

The Basics of AF/Frame and Screen Control Language

Author: Dana Rafiee, Instructor/Consultant -- Destiny Corporation / DDISC Group, Ltd.

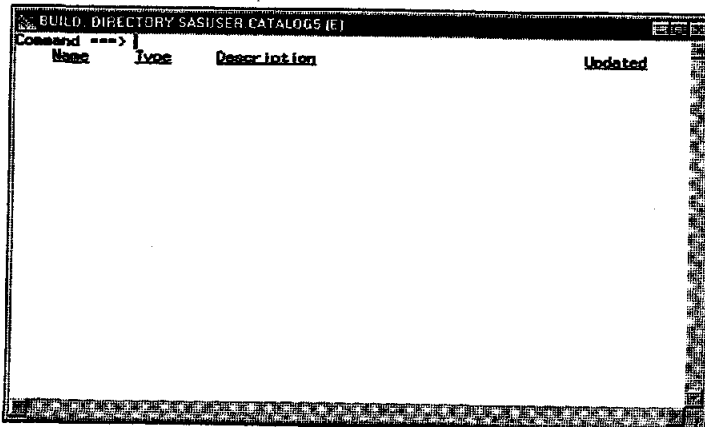
Preface

The use of Frame Entry is the graphical (GUI) front end developed for creating applications using SAS/AF® software. SAS/AF Software has existed for several years and versions, but its screens were character based. Frame Entries were established in version 6.08 under Windowing environments and have been continually enhanced ever since. The basis of this paper assumes you understand Screen Control Language and its uses in application development. This paper will be delivered in tutorial style. The presentation's basic demonstration of creating a simple frame is seen here.

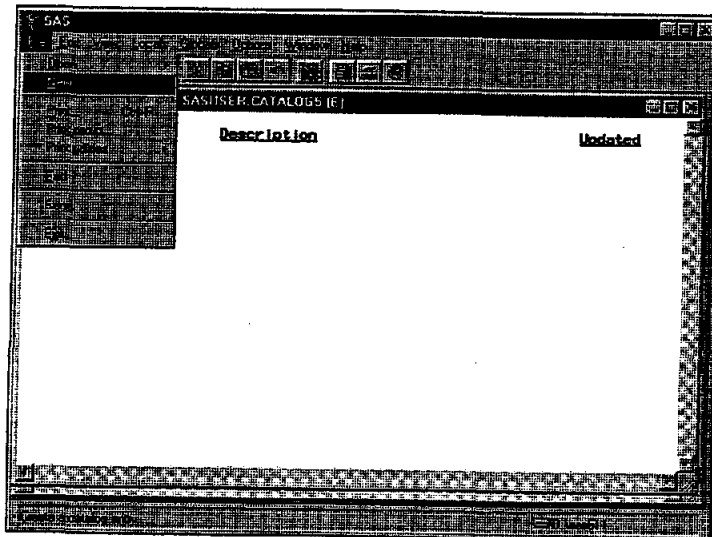
To start the process, utilize either the BUILD command or PROC BUILD.

```
PROGRAM EDITOR (Untitled)
Command ---> SUBMIT
000001 PROC BUILD CATALOG=SASUSER.CATALOGS;
000002 RUN;
000003
```

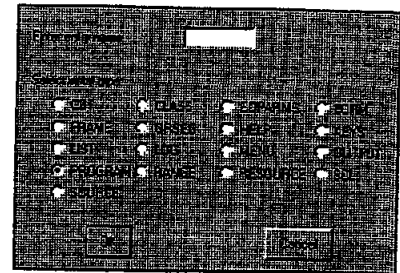
The following screen appears.



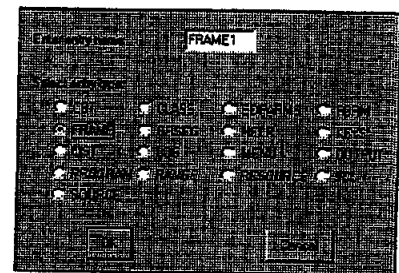
Select File, New from the pull down menus.



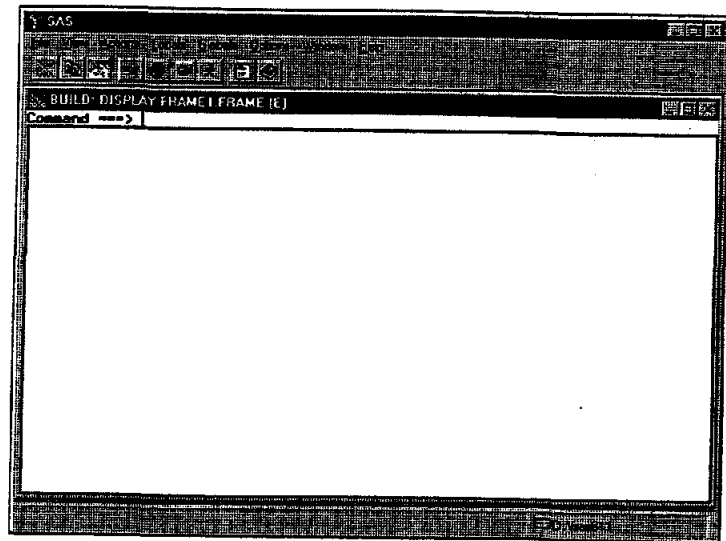
You are prompted for the type of entry you will create.



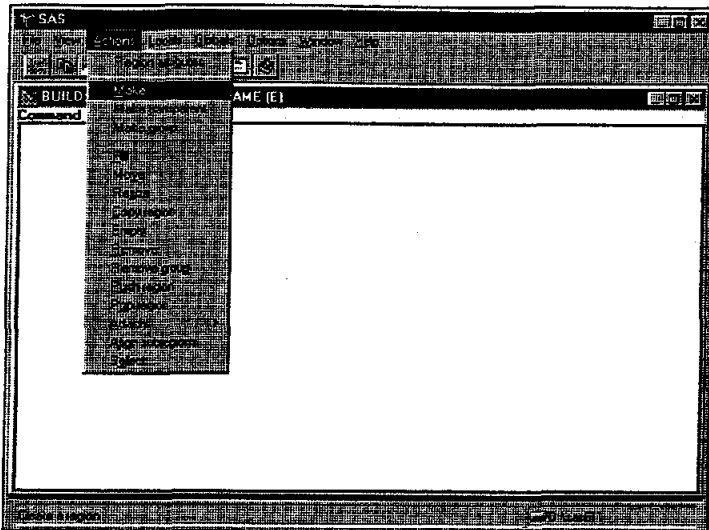
Select FRAME and enter the name of the entry.



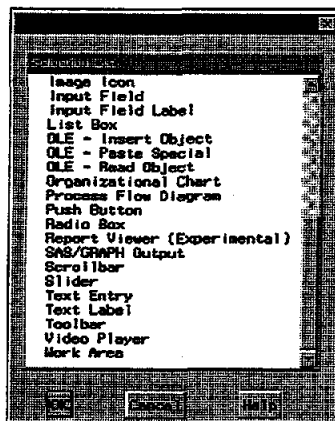
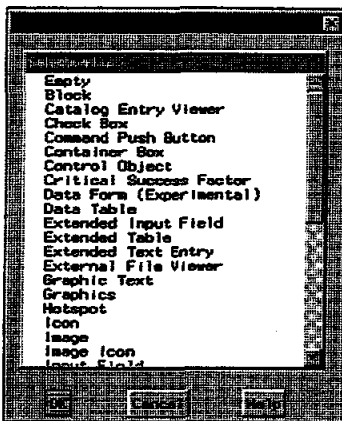
And click . You will be taken to the following Display screen.



To create your first widget, select Actions, Make from the pull down menu.

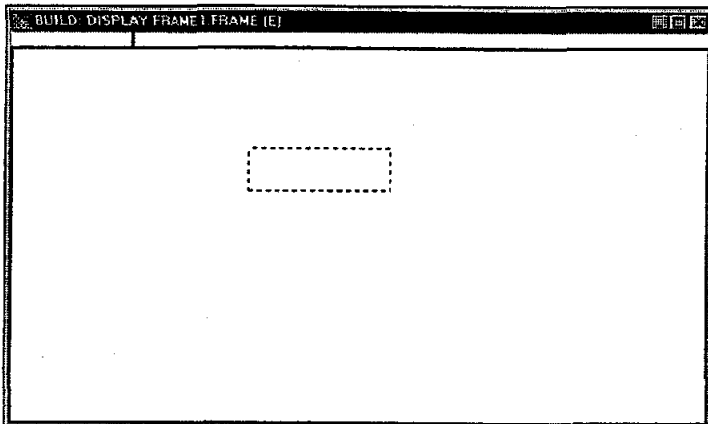


The following screen appears allowing you to select from many different types of widgets to appear on the screen.

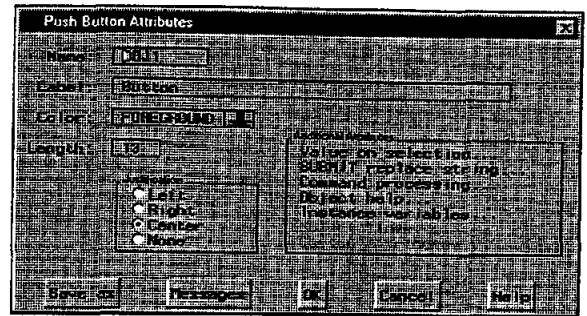


Select  after highlighting the Push Button widget.

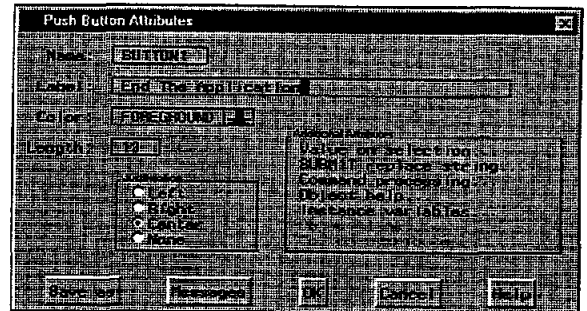
The display reappears to allow you to drop the Push Button on a particular location on the screen.



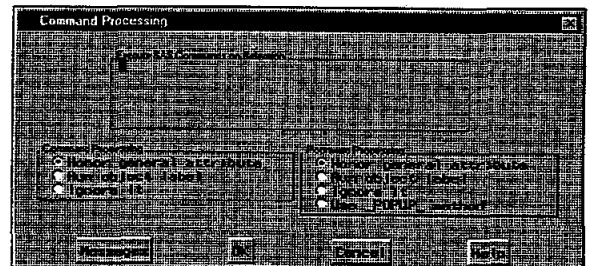
Click your left mouse button and the Push Button Attributes window appears.



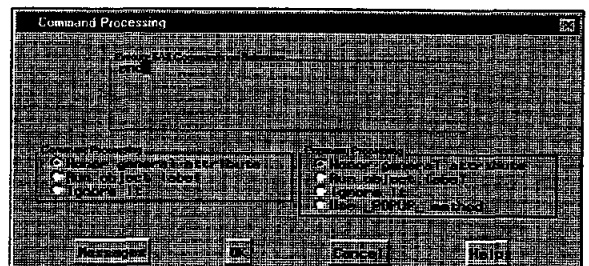
Here we will rename the button and its label.



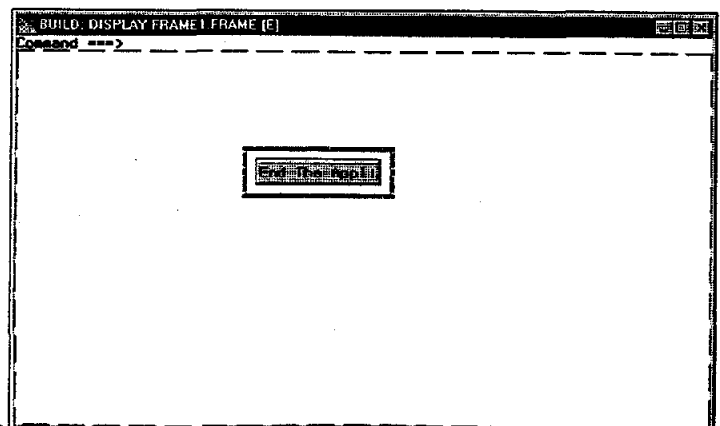
We can specify an internal command in the object. Select Command Processing to see the following screen.



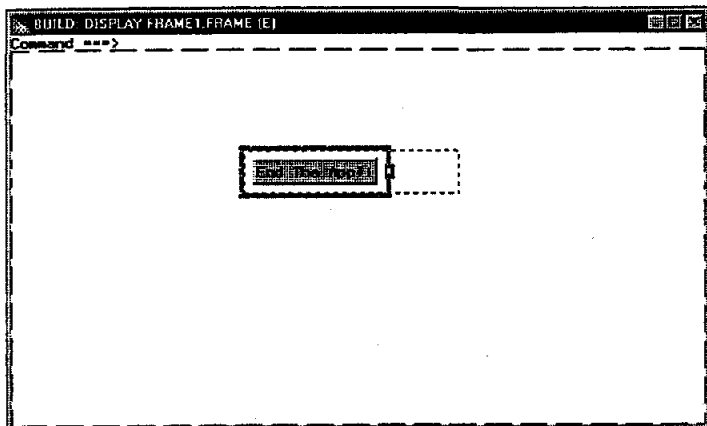
In the box, type the SAS command END.



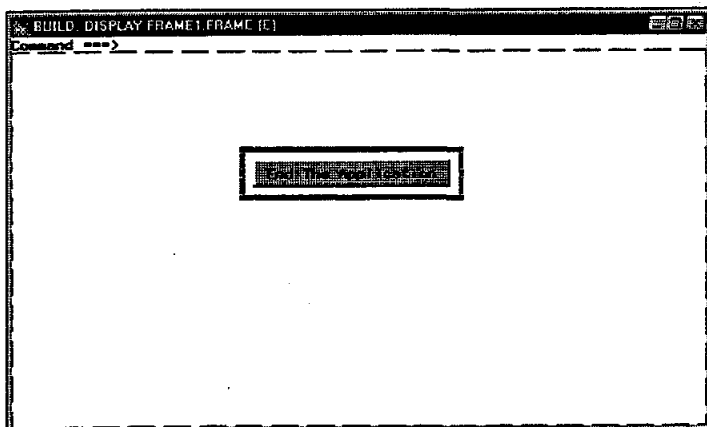
Select  when finished and  again to leave the Attributes window.



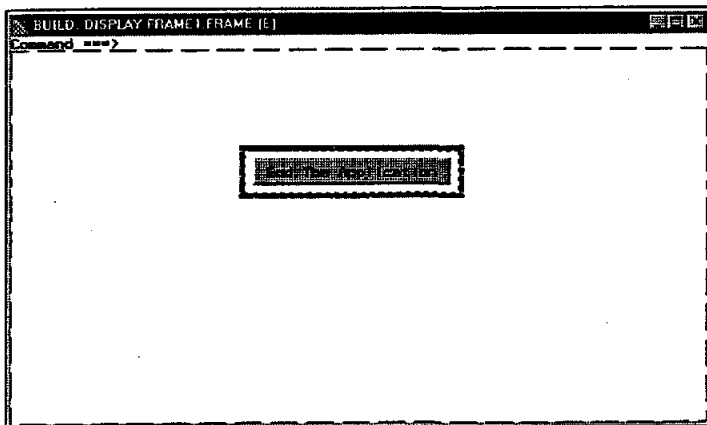
The button in its default size is placed on the display screen. To resize the button, move the mouse to the border until you obtain a double arrow. Hold the left mouse button down and stretch the widget to the desired size.



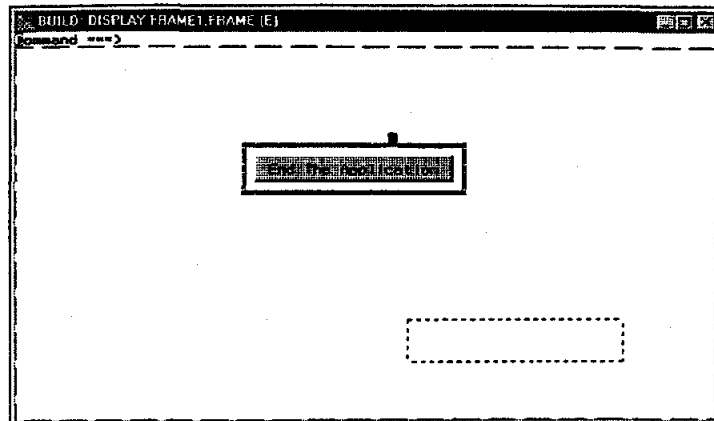
When you release the mouse button, the widget is resized similar to the following.



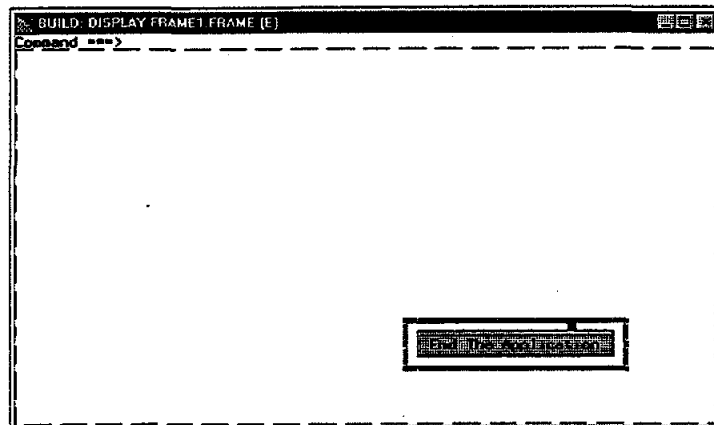
The widget can be moved to another location on the screen by selecting it



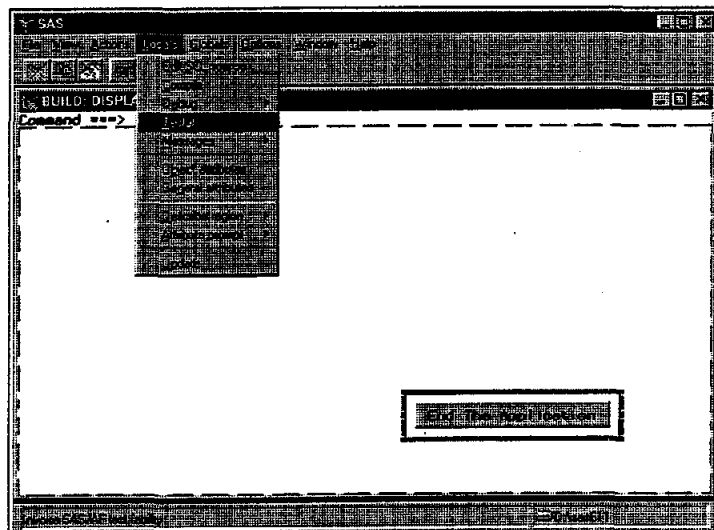
holding the left mouse button down and dragging the object to a different part of the display.



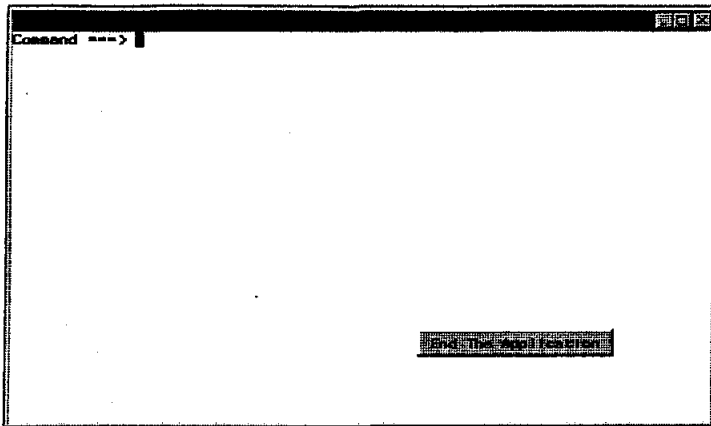
When releasing the object, it is dropped on the destination location.



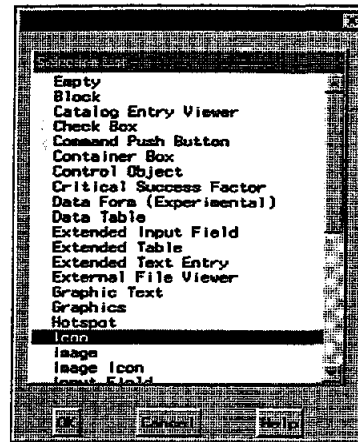
Testing the application is easy. To see how this small application works, simply select Actions, Testaf from the pull down menu.



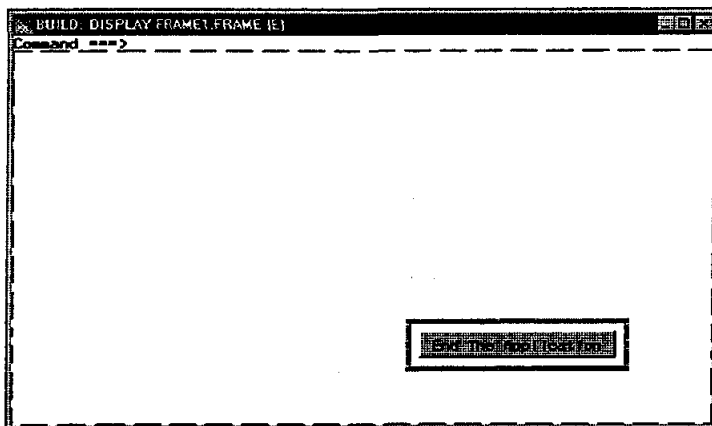
The application runs, displaying the following screen.



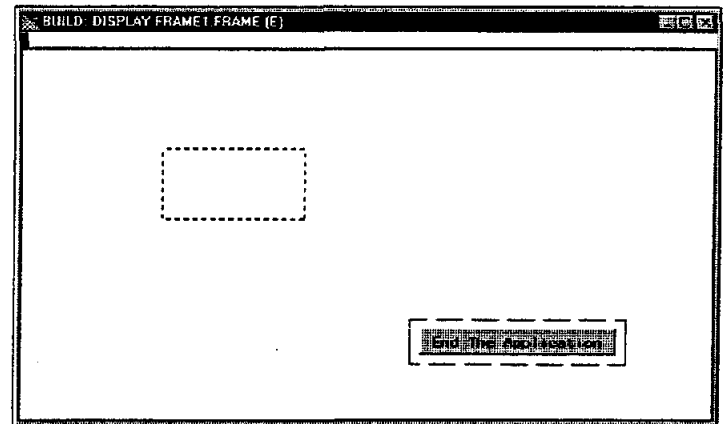
Click the **End The Application** button to issue the End command and end out of Testaf.



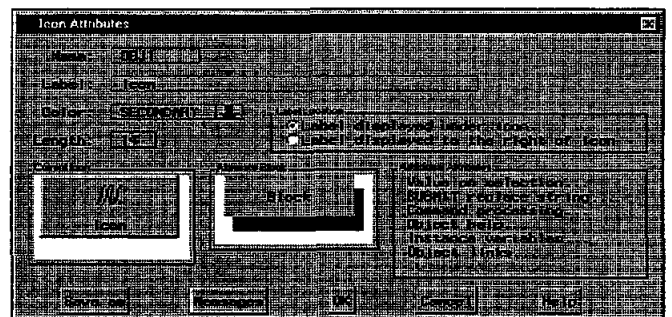
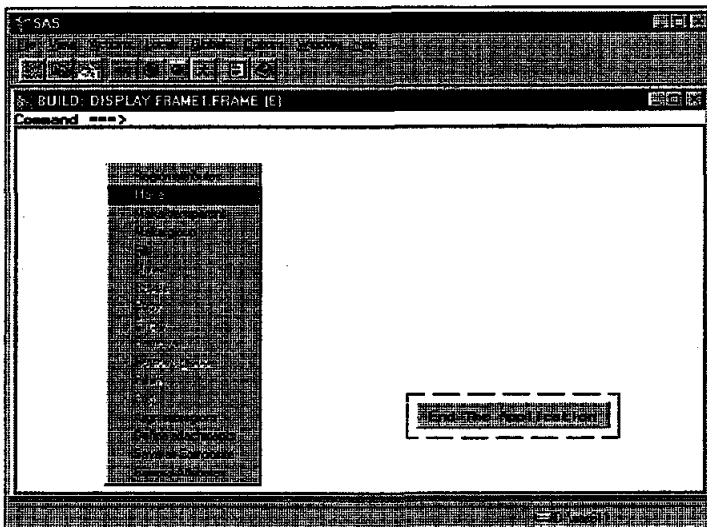
and click **OK**. Drop the icon on the screen.



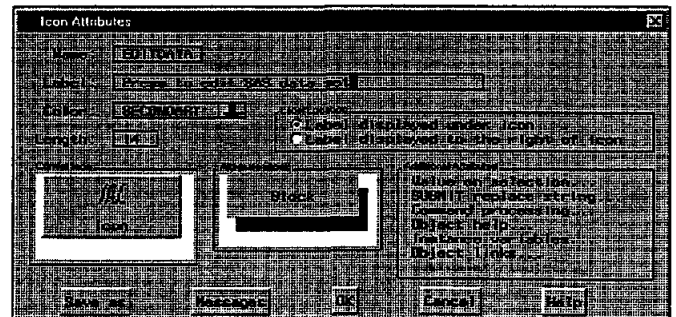
Let's create an Icon that uses Screen Control Language to perform an operation. A right click of the mouse button also displays the pop up menu. Select Make... Icon.



to see the Icon Attributes window.

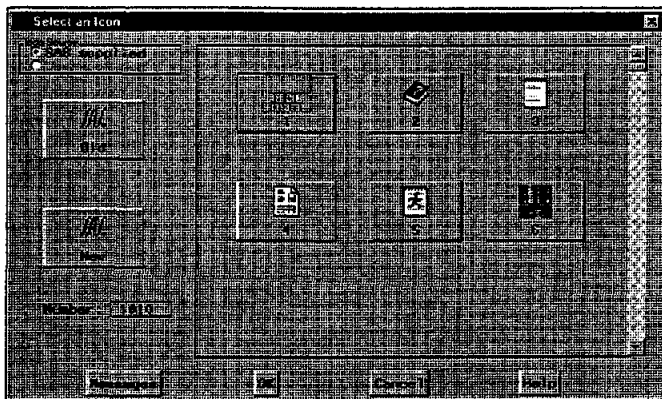


We will change the name of the widget to EDITDATA. Remember this name because it will be used as a label section in the SCL.

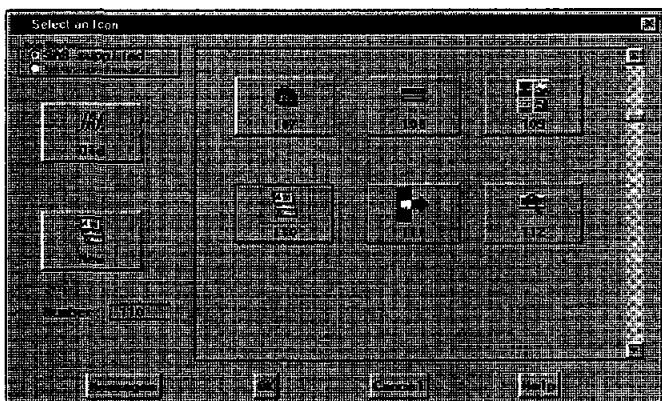


We will also change the default icon used by clicking on **Icon**.

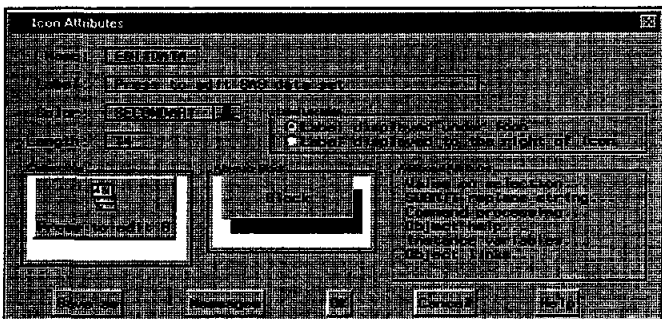
The following selection screen is displayed allowing us to choose a new icon.



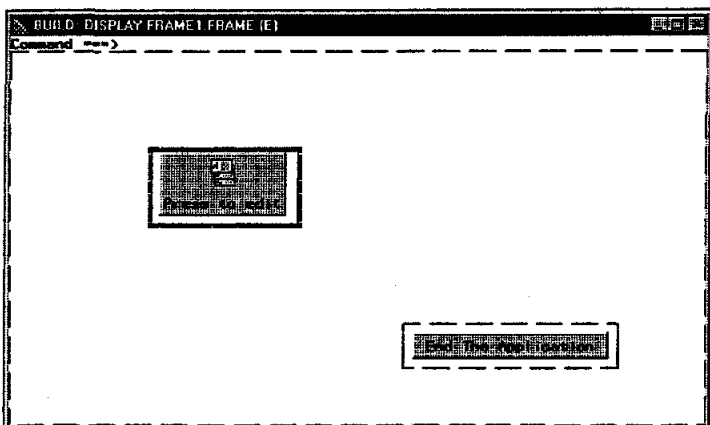
Select one. We chose icon 110



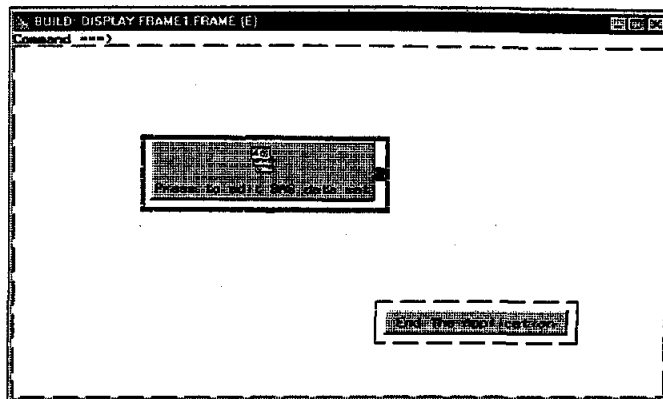
Select  to return to the Icon Attributes screen.



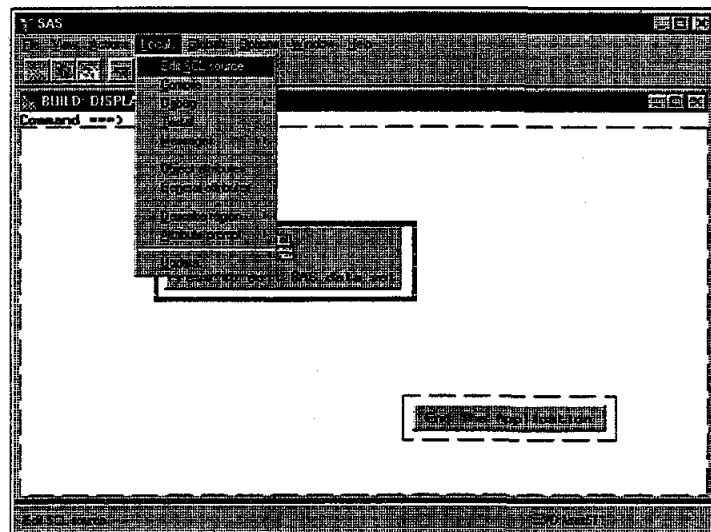
Select  again to return to the display screen.



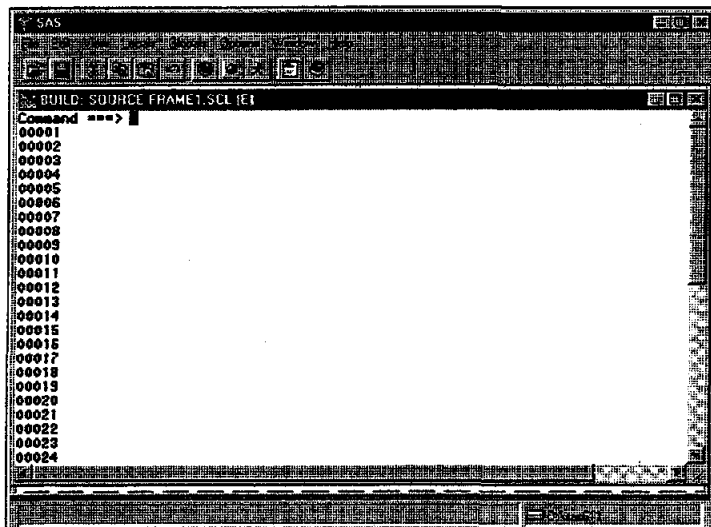
Resize the icon.



Now let's look at creating the SCL. Select Locals, Edit SCL Source from the pull down menu.



The source window is displayed. An SCL entry is created at the same time to store the code you will create here.



Create code similar to the following.

```

BUILD SOURCE FRAME1.SCL (E)
Command ==>
00001 EDITDATA;
00002 call fsedit('saved.customer');
00003 RETURN;
00004
00005
00006
00007
00008
00009
00010
00011
00012
00013
00014
00015
00016
00017
00018
00019
00020
00021
00022
00023
00024

```

The design of this code is simple. We assume a SAS data set called CUSTOMER exists in a library called SAVED and SAS/FSP® software is also licensed. This SCL contains a label called EDITDATA. This section of code executes when the Icon button is depressed. The code is designed to call an FSEDIT session on this SAS data set.

When finished, select Locals, Compile. All source code must be compiled in order to work.

```

SAS
BUILD SOURCE FRAME1.SCL (E)
Command ==>
00001 EDITDATA;
00002 call fsedit('saved.customer');
00003 RETURN;
00004
00005
00006
00007
00008
00009
00010
00011
00012
00013
00014
00015
00016
00017
00018
00019
00020
00021
00022
00023
00024

```

You will see a message on the screen stating the code has compiled successfully. If there are errors, it will not compile. See the message window (MSG) for these errors.

```

SAS
BUILD SOURCE FRAME1.SCL (E)
Command ==>
00001 EDITDATA;
00002 call fsedit('saved.customer');
00003 RETURN;
00004
00005
00006
00007
00008
00009
00010
00011
00012
00013
00014
00015
00016
00017
00018
00019
00020
00021
00022
00023
00024

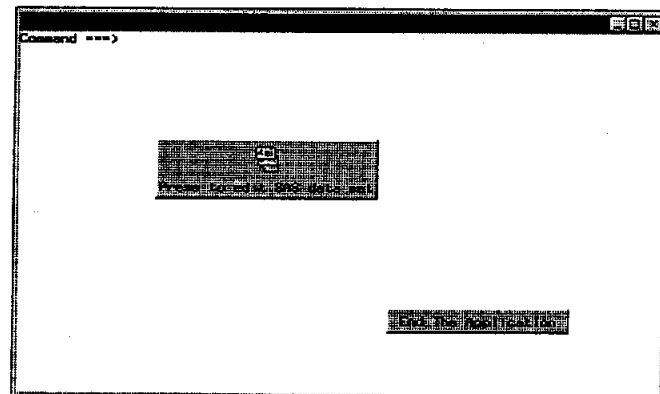
```

From here we can select TESTAF to test the application.

```

SAS
BUILD SOURCE FRAME1.SCL (E)
Command ==>
00001 EDITDATA;
00002 call fsedit('saved.customer');
00003 RETURN;
00004
00005
00006
00007
00008
00009
00010
00011
00012
00013
00014
00015
00016
00017
00018
00019
00020
00021
00022
00023
00024

```



Click the  button and the following screen appears.

```

FSEDIT SAVED CUSTOMER Obs 1
Command ==>

ADDR1: Unit 5,
ADDR2: Brassnose Business Park,
ADDR3: Brighton Road,
ADDR4: Hove
ADDR5: B21 5XL
COMPANY: Bodgit and Scarpa Builder
CUSTKEY: BSB
NAME: Dave Derry
PHONE: 0414 56434

```

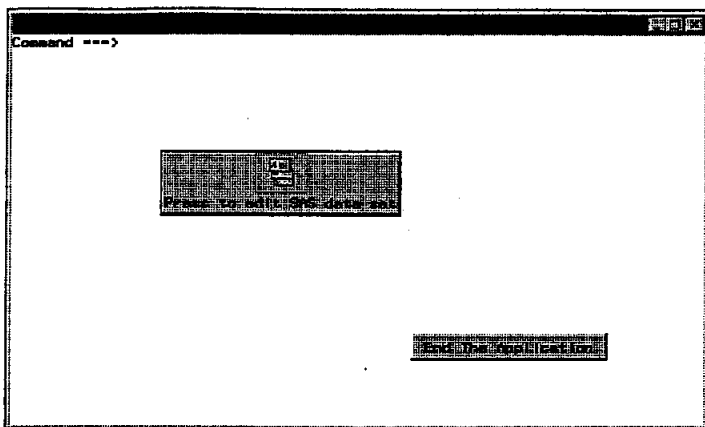
Issue the End command to return.

```

FSEDIT SAVED CUSTOMER Obs 1
Command ==> END

ADDR1: Unit 5,
ADDR2: Brassnose Business Park,
ADDR3: Brighton Road,
ADDR4: Hove
ADDR5: B21 5XL
COMPANY: Bodgit and Scarpa Builder
CUSTKEY: BSB
NAME: Dave Derry
PHONE: 0414 56434

```

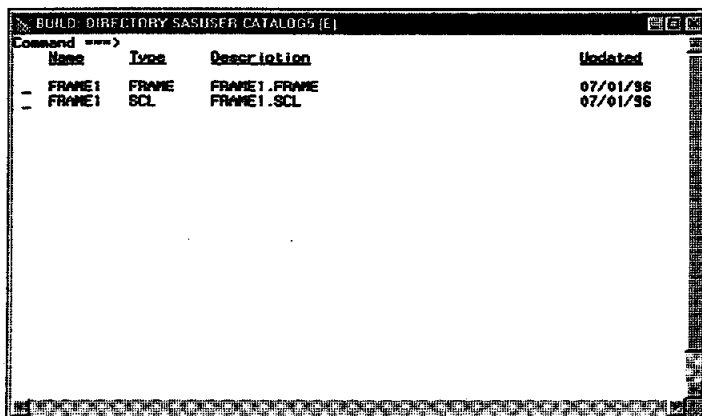


Click **End the application** to end the test.

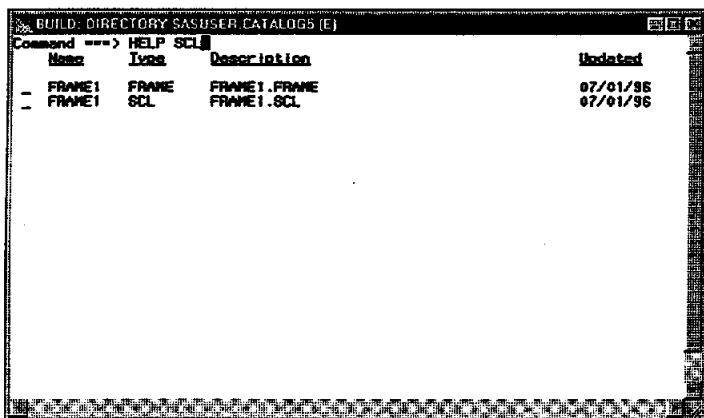
When you are finished creating the frame, select File, End.



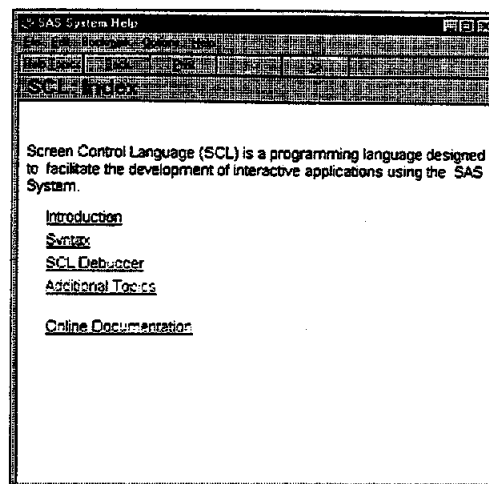
You can see Frame1.Frame and Frame.SCL have been created and live in the catalog.



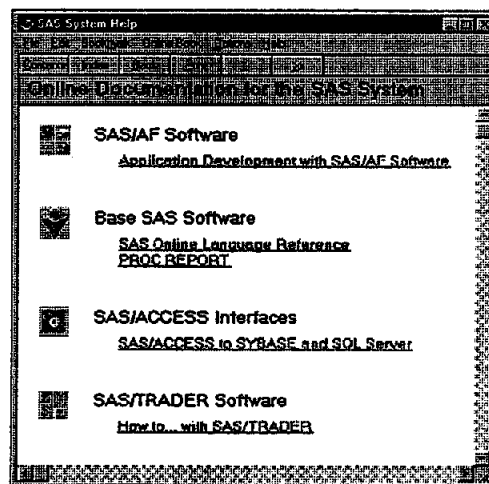
To find out more about the SCL available, issue the HELP SCL command.



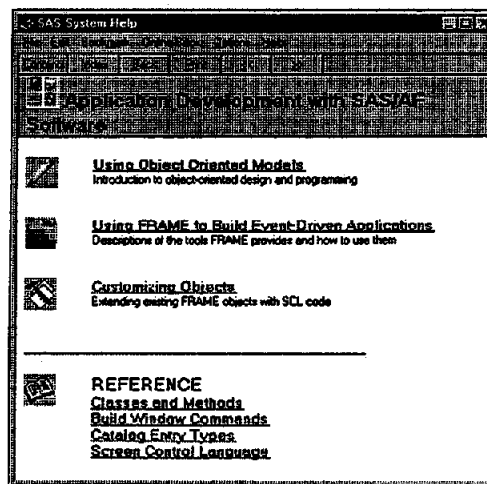
The SCL Index screen is displayed.



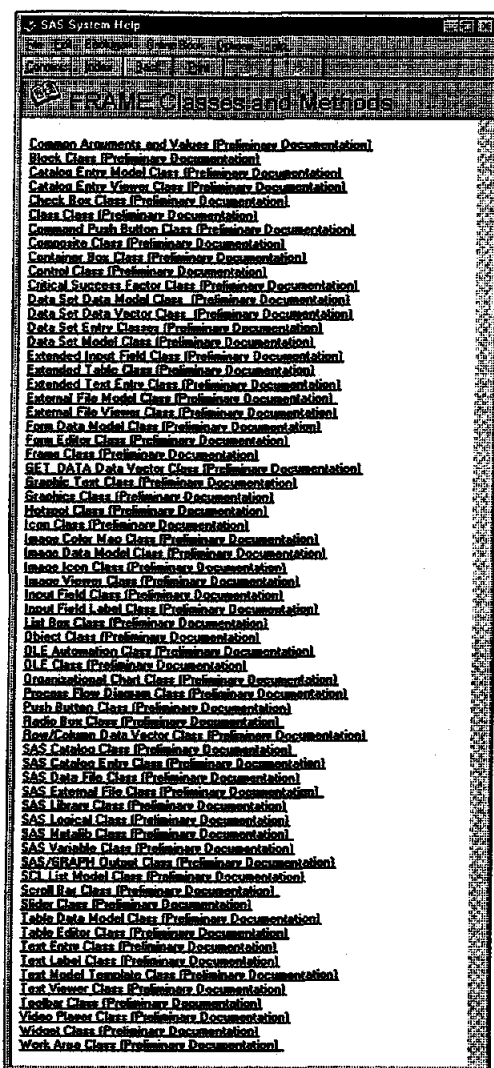
Select Online Documentation to see the following menu. Select Application Development with SAS/AF Software.



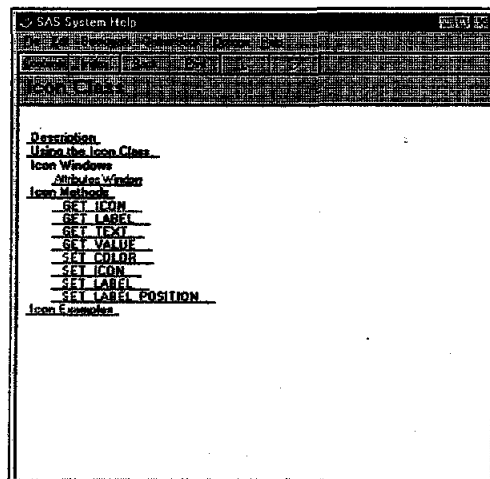
to see the Reference Menu.



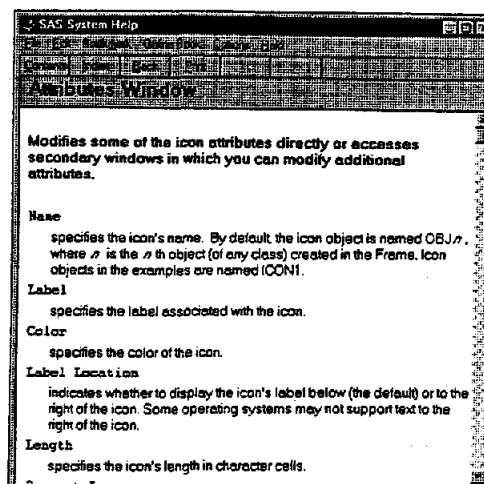
Select Classes and Methods to see an extensive list of the available classes and methods available in your current version of SAS.



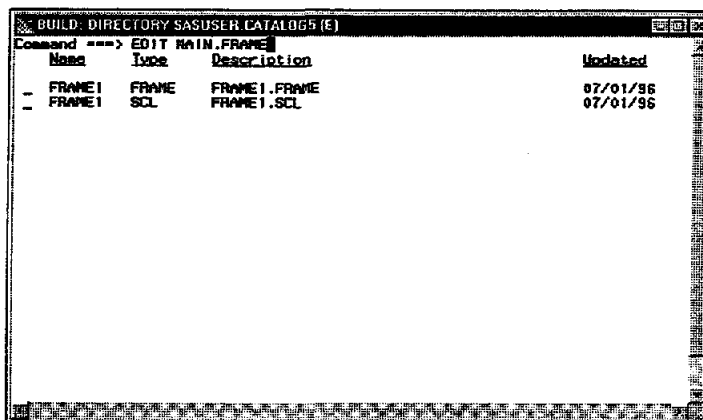
Let's examine the Icon Class. The following screen is displayed to show what is available with this widget. The Attributes screen and options are explained, along with several methods and some examples.



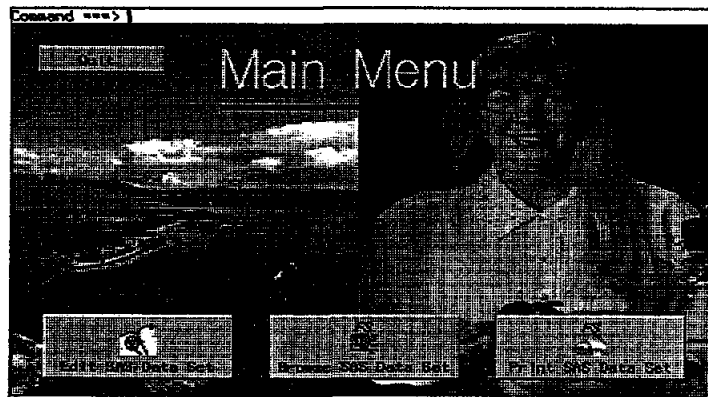
Selecting the Attributes Window shows the following.



When you have finished, you can back out to the Build screen. Let's edit another entry, this time via the EDIT command on the command line. If the entry does not exist, it will be created.



The following shows the result of the edit and some of the wonderful capabilities. The use of a transparent image is the background for this frame. The frame's display simply consists of six objects: A graphic text, push button and some icons.



Further uses of Frame technology will be discussed in this presentation. Any questions can be directed to the author, Dana Rafiee at (860)721-1684 x11.