Using GSUBMIT command to customize the interface in SAS®

Xin Wang, Fountain Medical Technology Co., ltd, Nanjing, China

One of the reasons that SAS is widely used as the statistical analysis tool in analyzing data from clinical trials is that SAS code can be documented and rerun at a later time. As a SAS programmer, during our daily work, we often write lots of small scripts to debug the code or explore the data. These scripts could be used repeatedly in the same programs or in different programs. Here we will introduce one SAS command GSUBMIT which will allows us to customize the SAS interface and run a short SAS program with one click. As a result, those small scripts can be used easily and efficiently.

Keyword: GSUBMIT, CUSTOMIZE

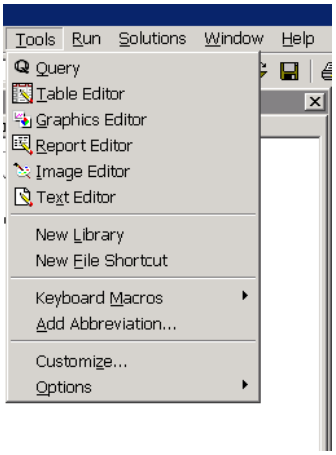
INTRODUCE

Different from other Interactive software, SAS code can be documented, so we can get the track of how SAS processing and analyzing data. But sometimes we may type some codes used for set up SAS environment or debug the program or explore the data which do not need to be documented. These kinds of codes may be used every day. In stand of typing or copying them day and day, GSUBMIT command allows us call them with one click by customizing SAS.

1. CUSTOMIZE TOOL BAR

Tool Bar is a very important part of SAS, it contains many icons which can submit shot commands. Like many other software, the tool bar in SAS can be customized by user easily. Each of the icons, their placement, and their meanings can be customized. New icons that perform your own tasks can be added. Here is the default tool bar:

When customizing it, use the main menus to select TOOLS – CUSTOMIZE or right click on the tool bar itself. The CUSTOMIZE TOOLS dialog box is shown.



❶ Create a new icon, and separator can add an  between the icons.

❷ Delete the task.

❸ Select an icon for your tool.

❹ Move the icon up or down.

❺ Enter your command here which will be executed when click the icon.

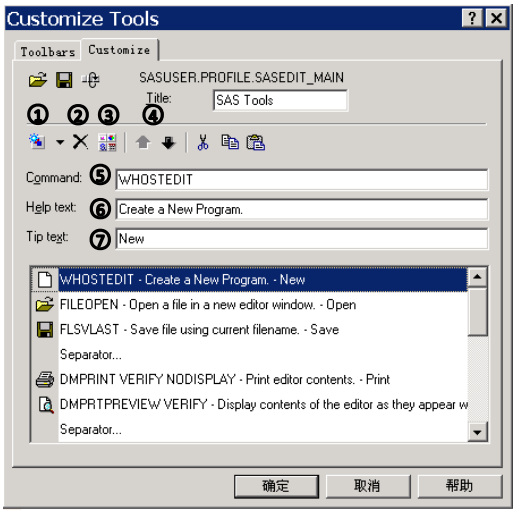
```
GSUBMIT "your SAS code"
```

❻ Short description about the task.

❼ The text which will show up when you put your mouse on the icon.

After you customized your tool bar, you will get it looks like this:

Below are some examples:



EXAMPLE 1

Delete the temporary data set in work library.

```
    GSUBMIT "proc datasets lib=work
nolist kill; quit;"
```

EXAMPLE 2

Close the options for debug macro.

```
    GSUBMIT “options nomlogic
nomprint nosymbolgen;”
```

EXAMPLE 3

The above two examples may be too simple to demonstrate the convenience of tool bar. However, we can use WINDOW statement to add interaction with tool bar. If we submit the below SAS code, SAS will open a window to ask us the name of data set, after we enter the data set name, all its variable names will be print on the log.

```
data _null_;
    window DSIN rows=8 columns=80
    irow=1 icolumn=2 color=black
    #2 @3 'Enter data set name: '
    color=gray dsin $41. required=yes
    attr=underline color=yellow;
    display DSIN blank;
    file log;
        length name $32;
        do dsid = open(dsin,'i') while(dsid ne 0);
            do i = 1 to attrn(dsid,'NVARS');
                name = varname(dsid,i);
                put name @;
            end;
            dsid = close(dsid);
        end;
    stop;
run;
```

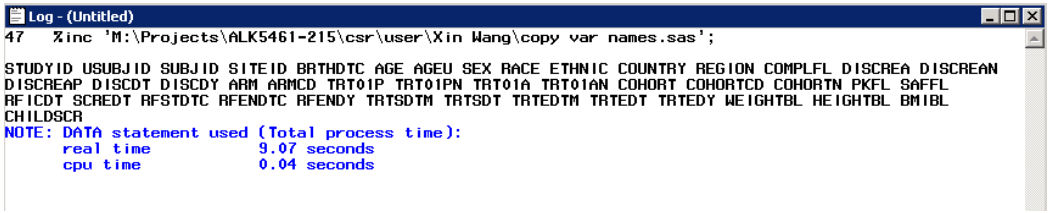
Actually the GSUBMIT command only allows 512 characters. So we cannot submit this program directly. This limitation can be circumvented by submitting a macro or an external program.

E.g. `GSUBMIT “%INC ‘your SAS program path/copy_var_name.sas’”`.

When using GSUBMIT in an Action Command, all the string must be quoted, and %INCLUDE also expects a quoted string or *a fileref*, if your path contains a macro variable, be careful, since strings with strings in macro language are often cause issues.

After you click the button, you will get this window:

Enter the data set name on the line, press enter key, you will see the variable names on your log.



EXAMPLE 4

Here is a very useful SAS program which make us check our log automatically, especially when you have a very long log. It helps us check the information we told it. If we add it on the tool bar by GSUBMIT, we can check the log by one click instead of manually through it. Don't forget to change "*your SAS path*" when you using it.

```
dm 'log; log; file "your SAS path\checklog.log" REPLACE;';
data log_err
     log_war
     log_uni
     log_mer
     log_mis
     log_chr
     log_num;
    infile " your SAS path\checklog.log"
    lrecl=2000 MISSOVER LENGTH=Ll FIRSTOBS=1;
    input a $varying2000. ll;
    if a=:'WARNING: Could not locate style reference' then delete;
    if index(a, 'ERROR:')>0 then output log_err;
    else if index(a, 'WARNING:')>0 then  output log_war;
    else if index(a, 'uninitialized')>0 then  output log_uni;
    else if index(a, 'MERGE statement has more than one data set with
repeats of BY values')>0 then  output log_mer;
    else if index(a, 'Missing values were generated')>0 then  output
log_mis;
    else if index(a, 'Character values have been converted to numeric')>0
then  output log_chr;
    else if index(a, 'Numeric values have been converted to character')>0
then  output log_num;
run;

proc sql noprint;
```

```
    select count(a) into:_logerr  from log_err;
    select count(a) into:_logwar  from log_war;
    select count(a) into:_loguni  from log_uni;
    select count(a) into:_logmer  from log_mer;
    select count(a) into:_logmis  from log_mis;
    select count(a) into:_logchr  from log_chr;
    select count(a) into:_lognum  from log_num;
quit;

data _null_;
    put @1 80*'=';
    put @1 "Checking log";
    put @1 80*'=';
    put @1 "log has &_logerr Error";
    put @1 "log has &_logwar Warning";
    put @1 "log has &_loguni Uninitialized variable";
    put @1 "log has &_logmer Merge by more than one value";
    put @1 "log has &_logmis Missing values were generated";
    put @1 "log has &_logchr Character values converted to numerical";
    put @1 "log has &_lognum Numeric values converted to character";
    put @1 80*'=';
    put @1 '    ';
    put @1 '    ';
run;
```

Obviously, we should use GSUBMIT "%INC 'your SAS *program path*/checklog.sas'", since it is longer than 512 character.

Below is what you will get in your log after you click the button. Instead of go through all the log, you could only read the last part of the log. Please be aware of that before you click the check log button, you should make sure the content in your log window is exactly what you want to be checked.

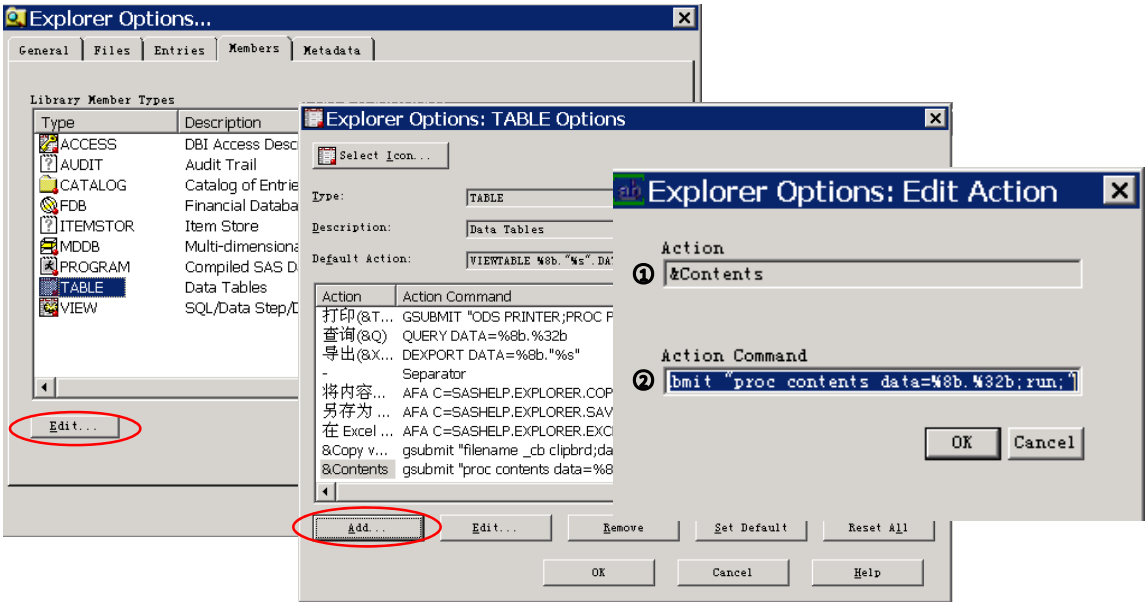
```
================================================================================
Checking log
================================================================================
log has        2 Error
log has        7 Warning
log has        2 Uninitialized variable
log has        0 Merge by more than one value
log has        4 Missing values were generated
log has        0 Character values converted to numerical
log has        0 Numeric values converted to character
================================================================================
NOTE: DATA statement used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds
```

2. CUSTOMIZE PULL DOWN MENUS

Instead of using WINDOW statement to tell SAS which data set should be manipulated, we can also customize the pull down menu to achieve this kind of functions.

Active Explorer window, select TOOLS – OPTIONS – EXPLORER, select MEMBERS tab, you will get

EXPLORERE OPTIONS window. Highlight TABLE line, clicking on the EDIT button, you will get TABLE OPTIONS window.

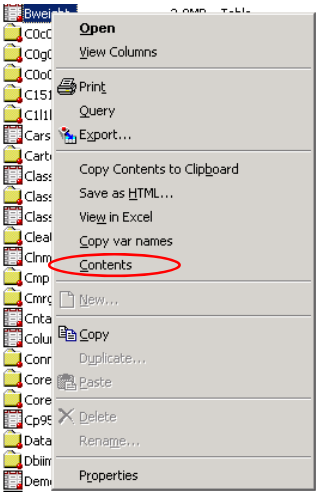


Click ADD button bring you a dialogue box to add your command.

❶ Enter a name for the action. & before one letter will underline this letter.

❷ Enter the action using %8b for the *libref*, and %32b for the data set name.

After done this, when you right click your mouse on one data set, you will see one more action on the list.



EXAMPLE 5

This action will accesses the desired variable names and puts the names into your system's clipboard. After right click your mouse on the data set and select this action, your clipboard will get the variable names of the data, then you can paste them to everywhere you want.

```
GSUBMIT "filename _cb clipbrd;DATA _null_;file _cb;
dsn='%8b'||'.'||'%32b';length name $32;do dsid =
open(dsn,'I') while(dsid ne 0);do i = 1 to attrn(dsid,'NVARS');name =
varname(dsid,i);put name @;end;dsid = close(dsid);end;run;filename _cb
clear;"
```

When you add action commands, removing unnecessary spacing wherever possible, as the action command can only contain up to 255 characters.

CONCLUSION

Hopefully this paper can give you an inspiration on how to use GSUBMIT to customize SAS interface. You can develop your own action which can made your daily work convenience.

REFERENCE

Arthur L. Carpenter. 2012. Doing More with the SAS® Display Manager: From Editor to View Table - Options and Tools You Should Know. Orlando Florida, US: SAS Global Forum.

ACKNOWLEDGEMENTS

I would like to thank Chao Wang for his timely efforts in reviewing and providing valuable feedback on this paper.

CONTACT INFORMATION

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

Name: Xin Wang
Enterprise: Fountain Medical Technology Co., ltd
Address: Room 403, Building 43, No.70, Headquarter Base, Phoenix Road, Jiangning District
City, State ZIP: Nanjing, China 210000
E-mail: xin.wang@fountain-med.com