ABSTRACT
The FRAME entry is a new catalog entry available in SAS/AF® software. This entry enables you to develop graphical user interface applications using object-oriented programming (OOP) techniques. In this first of four tutorials, you will learn about the set of supplied GUI building blocks (widgets), how to place these widgets in a FRAME entry, and how to construct a simple menu-driven application. Additionally, you will learn how to write simple Screen Control Language (SCL) programs to control execution of the application.

INTRODUCTION
Like PROGRAM entries, FRAME entries enable you to design, build, and execute interactive SAS/AF applications using SCL. Additionally, with FRAME entries you can
- quickly prototype an application using the interactive development environment
- develop applications that incorporate graphical user interface (GUI) building blocks such as icons, graphic text, SAS/GRAPH® output, push buttons, and control arrows as well as simple text
- employ object-oriented programming techniques that encourage developing reusable software modules.

A typical FRAME application incorporating graphics and icons is shown in Figure 1. The FRAME is designed to serve as a menu — users can click on the REPORTS icon or GRAPHS icon to branch to other FRAME entries.

Table 1 Comparison of FRAME Entries and PROGRAM Entries

<table>
<thead>
<tr>
<th>Feature</th>
<th>PROGRAM Entries</th>
<th>FRAME Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISPLAY Editor</td>
<td>SAS Text Editor</td>
<td>Region Manager</td>
</tr>
<tr>
<td>DISPLAY Items</td>
<td>text &amp; fields</td>
<td>widgets only</td>
</tr>
<tr>
<td>SCL program</td>
<td>stored internally</td>
<td>external SCL entry</td>
</tr>
</tbody>
</table>

CREATING A FRAME ENTRY
You create a FRAME entry much like any other SAS/AF catalog entry:

```
proc build c=sugi.tutorial.menu.frame;
run;
```

An alternative is to use the Build window and its pull down menus:

![Figure 2 Creating a New FRAME Entry](image)

Figure 2 Creating a New FRAME Entry

WORKING WITH REGION MANAGER
The new FRAME entry is displayed with an empty master region. You cannot type anything in this area. You create rectangular regions and fill each of them with a widget. A widget is a displayable object.

Figure 3 shows the design we are attempting to create. This entry contains six regions containing one SAS/GRAPH® Output widget, two Graphics Text widgets, and three Icon widgets.

COMPONENTS OF A FRAME ENTRY
If you are familiar with SAS/AF PROGRAM entries, it will be useful to note that both FRAME and PROGRAM entries
- are created with PROC BUILD
- have general attributes as well as item specific attributes
- are programmed with Screen Control Language
- can have supporting HELP, KEY, CBT, and PMENU entries.
You manipulate regions and objects by issuing commands. Although commands can be issued from a command line or with function keys, you will probably find it easier to use a mouse to activate pull-down menus and context sensitive pop-up menus.

To create a region with a mouse, use the rubberband placement technique: click the mouse select button to establish the upper left hand corner of a rectangle, and then drag the mouse pointer to the diagonally opposite corner. When the region is the desired size and shape, release the mouse button.

After creating a region, you can perform basic region operations like:
- move
- resize
- copy
- remove
- set region attributes
- fill with an object
- remove the object (empty)
- set object attributes.

To perform a region operation, you must first make the region active by clicking the mouse anywhere within or on the region border. The active region border has eight handles for resizing. You can click on a handle and drag it to resize a region. To move a region, click and drag any other part of the region border.

Begin creating the FRAME entry shown in Figure 3 by creating a region for the graphic text SUGI 18.

FILLING A REGION WITH A WIDGET

You create an object by filling the active region with one of eighteen supplied widget types. The FILL operation presents you with a widget selection list known as a resource list. An alternative to filling a region is to MAKE an object. This involves choosing a widget from the resource list and letting Region Manager automatically create a region of some default size to hold the widget.

To fill a region with a widget, you must first make the region active by clicking the mouse anywhere within or on the region border. The active region border has eight handles for resizing. You can click on a handle and drag it to resize a region. To move a region, click and drag any other part of the region border.

Begin creating the FRAME entry shown in Figure 3 by creating a region for the graphic text SUGI 18.

SETTING WIDGET ATTRIBUTES

After selecting Graphic Text, you are placed in the attribute window for this object (Figure 6).

The name of the object is automatically of the form OBJn, based on how many objects already exist in this FRAME entry. You can change this name to something more meaningful. The other attributes specify desired text, font, color, and aspect ratio for the object.

You can use a copy operation on the first object to create the second Graphic Text object. You only need to modify its text attribute to "Frame Tutorial."

The Statue of Liberty graphic was drawn with the Graphics Editor (part of SAS/GRAPH software). This picture, like other SAS/GRAPH output, is stored in a catalog entry of type GRSEG. When you create a SAS/GRAPH output object in a frame, you can specify the name of the GRSEG catalog entry to be displayed in the object's region (Figure 7). You will need to experiment with the region shape to get the desired aspect ratio.
Next, we will create some Icon objects (Figure 8).

The attributes for an Icon Widget are shown in Figure 9.

By clicking on the Current Icon, you get a selection list of all supplied icons.

After locating the desired icon, click on it. It becomes the Current Icon. Change the Label value to whatever text you want to appear by the icon. After confirming the Icon attributes, you can copy the icon named REPORT and modify its attributes to create an icon named GRAPH and another Icon named EXIT.

ATTRIBUTES SHARED BY ALL WIDGETS

All widget types have several common attributes. These appear in the Additional Attributes area. Command Processing allows you to define a command string to be issued when the widget is activated during execution. Figure 11 shows assigning the CANCEL command to the EXIT Icon.

Every FRAME entry has General Attributes. Figure 12 shows the Title attribute set to Main Menu. Notice the SCL entry attribute. It indicates that there will be an entry named MENU.SCL associated with this FRAME entry. FRAME entries can share a common SCL program or not use an SCL program. By default, an SCL entry of the same name as the FRAME entry is used.
PROGRAMMING A FRAME ENTRY

As with PROGRAM entries, the SOURCE command enables you to edit the SCL program. Figure 13 shows the complete SCL program for MENU.FRAME.

Figure 13 MENU.SCL Entry for MENU.FRAME

Nearly everything you know about SCL coding for PROGRAM entries can be used in developing SCL programs for FRAME Entries. Every widget in the master region automatically has two associated SCL features:

- **SCL Variable**: The name of the widget is the SCL variable name. The value of the SCL variable depends on the type of widget. For SAS/GRAPH output widgets, the SCL variable contains the name of the stored GRSEG catalog entry. For Icon widgets, the SCL variable contains the number or text assigned to the Value on Selection attribute.

- **SCL Label**: The name of the widget can be used as an SCL label (like INIT, MAIN, or TERM). The SCL label serves as an entry point to a SCL code segment that executes only when the specified widget is selected during execution. This reduces the amount of code that needs to be placed in MAIN. (If you include a MAIN section, it executes after all widget-labeled sections.) Field-labeled sections are also available in the SCL for PROGRAM entries, if you specify the LABEL option on a CONTROL statement.

As with PROGRAM entries, you must compile the SCL source program. After compilation, you can execute the FRAME with the commands TESTAF, AF, or AFA. The SCL debugger is also available for tracing execution.

CREATING ANOTHER FRAME ENTRY

The Icon widget named GRAPH calls another FRAME entry named GRAPH1.FRAME in the current catalog. Figure 14 shows the design of this frame. It uses a List Box widget named LISTBOX to handle user selection of a graph to be displayed. The large region to the right contains a SAS/GRAPH output widget named GRAPH. The GRAPH widget does not specify an initial graph to be displayed. The graph will be determined by the item selected in LISTBOX during FRAME execution.

Figure 14 GRAPH1.FRAME in BUILD Mode

The attributes for this List Box widget are shown in Figure 15. You indicate the selection items that appear in a List Box by specifying:

- the actual items
- an SCL text string
- an SCL array
- an SCL list
- a variable from a SAS data set
- a variable from an open SAS data set
- a catalog LIST entry.

As our example, we want the List Box to contain the current names and descriptions of all GRSEG entries in a given SAS catalog. To accomplish this, we have already created a SAS data set named SUGI.GVIEW. The source program for creating this VIEW file follows.
proc sql;
cREATE VIEW SUGI.GVIEW AS
SELECT objname1, lobjdesc AS descrip
FROM dictionary.catalogs
WHERE libname='SUGI' AND memname='GRAPHS'
AND objtype='GRSEG';

SUGI.GVIEW and the DESCRIP variable are specified the List Box Population attribute for SAS data set. The advantage of using this approach is that the contents of LISTBOX are determined dynamically.

Figure 16 List Box Population — SAS Data Set
The FRAME's complete SCL program, GRAPH1.SCL, is shown below.

This program uses a new SCL feature supported only in FRAME entries: the NOTIFY routine. In this program, the NOTIFY routine executes a program called _GET_LAST_SEL_ on the LISTBOX widget and passes three parameter variables to this program. The program examines the state of the widget and returns values to these user-defined SCL variables:

<table>
<thead>
<tr>
<th>ROW</th>
<th>ISSEL</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>the row number of the most recently selected row</td>
<td>a numeric value indicating whether the item is still selected</td>
<td>the text for the returned row.</td>
</tr>
</tbody>
</table>

Every widget has access to several built-in programs or methods specific to its type. In OOP terminology, objects send messages to objects via methods. When activated, the LISTBOX object sends the _GET_LAST_SEL_ message to itself. This message returns the text selected by the user to the SCL variable NAME. (The other two variables are required as "place holders" for coding purposes, but the values returned to them are not needed by this application.)

The NAME variable is used to construct a four-level catalog entry name which is then assigned to the GRAPH widget. When control returns to the FRAME entry, the GRAPH widget is refreshed and the selected graph appears.

We could have asked the LISTBOX widget to send a message to the GRAPH widget:

LISTBOX:
call notify('listbox', _GET_LAST_SEL_ , row, issel, name);
call notify('graph', _SET_GRAPH_ , SUGI.GRAPHS. ||trim(name)|| .GRSEG');
return;

The _SET_GRAPH_ method is recognized only by SAS/GRAPH output widgets.

Figure 17 shows GRAPH1.FRAME during execution after the user has activated LISTBOX.

Figure 17 GRAPH1.FRAME in Execution Mode

SUMMARY
This tutorial has demonstrated how quickly Graphical User Interfaces can be developed using the SAS/AF FRAME entry. Along the way, you have been introduced to basic concepts and terminology related to Object Oriented programming in a FRAME environment. The next tutorial explores additional FRAME features that permit different styles of user interaction.

REFERENCES
SAS/AF® Software: FRAME Entry Usage and Reference, Version 6, First Edition
SAS, SAS/AF, and SAS/GRAPH are registered trademarks of SAS Institute Inc. in the USA and other countries.® indicates USA registration.