

# To Macro or Not...that is the Question

Claudine Lougee, Dualenic, LLC

## ABSTRACT

Do you need a macro for your program? How do you know if it's worth the time to create one for your program? This paper will provide some guidelines, based on user experience, if it's worth the time to create a macro whether it's parameter driven macro or just a simple macro variable. This paper is geared towards new users and maybe experienced users who do not use macros.

## INTRODUCTION

This paper will consider the basic question, "It is worth the time to create a SAS® macro for a program?" Macros can be written for many reasons. It is up to the programmer to decide if it is worth their time, effort, and skill to create a macro for a program. There are several ways to write macros. The paper explores simple macro options to consider and does not teach the use of macros. The examples provided in this paper can be used by new programmers or experienced programmers.

## SIMPLE MACRO VARIABLE

A simple macro variable is a way to add text to a program which may be changed by different users. For example, a location of the SAS library or the output location of a file is a great use of a macro.

```
%let userloc=C:\lougee\SESUG;

ODS RTF FILE="%userloc.\class_report.rtf";
PROC PRINT DATA=SASHELP.CLASS;
RUN;
ODS RTF CLOSE;
```

In this example, the report user location is used as a simple macro variable to create the ODS output. The macro variable, *userloc*, could be used several times throughout the program. If a different user wanted to use the same code, they could easily change the macro variable to the different location and run the program with ease.

If the program had several reports and thousands of lines of code, it would be very useful then having to read through the code or do a find and replace several times and hope to capture all the right places to change within the code.

## %LET STATEMENT

The %LET statement can be used for system generated or user-generated macro variables. Adding the existing system generated macro, *sysdate*, to the code will add the system date to the file name. System generated or automatic macros can be seen in the log with the statement **%put \_automatic\_**; Global or user-generated macro can be found with **%put \_global\_**;

```
%let date=&sysdate.;

ODS RTF FILE="%userloc.\class_report_&date.rtf";
PROC PRINT DATA=SASHELP.CLASS;
RUN;
ODS RTF CLOSE;
```

System macros can be used without the let statement and resolve. There would be no need to create a macro variable called "date" by using the system date.

## ADDING A DATE STAMP TO THE LOG

In this example, a log is created for review every time the program is executed. If the program runs once a day and you want to review the output each time the program executes, a date stamp may be a nice addition.

```
proc printto log="%userloc.\logresults_&sysdate9..txt" new;
run;
proc printto print="%userloc.\printresults_&sysdate9..txt" new;
run;
```

**Note of caution:** some programmers do not shut down their programs or computer at night before going home. If you are one of those programmers, the sysdate is the date when SAS was initiated. If you want the today's date, you can use following code which generates the actual date and not the date that SAS was opened.

```
%let date=%TRIM(%SYSFUNC (DATE() , date7.));
```

## PARAMETER DRIVEN MACRO

The parameter driven macro can be used when multiple items may need to change within code. If the same report needed to be generated for only a certain age, a parameter driven macro may help if this changes often.

### %GETAGE MACRO

```
%let userloc=C:\lougee\SESUG;
%let folder=Reports;
%let rptname=class_report;
%let date=%TRIM(%SYSFUNC (DATE() , date7.));

%macro GETAGE(age);
ODS RTF FILE= "&userloc.\&folder.\&rptname._&date..rtf";
Title1 "Where Age is &age.";
PROC PRINT DATA=SASHELP.CLASS;
Where age =&age.;
RUN;
ODS RTF CLOSE;
%mend;
```

```
%GETAGE(11)
```

The macro code will only print the age specified in the macro call.

### *Where Age is 11*

Obs	Name	Sex	Age	Height	Weight
11	Joyce	F	11	51.3	50.5
18	Thomas	M	11	57.5	85.0

### Output 1. Output from a GETAGE macro

## WHEN TO USE A MACRO

Here are some basic questions to ask yourself when deciding to create a macro:

1. Do I need a macro?
2. Do I know how to create the right code for a macro variable?
3. Is this code I will run again later using similar techniques?
4. Will someone else inherit this code later?
5. Will it help me later if there is a macro within this code?

If you answered at least 3 of the above questions 'yes', then I would suggest you go for it!

Here are some basic answers to the above questions:

1. Even if you don't need a macro, a macro can be fun to create and learn if you don't know how
2. Everyone creates macro differently, no two macros are created equally (unless you borrowed code)
3. All code can be reused in some way (borrow, borrow, borrow)
4. Even if you nobody inherits your code, someone may borrow your code
5. I've never found a macro to hinder the process

Here are some additional reasons to create a macro:

- If the SAS program is repeating a process and needs input and output locations
- If a date or time stamp on the datasets or the report or log is helpful to you or others
- When consistency across the department in code techniques or naming convention helps
- If someone is not familiar with the code but needs to run it

## CONCLUSION

In summary, it is up to the programmer to decide if the macro helps or hinders the process. It is always good practice to set the macros in the beginning of the program if the macro variable is going to change every time the program is run. It is not good practice to add macros inside of macros which will need to change in the middle of the program every time. No programmer should add macros so elaborate without documentation explaining what the macro is doing. Macros can be used for programs which need little to no maintenance or used as a means create logs or time stamps on data input or program output.

## ACKNOWLEDGMENTS

Thank you SESUG 2015 Executive Committee, Section Chairs, volunteers and attendees for the opportunity to publish and present this paper. Thanks to all mentors, presenters, and instructors who provided me with material for this paper. Thanks to SAS® for allowing these conferences to continue to exist and serve the user group community.

## RECOMMENDED READING

- *SAS Macro Guide Made Easy* (Michele Burlew)
- *Carpenter's Complete Guide to the SAS Macro Language* (Art Carpenter)
- *The Little SAS Book* (Delwiche and Slaughter)

## CONTACT INFORMATION

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

Claudine Lougee  
Dualenic, LLC  
PO Box 1108  
Glen Allen, VA 23060  
724-467-0559  
claudine@dualenic.com  
[www.dualenic.com](http://www.dualenic.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.