Paper 015-2013

Take Home the ODS Crown Jewels: Master the New Production Features of ODS LAYOUT and Report Writing Interface Techniques

Daniel O'Connor, SAS Institute Inc.

ABSTRACT

The ODS "crown jewels" are the most sought-after features that allow the customer to create sophisticated documents that can readily be published via print, Web, or mobile devices. Let's journey with SAS® 9.4 as we explore the new enhancements in this production release of ODS Layout and Report Writing Interface. This example-driven content is intended to empower and captivate the novice ODS customer while challenging even the most advanced.

INTRODUCTION

"Crown Jewels" could be thought of as royal artifacts that are passed amongst monarchs to symbolize the right to rule, or as the most sought after traits, assets, and possessions of a company. For the sake of our conversation today the "ODS Crown Jewels" are to going to refer to the latest breakthrough technology that allows you to create rich data visualizations that combine multi-view dashboards in a manner that is easy to digest. I refer to this holistic approach of consolidated reporting as Enterprise Business Reporting. Enterprise Business Reporting is the evolution of our traditional reporting framework to address advanced reporting requirements for the 21st century. Whether the reports are going be distributed in print, on the web, via email, or on a mobile device the Enterprise Business Reporting framework will easily accommodate all types of published reports without any format specific knowledge. With SAS® 9.4 you will have all the production quality tools available to easily create sophisticated publications. In SAS® 9.4 you will find a variety of new enhancements with ODS Layout and the Report Writing Interface, as well as new features that handle the creation of structured text, dynamic lists, and enhanced Rich Text support.

ODS LAYOUT

The ODS Layout architecture provides the ability to arrange items such as graphs, images, text, and tables on a virtual page following a "Pixel Perfect Design". The concept of "Pixel Perfect" should not be a literal interpretation, but simply the process of aligning and sizing all of the content that makes up your page/report design. ODS Layout does provide the ability to do physical pixel perfect placement, but you will find your report will not scale correctly when things change which is why this approach should be used only when the report requirements are very static in nature. Prior to SAS® 9.4 ODS Layout was available in a pre-production status, and had limited support for many of its features. With the release of SAS® 9.4 all features previously included in the ODS Layout architecture as well as a number of new features are all production quality. ODS Layout has two distinct forms, absolute and gridded, each having 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. When publishing reports you have to take into consideration the output destinations because not all destinations support ODS Layout, or possibly do not support all features equally. For example, when producing a PDF report you must take into consideration the physical page constraints, but when producing an HTML report the WEB browser has scroll bars so you don't have any physical limitations. These external physical constraints provide additional restrictions on how your grid will behave.

ODS REPORT WRITING INTERFACE (RWI)

The ODS Report Writing Interface was designed exclusively to address the custom report writing requirements that go far beyond the traditional tabular output produced by procedures. The custom report writing capabilities are also more commonly referred to as DATA _NULL_ report writing which has long been an integral part of the SAS reporting solution. The ODS Report Writing Interface is intended to fully embrace ODS features such as proportional fonts, traffic lighting, colors, images, and Unicode characters, while at the same time providing pixel perfect placement capabilities. This interface has methods to allow you to produce tables, text, insert images, and control the placement of everything in the report. This interface is not only fully integrated with all the capabilities of the ODS System, but also takes advantage of the rich programming features that the DATA step offers, such as conditional logic, formatting capabilities, by-group processing, arrays, and a wealth of other features. The ODS Report Writing Interface is an object-oriented language that provides you with flexibility and control to address even the most rigid reporting requirements with ease.

PAGE CONSIDERATIONS

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

PAPERSIZE The PAPERSIZE= option can be specified by using a predefined paper size (i.e. LETTER, LEGAL, US POSTCARD), or explicitly specified with a custom paper width, and height (i.e. (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 (i.e. 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

RICH TEXT (IN-LINE FORMATTING)

Being flexible and showing off important details in your output is critical. Using in-line formatting is a quick and easy way to modify small portions of your output with customized characteristics. Using the ODS ESCAPECHAR statement and the inline formatting syntax, you can modify the styles of TITLEs, FOOTNOTEs, TEXT statements, and data cells.

```
ods escapechar=”^”;
```

The in-line 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)]> text}
```

The “^” character 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 name. Next, you can use any style element name 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 individual style attributes. Some common style attributes are color, font, and size. You can set as many style attributes as needed within the square brackets. The syntax ends with the text you want to be formatted followed by the ending curly bracket }. Instead of specifying the ESCAPECHAR in your program every time, the hard coded ESCAPECHAR place holder “(*ESC*)” is more convenient.

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

GLOBAL ODS LAYOUT FEATURES

TITLE BEHAVOIR

The use of the global TITLE and FOOTNOTE statements has significantly changed in SAS® 9.4. The global TITLE and FOOTNOTE statements are fully integrated with ODS Layout and provide a very powerful feature to customize your report. Let's first summarize how the global TITLE and FOOTNOTE statements worked in the past before examining the new behavior in layout.

NON-LAYOUT

Traditionally when a PROC executes the output is always started on a new page. Prior to the execution of the procedure the physical page is initialized (if it is a page based output destination), during which all known TITLEs are placed at the top of the page, and FOOTNOTEs at the bottom of the page. They will be repeated on every subsequent page until the output is either exhausted, or the TITLES and FOOTNOTEs are changed. If the output destination supports the STARTPAGE=<no | never> options TITLEs and FOOTNOTEs are all lost after the page initialization occurs.

LAYOUT

When the LAYOUT is the first thing on an empty (new) page the behavior is identical to the PROC. The physical page is initialized (if it is a page based output destination), all known TITLEs are placed at the top of the page, and FOOTNOTEs at the bottom of the page. They will be repeated at the top and bottom on every subsequent page until this LAYOUT is closed. However, if the LAYOUT is not the first thing on the page, ONLY the TITLEs and FOOTNOTEs that have changed will be displayed prior to the start of the LAYOUT. Similarly when a REGION statement is executed any TITLEs and FOOTNOTEs that have changed will be displayed at the beginning of the REGION. If the REGION contains multiple PROCs in conjunction with additional TITLE and FOOTNOTE statements, ONLY those that have changed will be rendered before and after each PROCs output respectively.

POSITIONING

In releases prior to SAS® 9.4 ABSOLUTE layout provided X and Y positioning capabilities, but GRIDDED layout did not. In SAS® 9.4 X and Y arguments are now valid on all layout types. This allows a layout to begin at an exact location on the physical page. The X and Y positioning capabilities are only supported in page based destinations.

Valid Destinations: **PRINTER (PDF, PCL, PSL)**

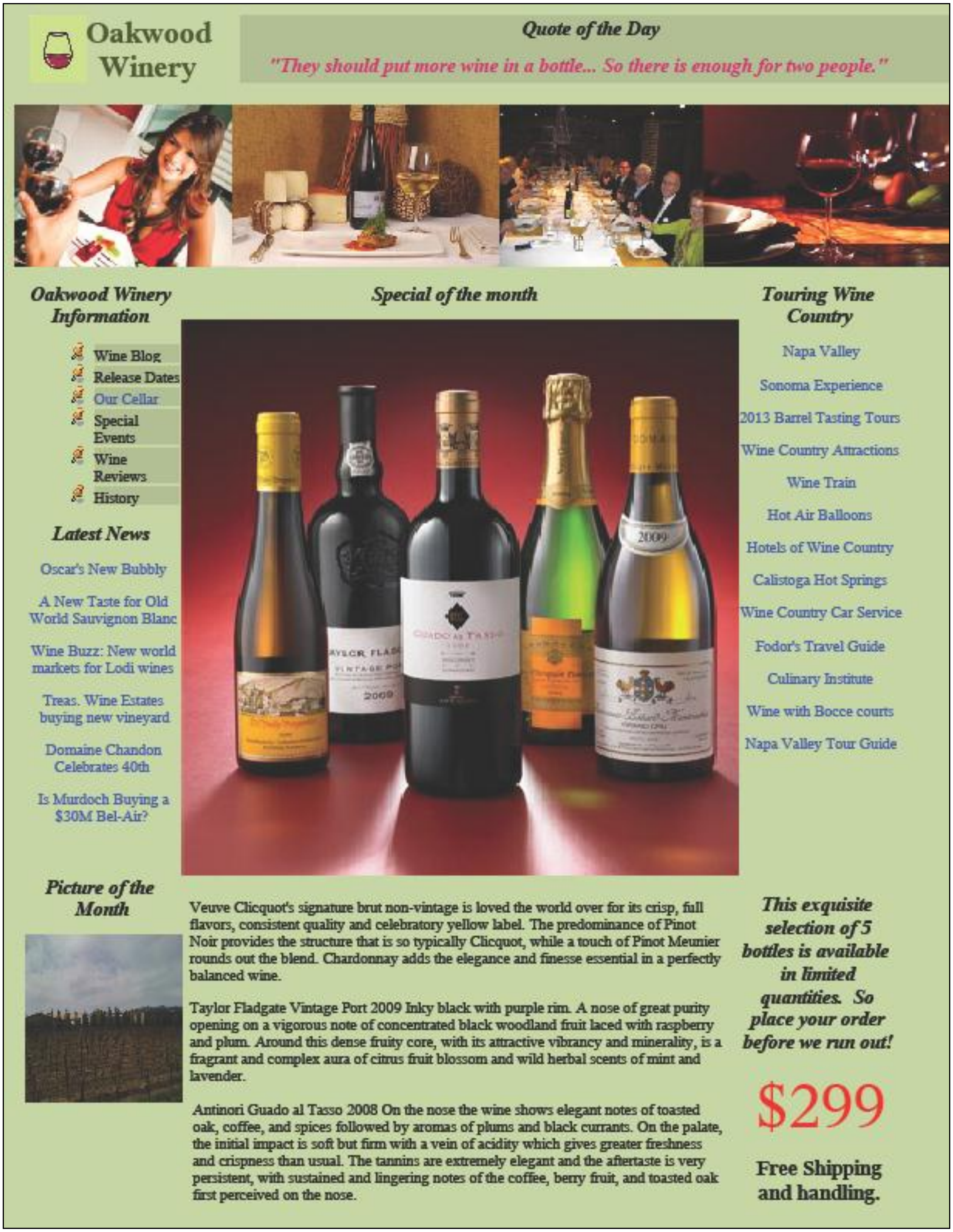
STYLE

The STYLE= option is accepted on either the ODS Layout statement or the ODS Region statement which provides a mechanism to supplement or override stylistic settings coming from the ODS Style. The syntax is similar to what is used for Rich Text support, and the REPORT, PRINT, and TABULATE procedures. The ODS LAYOUT STYLE option is considered pre-production in SAS® 9.4**.**

STYLE = <style element> [<style attributes>]

THE CROWN JEWEL OF CALIFORNIA

The paper uses an example driven approach to introduce new concepts. We are going to focus exclusively on examples using one of California's Crown Jewels (Wine Country) as our inspiration to drive our discussions. The Oakwood Winery is a fictitious vineyard and wine distribution company that is based in San Francisco. The business was founded in 1987 and distributes wine and accessories throughout the world. Oakwood Winery actively embraces technology in their business which has allowed them to grow revenue for 25 consecutive years. So let's explore how Oakwood Winery has utilized the new Enterprise Business Reporting enhancements available in SAS® 9.4 to further modernize their operations. Let's first look at the Oakwood Winery's home web page as a whole, and then we will break down sections of the report as we cover a variety of topics. As you would expect, a home page should be informative, appealing, and often contains general informational types of content that provides opportunities to navigate to more specific content. In Oakwood Winery's home page we can see that it contains a quote of the day, some flashy images, links for additional corporate information, news, information about planning a trip, as well as a special of the month. Home pages are generally not a data centric type of report so traditional ODS Layout features typically will suffice.



Quote of the Day

"They should put more wine in a bottle... So there is enough for two people."

Oakwood Winery Information

Wine Blog
Release Dates
Our Cellar
Special Events
Wine Reviews
History

Latest News

Oscar's New Bubbly
A New Taste for Old World Sauvignon Blanc
Wine Buzz: New world markets for Lodi wines
Treas. Wine Estates buying new vineyard
Domaine Chandon Celebrates 40th
Is Murdoch Buying a $30M Bel-Air?

Picture of the Month

Special of the month

Veuve Clicquot's signature brut non-vintage is loved the world over for its crisp, full flavors, consistent quality and celebratory yellow label. The predominance of Pinot Noir provides the structure that is so typically Clicquot, while a touch of Pinot Meunier rounds out the blend. Chardonnay adds the elegance and finesse essential in a perfectly balanced wine.

Taylor Fladgate Vintage Port 2009 Inky black with purple rim. A nose of great purity opening on a vigorous note of concentrated black woodland fruit laced with raspberry and plum. Around this dense fruity core, with its attractive vibrancy and minerality, is a fragrant and complex aura of citrus fruit blossom and wild herbal scents of mint and lavender.

Antinori Guado al Tasso 2008 On the nose the wine shows elegant notes of toasted oak, coffee, and spices followed by aromas of plums and black currants. On the palate, the initial impact is soft but firm with a vein of acidity which gives greater freshness and crispness than usual. The tannins are extremely elegant and the aftertaste is very persistent, with sustained and lingering notes of the coffee, berry fruit, and toasted oak first perceived on the nose.

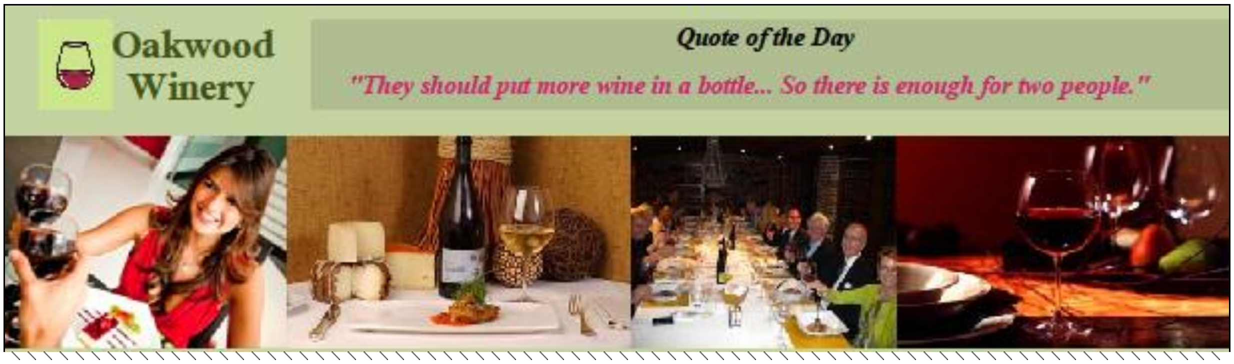
Touring Wine Country

This exquisite selection of 5 bottles is available in limited quantities. So place your order before we run out!

$299

Free Shipping and handling.

EXAMPLE 1



Oakwood Winery

Quote of the Day

"They should put more wine in a bottle... So there is enough for two people."

Clear any title or footnote.

```
title;
footnote;
```

In Example 1 we will create the heading layout section of the page. Instead of taking a screen shot of the page, and pasting it as an image into the Microsoft® WORD document, I chose to insert an Adobe Acrobat® PDF document with only the items contained in the heading layout. To create a PDF that contains only the heading items I changed the paper size to have a width of 8.5in and a height of 2.4in. I also specified the orientation of the paper to be portrait (which is the default) and changed all the margins to be .001in. Now you might think changing the margins to be .001in is very odd, but if you tried changing the margins to 0in the value would be ignored (because it means to use the printer's default margins). This little tip will allow the entire page to be filled with the background color. There are a couple of caveats to consider when creating a PDF with a background color. The PDF may look wonderful on the screen, but when you print it your printer may have physical margin limitations that prevent the printer from printing all the way to the edge of the page. You must also make sure not to place content within the physical printer's margin limitations because your output may be clipped. These limitations are printer specific.

```
options papersize    = (8.5in 2.4in)
        orientation  = portrait
        leftmargin   = .001in
        rightmargin  = .001in
        topmargin    = .001in
        bottommargin = .001in
        nodate
        nonumber;
```

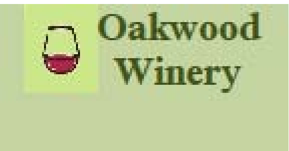
Use a custom ODS Style to change the background color of the entire page by modifying the BODY style element.

```
proc template;
define style Styles.Oakwood;
  parent = Styles.printer;
  class Body /
    backgroundcolor=cxc4d4a1;
end;
run;
ods pdf file="Example1.pdf" notoc style=Styles.Oakwood;
```

Create a gridded layout of 2 columns; I set a Y location for the layout to begin at .1in from the top of the page. The region width is specified in percentage units. By setting my region widths in percentages it makes it easy to size and position, but it also automatically adjusts based on the global PAPERSIZE and ORIENTATION options. The reason that I am specifying a Y position of .1in from the top of the page is that I know that my printer has a physical top margin of .1in and that if I allowed the layout to begin prior to the .1in physical margin my output would be clipped when I attempted to print the PDF. Setting all global margin options to .001in, and then setting a layout X and Y position allows me to both have the entire background of the PDF file to have the greenish background color (when viewing on-line), but also ensures that everything will still appear well sending content to the printer.

```
ods layout gridded x=.1in y=.1in columns=2;
ods region width=25%;
```

The following content is created by using the ODS Report Writing Interface (RWI). As I mentioned earlier the RWI is an object oriented language that is available inside the DATA STEP. To begin using the RWI the first thing to do is declare an ODS Report Writing class (odsout) object and assign it to a local variable name (i.e. obj). Also notice that I am nesting the RWI code inside an ODS statement based layout region with a width of 25 percent of the page. The following picture is the portion of the output the RWI code is producing.



Oakwood
Winery

```
data _null_;
dcl odsout obj();
```

Custom report writing can encompass a variety of different reporting requirements. In the following section, I am going to arrange an image and 2 strings in a grid by using the LAYOUT methods. Often times there is confusion around using ODS statement based LAYOUT vs. ODS Report Writing LAYOUT. There really is no reason to be alarmed because they do the exact same thing. You may need additional DATA step features such as conditional logic, formatting capabilities, by-group processing, arrays, or any other miscellaneous DATA step features to satisfy your reporting requirements. None of these are requirements in this example, but I just wanted to show the similarities in a simple scenario. The 'obj' is simply a data step variable and can be called anything you would like. The difference is that the variable does not have a literal numeric or character value, but provides a handle (mechanism) to communicate with ODS directly much like procedures. Create a grid of two columns, and eliminate the space between each row and column defined in the ODS style.

```
obj.layout_gridded(columns: 2,
                   column_gutter: "0in",
                   row_gutter: "0in");
```

Place an image in the first column of the grid. The REGION method simply instructs the image to be placed in the in the first column in the layout. The image method only has 1 required argument which is the filename to the image that is to be inserted. By using the optional WIDTH and HEIGHT arguments I am instructing ODS to resize the image if necessary to be .5in x .6in.

```
obj.region();
       obj.image(file:   "Images/wine.jpg",
                 width:  ".5in",
                 height: ".6in");
```

Place 2 strings in the second column of the grid. The REGION method is positioning the content into the next available column in the layout (Column 2). By using the FORMAT_TEXT method I am instructing ODS to place the fixed text with special rich text attributes.

```
obj.region();
       obj.format_text(data: "Oakwood",
                       style_attr: "color=cx405218
                                    font_weight=Bold
                                    font_size=20pt");

       obj.format_text(data: "Winery",
                       style_attr: "color=cx405218
                                    font_weight=Bold
```

```
                                                      font_size=20pt");
```

The LAYOUT_END method closes the current ODS layout grid, and we terminate the data step.

```
                obj.layout_end();
                run;
```

You are not required to use a layout when using the RWI. I only chose to use LAYOUT because I wanted to arrange the content in a grid. The IMAGE and FORMAT_TEXT methods can be used independently to produce output however you see fit.

The second column of the parent layout starts in column 2 and has a width of 75 percent of the page width. This region contains the “Quote of the Day” output. The background of the region will be a darker green because we are using the new STYLE= override option on the region statement. The STYLE= override statement is available both on the LAYOUT statement as well as the REGION statement. In this case I am only interested in altering the background of region.

```
            ods region width=75% style={backgroundcolor=cxafbd91};
```

Since title1 has changed we are going to generate the new title in the region prior to the first piece of output that is contained within the region. Typically when you think of output you normally think of PROC tabular or graphical output. The same rules apply for text beginning in SAS® 9.4.

```
            title "Quote of the Day";
            ods text="";
            ods text='(*ESC*){style SystemTitle [color=cxC92C6B]"They should put more
            wine in a bottle... So there is enough for two people."}';
```

Place the fun picture in ROW 2 spanning both columns, and resize the width of the image to be 100% of the page.

```
            ods region column_span=2;
            data _null_;
            dcl odsout obj();
            obj.image(file: "Images/funheader.jpg",
                      width: "100%");
            run;

            ods layout end;
            ods _all_ close;
```

We will continue with the rest of this example later in the paper after we have introduced some additional new features.

GRIDDED LAYOUT

A gridded layout is a flexible two-dimensional structure (similar to a spreadsheet or piece of graph paper) of intersecting vertical and horizontal containers that is used to arrange the content of a report. In today’s modern design framework the grid concept is by far the most powerful as well as the most common. Gridded layout is much more suitable for traditional SAS reporting. The dynamic nature of SAS reporting where data sources change daily may force your report design to be adjusted to allow more/less space dynamically. Today’s reporting needs have matured beyond simple reports, and include sophisticated dashboard of graphs, text, and images. Gridded layout has a variety of options to control various aspects of the grid, such as the number of rows or columns, gutter (space between rows/columns), the ability to span multiple rows/columns. All of these concepts are addressed in “Breaking New Ground With SAS® 9.2 ODS LAYOUT Enhancements”, but the real power of gridded layout is its elastic nature. When populating a grid from a data centric model you may not know how many categories of information may be available hence you may not want to hard code the number of rows or columns, but instead provide the grid a hint as to how to populate the grid elastically growing/shrinking the number of containers to meet the desired data centric model.

ADVANCE

ODS Layout is specifically designed to address the strengths of the SAS Enterprise Reporting architecture which is data centric in nature. ODS Layout can contain multiple procedures, a single procedure that iteratively produces output based on a category value, or may use analytic procedures that produce graphic and tabular output simultaneously. The ADVANCE option provides an additional feature that will populate the layout gird in a data centric manner.

ADVANCE = <EXPLICIT (Default) | PROC | OUTPUT (TABLE) | BYGROUP >

EXPLICIT By default, ODS Layout only moves to the next available region in the grid when an “ods region;” statement is explicitly executed.

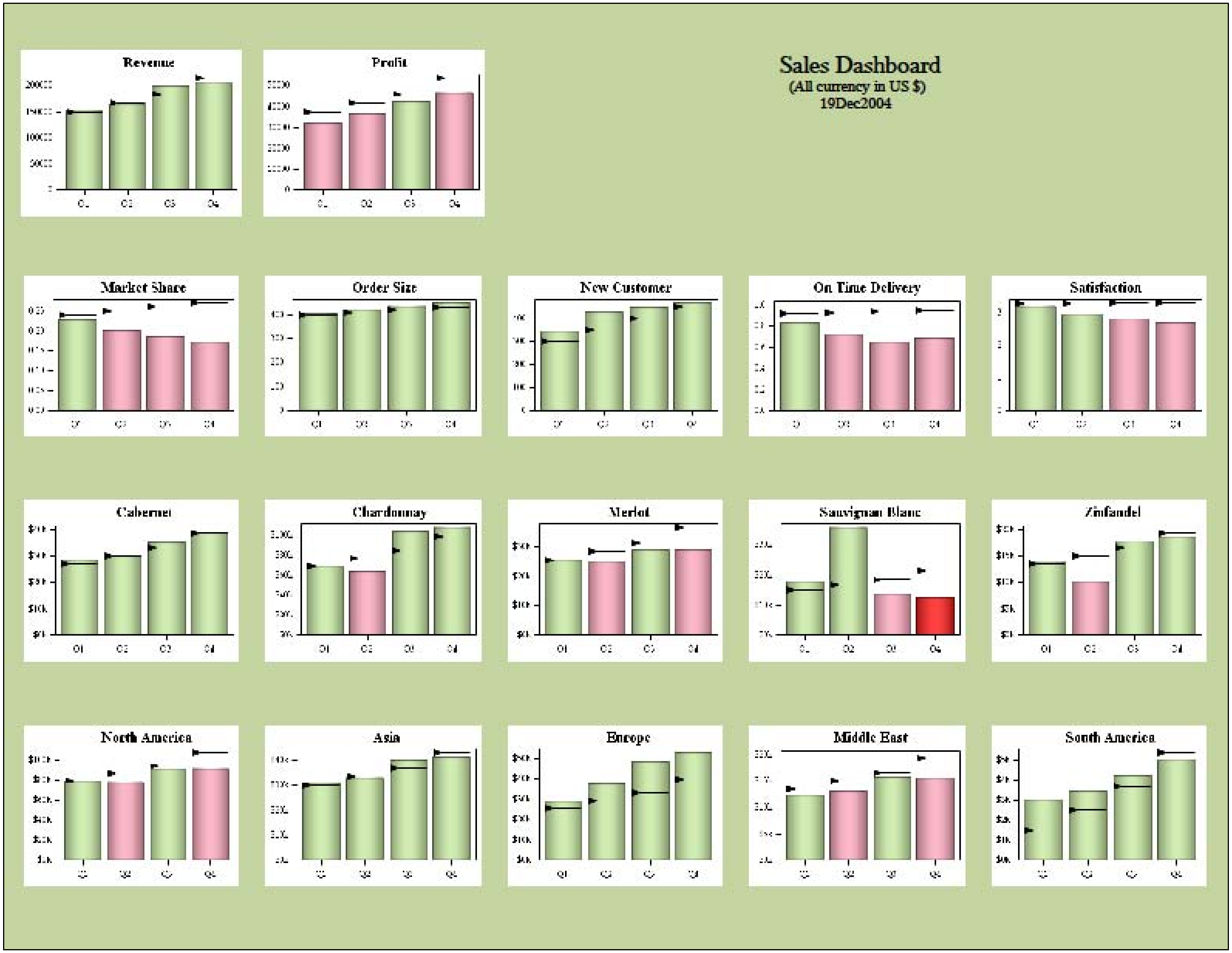
PROC When using the advance=PROC option, the ODS layout will advance to the next available region in the grid whenever a new PROC is encountered unless a new “ods region;” statement immediately proceeds the procedure.

OUTPUT When using the advance=OUTPUT option, the ODS layout will advance to the next available region in the grid whenever a new table or graph is encountered unless a new “ods region;” statement immediately proceeds the procedure.

BYGROUP When using the advance=BYGROUP option, the ODS layout will advance to the next available region in the grid whenever a new PROC or BYGROUP is encountered unless a new “ods region;” statement immediately proceeds the procedure. In the event that a procedure does not contain by processing the LAYOUT will advance on a new PROC boundary. This is done so that when using some PROCs with BYGROUP processing and some without the contents of the next PROC will always start in its own region.

DATA CENTRIC GRID

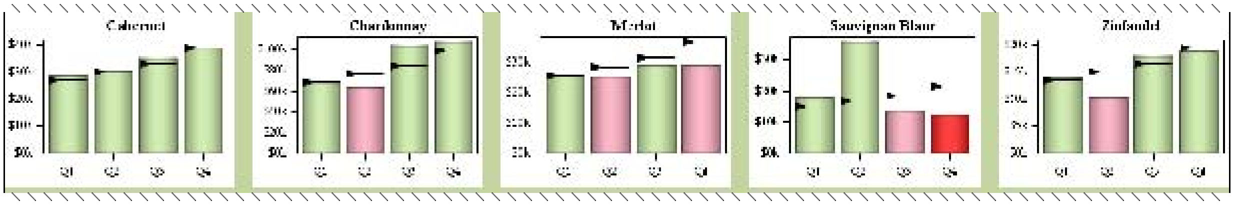
Most often reports are not as static as our Oakwood Winery home page example. Our next example will take a more traditional data centric approach that displays a dashboard of sales information. This report conveys sales information in a variety of categories, while traffic highlighting areas of concern, and showing expected targets. Once again we are using ODS statement based layout to arrange all the output on a single page. Let’s first take a look at the dashboard in its entirety and then we will further break down areas of interest.



Sales Dashboard

(All currency in US $)
19Dec2004

EXAMPLE 2



We are going to concentrate on the 3rd row of the output for this example. I opted to alter the paper size and orientation similar to the way I did in Example 1. We are also suppressing the automatic BYLINES by using the NOBYLINE global option. The intention is to generate the BYLINE information within the graph itself. We do this by using the “Entrytitle _byline_;” statement in the ODS Graphic template language(GTL) (see below).

```
options papersize=(8.5in 1.25in)
        orientation=portrait
        leftmargin=.001in
        rightmargin=.001in
        topmargin=.001in
        bottommargin=.001in
        nodate
        nonumber
        nobyline;
```

Create a reusable ODS Graphic template that describes the above graph segment, and use PROC SGRENDER to bind the data and render the actual graph. This same GTL template can be used to create every graph on this page because we use the “dynamic QUARTER RESPONSE GROUP FMT TARGET _BYVAL_;” GTL statement that allows us to map physical data set variable names to different dynamic roles in the template.

```
proc template;
define statgraph Quarterly_vbar / store = SASUSER.TEMPLAT;
dynamic QUARTER RESPONSE GROUP FMT TARGET _BYVAL_;
Begingraph;
       DiscreteAttrVar attrvar=GroupAttrs var=GROUP attrmap="AttrMap";
       DiscreteAttrMap name="AttrMap";
              Value "Good" / fillattrs=( color=CXC2E699);
              Value "Satisfactory" / fillattrs=( color=CXFA9FB5);
              Value "Poor" / fillattrs=( color=CXFF0000);
       EndDiscreteAttrMap;
       Entrytitle _BYVAL_;
       if ( exist(FMT) )
          Layout overlay / yaxisopts=(display=(ticks tickvalues)
          linearopts=(tickvalueformat=FMT))
          xaxisopts=(display=(ticks tickvalues));
             Barchart x=QUARTER y=RESPONSE / primary=true stat=sum
             group=GroupAttrs groupdisplay=cluster target=TARGET url=URL
             dataskin=pressed;
          Endlayout;
       else
          Layout overlay / yaxisopts=(display=(ticks tickvalues))
          xaxisopts=(display=(ticks tickvalues));
             Barchart x=QUARTER y=RESPONSE / primary=true stat=sum
             group=GroupAttrs groupdisplay=cluster target=TARGET url=URL
             dataskin=pressed;
          Endlayout;
       endif;
Endgraph;
```

```
      end;
      run;
```

Use the same custom style used in Example 1 to change the background of the PDF page.

```
             ods pdf file="Example2.pdf" notoc style=Styles.Oakwood;
```

To generate the sales dashboard create a gridded layout with 5 columns, and make each of the column uniformly 18.75% of the width of the parent container (in this case the physical page). Tell the layout to advance to the next available region every time you encounter a new by group.

```
             ods layout gridded columns=5
                                   column_widths=(18.75% 18.75% 18.75% 18.75% 18.75%)
                                  advance=bygroup;
```

Using the “product” data set, map the variables in the data set to the dynamic variables defined in the GTL code via the dynamic statement, define the variable (product) you would like to use for your BY processing, and reference the previously built named template (quarterly_vbar). Also notice that there are no REGION statements, and that the advance=bygroup is telling the layout to dynamically populate the grid. Another key point of interest is that there are no sizing arguments for the region, yet the graphs automatically fill the entire REGION container. When a SAS Graph or an ODS Graphic piece of output is contained inside a layout it will be automatically sized to fill the REGION if no overrides have been provided via GOPTIONS or ODS GRAPHIC statements. If the REGION has not specified any sizing criteria, and no sizing overrides have been provided the graphical item will default to the destination specific default graph size.

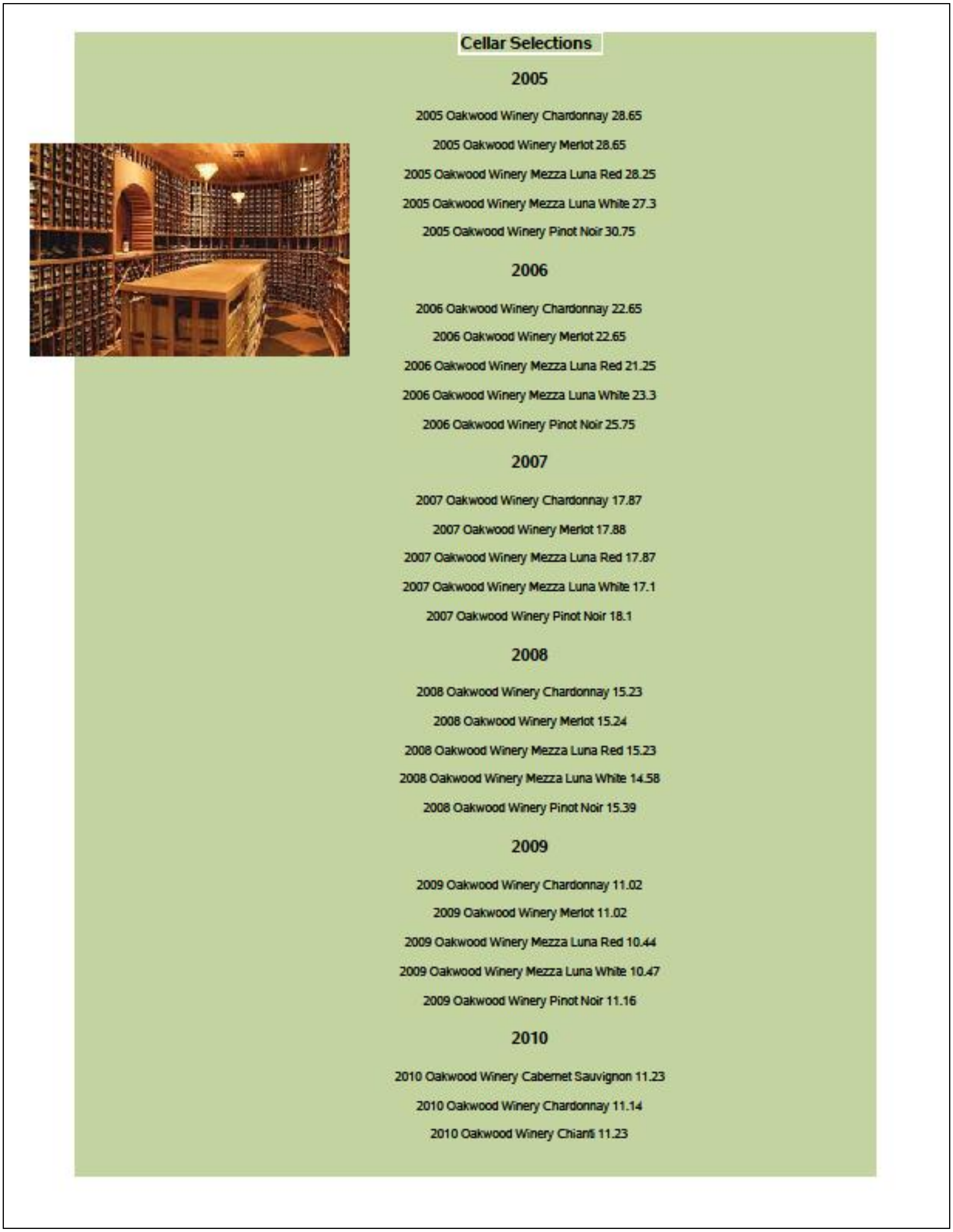
```
             ods graphics / noborder;
             proc sgrender data=product
                           template=quarterly_vbar;
               dynamic QUARTER="quarter"
                       RESPONSE="sales"
                       GROUP="performance"
                       FMT="kdollar."
                      TARGET="target_sales";
               by product;
             run;
```

End the layout and close the PDF file.

```
      ods layout end;
      ods _all_ close;
```

DATA CENTRIC REPORT WRITING INTERFACE (RWI)

In keeping with our data-centric approach of the Sales Dashboard report, ODS Report Writing reports are also most often a data-centric report as well. In our next example we are going to produce a report that shows all of the available wines that we have exclusively aged in our cellar. The data is coming from the inventory data base, and would be updated nightly so that the available wines accurately represent what is available. In this example we are using statement based layout to arrange the output, and using the DATA STEP internal BYGROUP processing to arrange the wines by year.



Cellar Selections

2005

2005 Oakwood Winery Chardonnay 28.65

2005 Oakwood Winery Merlot 28.65

2005 Oakwood Winery Mezza Luna Red 28.25

2005 Oakwood Winery Mezza Luna White 27.3

2005 Oakwood Winery Pinot Noir 30.75

2006

2006 Oakwood Winery Chardonnay 22.65

2006 Oakwood Winery Merlot 22.65

2006 Oakwood Winery Mezza Luna Red 21.25

2006 Oakwood Winery Mezza Luna White 23.3

2006 Oakwood Winery Pinot Noir 25.75

2007

2007 Oakwood Winery Chardonnay 17.87

2007 Oakwood Winery Merlot 17.88

2007 Oakwood Winery Mezza Luna Red 17.87

2007 Oakwood Winery Mezza Luna White 17.1

2007 Oakwood Winery Pinot Noir 18.1

2008

2008 Oakwood Winery Chardonnay 15.23

2008 Oakwood Winery Merlot 15.24

2008 Oakwood Winery Mezza Luna Red 15.23

2008 Oakwood Winery Mezza Luna White 14.58

2008 Oakwood Winery Pinot Noir 15.39

2009

2009 Oakwood Winery Chardonnay 11.02

2009 Oakwood Winery Merlot 11.02

2009 Oakwood Winery Mezza Luna Red 10.44

2009 Oakwood Winery Mezza Luna White 10.47

2009 Oakwood Winery Pinot Noir 11.16

2010

2010 Oakwood Winery Cabernet Sauvignon 11.23

2010 Oakwood Winery Chardonnay 11.14

2010 Oakwood Winery Chianti 11.23

EXAMPLE 3

Despite the fact that the background of the report looks very similar to the previous examples, notice that I am *not* using the custom ODS Style as I did in previous examples. Instead, I am changing the background of the layout to be the same light green color because I want to offset the picture of our wine cellar.

```
ods pdf file="Cellar.pdf" notoc startpage=never;
```

To generate the wine cellar report create a 2 column grid that starts at a horizontal position of 5% of the page, has a width of 90% of the page, and height of 98% of the page.

```
ods layout gridded columns=2 x=5% width=90% height=98%
                   style={background=cxc4d4a1};
```

The first region statement positions us in the first column of the grid, and sizes that column to be 2.5in.

```
ods region width=2.5in;
```

The next statement is another REGION statement which positions us in the second column of the grid. Notice no output is contained in column 1 yet it has a fixed size of 2.5in. This was done to offset the output that will be generated by the RWI in REGION 2 so that it will not collide with the cellar photo.

```
ods region;
```

The TITLE statement will be positioned inside of Region 2, and will be positioned prior to beginning the nested layout. A background color was provided for the title because the default title background color is inherited from the page background color which is white.

```
title "(*ESC*){style [background=cxc4d4a1]Cellar Selections}";
```

Another very powerful feature is the use of nested layouts. This nested layout is completely contained within REGION 2, however, notice that there is no width specified for REGION 2. Also note that there is no size for the nested layout or region, so how is this REGION 2 sized you may ask? The outermost LAYOUT specified a width of 90% of the page, and REGION 1 has a fixed size of 2.5in, so REGION 2 is given the remaining available space (90% of the page minus 2.5in). So the nested layout is bound and centered within the calculated width of REGION 2. The most common mistakes is that programmers feels compelled to always specify heights and widths on everything which generally makes your program significantly more difficult to maintain when changes are necessary. LAYOUT has a very robust sizing algorithm that considers the page dimensions or possibly the parent REGION dimensions in this case and adjusts accordingly. In the absence of any sizing dimensions the REGION container will size itself to accommodate the maximum size of the output which is contained by the REGION. When the content of a REGION does not fit the text is wrapped, a table is broken up into panels, and a graph would be resized to fit within the REGION.

```
ods layout gridded style={background=cxc4d4a1};
ods region;
```

As we previously discussed the Report Writing Interface is part of the DATA STEP, and surfaces advanced custom report writing features. These new features become incredibly powerful when combined with the programming features of the DATA STEP. Use the WHERE clause to include only the wines that are older than 2011. Group the wines by year using the BY statement. For a detailed explanation of the DATA step BY-group processing, please see the “BY-Group Processing in SAS Programs” chapter in the *SAS Language Reference: Concepts*.

```
data _null_;
set wine_cellar(where=(year < 2011)) end=eof;
by year;
if _n_ eq 1 then do;
   dcl odsout obj();
end;
```

The DATA step identifies the beginning and ending of each BY group by creating additional temporary variables: FIRST.<*BY-group variable*> and LAST.< BY-group variable>. During our observation loop processing if we are on the first observation of a BY group, the value of FIRST.<by group variable> will be set to 1. Likewise during our observation loop processing if we are on the last observation of a BY group, LAST.<BY-group variable> will be set to

1. For the purposes of our example when FIRST.YEAR=1, we tell ODS to print the year as text, and start a table that has no borders. Beginning in SAS® 9.4 we have 2 new optional arguments STYLE_ELEM and the STYLE_ATTR that are available on most method statements. These options directly correspond to the <style element> [<style attributes>] rich text syntax that we have seen throughout ODS. In releases prior to SAS® 9.4 these options were called STYLE and OVERRIDE, and were optionally supported on some methods. These old options are aliased to their respective production options, and may be deprecated in the future.

```
if first.year eq 1 then do;
   obj.format_text(data: year,
                    style_elem: "Systemtitle",
                    style_attr: " background=cxc4d4a1");
   obj.table_start(style_attr: 'frame=void
                                rules=none backgroundcolor=_undef_');
end;
```

For each observation loop of the DATA step a ROW and CELL will be created that contain the concatenated values of variables YEAR, BRAND, VARIETAL, and PRICE.

```
obj.row_start();
   obj.format_cell(data: year || ' ' || strip(brand) || ' ' ||
                         strip(varietal) || ' ' || strip(price) );
obj.row_end();
```

When we have identified the last observation in the BYGROUP (i.e. LAST.YEAR=1) we close the table by calling the TABLE_END method, before looping back to the top of the DATA step to retrieve the next observation.

```
if last.year eq 1 then do;
  obj.table_end();
end;
run;
```

Close the nested LAYOUT.

```
ods layout end;
```

Close the outermost LAYOUT. Until this point we have not discussed how the physical rendering to the page actually occurs, but this is very important. A GRIDDED LAYOUT is not rendered onto the physical page until the outermost layout is closed, so at this point we actually allow all of the stored drawing instructions to be flushed.

```
ods layout end;
```

Start a new LAYOUT which is purposely positioned to be directly on top of the first layout with a different X and Y position. By understanding the rendering order you can effectively create layers of content that overlap as we have done with this simple LAYOUT that only contains a picture of the cellar.

```
ods layout gridded x=0 y=1in;
ods region;
data _null_;
dcl odsout obj();
  obj.image(file: "Images/cellar1.jpeg");
run;
ods layout end;
ods _all_ close;
```

GRIDDED LAYOUT ORDERING

When dealing with a two-dimensional grid we have the opportunity to populate the grid in 2 ways. You populate the grid horizontally by populating each column in row 1 before proceeding to row 2 which is known as a row-major order.

1	2	3
4	5	6

Or you populate the grid in a vertically by populating each row in column 1 before proceeding to column 2 which is known as a column-major order.

1	3	5
2	4	6

The ORDER_TYPE gridded layout option controls the order in which to populate the grid.

ORDER_TYPE = < ROW_MAJOR (default) | COLUMN_MAJOR >

ROW_MAJOR

When using the ROW_MAJOR option you must have a fixed number of columns so that the layout knows when to proceed to the next row, but you can have an infinite number of rows. If the layout does not have an explicitly defined number of columns the layout will default to 1 column.

Valid Destinations: **HTML, PRINTER (PDF, PCL, PSL)**

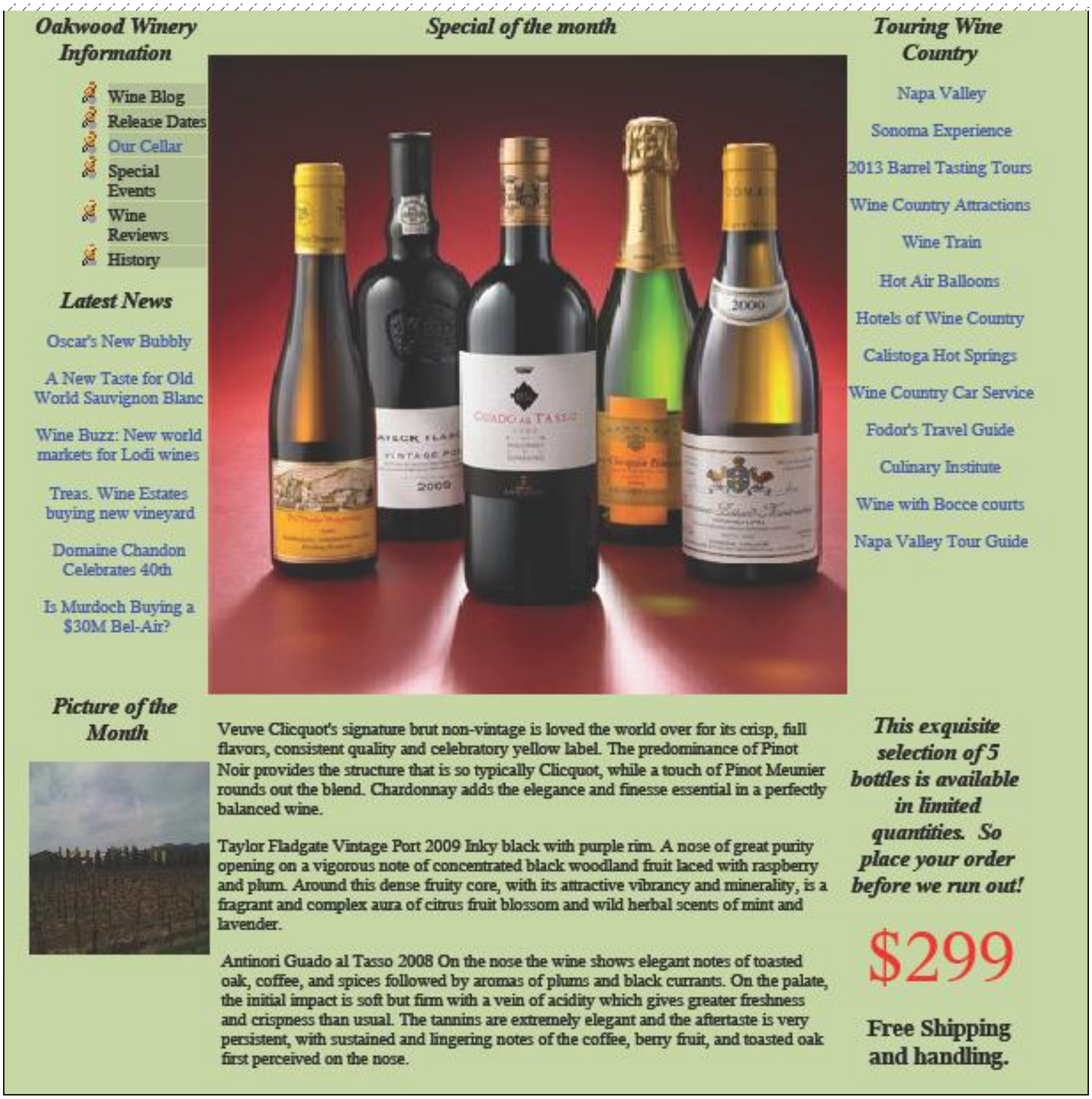
COLUMN_MAJOR

When using the COLUMN_MAJOR option you must have a fixed number of rows so that the layout knows when to proceed to the next column, but you can have an infinite number of columns. If the layout does not have an explicitly defined number of rows the layout will default to 1 row.

Valid Destinations: **PRINTER (PDF, PCL, PSL)**

STRUCTURED TEXT AND LIST BLOCKS

In SAS® 9.4 we have 2 new production procedures, PROC ODSLIST and PROC ODSTEXT. These allow you to create bulleted lists and formatted blocks of text in your reports. These lists and text blocks can be static content (i.e. strings) or dynamic content based on the information in a data set. PROC ODSTEXT is a much more powerful way of managing your text than using the "ods text=" statement that you have seen in previous examples. We are going to pick up with the second half of the Oakwood Winery's home page example. In this portion of the report I exclusively use these procedures to generate all the content within LAYOUT.



Oakwood Winery Information

- Wine Blog
- Release Dates
- Our Cellar
- Special Events
- Wine Reviews
- History

Latest News

Oscar's New Bubbly

A New Taste for Old World Sauvignon Blanc

Wine Buzz: New world markets for Lodi wines

Treas. Wine Estates buying new vineyard

Domaine Chandon Celebrates 40th

Is Murdoch Buying a $30M Bel-Air?

Picture of the Month

Special of the month

Veuve Clicquot's signature brut non-vintage is loved the world over for its crisp, full flavors, consistent quality and celebratory yellow label. The predominance of Pinot Noir provides the structure that is so typically Clicquot, while a touch of Pinot Meunier rounds out the blend. Chardonnay adds the elegance and finesse essential in a perfectly balanced wine.

Taylor Fladgate Vintage Port 2009 Inky black with purple rim. A nose of great purity opening on a vigorous note of concentrated black woodland fruit laced with raspberry and plum. Around this dense fruity core, with its attractive vibrancy and minerality, is a fragrant and complex aura of citrus fruit blossom and wild herbal scents of mint and lavender.

Antinori Guado al Tasso 2008 On the nose the wine shows elegant notes of toasted oak, coffee, and spices followed by aromas of plums and black currants. On the palate, the initial impact is soft but firm with a vein of acidity which gives greater freshness and crispness than usual. The tannins are extremely elegant and the aftertaste is very persistent, with sustained and lingering notes of the coffee, berry fruit, and toasted oak first perceived on the nose.

Touring Wine Country

Napa Valley

Sonoma Experience

2013 Barrel Tasting Tours

Wine Country Attractions

Wine Train

Hot Air Balloons

Hotels of Wine Country

Calistoga Hot Springs

Wine Country Car Service

Fodor's Travel Guide

Culinary Institute

Wine with Bocce courts

Napa Valley Tour Guide

This exquisite selection of 5 bottles is available in limited quantities. So place your order before we run out!

$299

Free Shipping and handling.

EXAMPLE 4

Create a gridded layout with 3 columns, 2 rows, and populate the layout in a COLUMN_MAJOR order fashion which means to fill all of colum1 before proceeding to column 2. Additionally set all 3 column widths to be 1.4in, 5in, and 1.5in, respectively.

```
        ods layout gridded x=.1in
                            columns=3
                            rows=2
                            order_type=column_major
                            column_widths=(1.4in 5in 1.5in);
```

Navigate to the first position in the grid which is row 1 and column 1, and render the ODSLIST and ODSTEXT procedures output. A common misconception is that each procedure must be contained within its own region (container) which is not true. In the first region we have 2 pieces of PROC output each containing its own unique TITLE information respectively.

```
ods region;
title "Oakwood Winery Information";
proc odslist;
cellstyle 1 as {backgroundcolor=cxafbd91};
item "Wine Blog" / style={liststyleimage='Images/tack.gif'};
item "Release Dates" / style={liststyleimage='Images/tack.gif'};
item "Our Cellar"  / style={url='Cellar.pdf'
                            linkcolor=cxc4d4a1
                            color=blue
                            liststyleimage='Images/tack.gif'};
item "Special Events" / style={liststyleimage='Images/tack.gif'};
item "Wine Reviews" / style={liststyleimage='Images/tack.gif'};
item "History" / style={liststyleimage='Images/tack.gif'};
run;
```

In the ODSTEXT example below I wanted to compare 2 different rich text approaches to providing style attribute customization. We have seen that stylistic customization can be performed on virtually any type of output that ODS produces. In previous examples you have seen me use the ESCAPECHAR place holder (*ESC*) approach. I also wanted to introduce how ODSTEXT and ODSLIST have a much simpler approach to providing stylistic customization through the use of the optional STYLE option. In the following code the first line of text uses the ESCAPECHAR place holder (*ESC*) approach, and the rest of the lines use the STYLE option approach.

```
title "Latest News";
proc odstext;
p "(*ESC*){style [url='http:<url path>
                 linkcolor=cxc4d4a1
                 color=blue]Oscar's New Bubbly}";
p "A New Taste for Old World Sauvignon Blanc" / style={url=<url path>
                                                      linkcolor=cxc4d4a1
                                                      color=blue
                                                      just=center};
p "Wine Buzz: New world markets for Lodi wines" / style={...};
p "Treas. Wine Estates buying new vineyard" / style ={...};
p "Domaine Chandon Celebrates 40th}" / style={...};
p "Is Murdoch Buying a $30M Bel-Air?" /style={...};
run;
```

Navigate to the next available position in the grid which is row 2 and column 1. Remember we are doing COLUMN_MAJOR grid navigation so we continue to populate all rows in column 1 until we have reached the maximum number of rows before advancing to column 2. We are going to generate the new title in the region prior to RWI (Picture of the Month) image. Typically when you think of output you are normally thinking of PROC tabular or graphic output, however the same rules apply for the RWI beginning in SAS® 9.4.

```
ods region;
title "Picture of the Month";
data _null_;
dcl odsout obj();
obj.image(file: "Images/PictureOfTheMonth.jpeg",
          width: "1.5in",
          height: "1.5in");
run;
```

Navigate to the next available position in our grid which is row 1 and column 2 because we have reached the maximum number of rows in the grid. We are once again rendering a new title and the RWI (Monthly Special) image.

```
ods region;
Title "Special of the month";
data _null_;
```

```
        dcl odsout obj();
        obj.image(file: "Images/MonthlySpecial.jpeg",
                  width: "5in",
                  height: "5in");
        run;
```

Navigate to the next available position in the grid which is now row 2 and column 2. Render the ODSTEXT output that contains very long text. Notice that all of this text is intuitively wrapped within the maximum region width of 5in.

```
        ods region;
        proc odstext;
        p "Veuve Clicquot's signature brut non-vintage is loved ...";
        p "Taylor Fladgate Vintage Port 2009 Inky black with ...";
        p "Antinori Guado al Tasso 2008 On the nose the wine shows ...";
        run;
```

Navigate to the next available position in the grid which is now row 1 and column 3. Render the new title and the ODSTEXT (Touring Wine Country) output.

```
        ods region;
        title "Touring Wine Country";
        proc odstext;
        p "Napa Valley" / style={...};
        p "Sonoma Experience" / style={...};
        p "2013 Barrel Tasting Tours" / style={...};
        p "Wine Country Attractions" / style={...};
        p "Wine Train" / style={...};
        p "Hot Air Balloons" / style={...};
        p "Hotels of Wine Country" / style={...};
        p "Calistoga Hot Springs" / style={...};
        p "Wine Country Car Service" / style={...};
        p "Fodor's Travel Guide" / style={...};
        p "Culinary Institute" / style={...};
        p "Wine with Bocce courts" / style={...};
        p "Napa Valley Tour Guide" / style={url=...};
        p "";
         p "";
        run;
```

Navigate to the last position in the 3 x 2 grid which is row 2 and column 3. Render the very long title and the ODSTEXT output. Notice that the very long title is intuitively wrapped within the maximum region width of 1.5in similar to the ODSTEXT in grid location row 2 and column 2.

```
        ods region;
        title "This exquisite selection of 5 bottles is available in limited
        quantities.  So place your order before we run out!";
        proc odstext;
        p "$299" / style={font_size=42pt color=red};
        p "Free Shipping and handling." / style={font_size=14pt font_weight=bold};
        run;
```

Close the layout.

```
        ods layout end;
```

CONCLUSION

As you can see the “ODS Crown Jewels” advancements in SAS® 9.4 provide a rich set of features to create elaborate visualizations that combine multi-view dashboards in a manner that is easy to digest. Whether the reports are being printed, viewed on the web, or on a mobile device these new features will provide a novice to advanced programmer an easy way to create sophisticated publications.

REFERENCES

O’Connor, Daniel and Huntley, Scott SAS Global Forum Paper 043-2009: Breaking New Ground with SAS® 9.2 ODS Layout Enhancements

O’Connor, Daniel SAS Global Forum Paper: The Power to Show: Ad Hoc Reporting, Custom Invoices, and Form Letters

ACKNOWLEDGMENTS

I would like to thank my colleagues for assisting me with a last minute editing.

David Kelley

Tim Hunter

Scott Huntley

Wayne Hester

Allison Crutchfield

Daniel Kummer

SAMPLE SOURCE CODE

Sample source code for all examples can obtained here.

RECOMMENDED READING

- *Base SAS® Procedures Guide*
- *SAS® Output Delivery Systems*

CONTACT INFORMATION

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

Daniel O’Connor
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
(919) 531-6171
Dan.O’Connor@sas.com