



PRAHEALTHSCIENCES

Getting organized for SAS Macro development: hints, tips and tools for a structured approach

Iain Humphreys, PRA Health Sciences, UK



Agenda

- Aims
- Best practices
- Commonly occurring tasks
- Broader considerations
- Conclusions



- Topics
 - Parameter validation
 - Error handling
 - Keeping the environment tidy
- Necessary, but...
 - Distraction
 - Repetitive
- How do we encourage and instil?



- Checking...
 - whether a value has been supplied for a (mandatory) parameter
 - whether an entity exists
 - SAS dataset
 - Variable
 - File or directory name
 - File or directory reference
 - against a list of permissible values



- Purpose: verify that a variable exists

```
%macro g_util_chkVarExist(ds=, var=) ;  
  %local _dsid _varexist _rc _vxist ;  
  %let _varexist = 0 ;  
  
  %let _dsid = %sysfunc(open(&ds)) ;  
  %if &_dsid > 0 %then %do ;  
    %let _vxist=%sysfunc(varnum(&_dsid, &var)) ;  
    %if &_vxist > 0 %then %let _varexist = 1 ;  
    %let _rc = %sysfunc(close(&_dsid)) ;  
  %end ;  
  &_varexist  
%mend g_util_chkVarExist ;
```

- Example call

```
%if not %g_util_chkVarExist(ds=&_ADaM..&data, var=&tmt_var)  
%then %do ;  
  * << appropriate action for a variable not found >> ;  
%end ;
```



- Return codes
- Log warnings
- Exit point

```
%global &_rc ;
%let _rc = _rc_&sysmacroname ;
%let &_rc = 99 ;

%if %g_util_chkParamBlank(&tmt_var) %then %do ;
    %put %str(W)ARNING: Treatment variable parameter is missing ;
    %let &_rc = 1 ;
    %goto exit ;
%end ;

* << Further macro processes >> ;

%let &_rc = 0 ;
%exit:
    %put NOTE: ENDING MACRO &sysmacroname, RC = &&&_rc ;
%mend ;
```



- Utilities for...
 - Removing unwanted temporary datasets
 - Deleting or re-setting global macro symbols
 - Re-setting SAS options
 - Releasing LIBNAMEs and FILENAMEs
- Can be called...
 - Initialization
 - Termination



- Determining the number of observations in a dataset
- Determining whether a variable is character or numeric
- Building a list of items to be processed
- Counting the number of items in a list



Commonly occurring tasks in macro development

Example utility – return a variable's type

```
%macro g_util_vtype(dset, var) ;  
    %local __dsid __vno __vtype __vfound __rc ;  
    %let __vfound = 0 ;  
    %let __dsid = %sysfunc(open(&dset)) ;  
    %if &__dsid > 0 %then %do ;  
        %let __vno = %sysfunc(varnum(&__dsid, &var)) ;  
        %if &__vno > 0 %then %do ;  
            %let __vfound = 1 ;  
            %let __vtype = %sysfunc(vartype(&__dsid, &__vno)) ;  
        %end ;  
        %let __rc = %sysfunc(close(&__dsid));  
        %if &__vfound eq 1 %then %do ;  
            &__vtype  
        %end ;  
        * << statements handling variable / dataset not found >> ;  
    %mend g_util_vtype ;
```

- Example call

```
%if %g_util_vtype(&inds, &avar) EQ C %then %do ;  
    * << statements for when variable type is character >> ;  
%end ;  
%else %do ;  
    * << statements for when variable type is not character >> ;  
%end ;
```



- Formal validation
- Macro compilation
- Naming conventions
- Documentation



- Pause and think...
 - Best practices
 - Frequently occurring tasks
- Get organised...
 - Develop, implement and use utilities
- Benefits
 - More robust macros
 - Consistent approach
 - Efficient development
 - Easier to validate



- Questions?

