

► Technical Paper

Using SAS® Studio to Open SAS® Enterprise Guide® Project Files (Experimental in SAS Studio 3.6)



Release Information

Content Version: 1.0 March 2017

Author

Jennifer Jeffreys-Chen

Trademarks and Patents

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

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 registered trademarks or trademarks of their respective companies.

ABSTRACT

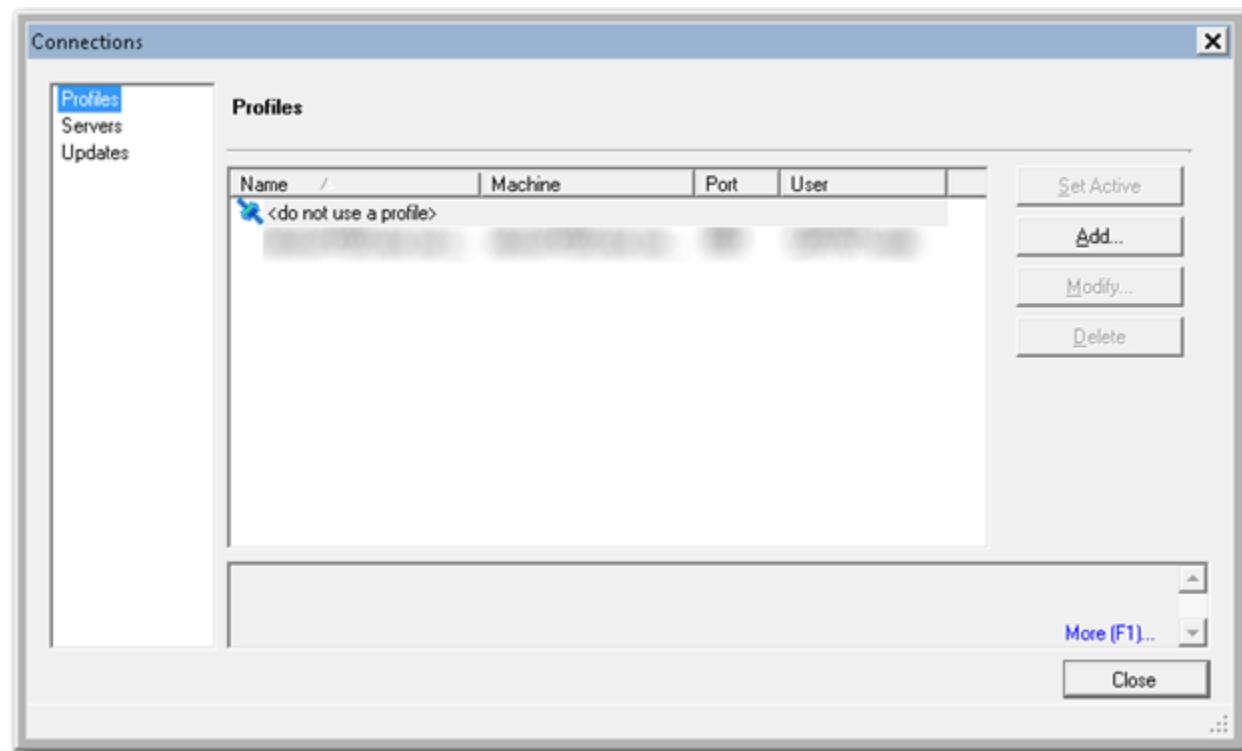
As a SAS® programmer, how often does it happen that you would like to open your SAS® Enterprise Guide® project in SAS® Studio? SAS Studio has a new feature that enables you to do just that! This paper covers how SAS Studio extracts SAS Enterprise Guide process flows from a SAS Enterprise Guide project file and converts the nodes to elements in a SAS Studio process flow.

INTRODUCTION

SAS Studio users have been requesting support for SAS Enterprise Guide projects since the introduction of the Visual Programming Perspective and SAS Studio process flows. SAS Studio 3.6 introduces an experimental feature that extracts process flows from a SAS Enterprise Guide project file (also referred to as an EGP file in this document) and converts the process flow to a SAS Studio process flow. Elements that are not supported by SAS Studio are either converted to different node types or are omitted from the process flow. A Conversion Report is provided to explain what happened to the nodes in the conversion process. This paper will help users understand how elements are converted and when a converted process flow might require manual intervention before it will run. SAS Studio Administrators must [enable this feature](#) since it is off by default. Users of SAS University Edition will find that this functionality is always enabled.

SERVER CONTEXT

Server context is very important for EGP files. In SAS Enterprise Guide, the active connection is saved in the project metadata . (To view the active connection in SAS Enterprise Guide, select **Tools -> Connections**. In the Connections dialog box, select **Profiles**).



Display 1 – Connections Dialog Box in SAS Enterprise Guide

If you try to open an EGP file in SAS Enterprise Guide that has a different connection than the current active connection, SAS Enterprise Guide shows a message.



Display 2 – Message for Mismatched Connections

Furthermore, the data and code nodes in a SAS Enterprise Guide process flow can be associated with different connections. These connections are available to the user when constructing the process flow in SAS Enterprise Guide.

Since SAS Enterprise Guide is a desktop application, it has easy access to the local file system and SAS server. You can also access any workspace server environments defined in the SAS Metadata Server that your SAS Enterprise Guide session is configured to use.

In contrast, SAS Studio is a web interface that can have just one SAS server configuration at a time. The Single-User edition of SAS Studio has access only to the local file system and the SAS installation on the desktop. The Basic edition of SAS Studio has access only to the single workspace server that it is configured to use and the file system available to that server. Local files must be uploaded for use in the Basic edition. Finally, the Enterprise edition of SAS Studio can have multiple workspace servers defined in the SAS Metadata Server that it is configured to use. However, a user can connect to only one workspace server at a time within SAS Studio.

It is important to understand these differences when opening an EGP file in SAS Studio. If the environment for the SAS Enterprise Guide project and the environment for SAS Studio do not match or if the data or executable nodes in the EGP do not match the SAS Studio environment where the conversion occurs, the conversion report will provide warnings about the mismatches.

The following display shows an example of the type of message that you will see if your EGP environment metadata does not match your SAS Studio environment.

Conversion Report Details:

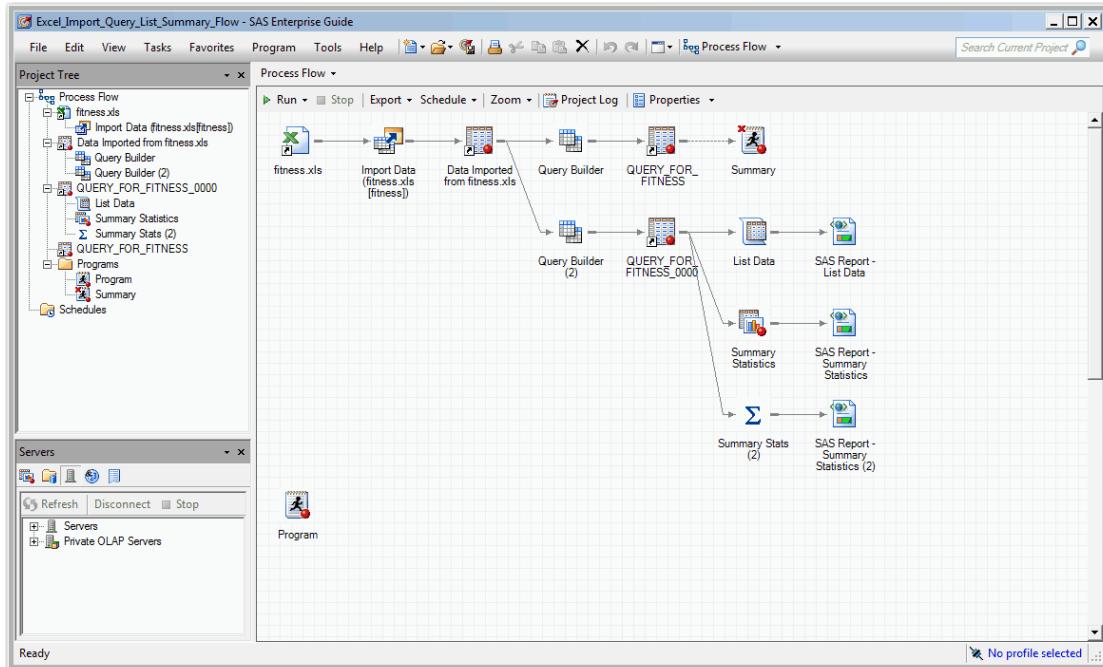
- NOTE: EG Version is 7.1
- NOTE: EG Name: 5_MultiDataSources
- NOTE: Modified By EG ID: [REDACTED]
- NOTE: Modified By: J. Jeffreys-Chen
- NOTE: Modified by EG version: 7.100.1.2651
- ERROR: The SAS Studio SAS connection is local, but the EGP has a Metadata Server connection defined. The converted process flow may not run.
- NOTE: Enterprise Guide Metadata Server Name: [REDACTED]
- NOTE: EG Metadata Host: [REDACTED]
- NOTE: EG Metadata Port: 8561
- WARNING: Process flow view Grid set to "True". Process flow Grid is not supported in SAS Studio.
- WARNING: Node "TaskDetails1" will not appear in the process flow because "Note" nodes are not supported in SAS Studio process flows. Note contents:
- Normal Regression, Log link: Example 31.2
- Input dataset: Jazz Test Data\normal
- Assign variables:
Dependent variable: y
Quantitative variable: x
- Model panel: set x as a [Main] effect
- Model Options panel:
cat Distribution -> Normal and Link Function -> Log

Display 3 - Mismatched SAS Studio and EGP Environments

OPENING SAS ENTERPRISE GUIDE PROJECT FILES

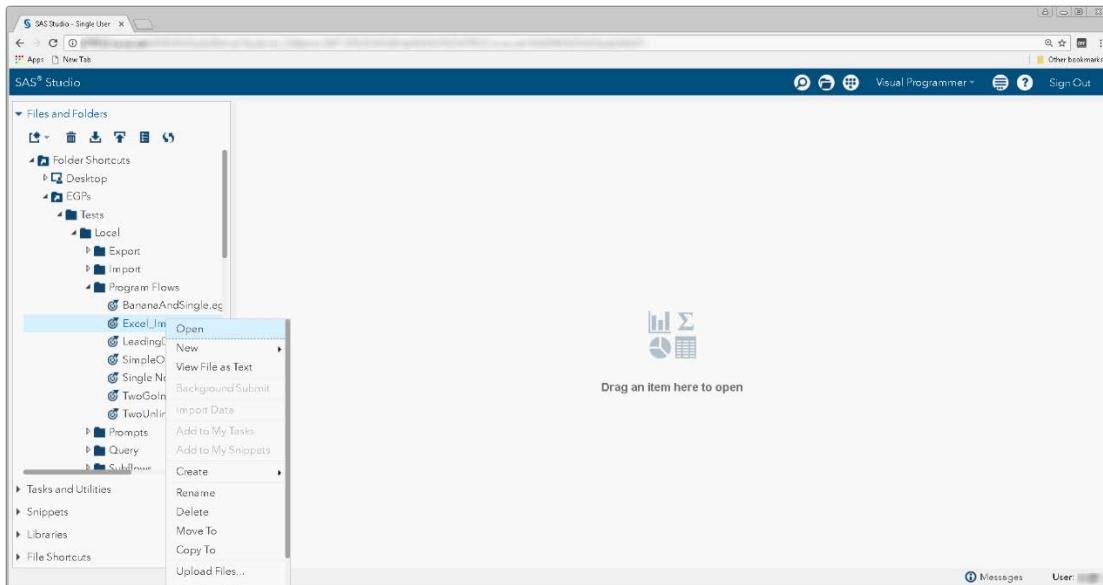
Opening an EGP file in SAS Studio results in a new Process Flow tab that contains the converted process flows. A **CONVERSION REPORT** tab indicates how EGP nodes were converted, and notes any potential problems with the converted SAS Studio process flow (.cpf file).

Here is a process flow displayed in SAS Enterprise Guide.



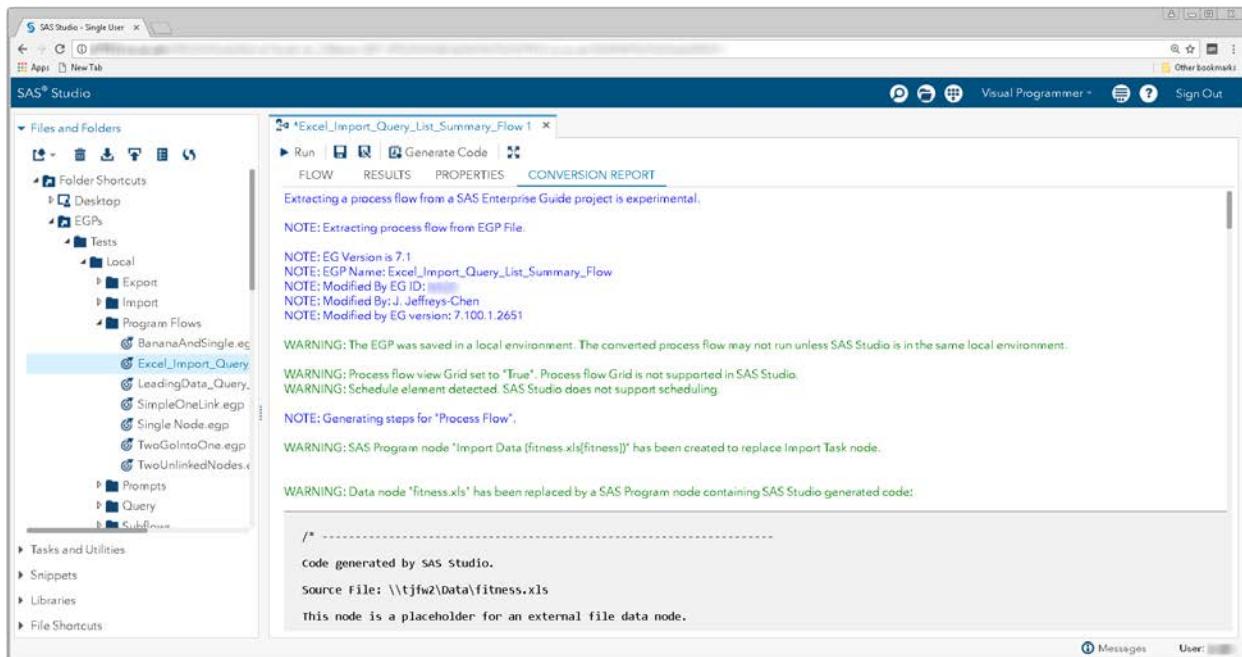
Display 4 – Process Flow in SAS Enterprise Guide

To open the EGP file in SAS Studio, open the Visual Programmer perspective in SAS Studio. Traverse to the directory that contains the EGP file. To open the file, either double click on the file or select **Open** from the pop-up menu.



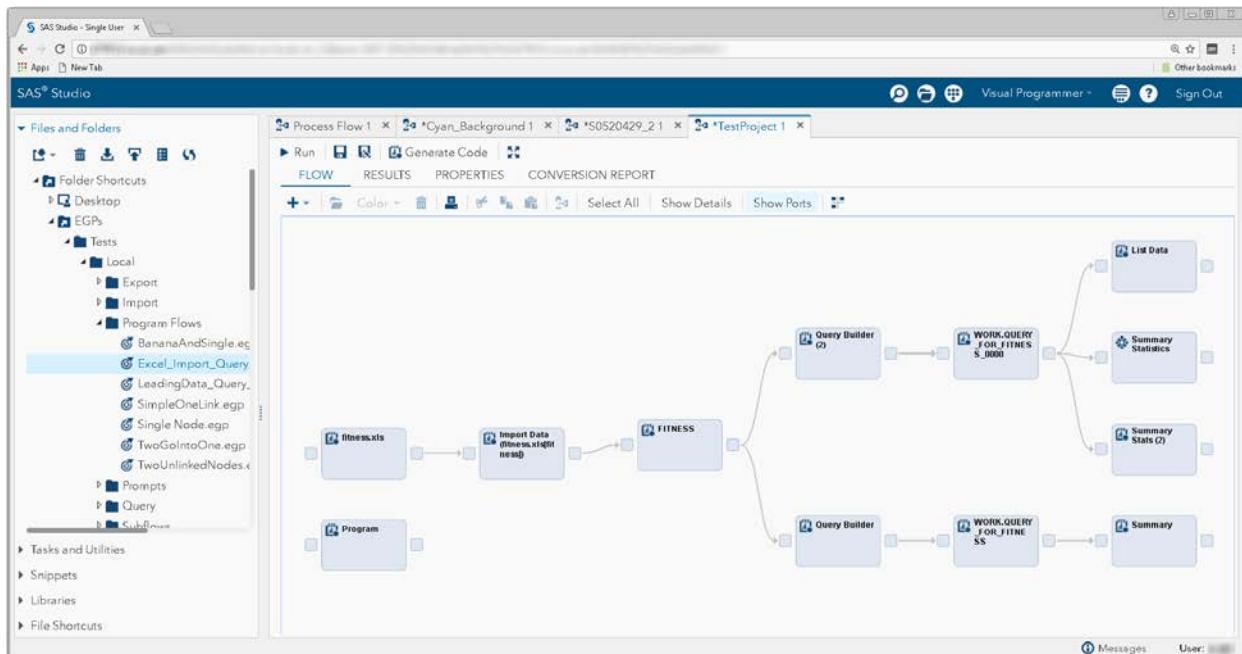
Display 5 - Opening an EGP File in SAS Studio

SAS Studio displays a process flow tab with the **CONVERSION REPORT** tab selected.



Display 6 – Contents of CONVERSION REPORT Tab

The converted flow appears in the **FLOW** tab. This display shows the converted process flow from Display 4.



Display 7 – A SAS Enterprise Guide Project in SAS Studio

You can save the converted flow to a SAS Studio process flow (.cpf) file, and the conversion report will persist with it. It is important to understand that the conversion report is static and will not reflect any changes made to the converted process flow.

The following processing occurs when an EGP file is converted:

- The EGP file is "unpacked" into a temporary directory.
- The project and its process flow nodes are analyzed to determine if the [server context](#) is consistent with the SAS Studio context where the EGP is being opened. Mismatched SAS connection contexts will result in conversion warnings.
- Where possible, nodes in the SAS Enterprise Guide process flow are converted to SAS Studio nodes.
- Multiple process flows in the EGP file become sub-flows in a single SAS Studio process flow.
- Prompts are replaced by generated macro code in SAS Studio. This macro code is added to the beginning of the code that depends on the prompts.
- The "unpacked" temporary directory is deleted unless the debug functionality is enabled in SAS Studio.

SUPPORTED NODES

Assuming that the environments for the EGP file and SAS Studio match, here are the node conversions (from SAS Enterprise Guide to SAS Studio) that you can expect.

CONVERTED

- [Program](#)
- [SAS Studio Tasks](#)

CONVERTED TO OTHER NODE TYPES

- [Data](#)
- [SAS Enterprise Guide Tasks](#)
- [Query Builder](#)
- [Import Data](#)
- [Export File](#)

NOT CONVERTED

- [Note](#)
- Stored processes
- OLAP cubes
- Log
- Project log
- ODS result

CONVERSION REPORT

The **CONVERSION REPORT** tab for a converted process flow provides information about the conversion process.

Here are sections in the report:

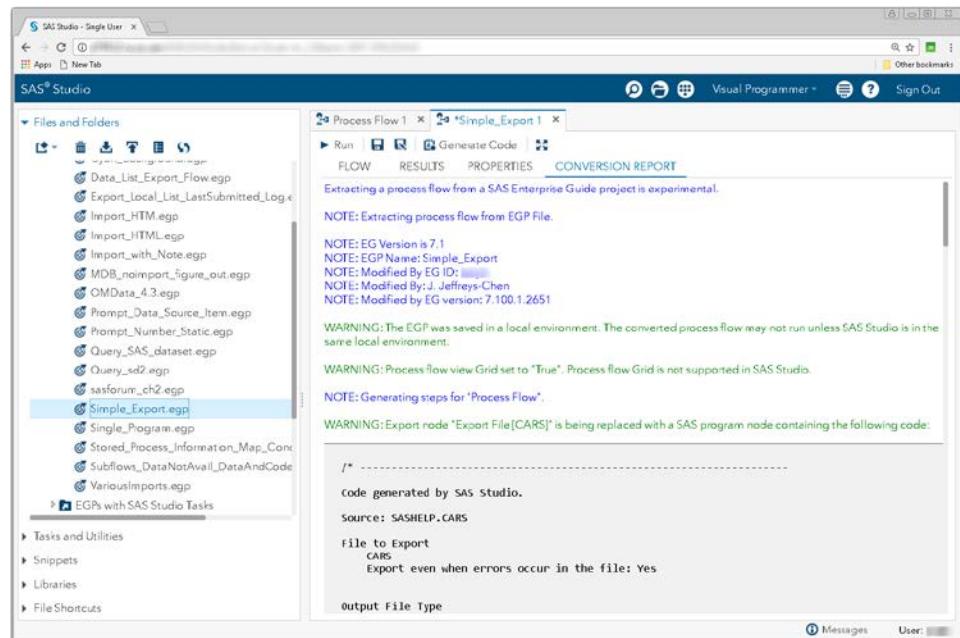
- an EGP environment summary
- a conversion log
- a node summary

ENVIRONMENT SUMMARY

The Environment Summary provides information about the environment associated with the EGP. Attributes in this section include the following information, which is extracted directly from the EGP metadata.

- EG Version – release of SAS Enterprise Guide where the EGP was created
- EGP Name – the name of the EGP
- Modified by EG ID – the user ID of the last person to modify the EGP
- Modified by – the name of the last person to modify the EGP
- Modified by EG version – the release of SAS Enterprise Guide where the EGP was last saved

If your SAS connection in SAS Studio does not match the SAS connection in SAS Enterprise Guide where the EGP was last saved, you will get an error. If the connections might not match, you get a warning. Here is an example of when you might get a warning. You are using the Single-User edition of SAS Studio, and the EGP was saved on a desktop installation of SAS Enterprise Guide. If these local connections are not the same, there could be a problem. It is very important to make sure that the connections in SAS Studio and SAS Enterprise Guide are the same in order for the converted SAS Studio process flow to be correct.



Display 8 - Environment Summary in the Conversion Report

The Environment Summary can also contain general warnings about the SAS Studio and SAS Enterprise Guide process flows. For example in the previous display, there is a warning stating that the process flow grid in SAS Enterprise Guide is not supported in SAS Studio.

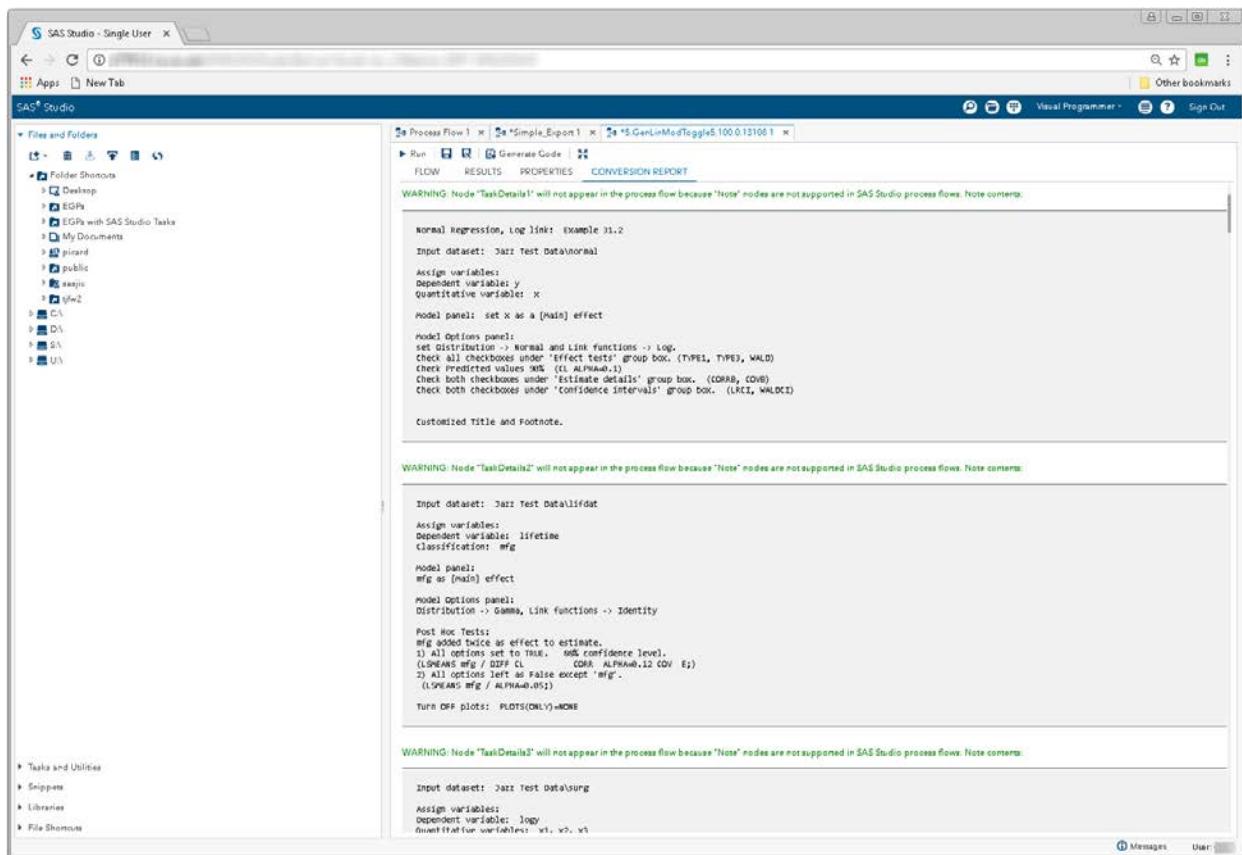
CONVERSION LOG

The conversion log shows the contents of any Note nodes from the SAS Enterprise Guide process flow as well as a running summary of the conversion activity. If you see errors in the report, your converted process flow probably will not run with the SAS connection that you are using in SAS Studio and will require manual intervention.

Note Contents

Since SAS Studio process flows do not support Note nodes, they will not appear in converted process flows. After the Environment Summary, there is a section that contains the contents of any Note nodes that have been eliminated.

The following example shows Note contents from the [Sample Process Flow](#).



WARNING: Node "TaskDetails1" will not appear in the process flow because "Note" nodes are not supported in SAS Studio process flows. Note contents:

```
Normal Regression, Log link: Example 31.2
Input dataset: JAZZ Test Data|normal
Assign variables:
Dependent variable: y
Quantitative variable: x
Model panel: set x as a [main] effect
Model Options panel:
set Distribution: normal and Link: functions -> Log.
Check both checkboxes under 'Effect tests' group box. (TYPE1, TYPE3, WALD)
Check predicted values ANS. (LL ALMAD)
Check both checkboxes under 'Estimate details' group box. (CORR, COVB)
Check both checkboxes under 'Confidence intervals' group box. (LRCI, WALDCI)

Customized Title and Footnote.
```

WARNING: Node "TaskDetails2" will not appear in the process flow because "Note" nodes are not supported in SAS Studio process flows. Note contents:

```
Input dataset: JAZZ Test Data|lifedat
Assign variables:
Dependent variable: lifetime
Classification: mfg
Model panel:
mfg as [main] effect
Model Options panel:
Distribution -> Gamma, Link functions -> Identity
Post Hoc Test:
mfg added twice as effect to estimate.
1) All options left as false except 'mfg'. (LSCM mfg / DDF CL COV ALPHAV=0.12 (COV E))
2) All options left as false except 'mfg'.
(LSCM mfg / ALPHAV=0.05)

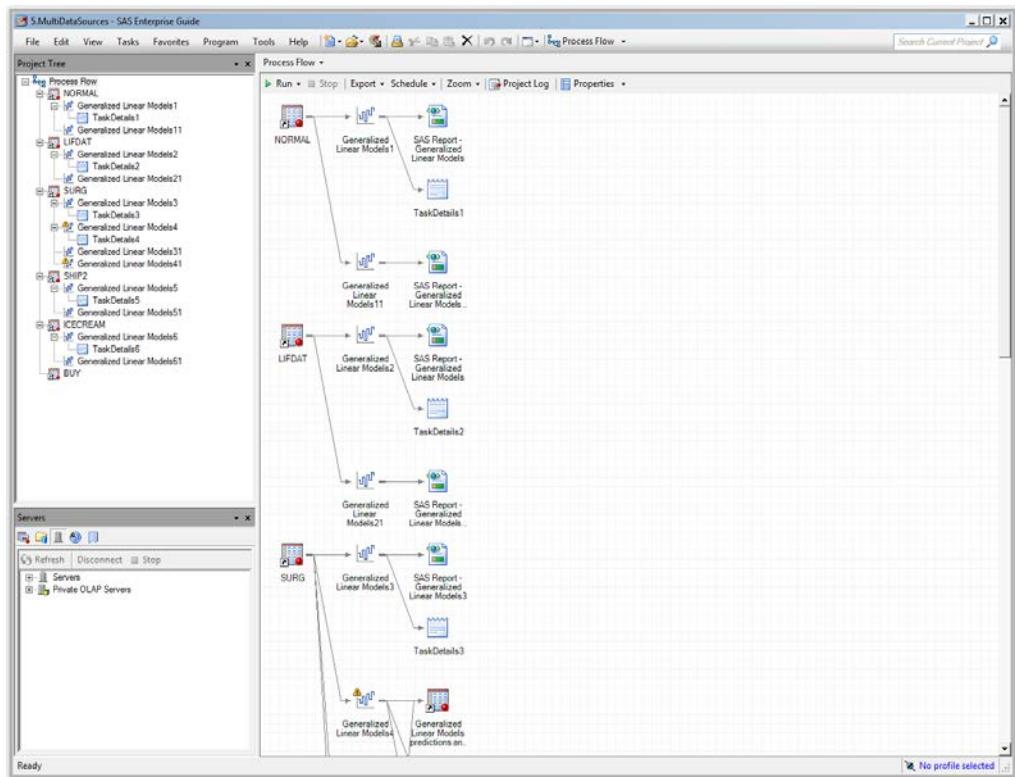
Turn OFF plots: PLOTS(NONE)=NONE
```

WARNING: Node "TaskDetails3" will not appear in the process flow because "Note" nodes are not supported in SAS Studio process flows. Note contents:

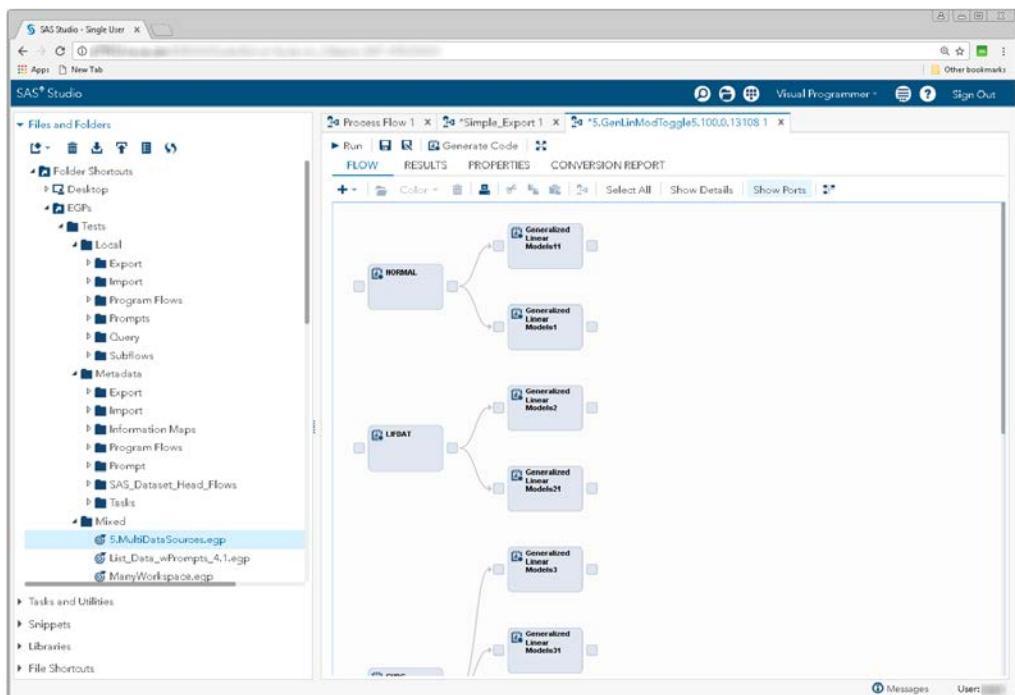
```
Input dataset: JAZZ Test Data|surg
Assign variables:
Dependent variable: logy
Independent variables: v1, v2, v3
```

Display 9 - Contents of Eliminated Note Nodes

Sample Process Flow



Display 10 - Sample Process Flow in SAS Enterprise Guide

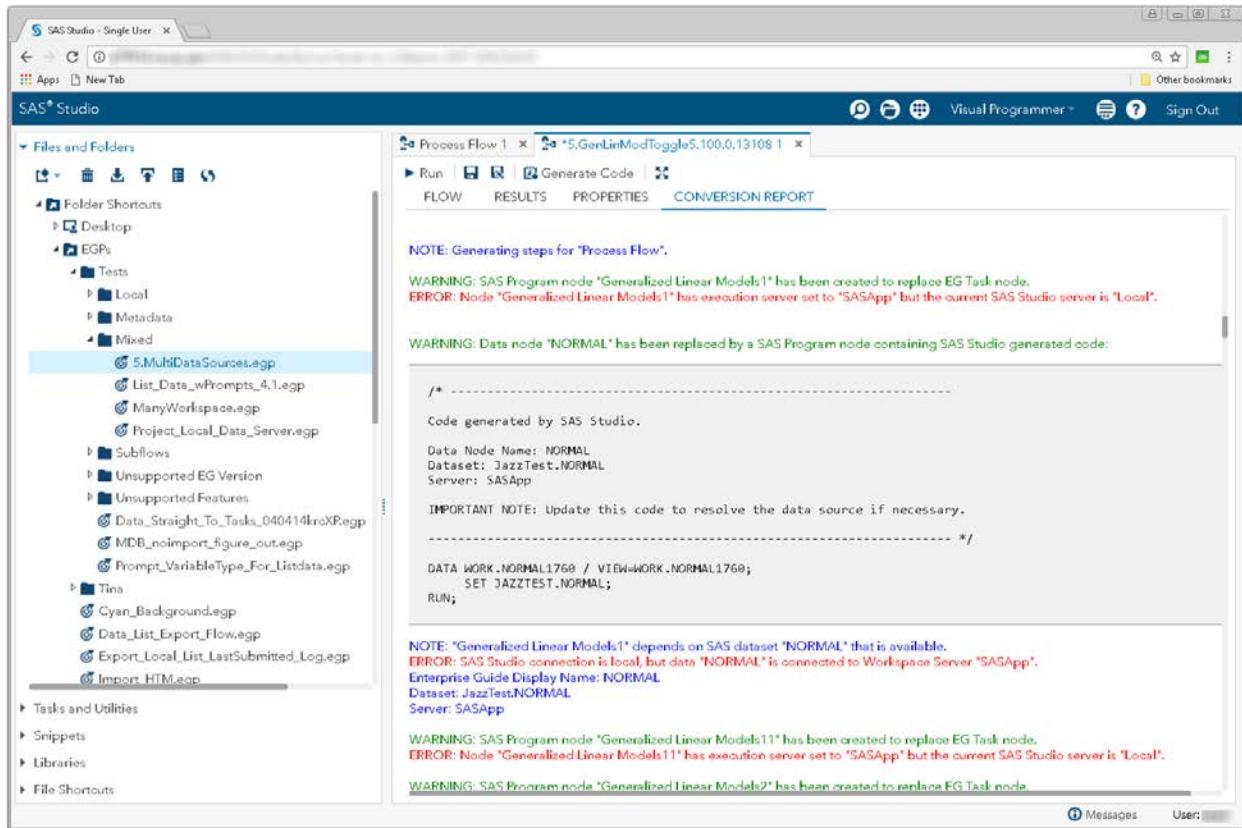


Display 11 - Sample Process Flow in SAS Studio

Generating Steps, Node Connectors, and Node Links

After the Note section of the Conversion Report, there is a running log of the conversion of the steps, ports, and connectors. Any warnings or errors for these items are noted.

In the following example, there are environment errors for SAS Enterprise Guide Task and Data nodes, conversion warnings for the tasks, and a conversion warning for a Data node.



Display 12 – Running Log of Conversion Steps

NODE SUMMARY

The Node Summary section of the Conversion Report provides a list of nodes that are included in the converted process flow and a list of nodes that are not included. Both lists are in alphabetical order. The node names are in bold, and any errors and warnings associated with the nodes are listed below the node name.

The following displays show Node Summary information for the [Sample Process Flow](#).

This screenshot shows the SAS Studio interface with the "Process Flow 1" tab selected. The left sidebar shows a file tree with various EG files and subflows. The main pane displays the "CONVERSION REPORT" tab, which contains the "Steps converted:" section. It shows a warning message: "WARNING: Data node 'BUY' has been replaced by a SAS Program node containing SAS Studio generated code." Below this, there is a code snippet generated by SAS Studio:

```
/* -----
Code generated by SAS Studio.

Data Node Name: BUY
Dataset: SASHELP.BUY
Server: SASApp

IMPORTANT NOTE: Update this code to resolve the data source if necessary.

-----
DATA WORK.BUY916 / VIBWWORK.BUY916;
  SET SASHELP.BUY;
RUN;
```

At the bottom of the report, there is another warning message: "WARNING: Data node 'Generalized Linear Models' depends on SAS dataset 'BUY' that is available. Either SAS Studio connection is local, but data 'BUY' is connected to Workspace Server 'SASApp'. Enterprise Guide Display Name: BUY Dataset: SASHELP.BUY Server: SASAPP".

Display 13 – Node Summary Showing Converted Nodes

This screenshot shows the SAS Studio interface with the "Process Flow 1" tab selected. The left sidebar shows a file tree with various EG files and subflows. The main pane displays the "CONVERSION REPORT" tab, which contains the "Steps omitted:" section. It lists several omitted steps, each with its type and ODS result:

- SAS Report - Generalized Linear Models
- SAS Report - Generalized Linear Models1
- SAS Report - Generalized Linear Models2
- SAS Report - Generalized Linear Models3
- SAS Report - Generalized Linear Models31
- SAS Report - Generalized Linear Models41
- SAS Report - Generalized Linear Model51
- SAS Report - Generalized Linear Model61

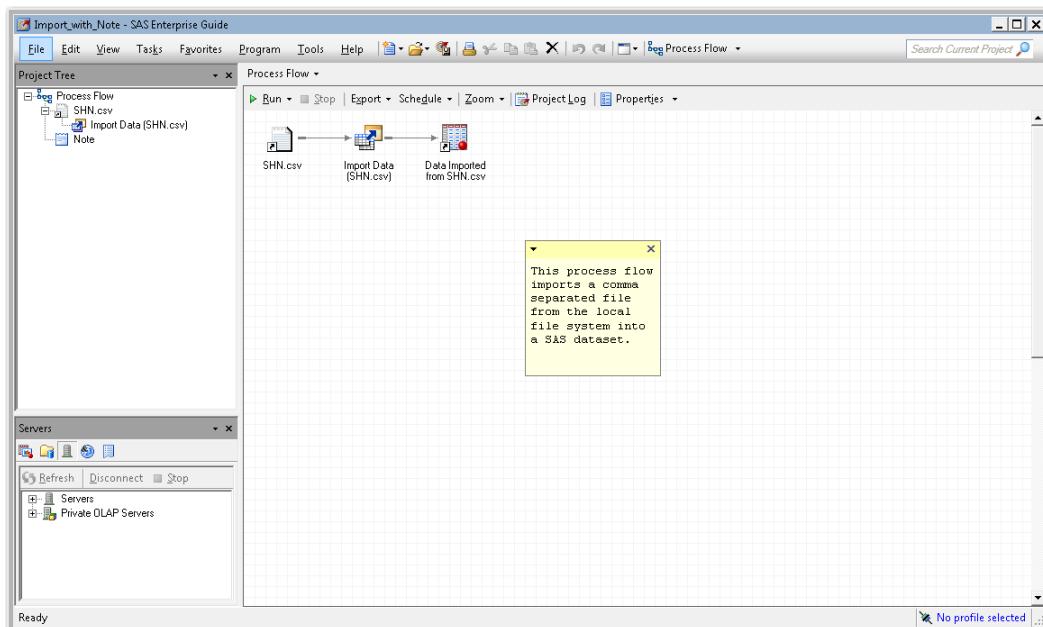
Display 14 - Node Summary Showing Nodes That Could Not Be Converted

ELEMENTS

NOTE NODES

SAS Studio does not support Note nodes, so Note elements do not appear in a converted flow. The contents of the Note elements are written to the [Conversion Report](#).

Here is an example of a Note node in a process flow in SAS Enterprise Guide.



Display 15 - Note Node in Process Flow in SAS Enterprise Guide

When you open the same EGP in SAS Studio, you get a message notifying you that the Note node will not appear in the converted process flow because Note nodes are not supported in SAS Studio. The contents of the Note node is shown in the Conversion Report in the following display.

A screenshot of the SAS Studio interface. On the left is a file browser showing various EGP files. In the center, a tab titled "Import_With_Note.egp" is active. The "CONVERSION REPORT" tab is selected, displaying the following text:

Extracting a process flow from a SAS Enterprise Guide project is experimental.

NOTE: Extracting process flow from EGP File.

NOTE: EG Version is 7.1
NOTE: EGP Name: Import_With_Note
NOTE: Modified By EG ID: [redacted]
NOTE: Modified By J. Jeffreys Chen
NOTE: Modified by EG version: 7.100.1.2651

WARNING: The EGP was saved in a local environment. The converted process flow may not run unless SAS Studio is in the same local environment.

WARNING: Process flow view Grid set to "True". Process flow Grid is not supported in SAS Studio.

WARNING: Node "Note" will not appear in the process flow because "Note" nodes are not supported in SAS Studio process flows.
Note contents:

this process flow imports a comma separated file from the local file system into a SAS dataset.

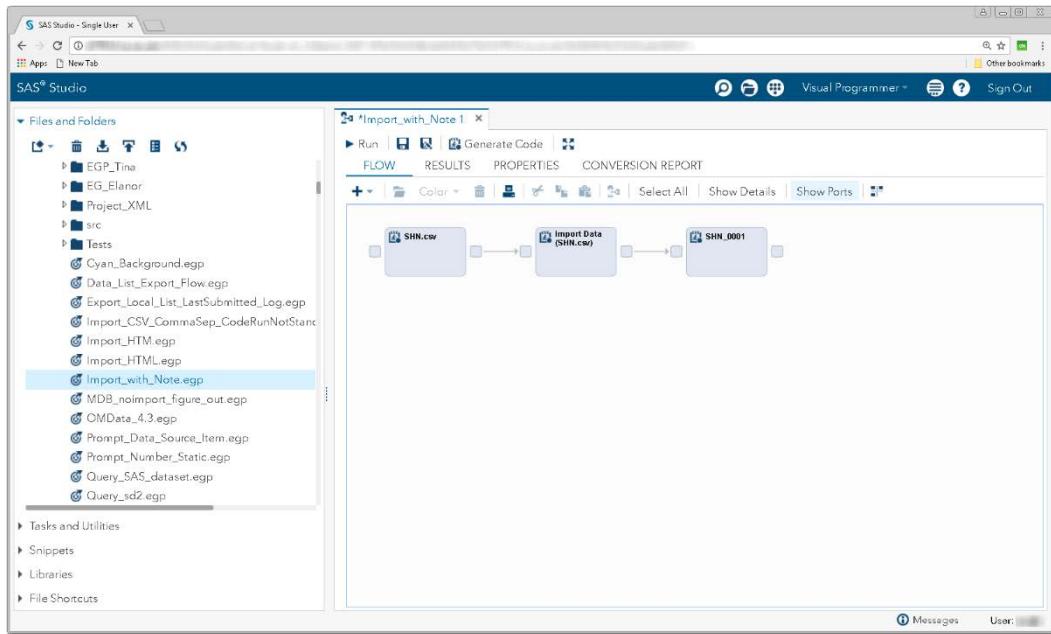
NOTE: Generating steps for "Process Flow".

WARNING: SAS Program node "Import Data (SHN.csv)" has been created to replace Import Task node.

WARNING: Data node "SHN.csv" has been replaced by a SAS Program node containing SAS Studio generated code!

Display 16 – Note Node Warning and Contents in SAS Studio Conversion Report

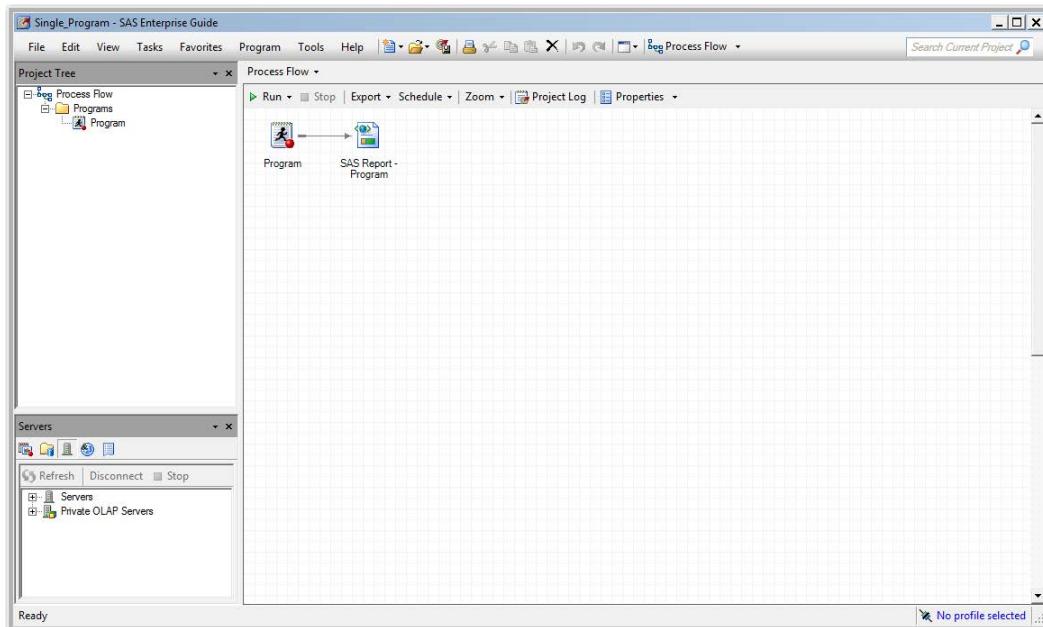
On the **FLOW** tab, you see that there is no Note node.



Display 17 - Note Node Is Not in SAS Studio Process Flow

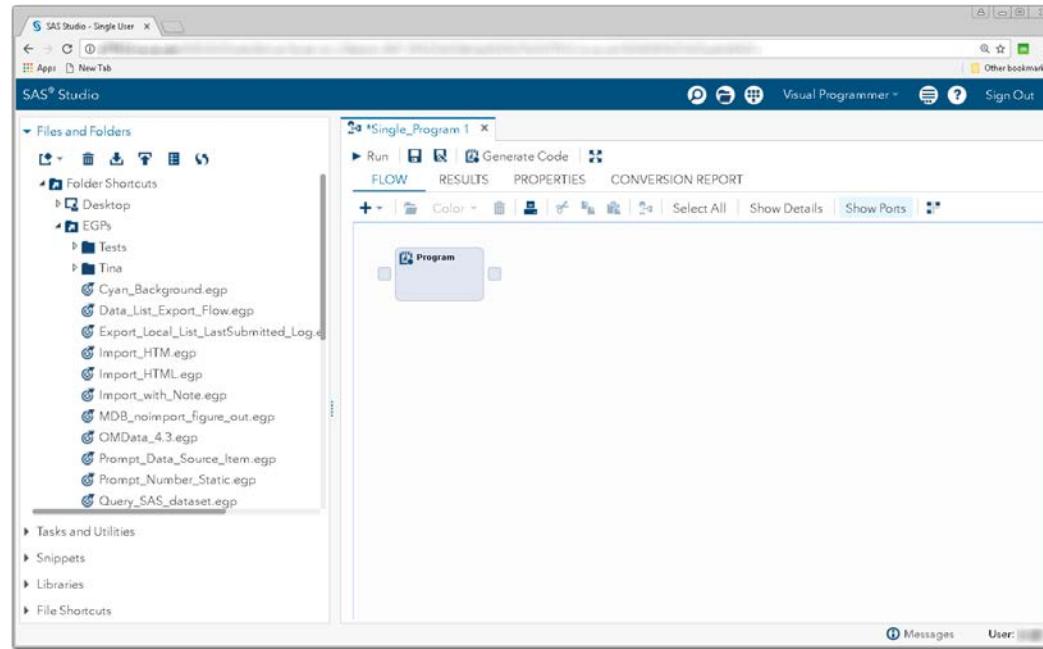
PROGRAM NODES

Program nodes are converted directly to SAS Program nodes in SAS Studio. If the contents of the Program node are stored in a file, the file must be available in order for the contents of the file to be read into converted Program node. An error is written to the Conversion Report if the file containing the program cannot be found. SAS Report nodes generated by Program nodes are not supported in SAS Studio process flows and will not appear in the converted process flow.



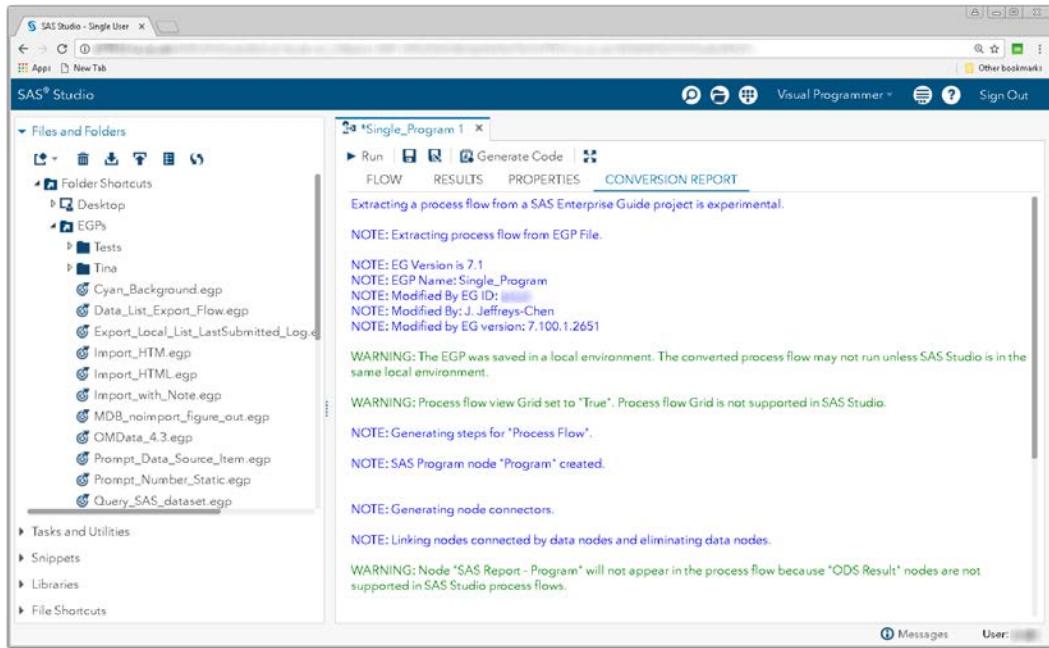
Display 18 – Program Node and SAS Report Node in SAS Enterprise Guide

Here is the converted process flow in SAS Studio. Note that the SAS Report node is not present.



Display 19 - SAS Studio Process Flow with Program Node But No SAS Report Node

There is a warning in the Conversion Report about the removal of the SAS Report node.



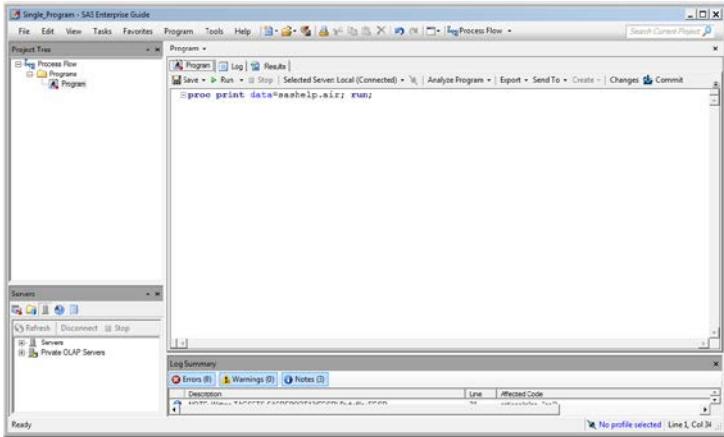
Display 20 - Warning about Removal of SAS Report Node

In SAS Enterprise Guide, Program nodes are associated with a particular SAS connection. If the SAS Studio connection does not match the connection associated with the Program node, an error appears in the log.

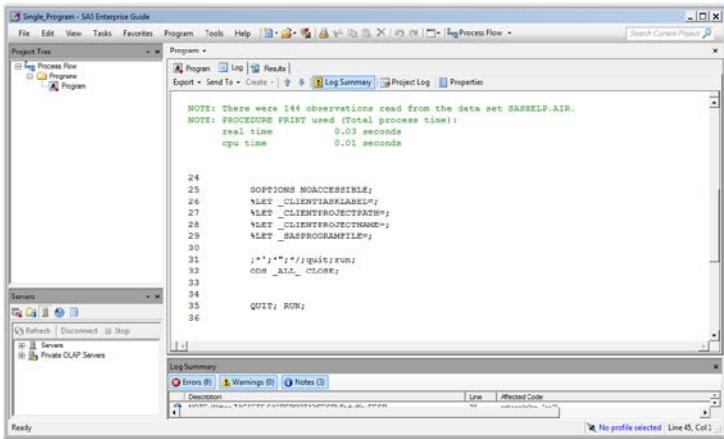
The contents of the Program node in SAS Enterprise Guide and the converted Program node in SAS Studio are very similar, as shown in the following displays.

Program Node in SAS Enterprise Guide

Contrast with [SAS Program Node in SAS Studio](#).



Display 21 – Contents of Program Node in SAS Enterprise Guide



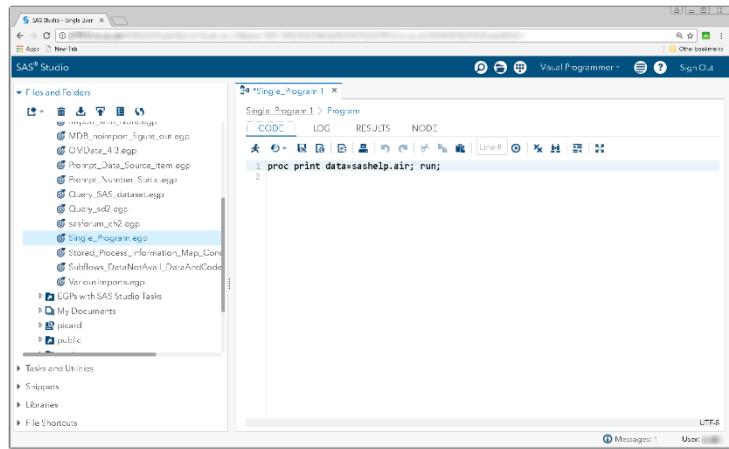
Display 22 – Log Generated by Running the Program Node in SAS Enterprise Guide

Obs	DATE	AIR
1	JAN49	118
2	FEB49	116
3	MAR49	132
4	APR49	129
5	MAY49	135
6	JUN49	135
7	JUL49	148
8	AUG49	148
9	SEP49	148
10	OCT49	119
11	NOV49	104
12	DEC49	118
13	JAN50	120
14	FEB50	126
15	MAR50	141
16	APR50	135
17	MAY50	125
18	JUN50	149
19	JUL50	170
20	AUG50	170
21	SEP50	158
22	OCT50	150
23	NOV50	114
24	DEC50	140
25	JAN51	150
26	FEB51	150
27	MAR51	178
28	APR51	163
29	MAY51	172

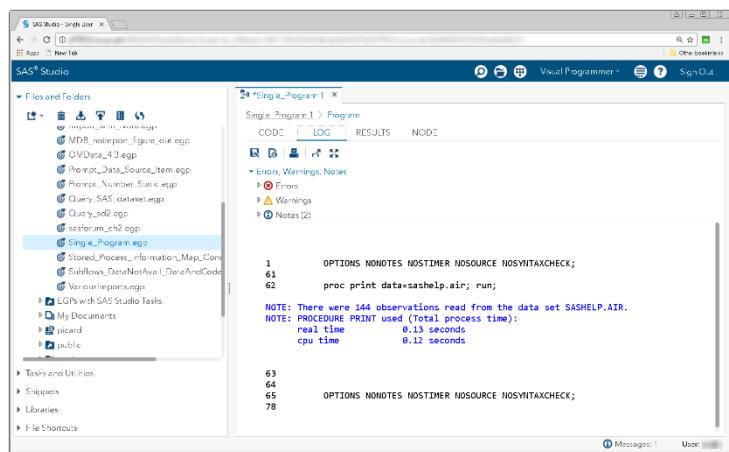
Display 23 – Results Generated by Running the Program Node in SAS Enterprise Guide

SAS Program Node in SAS Studio

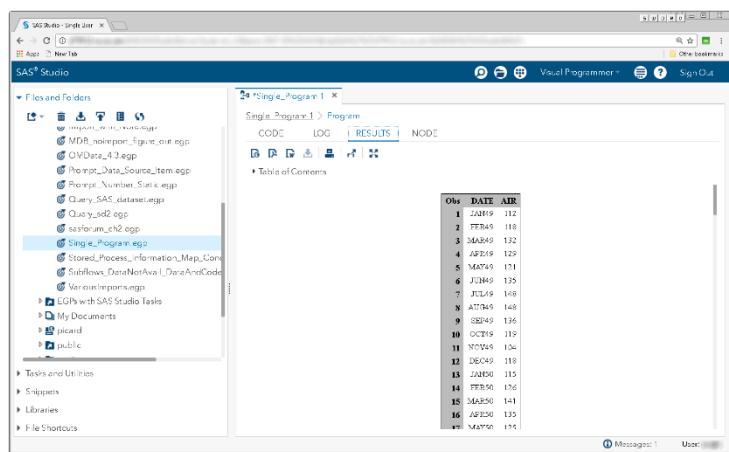
Contrast with [Program Node in SAS Enterprise Guide](#).



Display 24 – Contents of SAS Program Node in SAS Studio



Display 25 – Log Generated by Running the SAS Program Node in SAS Studio



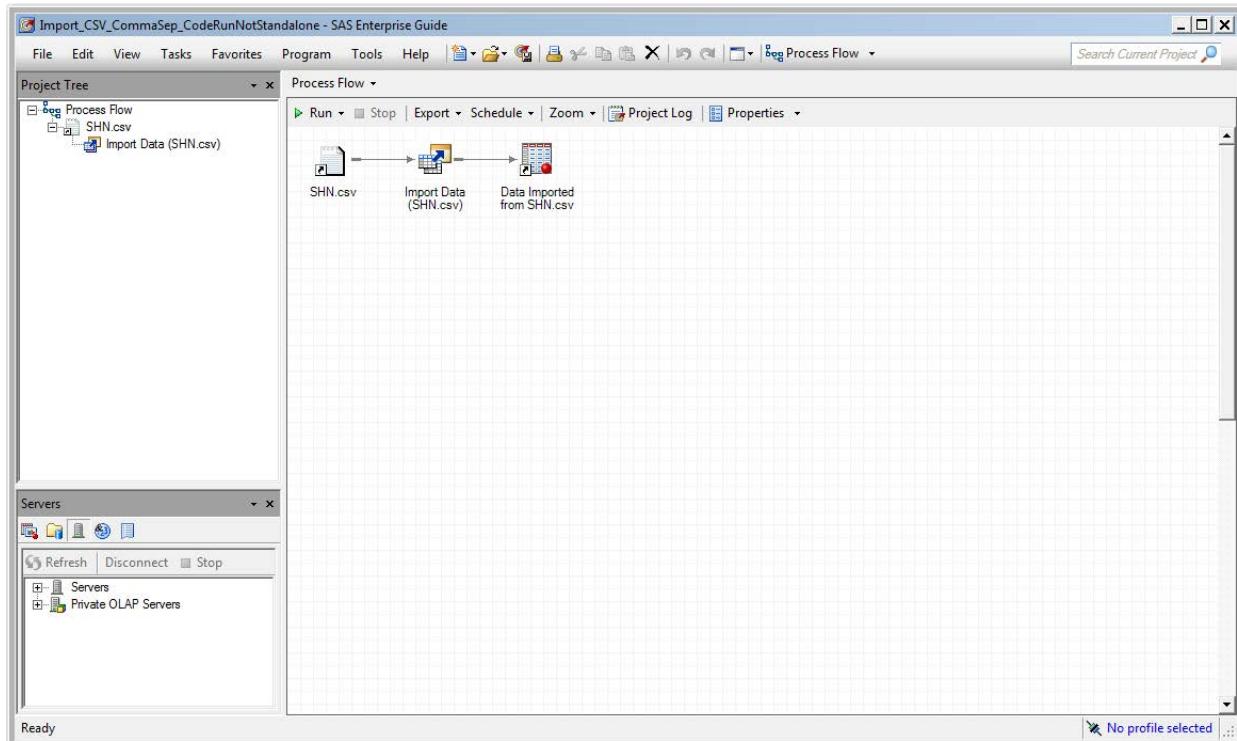
Display 26 – Results Generated by Running the SAS Program Node in SAS Studio

IMPORT DATA NODES

Because an Import Data node in SAS Enterprise Guide does not port directly to an Import Data task in SAS Studio, SAS Studio makes quite a few changes to Import Data nodes to support importing files.

SAS Enterprise Guide

In the following example, a .csv file is dropped into a SAS Enterprise Guide project, and the [Import Data Wizard](#) is used to specify the output data location, delimiter information, fields to import, and other advanced options. The result is three nodes in the SAS Enterprise Guide process flow.

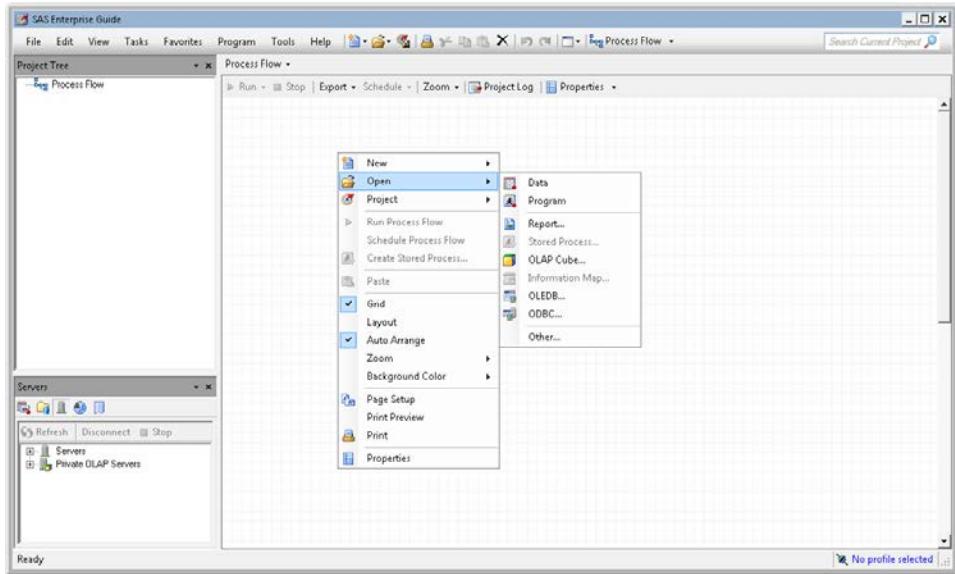


Display 27 - Simple Import Data in SAS Enterprise Guide

The following section delves into the Import Data wizard in SAS Enterprise Guide, the nodes that are generated, and how these nodes are converted when the EGP is opened in SAS Studio.

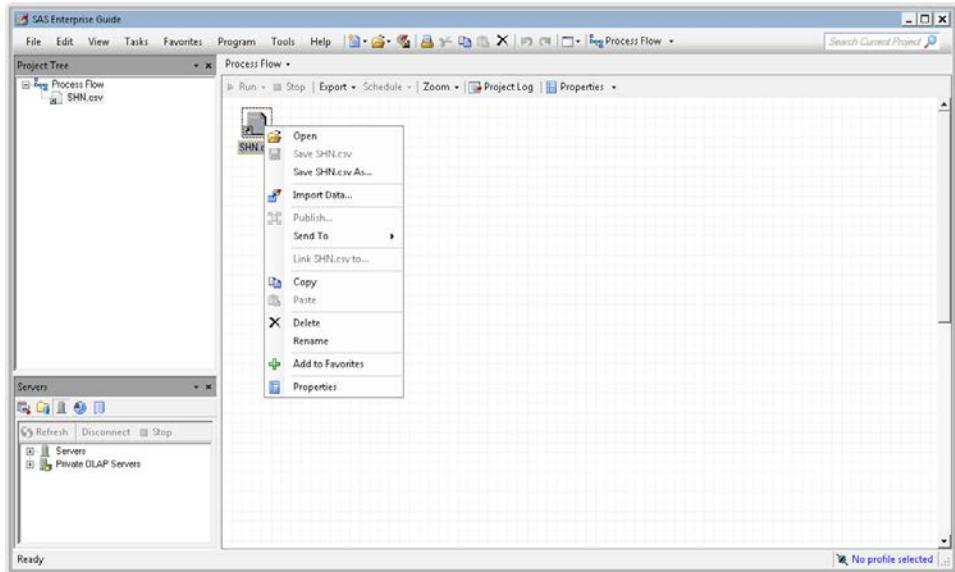
Import Data Wizard

In SAS Enterprise Guide, select **Open -> Data** to create a Data node.



Display 28 – Using Open->Data to Create a Data Node

If the Data node is for an external file, you can choose to import the data in that file to a SAS data set. Selecting **Import Data** on the Data node opens the Import Data wizard.



Display 29 - Import Data Option in Pop-Up Menu

The Specify Data panel of the Import Data wizard:

- displays the location of the source data file
- allows you to specify the encoding to use when importing the file
- allows you to specify the output SAS data set where you want to save the imported data

Click **Performance** to control how the wizard processes the data in the next two steps of the wizard.

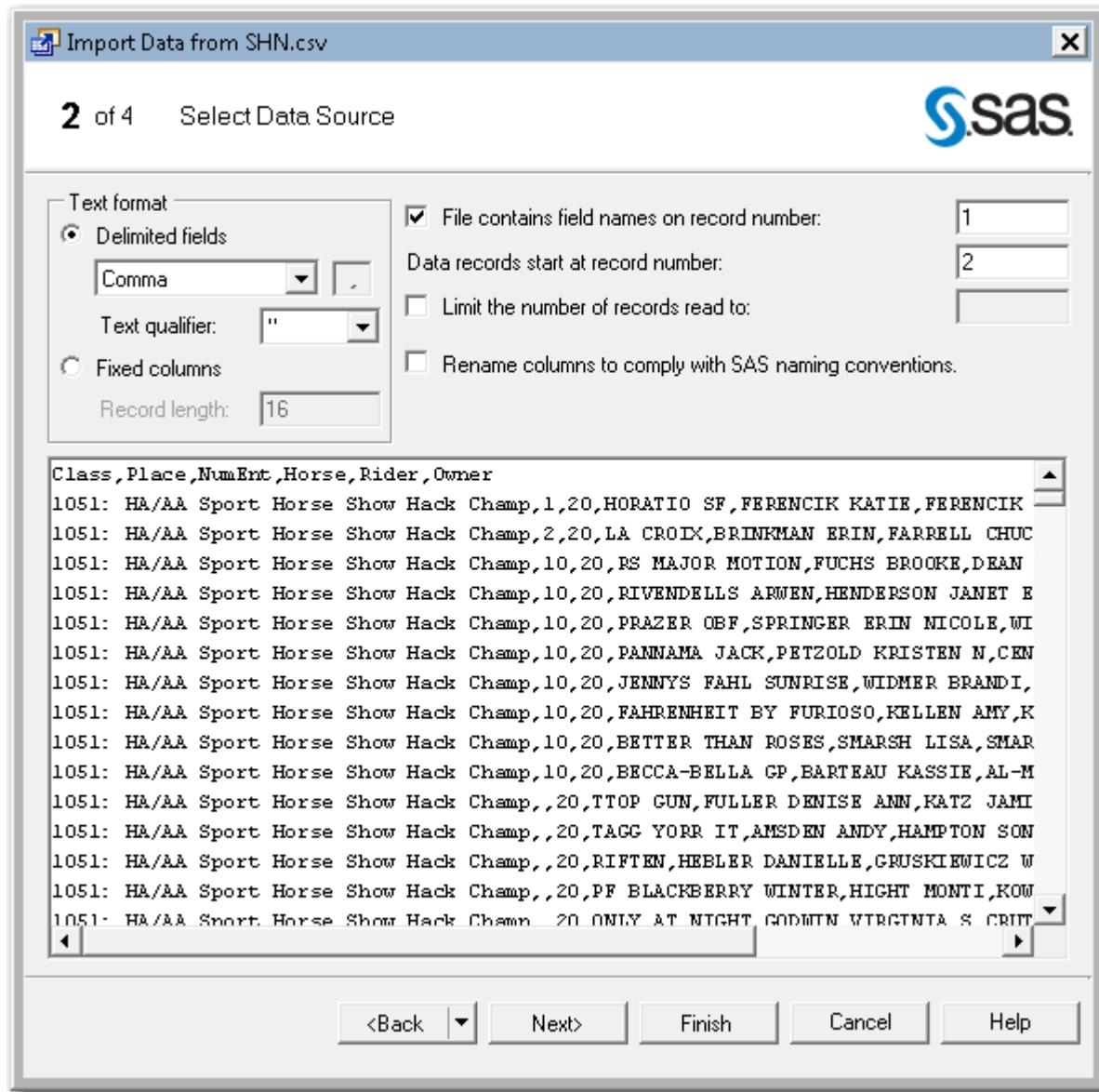
SAS Enterprise Guide creates a Data node for the output SAS data set. This node is converted to a [SAS Program node that displays a view of the SAS data set](#). The [encoding](#) is not yet honored by code generated by SAS Studio.



Display 30 - Specify Data Panel of Import Data Wizard

The Select Data Source panel shows data from the file and allows you to specify options, such as the text format, the record number that contains the field names, the record number where the data starts, and so on.

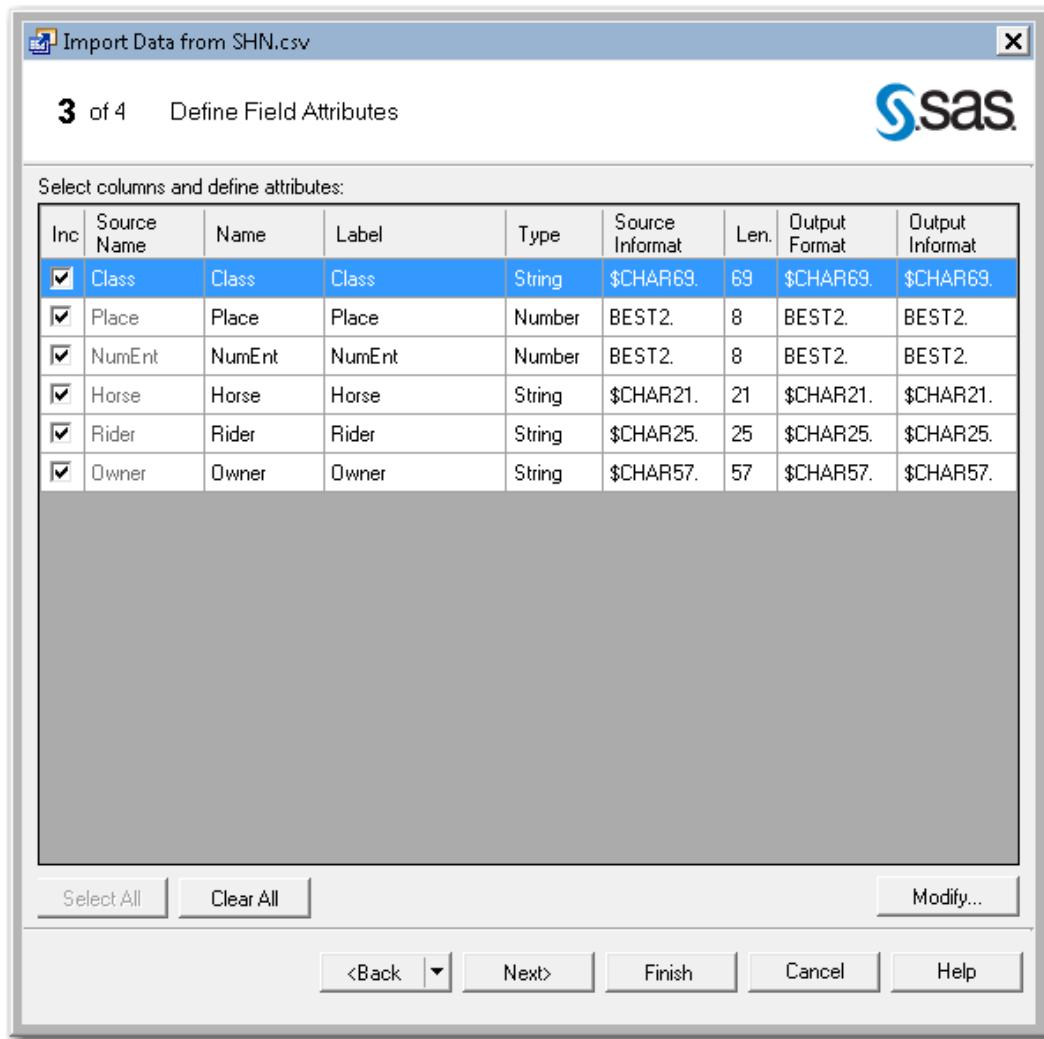
The data collected from this panel is used by SAS Studio when it generates import code.



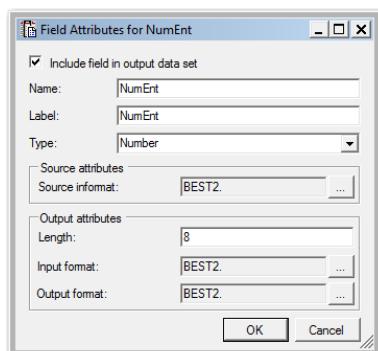
Display 31 - Select Data Source Panel in Import Data Wizard

On the Define Field Attributes panel, you can select which fields you would like to import and specify the formats to use for input and output.

The data collected from this panel is used by SAS Studio when it generates the import code.



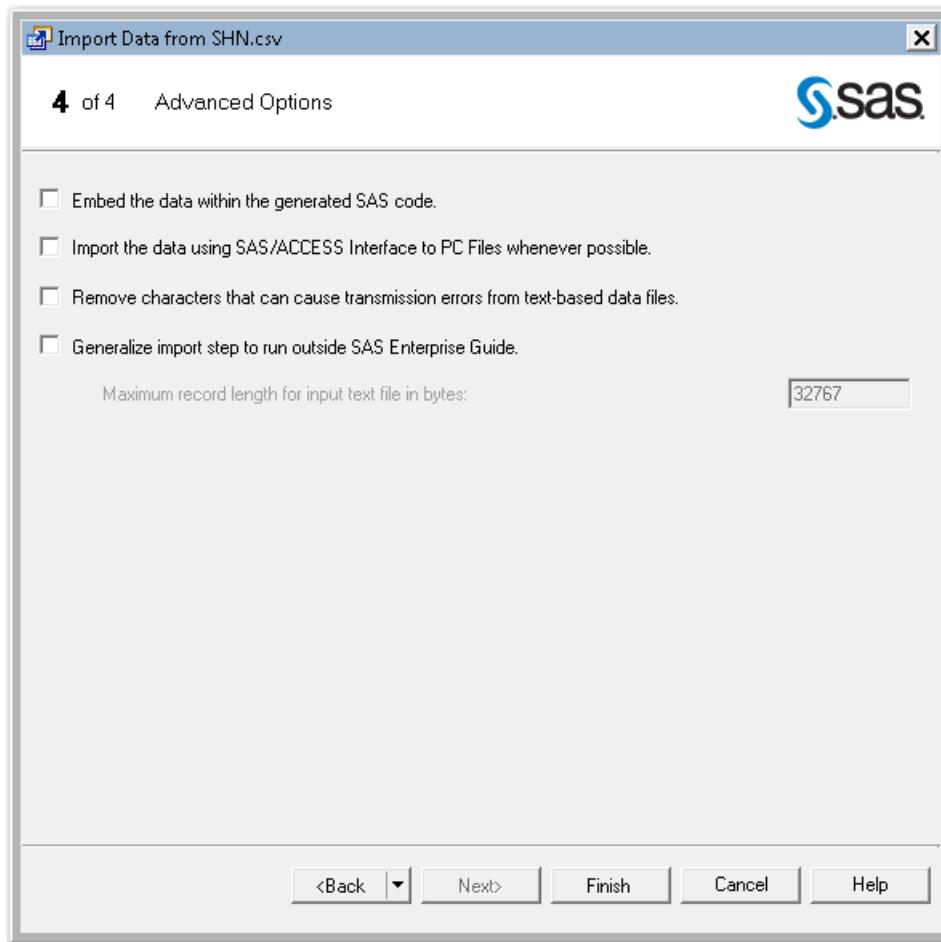
Display 31 - Define Field Attributes Panel in Import Data Wizard



Display 32 - Field Attributes Dialog Box for Modifying Field Input and Output Attributes

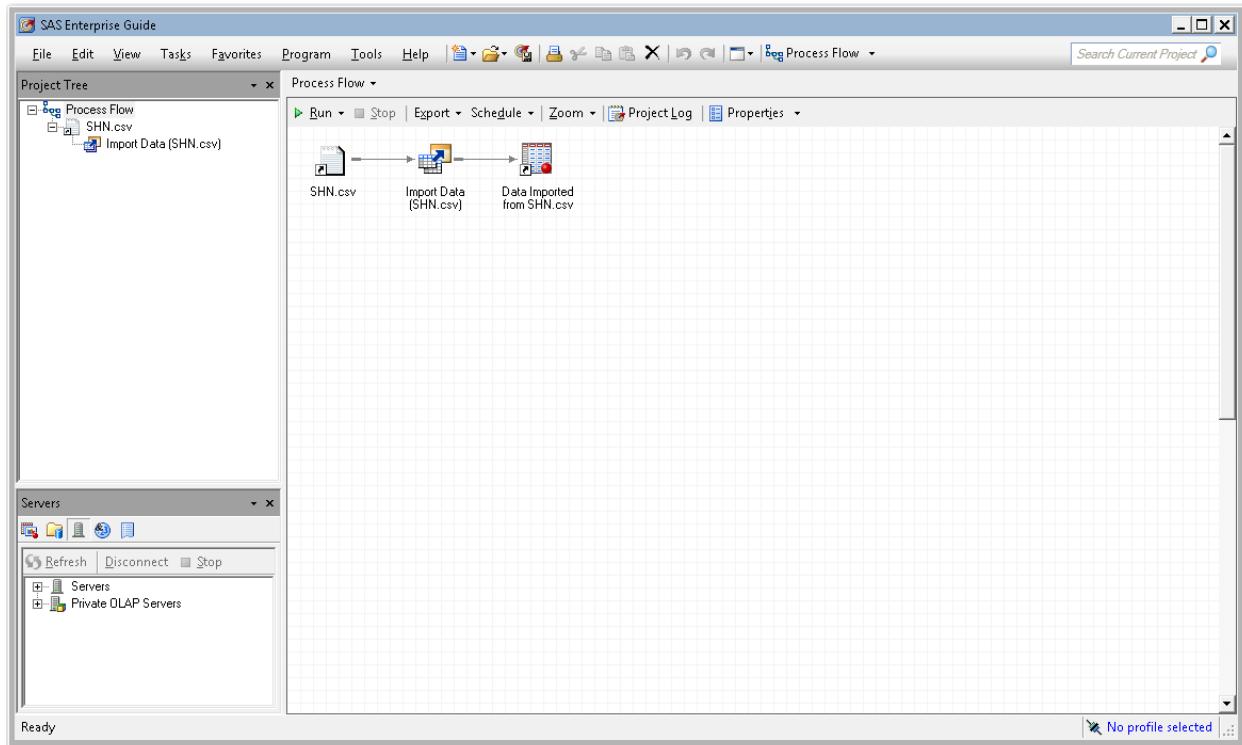
The Advanced Options panel of the Import Data wizard contains these options:

- **Embed the data within the generated SAS code**
If this option is checked, the Data node for the input data remains in the EGP and is shown in the converted SAS Studio process flow. However, any changes to the input data are not reflected in the flow in SAS Studio.
- **Import the data using SAS/ACCESS Interface to PC Files whenever possible**
SAS Studio does not support this functionality.
- **Remove characters that can cause transmission errors from text-based data files**
SAS Studio support for this functionality is unconfirmed.
- **Generalize import step to run outside SAS Enterprise Guide**
If this option is checked, your converted Import Data node in the SAS Studio process flow will contain the code that was generated by SAS Enterprise Guide for the Import Data node. If this option is not checked, SAS Studio generates the code for the converted Import Code node.
- **Maximum record length for input text file in bytes**
SAS Studio does not support this functionality.



Display 33 - Advanced Options Panel in the Import Data Wizard

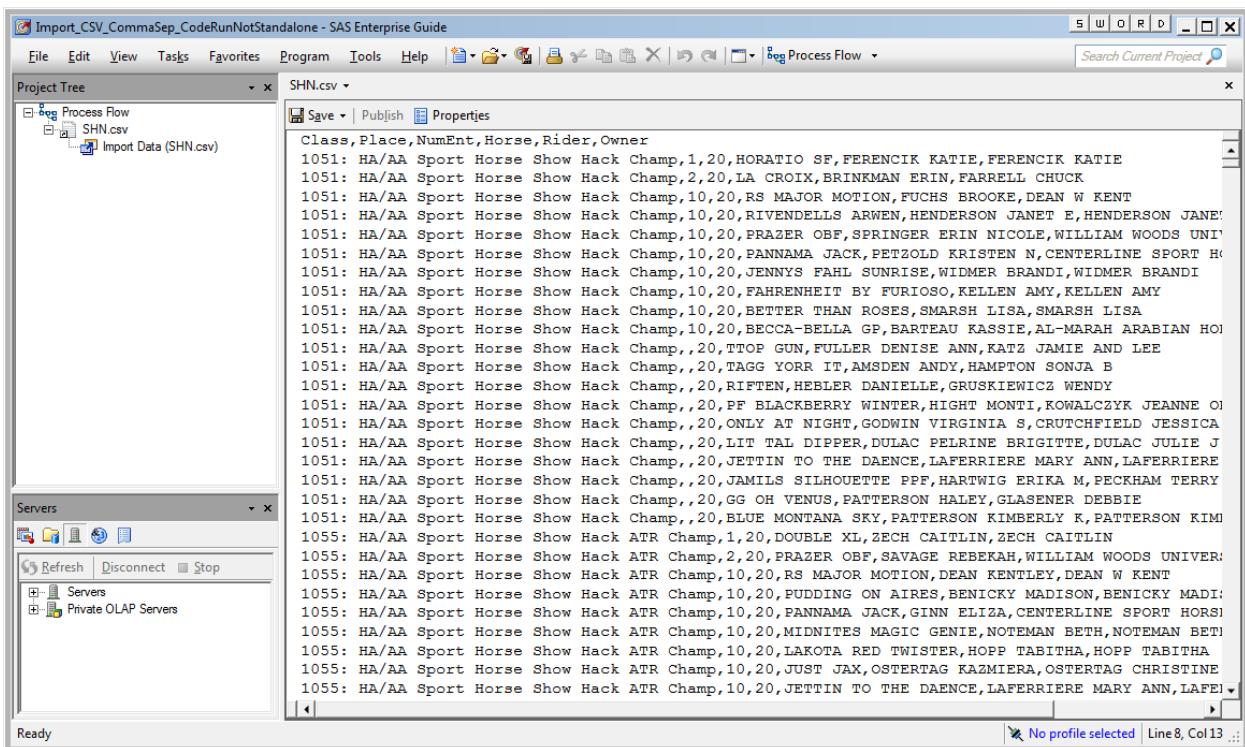
The resulting flow in SAS Enterprise Guide will have a node for the input data, the generated Import Data node, and a node for the the generated output data.



Display 32 - Resulting Process Flow in SAS Enterprise Guide

Data File Node in SAS Enterprise Guide

If you open the node for the input data in the process flow, you see the contents of the file to be imported. You can edit the file contents and save your changes. When the EGP is imported to SAS Studio, this type of node becomes a [SAS Program node that samples the file contents](#). You can see an example in [Converted Import Data Input File Node](#).



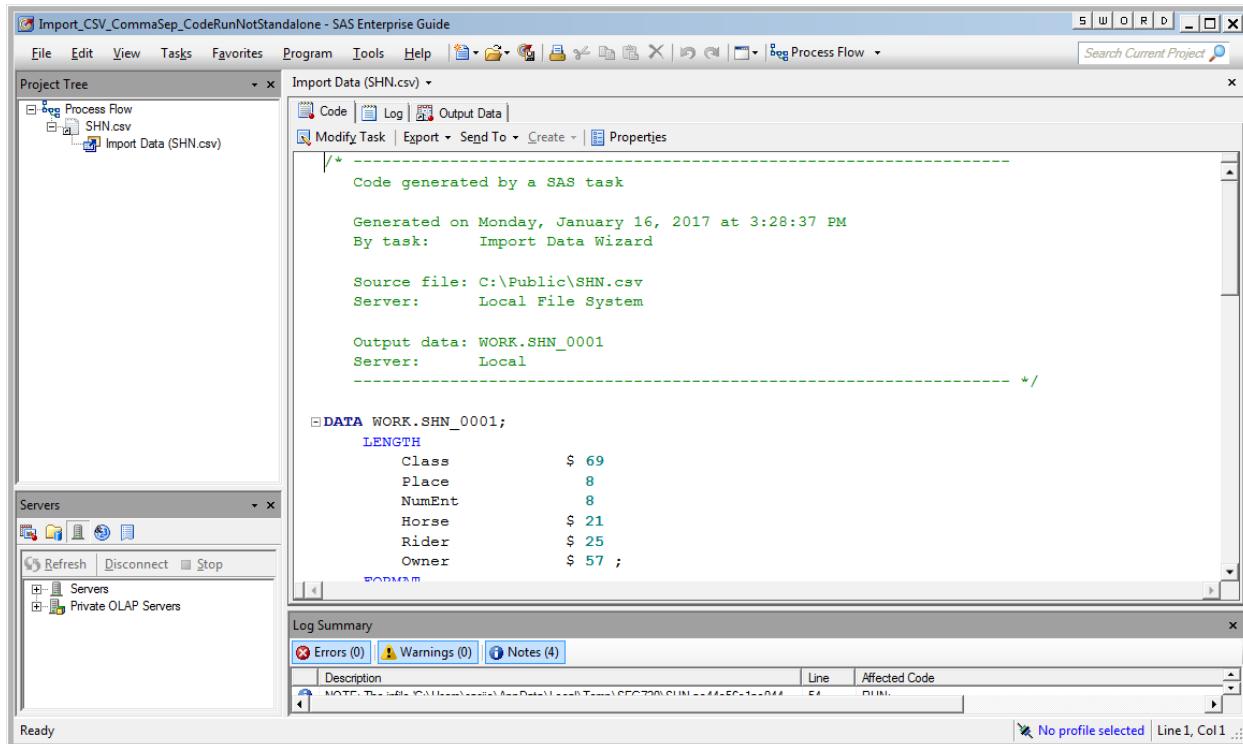
The screenshot shows the SAS Enterprise Guide interface with a project titled "Import_CSV_CommaSep_CodeRunNotStandalone - SAS Enterprise Guide". The "Project Tree" pane shows a "Process Flow" node containing "SHN.csv" and "Import Data (SHN.csv)". The main workspace displays the contents of "SHN.csv" in a large text area. The "Servers" pane shows a list of servers, including "Private OLAP Servers". The bottom status bar indicates "Ready" and "No profile selected | Line 8, Col 13 ::".

```
Class,Place,NumEnt,Horse,Rider,Owner
1051: HA/AA Sport Horse Show Hack Champ,,1,20,HORATIO SF,FERENCIK KATIE,FERENCIK KATIE
1051: HA/AA Sport Horse Show Hack Champ,,2,20,LA CROIX,BRINKMAN ERIN,FARRELL CHUCK
1051: HA/AA Sport Horse Show Hack Champ,,10,20,RS MAJOR MOTION,FUCHS BROOKE,DEAN W KENT
1051: HA/AA Sport Horse Show Hack Champ,,10,20,RIVENDELLS ARWEN,HENDERSON JANET E,HENDERSON JANE
1051: HA/AA Sport Horse Show Hack Champ,,10,20,PRAZER OBF,SPRINGER ERIN NICOLE,WILLIAM WOODS UNIVE
1051: HA/AA Sport Horse Show Hack Champ,,10,20,PANNAMA JACK,PETZOLD KRISTEN N,CENTERLINE SPORT HORSES
1051: HA/AA Sport Horse Show Hack Champ,,10,20,JENNYS FAHL SUNRISE,WIDMER BRANDI,WIDMER BRANDI
1051: HA/AA Sport Horse Show Hack Champ,,10,20,FAHRENHEIT BY FURIOSO,KELLEN AMY,KELLEN AMY
1051: HA/AA Sport Horse Show Hack Champ,,10,20,BETTER THAN ROSES,SMARSH LISA,SMARSH LISA
1051: HA/AA Sport Horse Show Hack Champ,,10,20,BECCA-BELLA GP,BARTEAU KASSIE,AL-MARAH ARABIAN HORSES
1051: HA/AA Sport Horse Show Hack Champ,,20,TTOP GUN,FULLER DENISE ANN,KATE JAMIE AND LEE
1051: HA/AA Sport Horse Show Hack Champ,,20,TAGG YORK IT,AMSDEN ANDY,HAMPTON SONJA B
1051: HA/AA Sport Horse Show Hack Champ,,20,RIFTEN,HEBLER DANIELLE,GRUSKIEWICZ WENDY
1051: HA/AA Sport Horse Show Hack Champ,,20,PF BLACKBERRY WINTER,HIGHT MONTI,KOWALCZYK JEANNE O
1051: HA/AA Sport Horse Show Hack Champ,,20,ONLY AT NIGHT,GODWIN VIRGINIA S,CRUTCHFIELD JESSICA
1051: HA/AA Sport Horse Show Hack Champ,,20,LIT TAL DIPPER,DULAC PELRINE BRIGITTE,DULAC JULIE J
1051: HA/AA Sport Horse Show Hack Champ,,20,JETTIN TO THE DAENCE,LAFERRIERE MARY ANN,LAFERRIERE
1051: HA/AA Sport Horse Show Hack Champ,,20,JAMILS SILHOUETTE PPF,HARTWIG ERIKA M,PECKHAM TERRY
1051: HA/AA Sport Horse Show Hack Champ,,20,GG ON VENUS,PATTERSON HALEY,GLASENER DEBBIE
1051: HA/AA Sport Horse Show Hack Champ,,20,BLUE MONTANA SKY,PATTERSON KIMBERLY K,PATTERSON KIM
1055: HA/AA Sport Horse Show Hack ATR Champ,,1,20,DOUBLE XL,ZECH CAITLIN,ZECH CAITLIN
1055: HA/AA Sport Horse Show Hack ATR Champ,,2,20,PRAZER OBF,SAVAGE REBEKAH,WILLIAM WOODS UNIVERSIT
1055: HA/AA Sport Horse Show Hack ATR Champ,,10,20,RS MAJOR MOTION,DEAN KENTLEY,DEAN W KENT
1055: HA/AA Sport Horse Show Hack ATR Champ,,10,20,PUDGING ON AIRES,BENICKY MADISON,BENICKY MADISON
1055: HA/AA Sport Horse Show Hack ATR Champ,,10,20,PANNAMA JACK,GINN ELIZA,CENTERLINE SPORT HORSES
1055: HA/AA Sport Horse Show Hack ATR Champ,,10,20,MIDNITES MAGIC GENIE,NOTE MAN BETH,NOTE MAN BETH
1055: HA/AA Sport Horse Show Hack ATR Champ,,10,20,LAKOTA RED TWISTER,HOPP TABITHA,HOPP TABITHA
1055: HA/AA Sport Horse Show Hack ATR Champ,,10,20,JUST JAX,OSTERTAG KAZMIERA,OSTERTAG CHRISTINE
1055: HA/AA Sport Horse Show Hack ATR Champ,,10,20,JETTIN TO THE DAENCE,LAFERRIERE MARY ANN,LAFERRIERE
```

Display 33 - Contents of Input File for Import Task in SAS Enterprise Guide

Import Data Node in SAS Enterprise Guide

If you open the Import Data node, you see the code generated by SAS Enterprise Guide after running the Specify Data wizard. If you want to change the import options, you cannot edit this code directly. Instead, you must right-click the node and select **Modify Import Data** from the pop-up menu. When the EGP is opened in SAS Studio, the Import Data node is converted to a SAS Program node. You can see an example of a converted SAS Program node in [Converted SAS Program Node for Import Data Node](#).



The screenshot shows the SAS Enterprise Guide interface with the title bar "Import_CSV_CommaSep_CodeRunNotStandalone - SAS Enterprise Guide". The main window displays the "Import Data (SHN.csv)" node under the "Process Flow" project. The "Code" tab is selected, showing the generated SAS code:

```
/* -----  
   Code generated by a SAS task  
  
   Generated on Monday, January 16, 2017 at 3:28:37 PM  
   By task:      Import Data Wizard  
  
   Source file: C:\Public\SHN.csv  
   Server:       Local File System  
  
   Output data: WORK.SHN_0001  
   Server:       Local  
----- */  
  
DATA WORK.SHN_0001;  
  LENGTH  
    Class      $ 69  
    Place      8  
    NumEnt    8  
    Horse      $ 21  
    Rider      $ 25  
    Owner      $ 57 ;
```

The "Log Summary" panel at the bottom shows 4 notes. The "Servers" panel on the left lists "Servers" and "Private OLAP Servers".

Display 34 - Import Data Code in SAS Enterprise Guide

SAS Data Set Node in SAS Enterprise Guide

If you open the output data node from the Import Data task, you see a SAS data set that contains the imported data. When the EGP file is opened in SAS Studio, the Data node for the output data set is converted to a [SAS Program node that samples](#) the output data set. An example of this can be seen in [Converted Output Data Set Node for Import Data Node](#).

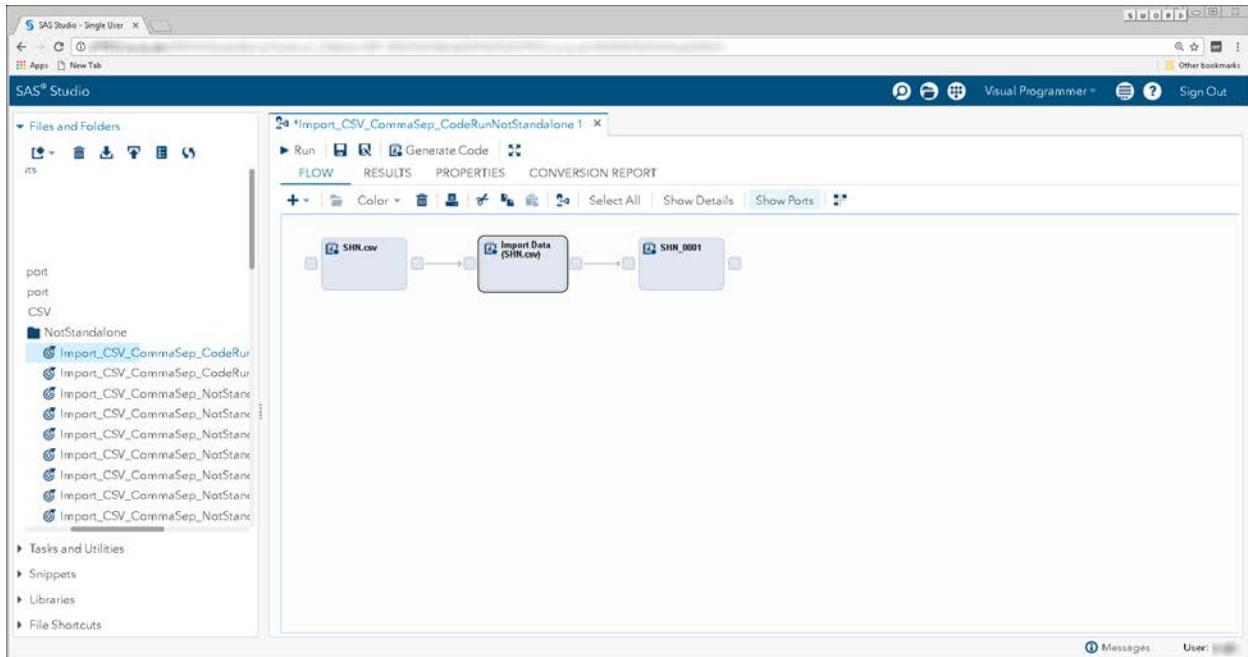
The screenshot shows the SAS Enterprise Guide interface with the title bar "Import_CSV_CommaSep_CodeRunNotStandalone - SAS Enterprise Guide". The menu bar includes File, Edit, View, Tasks, Favorites, Program, Tools, Help, and a Process Flow dropdown. The Project Tree on the left shows a "Process Flow" node containing "SHN.csv" and "Import Data (SHN.csv)". The main workspace displays a data grid titled "Import Data (SHN.csv)". The grid has columns: Class, Place, NumEnt, Horse, and Rider. The data consists of 22 rows, each representing a horse entry with its class, place, number of entries, horse name, and rider name. The first few rows are:

	Class	Place	NumEnt	Horse	Rider
1	1051: HA/AA Sport Horse Show Hack Champ	1	20	HORATIO SF	FERENCIK KATIE
2	1051: HA/AA Sport Horse Show Hack Champ	2	20	LA CROIX	BRINKMAN ERIN
3	1051: HA/AA Sport Horse Show Hack Champ	10	20	RS MAJOR MOTION	FUCHS BROOKE
4	1051: HA/AA Sport Horse Show Hack Champ	10	20	RIVENDELLS ARWEN	HENDERSON JANET E
5	1051: HA/AA Sport Horse Show Hack Champ	10	20	PRAZER OBF	SPRINGER ERIN NICOLE
6	1051: HA/AA Sport Horse Show Hack Champ	10	20	PANNAMA JACK	PETZOLD KRISTEN N
7	1051: HA/AA Sport Horse Show Hack Champ	10	20	JENNYS FAHL SUNRISE	WIDMER BRANDI
8	1051: HA/AA Sport Horse Show Hack Champ	10	20	FAHRENHEIT BY FURIOSO	KELLEN AMY
9	1051: HA/AA Sport Horse Show Hack Champ	10	20	BETTER THAN ROSES	SMARSH LISA
10	1051: HA/AA Sport Horse Show Hack Champ	10	20	BECCA-BELLA GP	BARTEAU KASSIE
11	1051: HA/AA Sport Horse Show Hack Champ	-	20	TTOP GUN	FULLER DENISE ANN
12	1051: HA/AA Sport Horse Show Hack Champ	-	20	TAGG YORR IT	AMSDEN ANDY
13	1051: HA/AA Sport Horse Show Hack Champ	-	20	RIFTEN	HEBLER DANIELLE
14	1051: HA/AA Sport Horse Show Hack Champ	-	20	PF BLACKBERRY WINTER	KOWALCZYK
15	1051: HA/AA Sport Horse Show Hack Champ	-	20	ONLY AT NIGHT	CRUTCHFIELD
16	1051: HA/AA Sport Horse Show Hack Champ	-	20	LIT TAL DIPPER	DULAC JULIE
17	1051: HA/AA Sport Horse Show Hack Champ	-	20	JETTIN TO THE DAENCE	LAFFERRIERE MARY ANN
18	1051: HA/AA Sport Horse Show Hack Champ	-	20	JAMILS SILHOUETTE PPF	PECKHAM TE
19	1051: HA/AA Sport Horse Show Hack Champ	-	20	GG OH VENUS	PATTERSON HALEY
20	1051: HA/AA Sport Horse Show Hack Champ	-	20	BLUE MONTANA SKY	PATTERSON KIMBERLY K
21	1055: HA/AA Sport Horse Show Hack ATR Ch...	1	20	DOUBLE XL	ZECH CAITLIN
22	1055: HA/AA Sport Horse Show Hack ATR Ch...	2	20	PRAZER OBF	WILLIAM WOC

Display 35 - Output Data Set in SAS Enterprise Guide

SAS Studio

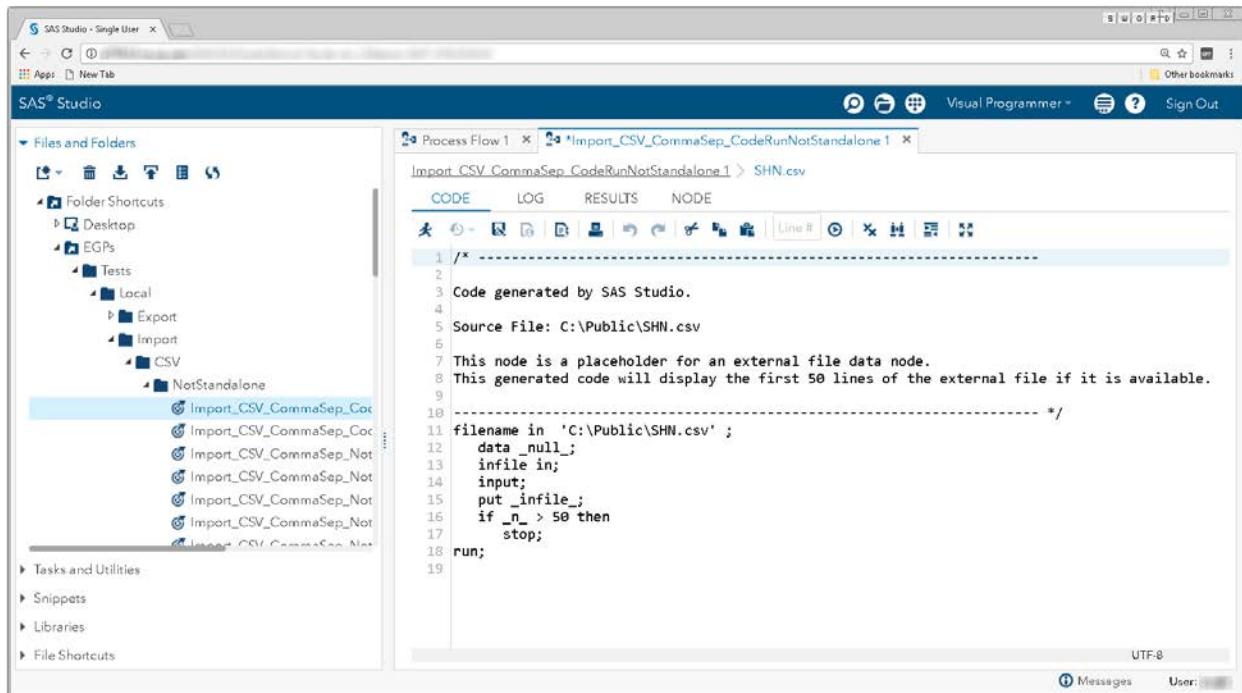
Opening the EGP in SAS Studio results in representative nodes for each node in the Import Data process flow, but these nodes are remarkably different in SAS Studio.



Display 36 - Converted Process Flow for Import Data Task in SAS Studio

Converted Import Data Input File Node

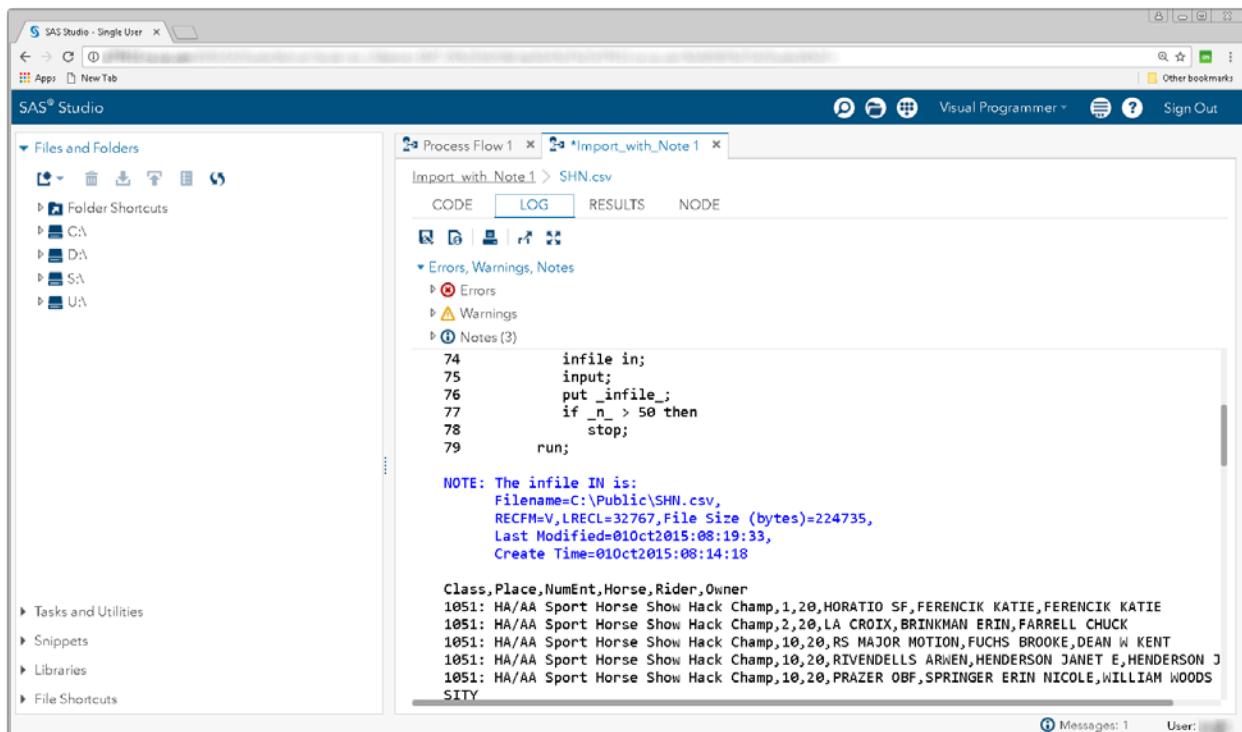
The input Data node for the Import Data node is converted to a SAS Program node in SAS Studio. The SAS Program node contains SAS Studio generated code that samples the input file contents. See the [Data Sampling](#) section of this document for more information about file sampling Data nodes. You can contrast this to the [Data File Node in SAS Enterprise Guide](#).



A screenshot of the SAS Studio interface. The left sidebar shows 'Files and Folders' with a tree view of 'Folder Shortcuts' containing 'Desktop', 'EGPs', 'Tests', 'Local', 'Export', 'Import', 'CSV', and 'NotStandalone'. Under 'Import', there are several sub-folders starting with 'Import_CSV_'. The main workspace shows a 'Process Flow 1' tab with a sub-tab 'Import_CSV_CommaSep_CodeRunNotStandalone 1 > SHN.csv'. The 'CODE' tab is selected, displaying the following SAS code:

```
1 /* -----
2
3 Code generated by SAS Studio.
4
5 Source File: C:\Public\SHN.csv
6
7 This node is a placeholder for an external file data node.
8 This generated code will display the first 50 lines of the external file if it is available.
9
10 -----
11 filename in 'C:\Public\SHN.csv';
12 data _null_;
13 infile in;
14 input;
15 put _infile_;
16 if _n_ > 50 then
17   stop;
18 run;
```

Display 37 - Converted Input Data File Node in SAS Studio



A screenshot of the SAS Studio interface. The left sidebar shows 'Files and Folders' with a tree view of 'Folder Shortcuts' containing 'CA', 'DA', 'SA', and 'UA'. The main workspace shows a 'Process Flow 1' tab with a sub-tab 'Import_with_Note 1 > SHN.csv'. The 'LOG' tab is selected, displaying the following log output:

Errors, Warnings, Notes

- Errors
- Warnings
- Notes (3)

```
74     infile in;
75     input;
76     put _infile_;
77     if _n_ > 50 then
78       stop;
79     run;
```

NOTE: The infile IN is:
Filename=C:\Public\SHN.csv,
RECFM=V,LRECL=32767,File Size (bytes)=224735,
Last Modified=01Oct2015:08:19:33,
Create Time=01Oct2015:08:14:18

```
Class,Place,NumEnt,Horse,Rider,Owner
1051: HA/AA Sport Horse Show Hack Champ,1,20,HORATIO SF,FERENCIK KATIE,FERENCIK KATIE
1051: HA/AA Sport Horse Show Hack Champ,2,20,LA CROIX,BRINKMAN ERIN,FARRELL CHUCK
1051: HA/AA Sport Horse Show Hack Champ,10,20,RS MAJOR MOTION,FUCHS BROOKE,DEAN W KENT
1051: HA/AA Sport Horse Show Hack Champ,10,20,RIVENDELLS ARVEN,HENDERSON JANET E,HENDERSON J
1051: HA/AA Sport Horse Show Hack Champ,10,20,PRAZER OBF,SPRINGER ERIN NICOLE,WILLIAM WOODS
SITY
```

Display 38 – Log for Converted Data Node

Converted Output Data Set Node for Import Data Node

Similarly, the node for the output data has been converted to a SAS Program node. When you run the converted process flow, the **OUTPUT DATA** tab for the converted node contains a view of the imported SAS data set.

The screenshot shows the SAS Studio interface. On the left, the 'Files and Folders' sidebar is open, displaying a tree structure with 'Folder Shortcuts', 'Desktop', 'EGPs', and 'Tests' (which further branches into 'Local', 'Export', 'Import', 'CSV', and 'NotStandalone'). Under 'NotStandalone', several files are listed, including 'Import_CSVDelimited_CodeRunNotStandalone.sas', which is currently selected and highlighted with a blue background. The main workspace contains a 'Process Flow 1' tab and a 'Visual Programmer' tab. The 'Visual Programmer' tab is active, showing a code editor with the following content:

```
1 /*-----  
2  
3 Code generated by SAS Studio.  
4  
5 Data Node Name: SHN_0001  
6 Dataset: WORK.SHN_0001  
7 Server: Local  
8  
9 IMPORTANT NOTE: Update this code to resolve the data source if necessary.  
10-----*/  
11  
12 DATA WORK.SHN_00011090 / VIEW=WORK.SHN_00011090;  
13     SET WORK.SHN_0001;  
14  
15 RUN;  
16
```

The status bar at the bottom right indicates 'UTF-8' encoding.

Display 39 – Contents of SAS Program Node for Output Data

The screenshot shows the SAS Studio interface with the following details:

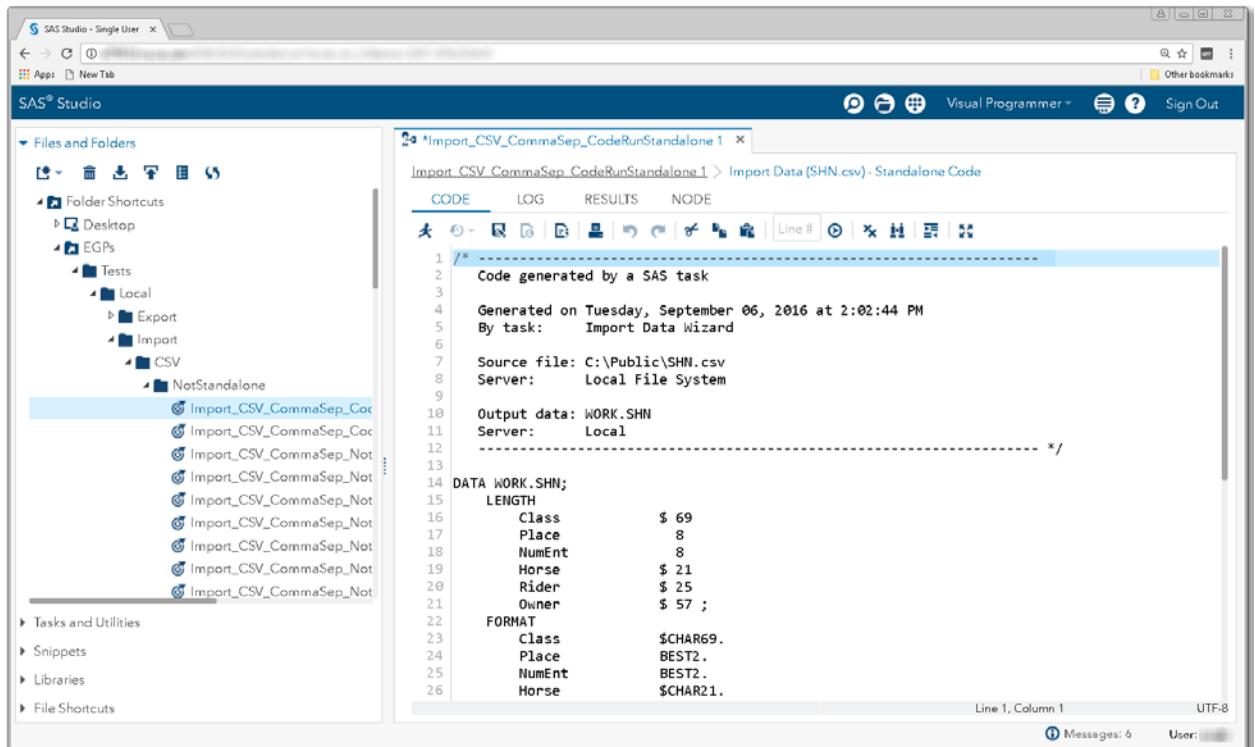
- Top Bar:** SAS Studio - Single User X, Apps, New Tab, Other bookmarks.
- SAS Studio Header:** Visual Programmer+, Sign Out.
- Left Sidebar:** Files and Folders (with icons for Folder Shortcuts, CA, DA, SA, UA), Tasks and Utilities, Snippets, Libraries, File Shortcuts.
- Central Area:** Process Flow 1 > *Import_with_Note_1 > SHN_0002
- Tab Bar:** CODE (selected), LOG, RESULTS, NODE, OUTPUT DATA.
- Table View:** Table: WORK.SHN_00023969, View: Column names, Filter: (none). Total rows: 2139, Total columns: 6.
- Column List:** Select all, Class, Place, NumEnt, Horse, Rider, Owner.
- Property Grid:** Property (Label, Name, Length, Type) and Value.
- Data Preview:** A grid showing 14 rows of data, all entries being "1051: HA/AA Sport Horse Show Hack Champ".

Display 40 – Output Data for Converted Node

Converted SAS Program Node for Import Data Node

An Import Data node in SAS Enterprise Guide becomes a SAS Program node in SAS Studio. You can contrast this to the [Import Data Node in SAS Enterprise Guide](#). Unlike the Import Data node in SAS Enterprise Guide, the code in the converted SAS Program node can be edited. The contents of this node will be either:

- code generated by SAS Enterprise Guide if the **Generalize step to run outside of SAS Enterprise Guide** option was selected in the Specify Data panel in the Import Task wizard. This code should function the same in SAS Studio and SAS Enterprise Guide when using an identical SAS server.



The screenshot shows the SAS Studio interface with the title bar "SAS Studio - Single User". The left sidebar displays "Files and Folders" with a tree view of "Local" and "NotStandalone" folders, including sub-folders like "Export", "Import", and "CSV". The main workspace is titled "Import_CSV_CommaSep_CodeRunStandalone_1 > Import Data (SHN.csv) - Standalone Code". It contains a "CODE" tab showing the generated SAS code:

```
1 /*-----  
2  Code generated by a SAS task  
3-----  
4 Generated on Tuesday, September 06, 2016 at 2:02:44 PM  
5 By task: Import Data Wizard  
6-----  
7 Source file: C:\Public\SHN.csv  
8 Server: Local File System  
9-----  
10 Output data: WORK.SHN  
11 Server: Local  
12-----*/  
13  
14 DATA WORK.SHN;  
15   LENGTH  
16     Class      $ 69  
17     Place      8  
18     NumEnt    8  
19     Horse      $ 21  
20     Rider      $ 25  
21     Owner      $ 57 ;  
22   FORMAT  
23     Class      $CHAR69.  
24     Place      BEST2.  
25     NumEnt    BEST2.  
26     Horse      $CHAR21.
```

The status bar at the bottom indicates "Line 1, Column 1" and "UTF-8".

Display 41 – Code Generated in SAS Enterprise Guide Is the Same in SAS Studio

- code generated by SAS Studio that tries to closely match the functionality provided by Import Data in SAS Enterprise Guide.

```

SAS® Studio
Process Flow 1 > Import_CS...CodeRunNotStandalone 1 > Import Data (SHN.csv)

CODE LOG RESULTS NODE
1 /* -----
2
3 Code generated by SAS Studio.
4
5 Source File: C:\Public\SHN.csv
6
7 Output Data: WORK.SHN_0001
8
9 -----
10 DATA WORK.SHN_0001;
11   LENGTH
12     Class      $ 69
13     Place      8
14     NumEnt    8
15     Horse      $ 21
16     Rider      $ 25
17     Owner      $ 57;
18
19   LABEL
20     Class = "Class"
21     Place = "Place"
22     NumEnt = "NumEnt"
23     Horse = "Horse"

```

Display 42 – Code Generated by SAS Studio for Converted Import Data Node

Unsupported Import Features and Limitations

Encoding

Encoding support is on the list of features to be added.

Import code generated by SAS Studio uses the default file encoding when reading the input data. If the default encoding is not applicable to your input file, add the INFILE statement to the code generated by SAS Studio to manually set the encoding of your input data.

```

INFILE 'C:\Public\SHN.csv'
  LRECL=32767
  FIRSTOBS=2
  ENCODING="WLATIN1"
  DLM='2c'x
  MISSOVER
  DSD ;

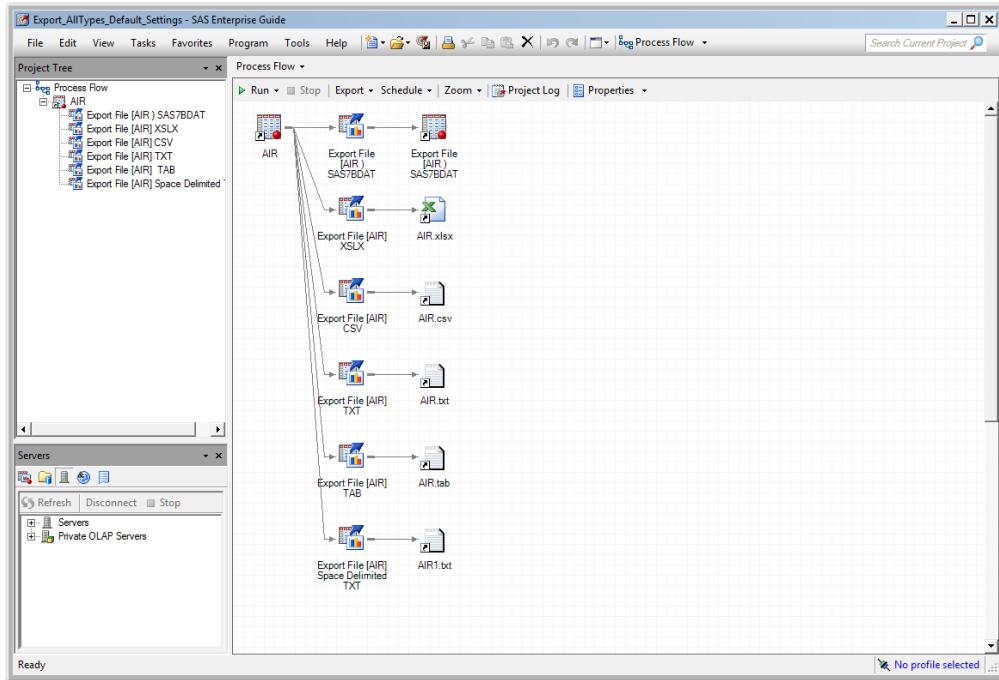
```

Unsupported Import Data

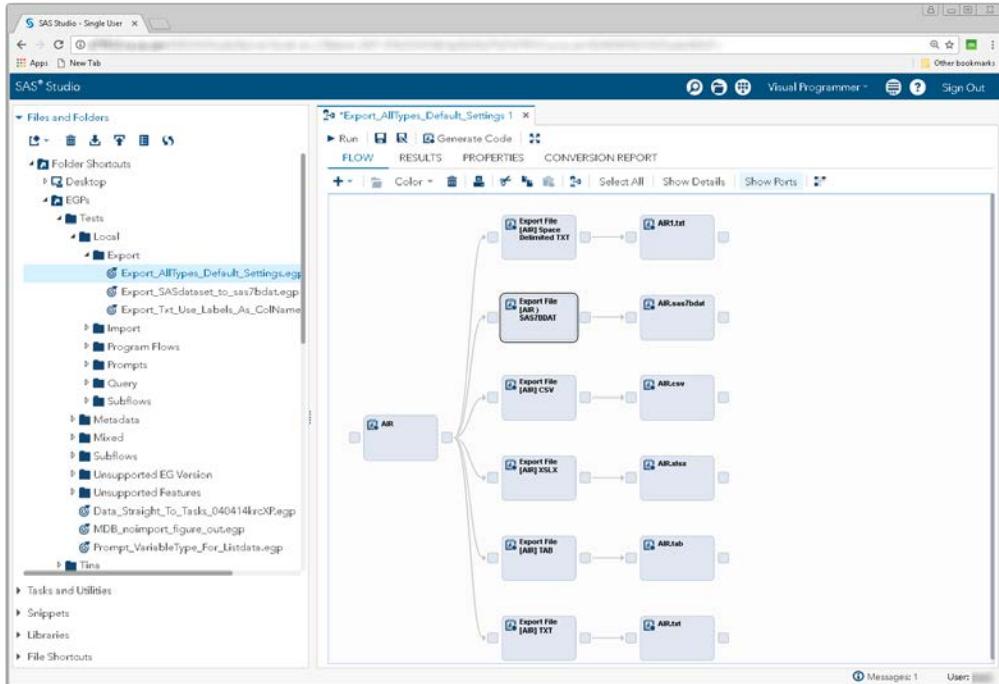
SAS Studio does not support importing data from HTML files. Currently, the code generated for importing HTML content is nonsensical.

EXPORT FILE NODES

Because an Export File node in SAS Enterprise Guide does not port directly to a SAS Studio node, SAS Studio makes quite a few changes to Export File nodes. While the nodes associated with exporting files in SAS Enterprise Guide are all represented in a converted process flow, the nodes are remarkably different. The following two displays show the same Export File process flow in SAS Enterprise Guide and SAS Studio.



Display 43 – Export File Nodes in SAS Enterprise Guide



Display 44 - Converted Export File Nodes in SAS Studio

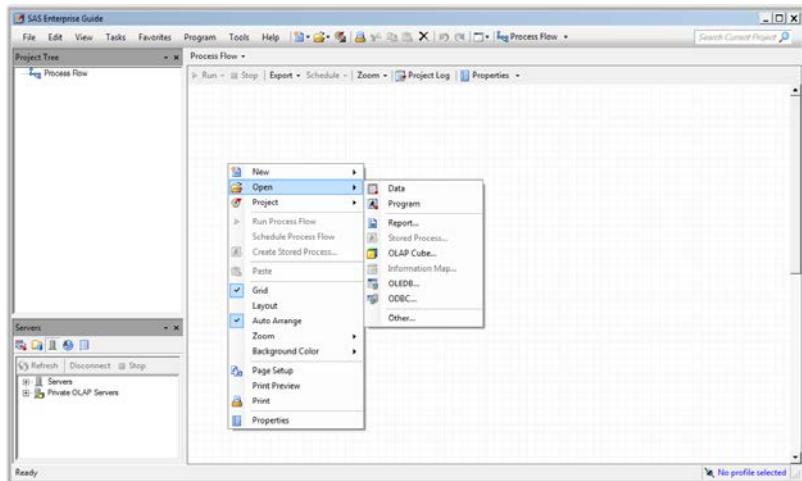
Enterprise Guide

Exporting Files in SAS Enterprise Guide

In the following example, a SAS data set is added to the SAS Enterprise Guide project and the [Export <Data Set> as a Step in the Project](#) option is used to specify the output data location, delimiter information, fields to import, and other advanced options. The resulting process flow contains a Data node for the input data set, an Export File node, and a Data node for the output data.

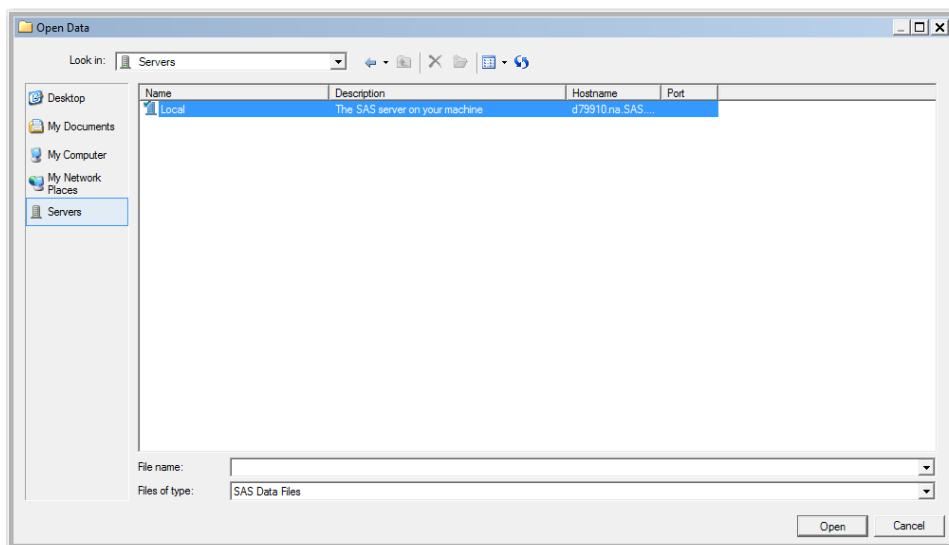
Select SAS Data Set to Export

1. Right-click in the project and select **Open -> Data**.



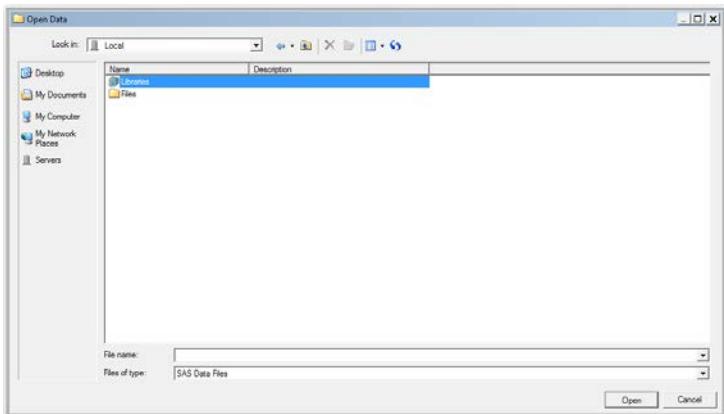
Display 45 – Opening Your Data from the Process Flow

2. Select the server that contains the SAS data set you want to export.



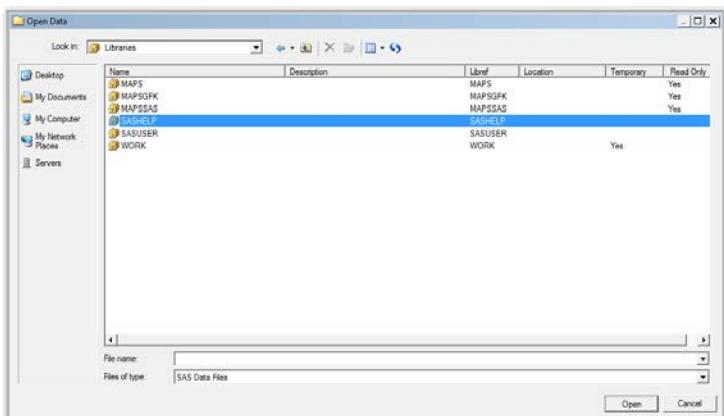
Display 46 – Selecting a Server in the Open Data Dialog Box

3. Select Libraries.



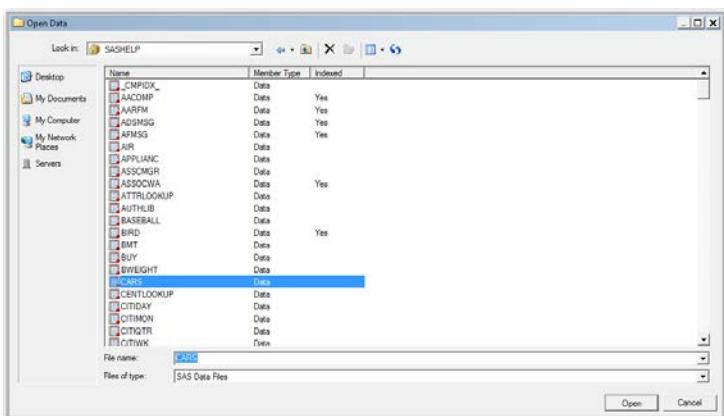
Display 47 – Opening the Libraries Folder in the Open Data Dialog Box

4. Select the library that contains the SAS data set you want to export.



Display 48 – Select the Library That Contains the SAS Data Set

5. Select the SAS data set you want to export.

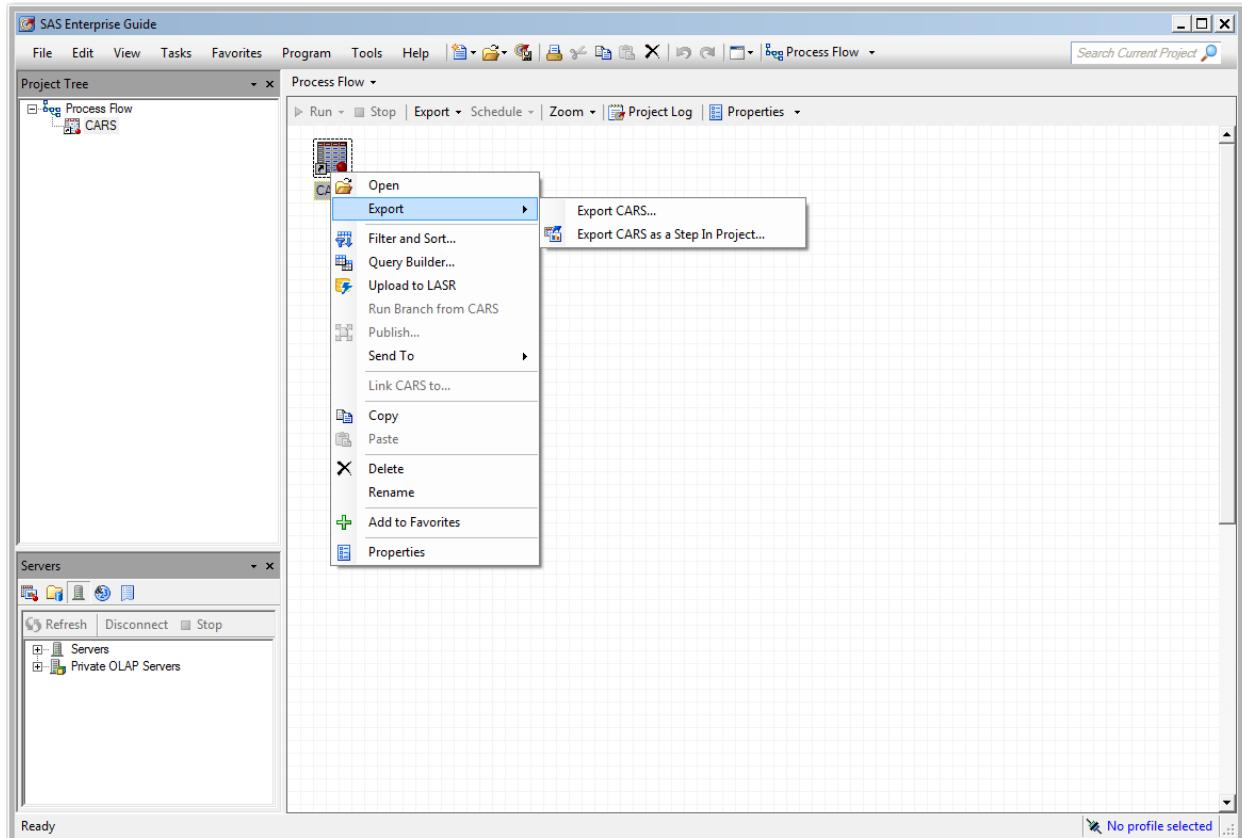


Display 49 – Select the Data Set in the Open Data Dialog Box

A Data node for the SAS data set appears in your SAS Enterprise Guide project.

Export <Data Set> as a Step in Project

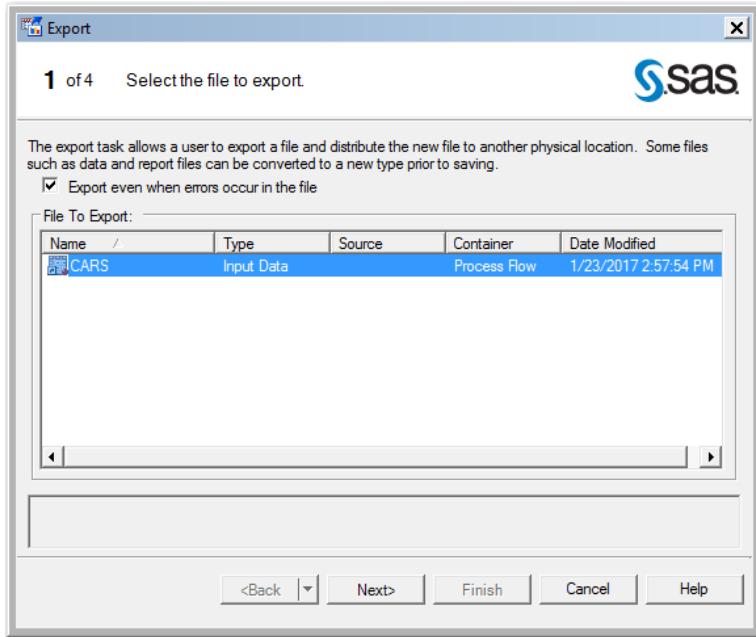
Right-click the Data node for the SAS data set to use the **Export <Data Set> as a Step In Project** menu item to open the Export wizard.



Display 50 – Exporting a Data Node in SAS Enterprise Guide

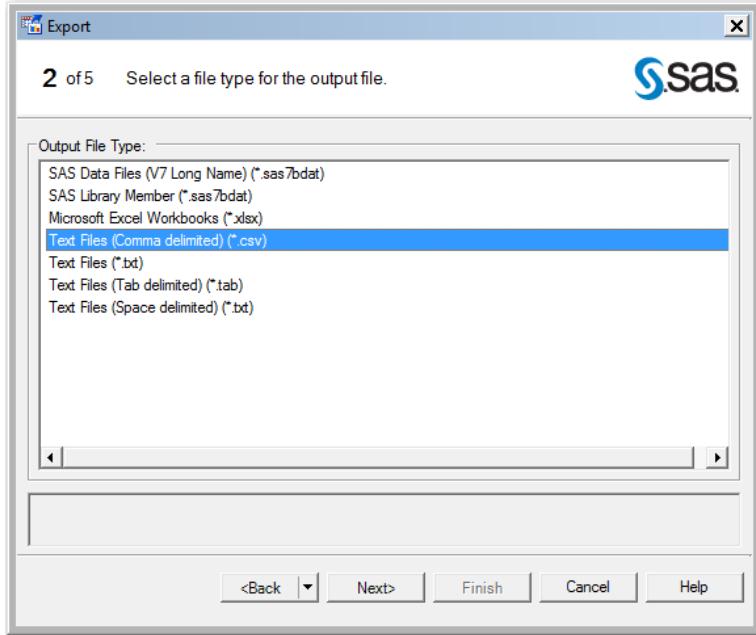
Export Wizard in SAS Enterprise Guide

The first panel of the Export wizard displays the SAS data set you have chosen to export. There is an **Export even when errors occur in the file** check box in this panel. The impact of this check box on a converted EGP in SAS Studio has not been researched.



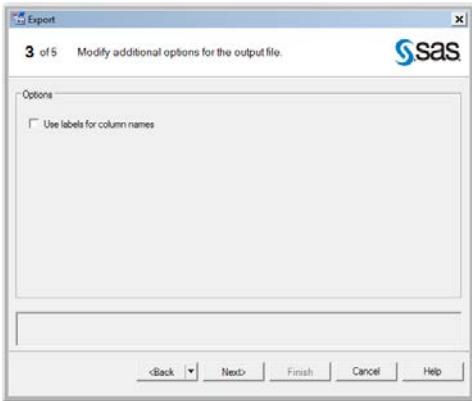
Display 51 – Selecting the File to Export in the Export Wizard

In the second step of the Export wizard, specify the type of the output file for the exported SAS data set. SAS Studio supports all file types.



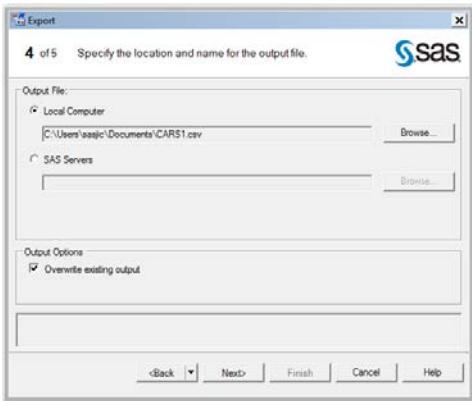
Display 52 - Selecting the File Type for the Exported Data

The third step has the **Use labels for column names** check box. SAS Studio supports this option.



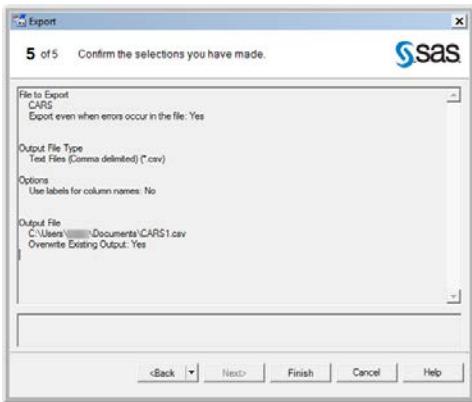
Display 53 - Additional Options in the Export Wizard

In the fourth step of the Export wizard, you specify where to write the exported output file. SAS Studio can write this file only to the file system available to its connected workspace server. If the workspace server in SAS Enterprise Guide is different from what is specified in SAS Studio, the Conversion Report will display a mismatched environment message.



Display 54 - Specifying the Output Location in the Export Wizard

The final step provides a summary of the selections you have made.

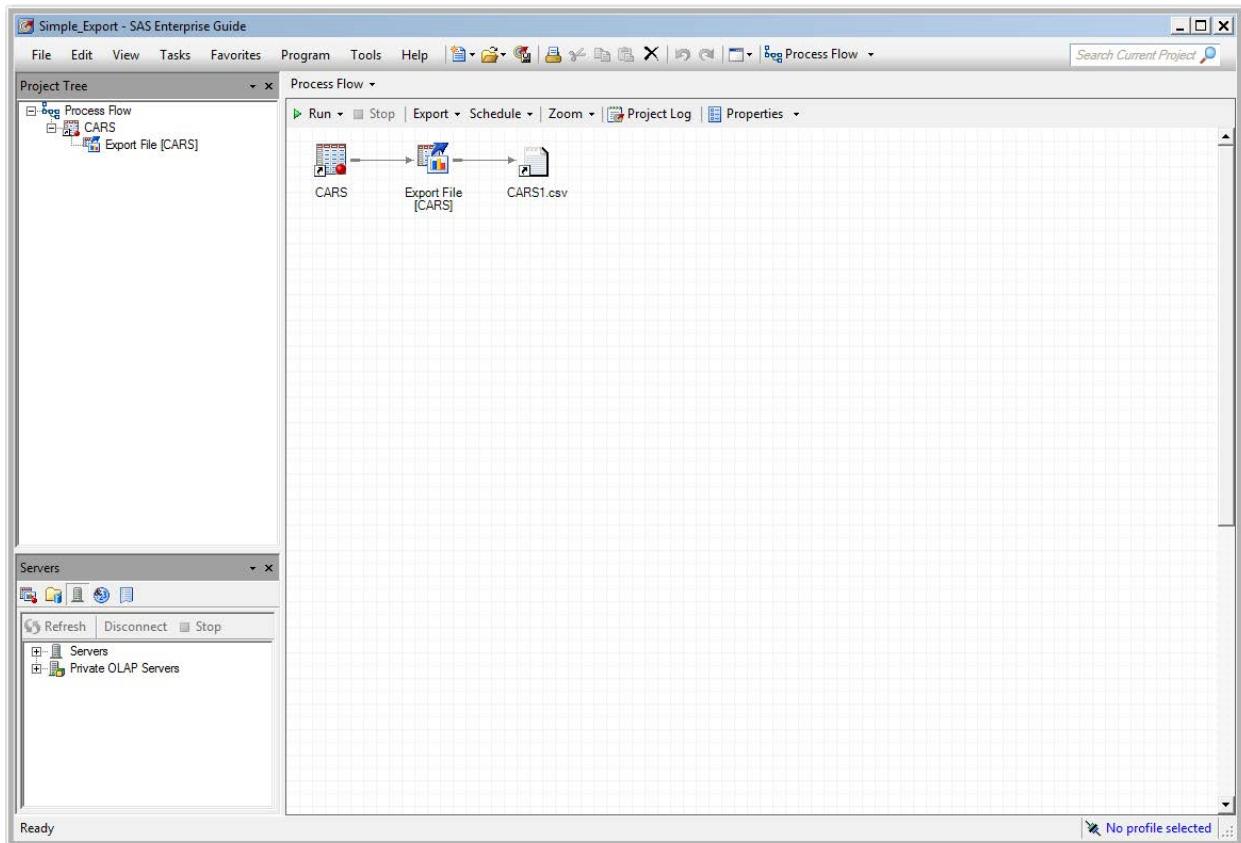


Display 55 - Summary Panel in the Export Wizard

The following display shows the resulting process flow when you run the Export wizard.

Process Flow for Export File in SAS Enterprise Guide

The process flow for Export File contains a Data node for the input SAS data set, an Export File node, and a Data node for the output file. You can contrast this flow to the [Converted Export Flow in SAS Studio](#).

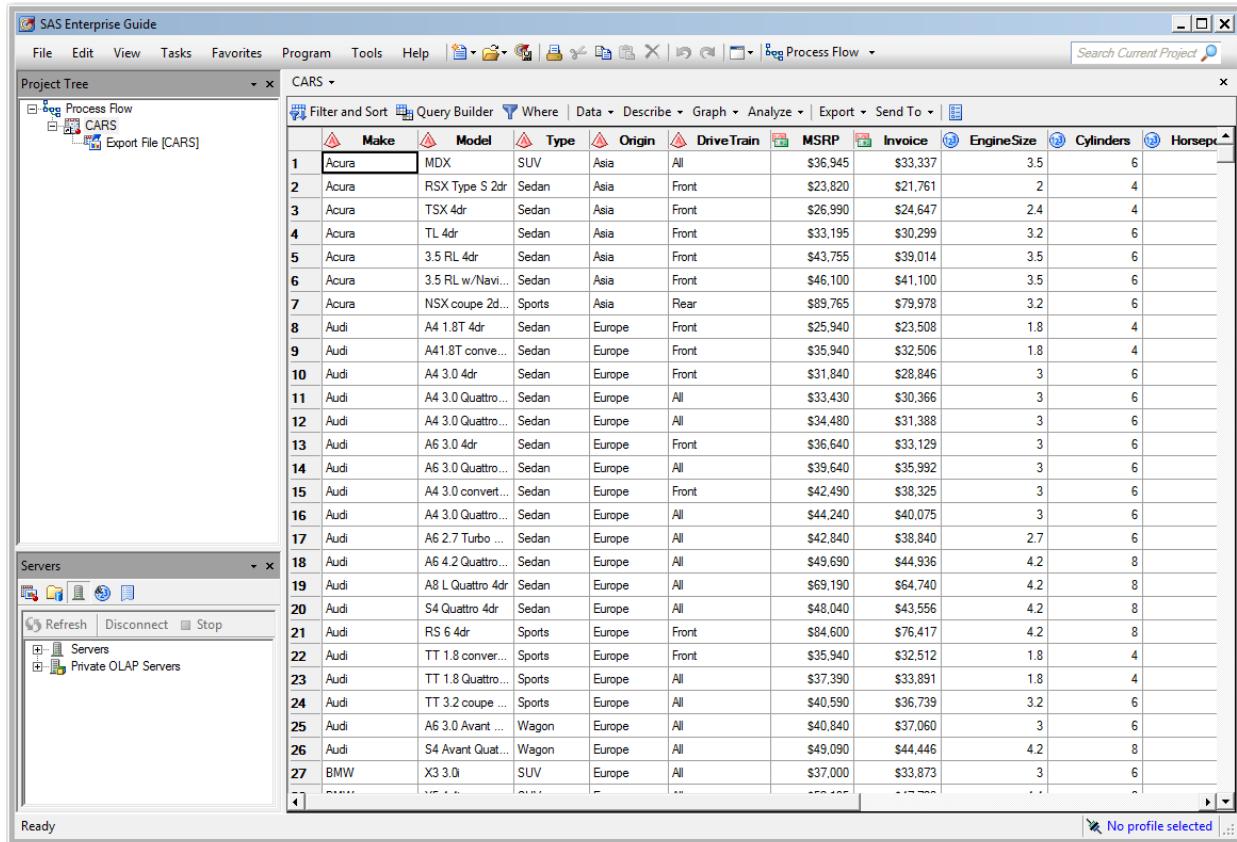


Display 56 - Resulting Process Flow in SAS Enterprise Guide

Nodes in the Process Flow for Export File in SAS Enterprise Guide

Node for Input SAS Data Set in Enterprise Guide

The first node in the process flow is a Data node for the input SAS data set. Open this node to see the contents of the data set, update the data, as well as many other functions. You can contrast this functionality to the converted [Data Set Node for Input Data in SAS Studio](#).



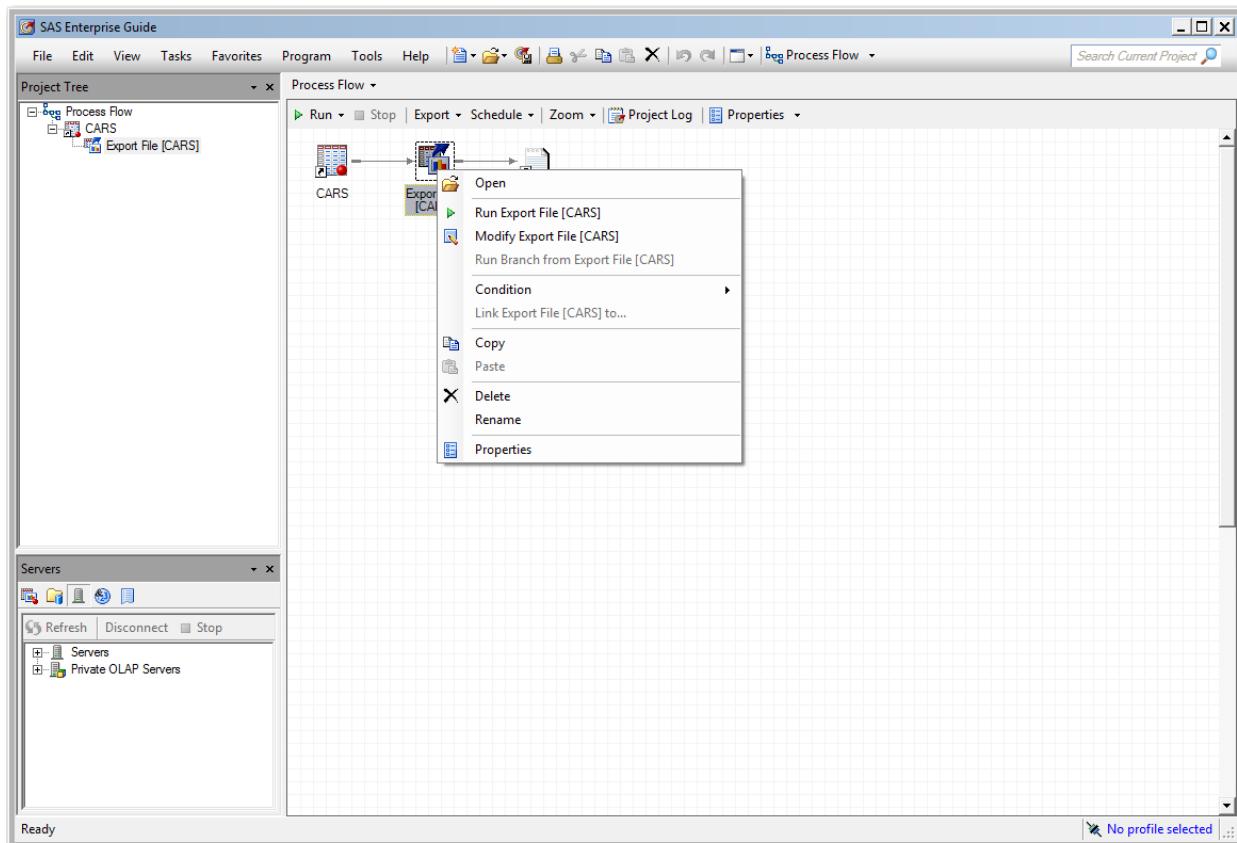
The screenshot shows the SAS Enterprise Guide interface with the 'CARS' data set open in a data grid. The data grid displays 27 rows of car information with columns for Make, Model, Type, Origin, DriveTrain, MSRP, Invoice, EngineSize, Cylinders, and Horsepower. The 'Make' column is sorted in ascending order, with 'Acura' at the top. The 'Servers' pane on the left shows a list of servers and private OLAP servers.

	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Horsepower
1	Acura	MDX	SUV	Asia	All	\$36,945	\$33,337	3.5	6	
2	Acura	RSX Type S 2dr	Sedan	Asia	Front	\$23,820	\$21,761	2	4	
3	Acura	TSX 4dr	Sedan	Asia	Front	\$25,990	\$24,647	2.4	4	
4	Acura	TL 4dr	Sedan	Asia	Front	\$33,195	\$30,299	3.2	6	
5	Acura	3.5 RL 4dr	Sedan	Asia	Front	\$43,755	\$39,014	3.5	6	
6	Acura	3.5 RL w/Navi...	Sedan	Asia	Front	\$46,100	\$41,100	3.5	6	
7	Acura	NSX coupe 2d...	Sports	Asia	Rear	\$89,765	\$79,978	3.2	6	
8	Audi	A4 1.8T 4dr	Sedan	Europe	Front	\$25,940	\$23,508	1.8	4	
9	Audi	A41.8T conver...	Sedan	Europe	Front	\$35,940	\$32,506	1.8	4	
10	Audi	A4 3.0 4dr	Sedan	Europe	Front	\$31,840	\$28,846	3	6	
11	Audi	A4 3.0 Quattro...	Sedan	Europe	All	\$33,430	\$30,366	3	6	
12	Audi	A4 3.0 Quattro ...	Sedan	Europe	All	\$34,480	\$31,388	3	6	
13	Audi	A6 3.0 4dr	Sedan	Europe	Front	\$36,640	\$33,129	3	6	
14	Audi	A6 3.0 Quattro...	Sedan	Europe	All	\$39,640	\$35,992	3	6	
15	Audi	A4 3.0 convert...	Sedan	Europe	Front	\$42,490	\$38,325	3	6	
16	Audi	A4 3.0 Quattro...	Sedan	Europe	All	\$44,240	\$40,075	3	6	
17	Audi	A6 2.7 Turbo ...	Sedan	Europe	All	\$42,840	\$38,840	2.7	6	
18	Audi	A6 4.2 Quattro ...	Sedan	Europe	All	\$49,690	\$44,936	4.2	8	
19	Audi	A8 L Quattro 4dr	Sedan	Europe	All	\$69,190	\$64,740	4.2	8	
20	Audi	S4 Quattro 4dr	Sedan	Europe	All	\$48,040	\$43,556	4.2	8	
21	Audi	RS 6 4dr	Sports	Europe	Front	\$84,600	\$76,417	4.2	8	
22	Audi	TT 1.8 conver...	Sports	Europe	Front	\$35,940	\$32,512	1.8	4	
23	Audi	TT 1.8 Quattro ...	Sports	Europe	All	\$37,390	\$33,891	1.8	4	
24	Audi	TT 3.2 coupe ...	Sports	Europe	All	\$40,590	\$36,739	3.2	6	
25	Audi	A6 3.0 Avant ...	Wagon	Europe	All	\$40,840	\$37,060	3	6	
26	Audi	S4 Avant Quat...	Wagon	Europe	All	\$49,090	\$44,446	4.2	8	
27	BMW	X3 3.0i	SUV	Europe	All	\$37,000	\$33,873	3	6	

Display 57 - Contents of Data Node for Input SAS Data Set in SAS Enterprise Guide

Export File Node in SAS Enterprise Guide

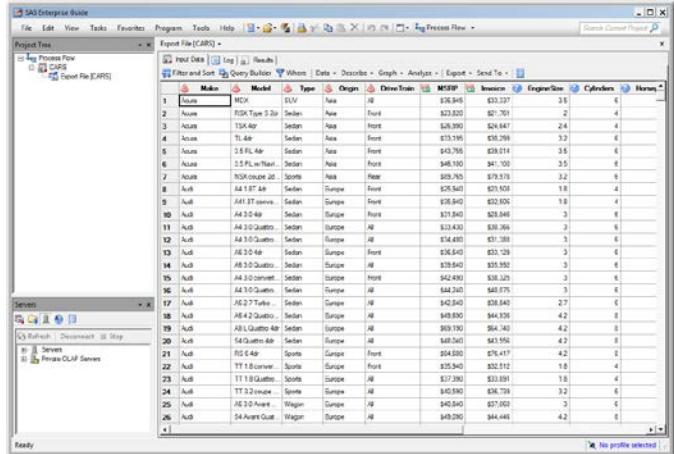
Right-click the Export File node to open or modify the export file. If you select **Modify Export File**, the [Export wizard](#) opens, so you can change your export settings. You can contrast this node to the [Converted Export File Node in SAS Studio](#).



Display 58 - Export File Node in SAS Enterprise Guide

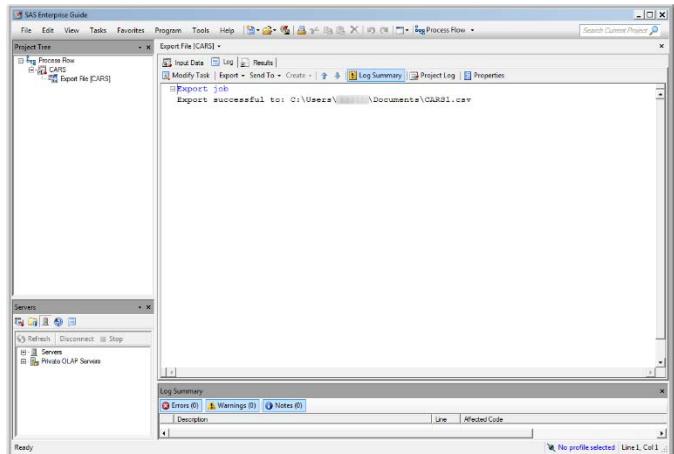
If you select **Open** from the pop-up menu, you see the **Input Data**, **Log**, and **Results** tabs as shown in the following displays.

The **Input Data** tab shows the contents of the input SAS data set.



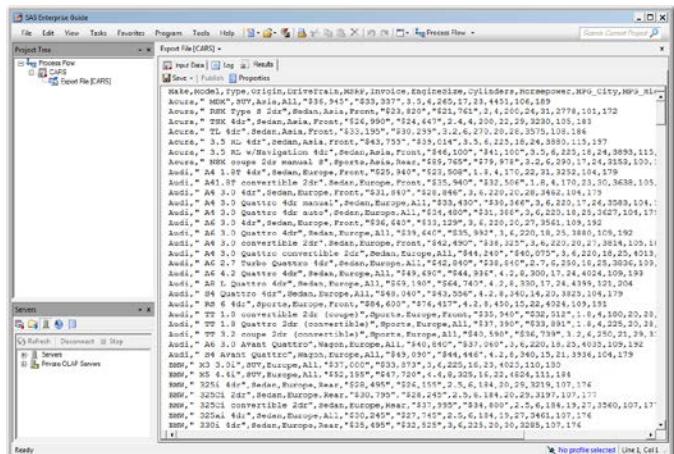
Display 59 - Contents of Input Data Tab for Export File Node in SAS Enterprise Guide

The **Log** tab provides the export status.



Display 60 - Log Tab for Export File Node in SAS Enterprise Guide

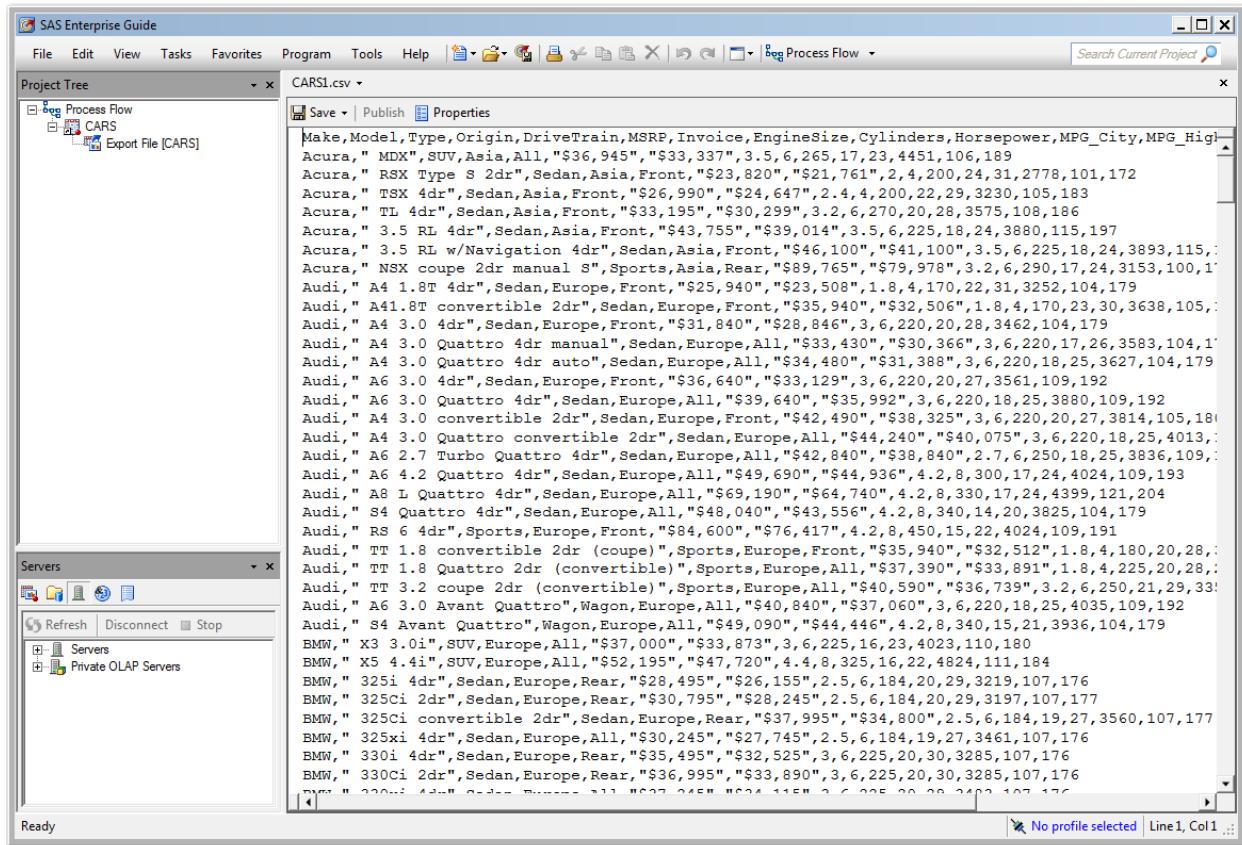
The **Results** tab displays the exported data.



Display 61 - Results Tab for Export File Node in SAS Enterprise Guide

Output Data Node for Export File in SAS Enterprise Guide

The Data node for the output data contains the exported data. You can edit and save the data in this node. Contrast this to the [SAS Program Node for Output File in SAS Studio](#).



```
Make,Model,Type,Origin,DriveTrain,MSRP,Invoice,EngineSize,Cylinders,Horsepower,MPG_City,MPG_High,Acura,"MDX",SUV,Asia,All,"$36,945","$33,337",3.5,6,265,17,23,4451,106,189,Acura,"RSX Type S 2dr",Sedan,Asia,Front,"$23,820","$21,761",2,4,200,24,31,2778,101,172,Acura,"TSX 4dr",Sedan,Asia,Front,"$26,990","$24,647",2,4,4,200,22,29,3230,105,183,Acura,"TL 4dr",Sedan,Asia,Front,"$33,195","$30,299",3,2,6,270,20,28,3575,108,186,Acura,"3.5 RL 4dr",Sedan,Asia,Front,"$43,755","$39,014",3.5,6,225,18,24,3880,115,197,Acura,"3.5 RL w/Navigation 4dr",Sedan,Asia,Front,"$46,100","$41,100",3.5,6,225,18,24,3893,115,Acura,"NSX coupe 2dr manual S",Sports,Asia,Rear,"$89,765","$79,978",3,2,6,290,17,24,3153,100,1'Audi,"A4 1.8T 4dr",Sedan,Europe,Front,"$25,940","$23,508",1,8,4,170,22,31,3252,104,179,Audi,"A4 1.8T convertible 2dr",Sedan,Europe,Front,"$35,940","$32,506",1,8,4,170,23,30,3638,105,Audi,"A4 3.0 4dr",Sedan,Europe,Front,"$31,840","$28,846",3,6,220,20,28,3462,104,179,Audi,"A4 3.0 Quattro 4dr manual",Sedan,Europe,All,"$33,430","$30,366",3,6,220,17,26,3583,104,1'Audi,"A4 3.0 Quattro 4dr auto",Sedan,Europe,All,"$34,480","$31,388",3,6,220,18,25,3627,104,179,Audi,"A6 3.0 4dr",Sedan,Europe,Front,"$36,640","$33,129",3,6,220,20,27,3561,109,192,Audi,"A6 3.0 Quattro 4dr",Sedan,Europe,All,"$39,640","$35,992",3,6,220,18,25,3880,109,192,Audi,"A4 3.0 convertible 2dr",Sedan,Europe,Front,"$42,490","$38,325",3,6,220,20,27,3814,105,181,Audi,"A4 3.0 Quattro convertible 2dr",Sedan,Europe,All,"$44,240","$40,075",3,6,220,18,25,4013,Audi,"A6 2.7 Turbo Quattro 4dr",Sedan,Europe,All,"$42,840","$38,840",2,7,6,250,18,25,3836,109,Audi,"A6 4.2 Quattro 4dr",Sedan,Europe,All,"$49,690","$44,936",4,2,8,300,17,24,4024,109,193,Audi,"A8 L Quattro 4dr",Sedan,Europe,All,"$69,190","$64,740",4,2,8,330,17,24,4399,121,204,Audi,"S4 Quattro 4dr",Sedan,Europe,All,"$48,040","$43,556",4,2,8,340,14,20,3825,104,179,Audi,"RS 6 4dr",Sports,Europe,Front,"$84,600","$76,417",4,2,8,450,15,22,4024,109,191,Audi,"TT 1.8 convertible 2dr (coupe)",Sports,Europe,Front,"$35,940","$32,512",1,8,4,180,20,28,Audi,"TT 1.8 Quattro 2dr (convertible)",Sports,Europe,All,"$37,390","$33,891",1,8,4,225,20,28,Audi,"TT 3.2 coupe 2dr (convertible)",Sports,Europe,All,"$40,590","$36,739",3,2,6,250,21,29,33,Audi,"A6 3.0 Avant Quattro",Wagon,Europe,All,"$40,840","$37,060",3,6,220,18,25,4035,109,192,Audi,"S4 Avant Quattro",Wagon,Europe,All,"$49,090","$44,446",4,2,8,340,15,21,3936,104,179,BMW,"X3 3.0i",SUV,Europe,All,"$37,000","$33,873",3,6,225,16,23,4023,110,180,BMW,"X5 4.4i",SUV,Europe,All,"$52,195","$47,720",4,4,8,325,16,22,4824,111,184,BMW,"325i 4dr",Sedan,Europe,Rear,"$28,495","$26,155",2,5,6,184,20,29,3219,107,176,BMW,"325Ci 2dr",Sedan,Europe,Rear,"$30,795","$28,245",2,5,6,184,20,29,3197,107,177,BMW,"325Ci convertible 2dr",Sedan,Europe,Rear,"$37,995","$34,800",2,5,6,184,19,27,3560,107,177,BMW,"325xi 4dr",Sedan,Europe,All,"$30,245","$27,745",2,5,6,184,19,27,3461,107,176,BMW,"330i 4dr",Sedan,Europe,Rear,"$35,495","$32,525",3,6,225,20,30,3285,107,176,BMW,"330Ci 2dr",Sedan,Europe,Rear,"$36,995","$33,890",3,6,225,20,30,3285,107,176,BMW,"320i 4dr",Sedan,Europe,Rear,"$30,245","$27,245",2,5,6,225,20,30,3285,107,176
```

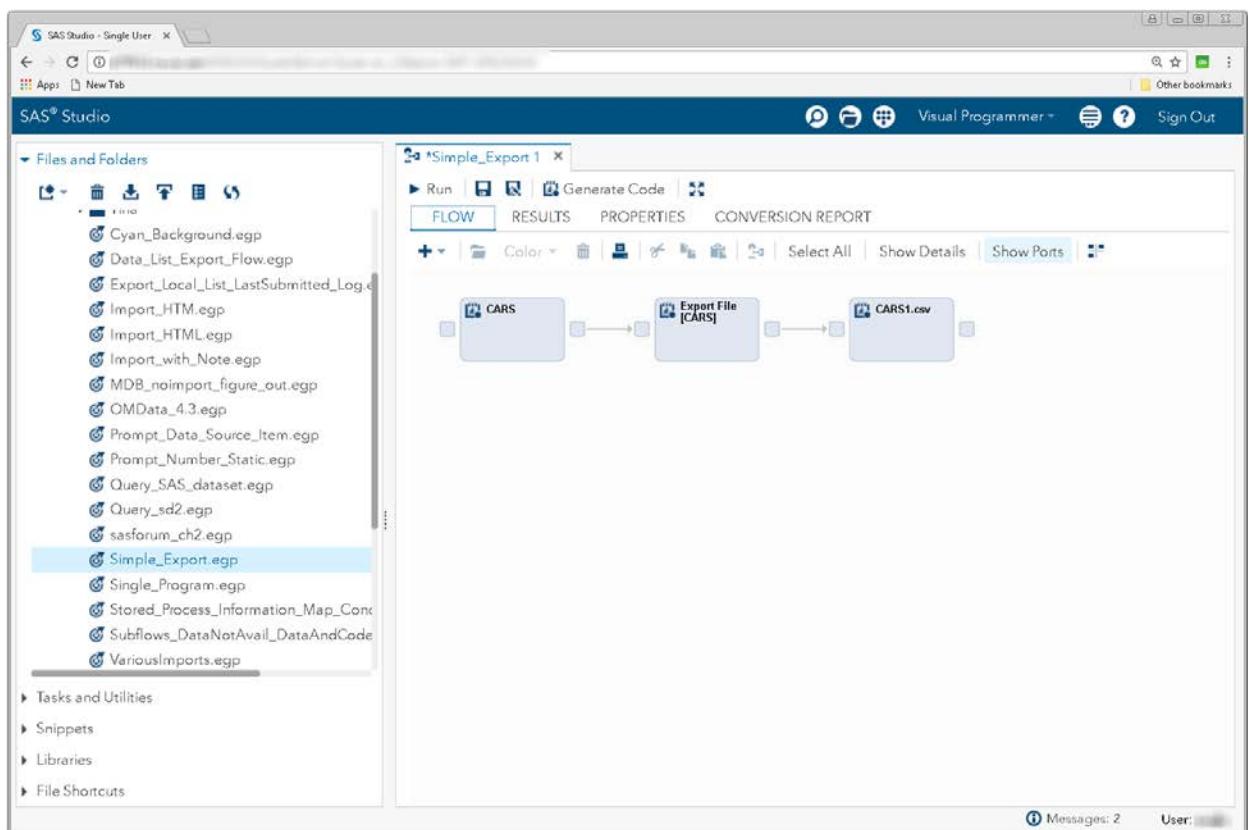
Display 62 - Contents of Node for Exported Data in SAS Enterprise Guide

SAS Studio

Converted Process Flow in SAS Studio

The process flow from SAS Enterprise looks very similar to the converted process flow in SAS Studio. While each of the nodes are represented in SAS Studio, these nodes are remarkably different in functionality.

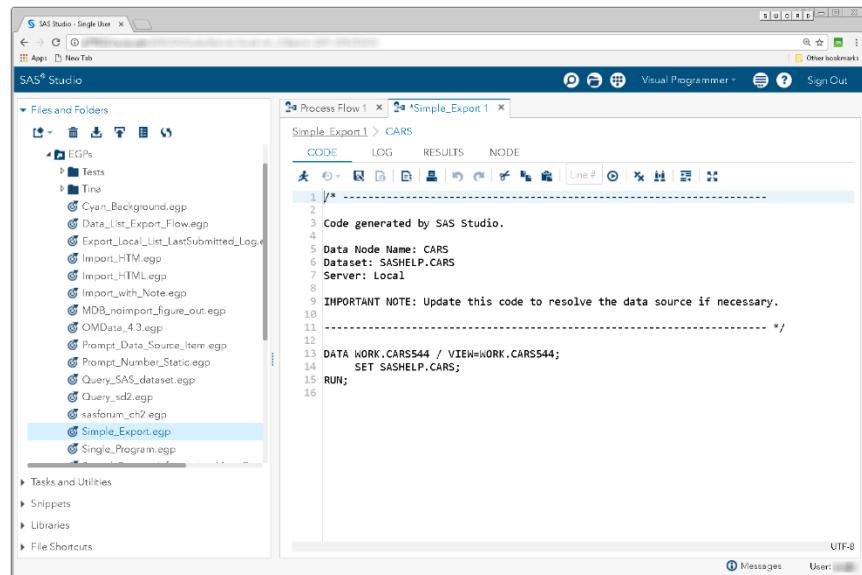
Contrast this display to the [Process flow for Export File SAS in Enterprise Guide](#).



Display 63 - Process Flow for Converted Export File Node in SAS Studio

Data Set Node for Input Data in SAS Studio

The Data node for the input data set is converted to a SAS Program node that displays the contents of the SAS data set. You can contrast this to the [Node for the Input SAS Data Set in SAS Enterprise Guide](#).

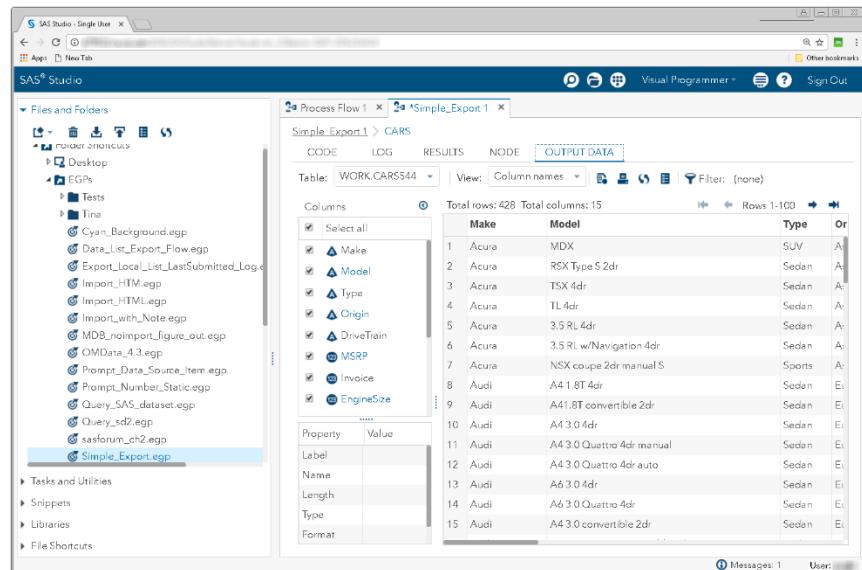


The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. On the left, the Files and Folders tree view shows various EGPs and a Simple_Export.egp file highlighted. The main workspace displays a SAS program node titled "Simple_Export" under the "CARS" process flow. The code pane contains the following SAS code:

```
1 /*-----  
2  
3 Code generated by SAS Studio.  
4  
5 Data Node Name: CARS  
6 Dataset: SASHHELP.CARS  
7 Server: Local  
8  
9 IMPORTANT NOTE: Update this code to resolve the data source if necessary.  
10  
11 -----*/  
12  
13 DATA WORK.CARS544 / VIEW=SASHHELP.CARS;  
14   SET SASHHELP.CARS;  
15 RUN;
```

Display 64 - Replacement for Export Node of Input Data in SAS Studio

When you run this node in SAS Studio, the resulting **OUTPUT DATA** tab contains the contents of the SAS data set to export.



The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. The "Simple_Export" node is running, and the "OUTPUT DATA" tab is active, displaying the contents of the CARS dataset. The table shows 428 rows and 15 columns. The columns are: Make, Model, Type, Origin, MSRP, Invoice, EngineSize, Label, Name, Length, Type, and Format. The data includes various car models like Acura MDX, Accord, and Honda Fit.

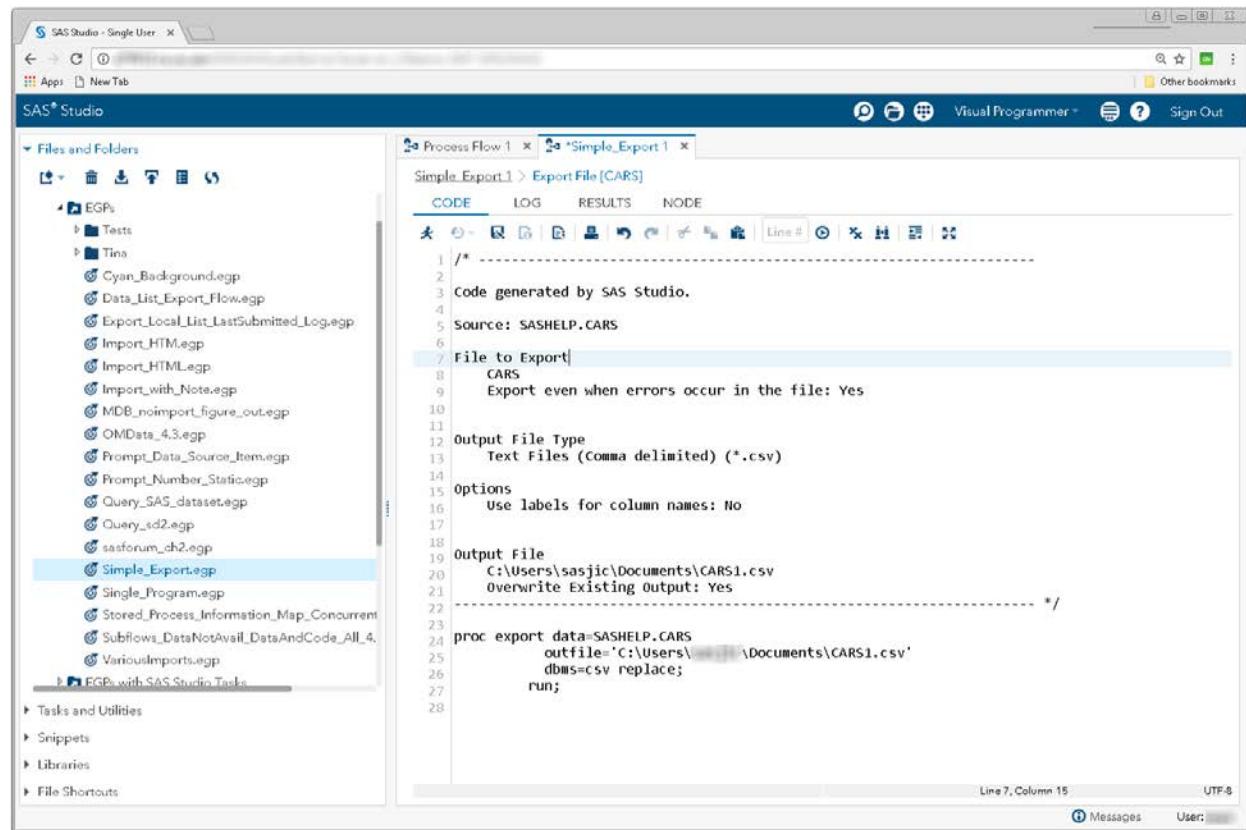
Make	Model	Type	Or
1 Acura	MDX	SUV	A
2 Acura	RDX Type S 2dr	Sedan	A
3 Acura	TSX 4dr	Sedan	A
4 Acura	TL 4dr	Sedan	A
5 Acura	3.5 RL 4dr	Sedan	A
6 Acura	3.5 RL w/Navigation 4dr	Sedan	A
7 Acura	NSX coupe 2dr manual S	Sports	A
8 Audi	A4 1.8T 4dr	Sedan	E
9 Audi	A4 1.8T convertible 2dr	Sedan	E
10 Audi	A4 3.0 4dr	Sedan	E
11 Audi	A4 3.0 Quattro 4dr manual	Sedan	E
12 Audi	A4 3.0 Quattro 4dr auto	Sedan	E
13 Audi	A6 3.0 4dr	Sedan	E
14 Audi	A6 3.0 Quattro 4dr	Sedan	E
15 Audi	A4 3.0 convertible 2dr	Sedan	E

Display 65 - Results of Running the SAS Program Node Replacement for the Input Data Node in SAS Studio

Converted Export File Node in SAS Studio

Export File nodes in SAS Enterprise Guide are converted to SAS Program nodes in SAS Studio.

SAS Enterprise Guide does not generate portable export code in the Export File nodes. Therefore, SAS Studio uses the metadata that is stored in the SAS Enterprise Guide project to generate code that can export the input data set. If the output file location specified in the SAS Enterprise Guide process flow is not available using the active connection in SAS Studio, an error is written to the Conversion Report, but the code is still generated. Contrast this display to the [Export File Node in SAS Enterprise Guide](#).



The screenshot shows the SAS Studio interface with the 'Visual Programmer' tab selected. On the left, the 'Files and Folders' sidebar lists various EGPs, including 'Simple_Export1.egp' which is currently selected. The main workspace displays the converted SAS program code:

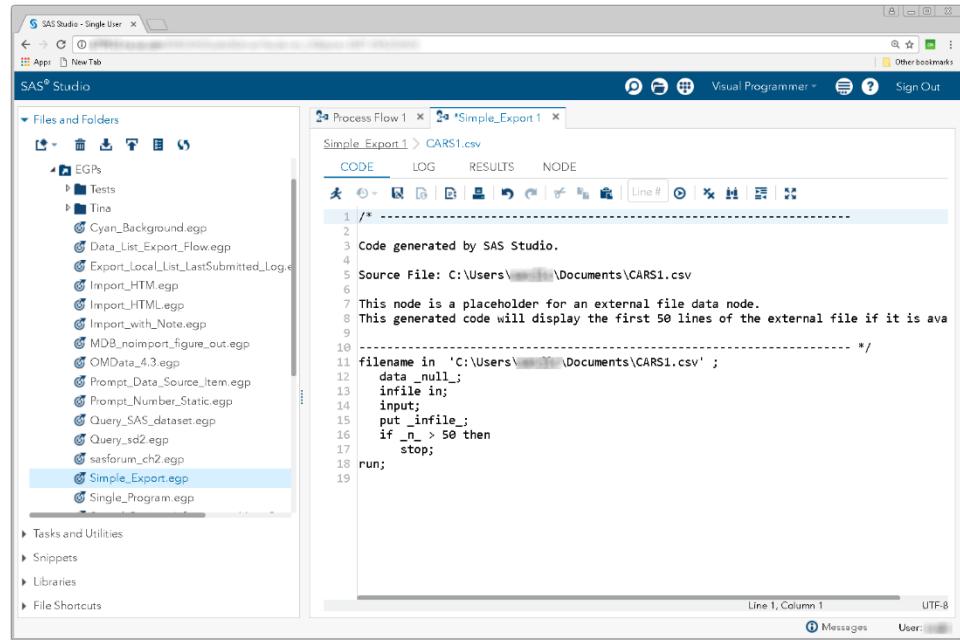
```
1 /* -----
2
3  Code generated by SAS Studio.
4
5  Source: SASHELP.CARS
6
7  File to Export|
8  CARS
9    Export even when errors occur in the file: Yes
10
11  Output File Type
12    Text Files (comma delimited) (*.csv)
13
14  Options
15    Use labels for column names: No
16
17  Output File
18    C:\Users\sasjic\Documents\CARS1.csv
19    Overwrite Existing Output: Yes
20
21
22
23  proc export data=SASHELP.CARS
24    outfile='C:\Users\sasjic\Documents\CARS1.csv'
25    dbms=csv replace;
26    run;
27
28
```

The code is a SAS PROC EXPORT statement that exports data from the SASHELP.CARS library to a CSV file named C:\Users\sasjic\Documents\CARS1.csv, replacing any existing file.

Display 66 - Converted Export File Node in SAS Studio

SAS Program Node for Output File in SAS Studio

The converted Data node for the output file becomes a [data sampling SAS Program node](#). Contrast this to the [Output Data Node for Export File in Enterprise Guide](#).

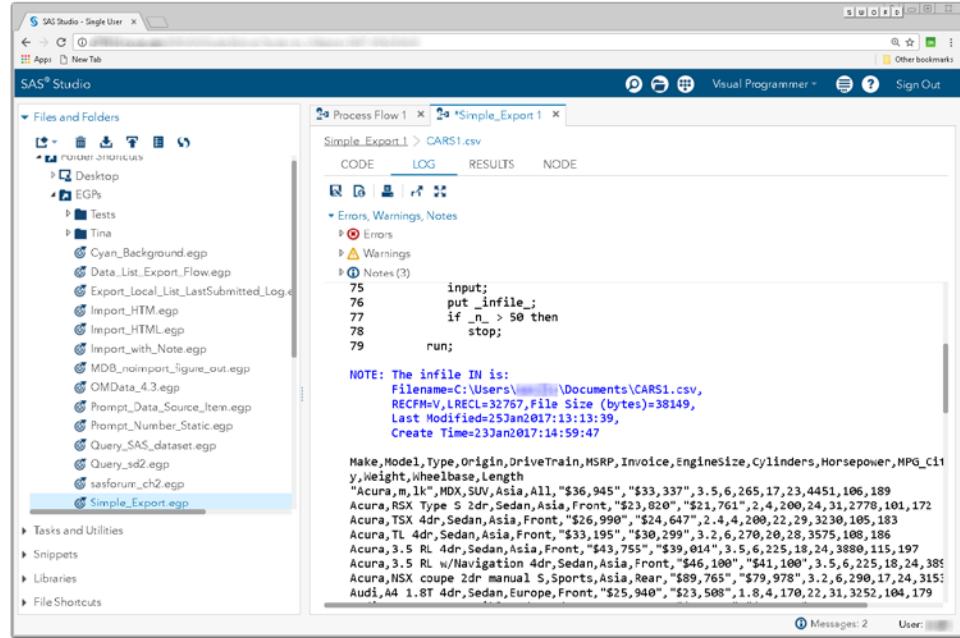


The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. On the left, the 'Files and Folders' sidebar lists various EGP files, including 'Simple_Export.egp'. In the main workspace, a process flow named 'Simple_Export 1' is displayed, with a node for 'CARS1.csv'. The 'CODE' tab is active, showing the generated SAS code:

```
/*
Code generated by SAS Studio.
Source File: C:\Users\[REDACTED]\Documents\CARS1.csv
This node is a placeholder for an external file data node.
This generated code will display the first 50 lines of the external file if it is available.
filename in 'C:\Users\[REDACTED]\Documents\CARS1.csv';
data _null_;
infile in;
input;
put _infile_;
if _n_ > 50 then
stop;
run;
```

Display 67 - Generated Code for the Converted Data Node in SAS Studio

If you run the data sampling node, you will see up to 50 lines of your output file in the **LOG** tab.



The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. On the left, the 'Files and Folders' sidebar lists various EGP files, including 'Simple_Export.egp'. In the main workspace, a process flow named 'Simple_Export 1' is displayed, with a node for 'CARS1.csv'. The 'LOG' tab is active, showing the output from running the node:

NOTE: The infile IN is:
Filename=C:\Users\[REDACTED]\Documents\CARS1.csv,
RECFM=V,LRECL=32767,File Size (bytes)=38149,
Last Modified=25Jan2017:13:13:39,
Create Time=23Jan2017:14:59:47

Make,Model,Type,Origin,DriveTrain,MSRP,Invoice,EngineSize,Cylinders,Horsepower,MPG_City,Weight,Wheelbase,Length
"Acura_m_1K",MDX,SUV,Asia,All,"\$36,945","\$33,337",3.5,6,265,17,23,4451,106,189
Acura,RSX Type S 2dr,Sedan,Asia,Front,"\$23,820","\$21,761",2,4,200,24,31,2778,101,172
Acura,TSX 4dr,Sedan,Asia,Front,"\$26,990","\$24,647",2,4,4,200,22,29,3230,105,183
Acura,TL 4dr,Sedan,Asia,Front,"\$33,195","\$30,299",3,2,6,270,20,28,3575,108,186
Acura,3.5 RL 4dr,Sedan,Asia,Front,"\$43,755","\$39,014",3,5,6,225,18,24,3880,115,197
Acura,3.5 RL w/Navigation 4dr,Sedan,Asia,Front,"\$46,100","\$41,100",3,5,6,225,18,24,385
Acura,NSX coupe 2dr manual S,Sports,Asia,Rear,"\$89,765","\$79,978",3,2,6,290,17,24,3153
Audi,A4 1.8T 4dr,Sedan,Europe,Front,"\$25,940","\$23,508",1,8,4,170,22,31,3252,104,179

Display 68 - Sampling the Output File

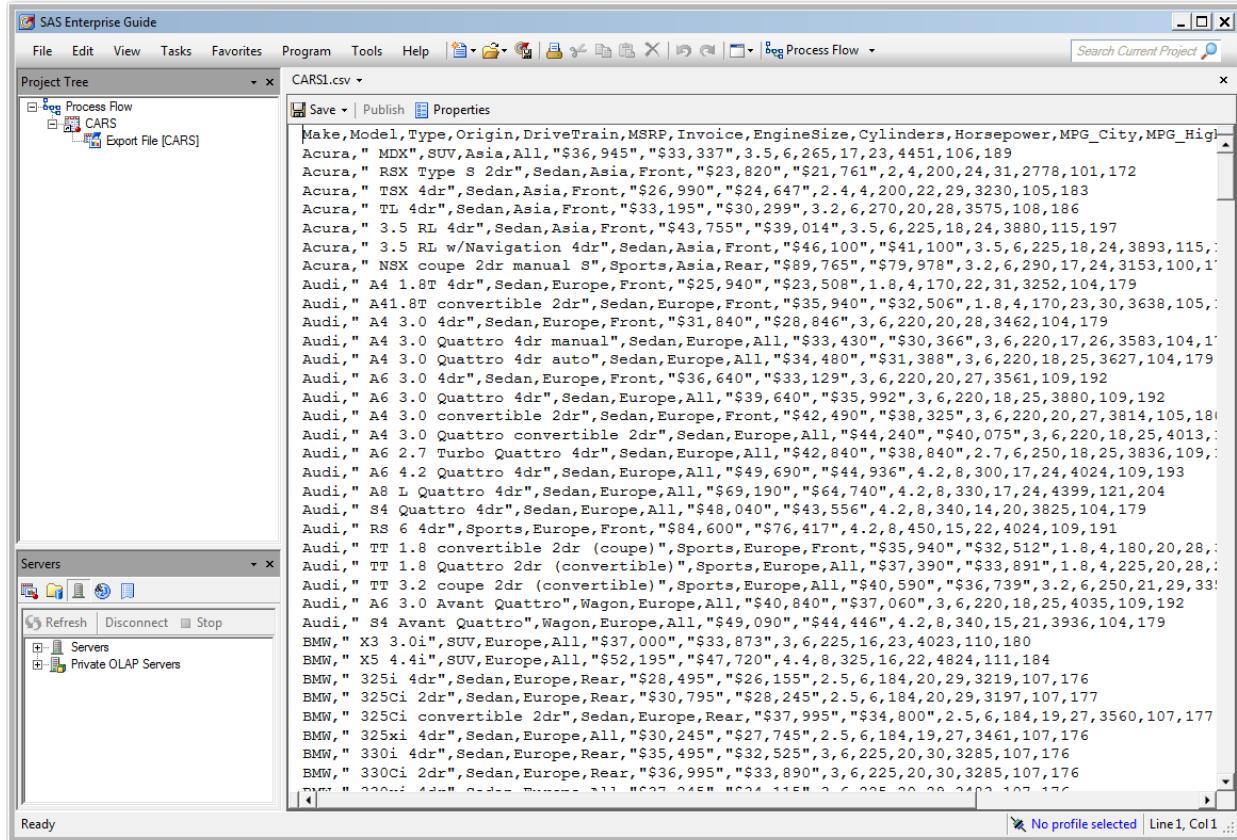
DATA NODES

Because process flows in SAS Studio 3.6 do not support Data nodes, the EGP conversion process converts them to SAS Program nodes.

The converted SAS Program node contains SAS code that:

- contains comments about the data location
- contains SAS code that samples the contents of the data in many cases. The generated code depends on the type of data in the Data node.

Contrast this functionality to Data nodes in SAS Enterprise Guide where you can open external files (excluding Microsoft Excel files) and edit them.



Display 69 Data Node for External File in SAS Enterprise Guide

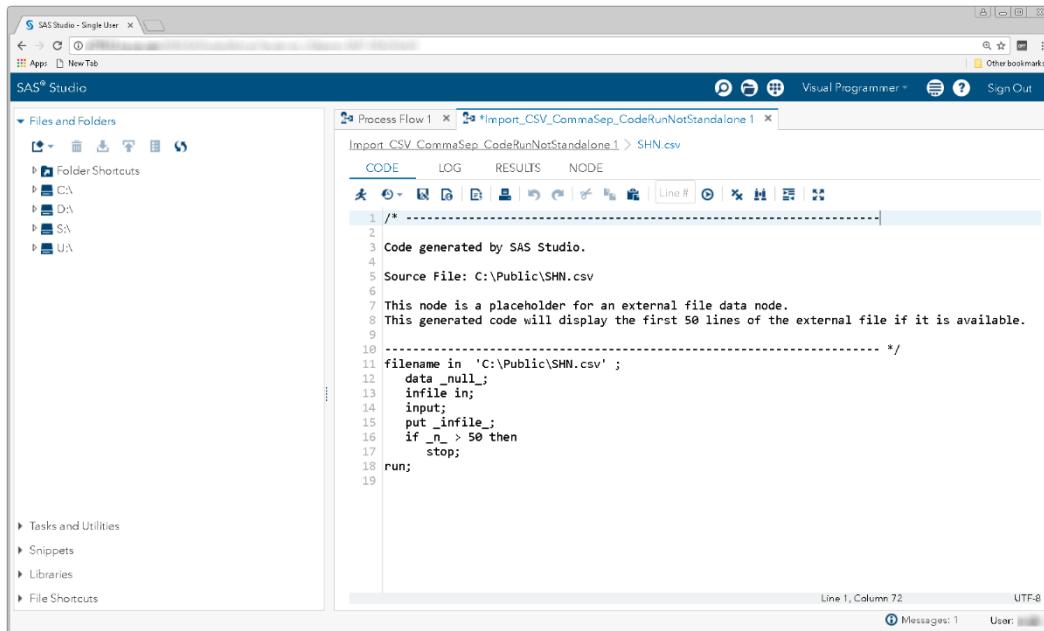
The current plan is for process flows in SAS Studio 5.1 to support Data nodes.

Data Sampling

The first 50 lines of any directly readable text file, including CSV files, fixed length text files, HTML files, and tab delimited text files, are sampled. Data nodes that represent SAS data sets show a view of table rows.

Here are some examples of what you will see for these converted nodes.

CSV

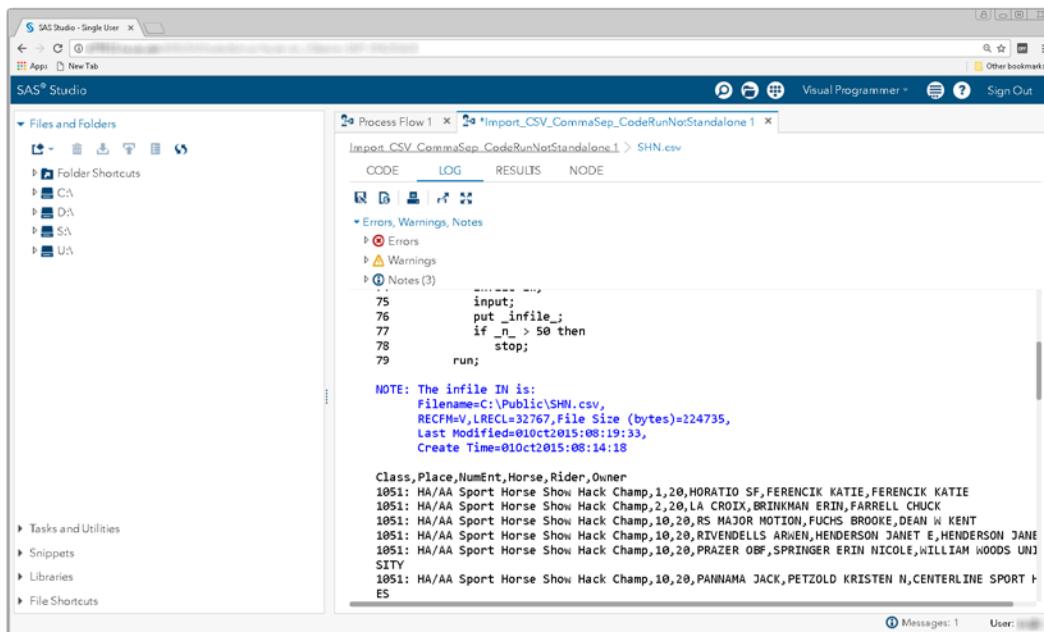


The screenshot shows the SAS Studio interface with the Visual Programmer tool open. The left sidebar contains navigation links like 'Files and Folders', 'Tasks and Utilities', 'Snippets', 'Libraries', and 'File Shortcuts'. The main workspace displays a code editor for a process flow named 'Import_CSVDotCommaSep_CodeRunNotStandalone1'. The 'CODE' tab is selected, showing the generated SAS code:

```
1 /* -----  
2  
3 Code generated by SAS Studio.  
4  
5 Source File: C:\Public\SHN.csv  
6  
7 This node is a placeholder for an external file data node.  
8 This generated code will display the first 50 lines of the external file if it is available.  
9----- */  
10 filename in 'C:\Public\SHN.csv';  
11 data _null_;  
12 infile in;  
13 input;  
14 put _infile_;  
15 if _n_ > 50 then  
16 stop;  
17 run;  
18  
19
```

The status bar at the bottom indicates 'Line 1, Column 72' and 'UTF-8' encoding.

Display 70 - Converted Data Node for CSV File



This screenshot is similar to Display 70, showing the Visual Programmer interface for the same process flow. The 'LOG' tab is now selected, displaying the log output:

```
-----  
75 input;  
76 put _infile_;  
77 if _n_ > 50 then  
78 stop;  
79 run;
```

Below the code, a 'NOTE' section provides details about the infile:

NOTE: The infile IN is:
Filename=C:\Public\SHN.csv;
RECFM=V,LRECL=32767,File Size (bytes)=124735,
Last Modified=01Oct2015:08:19:33,
Create Time=01Oct2015:08:14:18

Following the note is a list of data rows:

```
Class,Place,NumEnt,Horse,Rider,Owner  
1051: HA/AA Sport Horse Show Hack Champ,1,20,HORATIO SF,FERENCIK KATIE,FERENCIK KATIE  
1051: HA/AA Sport Horse Show Hack Champ,2,20,LA CROIX,BRINKMAN ERIN,FARRELL CHUCK  
1051: HA/AA Sport Horse Show Hack Champ,10,20,RS MAJOR NOTION,FUCHS BROOKE,DEAN W KENT  
1051: HA/AA Sport Horse Show Hack Champ,10,20,RIVENDELLS ARDEN,HENDERSON JANET E,HENDERSON JANE  
SITY  
1051: HA/AA Sport Horse Show Hack Champ,10,20,PRAZER OBF,SPRINGER ERIN NICOLE,WILLIAM WOODS UNI  
ES
```

The status bar at the bottom indicates 'Messages: 1' and 'User: [redacted]'

Display 71 - Log of Converted Data Node for CSV File

Fixed Length Text File

The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. On the left, the 'Files and Folders' sidebar is open, showing a tree structure with 'Tests' expanded, containing 'Local', 'Import', 'Fixed_Length', and several 'Import_Fixed_NoEmbed.egp' files. The main workspace displays a code editor for a process flow named 'Import_Fixed_NoEmbed1'. The code is as follows:

```

1  /*
2
3  Code generated by SAS Studio.
4
5  Source File: C:\Public\SHN_length_fields.txt
6
7  This node is a placeholder for an external file data node.
8  This generated code will display the first 50 lines of the external file if it is available.
9
10 -----
11 filename in 'C:\Public\SHN_length_fields.txt';
12 data _null_;
13 infile in;
14 input;
15 put _infile_;
16 if _n_ > 50 then
17   stop;
18 run;
19

```

Display 72 - Converted Data Node for Fixed Length Text File

The screenshot shows the SAS Studio interface with the 'LOG' tab selected in the Visual Programmer tab bar. The log window displays the generated code from Display 72, followed by a note about the infile and a table of data records.

NOTE: The infile IN is:

```

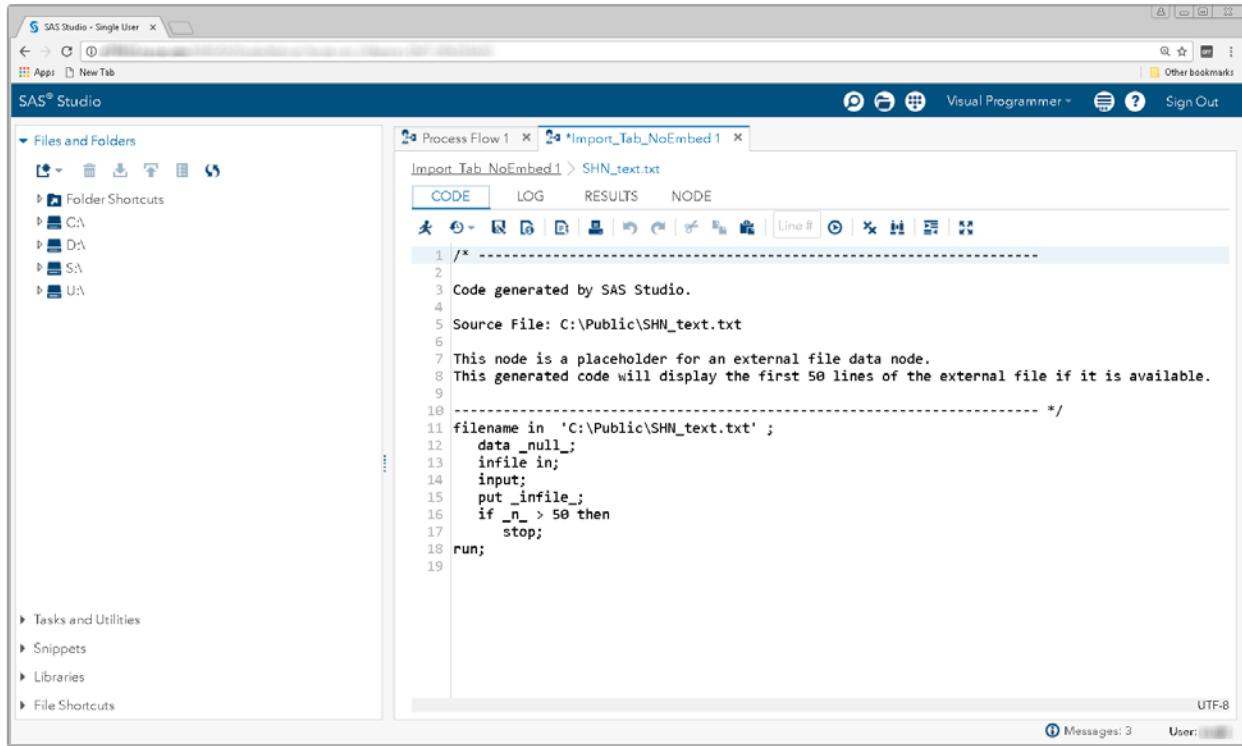
Filename=C:\Public\SHN_length_fields.txt,
RECFM=V,LRECL=32767,File Size (bytes)=2117,
Last Modified=13Sep2016:09:59:11,
Create Time=13Sep2016:09:44:24

```

1407:	HA/AA Dressage First Level AAOTR Champ JEFFREYS-CHEN JENNIFER K	DE LA NOIR
1407:	HA/AA Dressage First Level AAOTR Champ ROSENTHAL ALYSSA	PSYNISTER PLAYER
1408:	HA/AA Dressage First Level ATR Champ UPCURCH JENNA	SRC ALEJANDRO
1408:	HA/AA Dressage First Level ATR Champ PFEIL KATHERINE	HUDSON KD
1408:	HA/AA Dressage First Level ATR Champ JEFFREYS-CHEN JENNIFER K	DE LA NOIR
1502:	HA/AA Dressage Training Level AAOTR Champ	SPLASH DANCE KID

Display 73 - Log for Converted Data Node for Fixed Length Text File

Tab Delimited Text File



The screenshot shows the SAS Studio interface with the Visual Programmer tool open. The left sidebar contains 'Files and Folders' with shortcuts for CA, DA, SA, and UA. Below that are 'Tasks and Utilities', 'Snippets', 'Libraries', and 'File Shortcuts'. The main area displays a code editor titled 'Import_Tab_NoEmbed 1 > SHN_text.txt'. The 'CODE' tab is selected, showing the following SAS code:

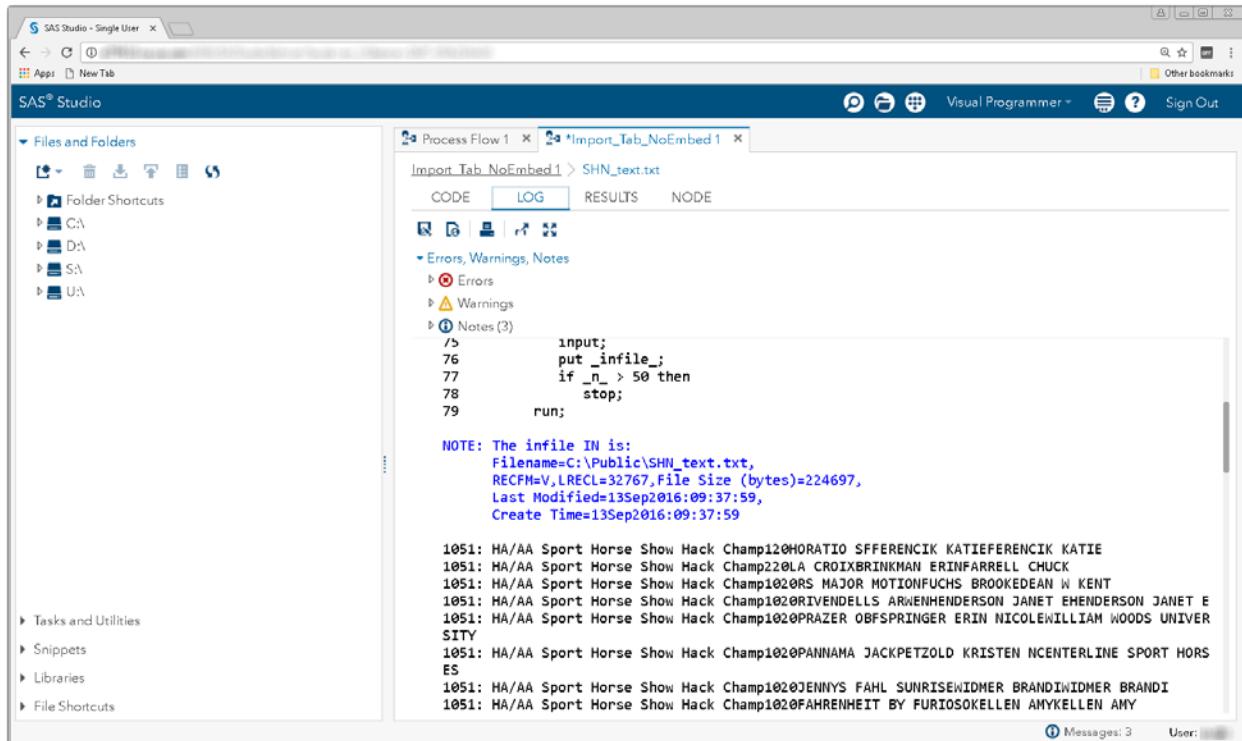
```

1 /* -----
2
3 Code generated by SAS Studio.
4
5 Source File: C:\Public\SHN_text.txt
6
7 This node is a placeholder for an external file data node.
8 This generated code will display the first 50 lines of the external file if it is available.
9
10 -----
11 filename in 'C:\Public\SHN_text.txt';
12 data _null_;
13 infile in;
14 input;
15 put _infile_;
16 if _n_ > 50 then
17 stop;
18 run;
19

```

The status bar at the bottom right shows 'UTF-8', 'Messages: 3', and 'User: [redacted]'

Display 74 - Code for Converted Data Node for Tabbed Text File



The screenshot shows the SAS Studio interface with the Visual Programmer tool open. The left sidebar contains 'Files and Folders' with shortcuts for CA, DA, SA, and UA. Below that are 'Tasks and Utilities', 'Snippets', 'Libraries', and 'File Shortcuts'. The main area displays a code editor titled 'Import_Tab_NoEmbed 1 > SHN_text.txt'. The 'LOG' tab is selected, showing the following log output:

- Errors, Warnings, Notes
- Errors
- Warnings
- Notes(3)


```

75      input;
76      put _infile_;
77      if _n_ > 50 then
78      stop;
79      run;

```

NOTE: The infile IN is:
 Filename=C:\Public\SHN_text.txt,
 RECFM=V,LRECL=32767,File Size (bytes)=224697,
 Last Modified=13Sep2016:09:37:59,
 Create Time=13Sep2016:09:37:59

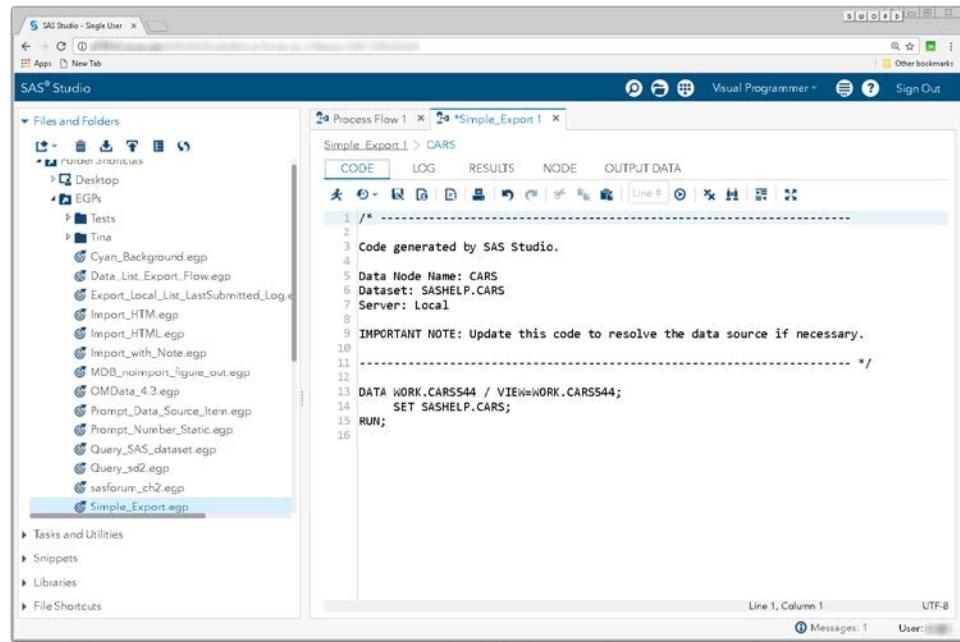
The log continues with numerous lines of data, each starting with a timestamp and a series of codes and names, such as '1051: HA/AA Sport Horse Show Hack Champ1020HORATIO SFFERENCIK KATIEFERENCIK KATIE', '1051: HA/AA Sport Horse Show Hack Champ220LA CROIXBRINKMAN ERINFARRELL CHUCK', etc.

The status bar at the bottom right shows 'Messages: 3', 'User: [redacted]', and 'UTF-8'

Display 75 - Log for Converted Data Node for Tabbed Text File SAS Data Set Nodes in

SAS Studio

SAS Studio converts Data nodes for SAS data sets to SAS Program nodes. These nodes contain code that displays a view of the table.

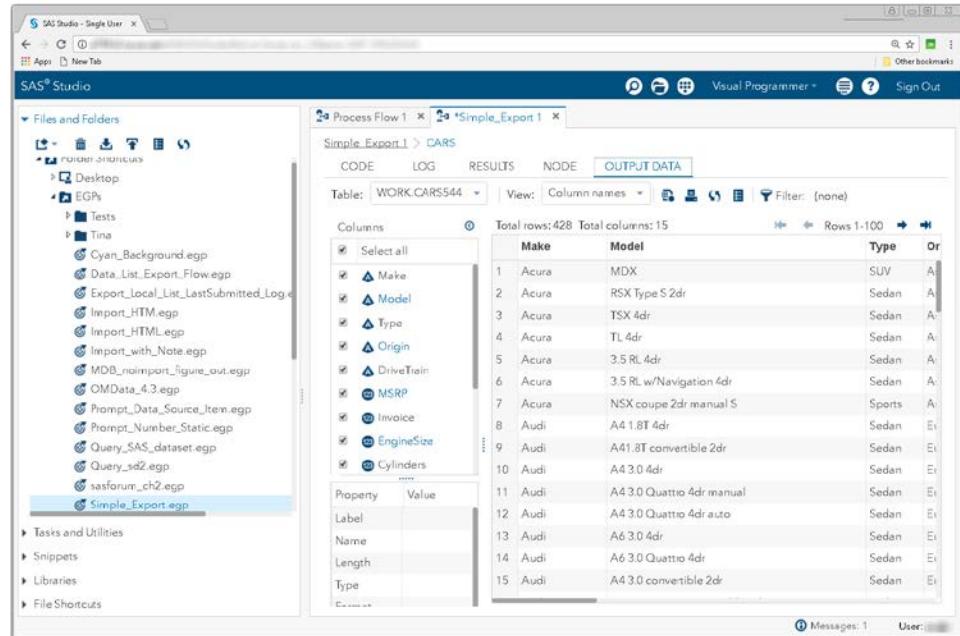


The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. On the left, the 'Files and Folders' sidebar lists various EGP files, including 'Simple_Export.egp' which is currently selected. The main area displays the code for a SAS program node named 'CARS'. The code is as follows:

```
/*
3 Code generated by SAS Studio.
4
5 Data Node Name: CARS
6 Dataset: SASHELP.CARS
7 Server: Local
8
9 IMPORTANT NOTE: Update this code to resolve the data source if necessary.
10 -----
11 -----
12 DATA WORK.CARS544 / VIEW=WORK.CARS544;
13   SET SASHELP.CARS;
14
15 RUN;
```

Display 76 - Code for Converted Data Node for SAS Data Set

When you run the process flow or the converted Data node for the SAS data set in the SAS Studio process flow, the **OUTPUT DATA** tab displays a view of the table. You cannot edit the table or perform any other functions in this view.



The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. The 'Files and Folders' sidebar shows 'Simple_Export.egp' is selected. The main area has the 'OUTPUT DATA' tab selected. It displays a table view of the 'CARS' dataset from 'SASHELP'. The table has columns: Make, Model, Type, and Type. The data is as follows:

Make	Model	Type	Type
Acura	MDX	SUV	A
Acura	RSX Type S 2dr	Sedan	A
Acura	TSX 4dr	Sedan	A
Acura	TL 4dr	Sedan	A
Acura	3.5 RL 4dr	Sedan	A
Acura	3.5 RL w/Navigation 4dr	Sedan	A
Acura	NSX coupe 2dr manual S	Sports	A
Audi	A4 1.8T 4dr	Sedan	E
Audi	A4 1.8T convertible 2dr	Sedan	E
Audi	A4 3.0 4dr	Sedan	E
Audi	A4 3.0 Quattro 4dr manual	Sedan	E
Audi	A4 3.0 Quattro 4dr auto	Sedan	E
Audi	A6 3.0 4dr	Sedan	E
Audi	A6 3.0 Quattro 4dr	Sedan	E
Audi	A4 3.0 convertible 2dr	Sedan	E

Display 77 - Output from Converted Data Node in SAS Studio

In SAS Enterprise Guide, you can also see the table contents of this node. However in SAS Enterprise Guide, you can perform many other functions including editing the contents of the table.

The screenshot shows the SAS Enterprise Guide interface with a data node for a CARS dataset. The Project Tree on the left shows a node named 'Export File [CARS]'. The main workspace displays a table with 27 rows and 11 columns. The columns are: Make, Model, Type, Origin, DriveTrain, MSRP, Invoice, EngineSize, Cylinders, Horsepower, and a row number column. The data includes various car models from brands like Acura, Audi, and BMW, along with their respective details such as engine size, cylinders, and horsepower. The bottom status bar indicates 'Ready'.

	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Horsepower
1	Acura	MDX	SUV	Asia	All	\$36,945	\$33,337	3.5	6	
2	Acura	RSX Type S 2dr	Sedan	Asia	Front	\$23,820	\$21,761	2	4	
3	Acura	TSX 4dr	Sedan	Asia	Front	\$26,990	\$24,647	2.4	4	
4	Acura	TL 4dr	Sedan	Asia	Front	\$33,195	\$30,299	3.2	6	
5	Acura	3.5 RL 4dr	Sedan	Asia	Front	\$43,755	\$39,014	3.5	6	
6	Acura	3.5 RL w/Navi...	Sedan	Asia	Front	\$46,100	\$41,100	3.5	6	
7	Acura	NSX coupe 2dr	Sports	Asia	Rear	\$89,765	\$79,978	3.2	6	
8	Audi	A4 1.8T 4dr	Sedan	Europe	Front	\$25,940	\$23,508	1.8	4	
9	Audi	A41.8T conv...	Sedan	Europe	Front	\$35,940	\$32,506	1.8	4	
10	Audi	A4 3.0 4dr	Sedan	Europe	Front	\$31,840	\$28,846	3	6	
11	Audi	A4 3.0 Quattro...	Sedan	Europe	All	\$33,430	\$30,366	3	6	
12	Audi	A4 3.0 Quattro ...	Sedan	Europe	All	\$34,480	\$31,388	3	6	
13	Audi	A6 3.0 4dr	Sedan	Europe	Front	\$36,640	\$33,129	3	6	
14	Audi	A6 3.0 Quattro...	Sedan	Europe	All	\$39,640	\$35,992	3	6	
15	Audi	A4 3.0 convert...	Sedan	Europe	Front	\$42,490	\$38,325	3	6	
16	Audi	A4 3.0 Quattro...	Sedan	Europe	All	\$44,240	\$40,075	3	6	
17	Audi	A6 2.7 Turbo ...	Sedan	Europe	All	\$42,840	\$38,840	2.7	6	
18	Audi	A6 4.2 Quattro...	Sedan	Europe	All	\$49,690	\$44,936	4.2	8	
19	Audi	A8 L Quattro 4dr	Sedan	Europe	All	\$69,190	\$64,740	4.2	8	
20	Audi	S4 Quattro 4dr	Sedan	Europe	All	\$48,040	\$43,556	4.2	8	
21	Audi	RS 6 4dr	Sports	Europe	Front	\$84,600	\$76,417	4.2	8	
22	Audi	TT 1.8 conver...	Sports	Europe	Front	\$35,940	\$32,512	1.8	4	
23	Audi	TT 1.8 Quattro...	Sports	Europe	All	\$37,390	\$33,881	1.8	4	
24	Audi	TT 3.2 coupe ...	Sports	Europe	All	\$40,590	\$36,739	3.2	6	
25	Audi	A6 3.0 Avant ...	Wagon	Europe	All	\$40,840	\$37,060	3	6	
26	Audi	S4 Avant Quat...	Wagon	Europe	All	\$49,090	\$44,446	4.2	8	
27	BMW	X3 3.0i	SUV	Europe	All	\$37,000	\$33,873	3	6	
	BMW	328i xDrive	Sedan	Europe	All	\$36,155	\$32,700	2.4	4	

Display 78 – Data Node for SAS Data Set in SAS Enterprise Guide

Data Nodes Not Sampled in SAS Studio

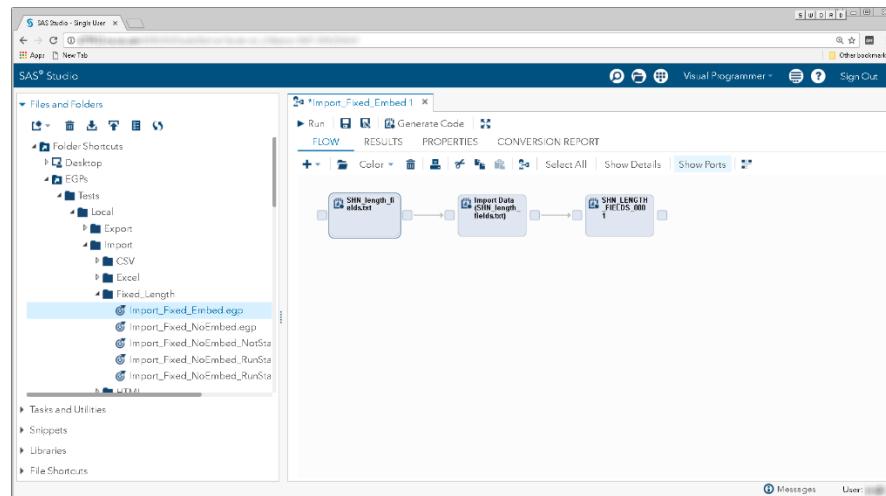
The contents of the Data node are not sampled in the converted node when:

- the EGP was configured to have data embedded in the generated SAS code. In this case, the data in the file is not used by the process flow. It is embedded in the task that uses the data.
- the data source is an Excel file. Excel data would have to be extracted to sample.

Data Is Embedded in the SAS Code

If you selected the **Embed the data within the generated SAS code** option, the converted node will contain just a comment.

The following converted process flow contains a Program File node that is used to identify data that is embedded into the Import Data node. As a result, the process flow in SAS Studio is visually similar to the one in SAS Enterprise Guide.



Display 79 - Contents of Source Data Are Embedded in Import Node

When you open the converted Data node, you see comments that provide the source location of the data that is embedded in the Import Data node. These comments also explain that updating the source data will not change the outcome of running the program.

```
/*
1 -----+
2
3 Code generated by SAS Studio.
4
5 "Embed the data within the generated SAS code" was set in the task
6 that imports this data, therefore changes made to this file since the
7 task was created will not be included in the imported contents.
8
9 Source File: C:\Public\SHN_length_fields.txt
10
11 */
```

Display 80 - Comments about the Embedded Data

If you open the Import Data node, you see the data from the input file in the DATALINES statement.

```

36   F1      $CHAR69.
37   F2      BEST4.
38   F3      BEST5.
39   F4      $CHAR22.
40   F5      $CHAR28.
41   F6      $CHAR25. ;
42 INFILE DATALINES4
43   DLN='7F'x
44   MISSOVER
45   DSD ;
46 INPUT
47   F1      :$CHAR69.
48   F2      :?? BEST4.
49   F3      :?? BEST5.
50   F4      :$CHAR22.
51   F5      :$CHAR28.
52   F6      :$CHAR25. ;
53 DATALINES4;
54 1407: HA/AA Dressage First Level AADTR Champ
55 1407: HA/AA Dressage First Level AADTR Champ
56 1408: HA/AA Dressage First Level ATR Champ
57 1408: HA/AA Dressage First level ATR Champ
58 1408: HA/AA Dressage Training Level AADTR Champ
59 1502: HA/AA Dressage Training Level AADTR Champ
60 1502: HA/AA Dressage Training Level AADTR Champ
61 1502: HA/AA Dressage Training Level AADTR Champ
62 1502: HA/AA Dressage Training Level AADTR Champ

```

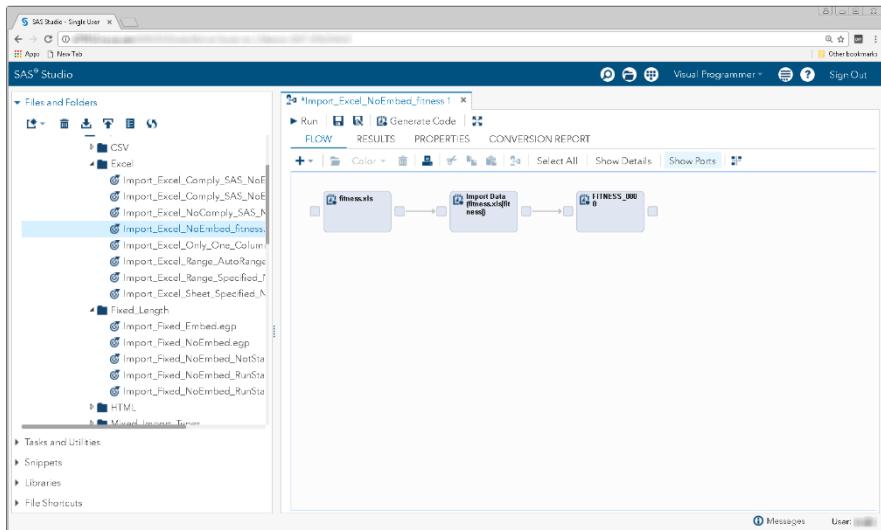
Line 53, Column 12 UTF-8

Display 81 - Data Embedded in Converted Import Node

Data Is from Excel

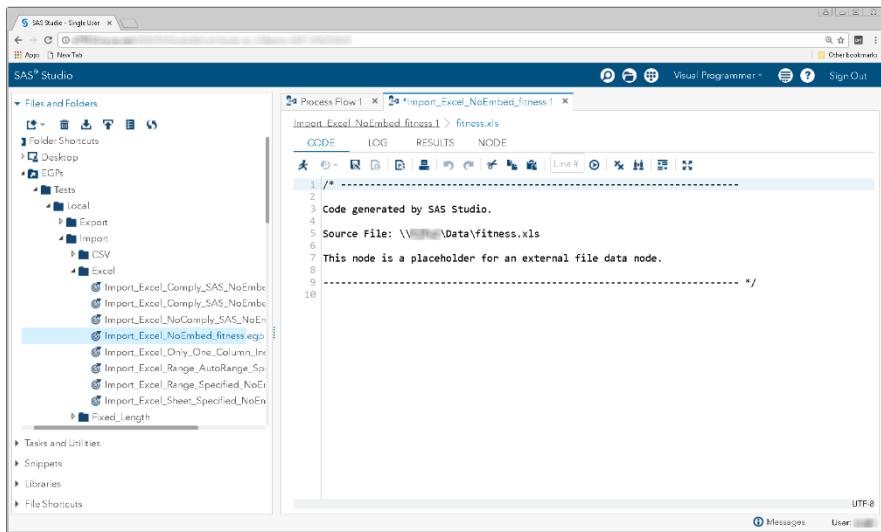
If the source for the Data node is an Excel file, the Data node is converted to a SAS Program node. The code in the converted SAS Program node in SAS Studio will not sample the contents of the Excel file because this sampling would require importing the file contents. The converted node contains only comments identifying the location of the Excel file. This node exists in the converted flow only to make the process flow look similar to how it looks in SAS Enterprise Guide.

In this flow, an Excel file is the data source for the Import Data node.



Display 82 - Data Node for Excel Data

The contents of the converted Data node for an Excel File is simply comments.



A screenshot of the SAS Studio interface. On the left, the 'File and Folders' sidebar shows a tree structure with 'Tests' expanded, containing 'local', 'Import', 'CSV', 'Excel' (which is selected), and 'Fixed Length'. The 'Excel' folder contains numerous sub-nodes related to importing Excel files. On the right, the main workspace displays a code editor titled 'Process Flow | Import_NoEmbed_fitness1 > fitness.xls'. The tab bar at the top of the editor shows 'CODE', 'LOG', 'RESULTS', and 'NODE'. The code itself is a single block of comments:

```
1 /*-----  
2  Code generated by SAS Studio.  
3  
4  Source File: \\Data\\fitness.xls  
5  
6  This node is a placeholder for an external file data node.  
7  
8 -----*/  
10
```

Display 83 - Converted Data Node When Input Data Is an Excel File

SAS ENTERPRISE GUIDE TASKS

Since SAS Enterprise Guide tasks do not port directly to SAS Studio, code generated by a SAS Enterprise Guide Task node is extracted and placed in a SAS Program node in SAS Studio.

The SAS Enterprise Guide Task nodes are associated with a particular SAS connection. If the SAS Studio connection does not match the connection associated with the Task node in SAS Enterprise Guide, an error appears in the log.

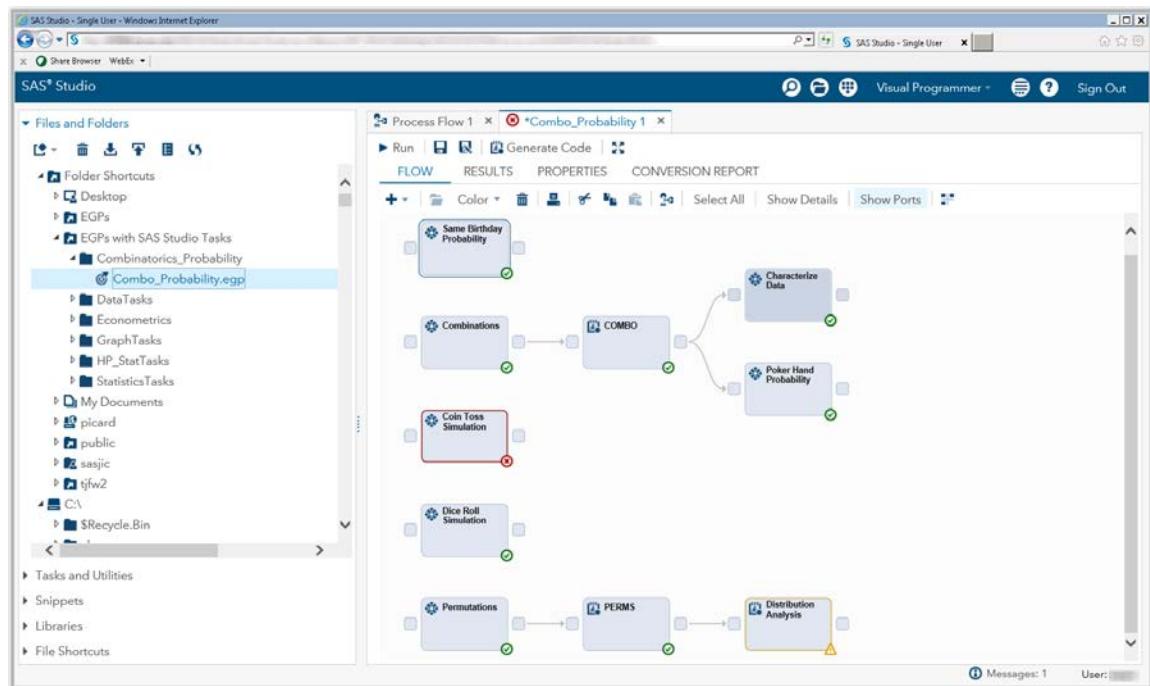
QUERY BUILDER NODES

Since SAS Enterprise Guide Query Builder nodes do not port directly to SAS Studio, code generated by a SAS Enterprise Guide Query Builder node is extracted and placed in a SAS Program node in SAS Studio.

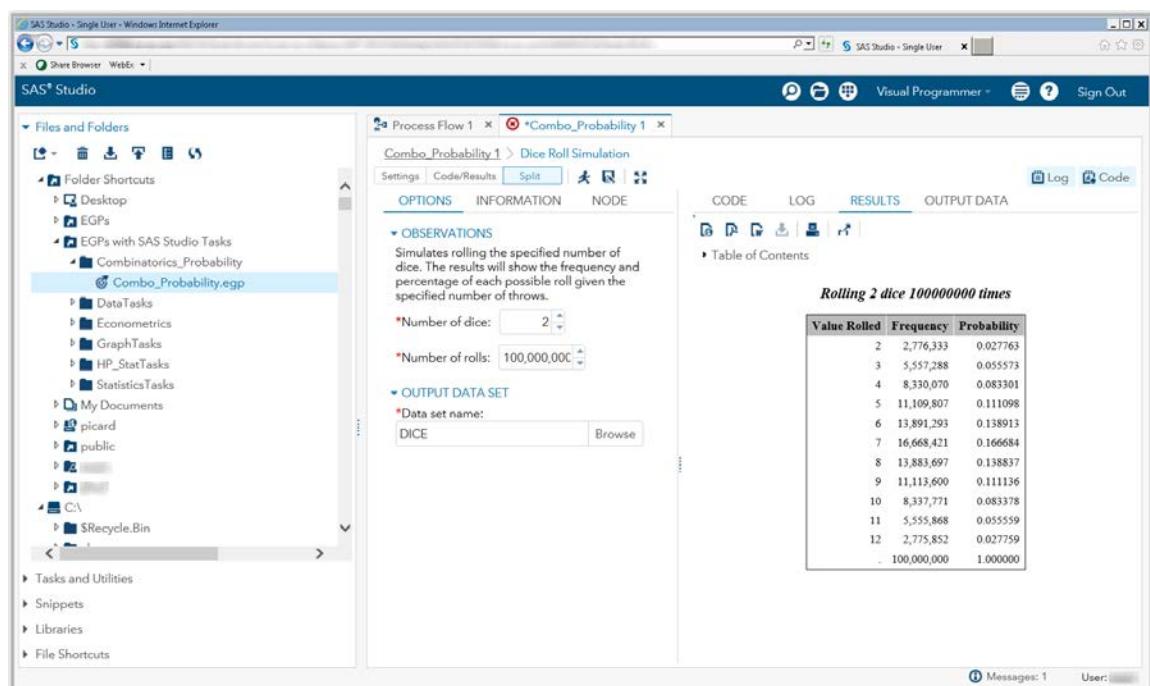
SAS STUDIO TASKS

SAS Studio tasks are also available in SAS Enterprise Guide. SAS Studio Task nodes convert seamlessly. Any parameters set on a SAS Studio Task node will be in the converted node.

Again, server context is very important since the code for the SAS Studio tasks are version dependent.



Display 84 - Converted Process Flow That Contains SAS Studio Tasks



Display 85 – User Interface and Results for SAS Studio Task Node

PROMPTS

SAS Enterprise Guide

In SAS Enterprise Guide when a process flow containing a program with prompts or a prompt dependent program is run, a dialog box that contains prompts is displayed.

When the user enters values in the prompt dialog box, these values are placed in macro variables with names based on the **Name** field in the prompt definition. Some prompts, such as a simple text or numeric input field, create a single macro variable with the prompt name. Other prompts such as a text range or a numeric range create multiple macro variables, such as Name_min and Name_max. Multiple nodes can depend on a prompt.

SAS Studio

SAS Studio does not support prompts. When an EGP that contains prompts is converted, code is added to the beginning of the converted Program nodes that have prompts associated with them. The added code will:

- contain comments about the values expected for the macro variables, such as types, minimums and maximums, and so on.
- define macro variables that the prompts would have created.
- set the macro variables to:
 - default values if the prompts have defaults
 - blank or empty if prompts do not have defaults
- delete the macro variables at the end of the program unless the prompt definition has the **Use prompt value throughout project** option selected. (You select this check box in the Prompt Manager in SAS Enterprise Guide.)

If you want to run your prompt dependent code against different parameters, you must manually change the values in the generated macro variables.

If a prompt is used for more than one node, the values of the macro variables would have to be manually changed in each node to simulate the prompt. This editing can be very tedious if there are many nodes depending on a single prompt.

If you would prefer to change the prompt values with a user interface, you can use a [SAS Studio task as a prompt replacement](#).

SAS Studio Task as Prompt Replacement

You can manually update a converted process flow to use a SAS Studio task to collect the information that the prompts would have collected. To do this, you need to:

- write a SAS Studio task that contains input controls for each of the macro variables that the prompts would have created. The prompt replacement code generated by SAS Studio provides information about the macro variables and the attributes of the macro variables that are defined by the prompts.
- add the SAS Studio task to your converted process flow.
- create links from the output port of the SAS Studio task to the input ports of the SAS Program nodes that depend on the prompt or prompts.
- in the converted Program nodes that depend on the prompts, comment out the call to the macro code that creates and sets global variables. The %SYMDEL code in single input cases can remain.

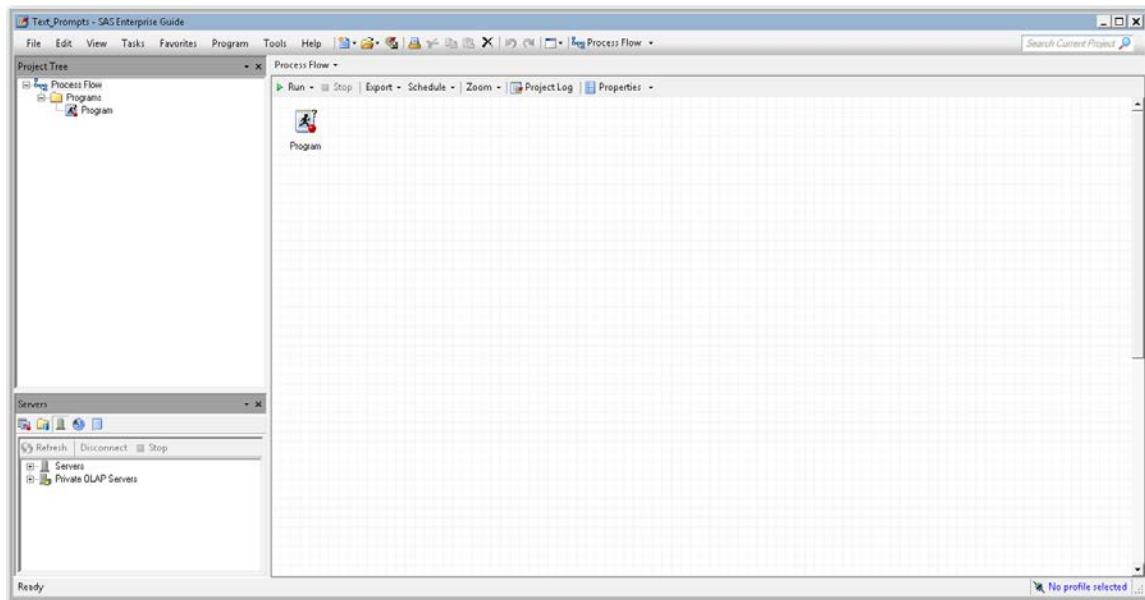
When using a SAS Studio task to simulate a SAS Enterprise Guide prompt, the task functions like a SAS Enterprise Guide prompt that has **Hide at Runtime** option selected. If you would like to run with different values, you must open the SAS Studio task, change the values, and then run the process flow or program.

For more information about how to write a task, see [*SAS Studio 3.6: Developer's Guide to Writing Custom Tasks*](#). A good starting point for writing a task is to view the Sample Task in SAS Studio. From the **Tasks and Utilities** section, click the New icon and select **Sample Task**. The Sample Task opens in a Task Editor window. Press the **Run** button to see the controls that the task creates. Then you can use the **New Task** option to create a new task and add the controls that you need for requesting the prompt input. Use the Sample Task as a resource.

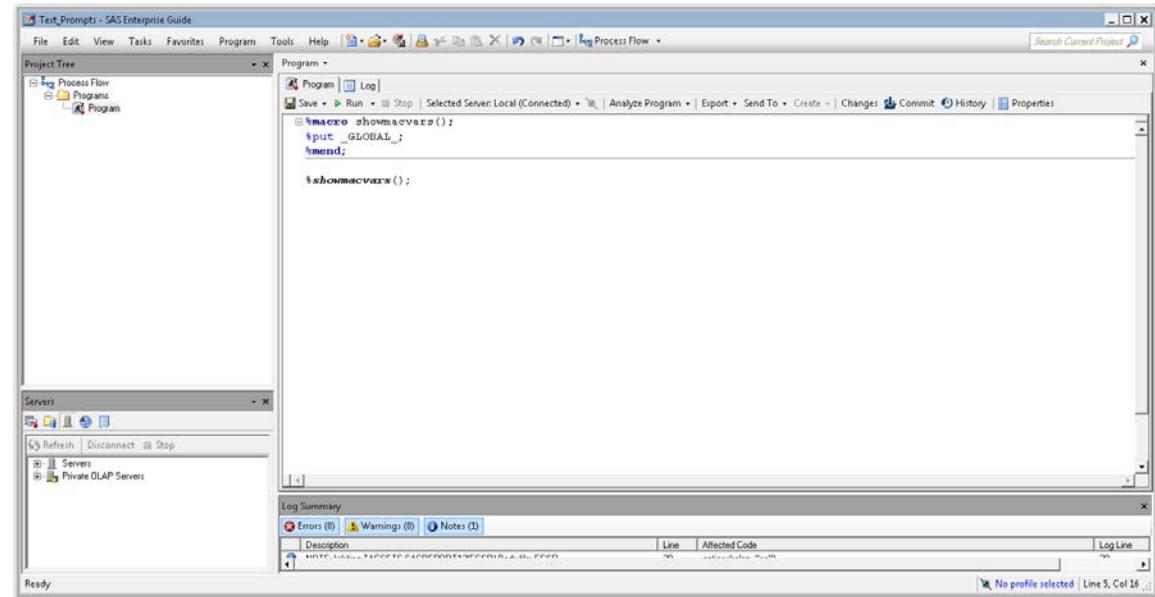
The following sections provide examples of the types of prompts supported by SAS Enterprise Guide and describe the macro variables that are generated for those types. Then the same examples are shown converted in SAS Studio with information about the generated macro code for prompt replacement in the SAS Program node. For many of the prompt types, there are explanations on how to use SAS Studio tasks to replace the prompts. Not all of the range and multiple selection prompt types have an example, but you can get the idea about how to code ranges for those types from the other range and multiple selection examples. All of the examples can be further enhanced to be more complete. For example, the range tasks could have error checks for minimums that are greater than maximums.

Program Using Sample Prompt Definitions

The following program is used for all of the prompt examples. This program displays the contents of the global variables.



Display 86 - Program Node in SAS Enterprise Guide



Display 87 – Code in the Program Node in SAS Enterprise Guide

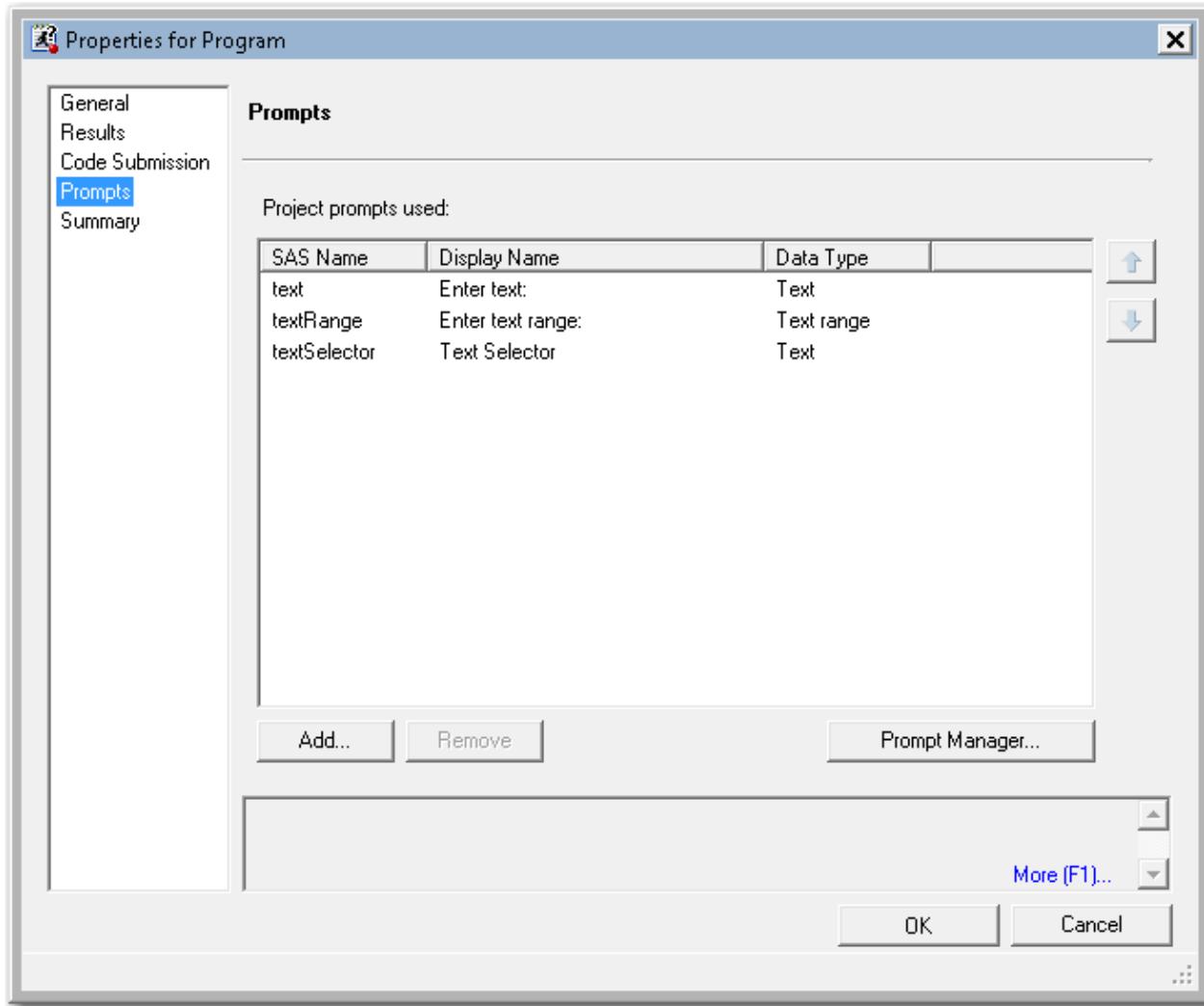
In SAS Enterprise Guide to define a prompt, right-click the Program node and select **Properties**. From the selection pane, select **Prompt** and click **Prompt Manager**. You define prompts in the Prompt Manager.

After you have defined your prompts, you can select the prompts to add to your Program nodes by clicking the **Add** button.

Text

The name of the macro variable or variables defined for a text prompt depends on whether it is a [single value](#), a [text range](#), or [multiple text](#) value prompt.

The following example contains the three different types of text prompts and describes the macro variables that are defined by SAS Enterprise Guide for each prompt type.

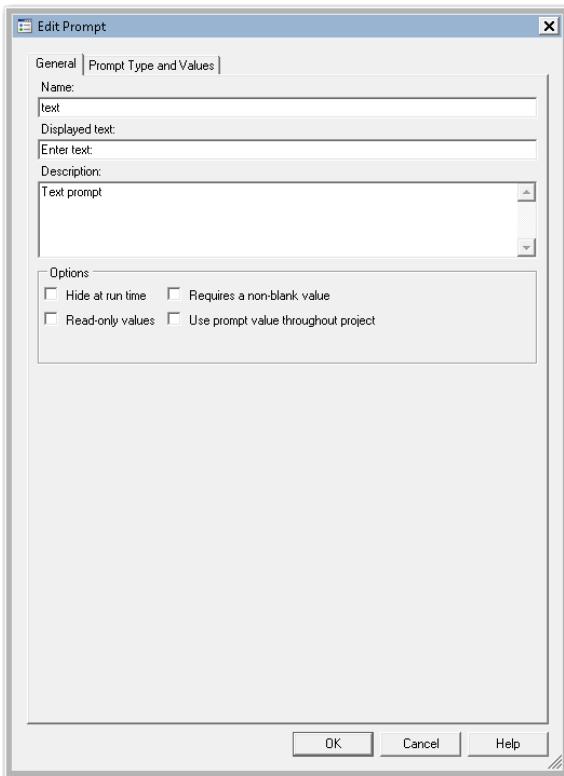


Display 88 - Text Prompts Defined in SAS Enterprise Guide

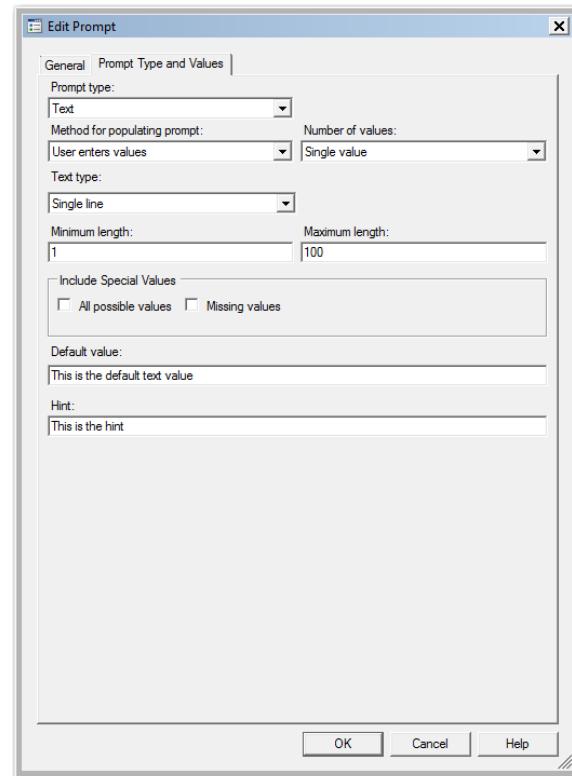
Single Value Text

SAS Enterprise Guide

In this example, a single value text prompt named **text** is defined.

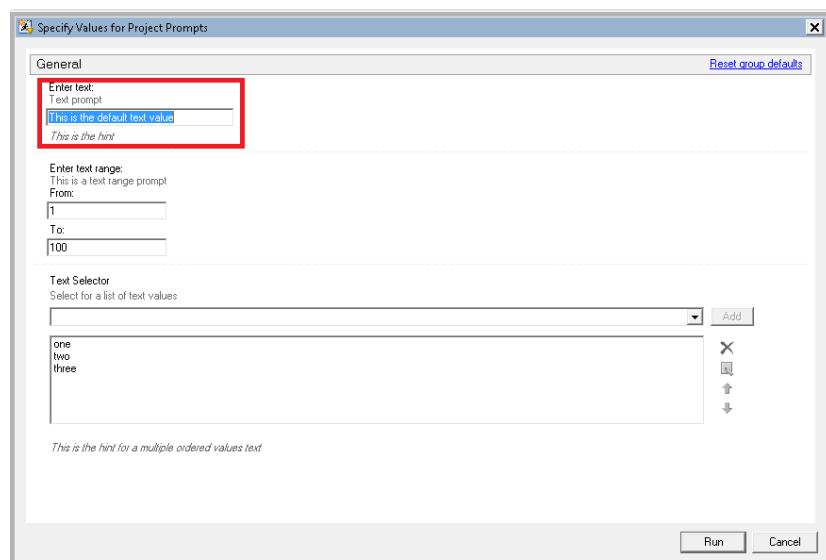


Display 89 - General Attributes for Text Prompt



Display 90 - Prompt Type and Values for Text Prompt

When you run the Program node that depends on this prompt, the following dialog box appears.



Display 91 - Single Text Value Prompt in Prompt Dialog Box

If the user leaves the default value in the single text value prompt field, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

A %LET statement assigns the value specified in the prompt dialog box to the text macro variable.

```

Text_Prompts - SAS Enterprise Guide
File Edit View Tasks Favorites Program Tools Help Process Flow Log Project Log Properties
Project Tree
Process Flow Programs Program
1 ;/*;quit;run;
2 OPTIONS PAGENO=MIN;
3 $LET _CLIENTTASKLABEL='Program';
4 $LET _CLIENTPROJECTPATH='C:\EGFs\Tests\Local\Prompts\Text\Text_Prompts.egp';
5 $LET _CLIENTPROJECTNAME='Text_Prompts.egp';
6 $LET _SASPROGRAMFILE='';
7 $LET text = $nrstr(This is the default text value);
8 $LET textRange_min = $nrstr(1);
9 $LET textRange_max = $nrstr(100);
10 $LET textSelector = $nrstr(one);
11 $LET textSelector_count = $nrstr(3);
12 $LET textSelector1 = $nrstr(one);
13 $LET textSelector0 = $nrstr(3);
14 $LET textSelector3 = $nrstr(three);
15 $LET textSelector2 = $nrstr(two);
16
17 ODS _ALL_ CLOSE;
18 OPTIONS DEV=ACTIVEBX;
19 GOPTIONS XPIXELS=0 YPIXELS=0;
20 FILENAME EGSR TEMP;
21 ODS tagsets.sasreport3(ID=EGSR) FILE=EGSR
22   STYLE=HtmlBlue
23   STYLESHEET=(URL="file:///C:/Program%20Files/SASHome/SASEnterpriseGuide/7.1/Styles/HtmlBlue.css")
24 NOGTITLE
-----
```

Log Summary

Description	Line	Affected Code	Log Line
No profile selected	Line 85, Col 1		

Display 92 - Assignment of %LET Statements for Single Value Text Prompt

The log of the [Program node using the prompt definition](#) displays the value of the global variable created by the prompt.

```

Text_Prompts - SAS Enterprise Guide
File Edit View Tasks Favorites Program Tools Help Process Flow Log Project Log Properties
Project Tree
Process Flow Programs Program
34 $mend;
35
36 $showmacvars();
GLOBAL _SASHWORKLOCATION "C:\Users\annchen\AnnData\Local\Temp\SEG10636\SAS Temporary Files\_TD12080_D79910_\Prc2/"
GLOBAL TEXT 'This is the default text value'.
GLOBAL TEXTRANGE_MAX '100';
GLOBAL TEXTRANGE_MIN '1';
GLOBAL TEXTSELECTOR 'one';
GLOBAL TEXTSELECTOR0 '3';
GLOBAL TEXTSELECTOR1 'one';
GLOBAL TEXTSELECTOR2 'two';
GLOBAL TEXTSELECTOR3 'three';
GLOBAL TEXTSELECTOR_COUNT '3';
GLOBAL _CLIENTAPP 'SAS Enterprise Guide';
GLOBAL _CLIENTAPPABREV EG;
GLOBAL _CLIENTMACHINE 'annchen';
GLOBAL _CLIENTPROJECTNAME 'Text_Prompts.egp';
GLOBAL _CLIENTPROJECTPATH 'C:\EGFs\Tests\Local\Prompts\Text\Text_Prompts.egp';
GLOBAL _CLIENTTASKLABEL 'Program';
GLOBAL _CLIENTUSERID 'ssjic';
GLOBAL _CLIENTUSERNAME 'J. Jeffreys-Chen';
GLOBAL _CLIENTVERSION '7.100.1.2651';
GLOBAL _INITIALIZEDSERVERID 1
-----
```

The SAS System 09:12 Thursday, July 10, 2008

Log Summary

Description	Line	Affected Code	Log Line
No profile selected	Line 85, Col 1		

Display 93 - Global Definition for Text Macro Variable

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statement removes the text macro variable at the end of the program.

The screenshot shows the SAS Enterprise Guide interface with the following components:

- Project Tree:** Shows a single node named "Program".
- Servers:** Shows a list of servers, including "Servers" and "Private OLAP Servers".
- Program Editor:** Displays a SAS program with several global statements at the top. Line 43 contains the statement "%SYMDEL text;". This line is highlighted with a red rectangular box.
- Log Summary:** A panel below the program editor showing log information. It includes tabs for Errors (0), Warnings (0), and Notes (1). The Notes tab is selected, displaying one note related to the highlighted code.

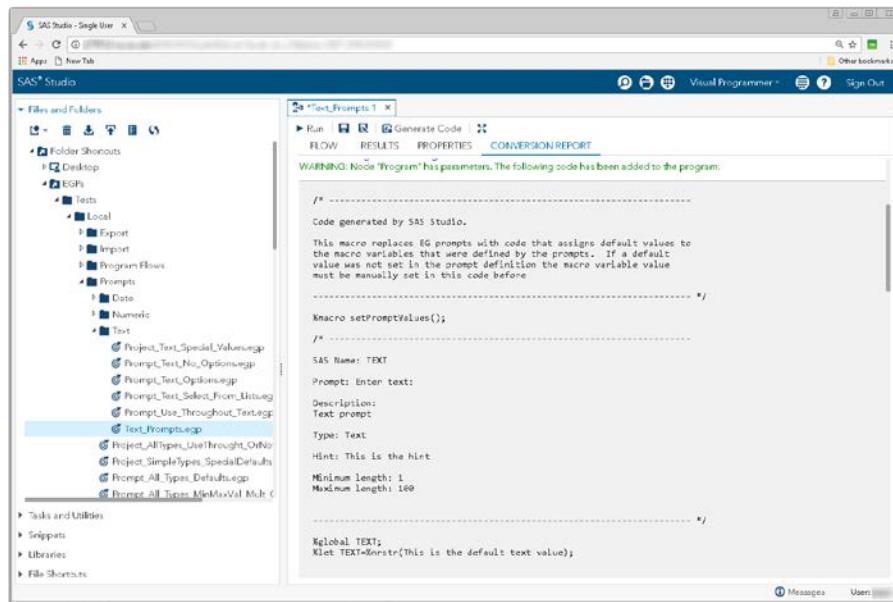
```
GLOBAL _SASHOSTNAME '*****';
GLOBAL _SASPROGRAMFILE;
GLOBAL _SASERVERNAME 'Local';
37
38      GOPTIONS NOACCESSIBLE;
39      %LET _CLIENTTASKLABEL="";
40      %LET _CLIENTPROJECTPATH="";
41      %LET _CLIENTPROJECTNAME="";
42      %LET _SASPROGRAMFILE="";
43      %SYMDEL text;
44      %SYMDEL textRange_max;
45      %SYMDEL textRange_min;
46      %SYMDEL textSelector;
47      %SYMDEL textSelector_count;
48      %SYMDEL textSelector0;
49      %SYMDEL textSelector3;
50      %SYMDEL textSelector2;
51
52      ;*';*/;quit;run;
53
54      ODS _ALL_ CLOSE;
55
56
```

Display 94 - %SYMDEL Statement Removes Text Macro Variable

SAS Studio

The following display shows the code that is added to the converted Program node for the text prompt in SAS Enterprise Guide.

A global variable named TEXT is created and a %LET statement assigns the default value to TEXT. If you want to run the process flow using a different value for the TEXT prompt, you must manually update the value of the macro variable in the %LET statement.



The screenshot shows the SAS Studio interface with a project tree on the left and a code editor on the right. The code editor displays the following macro code:

```
/*
Code generated by SAS Studio.

This macro replaces EG prompts with code that assigns default values to
macro variables. It also creates a global variable for each prompt. If a default
value was not set in the prompt definition the macro variable value
must be manually set in this code before

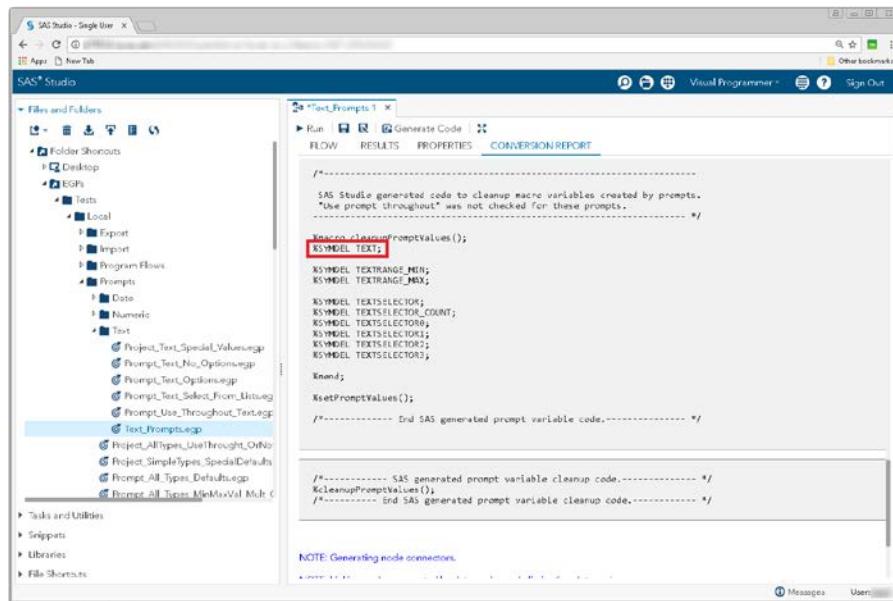
*/
Kmacro setPromptValues();

/*
SAS Name: TEXT
Prompt: Enter text:
Description: Text prompt
Type: Text
Hint: This is the hint
Minimum length: 1
Maximum length: 100

*/
Kglobal TEXT;
Klet TEXT=macrostr(This is the default text value);
```

Display 95 - Generated Macro Code for Single Value Text Prompt

Because the **Use prompt value throughout project** option is not checked, the %SYMDEL statement removes the TEXT macro variable.



The screenshot shows the SAS Studio interface with a project tree on the left and a code editor on the right. The code editor displays the following macro code:

```
/*
SAS Studio generated code to cleanup macro variables created by prompts.
*Use prompt throughout* was not checked for these prompts.
----- */

Kmacro cleanupPromptValues();
%SYMDEL TEXT;

KSYMDEL TEXT RANGE_MIN;
KSYMDEL TEXT RANGE_MAX;

KSYMDEL TEXT SELECTOR1;
KSYMDEL TEXT SELECTOR COUNT;
KSYMDEL TEXT SELECTOR0;
KSYMDEL TEXT SELECTOR1;
KSYMDEL TEXT SELECTOR0;
KSYMDEL TEXT SELECTOR3;
KSYMDEL TEXT SELECTOR2;

Kmacro;
KsetPromptValues();

/*----- End SAS generated prompt variable code.----- */

/*----- SAS generated prompt variable cleanup code.----- */
%cleanupPromptValues();
/*----- End SAS generated prompt variable cleanup code.----- */

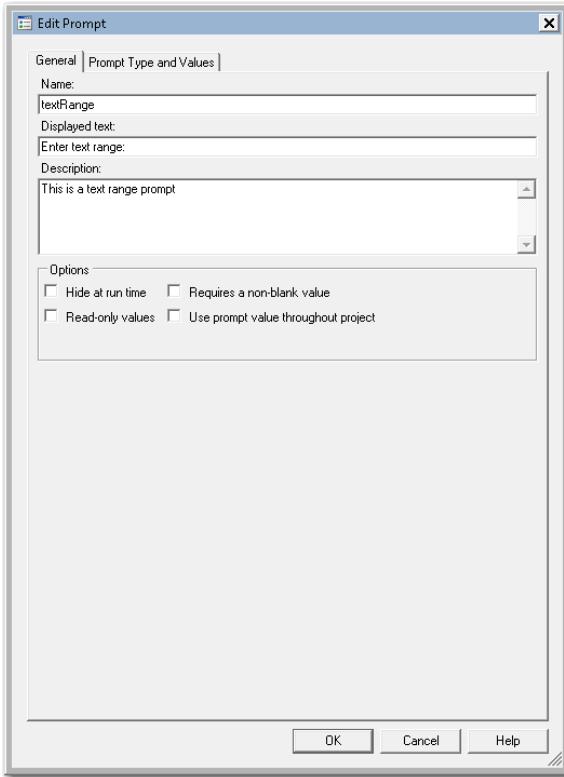
NOTE: Generating node connectors.
```

Display 96 - %SYMDEL Statement Removes TEXT Macro Variable

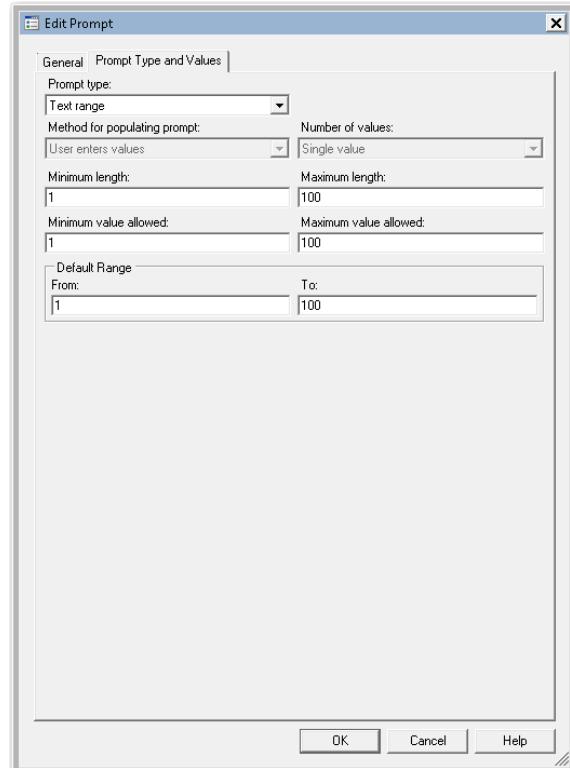
Text Range

SAS Enterprise Guide

In this example, a text range prompt named `textRange` is defined as shown in the following two displays.

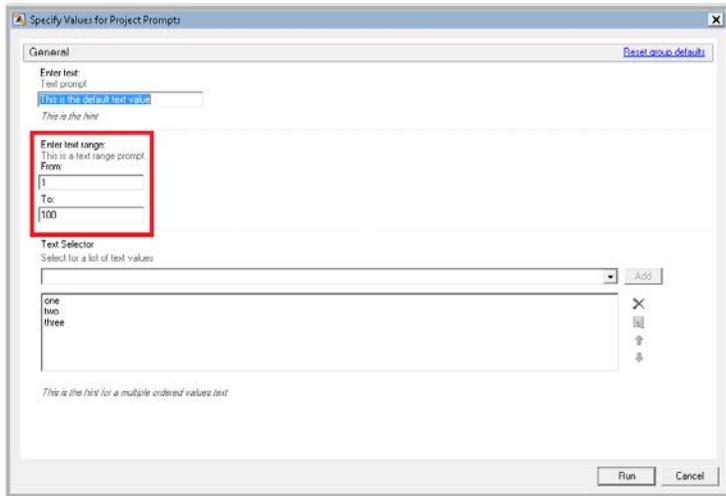


Display 97 - General Properties for Text Range Prompt



Display 98 - Type and Values for Text Range Prompt

When you run the Program node that depends on this prompt, the following dialog box appears.



Display 99 - Text Range Prompt in Prompt Dialog Box

If the user leaves the default values in the text range prompt fields, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt. The %LET statements assign the values specified in the prompt dialog box to the textRange_min and textRange_max macro variables.

```

1  ; */ *;*/;quit;run;
2  OPTIONS PAGE=MIN;
3  %LET _CLIENTTASKLABEL='program';
4  %LET _CLIENTPROJECTPATH='C:\BGGS\Tests\Local\Prompts\Text\Text_Prompts.egp';
5  %LET _CLIENTPROJECTNAME='Text_Prompts.egp';
6  %LET _SASPROGRAMFILE='';
7  %LET text = %nrstr(This is the default text value);
8  %LET textRange_min = %nrstr(1);
9  %LET textRange_max = %nrstr(100);
10 %LET textSelector = %nrstr(one);
11 %LET textSelector_count = %nrstr(3);
12 %LET textSelector0 = %nrstr(one);
13 %LET textSelector1 = %nrstr(3);
14 %LET textSelector2 = %nrstr(two);
15 %LET textSelector3 = %nrstr(three);
16
17 ODS _ALL_ CLOSE;
18 OPTIONS DEV=ACTIVEV;
19 GOPTIONS XPIXELS=0 YPIXELS=0;
20 FILENAME EGSR TEMP;
21 ODS tagsets.sasreport13(ID=EGSR) FILE=EGSR
22   STYLE=HTMLBlue
23   STYLESHEET=(URL="file:///C:/Program%20Files/SASHome/SASEnterpriseGuide/7.1/styles/HtmlBlue.css")
24 NOGTTITLE
25 -----
26
27
28
29
30
31
32
33
34 %mend;
35
36 %showmacvars();
GLOBAL SASWORKLOCATION "C:\Users\[REDACTED]\AppData\Local\Temp\SEG10636\SAS Temporary Files\[REDACTED]\Prc2\"
GLOBAL TEXT 'This is the default text value';
GLOBAL TEXTRANGE_MAX '100';
GLOBAL TEXTRANGE_MIN '1';
GLOBAL TEXTSELECTOR 'one';
GLOBAL TEXTSELECTOR0 '3';
GLOBAL TEXTSELECTOR1 'one';
GLOBAL TEXTSELECTOR2 'two';
GLOBAL TEXTSELECTOR3 'three';
GLOBAL TEXTSELECTOR_COUNT '3';
GLOBAL _CLIENTAPP 'SAS Enterprise Guide';
GLOBAL _CLIENTAPPBREV EG;
GLOBAL _CLIENTMACHINE '[REDACTED]';
GLOBAL _CLIENTPROJECTNAME 'Text_Prompts.egp';
GLOBAL _CLIENTPROJECTPATH 'C:\BGGS\Tests\Local\Prompts\Text\Text_Prompts.egp';
GLOBAL _CLIENTTASKLABEL 'Program';
GLOBAL _CLIENTUSERID '[REDACTED]';
GLOBAL _CLIENTUSERNAME 'J. Jeffreys-Chen';
GLOBAL _CLIENTVERSION '7.100.1.2651';
GLOBAL _INITIALIZEDSERVERID 1

```

Description	Line	Affected Code	Log Line
NOTE: XML_TACSETS.CACROSSTIMECODED.J-B_EGSR	88	%nrstr(1)	88

Display 100 - %LET Statements for textRange_min and textRange_max Macro Variables

The log of the [Program node using the prompt definition](#) displays the value of the global variables created by the prompt.

```

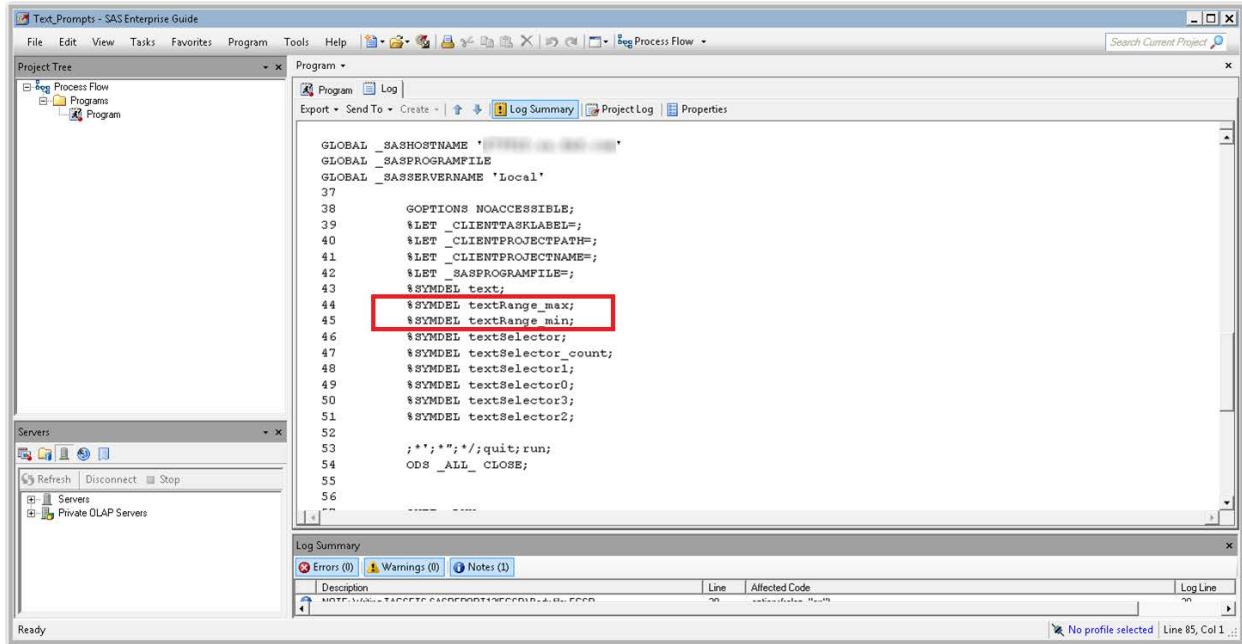
34 %mend;
35
36 %showmacvars();
GLOBAL SASWORKLOCATION "C:\Users\[REDACTED]\AppData\Local\Temp\SEG10636\SAS Temporary Files\[REDACTED]\Prc2\"
GLOBAL TEXT 'This is the default text value';
GLOBAL TEXTRANGE_MAX '100';
GLOBAL TEXTRANGE_MIN '1';
GLOBAL TEXTSELECTOR 'one';
GLOBAL TEXTSELECTOR0 '3';
GLOBAL TEXTSELECTOR1 'one';
GLOBAL TEXTSELECTOR2 'two';
GLOBAL TEXTSELECTOR3 'three';
GLOBAL TEXTSELECTOR_COUNT '3';
GLOBAL _CLIENTAPP 'SAS Enterprise Guide';
GLOBAL _CLIENTAPPBREV EG;
GLOBAL _CLIENTMACHINE '[REDACTED]';
GLOBAL _CLIENTPROJECTNAME 'Text_Prompts.egp';
GLOBAL _CLIENTPROJECTPATH 'C:\BGGS\Tests\Local\Prompts\Text\Text_Prompts.egp';
GLOBAL _CLIENTTASKLABEL 'Program';
GLOBAL _CLIENTUSERID '[REDACTED]';
GLOBAL _CLIENTUSERNAME 'J. Jeffreys-Chen';
GLOBAL _CLIENTVERSION '7.100.1.2651';
GLOBAL _INITIALIZEDSERVERID 1

```

Description	Line	Affected Code	Log Line
NOTE: XML_TACSETS.CACROSSTIMECODED.J-B_EGSR	88	%nrstr(1)	88

Display 101 – Values for Global Variables for Text Range Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.



The screenshot shows the SAS Enterprise Guide interface. The main window displays a SAS program code. Lines 44 and 45 contain the %SYMDEL statements, which are highlighted with a red rectangle. The code also includes global assignments and various %LET statements. The bottom right corner of the screen shows the log summary with no errors or warnings.

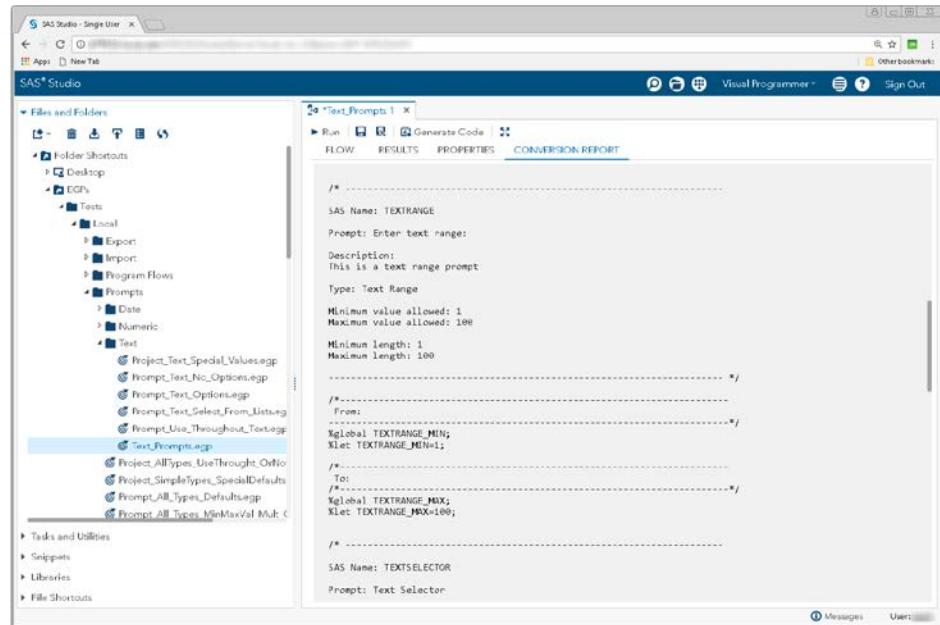
```
GLOBAL _SASHOSTNAME '';
GLOBAL _SASPROGRAMFILE '';
GLOBAL _SASSERVERNAME 'Local';
37
38      GOPTIONS NOACCESSIBLE;
39      %LET _CLIENTTASKLABEL="";
40      %LET _CLIENTPROJECTPATH="";
41      %LET _CLIENTPROJECTNAME="";
42      %LET _SASPROGRAMFILE="";
43      %SYMDEL text;
44      %SYMDEL textRange_max;
45      %SYMDEL textRange_min;
46      %SYMDEL textSelector;
47      %SYMDEL textSelector_count;
48      %SYMDEL textSelector1;
49      %SYMDEL textSelector0;
50      %SYMDEL textSelector3;
51      %SYMDEL textSelector2;
52
53      ; *; */;quit;run;
54      ODS _ALL_ CLOSE;
55
56      -----
```

Display 102 - %SYMDEL Statements Remove the textRange_max and textRange_min Macro Variables

SAS Studio

The following display shows the code that is added to the converted Program node for the text range prompt in SAS Enterprise Guide.

Global variables named TEXTRANGE_MIN and TEXTRANGE_MAX are created. The %LET statements assign the default values to the TEXTRANGE_MIN and TEXTRANGE_MAX variables. If you want to run your process flow using different values for the TEXTRANGE prompt, you must manually update the values of the macro variables in the %LET statements.



The screenshot shows the SAS Studio interface with the 'Text_Prompts.1' project open. The left sidebar shows a file tree with various EG files, including 'Text_Prompt.egp'. The main pane displays the generated macro code:

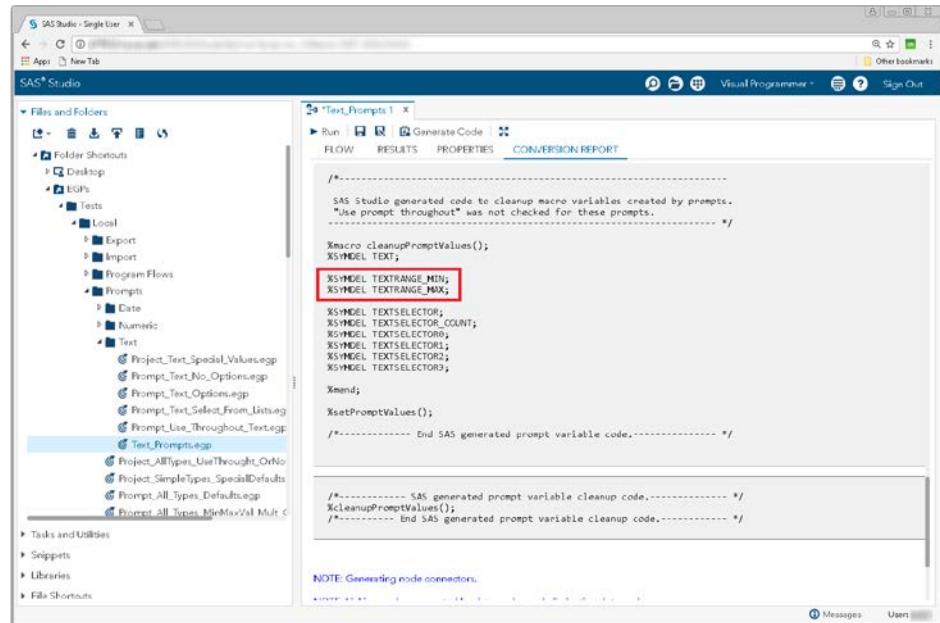
```
/*
SAS Name: TEXTRANGE
Prompt: Enter text range:
Description:
This is a text range prompt
Type: Text Range
Minimum value allowed: 1
Maximum value allowed: 100
Minimum length: 1
Maximum length: 100
*/
From:
Global TEXTRANGE_MIN;
Xlet TEXTRANGE_MIN=1;
/
To:
Global TEXTRANGE_MAX;
Xlet TEXTRANGE_MAX=100;

/*
SAS Name: TEXTSELECTOR
Prompt: Text Selector
*/


```

Display 103 - Generated Macro Code for Text Range Prompt

Because the **Use prompt value throughout project** option is not checked, the %SYMDEL statements remove the TEXTRANGE* macro variables.



The screenshot shows the SAS Studio interface with the 'Text_Prompts.1' project open. The left sidebar shows a file tree with various EG files, including 'Text_Prompt.egp'. The main pane displays the generated macro code, which includes %SYMDEL statements to remove the TEXTRANGE* macro variables:

```
/*
SAS Studio generated code to cleanup macro variables created by prompts.
*Use prompt throughout* was not checked for these prompts.
*/
Macro cleanupPromptValues();
%SYMDEL TEXT;
%SYMDEL TEXTRANGE_MIN;
%SYMDEL TEXTRANGE_MAX;

%SYMDEL TEXTSELECTOR;
%SYMDEL TEXTSELECTOR_COUNT;
%SYMDEL TEXTSELECTOR0;
%SYMDEL TEXTSELECTOR1;
%SYMDEL TEXTSELECTOR2;
%SYMDEL TEXTSELECTOR3;

Xmend;

XsetPromptValues();

/*
----- End SAS generated prompt variable code.-----
----- SAS generated prompt variable cleanup code.-----
%cleanupPromptValues();
----- End SAS generated prompt variable cleanup code.-----
*/

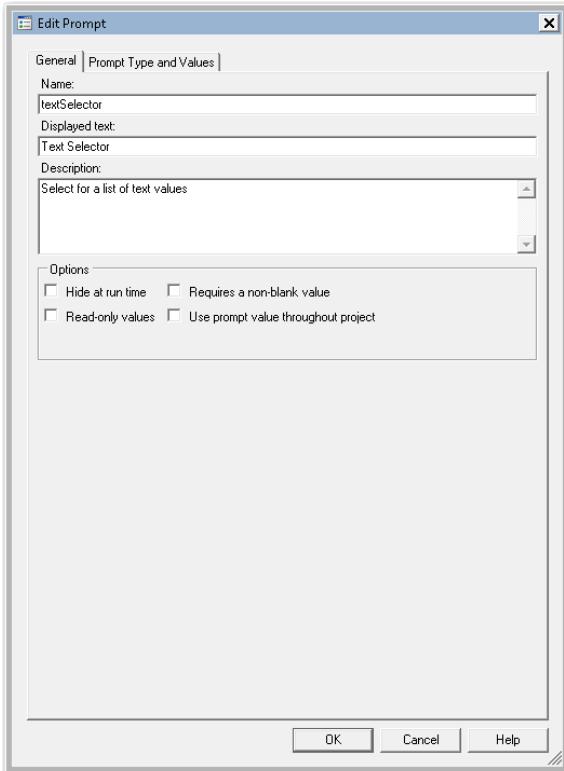
NOTE: Generating node connectors.
```

Display 104 - %SYMDEL Statements Remove the TEXTRANGE* Macro Variables

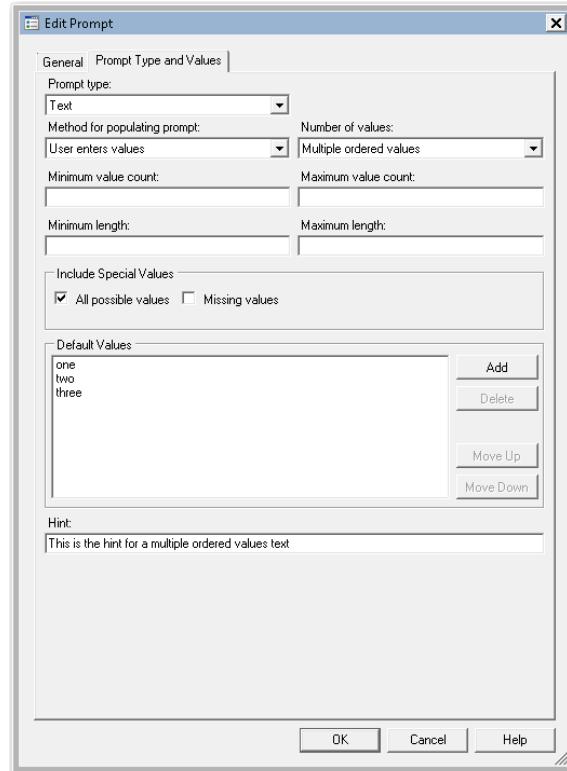
Multiple Text Values

SAS Enterprise Guide

In this example, a multiple ordered values text prompt named textSelector is defined as shown in the following two displays.

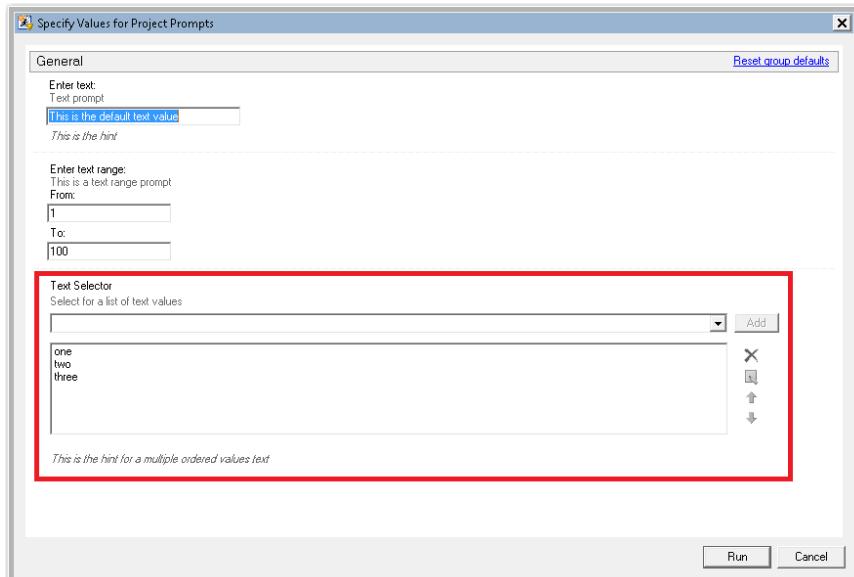


Display 105 - General Properties for Multiple Text Values Prompt



Display 106 - Type and Values for Multiple Text Values Prompt

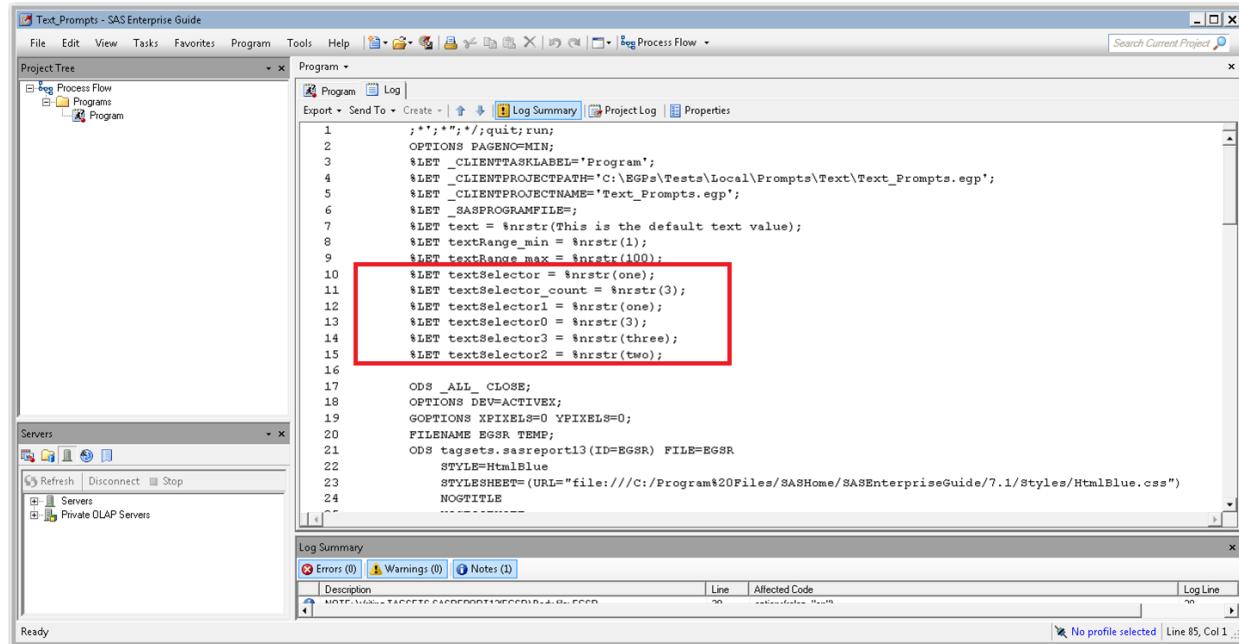
When you run the Program node that depends on this prompt, the following dialog box appears.



Display 107 - Multiple Text Values Prompt in Prompt Dialog Box

If the user leaves the default values for the multiple text values prompt, the `textSelector*` macro variables are generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The `%LET` statements assign the values specified in the prompt dialog box to the `textSelector*` macro variables.



The screenshot shows the SAS Enterprise Guide interface with the project tree containing a 'Program' node under 'Text_Prompts'. The main window displays the following code:

```

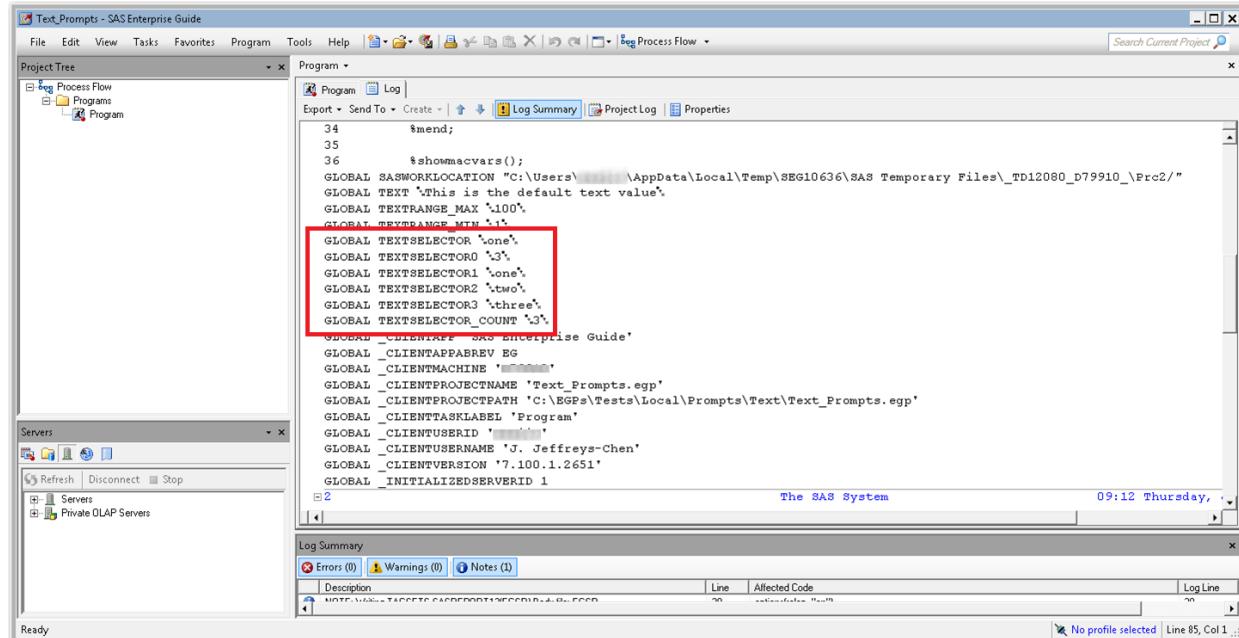
1 ;***;/quit;run;
2 OPTIONS PAGENO=MIN;
3 $LET _CLIENTTASKLABEL="Program";
4 $LET _CLIENTPROJECTPATH="C:\EGFs\Tests\Local\Prompts\Text\Text_Prompts.egp";
5 $LET _CLIENTPROJECTNAME="Text_Prompts.egp";
6 $LET _SASPROGRAMFILE="";
7 $LET text = $nrstr("This is the default text value");
8 $LET textRange_min = $nrstr(1);
9 $LET textRange_max = $nrstr(100);
10 $LET textSelector = $nrstr(one);
11 $LET textSelector_count = $nrstr(3);
12 $LET textSelector1 = $nrstr(one);
13 $LET textSelector0 = $nrstr(3);
14 $LET textSelector3 = $nrstr(three);
15 $LET textSelector2 = $nrstr(two);
16
17 ODS _ALL_ CLOSE;
18 OPTIONS DEV=ACTIVEBX;
19 GOPTIONS XPIXELS=0 YPIXELS=0;
20 FILENAME EGSR TEMP;
21 ODS tagsets.sasreport13(ID=EGSR) FILE=EGSR
22   STYLE=HtmlBlue
23   STYLESHEET=(URL="file:///C:/Program%20Files/SASHome/SASEnvironment/7.1/Styles/HtmlBlue.css")
24   NOSTITLE
25 -----

```

A red box highlights the section of code from line 10 to line 15, which defines the `textSelector` macro variable and its count.

Display 108 - %LET Statements for Multiple Value Text Prompt

The log of the [Program node using the prompt definition](#) displays the values of the global variables created by the prompt.



The screenshot shows the SAS Enterprise Guide interface with the project tree containing a 'Program' node under 'Text_Prompts'. The main window displays the following log output:

```

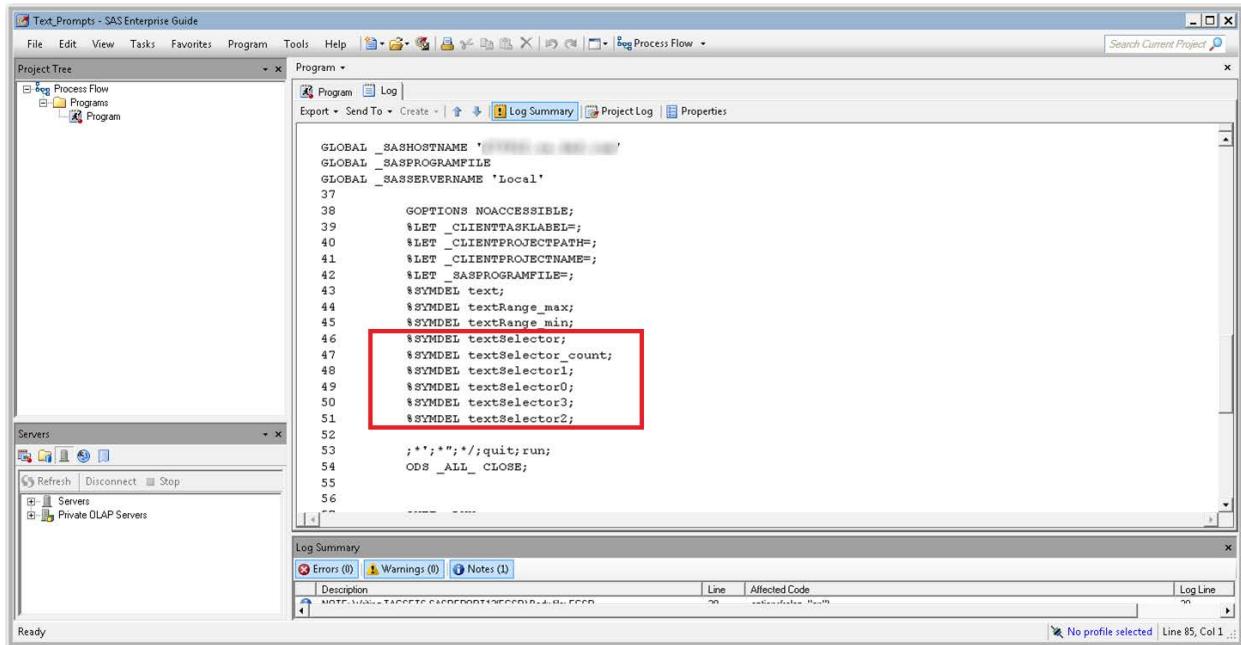
34 $mend;
35
36 %showmacvars();
GLOBAL SASWORKLOCATION "C:\Users\████████\AppData\Local\Temp\SEG10636\SAS Temporary Files\_TD12080_D79910_\Prc2\"
GLOBAL TEXT 'This is the default text value';
GLOBAL TEXTRANGE_MAX '100';
GLOBAL TEXTRANGE_MIN '1';
GLOBAL TEXTSELECTOR 'one';
GLOBAL TEXTSELECTOR0 '3';
GLOBAL TEXTSELECTOR1 'one';
GLOBAL TEXTSELECTOR2 'two';
GLOBAL TEXTSELECTOR3 'three';
GLOBAL TEXTSELECTOR_COUNT '3';
GLOBAL _CLIENTNAME 'SAS Enterprise Guide';
GLOBAL _CLIENTAPPNAME BG;
GLOBAL _CLIENTMACHINENAME '';
GLOBAL _CLIENTPROJECTNAME 'Text_Prompts.egp';
GLOBAL _CLIENTPROJECTPATH 'C:\EGFs\Tests\Local\Prompts\Text\Text_Prompts.egp';
GLOBAL _CLIENTTASKLABEL 'Program';
GLOBAL _CLIENTUSERID '████████';
GLOBAL _CLIENTUSERNAME 'J. Jeffreys-Chen';
GLOBAL _CLIENTVERSION '7.100.1.2651';
GLOBAL _INITIALIZEDSERVERID 1

```

A red box highlights the global variable assignments for `TEXTSELECTOR` and `TEXTSELECTOR_COUNT`.

Display 109 – Global Variables for Multiple Value Text Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.



The screenshot shows the SAS Enterprise Guide interface with a project titled "Text_Prompts". The "Program" tab is selected in the Project Tree. The main pane displays a SAS program:

```
GLOBAL _SASHOSTNAME '';
GLOBAL _SASPROGRAMFILE;
GLOBAL _SASSERVERNAME 'Local';
37
38      GOPTIONS NOACCESSIBLE;
39      %LET _CLIENTTASKLABEL="";
40      %LET _CLIENTPROJECTPATH="";
41      %LET _CLIENTPROJECTNAME="";
42      %LET _SASPROGRAMFILE="";
43      %SYMDEL text;
44      %SYMDEL textRange_max;
45      %SYMDEL textRange_min;
46      %SYMDEL textSelector;
47      %SYMDEL textSelector_count;
48      %SYMDEL textSelector1;
49      %SYMDEL textSelector0;
50      %SYMDEL textSelector3;
51      %SYMDEL textSelector2;
52
53      ; /* */; /* */;quit;run;
54      ODS _ALL_ CLOSE;
55
56
```

A red box highlights the lines 46 through 51, which contain the %SYMDEL statements. The "Log Summary" pane at the bottom shows no errors or warnings.

Display 110 - %SYMDEL Statements Remove the textSelector* Macro Variables

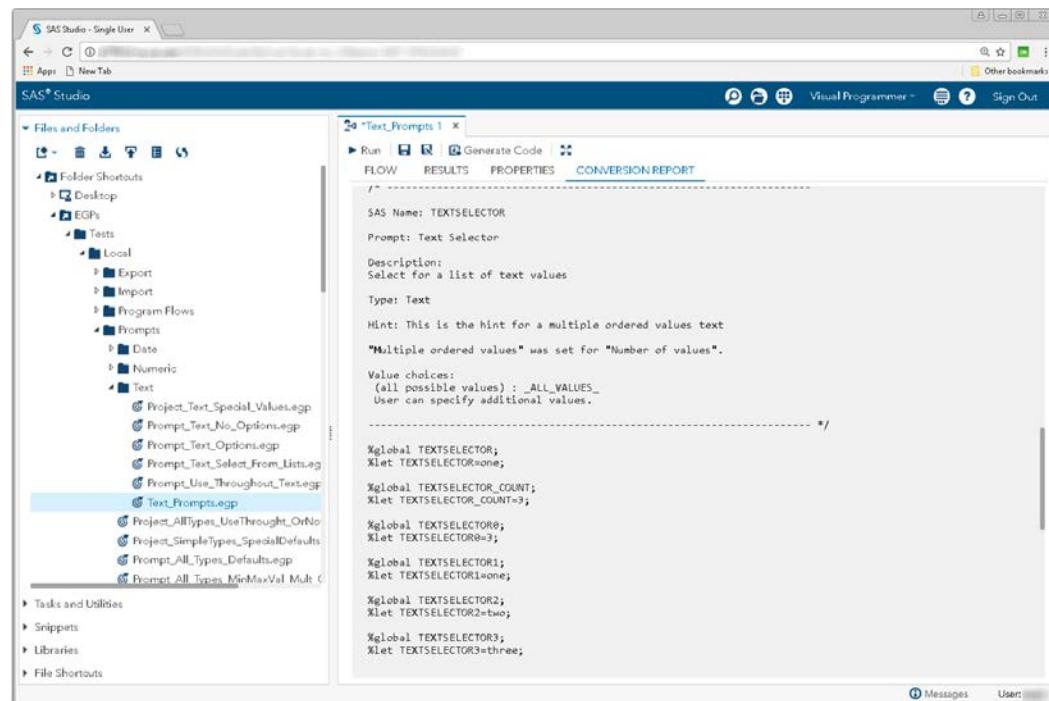
SAS Studio

The following display shows code that is added to the converted Program node for the multiple text values prompt in SAS Enterprise Guide. These global variables are created:

- TEXTSELECTOR
- TEXTSELECTOR_COUNT
- TEXTSELECTOR0
- TEXTSELECTOR1
- TEXTSELECTOR2
- TEXTSELECTOR3

The %LET statements assign the default values to the TEXTSELECTOR1, TEXTSELECTOR2, and TEXTSELECTOR3 variables.

If you want to run your process flow using different values for the TEXTSELECTOR prompt, you must manually update values of the macro variables values in the %LET statements. Note that in this example, the TEXTSELECTOR_COUNT and TEXTSELECTOR_0 variables must reflect the number of text selections you would like your program to process, and the TEXTSELECTORn variables must be in sequential order.



The screenshot shows the SAS Studio interface with the 'Text_Prompts 1' node selected. The left sidebar shows 'Files and Folders' with various project files listed under 'Text'. The main pane displays the node properties and its macro code. The properties tab shows the following details:

- SAS Name: TEXTSELECTOR
- Prompt: Text Selector
- Description: Select for a list of text values
- Type: Text
- Hint: This is the hint for a multiple ordered values text
- Value choices: (all possible values) : _ALL_VALUES_ User can specify additional values.

The macro code generated is:

```
%global TEXTSELECTOR;
%let TEXTSELECTOR=none;

%global TEXTSELECTOR_COUNT;
%let TEXTSELECTOR_COUNT=3;

%global TEXTSELECTOR0;
%let TEXTSELECTOR0=three;

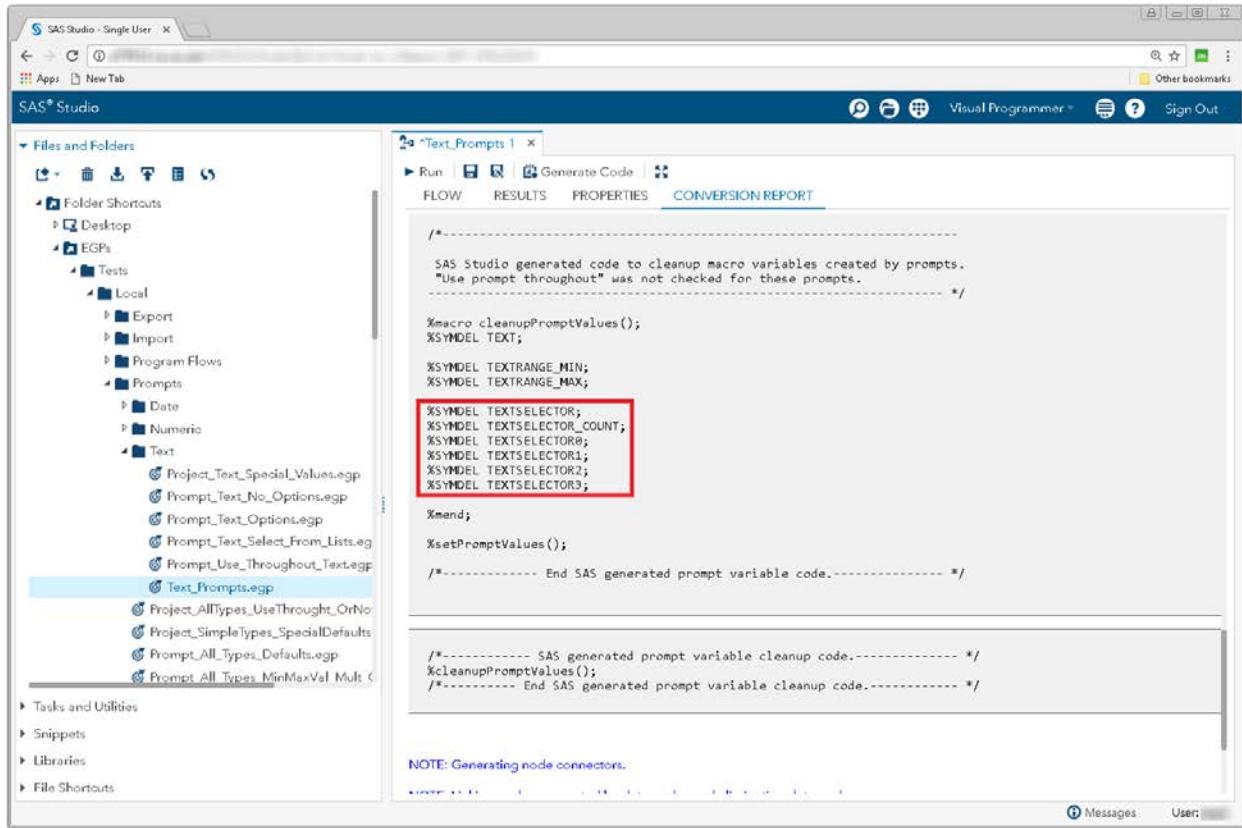
%global TEXTSELECTOR1;
%let TEXTSELECTOR1=one;

%global TEXTSELECTOR2;
%let TEXTSELECTOR2=two;

%global TEXTSELECTOR3;
%let TEXTSELECTOR3=three;
```

Display 111 - Macro Code for Multiple Text Values Prompt

Because the **Use prompt value throughout project** option is not checked, the %SYMDEL statements remove the TEXTSELECTOR* macro variables.



The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. The code editor displays the following SAS code:

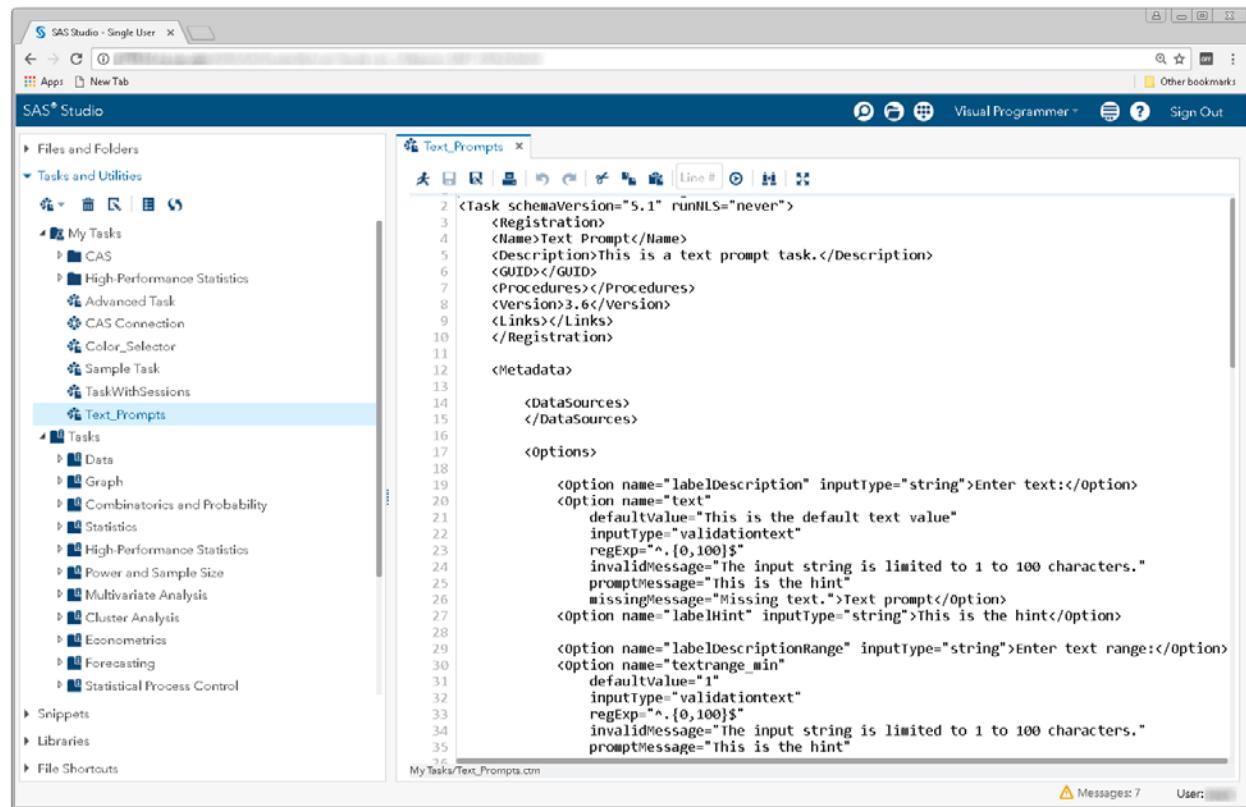
```
/*
-----  
SAS Studio generated code to cleanup macro variables created by prompts.  
"Use prompt throughout" was not checked for these prompts.  
----- */  
  
%macro cleanupPromptValues();  
%SYMDEL TEXT;  
  
%SYMDEL TEXT RANGE_MIN;  
%SYMDEL TEXT RANGE_MAX;  
  
%SYMDEL TEXTSELECTOR;  
%SYMDEL TEXTSELECTOR_COUNT;  
%SYMDEL TEXTSELECTOR0;  
%SYMDEL TEXTSELECTOR1;  
%SYMDEL TEXTSELECTOR2;  
%SYMDEL TEXTSELECTOR3;  
  
Xmend;  
  
XsetPromptValues();  
  
/*----- End SAS generated prompt variable code.----- */  
  
/*----- SAS generated prompt variable cleanup code.----- */  
%cleanupPromptValues();  
/*----- End SAS generated prompt variable cleanup code.----- */  
  
NOTE: Generating node connectors.  
----- */
```

A red box highlights the section of code where %SYMDEL statements are used to remove TEXTSELECTOR* macro variables.

Display 112 - %SYMDEL Statements Remove TEXTSELECTOR* Macro Variables

Substituting a SAS Studio Task for Text Prompts

1. Create a SAS Studio task with controls that represents the text prompts.
 - Add a label and text input controls. The names of the input controls should match the prompt names of text, textRange_min, textRange_max and so on.
 - Set the default values to the default values shown in the generated setPromptValues() macro in the converted Program node.
 - Change the strings of the input controls to match the strings specified in the prompts.



The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. On the left, the navigation pane displays 'Tasks and Utilities' with 'Text_Prompts' selected. The main workspace shows the following XML code:

```
2 <Task schemaVersion="5.1" runNLS="never">
3   <Registration>
4     <Name>Text Prompt</Name>
5     <Description>This is a text prompt task.</Description>
6     <GUID></GUID>
7     <Procedures></Procedures>
8     <version>3.6</Version>
9     <Links></Links>
10    </Registration>
11
12    <Metadata>
13
14      <DataSources>
15        </DataSources>
16
17      <Options>
18
19        <option name="labelDescription" inputType="string">Enter text:</option>
20        <option name="text"
21          defaultValue="This is the default text value"
22          inputType="validationtext"
23          regexp=".{0,100}$"
24          invalidMessage="The input string is limited to 1 to 100 characters."
25          promptMessage="This is the hint"
26          missingMessage="Missing text.">Text prompt</option>
27        <option name="labelHint" inputType="string">This is the hint</option>
28
29        <option name="labelDescriptionRange" inputType="string">Enter text range:</option>
30        <option name="textrange_min"
31          defaultValue="1"
32          inputType="validationtext"
33          regexp="^.{0,100}$"
34          invalidMessage="The input string is limited to 1 to 100 characters."
35          promptMessage="This is the hint">Text range</option>
36
37      </Options>
38
39    </Metadata>
40
41  </Task>
```

Display 113 - Replacement Task for Multiple Text Values Prompt

The following code is an example of a task that could be used for text prompts.

```
<?xml version="1.0" encoding="UTF-8"?>
<Task schemaVersion="5.1" runNLS="never">
    <Registration>
        <Name>Text Prompt</Name>
        <Description>This is a text prompt task.</Description>
        <GUID></GUID>
        <Procedures></Procedures>
        <Version>3.6</Version>
        <Links></Links>
    </Registration>
    <Metadata>

        <DataSources>
        </DataSources>

        <Options>

            <Option name="labelDescription" inputType="string">
                Enter text:
            </Option>
            <Option name="text">
                defaultValue="This is the default text value"
                inputType="validationtext"
                regExp=".{0,100}$"
                invalidMessage=
                    "The input string is limited to 1 to 100 characters."
                promptMessage="This is the hint"
                missingMessage="Missing text."
                Text prompt
            </Option>

            <Option name="labelHint" inputType="string">
                This is the hint
            </Option>
            <Option name="labelDescriptionRange" inputType="string">
                Enter text range:
            </Option>
            <Option name="textrange_min">
                defaultValue="1"
                inputType="validationtext"
                regExp="^.{0,100}$"
                invalidMessage=
                    "The input string is limited to 1 to 100 characters."
                promptMessage="This is the hint"
                missingMessage="Missing text."
                From:
            </Option>
            <Option name="textrange_max">
                defaultValue="100"
                inputType="validationtext"
                regExp=".{0,100}$"
                invalidMessage=
                    "The input string is limited to 1 to 100 characters."
                promptMessage="This is the hint"
                missingMessage="Missing text."
                to:
            </Option>

        </Options>
    </Metadata>
</Task>
```

```

        <Option name="labelDescriptionTextSel" inputType="string">
            Text selector
        </Option>
        <Option name="multientry" inputType="multientry">
            Select for a list of text values
        </Option>
        <Option name="one" inputType="string">one</Option>
        <Option name="two" inputType="string">two</Option>
        <Option name="three" inputType="string">three</Option>
        <Option name="labelDescriptionTextSelHint" inputType="string">
            This is the hint for a multiple ordered values text
        </Option>
    </Options>
</Metadata>
<UI>
    <OptionItem option="labelDescription"/>
    <OptionItem option="text"/>
    <OptionItem option="labelHint"/>

    <OptionItem option="labelDescriptionRange"/>
    <OptionItem option="textrange_min"/>
    <OptionItem option="textrange_max"/>

    <OptionItem option="labelDescriptionTextSel"/>

    <OptionChoice option="multientry">
        <OptionItem option="one"/>
        <OptionItem option="two"/>
        <OptionItem option="three"/>
    </OptionChoice>
    <OptionItem option="labelDescriptionTextSelHint"/>
</UI>
<CodeTemplate>
    <![CDATA[
%global text;
%let text=$text;

%global textrange_min;
%global textrange_max;
%let textrange_min=$textrange_min;
%let textrange_max=$textrange_max;

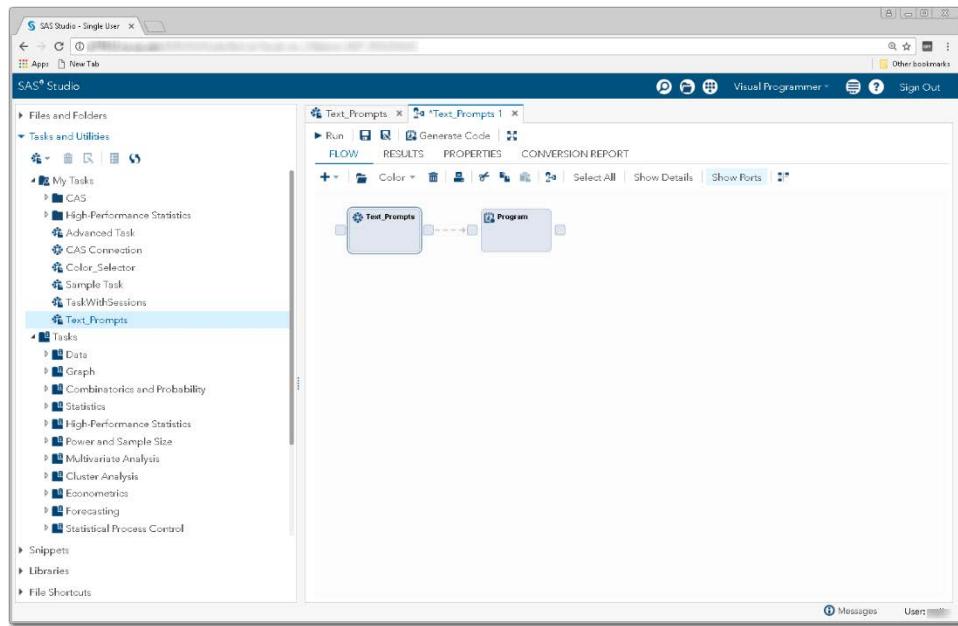
%global multientry;
%let multientry=$multientry;

#if ($multientry.size()>0)
%global textselector_count;
%let textselector_count=$multientry.size();
%global textselector0;
%let textselector0=$multientry.size();
%global textselector;
%let textselector=$multientry[0];

#set($counter = 1)
#foreach ($id in $multientry)
    %global textselector$counter;
    %let textselector$counter=$id;
    #set($counter = $counter + 1)
#end
#end
        ]]>
</CodeTemplate>
</Task>

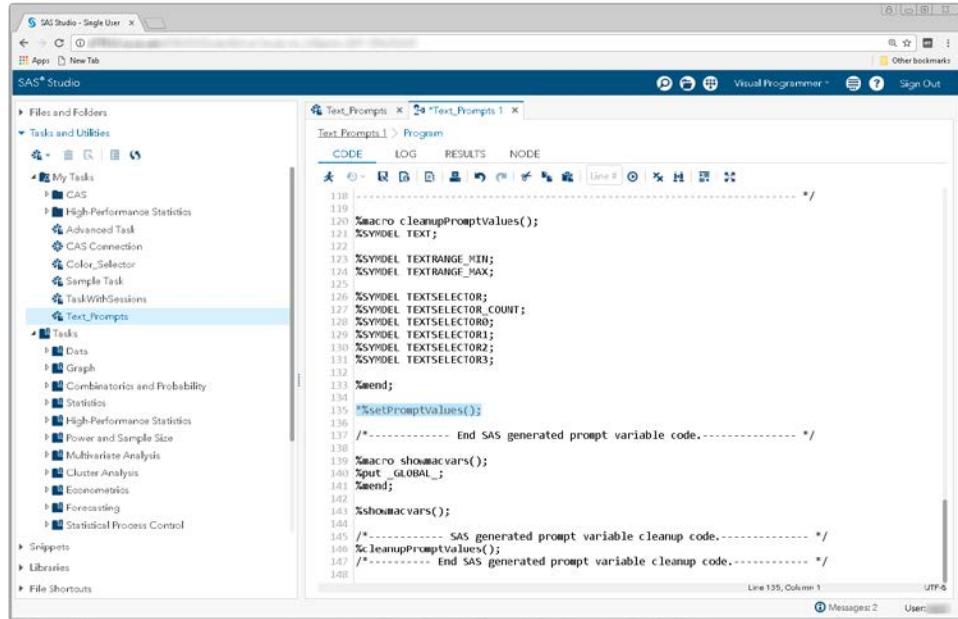
```

2. Save the prompt replacement task to your **My Tasks** folder.
 3. Drag the task from **My Tasks** into your converted process flow.
 4. Link the output port of the Task node to the input port for the converted Program node.



Display 114 – Linking the Output from the Task Node to the Program Node

- Comment out the `%setPromptValues()` macro call in the converted Program node. The macro code generated by the new task replaces this code.



Display 115 - Commented Out %setPromptValues() Macro Call

To run your flow with text values other than the default values, open the Text_Prompts node and specify different values.

The screenshot shows the SAS Studio interface with the 'Text_Prompts' task selected. On the left, the navigation pane lists various tasks and utilities. The main workspace displays the 'Text_Prompts' configuration window. In the 'Enter text:' section, 'Text prompt' is set to 'New value' and 'This is the hint' is 'Enter text range:'. Under 'From:' and 'to:', the values '1' and '100' are entered respectively. In the 'Text selector' section, 'Select for a list of text values' is shown with options 'one', 'two', 'three', and 'four', where 'four' is currently selected. The right side of the screen shows the generated SAS code:

```

8 * Generated on SAS platform 'X64_7PRO MIN'
9 * Generated on SAS version '9.04.01WIP12042013'
10 * Generated on browser 'Mozilla/5.0 (Windows NT 6.1; '
11 * Generated on web client 'http://d79910.na.sas.com: '
12 *
13 */
14
15 %global text;
16 %let text=<New value>;
17 %global textrange_min;
18 %global textrange_max;
19 %let textrange_min=1;
20 %let textrange_max=100;
21 %global multientry;
22 %let multientry=[one, two, three, four];
23 %global textselector_count;
24 %let textselector_count=4;
25 %global textselector0;
26 %let textselector0=;
27 %global textselector1;
28 %let textselector1=one;
29 %global textselector2;
30 %let textselector2=one;
31 %global textselector3;
32 %let textselector3=two;
33 %global textselector4;
34 %let textselector4=three;
35 %global textselector5;
36 %let textselector5=four;

```

Display 116 – User Interface and Generated SAS Code for Text_Prompts Task

When you execute the process flow, the global Text* variables are set to the values that you specified in the new task.

The screenshot shows the SAS Studio interface with the 'Text_Prompts' task selected. The generated SAS code is displayed in the 'CODE' tab of the 'Program' window. The code defines several global variables based on the specified text prompts:

```

GLOBAL TEXT New_value
GLOBAL TEXTRANGE_MAX 100
GLOBAL TEXTRANGE_MIN 1
GLOBAL TEXTSELECTOR one
GLOBAL TEXTSELECTOR0 4
GLOBAL TEXTSELECTOR1 one
GLOBAL TEXTSELECTOR2 two
GLOBAL TEXTSELECTOR3 three
GLOBAL TEXTSELECTOR4 four
GLOBAL TEXTSELECTOR_COUNT 4
GLOBAL USERDIR
GLOBAL BASEURL
GLOBAL CLIENTAPP SAS Studio
GLOBAL CLIENTAPPVERSION 3.6
GLOBAL CLIENTUSERID
GLOBAL CLIENTUSERNAME
GLOBAL EXECENV SASStudio
GLOBAL SASPROGRAMFILE
GLOBAL SASERVERNAME localhost
GLOBAL SASMTEMP .imagescbfd0fd3a02b448e84476c271fa570e5
GLOBAL _SASMS_ 205
206 /*----- SAS generated prompt variable cleanup code.-----*/
207 %cleanupPromptValues();
208 /*----- End SAS generated prompt variable cleanup code.-----*/
209
210 OPTIONS NONOTES NOSTIMER NOSOURCE NOSYNTAXCHECK;
211
212
213
214
215
216
217
218
219
220
221
222
223
224

```

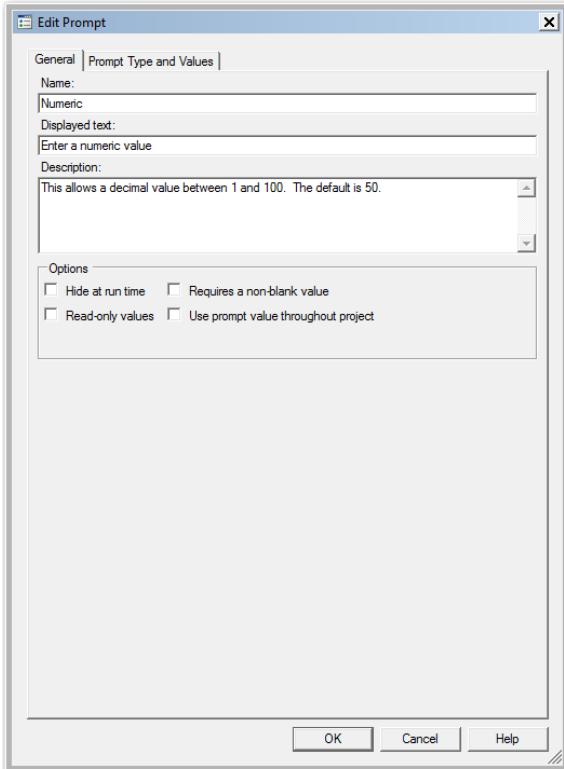
Display 117 - Text Prompt Variables with Updated Values

Numeric

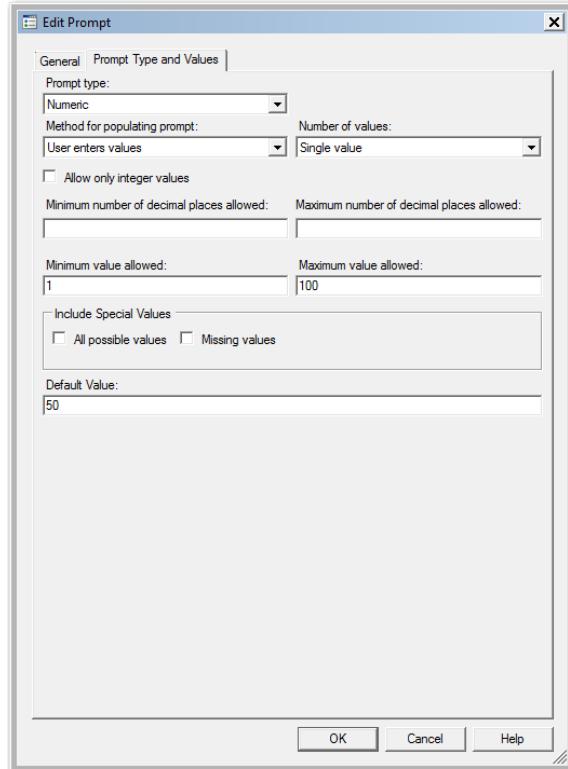
Single Number

SAS Enterprise Guide

In this example, a single value numeric prompt named Numeric is defined as shown in the following two displays.

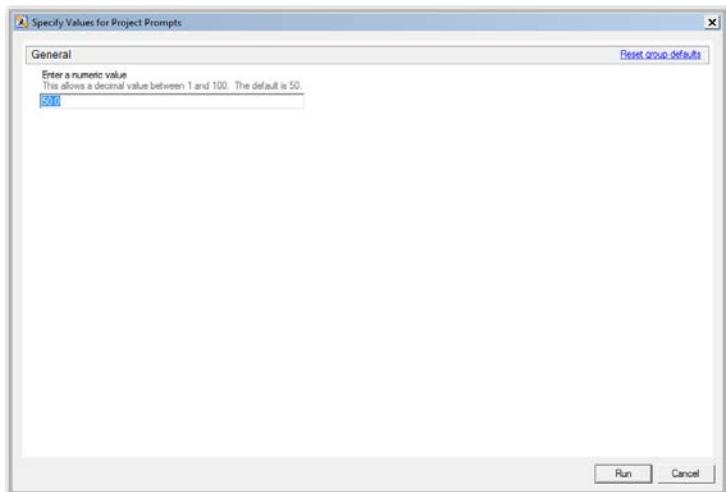


Display 118 - General Properties for Numeric Prompt



Display 119 - Type and Values for Numeric Prompt

When you run the Program node that depends on this prompt, the following dialog box appears:



Display 120 - Numeric Prompt in Prompt Dialog Box

If the user leaves the default value in the single numeric value prompt field, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

A %LET statement assigns the value specified in the prompt dialog box to the Numeric macro variable.

The log of the [Program node using the prompt definition](#) displays the value of the global variable created by the prompt.

The screenshot shows the SAS Enterprise Guide interface with a 'Program' node selected in the Project Tree. The code pane contains the following SAS code:

```

7  %LET Numeric = $nrstr(50.0);
8
9  ODS _ALL_ CLOSE;
10 OPTIONS DEACTIVEVEX;
11 GOPTIONS XPIXELS=0 YPIXELS=0;
12 FILENAME EGSR TEMP;
13 ODS tagsets.sasreport13(ID=EGSR) FILE=EGSR
14   STYLE=HTMLBlue
15   STYLESHEET=(URL="file:///C:/Program%20Files/SASHome/SASEnterpriseGuide/7.1/Sty
16   NOGRTITLE
17   NOGFOOTNOTE
18   GPATH="&worklocation
19   ENCODING=UTF8
20   options(roleap="on")
21 ;
22 NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR
23 GOPTIONS ACCESSIBLE;
24 %macro showmacvars();
25 %put _GLOBAL_;
26 %need;
27 %showmacvars();
28 GLOBAL NUMERIC '50.0';
29 GLOBAL SASWORKLOCATION "C:\Users\...\AppData\Local\Temp\$KG13096\SAS Temporary Files\_TDIC
30 _CLIENTAPPNAME 'SAS Enterprise Guide'
31 _CLIENTAPPREV '05'
32 _CLIENTMACHINE ' '
33 _CLIENTPROJECTNAME 'Prompt_Single_Numeric.egp'
34 _CLIENTPROJECTPATH 'C:\EGSF\Tests\Local\Prompts\Numeric\Prompt_Single_Numeric.egp'
35
36 /* *;*/quit;run;
37 ODS _ALL_ CLOSE;
38
39
40 QUIT; RUN;
41

```

The Log Summary window below shows a note about writing the report body file.

Display 121 - Global Variable and %LET Statement for Numeric Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statement removes the macro variable at the end of the program.

The screenshot shows the SAS Enterprise Guide interface with a 'Program' node selected in the Project Tree. The code pane contains the following SAS code:

```

34 $SYMDEL Numeric;
35
36 /* *;*/quit;run;
37 ODS _ALL_ CLOSE;
38
39
40 QUIT; RUN;
41

```

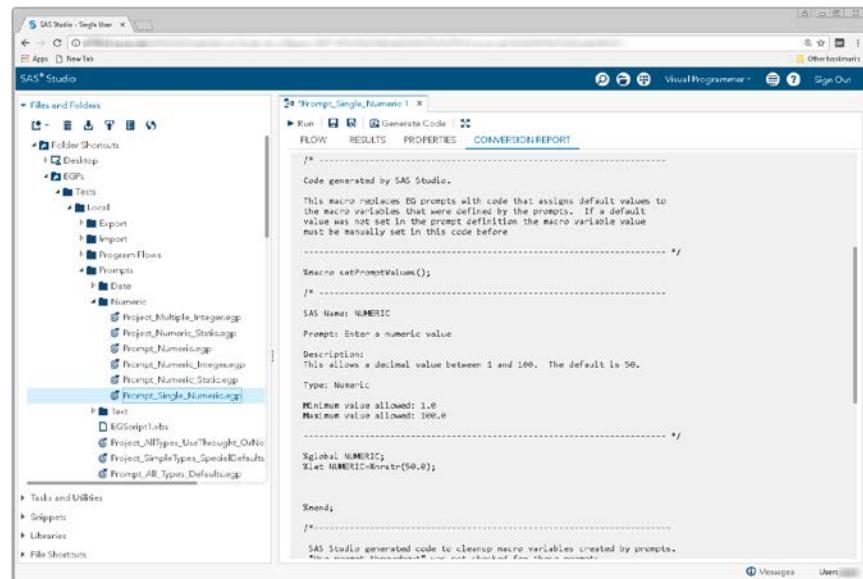
The Log Summary window below shows a note about writing the report body file.

Display 122 - %SYMDEL Statement Removes Numeric Macro Variable

SAS Studio

The following display shows code that is added to the converted Program node for the numeric prompt in SAS Enterprise Guide.

A global variable named NUMERIC is created and a %LET statement assigns the default value to NUMERIC. If you want to run your process flow using different values for the NUMERIC prompt, you must manually update value of the macro variable in the %LET statement.

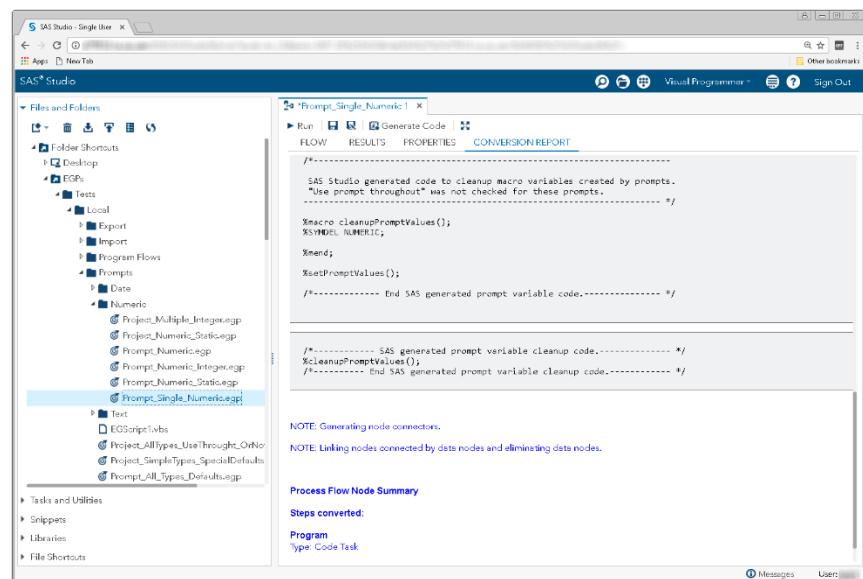


The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. On the left, the 'Files and Folders' tree view shows a project structure with various EG nodes and a 'Prompts' folder containing several EG files, one of which is highlighted: 'Prompt_Single_Numeric.egp'. The main workspace displays the generated SAS code for this node:

```
/*-----  
Code generated by SAS Studio.  
  
This macro replaces EG prompts with code that assigns default values to  
the macro variables that were defined by the prompts. If a default  
value was not set in the prompt definition the macro variable value  
must be manually set in this code before  
-----*/  
  
Macro setPromptValue();  
/*-----  
SAS Name: NUMERIC  
Prompt: Enter a numeric value  
Description:  
This allows a decimal value between 1 and 100. The default is 50.  
Type: Numeric  
Minimum value allowed: 1.0  
Maximum value allowed: 100.0  
-----*/  
  
SAS Name: NUMERIC;  
Set NUMERIC=%str(50.0);  
  
Xmend;  
/*-----  
SAS Studio generated code to cleanup macro variables created by prompts.  
-----*/
```

Display 123 - Macro Code for Numeric Prompt

Because the **Use prompt value throughout project** option is not checked, the %SYMDEL statements remove the NUMERIC macro variable.



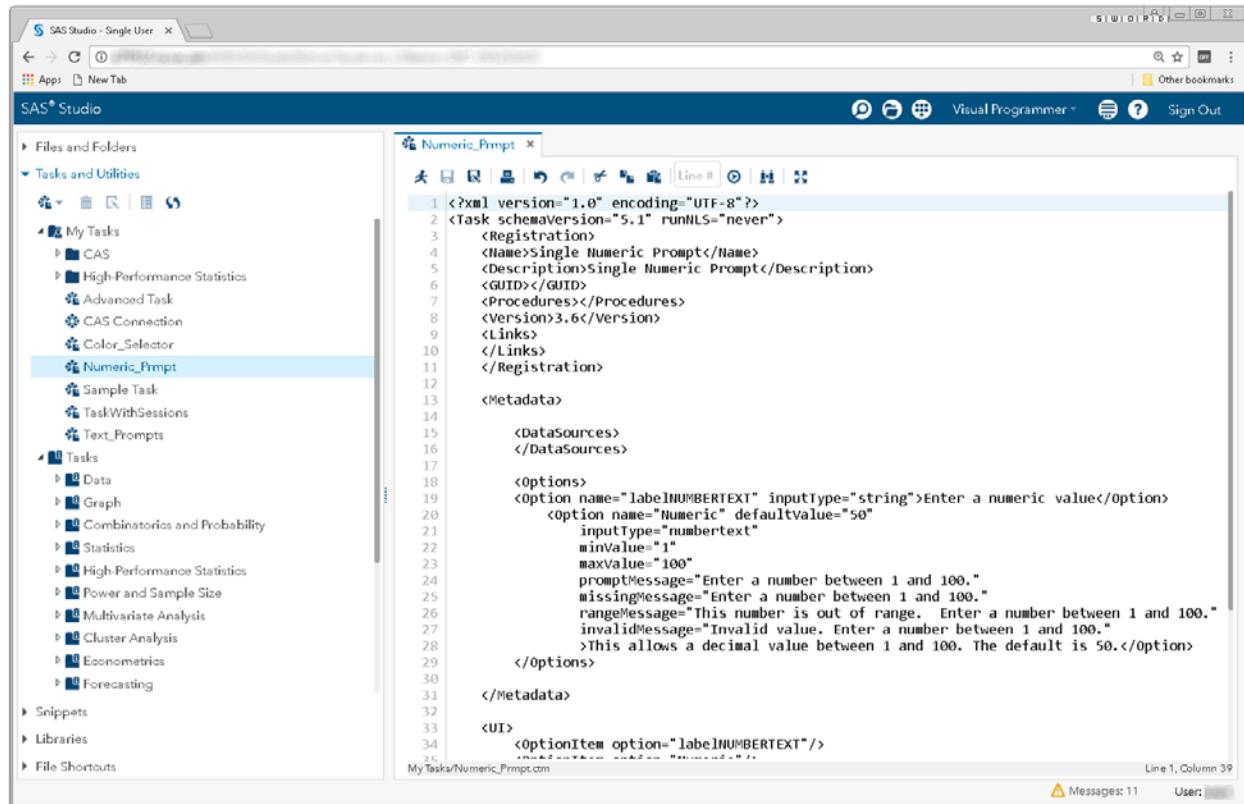
The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. On the left, the 'Files and Folders' tree view shows a project structure with various EG nodes and a 'Prompts' folder containing several EG files, one of which is highlighted: 'Prompt_Single_Numeric.egp'. The main workspace displays the generated SAS code for this node, which includes %SYMDEL statements to remove the NUMERIC macro variable:

```
/*-----  
SAS Studio generated code to cleanup macro variables created by prompts.  
"Use prompt throughout" was not checked for these prompts.  
-----*/  
  
Macro cleanupPromptValues();  
%SYMDEL NUMERIC;  
Xmend;  
XsetPromptValues();  
/*----- End SAS generated prompt variable code.-----*/  
  
/*----- SAS generated prompt variable cleanup code.----- */  
%cleanupPromptValues();  
/*----- End SAS generated prompt variable cleanup code.----- */  
  
NOTE: Generating node connectors.  
NOTE: Linking nodes connected by dists nodes and eliminating data nodes.  
  
Process Flow Node Summary  
Steps converted:  
Program  
Type: Code Task
```

Display 124 - %SYMDEL Statements Remove Numeric Macro Variable

Substituting a SAS Studio Task for Numeric Prompt

1. Create a SAS Studio task with a control that represents the Numeric prompt.
 - Add a label and a numbertext input control. The name of the input control should match the prompt name of Numeric.
 - Set the default value to the default value shown in the generated `setPromptValues()` macro in the converted Program node.
 - Change the prompt string to match the string specified in the prompt.



The screenshot shows the SAS Studio interface with the Visual Programmer tool open. The left sidebar shows a tree view of tasks and utilities, with 'Tasks and Utilities' expanded and 'Numeric_Prompt' selected. The main pane displays the XML code for the task:

```
<?xml version="1.0" encoding="UTF-8"?>
<Task schemaVersion="5.1" runMLS="never">
    <Registration>
        <Name>Single Numeric Prompt</Name>
        <Description>Single Numeric Prompt</Description>
        <GUID></GUID>
        <Procedures></Procedures>
        <Version>3.6</Version>
        <Links>
        </Links>
    </Registration>
    <Metadata>
        <DataSources>
        </DataSources>
        <Options>
            <Option name="labelNUMBERTEXT" inputType="string">Enter a numeric value</Option>
            <Option name="Numeric" defaultValue="50"
                    inputType="numbertext"
                    minValue="1"
                    maxValue="100"
                    promptMessage="Enter a number between 1 and 100."
                    missingMessage="Enter a number between 1 and 100."
                    rangeMessage="This number is out of range. Enter a number between 1 and 100."
                    invalidMessage="Invalid value. Enter a number between 1 and 100.">
                <!-- This allows a decimal value between 1 and 100. The default is 50.-->
            </Options>
        </Metadata>
        <UI>
            <OptionItem option="labelNUMBERTEXT"/>
        </UI>
    </Task>
```

Display 125 - Replacement Task for Numeric Prompt

The following code is an example of a task that could be used as the numeric prompt.

```
<?xml version="1.0" encoding="UTF-8"?>
<Task schemaVersion="5.1" runNLS="never">
    <Registration>
        <Name>Single Numeric Prompt</Name>
        <Description>Single Numeric Prompt</Description>
        <GUID></GUID>
        <Procedures></Procedures>
        <Version>3.6</Version>
        <Links>
        </Links>
    </Registration>

    <Metadata>

        <DataSources>
        </DataSources>

        <Options>
            <Option name="labelNUMBERTEXT" inputType="string">
                Enter a numeric value
            </Option>

            <Option name="Numeric" defaultValue="50"
                   inputType="numbertext"
                   minValue="1"
                   maxValue="100"
                   promptMessage="Enter a number between 1 and 100."
                   missingMessage="Enter a number between 1 and 100."
                   rangeMessage=
"This number is out of range. Enter a number between 1 and 100."
                   invalidMessage=
"Invalid value. Enter a number between 1 and 100."
            >
                This allows a decimal value between 1 and 100. The default is 50.
            </Option>
        </Options>

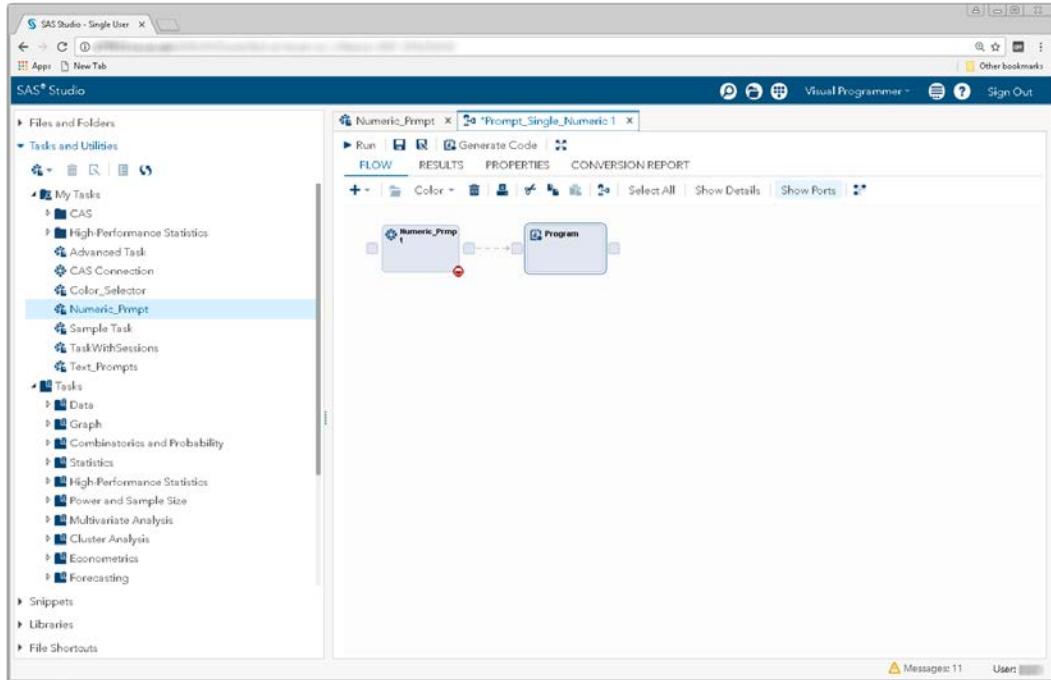
    </Metadata>

    <UI>
        <OptionItem option="labelNUMBERTEXT"/>
        <OptionItem option="Numeric"/>
    </UI>

    <CodeTemplate>
        <![CDATA[
%global Numeric;
%let Numeric=$Numeric;

        ]]>
    </CodeTemplate>
</Task>
```

2. Save the prompt replacement task to your **My Tasks** folder.
3. Drag the task from **My Tasks** into your converted process flow.
4. Link the output port of the new task to the input port of the converted Program node.



Display 126 - Numeric Input Task Linked to Program Node

5. Comment out the `%setPromptValues()` macro call from the converted Program node. The macro code generated by the numeric input task replaces this code.

```

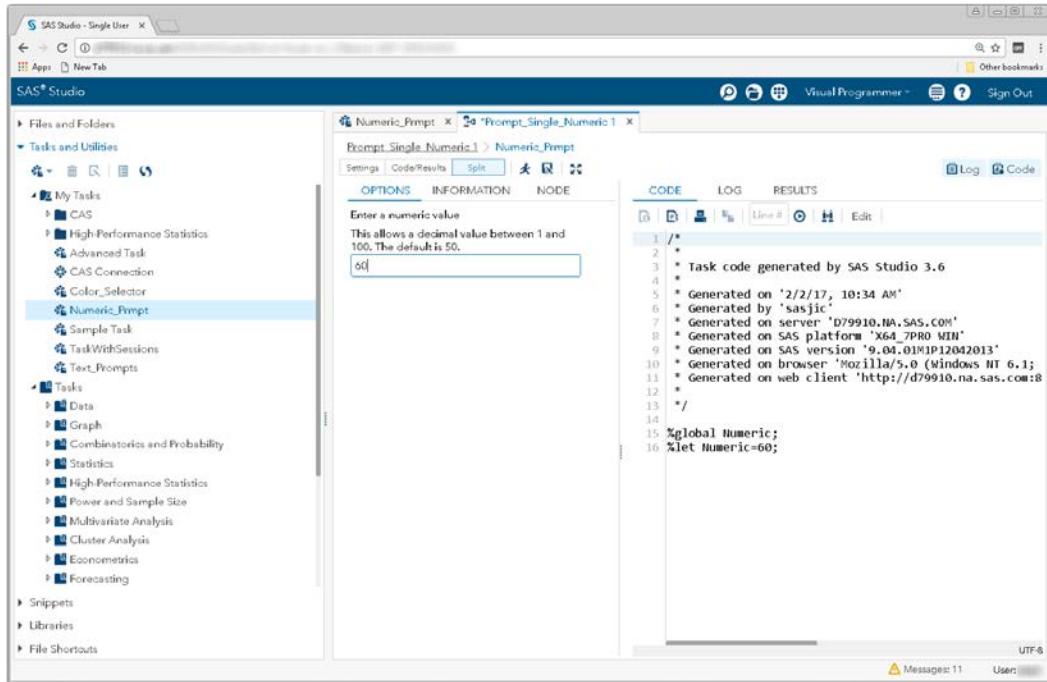
30 %global NUMERIC;
31 %let NUMERIC=%nrstr(50.0);
32
33
34
35 %end;
36
37 /*-----
38
39 SAS Studio generated code to cleanup macro variables created by prompts.
40 "use prompt throughout" was not checked for these prompts.
41 -----
42 %macro cleanupPromptValues();
43 %symdel NUMERIC;
44
45 %end;
46
47
48 /*%setPromptValues();
49
50 /*----- End SAS generated prompt variable code.----- */
51
52 %macro showmacvars();
53 %put _GLOBAL_;
54 %end;
55 %showmacvars();
56
57 /*----- SAS generated prompt variable cleanup code.----- */
58 %cleanupPromptValues();
59 /*----- End SAS generated prompt variable cleanup code.----- */
60

```

The screenshot shows the SAS Studio interface with the 'CODE' tab selected in the 'Prompt_Single_Numeric_1 > Program' editor. The code block contains several SAS macros and a comment indicating that code was generated to clean up macro variables. A specific line of code, `/*%setPromptValues();`, is highlighted with a light blue background, indicating it has been commented out. The top menu bar shows 'SAS Studio - Single User' and various toolbars.

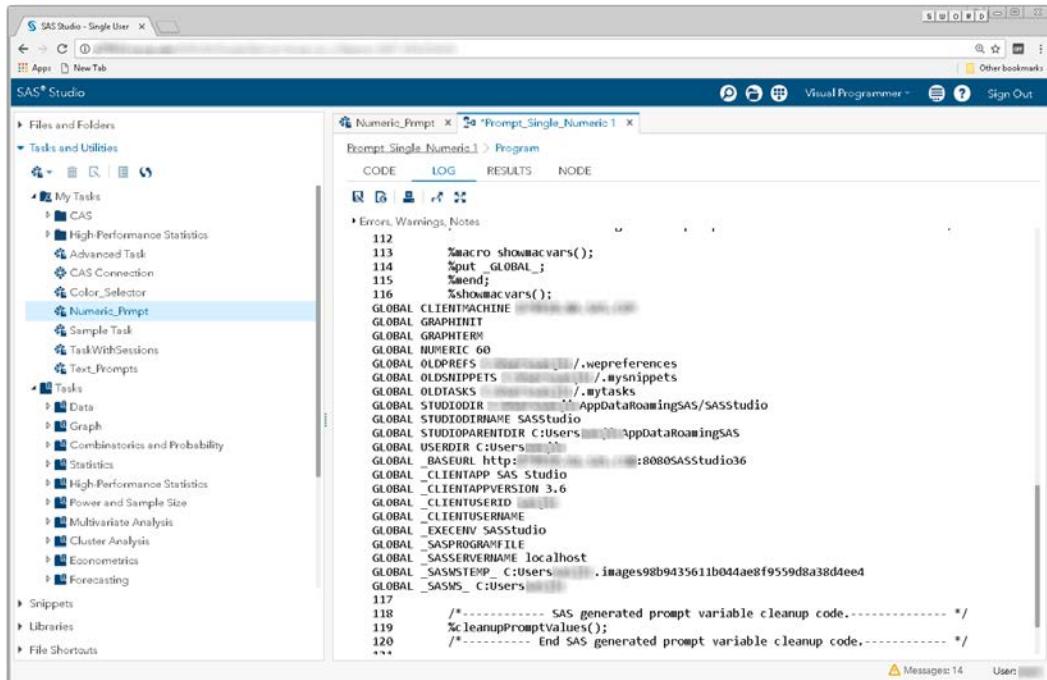
Display 127 - Commented Out `%setPromptValues` Macro Call

To run your flow with a different numeric value than the default value, open the Prompt_Single_Numeric node and specify a different number.



Display 128 – Running the Numeric Input Task

When you run the process flow, the global Numeric variable is set to the value specified in the task.

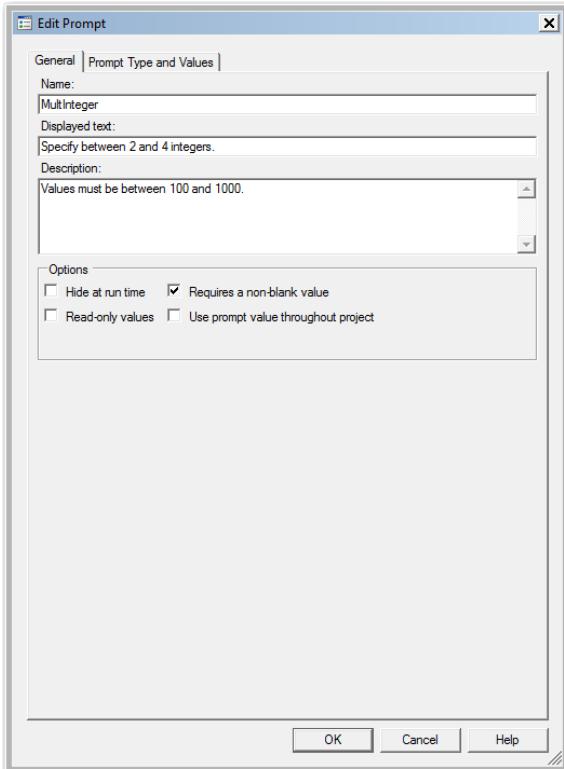


Display 129 - Numeric Prompt Variable with Updated Value

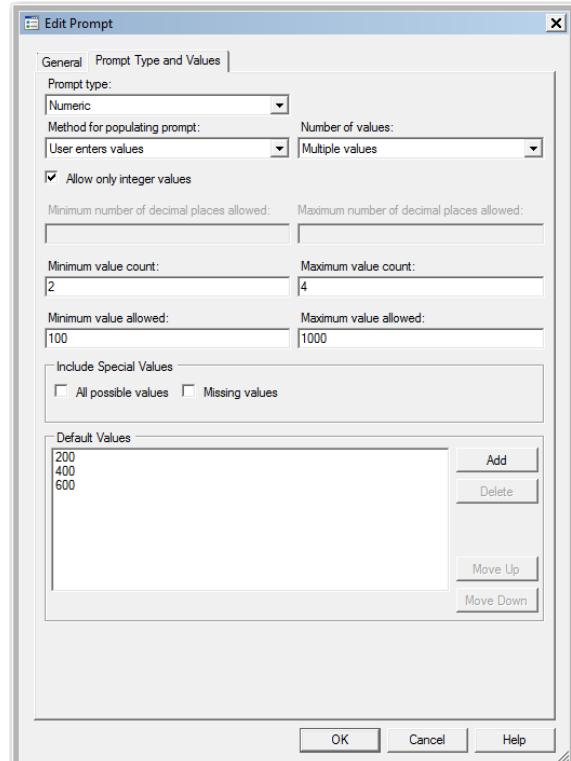
Multiple Numerics

SAS Enterprise Guide

In this example, a multiple values numeric prompt named MultInteger is defined as shown in the following two displays.

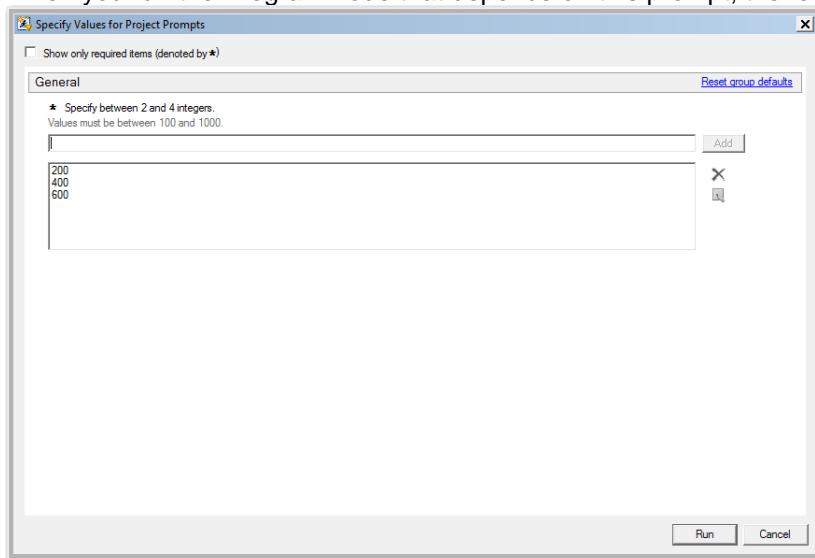


Display 130 - General Properties for Numeric Multiple Values Prompt



Display 131 - Type and Values for Numeric Multiple Values Prompt

When you run the Program node that depends on this prompt, the following dialog box appears.



Display 132 - Multiple Numeric Values Prompt in Prompt Dialog Box

If the user leaves the default value in the multiple numeric value prompt field, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The %LET statements assign the values specified in the prompt dialog box to the MultInteger* macro variables.

```

1  /*+; */quit;run;
2  OPTIONS PAGENO=MIN;
3  %LET _CLIENTTASKLABEL="Program";
4  %LET _CLIENTPROJECTPATH="";
5  %LET _CLIENTPROJECTNAME="";
6  %LET _SASPROGRAMFILE="";
7  %LET MultInteger1 = %nrstr(200);
8  %LET MultInteger_count = %nrstr(3);
9  %LET MultInteger2 = %nrstr(400);
10 %LET MultInteger3 = %nrstr(600);
11 %LET MultInteger = %nrstr(200);
12 %LET MultInteger0 = %nrstr(3);
13
14 ODS _ALL_ CLOSE;
15 OPTIONS DEV=ACTIVE;
16 GOPTIONS XPIXELS=0 YPIXELS=0;
17 FILENAME EGSR TEMP;
18 ODS tagsets.sasreport13 ID=EGSR FILE=EGSR
19   STYLE=HTMLBlue
20   STYLESHEET=(URL="file:///C:/Program%20Files/SASHome/SASEnterpriseGuide/7.1/sty
21   NOGRTITLE
22   NOGOOOTNOTE
23   GRATHF%masworklocation
24   ENCODING=UTF8
25   options(rolap="on")
26   ;
27 NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR
28 GOPTIONS ACCESSIBLE;

```

Description	Line	Affected Code
NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR	27	options(rolap="on")

Display 133 - %LET Statements for Multiple Numeric Values Prompt

The log of the [Program node using the prompt definition](#) displays the values of the global variables created by the prompt.

```

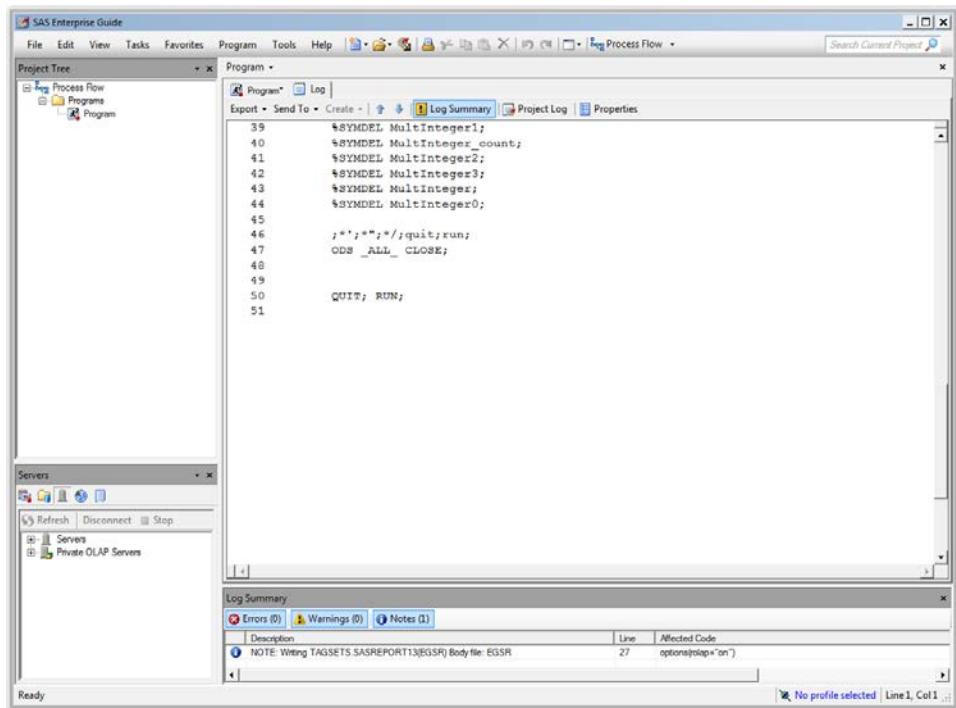
32  %showmacrovars();
GLOBAL MULTINTEGER '200';
GLOBAL MULTINTEGER0 '3';
GLOBAL MULTINTEGER1 '200';
GLOBAL MULTINTEGER2 '400';
GLOBAL MULTINTEGER3 '600';
GLOBAL MULTINTEGER_COUN '3';
GLOBAL SASWORKLOCATION "C:\Users\...\AppData\Local\Temp\SEG13096\SAS Temporary Files\_2D1C
GLOBAL _CLIENTAPP 'SAS Enterprise Guide'
GLOBAL _CLIENTAPPNAME EG
GLOBAL _CLIENTMACHINE ''
GLOBAL _CLIENTPROJECTNAME ''
GLOBAL _CLIENTPROJECTPATH ''
GLOBAL _CLIENTTASKLABEL 'Program'
GLOBAL _CLIENTUSERID ''
GLOBAL _CLIENTUSERNAME 'J. Jeffreys-Chen'
GLOBAL _CLIENTVERSION '7.100.1.2651'
GLOBAL _INITIALIZEDSERVERID 1
GLOBAL _SASHOSTNAME ''
GLOBAL _SASPROGRAMFILE ''
GLOBAL _SASSERVERNAME 'Local'
33
34  GOPTIONS NOACCESSIBLE;
35  %LET _CLIENTTASKLABEL="";
36  %LET _CLIENTPROJECTPATH="";
37
38  %LET _CLIENTPROJECTNAME="";
39  %LET _SASPROGRAMFILE="";

```

Description	Line	Affected Code
NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR	27	options(rolap="on")

Display 134 - Macro Variables for Multiple Numeric Values Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.



The screenshot shows the SAS Enterprise Guide interface. In the top menu bar, 'File', 'Edit', 'View', 'Tasks', 'Favorites', 'Program', 'Tools', 'Help' are visible. Below the menu is a toolbar with icons for opening files, saving, printing, and navigating. The main window has a 'Project Tree' on the left showing a 'Program' node under 'Programs'. The central area is a 'Program' editor with the following code:

```
39      %SYMDEL MultInteger1;
40      %SYMDEL MultInteger_count;
41      %SYMDEL MultInteger2;
42      %SYMDEL MultInteger3;
43      %SYMDEL MultInteger;
44      %SYMDEL MultInteger0;
45
46      /* */;*/;quit;run;
47      ODS _ALL_ CLOSE;
48
49
50      QUIT; RUN;
51
```

Below the editor is a 'Servers' panel showing a connection to 'Private OLAP Servers'. At the bottom is a 'Log Summary' panel with tabs for 'Errors (0)', 'Warnings (0)', and 'Notes (1)'. The 'Notes' tab is selected, displaying the following note:

Description	Line	Affected Code
NOTE: Writing TAGSETS SASREPORT13(EGSR) Body file: EGSR	27	options(tagsap=on')

Display 135 - %SYMDEL Statements Remove MultInteger* Macro Variables

SAS Studio

The following display shows the code that is added to the converted Program node for the multiple numeric values prompt in SAS Enterprise Guide.

These global variables are created:

- MULTINTEGER
- MULTINTEGER_COUNT
- MULTINTEGER0
- MULTINTEGER1
- MULTINTEGER2
- MULTINTEGER3

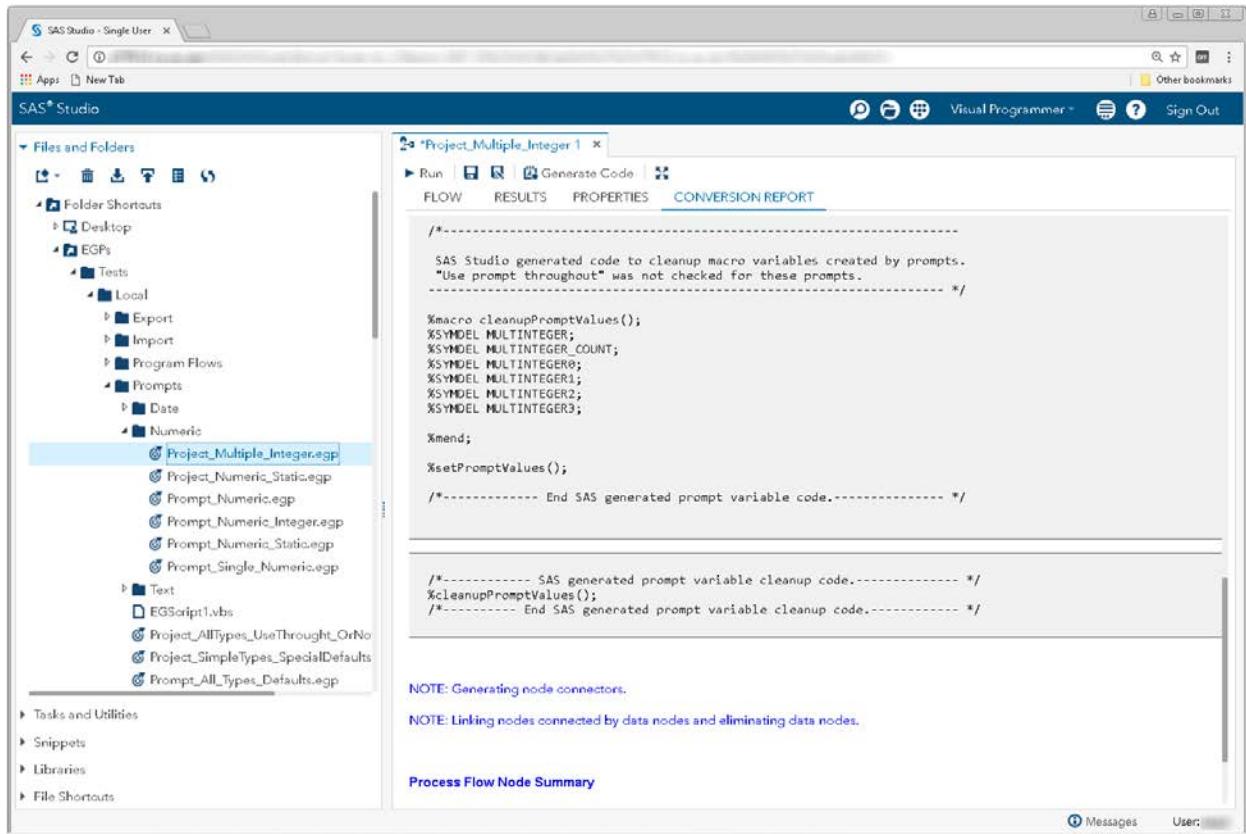
A %LET statement assigns the default values to the MULTINTEGER1, MULTINTEGER2, and MULTINTEGER3 variables.

If you want to run your process flow using different values for the MULTINTEGER prompt, you must manually update values of the macro variables in the %LET statements. Note that in this example, the MULTINTEGER_COUNT and MULTINTEGER_0 variables must reflect the number of text selections you want your program to process, and the MULTINTEGERn variables must be in sequential order.

```
%macro setPromptValues();
/* ----- */
*SAS Name: MULTINTEGER
Prompt: Specify between 2 and 4 integers.
Description:
Values must be between 100 and 1000.
Type: Numeric - Integer
Minimum value allowed: 100
Maximum value allowed: 1000
Between 2 and 4 values are required.
----- */
%global MULTINTEGER;
%let MULTINTEGER=200;
%global MULTINTEGER_COUNT;
%let MULTINTEGER_COUNT=3;
%global MULTINTEGER0;
%let MULTINTEGER0=3;
%global MULTINTEGER1;
%let MULTINTEGER1=200;
%global MULTINTEGER2;
%let MULTINTEGER2=400;
%global MULTINTEGER3;
%let MULTINTEGER3=600;
```

Display 136 - Code for Multiple Numeric Values Prompt

Because the **Use prompt value throughout project** option is not checked, the %SYMDEL statements remove the MULTINTEGER macro variables.



The screenshot shows the SAS Studio interface. On the left, the 'Files and Folders' browser displays a project structure with various EGP files. In the center, the 'Visual Programmer' window is open, showing a code editor with the following content:

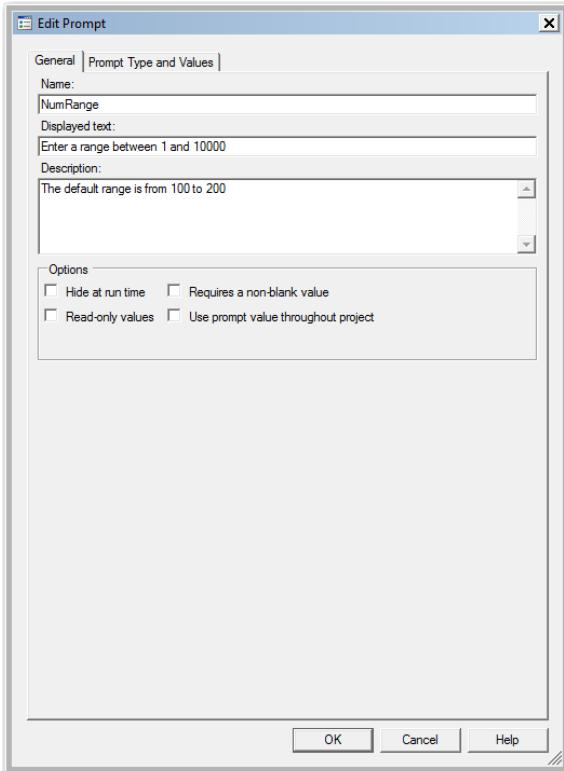
```
/*
-----  
SAS Studio generated code to cleanup macro variables created by prompts.  
"Use prompt throughout" was not checked for these prompts.  
----- */  
  
%macro cleanupPromptValues();  
%SYMDEL MULTINTEGER;  
%SYMDEL MULTINTEGER_COUNT;  
%SYMDEL MULTINTEGER8;  
%SYMDEL MULTINTEGER1;  
%SYMDEL MULTINTEGER2;  
%SYMDEL MULTINTEGER3;  
  
%mend;  
  
%setPromptValues();  
  
/*----- End SAS generated prompt variable code.----- */  
  
/*----- SAS generated prompt variable cleanup code.----- */  
%cleanupPromptValues();  
/*----- End SAS generated prompt variable cleanup code.----- */  
  
NOTE: Generating node connectors.  
NOTE: Linking nodes connected by data nodes and eliminating data nodes.  
  
Process Flow Node Summary
```

Display 137 - %SYMDEL Statements Remove the MULTINTEGER* Macro Variables

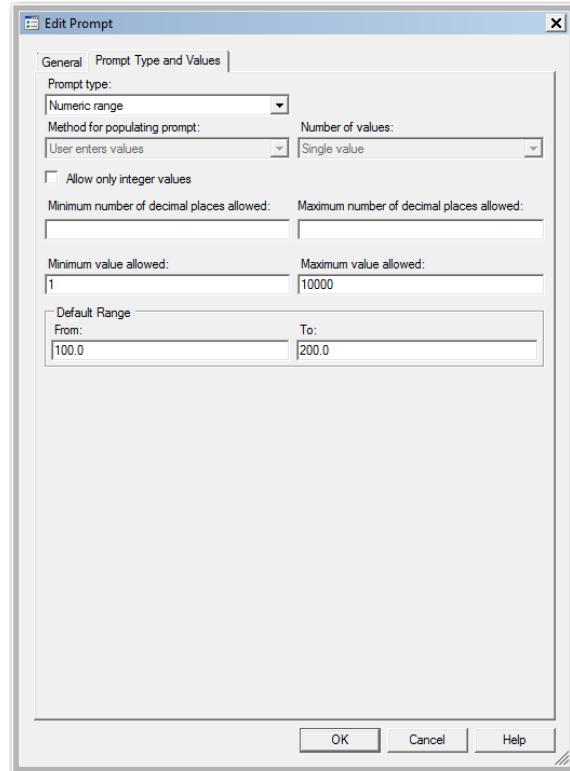
Numeric Range

SAS Enterprise Guide

In this example, a numeric range prompt named NumRange is defined as shown in the following two displays.

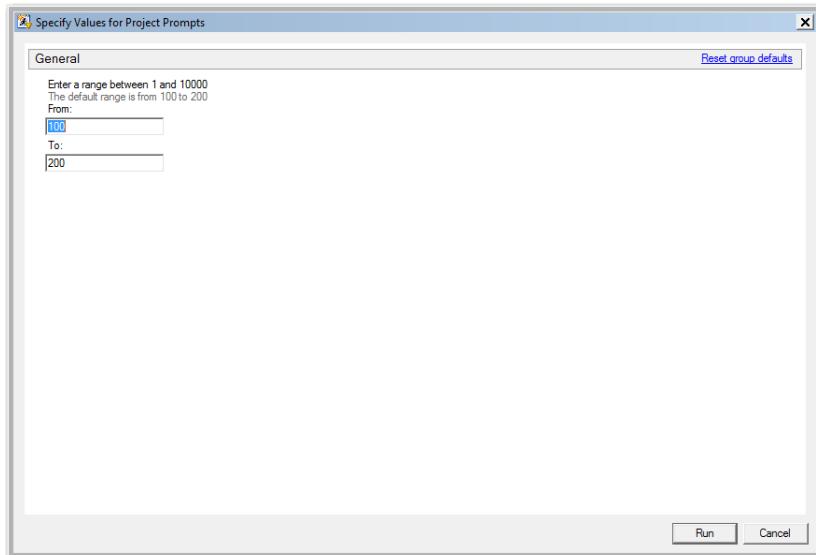


Display 138 - General Properties for Numeric Range Prompt



Display 139 - Type and Values for Numeric Range Prompt

When you run the Program node that depends on this prompt, the following dialog box appears.

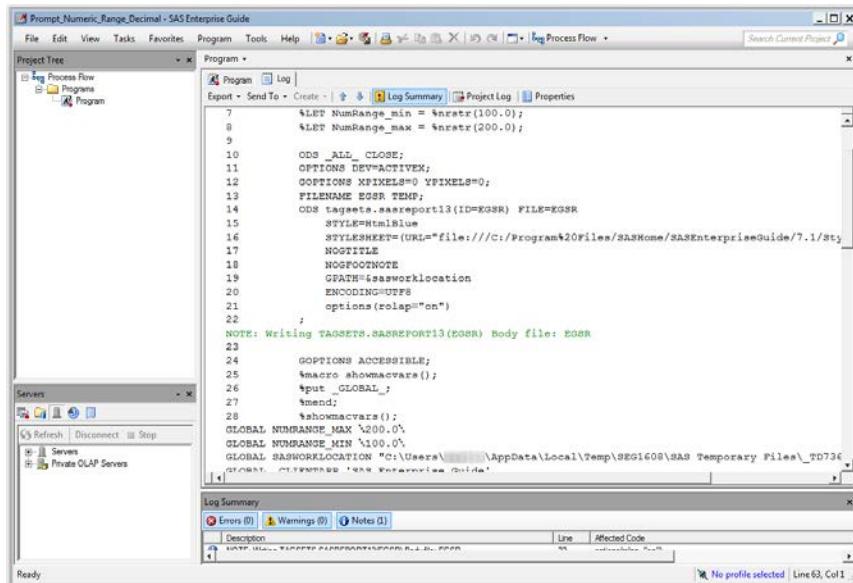


Display 140 - Numeric Range Prompt in Prompt Dialog Box

If the user leaves the default values in the numeric range value prompt fields, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The %LET statements assign the values specified in the prompt dialog box to the NumRange* macro variables.

The log of the [Program node using the prompt definition](#) displays the values of the global variables created by the prompt.



The screenshot shows the SAS Enterprise Guide interface with a 'Program' node selected in the Project Tree. The main pane displays the following SAS code:

```

7  %LET NumRange_min = %nrstr(100.0);
8  %LET NumRange_max = %nrstr(200.0);
9
10 ODS _ALL_ CLOSE;
11 OPTIONS DEVTIVICK;
12 OPTIONS XPIXELS=0 YPIXELS=0;
13 FILENAME EGSR TEMP;
14 ODS tagsets.sasreport13(ID=EGSR) FILE=EGSR
15   STYLE=RHTMLBlue
16   STYLESHEET=(URL="file:///C:/Program%20Files/SASHome/SASEnterpriseGuide/7.1/Sty
17   NOGTITLE
18   NOFOOTNOTE
19   GPATH=%sasworklocation
20   ENCODING=UTF8
21   options(rlabel="on")
22 ;
NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR
23
24 GOPTIONS ACCESSIBLE;
25 %macro showmacvars();
26 %put _GLOBAL_;
27 %mend;
28 %showmacvars();
GLOBAL NUMRANGE_MAX '200.0'
GLOBAL NUMRANGE_MIN '100.0'
GLOBAL SASWORKLOCATION "<:User\>\AppData\Local\Temp\S9E6109\SAS Temporary Files\_TD734
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43

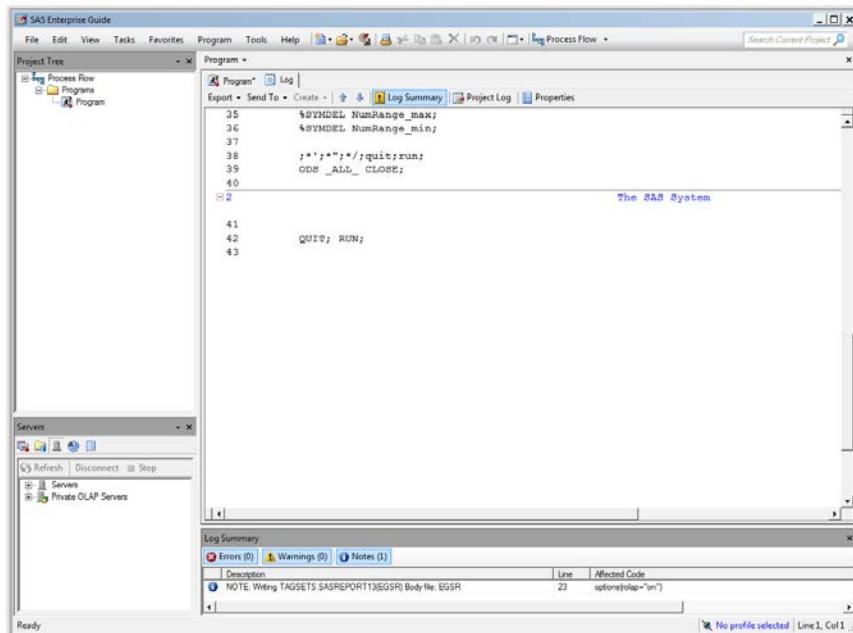
```

The Log Summary window below shows one note message:

Description	Line	Affected Code
NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR	23	options(rlabel="on")

Display 141 - Global Macro Variables and %LET Statements for Numeric Range Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.



The screenshot shows the SAS Enterprise Guide interface with a 'Program' node selected in the Project Tree. The main pane displays the following SAS code:

```

35  %SYMDEL NumRange_max;
36  %SYMDEL NumRange_min;
37
38  ;*:*/*:*/quit;xrun;
39  ODS _ALL_ CLOSE;
40
41
42  QUIT; RUN;
43

```

The Log Summary window below shows one note message:

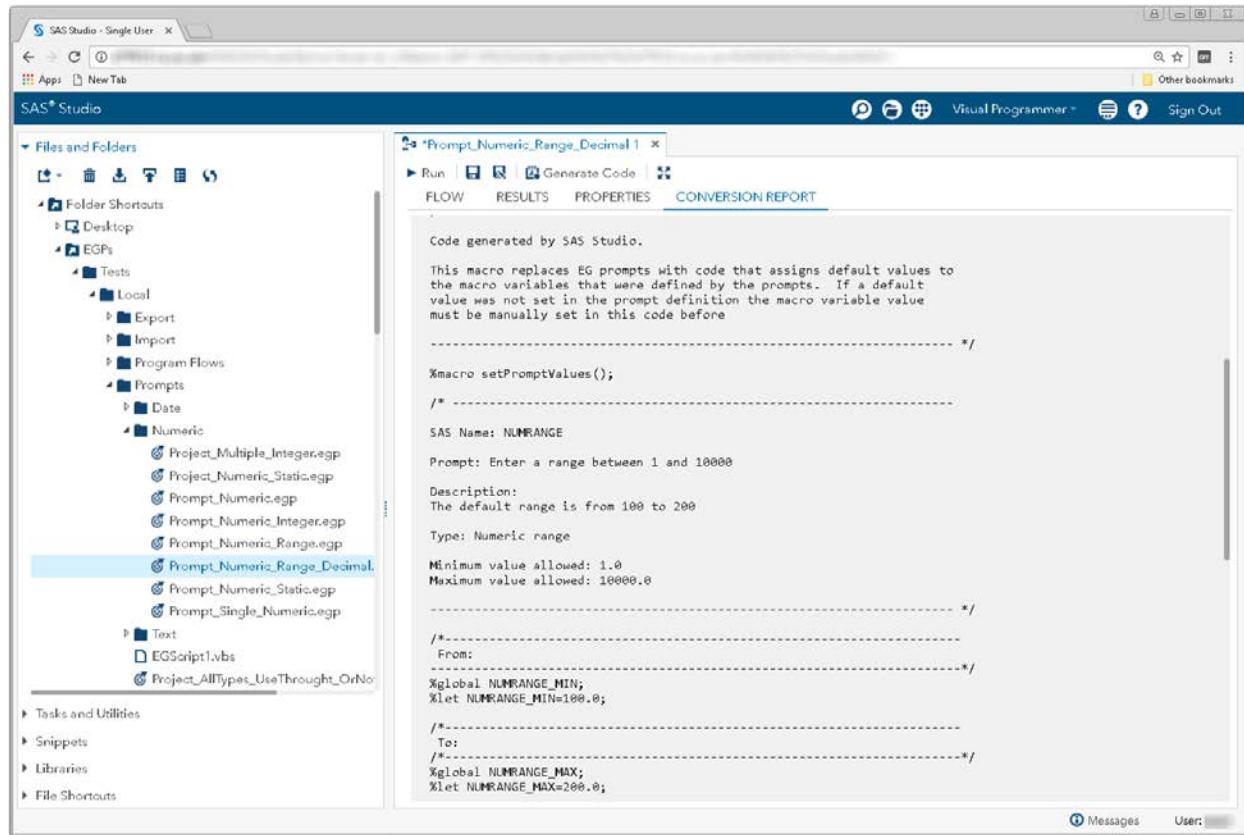
Description	Line	Affected Code
NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR	23	options(rlabel="on")

Display 142 - %SYMDEL Statements Remove NumRange* Macro Variables

SAS Studio

The following display shows code that is added to the converted Program node for the numeric range prompt in SAS Enterprise Guide.

Global variables named NUMRANGE_MIN and NUMRANGE_MAX are created. The %LET statements assign the default values to NUMRANGE_MIN and NUMRANGE_MAX. If you want to run your process flow using different values for the NUMRANGE prompt, you must manually update the values in the macro variables in the %LET statement.



The screenshot shows the SAS Studio interface. On the left, the 'Files and Folders' sidebar lists various EGP files, including 'Prompt_Numeric_Range.Decimal.1'. The main area is titled 'Prompt_Numeric_Range.Decimal.1' and contains a code editor with the following content:

```
Code generated by SAS Studio.

This macro replaces EG prompts with code that assigns default values to
the macro variables that were defined by the prompts. If a default
value was not set in the prompt definition the macro variable value
must be manually set in this code before

/*
 *macro setPromptValues();
 */

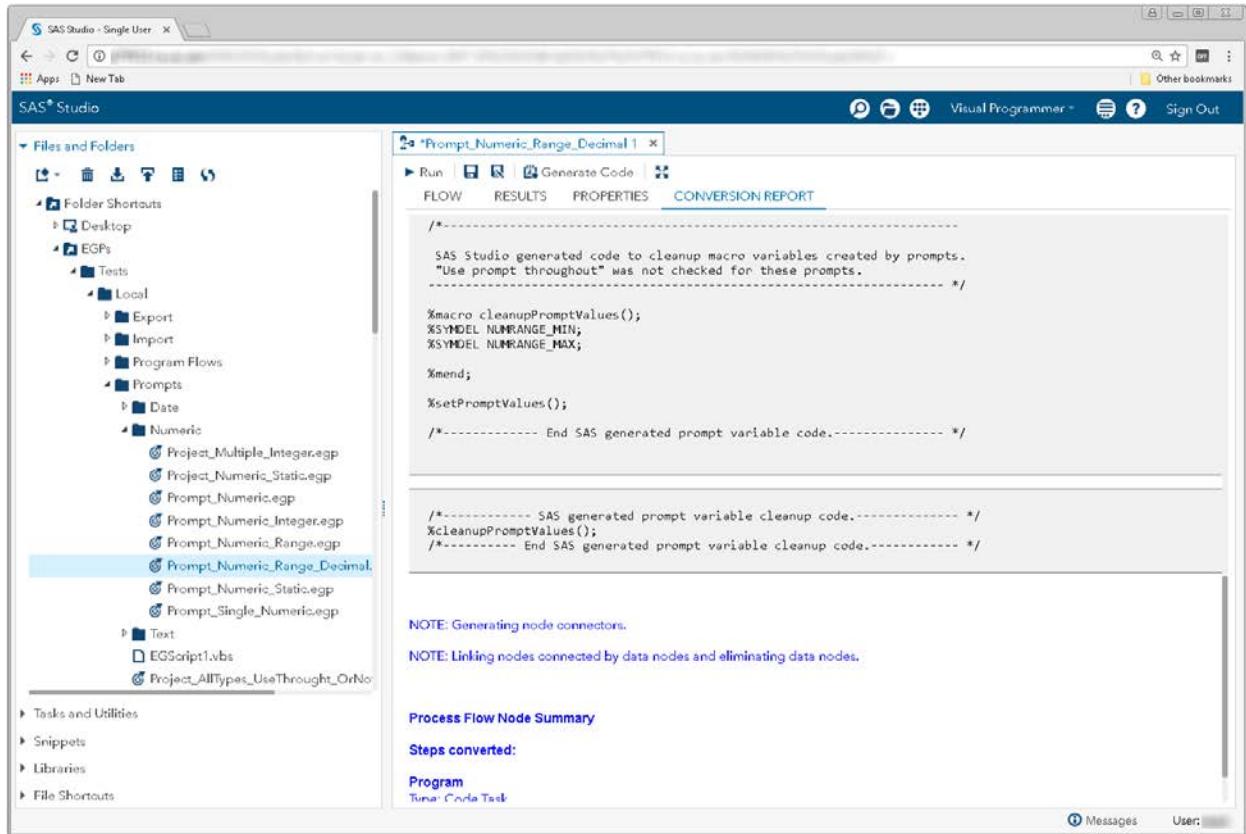
SAS Name: NUMRANGE
Prompt: Enter a range between 1 and 10000
Description:
The default range is from 100 to 200
Type: Numeric range
Minimum value allowed: 1.0
Maximum value allowed: 10000.0

/*
From:
*/
%global NUMRANGE_MIN;
%let NUMRANGE_MIN=100.0;

/*
To:
*/
%global NUMRANGE_MAX;
%let NUMRANGE_MAX=200.0;
```

Display 143 – Code for Numeric Range Prompt

Because the **Use prompt value throughout project** option is not checked, the %SYMDEL statements remove the NUMRANGE* macro variables.



The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. On the left, the Files and Folders tree view shows a folder structure with several EGP files, including "Prompt_Numeric_Range_Decimal1.egp". The main pane displays a "CONVERSION REPORT" for this file. The report contains the following code:

```
/*
-----  
SAS Studio generated code to cleanup macro variables created by prompts.  
"Use prompt throughout" was not checked for these prompts.  
----- */  
  
%macro cleanupPromptValues();  
%SYMDEL NUMRANGE_MIN;  
%SYMDEL NUMRANGE_MAX;  
  
%mend;  
  
%setPromptValues();  
/*----- End SAS generated prompt variable code.----- */  
  
/*----- SAS generated prompt variable cleanup code.----- */  
%cleanupPromptValues();  
/*----- End SAS generated prompt variable cleanup code.----- */  
  
NOTE: Generating node connectors.  
NOTE: Linking nodes connected by data nodes and eliminating data nodes.
```

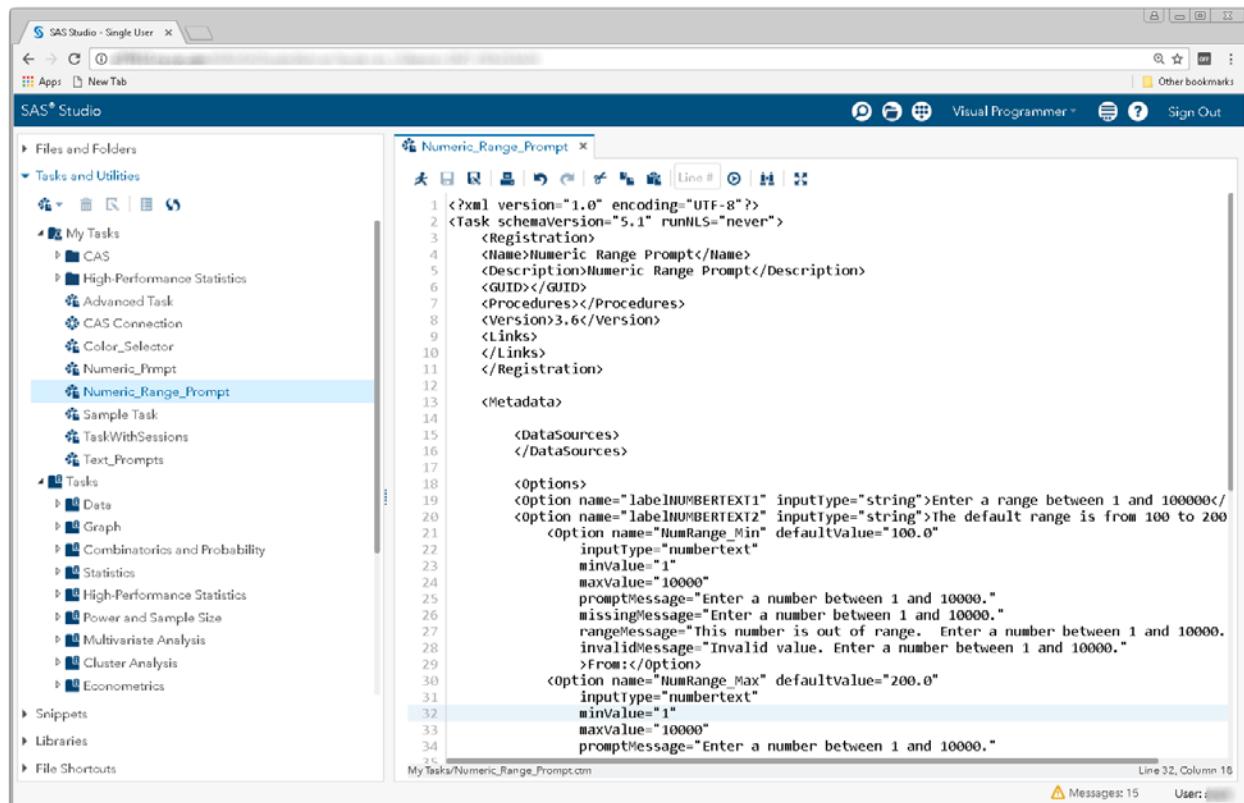
Below the code, there are sections for "Process Flow Node Summary" and "Steps converted:", both currently empty. At the bottom, it says "Program" and "Turns: Create Task".

Display 144 - %SYMDEL Statements Remove NUMRANGE* Macro Variable

Note: There is a SAS Studio bug that causes numeric range prompt information to not be processed when the **Allow only integer values** option is checked.

Substituting a SAS Studio Task a for Numeric Range Prompt

1. Create a SAS Studio task with a control that represents the prompt for a NumRange.
 - Add labels and two numbertext input controls. For the names of the input controls, use the prompt names of NumRange_Min and NumRange_Max.
 - Set the default values to the default values shown in the generated setPromptValues() macro in the converted Program node.
 - Change the strings of the input controls to match the strings specified in the prompt.



The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. On the left, the file tree shows a folder named 'My Tasks' containing several items, with 'Numeric_Range_Prompt' highlighted. The main pane displays the XML code for this task:

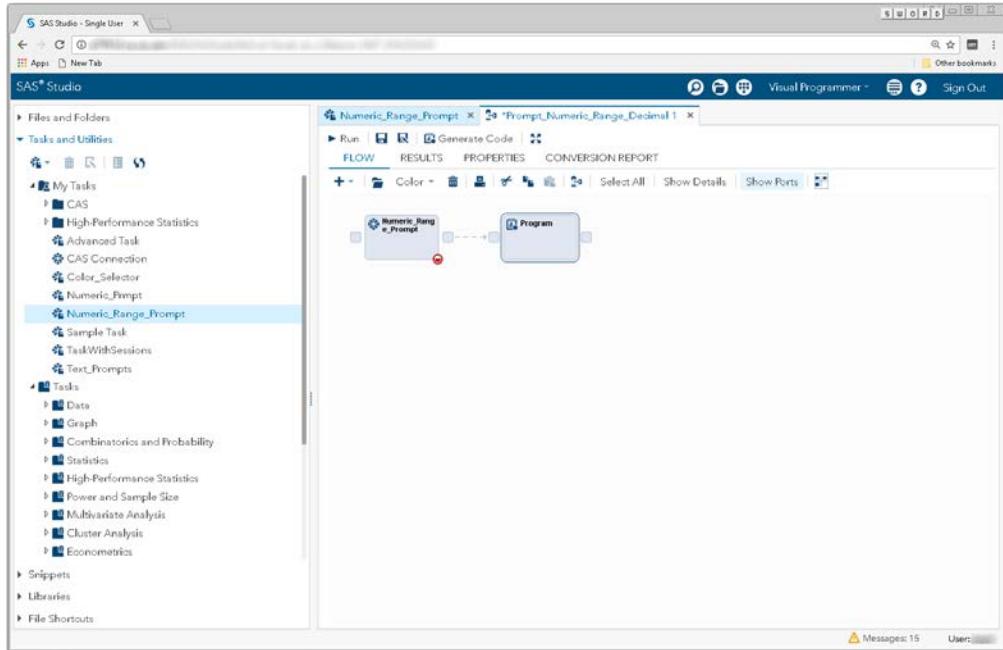
```
<?xml version="1.0" encoding="UTF-8"?>
<Task schemaVersion="5.1" runNLS="never">
  <Registration>
    <Name>Numeric Range Prompt</Name>
    <Description>Numeric Range Prompt</Description>
    <GUID></GUID>
    <Procedures></Procedures>
    <Version>3.6</Version>
    <Links>
    </Links>
  </Registration>
  <Metadata>
    <DataSources>
    </DataSources>
  </Metadata>
  <Options>
    <Option name="labelNUMBERTEXT1" inputType="string">Enter a range between 1 and 100000</Option>
    <Option name="labelNUMBERTEXT2" inputType="string">The default range is from 100 to 200</Option>
    <Option name="NumRange_Min" defaultValue="100.0" inputType="numbertext" minValue="1" maxValue="100000" promptMessage="Enter a number between 1 and 100000." missingMessage="Enter a number between 1 and 100000." rangeMessage="This number is out of range. Enter a number between 1 and 100000." invalidMessage="Invalid value. Enter a number between 1 and 100000.">From:</Option>
    <Option name="NumRange_Max" defaultValue="200.0" inputType="numbertext" minValue="1" maxValue="100000" promptMessage="Enter a number between 1 and 100000.">To:</Option>
  </Options>
</Task>
```

Display 145 - Replacement Task for Numeric Range Prompt

The following code is an example of a task that could be used to replace the numeric range prompt.

```
<?xml version="1.0" encoding="UTF-8"?>
<Task schemaVersion="5.1" runNLS="never">
    <Registration>
        <Name>Numeric Range Prompt</Name>
        <Description>Numeric Range Prompt</Description>
        <GUID></GUID>
        <Procedures></Procedures>
        <Version>3.6</Version>
        <Links></Links>
    </Registration>
    <Metadata>
        <DataSources>
        </DataSources>
        <Options>
            <Option name="labelNUMBERTEXT1" inputType="string">
                Enter a range between 1 and 100000
            </Option>
            <Option name="labelNUMBERTEXT2" inputType="string">
                The default range is from 100 to 200
            </Option>
            <Option name="NumRange_Min" defaultValue="100.0"
                    inputType="numbertext"
                    minValue="1"
                    maxValue="10000"
                    promptMessage="Enter a number between 1 and 10000."
                    missingMessage="Enter a number between 1 and 10000."
                    rangeMessage=
                    "This number is out of range. Enter a number between 1 and 10000."
                    invalidMessage=
                    "Invalid value. Enter a number between 1 and 10000.">
                From:
            </Option>
            <Option name="NumRange_Max" defaultValue="200.0"
                    inputType="numbertext"
                    minValue="1"
                    maxValue="10000"
                    promptMessage="Enter a number between 1 and 10000."
                    missingMessage="Enter a number between 1 and 10000."
                    rangeMessage=
                    "This number is out of range. Enter a number between 1 and 10000."
                    invalidMessage=
                    "Invalid value. Enter a number between 1 and 10000.">
                To:
            </Option>
        </Options>
    </Metadata>
    <UI>
        <OptionItem option="labelNUMBERTEXT1"/>
        <OptionItem option="labelNUMBERTEXT2"/>
        <OptionItem option="NumRange_Min"/>
        <OptionItem option="NumRange_Max"/>
    </UI>
    <CodeTemplate>
        <! [ CDATA[
%global NumRange_Min;
%let NumRange_Min=$NumRange_Min;
%global NumRange_Max;
%let NumRange_Max=$NumRange_Max;
        ] ]>
    </CodeTemplate>
</Task>
```

2. Save the prompt replacement task to your **My Tasks** folder.
3. Drag the task from **My Tasks** into your converted process flow.
4. Link the output port of the task to the input port of the converted Program node.



Display 146 – Numeric Range Input Task Linked to Program Node

5. Comment out the `%setPromptValues()` macro call from the converted Program node. The macro code generated by the numeric range input task replaces this code.

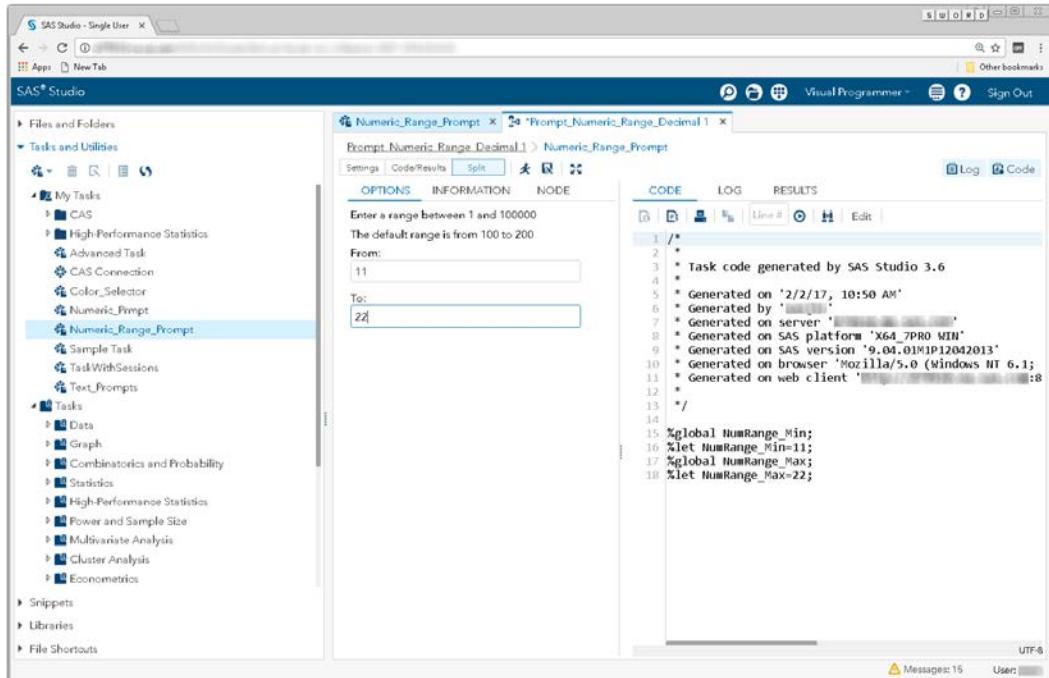
```

39 %global NUMRANGE_MAX;
40 %let NUMRANGE_MAX=200.0;
41
42 %xend;
43
44 /**
45 SAS Studio generated code to cleanup macro variables created by prompts.
46 *use prompt throughout* was not checked for these prompts.
47 */
48 %macro cleanupPromptvalues();
49 %SYMPDEL NUMRANGE_MIN;
50 %SYMPDEL NUMRANGE_MAX;
51
52 %xend;
53
54 /*----- End SAS generated prompt variable code.-----*/
55 %setPromptValues();
56
57 /*----- End SAS generated prompt variable code.-----*/
58 %macro showmacvars();
59 %put _GLOBAL_;
60 %xend;
61 %showmacvars();
62
63 /*----- SAS generated prompt variable cleanup code.-----*/
64 %cleanupPromptvalues();
65 /*----- End SAS generated prompt variable cleanup code.-----*/
66
67 /*----- SAS generated prompt variable cleanup code.-----*/
68 %cleanupPromptvalues();
69 /*----- End SAS generated prompt variable cleanup code.-----*/

```

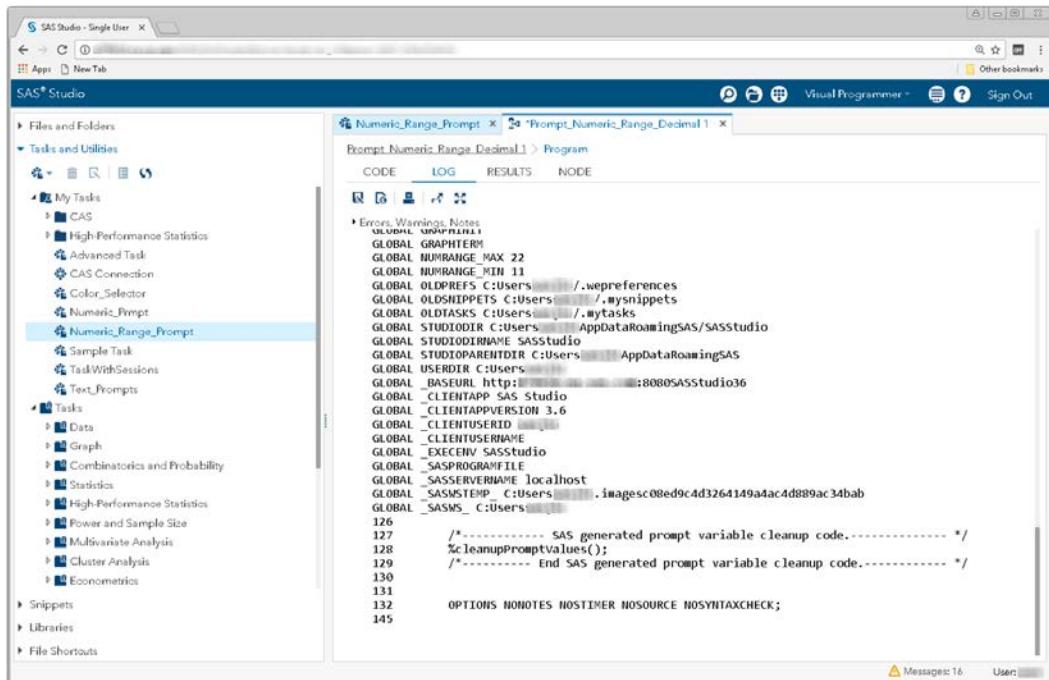
Display 147 - Commented Out `%setPromptValues` Macro Call

To run your flow with a different numeric value than the default value, open the Numeric_Range_Prompt node and specify a different numeric range.



Display 148 - Running the Numeric Range Input Task

When you run the process flow, the global NumRange* variables are set to the values specified in the task.

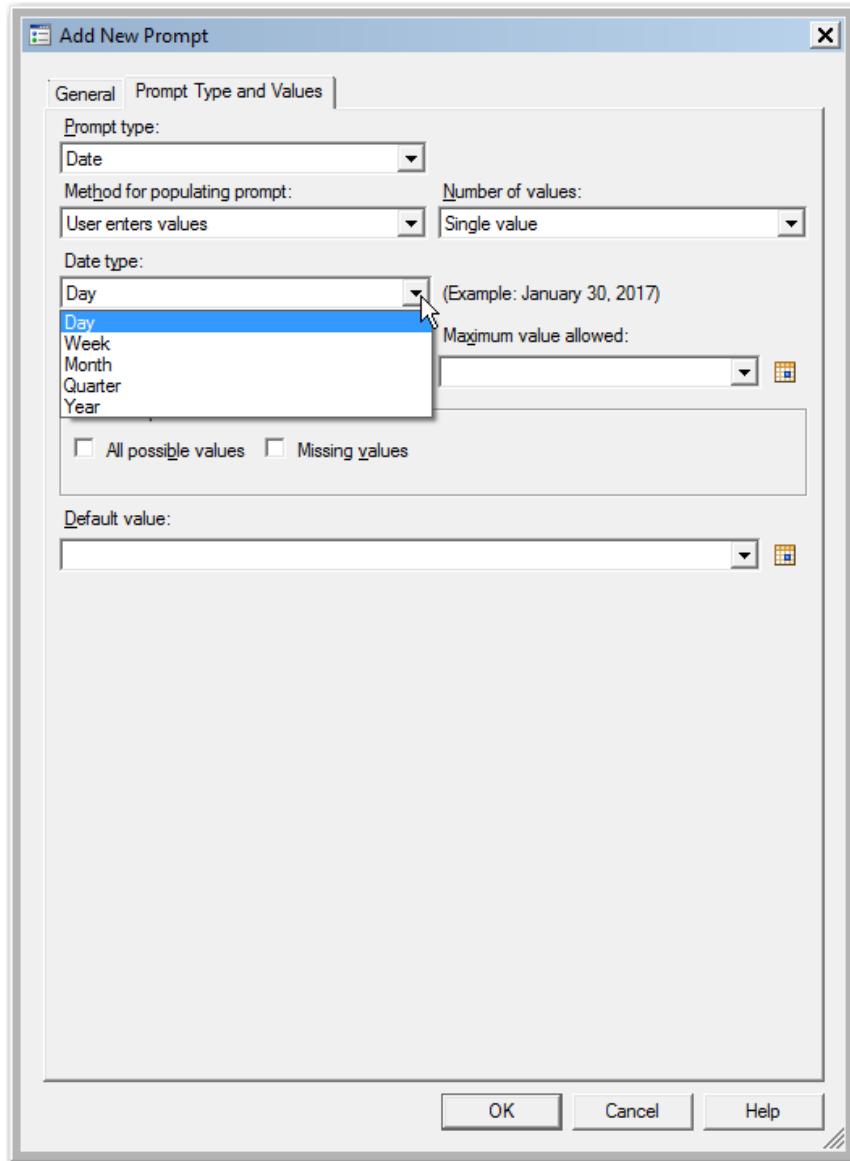


Display 149 – Numeric Range Prompt Variables with Updated Values

Date

For Date prompts, SAS Enterprise Guide supports many different types of dates including:

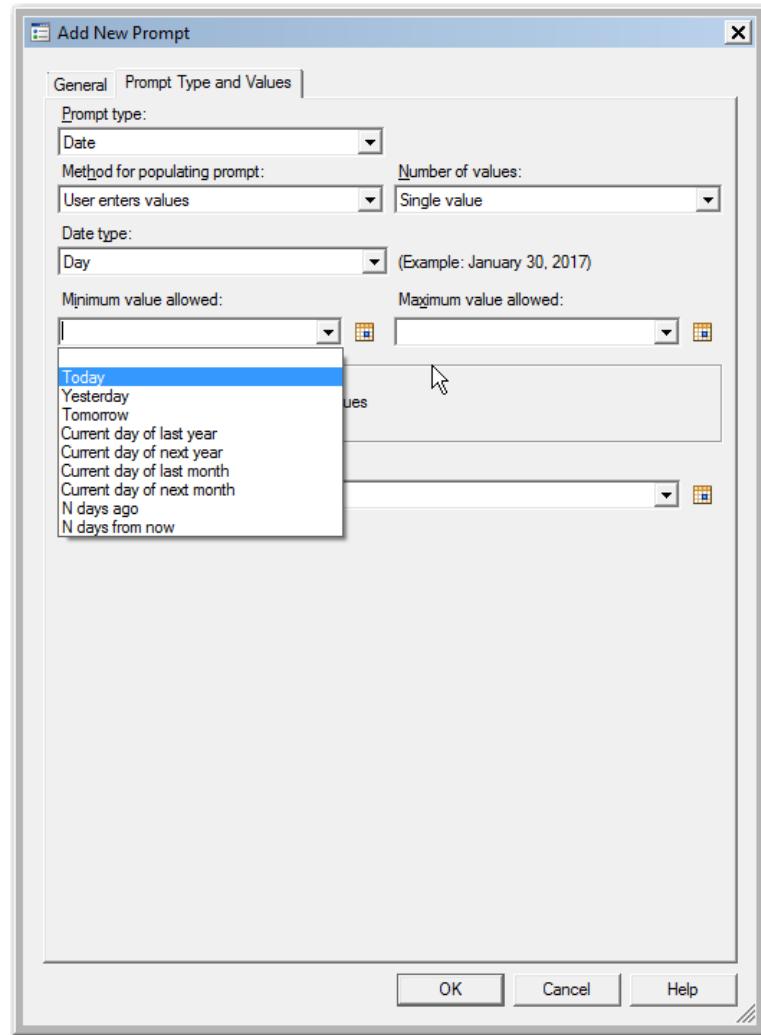
- Day
- Week
- Month
- Quarter
- Year



Display 150 - Types of Date Prompts

Explanation of Date Macro Variable

For each of the date types, you can set defaults, minimums, and maximums in the definitions. In the prompt dialog box, you can also specify absolute values or common values such as "Today", "Tomorrow", and so on.



Display 151 – Common Values for Dates

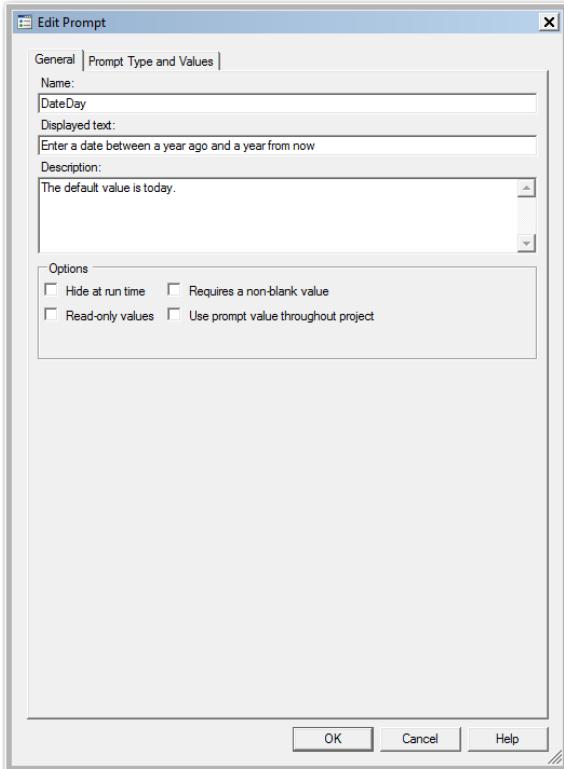
Macro variables generated by SAS Enterprise Guide for date prompts include the basic prompt name, name_rel, and name_label. If the prompt is a range, there is also a name_end macro variable.

- The name macro variable will contain the absolute date specified or the date of the common value, such as 30Jan2017
If the prompt is a period type prompt (such as Quarter), this value will be the beginning of the period. For example, "First Quarter" would have a date of 01Jan2017.
- The name_rel macro variable will contain a code for the common value. For example, "D0D" represents "Today".
- The name_label macro variable will contain the common value such as "Today" or "Tomorrow".
- The name_end macro variable will contain the date for the end of the period. For example, for "First Quarter" the end macro variable would have a date of 30Mar2017.

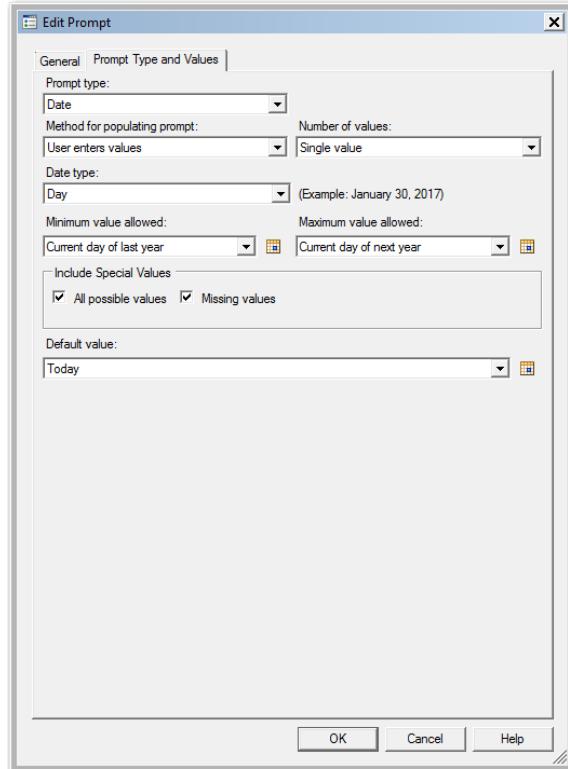
Single Date

SAS Enterprise Guide

In this example, a single date prompt named DateDay is defined as shown in the following two displays.

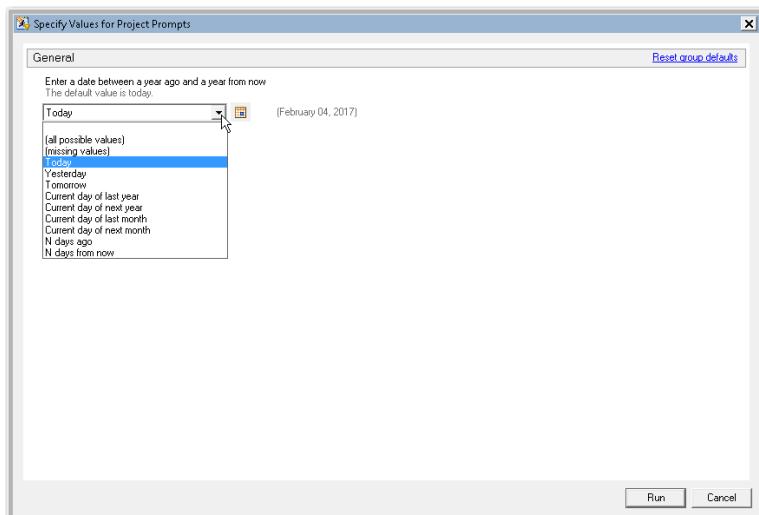


Display 152 - General Properties for Date Prompt



Display 153 - Type and Values for Date Prompt

When you run the Program node that depends on the prompt, the following prompt dialog box appears.



Display 154 - Date Prompt in Prompt Dialog Box

If the user leaves the default value in the date prompt field, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The %LET statements assign the value specified in the prompt dialog box to the DateDay* macro variables.

The log of the [Program node using the prompt definition](#) displays the value of the global variable created by the prompt.

```

7      %LET DateDay_label = %nrstr(Today);
8      %LET DateDay_rel = %nrstr(DOD);
9      %LET DateDay = %nrstr(30Jan2017);
10
11      ODS _ALL_CLOSE;
12      OPTIONS DEVMETHOD=VEX;
13      GOPTIONS XPIXELS=0 YPIXELS=0;
14      FILENAME EGSR TEMP;
15      ODS TAGSETS.SASREPORT13(ID=EGSR) FILE=EGSR
16          TITLE="HtmlBlue"
17          STYLESHEET=(URL="file:///c:/Program420Files/SASHome/SASEnterpriseGuide/7.1/sty
18          NOTITLE
19          NOFOOTNOTE
20          GPATH=&seaworklocation
21          ENCODING=UTF8
22          options(rolatile="on")
23      );
24
25      GOPTIONS ACCESSIBLE;
26      %macro showmacvars();
27      %put _GLOBAL_;
28      %mend;
29
30      %showmacvars();
31      GLOBAL DATEDAY '30JAN2017';
32      GLOBAL DATEDAY_REL 'DOD';
33      GLOBAL DASHBLOCATION "C:\Users\...\AppData\Local\Temp\SEG13096\SAS Temporary Files\_TDIC
34      GLOBAL _CLIENTAPP 'SAS Enterprise Guide'
35
36      %SYMDEL DateDay_label;
37      %SYMDEL DateDay_rel;
38      %SYMDEL DateDay;
39
40      /* */;/* */quit;run;
41      ODS _ALL_ CLOSE;
42
43
44      QUIT; RUN;
45

```

Display 155 - %LET Statements for Single Date Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.

```

36      %SYMDEL DateDay_label;
37      %SYMDEL DateDay_rel;
38      %SYMDEL DateDay;
39
40      /* */;/* */quit;run;
41      ODS _ALL_ CLOSE;
42
43
44      QUIT; RUN;
45

```

Display 156 - %SYMDEL Statements Remove DateDay* Macro Variables

SAS Studio

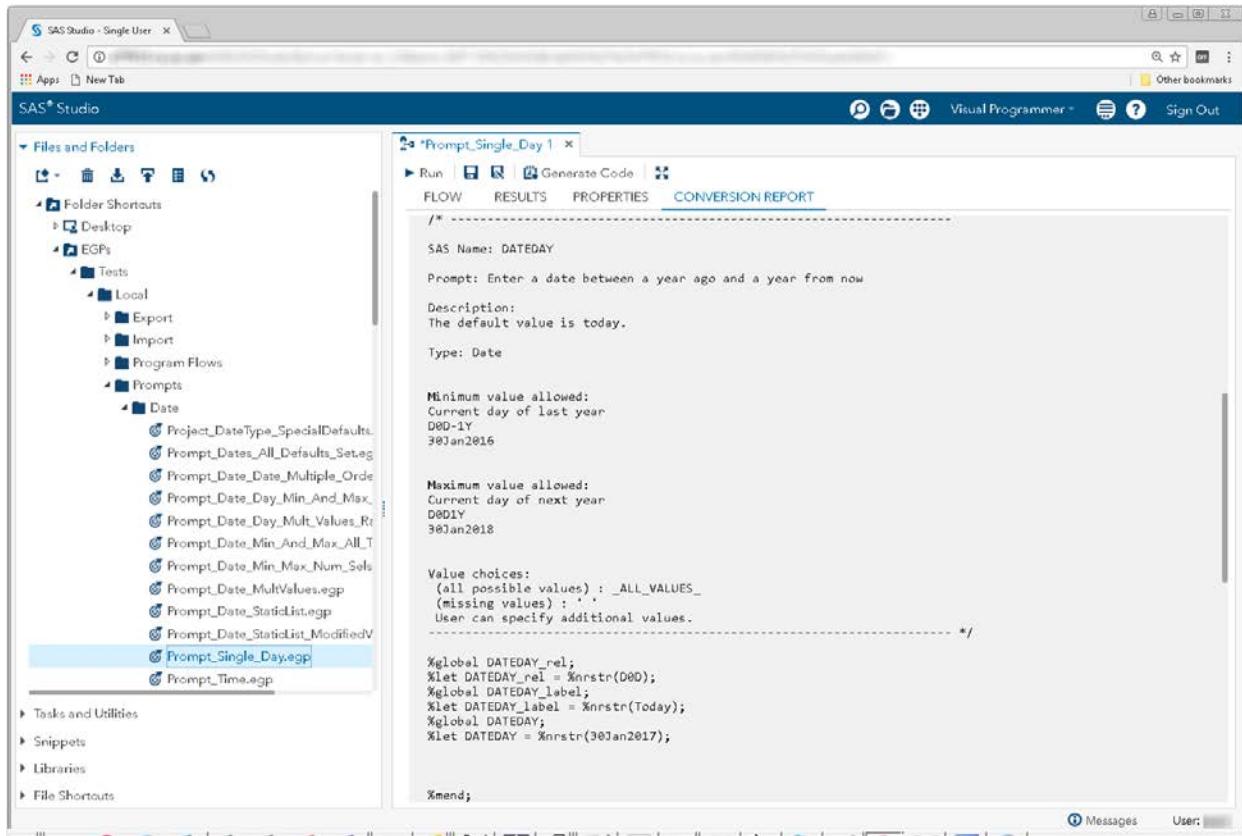
The following display shows the code that is added to the converted Program node for the date prompt in SAS Enterprise Guide.

These global [date macro variables](#) are created:

- DATEDAY_rel
- DATEDAY_label
- DATEDAY

The %LET statements assign the default value to the DATEDAY* macro variables.

If you want to run your process flow using different values for the DATEDAY prompt, you must manually update the values of the macro variables in the %LET statements.

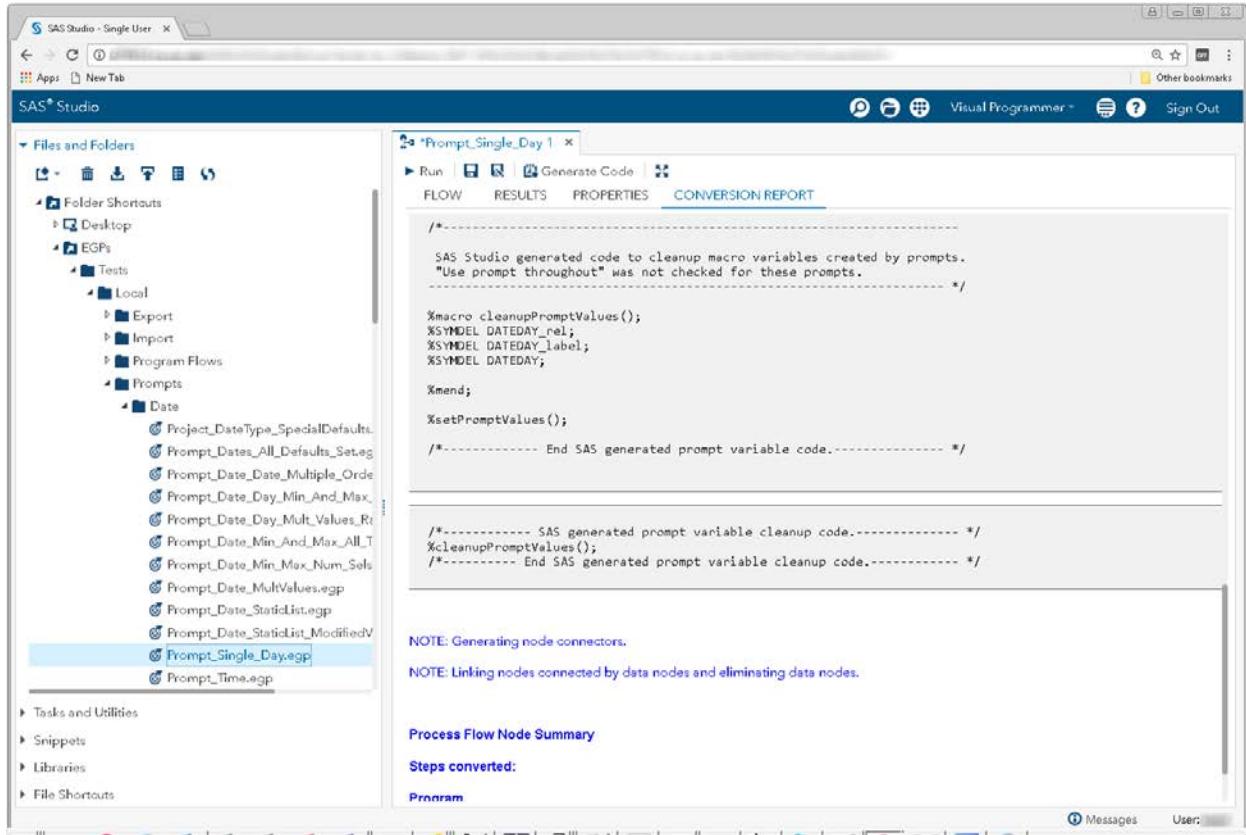


The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. On the left, the 'Files and Folders' tree view shows a folder structure under 'Local'. A file named 'Prompt_Single_Day.egp' is selected in the tree. The main workspace displays the following macro code:

```
/* -----  
SAS Name: DATEDAY  
  
Prompt: Enter a date between a year ago and a year from now  
  
Description:  
The default value is today.  
  
Type: Date  
  
Minimum value allowed:  
Current day of last year  
D0D-1Y  
30Jan2016  
  
Maximum value allowed:  
Current day of next year  
D0D1Y  
30Jan2018  
  
Value choices:  
(all possible values) : _ALL_VALUES_  
(missing values) : ''  
User can specify additional values.  
----- */  
  
%global DATEDAY_rel;  
%let DATEDAY_rel = %nrstr(D0D);  
%global DATEDAY_label;  
%let DATEDAY_label = %nrstr(Today);  
%global DATEDAY;  
%let DATEDAY = %nrstr(30Jan2017);  
  
%mend;
```

Display 157 - Macro Code for Date Prompt

Because the **Use prompt value throughout project** option is not checked, the %SYMDEL statements remove the DATEDAY macro variables.



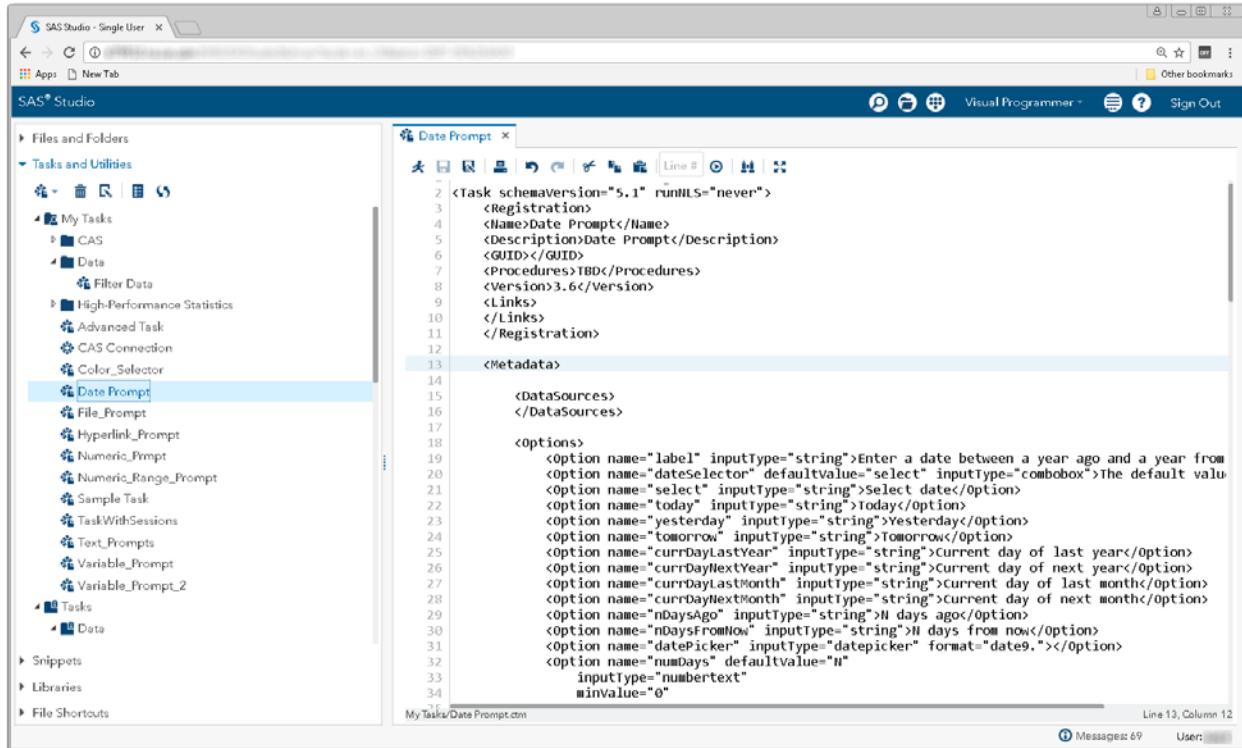
The screenshot shows the SAS Studio interface. On the left, the 'Files and Folders' browser displays a tree structure of local files and folders, including 'Local', 'Export', 'Import', 'Program Flows', 'Prompts', and 'Date' subfolders containing various EGPs. A file named 'Prompt_Single_Day.egp' is selected and highlighted in blue. On the right, the 'Visual Programmer' window is open, showing a code editor with the following content:

```
/*
-----  
SAS Studio generated code to cleanup macro variables created by prompts.  
"Use prompt throughout" was not checked for these prompts.  
----- */  
  
%macro cleanupPromptValues();  
%SYMDEL DATEDAY_rel;  
%SYMDEL DATEDAY_label;  
%SYMDEL DATEDAY;  
  
%mend;  
  
%setPromptValues();  
  
/*----- End SAS generated prompt variable code.----- */  
  
/*----- SAS generated prompt variable cleanup code.----- */  
%cleanupPromptValues();  
/*----- End SAS generated prompt variable cleanup code.----- */  
  
NOTE: Generating node connectors.  
NOTE: Linking nodes connected by data nodes and eliminating data nodes.  
  
Process Flow Node Summary  
Steps converted:  
Program
```

Display 158 - %SYMDEL Statements Remove the DATEDAY* Macro Variables

Substituting a SAS Studio Task for a Date Range Prompt

1. Create a SAS Studio task with a control that represents the Day type prompt for a Date.
 - Add controls as shown in the Date Prompt task.
 - Set the default value to the default value shown in the generated `setPromptValues()` macro in the converted Program node.
 - Change the strings for the input controls to match the strings specified in the prompt.



The screenshot shows the SAS Studio interface. On the left, the navigation pane displays 'Tasks and Utilities' with several task categories like My Tasks, CAS, Data, and a newly selected 'Date Prompt'. The main workspace is titled 'Date Prompt' and contains the XML code for a task definition. The code includes sections for registration, description, GUID, procedures, version, links, and metadata. It also defines data sources and a large set of options for date selection, including labels for 'today', 'yesterday', 'tomorrow', and various time intervals in days and months. The code is annotated with line numbers from 1 to 34. The bottom status bar indicates 'Line 10, Column 12'.

```
<Task schemaVersion="5.1" runMLS="never">
  <Registration>
    <Name>Date Prompt</Name>
    <Description>Date Prompt</Description>
    <GUID></GUID>
    <Procedures>TBD</Procedures>
    <Version>3.6</Version>
    <Links>
    </Links>
  </Registration>
  <Metadata>
    <DataSources>
    </DataSources>
  <options>
    <option name="label" inputType="string">Enter a date between a year ago and a year from now</option>
    <option name="dateSelector" defaultValue="select" inputType="checkbox">The default value is selected</option>
    <option name="select" inputType="string">Select date</option>
    <option name="today" inputType="string">Today</option>
    <option name="yesterday" inputType="string">Yesterday</option>
    <option name="tomorrow" inputType="string">Tomorrow</option>
    <option name="currDayLastYear" inputType="string">Current day of last year</option>
    <option name="currDayNextYear" inputType="string">Current day of next year</option>
    <option name="currDayLastMonth" inputType="string">Current day of last month</option>
    <option name="currDayNextMonth" inputType="string">Current day of next month</option>
    <option name="nDaysAgo" inputType="string">N days ago</option>
    <option name="nDaysFromNow" inputType="string">N days from now</option>
    <option name="datePicker" inputType="datepicker" format="date9."></option>
    <option name="numdays" defaultValue="1" inputType="numberText" maxvalue="0" minvalue="0" type="text">1</option>
  </options>
</Metadata>

```

Display 159 - Replacement Task for Date Prompt

The following code is an example of a task that could be used as the date day prompt.

```
<?xml version="1.0" encoding="UTF-8"?>
<Task schemaVersion="5.1" runNLS="never">
    <Registration>
        <Name>Date Prompt</Name>
        <Description>Date Prompt</Description>
        <GUID></GUID>
        <Procedures>TBD</Procedures>
        <Version>3.6</Version>
        <Links></Links>
    </Registration>

    <Metadata>

        <DataSources> </DataSources>

        <Options>
            <Option name="label" inputType="string">
                Enter a date between a year ago and a year from now
            </Option>
            <Option name="dateSelector"
                    defaultValue="select"
                    inputType="combobox">
                The default value is today
            </Option>
            <Option name="select" inputType="string">Select date</Option>
            <Option name="today" inputType="string">Today</Option>
            <Option name="yesterday" inputType="string">Yesterday</Option>
            <Option name="tomorrow" inputType="string">Tomorrow</Option>
            <Option name="currDayLastYear" inputType="string">
                Current day of last year
            </Option>
            <Option name="currDayNextYear" inputType="string">
                Current day of next year
            </Option>
            <Option name="currDayLastMonth" inputType="string">
                Current day of last month
            </Option>
            <Option name="currDayNextMonth" inputType="string">
                Current day of next month
            </Option>
            <Option name="nDaysAgo" inputType="string">
                N days ago
            </Option>
            <Option name="nDaysFromNow" inputType="string">
                N days from now
            </Option>
            <Option name="datePicker" inputType="datepicker" format="date9.">
            </Option>
            <Option name="numDays" defaultValue="N"
                    inputType="numbertext"
                    minValue="0"
                    promptMessage="Enter number of days"
                    missingMessage="Enter number of days"
                    rangeMessage=
                        "This number is out of range. Enter a positive number."
                    invalidMessage="Invalid value. Enter a positive integer.">
                Number of days:
            </Option>
        </Options>
    </Metadata>
```

```

<UI>
    <OptionItem option="label"/>
    <OptionChoice option="dateSelector">
        <OptionItem option="select"/>
        <OptionItem option="today"/>
        <OptionItem option="yesterday"/>
        <OptionItem option="tomorrow"/>
        <OptionItem option="currDayLastYear"/>
        <OptionItem option="currDayNextYear"/>
        <OptionItem option="currDayLastMonth"/>
        <OptionItem option="currDayNextMonth"/>
        <OptionItem option="nDaysAgo"/>
        <OptionItem option="nDaysFromNow"/>
    </OptionChoice>

    <OptionItem option="datePicker"/>
    <OptionItem option="numDays"/>
</UI>
<Dependencies>
    <Dependency condition="($dateSelector == 'select')">
        <Target action="hide" conditionResult="false" option="datePicker"/>
        <Target action="show" conditionResult="true" option="datePicker"/>
    </Dependency>
    <Dependency condition=
        "((($dateSelector == 'nDaysAgo') || ($dateSelector == 'nDaysFromNow')))">
        <Target action="hide" conditionResult="false" option="numDays"/>
        <Target action="show" conditionResult="true" option="numDays"/>
    </Dependency>
</Dependencies>

<CodeTemplate>
    <![CDATA[

%global dateday;
%global dateday_rel;
%global dateday_label;

#if ($dateSelector == 'select')
%let dateday = $datePicker;
%let dateday_label = %sysfunc(inputn(&dateday, date9.), WORDDATE20.);
%symdel dateday_rel;
#endif

#if ($dateSelector == 'today')
%let dateday = %sysfunc(intnx(Day,%sysfunc(date()),0), DATE9.);
%let dateday_label = Today;
%let dateday_rel = D0D;
#endif

#if ($dateSelector == 'tomorrow')
%let dateday = %sysfunc(intnx(Day,%sysfunc(date()),+1), DATE9.);
%let dateday_label = Tomorrow;
%let dateday_rel = D1D;
#endif

#if ($dateSelector == 'yesterday')
%let dateday = %sysfunc(intnx(Day,%sysfunc(date()),-1), DATE9.);
%let dateday_label = Yesterday;
%let dateday_rel = D-1D;
#endif

```

```

#if ($dateSelector == 'currDayLastYear')
%let dateday = %sysfunc(intnx(Year,%sysfunc(date()),-1,s), DATE9.);
%let dateday_label = Current day of last year;
%let dateday_rel = D-1Y;
#end

#if ($dateSelector == 'currDayNextYear')
%let dateday = %sysfunc(intnx(Year,%sysfunc(date()),+1,s), DATE9.);
%let dateday_label = Current day of next year;
%let dateday_rel = D1Y;
#end

#if ($dateSelector == 'currDayLastMonth')
%let dateday = %sysfunc(intnx(Month,%sysfunc(date()),-1,s), DATE9.);
%let dateday_label = Current day of last month;
%let dateday_rel = D-1M;
#end

#if ($dateSelector == 'currDayNextMonth')
%let dateday = %sysfunc(intnx(Month,%sysfunc(date()),+1,s), DATE9.);
%let dateday_label = Current day of next month;
%let dateday_rel = D1M;
#end

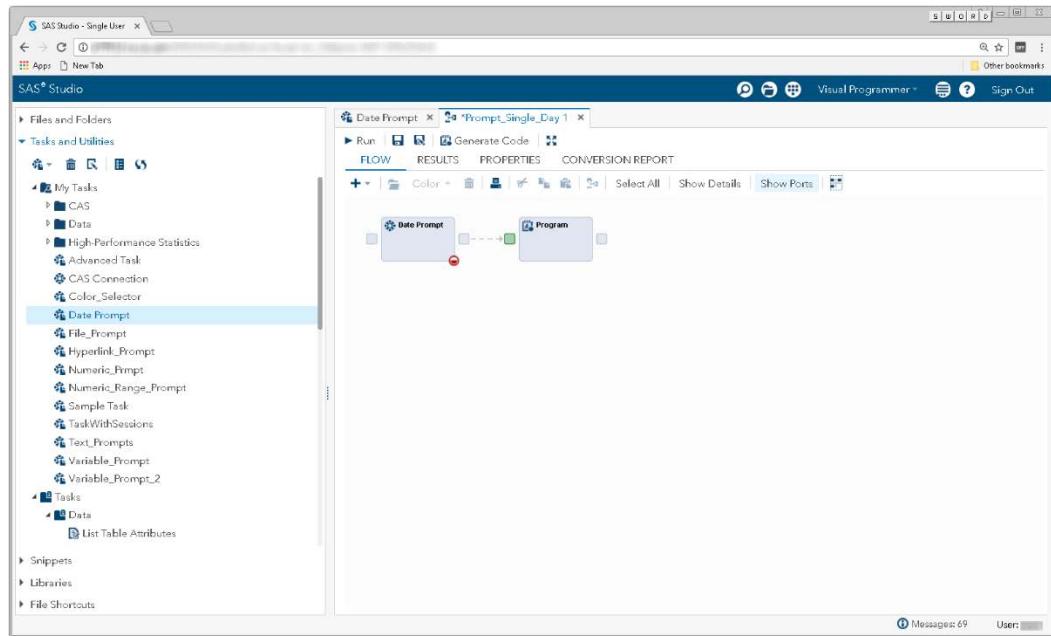
#if ($dateSelector == 'nDaysAgo')
%let dateday = %sysfunc(intnx(Day,%sysfunc(date()),-$numDays,s), DATE9.);
%let dateday_label = $numDays days ago;
%let dateday_rel = D-$numDaysD;
#end

#if ($dateSelector == 'nDaysFromNow')
%let dateday = %sysfunc(intnx(Day,%sysfunc(date()),+$numDays,s), DATE9.);
%let dateday_label = $numDays days from now;
%let dateday_rel = D$numDaysD;
#end

    ]]>
</CodeTemplate>
</Task>

```

2. Save the prompt replacement task to your **My Tasks** folder.
3. Drag the task from **My Tasks** into your converted process flow.
4. Link the output port of the new task to the input port of the converted Program node.



Display 160 – Date Prompt Task Linked to Program Node

5. Comment out the `%setPromptValues()` macro call from the converted Program node. The macro code generated by the Date Prompt input task replaces this code.

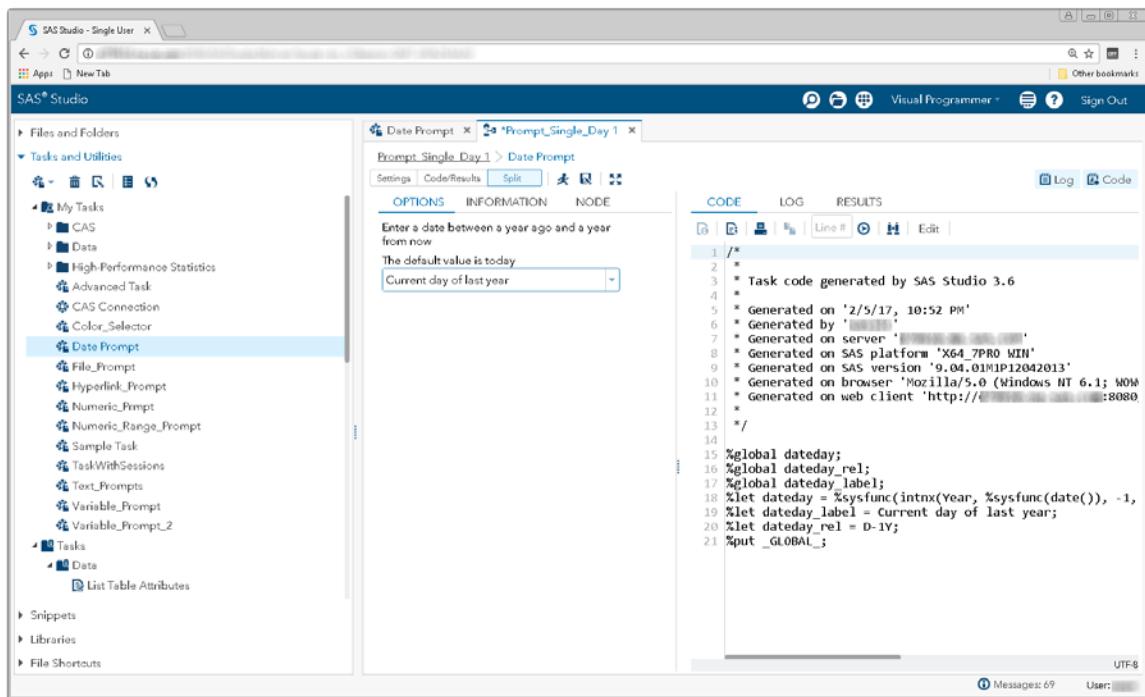
```

51 |
52 |
53 %end;
54 |
55 /*-----|
56 |
57 SAS studio generated code to cleanup macro variables created by prompts.
58 "Use prompt throughout" was not checked for these prompts.
59 -----*/
60 |
61 %macro cleanuppromptvalues();
62 %sympdel DATEDAY_rel;
63 %sympdel DATEDAY_label;
64 %sympdel DATEDAY;
65 |
66 %end;
67 |
68 /*%setPromptValues();*/
69 |
70 /*----- End SAS generated prompt variable code.-----*/
71 |
72 %macro showmacvars();
73 %put_GLOBAL_;
74 %end;
75 %showmacvars();
76 |
77 /*----- SAS generated prompt variable cleanup code.-----*/
78 %cleanuppromptvalues();
79 /*----- End SAS generated prompt variable cleanup code.-----*/
80

```

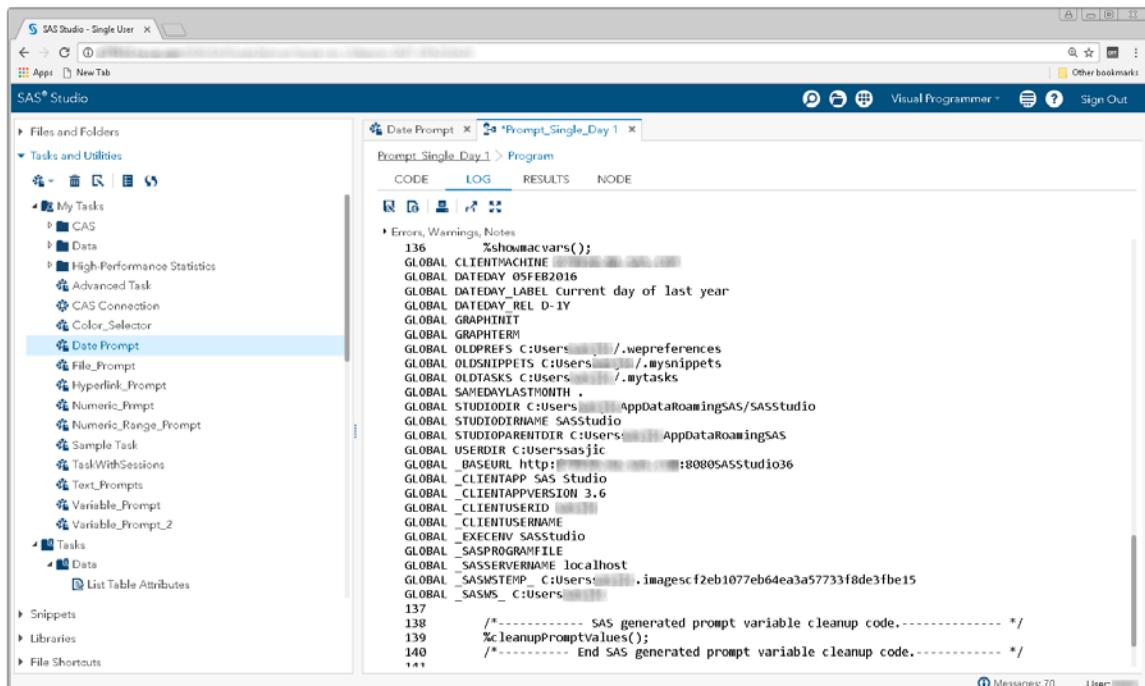
Display 161 – Commented out `%setPromptValues` in Program Node

To run your flow with a different date value than the default value, open the Date_Prompt node and specify a different date.



Display 162 – Task for Date Prompt

When you run the process flow, the global DateDay* variables are set to the value specified in the Date Prompt task.

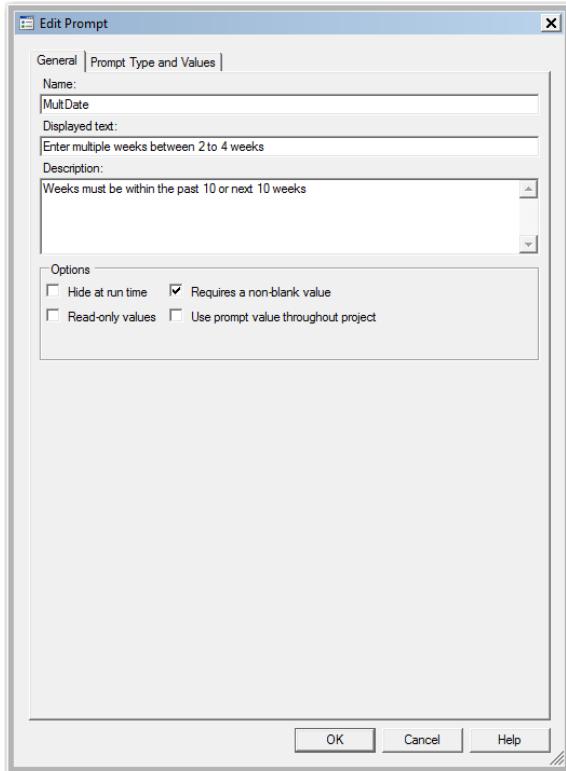


Display 163 – Updated Values for Variables in Date Prompt Task

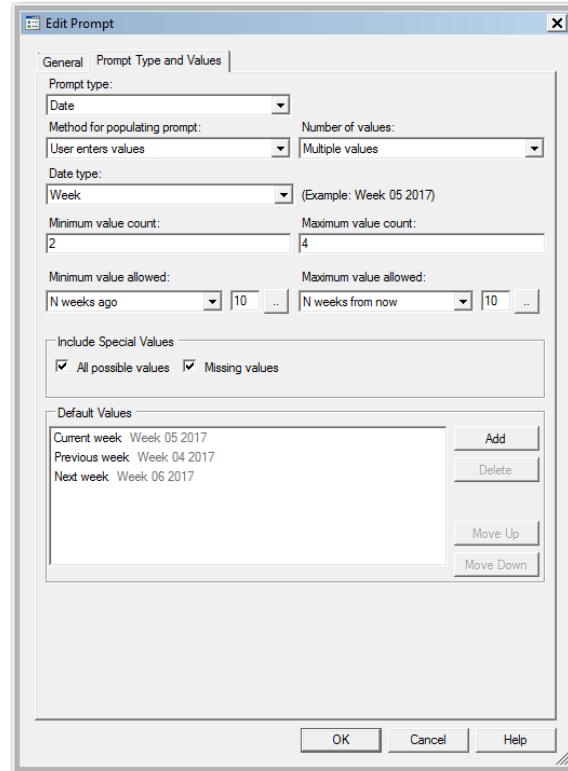
Multiple Dates

SAS Enterprise Guide

In this example, a multiple value date prompt named MultDate is defined as shown in the following two displays.

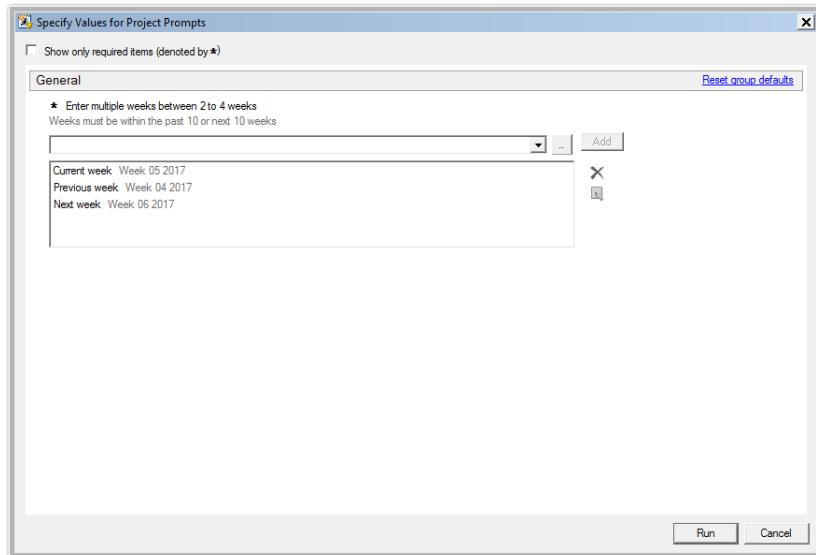


Display 164 - General Properties for Multiple Values Date Prompt



Display 165 - Type and Values for Multiple Values Date Prompt

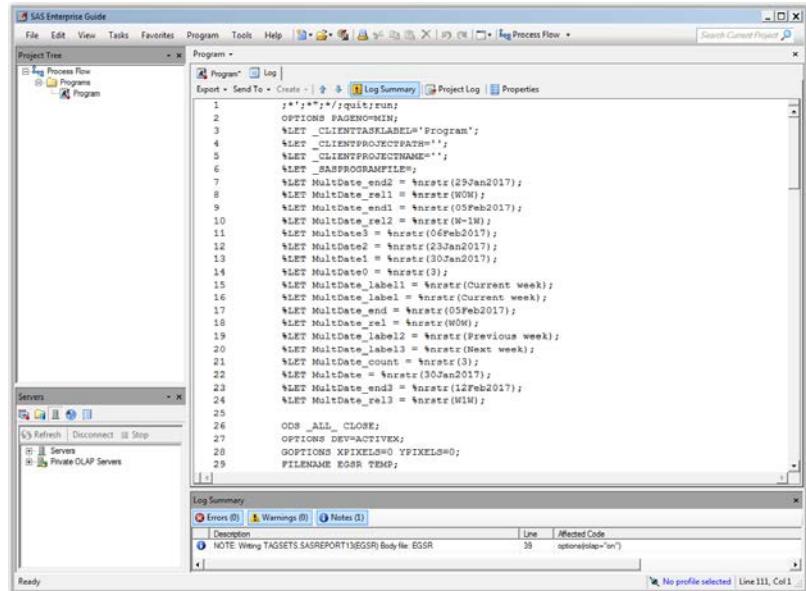
When you run the Program node that depends on the prompt, the following dialog box appears:



Display 166 - Multiple Value Date Prompt in Prompt Dialog Box

If the user leaves the default value in the multiple date prompt field, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The %LET statements assign the values specified in the prompt dialog box to the MultDate* macro variables.



```

SAS Enterprise Guide
File Edit View Tasks Favorites Program Tools Help <|> Process Flow Search Current Project

Project Tree
Programs Programs

Program Log
Program Log Summary Project Log Properties
1 /*+*/ quit;
2 OPTIONS PAGENO=MIN;
3 %LET _CLIENTTASKLABEL="Program";
4 %LET _CLIENTPROJECTNAME="";
5 %LET _CLIENTPROJECTFILE="";
6 %LET _SASPROGRAMFILE="";
7 %LET MultDate_end2 = %nrstr(29Jan2017);
8 %LET MultDate_rel1 = %nrstr(W0W);
9 %LET MultDate_end = %nrstr(05Feb2017);
10 %LET MultDate_end1 = %nrstr(04Feb2017);
11 %LET MultDate1 = %nrstr(06Feb2017);
12 %LET MultDate2 = %nrstr(23Jan2017);
13 %LET MultDate3 = %nrstr(30Jan2017);
14 %LET MultDate0 = %nrstr(3);
15 %LET MultDate_label1 = %nrstr(Current week);
16 %LET MultDate_label = %nrstr(Current week);
17 %LET MultDate_end = %nrstr(05Feb2017);
18 %LET MultDate_rel1 = %nrstr(W0W);
19 %LET MultDate_label2 = %nrstr(Previous week);
20 %LET MultDate_label3 = %nrstr(Next week);
21 %LET MultDate_count = %nrstr(0);
22 %LET MultDate_end0 = %nrstr(30Jan2017);
23 %LET MultDate_end1 = %nrstr(13Feb2017);
24 %LET MultDate_rel13 = %nrstr(W1W);
25

ODS_ALL CLOSE;
26 OPTIONS DEV=ACTIVE;
27 GOPTIONS XPIXELS=0 YPIXELS=0;
28 FILENAME EGSR TEMP;
29

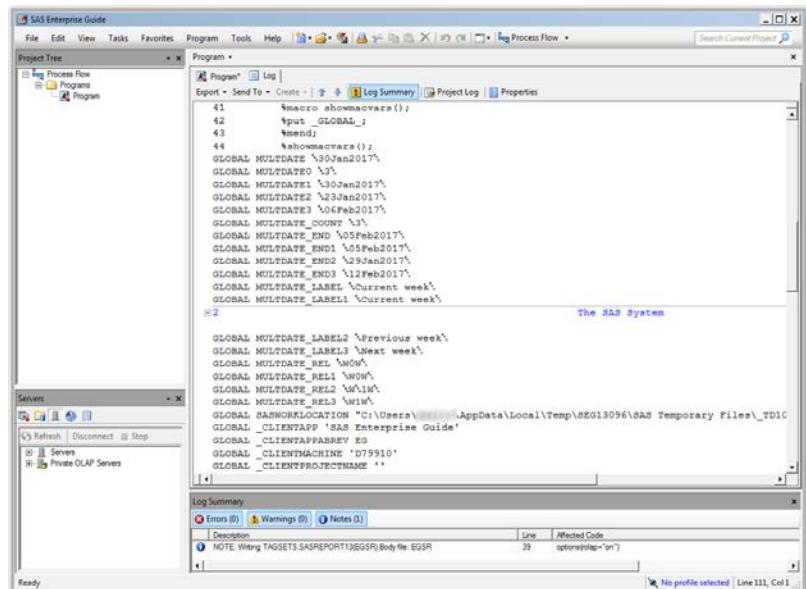
```

Log Summary

Description	Line	Affected Code
NOTE: Writing TAGSETS=SASREPORT1(BGSR) Body file: EGSR	39	options(slps='on')

Display 167 - %LET Statements for Multiple Value Date Prompt

The log of the [Program node using the prompt definition](#) displays the value of the global variables created by the prompt.



```

SAS Enterprise Guide
File Edit View Tasks Favorites Program Tools Help <|> Process Flow Search Current Project

Project Tree
Programs Programs

Program Log
Program Log Summary Project Log Properties
41 %macro showmacvars();
42   %put _GLOBAL_;
43   %mend;
44 %showmacvars();
GLOBAL MULTDATE0 '\30Jan2017';
GLOBAL MULTDATE1 '\3';
GLOBAL MULTDATE2 '\30Jan2017';
GLOBAL MULTDATE3 '\23Jan2017';
GLOBAL MULTDATE4 '\05Feb2017';
GLOBAL MULTDATE_COUNT '3';
GLOBAL MULTDATE_END '\05Feb2017';
GLOBAL MULTDATE_END1 '\05Feb2017';
GLOBAL MULTDATE_END2 '\23Jan2017';
GLOBAL MULTDATE_END3 '\12Feb2017';
GLOBAL MULTDATE_LABEL 'Current week';
GLOBAL MULTDATE_LABEL1 'Current week';

GLOBAL MULTDATE_LABEL2 'Previous week';
GLOBAL MULTDATE_LABEL3 'Next week';
GLOBAL MULTDATE_REL 'W0W';
GLOBAL MULTDATE_REL1 'W0W';
GLOBAL MULTDATE_REL2 'W1W';
GLOBAL MULTDATE_REL3 'W1W';
GLOBAL SASWORKLOCATION "C:\Users\...\AppData\Local\Temp\SEG13096\SAS Temporary Files\_7D1G";
GLOBAL _CLIENTAPP 'SAS Enterprise Guide';
GLOBAL _CLIENTAPPNAME 'EGSR';
GLOBAL _CLIENTMACHINE 'D79310';
GLOBAL _CLIENTPROJECTNAME '';

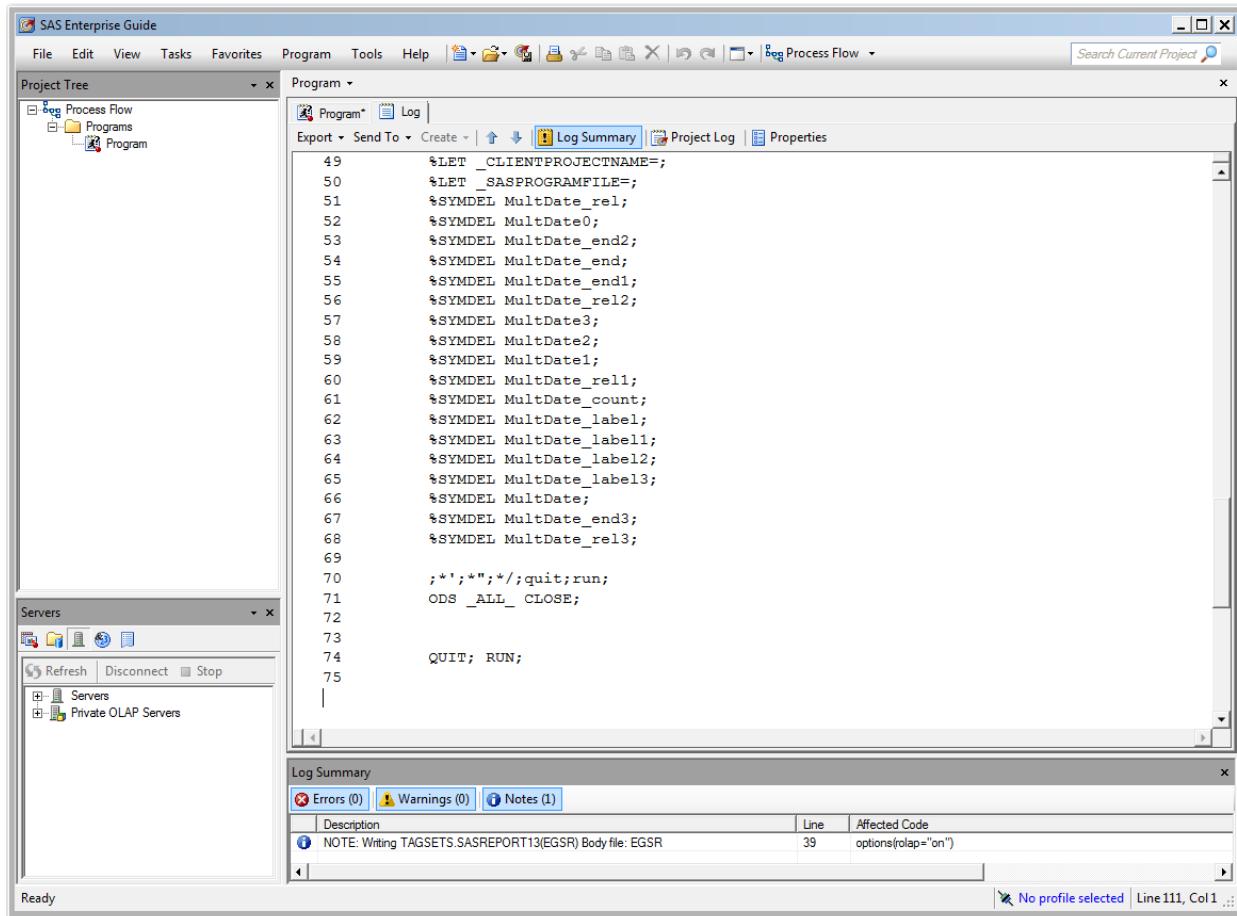
```

Log Summary

Description	Line	Affected Code
NOTE: Writing TAGSETS=SASREPORT1(BGSR) Body file: EGSR	39	options(slps='on')

Display 168 - Global Variables for Multiple Value Date Prompt

Because the **Use prompt value throughout project** is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.



The screenshot shows the SAS Enterprise Guide interface. In the top menu bar, File, Edit, View, Tasks, Favorites, Program, Tools, Help, and Process Flow are visible. A search bar "Search Current Project" is also present. The main window has a "Project Tree" sidebar with a "Program" node selected. The main area displays a "Program" tab with the following code:

```
49      %LET _CLIENTPROJECTNAME=;
50      %LET _SASPROGRAMFILE=;
51      %SYMDEL MultDate_rel;
52      %SYMDEL MultDate0;
53      %SYMDEL MultDate_end2;
54      %SYMDEL MultDate_end;
55      %SYMDEL MultDate_end1;
56      %SYMDEL MultDate_rel2;
57      %SYMDEL MultDate3;
58      %SYMDEL MultDate2;
59      %SYMDEL MultDate1;
60      %SYMDEL MultDate_rel1;
61      %SYMDEL MultDate_count;
62      %SYMDEL MultDate_label;
63      %SYMDEL MultDate_label1;
64      %SYMDEL MultDate_label2;
65      %SYMDEL MultDate_label3;
66      %SYMDEL MultDate;
67      %SYMDEL MultDate_end3;
68      %SYMDEL MultDate_rel3;
69
70      ;*'/*';quit;run;
71      ODS _ALL_ CLOSE;
72
73
74      QUIT; RUN;
75
```

Below the code editor is a "Log Summary" panel. It contains three tabs: Errors (0), Warnings (0), and Notes (1). The Notes tab shows one entry:

Description	Line	Affected Code
NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR	39	options(rolap="on")

At the bottom right of the Log Summary panel, it says "No profile selected | Line 111, Col 1".

Display 169 - %SYMDEL Statements Remove MultDate* Macro Variables

SAS Studio

The following display shows the code that is added to the converted Program node for the multiple values date prompt in SAS Enterprise Guide.

These global [date macro variables](#) are created:

- MULTDATE
- MULTDATE_COUNT
- MULTDATE0
- MULTDATE1
- MULTDATE_label1
- MULTDATE_end1
- MULTDATE_rel1
- MULTDATE2
- MULTDATE_label2
- MULTDATE_end2
- MULTDATE_rel2
- MULTDATE3
- MULTDATE_label3
- MULTDATE_end3
- MULTDATE_rel3

The %LET statements assign the default values to MULTDATEn and MULTDATE_*n variables where n is 1-3 and * is label, end, or rel.

If you want to run your process flow against different input for the MULTDATE prompt, you must manually update the values of the macro variables in the %LET statements. Note that in this example, the MULTDATE_COUNT and MULTDATE_0 variables must reflect the number of text selections you want your program to process, and the MULTDATEn and MULTDATE_*n variables must be in sequential order.

The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. A conversion report for a "Project_Multiple_Weeks.egp" file is displayed. The report details a macro replacement for a date prompt. It specifies that the macro replaces user prompts with code that assigns default values to macro variables defined by the prompts. If a default value was not set in the prompt definition, the macro variable value must be manually set in the code before the macro is called. The code shown is:

```
%macro setPromptValues();  
/* -----  
*SAS Name: MULTDATE  
  
Prompt: Enter multiple weeks between 2 to 4 weeks  
  
Description:  
Weeks must be within the past 10 or next 10 weeks  
Type: Date - week  
  
Minimum value allowed:  
10 weeks ago  
W-10w  
21Nov2016  
27Nov2016  
  
Maximum value allowed:  
10 weeks from now  
W10w  
18Apr2017  
16Apr2017  
  
Between 2 and 4 values are required.  
  
Value choices:  
(all possible values) : _ALL_VALUES_  
(missing values) :  
User can specify additional values.  
----- */
```

Display 170 - Code Comments for Multiple Date Prompt

```

29 *Project_Multiple_Weeks 1 X
Run RESULTS PROPERTIES CONVERSION REPORT
Xglobal MULTDATE_count;
Xlet MULTDATE_count = %nrstr(3);
Xglobal MULTDATE8;
Xlet MULTDATE8 = %nrstr(3);

Xglobal MULTDATE_label;
Xlet MULTDATE_label = %nrstr(Current week);
Xglobal MULTDATE1;
Xlet MULTDATE1 = %nrstr(30Jan2017);
Xglobal MULTDATE_end;
Xlet MULTDATE_end = %nrstr(05Feb2017);
Xglobal MULTDATE_rel;
Xlet MULTDATE_rel = %nrstr(WW);

Xglobal MULTDATE_label1;
Xlet MULTDATE_label1 = %nrstr(Previous week);
Xglobal MULTDATE1;
Xlet MULTDATE1 = %nrstr(23Jan2017);
Xglobal MULTDATE_end1;
Xlet MULTDATE_end1 = %nrstr(29Jan2017);
Xglobal MULTDATE_rel1;
Xlet MULTDATE_rel1 = %nrstr(W-1W);

Xglobal MULTDATE_label2;
Xlet MULTDATE_label2 = %nrstr(Next week);
Xglobal MULTDATE2;
Xlet MULTDATE2 = %nrstr(06Feb2017);
Xglobal MULTDATE_end2;
Xlet MULTDATE_end2 = %nrstr(12Feb2017);
Xglobal MULTDATE_rel2;
Xlet MULTDATE_rel2 = %nrstr(W+1W);

Xglobal MULTDATE_label3;
Xlet MULTDATE_label3 = %nrstr(Middle week);
Xglobal MULTDATE3;
Xlet MULTDATE3 = %nrstr(06Feb2017);
Xglobal MULTDATE_end3;
Xlet MULTDATE_end3 = %nrstr(12Feb2017);
Xglobal MULTDATE_rel3;
Xlet MULTDATE_rel3 = %nrstr(WW);

```

Display 171 - Macro Variable Code for Multiple Date Prompt

Because the **Use prompt value throughout project** option is not checked, the %SYMDEL statements remove the MULTDATE* macro variables.

```

29 *Project_Multiple_Weeks 1 X
Run RESULTS PROPERTIES CONVERSION REPORT
/*
-----+
SAS Studio generated code to cleanup macro variables created by prompts.
"Use prompt throughout" was not checked for these prompts.
-----+
*/
%macro cleanupPromptValues();
%SYMDEL MULTDATE_count;
%SYMDEL MULTDATE8;
%SYMDEL MULTDATE_label;
%SYMDEL MULTDATE1;
%SYMDEL MULTDATE_end;
%SYMDEL MULTDATE_rel;
%SYMDEL MULTDATE_label1;
%SYMDEL MULTDATE1;
%SYMDEL MULTDATE_end1;
%SYMDEL MULTDATE_rel1;
%SYMDEL MULTDATE_label2;
%SYMDEL MULTDATE2;
%SYMDEL MULTDATE_end2;
%SYMDEL MULTDATE_rel2;
%SYMDEL MULTDATE_label3;
%SYMDEL MULTDATE3;
%SYMDEL MULTDATE_end3;
%SYMDEL MULTDATE_rel3;
%end;
%setPromptValues();
/*----- End SAS generated prompt variable code.-----*/
/*----- SAS generated prompt variable cleanup code.----- */
%cleanupPromptValues();
/*----- End SAS generated prompt variable cleanup code.----- */

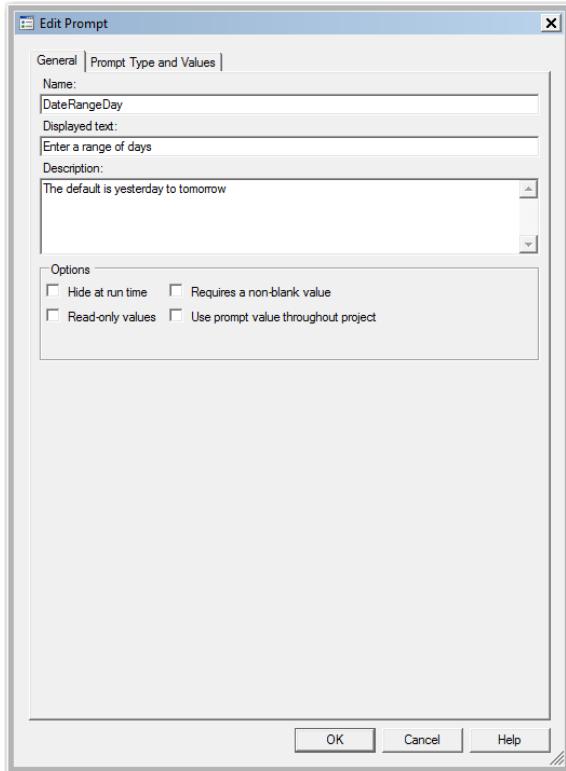
```

Display 172 - %SYMDEL Statements Remove the MULTDATE* Macro Variables

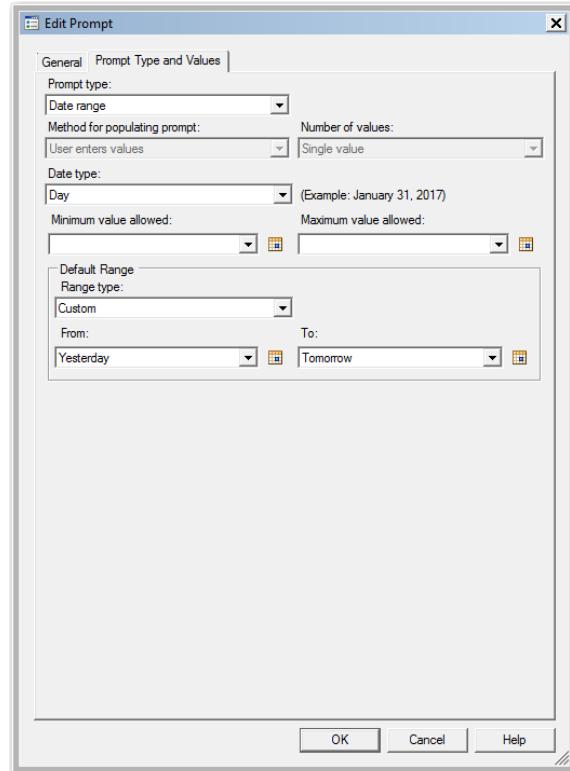
Date Range

SAS Enterprise Guide

In this example, a date range prompt named DateRangeDay is defined as shown in the following two displays.

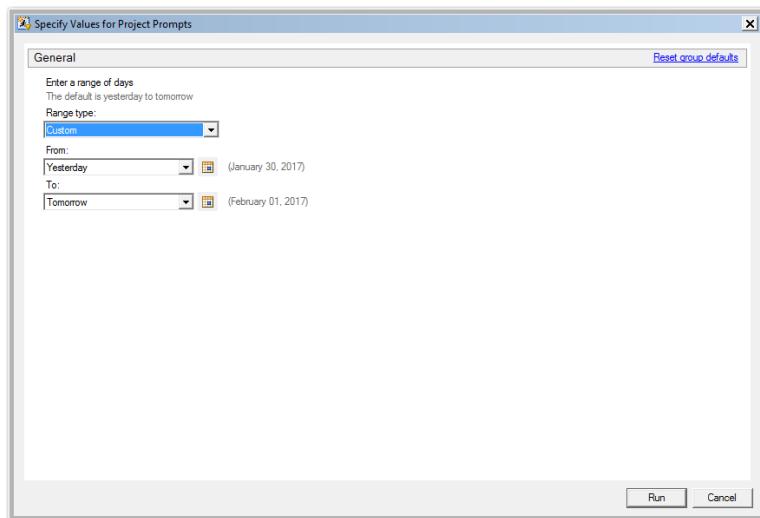


Display 173 - General Properties for Date Range Prompt



Display 174 - Type and Values for Date Range Prompt

When you run the Program node that depends on the prompt, the following dialog box appears:



Display 175 - Date Range Prompt in Prompt Dialog Box

If the user leaves the default values in the date range value prompt fields, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The %LET statements assign the values specified in the prompt dialog box to the DateRangeDay* macro variables.

```

%LET DateRangeDay_max = %nrstr(01Feb2017);
%LET DateRangeDay_min_label = %nrstr(Yesterday);
%LET DateRangeDay_max_label = %nrstr(Tomorrow);
%LET DateRangeDay_min_rel = %nrstr(D-1D);
%LET DateRangeDay_min = %nrstr(30Jan2017);
%LET DateRangeDay_max_rel = %nrstr(D1D);

ODS ALL CLOSE;
OPTIONS DEV=ACTIVE;
GOPTIONS XPIXELS=0 YPIXELS=0;
FILENAME EGSR TEMP;
ODS tagsets saasreport13(ID=EGSR) FILE=EGSR
STYLE=HtmlBlue
STYLESHEET=(URL="file:///C:/Program%20Files/SASHome/SASEnterpriseGuide/7.1/Sty
NOGTITLE
NOFOOTNOTE
GPATH=<saasworklocation>
ENCODING=UTF8
options (xolap="on")
;
NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR
27
28   GOPTIONS ACCESSIBLE;
29   %macro showmacvars();
30   %put _GLOBAL_;
31   %mend;
32
%showmacvars();
GLOBAL DATERANGE_DAY_MAX '01Feb2017';
GLOBAL DATERANGE_DAY_MAX_LABEL 'Tomorrow';
GLOBAL DATERANGE_DAY_MAX_REL 'D1D';
GLOBAL DATERANGE_DAY_MIN '30Jan2017';
GLOBAL DATERANGE_DAY_MIN_LABEL 'Yesterday';
GLOBAL DATERANGE_DAY_MIN_REL 'D1D';
GLOBAL SASWORKLOCATION "C:\Users\asajic\AppData\Local\Temp\SEG1609\SAS Temporary Files\_TD736
GLOBAL _CLIENTAPP 'SAS Enterprise Guide'
GLOBAL _CLIENTPAPREV EG
GLOBAL _CLIENTMACHINE ' '
GLOBAL _CLIENTPROJECTNAME 'Prompt_Date_Range_Day.egp'
GLOBAL _CLIENTPROJECTPATH 'C:\EGPs\Tests\Local\Prompts\Date\Prompt_Date_Range_Day.egp'
GLOBAL _CLIENTTASKLABEL 'Program'
GLOBAL _CLIENTUSERID ' '
GLOBAL _CLIENTUSERNAME 'J. Jeffreys-Chen'
GLOBAL _CLIENTVERSION '7.100.1.2651'
GLOBAL _INITIALIZEDSERVERID 1
GLOBAL _SASHOSTNAME ' '
GLOBAL _SASPROGRAMFILE ''
GLOBAL _SASSERVERNAME 'Local'
33
34   GOPTIONS NOACCESSIBLE;
35

```

Display 176 - %LET Statements for Date Range Prompt

The log of the [Program node using the prompt definition](#) displays the values of the global variables created by the prompt.

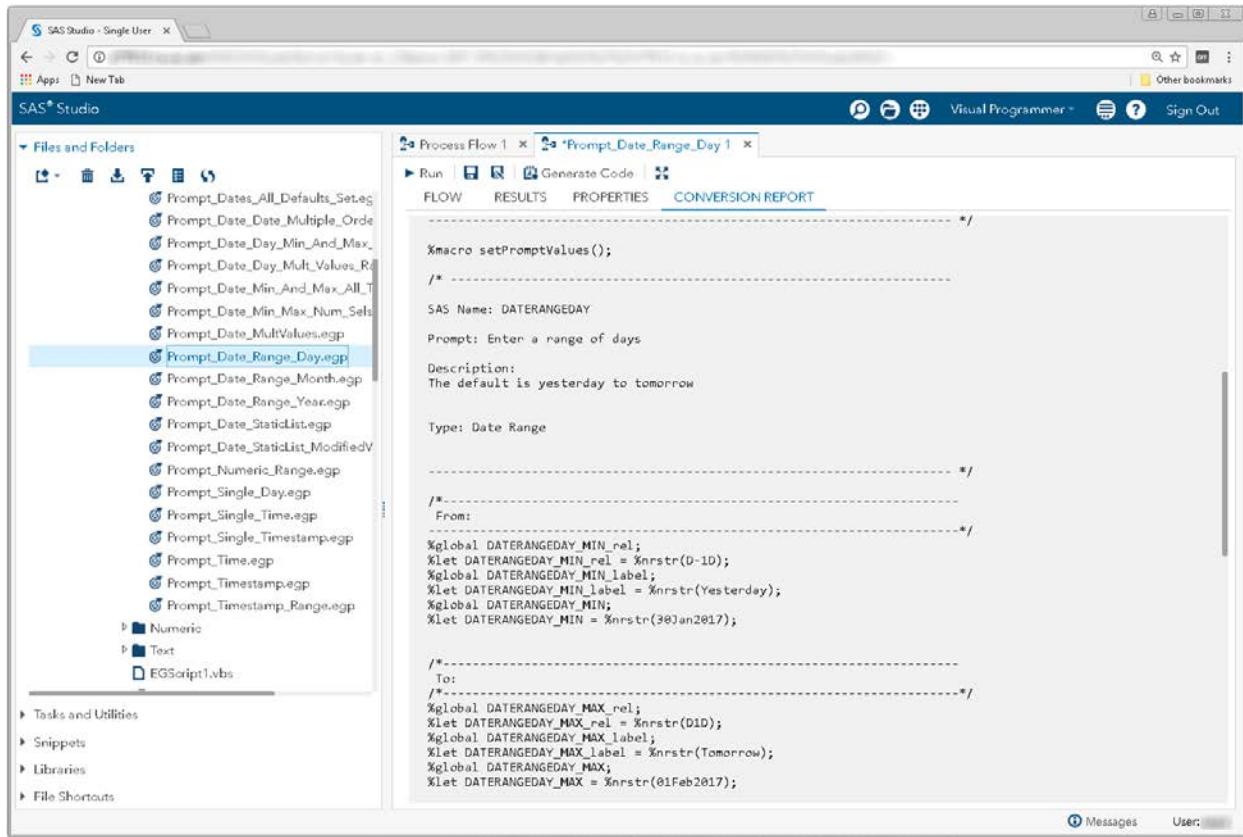
```

%macro showmacvars();
30
%put _GLOBAL_;
31
32
%showmacvars();
GLOBAL DATERANGE_DAY_MAX '01Feb2017';
GLOBAL DATERANGE_DAY_MAX_LABEL 'Tomorrow';
GLOBAL DATERANGE_DAY_MAX_REL 'D1D';
GLOBAL DATERANGE_DAY_MIN '30Jan2017';
GLOBAL DATERANGE_DAY_MIN_LABEL 'Yesterday';
GLOBAL DATERANGE_DAY_MIN_REL 'D1D';
GLOBAL SASWORKLOCATION "C:\Users\asajic\AppData\Local\Temp\SEG1609\SAS Temporary Files\_TD736
GLOBAL _CLIENTAPP 'SAS Enterprise Guide'
GLOBAL _CLIENTPAPREV EG
GLOBAL _CLIENTMACHINE ' '
GLOBAL _CLIENTPROJECTNAME 'Prompt_Date_Range_Day.egp'
GLOBAL _CLIENTPROJECTPATH 'C:\EGPs\Tests\Local\Prompts\Date\Prompt_Date_Range_Day.egp'
GLOBAL _CLIENTTASKLABEL 'Program'
GLOBAL _CLIENTUSERID ' '
GLOBAL _CLIENTUSERNAME 'J. Jeffreys-Chen'
GLOBAL _CLIENTVERSION '7.100.1.2651'
GLOBAL _INITIALIZEDSERVERID 1
GLOBAL _SASHOSTNAME ' '
GLOBAL _SASPROGRAMFILE ''
GLOBAL _SASSERVERNAME 'Local'
33
34   GOPTIONS NOACCESSIBLE;
35

```

Display 177 - Macro Variables for Date Range Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.



The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. The code editor displays the following EGP (Enterprise Guide Procedure) code:

```
/*-----*
 *macro setPromptValues();
 /*-----*
SAS Name: DATERANGEDAY
Prompt: Enter a range of days
Description:
The default is yesterday to tomorrow

Type: Date Range
/*-----*/
/*-----*
From:
/*-----*
%global DATERANGEDAY_MIN_rel;
%let DATERANGEDAY_MIN_rel = %nrstr(D-1D);
%global DATERANGEDAY_MIN_label;
%let DATERANGEDAY_MIN_label = %nrstr(Yesterday);
%global DATERANGEDAY_MIN;
%let DATERANGEDAY_MIN = %nrstr(30Jan2017);

/*-----*
To:
/*-----*/
%global DATERANGEDAY_MAX_rel;
%let DATERANGEDAY_MAX_rel = %nrstr(DID);
%global DATERANGEDAY_MAX_label;
%let DATERANGEDAY_MAX_label = %nrstr(Tomorrow);
%global DATERANGEDAY_MAX;
%let DATERANGEDAY_MAX = %nrstr(01Feb2017);
```

Display 178 - %SYMDEL Statements Remove DATERANGEDAY* Macro Variables

SAS Studio

The following display shows the code that is added to the converted Program node for the date range prompt in SAS Enterprise Guide.

These global [date macro variables](#) are created:

- DATERANGEDAY_MIN_rel
- DATERANGEDAY_MIN_label
- DATERANGEDAY_MIN
- DATERANGEDAY_MAX_rel
- DATERANGEDAY_MAX_label
- DATERANGEDAY_MAX

The %LET statements assign the default values to the DATERANGEDAY macro variables.

If you want to run your process flow using different values for the DATERANGEDAY prompt, you must manually update values of the macro variables in the %LET statements.

```
%macro setPromptValues();
/*
SAS Name: DATERANGEDAY
Prompt: Enter a range of days
Description:
The default is yesterday to tomorrow
Type: Date Range
*/
From:
%global DATERANGEDAY_MIN_rel;
%let DATERANGEDAY_MIN_rel = %nrstr(D-1D);
%global DATERANGEDAY_MIN_label;
%let DATERANGEDAY_MIN_label = %nrstr(Yesterday);
%global DATERANGEDAY_MIN;
%let DATERANGEDAY_MIN = %nrstr(30Jan2017);

To:
%global DATERANGEDAY_MAX_rel;
%let DATERANGEDAY_MAX_rel = %nrstr(D1D);
%global DATERANGEDAY_MAX_label;
%let DATERANGEDAY_MAX_label = %nrstr(Tomorrow);
%global DATERANGEDAY_MAX;
%let DATERANGEDAY_MAX = %nrstr(01Feb2017);
```

Display 179 - Macro Code for Date Range Values

Because the **Use prompt value throughout project** option is not checked, the %SYMDEL statements remove the DATERANGEDAY* macro variables.

The screenshot shows the SAS Studio interface. On the left, the 'Files and Folders' tree view lists various EGP files, with 'Prompt_Date_Range_Day.egp' selected. The main workspace contains a 'Process Flow 1' tab and a 'Prompt_Date_Range_Day 1' code editor. The code editor displays SAS macro code:

```
%macro cleanupPromptValues();
%SYMDEL DATERANGEDAY_MIN_rel;
%SYMDEL DATERANGEDAY_MIN_label;
%SYMDEL DATERANGEDAY_NIN;
%SYMDEL DATERANGEDAY_MAX_rel;
%SYMDEL DATERANGEDAY_MAX_label;
%SYMDEL DATERANGEDAY_NAX;

Xmend;

XsetPromptValues();

/*----- End SAS generated prompt variable code.----- */

/*----- SAS generated prompt variable cleanup code.----- */
%cleanupPromptValues();
/*----- End SAS generated prompt variable cleanup code.----- */

NOTE: Generating node connectors.
NOTE: Linking nodes connected by data nodes and eliminating data nodes.

Process Flow Node Summary
Steps converted:
Program
Type: Code Task
```

Display 180 - %SYMDEL Statements Remove DATERANGEDAY* Macro Variables

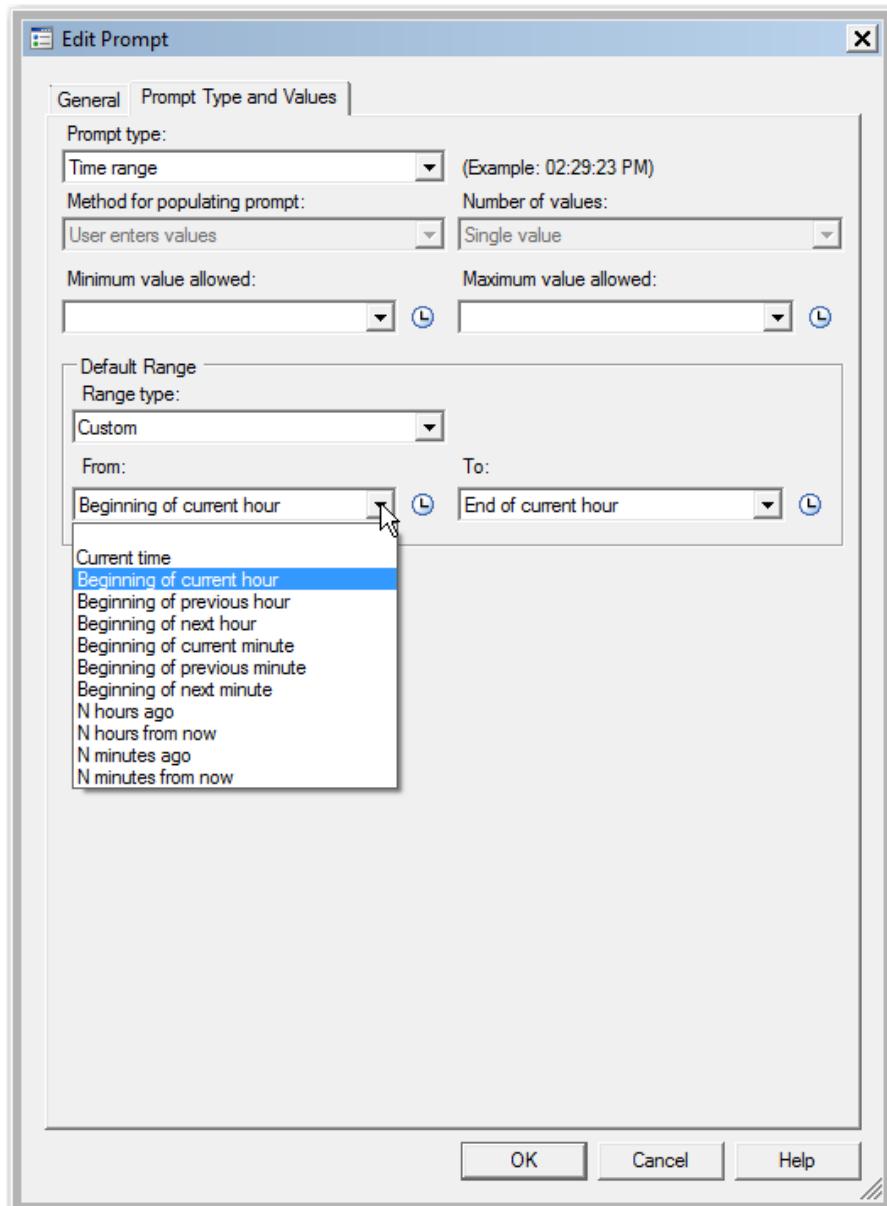
Note: For a Date Range prompt with a period of Month or Year, the macro code does not generate correctly in SAS Studio due to a bug.

Time

Time and Timestamp Macro Variable Explanation

Macro variables generated by SAS Enterprise Guide for time prompts include the basic prompt name, name_rel, and name_label.

- The name macro variable will contain the absolute time specified or the exact time of the common value, for example 15:46:33.
- The name_rel macro variable will contain a code for a common value. For example, "t0m" represents "Current Time".
- The name_label macro variable will contain the common value, such as "Current Time" or "Beginning of Current Hour".

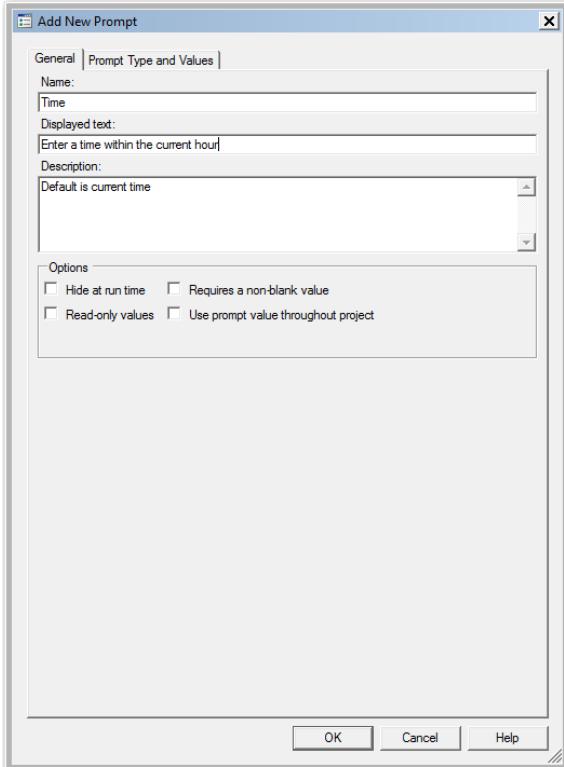


Display 181 – Common Values for Time

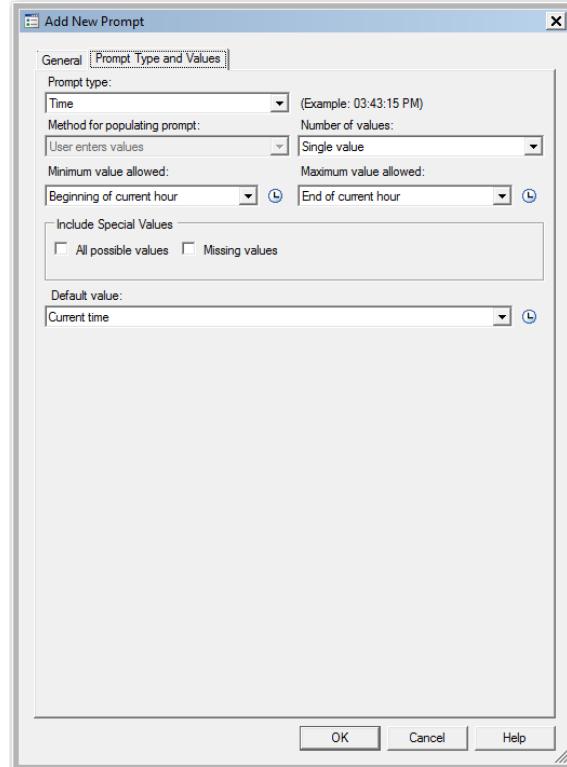
Single Time

SAS Enterprise Guide

In this example, a single value time prompt named Time is defined as shown in the following two displays.

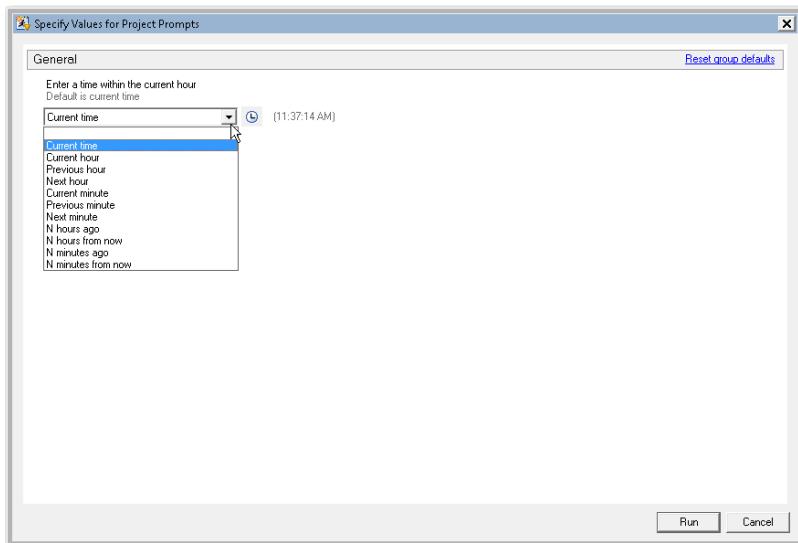


Display 182 - General Properties for Time Prompt



Display 183 - Type and Values for Time Prompt

When you run the Program node that depends on the prompt, the following dialog box appears.

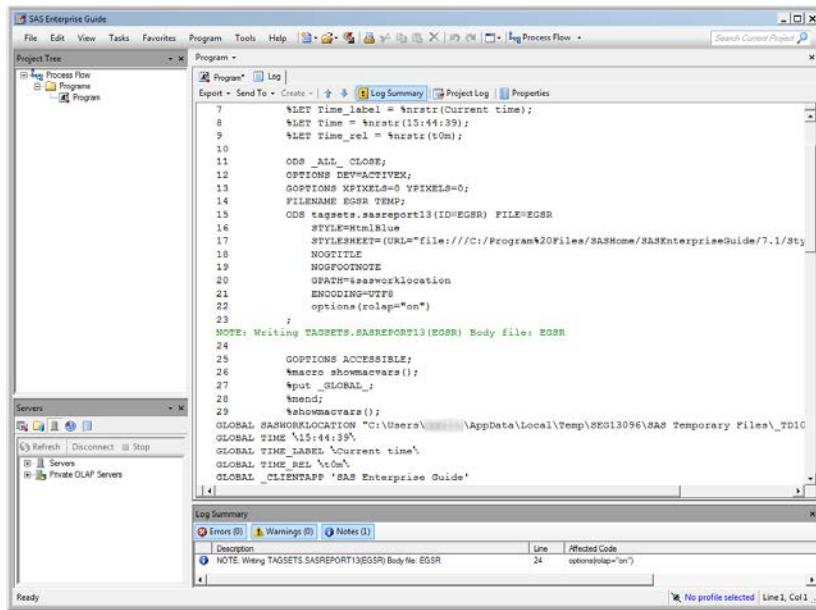


Display 184 - Time Prompt in Prompt Dialog Box

If the user leaves the default value in the time prompt field, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The %LET statements assign the value specified in the prompt dialog box to the macro Time* variables.

The log of the [Program node using the prompt definition](#) displays the values of the global variables created by the prompt.



The screenshot shows the SAS Enterprise Guide interface. The Project Tree on the left shows a 'Programs' folder. The main area is titled 'Program' and contains the following SAS code:

```

7   $LET Time_label = $nrstr(Current_time);
8   $LET Time = $nrstr(15:44:39);
9   $LET Time_rel = $nrstr(t0m);
10
11  ODS _ALL_ CLOSE;
12  OPTIONS DEV=ACTIVEVEX;
13  GOPTIONS XPIXELS=0 YPIXELS=0;
14  FILENAME TEMP;
15  ODS Output SASHELP=Report13(ID=EGSR) FILE=EGSR
16    STYLE=HollowBlue
17    STYLEHOLLOW=(URL=file:///C:/Program%20Files/SASHome/SASEnterpriseGuide/7.1/Sty
18  NOGTITLE
19  NOFOOTNOTE
20  GPATH=$asworklocation
21  ENCODING=UTF8
22  options(rollup='on')
23 ;
NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR
24
25  OPTIONS ACCESSIBLE;
26  %macro showmacrovars();
27  %put _GLOBAL_;
28  %end;
29  %showmacrovars();
GLOBAL _ASWORKLOCATION "C:\Users\...\AppData\Local\Temp\8E013096\SAS Temporary Files\_TD10
GLOBAL TIME '15:44:39';
GLOBAL TIME_LABEL 'Current time';
GLOBAL TIME_REL 't0m';
GLOBAL _CLIENTAPP 'SAS Enterprise Guide'

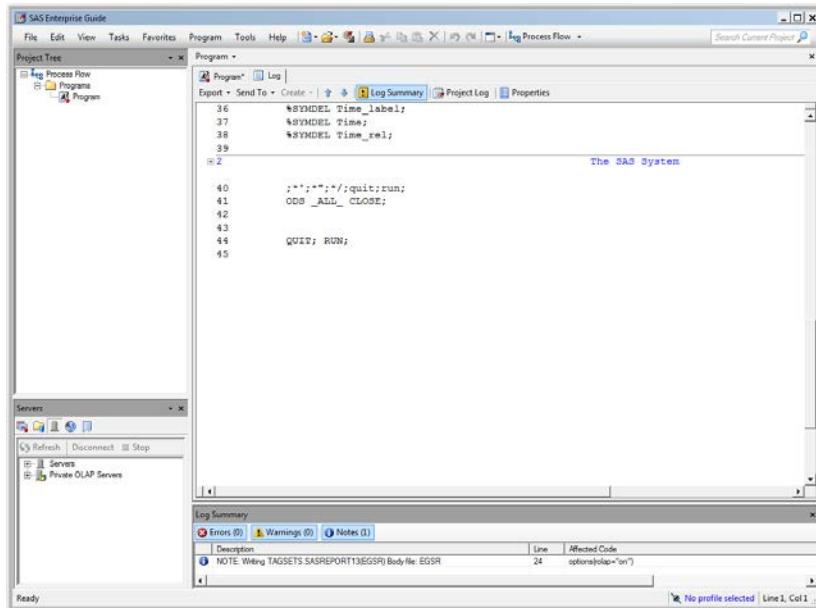
```

Below the code is a 'Log Summary' window showing one note:

Description	Line	Affected Code
NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR	24	options(rollup='on')

Display 185 - Global Macro Variables and %LET Statements for Time Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.



The screenshot shows the SAS Enterprise Guide interface. The Project Tree on the left shows a 'Programs' folder. The main area is titled 'Program' and contains the following SAS code:

```

36  %SYMDEL Time_label;
37  %SYMDEL Time;
38  %SYMDEL Time_rel;
39
40  /*";*/;quit;run;
41  ODS _ALL_ CLOSE;
42
43
44  QUIT; RUN;
45

```

Below the code is a 'Log Summary' window showing one note:

Description	Line	Affected Code
NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR	24	options(rollup='on')

Display 186 - %SYMDEL Statements Remove Time* Macro Variables

SAS Studio

The following display shows the code that is added to the converted Program node for the time prompt in SAS Enterprise Guide. These global [time macro variables](#) are created:

- TIME_rel
- TIME_label
- TIME

The %LET statements assign the default values to the TIME macro variables.

If you want to run your process flow using different values for the TIME prompt, you must manually update the values of the macro variables in the %LET statements.

The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. The left sidebar shows a tree view of files and folders, with 'Prompt_Single_Time.1' selected. The main pane displays the following macro code:

```
%macro setPromptValues();
/* -----
SAS Name: TIME
Prompt: Enter a time within the current hour
Description:
Default is current time
Type: Date - time

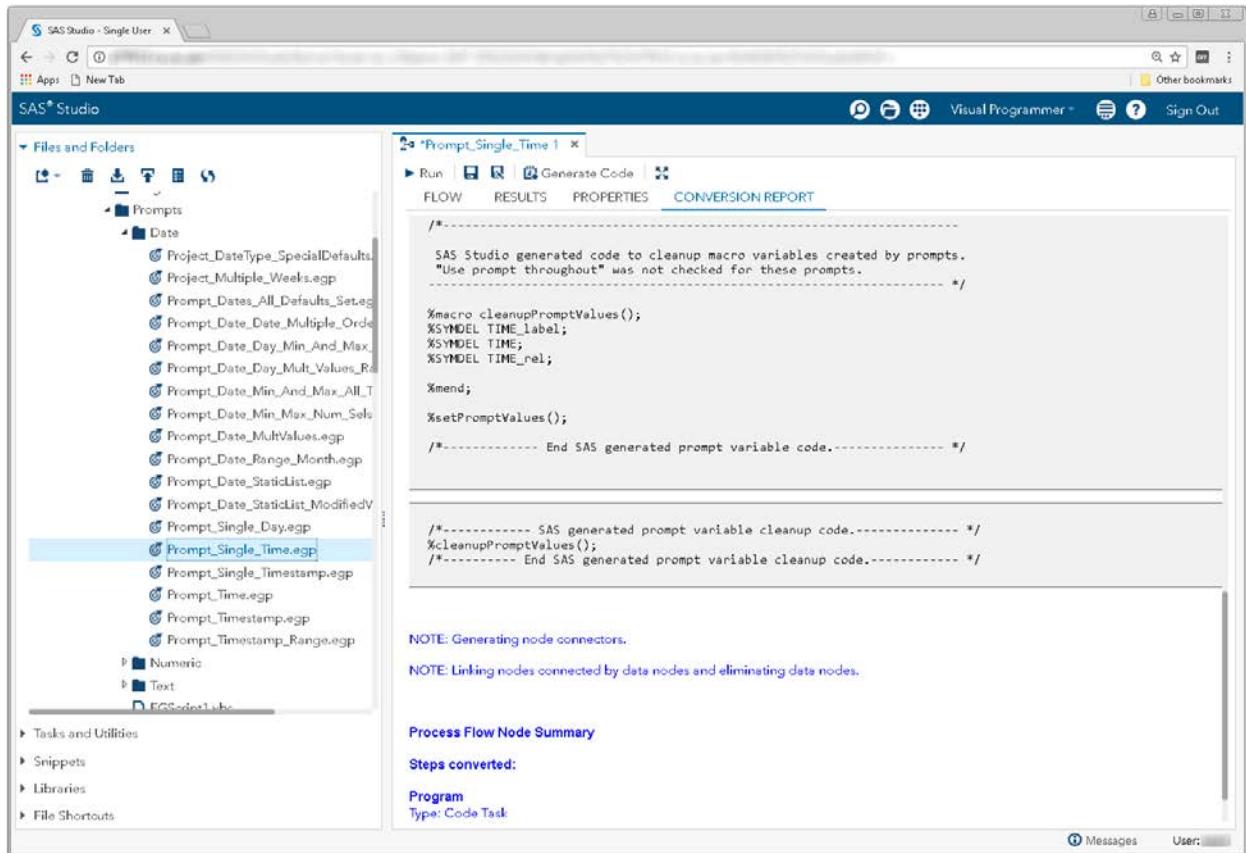
Minimum value allowed:
Beginning of current hour
t0m0H
15:00:00

Maximum value allowed:
End of current hour
t0mEH
15:59:59
-----*/
global TIME_label;
global TIME_label = %nrstr(Current time);
global TIME;
global TIME = %nrstr(15:46:33);
global TIME_rel;
global TIME_rel = %nrstr(t0m);

%end;
```

Display 187 - Macro Code for Time Prompt

Because the **Use prompt value throughout project** option is not checked, the %SYMDEL statements remove the TIME macro variables.



The screenshot shows the SAS Studio interface with the title bar "SAS Studio - Single User". The left sidebar displays a file tree under "Files and Folders" with a node "Prompt_Single_Time.1" selected. The main workspace shows the code for this file:

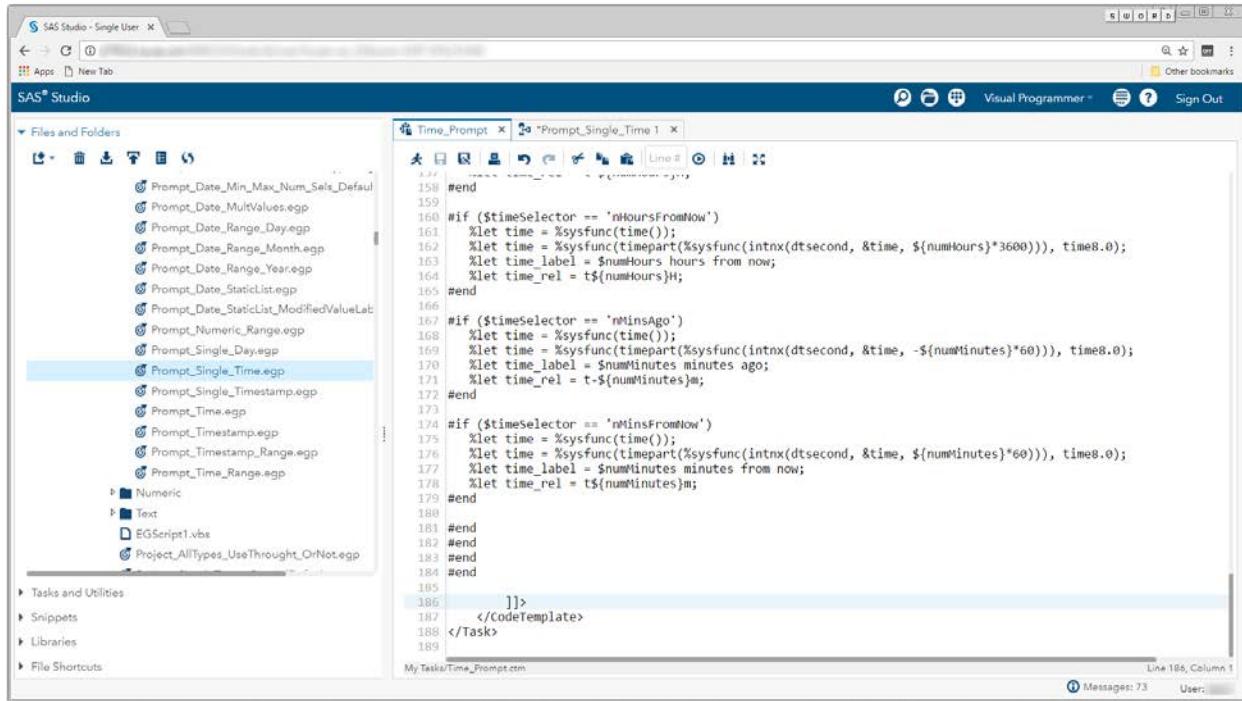
```
/*
-----  
SAS Studio generated code to cleanup macro variables created by prompts.  
"Use prompt throughout" was not checked for these prompts.  
----- */  
  
%macro cleanupPromptValues();  
%SYMDEL TIME_label;  
%SYMDEL TIME;  
%SYMDEL TIME_rel;  
  
%xend;  
  
%setPromptValues();  
  
/*----- End SAS generated prompt variable code.----- */  
  
/*----- SAS generated prompt variable cleanup code.----- */  
%cleanupPromptValues();  
/*----- End SAS generated prompt variable cleanup code.----- */  
  
NOTE: Generating node connectors.  
NOTE: Linking nodes connected by data nodes and eliminating data nodes.
```

Below the code, there are sections for "Process Flow Node Summary", "Steps converted:", and "Program" (Type: Code Task). The bottom right corner shows "Messages" and "User:".

Display 188 - %SYMDEL Statements Remove TIME* Macro Variables

Substituting a SAS Studio Task for Time Prompt

1. Create a SAS Studio task with a control that represents the Time prompt.
 - Add controls as shown in the Time_Prompt task.
 - Set the default value to the default value shown in the generated `setPromptValues()` macro in the converted Program node.
 - Change the strings for the input controls to match the strings specified in the prompt.



The screenshot shows the SAS Studio interface with a code editor window open. The file being edited is named "Time_Prompt.egg". The code is a SAS macro definition for a task. It includes logic to determine the time selector and calculate the time based on the selected unit (hours or minutes from now). The code editor has syntax highlighting and line numbers. The left sidebar shows a file tree with various SAS files like "Prompt_Single_Time.egg" selected. The top menu bar shows "SAS Studio - Single User" and the right side shows "Visual Programmer" and "Sign Out".

```
#end  
158 #if ($timeSelector == 'nHoursFromNow')  
159   %let time = %sysfunc(time());  
160   %let time = %sysfunc(timepart(%sysfunc(intnx(dtsecond, &time, ${numHours}*3600))), time8.0);  
161   %let time_label = ${numHours} hours from now;  
162   %let time_rel = t${numHours}H;  
163 #end  
164  
165 #if ($timeSelector == 'nMinsAgo')  
166   %let time = %sysfunc(time());  
167   %let time = %sysfunc(timepart(%sysfunc(intnx(dtsecond, &time, -${numMinutes}*60)), time8.0));  
168   %let time_label = ${numMinutes} minutes ago;  
169   %let time_rel = t-${numMinutes}m;  
170 #end  
171  
172 #if ($timeSelector == 'nMinsFromNow')  
173   %let time = %sysfunc(time());  
174   %let time = %sysfunc(timepart(%sysfunc(intnx(dtsecond, &time, ${numMinutes}*60)), time8.0));  
175   %let time_label = ${numMinutes} minutes from now;  
176   %let time_rel = t${numMinutes}m;  
177 #end  
178  
179 #end  
180  
181 #end  
182 #end  
183 #end  
184 #end  
185  
186   ]]]>  
187 </CodeTemplate>  
188 </Task>  
189
```

Display 189 - Replacement Task for Time Prompt

The following code is an example of a task that could be used as the time prompt.

```
<?xml version="1.0" encoding="UTF-8"?><Task runNLS="never" schemaVersion="5.1">
<Registration>
<Name>Time_Prompt</Name>
<Description>Time Prompt</Description>
<GUID/>
<Procedures>TBD</Procedures>
<Version>3.6</Version>
<Links></Links>
</Registration>
<Metadata>
    <DataSources> </DataSources>
    <Options>
        <Option inputType="string" name="label">
            Enter a time within the current hour
        </Option>
        <Option defaultValue="currentTime" inputType="combobox"
            name="timeSelector">
            Default is current time
        </Option>
        <Option inputType="string" name="specify">Specify time</Option>
        <Option inputType="string" name="currentTime">Current time</Option>
        <Option inputType="string" name="currentHour">Current hour</Option>
        <Option inputType="string" name="prevHour">Previous hour</Option>
        <Option inputType="string" name="nextHour">Next hour</Option>
        <Option inputType="string" name="currMin">Current minute</Option>
        <Option inputType="string" name="prevMin">Previous minute</Option>
        <Option inputType="string" name="nextMin">Next minute</Option>
        <Option inputType="string" name="nHoursAgo">N hours ago</Option>
        <Option inputType="string" name="nHoursFromNow">
            N hours from now
        </Option>
        <Option inputType="string" name="nMinsAgo">
            N minutes ago
        </Option>
        <Option inputType="string" name="nMinsFromNow">
            N minutes from now
        </Option>
        <Option inputType="inputtext" name="timeInput"
            promptMessage="Example: 01:23:45 PM" />
        <Option defaultValue="N" inputType="numbertext"
            invalidMessage="Invalid value. Enter a positive integer."
            minValue="0"
            missingMessage="Enter number of minutes"
            name="numMinutes"
            promptMessage="Enter number of minutes"
            rangeMessage=
                "This number is out of range. Enter a positive number.">
            Number of minutes:
        </Option>
        <Option defaultValue="N" inputType="numbertext"
            invalidMessage="Invalid value. Enter a positive integer."
            minValue="0" missingMessage="Enter number of hours"
            name="numHours"
            promptMessage="Enter number of hours"
            rangeMessage=
                "This number is out of range. Enter a positive number.">
            Number of hours:
        </Option>
    </Options>
</Metadata>
```

```

<UI>
    <OptionItem option="label"/>
    <OptionChoice options="timeSelector">
        <OptionItem option="specify"/>
        <OptionItem option="currentTime"/>
        <OptionItem option="currentHour"/>
        <OptionItem option="prevHour"/>
        <OptionItem option="nextHour"/>
        <OptionItem option="currMin"/>
        <OptionItem option="prevMin"/>
        <OptionItem option="nextMin"/>
        <OptionItem option="nHoursAgo"/>
        <OptionItem option="nHoursFromNow"/>
        <OptionItem option="nMinsAgo"/>
        <OptionItem option="nMinsFromNow"/>
    </OptionChoice>

    <OptionItem option="timeInput"/>
    <OptionItem option="numMinutes"/>
    <OptionItem option="numHours"/>
</UI>
<Dependencies>
    <Dependency condition="($timeSelector == 'specify')">
        <Target action="hide" conditionResult="false" option="timeInput"/>
        <Target action="show" conditionResult="true" option="timeInput"/>
    </Dependency>
    <Dependency condition="((($timeSelector == 'nMinsAgo') || ($timeSelector == 'nMinsFromNow')))">
        <Target action="hide" conditionResult="false" option="numMinutes"/>
        <Target action="show" conditionResult="true" option="numMinutes"/>
    </Dependency>
    <Dependency condition="((($timeSelector == 'nHoursAgo') || ($timeSelector == 'nHoursFromNow')))">
        <Target action="hide" conditionResult="false" option="numHours"/>
        <Target action="show" conditionResult="true" option="numHours"/>
    </Dependency>
</Dependencies>
<CodeTemplate>
    <![CDATA[
%global time;
%global time_rel;
%global time_label;

#if ($timeSelector == 'specify')
    %let time = $timeInput;
    %let time = %sysfunc(inputn(&time, TIME10.),TOD);
    %let time_label = $timeInput;
    %symdel time_rel;
#else
    #if ($timeSelector == 'currentTime')
        %let time = %sysfunc(time(),time8.0);
        %let time_label = Current time;
        %let time_rel = t0m;
    #else
        #if ($timeSelector == 'currentHour')
            %let time = %sysfunc(time());
            %let time = %sysfunc(intnx(hour, &time, 0, b), time8.0);
            %let time_label = Current hour;
            %let time_rel = H0H;
        #else
            #if ($timeSelector == 'currMin')
                %let time = %sysfunc(time());
                %let time = %sysfunc(intnx(minute, &time, 0, b), time8.0);
            #endif
        #endif
    #endif

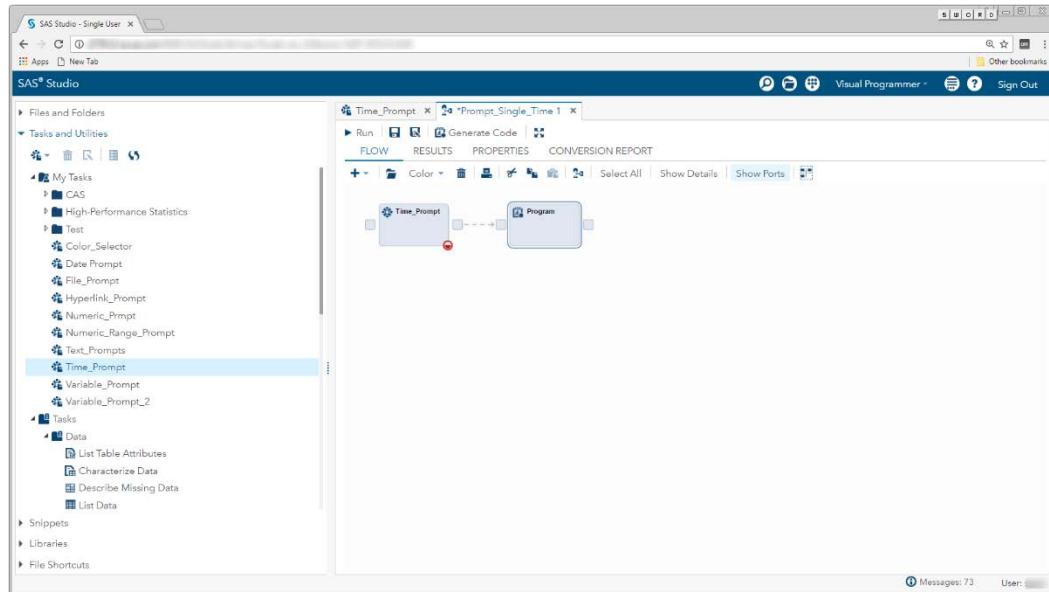
```

```

    %let time_label = Current minute;
    %let time_rel = mom;
#else
#if ($timeSelector == 'prevHour')
    %let time = %sysfunc(time());
    %let time = %sysfunc(timepart(%sysfunc(intnx(dthour, &time, -1))), time8.0);
    %let time_label = Previous hour;
    %let time_rel = H-1H;
#endif
#if ($timeSelector == 'nextHour')
    %let time = %sysfunc(time());
    %let time = %sysfunc(timepart(%sysfunc(intnx(dthour, &time, 1))), time8.0);
    %let time_label = Next hour;
    %let time_rel = H1H;
#endif
#if ($timeSelector == 'prevMin')
    %let time = %sysfunc(time());
    %let time = %sysfunc(timepart(%sysfunc(intnx(dtminute, &time, -1))), time8.0);
    %let time_label = Previous minute;
    %let time_rel = m-1m;
#endif
#if ($timeSelector == 'nextMin')
    %let time = %sysfunc(time());
    %let time = %sysfunc(timepart(%sysfunc(intnx(dtminute, &time, 1))), time8.0);
    %let time_label = Next minute;
    %let time_rel = m1m;
#endif
#if ($timeSelector == 'nHoursAgo')
    %let time = %sysfunc(time());
    %let time = %sysfunc(timepart(%sysfunc(intnx(dtsecond, &time, -${numHours}*3600))), time8.0);
    %let time_label = ${numHours} hours ago;
    %let time_rel = t-${numHours}H;
#endif
#if ($timeSelector == 'nHoursFromNow')
    %let time = %sysfunc(time());
    %let time = %sysfunc(timepart(%sysfunc(intnx(dtsecond, &time, ${numHours}*3600))), time8.0);
    %let time_label = ${numHours} hours from now;
    %let time_rel = t${numHours}H;
#endif
#if ($timeSelector == 'nMinsAgo')
    %let time = %sysfunc(time());
    %let time = %sysfunc(timepart(%sysfunc(intnx(dtsecond, &time, -${numMinutes}*60))), time8.0);
    %let time_label = ${numMinutes} minutes ago;
    %let time_rel = t-${numMinutes}m;
#endif
#if ($timeSelector == 'nMinsFromNow')
    %let time = %sysfunc(time());
    %let time = %sysfunc(timepart(%sysfunc(intnx(dtsecond, &time, ${numMinutes}*60))), time8.0);
    %let time_label = ${numMinutes} minutes from now;
    %let time_rel = t${numMinutes}m;
#endif
#endif
    ]]>
</CodeTemplate>
</Task>

```

2. Save the prompt replacement task to your **My Tasks** folder.
3. Drag the task from **My Tasks** into your converted process flow.
4. Link the output port of the task to the input port of the converted Program node.



Display 190 - Time Input Task Linked to Program Node

5. Comment out the `%setPromptValues()` macro call from the converted Program node. The macro code generated by the time input task replaces this code.

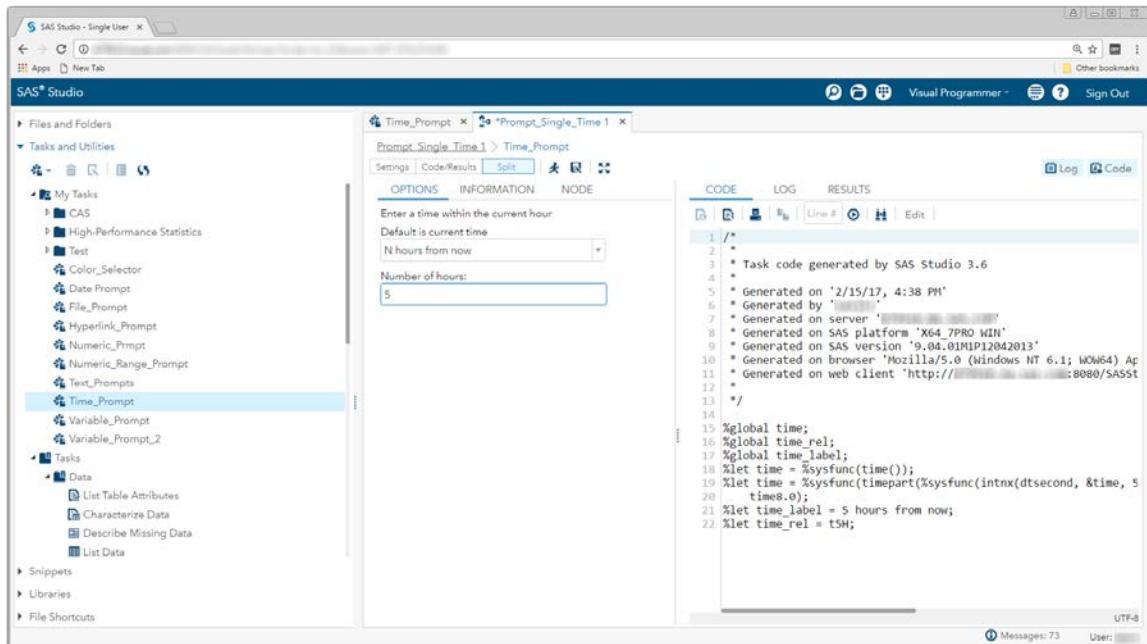
```

CODE LOG RESULTS NODE
52  SAS Studio generated code to cleanup macro variables created by prompts.
53  "use prompt throughout" was not checked for these prompts.
54  -----
55  -----
56  %macro cleanupPromptValues();
57  %SYMDEL TIME_label;
58  %SYMDEL TIME;
59  %SYMDEL TIME_rel;
60  %
61  %mend;
62  %
63  %
64  *%setPromptValues();
65  /*
66  ----- End SAS generated prompt variable code.-----
67  %
68  %macro showmacvars();
69  %put _GLOBAL_;
70  %mend;
71  %
72  /*----- SAS generated prompt variable cleanup code.----- */
73  %cleanupPromptValues();
74  /*
75  ----- End SAS generated prompt variable cleanup code.----- */
76

```

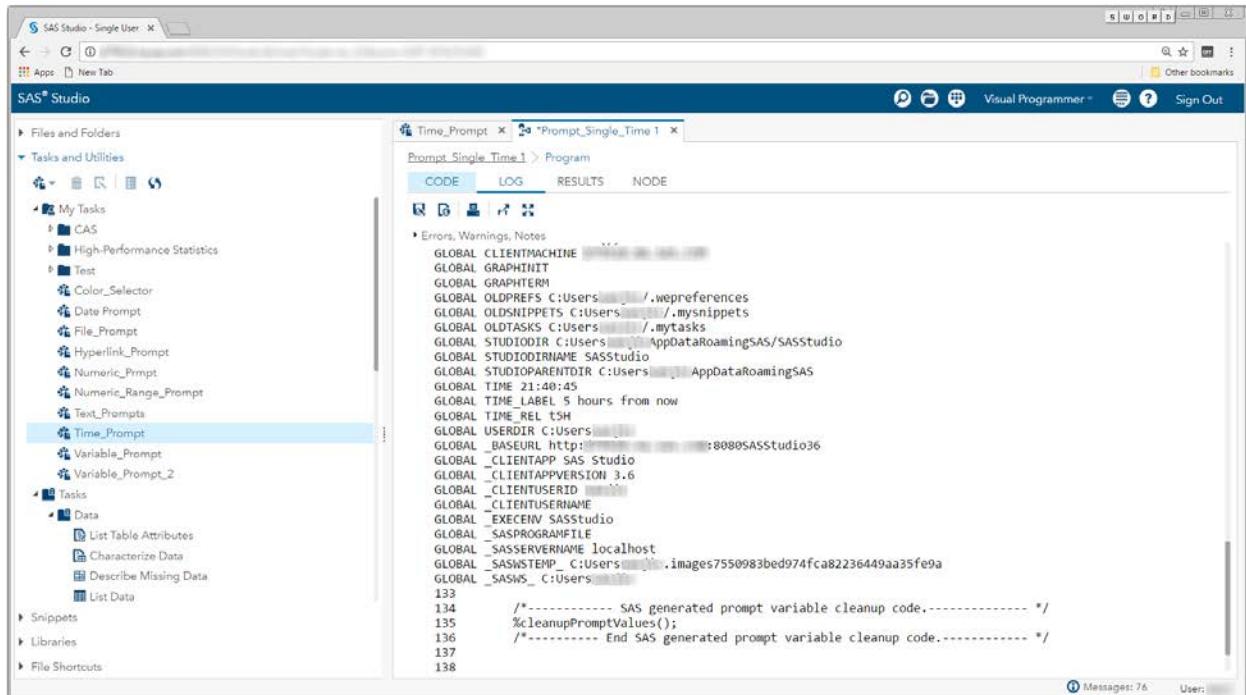
Display 191 – Commented out `%setPromptValues` in Program Node

To run your flow with a different time value than the default value, open the Time_Prompt node and specify a different time.



Display 192 – Time_Prompt Task

When you run the process flow, the global Time* variables are set to the value specified in the task.

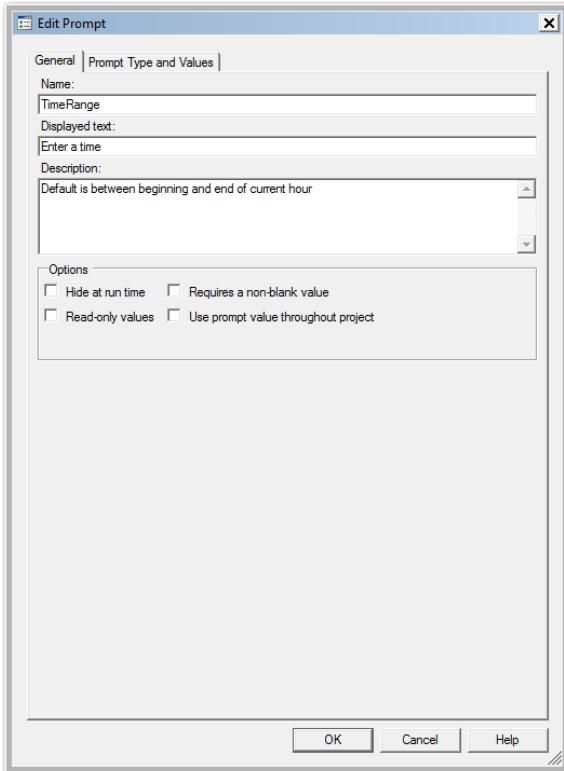


Display 193 – Time Prompt Variables with Updated Values

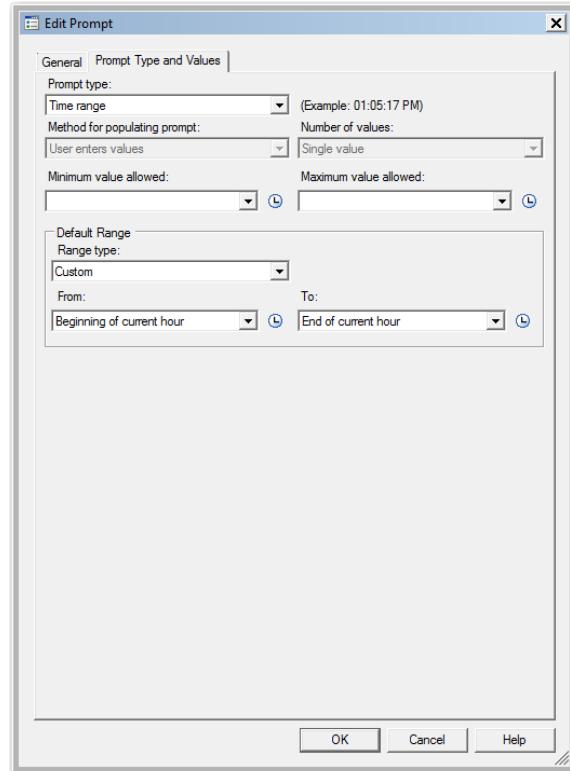
Time Range

SAS Enterprise Guide

In this example, a time range prompt named TimeRange is defined as shown in the following two displays.

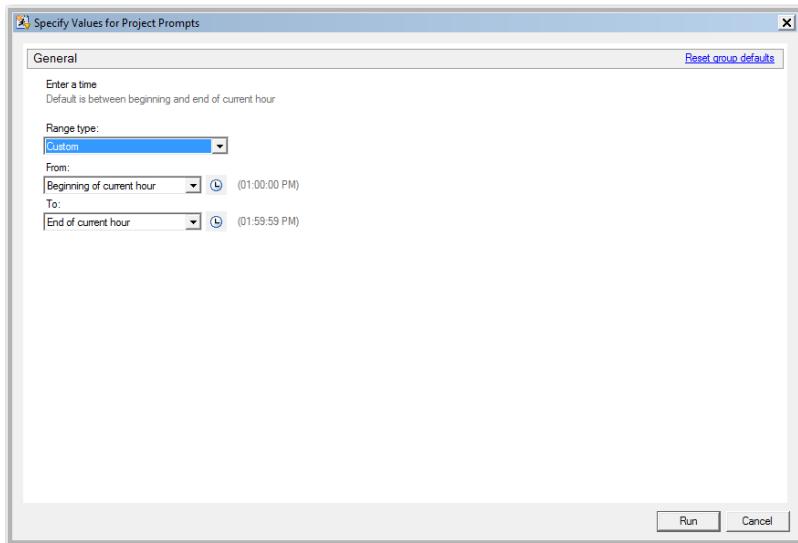


Display 194 - General Properties for Time Range Prompt



Display 195 - Type and Values for Time Range Prompt

When you run the Program node that depends on the prompt, the following dialog box appears:



Display 196 - Time Range Prompt in Prompt Dialog Box

If the user leaves the default values in the prompt fields, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The %LET statements assign the values specified in the prompt dialog box to the Timerange* macro variables.

The screenshot shows the SAS Enterprise Guide interface. The top menu bar includes File, Edit, View, Tasks, Favorites, Program, Tools, Help, and Log Process Flow. A search bar at the top right says "Search Current Project". The left sidebar has a "Project Tree" section with a "Programs" node expanded, showing a "Program" node. Below it is a "Servers" section with "Refresh", "Disconnect", "Stop", "Servers", and "Private OLAP Servers". The main workspace contains a "Program" tab with a code editor showing a SAS program. The code includes various SAS statements like LET, CDS, and options. It also includes a note about writing a report and setting global options. The bottom right corner shows a status bar with "No profile selected" and "Line 75, Col 1".

Display 197 - %LET Statements for Macro Variables of Time Range Prompt

The log of the [Program node using the prompt definition](#) displays the values of the global variables created by the prompt.

The screenshot shows the SAS Enterprise Guide interface with the following details:

- Project Tree:** A tree view showing a single node named "Prompt_Time_Range".
- Program Editor:** The main workspace displays the following SAS code:

```
GLOBAL _TIMEPARENCE_MAX '13:59:59';
GLOBAL _TIMEPARENCE_MAX_LABEL 'End of current hour';
GLOBAL _TIMEPARENCE_MIN '00:00:00';
GLOBAL _TIMEPARENCE_MIN_LABEL 'Beginning of current hour';
GLOBAL _CLIENTAPP 'SAS Enterprise Guide';
GLOBAL _CLIENTTASKAPPREV 23;
GLOBAL _CLIENTPROJECTNAME '';
GLOBAL _CLIENTPROJECTPATH 'C:\EDGPa\_TESTS\Local\Prompts\Date\Prompt_Time_Range.eqp';
GLOBAL _CLIENTTASKLABEL 'Program';
GLOBAL _CLIENTUSERID '';
GLOBAL _CLIENTUSERNAME 'J. Jeffreys-Chen';
GLOBAL _CLIENTVERSION '9.100.1.2651';
GLOBAL _INITIALIZEDDEVEVERID 1;
GLOBAL _VARHOSTNAME '';
GLOBAL _SASPROGRAMFILE '';
GLOBAL _SASERVERNAME 'Local';
33
34      OPTIONS NOACCESSIBLE;
35      %LET _CLIENTTASKLABEL="";
36      %LET _CLIENTPROJECTPATH="";
37
38      %LET _CLIENTPROJECTNAME="";
39      %LET _SASERVERNAME="';
```
- Servers View:** Shows a list of servers, including "Servers" and "Private CLAP Servers".
- Log Summary:** A summary of log activity with tabs for Errors, Warnings, and Notes.
- Status Bar:** Displays "Ready" and "No profile selected | Line 75, Col 1".

Display 198 – Global Macro Variables for Time Range Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.

The screenshot shows the SAS Enterprise Guide interface with the following components:

- Project Tree:** Shows a node for "Program".
- Servers:** Shows a list of servers including "Servers" and "Private OLAP Servers".
- Program Editor:** The main workspace displays a SAS program code:

```
39      %SYMDEL TimeRange_min_rel;
40      %SYMDEL TimeRange_max_label;
41      %SYMDEL TimeRange_min;
42      %SYMDEL TimeRange_min_label;
43      %SYMDEL TimeRange_max_rel;
44      %SYMDEL TimeRange_max;
45
46      ;/*";*/;quit;run;
47      ODS _ALL_ CLOSE;
48
49
50      QUIT; RUN;
51
```
- Log Summary:** A panel at the bottom shows log details with 1 note:

Description	Line	Affected Code
NOTE: WMLT-TACSETS.SASREPORT12(FCDF) B-4-B-FCDF	57	-----

Display 199 - %SYMDEL Statements Remove the TimeRange* Macro Variables

SAS Studio

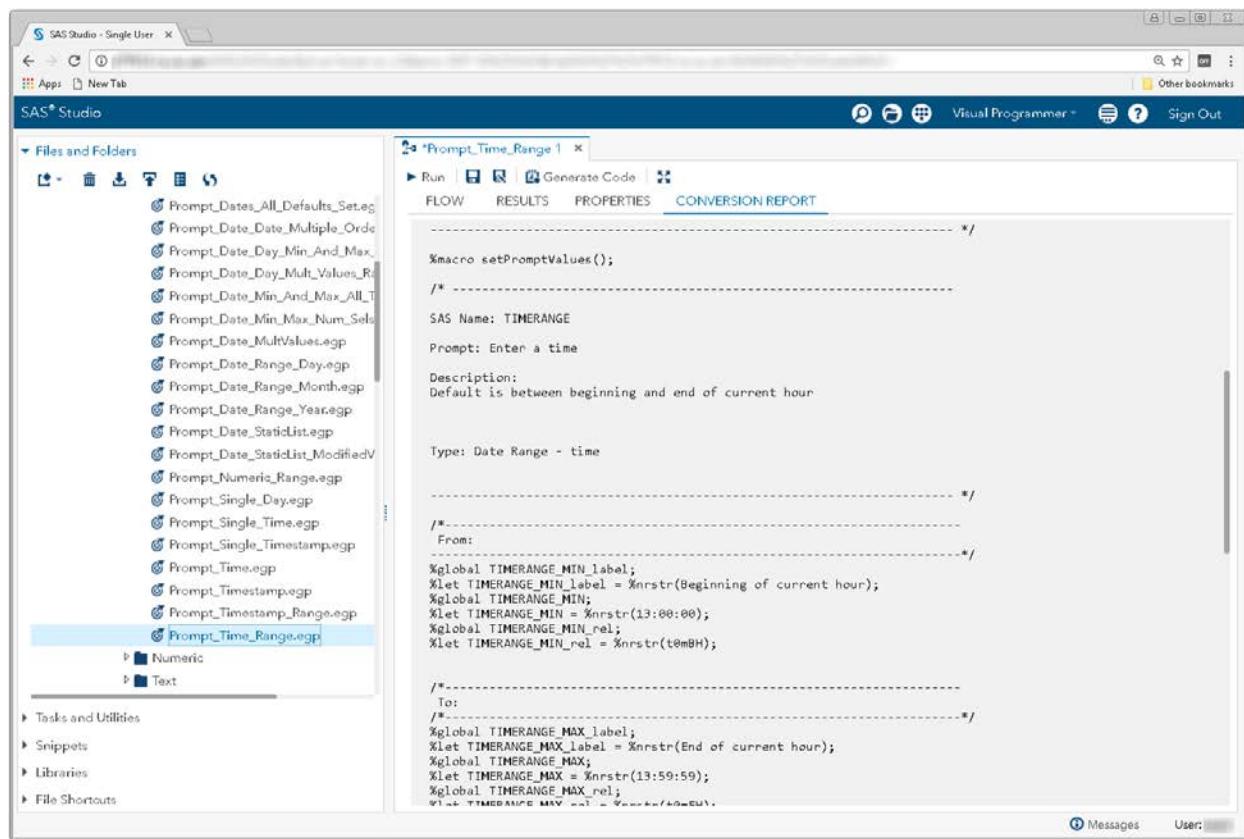
The following display shows code that is added to the converted Program node for the time range prompt in SAS Enterprise Guide.

These global time macro variables are created:

- TIMERANGE_MIN_rel
- TIMERANGE_MIN_label
- TIMERANGE_MIN
- TIMERANGE_MAX_rel
- TIMERANGE_MAX_label
- TIMERANGE_MAX

The %LET statements assign the default value to the TIMERANGE* macro variables.

If you want to run your process flow using different input for the TIMERANGE prompt, you must manually update the values of the macro variables in the %LET statements.

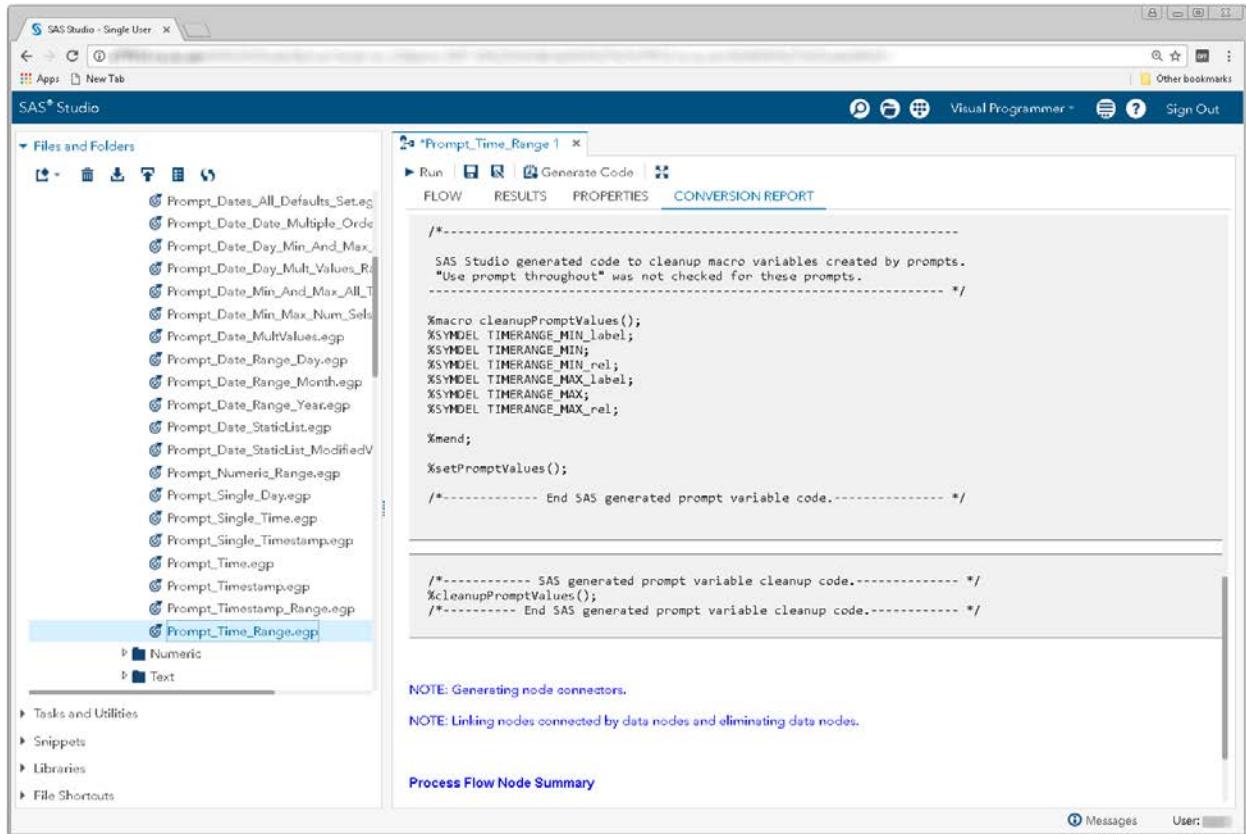


The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. The code editor displays the following macro code for a time range prompt:

```
%macro setPromptValues();  
/* -----  
SAS Name: TIMERANGE  
Prompt: Enter a time  
Description:  
Default is between beginning and end of current hour  
  
Type: Date Range - time  
----- */  
From:  
-----  
%global TIMERANGE_MIN_label;  
%let TIMERANGE_MIN_label = %nrstr(Beginning of current hour);  
%global TIMERANGE_MIN;  
%let TIMERANGE_MIN = %nrstr(13:00:00);  
%global TIMERANGE_MIN_rel;  
%let TIMERANGE_MIN_rel = %nrstr(t0mBH);  
  
To:  
-----  
%global TIMERANGE_MAX_label;  
%let TIMERANGE_MAX_label = %nrstr(End of current hour);  
%global TIMERANGE_MAX;  
%let TIMERANGE_MAX = %nrstr(13:59:59);  
%global TIMERANGE_MAX_rel;  
%let TIMERANGE_MAX_rel = %nrstr(t0mEH);
```

Display 200 – Code for Time Range Prompt

Because the **Use prompt value throughout project** option is not checked, the %SYMDEL statements remove the TIMERANGE macro variables.



The screenshot shows the SAS Studio interface. On the left, the 'Files and Folders' browser lists various EGP files related to date and time prompts. The main area is a code editor titled 'Prompt_Time_Range.1' showing SAS code. The code includes sections for macro cleanup and prompt variable cleanup, both of which contain %SYMDEL statements for TIMERANGE macro variables. A note at the bottom of the code editor states: 'NOTE: Generating node connectors.' and 'NOTE: Linking nodes connected by data nodes and eliminating data nodes.'

```
/*
-----  
SAS Studio generated code to cleanup macro variables created by prompts.  
"Use prompt throughout" was not checked for these prompts.  
----- */  
  
%macro cleanupPromptValues();  
  %SYMDEL TIMERANGE_MIN_label;  
  %SYMDEL TIMERANGE_MIN;  
  %SYMDEL TIMERANGE_MIN_rel;  
  %SYMDEL TIMERANGE_MAX_label;  
  %SYMDEL TIMERANGE_MAX;  
  %SYMDEL TIMERANGE_MAX_rel;  
  
  %mend;  
  
%setPromptValues();  
/*----- End SAS generated prompt variable code.----- */  
  
/*----- SAS generated prompt variable cleanup code.----- */  
/*cleanupPromptValues();*/  
/*----- End SAS generated prompt variable cleanup code.----- */  
  
NOTE: Generating node connectors.  
NOTE: Linking nodes connected by data nodes and eliminating data nodes.
```

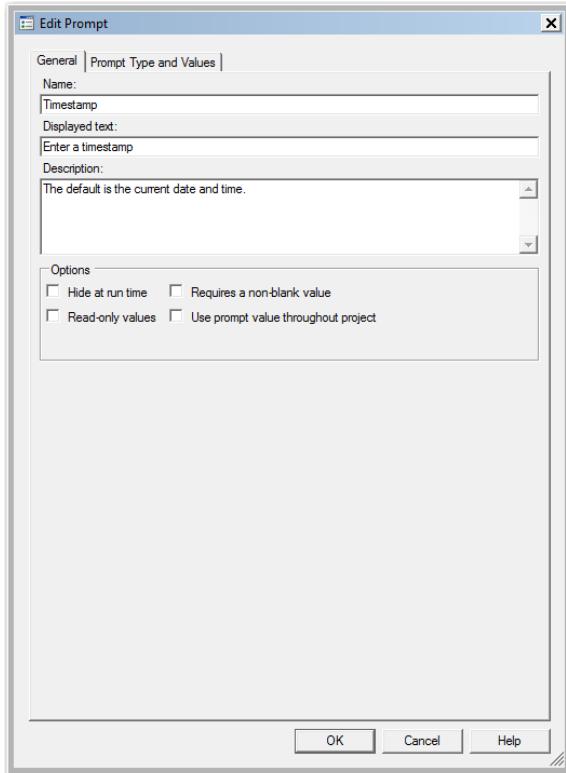
Display 201 - %SYMDEL Code Removes TIMERANGE* Macro Variables

Timestamp

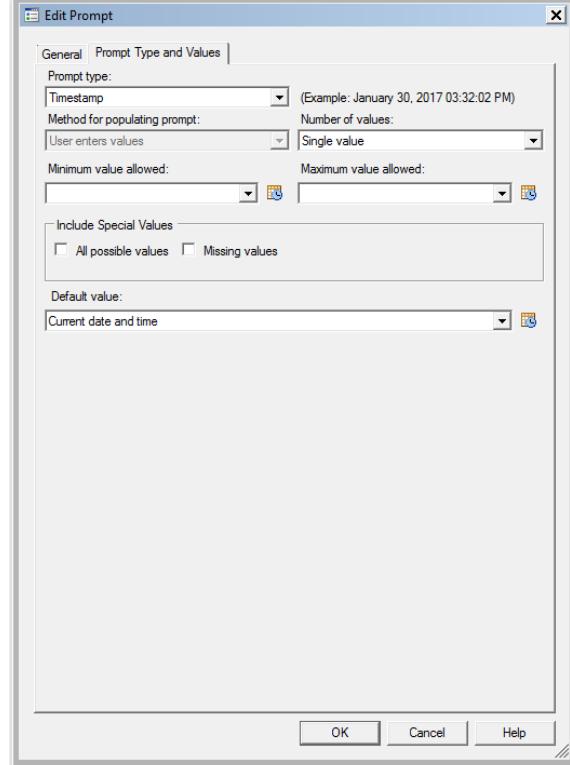
Single Timestamp

SAS Enterprise Guide

In this example, a timestamp prompt named Timestamp is defined as shown in the following two displays.

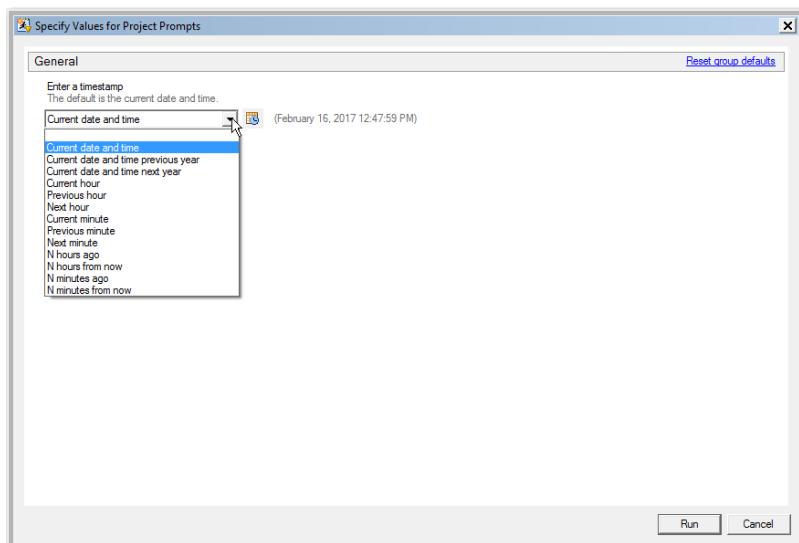


Display 202 - General Properties for Timestamp Prompt



Display 203 - Type and Values for Timestamp Prompt

When you run the Program node that depends on the prompt, the following dialog box appears:

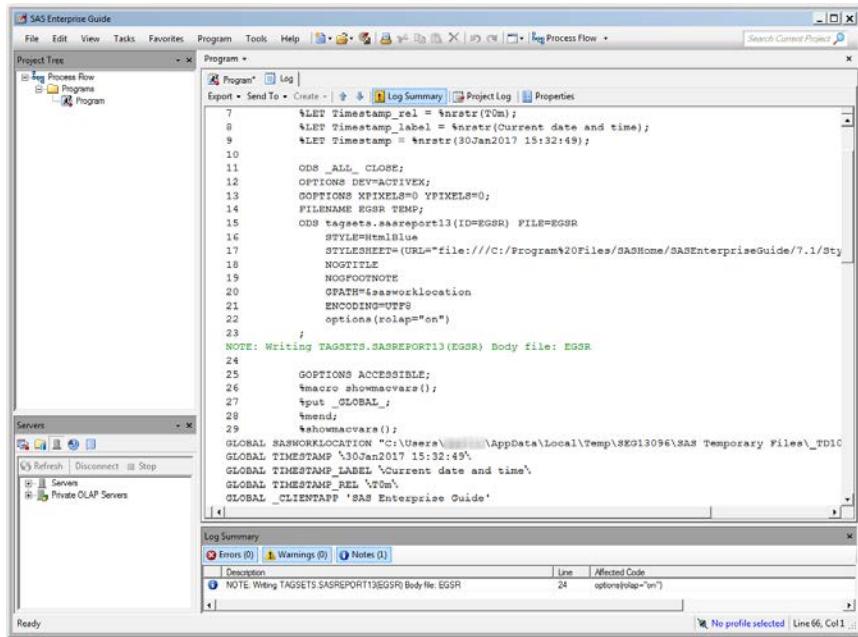


Display 204 - Timestamp Prompt in Prompt Dialog Box

If the user leaves the default value in the single timestamp prompt field, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The %LET statements assign the value specified in the prompt dialog box to the Timestamp* macro variables.

The log of the [Program node using the prompt definition](#) displays the values of the global variables created by the prompt.



The screenshot shows the SAS Enterprise Guide interface. The main window displays a SAS program code. The code includes %LET statements to define timestamp-related macro variables. It also contains ODS _ALL_ CLOSE, GOPTIONS, and other SAS options. The Log Summary window at the bottom shows one note message about writing a report.

```

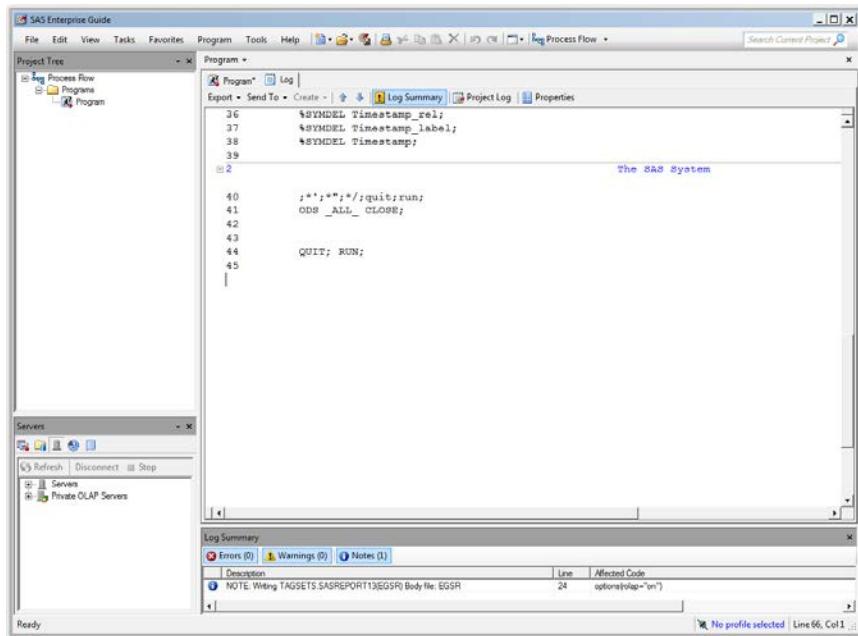
7   %LET timestamp_rel = %nrstr(20m);
8   %LET timestamp_label = %nrstr(Current date and time);
9   %LET timestamp = %nrstr(30Jan2017 15:32:49);
10
11  ODS _ALL_ CLOSE;
12  GOPTIONS DEVMODE=EX;
13  GOPTIONS XPIXELS=0 YPIXELS=0;
14  FILENAME EGSR TEMP;
15  ODS tagsets.sasreport13(ID=EGSR) FILE=EGSR
16    STYLE=HTMLBlue
17    STYLESHEET=(URL="file:///C:/Program%20Files/SASHome/SASEnterpriseGuide/7.1/Sty
18    NOGTITLE
19    NOFOOTNOTE
20    GPATH="&sasworklocation
21    ENCODING="UTF8
22    options(rlabel="on")
23    ;
24 NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR
25
26  GOPTIONS ACCESSIBLE;
27  %macro showmacvars();
28  %put _GLOBAL_;
29  %mend;
30  %showmacvars();
31
32  GLOBAL SASHWORKLOCATION "<:User>\AppData\Local\Temp\SEG13096\SAS Temporary Files\_TD1C
33  GLOBAL TIMESTAMP '30Jan2017 15:32:49';
34  GLOBAL TIMESTAMP_LABEL 'Current date and time';
35  GLOBAL TIMESTAMP_REL '20m';
36  GLOBAL _CLIENTAPP 'SAS Enterprise Guide';

```

Log Summary		
Errors (0)	Warnings (0)	Notes (1)
		Description: NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR Line: 24 Affected Code: options(rlabel="on")

Display 205 - Global Macro Variables and %LET Statements for the Timestamp Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.



The screenshot shows the SAS Enterprise Guide interface. The main window displays a SAS program code. It includes %SYNDEL statements to remove timestamp-related macro variables. The Log Summary window at the bottom shows one note message about writing a report.

```

36  %SYNDEL timestamp_rel;
37  %SYNDEL timestamp_label;
38  %SYNDEL timestamp;
39
40  ;**;*/;quit;run;
41  ODS _ALL_ CLOSE;
42
43
44  QUIT; RUN;
45

```

Log Summary		
Errors (0)	Warnings (0)	Notes (1)
		Description: NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR Line: 24 Affected Code: options(rlabel="on")

Display 206 - %SYMDEL Statements Remove Timestamp* Macro Variables

SAS Studio

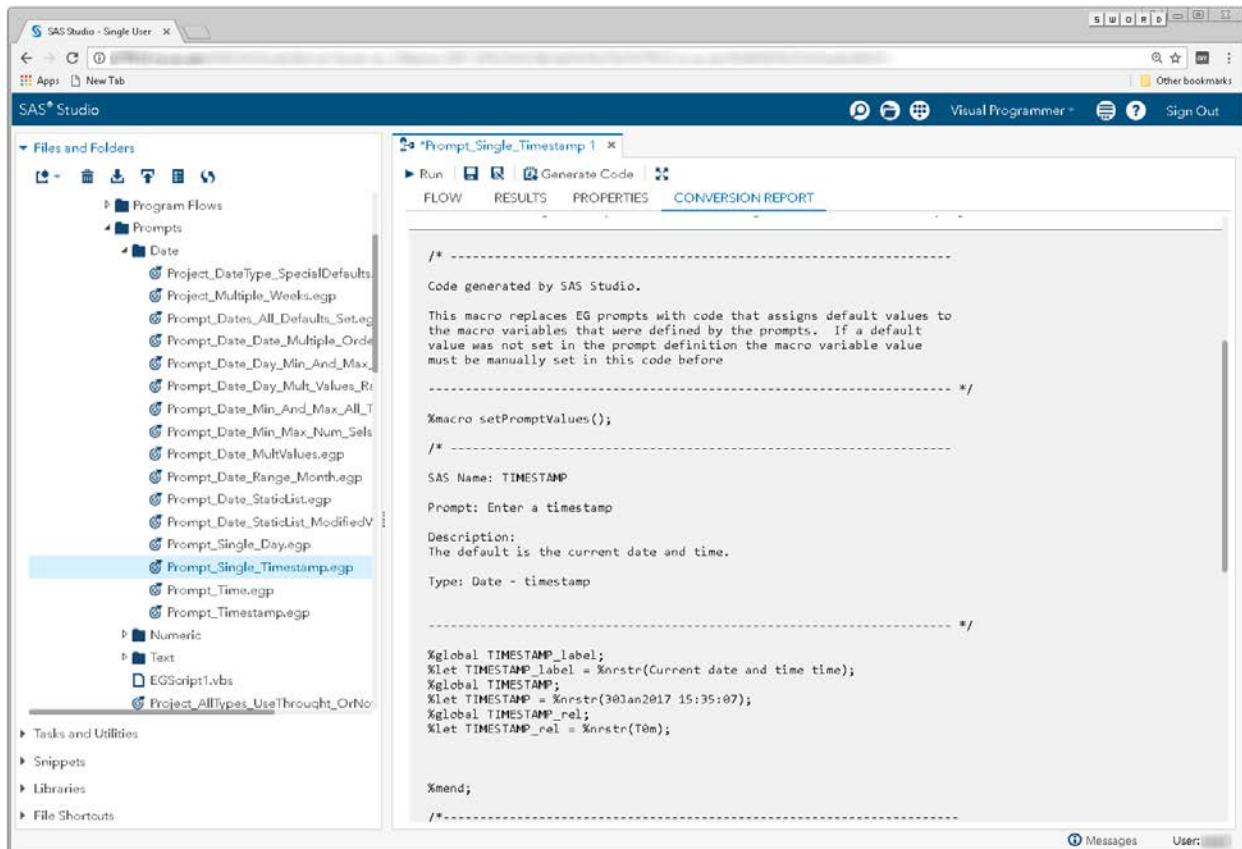
The following display shows the code that is added to the converted Program node for the time prompt in SAS Enterprise Guide.

These global [timestamp macro variables](#) are created:

- `TIMESTAMP_rel`
- `TIMESTAMP_label`
- `TIMESTAMP`

The `%LET` statements assign the default value to the `TIMESTAMP*` macro variables.

If you want to run your process flow using different values, you must manually update the values of the macro variables in the `%LET` statements.



The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. The left sidebar displays a file tree under 'Files and Folders' with various program flow and prompt files listed. A specific file, 'Prompt_Single_Timestamp.egp', is selected and highlighted with a blue background. The main workspace shows the generated macro code:

```
/*
-----  
Code generated by SAS Studio.  
  
This macro replaces EG prompts with code that assigns default values to  
the macro variables that were defined by the prompts. If a default  
value was not set in the prompt definition the macro variable value  
must be manually set in this code before  
-----*/  
  
%macro setPromptValues();  
/* -----  
SAS Name: TIMESTAMP  
  
Prompt: Enter a timestamp  
  
Description:  
The default is the current date and time.  
  
Type: Date - timestamp  
----- */  
  
%global TIMESTAMP_label;  
%let TIMESTAMP_label = %nrstr(Current date and time time);  
%global TIMESTAMP;  
%let TIMESTAMP = %nrstr(30Jan2017 15:35:07);  
%global TIMESTAMP_rel;  
%let TIMESTAMP_rel = %nrstr(T0m);  
  
%mend;  
/*-----*/
```

Display 207 - Code for Timestamp Prompt

Because the **Use prompt value throughout project** option is not checked, the %SYMDEL statements remove the **TIMESTAMP** macro variables.

The screenshot shows the SAS Studio interface with a process flow titled "Prompt_Single_Timestamp 1". The code editor displays the following SAS code:

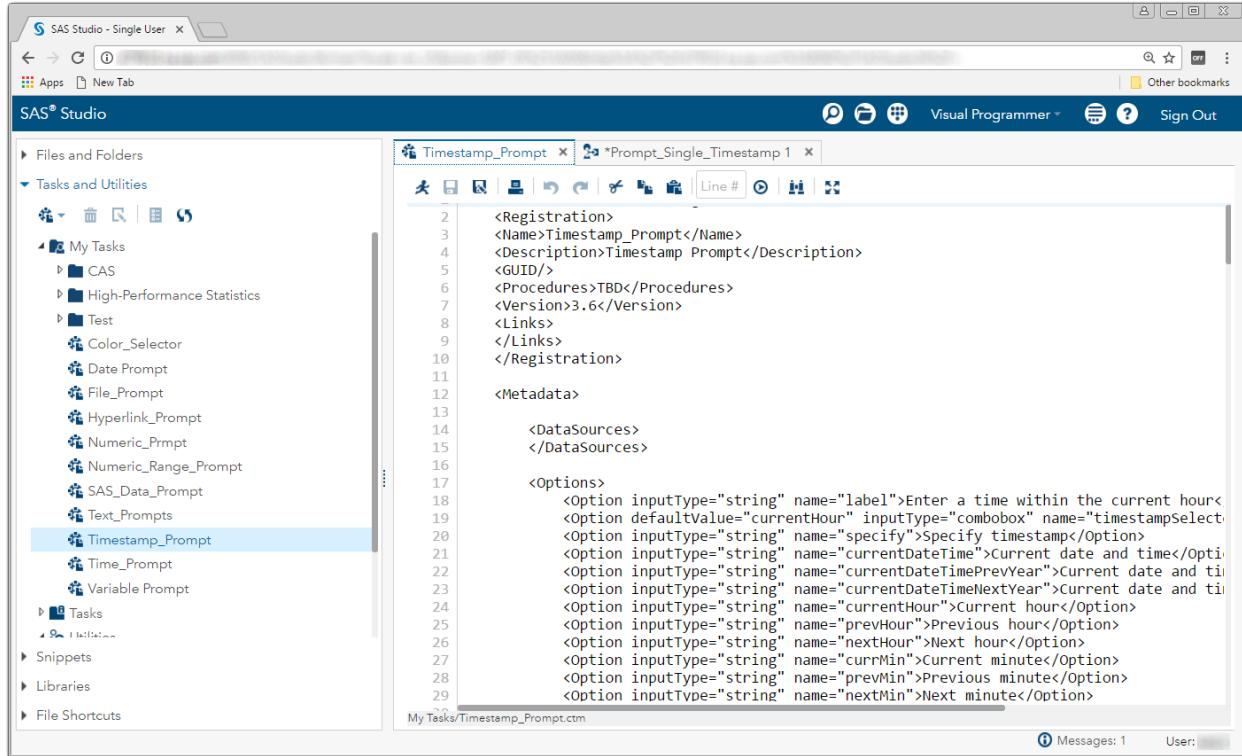
```
/*
-----  
SAS Studio generated code to cleanup macro variables created by prompts.  
"Use prompt throughout" was not checked for these prompts.----- */  
  
%macro cleanupPromptValues();  
%SYMDEL TIMESTAMP_label;  
%SYMDEL TIMESTAMP;  
%SYMDEL TIMESTAMP_rel;  
  
Xmend;  
  
%setPromptValues();  
  
/*----- End SAS generated prompt variable code.----- */  
  
/*----- SAS generated prompt variable cleanup code.----- */  
%cleanupPromptValues();  
/*----- End SAS generated prompt variable cleanup code.----- */  
  
NOTE: Generating node connectors.  
NOTE: Linking nodes connected by data nodes and eliminating data nodes.
```

Below the code, there is a "Process Flow Node Summary" section with a "Steps converted:" list containing "Program".

Display 208 - %SYMDEL Statements Remove **TIMESTAMP*** Macro Variables

Substituting a SAS Studio Task for Timestamp Prompt

1. Create a SAS Studio task with a control that represents the timestamp prompt.
 - Add controls as shown in the `Timestamp_Prompt` task.
 - Set the default value to the default value shown in the generated `setPromptValues()` macro in the converted Program node.
 - Change the strings for the input controls to match the strings specified in the prompt.



The screenshot shows the SAS Studio interface with the "Visual Programmer" tab selected. On the left, the "Tasks and Utilities" tree view has "Timestamp_Prompt" selected. The main editor window displays the XML code for the "Timestamp_Prompt" task. The code includes sections for registration, metadata, and options, with various timestamp-related options like "currentHour", "prevHour", "currMin", etc.

```
<Registration>
<Name>Timestamp_Prompt</Name>
<Description>Timestamp Prompt</Description>
<GUID/>
<Procedures>TBD</Procedures>
<Version>3.6</Version>
<Links>
</Links>
</Registration>

<Metadata>
<DataSources>
</DataSources>
<Options>
<Option inputType="string" name="label">Enter a time within the current hour</Option>
<Option defaultValue="currentHour" inputType="checkbox" name="timestampSelect">Specify timestamp</Option>
<Option inputType="string" name="specify">Specify timestamp</Option>
<Option inputType="string" name="currentDateTime">Current date and time</Option>
<Option inputType="string" name="currentDateTimePrevYear">Current date and time</Option>
<Option inputType="string" name="currentDateTimeNextYear">Current date and time</Option>
<Option inputType="string" name="currentHour">Current hour</Option>
<Option inputType="string" name="prevHour">Previous hour</Option>
<Option inputType="string" name="nextHour">Next hour</Option>
<Option inputType="string" name="currMin">Current minute</Option>
<Option inputType="string" name="prevMin">Previous minute</Option>
<Option inputType="string" name="nextMin">Next minute</Option>
</Options>
</Metadata>
```

Display 209 - Replacement Task for Timestamp Prompt

The following code is an example of a task that could be used as the timestamp prompt.

```
<?xml version="1.0" encoding="UTF-8"?><Task runNLS="never" schemaVersion="5.1">
    <Registration>
        <Name>Timestamp_Prompt</Name>
        <Description>Timestamp Prompt</Description>
        <GUID/>
        <Procedures>TBD</Procedures>
        <Version>3.6</Version>
        <Links></Links>
    </Registration>

    <Metadata>
        <DataSources>
        </DataSources>
    <Options>
        <Option inputType="string" name="label">
            Enter a time within the current hour
        </Option>
        <Option defaultValue="currentHour" inputType="combobox"
            name="timestampSelector">
            Default is current time
        </Option>
        <Option inputType="string" name="specify">Specify timestamp</Option>
        <Option inputType="string" name="currentDateTime">
            Current date and time
        </Option>
        <Option inputType="string" name="currentDateTimePrevYear">
            Current date and time previous year
        <Option inputType="string" name="currentDateTimeNextYear">
            Current date and time next year
        </Option>
        <Option inputType="string" name="currentHour">Current hour</Option>
        <Option inputType="string" name="prevHour">Previous hour</Option>
        <Option inputType="string" name="nextHour">Next hour</Option>
        <Option inputType="string" name="currMin">Current minute</Option>
        <Option inputType="string" name="prevMin">Previous minute</Option>
        <Option inputType="string" name="nextMin">Next minute</Option>
        <Option inputType="string" name="nHoursAgo">N hours ago</Option>
        <Option inputType="string" name="nHoursFromNow">
            N hours from now
        </Option>
        <Option inputType="string" name="nMinsAgo">N minutes ago</Option>
        <Option inputType="string" name="nMinsFromNow">
            N minutes from now
        </Option>
        <Option inputType="inputtext" name="timeInput"
            promptMessage="Example: 14Feb2017 01:23:45 PM" />
        <Option defaultValue="N" inputType="numbertext"
            invalidMessage="Invalid value. Enter a positive integer."
            minValue="0"
            missingMessage="Enter number of minutes"
            name="numMinutes"
            promptMessage="Enter number of minutes"
            rangeMessage=
                "This number is out of range. Enter a positive number.">
            Number of minutes:
        </Option>
    </Options>
</Metadata>
```

```

        <Option defaultValue="N" inputType="numbertext"
            invalidMessage="Invalid value. Enter a positive integer."
            minValue="0" missingMessage="Enter number of hours"
            name="numHours"
            promptMessage="Enter number of hours"
            rangeMessage=
                "This number is out of range. Enter a positive number.">
            Number of hours:
        </Option>

    </Options>

</Metadata>

<UI>

    <OptionItem option="label"/>
    <OptionChoice options="timestampSelector">
        <OptionItem option="specify"/>
        <OptionItem option="currentDateTime"/>
        <OptionItem option="currentDateTimePrevYear"/>
        <OptionItem option="currentDateTimeNextYear"/>
        <OptionItem option="currentHour"/>
        <OptionItem option="prevHour"/>
        <OptionItem option="nextHour"/>
        <OptionItem option="currMin"/>
        <OptionItem option="prevMin"/>
        <OptionItem option="nextMin"/>
        <OptionItem option="nHoursAgo"/>
        <OptionItem option="nHoursFromNow"/>
        <OptionItem option="nMinsAgo"/>
        <OptionItem option="nMinsFromNow"/>
    </OptionChoice>

    <OptionItem option="timeInput"/>
    <OptionItem option="numMinutes"/>
    <OptionItem option="numHours"/>

</UI>

<Dependencies>

    <Dependency condition="($timestampSelector == 'specify')">
        <Target action="hide" conditionResult="false" option="timeInput"/>
        <Target action="show" conditionResult="true" option="timeInput"/>
    </Dependency>
    <Dependency condition="(( $timestampSelector == 'nMinsAgo' ) ||
        ($timestampSelector == 'nMinsFromNow'))">
        <Target action="hide" conditionResult="false" option="numMinutes"/>
        <Target action="show" conditionResult="true" option="numMinutes"/>
    </Dependency>
    <Dependency condition="(( $timestampSelector == 'nHoursAgo' ) ||
        ($timestampSelector == 'nHoursFromNow'))">
        <Target action="hide" conditionResult="false" option="numHours"/>
        <Target action="show" conditionResult="true" option="numHours"/>
    </Dependency>

</Dependencies>

```

```

<CodeTemplate>
<! [CDATA[

%global timestamp;
%global timestamp_rel;
%global timestamp_label;

#if ($timestampSelector == 'specify')
  %let timestamp = $timeInput;
  %let timestamp_label = $timeInput; /* Format this as you see fit */
  %symdel timestamp_rel;
#else

#if ($timestampSelector == 'currentDateTime')
  %let timestamp = %sysfunc(datetime(), DATETIME20.);
  %let timestamp = %sysfunc(catx(%str( ), %substr(&timestamp,1,9),
                                %substr(&timestamp,11,8)));
  %let timestamp_label = Current date and time;
  %let timestamp_rel = T0m;
#else

#if ($timestampSelector == 'currentDateTimePrevYear')
  %global year;
  %global datetime;
  %let datetime = %sysfunc(datetime());
  %let datetime = %sysfunc(intnx(dtyear, &datetime, -1), DATETIME20.);
  %let year = %substr(&datetime, 6,4);

  %let datetime = %sysfunc(datetime(), DATETIME20.);
  %let timestamp = %sysfunc(catx(%str(), %substr(&datetime, 1, 5), &year));
  %let timestamp = %sysfunc(catx(%str( ), &timestamp, %substr(&datetime, 11, 8)));
  %symdel year;
  %symdel datetime;

  %let timestamp_label = Current date and time previous year;
  %let timestamp_rel = T-1Y;
#else

#if ($timestampSelector == 'currentDateTimeNextYear')
  %global year;
  %global datetime;
  %let datetime = %sysfunc(datetime());
  %let datetime = %sysfunc(intnx(dtyear, &datetime, 1), DATETIME20.);
  %let year = %substr(&datetime, 6,4);

  %let datetime = %sysfunc(datetime(), DATETIME20.);
  %let timestamp = %sysfunc(catx(%str(), %substr(&datetime, 1, 5), &year));
  %let timestamp = %sysfunc(catx(%str( ), &timestamp, %substr(&datetime, 11, 8)));
  %symdel year;
  %symdel datetime;
  %let timestamp_label = Current date and time next year;
  %let timestamp_rel = T1Y;
#else

#if ($timestampSelector == 'currentHour')
  %let timestamp = %sysfunc(datetime());
  %let timestamp = %sysfunc(intnx(hour, &timestamp, 0, b), DATETIME20.);
  %let timestamp = %sysfunc(catx(%str( ), %substr(&timestamp,1,9),
                                %substr(&timestamp,11,8)));
  %let timestamp_label = Current hour;
  %let timestamp_rel = H0H;
#else

```

```

#if ($timestampSelector == 'currMin')
    %let timestamp = %sysfunc(datetime());
    %let timestamp = %sysfunc(intnx(minute, &timestamp, 0, b), DATETIME20.);
    %let timestamp = %sysfunc(catx(%str( ), %substr(&timestamp,1,9),
                                  %substr(&timestamp,11,8)));
    %let timestamp_label = Current minute;
    %let timestamp_rel = mom;
#else

#if ($timestampSelector == 'prevHour')
    %let timestamp = %sysfunc(datetime());
    %let timestamp = %sysfunc(intnx(dthour, &timestamp, -1), DATETIME20.);
    %let timestamp = %sysfunc(catx(%str( ), %substr(&timestamp,1,9),
                                  %substr(&timestamp,11,8)));
    %let timestamp_label = Previous hour;
    %let timestamp_rel = H-1H;
#endif

#if ($timestampSelector == 'nextHour')
    %let timestamp = %sysfunc(datetime());
    %let timestamp = %sysfunc(intnx(dthour, &timestamp, 1), DATETIME20.);
    %let timestamp = %sysfunc(catx(%str( ), %substr(&timestamp,1,9),
                                  %substr(&timestamp,11,8)));
    %let timestamp_label = Next hour;
    %let timestamp_rel = H1H;
#endif

#if ($timestampSelector == 'prevMin')
    %let timestamp = %sysfunc(datetime());
    %let timestamp = %sysfunc(intnx(dtminute, &timestamp, -1), DATETIME20.);
    %let timestamp = %sysfunc(catx(%str( ), %substr(&timestamp,1,9),
                                  %substr(&timestamp,11,8)));
    %let timestamp_label = Previous minute;
    %let timestamp_rel = m-1m;
#endif

#if ($timestampSelector == 'nextMin')
    %let timestamp = %sysfunc(datetime());
    %let timestamp = %sysfunc(intnx(dtminute, &timestamp, 1), DATETIME20.);
    %let timestamp = %sysfunc(catx(%str( ), %substr(&timestamp,1,9),
                                  %substr(&timestamp,11,8)));
    %let timestamp_label = Next minute;
    %let timestamp_rel = m1m;
#endif

#if ($timestampSelector == 'nHoursAgo')
    %let timestamp = %sysfunc(datetime());
    %let timestamp = %sysfunc(intnx(dtsecond, &timestamp, -${numHours}*3600),
                             DATETIME20.);
    %let timestamp = %sysfunc(catx(%str( ), %substr(&timestamp,1,9),
                                  %substr(&timestamp,11,8)));
    %let timestamp_label = ${numHours} hours ago;
    %let timestamp_rel = T-${numHours}H;
#endif

```

```

#if ($timestampSelector == 'nHoursFromNow')
    %let timestamp = %sysfunc(datetime());
    %let timestamp = %sysfunc(intnx(dtsecond, &timestamp, ${numHours}*3600),
                                DATETIME20.);
    %let timestamp = %sysfunc(catx(%str( ), %substr(&timestamp,1,9),
                                %substr(&timestamp,11,8)));
    %let timestamp_label = $numHours hours from now;
    %let timestamp_rel = T${numHours}H;
#end

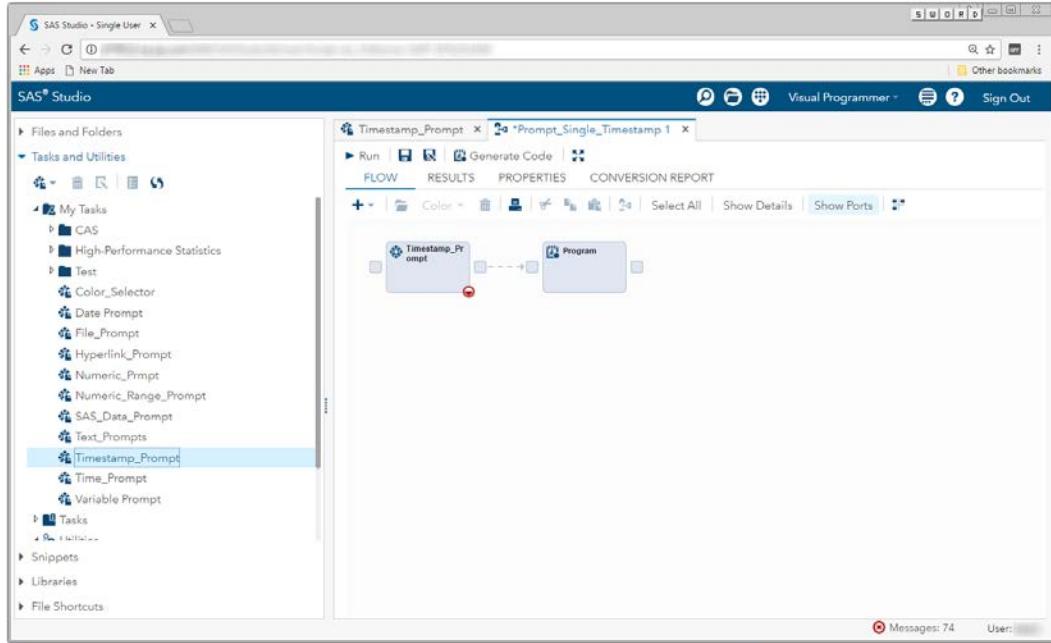
#if ($timestampSelector == 'nMinsAgo')
    %let timestamp = %sysfunc(datetime());
    %let timestamp = %sysfunc(intnx(dtsecond, &timestamp, -${numMinutes}*60),
                                DATETIME20.);
    %let timestamp = %sysfunc(catx(%str( ), %substr(&timestamp,1,9),
                                %substr(&timestamp,11,8)));
    %let timestamp_label = $numMinutes minutes ago;
    %let timestamp_rel = T-${numMinutes}m;
#end

#if ($timestampSelector == 'nMinsFromNow')
    %let timestamp = %sysfunc(datetime());
    %let timestamp = %sysfunc(intnx(dtsecond, &timestamp, ${numMinutes}*60),
                                DATETIME20.);
    %let timestamp = %sysfunc(catx(%str( ), %substr(&timestamp,1,9),
                                %substr(&timestamp,11,8)));
    %let timestamp_label = $numMinutes minutes from now;
    %let timestamp_rel = T${numMinutes}m;
#end

#end
#end
#end
#end
#end
#end
[]>
</CodeTemplate>
</Task>

```

2. Save the prompt replacement task to your **My Tasks** folder.
3. Drag the task from **My Tasks** into your converted process flow.
4. Link the output port of the task to the input port of the converted Program node.



Display 210 – Timestamp_Prompt Task Linked to Program Node

5. Comment out the `%setPromptValues()` macro call from the converted Program node. The macro code generated by the Timestamp_Prompt task replaces this code.

The screenshot shows the SAS Studio interface with the 'Program' node selected. The code editor displays the macro code generated by the Timestamp_Prompt task. A specific line of code, `/*----- *%setPromptValues();----- */`, is highlighted with a light blue selection bar. This line is part of a larger block of SAS code that includes cleanup logic for macro variables. The code editor has tabs for 'CODE', 'LOG', 'RESULTS', and 'NODE'. The status bar at the bottom right indicates 'Messages: 77' and 'User:'.

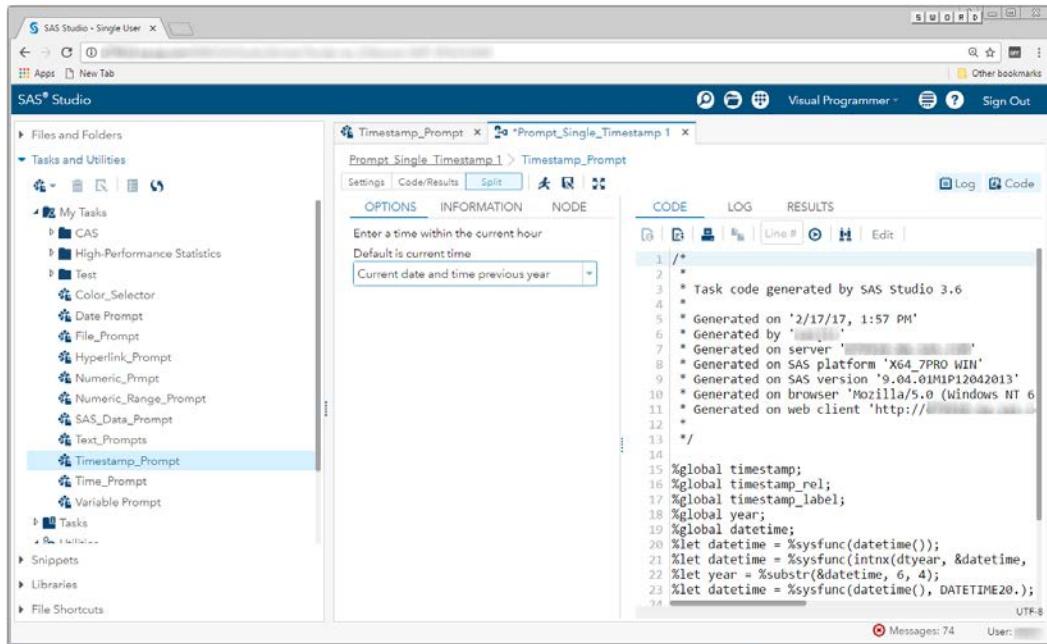
```

37 %mend;
38
39 /*-----
40
41 SAS Studio generated code to cleanup macro variables created by prompts.
42 "Use prompt throughout" was not checked for these prompts.
43 -----
44
45 %macro cleanupPromptValues();
46 %SYMDEL TIMESTAMP_label;
47 %SYMDEL TIMESTAMP;
48 %SYMDEL TIMESTAMP_rel;
49
50 %mend;
51
52 /*----- *%setPromptValues();----- */
53
54 /*----- End SAS generated prompt variable code.----- */
55
56 %put _GLOBAL_;
57
58 /*----- SAS generated prompt variable cleanup code.----- */
59 %cleanupPromptValues();
60 /*----- End SAS generated prompt variable cleanup code.----- */
61

```

Display 211 – Commented out %setPromptValues in Program Node

To run your flow with a different timestamp value than the default value, open the `Timestamp_Prompt` node and specify a different timestamp.



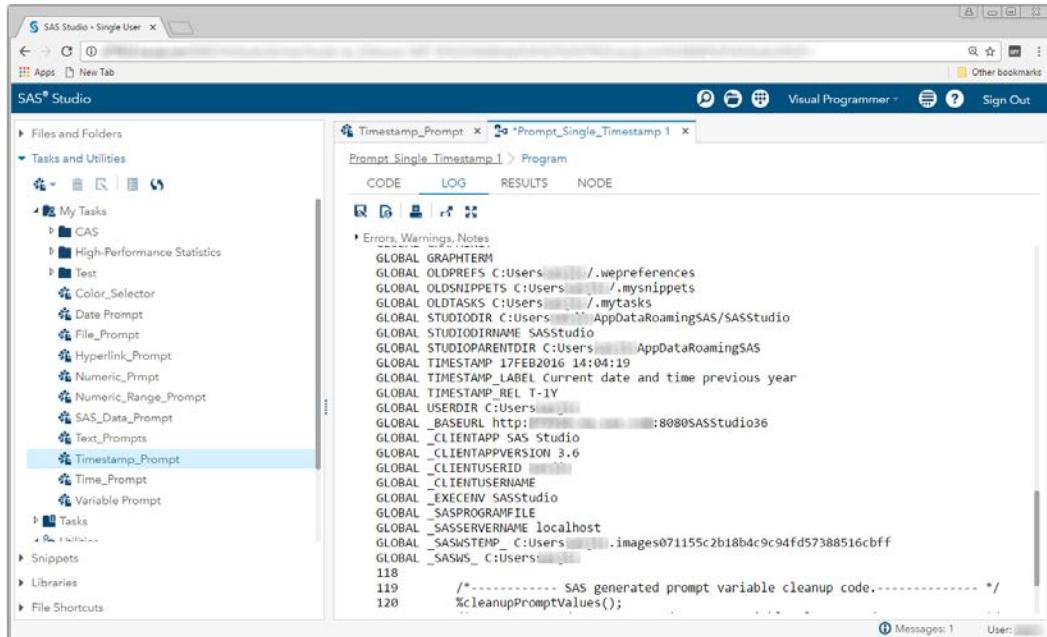
The screenshot shows the SAS Studio interface with the Visual Programmer tab active. A node named `Prompt_Single_Timestamp_1 > Timestamp_Prompt` is selected. In the properties pane, under the `NODE` tab, there is a dropdown menu set to `Current date and time previous year`. The code editor displays SAS code for generating a timestamp:

```

1 /*
2  * Task code generated by SAS Studio 3.6
3  */
4 *
5 * Generated on '2/17/17, 1:57 PM'
6 * Generated by '_____'
7 * Generated on server '_____'
8 * Generated on SAS platform 'X64_7PRO WIN'
9 * Generated on SAS version '9.04.01MP12042013'
10 * Generated on browser 'Mozilla/5.0 (Windows NT 6
11 * Generated on web client 'http://_____
12 *
13 */
14
15 %global timestamp;
16 %global timestamp_rel;
17 %global timestamp_label;
18 %global year;
19 %global datetime;
20 %let datetime = %sysfunc(datetime());
21 %let datetime = %sysfunc(intnx(dyyear, &datetime,
22 %let year = %substr(&datetime, 6, 4);
23 %let datetime = %sysfunc(datetime(), DATETIME20.);
```

Display 212 - Timestamp Prompt Task

When you run the process flow, the global `Timestamp*` variables are set to the values specified in the task.



The screenshot shows the SAS Studio interface with the Visual Programmer tab active. A node named `Prompt_Single_Timestamp_1 > Program` is selected. The code editor displays SAS code for generating a timestamp, and the results pane shows the updated global variables:

```

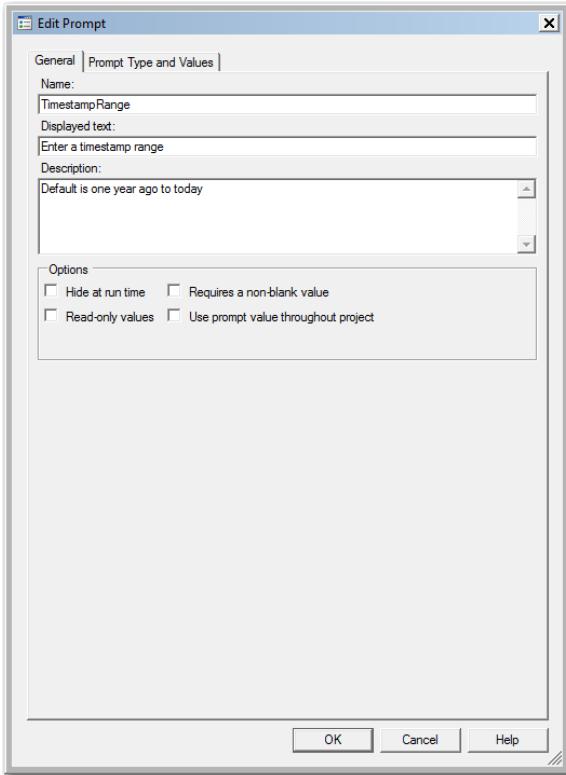
GLOBAL GRAPHTERM
GLOBAL OPRREFS C:/Users/____/.wepreferences
GLOBAL OLDSNIPPETS C:/Users/____/.mysnippets
GLOBAL OLDTASKS C:/Users/____/.mytasks
GLOBAL STUDIODIR C:/Users/____/AppDataRoamingSAS/SASStudio
GLOBAL STUDIODIRNAME SASstudio
GLOBAL STUDIOPARENTDIR C:/Users/____/AppDataRoamingSAS
GLOBAL TIMESTAMP 17FEB2016 14:04:19
GLOBAL TIMESTAMP_LABEL Current date and time previous year
GLOBAL TIMESTAMP_REL T-1Y
GLOBAL USERDIR C:/Users/____
GLOBAL _BASEURL http://_____:8080SASstudio36
GLOBAL _CLIENTAPP SAS Studio
GLOBAL _CLIENTAPPVERSION 3.6
GLOBAL _CLIENTUSERID _____
GLOBAL _CLIENTUSERNAME _____
GLOBAL _EXECENV SASstudio
GLOBAL _SASPROGRAMFILE _____
GLOBAL _SASSERVERNAME localhost
GLOBAL _SASSTEMP_C:/Users/____/.images071155c2b18b4c9c94fd57388516cbff
GLOBAL _SASW_C:/Users/_____
118
119 /*----- SAS generated prompt variable cleanup code.-----*/
120 %cleanupPromptValues();
```

Display 213 – Timestamp Prompt Variables with Updated Values

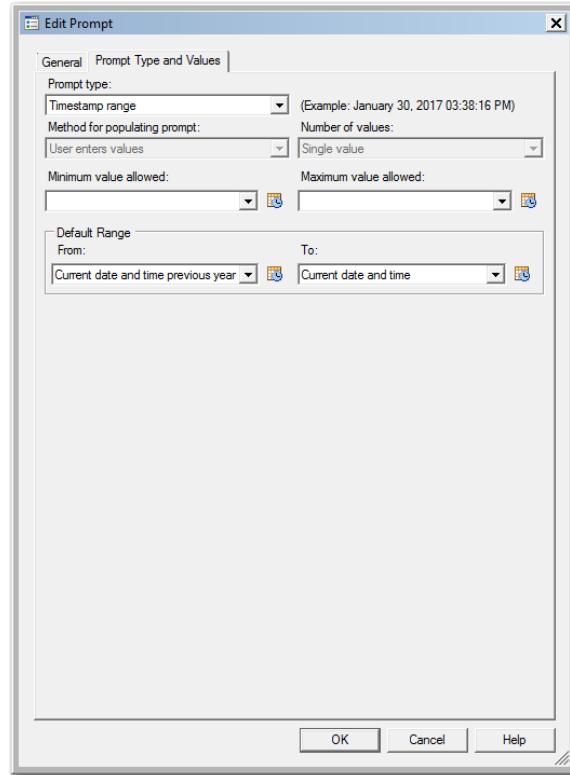
Timestamp Range

SAS Enterprise Guide

In this example, a timestamp range prompt named TimestampRange is defined as shown in the following two displays.

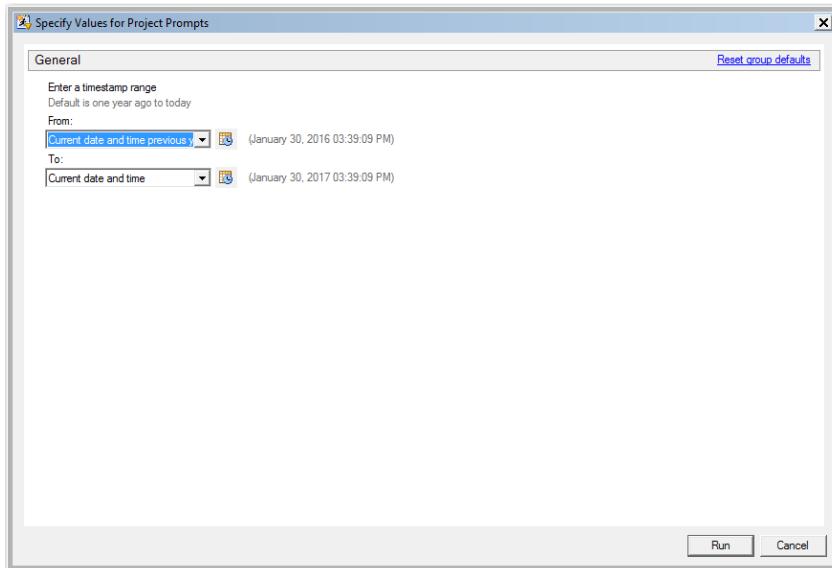


**Display 214 - General Properties for
Timestamp Range Prompt**



**Display 215 - Type and Values for Timestamp
Range Prompt**

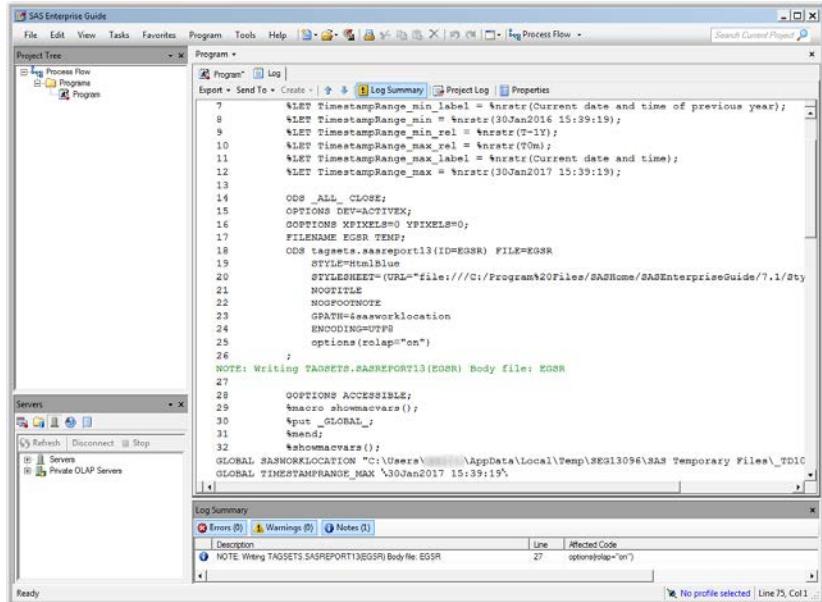
When you run the Program node that depends on this prompt, the following dialog box appears:



Display 216 - Timestamp Range Prompt in Prompt Dialog Box

If the user leaves the default value in the timestamp range prompt field, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The %LET statements assign the values specified in the prompt dialog box to the TimestampRange* macro variables.

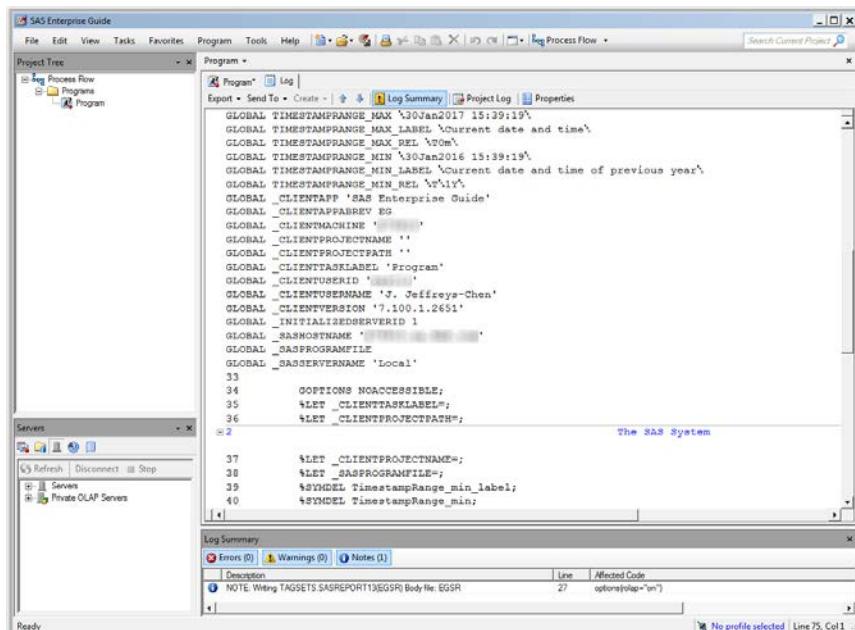


```

7   %LET TimestampRange_min_label = &nstrt(Current date and time of previous year);
8   %LET TimestampRange_min = &nstrt(30Jan2016 15:39:19);
9   %LET TimestampRange_min_rel = &nstrt(7-1Y);
10  %LET TimestampRange_max_rel = &nstrt(7D);
11  %LET TimestampRange_max_label = &nstrt(Current date and time);
12  %LET TimestampRange_max = &nstrt(30Jan2017 15:39:19);
13
14  ODS _ALL_ CLOSE;
15  OPTIONS DEV=ACTIVE;
16  OPTIONS XPIXELS=0 YPIXELS=0;
17  FILENAME EGSR TEMP;
18  ODS tagsets.sasreport13 ID=EGSR FILE=EGSR
19    STYLE=HTMLBlue
20    STYLESHEET=(URL="file:///C:/Program%20Files/SASHome/SASEnterpriseGuide/7.1/Sty
21    NOGTITLE
22    NOFOOTNOTE
23    GPATH=&sasworklocation
24    ENCODING=UTF8
25    options(rlsep="on")
26
27 NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR
28
29  OPTIONS ACCESSIBLE;
30  %macro showmacrovars();
31  %put _GLOBAL_;
32  %end;
33
34  %showmacrovars();
35
36 GLOBAL SASHWORKLOCATION "<C:\Users\> \AppData\Local\Temp\SEG13096\SAS Temporary Files\>_TDIC";
37 GLOBAL TIMESTAMPRANGE_MAX '30Jan2017 15:39:19';
38 
```

Display 217 - %LET Statements for Timestamp Range Prompt

The log of the [Program node using the prompt definition](#) displays the values of the global variables created by the prompt.

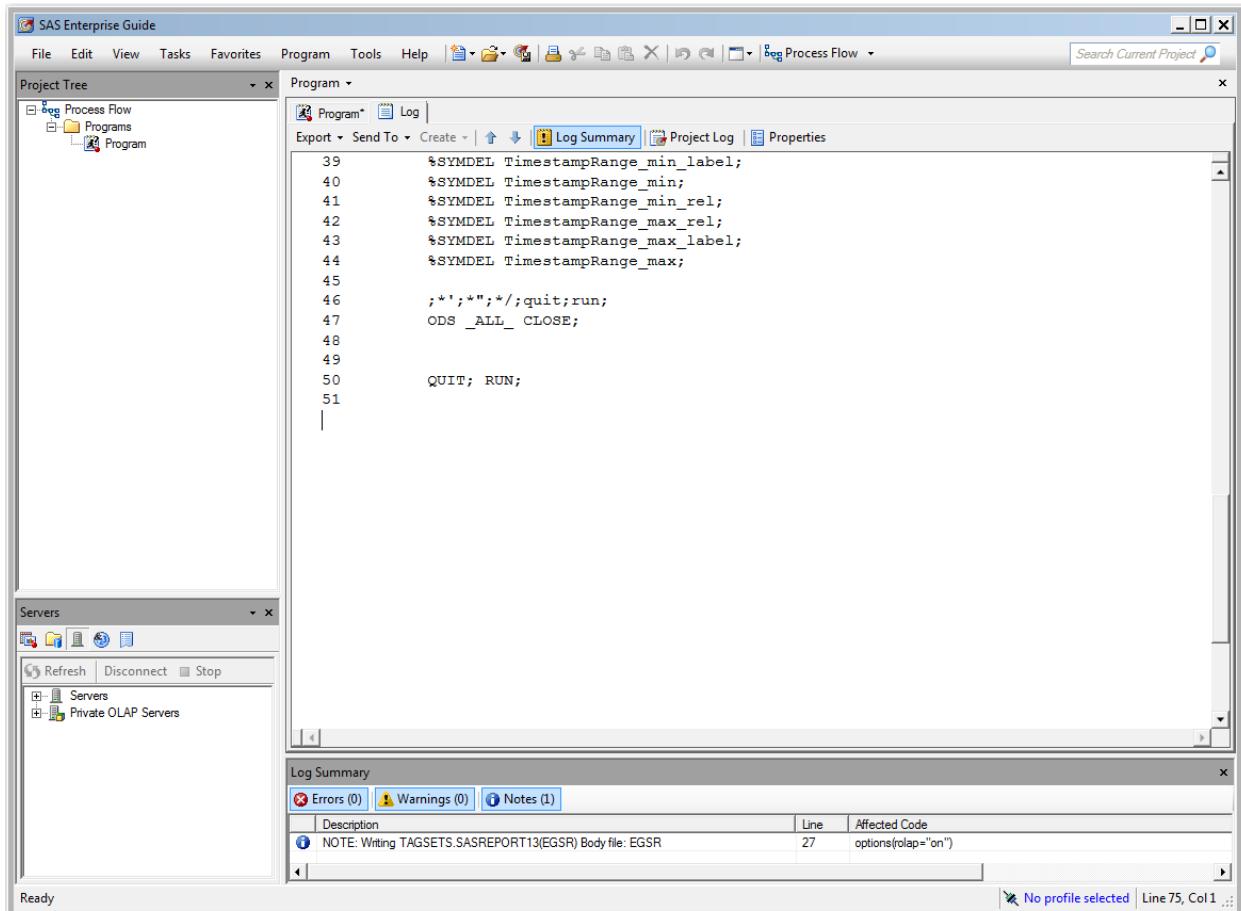


```

1  GLOBAL TIMESTAMPRANGE_MAX '30Jan2017 15:39:19';
2  GLOBAL TIMESTAMPRANGE_MAX_LABEL 'Current date and time';
3  GLOBAL TIMESTAMPRANGE_MAX_REL '7D';
4  GLOBAL TIMESTAMPRANGE_MIN '30Jan2016 15:39:19';
5  GLOBAL TIMESTAMPRANGE_MIN_LABEL 'Current date and time of previous year';
6  GLOBAL TIMESTAMPRANGE_MIN_REL '7Y';
7  GLOBAL _CLIENTAPP 'SAS Enterprise Guide';
8  GLOBAL _CLIENTAPPFILE EG
9  GLOBAL _CLIENTMACHINE '';
10  GLOBAL _CLIENTPROJECTNAME '';
11  GLOBAL _CLIENTPROJECTPATH '';
12  GLOBAL _CLIENTTASKLABEL 'Program';
13  GLOBAL _CLIENTUSERID '';
14  GLOBAL _CLIENTUSERNAME 'J. Jeffreys-Chen';
15  GLOBAL _CLIENTVERSION '7.100.1.2651';
16  GLOBAL _INITIALIZESEVERERID 1;
17  GLOBAL _SASNOSTNAME '';
18  GLOBAL _SASPROGRAMFILE '';
19  GLOBAL _SASSERVERNAME 'Local';
20  GLOBAL _SASVIEWNAME '';
21
22  OPTIONS NONOTES;
23  %LET _CLIENTPROJECTNAME="";
24  %LET _SASPROGRAMFILE="";
25  %SYNCDL TimestampRange_min_label;
26  %SYNCDL TimestampRange_min;
27
28  %LET _CLIENTPROJECTNAME="";
29  %LET _SASPROGRAMFILE="";
30  %SYNCDL TimestampRange_min_label;
31  %SYNCDL TimestampRange_min;
32
33
34
35
36
37
38
39
40 
```

Display 218 - Global Macro Variables for Timestamp Range Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.



The screenshot shows the SAS Enterprise Guide interface. The top menu bar includes File, Edit, View, Tasks, Favorites, Program, Tools, Help, and a Process Flow button. The Project Tree on the left shows a 'Program' node under 'Programs'. The main workspace contains a 'Program' tab with the following code:

```
39      %SYMDEL TimestampRange_min_label;
40      %SYMDEL TimestampRange_min;
41      %SYMDEL TimestampRange_min_rel;
42      %SYMDEL TimestampRange_max_rel;
43      %SYMDEL TimestampRange_max_label;
44      %SYMDEL TimestampRange_max;
45
46      ;*' ;*/;quit;run;
47      ODS _ALL_ CLOSE;
48
49
50      QUIT; RUN;
51
```

Below the code editor is a 'Servers' pane showing 'Servers' and 'Private OLAP Servers'. The bottom right corner displays the 'Log Summary' pane with the following table:

Description	Line	Affected Code
NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR	27	options(rolap="on")

The status bar at the bottom right indicates 'No profile selected | Line 75, Col 1 ::'.

Display 219 - %SYMDEL Statements Remove TimestampRange* Macro Variables

SAS Studio

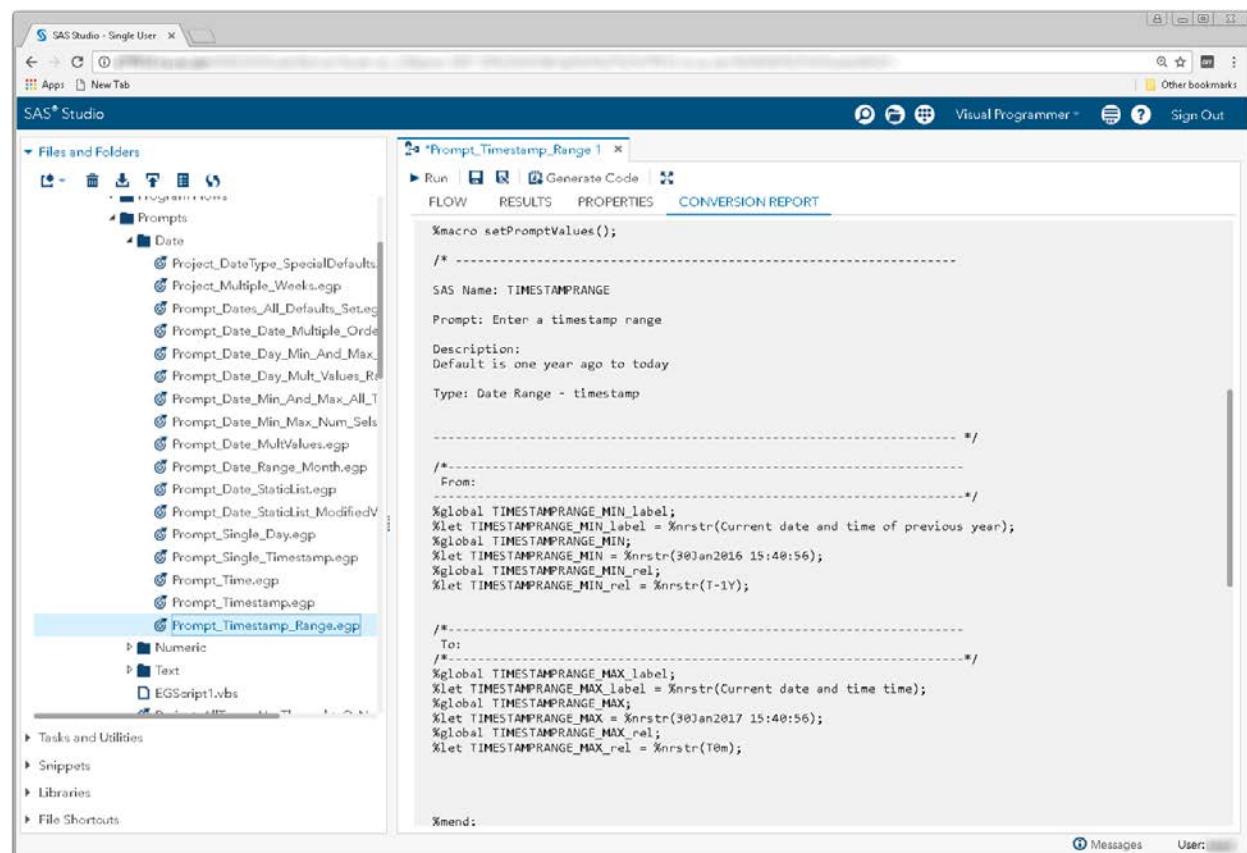
The following display shows the code that is added to the converted Program node for the timestamp range prompt in SAS Enterprise Guide.

These global macro variables are created:

- `TIMESTAMPRANGE_MIN_rel`
- `TIMESTAMPRANGE_MIN_label`
- `TIMESTAMPRANGE_MIN`
- `TIMESTAMPRANGE_MAX_rel`
- `TIMESTAMPRANGE_MAX_label`
- `TIMESTAMPRANGE_MAX`

The `%LET` statements assign the default value to the `TIMESTAMPRANGE*` macro variables.

If you want to run your process flow using different values for the `TIMESTAMPRANGE` prompt, you must manually update the values of the macro variables in the `%LET` statements.



The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. The code editor displays the following macro code:

```
%macro setPromptValues();  
/* -----  
SAS Name: TIMESTAMPRANGE  
Prompt: Enter a timestamp range  
Description:  
Default is one year ago to today  
Type: Date Range - timestamp  
----- */  
/*-----  
From:  
-----*/  
%global TIMESTAMPRANGE_MIN_label;  
%let TIMESTAMPRANGE_MIN_label = %nrstr(Current date and time of previous year);  
%global TIMESTAMPRANGE_MIN;  
%let TIMESTAMPRANGE_MIN = %nrstr(30Jan2016 15:40:56);  
%global TIMESTAMPRANGE_MIN_rel;  
%let TIMESTAMPRANGE_MIN_rel = %nrstr(T-1Y);  
  
/*-----  
To:  
-----*/  
%global TIMESTAMPRANGE_MAX_label;  
%let TIMESTAMPRANGE_MAX_label = %nrstr(Current date and time time);  
%global TIMESTAMPRANGE_MAX;  
%let TIMESTAMPRANGE_MAX = %nrstr(30Jan2017 15:40:56);  
%global TIMESTAMPRANGE_MAX_rel;  
%let TIMESTAMPRANGE_MAX_rel = %nrstr(T0m);  
  
%mend;
```

Display 220 - Macro Code for Timestamp Range Prompt

Because the **Use prompt value throughout project** is not checked, the %SYMDEL statements remove the %TSTAMP RANGE macro variables.

The screenshot shows the SAS Studio interface with the 'Prompt_Timestamp_Range.1' file open in the code editor. The code editor has tabs for FLOW, RESULTS, PROPERTIES, and CONVERSION REPORT, with CONVERSION REPORT selected. The code itself is as follows:

```
/*
-----  
SAS Studio generated code to cleanup macro variables created by prompts.  
"Use prompt throughout" was not checked for these prompts.----- */  
  
%macro cleanupPromptValues();  
%SYMDEL TSTAMP RANGE_MIN_label;  
%SYMDEL TSTAMP RANGE_MIN;  
%SYMDEL TSTAMP RANGE_MIN_rel;  
%SYMDEL TSTAMP RANGE_MAX_label;  
%SYMDEL TSTAMP RANGE_MAX;  
%SYMDEL TSTAMP RANGE_MAX_rel;  
  
%mend;  
  
%setPromptValues();  
/*----- End SAS generated prompt variable code.----- */  
  
/*----- SAS generated prompt variable cleanup code.----- */  
%cleanupPromptValues();  
/*----- End SAS generated prompt variable cleanup code.----- */  
  
NOTE: Generating node connectors.  
NOTE: Linking nodes connected by data nodes and eliminating data nodes.  
  
Process Flow Node Summary  
None evaluated.
```

The code editor also displays notes about generating node connectors and linking nodes. At the bottom, it shows a 'Process Flow Node Summary' section with the message 'None evaluated.'

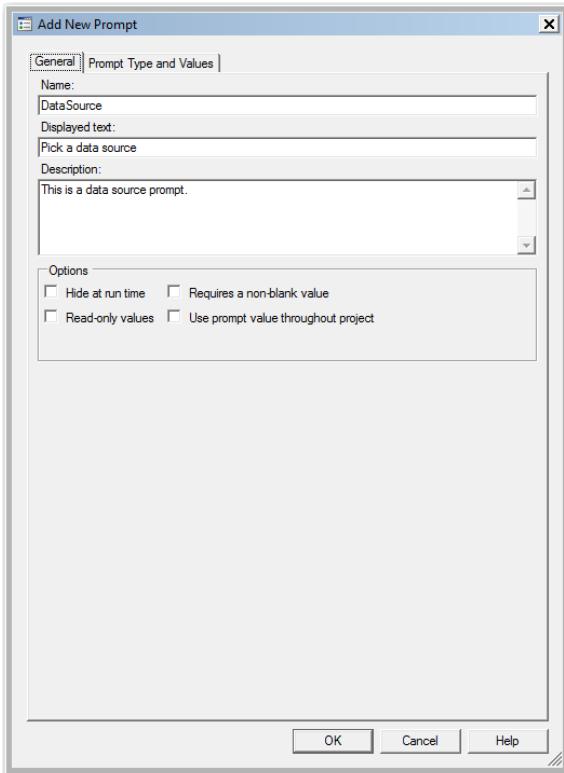
Display 221 - %SYMDEL Statements Remove the %TSTAMP RANGE* Macro Variables

Data

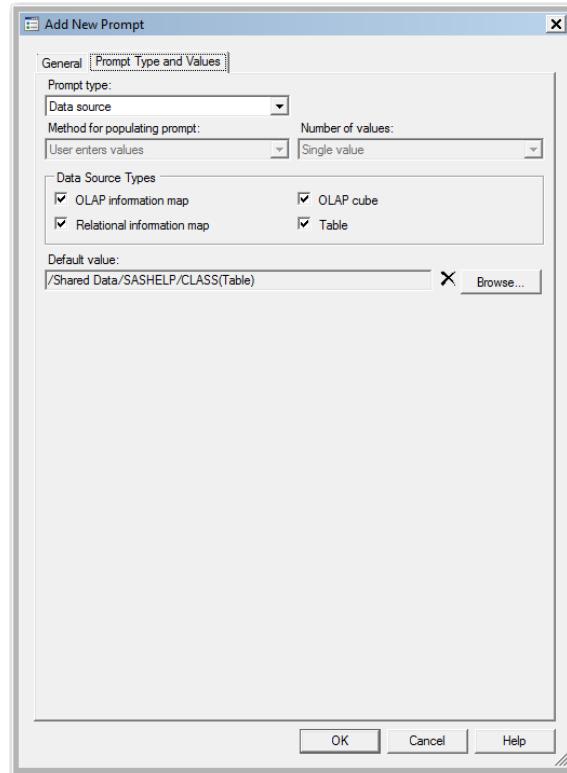
Data Source

SAS Enterprise Guide

In this example, a data source prompt named DataSource is defined as shown in the following two displays.

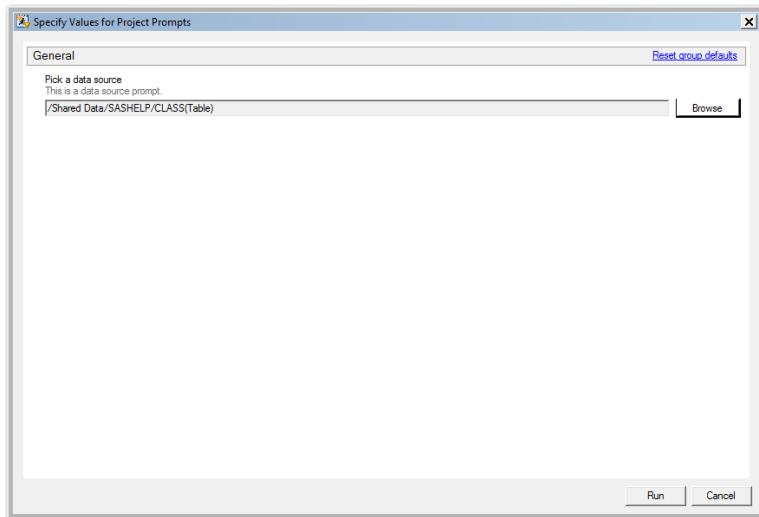


Display 222 - General Properties for Data Source Prompt



Display 223 - Type and Values for Data Source Prompt

When you run the Program node that depends on the prompt, the following dialog box appears:

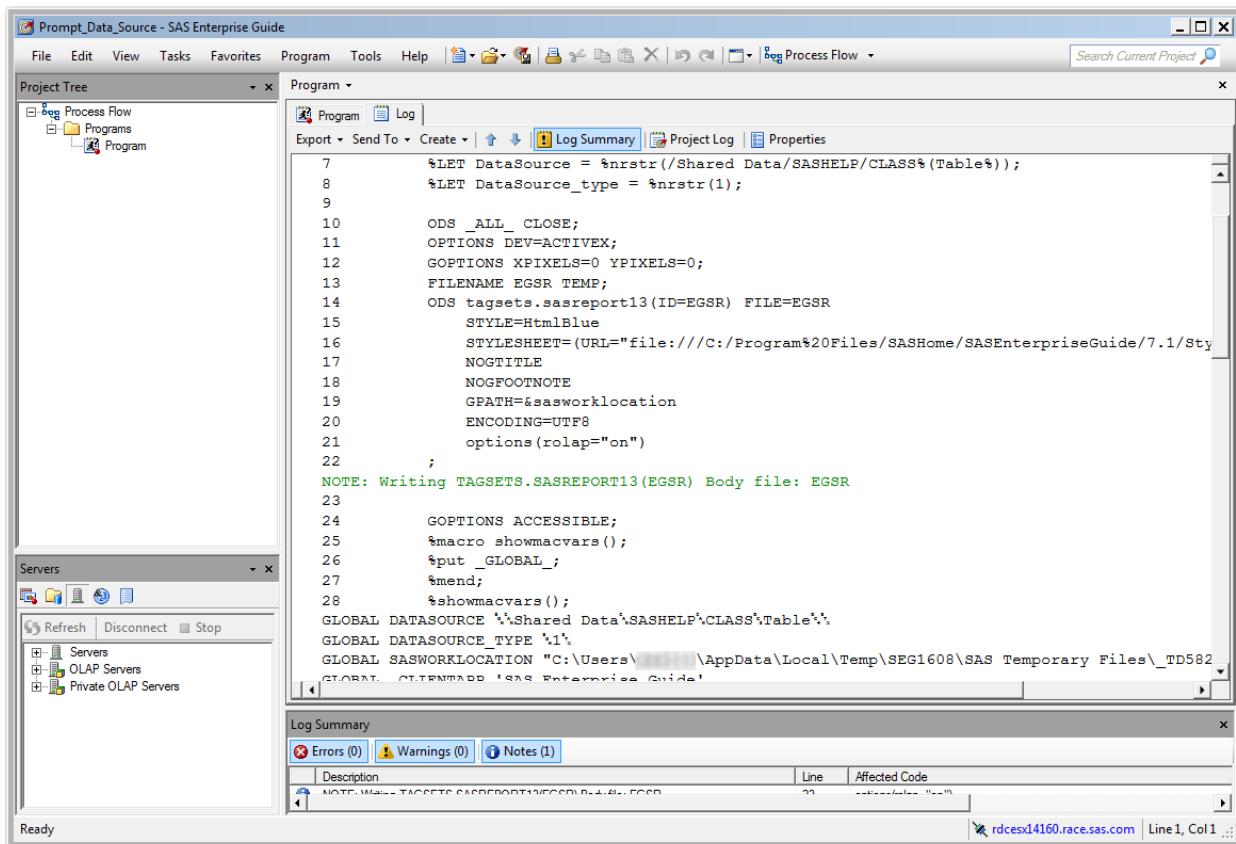


Display 224 - Data Source Prompt in Prompt Dialog Box

If the user leaves the default value in the data source value prompt field, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The %LET statements assign the values specified in the prompt dialog box to the DataSource* macro variables.

The log of the [Program node using the prompt definition](#) displays the values of the global variables created by the prompt.



The screenshot shows the SAS Enterprise Guide interface with the title bar "Prompt_Data Source - SAS Enterprise Guide". The menu bar includes File, Edit, View, Tasks, Favorites, Program, Tools, Help, and a Process Flow dropdown. The Project Tree on the left shows a "Program" node under "Process Flow". The main workspace contains a "Program" tab with the following code:

```
7      %LET DataSource = %nrstr(/Shared Data/SASHelp/CLASS%(Table%));
8      %LET DataSource_type = %nrstr(1);
9
10     ODS _ALL_ CLOSE;
11     OPTIONS DEV=ACTIVEX;
12     GOPTIONS XPIXELS=0 YPIXELS=0;
13     FILENAME EGSR TEMP;
14     ODS tagsets.sasreport13(ID=EGSR) FILE=EGSR
15         STYLE=HtmlBlue
16         STYLESHEET=(URL="file:///C:/Program%20Files/SASHome/SASEnterpriseGuide/7.1/Sty
17 NOGTITLE
18 NOGFOOTNOTE
19 GPATH=&sasworklocation
20 ENCODING=UTF8
21 options(rlap="on")
22 ;
NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR
23
24     GOPTIONS ACCESSIBLE;
25     %macro showmacvars();
26     %put _GLOBAL_;
27     %mend;
28     %showmacvars();
GLOBAL DATASOURCE '%Shared Data\SASHelp\CLASS\Table%';
GLOBAL DATASOURCE_TYPE '1';
GLOBAL SASWORKLOCATION "C:\Users\...\AppData\Local\Temp\SEG1608\SAS Temporary Files\_TD582
GLOBAT _CLIENTNAME 'SAS Enterprise Guide'
```

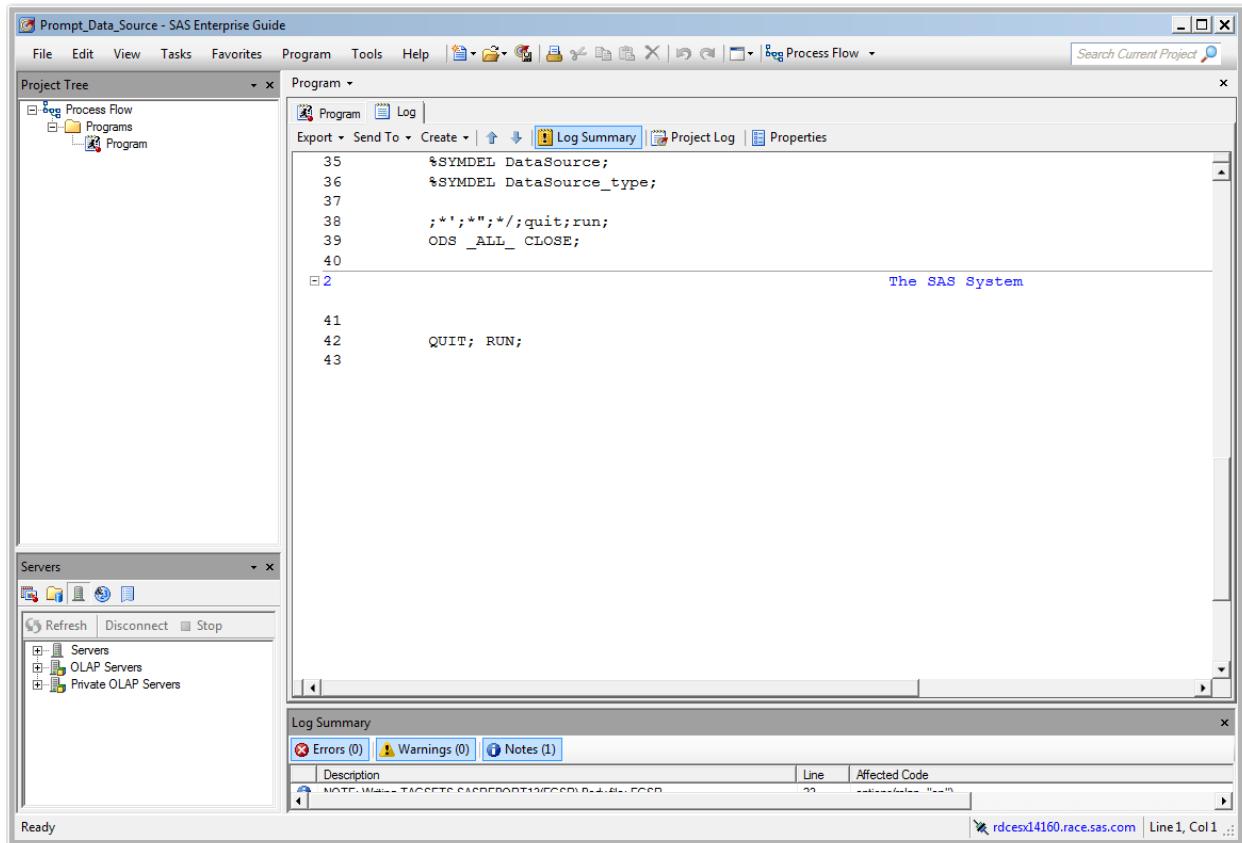
Below the code, the "Log Summary" pane shows one note:

Description	Line	Affected Code
NOTE: Main - TAGSETS.SASREPORT13(EGSR) BODY.FOOD	22	"_TD582

The bottom status bar indicates "rdcesx14160.race.sas.com | Line 1, Col 1".

Display 225 - %LET Statements for Data Source Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.



The screenshot shows the SAS Enterprise Guide interface. The top menu bar includes File, Edit, View, Tasks, Favorites, Program, Tools, Help, and a Process Flow section. The Project Tree on the left shows a 'Program' node under 'Programs'. The main workspace contains a 'Program' tab with the following code:

```
35      %SYMDEL DataSource;
36      %SYMDEL DataSource_type;
37
38      ;*';*/;quit;run;
39      ODS _ALL_ CLOSE;
40
41
42      QUIT; RUN;
43
```

The code is numbered from 35 to 43. A status message 'The SAS System' is displayed above line 41. The bottom right corner of the code area shows 'Line 1, Col 1'. Below the code area is a 'Log Summary' panel with tabs for Errors (0), Warnings (0), and Notes (1). The Notes tab is selected, showing one note: 'NOTE: Macro %SYMDEL removed macro variable "DataSource"'. The 'Servers' panel on the left shows a list of servers: Servers, OLAP Servers, and Private OLAP Servers. The status bar at the bottom indicates 'Ready'.

Display 226 - %SYMDEL Statements Remove DataSource* Macro Variables

SAS Studio

The following display shows the code that is added to the converted Program node for data source prompt in SAS Enterprise Guide.

These global variables are created:

- DATASOURCE
- DATASOURCE_type

The %LET statements assign the default value to the DATASOURCE* macro variables.

If you want to run your process flow using different values for the DATASOURCE prompt, you must manually update values of the macro variables in the %LET statements.

```
/*
SAS Name: DATASOURCE
Prompt: Pick a data source
Description:
This is a data source prompt.
Type: Data Source
*/
%global DATASOURCE;
%let DATASOURCE=URL:/Shared_Data/SASHHELP/CLASS(Table);
%global DATASOURCE_type;
%let DATASOURCE_type=1;

%mend;
/*
SAS Studio generated code to cleanup macro variables created by prompts.
"Use prompt throughout" was not checked for these prompts.
*/
%macro cleanupPromptValues();
%SYNDEL DATASOURCE;
%SYNDEL DATASOURCE_type;
%mend;
%setPromptValues();
/*----- End SAS generated prompt variable code.-----*/

```

Display 227 - Macro Variable Code for Data Source Prompt

Note: Currently, the _type macro variable is always set to 1. This value might not be correct for all cases.

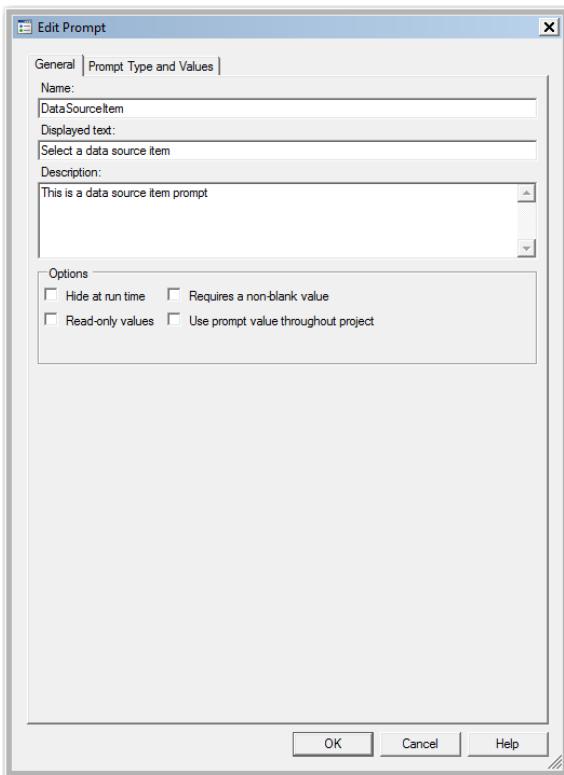
Substituting a SAS Studio Task for a Data Source Prompt

SAS Studio does not have controls with all of the capabilities of a data source prompt. You can create a task with a text input field to prompt for values, or you can use the [SAS Studio Task for Data Source Prompt](#) to mold the capability of the Datasource control to suit your input needs for a SAS library data source.

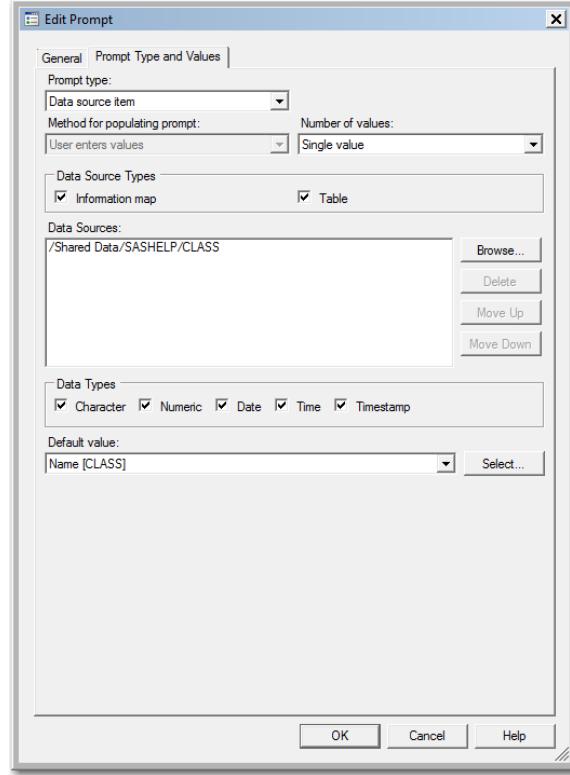
Data Source Item

SAS Enterprise Guide

In this example, a data source item prompt named DataSourceItem is defined as shown in the following two displays.

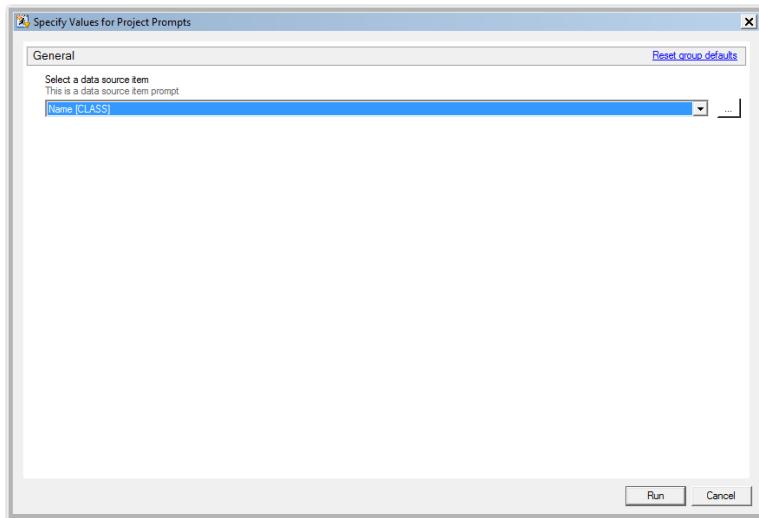


Display 228 - General Properties for Data Source Item Prompt



Display 229 - Type and Values for Data Source Item Prompt

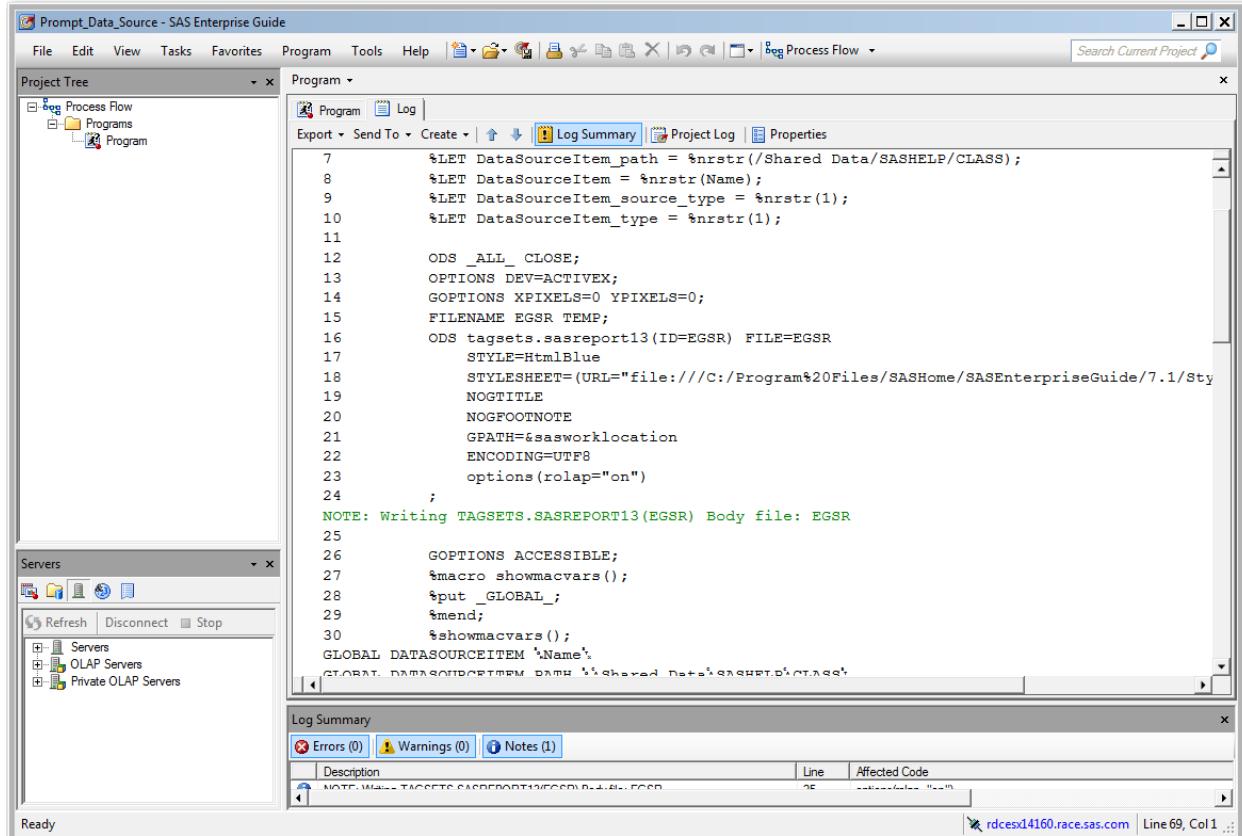
When the code for the prompt is run, the following dialog box appears:



Display 230 - Data Source Item Prompt in Prompt Dialog Box

If the user leaves the default value in the data source item prompt field, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The %LET statements assign the values specified in the prompt dialog box to the DataSourceItem* macro variables.



The screenshot shows the SAS Enterprise Guide interface with a project named "Prompt_Data_Source". The Project Tree shows a "Programs" folder containing a single "Program" item. The main workspace displays the following SAS code:

```
7      %LET DataSourceItem_path = %nrstr(/Shared Data/SASHHELP/CLASS);
8      %LET DataSourceItem = %nrstr(Name);
9      %LET DataSourceItem_source_type = %nrstr(1);
10     %LET DataSourceItem_type = %nrstr(1);
11
12     ODS _ALL_ CLOSE;
13     OPTIONS DEV=ACTIVE;
14     GOPTIONS XPIXELS=0 YPIXELS=0;
15     FILENAME EGSR TEMP;
16     ODS tagsets.sasreport13(ID=EGSR) FILE=EGSR
17       STYLE=HtmlBlue
18       STYLESHEET=(URL="file:///C:/Program%20Files/SASHome/SASEnterpriseGuide/7.1/Sty
19       NOGTITLE
20       NOFOOTNOTE
21       GPATH=%sasworklocation
22       ENCODING=UTF8
23       options(rolap="on")
24 ;
25 NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR
26
27     GOPTIONS ACCESSIBLE;
28     %macro showmacvars();
29     %put _GLOBAL_;
30     %mend;
31     %showmacvars();
32
33     GLOBAL DATASOURCEITEM `Name';
34
35     GLOBAL DATASOURCEITEM_PATH '%Shared Data\SASHHELP\CLASS';

```

The Log Summary panel at the bottom shows one note:

Description	Line	Affected Code
NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR	69	options(rolap="on")

The status bar at the bottom right indicates the log entry is at line 69, column 1.

Display 231 - %LET Statements for Data Source Item Prompt

Because the **Use prompt value throughout project** is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.

The screenshot shows the SAS Enterprise Guide interface with the following details:

- Project Tree:** Shows a node for "Program".
- Servers:** Shows a list of servers including "Servers", "OLAP Servers", and "Private OLAP Servers".
- Log Summary:** The main pane displays a log of SAS code. Lines 30-31 show environment setup with %showmacvars() and various GLOBAL statements. Lines 32-37 show options like NOACCESSIBLE and %LET statements for client variables.
- Log Summary Details:** A modal window at the bottom right shows one note:

Description	Line	Affected Code
NOTE: MACROS TACSETCS CACERREPORT12/ECOD1-D-LBL-FOOD	36	----- ----- -----

Display 232 - Values of Global Macro Variables for Data Source Item

SAS Studio

The following display shows code that is added to the converted Program node for the data source item prompt in SAS Enterprise Guide.

These global variables are created:

- DATASOURCEITEM
 - DATASOURCEITEM_type
 - DATASOURCEITEM_path
 - DATASOURCEITEM_source_type

The %LET statements assign the default value to the DATASOURCEITEM* macro variables.

If you want to run your process flow using different values for the DATASOURCEITEM prompt, you must manually update the values of the macro variables in the %LET statements.

The screenshot shows the SAS Studio interface with the following details:

- Top Bar:** Includes tabs for "SAS Studio - Single User" and "Visual Programmer".
- Left Sidebar:** "Files and Folders" tree view showing categories like "Folder Shortcuts", "EGPs", and specific EGP files such as "Prompt_All_Types_MinMaxVal_Static.egg", "Prompt_Data_Source.egp", and "Prompt_Data_Source_Item.egg".
- Central Area:** A code editor titled "Prompt_Data_Source_Item.1" with the "CONVERSION REPORT" tab selected.
- Code Content:** The code defines a prompt for selecting a data source item. It includes global macro variables for the item name, path, type, and source type, and a cleanup macro at the end.

```
/* -----  
SAS Name: DATASOURCEITEM  
  
Prompt: Select a data source item  
  
Description:  
This is a data source item prompt  
  
Type: Data Source Column  
----- */  
  
%global DATASOURCEITEM;  
%LET DATASOURCEITEM= %nrstr(%name);  
%global DATASOURCEITEM_path;  
%LET DATASOURCEITEM_path= %nrstr(URL::/Shared Data/SASHHELP/CLASS);  
%global DATASOURCEITEM_type;  
%LET DATASOURCEITEM_type = %nrstr(1);  
%global DATASOURCEITEM_source_type;  
%LET DATASOURCEITEM_source_type = %nrstr(1);  
  
-----  
%mend;  
  
/*-----  
SAS Studio generated code to cleanup macro variables created by prompts.  
"Use prompt throughout" was not checked for these prompts.  
----- */  
  
%macro cleanupPromptValues();  
%SYMDEL DATASOURCEITEM;  
%SYMDEL DATASOURCEITEM_path;  
%SYMDEL DATASOURCEITEM_type;  
%SYMDEL DATASOURCEITEM_source_type;  
%mend;
```

Display 233 - Macro Variable Code for Data Source Item Prompt

Note: Currently, the `_type` macro variables are always set to 1. This value might not be correct for all cases.

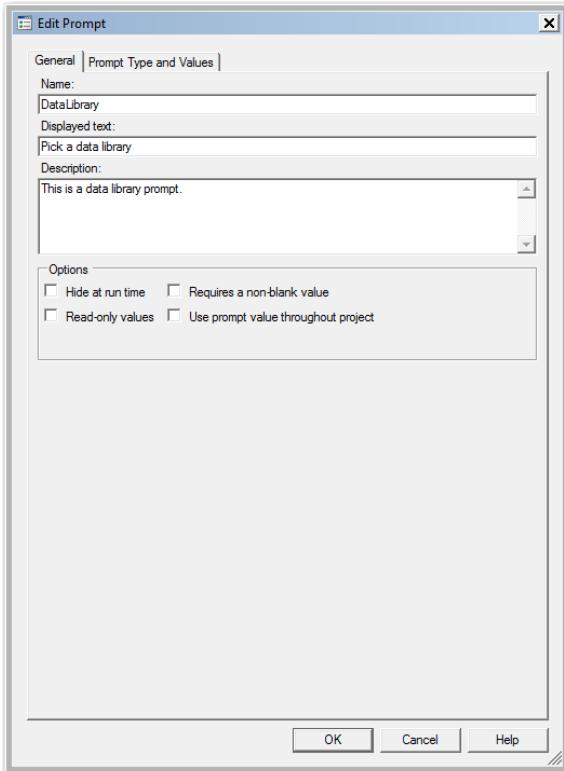
Substituting a SAS Studio Task for a Data Source Item Prompt

SAS Studio does not have controls with all the capabilities of a Data Source Item Prompt. You can create a task that has a text input field to prompt for values, or you can use the [SAS Studio Task Data Source Prompt](#) to mold the capability of the `Datasource` control to input a SAS library.

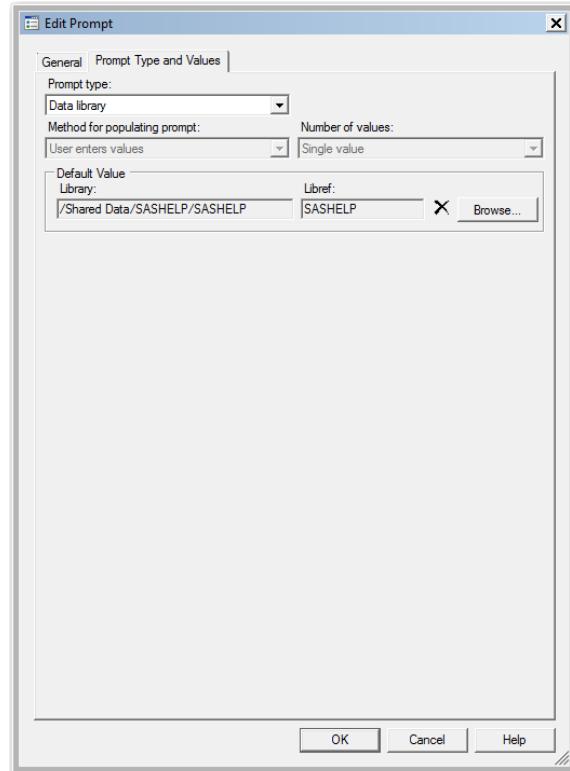
Data Library

SAS Enterprise Guide

In this example, a data library prompt named DataLibrary is defined as shown in the following two displays.

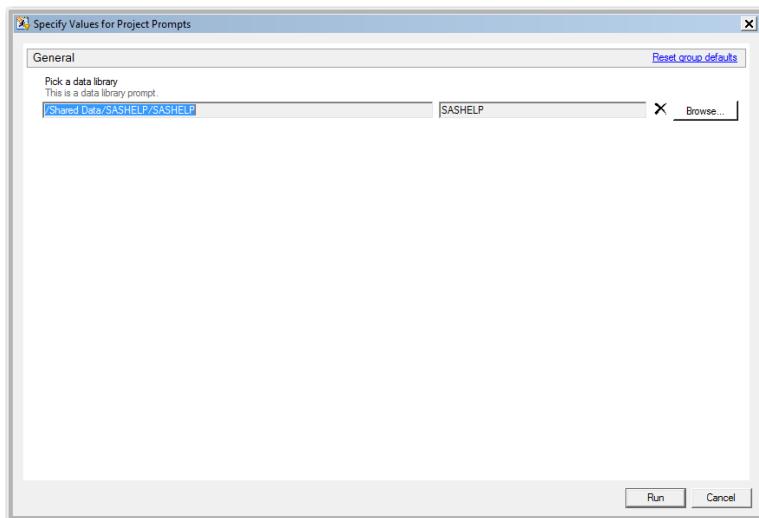


Display 234 - General Properties for Data Library Prompt



Display 235 - Type and Values for Data Library Prompt

When you run the Program node that depends on the prompt, the following dialog box appears:



Display 236 - Data Library Prompt in Prompt Dialog Box

If the user leaves the default value in the data library prompt field, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The %LET statements assign the values specified in the prompt dialog box to the DataLibrary* macro variables.

Because the **Use prompt value throughout project** is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.

The screenshot shows the SAS Enterprise Guide interface with the following details:

- Project Tree:** Shows a node for "Process Flow".
- Servers:** Shows a list of servers including "Servers", "OLAP Servers", and "Private OLAP Servers".
- Program Editor:** Displays a SAS program with code lines numbered 28 through 38. The code includes various global macro definitions such as `GLOBAL DATALIBRARY 'SASHELP'` and `GLOBAL _CLIENTAPP 'SAS Enterprise Guide'`. It also contains options like `GOPTIONS NOACCESSIBLE;` and data step statements like `DATA Library; SET File; RUN;`.
- Log Summary:** A modal window titled "Log Summary" is open, showing the following details:
 - Errors (0)**
 - Warnings (0)**
 - Notes (1)**
 - A table with columns "Description", "Line", and "Affected Code". The single note listed is: "NOTE: Macro TAGSETS is not supported in EGP. Line 33".
- Status Bar:** The status bar at the bottom right shows the URL "rdcesx14160.race.sas.com" and the message "Line 63, Col 1".

Display 237 - Global Variables and %SYMDEL Statements for Data Library Prompt

SAS Studio

The following display shows code that is added to the converted Program node for the data library prompt in SAS Enterprise Guide.

These global variables are created:

- DATALIBRARY
- DATALIBRARY_PATH

The %LET statements assign the default value to the DATALIBRARY* macro variables. If you want to run your process flow using different values for the DATALIBRARY prompt, you must manually update the values of the macro variables in the %LET statements.

The screenshot shows the SAS Studio interface with the 'Visual Programmer' tab selected. On the left, the 'Files and Folders' tree view shows a folder structure under 'EGPs' containing various EGP files, with one file named 'Prompt_Data_Library.1' highlighted. The main workspace displays the following macro code:

```
%macro setPromptValues();  
/* -----  
SAS Name: DATALIBRARY  
Prompt: Pick a data library  
Description:  
This is a data library prompt.  
Type: SAS library  
----- */  
  
%global DATALIBRARY;  
%let DATALIBRARY=%nrstr(SASHelp);  
%global DATALIBRARY_PATH;  
%let DATALIBRARY_PATH=%nrstr(URL:/Shared Data/SASHelp/SASHelp);  
  
%mend;  
/*-----  
SAS Studio generated code to cleanup macro variables created by prompts.  
"Use prompt throughout" was not checked for these prompts.  
----- */  
  
%macro cleanupPromptValues();  
%SYNDEL DATALIBRARY;  
%SYNDEL DATALIBRARY_PATH;  
%mend;  
  
%setPromptValues();
```

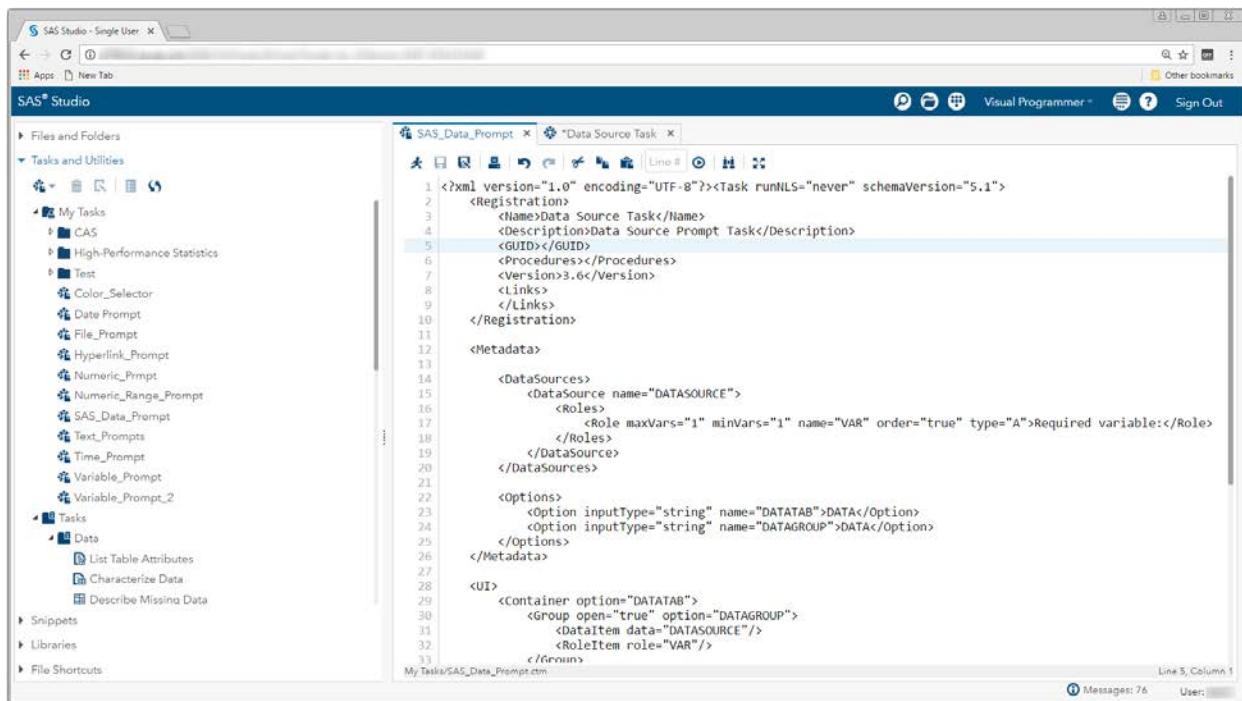
Display 238 - Macro Code for Data Library Prompt

Substituting a SAS Studio Task for a Data Library Prompt

SAS Studio does not have controls with all the capabilities of a data library prompt. You can create a task that has a text input field to prompt for values, or you can use the [SAS Studio Task for Data Source Prompt](#) to mold the capability of the Datasource control to input a SAS library.

SAS Studio Task for Data Source Prompt

SAS Studio does not have task controls that can be configured to provide all the capabilities in the data source, data source item, and data library prompts in SAS Enterprise Guide. However, SAS Studio does have controls that can be used to prompt for tables and variables. A task that prompts for a library and a table column is shown in the following displays.



The screenshot shows the SAS Studio interface with the Visual Programmer editor open. The left sidebar shows a tree view of tasks and utilities, including 'My Tasks' which contains several prompt-related tasks like 'SAS_Data_Prompt'. The main editor window displays the XML code for the 'SAS_Data_Prompt' task. The code defines a task with a name of 'Data Source Task', a description of 'Data Source Prompt Task', and a GUID. It includes sections for registration, metadata (with a data source named 'DATASOURCE'), options (with input types for string), and UI (with a container for a group of data items). The code is well-formatted with line numbers and syntax highlighting.

```
<?xml version="1.0" encoding="UTF-8"?><Task runNLS="never" schemaVersion="5.1">
  <Registration>
    <Name>Data Source Task</Name>
    <Description>Data Source Prompt Task</Description>
    <GUID></GUID>
  </Registration>
  <Metadata>
    <DataSources>
      <DataSource name="DATASOURCE">
        <Roles>
          <Role maxVars="1" minVars="1" name="VAR" order="true" type="A">Required variable:</Role>
        </Roles>
      </DataSource>
    </DataSources>
    <Options>
      <Option inputType="string" name="DATATAB">DATA</Option>
      <Option inputType="string" name="DATAGROUP">DATA</Option>
    </Options>
  </Metadata>
  <UI>
    <Container option="DATATAB">
      <Group open="true" option="DATAGROUP">
        <DataItem data="DATASOURCE"/>
        <RoleItem role="VAR"/>
      </Groups>
    </Container>
  </UI>
</Task>
```

Display 239 – Code for SAS_Data_Prompt Task

Here is the complete code for the SAS_Data_Prompt task.

```
<?xml version="1.0" encoding="UTF-8"?><Task runNLS="never" schemaVersion="5.1">
    <Registration>
        <Name>Data Source Task</Name>
        <Description>Data Source Prompt Task</Description>
        <GUID></GUID>
        <Procedures></Procedures>
        <Version>3.6</Version>
        <Links>
        </Links>
    </Registration>

    <Metadata>

        <DataSources>
            <DataSource name="DATASOURCE">
                <Roles>
                    <Role maxVars="1" minVars="1" name="VAR" order="true"
                          type="A">
                        Required variable:</Role>
                </Roles>
            </DataSource>
        </DataSources>

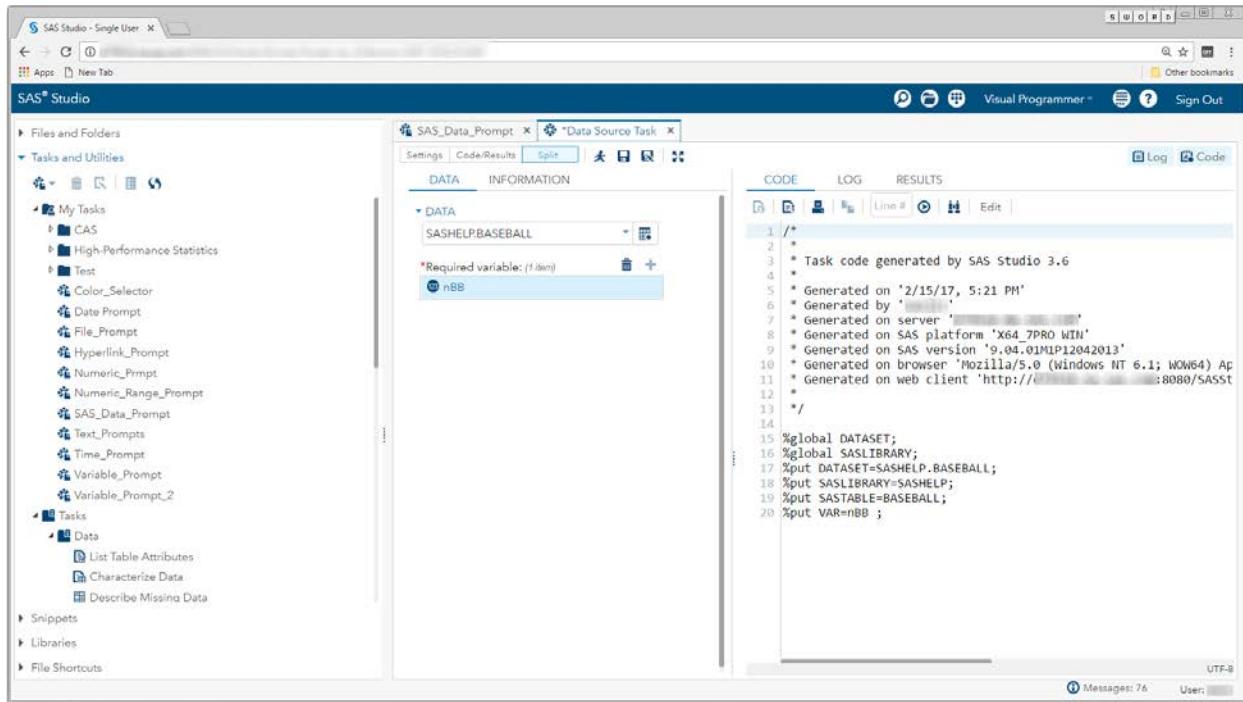
        <Options>
            <Option inputType="string" name="DATATAB">DATA</Option>
            <Option inputType="string" name="DATAGROUP">DATA</Option>
        </Options>
    </Metadata>

    <UI>
        <Container option="DATATAB">
            <Group open="true" option="DATAGROUP">
                <DataItem data="DATASOURCE" />
                <RoleItem role="VAR" />
            </Group>
        </Container>
    </UI>

    <CodeTemplate>
        <![CDATA[

%global DATASET;
%global SASLIBRARY;
%put DATASET=$DATASOURCE;
%put SASLIBRARY=$DATASOURCE.getLibrary();
%put SASTABLE=$DATASOURCE.getTable();
#if( $VAR.size() > 0 )%put VAR=#foreach( $item in $VAR )$item #end;#end

        ]]>
    </CodeTemplate>
</Task>
```

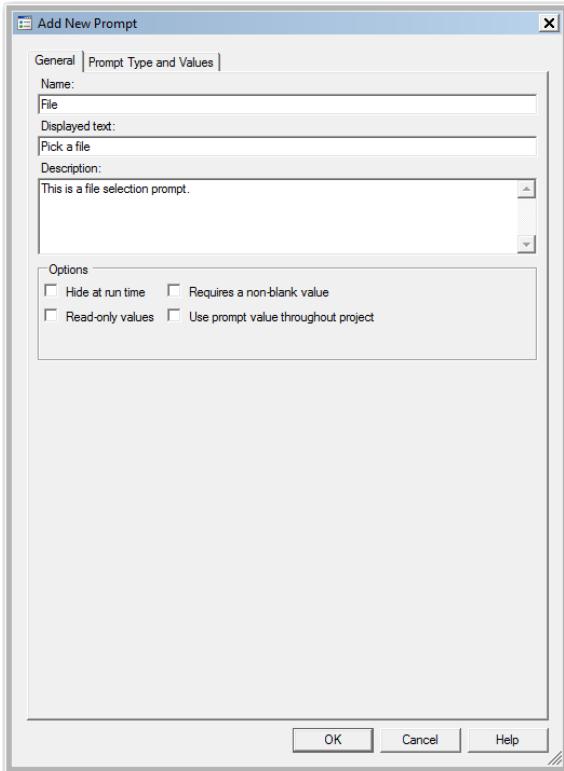


Display 240 - User Interface and Generated SAS Code for Data Source Task

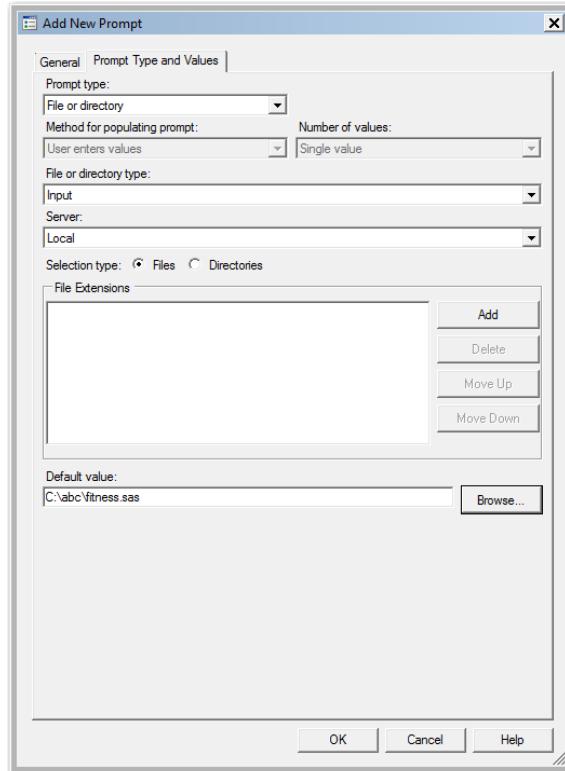
File or Directory

SAS Enterprise Guide

In this example, a file prompt named File is defined as shown in the following two displays.

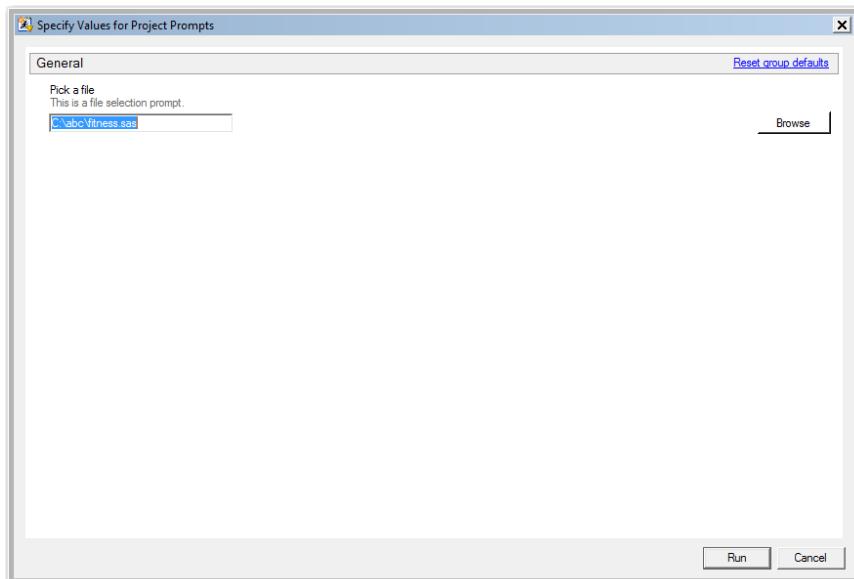


Display 241 - General Properties for File Prompt



Display 242 - Type and Values for File Prompt

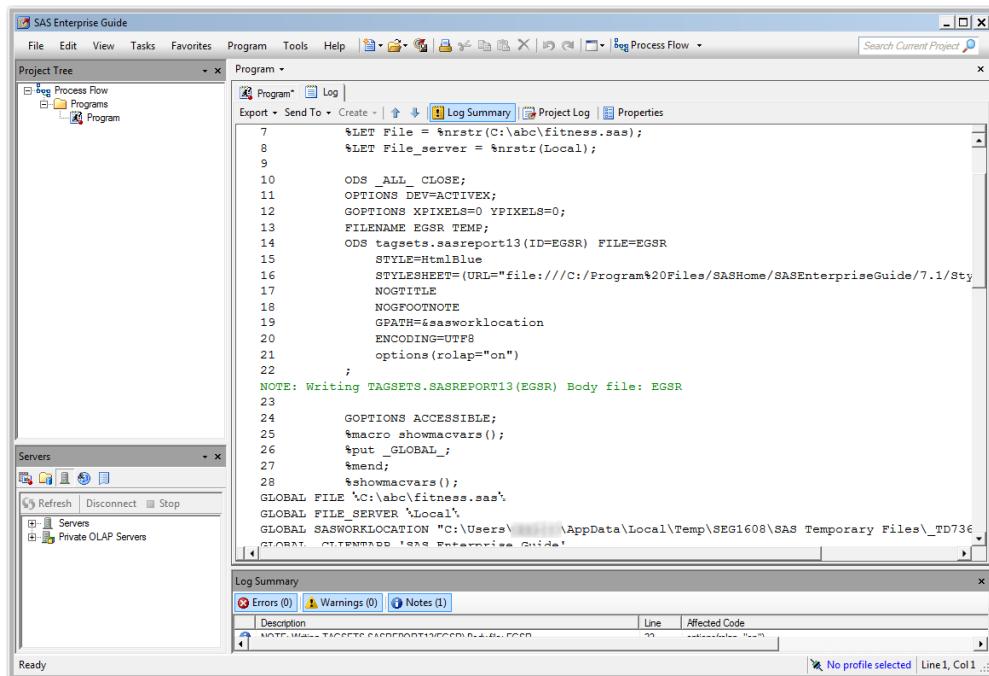
When running the code for this prompt, the following dialog box appears:



Display 243 - File Prompt in Prompt Dialog Box

If the user leaves the default value in the file prompt field, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The %LET statements assign the values specified in the prompt dialog box to the File* macro variables.



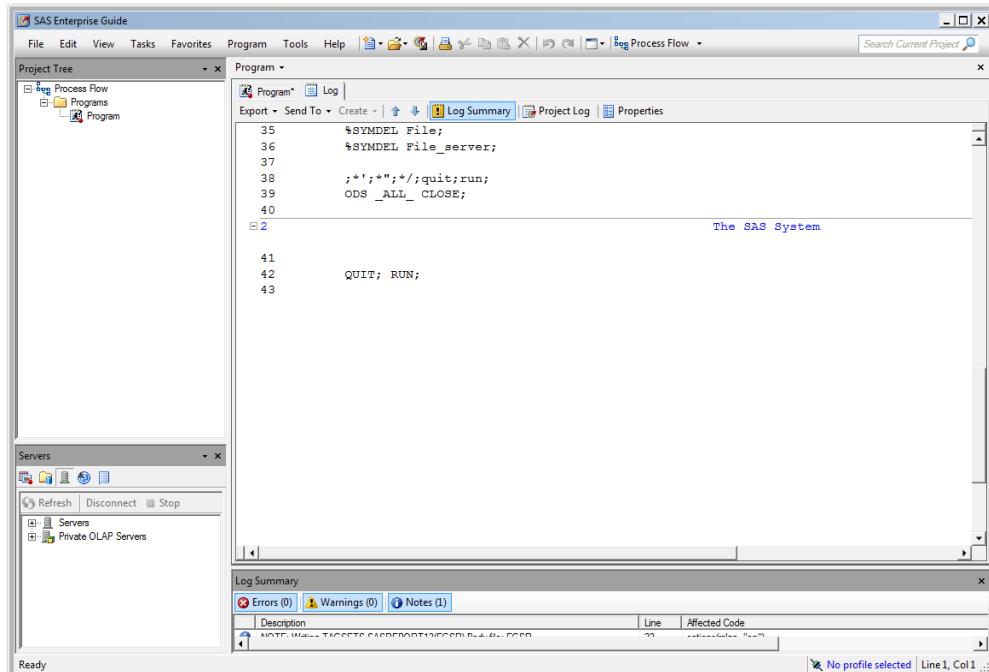
```

SAS Enterprise Guide
File Edit View Tasks Favorites Program Tools Help | Process Flow | Log | Properties
Project Tree
    Process Flow
        Programs
            Program
Program
    Log
    Export Send To Create Log Summary Project Log Properties
    7 %LET File = %nrstr(C:\abc\fitness.sas);
    8 %LET File_server = %nrstr(Local);
    9
    10 ODS _ALL_ CLOSE;
    11 OPTIONS DEV=ACTIVEX;
    12 GOPTIONS XPIXELS=0 YPIXELS=0;
    13 FILENAME EGSR TEMP;
    14 ODS tagsets.sasreport13(ID=EGSR) FILE=EGSR
    15     STYLE=HtmBlue;
    16     STYLESHEET=(URL="file:///C:/Program%20Files/SASHome/SASEnterpriseGuide/7.1/sty
    17     NOGTITLE;
    18     NOFOOTNOTE;
    19     GPATH=%sasworklocation
    20     ENCODING=UTF8
    21     options(rtolap="on")
    22 ;
    NOTE: Writing TAGSETS.SASREPORT13(EGSR) Body file: EGSR
    23
    24 GOPTIONS ACCESSIBLE;
    25 %macro showmacvars();
    26 %put _GLOBAL_;
    27 %mend;
    28 %showmacvars();
    GLOBAL FILE 'C:\abc\fitness.sas';
    GLOBAL FILE_SERVER 'Local';
    GLOBAL SASWORKLOCATION "C:\Users\...\AppData\Local\Temp\SEG1608\SAS Temporary Files\_TD736
    GLOBAL CLIENTAPP 'SAS Enterprise Guide'

```

Display 244 - %LET Statements and Values of Global Variables for File Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.



```

SAS Enterprise Guide
File Edit View Tasks Favorites Program Tools Help | Process Flow | Log | Properties
Project Tree
    Process Flow
        Programs
            Program
Program
    Log
    Export Send To Create Log Summary Project Log Properties
    35 %SYMDEL File;
    36 %SYMDEL File_server;
    37
    38 ;/*;*/quit;run;
    39 ODS _ALL_ CLOSE;
    40
    41 QUIT; RUN;
    42
    43

```

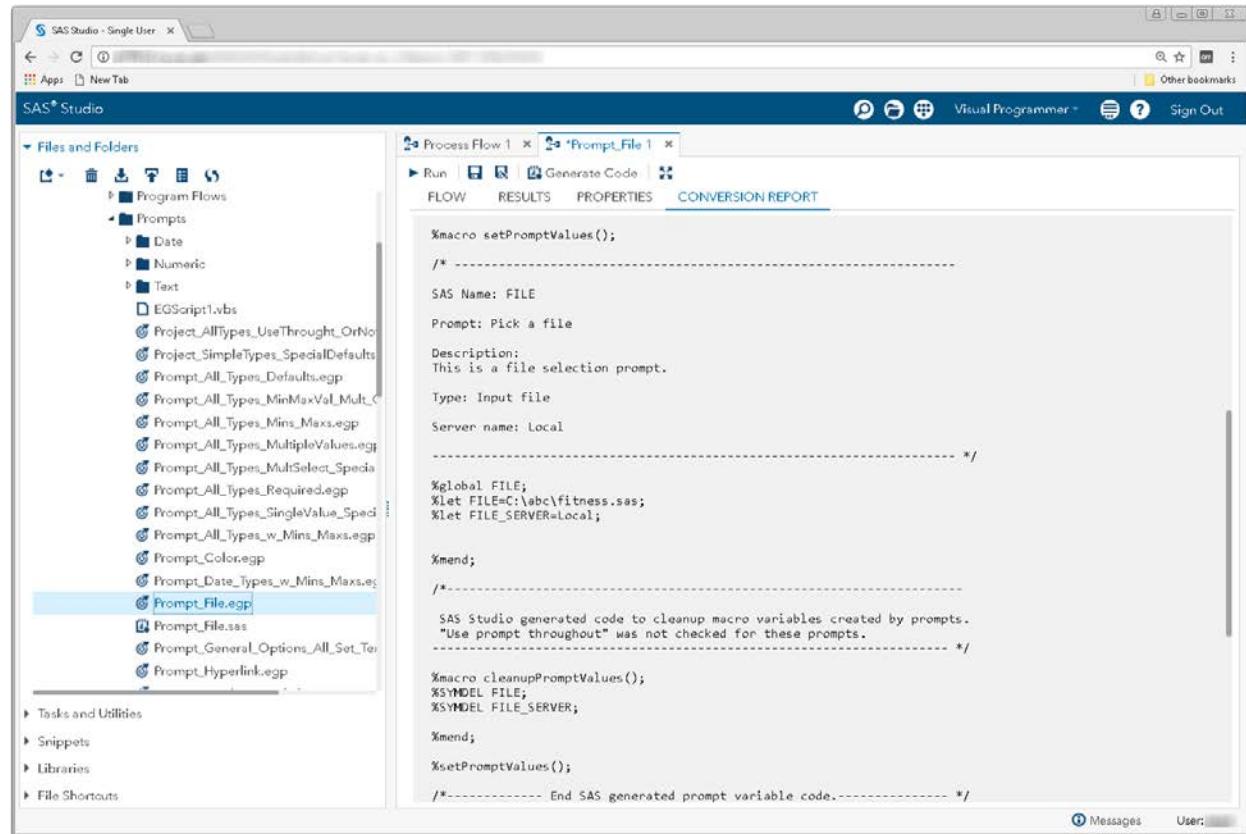
Display 245 - %SYMDEL Statements Remove File* Macro Variables

SAS Studio

The following display shows code that is added to the converted Program node for the file prompt in SAS Enterprise Guide. These global variables are created:

- FILE
- FILE_SERVER

The %LET statements assign the default value to the FILE* macro variables. If you want to run your process flow using different values for the FILE prompt, you must manually update the values of the macro variables in the %LET statements.



The screenshot shows the SAS Studio interface. On the left, the 'Files and Folders' tree view is open, showing a folder structure under 'Prompts' containing various EGScript files. A file named 'Prompt_File.egp' is selected. In the center, the 'Process Flow 1' window is active, showing a 'Prompt_File 1' node. The 'CONVERSION REPORT' tab is selected in the top navigation bar. The code editor displays the following SAS macro code:

```
%macro setPromptValues();
/*
-----*/
SAS Name: FILE
Prompt: Pick a file
Description: This is a file selection prompt.
Type: Input file
Server name: Local
-----
%global FILE;
%let FILE=C:\abc\fitness.sas;
%let FILE_SERVER=Local;

%mend;
/*
-----
SAS Studio generated code to cleanup macro variables created by prompts.
"Use prompt throughout" was not checked for these prompts.
-----
%macro cleanupPromptValues();
%$MDEL FILE;
%$MDEL FILE_SERVER;
%mend;
%setPromptValues();
/*
----- End SAS generated prompt variable code.----- */

```

Display 246 - Macro Variable Code for the File Prompt

Substituting a SAS Studio Task for a File Prompt

1. Create a SAS Studio task with a control that represents the prompt for a file.
 - Add a label and a sasserverpath input control. Set the name of the input control to FilePrompt.
 - Set the default value to the default value shown in the generated `setPromptValues()` macro in the converted Program node.
 - Change the string of the input control to match the string specified in the prompt.

SAS Studio - Single User

Apps New Tab

SAS® Studio

Files and Folders

Tasks and Utilities

- My Tasks
 - CAS
 - High-Performance Statistics
 - Advanced Task
 - CAS Connection
 - Color_Selector
 - File_Prompt
 - Numeric_Prompt
 - Numeric_Range_Prompt
 - Sample Task
 - TaskWithSessions
 - Text_Prompts
 - Tasks
 - Data
 - Graph
 - Combinatorics and Probability
 - Statistics
 - High-Performance Statistics
 - Power and Sample Size
 - Multivariate Analysis
 - Cluster Analysis
 - Snippets
 - Libraries
 - File Shortcuts

File_Prompt

```
<?xml version="1.0" encoding="UTF-8"?>
<Task schemaVersion="5.1" runNLS="never">
    <Registration>
        <Name>File Prompt</Name>
        <Description>File Prompt</Description>
        <GUID></GUID>
        <Procedures></Procedures>
        <version>3.6</Version>
        <Links>
            </Links>
        </Registration>
    <Metadata>
        <DataSources>
            </DataSources>
        <options>
            <option name="labelTEXT" inputType="string">Pick a file</option>
            <option name="filePrompt" defaultValue="c:\abc\fitness.sas"
                   inputType="sasserverpath"
                   promptMessage="Select a file">This is a file selection prompt</option>
        </options>
    </Metadata>
    <UI>
        <optionItem option="labelTEXT"/>
        <optionItem option="FilePrompt"/>
    </UI>
    <CodeTemplate>
        <![CDATA[

]]>
    </CodeTemplate>
</Task>
```

Line 39, Column 25

MyTasks\File_Prompt.cs

Messages: 16 User:

Display 247 - Code for File_Prompt Task

The following code is an example of a task that could be used as the file prompt.

```
<?xml version="1.0" encoding="UTF-8"?>
<Task schemaVersion="5.1" runNLS="never">
    <Registration>
        <Name>File Prompt</Name>
        <Description>File Prompt</Description>
        <GUID></GUID>
        <Procedures></Procedures>
        <Version>3.6</Version>
        <Links>
        </Links>
    </Registration>

    <Metadata>

        <DataSources>
        </DataSources>

        <Options>
            <Option name="labelTEXT" inputType="string">Pick a file</Option>
            <Option name="FilePrompt" defaultValue="c:\abc\fitness.sas"
                   inputType="sasserverpath"
                   promptMessage="Select a file">
                This is a file selection prompt
            </Option>
        </Options>
    </Metadata>

    <UI>
        <OptionItem option="labelTEXT"/>
        <OptionItem option="FilePrompt"/>
    </UI>

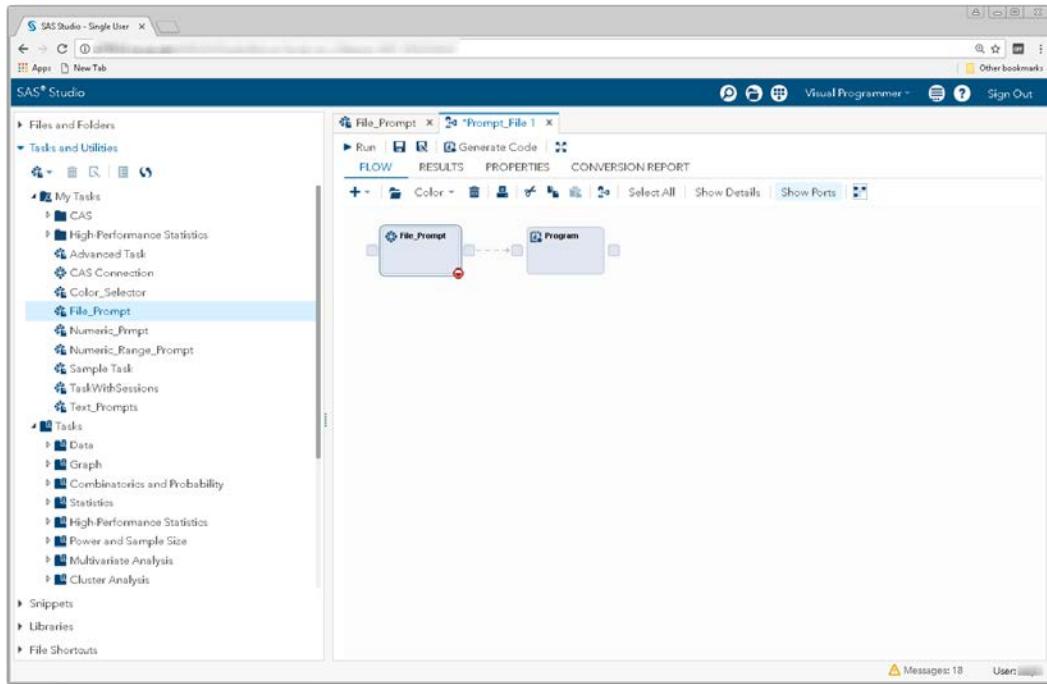
    <CodeTemplate>
        <![CDATA[

%global File_Server;
%let File_Server="Local";

%global File;
%let File=${FilePrompt.fullPath};

        ]]>
    </CodeTemplate>
</Task>
```

2. Save this task to your **My Tasks** folder.
3. Drag the task from **My Tasks** into your converted process flow.
4. Link the output port of the new task to the input port of the converted Program node.



Display 248 - File_Prompt Task Linked to Program Node

5. Comment out the `%setPromptValues()` macro call from the converted Program node. The macro code generated by the File_Prompt task replaces this code.

```

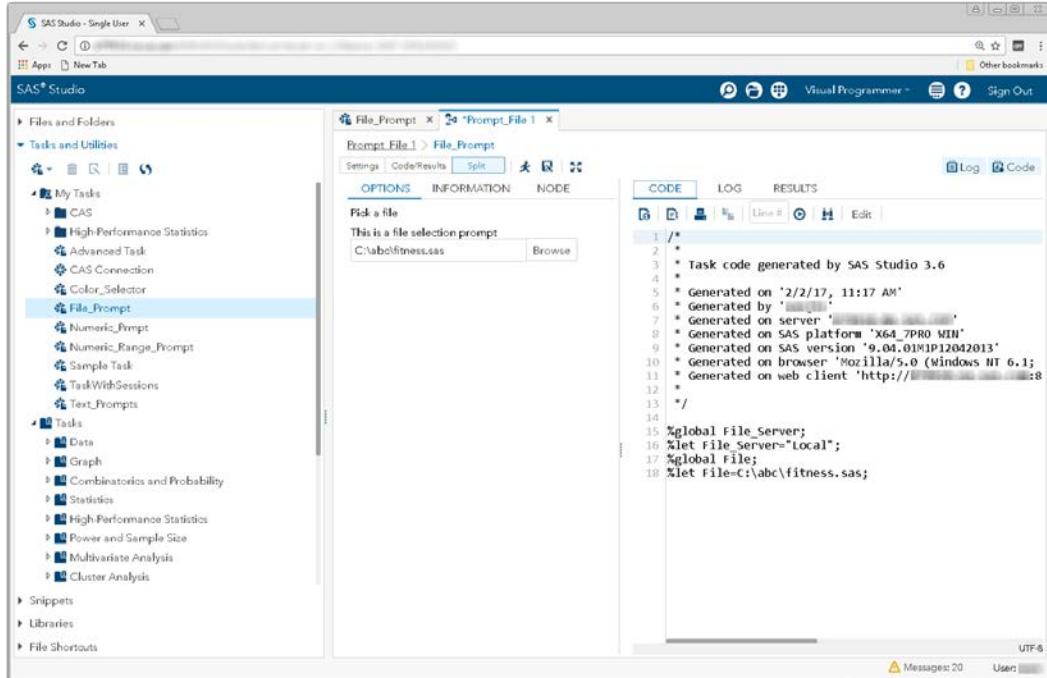
FILE=C:\abc\fitness.sas;
FILE_SERVER=Local;
xend;
/*-----*/
SAS studio generated code to cleanup macro variables created by prompts.
"use prompt throughout" was not checked for these prompts.
----- */
macro cleanupPromptValues();
xymodel FILE;
xymodel FILE_SERVER;
xend;
/*----- End SAS generated prompt variable code.----- */
macro showmacvars();
put _GLOBAL_;
xend;
xshowmacvars();
/*----- SAS generated prompt variable cleanup code.----- */
cleanupPromptValues();
/*----- End SAS generated prompt variable cleanup code.----- */

```

The screenshot shows the SAS Studio interface with the 'Visual Programmer' editor open. The 'CODE' tab is selected, showing SAS code. Line 48 contains the macro call `%setPromptValues();`, which is highlighted and preceded by a double slash `//`, indicating it is commented out. The rest of the code is standard SAS cleanup code. The top navigation bar shows 'SAS Studio - Single User' and 'Visual Programmer'.

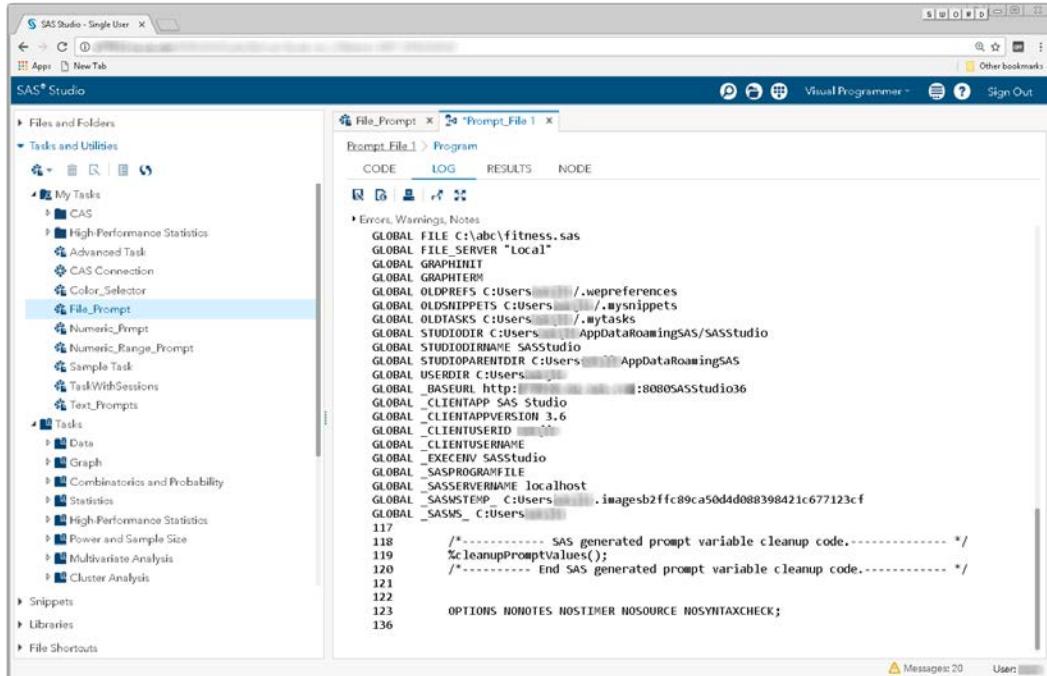
Display 249 - Commented Out %setPromptValues Macro Call

To run your flow with a different value than the default value, open the Prompt_File node and select a different file.



Display 250 - Running the File_Prompt Task

When you run the process flow, the global File* variables are set to the values specified in the task.

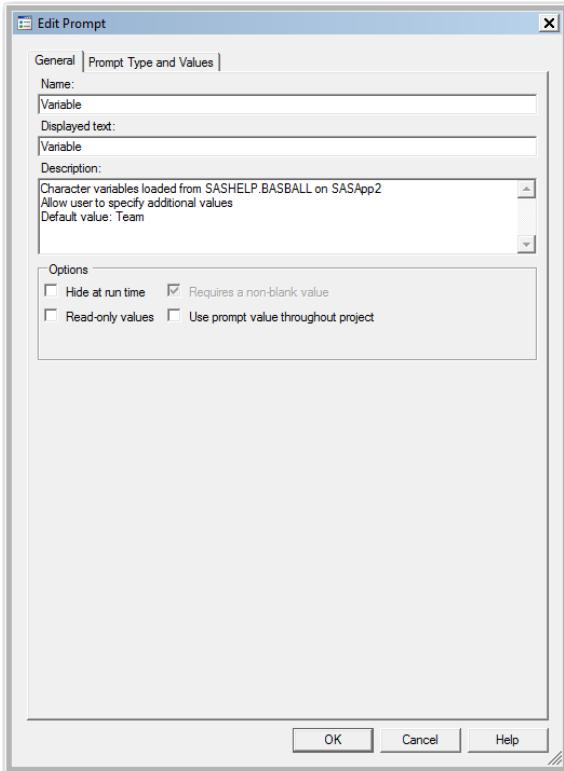


Display 251 - File Prompt Variables with Updated Values

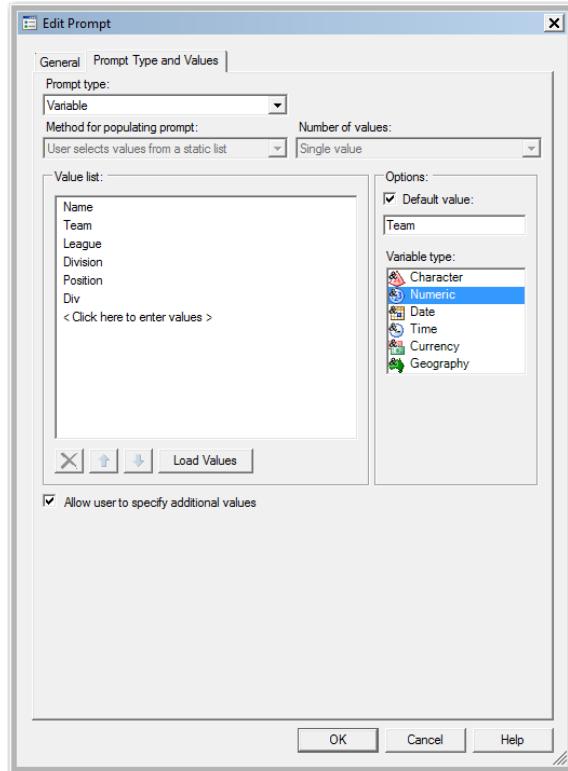
Variable

SAS Enterprise Guide

In this example, a variable prompt named Variable is defined as shown in the following two displays.

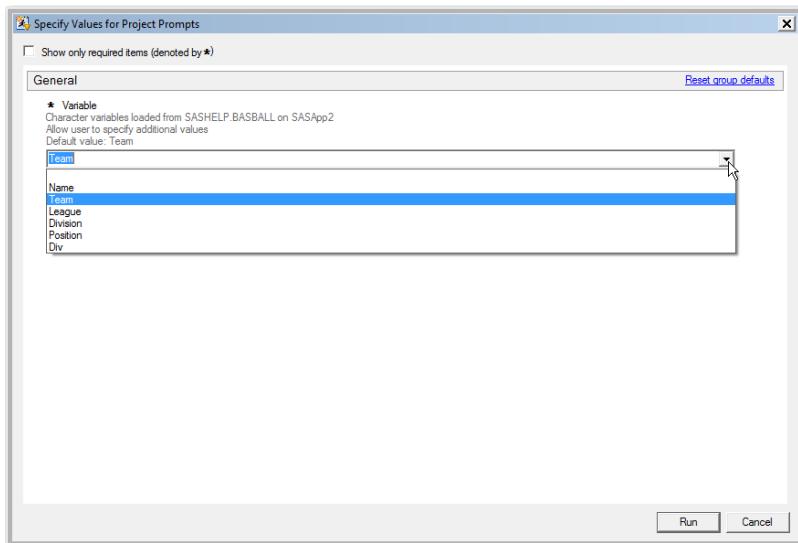


Display 252 - General Properties for Variable Prompt



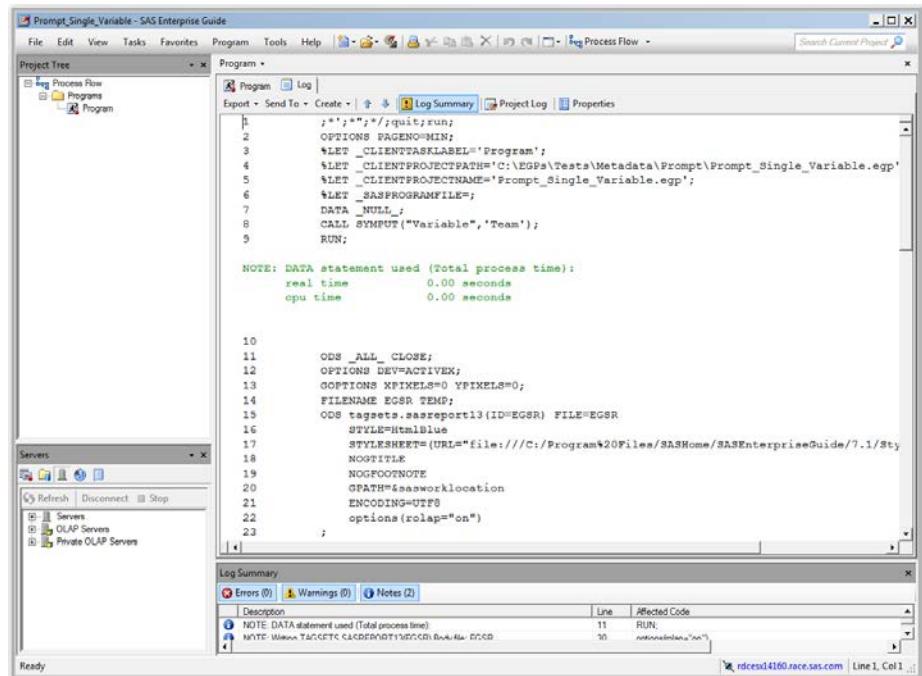
Display 253 - Type and Values for Variable Prompt

When you run the Program node that depends on the prompt, the following display appears:



Display 254 – Variable Prompt in the Prompt Dialog Box

If the user leaves the default value in the variable prompt field, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt. The %LET statements assign the value specified in the prompt dialog box to the Variable macro variable.



```

1 ;/*";*/quit;run;
2 OPTIONS PAGENO=MIN;
3 %LET _CLIENTTASKLABEL='Program';
4 %LET _CLIENTPROJECTPATH='C:\EGPa\Tests\Metadata\Prompt\Prompt_Single_Variable.egp';
5 %LET _CLIENTPROJECTNAME='Prompt_Single_Variable';
6 %LET _SASPROGRAMFILE='';
7 DATA _NULL_;
8 CALL SYMPUT("Variable","Team");
9 RUN;

NOTE: DATA statement used (Total process time):
      real time            0.00 seconds
      cpu time             0.00 seconds

10
11      ODS _ALL_ CLOSE;
12      OPTIONS DEV=ACTIVEV;
13      GOPTIONS XPIXELS=0 YPIXELS=0;
14      FILENAME EGSR TEMP;
15      ODS tagsets.sasreport13 ID=EGSR FILE=EGSR
16          STYLE=htmlBlue
17          STYLESHEET=(URL="file:///C:/Program%20Files/SASHome/SASEnterpriseGuide/7.1/Sty
18          NOGTTITLE
19          NOGFOOTNOTE
20          GPATH=%sasworklocation
21          ENCODING=UTF8
22          options(rolap="on")
23      ;

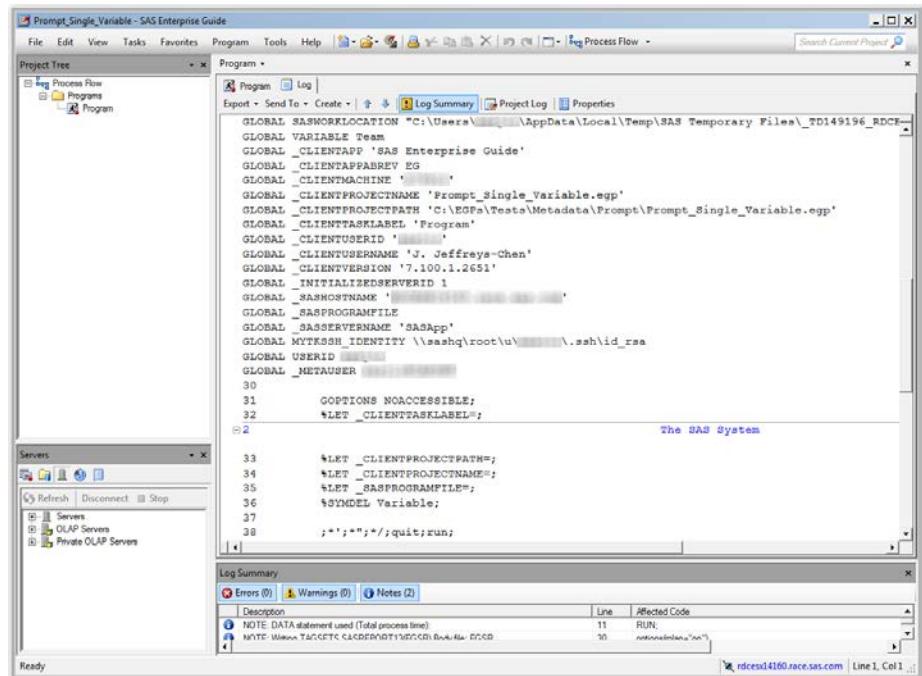
```

Log Summary

Description	Line	Affected Code
NOTE: DATA statement used (Total process time)	11	RUN;
NOTE: When TARGETC CANCELBYDTY(DCC) available. DCC	16	style=htmlBlue;

Display 255 - %LET Statements for Variable Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.



```

GLOBAL _SASWORKLOCATION "C:\Users\...\AppData\Local\Temp\SAS Temporary Files\_TD149196_RDCE...
GLOBAL _VARIENT Team
GLOBAL _CLIENTAPP 'SAS Enterprise Guide'
GLOBAL _CLIENTAPPREV EG
GLOBAL _CLIENTMACHINE '...'
GLOBAL _CLIENTPROJECTNAME 'Prompt_Single_Variable.egp'
GLOBAL _CLIENTPROJECTPATH 'C:\EGPa\Tests\Metadata\Prompt\Prompt_Single_Variable.egp'
GLOBAL _CLIENTTASKLABEL 'Program'
GLOBAL _CLIENTUSERID '...'
GLOBAL _CLIENTUSERNAME 'J. Jeffreys-Chen'
GLOBAL _CLIENTVERSION '7.100.1.2651'
GLOBAL _INITIALZEDSERVERID 1
GLOBAL _SASHOSTNAME '...'
GLOBAL _SASPROGRAMFILE
GLOBAL _SASSERVERNAME 'SASApp'
GLOBAL MYTICKS$N IDENTITY '\\sashq\root\u...\ssh\id_rsa'
GLOBAL _USERID ...
GLOBAL _METAUSER ...
30
31      OPTIONS NOACCESSIBLE;
32      %LET _CLIENTTASKLABEL=;

The SAS System

33      %LET _CLIENTPROJECTPATH=;
34      %LET _CLIENTPROJECTNAME=;
35      %LET _SASPROGRAMFILE=;
36      %SYMDEL Variable;
37
38 ;/*";*/quit;run;

```

Log Summary

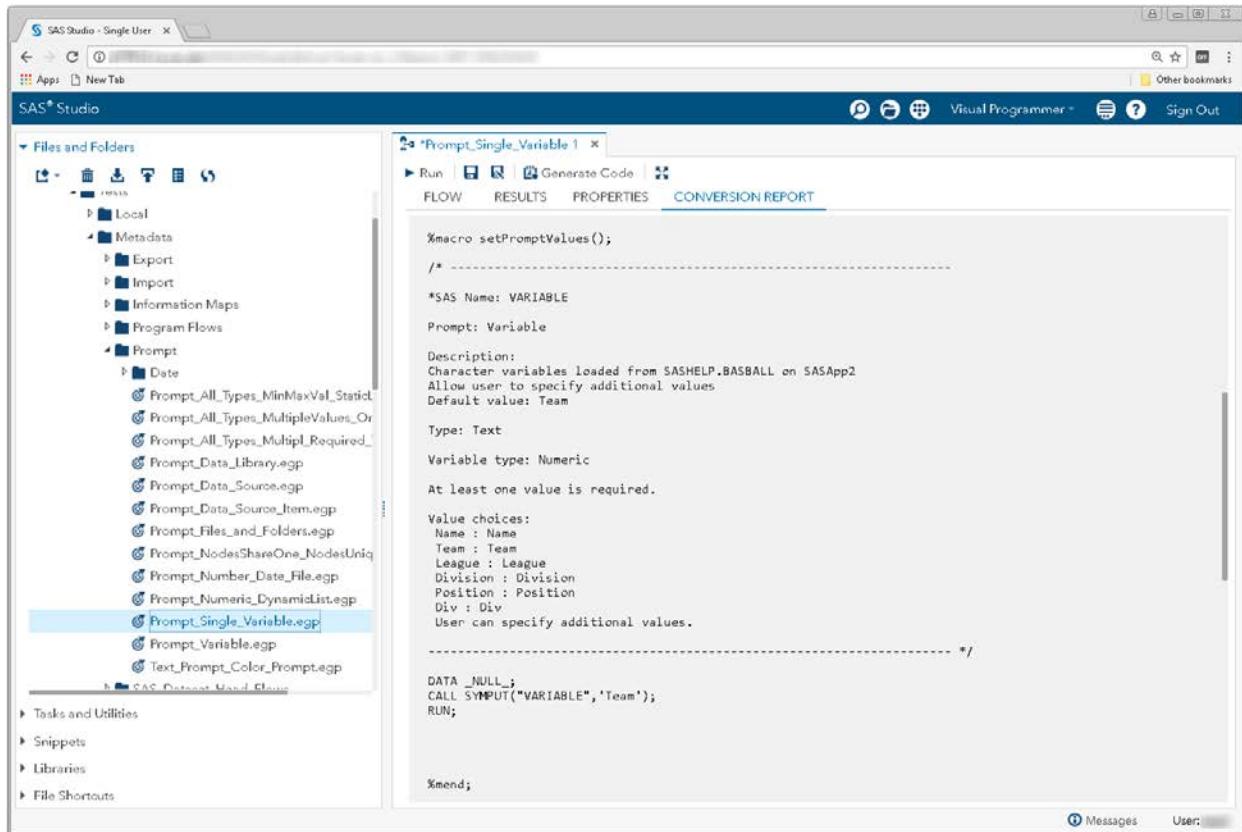
Description	Line	Affected Code
NOTE: DATA statement used (Total process time)	11	RUN;
NOTE: When TARGETC CANCELBYDTY(DCC) available. DCC	16	style=htmlBlue;

Display 256 – Values of Global Variables and %SYMDEL Statements for Variable Prompt

SAS Studio

The following display shows code that is added to the converted Program node for the variable prompt in SAS Enterprise Guide.

The global variable VARIABLE is created, and the %LET statement assigns the default value to the VARIABLE macro variable. If you want to run the process flow using a different value for the VARIABLE prompt, you must manually update the value of the macro variable in the %LET statement.



The screenshot shows the SAS Studio interface. On the left, the 'Files and Folders' tree view displays various SAS files under 'Local' and 'Program Flows'. A file named 'Prompt_Single_Variable.egp' is selected and highlighted with a blue border. The main workspace is titled 'Prompt_Single_Variable.1' and contains the following code:

```
%macro setPromptValues();
/*
*SAS Name: VARIABLE
Prompt: Variable
Description:
Character variables loaded from SASHELP.BASBALL on SASApp2
Allow user to specify additional values.
Default value: Team
Type: Text
Variable type: Numeric
At least one value is required.

Value choices:
Name : Name
Team : Team
League : League
Division : Division
Position : Position
Div : Div
User can specify additional values.

----- */
DATA _NULL_;
CALL SYMPUT("VARIABLE",'Team');
RUN;

%mend;
```

Display 257 - %SYMPUT Code for Variable Prompt

The screenshot shows the SAS Studio interface. On the left is a sidebar with 'Files and Folders' and 'Tasks and Utilities'. The main area has tabs for 'FLOW', 'RESULTS', 'PROPERTIES', and 'CONVERSION REPORT'. The 'CONVERSION REPORT' tab is selected, displaying generated SAS code. The code includes cleanup macros for prompt variables and notes about execution server settings.

```
/*
-----  
SAS Studio generated code to cleanup macro variables created by prompts.  
"Use prompt throughout" was not checked for these prompts.----- */  
  
%macro cleanupPromptValues();  
%SYMDEL VARIABLE;  
  
%mend;  
  
%setPromptValues();  
  
/*----- End SAS generated prompt variable code,----- */  
  
/*----- SAS generated prompt variable cleanup code,----- */  
%cleanupPromptValues();  
/*----- End SAS generated prompt variable cleanup code,----- */  
  
ERROR: Node 'Program' has execution server set to "SASApp" but the current SAS Studio server is "Local".  
  
NOTE: Generating node connectors.  
NOTE: Linking nodes connected by data nodes and eliminating data nodes.  
  
Process Flow Node Summary  
Steps converted:  
Program
```

Display 258 - %SYMDEL Code for Variable Prompt

Substituting a SAS Studio Task for Variable Prompt

When defining a variable prompt in SAS Enterprise Guide, the value choices for the prompt are gathered from a data source. These values are then available in a drop-down list when the prompt is displayed. So to replace this type of prompt in a SAS Studio task, you can simply use the combobox control.

1. Create a SAS Studio task with a control that represents the prompt for a Variable.
 - Add controls as shown in the Variable Prompt task.
 - Set the default value to the default value shown in the generated `setPromptValues()` macro in the converted Program node.
 - Change the strings of the input controls to match the strings specified in the prompt.

The screenshot shows the SAS Studio interface with the following details:

- Top Bar:** Includes the title "SAS Studio - Single User", a search bar, and a "Sign Out" button.
- Left Sidebar:** Contains sections for "Files and Folders", "Tasks and Utilities", "My Tasks" (with items like CAS, High-Performance Statistics, Test, Color_Selector, Date Prompt, File_Prompt, Hyperlink_Prompt, Numeric_Prompt, Numeric_Range_Prompt, SAS_Data_Prompt, Text_Prompts, Time_Prompt, Variable Prompt), "Tasks" (with Data, List Table Attributes, Characterize Data, and Display Missing Data), "Snippets", "Libraries", and "File Shortcuts".
- Code Editor:** The main area displays an XML file named "Prompt_Single_Variable.tsk". The code defines a variable prompt task with registration information, metadata, and options for input types and descriptions. The code editor includes syntax highlighting and a status bar at the bottom indicating "Line 51, Column 35".

```
<?xml version="1.0" encoding="UTF-8"?>
<Task schemaVersion="5.1" runNLS="never">
  <Registration>
    <Name>Variable Prompt</Name>
    <Description>This is a variable prompt task.</Description>
    <GUID></GUID>
  <Procedures></Procedures>
  <Version>3.6</Version>
  <Links></Links>
</Registration>

<Metadata>
  <DataSources>
    </DataSources>
  <Options>
    <Option name="labelDescriptionTextSel" inputType="string">
      * Variable</Option>
    <Option name="labelDescriptionTextSelHint1" inputType="string">
      Character variables loaded from SASHELP BA
    <Option name="labelDescriptionTextSelHint2" inputType="string">
      Allow user to specify additional values</Option>
    <Option name="labelDescriptionTextSelHint3" inputType="string">
      Default value: Team</Option>
    <Option name="variable" inputType="combobox" editable="true" defaultValue="Team"></Option>
    <Option name="Name" inputType="string">Name</Option>
    <Option name="Team" inputType="string">Team</Option>
    <Option name="League" inputType="string">League</Option>
  </Options>
</Metadata>
```

Display 259 - Replacement Task for Variable Prompt

The following code is an example of a task that could be used as the variable prompt.

```
<?xml version="1.0" encoding="UTF-8"?>
<Task schemaVersion="5.1" runNLS="never">
    <Registration>
        <Name>Variable Prompt</Name>
        <Description>This is a variable prompt task.</Description>
        <GUID></GUID>
        <Procedures></Procedures>
        <Version>3.6</Version>
        <Links></Links>
    </Registration>
    <Metadata>
        <DataSources>
        </DataSources>

        <Options>
            <Option name="labelDescriptionTextSel" inputType="string">
                * Variable
            </Option>

            <Option name="labelDescriptionTextSelHint1" inputType="string">
                Character variables loaded from SASHELP BASBALL on SASApp2
            </Option>

            <Option name="labelDescriptionTextSelHint2" inputType="string">
                Allow user to specify additional values
            </Option>

            <Option name="labelDescriptionTextSelHint3" inputType="string">
                Default value: Team
            </Option>

            <Option name="variable" inputType="checkbox" editable="true"
                   defaultValue="Team">
            </Option>

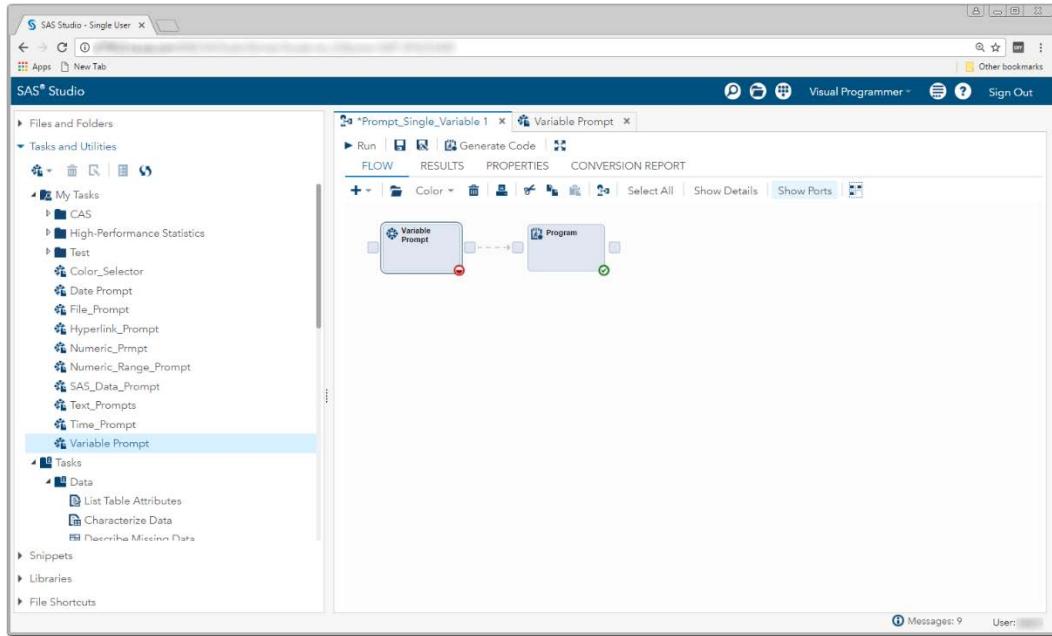
            <Option name="Name" inputType="string">Name</Option>
            <Option name="Team" inputType="string">Team</Option>
            <Option name="League" inputType="string">League</Option>
            <Option name="Division" inputType="string">Division</Option>
            <Option name="Position" inputType="string">Position</Option>
            <Option name="Div" inputType="string">Div</Option>
        </Options>
    </Metadata>

    <UI>
        <OptionItem option="labelDescriptionTextSel"/>
        <OptionItem option="labelDescriptionTextSelHint1"/>
        <OptionItem option="labelDescriptionTextSelHint2"/>
        <OptionItem option="labelDescriptionTextSelHint3"/>
            <OptionChoice option="variable">
                <OptionItem option="Name"/>
                <OptionItem option="Team"/>
                <OptionItem option="League"/>
                <OptionItem option="Division"/>
                <OptionItem option="Position"/>
                <OptionItem option="Div"/>
            </OptionChoice>
    </UI>

```

```
<CodeTemplate>
    <! [ CDATA[
DATA _NULL_;
    CALL SYMPUT("Variable", '${variable}');
run;
    ]]>
</CodeTemplate>
</Task>
```

2. Save the prompt replacement task to your **My Tasks** folder.
3. Drag the task from **My Tasks** into your converted process flow.
4. Link the output port of the Variable Prompt task to the input port of the converted Program node.



Display 260 – Variable Prompt Task Linked to Program Node

5. Comment out the `%setPromptValues()` macro call from the converted Program node. The macro code generated by the Variable Prompt task replaces this code.

```

47
48
49 %mend;
50
51 /*-----
52 SAS Studio generated code to cleanup macro variables created by prompts.
53 "Use prompt throughout" was not checked for these prompts.
54 -----*/
55
56 %macro cleanupPromptValues();
57 %SYMBOL VARIABLE;
58 %MEND;
59
60 %mend;
61
62 /*%setPromptValues();
63
64 /*----- End SAS generated prompt variable code.-----*/
65
66 %macro test();
67 %put _GLOBAL_;
68 %mend;
69 %test();
70
71 /*----- SAS generated prompt variable cleanup code.-----*/
72 %cleanupPromptValues();
73 /*----- End SAS generated prompt variable cleanup code.-----*/
74

```

The screenshot shows the SAS Studio interface with the 'Program' node selected. The code editor displays the generated SAS code. A specific line of code, `%setPromptValues();`, is highlighted with a light blue background, indicating it is being modified. The code also includes comments explaining the purpose of the generated code.

Display 261 – Commented out %setPromptValues in Program Node

To run your flow with a different variable than the default value, open the Variable Prompt node and specify a different value.

The screenshot shows the SAS Studio interface with the 'Variable Prompt' task selected in the left sidebar. The main pane displays the task configuration, specifically the 'Variable' section where the 'Default value' is set to 'League'. The generated code in the 'CODE' tab is as follows:

```

1 /*+
2 +
3 * Task code generated by SAS Studio 3.6
4 *
5 * Generated on '2/16/17, 11:29 AM'
6 * Generated by ''
7 * Generated on server ''
8 * Generated on SAS platform 'X64_7PRO WIN'
9 * Generated on SAS version '9.04.01MIP12042013'
10 * Generated on browser 'Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.36'
11 * Generated on web client 'http://[REDACTED]:8080'
12 */
13
14 DATA _NULL_;
15   CALL SYMPUT("Variable", "League");
16 RUN;

```

Display 262 – User Interface and Generated Code for Variable Prompt Task

When you run the process flow, the global Variable macro variable is set to the value specified in the task.

The screenshot shows the SAS Studio interface with the 'Variable Prompt' task selected in the left sidebar. The main pane displays the task configuration, specifically the 'Program' section which contains several global macro variable definitions. The generated code in the 'CODE' tab is as follows:

```

GLOBAL OLDSNIPPETS C:\Users\[REDACTED]\mysnippets
GLOBAL OLDTASKS C:\Users\[REDACTED]\mytasks
GLOBAL STUDIOODIR C:\Users\[REDACTED]\AppData\Roaming\SAS\Studio
GLOBAL STUDIOODIRNAME SASStudio
GLOBAL STUDIOOPARENTDIR C:\Users\[REDACTED]\AppData\Roaming\SAS
GLOBAL USERDIR C:\Users\[REDACTED]
GLOBAL VARIABLE League
GLOBAL _BASEURL http://[REDACTED]:8080/SASStudio36
GLOBAL _CLIENTAPP SAS Studio
GLOBAL _CLIENTAPPVERSION 3.6
GLOBAL _CLIENTUSERID [REDACTED]
GLOBAL _CLIENTUSERNAME
GLOBAL _EXECENV SASStudio
GLOBAL _SASPROGRAMFILE
GLOBAL _SASSERVERNAME localhost
GLOBAL _SASWSTEMP C:\Users\[REDACTED]\images5ec3d9c65971416bb6d5b61d7358dd40
GLOBAL _SASNIS C:\Users\[REDACTED]

131  /*----- SAS generated prompt variable cleanup code.-----*/
132  %cleanupPromptValues();
133  /*----- End SAS generated prompt variable cleanup code.-----*/
134
135
136
137  OPTIONS NONOTES NOSTIMER NOSOURCE NOSYNTAXCHECK;
150

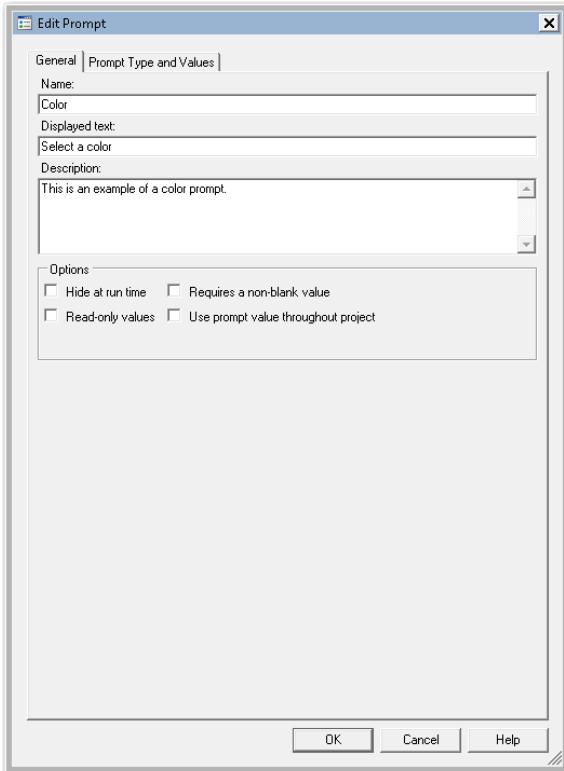
```

Display 263 – Variable Prompt Variable with Updated Values

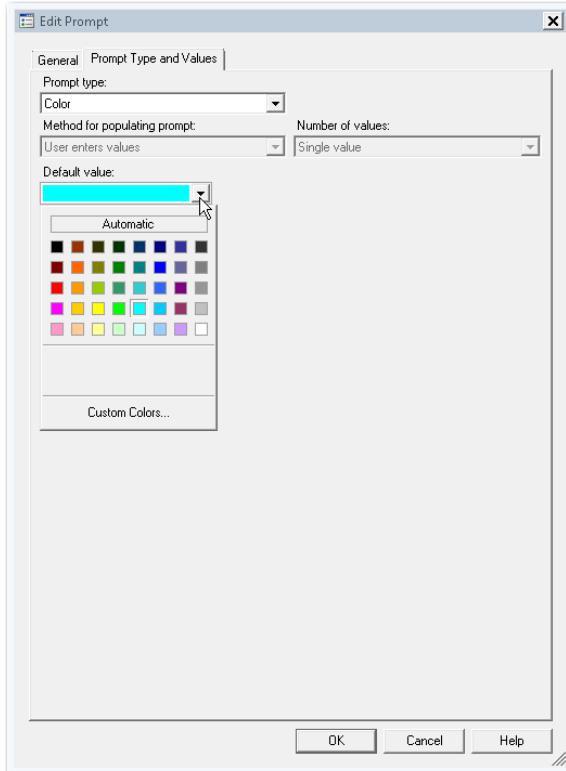
Color

SAS Enterprise Guide

In this example, a color prompt named Color is defined as shown in the following two displays.

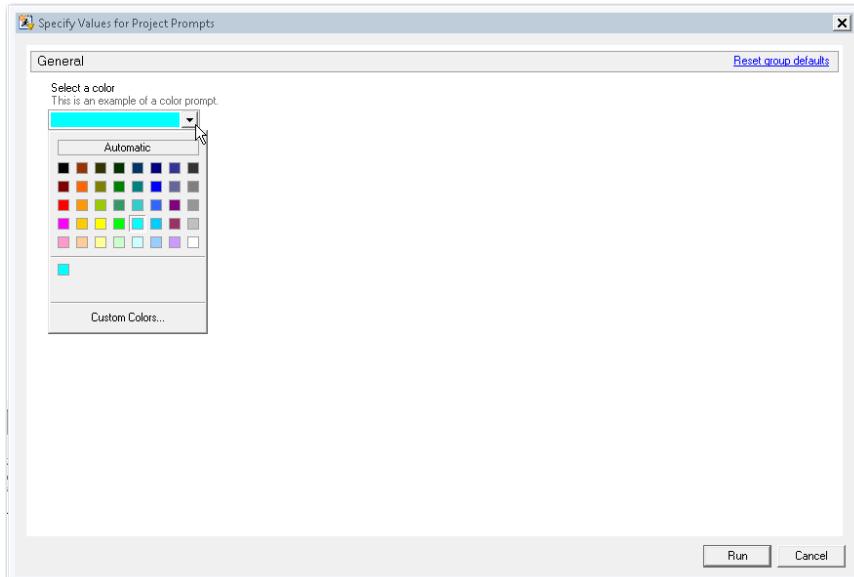


Display 264 - General Properties for Color Prompt



Display 265 - Type and Values for Color Prompt

When you run the Program node that depends on the prompt, the following dialog box appears:



Display 266 - Color Prompt in the Prompt Dialog Box

If the user leaves the default value in the color prompt, the following macro variable is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

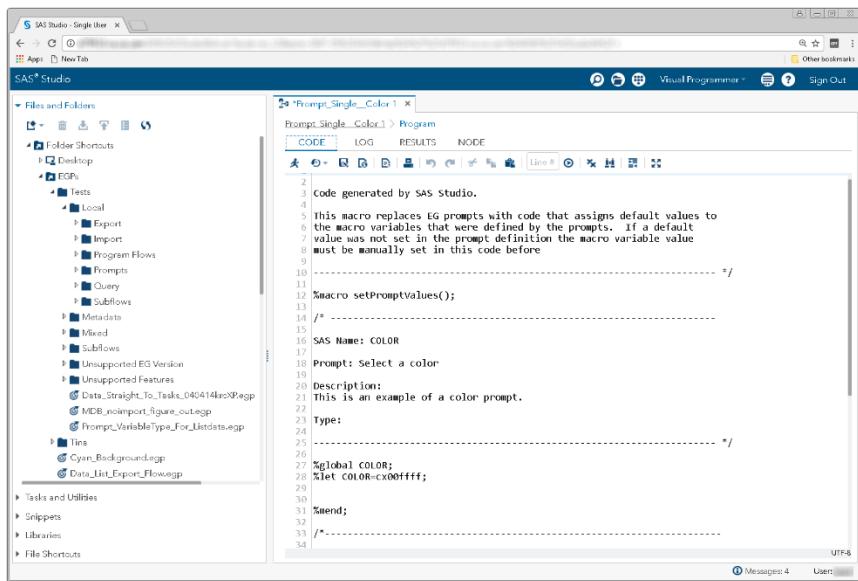
Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.

Display 267 – Values of GLOBAL Variables and and %SYMDEL Statement for Color Prompt

SAS Studio

The following display shows code that is added to the converted Program node for the color prompt in SAS Enterprise Guide.

A global variable named COLOR is created, and a %LET statement assigns the default value to COLOR. If you want to run your process flow using a different value for the COLOR prompt, you must manually update the value of the macro variable in the %LET statement.

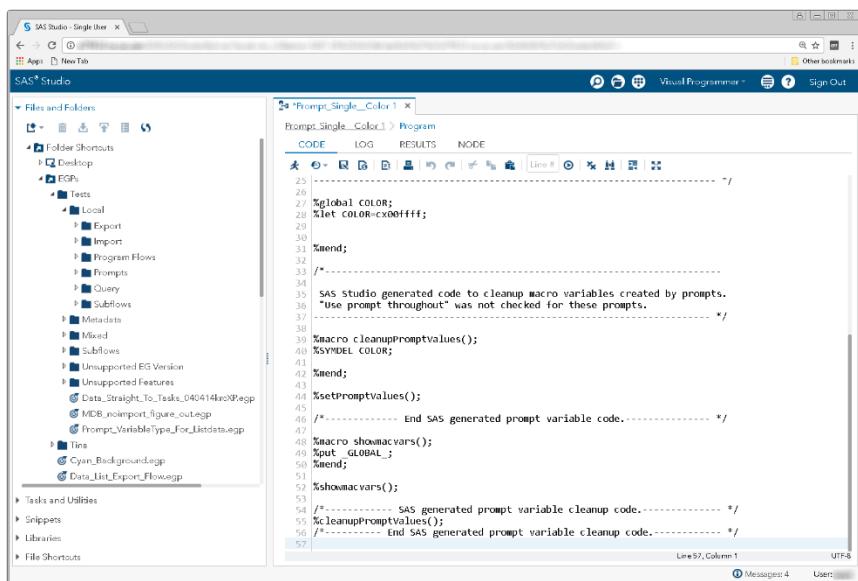


The screenshot shows the SAS Studio interface with a project titled "Prompt_Single_Color1". The left sidebar displays a file tree with various EG files and tasks. The main window shows a Visual Programmer editor with the following code:

```
2 Code generated by SAS Studio.
3
4 This macro replaces EG prompts with code that assigns default values to
5 the macro variables that were defined by the prompts. If a default
6 value was not set in the prompt definition the macro variable value
7 must be manually set in this code before
8
9 -----
10 Macro setPromptValues();
11 /**
12 * -----
13 * SAS Name: COLOR
14 *
15 * Prompt: Select a color
16 * Description:
17 * This is an example of a color prompt.
18 *
19 * Type:
20 */
21
22 Global COLOR;
23 Set COLOR=cccccccc;
24
25 /**
26 * -----
27 * End;
28 */
29
30 /**
31 * -----
32 * End;
33 */
34
```

Display 268 - Code for Color Prompt

Because the **Use prompt value throughout project** option is not checked, the %SYMDEL statement removes the COLOR macro variable.



The screenshot shows the SAS Studio interface with the same project and file tree as Display 268. The Visual Programmer editor now includes the %SYMDEL statement at the end of the code to remove the macro variable.

```
25 /**
26 * -----
27 Global COLOR;
28 Set COLOR=cccccccc;
29
30 /**
31 * -----
32 */
33
34 /**
35 * SAS studio generated code to cleanup macro variables created by prompts.
36 * "Use prompt throughout" was not checked for these prompts.
37 */
38
39 Macro cleanupPromptValues();
40 %SYMDEL COLOR;
41
42 /**
43 */
44 %setPromptValues();
45
46 /**
47 * ----- End SAS generated prompt variable code.
48 */
49 Macro showmacvars();
50 Input _GLOBAL_;
51
52 /**
53 */
54 /**
55 %cleanUpPromptValues();
56 */
57
```

Display 269 - %SYMDEL Code Removes the Color Macro Variable

Substituting a SAS Studio Task for a Color Prompt

1. Create a SAS Studio task with a control that represents the color prompt.
 - Add a label and a color input control. Enter Color as the name of the input control.
 - Set the default value to the default value shown in the generated `setPromptValues()` macro in the converted Program node.
 - Change the string for the input control to **Select a color**.

The following code is an example of a task that could be used as the color prompt.

```
<?xml version="1.0" encoding="UTF-8"?>
<Task schemaVersion="5.1" runNLS="never">
<Registration>
    <Name>Color selection</Name>
    <Description>Color selection</Description>
    <GUID></GUID>
    <Procedures>TBD</Procedures>
    <Version>3.6</Version>
    <Links>
        </Links>
</Registration>
<Metadata>
    <DataSources>
        </DataSources>
    <Options>
        <Option name="labelCOLOR" inputType="string">Select a color</Option>
        <Option name="color" defaultValue="cx00ffff" inputType="color">
            Choose a color
        </Option>
    </Options>
    </Metadata>
    <UI>
        <OptionItem option="labelCOLOR"/>
        <OptionItem option="color"/>
    </UI>
    <CodeTemplate>
        <![CDATA[
%global color;
%let color=$color;
        ]]>
    </CodeTemplate>
</Task>
```

The screenshot shows the SAS Studio interface with the following details:

- Title Bar:** SAS Studio - Single User
- File Menu:** File, New, Open, Save, Print, Exit, App, New Tab, Other bookmarks.
- SAS Studio Header:** Virtual Programmer, Sign Out.
- Left Sidebar:**
 - File and Folders:
 - Folder Shortcuts: Desktop, EGs, Tests (Local, Export, Import, Program Flows, Prompts, Query, Subflows).
 - Metadata, Mixed, Subflows.
 - Unsupported EG Version.
 - Unsupported Features:
 - Data_Straight_To_Tasks_040414kdgRegp
 - MDE_noimport_figure_out.egg
 - Prompt_Variable_Type_For_Listdata.egg
 - Tina, Cyan_Background.egg.
 - Tasks and Utilities.
 - Snippets.
 - Libraries.
 - File Shortcuts.- Code Editor:** The main area displays the XML code for a "Color_Selector" task. The code includes sections for registration, metadata, and a code template. It defines a color selection with a label and a color input field. The code template section contains logic to set a global color variable based on the selected color.

```
<Task version="1.0" encoding="UTF-8">
<Task schemaVersion="5.1" runILS="never">
<Registration>
  <Name>color selection</Name>
  <Description>color selection</Description>
  <GUIDs></GUIDs>
  <Procedures>TBD</Procedures>
  <version>3.6</version>
  <Links></Links>
</Registration>
<Metadata>
  <DataSources></DataSources>
  <Options>
    <option name="labelCOLOR" inputType="string">Select a color</option>
    <option name="color" defaultValue="cx00ffff" inputType="color">Choose a color</option>
  </Options>
</Metadata>
<UD>
  <optionItem option="labelCOLOR"/>
  <optionItem option="color"/>
</UD>
<UI>
<CodeTemplate>
<![CDATA[
%global color;
%let color=$color;
%put _GLOBAL_;
30
      ]>
</CodeTemplate>
</Task>
```

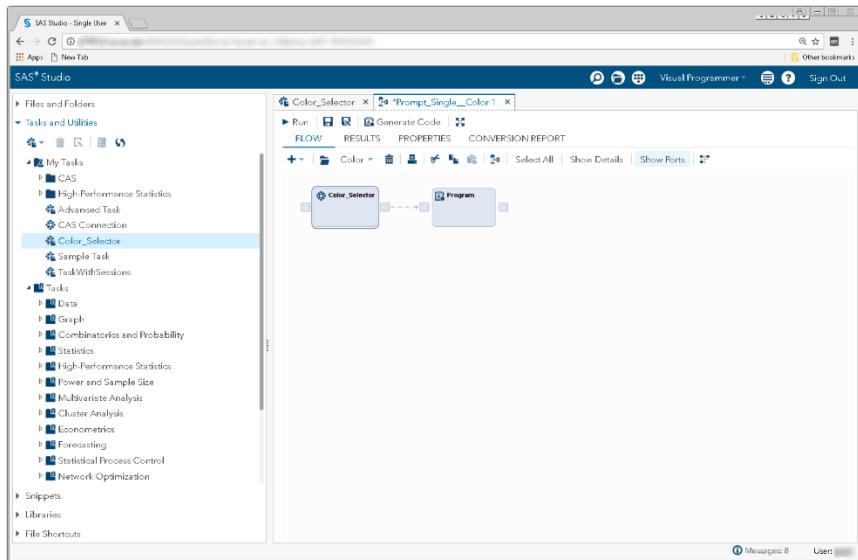
MyTasks/Color_Selector.egg

Line 16, Column 62

1 Messages: 10 User:

Display 270 - Replacement Task for Color Prompt

2. Save the replacement task to your **My Tasks** folder.
 3. Drag the task from **My Tasks** into your converted process flow.
 4. Link the output port of the Color_Selector node to the input port for the converted Program node.



Display 271 - Color_Selector Task Linked to Program Node

5. Comment out the `%setPromptValues()` macro call from the converted Program node. The macro code generated by the Color_Selector task replaces this code.

```

24 *Prompt_Single_Color.1 > Color_Selector
25
26 %global COLOR;
27 %let COLOR=cx00ffff;
28
29
30
31 %mend;
32
33 /*-----
34
35 SAS Studio generated code to cleanup macro variables created by prompts.
36 "Use prompt throughout" was not checked for these prompts.
37 -----*/
38
39 %macro cleanupPromptvalues();
40 %$PDEL COLOR;
41
42 %mend;
43
44 /*----- End SAS generated prompt variable code.-----*/
45
46 /*----- SAS generated prompt variable cleanup code.-----*/
47 %macro showmacvars();
48 %put _GLOBAL_;
49 %mend;
50
51
52 %showmacvars();
53
54 /*----- SAS generated prompt variable cleanup code.-----*/
55 %cleanupPromptvalues();
56 /*----- End SAS generated prompt variable cleanup code.-----*/

```

Display 272 - Commented Out %setPromptValues Macro Call

To run your flow with a different color value than the default value, open the Color_Selector node and select a color.

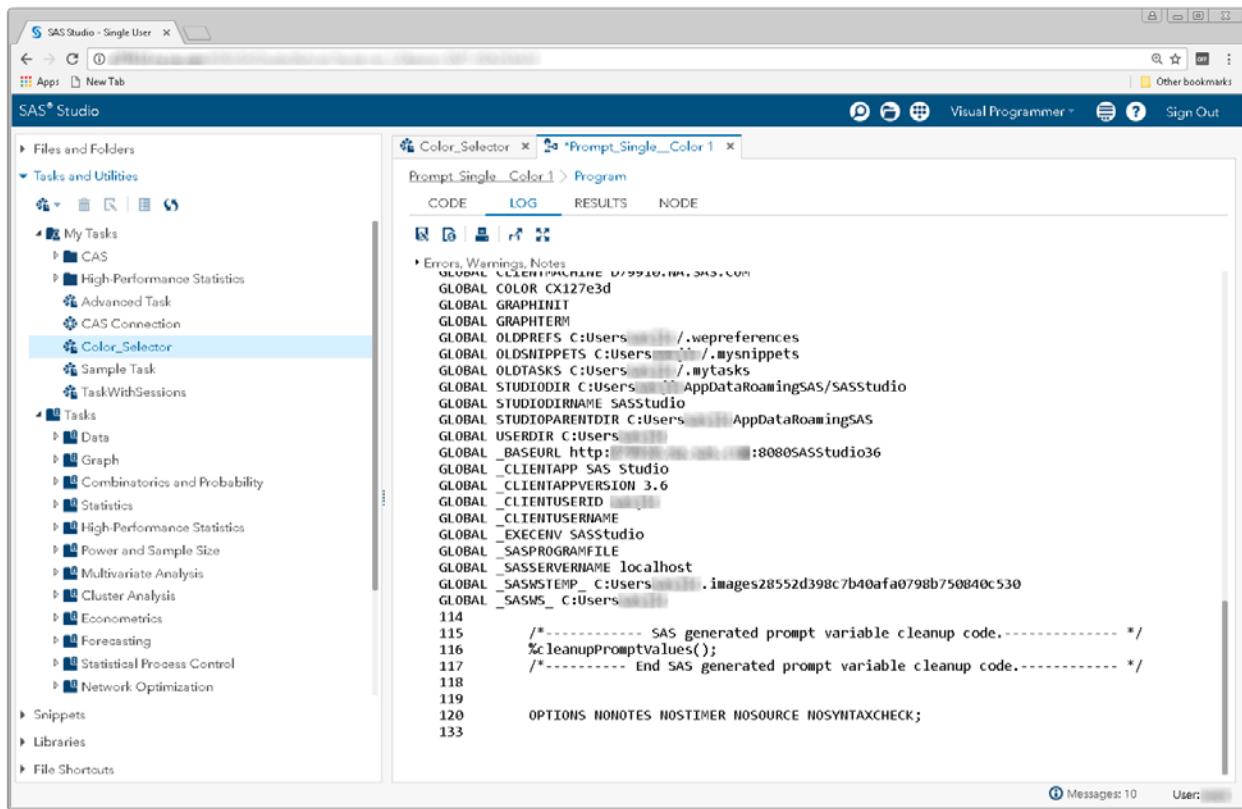
```

/*
1 * Task code generated by SAS studio 3.6
2 *
3 * Generated on '3/1/17, 11:04 AM'
4 * Generated by '_____'
5 * Generated on server '_____'
6 * Generated on SAS platform 'X64_7PRO MIN'
7 * Generated on SAS version '9.04.01MP12042013'
8 * Generated on browser 'Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.36'
9 * Generated on web client 'http://_____.com:8080'
10
11 */
12
13
14 %global color;
15 %let color=cx00ffff;
16 %put _GLOBAL_;
17

```

Display 273 – User Interface and Generated Code for Color_Selector Task

When you run the process flow, the global Color variable is set to the value of the selected color.



The screenshot shows the SAS Studio interface. On the left, the 'Tasks and Utilities' sidebar is open, displaying various task categories like My Tasks, CAS, High-Performance Statistics, etc. A specific task named 'Color_Selector' is highlighted. In the center, a process flow titled 'Prompt_Single_Color_1' is shown, with the 'Program' tab selected. The log window displays the following SAS code:

```
GLOBAL _CLIENTNAME 0/9910.10.0.0.0.0;
GLOBAL COLOR CX127e3d;
GLOBAL GRAPHINIT;
GLOBAL GRAPHTERM;
GLOBAL OLDPREFS C:Users/.../.wepreferences;
GLOBAL OLDSNIPPETS C:Users/.../.mysnippets;
GLOBAL OLDTASKS C:Users/.../.mytasks;
GLOBAL STUDIODIR C:Users/.../AppDataRoamingSAS/SASstudio;
GLOBAL STUDIODIRNAME SASstudio;
GLOBAL STUDIOPARENTDIR C:Users/.../AppDataRoamingSAS;
GLOBAL USERDIR C:Users/...
GLOBAL BASEURL http://...:8080SASstudio36;
GLOBAL _CLIENTAPP SAS Studio;
GLOBAL _CLIENTAPPVERSION 3.6;
GLOBAL _CLIENTUSERID ...;
GLOBAL _CLIENTUSERNAME ...;
GLOBAL _EXECENV SASstudio;
GLOBAL _SASPROGRAMFILE;
GLOBAL _SASSERVERNAME localhost;
GLOBAL _SASSTEMP_ C:Users/.../images28552d398c7b40afa0798b750840c530;
GLOBAL _SASWS_ C:Users/...
114
115 /*----- SAS generated prompt variable cleanup code,----- */
116 %cleanuppromptvalues();
117 /*----- End SAS generated prompt variable cleanup code.----- */
118
119
120 OPTIONS NONOTES NOSTIMER NOSOURCE NOSYNTAXCHECK;
133
```

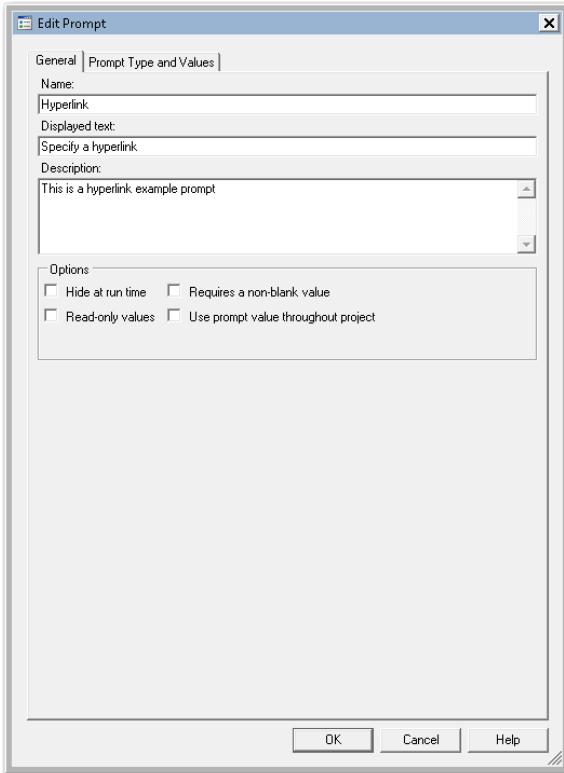
At the bottom right of the log window, there are status indicators: 'Messages: 10' and 'User: [redacted]'

Display 274 - Macro Variable With Updated Value for Color_Selector Task

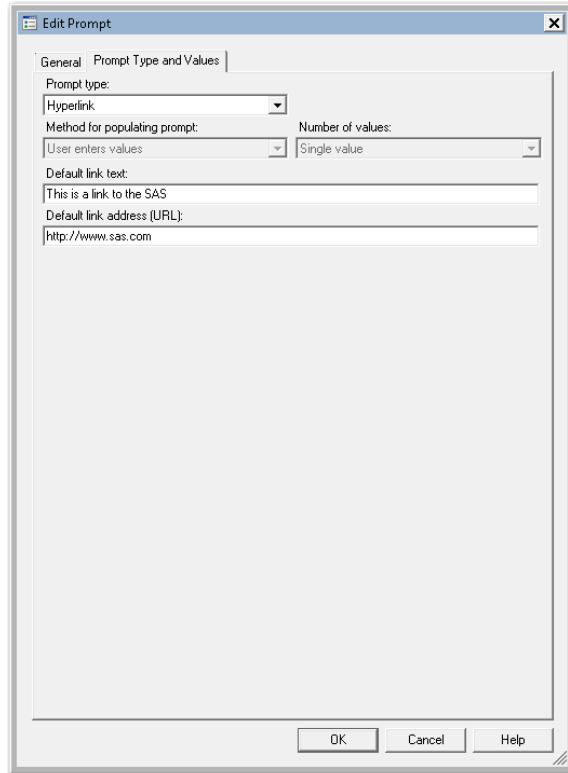
Hyperlink

SAS Enterprise Guide

In this example, a hyperlink prompt named Hyperlink is defined as shown in the following two displays.

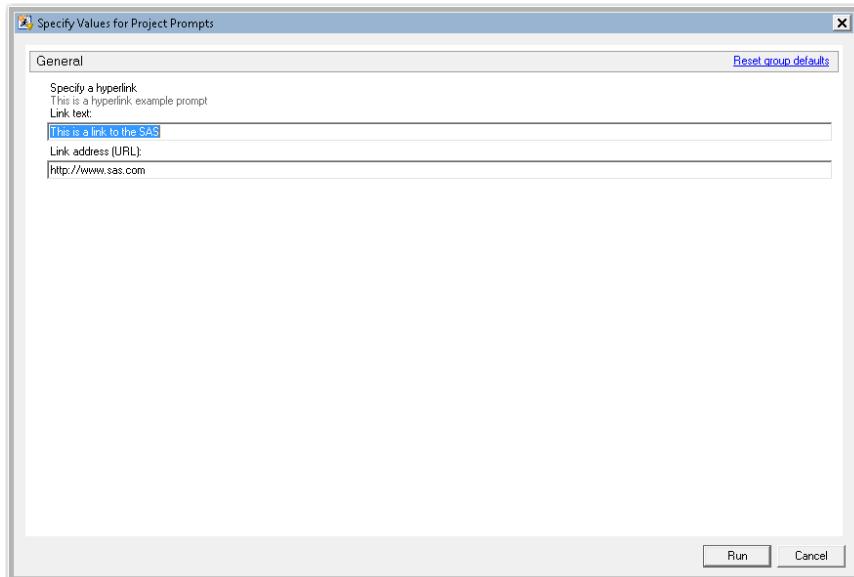


Display 275 - General Properties for Hyperlink Prompt



Display 276 - Type and Values for Hyperlink Prompt

When the code for the prompt is run, the following dialog box appears.



Display 277 - Hyperlink Prompt in Prompt Dialog Box

If the user leaves the default values in the hyperlink prompt field, the following code is generated by SAS Enterprise Guide in the Program node that depends on the prompt.

The %LET statements assign the values specified in the prompt dialog box to the Hyperlink* macro variables.

The screenshot shows the SAS Enterprise Guide interface with a "Program" node selected in the Project Tree. The code editor displays the following SAS code:

```

30      GOPTIONS NOACCESSIBLE;
31      %LET _CLIENTTASKLABEL="";
32      %LET _CLIENTPROJECTPATH="";
33      %LET _CLIENTPROJECTNAME="";
34      %LET _SASPROGRAMFILE="";
35      %SYMDEL Hyperlink_path;
36      %SYMDEL Hyperlink;
37
38      ;/*;*/quit;run;
39
40      ODS _ALL_ CLOSE;
41
42
43      QUIT; RUN;
44

```

The log summary pane shows one note:

Description	Line	Affected Code
NOTE: Values for macro variable _CLIENTPROJECTNAME are being passed from the previous step.	33	%LET _CLIENTPROJECTNAME=""

Display 278 - %LET Statements for Hyperlink Prompt

Because the **Use prompt value throughout project** option is not checked for this prompt, the %SYMDEL statements remove the macro variables at the end of the program.

The screenshot shows the SAS Enterprise Guide interface with a "Program" node selected in the Project Tree. The code editor displays the following SAS code, which includes extensive global macro variable definitions:

```

GLOBAL HYPERLINK 'This is a link to the SAS';
GLOBAL HYPERLINK_PATH 'http://www.sas.com';
GLOBAL SASHOMELOCATION "C:\Users\...\AppData\Local\Temp\SEG6376\SAS Temporary Files\_TD12516_D79910_\Prc2/*";
GLOBAL _CLIENTAPP 'SAS Enterprise Guide';
GLOBAL _CLIENTAPPAREV EG;
GLOBAL _CLIENTMACHINE '';
GLOBAL _CLIENTPROJECTNAME 'Prompt_Single_Hyperlink.egp';
GLOBAL _CLIENTPROJECTPATH 'C:\EGP\tests\Local\Prompts\Prompt_Single_Hyperlink.egp';
GLOBAL _CLIENTPROGRAM 'Program';
GLOBAL _CLIENTUSERID '';
GLOBAL _CLIENTUSERNAME 'J. Jeffreys-Chen';
GLOBAL _CLIENTVERSION '7.100.1.2651';
GLOBAL _INITIALIZEDSEVERID 1;
GLOBAL _SASHOSTNAME '';
GLOBAL _SASPROGRAMFILE '';
GLOBAL _SASSERVERNAME 'Local';
30
31      GOPTIONS NOACCESSIBLE;
32      %LET _CLIENTTASKLABEL="";
33      %LET _CLIENTPROJECTPATH="";
34      %LET _CLIENTPROJECTNAME="";
35      %LET _SASPROGRAMFILE="";
36      %SYMDEL Hyperlink_path;
37      %SYMDEL Hyperlink;
38
39      ;/*;*/quit;run;
40
41      ODS _ALL_ CLOSE;
42
43      QUIT; RUN;
44

```

The log summary pane shows one note:

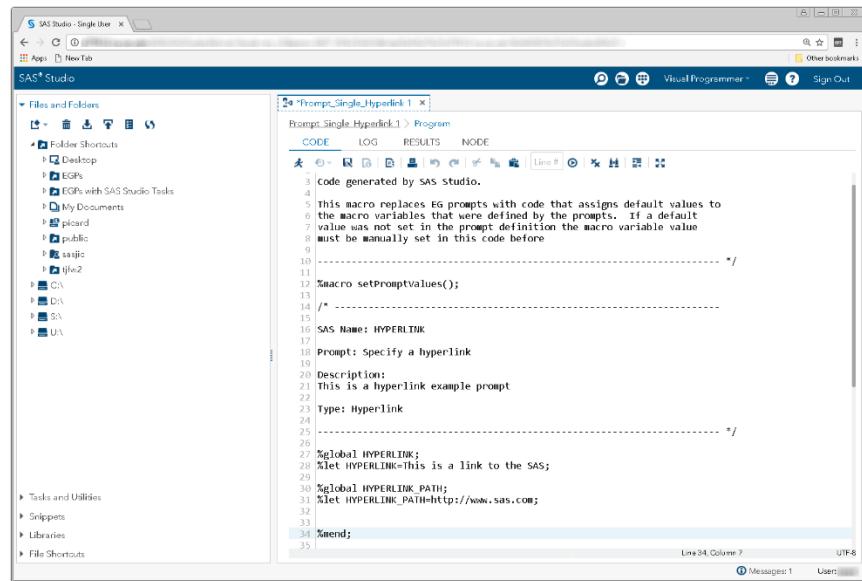
Description	Line	Affected Code
NOTE: Values for macro variable _CLIENTPROJECTNAME are being passed from the previous step.	33	%LET _CLIENTPROJECTNAME=""

Display 279 - Hyperlink Macro Variable GLOBAL definitions

SAS Studio

The following display shows code that is added to the converted Program node for the hyperlink prompt in SAS Enterprise Guide.

Global variables named HYPERLINK and HYPERLINK_PATH are created. The %LET statements assign the default values to HYPERLINK and HYPERLINK_PATH. If you want to run your process flow using a different input value for the HYPERLINK prompt, you must manually update the values of the macro variables in the %LET statement.

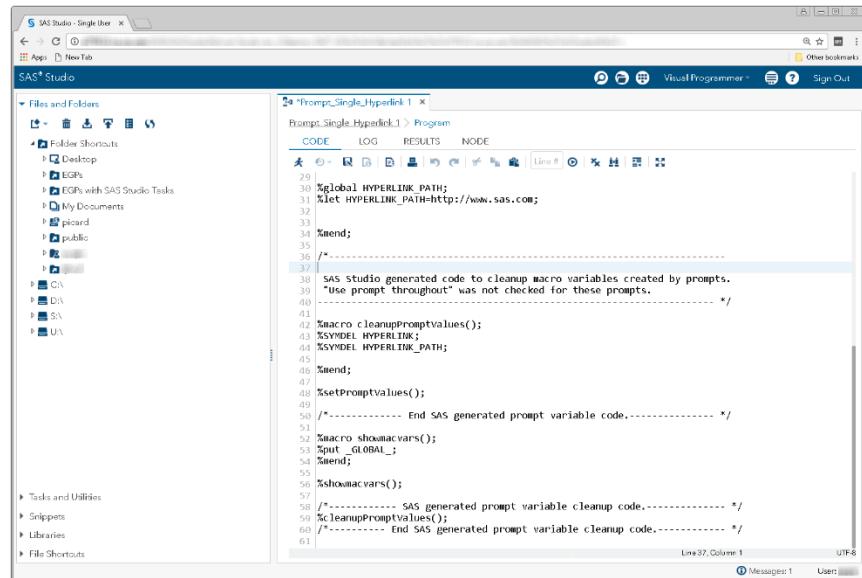


A screenshot of the SAS Studio interface. On the left is a sidebar with 'Files and Folders' containing 'Folder Shortcuts' (Desktop, EGP, EGP's with SAS Studio Tasks, My Documents, picard, public, sasjlt, tjev2), 'Tasks and Utilities', 'Snippets', 'Libraries', and 'File Shortcuts'. The main area shows a tab titled 'Prompt_Single_Hyperlink.1 > Program'. The code editor displays the following SAS code:

```
3 code generated by SAS studio.
4
5 This macro replaces EG prompts with code that assigns default values to
6 the macro variables that were defined by the prompts. If a default
7 value was not set in the prompt definition the macro variable value
8 must be manually set in this code before
9
10 -----
11 %macro setPromptValues();
12
13 /**
14 *-----
15 SAS Name: HYPERLINK
16
17 Prompt: Specify a hyperlink
18
19 Description:
20 This is a hyperlink example prompt
21
22 Type: Hyperlink
23
24 -----
25
26 %global HYPERLINK;
27 %let HYPERLINK=this is a link to the SAS;
28
29 %global HYPERLINK_PATH;
30 %let HYPERLINK_PATH=http://www.sas.com;
31
32
33
34 %mend;
35
```

Display 280 - Macro Variables for Hyperlink Prompt

Because the **Use prompt value throughout project** option is not checked, the %SYMDEL statements remove the HYPERLINK macro variables.



A screenshot of the SAS Studio interface, similar to the previous one but with additional code at the bottom. The code editor displays the following SAS code:

```
29 |
30 %global HYPERLINK_PATH;
31 %let HYPERLINK_PATH=http://www.sas.com;
32
33
34 %mend;
35
36 /**
37 SAS studio generated code to cleanup macro variables created by prompts.
38 "Use prompt throughout" was not checked for these prompts.
39
40 /**
41 %macro cleanupPromptvalues();
42 %symdel HYPERLINK;
43 %symdel HYPERLINK_PATH;
44
45 %mend;
46
47 %setPromptvalues();
48
49 /**
50 ----- End SAS generated prompt variable code.-----
51
52 %macro showmacvars();
53 %put _GLOBAL_;
54 %mend;
55
56 %showmacvars();
57
58 /**
59 ----- SAS generated prompt variable cleanup code.-----
60 %cleanupPromptvalues();
61 /**
62 ----- End SAS generated prompt variable cleanup code.-----
63
```

Display 281 - %SYMDEL Statements Remove the HYPERLINK Macro Variables

Substituting a SAS Studio Task for a Hyperlink Prompt

1. Create a SAS Studio task with a control that replaces the hyperlink prompt.
 - Add a label and a text input control for the hyperlink and the hyperlink path. For the name of the input control, use `Hyperlink` and `Hyperlink_Path`.
 - Set the default value to the default value shown in the generated `setPromptValues()` macro in the converted Program node.
 - Change the strings of the input controls to match the strings in the hyperlink prompt.

The following code is an example of a task that could be used as the hyperlink prompt.

```
<?xml version="1.0" encoding="UTF-8"?>
<Task schemaVersion="5.1" runNLS="never">
    <Registration>
        <Name>Hyperlink Prompt</Name>
        <Description>This is a hyperlink prompt task.</Description>
        <GUID></GUID>
        <Procedures></Procedures>
        <Version>3.6</Version>
        <Links></Links>
    </Registration>

    <Metadata>
        <DataSources>
        </DataSources>

        <Options>
            <Option name="labelDescription" inputType="string">
                Specify a hyperlink
            </Option>

            <Option name="Hyperlink"
                   defaultValue="This is a link to the SAS"
                   inputType="inputtext">
                Link text:
            </Option>

            <Option name="Hyperlink_Path"
                   defaultValue="http://www.sas.com"
                   inputType="inputtext">
                Link address (URL):
            </Option>
        </Options>
    </Metadata>

    <UI>
        <OptionItem option="labelDescription"/>
        <OptionItem option="Hyperlink"/>
        <OptionItem option="Hyperlink_Path"/>
    </UI>

    <CodeTemplate>
        <![CDATA[
%global Hyperlink;
%global Hyperlink_Path;
%let Hyperlink=$Hyperlink;
%let Hyperlink_Path=$Hyperlink_Path;

        ]]>
    </CodeTemplate>
</Task>
```

The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. On the left, the file tree shows a folder named 'Hyperlink_Prompt' containing a file 'Hyperlink_Prompt.sas'. The main area displays the SAS code for the task:

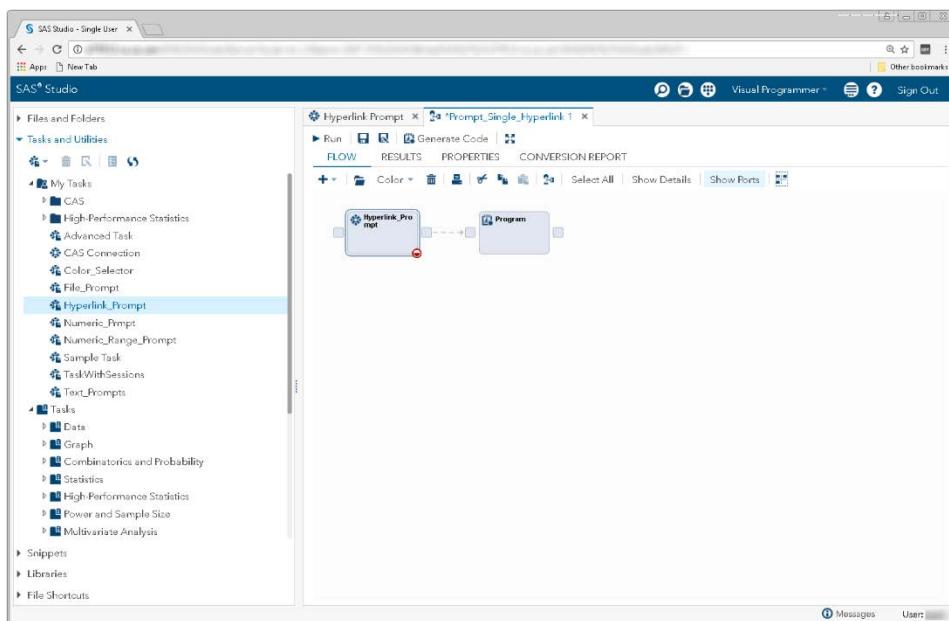
```

</Registration>
<metadata>
  <datasources>
    </datasources>
</metadata>
<options>
  <option name="labelDescription" inputtype="string">Specify a hyperlink</option>
  <option name="Hyperlink" defaultValue="This is a link to the SAS" inputtype="inputtext">Link text:</option>
  <option name="Hyperlink_Path" defaultValue="http://www.sas.com" inputtype="inputtext">Link address (URL):</option>
</options>
<UI>
  <optionItem option="labelDescription"/>
  <optionItem option="Hyperlink"/>
  <optionItem option="Hyperlink_Path"/>
</UI>
<CodeTemplate>
<![CDATA[
38 %global Hyperlink;
39 %global Hyperlink_Path;
40 %let Hyperlink=$Hyperlink;
41 %let Hyperlink_Path=$Hyperlink_Path;
42
43 ]]>

```

Display 282 - Replacement Task for Hyperlink Prompt

2. Save the new task to your **My Tasks** folder.
3. Drag the task from **My Tasks** into your converted process flow.
4. Link the output port from the **Hyperlink_Prompt** task to the input node for the converted Program node.



Display 283 - Hyperlink_Prompt Task Linked to Program

5. Comment out the `%setPromptValues()` macro call from the converted Program node. The macro code generated by the hyperlink task replaces this code.

```

3.1 %let HYPERLINK_PATH=http://www.sas.com;
3.2
3.3
3.4 %end;
3.5
3.6 /*-----
3.7   SAS Studio generated code to cleanup macro variables created by prompts.
3.8   "use prompt throughout" was not checked for these prompts.
3.9 */
3.10
3.11 %macro cCleanupPromptValues();
3.12   %$SYNDEL HYPERLINK;
3.13   %$SYNDEL HYPERLINK_PATH;
3.14
3.15 %end;
3.16
3.17 /*%setPromptValues();
3.18 */
3.19 /*----- End SAS generated prompt variable code.----- */
3.20
3.21 %macro showmacvars();
3.22   %put _GLOBAL_;
3.23 %end;
3.24
3.25 %showmacvars();
3.26
3.27 /*----- SAS generated prompt variable cleanup code.----- */
3.28 %cCleanupPromptValues();
3.29 /*----- End SAS generated prompt variable cleanup code.----- */
3.30
3.31
3.32
3.33
3.34
3.35
3.36
3.37
3.38
3.39
3.40
3.41
3.42
3.43
3.44
3.45
3.46
3.47
3.48
3.49
3.50
3.51
3.52
3.53
3.54
3.55
3.56
3.57
3.58
3.59
3.60
3.61

```

Display 284 - Commented Out `%setPromptValues` Macro Call

To run your flow with different Hyperlink values than the default values, open the `Hyperlink_Prompt` node and specify different values.

OPTIONS INFORMATION NODE

CODE LOG RESULTS

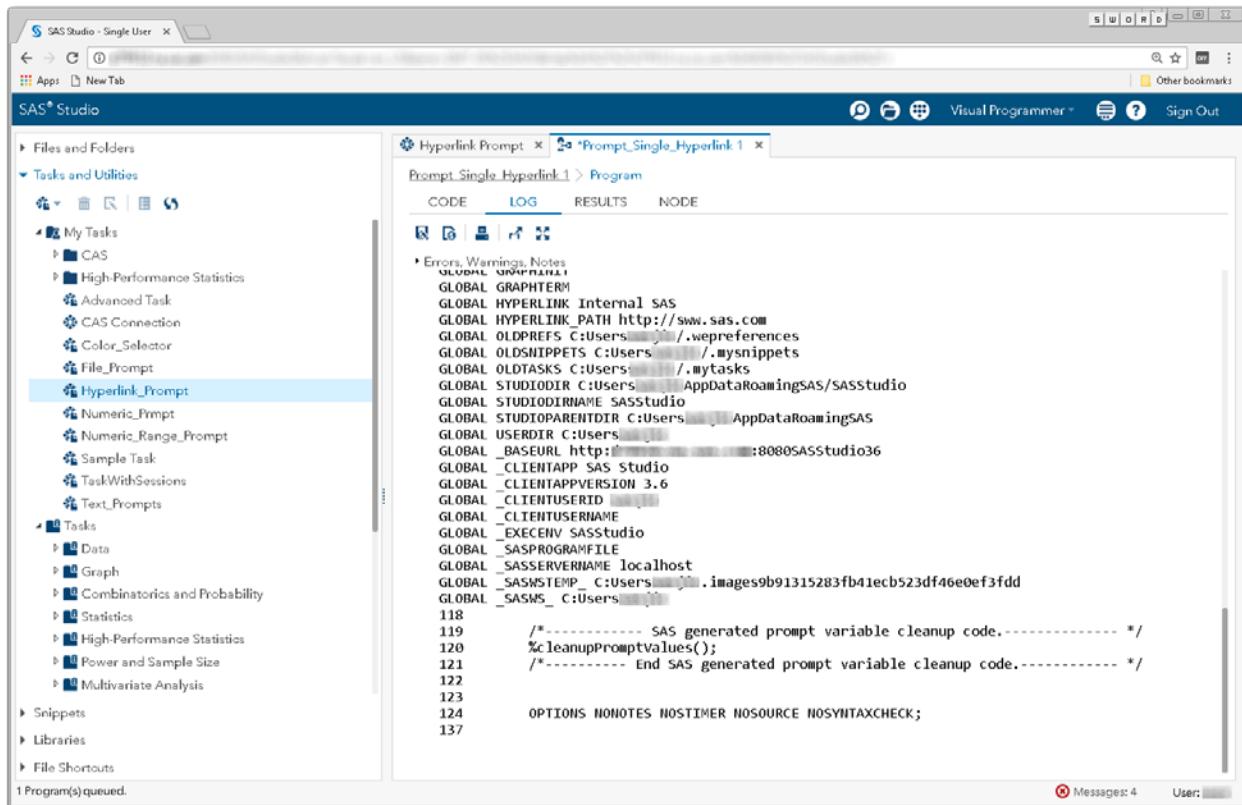
```

1 /*-
2 *
3 * Task code generated by SAS studio 3.6
4
5 * Generated on '2/2/17, 12:40 PM'
6 * Generated by ' '
7 * Generated on server ' '
8 * Generated on SAS platform 'X64_7PRO MIN'
9 * Generated on SAS version '9.04.011P12042013'
10 * Generated on browser 'Mozilla/5.0 (Windows NT 6.1; '
11 * Generated on web-client 'http:// '
12 *
13 */
14
15 %global Hyperlink;
16 %global Hyperlink_Path;
17 %let Hyperlink=Internal SAS;
18 %let Hyperlink_Path=http://www.sas.com;

```

Display 285 – User Interface and Generated SAS Code for `Hyperlink_Prompt` Task

When you run the process flow, the global Hyperlink* variables are set to the values specified in the task.



The screenshot shows the SAS Studio interface with the Visual Programmer tab selected. The left sidebar shows a tree view of tasks and utilities, with 'Hyperlink_Prompt' selected. The main area displays a log tab titled 'Prompt_Single_Hyperlink_1 > Program'. The log content shows various global macro variable assignments:

```
GLOBAL _GRAPHTERM
GLOBAL HYPERLINK_Internal_SAS
GLOBAL HYPERLINK_PATH http://www.sas.com
GLOBAL OLDPREFS C:\Users\[REDACTED]\.wepreferences
GLOBAL OLDSNIPPETS C:\Users\[REDACTED]\.mysnippets
GLOBAL OLDTASKS C:\Users\[REDACTED]\.mytasks
GLOBAL STUDIODIR C:\Users\[REDACTED]\AppDataRoamingSAS/SASstudio
GLOBAL STUDIODIRNAME SASstudio
GLOBAL STUDIOPARENTDIR C:\Users\[REDACTED]\AppDataRoamingSAS
GLOBAL USERDIR C:\Users\[REDACTED]
GLOBAL BASEURL http://[REDACTED]:8080SASstudio36
GLOBAL CLIENTAPP SAS studio
GLOBAL CLIENTAPPVERSION 3.6
GLOBAL CLIENTUSERID [REDACTED]
GLOBAL CLIENTUSERNAME [REDACTED]
GLOBAL EXECENV SASstudio
GLOBAL SASPROGRAMFILE
GLOBAL SASSERVERNAME localhost
GLOBAL SASNTEMP_ C:\Users\[REDACTED]\.images9b91315283fb41ecb523df46e0ef3fdd
GLOBAL SASN_ C:\Users\[REDACTED]
118
119      /*----- SAS generated prompt variable cleanup code.----- */
120      %cleanuppromptvalues();
121      /*----- End SAS generated prompt variable cleanup code.----- */
122
123
124      OPTIONS NONOTES NOSTIMER NOSOURCE NOSYNTAXCHECK;
137
```

At the bottom right of the log window, there are 'Messages: 4' and 'User: [REDACTED]' indicators.

Display 286 – Macro Variables for Hyperlink_Prompt Task with Updated Values

MULTIPLE FLOWS

If there are multiple process flows defined in a SAS Enterprise Guide project, they will appear as subflows when the EGP is converted to a SAS Studio process flow.

EGP ELEMENTS THAT ARE NOT SUPPORTED BY SAS STUDIO

Many elements of SAS Enterprise Guide projects are not currently supported by SAS Studio.

- project tree
- submit to grid
- Decision Manager elements
- scheduling
- Git source control
- explore data list
- process flow zoom – Note: In SAS Studio, you can use browser zoom instead.
- process flow grid view
- process flow layout

SAS ENTERPRISE GUIDE MACROS

Code generated by the SAS Enterprise Guide tasks sometimes includes references to macros. The following macros are also included in SAS Studio:

- _eg_conditional_dropds - Used by code generated for EGP Query nodes.

MACRO VARIABLES IN SAS ENTERPRISE GUIDE

AVAILABLE IN SAS STUDIO

- _SASSERVERNAME
- _CLIENTUSERID
- _CLIENTUSERNAME
- _CLIENTMACHINE

NOT AVAILABLE IN SAS STUDIO

Because these SAS Enterprise Guide macros variables are not set in SAS Studio, you see warnings if your code uses them.

- _CLIENTAPP
- _CLIENTPROJECTNAME
- _CLIENTPROJECTPATH
- _CLIENTTASKFILTER
- _CLIENTTASKLABEL
- _CLIENTUSERNAME
- _CLIENTVERSION
- _SASHOSTNAME

- _SASPROGRAMFILE

LIMITATIONS

- Only EGP files saved with SAS Enterprise Guide 7.1 or later are supported.

Interesting behaviors:

- When the process flows are opened the first time in SAS Studio, they are "Arranged" using the same functionality that you get when you press the "Arrange" button on the SAS Studio process flow.

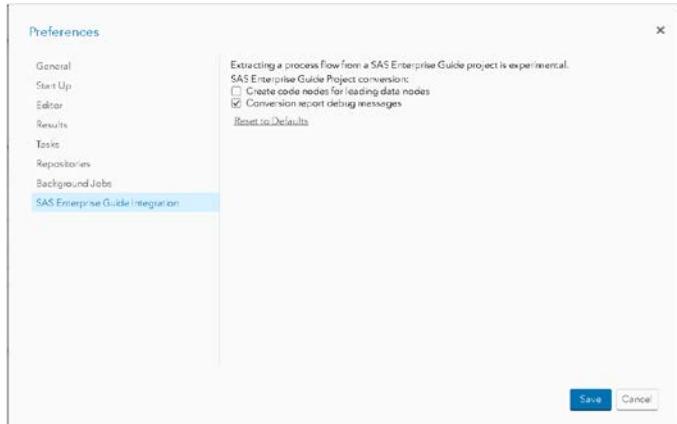
FUTURES

- Data node support in SAS Studio process flows
- Reformatted conversion report
- Command line interface
- Prompt support in SAS Studio process flows

DEBUGGING

To find the unpacked EGP directory, select the **Conversion report debug messages** option in SAS Studio. This option is available from the Preferences window.

Note: The **Create code node for leading data nodes** option is no longer supported (with the 3.6 hotfix) and will be removed in a future release. In SAS Studio 3.6 (with the 3.6 hotfix), all Data nodes are represented in the converted process flow.



Display 287 - SAS Enterprise Guide Integration Preferences

ENABLING EGP EXTRACTION FEATURE

To allow the opening of EGP files in SAS Studio, the SAS Studio Administrator can sets the `webdms.allowEGPOpen` property to `true` in the `config.properties` file:

```
webdms.allowEGPOpen=true
```

FREQUENTLY ASKED QUESTIONS

Q. Can I save my converted SAS Studio process flows back to SAS Enterprise Guide projects?

A. No.

REFERENCES

SAS Institute Inc. 2016. *SAS Studio 3.6: User's Guide*. Cary, NC: SAS Institute Inc.

SAS Institute Inc. 2016. [*SAS Studio 3.6: Developer's Guide to Writing Custom Tasks*](#). Cary, NC: SAS Institute Inc.

CONCLUSION

The new EGP conversion functionality is just beginning to take shape. Process flows from SAS Enterprise Guide can be opened in SAS Studio. Process flow nodes are transformed to nodes that SAS Studio supports, but these nodes might not function exactly the same way they do in SAS Enterprise Guide. Currently, SAS Studio process flows cannot be opened in SAS Enterprise Guide.

CONTACT INFORMATION

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

Jennifer Jeffreys-Chen
SAS Institute Inc.
919-531-6208
J.Jeffreys-Chen@sas.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.



To contact your local SAS office, please visit: sas.com/offices

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. Copyright © 2017, SAS Institute Inc. All rights reserved.