CHAPTER 20

Statements

SAS Statements under Windows

A SAS statement is a directive to the SAS System that either requests that SAS perform a certain operation or provides information to the system that might be necessary for later operations.

All SAS statements are described in *SAS Language Reference: Dictionary*. Those that are described here have syntax and usage that are specific to Windows.

ABORT

Stops executing the current DATA step, SAS job, or SAS session

Valid: in a DATA step

Windows specifics: Action of the ABEND and RETURN options; maximum value of *condition-code*

Syntax

ABORT <ABEND | RETURN> <*n*>;

ABEND
causes abnormal termination of the current SAS job or session for the current process. Further action is based on how your operating environment and site treat jobs that end abnormally.

RETURN
causes the immediate normal termination of the current SAS job or session. A condition code is returned indicating an error if a job ends abnormally.

n
specifies the condition code that is returned to the operating system. This value can range from 0 to 65535.

Details

The ABORT statement causes the SAS System to stop processing the current DATA step.

The ABEND and RETURN options both terminate the SAS process, job, or session. If *n* is not specified, the Windows batch variable ERRORLEVEL is set.

See Also

- ABORT statement in *SAS Language Reference: Dictionary*
- "Return Codes and Completion Status" on page 503

ATTRIB

Associates a format, informat, label, and length with one or more variables

Valid: in a DATA step
Windows specifics: length specification

Syntax

ATTRIB *variable-list-1 attribute-list-1…<variable-list-n attribute-list-n>*;

Note: Following is a simplified explanation of the ATTRIB statement syntax. For the complete syntax and its explanation, see the ATTRIB statement in *SAS Language Reference: Dictionary*. △

attribute-list

LENGTH=<$>*length*
specifies the length of the variables in *variable-list*. Under Windows, the length you can specify for a numeric variable ranges from 3 to 8 bytes.

Details

Using the ATTRIB statement in the DATA step permanently associates attributes with variables by changing the descriptor information of the SAS data set that contains the variables.

See Also

- ATTRIB statement in *SAS Language Reference: Dictionary*

FILE

Specifies the current output file for PUT statements

Valid: in a DATA step

Windows specifics: Valid values for *file specification*; valid options for *host-option-list*

Syntax

FILE *file-specification* <*option-list*> <*host-option-list*>;

file-specification
can be any of the file specification forms discussed in "Referencing External Files" on page 106.

Note: The words CON, NUL, PRN, LPT1 - LPT9, and COM1 - COM9 are reserved words under Windows. Do not use them as file names. △

host-option-list
names external I/O statement options specific to the Windows operating environment. They can be any of the following:

BLKSIZE=*block-size*
BLK=*block-size*
specifies the number of bytes that are physically read or written in an I/O operation. The default is 8K. The maximum is 1M.

BLOCK | NOBLOCK
is used only in the context of named pipes. This option indicates whether the client is to wait if no data are currently available. BLOCK is the default value.

BYTE | MESSAGE
is used only in the context of named pipes. This option indicates the type of pipe. BYTE is the default value.

COMMAND
is used only in the context of Dynamic Data Exchange (DDE). This option enables you to issue a remote command for applications that do not use the SYSTEM topic name. For more information, see to "Referencing the DDE External File" on page 218 and "Controlling Another Application Using DDE" on page 219.

EOFCONNECT
is used only in the context of named pipes and is valid only when defining the server. This option indicates that if an end-of-file (EOF) character is received from a client, the server should try to connect to the next client.

HOTLINK
is used only in the context of Dynamic Data Exchange (DDE). For a complete description and an example of using this option, see "Using the DDE HOTLINK" on page 223.

LRECL=*record-length*
specifies the record length (in bytes). Under Windows, the default is 256. The value of *record-length* can range from 1 to 1,048,576 (1 megabyte).

MOD
specifies that output should be appended to an existing file.

NOTAB
is used only in the context of Dynamic Data Exchange (DDE). This option enables you to use non-tab character delimiters between variables. For more information about this option, see "Using the NOTAB Option with DDE" on page 222.

RECFM=*record-format*
controls the record format. The following values are valid under Windows:

F	indicates fixed format.
N	indicates binary format and causes the file to be treated as a byte stream.
P	indicates print format.
S370V	indicates the variable S370 record format (V).
S370VB	indicates the variable block S370 record format (VB).
S370VBS	indicates the variable block with spanned records S370 record format (VBS).
V | D	indicates variable format. This is the default.

The S370 values are valid with OS/390-style files only—that is, files that are binary, have variable-length records, and are in EBCDIC format. If you want to use a fixed-format OS/390 (formerly known as MVS) file, first copy it to a variable-length, binary OS/390 file.

RETRY=*seconds*
is used only in the context of named pipes. This option specifies how long a named pipe client should wait for a busy pipe. The minimum (and default) value for *seconds* is 10.

SERVER | CLIENT
is used only in the context of named pipes. This option specifies the mode of a named pipe. The default value is SERVER.

Details

The FILE statement routes the output from the PUT statement to either the same external file to which procedure output is written or to a different external file.

See Also

- FILE statement in *SAS Language Reference: Dictionary* for portable options available
- "Named Pipe Examples" on page 232for examples of using some of these options
- "DDE Examples" on page 220for examples of using some of these options

FILENAME

Associates a SAS fileref with an external file or a logical file device

Valid: anywhere in a SAS program

Windows specifics: Valid values for *access-method*; valid values for *device-type*; valid filenames for *external-file*; valid options in *host-option-list*

Syntax

FILENAME *fileref* <*device-type*> '*external-file*' <*host-option-list*>;

FILENAME *fileref device-type* <'*external-file*'> <*host-option-list*>;

FILENAME *fileref* <*device-type*> ('*directory-1*'<,...*directory-n*'>) <*host-option-list*>;

Note: This is a simplified version of the FILENAME statement syntax. For the complete syntax and its explanation, see the FILENAME statement in *SAS Language Reference: Dictionary*. △

fileref
is any valid fileref, as discussed in "Using a Fileref" on page 107.

For examples of using filerefs in member name syntax (also called aggregate syntax), see "Assigning a Fileref to a Directory" on page 109. For the rules the SAS System uses when accessing files through filerefs, see "Understanding How Concatenated Directories Are Accessed" on page 114.

Note: The words CON, NUL, PRN, LPT1 - LPT9, and COM1 - COM9 are reserved words under Windows. Do not use them as filerefs. △

device-type
enables you to read and write data from devices rather than files. The following values are valid:

CATALOG
reads a SAS catalog as an external flat file.

COMMPORT
reads data from and writes data to a communications port.

DDE
reads data from and writes data to another application using Dynamic Data Exchange. For more information, see "DDE Syntax within SAS" on page 217.

DISK
reads data from and writes data to a disk file. Under Windows, DISK is the default for *device-type*.

DRIVEMAP
displays information about the available hard drives (local and networked).

DUMMY
specifies a null output device. This value is especially useful in testing situations.

EMAIL
lets you send electronic mail programmatically from the SAS System. For more information, see "Sending Electronic Mail from within the SAS System" on page 161.

FTP
lets you access information on other machines using TCP/IP. You must have TCP/IP software and a WINSOCK.DLL installed on your local machine. You must also be able to connect to a machine that can function as an FTP server. For more

information on the using the FTP access method, see the FILENAME statement in *SAS Language Reference: Dictionary*.

NAMEPIPE
writes data to a named pipe. For more information, see "Using Named Pipes" on page 230.

PIPE
writes data to an unnamed pipe. For more information, see "Using Unnamed Pipes" on page 228.

PLOTTER
indicates that you are accessing a plotter. The Windows Print Manager is not used. This device-type keyword is used solely in conjunction with SAS/GRAPH software.

PRINTER
indicates that you are accessing a printer file or device. By default, output is routed through the Windows Print Manager when you use this device-type keyword. For more information on altering your default printer, see the system option "SYSPRINT" on page 468.

SOCKET
lets you read and write information over a TCP/IP socket. You must have TCP/IP software and a WINSOCK.DLL installed on your local machine. The SOCKET access method uses the nonblocking method of issuing socket requests. For more information on the using the SOCKET access method, see the FILENAME statement and FILENAME function in *SAS Language Reference: Dictionary*.

TEMP
creates a temporary file that exists only as long as the filename is assigned. The temporary file can only be accessed through the logical name and is only available while the logical name exists. A physical path name is never shown to the user. If a physical path name is specified, an error is returned. Files manipulated by the TEMP device can have the same attributes and behave identically to DISK files.

For an example of specifying a device type in the FILENAME statement, see "Advanced External I/O Techniques" on page 120.

Note: The TAPE and TERMINAL device-type keywords (documented in *SAS Language Reference: Dictionary*) are not applicable to the Windows operating environment. If you use one of these device-type keywords in your SAS program under Windows, you receive an error message. Also, while the DISK device-type keyword is accepted under Windows, it is ignored because disk files are the default under Windows. △

external-file
can be any valid Windows file specification enclosed in quotes. (for more information, see "Referencing External Files" on page 106).

host-option-list
names external I/O statement options specific to Windows. They can be any of the following:

ALTDEST=*filename*
is for use only with the PRINTER device type. *Filename* specifies a file destination to write to when you direct output to the fileref. Although the output is written to disk and not to the printer, the output is still formatted using the printer driver associated with the printer that you specified with the *external-file* argument. For example,

```
filename groupHP printer
   "HP LaserJest 4si, 1st floor"
```

```
altdest=
   "C:\My SAS Files\Printer output\out.prn";
```

uses the printer driver associated with the named printer (an HP LaserJet 4si) to create the output in **out.prn**. No output is actually sent to the printer when you use this fileref.

BAUD=
sets the baud rate. This host-option is valid only if you specify the COMMPORT device-type keyword.

BITS=
sets the transmission bits. Values are 0-8. This host-option is valid only when you specify the COMMPORT device-type keyword.

BLKSIZE=*block-size*
BLK=*block-size*
specifies the number of btyes that are physically read or written in an I/O operation. The default is 8K. The maximum is 1M.

BLOCK | NOBLOCK
is used only in the context of named pipes. This option indicates whether the client is to wait if no data are currently available. BLOCK is the default value.

BYTE | MESSAGE
is used only in the context of named pipes. This option indicates the type of pipe. BYTE is the default value.

COMMAND
is used only in the context of Dynamic Data Exchange (DDE). This option enables you to issue a remote command for applications that do not use the SYSTEM topic name. For more information, see "Referencing the DDE External File" on page 218 and "Controlling Another Application Using DDE" on page 219.

COMTIMEOUT=*value*
controls how a communications port timeout is handled. A timeout occurs when no data are available at the communications port for a period of time, usually 60 seconds. The COMTIMEOUT= option can have the following values:

EOF	returns an end-of-file (EOF) character when a timeout occurs. This is the default behavior. The EOF character causes the current DATA step to terminate.
WAIT	instructs the communications port to wait forever for data. In other words, this value overrides the timeout. In this case, no record is returned to the DATA step until data are available. This can cause your program to go into an infinite loop, so use this value with caution.
ZERO	returns a record length of 0 bytes when a timeout occurs. However, the DATA step does not terminate; it simply tries to read data again.

This host-option is valid only if you specify the COMMPORT device-type keyword.

CONSOLE=*state*
specifies the state of the DOS window when an application is opened using pipes. Valid states are:

MAX	opens the DOS window maximized
MIN	opens the DOS window minimized

NORMAL opens the DOS window using the default for the machine.

This host-option is valid only if you specify the PIPE keyword.

DROPNULL=
is used to discard null bytes when received. The valid values are:

ON specifies to discard null bytes when received.

OFF specifies not to discard null bytes when received. OFF is the default value.

This host-option is valid only if you specify the COMMPORT device-type keyword. For example:

```
filename portin commport 'com1:' dropnull=off;
```

EOFCONNECT
is used only in the context of named pipes and is valid only when defining the server. This option indicates that if an end-of-file (EOF) character is received from a client, the server should try to connect to the next client.

FLOW=
controls the transmission control flow. Values are: XONXOFF, DTRDSR, or RTSCTS. This host-option is valid only if you specify the COMMPORT device-type keyword.

HOTLINK
is used only in the context of Dynamic Data Exchange (DDE). For a complete description and an example of using this option, see "Using the DDE HOTLINK" on page 223.

LRECL=*record-length*
specifies the record length (in bytes). Under Windows, the default is 256. The value of *record-length* can range from 1 to 1,048,576 (1 megabyte).

MOD
specifies that output should be appended to an existing file.

NOTAB
is used only in the context of Dynamic Data Exchange (DDE). This option enables you to use non-tab character delimiters between variables. For more information about this option, see "Using the NOTAB Option with DDE" on page 222.

PARITY=
sets the parity check bit. Values are NONE, ODD, EVEN, MARK, or SPACE. This host-option is valid only if you specify the COMMPORT device-type keyword.

RECFM=*record-format*
controls the record format. The following values are valid under Windows:

F indicates fixed format.

N indicates binary format and causes the file to be treated as a byte stream.

P indicates print format.

S370V indicates the variable S370 record format (V).

S370VB indicates the variable block S370 record format (VB).

S370VBS indicates the variable block with spanned records S370 record format (VBS).

V | D indicates variable format. This is the default.

The S370 values are valid with OS/390-style files only—that is, files that are binary, have variable-length records, and are in EBCDIC format. If you want to use a fixed-format OS/390 (formerly known as MVS) file, first copy it to a variable-length, binary OS/390 file.

RETRY=*seconds*
is used only in the context of named pipes. This option specifies how long a named pipe client should wait for a busy pipe. The minimum (and default) value for *seconds* is 10.

RCONST=*seconds*
specifies the initial read time-out value in 0.001 of a second (1000 = 1 second). The default is 8 seconds. This host-option is valid only if you specify the COMMPORT device-type keyword.

RMULTI= *seconds*
specifies the subsequent read time-out value in 0.001 of a second (1000 = 1 second). This host-option is valid only if you specify the COMMPORT device-type keyword.

SERVER | CLIENT
is used only in the context of named pipes. This option specifies the mode of a named pipe. The default value is SERVER.

STOP=
sets the stop bit. Values are ONE, TWO, ONEHALF. This host-option is valid only if you specify the COMMPORT device-type keyword.

WCONST=*seconds*
specifies the initial time-out value in 0.001 of a second (1000 = 1 second). This host-option is valid only if you specify the COMMPORT device-type keyword.

WMULTI=*seconds*
specifies the subsequent time-out value in 0.001 of a second (1000 = 1 second). This host-option is valid only if you specify the COMMPORT device-type keyword.

Details

The FILENAME statement temporarily associates a valid SAS name with an external file or an output device. An external file is a file created and maintained in the Windows operating environment from which you need to read data.

See Also

- FILENAME statement in *SAS Language Reference: Dictionary* for portable options available
- “Advanced External I/O Techniques” on page 120for examples of using some of these options

FOOTNOTE

Prints up to ten lines of text at the bottom of the procedure output

Valid: anywhere in a SAS program

Windows specifics: Maximum length of footnote

Syntax

FOOTNOTE <*n*> <'*text*' | "*text*">;

n
specifies the relative line to be occupied by the footnote.

text
specifies the text of the footnote in single or double quotation marks.

Details

The FOOTNOTE statement takes effect when the step or RUN group with which it is associated executes. Once you specify a footnote for a line, SAS repeats the same footnote on all pages until you cancel or redefine the footnote for that line.

The maximum footnote length under Windows is 256 characters. If the specified footnote is greater than the LINESIZE system option, the footnote is truncated to the line size.

See Also

□ FOOTNOTE statement in *SAS Language Reference: Dictionary*

%INCLUDE

Includes and executes SAS statements and data lines

Valid: anywhere

Windows specifics: *source*, if a file-specification is used; valid options for *host-options*

Syntax

%INCLUDE *source* </*host-options*>;

Note: This is a simplified version of the %INCLUDE statement syntax. For the complete syntax and its explanation, see the %INCLUDE statement in *SAS Language Reference: Dictionary*. △

source
describes the location of the information you want to access with the %INCLUDE statement. The two possible sources are a file specification or internal lines. Under Windows, you cannot use an asterisk (*) to specify keyboard entry. The file specification can be any of the file specification forms discussed in "Referencing External Files" on page 106.

host-options
consists of statement options valid under Windows. Remember to precede the options list with a forward slash (/). Currently, the following options are available under Windows:

BLKSIZE=*block-size*
BLK=*block-size*
specifies the number of bytes that are physically read or written in an I/O operation. The default is 8K. The maximum is 1M.

BLOCK | NOBLOCK
is used only in the context of named pipes. This option indicates whether the client is to wait if no data are currently available.

BYTE | MESSAGE
is used only in the context of named pipes. This option indicates the type of pipe; BYTE is the default value.

EOFCONNECT
is used only the context of named pipes and is valid only when defining the server. This option indicates that if an end-of-file (EOF) character is received from a client, the server should try to connect to the next client.

LRECL=*record-length*
specifies the record length (in bytes). Under Windows, the default is 256. The value of *record-length* can range from 1 to 1,048,576 (1 megabyte).

NOTAB
is used only in the context of Dynamic Data Exchange. This option enables you to use non-tab character delimiters between variables. For more information on this option, see "Using the NOTAB Option with DDE" on page 222

RECFM=*record-format*
controls the record format. The following values are valid under Windows:

F	indicates fixed format.
N	indicates binary format and causes the file to be treated as a byte stream.
P	indicates print format.
V|D	indicates variable format. This is the default.
S370V	indicates the variable S370 record format (V).
S370VB	indicates the variable block S370 record format (VB).
S370VBS	indicates the variable block with spanned records S370 record format (VBS).

The S370 values are valid with OS/390-style files only—that is, files that are binary, have variable-length records, and are in EBCDIC format. If you want to use a fixed-format OS/390 (formerly known as MVS) file, first copy it to a variable-length, binary OS/390 file.

Details

When you execute a program that contains the %INCLUDE statement, SAS executes your code, including any statements or data lines that you bring into the program with %INCLUDE.

See Also

□ %INCLUDE statement in *SAS Language Reference: Dictionary*

INFILE

Specifies an external file to read with an INPUT statement

Valid: in a DATA step

Windows specifics: Valid values for *file-specification*; valid values for *host-options*

Syntax

INFILE *file-specification* <*options*> <*host-options*>;

file-specification
identifies the source of input data records, usually an external file. The *file-specification* argument can be any of the file specification forms discussed in "Referencing External Files" on page 106. The reserved fileref CARDS enables the INFILE statement to reference instream data.

Note: The words CON, NUL, PRN, LPT1 - LPT9, and COM1 - COM9 are reserved words under Windows. Do not use them as filerefs. △

host-options
names external I/O statement options that are specific to the Windows operating environment. They can be any of the following:

BLKSIZE= *block-size*
BLK=*block-size*
specifies the number of bytes that are physically read or written in an I/O operation. The default is 8K. The maximum is 1M.

BLOCK | NOBLOCK
is used only in the context of named pipes. This option indicates whether the client is to wait if no data are currently available. The default value is BLOCK.

BYTE | MESSAGE
is used only in the context of named pipes. This option indicates the type of pipe. The default value is BYTE.

EOFCONNECT
is used only in the context of named pipes and is valid only when defining the server. This option indicates that if an end-of-file (EOF) character is received from a client, the server should try to connect to the next client.

HOTLINK
is used only in the context of Dynamic Data Exchange (DDE). For a complete description and an example of using this option, see "Using the DDE HOTLINK" on page 223.

LRECL=*record-length*
specifies the record length (in bytes). Under Windows, the default is 256. The value of *record-length* can range from 1 to 1,048,576 (1 megabyte).

NOTAB
is used only in the context of Dynamic Data Exchange (DDE). This option enables you to use nontab character delimiters between variables. For more information about this option, see "Using the NOTAB Option with DDE" on page 222.

RECFM=*record-format*
controls the record format. The following are valid values under Windows:

F	indicates fixed format.
N	indicates binary format and causes the file to be treated as a byte stream.
P	indicates print format.
S370V	indicates the variable S370 record format (V).
S370VB	indicates the variable block S370 record format (VB).
S370VBS	indicates the variable block with spanned records S370 record format (VBS).
V | D	indicates variable format. This is the default.

The S370 values are valid with OS/390–style files only that is, files that are binary, have variable-length records, and are in EBCDIC format. If you want to use a fixed-format OS/390 (formerly known as MVS) file, first copy it to a variable-length, binary OS/390 file.

RETRY=*seconds*
is used only in the context of named pipes. This option specifies how long a named pipe client should wait for a busy pipe. The minimum (and default) value for *seconds* is 10.

SERVER | CLIENT
is used only in the context of named pipes. This option specifies the mode of a named pipe. The default value is SERVER.

See Also

- INFILE statement in *SAS Language Reference: Dictionary* for portable options available
- "Named Pipe Examples" on page 232for examples of using some of these options
- "DDE Examples" on page 220for examples of using some of these options.

LENGTH

Specifies the number of bytes the SAS System uses to store numeric variables

Valid: in a DATA step

Windows specifics: Valid numeric variable lengths; valid values for *length*; valid values for *n*

Syntax

LENGTH <*variable-1*><*…variable-n*> <$> <*length*> <DEFAULT=*n*>;

length
Under Windows, can range from 3 to 8 bytes for numeric variables.

DEFAULT=*n*
changes the default number of bytes used for storing the values of newly created numeric variables from 8 to the value of *n*. Under Windows, the value of *n* can range from 3 to 8 bytes.

Details

The LENGTH statement specifies the number of bytes the SAS system is to use for storing values of variables in each data set being created.

CAUTION:
Any length less than 8 bytes may result in a loss of precision for the value of the variable. △

See Also

- LENGTH statement in *SAS Language Reference: Dictionary*
- "Length and Precision of Variables under Windows" on page 481

LIBNAME

Associates a libref with a SAS data library and lists file attributes for a SAS data library

Valid: anywhere in a SAS program

Windows specifics: Valid values for *engine*; specifications for *SAS-data-library*

Syntax

LIBNAME *libref* <*engine*> '('*SAS-data-library-1*' <,...'*SAS-data-library-n*'>)'<MEMLIB> <LONGFILEEXT | SHORTFILEEXT>;

LIBNAME *libref* _ALL_ LIST;

LIBNAME *libref* _ALL_ CLEAR;

Note: This is a simplified version of the LIBNAME statement syntax. For the complete syntax and its explanation, see the LIBNAME statement in *SAS Language Reference: Dictionary*. △

libref
is any valid libref, as documented in *SAS Language Reference: Dictionary*.

engine-name
is one of the following library engines supported under Windows:

V8	accesses Version 8 data sets. You can use the nickname BASE for this engine.
V7	accesses Version 7 data sets.

V6	accesses Release 6.08 through Release 6.12 data sets.
V612	accesses Release 6.12 data sets.
V611	accesses Release 6.11 data sets.
V610	accesses Release 6.10 data sets.
V609	accesses Release 6.09 data sets.
V608	accesses Release 6.08 data sets.
V604	accesses Release 6.03 and Release 6.04 data sets.
XPORT	accesses transport format files.
BMDP	accesses BMDP data files.
OSIRIS	accesses OSIRIS data files.
SPSS	accesses SPSS system files.

For more information about these engines, see "Multiple Engine Architecture" on page 83 and the discussion of engines in *SAS Language Reference: Dictionary*.

SAS-data-library
is the physical name of a SAS data library under Windows. It must be a valid Windows pathname. You can concatenate several Windows directories together to serve as a single SAS data library. When you specify multiple libraries, use parenthesis around the first and last library pathnames. For more information on concatenated SAS data libraries, see "Understanding How Concatenated SAS Data Libraries Are Accessed" on page 91.

MEMLIB
specifies to use Extended Server Memory Architecture (ESMA) memory for this library. For more information on using ESMA memory, see "Processing SAS Libraries in Extended Server Memory Architecture Memory" on page 151.

LONGFILEEXT | SHORTFILEEXT
specifies whether the library supports long file extensions or short file extensions (3 characters). LONGFILEEXT is the default.

If SAS is not able to create a file with a long file extension in a library, then the library supports only files with short file extensions. If you specify a file with a long file extension for a library that supports only short file extension, an error message informs you that the member name is too long for the system.

Details

The LIBNAME statement associates a libref with a permanent SAS data library. It also can be used to list the file attributes of a SAS data library. (The LIBNAME statement is also used to clear a libref. For more information, see "Clearing Librefs" on page 90.)

The SAS Explorer window provides an intuitive interface to assigning SAS data libraries. For information about managing SAS data libraries using this interactive interface, see SAS introduction books.

Note: The words CON, NUL, PRN, LPT1 - LPT9, and COM1 - COM9 are reserved words under Windows. Do not use them as librefs. △

Associating Librefs Use one of the following forms of the LIBNAME statement to associate a libref or an engine with a SAS data library:

LIBNAME *libref* <*engine*> '*SAS-data-library*'

LIBNAME *libref* <*engine*> (*SAS-data-library-1*<,…'*SAS-data-library-n*'>);

You can use the same arguments with these forms of the LIBNAME statement as shown in "Syntax" on page 384.

Listing Data Library Attributes With the LIST option, you can use the LIBNAME statement to list attributes of SAS data libraries. Output 20.1 on page 386 shows the results of the following LIBNAME statement:

```
libname sashelp list;
```

Output 20.1 Data Library Attributes Listed by the LIBNAME Statement

```
5    libname sashelp list;
NOTE: Libref=   SASHELP
      Scope=    Kernel
      Levels=   24
        -Level 1-
      Engine=   V8
      Physical Name= c:\program files\sasv8\SASCFG
      File Name=c:\program files\sasv8\SASCFG
        -Level 2-
      Engine=   V8
      Physical Name= c:\program files\sasv8\core\sashelp
      File Name=c:\program files\sasv8\core\sashelp
        -Level 3-
      Engine=   V8
      Physical Name= c:\program files\sasv8\base\sashelp
      File Name= c:\program files\sasv8\base\sashelp

      . . .

         -Level 24-
      Engine=   V8
      Physical Name= c:\program files\sasv8\whouse\sashelp
      File Name= c:\program files\sasv8\whouse\sashelp
6    run;
```

See Also

- LIBNAME statement in *SAS Language Reference: Dictionary*
- "Using Data Libraries" on page 85

SYSTASK

Executes, lists, or terminates asynchronous tasks

Valid in: anywhere in a SAS program

Windows specifics: all

Syntax

SYSTASK COMMAND "*host-command*"
    <XWAIT | NOXWAIT>
    <TASKNAME=*taskname*>
    <MNAME=*name-var*>

<STATUS=*stat-var*>
<SHELL<="*shell-command*">>;

SYSTASK LIST <_ALL_ | *taskname*>
STATE
<STATVAR>;

SYSTASK KILL *taskname* <*taskname*...>;

COMMAND
executes the *host-command*

LIST
lists either a specific active task or all of the active tasks in the system

KILL
forces the termination of the specified task(s).

host-command
specifies the name of a Windows command (including any command-specific options). Enclose the command in either single or double quotation marks. If the command options require quotes, repeat the quotes. For example:

```
systask command "find ""my text"" c:\mydir\myfile.sas"
```

Note: The *host-command* that you specify cannot require input from the keyboard. △

XWAIT | NOXWAIT
determines whether SYSTASK COMMAND suspends execution of the current SAS session until the task has completed. NOXWAIT is the default. For tasks that are started with the NOXWAIT option, you can use the WAITFOR statement when necessary to suspend execution of the SAS session until the task has finished.

TASKNAME=*taskname*
specifies a name that identifies the task. Task names must be unique among all active tasks. A task is active if it is running, or if it has completed and has not been waited for using the WAITFOR statement. Duplicate task names generate an error in the SAS log. If you do not specify a task name, SYSTASK will automatically generate a name. If the task name contains a blank character, enclose the task name in quotes.

MNAME=*name-var*
specifies a macro variable in which you want SYSTASK to store the task name that it automatically generated for the task. If you specify both the TASKNAME option and the MNAME option, SYSTASK copies the name you specified with TASKNAME into the variable that you specified with MNAME.

STATUS=*stat-var*
specifies a macro variable in which you want SYSTASK to store the status of the task. Status variable names must be unique among all active tasks.

SHELL<="*shell-command*">
specifies that the *host-command* should be executed with the host shell command. If you specify a shell-command, SYSTASK uses the shell command that you specify to invoke the shell; otherwise, SYSTASK uses the default shell. Enclose the shell command in quotes.

ALL
specifies all active tasks in the system.

STATE
displays the status of the task, which can be Start Failed, Running, or Complete.

STATVAR
displays the status variable associated with the task. The status variable is the variable that you assigned with the STATUS option in the SYSTASK COMMAND statement.

Details

SYSTASK allows you to execute host-specific commands from within your SAS session or application. Unlike the X statement, SYSTASK runs these commands as asynchronous tasks, which means that these tasks execute independently of all other tasks that are currently running. Asynchronous tasks run in the background, so you can perform additional tasks while the asynchronous task is still running.

For example, to execute the Windows DIR command in the SAS WORK directory, you might use this statement:

```
systask command "dir c:\temp\sas temporary files\"
         taskname="workdir" status=dirstat;
```

The return code from the DIR command is saved in the macro variable DIRSTAT. The output from the command is displayed in the the SAS log.

Note: Program steps that follow the SYSTASK statements in SAS applications usually depend on the successful execution of the SYSTASK statements. Therefore, syntax errors in some SYSTASK statements will cause your SAS application to abort. △

There are two types of tasks that can be run with SYSTASK:

Task
All tasks started with SYSTASK COMMAND are of type Task. For these tasks, if you do no specify STATVAR or STATE, then SYSTASK LIST displays the task name, type, and state, and the name of the status macro variable. You can use SYSTASK KILL to terminate only tasks of type Task.

SAS/Connect Process
Tasks started from SAS/Connect with the SYSTASK BEGIN statement are of type SAS/Connect Process. For SAS/Connect processes, SYSTASK LIST displays the task name, type, and state. You cannot use SYSTASK KILL to terminate a SAS/Connect process. For information on starting SAS/Connect processes with SYSTASK, see *SAS/CONNECT User's Guide*.

The SYSRC macro variable contains the return code for the SYSTASK statement. The status variable that you specify with the STATUS option contains the return code of the process started with SYSTASK COMMAND. To ensure that a task executes successfully, you should monitor both the status of the SYSTASK statement and the status of the process that is started by the SYSTASK statement.

If a SYSTASK statement cannot execute successfully, the SYSRC macro variable will contain a non-zero value. For example, there may be insufficient resources to complete a task, or the SYSTASK statement may contain systax errors. With the SYSTASK KILL statement, if one or more of the processes cannot be terminated, SYSRC is set to a non-zero value.

When a task is started, its status variable is set toNULL. You can use the status variables for each task to determine which tasks failed to complete. Any task whose status variable is NULL did not complete execution. See WAITFOR for more information about the status variables.

Unlike the X statement, you cannot use the SYSTASK statement to start a new interactive session.

See Also

- Statement: "WAITFOR" on page 389
- Statement: "X" on page 391

TITLE

Specifies title lines for SAS output

Valid: anywhere in a SAS program
Windows specifics: Maximum length of the title

Syntax

TITLE <*n*> <*'text'* | *"text"*>;

n
specifies the relative line that contains the title line.

'text'* | *"text"
specifies text that is enclosed in single or double quotation marks.

Details

The TITLE statement specifies title lines to be printed on SAS print files and other SAS output. A TITLE statement takes effect when the DATA or PROC step or RUN group with which it is associated executes. Once you specify a title for a line, it is used for all subsequent output until you cancel the title or define another title for that line.

Under Windows, the maximum title length is 256 characters. If the specified title is greater than the LINESIZE system option, the title is truncated to the line size.

See Also

- TITLE statement in *SAS Language Reference: Dictionary*

WAITFOR

Suspends execution of the current SAS session until the specified tasks finish executing

Valid in: anywhere in a SAS program
Windows specifics; all

Syntax

WAITFOR<_ANY_ | _ALL> *taskname* <*taskname*...><TIMEOUT=*seconds*>;

taskname
specifies the name of the task(s) that you want to wait for. See "SYSTASK" on page 386 for information about task names. The task name(s) that you specify must match exactly the task names assigned through the SYSTASK COMMAND statement. You cannot use wildcards to specify task names.

ANY | _ALL_
suspends execution of the current SAS session until either one or all of the specified tasks finishes executing. The default setting is _ANY_, which means that as soon as one of the specified task(s) completes executing, the WAITFOR statement will finish executing.

TIMEOUT=*seconds*
specifies the maximum number of seconds that WAITFOR should suspend the current SAS session. If you do not specify the TIMEOUT option, WAITFOR will suspend execution of the SAS session indefinitely.

Details

The WAITFOR statements suspends execution of the current SAS session until the specified task(s) finish executing or until the TIMEOUT interval (if specified) has elapsed. If the specified task was started with the XWAIT option, then the WAITFOR statement ignores that task. See "SYSTASK" on page 386for a description of the XWAIT option.

For example, the following statements start three different SAS jobs and suspend the execution of the current SAS session until those three jobs have finished executing:

```
systask command "sas myprog1.sas" taskname=sas1;
systask command "sas myprog2.sas" taskname=sas2;
systask command "sas myprog3.sas" taskname=sas3;
waitfor _all_ sas1 sas2 sas3;
```

The SYSRC macro variable contains the return code for the WAITFOR statement. If a WAITFOR statement cannot execute successfully, the SYSRC macro variable will contain a non-zero value. For example, the WAITFOR statement may contain syntax errors. If the number of seconds specified with the TIMEOUT option elapses, then the WAITFOR statement finishes executing, and SYSRC is set to a non-zero value if

- you specify a single task that does not finish executing
- you specify more than one task and the _ANY_ option (which is the default setting), but none of the tasks finishes executing
- you specify more than one task and the _ALL_ option, and any one of the tasks does not finish executing.

Any task whose status variable is still NULL after the WAITFOR statement has executed did not complete execution. See "SYSTASK" on page 386 for a description of status variables for individual tasks.

See Also

- Statement: “SYSTASK” on page 386
- “X” on page 391

X

Runs an operating system command or a Windows application from within a SAS session

Valid: anywhere in a SAS program

Windows specifics: Valid values for *command*

Syntax

X <*'command'*>;

no argument
places you in a Command prompt session, with an operating system prompt. Here you can execute Windows commands in the context of the SAS System working directory. There are some things that you cannot do from the Command prompt in this situation, such as define environment variables for use by your SAS session. (Environment variables must be defined before you invoke the SAS System). Type EXIT at the Command prompt and press ENTER to return to your SAS session.

command
specifies a Windows command or a Windows application. This argument can be anything you can specify at a DOS prompt (including the SAS command). Therefore, you can use the X statement to execute Windows applications. The command can be enclosed in quotes, but this is not required syntax.

The command is passed to Windows and executed in the context of the working directory. If errors occur, the appropriate error messages are displayed.

By default, you must type EXIT to return to your SAS session after the command has completed execution. Also by default, if you execute a Windows application such as Notepad, you must close the application before you can return to your SAS session. Specify NOXWAIT in an OPTIONS statement if you do not want to have to type EXIT. With NOXWAIT in effect, as soon as the command finishes execution, control is returned to your SAS session. Note, however, that if you execute a Windows application with the X statement, specifying NOXWAIT does not let you return to your SAS session until you close the application.

Another system option, XSYNC, controls whether you have to wait for the command to finish executing before you can return to your SAS session. If you specify NOXSYNC, you can start a Windows application with the X statement and return to your SAS session without closing the application. For additional details about these two system options, see “XWAIT” on page 478 and “XSYNC” on page 477.

Details

The X statement issues a host command from within a SAS session when you run SAS in windowing mode. The SAS System executes the X statement immediately.

Under Windows, you can issue the X statement without the *command* argument.

There are other ways of running operating environment commands besides the X statement (and the X command) under Windows.

See Also

- X statement in *SAS Language Reference: Dictionary*
- Command: "X" on page 314
- System option: "XSYNC" on page 477
- System option: "XWAIT" on page 478
- CALL routine: "CALL SYSTEM" on page 330
- Statement: "Macro Statements" on page 484
- "Running DOS or Windows Commands from within SAS" on page 20
- "Adding Applications to the Tools Menu" on page 69

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

ISBN 1–58025–524–8

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

1st printing, September 1999

The Institute is a private company devoted to the support and further development of its software and related services.