Paper 55

Getting Your SAS® Program to do Your Typing for You!

Nancy K. Wilson, Ally Financial

ABSTRACT

Do you have a SAS® program that requires adding file names to the input every time you run it? Aren't you tired of having to check for the files, check the names and type them in? Check out how this SAS® Enterprise Guide Project checks for files, figures out the file names and eliminates the need for having to type in the file names for the input data files!

INTRODUCTION

SAS® data set names within a folder location can easily be displayed using SAS® ‘LIBNAME’ statement and a PROC CONTENTS. What if the data files you need are not SAS® data sets? This paper will use the ‘X’ Command to demonstrate how your PC SAS® or UNIX SAS® program can read a variable number of file names in a folder and use those names as input for your SAS process.

BACKGROUND (IDENTIFYING THE CHALLENGE)

A standard report for a business team is run ‘upon request’ and must use the most recent data files provided by the members of the team. That team creates these text files in a standard format and saves them on a network shared folder: **‘H:\IO\’**. The file names are consistent: ‘dflist#.txt’ where # is a number. There is no limit on the number of input files.

Each time the process was executed the folder containing the data files needed to be manually reviewed and the SAS® code needs to be updated with the complete list of file names.

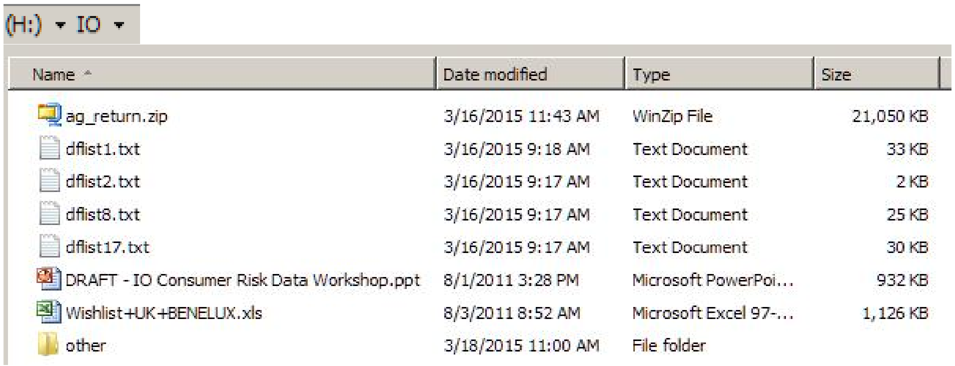
It didn’t take long for this to become a tedious process prone to errors!

ORIGINAL CODE:

```
PROC DATASETS ;  DELETE orig_widgit_count ;   run;
      *clean up data set for append below;
%macro readit(var);
      DATA temp;
            LENGTH        Count   3  order_cd $3  Unit_Name $16;
            FORMAT        Count   3. order_cd $3. Unit_Name $16.;
            INFORMAT      Count   3. order_cd $3. Unit_Name $16.;
            INFILE "H:\IO\&var"   LRECL=100  TRUNCOVER;
            INPUT
                  @1 count 3.
                  @5 order_cd $3.;
            UNIT_NAME=UPCASE("&var");
      RUN;
PROC APPEND BASE=Orig_widgit_count data=temp force;    run;
%mend readit;

run;

%readit(dflist1.txt);run;
%readit(dflist2.txt);run;
%readit(dflist8.txt);run;
%readit(dflist17.txt);run;
**Add new row for every file!;
```

These rows need to be manually updated for each execution!

FOLDER CONTENTS:



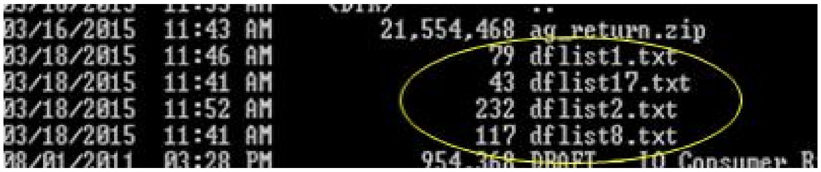
(H:) ▾ IO ▾

Name	Date modified	Type	Size
ag_return.zip	3/16/2015 11:43 AM	WinZip File	21,050 KB
dflist1.txt	3/16/2015 9:18 AM	Text Document	33 KB
dflist2.txt	3/16/2015 9:17 AM	Text Document	2 KB
dflist8.txt	3/16/2015 9:17 AM	Text Document	25 KB
dflist17.txt	3/16/2015 9:17 AM	Text Document	30 KB
DRAFT - IO Consumer Risk Data Workshop.ppt	8/1/2011 3:28 PM	Microsoft PowerPoi...	932 KB
Wishlist+UK+BENELUX.xls	8/3/2011 8:52 AM	Microsoft Excel 97-...	1,126 KB
other	3/18/2015 11:00 AM	File folder	

THE SOLUTION!

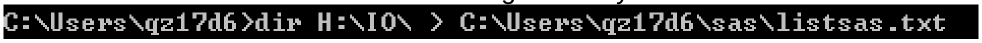
MS DOS COMMAND: DIR

The command ‘dir’ will display the available files and folders in a directory.



```
03/16/2015  11:43 AM        21,554,468 ag_return.zip
03/18/2015  11:46 AM                79 dflist1.txt
03/18/2015  11:41 AM                43 dflist17.txt
03/18/2015  11:52 AM               232 dflist2.txt
03/18/2015  11:41 AM               117 dflist8.txt
08/01/2011  03:28 PM           954,368 DRAFT - IO Consumer R
```

And the results can be routed to a file using the ‘>’ symbol:



```
C:\Users\qz17d6>dir H:\IO\ > C:\Users\qz17d6\sas\listsas.txt
```

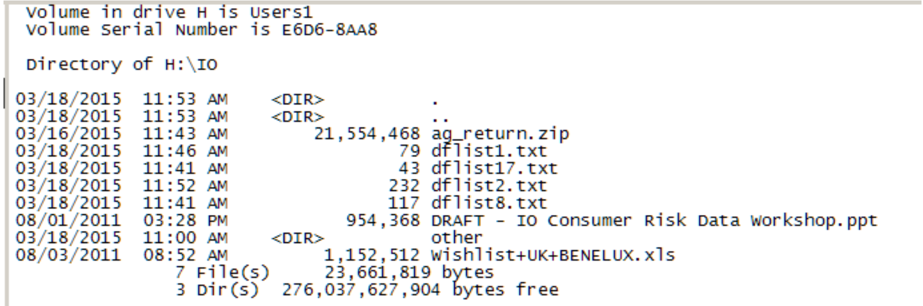
USE SAS® TO RUN THE COMMAND:

We can use the **X command** in SAS to run the ‘dir’ windows command and create text file. Once the file containing the list of files and folders is created, SAS can be used to read the file and determine which file names will be retained to be used in subsequent steps.

```
options NOXWAIT;run;

x 'dir H:\IO\ > C:\Users\qz17d6\sas\listsas.txt';run;
```

The option ‘NOXWAIT is needed so the command processor automatically returns to the SAS® session after the ‘dir’ command is executed.

The file ‘listsas.tst’ is created, and contains the results of ‘dir H:\IO\’:

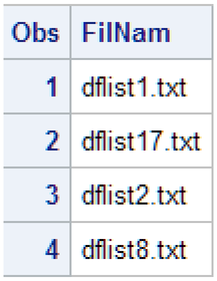


```
 Volume in drive H is Users1
 Volume Serial Number is E6D6-8AA8

 Directory of H:\IO

03/18/2015  11:53 AM    <DIR>          .
03/18/2015  11:53 AM    <DIR>          ..
03/16/2015  11:43 AM        21,554,468 ag_return.zip
03/18/2015  11:46 AM                79 dflist1.txt
03/18/2015  11:41 AM                43 dflist17.txt
03/18/2015  11:52 AM               232 dflist2.txt
03/18/2015  11:41 AM               117 dflist8.txt
08/01/2011  03:28 PM           954,368 DRAFT - IO Consumer Risk Data Workshop.ppt
03/18/2015  11:00 AM    <DIR>          other
08/03/2011  08:52 AM         1,152,512 Wishlist+UK+BENELUX.xls
               7 File(s)     23,661,819 bytes
               3 Dir(s)  276,037,627,904 bytes free
```

USE SAS® TO READ THE RESULTS:

Read the text file and keep the file names we need:

```
Data fillstS2;
   INFILE "C:\Users\qz17d6\sas\listsas.txt" TRUNCOVER;
   length var1 $80.;
   Input @1 var1 $80.;

   if index(var1,'dflist')>0  then
         do;
               FilNam= substr(var1,index(var1,'dflist'),20);
               output;
         end;
   else delete;
   drop var1;
run;

PROC PRINT data=fillstS2;   run;
```



Obs	FilNam
1	dflist1.txt
2	dflist17.txt
3	dflist2.txt
4	dflist8.txt

FINAL STEP: USE 'CALL EXECUTE' TO USE THE FILE NAMES IN THE CODE:

Use **call execute** within SAS to pass the list of file names to the SAS program to be used as input into the original macro. You no longer need to edit the code before running each time!

```
PROC DATASETS ;  DELETE widgit_count ;   run;
      *clean up data set for append below;
%macro readfil (var) ;
DATA temp;
    LENGTH        Count   3  order_cd $3  Unit_NM $15 ;
    FORMAT        Count   3. order_cd $3. Unit_NM $15. ;
    INFORMAT      Count   3. order_cd $3. Unit_NM $15. ;
    INFILE "H:\IO\&var"
        LRECL=100
        TRUNCOVER ;
    INPUT
        @1 count 3.
            @5 order_cd $3.         ;
            Unit_nm="&var" ;
            UNIT_NAME=UPCASE(unit_nm); drop UNIT_nm;
RUN;
PROC append base=widgit_count data=temp force;
%mend readfil; run;

data _null_ ;
 set fillstS2  ;
      * data set containing the names of the text files to be passed;
      * text file name found in varname: FilNam ;
call execute  ( '%readfil   ( %str(' ||compress(FilNam)|| ')   ) '  ) ;
run;
```

Check the log - Call Execute will generate some code, it's not pretty but it is the code from the macro with the input file name properly resolved. The PROC APPEND will execute, and then the next input file, and so on until the end.

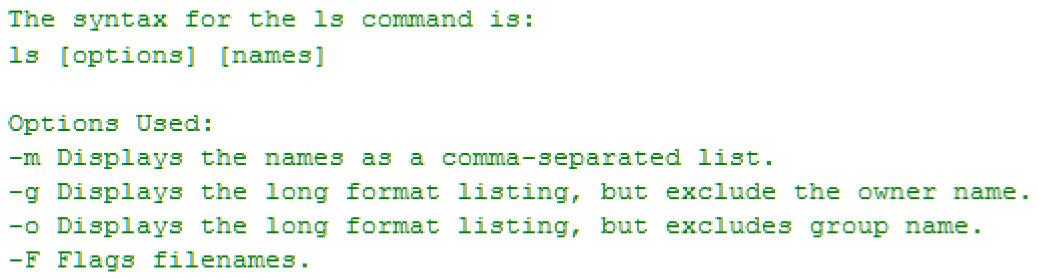
```
NOTE: CALL EXECUTE generated line.
    + DATA temp;     LENGTH        Count    3  order_cd $3  Unit_NM $15 ;     FORMAT
Count   3. order_cd $3. Unit_NM $15. ;     INFORMAT      Count   3. order_cd $3. Unit_NM
$15.
;     INFILE "H:\IO\dflist1.txt"         LRECL=100         TRUNCOVER ;
2    + INPUT         @1 count 3.         @5 order_cd $3.         ;
Unit_nm="dflist1.txt" ;         UNIT_NAME=UPCASE(unit_nm); drop UNIT_nm; RUN;
2    +
PROC append base=widgit_count data=temp force;
3    + DATA temp;     LENGTH        Count    3  order_cd $3  Unit_NM $15 ;     FORMAT
Count   3. order_cd $3. Unit_NM $15. ;     INFORMAT      Count   3. order_cd $3. Unit_NM
$15.
;     INFILE "H:\IO\dflist17.txt"         LRECL=100         TRUNCOVER ;
4    + INPUT         @1 count 3.         @5 order_cd $3.         ;
Unit_nm="dflist17.txt" ;         UNIT_NAME=UPCASE(unit_nm); drop UNIT_nm; RUN;
```

THE SOLUTION USING PC SAS®:

```
options NOXWAIT;run;
x 'dir H:\IO\ > C:\Users\qz17d6\sas\listsas.txt';
run;
Data fillstS2;
      INFILE "C:\Users\qz17d6\sas\listsas.txt" TRUNCOVER;
      length var1 $80.;
      Input @1 var1 $80.;

      if index(var1,'dflist')>0  then
            do;
                  FilNam= substr(var1,index(var1,'dflist'),20);
                  output;
            end;
      else delete;
      drop var1;
run;

PROC print data=fillstS2;run;

PROC datasets ;  delete widgit_count ;   run;
      *clean up data set for append below;
%macro readfil (var) ;
DATA temp;
    LENGTH        Count    3  order_cd $3  Unit_NM $15 ;
    FORMAT        Count    3. order_cd $3. Unit_NM $15. ;
    INFORMAT      Count    3. order_cd $3. Unit_NM $15. ;
    INFILE "H:\IO\&var"
        LRECL=100
        TRUNCOVER ;
    INPUT
        @1 count 3.
            @5 order_cd $3.         ;
            Unit_nm="&var" ;
            UNIT_NAME=UPCASE(unit_nm); drop UNIT_nm;
RUN;
PROC append base=widgit_count data=temp force;
%mend readfil; run;

data _null_ ;
 set fillstS2  ; * data set containing the names of the text file to be
passed;
      * text file name found in varname: FilNam ;
  call execute  ( '%readfil   ( %str(' ||compress(FilNam)|| ')   ) '  )
run;

PROC CONTENTS DATA=WIDGIT_COUNT ; RUN;
```

THE SOLUTION USING SAS® ON UNIX

The only differences between PC SAS® and UNIX are the system commands and the folder structures. In using the ‘`ls`’ command options to use are:



```
The syntax for the ls command is:
ls [options] [names]

Options Used:
-m Displays the names as a comma-separated list.
-g Displays the long format listing, but exclude the owner name.
-o Displays the long format listing, but excludes group name.
-F Flags filenames.
```

```
x ls -mgoF /model/qz17d6 > /model/qz17d6/listsas.txt;
run;

the location of the list of file C:\Users\qz17d6\sas\

Data fillstS2;
      INFILE "/model/qz17d6/listsas.txt" TRUNCOVER;
      length var1 $80.;
      Input @1 var1 $80.;

      if index(var1,'dflist')>0  then
            do;
                  FilNam= substr(var1,index(var1,'dflist'),20);
                  output;
            end;
      else delete;
      drop var1;
run;


PROC datasets ;  delete widgit_count ;   run;
      *clean up data set for append below;
%macro readfil (var) ;
DATA temp;
    LENGTH        Count   3  order_cd $3  Unit_NM $15 ;
    FORMAT        Count   3. order_cd $3. Unit_NM $15. ;
    INFORMAT      Count   3. order_cd $3. Unit_NM $15. ;
    INFILE "/model/qz17d6/&var"
        LRECL=100
        TRUNCOVER ;
    INPUT
        @1 count 3.
            @5 order_cd $3.         ;
            Unit_nm="&var" ;
            UNIT_NAME=UPCASE(unit_nm); drop UNIT_nm;
RUN;
PROC append base=widgit_count data=temp force;
%mend readfil; run;

data _null_ ;
 set fillstS2  ; * data set containing the names of the text file to be
passed;
      * text file name found in varname: FilNam ;
  call execute  ( '%readfil   ( %str(' ||compress(FilNam)|| ')   ) '  )
run;

PROC CONTENTS DATA=WIDGIT_COUNT ; RUN;
```

CONCLUSION

Once a SAS® program is complete there may be further ways to improve the process by eliminating manual steps!

CONTACT INFORMATION

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

Nancy K. Wilson
www.linkedin.com/in/nancy88wilson/

This presentation – including all updates – can be found here:
http://www.sascommunity.org/wiki/User:NancyWilson