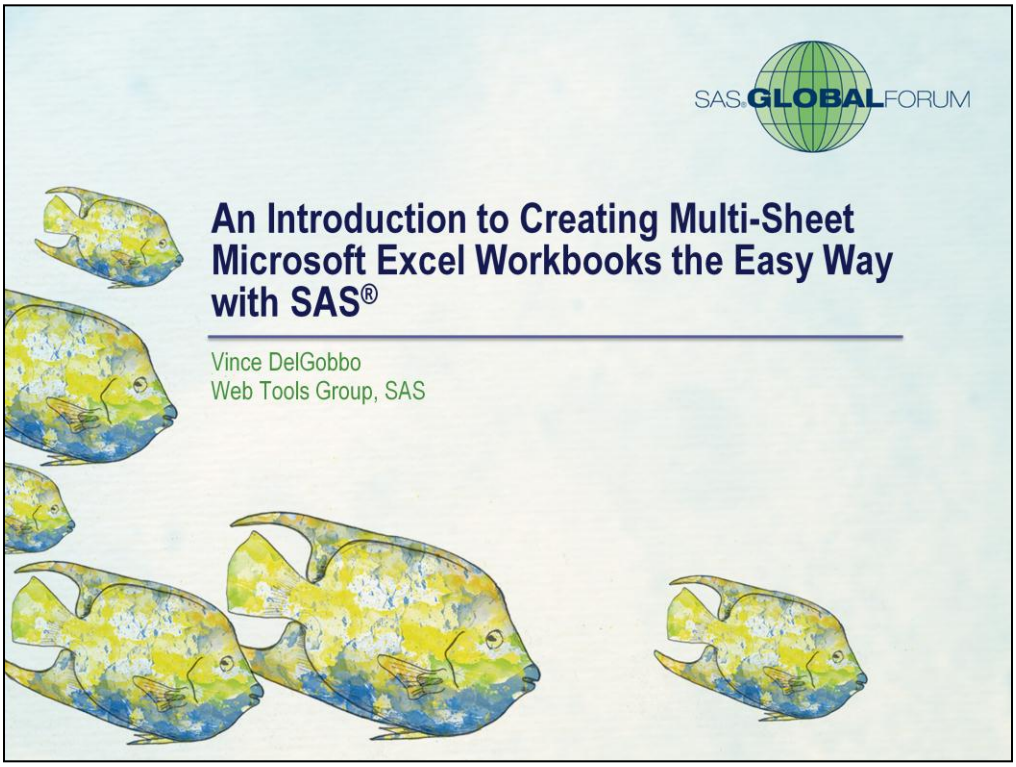


SAS Global Forum 2012
April 22 – 25, 2012
Walt Disney World Swan and Dolphin Resort
Orlando, FL

A special "thank you" to Ann Stephan and Maribeth Johnson for inviting me to present this topic, and to Chris Barrett of SAS Institute Inc. for his valuable contributions to the accompanying paper.

Goals

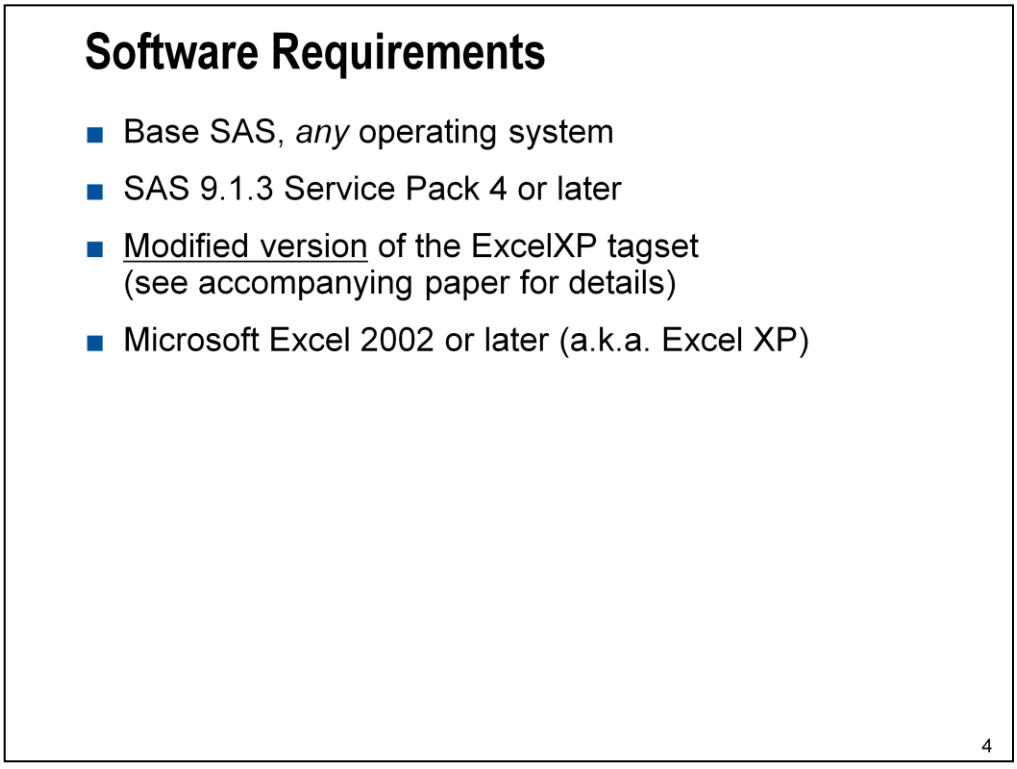
- Integrate SAS output w/ Excel
- Give you something you can use TODAY

2

Agenda

- Background information
- ODS basics
- Generating XML for Excel
- Opening output in Excel
- Fix formatting

3



The version of the ExcelXP tagset that was shipped with Base SAS®9 has undergone many revisions since its initial release. To take advantage of the features discussed in this paper, you must download an updated version of the tagset and install it on your system.

Refer to the accompanying paper for details.

ODS ExcelXP Tagset

http://support.sas.com/rnd/base/ods/odsmarkup/

General Steps

1. Run SAS code to create output
2. Store output where Excel can access it
3. Open output with Excel
4. Modify SAS code to correct formatting problems

5

We use ODS to create an XML file that is stored in a location where Excel can access it. In your production system, SAS and Excel may reside on two different systems. Thus, you may have to make use of network drives, FTP or some other means to move the SAS output to a location that Excel can access.

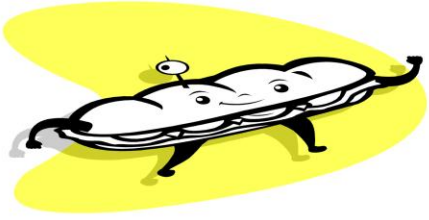
Then the ODS output is opened using Excel. If you have ever done this before, you have probably encountered formatting problems. We fix those problems, and then explore techniques to instruct ODS to create output that is more attractive.

ODS Basics

- Part of Base SAS
- Easily generate multiple output types (HTML, RTF, PDF, XML, etc.)
- A "destination" creates the actual output
- A "style" controls the appearance
- Usage:

HTML or RTF or PDF ...

```
ods DestName style=StyleName file=... ;
  *  Your SAS procedure code here;
ods DestName close;
```



6

ODS is the part of Base SAS software that enables you to generate different types of output from your procedure code. An ODS *destination* controls the type of output that is generated (HTML, RTF, PDF, etc.). An ODS *style* controls the appearance of the output (colors, fonts, border lines, etc.).

Both a destination and a style are needed to generate output. If you do not specify a style, a default style is used.

We use a special type of ODS destination called a "tagset". ODS tagsets can be modified to meet your specific needs using the TEMPLATE procedure, which is part of the Base SAS product. And you can use the TEMPLATE procedure to create your own tagsets.

TEMPLATE Procedure Overview

SAS 9.1: http://tinyurl.com/3mnkxpg

SAS 9.2: http://tinyurl.com/3hkglxw

SAS 9.3: http://tinyurl.com/3tycrgl

ODS Basics – Output for Excel

- Excel can open specially made XML files as multi-sheet workbooks (graphics not supported)
- Use the ExcelXP tagset and Printer style:

```
ods listing close;
ods tagsets.ExcelXP style=Printer
 file=... ;
 *  Your SAS procedure code here;
ods tagsets.ExcelXP close;
```

7

We use the ODS tagset named ExcelXP to create XML output that can be opened using Microsoft Excel 2002 and later. When opened with Excel, the XML file is rendered as a multi-sheet Excel workbook. Additionally, we use an ODS style named Printer, supplied by SAS, to control the appearance of the output.

Because the ExcelXP ODS tagset creates files that conform to the Microsoft XML Spreadsheet Specification, you can create multi-sheet Excel workbooks containing the output from almost any SAS procedure. The exception is that the Microsoft XML Spreadsheet Specification does not support images, so the output from SAS/GRAPH® software procedures cannot be used.

Microsoft XML Spreadsheet Reference

http://tinyurl.com/3p6sked

Sample SAS Code

```
title 'The CLASS Dataset';
footnote '(From the SASHELP library)';

proc print data=sashelp.class noobs;
  where (sex eq 'M');
  var name age height weight;
run; quit;

proc print data=sashelp.class noobs;
  where (sex eq 'F');
  var name age height weight;
run; quit;
```

8

The PRINT procedure is run once to create a worksheet containing data for the male students, and then again to create a worksheet for female students.

We want the information for the male and female students to be in separate worksheets, and the titles and footnotes included within the worksheet body.

SAS Listing Output

Name	Age	Height	Weight
Alfred	14	69.0	112.5
Henry	14	63.5	102.5
James	12	57.3	83.0
...			

Name	Age	Height	Weight
Alice	13	56.5	84.0
Barbara	13	65.3	98.0
Carol	14	62.8	102.5
...			

9

Here is an abbreviated view of PROC PRINT output viewed in the SAS Display Manager Output window.

Titles and footnotes omitted for brevity.

Note that the height and weight values are formatted to display one decimal place.

Using ODS and the ExcelXP Tagset

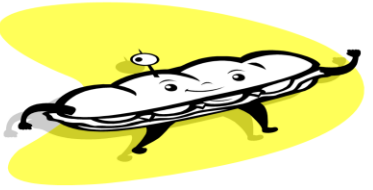
```
title 'The CLASS Dataset';
footnote '(From the SASHELP library)';

proc print data=sashelp.class noobs;
  where (sex eq 'M');
  var name age height weight;
run; quit;

proc print data=sashelp.class noobs;
  where (sex eq 'F');
  var name age height weight;
run; quit;
```

10

Review of the SAS code.

Using ODS and the ExcelXP Tagset

```
ods tagsets.ExcelXP file='MyWorkbook.xml'
  style=Printer;

  title 'The CLASS Dataset';
  footnote '(From the SASHELP library)';

  proc print data=sashelp.class noobs;
    where (sex eq 'M');
    var name age height weight;
  run; quit;

  proc print data=sashelp.class noobs;
    where (sex eq 'F');
    var name age height weight;
  run; quit;

ods tagsets.ExcelXP close;
```



11

This is our sample code with the ODS statements included.

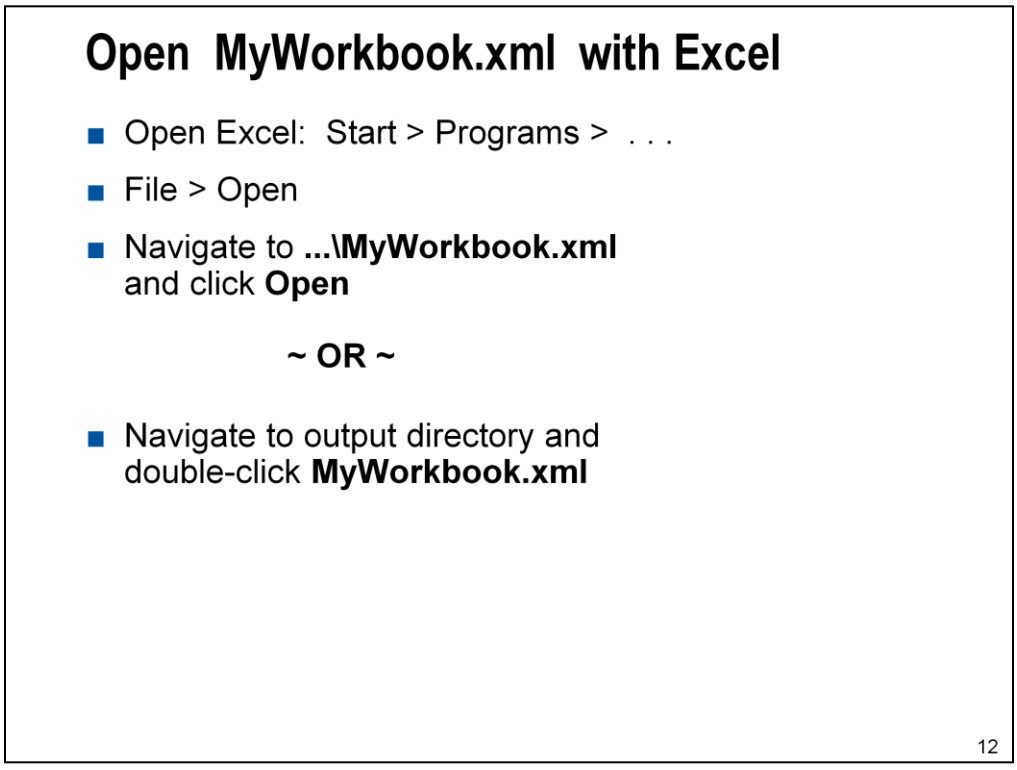
An XML file named "MyWorkbook.xml" is created, and the Printer style, supplied with Base SAS, controls the appearance.

NOTE: You can specify any name you like for the file name, but you should use the "xml" extension instead of "xls" or "xlsx", because Excel 2007 and 2010 display a warning if the "xml" extension is not used. This new feature of Excel 2007 is called "Extension Hardening".

The output from each execution of the PRINT procedure code appears in a separate worksheet.

Reference

http://support.microsoft.com/kb/948615



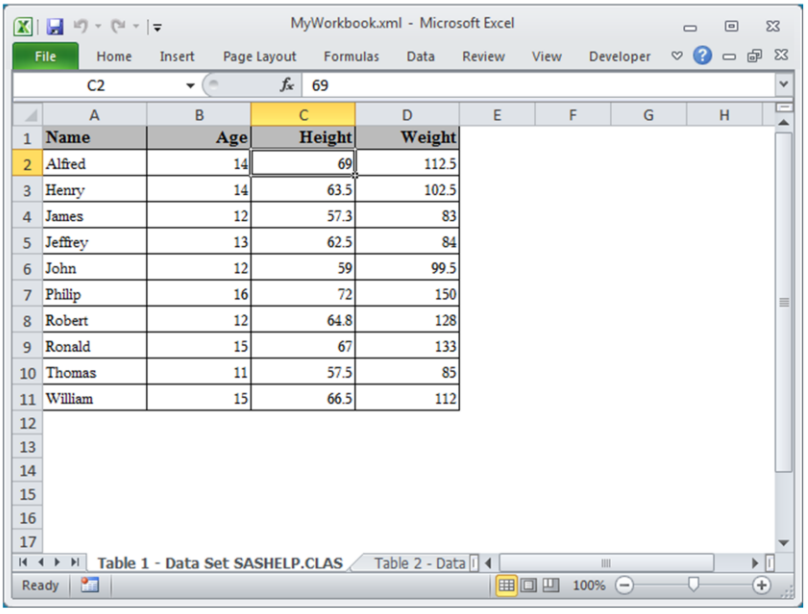
The XML file created by ODS can now be opened with Microsoft Excel.

Excel reads and converts the XML file to the Excel format. After the conversion, you can perform any Excel function on the data.

To save a copy of the file in Excel binary (xls) format using Excel 2002, 2003 or 2010, select **File ➔ Save As** and then, from the **Save as type** drop-down list, select **Microsoft Excel Workbook (*.xls)**. If you're using Excel 2007, click the Microsoft Office Button, and then select **Save As ➔ Excel 97-2003 Workbook**.

If you're using Excel 2007 or 2010 and want to save the document in the Microsoft Office Open XML format, choose **Excel Workbook (*.xlsx)** from the **Save as type** drop-down list.

MyWorkbook.xml Viewed with Excel



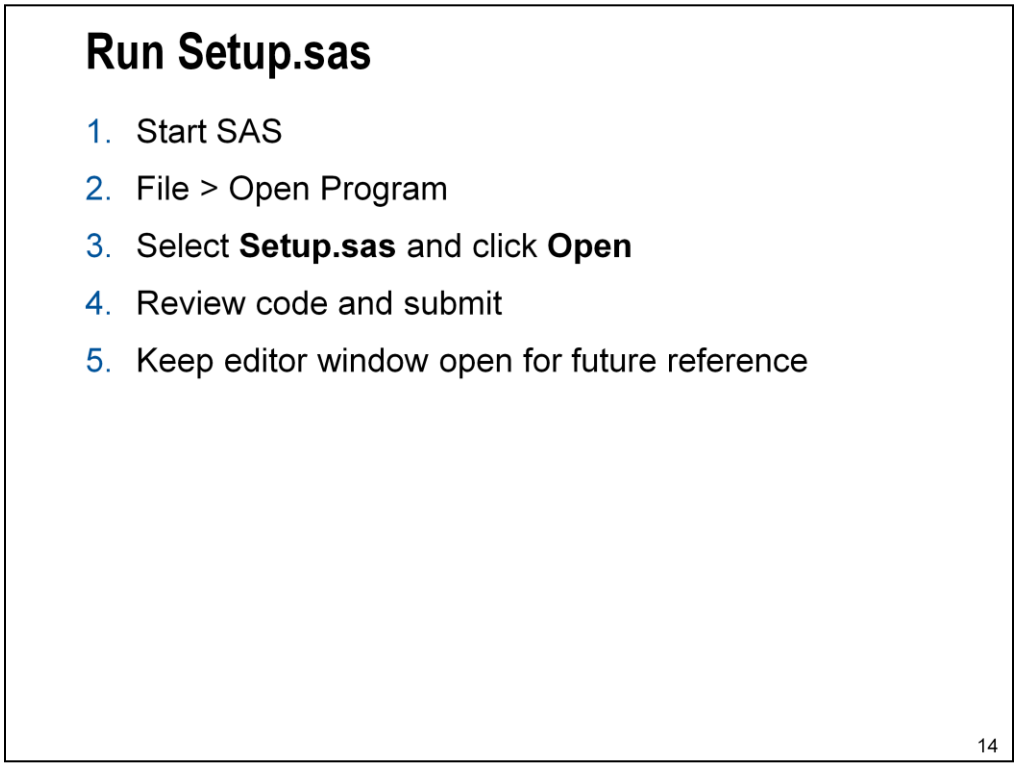
	A	B	C	D
1	Name	Age	Height	Weight
2	Alfred	14	69	112.5
3	Henry	14	63.5	102.5
4	James	12	57.3	83
5	Jeffrey	13	62.5	84
6	John	12	59	99.5
7	Philip	16	72	150
8	Robert	12	64.8	128
9	Ronald	15	67	133
10	Thomas	11	57.5	85
11	William	15	66.5	112

13

There are some problems with the Excel workbook:

1. Unattractive, default worksheet names are used.
2. Title and footnote text is missing.
3. We would like to have an Excel AutoFilter on column B.
4. The justification of column heading text is inconsistent.
5. Not all of the data values for height and weight are displayed with 1 decimal place.

We can now change the basic SAS code to correct these issues.



The SAMPDIR global macro variable specifies the directory containing our sample code and data, as well as the ODS-generated XML output.

The program assigns a SAS library (SAMPLE) for the input data and one (MYLIB) for the compiled ODS tagset. Although you can temporarily store the tagset in the WORK or SASUSER libraries, it is more efficient to compile it once, and then store it in a permanent library so that you can reference it in other SAS programs.

The ODS PATH statement specifies the locations of, and the order in which to search for, ODS tagsets and styles. Notice that the access mode for `mylib.tmplmst` is specified as "update" and it is first in the search path as a result of the "prepend" option.

Because ODS searches the path in the order given, and the access mode for `mylib.tmplmst` is "update", PROC TEMPLATE, compiles and stores the tagset in a file named "tmplmst.sas7bitm" in the directory that is associated with the "mylib" library.

The ODS ExcelXP tagset has undergone many revisions since SAS 9 was released, so we import a newer copy and store it in the MYLIB library. Be sure to use a recent version of the ExcelXP tagset in your production code. (See the accompanying paper for details.)

Ex. 1 – Create the Initial Workbook

1. File > Open Program > **Exercise1.sas**
2. Follow TO DO instructions and submit code
3. View output in Excel:

   Start > Programs > ...

   File > Open > **C:\HOW\DelGobbo\MyWorkbook.xml**

4. Close the document (child) window (leave Excel running, but minimize it)

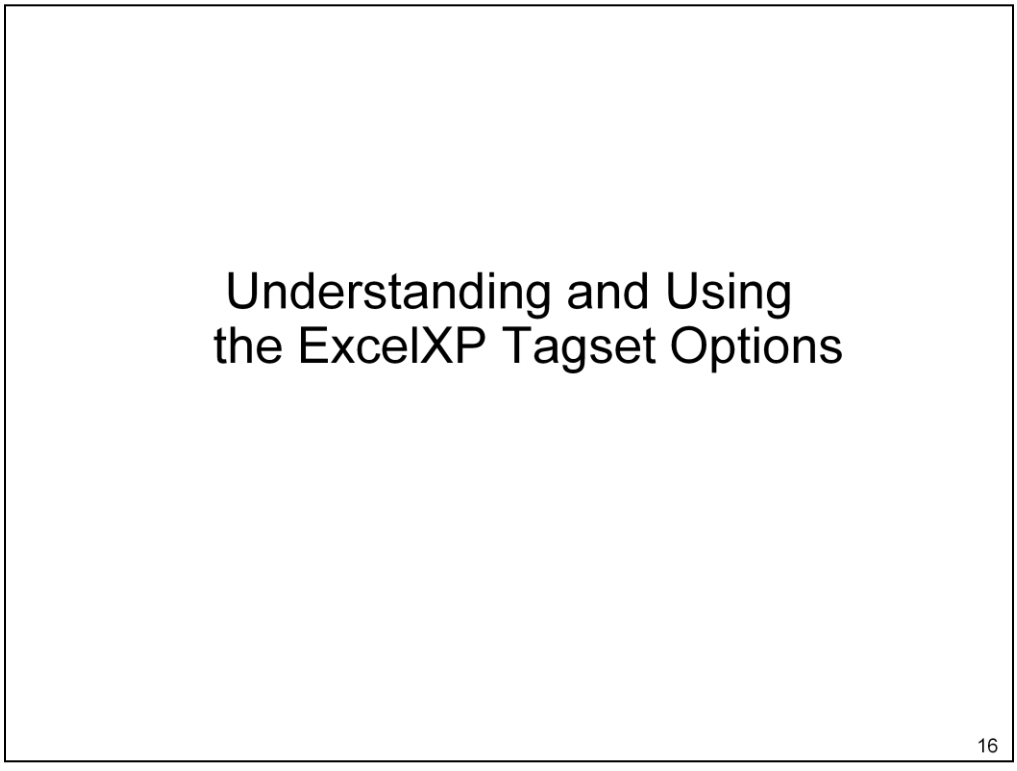
15

TO DO:

Lines 7 and 22: Complete the ODS sandwich.

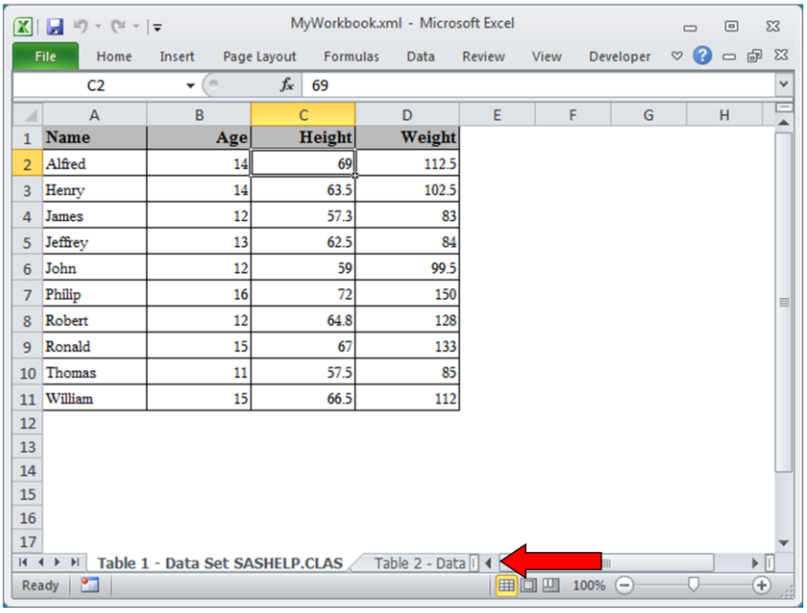
Solution:

```
ods tagsets.ExcelXP path="&SAMPDIR" file='MyWorkbook.xml'
    style=Printer;

*  PROC PRINT code here;

ods tagsets.ExcelXP close;
```



The following techniques apply to all SAS procedure output.

Supply Your Own Worksheet Names



	A	B	C	D
1	Name	Age	Height	Weight
2	Alfred	14	69	112.5
3	Henry	14	63.5	102.5
4	James	12	57.3	83
5	Jeffrey	13	62.5	84
6	John	12	59	99.5
7	Philip	16	72	150
8	Robert	12	64.8	128
9	Ronald	15	67	133
10	Thomas	11	57.5	85
11	William	15	66.5	112

17

ODS generates a unique name for each worksheet, as required by Excel. However, the names are generally not very attractive. There are, however, several tagset options that you can use to alter the names of the worksheets.

ExcelXP Supports Tagset Options

- Syntax: `options(option-name='option-value')`
- Can control the worksheet name:

```
options(sheet_name='worksheet-name');
```

- Can have multiple ODS statements
- Options remain in effect until changed

18

The ExcelXP tagset supports many options that control both the appearance and functionality of the Excel workbook. Most of these tagset options are simply tied directly to existing Excel options or features. Tagset options are specified on the ODS statement using the OPTIONS keyword, as shown above.

ODS automatically generated unique worksheet names, but we can use the SHEET_NAME option to explicitly specify a worksheet name.

IMPORTANT NOTE: Tagset options remain in effect until they are changed !

Supply Your Own Worksheet Names

```
ods tagsets.ExcelXP style=Printer file= ... ;
  title ...; footnote ...;
  ods tagsets.ExcelXP options(sheet_name='Male
    Students');
  proc print ...;
    where (sex eq 'M');
    ... ;
  run; quit;
  ods tagsets.ExcelXP options(sheet_name='Female
    Students');
  proc print ...;
    where (sex eq 'F');
    ... ;
  run; quit;
ods tagsets.ExcelXP close;
```

19

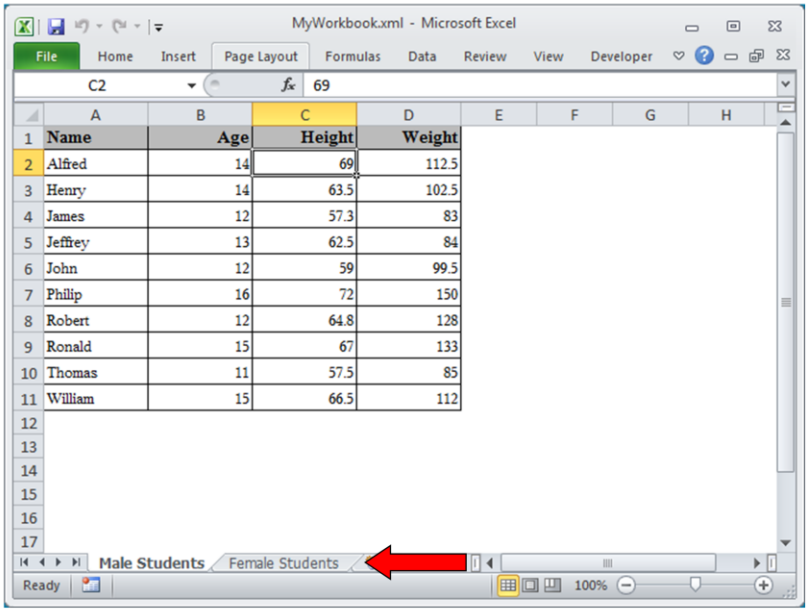
Two new ODS statements were added to specify the values for the worksheet names.

When specifying *additional* ODS statements as shown here, do not specify the `file`, `style`, or any other keyword or option that is supported by ODS. Those options should be specified only on the initial ODS statement. Only specify the name of the destination (`tagsets.ExcelXP`) and the OPTIONS keyword.

The first worksheet is named "Male Students" and contains data for the male students. Similarly, the second worksheet is named "Female Students" and contains information for female students.

Recall that tagset options remain in effect until the ExcelXP destination is closed. We specify the option twice because we want a different name for each worksheet.

Supply Your Own Worksheet Names



	A	B	C	D
1	Name	Age	Height	Weight
2	Alfred	14	69	112.5
3	Henry	14	63.5	102.5
4	James	12	57.3	83
5	Jeffrey	13	62.5	84
6	John	12	59	99.5
7	Philip	16	72	150
8	Robert	12	64.8	128
9	Ronald	15	67	133
10	Thomas	11	57.5	85
11	William	15	66.5	112

20

When we open the XML file with Excel, we see the worksheet names that we specified in the SHEET_NAME option, instead of those automatically generated by ODS.

Ex. 2 – Supply Worksheet Names

1. Go to SAS
2. File > Open Program > **Exercise2.sas**
3. Follow TO DO instructions and submit code
4. Go to Excel
5. File > **\HOW\DelGobbo\MyWorkbook.xml**
   - or -
   MyWorkbook.xml (from the recent file list)
6. Close the document (child) window (leave Excel running, but minimize it)

21

TO DO:

Lines 13 and 20: Add option to change the worksheet names.

Solution:

```
ods tagsets.ExcelXP options(sheet_name='Male Students');
ods tagsets.ExcelXP options(sheet_name='Female Students');
```

Display Titles & Footnotes in Worksheet

- Title text → Excel print header
- Footnote text → Excel print footer
- Can control location of title & footnote text:

```
options(embedded_titles='yes'
        embedded_footnotes='yes')
```

22

By default, SAS titles and footnotes appear as Excel print headers and print footers, respectively, which are displayed when the Excel document is printed. You can confirm this by viewing the Excel "Header/Footer" tab of the "Page Setup" dialog box.

To include title and footnote text on-screen, in the worksheet body, use the EMBEDDED_TITLES and EMBEDDED_FOOTNOTES options.

Display Titles & Footnotes in Worksheet

```
ods tagsets.ExcelXP style=Printer file= ... ;
  title ...; footnote ...;
  *  Set some "global" tagset options;
  ods tagsets.ExcelXP
    options(embedded_titles='yes'
            embedded_footnotes='yes');
  ods tagsets.ExcelXP options(sheet_name=...);
  proc print ...; run; quit;
  ods tagsets.ExcelXP options(sheet_name=...);
  proc print ...; run; quit;
ods tagsets.ExcelXP close;
```

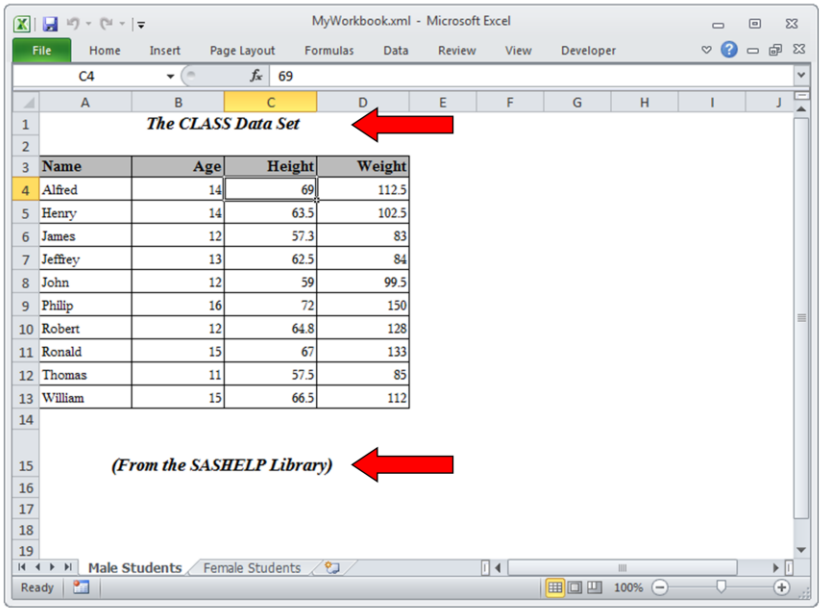
23

The tagset options could have been added to the original ODS statement.

Instead, a new statement is issued in order to increase readability of the slide and the SAS code, and also to isolate "global" tagset options – those that affect all worksheets.

Because tagset options remain in effect until their value is changed or the destination is closed, the EMBEDDED_TITLES and EMBEDDED_FOOTNOTES options affect both worksheets.

Display Titles & Footnotes in Worksheet



Name	Age	Height	Weight
Alfred	14	69	112.5
Henry	14	63.5	102.5
James	12	57.3	83
Jeffrey	13	62.5	84
John	12	59	99.5
Philip	16	72	150
Robert	12	64.8	128
Ronald	15	67	133
Thomas	11	57.5	85
William	15	66.5	112

24

Title and footnote text is included on-screen, in the worksheet body.

Ex. 3 – Display Titles & Footnotes

1. Go to SAS
2. File > Open Program > **Exercise3.sas**
3. Follow TO DO instructions and submit code
4. Go to Excel
5. File > **\HOW\DelGobbo\MyWorkbook.xml**
   - or -
   MyWorkbook.xml (from the recent file list)
6. Close the document (child) window (leave Excel running, but minimize it)

25

TO DO:

Line 15: Add options to embed title and footnote text.

Solution:

```
ods tagsets.ExcelXP options(embedded_titles='yes'
                            embedded_footnotes='yes');
```

Can Also Have Print Headers & Footers

```
options(print_header='header-text'
        print_footer='footer-text')
```

Example:

```
print_header='&C&A&RPage &P of &N'

print_footer='&RPrinted &D at &T'
```

26

Although the name of each worksheet includes the gender presented in the data, this information is not shown when the workbook is printed. You can add Excel print headers and footers to display this information. Excel allows you to define custom print headers and footers from the "Header/Footer" tab of the "Page Setup" dialog box.

You can also control the header and footer from SAS code using the PRINT_HEADER and PRINT_FOOTER tagset options.

Can Also Have Print Headers & Footers

```
print_header='&C&A&RPage &P of &N'
print_footer='&RPrinted &D at &T'
```

Control Sequence	Function
&C	Center text
&A	Insert sheet name
&R	Right-justify text
&P	Insert page number
&N	Insert number of pages
&D	Insert date printed
&T	Insert time printed
&F	Insert file name
&Z	Insert file path

27

In addition to plain text, you can specify Excel control sequences that provide the same functionality as the buttons in the "Page Setup" dialog box. The control sequence always starts with an ampersand ("&"), and is followed by a single character.

For example, use `&A` to insert the worksheet name into the header.

The ampersand and the single character do not appear in your printed output.

Additional control sequences are listed in the accompanying paper.

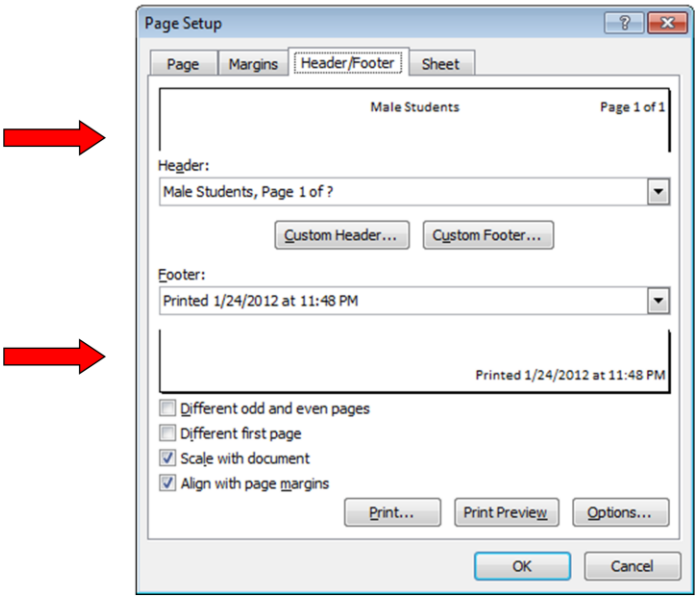
Can Also Have Print Headers & Footers

```
ods tagsets.ExcelXP style=Printer file= ... ;
  title ...; footnote ...;
  *  Set some "global" tagset options;
  ods tagsets.ExcelXP
    options(embedded_titles='yes'
            embedded_footnotes='yes'
            print_header='&C&A&RPage &P of &N'
            print_footer='&RPrinted &D at &T');
  ods tagsets.ExcelXP options(sheet_name=...);
  proc print ...; run; quit;
  ods tagsets.ExcelXP options(sheet_name=...);
  proc print ...; run; quit;
ods tagsets.ExcelXP close;
```

28

Add PRINT_HEADER and PRINT_FOOTER to our "global" tagset options.

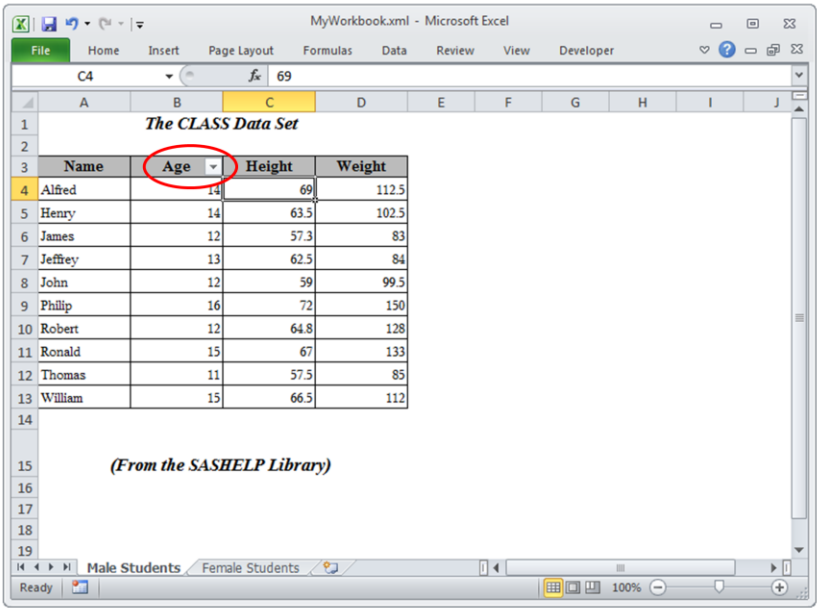
Can Also Have Print Headers & Footers



29

The resulting print headers and footers.

AutoFilters



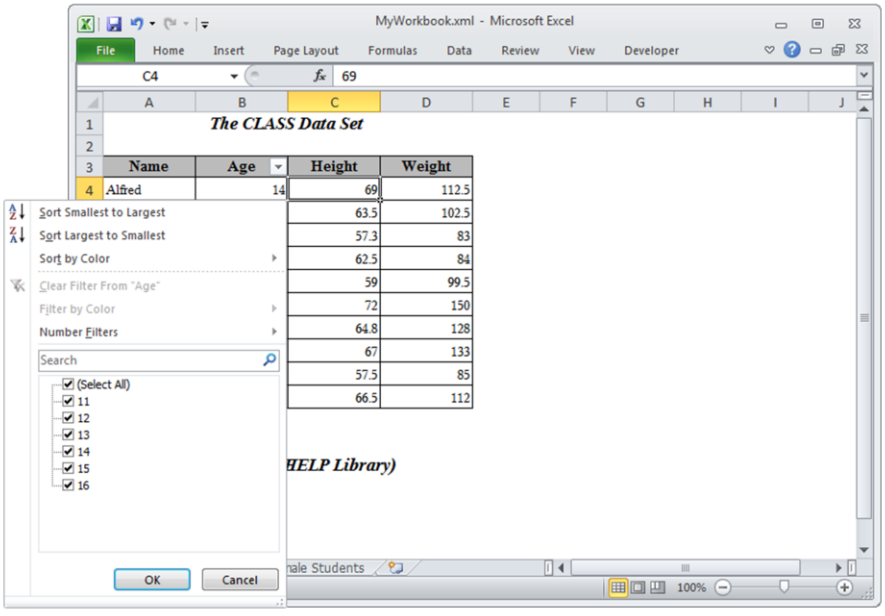
30

An Excel AutoFilter enables you to filter, or subset, the data that is being displayed in a worksheet. A column of data containing an AutoFilter is indicated by an arrow button in the header cell of that column. For example, the "Age" column contains an AutoFilter.

Suppose that you want to view only records for 15-year old students.

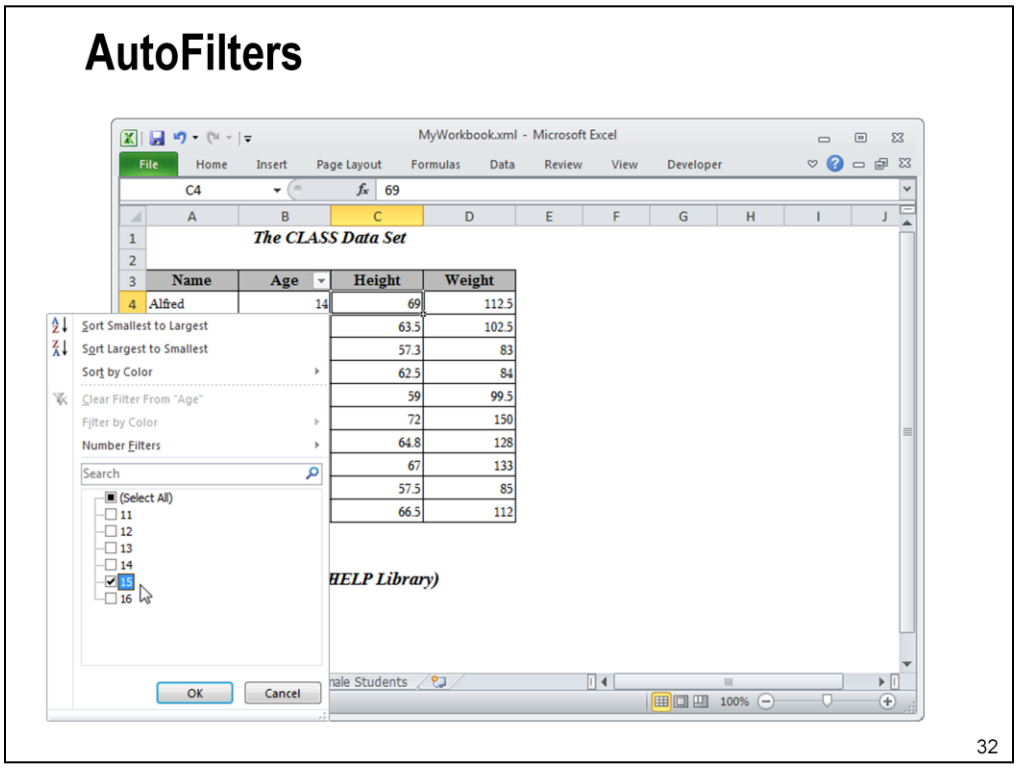
You click the AutoFilter button on the "Age" column.

AutoFilters



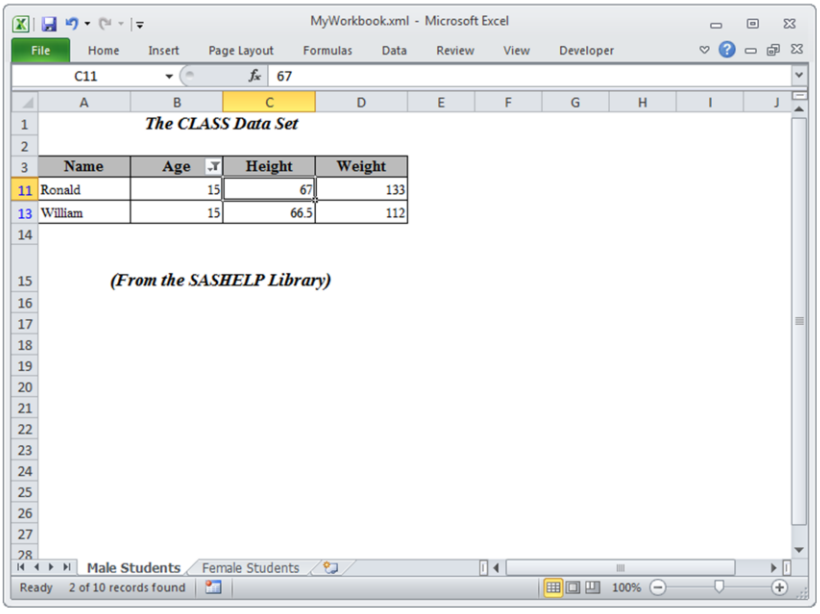
31

Excel provides you with, among other things, a list of all the unique data values for that column.



Select "15" from the list, and then click `OK`.

AutoFilters



33

All records in the worksheet that do not have "15" as a value in the "Age" column are hidden.

The data is still present in the worksheet, but it is not displayed due to the filtering.

AutoFilters

```
*  Set some "global" tagset options;

ods tagsets.ExcelXP
  options(embedded_titles='yes'
          embedded_footnotes='yes'
          print_header='&C&A&RPage &P of &N'
          print_footer='&RPrinted &D at &T'
          autofilter='2');

ods tagsets.ExcelXP options(sheet_name=...);

proc print ...; run; quit;

ods tagsets.ExcelXP options(sheet_name=...);

proc print ...; run; quit;
```

34

You can control which columns have AutoFilters by specifying "all", "none", or a range of contiguous columns ("2-4", for example).

Ex. 4 – Print Headers/Footers & AutoFilter

1. Go to SAS
2. File > Open Program > **Exercise4.sas**
3. Follow TO DO instructions and submit code
4. Go to Excel
5. File > **\HOW\DelGobbo\MyWorkbook.xml**
   - or -
   MyWorkbook.xml (from the recent file list)
6. Close the document (child) window (leave Excel running, but minimize it)

35

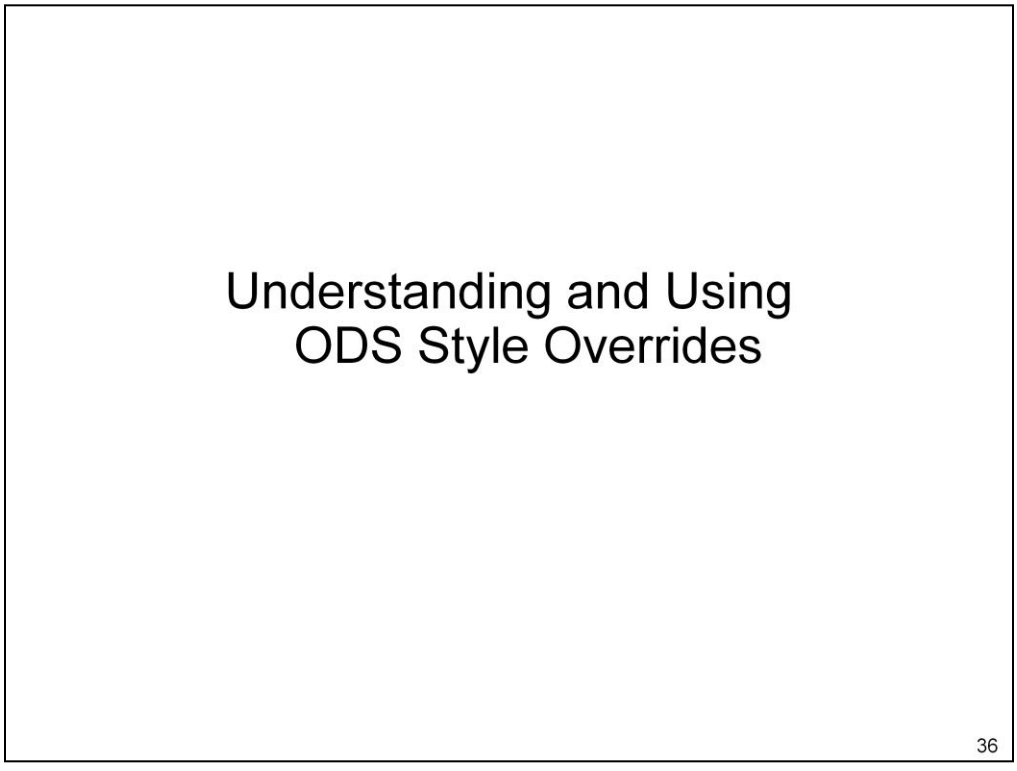
TO DO:

Lines 19-20: Note print header and footer.

Line 21: Add option to apply an AutoFilter to column #2.

Solution:

```
ods tagsets.ExcelXP options(embedded_titles='yes'
                            embedded_footnotes='yes'
                            print_header='&CA&RPage &P of &N'
                            print_footer='&RPrinted &D at &T'
                            autofilter='2');
```



The techniques described *in this section only* work only with the PRINT, REPORT, and TABULATE procedures.

Changing Display Attributes & Number Formats

- Gender-appropriate background 
- Centered column heading text
- Decimal place for Height and Weight
- Supported by PRINT, REPORT and TABULATE

Name	Age	Height	Weight
Alfred	14	69.0	112.5
Henry	14	63.5	102.5
James	12	57.3	83.0
Jeffrey	13	62.5	84.0
John	12	59.0	99.5
Philip	16	72.0	150.0
Robert	12	64.8	128.0
Ronald	15	67.0	133.0
Thomas	11	57.5	85.0
William	15	66.5	112.0

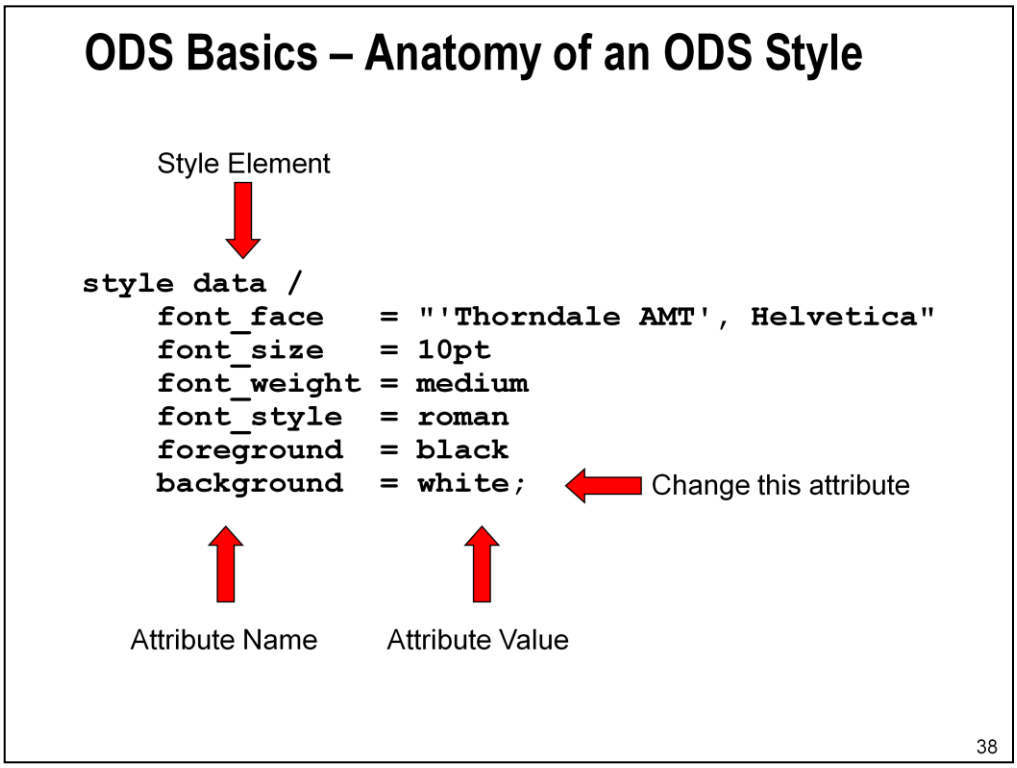
Name	Age	Height	Weight
Alice	13	56.5	84.0
Barbara	13	65.3	98.0
Carol	14	62.8	102.5
Jane	12	59.8	84.5
Janet	15	62.5	112.5
Joyce	11	51.3	50.5
Judy	14	64.3	90.0
Louise	12	56.3	77.0
Mary	15	66.5	112.0

37

We use ODS style overrides to change the appearance of our output.

Here the data cells for males have a blue background, and females have a pink background.

All the cells in the "Height" and "Weight" columns have one decimal place, with insignificant zeroes displayed as needed. Prior to this, insignificant zeroes were not displayed.

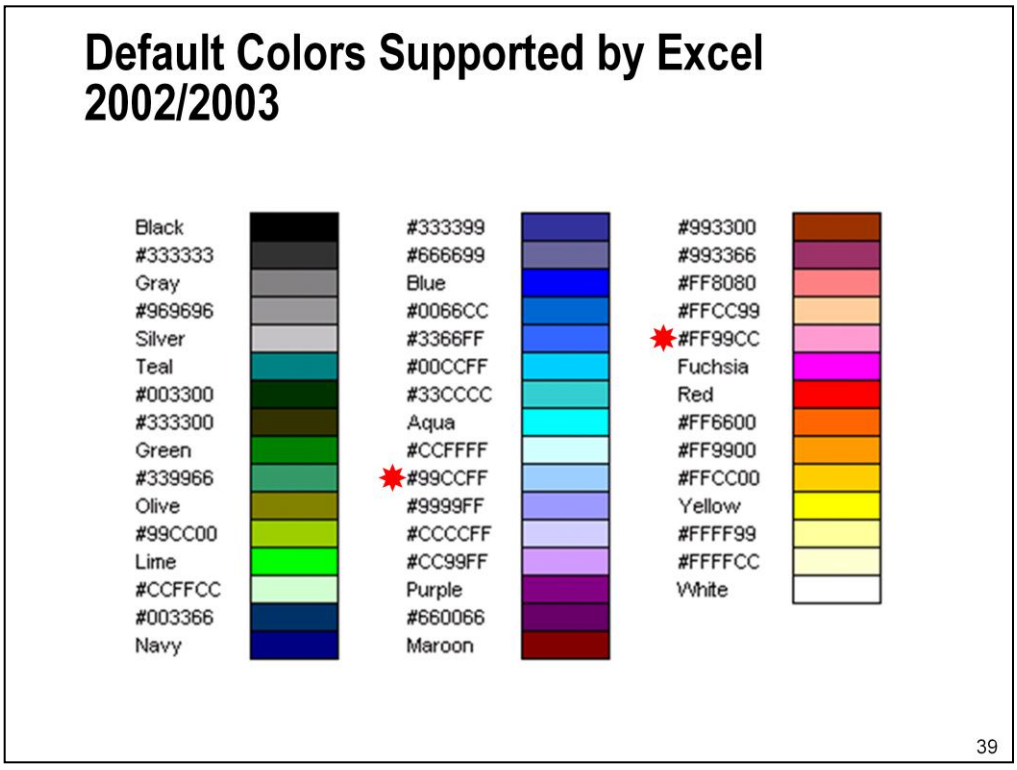


ODS styles are similar to HTML Cascading Style Sheet styles (CSS).

An ODS style is composed of many *style elements,* each of which control the appearance of a particular part of the output. For example, styles contain a style element named "data" that controls the appearance of the "data" cells in PROC PRINT output.

Style elements consist of collections of *style attributes*, such as the background color and font size. The definition of the style element named "data" looks like what is show here.

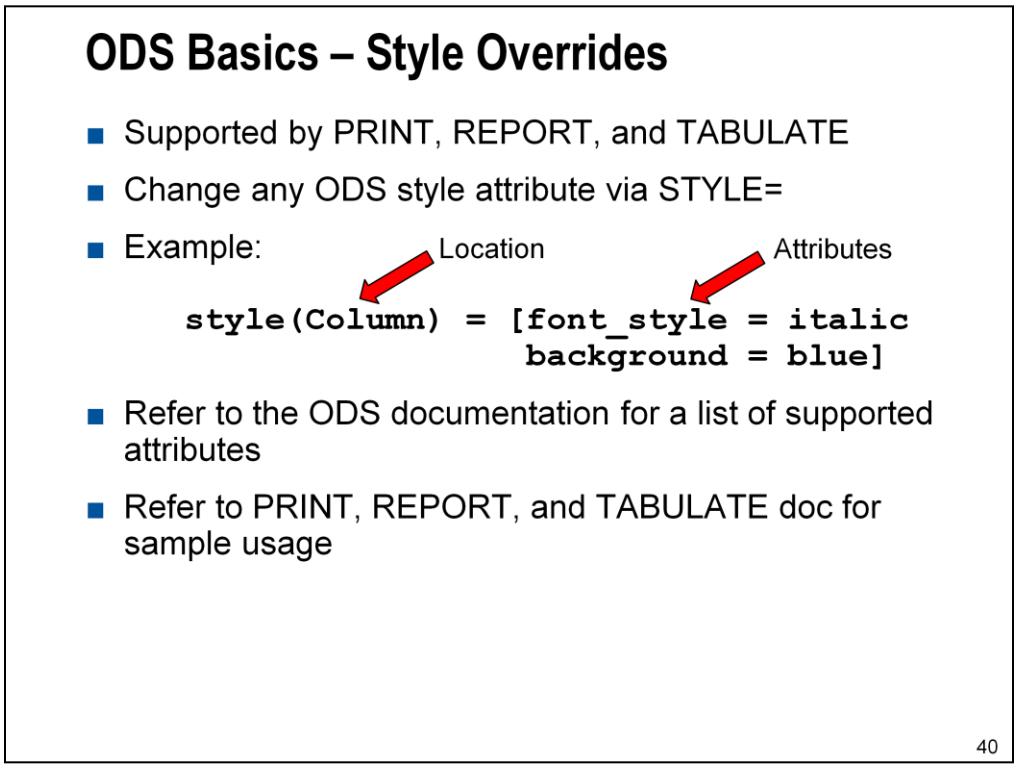
We change the background color to blue for male students, and pink for female students by changing the value of the "background" style attribute.



Excel versions 2002 and 2003 have a limited color palette, and the default colors are shown here.

If you plan to view your workbooks using one of these versions of Excel, choose colors that are listed in the default palette. Otherwise Excel maps the unsupported color to a color that is supported.

The color values used in the style overrides are indicated by a star (✸).



Style overrides are supported by the PRINT, REPORT, and TABULATE procedures, and can be specified in several ways, with the most common shown here.

While this is the most commonly used format, it has some disadvantages. To use the same style override for different variables, you must apply it in multiple places, making your SAS code harder to read and maintain. Refer to the accompanying paper for alternate techniques.

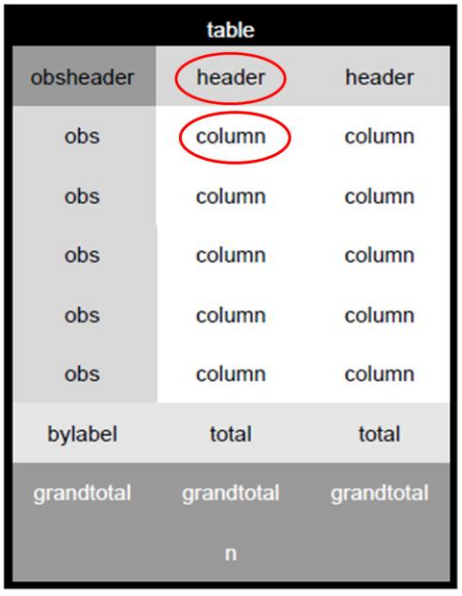
ODS Style Attributes

SAS 9.1: http://tinyurl.com/3dumjtp

SAS 9.2: http://tinyurl.com/3n6gjuc

SAS 9.3: http://tinyurl.com/3mnc3c2

Location Values for PROC PRINT



41

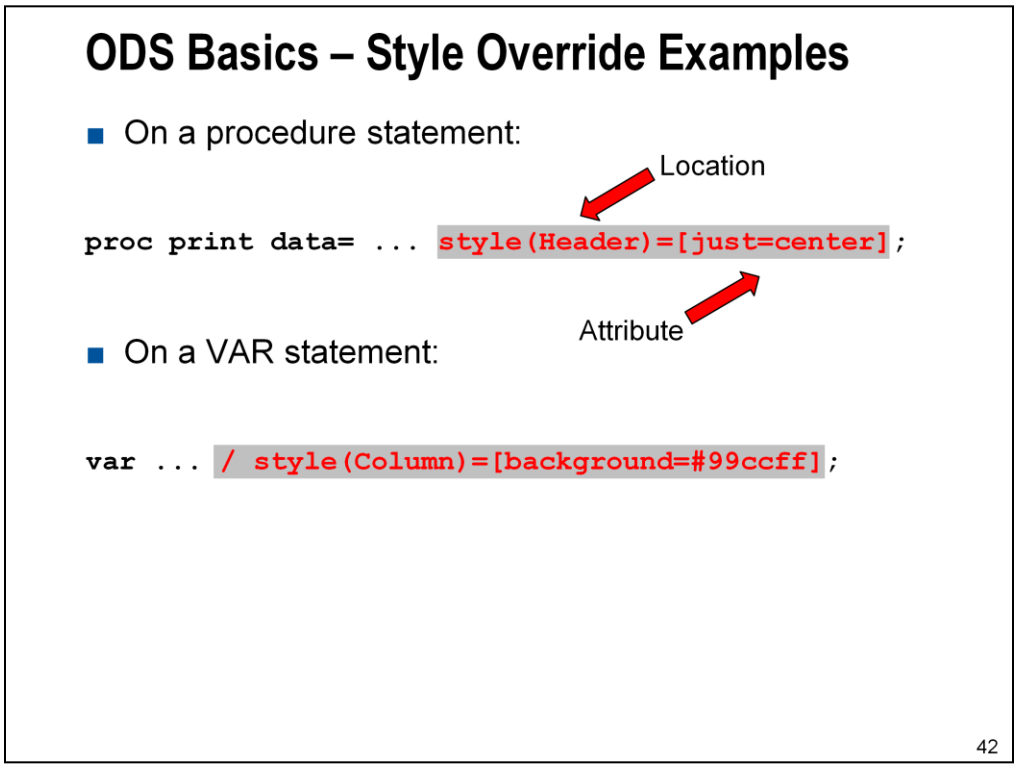
Style overrides are used to apply style attributes or elements to specific parts of SAS output called *locations*. Here you can see the locations that are pertinent to the PROC PRINT output.

To change the text justification of the header cells in our output, we apply a style overrides to the "Header" location.

To change the background color of the data cells in our output, we apply style overrides to the "Column" location.

SAS 9® Reporting Styles Tip Sheet

http://tinyurl.com/3cfqdv9



When used on a procedure statement, the style override applies to all cells for the specified location.

Style overrides specified on a VAR statement apply only to the cells for that variable and location.

Center Headings & Change Backgrounds

```
proc print ... style(Header)=[just=center]; ❶
  where (sex eq 'M');
  var name age /
    style(Column)=[background=#99ccff];
  var height weight /                      ❷
    style(Column)=[background=#99ccff];
run; quit;

*  Other ODS statement here...

proc print ... style(Header)=[just=center]; ❶
  where (sex eq 'F');
  var name age /
    style(Column)=[background=#ff99cc];
  var height weight /                      ❷
    style(Column)=[background=#ff99cc];
run; quit;
```

43

Although we could use the inline format of the STYLE option to center the column heading text by specifying the style override for each variable, it is easier to use a single style override on the procedure statement (❶) to change the justification of *all* column headings.

To apply one ODS style override to the "name" and "age" columns, and later, a different style override the "height" and "weight" columns, we need to split the single VAR statements into 2 separate statements. Style overrides (❷) are applied to the "Column" location of each variable to change the background color of the data cells.

Center Headings & Change Backgrounds

Name	Age	Height	Weight
Alfred	14	69	112.5
Henry	14	63.5	102.5
James	12	57.3	83
Jeffrey	13	62.5	84
John	12	59	99.5
Philip	16	72	150
Robert	12	64.8	128
Ronald	15	67	133
Thomas	11	57.5	85
William	15	66.5	112

Name	Age	Height	Weight
Alice	13	56.5	84
Barbara	13	65.3	98
Carol	14	62.8	102.5
Jane	12	59.8	84.5
Janet	15	62.5	112.5
Joyce	11	51.3	50.5
Judy	14	64.3	90
Louise	12	56.3	77
Mary	15	66.5	112

44

Style overrides applied to column heading and data cells.

Ex. 5 – Style Overrides

1. Go to SAS
2. File > Open Program > **Exercise5.sas**
3. Follow TO DO instructions and submit code
4. Go to Excel
5. File > **\HOW\DelGobbo\MyWorkbook.xml**
   - or -
   MyWorkbook.xml (from the recent file list)
6. Close the document (child) window (leave Excel running, but minimize it)

45

TO DO:

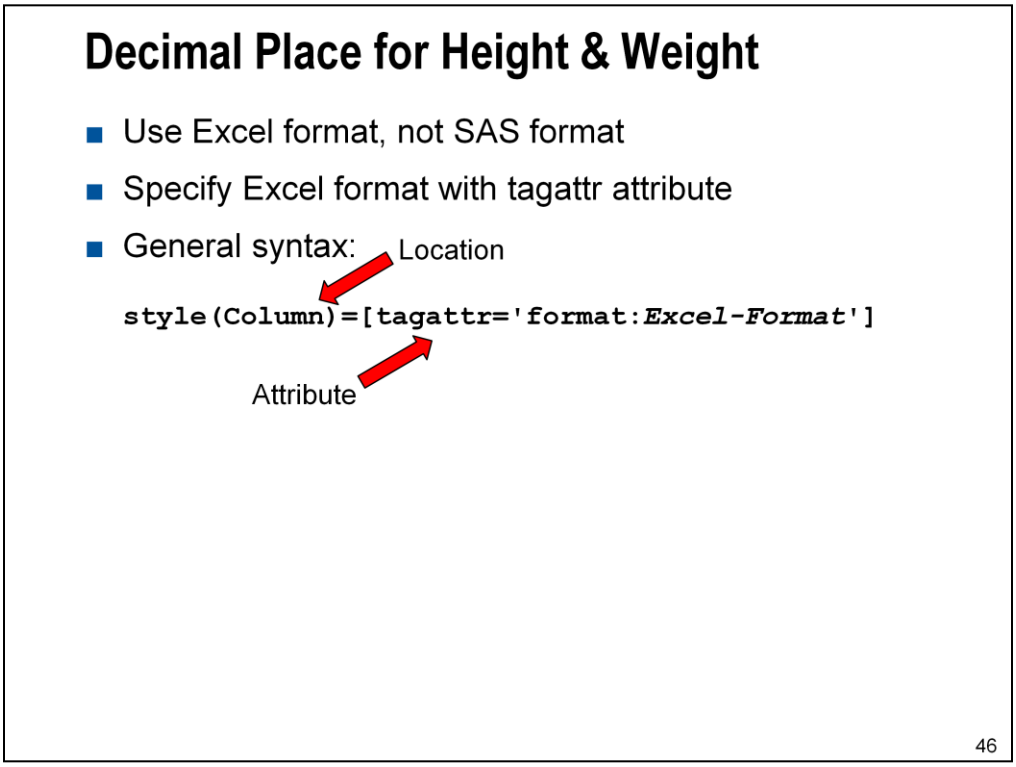
Lines 27 and 35: Specify attribute setting to center column heading text.

Lines 29-30 and 37-38: Note "split" VAR statements.

Lines 29-30 and 37-38: Specify the background color values `#99ccff` for and males, and `#ff99cc` for females.

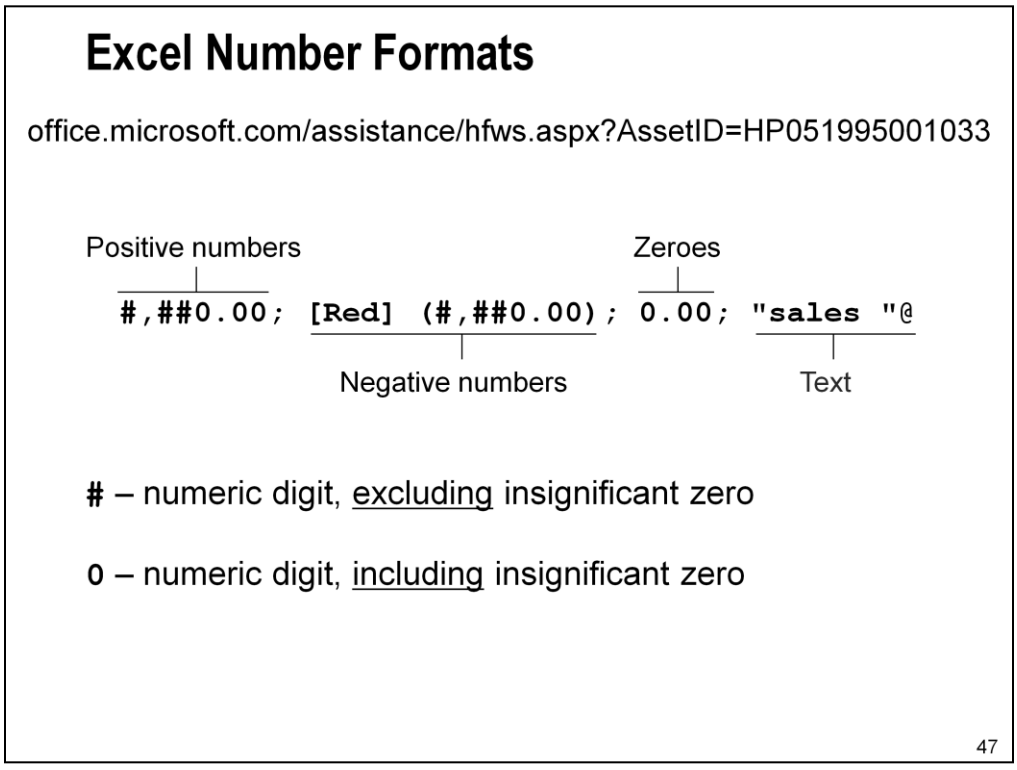
Solution:

```
proc print data=sashelp.class noobs style(Header)={just=center};

var name age      / style(Column)={background=#99ccff};
var height weight / style(Column)={background=#99ccff};

proc print data=sashelp.class noobs style(Header)={just=center};

var name age      / style(Column)={background=#ff99cc};
var height weight / style(Column)={background=#ff99cc};
```



Finally, we use a style override to apply an Excel format to display 1 decimal place in the values of the "height" and "weight" columns. You may sometimes have success using SAS formats to control the Excel display values, but it is usually better to instead use Excel formats.

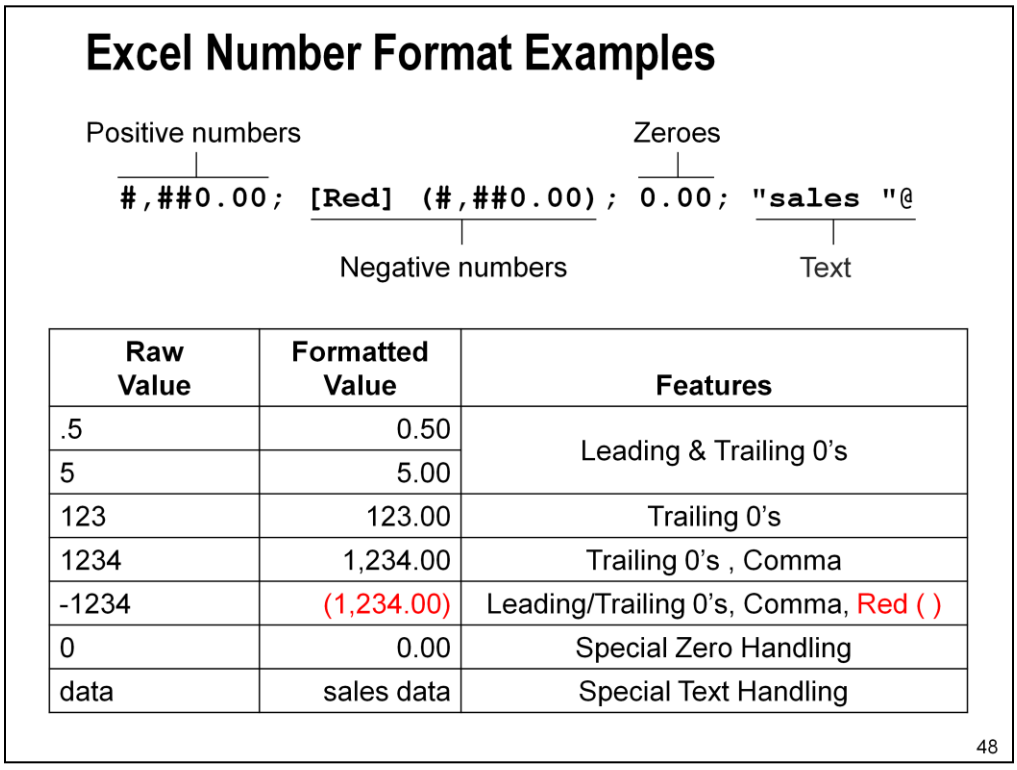
The general syntax for the style override is the same as before, specifying a location and an attribute value. We use the "tagattr" attribute to specify the Excel format.



The general structure of Excel formats is shown here.

The pound sign (#) in an Excel format is used to represent a numeric digit, excluding insignificant zeroes, and a zero (0) displays a numeric digit including insignificant zeroes.

Use zeros in Excel formats when you want to retain leading or trailing zeroes.



This table shows the results of applying the Excel format shown to several data values.

Excel 2003

http://office.microsoft.com/assistance/hfws.aspx?AssetID=HP005199500

Excel 2007

http://office.microsoft.com/assistance/hfws.aspx?AssetID=HP001216503

Excel 2010

http://office.microsoft.com/assistance/hfws.aspx?AssetID=HP010342372

Number Format Codes

http://office.microsoft.com/assistance/hfws.aspx?AssetID=HP005198679

Decimal Place for Height & Weight

```
proc print ... style(Header)=[just=center];
  where (sex eq 'M');
  var name age /
    style(Column)=[background=#99ccff];
  var height weight /
    style(Column)=[background=#99ccff
                   tagattr='format:#.0'];
run; quit;

*  Other ODS statement here...

proc print ... style(Header)=[just=center];
  where (sex eq 'F');
  var name age /
    style(Column)=[background=#ff99cc];
  var height weight /
    style(Column)=[background=#ff99cc
                   tagattr='format:#.0'];
run; quit;
```

49

Because we are working only with positive values, we use the Excel format `#.0` with the ODS "tagattr" attribute to specify an Excel format.

Be sure to quote the entire attribute value and include the "format:" keyword.

Final Workbook

- Gender-appropriate background
- Centered column heading text
- Decimal place for Height and Weight
- Supported by PRINT, REPORT and TABULATE

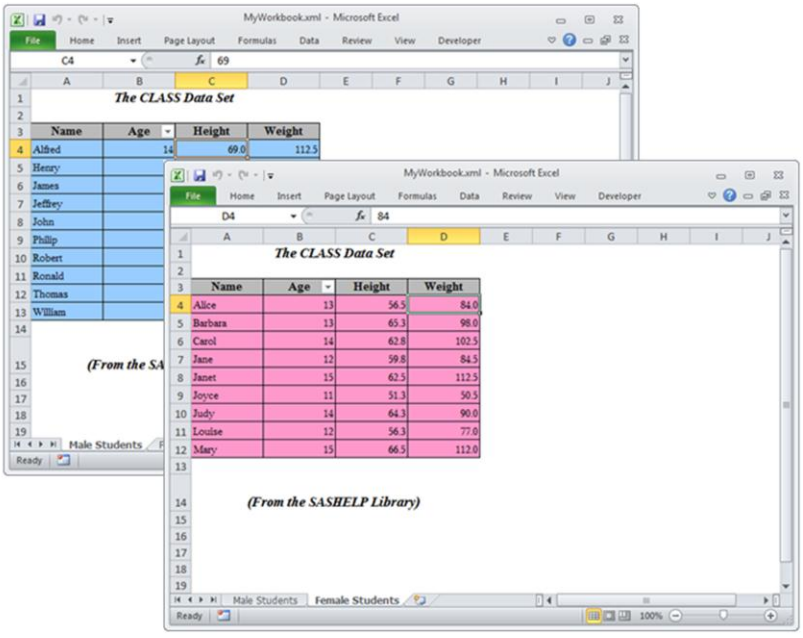
Name	Age	Height	Weight
Alfred	14	69.0	112.5
Henry	14	63.5	102.5
James	12	57.3	83.0
Jeffrey	13	62.5	84.0
John	12	59.0	99.5
Philip	16	72.0	150.0
Robert	12	64.8	128.0
Ronald	15	67.0	133.0
Thomas	11	57.5	85.0
William	15	66.5	112.0

Name	Age	Height	Weight
Alice	13	56.5	84.0
Barbara	13	65.3	98.0
Carol	14	62.8	102.5
Jane	12	59.8	84.5
Janet	15	62.5	112.5
Joyce	11	51.3	50.5
Judy	14	64.3	90.0
Louise	12	56.3	77.0
Mary	15	66.5	112.0

50

The resulting workbook with all the code modifications in place.

Final Workbook



51

Another view of the final workbook.

Ex. 6 – More Style Overrides

1. Go to SAS
2. File > Open Program > **Exercise6.sas**
3. Follow TO DO instructions and submit code
4. Go to Excel
5. File > **\HOW\DelGobbo\MyWorkbook.xml**
   - or -
   MyWorkbook.xml (from the recent file list)
6. Close the document (child) window (leave Excel running, but minimize it)

52

TO DO:

Lines 26 and 35: Specify the Excel format to retain 1 decimal place.

Solution:

For each instance of PROC PRINT:

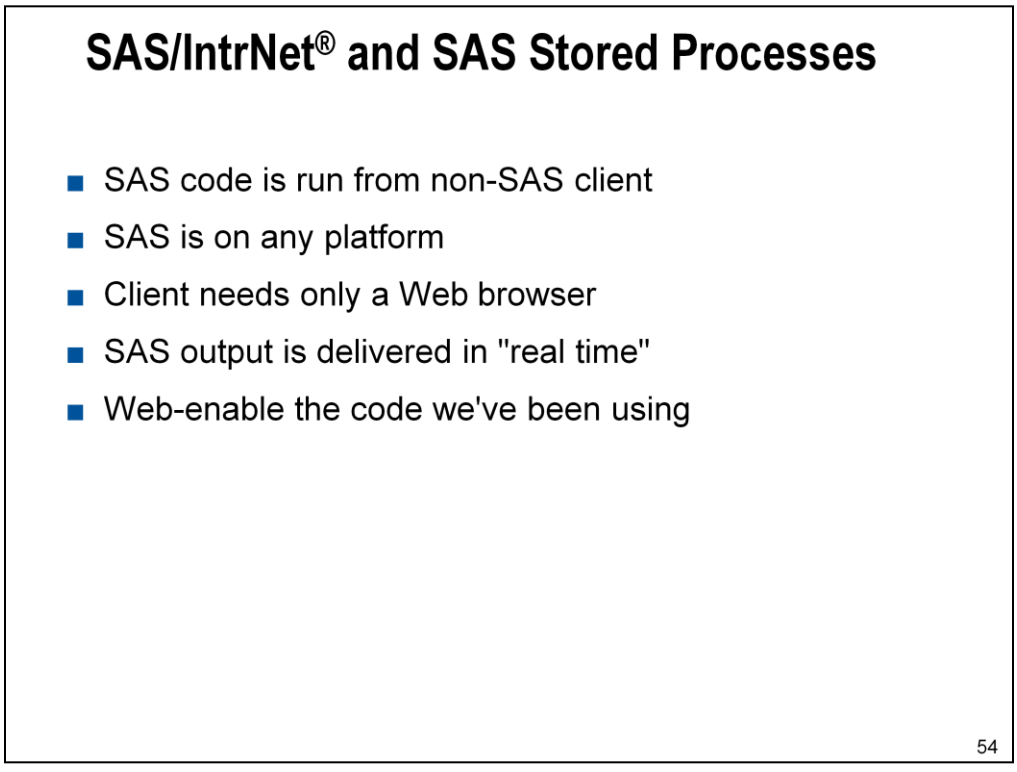
```
var height weight / style(Column)=[background=#99ccff
    tagattr='format:#.0'];
```

Using SAS/IntrNet®

and

SAS Stored Processes

53



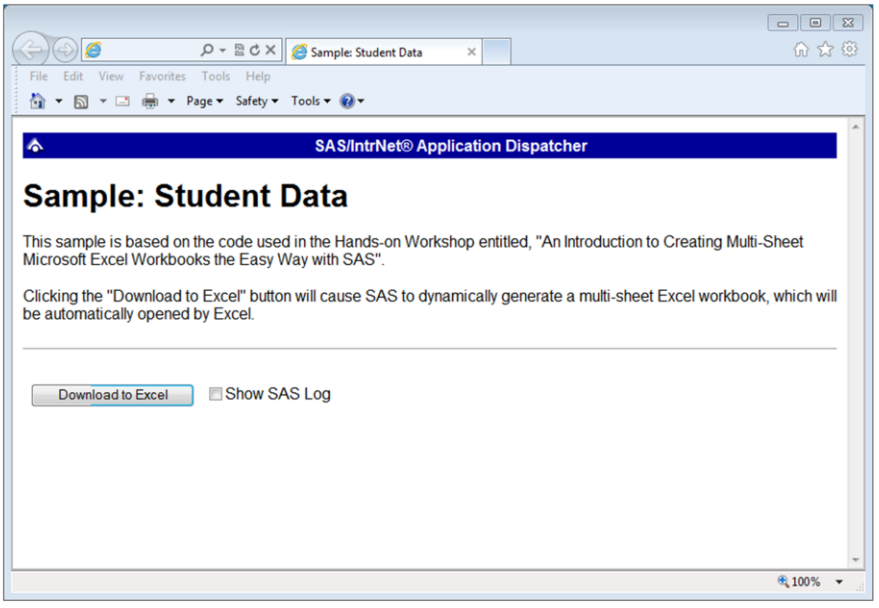
The purpose of the SAS/IntrNet® Application Dispatcher or SAS Stored Processes are to allow you to execute SAS programs from a client machine that does not have SAS installed. The client machine *may* have SAS installed, but that is not required.

A typical client-server model is followed. The SAS server can reside on any hardware platform (Windows, UNIX, z/OS, etc.) and is standing by, waiting to execute a SAS program. The most common client is a Web browser, again, running on any platform.

When the "OK" button of the Web page is clicked, input parameters, if any, are sent to the SAS server. Your SAS code executes, and the output is delivered in "real time" to the Web browser.

The following slides illustrate this process, using a "Web-enabled" version of the SAS code we have been working with.

Dynamically-Generated XML

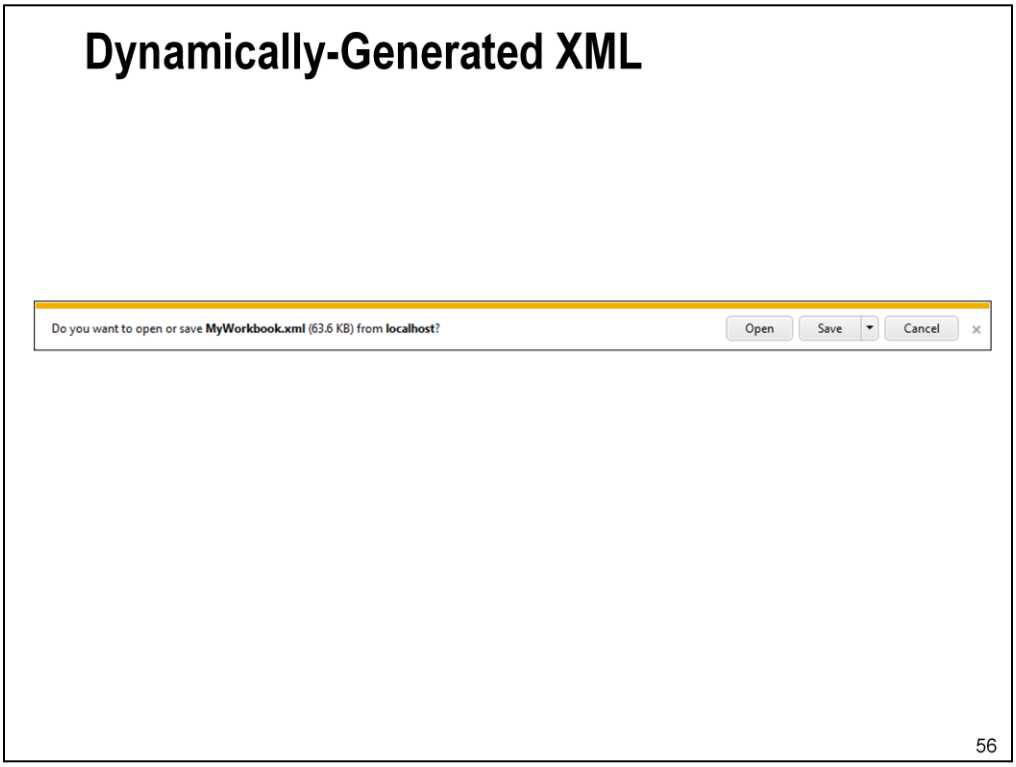


55

Here is a simple Web page that is used to execute SAS code stored on a server using the SAS/IntrNet® Application Dispatcher.

The code that is executed is substantially similar to the final version of the code that we used to generate XML file, with a few changes to "Web-enable" it.

Clicking "Download to Excel" causes the SAS program to execute on the server.



Once the SAS program executes, the results are sent back to the Web browser.

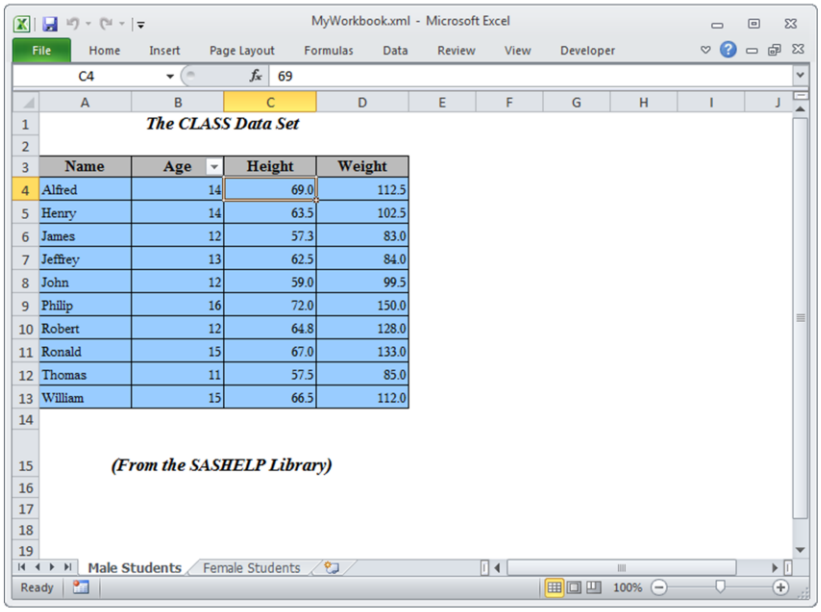
Note that you are presented with a "File Download" dialog box, instead of the results being displayed in the Web browser. Clicking "Open" causes the SAS output to be displayed in Excel, provided that Excel is installed on the client machine.

This code, specific to SAS/IntrNet®, must be specified before any ODS statements. It causes the SAS output to be displayed in Excel instead of the Web browser:

```
%let RV =%sysfunc(appsrv_header(Content-type,application/vnd.ms-excel));
```

This code will also work with SAS Stored Processes.

Dynamically-Generated XML



Name	Age	Height	Weight
Alfred	14	69.0	112.5
Henry	14	63.5	102.5
James	12	57.3	83.0
Jeffrey	13	62.5	84.0
John	12	59.0	99.5
Philip	16	72.0	150.0
Robert	12	64.8	128.0
Ronald	15	67.0	133.0
Thomas	11	57.5	85.0
William	15	66.5	112.0

57

Here is the SAS output, created in "real time", and delivered to the client. Note that Excel is used to view the SAS output, not the Web browser.

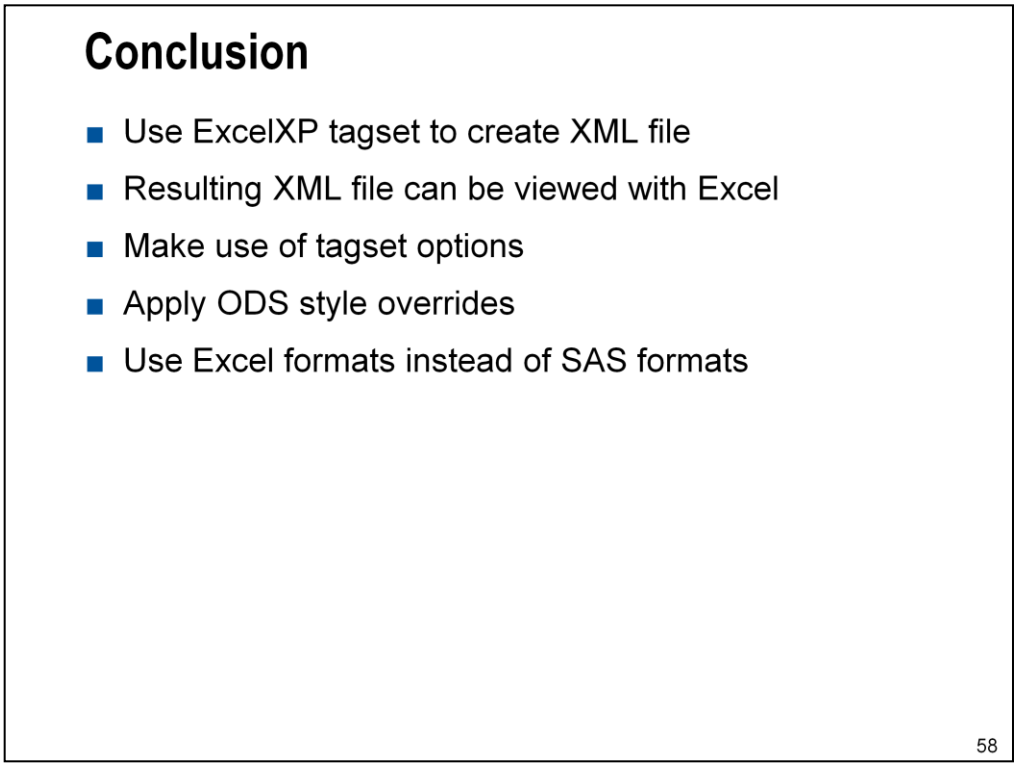
Refer to the accompanying paper for more information about this topic.

SAS/IntrNet® Application Dispatcher

SAS 9.1: http://tinyurl.com/3k8sw7b
SAS 9.2: http://tinyurl.com/3mu4gdl
SAS 9.3: http://tinyurl.com/3qpxzlg

SAS Stored Processes

SAS 9.1: http://tinyurl.com/3hcpgfl
SAS 9.2: http://tinyurl.com/3qxc7ch
SAS 9.3: http://tinyurl.com/3wyw6fk



Using ODS to generate specially-formatted XML output is an effective method of incorporating SAS output into Excel documents.

The SAS®9 ExcelXP ODS tagset provides an easy way to export your SAS data to Excel workbooks that contain multiple worksheets. By using ODS styles, style overrides, and a tagset that complies with the Microsoft XML Spreadsheet Specification, you can customize the output to achieve your design goals.

SAS Institute continues to work toward better Microsoft Office integration, and future releases of SAS software will provide even more robust means of using SAS content with Microsoft Office applications.

Resources

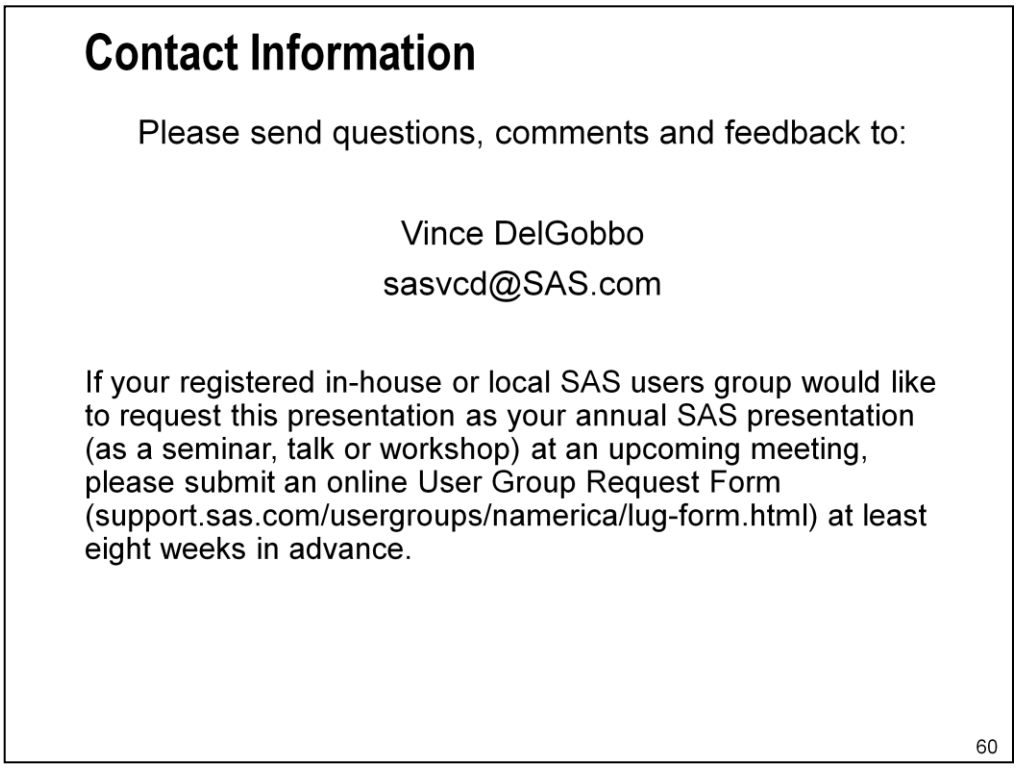
- Paper & Download Package

  support.sas.com/rnd/papers/index.html#excel2012

- Vince's ExcelXP Resources

  www.sas.com/reg/gen/corp/867226?page=Resources

59



About the author:

Vince DelGobbo is a Senior Systems Developer in the Web Tools group at SAS. This group's responsibilities include the SAS/IntrNet Application Dispatcher and SAS Stored Processes. He is involved in the development of new Web- and server-based technologies, as well as integrating SAS output with Microsoft Office. He was also involved in the early development of the ExcelXP ODS tagset. Vince has been a SAS Software user since 1982, and joined SAS in 1992.