Paper 090-2010

Make Your SAS® Code Environmentally Aware

Clarke Thacher, SAS Institute, Cary, NC

ABSTRACT

SAS global macro variables and operating system environment variables contain a wealth of information that can be used to add extra power to your SAS programs. This information can be used to add additional diagnostic information to your logs and add customization to reports.

We will demonstrate a simple program to access all current operating system variables. We will describe some of the most common variables and suggest how they might be used.

As an added bonus, we will show how to define SAS LIBNAMEs without writing a single line of SAS code.

INTRODUCTION

Environment variables offer SAS programmers the opportunity to customize their programs to the system environment where they are running. Environment variables are set by the operating system, command shell, user customizations, or the SAS program installation or configuration. They can be referenced in SAS programs using several methods.

WHAT ARE ENVIRONMENT VARIABLES?

Environment variables are a convenient way to pass information to programs running in the Windows, Linux, and UNIX environments. Environment variables establish an association between a name and a character string value. These environment variables can be accessed within shell scripts or in application programs such as SAS. Names and values can be arbitrarily long, although most names are less than 20 characters in length. In the UNIX and Linux environment, environment variable names are case sensitive, that is, **ThisVar** is not the same as **THISVAR** or **thisvar**.

USING ENVIRONMENT VARIABLES IN THE OPERATING ENVIRONMENT

UNIX AND LINUX

Environment variables in UNIX are usually set by the shell, either from the command line or within a shell script. Often, an environment variable will be set with a concatenation of the previous value of the variable with a new value. There are many shells available in UNIX and they often have different commands to set environment variables.

Table 1 summarizes the commands for setting environment variables for three of the most common shells. The third column shows the files where variables can be set automatically. The fourth column shows the name of the file where variables can be set per user. The last column shows the file to modify to set variables for users within a SAS process. The files are located in the “bin” directory in the SAS installation directory.

Environment variables settings are inherited from the process that launched them at process creation time. This means that changes to environment variables made to the shell process after the SAS session is initiated will not be seen by the SAS process. This also means that any changes to the environment within a SAS process will not be seen by the parent process, although the changes will be inherited by any processes that are launched by the SAS process.

Shell	Command	Set variables globally	Set variables per user	Set variables for SAS processes
Bourne Shell /bin/sh	VAR=value export VAR	/etc/profile	~/.profile	sasenv_local
k shell /bin/ksh	export VAR=value	/etc/profile	~/.profile	sasenv_local.ksh
c shell /bin/csh	setenv VAR value	/etc/csh.cshrc	~/.cshrc	sasenv_local.csh

Table 1 Setting Environment Variables in the UNIX and Linux Environments

WINDOWS

Windows environment variables can be set in various ways on Windows. The easiest way to set environment variables on Windows is to use a dialog box. The SAS process will read the current environment variables at the start of the process. It will not see changes to environment variables after the process has started, although there are ways to set and modify environment variables within a SAS session.

You can access the dialog box by following these steps:

1. Right-click **My Computer** from the start menu. Select **properties.**
2. In the System Properties window, go to the **Advanced** tab, click the **Environment Variables** button, which gives you the Environment Variables window.
3. To create a new environment variable click the **New** button in the **User variables** panel.
4. In this panel, enter the name and value for your new environment variable.

Figure 1 shows these dialog boxes.

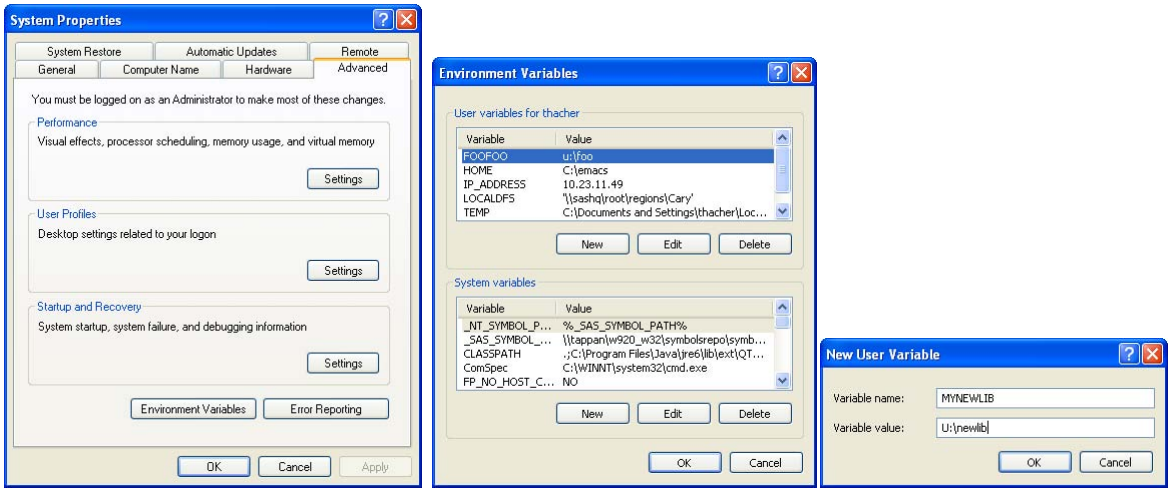

Figure 1 Dialog Boxes Used to Set Environment Variables in Windows

SETTING ENVIRONMENT VARIABLES WITH SAS

The SET system option enables you to set an environment variable within a SAS session. It is possible to add this option to the SAS configuration file to predefine environment variables for all SAS invocations. Environment variables set in a SAS session will be inherited by child processes that are launched by the SAS process.

The SAS installation process uses the SET option in the sasv9.cfg file to configure SAS. The following is an excerpt from the 9.2 Windows install configuration file that demonstrates the use of the set option.

```
/* DO NOT EDIT BELOW THIS LINE - INSTALL Application edits below this line */
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
-SET sasext0 "C:\Program Files\SAS92\SASFoundation\9.2"
-SET sasroot "C:\Program Files\SAS92\SASFoundation\9.2"
-SET sasext1 "C:\Program Files\SAS92\SASFoundation\9.2\nls"
```

It is also possible to set environment variables on the options statement:

```
options set=scratch='!TEMP';
```

USING ENVIRONMENT VARIABLES WITHIN SAS

However environment variables are set, they can easily be used within a SAS program. Environment variables can be accessed through a filename "pipe" and DATA step functions.

USING EXTERNAL COMMANDS WITH A FILENAME PIPE

You can easily look at all of your defined environment variables in the CMD window by issuing the SET command:

```
C:\Documents and Settings\thacher>set
ALLUSERSPROFILE=C:\Documents and Settings\All Users
APPDATA=C:\Documents and Settings\thacher\Application Data
...
```

The following SAS program demonstrates the use of the filename pipe statement to invoke the windows SET command to produce a data set with one observation per environment variable.

```
filename envcmd pipe 'set' lrecl=1024;
data save.xpset;
   infile envcmd dlm='=' missover;
   length name $ 32 value $ 1024;
   input name $ value $;
run;
```

The program uses the Windows SET command to return a list of the current environment variables to a filename pipe. This pipe is read by the DATA step using the equal sign (**=**) as the delimiter between the variable name and the value. We set the record length to 1024, since some variables can have very long values. A similar program can be used in the UNIX and Linux environments.

You can browse the resulting data set to discover environment variables that might be of use. Some environment variables shown through this method are available only when the cmd shell is active and will not be accessible through the SYSGET function.

USING DATASTEP FUNCTIONS

If you know the name of the environment variable that you want to reference, you can use the SYSGET function. SYSGET takes a character string as an argument and returns a character string that is the value of that environment variable. An error message will be issued if the environment variable is not defined. The ENVLEN function will return the length of the string associated with the environment variable name passed as an argument. If the environment variable is not recognized, a value of -1 will be returned. Environment variable names passed to these functions should be trimmed of trailing blanks and, in UNIX and Linux, match case.

The following DATA step program shows the use of the SYSGET and ENVLEN functions to retrieve and display a list of environment variables and their values.

```
data vars;
   length name $ 32 value $ 200;
   input name $;
   if envlen(trim(name)) > 0 then value = sysget(trim(name));
   put name= value=;
datalines;
USERNAME
USERDOMAIN
TEMP
;;;
```

The next program retrieves the value of the environment variable **MYPATH**. This value is used to name the input file that contains the raw data for the DATA step.

```
data new;
   length P $ 200;
   if _n_ = 1 then do;
      if envlen('MYPATH') < 0 then do;
         put 'ERROR: MYPATH HAS NOT BEEN DEFINED.';
      stop;
      end;
   p=sysget('MYPATH');
   infile t filevar=p;
   end;
   input X y Z;
run;
```

SYSGET can also be used as a macro function:

```
%PUT %SYSGET(USERNAME);
```

IMPLICIT REFERENCE IN PATHNAMES

It is also possible to reference environment variables through pathname references. If a pathname reference in a filename or libname begins with an exclamation point (**!**), SAS will attempt to resolve the first portion of the name as an environment variable. The environment variable in the string will be replaced by the value of that variable. This feature is used extensively in SAS configuration processing.

The following program creates a filename reference to a file in the system temporary directory defined by the TEMP environment variable. The first step writes to the file, the second reads from the file, and the third step deletes the temporary file.

```
filename tmp "!TEMP\T0001.DAT";
data _null_;
   file tmp;
   put "Hello";
run;

data _null_;
   infile tmp end=done;
   input;
   put _infile_;
run;

data _null_;
   rc = fdelete("tmp");
   if (rc = 0) put "tmp file deleted."
run;
```

This program uses the SASCFG environment variable on Windows to write the SAS configuration file in the installation directory:

```
data _null_;
   infile "!SASCFG\sasv9.cfg";
   input;
   put _infile_;
run;
```

PRE-DEFINED ENVIRONMENT VARIABLES

Windows and UNIX environments provide many environment variables that can be queried for valuable information. The following tables summarize some of the variables that can be available. There are many variations between different operating system versions, so you should check the OS documentation. The first table shows some environment variables that can be available in the UNIX environment.

Name	Description
path	specifies the directories to search for commands.
ld_library_path shlib_path libpath	specifies the directories to search for shared run-time libraries. Consult your operating system documentation.
lang	specifies the locale that is used by the shell.
logname	specifies the user login name.
tz	specifies the time zone that is used by the shell.
pwd	specifies the current working directory.
home	specifies the user home directory.
shell	specifies the shell command. This command determines how the SAS X command processes system commands.
_ (underscore)	specifies the command line that is used to invoke the session.

Table 2 Some Environment Variables in the UNIX and Linux Environments

Once again, remember that UNIX environment variables are case sensitive, so that **HOME** is not the same as **home**.

Windows also provides many environment variables that might be useful to you. The table below shows some of these variables. Once again, this list can change with different versions of Windows.

Name	Description
ALLUSERSPROFILE	returns the location of the All Users Profile.
APPDATA	specifies the location where applications store data by default.
CD	returns the current directory string.
COMPUTERNAME	returns the name of the computer.
HOMEDRIVE	specifies the drive letter that is connected to the user's home directory.
HOMEPATH	specifies the full path of the user's home directory.
NUMBER_OF_PROCESSORS	specifies the number of processors that are installed on the computer.
OS	specifies the operating system name.
PATH	specifies the search path for executable files.
PROCESSOR_ARCHITECTURE	specifies the chip architecture of the processor.
PROCESSOR_IDENTIFIER	is a description of the processor.
PROCESSOR_LEVEL	specifies the model number of the processor that is installed on the computer.
PROCESSOR_REVISION	specifies the revision number of the processor.
TEMP and TMP	specifies the default temporary directories. Some applications require TEMP and others require TMP.
USERNAME	specifies the name of the user who is currently logged on.
USERPROFILE	specifies the location of the profile for the current user.
WINDIR	specifies the location of the operating system directory.

Table 3 Some Windows Environment Variables

In addition to the operating system provided variables, each SAS session defines some useful environment variables. These are listed in the table below.

Name	UNIX	Windows	Description
SASV9_OPTIONS	•		specifies the options to use for SAS sessions.
SASV9_CONFIG	•		specifies an alternate location of configuration file.
TKPATH	•	•	specifies a search path for SAS shared executable libraries.
SASROOT	•	•	specifies the SAS install directory.
SAMPSRC	•	•	specifies the location of sample source files.
SAS_EXECFILENAME		•	specifies the filename of the current SAS source file, if it is submitted from the enhanced editor.
SAS_EXECFILEPATH		•	specifies the pathname of the current SAS source file, if it is submitted from the enhanced editor.
MYSASFILES		•	specifies the default location for finding and saving SAS source files.
SAS_SYS_CONFIG		•	if defined, specifies the system configuration file to use for all users on the system.
SAS_USER_CONFIG		•	if defined, specifies a user-specific configuration file.
SAS_OPTIONS		•	specifies the options to use for SAS sessions.

Table 4 SAS Environment Variables

CONCLUSION

We have shown how to set environment variables through operating system commands and SAS options. We have also shown three different ways to access the information contained in these variables. Using you own environment variable definitions combined with environment variables provided by your operating system provides you with a wealth of information that can be combined with the powerful SAS language features. You will be able to create powerful and robust programs that can adapt to changes without the chore of making changes to your code. Environment variables enable you to save time making changes to your existing programs.

RECOMMENDED READING

More information about using environment variables with SAS can be found in **SAS(R) 9.2 Companion for UNIX Environments** and **SAS(R) 9.2 Companion for Windows.** Specific information about the ENVLEN and SYSGET functions can be found in **SAS Language Reference: Dictionary**.

SAS Institute, Inc. 2009. *SAS 9.2 Language Reference: Dictionary*, Second Edition. Cary, NC: SAS Institute, Inc.

(for specific information about the ENVLEN and SYSGET functions)

SAS Institute, Inc. 2009. *SAS 9.2 Companion for UNIX Environments.* Cary, NC: SAS Institute, Inc.

SAS Institute, Inc. 2009. *SAS 9.2 Companion for Windows*. Cary, NC: SAS Institute, Inc.

CONTACT INFORMATION

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

Name: Clarke Thacher
Enterprise: SAS Institute
Address: 500 SAS Campus Drive

City, State ZIP: Cary, NC 27513
Work Phone: 919-531-7786
Fax: 919-677-4444
E-mail: Clarke.Thacher@sas.com
Web: http://www.sas.com