

SM17: Programmatically Modifying a Document Created by ODS Excel: Adding a Password and Concatenating Multiple Worksheets into One

Jeffrey Meyers: 16 years of health science research programming experience with a focus on developing macros, creating graphics, and producing reports.

Jeff has presented at multiple conferences including PharmaSUG, MidWest SUG, and SAS Global Forum

P H U S E U S C o n n e c t 2 0 2 6

Programmatically Modifying a Document Created by ODS Excel: Adding a Password and Concatenating Multiple Worksheets into One

Jeffrey Meyers

REGENERON[®]

Contents

- Accessing XML files within an XLSX file
- Adding password protection programmatically
- Concatenating multiple worksheets into one programmatically

Excel Manipulation Methods

1. Relies on two facts about XLSX formats
 - Worksheets and workbooks are built out of XML files
 - XLSX files are also technically .zip files
2. XML files can be read into SAS and written out by SAS
3. .zip files can also be read into SAS and written to by SAS

XLSX as .ZIP file

```
data _null_;
```

```
rc=rename('test.xlsx', 'test.zip','file');
```

```
run;
```

☐ Name	Type	Compressed size	Password ...	Size	Ratio
._rels	File folder				
docProps	File folder				
xl	File folder				
[Content_Types]	XML Document	1 KB	No	2 KB	75%

☐ Name	Type	Compressed size	Password ...	Size	Ratio
._rels	File folder				
theme	File folder				
worksheets	File folder				
sharedStrings	XML Document	1 KB	No	1 KB	63%
styles	XML Document	1 KB	No	4 KB	78%
workbook	XML Document	1 KB	No	1 KB	41%

☐ Name	Type	Compressed size	Password ...	Size	Ratio
._rels	File folder				
sheet1	XML Document	2 KB	No	5 KB	71%
sheet2	XML Document	5 KB	No	33 KB	86%
sheet3	XML Document	2 KB	No	10 KB	81%

XLSX as XML files

1. “Three” important XML files

- Styles: contains font and other style elements in workbook
- Workbook: workbook global options
- SheetN: one XML for each worksheet containing data

```
- <cellXfs count="6">
  - <xf numFmtId="0" borderId="0" fillId="2" fontId="0" applyAlignment="1" applyBorder="1" applyFill="1" applyFont="1" xfid="0">
    <alignment horizontal="left"/>
  </xf>
  - <xf numFmtId="0" borderId="1" fillId="3" fontId="1" applyAlignment="1" applyBorder="1" applyFill="1" applyFont="1" xfid="0">
    <alignment horizontal="right"/>
  </xf>
  - <xf numFmtId="0" borderId="1" fillId="3" fontId="1" applyAlignment="1" applyBorder="1" applyFill="1" applyFont="1" xfid="0">
    <alignment horizontal="left"/>
  </xf>
  - <xf numFmtId="0" borderId="2" fillId="4" fontId="0" applyAlignment="1" applyBorder="1" applyFill="1" applyFont="1" xfid="0">
    <alignment horizontal="left"/>
  </xf>
  - <xf numFmtId="0" borderId="2" fillId="4" fontId="0" applyAlignment="1" applyBorder="1" applyFill="1" applyFont="1" xfid="0">
    <alignment horizontal="right"/>
  </xf>
  - <xf numFmtId="164" borderId="2" fillId="4" fontId="0" applyAlignment="1" applyBorder="1" applyFill="1" applyFont="1" xfid="0" applyNumberFormat="0">
    <alignment horizontal="right"/>
  </xf>
</cellXfs>
```

Importing the XLSX XML Files

```
filename inZip zip 'test.xlsx';
```

```
data sheet1;
```

```
infile inzip("xl/worksheets/sheet1.xml") trunccover;
```

```
input code $10000. ;
```

```
run;
```

```
<sheetProtection scenarios="1" objects="1" sheet="1"/> <printOptions headings="0" horizontalCentered="0" verticalCentered="0" gridLines="0"/>
```

```
<pageMargins left="0.05" right="0.05" top="0.5" bottom="0.5" header="0" footer="0"/>
```

```
<pageSetup paperSize="1" fitToWidth="1" fitToHeight="1" scale="100" orientation="portrait" horizontalDpi="300" verticalDpi="300"/>
```

Password Protection

1. Microsoft Excel has three levels of password protecting
 1. Worksheets
 2. Workbook
 3. Entire file
2. SAS Excel destination includes option to “protect worksheets”
 - This does not include option to add a password
3. Presentation will include methods to add password protecting to levels 1 and 2
 - Level three fundamentally changes structure of XLSX

ODS Excel Option

```
ods excel file='test.xlsx' options(protect_worksheet='on');
```

```
proc print data=sashelp.class;
```

```
run;
```

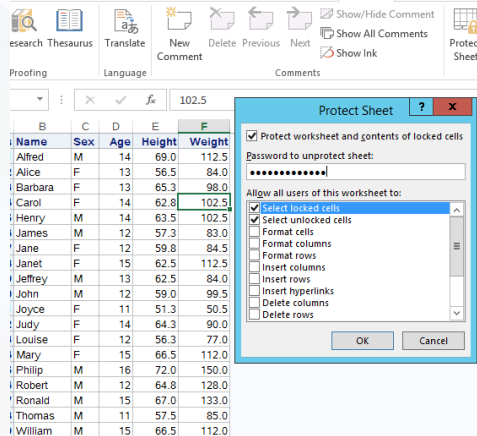
```
ods excel close;
```

The screenshot shows the Microsoft Excel interface with the 'REVIEW' tab selected. The 'Unprotect Sheet' button is highlighted with a red box. Below the worksheet, a dialog box titled 'Microsoft Excel' displays an error message: 'The cell or chart you're trying to change is on a protected sheet. To make changes, click Unprotect Sheet in the Review tab (you might need a password).'

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1		Obs	Name	Sex	Age	Height	Weight							
2		1	Alfred	M	14	69.0	112.5							
3		2	Alice	F	13	56.5	84.0							
4		3	Barbara	F	13	65.3	98.0							
5		4	Carol	F	14	62.8	102.5							
6		5	Henry	M	14	63.5	102.5							
7		6	James	M	12	57.3	83.0							
8		7	Jane	F	12	59.0								
9		8	Janet	F	15	62.5								
10		9	Jeffrey	M	13	62.5								
11		10	John	M	12	59.0								
12		11	Joyce	F	11	51.3								
13		12	Judy	F	14	64.1								
14		13	Louise	F	12	56.1								
15		14	Mary	F	15	66.5	112.0							
16		15	Phillip	M	16	72.0	150.0							
17		16	Robert	M	12	64.8	128.0							
18		17	Ronald	M	15	67.0	133.0							
19		18	Thomas	M	11	57.5	85.0							
20		19	William	M	15	66.5	112.0							

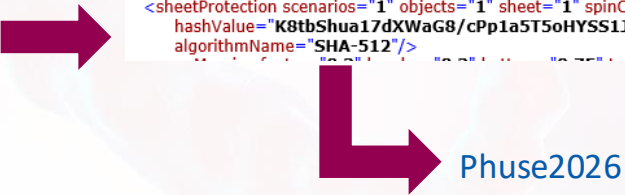
Generate a Password Hash

1. Passwords are encrypted within XML
2. Unknown how to convert text into this encryption
3. Manually add password that I want to random file, then pull it from the XML



The screenshot shows an Excel worksheet with a table of names and physical attributes. The 'Protect Sheet' dialog box is open, showing options to protect the worksheet and its contents. The 'Password to unprotect sheet:' field is empty. The 'Allow all users of this worksheet to:' section is expanded, showing options for selecting locked cells, selecting unlocked cells, format cells, format columns, format rows, insert columns, insert rows, insert hyperlinks, delete columns, and delete rows. The 'Protect sheet and contents of locked cells' checkbox is checked.

Name	Sex	Age	Height	Weight
Alfred	M	14	69.0	112.5
Alice	F	13	56.5	84.0
Barbara	F	13	65.3	98.0
Carol	F	14	62.8	102.5
Henry	M	14	63.5	102.5
James	M	12	57.3	83.0
Jane	F	12	59.8	84.5
Janet	F	15	62.5	112.5
Jeffrey	M	13	62.5	84.0
John	M	12	59.0	99.5
Joyce	F	11	51.3	50.5
Judy	F	14	64.3	90.0
Louise	F	12	56.3	77.0
Mary	F	15	66.5	112.0
Philip	M	16	72.0	150.0
Robert	M	12	64.8	128.0
Ronald	M	15	67.0	133.0
Thomas	M	11	57.5	85.0
William	M	15	66.5	112.0



The diagram shows a flow from the 'Protect Sheet' dialog box to the XML hash value and then to the Phuse2026 website. A red arrow points from the 'Protect Sheet' dialog box to the XML hash value, and another red arrow points from the XML hash value to the Phuse2026 website.

```
<sheetProtection scenarios="1" objects="1" sheet="1" spinCount="100000" saltValue="R6G1aQHyoDZFhRP9zkRRNg==" hashValue="K8tbShua17dXWaG8/cPp1a5T5oHYSS1JYdWLiCAvpGpV/HZGpRYT8J4ouRr/1uZAsKcdAQhoiEsQcdoosc0xBA==" algorithmName="SHA-512"/>
```

Phuse2026

Inject Password into XML

...Previous import code here

```
if find(code,"<sheetProtection","i")>0 then do;  
  start=find(code,'/','i',find(code,'<sheetProtection','i')); **find end tag for sheetProtection;  
  code=catx(' ',substr(code,1,start-1),  
            Hash code from earlier  
            substr(code,start));  
  
end;  
drop start;  
run;  
***Write modified XML code back to file;  
data _null_;  
  set sheet1;  
  file inzip("xl/worksheets/sheet1.xml");  
  put code;  
run;
```

Worksheet is now Password Protected













Proofing Language Comments Unprotect Sheet

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83.0

Unprotect Sheet

?

x

Password:

OK

Cancel

Password Protection

1. Room for more efficient coding
2. Code for level 2 is similar and is included in the paper
3. Example in paper for unlocking specific columns for editing
4. Level 3 changes the structure of the XLSX file and couldn't open in XML viewer for testing



Microsoft Excel
Worksheet

Concatenating Worksheets into One

1. Printing tables to the Excel destination is memory intensive
2. Longer/wider tables with style modifications can easily run out of available memory and throw an error preventing printing
3. One solution is to split table into multiple worksheets
 - Memory limits apply to each worksheet separately
4. Returning output to the intended single worksheet is tedious and manual effort
5. XML can be modified to automatically concatenate worksheets while preserving style modifications

Example Output

1. Created with the REPORT procedure (code is in paper)
2. Sets up column widths, frozen spanned headers, merged cells, alternating shading, traffic highlighting and bold lines for visualizing
3. Created from SASHELP.CARS for simplicity
4. One Sheet for each ORIGIN (Asia, Europe, USA)
5. Goal is to recombined three sheets back together while preserving the style elements



Microsoft Excel
Worksheet

Why not Use the XLSX Libname?

1. XLSX engine in the libname statement allows worksheets to be set on top of each other in a data step set statement
2. Doing so eliminates all formatting and style settings in the workbook
3. XLSX does not allow deleting worksheets to clean up after combining (at least in Linux)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Car Category				Cost		Engine					Size			
2	Origin	Make	Model	Type	MSRP	Invoice	DriveTrain	Engine Size (L)	Cylinders	Performance	MPG (City)	MPG (Highway)	Weight (LBS)	Wheelbase (IN)	Length (IN)
3	Asia	Acura	MDX	SUV	36945	33337	All	3.5	6	265	17	23	4451	106	189
4			RSX Type S 2dr	Sedan	23820	21761	Front	2	4	200	24	31	2778	101	172
5			TSX 4dr	Sedan	26990	24647	Front	2.4	4	200	22	29	3230	105	183
6			TL 4dr	Sedan	33195	30299	Front	3.2	6	270	20	28	3575	108	186
7			3.5 RL 4dr	Sedan	43755	39014	Front	3.5	6	225	18	24	3880	115	197
8			3.5 RL w/Navigation 4dr	Sedan	46100	41100	Front	3.5	6	225	18	24	3893	115	197
9			NSX coupe 2dr manual S	Sports	89765	79978	Rear	3.2	6	290	17	24	3153	100	174
10		Honda	Civic Hybrid 4dr manual (gas/electric)	Hybrid	20140	18451	Front	1.4	4	93	46	51	2732	103	175
11			Insight 2dr (gas/electric)	Hybrid	19110	17911	Front	2	3	73	60	66	1850	95	155
12			Pilot LX	SUV	27560	24843	All	3.5	6	240	17	22	4387	106	188

XML Approach

1. Separate each worksheet XML file into a header, body and footer section
2. Keep the header and footer from worksheet one and concatenate the body sections from all worksheets in between
 - Keeps the style and protection settings intact
3. Update any cell references with corrected row numbers after concatenating
4. Delete concatenated worksheets out of the workbook
5. Update any meta information needed

Header Section

1. Every row from start until <sheetData> tag
2. worksheet meta information, column widths, frozen headers, zoom and other information

	code
CARS	1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
_WORKSHEETS	2 <worksheet xmlns="http://schemas.openxmlformats.org/spreadsheetml/2006/main" xmlns:r="http://schemas.openxmlfor...
_WS1_HEAD	3 <sheetPr filterMode="0" enableFormatConditionsCalculation="0"/>
_WS1_BODY	4 <pageSet UpPr fitToPage="0"/>
_WS1_FOOT	5 </sheetPr>
_WS2_BODY	6 <dimension ref="A1:O160"/>
_WS2_MC	7 <sheetViews>
_WS3_BODY	8 <sheetView zoomScale="100" zoomScaleNormal="100" zoomScalePageLayoutView="100" workbookViewId="0">
_WS3_MC	9 <pane xSplit="1" ySplit="2" topLeftCell="B3" activePane="bottomRight" state="frozen"/>
_COMBINE	10 </sheetView>
_WORKBOOK	11 </sheetViews>
	12 <sheetFormatPr baseColWidth="10" defaultRowHeight="12" customHeight="1" x14ac:dyDescent="0"/>
	13 <cols>
	14 <col min="1" max="1" width="12.71" bestFit="1" customWidth="1"/>
	15 <col min="2" max="2" width="16.71" bestFit="1" customWidth="1"/>
	16 <col min="3" max="3" width="35.71" bestFit="1" customWidth="1"/>
	17 <col min="4" max="4" width="10.71" bestFit="1" customWidth="1"/>
	18 <col min="5" max="5" width="10.71" bestFit="1" customWidth="1"/>
	19 <col min="6" max="6" width="10.71" bestFit="1" customWidth="1"/>
	20 <col min="7" max="7" width="12.71" bestFit="1" customWidth="1"/>
	21 <col min="8" max="8" width="15.71" bestFit="1" customWidth="1"/>
	22 <col min="9" max="9" width="11.71" bestFit="1" customWidth="1"/>
	23 <col min="10" max="10" width="13.71" bestFit="1" customWidth="1"/>
	24 <col min="11" max="11" width="10.71" bestFit="1" customWidth="1"/>
	25 <col min="12" max="12" width="15.71" bestFit="1" customWidth="1"/>
	26 <col min="13" max="13" width="13.71" bestFit="1" customWidth="1"/>
	27 <col min="14" max="14" width="15.71" bestFit="1" customWidth="1"/>
	28 <col min="15" max="15" width="11.71" bestFit="1" customWidth="1"/>
	29 </cols>
	30 <sheetData>

Body Section

1. Contains the information in the cells of the worksheet (including headers)
2. Numeric values can be seen between the <v> tags
3. Character strings are stored in another XML and referenced
4. r= tag refers to the cell reference
 - This will need to be updated when combined

1	<row r="1" ht="14" customHeight="1">	
2	<C r="A1" s="1" t="s"><v>0</v></C>	
3	<C r="B1" s="1"/>	
4	<C r="C1" s="1"/>	
5	<C r="D1" s="1"/>	
6	<C r="E1" s="1" t="s"><v>1</v></C>	
7	<C r="F1" s="1"/>	
8	<C r="G1" s="1" t="s"><v>2</v></C>	
9	<C r="H1" s="1"/>	
10	<C r="I1" s="1"/>	
11	<C r="J1" s="1" t="s"><v>3</v></C>	
12	<C r="K1" s="1"/>	
13	<C r="L1" s="1"/>	
14	<C r="M1" s="1" t="s"><v>4</v></C>	
15	<C r="N1" s="1"/>	
16	<C r="O1" s="1"/>	
17	</row>	

Footer Section

1. Contains the closing information for the cells, password protection, merged cells, and page setup
2. Merged Cell section contains any spanning headers or vertically merged rows (SPANROWS)
 - Cell references will need to be updated when combined
 - Merged cells from other worksheets will need to be collected and added between <mergeCells> tags

CARS		code
_WORKSHEETS	1	</sheetData>
_WS1_HEAD	2	<mergeCells count="20">
_WS1_BODY	3	<mergeCell ref="A1:D1"/>
_WS1_FOOT	4	<mergeCell ref="E1:F1"/>
_WS2_BODY	5	<mergeCell ref="G1:I1"/>
_WS2_MC	6	<mergeCell ref="J1:L1"/>
_WS3_BODY	7	<mergeCell ref="M1:O1"/>
_WS3_MC	8	<mergeCell ref="A3:A160"/>
_COMBINE	9	<mergeCell ref="B3:B9"/>
_WORKBOOK	10	<mergeCell ref="B10:B26"/>
	11	<mergeCell ref="B27:B38"/>
	12	<mergeCell ref="B39:B46"/>
	13	<mergeCell ref="B47:B48"/>
	14	<mergeCell ref="B49:B59"/>
	15	<mergeCell ref="B60:B70"/>
	16	<mergeCell ref="B71:B81"/>
	17	<mergeCell ref="B82:B94"/>
	18	<mergeCell ref="B95:B111"/>
	19	<mergeCell ref="B112:B113"/>
	20	<mergeCell ref="B114:B124"/>
	21	<mergeCell ref="B125:B132"/>
	22	<mergeCell ref="B133:B160"/>
	23	</mergeCells>
	24	<printOptions headings="0" horizontalCentered="0" verticalCentered="0" gridLines="0"/>
	25	<pageMargins left="0.05" right="0.05" top="0.5" bottom="0.5" header="0" footer="0"/>
	26	<pageSetup paperSize="1" fitToWidth="1" fitToHeight="1" scale="100" orientation="portr...>
	27	<headerFooter>
	28	<oddHeader></oddHeader>
	29	<oddFooter></oddFooter>
	30	</headerFooter>
	31	</worksheet>

Extraction code

1. Extensive code is displayed within the paper
 - Data steps with retain statements to output to correct datasets (e.g. __ws1_header)
2. Regular expressions used to extract and update row/column references

if find(code,'row r=','i') then

```
row=input(prxchange('s/.+row r="(\d+)".+/$1/',-1,code),??best.);
```

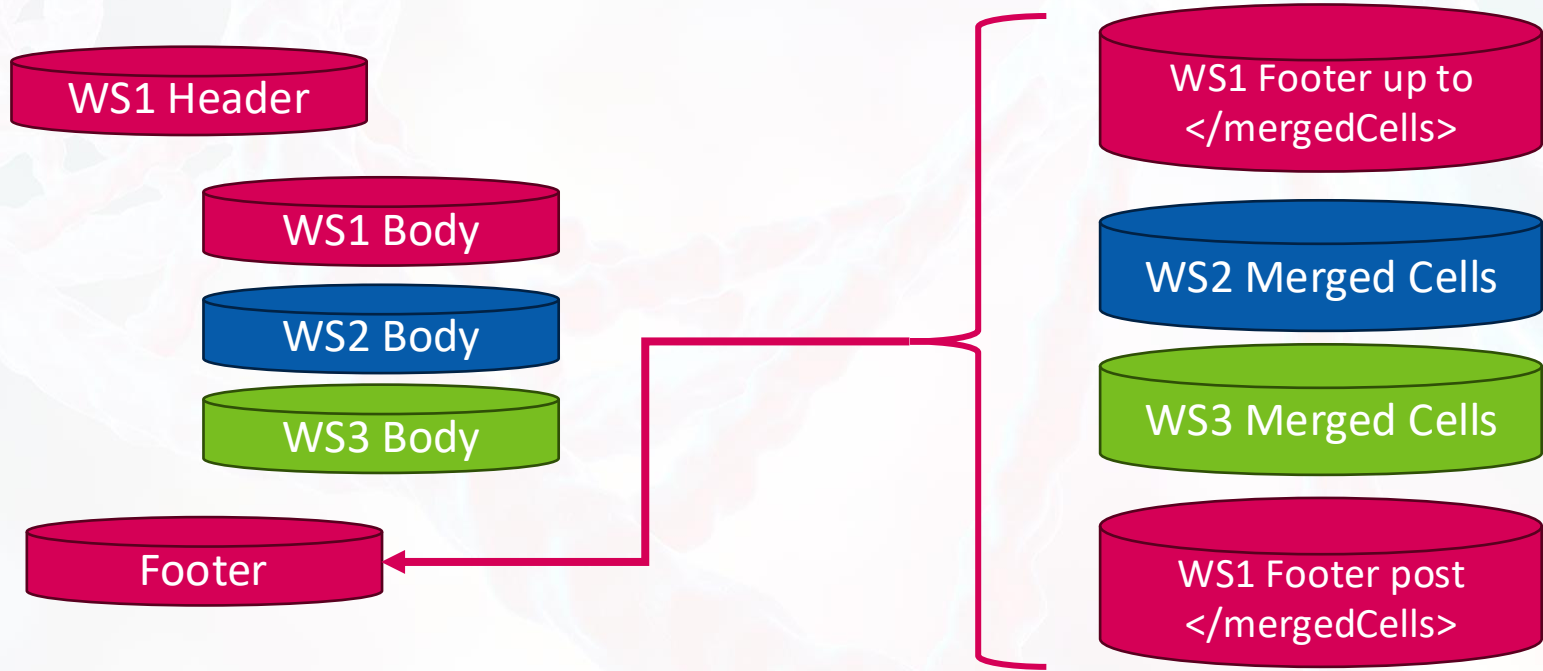
else if find(code,'/row','i')=0 then

```
column=prxchange('s/.+r="([a-zA-Z]+)\d+".+/$1/',-1,code);
```

```
code=prxchange('s/r=""||strip(catt(column,row))||
```

```
""/r=""||strip(catt(column,row+&lastRow.-&headerRows))||""/',-1,code);
```

Combine Worksheets



Delete Worksheets

1. Assign specific filenames for each worksheet to be deleted and then use the FEXIST function to delete the XML
2. Update the WORKBOOK XML to remove the references to these worksheets

```
%do i=2 %to &nsheets;

  filename inzip&i zip "&infile"

  member="xl/worksheets/sheet&i..xml";

  data _null_;
    x=fexist("inzip&i");

    if x then do;
      rc=fdelete("inzip&i");

    end;

  run;

  filename inzip&i clear;

%end;
```

```
data __workbook;

  infile inzip("xl/workbook.xml") truncover;

  input code $10000. ;

  if find(code,'sheet name=', 'i') and find(code,'sheetId="1"', 'i')=0 then delete;

run;

data _null_;

  set __workbook;

  file inzip("xl/workbook.xml");

  put code;

run;
```

Final Product

1. Updated workbook now contains all three worksheets' data concatenated into the first worksheet
2. All style modifications and settings have been preserved
3. Concatenated worksheets have been deleted



Microsoft Excel
Worksheet

Conclusion

1. Programmatically modifying the XML of an XLSX document opens more routes than normally available to SAS Programming for creative solutions
2. Requires study of the XML tags and how the different XML documents relate and communicate with each other behind the XLSX
3. Other potential additions such as adding GROUPING to tables

Contact Information

Name: Jeffrey Meyers

Affiliation: Regeneron Pharmaceuticals

E-mail: Jeffrey.Meyers@regeneron.com

Website: [SAS Communities Page](#)