PharmaSUG 2025 - Paper SD-230

A SAS® System PowerShell macro to report directory-level metrics by Owner (programmer)

Kevin R. Viel, Ph.D., Histonis LLC and Navitas Data Sciences

ABSTRACT

In the Windows operating system, the Owner of a file is typically the person who last saved a file, which we expect to be the author or responsible person for the file. The SAS® System includes the FINFO() function, which returns file metadata, but not the Owner on the Windows operating system. The Windows PowerShell, however, does return the Owner. While auditing a directory or set of directories, such information is required by a Lead Programmer or other manager and can be used to provide metrics, such as the distribution of times required to execute programs or the use of programming techniques or code, such as macros. The goal of this paper is to describe a SAS macro, MAC_U_FINFO, to abstract file metadata such as datetime (of last modification) or bytes, and, a SAS macro, MAC_PS_GETCHILDITEM, which, if when on the Windows operating system, abstracts the Owner of a file. Such data can help compile programmer-specific metrics, assure that the program header correctly lists the responsible author, or determine how often a specific macro or code was used.

INTRODUCTION

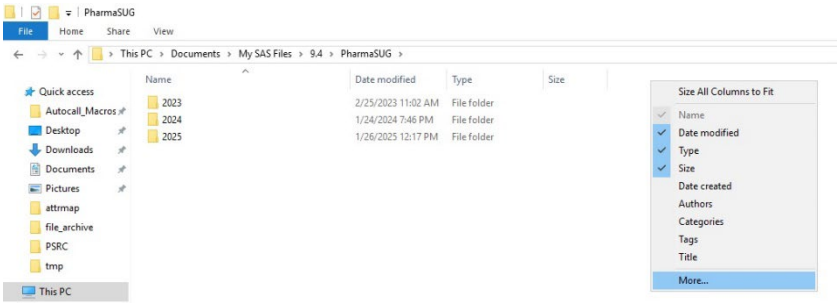
At times, metadata of files such as Last Modified or Owner, like file size, can be important. Although not pertinent to a submission directly, both can facilitate or even be required for the audit or reconciliation of a delivery. For instance, given certain SOPs (Standard Operating Procedures), WIs (Working Instructions), and team conventions, the owners of the files might be required to be the persons attesting to the successful completion on a given date (and time) on Validated Forms.

In addition, the age of files relative to one another might be important. Knowing the “Last Modified” datetime stamp could reveal important information that would be other hard to impossible to conclude without it. Besides manually inspecting files, which is not feasible for almost any size project, the SAS® System provides the FINFO() function and Microsoft Windows® provides the PowerShell to obtain metadata programmatically.

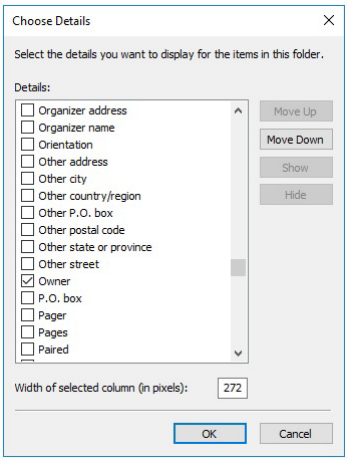
The **goals** of this paper are 1) to describe a SAS macro, MAC_U_FINFO, to abstract file metadata such as datetime (of last modification) or bytes, and 2) to describe a SAS macro, MAC_PS_GETCHILDITEM, which, if when on the Windows operating system, abstracts the Owner of a file.

WINDOWS FILE METADATA

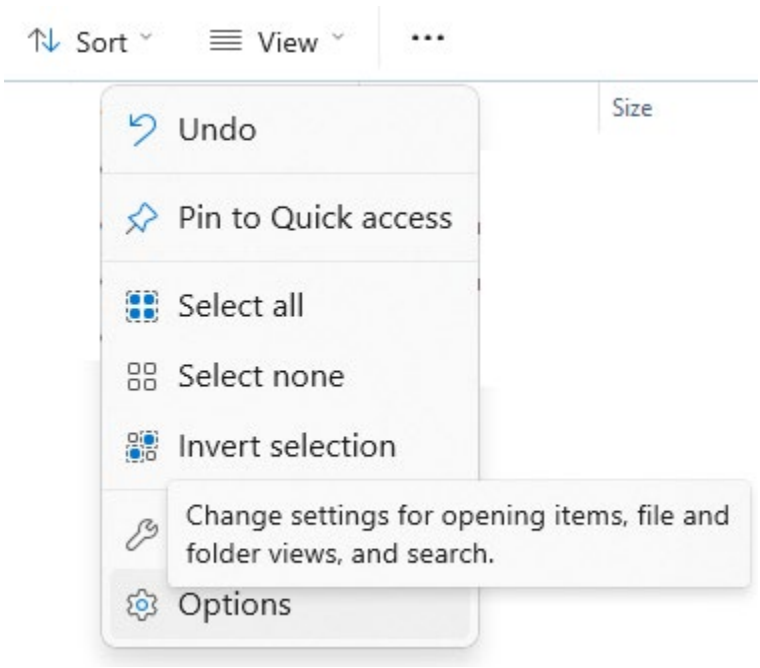
The Windows File Explorer provides a subset of metadata, like Date Modified (Last Modified) available to the operating system. In the File Explorer, one can select and add the metadata to display by right clicking the header and selecting the variable of choice:



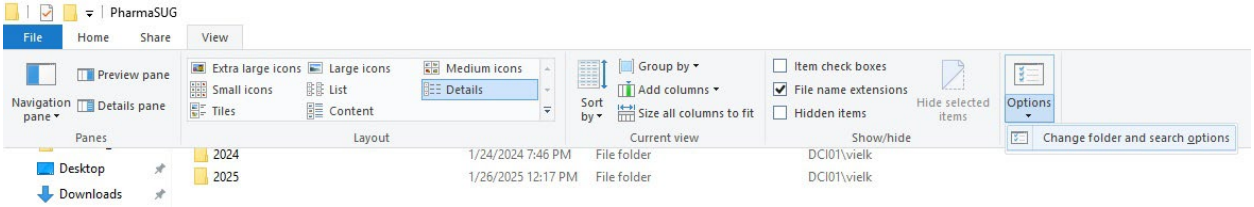
After clicking “More…” and scroll down to “Owner” and click it:



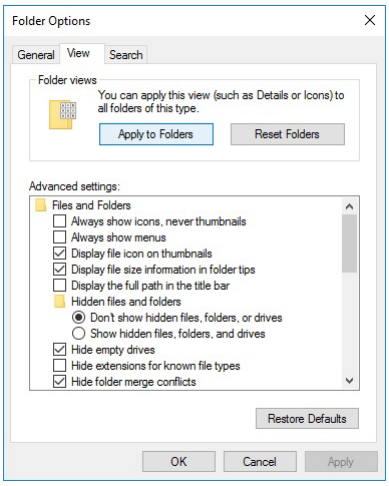
One can make this the default view of File Explorer in other folders. To do so, in the “…” tab of the ribbon of the File Explorer, select Options:



One may also see:



In the “Folder Options” pop up, again choose the View tab and click “Apply to Folders”:



THE SAS FINFO() FUNCTION

The SAS System function FINFO() “returns the value of a system-dependent information item for an external file”[1]. On the Windows operating system, the author has not been able to abstract Owner using the FINFO() function:

```
proc setinit ;
run ;

Current version: 9.04.01M5P091317
Operating System:   WX64_E  .

/* See Page 747 of SAS® 9.4 Functions and CALL Routines: Reference, Fifth
   Edition

   I changed the length of INFOVAL and substituted the 'physical-filename'.
*/
data info;
 length infoname $60 infoval $ 200 ;
 drop rc fid infonum i close;
 rc=filename('abc', 'C:\Users\vielk\Documents\My SAS
Files\9.4\Autocall_Macros\mac_ps_getchilditem.sas');
 fid=fopen('abc');
 infonum=foptnum(fid);
 do i=1 to infonum;
 infoname=foptname(fid, i);
 infoval=finfo(fid, infoname);
 output;
 end;
 close=fclose(fid);
run;

proc print
   data = info ;
run ;

Obs     infoname              infoval

 1      Filename              C:\Users\vielk\Documents\My SAS Files\9.4\Autocall_Macros\mac_ps_getchilditem.sas
 2      RECFM                 V
 3      LRECL                 131068
```

```
4     File Size (bytes)     7971
5     Last Modified         11Apr2024:10:23:54
6     Create Time           26Jan2025:12:10:37
```

The author presented one use of the FINFO() function in the macro MAC_U_DIRECTORY_READ[2].

MAC_U_FINFO MACRO

Appendix 1 presents the macro MAC_U_FINFO. **Log 1** shows the help section printed to the SAS Log when HELP = Y with a brief description of the macro parameters and their default values, if any. The code relies on SAS functions, so one would hope that the macro would also be as agnostic with respect to the OS (Windows versus Linux, for instance) as the functions. Regardless, the reader might wish to update or even make certain code more flexible, for instance, the values of the FOPTNAME (for example, **Lines 207-214** of Appendix 1).

```
Purpose of program:     This utility macro reports the file metadata (FINFO)

Macro Parameter         Description
_______________         ___________________________________________________________
in_ds                 = Data set contain path and file or pathfile.
                            Default: files
pathfile              = Path and file name of a single file.
                            Default:
keep                  = Variables to keep in the output data set.
                            Default: path
                                     file
out                   = Output data set
                            Default: finfo
foptname              = File information to obtain:
                            "Filename"          -> filename
                            "Owner Name"        -> owner
                            "Group Name"        -> group
                            "Access Permission" -> access_permission
                            "Last Modified"     -> last_modified
                            "Create Time"       -> create_time
                            "File Size"         -> file_size
                            "File Size (bytes)" -> file_size_bytes
                            Default:   Last Modified
                                     # Owner Name
foptname_delim        = Delimited for the list of foptname.  This SHOULD NOT be space for
                        certain FOPTNAMEs.
                            Default: #
print                 = Whether to print the results to the LST file.
                            Default: Y
put_header            = The header of the print.
                            Default: @1   "Path"
                                     @120 "File"
                                     @160 "Last Modified"
put_variables         = The variables to print.
                            Default: @1   path
                                     @120 file
                                     @160 last_modified
delete                = Data sets to delete.
                            Default: files
                                     finfo
os_path_delim         = The delimiter of the path in the operating system.
                            Default:  \
message_err           = Whether to report "ER" "ROR:" if the files is missing.
                            Whether to replace "does" " not" " exist" with "dne".
                            The message is report with or without the words above.
                            Default:  N
```

Log 1. The Help section of MAC_U_7ZIP.

EXAMPLE OF MAC_U_FINFO

The author includes a version of the following at the start of every program:

```
data files ;

  length path $ 200
         file $  30
         ;

  path = pathname( "sdtm" ) ;
  file = "dm.sas7bdat" ;
  output ;
  file = "lb.sas7bdat" ;
  output ;
  file = "supplb.sas7bdat" ;
  output ;

  /***/
  path = pathname( "adam" ) ;
  file = "adsl.sas7bdat" ;
  output ;

  /***/
  path = "&raw_path." ;
  file = "&local_lab_ranges." ;
  output ;

run ;

%mac_u_finfo
   ( in_ds          =  files ) ;
```

The output in this contrived case is:

```
Path                    File                                Last Modified
~\PharmaSUG\2025\SDTM\  dm.sas7bdat                         26JAN2025:14:09:13
~\PharmaSUG\2025\SDTM\  lb.sas7bdat                         26JAN2025:14:09:13
~\PharmaSUG\2025\SDTM\  supplb.sas7bdat                     26JAN2025:14:09:13
~\PharmaSUG\2025\ADAM\  adsl.sas7bdat                       26JAN2025:14:09:13
~\PharmaSUG\2025\RAW\   Local_Lab_Ranges_20250126.xlsx      26JAN2025:14:01:42
```

This provides the user with the ability to flag issues, like if the age of DM is younger than the age of ADSL. It also provides a record that the user can use to investigate issues, like why output changed. Without some record in the .log or .lst file one might not realize that, inadvertently or with purpose, a data set was updated. When the author is a Validation programmer, he includes the program, log, and output (data sets and/or PDF/RTF files, for instance), among the list of files. If the Last Modified datetime of the .sas file is younger than its other associated file, the file could have been updated but not re-run. Depending on the SOP/WI (Standard Operating Procedures/Working Instructions) governing the process, the Validation program could re-run the program or request that the Main (Production) peer-programmer or Lead Programmer re-run the program. Note that MAC_U_BATCH_SUBMIT[2] allows a user to blindly execute programs without needing to open them, that is, view potentially more than the header, which should not occur in a 100% independent double programming environment.

MAC_PS_GETCHILDITEM MACRO

Appendix 2 presents the macro MAC_PS_GETCHILDITEM. In the name of the macro, the author chose "PS" as the descriptive portion, for **P**ower**S**hell, as opposed to the "U", for **U**tility, in MAC_U_FINFO. **Log 2** shows the help section printed to the SAS Log when HELP = Y with a brief description of the macro parameters and their default values, if any.

```
Purpose of program:      This (Microsoft Windows) PowerShell macro uses the Get-ChildItem (gci)
                         to obtain the filename and owner of files in a path (using the default
                         parameters values).

Macro Parameter          Description
_______________      = _____________________________________________________________________
path                 = The path of interest.
filter               = The FILTER parameter and its argument.
                           Default: -filter *.sas
attributes           = The ATTRIBUTES parameter and its argument.
                           Default: -Attributes !Directory
extra                = Any extra parameter/argument pairs.
                           Default: %str()
so_property          = Arguments following Select-Object -Property
                           Default: Name%str(,)
out_ds               = The name of the output data set.
                           Default: getchilditem_file_owner
code1                = Any SAS code in the data step that creates the output data set, which
appears just
                            statement in the DO-BLOCK of the pattern match.
                           For example: if prxmatch( "/^[tfl]\d+.*\.sas$/i" , strip( file )) then
                           Default: %str()
code2                = Any SAS code in the data step that creates the output data set, which
                         appears just before the RUN statement.
                           Default: %str()
debug                = Whether to drop LINE from the output data set and to put the non-
                         matching LINEs to the log.
                            Default: N
_____________________________________________________________________________________________
End of help
```

Log 2. The Help section of MAC_PS_GETCHILDITEM.

This macro to writes the command line analogous to what one would write “by hand” on the command line (**Figure 1**). Please note that Windows is case insensitive.

A

B

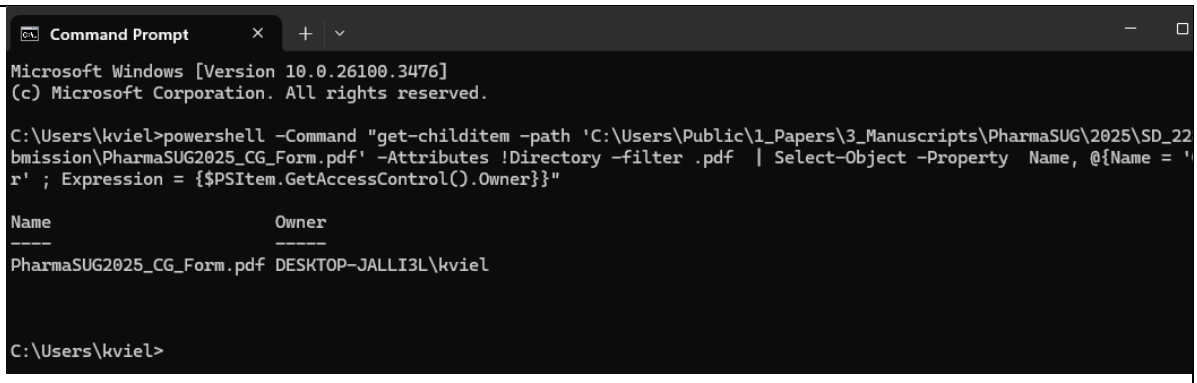


Figure 1. A partial display of the results of calls to PowerShell on Windows using the command line.

The command that resulted in Figure 1B follows, but should be one line:

```
powershell -Command "get-childitem -path
'C:\Users\Public\1_Papers\3_Manuscripts\PharmaSUG\2025\SD_228\Submission\PharmaSUG2025_CG
_Form.pdf' -Attributes !Directory -filter .pdf  | Select-Object -Property  Name, @{Name =
'Owner' ; Expression = {$PSItem.GetAccessControl().Owner}}"
```

This macro uses the PIPE to the FILENAME statement, capturing the results returned by the call. Note, one can simplify the call to PowerShell, if one prefers to perform actions like subsetting in SAS instead of using the -filter option to Powershell.

EXAMPLES

Example 1: Select files in a given directory (folder) from a specific user.

```
%mac_ps_getchilditem
   ( path            = C:\Users\vielk\Autocall_Macros
   , code1           = if prxmatch( "/vielk/i" , owner ) then
   ) ;
```

Example 2: (Positive Control) Select the files with a specific pattern found

```
data postive_control ;
  length pathfile   $ 100
         line       $ 500
         fn         $ 200
         ;
  infile "C:\Users\vielk\Autocall_Macros\*.sas"
         filename = fn
         ;
  input ;
  if prxmatch( '/%macro\s/i' , _infile_ )
  then
    do ;
       pathfile = fn ;
       line     = _infile_ ;
       output ;
    end ;
run ;
```

In this case, every SAS file in an Autocall_Macros should have a %MACRO statement. In fact, the data set can be used to investigate whether each file has only one such statement .

Example 3: Select the owner of files with a specific pattern found

```
%mac_ps_getchilditem
   ( path = C:\Users\vielk\Autocall_Macros ) ;

proc sql ;
  select distinct strip( line ) as line
  from files
  ;
quit ;

line
--------------------------------------------------------------------------------
%mac_u_delete
SAS Macro Used   ::  %mac_u_delete
```

Consider that a lead programmer might want to know what programs from which programmers used a certain macro. In this case, the lead programmer knows that the line starting with “SAS Macro Used” is from the header and, without checking the log, decides to omit this line. The macro may be executed, however, in the body of the program.

```
proc sql ;
  select distinct a.owner
       , a.file
  from getchilditem_file_owner as a
     , files                   as b
  where     catx( "\"
                , a.file
                , a.owner
                ) = b.pathfile
        and prxmatch( "/SAS Macro Used/" , b.line ) = 0
  ;
quit ;
```

The lead programmer now has a list of programs of interest linked to their owners. In this case, the Lead could also contact the owner of the program to request that she or he removes the percent from the name of the program in the header. If one can email within SAS, such a data set makes writing and sending the emails easy. Further, one might want to send one email to the owners with a list of the “offending” programs, instead of one email per program.

Case study: In a moderately sized delivery (100-200 tables), certain code was not being executed. The biostatistician requested that the ancillary data sets have variables in upper case. Although the AUTOEXEC.sas file included VALIDVARNAME = UPCASE, certain programs either added this SAS System OPTION or %INCLUDE’ed an outdated SETUP.SAS file (that %INCLUDE’ed another file, creating a bit of a web). While examining the ancillary data sets would reveal which did not adhere to the request of the biostatistician rather easily in SAS, for instance, by using SASHELP.VCOLUMN, determine the root causes might require manual review of a subset or all of the programs. The programmatic search and listing of owners saved significant time and resulted in a very short correction of this issue.

CONCLUSION

This paper contributes to data available that programmers, biostatisticians, project managers, and managers can use to audit and reconcile a large number of files in a rapid, auditable, and repeatable manner. The ability to see data such as Last Modified datetime concisely at the time of execution without directly using another program may aid the programmers in understanding issues, like discrepancies that arose because different versions (ages) of a given data set were used. Such knowledge can spare the programmers a fruitless search for logic, algorithm, or code issues, a sometime time-consuming predicament. Having datetimes stamps also assures the user that the relative ages of files of interest are appropriately ordered saves time and, if necessary, quickly allows one to alert other team members with confidence and supporting data easily copied and pasted, which is in contrast to obtaining, for instance, screenshots from the File Explorer after navigating to the files of interest. One can email in SAS under the right configurations, so this paper offers the basis for rapidly and accurately collecting data that a Lead Programmer might need to contact the various programmers responsible for the files of interest. Finally, these macros allow programmatic derivation of variety of metrics of choice, including those to investigate security or process from a high level.

REFERENCES

[1] SAS Institute Inc. 2016. SAS® 9.4 Functions and CALL Routines: Reference, Fifth Edition. Cary, NC: SAS Institute Inc. Page 747. FINFO Function.

[2] Viel, KR. 2023. “SAS® System Macros to Summarize the COMPARE Procedure Results and SAS Logs for a Directory or Single File.” Proceedings of the PharmaSUG 2023 Conference, San Diego, CA: PharmaSUG. https://pharmasug.org/proceedings/2023/SD/PharmaSUG-2023-SD-221.pdf

CONTACT INFORMATION

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

Kevin R. Viel, Ph.D.
Histonis LLC
Navitas Data Sciences
genepistat@gmail.com

Appendix 1. The MAC_U_INFO macro.

```
%macro mac_u_finfo
   ( in_ds                = files
   , pathfile             = %str()
   , keep                 = path
                            file
   , out                  = finfo
   , foptname             = Last Modified
   , foptname_delim       = #
   , print                = Y
   , put_header           = @1   "Path"
                            @120 "File"
                            @160 "Last Modified"
   , put_variables        = @1   path
                            @120 file
                            @160 last_modified
   , delete               = files
                            finfo
   , os_path_delim        = \
   , message_err          = N
   , help                 = N
   ) ;

  %if &help. = Y
  %then
     %do ;
         %let mprint_orig = %sysfunc(getoption(mprint)) ;
         options nomprint ;

         skip ;
         skip ;
         %put ____________________________________________________________________________________________________ ;
         %put Purpose of program:     This utility macro reports the file metadata (FINFO)                       ;
         %put                                                                                                    ;
         %put Macro Parameter         Description                                                                ;
         %put _________________       ___________________________________________________________                ;
         %put in_ds                 = Data set contain path and file or pathfile.                                ;
         %put %str(                            )Default: files                                                   ;
         %put pathfile              = Path and file name of a single file.                                       ;
         %put %str(                            )Default: %str()                                                 ;
         %put keep                  = Variables to keep in the output data set.                                  ;
         %put %str(                            )Default: path                                                   ;
         %put %str(                                    )file                                                   ;
         %put out                   = Output data set                                                            ;
         %put %str(                            )Default: finfo                                                  ;
         %put foptname              = File information to obtain:                                                ;
         %put %str(                            )"Filename"          -> filename                                 ;
         %put %str(                            )"Owner Name"        -> owner                                    ;
         %put %str(                            )"Group Name"        -> group                                    ;
         %put %str(                            )"Access Permission" -> access_permission                        ;
         %put %str(                            )"Last Modified"     -> last_modified                            ;
         %put %str(                            )"Create Time"       -> create_time                              ;
         %put %str(                            )"File Size"         -> file_size                                ;
         %put %str(                            )"File Size (bytes)" -> file_size_bytes                          ;
         %put %str(                            )Default:   Last Modified                                        ;
         %put %str(                                    )# Owner Name                                            ;
         %put foptname_delim        = Delimited for the list of foptname.  This SHOULD NOT be space for certain ;
         %put %str(                         )FOPTNAMEs.                                                        ;
         %put %str(                            )Default: #                                                      ;
         %put print                 = Whether to print the results to the LST file.                              ;
         %put %str(                            )Default: Y                                                      ;
         %put put_header            = The header of the print.                                                   ;
         %put %str(                            )Default: @1   "Path"                                            ;
         %put %str(                                    )@120 "File"                                             ;
         %put %str(                                    )@160 "Last Modified"                                    ;
         %put put_variables         = The variables to print.                                                    ;
         %put %str(                            )Default: @1   path                                              ;
         %put %str(                                    )@120 file                                               ;
         %put %str(                                    )@160 last_modified                                      ;
         %put delete                = Data sets to delete.                                                       ;
         %put %str(                            )Default: files                                                  ;
         %put %str(                                    )finfo                                                   ;
         %put os_path_delim         = The delimiter of the path in the operating system.                         ;
         %put %str(                            )Default:  \                                                     ;
         %put message_err           = Whether to report "ER" "ROR:" if the files is missing.                     ;
         %put %str(                            )Whether to replace "does" " not" " exist" with "dne".           ;
         %put %str(                            )The message is report with or without the words above.          ;
         %put %str(                            )Default:  N                                                     ;
         %put ____________________________________________________________________________________________________ ;
         %put End of help                                                                                        ;

         options &mprint_orig. ;

         %goto __END ;
     %end ;

  %if     &in_ds.               = %str()
      and %nrbquote(&pathfile.) = %str()
  %then
     %do ;
         %put W%str(ARNING: ) IN_DS and PATHFILE cannot both be missing (null) ;
         %goto __END ;
     %end ;
```

```
 93
 94    %if     &in_ds.               ne %str()
 95        and %nrbquote(&pathfile.) ne %str()
 96    %then
 97       %do ;
 98           %put W%str(ARNING: ) IN_DS and PATHFILE both are non-missing.  Please provide a value for only one. ;
 99           %goto __END ;
100       %end ;
101
102    %if     &in_ds.                     ne %str()
103        and %sysfunc( exist( &in_ds. )) =  0
104    %then
105       %do ;
106           %put W%str(ARNING: ) &in_ds. does not exist. ;
107           %goto __END ;
108       %end ;
109
110    %if     %nrbquote(&pathfile.)              ne %str()
111        and %sysfunc( fileexist( &pathfile. )) =  0
112    %then
113       %do ;
114           %put W%str(ARNING: ) &pathfile. does not exist. ;
115           %goto __END ;
116       %end ;
117
118
119    /******/
120    data &out.
121       ( keep = &keep.
122                %if %sysfunc( prxmatch( /Filename/i            , &foptname. )) %then filename           ;
123                %if %sysfunc( prxmatch( /Owner Name/i          , &foptname. )) %then owner              ;
124                %if %sysfunc( prxmatch( /Group Name/i          , &foptname. )) %then goup               ;
125                %if %sysfunc( prxmatch( /Access Permission/i   , &foptname. )) %then access_permission  ;
126                %if %sysfunc( prxmatch( /Last Modified/i       , &foptname. )) %then last_modified      ;
127                %if %sysfunc( prxmatch( /Create Time/i         , &foptname. )) %then create_time        ;
128                %if %sysfunc( prxmatch( /File Size \(bytes\)/i , &foptname. )) %then file_size_bytes    ;
129       ) ;
130
131      length path     $ 256
132             file     $ 256
133             pathfile $ 512
134             foptname $ 100
135             finfo    $ 512
136             msg      $ 512
137             %if %sysfunc( prxmatch( /Filename/i            , &foptname. )) %then filename           $ 512 ;
138             %if %sysfunc( prxmatch( /Owner Name/i          , &foptname. )) %then owner              $  30 ;
139             %if %sysfunc( prxmatch( /Group Name/i          , &foptname. )) %then goup               $  50 ;
140             %if %sysfunc( prxmatch( /Access Permission/i   , &foptname. )) %then access_permission  $  30 ;
141             %if %sysfunc( prxmatch( /Last Modified/i       , &foptname. )) %then last_modified      8     ;
142             %if %sysfunc( prxmatch( /Create Time/i         , &foptname. )) %then create_time        8     ;
143             %if %sysfunc( prxmatch( /File Size \(bytes\)/i , &foptname. )) %then file_size_bytes    8     ;
144             ;
145
146      if _n_ = 1
147      then
148        do ;
149           call missing( path
150                       , file
151                       , pathfile
152                       ) ;
153           __rc = prxparse( "/(.*)\&os_path_delim.([^\&os_path_delim.]*)$/" ) ;
154        end ;
155
156      retain __rc ;
157
158      %if %nrbquote(&pathfile.) = %str() %then set &in_ds. %str(;) ;
159       %else pathfile = "&pathfile." %str(;) ;
160
161      if path ne " "
162      then
163        do ;
164           if prxmatch( "/\&os_path_delim.$/" , strip( path )) = 0
165           then path = cats( path , "&os_path_delim." ) ;
166
167           if pathfile = " " then pathfile = cats( path , file ) ;
168
169        end ;
170
171       else
172         do ;
173            if prxmatch( __rc , trim( pathfile ))
174            then
175              do ;
176                 path = prxposn( __rc , 1 , trim( pathfile )) ;
177                 file = prxposn( __rc , 2 , trim( pathfile )) ;
178              end ;
179         end ;
180
181      rc  = filename( "fn" , pathfile ) ;
182      fid = fopen( "fn" ) ;
183      if fid = 0
184      then
185        do ;
186           msg = sysmsg() ;
187           %if &message_err. = N
```

```
188          %then msg = prxchange( "s/ER%str(ROR: )//i"
189                               , 1
190                               , prxchange( "s/does not exist/dne/i"
191                                          , 1
192                                          , msg
193                                          )
194                               ) %str(;) ;
195          put msg= ;
196          output ;
197       end ;
198
199       else
200        do ;
201          do i = 1 to countc( "&foptname." , "&foptname_delim." ) + 1 ;
202            call missing ( finfo ) ;
203            foptname = strip( scan( "&foptname." , i , "&foptname_delim." )) ;
204
205            finfo    = finfo( fid , foptname ) ;
206
207            select ( foptname ) ;
208              when ( "Filename"          ) filename          = finfo ;
209              when ( "Owner Name"        ) owner             = finfo ;
210              when ( "Group Name"        ) group             = finfo ;
211              when ( "Access Permission" ) access_permission = finfo ;
212              when ( "Last Modified"     ) last_modified     = input( finfo , datetime20. ) ;
213              when ( "Create Time"       ) create_time       = input( finfo , datetime20. ) ;
214              when ( "File Size (bytes)" ) file_size_bytes   = input( finfo , best.       ) ;
215               otherwise put "WAR" "NING: " foptname= finfo= ;
216            end ;
217
218          end ;
219          output ;
220
221          close = fclose( fid ) ;
222
223       end ;
224
225      %if %sysfunc( prxmatch( /Last Modified/i , &foptname. )) %then format last_modified datetime20. %str(;) ;
226      %if %sysfunc( prxmatch( /Create Time/i   , &foptname. )) %then format create_time   datetime20. %str(;) ;
227
228    run ;
229
230    %if &print. = Y
231    %then
232       %do ;
233          data _null_ ;
234            file print ;
235            if _n_ = 1
236            then
237              do ;
238                 put &put_header. ;
239              end ;
240            set &out ;
241            put &put_variables. ;
242          run ;
243      %end ;
244
245    %if &delete. ne %str()
246    %then
247       %do ;
248          %mac_u_delete
249             ( ds = &delete. ) ;
250       %end ;
251
252     %__END:
253
254  %mend mac_u_finfo ;
```

Appendix 2. The MAC_PS_GETCHILDITEM macro.

```
%macro mac_ps_getchilditem
   ( powershell_help = N
   , path            =
   , filter          = -filter *.sas
   , attributes      = -Attributes !Directory
   , extra           = %str()
   , so_property     = Name%str(,)
   , out_ds          = getchilditem_file_owner
   , code1           = %str()
   , code2           = %str()
   , debug           = N
   , help            = N
   ) ;

  %let xwait   = %sysfunc( getoption ( xwait )) ;
  %let xsync   = %sysfunc( getoption ( xsync )) ;
  %let xmin    = %sysfunc( getoption ( xmin  )) ;

  %if &help. = Y
  %then
     %do ;
         %let mprint_orig = %sysfunc(getoption(mprint)) ;
         options nomprint ;

         skip ;
         skip ;
         %put ____________________________________________________________________________________________________________________________________;
         %put Purpose of program:     This (Microsoft Windows) PowerShell macro uses the Get-ChildItem (gci) to obtain the   ;
         %put %str(                        ) filename and owner of files in a path (using the default parameters values).    ;
         skip ;
         %put Macro Parameter         Description                                                                             ;
         %put ____________________    ____________________________________________________________________________________;
         %put path                  = The path of interest.                                                                   ;
         %put filter                = The FILTER parameter and its argument.                                                  ;
         %put %str(                           )Default: -filter *.sas                                                        ;
         %put attributes            = The ATTRIBUTES parameter and its argument.                                              ;
         %put %str(                           )Default: -Attributes !Directory                                               ;
         %put extra                 = Any extra parameter/argument pairs.                                                     ;
         %put %str(                           )Default: %nrstr(%%)str()                                                      ;
         %put so_property           = Arguments following Select-Object -Property                                             ;
         %put %str(                           )Default: Name%nrstr(%%)str(,)                                                 ;
         %put out_ds                = The name of the output data set.                                                        ;
         %put %str(                           )Default: getchilditem_file_owner                                              ;
         %put code1                 = Any SAS code in the data step that creates the output data set, which appears just      ;
         %put %str(                         )statement in the DO-BLOCK of the pattern match.                                 ;
         %put %str(                         )For example: if prxmatch( "/^[tfl]\d+.*\.sas$/i" , strip( file )) then          ;
         %put %str(                           )Default: %nrstr(%%)str()                                                      ;
         %put code2                 = Any SAS code in the data step that creates the output data set, which appears just      ;
         %put %str(                         )before the RUN statement.                                                       ;
         %put %str(                           )Default: %nrstr(%%)str()                                                      ;
         %put debug                 = Whether to drop LINE from the output data set and to put the non-matching LINEs to the ;
         %put %str(                         )log.                                                                            ;
         %put %str(                           )Default: N                                                                    ;
         skip ;
         %put ____________________________________________________________________________________________________________________________________;
         %put End of help                                                                                                     ;

         options &mprint_orig. ;

         %goto __END ;
     %end ;

  options xmin
          xwait
          xsync
          ;

  /*********************************/
  %if &powershell_help. = Y
  %then
     %do ;
         filename process
                  pipe
                  "powershell help get-childitem"
                  ;

         data _null_ ;
           infile process ;
           input ;
           put _infile_ ;
         run ;

         filename process ;

         %goto __END ;
     %end ;

  /*************************************************/
  %if %sysfunc( fileexist( &path. )) = 0
  %then
     %do ;
         %put E%str(RROR: ) PATH = &path. does not exist. ;
```

```
          %goto __END ;
      %end ;

   filename process
            pipe
            %unquote(%str(%")powershell -Command ""get-childitem -path %str(%')&path.%str(%') &attributes. &filter. &extra. | Select-
Object -Property  &so_property. @{Name = %str(%')Owner%str(%') ;Expression = {$PSItem.GetAccessControl().Owner}} "" %str(%"))
            ;

   data &out_ds.
           ( drop   = __rc
                      %if &debug. = N
                      %then line ;
           )
        ;

     length path   $ 500
            file   $ 100
            owner  $ 100
            ;

     retain path "&path." ;

     infile process
            length = len
            ;

     input line $varying5000. len ;

     __rc = prxparse( "/(.*\.[a-z0-9]+)\s+(.*)/" ) ;
     retain __rc ;

     if prxmatch( __rc , trim( line ))
     then
       do ;
          file  = prxposn( __rc , 1 , trim( line )) ;
          owner = prxposn( __rc , 2 , trim( line )) ;
          &code1. output ;
       end ;
      %if &debug. = Y %then else put line= %str(;) ;

     &code2. ;

   run ;

   filename process ;

   %__END:

   options &xwait.
           &xsync.
           &xmin.
           ;

%mend  mac_ps_getchilditem ;
```