

## Paper 100-2011

## Create a Main Program to Run Multiple SAS® Programs: Utilizing the %INCLUDE Statement

Lisa Mendez, SRA International Inc., San Antonio , TX.

### ABSTRACT

Learn a quick and easy way to run multiple SAS programs from one “Main” program. This method provides flexibility and helps with the organization of macro variables that are used across multiple programs. By using the %INCLUDE statement you can retrieve and run external SAS programs. This paper will help you prepare to write a Main program, organize your macro variables and library references, and utilize the %INCLUDE statement.

### INTRODUCTION

When I was tasked to review and re-write the MEPRS Early Warning And Control System (MEWACS) SAS programs, I found that the process included running multiple SAS programs one at a time, and also doing some outside manual manipulation. The process was time consuming and extremely exhaustive and if you were not familiar with the process you would have to rely on comprehensive, step-by-step directions. If you missed a step, you wouldn't know until later in the process, and then you would have to redo the process from the beginning. I wanted to automate everything, but it wasn't feasible to have one large SAS program, so I decided to create a Main program to run the SAS programs in a specific order. By using the %INCLUDE statement I only have to submit one SAS program and multiple SAS programs are executed.

### WHY CREATE A MAIN PROGRAM?

The initial MEWACS process contained 13 SAS programs. Each program input one or more text files, cleaned the data, and then output one or more text files. In addition, each program used various macro variables, some that were used across programs. You had to ensure you updated the macro variables for each program before running it, which was time consuming and redundant.

Although the process of running each program separately was feasible, it had some disadvantages. First, you had to monitor each program until it finished before you could run the next one. Second, manually running each program increased the likelihood of errors due to skipping a step or not updating a macro variable. Third, there was redundant coding in some of the programs, such as macro variables and library references. Lastly, some programs used the same input, which meant the data file was read in multiple times.

To eliminate these problems we can batch our programs together by writing one SAS program to call and run all of the others. Instead of creating one large program, I was able to keep the structure of the process somewhat the same, which is what my co-worker preferred. It took some convincing, but she agreed to the Main program concept as long as she was able to run one program by itself, if needed, and not have to find it in a larger program.

### PREPARATION

Before writing the main program, ensure that each separate program you want to include runs correctly. Review all SAS programs to be included for similar characteristics. For example, the programs I encountered had the same SAS code in each program:

- macro variables
- library references (LIBNAME)
- input files

By noting the similarities across the programs, you can then condense them into the main program to reduce redundancy.

## PROGRAM HEADER

All SAS programs should contain a program header. This will help identify which programs will be called and run. Figure 1 shows an example of a program header for a Main SAS program. You can see that this Main program will call in five SAS programs.

**Figure 1. Program Header Example**

```

/*****
/* Program Name: M4.0 P1 00 MAIN Program.sas */
/* Date Created: June 29, 2009 */
/* Author: Lisa Mendez */
/* Purpose: Runs the programs for the compliance memo */
/* Inputs: M4.0 P1 01 Raw Data.sas */
/*          M4.0 P1 02 Transmission Dates.sas */
/*          M4.0 P1 03 Compliance Memo Datasets.sas */
/*          M4.0 P1 04 Compliance Memo Graphs.sas */
/*          M4.0 P1 05 Compliance Memo Doc.sas */
/* Outputs: See individual SAS programs for output data */
/* ----- */
/* Macros: None */
/* ----- */
/* Notes: Must update MACRO VARIABLES each month */
/* ----- */
/* Date Modified: */
/* Modified by: */
/* Reason for Modification: */
/* Description of Modification: */
/* ----- */

```

## MACRO VARIABLES AND LIBRARY REFERENCES

In order to eliminate the need to update macro variables and library references in each program, the Main program will house all of them. Remember macro variables are used to store and manipulate character strings and must follow SAS naming rules. They are not the same as the DATA step variables and are stored in memory in a macro symbol table. Library references are created in order for SAS to read or write a SAS data set to a directory (or folder). SAS calls the directory or folder a library and assigns "nicknames" to these libraries. By coding the Main program to house all of the macro variables and LIBNAMEs, you will only need to update once before executing the Main program. Figure 2 shows an example of macro variables and library reference (LIBNAMEs) in a Main program. Comment blocks are used to further describe the macro variables.

Remember to remove all macro variables and library references from the included SAS programs and place them in the Main program. This will allow you to create, modify, and update all from one location.

**Figure 2. Macro variables and library references example**

```

/*-----*/
/*!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!*/
/*          MACRO VARIABLES TO BE UPDATED EACH MONTH */
/*!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!*/
%Let Path = D;
%Let ExtractDate = 201011_OCT;
/* YYYYMM_Mon = YYYY (year), FM (numeric fiscal month), Mon (fiscal month abbreviation) */
%Let FM = 11;
%Let FY = 2010; /* Current fiscal year */
%Let PFY = 2009; /* Previous fiscal year */
/*-----*/
/*          MACRO VARIABLES FOR COMPLIANCE MEMO DOCUMENT */
/*-----*/
%Let FYears = 1; /* Number of Fiscal Years to include in Compliance Memo */
%Let CurrentMonth = October; /* for Compliance Memo & execsummary*/
%Let ReportingPeriod = August 2010; /* for Compliance Memo & execsummary*/
%Let ExtractionDate = 16 October 2010; /* for Compliance Memo & execsummary*/
%Let CurrentDate1 = October 16, 2010; /* for Exec Summary */
%Let CurrentDate2 = 10/16/2010; /* for Exec Summary */
/*-----*/
/* GLOBAL LIBRARY REFERENCES */
/*-----*/
libname M4RawDS "%Path:\11970.163\Products\MEWACS\MEWACS4.0\%ExtractDate\Raw Data";
libname Files "%Path:\11970.163\Products\MEWACS\MEWACS4.0\Files for Comp Memo & Exec Sum";
/*-----*/

```

There are several host-options you can use directly in the %INCLUDE statement. One example is if you need to increase the length of the source code from the default of 256 characters. You can specify a longer length using the LRECL=xxxx option following the / (forward slash). For example:

```
%INCLUDE "&Path:\myfiles\sample_SAS_Code.sas" / lrecl=500;
```

For specific information on the options you can use in the %INCLUDE statement see the SAS documentation for your operating environment.

Figure 3, located on page 4, shows an example of %INCLUDE statements in a Main program. Each of the SAS programs reside on a network drive. The path is determined by a macro variable. Note the use of comments. This helps identify the beginning and end of each SAS program in the SAS log.

Also illustrated in Figure 3, is a %INCLUDE statement in a DATA step. This allows you to use conditional coding, such as IF-THEN-ELSE statements. In the example above, there are two SAS programs to be used for the Compliance Memo depending on the number of fiscal years, which is determined by the FYears macro variable.

The SAS log will show the code for the Main SAS program and details regarding input and output datasets for the included SAS programs.

## COMMENT BLOCKS

The comment blocks I use within a main program are used to help read the SAS Log. Without the comment blocks the log will output a line signifying the end execution of a SAS program, which is hard to see.

```
MLOGIC(CREATEEXCELGRAPH): Ending execution.
```

Using comment blocks will help differentiate the start and end of a SAS program, which helps when reviewing the log. Note in Figure 3 that I use number signs for a comment block signifying the end of a SAS program, and @ symbols for a comment block signifying the start of a SAS program.

### Figure 3. Part of the SAS log of a main program.

```
NOTE: DATA statement used (Total process time):
      real time           0.01 seconds
      user cpu time       0.01 seconds
      system cpu time     0.00 seconds
      Memory              225k
      OS Memory           11120k
      Timestamp           12/9/2010 12:52:10 PM

1269 /******//
1270 /*                               End Part 1 Raw Data                               */
1271 /******//
1272
1273
SYMBOLGEN: Macro variable PATH resolves to D
1274
1275 /*@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@*/
1276 /*                               Start Transmission Dates                               */
1277 /*@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@*/
1278 %include "&Path:\11970.163\Products\MEWACS\MEWACS4.0\SAS Programs\M4.0 P1 02 Transmission
1278! Dates.sas";
```

```
NOTE: There were 3097 observations read from the data set M4RAWD.RAW_COMMANDS.
```

Figure 4. Main Program code example

```

/*-----*/
/*                               Start Part 1 Raw Data                               */
/*-----*/
%include "&Path:\11970.163\Products\MEWACS\MEWACS4.0\SAS Programs\M4.0 P1 01 Raw Data.sas";
/*-----*/
/*                               End Part 1 Raw Data                               */
/*-----*/

/*-----*/
/*                               Start Transmission Dates                               */
/*-----*/
%include "&Path:\11970.163\Products\MEWACS\MEWACS4.0\SAS Programs\M4.0 P1 02 Transmission Dates.sas";
/*-----*/
/*                               End Transmission Dates                               */
/*-----*/

/*-----*/
/*                               Start Compliance Memo Datasets                               */
/*-----*/
%include "&Path:\11970.163\Products\MEWACS\MEWACS4.0\SAS Programs\M4.0 P1 03 Compliance Memo Datasets.sas";
/*-----*/
/*                               End Compliance Memo Datasets                               */
/*-----*/

/*-----*/
/*                               Start Compliance Memo Graphs                               */
/*-----*/
%include "&Path:\11970.163\Products\MEWACS\MEWACS4.0\SAS Programs\M4.0 P1 04 Compliance Memo Graphs.sas";
/*-----*/
/*                               End Compliance Memo Graphs                               */
/*-----*/

|
/*-----*/
/*                               Start Compliance Memo Doc                               */
/*-----*/
Data _null_;
    If FYears = 2 then;
        %include "&Path:\11970.163\Products\MEWACS\MEWACS4.0\SAS Programs\M4.0 P1 05 Compliance Memo Doc.sas";
    Else %include "&Path:\11970.163\Products\MEWACS\MEWACS4.0\SAS Programs\M4.0 P1 05 Compliance Memo Doc CurrentFX Only.sas";

Run;
/*-----*/
/*                               End Compliance Memo Doc                               */
/*-----*/

```

## CONCLUSION

You can create a main SAS program to help you organize and run multiple SAS programs at once. The main things to think about when creating a main program are preparation, organization, and documentation. Be sure to review all of the programs you will use looking at macro variables and LIBNAME and FILENAME references. Remember to use a program header to help organize all of the inputs and outputs of the program and utilize comment blocks to keep track of the programs within the SAS Log.

## REFERENCES

SAS Documentation for 9.2.3.

## CONTACT INFORMATION

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

Lisa Mendez  
 SRA International  
 1777 NE Loop 410, Ste 510  
 San Antonio, TX 78217  
 Phone: 210-832-5213  
 Fax: 210-824-9578  
 E-mail: lisa\_mendez@sra.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.