Paper 227-31

Let the ODS PRINTER Statement Take Your Output into the Twenty-First Century

Scott Huntley, SAS Institute Inc., Cary, NC

ABSTRACT

Since its debut in SAS® 8.2, the ODS PRINTER statement has steadily become more versatile, more flexible, and more robust. Working closely with Technical Support and the balloting system, we have added new items in response to our users' needs. Inline formatting, table-of-contents handling, and Unicode support are some of the many new enhancements available to users of SAS® 9.2. With all the new changes, ODS PRINTER will take your output into the twenty-first century and beyond.

INTRODUCTION

Staying on top of things is crucial in today's world. How your work is presented is important in creating that great first impression. ODS PRINTER (which includes Printer, PCL, PostScript, and PDF destinations) is constantly changing to provide you with the best possible output.

This paper highlights several new features that have been implemented since SAS®9. It demonstrates several examples of the new options that are available to you now in SAS 9.1 and coming in SAS 9.2.

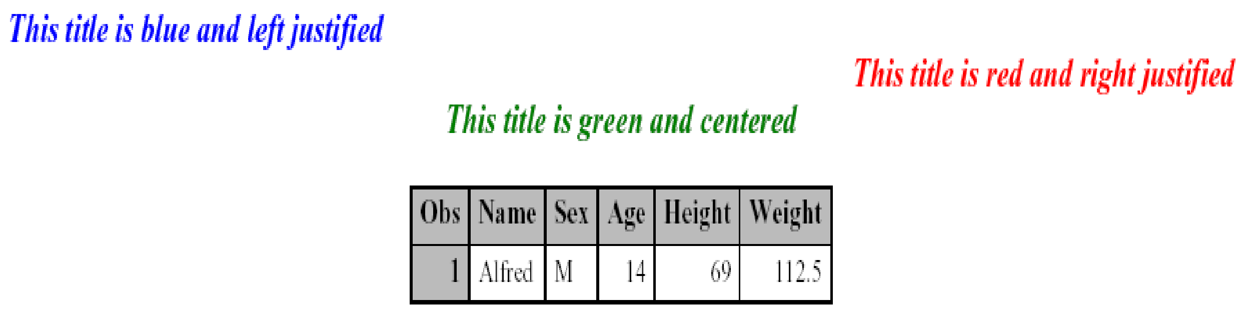
NEW WAYS TO FORMAT YOUR OUTPUT

BASIC INLINE FORMATTING

Simple inline formatting was made available in SAS 8.2. Using the ODS ESCAPECHAR statement and inline syntax you can change things such as the justification or the color of your title or text statements.

```
options nonumber nodate;
ods escapechar = "^";
ods pdf file="example1.pdf";
title1 "^S={just=L color=blue}This title is blue and left justified";
title2 "^S={just=R color=red}This title is red and right justified";
title3 "^S={just=C color=green}This title is green and centered";
proc print data=sashelp.class(obs=1);run;
ods _all_ close;
```

The output looks like this:



This title is blue and left justified

This title is red and right justified

This title is green and centered

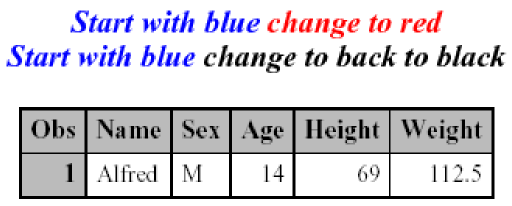
Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5

It's important to remember that the specified style will stay in effect until the end of the string or until you specify another style. After closing the style, you can start another or just use the default style. To stop the currently used style and reset the style back to the default setting, use this syntax: `^S={}`. Here is another example, but this time the code stops using one style and then implements another:

```
options nonumber nodate;
ods escapechar = "^";
ods pdf file="example2.pdf";
title1 "^S={color=blue}Start with blue ^S={color=red}change to red";
title2 "^S={color=blue}Start with blue ^S={}change to back to black";
```

```
   proc print data=sashelp.class(obs=1);
   run;
   ods _all_ close;
```

The output looks like this:



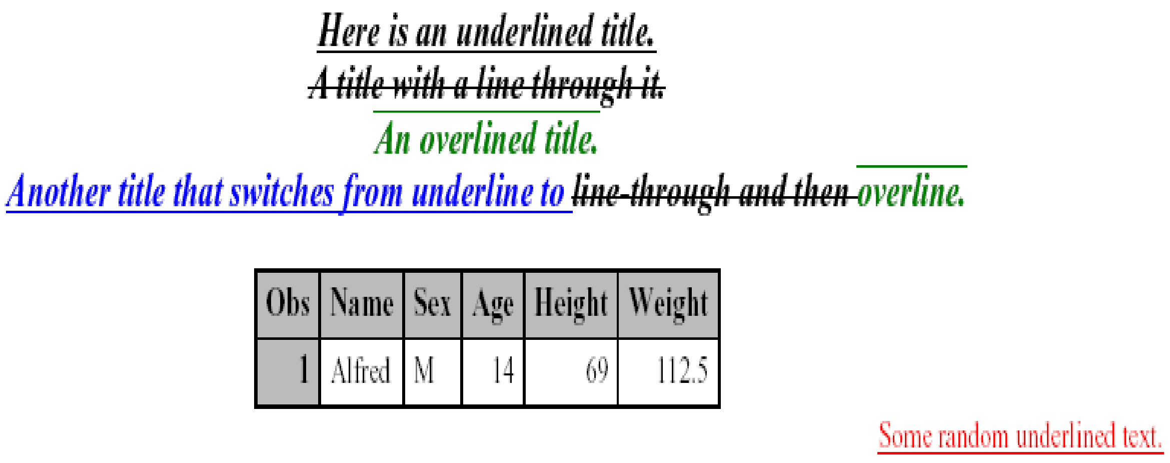
Start with blue change to red
Start with blue change to back to black

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5

ADDING NEW STYLE ATTRIBUTES
Style attributes other than color are useful too. Font size, underlining, overlining, and strikethrough are other style attributes that can make output stand out. ODS PRINTER in SAS 9.2 will support two style settings for underlining. ODS PRINTER recognizes the SAS/GRAPH syntax `UNDERLIN=1,2,3` for underlining text but it does not change the thickness of the line. The new style element for ODS PRINTER in SAS 9.2 is TextDecoration. It allows you to set underline, overline, or strike-through on titles and text strings. Here is an example showing the three styles that the TextDecoration style element provides in SAS 9.2:

```
   options nodate nonumber;
   ods pdf file="line.pdf";
   ods escapechar='^';
   title underlin=1 "Here is an underlined title.";
   title2 "^S={textdecoration=line_through}A title with a line through it.";
   title3 "^S={textdecoration=overline color=green}An overlined title.";
   title4 "^S={textdecoration=underline color=blue}Another title that switches from
   underline to ^S={textdecoration=line_through}line-through and then
   ^S={textdecoration=overline color=green}overline.";

   proc print data=sashelp.class(obs=1);run;

   ods pdf text="^S={just=r textdecoration=underline color=red}Some random underlined
   text.";
   ods _all_ close;
```

The output looks like this:



Here is an underlined title.
~~*A title with a line through it.*~~
An overlined title.
Another title that switches from underline to ~~line-through and then~~ overline.

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5

Some random underlined text.

NESTING INLINE STYLE ATTRIBUTES
Also new in SAS 9.2 is nested inline formatting. This feature enables users to set several style attributes for a string without resetting the previously used style. Users can start a string with one set of style attributes and add to them later in the string. The first thing to notice is the new syntax. The old syntax (**^S={}**) is still valid, but to use the new SAS 9.2 nested inline formatting, use the new syntax as follows:

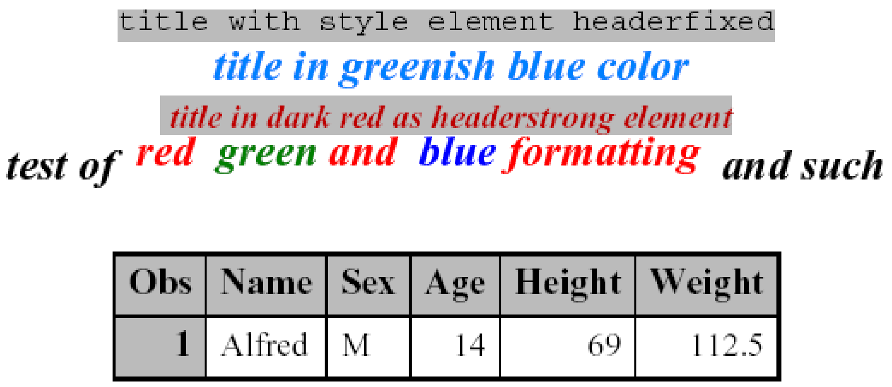
```
^{style <style-element-name><[style-attribute-specification(s)]>
formatted text}
```

The syntax begins with the function name Style. Next you can add a style element like Headerfixed, SystemTitle, and so on, as needed. Then you can add new attributes such as fontstyle, color, and so on within brackets. The syntax then ends with the text you want to format.

Here is an example showing how to use the new syntax:

```
options nodate nonumber;
ods escapechar="^" ;
ods pdf file="test.pdf";

title "^{style headerfixed title with style element headerfixed}";
title2 "^{style [color = greenish blue] title in greenish blue color}";
title3 "^{style headerstrong[color = dark red fontstyle=italic] title in dark red as
headerstrong element}";

title4 "test of ^{super ^{style [color=red] red ^{style [color=green] green} and
^{style [color=blue] blue }
formatting }} and such" ;

proc print data=sashelp.class(obs=1) ;
run;
ods _all_ close ;
```

The output looks like this:



```
title with style element headerfixed
title in greenish blue color
title in dark red as headerstrong element
test of red green and blue formatting and such
```

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5

In the example let's examine the last title statement to see how the nesting inline formatting works. `^{super <text>` is invoked to start using the superscript function. Then we use the style function to add another style attribute `^{style foreground=red <text>` for our text. Think of this as a FILO (first in – last out) stack that we are building. We push the superscript function onto the stack. Then we use the style function to push a red color onto the stack. Our result is a text string that is superscripted and red. We continue by pushing a green color onto the stack. Because the new style attribute is a color we change the text to the new color. Following along our string, we close this style attribute with a close bracket. After the green color is closed or popped off the stack the red color is now the active style attribute for the text. Now we push a blue color onto the stack and the text string uses that color. After we close the blue color we proceed to close the red and superscript. Now our stack is empty so we use the default style attributes to finish up our text string.

To help show how this FILO stack works with nested style attributes, here is a text representation showing how attributes get pushed and pulled from the stack.

Default style attribute (text is normal with a black color)
Super (text is superscript and the color is black)
Red (text is superscript and color is red)
Green (text is superscript and color is green)
Red (text is superscript and color is red)
Blue (text is superscript and color is blue)
Red (text is superscript and color is red)
Default style attribute (text is normal with a black color)

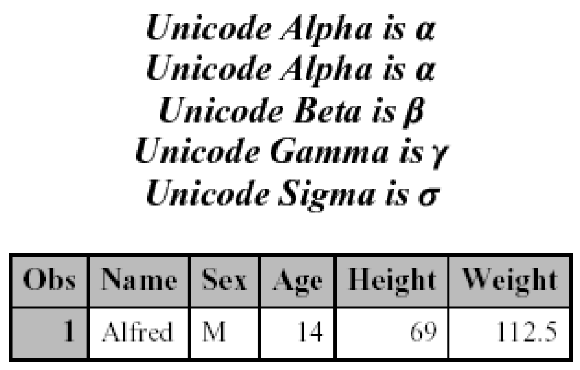
By allowing nested styles, we can dramatically change our titles and text strings without having to close styles and restart new styles.

USING UNICODE SYMBOLS IN ODS PRINTER

Another new SAS 9.2 enhancement is the ability to incorporate Unicode symbols such as Greek symbols into your output. The new style function is Unicode, and the syntax is `^{unicode <value>)`. It is similar to other inline style function syntax but also unique in that you can put a 4-digit Unicode value in OR a value from a predefined list stored as a tagset template. First the example:

```
options nonumber nodate;
ods escapechar='^';
ods pdf file="foo.pdf";

title  'Unicode Alpha is ^{unicode 03B1}';
title2 'Unicode Alpha is ^{unicode alpha}';
title3 'Unicode Beta is ^{unicode 03B2}';
title4 'Unicode Gamma is ^{unicode gamma}';
title5 'Unicode Sigma is ^{unicode 03C3}';

proc print data=sashelp.class(obs=1); run;
ods _all_ close;
```

The output looks like this:

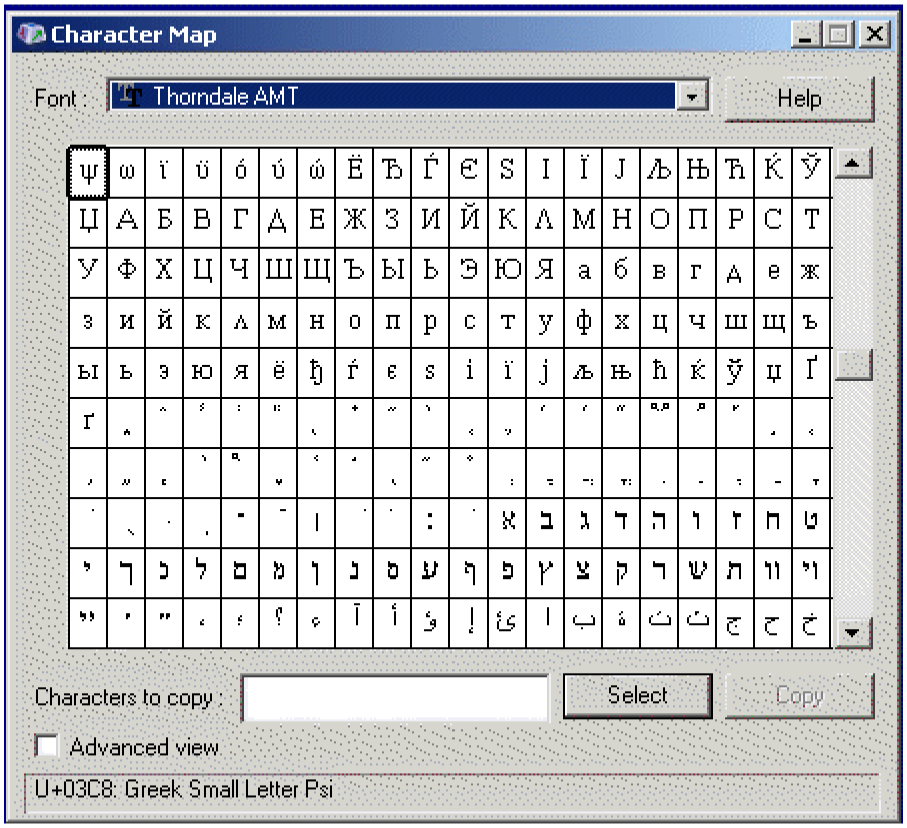


Unicode Alpha is α
Unicode Alpha is α
Unicode Beta is β
Unicode Gamma is γ
Unicode Sigma is σ

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5

Using the Unicode function requires more explanation. There is a new tagset template that contains a predefined list of common Greek symbols and their Unicode values. You can update this template as needed. The template increases the flexibility of this new inline style function. Looking at the old inline style functions, `^{dagger}` and `^{sigma}`, we felt it was important to maintain their ease of use. So any of the values that are listed in the template can easily be called by using this syntax **^{<value>}**. The special sequence used to produce the copyright, trademark, and registered trademark symbols have been replaced. You can no longer use the special sequences of '01,' '02,' and '04' to designate these values because they are also valid Unicode values.

So how can you find out what symbols are available for your ODS PRINTER output? For example, in Windows XP you can click on **Start →Programs → Accessories → System Tools → Character Map**. You will see the screen shown in Figure 1.



Font : Thorndale AMT

U+03C8: Greek Small Letter Psi

Figure 1. Character Map Window

The screen in Figure 1 shows the character map of the Thorndale AMT font. Listed here are all the available symbols that can be displayed using this font. The symbol for Greek Small Letter Psi is highlighted. At the bottom you can see the Unicode value 03C8 for this symbol. By using this Unicode value in the new UNICODE style function we can now display this symbol in our ODS PRINTER output.

You can use the UNICODE function to select any symbol from the character map of the font that is currently selected. Simply find the 4-digit Unicode value for the character and use it as the argument to the UNICODE function. You can also create a mnemonic for commonly use Unicode characters using PROC TEMPLATE.

The TAGSETS.CORE tagset contains a table of Unicode values and their mnemonics. To add or change a mnemonic, you must specify the tagset with the SOURCE statement, make the desired changes, and then run the modified tagset. The following code creates a file called `core.tpl` in the current directory that contains the TAGSETS.CORE tagset:

```
proc template;
     source tagsets.grandparent / file="core.tpl"; run;
```

Open the file. You will notice some text like this:

```
set $unicodeMap["ALPHA" ] "03B1";
set $unicodeMap["BETA" ] "03B2";
set $unicodeMap["DAGGER" ] "2020";
```

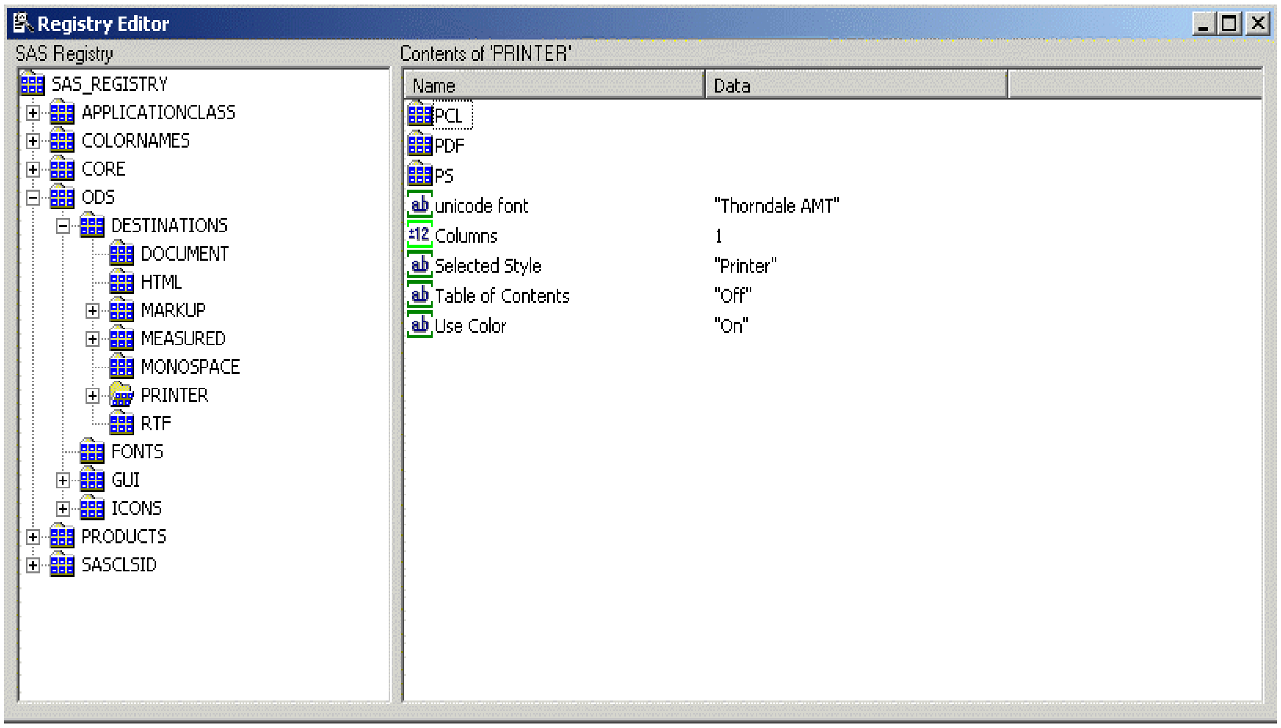
Now update the file with your new mnemonic and corresponding Unicode value.

```
set $unicodeMap["<new function name>" ] "<Unicode value>";
```

Save the file, and compile the modified tagset using PROC TEMPLATE. The modified tagset will be stored in the first writable template store in your ODS path.

```
proc template;
     %inc "core.tpl";
run;
```

An additional feature of this new functionality is a new registry setting in SAS that holds the Unicode font value. Its initial value will be Thorndale AMT (more about this font later) but it can be changed as needed. The new registry setting within SAS can be seen if you invoke **regedit** to bring up the registry editor (Figure 2).



Name	Data
PCL	
PDF	
PS	
unicode font	"Thorndale AMT"
Columns	1
Selected Style	"Printer"
Table of Contents	"Off"
Use Color	"On"

Figure 2. Registry Editor in SAS

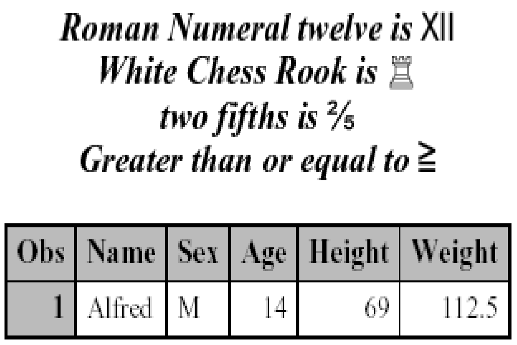
Users can modify the setting for **unicode font** with any valid font that is installed on their computer and is recognized by SAS. To ensure that SAS recognizes all the fonts that are installed on your computer, you will need to run PROC FONTREG. The FONTREG procedure enables you to update the SAS registry to include system fonts, which can then be used in SAS output. Here is sample code using PROC FONTREG:

```
proc fontreg mode=all;
fontpath 'c:\winnt\fonts';
run;
```

By specifying which font to use by the Unicode element, you can display many interesting symbols. The following example shows a few possibilities, using the Arial Unicode MS font.

```
options nonumber nodate;
ods escapechar='^';
ods pdf file="foo.pdf" notoc;

title  'Roman Numeral twelve is ^{unicode 216B}';
title2 'White Chess Rook is ^{unicode 2656}';
title3 'two fifths is ^{unicode 2156}';
title4 'Greater than or equal to ^{unicode 2267}';

proc print data=sashelp.class(obs=1);
run;
ods _all_ close;
```

The output looks like this:



Roman Numeral twelve is XII
White Chess Rook is ♖
two fifths is ⅖
Greater than or equal to ≧

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5

NEW FONTS IN SAS 9.2
The Thorndale AMT font that was mentioned above is not familiar to most users. SAS now licenses three font families from Agfa Monotype Corporation. Each of those font families has four members. In addition to these three font families, there are 10 other fonts licensed to SAS. Two are symbol fonts and eight have support for Asian languages. These typefaces will yield 100% metric compatibility to the Microsoft Core Fonts when used in the same computing environment:

Albany AMT = Arial
Thorndale AMT = Times New Roman
Cumberland AMT = Courier New

Starting in SAS 9.2, using these fonts enables SAS applications and output to display and print text consistently, portably, efficiently, and internationally. These fonts can be rendered in weights and styles of normal, italic, bold, and bold italic. The fonts are available on every platform that SAS ships.

With these new True Type fonts also come the possibility of larger file sizes when embedding is enabled. In SAS 9.2 four new system options will be available to control file sizes. Here is a brief description of the new options:

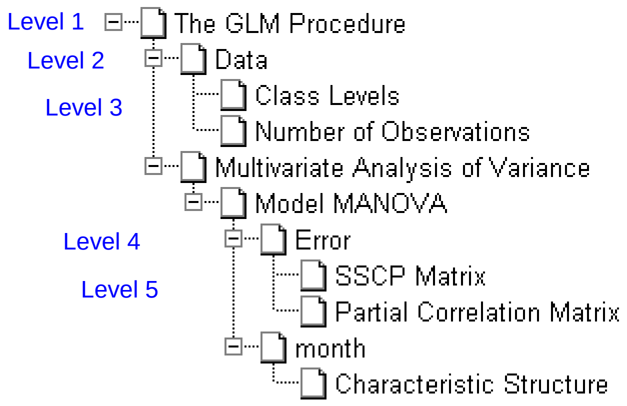
1. FONTEMBEDDING / NOFONTEMBEDDING – this enables / disables font embedding. This option allows more direct user control over font embedding and, therefore, output file size. For SAS 9.2 this option is only available for PDF and Scalable Vector Graphics (SVG) drivers. Postscript currently ignores this option, and other drivers render the fonts as part of the image. The default for this option is FONTEMBEDDING.
2. UPRINTCOMPRESSION / NOUPRINTCOMPRESSION – this enables / disables UP compression. Used with the next two options, this option is a master switch that enables / disables file compression. The default for this option is UPRINTCOMPRESSION. The alias is UPC / NOUPC.
3. DEFLATION=n where n is 0 to 9 and 6 is the default. This option controls the level of data compression that is used by the Deflate Compression method. It is only used by the PDF and SVG drivers. The alias is DEFLATE.
4. JPEGQUALITY=n where n ranges from 0 to 100 and 75 is the default. This option controls image quality and compression such that image quality decreases as compression increases. A value of 0 produces the lowest quality image and highest level of compression.

More information on these new options and their use will be available soon on the SAS Technical Support Web site: support.sas.com.

COLLAPSING THE TABLE OF CONTENTS
One last addition to formatting in SAS 9.2 is the handling of Table of Contents (TOC) in the PDF destination. TOCs are helpful, especially when PDF files have numerous pages. One problem with the TOC is that it can have so many levels that it is hard to find that one link that you need. In response to past suggestions, SAS introduces TOC compression. By setting the PDFTOC option on your ODS PDF file statement you can set the level of collapsing that you would like your TOC to display when you initially open the file. For example, running a sample data set with PROC GLM creates a large TOC. Using the new PDFTOC option, you can collapse the TOC to be more manageable if desired.

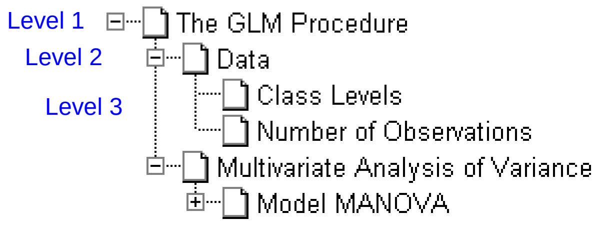
```
options nonumber nodate;
title1 'notice compression of TOC';
ods pdf file="test.pdf" pdftoc=3;

data one;
  do month = 1 to 12;
    age  = 2 + 0.3*rannor(345467);
    age2 = 3 + 0.3*rannor(345467);
    age3 = 4 + 0.4*rannor(345467);
    output;
    end;
  run;

proc glm;
  class month;
  model age age2 age3=month / nouni; manova h=month /printe;
  run;
ods _all_ close;
```

Without the PDFTOC option in the file statement, here is how the TOC looks when it is opened. You can see that it is 5 levels deep.



```
The GLM Procedure
  Data
    Class Levels
    Number of Observations
  Multivariate Analysis of Variance
    Model MANOVA
      Error
        SSCP Matrix
        Partial Correlation Matrix
      month
        Characteristic Structure
```

However, in our example we specify **`pdftoc=3`**; here is how that TOC looks:



```
The GLM Procedure
  Data
    Class Levels
    Number of Observations
  Multivariate Analysis of Variance
    Model MANOVA
```

The default for PDFTOC is 0, which will result in full expansion.

FORMATTING YOUR OUTPUT - CONCLUSION
Text decoration, nested inline styles, Greek symbols / Unicode support, and TOC collapsing are new ways to enhance your output in ODS PRINTER. Used individually or together, these enhancements easily add a new look to your output.

NEW WAYS TO SEPARATE YOUR OUTPUT

ODS PRINTER creates an output file of your choice such as PDF or PostScript file. A new option has been added that enables you to create several new output files from one SAS program. The NEWFILE option allows users to determine a starting point in their output. When a starting point is reached, a new output file is created. ODS

PRINTER will automatically increment the name of the new file based on the original file name. If your file name is `output.pdf` and you use the NEWFILE option, additional files are named `output1.pdf`, `output2.pdf`, and so on. The syntax is simple; on the file statement, just add **`NEWFILE=<`*`starting-point`*`>`**. ***`Starting-point`*** can be one of the following:

BYGROUP	start a new file for each bygroup value
TABLE / OUTPUT	start a new file for each output object
PAGE	start a new file for each page of output. When an explicit page break is sent to the output then a new file will be created.
PROC	start a new file for each procedure
NONE	the default; everything is written to the currently open file

Let's sort and print `sashelp.class` with the NEWFILE=BYGROUP option. First we must sort the file in the bygroup order that we want:

```
ods pdf file="newfile.pdf" newfile=bygroup;
proc sort in=sashelp.class out=sorted_class; by age; run;
proc print data=sorted_class; by age; run;
ods _all_ close;
```

Running the code creates six new files. Below are some resulting log messages that you see:

```
NOTE: ODS PDF printed 1 page to C:\temp\newfile.pdf.
NOTE: ODS PDF printed 1 page to C:\temp\newfile1.pdf.
NOTE: ODS PDF printed 1 page to C:\temp\newfile2.pdf.
NOTE: ODS PDF printed 1 page to C:\temp\newfile3.pdf.
NOTE: ODS PDF printed 1 page to C:\temp\newfile4.pdf.
NOTE: ODS PDF printed 1 page to C:\temp\newfile5.pdf.
```

Each file contains observations for one data set, with the data ordered by the BYGROUP option, which in this example is by each value of AGE. Having separate physical files can be useful and eliminate the tedious work of manually cutting and pasting files together to create new files. The NEWFILE option was introduced with ODS HTML but is now production in SAS 9.2 with ODS PRINTER.

NEW OUTPUT FOR THE ODS PRINTER DESTINATION

SVG (Scalable Vector Graphics) files can now be created within the ODS PRINTER statement. SVG is a language for describing two-dimensional graphics and graphical applications in XML. SVG is an application of XML and thus supports Unicode, which defines a standard universal character set. This driver is being implemented with SAS version 9.2. By default, ODS PRINTER sets the device and target device to SASPRTC. The syntax is easy. Use the GOPTIONS and PRINTERPATH statements to specify the new SVG driver and update your ODS PRINTER statement to create a SVG file. For example:

```
options printerpath='SVG';
goptions dev=SVG;
title1 'Scalable Vector Graphics';
ods printer;
proc gchart data=sashelp.class;
vbar name/sumvar=height;
run;
quit;
ods printer close;
```

This creates a file named `sasprt.svg` which can be viewed in your Web browser. The Adobe SVG Viewer plugin (http://www.adobe.com/svg/viewer/install/main.html) must be installed to view SVG files. This file can also be embedded as a component into a parent Web page, such as an HTML or XHTML page.

You could add more procedures such as PRINT and GSLIDE in the SAS program, but you might want to use the NEWFILE option mentioned above to put each procedural new page into a separate file.

Additional information on SVG and this new destination within ODS PRINTER will be available soon on the SAS Technical Support Web site (support.sas.com).

CONCLUSION

As the examples show, many new features and options are available in SAS 9.2 for the ODS PRINTER statement. The suggestions from customers and input from Technical Support have driven these enhancements. All the improvements increase the functionality of the product and give SAS users more power and flexibility when creating ODS PRINTER output. ODS PRINTER is always evolving and will continue to move forward into the twenty-first century and beyond.

ACKNOWLEDGMENTS

The author thanks Helen Wolfson, Bari Lawhorn, Chevell Parker, Donna Antle, and Woody Middleton for their contributions to this paper.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author:

Scott Huntley
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
Scott.Huntley@sas.com