SAS® with Style: Creating your own ODS Style Template

Lauren Haworth, Genentech, Inc., South San Francisco, CA

Abstract

Once you’ve started using the Output Delivery System, you’ll quickly discover that your taste in output design probably doesn’t coincide with the built-in ODS styles shipped with SAS software. To create output with more panache, you need to create your own style template. This workshop will take you step by step through the process of creating a custom style for your output.

You’ll learn how to make minor modifications, and how to give your output a complete makeover. If you’d like all of your SAS output to be in hot pink on an acid green background with a gigantic script font, this workshop will show you how! Or, if you’d just like to make the output smaller and use colors that coordinate with your company logo, you can do that too.

The workshop will walk through the TEMPLATE procedure, showing how you can redefine the default style elements and attributes to customize fonts, colors, and borders to suit your own personal or corporate style. You’ll be given a basic style template that you can customize during the workshop and then take home to try out on your ODS output.

The workshop is aimed at beginning to intermediate ODS users, and is based on SAS version 8.

Introduction

ODS styles are a huge topic, and there’s no way to cover them in depth in this format. This workshop will take a fast-track approach. We’ll cover just enough of the syntax to get you started. Then, we’ll use a sample style template that can be edited to modify color, fonts, spacing, rules, and borders.

This will allow you to customize most aspects of your output. However, to truly control the look of your output, plan on taking a much more in-depth course.

Using the Style= Option

You may not have realized it, but whenever you issue an ODS command you are using a style definition. By default, ODS uses a standard style for each output destination. When you issue an ODS statement like:

```ods html body='body.html';```

You’re really issuing the following:

```ods html body='body.html' style=Default;```

So if you wish to switch to another style, all you have to do is add a STYLE= option and specify the name of a different style. However, the only choices you have are the standard styles shipped with your SAS software.

PROC TEMPLATE

To truly change the look of your output, you need to create your own style. This is done by using the TEMPLATE procedure. This new procedure has statements that allow you to define every aspect of a style.

However, if we had to specify every aspect of every new style, we’d spend all of our time typing PROC TEMPLATE code. A complete style definition could run to hundreds of lines of code.

To make our life easier, we have the PARENT statement. It allows a new style to be based on an existing style. Then you can add lines of code for only those things you want to change.

The Example Program

Rather than try to explain all of the statements and syntax available for PROC TEMPLATE, let’s just look at our example program (Appendix A). This program creates a new custom style.

The first section of code sets up the name of the style (Custom) and indicates that it will be based on the Default style.

```proc template;
define style Styles.Custom;
parent = Styles.Default;
```

The next section of code sets up a list of font names and assigns them characteristics. This list is used later in the program as a shorthand way to specify fonts.

```style fonts from fonts /
'TitleFont'=('Arial, Helvetica, Helv',14pt,Bold Italic)
'TitleFont2'=('Arial, Helvetica, Helv',12pt,Bold Italic)
'StrongFont'=('Arial, Helvetica, Helv',12pt,Bold)
'EmphasisFont'=('Arial, Helvetica, Helv',10pt,Italic)
'headingFont'=('Arial, Helvetica, Helv',12pt,Bold)
'docFont'=('Arial, Helvetica, Helv',11pt)
'footFont'=('Arial, Helvetica, Helv',8pt);
```
Each attribute includes three characteristics in parentheses. Commas separate each characteristic. The first thing we specify is the typeface. In this example, you will see that each font name has a list of three typefaces. This is done to take advantage of one of the features of HTML output. Our first choice font is listed first; the other two are alternates in case the person browsing our web page does not have the first choice font.

The next section of code is very similar to the font style element. Instead of a list of font names, this one is a list of font colors. In this case a replace statement is used since we're going to replace the entire list.

```
replace color_list /
   'fgB2' = blue /* links */
   'fgB1' = darkmagenta /* visited links */
   'fgA1' = black /* table cell foreground */
   'bgA3' = lightgrey /* table cell background */
   'bgA1' = lightgrey /* table background */
   'fgR' = darkblue /* row header foreground */
   'bgR' = darkgray /* row header background */
   'fgA2' = darkblue /* column header foreground */
   'bgA2' = darkgray /* column header background */
   'fgA' = navy /* foreground: other */
   'bgA' = white /* background: other */
   'bgP' = white /* page background */;
```

The cryptic color names like ‘fgA1’ and ‘bgP’ are used by the style definition to apply these colors to various parts of the output.

The next section of code sets up the style element that controls rules, borders, and spacing for all tables. Since virtually all ODS output is in the form of tables, this is an important style element.

```
replace Output from Container /
   frame = void /* outside borders */
   rules = none /* internal borders */
   borderwidth = 1pt /* width of borders and rules */
   bordercolor = color_list('fga1') /* border color */
   cellpadding = 7pt /* space around cell contents */
   cellspacing = 0pt /* space between table cells */;
```

Unlike the previous sections, this one is not a list of names to be used elsewhere. This element lists the actual style attributes and applies settings.

The next section of code will not be covered in this workshop. It uses the colors and fonts to modify a number of other style elements. Just ignore the style statements for Body, Contents, Data, SystemTitle, SystemFooter, RowHeader, and Header.

In addition to this section that modifies some style elements, there are dozens of other style statements that are “included” in our style. Those elements are part of the Default style, and are included by way of the PARENT statement at the beginning of our PROC TEMPLATE. (If you’d like to see the full Default style, issue a PROC TEMPLATE with a single statement: “source styles.default;” and a RUN. This will dump the full definition to the log. For the purposes of this workshop, you don’t need to understand the last section of code, or the code in the Default style. We’re just going to work with the top parts.

At the end of the example PROC TEMPLATE are two more lines of code. These end the style definition that began with the DEFINE STYLE statement, and run the procedure.

```
end;
run;
```

After the PROC TEMPLATE, the example program includes some code to run a sample procedure so we can see what our style looks like. This code starts with some options settings.

```
options nodate nonumber;
ods noptitle;
ods proclabel 'Frequencies';
```

The OPTIONS statement gets rid of dates and page numbers. The first ODS statement turns off the standard procedure titles (“The FREQ Procedure”) so they don’t clutter up our output. The second ODS statement is used to control the procedure labels in the HTML table of contents. Instead of the default title “The FREQ Procedure”, our table of contents will use “Frequencies”.

The remaining lines of example code are a simple PROC FREQ, and the ODS statements needed to create HTML output. This example produces web output for ease of review during this workshop. This same code will work for RTF or PDF output as well, with a simple change to the ODS calls before and after the PROC FREQ.

```
ods html file='c:\body.html' style=Custom;
   title 'My Sample Title';
   footnote 'My Sample Footnote';
   proc freq data=sashelp.class;
      tables sex;
   run;
ods html close;
```

The only important thing to note here is the style=Custom option on the ODS statement. This calls our newly created style and applies it to the results.

That’s it for the sample program. It’s a very simple example of customizing a style, but it can be very powerful, as you’ll see later.

Before going any further, try running this sample program. Open the output file to see how the style looks right now.
If you’ve used the Default style before, you’ll realize that right now the Custom style doesn’t look very different from Default. The remainder of this workshop will be devoted to customizing the style.

**CHANGING THE TYPEFACES**

The first thing we will learn how to modify is the fonts. We’ll be working with the fonts style element. To change the font in part of your output, all you have to do is use PROC TEMPLATE to modify the font definition that applies to that part of your output. Appendix B lists each of the font names, and where they apply.

For each font name, we can modify three characteristics. The first is the typeface. To make a change, simply replace the typefaces listed between quotes with typefaces of your choice. Keep in mind that the person receiving your output will need to have the same fonts in order to view the web page, RTF file, or PDF file properly. You want to pick fonts that are commonly available. Appendix C lists some good font combinations to try.

Using the sample program, try changing the typefaces used in the fonts style element. Before doing this, you may want to save the sample program using a different name so that you can go back if you make a mistake. Try changing a couple of the typefaces and then re-running the program.

View the HTML file again to see how the change affected the output. You can leave the web page open, and just use the refresh button on the browser to see each new version of the output.

### A sample modification:

```plaintext
style fonts from fonts /
  'TitleFont' = ("Comic Sans MS, Arial, Helvetica", 14pt,Bold Italic)
  'TitleFont2' = ("Comic Sans MS, Arial, Helvetica", 12pt,Bold Italic)
  'StrongFont' = ("Comic Sans MS, Arial, Helvetica", 12pt,Bold)
  'EmphasisFont' = ("Comic Sans MS, Arial, Helvetica", 10pt,Italic)
  'headingFont' = ("Comic Sans MS, Arial, Helvetica", 12pt,Bold)
  'docFont' = ("Trebuchet MS, Arial, Helvetica", 12pt)
  'footFont' = ("Arial, Helvetica, Helv", 8pt);
```

The resulting output:

### CHANGING THE FONT SIZES, WEIGHTS, AND STYLES

Now that we’ve got the typefaces we want, we can turn to the font sizes. You may have noticed that the default HTML output from ODS uses very large fonts. Our example style has been set up with somewhat smaller fonts already. You can choose to keep them this size, or make them larger or even smaller.

The Default style uses the same font specification for titles and footnotes. Our example style uses a different setting for each, so that you can have large titles and small footnotes.

Try making some of the fonts bigger or smaller and see how this affects the output.

---

1 As of version 8.2, SAS does not embed fonts in PDF documents. Hopefully in the future this functionality will be added, allowing the use of any font in PDF output.
A sample modification:

```plaintext
style fonts from fonts /
  'TitleFont' = ("Comic Sans MS, Arial, Helvetica", 18pt, Bold Italic)
  'TitleFont2' = ("Comic Sans MS, Arial, Helvetica", 12pt, Bold Italic)
  'StrongFont' = ("Comic Sans MS, Arial, Helvetica", 12pt, Bold)
  'EmphasisFont' = ("Comic Sans MS, Arial, Helvetica", 10pt, Italic)
  'headingFont' = ("Comic Sans MS, Arial, Helvetica", 11pt, Bold)
  'docFont' = ("Trebuchet MS, Arial, Helvetica", 10pt)
  'footFont' = ("Arial, Helvetica, Helv", 8pt);
```

The resulting output:

```plaintext
The other thing you can change about fonts is the font weight (Medium, Bold, Light) and the font style (Italic, Roman, Slant). You may also be able to control the font width, though few fonts honor settings like Compressed or Expanded. To use any of these settings, just list the appropriate keyword(s) after the font size specification. Generally, the only two settings that you'll want to add are Bold and/or Italic. If you leave this setting blank, the fonts are set to Medium Roman.

Try changing some of these settings to see what happens.

A sample modification:

```plaintext
style fonts from fonts /
  'TitleFont' = ("Comic Sans MS, Arial, Helvetica", 18pt, Bold)
  'TitleFont2' = ("Comic Sans MS, Arial, Helvetica", 12pt, Bold)
  'StrongFont' = ("Comic Sans MS, Arial, Helvetica", 12pt, Bold)
  'EmphasisFont' = ("Comic Sans MS, Arial, Helvetica", 10pt, Italic)
  'headingFont' = ("Comic Sans MS, Arial, Helvetica", 11pt, Bold)
  'docFont' = ("Trebuchet MS, Arial, Helvetica", 10pt)
  'footFont' = ("Arial, Helvetica, Helv", 8pt, Italic);
```

The other thing you can change about fonts is the font weight (Medium, Bold, Light) and the font style (Italic, Roman, Slant). You may also be able to control the font width, though few fonts honor settings like Compressed or Expanded. To use any of these settings, just list the appropriate keyword(s) after the font size specification. Generally, the only two settings that you’ll want to add are Bold and/or Italic. If you leave this setting blank, the fonts are set to Medium Roman.

Try changing some of these settings to see what happens.

A sample modification:

```plaintext
style fonts from fonts /
  'TitleFont' = ("Comic Sans MS, Arial, Helvetica", 18pt, Bold)
  'TitleFont2' = ("Comic Sans MS, Arial, Helvetica", 12pt, Bold)
  'StrongFont' = ("Comic Sans MS, Arial, Helvetica", 12pt, Bold)
  'EmphasisFont' = ("Comic Sans MS, Arial, Helvetica", 10pt, Italic)
  'headingFont' = ("Comic Sans MS, Arial, Helvetica", 11pt, Bold)
  'docFont' = ("Trebuchet MS, Arial, Helvetica", 10pt)
  'footFont' = ("Arial, Helvetica, Helv", 8pt, Italic);
```

### Changing the Colors

Changing the fonts is a fairly subtle thing. This next section lets you make big bold changes to your output. This section considers the color scheme.

ODS allows you to set the foreground (text) colors and background colors of every part of your output. These colors are set by defining a color scheme in the colors style element.

In the example program, each color is identified by name. Appendix D lists the color names you can use. This gives you a palette of 216 colors. This is a list of web-safe colors.

You also have the option of specifying custom colors by using their RGB values given in hexadecimal. For example, white would be cFFFFFF, and black would be cx000000 (the “cx” tells SAS that the following value is a hexadecimal color). For the purposes of this workshop, let’s stick to the named colors.

When you modify these colors, notice that some of the names start in “fg” and represent foreground colors. Others start in “bg” and represent background colors. These colors work in pairs, and you need to be sure that you pick pairs of colors that will be readable. For example, pink foreground text on a red background would be a problem.

Try creating a new color scheme. See how it looks. One warning here: if you use this style for a destination other than HTML, there may be some minor differences in how the colors are used. When you get your style set up for HTML, you’ll want to test it on other destinations.
A sample modification:

```plaintext
replace color_list /
'fgB2' = blue
'fgB1' = darkmagenta
'fgA1' = orchid
'bgA3' = lime
'bgA1' = white
'fgR' = lime
'bgR' = darkorchid
'fgA2' = lime
'bgA2' = darkorchid
'fgA' = deeppink
'bgA' = white
'bgP' = white;
```

The resulting output:

If you’re not very creative, there’s a web site that will help you design an attractive color scheme. Go to http://www.colorschemer.com/online/ and click on a color that you like. The web site will generate a group of 16 related colors that create an attractive color scheme. You can then copy down the hex codes for these colors and use them in your style.

Another way to pick colors for your scheme is to use colors from your corporate logo. Ask your graphics department for the correct color codes. They should be able to give you the RGB values (you can find an RGB/hex converter on the web).

CHANGING THE TABLE RULES AND BORDERS

The next thing we will modify is the table rules and borders. The lines around the table and between rows and columns are controlled by two attributes: rules and frame.

The frame attribute specifies whether there will be any lines around the outside of your tables. The frame is currently set to void, which means there will be no lines around the table. Another setting to try is box, which puts a line around the entire table. There are also settings that let you have borders top and bottom, both sides, or any individual edge.

Try changing the frame setting. Don’t worry about the line width or color right now; we’ll get to that later.

A sample modification:

```plaintext
replace Output from Container /
frame = box
rules = none
borderwidth = 1pt
bordercolor = color_list('fga1')
cellpadding = 7pt
cellspacing = 0pt;
```

The resulting output is below. This change is a little hard to see, but lines have been added around the table.

![My Sample Title](image)

The rules attribute controls the lines that appear inside your tables. This attribute is currently set to none, so there are no lines at all. Other settings to try are all and group. All turns on all possible lines, creating a table grid. Group puts a border between row and column headers and footers and the rest of the table body. Other settings include rows and cols, which include only row dividers or column dividers.

Try changing the rules setting. You may also want to experiment with combinations of frame and rules settings.

A sample modification:

```plaintext
replace Output from Container /
frame = box
rules = all
borderwidth = 1pt
bordercolor = color_list('fga1')
cellpadding = 7pt
cellspacing = 0pt;
```

The resulting output:

![My Sample Title](image)

Now that you have all of the lines you want, we can look at the width and color for those lines. These are controlled

---

5 Note: the site only works with Internet Explorer.
by the borderwidth and bordercolor attributes. Don’t forget that neither of these settings will have any effect unless you’ve specified some lines for your table. With frame=void and rules=none, the border width and color are irrelevant.

Borderwidth is simply the line width. It affects the width of the table border, but not the rules. Use a number followed by “pt” to set the width in points.

Bordercolor is assigned using a color from your list above. Note the syntax here in the original program: it uses the color_list element name, followed by the name of one of the items in that element (in quotes and inside parentheses). You can also just use a color name or code. Unlike borderwidth, bordercolor applies to all lines, both borders and rules.

Try experimenting with the border width and color. If your custom style will not have any lines, go ahead and turn on frame=box and rules=all so that you can at least see how they work. You can reset frame and rules later.

A sample modification:
replace Output from Container /
   frame = box
   rules = all
   borderwidth = 2pt
   bordercolor = black
   cellpadding = 7pt
   cellspacing = 0pt;

The resulting output:

My Sample Title

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>9</td>
<td>9</td>
<td>47.4</td>
</tr>
<tr>
<td>M</td>
<td>10</td>
<td>19</td>
<td>100.0</td>
</tr>
</tbody>
</table>

My Sample Footnote

> **SAVING YOUR STYLE**

Once you’ve created your custom style, you can save the program that created. This will allow you to regenerate it at any time.

But you don’t need to run this PROC TEMPLATE every time you want to use your new style. SAS has saved the style for you in the sasuser library.

If this style is for you alone, this will work just fine. But if you want to share your style, you will need to make a couple of changes.

First, set up a libname for your custom style in a commonly accessible area. Then, you’ll need to learn about the ODS PATH statement, which you can use to route your custom style to this libname. Other users can set up the same ODS PATH statement in their programs to reference this libname and access your style.

> **CONCLUSIONS**

This workshop has been a short cut to using some basic style functionality. If you just need to quickly modify a style, this may be enough.

However, this template only allows you to modify certain aspects of your output. You may find that you want to control other aspects. To do that, you’re going to have to learn a lot more about PROC TEMPLATE syntax.

---

3 Warning: these settings work much better in Internet Explorer than Netscape. Your HTML output may not look the same in the two browsers.
Resources
PROC TEMPLATE documentation is in the References chapter of:


Preliminary documentation of new features and sample programs can be found at:

http://www.sas.com/rnd/base/index-ods-resources.html

My book on ODS has a number of chapters on modifying ODS styles:


Acknowledgements
SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Contacting the Author
Please direct any questions or feedback to the author at:
info@laurenhaworth.com
**APPENDIX A**

```sas
proc template;
define style Styles.Custom;
  parent = Styles.Default;
  style fonts from fonts /
    'TitleFont' = ('Arial, Helvetica, Helv',14pt,Bold Italic)
    'TitleFont2' = ('Arial, Helvetica, Helv',12pt,Bold Italic)
    'StrongFont' = ('Arial, Helvetica, Helv',12pt,Bold)
    'EmphasisFont' = ('Arial, Helvetica, Helv',10pt,Italic)
    'headingFont' = ('Arial, Helvetica, Helv',12pt,Bold)
    'docFont' = ('Arial, Helvetica, Helv',11pt)
    'footFont' = ('Arial, Helvetica, Helv',8pt);
  replace color_list /
    'fgB2' = blue /* links */
    'fgB1' = darkmagenta /* visited links */
    'fgA1' = black /* table cell foreground */
    'bgA3' = Lightgrey /* table cell background */
    'bgA1' = Lightgrey /* table background - shows through if cellspacing>0 */
    'fgR' = darkblue /* row header foreground */
    'bgR' = darkgray /* row header background */
    'fgA2' = darkblue /* column header foreground */
    'bgA2' = darkgray /* column header background */
    'fgA' = navy /* foreground for everything else: notes, proc titles, ... */
    'bgA' = white /* background for everything else: notes, proc titles, ... */
    'bgP' = white /* page background */;
  replace Output from Container /
    frame = void /* outside borders: void, box, above/below, vsides/hsides, lhs/rhs */
    rules = none /* internal borders: none, all, cols, rows, group */
    borderwidth = 1pt /* the width of the borders and rules */
    bordercolor = color_list('fgA1') /* the color of the borders and rules */
    cellpadding = 7pt /* the space between table cell contents and the cell border */
    cellspacing = 0pt /* the space between table cells, allows background to show */;
  * Leave code below this line alone ;
  style Body from Body /
    background = color_list('bgP');
  style Contents from Contents /
    background = color_list('bgP');
  style Data from Data /
    font = fonts('docFont');
  style SystemTitle from SystemTitle /
    font = fonts('TitleFont');
  style SystemFooter from SystemFooter /
    font = fonts('footFont');
  style RowHeader from Header /
    font = fonts('headingFont');
  style Header from Header /
    font = fonts('headingFont');
end;
run;
```

```sas
options nodate nonumber;
ods noptitle;
ods proclabel 'Frequencies';
ods html file='c:\body.html' style=Custom;
title 'My Sample Title';
footnote 'My Sample Footnote';
proc freq data=sashelp.class;
tables sex;
run;
ods html close;
```
APPENDIX B

<table>
<thead>
<tr>
<th>Font Style</th>
<th>Portion of Output it Controls</th>
</tr>
</thead>
<tbody>
<tr>
<td>TitleFont</td>
<td>Titles generated with TITLE statement</td>
</tr>
<tr>
<td>TitleFont2</td>
<td>Titles for procedures (“The ________ Procedure”)</td>
</tr>
<tr>
<td>StrongFont</td>
<td>Strong (more emphasized) table headings and footers, page numbers</td>
</tr>
<tr>
<td>EmphasisFont</td>
<td>Titles for table of contents and table of pages, emphasized table headings and footers</td>
</tr>
<tr>
<td>headingFont</td>
<td>Table column and row headings and footers, by-group headings</td>
</tr>
<tr>
<td>docFont</td>
<td>Data in table cells</td>
</tr>
<tr>
<td>footFont</td>
<td>Footnotes generated with FOOTNOTE statement</td>
</tr>
</tbody>
</table>

APPENDIX C

<table>
<thead>
<tr>
<th>“Safe” fonts4,5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Times New Roman, Times</td>
</tr>
<tr>
<td>Arial, Helvetica</td>
</tr>
<tr>
<td><strong>Arial Black</strong>, Arial, Helvetica</td>
</tr>
<tr>
<td>Book Antigua, Times New Roman, Times</td>
</tr>
<tr>
<td>Courier New, Courier</td>
</tr>
<tr>
<td><strong>Comic Sans MS</strong>, Arial, Helvetica</td>
</tr>
<tr>
<td>Verdana, Arial, Helvetica</td>
</tr>
<tr>
<td><strong>Impact, Arial Black</strong>, Helvetica</td>
</tr>
<tr>
<td>Georgia, Times New Roman, Times</td>
</tr>
<tr>
<td>News Gothic MT, Arial, Helvetica</td>
</tr>
<tr>
<td><strong>Tahoma, Arial, Helvetica</strong></td>
</tr>
<tr>
<td>Trebuchet MS, Arial, Helvetica</td>
</tr>
</tbody>
</table>

---

4 Although these fonts are fairly safe, good alternate fonts have also been listed for each item. Also, this list is based on standard Windows fonts. If you have a lot of Mac users, you may want to list some Mac fonts like Chicago, Geneva, Helvetica, Monaco, New York, Times and Palatino as alternatives.

5 Two very unsafe fonts are SAS Monospace and SAS Monospace Bold.
<table>
<thead>
<tr>
<th>Color Name</th>
<th>Color Name</th>
<th>Color Name</th>
<th>Color Name</th>
<th>Color Name</th>
<th>Color Name</th>
<th>Color Name</th>
<th>Color Name</th>
<th>Color Name</th>
<th>Color Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Floralwhite</td>
<td>Beige</td>
<td>Tan</td>
<td>White smoke</td>
<td>Darkkhaki</td>
<td>Peru</td>
<td>Firebrick</td>
<td>Lavender</td>
<td>Lightgoldenearyellow</td>
<td>Limegreen</td>
</tr>
<tr>
<td>Ivory</td>
<td>Alceblue</td>
<td>Blanchedalmond</td>
<td>Silver</td>
<td>Khaki</td>
<td>Burlywood</td>
<td>Palegoldenrod</td>
<td>Lavender</td>
<td>Mediumaquamarine</td>
<td>Mediumturquoise</td>
</tr>
<tr>
<td>Honeysuckle</td>
<td>Navajowhite</td>
<td>Gainsboro</td>
<td>Lemonchiffon</td>
<td>Sandybrown</td>
<td>Mistyrose</td>
<td>Palerouge</td>
<td>Limegreen</td>
<td>Turquoise</td>
<td>Skyblue</td>
</tr>
<tr>
<td>Gainsboro</td>
<td>Mintyrose</td>
<td>Oldlace</td>
<td>Red</td>
<td>Tomato</td>
<td>Lightgoldenrod</td>
<td>Mediumseagreen</td>
<td>Palegreen</td>
<td>Teal</td>
<td>Lightsteelblue</td>
</tr>
<tr>
<td>Antiquewhite</td>
<td>Mintcream</td>
<td>Moccasin</td>
<td>Darkgray</td>
<td>Chocolate</td>
<td>Darkred</td>
<td>Peachpuff</td>
<td>Mediumspringgreen</td>
<td>Seagreen</td>
<td>Mediumslateblue</td>
</tr>
<tr>
<td>Cornsilk</td>
<td>Azure</td>
<td>Bisque</td>
<td>Lightgray</td>
<td>Sienna</td>
<td>Tomato</td>
<td>Maroon</td>
<td>Mediumturquoise</td>
<td>Lightcyan</td>
<td>Mediumvioletblue</td>
</tr>
<tr>
<td>Healthcare</td>
<td>Brown</td>
<td>Gray</td>
<td>Crimson</td>
<td>Orange</td>
<td>Yellowgreen</td>
<td>Lightseagreen</td>
<td>Mediumslateblue</td>
<td>Cyan</td>
<td>Indigo</td>
</tr>
<tr>
<td>White</td>
<td>Snow</td>
<td>Oldlace</td>
<td>Black</td>
<td>Brown</td>
<td>Red</td>
<td>Lightyellow</td>
<td>Darkgreen</td>
<td>Lightcyan</td>
<td>Mediumpurple</td>
</tr>
<tr>
<td>Pink</td>
<td>Aqua</td>
<td>Olive</td>
<td>Darkgray</td>
<td>Slategray</td>
<td>Tomato</td>
<td>Maroon</td>
<td>Darkgray</td>
<td>Cyan</td>
<td>Mediumvioletblue</td>
</tr>
<tr>
<td>SUGI 28 Hands-on</td>
<td>Workshops</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>