Breaking New Ground with SAS® 9.2 ODS Layout Enhancements

Daniel O'Connor and Scott Huntley, SAS Institute Inc., Cary, NC

ABSTRACT

Produce stunning brochures, killer Web sites, jaw-dropping dashboards, and breathtaking documents. This innovative enhancement will allow you to realize the full potential of your SAS® investment by leveraging the latest enhancements to ODS Layout. Easily mix graphics, images, text, and tables, and arrange them on a page however you want. Take today's challenges and make them tomorrow's accomplishments.

INTRODUCTION

Today's reporting needs have evolved past the simple generation of reports and graphs, and have matured into highly sophisticated executive summaries, dashboards, elaborate forms, and brochure-quality presentations. These complex reporting requirements necessitate flexible page-management control.

IN THE BEGINNING WERE COLUMNS AND INLINE FORMATTING…AND IT WAS GOOD

Most people think of ODS Layout when you talk about managing your output. There are several ways to control your output using SAS 9.2 production software. Using inline formatting, COLUMNS= and STARTPAGE= syntax, you can do things you never knew you could. This syntax also provides a great foundation for layout. Once you start to use and understand this syntax, using ODS Layout becomes easier.

INLINE FORMATTING

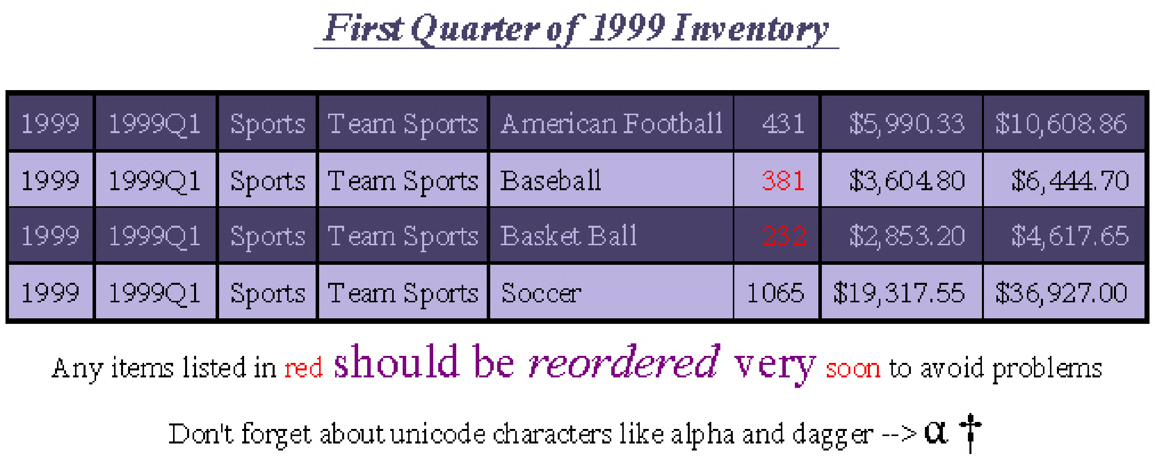
Being flexible and showing off important details in your output is critical. You only get that one chance to grab someone's attention. Using inline formatting is a quick and easy way to modify small portions of your output the way you want it to look.

Using the ODS ESCAPECHAR statement and the inline formatting syntax, you can modify the styles of TITLE statements, TEXT statements, or data cells. Inline formatting syntax enables you to set several style attributes for one text string, without having to reset the previously used style. Let's look at the inline syntax and how it works:

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

The ^ symbol is defined when you issue an ODS ESCAPECHAR statement at the beginning of your SAS program. After the escape character, begin the syntax with the left curly bracket {, then use the STYLE function. Next, you can use a style element like SYSTEMTITLE, HEADERFIXED, or SYSTEMFOOTER. A style element is a collection of style attributes that apply to a feature or aspect of the output. Values are specified for each attribute in a style definition. After the style element, you can set the style attributes. Some common style attributes are color, font, and size. You can set as many style attributes as needed within square brackets. The syntax ends with the text you want to be formatted.

Let's consider an example using our style syntax and see how it changes the appearance of our output. In the appendix of this paper, we have put SAS programs that create every example you will see in this paper. After each figure will be a caption. Use that caption name to find in the appendix the corresponding SAS program. For example, the next example has the caption **Figure 1.1**. Look in the appendix for **Example 1.1** to find the source code used to create that example.



First Quarter of 1999 Inventory

1999	1999Q1	Sports	Team Sports	American Football	431	$5,990.33	$10,608.86
1999	1999Q1	Sports	Team Sports	Baseball	381	$3,604.80	$6,444.70
1999	1999Q1	Sports	Team Sports	Basket Ball	232	$2,853.20	$4,617.65
1999	1999Q1	Sports	Team Sports	Soccer	1065	$19,317.55	$36,927.00

Any items listed in red should be *reordered* very soon to avoid problems

Don't forget about unicode characters like alpha and dagger --> α †

Figure 1.1. Showing the REPORT Procedure Using Inline Formatting

Looking over the example, let's examine the TITLE statement first to see how inline formatting works.

```
title '^{style [textdecoration=underline color=cx494068] First Quarter of 1999
Inventory}';
```

We start with ^{STYLE to invoke using the STYLE function. Then, we use the style attribute TEXTDECORATION to designate that we want to underline the following text. We also use the COLOR attribute to change the text color to purple in the TITLE statement. That is a simple example to show how easy it is to modify your default output.

Now, let's examine the PROC REPORT output and how inline styles are used there. The syntax inside a PROC REPORT call is a little different from the basic inline style syntax. Inside the compute block for the Quantity column, we use a CALL DEFINE statement to set the inline style. You don't need the escape character or the curly brackets. The square brackets contain the style attribute you want to modify.

```
compute quantity;
if quantity2 < 400 then
call define (_col_, "style","style=[color=red]");
endcomp;
```

When the condition on QUANTITY was resolved to TRUE, the style was applied to the column and data attributes.

The second compute block works with the column Year. We modify the row using the BACKGROUND style attribute.

```
compute year;
    bg + 1;
    if mod(bg, 2) = 1 then
       call define(_row_, "style", "style={background=cx494068 color=cxbbb2e0}");
    else
        call define(_row_, "style", "style={background=cxbbb2e0}");
endcomp;
```

You can get a nice interlacing look on your PROC REPORT output using inline styles in this way.

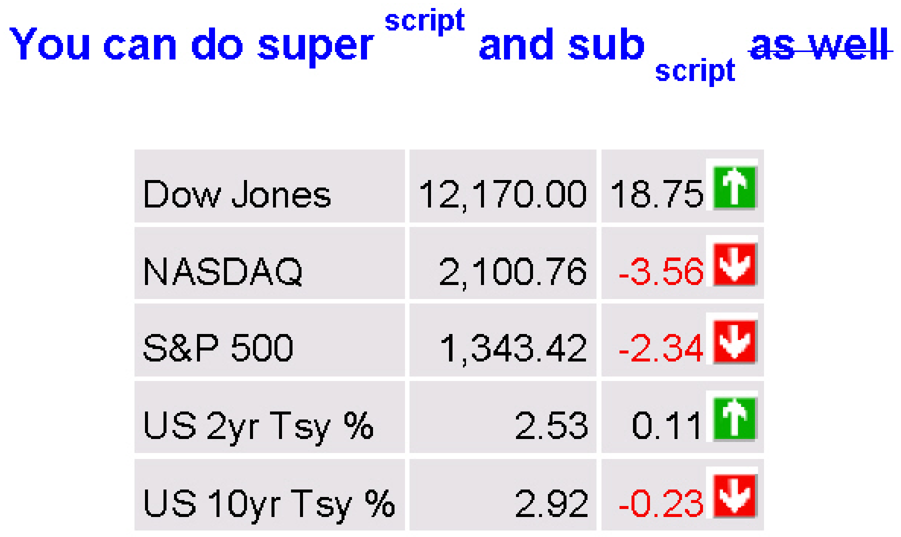
Now, we can look at the two PDF TEXT statements that are below the table. The first one is showing how you can nest styles in one statement. You can start another style within a style if it's inside the curly brackets. We first open a style to set the justification of the TEXT statement to be centered on the page. Inside of that, we start another style using the color red. Then, within that style, we open another style using the color purple and change the font size. Then, we add another style that is using italic as the font style. You'll notice the word "reordered" in the text string has merged the color purple with the bigger font size and the italic. When we close with a right curly bracket on the italic, the style reverts back to just purple with a big font size. Now, when we close the purple style with a right curly bracket, the red style has control. Finally, after we close the red style, the default style takes over.

```
ods pdf text = "^{style[just=c]Any items listed in  ^{style [color=red] red ^{style
[color=purple font_size=16pt] should be ^{style [font_style=italic] reordered }very}
soon} to avoid problems}";
```

The next PDF TEXT statement is showing you how to use the Unicode inline style to designate Unicode symbols in your output. You can set the parameter to be either a four-digit Unicode value, or a value from a predefined list stored as a tagset template.

```
ods pdf text = "^{style[just=c]Don't forget about unicode characters like alpha and
dagger -->   ^{style [font_size=16pt] ^{unicode alpha}  ^{unicode dagger}}}";
```

Here is another example showing the power of inline styles:



You can do super$^{\text{script}}$ and sub$_{\text{script}}$ ~~as well~~

Dow Jones	12,170.00	18.75 ⬆
NASDAQ	2,100.76	-3.56 ⬇
S&P 500	1,343.42	-2.34 ⬇
US 2yr Tsy %	2.53	0.11 ⬆
US 10yr Tsy %	2.92	-0.23 ⬇

Figure 1.2. Another PROC REPORT Using Inline Formatting

```
title 'You can do super ^{super script} and sub ^{sub script} ^{style
[textdecoration=line_through]as well}';
```

Notice the title is using the SUPER and SUB inline functions to show how easy super- or subscripting can be. Also, we added another TEXTDECORATION option called LINE_THROUGH at the end. Looking at the PROC REPORT output, you can see that we added an image to the cells using the POSTIMAGE style attribute. Once again, using a COMPUTE block, you can change the style attributes for the row or column.

```
compute change;
if change2 < 0 then
   call define(_col_, "style", "style=[color=red
   postimage='c:\sgf2009\down_red.png']");
else
   call define(_col_, "style", "style=[postimage='c:\sgf2009\up_green.png']");
endcomp;
```

We turned the cell attribute to red when the value was negative. Also, we displayed one of two images based on the positive or negative value.

Using inline formatting, you can add to your output. Highlight data or titles to make them stand out. Add images to help a user visualize the data better. There are more style attributes that can be added to your output. See the SAS 9.2 documentation for additional information and other inline functions.

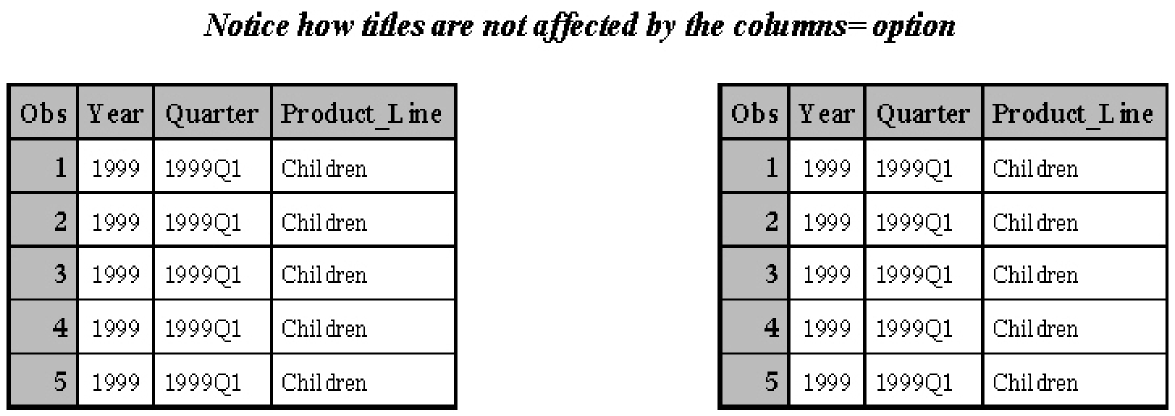
MANAGING MULTIPLE COLUMNS OF OUTPUT

There are ways of controlling your output without using ODS Layout. It is important to understand these basic features first, before diving into the huge ocean of ODS Layout. Controlling your output is a fancy way of saying "putting data on a page where you want the data to be." An easy first step is to use the COLUMNS= option on your ODS statement.

COLUMNS= OPTION

This option is supported in the HTML, RTF, MEASURED RTF, and the PRINTER destinations within ODS. The options will evenly divide a physical page into *n* (user-specified) number of virtual pages. For example, having COLUMNS=2 in the ODS statement will split the physical page into two virtual pages.

Now, let's do an example using COLUMNS=2. There are two things to notice in Figure 1.3. The first is how the page is split into two sections and each table is started inside its own column. We will learn later how to control that, but, for now, just think of each column as a new page for output. The second thing to notice is how the title is not affected by the option. Titles and footnotes always maintain their normal position on the page and will not reside in one column.



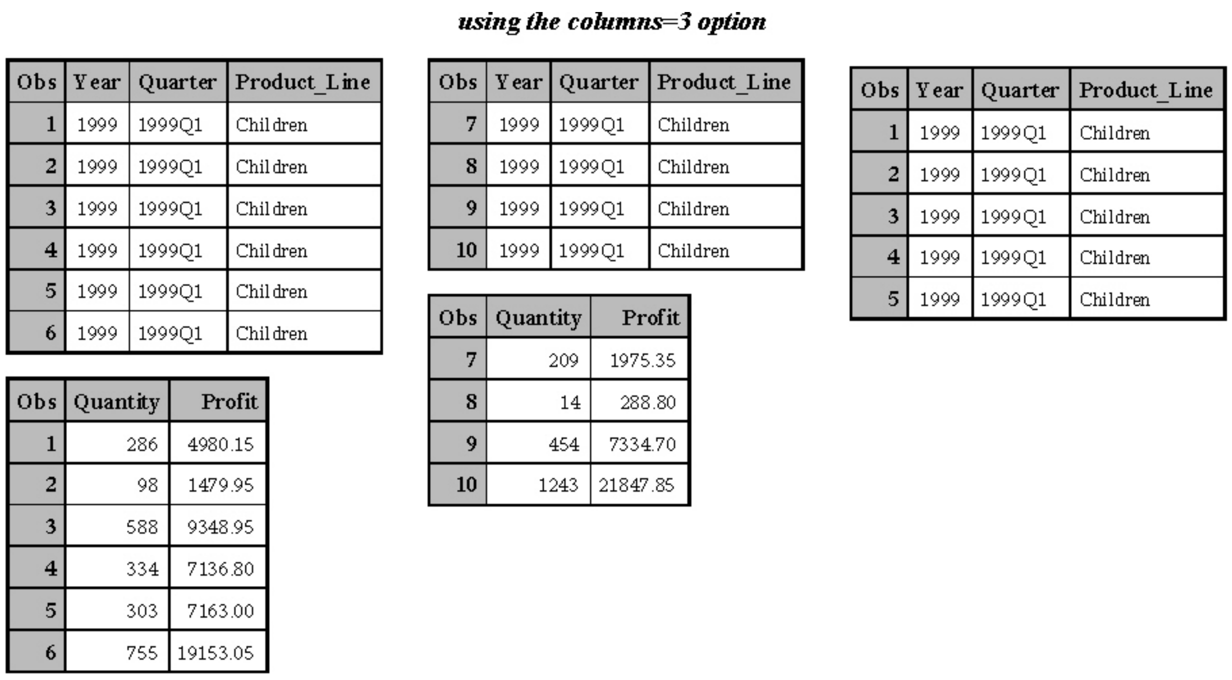
Notice how titles are not affected by the columns= option

Obs	Year	Quarter	Product_Line
1	1999	1999Q1	Children
2	1999	1999Q1	Children
3	1999	1999Q1	Children
4	1999	1999Q1	Children
5	1999	1999Q1	Children

Obs	Year	Quarter	Product_Line
1	1999	1999Q1	Children
2	1999	1999Q1	Children
3	1999	1999Q1	Children
4	1999	1999Q1	Children
5	1999	1999Q1	Children

Figure 1.3. Showing Output Using COLUMNS=2

Now what happens if you choose so many columns for your output that a table cannot fit into the available space provided? Let's say we use COLUMNS=3 in our above example. The table in the left-hand side is too wide for the space. ODS will use the concept we call paneling. Panels are created when the width of a table is wider than a page. The first panel will hold enough of the data for a single observation to fit on one line. The second panel holds the next page width of columns (or data), and so forth. This continues until the entire width of the table is produced. The number of rows for each panel is determined by what will fit on a logical page.



using the columns=3 option

Obs	Year	Quarter	Product_Line
1	1999	1999Q1	Children
2	1999	1999Q1	Children
3	1999	1999Q1	Children
4	1999	1999Q1	Children
5	1999	1999Q1	Children
6	1999	1999Q1	Children

Obs	Quantity	Profit
1	286	4980.15
2	98	1479.95
3	588	9348.95
4	334	7136.80
5	303	7163.00
6	755	19153.05

Obs	Year	Quarter	Product_Line
7	1999	1999Q1	Children
8	1999	1999Q1	Children
9	1999	1999Q1	Children
10	1999	1999Q1	Children

Obs	Quantity	Profit
7	209	1975.35
8	14	288.80
9	454	7334.70
10	1243	21847.85

Obs	Year	Quarter	Product_Line
1	1999	1999Q1	Children
2	1999	1999Q1	Children
3	1999	1999Q1	Children
4	1999	1999Q1	Children
5	1999	1999Q1	Children

Figure 1.4. Showing Output Using COLUMNS=3 and Paneling

When the page is divided equally into three parts, there isn't room for the entire table to be displayed. The first table has six columns (counting the OBS column). Only the first four columns of the table fit in the first virtual page. The logic of paneling looks ahead and realizes it can fit only six rows of data within the first virtual page using paneling. Looking at the second virtual page, the remainder of the rows of the first table is printed. It, again, has to use paneling to get everything to fit within the virtual page.

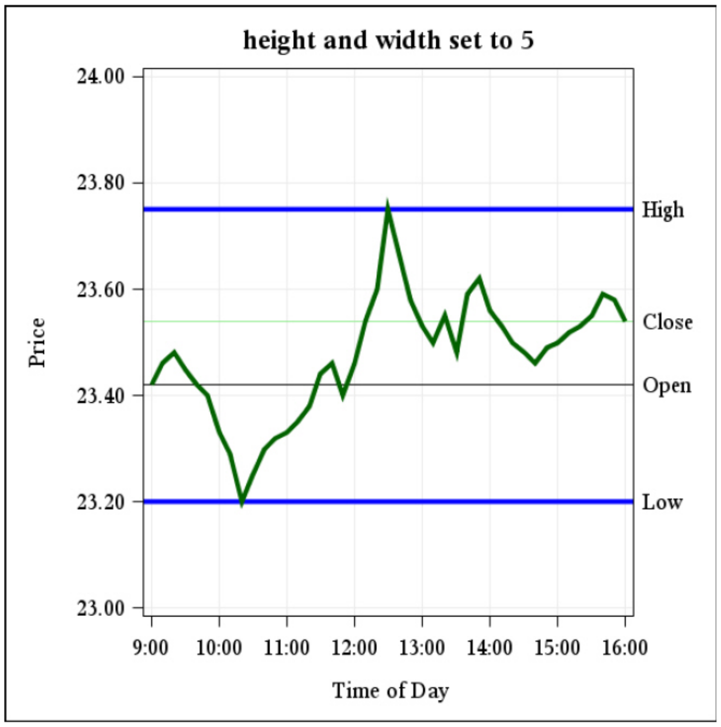
It's important to know how wide your output will be when using the COLUMNS= option because paneling might occur and it might not present the data the way you want it to look. The good news is that paneling will preserve your output. The bad news is that it might have to break it up so it fits on one page.

COLUMNS= OPTION WITH GRAPHS

The COLUMNS= option also affects our graph output. Earlier, we mentioned how each column is basically a virtual page, and data will flow into that page. When you split the page up, the available space might be smaller than a graph needs. The graphs will automatically size to fit within the available space. You can also set the size of an ODS GRAPHICS image by setting the HEIGHT and WIDTH options using the ODS GRAPHICS statement. But, those settings will be overridden by the available space of the page. To show how this works, in the next example we will use the COLUMNS=2 option, but request a wider graph than the page allocated. When you set COLUMNS=2, your physical page (which is normally 8.5 inches wide) is divided by 2. That sets aside approximately 4 inches for each column (or virtual page). Using the ODS GRAPHICS statement, we are requesting our graph to be 5x5 inches.

```
ods graphics on / height=5in width=5in;
```

Looking at the example, the image fits nicely into the space allotted. The vertical size and horizontal size is only 4 inches because of the COLUMNS= statement. The aspect ratio for the graph is maintained when the size changes. So, because the horizontal can go only 4 inches, the vertical will go only 4 inches.



Obs	Year	Quarter	Quantity
1	1999	1999Q1	286
2	1999	1999Q1	98
3	1999	1999Q1	588
4	1999	1999Q1	334
5	1999	1999Q1	303
6	1999	1999Q1	755
7	1999	1999Q1	209
8	1999	1999Q1	14
9	1999	1999Q1	454
10	1999	1999Q1	1243
11	1999	1999Q1	139
12	1999	1999Q1	2938
13	1999	1999Q1	171
14	1999	1999Q1	1554
15	1999	1999Q1	335

Figure 1.5 – Showing Output with COLUMNS=2 Using Graphics

Using COLUMNS= is a quick and easy way to put output objects in different places on a page. Remember, as you divide up the page, the available space for the output will be smaller. You will see resizing of graphs or paneling of tables so that the output will fit into the allocated space.

STARTPAGE=

The STARTPAGE= option is another easy option that controls the placement of output. The syntax is simple. In the ODS statement, type STARTPAGE=YES|NO|NOW to invoke the option.

Let's go through all three options and explain how each of them works. The first option—YES—is the default. Invoking STARTPAGE=YES inserts a new page at the beginning of each procedure. It's basically the normal behavior you see when you don't set the option. When a procedure's output is laid out, a new page will be invoked at the end. The second option is NO. When you invoke STARTPAGE=NO, this means there will be no page insertions before each procedure. A new page will occur only when the current page is filled. The third option is NOW. This option forces the immediate insertion of a new page into the output.

In all of the examples above using the COLUMNS= option, the STARTPAGE= option has not been set, so the default STARTPAGE=YES was being used. Let's look at Example 1.6 using STARTPAGE=NO and see how the output looks different. We'll set COLUMNS=2 and set STARTPAGE=NO. Rather than seeing the second table placed on the second column, we see that it fits right beneath the first table. Our output will continue in this manner until we reset the STARTPAGE= option.

```
ods pdf file='cols2.pdf' columns=2 startpage=no;
```

startpage=no and five obs of each dataset

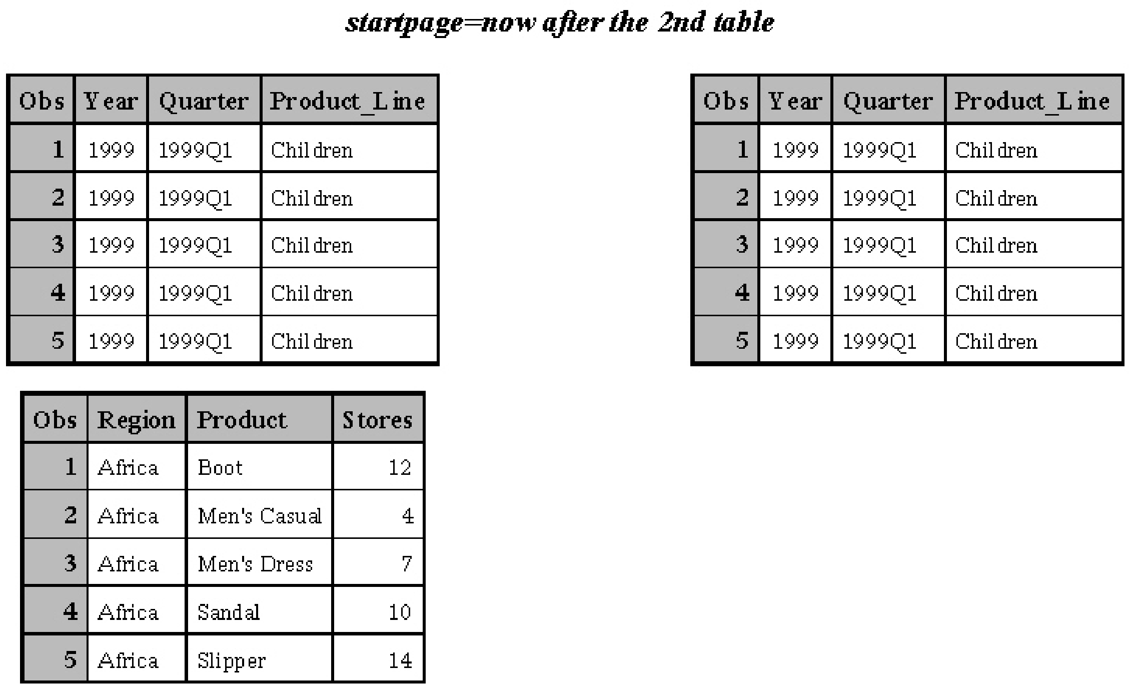
Obs	Year	Quarter	Product_Line
1	1999	1999Q1	Children
2	1999	1999Q1	Children
3	1999	1999Q1	Children
4	1999	1999Q1	Children
5	1999	1999Q1	Children

Obs	Year	Quarter	Product_Line
1	1999	1999Q1	Children
2	1999	1999Q1	Children
3	1999	1999Q1	Children
4	1999	1999Q1	Children
5	1999	1999Q1	Children

Figure 1.6. Showing Output Using STARTPAGE=NO

Now, we need to show how STARTPAGE=NOW works. Using our previous example, if we had included a third table in the output, it would go right after the second table. But, if we issue a STARTPAGE=NOW statement before the third table, it will go to a new page. Because we are doing COLUMNS=2, the next virtual page is the next column.

```
options nodate nonumber;
title 'startpage=now after the 2nd table';
ods pdf file='cols_now.pdf' columns=2 startpage=no;
  proc print data=sashelp.orsales(obs=5);
  var year quarter product_line; run;
  proc print data=sashelp.shoes(obs=5);
  var region product stores;run;
  ods pdf startpage=now;
  proc print data=sashelp.orsales(obs=5);
  var year quarter product_line; run;
ods pdf close;
```



startpage=now after the 2nd table

Obs	Year	Quarter	Product_Line
1	1999	1999Q1	Children
2	1999	1999Q1	Children
3	1999	1999Q1	Children
4	1999	1999Q1	Children
5	1999	1999Q1	Children

Obs	Year	Quarter	Product_Line
1	1999	1999Q1	Children
2	1999	1999Q1	Children
3	1999	1999Q1	Children
4	1999	1999Q1	Children
5	1999	1999Q1	Children

Obs	Region	Product	Stores
1	Africa	Boot	12
2	Africa	Men's Casual	4
3	Africa	Men's Dress	7
4	Africa	Sandal	10
5	Africa	Slipper	14

Figure 1.7. Showing Output Using STARTPAGE=NOW

As you can see, combining the power of COLUMNS= and STARTPAGE= gives you basic control of output objects. Understanding these basic principles helps in understanding ODS Layout better. It serves two purposes. First, you realize you have the power to control your output without ODS Layout. Second, you are gaining an understanding of spacing and how output objects are placed.

SUMMARY

So, let's weigh the advantages versus disadvantages of using the combined COLUMNS= and STARTPAGE= approach. The advantages are that it's simple to do and simple to understand. You can almost picture how your output will look as you code it. If you want a page break, just use STARTPAGE=NOW to get one. If you want three columns of output, just set COLUMNS=3 in the ODS statement. Another advantage is that it's production quality and available to you right now. Everything mentioned above is production in SAS 9.2: inline formatting, COLUMNS=, and STARTPAGE=. One disadvantage is that it's not very powerful. You don't have full control of the spacing of the columns. Let's say you want the first column to be 3 inches and the second column to be 5 inches. You need Layout regions for that. You also don't have the ability to set an output object at an exact X, Y location on the page. You would need absolute Layout for that task.

Moving forward, we'll learn about ODS Layout and the power and potential it has in store for you.

THE TRILOGY CONTINUES…EPISODE II: A NEW HOPE

Despite the fact that the COLUMNS= option is simple and intuitive, it often is not flexible enough to address more demanding displays. When the placement of output on a page necessitates a more detailed level of control, you will find ODS Layout to be your solution. Although it is preproduction in SAS 9.2, you will find many enhanced features that are designed to provide you with unlimited control of your page. ODS Layout has two distinct forms, absolute and gridded, which both have their unique strengths and weaknesses. The most important decision when using ODS Layout is in selecting the type that will best suit your report design. ODS Layout follows the traditional ODS paradigm where statements are wrapped (sandwiched) around your PROC code with a definitive starting and ending location. ODS Layout is designed to allow nested layouts (containers) to provide endless customization. You are not limited to a single ODS Layout type. A gridded layout can contain absolute layouts, and vice versa. This is where adequate design work will greatly benefit your coding efforts.

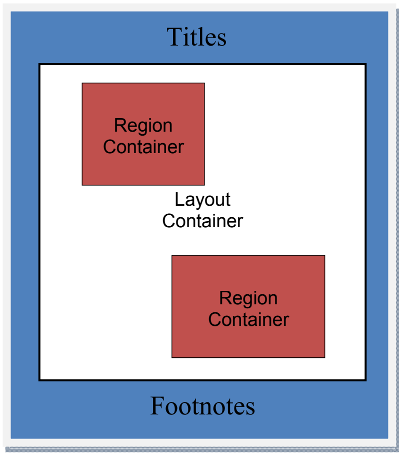
NOT ALL OUTPUT DESTINATIONS ARE CREATED EQUAL

There are a variety of limitations imposed on an ODS Layout by the destination itself. For example, the PDF (or PRINTER) destinations are limited to the physical page size of the piece of paper that we are writing to, whereas in the case of HTML, there really is no concept of a physical page size or an X or Y position because the browser is managing the placement of the output in relation to the size and placement of the browser window. Because of some of these external restrictions that are being imposed on our report design, we might get varying results when using different output destinations. However, you will find that we will always make an effort to provide the most intuitive defaults when dealing with these limitations, and we will make every effort to produce reasonably similar-looking output.

ODS LAYOUT TERMINOLOGY, CONCEPTS, AND THINGS TO CONSIDER

layout container is an area that contains a collection of regions. Layouts can ONLY contain regions. Layouts can have a fixed size (like WIDTH=3in and HEIGHT=4in), or can be dynamically sized to accommodate the child regions.

region container is an area that contains output (like text, tables, graphs) or nested layout containers. Regions can also have a fixed size, or can be dynamically sized to accommodate the collection of child output.



dimension unit is a nonnegative number, optionally followed by one of the following units of measure. A number without a unit of measure will be interpreted as pixels. It is not recommended that you size things in pixels because of adverse dependencies on resolution that can differ between destinations.

cm	centimeters
in	inches
mm	millimeters
pt	a printer's point

titles and footnotes are placed outside the outermost layout container. Title and footnote processing is always done before any output is produced on the physical page. This is often referred to as page initialization time.

proctitles are placed inside the region adjacent to the PROC output.

PAGE CONSIDERATIONS

Some output destinations are designed to be printed on a physical piece of paper and, therefore, have additional restrictions that have an effect on ODS Layout. The following SAS global options define the physical page attributes and must be specified before the output destination initialization.

PAPERSIZE The PAPERSIZE= option can be specified by using a predefined paper size (like LETTER, LEGAL, US POSTCARD), or explicitly by specifying the paper width and height (like ("4in", "3in")). See SAS system options documentation for additional help with the PAPERSIZE= option. The default is LETTER.

ORIENTATION The ORIENTATION= option specifies the paper orientation (like PORTRAIT, LANDSCAPE, REVERSEPORTRAIT, REVERSELANDSCAPE). See SAS system options documentation for additional help with the ORIENTATION= option. The default is PORTRAIT.

TOPMARGIN
BOTTOMMARGIN
LEFTMARGIN
RIGHTMARGIN Specifies the size of the margin on each area of the physical page. These options accept a nonnegative number, optionally followed by one of the following units of measure. A number without a unit of measure will be interpreted as inches. The default margins are derived from the default printer driver being used. See the SAS system options documentation for additional help with the PRINTERPATH= option to alter the default printer driver.

cm	centimeters
in	inches

DPI	Defines the resolution of the page and is currently only available with the ODS PRINTER destinations. All output (table, text, and graphics) will be rendered at this resolution on the page. Increasing the DPI will have a dramatic effect on the physical size of the file that gets created. This option accepts a nonnegative number (integer). The default is 150.

ABSOLUTE LAYOUT

Absolute layout enables you to specify the exact location on the page to place a layout and region container. This approach is very precise, but somewhat error-prone. The sole responsibility for correctly placing each container falls on the programmer. Often this creates an overly complicated program because each container needs to be explicitly placed to ensure that there is no overlap and that programmatic alignment is retained. This creates a code-management nightmare because if one container's position is altered, you might need to manually alter all of the other containers to maintain your report's integrity. Absolute layout is also restricted to a single page. If the output is too large to fit in the fixed-size container, the output will be discarded and you will receive a blank region and a warning in your log. With all that doom and gloom out of the way, absolute layout does have its merits. Absolute layout is perfectly suitable for very static types of output, placing output in a specific location on a preprinted form, creating cover pages, or precisely placing output in a nested region container.

Valid Destinations: PRINTER destinations only (like PDF, PS, PCL)

Syntax:

ods layout start (absolute)
    [x=<dimension *unit*>]
    [y=<dimension *unit*>]
    [width=<dimension *unit*>]
    [height=<dimension *unit*>]
    ;

X	Horizontal position of the layout, which will extend to the right of this position for WIDTH. If omitted, it defaults to 0.
Y	Vertical position of the layout, which will extend down from this position for HEIGHT. If omitted, it defaults to the current vertical position on the page.
width	Horizontal width of the layout. If omitted, it defaults to the maximum horizontal space needed to display all regions.
height	Vertical height of the layout. If omitted, it defaults to the maximum vertical space needed to display all regions.

ods region
    [x=<dimension *unit*>]
    [y=<dimension *unit*>]
    [width=<dimension *unit*>]
    [height=<dimension *unit*>]
    ;

X	Horizontal position of the region within the layout, which will extend to the right of this position for WIDTH. If omitted, it defaults to 0.
Y	Vertical position of the region within the layout, which will extend down from this position for HEIGHT. If omitted, it defaults to the current vertical position on the page.
width	Horizontal width of the region, which is restricted by the layout's dimensions. If omitted, it defaults to the maximum horizontal space needed to display the output contained in the region. If the region extends beyond the layout's horizontal dimension, then the X position will be shifted to the left to show as much of the output as possible.
height	Vertical height of the region, which is restricted by the layout's dimensions. If omitted, it defaults to the maximum vertical space needed to display the output contained in the region. If the region extends beyond the layout’s vertical dimension, then the Y position will be shifted upward to show as much of the output as possible.

ods layout end;

USING ABSOLUTE LAYOUT TO CREATE A COVER PAGE

The purpose of a cover page is to provide a simple, but friendly opening to the topic you plan on addressing in your document. It helps to set the goals, introduce the content, and set the expectations for what the audience is about to read. The content is almost exclusively static, informational, and will very seldom contain any type of report or graph. In this example, we extensively use inline formatting, which was introduced earlier, and the ODS TEXT statement. If you are unfamiliar with the ODS TEXT statement, the purpose of this statement is to simply add some narrative text to your document. The active output destinations will display the text in each respective document.

The ODS TEXT syntax is extremely simple:

ods text="<narrative text>";

The entire program is contained in the appendix in Example 2.1. One of the most confusing issues with ODS Layout is that both the absolute layout and the gridded layout start with the same ODS LAYOUT START statement. The single, most common question that we receive is: "How do I know whether I am using absolute or gridded layout?" The answer is: "If you use a ROW= or a COLUMN= option on your ODS LAYOUT START statement, you are using gridded layout. Otherwise, you are using absolute layout." We intend on changing this behavior to be explicit when ODS Layout becomes production, but it will be backward-compatible so that existing programs will continue to run.

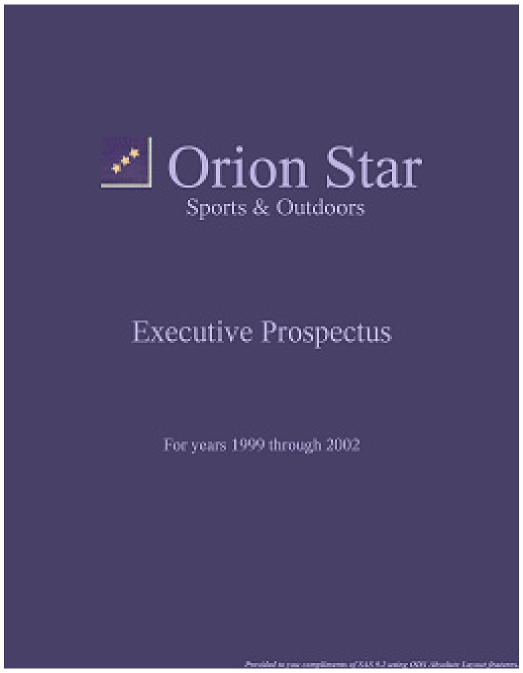

Orion Star
Sports & Outdoors
Executive Prospectus
For years 1999 through 2002
Provided to you compliments of SAS 9.2 using ODS Absolute Layout features

Figure 2.1. Creating a Cover Page with Absolute Layout

Let's just focus on the absolute layout code exclusively. Once the absolute layout has been created, we create region containers at specific Y (vertical) positions on the page. Notice in the following example that we do not specify the X, WIDTH, or HEIGHT. ODS will calculate these values for us. The reason that we do not specify these is because we have no idea how much space the text will need using a 72-point font and the image. Without knowing the WIDTH of the text, we cannot possibly make an intelligent guess as to where the X (horizontal) position should start to center the content. However, we can provide inline formatting instructions to center the text once the dimensions have been calculated. Notice the second region actually has both an X and Y specified because we did not want the text to be centered on the page, we wanted it beneath the "Orion Star" text. This took several tries to get the desired position.

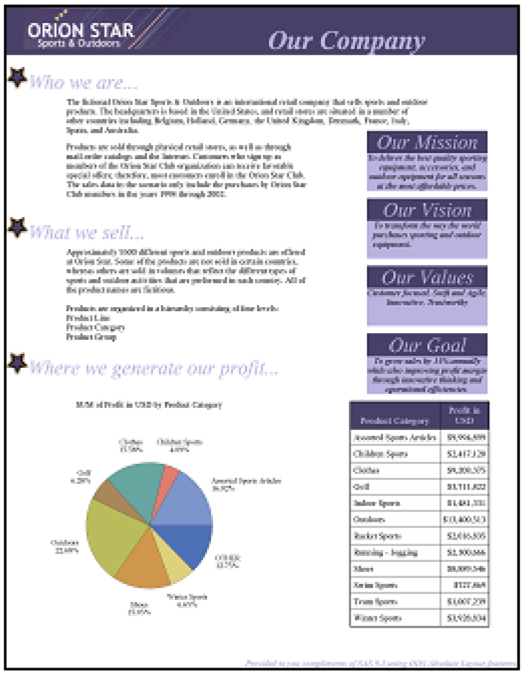
```
ods layout start;
ods region y=2in ;
  ods text="~{style [preimage='c:\Public\SGF 2009\Orion Star
Info\or_internal\orionLogoMedium.jpg' color=cxbbb2e0 just=center font_size=72pt] Orion Star}";

ods region y=3in x=3in;
  ods text="~{style [color=cxbbb2e0 font_size=28pt]Sports & Outdoors}";
ods region y=5in;
  ods text="~{style [just=Center font_size=36pt color=cxbbb2e0]Executive Prospectus}";

ods region y=7in;
  ods text="~{style [just=Center font_size=20pt color=cxbbb2e0]For years 1999 through 2002}";
ods layout end;
```

USING ABSOLUTE LAYOUT TO PRODUCE STATIC OUTPUT

Absolute layouts are not restricted to producing text; they can contain tables and graphs as well. In the following example, we will explore combining static text with tabular and graphical output. The narrative text on the left-hand side of our page was produced in a similar manner to our previous example, which uses inline formatting and the ODS TEXT statement, so we will not belabor that approach. The entire program is contained in the appendix in Example 2.2. We will start by looking at the code that produces the four boxes (for example, "Our Mission… ") on the right-hand area of the page. Layout containers and region containers are both fully integrated with ODS styles. Each has a respective style element in the default styles for an output destination, called LAYOUTCONTAINER and LAYOUTREGION. First of all, what is a style element? A style element is a collection of style attributes that apply to a particular area of our output. In this case, our LAYOUTCONTAINER style element will define the style attributes that will be applied to our layout container, and the LAYOUTREGION style element will define the style attributes that will be applied to our region container.


ORION STAR Sports & Outdoors

Our Company

Who we are...

The fictional Orion Star Sports & Outdoors is an international retail company that sells sports and outdoor products. The headquarters is based in the United States, and retail stores are situated in a number of other countries including Belgium, Holland, Germany, the United Kingdom, Denmark, France, Italy, Spain, and Australia.

Products are sold through physical retail stores, as well as through mail-order catalogs, and the Internet. Customers who sign up as members of the Orion Star Club organization can receive favorable special offers; therefore, most customers enroll in the Orion Star Club. The sales data in the scenario only include the purchases by Orion Star Club members in the years 1998 through 2002.

What we sell...

Approximately 5500 different sports and outdoors products are offered at Orion Star. Some of the products are not sold in certain countries, whereas others are sold in volumes that reflect the different types of sports and outdoor activities that are performed in each country. All of the product names are fictitious.

Products are organized in a hierarchy consisting of four levels:
Product Line
Product Category
Product Group

Where we generate our profit...

Our Mission
To deliver the best quality sporting equipment, accessories, and outdoor equipment for all seasons at the most affordable prices.

Our Vision
To transform the way the world purchases sporting and outdoor equipment.

Our Values
Customer focused, Swift and Agile, Innovative, Trustworthy

Our Goal
To grow sales by 15% annually while also improving profit margins through innovative thinking and operational efficiencies.

SUM of Profit in USD by Product Category

Product Category	Profit in USD
Assorted Sports Articles	$9,994,899
Children Sports	$2,417,129
Clothes	$9,306,575
Golf	$3,111,822
Indoor Sports	$1,481,331
Outdoors	$13,400,513
Racket Sports	$2,016,835
Running - Jogging	$2,300,666
Shoes	$8,809,346
Swim Sports	$727,869
Team Sports	$1,607,239
Winter Sports	$3,926,834

Provided to you compliments of SAS 9.1 using ODS Absolute Layout features

Figure 2.2. Absolute Layout to Arrange Text, Graphics, and Tables

In this portion of the example, we have created a new ODS style that inherits all the characteristics of the default ODS PRINTER style, and simply redefined the background color attribute for the LAYOUTREGION element to be a light-purple color (cxbbb2e0).

```
proc template;
define style Styles.OrionCalloutBlock;
 parent =Styles.Printer;
 style  LayoutRegion/
       background=cxbbb2e0;
 end;
run;
```

Later in the program, we change the current ODS style for the PDF destination from the default STYLES.PRINTER to be STYLES.ORIONCALLOUTBLOCK, so that when a new region container is created, the new attributes will be applied. When we are done with creating the boxes, we can change the style back to the default and continue.

```
ods pdf style=Styles.OrionCalloutBlock;
ods region x=6in y=1.0625in width=2in height=1in;
  ods text="~{style [background=cx494068 color=cxbbb2e0 font_size=24pt just=center
font_style=italic width=100pct] Our Mission }";

ods pdf style=Styles.Printer;
```

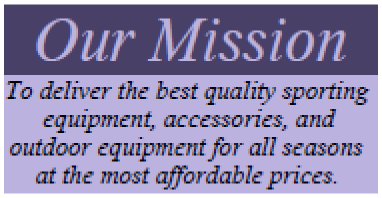

Our Mission
To deliver the best quality sporting equipment, accessories, and outdoor equipment for all seasons at the most affordable prices.

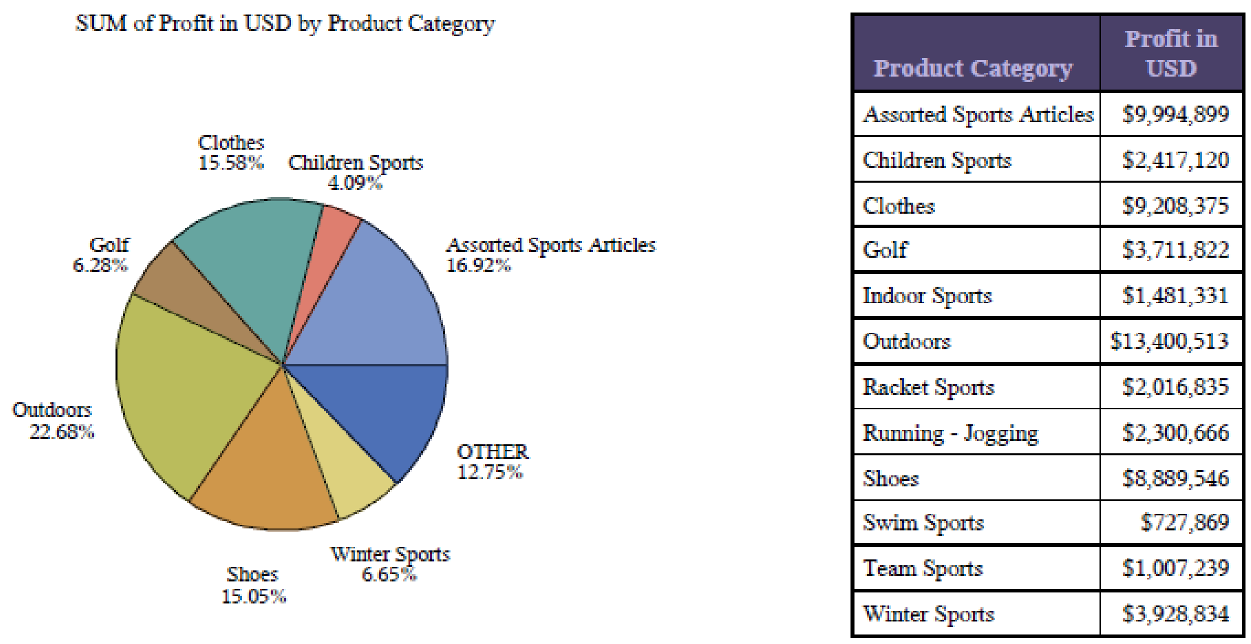
Figure 2.3. Altering an ODS Style to Control Layout Characteristics

Integrating graphical and tabular output is just as easy as text. However, there are some interactions that you need to be aware of that are similar to those when using the ODS COLUMNS= statement. Graphics output will attempt to fill a ODS region unless restricted by the HSIZE or VSIZE GOPTIONS. In the example, we have specified a HEIGHT and WIDTH on the region, which will determine the GCHART size. If the program had specified `goptions hsize=3in vsize=3in`, the GCHART would honor the GOPTIONS statement because it does not exceed the regions specifications. However, if the program had specified `goptions hsize=5in vsize=5in`, which exceeds the region's size specifications, the graph would be limited to the region's dimensions.

```
ods region y=5.5in width=4.75in height=4in;
goptions device=png;
proc gchart data=sashelp.orsales;
pie product_category /
         sumvar=profit
       value=none
       percent=outside slice=outside;
run;
```

Tabular output can be more difficult, however, because of the possibility that our report can have a large number of variables or observations. If there is insufficient real estate to display all of the report, the region will remain blank, and a log message will be displayed indicating that we could not display the requested information in the space provided. We take a much more aggressive approach because we consider not showing the entire report a data integrity issue. In the following example, we are showing PROFITS by a fixed number of product categories, so we can make an assumption that our tabular output will not change and our region dimensions will suffice.

```
ods region y=5.5in x=4.625in width=4.5in height=5in;
proc report nowd data=sashelp.orsales style(header)={background=cx494068
color=cxbbb2e0};
 columns product_category profit;
 define product_category / group;
 define profit /analysis sum format=dollar14.;
run;
quit;
```



Product Category	Profit in USD
Assorted Sports Articles	$9,994,899
Children Sports	$2,417,120
Clothes	$9,208,375
Golf	$3,711,822
Indoor Sports	$1,481,331
Outdoors	$13,400,513
Racket Sports	$2,016,835
Running - Jogging	$2,300,666
Shoes	$8,889,546
Swim Sports	$727,869
Team Sports	$1,007,239
Winter Sports	$3,928,834

Figure 2.4. Integrating Graphics and Tabular Output in an ODS Layout

GRIDDED LAYOUT

Gridded layout enables you to arrange output in a two-dimensional relative grid structure (like a spreadsheet or a piece of graph paper). It is the preferred approach to managing output on the page. Gridded layout is a much simpler mechanism for arranging output. Yet, at the same time, it is a much more powerful alternative than absolute layout. Gridded layout addresses all of the limitations that absolute layout has to offer, like enforcing automatic alignment of respective grid cells, the ability to have a layout continue onto the next page, the ability to dynamically compute the size of a grid cell, and the ease in which you can maintain the integrity of the report (or program).

Valid Destinations: HTML and PRINTER destinations

Syntax:

ods layout start (gridded)
    [x=<dimension *unit*>]
    [y=<dimension *unit*>]
    [width=<dimension *unit*>]
    [height=<dimension *unit*>]

    [rows=<int>]
    [row_heights=(<dimension unit>, … <n dimension units>)]
    [row_gutter=<dimension unit>]

    [columns=<int>]
    [column_widths=(<dimension unit>, … <n dimension units>)]
    [column_gutter=<dimension unit>]
    ;

X	Horizontal position of the layout, which will extend to the right of this position for WIDTH. If omitted, it defaults to 0.
Y	Vertical position of the layout, which will extend down from this position for HEIGHT. If omitted, it defaults to the current vertical position on the page.
width	Horizontal width of the layout. If omitted, it defaults to the maximum horizontal space needed to display all regions.

height	Vertical height of the layout. If omitted, it defaults to the maximum vertical space needed to display all regions.
columns	Fixed number of columns in the gridded layout. If omitted, it defaults to one column.
column_widths	Width of each column specified. The number of horizontal sizes must match the number of columns specified or a warning will be produced and the option will be ignored.
column_gutter	Horizontal space between each column. If omitted, it defaults to the CELL_SPACING style attribute.
rows	Fixed number of rows in the gridded layout. If omitted, it defaults to the maximum number of regions created in the vertical direction. This option should be used very sparingly.
row_heights	Height of each row specified. The number of vertical sizes must match the number of rows specified or a warning will be produced and the option will be ignored.
row_gutter	Vertical space between each row. If omitted, it defaults to the CELL_SPACING style attribute.

ods region
[width=<dimension *unit*>]
[height=<dimension *unit*>]

[row=<int>]
[row_span=<int>]

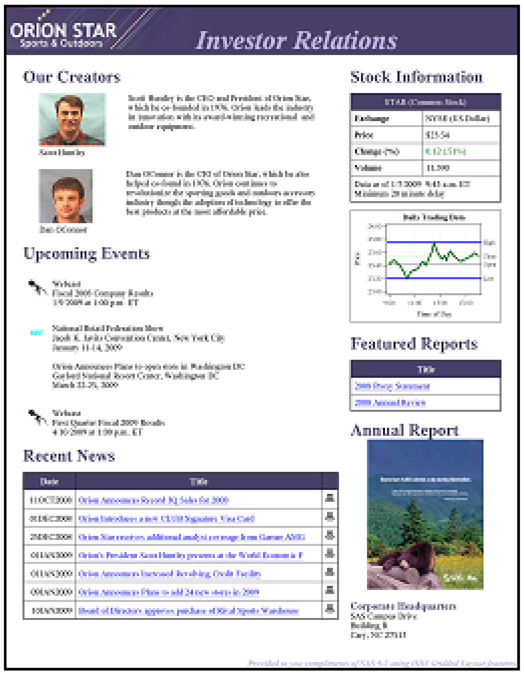
[column=<int>]
[column_span=<int>]
;

width	Horizontal width of the region, which is restricted by the layout's dimensions. If omitted, it defaults to the maximum horizontal space needed to display the output contained in the region. The sum of all region widths cannot exceed the layout's horizontal dimension.
height	Vertical height of the region, which is restricted by the layout's dimensions. If omitted, it defaults to the maximum vertical space needed to display the output contained in the region. The sum of all region heights cannot exceed the layout's vertical dimension. This option should be used very sparingly.
column	Enables you to specify the current grid column position in the gridded layout. This is generally useful when you want to skip regions in the gridded layout. This option should be used very sparingly. The gridded layout automatically tracks the current grid column position and will be incremented for every REGION statement. Once you have skipped a grid column, you cannot go back to it. Random access of grid rows and columns is not supported.
column_span	Enables you to specify the number of grid columns that the region will occupy. It simply enables you to combine adjacent grid columns in gridded layout. The default is 1.
row	Enables you to specify the current grid row position in the gridded layout. This is generally useful when you want to skip regions in the gridded layout. This option should be used very sparingly. The gridded layout automatically tracks the current row position and will be incremented for every REGION statement. Once you have skipped a row, you cannot go back to it. Random access to row and columns is not supported.
row_span	Enables you to specify the number of grid rows that the region will occupy. It simply enables you to combine adjacent grid rows in gridded layout. The default is 1.

ods layout end;

USING GRIDDED LAYOUT TO DISPLAY A PAGE OF DYNAMIC REPORTS

Gridded layout is much more suitable for traditional SAS reporting's dynamic nature. Unlike our previous examples, which contained mostly narrative text and a fixed size table, this example focuses on multiple data sources that are subject to change almost every day. We could have taken a traditional report writing approach and serialized each report onto multiple pages, but that would not have been the most effective way to share our Investor Relations information. Today's reporting needs have matured beyond simple reports to include more sophisticated arrangements of information. This page has six unique categories (Our Creators, Upcoming Events, Recent News, Stock Information, Featured Reports, and Annual Report) of information, all emanating from unique data sources. We could have attempted to create this page using absolute layout, but that would require us to modify our X, Y, HEIGHT, and WIDTH arguments every time any of the data sources changed, which is not desirable. These demands are perfectly suited for gridded layout. The entire program is contained in the appendix in Example 2.5. This example uses a combination of the following features—inline formatting with ODS TEXT=, PROC REPORT, PROC SGPLOT, and the ODS report writing interface. The ODS report writing interface is a new preproduction feature that can produce custom data-centric reports in an easy-to-use, object-oriented manner. For more information about the ODS report writing interface, see the SAS Global Forum 2009 paper *The Power to Show: Ad Hoc Reporting, Custom Invoices, and Form Letters* by Daniel O'Connor, or go to http://support.sas.com/rnd/base/ods/index.html.



Investor Relations

Our Creators

Scott Huntley is the CEO and President of Orion Star, which he co-founded in 1976. Orion leads the industry in innovation with its award-winning recreational and outdoor equipment.

Scott Huntley

Dan O'Connor is the CIO of Orion Star, which he also helped co-found in 1976. Orion continues to revolutionize the sporting goods and outdoors accessory industry through the adoption of technology to offer the best products at the most affordable price.

Dan O'Connor

Upcoming Events

Webcast
Fiscal 2008 Company Results
1/9/2009 at 1:00 p.m. ET

National Retail Federation Show
Jacob K. Javits Convention Center, New York City
January 11-14, 2009

Orion Announces Plans to open store in Washington DC
Gaylord National Resort Center, Washington DC
March 22-25, 2009

Webcast
First Quarter Fiscal 2009 Results
4/10/2009 at 1:00 p.m. ET

Recent News

Date	Title	
11OCT2008	Orion Announces Record 3Q Sales for 2008	
01DEC2008	Orion Introduces a new CLUB Signature Visa Card	
29DEC2008	Orion Star receives additional analyst coverage from Garner AMG	
01JAN2009	Orion's President Scott Huntley presents at the World Economic F	
01JAN2009	Orion Announces Increased Revolving Credit Facility	
09JAN2009	Orion Announces Plans to add 24 new stores in 2009	
10JAN2009	Board of Directors approves purchase of Rival Sports Warehouse	

Stock Information

STAR (Common Stock)	
Exchange	NYSE (US Dollar)
Price	$23.54
Change (%)	0.12 (.51%)
Volume	11,500
Data as of 1/7/2009 9:41 a.m. ET Minimum 20 minute delay	

Featured Reports

Title
2009 Proxy Statement
2008 Annual Review

Annual Report

Corporate Headquarters
SAS Campus Drive
Building 8
Cary, NC 27513

Provided to you compliments of SAS 9.2 using ODS Gridded Layout features

Figure 2.5. Gridded Layout to Display a Page of Dynamic Reports

Notice that all six categories are displayed in two independently sized columns, with each category aligning vertically with each other in their respective columns. Thus far, we have *only* arranged PROC output in its own respective region containers. However, in this example, we group multiple reports into a single region. So, we have established that this page is going to have two columns in the gridded layout. Let's first compare this to the traditional COLUMNS= option approach. When using a traditional COLUMNS= option approach, the physical page was evenly divided into *n* virtual pages (columns), and we had no control over the width or height of the virtual pages that were created. Contrast that with gridded layout, where each gridded layout has the ability to define *n* virtual pages (columns), as well as the ability to control the starting position, width, and height, as long they don't exceed the physical limitation of the page.

So let's look at some actual ODS LAYOUT code that was used to create this example. The ODS LAYOUT START statement specifies two columns (which means we are doing gridded layout and not absolute), and optionally specifies the COLUMN_WIDTHS argument, which is an easy way to specify a fixed width for each respective region container (column). There is no X, Y, or HEIGHT specifications at all. Gridded layout calculates the HEIGHT for you, while maintaining the current X, Y position for each respective piece of output on the page. As the data in the categories change, so does our HEIGHT of each respective piece of output region, and the respective X, Y positions for each piece of output contained within the layout. If the overall height of the layout or region becomes larger than the available height of the page, it will continue onto the next page after repeating the title and footnote processing. You start a region container by using the ODS REGION statement, and all of your subsequent output will be positioned in this container until you create a new region container. When the next REGION statement is executed, it looks for the next available column in our layout. In this test case, a second REGION statement would be positioned at row 1:column 2 of our gridded layout. Gridded layouts always populate in a column-wise fashion, or from left to right and top to bottom.

```
ods escapechar="~";
title "~{style [preimage='c:\Public\SGF 2009\Orion Star
Info\or_internal\orionstarHeader.jpg' width=100pct background=cx494068 color=cxbbb2e0
font_size=32pt just=left] Investor Relations}";
footnote "~{style [font_size=10pt just=right color=cxbbb2e0]Provided to you
compliments of SAS 9.2 using ODS Gridded Layout features.}";

ods layout start columns=2 column_widths=(5.25in 2.5in);
ods region;

< ODS text="… Our Creators" >
< ODS Report Writing Interface >

< ODS text="… Upcoming Events" >
< ODS Report Writing Interface >

< ODS text="… Recent News" >
< PROC Report>

ods region;

< ODS text="… Stock Information" >
< ODS Report Writing Interface >
< PROC Sgplot >

< ODS text="… Featured Reports" >
< PROC Report >
< ODS text="… Annual Report Information and Corporate Address" >

ods layout end;
```

However, an alternative would have been to specify the width individually on each REGION statement.

```
ods escapechar="~";
title "~{style [preimage='c:\Public\SGF 2009\Orion Star
Info\or_internal\orionstarHeader.jpg' width=100pct background=cx494068 color=cxbbb2e0
font_size=32pt just=left] Investor Relations}";
footnote "~{style [font_size=10pt just=right color=cxbbb2e0]Provided to you
compliments of SAS 9.2 using ODS Gridded Layout features.}";

ods layout start columns=2;
ods region width=5.25in;

< ODS text="… Our Creators" >
< ODS Report Writing Interface >

< ODS text="… Upcoming Events" >
< ODS Report Writing Interface >

< ODS text="… Recent News" >
< PROC Report>

ods region width=2.5in;

< ODS text="… Stock Information" >
< ODS Report Writing Interface >
< PROC Sgplot >

< ODS text="… Featured Reports" >
< PROC Report >
< ODS text="… Annual Report Information and Corporate Address" >

ods layout end;
```

This particular example specified a fixed width for each region container (column); however, that is not a requirement. When the width has not been specified, the region container will dynamically measure all output contained in each region and will adjust the region width to be as wide as the widest piece of output that it contains, as long as there is sufficient physical space on the page. So far, gridded layout requires less programming, while at the same time, provides significantly more control over your page. Gridded layout will calculate any of the unspecified positional options. While gridded layout seems like the most powerful approach to any reporting need, it does have some limitations because of its completely dynamic nature. Let's consider a hypothetical gridded layout that has four columns that is completely dynamic (no WIDTH or HEIGHT options at all). Let's assume that the first column (region container) contains output that consumes 7.5 inches of our available 8-inch page. We are now left with .5 inches to display the remaining three columns. In this scenario, you will probably want to make additional restrictions on your report design to produce a result that might be more desirable than the default result.

CONCLUSION

Creating custom reports that show your information in just the right way is now possible. As you read through the paper, you can start small using the basic options like inline formatting, STARTPAGE=, or the traditional COLUMNS= option to get a feel of how things can be laid out on the page. If your output requires a more polished presentation, then ODS Layout can provide you the tools you need. Even though ODS Layout is preproduction in SAS 9.2, you will find that it is useful and stable enough for your everyday use.

ACKNOWLEDGMENTS

The authors would like to thank Dan Heath, Tim Hunter, and Allison Crutchfield for their contributions to this paper.

CONTACT INFORMATION

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

Daniel O'Connor
SAS Institute Inc.
Building R, SAS Campus Drive
Cary, NC 27513
E-mail: Dan.OConnor@sas.com

Scott Huntley
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
E-mail: Scott.Huntley@sas.com

APPENDIX

This appendix includes the code for every example in this paper.

Example 1.1

```
options nodate nonumber;
ods escapechar = "^";
title '^{style [textdecoration=underline color=cx494068] First Quarter of 1999
Inventory}';
ods pdf file='sgf.pdf';

proc report data=sashelp.orsales nowd; where product_category='Team Sports' & quarter
= '1999Q1';
column year quarter product_line product_category product_group quantity=quantity2
quantity profit total_retail_price;
format profit dollar11.2 total_retail_price dollar11.2;
define year / '';
define quarter / '';
define product_line / '';
define product_category / '';
define product_group / '';
define quantity / '';
define quantity2 / noprint;
define profit / '';
define total_retail_price / '';
compute quantity;
if quantity2 < 400 then
call define (_col_, "style","style=[color=red]");
endcomp;
compute year;
    bg + 1;
    if mod(bg, 2) = 1 then
       call define(_row_, "style", "style={background=cx494068 color=cxbbb2e0}");
    else
       call define(_row_, "style", "style={background=cxbbb2e0}");
endcomp;

run;

ods pdf text = "^{style[just=c]Any items listed in  ^{style [color=red] red ^{style
[color=purple font_size=16pt] should be ^{style [font_style=italic] reordered }very}
soon} to avoid problems}";
ods pdf text = "^{style[just=c]Don't forget about unicode characters like alpha and
dagger -->   ^{style [font_size=16pt] ^{unicode alpha}  ^{unicode dagger}}}";
ods pdf close;
```

Example 1.2

```
options nodate nonumber;
data markets;
input  market $1 -15  total change;
cards;
Dow Jones       12170.00 18.75
NASDAQ          2100.76 -3.56
S&P 500         1343.42 -2.34
US 2yr Tsy %    2.53 .11
US 10yr Tsy %   2.92 -.23
;
ods escapechar = '^';
title 'You can do super ^{super script} and sub ^{sub script} ^{style
[textdecoration=line_through]as well}';
ods pdf file='stock.pdf' style=barrettsblue;
proc report data=markets nowd;
column market total change=change2 change;
define market / ' ';
define total / ' ';
define change2 / noprint;
define change / ' ';
format total comma12.2;
compute change;
if change2 < 0 then
call define(_col_, "style", "style=[color=red postimage='c:\sgf2009\down_red.png']");
else
call define(_col_, "style", "style=[postimage='c:\sgf2009\up_green.png']");
endcomp;
run;
ods pdf close;
```

Example 1.3

```
options nodate nonumber;
title 'Notice how titles are not affected by the columns= option';
ods pdf file='cols2.pdf' columns=2;
  proc print data=sashelp.orsales(obs=5);
  var year quarter product_line; run;

  proc print data=sashelp.orsales(obs=5);
   var year quarter product_line;
  run;

ods pdf close;
```

Example 1.4

```
options nodate nonumber;
title 'using the columns=3 option';
ods pdf file='cols3a.pdf' columns=3;

  proc print data=sashelp.orsales(obs=20);
  var year quarter product_line quantity profit; run;

  proc print data=sashelp.orsales(obs=5);
  var year quarter product_line; run;

ods pdf close;
```

Example 1.5

```
options nodate nonumber;
data stocks;
input time time8. average;
cards;
09:00am 23.42
09:10am 23.46
09:20am 23.48
09:30am 23.45
09:40am 23.42
09:50am 23.40
10:00am 23.33
10:10am 23.29
10:20am 23.20
10:30am 23.25
10:40am 23.30
10:50am 23.32
11:00am 23.33
11:10am 23.35
11:20am 23.38
11:30am 23.44
11:40am 23.46
11:50am 23.40
12:00pm 23.46
12:10pm 23.54
```

```
12:20pm 23.60
12:30pm 23.75
12:40pm 23.66
12:50pm 23.58
01:00pm 23.53
01:10pm 23.50
01:20pm 23.55
01:30pm 23.48
01:40pm 23.59
01:50pm 23.62
02:00pm 23.56
02:10pm 23.53
02:20pm 23.50
02:30pm 23.48
02:40pm 23.46
02:50pm 23.49
03:00pm 23.50
03:10pm 23.52
03:20pm 23.53
03:30pm 23.55
03:40pm 23.59
03:50pm 23.58
04:00pm 23.54
;

ods pdf file='graph.pdf' columns=2;
title "height and width set to 5";

ods graphics on / height=5in width=5in;
proc sgplot data=stocks;
label time = "Time of Day" average="Price";
format time timeampm5. average comma6.2;
xaxis grid notimesplit;
yaxis grid max=24 min=23 tickvalueformat=Data;
refline 23.42 / label="Open" lineattrs=(color=black);
refline 23.75 / label="High" lineattrs=(color=blue thickness=3);
refline 23.54 / label="Close" lineattrs=(color=lightgreen);
refline 23.20 / label="Low" lineattrs=(color=blue thickness=3);
series x=time y=average / lineattrs=(color=darkgreen thickness=3);
run;
ods graphics off;

proc print data=sashelp.orsales(obs=15);
var year quarter quantity; run;
ods _all_ close;
```

Example 1.6

```
options nodate nonumber;
title 'startpage=no and five obs of each dataset';
ods pdf file='cols_no.pdf' columns=2 startpage=no;
  proc print data=sashelp.orsales(obs=5);
  var year quarter product_line; run;
  proc print data=sashelp.orsales(obs=5);
  var year quarter product_line; run;
ods pdf close;
```

Example 1.7

```
options nodate nonumber;
title 'startpage=now after the 2nd table';
ods pdf file='cols_now.pdf' columns=2 startpage=no;
  proc print data=sashelp.orsales(obs=5);
  var year quarter product_line; run;
  proc print data=sashelp.shoes(obs=5);
  var region product stores;run;
  ods pdf startpage=now;
  proc print data=sashelp.orsales(obs=5);
  var year quarter product_line; run;
ods pdf close;
```

Example 2.1

```
ods listing close;

proc template;
define style Styles.Orionbackground;
 parent =Styles.Printer;
 style body /
       background=cx494068;
 end;
run;

options nodate nonumber;
ods escapechar="~";
title;
footnote "~{style [font_size=10pt just=right color=cxbbb2e0]Provided to you
compliments of SAS 9.2 using ODS Absolute Layout features.}";
ods pdf file="ExecutiveSummary.pdf" style=Styles.Orionbackground notoc;

ods layout start;

ods region y=2in ;
  ods text="~{style [preimage='c:\Public\SGF 2009\Orion Star
Info\or_internal\orionLogoMedium.jpg' color=cxbbb2e0 just=center font_size=72pt] Orion
Star}";

ods region y=3in x=3in;
ods text="~{style [color=cxbbb2e0 font_size=28pt]Sports & Outdoors}";

ods region y=5in;
  ods text="~{style [just=Center font_size=36pt color=cxbbb2e0]Executive Prospectus}";

ods region y=7in;
  ods text="~{style [just=Center font_size=20pt color=cxbbb2e0]For years 1999 through
2002}";

ods layout end;
title;
ods pdf close;
```

Example 2.2

```
ods listing close;

proc template;
define style Styles.OrionCalloutBlock;
 parent =Styles.Printer;
 style  LayoutRegion/
       background=cxbbb2e0;
 end;
run;

options nodate nonumber;
ods escapechar="~";
title "~{style [preimage='c:\Public\SGF 2009\Orion Star
Info\or_internal\orionstarHeader.jpg' width=100pct background=cx494068 color=cxbbb2e0
font_size=32pt]          Our Company           }";
footnote "~{style [font_size=10pt just=right color=cxbbb2e0]Provided to you
compliments of SAS 9.2 using ODS Absolute Layout features.}";
ods pdf file="StaticOutput.pdf" notoc;
ods layout start;
ods region;
ods text="~{style [preimage='c:\Public\SGF 2009\Orion Star
Info\or_internal\starLarge.gif' font_style=italic font_size=24pt color=cxbbb2e0]Who we
are...}";
ods region y=0.5in x=1in width=6in;
ods text="The fictional Orion Star Sports & Outdoors is an international retail
company that sells sports and outdoor products. The headquarters is based in the
United States, and retail stores are situated in a number of other countries including
Belgium, Holland, Germany, the United Kingdom, Denmark, France, Italy, Spain, and
Australia.";
ods region y=1.25in x=1in width=4in;
ods text="Products are sold through physical retail stores, as well as through mail-
order catalogs and the Internet. Customers who sign up as members of the Orion Star
Club organization can receive favorable special offers; therefore, most customers
enroll in the Orion Star Club. The sales data in the scenario only include the
purchases by Orion Star Club members in the years 1998 through 2002.";
ods region y=2.5in height=1in width=3in;
ods text="~{style [preimage='c:\Public\SGF 2009\Orion Star
Info\or_internal\starLarge.gif' font_style=italic font_size=24pt color=cxbbb2e0]What
we sell...}";
ods region y=3in x=1in width=4in height=1.75in;
ods text="Approximately 5500 different sports and outdoors products are offered at
Orion Star. Some of the products are not sold in certain countries, whereas others are
sold in volumes that reflect the different types of sports and outdoor activities that
are performed in each country. All of the product names are fictitious.";
ods text="~{newline}Products are organized in a hierarchy consisting of four levels:";
ods text="Product Line";
ods text="Product Category";
ods text="Product Group";
ods region y=4.75in height=1in width=5in;
ods text="~{style [preimage='c:\Public\SGF 2009\Orion Star
Info\or_internal\starLarge.gif' font_style=italic font_size=24pt color=cxbbb2e0]Where
we generate our profit...}";
ods region y=5.5in width=4.75in height=4in;
goptions device=png;
proc gchart data=sashelp.orsales;
pie product_category /
         sumvar=profit
      value=none
      percent=outside slice=outside;
run;
ods region y=5.5in x=4.625in width=4.5in height=5in;
proc report nowd data=sashelp.orsales style(header)={background=cx494068
color=cxbbb2e0};
 columns product_category profit;
 define product_category / group;
 define profit /analysis sum format=dollar14.;
run;
quit;

ods pdf style=Styles.OrionCalloutBlock;
ods region x=6in y=1.0625in width=2in height=1in;
  ods text="~{style [background=cx494068 color=cxbbb2e0 font_size=24pt just=center
font_style=italic width=100pct] Our Mission }";
  ods text="~{style [font_style=italic vjust=center font_size=10pt just=center]To
deliver the best quality sporting equipment, accessories, and outdoor equipment for
all seasons at the most affordable prices.}";
ods region x=6in y=2.1875in width=2in height=1in;
  ods text="~{style [background=cx494068 color=cxbbb2e0 font_size=24pt just=center
font_style=italic width=100pct] Our Vision }";
  ods text="~{style [font_style=italic vjust=center font_size=10pt just=center]To
transform the way the world purchases sporting and outdoor equipment.}";
ods region x=6in y=3.3125in width=2in height=1in;
  ods text="~{style [background=cx494068 color=cxbbb2e0 font_size=24pt just=center
font_style=italic width=100pct] Our Values }";
  ods text="~{style [font_style=italic vjust=center font_size=10pt
just=center]Customer focused, Swift and Agile, Innovative, Trustworthy}";
ods region x=6in y=4.4375in width=2in height=1in;
  ods text="~{style [background=cx494068 color=cxbbb2e0 font_size=24pt just=center
font_style=italic width=100pct] Our Goal }";
  ods text="~{style [font_style=italic vjust=center font_size=10pt just=center]To grow
sales by 15% annually while also improving profit margin through innovative thinking
and operational efficiencies.}";
```

```
ods pdf style=Styles.Printer;
ods layout end;
title;
ods pdf close;
```

Example 2.5

```
ods listing close;
options nodate nonumber;

data execs;
length photo $64 name $32 bio $256;
input photo $64. / name $32.  bio $256.;
datalines;
c:\Public\SGF 2009\Orion Star Info\or_internal\scott.jpg
Scott Huntley
Scott Huntley is the CEO and President of Orion Star, which he co-founded in 1976.
Orion leads the industry in innovation with its award-winning recreational  and
outdoor equipment.
c:\Public\SGF 2009\Orion Star Info\or_internal\dan.jpg
Dan OConnor
Dan OConnor is the CIO of Orion Star, which he also helped co-found in 1976. Orion
continues to revolutionize the sporting goods and outdoors accessory industry though
the adoption of technology to offer the best products at the most affordable price.
;

data reports;
label title="Title";
length title $64 url $64;
input title $64. / url $64.;
datalines;
2008 Proxy Statement
http://www.orionstar.com/reports/proxy2008.pdf
2008 Annual Review
http://www.orionstar.com/reports/annualreport2008.pdf
;

data events;
length icon $64 type $64 title $64 title2 $64 location $64 dateandtime $64;
input  icon $64. / type $64. / title $64.  / title2 $64. / location $64. / dateandtime
$64.;
cards;

Webcast
Fiscal 2008 Company Results


1/9/2009 at 1:00 p.m. ET
c:\Public\SGF 2009\Orion Star Info\or_internal\nrf.gif

National Retail Federation Show

Jacob K. Javits Convention Center, New York City
January 11-14, 2009


Orion Announces Plans to open store in Washington DC

Gaylord National Resort Center, Washington DC
March 22-25, 2009

Webcast
First Quarter Fiscal 2009 Results


4/10/2009 at 1:00 p.m. ET
;
```

```
data news;
length title $64 url $64 printablefile $64;
label date='Date' title='Title';
informat date date9.;
format date date9.;
input  date date9. / title $64. / url $64. / printablefile $64.;
cards;
11Oct2008
Orion Announces Record 3Q Sales for 2008
http:\\www.orionstar.com\news\Entry7.html
http:\\www.orionstar.com\printablenews\Entry7.pdf
01Dec2008
Orion Introduces a new CLUB Signature Visa Card
http:\\www.orionstar.com\news\Entry1.html
http:\\www.orionstar.com\printablenews\Entry1.pdf
25Dec2008
Orion Star receives additional analyst coverage from Garner AMG
http:\\www.orionstar.com\news\Entry2.html
http:\\www.orionstar.com\printablenews\Entry2.pdf
01Jan2009
Orion's President Scott Huntley presents at the World Economic Forum
http:\\www.orionstar.com\news\Entry3.html
http:\\www.orionstar.com\printablenews\Entry3.pdf
01Jan2009
Orion Announces Increased Revolving Credit Facility
http:\\www.orionstar.com\news\Entry4.html
http:\\www.orionstar.com\printablenews\Entry4.pdf
09Jan2009
Orion Announces Plans to add 24 new stores in 2009
http:\\www.orionstar.com\news\Entry5.html
http:\\www.orionstar.com\printablenews\Entry5.pdf
10Jan2009
Board of Directors approves purchase of Rival Sports Warehouse
http:\\www.orionstar.com\news\Entry6.html
http:\\www.orionstar.com\printablenews\Entry6.pdf
;


data stocks;
input time time8. average;
cards;
09:00am 23.42
09:10am 23.46
09:20am 23.48
09:30am 23.45
09:40am 23.42
09:50am 23.40
10:00am 23.33
10:10am 23.29
10:20am 23.20
10:30am 23.25
10:40am 23.30
10:50am 23.32
11:00am 23.33
11:10am 23.35
11:20am 23.38
11:30am 23.44
11:40am 23.46
11:50am 23.40
12:00pm 23.46
12:10pm 23.54
12:20pm 23.60
12:30pm 23.75
12:40pm 23.66
12:50pm 23.58
01:00pm 23.53
01:10pm 23.50
```

```
01:20pm 23.55
01:30pm 23.48
01:40pm 23.59
01:50pm 23.62
02:00pm 23.56
02:10pm 23.53
02:20pm 23.50
02:30pm 23.48
02:40pm 23.46
02:50pm 23.49
03:00pm 23.50
03:10pm 23.52
03:20pm 23.53
03:30pm 23.55
03:40pm 23.59
03:50pm 23.58
04:00pm 23.54
;
ods escapechar="~";
title "~{style [preimage='c:\Public\SGF 2009\Orion Star
Info\or_internal\orionstarHeader.jpg' width=100pct background=cx494068 color=cxbbb2e0
font_size=32pt just=left] Investor Relations}";
footnote "~{style [font_size=10pt just=right color=cxbbb2e0]Provided to you
compliments of SAS 9.2 using ODS Gridded Layout features.}";
ods pdf file="GriddedOutput.pdf" notoc;
ods layout start columns=2 column_widths=(5.25in 2.5in);
ods region;
ods text="~{style [just=left color=cx494068 font_size=20pt font_weight=bold
width=100pct]Our Creators}";
data _null_;
set execs end=eof;
if _n_=1 then do;
   dcl odsout obj();
   obj.layout_gridded(columns: 2,
                      overrides: "just=left");
end;
obj.region(width: "1.25in");
obj.image(file: photo,
          overrides: "just=left width=100pct");
obj.format_text(data: strip(name),
          overrides: "just=left width=100pct");
obj.region(width: "3.25in");
obj.format_text(data: strip(bio));

if eof ne 1 then
   obj.format_text(data: "");
else
   obj.layout_end();
;run;

ods text="~{newline}";
ods text="~{style [just=left color=cx494068 font_size=20pt font_weight=bold
width=100pct]Upcoming Events}";
data _null_;
set events end=eof;
if _n_ = 1 then do;
   dcl odsout obj();
   obj.table_start(overrides: "width=100pct just=left rules=none frame=void
cellpadding=0 cellspacing=0");
   first_row=1;
end;
/* Strip off the blanks */
type = strip(type);
icon = strip(icon);
title = strip(title);
title2 = strip(title2);
location = strip(location);
dateandtime = strip(dateandtime);
```

```
/* How many rows do I need to span? */
row_span = 0;
if type        ne "" then row_span=row_span+1;
if title       ne "" then row_span=row_span+1;
if title2      ne "" then row_span=row_span+1;
if location    ne "" then row_span=row_span+1;
if dateandtime ne "" then row_span=row_span+1;

/* Do we need to substitute a predefined icon? */
if icon = "" then do;
   if type = "Webcast" then
      icon = "c:\Public\SGF 2009\Orion Star Info\or_internal\microphone.jpg";
end;

/* Display icon */
obj.row_start();
if icon = "" then do;
   obj.format_cell(data: "",
                   row_span: row_span);
end;
else do;
   obj.format_cell(data: "",
                   row_span: row_span,
                   overrides: "vjust=center preimage='" || icon || "'");
end;

/* Write out descriptive information */
if type ne "" then do;
  if first_row = 0 then
     obj.row_start();
  obj.format_cell(data: type,
                  overrides: "just=left");
  obj.row_end();
  first_row = 0;
end;
if title ne "" then do;
  if first_row = 0 then
     obj.row_start();
  obj.format_cell(data: title,
                  overrides: "just=left");
  obj.row_end();
  first_row = 0;
end;
if title2 ne "" then do;
  if first_row = 0 then
     obj.row_start();
  obj.format_cell(data: title2,
                  overrides: "just=left");
  obj.row_end();
  first_row = 0;
end;
if location ne "" then do;
  if first_row = 0 then
     obj.row_start();
  obj.format_cell(data: location,
                  overrides: "just=left");
  obj.row_end();
  first_row = 0;
end;
if dateandtime ne "" then do;
  if first_row = 0 then
     obj.row_start();
  obj.format_cell(data: dateandtime,
                  overrides: "just=left");
  obj.row_end();
  first_row = 0;
end;
```

```
if eof = 1 then do;
  obj.table_end();
end;
else do;
  /* Write separation line between events */
  obj.row_start();
  obj.format_cell(data: "",
                  column_span: 2 );
  obj.row_end();
end;
;run;

options missing="";
ods text="~{newline}";
ods text="~{style [just=left color=cx494068 font_size=20pt width=100pct
font_weight=bold]Recent News}";
proc report data=news nowd style(report)={width=100pct}
style(header)={background=cx494068 color=white};
columns date title printable;
define printable / computed " ";
compute title;
 if strip(url) ne "" then
    call define(_col_, "style", "style=[color=blue textdecoration=underline url='" ||
strip(url) || "']");
endcomp;
compute printable;
 call define(_col_, "style", "style=[just=center preimage='c:\Public\SGF 2009\Orion
Star Info\or_internal\printer.gif' url='" || strip(pritablefile) ||"']");
endcomp;
run;
options missing=.;

ods region;
ods text="~{style [just=left color=cx494068 font_size=20pt width=100pct
font_weight=bold]Stock Information}";
data _null_;
dcl odsout obj();
obj.table_start(name: "Stock",
                overrides: "width=100pct");
obj.row_start();
obj.format_cell(data: "STAR (Common Stock)",
                column_span: 2,
                           overrides:   "backgroundcolor=cx494068 color=white");
obj.row_end();
obj.row_start();
obj.format_cell(data: "Exchange",
                overrides: "just=left font_weight=bold");
obj.format_cell(data: "NYSE (US Dollar)",
                overrides: "just=left");
obj.row_end();
obj.row_start();
obj.format_cell(data: "Price",
                overrides: "just=left font_weight=bold");
obj.format_cell(data: "$23.54",
                overrides: "just=left");
obj.row_end();
obj.row_start();
obj.format_cell(data: "Change (%)",
                overrides: "just=left font_weight=bold");
obj.format_cell(data: "0.12 (.51%)",
                overrides: "just=left color=green");
obj.row_end();
obj.row_start();
obj.format_cell(data: "Volume",
                overrides: "just=left font_weight=bold");
obj.format_cell(data: "11,390",
                overrides: "just=left");
obj.row_end();
obj.row_start();
obj.format_cell(data: "Data as of 1/7/2009  9:43 a.m. ET ~{newline} Minimum 20 minute
delay",
                column_span: 2,
                           overrides: "font_size=10pt just=left");
obj.row_end();
obj.table_end();
run;

Title "Daily Trading Data";
proc sgplot data=stocks;
label time = "Time of Day" average="Price";
format time timeampm5. average comma6.2;
xaxis grid notimesplit;
yaxis grid max=24 min=23 tickvalueformat=Data;
refline 23.42 / label="Open" lineattrs=(color=black);
refline 23.75 / label="High" lineattrs=(color=blue thickness=3);
refline 23.54 / label="Close" lineattrs=(color=lightgreen);
refline 23.20 / label="Low" lineattrs=(color=blue thickness=3);
series x=time y=average / lineattrs=(color=darkgreen thickness=3);
run;
Title;

ods text="~{newline}";
ods text="~{style [just=left color=cx494068 font_size=20pt width=100pct
font_weight=bold]Featured Reports}";
proc report data=reports nowd style(Report)={width=100pct}
style(header)={background=cx494068 color=white};
columns title;
compute title;
 if strip(url) ne "" then
    call define(_col_, "style", "style=[color=blue textdecoration=underline url='" ||
strip(url) || "']");
endcomp;
run;
ods text="~{newline}";
ods text="~{style [just=left color=cx494068 font_size=20pt width=100pct
font_weight=bold]Annual Report}";
ods text="~{style [preimage='c:\Public\SGF 2009\Orion Star
Info\or_internal\AnnualReport1.jpg']}";
ods text="~{newline}";
ods text="~{style [just=left color=cx494068 font_size=12pt width=100pct
font_weight=bold]Corporate Headquarters}";
ods text="~{style [just=left width=100pct]SAS Campus Drive}";
ods text="~{style [just=left width=100pct]Building R}";
ods text="~{style [just=left width=100pct]Cary, NC 27513}";
ods layout end;
ods pdf close;
title;
footnote;
```