

## Macros and Conventional Macro Variables: Effective Tools for Customizing Tabular Output in SAS® ODS

Perry Watts, Stakana Analytics, Elkins Park, PA

### ABSTRACT

While it is understood that macro variables are referenced in ODS by using MVAR and NMVAR statements, macro programs continue to play a vital role in ODS. To back up this claim, the TABLETEMPLATE macro that generates a customized table in ODS is reviewed in depth. Looping in the macro eliminates the need for hard coding, and multiple ampersand macro variables (&&i) provide the only viable means for inserting data dependent text strings into a table. Pre and Post macro versions of the code are presented to emphasize the role played by macros in ODS.

### PROBLEM DEFINITION

To generalize PROC TEMPLATE table definitions, it is recommended that DYNAMIC variables reference structural components such as headers or footers and that MVAR and NMVAR be reserved for constants that can change each time a compiled template is called in a data step. However, what if the macro variables you need to define come from the set of unique values in a column from the input data? Furthermore, what if you need a different outcome in a PROC TEMPLATE CELLSTYLE-AS statement for each unique value in the column? In this situation, a reliable table definition requires the assistance of a full-fledged macro with looping capabilities. What the paper describes is an application where an ODS table definition is nested inside the macro TABLETEMPLATE. The table definition uses MVAR and DYNAMIC variables in addition to conventional macro variables with multiple ampersands for managing the CELLSTYLE-AS statement.

### TRACING FONTS IN THE ODS STYLES.DEFAULT TEMPLATE FOR VERSION 9.2 SAS

Fonts comprise a sizable fraction of the attributes associated with CONTAINER-derived style elements in the ODS Styles.Default template. Identifying an actual font that is assigned to a particular style element can present a challenge because of an indirect mapping sequence embedded in the template. How a few fonts are mapped is illustrated in Figure 1 below.

**Figure 1.** A partial listing of a numbered version of the ODS Styles.Default template is presented in linear order. Fonts are referenced by name rather than by value throughout the template. This means you have to go back to the beginning of the template to find out that the default value for DOCFONT in the CONTAINER style element is <sans-serif>, Helvetica, sans-serif, relative size 3. BATCHFIXEDFONT maps to SAS Monospace, <monospace>, Courier, monospace" relative size 2. (Gray-overlay fonts below are not used by any of the CONTAINER style elements).

Obs	Style Definition
---	-----
1	class fonts
2	"Fonts used in the default style" /
3	'TitleFont2' = ("<sans-serif>, Helvetica, sans-serif",4,bold italic)
4	'TitleFont' = ("<sans-serif>, Helvetica, sans-serif",5,bold italic)
5	'StrongFont' = ("<sans-serif>, Helvetica, sans-serif",4,bold)
6	'EmphasisFont' = ("<sans-serif>, Helvetica, sans-serif",3,italic)
7	'FixedEmphasisFont' = ("<monospace>, Courier, monospace",2,italic)
8	'FixedStrongFont' = ("<monospace>, Courier, monospace",2,bold)
9	'FixedHeadingFont' = ("<monospace>, Courier, monospace",2)
10	'BatchFixedFont' = ("SAS Monospace, <monospace>, Courier, monospace",2)
11	'FixedFont' = ("<monospace>, Courier",2)
12	'headingEmphasisFont' = ("<sans-serif>, Helvetica, sans-serif",4,bold italic)
13	'headingFont' = ("<sans-serif>, Helvetica, sans-serif",4,bold)
14	'docFont' = ("<sans-serif>, Helvetica, sans-serif",3);
...	
175	class Container /
176	font = Fonts('DocFont')
177	color = colors('docfg')
178	backgroundcolor = colors('docbg');
...	
349	class Batch /
350	font = fonts('BatchFixedFont')
351	color = colors('batchfg')
352	backgroundcolor = colors('batchbg');

Needed here is a table that links a font directly to the style element that uses it. In Figure 2, a few records from the input data set, FONTTRACER, are formatted for output by an application of the TABLETEMPLATE macro. What gives the final output its pizzazz is that the font listed in the FONT column is applied to the very same text displayed in that column.

The enhanced table in Figure 2 employs the same method used for displaying hexadecimal color codes in a color chart. The text for the color code is superimposed over a sample of the actual color. To look at [color charts](#), check out the author's screencast.com account. The HTML font tracer is also available for [download](#), and it is reproduced as a screen snapshot in the [Appendix](#).

**Figure 2.** The input data set, FONTTRACER, is derived from the styles.default template. FONTTRACER is transformed into a font tracer *table* with an application of the TABLETEMPLATE macro. By looking at a display of the actual fonts in the table, a more informed decision can be made when attributes need to be changed in a style element.

Input Data Set: fontTracer			
Obs	font	Font Name	Style Element
1	"SAS Monospace, <monospace>, Courier, monospace",2	batchfixedfont	Batch
2	"<sans-serif>, Helvetica, sans-serif",3	docfont	Container
...			
42	"<sans-serif>, Helvetica, sans-serif",5,bold italic	titlefont	SystemTitle
43	"<sans-serif>, Helvetica, sans-serif",5,bold italic	titlefont	SystemFooter
44	"<sans-serif>, Helvetica, sans-serif",4,bold italic	titlefont2	TitlesAndFooters

Font	Font Name	Style Element
"SAS Monospace, <monospace>, Courier, monospace",2	batchfixedfont	Batch
"<sans-serif>, Helvetica, sans-serif",3	docfont	Container
<b>"&lt;sans-serif&gt;, Helvetica, sans-serif",5,bold italic</b>	titlefont	SystemTitle
		SystemFooter
<b>"&lt;sans-serif&gt;, Helvetica, sans-serif",4,bold italic</b>	titlefont2	TitlesAndFooters

## THE PRE-MACRO VERSION OF THE SOURCE CODE FOR THE FONT TRACER

Yellow highlighted code in the listing below is changed in the TABLETEMPLATE macro. The repetitive code that is highlighted indicates the need for a macro program. Gray highlights reference MVAR and DYNAMIC variables. To see how COLHD is actually used, look at the DATA \_NULL\_ step in the full version of the source code available by [download](#). The DYNAMIC statement is also covered on pages, 342,348, and 391 in the ODS User's Guide [\[2\]](#).

```

/* CREATE MACRO VARIABLES */
proc sql noprint;
  create table uniqueFonts as
    select distinct font from fontTracer;
  select count(*) into :n
    from uniqueFonts;
  select font into :f1 - :f%left(&n)
    from uniqueFonts;
quit;
%let n = %left(%trim(&n));

/* CREATE A TABLE TEMPLATE */
proc template;
  define Table TABLES.fontTbl;
    dynamic colhd;
    mvar f1 f2 f3 f4 f5 f6 f7 f8 f9;
    classlevels=on;
    define column columnFont;
      blank_dups=on;
      define header colhd;
        style=header;
        just=Center;
      end;
      header=colhd;
    CELLSTYLE
      val EQ F1 as {font = ("<monospace>, Courier",3)},
      val EQ F2 as {font = ("<monospace>, Courier, monospace",2,bold)},

```

```

    val EQ f3 as {font = ("

```

The CELLSTYLE-AS statement is very similar to the SELECT statement used in base SAS. Each time a record is processed, `_VAL_` is compared to the contents of F1-F9. When a match is found, ODS applies the corresponding font to the output and moves on to the next record. If there is no match, the default font associated with the DATA style element is applied to the output. (1 AS data in the code plays the same role as OTHERWISE in a SELECT statement).

As a preliminary step, it is necessary to print out the data set UNIQUEFONTS in order to hard-code the values for the MVAR variables. Hard-coding is required for conformance to style-attribute specification in the syntax:

`CELLSTYLE expression AS {style-attribute-name = style-attribute-value}` [2, p. 389].

The *style-attribute-value* cannot reference a variable, not even an MVAR variable. It only works with text strings, and in this program, the text string on the right side of the equation is equivalent to the MVAR variable in the expression. Thus:

```
_val EQ F1 as {font = ("

```

translates to:

```
_val EQ "<monospace>, Courier",3 as {font = ("

```

This is how the font attribute is formatted by the referenced font in Figure 2.

CELLSTYLE-AS is typically used to check data set values against ranges, not point values. *Output Delivery System: The Basics and Beyond*, shows by example how to assign traffic lighting by specifying ranges with user-defined endpoints in a CELLSTYLE-AS statement [1, p. 282].

## USING THE TABLETEMPLATE MACRO TO GENERATE THE FONT TRACER

Below is the revised code that replaces hard-coding with the TABLETEMPLATE macro. Not only is the code shorter in length, it is also more reliable. There is no need, for example, to obtain intermediate printouts for the number and values of the MVAR variables. Also, there really is no need for the catchall 1 as data in the CELLSTYLE-AS statement. The data set FONTTRACER supplies values to the macro variables from the same column that is being processed in PROC TEMPLATE.

```

/* CREATE MACRO VARIABLES */
proc sql noprint;
    create table uniqueFonts as
        select distinct font from fontTracer;
    select count(*) into :n
        from uniqueFonts;
    select font into :f1 - :f%left(&n)
        from uniqueFonts;
quit;
%let n = %left(%trim(&n));

/* CREATE A TABLE TEMPLATE */
%macro tableTemplate;
proc template;
    define Table TABLES.fontTbl;
        dynamic colhd;
        mvar %do i = 1 %to &n; f&i %end; ;
        classlevels=on;
        define column columnFont;
            blank_dups=on;
            define header colhd;
                style=header;

```

```

        just=Center;
    end;
    header=colhd;
    cellstyle
        %do i= 1 %to &n;
            _val_ EQ F&i as {font = (&&F&i)},
        %end;
        1 as data;
    end; /* DEFINE COLUMN */
    define column restOfData;
        generic=on;
        blank_dups=on;
        define header colhd;
            style=header;
            just=Center;
        end;
        header=colhd;
    end; /* DEFINE COLUMN */
end; /* DEFINE TABLE */
run;
%mend tableTemplate;
%tableTemplate

```

## SUMMARY AND CONCLUSIONS

The TABLETEMPLATE macro described in this paper shows that macro programs can play a vital role in the construction of ODS table templates. With this macro, repetitive coding is eliminated and program reliability is increased. The macro also makes it possible to extend the capability of the CELLSTYLE-AS statement to check for point value equalities. In addition, greater transparency is achieved with TABLETEMPLATE. The relationship between an MVAR variable and its matching font assignment is clarified with `_val_ EQ F&i as {font = (&&F&i)}` in CELLSTYLE-AS.

## COPYRIGHT STATEMENT

The paper, *Macros and Conventional Macro Variables: Effective Tools for Customizing Tabular Output in SAS® ODS*, is protected by copyright law. This means if you would like to paraphrase original ideas, adapt output from figures or attachments for your own use, or quote text from the paper in any type of publication you are welcome to do so. All you need to do is to cite the paper. For all uses that result in corporate or individual profit, written permission must be obtained from the author. Conditions for usage have been modified from <http://www.whatiscopyright.org>.

## REFERENCES

- [1] Haworth, Lauren E., Cynthia L. Zender, and Michele M. Burlew. *Output Delivery System: The Basics and Beyond*. Cary, NC: SAS Institute Inc., 2009.
- [2] SAS Institute Inc. *SAS® 9.1 Output Delivery System: User's Guide*. Cary NC: SAS Institute Inc., 2004.

## HOW TO LOCATE 426-2011.ZIP WHERE RELATED FILES ARE STORED:

### AT SCREENCAST.COM

- 1) Go to <http://www.screencast.com/users/PerryWatts>
- 2) Click on README for a quick guide to website navigation.
- 3) Click on the folder icon labeled *ODS Style Inheritance*. This will take you to a subdirectory that contains abstracts for related papers. Click on *Macros and MacroVariables are Effective Tools for Building a Font Tracer (9.2 SAS)*, then click on *Attachments*.

### AT SASCOMMUNITY.ORG

- 1) Go to <http://www.sascommunity.org>
- 2) Follow sidebar links: Navigate>Popular Links>Presentations>SGF2011 Presentations

## FILES ON 426-2011.ZIP

- 1) Text File: **StylesDefault92.txt**
- 2) SAS data set: **looklike913.sas7bdat**
- 3) SAS Programs: **Mk92StylesDefaultTxtData.sas** and **mkFontTracer92.sas**
- 4) The Font Tracer: **containerFontTracer92.HTML**

## HOW TO LOCATE COLOR CHARTS

### AT SCREENCAST.COM

- 1) Go to <http://www.screencast.com/users/PerryWatts>
- 2) Click on the folder icon labeled *Color*. This will take you to a subdirectory contains abstracts for the four papers listed below. Color charts can either be found in the papers or in a couple of the image files posted in the subdirectory.

### PAPERS AVAILABLE FOR DOWNLOAD

Watts, Perry. *Using ODS and the Macro Facility to Construct Color Charts and Scales for SAS® Software Applications*. Proceedings of the Twenty-Seventh SAS® User Group International Conference, Cary, NC: SAS Institute, 2002, paper #125.

<sup>5)</sup> Watts, Perry. *Working with RGB and HLS Color Coding Systems in SAS® Software*. Proceedings of the Twenty-Eighth SAS® User Group International Conference, Seattle, WA, 2003, paper #136.

<sup>6)</sup> Watts, Perry. *Advanced Programming Techniques for Working with Color in SAS® Software*. Proceedings of the Twenty-Ninth SAS® User Group International Conference, Cary, NC: SAS Institute, 2004, paper #091.

<sup>7)</sup> Watts, Perry. *New Palettes for SAS® Color Utility Macros*. Proceedings of the Twenty-Ninth SAS® User Group International Conference, Cary, NC: SAS Institute, 2004, paper #162.

### TRADEMARK CITATION

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

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

### CONTACT INFORMATION

The author welcomes feedback via email at [perryWatts@comcast.net](mailto:perryWatts@comcast.net)

## APPENDIX: SCREEN SNAPSHOT OF THE FONT TRACER

### Font Tracer for the SAS STYLES.DEFAULT Container Style Elements

Font	Font Name	Style Element
"SAS Monospace, <monospace>, Courier, monospace",2	batchfixedfont	Batch
"<sans-serif>, Helvetica, sans-serif",3	docfont	Container
		Listitem
		Paragraph
"<sans-serif>, Helvetica, sans-serif",3,italic	emphasisfont	IndexTitle
		ExtendedPage
		DataEmphasis
		HeaderEmphasis
		RowHeaderEmphasis
		FooterEmphasis
		RowFooterEmphasis
"<monospace>, Courier, monospace",2,italic	fixedemphasisfont	DataEmphasisFixed
		HeaderEmphasisFixed
		RowHeaderEmphasisFixed
		FooterEmphasisFixed
		RowFooterEmphasisFixed
"<monospace>, Courier",2	fixedfont	NoteContentFixed
		WarnContentFixed
		ErrorContentFixed
		FatalContentFixed
		DataFixed
		HeaderFixed
		RowHeaderFixed
		FooterFixed
		RowFooterFixed
"<monospace>, Courier, monospace",2,bold	fixedstrongfont	ProcTitleFixed
		DataStrongFixed
		HeaderStrongFixed
		RowHeaderStrongFixed
		FooterStrongFixed
		RowFooterStrongFixed
"<sans-serif>, Helvetica, sans-serif",4,bold	headingfont	Byline
		Parskip
		Continued
		HeadersAndFooters
	strongfont	PageNo
		DataStrong
		HeaderStrong
		RowHeaderStrong
		FooterStrong
		RowFooterStrong
"<sans-serif>, Helvetica, sans-serif",5,bold italic	titlefont	SystemTitle
		SystemFooter
"<sans-serif>, Helvetica, sans-serif",4,bold italic	titlefont2	TitlesAndFooters