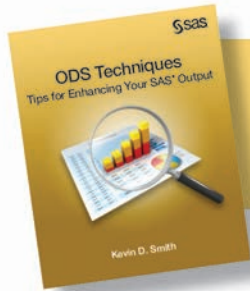


ODS Techniques

Tips for Enhancing Your SAS® Output



Kevin D. Smith



From *ODS Techniques*. Full book available for purchase [here](#).

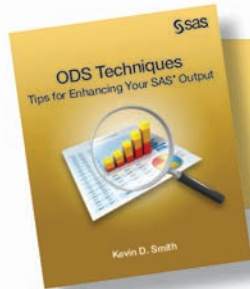
Contents

About This Book	ix
About The Author	xiii
Acknowledgements	xv
Chapter 1: Introduction	1
Chapter 2: General Tips.....	3
Controlling Your ODS Output.....	3
The ODS Sandwich.....	4
Always Use ODS CLOSE.....	6
Closing All ODS Destinations at Once.....	6
Interleaving ODS Destinations	7
Running Multiple Instances of One Destination	8
Displaying Output Object Information.....	11
Selecting and Excluding Output Objects	12
Suspending an ODS Destination.....	14
Querying Open ODS Destination Information	16
Creating Output Data Sets	17
Using ODS Inline Functions and Formatting	18
Inserting Special Characters	20
Generating Output Using Internet Services	21
Emailing Output	21
Sending Output to an FTP Server	23
The ODS Path Macro Variable	24
Setting the Default Path for Output Files.....	26
Exporting ODS Output as “Character” Separated Values	27
Chapter 3: Table and Text Tips.....	29
Procedure-specific Table Features	29

Creating One-Shot Tables	29
Controlling Blank Lines in PROC PRINT	31
Creating Spanned Rows in PROC REPORT	33
Preventing Cell Merging in PROC TABULATE	34
Creating Tables with DATA _NULL_	35
Styling All Template-Based Tables at Once.....	37
Ad Hoc Tables with the Report Writing Interface.....	38
Generating Bulleted Lists and Blocks of Text	40
Creating Bulleted Lists.....	40
Creating Nested Lists.....	41
Creating Blocks of Text	43
Chapter 4: Style Tips	45
Getting Started with the Basics	45
Generating Samples of All Styles.....	46
Creating a Simple Style	48
Creating a Simple CSS Style	49
Determining the Correct Style Element to Use	51
Dealing with Borders and Padding.....	52
Using CELLPADDING Versus PADDING	52
What Are BORDERSPACING and BORDERCOLLAPSE?.....	54
Controlling Borders Individually.....	54
Alternating Table Row Colors	56
Using RGBA Colors	58
Advanced Traffic Lighting Functions	59
Using CSS	61
Referencing CSS Styles	62
Importing CSS from CSS	62
Style Selection Based on Media Types in CSS.....	63
CSS Color Models.....	64
Displaying ODS's Document Object Model	65
Setting a Global Font for the Printer Destinations in CSS.....	66
Using IDs in CSS	67
BY Group Styling in CSS	69
Chapter 5: HTML Tips.....	73

HTML3 vs HTML4 vs HTML5.....	74
ODS HTML 3.....	74
ODS HTML 4.....	74
ODS HTML 5.....	74
Cascading Style Sheet Tips	75
Generating an External CSS File	75
Specifying Your Own CSS File	76
Specifying Multiple CSS Files.....	78
Setting the Document Title	78
Creating Document Fragments	80
Inserting Arbitrary Markup in the Document Preamble	80
Using External Resources	82
Utilizing Web Fonts.....	82
Linking to Your Own Custom JavaScript.....	84
Change the Image Format Used by Graphs	85
Background Effects	86
Using a Background Image on a Table	87
Using Gradients Backgrounds	88
Preserving White Space	89
ODS HTML5 Techniques	91
Turning Off Inline SVGs	91
Inlining Bitmapped Images	92
Inline All Style Attributes	92
Chapter 6: Printer Tips	95
Multicolumn Output without Layout.....	95
Page Breaks and Orientation.....	96
Changing Page Orientation	96
Controlling Page Breaks	98
Styles in ODS PDF	100
Using CSS-based Styles	100
Installing Custom Fonts	103
Adding Background Images and Watermark Effects	103
Creating Bitmaps of Tables.....	105
Adding Metadata to Your PDF Documents	106
Creating Secured PDF Documents	108

Specifying the Initial PDF Page View	109
Chapter 7: Excel Tips	111
Comparing ODS Excel Formats	111
TAGSETS.EXCELXP	112
TAGSETS.MSOFFICE2K	112
HTML	112
TAGSETS.CSV	112
Displaying Help	112
Using Excel Formats	113
Preserving Leading Zeros	115
Alternate Solution	116
TAGSETS.EXCELXP Tips	117
Using Excel Formulas in TAGSETS.EXCELXP	117
Controlling Worksheet Names Manually	119
Using BY Group Information in Worksheet Names	120
Using Graphical Output in Excel	122
Preventing Cell Merging in PROC TABULATE	123
More Excel Tagsets Online	123
Chapter 8: ODS Document Tips	125
Customizing Your Report Structure	125
Creating a Completely Custom Table of Contents	125
Recording Actions in the Documents Window	128
Using Links to Create Custom Tables of Contents	128
Using Links to Create Aggregate Reports	130
Using WHERE Clauses in Documents	131
Templates and Dynamic Variables	133
Display the Template Used by an Object in a Document	133
Displaying Dynamic Variables Used by an Output Object	134
Swapping Templates during Replay	135
Accessing the Data in Document Output Objects Directly	137
Inserting Plain Text Objects into a Document	138
Operating on a Document Using CALL EXECUTE	140
Index	143



From *ODS Techniques*. Full book available for purchase [here](#).

Chapter 2: General Tips

Controlling Your ODS Output.....	3
The ODS Sandwich	4
Always Use ODS CLOSE	6
Closing all ODS Destinations at Once	6
Interleaving ODS Destinations	7
Running Multiple Instances of One Destination.....	8
Displaying Output Object Information.....	11
Selecting and Excluding Output Objects	12
Suspending an ODS Destination.....	14
Querying Open ODS Destination Information	16
Creating Output Data Sets	17
Using ODS Inline Functions and Formatting.....	18
Inserting Special Characters	20
Generating Output Using Internet Services.....	21
Emailing Output.....	21
Sending Output to an FTP Server	23
The ODS Path Macro Variable	24
Setting the Default Path for Output Files	26
Exporting ODS Output as “Character” Separated Values	27

This section includes tips that span all of the Output Delivery System (ODS) world. It includes statements and options that work across all ODS destinations or affect ODS globally.

Controlling Your ODS Output

ODS offers many options and techniques to control your output. This section describes various ways of controlling ODS destinations using some common and uncommon techniques.

The ODS Sandwich

The *ODS sandwich* is the term used to describe the statements used to open and close an ODS destination. The ODS sandwich is the most fundamental concept in ODS and if you've used ODS at all before, you are likely already familiar with this technique.

The technique is fairly simple. You use one statement to open an ODS destination, then another statement to close the destination. Opening a destination causes all SAS output from that point on to be rendered to that destination. Closing it causes the destination to be closed and prevents any more SAS output from going to the destination.

If you've used SAS at all since Version 8, you've already been using at least part of this technique even if you haven't realized it. Up until SAS 9.3, the listing destination was automatically opened by SAS when you started a SAS session. This is essentially the same as invoking the following command:

```
ODS LISTING;
```

This statement opens the listing destination, and any procedure or DATA step output from this point on will be sent to the listing destination. To close the destination and complete the ODS sandwich, you would use the ODS CLOSE command as follows:

```
ODS LISTING CLOSE;
```

In SAS 9.3 (DMS), the default destination was changed to HTML. The ODS sandwich for HTML would look like the following:

```
ODS HTML;
```

```
... procedure and/or DATA step code ...
```

```
ODS HTML CLOSE;
```

This same technique works for all ODS destinations (for example, PDF, RTF, Tagsets.ExcelXP, PowerPoint, etc.). In fact, there is nothing preventing you from generating multiple outputs at one time. We will finish this tip with a code sample that demonstrates the ODS sandwich technique using multiple destinations.

```
ODS HTML;
```

```
ODS PDF;
```

```
ODS RTF;
```

```
PROC PRINT DATA=sashelp.class(OBS=5);
```

```
RUN;
```

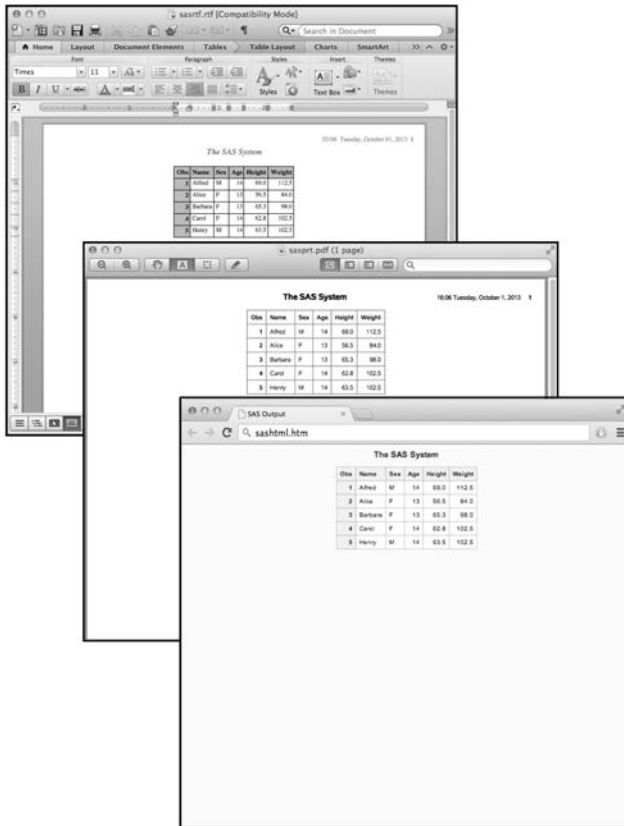
```
ODS RTF CLOSE;
```

```
ODS PDF CLOSE;
```

```
ODS HTML CLOSE;
```

Figure 2.1 shows the output from the code above.

Figure 2.1: Output generated by multiple destinations simultaneously



Always Use ODS CLOSE

A common pitfall to the ODS sandwich is forgetting close the destination. You may have noticed in your use of ODS that, in some destinations, after you open the ODS destination and run some procedures that the output file has content in it and you can open the file even without closing the ODS destination. Don't assume that because this works from time to time that you don't need the ODS CLOSE statement.

If you don't explicitly close each destination, you can end up with a partial output file which may result in a file that won't open, or it won't contain all of the output that you expected. So just remember to always close all of your ODS destinations.

Closing All ODS Destinations at Once

Using the ODS CLOSE statement can become tedious if you are changing destinations a lot in your SAS programs, so there is a shortcut that closes all ODS destinations (including listing) at once. The statement that does this is shown below.

```
ODS _ALL_ CLOSE;
```

This is used in place of closing each destination independently. For example, you may want to avoid using the following code to open and close a number of ODS destinations:

```
ODS HTML;
ODS PDF;
ODS RTF;

PROC PRINT DATA=sashelp.class(OBS=5);
RUN;

ODS RTF CLOSE;
ODS PDF CLOSE;
ODS HTML CLOSE;
```

You could simply do this.

```
ODS HTML;
ODS PDF;
ODS RTF;

PROC PRINT DATA=sashelp.class(OBS=5);
RUN;

ODS _ALL_ CLOSE;
```

Keep in mind that this will close any destinations that are open by default in SAS as well. So if the listing destination was opened by SAS at the beginning of the session, it will be closed as well. To make sure that the listing destination stays open, this is a common idiom.

```
ODS _ALL_ CLOSE;  
ODS LISTING;
```

Interleaving ODS Destinations

All of the examples from our previous tips have assumed that all ODS destinations begin and end at the same time. It doesn't have to be this way though. You can open and close each ODS destination independently of each other. This is most easily shown with an example.

Let's say that we have a report that includes the output from three procedures. We want the whole report to be rendered to HTML, but we need only the second procedure output in PDF. To do this, we simply put our ODS HTML sandwich around the entire report, and our ODS PDF sandwich around only the second procedure as follows.

```
ODS HTML;  
  
PROC CONTENTS DATA=sashelp.class;  
RUN;  
  
ODS PDF;  
  
PROC PRINT DATA=sashelp.class(OBS=3);  
RUN;  
  
ODS PDF CLOSE;  
  
PROC MEANS DATA=sashelp.class;  
  VAR age;  
RUN;  
  
ODS HTML CLOSE;
```

Figure 2.2 and Figure 2.3 show the HTML and PDF output that results, respectively. Notice that the HTML output has more tables than the PDF output because ODS PDF was used only for the PROC PRINT output.

8 ODS Techniques: Tips for Enhancing Your SAS Output

Figure 2.2: HTML output

The screenshot displays the SAS HTML output for the CONSENT procedure. It includes a metadata table, a table of dependent information, an alphabetical list of variables, a data table, and a summary table.

Item	Name	Sex	Age	Height	Weight
1	Adam	M	14	88.3	112.9
2	Alice	F	13	96.3	84.3
3	Barbara	F	13	86.3	89.3

Item	Name	Sex	Age	Height	Weight
1	Adam	M	14	88.3	112.9
2	Alice	F	13	96.3	84.3
3	Barbara	F	13	86.3	89.3

Figure 2.3: PDF output

The screenshot displays the SAS PDF output for the CONSENT procedure. It shows the same data table as Figure 2.2, but in a compact format suitable for a PDF document.

Item	Name	Sex	Age	Height	Weight
1	Adam	M	14	88.3	112.9
2	Alice	F	13	96.3	84.3
3	Barbara	F	13	86.3	89.3

Running Multiple Instances of One Destination

In addition to being able to run multiple ODS destinations simultaneously, you can also run multiple instances of a single ODS destination. This can be useful if you want to create different subsets of a report in one format, or if you want to create reports in the same format but different styles.

The way to do so is by giving each instance of the destination a unique identifier. This identifier goes in parentheses after the destination name. Below is an example of creating two HTML reports simultaneously, but using different styles as well as including only part of the report in the second HTML file.

```
ODS HTML(1) FILE='one.html';

PROC CONTENTS DATA=sashelp.class;
RUN;

ODS HTML(2) STYLE=styles.journal FILE='two.html';

PROC PRINT DATA=sashelp.class(OBS=3);
RUN;

ODS HTML(1) CLOSE;

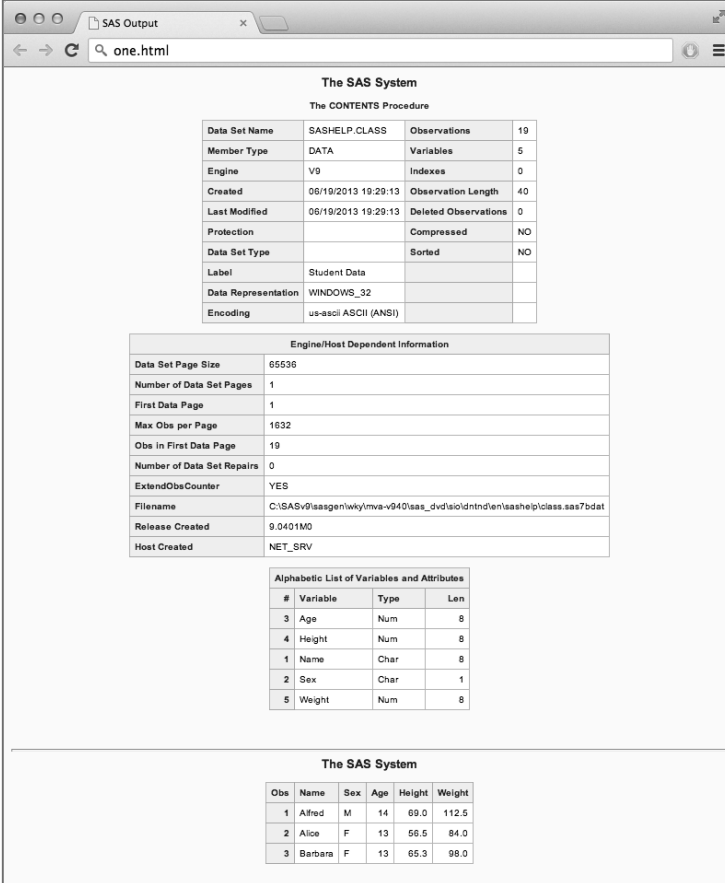
PROC MEANS DATA=sashelp.class;
  VAR age;
RUN;

ODS HTML(2) CLOSE;
```

10 ODS Techniques: Tips for Enhancing Your SAS Output

Figures 2.4 and 2.5 show the output from the two HTML reports.

Figure 2.4: Output from ODS HTML(1)



The screenshot shows a web browser window titled 'SAS Output' with the address bar containing 'one.html'. The main content area displays the following information:

The SAS System
The CONTENTS Procedure

Data Set Name	SASHELP.CLASS	Observations	19
Member Type	DATA	Variables	5
Engine	V9	Indexes	0
Created	06/19/2013 19:29:13	Observation Length	40
Last Modified	06/19/2013 19:29:13	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label	Student Data		
Data Representation	WINDOWS_32		
Encoding	us-ascii ASCII (ANSI)		

Engine/Host Dependent Information

Data Set Page Size	65536
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	1632
Obs in First Data Page	19
Number of Data Set Repairs	0
ExtendObsCounter	YES
Filename	C:\SASV9\asgen\wky\mva-v940\ass_dvd\oldntnd\en\sasheip\class.sas7bdat
Release Created	9.0401M0
Host Created	NET_SRV

Alphabetic List of Variables and Attributes

#	Variable	Type	Len
3	Age	Num	8
4	Height	Num	8
1	Name	Char	8
2	Sex	Char	1
5	Weight	Num	8

The SAS System

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0

Figure 2.5: Output from ODS HTML(2)

The SAS System					
Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0

The SAS System					
The MEANS Procedure					
Analysis Variable : Age					
N	Mean	Std Dev	Minimum	Maximum	
19	13.3157895	1.4926722	11.0000000	16.0000000	

Displaying Output Object Information

Every object that comes out of ODS has information associated with it by the procedure that created it, such as the output object name and label, the template name, and the object path. These pieces of information can be used in conjunction with other ODS statements and procedures to customize your reports.

To turn output object tracing on, you simply run the following command.

```
ODS TRACE ON;
```

You can also turn on the label path using the LABEL option as follows.

```
ODS TRACE ON / LABEL;
```

The code below demonstrates how to trace the output object information.

```
ODS TRACE ON / LABEL;
PROC MEANS DATA=sashelp.class;
  VAR age;
RUN;
```

The listing and log from this program code appear as Output 2.1.

Output 2.1: Output Object Trace Information

```
Output Added:
-----
Name:         Summary
Label:        Summary statistics
Template:     base.summary
```

12 ODS Techniques: Tips for Enhancing Your SAS Output

```
Path:          Means.Summary
Label Path:    'The Means Procedure'. 'Summary statistics'
-----
```

Selecting and Excluding Output Objects

There may be times when you want to exclude some of the outputs from a procedure, or conversely, display only some outputs. This is possible with the use of the ODS SELECT and ODS EXCLUDE statements.

This method is a two-step process. First you turn on output object tracing, using ODS TRACE ON; to get the name of the output object that you want to select or exclude, then you use the ODS SELECT/EXCLUDE statement to enable or disable it.

We saw an example of using ODS TRACE ON in the tip on “Displaying Output Object Information”. We will use it with PROC CONTENTS here in the following code:

```
/* First pass to get output object names */

ODS TRACE ON;

PROC CONTENTS DATA=sashelp.class;
RUN;

/* Select just the Variables output object */

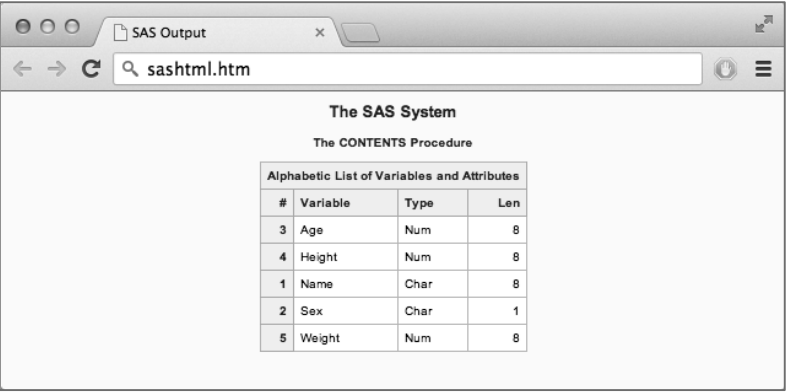
ODS SELECT Variables;

ODS HTML;
PROC CONTENTS DATA=sashelp.class;

RUN;
ODS HTML CLOSE;
```

Figure 2.6 shows the output. As you see, we receive only one output object this time.

Figure 2.6: Demonstration of ODS SELECT statement



The SAS System

The CONTENTS Procedure

Alphabetic List of Variables and Attributes

#	Variable	Type	Len
3	Age	Num	8
4	Height	Num	8
1	Name	Char	8
2	Sex	Char	1
5	Weight	Num	8

Now let's try excluding just the Variables object.

ODS EXCLUDE Variables;

```
ODS HTML;
PROC CONTENTS DATA=sashelp.class;
RUN;
ODS HTML CLOSE;
```

Figure 2.7 shows the output from the code above.

Figure 2.7: Demonstration of the ODS EXCLUDE statement



The SAS System

The CONTENTS Procedure

Data Set Name	SASHHELP.CLASS	Observations	19
Member Type	DATA	Variables	5
Engine	V9	Indexes	0
Created	06/19/2013 19:29:13	Observation Length	42
Last Modified	06/19/2013 19:29:13	Sorted Observations	0
Production		Compressed	NO
Data Set Type		Sorted	NO
Label	Student Data		
Data Representation	WINDOWS_32		
Encoding	win-ascii ASCII (ANSI)		

Engine/Host Dependent Information

Data Set Page Size	65536
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	1024
Obs in First Data Page	19
Number of Data Set Repairs	0
ExtentOfCounter	YES
Filename	C:\SAS\Program\utils\msvcr90\data_set\sorted\SASHHELP\CLASS.SAS7BDAT
Release Created	0.241190
Host Created	NET_SVR

You are not limited to just one name on the SELECT/EXCLUDE statements. You can specify as many output objects as you wish. The list will, however, be reset after each procedure unless you use the PERSIST option.

14 ODS Techniques: Tips for Enhancing Your SAS Output

Even though both the SELECT statement and the EXCLUDE statement exist, you should never try to use both of them at the same time. That behavior is undefined.

Suspending an ODS Destination

Although there isn't exactly a "pause" button in ODS, you can effectively cause that behavior by using ODS SELECT and ODS EXCLUDE with the ALL and NONE keywords. You can turn off all of the output objects in ODS using the following command:

```
ODS SELECT NONE;
```

Or equivalently:

```
ODS EXCLUDE ALL;
```

If either of these is executed, you will no longer see any output generated by ODS destinations. To enable output again, you would use:

```
ODS SELECT ALL;
```

Or alternatively:

```
ODS EXCLUDE NONE;
```

The statements above are global to ODS and would suspend/enable all ODS destinations. You can also suspend/enable individual ODS destinations by including the destination name after "ODS". For example, to suspend all output to PDF, but leave all other destinations untouched, you would use:

```
ODS PDF EXCLUDE ALL;
```

To enable output to PDF again, you would use EXCLUDE NONE as follows:

```
ODS PDF EXCLUDE NONE;
```

Here is a complete example that begins with PDF and HTML being generated. We then suspend PDF output using ODS PDF EXCLUDE ALL; and enable it again with ODS PDF EXCLUDE NONE; later on. All the while, HTML gets all of the output.

```
ODS HTML;  
ODS PDF STARTPAGE=NO;  
  
PROC PRINT DATA=sashelp.class(obs=3);  
RUN;  
  
ODS PDF EXCLUDE ALL;  
  
PROC MEANS DATA=sashelp.class;
```

```

VAR age;
RUN;

PROC MEANS DATA=sashelp.class;
  VAR height weight;
RUN;

ODS PDF EXCLUDE NONE;

PROC REPORT DATA=sashelp.class(obs=3);
RUN;

ODS HTML CLOSE;
ODS PDF CLOSE;

```

Figures 2.8 and 2.9 show the output from ODS HTML and ODS PDF generated by the code above.

Figure 2.8: ODS HTML output showing all tables

The SAS Output window displays the following HTML output for `sashtml.htm`:

The SAS System

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0

The SAS System
The MEANS Procedure

Analysis Variable : Age

N	Mean	Std Dev	Minimum	Maximum
19	13.3157895	1.4926722	11.0000000	16.0000000

The SAS System
The MEANS Procedure

Variable	N	Mean	Std Dev	Minimum	Maximum
Height	19	62.3368421	5.1270752	51.3000000	72.0000000
Weight	19	100.0263158	22.7739335	50.5000000	150.0000000

The SAS System

Name	Sex	Age	Height	Weight
Alfred	M	14	69	112.5
Alice	F	13	56.5	84
Barbara	F	13	65.3	98

Figure 2.9: ODS PDF showing only tables not excluded

The SAS System 16:42 Tuesday, October 1, 2013 1

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0

Name	Sex	Age	Height	Weight
Alfred	M	14	69	112.5
Alice	F	13	56.5	84
Barbara	F	13	65.3	98

Querying Open ODS Destination Information

There may be times when you want to query ODS to see which destinations are currently open. There is a built-in data set view that allows you to do so. It is called SASHELP.VDEST. This view can be used as an input data set for the DATA step or procedures. Here is an example that simply prints out the open destination information. We will open a couple of destinations first so that there will be something to report.

```
/* Close everything out and start fresh */
ODS _ALL_ CLOSE;

/* Open a few destinations */
ODS LISTING;
ODS PDF;
ODS RTF;
ODS HTML;

/* Print the contents of SASHELP.VDEST */
PROC PRINT DATA=sashelp.vdest;
RUN;

/* Close all destinations */
ODS _ALL_ CLOSE;
```

Output 2.2 shows the output from PROC PRINT.

Output 2.2: PROC PRINT rendering of SASHELP.VDEST

Obs	destination	style
1	LISTING	Listing
2	PDF	Pearl

3	RTF	Rtf
4	HTML	HTMLBlue

Creating Output Data Sets

You can capture the output data from nearly every procedure by using the ODS output destination. This destination is a bit different than other destinations in that it doesn't create files that you view, it creates a SAS data set. The syntax for opening the ODS output destination is as follows:

```
ODS OUTPUT output-object-name=data-set-spec;
```

where *output-object-name* is the name of the output object, and *data-set-spec* is the output data set name and options.

The name of the output object can be acquired using ODS TRACE (see the “Displaying Output Object Information” tip for more information). Basically, you turn on ODS tracing to get the output object name and run your SAS program. The output object name will show up in the tracing information.

The data set name is arbitrary. You can specify whatever data set name you wish. You can also specify data set options in parentheses as usual.

Here is a simple example using PROC CONTENTS. The example here turns on ODS tracing to show the output object name, then runs PROC CONTENTS again to capture the output data set. In this example, we are only keeping the Variable and Type columns by using KEEP= in the output data set options.

```
/* Get the output object name */
ODS TRACE ON;
PROC CONTENTS DATA=sashelp.class;
RUN;
ODS TRACE OFF;

/* Open the Output destination and
   run the procedure again to capture the data set */
ODS OUTPUT Variables=MyDataSet(KEEP=Variable Type);
PROC CONTENTS DATA=sashelp.class;
RUN;

/* Print out the output data set */
ODS HTML;
PROC PRINT DATA=MyDataSet;

RUN;
ODS HTML CLOSE;
```

Figure 2.10 shows the output from the above code.

Figure 2.10: PROC PRINT rendering of ODS output data set

The screenshot shows a web browser window with the title 'SAS Output' and the address bar containing 'sashtml.htm'. The main content area displays a table titled 'The SAS System' with the following data:

Obs	Variable	Type
1	Age	Num
2	Height	Num
3	Name	Char
4	Sex	Char
5	Weight	Num

Using ODS Inline Functions and Formatting

For most of ODS' existence, applying styles could only be done down to a table cell level. If you wanted to put formatting within a cell, you had to put destination-specific strings into your data, which would then be embedded in your output file. The problem with this is that different data had to be created for each destination. In addition, some destinations like PDF are binary files, which are nearly impossible to embed information into from a data set. To alleviate this situation ODS allows you to embed special functions and formatting calls in a destination-agnostic way so that they can be supported across destinations.

The general syntax for inline functions is:

```
ods-escape-char{function arguments}
```

where *ods-escape-char* is a user-specified character that indicates the beginning of an inline function, *function* is the name of the function, and *arguments* are the function arguments.

The ODS escape character is simply a single character that you specify to indicate the start of an inline function call. It can be any character, but you should choose a character that doesn't exist in your data. To tell ODS what character you will use, you use the ODS ESCAPECHAR statement. Here is an example of the ODS ESCAPECHAR statement that sets the escape character to caret (^).

```
ODS ESCAPECHAR '^';
```

The function names are specified by ODS and include things such as dagger (dagger character), sigma (sigma character), Unicode (Unicode character), super (superscript), sub (subscript), raw (destination specific formatting), and style (font and color styling). The most common inline function is style. We will look at that one here. Other inline functions for inserting special characters and accenting characters are covered in the following tips.

To add inline styling to your reports, you simply use the general syntax shown above with ‘style’ as the function name. The arguments to the style function are the style override and the text to style. The style override is the same format as STYLE= in ODS templates and Base reporting procedures. You can specify a style element name, style overrides, or both. Here is an example of some titles with inline formatting added.

```

/* Set the ODS escape character */
ODS ESCAPECHAR '^';

ODS PDF;
ODS HTML;

/* Add inline function syntax to text */
TITLE 'This is the ^{style dataemphasis First} title';
TITLE2 'This is the ^{style [color=red] Second} title';
TITLE3 'This is the ^{style datafixed[color=blue] Third}
      title';

PROC PRINT DATA=sashelp.class(obs=3);
RUN;

ODS HTML CLOSE;
ODS PDF CLOSE;

```

Figures 2.11 and 2.12 show the output in ODS HTML and ODS PDF, respectively.

Figure 2.11: ODS HTML output demonstrating inline formatting

This is the **First** title
This is the Second title
This is the Third title

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0

Figure 2.12: ODS PDF output demonstrating inline formatting

The screenshot shows a PDF viewer window titled 'sasprt.pdf (1 page)'. The content of the PDF is as follows:

This is the First title
 This is the Second title
 This is the Third title

16:57 Tuesday, October 1, 2013 1

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0

You are not limited to putting inline functions in titles. All titles and data elements go through the same processing, so you can add similar formatting inside your data cells by using the inline function syntax in your data values.

Inserting Special Characters

Inserting special characters into your reports uses the inline function syntax described in the previous tip. There are several built-in function names for special characters that are more common. In addition, there is a unicode function to insert arbitrary unicode characters. The most commonly used special characters are Greek characters. They are available simply as the name of the character spelled out (e.g., ALPHA, BETA, GAMMA, DELTA, etc.). For uppercase characters, you add “_U” as a suffix to the name (i.e., ALPHA_U, BETA_U, GAMMA_U, DELTA_U, etc.). For example, to print an Alpha character, you would enter the following in your title or data string.

```
^{\unicode alpha}
```

This is assuming that your ODS escape character is set to ‘^’.

For arbitrary Unicode characters, you can specify the Unicode decimal value in the unicode function. The decimal values can be obtained many places on the Internet such as <http://www.unicode-table.com/>. Let’s say we wanted to print a check mark. The Unicode value for that character is 2714 (there are more Unicode check marks, but this one will work for our example). We then use the unicode function as follows to print a check mark.

```
^{\unicode 2714}
```

Some other common codes that you may be interested in are greater-than or equal to (2265), less-than or equal to (2264), and not equal to (2260). Let’s put a check mark character into some SAS code so we can run it and see the results.

```
/* Set the ODS escape character */
ODS ESCAPECHAR '^';
```

```

/* Create a title with a check mark character in it */
TITLE '^{\unicode 2714} Class Information';

/* Generate some output */
ODS PDF;

PROC PRINT DATA=sashelp.class(obs=3);
RUN;

ODS PDF CLOSE;

```

Figure 2.13 shows the output.

Figure 2.13: ODS PDF including a check mark character in the title

The screenshot shows a PDF viewer window titled 'sasprt.pdf (1 page)'. The main content area displays the following information:

✓ Class Information 17:01 Tuesday, October 1, 2013 1

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0

Generating Output Using Internet Services

Generating output to a file is the most common way of using ODS, but sometimes you may want to send the report automatically using email or publishing it to an FTP server. This section demonstrates these techniques.

Emailing Output

This isn't so much an ODS tip as it is a FILENAME tip. You can create a fileref that writes to email rather than to a plain file. You then use this fileref on the FILE= option of the ODS destination. The first thing you should do though is to configure an email server. This can be done using the EMAILSYS and EMAILHOST system options.

```
OPTIONS EMAILSYS=SMTP EMAILHOST='mail.company.com';
```

Then using the FILENAME statement we create an email-based fileref.

```
FILENAME eoutput EMAIL TO='my.address@myemail.com'
SUBJECT='Here is a copy of my report'
FROM='me@myemail.com'
CONTENT_TYPE='text/html';
```


22 ODS Techniques: Tips for Enhancing Your SAS Output

This creates a fileref of the name “eoutput”. You can now use that on the ODS HTML statement to email the report.

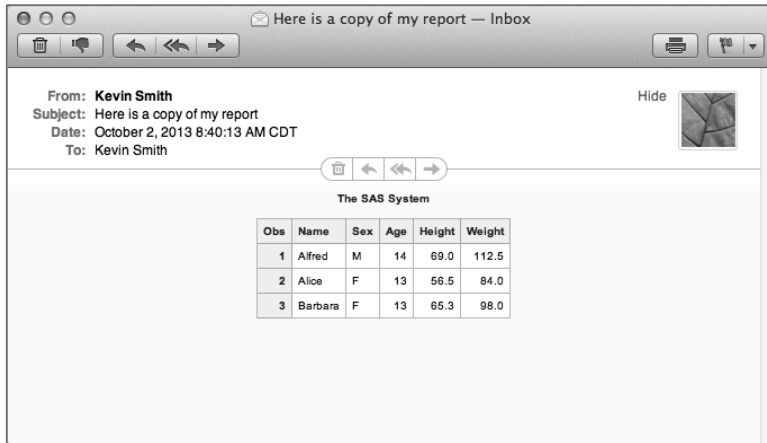
```
ODS HTML FILE=eoutput;  
PROC PRINT DATA=sashelp.class(OBS=3);  
RUN;  
ODS HTML CLOSE;
```

To send a report as an attachment rather than an HTML formatted message, you would use the following code. This is handy when you want to send a report that a mail client couldn't read natively like HTML or plain text. You will want to use this technique for mailing output formats such as PDF or RTF. This is a two-step process. The first is creating the output; the second is sending it using the DATA step.

```
/* Create output file */  
ODS PDF FILE='attachment.pdf';  
PROC PRINT DATA=sashelp.class(OBS=3);  
RUN;  
ODS PDF CLOSE;  
  
/* Email it as an attachment */  
FILENAME attach EMAIL TO='my.address@myemail.com'  
      SUBJECT='See attachment for my report'  
      FROM='me@myemail.com'  
      ATTACH='attachment.pdf'  
      CONTENT_TYPE='application/pdf';  
  
DATA _NULL_;  
FILE attach;  
PUT 'Here is the report I promised you.';  
RUN;
```

Figure 2.14 shows a screen shot of the HTML report in a mail client.

Figure 2.14: Apple Mail displaying an ODS HTML formatted email message



Sending Output to an FTP Server

In addition to being able to send output to email (as demonstrated in the previous tip), you can also send output to an FTP server. This technique is just like the email technique except that you use the FTP access method of the FILENAME statement rather than email. To do this, your FILENAME statement will look like the following.

```
FILENAME ftput FTP '/path/to/remote/file.html'
          HOST='ftp.server.com'
          USER='username' PASS='password'
          RECFM=S; * PROMPT DEBUG;
```

24 ODS Techniques: Tips for Enhancing Your SAS Output

Just as before, we use this fileref on our ODS destination statement in the FILE= option.

```
ODS HTML FILE=ftpout ;

PROC PRINT DATA=sashelp.class(OBS=3) ;
RUN ;

ODS HTML CLOSE ;
```

The PROMPT option can be used to prompt for login information if necessary. The DEBUG option can be used to display information about the FTP connection in case you are having problems. Output 2.3 shows the log output with the DEBUG option turned on.

Output 2.3: Debug output from the SAS log when using the FTP libname engine for ODS output

```
NOTE: Writing HTML Body file: FTPOUT
NOTE: 220 (vsFTPD 2.0.1)
NOTE: <<< 220 (vsFTPD 2.0.1)
NOTE: >>> USER username
NOTE: <<< 331 Please specify the password.
NOTE: >>> PASS XXXXXXXX
NOTE: <<< 230 Login successful.
NOTE: >>> PORT 10,40,11,189,221,70
NOTE: <<< 200 PORT command successful. Consider using PASV.
NOTE: >>> TYPE I
NOTE: <<< 200 Switching to Binary mode.
NOTE: >>> PWD
NOTE: <<< 257 "/users/username"
NOTE: >>> STOR file.html
NOTE: <<< 150 Ok to send data.
NOTE: User username has connected to FTP server on Host ftp.server.com
.
...

NOTE: <<< 226 File receive OK.
NOTE: >>> QUIT
```

In addition to standard FTP, it is also possible to write to an SFTP server using the SFTP access method in a similar manner.

The ODS Path Macro Variable

The ODS path holds the locations of all of the template stores that can be used for table, style, graph, and tagset templates. The default ODS path is SASUSER.TEMPLAT(UPDATE) SASHELP.TMPLMST(READ), where SASUSER.TEMPLAT is used to store your personal templates and SASHELP.TMPLMST holds the templates shipped with SAS. Some users will

change the ODS path in order to store templates in other locations that may be shared with other users at their site. Others may change the path to load templates from a location temporarily, then change it back.

This last technique of changing the ODS path temporarily can cause some headaches in that in order to change it back to what it was previously, you have to know what the path was initially. You can print the ODS path information using the following statement.

```
ODS PATH SHOW;
```

However, this doesn't help in automated processes because it just goes to the log. Luckily, there is a read-only macro variable that contains the ODS path information. The macro variable name is `SYSODSPATH`. Not only does it contain the path information, but it does it in the same syntax that the ODS PATH statement uses. Here is the output from `'%PUT &SYSODSPATH;'`.

```
SASUSER.TEMPLAT(UPDATE) SASHELP.TMPLMST(READ)
```

Here is an example of swapping out the ODS PATH using the `SYSODSPATH` macro variable.

```
/* Display default path */
ODS PATH SHOW;

/* Store a copy of the current ODS path */
%LET MYPATH=&SYSODSPATH;

/* Change the path to something else */
ODS PATH SASHELP.TMPLMST(READ);
ODS PATH SHOW;

/* ... Do operations that require new path ... */

/* Change the ODS path back to the initial value */
ODS PATH &MYPATH;
ODS PATH SHOW;
```

Here is the log output showing that the ODS path was changed and changed back. Obviously, you would run whatever code required the alternate ODS path between changing it and setting it back. This isn't something everyone would need to do, but it's a handy tip for when the occasion comes up.

Here is the original ODS path:

```
1. SASUSER.TEMPLAT(UPDATE)
2. SASHELP.TMPLMST(READ)
```

Here is the path after setting it explicitly:

```
1. SASHELP.TMPLMST(READ)
```

This is the path after setting it using the macro variable contents:

1. SASUSER.TEMPLAT(UPDATE)
2. SASHELP.TMPLMST(READ)

Setting the Default Path for Output Files

Under normal circumstances, the output files from ODS would go to the directory where SAS was invoked. However, this isn't always the directory where you want your files to go. You could add the full path to the front of every output file in your ODS destination statements, as in the code below, but this can be very tedious and error prone.

```
ODS RTF FILE='/path/to/my/directory/file.rtf';
ODS HTML FRAME='/path/to/my/directory/file.html'
          CONTENTS='/path/to/my/directory/contents.html'
          BODY='/path/to/my/directory/body.html';
```

There is an option on the ODS destination statements (RTF family and markup family only) that can clean this up a lot and reduce the possibility of error. This option is the PATH= option. Rather than putting the entire directory in every filename option, you just put the filename itself. Then in the PATH= option, you specify the directory that should be prepended to all of the filenames. Using this option, the code above becomes this.

```
ODS RTF PATH='/path/to/my/directory/' FILE='file.rtf';
ODS HTML PATH='/path/to/my/directory/'
          FRAME='file.html' CONTENTS='contents.html'
          BODY='body.html';
```

You could clean this up even more using a macro variable if you have multiple ODS statements that use the same PATH=.

```
%LET MYPATH=/path/to/my/directory/;
ODS RTF PATH="&MYPATH" FILE='file.rtf';
ODS HTML PATH="&MYPATH"
          FRAME='file.html' CONTENTS='contents.html'
          BODY='body.html';
```

In addition to the content files going to the PATH= directory, graphs will also go to this directory. For destinations that don't support PATH=, you can use GPATH= to specify a separate directory for the graphics files to go.

Exporting ODS Output as “Character” Separated Values

When most users hear the term *CSV*, they probably think “comma-separated values.” In ODS, there is a destination called `TAGSETS.CSV`. While the *C* in this name started out meaning *comma*, the delimiter character is configurable through an option, so I like to call it *character-separated values* to remind people that they don’t have to use commas.

The easiest way to get a comma-separated value file of your ODS output is to use the `TAGSETS.CSV` destination with the ODS sandwich technique.

```
ODS TAGSETS.CSV FILE='output.csv';

PROC PRINT DATA=sashelp.class(OBS=5);
RUN;

ODS TAGSETS.CSV CLOSE;
```

The output for this will look like the Output 2.4.

Output 2.4: Comma-separated value output

```
"Obs", "Name", "Sex", "Age", "Height", "Weight"
"1", "Alfred", "M", 14, 69.0, 112.5
"2", "Alice", "F", 13, 56.5, 84.0
"3", "Barbara", "F", 13, 65.3, 98.0
"4", "Carol", "F", 14, 62.8, 102.5
"5", "Henry", "M", 14, 63.5, 102.5
```

You don’t have to use commas to delimit the data points in the file: Using the `DELIMITER` option, we can change the character to whatever we want. The code below demonstrates an ampersand (&) delimited file.

```
ODS TAGSETS.CSV FILE='output.txt' OPTIONS(DELIMITER='&');

PROC PRINT DATA=sashelp.class(OBS=5);
RUN;

ODS TAGSETS.CSV CLOSE;
```

This program code generates Output 2.5.

Output 2.5: Character-separated value output using ampersand as the delimiter

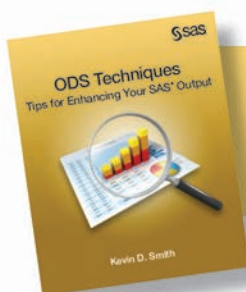
```
"Obs"&"Name"&"Sex"&"Age"&"Height"&"Weight"
"1"&"Alfred"&"M"&14&69.0&112.5
"2"&"Alice"&"F"&13&56.5&84.0
"3"&"Barbara"&"F"&13&65.3&98.0
"4"&"Carol"&"F"&14&62.8&102.5
"5"&"Henry"&"M"&14&63.5&102.5
```

28 *ODS Techniques: Tips for Enhancing Your SAS Output*

There are many other options to control what happens in your CSV files. To get documentation on these other options, run the destination using the `OPTIONS(DOC='HELP')` option.

```
ODS TAGSETS.CSV FILE='output.txt' OPTIONS(DOC='HELP');
```

From *ODS Techniques: Tips for Enhancing Your SAS® Output* by Kevin D. Smith.
Copyright © 2014, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.



From *ODS Techniques*. Full book available for purchase [here](#).

Index

Symbols

^ symbol 27–28

A

accessing data in document output objects
 directly 137-138
 actions, recording in Documents window 128
 ad hoc tables, with report writing interface
 38-40
 adding
 background images 103-105
 JavaScript effects to ODS HTML 84-85
 metadata to PDF documents 106-107
 watermark effects 103-105
 aggregate reports, creating 130-131
 ANCHOR= option 67
 attributes 92-93
 See also specific attributes
 AUTHOR= option 107

B

background effects
 using background images on tables
 58-59
 using gradients backgrounds 88-89
 background images
 adding 103-105
 using on tables 58-59
 BACKGROUNDIMAGE attribute 88
 BITMAP_MODE sub-option, OPTIONS
 statement 92
 bitmaps
 creating of tables 105-106
 inlining 92

blank lines, controlling in PRINT procedure
 31-33

blocks of text, generating 40-42

BODY= option 80

BORDERCOLLAPSE attribute 54

borders and padding

 about 52

 BORDERCOLLAPSE attribute 54

 BORDERSPACING attribute 54

 CELLPADDING attribute 52-53

 PADDING attribute 52-53

BORDERSPACING attribute 54

bulleted lists, generating 40-44

BY group

 in CSS 69-71

 using in worksheet names 120-121

BY variable 131-132

C

CALL EXECUTE, operating on documents
 using 140-142

cascading style sheets (CSS)

 alternating row colors in 56-57

 color models 64-65

 creating styles 49-50

 displaying Document Object Model 65-66

 generating external CSS files 75-76

 BY group styling in 69-71

 importing from CSS 62

 referencing CSS styles 62

 selecting styles based on media types in
 63-64

 setting global font for printer destinations in
 66-67

 specifying multiple CSS files 78

 specifying your own CSS files 76-78

 using 61

 using IDs in 67-68

 using styles in 100-102

- cell merging, preventing in TABULATE
 - procedure 34-35, 123
- CELLPADDING attribute 52-53
- CELLSTYLE_AS statement 37, 55-56
- changing
 - image format used by graphs 85-86
 - page orientation 96-97
- closing all ODS destinations 6
- CMYK(...) 64
- CMYKA(...) 64
- color models, CSS 64
- COLUMN statement 36
- COLUMNS= option 95-96
- COLUMN_SPAN 39
- comparing Excel formats 111-112
- CONTENTS procedure 7, 9, 12, 17, 38
- controlling
 - blank lines in PRINT procedure 31-32
 - borders individually 54-55
 - ODS (Output Delivery System) 3-27
 - page breaks 96, 98-100
 - worksheet names manually 119-120
- COUNT= option 32
- creating
 - aggregate reports 130-131
 - bitmaps of tables 105-106
 - blocks of text 40-44
 - bulleted lists 40-43
 - CSS styles 49-50
 - custom table of contents 125-130
 - document fragments 80
 - external CSS files 75-76
 - nested lists 41-42
 - one-shot tables 29-31
 - output data sets 17-18
 - output using Internet services 21-24
 - samples of styles 46-50
 - secured PDF documents 108
 - spanned rows in REPORT procedure 33-34
 - styles 46-50
 - tables with DATA_NULL_ 35-37
- CSS (cascading style sheets)
 - alternating row colors in 65

- color models 64-65
- creating styles 46-48
- displaying Document Object Model 65-66
- generating external CSS files 75-76
- BY group styling in 69-71
- importing from CSS 61
- referencing CSS styles 62
- selecting styles based on media types in 63-64
- setting global font for printer destinations in 66-67
- specifying multiple CSS files 78
- specifying your own CSS files 76-77
- using 61-71
- using IDs in 67-69
- using styles in 100-102
- CSSSTYLE= option 62, 64, 70, 100-102
- custom fonts, installing 103
- customizing report structure 125-131

D

- data, accessing in document output objects
 - directly 137-138
- DATA argument 39
- DATA= option 41
- DATA step 16, 35-36, 140-142
- DATA_NULL_, creating tables with 35-37
- DEBUG option 23
- DELIMITER option 27
- destinations
 - closing all 4-6
 - interleaving 7-8
 - querying open 16-17
 - running multiple instances of one 8-11
 - suspending 14-16
- determining correct style elements 51-52
- displaying
 - Document Object Model 65-66
 - dynamic variables used by output objects 134-135
 - help in Excel 112
 - output object information 11-12

- templates used by objects 133-134
- DOC_SEQNO= option 137-138
- Document Object Model, displaying 65-66
- DOCUMENT procedure
 - about 128
 - displaying dynamic variables used by output objects 134-135
 - displaying templates used by objects in documents 133-134
- IMPORT TO statement 138-140
- inserting plain text objects into documents 138-140
- LIST statement 140-142
- OBTEMPL statement 133
- operating on documents using CALL EXECUTE 140-142
- REPLAY statement 135-137
- swapping templates during replay 135-137
- using links to create custom tables of contents 128-130
- WHERE clauses 131-132
- Document Recorder 128
- documents
 - See also ODS Document*
 - creating fragments 80
 - inserting arbitrary markup in preamble 80-82
 - inserting plain text objects into 138-140
 - operating on using CALL EXECUTE 140-142
 - PDF 106-110
 - setting titles 78-79
- dynamic variables 133-137

E

- EMAILHOST option 21-22
- emailing output 21-23
- EMAILSYS option 21-22
- END= attribute 30-31
- END statement 41-43
- escape character 18
- Excel (Microsoft)

- See also TAGSETS.EXCELXP*
- about 111
- comparing formats 111-112
- controlling worksheet names manually 119-120
- displaying help 119-120
- formats 111-112, 113-114
- preserving leading zeros 115-117
- preventing cell merging in TABULATE procedure 123
- tagsets online 123-124
- using BY group information in worksheet names 120-121
- using graphical output in 122
- excluding output objects 12-14
- exporting ODS output as "character" separated values 27-28
- EXPRESSION(...) function 60-61
- external CSS files, generating 75-76
- external resources
 - linking to your own custom JavaScript 84-85
 - using 82-85
 - web fonts 82-83

F

- FILE= option 78-80, 112
- FILE PRINT ODS statement 35-37
- FILENAME statement 21-22, 23-24
- FONTPATH statement 103
- FONTREG procedure 103
- fonts, installing custom 103
- FORMAT_CELL method 39
- formats, Excel 111-112, 113-114
- FTP servers, sending output to 23-24

G

- generating
 - aggregate reports 130-131
 - bitmaps of tables 105-106

- blocks of text 40-44
- bulleted lists 40-44
- CSS styles 49-50
- custom table of contents 125-127, 128-130
- document fragments 80
- external CSS files 75-76
- nested lists 41-43
- one-shot tables 29-31
- output data sets 17-18
- output using Internet services 21-24
- samples of styles 46-47
- secured PDF documents 108
- spanned rows in REPORT procedure 33-34
- styles 48-49
- tables with DATA_NULL_ 35-37

GIF format 86

global font, setting for printer destinations in
CSS 66-67

Google Fonts (website) 82-83

GOPTIONS statement 86

gradients backgrounds 88-89

graphical output, using in Excel 122

GRAPHICS statement 86

graphs, changing image format used by 85-86

H

HEADTEXT= option 81-82, 84-85

help, displaying in Excel 112

HSL(...) 65

HSLA(...) 65

HTML

- about 112
- background effects 86-89
- cascading style sheet (CSS) 75-78
- changing image format used by graphs
85-86
- creating document fragments 80
- generating external CSS files 75-76
- inlining all style attributes 92-93
- inlining bitmapped images 92
- inserting arbitrary markup in document
preamble 80-82

- linking to custom JavaScript 84-85
- ODS HTML5 techniques 91-93
- preserving white space 89-91
- setting document title 78-79
- specifying CSS files 76-78
- turning off inline SVGs 91-92
- using background images on tables 87-88
- using external resources 82-85
- using gradients backgrounds 88-89
- versions 73-75
- web fonts 82-83

HTMLSTYLE= attribute 114

I

IDs, using in CSS 67-69

images

- adding to background 103-105
- bitmap 92, 105-106
- changing formats used by graphs 85-86
- using background images on tables
87-88

IMPORT TO statement, DOCUMENT
procedure 138-140

importing CSS from CSS 62

inline functions, general syntax for 18-19

inline styling, adding to reports 19-20

inline SVGs, turning off 91-92

inlining

- all style attributes 92-93
- bitmapped images 92

inserting

- arbitrary markup in document preamble
80-82
- plain text objects into documents 138-140
- special characters 20-21

installing custom fonts 103

interleaving ODS destinations 7-8

Internet services, generating output using 21-24

ITEM statement 41-43

J

JavaScript, linking to your own custom 84-85
 jQuery plug-in 84-85

K

KEYWORDS= option 107

L

LABEL option, ODS TRACE ON statement
 11-12

leading zeros, preserving 115-116

links

creating aggregate reports with 130-131
 using to create custom tables of contents
 128-130

to your own custom JavaScript 84-85

LIST statement, DOCUMENT procedure
 140-142

M

markup, inserting in document preamble
 80-82

metadata, adding to PDF documents 106-107

Microsoft Excel

See Excel (Microsoft)

multicolumn output without layout 95-96

N

nested lists, creating 41-43

NO_BOTTOM_MATTER= sub-option, FILE=
 option 80

NOCELLMERGE option, TABULATE
 procedure 34, 123

NO_TOP_MATTER= sub-option, FILE= option
 80

NOWRAP value 89-90

O

OBTEMPL statement, DOCUMENT procedure
 133

ODS (Output Delivery System)

See also specific topics

controlling 3-16

creating output data sets 17

displaying output object information 11-12

emailing output 21-23

excluding output objects 12-14

exporting ODS output as "character"
 separated values 27-28

generating output using Internet services
 21-24

inserting special characters 20-21

interleaving ODS destinations 7-8

ODS CLOSE statement 6

ODS path macro variable 24-26

ODS sandwich 4-5

ODS_ALL_CLOSE statement 6-7

querying open ODS destination information
 16-17

running multiple ODS destinations 8-11

selecting output objects 12-14

sending output to FTP servers 23-24

setting default path for output files 26

suspending ODS destinations 14-161

using ODS inline functions and formatting
 18-21

ODS CLOSE statement 6

ODS destinations

closing all 6-7

interleaving 7-8

querying open 16-17

running multiple instances of one 8-11

suspending 14-16

ODS Document

about 125

accessing data in document output objects
 directly 137-138

creating aggregate reports 130-131

- creating custom table of contents 125-127, 128-130
- customizing report structure 125-131
- displaying dynamic variables used by output objects 134-135
- displaying templates used by objects 133-134
- dynamic variables 133-137
- inserting plain text objects into documents 138-140
- operating on documents using CALL EXECUTE 140-142
- recording actions in Documents window 128
- swapping templates during replay 135-137
- templates 133-137
- using WHERE clauses in 131-132
- ODS DOM 67
- ODS EXCLUDE ALL; command 14
- ODS EXCLUDE NONE; command 14
- ODS EXCLUDE statement 12-14, 14-16
- ODS HTML CLOSE; command 4
- ODS HTML; command 4, 67, 100-102
- ODS HTML CSSSTYLE= option 68-69
- ODS HTML statement 70-71, 78
- ODS HTML3 74
- ODS HTML4 74
- ODS HTML5
 - about 74
 - inlining all style attributes 92-93
 - inlining bitmapped images 92
 - techniques 91-93
 - turning off inline SVGs 91-92
- ODS LISTING CLOSE; command 4-5
- ODS LISTING; command 4-5
- ODS path macro variable 24-26
- ODS PATH statement 24-26
- ODS PDF EXCLUDE ALL; command 14
- ODS PDF EXCLUDE NONE; command 14
- ODS PDF statement 96-97, 100-105
- ODS PRINTER destination 105
- ODS sandwich 4-5
- ODS SELECT ALL; command 14

- ODS SELECT NONE; command 14
- ODS SELECT statement 12-14, 14-16
- ODS TRACE 17
- ODS TRACE DOM statement 66-67
- ODS TRACE ON statement 11-14, 66
- ODS_ALL_CLOSE; statement 6-7
- ODSLIST procedure 40-43, 43-44
- ODSTABLE procedure 29-31, 35-37, 60
- ODSTEXT procedure 43-44
- one-shot tables, creating 29-31
- online tagsets 123-124
- operating on documents using CALL EXECUTE 140-142
- options
 - See specific options*
 - OPTIONS PDFPASSWORD= option 108
 - OPTIONS PDFSECURITY= option 108
 - OPTIONS statement 91-92, 96-97, 105-106, 112
- orientation, page 96-98
- output
 - emailing 21-23
 - exporting as "character" separated values 27-28
 - generating using Internet services 21-24
 - sending to FTP servers 23-24
- output data sets, creating 17-18
- Output Delivery System
 - See ODS (Output Delivery System)*
- output files, setting default path for 26
- output objects
 - displaying 11-12
 - excluding 12-14
 - selecting 12-14

P

- padding
 - See borders and padding*
- PADDING attribute 52-53
- page breaks, controlling 96, 98-100
- page orientation, changing 96-97
- PATH= option 26

PDF documents
 adding metadata to 106-107
 creating secured 108
 specifying initial page view 109-110

PDF ESCAPECHAR statement 18

PDF PASSWORD= option 108

PDFPAGEVIEW= option 109-110

PDFSECURITY= option 108

plain text objects, inserting into documents
 138-140

PNG format 86

PRE.NOWRAP value 89-91

preserving
 leading zeros 115-117
 white space 89-91

preventing cell merging in TABULATE
 procedure 34-35, 123

PRINT procedure
 adding JavaScript effects to ODS HTML
 84-85
 controlling blank lines in 31-33
 creating one-shot tables with 29
 preserving whitespace 89-91
 using Excel formats 113
 using IDs in CSS 68-69

PRINTERPATH= option, OPTIONS statement
 105-106

printers
 adding background images 103-105
 adding metadata to PDF documents
 106-107
 adding watermark effects 103-105
 creating bitmaps of tables 105-106
 creating secured PDF documents 108
 installing custom fonts 103
 multicolumn output without layout 95-96
 orientation 96-97
 page breaks 96, 98-100
 specifying initial PDF page view 109-110
 styles in ODS PDF 100-105
 using CSS-based styles 100-102

procedures
See specific procedures

PROCLABEL= option 125

PROMPT option 24

Q

querying open ODS destination information
 16-17

R

recording actions in Documents window 128

replay, swapping templates during 135-137

REPLAY statement, DOCUMENT procedure
 135-137

REPORT procedure
 alternating row colors in 57
 creating one-shot tables with 29
 creating spanned rows in 33-34
 SPANROWS option 33-34
 using Excel formats 113-114
 using IDs in CSS 68-69

report writing interface, ad hoc tables with
 38-40

reports
 aggregate 130-131
 customizing structure 125-131

RESOLVE(...) function 59-60

RGB(...) 64-65

RGBA(...) 64-65

RGBA colors 58-59

Roboto font 82-83

ROW_SPAN 39

running multiple instances of one destination
 8-11

S

SAS 9.4 Output Delivery System User's Guide 1

SAS System Options: Reference 108

SASEDOC libname 137-138

SASHELP.VDEST data set view 16

- selecting
 - output objects 12-14
 - styles based on media types in CSS 63-64
- sending output to FTP servers 23-24
- setting
 - default path for output files 26
 - document title 78-79
 - global font for printer destinations in CSS 66-67
- SHEET_INTERVAL= option 119, 120-121
- SHEET_NAME= option 119
- spanned rows, creating in REPORT procedure 33-34
- SPANROWS option, REPORT procedure 33-34
- special characters, inserting 20-21
- specifying
 - initial PDF page view 109-110
 - multiple CSS files 78
 - your own CSS files 76-78
- START= attribute 30-31
- STARTPAGE= option 98-100
- style
 - advanced traffic lighting functions 59-61
 - alternating table row colors 56-58
 - BORDERCOLLAPSE attribute 54
 - borders 52-55
 - BORDERSPACING attribute 54
 - CELLPADDING attribute 52-53
 - controlling borders individually 54-55
 - creating 48-49
 - creating CSS 49-50
 - CSS 61-71
 - CSS color models 64-65
 - determining correct style elements to use 51-52
 - displaying Document Object Model 65-66
 - generating samples of 46-47
 - BY group styling in CSS 69-71
 - importing CSS from CSS 62
 - in ODS PDF 100-105
 - padding 52-55
 - PADDING attribute 52-53

- referencing CSS styles 62
- RGBA colors 58-59
- selecting based on media types in CSS 63-64
- setting global font for printer destinations in CSS 66-67
- using CSS-based 100-102
- using IDs in CSS 67-69
- STYLE= option 32-33, 104
- STYLE_ATTR argument 39
- STYLE_ELEM argument 39
- STYLESHEET= option 75-76, 76-78
- styling template-based tables 37-38
- SUBJECT= option 106-107
- suspending ODS destinations 14-16
- SVG_MODE sub-option, OPTIONS statement 91-92
- swapping templates during replay 135-137
- SYMGET(...) function 59-60
- %SYSGET macro 103
- SYSODSPATH macro variable 24-26

T

- table of contents 125-127, 128-130
- tablecloth jQuery plug-in 84-85
- tables
 - ad hoc with report writing interface 38-40
 - alternating row colors 56-58
 - controlling blank lines in PRINT procedure 31-33
 - creating bitmaps of 105-106
 - creating one-shot tables 29-31
 - creating spanned rows in REPORT procedure 33-34
 - creating with DATA_NULL_ 35-37
 - preventing cell merging in TABULATE procedure 34-35
 - styling template-based 37-38
 - using background images on 87-88
- TABULATE procedure
 - NOCELLMERGE option 34-35, 123
 - preventing cell merging in 34-35, 123-124

TAGATTR= attribute 113, 115, 117-118
 tagsets 123-124
 TAGSETS.CSV destination 27-28, 112
 TAGSETS.EXCELXP
 about 112, 123
 controlling worksheet names manually
 119-120
 displaying help 112
 preserving leading zeros 115-116
 preventing cell merging in TABULATE
 procedure 123
 using 113-114
 using BY group information in worksheet
 names 120-121
 using Excel formulas in 117-119
 using graphical output in Excel 122
 TAGSETS.MSOFFICE2K
 about 112, 113-114
 preserving leading zeros 115-116
 using graphical output in Excel 123-124
 TAGSETS.MSOFFICE2K_X 123-124
 TAGSETS.STYLE_POPUP destination 51-52
 TEMPLATE procedure
 about 29, 35
 applying padding to cells 53
 creating CSS styles 49-50
 creating styles 48-49
 CSS color models 64-65
 displaying Document Object Model 65-66
 template-based tables, styling 37-38
 templates
 displaying used by objects 133-134
 swapping during replay 135-136
 text blocks 40- 41
 TEXTFILE= option 138 - 140
 TITLE= option 107
 TITLE= sub-option, FILE option 21-22
 traffic lighting 59-61
 turning off inline SVGs 91-92

V

variables, dynamic 133-137

W

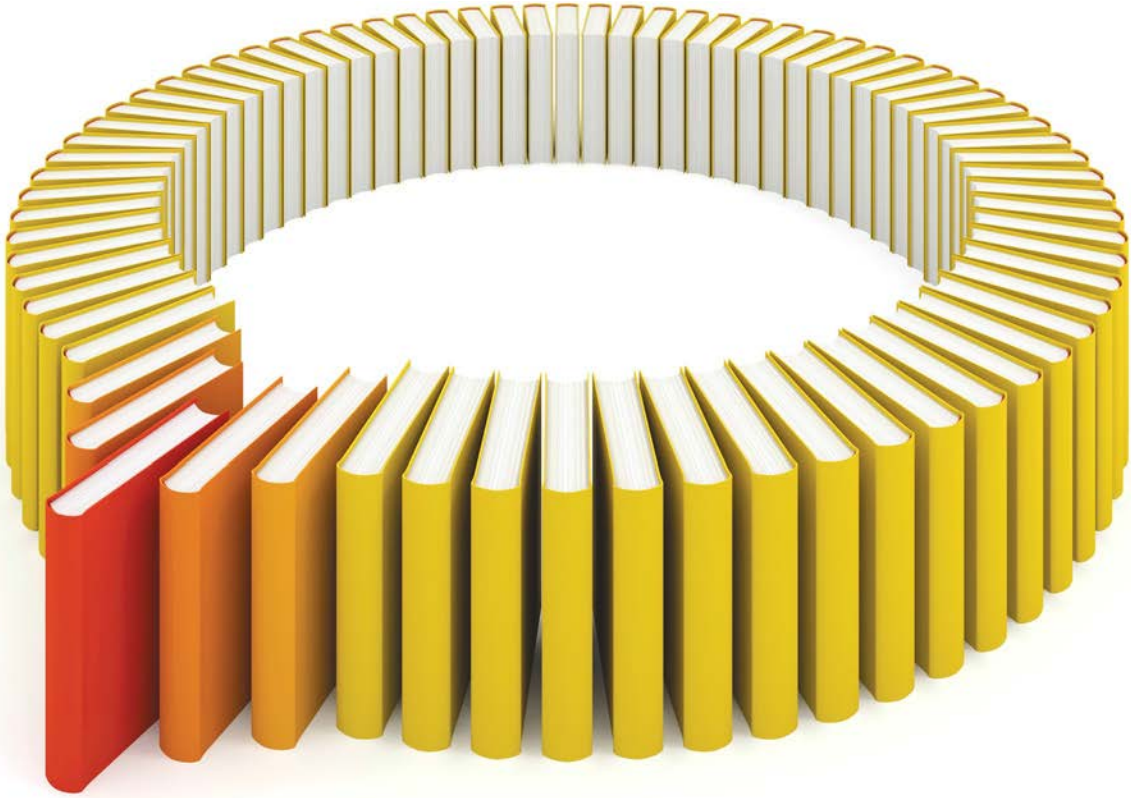
watermark effects 103-105
 web fonts, utilizing 82-83
 WHERE clauses, using in documents 131-132
 white space, preserving 89-91
 WHITESPACE attribute 89-91
 WRITE access method 129

About The Author



Kevin D. Smith has been a software developer at SAS since 1997. He has supported PROC TEMPLATE and other underlying ODS technologies for most of that time. Kevin has spoken at numerous SAS Global Forums as well as at regional and local SAS users groups with the "From Scratch" series of presentations that were created to help users of any level master various ODS technologies. Kevin is also the author of many of the ODS tip sheets available on the SAS support Web site.

Learn more about this author by visiting his author page at http://support.sas.com/publishing/authors/smith_kevin.html. There you can download free book excerpts, access example code and data, read the latest reviews, get updates, and more.



Gain Greater Insight into Your SAS[®] Software with SAS Books.

Discover all that you need on your journey to knowledge and empowerment.

 support.sas.com/bookstore
for additional books and resources.


THE POWER TO KNOW.®