

**Communication-Effective Reporting with
Email/BlackBerry/iPhone, PowerPoint, Web/HTML, PDF, RTF/Word,
Animation, Images, Audio, Video, 3D, Hardcopy, or Excel**

LeRoy Bessler, Assurant Health, Milwaukee, Wisconsin, USA

Abstract

This presentation explores all the channels one can use to deliver information. It emphasizes communication effective design and construction. It is intended for new or experienced users of Version 9.1.3 or 9.2 of SAS®, SAS/GRAPH®, and ODS.

Introduction

The title and abstract above were written long before I sat down to write this paper.

To cover all of the channels promised, it was impossible to spend much time on communication-effective design. You can find my prior works on communication-effective design elsewhere. Here I have provided some of my design principles and an example of implementation of some key principles in an opening detailed mini-tutorial.

Also, although the code here will work in both versions of the software (any exceptions already known to me are noted), it focused on Version 9.2. There is not the systematic comparison of the two versions that I would have liked to deliver, but the code used should stand up for the immediate future.

Much, if not almost all, of what I take up here is material that you will not find covered by other SAS authors. This paper and presentation are drawn from knowledge and experience acquired during 29 years of using SAS/GRAPH and 9 years of using ODS.

For additional information about any topics that have been covered more expansively elsewhere, I provide references.

Code presented, in most cases, is complete. The level of commenting in it will vary.

I like to emphasize best practices for communication effectiveness, but for a few of the channels (e.g., audio and video) the scope is limited to showing you how to communicate using SAS for non-typical media, without concern for any subtleties as to what is the “best” way to do that.

Communication-Effective Design and Construction of Graphs

This preliminary mini-tutorial is in three parts: (a) some of my design principles; (b) a slide-based step-by-step demo of progressively implementing those principles for a specific type of chart; and (c) the composite program used to generate the end target design in one step.

The discussion of design actually applies to graphs created with any software.

The slide demo shows the specific effect of individual code changes.

The composite program pulls it all together.

Communication-Effective Design – A Too Short Introduction

Software defaults are designed to give you results, not necessarily the best results, or the results that you really want. Someone else's idea of what is communication-effective might not agree with your own standards of communication effectiveness, and, as implemented, might actually be counter to the intended objective.

The Power to Show™

Show them what is important. That is one of my personal aphorisms, which has been used more than once in the titles of some of my papers.

I believe in The Power of Simplicity™.

My guiding principle in design is elegance—

Everything that is needed, but only what is needed.

One thing that some people seem to fancy are background images, or textured backgrounds, or color gradient backgrounds. Or bar charts where the bars are composed of little images, rather than simple solid color bars. These techniques and the like confuse the possible with the necessary.

There is no benefit to area fill under a plot line. If the graph is a multi-line plot, one plot line's area fill has the potential to overlay part of one or more other plot lines.

Make all text and numbers readable: sufficient size with high contrast on a plain background.

Your graph should not be a vision test. Use big fonts wherever practical.

If you are using color, realize that if lines or characters are too thinly drawn, their color is hard to distinguish. Similarly, plot point markers must be sufficiently big. One of the communication defects in Excel is that the color swatches in legends are so small that some colors are hard to distinguish, which defeats the purpose of the legend. The sizes of the color swatches in Excel are not controllable. In SAS/GRAPH, everything is controllable. (We enjoy the mixed blessing of what I call "Options Over-choice".)

If your graph does not require color (e.g., to distinguish different plot lines, pie slices, or bars), the best color combination is black and white. For more about communication-effective color combinations or communication with color in other respects, see Reference 4, my most comprehensive paper on this topic.

If possible, make every graph title a headline—TELL viewers the implication or revelation, rather than only what variables are being graphed. However, this principle is not exemplified in this tutorial, which is more about construction techniques than about best graphic design.

Make a reasonable best effort to keep all of your text horizontal.

We read (English) text left-to-right, not bottom-to-top. I think that this slide image makes the point well:

Axis Labels Are Rarely Needed

- Put the info in graph title or subtitle
- Avoid use of vertical anti-readable label for the vertical axis—a common way try to save display space width is with vertical text
- We read left-to-right, not bottom-to-top

Is text at left OK?

Tell me the difference between it and vertical labels.

Also, I do not recommend vertical or tilted labels for horizontal axis tick marks.

Communication-Effective 3D Cylinder Chart: A Step-by-Step Slide Show

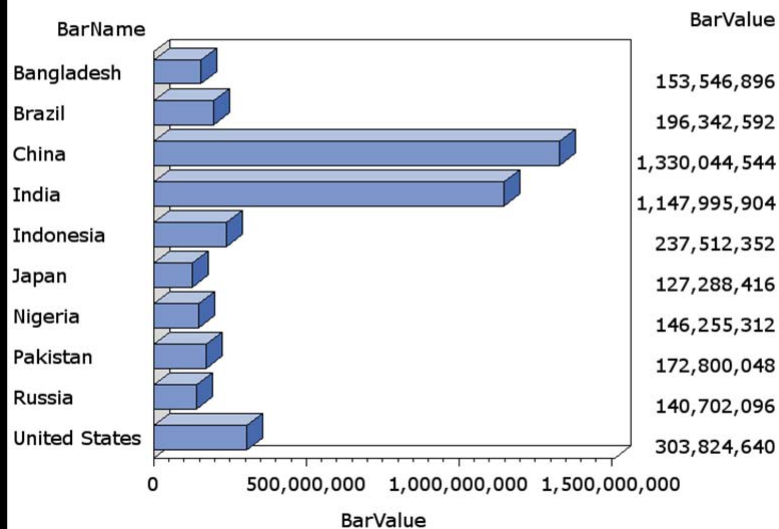
Where We Want To Finish

Top 10 Countries by Population



The Near-Default 3D Starting Point

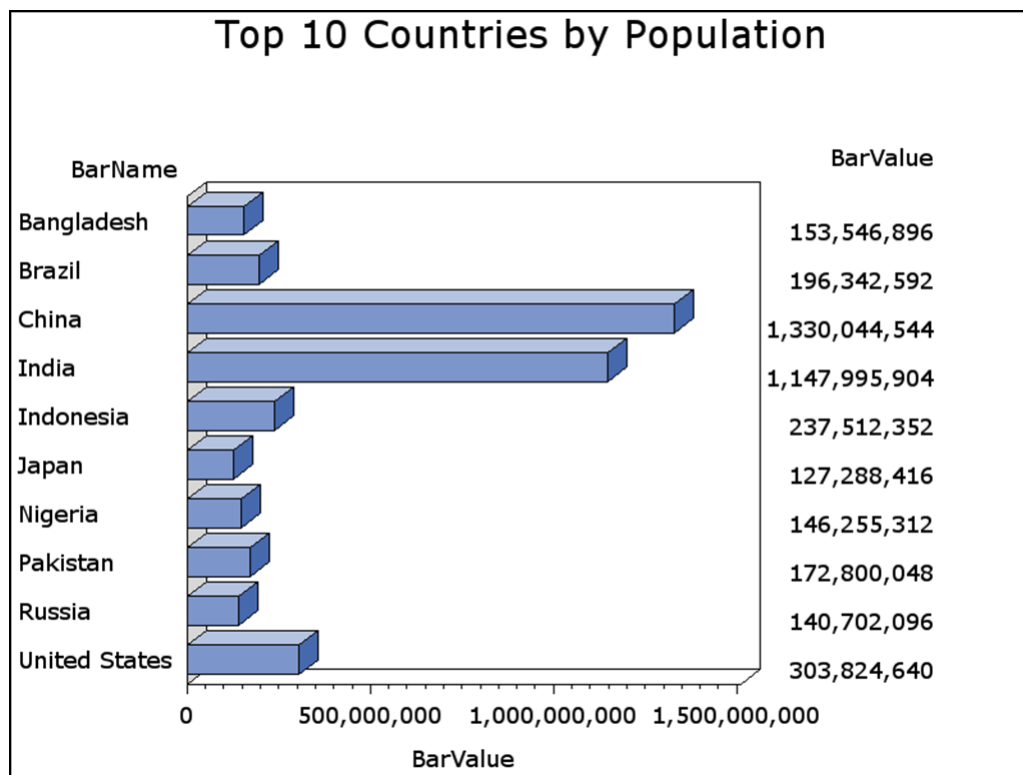
Top 10 Countries by Population

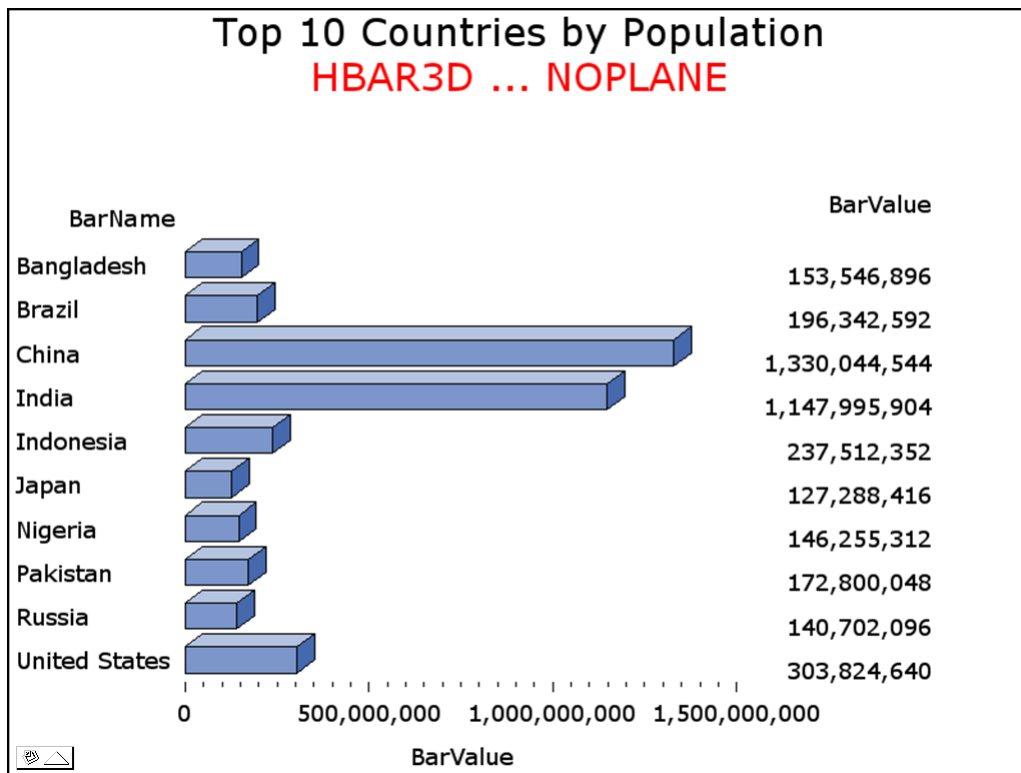
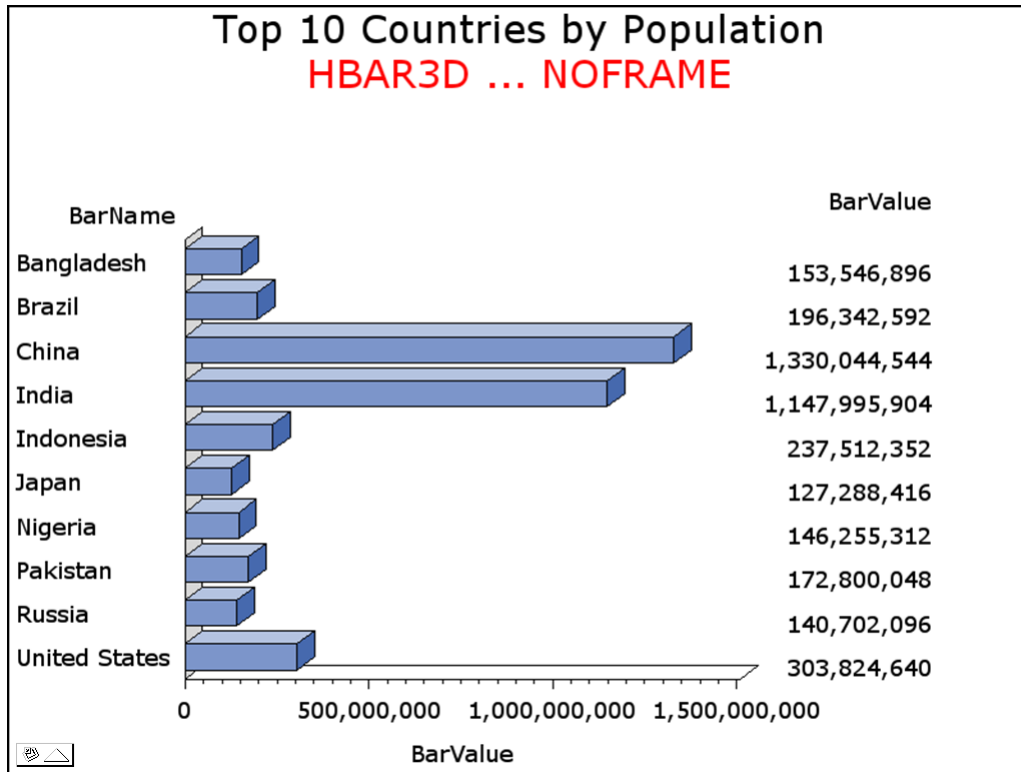


```

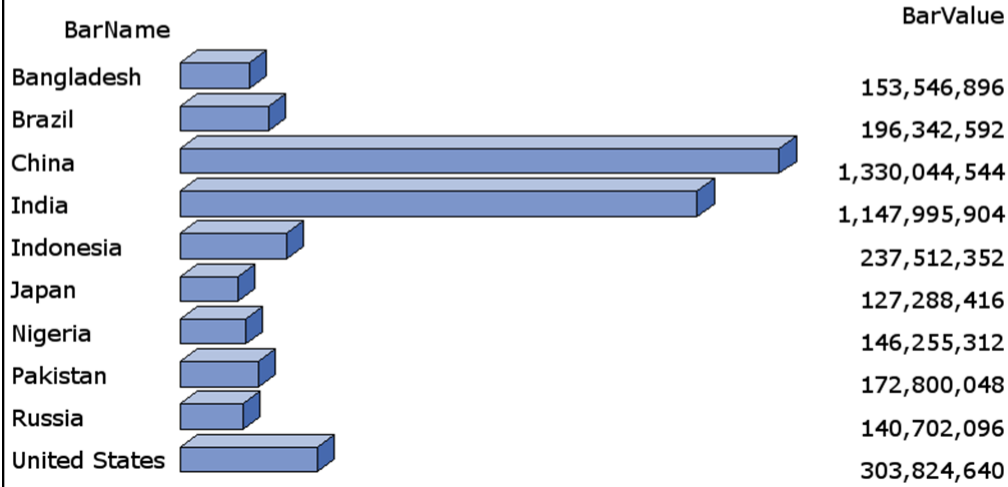
/* except for red code, this is a default 3d bar chart */
goptions reset=all;
goptions device=PNG;
goptions gsfmode=replace gsfname=anyname;
filename anyname "C:\FileName.PNG";
goptions border htext=3PCT ftext='Verdana';
title1 font='Verdana' height=5PCT
"Top 10 Countries by Population";
footnote;
proc gchart data=work.Top10;
hbar3d BarName / sumvar=BarValue;
format BarValue comma13.;
run; quit;

```

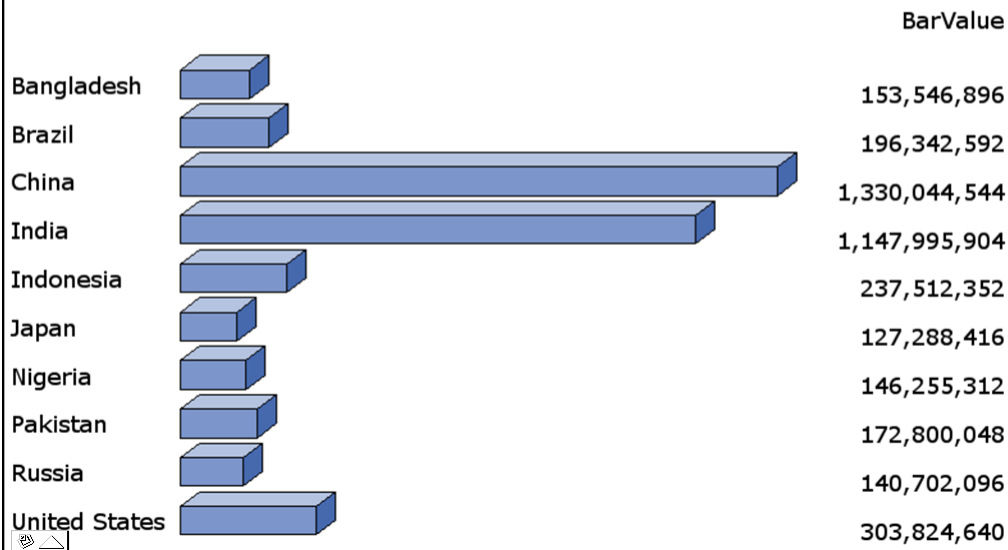




Top 10 Countries by Population
HBAR3D ... RAXIS=AXIS1;
AXIS1 LABEL=NONE VALUE=NONE
MAJOR=NONE MINOR=NONE;

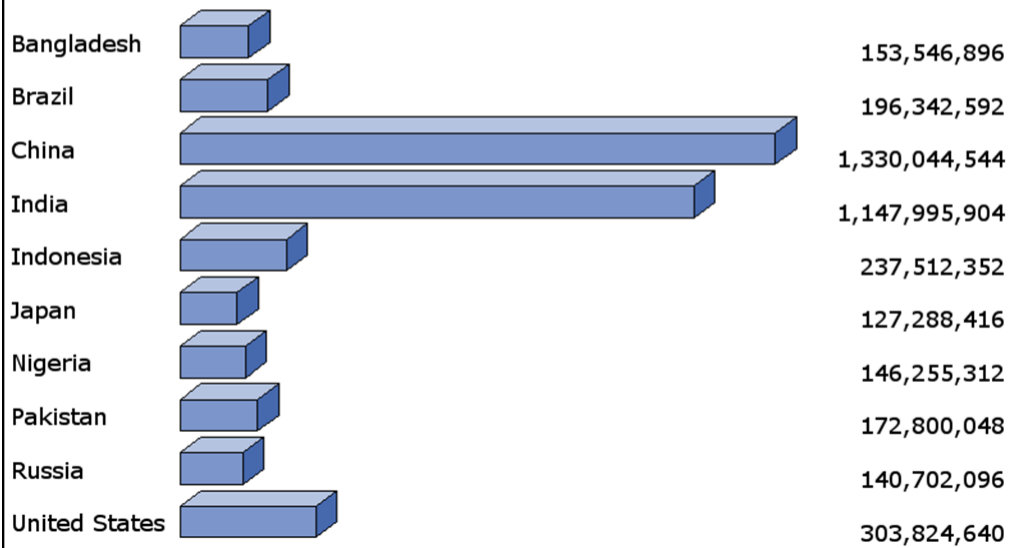


Top 10 Countries by Population
HBAR3D ... MAXIS=AXIS2;
AXIS2 LABEL=NONE;



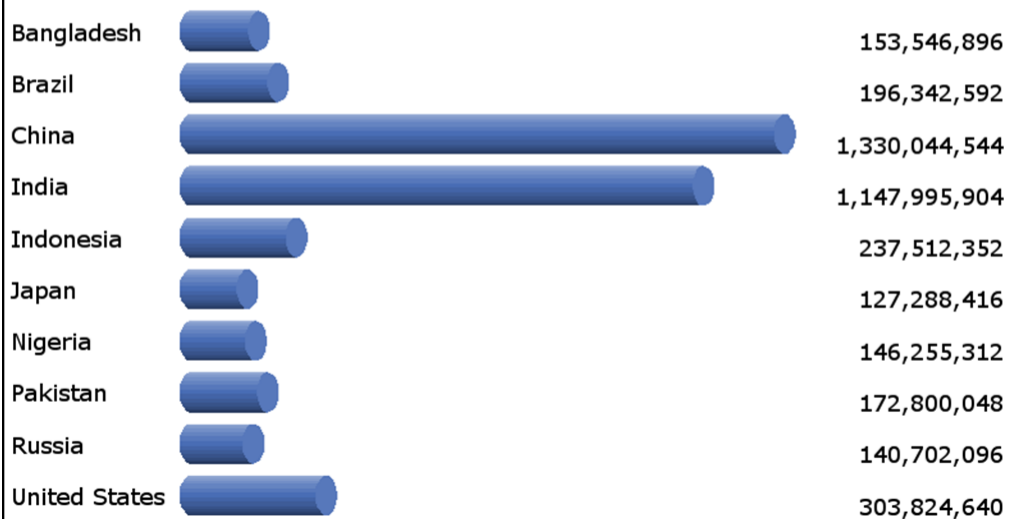
Top 10 Countries by Population

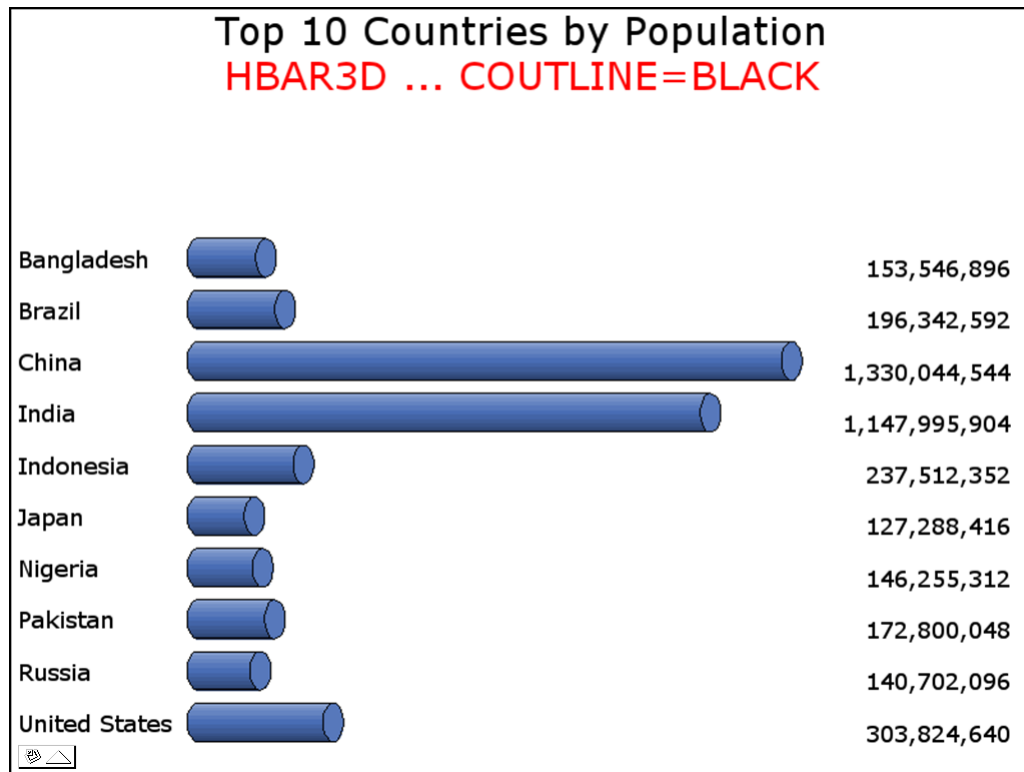
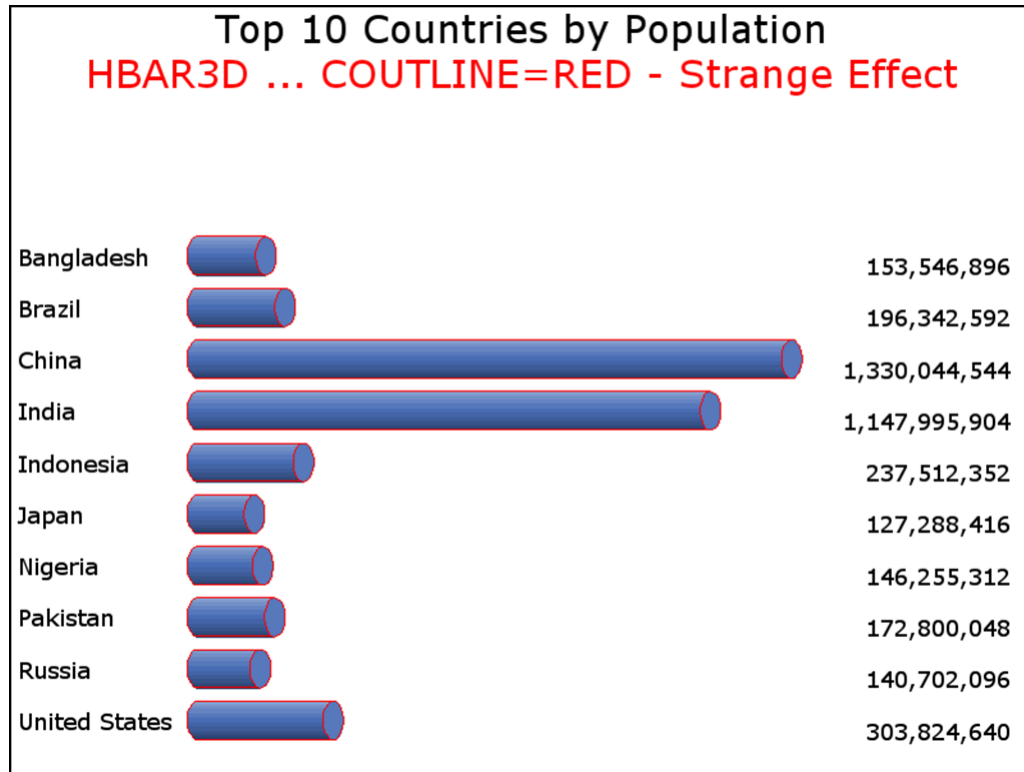
HBAR3D ... SUMLABEL=NONE



Top 10 Countries by Population

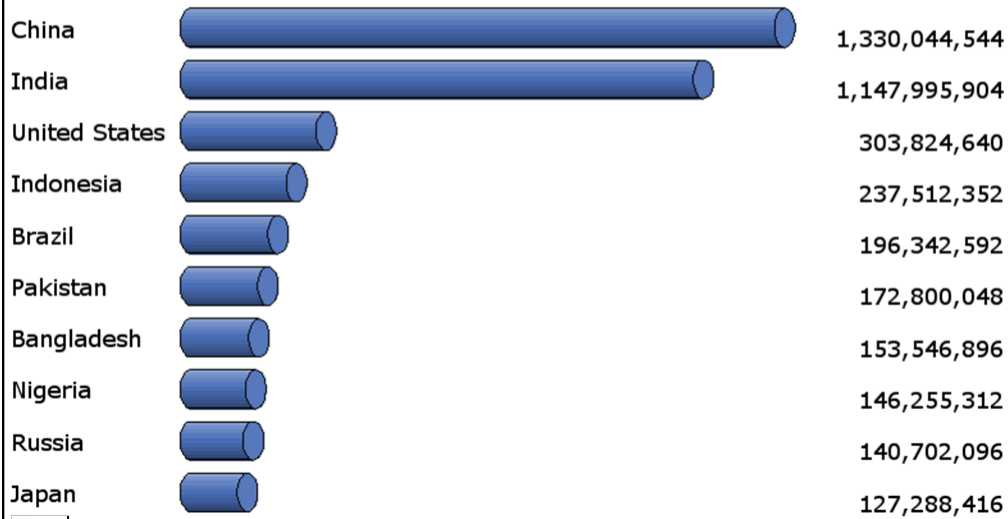
HBAR3D ... SHAPE=CYLINDER





Top 10 Countries by Population

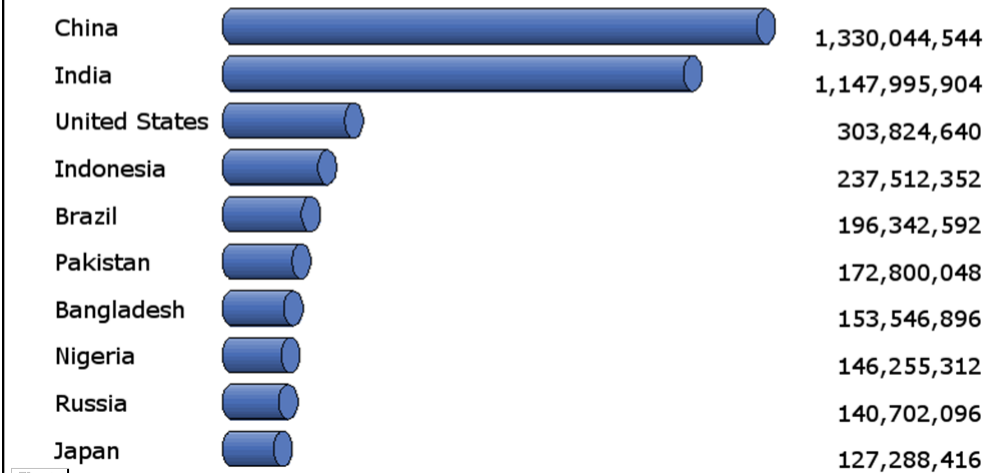
HBAR3D ... DESCENDING



Top 10 Countries by Population

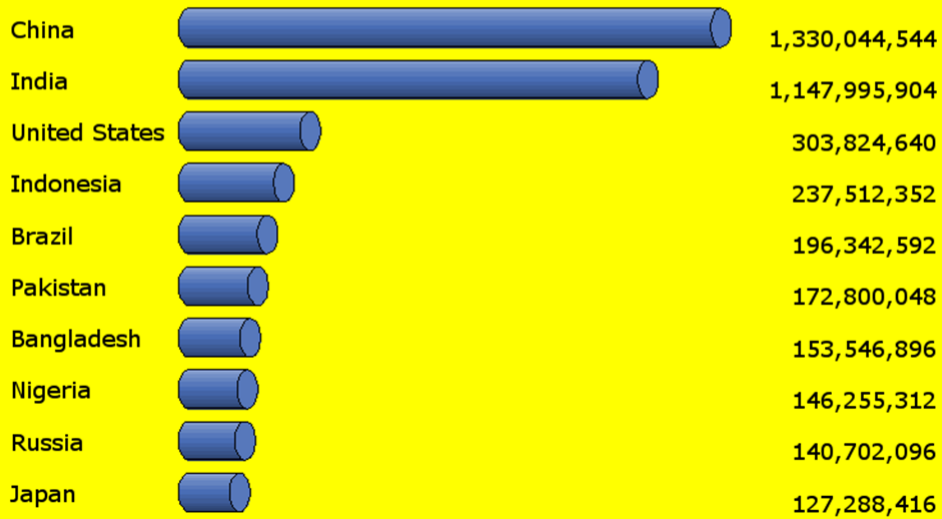
footnote1 angle=+90 height=2PCT ' ';

footnote2 angle=-90 height=5PCT ' ';



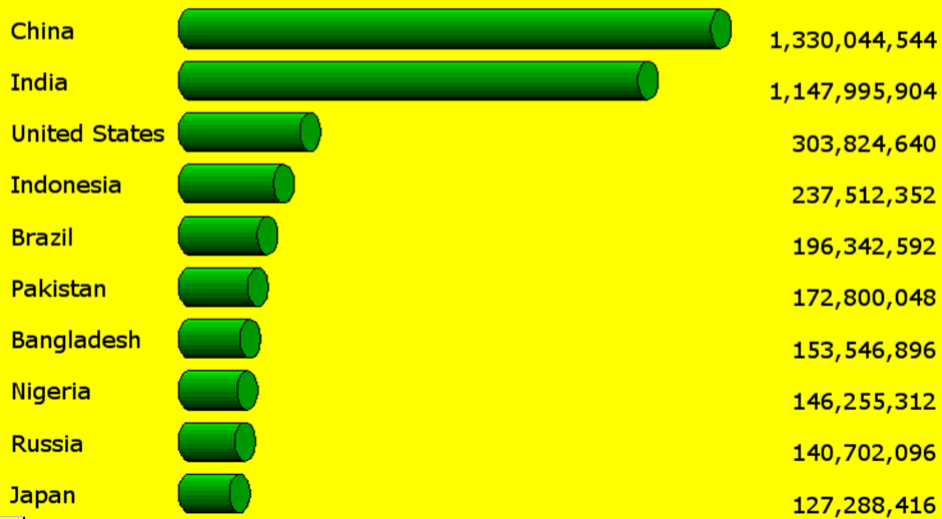
Top 10 Countries by Population

GOPTIONS CBACK=YELLOW;

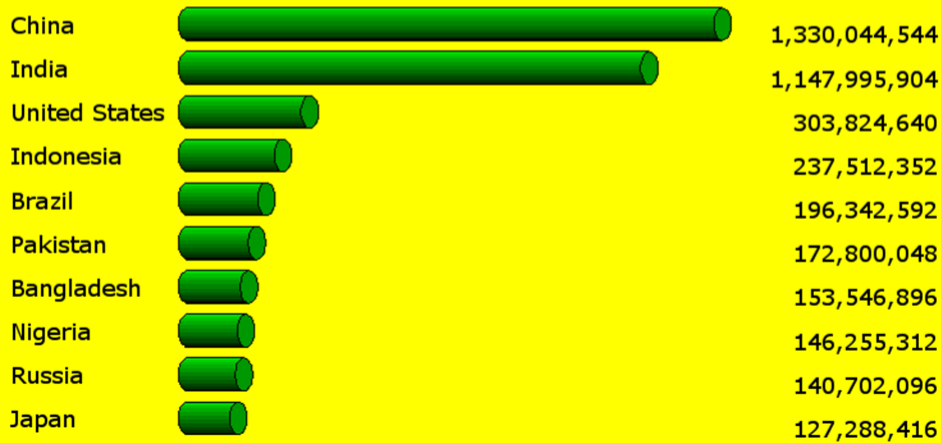


Top 10 Countries by Population

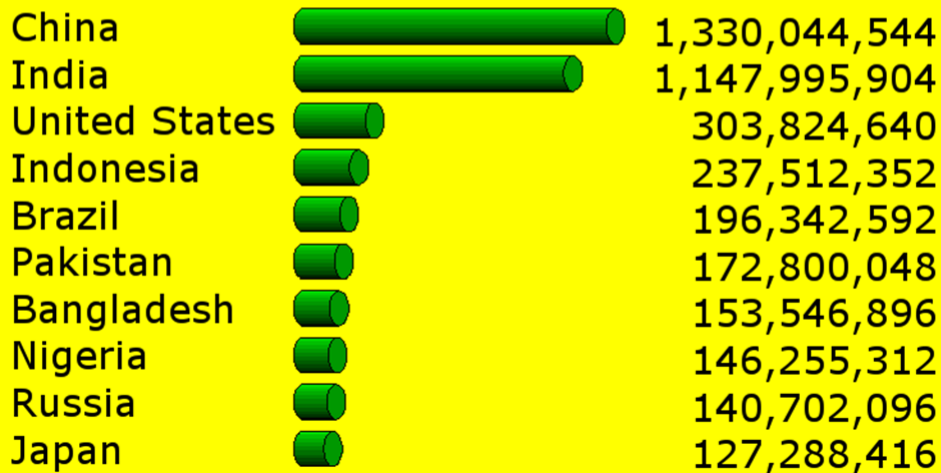
pattern1 value=solid color=CX006600;



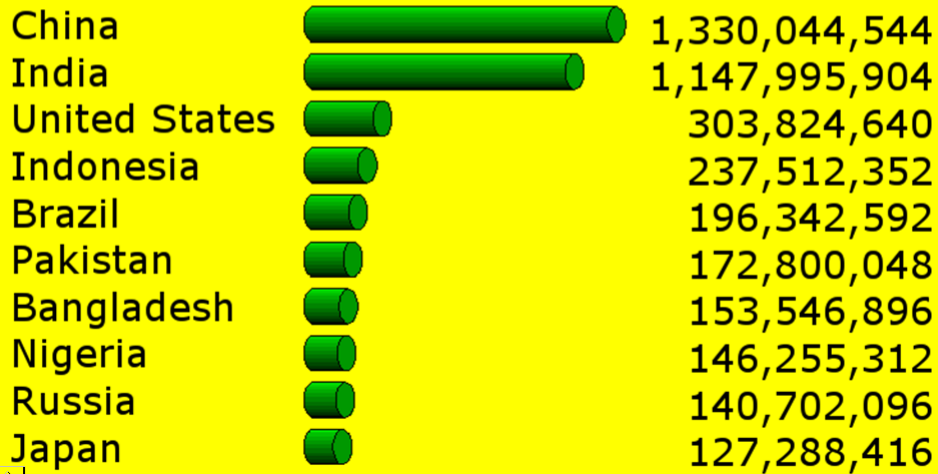
Top 10 Countries by Population
 title1 font="Verdana" height=5PCT " "
 justify=CENTER height=5PCT "Top 10 ..."



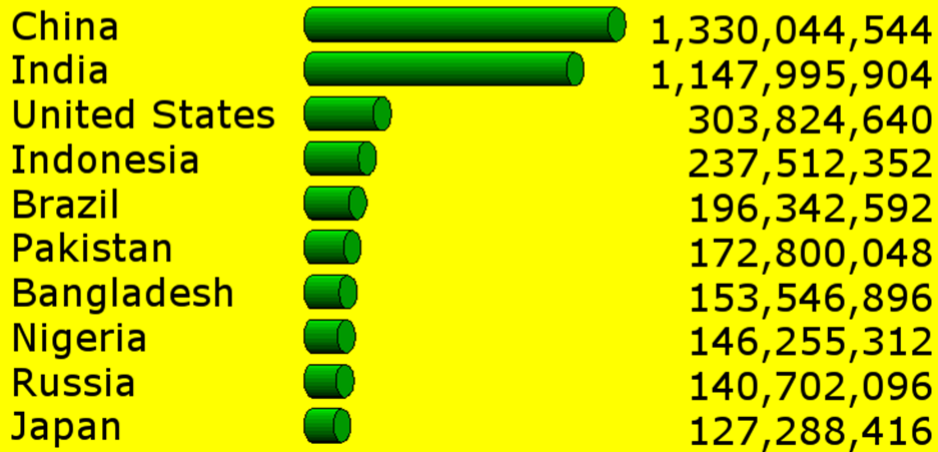
Top 10 Countries by Population
 GOPTIONS HTEXT=5PCT FTEXT="Verdana";



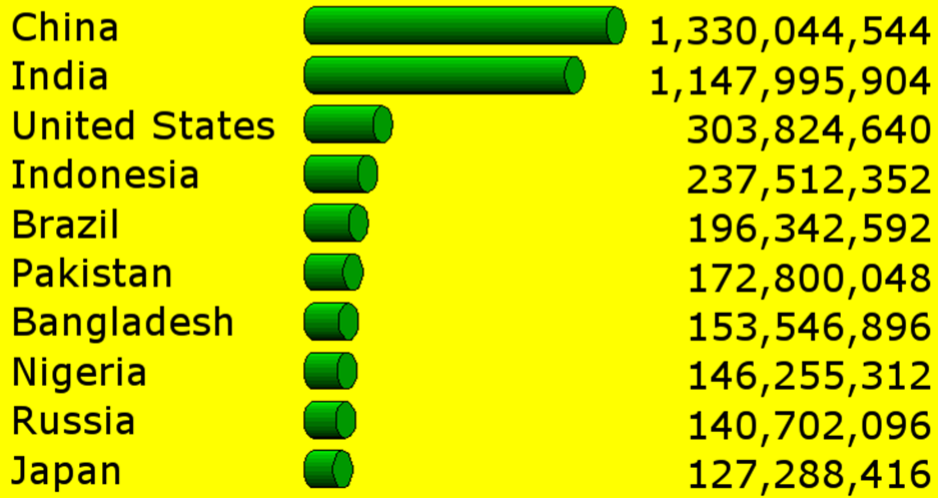
Top 10 Countries by Population
 AXIS1 ... OFFSET=(1 PCT, 0 PCT)



Top 10 Countries by Population
 FOOTNOTE3 HEIGHT=1PCT ' ';



Top 10 Countries by Population



See next page for an annotated consolidation of all of the steps.

Communication-Effective 3D Cylinder Chart: All of the Code Together

```
/* near-default starting code is in black */
/* communication-effective changes to near-default starting code are in red */
/* extra adjustments to near-default starting code are in blue */

goptions reset=all;
/* do this before every graph,
   unless you want to carry over settings from a prior graph */

goptions device=PNG;
goptions gsfname=replace gsfname=anyname;
filename anyname "C:\Folder\File.PNG";
goptions border;
/* invisible on black background of slides */
/* useful on .doc or .rtf or .pdf */
/* after insert to .doc, add Word's own border */

goptions htext=5PCT ftext="Verdana";

goptions cback=yellow;
pattern1 v=solid color=CX006600;
/* yellow & CX006600 is a communication-effective combination */
/* but the initial white & blue combination was also */

title1 font='Verdana' height=5PCT
" "
justify=CENTER
"Top 10 Countries by Population";
footnote1 angle=+90 height=2PCT ' ';
footnote2 angle=-90 height=5PCT ' ';
footnote3 height=1PCT ' ';
/* increase space around all four sides */

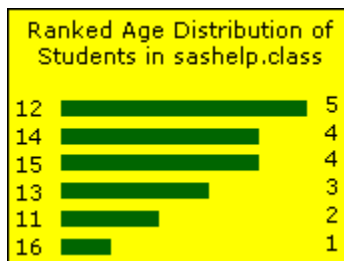
axis1 label=none value=none major=none minor=none offset=(1 PCT, 0 PCT);
axis2 label=none;
proc gchart data=work.Top10;
hbar3d BarName / sumvar=BarValue noframe noplane
  raxis=axis1 maxis=axis2
  sumlabel=none
  shape=cylinder coutline=black
  descending;
format BarValue comma13.;
run; quit;
```


Email / BlackBerry / iPhone— Extreme SAS/GRAPH

You might already know how to send email from SAS. The challenge addressed here is how to format a readable graph for a BlackBerry, an iPhone, or other very small screen communication device. This section is based on my previous work published in Reference 1.

I have long advocated for exploiting The Power of Simplicity in graphic design. The extreme conditions of the very small screen demand it. Here is a simple bar chart:

Actual Size:



Here is the code used to create and send the image above:

```
goptions reset=all;
goptions device=GIF;
goptions xpixels=172 ypixels=129; /* Default size for a GIF image is 800 X 600.
    Preserve 4-to-3 aspect ratio, but adjust the size for a BlackBerry or iPhone. */
goptions border; /* Put the graph in a box. */
goptions cback=CXFFFF00; /* Use RGB Yellow for background. */
goptions htext=10pt ftext='Verdana';
    /* Specify height and font used for those parts of the graph
       for which you do not make an explicit assignment,
       or for which no direct controls are available in SAS/GRAPH. */
goptions gsfmode=replace gsfname=anyname;
filename anyname "C:\FolderName\FileName.GIF";

title1 font='Verdana'
    color=CX000000 /* RGB black */
    height=3 PCT ' ' /* Insert space at the top. */
    height=10pt
    justify=CENTER /* Force a new line. */
    "Ranked Age Distribution of"
    justify=CENTER /* Force a new line. */
    "Students in sashelp.class";

footnote1 angle=+90 font=none height=1 ' ';
/* This "footnote" is a blank "right-side-note"
   to increase space at the side of the image. */
/* Use angle=-90 to create a "left-side-note" if needed. */

pattern1 v=solid color= CX006600; /* Use an RGB Medium Dark Green for the bars. */

/* Remove all axis paraphernalia: */
axis1 label=none major=none minor=none style=0;
axis2 label=none major=none minor=none style=0 value=none offset=(5 PCT,5 PCT);
/* Offset the horizontal axis to insert extra space between bar midpoint label
   and bar start, and between bar end and bar response value. */
```

```

proc gchart data=sashelp.class;
hbar Age /
  discrete      /* Use Ages, not Age subranges. */
  freq
  freqlabel=' ' /* Suppress the Frequency column heading. */
  descending
  maxis=axis1 raxis=axis2
  width=2.5 space=2.5; /* bar width and spacing */
run; quit;

filename anyname2 email
  to="Le_Roy_Bessler@wi.rr.com"
  subject="Graph for Your BlackBerry"
  attach="C:\FolderName\FileName.GIF";

data _null_;
file anyname2;
put 'Please see the attached graph.';
run;

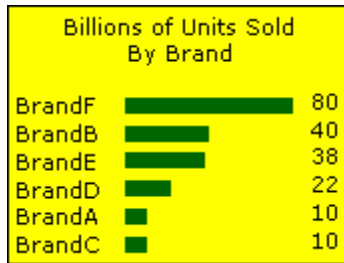
```

Your SAS program can send email to multiple addresses, with multiple attachments, with a CC and/or BCC, etc.

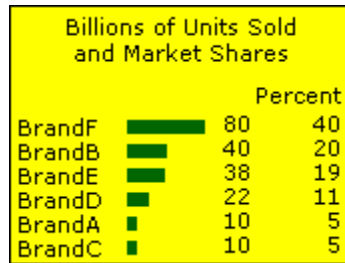
In the remainder of this section I present groups of related figures and comments, but none of the SAS code to produce them is presented here. Some of it is in Reference 1. Any code that is not in Reference 1 can be requested from the author.

Bar Charts and Pie Charts for the Small Screen (All examples are Actual Size)

Sum Bar Chart

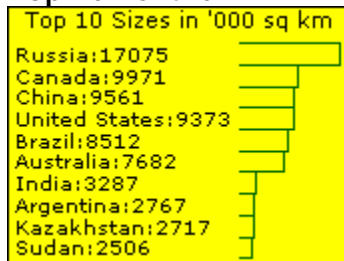


Pie Chart Alternative



To maximize the number of bars that can be displayed, remove the space between them and append the column of bar values to the bar names as shown below.

Top Ten Chart

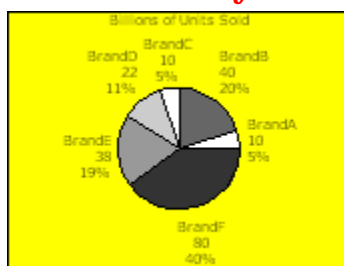


Maximum Capacity

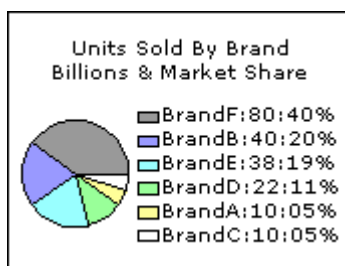


With 13 bars, the rows of text start to touch each other. The next smaller font is readable, but fuzzy shown below:

Text Too Fuzzy



Readable and Ordered

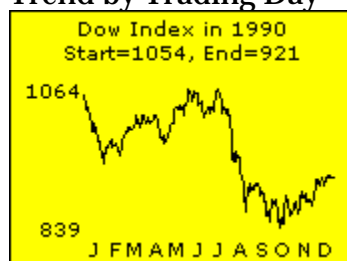


The text in the figure above at the left is too fuzzy, but the next larger font causes pie slice label overlap. I have long recommended ordering bars and pie slices by size to "Show the Viewer What Is Important." Even with the small fuzzy font, descending slice order would force slice label overlap. The custom legend in the example above at the right is the only very readable pie chart solution, and it can always support descending or ascending slice size. Since this example uses a color-coded legend, I prefer to use a white background. Amazingly, the fuzzy text in the example above at the left is harder to read on a white background than on a yellow background.

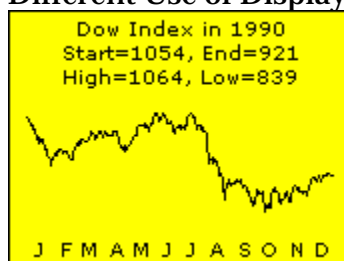
Trend Charts for the Small Screen (All examples are Actual Size)

For at least fifteen years, I have been making the case that a sparse trend chart is a sufficient trend chart. It enables easy and quick visual inferences. Precise numbers are best provided in a companion table. However, there are always four precise values of interest to a typical viewer of a trend chart: start, end, minimum, and maximum. There can be other points of interest on a trend chart, but those points are not amenable to automated programmatic detection and can be leveraged only by iterative ad hoc graph creation. I prefer a robust solution for hands-off, fully automated graph production applications. For iterative development of a graph for an ad hoc request, a robust solution reduces your turnaround time to meet the needs of the requestor.

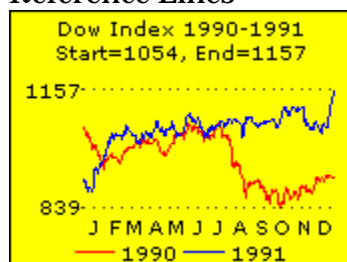
Trend by Trading Day



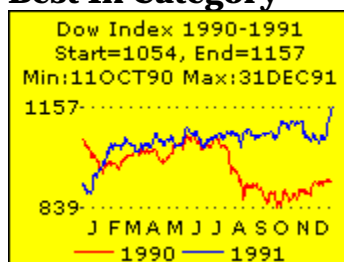
Different Use of Display Space



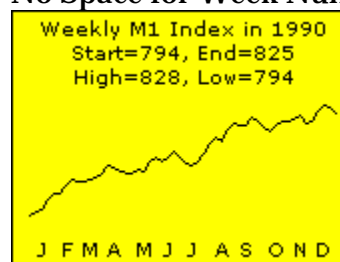
Reference Lines



Best In Category

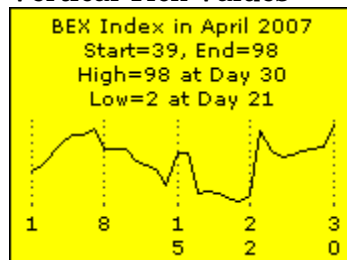


No Space for Week Numbers

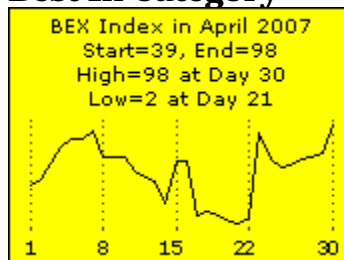


In all of these trend charts, there is not enough space on the horizontal axis to present many tick mark labels. The irregular spacing of the month name initials above is due to the programming algorithm used to locate them. Since there is an average of 22 trading days in each month, trading day 11 is used. Since the actual number of trading days varies by month, so does the resulting spacing of labels. For the weekly graph, where there are a variable number of weekly values in each month, week 3 is used, with irregular consequences for label spacing.

Vertical Tick Values



Best In Category



The stacked digits in the above example at the left are the SAS/GRAPH default action for this data. I do not recommend vertical or tilted horizontal axis values. English is read left-to-right, not vertically or tilted. The solution for daily tick marks in a month always displays only 1, 8, 15, 22, and the last day of the month. The trick used actually lays down the first digit of each two-digit day at the location of the day prior—to outwit SAS/GRAPH. The reference lines are used to overcome ambiguity. For a month of days, there are few enough points to be able to display plot symbols/markers, but I did not do so. For denser trends, they merge into a thick line.

If the trend chart design solutions above do not provide sufficient information, additional precise numbers can be provided in a companion text message or a companion table. The usability of a large table on the small screen is a challenge—scrolling is a sub-optimal and unwanted user experience even on the large screen.

PowerPoint

From SAS, it is easy to create a stand-alone PowerPoint slide that contains a graph (or a table).

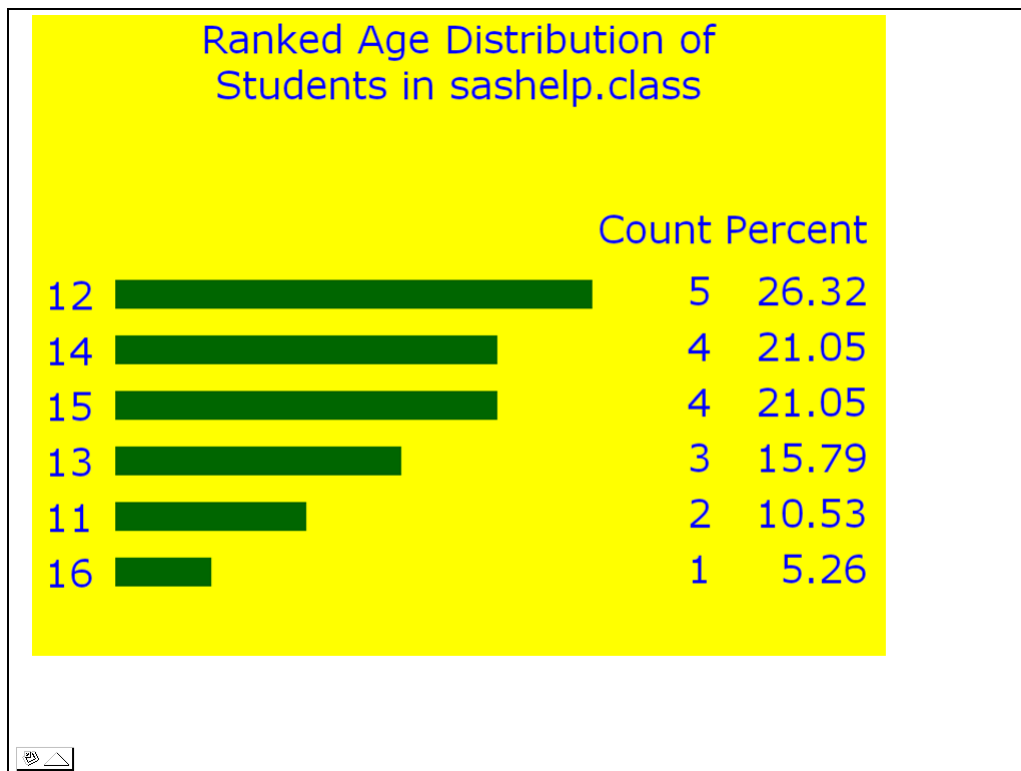
The file can be created with file extension ppt, in which case it opens in Edit mode, or it can be created with file extension pps, in which case it opens directly in full-screen display mode. If a collection of such distinct slide files is created, any one of them can be opened with PowerPoint in Edit mode, and the remaining slides can be manually inserted into that file in any desired order to create a traditional package of slides.

For how to create a package of scrollable slides directly, which is out of the scope of this section, please see Appendix 2 in Reference 2.

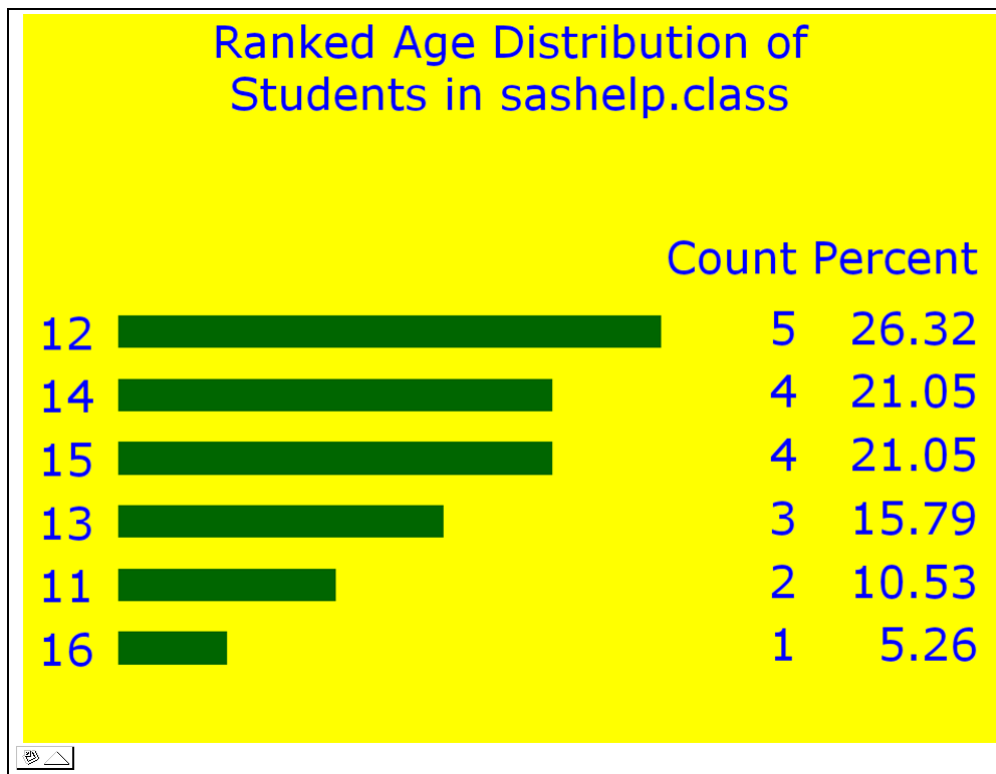
First, I will show how to create a slide for a graph, then how to create a slide for a table.

To see how to get an optimally sized and centered single disconnected slide, please compare the following three screen prints from PowerPoint display. (Code is presented at the end of this section.)

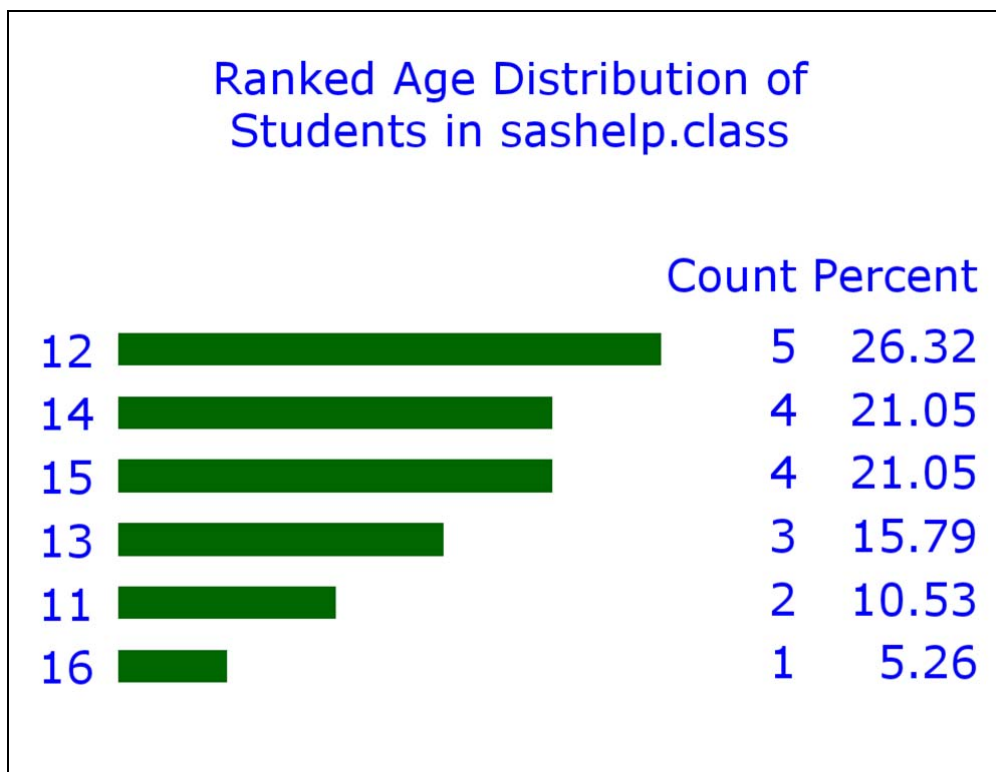
Default (a background is provided to clarify the graph size and position on the slide):



Created at maximum size that arrives centered on the slide:



Finished (yellow background removed and space added above title):



Here is the code for the three slides of graphs above:

```
* Begin set-up used for all three slides *;

goptions reset=all;
goptions device=PNG;
goptions htext=6 PCT ftext='Verdana' ctext=CX0000FF;
    /* Specify height, font, and color used for those parts of the graph
       for which you do not make an explicit assignment,
       or for which no direct controls are available in SAS/GRAPH. */
pattern1 v=solid color= CX006600; /* Use an RGB Medium Dark Green for the bars. */
    /* Remove all axis paraphernalia: */
axis1 label=none major=none minor=none style=0;
axis2 label=none major=none minor=none style=0 value=none
    offset=(2 PCT,0 PCT);
    /* Offset the horizontal axis to insert extra space between bar midpoint label
       and bar start, but not between bar end and bar response value. */

proc catalog cat=work.gseg kill; run; quit;

ods noresults;
ods listing close;

* End set-up used for all three slides    *;

* Begin set-up used for first two slides *;

goptions cback=CXFFFF00; /* RGB yellow background */

title1 font='Verdana'
    height=6 PCT
    color=CX0000FF    /* RGB blue */
    "Ranked Age Distribution of"
    justify=CENTER    /* Force a new line. */
    "Students in sashelp.class";

* End set-up used for first two slides    *;

ods html path="C:\MWSUG 2009\Paper 2\9.2"
    (url=none) /* slide file and its graph can be relocated */
    body="DefaultSlideFromSAS.ppt"
    style=Styles.Minimal;

proc gchart data=sashelp.class;
hbar Age / name='Default'
    discrete /* Use Ages, not Age subranges. */
    freq
    freqlabel='Count' /* Suppress the Frequency column heading. */
    percent
    percentlabel='Percent'
    descending maxis=axis1 raxis=axis2 width=2 space=2
    coutline=CX006600; /* override default bar outline color which is black */
run; quit;

ods html close;

ods html path="C:\MWSUG 2009\Paper 2\9.2"
    (url=none) /* slide file and its graph can be relocated */
    body="CenteredAndRightSizedSlideFromSAS.ppt"
    style=Styles.Minimal;

goptions xpixels=932 ypixels=699; /* right-size and center it */
```



```

proc gchart data=sashelp.class;
hbar Age / name='RightSiz'
  discrete /* Use Ages, not Age subranges. */
  freq
  freqlabel='Count' /* Suppress the Frequency column heading. */
  percent
  percentlabel='Percent'
  descending maxis=axis1 raxis=axis2 width=2 space=2
  coutline=CX006600; /* override default bar outline color which is black */
run; quit;

ods html close;

options cback=white; /* turn off colored background */

title1 font='Verdana'

  /* Begin code to insert space at the top */
  height=4 PCT ' ' /* Make an empty line. */
  justify=CENTER /* Force a new line. */
  /* End code to insert space at the top */

  height=6 PCT
  color=CX0000FF /* RGB blue */
  "Ranked Age Distribution of"
  justify=CENTER /* Force a new line. */
  "Students in sashelp.class";

ods html path="C:\MWSUG 2009\Paper 2\9.2"
  (url=none) /* slide file and its graph can be relocated */
  body="FinishedSlideFromSAS.ppt"
  style=Styles.Minimal;

options xpixels=932 ypixels=699; /* right-size and center it */

proc gchart data=sashelp.class;
hbar Age / name='Finished'
  discrete /* Use Ages, not Age subranges. */
  freq
  freqlabel='Count' /* Suppress the Frequency column heading. */
  percent
  percentlabel='Percent'
  descending maxis=axis1 raxis=axis2 width=2 space=2
  coutline=CX006600; /* override default bar outline color which is black */
run; quit;

ods html close;

options reset=all;
ods html path="C:\MWSUG 2009\Paper 2\9.2"
  /* (url=none) is not needed because only one file is created,
  with no possible relocatability concern */
  body="NearlyDefaultTableInSlideFromSAS.ppt"
  style=Styles.Minimal;

title font='Verdana' height=24 pt "Students in sashelp.class";
proc print data=sashelp.class;
run;

ods html close;

ods listing;

```

Slide of a Table with Customized Title

Students in sashelp.class

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83.0
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84.0
10	John	M	12	59.0	99.5
11	Joyce	F	11	51.3	50.5
12	Judy	F	14	64.3	90.0
13	Louise	F	12	56.3	77.0
14	Mary	F	15	66.5	112.0
15	Philip	M	16	72.0	150.0
16	Robert	M	12	64.8	128.0
17	Ronald	M	15	67.0	133.0
18	Thomas	M	11	57.5	85.0
19	William	M	15	66.5	112.0

Here is the code to create the slide above:

```
goptions reset=all;

ods html path="C:\MWSUG 2009\Paper 2\9.2"
/* (url=none) is not needed because only one file is created,
   with no possible relocatability concern */
body="NearlyDefaultTableInSlideFromSAS.ppt"
style=Styles.Minimal;

title font='Verdana' height=24 pt "Students in sashelp.class";
proc print data=sashelp.class;
run;

ods html close;
```

Web / HTML

This was the first destination to be supported by ODS, and has been extensively documented already. You might find something of interest in Reference 3, but here I will focus on just a few key matters: use of TITLE and METATEXT to make your web page easy to find; relocatability of web pages and any of their accessory files; zoomability of a web page; navigation through a collection of web pages; browser-safe colors; and right-sizing a graph for web page display (i.e., TRYING to assure that it will be viewable in full without scrolling). For a discussion of ALT text, please see the section on Images.

Getting Your Web Page Found

Some web sites have a captive audience. It might be a web site on a corporate, university, or other organizational intranet. It might be a web site that serves a purchaser/supplier or other business-to-business electronic commerce relationship.

But not all web applications have a captive audience.

An easy thing you can do to get discovered on the web is to use the TITLE= option on the BODY= parameter (which identifies your web page body file). Here's the code:

```
ods html body="yourpagename.html" (title="this text identifies your content");
```

Here are various uses of the TITLE text, where the first one makes it easier to find your web page the first time, and where the second and third make it easier to find it again:

1. captured by search engines
2. default text for Internet Explorer Favorite
3. web page browse History list entry
4. title bar for the browser window

If the web page uses frames (e.g., an ODS Table-of-Contents-based presentation of information), then it is the TITLE= assignment for the frame file that appears in the browser window title bar, and presumably serves the other uses listed above.

Search engines look at META tags—use them. Here is ODS code to use META tags:

```
ods html . . .  
  metatext='name="keywords" contents="word1, word2, ..."';
```

There are a huge number of possible kinds of META tags, identified by different assignments to the NAME= parameter. (If needing to use METATEXT= for web control functions, the functional parameters are assigned with CONTENTS= and the type of control is identified with HTTP-EQUIV= rather than NAME=.)

A source for more information about META tags is

<http://vancouver-webpages.com/META/metatags.detail.html>

If you go to this web site, not only can you explore the documentation it provides about META tags, but also you should click VIEW and then SOURCE to look at how the web site uses META tags for its own web pages. Despite the richness of possible uses for META tags, ODS still supports only a maximum of 255 characters for your entire METATEXT string. HTML takes an unlimited number of META tags. (Some of the possibilities are: keywords, description, author, copyright, and publisher.) If you are using ODS to create web pages for the external worldwide web, and have a serious interest in getting your web pages found, you can post-process the ODS output to insert more than 255 characters of META text.

Relocatability

Relocatability means that your web pages and anything to which they are connected can be picked up as a unit and copied or moved to another location, such as a web server or a remote folder on the LAN, or can be zipped up and emailed to someone else, who can unzip the package and store it anywhere convenient. Without relocatable design, connections from one web page to another will not work, and pointers to images will not deliver the images to the web page display screen. Relocatability relies on using (URL=NONE) on the ODS HTML statement and on defining the path to other web pages or to anything else referenced by a web page (e.g., image files) in terms of relationship to where the web page is located. Below are five coding examples. All use (URL=NONE), but they differ in where the image file is stored versus where the web page is stored.

```
goptions reset=all;
title 'Pre-Teen Students';

ods html path='C:\MWSUG 2009 Web Demo\9.2\WebPages' (url=none)
      body='ImageInFolderAtSameLevelAsThisWebPagesFolder.html';
proc print data=sashelp.class(where=(age LT 13)) noobs
      style = [posthtml="<img src='../Images/sasanim_ParallelFolder.gif'
                        alt='Powered by SAS' vspace=10>"]; run;
ods html close;

ods html path='C:\MWSUG 2009 Web Demo\9.2\WebPages' (url=none)
      body='ImageIsInFolderUpOneLevelFromThisWebPagesFolder.html';
proc print data=sashelp.class(where=(age LT 13)) noobs
      style = [posthtml="<img src='../sasanim_FolderUpOneLevel.gif'
                        alt='Powered by SAS' vspace=10>"]; run;
ods html close;

ods html path='C:\MWSUG 2009 Web Demo\9.2\WebPages' (url=none)
      body='ImageIsInFolderUpTwoLevelsFromThisWebPagesFolder.html';
proc print data=sashelp.class(where=(age LT 13)) noobs
      style = [posthtml="<img src='../../sasanim_FolderUpTwoLevels.gif'
                        alt='Powered by SAS' vspace=10>"]; run;
ods html close;

ods html path='C:\MWSUG 2009 Web Demo\9.2\WebPages' (url=none)
      body='ImageIsInThisWebPagesFolder.html';
proc print data=sashelp.class(where=(age LT 13)) noobs
      style = [posthtml="<img src='sasanim_ThisFolder.gif'
                        alt='Powered by SAS' vspace=10>"]; run;
ods html close;

ods html path='C:\MWSUG 2009 Web Demo\9.2\WebPages' (url=none)
      body='ImageIsInSubfolderOfThisWebPagesFolder.html';
proc print data=sashelp.class(where=(age LT 13)) noobs
      style = [posthtml="<img src='Images/sasanim_Subfolder.gif'
                        alt='Powered by SAS' vspace=10>"]; run;
ods html close;
```

Notes:

VSPACE=10 inserts ten pixels of space above (and below) the image.

ALT= provides pop-up text when the viewer rests the mouse on the image.

If there were a footnote, it would appear below the image.

Using PREHTML= would put the image between the title and the table.

Zoomability

Suppose you have a graph with a lot of detail, but you want the default presentation of it to fit within the available live space of the web page window. I.e., you want the viewer to see the total picture. Scrolling a large image only presents it to the viewer in pieces. Nevertheless, you would presumably want the viewer to be able to see the detail. A large map inlaid with information in each unit geographic area would be one case. A case that occurred for me was a horizontal bar chart with many bars. Rather than show you the solution with a graph, let me present it with minimal coding and just use a table, but one where I force a small font size.

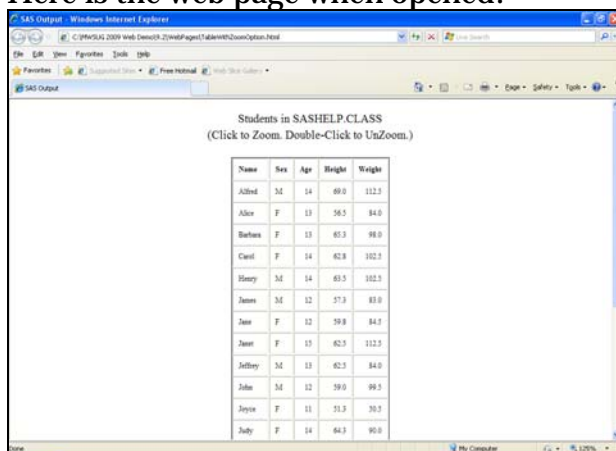
NOTE: When the web page is opened, the viewer might be prompted with this: “To help protect your security, Internet Explorer has restricted this webpage from running scripts or ActiveX controls that could access your computer. Click here for options . . .” To enable the Zoom, the viewer will need to select the option to “Allow blocked content”. It might be useful to add another subtitle to explain that.

Here is the code (where the zoom percents can be whatever you prefer to code):

```
proc template;
edit styles.Minimal as styles.MinimalZoomable;
style body / tagattr="onclick=this.style.zoom='200%'
                  ondblclick=this.style.zoom='100%'";
end;
run;

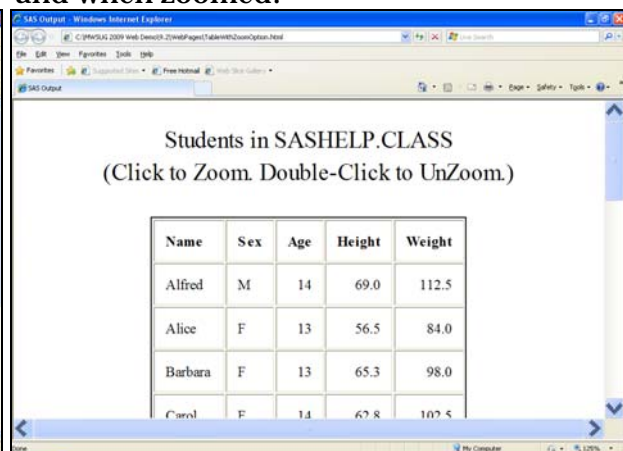
ods html path='C:\MWSUG 2009 Web Demo\9.2\WebPages' (url=none)
        body='TableWithZoomOption.html'
        style=Styles.MinimalZoomable;
title 'Students in SASHELP.CLASS';
title2 '(Click to Zoom. Double-Click to UnZoom.)';
proc print data=sashelp.class noobs;
var name sex age height weight /
    style=[font_size=1]; /* Use smallest web font from sizes 1 to 7 */
run;
ods html close;
```

Here is the web page when opened:



Name	Sex	Age	Height	Weight
Alfred	M	14	69.0	112.5
Alice	F	13	56.5	84.0
Barbara	F	13	65.3	98.0
Carol	F	14	67.8	102.5
Henry	M	14	63.5	102.5
James	M	12	57.3	85.0
Jan	F	12	59.8	84.5
Janet	F	13	62.5	112.5
Jeffrey	M	13	62.5	84.0
John	M	12	59.0	99.5
Joyce	F	11	51.3	30.5
Judy	F	14	64.3	90.0

and when zoomed:



Name	Sex	Age	Height	Weight
Alfred	M	14	69.0	112.5
Alice	F	13	56.5	84.0
Barbara	F	13	65.3	98.0
Carol	F	14	67.8	102.5

Navigation

These are tips about design, not SAS coding.

If you build a hierarchical network of web pages: (a) provide a link to a home page on every web page; and (b) provide cross-links between web pages that are related so that the user can toggle between different aspects of the same topic of interest. Item (a) assures that one does not have to climb back up a chain of web pages step-by-step every time to get to the home page. Item (b) allows the viewer, in some situations, to jump directly to a target web page of interest, without using the hierarchical network.

Avoid creating a huge long web page with links to other points within itself. It creates the false impression of a collection of distinct web pages, and when the viewer seeks to print a discrete copy of what she/he is looking at, the result is annoying stack of paper which has what is really wanted buried within it, possibly spanning a page break.

Browser-Safe Colors

Ideally, no one should have their PC configured to display only a maximum of 255 colors, but some people do.

(I have no idea what the color capacity is of the new small and tiny screen devices.)

If you are designing and creating color web pages, you presumably would like the viewer to see the same color that you saw during development and testing.

To maximize the probability that a color web page can look the same to all viewers, there is an agreed standard set of colors that is reproducible as intended on any screen with a capacity of 255 colors. These colors are called “browser-safe” or “web-safe”.

They are defined in SAS by using color names of the form CXRRGGBB, where the strings RR, GG, and BB are limited to the six values 00, 33, 66, 99, CC, FF. This provides $6 \times 6 \times 6 = 216$ colors, which is a palette sufficient to meet the communication needs of any business or statistical graphic application.

For samples of browser-safe colors, and for other important aspects of communicating with color in any context, not just on the web, please see Reference 4.

Right-Sizing a Graph for the Web: Viewers DO NOT Really Want to Scroll

This concern also applies to tables, but if the table has too many rows, scrolling is inevitable. For a graph or a table, having to scroll sideways is even more irritating than having to scroll up and down.

As a working hypothesis, despite the arrival of those wide-screen monitors designed for watching movies, you should assume that most of the likely viewers of your web page will have their monitor's screen resolution set to 1024 X 768. That also happens to be the assumption of LED projectors, and happens to have the 4 X 3 proportions of any non-movie projection screen. The size of a graph is defined by its xpixels and ypixels. The default size of graphs created by the GIF and PNG device drivers is 800 X 600. Unfortunately, the available pixels on a web page viewer's screen are also consumed by the basic framing of Windows, plus the framing required by the web browser, plus possible optional toolbars (e.g., for Google or Yahoo or . . .). The remaining space is referred to as the "live space" in your web page window. Furthermore, the HTML code that presents the graph on the web page does not place it completely adjacent to the top and left-hand side of the space that is nominally available after all of the subtractions noted above. Please see my previously published work on right-sizing graphs for a web page in Reference 2.

Remember, your web page viewers do not want to scroll, even if it is more convenient to develop web pages with total disregard of that fact. Design for the most likely minimum live space, based on your expected audience and on what you know about their web browser, if anything. It is impractical to design for the worst possible minimum live space unless there is a reasonable likelihood of its being that of some viewers of your web page.

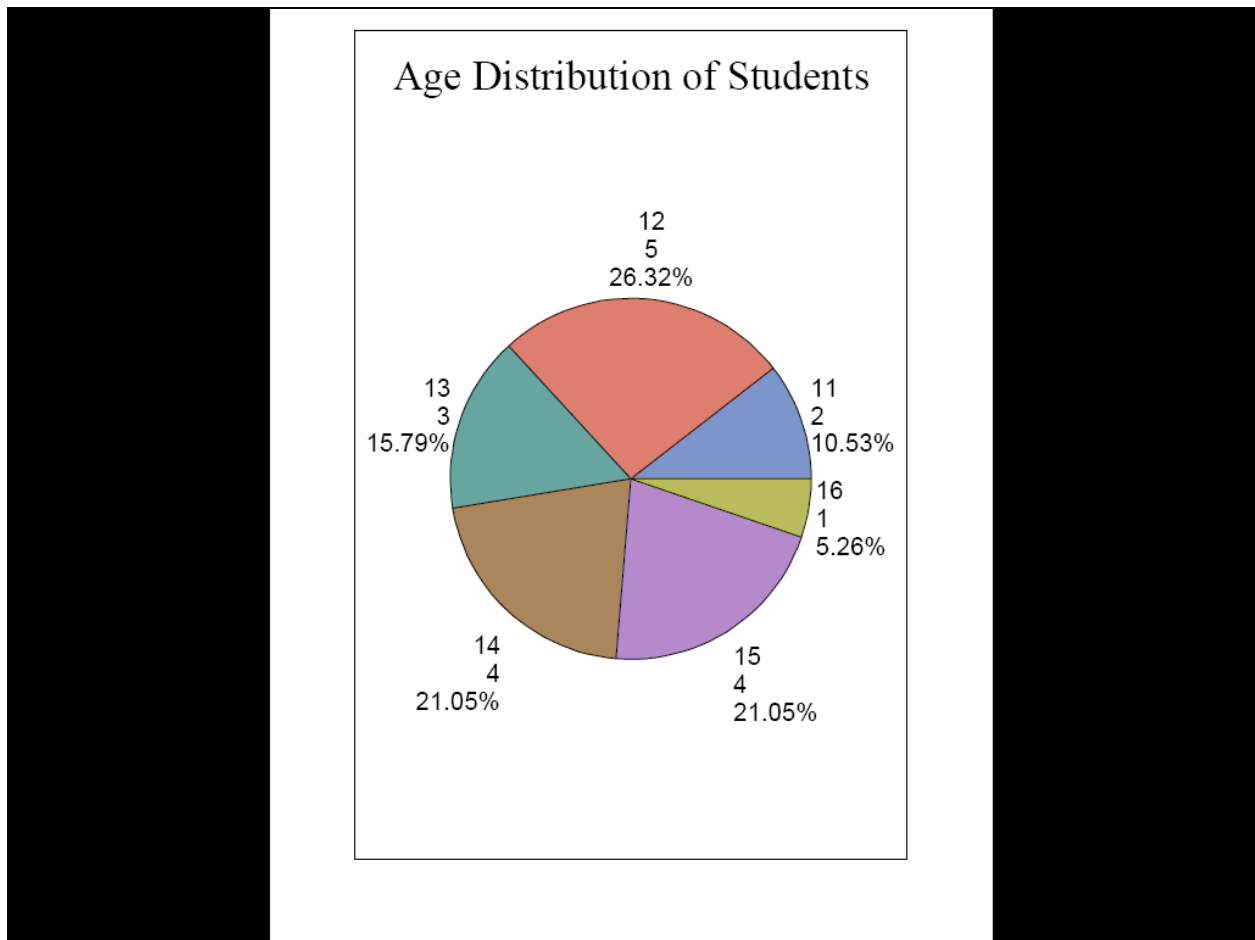
It is my opinion that the only visual displays that anyone can stand scrolling are trend charts that cover a huge time span and maps. For a long trend chart, scrolling sideways is natural, even if inconvenient, since you can not see the whole time span at once. A standing To Do I have set for myself is to determine whether there is a better way to handle long time spans.

PDF

When packaging your work as an RTF or DOC file by using ODS RTF, or as web pages by using ODS HTML, it can end up looking different for different viewers. Furthermore, anyone can modify your finished product, except for the graphs (if you have used an uneditable graphic file type), and pass along the revised result as yours. In Internet Explorer, anyone can click View and then Source, and proceed to edit the HTML code. Even if the web page is READ-ONLY, it can be saved to a different folder, and then passed along from there. A PDF file usually looks the same for every viewer (the “P” in “PDF” stands for Portable), and unless one has special tools, the finished product can not be modified.

This section is based, in part, on my prior published work on PDF. For more information about use of ODS PDF, see Reference 5.

In one my prior papers (Version 9.1.3) on ODS and SAS/GRAPH (see Reference 6), I found that to get satisfactory results in PDF you need to use the SASPRTC device driver for color (SASPRTG for gray shades and SASPRTM for black-and-white). Below is what I got by explicitly coding GOPTIONS DEVICE=SASPRTC; in my Version 9.2 program:



Here is the code that I used:

```
proc catalog cat=work.gseg kill; run; quit;
  /* clear out any leftovers from a prior code run during the SAS session */
OPTIONS ORIENTATION=PORTRAIT;
OPTIONS NODATE NONUMBER;
OPTIONS RIGHTMARGIN=1 IN LEFTMARGIN=1 IN BOTTOMMARGIN=1 IN; /* top margin automatic */
ODS NORESULTS;
ODS LISTING CLOSE;
ODS PDF FILE="C:\MWSUG 2009\Paper 2\9.2\ExplicitlySpecifiedSASPRTC.PDF"
  NOTOC /* turn off Table of Contents */
  TITLE="Pie Chart in PDF File When Explicitly Specifying the SASPRTC Driver for ODS
PDF Destination in Version 9.2"
  AUTHOR="LeRoy Bessler PhD" SUBJECT="Demonstrate PDF Pie Chart with SASPRTC Driver"
  KEYWORDS="PDF SAS/GRAPH ODS 'SASPRTC Device Driver'";
  /* accepting default ODS style Styles.Printer */
goptions reset=all;
goptions device=SASPRTC;
goptions border;
goptions ftext='Helvetica' htext=3 PCT;
title1 height=2.5 PCT ' ' /* blank space above title */
justify=center /* force a line break */
font='Times' height=5 PCT 'Age Distribution of Students';
proc gchart data=sashelp.class;
pie age / discrete noheading percent=outside name="SASPRTC";
run; quit;
ODS PDF CLOSE;
```

NOTE: Especially if you are creating a one-page or otherwise rather simple PDF, specify NOTOC on your ODS PDF statement. Often, the present-by-default Table of Contents is a nuisance or at least a waste of page display space. The viewer should not be forced to manually turn it off, and might not even realize that it can be turned off.

An important change for Version 9.2 is that ODS automatically selects the correct device driver for each destination. Using essentially the same code as above, but with the statement **goptions device=SASPRTC;** removed, the result is identical to that obtained with the statement included.

Here is a screen print of a PDF file containing a noncustomized table:

Name	Sex	Age	Height	Weight
Alfred	M	14	69.0	112.5
Alice	F	13	56.5	84.0
Barbara	F	13	65.3	98.0
Carol	F	14	62.8	102.5
Henry	M	14	63.5	102.5
James	M	12	57.3	83.0
Jane	F	12	59.8	84.5
Janet	F	15	62.5	112.5
Jeffrey	M	13	62.5	84.0
John	M	12	59.0	99.5
Joyce	F	11	51.3	50.5
Judy	F	14	64.3	90.0
Louise	F	12	56.3	77.0
Mary	F	15	66.5	112.0
Philip	M	16	72.0	150.0
Robert	M	12	64.8	128.0
Ronald	M	15	67.0	133.0
Thomas	M	11	57.5	85.0
William	M	15	66.5	112.0

Here is the code used to create the document above:

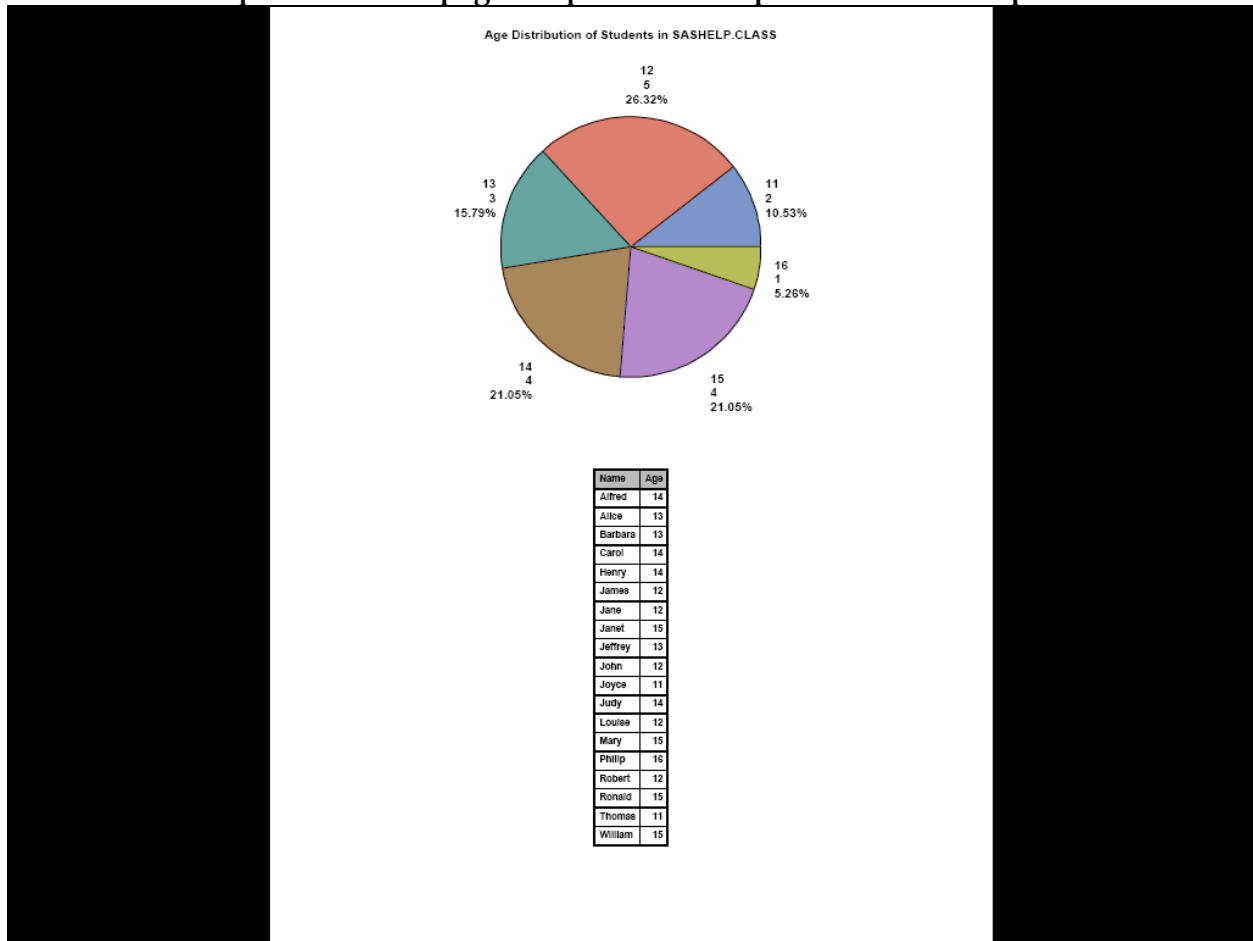
```
OPTIONS ORIENTATION=PORTRAIT;
OPTIONS NODATE NONUMBER;
OPTIONS RIGHTMARGIN=1 IN BOTTOMMARGIN=1 IN LEFTMARGIN=1 IN; /* Top Margin automatic */
ODS NORESULTS;
ODS LISTING CLOSE;

ODS PDF FILE="C:\MWSUG 2009\Paper 2\9.2\Table.PDF"
  NOTOC /* turn off Table of Contents */
TITLE="Table in PDF File in Version 9.2" AUTHOR="LeRoy Bessler PhD"
SUBJECT="Demonstrate PDF Table" KEYWORDS="PDF Table ODS";
/* accepting default ODS style Styles.Printer */

goptions reset=all;
title1 height=2.5 PCT ' ' /* blank space above title */
justify=center          /* force a line break          */
font='Times' height=5 PCT
'Students in SASHELP.CLASS';
proc print data=sashelp.class noobs;
run;

ODS PDF CLOSE;
```

Here is a screen print of a one-page Graph-Plus-Companion-Table composite.



Here is the code used to create the PDF document above:

```
proc catalog cat=work.gseg kill; run; quit;
options reset=all;
options nodate nonumber orientation=portrait;
ODS PDF FILE="C:\MWSUG 2009\Paper 2\9.2\GraphAndTable.PDF"
  STARTPAGE=NO /* prevent automatic new page for table after graph */
  NOTOC /* turn off Table of Contents */
  TITLE="Graph and Table in PDF File in Version 9.2" AUTHOR="LeRoy Bessler PhD"
  SUBJECT="Demonstrate PDF Table" KEYWORDS="PDF Graph Table ODS";
  /* accepting default ODS style Styles.Printer */
goptions hsize=6.5 IN vsize=4.875 IN ftext='Helvetica/Bold' htext=3 PCT;
title 'Age Distribution of Students in SASHELP.CLASS';
proc gchart data=sashelp.class;
pie age / discrete noheading percent=outside name="PieChart"; run; quit;
ods escapechar='^';
ods pdf text='^{\newline 1}'; /* This spacer is mandatory.
  Without it, the top of the table will overlay the bottom of the pie chart. */
* ods pdf text='^S={just=center font=( "Times Roman",16PT,Bold)}Students in
SASHELP.CLASS';
  /* Uncomment the above statement to get title text for the table.
  A TITLE statement will not be displayed. */
proc print data=sashelp.class noobs;
var name age / style=[font_face='Helvetica' font_weight=Bold font_size=8 pt];
run;
ODS PDF CLOSE;
```

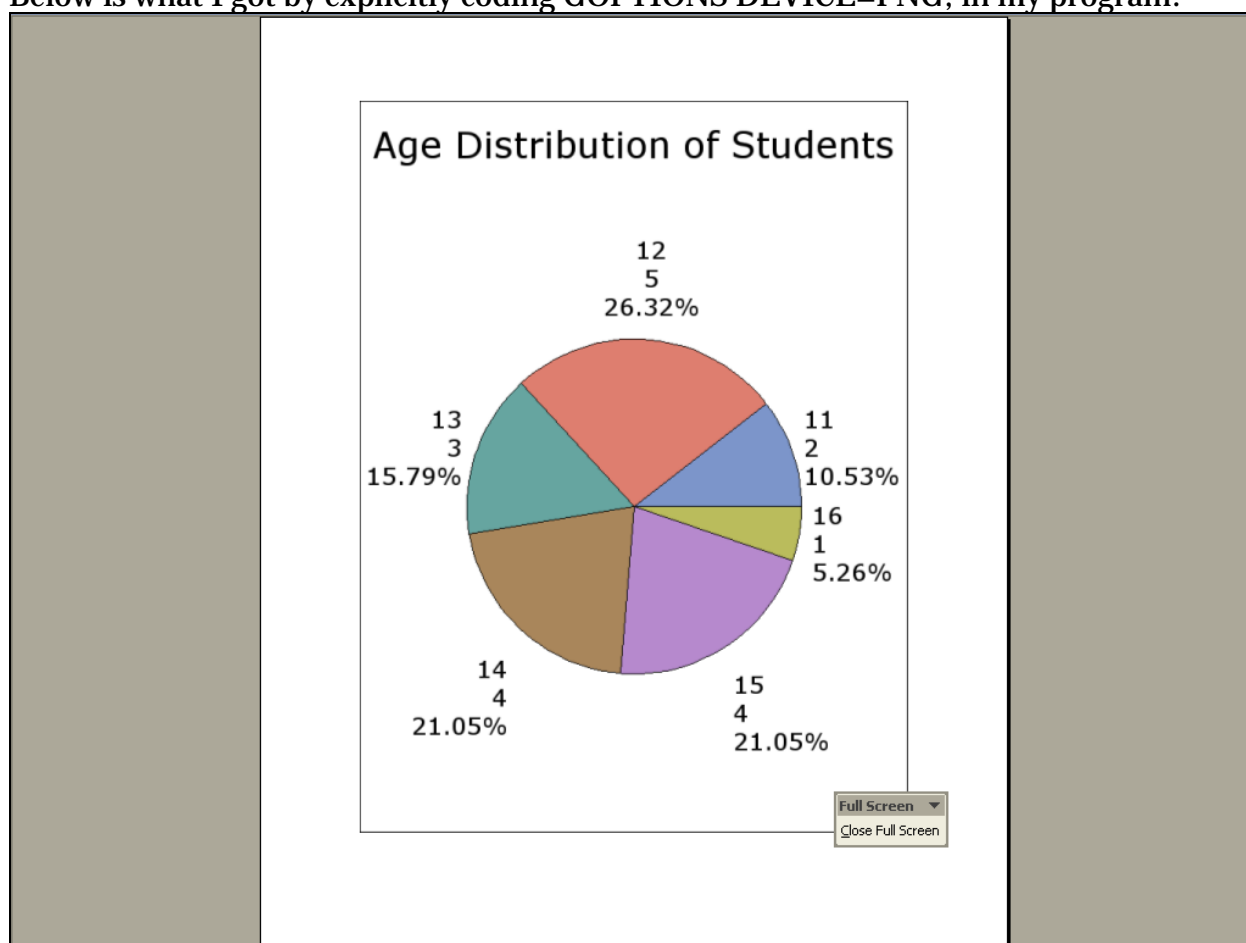
RTF / Word

If preparing a Word document with a mix of graphs, tables, and text, it can be very convenient to create the graphs as graphic stream files directly on disk, and then manually Insert them into an opened Word document, exactly where you want them, and then resize them, if desired, to your satisfaction. For the tables, they will be more easily inserted and auto-sized to fit with ODS. I.e., creating a graph-free Word document would be your first step. Last, text can be typed in. But for repetitive production, building the whole package, including text, with automation will pay for itself. Any data-dependent text can be delivered with dynamically loaded symbolic variables.

When you use the ODS destination, you can define the output RTF file with file extension doc, rather than rtf. That way anyone who receives the file from you can not be confused by what might be a file extension that is unfamiliar to them.

Rather than spend time on RTF for matters already well covered by other authors (see, e.g., Reference 7), I will focus primarily, but not exclusively, on delivery of graphs via RTF. In my prior (Version 9.1.3) work on ODS and SAS/GRAPH (see Reference 6), I found that to get satisfactory results in RTF you need to use the PNG device driver.

Below is what I got by explicitly coding `GOPTIONS DEVICE=PNG;` in my program:



I was surprised to see the graph nearly fill the vertical space. If you create the same graph on disk without ODS RTF packaging, it has a landscape orientation. Here is the code that I used:

```
proc catalog cat=work.gseg kill; run; quit;
/* clear out any leftovers from a prior code run during the SAS session */

proc template; /* Create a Custom Style based on the default style designed for RTF */
define style styles.RTFwithOneInchMargins;
parent=styles.rtf;
style body from document /
topmargin=0.75 in /* there is a 0.25 reserve margin in addition to this */
rightmargin=1 in
bottommargin=1 in
leftmargin=1 in;
end;
run;

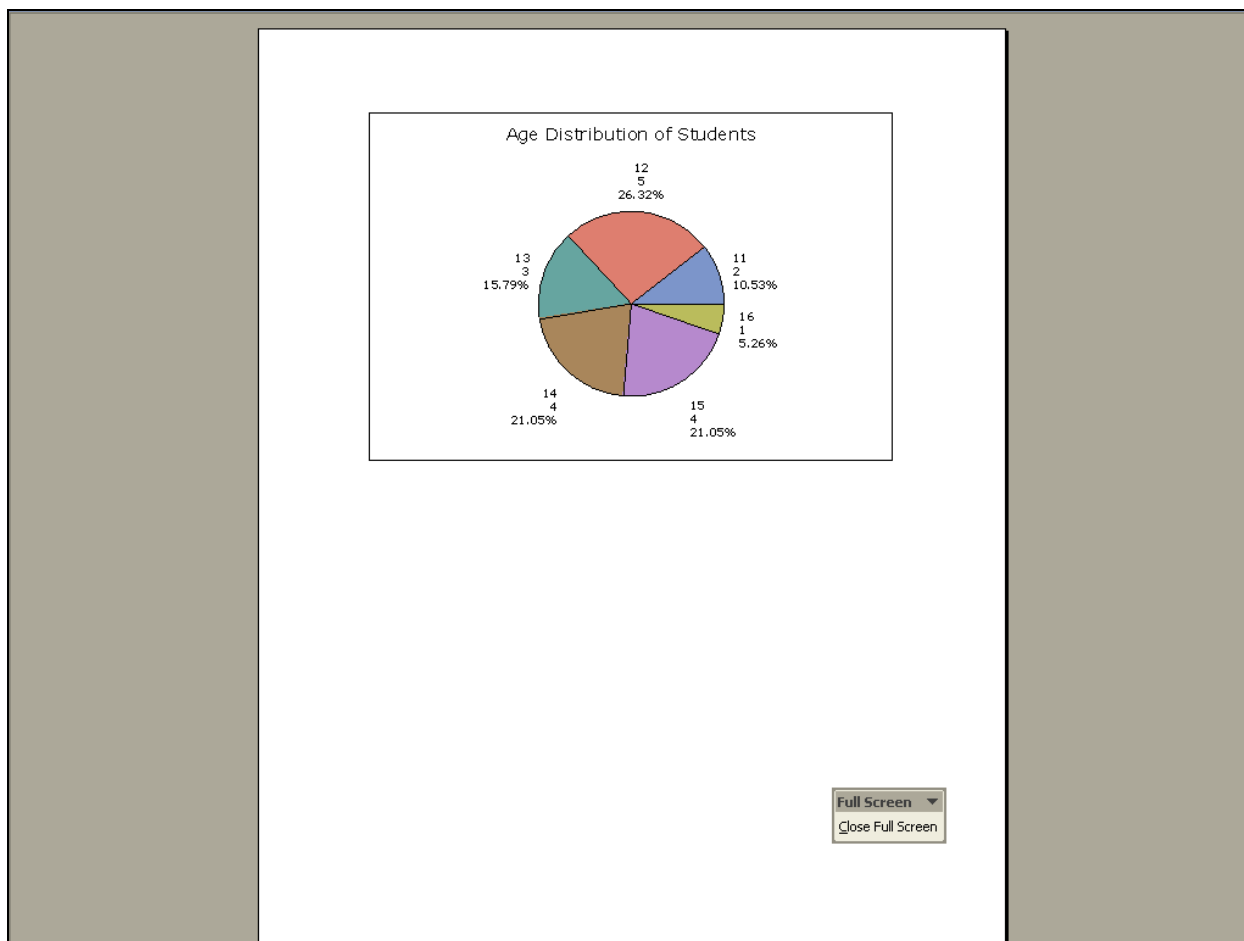
OPTIONS ORIENTATION=PORTRAIT;
OPTIONS NODATE NONUMBER;
ODS NORESULTS;
ODS LISTING CLOSE;
ODS RTF
FILE="C:\MWSUG 2009\Paper 2\9.2\ExplicitlySpecifiedPNG.DOC"
TITLE="Pie Chart in Word DOC File When Explicitly Specifying the PNG Driver for ODS
RTF Destination in Version 9.2"
AUTHOR="LeRoy Bessler PhD"
STYLE=Styles.RTFwithOneInchMargins;

goptions reset=all;
goptions device=PNG;
goptions border;
goptions ftext='Verdana' htext=3.25 PCT;
title1 height=2.5 PCT ' ' /* blank space above title */
justify=center /* force a line break */
height=5 PCT font='Verdana' 'Age Distribution of Students';
proc gchart data=sashelp.class;
pie age / discrete noheading percent=outside name="ForcePNG";
run; quit;

ODS RTF CLOSE;
```

If in the code above you replace PNG with GIF everywhere, you will get an unsatisfactory result, in terms of font rendering. For reasons not known to me, no Windows fonts can be used in the combination of RTF and GIF. The ugly SIMULATE font is substituted, as I also found in my prior work for Version 9.1.3.

An important change for Version 9.2 is that ODS automatically selects the correct device driver for each destination. Using essentially the same code as above, but with the statement **goptions device=PNG;** removed, the result is as shown on the next page, showing the inherent landscape orientation designed into the PNG device driver.



Here is the code used to create the Word document shown above:

```

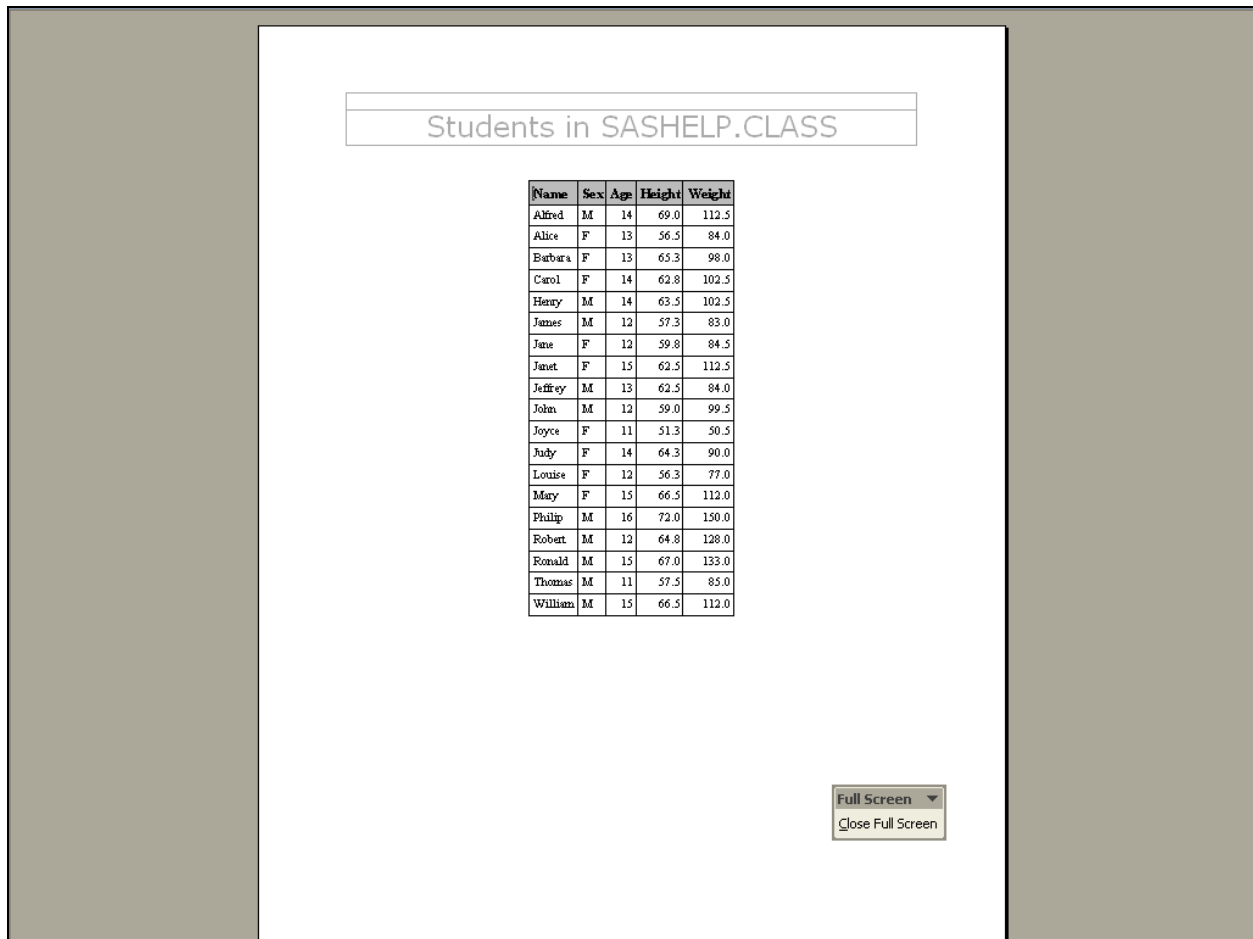
OPTIONS ORIENTATION=PORTRAIT;
OPTIONS NODATE NONUMBER;
ODS NORESULTS;
ODS LISTING CLOSE;
ODS RTF
FILE="C:\MWSUG 2009\Paper 2\9.2\PieChart.DOC"
TITLE="Pie Chart in Word DOC File for ODS RTF Destination in Version 9.2"
AUTHOR="LeRoy Bessler PhD"
STYLE=Styles.RTFwithOneInchMargins;

goptions reset=all;
goptions border;
goptions ftext='Verdana' htext=3.25 PCT;
title1 height=2.5 PCT ' ' /* blank space above title */
justify=center          /* force a line break          */
height=5 PCT font='Verdana'
'Age Distribution of Students';
proc gchart data=sashelp.class;
pie age / discrete noheading percent=outside name="PieAlone";
run; quit;

ODS RTF CLOSE;

```


Here is a screen print of the Word document containing a noncustomized table created with ODS RTF (when printed, the title is black and the title framing disappears):



Here is the code used to create the document above:

```

OPTIONS ORIENTATION=PORTRAIT;
OPTIONS NODATE NONUMBER;
ODS NORESULTS;
ODS LISTING CLOSE;
ODS RTF
FILE="C:\MWSUG 2009\Paper 2\9.2\Table.DOC"
TITLE="Table in Word DOC File Using ODS RTF Destination in Version 9.2"
AUTHOR="LeRoy Bessler PhD"
STYLE=Styles.RTFwithOneInchMargins;

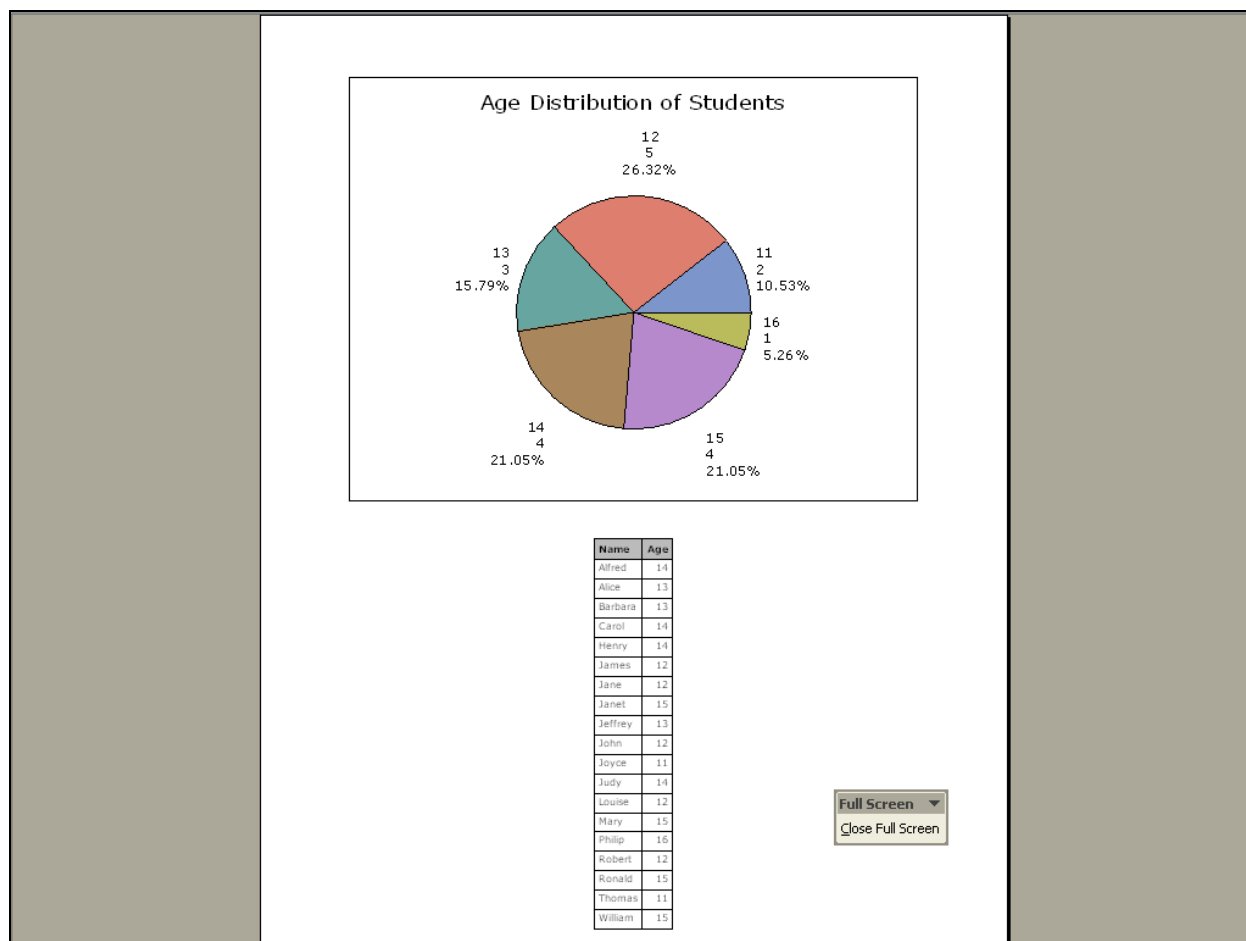
goptions reset=all;
title1 height=2.5 PCT ' ' /* blank space above title */
justify=center           /* force a line break          */
height=5 PCT font='Verdana'
'Students in SASHELP.CLASS';
proc print data=sashelp.class noobs;
run;

ODS RTF CLOSE;

```

Below is a screen print of a one-page Graph-Plus-Companion-Table composite.

When the page image is shrunk down from 11 inches to 4.88 inches for this paper, the table is hard to read. Of course, the graph could have been made smaller, but I wanted to accept the device driver's native dimensions, and let the composition process auto-fit it to the full width of the page, less margins.



Here is the code used to create the Word document above:

```
proc template; /* Create a Custom Style based on the default style designed for RTF */
define style styles.RTFwithQtrInchTopBottom;
parent=styles.rtf;
style body from document /
topmargin=0.0 in /* there is a 0.25 reserve margin in addition to this */
rightmargin=1 in
bottommargin=0.25 in
leftmargin=1 in;
end;
run;

OPTIONS ORIENTATION=PORTRAIT;
OPTIONS NODATE NONUMBER;
ODS NORESULTS;
ODS LISTING CLOSE;
```

```

ODS RTF
FILE="C:\MWSUG 2009\Paper 2\9.2\GraphAndTable.DOC"
TITLE="Graph and Table in Word DOC File Using ODS RTF Destination in Version 9.2"
AUTHOR="LeRoy Bessler PhD"
STYLE=Styles.RTFwithQtrInchTopBottom
STARTPAGE=NO;

goptions reset=all;
goptions hsize=6.5 IN vsize=4.875 IN;
goptions border;
goptions ftext='Verdana' htext=3.25 PCT;
title1 height=2.5 PCT ' ' /* blank space above title */
justify=center /* force a line break */
height=5 PCT font='Verdana'
'Age Distribution of Students';
proc gchart data=sashelp.class;
pie age / discrete noheading percent=outside name="PieChart";
run; quit;

title;
proc print data=sashelp.class noobs;
var name age / style=[font_face='Verdana' font_size=8 pt];
run;

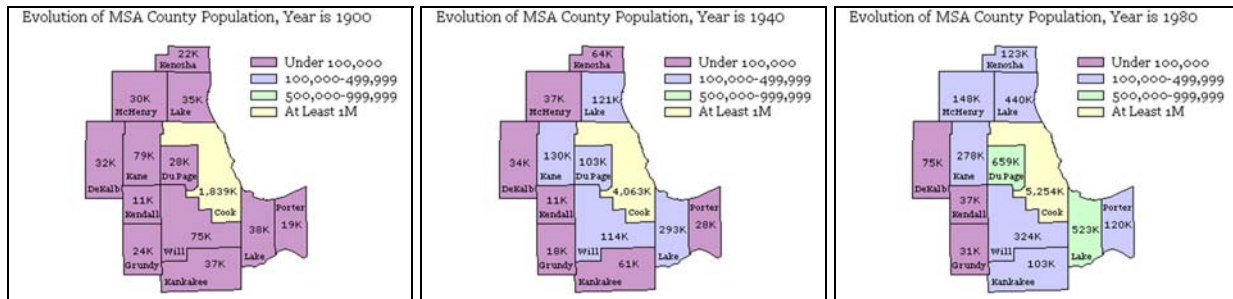
ODS RTF CLOSE;

```

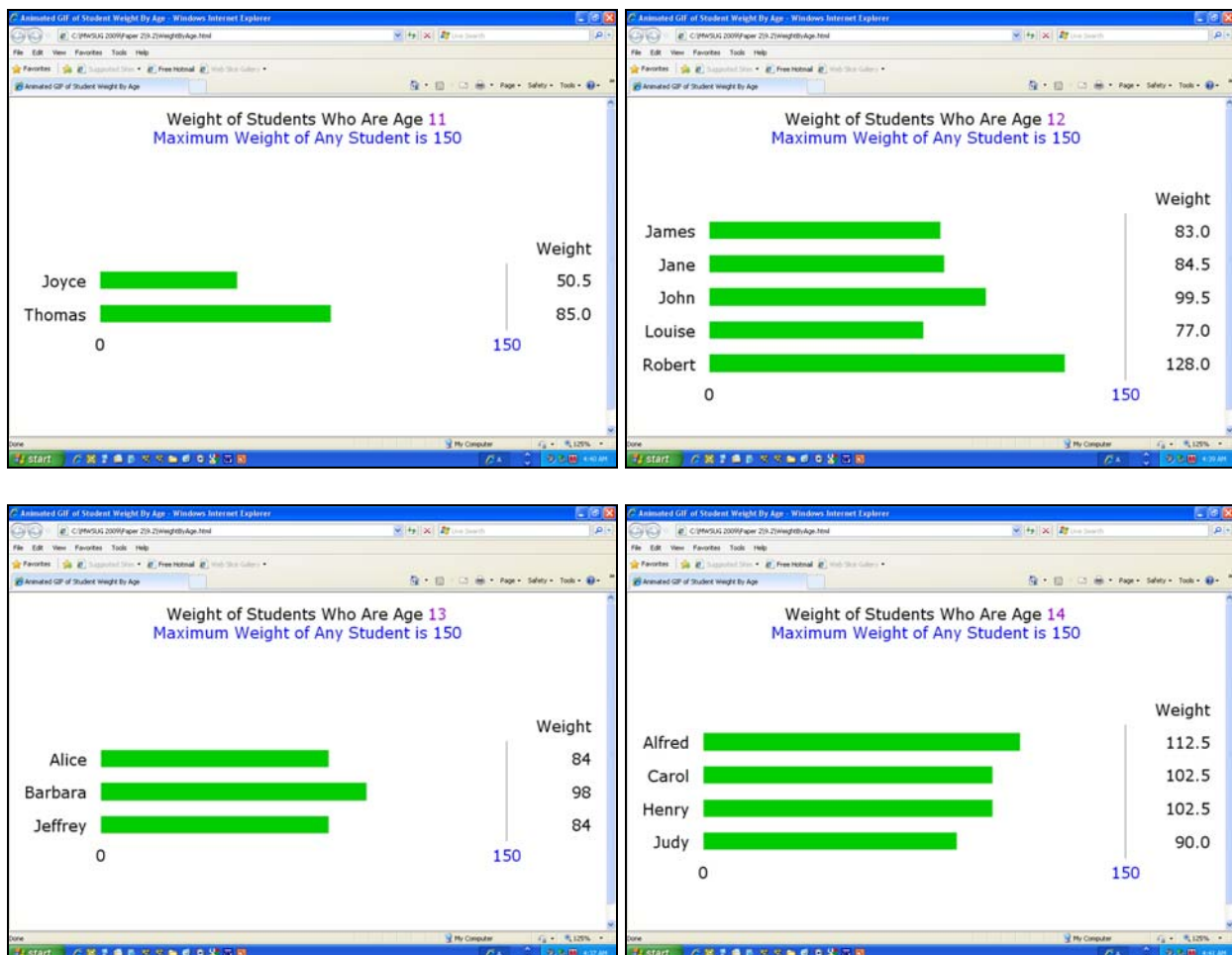
If the idea of animated titles in an ODS-created Word document appeals to you, please see the later section on Animated RTF Text.

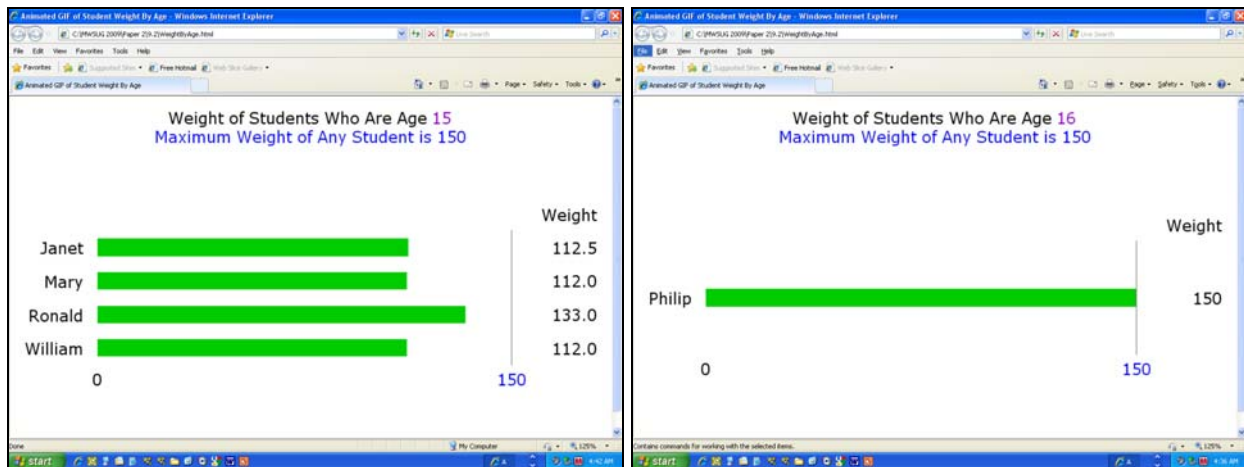
Animation for Communication, Not Decoration or Recreation

In the 2D world of the screen, time can be handled as a third dimension, by serially presenting two-dimensional graphs using animation. The graphs can be any type: pie chart, bar chart, trend line, scatter plot, map, etc. Below are only three frames from my first animation (Population Growth around Chicago).



Though the method is the same, I would rather show you how to create an animation with sample data that you already have with your SAS software and without having to get into PROC GMAP and annotation of a map. Below are all of the images from an animated web graph that shows weight, in a sense, as a function of age.





Here is the code used to create the animation example shown above:

```
/* The use of custom XPIXELS and YPIXELS in sizing below
   requires creation of a modified graph device driver.
   For more information, see SAS Note SN-005366 at support.sas.com. */
libname gdevice0 "C:\MWSUG 2009\Paper 2\9.2";
proc gdevice nofs cat=gdevice0.devices;
delete modanim;
copy gifanim from=sashelp.devices newname=modanim;
modify modanim hsize=0 vsize=0;
run; quit;

proc catalog cat=work.gseg kill; run; quit;
/* clear out any leftovers from a prior code run during the SAS session */

/* This section of code needed only to web publish the AnimDemo.GIF file. */

ods noresults;
ods listing close;
ods html path="C:\MWSUG 2009\Paper 2\9.2\ForIE8" (url=none)
      body="WeightByAge.html" (title="Animated GIF of Student Weight By Age")
      gtitle gfootnote
      style=styles.Minimal;

GOPTIONS RESET=ALL DEVICE=GIF;
GOPTIONS XPIXELS=980 YPIXELS=525; /* size to fit in a 1024 X 768 screen
   with an extra tool bar using Internet Explorer 8 as the web browser */

PROC GSLIDE NAME="AnimDemo" DESCRIPTION=' ';
TITLE1 'this mandatory static placeholder image will never actually display';
RUN; QUIT;

ods html close;
ods listing;

/* End of setup code needed only to web publish the AnimDemo.GIF file. */

proc sort data=sashelp.class out=work.ToChart;
by Age;
run;
```

```

proc means data=sashelp.class noprint max;
var Weight;
output out=FoundMax Max=MaximumWeight;
run;

data _null_;
set FoundMax;
call symput('MaxWeight',trim(left(MaximumWeight)));
run;

GOPTIONS RESET=ALL DEVICE=modanim
ITERATION=3 /* show the sequence of animation panels 3 extra times
              after the first time */
DELAY=100 /* display time for each animation panel in units of 0.01 seconds,
            but actual time is display-tool-dependent */
GOPTIONS XPIXELS=980 YPIXELS=525; /* Size to fit in a 1024 X 768 screen
            with an extra tool bar using Internet Explorer 8 as the web browser.
            If not web publishing, use XPIXELS=800 YPIXELS=600. */
GSFNAME=animout GSFMODE=replace
HTEXT=5 PCT FTEXT="Verdana" CTEXT=CX000000;

TITLE1 H=5 PCT F="Verdana" COLOR=CX000000
      "Weight of Students Who Are Age " COLOR=CX9900CC "#BYVAL(Age)"
JUSTIFY=CENTER COLOR=CX0000FF
      "Maximum Weight of Any Student is &MaxWeight";

PATTERN1 V=SOLID COLOR=CX00CC00;

AXIS1 LABEL=NONE MAJOR=NONE MINOR=NONE STYLE=0 VALUE=(JUSTIFY=RIGHT);
AXIS2 LABEL=NONE MAJOR=NONE MINOR=NONE STYLE=0
      VALUE=(TICK=1 '0' TICK=2 COLOR=CX0000FF "&MaxWeight")
      ORDER=(0 &MaxWeight) OFFSET=(+3,+7);

OPTIONS NOBYLINE;

FILENAME animout "C:\MWSUG 2009\Paper 2\9.2\ForIE8\AnimDemo.gif";
PROC GCHART DATA = work.ToChart;
BY Age;
HBAR Name / SUMVAR=Weight SUMLABEL="Weight"
      MAXIS=AXIS1 RAXIS=AXIS2
      WIDTH=2 SPACE=2 COUTLINE=CX00CC00
      HREF=&MaxWeight;
RUN; QUIT;
FILENAME animout CLEAR;

```

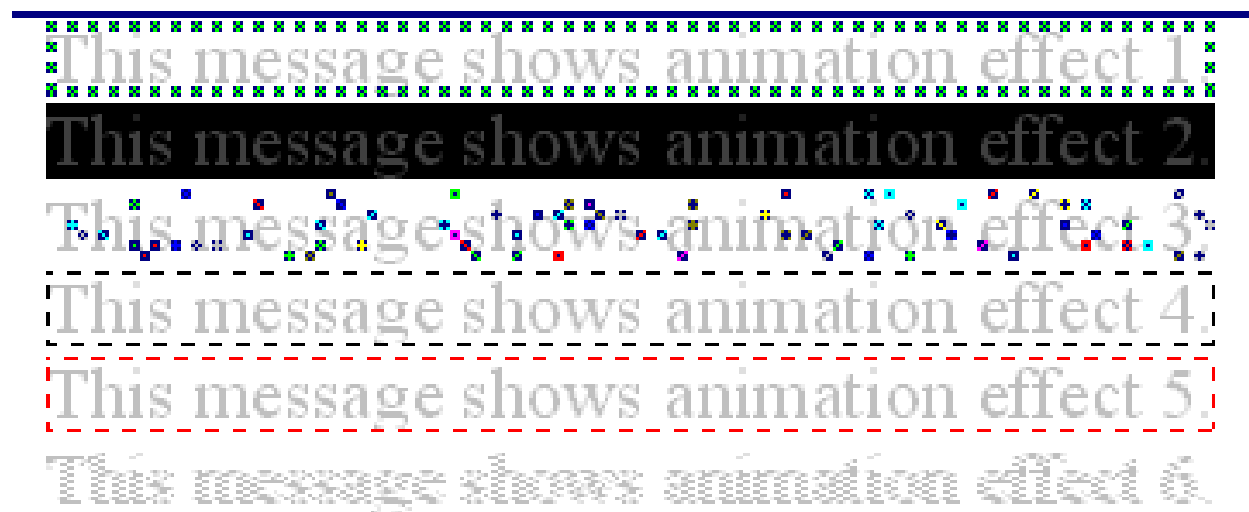
An animated GIF file can be displayed outside of html. E.g., you can insert it as a picture in a Microsoft PowerPoint slide. For that purpose, you can use the same code as above except that you: (a) must change XPIXELS and YPIXELS to 800 and 600 in order to make best use of the slide display area; and (b) can omit the web publishing code.

Animated RTF Text: Confusing the Possible With Useful and Appropriate

You can use ODS to create an RTF file with imbedded images, but animated GIFs do NOT work in RTF or Word documents.

Apart from drawing attention to text without use of bold or underline or large print, the value and purpose of animated text as implemented in RTF is questionable.

Below is a clipped out and enlarged static example of all six types of RTF text animation. Though it would be better for you to just run the code and see the result yourself, let me try to describe the animated effects. In Effect 1, the rectangular border of symbols switches from green X's to blue solid squares to blue-green diamonds to green X's in a continuous cycle. In Effect 2, the black overlay appears and disappears on top of the "Peek-A-Boo" text. In Effect 3, the multi-colored symbols "twinkle", but intermittently all vanish so you can briefly see all text clearly. In Effects 4 and 5, the black and red dashes travel around the rectangular track. In Effect 6, the text alternates between a solid font and fuzzy font that you see in the screen print. When text is printed, none of these effects are present. In my opinion, Effect 5 is conspicuous, without being irritating or obstructive. (But the rotating border does obscure the bottom of the letter "g".)



The **\ANIMTEXTn** control word (also available for FOOTNOTE statements) is used:

```
ods rtf body='c:\RTFdemo\ProbablyFrivolous.rtf' style=styles.minimal;
title1 h=16pt '{\animtext1 This message shows animation effect 1.}';
title2 h=16pt '{\animtext2 This message shows animation effect 2.}';
title3 h=16pt '{\animtext3 This message shows animation effect 3.}';
title4 h=16pt '{\animtext4 This message shows animation effect 4.}';
title5 h=16pt '{\animtext5 This message shows animation effect 5.}';
title6 h=16pt '{\animtext6 This message shows animation effect 6.}';
proc print data=sashelp.class; run;
ods rtf close;
```

Images

Here I will show you how to use image files in your SAS and SAS/Graph output in ways that I think are the most likely to be of interest. For more ways, please see Reference 8.

The commonest image files that you can find on the web have file extensions JPG (or JPEG) and GIF. Image files that you might create with a digital camera or a scanner, regardless of the file type or format that is used, are handled exactly the same way as JPG, GIF, or any other image file. My focus is on static image files, not output from the ActiveX or Java drivers.

When creating graphs in stand-alone code (i.e., outside of ODS) to be programmatically imbedded with the techniques shown here, I would use GIF for Version 9.1.3, and possibly PNG for Version 9.2, if not using GIF. When using an ODS code block that includes invocation of a SAS/GRAPH procedure, Version 9.2 automatically selects the correct image file format (SAS/GRAPH device driver). In Version 9.1.3, you must explicitly specify choice of device driver. Use GIF for HTML, PNG for RTF, and SASPRTC (color), SASPRTG (gray shades), or SASPRTM (black-and-white) for PDF.

Though SAS/GRAPH supports a JPEG (or JPG) device driver, I can think of no reason to use it. The JPG image file format was invented for the Joint Photographic Experts Group expressly to support photographic images. JPG files support a huge number of colors for continuous tone images. Business graphics and typical logos do not require a large number of distinct colors. GIF is adequate to such needs. PNG is the device driver that the SAS/GRAPH development team favors for Version 9.2 graphics.

I will show you one way to deliver an image inlaid within the area of a graph and several ways to deliver an image with a table. (See Reference 8 for more ways, including imbedding data-dependent images or icons inside table cells.)

First, I will take up tables. I am fond of PROC PRINT, since I create simple tables, but the similar methods work for PROC TABULATE and PROC REPORT for fancier reports.

How To Display Images Along with an ODS Table in a Web Page

The easiest way to get an image immediately above or immediately below a table requires only some extra STYLE specification code on the PROC PRINT statement.

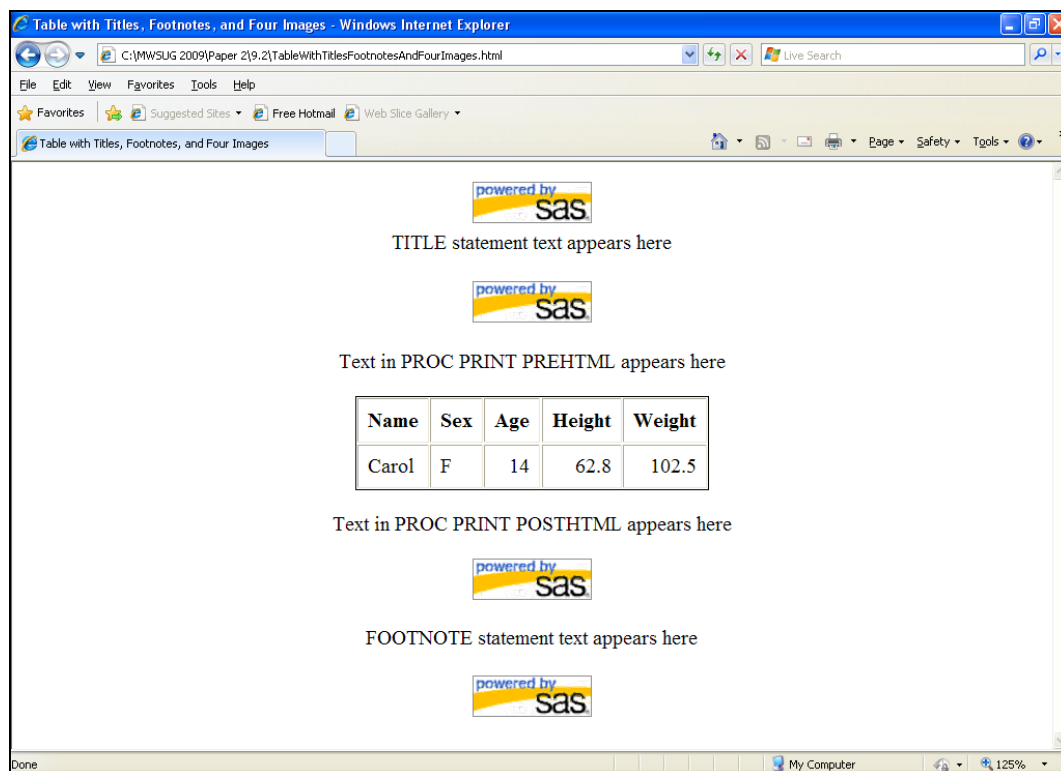
In this case, however, the image is located between the TITLE text and the top of the table if above the table, or between the bottom of the table and above the FOOTNOTE text if below the table.

It is possible to supply title text and footnote text without the TITLE and FOOTNOTE statements, and to position such text below an “Over” image and above an “Under” image, respectively.

By default, these easily delivered images, whether or not you relocate the title and/or footnote text, are horizontally centered. With some extra coding, you can left or right justify the images, with or without some horizontal space wedged between the image and the web page margin, if you want do not want it at the edge of the web page.

To automatically deliver an image above TITLE text or below FOOTNOTE text it is necessary to use PROC TEMPLATE. In this case, the image is left-justified by default. With some extra PROC TEMPLATE coding, it is possible to center the image as shown here or to otherwise adjust its horizontal position.

Below is an unrealistic demonstration web page with four instances of the same image. All have ALT text, and could have been defined as hyperlinks to other web pages.



Here is the code used to create the demonstration web page:

```
/* src= in all four places in this code
   assumes that the image file is in the target folder for the web page */

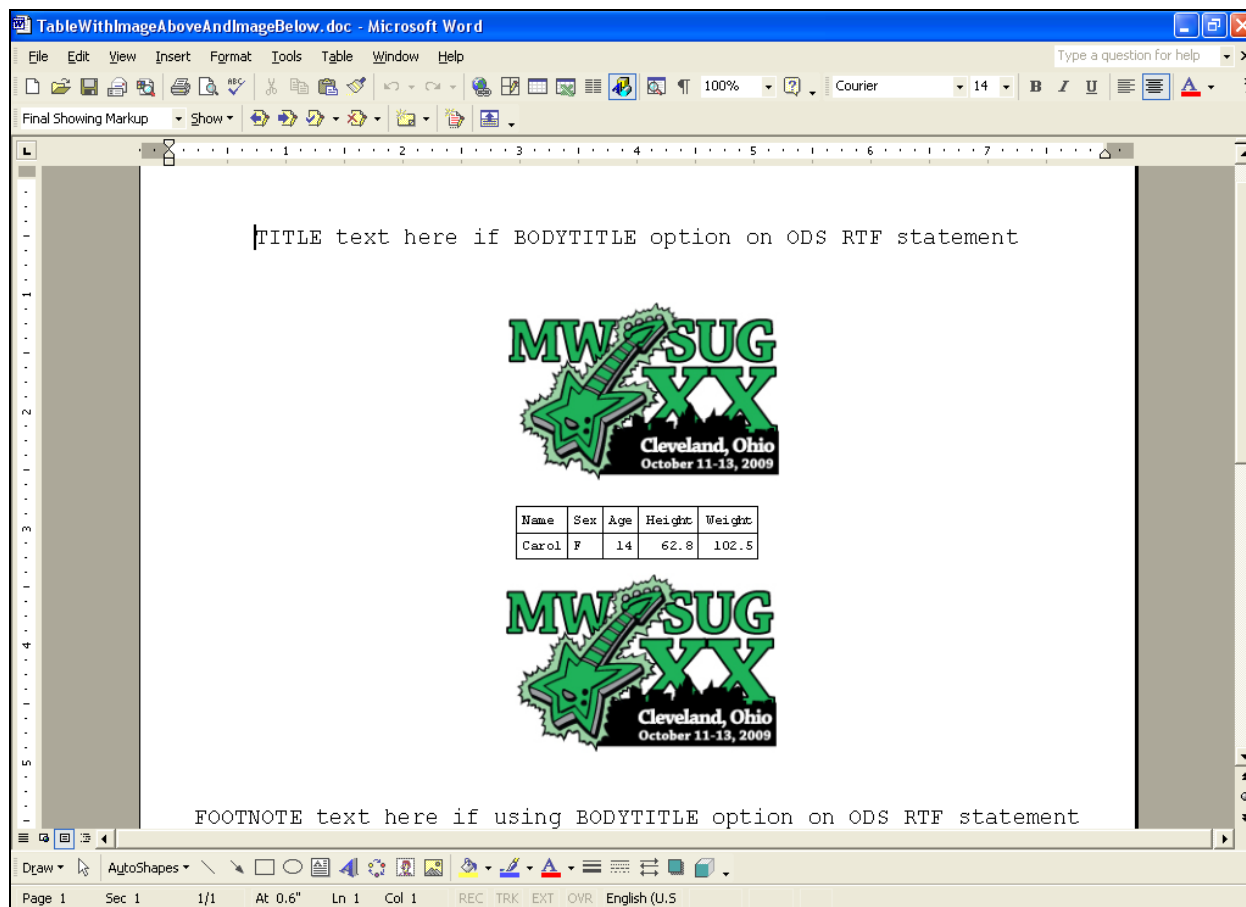
proc template;
edit Styles.Minimal
  as Styles.MinimalImagesTopAndBottom;
style body /
  prehtml = '<center>
             '
  posthtml = '<center>
             ';
end;
run;

/* Without <center>, images from this custom style would be left-justified. */

ods noresults;
ods listing close;
ods html path='C:\MWSUG 2009\Paper 2\9.2' (url=none)
  body='TableWithTitlesFootnotesAndFourImages.html'
  (title='Table with Titles, Footnotes, and Four Images')
  style=Styles.MinimalImagesTopAndBottom;
title 'TITLE statement text appears here';
footnote 'FOOTNOTE statement text appears here';
proc print data=sashelp.class(where=(Name EQ 'Carol')) noobs
style =
[prehtml="<img src='saslanim_ImageInSameFolderAsWebPage.gif'
  alt='Image Above Table and Below TITLE Statement Text: Powered by SAS'>
</br></br>
Text in PROC PRINT PREHTML appears here
</br></br>"
posthtml="</br>
Text in PROC PRINT POSTHTML appears here
</br></br>
<img src='saslanim_ImageInSameFolderAsWebPage.gif'
alt='Image Below Table and Above FOOTNOTE Statement Text: Powered by SAS'>"];
/* One </br> forces a line break.
Second </br> makes that line blank by forcing another line break.
However, to get an empty line between the table and
"Text in PROC PRINT POSTHTML appears here" requires only one </br>. */
run;
ods html close;
ods listing;
```

How To Display Images Along with an ODS Table in a Word/RTF Page

The easiest way to get an image immediately above or immediately below a table requires only some extra STYLE specification code on the PROC PRINT statement. The image is located between the TITLE and the top of the table if above the table, or between the bottom of the table and above the FOOTNOTE if below the table. For more ways to work with images (or hyperlinks) using ODS RTF, please see Reference 7.



Here is the code used to create the demonstration Word page:

```
options nodate nonumber;
options reset=all;
ods noresults;
ods listing close;
ods rtf file='C:\MWSUG 2009\Paper 2\9.2\TableWithImageAboveAndImageBelow.doc'
      style=Styles.Minimal bodytitle;
title h=14pt "TITLE text here if BODYTITLE option on ODS RTF statement";
footnote h=14pt "FOOTNOTE text here if using BODYTITLE option on ODS RTF statement";
proc print data=sashelp.class(where=(Name EQ 'Carol')) noobs
style =
  [preimage ='C:\MWSUG 2009\Paper 2\9.2\MWSUG_2009_logo.jpg'
   postimage='C:\MWSUG 2009\Paper 2\9.2\MWSUG_2009_logo.jpg'];
run;
ods rtf close;
ods listing;
```

Presenting an Image As Hyperlink (with Flyover Text) and a Table in a PDF

PREIMAGE and/or POSTIMAGE and/or BACKGROUNDIMAGE options can be used anywhere in SAS tabular report PROC output to display an image (or multiple images). The URL option can be used in TITLE, FOOTNOTE, or PDF TEXT statements, or in cells for data or for column headers, to create the hyperlink(s). In this example, the image is a logo, but it could be a SAS graph previously built on disk outside of ODS.

Here is the code used to create the PDF displayed on the following page:

```
goptions reset=all;
ods noresults;
ods listing close;
options nodate nonumber orientation=portrait;

ods pdf notoc
    file="C:\MWSUG 2009\Paper 2\9.2\TableWithAnImageAsHyperlinkWithFlyover.pdf"
    style=Styles.Minimal startpage=no;

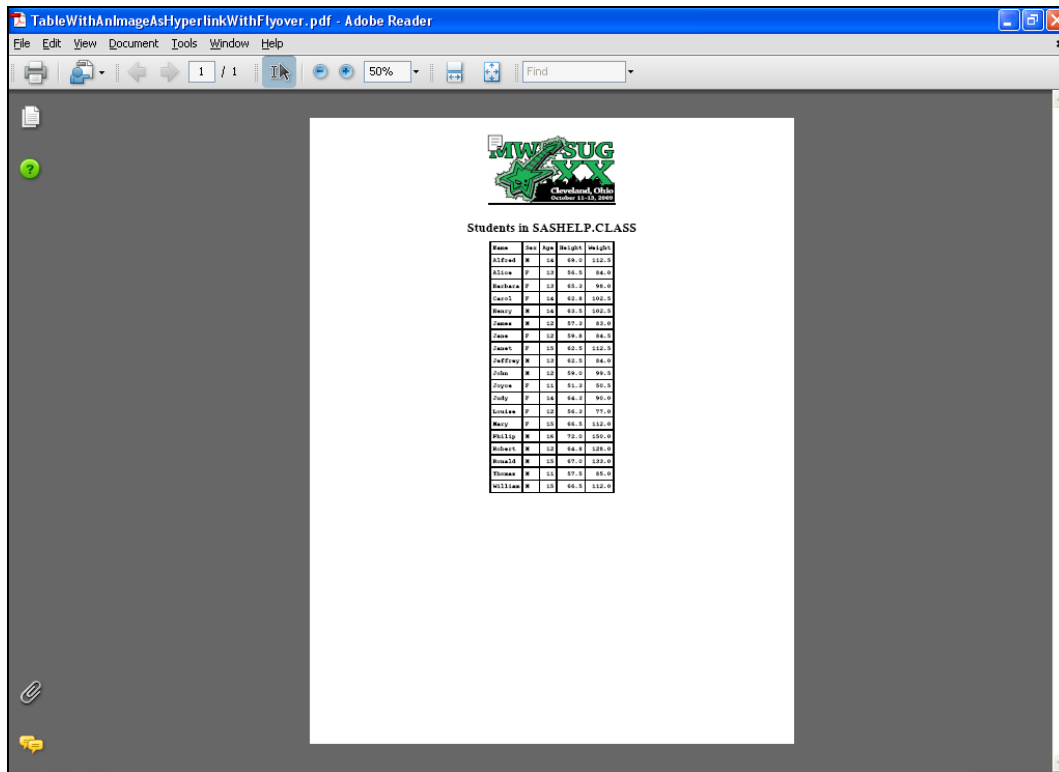
    /* First create a fake table to hold the image with hyperlink and flyover */
title; footnote;
proc print data=sashelp.class
    (where=(name='Alfred')) noobs label
    style(table) =[frame=void]
    style(header)=[
        url='http://www.mwsug.org'
        flyover='Click here to go to MWSUG Home Page'
        backgroundimage="C:\MWSUG 2009\Paper 2\9.2\MWSUG_2009_logo.jpg"
        foreground=white
        background=white
        cellwidth=5.62 cm cellheight=3.04 cm] /* Size however desired.
        4.62 cm X 3.04 cm would seem to be required to preserve the shape of the logo,
        which is 231 pixels X 152 pixels, but that distorts the text at the bottom
        of the logo. */
    style(data) =[foreground=white background=white font_size=1 pt];
var Age;
label Age='00'X; /* Use an unprintable character for the label. */
run;

ods escapechar='^';
ods pdf text='^{newline 1}'; /* This spacer is mandatory to avoid
    the top of the table overlapping the bottom of the image. */
ods pdf text='^S={just=center font=("Times Roman",16PT,Bold)}Students in
SASHELP.CLASS';
    /* Text from a TITLE statement will not be displayed. */

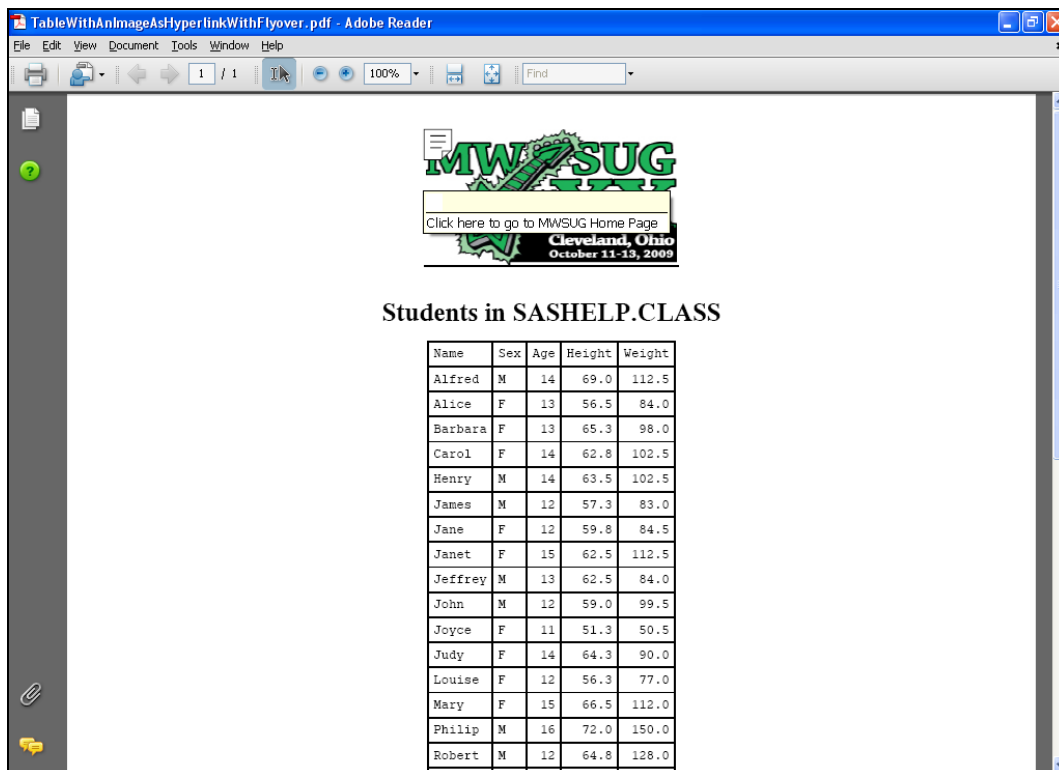
proc print data=sashelp.class noobs;
run;

ods pdf close;
```

Here is the PDF page:



Here is an enlargement with the mouse resting on the note icon at the upper left corner of the logo.



How To Imbed Images in Your SAS Graphs

In the web page below, the MWSUG 2009 logo has been imbedded. I hyperlinked the logo to www.mwsug.org, and supplied ALT text for it saying, "Logo of MWSUG 2009. Here linked to Midwest SAS Users Group Home Page."

The pie chart, rather than being based on real data, is a contrived sample color chart.

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, 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 defines 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.

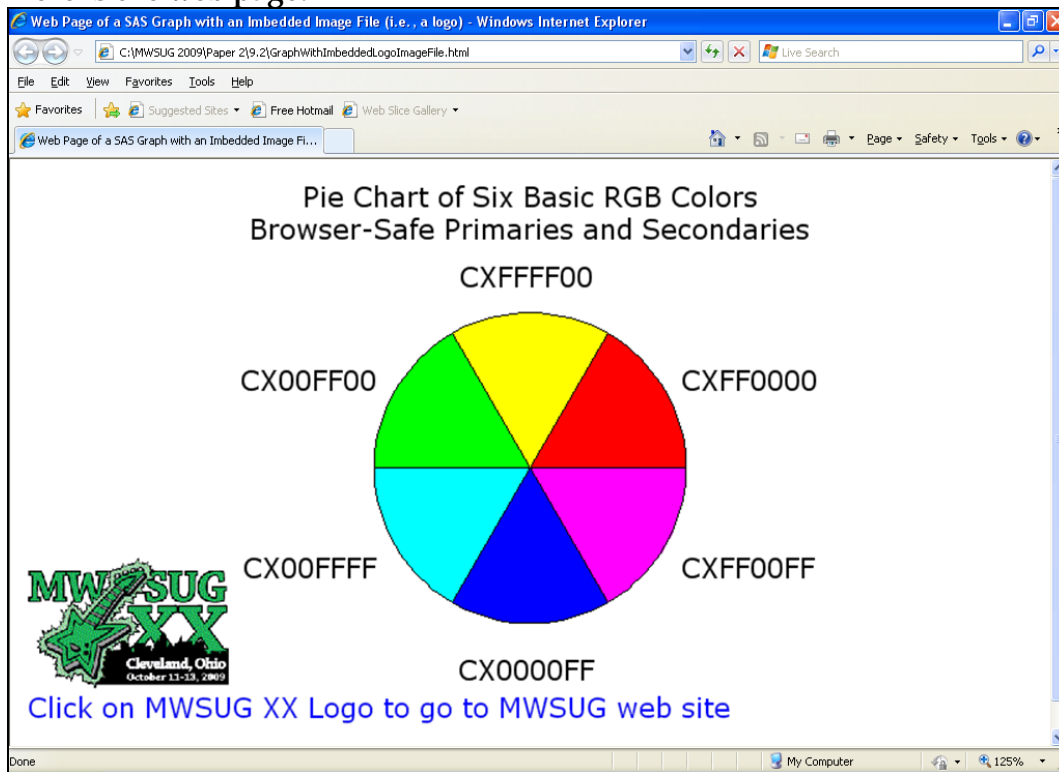
When you use FIT, if you under-allocate or over-allocate space for the image, it will shrink or stretch to fit. If the ratio of the dimensions of your allocation does not match the ratio of the dimensions of your source image, the result will be distorted. I rested my mouse on browser display of the original image, and pressed the right mouse button to see the properties of the image. Knowing those pixel counts for horizontal and vertical is needed in order to be able to maintain the relative height and width proportions of the logo as it is possibly shrunken or enlarged. Suppose the target graph has dimensions of 800 pixels wide by 600 pixels tall, but the source inlay image has native dimensions of 100 pixels by 50 pixels. Suppose you want the inlaid image to be 200 pixels wide, i.e., 25% of the width of the target area. To maintain the proportions of the inlay, you need to make it 100 pixels high. 100 pixels is 16.67% of the height of the target area. When doing the annotation, you must define the inlay using percents of the target area.

(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 might be truncated in one or the other direction, or both. When there is enough space for multiple tiles, but not for a whole number of multiples, those on the right and/or bottom edges will be truncated.)

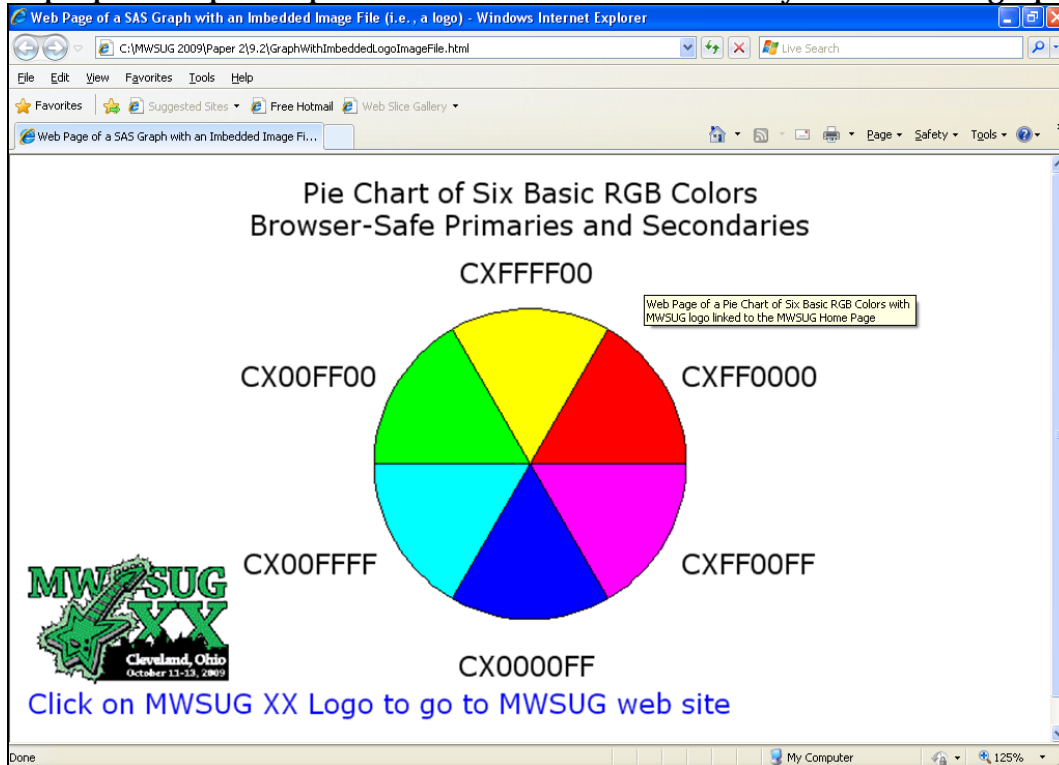
The Annotate WHEN variable is set to 'B', so the image is laid down in the graphic area BEFORE the graph is drawn. In this way, you can use an image as a background. You could instead overlay the graph, if that were your design objective, by setting WHEN to 'A', to have the image laid down in the graphic area AFTER the graph is drawn.

On the following pages, see illustrations of a web page of a graph with an inlaid logo, which also serves as a hyperlink, and the code used.

Here is the web page:



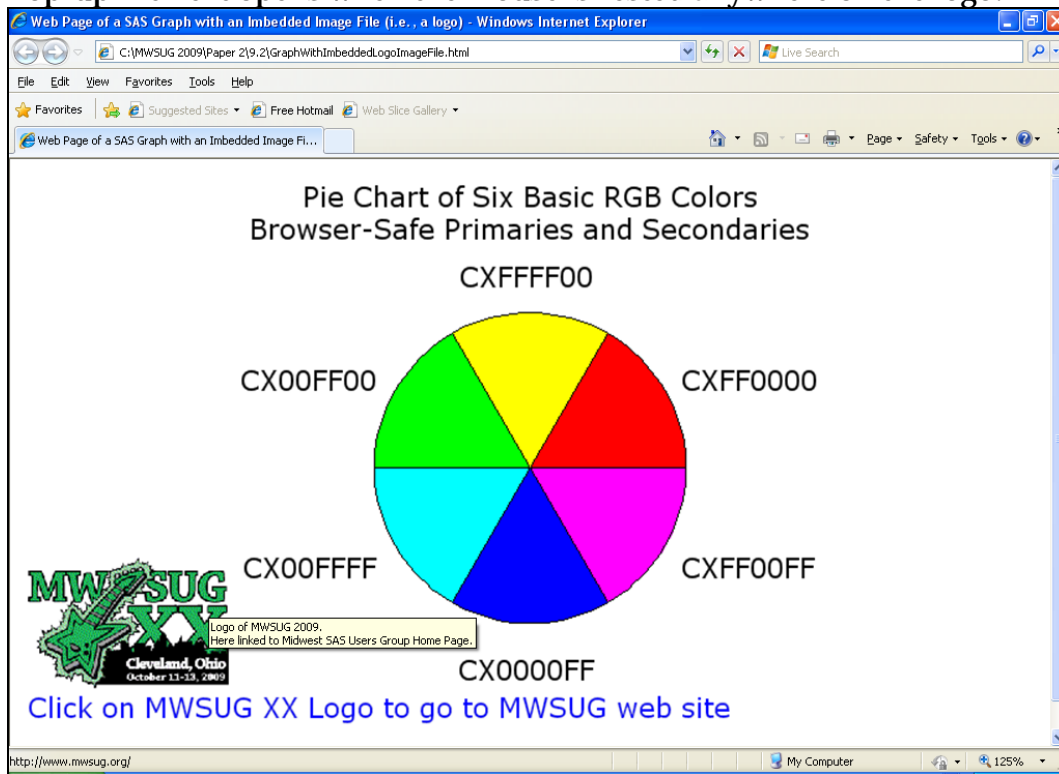
Pop-up Description opens when the mouse is rested anywhere on the graph:



Here is the pop-up magnified:

Web Page of a Pie Chart of Six Basic RGB Colors with MWSUG logo linked to the MWSUG Home Page

Pop-up Alt Text opens when the mouse is rested anywhere on the logo:



Here is the pop-up magnified:

Logo of MWSUG 2009.
Here linked to Midwest SAS Users Group Home Page.

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';
/* define a white clickable vertical bar for hyperlink and ALT text */
function='move';
x=0; y=7;
output;
function='bar';
x=20; y=31;
style='solid'; color='CXFFFFFFF';
html='href="http://www.mwsug.org" alt="Logo of MWSUG 2009.'
|| 'OD'X /* this causes a line break in the pop-up text */
|| 'Here linked to Midwest SAS Users Group Home Page.'';
output;
```



```

/* fill the same bar area with the image */
function='move';
x=0; y=7;
output;
function='image';
x=20; y=31;
imgpath="C:\MWSUG 2009\Paper 2\9.2\MWSUG_2009_logo.jpg";
style='fit'; /* fit image in rectangle with LL corner at 0,7 and UR corner at 20,31 */
output;
run;

proc catalog cat=work.gseg kill; run; quit;

options reset=all;
ods html path="C:\MWSUG 2009\Paper 2\9.2" (url=none)
    style=Styles.Minimal gtitle gfootnote
    body="GraphWithImbeddedLogoImageFile.html"
    (title="Web Page of a SAS Graph with an Imbedded Image File (i.e., a logo)");
options device=gif;
GOPTIONS XPIXELS=980 YPIXELS=525; /* size to fit in a 1024 X 768 screen
    with an extra tool bar using Internet Explorer 8 as the web browser */
options 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;
title 'Pie Chart of Six Basic RGB Colors'
    justify=center 'Browser-Safe Primaries and Secondaries';
footnotel justify=left color=CX0000FF
    'Click on MWSUG XX Logo to go to MWSUG web site';
footnote2 height=1 PCT ' '; /* prevents truncation of bottom of "g" in footnote */
pie RGBcolor / name='imgwlogo' description='Web Page of a Pie Chart of Six Basic RGB
Colors with MWSUG logo linked to the MWSUG Home Page'
    sumvar=PieValue noheading coutline=CX000000
    midpoints='CXFF0000' 'CXFFFF00' 'CX00FF00' 'CX00FFFF' 'CX0000FF' 'CXFF00FF'
    slice=outside value=none percent=none;
run; quit;
ods html close;

```

Multi-Media Wizardry to Make ODS Output Dance and Sing

Augmenting simple, static, passive output should not dilute, obstruct, or distract from the message in and the meaning of your data.

Audio can be a useful, extra, concurrent communication channel. And there are other multi-media options.

For web pages, this section shows you how to add audio (including background audio, which does not require launch of a media player on all or part of your screen), video, animation, a marquee ("traveling text", which is what you sometimes see at the bottom of a television news broadcast), and custom-sized images.

For PDF documents, this section shows you how to: include pop-up text (possibly data-dependent); create hyperlinks (possibly data-dependent) to audio clips, video clips, and animated graphics outside your PDF document; and imbed and re-size images.

For RTF/Word documents, the only way to access audio or video is via hyperlinks imbedded in lines of text or in the formatted values of a table. For how to create hyperlinks in titles, footnotes, formatted values, or labels, please see Reference 7. Such links could be used in a manner analogous to that demonstrated below in the section Action from PDF.

The topics presented do not require the use of PROC TEMPLATE and are suitable for users with little or no ODS experience.

My slide presentation can "dance and sing", but this paper's static collection of screen images cannot.

The Lively Web: Adding Sound and Motion to Your Web Pages (and Working with Images)



The web page above, when opened, plays “God Save the Queen”. The Queen’s personal flag (animated) is waving continuously. Both that image and the photo display pop-up text (a.k.a. HTML “ALT text”) when you rest your mouse on them. (When the pop-up window opens, note that a tool bar also appears near the upper left corner of the image.) One pop-up includes a line break—with multiple line breaks you can create a stack of multi-line pop-up text as high as you need/want, even to the extent of displaying a simple table. Software for visually impaired web users can read your HTML file, including the ALT text description of images, to convert that text to audio. LINK=, used to provide the TITLE hyperlink, can also be used in a FOOTNOTE. An explanation follows this listing of the code:

```
data _null_;
text = "'Queen & Prince board airplane' || '0D'X || 'Click for the British Monarchy web
site'";
call symput('AltText',text); run;

ods html path='C:\demo\WebPages' (url=None) file='Aug2005.html' style=styles.minimal;
title color=CX0000FF font='Georgia' height=20 pt
link='http://www.views-uk.org' 'Go To VIEWS Home Page';
proc print label noobs data=sashelp.class(where=(name EQ 'Philip'))
style(header) = [font_face=Georgia font_size=6 cellwidth=300]
```

```

style(data)    = [foreground=CXFFFFFF cellheight=1 cellwidth=1]
style = [rules=none cellspacing=0 frame=void borderwidth=0 cellpadding=0]
prehtml = "<img src='../Images\WavingFlagOfQueen.gif' width=89 height=101
        alt='Waving Flag of Her Majesty The Queen'>
        <font face=Georgia size=7 color=#FFFF00>
        <marquee bgcolor=#000099 width=100% hspace=20 vspace=10>
God save our gracious Queen. Long live our noble Queen. God save the Queen! Send her
victorious, happy, and glorious, long to reign over us. God save the Queen!
        </marquee>
        </font>
        <a href='http://www.royal.gov.uk'>
        <img src='../Images\Boarding.jpg' vspace=10 width=300 height=225 alt=&AltText>
        </a>
        <bgsound loop=infinite src='../Audios\UK_anthem.wav'>"];
var name; label name='Returning Home';
run;
ods html close;

```

Instead of relying on PROC TEMPLATE and a custom style, the ODS controls are supplied via the PROC PRINT statement. Here, for demonstration, the PROC PRINT presents an image caption (provided by the LABEL statement) via column header cell content. The data cell is shrunk and hidden. A real application, rather than this all-in-one proof of concept, would include a real data table and/or graph, and would use only audio and visual features that suit the application's total communication goal.

The HTML code used here is delivered via ODS using the PREHTML parameter. One can also or instead use a POSTHTML parameter to deliver HTML code that is applied after the tabular (or graphic) PROC output in the web page. E.g., a marquee and/or image(s) can be located below the PROC output.

The BGSOUND tag causes the web browser PC to play the sound file without opening a media player window, unlike what happens when you click on a hyperlink to a sound file. You can set LOOP to any integer you like, rather than to INFINITE.

HTML colors are RGB colors. You can use SAS RGB color values, but with # replacing the CX prefix. BGCOLOR in the MARQUEE tag is the color of the marquee background.

The MARQUEE tag “packages” the traveling text, but formatting the text must be done with the “surrounding” FONT tag. HTML font sizes are integers from 1 to 7. (If you use point sizes in your SAS code, ODS converts them HTML font sizes.)

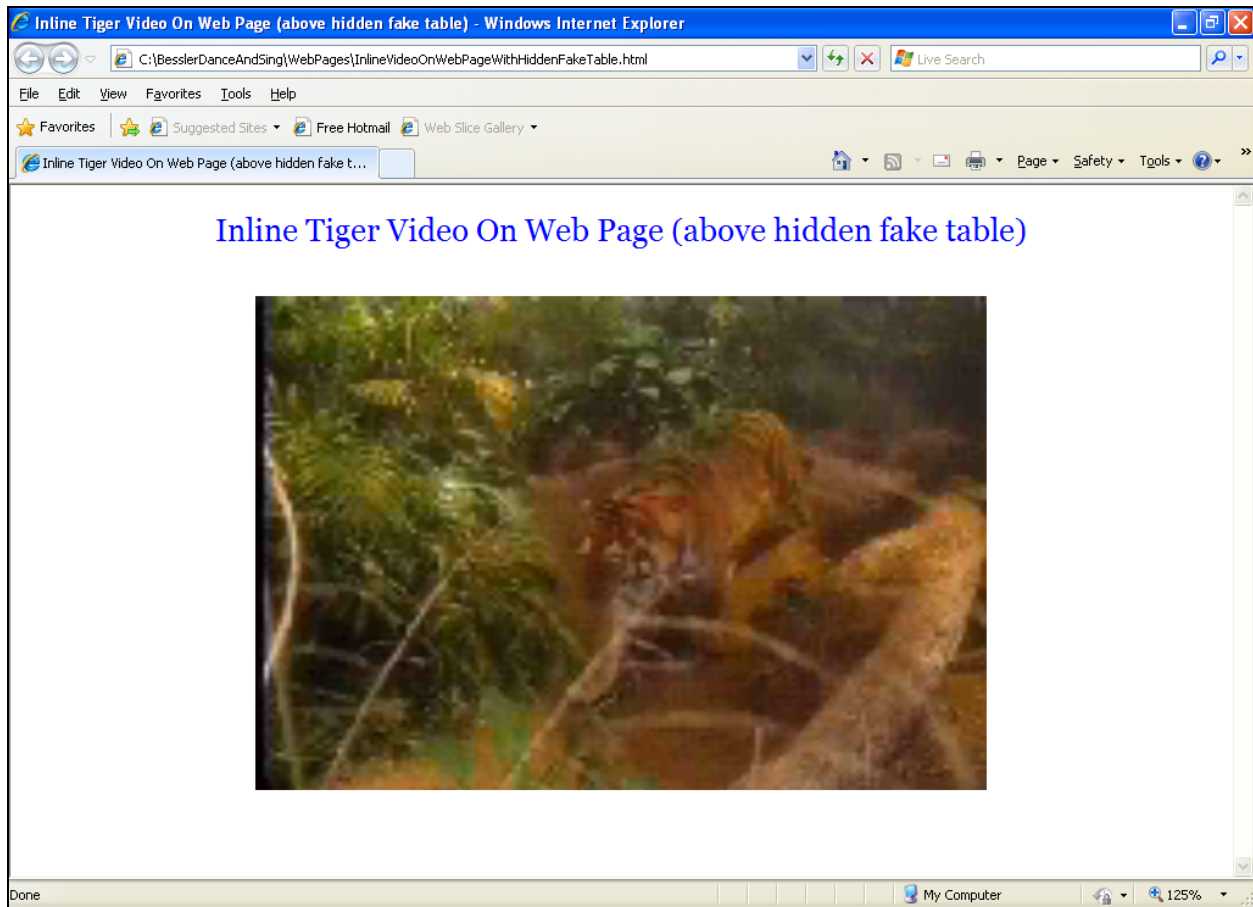
WIDTH, HEIGHT, HSPACE, and VSPACE HTML values are in pixels, except that a percent assignment for WIDTH is percent of web page width. VSPACE is space above *and* below the marquee or image. HSPACE is space at the left *and* right. Requesting the insertion of space only above, only below, only to the left, or only to the right is not supported in HTML.

SRC= identifies the location of the image or sound file. Both the animated GIF and the static JPG file are resized, but with care to preserve the aspect ratio (i.e., the ratio of picture height to picture width) to avoid distortion. (The Queen's flag is a square as shown, not the rectangular shape that is usually expected for flags.)

The ANCHOR tag (<a . . . >) with HREF= makes the photo image a hyperlink. (Animated images can be hyperlinks, too.)

Inline Video on the Web Page

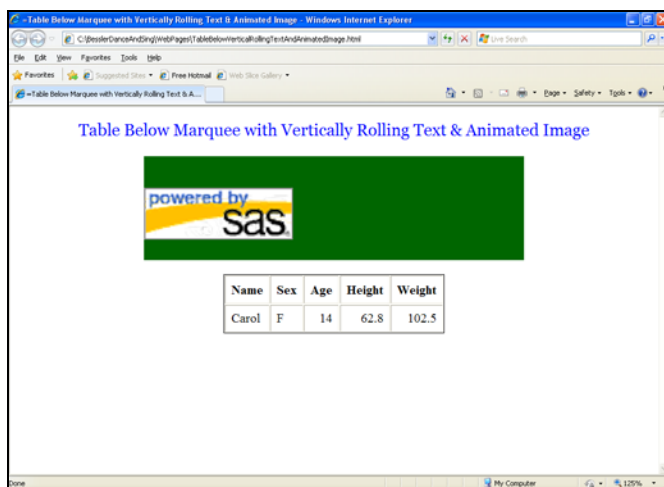
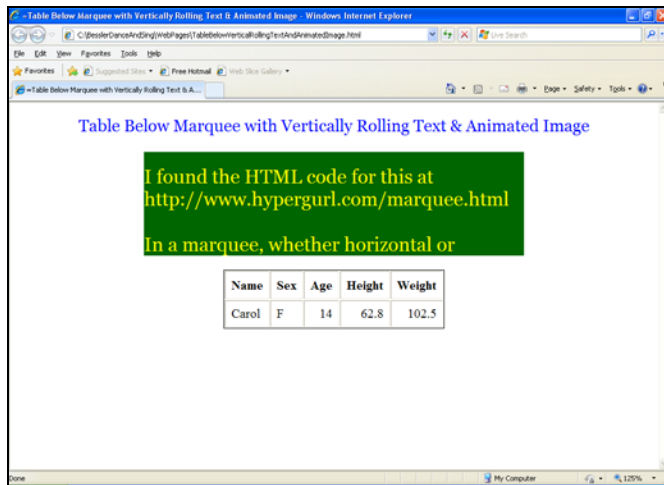
This requires that the browser be able to run inline video, and that it have the capability turned on. The example displays the video above a fake table—the PROC PRINT is needed to create a web page. The web page could present a real table below a smaller video screen. The size of the video window can be controlled.



```
options reset=all;
data _null_;
text = "'If the viewer had a web browser" || '0D'X ||
      "that supports inline video" || '0D'X ||
      "then a tiger video would be run here.'";
call symput('AltTextForVideo',text); run;
ods html path='C:\BesslerDanceAndSing\WebPages' (url=None)
      file='InlineVideoOnWebPageWithHiddenFakeTable.html'
      (title='Inline Tiger Video On Web Page (above hidden fake table)')
      style=styles.minimal;
title color=CX0000FF font='Georgia' height=16 pt
      'Inline Tiger Video On Web Page (above hidden fake table)';
proc print label noobs data=sashelp.class(where=(name EQ 'Philip'))
style=[font_size=1 foreground=CXFFFFFF cellheight=1 cellwidth=1
      rules=None cellspacing=0 frame=void borderwidth=0 cellpadding=0
      prehtml = "<img src='../images/TigerScreenShotFromTigerVideo.jpg'
                dynsrc='../videos/tiger.avi' start=mouseover
                vspace=10 width=480 height=324 alt=&AltTextForVideo>"];
var name; label name='00'X; run;
ods html close;
```

Vertical Marquee with Imbedded Animated Image above Table in Web Page

Below is an incomplete series of screen prints as the marquee text rolls upward.



Here is the code used:

```
goptions reset=all;
ods listing close;
ods html path='C:\BesslerDanceAndSing\WebPages' (url=none)
  file='TableBelowVerticalRollingTextAndAnimatedImage.html'
  (title='Table Below Marquee with Vertically Rolling Text & Animated Image')
  style=styles.minimal;
title color=CX0000FF font='Georgia' height=16 pt
  'Table Below Marquee with Vertically Rolling Text & Animated Image';
proc print label noobs data=sashelp.class(where=(Name eq 'Carol'))
style=[prehtml =
"<font face=Georgia size=5 color=#FFFF00>
  <marquee bgcolor=#006600 width=60% height=128 direction=up>
    <img src='../images\saslanim.gif' HEIGHT=64 WIDTH=184>
  <br>
  <br>I found the HTML code for this at
  <br>http://www.hypergurl.com/marquee.html
  <br>
  <br>In a marquee, whether horizontal or vertical,
  <br>you can include an image, including an animated GIF
  <br>
  <br>This GIF will cycle a few times,
  <br>and then becomes a stationary image
  </marquee>
  </font>
  <br><br>"];
run;
ods html close;
ods listing;
```

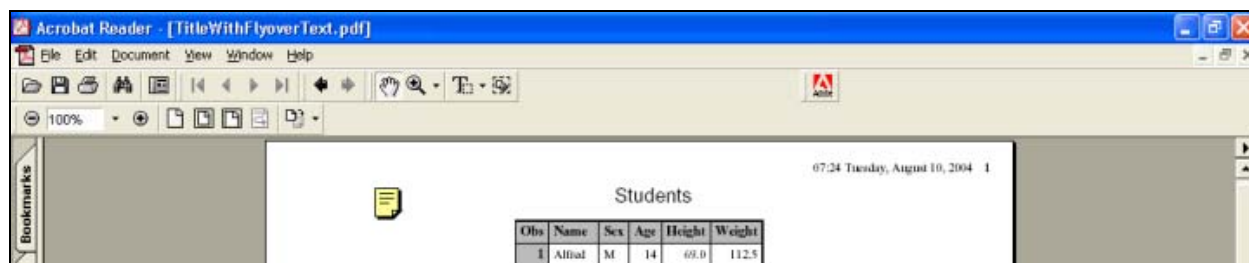
Action In PDF, Part 1: Pop-Up Text for Your PDF Document

If you would like some information as flyover/pop-up text, not hardcopy, in your PDF document, you can create a (scrollable) note box “behind” an icon. The icon does not print, and the note does not print. You can right-click the icon to open the note box. There is no apparent limit on the length of note text. You can define the width of the note box. If you take the default box width in this example, the icon will overlay the left end of the “hard text” on the page. Increasing its height would push the table around.

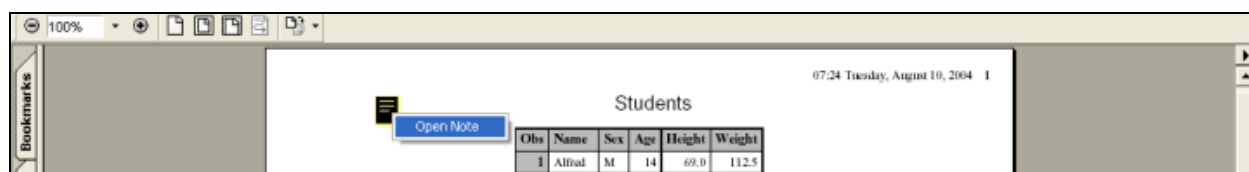
With this code:

```
goptions reset=all;
options orientation=portrait;
ods pdf notoc startpage=never file="&FolderForPDF.\TitleWithFlyoverText.pdf";
ods pdf text="^S={FLYOVER='This is the CLASS data set from the SASHELP data library'
                CELLWIDTH=6IN JUST=CENTER FONT_SIZE=16PT FONT_FACE=Helvetica}Students";
proc print data=sashelp.class; run;
ods pdf close;
```

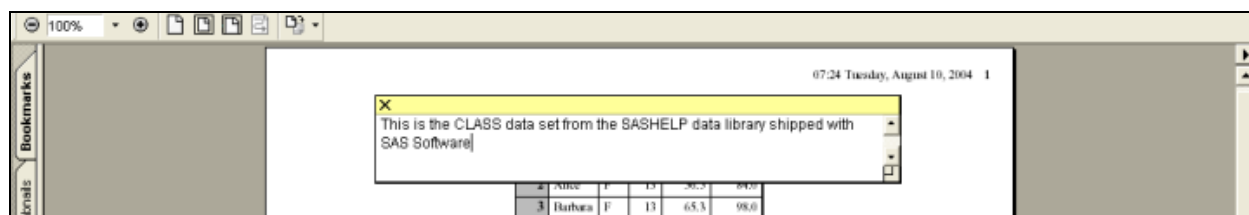
you get this result (in a series of clippings from the PDF page), starting with display of the icon:



Then resting the mouse on the icon:



And finally opening the note:



Action In PDF, Part 2: Data-Dependent Pop-Up Text in Table Cells

Use a SAS format to selectively define the pop-up text based on data value, with code such as this:

```
proc format lib=work;
value AgeNote
    11 = 'Youngest'
    16 = 'Oldest'
    Other = ' ';
run;

goptions reset=all;
options orientation=portrait;
ods pdf notoc file="&FolderForPDF.\SomeDataCellsWithFlyoverText.pdf";
proc print data=sashelp.class;
var Name Sex;
var Age / style=[CELLWIDTH=1.2IN /* widen the cells to prevent the notes
                                from overlaying the data */
                just=right flyover=AgeNote.] ;
var Height Weight;
run;
ods pdf close;
```

Here is a clipping from the PDF page with one of the data-dependent pop-up notes opened:

10	John	M	12	59.0	99.5
11	Joyce	F	11	51.3	50.5
12	Judy	F	14	64.3	90.0
13	Louise	F	12	56.3	77.0
14	Mary	F	15	66.5	112.0
15	Philip	M	16	72.0	150.0
16	Robert	M	12	64.8	128.0
17	Ronald	M	15	67.0	133.0
18	Thomas	M	11	57.5	85.0
19	William	M	15	66.5	112.0

Action From PDF: Image, Audio, and Video for a Multi-media PDF Extravaganza

I collected elements for this example PDF document as follows.

At <http://www.phuse.info/photos/photo1.html>, I right-clicked on the image, then selected "Save Picture As", and stored the image as photo1.jpg in C:\PDF\images. I downloaded the Bavarian Anthem sound file from david.national-anthems.net/bav.htm, and stored it as bav.wav in C:\PDF\audios. Using Microsoft Windows Explorer, I searched for PC-resident files of the form *.avi. I selected one from the Program Files for SAS (in C:\Program Files\SAS Institute\SAS\V8\core\sasmisc) and stored it as tiger.avi in C:\PDF\videos.

```
%let urlPath = %str(C:/PDF);
%let filePath = %str(C:\PDF);
%let commonfmt = %str(JUST=CENTER VJUST=MIDDLE FONT_SIZE=12PT CELLHEIGHT=24PT);
%let TOCfmt = &commonfmt%str( CELLWIDTH=10CM);
%let TXTfmt = &commonfmt%str( CELLWIDTH=20CM);

proc format;
value $AVlnks
'Inside Stadthalle' = "&urlPath./images/photo6.jpg"
'Anthem of Bavaria' = "&urlPath./audios/bav.wav"
'SAS Video'         = "&urlPath./videos/tiger.avi";
run;

data work.ChoicesForImageAudioVideo;
length medium $ 17;
medium = 'Inside Stadthalle'; output;
medium = 'Anthem of Bavaria'; output;
medium = 'SAS Video';          output;
run;

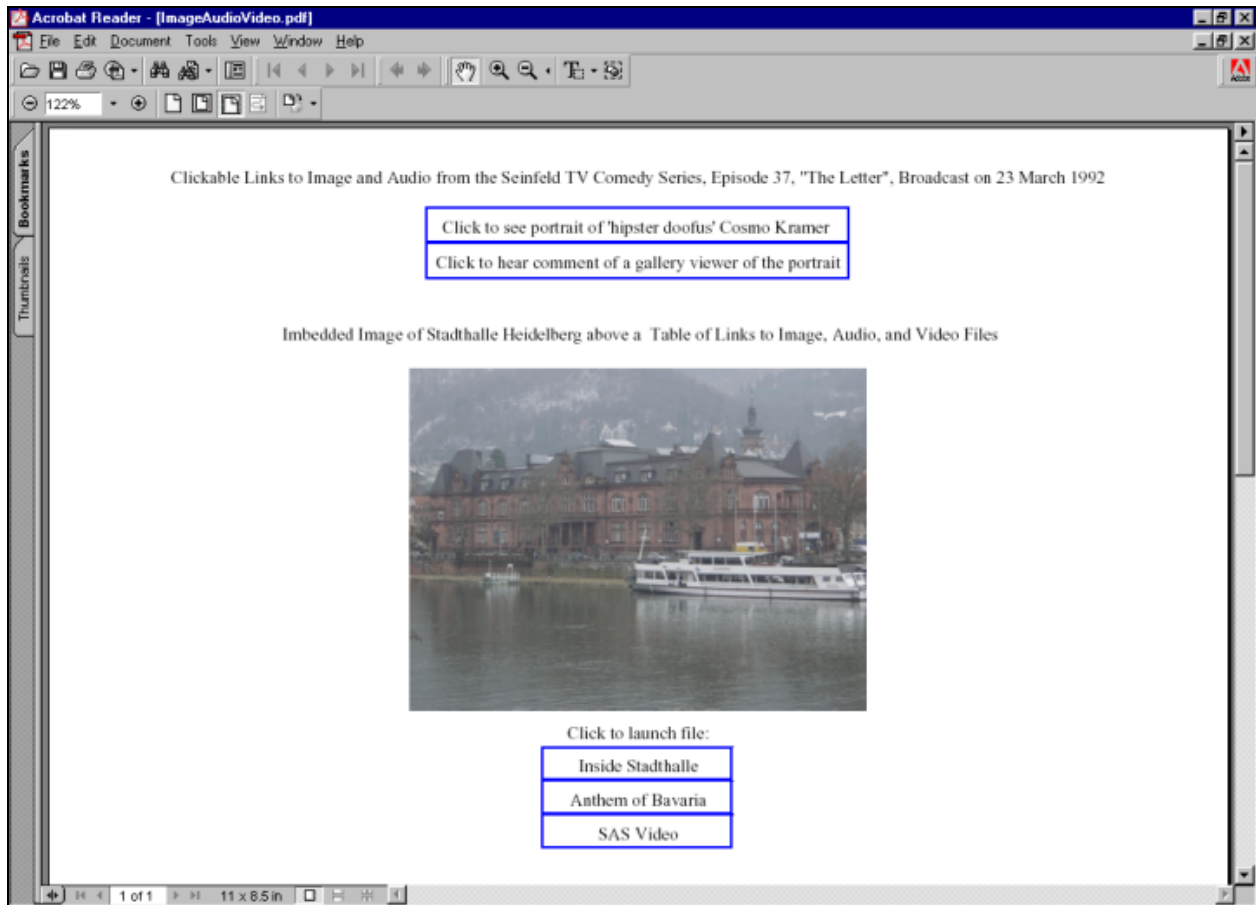
options nodate nonumber orientation=landscape;
ods escapechar="^";
ods noresults;
ods listing close;
ods pdf notoc file=" . . . " startpage=never;

title1 . . . ; title2 . . . ;
ods pdf text="^S={&TOCfmt URL=""&urlPath./images/CosmoKramer.jpg""}Click . . . ";
ods pdf text="^S={&TOCfmt URL=""&urlPath./audios/brute.wav""} . . . ";
ods pdf text="^S={&TXTfmt}";
ods pdf text="^S={&TXTfmt} . . . ";

proc print data=work.ChoicesForImageAudioVideo noobs label
  style=[preimage="&filePath.\images\photo1.jpg" frame=void];
var medium /
  style(data) =[URL=$AVlnks. font_size=12pt
                cellwidth=4.5 cm just=center cellheight=0.8 cm vjust=middle]
  style(header)=[background=white font_size=12pt font_weight=medium
                just=center cellheight=0.8 cm vjust=middle];
label Medium='Click to launch file: ';
run;

ods pdf close;
ods listing;
```

Here is the result for the “home page”, but no illustrations of the screen images of the linked-to destinations:



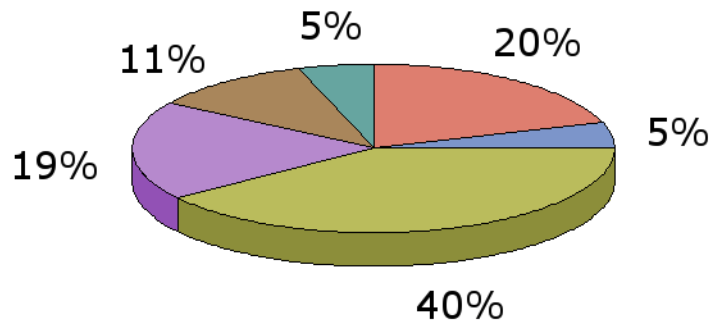
3D

After years of scrupulously discouraging the use of needless and inappropriate 3D, while I was preparing a proof of concept for an executive dashboard, I decided to see what I could do to “glitzify” it a bit, but without effects detrimental to communication.

Before I show you my result, I will try to dissuade you from creating two types of 3D charts shown on the following pages.

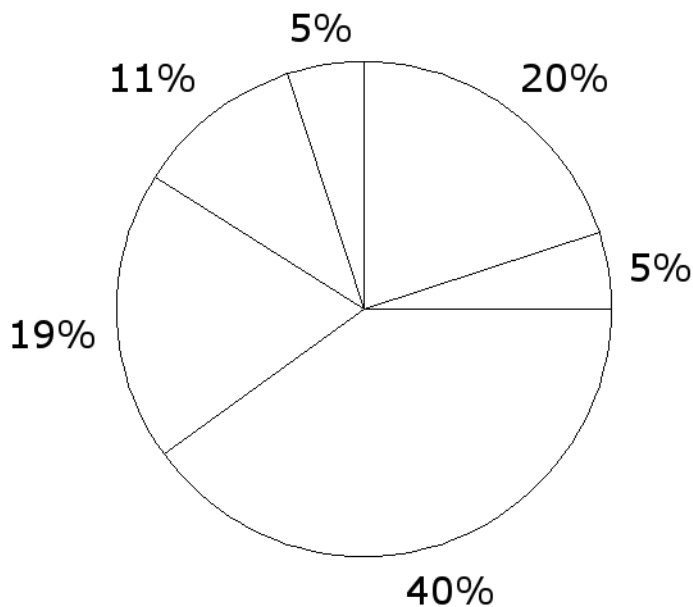
Please compare the two pie charts below. Having compared them, how can you ever justify use of a 3D pie chart again?

**The Unfortunately Popular 3D Pie Chart
Always Distorts the Relative Size of Slices**



**5% slice at rear is twice as big as 5% slice at right
11% slice is almost as big as 19% slice**

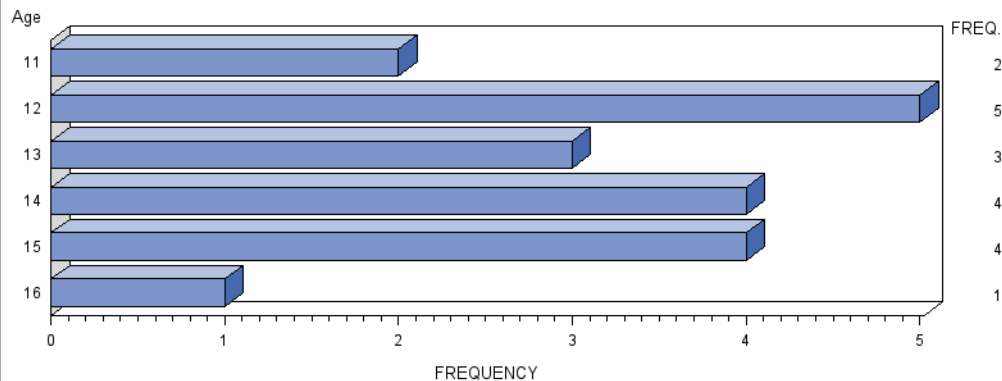
**The Recommended Simpler 2D Pie Chart
Accurately Presents the Relative Size of Slices**



Please compare the two bar charts below. If your manager, your client, or you require 3D, can cylinders without a frame suffice?

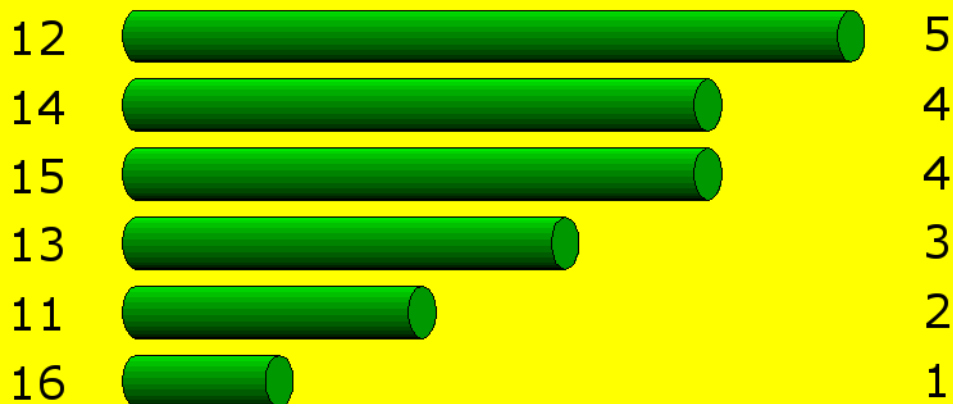
Needlessly Complex Default 3D Bar Chart

Ranked Age Distribution of Students in sashelp.class



The Recommended Simpler 3D Bar Chart

Ranked Age Distribution of Students in sashelp.class



Here is the code to create the recommended 3D bar chart (a cylinder chart):

```

goptions reset=all;
goptions device=PNG;
goptions border; /* Put the graph in a box. */
goptions cback=CXFFFF00; /* Use RGB Yellow for background. */
goptions htext=6 PCT ftext='Verdana';
    /* Specify height and font used for those parts of the graph
       for which you do not make an explicit assignment,
       or for which no direct controls are available in SAS/GRAPH. */
goptions gsfname=replace gsfname=anyname;
filename anyname "C:\MWSUG 2009\V92_HorizontalCylinderChart.PNG";
title1 font='Verdana'
    height=4 PCT ' ' /* Insert space at the top. */
    justify=CENTER /* Force a new line. */
    height=6 PCT
    color=CX0000FF /* RGB blue */
    "The Recommended Simpler 3D Bar Chart"
    justify=CENTER /* Force a new line. */
    color=CX000000 /* RGB black */
    "Ranked Age Distribution of"
    justify=CENTER /* Force a new line. */
    "Students in sashelp.class";
footnote1 angle=+90 font=none height=1 ' ' ;
    /* This is a blank "right-side-note" to increase space at the side of the image. */
footnote2 angle=-90 font=none height=1 ' ' ; /* This a blank "left-side-note". */
pattern1 v=solid color= CX006600; /* Use an RGB Medium Dark Green for the bars. */
    /* Remove all axis paraphernalia: */
axis1 label=none major=none minor=none style=0;
axis2 label=none major=none minor=none style=0 value=none
    offset=(5 PCT,5 PCT);
    /* Offset the horizontal axis to insert extra space between bar midpoint label
       and bar start, and between bar end and bar response value. */

proc gchart data=sashelp.class;
hbar3d Age / freq
    discrete /* Use Ages, not Age subranges. */
    freqlabel=' ' /* Suppress the Frequency column heading. */
    descending maxis=axis1 raxis=axis2 shape=cylinder noframe
    noplane /* NOPLANE must be used in AXIS statements instead for 9.1.3 */
    coutline=CX000000; /* override default bar outline color which is black */
run; quit;

```

Version 9.2 Notes:

Bar label text now can be as long as 256 characters, which might be more than necessary, and would be impossible to read with any font small enough to fit on a typical display. The old limit of 32 characters is adequate for this bar chart.

In Version 9.1.3, the walls of a 3D bar chart can be removed only by putting the NOPLANE option on the vertical and horizontal AXIS statements, whereas in Version 9.2, it can be put on the HBAR3D statement directly.

In Version 9.1.3, the COUTLINE option had no effect for SHAPE=CYLINDER. In Version 9.2, the COUTLINE option now works as expected. However, since the 3D effect for cylinders is achieved by painting the round wall of the cylinder with stripes of the chosen color, progressing from the true color to its darkest shade, which is actually black, if you use a COUTLINE (Outline Color) other than black, the result is an undesirable visual effect.

Hardcopy

There are three aspects to this topic: (1) how to get SAS output to a printer and make it look the way you want it to look; (2) special concerns with respect to printed output; and (3) design considerations for hardcopy.

Printing SAS Output

It is possible for a SAS program to route output directly to a printer. Since I have not had a need to do this in over ten years, it is best for me to refer you to Reference 7. Possibly support.sas.com can help. This problem is both printer dependent and operating system dependent.

An alternate approach is to develop your output for the RTF or PDF destination, proof it on your own PC, print it from your own PC to your preferred printer, revise your SAS code if needed, and print again, going over and over through this loop till you are satisfied. If you are delivering the finished hardcopy to the intended users, this will suffice. If you are sending the intended users an electronic document to be printed by the users, you must use PDF. RTF documents might lay out on the displayed page differently for different users, and might print differently for different users (possibly also differently from how they are displayed). PDF documents should look the same and print the same for all users.

Printed Color

If you are producing color hardcopy, you should never assume that what you see on your PC screen is what you will see on hardcopy.

Printed color does not look the same as electronically displayed color. Displayed color shines at you. Printed color is viewed via light that reflects off the printed page.

Furthermore, different printers will present the same color differently.

Finally, choice of paper will affect the appearance of the hardcopy, even for black on white. The brightest white and most opaque paper you can find will yield the best looking hardcopy. A heavy paper is inherently more opaque than a flimsy paper. Color looks best on high gloss paper. Be aware, however, that a high gloss paper will display any bends, kinks, or dents prominently if the paper is not sturdy enough to withstand handling.

What to Put on a Page (Hardcopy or Otherwise)

A long, long time ago, when I was agonizing over preparation of a report destined for executive management, Kenneth J. Wesley counseled me thus: “If it doesn’t fit on one page, they won’t read it.” Well, though I have had to produce multi-page reports, I did take this advice to heart in one sense.

Any listing report, unless it is a comprehensive reference that must cover every entity in the class of entities under study, should be limited to one page.

This is the basis of my report design concept of the Top N and Top PQ% reports. Both reports embody these principles of communication effectiveness:

- Show them what’s important.
- Let part stand for the whole.

I think that “Show . . .” is mine, but that James White said “Let . . .” about fifteen years ago, in a newsletter (title and issue unknown) for people interested in printing.

The principles are implemented by ranking and subsetting the data.

The Top N report is hardly an astonishing bit of creativity on my part. You will see, however, that I do include a little distinctive added-value feature in my Top N report. It has been my empirical finding that a table on one printed page can often, if not usually, cover at least 80% or more of what’s important. When delivering only part of the information to the viewer, it is useful to let the viewer know how much of the picture you have presented. Therefore, my Top N report always includes a programmatically dynamically created subtitle that explains what percent of the total of the measure of interest has been covered. For example, in a report on the Top 10 Products By Sales Dollars, you can list what their subtotal is as a percent of total dollars for all sales.

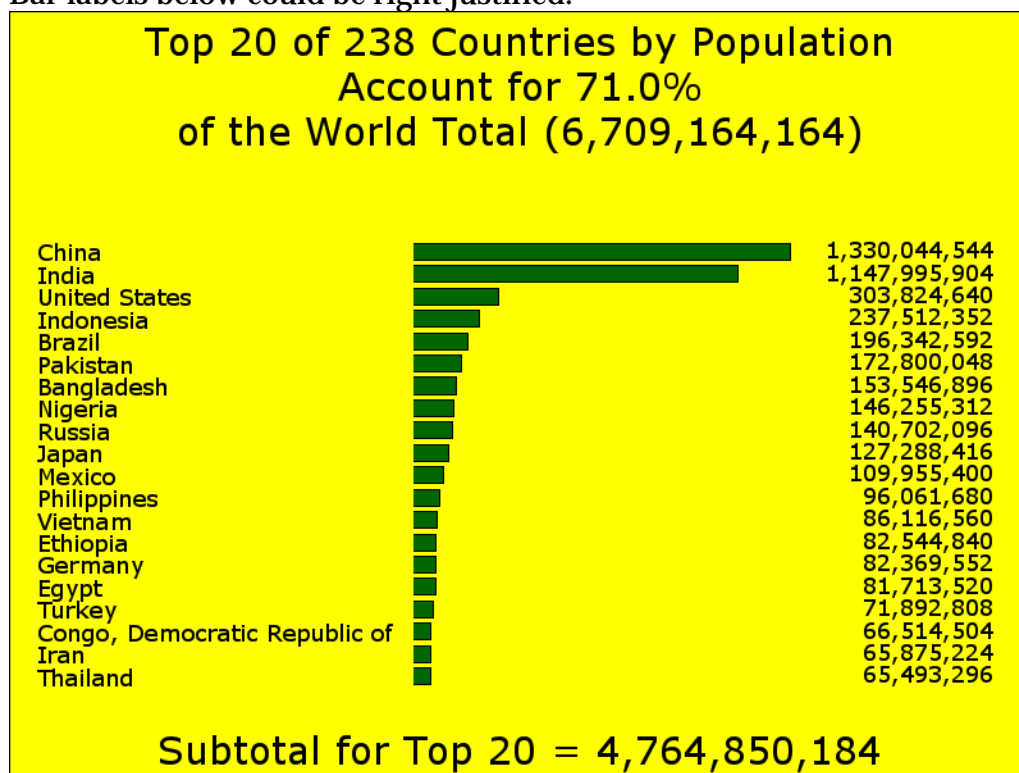
The Top PQ% report turns the reporting objective around. It lists as many of the entities as are needed to account for at least PQ% of the total measure of interest. For example, it can answer this question: “Which product lines accounted for 90% of total sales?”

A third subsetting report concept is the simple Cutoff Report. It applies a filter to show only those entities that satisfy the criterion. For example, it can answer this question: “Which product lines each generated at least \$1M in sales?”

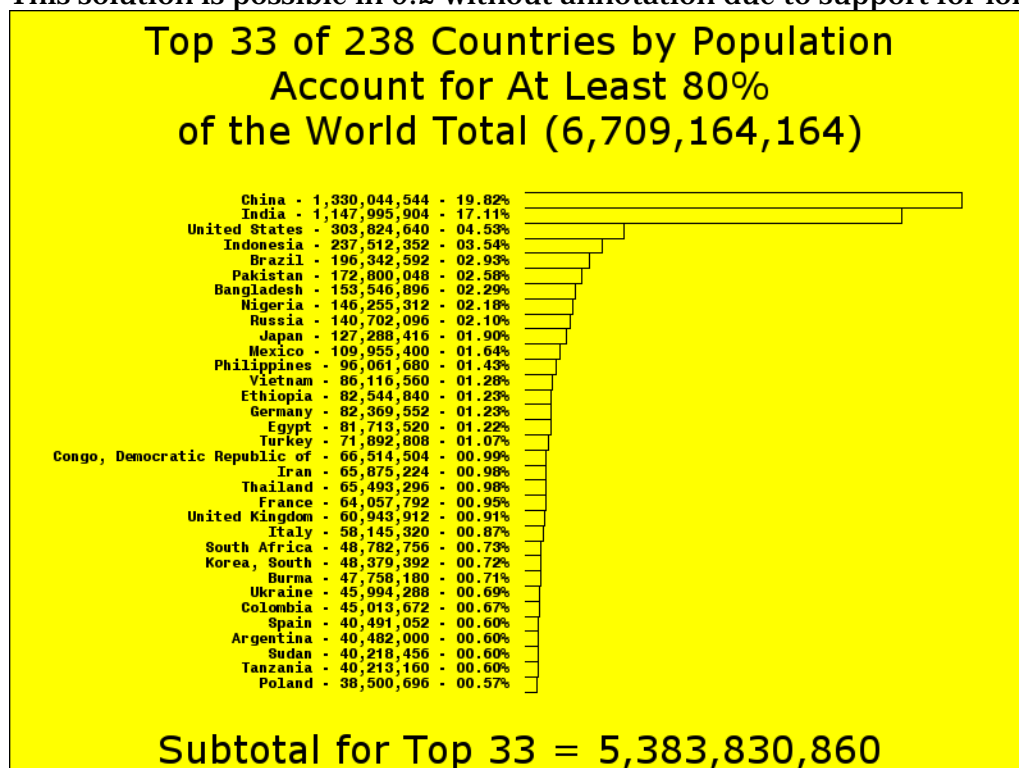
These ranking and subsetting concepts are not unique in their applicability to hardcopy reports. They work for listings in any format. They have applicability to the web, where the vertical space is smaller, and where forcing people to scroll is best avoided if at all possible. They have been exploited by me in horizontal bar charts.

On the following page, see examples of Top N and Top PQ% reports as bar charts.

Bar labels below could be right justified.



This solution is possible in 9.2 without annotation due to support for long bar labels.



Code for these is in Reference 9. **For the Top 80% chart, revise the code with:**
axis1 value=(font='SAS Monospace/Bold' justify=right); hbar ... maxis=axis1;

Excel

A frequent fate of data prepared with SAS software is delivery as a spreadsheet. Every data user knows how to use Excel, and can manipulate the data further to her/his satisfaction. And the initial data recipient can pass it on to someone with the same skills.

From the standpoint of communication effectiveness, there is not a lot that I have to say, except that if you are choosing to “traffic light” your data, or to put a fill behind the data in the spreadsheet cells, please be aware of two things:

1. The commonest color blindness can not distinguish red and green. A viewer of color coding can probably guess that, if red is bad, blue should be interpreted as good.
2. To assure readability of cell content, use background colors with high contrast versus the text/numbers. Black is readable over white, yellow, and light shades of other hues.

For more about communication with color, see Reference 4.

When it comes to getting the data from SAS to Excel, there are (too) numerous options, for which there is no all-in-one-place reference.

For simple needs, please see my MWSUG 2009 paper “Visual + Detail = Effective Communication: Web-enabled Graph + Spreadsheet, Using SAS/GRAPH, ODS, PROC PRINT, and Excel”, which is available in these same conference proceedings.

For more elaborate needs, I rely on Dynamic Data Exchange. DDE is an old Microsoft technology, which continues to work. In effect, it allows you to do, using SAS program statement commands to Excel, anything that you could do with mouse and keyboard manually inside an Excel session.

For my most comprehensive publication on DDE (which contains citations of related work by other authors), please see my MWSUG 2007 paper “SAS-with-Excel Application Development Tools and Techniques, Expanded Edition”. Find it on the web at www.lexjansen.com/mwsug/2007/ApplicationDev/A01-2007.pdf.

The only drawback to DDE is that it is clumsy to work with when SAS is running on a remote server, rather than your own PC. A SAS DDE program must open an Excel session to which it is sending commands. At normal end of processing, the SAS program closes the Excel session. However, if the SAS program makes an error, the Excel session remains stuck open. You are not necessarily allowed to log in to a remote server to close the Excel session. Furthermore, your erroneous SAS program might be hung. To kill a SAS program on your own PC is straightforward. If you are running your SAS program via SAS Enterprise Guide, you might have a problem trying to get it to terminate. For a tool to kill a SAS program on a remote server, please see Reference 10. The custom SAS macro tool provided in that reference can be revised to also allow you to kill an Excel session on the remote server. If you have a question on how to do that, please email me.

Bibliography of Related Work (By This Author, Unless Otherwise Noted)

1. "SAS Graphs for a BlackBerry, iPhone, or Other Small Email Screen: Extreme SAS/GRAPH and the Necessity and Power of Simplicity", *Proceedings of SAS Global Forum 2008*, SAS Institute Inc. (Cary, NC), 2008. Find it on the web at <http://www2.sas.com/proceedings/forum2008/034-2008.pdf>.
2. "SAS Graphs for BlackBerry, iPhone, PowerPoint, Word, Web/HTML, PDF, or RTF", *Proceedings of Midwest SAS Users Group Conference 2007*, Midwest SAS Users Group (Des Moines, IA), 2007. Find it on the web at www.lexjansen.com/mwsug/2007/DataVisualization/D03-2007.pdf.
3. "Web Communication Effectiveness: Design and Methods to Get the Best Out of ODS, SAS, and SAS/GRAPH", *Proceedings of the Twenty-Eighth Annual SAS Users Group International Conference*, SAS Institute Inc. (Cary, NC), 2003. Find it on the web at <http://www2.sas.com/proceedings/sugi28/130-28.pdf>.
4. "Communication-Effective Use of Color for Web Pages, Graphs, Tables, Maps, Text, and Print", *Proceedings of the Twenty-Ninth Annual SAS Users Group International Conference*, SAS Institute Inc. (Cary, NC), 2004. Find it on the web at <http://www2.sas.com/proceedings/sugi29/176-29.pdf>.
5. "Getting Started with, and Getting the Most out of, SAS ODS PDF: No Mastery of PROC TEMPLATE Required", *Proceedings of the PhUSE 2005 Conference*, Pharmaceutical Users Software Exchange (Heidelberg, Germany), 2005. The paper extends some work I did with Deborah Skinner. Find it on the web at <http://www.lexjansen.com/phuse/2005/ts/ts05.pdf>.
6. "Get the Best out of SAS/GRAPH and ODS", *Proceedings of SAS Global Forum 2007*, SAS Institute Inc. (Cary, NC), 2007. Find it on the web at <http://www2.sas.com/proceedings/forum2007/228-2007.pdf>.
7. Lauren E. Haworth, Cynthia Zender, and Michele M. Burlew, "Output Delivery System: The Basics and Beyond", SAS Institute Inc. (Cary, NC), 2009.
8. "The Power of Pictures and Paint: Using Image Files and Color with ODS, SAS, and SAS/GRAPH", *Proceedings of the Twenty-Eighth Annual SAS Users Group International Conference*, SAS Institute Inc. (Cary, NC), 2003. Find it on the web at <http://www2.sas.com/proceedings/sugi28/013-28.pdf>.
9. "Show Them What's Important with Communication-Effective SAS Graphs: Solutions for a Finite Work Day in an Era of Information Overload", *Proceedings of SAS Global Forum 2009*, SAS Institute Inc. (Cary, NC), 2009. Find it on the web at <http://support.sas.com/resources/papers/proceedings09/220-2009.pdf>.

(Bibliography Continued)

10. “Using SAS to Manage, Monitor, and Control the SAS BI Server: User-Developed Custom Tools for the SAS Server Administrator, User, or Manager”, *Proceedings of SAS Global Forum 2009*, SAS Institute Inc. (Cary, NC), 2009. Find it on the web at <http://support.sas.com/resources/papers/proceedings09/274-2009.pdf>.

About the Author

A SAS user since 1978, Dr. LeRoy Bessler has shared his knowledge and experience with other users at conferences throughout the USA and in Montreal, London, Heidelberg, and Dublin. At Assurant Health, he supports the SAS users, server, software, and data, provides internal consulting and occasional training, and gets to do some programming—especially if it involves SAS/GRAPH, ODS, highly formatted Excel reporting, web information delivery, automated dynamic HTML-formatted email, or tools to support the SAS server and its users. LeRoy is a regular contributor to *VIEWS News*, an electronic web newsletter of the VIEWS International SAS Programmer Community. He has organized one regional and numerous subregional SAS users conferences in the north central USA since 1989. Next year, Dr. Bessler will serve with Craig Wildeman and Alexandra Riley as a co-chair for the MidWest SAS Users Group Conference 2010 in Milwaukee.

Contact Information, Etc.

I am always interested in user suggestions, comments, and questions about visual communication with SAS, SAS/GRAPH, and ODS.

LeRoy Bessler PhD
Le_Roy_Bessler@wi.rr.com

The Power to Show™
The Power of Simplicity™

Zum sehen geboren,
Zum schauen bestellt.
—Goethe

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other product and brand names are trademarks or registered trademarks of their respective owners.

The Power to Show and The Power of Simplicity are trademarks of LeRoy Bessler PhD.