Paper 5081-2020

Building an Internal Application Using the SAS® Stored Process Web Application

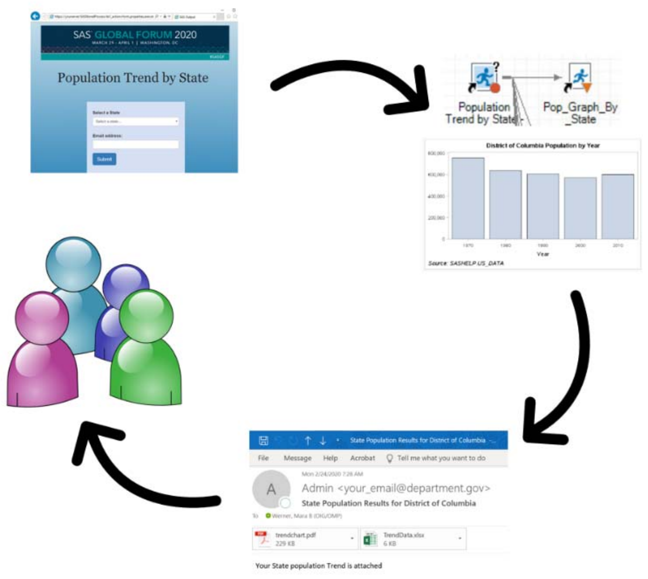
Mara Werner, Department of Health and Human Services

ABSTRACT

Using the SAS Stored Process Web Application and SAS® prompts you can take existing SAS code and modify it to accommodate different variations of your most common requests. You can use a custom HTML front end to provide an intuitive, modern-looking user interface, ODS PDF to present a summary of your results, ODS Excel for supporting data, and you can deliver it all by email. Your customers will love getting results on-demand and you'll love that they can fish for themselves, freeing you from repetitive queries and allowing you to focus on more interesting and productive work.

INTRODUCTION

This paper shows step-by-step how to set up an internal web application that can provide data on-demand to your organization using SAS Enterprise Guide, the SAS Stored Process Web Application, and an SMTP Mail Server.


SAS GLOBAL FORUM 2020
Population Trend by State
Submit
Population Trend by State
Pop_Graph_By_State
District of Columbia Population by Year
Year
Source: SASHELP.US_DATA
State Population Results for District of Columbia
Admin <your_email@department.gov>
trendchart.pdf
TrendData.xlsx
Your State population Trend is attached

This paper will guide you as you build an example web application called "Population Trend by State". When a user selects a state, it gets passed to SAS as a parameter and is used in your code to filter data and make a chart. The data is exported to Excel and the chart is embedded into a PDF. Both files are attached to an email and delivered back to the user.

The dataset referenced in the sample code is SASHELP.US_DATA (available in SAS 9.4 and higher). The screen shots are from SAS Enterprise Guide 8.2.

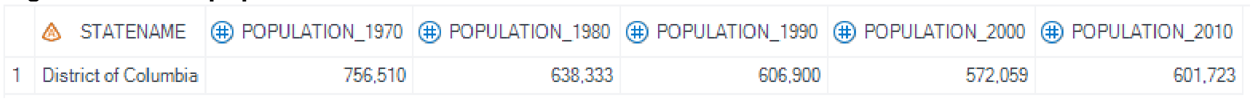
STEP 1: WRITE THE SAS CODE

The heart of the web application is SAS and SQL code to create a table of population counts for a state, transpose the table from wide to long, and then make a trend chart.

For the final application you will use macro variables to parameterize the code so it will run for any state in the United States. But first it is helpful to write everything out fully for an example and confirm that it works as expected. Among other things, the dataset SASHELP.US_DATA contains state names and population counts for several years. Select these variables and filter for the District of Columbia:

```
PROC SQL;
   create table state_pop as
   select statename
      , population_1970
      , population_1980
      , population_1990
      , population_2000
      , population_2010
   from sashelp.us_data
   where Statename = 'District of Columbia'
   ;QUIT;
```

Figure 1. 'State_pop' Table for District of Columbia



	STATENAME	POPULATION_1970	POPULATION_1980	POPULATION_1990	POPULATION_2000	POPULATION_2010
1	District of Columbia	756,510	638,333	606,900	572,059	601,723

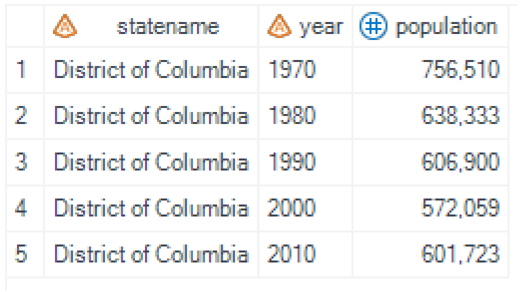
To create a trend chart from the 'state_pop' data, the table needs to be long rather than wide. Using the TRANSPOSE procedure allows you reshape the dataset:

```
   PROC TRANSPOSE data=state_pop out=state_pop_transpose;
      by statename;
   RUN;
```

Follow this up with a data step to clean up the variable names:

```
   DATA state_pop_transpose2;
     retain statename year population;
     set state_pop_transpose (rename=(col1=population));
     year=substr(_label_,1,4);
     drop _name_ _label_;
   RUN;
```

Figure 2. 'State_pop_transpose2' Table for the District of Columbia



	statename	year	population
1	District of Columbia	1970	756,510
2	District of Columbia	1980	638,333
3	District of Columbia	1990	606,900
4	District of Columbia	2000	572,059
5	District of Columbia	2010	601,723

STEP 2: SUBSTITUTE MACRO VARIABLE FOR TEXT

Now that the code works for a single state, you can substitute a macro variable for the state reference. This adds the flexibility to run your code for any state a user is interested in.

```
%let state = District of Columbia;

PROC SQL;
create table state_pop as
select statename
   , population_1970
   , population_1980
   , population_1990
   , population_2000
   , population_2010
from sashelp.us_data
where statename = "&state"
;QUIT;
```

Update the rest of the code in the same way. Anywhere you see 'District of Columbia', change it to '&state'.

Note the double quotes when the text contains a macro variable reference. Often in SAS the double and single quotes are interchangeable, but here the double quotes are necessary. SAS will not resolve a macro variable within single quotes.

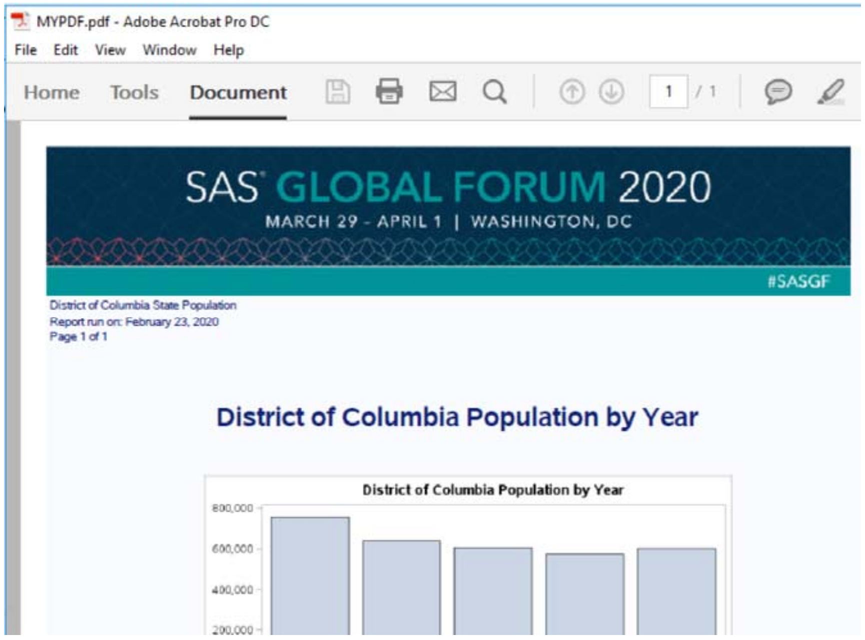
It is also possible to derive the state name from two letter state postal code. You may like this better because testing the code goes faster when you have less to type. In that case, you would first define a macro variable ('st') to take the two-letter state code and then transform it with the STNAMEL function to define the macro variable 'state':

```
%let st = DC;

%let state = %sysfunc(stnamel(&st));
```

STEP 3: OUTPUT RESULTS TO PDF

In this sample application, the dataset is small and you'll get results in the browser within moments. However, your real-world application may include computations that take longer to run, and your users may prefer a portable copy of the results. For these reasons, you may want to save your chart in a PDF document.

Figure 3. PDF Results for "Population Trend by State"


MYPDF.pdf - Adobe Acrobat Pro DC
File Edit View Window Help
Home Tools Document
1 / 1
SAS GLOBAL FORUM 2020
MARCH 29 - APRIL 1 | WASHINGTON, DC
#SASGF
800,000
600,000
400,000
200,000

District of Columbia State Population
Report run on: February 23, 2020
Page 1 of 1

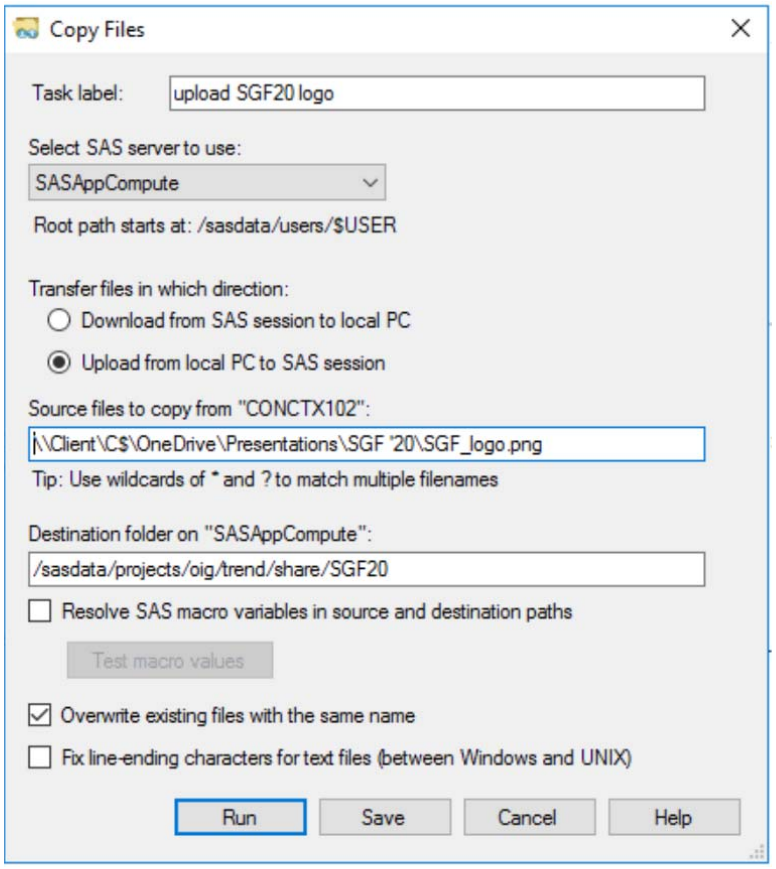
District of Columbia Population by Year

District of Columbia Population by Year

UPLOAD AN IMAGE

The header logo in this example is a screen shot from the SAS Global Forum 2020 website. You can upload an image from your computer to a server location on SASEG using the 'Copy Files' task. If your image isn't the right size, the best practice is to adjust this outside of SAS and then copy the file in again. It may require some trial and error. Some images appear blurry in the PDF, if this happens to you consider opening a ticket with SAS Support.

Figure 4. Copy Files Task to Upload an Image.



Copy Files

Task label: upload SGF20 logo

Select SAS server to use: SASAppCompute

Root path starts at: /sasdata/users/$USER

Transfer files in which direction:
- ○ Download from SAS session to local PC
- ● Upload from local PC to SAS session

Source files to copy from "CONCTX102": \\Client\C$\OneDrive\Presentations\SGF '20\SGF_logo.png

Tip: Use wildcards of * and ? to match multiple filenames

Destination folder on "SASAppCompute": /sasdata/projects/oig/trend/share/SGF20

- ☐ Resolve SAS macro variables in source and destination paths
- ☑ Overwrite existing files with the same name
- ☐ Fix line-ending characters for text files (between Windows and UNIX)

CREATE A TEMPORARY FILE PATH

To create a PDF in SAS, you need to specify a location to save the file. If there is a network location that is accessible from the SAS Web Application Server and your users can access it, you can save your PDF there. If not, use the following code to create a temporary file. Then save the file path to a macro variable so you can reference it later when you add this file as an email attachment:

```
/* create a temporary file for the PDF called 'PDFtmp' and save the
   path to the macro variable 'PDFpath' */
filename pdftmp temp;
%let pdfpath=%sysfunc(pathname(pdftmp));
%put &=pdfpath;
```

WRITE TO PDF

ODS PDF is like many of the other ODS calls – there is an outer wrapper and then code inside. The beginning of the wrapper defines the name of the PDF, a style, and a location to save the file.

To use the temporary file path that you just created, reference the “pdftmp” fileref. In the beginning of the ODS PDF wrapper you can also specify other parameters, like turning off automatic bookmarks or printing the date and page numbers in the PDF header. You can also define an ‘ODS escapechar’. Defining an escape character is necessary for PDFs that will have a header logo or other text formatting. This example uses a caret (^) as the escape character, but you can use any symbol that does not show up for other reasons in your code. The last line is the end of the wrapper, closing the PDF:

```
/*start a pdf document for the results*/
ODS PDF (id=emailpdf)
   style=htmlblue
   file= pdftmp
   bookmarklist=none;

   options nodate nonumber;

   ods escapechar='^';

 … body code here…

/*close pdf destination to finish document*/
ODS PDF(emailpdf) close;
```

Inside the wrapper, you’ll put all the code that tells SAS what to include in your PDF document. For this example, you’ll include a header logo, header text, a title, and a chart.

Header Logo

One simple way to insert a header logo is using the preimage style inside an ODS TEXT statement. Use the escapechar (^) to indicate that you are about to give some instructions and then ‘S’ to indicate that those instructions are style-related. The preimage style attribute allows you to specify the file path for the logo image:

```
/* Add a document header/banner */
ODS text= "^S={preimage='/…filepath…/SGF20_logo.png'}";
```

Header Text

To write header text, use ODS TEXT again. Start each line with the escape character. This time, instead of the preimage style, you can specify a font size and justification for the text. The text is inside double quotes (instead of single quotes) so that the macro variables will resolve.

```
/* Add header notes */
ODS TEXT = "^S={font_size=8PT Just=L} &State State Population";
ODS TEXT = "^S={font_size=8PT Just=L} Report run on: &today";
ODS TEXT = "^S={font_size=8PT Just=L} Page ^{thispage} of ^{lastpage}";
```

To include the run-date in the report, you can define a macro variable called ‘today’ using the TODAY function in SAS. This macro definition goes at the top of your code so that it will be available when you call it from the PDF. The TODAY function produces a date in the SAS format – that is the number of days between January 1, 1960 and today. Before including it in your report, you’ll want to choose a more readable format, like worddate:

```
%let today = %sysfunc(today(), worddate);
```

If your report has multiple pages, you may want to include a page count in the header text. ODS PDF will automatically generate variable references for each page of your document. You can reference those using ^{thispage} and ^{lastpage}.

Title

To make a title for your graph, use ODS TEXT with a bigger font size and justified center:

```
/* Write a Title */
 ODS TEXT= "^S={font_size=18pt just=C font_weight=bold}^2n&State Population
 by Year^2n";
```

You can use the escape character to insert line breaks before and after the title. '^n' is the code for one line break. '^2n' means two line breaks.

Chart

To print the graph inside the PDF use the SGPLOT procedure. This will make a very large square chart. If you prefer something smaller or with different dimensions, you can use the ODS graphics options to specify width and height:

```
ODS graphics on / width=5in height=3in;

PROC SGPLOT data = state_pop_transpose2;
   title1 "&state Population by Year";
   footnote justify=left italic 'Source: SASHELP.US_DATA';
   vbar year / response=population;
   xaxis label = 'Year';
   yaxis display=(nolabel);
RUN;
```

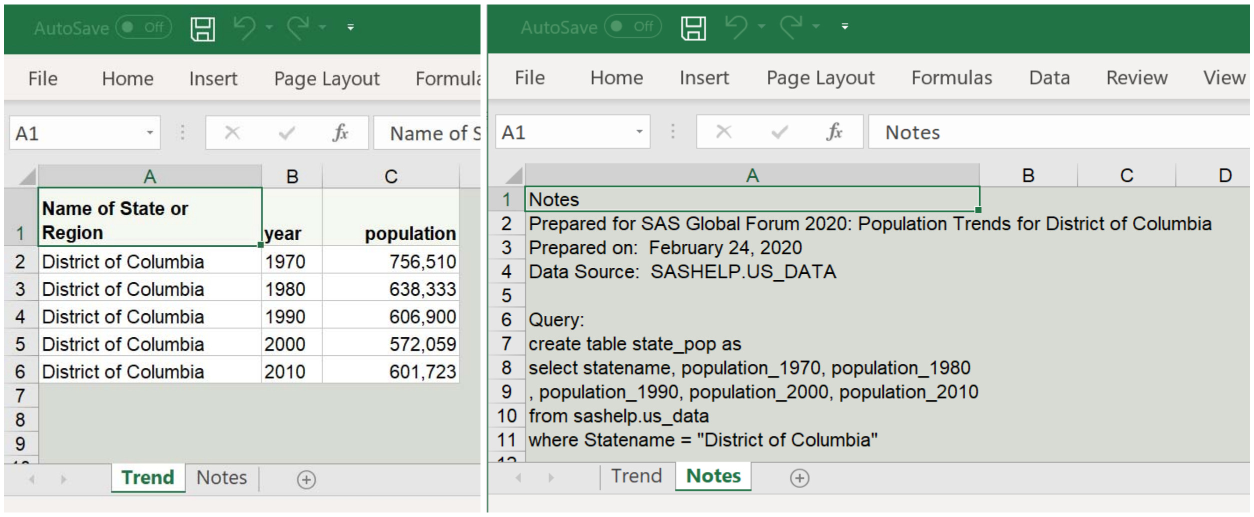
Once you have code for everything you want in your PDF, remember to close the PDF destination to complete the document:

```
/*close pdf destination to finish document*/
ODS PDF(emailpdf) close;
```

STEP 4: OUTPUT RESULTS TO EXCEL

Printing a chart to PDF is great, but some people want the actual numbers. For the numbers people, having the data in Excel will be easier. You can create an Excel workbook using ODS EXCEL. You can even add an extra tab with methodology notes – like the analysis date, analyst name, and the query code using the EXPORT procedure.

Figure 5. Excel Book with Two Sheets Containing Data and Notes



Name of State or Region	year	population
District of Columbia	1970	756,510
District of Columbia	1980	638,333
District of Columbia	1990	606,900
District of Columbia	2000	572,059
District of Columbia	2010	601,723

```
Notes
Prepared for SAS Global Forum 2020: Population Trends for District of Columbia
Prepared on:  February 24, 2020
Data Source:  SASHELP.US_DATA

Query:
create table state_pop as
select statename, population_1970, population_1980
, population_1990, population_2000, population_2010
from sashelp.us_data
where Statename = "District of Columbia"
```

CREATE A TEMPORARY FILE PATH

Just like you did for the PDF, create a temporary file and save the file path to a macro variable so that you can reference it later. Here the fileref is ‘exceltmp’ and the macro variable containing the file path is ‘excelpath’:

```
/* create a temporary file for the PDF called ‘Exceltmp’ and save
   the path to the macro variable ‘Excelpath’ */
filename exceltmp temp;
%let excelpath=%sysfunc(pathname(exceltmp));
%put &=excelpath;
```

WRITE TO EXCEL

Before writing to ODS EXCEL, reset the title and footnote statements:

```
title; footnote;
```

Like ODS PDF, ODS EXCEL uses a wrapper to open and close the destination. In the opening statement, specify the temporary file location, a style, and name the first tab. The body of the ODS EXCEL statement here is the PRINT procedure. Once you have finished writing to Excel, remember to close the ODS EXCEL destination:

```
/*send the data to excel in a way that keeps the formatting*/

ODS EXCEL file="&excelpath..xlsx" style=statistical
   options(sheet_name="Trend");

PROC PRINT data=state_pop_transpose2 noobs label;
RUN; QUIT;

ODS EXCEL CLOSE;
```

To create a simple notes tab, start by writing the text into a SAS dataset and then use the EXPORT procedure to add that text as a new tab in the same Excel workbook you just created. Here you are using the cards statement and defining ‘*’ as the delimiter for a new row. Note that there are two variables, ‘notesprep’ and ‘notes’. Initially you write the raw text to ‘notesprep’, then you create a new variable called ‘notes’ and use the RESOLVE function to resolve the macro variables. Because the text contains special characters, use the QUOTE and DEQUOTE functions around the RESOLVE function. Once you have created ‘notes’, you can drop ‘notesprep’:

```
DATA Notes (drop=notesprep);
   infile cards delimiter='*';
   input notesprep :$100.;
   Notes=DEQUOTE(RESOLVE(QUOTE(Notesprep)));
   cards;
   Prepared for SAS Global Forum 2020: Population Trends for &state
   Prepared on:  &today *
   Data Source:  SASHELP.US_DATA *
   *
   Query: *
      create table state_pop as*
      select statename, population_1970, population_1980*
         , population_1990, population_2000, population_2010*
      from sashelp.us_data*
      where Statename = "&state"*
;RUN;
```

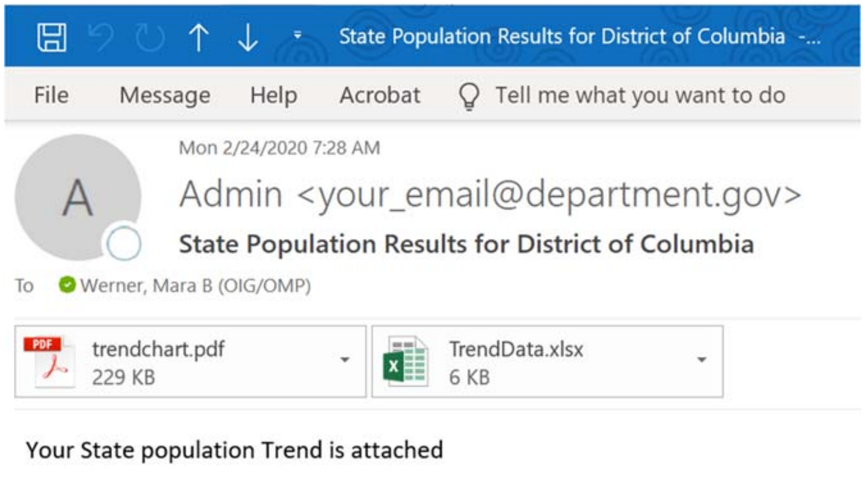
To add the notes tab to your Excel book, use PROC EXPORT to write the contents of the notes dataset to your Excel workbook:

```
PROC EXPORT data= notes
   outfile="&excelpath"
   dbms=xlsx
   replace;
   sheet = 'Notes'
;QUIT;
```

STEP 5: SEND RESULTS BY EMAIL

Some SAS installations include an SMTP mail server that is available to the SAS Web Application Server. If yours does, you have the option to send the PDF and Excel workbook directly to the user. For a web application that takes some time to run, this allows the user to request a report and move on to other things. When the report is ready, it will show up in their email inbox. If your installation does not include the mail server, but there is a common storage location that all your users can access, you could have SAS save files there instead of the temporary locations shown in this paper.

Figure 6. Email with PDF and Excel Attachments


State Population Results for District of Columbia -...
File
Message
Help
Acrobat
Tell me what you want to do

Mon 2/24/2020 7:28 AM

Admin <your_email@department.gov>

State Population Results for District of Columbia

To Werner, Mara B (OIG/OMP)

trendchart.pdf 229 KB

TrendData.xlsx 6 KB

Your State population Trend is attached

When your application is running on the web, you can have users enter their own email address. As a placeholder while you are testing your code, define a macro variable named ‘email’ with your email address:

```
%let email = your_email@department.gov;
```

The code to send an email through SAS starts by defining a temporary file with a fileref (‘em’) so that you can reference it later. Within this fileref definition, you can specify that this will be an email, the to/from addresses, you can name any attachments and specify the filepath and type, and you can write the subject line. As before, any text strings that include macro variable references are in double quotes.

Here you are using the option to write in a ‘from’ address. This will make the email appear to be from you instead of the actual sender which is a SAS server. While the deception may give you pause, this does make it easier for users to find a contact if they have questions. They can hit reply and their email will go to a monitored box rather than off into the ether.

Here is the filename statement for your email:

```
/* Generate email*/
filename em email
   to="&email"
   from="Admin <your_email@department.gov>"
   attach=("&pdfpath" name="trendchart"
           content_type="application/pdf" ext="pdf")
   attach=("&excelpath..xlsx" name="TrendData"
           content_type="application/excel" ext="xlsx")
   subject="State Population Trend for &state";
```

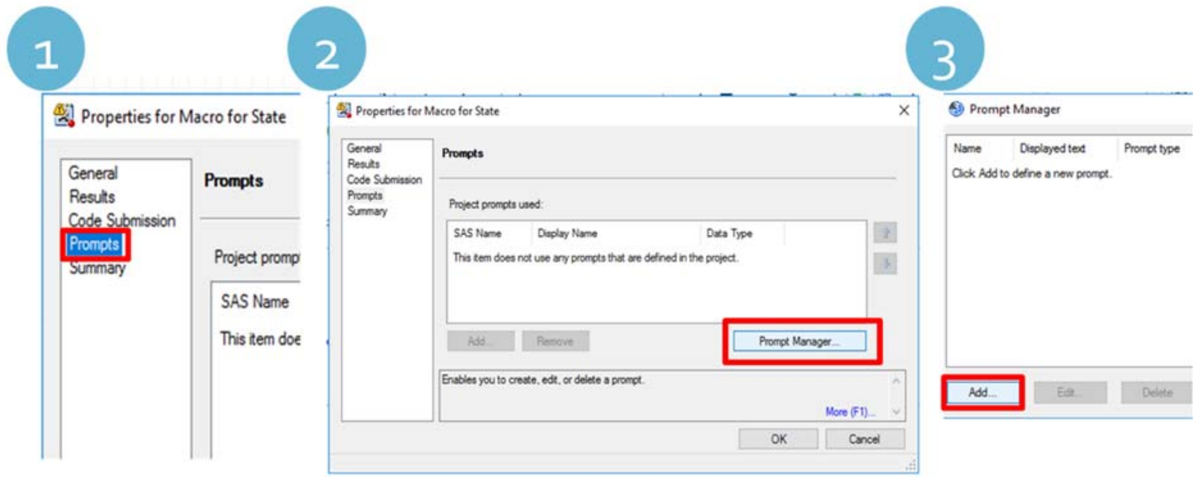
To trigger the email, call your fileref from a DATA _NULL_ statement. Here you can also write body text for the email. In this example, you indicate that the State population trends are attached:

```
/* Call the temporary email file and add body text */
DATA _null_;
   file em;
   put 'Your state population Trend is attached';
   put /;
RUN;
```

STEP 6: CREATE THE MACRO VARIABLES AS PROMPTS IN ENTERPRISE GUIDE

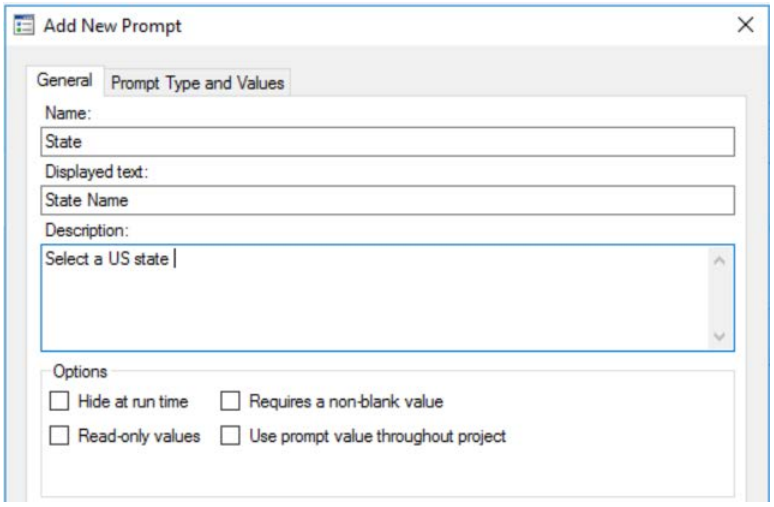
Creating prompts in SAS Enterprise Guide will allow you to test your code as a stored process and to run it directly from the Stored Process Web Application without a custom front end. If you do make a custom front end, you'll use that to pass parameters back to SAS instead of the prompting framework. To create prompts in Enterprise Guide, right-click in the code window of the program, select "Properties", select "Prompts", "Prompt Manager…", and then "Add…".

Figure 7. The Prompt Manager in Enterprise Guide



When you create your prompt, the name should be the same as the macro variable it is replacing, in this example 'state'. The displayed text is like a label, it does not need to conform to SAS naming conventions. The description will appear under the name, this is an opportunity to give the user more instructions about what to enter. Because the 'state' macro variable needs to be defined for your code to run, check the box saying that it 'requires a non-blank value'.

Figure 8. The 'Add New Prompt' Wizard, First Tab



Add New Prompt

General | Prompt Type and Values

Name:
State

Displayed text:
State Name

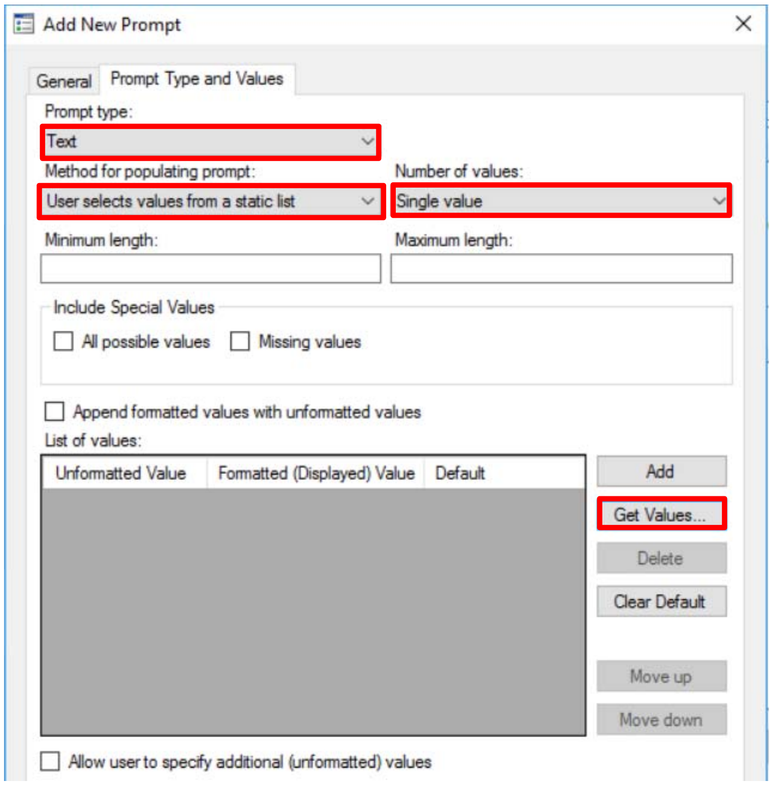
Description:
Select a US state

Options
☐ Hide at run time ☐ Requires a non-blank value
☐ Read-only values ☐ Use prompt value throughout project

On the second tab of that same window, you can define the prompt type and values. For this example a simple text input box would be sufficient, but it is also possible to help the users out and provide all the state options in a drop-down menu. This is an example of error prevention, reducing opportunities for someone to get unexpected results from, say, mistyping a state name.

The prompt type defaults to text, leave this, and then specify that the user selects a single value from a static list. On the bottom half of the screen, select 'Get Values' to import a list of state names from the SASHELP.US_STATES dataset.

Figure 9. 'Add New Prompt' Wizard, Second Tab



Add New Prompt

General | Prompt Type and Values

Prompt type:
Text

Method for populating prompt:
User selects values from a static list

Number of values:
Single value

Minimum length:

Maximum length:

Include Special Values
☐ All possible values ☐ Missing values

☐ Append formatted values with unformatted values

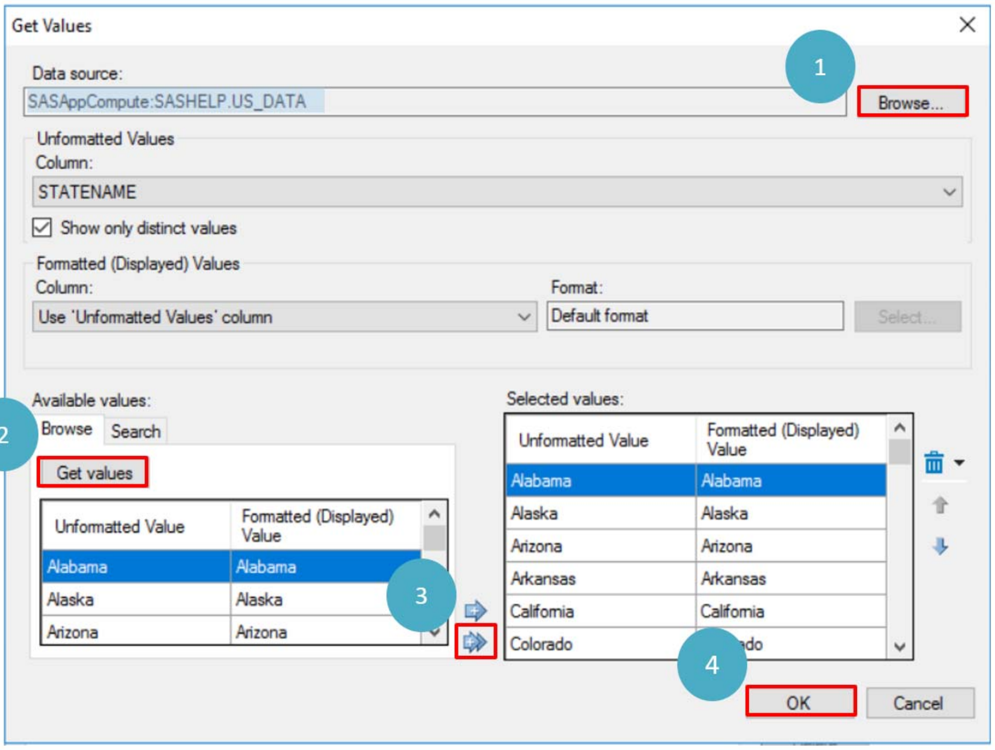
List of values:

Unformatted Value	Formatted (Displayed) Value	Default

☐ Allow user to specify additional (unformatted) values

In the 'Get Values' wizard, browse for SASHELP.US_DATA. On the bottom left, select the 'Get values' button to pull in a distinct list of states. The double blue arrow with the '+' will move all those state names into the ‘Selected values’ panel. Selecting ‘OK’ will save this and close the wizard. Back in the ‘Add New Prompt’ window, select ‘OK’ to create your prompt and close the window.

Figure 10. ‘Get Values’ Wizard


Get Values

Data source: SASAppCompute:SASHELP.US_DATA

Unformatted Values
Column: STATENAME
☑ Show only distinct values

Formatted (Displayed) Values
Column: Use 'Unformatted Values' column
Format: Default format

Available values:

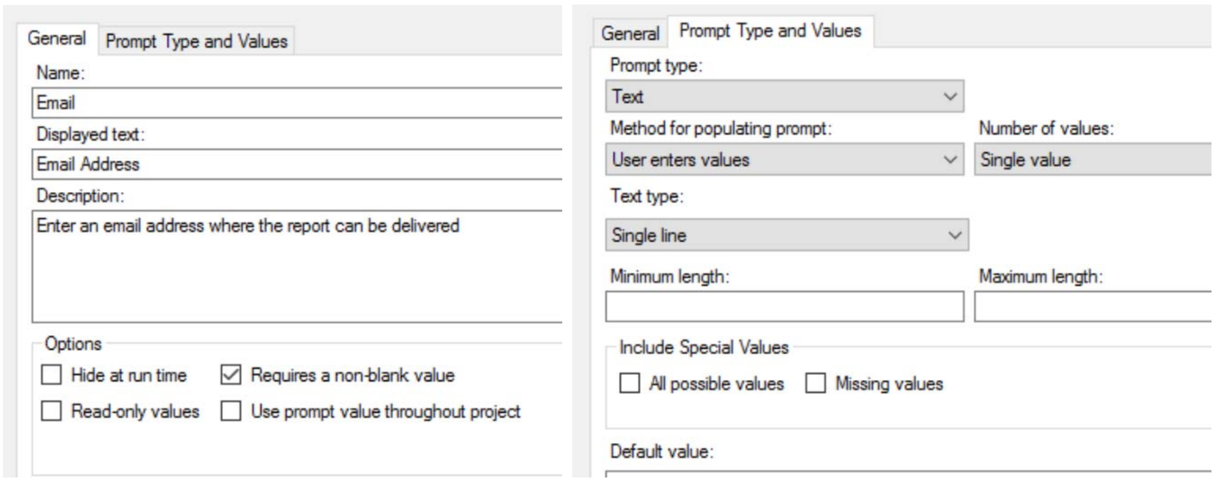
Unformatted Value	Formatted (Displayed) Value
Alabama	Alabama
Alaska	Alaska
Arizona	Arizona

Selected values:

Unformatted Value	Formatted (Displayed) Value
Alabama	Alabama
Alaska	Alaska
Arizona	Arizona
Arkansas	Arkansas
California	California
Colorado	[illegible]ado

Add a second prompt to the code file for ‘email’.

Figure 11. 'Add New Prompt' Wizard for Email


General | Prompt Type and Values

Name: Email
Displayed text: Email Address
Description: Enter an email address where the report can be delivered

Options
☐ Hide at run time ☑ Requires a non-blank value
☐ Read-only values ☐ Use prompt value throughout project

General | Prompt Type and Values

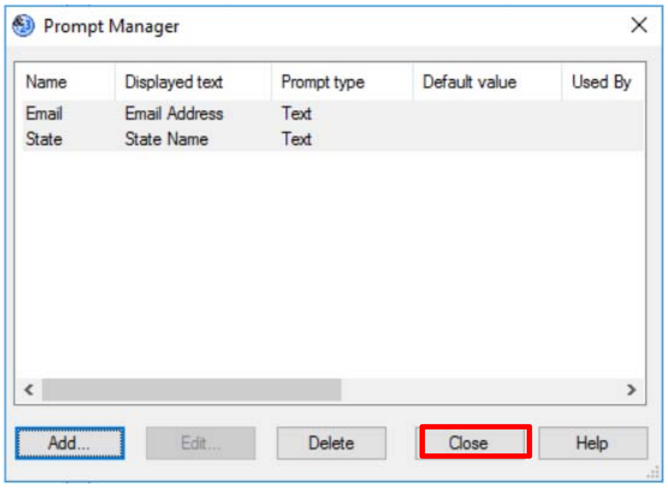
Prompt type: Text
Method for populating prompt: User enters values
Number of values: Single value
Text type: Single line
Minimum length:
Maximum length:

Include Special Values
☐ All possible values ☐ Missing values

Default value:

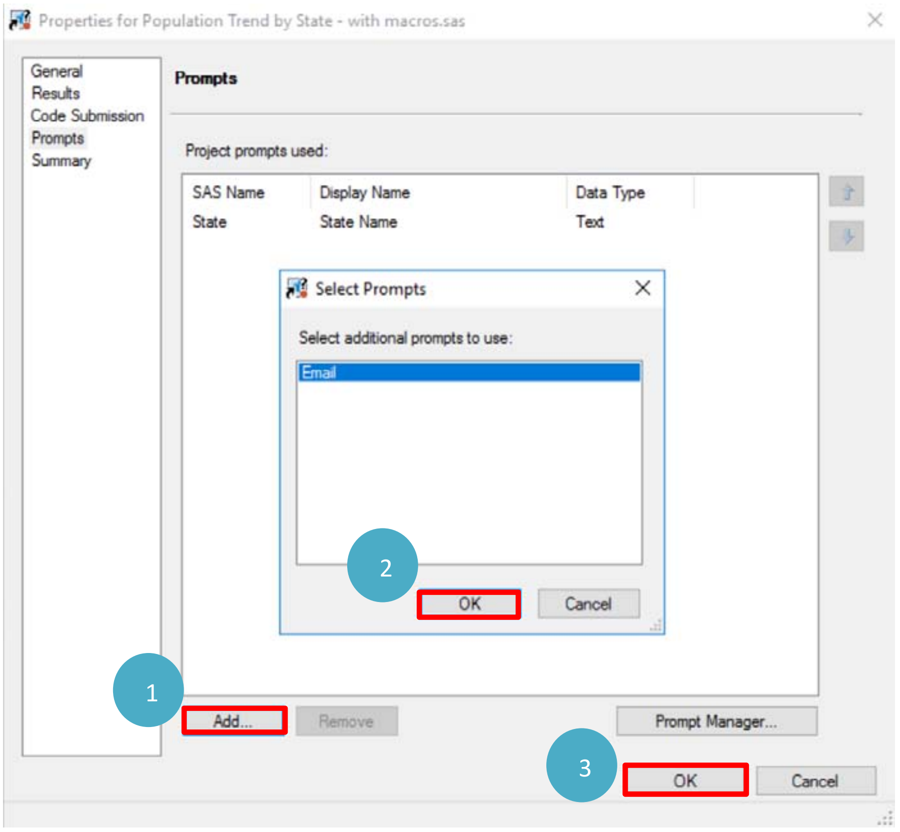
Now that ‘state’ and ‘email’ are defined as prompts, add them to your code.

Figure 12. Prompt Manager


Prompt Manager

Name	Displayed text	Prompt type	Default value	Used By
Email	Email Address	Text		
State	State Name	Text		

Figure 13. Add Prompts to Project


Properties for Population Trend by State - with macros.sas

General
Results
Code Submission
Prompts
Summary

Prompts

Project prompts used:

SAS Name	Display Name	Data Type
State	State Name	Text

Select Prompts

Select additional prompts to use:

Email

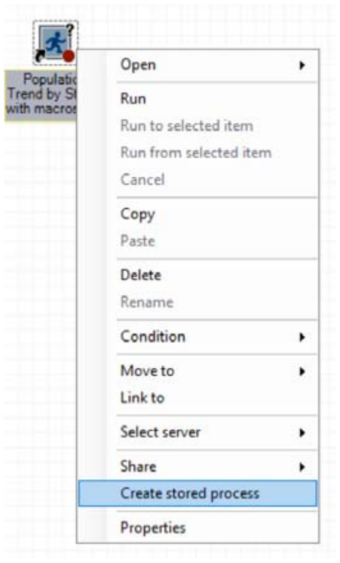
The values for the macro variables ‘state’ and ‘email’ will now be supplied by the user via the prompts, so make sure to delete the hard coded %let statements from your code:

```
~~%let state = District of Columbia;~~
~~%let email = your_email@department.gov;~~
```

STEP 7: TURN THE CODE INTO A SAS STORED PROCESS

Turning the code into a stored process allows users who don’t have SAS installed on their computer to access this code from the web. From Enterprise Guide, right-click on the code node and select “Create Stored Process” to launch the wizard. If your SAS installation does not include a Stored Process Server, this menu option may be grayed out.

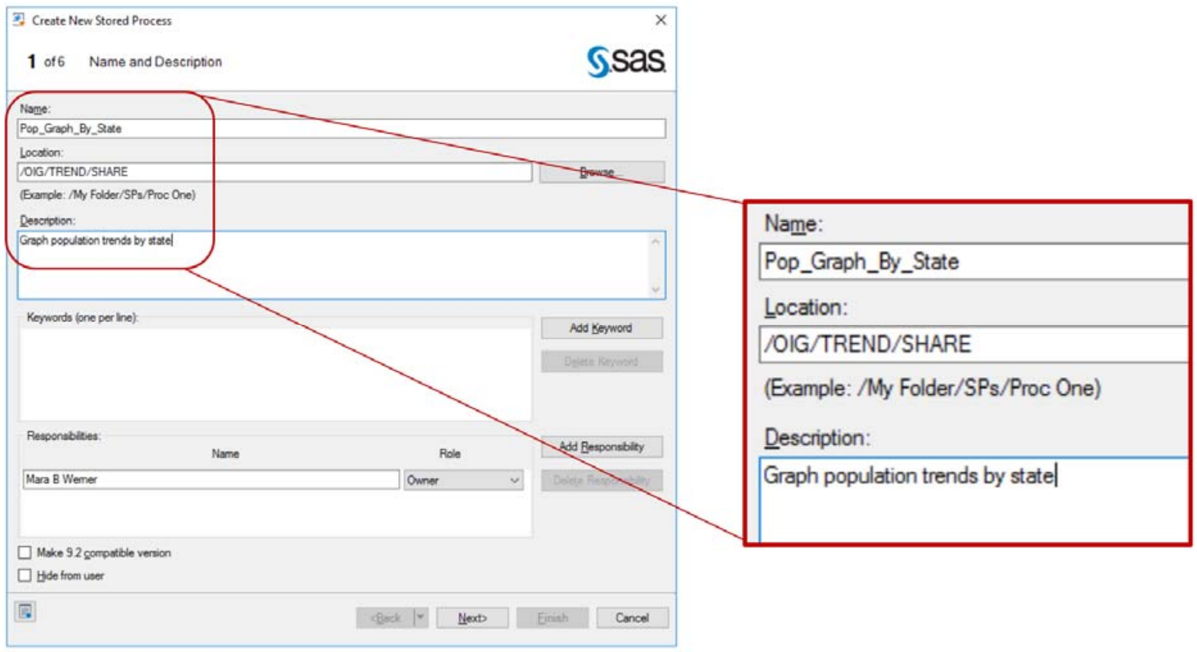
Figure 14. The Right-Click Menu to Create a Stored Process



NAME AND DESCRIPTION

The first screen in the stored process wizard is ‘Name and Description’. In addition to the name and description, you choose a folder location where you will save your stored process. The location defaults to ‘My Folder’ which is convenient for testing and personal use. For a wider audience, chose a server location that other users can also access. If you do not have metadata permissions to save in a common folder you may need admin support. Note: in some server environments, spaces in the stored process name can be a challenge, but SAS itself does not impose any restrictions.

Figure 15. The ‘Name and Description’ Screen in the ‘Create New Stored Process’ Wizard



Create New Stored Process

1 of 6 Name and Description

Name: Pop_Graph_By_State

Location: /OIG/TREND/SHARE

(Example: /My Folder/SPs/Proc One)

Description: Graph population trends by state

Keywords (one per line):

Responsibilities:

Name	Role
Mara B Werner	Owner

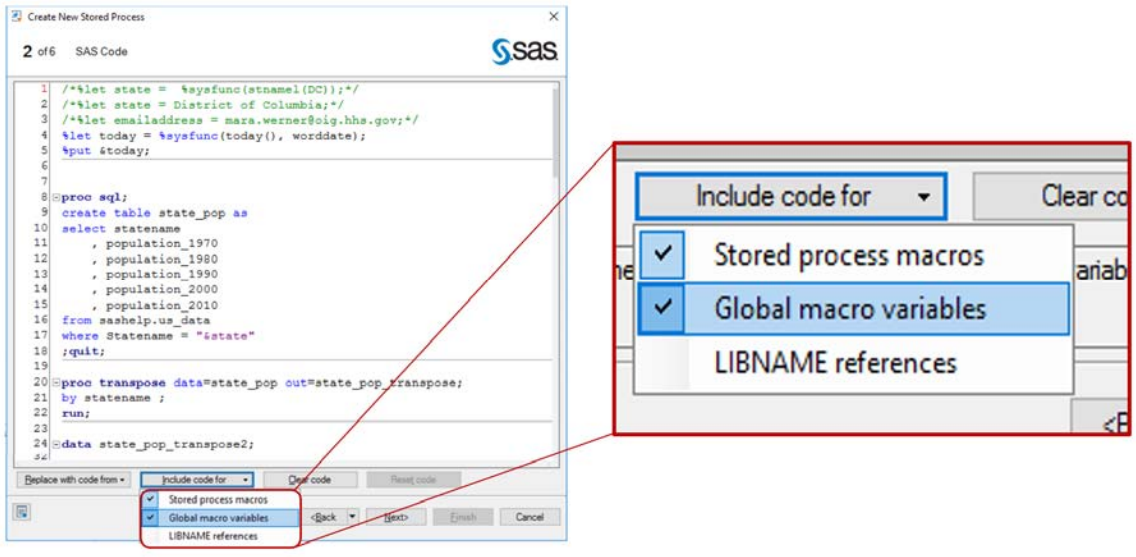
☐ Make 9.2 compatible version

☐ Hide from user

SAS CODE

The second screen shows your SAS code. Avoid the temptation to edit your code on this screen as there is no version control. On the bottom there is a menu to 'Include code for', make sure the first two options are checked.

Figure 16. The 'SAS Code' Screen in the 'Create New Stored Process' Wizard



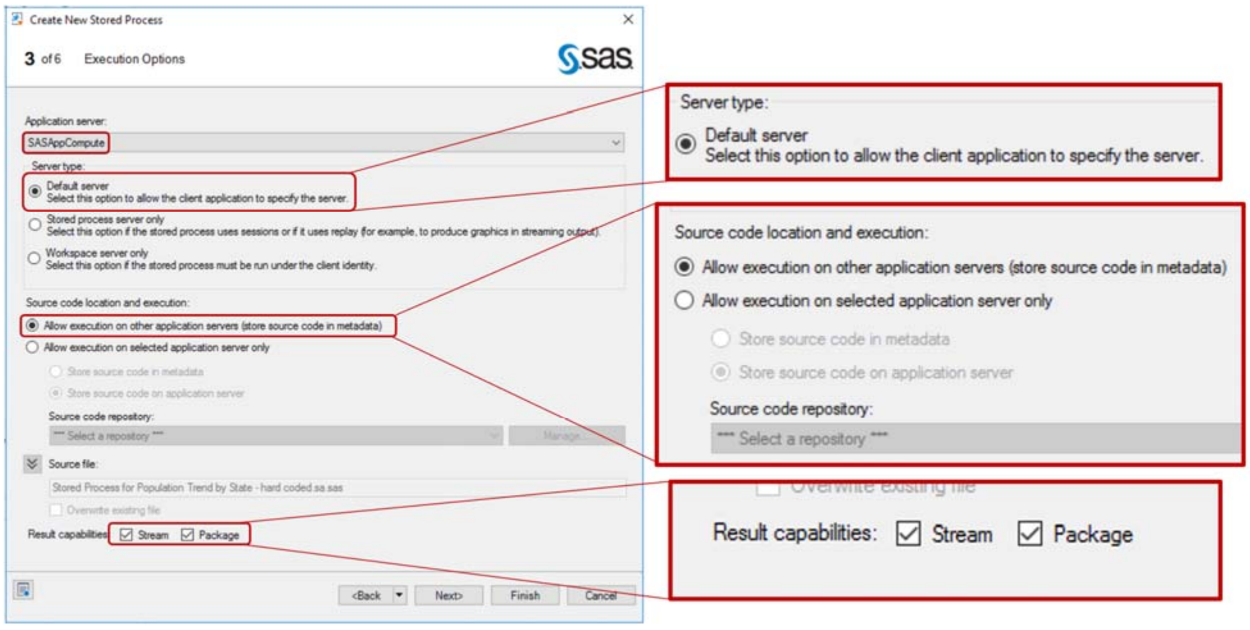
```
/*%let state =  %sysfunc(stnamel(DC));*/
/*%let state = District of Columbia;*/
/*%let emailaddress = mara.werner@oig.hhs.gov;*/
%let today = %sysfunc(today(), worddate);
%put &today;


proc sql;
create table state_pop as
select statename
     , population_1970
     , population_1980
     , population_1990
     , population_2000
     , population_2010
from sashelp.us_data
where Statename = "&state"
;quit;

proc transpose data=state_pop out=state_pop_transpose;
by statename ;
run;

data state_pop_transpose2;
```

Include code for:
- ✓ Stored process macros
- ✓ Global macro variables
- LIBNAME references

EXECUTION OPTIONS

On the execution options screen, select a server from the dropdown menu. You have the option to chose whether the stored process should run on the stored process server or the workspace server. In most cases, the web application will run on the stored process server, but choosing 'Default server' will allow it to run on either. For source code location, select the first radio button to store the code in metadata. In result capabilities, make sure both check boxes are marked. 'Stream' allows your trend chart to show in the browser window and 'package' allows SAS to create the PDF and Excel workbook.

Figure 17. The 'Execution Options' Screen in the 'Create New Stored Process' Wizard



Application server: SASAppCompute

Server type:
- ● Default server — Select this option to allow the client application to specify the server.
- ○ Stored process server only — Select this option if the stored process uses sessions or if it uses replay (for example, to produce graphics in streaming output).
- ○ Workspace server only — Select this option if the stored process must be run under the client identity.

Source code location and execution:
- ● Allow execution on other application servers (store source code in metadata)
- ○ Allow execution on selected application server only
  - ○ Store source code in metadata
  - ● Store source code on application server

Source code repository: *** Select a repository ***

Source file: Stored Process for Population Trend by State - hard coded sa.sas

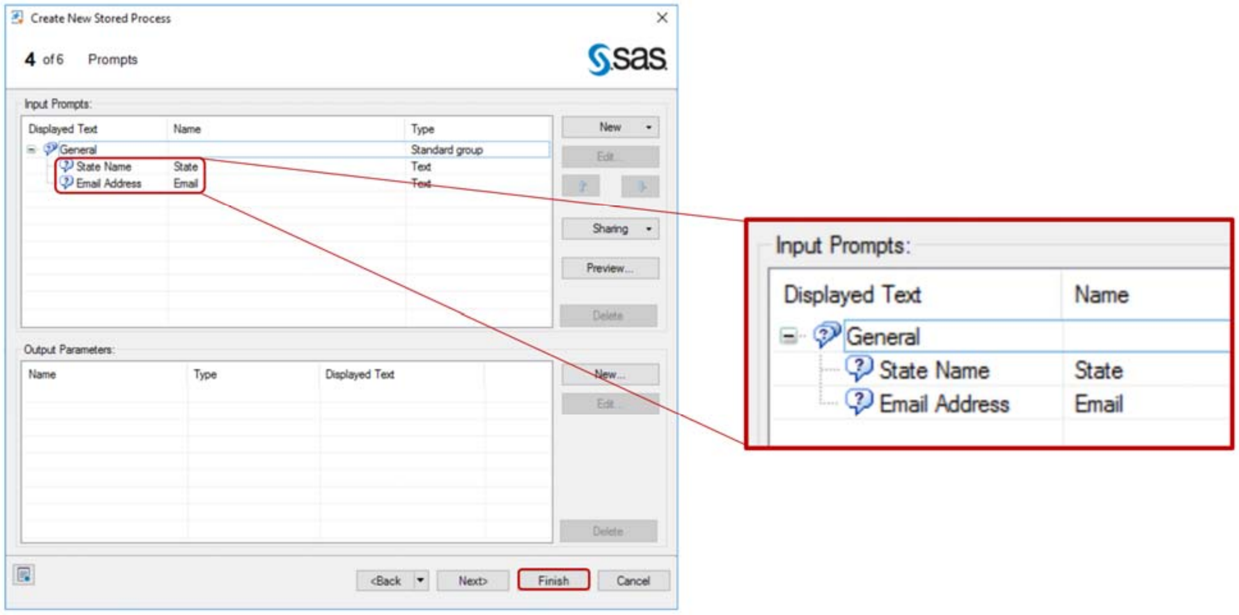
Overwrite existing file

Result capabilities: ✓ Stream ✓ Package

PROMPTS

On the prompt screen, make sure that both the ‘state’ and ‘email’ prompts are selected.

Figure 18. The ‘Prompts’ Screen in the ‘Create New Stored Process’ Wizard

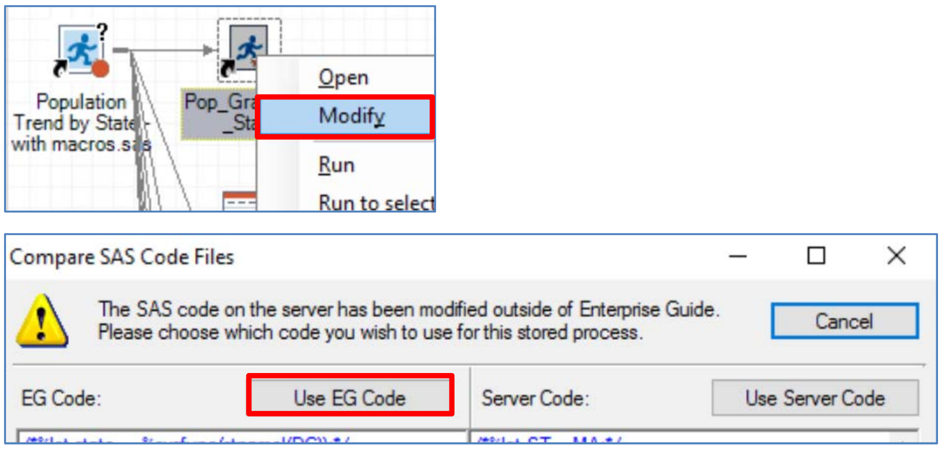


There are a couple more screens in the wizard, but now the ‘Finish’ button is available you can skip the rest of them. Click ‘Finish’ to create your stored process.

MODIFING A STORED PROCESS

While it is possible to modify the code in the ‘SAS Code’ screen of the stored process wizard, it’s risky. There is no backup and no version control on stored processes. For this reason, it is better practice to do all your editing in the SAS code node. After you make changes in the code node, return to the process flow, right click the stored process icon, select ‘Modify’, and then “Use EG Code”.

Figure 19. Right-Click Menu on a Stored Process and the ‘Compare SAS Code Files’ Window

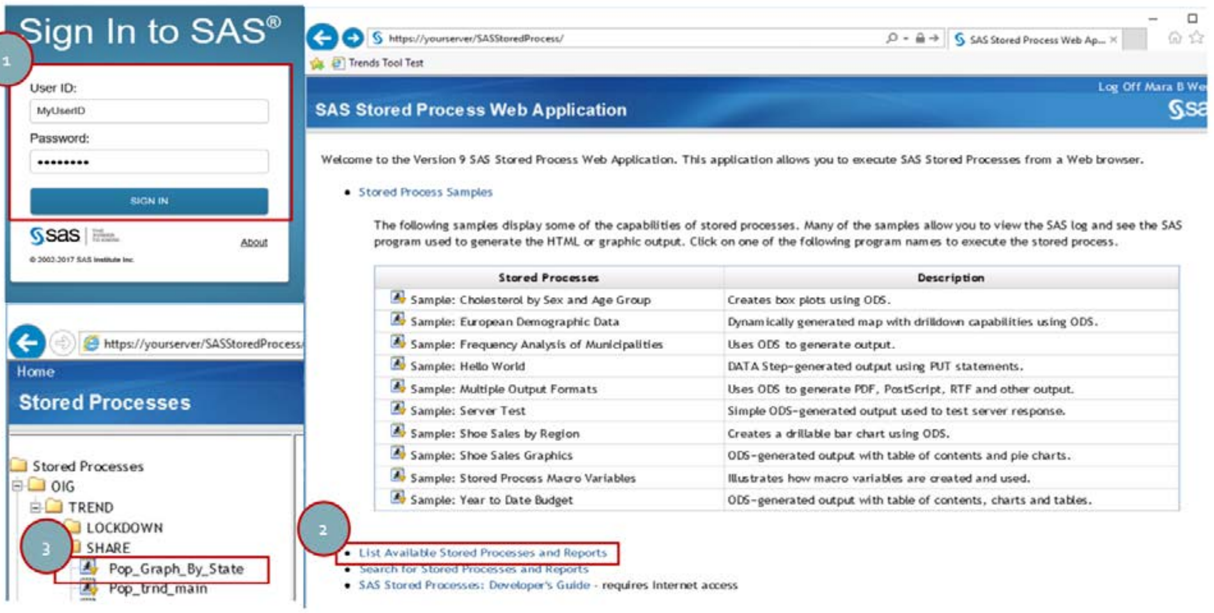


The SAS code on the server has been modified outside of Enterprise Guide.
Please choose which code you wish to use for this stored process.

STEP 8: USING THE SAS STORED PROCESS WEB APPLICATION

Once the stored process is created, if you have the server installation of SAS, it can be accessed from the web using the SAS Stored Process Web Application. The default URL is://*yourserver.com:8080*/SASStoredProcess/do; yours may look different. This will take you to a log in screen, and then to a home page.

Figure 20. Login Screen, Welcome Page, and Folder Tree for the SAS Stored Process Web Application



Sign In to SAS®

User ID: MyUserID

Password: ••••••••

SIGN IN

About

© 2002-2017 SAS Institute Inc.

https://yourserver/SASStoredProcess/

Trends Tool Test

Log Off Mara B We

SAS Stored Process Web Application

Welcome to the Version 9 SAS Stored Process Web Application. This application allows you to execute SAS Stored Processes from a Web browser.

- Stored Process Samples

The following samples display some of the capabilities of stored processes. Many of the samples allow you to view the SAS log and see the SAS program used to generate the HTML or graphic output. Click on one of the following program names to execute the stored process.

Stored Processes	Description
Sample: Cholesterol by Sex and Age Group	Creates box plots using ODS.
Sample: European Demographic Data	Dynamically generated map with drilldown capabilities using ODS.
Sample: Frequency Analysis of Municipalities	Uses ODS to generate output.
Sample: Hello World	DATA Step-generated output using PUT statements.
Sample: Multiple Output Formats	Uses ODS to generate PDF, PostScript, RTF and other output.
Sample: Server Test	Simple ODS-generated output used to test server response.
Sample: Shoe Sales by Region	Creates a drillable bar chart using ODS.
Sample: Shoe Sales Graphics	ODS-generated output with table of contents and pie charts.
Sample: Stored Process Macro Variables	Illustrates how macro variables are created and used.
Sample: Year to Date Budget	ODS-generated output with table of contents, charts and tables.

- List Available Stored Processes and Reports
- Search for Stored Processes and Reports
- SAS Stored Processes: Developer's Guide - requires Internet access

https://yourserver/SASStoredProcess/

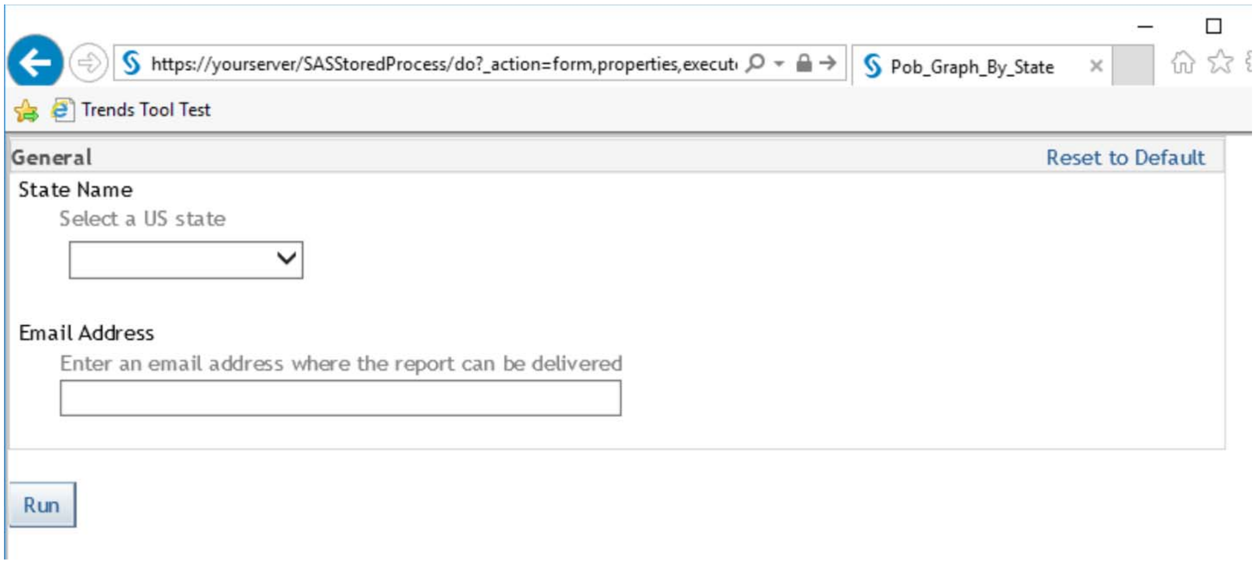
Home

Stored Processes

```
Stored Processes
  OIG
    TREND
      LOCKDOWN
      SHARE
        Pop_Graph_By_State
        Pop_trnd_main
```

Clicking on the name of the stored process ‘Pop_Graph_By_State’ in the browser opens up a new window with a form to fill in values for the ‘State’ and ‘Email’ prompts.

Figure 21. The Default Prompt Page for a Stored Process



https://yourserver/SASStoredProcess/do?_action=form,properties,executi

Pob_Graph_By_State

Trends Tool Test

General — Reset to Default

State Name

Select a US state

Email Address

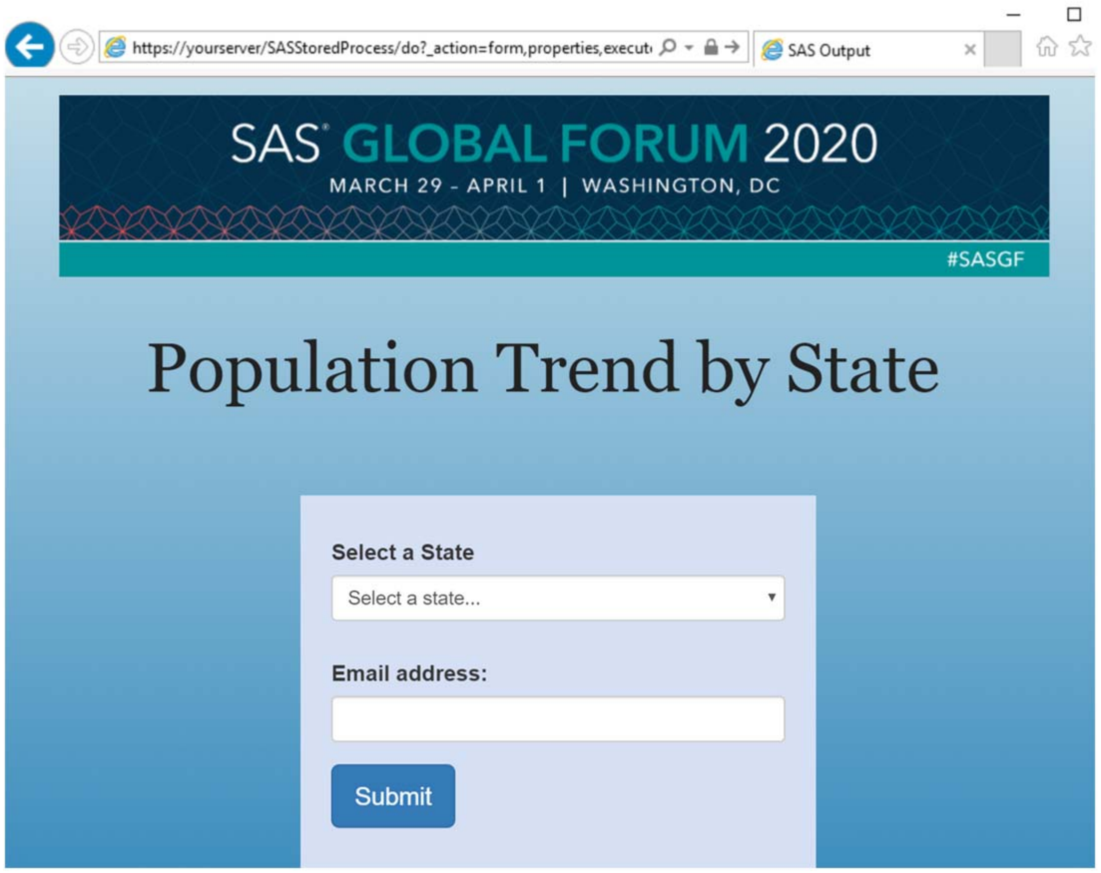
Enter an email address where the report can be delivered

Run

STEP 9: DESIGN A CUSTOM FRONT END

The default prompt page is functional, but it’s not pretty. You can improve the user experience of your web application with a custom html front end.

Figure 22. Example of a Custom Front End for a Stored Process


https://yourserver/SASStoredProcess/do?_action=form,properties,executi
SAS Output
SAS GLOBAL FORUM 2020
MARCH 29 - APRIL 1 | WASHINGTON, DC
#SASGF
Population Trend by State
Select a State
Select a state...
Email address:
Submit

The custom front end shown is written with HTML and the Bootstrap CSS framework. The main component is an HTML form which collects information and stores it as name-value pairs. Using the 'action' attribute, you tell the webpage to send these name-value pairs to your stored process, which will read them in as macro variables. I provide programs for this example front end, but a discussion of the code is outside the scope of this paper. Suffice to say that even without any background in web programming, it is still possible to make a clean looking user interface for your application.

To take advantage of amenities like syntax highlighting, debugging, and auto-completion, you may want to write the web code in an HTML editor rather than SAS. Once you have all the web code written – import it into a SAS dataset (in this example, 'frontend_code') and save the file in the same server location as your stored process.

To call your custom web code, make a second stored process. The second stored process sends all the html code you saved in 'frontend_code' to the '_webout' destination. The browser will receive the web code from SAS and render your user interface:

```
%stpbegin;
%stpend;

*ProcessBody;
DATA _null_;
   set trend.frontend_code;
   file _webout;
   put line;
RUN;
```

CONCLUSION

You have walked through all the steps to make a simple web application. Using this process, you can give anyone in your organization access to summarized data or analytics on-demand in a format that is easy to understand. You can take yourself out of the business of mundane requests and use that time to keep your brain active with more stimulating projects, or build more applications! I hope you find as I have, that this is a fun and interesting challenge, and an opportunity to amplify the reach of your work using SAS.

ACKNOWLEDGMENTS

Many thanks to Scott Hutchison who built the first internal web application with SAS for our organization and showed the rest of us the way. You can read his SGF paper here: https://support.sas.com/resources/papers/proceedings14/1488-2014.pdf

RECOMMENDED READING

- *SAS® 9.4 Stored Processes: Developers Guide, Third Edition*

CONTACT INFORMATION

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

Mara Werner
Department of Health and Human Services, Office of Inspector General
Mara.Werner@oig.hhs.gov