture. This brings up a Format Picture window with five active tabs. The ones of interest are Picture and Size. On the Picture tab, use the Crop From section, which allows you to remove a strip of any width from the Left and/or Right and/or Top and/or Bottom. On the Size tab, use the Size and Rotate section. When adjusting size, do not change options in the Scale section. The Lock Aspect Ratio box in the Scale section may already be checked. If not, check it before changing the size. With Lock Aspect Ratio checked, you can change either Height or Width, and the other dimension will be automatically scaled to prevent image distortion. Having cropped and/or resized the image, now "export" it. Here's how. In this MS Word "pseudo-document", click File, then Save As. For "Save as type", select "Web Page". Enter a folder and filename, and click Save. Now go to that folder. There you will find an HTML file, with the name you selected. It is a web page containing only the edited image. This is NOT what you want. You will also find a subfolder with name "YourHTMLfilename_files", where by "YourHTMLfilename" I mean the name specified when you saved the web page. Inside the folder are three files. The one you want is called image002, which is the result of your editing. Rename image002 to something meaningful, and move it to a permanent folder location. The other three outputs can be deleted.

Getting Pre-existing Images

Formal Means. You can find stock photo and clip art suppliers on the Internet. Any search engine should help you. These images fall into three categories: (a) free and unrestricted re-use; (b) free re-use for non-commercial purposes; and (c) for-fee only.

Informal Means. Besides using the above formally offered sources of images, you can save to your disk almost any image that is displayed by a web page. If you place your mouse over the image, and right click it, a window opens to permit you to do several things. You can display its Properties: name of the file; type of image file; fully qualified URL for where the source image file is; file size in bytes (i.e., to judge the impact of storage on your disk); dimensions of the image in pixels; and dates of creation and modification. You can save the image to your disk, in either the same format, or as a BMP (always a much bigger size file).

Caution. Some images are marked with a copyright. Even if it is permissible to reuse such an image for non-commercial purposes, it would be tacky, and probably illegal, to crop off the copyright, and republish the image. A second-hand viewer could assume there was no copyright asserted for the image, and might reuse it commercially.

Sizing Considerations for Web Images

Screen Resolution. Sizing a web image in centimeters or inches has no value. The physical dimensions of an image are a function of screen resolution. It is frequently claimed that the typical screen resolution is as low as 72 pixels per inch. It is a fact that you can get screen resolutions at least as high as 133 pixels per inch. So, e.g., an image with a width of 400 pixels will display with a width of 5.5 inches on the low-resolution screen, but only 3 inches on the high-resolution screen. In any case, you should plan your web page for the lowest probable resolution, which is presumably 72 pixels per inch. Also, PCs with higher resolutions are often set to lower resolutions, perhaps since the user does not know what is possible.

"Live" Area in the Web Browser Window. Not all the space on the PC screen is available for web page display. Besides a strip that Windows uses, the web browser uses space on all four sides, especially the top. How much depends on the web browser, and its version. And "progress" is not necessarily increasing the live space. Just as screen resolution determines physical size of an image, so, too, it determines how much live space is available to display your images and the rest of your web page. Limiting consideration to the smaller screen pixel dimensions, 640 X 480, 800 X 600, and 1024 X 768, some minimum live areas are, roughly, 625X275, 785 X 400, 1000 X 565.

Scrolling. Do not use scrolling, especially in the horizontal direction, as a web page design solution. The web page should be designed to fit entirely in the smallest probable web browser live space in which it will be displayed. And, certainly, an image—which is intended to be viewed as a whole—should almost never be sized so as to require scrolling. But a large detailed map would be a valid application for scrolling.

Creating Images with SAS/GRAPH

```
goptions reset=all;
goptions device=gif; /* make a simple GIF */
goptions gsfname=anyname gsfmode=replace;
filename anyname
        "&IMGPATH.\ImageAtTopOfBody.gif";
goptions border cback=CX99FFFF;
goptions ftext=CENTX htext=30 PCT;
goptions xpixels=50 ypixels=50; /* size */
proc gslide; /* could be any graph PROC */
title j=C move=(+0 PCT,-5 PCT)
     'Top of' j=C 'Body';
run; quit;
```

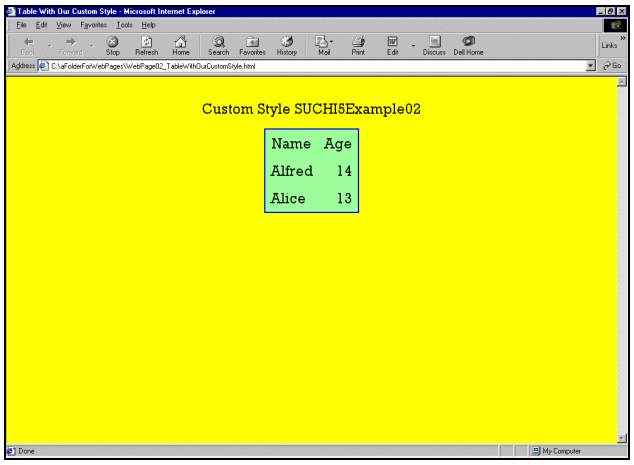
&IMGPATH is a symbolic (or macro) variable that has been previously initialized with a %LET statement. The above code creates a boring image of two lines of text in a small box. Most images in this tutorial are boring. They are not meant to impress you. They are convenient, compact objects used to illustrate techniques for imbedding images in web pages. It is necessary in some situations to imbed a blank (or, really, transparent) GIF as filler. Here is the code used to make one:

```
goptions reset=all;
goptions device=gif; /* make a filler GIF */
goptions gsfname=anyname gsfmode=replace;
filename anyname "&IMGPATH.\Space50X50.gif";
goptions noborder transparency;
goptions xpixels=100 ypixels=50;
proc gslide;       /* make an empty "slide" */
title;
run; quit;
```

The Base Web Page Before Image Enhancements

Most of the examples involve a simple table, created with PROC PRINT and the Names and Ages of some students from SAS sample data set SASHELP.CLASS. The base ODS Style used is a custom style developed in my web publishing projects with Dr. Francesca Pierri at the University of Perugia. You can request improved ODS code for our style via email, or find an early edition in Reference 3. Colors used for image examples below are NOT inherent to the custom style. Nor are they necessarily recommended for your web page design. They were chosen to be bright—for demonstration purposes—and to be browser-safe. You will see use of base style Styles.SUCHI5Example02, which is a derivative of our custom style, but amended to the purposes of a (more detailed) tutorial on images that I presented at the Fifth Annual SAS Users Chicago International Conference. Many examples in this SUGI Advanced Tutorial are drawn from that introductory, longer tutorial.

Below is a display of the basic web page. "It sets the data free." There are no grid lines or cells for the data. Just as graphic software defaults reflect the graphic roots of grid paper, pen, and ink, ODS-formatted tables—because there now is software support—tend to be needlessly complicated with the grid from the spreadsheet software venue, which in turn was influenced by its hardcopy roots: a big pad of paper worksheets with a pre-printed grid for rows and columns.



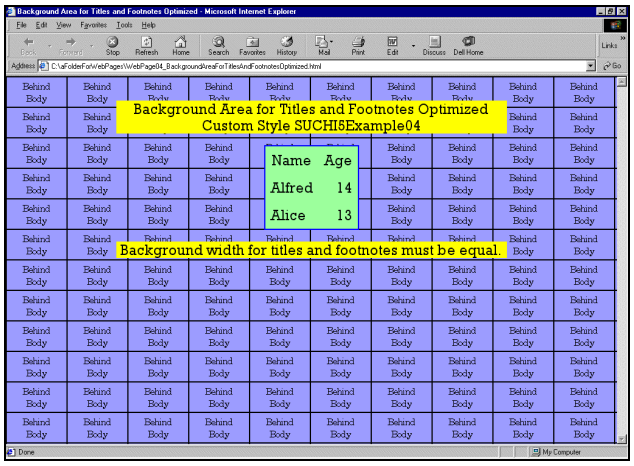
Here is the code to create this web page:

```
ods listing close; ods noresults;
ods html path  = "&PATH" (url=none)
         body  = "WebPage02.html"
         style = styles.SUCHI5Example02;
goptions reset=all;
proc print data= . . . ;
id name; var age; run;
ods html close; ods listing;
```

Getting Started with Image Imbedding

To work with images in ODS, we will primarily use these style parameters: backgroundimage=, preimage=, postimage=, prehtml=, posthtml=. Their meaning and use is obvious by comparing web page examples with code. The last two parameters involve HTML coding, but it is neither difficult nor obscure. They enable you to adjust the position and spacing when use of preimage= or postimage= does not suffice.

The first example is application of a background image. If the image is smaller than the web page, it fills the page with tiles, as shown below.



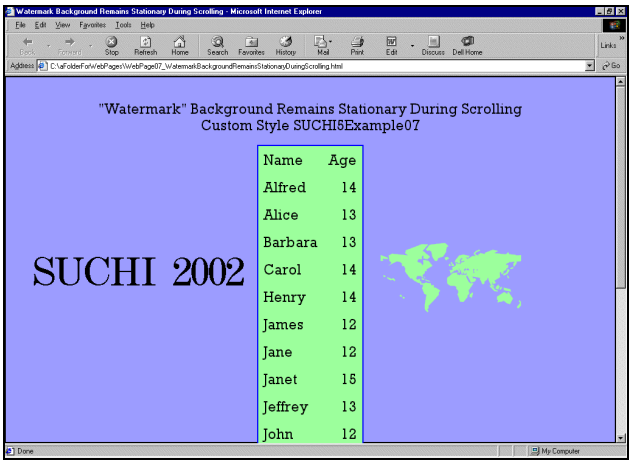
With ODS, titles and footnotes are displayed each in a table of their own, with a user-specifiable background color. If you do not adjust the width, the result is a pair of peculiar looking bars across the full width of the web page. Here is the code to create the style for this web page:

```
proc template;
edit Styles.SUCHI5Example02
  as Styles.SUCHI5Example04;
style body / backgroundimage =
      "&IMGPATH.\ImageFileNameX.gif";
style SysTitleAndFooterContainer /
      outputwidth = 65%;
end; run;
```

I do not recommend a tiled background area. The concept is typically used, however, to compose a finely textured background out of tiles—which I do not recommend either. The best background is a solid color in high contrast with the foreground text color.

However, if using a background, it can be useful to make it stay fixed during scrolling, and to make the title and footnote areas transparent. The fixed background is called a "watermark".

By transparent, I mean that the text of the titles and footnotes are opaque, but their background areas, which were yellow in the above example, now permit the web page background to show through.



Here is the code to create the style for this web page:

```
proc template;
edit Styles. SUCHI5Example02
  as Styles.SUCHI5Example07;
style body /
  backgroundimage =
  "&IMGPATH.\SimpleImage.gif"
  watermark = ON;
style SysTitleAndFooterContainer /
      background = _undef_;
style systemtitle  / background = _undef_;
style systemfooter / background = _undef_;
end;
run;
```

A background image too often is merely decorative, or an inessential add-on. However, it can be useful and appropriate to use a logo, or a consistent "frame" or backdrop, for every web page in a set.

The ODS style which provides this standard backdrop becomes the universally applied style for every web page in the set.

When a specific web page needs some other ODS-template-enabled feature, that standard style serves as the parent style for PROC TEMPLATE code. It is used as in this statement:

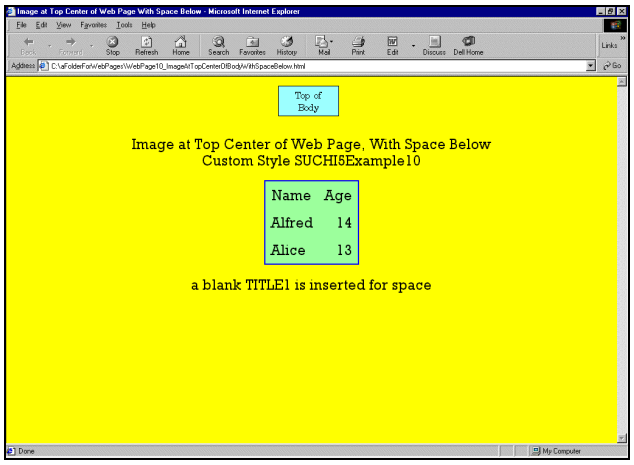
```
edit Styles.StandardBase as
     Styles.SpecialFeature;
```

Tip: You can make your SAS/GRAPH output transparent (i.e., with only the text and graphic elements opaque to the web page background) by specifying "GOPTIONS TRANSPARENCY;" and using the GIF driver.

Let's move on to foreground images.

First, we will use a preimage for the web page body.

By default, it would be positioned in the upper left corner. But it can be centered, as in the example below. A blank TITLE1 is used to get some space between the image and the first real title line.



Here is the code to create this web page:

```
proc template;
edit Styles.SUCHI5Example02
  as Styles.SUCHI5Example10;
style body /
  preimage = "&IMGPATH.\ImageAtTopOfBody.gif"
  prehtml = "<center>" ; /* center the image */
end; run;

ods html path = "&PATH" (url=none)
         body = "WebPage10.html"
         style=styles.SUCHI5Example10;
title1 ' '; /* blank title */
title2 "Real Title";
proc print data= . . . ;
 . . .
run;
ods html close;
```

How To Hide Preloaded Images for Use Later on Other Pages.

Use the same template code as above, with one change:

```
style body / prehtml =
"<IMG src='&IMGPATH.BigImage1.gif'
height=1 width=1 border=0>
 /* more image files here */
 <IMG src='&IMGPATH.BigImageN.gif'
height=1 width=1 border=0>";
```

Use "border=0" to hide the visible box that would be drawn around the invisible image.

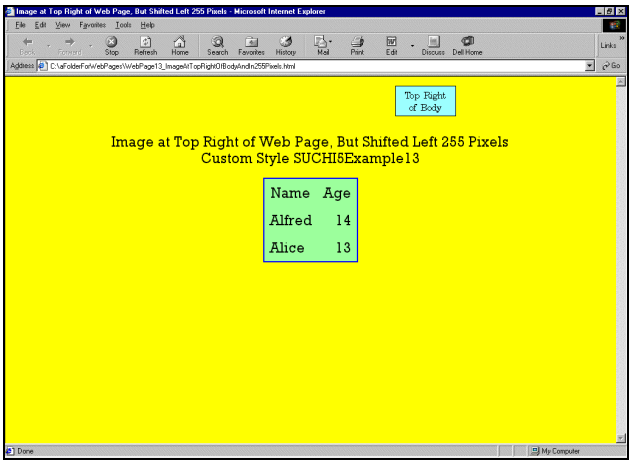
You can insert such IMG tags "by hand" in your HTML file after it has been built with anything in it that you want or need, and built by any means, not necessarily ODS. They should be placed after the BODY tag.

You can display the web page, click on View, then Source. That opens the HTML file with NotePad. You can edit the source. When you close it, NotePad will ask you, "Do you want to save the changes?" Click Yes. When the web page is refreshed, your changes take effect.

Now let's position the image above the titles, but toward the right end of the title block. It can be done by aligning the image in the upper right corner, and then shifting it back to the left with a horizontal spacer.

It is an idiosyncrasy that when an image is aligned in the upper right corner of the web page, it reserves no vertical space at the top of the page. The solution for this problem is a TITLE1 to create a blank title with sufficient vertical space to force the first real title below the image.

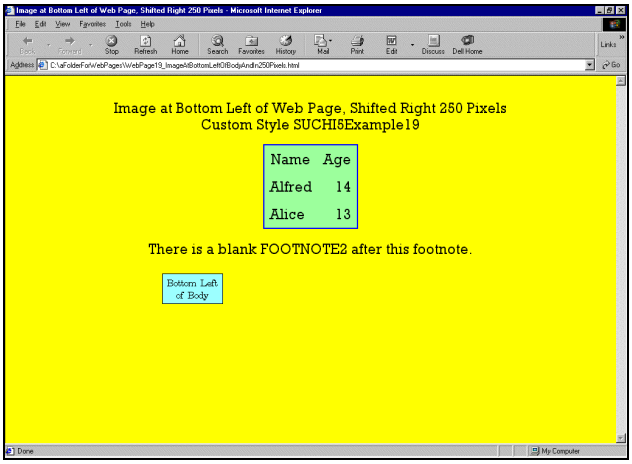
Here is the result.



Here is the code to create this web page:

```
proc template;
edit Styles.SUCHI5Example02
  as Styles.SUCHI5Example13;
style body / prehtml =
"<IMG src = &IMGPATH.\ImageAtTopRightOfBody.gif'
      align='right' hspace=255 >";
end; run;

ods html . . . style=styles.SUCHI5Example13;
title1 h = 1 cm ' ';
 . . .
ods html close;
```

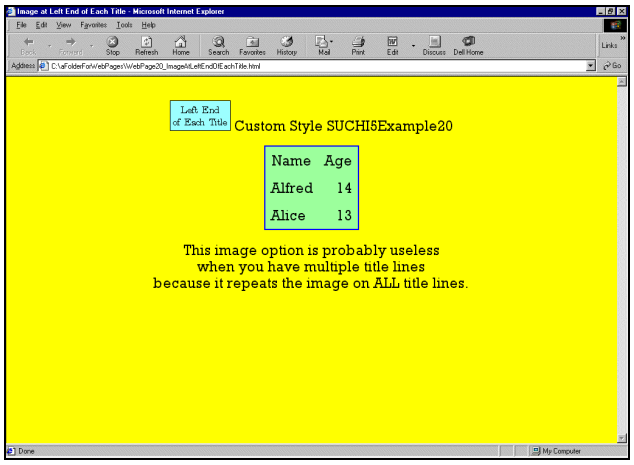
Here is the "converse" of the above example.



Here is the code to create this web page:

```
proc template;
edit Styles.SUCHI5Example02
  as Styles.SUCHI5Example19;
style body / posthtml = "<IMG
 src='&IMGPATH.ImageAtBottomLeftOfBody.gif'
  align='left' hspace=250 >";
end; run;
ods html . . . style=styles.SUCHI5Example19;
footnote1 "Real Footnote";
footnote2 ' '; /* blank footnote */
proc print . . . ; run;
ods html close;
```

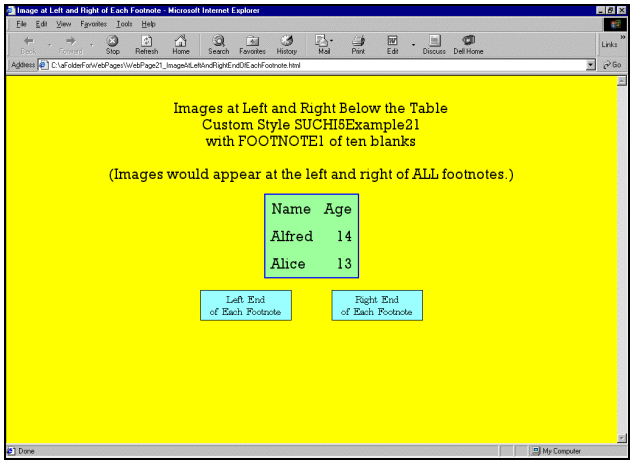
Rather than imbed the image before the titles or after the footnotes, it is possible to put an image in a title line, as below.



This is probably useless if you have multiple title lines, since the image repeats on every title line. Here is the code to create the style for this web page:

```
proc template;
edit Styles.SUCHI5Example02
  as Styles.SUCHI5Example20;
style systemtitle / preimage =
  "&IMGPATH.\ImageAtLeftEndOfEachTitle.gif";
end; run;
```

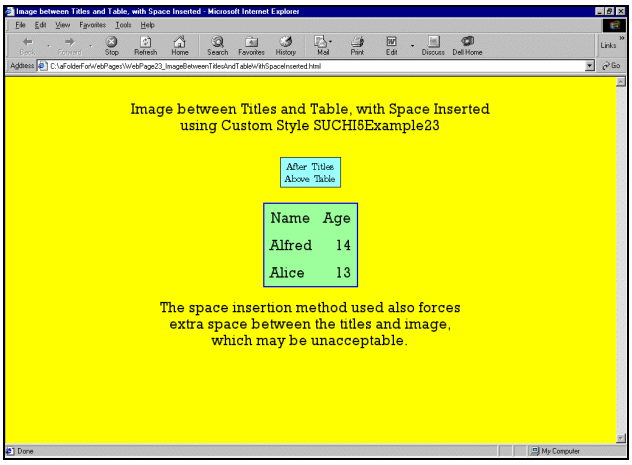
And/or you can put images on the footnote line, as shown below. In fact, you don't even need any footnote text, as you can see.



Here is the code to create this web page:

```
proc template;
edit Styles.SUCHI5Example02
  as Styles.SUCHI5Example21;
style systemfooter /
  preimage  =
"&IMGPATH.\ImageAtLeftEndOfEachFootnote.gif"
  postimage =
"&IMGPATH.\ImageAtRightEndOfEachFootnote.gif";
end;
run;

ods html . . .
    style=styles.SUCHI5Example21;
footnote6 '          ';
proc print . . . ;
run;
ods html close;
```

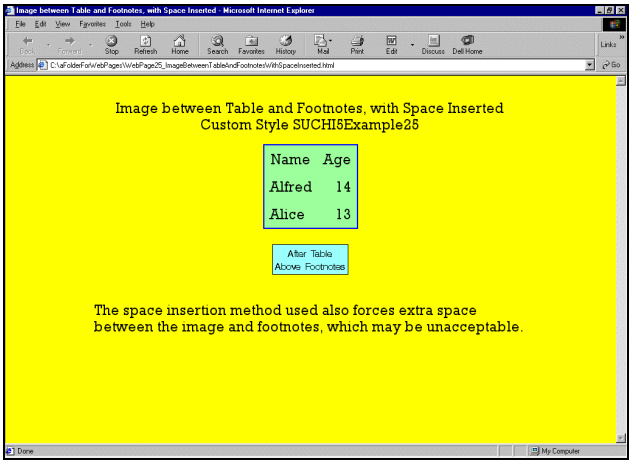
You can put the image between the titles and the table, as shown below.



By default, the image would be placed too close to the table. Here is the code, which includes a space insertion of 20 pixels, to create the style for this web page:

```
proc template;
edit Styles.SUCHI5Example02
  as Styles.SUCHI5Example23;
style table / prehtml = "<IMG
  src='&IMGPATH.\ImageBetweenTitlesAndTable.gif'
  align='bottom' vspace=20 >" ;
end; run;
```

And here is an image between the table and the footnotes.

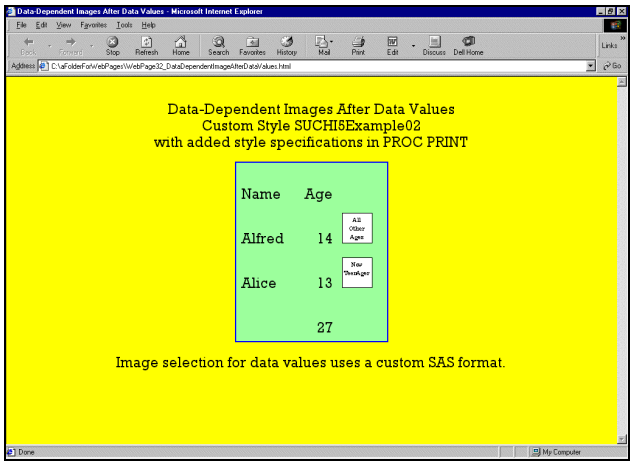


And the code to create the style for this web page:

```
proc template;
edit Styles.SUCHI5Example02
  as Styles.SUCHI5Example25;
style SysTitleAndFooterContainer /
  outputwidth = 72%;
  /* squeeze the area to be able to use j=L with indent */
style table / posthtml = "<IMG src=
  '&IMGPATH.\ImageBetweenTableAndFootnotes.gif'
  align='top' vspace=25 >" ;
end; run;
```

Images can be imbedded in various parts of a table. You can put an image to the left of (before) the contents of every cell, and/or to the right of (after) the contents of every cell, and/or behind the contents of every cell. You can put an image before and/or after and/or behind the table column headings. You can put an image before and/or after and/or behind an ID column heading. You can put an image before and/or after and/or behind ID values. You can put an image before and/or after and/or behind data values in table columns or table cells. These five types of image applications all require some special coding to get things

to look as you like. If you have an interest in any of those cases, send me an email request for a solution. Below you can see an important variation on just one possibility from the extravaganza of possibilities that I worked on. It is the use of data-dependent images for table data.



The little box after Age value 13 says, “New Teenager, and the little box after Age value 14 says, “All other ages”. In a more realistic application, you might use different icons to flag different values or value ranges. Here is the code to create the web page:

```
proc format;
value ageimg 13 =
  "<IMG src = '&IMGPATH.\NewTeenAger.gif'
    hspace=15 >"
             other =
  "<IMG src = '&IMGPATH.\AllOtherAges.gif'
    hspace=15 >"; run;
ods html . . . style=styles.SUCHI5Example02;
proc print . . .
  style(total) = [just=right posthtml =
      "<IMG src = '&IMGPATH.\Space50BY50.gif'
      hspace=15 >"];
id name /
  style(header) = [just=left
      postimage = _undef_ posthtml =
      "<IMG src = '&IMGPATH.\Space8BY50.gif'>"]
  style(data) = [just=left
      postimage = _undef_ posthtml =
      "<IMG src = '&IMGPATH.\Space8BY50.gif'>"];
var age /
  style(header) = [just=right posthtml =
      "<IMG src = '&IMGPATH.\Space50BY50.gif'
        hspace=15 >"]
  style(data) = [just=right posthtml = ageimg.];
ods html close;
```

The base style is still the base established for the first example shown, the table with no images imbedded anywhere in the web page. All the heavy work has been done by using the STYLE option available in PROC PRINT since Release 8.2.

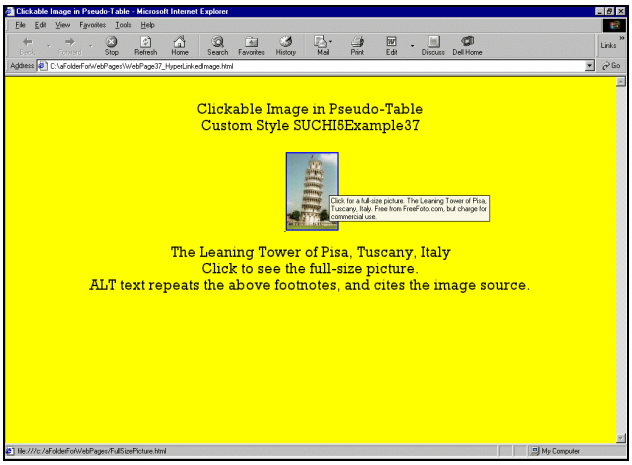
I am sure you can understand the purpose of posthtml = ageimg. in style(data)= for the var age statement. But why all of these other complications? Well, as soon as you insert an image anywhere in the cells of a table, be it in a column heading, a row header, a data cell, etc., the situation gets complicated by the need to preserve expected common alignments of values all the way down each column, and the expected common height of cells all the way across each row. There is not space here to show you what happens, but I suggest that back at your office you create the web page, but omit all the PROC PRINT style code except that which uses the ageimg. format. If you restore the PROC PRINT style options one at a time, you will better understand their respective effects On your ODS HTML statement you can use Styles.Default. (Use of the ODS default, instead of the custom style used here, will not cause any problem or diminish the value of your experiment.) You can create both of the space filler GIFs. The code for 50BY50 was shown earlier. If you do the exercise, you will see why those amendments had to be done.

Delivery of Full-Size Images Only Upon Demand

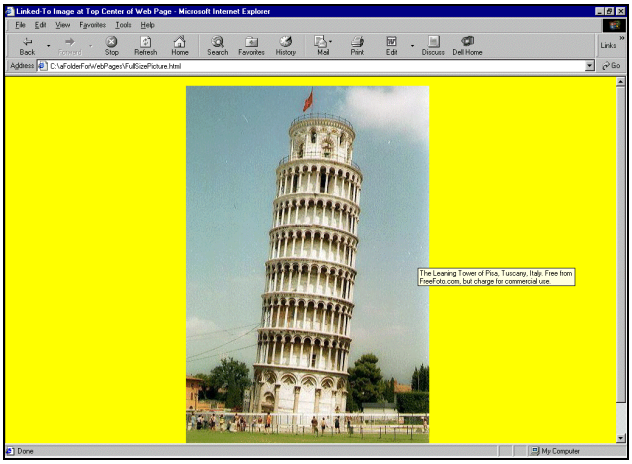
It is essential to make wise use of the limited live space on web pages, and to minimize download time to deliver images. Web pages may include images that a web viewer may not be interested in, or that other web viewers are content to see at reduced size. Rather than forcibly present the full image, a predecessor web page can present a thumbnail image. Clicking on the thumbnail links to a display of the full image.

The thumbnail may be a size-restricted version of the full-size image file that is downloaded, or a cropped version of the full-size image file. If the thumbnail is just a size-restricted version, then the click brings up the full image instantly; if a different file, then the click triggers a download and delay. By “size-restricted” I mean the HTML IMG tag HEIGHT and WIDTH parameters are used to determine display size, rather than the natural pixel-count dimensions of the image.

Here is a pair of thumbnail-linked web pages.



Here is the linked web page that displays the full-size image.



ALT Text. The popped-up boxes of text where I rested my mouse are “ALT text”. You can create ALT text for your images with ODS, and/or with SAS/GRAPH. When working with already finished images, your own or downloaded ones, you have another option. In the exercise where I explained how to do image editing with MS Word, there is a point in the process—when you bring up the Picture and Size tabs—that a Web tab is also available to which you can click if you wish to assign “Alternative text” for the image. Please see Reference 2 for a fuller discussion of ALT text, and why to use it.

Here is code to create the first web page in two different ways.

Option 1. Using a custom thumbnail image, which could be a cropped version of the full image, not just a reduced version.

```
ods html . . . /* Custom Thumbnail Preview */
    body  = "ThumbnailImage1.html"
    style = styles.SUCHI5Example37;
 /* titles and footnotes here */
proc print data=fakedata noobs label;
var fakevar / style(data) = [just=right
    posthtml =
    "<a href='&PATH.\FullSizePicture.html'>
     <IMG src='&IMGPATH.\PisaThumbnail.jpg'
          alt='...' > </a>"]; run;
ods html close;
```

Option 2. Use the full-size image file, but reduce its display area on the web page with HTML controls for HEIGHT and WIDTH.

```
ods html . . . /* Reduce Full Image for Thumbnail Preview */
    body  = "ThumbnailImage2.html"
    style = styles.SUCHI5Example37;
 /* titles and footnotes here */
proc print data=fakedata noobs label;
var fakevar / style(data) = [just=right
    posthtml =
    "<a href='&PATH.\FullSizePicture.html'>
    <IMG src='&IMGPATH.\PisaFullSizePicture.jpg'
    height=125 width=83 alt='...' > </a>"]; run;
ods html close;
```

Here is the code to create the second web page:

```
ods html . . . /* Full-Size View */
    body = "FullSizePicture.html"
    style=Styles.SUCHI5PisaPhoto;
title1; footnote1; /* NO "hard" text on the page */
proc print data=fakedata noobs label;
var fakevar; run;
ods html close;
```

Here is the code to create the "fake data":

```
data fakedata;
length fakevar $ 1; label fakevar = '00'X;
fakevar = ' '; run;
```

A pseudo-table is used. It consists of one blank as the data, with an invisible heading. Here is the code to create the styles for the web pages:

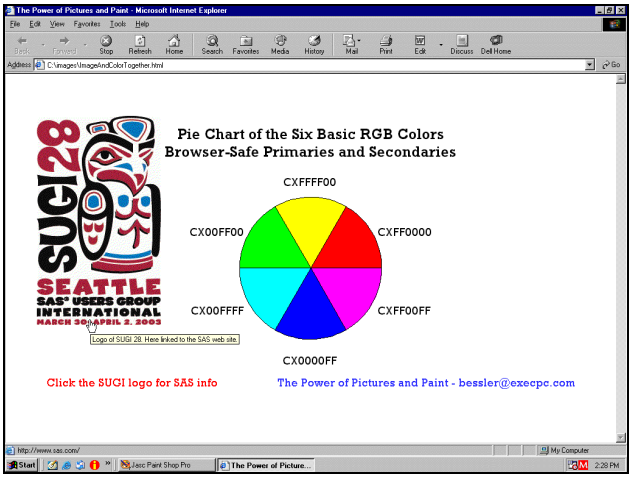
```
%MakeStdStyle(
StyleName=SUCHI5Example37,      /* for the full-size photo */
TableHeadingSize = 1,
TableDataSize    = 1,
WebPageBackgroundRGBcolor = CXFFFF00,
TableRGBcolor             = CXFFFF00,
TableFrame = VOID, TableSpacing = 0)
run;

proc template;                  /* for the thumbnail */
edit Styles.SUCHI5Example37
  as Styles.SUCHI5PisaPhoto;
style body / prehtml = "<center>
    <IMG src='&IMGPATH.\PisaFullSizePicture.jpg'
         alt=' . . . ' >";
end; run;
```

You may request the %MakeStdStyle macro code via email. Here are the functions of the macro parameter selections. The pseudo-table is designed to minimize space use and to make all *unwanted* parts of the table invisible in the context. Fonts for column heading and data are set to the minimum. All colors of anything that might show are set to the web page background color (RGB Yellow). All grid or frame elements are set to null or zero width. For Option 2, the thumbnail image is laid inside the pseudo-table, with nothing else visible. The full-size image sits, horizontally centered on its web page, above the tiny invisible pseudo-table—maximizing web page live space available for the image.

If you want ODS to create the HTML file for you, it is necessary to execute some SAS (tabular or graphic) PROC inside an ODS HTML code block. Hence, a pseudo-table or pseudo-graph must be created.

How To Imbed Images in Your SAS Graphs

In the web page of the pie chart below, the SUGI 28 logo has been imbedded. I hyperlinked the logo to www.sas.com, and supplied ALT text for it that says, "Logo of SUGI 28. Here linked to the SAS web site." The pie chart, rather than being based on real data, is contrived to be a sample color chart. The pie really should be shifted to the right to balance the picture, but my effort is focused on showing how to imbed the image. The key graphic programming tool for this is the SAS/GRAPH Annotate facility. I will not try to explain all of the Annotate code. Please see documentation and/or SAS Help, if needed. The White bar that is (invisibly) annotated behind the image is used simply as a means to make the image clickable. The HTML Annotate variable defines the hyperlink with HREF= and the ALT text with ALT=. The HTML variable is not yet supported by the IMAGE Annotate function. (Maybe it will be supported in a future release of SAS/GRAPH.) The Annotate IMAGE function requires you to identify the image file with the IMGPATH variable. For the IMAGE function, the (optional) Annotate STYLE variable is either TILE (the default) or FIT, as used here. When you use FIT, if you under-allocate or over-allocate space for the image, it will shrink or stretch to fit. If the ratio of the dimensions of your allocation does not match the ratio of the dimensions of your source image, the result will be distorted. I rested my mouse on browser display of the original image, and pressed the right mouse button to get to see the properties of the image. Knowing those pixel counts for horizontal and vertical, I was able to size the target area correctly. (When you specify TILE, the image is laid out like tiles, but starting at top left and sweeping across and down, to fill the allocated space. Of course, if the image is too big, you will not see tiles. It may just be truncated in one or the other direction, or both.) The Annotate WHEN variable is set to `'B'`, so that the image is laid down in the graphic area BEFORE the graph is drawn. In this way, you could use an image as a background, if you wish. You could overlay the graph, if that were your design objective, by setting WHEN to `'A'`, to have the image laid down in the graphic area AFTER the graph is drawn.



Here is the code to create this web page:

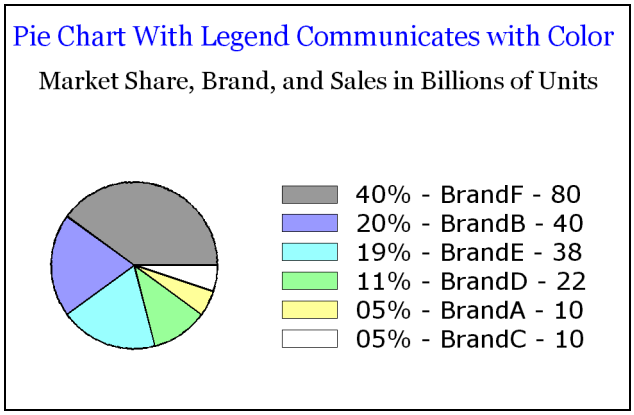
```
data piedata;
infile cards;
input @1  RGBcolor $8. @10 PieValue 1.;
cards;
CXFF0000 1
CXFFFF00 1
CX00FF00 1
CX00FFFF 1
CX0000FF 1
CXFF00FF 1
;
run;
```

```
data logoanno;
length function $ 8 imgpath html $ 200;
retain xsys ysys '3' when 'B';
function='move';
x=0; y=23.56;
output;
function='bar';
x=22.67; y=100;
style='solid'; color='CXFFFFFF';
html='href="http://www.sas.com" alt="Logo of
SUGI 28. Here linked to ..."';
output;
function='move';
 x=0; y=23.56;
output;
function='image';
x=22.67; y=100;
imgpath="&Folder.\sugi28logo.gif";
style='fit';
output;
run;

goptions reset=all;
ods html path="&Folder" (url=none)
style=Styles.our_style gtitle gfootnote
body="ImageAndColorTogether.html"
(title="The Power of Pictures and Paint");
goptions device=gif transparency;
goptions xpixels=900 ypixels=450;
goptions ftext='Verdana' htext=5 PCT;
proc gchart data=piedata anno=logoanno;
pattern1 v=psolid c=CXFF0000;
pattern2 v=psolid c=CXFFFF00;
pattern3 v=psolid c=CX00FF00;
pattern4 v=psolid c=CX00FFFF;
pattern5 v=psolid c=CX0000FF;
pattern6 v=psolid c=CXFF00FF;
 /* titles and footnotes go here */
pie RGBcolor /
    sumvar=PieValue noheading coutline=CX000000
    midpoints='CXFF0000' 'CXFFFF00' 'CX00FF00'
              'CX00FFFF' 'CX0000FF' 'CXFF00FF'
    slice=outside value=none percent=none;
run;
quit;
ods html close;
```

Using Color for Communication

Color Does Not Improve Bad Design:
Use Color To Communicate, Not To Decorate

The pie chart below uses color to communicate. If you have no need to distinguish response levels or categories, use Black and White, or some other high-contrast color pair for foreground and background. If you have a few levels or categories, gray shades may suffice. If you have many levels or categories, color is necessary.



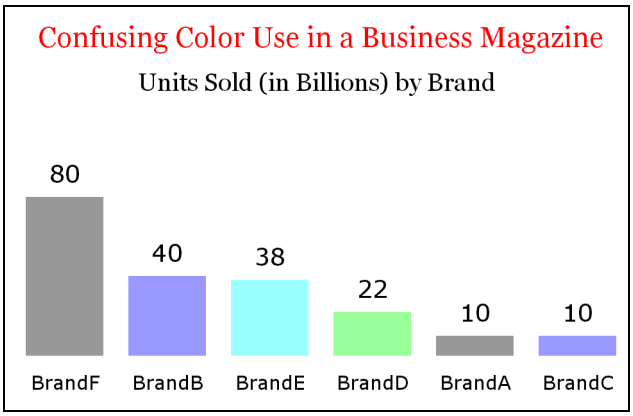
Tip: It is impossible to reliably distinguish more than five shades of a single hue. Of course, you may be able to expand the palette with Black and/or White, depending on the application. The proviso of "depending on the application" means, e.g., that you cannot use Black as an area fill on a map with Black boundaries.

Use of Color Can Confuse, Rather Than Communicate

Viewers attribute significance/meaning to your use of color, even when none is intended. So, be careful what you do, whenever you use color. Use of color without a design objective can disorient, confuse, or even mislead the viewer.

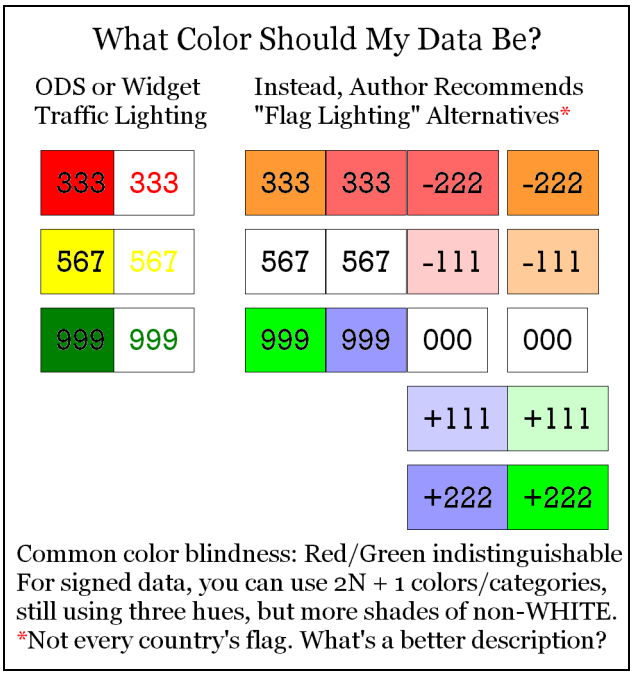
Failed communication is always the fault of the transmitter, not the receiver.

The content of the example below is actually different from the magazine illustration I saw, but its misuse of color is exactly parallel. There is NO relationship between BrandF and BrandA, and none between BrandB and BrandC. What does this use of color mean?



For Those Who Can't See a Color Difference, There Is None

The commonest color blindness cannot distinguish red and green, a frequently used color combination in the USA. Prof. Jay Neitz of the Eye Institute of the Medical College of Wisconsin: 8 to 10 percent of American males have some form of color blindness (due to genetic differences, only about 0.5 percent of females).



Maximize Color Contrast between Text and Background

Contrast between foreground and background is essential to communication. ODS opened the door to "enhancing" tables with color. Besides the unfortunately popular "Traffic Lighting", there are problems using Yellow with White, or Black (or other dark) text on dark or intense background colors. Evaluate the text-background combinations in the illustration above. See also the contrast demonstration chart in Reference 2. It should be noted that adequate contrast for online display does not guarantee the same for hardcopy, which is not brightly backlit.

Colored Text and Lines Should Be Thicker

Use of Black and White for print in newspapers, magazines, and books is no accident. Their high contrast makes them the most readable foreground-background combination. Colored text and lines on a White background are harder to see. Colored lines should be thickened. SAS/GRAPH enables this with the W= option for plot lines in the SYMBOL statement, with the SHAPE=LINE(width-number) option in the LEGEND statement, and with the WOUTLINE= option for the statements used with GCHART and GMAP PROCs. ODS lets you specify Bold for fonts, and SAS/GRAPH has bold versions of many of its own fonts, as well as allowing you to use Windows TrueType fonts with Bold (e.g., as in font='Georgia/Bold').

Use "Browser-Safe" or "Web-Safe" Colors

Unlikely as it may seem, many, if not most, web users have displays or video cards limited to 256 colors. Even when the hardware has a higher capability, it may be set to display only 256 colors. (To check or change your own setting on Windows, click Start > Settings > Control Panel > Display > Settings > Colors.) You may wish to design for the lowest common denominator. Here is why and how.

To deal with equipment diversity, web browsers determine the currently set limits of the display, and, if needed, remap colors. Video displays produce colors as combinations of Red, Green, and Blue. All web browsers agree on a universal common subset of 216 "browser-safe" colors. They are RGB colors (Red-Green-Blue combinations), with names, in SAS language, of the form CXrrggbb. The browser-safe RGB colors restrict rr, gg, and bb to the six values 00, 33, 66, 99, CC, and FF. (216 = 6 X 6 X 6.) If a web browser detects a color outside this set on a web page to be shown on a 256-color display, it remaps it to a browser-safe one. Then, Web Designer Color does not equal Web Viewer Color. There are 16.7 million RGB colors, but only 216 are browser-safe.

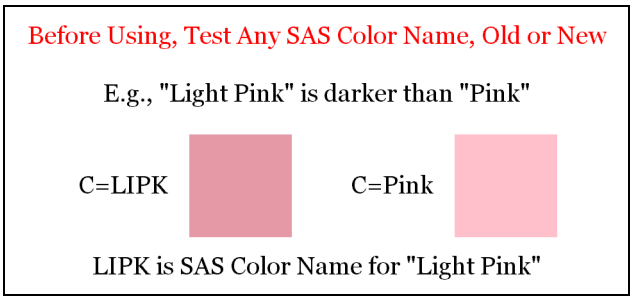
Both all of the SAS "predefined colors" (see below) and all of the HTML colors (see below) have RGB equivalents, but only (the same) seven of each are browser-safe. SAS GREEN, contrary to the RGB value still listed in the manual, changed in Version 6.12, and is no longer browser-safe—even though Green is one of the three RGB primaries.

See the Appendix for 81 samples of browser-safe colors. The "basic" colors are Red (CXFF0000), Yellow (CXFFFF00), Green (CX00FF00), Cyan (CX00FFFF), Blue (CX0000FF), Magenta (CXFF00FF), Black (CX000000), and White (CXFFFFFF). The upper chart shows the only way for RGB colors to vary in lightness with constant hue. It is easy to vary lightness with constant hue when using the HLS color system. If your target is hardcopy, HLS colors are an excellent choice, also providing easy tunability of transition in hue and saturation. See the SAS/GRAPH documentation for more about HLS colors.

Beware of SAS Color Names, Old and New

There are 292 "SAS Predefined Color Names", listed in Table 7.2 in the Version 6 and Version 8 SAS/GRAPH documentation. They have names such as "PINK", or "LIPK" for "Light Pink". However, many of the names are misleading. If you display or print PINK and LIPK, you will find that SAS Light Pink is darker than SAS Pink. See the illustration below. There are other contradictions like this. Also, many of the colors are too dark to be useful. Make color samples. See example code below.

There are also about 150 new color names in the SAS Color Registry. You can find them, and their RGB codes, using this click sequence in your SAS session: Solutions > Accessories > Registry Editor > Colornames > HTML. With these, too, assume nothing. Make yourself a sample chart. Only seven of these HTML color names intended for web use are browser-safe (i.e., web-safe).



Here is the code to create the color samples above:

```
proc gslide;
note j=C
f='Georgia'         h=1 c=CX000000 ' C=LIPK'
move=(+1,-1.5)
f='Monotype Sorts' h=5 c=LIPK     '6E'X
move=(+3,+1.5)
f='Georgia'         h=1 c=CX000000 'C=Pink'
move=(+1,-1.5)
f='Monotype Sorts' h=5 c=PINK     '6E'X;
run; quit;
```

If you cannot use these Windows TrueType fonts, use f=CENTX and replace '03'X with '6E'X.

Do You See What I See? Besides the use of browser-unsafe colors, and the problem of gamma differences between PC, Mac, and Unix, there are other technology-related sources of variation. CRT monitor color and LED flat panel color differ. And on an LED panel the lightest RGB colors wash out to near-White. LED projector color differs from that on a presenter's attached PC. CRT or LED color differs from printer color.

Acknowledgements

My thanks to Chris Noto for explaining how to imbed an image in a graph, and to all the people who have aided me in other ways.

Related Work By the Author

1. "Easy, Elegant, and Effective SAS Graphs: Inform and Influence with Your Data", elsewhere in these SUGI 28 Proceedings.
2. "Web Communication Effectiveness: Design and Methods to Get the Best Out of ODS, SAS, and SAS/GRAPH", in SUGI 28 Proceedings.
3. With Francesca Pierri, "Show Your Graphs and Tables at Their Best on the Web with ODS", *Proceedings of the Twenty-Seventh Annual SAS Users Group International Conference*, SAS Institute (Cary, NC), 2002.

Notices

SAS/GRAPH and SAS are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® denotes USA registration. Other product and brand names are trademarks or registered trademarks of their respective owners.

Author Information

LeRoy Bessler PhD
Bessler Consulting and Research
PO Box 96. Milwaukee, WI 53201-0096, USA
Phone: 1 414 351 6748
Email: bessler@execpc.com

LeRoy Bessler does general SAS application development, and communication-effective design and construction of reports, tables, graphs, and maps for the web and other media. He has expertise in Software-Intelligent application development, which yields solutions that are reliable, reusable, maintainable, and extendable.

Appendix: Samples of Browser-Safe SAS/GRAPH Colors, with Their RGB Codes

Shades of the RGB Primary & Secondary Colors, Grays, & White (CXFFFFFF)

No more than five shades of the same hue are easily distinguished

CX330000	CX333300	CX003300	CX003333	CX000033	CX330033	CXFFFFFF
CX660000	CX666600	CX006600	CX006666	CX000066	CX660066	
CX990000	CX999900	CX009900	CX009999	CX000099	CX990099	
CXCC0000	CXCCCC00	CX00CC00	CX00CCCC	CX0000CC	CXCC00CC	
CXFF0000	CXFFFF00	CX00FF00	CX00FFFF	CX0000FF	CXFF00FF	CX000000
CXFF3333	CXFFFF33	CX33FF33	CX33FFFF	CX3333FF	CXFF33FF	CX333333
CXFF6666	CXFFFF66	CX66FF66	CX66FFFF	CX6666FF	CXFF66FF	CX666666
CXFF9999	CXFFFF99	CX99FF99	CX99FFFF	CX9999FF	CXFF99FF	CX999999
CXFFCCCC	CXFFFFCC	CXCCFFCC	CXCCFFFF	CXCCCCFF	CXFFCCFF	CXCCCCCC

Other Browser-Safe SAS/GRAPH Colors

CX6633CC	CX9966FF	CX663300	CX996633	CXCC9966	CXFFCC99	CX9900CC
CX663333	CX996666	CXCC9999	CX999966	CXCCCC99	CXCC6600	CXFF9933
CX663366	CX996699	CXCC99CC	CX99CC00	CXCCFF33	CX336666	CX669999

bessler@execpc.com - 16Dec2002 LeRoy Bessler - The Power to Show