

# BUILDING AND USING MACRO VARIABLE LISTS

Clark Roberts, Decision Analytics

## ABSTRACT

It is often necessary, in writing programs, to process the same logic over a set of values. Everyone is familiar with the purpose of ARRAYS and DO loops, to apply similar logic to several different variables without repeating the code for each one. The SAS® System provides several varieties of DO loop syntax, one in particular, the DO iterative syntax has several flavors, including DO *index* = list, which is especially useful if the values of the *index* variable are not contiguous. Unfortunately, this syntax is limited to data step processing and there isn't any equivalent to this in the macro facility. This paper will discuss how to use the SAS Macro facility to emulate the DO list statement for processing a list of character variables that can provide an application with dynamic data driven capability. It will also discuss different ways that the macro variable lists can be constructed. Several examples will be presented to demonstrate the creation of the lists, including examples that use information from the SAS Help Dictionary views, and applications that use these lists to control process flow.

## USING MACRO VARIABLE LISTS

The heart of the approach is the use of the %DO %WHILE construct combined with the %SCAN function. The basic syntax for the loop is:

```
%let i = 1 ;
%do %while(%scan(&macList,&I,%str( )) ^= %str( )) ;
  %let var = %scan(&macList,&i,%str( )) ;
  . . . . .
  more SAS code
  . . . . .
  %let i = %eval(&i + 1 ) ;
%end ;
```

The ALLCAPS.SAS macro (figure 1) in the examples section demonstrates the use a macro variable list named CVARLIST in a %DO %WHILE loop to capitalize all the variables in a given SAS data set. The CVARLIST macro variable is built using the GETCVAR macro (figure 1) discussed in the following section.

## BUILDING MACRO VARIABLE LISTS

There are several ways to build the macro variable list that can be used in the above loop structure:

1. Use a %LET statement to assign values to the list.
2. Assign values in a parameter of the macro.
3. Read the values from a file and dynamically build the list.

The third option provides the most flexibility and allows dynamic execution of the macro. An example of this is the GETCVAR.SAS (figure 2) in the examples below which accesses the SAS Dictionary Views in the SASHELP library to extract the names of all character variables, insert the names into separate local macro variables, and then create a global macro variable named CVARLIST which contains all the names in a space delimited list format.

## CONCLUSIONS

This ALLCAPS.SAS example could have been written more concisely using the CALL EXECUTE statement as demonstrated in the uncommented ALLCAPS2.SAS macro (figure 3). However, I tend to build programs from existing modules that have been previously tested, as much as possible. This approach can save a lot of time when developing applications. An exception to this is when performance is an issue and the existing macros are not providing adequate efficiency. Another reason this example was used was to keep within the space limitations of the proceedings.

The use of macro variable lists is more efficient than using CALL EXECUTE when the list will be used in several places within an application program. Since these types of applications tend to be lengthy, they were excluded from the examples. It should be a simple matter, however to extrapolate the ALLCAPS.SAS example to more complex applications. Additional examples can also be obtained by contacting the author.

## ACKNOWLEDGMENTS

SAS is a registered trademark of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

## CONTACT INFORMATION

For questions or further information the author can be contacted at the following address:

Clark Roberts, Principal Consultant  
Decision Analytics, A SAS Quality Partner  
5663 Balboa Avenue, Suite 400  
San Diego, California 92111

or by

Phone: (619) 565-9627 or (619) 565-9998  
Fax: (619) 565-9627  
Voice Mail/Pager: (619) 975-0758  
e-mail: dacmr@mindspring.com

Code for the examples in this paper and related programs can be downloaded from the SAS File Contribution Server FTP site at:

<ftp://ftp.uga.edu/pub/sas/contrib>

or by contacting the author.

## EXAMPLES

### FIGURE 1 - ALLCAPS.SAS

```
*****
****
* FACILITY:
*          DECISION ANALYTICS
*
* SYSTEM NAME:
```

```

* COMMON SAS TOOLS LIBRARY
*
* PROGRAM:
* ALLCAPS.SAS
*
* LANGUAGE/VERSION:
* SAS 6.08
*
* DESCRIPTION:
*
* THIS MACRO CONVERTS ALL CHARACTER VARIABLES IN THE
* &inlib.&inds DATA SET TO UPPER CASE AND WRITES THE
* CONVERTED RECORDS TO THE &outlib.&outds DATA SET.
* THE NAMES OF THE CHARACTER VARIABLES ARE OBTAINED
* BY A CALL TO THE %getvar MACRO. IF &inlib IS BLANK,
* THEN THE DATA SET IS READ FROM THE SAS WORK LIBRARY. IF
* &outlib OR &outds IS BLANK, THEY ARE SET TO THEIR
* RESPECTIVE &in COUNTERPARTS. IF THE SASHELP LIBRARY
* HAS NOT BEEN ALLOCATED, OR THE SPECIFIED SAS DATA SET
* IS EMPTY OR DOES NOT EXIST, THEN THE GLOBAL MACRO VARIABLE
* &ncvars WILL RETURN A VALUE OF 0 (zero)
*
* CALLED BY:
* [various programs]
*
* MACROS CALLED:
* GETCVAR -
* RETURNS THE NAMES OF ALL CHARACTER VARIABLES IN A
* SPECIFIED SAS DATA SET IN A MACROVARIABLE LIST
*
* FILES READ:
* [none]
*
* DATA SETS READ:
* &inlib.&inds
*
* FILES CREATED:
* [none]
*
* DATA SETS CREATED:
* &outlib.&outds
*
* FILES INCLUDED:
* GETCVAR.SAS -
* Refer to definition in MACROS CALLED section above
*
* MACRO VARIABLES:
* [passed from calling program]
*
* &inlib THE LIBRARY WHERE THE INPUT DATA SET RESIDES
*
* &inds THE SAS DATA SET CONTAINING THE CHARACTER
* VARIABLES TO BE CONVERTED TO UPPERCASE
*
* &outlib THE LIBRARY WHERE THE RESULTING DATA SET
* WILL BE WRITTEN TO. IF BLANK THEN
* THE LIBRARY IN &inlib WILL BE ASSUMED
*
* &outds THE OUTPUT DATA SET WHERE THE CONVERTED
* DATA WILL BE WRITTEN. IF BLANK, THEN THE
* DATA SET SPECIFIED IN &inds WILL BE ASSUMED
*
* [internal]
*
* &i USED AS AN INDEX FOR LOOPING
*
* [globals used]
*
* &ncvars THE NUMBER OF CHARACTER VARIABLES IN THE
* INPUT DATA SET
*
* &cvarlist A LIST OF THE NAMES OF THE CHARACTER
* VARIABLES
* IN THE INPUT DATA SET
*
*****
* REVISION HISTORY:
*
* V01.001 CLARK ROBERTS 13-FEB-
*
* INITIAL VERSION
*
*****
*
* INCLUDE REQUIRED FILES
*
%include 'getvar.sas';
*
%macro allcaps(inlib = work,
inds = ;
outlib = ;
outds = ;
)
;
%local i;
%global ncvars;
*****
*** TEST THE PASSED PARAMETERS AND ASSIGN DEFAULTS IF
*** NECESSARY. IF &inds IS BLANK, THEN SET NCVARS TO 0
*** AND GENERATE A WARNING MESSAGE ON THE SAS LOG
***
*****
%if &inds ^= %str() %then %do;
%if &inlib = %str() %then %let inlib = work;
%if &outlib = %str() %then %let outlib = &inlib;
%if &outds = %str() %then %let outds = &inds;
*****
*** CALL THE %getvar MACRO TO OBTAIN THE NUMBER OF
*** CHARACTER VARIABLES IN THE &inlib SAS DATA SET AND A
*** LIST OF THESE VARIABLES.
*****
%getvar(lib=&inlib,ds=&inds);
*****
*** USE THE RETURNED LIST TO CAPITALIZE ALL CHARACTER
*** &inlib SAS DATA SET AND
*****
%let i = 1;
data &outlib.&outds;
set &inlib.&inds;
%do %while(%scan(&cvarlist,&i,%str( )) ^= %str());
%scan(&cvarlist,&i,%str( )) =
upcase(%scan(&cvarlist,&i,%str( )));
%let i = %eval(&i + 1);
%end;
%let ncvars = %eval(&i - 1);
run;
%end;
%else %let ncvars = %str(-1);
*****
*** GENERATE A WARNING MESSAGE ON THE SAS LOG IF ANY
*** PROBLEMS OCCUR
***
*****
%if &ncvars <= 0 %then %do;
data _null_;
put // ;
put @1 '*****';
put @4 'WARNING: Character Variables were not
put @4 ' converted to uppercase. Either
put @4 ' the SASHELP library was not
put @4 ' allocated, or the specified SAS
put @4 ' data set is empty or missing.
put @4 '
put @4 ' (ref: ALLCAPS.SAS)
put @1 '*****';
put // ;
run;
%end;
%end allcaps;
*****
*** END OF PROGRAM: ALLCAPS.SAS
*****

```

;;;;;

## FIGURE 2 - GETCVAR.SAS

```

*****
* FACILITY:
*   DECISION ANALYTICS
*
* SYSTEM NAME:
*   COMMON SAS TOOLS LIBRARY
*
* PROGRAM:
*   GETCVAR.SAS
*
* LANGUAGE/VERSION:
*   SAS 6.08
*
* DESCRIPTION:
*
*   THIS MACRO USES THE VCOLUMN SAS DATA DICTIONARY VIEW
*   IN THE SASHELP LIBRARY TO EXTRACT THE NAMES OF ALL THE
*   CHARACTER VARIABLES IN THE SAS DATA SET GIVEN BY THE
*   &lib and &ds PARAMETERS PASSED TO GETCVAR. A GLOBAL
*   MACRO VARIABLE IS CREATED AS OUTPUT FROM THE PROCESS:
*   A SPACE DELIMITED LIST CALLED &cvarlist IS POPULATED
*   WITH THE NAMES OF THE CHARACTER VARIABLES.
*
* CALLED BY:
*   [various programs]
*
* MACROS CALLED:
*   [none]
*
* FILES READ:
*   [none]
*
* DATA SETS (SAS VIEWS) READ:
*   SASHELP.VCOLUMN
*   A SAS SQL VIEW THAT CONTAINS INFORMATION DOWN TO THE
*   VARIABLE LEVEL FOR EACH DATA SET IN EVERY SAS DATA LIBRARY
*   THAT IS CURRENTLY DEFINED WITH A LIBNAME
*
* FILES CREATED:
*   [none]
*
* DATA SETS CREATED:
*   [none]
*
* FILES INCLUDED:
*   [none]
*
* MACRO VARIABLES:
*   [passed from calling program]
*
*   &lib          THE NAME OF THE DATA LIBRARY THAT CONTAINS
*                 THE SAS DATA SET TO BE QUERIED
*
*   &ds           THE NAME OF THE SAS DATA SET WHERE THE NAMES
*                 OF THE CHARACTER VARIABLES WILL BE EXTRACTED
*                 FROM
*
*   [internal]
*
*   &i            USED AS A LOOPING INDEX
*
*   &ncvars       NUMBER OF CHARACTER VARIABLES IN THE DATA SET
*
*   [globals created]
*
*   &cvarlist     A SPACE DELIMITED LIST CONTAINING THE NAMES
*                 OF THE CHARACTER VARIABLES IN THE DATA SET
*
* REVISION HISTORY:
*
*   V01.001 CLARK ROBERTS          13-FEB-
1995
*
*   INITIAL VERSION
*****
*
*
*macro getovar(lib      = work,
              ds        =
              )

```

```

;
%global cvarlist;

%local i
      ncvars
      ;

%let ncvars = 0;
%let cvarlist = ;

*****
***   EXTRACT THE NAMES OF THE CHARACTER VARIABLES IN THE
***
***   &lib..&ds DATA SET FROM THE VCOLUMN SAS DATA DICTIONARY
***
***   VIEW IN THE SASHELP LIBRARY.
***
*****
;

data _null_;
  set sashelp.vcolumn(where=(libname = '%upcase(&lib)' and
                             memname = '%upcase(&ds)' and
                             upcase(memtype) = 'DATA' and
                             upcase(type) = 'C'
                             )
                    ) end = eof;
  retain n 0;
  n + 1;
  call symput('cvar'||left(put(n,4.0)),name);
  if eof then call symput('ncvars',put(n,4.0));
run;

*****
***   POPULATE THE GLOBAL MACRO VARIABLE &cvarlist WITH THE
***
***   NAMES OF THE CHARACTER VARIABLES EXTRACTED IN THE
***
***   PREVIOUS STEP. DELIMIT THE ENTRIES WITH ONE BLANK SPACE
***
*****
;

%do i = 1 %to &ncvars;
  %let cvarlist = &cvarlist&str( )&&cvar&i;
%end;

%mend getovar;

*****
***   END OF PROGRAM: GETCVAR.SAS
*****
;

```

## FIGURE 3 - ALLCAPS2.SAS

```

%macro allcaps2(inlib    = work ,
               inds      = ,
               outlib    = ,
               outds     =
               )
;

%if &inlib = %then %let inlib = work;
%if &inds = %then %let inds = temp;
%if &outlib = %then %let outlib = &inlib;
%if &outds = %then %let outds = &inds;

data _null_;
  set sashelp.vcolumn(where=(libname = '%upcase(&inlib)' and
                             memname = '%upcase(&inds)' and
                             upcase(memtype) = 'DATA' and
                             upcase(type) = 'C'
                             )
                    ) end = eof;
  ;
  if _n_ = 1 then
    call execute('data &outlib..&outds; set &inlib..&inds;');
  call execute(name||' = upcase('||name||'););
  if eof then call execute('run;');
run;

%mend allcaps2;

```