CHAPTER 2

The BUILD Procedure

Overview

You use the BUILD procedure in SAS/AF software to create applications that can

□ display or update the contents of SAS data sets, catalogs, or external files, using either a graphical or character-based user interface

□ create SAS statements to be submitted for processing by other SAS System procedures

□ provide menus for selecting other applications

□ provide computer-based training

□ provide online Help.

This section provides reference information about the BUILD procedure, including the complete syntax of the statements that are used in the procedure.

Note: You can also open the BUILD procedure windows by using

□ the BUILD command

□ the CALL BUILD routine in SAS Component Language

□ the SAS Explorer window (when you select one of the SAS/AF catalog entry types described in Chapter 3, “SAS/AF Catalog Entry Types,” on page 21).

See “BUILD Command” on page 18 for details about the BUILD command. Refer to *SAS Component Language: Reference* for information about using the CALL BUILD routine. Refer to base SAS documentation for information about using the Explorer window. △

BUILD Procedure Syntax

Note: You can use any number of COMPILE, MERGE, MLINK, PRINT, and SYNC statements with the PROC BUILD statement. The statements execute before any procedure windows are opened. All MERGE statements are executed first, followed by all PRINT statements, all MLINK statements, all SYNC statements, and finally all COMPILE statements.

PROC BUILD <CATALOG=<*libref.*>*catalog-name*<*.entry-name.entry-type*>>
<*options*>;

COMPILE <*options*>;

CROSSREF PROJECT=*libref.catalog-name* | (*catalog-list*) <*options*>;

MERGE CATALOG=*libref.catalog-name* <*options*>;

MLINK <*options*>;

PRINT *items* <*options*>;

SYNC <*options*>;

The PROC BUILD statement is required. The other statements are optional and are used as follows:

To do this	Use this statement
Compile the SAS Component Language code in FRAME, PROGRAM, and SCL entries	COMPILE
Collect information for the Static Analyzer performance analysis tool	CROSSREF
Combine entries from another SAS catalog into the current catalog	MERGE
Generate submenu links between MENU entries	MLINK
Print the contents of catalog entries	PRINT
Synchronize changes in RESOURCE, CLASS, or FRAME entries	SYNC

PROC BUILD Statement

Starts the BUILD procedure and specifies the catalog or catalog entry to open.

Tip: When you specify a four-level name for the CATALOG= argument, a BUILD procedure window opens in which you can edit the specified entry. By default, when you close the BUILD window, the procedure opens an Explorer window showing the contents of the catalog that contains the specified entry. If you want the procedure to end when you close the BUILD window, use the NODIR option in the PROC BUILD statement.

PROC BUILD <BATCH>
<BROWSE>
<CATALOG=<*libref.*>*catalog-name*<*.entry-name*<*.entry-type*>>>

<ENTRYTYPE=*entry-type*>

<NODIR>

<PADCHAR='*character*'>

<RESOURCE=<*libref.catalog-name.*>*resource-name*<.RESOURCE>>

<TESTAF <DEBUG>>

<TEXTLENGTH=*n*>;

Options

You can use the following options in the PROC BUILD statement:

BATCH

executes the BUILD procedure in batch mode rather than interactively. You cannot build individual entries in batch mode, but it is a convenient way to execute PRINT, COMPILE, MERGE, MLINK, and SYNC statements when you do not need to view the catalog or catalog entry. If you use the BATCH option without a PRINT, COMPILE, MERGE, MLINK, or SYNC statement, the specified catalog or catalog entry is not opened.

Note: When you use the BATCH option, all of the other PROC BUILD options except for CATALOG= are ignored. △

BROWSE

opens the specified catalog entry for browsing only. By default, the BUILD procedure tries to open the specified catalog entry for editing. Use the BROWSE option if you only want to view the entry without making changes.

Note: Catalog entries for which you have read-only access are automatically opened for browsing. △

CATALOG=<*libref.*>*catalog-name*<*.entry-name.entry-type*>
CAT=<*libref.*>*catalog-name*<*.entry-name.entry-type*>
C=<*libref.*>*catalog-name*<*.entry-name.entry-type*>

specifies the current catalog or the specific catalog entry to create, edit, or browse. The name that you specify with this argument is interpreted as follows:

- □ A one-level name identifies a catalog in the default library, WORK.
- □ A two-level name identifies a catalog in a specified library.
- □ A three-level name also identifies a catalog in a specified library. The third level is ignored and the first two levels (*libref.catalog-name*) are used.
- □ A four-level name identifies an entry of a specified type in a specified catalog.

Unless you also use the BATCH option, the procedure opens a window.

- □ If you supply a one-, two-, or three-level name, an Explorer window opens to display the contents of the specified catalog.
- □ If you specify a four-level name, the appropriate window opens for the specified entry type. See "BUILD Procedure Windows" on page 19 for information about the corresponding window for each entry type. Refer to Chapter 3, "SAS/AF Catalog Entry Types," on page 21 for information about the uses of the different catalog entry types. If the specified entry does not already exist, a new entry of the specified type is created.

If you omit the CATALOG= argument, a SAS Explorer window opens from which you can select a library and catalog and then create a new entry or select an existing entry to edit or browse.

ENTRYTYPE=*entry-type*
ETYPE=*entry-type*
ET=*entry-type*

specifies the default catalog entry type for the EDIT and BROWSE commands in BUILD procedure windows. The initial default entry type is PROGRAM. Use the ENTRYTYPE= option to change the default entry type to CBT, FRAME, HELP, MENU, or SCL.

NODIR

prevents the Explorer window from opening when a specified catalog entry is closed. By default, the Explorer window opens with the contents of the current catalog displayed when you close the BUILD procedure window. This makes it easy to continue working in the same catalog. Use the NODIR option to prevent this default behavior, in which case the BUILD procedure ends when you close the BUILD procedure window for the entry.

Note: The NODIR option is valid only when you specify a four-level name in the CATALOG= argument. △

PADCHAR='*character*'

specifies the default pad character that is displayed for empty user fields in PROGRAM entry windows. By default, the pad character is the underscore (_). Use the PADCHAR= option to specify a different default pad character for the BUILD session.

Note: The pad character for individual fields can be specified in the ATTR window. △

RESOURCE=<*libref.catalog-name.*>*resource-name*<.RESOURCE>

specifies the RESOURCE entry that is associated with any FRAME entries you create with the BUILD procedure. By default, new FRAME entries use the RESOURCE entry that is specified in the SAS registry. You can use the BUILD= option to override the default for the current BUILD session. The *resource-name* value must be the name of an existing RESOURCE entry. If you omit the *libref.catalog-name* value, the procedure looks for the specified RESOURCE entry in the current catalog, which is identified in the CATALOG= option.

Note: You can use the RESOURCE command in a BUILD procedure window to change the default RESOURCE entry during the BUILD session. △

TESTAF <DEBUG>

executes the specified CBT, FRAME, HELP, MENU, PROGRAM, or SCL entry in a test environment. You can test features such as field validation, the appearance of windows, and flow of control. Statements within submit blocks in PROGRAM entries are not submitted to the SAS System for processing. When you close the entry in the test environment, the Explorer window opens unless you also use the NODIR option.

Note: The TESTAF option is valid only when you specify a four-level name in the CATALOG= argument. △

The DEBUG option activates the SAS Component Language source-level debugger, provided that the SCL program in the specified entry was previously compiled with the DEBUG compile option.

TEXTLENGTH=*n*
TEXTLEN=*n*

specifies the length of the text line in the DISPLAY window for catalog entry types that use the SAS text editor (CBT, HELP, MENU, and PROGRAM). The default value is 78. Use the TEXTLENGTH= option to specify a shorter or longer line length. Valid values for *n* are 1 through 255.

This option is especially useful when you are developing applications that will be used on displays that are wider than the one on which they are being created.

Using the PROC BUILD Statement

The BUILD procedure enables you to create, edit, and manage SAS/AF catalogs and catalog entries. The types of entries you can build depend on your display environment.

- In graphical display environments, FRAME entries provide a graphical user interface and object-oriented programming tools for application development. The information in CLASS and RESOURCE entries defines the objects that are available in the development environment.
- In character-based display environments, SAS/AF applications usually consist of a combination of PROGRAM, CBT, MENU, and HELP entries. The information in EDPARMS, FORM, and KEYS entries supplements the application.

Refer to Chapter 3, “SAS/AF Catalog Entry Types,” on page 21 for more information about the different types of catalog entries you can create and edit with the BUILD procedure.

COMPILE Statement

Compiles an application’s source programs into stored, executable code.

Note: The compile operation is performed before the procedure opens any windows.

Tip: By default, all PROGRAM entries in the current catalog are compiled. You can use the SELECT= option to select individual entries to compile, or use the EXCLUDE= option to prevent certain entries from being compiled.

COMPILE <DEBUG>
<EXCLUDE=*entry-name* | (*entry-list*)> | <SELECT=*entry-name*<*.entry-type*> | (*entry-list*)>;

Options

You can use the following options in the COMPILE statement:

DEBUG
specifies that extra information is collected for the SAS Component Language (SCL) debugger when programs are compiled.

EXCLUDE=*entry-name* | (*name-list*)
SELECT=*entry-name*<*.entry-type*> | (*name-list*)
specify entries to exclude from or select for the compile operation. By default, all PROGRAM entries in the current catalog are compiled. Use the EXCLUDE= option to prevent specific PROGRAM entries from being compiled. Use the SELECT= option to compile only the specified entries. To compile FRAME or SCL entries, you must explicitly specify the entry names, including the entry type, by using the SELECT= option. An error occurs if you specify an entry of any type other than FRAME, PROGRAM, or SCL. If you specify more than one entry name for either of these

options, enclose the list in parentheses and separate the names with at least one space.

Each COMPILE statement can include only one EXCLUDE= option or one SELECT= option. However, you can use multiple COMPILE statements in the same PROC BUILD statement.

Using the COMPILE Statement

The SAS Component Language code in FRAME, PROGRAM, and SCL entries must be compiled before the source programs can run. If you do not use the COMPILE command while you are working in the catalog entries, you can use the COMPILE statement to compile the code in existing entries.

If error or warning messages are generated by the compile process, they are written to the SAS log.

COMPILE Statement Example

The following example illustrates a COMPILE statement that compiles three SCL source programs and collects the extra information needed to run the SCL debugger. In this example, the PROGRAM entries named ONE, TWO, and THREE in the catalog EX.APPLIC are compiled in batch mode. No interactive BUILD session is initiated.

```
proc build catalog=ex.applic batch;
   compile debug select=(one two three);
run;
```

CROSSREF Statement

Collects information about the SAS Component Language code in a SAS/AF application for use by the Static Analyzer tool.

Restriction: When you use a CROSSREF statement with the PROC BUILD statement, no other BUILD procedure statements can be used.

Note: When you use the CROSSREF statement, the procedure ends after the analysis data is collected. No BUILD procedure windows are opened, and any options that you specify for the PROC BUILD statement are ignored.

CROSSREF PROJECT=<*libref.*>*catalog-name* | (*catalog-list*)

<DETAIL=ALL | NONE | <*libref.*>*catalog-name* | (*catalog-list)*>

<EXCLUDE=*entry-name.entry-type* | (*entry-list*)> | <SELECT=*entry-name.entry-type* | (*entry-list*)>

<OUTLIB=*libref*>

<SEARCH=<*libref.*>*catalog-name* | (*catalog-list*)>;

Required Argument

You must always supply the following argument with the CROSSREF statement:

PROJECT= <*libref.*>*catalog-name* | (*catalog-list*)
specifies one or more SAS catalogs that comprise the project you want to analyze. If you specify more than one catalog, enclose the list in parentheses and separate the names with spaces.

Options

You can use the following options in the CROSSREF statement:

DETAIL=ALL | NONE | <*libref.*>*catalog-name* | (*catalog-list*)
controls the depth of data collection when functions that refer to other entries (such as CALL DISPLAY, CALL METHOD, or CALL GOTO) are encountered. Specify one of the following values:

ALL
the analysis traverses all links and collects data on all other entries that are referenced in the project. (This is the default behavior.)

NONE
the analysis does not traverse any links nor does it collect data on any entries outside the specified project catalogs.

<*libref.*>*catalog-name* | (*catalog-list*)
the analysis traverses links and collects data on only those entries that reside in the specified catalogs. If you specify more than one catalog name, you must enclose the list in parentheses and separate the names with spaces.

EXCLUDE=*entry-name.entry-type* | (*entry-list*)
SELECT=*entry-name.entry-type* | (*entry-list*)
specifies one or more entries for which no data is collected or for which data is collected. Use the EXCLUDE= option to prevent data from being collected for the specified entries. Data is still collected for the remaining entries in the catalogs specified in the PROJECT= or DETAIL= arguments. Use the SELECT= option to collect data for only the specified entries in the catalogs specified in the PROJECT= or DETAIL= arguments. If you specify more than one catalog entry name, you must enclose the list in parentheses and separate the names with spaces.

The EXCLUDE= and SELECT= options are mutually exclusive. You cannot use both options in the same CROSSREF statement.

OUTLIB=*libref*
specifies the libref for the SAS data library in which the analysis data sets are generated. The new analysis data sets replace any previous analysis data sets in the specified library. If you omit this option, the default is the WORK library.

SEARCH=<*libref.*>*catalog-name* | (*catalog-list*)
specifies one or more SAS catalog names that are used as the catalog search rule. The search rule is applied when a function that refers to another entry (such as CALL DISPLAY, CALL METHOD or CALL GOTO) includes a parameter that has a two-level name in the form *entry-name.entry-type*. The catalogs in the search rule are searched in the specified order for an entry that matches the two-level entry name. When no search rule is in effect, the default is to search only the catalog that contains the entry that includes the function.

If you specify more than one catalog name, you must enclose the list in parentheses and separate the names with spaces.

Using the CROSSREF Statement

The CROSSREF statement performs the data collection phase for the Static Analyzer, one of the performance analysis tools that are provided with SAS/AF software. The

Static Analyzer scans the SCL code for a project and reports usage information for the SCL program elements, including which functions, variables, constants, widgets, SAS macros, and SAS options are used in the project. For purposes of the analysis, you can define a project to include entries in multiple SAS catalogs and SAS data libraries.

After using the CROSSREF statement to collect data, you can view the results by issuing the following command in any SAS System window:

SCLPROF STATIC <LIB=*analysis-library*>

Note: Use the LIB= option in the SCLPROF command if you used the OUTLIB= option in the CROSSREF statement to specify a library other than the default. △

For more information about the Static Analyzer, refer to the online Help for the tool.

CROSSREF Statement Examples

The following example illustrates the use of the SELECT= option to restrict data collection to a single entry:

```
proc build;
   crossref project=sashelp.aftools
            select=astopts.frame;
run;
```

The following example illustrates the use of the DETAIL= option to limit the depth of data collection in a large project:

```
proc build;
   crossref project=(sashelp.eis sashelp.eisbol1)
            detail=(sashelp.afclass sashelp.eisbol1
                    sashelp.fsp sashelp.mb)
            search=(sashelp.eisbol1 sashelp.mb
                    sashelp.eis sashelp.fsp
                    sashelp.assist sashelp.calc)
      outlib=sasuser;
run;
```

MERGE Statement

Copies entries from the catalog specified in the MERGE statement into the current catalog.

Note: The merge operation is performed before the procedure opens any windows.

Tip: By default, all entries in the specified catalog are copied to the current catalog (the catalog that was specified in the PROC BUILD statement). You can use the SELECT= option to select individual entries to copy, or use the EXCLUDE= option to prevent certain entries from being copied.

MERGE CATALOG=<*libref.*>*catalog-name*

<ENTRYTYPE=*type*>

<EXCLUDE=*entry-name.entry-type* | (*entry-list*)> | <SELECT=*entry-name.entry-type* | (*entry-list*)>

<NOEDIT>

<NOSOURCE>

<REPLACE>

<UPCASE>;

Required Argument

You must always supply the following argument with the MERGE statement:

CATALOG=<*libref.*>*catalog-name*
CAT=<*libref.*>*catalog-name*
C=<*libref.*>*catalog-name*
specifies the SAS catalog from which entries are copied. If you specify a one-level name, it is assumed to be a catalog name in the default WORK library.

Options

You can use the following options in the MERGE statement:

ENTRYTYPE=*entry-type*
ETYPE=*entry-type*
ET=*entry-type*
specifies the type of entry to copy into the merged catalog. By default, entries of all types are copied (unless you also use the SELECT= or EXCLUDE= option). Use the ENTRYTYPE= option to copy only entries of the specified type.

To copy entries of more than one type, use a separate MERGE statement for each entry type.

EXCLUDE=*entry-name.entry-type* | (*entry-list*)
SELECT=*entry-name.entry-type* | (*entry-list*)
specify entries to exclude from or select for the merge operation. By default, all entries in the specified catalog are copied into the current catalog (unless you also use the ENTRYTYPE= option). Use the EXCLUDE= option to prevent specific entries from being copied. Use the SELECT= option to copy only the specified entries. If you specify more than one entry name for either of these options, enclose the list in parentheses and separate the names with at least one space.

If you use the ENTRYTYPE= option in the same MERGE statement, you can omit the *entry-type* portion of the entry specification for the EXCLUDE= and SELECT= options because only entries of the type specified in the ENTRYTYPE= option can be copied.

Each MERGE statement can include only one EXCLUDE= option or one SELECT= option. However, you can use multiple MERGE statements with the same PROC BUILD statement.

NOEDIT
NOED
specifies that the entries that are merged into the current catalog cannot be edited with the BUILD procedure. This option is useful when you are moving entries from a development catalog into a production catalog and you want to prevent changes to the entries in the production catalog.

NOSOURCE
NOSRC
specifies that the SCL source programs for PROGRAM entries are not copied.

REPLACE
specifies that entries from the catalog specified in the MERGE statement replace like-named entries in the current catalog. By default, existing entries in the current catalog are not replaced.

UPCASE
converts all text to uppercase during the merge.

Using the MERGE Statement

The MERGE statement merges entries from the specified catalog, sorted in alphabetical order by entry type. Unless you use the ENTRYTYPE=, EXCLUDE=, or SELECT= options, the merge operation attempts to copy all entries from the specified catalog to the current catalog (the catalog that was specified in the PROC BUILD statement). However, if any entries in the current catalog have the same names and types as entries in the specified catalog, then the existing entries in the current catalog are not replaced unless you use the REPLACE option.

MERGE Statement Examples

Example 1: Using the SELECT= and ENTRYTYPE= Options The following example copies only the entry AAA.HELP from the catalog OLD.CAT1 into the catalog NEW.CAT2:

```
proc build catalog=new.cat2;
   merge catalog=old.cat1 entrytype=help select=aaa;
run;
```

Example 2: Using the EXCLUDE= and ENTRYTYPE= Options The following example copies all the HELP entries except for AAA.HELP and BBB.HELP from the catalog OLD.CAT1 into the catalog NEW.CAT2:

```
proc build catalog=new.cat2;
   merge catalog=old.cat1 entrytype=help exclude=(aaa bbb);
run;
```

Example 3: Using the SELECT= Option without the ENTRYTYPE= Option The following example copies only the entries A.MENU and B.PROGRAM from the catalog OLD.CAT1 into the catalog NEW.CAT2:

```
proc build catalog=new.cat2;
   merge catalog=old.cat1 select=(a.menu b.program);
run;
```

Example 4: Using the REPLACE Option The following example copies all entries of type CBT from the catalog EX.FINAL into the catalog EX.COURSES. If a CBT entry in EX.COURSES has the same name as one that is being copied, the procedure replaces it with the copied entry.

```
proc build c=ex.courses;
   merge c=ex.final entrytype=cbt replace;
run;
```

MLINK Statement

Generates menu links for MENU entries that have the Menu-Link attribute in the catalog specified in the PROC BUILD statement.

Note: The menu-linking operation is performed before the procedure opens any windows.

Tip: By default, all MENU entries in the current catalog (the catalog that was specified in the PROC BUILD statement) are checked for selections that have the Menu-Link attribute. You can use the SELECT= option to select individual entries to check, or use the EXCLUDE= option to prevent certain entries from being checked.

MLINK <EXCLUDE=*entry-name* | (*entry-list*)> | <SELECT=*entry-name* | (*entry-list*)>
<LEVELS=*n* | _MAX_>
<VERBOSE>;

Options

You can use the following options in the MLINK statement:

EXCLUDE=*entry-name* | (*name-list*)
SELECT=*entry-name* | (*name-list*)
specify the MENU entries to exclude from or select for the linking operation. By default, all MENU entries in the current catalog are checked for selections that have the Menu-Link attribute. Use the EXCLUDE option to prevent specific MENU entries from being checked for links. Use the SELECT= option to check only the specified entries for links. If you specify more than one entry for either of these options, enclose the names in parentheses and separate the names with spaces.

LEVELS=*n* | _MAX_
specifies the number of levels of submenus to be linked. By default, only one level of submenu selections is linked. Use the LEVELS= option to specify a different number of levels. Use the _MAX_ option value to link all designated submenus.

VERBOSE
produces additional messages for each menu that has the Menu-Link attribute. These messages identify the name of the MENU entry from which selections are being linked, as well as the level number of the link. By default, messages for each MENU entry report only the start and completion of the MLINK operation, the name of MENU entries that contain no links, and any error conditions that are found.

Using the MLINK Statement

Menu links enable users to access submenu choices directly from an application's higher-level menu. Linked menus are useful to users who have become familiar with a system and who want to bypass intermediate menus to directly invoke choices on secondary menus. For example, suppose a MENU entry includes a selection **3** that opens another MENU entry, and that the second MENU entry includes a selection **SALES** that displays a sales report. If menu links are generated for the first MENU entry, then users can go directly to the sales report by specifying **SALES** in the first menu, without having to open the second menu.

Menu linking is applicable only to MENU entries. In order for a menu to be linked, the Menu-Link attribute must be assigned to it in the ATTR window of the calling

MENU entry. Each time you change menus or submenus, you should re-generate the links to ensure that all linked selections are available from the highest-level menu.

Note: You can also generate menu links by issuing the MLINK command while building the MENU entries. △

PRINT Statement

Prints the contents of entries from the catalog specified in the PROC BUILD statement.

Restriction: The PRINT statement can print the contents of CBT, HELP, LIST, MENU, PROGRAM, SCL, and SOURCE entries. It cannot print the contents of FRAME, CLASS, RANGE, and RESOURCE entries or of any other entry types that are not supported by the BUILD procedure.

Note: The print operation is performed before the procedure opens any windows.

Tip: By default, the procedure attempts to print all entries in the current catalog (the catalog that was specified in the PROC BUILD statement). You can use the SELECT= option to select individual entries to print, or use the EXCLUDE= option to prevent certain entries from being printed.

PRINT *items*

<ENTRYTYPE=*entry-type*>

<EXCLUDE=*entry-name.entry-type* | (*entry-list*)> | <SELECT=*entry-name.entry-type* | (*entry-list*) <XREF>>

<FORM=*form-name*>

<LEFT>

<LINENUM>

<NOPAGEBREAK>

<PRTFILE='*filename*' | *fileref* <APPEND>>;

where *items* must be one or more of the following:

ATTR

DISPLAY<SHOWPAD>

LISTDIR

SOURCE

Required Arguments

You must always supply at least one of the following arguments with the PRINT statement. Any combination of the following arguments can be specified.

ATTR
prints all of the attribute information that is associated with each entry.

DISPLAY <SHOWPAD | SP>
DISP <SHOWPAD | SP>
prints the display portion of each entry.

For PROGRAM entries, user fields in the display are by default represented by their respective field names. Use the SHOWPAD option in conjunction with the

DISPLAY option to represent user fields by their corresponding pad characters instead.

For LIST entries, the values in the list are printed.

LISTDIR
LD

prints a listing of the contents of the catalog specified in the PROC BUILD statement.

SOURCE
SRC

prints the SAS Component Language source code from each PROGRAM, SCL, or SOURCE entry.

Options

You can use the following options in the PRINT statement:

ENTRYTYPE=*entry-type*
ETYPE=*entry-type*
ET=*entry-type*

specifies the type of entry to print. By default, all CBT, HELP, LIST, MENU, PROGRAM, SCL, and SOURCE entries in the current catalog are printed (unless you also use the SELECT= or EXCLUDE= option). Use the ENTRYTYPE= option to print only entries of the specified type.

To print entries of more than one type, use a separate PRINT statement for each entry type.

EXCLUDE=*entry-name.entry-type* | (*entry-list*)
SELECT=*entry-name.entry-type* | (*entry-list*) <XREF>

specify catalog entries to exclude from or select for printing. By default, all CBT, HELP, LIST, MENU, PROGRAM, SCL, and SOURCE entries in the current catalog are printed (unless you also use the ENTRYTYPE= option). Use the EXCLUDE= option to prevent specified entries from being printed. Use the SELECT= option to print only specified entries. If you specify more than one entry for either of these options, enclose the list in parentheses and separate the names with spaces.

If you use the ENTRYTYPE= option in the same PRINT statement, you can omit the *entry-type* portion of the entry specification for the EXCLUDE= or SELECT= options because only entries of the type specified in the ENTRYTYPE= option can be printed.

When you use the SELECT= option to print CBT or MENU entries, you can also use the XREF option to print two cross-reference tables for each MENU or CBT entry. One table lists all entries that are called by that entry, and the other table lists all entries that call the entry.

Each PRINT statement can include only one EXCLUDE= option or one SELECT= option. However, you can use multiple PRINT statements with the same PROC BUILD statement.

FORM=*form-name*

specifies the name of a FORM entry to control the output generated by the PRINT statement. Specify either a one- or three-level name for *form-name*. A one-level name is assumed to be the name of a FORM entry in the current catalog. If the FORM entry is in a different catalog, use a three-level name (*libref.catalog-name.entry-name*). The default is the standard system form, SASHELP.FSP.DEFAULT.FORM.

LEFT

specifies that output produced by the PRINT statement is aligned at the left margin of the page or print file. By default, output is indented four spaces.

LINENUM

prints line numbers (up to 99999) at the beginning of each line of SCL source code.

Note: The LINENUM option is valid only when SOURCE is specified for the *items* argument . △

NOPAGEBREAK

prints entries without page breaks. By default, a page break is generated each time the number of lines per page defined by your FORMS entry is reached, and each page of the program listing has a page header. Blank lines are printed at the end of the program listing until the specified page length is reached. Use the NOPAGEBREAK option to print the program listing without page breaks. Headers are printed only at the top of the entry and every 32,768 lines thereafter.

Note: The NOPAGEBREAK option is valid only when SOURCE is specified for the *items* argument . △

PRTFILE='*filename*' | *fileref* <APPEND>

specifies a file to receive the output of the PRINT statement. By default, output is sent to the default printer or to the printer specified in the form identified in the FORM= option. Use the PRTFILE= option to redirect the output to a file instead. Specify the print file using either

- a fully-qualified filename enclosed in quotes
- a fileref that has been assigned to the file with a FILENAME statement or with an external allocation.

By default, if output is sent to a print file that already exists, the output overwrites the current contents of the file. Add the APPEND option to append printed output to the end of the file if it already exists.

Using the PRINT Statement

You can use the PRINT statement to generate hardcopy documentation of the displays, attributes, and source code in your applications. By default, output is sent to the default printer or to the printer specified in the FORM= option. You can use the PRTFILE= option to route output to a file instead.

PRINT Statement Examples

Example 1: Selecting Entries to Print The following example shows how to print the DISPLAY window views for two entries, GETFIELD.PROGRAM and GETTYPE.PROGRAM. Pad characters are printed to show the DISPLAY windows as users see them. Because the PRTFILE= option is not specified, the output is sent to the default printer. This example uses batch mode, which is the optimum way to print entries when you do not need to view the catalog or its entries.

```
proc build cat=mylib.mycat batch;
   print display showpad et=program
         select=(getfield gettype);
run;
```

Example 2: Excluding Entries from Printing The following example shows how to print the SCL source programs for all PROGRAM entries in the MYLIB.MYCAT catalog except GETFIELD.PROGRAM and GETTYPE.PROGRAM. Because the BATCH option is not specified, the Explorer window opens after the printing is performed so that you can work in the BUILD session.

```
proc build cat=mylib.newcat;
   print source etype=program
```

```
          exclude=(getfield gettype);
run;
```

Example 3: Printing a Cross Reference The following example prints a cross reference of entries that call or are called by the MAIN.MENU entry. Output is routed to the file identified by the fileref MENUREF, which you must have previously assigned.

```
proc build cat=mylib.newcat batch;
   print xref select=main.menu
         prtfile=menuref;
run;
```

SYNC Statement

Updates RESOURCE entries to incorporate changes made to one or more CLASS entries contained in the RESOURCE entries, or updates CLASS and FRAME entries to re-establish links from instance variables in the CLASS or FRAME entries to the parent classes.

Note: The synchronization operation is performed before the procedure opens any windows.

Tip: By default, all CLASS, FRAME, and RESOURCE entries in the current catalog (the catalog that was specified in the PROC BUILD statement) are synchronized. You can use the SELECT= option to select individual entries to synchronize, or use the EXCLUDE= option to prevent certain entries from being synchronized.

SYNC <ENTRYTYPE=*entry-type*>

<EXCLUDE=*entry-name.entry-type* | (*entry-list*)> | <SELECT=*entry-name.entry-type* | (*entry-list*) >

ENTRYTYPE=*entry-type*
ET=*entry-type*

specifies the type of entry to synchronize. By default, all CLASS, FRAME, and RESOURCE entries in the current catalog are synchronized (unless you also use the SELECT= or EXCLUDE= option). Use the ENTRYTYPE= option to synchronize only entries of the specified type.

To synchronize entries of more than one type, use a separate SYNC statement for each entry type.

EXCLUDE=*entry-name.entry-type* | (*entry-list*)
SELECT=*entry-name.entry-type* | (*entry-list*)

specify catalog entries to exclude from or select for synchronizing. By default, all CLASS, FRAME, and RESOURCE entries in the current catalog are synchronized (unless you also use the ENTRYTYPE= option). Use the EXCLUDE= option to prevent specified entries from being synchronized. Use the SELECT= option to synchronize only specified entries. If you specify more than one entry for either of these options, enclose the list in parentheses and separate the names with spaces.

If you use the ENTRYTYPE= option in the same SYNC statement, you can omit the *entry-type* portion of the entry specification for the EXCLUDE= or SELECT=

options because only entries of the type specified in the ENTRYTYPE= option can be synchronized.

Each SYNC statement can include only one EXCLUDE= option or one SELECT= option. However, you can use multiple SYNC statements with the same PROC BUILD statement.

Using the SYNC Statement

Objects and classes can link their instance variables to their parent classes, enabling you to change instance variables in a class. The change will also affect instances in subclasses of that class. However, it is possible to break the link to the parent class by changing a value to something other than the value specified in the parent, then changing it back. This gives you an instance variable that has the same value as the variable in the parent class. Synchronization restores this link by removing the duplicate item from the object attribute list (for FRAME entries) or from the class instance variables list (for CLASS and RESOURCE entries).

Note: You can also synchronize CLASS, FRAME, and RESOURCE entries by issuing the SYNC command in the BUILD procedure windows while editing the entries. △

BUILD Command Syntax

Tip: The BUILD command provides an easy way to open a BUILD window from any SAS System command line.

BUILD <<*libref.*>*catalog-name*<*.entry-name*<*.entry-type*>>>
<RESOURCE=<*libref.catalog-name.*>*resource-name*<.RESOURCE>>

BUILD Command

Initiates a BUILD session.

BUILD <<*libref.*>*catalog-name*<*.entry-name*<*.entry-type*>>>
<RESOURCE=<*libref.catalog-name.*>*resource-name*<.RESOURCE>>

Options

You can specify the following optional arguments with the BUILD command:

<libref.>catalog-name<.entry-name<.entry-type>>
specifies the catalog or catalog entry to open. Argument values are interpreted as follows:

- □ A one-level name (*catalog-name*) identifies a catalog in the default library, WORK. Remember that the contents of the WORK library are deleted when the SAS session ends.
- □ A two-level name (*libref.catalog-name*) identifies a catalog in a specified library.
- □ A three-level name (*libref.catalog-name.entry-name*) identifies a PROGRAM entry in a specified catalog.
- □ A four-level name (*libref.catalog-name.entry-name.entry-type*) identifies an entry of a specified type in a specified catalog.

The form of the argument determines which window the BUILD command opens, as follows:

- □ If you supply a one- or two-level name, an Explorer window opens to display the contents of the catalog. You can then select an existing entry in the catalog to edit, or you can create a new catalog entry.
- □ If you supply a three-level name, the DISPLAY window opens for editing the specified PROGRAM entry.
- □ If you supply a four-level name, the appropriate window opens for the specified entry type.

See "BUILD Procedure Windows" on page 19 for information about the corresponding window for each entry type. If you specify an entry name that does not already exist, a new entry of the specified type is created. Refer to Chapter 3, "SAS/AF Catalog Entry Types," on page 21 for information about the uses of the different catalog entry types.

If you issue a BUILD command with no catalog or entry name argument, a SAS Explorer window opens, from which you can select a library and catalog and then either select an existing catalog entry or create a new catalog entry.

RESOURCE=<*libref.catalog-name.*>*resource-name*<.RESOURCE>
specifies the RESOURCE entry that is associated with any FRAME entries you create during the BUILD session. By default, new FRAME entries use the RESOURCE entry that is specified in the SAS registry. You can use the BUILD= option to override the default for the current BUILD session. The *resource-name* value must be the name of an existing RESOURCE entry. If you omit the *libref.catalog-name* value, the procedure looks for the specified RESOURCE entry in the current catalog (the catalog that was specified in the BUILD command's *catalog-name* argument).

Note: You can use the RESOURCE command in a BUILD procedure window to change the default RESOURCE entry during the BUILD session. △

Using the BUILD Command

The BUILD command can be issued from any SAS window. Each BUILD command starts a separate BUILD session, so you can have several BUILD sessions running at the same time, each building different entries in the same or different catalogs. If you attempt to open an entry that is already open for editing in a different BUILD session, it is opened for browsing rather than editing.

BUILD Procedure Windows

When you specify a catalog entry to create, edit, or browse, the BUILD procedure opens the entry in the appropriate BUILD window for the entry type. The following table shows the corresponding window for each of the catalog entry types that the BUILD procedure supports:

Entry Type	**Window Opened**
CBT	DISPLAY
CLASS	Class Editor
EDPARMS	EDPARMS
FORM	FORM
FRAME	DISPLAY and Components
HELP	DISPLAY
INTRFACE	Interface Editor
KEYS	KEYS
LIST	LISTATTR and LISTVALUES
MENU	DISPLAY
PROGRAM	DISPLAY
RANGE	RANGE
RESOURCE	Resource Editor
SCL	SOURCE

For more information about the behavior of each of the BUILD procedure windows, including details of the commands you can use in the windows, refer to the online Help for the corresponding window. For information about the entry features that you can define in the BUILD procedure windows, see Chapter 3, "SAS/AF Catalog Entry Types," on page 21.

Note: The EDPARMS, FORM, and KEYS windows are SAS windowing environment windows that the BUILD procedure opens as a convenience for application developers. Refer to base SAS documentation for information about the EDPARMS, FORM, and KEYS windows. △

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/AF® Software Procedure Guide, Version 8*, Cary, NC: SAS Institute Inc., 1999.

SAS/AF® Software Procedure Guide, Version 8

ISBN 1–58025–516–7

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

1st printing, October 1999