#### PharmaSUG 2013 - Paper CC18

# Creating a Batch Command File for Executing SAS with Dynamic and Custom System Options

Gary E. Moore, Moore Computing Services, Inc., Little Rock, Arkansas

#### **ABSTRACT**

You would like to customize your Windows SAS environment, but as an employee or contractor without administrative privileges, you can't make the necessary changes to the configuration file in the !SASROOT directory. Or perhaps you would like to dynamically assign the destination of the .LOG and .LST file. How do I accomplish these with only user privileges? With a little knowledge of the Batch Command Language, you can create a batch file that will invoke SAS with the system options you need.

This paper describes how this can be accomplished and gives examples of using batch files and the SAS System Options at SAS invocation.

#### INTRODUCTION

The provided work environment may not be the most efficient for the demands of a project. If the work environment is on a server or a corporate computer, we do not usually have administrative privileges to make the necessary changes to the SAS configuration file to create an efficient SAS project environment.

Even without administration privileges an efficient SAS environment can be created using SAS System Options and a batch file for SAS invocation.

#### **DEFINITIONS AND EXAMPLES**

SAS Systems Options controls how a SAS session behaves. SAS System Options can be specified in numerous ways. But, there are only two ways to specify all of the SAS System Options: the SAS configuration file and SAS invocation at the command prompt.

If the default SAS configuration file cannot be modified, a copy can be created and modified. Using this modified copy of the SAS configuration file and the SAS System Option, config, at SAS invocation, a customized environment can be achieved.



## **Output 1. Command Prompt Invocation of SAS**

```
 \begin{tabular}{ll} C:\Program Files\SAS\SASFoundation\9.2\sas.exe" -config "c:\project1\sas.cfg" \\ \end{tabular}
```

If only a few SAS System Options need to be modified, like linesize and log destination, these can be called individually at SAS invocation.

```
C:\> "C:\Program Files\SAS\SASFoundation\9.2\sas.exe" -linesize 132 -log
"c:\project1\logs"
```

Since executing these at the command prompt can be time consuming process, a better solution is to create a batch file containing the desired SAS invocation.

Batch files are files with the .BAT or .CMD file extensions that contain Batch Command Language. The Batch Command Language is a command scripting language. There is a version of the Batch Command Language

available in all MicroSoft operating systems. It is executed by the operating system's command interpreter and therefore may contain any command available at the command prompt. But, it can also have logic and branching elements. In non-MicroSoft operating environments, it would be synonymous with IBM Job Control Language or UNIX Script Language.

The syntax for the Batch Command Language for the following examples is given in the Appendix. Once the command is saved in a batch file, it can be invoked clicking on the batch file icon.

Dynamic customization of the SAS invocation can be accomplished by using parameters within the batch file. User interaction can be achieved by requesting input from the user for a parameter value, like linesize.

```
REM batch file - user input and log destination @ECHO off
SET lnsize=
SET /p lnsize="Line Size "
"C:\Program Files\SAS\SASFoundation\9.2\sas.exe" -linesize %lnsize% -log
"c:\project1\logs"
PAUSE
```

If a SAS program file is dropped onto the batch file icon, it is processed as a parameter to the batch file. This will run whatever SAS program is dropped onto the batch file icon.

```
REM batch file - user input, log destination, program as parameter @ECHO off
SET lnsize=
SET /p lnsize="Line Size "
"C:\Program Files\SAS\SASFoundation\9.2\sas.exe" -linesize %lnsize% -log
"c:\project1\logs" -sysin %1
PAUSE
```

If two SAS program files are dropped onto the batch file icon, these can be processed as individual parameters. This will run the SAS programs one at a time and has the benefit of not tying up the CPU running the programs concurrently.

```
REM batch file - user input, log destination, 2 programs as parameters @ECHO off
SET lnsize=
SET /p lnsize="Line Size "
"C:\Program Files\SAS\SASFoundation\9.2\sas.exe" -linesize %lnsize% -log
"c:\project1\logs" -sysin %1
"C:\Program Files\SAS\SASFoundation\9.2\sas.exe" -linesize %lnsize% -log
"c:\project1\logs" -sysin %2
PAUSE
```

If multiple SAS programs are dropped onto the batch file icon this parameter list can be stepped through one at a time using the FOR DO loop assigning them into a parameter array %%A to run each SAS program one at a time. Using the name parameter option %~n1, the status and name of the program can be displayed. Note that when using the FOR DO loop parameter options, like %~n1, can be processed using the %%A parameter array and can also be addressed an array, %%~nA.

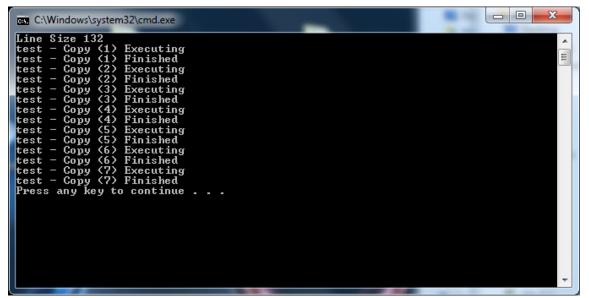
```
REM batch file - user input, log destination, multiple programs as parameters looped through @ECHO off
SET lnsize=
SET /p lnsize="Line Size "
FOR %%A IN (%*) DO (
ECHO %%~nA Executing
"C:\Program Files\SAS\SASFoundation\9.2\sas.exe" -linesize %lnsize% -log
"c:\project1\logs" -sysin %%A
ECHO %%~nA Finished
)
PAUSE
```

Additional dynamic assignment of SAS system options can be created using parameter options. In the previous example, the destination of the log files is statically defined to c:\project1\logs. But, since most projects use a similar directory structure, if we want to define the location of the log files dynamically, we can use some parameter options.

With knowledge of the project directory structure, if the program files are under a subdirectory, like c:\projectx\pgms, and the log files are in an adjacent directory, then we can define the destination of the log files dynamically using the

path parameter option %~p1 that is based on the directory path of the SAS program. Then this becomes no longer hard coded and the batch file can be used for all projects with similar directory structures.

```
REM batch file - user input, dynamic log destination, multiple programs as parameters looped through @ECHO off
SET lnsize=
SET /p lnsize="Line Size "
ECHO %~nl Executing
FOR %%A IN (%*) DO (
ECHO %%~nA Executing
"C:\Program Files\SAS\SASFoundation\9.2\sas.exe" -linesize %lnsize% -log ""%%~pA..\logs\"
-sysin %%A
ECHO %%~nA Finished
)
PAUSE
```



Output 2. Batch File processing of SAS program files

#### **CONCLUSION**

Provided with a little knowledge of the Batch Command Language and using SAS System Options, a batch file can be created that will invoke SAS with the system options you need. It can be interactive with the user. It can be dynamic with its environment.

#### **REFERENCES**

SAS 9.2 Companion for Windows - System Options under Windows

Rob van der Woude's Scripting Pages - http://www.robvanderwoude.com/

Simon Sheppard's Command line reference website - http://ss64.com/nt/

### **CONTACT INFORMATION**

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

Name: Gary E. Moore

Enterprise: Moore Computing Services, Inc.

Address: 2006 Beckenham Cove City, State ZIP: Little Rock, AR 72212

Work Phone: 501-225-8689
E-mail: gary@moorecsi.net
Web: www.moorecsi.net

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.

## **APPENDIX**

# **Syntax of Batch Command Examples**

REM	remark/comment line
@	Does not display the text on that line
ECHO	
Off	does not display text from that point on
On	displays the text from that point on
message or variable	displays the following text or variable
SET	sets an environment variable
SET /p	prompts for user input to define an environment variable
%var%	environment variable
%1	the first parameter passed to the batch file
FOR %%var IN (set) DO (command)	conditionally perform a command several times
%%var	array parameter
%~n1	the name of the first parameter passed to the batch file
%%~nVAR	the name of the array parameter passed to the batch file
%~p1	the path of the first parameter passed to the batch file
%%~pVAR	the path of the array parameter passed to the batch file
PAUSE	pause the execution of the batch file until a key is pressed