PharmaSUG 2014 - Paper TT15

"Ma, How Long Do I Cook The Turkey For?"

David Franklin, TheProgrammersCabin.com, Litchfield, NH

ABSTRACT

In November it will be Thanksgiving and the cry will go out from the kitchen in some households "Ma, How Long Do I Cook The Turkey For?" This paper aims to give a light-hearted introduction to ODS RTF output using information of cooking times for that traditional of meats' at Thanksgiving, the turkey. Along the way there will be some turkey facts added in for little bit of light relief.

INTRODUCTION

On November 27th, 2014, turkeys across the nation will be un-wrapped and put in the oven for the Thanksgiving meal later in the day. Most will have instructions on how long to cook the turkey, but if you, like me, buy one off the farm as opposed to the supermarket, you don't have those instructions. This paper gives an example of creating a table and a graph using ODS/RTF of time needed to cook your turkey along with some tips and other information relating to the turkey.

One of the biggest misapprehensions of using ODS/RTF in particular is setting up the template, using the TEMPLATE procedure. In the example of PROC TEMPLATE code given here, this has been used for many years as a standard template. Also some programmers are apprehensive about moving to ODS/RTF because it is different – the most notable example is that instead of the options CENTER and WIDTH= on the DEFINE statement within the REPORT procedure call is replaced with a STYLE option with JUST=CENTER CELLWIDTH=. RTF tags that are shown in the following examples are not required but do demonstrate that using these tags can enhance the output significantly.

Now lets start on our journey.

COOKING THE TURKEY

WHAT YOU NEED

- 1 Turkey
- Kosher salt and ground pepper
- Dried herbs and spices of choice: sage, thyme, garlic powder, onion powder
- Vegetable oil

PREPARATION

- Preheat the oven to 325°F.
- Brush the turkey breast with vegetable oil. Sprinkle liberally with salt, pepper, herbs, and spices.
- Insert meat thermometer to the center of the breast. Wrap and seal the breast in extra-heavy aluminum foil, with the face of the meat thermometer on the outside of the foil for viewing. Place on a wire rack in a shallow roasting pan.
- Roast the turkey until the meat thermometer reaches 160 degrees F. Open the aluminum foil about 30 minutes before the turkey is done and baste two times with the drippings.

CALCULATING THE TIME TO COOK THE TURKEY

There are MANY calculations for time to cook your turkey for Thanksgiving dinner. For the purposes here two general formula will be used for this paper:

unstuffed: time(minutes) = weight(lbs) * 20
stuffed: time(minutes) = weight(lbs) * 25

In both cases the temperature of the oven is preheated to 325°F and the turkey thawed beforehand. It must be stressed that this is a quick formula only and the USDA offers guidance on the internal temperatures required for a turkey deemed to be cooked (USDA guidance released April 5, 2006: "A whole turkey (and turkey parts) is safe when cooked to a minimum internal temperature of 165°F as measured with a food thermometer. Check the internal temperature in the innermost part of the thigh and wing and the thickest part of the breast. For reasons of personal preference, consumers may choose to cook turkey to higher temperatures.").

We could simply do a calculation and have the times but the purpose of this paper is to create a table and a graph showing the times using ODS RTF.

PRODUCE A TABLE AND GRAPH OF COOKING TIMES

The very first thing I want to do in my program is initialize my options and the ODS path for the program:

```
options pageno=1 nodate soruce2 mergenoby=warn msglevel=i nobyline
        orientation=landscape nonumber symbolgen mprint mlogic mprintnest
        formchar="|———|+|——+=|—/\<>*"; *Set general options, including setting
                                          output to orientation=landscape;
goptions reset=all; *Reset all my graphic options;

ods escapechar '!'; *Set a character to set start of raw RTF code sequences;
libname library "E:\Users\Franklin\Utils\Template"; *Location of custom template;
ods path library.styles(read)
         sasuser.templat(read)
         sashelp.tmplmst(read); *Search order for templates;
ods path show; *Show in the log the path of templates to be used;
```

The ODS ESCAPECHAR setting is most interesting -- this is where it is possible to put a RTF code sequence inside the output document, like writing raw RTF code into the document. SAS can do some things well but occasionally, as we shall see below, there is a need to trick SAS into writing specific codes (there are not many but this paper will show a couple of these that are the most useful).

The code used to create the custom template is in Appendix A. There are several parts to the setting up of the template (and there are papers around that go into detail of setting this up) but this is beyond the scope of this paper. However, there are two sections that are of interest, first the setting of the fonts:

```
replace fonts /
   'TitleFont2' = ("Times New Roman", 8pt)
   'TitleFont' = ("Times New Roman", 8pt)
   'StrongFont' = ("Times New Roman", 8pt)
   'EmphasisFont' = ("Times New Roman", 8pt, Bold)
   'FixedEmphasisFont' = ("Times New Roman", 8pt, Bold)
   'FixedStrongFont' = ("Times New Roman", 8pt)
   'FixedHeadingFont' = ("Times New Roman", 8pt)
   'BatchFixedFont' = ("Times New Roman", 8pt)
   'FixedFont' = ("Times New Roman", 8pt)
   'headingEmphasisFont' = ("Times New Roman", 8pt)
   'headingFont' = ("Times New Roman", 8pt)
   'docFont' = ("Times New Roman", 8pt);
```

Here we have set the font to a Times New Roman 8pt font for our output, but it is possible to use any font you may have on your computer. However it cannot be assumed that the person receiving the output generated has that font installed so it is best to stick to Times New Roman, Courier New or Arial as these are generally available.

The second section of code that is interesting is the margin settings as given below:

```
replace Body from Document /
   bottommargin = 1.0in
   topmargin = 1.0in
   rightmargin = 1.0in
```

```
    leftmargin = 1.0in
    pagebreakhtml = html('PageBreakLine');
```

While these can be changed I usually stick to the 1in margins all around.

The next set of code in our program will create the dataset we want to use to create the table and the graph:

```
data turkey0;
   do weight=5 to 20;
      time_usf=weight*20; *Un-stuffed time;
      time_stf=weight*25; *Stuffed time;
      output;
   end;
run;
```

This datastep creates three variables -- weight in pounds (weight), time in minutes to cook if the turkey is unstuffed (time_usf) and time to cook in minutes if the turkey is stuffed (time_stf).

Now we set up the titles and footnotes that are common for our two outputs:

```
title1 j=l "Cook and Eat (Yum Yum), Inc." j=r "Page !{pageof}";
title2 j=l "Turkey";
footnote1 '!R"\brdrt\brdrs\brdrw20 "'
          "Program Name: E:\Users\Franklin\tmp\Turkey.sas "
          "Creation Date and Time: &sysdate9:&systime";
footnote2 '!R"\fi-720\li720\tx720 Note:\tab USDA guidance released '
          'April 5, 2006: \ldblquote\i A whole turkey (and turkey parts) is '
          'safe when cooked to a minimum internal temperature of 165°F as '
          'measured with a food thermometer. Check the internal '
          'temperature in the innermost part of the thigh and wing and the '
          'thickest part of the breast. For reasons of personal '
          'preference, consumers may choose to cook turkey to higher '
          'temperatures.\i0\rdblquote "';
```

One of the advantages of ODS RTF is that we can split a title into three sections -- a left, center and right. In TITLE1 two sections are created, a left with the text "Cook and Eat (Yum Yum), Inc." and a section of text that will create "Page x of y" where x is the page number and y is the number of pages in the document. Note the "!" before the "{pageof}" that tells SAS to output the text as RTF code.

The FOOTNOTE1 text '!R"\brdrt\brdrs\brdrw20 "' is another interesting use of RTF text that adds a border above the first line of text -- note that the RTF text is surrounded in double quotes and is preceded by a '!R' sequence, again telling SAS to put this out as pure RTF text.

FOOTNOTE2 has its own interesting use of RTF. The code "\fi-720\li720\tx720" sets up the format of a line indent of in 0.5in (a value of 1440=1in so 720=0.5in) but putting the first line back to the margin, and setting up a tab at 0.5in. The "\tab" after the word "Note:" puts a tab character in the text just as you would get if you were typing the text. The last code of interest is the "\ldblquote\i" and "\i0\rdblquote" sequences which switch on and off quoting the italics for the text surrounding these sequences.

Now to generate the table:

```
*Close our LISTING destination and open an RTF destination using our custom template;
ods listing close;
ods rtf file="E:\Users\Franklin\tmp\Turkey.rtf" style=TPC_std
        nogtitle nogfootnote; *Output file and location, and use procedure titles
                               and footnotes;

*Set up a RTF sequence for underlines;
%let undrln=%str(!R'\brdrb\brdrs\brdrw20 ');

*Titles for the table;
```

```
title3 j=c "Table 1";
title4 j=c "Cooking Times for Turkey, Thawed (Oven Pre-heated to 325F)";

*Generate the table;
proc report data=turkey0 nowindows headline split='|';
   columns weight
      ("&undrln.Turkey Cooking Time|(minutes)"
      (time_usf time_stf));
   define weight /display order=internal 'Weight (lbs)'
      style={cellwidth=1in just=left protectspecialchars=off
      pretext='\tqdec\tx720 '};
   define time_usf /display 'Unstuffed'
      style(column)={cellwidth=1in just=center};
   define time_stf /display 'Unstuffed'
      style(column)={cellwidth=1in just=center};
   quit;
run;
```

This section of code has four steps -- close our LISTING destination and open an RTF destination using our custom template; set up a RTF sequence for underlines (this can be set beforehand but am showing this in this section); set titles for the table (these were not in the general section and note they are centered); and finally generate the table.

The use of the &undrln will put a underline under the header "&undrln.Turkey Cooking Time|(minutes)" without losing a line in the output.

The times of Unstuffed and Stuffed are centered and use the STYLE statement of

```
style(column)={cellwidth=1in just=center};
```

which is set the column to 1in wide and centered.

The display of the WEIGHT variable is decimal aligned and uses the code:

```
style={cellwidth=1in just=left
       protectspecialchars=off pretext='\tqdec\tx720 '};
```

An explanation of the use of styles is beyond this paper but this code will decimal align the numbers to 0.5in (remember the a value of 1440=1in so 720=0.5in as given above) -- SAS effectively left aligns the output and adds text in the RTF to align it to a decimal centered tab of 0.5in in this case.

The output from the PROC REPORT call is in Figure1, below.

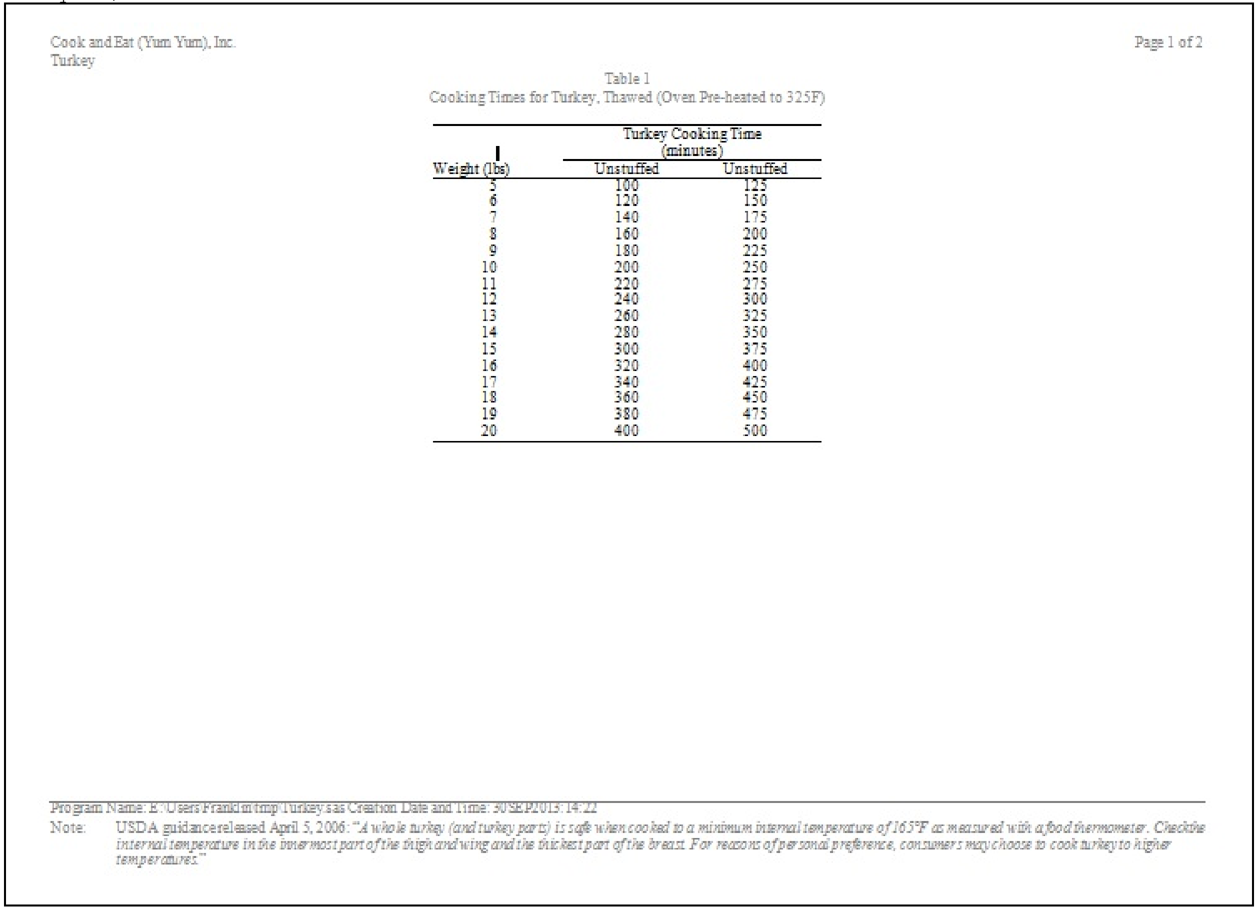
Next is the figure.

```
*Set our options for the figure;
goptions device=png target=png ftext="swissb"
xpixels=4375 ypixels=2200 xmax=8.75 in ymax=4.9 in;
axis1 label=(h=1 'Weight (lbs)') offset=(0.10 in) width=2 value=(h=1.0);
axis2 order=(100 to 500 by 50) value=(h=1.0) length=2.90 in
      label=(h=1.0 a=90 'Time (minutes)') minor=(h=0.5 w=0.5) width=2;
symbol1 value=dot h=1.0 width=3 line=1 i=join color=red;
symbol2 value=square h=1.4 width=3 line=1 i=join color=blue;
legend1 label=('Turkey Type') position=(bottom right inside)
        value=(tick=1 'Unstuffed' tick=2 'Stuffed');

*Set the titles;
title3 "Figure 1";
title4 "Cooking Times for Turkey, Thawed, Oven Pre-heated to 325F";

*Produce the plot;
```

```
proc gplot data=turkey0;
plot time_usf*weight=1 time_stf*weight=2
     / overlay noframe legend=legend1 haxis=axis1 vaxis=axis2;
run;
quit;
```



Cook and Eat (Yum Yum), Inc.
Turkey

Page 1 of 2

Table 1
Cooking Times for Turkey, Thawed (Oven Pre-heated to 325F)

	Turkey Cooking Time (minutes)	
Weight (lbs)	Unstuffed	Unstuffed
5	100	125
6	120	150
7	140	175
8	160	200
9	180	225
10	200	250
11	220	275
12	240	300
13	260	325
14	280	350
15	300	375
16	320	400
17	340	425
18	360	450
19	380	475
20	400	500

Program Name: E:\Users\Franklin\tmp\Turkey.sas Creation Date and Time: 30SEP2013:14:22
Note: USDA guidancereleased April 5, 2006: *"A whole turkey (and turkey parts) is safe when cooked to a minimum internal temperature of 165°F as measured with a food thermometer. Checkthe internal temperature in the innermost part of the thigh and wing and the thickest part of the breast. For reasons of personal preference, consumers may choose to cook turkey to higher temperatures."*

Figure 1: Output from PROC REPORT showing table of cooking times for Stuffed and Unstuffed Turkey

The output for the figure is given in Figure 2.

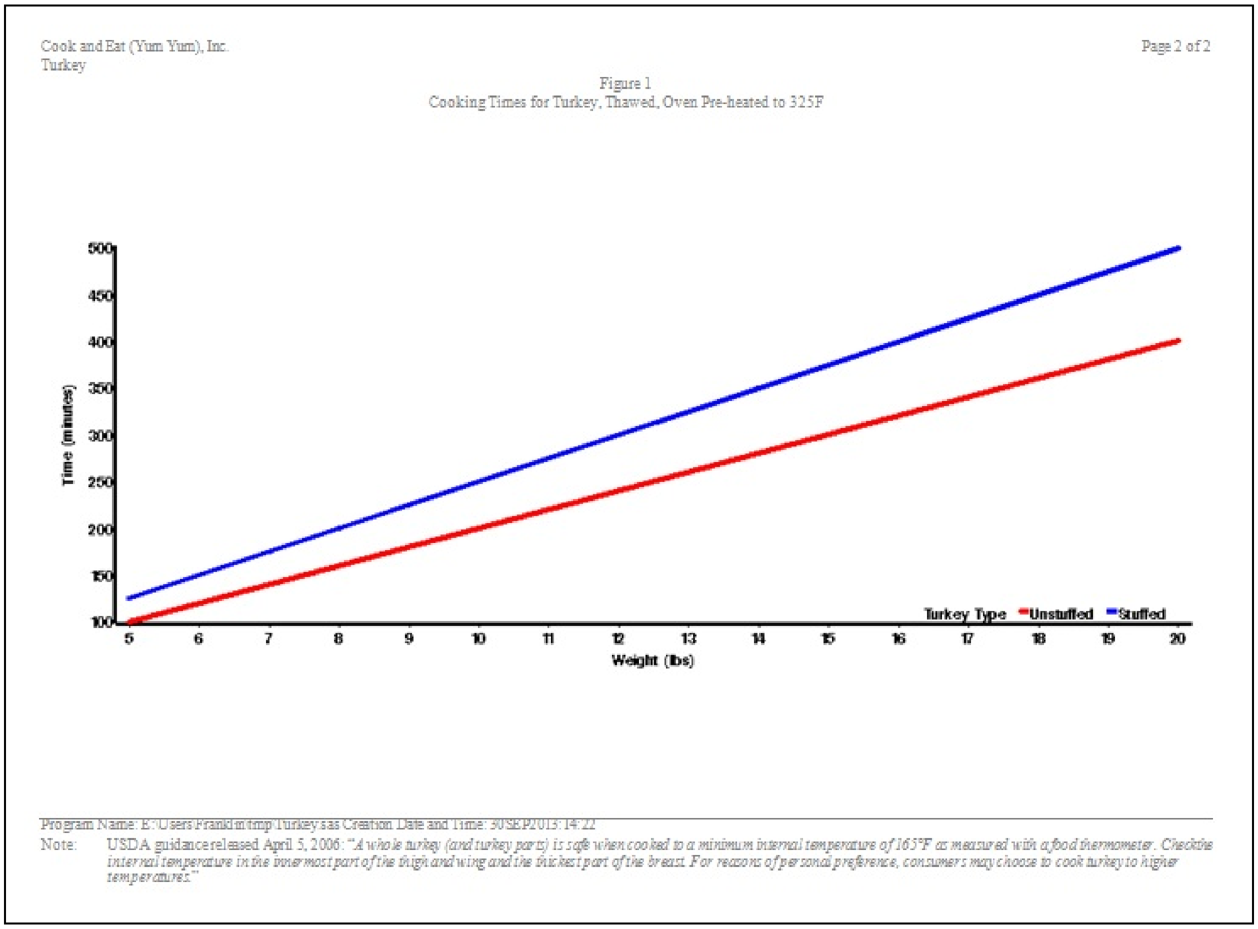
Note that this code is the same as if the request was for a PNG format file but because it is still enclosed in an ODS RTF call the output is inserted into the RTF file.

Finally, to close the RTF file that is being created, and restart the LISTING destination the code would be:

```
ods rtf close;
ods listing;
options number;
run;
```

After the last step has run the RTF file created is available to be read using software such as Microsoft Word.

One thing to note is that the file was created using an RTF extension but it could be a DOC extension if your users are unsure of the RTF extension.



Cook and Eat (Yum Yum), Inc.
Turkey

Page 2 of 2

Figure 1
Cooking Times for Turkey, Thawed, Oven Pre-heated to 325F

Program Name: E:\Users\Franklin\tmp\Turkey.sas Creation Date and Time: 30SEP2013:14:22
Note: USDA guidance released April 5, 2006: "*A whole turkey (and turkey parts) is safe when cooked to a minimum internal temperature of 165°F as measured with a food thermometer. Check the internal temperature in the innermost part of the thigh and wing and the thickest part of the breast. For reasons of personal preference, consumers may choose to cook turkey to higher temperatures.*"

Figure 2: Output from PROC GPLOT showing graphs of cooking times for Stuffed and Unstuffed Turkey

The Stuffing
Important for any turkey on the table at Thanksgiving:

- 1 large loaf stale white bread (1-1/2 pounds)
- 1 cup diced celery
- 1 Tbsp chopped onion
- 1/4 cup butter or rendered turkey fat
- 2 tsp poultry seasoning
- 1-1/2 tsp salt and 1/8 tsp black pepper
- 3/4 cup cooled broth from cooking giblets (or chicken stock)

Remove crusts from bread and cut in 1-inch dice. Saute celery and onion in butter until soft and yellow. Add bread and seasonings and toss together until well mixed. Cool. Add broth last and again toss until mixed. Stuff into turkey.

SOME INTERESTING FACTS ABOUT THE TURKEY

- When turkeys reach maturity they can have as many as 3,500 feathers
- Wild turkeys can fly up to 55 miles per hour
- Wild turkeys spend the night in trees
- 45 million turkeys are eaten each Thanksgiving

CONCLUSION

A table and a graph for cooking a turkey on Thanksgiving day is generated and ready for all to see. This was done using a standard formula for cooking the turkey and some simple code using ODS/RTF. The output was produced in one step and could be quite easily taken and put into any publication with ease.

For those considering the move to ODS/RTF, despite the very small cost in learning how to program for this format, the benefits of using ODS RTF are substantial in that production ready output is available to the customer immediately and they can use it directly in their reports.

CONTACT INFORMATION

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

Name:	David Franklin
Enterprise:	TheProgrammersCabin.com
Work Phone:	603-275-6809
Email:	dfranklin@TheProgrammersCabin.com
Web:	TheProgrammersCabin.com

Appendix A -- example of code to create a template

```
proc template;
   define style TPC_std;
   parent = Styles.Default;
   *Fonts useful are Times New Roman,Courier New,Arial;
   replace fonts /
      'TitleFont2' = ("Times New Roman", 8pt)
      'TitleFont' = ("Times New Roman", 8pt)
      'StrongFont' = ("Times New Roman", 8pt)
      'EmphasisFont' = ("Times New Roman", 8pt, Bold)
      'FixedEmphasisFont' = ("Times New Roman", 8pt, Bold)
      'FixedStrongFont' = ("Times New Roman", 8pt)
      'FixedHeadingFont' = ("Times New Roman", 8pt)
      'BatchFixedFont' = ("Times New Roman", 8pt)
      'FixedFont' = ("Times New Roman", 8pt)
      'headingEmphasisFont' = ("Times New Roman", 8pt)
      'headingFont' = ("Times New Roman", 8pt)
      'docFont' = ("Times New Roman", 8pt);
   style SystemTitle from TitlesAndFooters /
      protectspecialchars=off
      asis=on
      font=Fonts('TitleFont');
   replace Output from Container /
      frame = VOID
      rules = NONE
      background=_undef_
      frameborder=OFF;
   replace color_list /
      'bg' = cxFFFFFF
      'fg' = cx000000;
   replace colors /
      'headerfgemph' = color_list('fg')
      'headerbgemph' = color_list('bg')
      'headerfgstrong' = color_list('fg')
      'headerbgstrong' = color_list('bg')
      'headerfg' = color_list('fg')
      'headerbg' = color_list('bg')
      'datafgemph' = color_list('fg')
      'databgemph' = color_list('bg')
      'datafgstrong' = color_list('fg')
      'databgstrong' = color_list('bg')
      'datafg' = color_list('fg')
      'databg' = color_list('bg')
      'batchfg' = color_list('fg')
      'batchbg' = color_list('bg')
      'tableborder' = color_list('fg')
      'tablebg' = color_list('bg')
      'notefg' = color_list('fg')
      'notebg' = color_list('bg')
      'bylinefg' = color_list('fg')
      'bylinebg' = color_list('bg')
      'captionfg' = color_list('fg')
      'captionbg' = color_list('bg')
      'proctitlefg' = color_list('fg')
      'proctitlebg' = color_list('bg')
      'titlefg' = color_list('fg')
      'titlebg' = color_list('bg')
      'systitlefg' = color_list('fg')
      'systitlebg' = color_list('bg')
      'Conentryfg' = color_list('fg')
      'Confolderfg' = color_list('fg')
      'Contitlefg' = color_list('fg')
      'link2' = color_list('fg')
      'link1' = color_list('fg')
      'contentfg' = color_list('fg')
      'contentbg' = color_list('bg')
      'docfg' = color_list('fg')
      'docbg' = color_list('bg');
   replace Body from Document /
      bottommargin = 1.0in
      topmargin = 1.0in
      rightmargin = 1.0in
      leftmargin = 1.0in
      pagebreakhtml = html('PageBreakLine');
   replace SystemFooter from TitlesAndFooters /
      font = Fonts('TitleFont')
      just=left
      cellpadding=0
      cellspacing=0;
   style table from output /
      background=_Undef_
      frame=hsides
      rules=groups
      borderwidth = 1pt
      cellpadding = 0
      cellspacing = 0;
   style Header from header /
      background=_undef_
      frame=below
      rules=rows
      font = fonts('HeadingFont')
      foreground = colors('headerfg')
      background = colors('headerbg');
   style Rowheader from Rowheader /
      rules=rows
      background=_undef_
      frame=below;
   end;

run;
```