You Want that Program to Run on the PC in New Jersey and the Unix in Basel, without Modification??

David Franklin, TheProgrammersCabin.com, Litchfield, NH

ABSTRACT

With offices in many parts of the world, SAS® programmers are being asked more and more to write SAS programs in one environment to run on many others, including different versions of SAS and different operating systems. The underlying architecture of SAS allows SAS programmers to do this easily, with a little bit of planning. This paper introduces a few techniques and uses a number of SAS environment variables, including SYSSCPL, SYSSITE and SYSVER, to make that program run in multiple environments with no modification.

INTRODUCTION

"That program you wrote for New York last week, can you make it run on the system in Basel? And by the way, New Jersey were impressed with it and wondered if you could adapt it to go on their system -- remember they are still on 8.2."

I received a similar request from a client not too long ago -- the locations and operating systems are different but the situation was the same.

New York was running SAS version 9.1.3 on Windows XP, Basel was running the same version but on Unix (on an AIX64 box), and New Jersey were running SAS version 8.2 on Windows XP with a different directory structure. To add to the mix, if the operating system was Unix based then any sorting of data was to use the host sort when the size of the dataset was greater than 50M.

It would have been easy to take the original version made for New York and adapt it for each site, ending up with three different versions. But what if we could just adapt the one version for use on all three sites?

OUR INITIAL PROGRAM

First, let's look at our initial program.

```
     ** Bring in the data. The record for Colorado is duplicated for
        instructional purposes. Source: USGS 87-242, May 1990;
   data River_US (label='Rivers in US over 1000 miles in Length');
      attrib name length=$40 label='River Name'
             distance length=8 label='Distance (miles)';
      length name $40 distance 8;
      input Name $ & distance @@;
   cards;
   Mississippi  2340  Columbia  1240  Colorado  1450
   St. Lawrence  1900  Colorado  1450  Missouri  2540
   Ohio  1310  Rio Grande  1900  Brazos  1280
   Snake  1040  Atchafalaya  1420  Yukon  1980
   Red  1290  Arkansas  1460
   ;
   run;

   libname derived "E:\TAUI\DERIVED";

   ** Remove Duplicates and put in dataset _DUPRECS;
   proc sort data=River_US out=derived.River_US
             noduprec dupout=_duprecs;
      by descending distance;
   run;
```

```
** Output Listings;
title1 "Duplicate Records Found and Removed in Load of RIVER_US Data";
proc print data=_DUPRECS label;
   var name distance;
   format distance comma8.;
run;

title1 "Longest Rivers in US Over 1,000 miles, By Distance";
proc print data=derived.River_US label;
   var name distance;
   format distance comma8.;
run;
```

As can seen, the program will load the data using the CARDS statement, find and output any duplicates found using the SORT procedure, then report any duplicates found and finally the listing. This version of the program will one on one version and one operating system and my not work on other SAS versions or operating environments.

How what if we could adapt this program to work on multiple SAS versions and multiple operating systems? The LIBNAME statement is written for Windows operating systems and the DUPOUT option was not available in the SORT procedure until SAS version 9.1.

KNOWING WHAT SAS KNOWS ABOUT YOUR OPERATING ENVIRONMENT

Before we can consider writing the SAS code to another environment we have to find out what the operating system is and the SAS version being run in that environment. Fortunately SAS can help us.

SAS already knows quite a bit about your operating environment during SAS initialization. Some of this information can be located in the SAS View VMACRO that can be found in the SAS Help library. These macro variables are automatically set up during SAS initialization and cannot be changed by the users. For the purposes of this paper the four macro variables will be looked at are:

```
SYSSCP
SYSSCPL
SYSSITE
SYSVER
```

The SYSSCP and SYSSCPL are linked and contain an identifier for your operating environment. It is useful to use these variables to execute applicable system commands. For our example, we are going to run this under Windows and Unix so the relevant system macro variables are:

```
Platform          SYSSCP Value    SYSSCPL Value
----------------------------------------------
R64 or AIX64 or    AIX 64           AIX
  AIX on POWER
Windows XP Pro       WIN          XP_PRO
```

Refer to the documentation listed in the Reference section below for a more detailed list of SYSSCP and SYSSCPL values based on SAS version.

SYSSITE is the value that SAS has assigned to your site license. This number is displayed in the SAS Log. Every site license number is unique.

Finally, SYSVER. This is the SAS version of the SAS software that you are running on your site. With this information it is possible to use features that are available to a site that is running a later version that is not available in a previous version.

ADAPTING OUR PREVIOUS CODE TO RUN ON DIFFERENT ENVIRONMENTS

Examining our request, and refining it after asking some questions, we know that New York (where our program was originally run on) is running SAS version 9.1.3 using Windows XP Pro with license number 9999US, Basel is also running SAS version 9.1.3 using Unix on an AIX box with license number 9999EU, and New Jersey is running SAS version 8.2 using Windows XP Pro with the New York license.

The first task to do is identify what code is operating system or SAS version dependent. In the above example there are two sections.

First is the LIBNAME statement. This statement is dependent on operating system and on SAS site requirements -- directory structure is dependent on SAS site. In our example New York has the directory "E:\TAUI\DERIVED" while Basel has the structure "//CharingCross/Production/Projects/Taui/Derived". New Jersey uses the same directory as New York.

With this in mind the code for the LIBNAME statement could be adapted to:

```
%if &syssite=9999US %then %do;
   libname derived "E:\TAUI\DERIVED";
%end;
%else %if &syssite=9999EU %then %do;
   libname derived "//CharingCross/Production/Projects/Taui/Derived";
%end;
%else %do;
   %put %str(ERR) OR: SYSSITE &syssite is not recognized.;
   %put        Consult your site administrator for assistance.;
   %put        Program will now terminate.;
   endsas;
   run;
%end;
```

In the code above, if the SYSSITE value is not recognized then the program will put a message to the SAS LOG and terminate. This makes the code very site specific.

The second piece of code that will not work right is the DUPOUT option in the SORT procedure -- this was not available until SAS version 9.1 but New Jersey is running SAS version 8.2. The SYSVER option is useful for this situation, as the following code will show:

```
**   Remove Duplicates and put in dataset _DUPRECS;

%if (&sysver >= 9.1) %then %do;
   ** If SAS version 9.1 or higher;
   proc sort data=River_US out=derived.River_US
         noduprec dupout=_duprecs;
     by descending distance;
   run;
%end;

%else %do;
   ** If SAS version before 9.1;
   proc sort data=River_US;
     by descending distance name;
   run;
   data derived.River_US
       _duprecs;
     by descending distance name;
     if first.name then output derived.River_US;
     else output _duprecs;
   run;
%end;
```

To finish off the code we have to cater for the case where if the operating system was Unix based then any sorting of data was to use the host sort when the size of the dataset was greater than 50M. This can be done using the SYSSCPL global macro variable as the following code demonstrates:

```
%if &sysscpl=AIX %then %do;
   options sortpgm=host sortcut=50M;
%end;
```

In the example we used the SYSSCPL value but we could have used the SYSSCP as well -- the SYSSCP value is a superset of the SYSSCPL value (refer to).

After all the adoptions to the code noted above, it is possible to now copy the program (code is in Appendix) onto the systems in New York, Basel and New Jersey without adaptation.

CONCLUSION

A brief introduction to four SAS automatic variables then resulted in the original program created for New York being able to be adapted for use on all three sites without having three separate programs being written. It is also possible to use these same techniques shown here when using two or more operating environments and/or SAS versions within the one site, or when upgrading from one SAS version to another.

REFERENCES

Largest Rivers in the United States, Water Fact Sheet, U.S. Geological Survey, Department of the Interior, OFR 87-242, May 1990

SAS Macro Language: Reference, Version 8.2, SYSSCP and SYSSCPL

Usage Note 11150: Table of SAS 9.1 values for SYSSCP and SYSSCPL based on operating system

Usage Note 36549: SYSSCP and SYSSCPL Values for Platforms Running SAS 9.2 or Later

CONTACT INFORMATION

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

David Franklin
TheProgrammersCabin.com
16 Roberts Road
Litchfield, NH 03052
Cell: 603-275-6809
Email: dfranklin@theprogrammerscabin.com
Web: http://www.TheProgrammersCabin.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

APPENDIX: CODE AFTER ADAPTION FOR USE ON MULTIPLE ENVIRONMENTS

```
** Bring in the data. The record for Colorado is duplicated for
   instructional purposes. Source: USGS 87-242, May 1990;
data River_US (label='Rivers in US over 1000 miles in Length');
   attrib name length=$40 label='River Name'
          distance length=8 label='Distance (miles)';
   length name $40 distance 8;
   input Name $ & distance @@;
cards;
Mississippi  2340  Columbia  1240  Colorado  1450
St. Lawrence  1900  Colorado  1450  Missouri  2540
Ohio  1310  Rio Grande  1900  Brazos  1280
Snake  1040  Atchafalaya  1420  Yukon  1980
Red  1290  Arkansas  1460
;
run;


%macro runlist;

   %if &syssite=9999US %then %do;
      libname derived "E:\TAUI\DERIVED";
   %end;
   %else %if &syssite=9999EU %then %do;
      libname derived "//CharingCross/Production/Projects/Taui/Derived";
   %end;
   %else %do;
      %put %str(ERR) OR: SYSSITE &syssite is not recognized.;
      %put       Consult your site administrator for assistance.;
      %put        Program will now terminate.;
      endsas;
      run;
   %end;

   %if &sysscpl=AIX %then %do;
      options sortpgm=host sortcut=50M;
   %end;

    ** Remove Duplicates and put in dataset _DUPRECS;
   %if (&sysver >= 9.1) %then %do;
      ** If SAS version 9.1 or higher;
      proc sort data=River_US out=derived.River_US
              noduprec dupout=_duprecs;
         by descending distance;
      run;
   %end;

   %else %do;
      ** If SAS version before 9.1;
      proc sort data=River_US;
         by descending distance name;
      run;
      data derived.River_US
           _duprecs;
         by descending distance name;
         if first.name then output derived.River_US;
         else output _duprecs;
      run;
   %end;


    ** Output Listings;
```

5

```
    title1 "Duplicate Records Found and Removed in Load of RIVER_US Data";
    proc print data=_DUPRECS label;
        var name distance;
        format distance comma8.;
    run;

    title1 "Longest Rivers in US Over 1,000 miles, By Distance";
    proc print data=derived.River_US label;
        var name distance;
        format distance comma8.;
    run;

%mend runlist;
%runlist;
run;
```