ABSTRACT

In many situations, SAS Programmers create, write and validate programs that are to be used by other departments. In most of these cases, the programs have to be dynamic enough to be able to handle all of the different constraints and subsets set forth by the user. Due to this need, often times non-SAS users are accessing validated code to add or remove macro variables that allow for this subsetting. In the wake lies an invalid program, and the integrity of the program is then breached.

This presentation will offer a simple programming solution to this problem, implementing the use of pop-up user windows that allow the user to enter information for the subsetting of data, without actually changing the SAS code in the program. With the use of the WINDOW and DISPLAY statements, and a host of interesting and fun options, a SAS program turns into a controlled, locked and validated piece of code that is both easily usable by users and easily maintainable by the author.

The Concept

At the time that this concept was originated, our data management group was in need of a program that would allow them to change the sent date of queries being sent out to investigational sites. Because these queries were first going through the client, then to the sites, the actual sent dates needed modification.

The standard way of making these changes was to actually access the dataset and physically change the dates for all of the required changes. This was dangerous, as any mistakes could be made and no trace of the actual values could be found. We needed an application that would be friendly enough to accept many different parameters, but also maintain the integrity of the data and the SAS code.

The Solution

The solution was to create a SAS program implementing the use of the WINDOW and DISPLAY statements. WINDOW statements allow the programmer to create pop-up user windows with run-time variables whose value is entered by the user. The DISPLAY statement tells SAS when to put out the windows for data collection. Once the windows have been filled in with data, the fields can then be locked and redisplayed, for verification by the user.

The Code

The Simple Example

* * * * * * * * * * * * * *;
* Display reminder window *;
* * * * * * * * * * * * * *;

data _null_;

**** WINDOW FOR STUDY SELECTION ********;

window select color=blue

#8 @40 "REMINDER" color=white
attr=highlight

#10 @10 "Please run the EXPORT procedure to export the OUTDS dataset to a text file." color=white

#12 @15 "This procedure is defined in the CBIT Formatting Macro Usage Notes." color=white

#20 @35 "Press Enter to continue" color = white;

*** SUBROUTINE FOR WHOLE PROCESS ***;
The Advance Example

/* Have the whole log go to a filename of the same name as the program in the same directory */
proc printto log = 'sndt_chg.log' new;
run;

/* Initialize these macros variables */
%global cont1 cont2 run password;

/* Set up macro variables to define Window attributes */
%let fields = %str(c=blue a=underline);

/* Window background attributes */
%let backgrnd = %str(color=gray);

/* Macro to be called if any window is prematurely extinguished */
/* If a window is closed then SAS will shut down to prevent */
/* the rest of the program to run inaccurately */
%macro endsas(Text);

data _null_; %mend;

/* Data step to set up which study is wanted and set up a macro variable to run the init file */
data _null_;
  length study $1 yesno $1 studname $8;
/* Variable used in the windows */
  *** WINDOW FOR STUDY SELECTION ***;
  window select &backgrnd
    #5 @10 "&text"
    #7 @10 "You have no choice but to press Enter and Exit SAS";
run;

dm 'endsas';
%let rc = %sysfunc(close(&rc1));
* Close the file */
/* Stop printing to the Log */
stop;
run;

/* Stop printing to the Log */
proc printto;
run;

/* terminate SAS */
dm 'endsas';
run;
** WINDOW TO CONFIRM ENTRY **;

    window really &backgrnd
    #4 @10 'You've selected study: ' +1 study &fields; studname p=yes
    #6 @10 'Is this correct (Y/N)? ' +1 yesno &fields p=no;

    * SUBROUTINE FOR WHOLE PROCESS *;

    SELECT:
    display select; /* Display the Study Selection window */

    /* Check for validity of Study response */
    if (study not in('1','2','3','4','5')) then do;
        _msg_ = 'Invalid entry';
        link SELECT;
    end;

    /* If ok then jump to confirm window */
    else do;
        studname=put(study,$study.);
        link REALLY;
    end;
    return;

    REALLY:
    display really; /* Display the Confirmation window */

    /* Confirm if the entry is what is wanted */
    if (upcase(yesno) eq 'Y') then do;
        call symput('study',put(study,$study.));
        call symput('cont1','YES');
        stop;
    end;

    /* If wrong choice is made then go back to study window */
    else if (upcase(yesno) eq 'N') then do;
        link SELECT;
    end;

    /* Otherwise invalid entry */
    else do;
        _msg_ = 'Invalid Entry';
        yesno = '';
        link REALLY;
    end;
    return;

run;

/* This macro checks that the STUDY windows were not closed if the window was closed by the user then call endsas macro to shut down */

%macro check1;
%if (&cont1 ne YES) %then do;
%endsas(You have prematurely closed out the Study Selection Windows);
%end;
%mend;

%check1;

/* Open and lock the requested dataset */
%openlock(usr, <filename>, 1);

/* This whole process is macroitized so it will be able to check if the dataset is in use, since this program overwrites the existing dataset it is necessary for everyone to be out of it */

%macro sndt_chg;
/* These options hide the printing of macro variables in the LOG window */
/* Commented out since the log is being rerouted to a file and the opposite */
/* of these options help debugging any problems easier */
* options nosymbolgen nomprint nomlogic;

/* Window Interface to walk the user through changing Sent Dates in file */
data _null_;

length senddt $8 newdate $8 sitevar $3
patlist $200 statvar $1
okay $5 date1 date2 8.;
format date1 date2 date9.;

*** WINDOW TO GET SENT DATE ***;
  window getsent &backgrnd
    #5 @10 "Please Enter the Current Sent Date that you would like to change"
    #6 @10 "(e.g. MM/DD/YY)-->", +1
senddt &fields;

** WINDOW TO PROMPT FOR NEW SENT DATE **;
  window newdate &backgrnd
    #5 @10 "Please enter the new date you would like to apply (e.g. MM/DD/YY) - .->", +1
newdate &fields;

*** WINDOW TO GET THE SITE ***;
  window getsite &backgrnd
    #5 @10 "Please enter the Investigator Site for these dates -->", +1
sitevar required=YES &fields;

*** WINDOW TO GET PATIENT ***;
  window getpat &backgrnd
    #5 @10 "Please enter the list of Patients you would like to change,"
    #6 @10 "Use a comma to separate the patients and you MUST use quotes,"
    #7 @10 "Or Leave blank to include all patients for the selected site"
    #8 @10 "(e.g. '0001','0002', ...) - .->", +1 patlist &fields;

*** WINDOW TO GET STATUS ***;
  window getstat &backgrnd
    #5 @10 "Please enter the Status for the Sent date you would like to change-->", +1 statvar required=YES &fields;

*** WINDOW TO GENERATE REPORT ***;
  window selected &backgrnd
    #5 @10 "You have chosen the following criteria:
    #7 @15 "Sent Date to change: ", +1 date1 p=yes c=black
    #8 @15 "New Date to apply : ", +1 date2 p=yes c=black
    #9 @15 "For Investigator : ", +1 sitevar p=yes c=black
    #10 @15 "For Patients : ", +1 patlist p=yes c=black
    #11 @15 "For Status : ", +1 statvar p=yes c=black
    #13 @10 "Type OK to generate report or OVER to start over --> ", +1 okay p=no &fields;

*** SUBROUTINES FOR WHOLE PROCESS ***;

SENT:
  display getsent; /* Display window to get the sent date */

  /* Added this step since a missing senddt is ok */
  if (senddt eq '') then do;
    call symput('sentdt', input(senddt,8.));
    goto NEWSENT;
  end;

  /* But if the user enters a date make sure its a valid date */
  else do;

    /* Read the date and see if it is valid */
    date1 = input(senddt, mmddyy8.);

    /* If not valid tell the user so */
    if (_error_ eq 1) then do;
      _msg_ = 'Invalid Date or Format, use MM/DD/YY';
      put _msg_;
      _error_ = 0;
      senddt = '';
      goto SENT;
    end;

    /* If it is valid then create SENTDT macro variable and continue */
    else do;
      call symput('sentdt', date1);
      goto NEWSENT;
  end;
end; /* End checking for a valid date */
return;

NEWSENT:
  display newdate; /* Display window to get new sent date */
/* Read the date see if it is valid */
date2 = input(newdate, mmddyy8.);
/* If not valid then tell the user so */
if (_error_ eq 1) then do;
  _msg_ = 'Invalid Date or Format, use MM/DD/YY';
  put _msg_;
  _error_ = 0;
  newdate = '';
  goto NEWSENT;
end;
/* If it is valid then create NEWDATE macro variable and continue */
else do;
  call symput('newdate', date2);
  goto SITE;
end;
return;

SITE:
  display getsite; /* Display window to prompt for which Site */
/* No confirmation is needed but it is a required variable for the window */
  call symput('site', sitevar);
  goto PAT;
return;

PAT:
  display getpat; /* Display window to get list of Pats */
/* Either keep the entered Patlist or set list to ALL if blank */
  if (patlist eq '') then patlist = 'ALL';
  call symput('patlist', patlist);
  goto STATUS;
return;

STATUS:
  display getstat; /* Display window to prompt for new Status */
  statvar = upcase(statvar);
/* No confirmation is needed but it is a required variable for the window */
  call symput('statvar', statvar);
  goto SELECTED;
return;

SELECTED:
  display selected; /* Display Selections window to confirm */
/* If user confirms then continue on */
  if (upcase(okay) eq 'OK') then goto REPORT;
/* If they don't want to continue then exit window and then go back in to reset the link-logic counter back to 0 */
  else if (upcase(okay) eq 'OVER') then do;
    okay = '';
    goto SENT;
  end;
/* Otherwise invalid entry */
  else do;
    _msg_ = "Please enter 'OK' to continue or 'OVER' to start over"
    put _msg_;
    goto SELECTED;
  end;
return;

REPORT:
  a = round((ranuni(-1)*1000000),1.);
  call symput('password', trim(left(put(a,8.))));
  call symput('cont2', 'YES');
  stop;
run;

/* This macro checks that the SENTDT windows were not closed if the window
was closed by the user then call
endsas macro to shut down */

%macro check2;
  %if (&cont2 ne YES) %then %do;
    %endsas(You have prematurely exited
the Sent Date windows);
  %end;
%end;
%check2;

/* Report section to printout what
changes will take effect */

%macro report;
  /* If they selected ALL patients then
do this data step */
  %if (&patlist eq ALL) %then %do;
    data file;
    set usr.file;
    length oldsent 8.;
    format oldsent sentdt date9.;
    flag = 0;
    oldsent = .;
    /* If they selected ALL the patients
then do this step */
    if (center eq '&site') and (sentdt eq &sentdt) and
    (status eq '&statvar') then do;
      oldsent = sentdt;
      sentdt = &newdate;
      flag = 1;
    end;
  run;
  %end;
  /* Otherwise if they selected some
patients do this step */
  %else %do;
    /* Search the file for criteria that
match what the user selected */
    data file;
    set usr.file;
    length oldsent 8.;
    format oldsent sentdt date9.;
    /* Start everyone off as not having a
flag */
    flag = 0;
    oldsent = .;
    /* If they selected some patients then
do this step */
    if (center eq '&site') and (patid in(&patlist)) and
    (sentdt eq &sentdt) and
    (status eq '&statvar') then do;
      oldsent = sentdt;
      sentdt = &newdate;
      flag = 1;
    end;
  run;
  %end;

  /* Resort the current data step only
keeping the records that
were changed from above */
  proc sort data = file
  out = prtfile;
  by center patid crfpage1;
  where (flag eq 1);
  run;
  /* Check to see if there were any
observations that met the criteria */
  data _null_;
  call symput('numobs',
  put(numobs,8.));
  stop;
  set prtfile nobs=numobs;
  run;
  /* Set up titles */
  title3 "Changes that will be made to
the file based on Selections";
  title4 "&solidln";

  /* Reset the pagename and set up the
printer to be the printer */
  options pageno=1;
  options sysprint='&printer' 'PRINTER
PATH';

  /* If there were observations then
print them out */
  %if (&numobs ne 0) %then %do;
    proc print data=prtfile;
    var center patid crfpage1 query
    oldsent sentdt status;
    footnote1 "&solidln";
    footnote2 "Password to make these
changes in effect &password";
run;
%end;
/* Otherwise show there were no observations */
%else %do;
data _null_;  
  file print;  
  put /// @50 'No observations satisfy criteria'  
    // @40 'SAS will terminate since no changes will be made';  
run;
%end;
/* Print the output to the PRINTER FROM ABOVE and clear the output */
/*  this will make the user actually get and hopefully review the output before putting in the password to continue */
dm output "print; clear" pgm;
/* If there were no observations that fit the criteria specified */
/* Then we do NOT want to continue so let the user know that */
%if (&numobs eq 0) %then %do;
  %endsas(No observations were found with the selections made);
%end;
/* Reset the titles and footnotes */
title3;
footnote1;
%mend report;
/* Call the macro to do all the above steps */
%report;
/* New window to enter the password from the report in order to continue */
data _null_;  
  length pwd $10;  
 *** WINDOW TO GET PASSWORD ***;
  window getpass &backgrnd  
    #6 @10 'Review the Report and If you would like to make the requested changes...'
    #7 @10 'If you choose not to make the changes close this window'
    #9 @10 'Please Enter the Password ->'
          +1 pwd &fields;
  PASS:  
    display getpass;
/* Check to see if password is valid */
/* If so then set macro variable RUN to actually reset the file */
if (pwd eq &password) then do;
  call symsput('run', 'YES');  
  stop;
  end;
/* If not Valid then let the user no so and have them re-enter */
else do;
  _msg_ = 'Incorrect Password Re-Enter';  
  put _msg_;  
  pwd = '';  
  goto PASS;
  end;
return;
run;
/* This macro will update the file if all appropriate steps were made */
%macro runit;
/* If the RUN macro variable from the Password window was set then update the file */
%if (&run eq YES) %then %do;
/* Close the file so it can be written to */
  %let rc = %sysfunc(close(&rc1));
/* Overwrite the file with the current changes */
data usr.file(drop=flag oldsent);  
  set file;
run;
data _null_;
***************  WINDOW FOR FILE IS FINISHED **********************;
    window finish &backgrnd
      #5 @10 "File has been updated with the requested changes"
      #6 @12 "for report with password &password"
      #8 @10 "The audit trail has also been updated"
      #10 @10 'Press Enter to Exit SAS';

    /* Display the window and then close it upon user pressing enter */
    display finish;
    stop;

run;

    /* Stop printing to the log */
    proc printto;
    run;

    /* Exit SAS */
    dm 'endsas';

    %end;

    /* Otherwise if the user did not put in the correct password and closed the window then endsas */
    %else %do;
      %endsas(You have chosen not to update the file with your requested changes);
      %end;
    %end;

    /* Call the macro above to update the file */
    %runit;

%end sndt_chg;   /* This ends the entire program */

/* If the data set is available then run the program, otherwise let the user know that the data set is in use */
%donotdo(sndt_chg);