CHAPTER 23 Macro Facility

SAS Macro Facility under Windows

In general, the SAS macro language is portable across operating environments. This section discusses those components of the macro facility that have system dependencies. For more information, see the SAS online Help for the macro facility or *SAS Macro Language: Reference* and *SAS Language Reference: Dictionary*.

Note: The words CON, NUL, PRN, COM1 - COM9, and LPT1 - LPT9 are reserved words under Windows. Do not use these reserved words as the name of a macro variable. △

Automatic Macro Variables

The following automatic macro variables have values that are specific to Windows:

SYSCC
: contains the current SAS condition code that SAS returns to Windows when the SAS System exits. Upon exit, SAS translates this condition code to a return code that has a meaningful value for the operating system.

SYSDEVIC
: gives the name of the current graphics device. The current graphics device is determined by the DEVICE system option. Contact your SAS Support Consultant to determine which graphics devices are available at your site. For information about the DEVICE system option, see "DEVICE" on page 420 and *SAS Language Reference: Dictionary*.

SYSENV
: always contains the value `FORE` under Windows.

SYSJOBID
: returns a number that uniquely identifies the SAS task under Windows.

Operating Environment Information for Windows 95 Users: When using SYSJOBID under Windows 95, you may receive an overflow error message if SYSJOBID is used in a conditional statement. △

SYSMAXLONG
returns the maximum long integer value allowed under Windows, which is 2,147,483,647.

SYSRC
holds the Windows status of Windows commands performed during your SAS session. The variable holds a character string that is the text form of the decimal value of the Windows command status.

Note: This macro variable is useful only with Windows NT. Windows 95 does not allow the SAS System to retrieve the status of the command, and always returns 0. △

For example, consider the following statements:

```
options noxwait;
x 'dirf'; /* Invalid Windows command */
%put This Windows status is &sysrc;
x 'dir'; /* Valid Windows command  */
%put The corrected Windows status is &sysrc;
```

The following lines are written to the SAS log:

```
This Windows status is 1
The corrected Windows status is 0
```

The OPTIONS statement turns the XWAIT option off so that the Windows command prompt window disappears automatically without your having to type EXIT to return to your SAS session. If you run this example with the XWAIT option on, it does not work because you get a value of 0 in both cases; 0 is the return code for the EXIT command. If NOXSYNC is on, the value of SYSRC is automatically 0.

SYSSCP
returns the operating system abbreviation WIN.

SYSSCPL
returns the name of the specific Windows environment you are using. For Version 8 under Windows, the possible return values are

WIN_95
Windows 95

Win_98
Windows 98

WIN_NT
Windows NT Workstation

WIN_NTSV
Windows NT Server or Windows NT Advanced Server

Macro Statements

The following macro statements have behavior specific to Windows:

%KEYDEF

enables you to define function keys. It is analogous to the KEYDEF display manager command. The %KEYDEF statement has the following syntax:

%KEYDEF *key-name* | '*key-name*' <'*definition*'>;

If the *definition* argument is omitted, a message is printed to the log showing the current definition of the key; otherwise, the key's definition is changed to whatever you specify.

Key names might vary depending on your workstation. You can define any key listed in the KEYS window, provided it is not reserved by Windows. You must enclose in quotes any key name that is hyphenated or contains a space, such as SHF F2 or CTRL Z.

You can also use the %KEYDEF statement to define commands that are executed when the mouse buttons are pressed. The following key sequences are accepted as the *key-name* argument:

Key Name	**Key Sequence**
RMB	right mouse button
SHF RMB	SHIFT key, right mouse button
CTL RMB	CONTROL key, right mouse button
MMB	middle mouse button
SHF MMB	SHIFT key, middle mouse button
CTL RMB	CONTROL key, right mouse button

For example, to assign the ZOOM command to the CTL RMB key sequence, submit the following command:

```
%keydef 'ctl rmb' 'zoom';
```

%SYSEXEC

executes operating system commands immediately and places the return code in the SYSRC automatic macro variable. The %SYSEXEC statement is similar to the X statement described in "Running DOS or Windows Commands from within SAS" on page 20. You can use the %SYSEXEC statement inside a macro or in open code. The %SYSEXEC statement has the following syntax:

%SYSEXEC <*command*>;

The *command* argument can be any operating system command or any sequence of macro operations that generate an operating system command. You can also use the *command* argument to invoke a Windows application such as Notepad.

Omitting the *command* argument launches a command prompt subprocess, which is interactive. To return to your SAS session, type EXIT at the command prompt and press ENTER. The SYSRC automatic variable is set to 0 if you omit the *command* argument in the %SYSEXEC statement.

Here is a simple example of %SYSEXEC:

```
%sysexec time;
```

This statement launches a command prompt session that displays the following lines:

```
The current time is: 16:32:45.16
Enter new time:
```

Note: The %SYSEXEC statement uses the XSYNC and XWAIT system option values just like the X statement and X command do. For more information about these system options, see "XSYNC System Option" on page 22 and "XWAIT System Option" on page 22. △

Macro Functions

The behavior of the %SYSGET macro function is specific to Windows:

%SYSGET
returns the character string that is the value of the Windows environment variable passed as the argument. Both Windows and SAS environment variables can be translated using the %SYSGET function. A warning message is printed if the environment variable does not exist. The %SYSGET function has the following syntax:

%SYSGET(*environment-variable-name*);

Here is an example of using the %SYSGET function:

```
%let var1=%sysget(comspec);
%put The COMSPEC environment variable
     is &var1;
```

The following line is written to the SAS log:

```
The COMSPEC environment variable is
C:\winnt\system\command.exe
```

Autocall Libraries

This section discusses the system dependencies of using autocall libraries. For general information, see the *SAS Macro Language: Reference*.

An autocall library contains files that define SAS macros. SAS Institute supplies some autocall macros. To use the autocall facility, you must have the SAS system option MAUTOSOURCE set. When the SAS System is installed, the SASAUTOS system option is used in the SAS configuration file to tell the SAS System where to find the default macros supplied by the Institute. You can also define your own autocall macros and store them in a Windows directory.

If you store autocall macros in a Windows directory, the file extension must be .SAS. Each macro file in the directory must contain a macro definition with a macro name the same as the filename. For example, a file named PRTDATA.SAS stored in a directory must define a macro named PRTDATA.

SASAUTOS System Option

To use your own autocall macros in your SAS programs, you must tell the SAS System where to find them using the SASAUTOS system option. The syntax of the SASAUTOS option is given in "SASAUTOS" on page 456.

You can set the SASAUTOS system option when you start the SAS System, or you can use it in an OPTIONS statement during your SAS session. You should edit your SAS configuration file to add your autocall library to the library concatenation supplied by SAS Institute, as in the following example:

```
-sasautos (c:\mymacros
          !sasroot\core\sasmacro
          !sasroot\base\sasmacro
          !sasroot\stat\sasmacro
          more library specifications
          )
```

Autocall libraries are searched in the order you specify them. So if you use the above SASAUTOS option setting and call a macro named PRTDATA, the directory C:\MYMACROS is searched for the macro; then each of the !SASROOT libraries is searched.

COPYSAS Autocall Macro

Parses the log created by invoking the SAS System with the RTRACE and RTRACELOC log system options that are specified, and creates a copy script

Syntax

%COPYSAS(*copydir, rtracelog, cpcmd, scriptloc, mkcmd*)

copydir
specifies the destination directory to use as the SAS root directory for the copied files. You can specify *copydir* as a local directory (for example, **`c:\mysas`**) or as a network path (for example **`\\server\share\pubsas`**). You must specify the *copydir*.

rtracelog
specifies the location of the log file created by using the RTRACE and RTRACELOC system options. Because you use the RTRACELOC system option to specify where to store the log, you should specify the same value for *rtracelog* as you did for the system option. You must specify the *rtracelog*.

cpcmd
specifies the DOS copy command to use. If you omit this argument, the default command is **`copy /v`**. If the command you specify requires options, you can specify them in the usual way; that is, separate them from the command with a forward slash (/) or a hyphen (-).

scriptloc
specifies the path and name of the file to which to write the copy script. If you do not specify this argument, the default is COPYSAS.BAT and is placed in the SASUSER subdirectory.

mkcmd
specifies the DOS make directory command to use. If you omit this argument, the default command is **`mkdir`**. If the command you specify requires options, you can specify them in the usual way; that is, separate them from the command with a forward slash (/) or a hyphen (-).

Note that you specify values for all arguments *without* quotes.

Details

The COPYSAS autocall macro parses a log that you create by running the SAS System with the RTRACE and RTRACELOC system options specified. The log contains a record of every file that was used during the SAS session. As it parses the log, the COPYSAS macro builds a copy script (as a DOS batch file) to facilitate copying these files to another destination. This allows you to create a scaled-down copy of the SAS System, optimized for a particular use, such as running a SAS/AF application.

Note: The COPYSAS macro itself does not create a scaled-down copy of the SAS System; it only creates a DOS batch file that allows *you* to create the scaled-down copy. The batch file contains the commands to create destination directories and copy files to the destination you specify. △

CAUTION:

Thoroughly test any scaled-down configuration of the SAS System that you create. To ensure that all of the necessary file resources are recorded, you must traverse *every* path in your SAS session that an end user of the scaled-down configuration should be able to traverse. (If you are running a SAS/AF application, you should exercise every possible path, including error conditions.) Otherwise, you might fail to include a necessary file in your run-time version of the SAS System. If you do not include all of the necessary files in your scaled-down copy of the SAS System, you might get unexpected results from your SAS application. △

If you want to omit the optional arguments and use their default values, you can do so; but you must specify a comma separator as a placeholder for the argument. For example, if you want to use the default *copycmd* and *scriptloc* but use **md** to create directories, your call would look like

```
%copysas(d:\sasjr, c:\sas\filelist.log,,,md)
```

You do not have to provide the commas if no arguments follow the last argument you provided. For example, the following invocation is valid:

```
%copysas(d:\sasjr, c:\sas\filelist.log)
```

Note: Keep in mind that the configuration files that you copy (such as any SAS configuration files and the AUTOEXEC.SAS file) are probably customized for your installation and might not be usable on the scaled-down copy without some modifications. Check to make sure that the options in these files reflect the configuration that you want to convey to another user. △

For a complete description of how to create a scaled-down copy of the SAS System using the RTRACE and RTRACELOC system options in conjunction with the COPYSAS macro, see "Creating a Scaled-Down Version of the SAS System for Distribution" on page 263.

Using %COPYSAS to Create a Scaled-Down Copy of the SAS System

Suppose that in a previous SAS session, you used the RTRACE and RTRACELOC system options to create a log of files used during the session. For example, you might have invoked the SAS System with the options

```
-rtrace all -rtraceloc
            c:\sas\sasuser\filelist.log
```

In a subsequent SAS session (in which you do not specify these options), you can use the COPYSAS macro to create a copy script COPYSAS.BAT. For example, if you submit the following statement:

```
%copysas(c:\mysas,c:\sas\sasuser\filelist.log)
```

the COPYSAS macro parses C:\SAS\SASUSER\FILELIST.LOG and creates the DOS batch file C:\SAS\SASUSER\COPYSAS.BAT.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS Companion for the Microsoft Windows Environment, Version 8,* Cary, NC: SAS Institute Inc., 1999. pp.555.

SAS Companion for the Microsoft Windows Environment, Version 8

Copyright © 1999 by SAS Institute Inc., Cary, NC, USA.

ISBN 1–58025–524–8

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, September 1999