Paper SAS302-2014

Introduction to SAS® Studio

Michael A. Monaco, Marie Dexter, Jennifer Tamburro

SAS Institute Inc., Cary, NC

ABSTRACT

This paper is an introduction to SAS® Studio and covers how to perform basic programming tasks in SAS Studio. Many people program in the SAS® language by using SAS Display Manager or SAS® Enterprise Guide®. SAS Studio is different because it enables you to write and run SAS code by using the most popular web browsers, without requiring a SAS® 9.4 installation on your machine. With SAS Studio, you can access your data files, libraries, and existing programs, and write new programs while using SAS software behind the scenes. SAS Studio connects to a SAS server in order to process SAS programs. The SAS server can be a hosted server in a cloud environment, a server in your local environment, or a copy of SAS on your local machine.

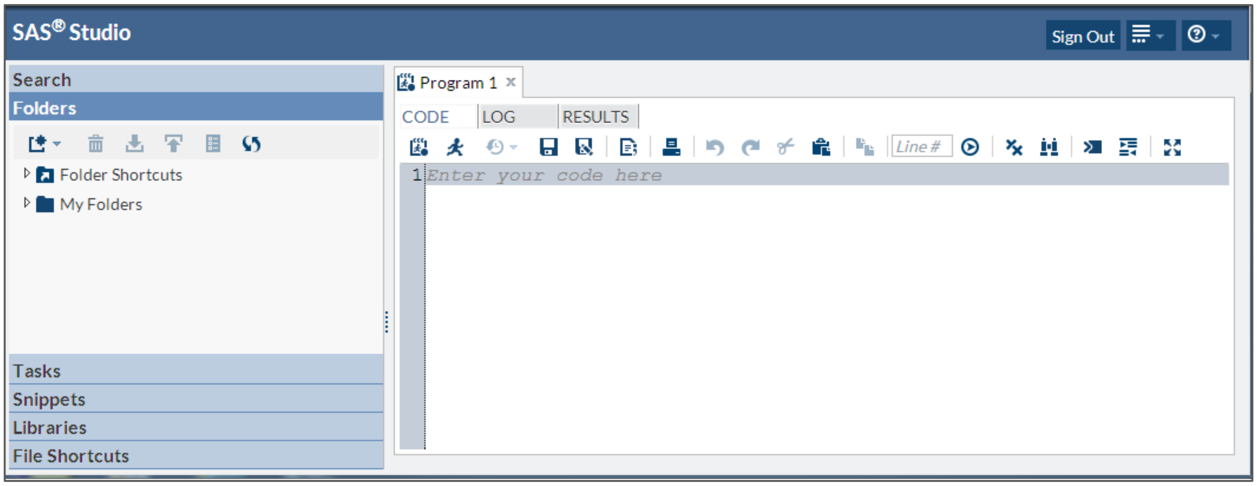
INTRODUCTION

SAS Studio is a pure HTML5 application that requires no browser plug-ins and supports all the major web browsers, such as Microsoft Internet Explorer 9, 10 and 11, Apple Safari 6+, Mozilla Firefox 21+, and Google Chrome 27+. The browser-based interface enables SAS Studio to run on the iPad as well as laptop or desktop computers. In most deployments to run SAS Studio, you simply go to a URL and log on. This paper covers the following topics:

- Elements of the SAS Studio Interface
- Programming with SAS Studio
- Working with Data in SAS Studio
- Introduction to the Common Task Model and Authoring SAS Studio Tasks

ELEMENTS OF THE SAS STUDIO INTERFACE

When you log on to SAS Studio, the main SAS Studio window appears.



The main window of SAS Studio consists of a navigation pane on the left and a work area on the right. The navigation pane provides access to the search feature, your folders and folder shortcuts, your tasks and snippets, the libraries that you have access to, and your file shortcuts. The Folders section is displayed by default.

The work area is used to display your data, code, logs, and results. When you first open SAS Studio, the work area displays a new program window. When you open data and run tasks, other windows open in the work area with a tabbed interface.

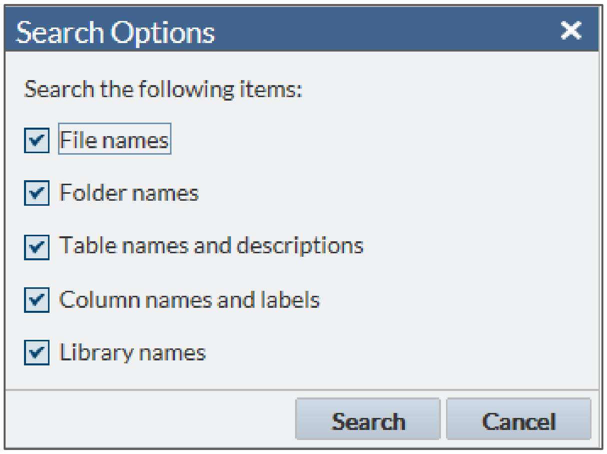
USING THE NAVIGATION PANE

You can expand the sections of the navigation pane by clicking the section that you want to view.

Using the Search Pane

The Search section of the navigation pane enables you to search your directories for filenames, folder names, table names and descriptions, column names and labels, and library names. You can open items directly from your search results by double-clicking them or using a drag-and-drop operation to move them to the work area.

1. In the navigation pane, click **Search** to open the Search section.
2. In the search box, enter the text that you want to search for.
3. Click to open the Search Options window and specify the types of items that you want to search. By default, all types of items are searched.



Search Options

Search the following items:

☑ File names

☑ Folder names

☑ Table names and descriptions

☑ Column names and labels

☑ Library names

4. Click **Search**.

Using the Folders Pane

The Folders section of the navigation pane enables you to access your folders, create folder shortcuts, download and upload files, and create new SAS programs. You can expand and collapse folders, and you can open items in the folders, such as a SAS program or table, by double-clicking them or by using a drag-and-drop operation to move them to the work area.

To create a new folder shortcut, click and select **Folder Shortcut**. Enter the shortcut name and full path to the folder location, and then click **Save**. The new shortcut is added to the list of folder shortcuts.

The Folders section of the navigation pane references files that are stored on the same machine as the SAS server. These files can be downloaded to your workstation. Local files on your workstation must be uploaded to the server in order to be used with SAS Studio.

To download a file, select the file that you want to download and click . You are prompted to open the file in the default application or save it to your local computer.

To upload one or more files from your local computer, click . Specify the folder to which you want to upload the files, and click **Choose Files** to browse for the files that you want to upload.

Note that all files are stored on the same machine as the SAS server. Local files on your workstation must be uploaded to the server in order to be used with SAS Studio.

Using the Tasks Pane

The Tasks section of the navigation pane enables you to access tasks in SAS Studio. Tasks are based on SAS procedures and enable you to generate SAS code and formatted results, based on the entry of user-supplied inputs and parameter values. SAS Studio is shipped with several predefined tasks that you can run. You can also edit a copy of a predefined task, and you can create your own new tasks.

To create a new task, click . SAS Studio creates a template in the work area that you can modify to develop custom tasks for your site. Custom tasks can be accessed from the My Tasks folder. For more information, see the “Introduction to the Common Task Model and Authoring SAS Studio Tasks” section in this paper.

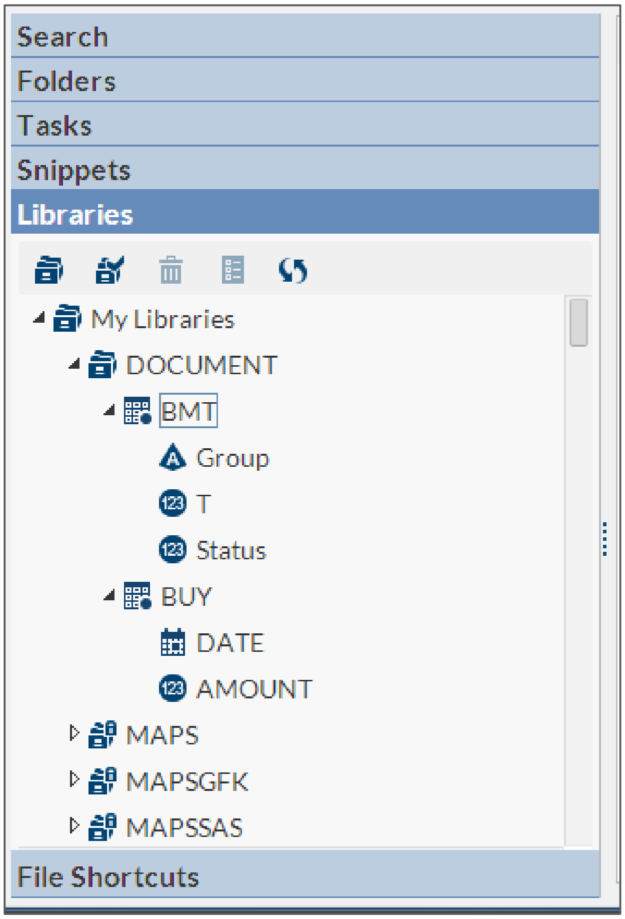
To edit a task that you have created, select the task from the My Tasks folder and click . The XML code that is used to create the task is opened in the work area. If you want to edit a predefined task, you must first right-click the task and select **Add to My Tasks**. Note that you can edit only the tasks that are in the My Tasks folder.

Using the Snippets Pane

The Snippets section of the navigation pane enables you to access your code snippets, which are samples of commonly used SAS code that you can insert into your SAS program. SAS Studio is shipped with several predefined code snippets that you can use. You can also save copies of these snippets and edit them to create your own custom snippets.

Using the Libraries Pane

The Libraries section of the navigation pane enables you to access all of your SAS table libraries and their contents. You can expand a table and view the columns in that table. The icon in front of the column name indicates the data type of the column. In addition to viewing the column names, you can use a drag-and-drop operation to move the columns from the tree directly into your SAS code.



Using the File Shortcuts Pane

File shortcuts, also known as SAS File References, enable you to quickly access files. You can create a file shortcut to a file on your SAS server or via a URL.

To create a new file shortcut, expand the File Shortcuts section of the navigation pane and click ↗. You can define the shortcut by specifying a complete path and filename or by specifying a URL. If you want this shortcut to be available the next time you use SAS Studio, select **Re-create this file shortcut at start-up**.

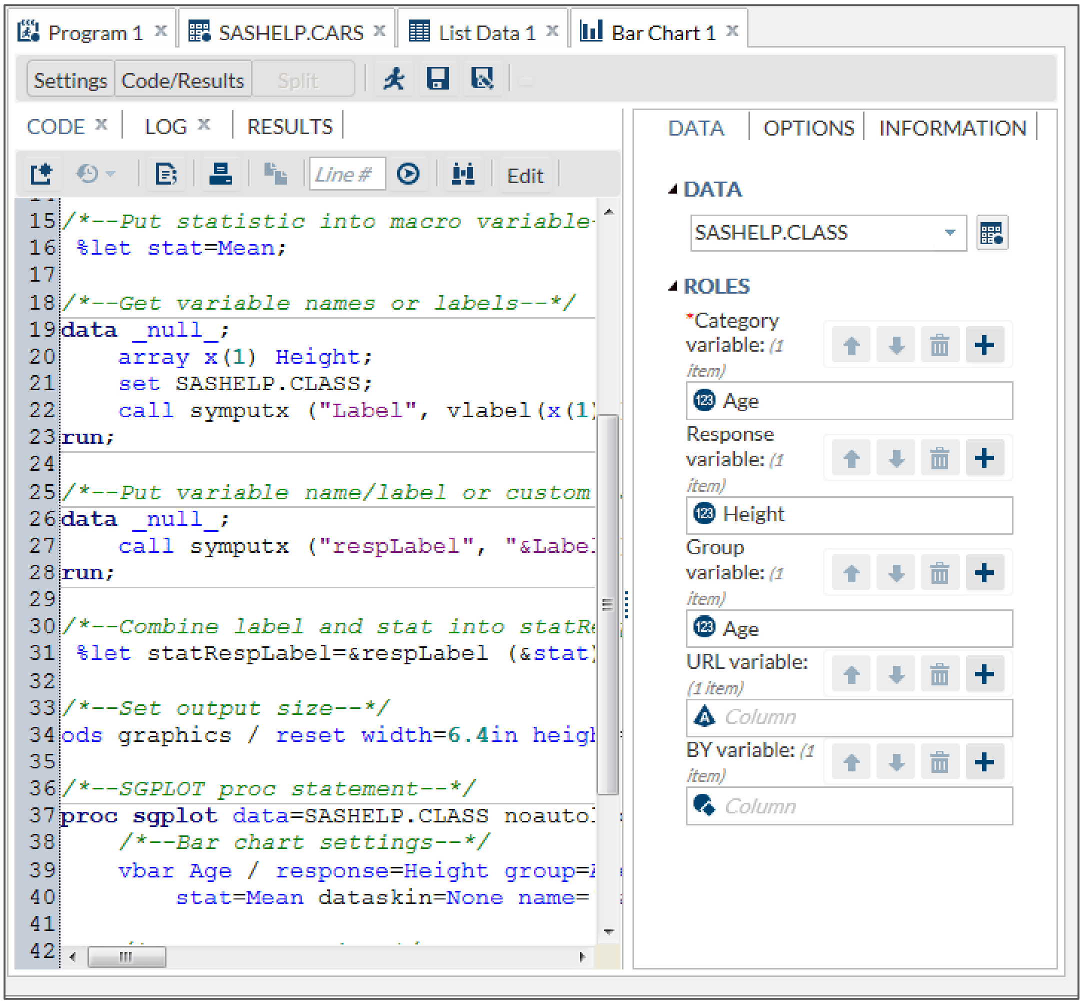
You can open a file from a file shortcut by double-clicking it or by using a drag-and-drop operation to move it to the work area.

Customizing the Navigation Pane

By default, all six sections of the navigation pane are displayed when you open SAS Studio. To customize which sections are displayed, click ☰ in the upper right corner of the application and select **View**. Select or clear any sections that you want to add or remove. The navigation pane is updated immediately.

USING THE WORK AREA

The work area is the main portion of the SAS Studio application for accessing programs and tasks and for viewing data. The work area is always displayed and cannot be minimized. When you open a program, task, or table, the windows open as new tabs in the work area. The code, log, and results that are associated with programs and tasks are grouped together under the main tab for the program or task.



```
/*--Put statistic into macro variable-
 %let stat=Mean;

/*--Get variable names or labels--*/
data _null_;
    array x(1) Height;
    set SASHELP.CLASS;
    call symputx ("Label", vlabel(x(1)
run;

/*--Put variable name/label or custom
data _null_;
    call symputx ("respLabel", "&Label
run;

/*--Combine label and stat into statRe
 %let statRespLabel=&respLabel (&stat)

/*--Set output size--*/
ods graphics / reset width=6.4in heigh

/*--SGPLOT proc statement--*/
proc sgplot data=SASHELP.CLASS noautol
    /*--Bar chart settings--*/
    vbar Age / response=Height group=
        stat=Mean dataskin=None name=
```

DATA

SASHELP.CLASS

ROLES

*Category variable: (1 item)
Age

Response variable: (1 item)
Height

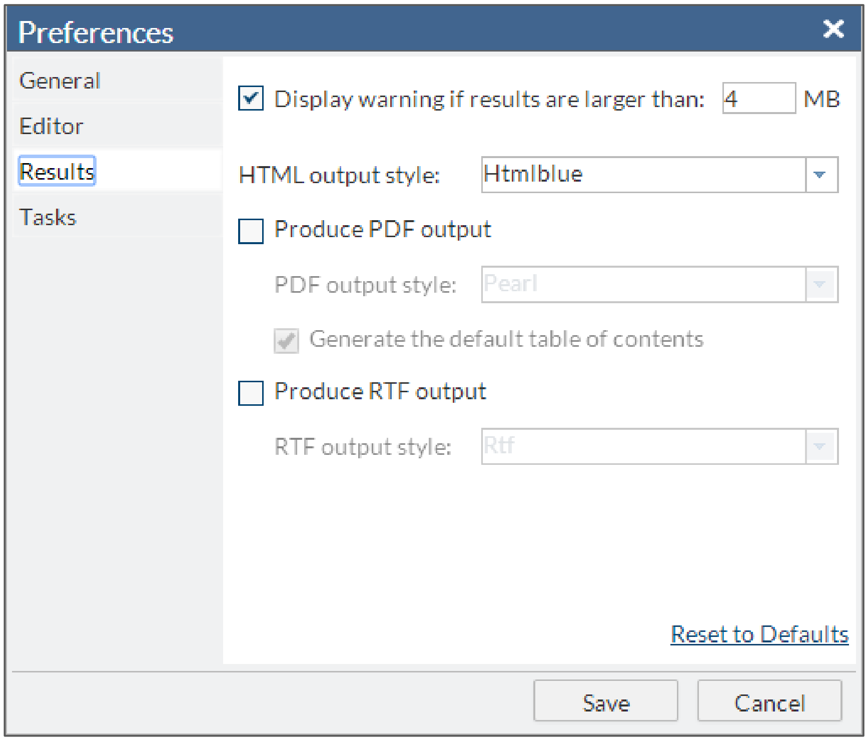
Group variable: (1 item)
Age

URL variable: (1 item)
Column

BY variable: (1 item)
Column

SETTING PREFERENCES

The Preferences window enables you to change several options that affect SAS Studio. To access the general options, click ☰ in the upper right corner of the application and select **Preferences**.



☑ Display warning if results are larger than: 4 MB

HTML output style: Htmlblue

□ Produce PDF output

PDF output style: Pearl

☑ Generate the default table of contents

□ Produce RTF output

RTF output style: Rtf

PROGRAMMING WITH SAS STUDIO

About the Code Editor

SAS Studio includes a color-coded, syntax-checking SAS language editor for editing new or existing SAS programs. The editor includes a wide variety of features such as autocomplete, automatic formatting, and pop-up syntax help. With the code editor, you can write, run, and save SAS programs. You can also modify and save the code that is automatically generated when you run a task.

SAS Studio also includes several sample code snippets that you can use to make programming common tasks easier.

Opening and Creating Programs

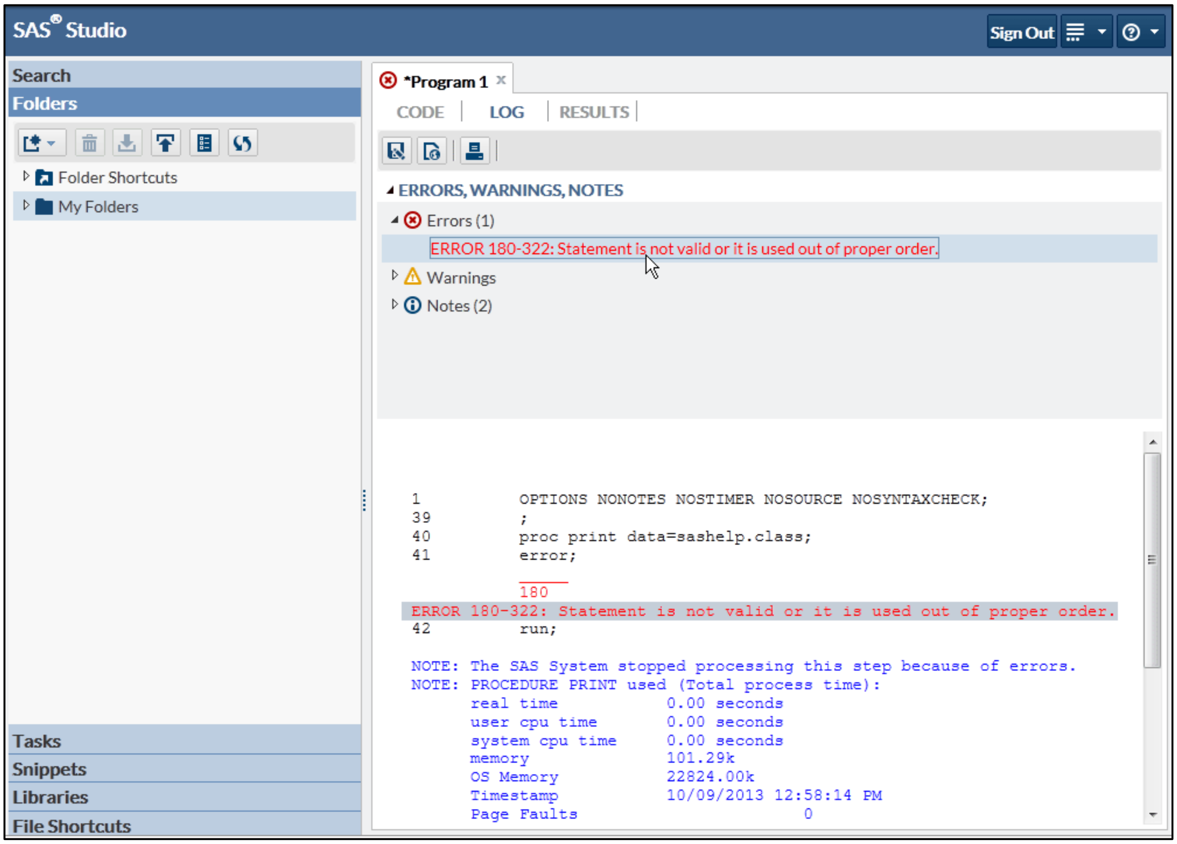
You can open SAS programs from the Folders section of the navigation pane. To open a program, expand the appropriate folder and double-click the program that you want to open, or use a drag-and-drop operation to move it to the work area. The program opens in a new tab in the work area.

You can create a new SAS program from the Folders section of the navigation pane. To create a new program, click 🗋 and select **SAS Program**. A program window appears in a new tab in the work area.

Note: You can also click 🗋 on the toolbar in a program window or press F4.

Running a Program

After you have written your program, you can run it by clicking 🏃 or pressing F3. If there are no errors, the results open automatically. If there are errors, the **Log** tab opens by default. You can expand the **Errors**, **Warnings,** and **Notes** sections to view the messages. When you click on a message, SAS Studio highlights it for you in the log so that you can see exactly where the message occurs in the log.

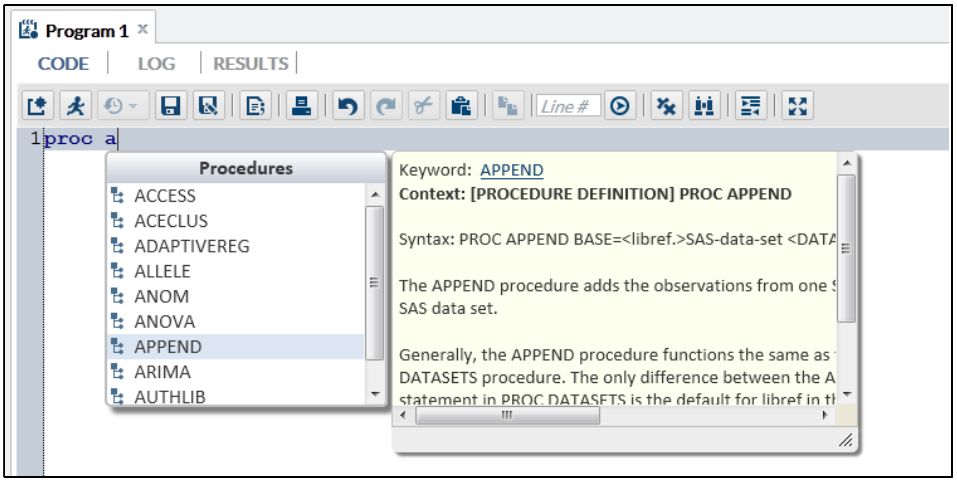


ERROR 180-322: Statement is not valid or it is used out of proper order.

```
1          OPTIONS NONOTES NOSTIMER NOSOURCE NOSYNTAXCHECK;
39         ;
40         proc print data=sashelp.class;
41         error;
           _____
           180
ERROR 180-322: Statement is not valid or it is used out of proper order.
42         run;

NOTE: The SAS System stopped processing this step because of errors.
NOTE: PROCEDURE PRINT used (Total process time):
      real time           0.00 seconds
      user cpu time       0.00 seconds
      system cpu time     0.00 seconds
      memory              101.29k
      OS Memory           22824.00k
      Timestamp           10/09/2013 12:58:14 PM
      Page Faults                       0
```

The autocomplete, or code completion, feature in the code editor can predict the next word that you want to enter before you actually enter it completely. The autocomplete feature can complete keywords that are associated with SAS procedures, statements, macros, functions, CALL routines, formats, informats, macro variables, SAS colors, style elements, style attributes, and statistics keywords, and various SAS statement and procedure options.

This example shows the keywords and help that appear when you enter `proc a` in the code editor.



```
1 proc a
```

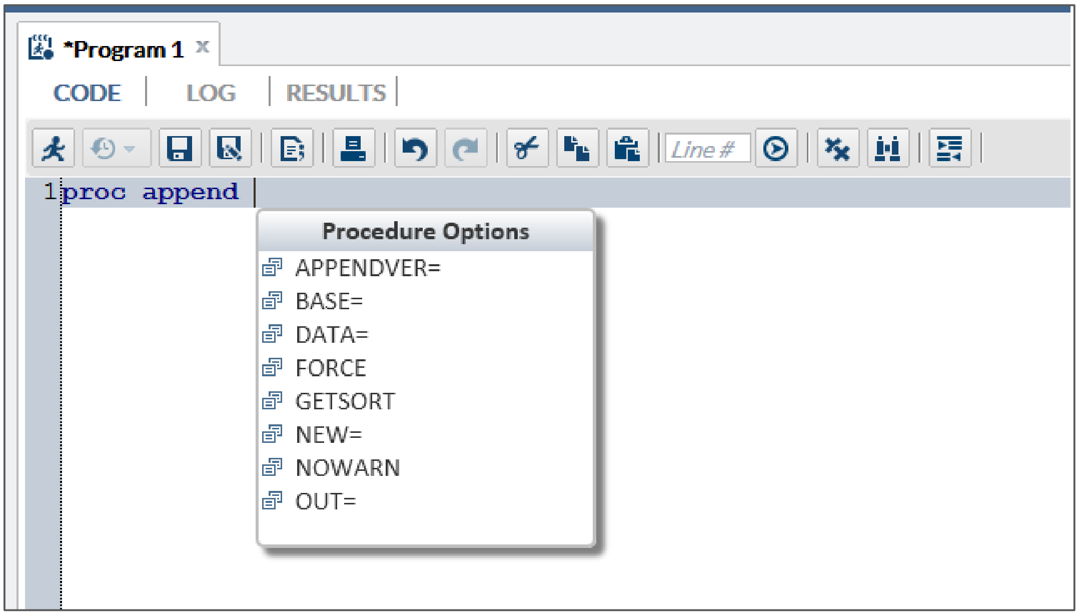
Keyword: APPEND
Context: [PROCEDURE DEFINITION] PROC APPEND

Syntax: PROC APPEND BASE=<libref.>SAS-data-set <DATA

The APPEND procedure adds the observations from one S
SAS data set.

Generally, the APPEND procedure functions the same as t
DATASETS procedure. The only difference between the A
statement in PROC DATASETS is the default for libref in th

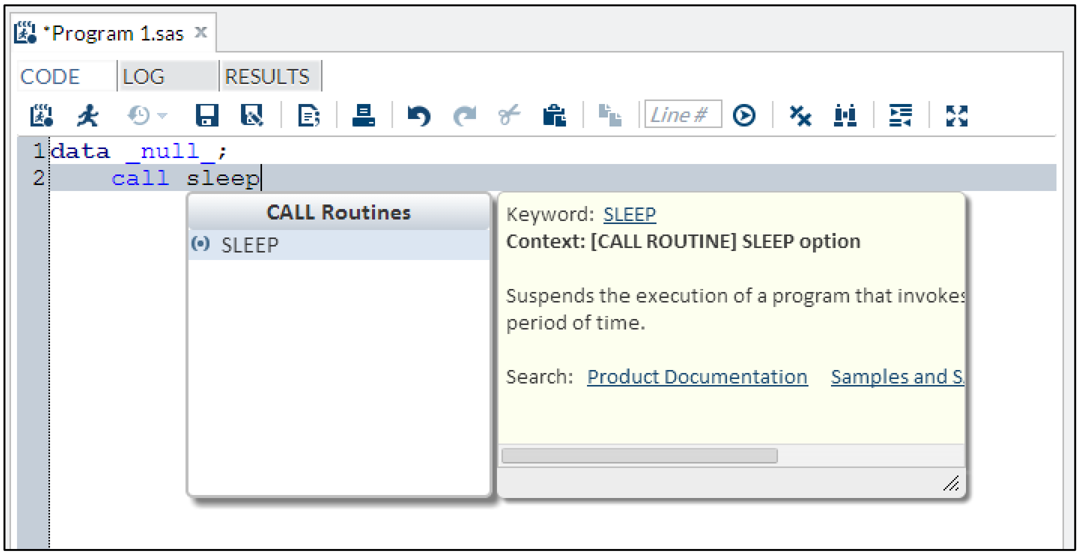
In this example, you select **APPEND** from the list of procedures so that `proc append` appears in the code editor. When you enter a space, the code editor displays a list of options for the APPEND procedure.



```
proc append
```

How you open the autocomplete list depends on the keyword that you want to add.

If you want to add a global statement, DATA step statement, CALL routine, procedure, macro statement, or automatic macro variable, enter the first one or more letters of the keyword that you want to use. A window opens with a list of suggested keywords that begin with those letters.



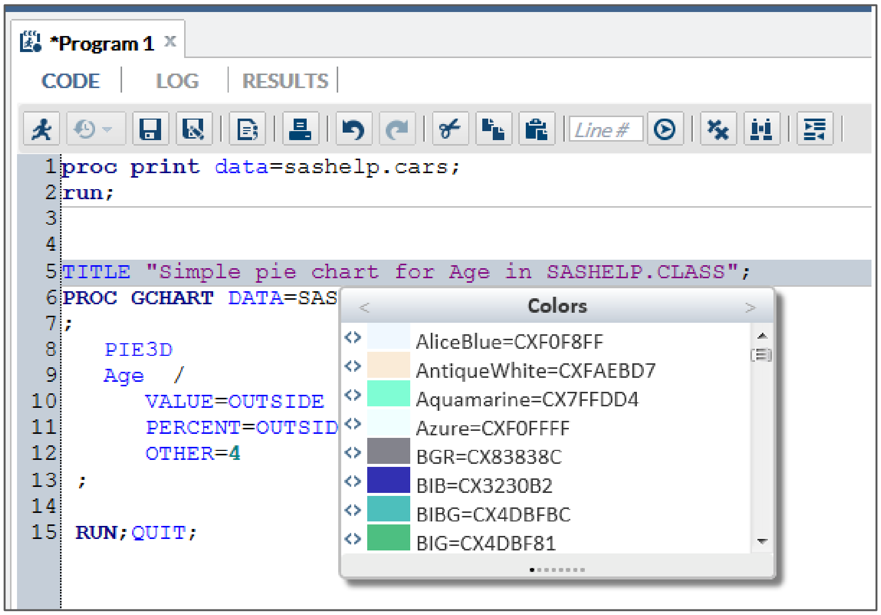
```
data _null_;
     call sleep
```

Keyword: SLEEP
Context: [CALL ROUTINE] SLEEP option

Suspends the execution of a program that invokes
period of time.

If you want to specify colors, formats, informats, macro functions, SAS functions, statistics keywords, style elements, or style attributes, position your mouse pointer in a comment and press Ctrl+spacebar. To navigate through the list of options backward, press Ctrl+Shift+spacebar.

Note: The code completion feature can be turned off in the Preferences window. In the Preferences window, click **Editor** and deselect **Enable autocomplete**. Even if you have deselected the Enable autocomplete option, these keyboard shortcuts are available.



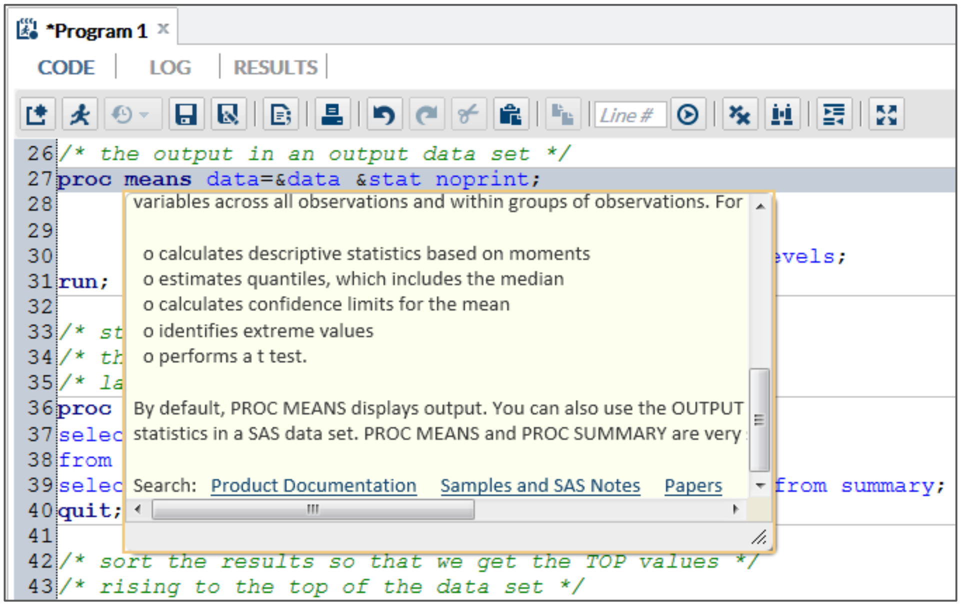
```
proc print data=sashelp.cars;
run;


TITLE "Simple pie chart for Age in SASHELP.CLASS";
PROC GCHART DATA=SAS
;
   PIE3D
   Age   /
      VALUE=OUTSIDE
      PERCENT=OUTSID
      OTHER=4
 ;

 RUN;QUIT;
```

Colors

- AliceBlue=CXF0F8FF
- AntiqueWhite=CXFAEBD7
- Aquamarine=CX7FFDD4
- Azure=CXF0FFFF
- BGR=CX83838C
- BIB=CX3230B2
- BIBG=CX4DBFBC
- BIG=CX4DBF81

The code editor displays brief SAS syntax documentation as you write and edit your programs. You can display the Help by right-clicking a keyword in your program and selecting **Syntax Help,** or simply start entering a valid SAS keyword and then click a suggested keyword in the autocomplete window.

The SAS Product Documentation provides more comprehensive usage information about the SAS language, but the syntax help in the code editor can get you started with a hint about the syntax or a brief description of the keyword. Click the links at the bottom of the window to search for the keyword in the SAS Product Documentation, Samples and SAS Notes, and SAS Technical Papers.



```
/* the output in an output data set */
proc means data=&data &stat noprint;


                                                       evels;
run;

/* st
/* th
/* la
proc
selec
from
selec                                                  from summary;
quit;

/* sort the results so that we get the TOP values */
/* rising to the top of the data set */
```

variables across all observations and within groups of observations. For

o calculates descriptive statistics based on moments
o estimates quantiles, which includes the median
o calculates confidence limits for the mean
o identifies extreme values
o performs a t test.

By default, PROC MEANS displays output. You can also use the OUTPUT
statistics in a SAS data set. PROC MEANS and PROC SUMMARY are very

Search: Product Documentation Samples and SAS Notes Papers

You can use the code editor to make your programs easier to read by automatically formatting your code. When you automatically format your code, line breaks are added, and each line is correctly indented according to its nesting level. To format the code in the code editor, click ☰.

For example, the following code is difficult to read because it lacks indention and logical line breaks:

After you use the automatic code-formatting feature, the program looks like this:

```
data topn;
length rank 8; label rank="Rank";
set topn; by &category descending &measure;
if first.&category then rank=0; rank+1;
if rank le &n then output;
run;
```

```
data topn;
     length rank 8;
     label rank="Rank";
     set topn;
     by &category descending &measure;

     if first.&category then
         rank=0;
     rank+1;

     if rank le &n then
         output;
run;
```

SAS Studio also maintains a log with entries for each time you run a program or task. You can use this log, or submission history, to access prior versions of your submitted code. To view your submission history, click the **Code** tab in your program or task window. On the toolbar, click and select the version that you want to open. The prior version of the program opens in a new window from which you can copy and paste the code as needed. This functionality replaces the RECALL functionality in SAS Display Manager.

Working with Results

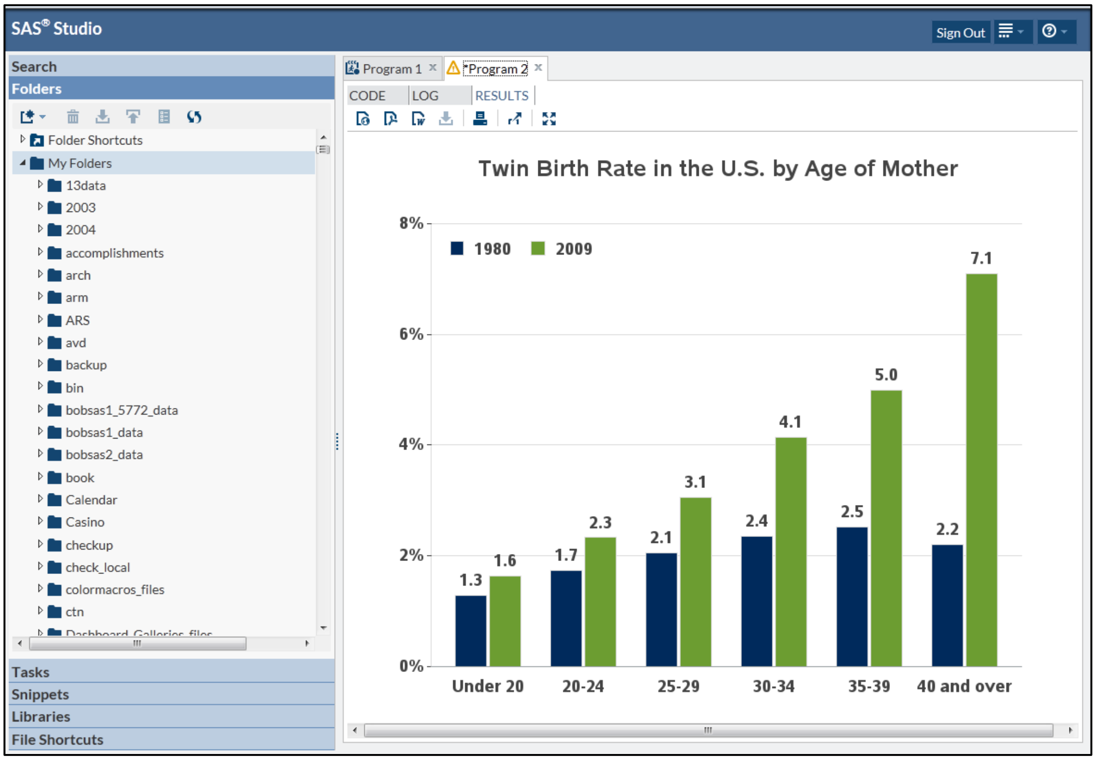
When you run a task or a program in SAS Studio, the results are displayed in the work area. You can save the results as an HTML, PDF, or RTF file. You can also download any generated data. The SAS Output Delivery System (ODS) used by SAS Studio gives you greater flexibility in generating, storing, and reproducing SAS procedure and DATA step output along with a wide range of formatting options. ODS provides formatting functionality that is not available when using individual procedures or the DATA step without ODS. To see the default output destinations, simply submit some basic SAS code to produce output.

```
proc print data=sashelp.cars;run;
proc gtestit;run;
```

SAS Studio uses very specific ODS options and the GOPTIONS statements so that the output displays properly in the web environment. To view all of the ODS options in your code, click and select **Preferences**. In the Preferences window, click **General** and select the **Show generated code in the SAS log** option.

Note: To ensure that your output displays properly, use great care when explicitly setting ODS options or GOPTIONS statements in your code.

The Preferences window enables you to change several options that affect how your results are displayed. To access the editor option, click in the upper right corner of the application, select **Preferences**, and then click **Results.** This option enables you to set the HTML, PDF, and RTF output style. It also enables you to choose whether PDF and RTF output will automatically be generated. Generating PDF and RTF output might increase execution run-time performance for complex output.

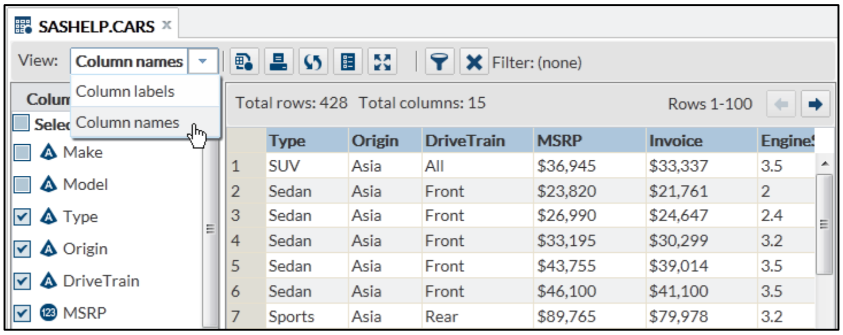


WORKING WITH DATA IN SAS STUDIO

You can access tables in SAS Studio in several ways:

- You can double-click a file in the **Folders** and **Libraries** sections.
- You can use a drag-and-drop operation to move a file from the **Folders** and **Libraries** sections to the work area.
- You can search for a file in the **Search** section and open it from the search results. You can open the file by double-clicking it or by using a drag-and-drop operation to move it to the work area.

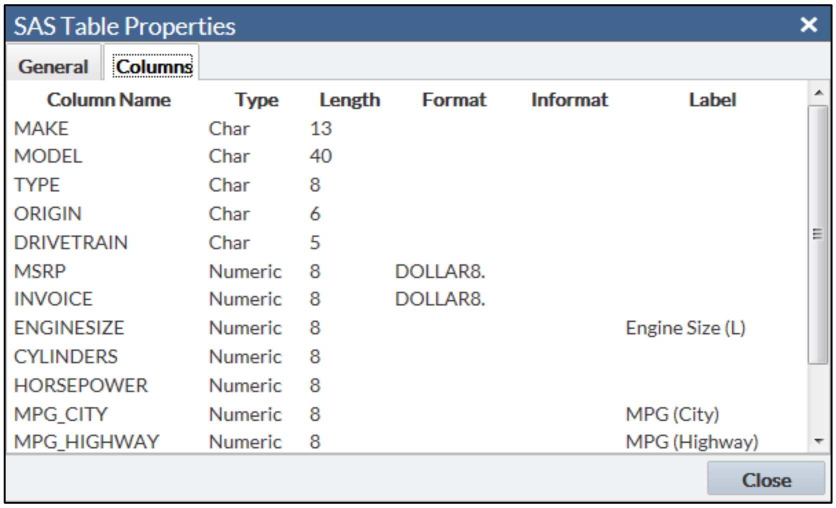
When you open a table, all of the columns in the table are displayed. You can use the Columns area to specify which columns you want to include in the table viewer. By default, the column names are displayed, but you can choose to display the column labels by selecting **Column labels** from the **View** drop-down list.



Total rows: 428 Total columns: 15 Rows 1-100

	Type	Origin	DriveTrain	MSRP	Invoice	Engine!
1	SUV	Asia	All	$36,945	$33,337	3.5
2	Sedan	Asia	Front	$23,820	$21,761	2
3	Sedan	Asia	Front	$26,990	$24,647	2.4
4	Sedan	Asia	Front	$33,195	$30,299	3.2
5	Sedan	Asia	Front	$43,755	$39,014	3.5
6	Sedan	Asia	Front	$46,100	$41,100	3.5
7	Sports	Asia	Rear	$89,765	$79,978	3.2

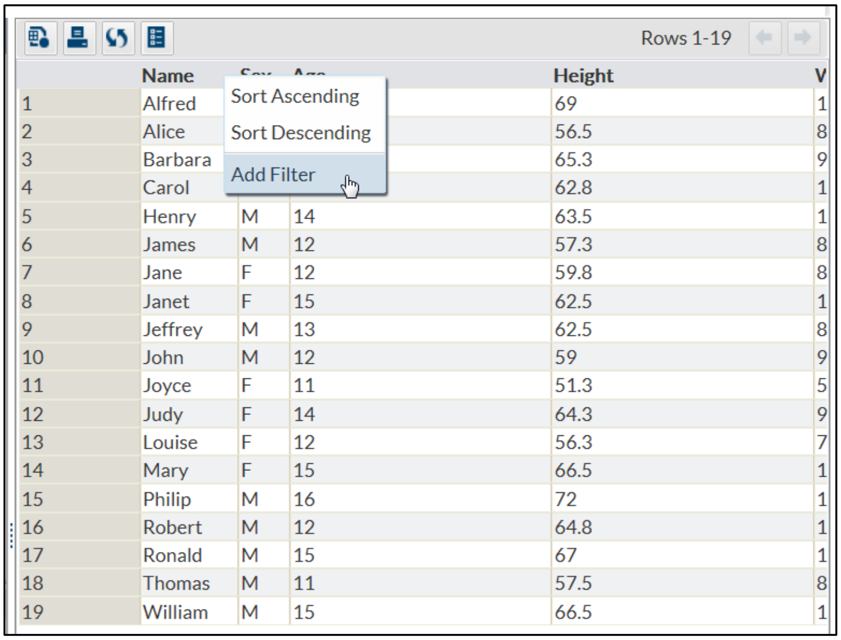
You can view the properties of the table and its columns by clicking ▤ on the toolbar.



Column Name	Type	Length	Format	Informat	Label
MAKE	Char	13			
MODEL	Char	40			
TYPE	Char	8			
ORIGIN	Char	6			
DRIVETRAIN	Char	5			
MSRP	Numeric	8	DOLLAR8.		
INVOICE	Numeric	8	DOLLAR8.		
ENGINESIZE	Numeric	8			Engine Size (L)
CYLINDERS	Numeric	8			
HORSEPOWER	Numeric	8			
MPG_CITY	Numeric	8			MPG (City)
MPG_HIGHWAY	Numeric	8			MPG (Highway)

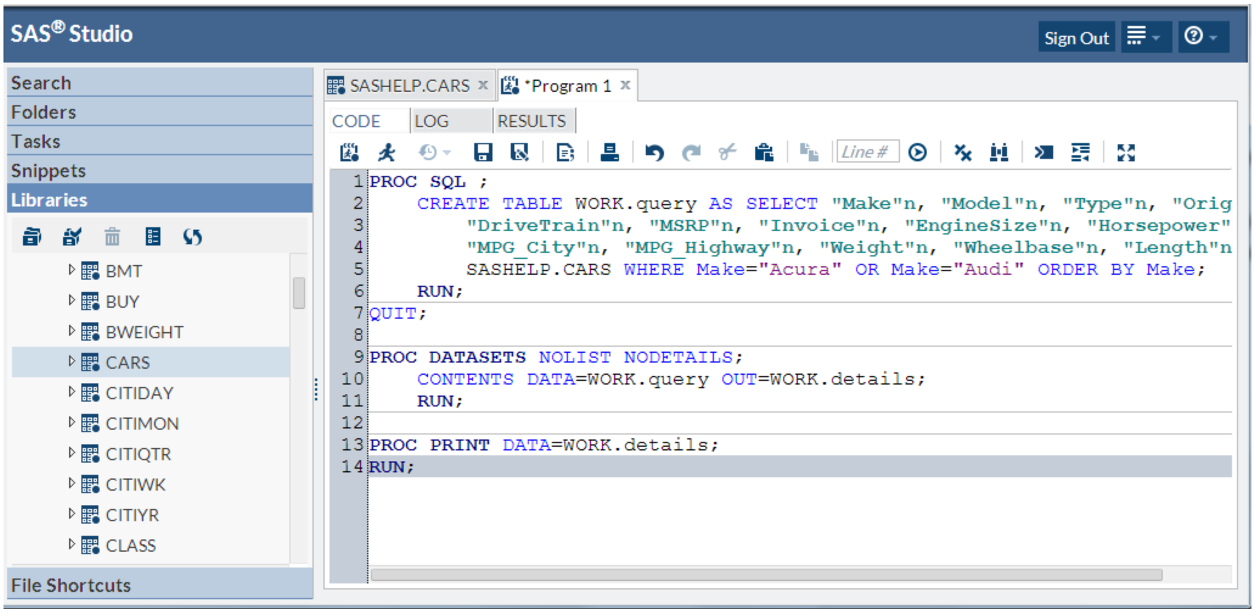
Filtering and Sorting Data

In the table viewer, you can right-click a column heading to filter and sort the data by that column.



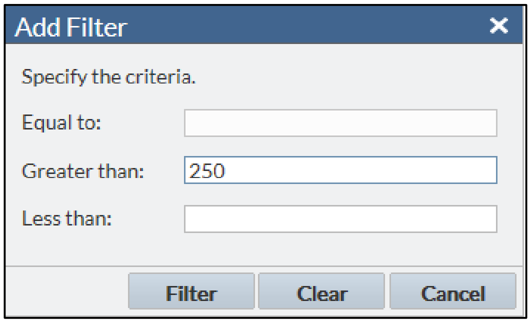
	Name	Sex	Age	Height	V
1	Alfred			69	1
2	Alice			56.5	8
3	Barbara			65.3	9
4	Carol			62.8	1
5	Henry	M	14	63.5	1
6	James	M	12	57.3	8
7	Jane	F	12	59.8	8
8	Janet	F	15	62.5	1
9	Jeffrey	M	13	62.5	8
10	John	M	12	59	9
11	Joyce	F	11	51.3	5
12	Judy	F	14	64.3	9
13	Louise	F	12	56.3	7
14	Mary	F	15	66.5	1
15	Philip	M	16	72	1
16	Robert	M	12	64.8	1
17	Ronald	M	15	67	1
18	Thomas	M	11	57.5	8
19	William	M	15	66.5	1

When you customize the table view by filtering, sorting, and selecting columns, SAS Studio is generating SAS code. To view the generated code, click on the toolbar. A new program window appears with the code that was used to create the view of the table in the table viewer. The program is a copy of the query code and is no longer associated with the original query. Editing the code does not affect the data that is displayed in the table viewer, and modifying the table viewer does not affect the contents of the code.



```
PROC SQL ;
    CREATE TABLE WORK.query AS SELECT "Make"n, "Model"n, "Type"n, "Orig
        "DriveTrain"n, "MSRP"n, "Invoice"n, "EngineSize"n, "Horsepower"
        "MPG_City"n, "MPG_Highway"n, "Weight"n, "Wheelbase"n, "Length"n
        SASHELP.CARS WHERE Make="Acura" OR Make="Audi" ORDER BY Make;
    RUN;
QUIT;

PROC DATASETS NOLIST NODETAILS;
    CONTENTS DATA=WORK.query OUT=WORK.details;
    RUN;

PROC PRINT DATA=WORK.details;
RUN;
```

The filter options vary depending on the type of column that you have selected. The Add Filter window for a numeric column enables you to specify a single value for each criterion.



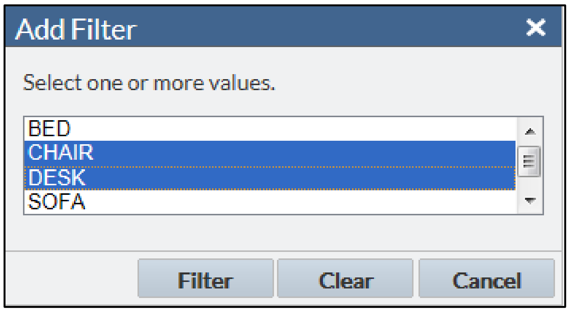
Specify the criteria.

Equal to:

Greater than: 250

Less than:

The Add Filter window for a character column enables you to select one or more values in the column.



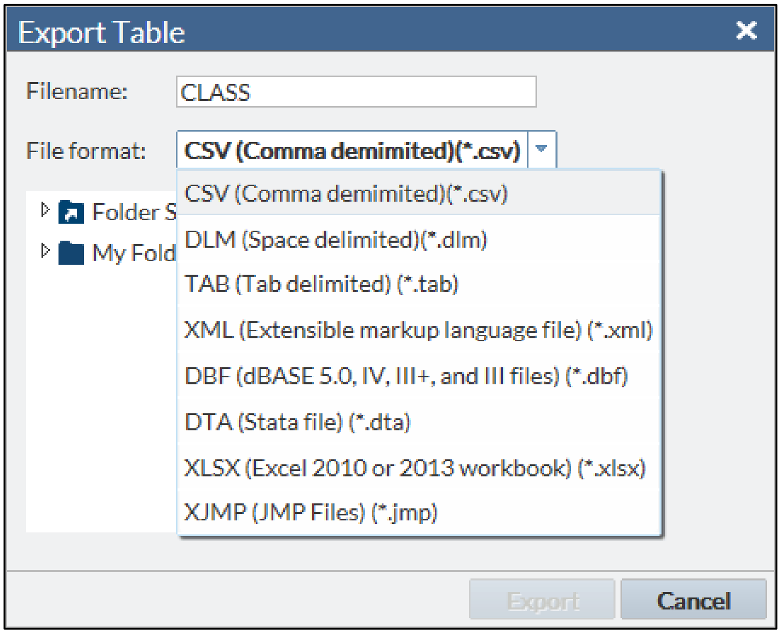
Select one or more values.

BED
CHAIR
DESK
SOFA

Exporting Data

You can use SAS Studio to export your data as another file type to a folder that you specify.
To export your data:

- Click **Libraries** in the navigation pane and browse to find the file that you want to export.
- Right-click the file and select **Export**. The **Export Table** window appears.
- In the **Filename** box, enter the name of the exported file.
- From the **File format** drop-down list, select the format of the exported file.



Export Table

Filename: CLASS

File format: CSV (Comma demimited)(*.csv)

Folder S
My Fold
CSV (Comma demimited)(*.csv)
DLM (Space delimited)(*.dlm)
TAB (Tab delimited) (*.tab)
XML (Extensible markup language file) (*.xml)
DBF (dBASE 5.0, IV, III+, and III files) (*.dbf)
DTA (Stata file) (*.dta)
XLSX (Excel 2010 or 2013 workbook) (*.xlsx)
XJMP (JMP Files) (*.jmp)
Export
Cancel

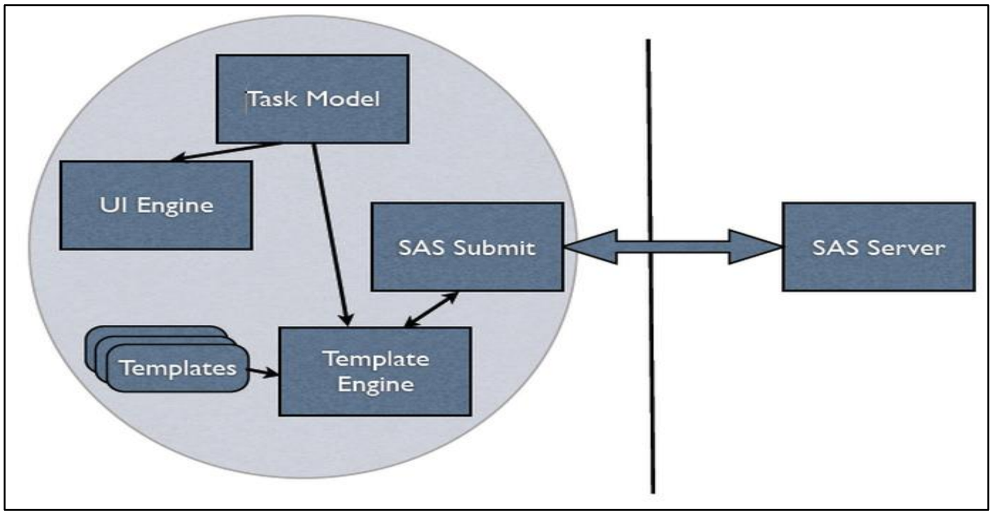
- Select the folder in which you want to save the exported file.
- Click **Export** to export the file.

Note: SAS/ACCESS® to PC Files must be licensed for some export formats.

INTRODUCTION TO THE COMMON TASK MODEL AND AUTHORING SAS STUDIO TASKS

SAS Studio is shipped with several predefined tasks, which are point-and-click user interfaces that guide the user through an analytical process. For example, tasks enable users to create a bar chart, run a correlation analysis, or rank data. When a user selects a task option, SAS code is generated and run on the SAS server. Any output (such as graphical results or data) is displayed in SAS Studio. Because of the flexibility of the task framework, you can create tasks for your site or modify tasks that have been shipped with SAS Studio. In SAS Studio, all tasks use the same common task model and the Velocity Template Language.

The Common Task Model (CTM) defines customizable or parameterized SAS execution tasks. A task is defined by describing the type of data it needs and the options it presents to the user.



SAS Studio tasks are defined by an XML specification. The general format is described here.

A top-level `Task` element has these children:

`Registration`

The `Registration` element identifies the type of task. In this element, you define the task name and unique identifier.

`Metadata`

The `Metadata` element specifies whether an input data source is required to run the task, any role assignments, and the options in the task.

- The UI engine reads the `Roles` element to determine what types of variables are required by the task. Here is the information that you would specify in this element:
  - the type of variable that the user can assign to this role (for example, numeric or character)
  - the minimum or maximum number of variables that you can assign to a role
  - the label or description of the role that appears in the user interface
- The UI engine reads the `Options` element to determine how to display the options in the user interface.

`UI`

The `UI` element describes how to present the user interface to the user. A top-down layout is supported.

`Dependencies`

The `Dependencies` element describes any dependencies that options might have on one another. For example, selecting a check box could enable a text box.

`Requirements`

The `Requirements` element specifies what conditions must be met for code to be generated.

`Code Template`

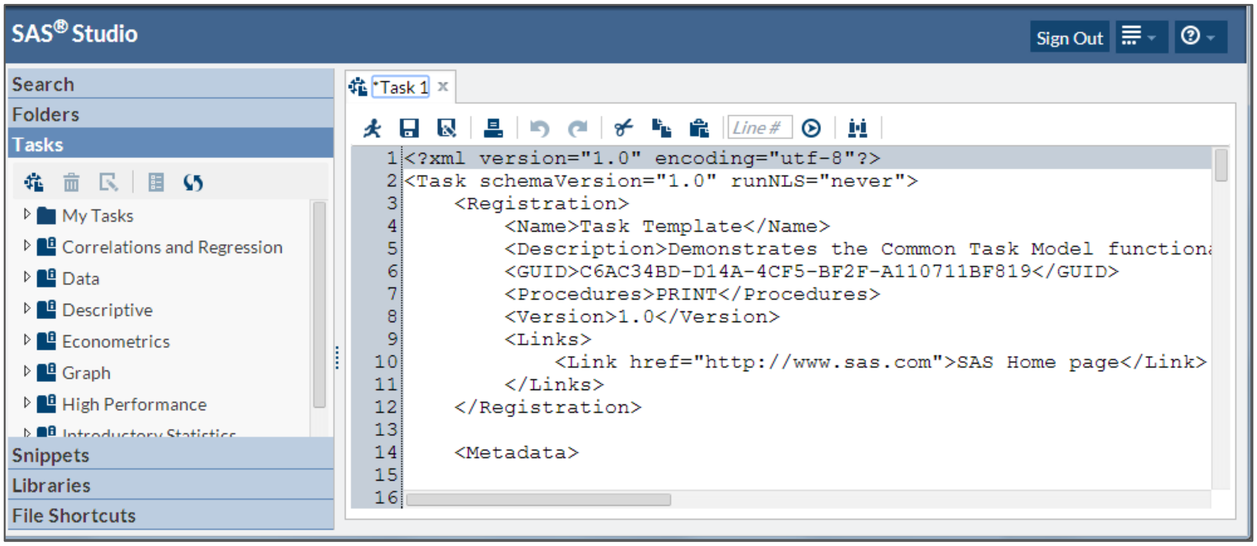
The `Code Template` element determines the output of the task defined by the Velocity Template Language.

For more information about these XML elements, see *SAS Studio 3.1 Developer's Guide*.

Using the Example Task to Get Started

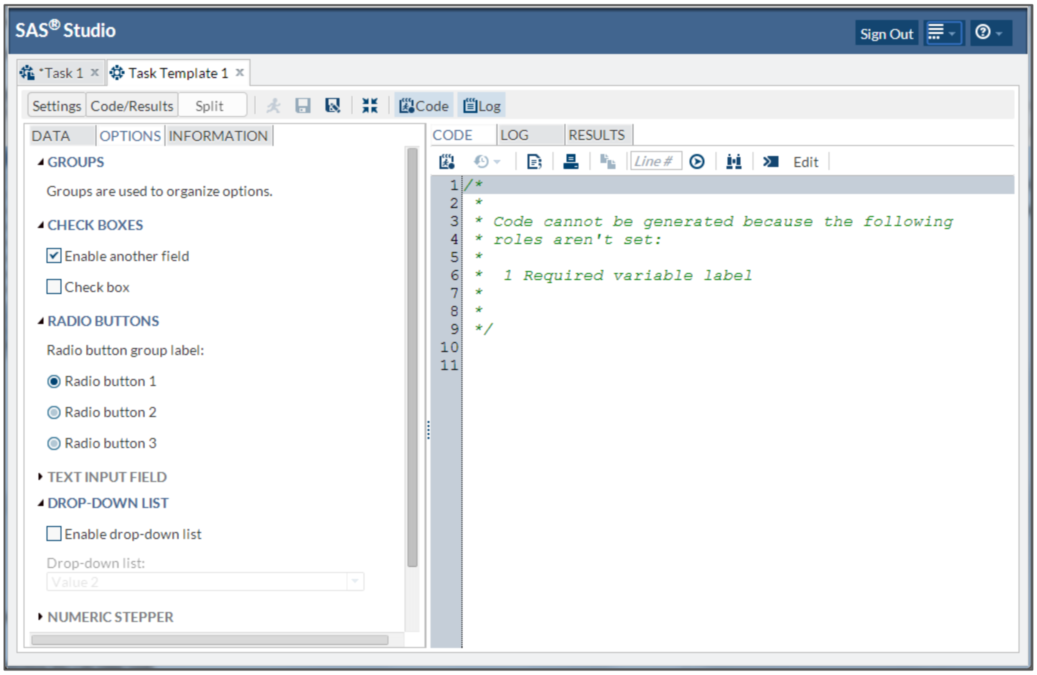
When you create a new task in SAS Studio, the application brings up an example task that demonstrates all the user interface elements you can use in a task in a full functional example.

To create a new task, click in the **Tasks** section of the navigation pane.



```
<?xml version="1.0" encoding="utf-8"?>
<Task schemaVersion="1.0" runNLS="never">
    <Registration>
        <Name>Task Template</Name>
        <Description>Demonstrates the Common Task Model functiona
        <GUID>C6AC34BD-D14A-4CF5-BF2F-A110711BF819</GUID>
        <Procedures>PRINT</Procedures>
        <Version>1.0</Version>
        <Links>
            <Link href="http://www.sas.com">SAS Home page</Link>
        </Links>
    </Registration>

    <Metadata>

```

To test the task, click or press F3 and the example task will run.



GROUPS

Groups are used to organize options.

CHECK BOXES

☑ Enable another field

□ Check box

RADIO BUTTONS

Radio button group label:

● Radio button 1

○ Radio button 2

○ Radio button 3

TEXT INPUT FIELD

DROP-DOWN LIST

□ Enable drop-down list

Drop-down list:

Value 2

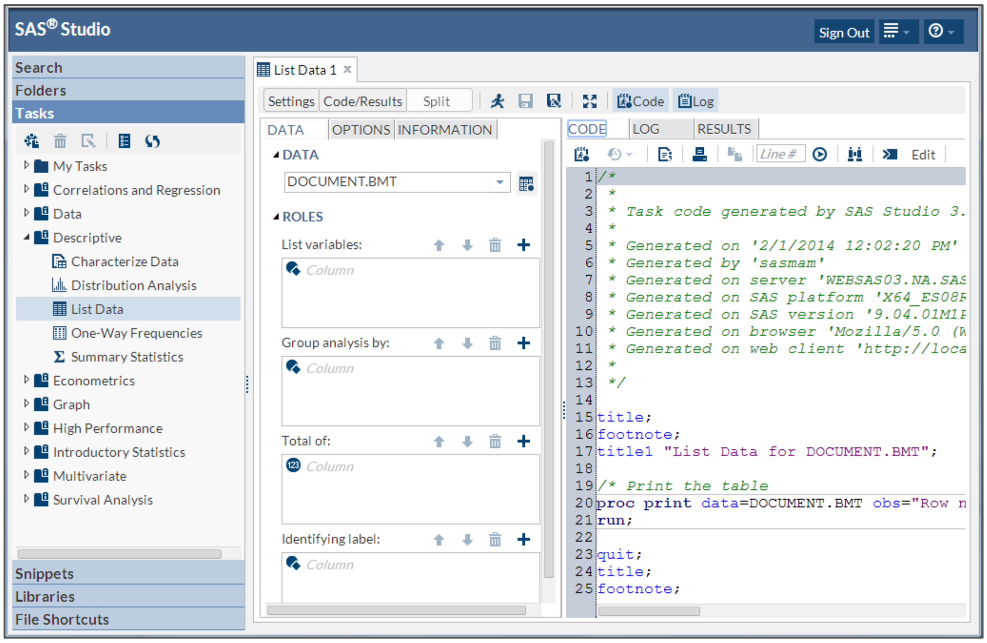
NUMERIC STEPPER

```
/*
 *
 * Code cannot be generated because the following
 * roles aren't set:
 *
 *  1 Required variable label
 *
 *
 */


```

Understanding the List Data Task

To view the user interface for a predefined task:

1. In the navigation pane, open the **Tasks** section.
2. Expand the folder that contains the task.
3. Right-click the task name and select **Open**. Alternatively, you can double-click the task to open it. The task opens to the right of the work area.

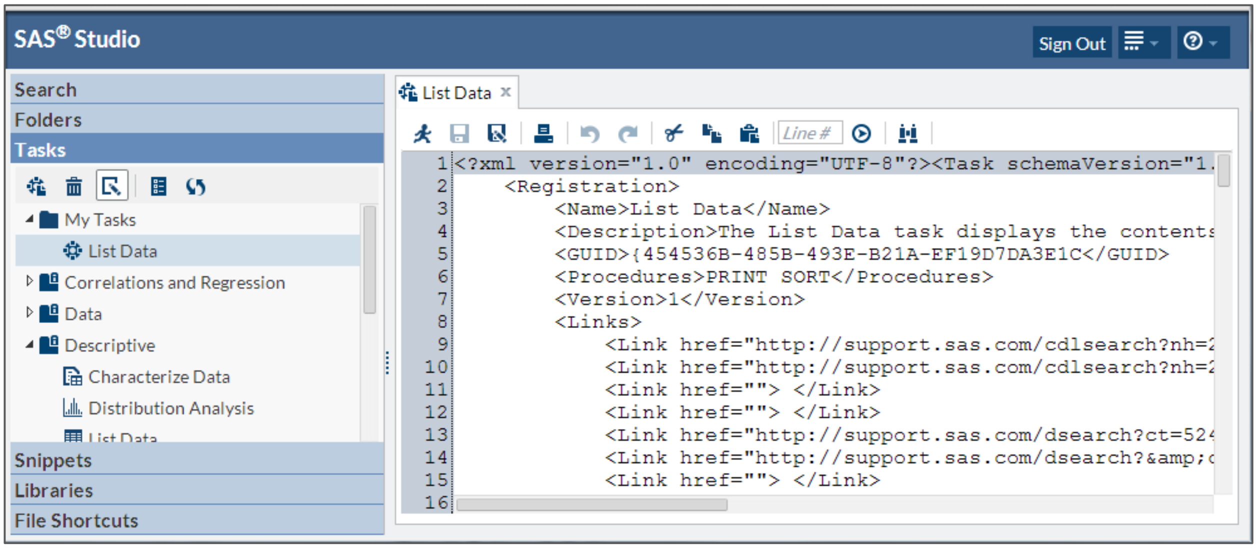


```
/*
 *
 * Task code generated by SAS Studio 3.
 *
 * Generated on '2/1/2014 12:02:20 PM'
 * Generated by 'sasmam'
 * Generated on server 'WEBSAS03.NA.SAS
 * Generated on SAS platform 'X64_ES08R
 * Generated on SAS version '9.04.01M1P
 * Generated on browser 'Mozilla/5.0 (W
 * Generated on web client 'http://loca
 *
 */

title;
footnote;
title1 "List Data for DOCUMENT.BMT";

/* Print the table
proc print data=DOCUMENT.BMT obs="Row n
run;

quit;
title;
footnote;
```

You cannot edit the code for a predefined task. However, you can copy the task code and edit the copy.

To view the code for a predefined task:

1. In the navigation pane, open the **Tasks** section.
2. Expand the **Descriptive** folder.
3. Right-click **List Data** and select **Add to My Tasks**. A copy of the task is added to your **My Tasks** folder.
4. Open the **My Tasks** folder and select the copied task.
5. Click . The XML and Velocity code for the task appears.



```
<?xml version="1.0" encoding="UTF-8"?><Task schemaVersion="1.
    <Registration>
        <Name>List Data</Name>
        <Description>The List Data task displays the contents
        <GUID>{454536B-485B-493E-B21A-EF19D7DA3E1C</GUID>
        <Procedures>PRINT SORT</Procedures>
        <Version>1</Version>
        <Links>
            <Link href="http://support.sas.com/cdlsearch?nh=2
            <Link href="http://support.sas.com/cdlsearch?nh=2
            <Link href=""> </Link>
            <Link href=""> </Link>
            <Link href="http://support.sas.com/dsearch?ct=524
            <Link href="http://support.sas.com/dsearch?&amp;c
            <Link href=""> </Link>
```

This example shows the task definition for the List Data task and labels each element in the XML code.

```
<?xml version="1.0" encoding="UTF-16"?><Task schemaVersion="1.0">
1<Registration>
      <Name>List Data</Name>
      <Description>The List Data task displays the contents of a table as a
         report.</Description>
      <GUID>3C74D5C3-B845-4926-A749-E56CBA1283E2</GUID>
      <Procedures>PRINT SORT</Procedures>
      <Version>1</Version>
      <Links>
         <Link href="http://support.sas.com/cdlsearch?nh=25&amp;ct=80000&amp;
            qt=PROC+PRINT">PROC PRINT Documentation</Link>
         <Link href="http://support.sas.com/cdlsearch?nh=25&amp;ct=80000&amp;
            qt=PROC+SORT">PROC SORT Documentation</Link>
      </Links>
</Registration>
2<Metadata>
   <DataSources>
      <DataSource name="DATASOURCE">
         <Roles>
            <Role maxVars="0" minVars="0" name="VAR" order="true"
               type="A">List variables</Role>
            <Role maxVars="0" minVars="0" name="BY" order="true"
               type="A">Group analysis by</Role>
            <Role maxVars="0" minVars="0" name="SUM" order="true"
               type="N">Total of</Role>
            <Role maxVars="0" minVars="0" name="ID" order="true"
               type="A">Identifying label</Role>
         </Roles>
      </DataSource>
   </DataSources>

   <Options>
      <Option defaultValue="all" inputType="combobox" name="ROWS2LIST"
         width="264px">Rows to list:</Option>
      <Option inputType="string" name="all">All rows</Option>
      <Option inputType="string" name="firstnrows">First n rows</Option>
      <!--<Option inputType="string" name="firstnpct">First n percent
         of rows</Option>-->
      <!--<Option inputType="string" name="everynrow">Every nth row</Option>-->
         <Option decimalPlaces="0" defaultValue="10" indent="1"
            inputType="numstepper" maxValue="999999" minValue="1"
            name="NVALUE">Amount (n):</Option>
         <!--<Option decimalPlaces="0" defaultValue="25" indent="1"
            inputType="numstepper" maxValue="100" minValue="1"
            name="NPERCENT">Percent (n):</Option>-->

      <Option inputType="string" name="DATATAB">DATA</Option>
      <Option inputType="string" name="DATAGROUP">DATA</Option>
      <Option inputType="string" name="ROLESGROUP">ROLES</Option>

      <Option inputType="string" name="OPTIONSTAB">OPTIONS</Option>
      <Option inputType="string" name="BASICOPTIONS">BASIC OPTIONS</Option>

      <Option defaultValue="1" inputType="checkbox" name="OBS">
         Display row numbers</Option>
      <Option defaultValue="Row number" indent="1" inputType="inputtext"
         name="OBSHEADING" width="250px">Column label</Option>

      <Option defaultValue="1" inputType="checkbox" name="LABEL">Use column
         labels as column headings</Options>

      <Option defaultValue="0" inputType="checkbox" name="PRINTNUMROWS">Display
         number of rows</Option>

      <Option defaultValue="0" inputType="checkbox" name="ROUND">Round values
         before summing the variable</Option>

      <Option defaultValue="default" inputType="combobox" name="HEADING"
         width="264px">Heading direction:</Option>
      <Option inputType="string" name="default">Default</Option>
      <Option inputType="string" name="horizontal">Horizontal</Option>
      <Option inputType="string" name="vertical">Vertical</Option>

      <Option defaultValue="default" inputType="combobox" name="WIDTH"
         width="264px">Column width</Option>
      <Option inputType="string" name="full">Full</Option>
      <Option inputType="string" name="minimum">Minimum</Option>
      <Option inputType="string" name="uniform">Uniform</Option?
      <Option inputType="string" name="uniformby">Uniform by</Option>

      <Option defaultValue="0" inputType="checkbox"
         name="SPLITLABEL">Split labels</Option>
      <Option defaultValue="*" indent="1" inputType="combobox"
         name="SPLITLABELVALUE"
         width="80px">Split character</Option>
   </Options>
</Metadata>
3<UI>
   <Container option="DATATAB">
      <Group open="true" option="DATAGROUP">
         <DataItem data="DATASOURCE"/>
      </Group>
      <Group open="true" option="ROLESGROUP">
         <RoleItem role="VAR"/>
         <RoleItem role="BY"/>
         <RoleItem role="SUM"/>
         <RoleItem role="ID"/>
      </Group>
   </Container>

   <Container option="OPTIONSTAB">
```

```
      <Group open="true" option="BASICOPTIONS">

         <OptionItem option="OBS"/>
         <OptionItem option="OBSHEADING"/>
         <OptionItem option="LABEL"/>
         <OptionItem option="PRINTNUMROWS"/>
         <OptionItem option="ROUND"/>

         <OptionChoice option="HEADING"/>
            <OptionItem option="default"/>
            <OptionItem option="horizontal"/>
            <OptionItem option="vertical"/>
         </OptionChoice>

         <OptionChoice option="WIDTH"/>
            <OptionItem option="default"/>
            <OptionItem option="full"/>
            <OptionItem option="minimum"/>
            <OptionItem option="uniform"/>
            <OptionItem option="uniformby"/>
         </OptionChoice>

         <OptionItem option="SPLITLABEL"/>
            <OptionChoice option="SPLITLABELVALUE"/>
               <OptionValue>*</OptionValue>
               <OptionValue>!</OptionValue>
               <OptionValue>@</OptionValue>
               <OptionValue>#</OptionValue>
               <OptionValue>$</OptionValue>
               <OptionValue>%</OptionValue>
               <OptionValue>^</OptionValue>
               <OptionValue>amp</OptionValue>
               <OptionValue>+</OptionValue>
         </OptionChoice>
      </Group>
   </Container>
</UI>
4<Dependencies>
   <Dependency condition="OBS=='1'">
      <Target action="enable" conditionResult="true" option="OBSHEADING"/>
      <Target action="disable" conditionResult="false" option="OBSHEADING"/>
   </Dependency>
   <Dependency condition="$SPLITLABEL=='1'">
      <Target action="enable" conditionResult="true" option="SPLITLABELVALUE"/>
      <Target action="disable" conditionResult="false" option="SPLITLABELVALUE"/
   </Dependency>
   <Dependency condition="$ROWS2LIST.equalsIgnoreCase('firstnrows')">
      <Target action="enable" conditionResult="true" option="NVALUE"/>
      <Target action="disable" conditionResult="false" option="NVALUE"/
   </Dependency>

</Dependencies>
5<CodeTemplate>
   <![CDATA[

#set( $TABLE = $DATASOURCE )
title;
footnote;

title1 "List Data for $DATASOURCE";

#if( $BY.size()>0 ) /*Sort $DATASOURCE for BY group processing. */
proc sort data=$DATASOURCE out=WORK.SORTTEMP;
   by #foreach($item in $BY) $item#end;
run;
#set( $TABLE = "WORK.SORTTEMP" )
#end

/* Print the table     */

proc print data=$TABLE
#if($ROWS2LIST.equalsIgnoreCase("firstnrows"))
      (obs=$NVALUE)
#end
#if ($OBS == '1')
      obs="$OBSHEADING"
#else
      noobs
#end
#if ($HEADING && ($HEADING !="default"))
      heading=$HEADING
#end
#if ($LABEL == '1')
      label
#end
#if ($PRINTNUMROWS == '1')
      n
#end
#if ($ROUND == '1')
      round
#end
#if ($DIVIDE == "0")
      rows=page
#end
#if($WIDTH && ($WIDTH !='default'))
      width=$WIDTH
#end
#if ($SPLITLABEL == '1')
      split='$SPLITLABELVALUE'
#end
   ;
#if( $VAR.size()>0 )
      var #foreach ( $item in $VAR ) $item#end;
#end
#if( $BY.size()>0 )
      by #foreach ( $item in $BY ) $item#end;
#end
#if( $SUM.size()>0  )
      sum #foreach ( $item in $SUM ) $item#end;
#end
#if( $ID.size()>0  )
      id #foreach ( $item in $ID ) $item#end;
#end
run;
quit;

title;
footnote;
         ]]>
   </CodeTemplate>
</Task>
```

Use this list to identify each section in the preceding code:

1. The `Registration` element defines how the task is displayed in the Task tree.
2. The `Metadata` element, which specifies whether an input data source is required, defines the roles for columns in the input data source, and any options that are required for the task to run.
3. The `UI` element specifies the layout of the task in the user interface.
4. The `Dependencies` element specifies the options that rely on one another in order for the task to work.
5. The `Code Template` element generates the SAS code for the task. The Code Template is defined by the Velocity Template Language.

CONCLUSION

Many people program in the SAS language by using SAS Display Manager or SAS Enterprise Guide. SAS Studio is different because it is a tool that you can use to write and run SAS code through all major web browsers without any software installed on the client machine. This paper introduced the essential elements of this new programming interface to the SAS language and presented the concepts of writing custom tasks that use the Common Task Model. SAS Studio will be available on top of the first maintenance release for SAS 9.4 in March 2014. Subsequent releases of SAS Studio will combine SAS Enterprise Guide features to provide one unified programming environment for advanced SAS programmers and new SAS programmers requiring more active assistance.

REFERENCES

- The Apache Velocity Project. *The Velocity User Guide*. Available at http://velocity.apache.org/engine/releases/velocity-1.5/user-guide.html.

- "Apache Velocity." Wikipedia. Available at http://en.wikipedia.org/wiki/Apache_Velocity.

- The Apache Velocity Project. Available at http://velocity.apache.org/.

RECOMMENDED READING

- *Getting Started with Programming in SAS Studio 3.1*
- *SAS Studio 3.1: Administrator's Guide*
- *SAS Studio 3.1: User's Guide*
- *SAS Studio 3.1: Developer's Guide*

ACKNOWLEDGMENTS

The authors would like to acknowledge Christie Corcoran, Cummins Mebane, Danny Zimmerman, Dave Mackie, Fran Miles, J. Jeffreys-Chen, Kris Kiser, Mike Porter, Pat Bostic, Philip Johns, Rick Pappas, Satish Chittineni, Stephen Slocum, Swapnil Ghan, Samantha DuPont, Shannon Smith, Richard Fox, Anjali Matange, Raj Patel, Tina Tart, Amy Peters, and Becky Williamson for reviews and assistance on this paper.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors:

Michael A. Monaco
100 SAS Campus Drive
Cary, NC 27513
SAS Institute Inc.
Michael.Monaco@sas.com
http://www.sas.com

Marie Dexter
100 SAS Campus Drive
Cary, NC 27513
SAS Institute Inc.
Marie.Dexter@sas.com
http://www.sas.com

Jennifer Tamburro
100 SAS Campus Drive
Cary, NC 27513
SAS Institute Inc.
Jennifer.Tamburro@sas.com
http://www.sas.com