Paper 130-28

Web Communication Effectiveness: Design and Methods to Get the Best Out of ODS, SAS®, and SAS/GRAPH®

LeRoy Bessler, Bessler Consulting & Research
Fox Point, Milwaukee, Wisconsin, USA, bessler@execpc.com

Abstract

Web Presence is no guarantee of Web Accessibility or Web Usability. Communication effectiveness on the Web requires you to: (a) get your message *found* (you want those search engines to find your web pages); (b) deliver your message *quickly* (download time is the Number One web user concern); (c) assure that the receiver sees the *same* message that you, the sender, saw when you created it (this is NOT automatic, since unexpected things can happen with, e.g., colors and fonts); and (d) have the meaning of your message *understood* (superfluous dimensionality and other paraphernalia routinely introduced for graphs or tables can actually obstruct or delay, rather than enhance or accelerate, communication and comprehension). This is a tutorial about best technical practices, key design ideas, and The Power of Simplicity for publishing information on the Web. This tutorial is suitable for all skill levels. Design principles and ideas are software independent.

Introduction

Some TV ads make you remember action and images, but not what was being promoted. The time and attention of the "audience" for your web pages are precious, as are your time and effort to create the web pages. Software and hardware are power tools, but defaults produce lots of sub-optimal results quickly. With the web, the whole world is watching.

Web pages can be designed to inform and to influence, not to impress, so that they deliver the messages and the meaning in your data with images that are easily and quickly interpreted, and that have memorable *significance*. Graphs can be constructed to deliver overview impact, to accelerate decision-making, and—for situations where it is needed—precise detail, to facilitate reliable decision-making. Tables can be constructed to designed to accelerate information-finding, rather than to adhere to the old spreadsheet paradigm of a big pad of ruled paper.

There are three companion papers in these SUGI 28 Proceedings, one published with Dr. Francesca Pierri. They contain illustrations and coding details on related topics in visual communication effectiveness with SAS software: (a) use of images and color; (b) SAS/GRAPH; and (c) trend charts (for web-based Enterprise Performance Reporting). All are relevant to web page design and construction. I have tried to minimize content overlap. Please see those papers for more information. This presentation includes web page demos, also available via email request. This paper contains a few web pages, but no demo descriptions.

The Power of Simplicity

Simple Downloads Faster. Dilbert (Scott Adams' cartoon character) said, "I made a study of Internet use in the workplace. The time spent waiting for web pages to load has wiped out all the productivity gains of The Information Age." The Number One usability concern of web users is download time.

Simple is Safer. While writing this paper, I received a Critical Windows Update notice from Microsoft Corp, about a security vulnerability that involved possible attacks on web users' computers from web sites that use a fancy way to produce web pages. Software built with mature and simple technology tends to be reliable. Changes and enhancements may come with new defects, and new vulnerabilities. The possible is not automatically necessary, nor what is always desirable.

Simple is Accepted. Many web users—even if they have a web browser with all the latest bells and whistles—deliberately disable special features. Just because you want to communicate with them, they are under no obligation to accept your terms. If your web pages don't display for these web users, it's your fault, not theirs.

Simple Is Focused and Communicates Better. Simple web pages accelerate understanding, inference, and decisions. (See Figures here and supporting code in the Appendix, and References 1, 3, 4, and 5.)

Web Communication and Web Accessibility

Communication Is the Mission

Unless it's for entertainment, the main mission of a web page is communication of information, not decoration. Even if it is for electronic commerce.

Communication Is Effective When:

- the Receiver Gets/Finds the Message From the Sender;
- the Receiver Gets the Same Message As was Sent; and
- the Receiver Gets the Message Quickly (enough).

Failures in communication are always a failure of the sender.

Get Your Web Site Found: Achieve Genuine Web Presence

Some web sites have a captive audience. It may be a web site on the corporate, university, or other organizational intranet. It may 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. And we live in an era when everyone (with sufficient time, or sufficient money to hire someone's time) deems it essential to establish "a web presence".

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 its various uses, where No. 1 may be the most important to you:

1. captured by search engines
2. default text for Internet Explorer Favorite or Netscape Bookmark
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 skeleton 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=.)

Probably the best resource for more information about META tags is www.vancouver-webpages.com/META/. 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. (The home page for this web site uses frames. You can also inspect the source for each frame by use of the right mouse button and its View Source option.)

Despite the richness of possible uses for META tags, ODS currently supports only a maximum of 258 characters across all tag definitions. HTML takes an unlimited number of META tags. (Some of the possibilities are: keywords, description, author, copyright, publisher.)

For how to define multiple tags in the ODS METATEXT parameter, send me an email. If you need to exceed the 258-character limit, I may be able to provide an automatic ODS postprocessor. The HTML META tag *does* support an unlimited number of NAME= CONTENTS= pairs.

Web Resources for Search Engines

For tips on how to get your web site found, see:
www.searchenginewatch.com.
For tips on how to make your own searches more effective, see:
www.searchengineshowdown.com.

Their Web Window to the World (and Your Web Pages)

The most recent report I have seen said that the commonest resolution on PCs is still only 800 X 600. If it increases, or has increased, it is at, or will be, probably no more than 1024 X 768. A non-trivial fraction is used by the web browser (and a little bit is used by Windows itself). The remainder is called "live space". Use it wisely. The most conservative design would be for a resolution of 640 X 480.

Your Web Page Viewers Don't Really Want to Scroll

Try to design and build your web pages for FULL view on the smallest probable screen that will display it. The live space varies for PC vs. Mac, Netscape vs. Internet Explorer. For more information about live space, see also Reference 2.

Vertical scrolling is tolerable, but not preferred. If your web page absolutely requires scrolling, put the most important information at the top of the page, and the least important at the bottom.

Horizontal scrolling not only is disliked by most web viewers, but also can frustrate effective viewing. Requirement for scrolling in both directions on the same page is unacceptable. However, there are exceptions. A large complex map cannot be displayed on the screen of any desktop or laptop PC, no matter how big the screen. A very large dense plot would be another exception. For a good example of the latter, see "Visualizing Patterns with Scrollable Web Graphics" by Eric C. Brinsfield and Caroline C. Bahler, in *Proceedings of the Twenty-Seventh SAS Users Group International Conference*, SAS Institute Inc. (Cary, N.C.), 2002.

Focus Your Web Page

Normally deliver only one table, graph, or composite per web page, if possible. If you can fit multiples on one screen, this suggestion does not apply. However, a scrollable multi-image, multi-element page can confuse the viewer. Also, if you click on a hyperlink whose target is really just part of a long page, and then print what you think is a small package of information about that topic, it is not a welcome experience to find 10 or 20 pages of output on your printer, with most of it being irrelevant to your interest.

Avoid Anti-Focus When Using an ODS Table of Contents

While testing an early edition of a custom ODS Table of Contents Style for Reference 4, I had a confusing experience. I clicked on the last entry in the TOC index. What appeared at the top of my screen was *a different entry*. I was puzzled by this until I realized I was looking at the second-from-last entry in the TOC, but had, in fact, been taken to the entry of my interest, which was at the bottom of my screen. I had my 20-inch monitor display set to resolution 1600X1200. Since I believe in simple focused graphs and tables, the entries presented by the TOC were small and compact, and my screen held more than one TOC entry at a time.

> One Table of Contents item per web page prevents confusion. And the web user of your package of information organized, managed, and presented via a TOC can print one item at a time.

When you build a Table of Contents package of web pages with ODS, you first define the FRAME file and the CONTENTS file. Between that definition and the ODS HTML CLOSE statement, you put all your code to create the elements to be indexed by the TOC. You are not required to create a specifically named BODY file for each element to be supported by the TOC, even if created by using different PROCs. In some cases, you could be using BY processing for convenience and concision, or other repetitive processing. On the ODS HTML statement, you can use the NEWFILE= parameter to force the different elements into separate web pages. The default value is NONE, which dumps everything into one big BODY file—usually not good. Other options include OUTPUT (new file for each output object created by a PROC), PAGE (new file for each page created by a PROC, but this does not pay attention to page size), and PROC (new file for each PROC Step code block). To achieve separation for BY processing, which is a single PROC invocation, you need to use either PAGE, or the new option BYGROUP. Figure 1 shows the ODS Default Style Table of Contents. Figure 2 shows a recommended custom TOC. Code for both is in the Appendix.

Navigation Alternatives

For SUGI 27, Francesca Pierri and I comprehensively demonstrated the range of web publishing and web linking possibilities using ODS, SAS, and SAS/GRAPH. See Reference 4. We compared the SAS/GRAPH WEBFRAME driver, the ODS Table of Contents, and what we call our "CrossLink Method". The Table of Contents provided by the ODS Default Style needed, and got, various functional and appearance customizations. What you see here in this paper is an extension of that work. The WEBFRAME driver provides a thumbnail index (rather than the text index of a Table of Contents, but supports only graphic output—any table must be created with PROC GPRINT or GSLIDE, which are not necessarily convenient or best for that purpose). CrossLinks take you beyond the wide range of drill-down possibilities in SAS/GRAPH and in ODS tables. We use the LINK= option for FOOTNOTE or TITLE statements supported by ODS. As of Release 8.2, LINK= cannot be used in SAS/GRAPH output, even if the graph is web-packaged with ODS. We circumvented that by appending a tiny empty table with PROC PRINT below graphic output on the web page. With our CrossLink method, you do not need to sacrifice screen width to support navigation between web pages that are not linked by drill-down. **The presentation includes demos of the default Table of Contents, the custom Table of Contents, the CrossLink Method, and a web application (from Reference 3) that is a hybrid of Table of Contents and CrossLink.**

Do Your Web Pages Actually Work As You Intend?

The World Wide Web Consortium provides a facility to validate your web pages (and to evaluate conformance with their own standards) at validator.w3.org.

> Do not assume that your web pages, which work as desired with Internet Explorer, will work the same way with Netscape, or that they will work the same with all versions of a particular browser. Unless you have a captive browser-specific audience, avoid browser-specific web design.

Can They See Your Web Pages?

The power of the web is in its universality.
Access by everyone, regardless of disability, is an essential aspect.
- Tim Berners-Lee

Resources on Accessibility for Impaired Users

- See the Web Accessibility Initiative at www.w3.org/WAI/.
- For ODS, go to www.sas.com/service/techtips/ts_qa/ods508.htm.
- If interested in using native html, see "SAS User Documentation: Web Page Design Made Easy" by Bruce Gilsen & Scott Hoenig, in *Proceedings of the Twenty-Seventh SAS Users Group International Conference*, SAS Institute Inc. (Cary, N.C.), 2002.

To have your web pages evaluated for compliance with standards for accessibility by people with disabilities, go to bobby.watchfire.com.

ALT Text: Enhancing Accessibility for Impaired, and All, Users

Those (usually small) boxes of text that pop up when you rest your mouse on an image, or on the place on the web page where an image is destined to appear when it completes download, are called ALT text. "ALT" is short for "Alternative", as in "Alternative to the Image".

It is a recommended practice for web page developers to provide ALT text for all images. It has two benefits. Vision-impaired users can use web-page-reading software that converts the ALT text to audible speech. From ALT text, unimpaired web users can find out something about the

image file that they are waiting to download until the picture is fully painted. Some viewers may skip the wait and move on to some linked web page, or may focus on a different aspect of the current web page. Furthermore, with ALT text you do not need to sacrifice any live space on the web page to provide a "hard label" for the image. You can be as concise, or as verbose, as you like with ALT text.

In HTML, ALT= is a parameter that can be used with the IMG tag. In ODS, ALT= is a parameter that can be used with the PREHTML= and POSTHTML= options of the STYLE statement.

It is also possible to provide ALT text with SAS/GRAPH for various parts of a graphic image, regardless of whether or not they are hyperlinked to other web pages. It can be assigned with the HTML parameter available for commonly used graphic PROCs. It can also be assigned with the HTML variable available for many Annotate functions. Here is an example of how you assign ALT text, after you have determined where to use the HTML (and/or HTML_LEGEND, if your graph has a legend) parameter in your SAS/GRAPH PROC:

```
html= (or html_legend=)
      ' alt="describe this area/point"
        href="OtherPageName.html" '
```

You need not use href= (i.e., to assign a hyperlink) in order to use alt=.

In Reference 2, there is an example of using Annotate to imbed an image in a graph and to assign ALT text for (and a hyperlink from) the image. Also in Reference 2 is an explanation of how you can use Microsoft Word to add ALT text to any image file that you might already have.

For some graphs, such as a high-point-density or multi-line trend plot, it may be impossible to annotate with precise values. If your web page viewers are not expected to need a printable record, ALT text can avoid forcing them to download a separate companion look-up table (which might be too large to fit on the same web page as the graph).

Communicate with Color (Find details in Reference 2.)

1. The commonest color blindness cannot distinguish red and green.
2. Color contrast between text and background is essential.
3. If using several different shades (i.e., degrees of lightness) for a constant hue, no more than five shades can be reliably distinguished. Depending on the application, you may be able to augment the palette with White or Black.
4. Always use "browser-safe" (a.k.a. "web-safe") colors.
5. Use RGB color codes, not "SAS Predefined Color Names", and not color names from the SAS HTML Color Registry.

For details and illustrations of all the above, please see Reference 2. For Point 2, also see Figure 3 and supporting code in the Appendix.

- Use Color to Communicate, Not to Decorate.
- Color does not improve a bad design.

Communicate with Text and Fonts

Pulitzer's First Rule: "Make it brief so they will read it."

Please keep text horizontal. That's the way we like to read. And that's the way our eyes are oriented if not sleeping. Do not permit software to stack the letters of a report column label. Do not direct software to apply a rotated label to the vertical axis of a graph just because it will fit nicely. It should be possible to create a sufficiently informative title or subtitle for any graph so that axis labels would be superfluous.

No Blinking: It's annoying. And it frustrates web-page-reading software for the visually impaired.

Make your web page title your headline. If you are using the web page to persuade and/or reveal, don't be reluctant to tell the viewer what you know (or you think) it implies and/or shows.

Use Sparse Text to make the web page talk. Be sure every letter or number must be there. Superfluity detracts from the real message.

Use high contrast (well exemplified by Black with White or Yellow). On light backgrounds, colored text or line must be sufficiently thick. Never use Black on Dark or Medium Blue, Yellow on White, etc. **For a utility to check readability of combinations of text and background color, please see Figure 3 and supporting code in the Appendix.**

CHOOSE your background. The ODS default web page background of gray is boring, and does not enhance readability of foreground text. My recommendation: **use one solid color**. Textures and backgrounds vary the contrast with foreground text, impairing readability, besides being unnecessary and sometimes actually annoying. The presence of fancy backgrounds on web pages too often is simply a case of confusing the possible with the necessary.

For emphasis, first consider use of **Bold**, or *Italic*, not color. Do not use Underline. On the web, Underline is a standard highlight for hyperlink text. If using ALL CAPS for emphasis, use it SPARINGLY**.** ALL CAPS is hard and slow to read. To convince yourself, prepare and read a long paragraph in ALL CAPS, and compare that with Mixed Case.

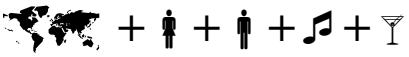
When creating graphs for your web pages, check your SAS log for a note that the SIMULATE font has been substituted for your requested font. (Still as of SAS Version 8.2, you are NOT always notified.) Fix the problem, if you can. A common error is a misspelled font name, in which case the SAS log seems to always notify you of substitution. For a sample of this unwanted and undesirable font (or any other SAS/GRAPH font specifiable below with **name**=), use:

```
proc gfont name=SIMULATE nobuild
     SHOWROMAN HEX H=2; run; quit;
```

Wherever you use f=NONE, or fail to specify f= where applicable, in your SAS/GRAPH program, the device driver uses its default font. If, when using a default font, you specify a height other than one cell (i.e., not h=1), then SIMULATE will be used, but you will not be notified. Furthermore, be aware that some drivers (e.g., the EMF driver) do not have a default font—SIMULATE is used without notification. GOPTIONS SIMFONT= assigns the default font. The software as shipped has this set to SIMULATE. You might wish to override it.

Generate Font Samples. Using PROC GFONT as above, you can easily produce samples of any SAS/GRAPH software font. You will find things that you did not know are there, but which can be used by defining a text string (in a SAS/GRAPH application) with hexadecimal codes—e.g., '03'X yields a solid square. *Unfortunately, PROC GFONT does not work with Windows TrueType fonts.* I have a font utility in development just for that purpose. I call it BFONT. But **to create samples of** only **keyboard characters, I have provided code in the Appendix. See example output in Figure 4.**

Explore Those Fonts. Just as SAS/GRAPH software fonts have more content than letters and numbers, so do the Windows TrueType fonts. Short of using the aforementioned BFONT, you can interactively explore their content with the Insert Symbol window in MS Word. It will display all 256 characters available in each font. Also, look at the more exotic fonts like Webdings, Wingdings, Wingdings 2, etc. You may have some useful application for them. Not a serious use, but for fun, here is a rebus where the icons are Webdings and the plus signs are Times New Roman Bold (the map is 28 point, all others are 20 point):



= "SUG International Kickback Party", an annual SUGI highlight.

Set SAS/GRAPH defaults with FTEXT, HTEXT, CTEXT. Some features of certain SAS/GRAPH charts are not controllable with F=, H=, C=. For them, the above three GOPTIONS parameters for default text font, default text height, and default text color are your only recourse.

Even in your graphs, **use Windows TrueType fonts**. Good fonts include Matthew Carter's creations designed for readability on the screen and the web, Verdana (sans serif—useful for small letters and numbers) and Georgia (serif—useful for titles and prominent footnotes). Besides these two fonts, I have a fondness for Rockwell. All its characters are thickly drawn along their entire contour. I find it very easy to read. In Figures 2-4, all titles are rendered with Rockwell, and all table text and Table of Contents entries are rendered with Verdana. Unfortunately, not all PC's have the Rockwell font. Designing for the web is not the same as designing for a hardcopy paper.

Fonts and Sizing

Georgia is best for large print.
Verdana is best for small print.

Sizing fonts on a graph or in a table:

- Title large
- Main body medium, or small if space constraints
- Footnote large if substantive message
- Small footnotes only if not really expected to be read

NOTE: Any font size assignment will display bigger on a Mac.

Do They See What You See?

Can You Preserve Text Appearance? NO

Fonts used inside graphs on a web page are embedded in the graph file that is part of the web page. So, the appearance of graph text is the same for all web viewers, and the same as that for the graph creator, except for screen resolution differences.

However, text outside graphs on a web page is affected by ODS, by the web browser, and, potentially, by the web browser user. If you define text font sizes to ODS with point sizes, ODS converts those point sizes to HTML sizes 1-7. There are many more than seven possible point sizes. At the user end, the web browser converts those HTML font sizes back into point sizes. Since the ODS-to-HTML conversion was many-to-few, it is improbable that the web browser reversal choice will restore the particular point sizes you chose. To add to the ambiguity, the web browser user has the option to influence how the browser will perform the conversion. Here is how. Open a web page. Click on View, then Text Size. You will see five choices: Largest, Larger, Medium, Smaller, Smallest. Medium is the default. These choices enable the viewer to control the size of the text parts of the web page. Such control can be an aid to visually impaired web users, or can condense the length of web pages for web users who—understandably—dislike scrolling.

Tip: In ODS you can define font sizes with the HTML numbers 1-7, rather than point sizes. Presuming that most web users do not alter the Text Size from the default (many, if not most, web users may not even realize that they have the power), you can increase the probability that "They Will See What You See" if you use those HTML font sizes, and IF they have the same web browser as you have. But there is still the problem of possible font substitution, described below.

You Can and Should Control Font Substitution

For the non-graphic, non-image parts of your web page, the web page viewer's PC may not have fonts you specify in your ODS Style. When specifying fonts in your Style, you should list alternatives. The web browser will use the first font in your list that it finds on the computer.

For web page elements you want rendered with serif fonts, use these: "Georgia, Times New Roman, Times".

For sans serif fonts, use these: "Verdana, Arial, Helvetica".

These three choices in each case are best for, respectively:
modern Windows, old Windows, Mac/UNIX.

Font Inconsistency Within the Same Web Page

You can assign the same font and the same point size for embedded graphic fonts as for the fonts you specify in your ODS Style for the non-graphic part of page. Despite this, you cannot expect such text to look identical. There are multiple reasons.

The first of them is obvious from the discussion above.

Another factor is that the graphic fonts are being rendered in the graphic file by SAS code, not being retrieved from a local disk font library by the web browser on the web page viewer's computer. That text outside of the graph is rendered at the web page viewer's computer.

FORCE Font Embedding for Graphs When Using ODS

When putting SAS graphs in your web page, **always use ODS HTML options GTITLE and GFOOTNOTE**. This forces the TITLE and FOOTNOTE statements to be built inside your graph. This will embed the fonts in your graph. Some Windows TrueType fonts are available on almost any PC, others are less commonly available. Even if you use an "exotic" TrueType font, your choice will be available on every web browser user's computer because its rendering will be inside your graph.

As of Version 8.2 of SAS, the **NOGTITLE and NOGFOOTNOTE options have undesirable consequences**. With these options, your graph titles and footnotes are placed outside of the graph area, and outside of the graph file that is put on the web page. This means that they are rendered with the fonts that you have specified (or taken defaults for) in your ODS Style. In principle, that sounds like an advantage. Unfortunately, SAS/GRAPH reserves space in the graph display area for the titles and footnotes that are NOT put there. This creates strange voids at the top and bottom of the graph display area, especially bad if you have multiple title and/or footnote lines. Not only does this look strange, but also it unnecessarily compresses the graph. As far as I know, there is no relief in sight for this situation in Version 9.

Font Embedding for Tables and Text:
With the SAS System Always Possible, But Not Always Easy

If you have energy and patience, you can, in fact, use SAS to create tables and text for your web pages with embedded fonts. The key is to use SAS/GRAPH to create tables and text panels (slides).

You can use PROC GPRINT for Tables. Whenever making a table with SAS/GRAPH, to assure alignment of decimal positions and decimal points in columns of numbers, you must use fixed-pitch (a.k.a. "fixed-width" or "Uniform") fonts. All the non-symbol SAS fonts have a Uniform version, with suffix U on the font name. The fixed-pitch TrueType fonts are Courier New (predecessor was Courier) and Lucida Console. You can also access the fixed-pitch "SAS Monospace" font in the same way as TrueType fonts.

You can use PROC GSLIDE for text. There is no support for automatic word wrap across lines. This is not a word processing tool. But, for simple blocks of text, formatting with TITLE, NOTE, and FOOTNOTE statements is easy. You can use any font you have available (SAS or Windows TrueType). Most of the SAS fonts suitable for bona fide text (as opposed to only graphs or tables) have bold versions and/or italic versions (suffix B and/or I). TrueType fonts can be used with Bold or Italic, by specifying, e.g., f='Georgia/Bold'. For TITLEs, NOTEs, and FOOTNOTEs you also have access to the BOX option. Do not use the UNDERLINE option. On the web, UNDERLINE is a standard highlight for hyperlink text.

Furthermore, simple tables can be rendered with PROC GSLIDE, perhaps more easily than with PROC GPRINT. But, in neither case can you create (the ODS frill of) a table grid.

Finally, you can use the SAS/GRAPH Annotate facility to do almost anything with text, though it is not necessarily easy and quick.

Putting Active Images on Your Web Pages

When it comes to technology, there is a tendency to confuse the possible with the necessary, and new options with progress. If you have a valid communication objective that requires active parts on your web pages, that's a different situation. But fancy web publishing tools use facilities that may be totally unavailable for some web browser users, or may be deliberately disabled—intentionally—by some who have the option to use them. **A paramount design objective should be web pages that work as intended for as many web users as possible.** There is one active web page feature that will work for everyone in your web publishing audience—Animation. But you should have a communication purpose—one valid animation use is a map to show population distribution change over time. It requires an animated GIF. These have been around a long time. You can build your own with SAS/GRAPH. (I can send you sample code, or you can find some in Reference 3.

Communication-Effective Graphs & Tables

3D is for Three Variables. If the pie chart and bar chart illustrations of 3D versus 2D in Reference 1, which demonstrate outright distortion of data significance and needless complexity, do not persuade you that 3D is inappropriate for two-variable charts, there is nothing I can say. 3D maps are anti-communicative (PROC GMAP's PRISM and BLOCK maps have solids for high response areas hiding those for low response areas), or are impractical (the PROC GMAP SURFACE map).

Focus on the Data. Axis lines, tick marks, axis labels that are obvious from the graph title, etc. should be stripped out. They add nothing. They are a distraction, not an aid to communication. "Let the data talk."

About Tables: Set Your Data Free

Get your data out of jail—no cells. Usually just say "No" to grids. Though there may be situations that warrant a grid, the fact is that this is just a software imitation of the big pad of ruled paper from the days of manual hardcopy spreadsheets. Tables do not inherently need grids (or decorative backgrounds). As shown previously in Reference 3, you can easily improve the appearance of ODS-packaged tables if you use:

```
proc template;
edit styles.Default as styles.SimplerTable;
style output from container /
frame=void       /* no walls around the table */
rules=none       /* no walls between labels and data */
cellspacing=0;   /* no walls between the data */
end; run;
```

For more examples of minimalist, communication-effective graphs and tables for web pages, please see Figure 2, References 1, 3, 4, and 5, and the pie chart on this page. Compare Figure 2 with the alternative of the results of using the ODS Default Style in Figure 1. The use of a colored background and a frame for the table in Figure 2 is not necessarily recommended, but is included to demonstrate options supported by the macro provided in the Appendix to build custom Styles. The macros in the Appendix are an extension of work published in References 3-5.

Show Them What's Important. Rank data in bar charts and pie charts, as demonstrated in Reference 1. On maps, you can supply rank as an annotation. Ranking provides focus, and simplifies communication.

Let Part Stand for the Whole. For tabular data, **the subsetted ranking report** is very effective. (Below is one formatted with Microsoft Word, rather than as a web page.) This report can be done with a SAS macro, previously published elsewhere by me. Its control is the cutoff count (here, set to 10). Its functions, besides producing the ranked table, include dynamically determining the total item count, the grand total of the measurement, and the percent of the total represented by the items listed. The titles retrieve that information as macro variables. Concise tables of key information are always communication-effective. In the scarce screen territory available on a web page, a short table is always very useful.

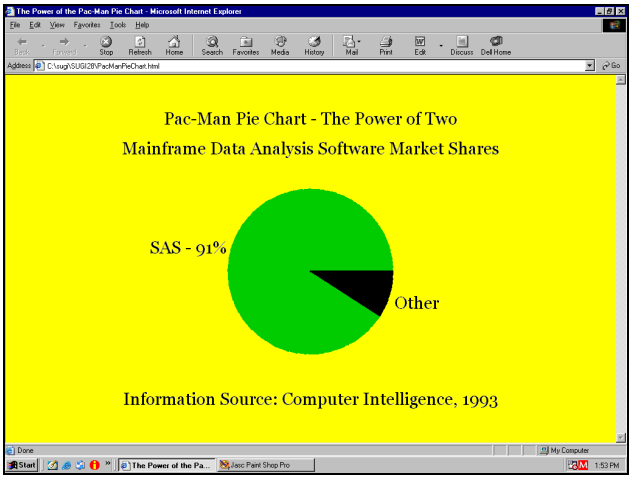
Top 10 of 51* States Sent 66.4%
of SUGI Attendees to San Francisco
(Total SUGI Attendance = 3350)

California	736
North Carolina	447
Texas	178
New York	173
New Jersey	141
Pennsylvania	127
Ohio	110
Illinois	109
Virginia	103
Maryland	99

*51 includes District of Columbia

Use Sparse Annotation. (See References 3 and 5.) Whenever sufficient, annotate y-values only for the critical points of a plot line, and label only corresponding (invisible) x-axis tick marks. For a crowded or crossing multi-line plot, put critical values in the legend. *Critical points* are start, end, maximum, minimum, and points where the rate of growth or decline persistently changes.

When There Is a Very Dominant Share, Use the Pac-Man Pie Chart. A two-part pie chart may seem trivial or silly. But, when the share of interest to your message is tiny or huge, the image is very "impactful" and, therefore, memorable. If needed, supply details for "Other" with a table below the chart, or, on the web, with flyover text and/or drill-down. Unless required for communication, do not blunt the message by splitting the large wedge into little ones that may be as small as, or smaller than, the wedge whose smallness you are emphasizing.



Acknowledgements

My thanks to Chevell Parker, Bari Lawhorn, and others who have helped me to increase my understanding of ODS.

Recent Related Work By the Author

1. "Easy, Elegant, and Effective SAS Graphs: Inform and Influence with Your Data", elsewhere in these SUGI 28 Proceedings.

2. "The Power of Pictures and Paint: Using Image Files and Color with ODS, SAS, and SAS/GRAPH", in SUGI 28 Proceedings.

3. With Francesca Pierri, "Tell Them What's Important: Communication-Effective Web-and-Email-Based Software-Intelligent Enterprise Performance Reporting", in SUGI 28 Proceedings.

4. With Francesca Pierri, "Show Your Graphs and Tables at Their Best on the Web with ODS", *Proceedings of the Twenty-Seventh Annual SAS Users Group International Conference*, SAS Institute (Cary, NC), 2002.

5. With Francesca Pierri, "%TREND: A Macro to Produce Maximally Informative Trend Charts with SAS/GRAPH, SAS, and ODS for the Web or Hardcopy", *Proceedings of the Twenty-Seventh Annual SAS Users Group International Conference*, SAS Institute (Cary, NC), 2002.

Notices

Author Information

Your comments, questions, suggestions, and requests are welcome.

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

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

Simple Web Pages: faster, more reliable, more accessible, more focused, and more communication-effective.

Figure 1. Table of Contents and Scrollable Web Page Using ODS Default Style (See code in the Appendix.)

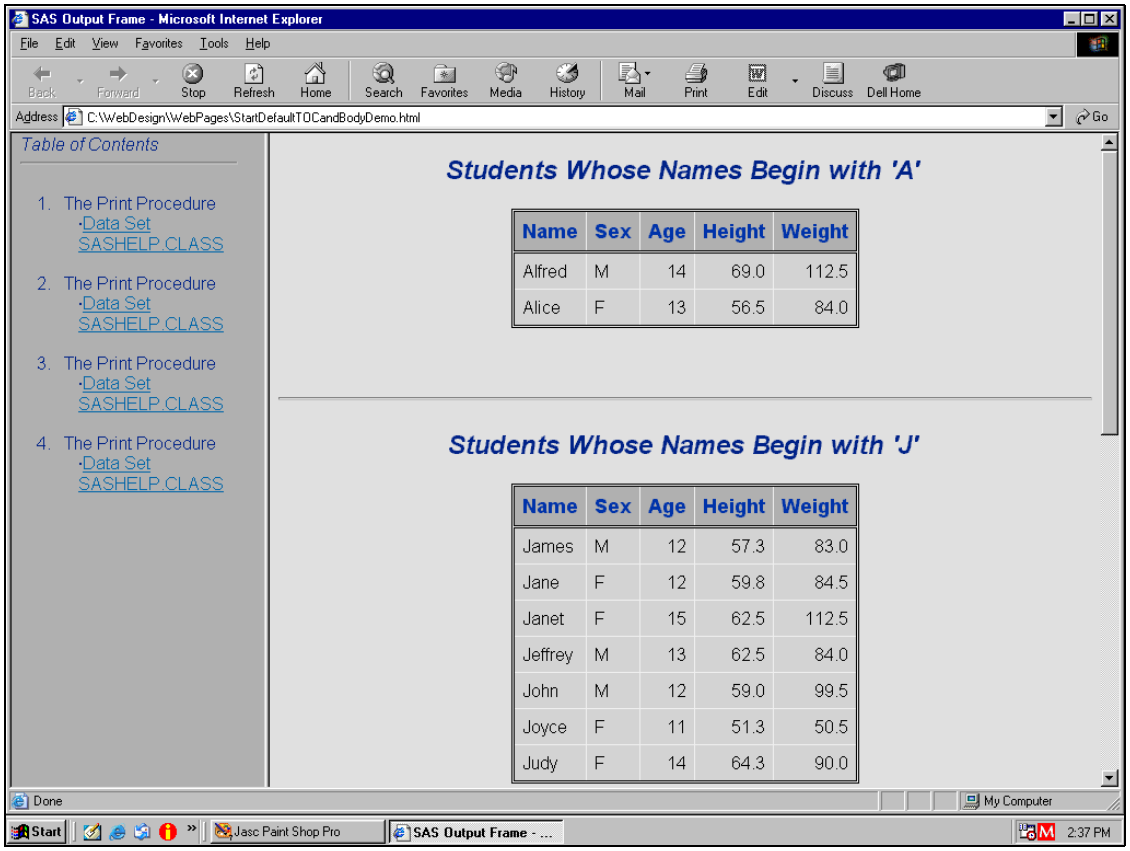


Figure 2. Custom Table of Contents and Custom Separate Web Pages (See code, including macros used, in the Appendix.)

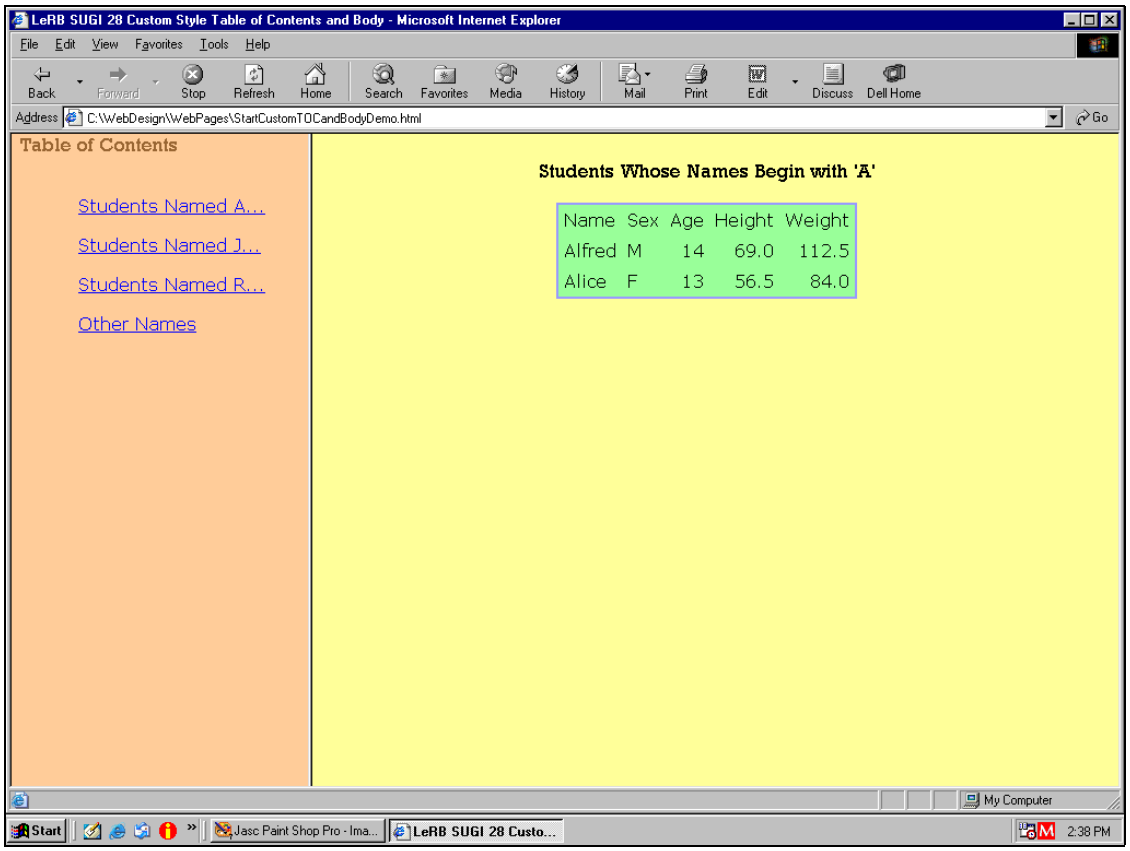


Figure 3. Utility to Check the Readability of Combinations of Text and Background Color (See code in the Appendix.)

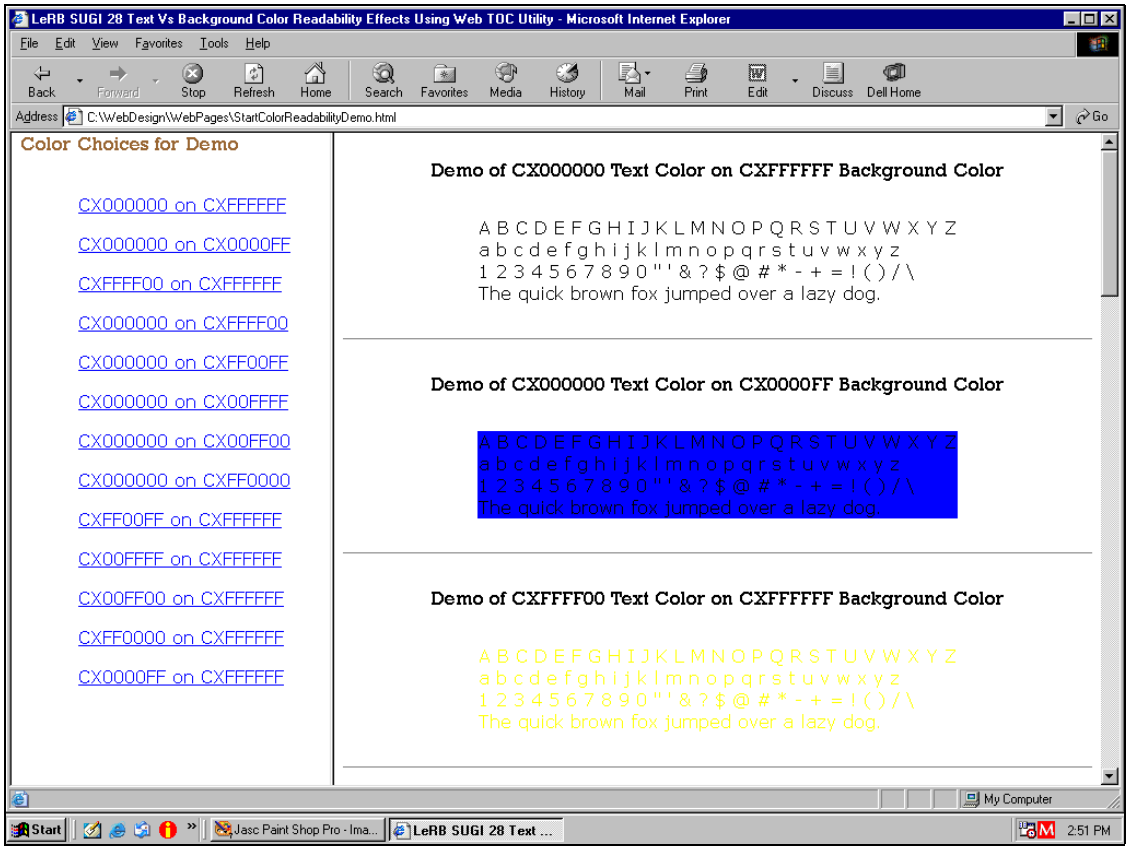
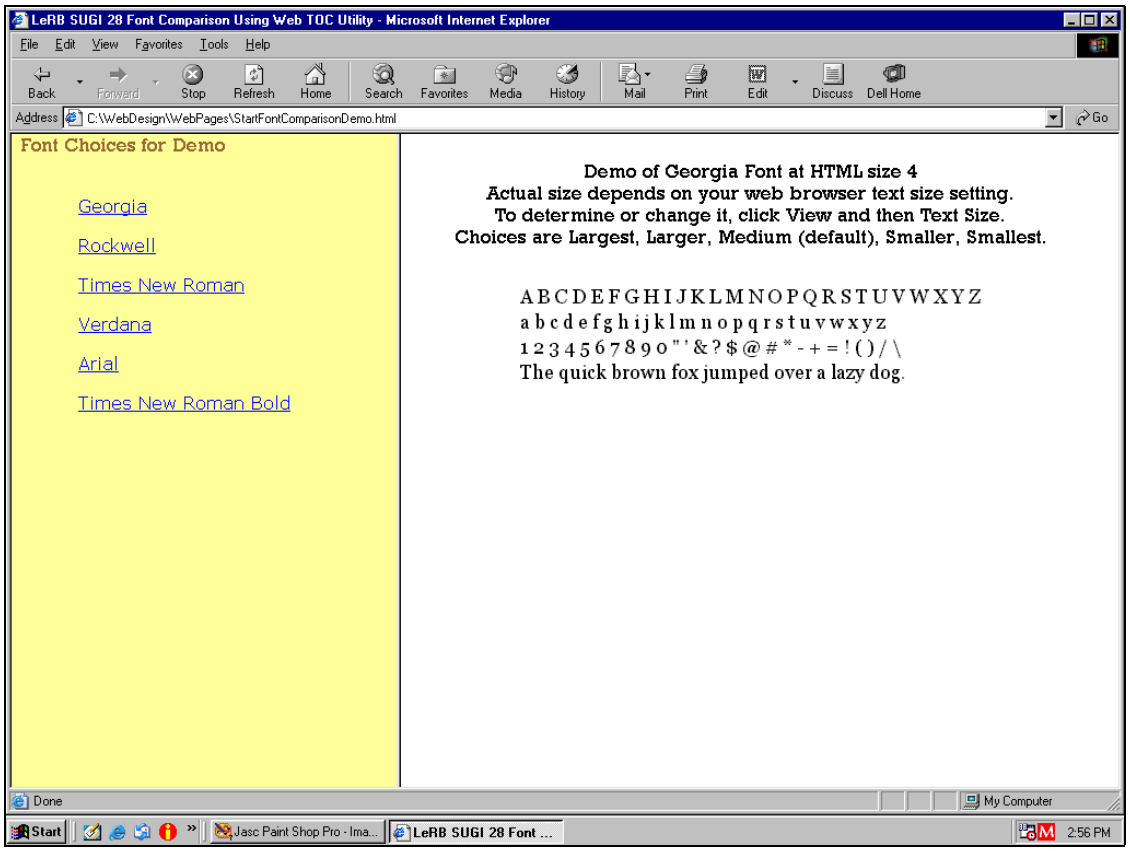


Figure 4. Utility to Demonstrate Windows True Type Fonts (See code in the Appendix. To display all 256 characters, use BFONT.)



Appendix.
Style Template Macros and Invoking Code Used for Figures

NOTES: The effect of some of the macro parameters is best understood by reading the comments inside the PROC TEMPLATE code. Below, my use of ellipsis (. . .) signifies that title text or some should-be-obvious code has been omitted. The macros have been tested, but no guarantee can be provided. You must verify them for your own uses.

```
%macro CustomBaseStyleBuild(
StyleName=LeRBsugi28CustomStyle,
PROCoutputSeparators=NO, /* YES to put horizontal rule between
  successive PROC outputs in the same web page, but
  NO EFFECT if using NEWFILE = PROC. */
PROCoutputSepLineThickness=1, /* 2 for thicker, 3 is ODS Default */
WebPageBackgroundRGBcolor=CXFFFF99,
              /* CXFFFF99 is light (not lightest) Browser-Safe yellow */
TitleFootnoteBackgroundRGBcolor=CXFFFF99,
TitleFootnoteBackgrdTransparency=NO, /* YES to let web page
background show through */
TitleFootnoteRGBcolor=CX000000, /* Browser-Safe black */
TitleFootnoteFont=Rockwell,
TitleFootnoteSize=4,
TableBackgroundRGBcolor=CXFFFF99,
TableContentRGBcolor=CX000000, /* data and heading color */
TableHeadingFont=Verdana,
TableHeadingSize=3,
TableDataFont=Verdana,
TableDataSize=3,
TableFrame=box, /* use TableFrame=void to remove frame */
TableFrameRGBcolor=CX9999FF, /* light Browser-Safe blue */
TableGrid=NO, /* use YES to turn on grid between data cells */
TableSpacing=5); /* the SAS-shipped default is 7.
                    This is the space between cell data and cell boundary. */

proc template;
edit styles.Default as styles.&StyleName;
  /* Create a modified style based on the ODS STYLES.DEFAULT.
     Anything not referenced or overridden here
     will be controlled by the ODS Default Style. */
style fonts /
  'TitleFont' = ("&TitleFootnoteFont, Times New Roman, Times",
&TitleFootnoteSize) /* "system" titles and footnotes */
  'HeadingFont' = ("&TableHeadingFont, Times New Roman, Times",
&TableHeadingSize) /* column & row headings
                      (including obs numbers and id var values) */
  'DataFont' = ("&TableDataFont, Arial, Helvetica", &TableDataSize)
     /* table data. DataFont added by LeRB.  Not in Default style. */
  'DocFont'       = ("Comic Sans MS, Courier",4);
     /* default for unassigned fonts. Conspicuous font chosen here to be
        obvious whenever used by ODS. Then a way can be found
        to assign a preferred font, rather than use default. */
style color_list /
  'LeRBred'                       = CXFF0000  /* Browser-Safe red */
  'LeRBblue'                      = CX0000FF  /* Browser-Safe blue */
  'LeRBmagenta'       = CXFF00FF  /* Browser-Safe magenta */
  'WebPageBackgroundColor'      = &WebPageBackgroundRGBcolor
  'TitleFootnoteBackgroundColor'
              = &TitleFootnoteBackgroundRGBcolor
  'TitleFootnoteColor'            = &TitleFootnoteRGBcolor
  'TableBackgroundColor'          = &TableBackgroundRGBcolor
  'TableContentColor'             = &TableContentRGBcolor
  'TableFrameColor'               = &TableFrameRGBcolor;
style colors /
  'systitlefg'              = color_list('TitleFootnoteColor')
     /* "system" titles & footnotes */
  'systitlebg'              = color_list('TitleFootnoteBackgroundColor')
     /* background for "system" title/footnote areas
        However, if transparency selected, this color is ignored. */
  'headerfg'                = color_list('TableContentColor')
     /* override fgA2, table row & column labels */
  'headerbg'                = color_list('TableBackgroundColor')
     /* background for table row & column labels */
  'datafg'                  = color_list('TableContentColor')
     /* table data */
  'databg'                  = color_list('TableBackgroundColor')
     /* background for table data */
  'docfg'                   = color_list('LeRBmagenta')
     /* default for unassigned foreground colors. Conspicuous color
        chosen here to be obvious whenever used by ODS. Then a way
        can be found to assign preferred color, rather than use default. */
  'docbg'                   = color_list('WebPageBackgroundColor')
     /* background for web page and ??? */
  'tableborder'             = color_list('TableFrameColor')
     /* table frame AND table rules */
  'TableGrid'               = color_list('TableFrameColor')
     /* (TableGrid is an LeRB replacement
        for where tablebg is used by ODS default style)
        table grid besides rules if any,
        when cellspacing > 0 AND
        style table does not assign background= */
  'link2'       = color_list('LeRBblue') /* standard for  unvisited links */
  'link1'       = color_list('LeRBred');  /* standard for visited links */
style SysTitleAndFooterContainer from Container /
  cellpadding = 0         /* compact the title/footnote area */
  cellspacing = 0         /* no grid in title/footnote area */
%if %upcase(&TitleFootnoteBackgrdTransparency) = YES %then %do;
                          background = _undef_;
style systemtitle /       background = _undef_;
style systemfooter /      background = _undef_;
     /* Three instances of background = _undef_ above
        make the title and footnote areas transparent.
        I.e., they let the web page background show through.
        If this option is selected, the systitlebg color is actually ignored. */
%end;
%else %do;
 ;   /* needed to end this STYLE statement */
%end;

style Output from Container /
     /* next three lines control table grid and table border */
  rules = NONE /* override GROUPS.
                          NO line between table labels & data. */
%if %upcase(&TableGrid) eq NO
%then %do;
  frame = &TableFrame /* BOX for on, VOID for off */
  cellspacing = 0 /* override 1. This is the space between cells.
                          When set to zero, the table grid is invisible. */
%end;
%else %do;
  frame = VOID
%end;
     /* next line controls separation of rows and columns */
  cellpadding = &TableSpacing /* override 7 */
     /* Because the table grid (rules=) is turned off above,
        the next line has no practical effect.
        It is included for completeness,
        in case you decide to turn the grid on.
        rules=ALL, rather than rules=GROUP (the default),
        would turn on a full table grid. */
  background = colors('TableGrid')
     /* table grid (LeRB replacement for tablebg),
        if cellspacing > 0
        AND style table does not assign background=.
        NOT the background of the table on the web page.*/
  bordercolor = colors('tableborder')
     /* table frame and table rules/grid */
  borderwidth = 1;
     /* table frame thickness, same as default */

style Data from Cell / font = fonts('DataFont');
     /* Added to override default use of DocFont */

%if %upcase(&PROCoutputSeparators) = NO %then %do;
style Body / pagebreakhtml = _undef_;
     /* suppress rule between successive PROC outputs */
%end;
%else %do;
```

```
style html / 'ThinLineAfterSpace' =
  " <hr size=&PROCoutputSepLineThickness>";
style Body / pagebreakhtml = html('ThinLineAfterSpace');
     /* one space and thin line between successive PROC outputs */
%end;

end; run; quit;

%mend CustomBaseStyleBuild;

%macro TableOfContentsStyleBuild(
StyleName=LeRBsugi28CustomTOCStyle,
BaseStyle=LeRBsugi28CustomStyle,
Bullet=NO, /* suppress bullets (text shifts left)
   YES gives bullet bigger than ODS default style, with a space after */
PROClabelNumbers=NO,          /* suppress numbers for PROC labels,
                                 but ODS reserves the space */
TOCbackgroundRGBcolor=CXFFCC99, /* Web-Safe light orange */
TOCmouseoverRGBcolor=CX009900, /* Web-Safe dark green */
TOCtitleANDprocRGBcolor=CX996633, /* Web-Safe brown */
TOCbulletRGBcolor=CXFF00FF, /* Web-Safe magenta */
TOCtitleFont=Rockwell,
TOCtitleSize=4,
TOCentryFont=Verdana,
TOCentrySize=3,
TOCproclabelANDmouseoverFont=Georgia,
TOCprocLabelSize=3,
TOCtitletext=%str(), /* put text in parenthesis to override default title */
TOCwidthPercentOfWebPage=25); /* Always adjust this.
   ODS default is 23%. Minimize space for TOC to maximize the
   remainder,  but probably want to avoid line breaks in TOC entries.
   If too narrow, there may be no apparent line breaks in the TOC,
   but may be an extra blank line after some entries.*/

proc template;
  edit Styles.&BaseStyle as Styles.&StyleName;
    /* Anything not referenced or overridden here will be controlled
       by Style.&BaseStyle, and its "ancestor styles", if any. */

style fonts /
  'TOCtitleFont' = ("&TOCTitleFont, Times New Roman, Times",
&TOCtitleSize)
  'TOCentryFont' = ("&TOCentryFont, Arial, Helvetica",
&TOCentrySize)
  'TOCproclabelANDmouseoverFont' =
     ("&TOCproclabelANDmouseoverFont, Times New Roman, Times",
&TOCprocLabelSize);

style color_list /
  'LeRBcyan'                  = CXFF0000  /* Web-Safe cyan */
  'TOCbackgroundColor'        = &TOCbackgroundRGBcolor
  'TOCmouseoverColor'         = &TOCmouseoverRGBcolor
  'TOCtitleANDprocColor'      = &TOCtitleANDprocRGBcolor
  'TOCbulletColor'            = &TOCbulletRGBcolor;

style colors /

  'contentfg'   = color_list('TOCmouseoverColor')
    /* mouse-over color of TOC title.
       mouse-over color of TOC PROC name,
       if PROCLABEL not suppressed.
       mouse-over color of TOC index items. */

  'contentbg'   = color_list('TOCbackgroundColor')
             /* background for TOC area */

  'conentryfg'  = color_list('TOCbulletColor')
             /* TOC link bullet */

  'confolderfg' = color_list('LeRBcyan')
    /* no one can explain what confolderfg is used for */

  'contitlefg'  = color_list('TOCtitleANDprocColor');
    /* TOC title */
    /* Also, TOC PROC name and/or TOC index number,
       if PROCLABEL and/or index number not suppressed */

%if %length(&TOCtitletext) ne 0 %then %do;
style Text / 'Content Title' = "&TOCtitletext";
             /* override default title "Table of Contents" */
%end;

style Index / font = fonts('TOCproclabelANDmouseoverFont');
  /* override inheritance from container, which defaults to docfont.
     This font face and size used for PROC labels and their numbers,
     which may be suppressed. This font face always used for mouseover,
     but size of index entries & TOC title does not change to this size. */

style IndexTitle from Index /
  font = fonts('TOCtitleFont')          /* override Default */
  posthtml = html('posthtml flyover'); /* remove rule below TOC title*/

style Frame from Document /
  framespacing = 0 /* make separator between TOC & body thinner */
  frameborderwidth = 4 /* this is ODS style default.
             I have found no apparent effect on text or tables,
             but it may affect margins for images. */
  contentsize = &TOCwidthPercentofWebPage.%; /* ODS default 23%.
             Adjust this to prevent line breaks in your TOC entries,
             which have lengths unique to your application. Keep it
             as narrow as possible, to maximize screen width use. */

style Contents from Document / pagebreakhtml = _undef_;
             /* remove useless extra white space between TOC entries */

%if %upcase(&PROClabelNumbers) eq NO %then %do;
style IndexProcName from Index / bullet = none;
     /* Suppress numbers before PROC labels in TOC.
        But horizontal space for them is retained. */
%end;

%if %upcase(&Bullet) eq YES %then %do;
style html / 'prehtml flyover CustomBullet' =
             %nrstr ("<SPAN><b>&#149;&#032;</b>");
style IndexItem / prehtml = html ('prehtml flyover CustomBullet');
%end;

style ContentItem from IndexItem / font = fonts('TOCentryFont')
%if %upcase(&Bullet) eq NO %then %do;
     /* Next line removes the bullet in front of TOC index items */
  prehtml=_undef_ posthtml=_undef_
%end;
     /* Remaining lines are a V8.2 fix needed
        to get mouse-over color to work on TOC index items. */
  listentryanchor = yes
  pretext='<span>' posttext='</span>';

end; run; quit;
%mend TableOfContentsStyleBuild;
```

Create Figure 1: Default Table of Contents and Default Body

```
ods listing close; ods noresults; goptions reset=all;
ods html path = "c:\WebDesign\WebPages" (url=none)
   frame     = "StartDefaultTOCandBodyDemo.html"
   contents  = "DefaultStyleTOC.html"
   body      = "DefaultStyleBody.html"
   style     = Styles.Default;
title1 "Students Whose Names Begin with 'A'"; footnote;
proc print noobs label
     data=sashelp.class(where=(Name =: 'A')); run;
 . . .
title1 "Students With Other Names"; footnote;
proc print noobs label data=sashelp.class
     (where=(substr(Name,1,1) not in ('A' 'J' 'R'))); run;
ods html close; ods listing;
```

Create Figure 2: Custom Table of Contents and Custom Body

```
%CustomBaseStyleBuild(StyleName=LeRBsugi28CustomBase,
     TableBackgroundRGBcolor=CX99FF99, /* light Web-Safe green */
     TitleFootnoteBackgrdTransparency=YES); run;
%TableOfContentsStyleBuild(StyleName=LeRBsugi28CustomTOC,
     TOCwidthPercentOfWebPage=27,
     BaseStyle=LeRBsugi28CustomBase); run;

ods listing close; ods noresults; goptions reset=all;
ods html path = "c:\WebDesign\WebPages" (url=none)
  frame     = "StartCustomTOCandBodyDemo.html" (title=" . . . ")
  contents  = "LeRBsugi28CustomTOC.html" (title=" . . . ")
  body      = "CustBody.html" (title=" . . . ")
  style     = Styles.LeRBsugi28CustomTOC
  newfile   = PROC;
ods proclabel ' ';  /* suppress the PROC label in TOC */
* ods proclabel 'Your Text for First PROC Label';
 /* with no proclabel statement here,
    the default in the TOC is "The Print Procedure" */
title1 "Students Whose Names Begin with 'A'"; footnote;
proc print noobs label contents="Students Named A..."
         data=sashelp.class(where=(Name =: 'A')); run;
 . . .
ods proclabel ' ';
title1 "Students With Other Names"; footnote;
proc print noobs label contents="Other Names"
    data=sashelp.class
    (where=(substr(Name,1,1) not in ('A' 'J' 'R'))); run;
ods html close; ods listing;
```

Create Figure 3:
Readability of Text Foreground Color with Background Color

```
data FontCharacters; label FontCharacters='00'X;
infile cards; input @1 FontCharacters $51.;
cards;
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n o p q r s t u v w x y z
1 2 3 4 5 6 7 8 9 0 " ' & ? $ @ # * - + = ! ( ) / \
The quick brown fox jumped over a lazy dog.
; run;

%macro ColorTable(TextColor=,BackgroundColor=);
ods proclabel ' ';
title1 " . . . &TextColor . . . on &BackgroundColor  . . . ";
proc print data=FontCharacters noobs label
    style(data)=[foreground=&TextColor
                 background=&BackgroundColor]
    contents="&TextColor on &BackgroundColor";
run;
%mend  ColorTable;

%CustomBaseStyleBuild(StyleName=LeRBsugi28BaseForColorDemo,
     TableHeadingSize=1, TableFrame=void, TableSpacing=1,
     WebPageBackgroundRGBcolor=CXFFFFFF, /* Web-Safe white */
     TableBackgroundRGBcolor=CXFFFFFF,
     TitleFootnoteBackgrdTransparency=YES,
     PROCoutputSeparators=YES)
run;
%TableOfContentsStyleBuild(Stylename=LeRBsugi28ColorsTOC,
     BaseStyle=LeRBsugi28BaseForColorDemo,
     TOCwidthPercentOfWebPage=29,
     TOCbackgroundRGBcolor=CXFFFFFF,
     TOCtitletext=%str(Color Choices for Demo))
run;

ods listing close; ods noresults; goptions reset=all;
ods html path = "c:\WebDesign\WebPages" (url=none)
  frame     = "StartColorReadabilityDemo.html" (title=" . . . ")
  contents  = "LeRBsugi28ColorsTOC.html" (title=" . . . ")
  body      = "ColBody.html" (title=" . . . ")
  style     = Styles.LeRBsugi28ColorsTOC
  newfile   = NONE; /* one continuous scrollable web page body file */
%ColorTable(TextColor=CX000000,BackgroundColor=CXFFFFFF);
run;
 . . .
%ColorTable(TextColor=CX0000FF,BackgroundColor=CXFFFFFF);
run;
ods html close; ods listing;
```

Create Figure 4:
Samples of Windows TrueType Fonts (keyboard characters only)

NOTES: PROC GFONT is not usable for this. To produce samples for the full set of 256 characters, you need a tool like my BFONT utility.

```
%macro FontTable(font=,size=4,bold=NO);
ods proclabel ' ';
title1 "Demo of &font Font "
%if %upcase(&bold) eq YES %then %do;
      "(with weight=Bold) "
%end;
      "at HTML size &size";
title3 'Actual size depends on your web browser text size setting.';
title4 'To determine or change it, click View and then Text Size.';
title5 'Choices are Largest, Larger, Medium (default), Smaller,
Smallest.';
proc print data=FontCharacters noobs label
    style(data)=[font_face="&font"
%if %upcase(&bold) eq YES %then %do;
                 font_weight=BOLD
%end;
                 font_size=&size]
    contents=
%if %upcase(&bold) eq YES %then %do;
    "&font Bold";
%end;
%else %do;
    "&font";
%end;
run;
%mend  FontTable;

%CustomBaseStyleBuild(StyleName=LeRBsugi28BaseForFontDemo,
     TableHeadingSize=1, TableFrame=void, TableSpacing=1,
     WebPageBackgroundRGBcolor=CXFFFFFF, /* Web-Safe white */
     TableBackgroundRGBcolor=CXFFFFFF,
     TitleFootnoteBackgrdTransparency=YES)
run;
%TableOfContentsStyleBuild(Stylename=LeRBsugi28FontsTOC,
     BaseStyle= LeRBsugi28BaseForFontDemo,
     TOCwidthPercentOfWebPage=35,
     TOCbackgroundRGBcolor=CXFFFF99, /* Web-Safe light yellow */
     TOCtitletext=%str(Font Choices for Demo))
run;

ods listing close; ods noresults; goptions reset=all;
ods html path = "c:\WebDesign\WebPages" (url=none)
  frame     = "StartFontComparisonDemo.html" (title=" . . . ")
  contents  = "LeRBsugi28FontsTOC.html" (title=" . . . ")
  body      = "FontBody.html" (title=" . . . ")
  style     = Styles.LeRBsugi28FontsTOC
  newfile   = PROC; /* each PROC invocation has its own web page */
%FontTable(font=Georgia); run;
 . . .
%FontTable(font=Times New Roman, bold=YES); run;
ods html close; ods listing;
```