

Paper 118-2011

Using SAS® to Enter Variables in a DOS Window

Brandon Harvey, Pearson, Iowa City, IA

Robert D. Parker, Pearson, Iowa City, IA

ABSTRACT

Do you have an old DOS application that prompts you to type responses in the DOS window? Let SAS do it for you. This paper covers all the code that you need to replace the manual process of key-entering responses to DOS window prompts. You can automate the entire process. Simply use SAS to build a text file of the responses you would have otherwise had to type into the DOS window manually and tell DOS to use that file. Add a few data driven macro techniques and you have a complete automated solution. While the solution presented below is specific to a particular DOS program (GF98), it may be applied to any DOS program that requires responses to prompts.

KEYWORDS

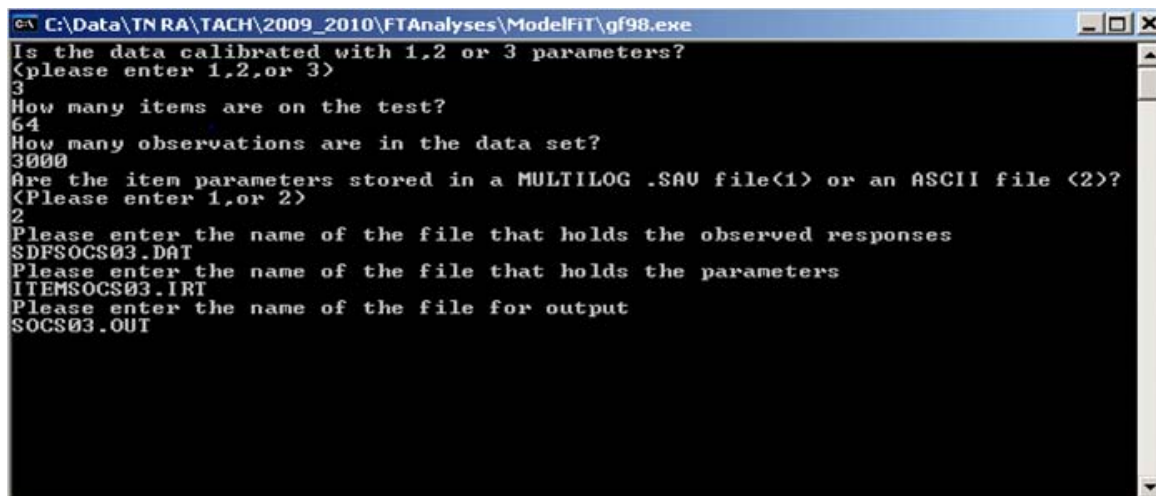
DOS, X commands, %LET, %MACRO, %SYMDEL, %DO, %END

INTRODUCTION

An automated solution to execute a DOS program was desired. The DOS program (GF98) required keyboard entry of responses to prompts issued by the program. In this particular case there were seven prompts that required responses before the program would execute. The desired outcomes of the automation were to reduce typing errors and speed up the process for multiple executions. This paper details a solution for automating the manual keyboard entry with the use of a DOS input/output redirector and using SAS to drive the process. This solution was executed running SAS 9.2 in the Windows XP environment.

1.0 THE FIRST ISSUE – REPLACING MANUAL KEY ENTRY WITH A FILE OF RESPONSES AND MAKING DOS RECOGNIZE THE FILE OF RESPONSES AS INPUT

A user required a specific program (GF98) which runs in a DOS window. In order to execute the program a user would have to respond to several prompts in the DOS window. There was a desire to automate the process so the user did not have to physically key in responses to the prompts. Below is an example of the prompts and responses for a test execution.

Example 1:

```
C:\Data\TN RA\TACH\2009_2010\FTAnalyses\ModelFIT\gf98.exe
Is the data calibrated with 1,2 or 3 parameters?
<please enter 1,2,or 3>
3
How many items are on the test?
64
How many observations are in the data set?
3000
Are the item parameters stored in a MULTILog .SAU file<1> or an ASCII file <2>?
<Please enter 1,or 2>
2
Please enter the name of the file that holds the observed responses
SDFSOC$03.DAT
Please enter the name of the file that holds the parameters
ITEMSOC$03.IRT
Please enter the name of the file for output
SOC$03.OUT
```

The first step to a solution is to create a file of responses to the prompts. That can be done by either keying them into Notepad, but keying is still required, or doing it in SAS with a program. Using a SAS program is the first step to automation. Below is an example of creating a file of responses in SAS.

Code 1.1

```
*****;
** CREATE THE CONTROL FILE FOR RUNNING GF98
*****;
DATA _NULL_;
  FILE "C:\GF98\CNTL.TXT";
  PUT "3" /
      "64" /
      "3000" /
      "2" /
      "SDFMATH03.DAT" /
      "ITEMMATH03.IRT" /
      "MATH03.OUT";
RUN;
```

It was easy to write a program to generate responses to the seven prompts and put them in a file. The challenge was to find a way to get the DOS program to recognize the file of responses rather than requiring entry via a keyboard.

THE SOLUTION TO THE FIRST ISSUE

Attempts to use multiple X commands from SAS did not work. A search of the Web and other sources did not reveal a solution. The solution was found in one sentence of an old dusty 1987 IBM DOS manual. The solution is to use a DOS input redirector (<). The left arrow (less than sign) < tells DOS to use whatever follows it as input. The solution was tested by entering the responses to the prompts in a text file. The file was named CNTL.TXT. The DOS program GF98 and the file of responses were stored in a folder on the C: drive named GF98. The GF98 program was run in a DOS window by entering at the DOS prompt **C:\GF98\GF98.EXE**. To run the program using the file of responses, CNTL.TXT, requires the command **C:\GF98\GF98.EXE <C:\GF98\CNTL.TXT**. The program executed successfully without having to key enter responses to any prompts.

2.0 THE SECOND ISSUE – GETTING SAS TO EXECUTE A DOS PROGRAM WITH A DOS INPUT REDIRECTOR

With the first issue solved, we attempted to automate the process from a SAS program by building the file of responses in a DATA step and invoking the DOS program using an X command. The output file of responses was named CNTL.TXT. An attempt was made to issue the same DOS command referenced previously using an X command in SAS. The X command used was **X C:\GF98\GF98.EXE <C:\GF98\CNTL.TXT**. The DOS program did not use the file of responses. It acted as if it did not recognize the input redirector or the file of responses.

THE SOLUTIONS TO THE SECOND ISSUE

There are two possible solutions for getting the DOS command, with the input redirector, to work when issued from SAS with an X command.

SOLUTION 1: The first solution is to put the command(s) in a .BAT file. Use a DATA step to create a .BAT file followed by an X command to run the .BAT file. You may also delete the BAT file when finished.

Code 2.1

```
*****;
** METHOD 1:  CREATE THE BATCH FILE FOR RUNNING GF98
*****;
DATA _NULL_;
  FILE "C:\GF98\BATFILE.BAT";
  PUT "C:\GF98\GF98.EXE <C:\GF98\CNTL.TXT";
RUN;
OPTIONS NOXWAIT;
*****;
** RUN GF98 USING THE BATCH FILE AND THEN DELETE THE BATCH FILE
*****;
X "C:\GF98\BATFILE.BAT";
X "DEL C:\GF98\BATFILE.BAT";
```

SOLUTION 2: The second solution came from SAS technical support. You must issue the X command followed by CMD to start the DOS command window prior to issuing the DOS command with an input redirector.

Code 2.2

```
*****;
** METHOD 2:  RUN GF98 USING WINDOWS CMD COMMAND PROCESSOR
*****;
X CMD /K "C:\GF98\GF98.EXE <C:\GF98\CNTL.TXT";
*****;
** SAS OPTIONS XWAIT AND XNOWAIT HAVE NO EFFECT.  *;
** /K KEEPS THE DOS WINDOW OPEN WHEN DONE.      *;
** /C CLOSSES THE DOS WINDOW WHEN DONE.         *;
*****;
```

3.0 ADDING THE BELLS AND WHISTLES – A DATA DRIVEN MACRO APPROACH

The code segments 3.1- 3.8 automate the program. The program will look in the directory folder, find all pairs of input files, and create the output file for each pair. By cutting and pasting the directory where the input files are located into the program, you can eliminate the chance of making a transcription error.

Code 3.1

Run the DELETE ALL GLOBAL MACROS code at the beginning of the program. This is a precaution to make sure old global macro variables are not reused.

```
*****;
** DELETE ALL GLOBAL MACROS
*****;
%MACRO DELVARS;
DATA VARS;
  SET SASHELP.VMACRO;
RUN;
DATA _NULL_;
  SET VARS;
  IF SCOPE='GLOBAL' THEN
    CALL EXECUTE ('||TRIM(LEFT(NAME))||');
RUN;
%MEND;
%DELVARS
```

Code 3.2

Clear the log & output windows and delete any SAS data sets in the WORK library. Assign the location of the input files to the global variable &DIR.

```
*****;
** CLEAR LOG, OUTPUT, AND DELETE WORK LIBRARY
*****;
DM 'LOG;CLEAR;OUT;CLEAR;';
PROC DATASETS LIBRARY=WORK KILL;
RUN;
*****;
%LET DIR = C:\DATA\TN RA\TACH\2009_2010\FTANALYSES\MODELFIT;
```

Code 3.3

Create the text file,"FILES.TXT", that lists all file names in the folder located at &DIR.

```
*****;
** USE X COMMANDS TO CREATE A TEXT FILE WITH EVERYTHING IN THE DIRECTORY
*****;
OPTIONS XMIN NOXWAIT;
X CHDIR &DIR.;
X DIR /B > "&DIR.\FILES.TXT";
OPTIONS NOXMIN XWAIT;
```

Code 3.4

This part of the code creates two data sets (FileNameDAT and FileNameIRT). The data sets will contain a list of the names of all the files in the directory that have the extension .DAT and .IRT respectively.

```
*****;
** CREATE TWO DATA SETS (1 FOR EACH INPUT FILE EXTENSION
** NEEDED). SORT THE DATA FILES AND THEN MERGE THEM
*****;
DATA WORK.FILESNAMEDAT WORK.FILENAMEIRT;
  INFILE "&DIR.\FILES.TXT" DLM="09"X DSD LRECL=250 PAD MISSEVER;
  FORMAT FILENAME $CHAR50.
         IRTFILE $CHAR50.
         DATFILE $CHAR50.;
  INPUT FILENAME;
  X=LENGTH(FILENAME);
  FILENAMEEXTENSION = SUBSTRN(FILENAME,X-3);
  IF FILENAMEEXTENSION = ".IRT" THEN DO;
    TEST = SUBSTRN(FILENAME,5,X-8);
    IRTFILE = FILENAME;
  END;
  IF FILENAMEEXTENSION = ".DAT" THEN DO;
    TEST = SUBSTRN(FILENAME,4,X-7);
    DATFILE = FILENAME;
  END;
  IF LOWCASE(FILENAMEEXTENSION) IN ('.IRT') THEN
    OUTPUT WORK.FILENAMEIRT;
  IF LOWCASE(FILENAMEEXTENSION) IN ('.DAT') THEN
    OUTPUT WORK.FILESNAMEDAT;
RUN;
```

Code 3.5

Process the two data sets so they can be merged to create a new data set (MERGED) that contains a list of all pairs of input files. This step ensures that only paired input files are processed. If only one input file exists instead of two, as expected, that input file is dropped from processing.

```
DATA WORK.FILESNAME DAT (DROP = IRTFILE);
  SET WORK.FILESNAME DAT ;
RUN;
DATA WORK.FILESNAME IRT (DROP = DATFILE);
  SET WORK.FILESNAME IRT ;
RUN;
PROC SORT DATA=WORK.FILESNAME IRT OUT=WORK.FILESNAME IRT;
  BY TEST;
RUN;
PROC SORT DATA=WORK.FILESNAME DAT OUT=WORK.FILESNAME DAT;
  BY TEST;
RUN;
DATA MERGED (KEEP = DATFILE TEST IRTFILE);
  MERGE WORK.FILESNAME DAT (IN=DAT) WORK.FILESNAME IRT (IN=IRT);
  BY TEST;
  IF DAT=1 AND IRT=1;
RUN;
```

Code 3.6

Create global variables of the common portion of the paired input files and one additional macro variable containing the total number input file pairs.

```
*****;
** CREATE A MACRO VARIABLE FOR EACH TEST AND THE NUMBER
** OF TEST
*****;
DATA _NULL_;
  SET WORK.MERGED END = LAST;
  CNT + 1;
  INDEX = COMPRESS('TEST' || PUT(CNT, 3.), ' ');
  /* GET THE NUMBER OF VARS */
  IF LAST THEN CALL SYMPUT('LAST', LEFT(TRIM(PUT(CNT, 3.))));
  CALL SYMPUT(INDEX, LEFT(TRIM(TEST)));
RUN;
```

Code 3.7

The first macro (RUNIT) loops through the paired input files. The second macro (RUNCK) is the same code shown in code segments 1.1 and 2.1. For all output files to be created, the RUNCK macro deletes any previously created version. The macro continues by creating a control file, a DOS BAT file, running the BAT file, and deleting the BAT file when finished.

```

OPTIONS NOXWAIT NOXMIN;
*****;
** LOOP THROUGH EACH TEST AND RUN GF98.EXE PROGRAM
*****;
%MACRO RUNIT;
  %DO I = 1 %TO &LAST.; /* LOOP THROUGH EACH TEST */

  %MACRO RUNCK(TEST); /* USES CODE SEGMENTS 1.1 AND 2.1 */
    ***** DELETE THE OUTPUT FILES FROM GF98 *****;
    X CHDIR &DIR.;
    X DEL C:\GF98\&TEST..OUT;
    X DEL C:\GF98\GOODFIT.LIS;
    *****;
    ** CREATE THE CONTROL FILE FOR RUNNING GF98
    *****;
    DATA _NULL_;
      FILE "&DIR.\&TEST._CNTL.TXT";
      PUT "3" /
          "64" /
          "3000" /
          "2" /
          "SDF&TEST..DAT" /
          "ITEM&TEST..IRT" /
          "&TEST..OUT" ;

    RUN;
    *****;
    ** CREATE THE BATCH FILE FOR RUNNING GF98
    *****;
    X CHDIR &DIR.;

    DATA _NULL_;
      FILE 'X.BAT';
      PUT "C:" /
          "CD &DIR." /
          "GF98.EXE <&TEST._CNTL.TXT";

    RUN;
    *****;
    ** RUN GF98 USING THE BATCH FILE AND DELETE THE BATCH FILE
    *****;
    X CD &DIR.;
    X X.BAT;
    X DEL X.BAT;
    X DEL GOODFIT.LIS;
  %MEND RUNCK;

  %RUNCK(&&TEST&I.)
%END;

/*THIS MACRO CYCLES THROUGH EACH COLUMN*/
%MEND RUNIT;
%RUNIT

```

Code 3.8

Delete any leftover files.

```
*****;
** DELETE LEFTOVER FILES
*****;
X CHDIR &DIR.;
X DEL FILES.TXT;
X DEL X.BAT;
```

Together, code segments 3.1-3.8 do all of the work for you by using a data driven macro approach. The program finds all the pairs of input files, creates all the control files needed, and runs them in a batch file.

CONCLUSION

Once a solution was developed and publicized it became apparent that others were struggling with the same issue. While not groundbreaking or revolutionary, this solution is very applicable to any DOS program that prompts for responses. Although technology continues to advance and applications become more sophisticated, many old DOS applications persist. They get the job done and are used regularly. While there is no actual measurement of the number of DOS programs that are still being used on a regular basis, there are many. This may be particularly true in academia. If you must use that old dusty DOS program with prompts and you want to automate it, the solution presented above should help you. This solution greatly reduces the possibility of error in responding to prompts and is much more efficient than manually running the DOS program.

REFERENCES

IBM Disk Operating System Version 3.30 First edition (April 1987).
Code in section 3.1 was first discussed by diTommaso 2003.

ACKNOWLEDGMENTS

Thanks to SAS Technical Support for their assistance with the X command and DOS redirector.

CONTACT INFORMATION

Brandon Harvey
Pearson
2510 N. Dodge Street, P.O. Box 30
Iowa City, Iowa 52245-9945

319-354-9200 x216158
Brandon.Harvey@Pearson.com

Bob Parker
Pearson
2510 N. Dodge Street, P.O. Box 30
Iowa City, Iowa 52245-9945

319-354-9200 x216298
Bob.Parker@Pearson.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.