

Some New and Not so New Batch Processing Techniques

Brian Varney, COMSYS, a Manpower Company, Portage, MI

Abstract

By using a Windows batch file to submit SAS® programs, one can minimize the maintenance and maximize the repeatability of SAS program submissions. This paper intends to discuss strategies as well as some new functionality in SAS 9.2. This paper also addresses ways to structure SAS parameterized programs that will make them easier to maintain thus reducing the chance of human error.

Introduction

The purpose of this paper is to help others transform programs needing significant manual effort into more robust and repeatable programs requiring less maintenance. This will involve utilizing Windows BAT files and environment variables as well as different methods for passing parameters into SAS programs. This paper does not address scheduling. Although once a program is set up to submit using batch techniques, it will be very easy to call from a scheduler.

In SAS 9.2, some new features were introduced called “Checkpoint Mode” and “Restart Mode”. These features allow batch programs that terminate before completing to be resubmitted, resuming execution with the DATA or PROC step that was executing when the failure occurred. These two new modes will be discussed in this paper.

Batch Processing

For the purpose of this paper, batch processing is the execution of a SAS program or series of SAS programs without manual intervention.

Some advantages of using batch processing to run SAS program are:

- Creates a more repeatable process regardless of who runs the programs
- Reduces manual interventions to SAS programs
- Creates processes that are easier to schedule

If you have a file with an extension of .bat, you can just double click the filename and Windows will execute the program as a batch process.

A sample .bat file could be ...

rem a line that begins with rem is a remark a.k.a. comment.

rem echo tells Windows to replay the value of the text or variable name
echo hello

rem to refer to a Windows environment variable begin and end with a percent sign (%)
echo %username%

rem set is the method for assigning a value to a Windows environment variable.
rem this value goes away once the batch file is done processing
set value=MWSUG

echo %value%

This would result in the following output...

```
hello
bvarney
MWSUG
```

In preparing a batch environment for processing a SAS program, the bat file may look like the following...

```
rem set up pointers to the SAS executable, config, autoexec, etc.

set _saspath_=C:\Program Files\SAS92\SASFoundation\9.2
set _sasexe_="%_saspath92_%\sas.exe"
set _config_="%_saspath92_%\nls\en\SASV9.CFG"

set _base_path_=c:\_mystuff\SAS Stuff\SUGI Stuff\mwsug\2010

set _auto_="%_base_path_%\autoexec.sas"

rem *****
set _sysin_="%_base_path_%\example.sas"
set _parm_=1
set _log_=%_base_path_%\SASLogs\example%_sysparm_%.log
%_sasexe_% -config %_config_% -SYSIN %_sysin_% -autoexec %_auto_% -sysparm %_parm_% -log "%_log_%" -
ICON -NOSPLASH
```

Regardless of what ID this is run under, the SAS program will use the desired configuration and autoexec files.

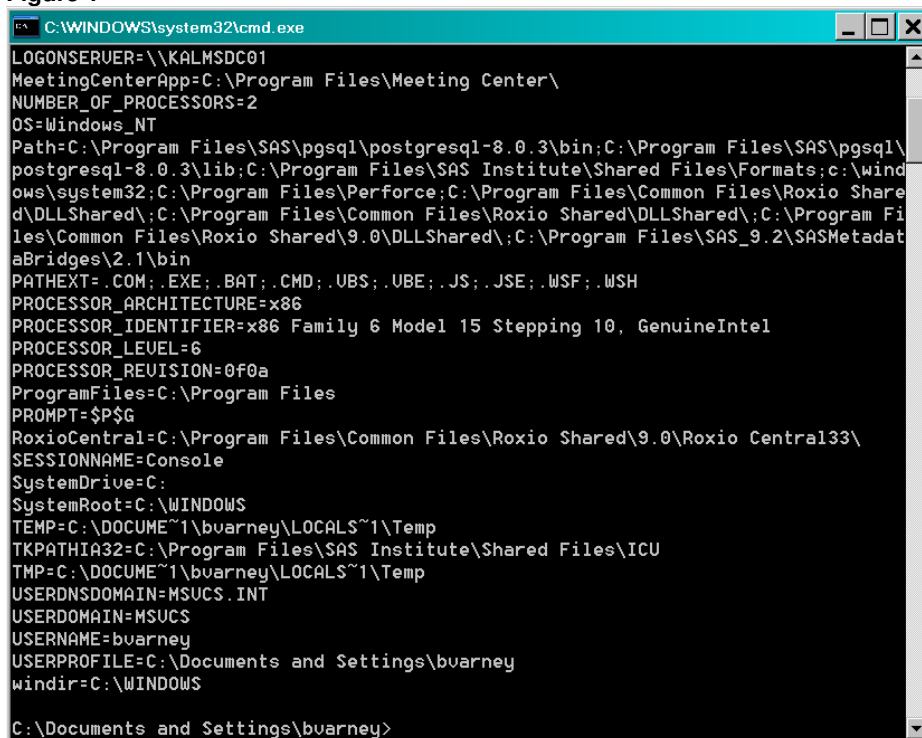
Windows Environment Variables

Windows environment variables are variables that are available to Windows processes. These can be defined on a more permanent nature or one can set them within a Windows batch file and they will go away as soon as the batch process is finished.

One way to view your current Windows environment variables and their respective values is to open up a DOS command prompt and execute the "set" command. You will see all of your current Windows environment variable value pairs scroll by.

Notice in the DOS window in Figure 1, the windows environment variable names followed by the values.

Figure 1



```
C:\WINDOWS\system32\cmd.exe
LOGONSERUER=\\KALMSDC01
MeetingCenterApp=C:\Program Files\Meeting Center\
NUMBER_OF_PROCESSORS=2
OS=Windows_NT
Path=C:\Program Files\SAS\pgsql\postgresql-8.0.3\bin;C:\Program Files\SAS\pgsql\
postgresql-8.0.3\lib;C:\Program Files\SAS Institute\Shared Files\Formats;c:\wind
ows\system32;C:\Program Files\Perforce;C:\Program Files\Common Files\Roxio Share
d\DLLShared;C:\Program Files\Common Files\Roxio Shared\DLLShared;C:\Program Fil
es\Common Files\Roxio Shared\9.0\DLLShared;C:\Program Files\SAS_9.2\SASMetadat
aBridges\2.1\bin
PATHEXT=.COM;.EXE;.BAT;.CMD;.UBS;.UBE;.JS;.JSE;.WSF;.WSH
PROCESSOR_ARCHITECTURE=x86
PROCESSOR_IDENTIFIER=x86 Family 6 Model 15 Stepping 10, GenuineIntel
PROCESSOR_LEVEL=6
PROCESSOR_REVISION=0f0a
ProgramFiles=C:\Program Files
PROMPT=$P$G
RoxioCentral=C:\Program Files\Common Files\Roxio Shared\9.0\Roxio Central133\
SESSIONNAME=Console
SystemDrive=C:
SystemRoot=C:\WINDOWS
TEMP=C:\DOCUME~1\buarney\LOCALS~1\Temp
TKPATHIA32=C:\Program Files\SAS Institute\Shared Files\ICU
TMP=C:\DOCUME~1\buarney\LOCALS~1\Temp
USERDNSDOMAIN=MSUCS.INT
USERDOMAIN=MSUCS
USERNAME=buarney
USERPROFILE=C:\Documents and Settings\buarney
windir=C:\WINDOWS

C:\Documents and Settings\buarney>
```

SAS Macro Variables Leveraging Windows Environment Variables

There are a couple of methods for leveraging Windows Environment variables from a SAS program.

The code below reads all of the current Windows environment variables using the set statement and loads them into SAS macro variables using a call symput statement.

```
filename fnset pipe "set";

data win_env_vars;
  infile fnset trunccover;
  input winenvvars_line $char500.;
  length wevar $50 weval $500;
  wevar=scan(winenvvars_line,1,'=');
  weval=scan(winenvvars_line,2,'=');
  call symput(wevar,weval);
run;
```

Another method is to use the %sysget to obtain a specific Windows environment variable.

```
%let udomain=%sysget(userdomain);
```

Although in most cases, there would be no reason to do this there is a way to write a value to a Windows environment variable from a SAS program.

```
* load values MWSUG and 2010 into the Windows environment variables rug and
rug year respectively.
options set=rug=MWSUG
        set=rugyear=2010;

* read them into SAS macro variables and display the values to the log.
%let rug=%sysget(rug);
%let rugyear=%sysget(rugyear);

%put &rug &rugyear.;
```

Checkpoint and Restart Mode (new in 9.2!)

These two options can be used together to allow batch programs that terminate before completing to be resubmitted and resume where they left off. When resubmitted, the program will resume execution with the data or proc step that was executing when the failure occurred. Data and proc steps that already completed will not be re-executed.

For example, by starting a SAS program using the following statement...

```
sas -sysin 'c:\mysas\myprogram.sas' -stepchkpt -noworkterm -noworkinit
    -errorcheck strict -errorabend
```

To resubmit a batch SAS program using the checkpoint-restart data...

```
sas -sysin 'c:\mysas\mysasprogram.sas' -stepchkpt -steprestart -noworkinit
    -noworkterm -errorcheck strict -errorabend
```

Checkpoint and Restart Option Reference

- SYSIN, if required in your operating environment, names the batch program
- STEPCHKPT continues checkpoint mode
- STEPRESTART enables restart mode, indicating to SAS to use the checkpoint-restart data
- NOWORKINIT starts SAS using the WORK library from the previous SAS session
- NOWORKTERM saves the WORK library when SAS ends
- ERRORCHECK STRICT puts SAS in syntax-check mode when an error occurs in the
- LIBNAME, FILENAME, %INCLUDE, and LOCK statements
- ERRORABEND specifies whether SAS terminates for most errors

Conclusion

By setting up a program or system of SAS programs to be submitted via Windows batch programs, it is easier to establish a controlled and repeatable process than if someone is submitting these programs in interactive SAS.

To make the SAS program or system of SAS programs even more repeatable and maintainable, modularize and parameterize your programs such that changes only need to be made in one place or one file.

References

SAS® 9.2 Language Reference: Concepts, Second Edition

Contact Information

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

Brian Varney
COMSYS, a Manpower Company
5220 Lovers Lane
Portage, MI 49002
Phone: 269-365-1755
Fax: 269-553-5101
E-mail: bvarney@comsys.com
Web: www.comsys.com/analytics

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.