Technical Paper

Using Style Elements in the REPORT and TABULATE Procedures

Table of Contents

Introduction

When you invoke the SAS® Output Delivery System (ODS) to generate output, the application uses a default style. Yet, users often find that they need to customize their output even more. With many SAS procedures, you can use the TEMPLATE procedure to modify your output. However, for a few procedures, such as the REPORT procedure and the TABULATE procedure, the output cannot be fully modified using PROC TEMPLATE.

You can create customized ODS output with PROC REPORT and PROC TABULATE by using the STYLE= option directly in the procedures. Knowing which option to use in order to change the attribute of a particular item in the output report can be a challenge. This paper addresses this challenge and illustrates how to use the STYLE= option in the REPORT and TABULATE procedures to customize output.

STYLE= Option: Syntax and General Description

The *STYLE= option* enables you to specify style (presentation) elements for various parts of a report. The syntax for the STYLE= option follows:

STYLE<(*location(s)*)>=<*style-element-name*><[*style-attribute-specification(s)*]>

Note: You can use braces instead of the square brackets around the style attribute specifications.

STYLE<(*location(s)*)>=<*style-element-name*><{*style-attribute-specification(s)*}>

The STYLE= option uses six different location values to identify the part of the report that the option affects. The location values and their associated default style elements follow:

- **REPORT—**Denotes the structural, or underlying, part of the report. **Default style element:** Table
- **COLUMN—**Denotes the cells in all the columns. **Default style element:** Data
- **HEADER—**Denotes all column headers, including spanned headers. **Default style element:** Header
- **SUMMARY—**Denotes all of the default summary lines that are produced by a BREAK or an RBREAK statement. **Default style element:** DataEmphasis
- **LINES—**Denotes all LINE statements in all COMPUTE blocks. **Default style element:** NoteContent
- **CALLDEF—**Denotes all cells that are identified by a CALL DEFINE statement. **Default style element:** Data

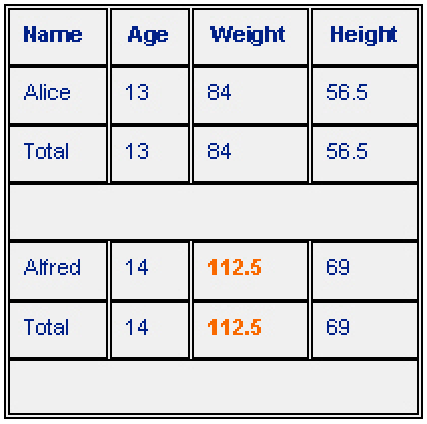
If a location value is not specified in the STYLE= option, ODS defaults to the REPORT value.

Using Styles in the REPORT Procedure

In the REPORT procedure, the PROC REPORT statement is the highest level at which you can request styles. The following example illustrates the use of the six STYLE= location values in the PROC REPORT statement:

```
ods html file='my-file.htm';

proc report nowd data=sashelp.class(obs=2)
            style(report)=[background=green cellspacing=10]
            style(column)=[background=yellow]
            style(header)=[background=pink]
            style(summary)=[background=lightblue]
            style(lines)=[background=white]
            style(calldef)=[background=purple];
   col name age weight height;
   define age / order;
   break after age / summarize;

   compute weight;
      if weight.sum > 100 then
         call define ('weight.sum','style','style=[foreground=orange
                        font_weight=bold]');
   endcomp;

   compute after age;
      name='Total';
      line ' ';
   endcomp;

run;

ods html close;
```

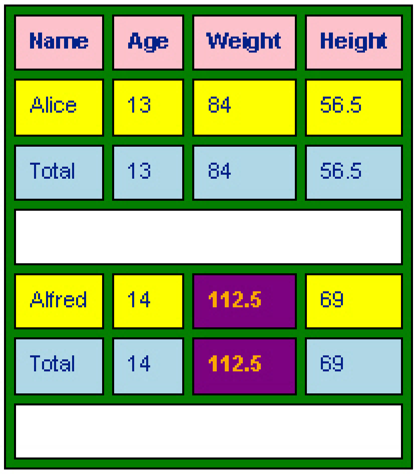
Without the instances of the STYLE= option in the PROC REPORT statement, ODS generates the following default output:



Name	Age	Weight	Height
Alice	13	84	56.5
Total	13	84	56.5
Alfred	14	**112.5**	69
Total	14	**112.5**	69

Output 1. Default Output without the STYLE= Option

With the six instances of the STYLE= option, ODS generates the following output:



Name	Age	Weight	Height
Alice	13	84	56.5
Total	13	84	56.5
Alfred	14	112.5	69
Total	14	112.5	69

Output 2. Output Generated with the STYLE= Option

In this example, all six locations in the report have a background color change. Note the cell-spacing attribute in the STYLE= option that specifies the report location. This attribute enhances the report further by designating the amount of space, in points (CELLSPACING=10), between cells. You should also be aware that you cannot specify STYLE(CALLDEF) unless you have a CALL DEFINE statement in the code.

Using Styles in the DEFINE Statement

In PROC REPORT, the DEFINE statement enables you to specify styles for the column or header locations for an individual variable. If no location value is listed in the DEFINE statement, the default location value is HEADER.

In the DEFINE statement:

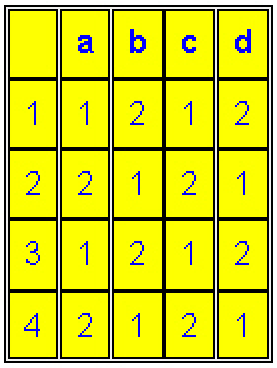
- Styles that are specified in a DEFINE statement override the styles that are listed in the PROC REPORT statement.
- The specification STYLE(COLUMN)= applies to the entire column.
- The specification STYLE(HEADER)= applies to the header.
- The specification STYLE(COLUMN HEADER) applies to both locations.

The following example illustrates the use of the STYLE= option in DEFINE statement.

```
proc format;
   value colorme 1='white' 2='purple';
```

```
data flower;
   input row cols $ value;
   cards;
1 a 1
1 b 2
1 c 1
1 d 2
2 a 2
2 b 1
2 c 2
2 d 1
3 a 1
3 b 2
3 c 1
3 d 2
4 a 2
4 b 1
4 c 2
4 d 1
;
run;

ods html file='my-file.htm';

proc report nowd data=flower
            style=[cellwidth=.1in rules=group frame=void]
            style(column header)=[background=yellow foreground=blue];
   col row cols , value;
   define row   / group ' '
          style(column header)=[foreground=green background=green];
   define value /
          style(column)=[background=colorme. foreground=colorme.] ' ';
   define cols   / across
          style(header)=[foreground=green background=green] ' ';
run;

ods html close;
```

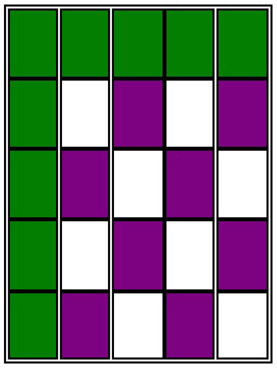
Without the STYLE= option in the DEFINE statements, ODS generates the following output:



	a	b	c	d
1	1	2	1	2
2	2	1	2	1
3	1	2	1	2
4	2	1	2	1

Output 3. Output Generated without the STYLE= Option in the DEFINE Statement

With the STYLE= option in the DEFINE statements, the following output is generated:



Output 4. Output Generated with the STYLE= Option in the DEFINE Statement

In this example, the BACKGROUND=YELLOW and FOREGROUND=BLUE assignments in the PROC REPORT statement are overridden, as follows, by the STYLE= assignments in the DEFINE statements:

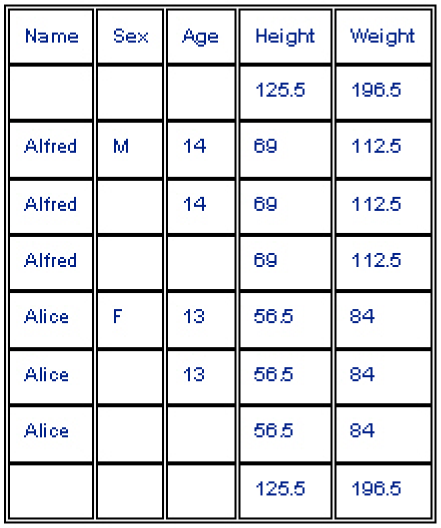
- The background and the foreground colors for the Row variable's column and header are set to green.
- The background and foreground colors for the Value variable's column are set to the colors found in the COLORME. format.
- The background and foreground colors for the Cols variable's header are set to green.

Using Styles in the BREAK and RBREAK Statements

When you apply a style in the BREAK or RBREAK statements, that style overrides any style that is specified in the PROC REPORT statement. The style is applied to only the particular BREAK statement variable or to the RBREAK level. You can use this technique to help distinguish one break row from another, as shown in the following example:

```
ods html file='my-file.htm';

proc report nowd data=sashelp.class (obs=2)
            style(column header summary)=[font_size=1 background=white];
   define name / order;
   define age  / order;
   break after name / summarize style=[background=green font_weight=bold];
   break after age  / summarize style=[background=yellow font_style=roman];
   rbreak after / summarize style=[background=orange];
   rbreak before / summarize;
run;

ods html close;
```

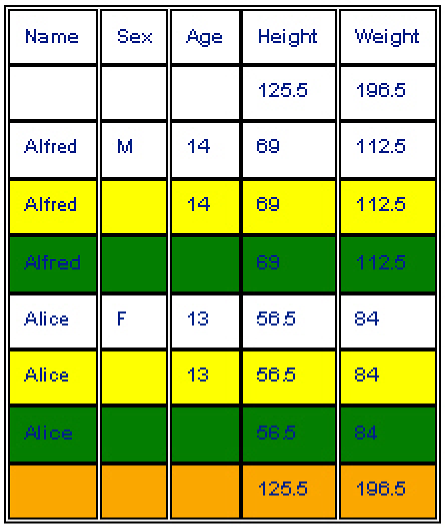
Without the STYLE= option in the BREAK and RBREAK statements, ODS generates the following output:



Name	Sex	Age	Height	Weight
			125.5	196.5
Alfred	M	14	69	112.5
Alfred		14	69	112.5
Alfred			69	112.5
Alice	F	13	56.5	84
Alice		13	56.5	84
Alice			56.5	84
			125.5	196.5

Output 5. Output Generated without the STYLE= Option in the BREAK or RBREAK Statements

With the STYLE= option in the BREAK and RBREAK statements, ODS generates the following output:



Name	Sex	Age	Height	Weight
			125.5	196.5
Alfred	M	14	69	112.5
Alfred		14	69	112.5
Alfred			69	112.5
Alice	F	13	56.5	84
Alice		13	56.5	84
Alice			56.5	84
			125.5	196.5

Output 6. Output Generated with the STYLE= Option in the BREAK or RBREAK Statements

In this example, the assignments of FONT_SIZE=1 and BACKGROUND=WHITE in the PROC REPORT statement for the columns, the header, and the summary rows are overridden, as follows, by the STYLE= assignments in the BREAK and RBREAK statements.

- The BREAK AFTER NAME statement sets the background of the break row for NAME to green and the weight of the font to bold.
- The BREAK AFTER AGE statement sets the background of the break row for AGE to yellow and the font style to Roman.

- The RBREAK AFTER sets the background color for the summary row to orange.
- The RBREAK BEFORE statement reverts back to the style from the PROC REPORT statement.

Using Styles in the COMPUTE Statement

The COMPUTE statement begins a *compute block*, which contains one or more statements that PROC REPORT executes as it builds a report. In a compute block, any STYLE= settings apply to the LINE statements within that block. You can use in-line formatting for different parts of the LINE statement, but some options (such as the JUST= option) are applied only via the STYLE= option in the COMPUTE statement.

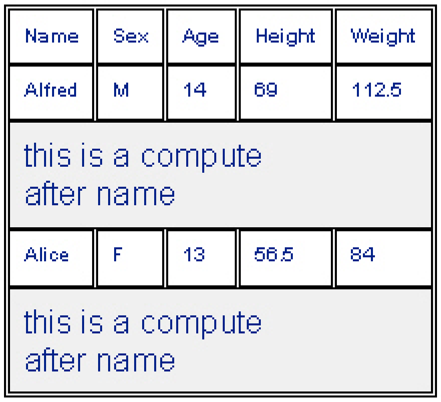
Regardless of the number of LINE statements in a compute block, a LINE statement is treated as one, long string. As is the case with other statements, when you apply a style in a compute block, that style overrides any style that is specified in the PROC REPORT statement.

The following example illustrates the use of the STYLE= option in a COMPUTE statement:

```
ods escapechar='^';
ods html file='my-file.htm';

proc report nowd data=sashelp.class (obs=2)
            style(column header summary)=[font_size=1 background=white];
   define name / order;
   define age  / order;

   compute after name / style=[background=pink ];
      line '^S={background=green} this is a compute';
      line 'after name';
   endcomp;

run;

ods html close;
```

Note: The compute block requires a forward slash (/) prior to the STYLE= option.

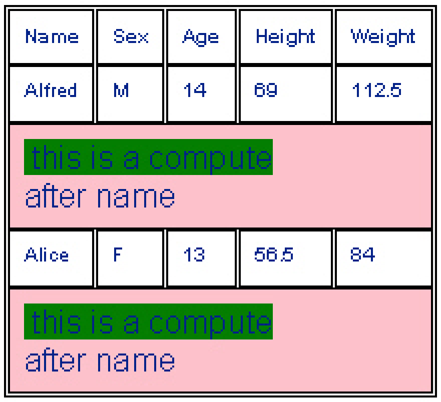
Without the STYLE= option in the COMPUTE statement, ODS generates the following output:



Name	Sex	Age	Height	Weight
Alfred	M	14	69	112.5
this is a compute after name				
Alice	F	13	56.5	84
this is a compute after name				

Output 7. Output Generated without the STYLE= Option in the COMPUTE Statement

With the STYLE= option in the COMPUTE statement and an in-line format, ODS generates the following output:



Name	Sex	Age	Height	Weight
Alfred	M	14	69	112.5
this is a compute after name				
Alice	F	13	56.5	84
this is a compute after name				

Output 8. Output Generated with the STYLE= Option in the COMPUTE Statement

In this example, the STYLE= option changes the LINE statement background to pink, and the in-line formatting changes the background of the text from the first LINE statement to green. Note that the in-line formatting only applies to the text string in the LINE statement and not to the entire line.

Styles in a CALL DEFINE Statement

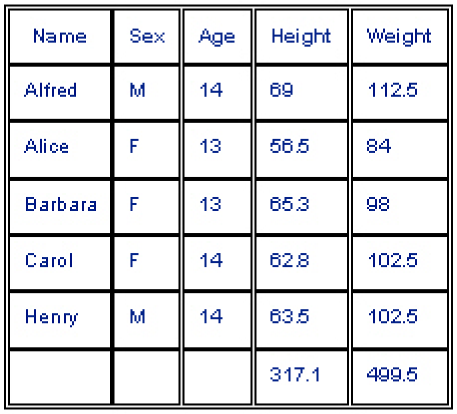
The *STYLE attribute* indicates which style element to use in the cells that are affected by the CALL DEFINE statement. In the CALL DEFINE statement, the *STYLE= value* (which is different than the STYLE attribute) functions like the STYLE= option functions in other PROC REPORT statements. However, because the CALL DEFINE statement requires values for the *column-id, attribute-name,* and *value* arguments, the STYLE= value is treated as the value for the STYLE attribute rather than as an option.

In the following program, the STYLE= value in the CALL DEFINE statement overrides the STYLE(CALLDEF)= option in the PROC REPORT statement. Keep in mind that the styles are replaced rather than concatenated when they are applied from a CALL DEFINE statement. In SAS 9.2, a new attribute called STYLE/MERGE will concatenate the styles.

The following program illustrates the use of the STYLE= value in a CALL DEFINE statement:

```
ods html file='myfile.htm';

proc report nowd data=sashelp.class (obs=5)
            style(calldef)=[background=green]
            style(column header)=[font_size=1 background=white];
   define name / order;
   define age  / order;
   rbreak after / summarize;
   compute age;
     if age=13 then do;
        call define(_row_,'STYLE','STYLE=[foreground=red]');
        call define('age','STYLE','STYLE=[background=yellow]');
     end;

     if _break_='_RBREAK_' then
        call define(_row_,'STYLE','STYLE=[foreground=white]');
   endcomp;

run;

ods html close;
```

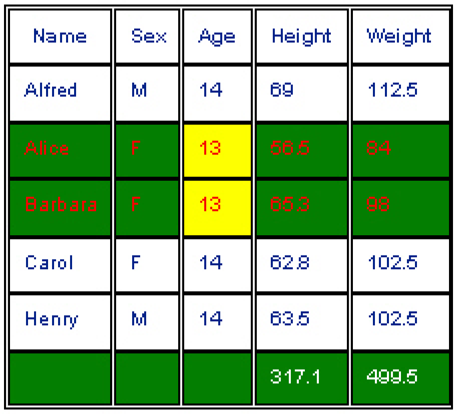
Without the STYLE= value in the CALL DEFINE statement, ODS generates the following output:



Name	Sex	Age	Height	Weight
Alfred	M	14	69	112.5
Alice	F	13	56.5	84
Barbara	F	13	65.3	98
Carol	F	14	62.8	102.5
Henry	M	14	63.5	102.5
			317.1	499.5

Output 9. Output Generated with the STYLE= Option in the COMPUTE Statement

With the STYLE attribute and the STYLE= value within a CALL DEFINE statement, ODS generates the following output:



Name	Sex	Age	Height	Weight
Alfred	M	14	69	112.5
Alice	F	13	56.5	84
Barbara	F	13	65.3	98
Carol	F	14	62.8	102.5
Henry	M	14	63.5	102.5
			317.1	499.5

Output 10. Output Generated with the STYLE= Option in the COMPUTE Statement

Note: The STYLE(CALLDEF)= option only applies if a CALL DEFINE statement is used in a compute block.

In this example, the STYLE(CALLDEF)= option sets the background to green when the CALL DEFINE statement is requested. In the compute block, when the IF condition it true, the CALL DEFINE statement changes the foreground to red and the background for the Age cells to yellow. At the row break, the CALL DEFINE statement changes the foreground to white.

Styles in the TABULATE Procedure

Adding styles for reports generated by PROC TABULATE is similar to adding them for reports generated by PROC REPORT.

In PROC TABULATE, the syntax for the STYLE= option is as follows:

STYLE=<*style-element-name*|<PARENT>>[*style-attribute-name*=*style-attribute-value*<... *style-attribute-name*=*style-attribute-value*>]

The following TABULATE procedure statements support style options:

- **PROC TABULATE**—The style attributes used in this statement affect the data in the table cells.
- **CLASS**—The style attributes used in this statement affect the CLASS variable headings.
- **CLASSLEV**—The style attributes used in this statement affect the levels (values) of the CLASS variables.
- **VAR**—The style attributes used in this statement affect the analysis variable headings.
- **TABLE**—The style attributes used in this statement can affect the variable headings, the empty box above the row titles, the data in the table cells, and table attributes such as gridlines and spacing.
- **KEYWORD**—The style attributes specified in this statement affect the headings for PROC TABULATE keywords, such as ALL, N, PCTN, and so on.

The following style attributes are valid for all PROC TABULATE statements except the TABLE statement:

ASIS=	FONT_WIDTH=
BACKGROUND=	HREFTARGET=
BACKGROUNDIMAGE=	JUST=
BORDERCOLOR=	NOBREAKSPACE=
BORDERCOLORDARK=	POSTHTML=
BORDERCOLORLIGHT=	POSTIMAGE=
BORDERWIDTH=	POSTTEXT=
CELLHEIGHT=	PREHTML=
FLYOVER=	PRETEXT=
FONT=	PROTECTSPECIALCHARS=
FONT_FACE=	TAGATTR=
FONT_SIZE=	URL=
FONT_STYLE=	VJUST=
FONT_WEIGHT=	

Valid style attributes for the TABLE statement follow:

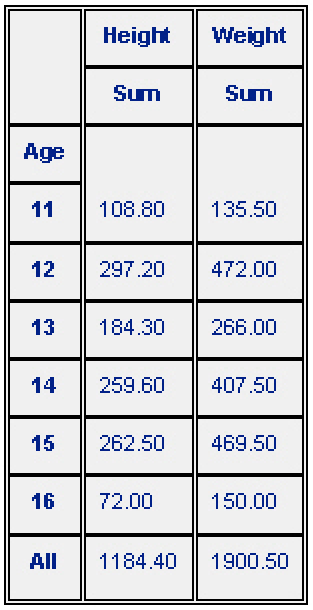
BACKGROUND=	CELLSPACING=
BACKGROUNDIMAGE=	FONT=
BORDERCOLOR=	FONT_FACE=
BORDERCOLORDARK=	FONT_SIZE=
BORDERCOLORLIGHT=	FONT_STYLE=
BORDERWIDTH=	FONT_WEIGHT=
CELLPADDING=	FONT_WIDTH=
FOREGROUND=	POSTIMAGE=
FRAME=	POSTTEXT=
HTMLCLASS=	PREHTML=
JUST=	PREIMAGE=
OUTPUTWIDTH=	PRETEXT=
POSTHMTL=	RULES=

Styles in the CLASS, VAR, and KEYWORD Statements

In the following program, the STYLE= option is used to customize the headings that are created by the CLASS, VAR, and KEYWORD statements:

```
ods html file='test.html';

proc tabulate data=sashelp.class;
   class age / style=[background=red];
   var height / style=[foreground=blue];
   var weight / style=[foreground=green];


   table age all, height weight;
   keyword sum all / style=[font_style=italic];
run;

ods html close;
```

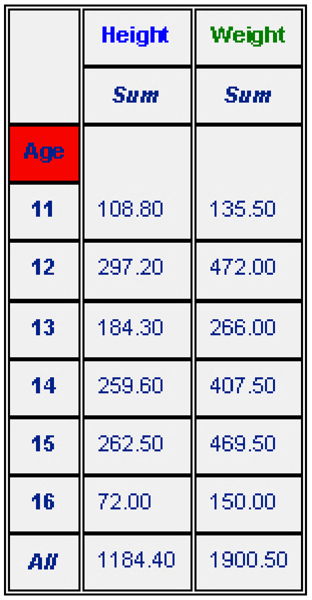
Without the STYLE= option in the CLASS, VAR, and KEYWORD statements, ODS generates the following output:



	Height	Weight
	Sum	Sum
Age		
11	108.80	135.50
12	297.20	472.00
13	184.30	266.00
14	259.60	407.50
15	262.50	469.50
16	72.00	150.00
All	1184.40	1900.50

Output 11. Output Generated without the STYLE= Option in the CLASS, VAR, and KEYWORD Statements

With the STYLE= option in the CLASS, VAR, and KEYWORD statements, ODS generates the following output:



	Height	Weight
	Sum	*Sum*
Age		
11	108.80	135.50
12	297.20	472.00
13	184.30	266.00
14	259.60	407.50
15	262.50	469.50
16	72.00	150.00
All	1184.40	1900.50

Output 12. Output Generated with the STYLE= Option in the CLASS, VAR, and KEYWORD Statements

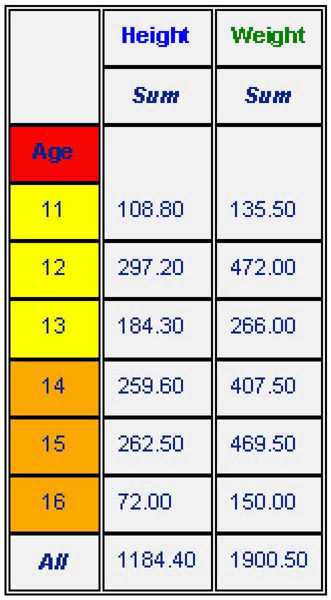
In this example, the program applies colors to the Age, Height, and Weight variable headings as well as to the Sum and All keyword labels (shown in italic).

Styles in the CLASSLEV Statement

To apply styles to the levels (values) of a CLASS variable, use the CLASSLEV statement. To apply colors to the values, create a format with the FORMAT procedure. This format **does not** group the values.

```
ods listing close;
ods html file=’test.html';

proc format;
  value colfmt 11-13='yellow' 14-16='orange';
run;

proc tabulate data=sashelp.class;
   class age / style=[background=red];
   classlev age / style=[font_weight=medium background=colfmt.];
   var height / style=[foreground=blue];
   var weight / style=[foreground=green];
   table age all, height weight;
   keyword sum all / style=[font_style=italic];
run;

ods listing;
ods html close;
```

This program generates the following output:



	Height	Weight
	Sum	*Sum*
Age		
11	108.80	135.50
12	297.20	472.00
13	184.30	266.00
14	259.60	407.50
15	262.50	469.50
16	72.00	150.00
All	1184.40	1900.50

Output 13. Output Generated with the STYLE= Option in the CLASSLEV Statement

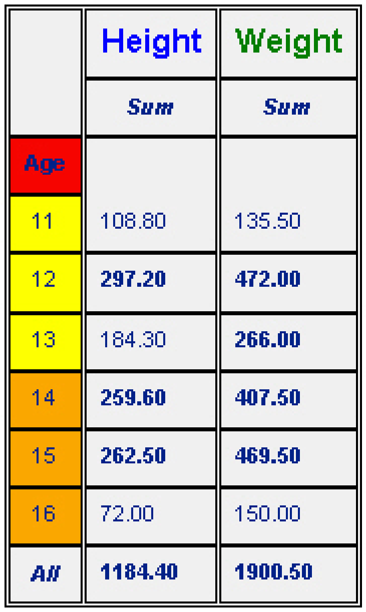
In addition to changing the heading from the previous example (Output 11), the new format, COLFMT, assigns colors to the values of the Age variable.

Styles in the PROC TABULATE Statement

To customize the data in table cells, use the STYLE= option in the PROC TABULATE statement.

```
ods listing close;
ods html file='test.html';

proc format;
   value colfmt 11-13='yellow' 14-16='orange';
   value datafmt low-<200='light' 200-high='bold';
run;

proc tabulate data=sashelp.class
   style=[font_weight=datafmt.];
   class age / style=[background=red];
   classlev age / style=[font_weight=medium background=colfmt.];
   var height / style=[foreground=blue];
   var weight / style=[foreground=green];
   table age all, height weight;
   keyword sum all / style=[font_style=italic];
run;

ods listing;
ods html close;
```

This program generates the following output:



	Height	Weight
	Sum	*Sum*
Age		
11	108.80	135.50
12	**297.20**	**472.00**
13	184.30	**266.00**
14	**259.60**	**407.50**
15	**262.50**	**469.50**
16	72.00	150.00
All	**1184.40**	**1900.50**

Output 14. Output Generated with the STYLE= Option in the PROC TABULATE Statement

The **`Height`** and **`Weight`** values in the table cells are denoted with either bold or light font, based on the assignments in the second VALUE statement in the FORMAT procedure.

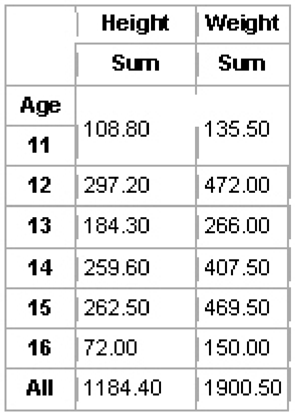
Styles Used to Change Gridlines

Table gridlines are typically controlled by the RULES= and FRAME= style attributes. You can also enhance gridlines with the STYLE= option. Because gridlines are part of the table, you must place the STYLE= option in the TABLE statement.

The following program enhances the gridlines in the previous example (Output 14). The background shading from the table is also removed using PROC TEMPLATE.

```
proc template;
   define style styles.test;
   parent=styles.printer;
   style header from headersandfooters / background=white;
   style rowheader from headersandfooters / background=white;
end;
run;

ods html file='test.html' style=styles.test;

proc tabulate data=sashelp.class;
   class age;
   var height;
   var weight;
   table age all, height weight / style=[rules=none frame=void];
run;

ods html close;

ods rtf file='test.rtf';

proc tabulate data=sashelp.class;
   class age;
   var height;
   var weight;
   table age all, height weight / style=[cellspacing=5 borderwidth=5];
run;

ods rtf close;
```

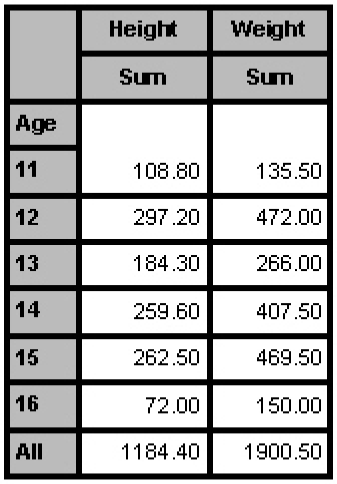
The following table is generated by the first TABULATE procedure in this example code. The gridlines and borders are removed by using the STYLE= option's FRAME= and RULES= attributes in the TABLE statement.



	Height	Weight
	Sum	Sum
Age		
11	108.80	135.50
12	297.20	472.00
13	184.30	266.00
14	259.60	407.50
15	262.50	469.50
16	72.00	150.00
All	1184.40	1900.50

Output 15. Gridlines and Borders Removed With the FRAME= and RULES Style Attributes

The following table is generated by the second TABULATE procedure, and the gridlines and borders are enhanced by using the CELLSPACING= and the BORDERWIDTH= attributes in the TABLE statement.



	Height	Weight
	Sum	Sum
Age		
11	108.80	135.50
12	297.20	472.00
13	184.30	266.00
14	259.60	407.50
15	262.50	469.50
16	72.00	150.00
All	1184.40	1900.50

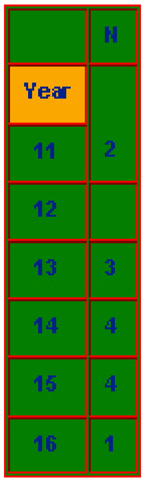
Output 16. Gridlines and Borders Enhanced with the CELLSPACING= and BORDERWIDTH= Style Attributes

Multiple Styles in the TABLE Statement

To control variable labels, table cells, gridlines, spacing, the empty box above the row titles, and so on, you can use style options in the TABLE statement in PROC TABULATE, as shown in the following example:

```
ods listing close;
ods html body='test.html';

proc tabulate data=sashelp.class;
   class age;
   table age=[label='Year' s=[background=orange]]*[s=[font_weight=bold]], n /
         style=[bordercolor=red background=green];
run;

ods listing;
ods html close;
```

The preceding program generates the following output:



	N
Year	
11	2
12	
13	3
14	4
15	4
16	1

Output 17. Output Generated by Using Multiple Styles in the TABLE Statement

In this example:

- A style attribute that follows an equal sign (=) in the TABLE statement applies to the variable heading.
- A style attribute that is crossed using an asterisk (*) in the TABLE statement applies to the table cells.
- Style attributes that follow a forward slash (/) in the TABLE statement typically define spacing and gridline attributes and the text string for the BOX= option.

Inherited Attributes for a Column

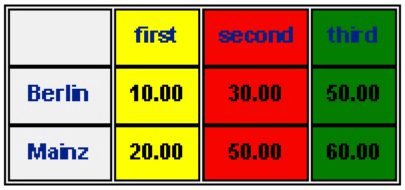
To apply formatting that is inherited throughout a column, you can use the option STYLE=<PARENT>, as shown in the following example. You can also use the alias S=<PARENT>. The *parent style element* causes a data cell to inherit the style element of its parent heading.

The following program illustrates the use of the STYLE=<PARENT> option:

```
data test;
   input name $1-7 category $8-14 val;
   datalines;
Berlin first  10
Mainz  first  20
Berlin second 30
Mainz  second 50
Berlin third  50
Mainz  third  60
;
run;

ods listing close;
ods html body="test.html";
```

```
proc format;
   value $cat 'first'='yellow'
              'second'='red'
              'third'='green';
run;

proc tabulate data=test ;
   class name category;
   classlev category / s=[background=$cat.];
   var val / style=<parent>;
   keyword sum / style=<parent>;
   keylabel sum=' ';
   table name=' ',category=' '*val=' '*
              {style=<parent> {foreground=black}};
run;

ods html close;
ods listing;
```

This program generates the following output:



	first	second	third
Berlin	10.00	30.00	50.00
Mainz	20.00	50.00	60.00

Output 18. Inherited Column Attributes Created by the STYLE=<PARENT> Option

In this example:

- Style attributes are assigned to the top level that forms the columns. In this example, that level is in the CLASSLEV statement.

- Every element that appears below the CLASSLEV variable in the column dimension (CATEGORY *VAL) includes the STYLE=<PARENT> option to specify inheritance. Even if a label is blank, you still need to include STYLE=<PARENT>.

  For the column dimension in the TABLE statement, STYLE=<PARENT> causes all cells in a column to inherit the parent cell's style attributes.

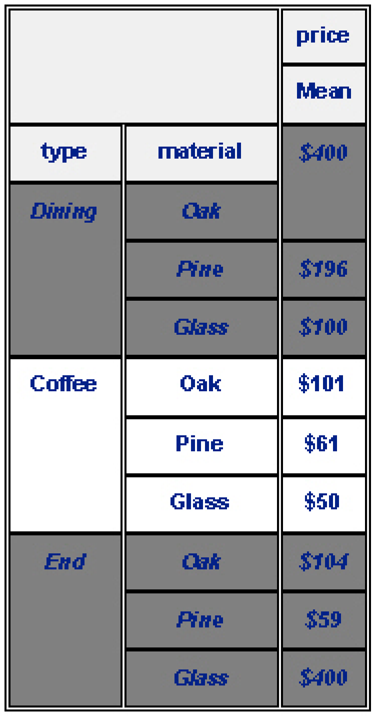
Inherited Attributes for a Row

You can also use STYLE=<PARENT> to apply inherited formatting across rows, as shown in the following example:

```
data one;
   input type material price;
   datalines;
1 2 196
1 3 100
1 1 400
```

```
2 2 61
2 3 50
2 1 101
3 2 59
3 3 400
3 1 104
;
run;

ods escapechar='^';

proc format;
   value type 1='Dining'
              2='Coffee'
              3='End';
   value mat 1='Oak'
             2='Pine'
             3='Glass';
   value typei 1,3='italic'
               2='roman';
   value typecol 1,3='gray'
                 2='white';
run;

ods listing close;
ods html body='test.html';

proc tabulate;
   class type material;
   classlev type / style=[background=typecol. font_style=typei.];
   classlev material / style=<parent>;
   var price;
   format type type. material mat.;
   table type*material*[style=<parent>], price*mean*f=dollar8.;
 run;

ods listing;
ods html close;
```

The preceding program generates the following output:



		price
		Mean
type	material	$400
Dining	*Oak*	
	Pine	*$196*
	Glass	*$100*
Coffee	Oak	$101
	Pine	$61
	Glass	$50
End	*Oak*	*$104*
	Pine	*$59*
	Glass	*$400*

Output 19. Inherited Row Attributes Created by the S=<PARENT> Option

In this example:

- Style attributes are assigned to the left-most column that forms the rows. In this example, that column is **TYPE**, which appears in the CLASSLEV statement.
- Because STYLE=<PARENT> is used in the row dimension of the TABLE statement, the left-most column's style attributes are inherited across the entire row.

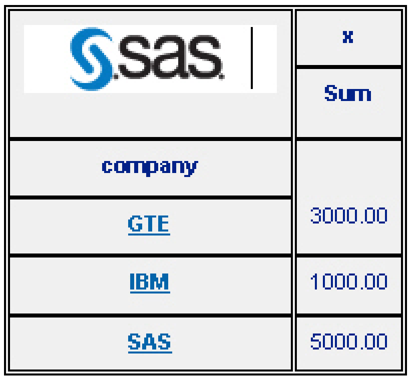
URL Addresses and Images

You can enhance your output even further by adding URL links and images. To do that, you need to use the URL=, PREIMAGE=, and POSTIMAGE= style attributes. The following example illustrates the use of the URL= and the PREIMAGE= attributes:

```
data one;
  input company $ x;
  cards;
IBM 1000
GTE 3000
SAS 5000
;
run;
```

```
proc format;
   value $urlfmt 'SAS'="http://www.sas.com/"
                 'IBM'="http://www.ibm.com/"
                 'GTE'="http://www.gte.com/";
run;

ods listing close;
ods html body='test.html';

proc tabulate;
   class company;
   var x;
   classlev company / style=[url=$urlfmt.];
   table company,x / box=[style=[preimage='c:\tests\saslogo.gif']];
run;

ods html close;
ods listing;
```

This program generates the following output:



	x
	Sum
company	
GTE	3000.00
IBM	1000.00
SAS	5000.00

Output 20. Inherited Row Attributes Created by the S=<PARENT> Option

In this example:

- The URL= style attribute is associated with a format ($URLFMT) that assigns links.
- The PREIMAGE= style attribute adds a specified image to the output.

Conclusion

PROC REPORT and PROC TABULATE enable you to use style elements to customize ODS output. Each output report location requires a specific style element and syntax. The code and output examples in this paper highlight the appropriate style element that is needed for each location in order to produce a customized report output using PROC REPORT and PROC TABULATE.

References

SAS Institute Inc. 2000. SAS Note 2457, “Incorrect values for style elements generate misleading error messages from the REPORT and the TABULATE procedures.” Cary, NC. Available at support.sas.com/kb/2/457.html.

SAS Institute Inc. 2000. SAS Note 3086, “A syntax error might occur when style attributes in the REPORT procedure continue to the next line.” Cary, NC. Available at support.sas.com/kb/3/086.html.

SAS Institute Inc. 2003. SAS Note 10559, “The CALL DEFINE statement in the REPORT procedure requires quotation marks around the column number and the variable name for the column ID.” Cary, NC. Available at support.sas.com/kb/10/559.html.