

Tips for Using ODS in the Z/OS Environment

This document gives pointers on running ODS on Z/OS. It also discusses some things to watch for and some of the most frequently asked questions when running ODS on Z/OS. There is good information in the “Complete Guide to the SAS Output Delivery System” on running ODS in the various environments, which should be used as a reference. This document discusses the syntax as well as the pitfalls.

Status of Available Output Destinations with ODS in Version 9.2

HTML	Production 7.0
RTF	Production 8.1
TAGSETS.RTF	Production 9.2
POSTSCRIPT/PCL	Production 8.0
OUTPUT	Production 7.0
PDF	Production 8.1
XML	Production 9.0
CSV	Production 9.0
DOCUMENT	Production 9.0
LAYOUT	Pre-Production
ODS GRAPHICS	Production 9.2
PACKAGES	Production 9.2

Storing Files Generated with ODS

I. It is recommended that you create a PDSE to store files created with ODS unless running under HFS. Some of the benefits are discussed below.

A. The ability to create multiple body files based on the NEWFILE= option.

The NEWFILE= option allows you to create separate body files based on the by-group, page, procedure, or output. The creation of multiple body files with the NEWFILE= option is not allowed when writing to a sequential file which is done for obvious reasons.

B. Storing supporting files that are created with the BODY=, CONTENTS=, PAGE=, and the FRAME= options with ODS HTML generates errors in SAS 8.2 when writing to non- sms managed PDSE. Unless running with HFS, storing the files in a PDSE is the preferred method. A PDSE should be allocated with the with a DSNTYPE=LIBRARY. Allocating the file with a DISP=SHR or a DISP=OLD will generate the file in use error in SAS 8.2. If a PDSE is truly SMS managed, the Maximum directory blocks has a value of NOLIMIT.

ERROR: Current file XXX is in use

Data Set Name . . . : XXX.PDSE.HTML

General Data		Current Allocation	
Management class . . . :	STD	Allocated tracks . . :	19
Storage class :	STD	Allocated extents . . :	1
Volume serial :	SMS005	Maximum dir. blocks :	NOLIMIT
Device type :	3390		
Data class :	STD		
Organization :	PO	Current Utilization	
Record format :	VB	Used pages :	217
Record length :	8196	% Utilized :	95
Block size :	27998	Number of members . . :	29
1st extent tracks . . :	19		
Secondary tracks . . :	5		
Data set name type :	LIBRARY		

Creation date . . . : 2000/06/14 Referenced date . . : 2001/11/28
Expiration date . . : ***None***

Note- If you are unable to create a PDSE, the files can be created as sequential files rather than members of a PDSE. If you are downloading the files, rather than viewing them with an MVS Web server, then use the URL= sub-option with the name of the file or how it's addressed on the PC. On the target operating system, name the files the same as specified on the URL= sub-option. Below is an example.

```
FILENAME BODY '.BODY.HTML' RECFM=VB LRECL=8196 BLKSIZE=27998 DISP=NEW;

FILENAME CONTENTS '.CONTENTS.HTML' RECFM=VB LRECL=8196 BLKSIZE=27998
DISP=NEW;

FILENAME FRAME '.FRAME.HTML' RECFM=VB LRECL=8196 BLKSIZE=27998
DISP=NEW;

ODS HTML BODY=BODY(URL="BODY.HTML")
      CONTENTS=CONTENTS(URL="CONTENT.HTML")
      FRAME=FRAME(URL="FRAME.HTML")
      RS=NONE;

PROC PRINT DATA=SASHELP.CLASS;
RUN;

ODS HTML CLOSE;
```

Recommended DCB attributes for file Generated with ODS

I. The recommended DCB information for creating files on Z/OS is RECFM=VB, LRECL=8196 and BLKSIZE=27998 for non-PDF files. For PDF files, the RECFM and the BLKSIZE with the LRECL=259. This happens to be the default when using the FILENAME statement within SAS to allocate a PDSE. The DCB

characteristics are not cut in stone, but work very well. Below are some reasons for the above DCB attributes.

- A. The LRECL= option needs to be about 50 bytes longer than the largest record to avoid splitting words or tags over a record. When the tag splits over a line, this causes problems when the file is moved to an ASCII Web server. If you are not sure of the record length, which can be difficult to determine with HTML, the file can be created with an alternative method which is discussed in the section explaining moving the files to an ASCII based operating system.
- B. Specifying a RECFM=FB while the RS=NONE HTML ODS option is specified generates the below error. The RS=NONE option allows ODS to write one observation at a time.

ERROR: Invalid record type RECFM=.

Empty Files Generated when Creating Files with ODS

- I. An empty file will result if a step boundary is not provided by the procedure or data step when generating output with ODS. Ordinarily, when running in the batch environment on Z/OS, a step boundary is not required for procedure or data steps; However, when running in the batch environment on Z/OS, a step boundary such as a RUN or PROC statement is required . Failure to add this step boundary will result in an empty file. Below are the steps that can be used to prevent an empty file.
 - A: Add a step boundary either a RUN or a PROC statement to the end of the procedure or data step.
 - B: Remove the ODS HTML CLOSE statements. When the job is executed, an implied ENDSAS statement is issued which will close the file created with ODS.

Moving Files Generated with ODS to an ASCII Based Operating System

- 1. Moving files from Z/OS to an ASCII based platform may generate JavaScript errors when viewing the file with a browser. If you don't get the JavaScript errors, then you might see some strange characters in the output or the actual HTML tags as a result of the output wrapping. There are two separate approaches to moving files from Z/OS to an ASCII based platform. We will discuss both approaches below starting with creating an EBCDIC file and downloading to an ASCII based platform.

EBCDIC

HTML

A: Use the HTML option PATH= with the URL=NONE sub-option when generating graphics or when creating a table of contents or a frame file with the CONTENTS=, and the FRAME= options. The URL=NONE sub-option will prevent ODS from hard-coding the operating system data set name in the embedded links.

The PATH= option determines the location that we write the HTML files that create. This will also be the location where graphics files are stored unless the GPATH option is used. If both are used, the HTML files will be sent to the location specified with the PATH= option, and the graphics files will be sent to the location used with the GPATH= option.

B: Specify the URL= options on the BODY=, CONTENTS=, or FRAME= options with the name of the file that will be used in the links on the ASCII based server. Specify the RS=NONE which will allow us to write 1 record at a time.

Example1 .

```
FILENAME PDSE '.SAS.PDSE' DSNTYPE=LIBRARY DISP=(NEW,CATLG,DELETE)
DSORG=PO;

ODS HTML PATH=PDSE(URL=NONE)
      BODY='BODY' (URL="BODY.HTML")
      CONTENTS='CONTENTS' (URL="CONTENTS.HTML")
      FRAME='FRAME' (URL="FRAME.HTML")
      RS=NONE;

PROC FREQ DATA=SASHELP.CALSS;
RUN;

ODS _ALL _ CLOSE;
```

Example2.

```
// EXEC SAS
//PDSE DD DSN=PDSE.LIB,DISP=OLD
//SYSIN DD*

ODS HTML PATH=PDSE(URL=NONE)
      BODY='BODY' (URL="BODY.HTML")
      CONTENTS='CONTENTS' (URL="CONTENTS.HTML")
      FRAME='FRAME' (URL="FRAME.HTML")
      RS=NONE;

PROC PINT DATA=SASHELP.CLASS;
RUN;

ODS HTML CLOSE;
```

C: Move the HTML file as ASCII so that FTP or the communication software used will generate the correct end of line characters for the files . Be sure to give the file the .HTM or the .HTML file extension which will allow the registered application for this file type such as the browser.

2. The alternative to creating EBCDIC files as the above is to generate ASCII files. All non PDF files created with ODS are generated as EBCDIC by default on the mainframe. The TRANTAB= option can be used to create ASCII files on Z/OS with the various ODS destinations with the exception of the PDF destination which is generated in ASCII on all operating systems. When creating the file as ASCII, we will use the same steps as the above with the exception of using the TRANTAB= option in place of the RS=NONE option, and how the file is moved. The file will be moved as binary or image rather than ASCII.

ASCII

/* Generating HTML files */

```
FILENAME PDSE ` .SAS.PDSE' DSNTYPE=LIBRARY
DISP=(NEW,CATLG,DELETE) DSORG=PO;

ODS HTML PATH=PDSE(URL=NONE)
      BODY=' BODY' (URL="BODY.HTML")
      CONTENTS=' CONTENTS' (URL="CONTENTS.HTML")
      FRAME=' FRAME' (URL="FRAME.HTML")
      TRANTAB=ASCII;

PROC FREQ DATA=SASHELP.CLASS;
RUN;
ODS HTML CLOSE;
```

/* Generating Postscript files */

```
FILENAME OUTPUT ` .SAS.PRT' DISP=NEW RECFM=VB LRECL=8196 BLKSIZE=27998;

ODS PRINTER FILE=OUTPUT;
PROC PRINT DATA=SASHELP.CLASS;
RUN;

ODS PRINTER CLOSE;
```

/* Generating RTF/XML/CSV files */

```
FILENAME PDSE ` .SAS.PDSE' DSNTYPE=LIBRARY DISP=(NEW,CATLG,DELETE) DSORG=PO;

ODS RTF BODY=' SASCTP.SAS.SOURCE(RTF)' TRANTAB=ASCII;
ODS XML BODY=' SASCTP.SAS.SOURCE(XML)' TRANTAB=ASCII;
ODS CVS BODY=' SASCTP.SAS.SOURCE(CSV)' TRANTAB=ASCII;

PROC FREQ DATA=SASHELP.CLASS;
```

```
RUN;
```

```
ODS _ALL _ CLOSE;
```

- ✓ Move the files as binary files when using communication software. This will prevent the communication software from adding extra end-of-line characters to the file.

Creating Other ODS Files to be will be Viewed on an Z/OS Web server

1. The RS=NONE option should be specified which builds one record at a time. This allows the file to be structured and prevents undesired wrapping of the HTML tags which could cause problems.

2. When creating a frame file on Z/OS, many web servers require that the path of the file have single quotes around the PDSE and the member name. The links that we create in the contents and pages files do not have these quotes around the PDSE and the member names. Therefore, this might cause problem when attempting to load the contents and the pages files within a frame. To avoid this problem, using HFS is a viable alternative as well as using the MVHTML tagset to generate output for HTML.

3. Use the suggested DCB parameters.

/* Sending Postscript output directly to the printer. */

```
FILENAME LOCAL SYSOUT=A DEST=CHPLJJ43;  
ODS PRINTER FILE=LOCAL;  
PROC PRINT DATA=SASHELP.CLASS;  
RUN;  
ODS PRINTER CLOSE;
```

/* Viewing/RTF/XML/CSV output on an Z/OS web server */

```
FILENAME PDSE `'.SAS.PDSE' DSNTYPE=LIBRARY DISP=(NEW,CATLG,DELETE) DSORG=PO;  
ODS RTF BODY='SASCTP.SAS.SOURCE(RTF)' RS=NONE;  
ODS XML BODY='SASCTP.SAS.SOURCE(XML)' RS=NONE;  
ODS CSV BODY='SASCTP.SAS.SOURCE(CSV)' RS=NONE;  
  
PROC FREQ DATA=SASHELP.CLASS;  
RUN;  
  
ODS _ALL _ CLOSE;
```

Creating Files with HFS that are Viewed on a Z/OS server

A: Use the ODS HTML options PATH= with the URL=NONE sub-option. This will prevent ODS from hard-coding the OS data set name in the contents and the frame files when we prepare to make the links. This will also be the location where graphics files are stored unless the GPATH statement is used. If both are used, then the HTML files will be sent to the location with the PATH= option, and the graphics files will be sent to the location used with the

/* Syntax for generated HFS on an Z/OS Web server */

```
OPTIONS FILESYSTEM='HFS' ;

FILENAME ODSOUT '/U/USERID/PUBLIC_HTML' ;

ODS HTML PATH= ODSOUT(URL=NONE)
           BODY='BODY.HTML'
           CONTENTS='CONTENTS.HTML'
           FRAME='FRAME.HTML' ;

ODS RTF BODY='/U/SASCTP/PUBLIC_HTML/BODY.RTF' ;
ODS XML BODY='/U/SASCTP/PUBLIC_HTML/BODY.XML' ;
ODS CSV BODY='/U/SASCTP/PUBLIC_HTML/BODY.CSV' ;

PROC FREQ DATA=SASHELP.CLASS ;
RUN ;

ODS _ALL _ CLOSE ;
```

Creating Files with HFS that are Viewed on an ASCII Web Server

- A:** Use the HTML options PATH= with the URL=NONE sub-option. This will prevent us from hard-coding the OS data set name in the contents and the frame when we prepare to make the links. This will also be the location where graphics files are stored unless the GPATH statement is used. If both are used, then the HTML files will be sent to the location with the PATH= option, and the graphics files will be sent to the location used with the GPATH= option.
- B:** Use the TRANTAB=ASCII option and move the file as binary to the ASCII based Web server to prevent adding extra NL characters to the file.

/* Syntax for generating HFS on an ASCII based operating system */

```
OPTIONS FILESYSTEM='HFS' ;
FILENAME ODSOUT '/U/USERID/PUBLIC_HTML' ;

ODS HTML PATH= ODSOUT
        BODY=' BODY.HTML '
        CONTENTS=' CONTENTS.HTML '
        FRAME=' FRAME.HTML ' ;
        TRANTAB=ASCII ;

ODS RTF BODY=' /U/SASCTP/PUBLIC_HTML/BODY.RTF '
        TRANTAB=ASCII ;

ODS XML BODY=' /U/SASCTP/PUBLIC_HTML/BODY.XML '
        TRANTAB=ASCII ;

ODS CSV BODY=' /U/SASCTP/PUBLIC_HTML/BODY.CSV '
        TRANTAB=ASCII ;

PROC FREQ DATA=SASHELP.CLASS ;
RUN ;

ODS _ALL _ CLOSE ;
```

Creating Graphics with ODS and SAS/Graph

Graphics can be generated with SAS/Graph and the various ODS destinations as well as with ODS Graphics which became production with SAS 9.2. When creating graphics with SAS/Graph and ODS destinations such as RTF and PDF, the image is embedded within the file. However, the HTML destination generates a separate file for the image and adds an image tag to the HTML file which points to this image file. So that the image is located by the browser, it's a good idea to use either the PATH= or GPATH= statement which will specify the location where the image is stored. With SAS 9.2, the device will not need to be specified. The device is handled internally based on the destination when creating the image with SAS/Graph.

If you are creating output with ODS Graphics, you can generate default graphics from one of the many statistical procedures that it supports. All that is needed is an ODS Graphics statement. This requires a SAS/Graph license with SAS 9.2 and beyond.

HTML

```
FILENAME ODSOUT '.SAS.HTML' DISP=(NEW,CATLG,DELETE) RECFM=VB LRECL=8196
BLKSIZE=27998 DSNTYPE=LIBRARY ;
```

```
GOPTIONS RESET=ALL DEVICE=PNG;
ODS HTML PATH=ODSOUT(URL=NONE)
          BODY='BODY' (URL="BODY.HTML")
          CONTENTS='CONTENTS' (URL="CONTENTS.HTML")
          FRAME='FRAME' (URL="FRAME.HTML")
          RS=NONE;
```

```
PROC GCHART DATA=SASHELP.CLASS;
VBAR AGE;
RUN;
QUIT;
```

```
ODS HTML CLOSE;
```

RTF

```
FILENAME ODSOUT 'SASXXX.SAS.RTF' DISP=(NEW,CATLG,DELETE) RECFM=VB LRECL=8196
BLKSIZE=27998;
```

```
GOPTIONS RESET=ALL DEVICE=EMF
ODS RTF FILE=ODSOUT RS=NONE;
```

```
PROC GCHART DATA=SASHELP.CLASS;
VBAR SEX;
RUN;
QUIT;
```

```
ODS RTF CLOSE;
```

PDF

```
FILENAME ODSOUT 'SASXXX.SAS.PDF' DISP=(NEW,CATLG,DELETE) RECFM=VB LRECL=259
BLKSIZE=27998;
```

```
GOPTIONS RESET=ALL DEVICE=PDFC;
ODS PDF FILE=ODSOUT RS=NONE;
```

```
PROC GCHART DATA=SASHELP.CLASS;
VBAR SEX;
RUN;
QUIT;
```

```
ODS PDF CLOSE;
```

- ✓ When moving graphic files generated with SAS/Graph and ODS, be sure to always move this graphics file as a **Binary** file when moving to an ASCII Web server or operating system.

Creating Postscript Files on Z/OS

1. The recommended DCB information for files generated with ODS on Z/OS is RECFM=VB, LRECL=8196 and BLKSIZE=27998. This happens to be the default when using the FILENAME statement within SAS to allocate the PDSE. The DCB characteristics are not cut in stone, but work very well.
 - A. When generating Postscript files with output from SAS/GRAPH, the option PRINTERPATH=POSTSCRIPT system option should also be specified.
 - B. Postscript files are generated as EBCDIC on the mainframe, therefore, if they are moved to an ASCII platform the files are moved as ASCII/text. Also a .PS extension should be given to the file when downloaded.

Generating Postscript files

```
FILENAME OUTPUT '..SAS.PRT' DISP=NEW RECFM=VB LRECL=8196 BLKSIZE=27998;

ODS PRINTER FILE=OUTPUT PRINTERPATH=POSTSCRIPT;
PROC PRINT DATA=SASHELP.CLASS;
RUN;

ODS PRINTER CLOSE;
```

Generating PDF Files on Z/OS

There are many new exciting features for PDF in SAS 9.2. There is the ability to apply passwords as well as the ability to display how many nodes to display at one time in the bookmark list. There is also individual border control over cells. Below are some of the specifics to take into consideration when creating PDF file. Beginning in SAS 9.0, PDF files are compressed unlike the previous releases and the fonts are embedded.

- 1) The largest record for a PDF file is 255 and 4 bytes for the RDW which is a total of 259. The suggested DCB information for a PDF file is RECFM=VB, LRECL=259 and a BLKSIZE=27998.

SAS 8.2 and PDF

- 1) Out of space conditions may exist when generating PDF files from procedures or a data step that generate large amounts of output. The PDF files can get very large and the way that they are handled on Z/OS can generate an out of space condition. Usually a B37 abend will accompany the out of space condition when generating the PDF files. Below are some tips for getting around this problem on SAS 8.2

- A) Apply the 8.2 PDF hotfix which will use basically the same architecture as SAS 9.1 uses to generate the file.
- B) The internals of ODS and PDF is that when output is created, the output is spooled to a temporary file by piping the output to '&temp'. This generates temporary OS data sets which are prefixed SYS00? for each page of output. When an out of space condition occurs, it will occur in the temporary files that are spooled.

To add space to these files, increase the allocation for the FILESPPRI= and the FILESPSEC= options which handles space for the primary and secondary allocation of temporary file created in the SAS session. These values have a default value of 1 cylinder for the primary and the secondary allocations.

/* Generating PDF files */

```
FILENAME OUTPUT '..SAS.PRT' DISP=NEW RECFM=VB LRECL=259 BLKSIZE=27998;

ODS PDF FILE=OUTPUT;
PROC PRINT DATA=SASHELP.CLASS;
RUN;

ODS PDF CLOSE;
```

- ✓ The PDF file is created as a text file on Z/OS, therefore when moving the file to an ASCII host, then file will need to be moved as binary .

Tips for E-mailing Files Generated from ODS and Z/OS

- 1) Emailing HTML from Z/OS can be done a couple of ways. The first method writes the output directly to the body of the email which will work with the HTML destination.
 - A. The first method discussed is to email the HTML file directly without creating an intermediate file. This is done by setting the content-type to text/html with the TYPE= option on the FILENAME statement along with the EMAIL access method. For this method to work, the EMAILSYS option will need to be set to SMTP and the EMAILHOST option will be set to the name of the mail server.

```

FILENAME TEMP EMAIL T0='JOEBLOW@SOMEWHERE.COM'
                        SUBJECT="THIS IS A TEST"
                        TYPE='TEXT/HTML';

ODS HTML FILE=TEMP RS=NONE;
PROC PRINT DATA=SASHELP.CLASS;
RUN;

ODS HTML CLOSE;

```

- B. The second method discussed creates an intermediate file and sends the file as an attachment. The intermediate file generated with ODS cannot be a temporary file generated with &temp in 8.2, otherwise errors are generated attempting to email the files as attachments.

```

FILENAME TEMP  'SASCTP.SAS.HTML' DISP=OLD;

ODS HTML FILE=TEMP RS=NONE;
  PROC PRINT DATA=SASHELP.CLASS;
  RUN;
ODS HTML CLOSE;

```

```

FILENAME NEW EMAIL T0='JOEBLOE@SOMEWHERE.COM'
ATTACH=("SASCTP.SAS.HTML" ct="text/html" ext="html");
DATA _NULL _;
  FILE NEW;
  PUT 'THIS IS A TEST';
RUN;

```

- C. To send the output as text, the argument to the TYPE= option can be specified as TEXT/PLAIN. The procedure PRINTTO can then be used to send the out to the mailer. This might be an alternative to HTML if you need smaller files.

```

FILENAME TEMP EMAIL T0='JOEBLOW@SOMEWHERE.COM'
                        TYPE='TEXT/PLAIN';

PROC PRINTTO PRINT=TEMP;

PROC PRINT DATA=SASHELP.CLASS;
RUN;

PROC PRINTTO;
RUN;

```

- 2) Emailing RTF files can be done by sending the files as attachments. The only ODS option needed is the RS=NONE option.

```

FILENAME RTFFILE 'SASXXX.SAS.TEST' DISP=(NEW,CATLG,DELETE)
RECFM=VB LRECL=8196 BLKSIZE=27998;

```

```

ODS RTF FILE=RTFFILE RS=NONE;

PROC PRINT DATA=SASHELP.CLASS;
RUN;

ODS RTF CLOSE;

FILENAME TEMP EMAIL TO='JOEBLOE@WNT.SAS.COM'
                    SUBJECT="TESTING EMAIL INTERFACE"
                    ATTACH=("SASXXX.SAS.TEST" type="application/msword"
EXTENSION="RTF" );

DATA _NULL_;
    FILE TEMP;
    PUT 'THIS IS A TEST';
RUN;

```

- 3) To email PDF files, the content-type is set to plain/text. Below is example code for emailing PDF files.

```

FILENAME PDFFILE 'SASXXX.SAS.TEST' DISP=(NEW,CATLG,DELETE)
RECFM=VB LRECL=259 BLKSIZE=27998;

ODS PDF FILE=PDFFILE;

PROC PRINT DATA=SASHELP.CLASS;
RUN;

ODS PDF CLOSE;

FILENAME TEMP EMAIL TO='JOEBLOE@WNT.SAS.COM'
                    SUBJECT="TESTING EMAIL INTERFACE"
                    ATTACH=("SASXXX.SAS.TEST" CT='application/pdf'
EXTENSION="PDF" );

DATA _NULL_;
    FILE TEMP;
    PUT 'THIS IS A TEST';
RUN;

```

FTP Output from Z/OS

You can FTP output easily to and from Z/OS using the FTP access method via the FILENAME statement. Using the FTP access method provides a one step method of sending output to the server, or one or more platforms. There are a couple of ways that we can send and receive output using the FTP access method. If we are sending output to an ASCII platform, the output can be sent in either ASCII format or BINARY depending on the options specified on the ODS statement.

- 1) If you specify the TRANTAB=ASCII option on the ODS statement to create ASCII output, the output should be moved in binary format. You can do this by adding the RECFM=S option. Specifying stream as the value will allow you to send this in binary or image format.

```

FILENAME MOVE FTP ` /U/USERID/TEST'
  HOST=' 140.14.93'
  USER=' SAXXX'
  PASS=' XXXXXX'
  RECFM=S
  LRECL=8196
  DEBUG;

```

```
ODS HTML FILE=MOVE;
```

```
PROC PRINT DATA=SASHELP.CLASS;
RUN;
```

```
ODS HTML CLOSE;
```

- 2) When the RS=NONE ODS option is specified without the TRANTAB= option the output is generated in EBCDIC format, therefore the file would need to be moved as text or ASCII. This can be done by adding the RECFM=V option on the FILENAME statement with the FTP access method.

```

FILENAME MOVE FTP ` /U/USERID/TEST'
  HOST=' 140.14.93'
  USER=' SAXXX'
  PASS=' XXXXXX'
  RECFM=V
  LRECL=8196
  DEBUG;

```

```
ODS HTML FILE=MOVE;
```

```
PROC PRINT DATA=SASHELP.CLASS;
RUN;
```

```
ODS HTML CLOSE;
```

Creating Excel Output On Z/OS

ODS does not generate native Excel files, however, it can generate files which can be read with Excel such as XML, HTML ad CSV. If you have at least SAS 9.1 and Excel 2002, the best method of generating output which can be used with Excel is to use the Tagsets.ExcelXP destination. The output of this destination emulates the SpreadsheetML markup language which Excel generates. Some of the benefits of this destination is:

- 1) This destination allows you have multiple worksheets per workbook as a default.
- 2) It allows you to drive the entire worksheet using ODS Markup options. In the current version of the tagset which follows the 2003 spec of the SpreadsheetML markup language, images are not supported.

The other destination which we recommend in SAS 9.1.3 is the MSOffice2k Destination. This destination generates HTML which can be used with Excel as well. This destination

allows graphics, however by default does not have options. The destination is very flexible since it is HTML as far as style changes.

Installation

The version of the ExcelXP tagset installed with SAS 9.1.3sp4 is v1.28. Because this is an actively developed tagset, you will not have the most current version of the tagset unless you download it from the Base R&D page below. This latest version includes both enhancements and fixes. There are some characters in the tagset which can cause problems for some installations depending on the codepage or emulation. These characters are the left and right bracket and the carrot “[] ^”. If you have a problem with these characters, contact Technical Support and a binary item store can be sent rather than the tagset.

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

You can install this tagset to a SAS library and use it by adding the DD or Libname statement along with an ODS PATH statement which can add the library to the default search path. The default path only includes SASUSER and SASHELP. The below example demonstrates saving the ExcelXP updated tags to an OS data set which is allocated by the DD or fileref EXCELTAG. This file in %Included which will execute and run the PROC TEMPLATE code that compiles and updates the ExcelXP tagset. The DD or libref EXCELIB is used as the location where the tagset is compiled to. To compile and read the ExcelXP from this destination, an ODS PATH statement is added with the PREPEND option to add this path to the beginning of the search path. For more information on installation of the ExcelXP tagset, see the below.

<http://support.sas.com/kb/32/394.html>

Example.

```
LINAME EXCELIB '.EXCEL.LIB' DISP=NEW;
FILENAME EXCLTAG ".EXCEL.TAGS";
FILENAME OUTPUT ".ODS.OUTPUT";

ODS PATH(PREPEND) EXCELIB.TEMPLAT(UPDATE);

%INCLUDE EXCLTAG;

ODS TAGSETS.EXCELXP FILE=OUTPUT RS=NONE;

PROC PRINT DATA=SASHELP.CLASS;
RUN;

ODS TAGSETS.EXCELXP CLOSE;
```

Sample JCL

```
// EXEC SAS
//EXCELIB DD DSN=EXCEL.LIB,DCB=(RECFM=FS,LRECL=6144,BLKSIZE=6144),
//          SPACE=(CYL,(1,1),DISP=(NEW,CATLG,DELETE)
//EXCELTAG DD DSN=EXCEL.TAGS,DISP=OLD
//OUTPUT DD DSN=ODS.OUTPUT, DCB=(RECFM=VB,LRECL=8196,BLKSIZE=27998),
//          SPACE=(CYL,(1,1),DISP=(NEW,CATLG,DELETE)
//SYSIN DD *

ODS PATH(PREPEND) EXCELIB.TEMPLAT(UPDATE);

%INCLUDE EXCELTAG;

ODS TAGSETS.EXCELXP FILE=OUTPUT RS=NONE;

PROC PRINT DATA=SASHELP.CLASS;
RUN;

ODS TAGSETS.EXCELXP CLOSE;
```

Options and ExcelXP

There are options which can be applied using the options parameter on the ODS statement if you are using at least 9.1.3. These options allow you to drive essentially the entire Excel worksheet and workbook. This involves everything from naming the worksheet, adding filters, freezing headers, setting the page setup items and more. To get the complete syntax along with the options and the usage, add the options `Doc="Help"`.

```
ODS TAGSETS.EXCELXP FILE=OUTPUT options(embedded_titles="yes"
                                         Orientation="landscape"
                                         Autofilters="header"
                                         Doc="help") RS=NONE;

PROC PRINT DATA=SASHELP.CLASS;
RUN;

ODS TAGSETS.EXCELXP CLOSE;
```

Excel and SAS 8.2

To create output for Excel on Z/OS in SAS 8.2, you have the option of using either the HTML or CSV destination. If you need formatting such as fonts and colors, then the HTML destination will be a good choice. If size is a factor, then perhaps the CSV destination will be a good choice, however, this does not allow any formatting. The `RS=NONE` option should be specified which will write one record at a time which will prevent wrapping and an unstructured HTML file.

```
filename temp '.test.html' recfm=vb lrecl=8196 blksize=27998 disp=new;
```

```
ods html file=temp rs=none;

proc print data=sashelp.class;
run;

ods html close;
```

Creating XML files on Z/OS

Creating XML files on Z/OS can be done using the ODS XML or ODS MARKUP statements, the Tagsets.ExcelXP, and the XML libname engine. Which one you would use depends on your needs. The output generated by all 3 methods mentioned can be modified with the ODS Markup language to transform the output generated. Another distinguishing factor between the methods mentioned is that the XML libname engine will allow you to both read and write XML files. There are some predefined XML types which allow you to generate XML for specific targets such as MS Access, ODM, Export and a few other types using the XMLTYPE= or TAGSET= options on the libname statement.

The DCB information for the XML file generated is much the same as the ODS files. The RECFM=VB, LRECL=8196, and BLKSIZE=27998. You will want to make sure that it is large enough so that the output is not get truncated. If the RECFM= is not VB, and error is generated.

XML Engine

```
FILENAME TEST XML '.TEST.XML' RECFM=VB LRECL=8196 BLKSIZE=27998 DISP=NEW;

PROC COPY IN=SASHELP OUT=TEST;
SELECT CLASS;
RUN;
```

ODS XML

```
FILENAME TEST XML '.TEST.XML' RECFM=VB LRECL=8196 BLKSIZE=27998 DISP=NEW;

ODS XML FILE=TEST RS=NONE;
PROC PRINT DATA=SASHELP.CLASS;
RUN;
ODS XML CLOSE;
```

JCL Sample

```
// EXEC SAS
//OUTPUT DD DSN=XML.OUTPUT, DCB=(RECFM=VB,LRECL=8196,BLKSIZE=27998),
//          SPACE=(CYL,(1,1),DISP=(NEW,CATLG,DELETE)
//SYSIN DD *

ODS XML FILE=TEST RS=NONE;
PROC PRINT DATA=SASHELP.CLASS;
RUN;
ODS XML CLOSE;
```

To read XML files, the XML Engine is the method used. By default the XML engine can read generic XML files which are rectangular. If the files are not rectangular, an XMLMap is needed to instruct the engine on how to create columns and rows. The XML Mapper product can be used to create the an XML Map dynamically, although this product is not supported for the mainframe. The XML Mapper is a stand alone product, therefore, you can install this on the PC or UNIX and create a map from the XML file or schema and FTP the resulting XMLMap file to the mainframe. For more information, see the below url.

<http://support.sas.com/rnd/base/xmlengine/index.html>

Version 8.2 on the mainframe had problems when it came to reading an XML file that it produced. This is because it created a file which was absent from any type new line marker. The SAS 8.2 hotfix which also allowed XML Engine to become production for reading files as well will take care of this problem. This hotfix allows the ability to apply XMLMaps with SAS 8.2. The hotfix is 82bc01.