

Homemade SCL List Functions

Melissa J. Kerner, The Lubrizol Corporation, Wickliffe, Ohio

Abstract

Lists are a robust feature of Screen Control Language. They are easy to manipulate, yet they are capable of storing a mix of data types in a vertical format. "Grouped" lists are conceptually similar to flipping SAS data set observations vertically and stacking them. Each observation becomes a group of items in the list.

This paper presents ten functions written to work with grouped lists which include:

- loading data into a list from a SAS data set, and the converse
- locating an item's position
- listing unique item names
- merging two lists
- adding and deleting items
- subsetting a list.

Introduction

At Lubrizol, we use SCL lists extensively within SAS/AF screens to display data and options, to collect responses from end-users, and to control processing. When lists were first introduced, we had just started developing a new interactive application. We used extended tables for: listing features, providing tabular data entry, and displaying selection lists. We stored meta data in SAS data sets about the data displayed on the screens and about how the screens flowed for different options.

With data being transferred between SAS data sets, lists, and extended tables, the need to create functions to perform these operations became very apparent. Such functions were developed as stand alone SCL methods that accept and return parameter values. They are stored in a SAS catalog named UTILITY. This catalog is kept in a SAS library that is referenced by the libname SCREENS in

all of our applications. This scheme allows us to use these methods in any of our systems by issuing a command like: CALL METHOD ('SCREENS.UTILITY.method.SCL',...).

Nevertheless, the examples shown in this paper omit the SAS library and catalog names when calling these methods.

The code written for these methods is provided in the Appendix.

Load a Data Set into a List

Use the DS2LIST method to load data from data set ROCKDATA into a new list identified as ROCKHALL.

Data set: ROCKDATA

ARTIST	SONG	INDUCTED
THE BEATLES	HEY JUDE	1988
CHUCK BERRY	JOHNNY B. GOOD	1986
JAMES BROWN	I GOT YOU	1986
THE COASTERS	YAKETY YAK	1987
ELTON JOHN	YOUR SONG	1994
ELVIS PRESLEY	JAILHOUSE ROCK	1986

```
CALL METHOD('DS2LIST.SCL','INIT',  
           'ROCKDATA',ROCKHALL);
```

The list returned is:

```
ROCKHALL(ARTIST='THE BEATLES'  
         SONG='HEY JUDE'  
         INDUCTED=1988  
         ARTIST='CHUCK BERRY'  
         SONG='JOHNNY B. GOOD'  
         INDUCTED=1986  
         ARTIST='JAMES BROWN'  
         SONG='I GOT YOU'  
         INDUCTED=1986  
         ARTIST='THE COASTERS'  
         SONG='YAKETY YAK'  
         INDUCTED=1987  
         ARTIST='ELTON JOHN'  
         SONG='YOUR SONG'  
         INDUCTED=1994  
         ARTIST='ELVIS PRESLEY'  
         SONG='JAILHOUSE ROCK'  
         INDUCTED=1986  
)
```

To only load data for artists inducted in 1986, pass a value to the optional "where clause" parameter of method DS2LIST.

```
CALL METHOD('DS2LIST.SCL','INIT',
           'ROCKDATA',IN1986,
           'INDUCTED=1986');
```

The list returned is:

```
IN1986(ARTIST='CHUCK BERRY'
       SONG='JOHNNY B. GOOD'
       INDUCTED=1986
       ARTIST='JAMES BROWN'
       SONG='I GOT YOU'
       INDUCTED=1986
       ARTIST='ELVIS PRESLEY'
       SONG='JAILHOUSE ROCK'
       INDUCTED=1986
       )
```

NOTE: When a character item is specified in the "where clause" parameter, enclose the value in quotes too. (e.g.: 'ARTIST="CHUCK BERRY"')

To only load data for variables ARTIST and SONG, use the VAR2LIST method.

```
VARs='ARTIST,SONG';
CALL METHOD('VAR2LIST.SCL','INIT',
           'ROCKDATA',VARs,
           SONGLIST);
```

The list returned is:

```
SONGLIST(ARTIST='THE BEATLES'
         SONG='HEY JUDE'
         ARTIST='CHUCK BERRY'
         SONG='JOHNNY B. GOOD'
         ARTIST='JAMES BROWN'
         SONG='I GOT YOU'
         ARTIST='THE COASTERS'
         SONG='YAKETY YAK'
         ARTIST='ELTON JOHN'
         SONG='YOUR SONG'
         ARTIST='ELVIS PRESLEY'
         SONG='JAILHOUSE ROCK'
         )
```

Get the Number of Groups and Items in a List

Use the LISTINFO method to find out how many groups are in a list and how many items are in each group.

```
CALL METHOD('LISTINFO.SCL','INIT',
           ROCKHALL,GROUPS,ITEMS);
```

The values returned are: GROUPS=6
ITEMS=3

Merge Two Lists

To illustrate merging two lists, picture this scenario:

A SAS/AF screen displays a list of Rock Hall of Fame artists. It prompts end-users to select the artists they want to know more about.

CHOOSE SOME ARTISTS:

```
X   THE BEATLES
—   CHUCK BERRY
X   JAMES BROWN
—   THE COASTERS
—   ELTON JOHN
—   ELVIS PRESLEY
```

Suppose the screen contains SCL code that captures the selected artists and stores them in a list like this:

```
ARTISTS(ARTIST='THE BEATLES'
        ARTIST='JAMES BROWN'
        )
```

Now, method LMERGE can be used to merge lists ARTISTS and ROCKHALL by matching on the character item ARTIST. This method will produce a list showing all data from list ROCKHALL for each artist in list ARTISTS.

```
CALL METHOD('LMERGE.SCL','INIT',
           ARTISTS,ROCKHALL,
           SELECTED);
```

The list returned is:

```
SELECTED(ARTIST='THE BEATLES'
         SONG='HEY JUDE'
         INDUCTED=1988
         ARTIST='JAMES BROWN'
         SONG='I GOT YOU'
         INDUCTED=1986
         )
```

Create a List of Item Names

Imagine an application designed to retrieve information about the Rock Hall of Fame inductees. It may contain a screen such as this:

CHOOSE A TOPIC:

— ARTIST
— SONG
— INDUCTED

This screen could be an extended table that is populated by a list containing topic values.

The list of topics shown above is simply all of the item names in the list ROCKHALL. Use method ITEMLIST to create a unique list of item names.

```
CALL METHOD('ITEMLIST.SCL','INIT',  
           ROCKHALL, TOPICS);
```

The list returned is:

```
TOPICS('ARTIST'  
       'SONG'  
       'INDUCTED'  
       )
```

Locate a Specific Item and Value in a List

Each item in a list can be referenced by its index number. Knowing an item's physical position in a list can be useful. Many of the SCL list functions accept values for an index.

Call the FINDITEM method to get the index number of the item SONG = 'YAKETY YAK' in the list ROCKHALL.

```
CALL METHOD('FINDITEM.SCL',  
           'INITC', ROCKHALL, 'SONG',  
           'YAKETY YAK', POSITION);
```

11 is returned to the variable POSITION to show that SONG = 'YAKETY YAK' is the 11th item in the list.

NOTE: Label INITC is used because SONG is a character item.

Subset a Group from a List

To get all group data associated with an item, use method SUBGROUP to subset the group of items.

This example gets all data related to item 'SONG = JAILHOUSE ROCK' and returns the data as list JAILROCK.

```
CALL METHOD('SUBGROUP.SCL',  
           'INITC', ROCKHALL, 'SONG',  
           'JAILHOUSE ROCK', JAILROCK);
```

The list returned is:

```
JAILROCK(ARTIST='ELVIS PRESLEY'  
         SONG='JAILHOUSE ROCK'  
         INDUCTED=1986  
         )
```

Add Item(s) to a List

Suppose a screen is created to collect the number of votes received for each song from the 1986 inductees. If a numeric item named VOTES is added to each group in list IN1986, then these numbers could be recorded in that list.

ENTER VOTES FOR EACH SONG:

50	JOHNNY B. GOOD
100	I GOT YOU
75	JAILHOUSE ROCK

By calling method ADDITEM, VOTES can be added to each group in list IN1986. This new item could then be populated with the entered values using the SCL function SETNITEMN.

```
CALL METHOD('ADDITEM.SCL','INIT',  
           IN1986, 'VOTES', 'N');
```

```
IN1986(ARTIST='CHUCK BERRY'  
       SONG='JOHNNY B. GOOD'  
       INDUCTED=1986  
       VOTES=50  
       ARTIST='JAMES BROWN'  
       SONG='I GOT YOU'  
       INDUCTED=1986  
       VOTES=100  
       ARTIST='ELVIS PRESLEY'  
       SONG='JAILHOUSE ROCK'  
       INDUCTED=1986  
       VOTES=75  
       )
```

Delete Item(s) from a List

Use method DELITEM to remove items ARTIST and INDUCTED from each group in list IN1986.

```
CALL METHOD('DELITEM.SCL','INIT',
           IN1986,
           'ARTIST,INDUCTED');
```

The list now looks like this:

```
IN1986(SONG='JOHNNY B. GOOD'
       VOTES=50
       SONG='I GOT YOU'
       VOTES=100
       SONG='JAILHOUSE ROCK'
       VOTES=75
       )
```

Load a List to a Data Set

Data in a list can be flipped horizontally and placed into a new SAS data set. Method LIST2DS applies these rules during the conversion:

- item names become variables
- item values are assigned to the corresponding variables
- item types of either character or integer are given to the variables
- for each item name, the maximum length, of any of its values, is assigned as the length of the related variable.

Use method LIST2DS to load data from list IN1986 to data set DATA1986.

```
CALL METHOD('LIST2DS.SCL','INIT',
           IN1986,'DATA1986');
```

Data set DATA1986 looks like this:

SONG	VOTES
JOHNNY B. GOOD	50
I GOT YOU	100
JAILHOUSE ROCK	75

Conclusion

Although the examples shown in this paper are quite simple, the ten methods presented can be combined to create some powerful, yet concise modules--particularly when working with SAS/AF screens. The keys to fully utilizing these methods are: (1) to know the functionality of each one and (2) to envision the use of SCL lists to promote interaction between screens and data and end-users.

These methods have the potential to be expanded to include some additional checks and return codes. They may also inspire ideas for other utility-type methods.

References

SAS Institute Inc. (1990), SAS® Screen Control Language: Reference, Version 6, First Edition, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1991), SAS® Technical Report P-216, SAS/AF® Software, SAS/FSP® Software, and SAS® Screen Control Language: Changes and Enhancements, Release 6.07, Cary, NC: SAS Institute Inc.

SAS and SAS/AF are registered trademarks of SAS Institute, Inc., Cary, NC, USA.

Melissa J. Kerner
The Lubrizol Corporation
29400 Lakeland Blvd.
Wickliffe, Ohio 44092-2298
Internet: mjke@lubrizol.com

Appendix

Method	Description	Values Passed	Values Returned
DS2LIST	Load a SAS data set into a list	Data set name Where clause (opt.)	List identifier
VAR2LIST	Load specific data set variables into a list	Data set name Variable names Where clause (opt.)	List identifier
LISTINFO	Get the number of groups in a list and the number of items in each group	List identifier	# of groups # of items in each group
LMERGE	Merge two lists	List identifier of values List identifier of master data	List identifier
ITEMLIST	Create a list of unique item names	List identifier	List identifier
FINDITEM	Locate the position of a specific item in a list	List identifier Item name Value Start position (opt.)	Position in list
SUBGROUP	Subset a group from a list	List identifier Item name Value	List identifier
ADDITEM	Add item(s) to each group in a list	List identifier Item name(s) Item type Position (opt.)	
DELITEM	Delete item(s) from each group in a list	List identifier Item name(s)	
LIST2DS	Load a list into a data set	List identifier Data set name	

** For character items, specify label INITC when calling the method.
For numeric items, indicate INITN.

DS2LIST: Load a SAS data set into a list

```
length rc dsid 8;
INIT: METHOD dataset $ listid 8 optional=wclause $;
if wclause = _blank_ then
    dsid=open(dataset,'i');
else
    dsid=open(dataset||' (where=('||wclause||'))','i');
numvars=attrn(dsid,'NVAR');
listid=makelist(0,'G');
do while(fetch(dsid)=0);
    do i=1 to numvars;
        if vartype(dsid,i) = 'C' then
            listid=insertc(listid,getvarc(dsid,i),-1,varname(dsid,i));
        else
            listid=insertn(listid,getvarn(dsid,i),-1,varname(dsid,i));
        end;
    end;
end;
rc=close(dsid);
ENDMETHOD;
RETURN;
```

VAR2LIST: Load specific data set variables into a list

```
length rc dsid i specvars 8 varname $8;
INIT: METHOD dataset vars $ listid 8 optional=wclause $;
if wclause = _blank_ then
  dsid=open(dataset,'i');
else
  dsid=open(dataset||'(where=('||wclause||'))','i');
specvars=makelist(0);
i=1;
varname=scan(vars,i,',');
do until(varname = _blank_);
  specvars=insertc(specvars,varname,-1);
  i+1;
  varname=scan(vars,i,',');
end;
listid=makelist(0,'G');
do while(fetch(dsid)=0);
  do i=1 to listlen(specvars);
    varname=getitemc(specvars,i);
    if vartype(dsid,varnum(dsid,varname))='C' then
      listid=insertc(listid,getvarc(dsid,varnum(dsid,varname)),-1,varname);
    else
      listid=insertn(listid,getvarn(dsid,varnum(dsid,varname)),-1,varname);
    end;
  end;
end;
rc=close(dsid);
rc=dellist(specvars);
ENDMETHOD;
RETURN;
```

LISTINFO: Get the number of groups in a list and the number of items in each group

```
length nextname itemname $80 len i j rc listid2 8;
INIT: METHOD listid numgroup numitems 8;
len=listlen(listid);
if len > 0 then do;
  call method('SCREENS.UTILITY.ITEMLIST.SCL','INIT',listid,listid2);
  numitems=listlen(listid2);
  j=1;
  do i=1 to len;
    itemname=nameitem(listid,i);
    nextname=getitemc(listid2,j);
    if nextname ^= itemname then do;
      len=0;
      leave;
    end;
    if mod(i,numitems) = 0 then
      j=1;
    else
      j=j+1;
  end;
  numgroup=len/numitems;
  rc=dellist(listid2);
end;
else do;
  numgroup=0;
  numitems=0;
end;
ENDMETHOD;
RETURN;
```

LMERGE: Merge two lists

```
length i j numval numitems numgroup lowlimit uplimit 8 type $1
      name $8 charval $200;
INIT: METHOD listid listid2 listid3 8;
listid3=makelist(0,'G');
call method('SCREENS.UTILITY.LISTINFO.SCL','INIT',listid2,numgroup,numitems);
name=nameitem(listid,1);
type=itemtype(listid,1);
if type = 'N' then
  do i=1 to numgroup;
    numval=getnitemn(listid2,name,i,1);
    if searchn(listid,numval) > 0 then
      link WRITE;
  end;
else
  do i=1 to numgroup;
    charval=getnitemc(listid2,name,i,1);
    if searchc(listid,charval) > 0 then
      link WRITE;
  end;
RETURN;

WRITE:
uplimit=i * numitems;
lowlimit=uplimit - numitems + 1;
do j=lowlimit to uplimit;
  if itemtype(listid2,j) = 'N' then
    listid3=insertn(listid3,getitemn(listid2,j),-1,nameitem(listid2,j));
  else
    listid3=insertc(listid3,getitemc(listid2,j),-1,nameitem(listid2,j));
end;
RETURN;
ENDMETHOD;
RETURN;
```

ITEMLIST: Create a list of unique item names

```
length len i 8 item $200;
INIT: METHOD listid listid2 8;
len=listlen(listid);
listid2=makelist(0,'G');
do i=1 to len;
  item=nameitem(listid,i);
  if searchc(listid2,item) = 0 then
    listid2=insertc(listid2,item,-1);
  else
    leave;
end;
ENDMETHOD;
RETURN;
```

FINDITEM: Locate the position of a specific item in a list

```
length itempos valpos 8;
INITC: METHOD listid 8 itemname $ charval $ pos 8 optional=startpos 8;
type='C';
link GETPOS;
ENDMETHOD;
RETURN;
INITN: METHOD listid 8 itemname $ numval 8 pos 8 optional=startpos 8;
type='N';
link GETPOS;
ENDMETHOD;
RETURN;
GETPOS:
if startpos = . then startpos = 1;
itempos=nameditem(listid,itemname,1,startpos);
if type = 'C' then valpos=searchc(listid,charval,1,startpos);
else valpos=searchn(listid,numval,1,startpos);
do while((itempos^=valpos) and (valpos^=0) and (itempos^=0));
  if itempos > valpos then do;
    startpos=itempos;
    if type = 'C' then valpos=searchc(listid,charval,1,startpos);
    else valpos=searchn(listid,numval,1,startpos);
  end;
  else do;
    startpos=valpos;
    itempos=nameditem(listid,itemname,1,startpos);
  end;
end;
if (itempos = valpos) and (itempos ^= 0) then pos=itempos;
else pos=0;
RETURN;
```

SUBGROUP: Subset a group from a list

```
length i pos begin end 8;
INITN:
METHOD listid 8 item $ nvalue listid2 8;
call method('SCREENS.UTILITY.FINDITEM.SCL','INITN',listid,item,nvalue,pos);
if pos > 0 then link SUBSET;
ENDMETHOD;
RETURN;
INITC:
METHOD listid 8 item cvalue $ listid2 8;
call method('SCREENS.UTILITY.FINDITEM.SCL','INITC',listid,item,cvalue,pos);
if pos > 0 then link SUBSET;
RETURN;
ENDMETHOD;
SUBSET:
call method('SCREENS.UTILITY.LISTINFO.SCL','INIT',listid,groups,items);
end=ceil(pos/items) * items;
begin=end - items +1;
listid2=makelist(0,'G');
do i=begin to end;
  if itemtype(listid,i) = 'C' then
    listid2=insertc(listid2,getitemc(listid,i),-1,nameitem(listid,i));
  else
    listid2=insertn(listid2,getitemn(listid,i),-1,nameitem(listid,i));
end;
RETURN;
```

ADDITEM: Add item(s) to each group in a list

```
length i j inspos numgroup numitems 8 before after itemname $8;
INIT: METHOD listid 8 items $200 type $1
      optional=position 8;
i=1;
itemname=scan(items,i,',');
do until (itemname = _blank_);
  call method('SCREENS.UTILITY.LISTINFO.SCL 'INIT',listid,numgroup,numitems);
  if position > 0 then do;
    before=nameitem(listid,position);
    do j=1 to numgroup;
      inspos=nameditem(listid,before,j,1);
      if type = 'C' then
        listid=insertc(listid,' ',inspos,itemname);
      else
        listid=insertn(listid,..,inspos,itemname);
    end;
    position=position + 1;
  end;
else do;
  after=nameitem(listid,numitems);
  do j=1 to numgroup;
    inspos=nameditem((listid,after,j,1) + 1);
    if type = 'C' then
      listid=insertc(listid,' ',inspos,itemname);
    else
      listid=insertn(listid,..,inspos,itemname);
  end;
end;
i+1;
itemname=scan(items,i,',');
end;
ENDMETHOD;
RETURN;
```

DELITEM: Delete item(s) from each group in a list

```
length delpos 8;
INIT: METHOD listid 8 items $200;
i=1;
itemname=scan(items,i,',');
do until (itemname=_blank_);
  delpos=nameditem(listid,itemname);
  do while (delpos > 0);
    listid=delitem(listid,delpos);
    delpos=nameditem(listid,itemname);
  end;
  i+1;
  itemname=scan(items,i,',');
end;
ENDMETHOD;
RETURN;
```

LIST2DS: Load a list into a data set

```
length info pointer maxlen numitems numgroup len i j rc dsid 8 name $8
      type $1;
INIT: METHOD listid 8 dataset $;
call method('SCREENS.UTILITY.LISTINFO.SCL','INIT',listid,numgroup,numitems);
info=makelist(0,'G');
len=0;      maxlen=0;
do i=1 to numitems;
    name=nameitem(listid,i);
    type=itemtype(listid,i);
    info=insertc(info,name,-1,'NAME');
    info=insertc(info,type,-1,'TYPE');
    info=insertn(info,len,-1,'LEN');
end;
do i=1 to numgroup;
    do j=1 to numitems;
        pointer=j+((i-1)*numitems);
        name=nameitem(listid,pointer);
        type=itemtype(listid,pointer);
        if type = 'C' then len=length(getitemc(listid,pointer));
        else do;
            len=length(getitemn(listid,pointer));
            if len < 4 then len=4;
        end;
        call method('SCREENS.UTILITY.FINDITEM.SCL','INITC',info,'NAME',name,pos);
        if len > getnitemn(info,'LEN',1,pos) then
            info=setnitemn(info,len,'LEN',1,pos);
        end;
    end;
end;
call method('SCREENS.UTILITY.LISTINFO.SCL','INIT',info,numgroup,numitems);
if exist(dataset) then rc=delete(dataset);
dsid=open(dataset,'N');
do i=1 to numgroup;
    name=getnitemc(info,'NAME',i,1);
    type=getnitemc(info,'TYPE',i,1);
    len=getnitemn(info,'LEN',i,1);
    rc=newvar(dsid,name,type,len);
end;
rc=close(dsid);
call method('SCREENS.UTILITY.LISTINFO.SCL','INIT',listid,numgroup,numitems);
dsid=open(dataset,'u');
do i=1 to numgroup;
    rc=append(dsid);
end;
rc=close(dsid);
dsid=open(dataset,'u');
j=1;
rc=fetch(dsid);
do i=1 to listlen(listid);
    if getnitemc(info,'TYPE',j,1) = 'C' then
        call putvarc(dsid,j,getitemc(listid,i));
    else
        call putvarn(dsid,j,getitemn(listid,i));
    if mod(i,numitems) = 0 then do;
        rc=update(dsid);
        rc=fetch(dsid);
        j=1;
    end;
    else j=j+1;
end;
rc=close(dsid);      rc=dellist(info);      rc=dellist(listid);
ENDMETHOD;
RETURN;
```