PharmaSUG 2016 - Paper AD13

The Little-Known DOCUMENT Procedure, a Utility for Manipulating Output Delivery System (ODS) Content

Roger D. Muller, Ph.D., Data To Events Inc.

ABSTRACT

The DOCUMENT procedure is a little-known procedure that can save you vast amounts of time and effort when managing the output of your SAS® programming efforts. This procedure is deeply associated with the mechanism by which SAS controls output in the Output Delivery System (ODS). Have you ever wished you didn't have to modify and rerun the report-generating program every time there was some tweak in the desired report? PROC DOCUMENT enables you to store one version of the report as an ODS Document Object and then call it out in many different output forms, such as PDF, HTML, listing, RTF, and so on, without rerunning the code. Have you ever wished you could extract those pages of the output that apply to certain "BY variables" such as State, StudentName, or CarModel? With PROC DOCUMENT, you have where capabilities to extract these. Do you want to customize the table of contents that assorted SAS procedures produce when you make frames for the table of contents with HTML, or use the facilities available for PDF? PROC DOCUMENT enables you to get to the inner workings of ODS and manipulate them. This paper addresses PROC DOCUMENT from the viewpoint of end results, rather than provide a complete technical review of how to do the task at hand. The emphasis is on the benefits of using the procedure, not on detailed mechanics.

INTRODUCTION

This paper is oriented towards the SAS programmer who has little or no knowledge of Proc Document. The procedure can greatly expand the use and flexibility of the Output Delivery System (ODS). Proc Document could be considered a huge utility program for working with output from SAS

ABOUT ODS

The Output Delivery System (ODS) is at the heart of the mechanism whereby SAS gathers, organizes, and shapes/converts the output to return it to the user as data processing occurs. ODS takes a core data feed and combines it with assorted templates to convert it to destination formats such things as HTML, PDF, RTF, assorted printer formats and more. At the heart of this system before output is redirected to these formats, information resides as an Output Object. This Output Object can be stored and then Proc Document can be used as the utility to further shape the information to the desired output format.

WHY WOULD YOU NOT WANT TO USE PROC DOCUMENT?

- You've never heard of it.
- You don't know what it is for or to how to use it.
- You think that the outputting of procedure output to a document such as HTML, PDF, or whatever is the "end of the line" when it comes to producing output from SAS.
- You just go ahead and create the output from SAS in a multitude of output formats, hoping that the end user will find something they like. You put all of the "by variable" and potential "where" data cuts out there hoping they will find what they need and ignore the rest.
- You have thought it was not possible via very simple SAS code to alter the output format, select only parts of a file, reorder parts of the file, change the appearance style, etc. at the time the document is being brought up for display or being printed.
- In your world, this capability for custom shaping after summarization seems like something that would be tagged on after the completion of a project. You don't work that way!
- You think that simple listings of items in a database might best be handled by some simple database reporting tool. This may well be if the output is things like simple listings, simple means, etc. But what if there are many intermediate calculations, data transformations, etc. that require the powerful tools of SAS?
- You might think "this is totally foreign concept – doing anything with a document after is produced".

- You might be scared that quality control issues will arise if the final shaping and formatting is potentially done in an end-user customer-type of environment. This may be a very legitimate concern.

WHAT ARE EXAMPLES OF THINGS PROC DOCUMENT COULD DO WITH AN ODS OUTPUT OBJECT?

- Change the output format (i.e. HTML, PDF or RTF, etc.) as the need arises.
- Pick only parts of the document to display.
- Search for some simple "where" or "by" types of variables and values and present only them in the current report, excluding the others.
- Do all of this and more by not rerunning the originating program multiple times with all sorts of multiple outputs to be maintained in the file system.
- Do all of this by running a very simple SAS program that could be initiated by a variety of means by programmers, end-users etc.
    - Do all of this from assorted environments such as:
    - Small programs created with tools such as Visual Basic, Java, etc. and implemented from a browser.
    - From SAS Environments such as Enterprise Guide, PC SAS, etc.
    - From Microsoft Excel, Word, PowerPoint via the SAS Add Ins for Microsoft Office.
    - As you can see, implementation is very flexible and is probably limited more by imagination than anything. All kinds of end-user needs and environments can readily be accommodated.

GETTING STARTED

This is only a beginning list of things that could be done and how they might be implemented after the initial programming has been completed. A very simple example will show how to create the ODS Document Object and modify it with Proc Document. See the discussion below and the Reference section at the then end of the paper for numerous other capabilities. This paper does not address all of these issues.

WHERE TO FIND INFORMATION ON PROC DOCUMENT

As of the time of the creation of this presentation, there has not been a User's Guide or Pocedures Guide published. There are two excellent books published by SAS Press:

1. Smith, Kevin S., 2014. ODS Techniques – Tips for Enhancing Your SAS Ouput. Cary, NC: SAS Institute.
2. Tuchman, Michael 2014. PROC DOCUMENT by Example Using SAS. Cary, NC: SAS Institute.

   If you are serious about using Proc Document, it is a must to buy the above books.
3. SAS ODS Users Guide may be of use when you get deeper into ODS. It does not address Proc Document directly. (See References).

In addition, the reference papers at the end of this document are very useful: (1) Zender, (2) Smith and (3) Pabbaraju.

AN EXAMPLE OF CREATING A SAS ODS OUTPUT OBJECT AND THEN MODIFYING OUTPUT WITH PROC DOCUMENT:

The code in the following table was generated to give you a working example of how Proc Document might be used to manipulate output. IMPORTANT: This example generates a large amount of output that is too much to show in this paper, although brief snapshots will be placed in later figures. It is highly suggested that you copy the SAS code into your interactive PC SAS environment or your SAS Enterprise Guide environment to run it and see the true output

and become familiar with interacting with Proc Document. You will have to modify the libname for permanent storage if you plan on using the two-level names to store the ODS Document Object. Alternatively, you can use a one level name if you want the object to not be retained beyond the current session. You can also run this code in batch mode if that is the environment in which you work.

This code was run in SAS 9.4 M1R2 remotely via VMWARE on a SAS/Grid Linux Server. Enterprise Guide 7.11 was on the local PC. Most, if not all, of this should run on much earlier versions of SAS. EG is not required. Everything required to do this example code is part of base SAS.

Code	What's Happening
*ODS Proc Document Use; *Setup a libname for permanent storage of Output Object; libname keepit '/[redacted]/rmuller' ; run;	Setting up a permanent libname so that a 2-level document name can be used in SAS. Two-level names are permanent and can be retrieved at a later date and time. Some of the folder names are redacted.

*send output to document object file only,do not create final print file;
 	
ods document name=keepit.try1;
 	
proc print data=sashelp.cars;
 	
run;
 	
ods document close;
	A simple proc print object document will be written to keepit.try1. Note that no html file or pdf file or print file is created.

*document object is now created. Let's look at in a listing;
 	
ods listing;
 	
proc document name=keepit.try1;
 	
list /levels=all;
 	
run;
 	
ods listing close;
	No printout appeared. Let's look at what is in keepit.try1. In order to display it on screen, we will create a listing file, run code and close the listing.
	
	Output is shown in Figure 1.

*greatly expand contents of new document;
 	
ods document name=keepit.try2;
 	
proc sort data=sashelp.cars out=work.cars;by origin type;run;
 	
proc print data=work.cars;by origin type;run;
 	
proc means data=work.cars;var msrp;by origin type;run;
 	
run;
 	
ods document close;
	Now we are making a much bigger document file. First, sort the data by origin and type. Then do a complete proc print of all variables (by origin and type) and a proc means on the MSRP variable (by origin and type).
	
	Again the only output created is an object file -- keepit.try2. The resulting file would be visible in systems directory viewing tools such as Windows Explorer as try2.sas7bitm in the directory that keepit is mapped.

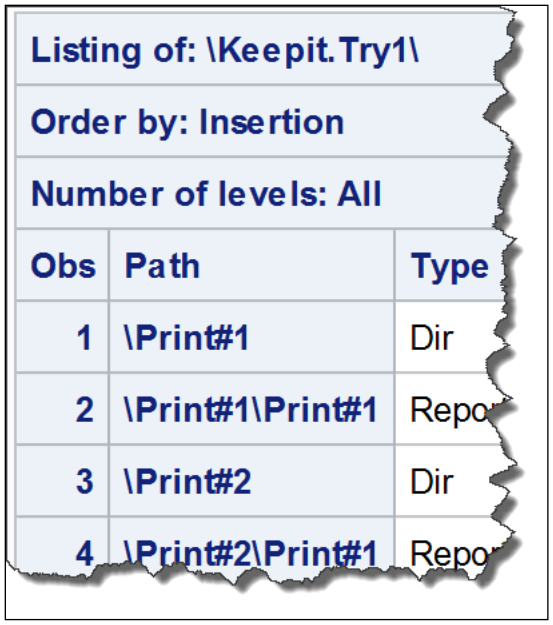
``*look at what is in the object file;
ods listing;
proc document name=keepit.try2;
list /levels=all;
run;
ods listing close;```
	Look at the contents of the document file keepit.try2. There are many more levels because of multiple procs and the by variables.
	
	Partial output is shown in Figures 2 and 3. Do not worry about fully understanding what is here. Its functionality will emerge as you work further thru this exercise. Too much studying of this content at this time can be counterproductive.

``*convert to output formats and demonstrate selection
techniques;
ods html;*just creating temporary files;
proc document name=keepit.try2;
replay ^(where=(origin='Asia' AND type='SUV'));
run;
quit;
ods html close;```
	Let's create an html document that an end-user wants out of the content of the ODS Document Object stored as keepit.try2.
	
	Proc Document will convert to HTML and replay only where origin='Asia' and type= 'SUV'. The rest of information is not printed. Note that the data was originally sorted by origin and type.
	
	Information from Proc Print is in Figure 4 and from Proc Means in Figure 5.

``proc document name=keepit.try2;
replay ^(where=(origin='Europe' and type='SUV'));
run;
quit;
ods html close;```
	Here is a "where" pulling out specific origin and type values with the same order as the original sort.
	
	This works fine, only those values will show for both the proc print and the proc means.
	
	Output is not shown in the interest of brevity.

``ods html;
proc document name=keepit.try2;
replay ^(where=(type='SUV' and origin='Europe'));
run;
quit;
ods html close;```
	Here is a "where" pulling out specific origin and type with a different order than the original sort.
	
	This works fine, only those values specified will show for both proc print and proc means. Output is not shown.
	
	That concludes the code demonstration. Run it yourself to better understand how Proc Document works.

The following figures show a few selected printouts of the above coding exercise:



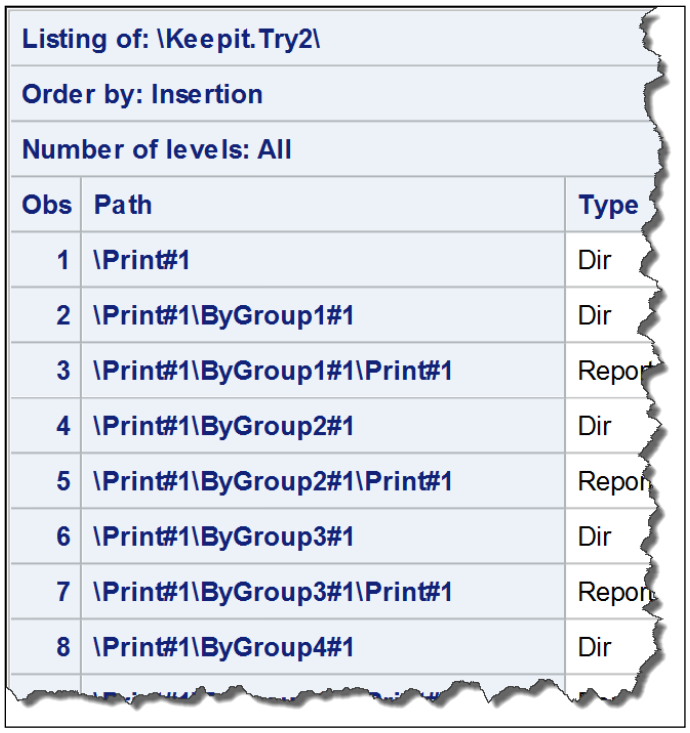
Listing of: \Keepit.Try1\

Order by: Insertion

Number of levels: All

Obs	Path	Type
1	\Print#1	Dir
2	\Print#1\Print#1	Repo
3	\Print#2	Dir
4	\Print#2\Print#1	Repo

Figure 1. A Listing of the Levels in an ODS Document Object.



Listing of: \Keepit.Try2\

Order by: Insertion

Number of levels: All

Obs	Path	Type
1	\Print#1	Dir
2	\Print#1\ByGroup1#1	Dir
3	\Print#1\ByGroup1#1\Print#1	Repor
4	\Print#1\ByGroup2#1	Dir
5	\Print#1\ByGroup2#1\Print#1	Repor
6	\Print#1\ByGroup3#1	Dir
7	\Print#1\ByGroup3#1\Print#1	Repor
8	\Print#1\ByGroup4#1	Dir

Figure 2. Partial Printout of Level Information. Note "Print" and "ByGroups" are Present

61	\Print#2\ByGroup15#1	Dir
62	\Print#2\ByGroup15#1\Print#1	Repor
63	\Means#1	Dir
64	\Means#1\ByGroup1#1	Dir
65	\Means#1\ByGroup1#1\Summary#1	Table
66	\Means#1\ByGroup2#1	Dir
67	\Means#1\ByGroup2#1\Summary#1	Table
68	\Means#1\ByGroup3#1	Dir
69	\Means#1\ByGroup3#1\Summary#1	Table
70	\Means#1\ByGroup4#1	Dir
71	\Means#1\ByGroup4#1\Summary#1	Table

Figure 3. A Mid-Stream Printout of Level Information, Continued from Above.

Note: the End of the Print Section and the Beginning of Means Section and All of the "By" Information. Needless to Say, Lots of Levels!

ORIGIN=Asia TYPE=SUV

Obs	MAKE	MODEL	DRIVETRAIN	MSRP	INVOICE	ENGINESIZE	CYLINDERS
4	Acura	MDX	All	$36,945	$33,337	3.5	6
5	Honda	Pilot LX	All	$27,560	$24,843	3.5	6
6	Honda	CR-V LX	All	$19,860	$18,419	2.4	4
7	Honda	Element LX	All	$18,690	$17,334	2.4	4
8	Hyundai	Santa Fe GLS	Front	$21,589	$20,201	2.7	6
9	Isuzu	Ascender	All	$31,849	$29,977	4.2	6

Figure 4. Partial Output from Proc Print for Asian SUV's. All Other Data in the Original Document is Not Included in Printout Because of the Coding in Proc Document.

ORIGIN=Asia TYPE=SUV

Analysis Variable : MSRP

N	Mean	Std Dev	Minimum	Maximum
25	29569.00	11842.55	17163.00	64800.00

Figure 5. Means for Asian SUV's. All Other Means in the Original Document Are Not Presented Because of Proc Document Coding.

DISCUSSION OF OTHER CAPABILITIES

Here's a "big one" we haven't discussed. With minimal input the Style associated with the creation of a report can be changed. By default, HTML documents are created with a style call "HTMLblue". This was used in the figures shown earlier. If you wish to change to some other color, just issue an ODS HTML statement with the style you want. Example: ods html style=Harvest file='/biometrics/perftest/rmuller/whatever.html';. There are currently 54 styles furnished with base SAS. To see a listing of their names, submit the following code:

```
proc template;
 list styles;
 run;
```

The best way to find out what is in them and what they affect is to run test code with a variety of data and graphics output.

Proc Document has many other capabilities above and beyond those show here. Zinder referenced 22 other capabilities and made comment that there are even more http://support.sas.com/resources/papers/sgf09/318-2009.pdf.

CONCLUSION

Proc Document adds exciting capabilities to further modify the output of SAS operations. With minimal coding requirements, minimal computing power and minimal interaction by users, customized output can readily be generated further enhancing the original report. Yes, you should learn Proc Document!

REFERENCES

SAS Institute, 2013. *SAS 9.4 Output Delivery System User's Guide, Third Edition*. 1129 pages. Published by SAS Institute, Cary NC

Smith, Kevin, 2012. ODS DOCUMENT From Scratch. http://support.sas.com/resources/papers/proceedings12/273-2012.pdf

Zinder, Cynthia L. Rearrange and Replay Your Output with ODS DOCUMENT. http://support.sas.com/resources/papers/sgf09/318-2009.pdf

Pabbaraju, Jyothi, 2010. See All, Know All: Using PROC DOCUMENT to Produce Integrated Data Set Documentation. http://www.lexjansen.com/nesug/nesug10/po/po23.pdf

Smith, Kevin S., 2014. *ODS Techniques – Tips for Enhancing Your SAS Ouput*. Cary, NC: SAS Institute.

Tuchman, Michael 2014. *PROC DOCUMENT by Example Using SAS*. Cary, NC: SAS Institute.

CONTACT INFORMATION

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

Roger D. Muller, Ph.D.
Data-To-Events, Inc.
14475 Stephanie St.
Carmel, IN USA
317/985-0132(cell) 317/846-5782
rdmuller@hotmail.com
www.data-to-events.com, www.rogermullervideography.com