

A Quick Bite on SAS® Studio Custom Tasks

Michael Kola, Salient CRGT; Girish Patel, Jay Consultants, LLC

ABSTRACT

Custom Tasks in SAS Studio help users generate reports through a point and click user interface (UI). As a developer, there will be a need to create custom tasks for non-SAS programmers as well as novice SAS users or programmers. After creating a custom task, they can be shared with stakeholders using the common task model (CTM) or common task kit (CTK) files. Tasks are saved as CTM files or CTK files. CTK files are CTM files with roles and options pre-selected. The file could be shared through email or by just placing it on a shared area. The user simply has to run the file and it generates the User Interface, as the user navigates through the UI, the SAS code is generated simultaneously, which will help the user review the SAS code related to the UI. The task framework is flexible. All tasks use the same common task model and the Velocity Template Language. The task consists of task elements and they are: Registration, Metadata, UI, Dependencies, Requirements and Code Template. We will use all these elements to build our example custom task.

INTRODUCTION

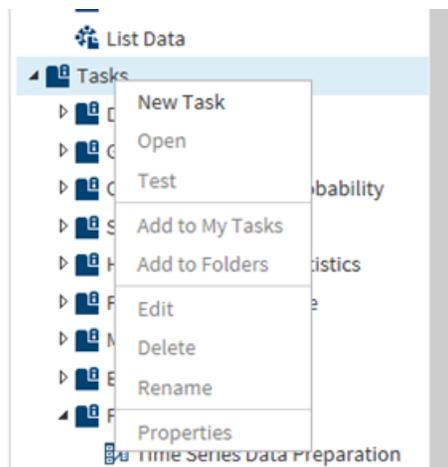
The custom task that is built in this paper is a UI screen with the option to choose datasets and variables with filters, options and statistics tabs (used only for Proc means). The task has a Proc Group in it, which allows the user to choose from Proc Means, Proc Freq and Proc Print.

The custom task is built using HTML and velocity code. The HTML code that the custom task uses has a framework and the framework includes task elements such as REGISTRATION, METADATA, UI, DEPENDENCIES and a CODE TEMPLATE.

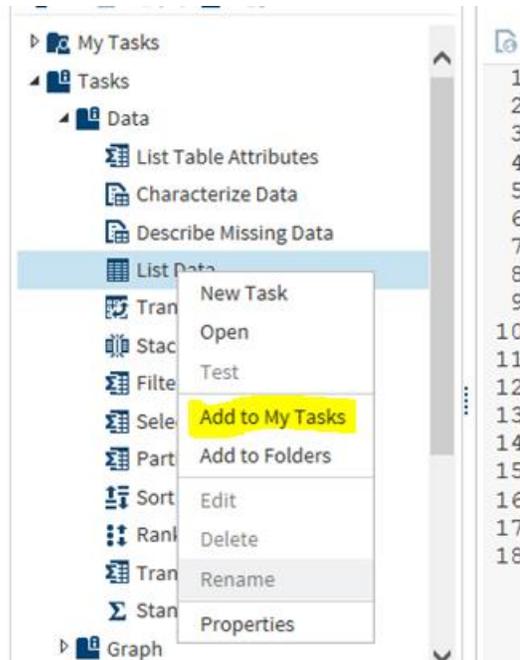
This paper discusses all of the aforementioned elements and provides an explanation for each task along with their specific application in the developed Custom Task.

CREATING A CUSTOM TASK

There are two ways to create custom tasks. One way to create it is by expanding the “Tasks and Utilities,” right clicking on “Tasks” and selecting “New Task”



Another way to create it is by selecting the existing SAS task and adding it to my tasks. This can be done by right clicking on the existing SAS task (in this situation, “List Data”) and selecting “Add to My Tasks” This will add the existing task to My Tasks and now the user will have the ability to edit this task.



TASK ELEMENTS (HTML) INVOLVED IN BUILDING THE CUSTOM TASK

REGISTRATION:

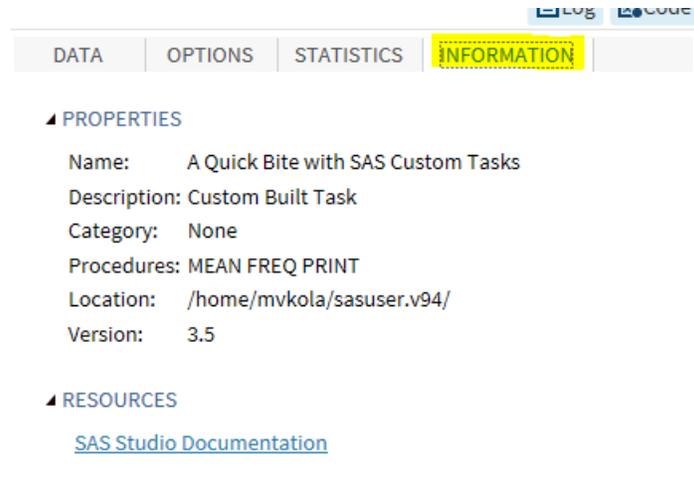
The Registration element represents a collection of metadata for the task. This element is required in order to know the type of task.

In the code below, the registration element contains child elements Name, Description, GUID, Procedures, Version and Links. The values of these elements are self-explanatory. Since this task is built for the sole purpose of this paper, the child elements describe the current task type in the Registration element.

```

<Registration>
  <Name>A Quick Bite with SAS Custom Tasks</Name>
  <Description>Custom Built Task</Description>
  <GUID>726B2B12-4542-4200-96F4-960C645D767F</GUID>
  <Procedures>MEAN FREQ PRINT</Procedures>
  <Version>3.5</Version>
  <Links>
    <Link href="http://support.sas.com/software/products/sasstudio/index.html#s1=2">SAS Studio Documentation</Link>
  </Links>
</Registration>
  
```

When this code is executed, the following User Interface is displayed under INFORMATION tab:



METADATA:

The following code is for the Metadata element. This element contains child elements which are DataSources and Options.

The DataSources contains child elements DataSource and Roles. The DataSource element specifies the information about the dataset for the task.

The Roles element identifies the variables that must be assigned in order to run the task and groups the individual role assignments that are needed for the task.

The Roles tag has a child element called Role, which describes one type of role assignment for the task.

```
<Metadata>
  <DataSources>
    <DataSource name="dataset">
      <Roles>
        <Role maxVars="0" minVars="0" name="var1" nlskey="var1key"
          order="true" type="A">Variable 1:</Role>
        <Role maxVars="1" minVars="0" name="var2" nlskey="var2key"
          order="true" type="A">Variable 2:</Role>
        <Role maxVars="1" minVars="0" name="var3" nlskey="var3key"
          order="true" type="A">Variable 3:</Role>
      </Roles>
    </DataSource>
  </DataSources>
</Metadata>
```

When you run this code, you get the data and roles section as follows:

The DataSource element helps display the dataset used and the Roles are Variable 1, Variable 2 and Variable 3:

DATA

OPTIONS

STATISTICS

INFORMATION

▲ DATASOURCE

SASHELP.CARS

▶ PROC GROUP

▲ FILTER 1

Variable 1:



Column

Variable 2: (1 item)



Column

Value:

Select a value

▲ FILTER 2

Variable 3: (1 item)



Column

Value2:

Select a value

OPTIONS:

The Options element identifies the options that are required in order to run the task. The option tag, which is the only child of the options element, describes the assigned option.

```
<Options>
  <Option inputType="string" name="datatab">DATA</Option>
  <Option inputType="string" name="datagr">DATASOURCE</Option>
  <Option name="PROCS" inputType="string">PROC GROUP</Option>
  <Option defaultValue="Print" inputType="combobox"
    name="List" width="264px">Select from the List:</Option>
  <Option inputType="string" name="Print">Print</Option>
  <Option inputType="string" name="Freq">Freq</Option>
  <Option inputType="string" name="Means">Means</Option>
  <Option inputType="string" name="filter1">FILTER 1</Option>
  <Option indent="1" inputType="distinct" name="values1" source="var2" max="200">Value:</Option>
  <Option inputType="string" name="filter2">FILTER 2</Option>
  <Option indent="1" inputType="distinct" name="values2" source="var3">Value2:</Option>
</Options>
```

The output for the above options will display the following screen:

The screenshot shows a software interface with four tabs: DATA, OPTIONS, STATISTICS, and INFORMATION. The OPTIONS tab is selected. Below the tabs, there are four sections, each with a dropdown arrow on the left:

- DATASOURCE:** A dropdown menu showing "SASHELP.CARS" and a small icon to the right.
- PROC GROUP:** A label "Select from the List:" followed by a dropdown menu showing "Print".
- FILTER 1:** A label "Variable 1:" followed by a text input field containing "Column". Above the field are icons for up, down, delete, and add. Below the field is a "Value:" label and a dropdown menu showing "Select a value".
- FILTER 2:** A label "Variable 3: (1 item)" followed by a text input field containing "Column". Above the field are icons for delete and add. Below the field is a "Value2:" label and a dropdown menu showing "Select a value".

```

<Option inputType="string" name="optionstab">OPTIONS</Option>
<Option inputType="string" name="outputgrp">OUTPUT DATA SET</Option>
<Option defaultValue="filter" indent="1"
inputType="outputdata" name="outputDS" required="true">Data set name (Please name your output data set):</Option>

```

```

<Option name="BASICOPTIONS" inputType="string">BASIC OPTIONS</Option>
<Option defaultValue="Print" inputType="combobox" name="ROWS2LIST" width="264px">Rows to list:</Option>
<Option inputType="string" name="all">All rows</Option>
<Option inputType="string" name="firstnrows">First n rows</Option>
<Option decimalPlaces="0" defaultValue="10" indent="1" inputType="numstepper" maxValue="999999" minValue="1"
name="NVALUE" required="true">Rows (n):</Option>

```

DATA **OPTIONS** STATISTICS INFORMATION

▲ OUTPUT DATA SET

* Data set name (Please name your output data set):

SASGF18

▲ BASIC OPTIONS

Rows to list: All rows

* Rows (n): 10

```

<Option inputType="string" name="statistics">STATISTICS</Option>
<Option inputType="string" name="GROUPCHECK">Means STATISTICS</Option>
<Option inputType="string" indent="1" name="labelCHECK">Select the statistics you would like included in the output dataset.</Option>
<Option defaultValue="0" indent="1" inputType="checkbox" name="chkMEAN">Mean</Option>
<Option defaultValue="0" indent="1" inputType="checkbox" name="chkMIN">Minimum</Option>
<Option defaultValue="0" indent="1" inputType="checkbox" name="chkMAX">Maximum</Option>
<Option defaultValue="0" indent="1" inputType="checkbox" name="chkSTD">Standard Deviation</Option>
<Option defaultValue="0" indent="1" inputType="checkbox" name="chkLCLM">LCLM</Option>
<Option defaultValue="0" indent="1" inputType="checkbox" name="chkUCLM">UCLM</Option>
<Option defaultValue="0" indent="1" inputType="checkbox" name="chkP25">25th Percentile</Option>
<Option defaultValue="0" indent="1" inputType="checkbox" name="chkP75">75th Percentile</Option>
</Options>

```

</Metadata>

DATA OPTIONS **STATISTICS** INFORMATION

▲ Means STATISTICS

Select the statistics you would like included in the output dataset.

- Mean
- Minimum
- Maximum
- Standard Deviation
- LCLM
- UCLM
- 25th Percentile
- 75th Percentile

At the end of the options tag the Metadata parent element will be closed.

USER INTERFACE (UI):

The UI Element contains Container child element and the Container contains Group element as a child and the Group contains the DataItem, OptionChoice and RoleItem elements as its child. The OptionChoice contains OptionItem as a child.

```
<UI>
  <Container option="datatab">
    <Group open="true" option="datagrpf">
      <DataItem data="dataset"/>
    </Group>
    <Group open="true" option="PROCS">
      <OptionChoice option="List">
        <OptionItem option="Print"/>
        <OptionItem option="Freq"/>
        <OptionItem option="Means"/>
      </OptionChoice>
    </Group>
    <Group open="true" option="filter1">
      <RoleItem role="var1"/>
      <RoleItem role="var2"/>
      <OptionItem option="values1"/>
    </Group>
    <Group open="true" option="filter2">
      <RoleItem role="var3"/>
      <OptionItem option="values2"/>
    </Group>
  </Container>
```

The following is the UI for the DATA container:

DATA | OPTIONS | STATISTICS | INFORMATION

DATA SOURCE: SASHELP.CARS

PROC GROUP: Select from the List: Print

FILTER 1: Variable 1: Column

Variable 2: (1 item) Column

Value: Select a value

FILTER 2: Variable 3: (1 item) Column

Value2: Select a value

```

<Container option="optionstab">
  <Group open="true" option="outputgrp">
    <OptionItem option="outputDS"/>
  </Group>
  <Group open="true" option="BASICOPTIONS">
    <OptionChoice option="ROWS2LIST">
      <OptionItem option="all"/>
      <OptionItem option="firstnrows"/>
    </OptionChoice>
    <OptionItem option="NVALUE"/>
  </Group>
</Container>

```

DATA	OPTIONS	STATISTICS	INFORMATION
------	----------------	------------	-------------

▲ OUTPUT DATA SET

*Data set name (Please name your output data set):

▲ BASIC OPTIONS

Rows to list:

*Rows (n):

```

<Container option="statistics">
<Group open="true" option="GROUPCHECK">
  <OptionItem option="labelCHECK"/>
    <OptionItem option="chkMEAN"/>
    <OptionItem option="chkMIN"/>
    <OptionItem option="chkMAX"/>
    <OptionItem option="chkSTD"/>
    <OptionItem option="chkLCLM"/>
    <OptionItem option="chkUCLM"/>
    <OptionItem option="chkP25"/>
    <OptionItem option="chkP75"/>
  </Group>
</Container>
</UI>

```

DATA	OPTIONS	STATISTICS	INFORMATION
------	---------	-------------------	-------------

▲ Means STATISTICS

Select the statistics you would like included in the output dataset.

- Mean
- Minimum
- Maximum
- Standard Deviation
- LCLM
- UCLM
- 25th Percentile
- 75th Percentile

At the end of the Containers, the UI element is closed. Each container is a tab across the top on the UI. The above code contains three tabs; DATA, OPTIONS and STATISTICS.

DEPENDENCIES:

The Dependencies element specifies how certain roles rely on one another in order for the task to work properly. In the following code the Dependency element for Frequency has a condition when the Freq is select the variables var1 (Variable 1) and var2 (Variable 2) are disabled.

```
<Dependencies>
  <Dependency condition="$List.equalsIgnoreCase('Freq')">
    <Target action="disable" conditionResult="true" option="var2"/>
    <Target action="disable" conditionResult="true" option="var3"/>
  </Dependency>
  <Dependency condition="$List.equalsIgnoreCase('Print')">
    <Target action="enable" conditionResult="true" option="var2"/>
    <Target action="enable" conditionResult="true" option="var3"/>
  </Dependency>
  <Dependency condition="$List.equalsIgnoreCase('Means')">
    <Target action="disable" conditionResult="true" option="var2"/>
    <Target action="disable" conditionResult="true" option="var3"/>
    <Target action="enable" conditionResult="true" option="chkMEAN"/>
    <Target action="disable" conditionResult="false" option="chkMEAN"/>
    <Target action="enable" conditionResult="true" option="chkMIN"/>
    <Target action="disable" conditionResult="false" option="chkMIN"/>
    <Target action="enable" conditionResult="true" option="chkMAX"/>
    <Target action="disable" conditionResult="false" option="chkMAX"/>
    <Target action="enable" conditionResult="true" option="chkSTD"/>
    <Target action="disable" conditionResult="false" option="chkSTD"/>
    <Target action="enable" conditionResult="true" option="chkLCLM"/>
    <Target action="disable" conditionResult="false" option="chkLCLM"/>
    <Target action="enable" conditionResult="true" option="chkUCLM"/>
    <Target action="disable" conditionResult="false" option="chkUCLM"/>
    <Target action="enable" conditionResult="true" option="chkP25"/>
    <Target action="disable" conditionResult="false" option="chkP25"/>
    <Target action="enable" conditionResult="true" option="chkP75"/>
    <Target action="disable" conditionResult="false" option="chkP75"/>
    <Target action="enable" conditionResult="false" option="BASICOPTIONS"/>
    <Target action="disable" conditionResult="true" option="BASICOPTIONS"/>
  </Dependency>
  <Dependency condition="$var2.size() > 0">
    <Target action="enable" conditionResult="true" option="var3"/>
    <Target action="disable" conditionResult="false" option="var3"/>
  </Dependency>
  <Dependency condition="$ROWS2LIST.equalsIgnoreCase('firstrows')">
    <Target action="enable" conditionResult="true" option="NVALUE"/>
    <Target action="disable" conditionResult="false" option="NVALUE"/>
  </Dependency>
  <Dependency condition="$ROWS2LIST.equalsIgnoreCase('all')">
    <Target action="enable" conditionResult="false" option="NVALUE"/>
    <Target action="disable" conditionResult="true" option="NVALUE"/>
  </Dependency>
</Dependencies>
```

When the above code is executed, all the dependency conditions are applied. The variables Variable1 and Variable2 are disabled when Freq is selected.

DATA	OPTIONS	STATISTICS	INFORMATION
▲ DATASOURCE SASHELP.CARS			
▲ PROC GROUP Select from the List: Freq			
▲ FILTER 1 Variable 1: ↑ ↓ 🗑️ + <input type="text" value="Column"/>			
Variable 2: (1 item) 🗑️ + <input type="text" value="Column"/> Value: <input type="text" value="Select a value"/>			
▲ FILTER 2 Variable 3: (1 item) 🗑️ + <input type="text" value="Column"/> Value2: <input type="text" value="Select a value"/>			

REQUIREMENTS:

This element acts like the DEPENDENCY element. This element specifies a list of conditions that must be met in order for the task to run. If the condition is true, SAS code can be generated. If the condition is false, no code is generated. When defining a requirement, you can specify the message to display when the requirement is not met.

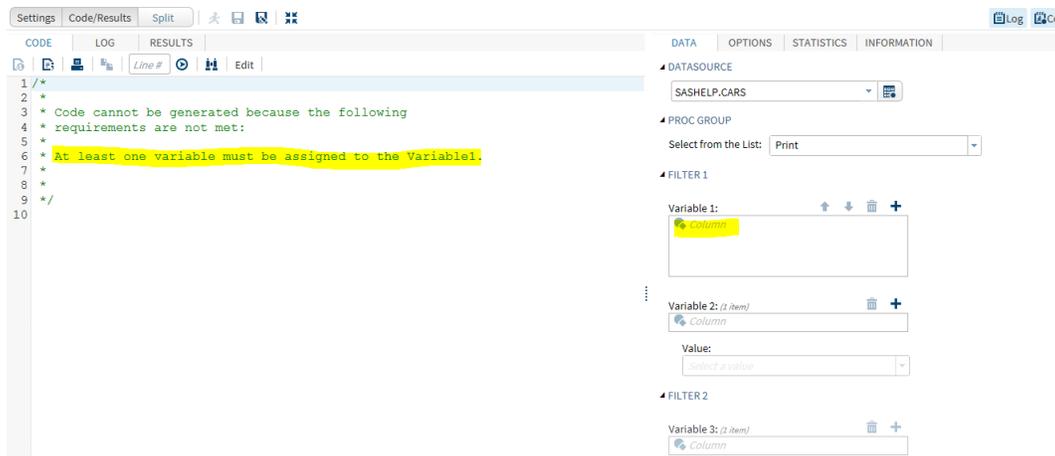
In the below code, the condition is (\$var1.size () > 0), which means that if the value of the var1 is greater than 0 then the condition is met and if the value is not great then 0, i.e.; no value is entered, the condition is not met and the message will be displayed.

```

<Requirements>
  <Requirement condition="$var1.size() &gt; 0">
    <Message>At least one variable must be assigned to the Variable1.</Message>
  </Requirement>
</Requirements>

```

Once the above code is executed the following screen is displayed. If you notice the highlighted areas, the alert "At least one variable must be assigned to the Variable1" is displayed because Variable1 does not have any columns selected.



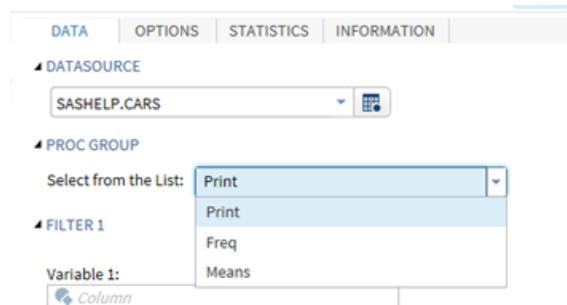
CODE TEMPLATE:

The code template creates the string output of the task. For most tasks, this output is SAS code. The code Template element contains a CDATA block from the Apache Velocity scripting language. A string output is produced using this scripting language.

Here is the code that we are using for PROC PRINT. The IF statement in the velocity code is used to choose the Print/Freq/Means options. The \$List is the velocity variable which gets assigned a selected value as a result of the option selected from the combo box. In this case, Print is selected from the choice of Print/Freq/Means.

```
<option defaultValue="Print" inputType="combobox"
name="$List" width="264px">Select from the List:</option>
```

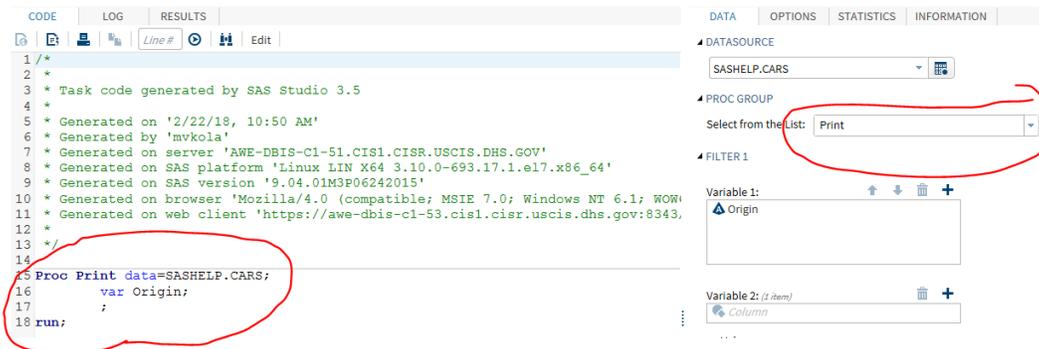
The UI has the option to choose from Print, Freq and Means, as seen below.



In the following Velocity code if the if statement has PRINT it will display SAS code for PRINT

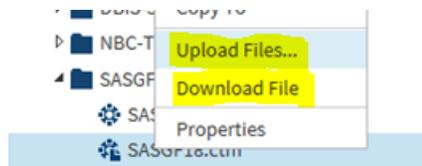
```
#if($List.equals("Print"))
  Proc Print data=$dataset #if($ROWS2LIST.equals("firstnrows")) (obs=$NVALUE) #end
    #if($ROWS2LIST.equals("all"))
      #end;
    #if( $var1.size() > 0 )
      var #foreach( $item in $var1 ) $item#end;
    #end
    #if( $var2.size() > 0 )
      Where #foreach( $item in $var2 ) $item#end = '$values1'
    #end
    #if( $var3.size() > 0 )
      AND #foreach( $item in $var3 ) $item#end = '$values2'
    #end;
  run;
#end
```

The following screen shows the SAS code on the left which is displayed when the Print option is selected in the Proc Group and code for Freq is displayed when the Freq is selected and same for the Means.



All the elements which are used in this current custom task are explained along with how they are used in building this custom task. The Custom Task is created and now the task can be saved as CTM or CTK files to be shared with the users by downloading from SAS Studio, emailed or placed in a share drive. The user will have to upload the files to SAS Studio in order to execute and produce the result.

The upload and download can be done by right clicking on the file and selecting Upload Files/Download File and then following the directions to select the path where the file needs to be saved.



CONCLUSION

All the elements which are used in the custom task building must be put together in order for the task to generate the application/UI Screen. Executing the task individually may not produce the desired results. The Registration is used for identity and the DATASOURCES and ROLES are used for SAS Data Source and variable selection. UI is used to display User Interface part of the task. CODE Template is used for writing the Velocity Code.

The entire CTM code is executed to produce the CTK output which is the User Interface. Once the UI is displayed the Data Set, variables etc. are selected. After the data selection the SAS code is generated on the code window. The SAS code needs to be submitted in order to produce output, which is done by clicking the submit button.

ACKNOWLEDGEMENT

I would like to recognize the following people and thank them for their contribution:

Girish Patel for helping with the paper concept, writing and co-presenting, Richard Bright for encouraging me to use SAS Studio and specifically SAS Custom Tasks and Nirav Patel in proof reading and editing and my contracting company SalientCRGT for supporting my travel and stay at Denver and my Prime contracting company Sevatec for support with the hours I spent at the conference.

REFERENCES

SAS Institute Inc. 2016 SAS® Studio 3.5: Developer's Guide to Writing Custom Tasks. Cary, NC: SAS Institute Inc.

CONTACT INFORMATION

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

Michael Kola

Salient CRGT

240-483-6963

Kola_Mail@yahoo.com

Michael.Kola@SailentCRGT.Com

Girish Patel

Jay Consultants, LLC

703-582-3025

g.patel@jayconsultants.com