

# Macros to Help Organize and Standardize SAS Projects

S. Patrick Thornton, SRI International, Menlo Park, CA

## ABSTRACT

Organizing SAS® System project files, creating library names (i.e., librefs), and typing file paths are all common activities throughout a SAS project. Each of these ubiquitous activities and decisions subtly increases the time to complete a project and may introduce inconsistencies that make documenting, migrating, or working across projects a challenge. This paper discusses a macro program for creating and using folders to organize files across SAS projects. At the start of a SAS project, the macro was used to create user-defined folders to store files. At the start of a SAS session, the macro was used to establish a library name for each folder and to store the path for each folder in a macro variable. The consistent use of folders, library names, and macro variables to resolve folder paths is shown to help with documentation, project migration to new locations, and cross-project and/or cross-programmer readability.

## INTRODUCTION

Organizing SAS project files, creating library names, and referencing file paths are all activities that programmers do to achieve input and output of data and results. For example, a typical syntax for reading a SAS data set from a permanent location is:

```
Libname schools "c:\projects\schools\";
Data sch1;
set schools.sch1;
run;
```

The LIBNAME statement is used to create the libref SCHOOLS that references the physical location "c:\projects\schools". Thus, when SAS processes the statement SET SCHOOLS.SCH1 the SCH1 data set is obtained from the physical location of SCHOOLS. If the school data were moved, the program could operate correctly only after the absolute path (c:\projects\schools) was changed to the new location of the school data.

In addition to the LIBNAME statement, programs may have any number of other statements or procedures that use an absolute folder and/or file path. For example:

```
Footnote1 "Data: c:\projects\schools\";
Data sch1.schimport;
  Infile "c:\projects\schools\schoolimport.txt" lrecl=4230 pad missover;
Run;
ods html file="myfiletoedit.xls" path=" c:\projects\schools\";
proc print data= sch1.schimport noobs;
run;
ods html close;
```

Typing or using copy and paste to place these absolute folder/file paths into SAS syntax wherever needed is perhaps the most direct technique; however, it may create more work later. If the physical location of the data is ever changed, every folder/file path throughout all project programs will have to be changed. In addition, each new program is likely to require LIBNAME statements. Creating the statements each time may create inconsistency across programs. It's not uncommon on large projects to find several librefs used to define the same physical location. For example, program 1 may define the SCHS libref, while program 2 defines the ALLSCHS libref:

```
Libname schs "c:\projects\schools\";
Libname allschs "c:\projects\schools\";
```

Inconsistency may be compounded when there are multiple permanent data locations, all of which may have a number of different librefs across programs. This kind of inconsistency needlessly makes it more difficult for new programmers to understand the data sources used in the programs.

The program and macro presented in this paper were created to aid programmers by providing a tool to:

- (a) Create a standard set of folders for each new project
- (b) Create a standard libref for each folder at the beginning of a SAS session
- (c) Create a macro variable storing the path to each folder at the beginning of a SAS session

## FOLDER STRUCTURE

On large projects, it is particularly useful to establish a folder structure for organizing files, and such an organization pays further dividends when used consistently across projects. The macro in this paper can be used to create any number of first-level folders with the names provided by the user. Table 1 shows an example of folders and descriptions used to organize files for a project.

**TABLE 1. EXAMPLE FOLDER STRUCTURE TO ORGANIZE PROJECT FILES**

Folder	SAS Libref	Description
SASdata	Sdata	All SAS data sets obtained and/or created through syntax files.
SASdoc	Sdoc	All documentation for the project.
SASexport	Sexport	All non-SAS data files that are created through syntax files for the purpose of export outside the project.
SASformat	sformat	All SAS format catalogues.
SASHTML	Shtml	The contents are HTML output created through syntax files.
SASImport	Simport	All original non-SAS data used in the project.
SASintra	Sintra	All non-SAS data files created, edited, and used by syntax files for further processing (e.g., data cleaning and recoding).
SASlis	Slis	Any output saved from the SAS output window.
SASlog	Slog	Any output saved from SAS log window.
SASmacro	Smacro	Macro programs that are used only in this project.
SASsyntax	Ssyntax	SAS syntax files used in the current project.

## A NEW PROJECT: CREATING FOLDERS

At the beginning of a new project, the macro SETDIR (see Syntax section) was used to create a project folder and the subfolders defined in Table 1. To create the folders, SETDIR required the following information:

- (a) Global root folder path – the location of a folder with the subfolder “.\macro” containing SETDIR
- (b) Global projects folder path – the location where you want to create a new folder for the project
- (c) Project folder – the name of the new folder to create within the global projects folder
- (d) Author – the name of the project creator
- (e) Subfolder names – a list of folder names to create the project folder
- (f) Library names – a list of names to use as SAS library names
- (g) Short Folder Descriptions – a list that describes the contents of each folder

The macro SETDIR was called in the program NEWSETDIR.SAS (see Syntax section). The information needed by SETDIR was entered as CARDS in the data step CONTROL. The data step created macro variables that were passed to macro SETDIR. This strategy was chosen over creating macro variables directly with %LET statements because the information was saved to the data set CONTROL and to macro variables at the same time. The key to the strategy was CALL SYMPUT. The routine was used to create macro variables from each CARDS line that began with “let.” For example, the information needed for the global root path (a) (e.g., c:\sas resources) was obtained from the following CARDS line and saved to the macro variable GROOT by CALL SYMPUT:

```
let groot = C:\sas resources
```

It's important to note that some of the CARD lines beginning with “let,” as shown in the Syntax section, have a soft return due to text wrapping in the Microsoft Word document. All information for each “let” must be on one CARD line when submitted in SAS.

After the data step, the program that contains the macro SETDIR was included, followed by a call to SETDIR, in order to create the new folders:

```
%include "&groot.\macros\setdir.sas"/source2;
%setdir(%str(&folders), %str(&lib), %str(&des), sassyntax, cf=y);
```

The CALL SYMPUT routine in the data step created the macro variables FOLDERS, LIB, and DES. The macro variables resolved to the new folder names, the librefs, and the folder descriptions, respectively. The macro parameter CF set to Y (i.e., cf=y) directed the macro SETDIR to create new folders and to create the program CONTROL.SAS.

## USING CONTROL.SAS AT THE START OF A SAS SESSION

CONTROL.SAS (see Syntax section) was a standalone program created by SETDIR during the creation of the new project. The program was submitted at the beginning of a SAS session to establish librefs and macro variables for the project. The program has a %LET statement for each piece of information that was provided during the project creation (e.g., %let groot = C:\sas resources). The following lines were used to include and call SETDIR:

```
%include "&groot.\macros\setdir.sas"/source2;
%setdir (&folders, &lib, %str(&des), sassyntax);
```

Since the macro parameter CF was not specified as it was during project creation, folders were not created; rather, only the librefs and macro variables were created (Table 2).

**TABLE 2. EXAMPLE LIBNAME STATEMENTS AND MACRO VARIABLES CREATED BY CONTROL.SAS**

Libref Created with LIBNAME Statement	Macro Variable	Macro Variable Resolution
libname sdata "C:\projects\nbrcc3\sasdata";	sasdata	C:\projects\nbrcc3\sasdata
libname sdoc "C:\projects\nbrcc3\sasdoc";	sasdoc	C:\projects\nbrcc3\sasdoc
libname sexport "C:\projects\nbrcc3\sasexport";	sasexport	C:\projects\nbrcc3\sasexport
libname sformat "C:\projects\nbrcc3\sasformat";	sasformat	C:\projects\nbrcc3\sasformat
libname shtml "C:\projects\nbrcc3\sashtml";	sashtml	C:\projects\nbrcc3\sashtml
libname simport "C:\projects\nbrcc3\sasimport";	sasimport	C:\projects\nbrcc3\sasimport
libname sintra "C:\projects\nbrcc3\sasindra";	sasindra	C:\projects\nbrcc3\sasindra
libname slis "C:\projects\nbrcc3\saslis";	saslis	C:\projects\nbrcc3\saslis
libname slog "C:\projects\nbrcc3\saslog";	saslog	C:\projects\nbrcc3\saslog
libname smacro "C:\projects\nbrcc3\sasmacro";	sasmacro	C:\projects\nbrcc3\sasmacro
libname ssyntax "C:\projects\nbrcc3\sassyntax";	sassyntax	C:\projects\nbrcc3\sassyntax

The librefs and macro variables shown in Table 2 were used across all programs in the project in a variety of statements and procedures; for example:

```
Footnote1 "Data: &sasdata";
Footnote2 "Programs: &sassyntax";
Data sch1.schimport;
  Infile "&sasimport.\schoolimport.txt" lrecl=4230 pad missover;
Run;
ods html file="myfiletoedit.xls" path="&sasindra";
proc print data= sdata.schimport noobs;
run;
ods html close;
```

Consistent use of librefs and macro variables, allowed the programs to be relocated with only the effort needed to alter CONTROL.SAS and re-execute SETDIR. For example, when the project folder NBRCC3 was renamed to NBRCC and moved to "c:\myprojects" the following two macro variables in CONTROL.SAS were changed to the following:

```
%let projroot = C:\myprojects;
%let projfolder = nbrcc;
```

On executing CONTROL.SAS, the macro SETDIR established the library names and macro variables using the new absolute path to the project ("c:\myprojects\nbrcc"), thus all programs operated from the new location. Using consistent librefs and macro variables also added clarity by eliminating duplicate naming for the same locations across programs. To examine the syntax in more detail see the Syntax section, otherwise turn to the conclusion for a summary of the paper.

## SYNTAX

### MACRO SETDIR

```
%macro setdir(folders,lib,des,syn,max=100,cf=no,fr=,fn=);
%global fnc full groot projroot thisprojroot sasmacro &folders;
%let sasmacro = &groot.\macros;
%if &fr = %then %do;
  /*if new project, cf default is changed make new project folder*/
  %if &cf ne no %then %do;
    /*make new project subfolder*/
    %sysexec cd &projroot;
    %sysexec md &projfolder;
  %end;
  /*document libnames and paths for project*/
  %let thisprojroot=&projroot.\&projfolder;
  %let s = 1;
  data _null_;
    file "&thisprojroot.\record.txt";
```

```

%do %until (%scan(&folders,&s)= or &s > &max);
  %let ind = %str(sp&s);;
  %global &ind;
  %let desc = %scan(&des,&s,*);
  %let t = %scan(&folders,&s);
  %let l = %scan(&lib,&s);
  /*create global marcos storing folder paths*/
  %let &t = &thisprojroot.\&t;
  %let sp&s = &t;
  %let lib&s = &l;
  %put &desc;
  %put sp&s &&sp&s &&&t;
  %let s = %eval(&s + 1);
  put "&desc:";
  put "&&&t";
%end;
run;
%let fcnt = %eval(&s-1);
/*if new project create sub folders*/
%if &cf ne no %then %do;
  %sysexec cd &projfolder;
  %do s = 1 %to &fcnt;
    %let t = &&sp&s;
    %sysexec md &t;
  %end;
  data _null_;
    file "&&&syn.\control.sas";
    set control;
    put bigv ' ';
  run;
  %put "sysexec cd " &projroot;
%end;
/*set library names*/
%do s = 1 %to &fcnt;
  %let t = &&sp&s;
  libname &&lib&s "&&&t";
%end;
%end;
%else %do;
  data _null_;
    file "&&&fr.\&fn..sas";
    set head;
    put bigv ' ';
  run;
%end;
%mend;

```

#### NEWSETDIR.SAS

```

options mprint noxwait;
data control;
  input bigv & $200.;
  fc = substr(bigv,1,7);
  if index(fc,'let')=1 then do;
    mn = scan(bigv,2); *let variable;
    ty = scan(bigv,3); *equal;
    if ty = '=' then do;
      p = index(bigv,'=') + 1;
      le = length(bigv); *total length of line;
      mv = substr(bigv,p,le + 1 - p);
      call symput(mn,trim(left(mv)));
      bigv = "%||trim(left(bigv));
    end;
  end;
  else if index(fc,'include')=1 then do;
    bigv = "%||trim(left(bigv));
  end;

```

```

        end;
        else if index(fc,'setdir')=1 then do;
        bigv = "%||trim(left(bigv));
        end;
cards;
*Instructions:
*Global resource root-i.e. formats, macros to share across projects
*A ./macro subfolder is assumed e.g. if your global root is c:\ then c:\macros\
let groot = C:\sas resources
include "&groot.\macros\setdir.sas"/source2
*The project root should be the parent folder for all projects you want to create.
*Define the project root by setting the macro variable &projroot
let projroot = C:\projects
*Set the macro variable &newprojfolder to the name of the sub folder to create in
your
*project root. This is the folder within the project root where you want
*your specific project to be create
let projfolder = nbrcc3
*author
let author = Patrick Thornton
*Set the names and descriptions of folders and libnames
let folders = sasdata sasdoc sasexport sasformat sashtml sasimport sasintra saslis
saslog sasmacro sassyntax
let lib = sdata sdoc sexport sformat shtml simport sintra slis slog smacro ssyntax
let des1 = All SAS data sets obtained and/or created through syntax files.
let des2 = All documentation for the project.
let des3 = All non-SAS data files that are created through syntax files for the
purpose of export outside the project.
let des4 = All SAS format catalogues.
let des5 = The contents are HTML output created through syntax files.
let des6 = All original non-SAS data used in the project.
let des7 = All non-SAS data files created edited and used by syntax files for
further processing
let des8 = Any output saved from the SAS output window.
let des9 = Any output saved from SAS log window.
let des10 = Macro programs that are used only in this project.
let des11 = SAS syntax files used in the current project
let des = &des1*&des2*&des3*&des4*&des5*&des6*&des7*&des8*&des9*&des10*&des11
*After you create control.sas with this program, can use the syntax each
*time you begin to work on the project to set libnames and macro variables. You may
also
*want to use any number of include statements in your control.sas file to run and
document
*various SAS syntax files used and developed in the project.
*this call of setdir establishes libnames
setdir (&folders,&lib,%str(&des),sassyntax)
;
run;
%put Global Resource Root: &groot;
%put Project Root: &projroot;
%put Projfolder: &projfolder;
%put Folders: &folders;
%put Library Names: &lib;
%let des = &des1*&des2*&des3*&des4*&des5*&des6*&des7*&des8*&des9*&des10*&des11;
%put Descriptions: &des;
%include "&groot.\macros\setdir.sas"/source2;
*create folders;
%setdir(%str(&folders),%str(&lib),%str(&des),sassyntax,cf=y);

```

#### CONTROL.SAS

```

*Instructions: ;
*Global resource root-i.e. formats, macros to share across projects ;
*A ./macro subfolder is assumed e.g. if your global root is c:\ then c:\macros\ ;
%let groot = C:\sas resources ;
%include "&groot.\macros\setdir.sas"/source2 ;

```

```

*The project root should be the parent folder for all projects you want to create. ;
*Define the project root by setting the macro variable &projroot ;
%let projroot = C:\projects ;
*Set the macro variable &newprojfolder to the name of the sub folder to create in
your ;
*project root. This is the folder within the project root where you want ;
*your specific project to be create ;
%let projfolder = nbrcc3 ;
*author ;
%let author = Patrick Thornton ;
*Set the names and descriptions of folders and libnames ;
%let folders = sasdata sasdoc sasexport sasformat sashtml sasimport sasintra saslis
saslog sasmacro sassyntax ;
%let lib = sdata sdoc sexport sformat shtml simpport sintra slis slog smacro ssyntax
;
%let des1 = All SAS data sets obtained and/or created through syntax files. ;
%let des2 = All documentation for the project. ;
%let des3 = All non-SAS data files that are created through syntax files for the
purpose of export outside the project. ;
%let des4 = All SAS format catalogues. ;
%let des5 = The contents are HTML output created through syntax files. ;
%let des6 = All original non-SAS data used in the project. ;
%let des7 = All non-SAS data files created edited and used by syntax files for
further processing ;
%let des8 = Any output saved from the SAS output window. ;
%let des9 = Any output saved from SAS log window. ;
%let des10 = Macro programs that are used only in this project. ;
%let des11 = SAS syntax files used in the current project ;
%let des = &des1*&des2*&des3*&des4*&des5*&des6*&des7*&des8*&des9*&des10*&des11 ;
*After you create control.sas with this program, can use the syntax each ;
*time you begin to work on the project to set libnames and macro variables. You may
also ;
*want to use any number of include statements in your control.sas file to run and
document ;
*various SAS syntax files used and developed in the project. ;
*this call of setdir establishes libnames ;
%setdir (&folders,&lib,%str(&des),sassyntax) ;

```

## CONCLUSION

Using macro SETDIR and its calling program NEWSETDIR.SAS allowed the easy creation of user-defined folders for a project and the creation of the program CONTROL.SAS. CONTROL.SAS was used to call SETDIR and create librefs and macro variables at the start of a SAS session. The librefs and macro variables were used consistently in statements and procedures used for input/output of data and results across programs. Consistent use of folder names, librefs, and macro variable names increased organization and clarity, while reducing the effort needed to relocate and document programs.

## CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Patrick Thornton, Ph.D.  
SRI International  
333 Ravenswood Avenue  
Menlo Park, CA 94025-3493  
Work Phone: 650 859-5583  
Email: Patrick.thornton@sri.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.