Syntax Conventions for SAS® Programming Languages

SAS Syntax Components

Keywords

A keyword is one or more literal name components of a language element. Keywords are uppercase, and in reference documentation, they are bold. In other documents, such as user guides, keywords are uppercase only.

Single Keywords

CHAR (*string, position*)
ALTER (*alter-password*)
BEST *w*.
REMOVE <*data-set-name*>

Multiple Keywords in CALL Routines

CALL routines have two keywords, CALL and the routine name:

CALL RANBIN (*seed*, *n*, *p*, *x*)

Keywords with No Arguments

DO;
  ... *SAS code* ...
END;

Multiple Keywords in Procedure Statements

Some procedure statements have multiple keywords throughout the statement syntax:

CREATE <UNIQUE> **INDEX** *index-name* **ON** *table-name* (*column-1* <, *column-2*, …>)

Choice between Multiple Keywords

Some system options require that you choose between one of two keywords:

DUPLEX | **NODUPLEX**

Arguments

An argument follows a keyword or is on the right side of an assignment statement. Arguments specify a numeric or character constant, variable, or expression that is used by SAS to process the language element. In most cases, arguments appear in italic. Arguments can be required or optional. In syntax, optional arguments are enclosed in angle brackets (< >).

Required Arguments

Required arguments are not enclosed in angle brackets (< >).

CHAR (*string, position*)

In this example, the CHAR function has two required arguments, *string* and *position*.

In this code example, the value 'summer' is the *string* argument and the value 4 is the *position* argument:

```
x=char('summer', 4);
```

Optional Arguments

Optional arguments are enclosed in angle brackets (< >).

COMPRESS (*source* <, *characters*> <, *modifier(s)*>);

In this example, *source* is a required argument, whereas *characters* and *modifier(s)* are optional arguments.

In this code example, the argument string is the source, "12345" are the characters, and "d" is the modifier:

```
compress(string, "12345", "d");
```

Argument Lists

In syntax, *argument(s)* specifies that one argument is required and that multiple arguments are allowed. Use a space to separate arguments. Punctuation, such as a comma (,) is not required between arguments.

The MISSING statement is an example.

MISSING *character(s)*;

```
data survey;
   missing a r;
   input id answer;
   datalines;
001 2
002 R
003 1
004 A
```

```
005 2
;
```

Another type of argument list is *argument-1* <, *argument-2*, ...>. For example:

AUTHPROVIDERDOMAIN (*provider-1:domain-1* <, *provider-2:domain-2*, …>
INTO :*macro-variable-specification-1* <, :*macro-variable-specification-2*, …>

Keyword-Argument Pairs

An argument pair associates a keyword with an argument. You can specify multiple keyword-argument pairs. No punctuation is required between keyword-argument pairs. The ellipsis (...) indicates that additional keyword-argument pairs are allowed. For example:

BY <DESCENDING> *variable-1* <<DESCENDING> *variable-2* …>;

In this syntax example, DESCENDING is the keyword and *variable-x* is the argument.

The BY statement can be specified in several procedures. In this SORT procedure, the data set status.qtr4 is sorted in descending order by the value of the variable Week:

```
proc sort data=status.qtr4;
  by descending week;
run;
```

Argument-Option Pairs

You can associate some arguments with options. No punctuation is required between the argument and the option. The ellipsis (...) indicates that additional arguments with an associated option are allowed:

argument-1 <*option(s)*> <*argument-2* <*option(s)*> ...>

The FORMAT procedure PICTURE statement is an example of an argument-option pair. For each *value-range-set* that you specify, you can provide *picture-option(s)*:

PICTURE *name* <(*format-option(s)*)>
<*value-range-set-1* <(*picture-1-option(s)*)>
<*value-range-set-2* <(*picture-2-option(s)*)> …>>;

In this code example, the first value-range-set 1E06–1000000000='0000 M' has two options, prefix='$' and mult=.000001:

```
proc format;
   picture bigmoney (fuzz=0)
      1E06-<1000000000='0000 M' (prefix='$' mult=.000001)
      1E09-<1000000000000='0000 B' (prefix='$' mult=1E-09)
      1E12-<1000000000000000='0000 T' (prefix='$' mult=1E-012);
```

Argument Assignments

Some syntax allows one or more arguments to be assigned a value. No punctuation is required between arguments. The ellipsis (...) indicates that additional arguments are allowed:

argument-1=value-1 <argument-2=value-2 ...>

The LABEL statement is an example of multiple arguments:

LABEL *variable-1=label-1 <variable-2=label-2 …>*;

```
label score1="Grade on April 1 Test" score2="Grade on May 1 Test";
```

Styles and Special Characters

Style

The style conventions that are used in documenting SAS syntax include uppercase bold, uppercase, and italic.

UPPERCASE BOLD
identifies SAS keywords such as the names of functions or statements. In this example, the keyword ERROR is written in uppercase bold:

ERROR <*message*>;

UPPERCASE
identifies arguments that are literals. In this example of the CMPMODEL= system option, the literals are BOTH, CATALOG, and XML:

CMPMODEL=BOTH | CATALOG | XML |

italic
identifies arguments or values that you supply. Items in italic represent user-supplied values that are either nonliteral arguments or nonliteral values:

- nonliteral arguments. In this example of the LINK statement, the argument *label* is a user-supplied value and therefore appears in italic:

  LINK *label*;

- nonliteral values that are assigned to an argument. In this example of the FORMAT statement, the argument DEFAULT is assigned the variable *default-format*:

  FORMAT *variable(s)* <*format* > <DEFAULT = *default-format*>;

Special Characters

The syntax of SAS language elements can contain these special characters:

=
An equal sign identifies a value for a literal in some language elements such as system options.

In this example of the MAPS system option, the equal sign sets the value of MAPS:

MAPS=*location-of-maps*

< >
Angle brackets identify optional arguments. A required argument is not enclosed in angle brackets.

In this example of the CAT function, at least one item is required:

CAT (*item-1* <, *item-2*, …>)

|
A vertical bar indicates that you can choose one value from a group of values. Values that are separated by the vertical bar are mutually exclusive.

In this example of the CMPMODEL= system option, you can choose only one of the literal arguments:

CMPMODEL=BOTH | CATALOG | XML

...
An ellipsis indicates that an argument can be repeated. If the argument and the ellipsis are enclosed in angle brackets, then the argument is optional. The repeated argument must contain punctuation if it appears before or after the initial argument.

In this example of the CAT function, multiple *item* arguments are allowed, and they must be separated by a comma:

CAT (*item-1* <*, item-2*, …>)

'*value*' or "*value*"
Single or double quotation marks that enclose an argument indicate that the argument must have a value that is also enclosed in single or double quotation marks.

In this example of the FOOTNOTE statement, the argument *text* is enclosed in quotation marks:

FOOTNOTE <*n*> <*ods-format-options* '*text*' | "*text*">;

;
A semicolon indicates the end of a statement or CALL routine.

In this example, each statement ends with a semicolon:

```
data namegame;
   length color name $8;
   color = 'black';
   name = 'jack';
   game = trim(color) || name;
run;
```

References to SAS Libraries and External Files

Many SAS statements and other language elements refer to SAS libraries and external files. You can make the reference through a logical name (a libref or fileref), or you can use the physical filename enclosed in quotation marks. If you use a logical name to make the reference, you typically use a SAS statement (LIBNAME or FILENAME) or the operating environment's control language. Several methods of referring to SAS libraries and external files are available, and some of these methods depend on your operating environment.

In the examples that use SAS libraries, SAS documentation uses the italicized phrase *SAS-library* enclosed in quotation marks. In the examples that use external files, SAS documentation uses the italicized phrase *file-specification:*

```
infile file-specification obs = 100;
libname libref 'SAS-library';
```