

# Macros

## What are the advantages of using nested macro program definitions?

Macro variables created in 'open code' i.e. outwith a macro program definition are stored on the GLOBAL Symbol Table, and as such are globally available in the current session: values persist until changed, deleted or the session is ended. This can result in inadvertent contamination of macro variable values from previously-run code.

Macro variables created 'within' a macro program definition (e.g. by using parameters) are stored on a LOCAL Symbol Table, can only be referenced within the macro program, and are automatically deleted when the macro program ceases execution.

A nested macro program can reference macro variables on each successive nested Symbol Table according to hierarchical rules.

The code:

```
%macro reportit ( helpme
                    ,dsn          = &syslast
                    ,repvars      = _numeric_
                    ,classvars    = _character_
                    ,rep_type     = LIST
                    ) ;
%if &helpme = ? or %upcase(&helpme) = HELP %then
%do ;
    %put NOTE: The parameters for this macro are: ;
%put NOTE- HELPM     Passing the value ? or HELP will generate these
NOTES. ;
%put NOTE- DSN=       Specify the dataset to be reported. ;
    %put NOTE-       If omitted, the last created dataset will
be used. ;
    %put NOTE- REPVAR= Specify in a space-
separated list the analysis variable(s). ;
    %put NOTE-       If omitted, all NUMERIC variables will be proce
ssed. ;
    %put NOTE- CLASSVAR= Specify in a space-
separated list the classification variable(s). ;
    %put NOTE-       If omitted, all CHARACTER variables will
be used. ;
%goto endmac ;
%end ;

%macro listit ;
    %put *** LISTIT LOCAL MACRO VARIABLES *** ;
    %put _local_ ;

proc print data = &dsn ;
    var &repvars ;
    by &classvars ;
```

# Macros

```
run ;
%mend listit ;

%macro submit ;
  %put *** SUMMIT LOCAL MACRO VARIABLES *** ;
  %put _local_ ;

  proc means data = &dsn sum mean ;
    var &repvars ;
    class &classvars ;
  run ;
%mend submit ;

%if %upcase(&rep_type) = LIST %then %listit ;
  %else %if %upcase(&rep_type) = SUMMARY %then %submit ;
%else %put ERROR: Valid rep_type options are LISTING or SUMMARY. ;

%put *** REPORTIT LOCAL MACRO VARIABLES *** ;
%put _local_ ;
%endmac ;
%mend reportit ;
```

contains a 'parent' macro REPORTIT and two nested 'child' macros LISTIT and SUMMIT. The 'child' macro programs have no parameters, therefore no LOCAL symbol table is created for these, however the macro parameters for REPORTIT exist on a LOCAL Symbol Table and can be accessed by each of these nested macros.

Submitting a series of macro program calls to the REPORTIT macro using the MPRINT option to display the generated code, and the MPRINTNEST option to indicate the hierarchy of the nested macro programs:

```
options mprint mprintnest ;

%reportit(?)

%reportit(dsn = sashelp.class, repvars = _all_, classvars = )

%reportit(dsn = sashelp.class, classvars = sex, rep_type = SUMMARY)
```

Generated LOG messages are as follows:

```
51  %reportit(?)
NOTE: The parameters for this macro are:
```

# Macros

HELPME        Passing the value ? or HELP will generate these NOTES

.

DSN=            Specify the dataset to be reported.  
                 If omitted, the last created dataset will be used.

REPVAR=        Specify in a space-separated list the analysis variable(s).  
                 If omitted, all NUMERIC variables will be processed.

CLASSVAR=      Specify in a space-separated list the classification variable(s).  
                 If omitted, all CHARACTER variables will be used.

```
53  %reportit(dsn = sashelp.class, repvars = _all_, classvars = )
*** LISTIT LOCAL MACRO VARIABLES ***
MPRINT(REPORTIT.LISTIT):  proc print data = sashelp.class ;
MPRINT(REPORTIT.LISTIT):  var _all_ ;
MPRINT(REPORTIT.LISTIT):  by ;
MPRINT(REPORTIT.LISTIT):  run ;
```

NOTE: There were 19 observations read from the data set SASHELP.CLASS.

NOTE: PROCEDURE PRINT used (Total process time):

real time	0.01 seconds
cpu time	0.01 seconds

```
*** REPORTIT LOCAL MACRO VARIABLES ***
REPORTIT DSN sashelp.class
REPORTIT HELPME
REPORTIT CLASSVAR
REPORTIT REP_TYPE LIST
REPORTIT REPVAR _all_
```

Note that there are no LOCAL macro variables for LISTIT, but values have been accessed from the REPORTIT Symbol Table. The MPRINT messages also show the MPRINT(REPORTIT.LISTIT) hierarchy.

Similarly for the SUMMIT macro program call:

```
55  %reportit(dsn = sashelp.class, classvars = sex, rep_type = SUMMARY)
*** SUMMIT LOCAL MACRO VARIABLES ***
MPRINT(REPORTIT.SUMMIT):  proc means data = sashelp.class sum mean ;
```

# Macros

```
MPRINT(REPORTIT.SUMMIT):    var _numeric_ ;  
MPRINT(REPORTIT.SUMMIT):    class sex ;  
MPRINT(REPORTIT.SUMMIT):    run ;
```

NOTE: There were 19 observations read from the data set SASHELP.CLASS.

NOTE: PROCEDURE MEANS used (Total process time):

real time	0.00 seconds
cpu time	0.00 seconds

\*\*\* REPORTIT LOCAL MACRO VARIABLES \*\*\*

```
REPORTIT DSN sashelp.class  
REPORTIT HELPME  
REPORTIT CLASSVARS sex  
REPORTIT REP_TYPE SUMMARY  
REPORTIT REPVARS _numeric_
```

Unique solution ID: #1022

Author: Alan D Rudland

Last update: 2017-04-12 12:28