Quickly and Easily Make Your SAS Programs Interactive with Macro QCKGUI

Vineet Jain, Smith Hanley Consulting Group, Jersey City, NJ

ABSTRACT
Often clinical programming teams create tools and utilities using BASE SAS Software. These programs are generally not interactive and are hard coded with parameters, which routinely need to be updated every time the program is run. When such programs are centrally located, programmers have to copy & modify the program (or write code to call the macro) in a local directory to use the program. This approach requires the user to understand the program and its parameters before actually using it. Making the program interactive relieves the users and allows dynamic interaction (i.e. data dependent) with the user. SAS provides multiple ways to make a program interactive, such as the window statement in data step and SAS AF. However, most programmers do not use these tools, as they feel that learning & using these for everyday tasks, such as selection of a directory path from the user, is not worth the time and effort.

This paper presents a frame catalog and macro QCKGUI to build simple custom interactive screens in minutes. This macro enables a programmer to bypass the learning & development phase of SAS AF/SCL code for such applications.

INTRODUCTION
SAS/AF allows programmers to create highly customized GUI (Graphical User Interface) applications using frame entries & accompanying SCL code, and has a wide variety of controls available, such as Text Box, buttons, Combo Box & Check Box. It is useful to create applications as simple as a message box to a complex GUI.

Message box, the simplest and most common form of GUI, can display messages such as errors, success, intended usage or status of program to user, while eliminating the need to dig through log files to see any important messages. Controls such as buttons, Text Boxes, and List Boxes are useful to take input, while relieving the user from manually setting parameters in a program. Though these applications are very helpful, there is no simple way to create even very basic form of GUI from BASE SAS software.

In this paper, the macro QCKGUI, with accompanying frame catalog, aims at providing a quick solution to these problems. It allows creating most common interfaces to get user feedback and display messages.

QCKGUI SYSTEM
Figure 1 shows the structure of SAS/AF based QCKGUI system.

To create an interface using QCKGUI system, base SAS program calls macro QCKGUI with control and attribute specifications defined in macro parameters. The macro QCKGUI uses the passed parameters to insert the GUI frame controls, set corresponding control attributes, create datasets QCKGUI_P & QCKGUI, and execute SAS AF application SCLFRAME.MAIN_F.FRAME. SCLFRAME is a FRAME CATALOG consisting of a pre-built frame, MAIN_F, with accompanying SCL code. MAIN_F has multiple copies of several disabled pre-defined controls. The
accompanying SCL code to MAIN_F frame, sets control attributes using records in QCKGUI_P dataset, creates items (for Radio Box, List Box & Combo Box) using QCKGUI dataset, and returns the user selections and text entries to QCKGUI dataset. The datasets are discussed later in detail.

**SETTING UP THE ENVIRONMENT**

To start using QCKGUI system, import the SCLFRAME catalog using CIMPORT procedure and then store the Macro QCKGUI & frame catalog SCLFRAME in a program library. Macro QCKGUI uses PROC DISPLAY to execute the SAS AF application. Replace the libref in the PROC DISPLAY procedure with the library name where frame catalog is stored.

```
PROC DISPLAY CATALOG=libref.SCLFRAME.MAIN_F.frame;
```

PROC DISPLAY is a base SAS procedure for executing pre-compiled frame entries. Since, it does not allow passing parameters to or returning parameters from SCL code, QCKGUI uses macro variables to pass values of control attributes from QCKGUI macro to frame SCL code.

The system demonstrated in this paper works fine for both windows based and UNIX based environment. However, the frame and its components may not look the same, when rendered across these environments. SAS runs the application successfully even when the operating environment does not support some of control’s attributes defined via QCKGUI macro. SAS prints a warning in the log file for such cases, and ignores the value assigned to the control attribute. In this paper, the presented GUI screens were captured from SAS ver.9.1 system running under MS Windows XP operating system.

The code for Macro QCKGUI is attached in Appendix. To get a copy of SCLFRAME catalog, contact at my email: vineet7878@gmail.com.

**SAMPLE APPLICATION**

The paper demonstrates the features and use of the presented macro by creating a sample graphical user interface shown in Figure 2. This relatively complex interface accepts options and text entry from user for an application where programmers need to upload SAS reports to a server. The following sections demonstrate the creation of the application piece by piece, while explaining the features and use of several controls available through QCKGUI.

**Figure 2: Sample Application: Reports Uploader Tool**

The QCKGUI Macro allows creation of seven types of controls in the interface - Text Label, Push Button, Text Entry, Check Box, Radio Box, List Box and Combo Box. Following sections explain the use and customization for each of these controls. The paper does not provide details for the use of frame and control attributes as it is out of scope. Please refer to SAS documentation to get further details on the use of specific controls & attributes.
GETTING STARTED WITH FRAME INTERFACE

The Frame Class is the foundation of the GUI and serves as the container for the controls available to the interface. When macro QCKGUI runs without any parameters, a small screen appears without any controls and just the frame with default attributes as shown in figure 3.A.

\%QCKGUI;

QCKGUI has four parameters to set the basic attributes for the frame - TITLE, BCKCOLOR, HEIGHT & WIDTH. These parameters set the title, background color, height and width for the frame, respectively. Figure 3.B shows the frame with user specified values for these parameters. To set additional attributes use MISC parameter. Its use is discussed in later sections.

\%QCKGUI (TITLE=This is my title, WIDTH=70, BCKCOLOR=red);

Figure 3: Empty Frames
3.A (with default attributes)           3.B (with customized attributes)

ADDITION & CUSTOMIZING CONTROLS

After getting a feel of the frame container, next step is to understand how to insert one or more controls to the interface and define corresponding attribute definitions. QCKGUI uses a parameter called CONTROL to specify the names of required controls for the GUI. It parses individual control names passed by CONTROL parameter using a split character (default '\') inserted between adjacent control names. Use same split character to separate multiple control attributes passed through other parameters. If needed, change the split character using the SPLIT parameter. Table 1 provides a brief summary of the parameters available in QCKGUI macro.

<table>
<thead>
<tr>
<th>To do This</th>
<th>Use this Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change Background Color of frame</td>
<td>BCKCOLOR</td>
</tr>
<tr>
<td>Insert data items for Radio Box, List Box or Combo Box</td>
<td>C_DATA</td>
</tr>
<tr>
<td>Specify Control Heights</td>
<td>C_HEIGHT</td>
</tr>
<tr>
<td>Change Control labels (only for applicable controls)</td>
<td>C_LABEL</td>
</tr>
<tr>
<td>Specify Control Locations – Horizontal</td>
<td>C_LOC_H</td>
</tr>
<tr>
<td>Specify Control Locations – Vertical</td>
<td>C_LOC_V</td>
</tr>
<tr>
<td>Specify Control Widths</td>
<td>C_WIDTH</td>
</tr>
<tr>
<td>Insert one or more controls to the frame</td>
<td>CONTROL</td>
</tr>
<tr>
<td>Specify frame Height</td>
<td>HEIGHT</td>
</tr>
<tr>
<td>Specify miscellaneous frame &amp; control attributes</td>
<td>MISC</td>
</tr>
<tr>
<td>Change Split Character (default = \”)</td>
<td>SPLIT</td>
</tr>
<tr>
<td>Specify Application Title</td>
<td>TITLE</td>
</tr>
<tr>
<td>Specify Frame Width</td>
<td>WIDTH</td>
</tr>
</tbody>
</table>

INSERTING FIRST CONTROL - TEXT LABEL

Text Label control creates un-editable text fields. In QCKGUI macro, Text Label controls are named as TL1, TL2 and so on... These controls (TL1, TL2...) are independent of each other and hence attributes of each control have to be assigned individually. The following code shows how to insert a single Text Label, with custom location, size and label to the interface. Figure 4 shows the GUI for the following code. Other editable attributes that can be set using MISC parameter (discussed later) are SizingUnits, BackgroundColor, BorderColor, BorderStyle, BorderWidth, TextOrientation, Justification & LabelColor.
Since all the copies of components are predefined in MAIN_F frame, the number of Text Label components for use in an interface is limited to 20, which is sufficient for most of the regular applications. Similarly, at most ten independent copies of each of other controls are available for the interface.

**ADDING MORE CONTROLS - PUSH BUTTON**

SAS AF allows programming complex operations using Push Button control, but to simplify the design, Push Button created using QCKGUI macro is allowed only to close the interface and pass the control back to QCKGUI macro. Similar to Text Labels, Push Buttons are named as PB1, PB2..., and PB10. The following code demonstrates how to use the Push Button control and how to insert multiple controls in a frame. Figure 5 shows the GUI created using the given code. QCKGUI system supports editing of several other attributes. For the complete list of supported attributes, refer to Table 4 in Appendix.

The above macro call requests multiple controls using split character, which is the default '\\' in this case. NOTE: if any of the basic control parameters - C_HEIGHT, C_WIDTH, C_LABEL, C_LOC_V, C_LOC_H and C_DATA - is assigned a non-missing value, then the number of attributes defined in the parameter, separated by split characters, should match with number of controls created in CONTROL parameter. This is to ensure correct match between attributes values and controls. Check the macro call in next example to understand the usage of split character when at least one of attribute values is missing for a parameter.

**SPECIFYING MISC ATTRIBUTES**

The following example introduces the Text Box (referred as Text Entry in SAS AF) control to design a regular user login screen and introduces the MISC parameter. Text Box control (or Text Entry Control) creates a text field to
accept text input from the user. It follows the naming convention similar to previous controls, i.e. TB1, TB2..., and TB10.

The MISC parameter defines the attributes other than those defined by basic control parameters - C_HEIGHT, C_WIDTH, C_LABEL, C_LOC_V, C_LOC_H and C_DATA. To assign a value to a control attribute use the syntax - ControlName.AttributeName=AttributeValue - in MISC parameter. It can define any number of control attributes separated by split character ". In the following example, the MISC parameter does two attribute assignments – 'TB2.maskcharacter=*' and 'TL1.labelcolor=white'. Table 4 in Appendix provides complete list of attributes supported by QCKGUI. Figure 6 shows the GUI created for following code.

```%QCKGUI( CONTROL = TL1\TB1\TL2\TB2, 
C_HEIGHT = 30\30\30\30, 
C_WIDTH = 150\150\150\150, 
C_LABEL = Username\Password, 
C_LOC_V = 50\220\50\220, 
C_LOC_H = 50\100\100, 
MISC = TB2.maskcharacter=\TL1.labelcolor=white 
);```

Figure 6: Using MISC parameter to assign Miscellaneous Attributes

USING SELECTION CONTROLS – CHECK BOX, RADIO BOX, COMBO BOX & LIST BOX
QCKGUI supports four controls for selection of one or more items – Check Box to create a switch (yes/no), Radio Box & Combo Box to create controls to select one among many choices, and List Box to create a control allowing multiple selections. These four controls have naming convention similar to previous controls, Check Box – CB1, CB2..., and CB10, Radio Box – RB1, RB2..., and RB10, Combo Box – CM1, CM2..., and CM10 and List Box – LB1, LB2..., and LB10.

To populate these controls (except Check Box control) with a list of items use C_DATA parameter. C_DATA parameter's use is similar to other basic control parameters like C_LABEL. It allows specifying the items in two different ways – using the records in a dataset or entering the list of items in parameter C_DATA. To enter the items within the parameter use the keyword 'ITEMS' in C_DATA parameter followed by list the items, e.g. ITEMS selection1 selection2 ... To enter the items using a dataset use the keyword 'DATA' instead of 'ITEMS' followed by dataset name, e.g. C_DATA=DATA 'itemsource'. The item source dataset should consist of a variable, called as 'DATA', containing the items as shown in DATA step of following code. Since Check Box control only has one possible selection, use C_LABEL parameter to enter the text for selection.

The following example demonstrates the use of Check Box control, and C_DATA parameter for multi-item selection control with both possible keywords – ITEMS or DATA. Figure 7 shows the GUI created using the following code.

```DATA itemset;
data='alice@companyname.com  '; output;
data='tom@companyname.com    '; output;
data='mary@companyname.com   '; output;
data='lisa@companyname.com   '; output;
data='mona@companyname.com   '; output;
data='victor@companyname.com '; output;
data='john@companyname.com   '; output;
run;

%QCKGUI( CONTROL = CB1\RB1\LB1, 
C_HEIGHT = 30\150\150, 
C_WIDTH = 250\150\150,```
The Macro call below combines the previously defined controls (with some modifications) and adds some more control elements to create the GUI presented in Figure 2. It shows how easy it is to create even a relatively complex GUI with QCKGUI.

```
% QCKGUI(HEIGHT = 35,
WIDTH = 65,
TITLE = REPORTS UPLOADER TOOL,
CONTROL = TL4\TL1\TL2\TB1\TB2\CB1\CB2\RB1\LB1\TL5\TB3\PB1\PB2\CM1\TL6,
C_HEIGHT = 30\30\30\30\30\30\30\30\30\150\150\150\150\30\30\30\30\24\30,
C_WIDTH = 250\250\100\100\100\100\220\220\200\200\100\100\100\100\150\150,
C_LABEL = UPLOAD REPORTS TO THE WEB\Enter the Web Server login Details\Backup previous set of reports\Delete Previous reports\Location of reports\UPLOAD\CANCEL\Label Report,
C_LOC_V = 20\60\100\100\100\100\100\100\250\250\200\200\430\430\60\60,
C_LOC_H = 180\50\30\150\30\150\350\350\50\320\30\140\200\200\340\440\340,
C_DATA =\\\items REPORTS LISTINGS BOTH\data itemset\items FileName TITLE1 TITLE2 TITLE3 TITLE4 ,
MISC = LB1.title=Email To(Select one or more)\RB1.bordertitle=Upload (Select One)\TB2.maskcharacter=*
);
```

**CHECKING SUCCESSFULLY PARSED ATTRIBUTES**

QCKGUI macro creates a dataset called QCKGUI_P consisting of a record for each successfully parsed attribute for frame and controls. This dataset is very helpful to debug errors in QCKGUI parameters while creating the GUI interface.

QCKGUI_P dataset consists of three variables – CONTROL, PARAM & VALUE, where CONTROL variable has the control name, PARAM variable has the attribute name and VALUE variable has the attribute value assigned in the QCKGUI macro call.

Note: MAIN_F_SCL code assigns only the ‘supported’ attributes with valid values passed through the macro. Table 2 on next page, shows the partial set of records for the QCKGUI_P dataset, generated by macro for sample application.

**USER RESPONSE, SELECTION AND TEXT ENTRY**

QCKGUI macro also creates a dataset called QCKGUI, which consists of records for user response for each possible user selection in the interface. This dataset is useful to know the clicked button, entered text in the Text Boxes, and selection in Check Boxes, Radio Boxes, List Boxes and Combo Boxes, by the user.

QCKGUI dataset consists of five variables – CNT, TYPE, DATA, SELECTED & TEXT, where CNT has the control name, TYPE has the control type (e.g. Check Box, List Box etc), DATA has the items for selection for Combo Box,
List Box and Radio Box controls, SELECTED has user response (1 if selected, 0 or missing if not selected) and TEXT has text entered by user in Text Boxes. This dataset has an additional record with CNT value as PB0 and TYPE value as ‘no-button’. The value of SELECTED variable for this record is set to 1 if none of the Push Buttons were used to close the application. For example, when user closes the application using application close button at the top right of interface window, the value of SELECTED variable is set to 1. Table 3 shows the records in the QCKGUI dataset generated by the last QCGUI macro call for user selection and entry in Figure 2.

<table>
<thead>
<tr>
<th>param</th>
<th>control</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>cb1</td>
<td>30</td>
</tr>
<tr>
<td>horizontalposition</td>
<td>cb1</td>
<td>350</td>
</tr>
<tr>
<td>label</td>
<td>cb1</td>
<td>Backup previous set of reports</td>
</tr>
<tr>
<td>verticalposition</td>
<td>cb1</td>
<td>100</td>
</tr>
<tr>
<td>width</td>
<td>cb1</td>
<td>220</td>
</tr>
<tr>
<td>height</td>
<td>cb2</td>
<td>30</td>
</tr>
<tr>
<td>horizontalposition</td>
<td>cb2</td>
<td>350</td>
</tr>
<tr>
<td>label</td>
<td>cb2</td>
<td>Delete Previous reports</td>
</tr>
<tr>
<td>verticalposition</td>
<td>cb2</td>
<td>140</td>
</tr>
<tr>
<td>width</td>
<td>cb2</td>
<td>220</td>
</tr>
<tr>
<td>height</td>
<td>cm1</td>
<td>24</td>
</tr>
<tr>
<td>horizontalposition</td>
<td>cm1</td>
<td>440</td>
</tr>
<tr>
<td>Label</td>
<td>cm1</td>
<td></td>
</tr>
<tr>
<td>Verticalposition</td>
<td>cm1</td>
<td>60</td>
</tr>
<tr>
<td>Width</td>
<td>cm1</td>
<td>150</td>
</tr>
<tr>
<td>Backgroundcolor</td>
<td>frm</td>
<td>Gray</td>
</tr>
<tr>
<td>Height</td>
<td>frm</td>
<td>35</td>
</tr>
<tr>
<td>Title</td>
<td>frm</td>
<td>REPORTS UPLOADER TOOL</td>
</tr>
<tr>
<td>Width</td>
<td>frm</td>
<td>65</td>
</tr>
</tbody>
</table>

**TABLE 3: QCKGUI dataset for user selection and entry in Figure 2**

<table>
<thead>
<tr>
<th>cnt</th>
<th>type</th>
<th>Data</th>
<th>Selected</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>cb1</td>
<td>Check Box</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>cb2</td>
<td>Check Box</td>
<td></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>cm1</td>
<td>Combo Box</td>
<td>FileName</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>cm1</td>
<td>Combo Box</td>
<td>TITLE1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>cm1</td>
<td>Combo Box</td>
<td>TITLE2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>cm1</td>
<td>Combo Box</td>
<td>TITLE3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>cm1</td>
<td>Combo Box</td>
<td>TITLE4</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>lb1</td>
<td>List Box</td>
<td><a href="mailto:alice@companyname.com">alice@companyname.com</a></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>lb1</td>
<td>List Box</td>
<td><a href="mailto:tom@companyname.com">tom@companyname.com</a></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>lb1</td>
<td>List Box</td>
<td><a href="mailto:mary@companyname.com">mary@companyname.com</a></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>lb1</td>
<td>List Box</td>
<td><a href="mailto:lisa@companyname.com">lisa@companyname.com</a></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>lb1</td>
<td>List Box</td>
<td><a href="mailto:mona@companyname.com">mona@companyname.com</a></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>lb1</td>
<td>List Box</td>
<td><a href="mailto:victor@companyname.com">victor@companyname.com</a></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>lb1</td>
<td>List Box</td>
<td><a href="mailto:john@companyname.com">john@companyname.com</a></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>pb0</td>
<td>no-button</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>pb1</td>
<td>Push Button</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>pb2</td>
<td>Push Button</td>
<td></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>rb1</td>
<td>Radio Box</td>
<td>REPORTS</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>rb1</td>
<td>Radio Box</td>
<td>LISTINGS</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rb1</td>
<td>Radio Box</td>
<td>BOTH</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>tb1</td>
<td>Text Box</td>
<td>/drugname/protname/reports</td>
<td>Vineetjain</td>
<td></td>
</tr>
<tr>
<td>tb2</td>
<td>Text Box</td>
<td>Mypassword</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tb3</td>
<td>Text Box</td>
<td>/drugname/protname/reports</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tl1</td>
<td>Text Label</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tl2</td>
<td>Text Label</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tl3</td>
<td>Text Label</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tl4</td>
<td>Text Label</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tl5</td>
<td>Text Label</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tl6</td>
<td>Text Label</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**CONCLUSION**

Macro QCKGUI provides a basic set of visual controls making simple GUI development quick and easy for Base SAS programmers. This greatly enhances the usability of utilities developed using base SAS software.
ACKNOWLEDGMENTS
Many thanks are extended to my wife Shweta Jain, who provided exceptional coding and writing guidance for this paper, to Shridhar Patel for his critical reading of this paper and to Priyanka Jain for editing assistance of this paper.

RECOMMENDED READING
In the SAS Online Documentation: SAS AF/SCL

CONTACT INFORMATION
Your comments and questions are valued and encouraged. Contact the author at:
Vineet Jain
Statistical Programmer/Analyst
45 W 3rd St. #1R
South Orange, NJ 07079
Phone: 973-821-5008
Email: vineet7878@gmail.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.

APPENDIX

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Frame</th>
<th>Check Box</th>
<th>Combo Box</th>
<th>List Box</th>
<th>Push Button</th>
<th>Radio Box</th>
<th>Text Entry Box</th>
<th>Text Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>BackgroundColor</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>BorderAttribute</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BorderColor</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>BorderStyle</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>BorderRadius</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ButtonStyle</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ComboBoxStyle</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ConformSize</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ConfirmOnExit</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CursorPlacement</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CursorPosition</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DefaultPushButton</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Editable</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enabled</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ForegroundColor</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ForegroundColor</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Height</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HorizontalPosition</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Justification</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Label</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LabelColor</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>MaskCharacter</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MaximumCharacters</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Orientation</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selected</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SelectedIndex</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>SelectionMode</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SizingUnits</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TextColor</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TextCompletion</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TextOrientation</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Title</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UpperCase</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Visible</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>VerticalPosition</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Width</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
MACRO QCKGUI SAS CODE
** Format to create macro variables names for control attributes **;
proc format;
    value $qckprm
        'backgroundcolor'        = 'bkc' 'borderattribute'        = 'boa'
        'bordercolor'            = 'boc' 'borderstyle'            = 'bos'
        'bordertitle'            = 'bot' 'borderwidth'            = 'bow'
        'buttonstyle'            = 'bts' 'comboboxstyle'          = 'cms'
        'confirmonexit'          = 'cfe' 'conformsize'            = 'cs'
        'cursorplacement'        = 'cp' 'cursorposition'         = 'cps'
        'defaultpushbutton'      = 'dpb' 'editable'               = 'ed'
        'enabled'                = 'enb' 'foregroundattribute'    = 'fra'
        'foregroundcolor'        = 'frc' 'height'                 = 'hei'
        'horizontalposition'     = 'hor' 'justification'          = 'jus'
        'label'                  = 'lb' 'labelcolor'             = 'lbc'
        'maskcharacter'          = 'ms' 'maximumcharacters'      = 'max'
        'orientation'            = 'or' 'selected'               = 'sl'
        'selectedindex'          = 'sli' 'selectionmode'          = 'slm'
        'sizingunits'            = 'su' 'text'                   = 'txt'
        'textcolor'              = 'txc' 'textcompletion'         = 'tc'
        'textorientation'        = 'to' 'title'                  = 'tit'
        'uppercase'              = 'upp' 'verticalposition'       = 'ver'
        'visible'                = 'vis' 'width'                  = 'wid'
        other                    = ' '; 
%macro qckgui1(string=);
    cnt_str = &string;
    if cnt_str ne ' ' then do;
        str_st = 1;
        if param_n = 'control' then cnt_str = lowcase(cnt_str);
        str_end = length(cnt_str);
        if count(cnt_str,&split) + 1 = &counter then do;
            %do i = 1 %to &counter;
                text&i = ''; 
                y = find(cnt_str,&split,str_st);
                if y = 0 and str_st <= str_end then text&i = substr(cnt_str,str_st,str_end
                                                                                       -str_st+1);
                else if y = 0 and str_st > str_end then text&i = ''; 
                else if y ne str_st then text&i = substr(cnt_str,str_st,find(cnt_str,
                                                                                   &split,str_st)-str_st );
                else text&i = ''; 
                str_st = find(cnt_str,&split,str_st) + 1;
            if strip(param_n) = 'control' then do;
                call symput('cnt_n&i',text&i);
                call symput('cnt_t&i',substr(text&i,1,2));
            end;
        end;
        else call symput('paramerr','yes'); 
    end;
%mend qckgui1;

**macro to parse control parameters i.e. control, c_label ... into separate strings**;
%macro qckgui(title=SAS AF application, bckcolor=gray, width=40, height=15, control=, c_label=,
        c_loc_v=, c_loc_h=, c_height=, c_width=, c_data=, misc=, split='\');
    %macro qckgui1(string=);
    cnt_str = &string;
    if cnt_str ne ' ' then do;
        str_st = 1;
        if param_n = 'control' then cnt_str = lowcase(cnt_str);
        str_end = length(cnt_str);
        if count(cnt_str,&split) + 1 = &counter then do;
            %do i = 1 %to &counter;
                text&i = ''; 
                y = find(cnt_str,&split,str_st);
                if y = 0 and str_st <= str_end then text&i = substr(cnt_str,str_st,str_end
                                                                                       -str_st+1);
                else if y = 0 and str_st > str_end then text&i = ''; 
                else if y ne str_st then text&i = substr(cnt_str,str_st,find(cnt_str,
                                                                                   &split,str_st)-str_st );
                else text&i = ''; 
                str_st = find(cnt_str,&split,str_st) + 1;
            if strip(param_n) = 'control' then do;
                call symput('cnt_n&i',text&i);
                call symput('cnt_t&i',substr(text&i,1,2));
            end;
        end;
        else call symput('paramerr','yes'); 
    end;
%mend qckgui1;

** Count the number of Components from control parameter **;
data _null_; 
    call symput('counter',count('&control',&split) + 1); 
run;
** Macro variable to track parsing errors in qckgui macro, It is initially set to NO **;
%let paramerr = no;

** Dataset qckgui_p created to store component params by parsing params Control and C_**;
data qckgui_p(keep=param_n text:);
 length cnt_str $500. %do i = 1 %to &counter; text&i $200. %end; ;
 param_n = 'control '; %qckgui1(string="&control"); output;
 param_n = 'c_label '; %qckgui1(string="&c_label"); output;
 param_n = 'c_loc_v '; %qckgui1(string="&c_loc_v"); output;
 param_n = 'c_loc_h '; %qckgui1(string="&c_loc_h"); output;
 param_n = 'c_height'; %qckgui1(string="&c_height"); output;
 param_n = 'c_width'; %qckgui1(string="&c_width"); output;
 param_n = 'c_data '; %qckgui1(string="&c_data"); output;

proc transpose data=qckgui_p out=qcktmp(drop=_name_); var text:; id param_n;

data qckgui_p;
 length param $30.;
 set qcktmp end=lastone;
 if control ne ' ' then do;
 call symput (control,1);
 array c_{*} c_:;
 do i = 1 to dim(c_);
 select(i);
 when(1) param = 'label';
 when(2) param = 'verticalposition';
 when(3) param = 'horizontalposition';
 when(4) param = 'height';
 when(5) param = 'width';
 otherwise param = ' '; end;
 value = c_(i);
 if param ne ' ' then output;
 end; end;
 if lastone = 1 then do;
 control = 'frm';
 if "&title" ne " " then do; param = 'title'; value = "&title"; output; end;
 if "&bckcolor" ne " " then do; param = 'backgroundcolor'; value = "&bckcolor";
 output; end;
 if "&width" ne " " then do; param = 'width'; value = "&width"; output; end;
 if "&height" ne " " then do; param = 'height'; value = "&height"; output; end;
 end;

proc sort data=qckgui_p; by control param; where control ne ' '; run;

**If &misc param is not empty then parse the string passed to misc parameter **;
%if "&misc" ne %then %do;
data misc(keep=control param value text);
 length text control value $200. param $30.;
 misc = "&misc";
 count = count(misc,&split) + 1;
 str_st = 1; str_end = length(misc);
 do i = 1 to count;
 y = find(misc,&split,str_st);
 if y = 0 and str_st <= str_end then text = substr(misc,str_st,str_end- str_st+1);
 else if y = 0 and str_st > str_end then text = '';
 else if y ne str_st then text = substr(misc,str_st,y-str_st );
 else text = '';
 str_st = y + 1;
i_dot = index(text,'. ');
i_eql = index(text,'=');
control = ' '; param = ' '; value = ' ';
if text ne ' ' and i_dot > 0 and i_eql > i_dot then do;
control = lowcase(strip(substr(text,1,i_dot-1)));
if i_eql-i_dot >1 then param=strip(substr(text,i_dot+1,i_eql-i_dot-1));
if length(text) >i_eql then value=strip(substr(text,i_eql+1, length(text)-i_eql));
if control ne ' ' and param ne ' ' then output;
end;end;

proc sort data=misc; by control param;

proc sort data=qckt tmp; by control;

data misc;
merge qckt tmp(in=a) misc(in=b);
by control;
if (a and b) or control = 'frm';

data qckgui_p; set qckgui_p misc; by control param;
%end;

** Merge the component attributes obtained from misc parameter into qckgui_p **;
** Create a macro variable for each successfully resolved component attribute **;
data qckgui_p(keep=control param value);
set qckgui_p;
param = lowcase(param);
param_f = put(param,$qckprm.);
if param_f ne ' ';
call symput (compress(control || '_ '|| param_f),strip(value));
run;

** Parameter parsing error reported, If any **;
%if &paramerr = yes %then %do;
%put ERROR: No. of elements in one of parameters is different than no. of control elements;
%end;

** Create dataset qckgui that passes and retrieves the user interaction **;
** It has component name, type, component items, user selection and text entry **;
data qckgui;
length cnt $10. type $20. data $200. selected 8. text $200.;
cnt = 'pb0'; type = 'no-button'; output;
%if &counter ne 1 %then %do;
%do i = 1 %to &counter;
cnt = "&&cnt_n&i';
select(substr(cnt,1,2));
when('tl') type = 'Text Label';
when('cb') type = 'Check Box';
when('lb') type = 'List Box';
when('cm') type = 'Combo Box';
when('rb') type = 'Radio Box';
when('tb') type = 'Text Box';
when('pb') type = 'Push Button';
otherwise type = 'Unknown Component';
end;
output;
%end;%end;

proc sort data=qckgui; by cnt;
run;
** Read a qualified dataset/item list for radiobox, combobox or listbox **;
%if &counter > 1 %then %do;
  %do i = 1 %to &counter;
  %if (&&cnt_t&i = rb or &&cnt_t&i = cm or &&cnt_t&i = lb) and "&&cnt_d&i" ne %then %do;
    ** Read items list if ITEMS is specified in c_data parameter **;
    %if %upcase("%scan(&&cnt_d&i,1)") = "ITEMS" and %upcase("%scan(&&cnt_d&i, 2)") ne %then %do;
      data temp;
      length cnt $10.;
      length data $200.;
      x = "&&cnt_d&i";
      cnt = "&&cnt_n&i";
      j = 2;
      do while(scan(x,j) ne ' ');
        data = scan(x,j);
        j + 1;
        output;
      end;
      data qckgui(drop=x j); merge qckgui temp; by cnt;
    %end;
  %end;
%end; %end; %end; %end;

** Read variable data from data set associated with DATA is the c_data **;
%if %upcase("%scan(&&cnt_d&i,1)") = "DATA" and %upcase("%scan(&&cnt_d&i,2)") ne %then %do;
  data temp;
  length cnt $10.;
  length data $200.;
  set %scan(&&cnt_d&i,2);
  cnt = "&&cnt_n&i";
  data qckgui; merge qckgui temp; by cnt;
%end; %end; %end; %end;

*** Display the frame from pre-compiled catalog entry ***;
proc display catalog=home.sclframe.main_f.frame; run;

** Check the AF application exit method **;
%let butpres = no;
data temp;
  set qckgui;
  where substr(cnt,1,2) = 'pb' and selected = 1;
  call symput('butpres','yes');
run;

** If no button was pressed then set pb0 selected **;
%if &butpres = no %then %do;
data qckgui;
  set qckgui;
  if cnt = 'pb0' then selected = 1;
%end;

** Clear temporary datasets **;
proc datasets nolist;
  delete misc temp qcktmp;
quit;
run;
%mend qckgui;

12