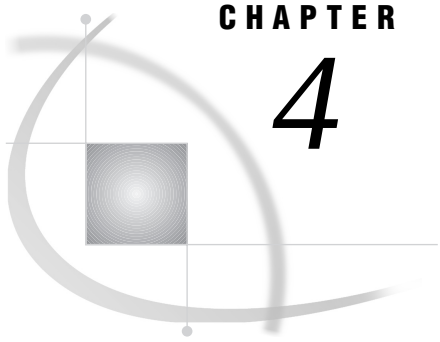


CHAPTER

4

Customizing the SAS Windowing Environment

Introduction

The SAS Windowing environment supports the use of X-based graphical user interfaces. This section enables you to customize your SAS windowing environment by describing the various resources that constitute the SAS interface to Motif and by explaining how to specify different values for those resources.

Motif is an X Window System. On an X Windows display device, the SAS System functions as an X Windows client; that is, many aspects of the appearance and behavior of the SAS windowing environment are controlled by X resources.

Understanding SAS in the X Environment

The X Window System is a networked windowing system. If several machines are linked together in a network, you can run an X application program, or *client*, on one machine in the network and display it on any other machine in the network that is running an X *server*.

X Window Managers

Window managers are X clients that enable you to manage the windows on a display by moving, resizing, and iconifying the windows. The Motif interface to the SAS System can be used with any window manager that is compliant with the *Inter-Client Communication Conventions Manual* (ICCCM). Vendors provide at least one window manager with the X Window System environment.

All window managers perform the same basic functions, but they differ in their style and in their advanced functions. The appearance and function of the interface to SAS depends to some extent on your X window manager. Most window managers provide

some kind of frame around a window. The window manager also governs the placement, sizing, stacking, and appearance of windows, as well as their interaction with the keyboard.

SAS Window Session ID

When you run the SAS System on an X workstation, the SAS System shares the display with other X applications, including other SAS sessions. To enable you to distinguish between different applications and SAS sessions, the SAS System generates a SAS window session ID for each session by appending a number to the application name, which by default is **SAS**. This session ID appears in the window title bar for each SAS window and in the window icon title. The SAS sessions are assigned sequentially. Your first SAS session is not assigned a number, so the session ID is **SAS**; your second SAS session is assigned the session ID **SAS2**; and so on. Although the default application name is **SAS**, you can use the **-name** X option to change the instance name. The instance name can be up to six characters long.

Workspace and Gravity in a SAS Session

When you use the SAS System on an X workstation, the display may be shared by many concurrent applications. When SAS windows from several different sessions and windows from other applications appear on the display, the display can become cluttered. To help alleviate this problem, the windows for a SAS session first appear within an *application workspace* (AWS). The AWS defines a rectangular region that represents a virtual display in which SAS window are initially created. SAS attempts to position the AWS in relation to the upper-left corner of your display. In other words, the workspace gravitates toward a certain direction (*session gravity*) on the display. Some window manager configurations may override the placement that the SAS System has chosen for a window.

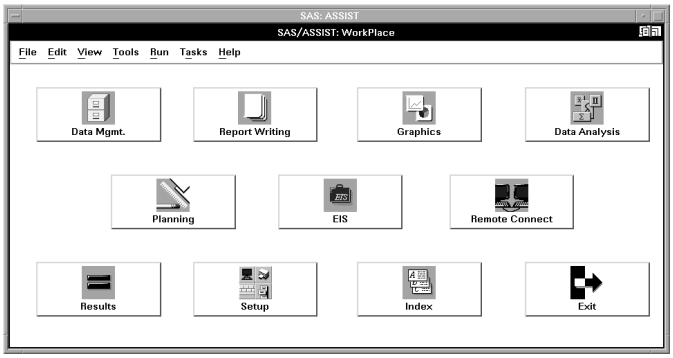
If you issue windowing commands or execute SAS System procedures that create new SAS window, the same rules of initial position and size apply to these windows: they are initially placed in the SAS AWS. You can use the WSAVE command to save the current window positions (or geometry). For details, see “Customizing Session Workspace, Session Gravity, and Window Sizes” on page 115.

Window Types

The SAS System uses primary and interior windows. Some SAS applications consist of one or more primary windows controlled by the X window manager in addition to the interior windows controlled by the SAS System. The SAS windowing environment primary windows, as well as most SAS application windows, initially appear as *top-level windows*. Top-level windows interact directly with the X window manager. They have a full title bar along with other window manager decorations. You can manipulate them individually once they appear on the display. For details about the top-level windows, refer to the SAS online Help.

Interior windows behave differently than primary windows. SAS/ASSIST software is an example of an application with interior windows. Interior windows are contained within *container windows*, which may or may not be primary windows. Display 4.1 on page 74 shows an interior window in SAS/ASSIST software.

Display 4.1 Sample Interior Window



The SAS System provides some degree of window management for interior windows. Specifically, interior windows have the following sizing and movement capabilities:

- You can move interior windows by clicking the left mouse button on the interior window title bar and dragging the window to the desired location. If the destination of the interior window is outside the bounds of the container window, the container window changes according to the value of the **SAS.awsResizePolicy** resource. (The space within the container window is the application workspace, which is described in "Workspace and Gravity in a SAS Session" on page 73.) For more information, see "Using X Resources to Customize the Motif Interface" on page 74.
- Interior windows cannot be iconified individually. Clicking on the container window icon button iconifies the container window and its interior windows.
- A *push-to-back button* (the small overlapping squares in the upper-right corner) is also available with interior windows. However, you cannot push an active window behind an inactive window.
- You can raise a window by clicking on its title bar.

Using X Resources to Customize the Motif Interface

X clients usually have characteristics that can be customized by the system administrator or by the user; these properties are known as *X resources*. For example, X resources can be used to define a font, a background color, or a window size. The resources for an application, such as SAS, are placed in a *resource database*.

The SAS System functions correctly without any modifications to the resource database. However, you may want to change the default behavior or appearance of the interface. There are several ways to specify your customizations. Some methods modify all SAS sessions displayed on a particular X server. Some methods affect all SAS sessions run on a particular host. Other methods affect only a single SAS session. You can use one of the following methods to specify your customizations:

- Edit the X resource file. (See "Editing the X Resource File" on page 75.)
- Use the Preferences dialog box to modify X resource settings. (See "Using the Preferences Dialog Box to Modify X Resource Settings" on page 75.)
- Submit the /XRES command line option. (See the system option "XRESOURCES=" on page 456.)

The following sections describe how to customize the resource database. If you need more information on X Window System clients and X resources, refer to the documentation provided by your vendor.

Specifying X Resources

A resource specification has the following format:

resource-string: *value*

The *resource string* usually contains two identifiers and a separator. The first identifier is the client or application name (**SAS**), the separator is a period (.) or asterisk (*) character, and the second identifier is the name of the specific resource. The *value* given may be a Boolean value (**True** or **False**), a number, or a character string, depending on the resource type.

The application name and resource name can both specify an *instance value* or a *class value*. A specification for a class applies to a larger scope than a single instance.

The following are sample resource specifications:

```
SAS.maxWindowHeight: 100
SAS.awsResizePolicy: grow
```

For more information about resource specifications, refer to your X Window System documentation.

Editing the X Resource File

When you start a SAS session, the SAS interface to Motif calls the X Resource Manager to create a resource database for that session. The interface then examines the resource database for any user-defined X resource definitions.

The X resources for the SAS System are defined in the SAS$XDEFAULTS.DAT resource file in the SYS$LOGIN directory. An example of this file is located in the SAS$ROOT:[TOOLS] directory.

You can change the settings of the X resources in this file without affecting other X Window System applications.

Using the Preferences Dialog Box to Modify X Resource Settings

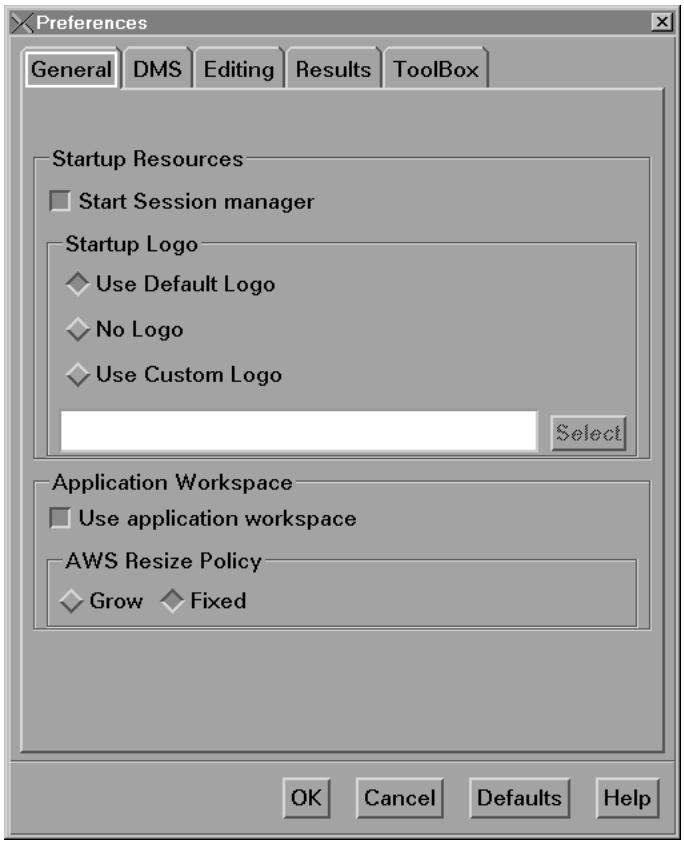
The Preferences dialog box allows you to control the settings of certain X resources. Changes made through the Preferences dialog box (with the exception of those resources on the General tab) become effective immediately, and the settings are saved in SASUSERPREFS.DAT file in your SASUSER directory.

You can open the Preferences dialog box by issuing the DLGPREF command in the command window, or from the **Tools** menu, select **Options**, and then select **Preferences**. Select the tabs at the top of the window to move between the various settings.

Modifying the General Settings

To modify the General settings, select the General tab in the Preferences dialog box. Display 4.2 on page 76 shows the default general settings.

Display 4.2 General Tab in the Preferences Dialog Box



The following is an explanation of the settings:

Start Session manager
This setting has no effect in the OpenVMS operating environment.

Startup Logo
specifies whether you want SAS to display an XPM file while your SAS session is being initialized and, if so, which file.

If you select **Use Default Logo**, SAS uses the default file for your site. If you select **No Logo**, then no file is displayed. If you select **Use Custom Logo**, then you can either enter the XPM filename directly in the text field or click on Select to open the Startup Logo dialog box. These check boxes set the **SAS.startupLogo** resource.

Use application workspace
confines all windows displayed by an application to a single application workspace. This check box sets the **SAS.noAWS** resource. You must exit and reopen the windows for changes to this resource to take effect.

AWS Resize Policy
controls the policy for resizing the application workspace windows as interior windows are added and removed. (For more information, see "Understanding SAS in the X Environment" on page 72 and "Window Types" on page 73.)

Grow
The application workspace window will attempt to grow anytime an interior window is grown or moved (to make all of its interior windows visible), but it will not shrink to remove unused areas.

Fixed
The application workspace window will attempt to size itself to the size of the first interior window and will not attempt any further size changes.

This area sets the **SAS.awsResizePolicy** resource.

After you have changed the settings, click on OK to implement your choices or click on Cancel to cancel your choices. Click on Defaults to return to the default settings. Click on Help for help about the Preferences dialog box.

Modifying the DMS Settings

To modify the display manager settings, select the DMS tab in the Preferences dialog box. Display 4.3 on page 77 shows the default display manager settings.

Display 4.3 DMS Tab in the Preferences Dialog Box

The following is an explanation of the settings:

Use menu access keys
activates menu mnemonics. When mnemonics are turned on, you can select menu items by typing the single, underlined letter in the item. This check box sets the **SAS.usePmenuMnemonics** resource.

Confirm exit
displays the Exit dialog box when you exit your SAS session. This check box sets the **SAS.confirmSASExit** resource.

Save settings on exit
tells SAS to issue the WSAVE ALL command when you exit your SAS session. This command saves the global settings, such as window color and window

position, that are in effect for all windows that are currently open. This check box sets the **SAS.wsaveAllExit** resource.

Backup Documents
allows you to specify whether you want SAS to automatically save (at the interval specified by the **SAS.autoSaveInterval** resource) the documents that you currently have open. This check box sets the **SAS.autoSaveOn** resource.

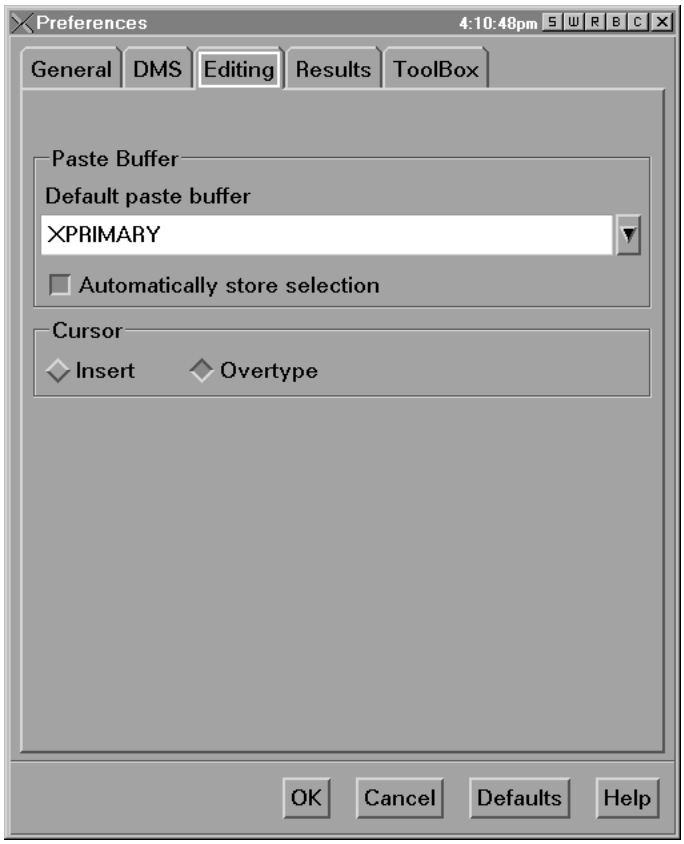
Help & Documentation Browser -- Netscape Path
specifies the location of the web browser executable that you want to use to view the SAS online Help and the CD-ROM documentation. This field sets the **SAS.helpBrowser** resource.

After you have changed the settings, click on OK to implement your choices or click on Cancel to cancel your choices. Click on Defaults to return to the default settings. Click on Help for help about the Preferences dialog box.

Modifying the Editing Settings

To modify the Editing settings, select the Editing tab in the Preferences dialog box. Display 4.4 on page 78 shows the default editing settings.

Display 4.4 Editing Tab in the Preferences Dialog Box



The following is an explanation of the settings:

Default paste buffer
defines an alias for the default SAS buffer. The following list describes the paste buffer alias names and the X buffer with which each name is associated:

XPRIMARY
is the X primary selection (**PRIMARY**). This is the default.

XTERM
is the exchange protocol used by the xterm client.

XSCNDARY
is the X secondary selection (**SECONDARY**).

XCLIPBRD
is the X clipboard (**CLIPBOARD**).

XCUT*n*
is the X cut buffer where *n* is between 0 and 7, inclusive.

This field sets the **SAS.defaultPasteBuffer** resource. For more information about paste buffers, see "Customizing Cut-and-Paste Operations" on page 110.

Automatically store selection
generates a STORE command every time that you mark a region of text with the mouse. This check box sets the **SAS.markPasteBuffer** resource.

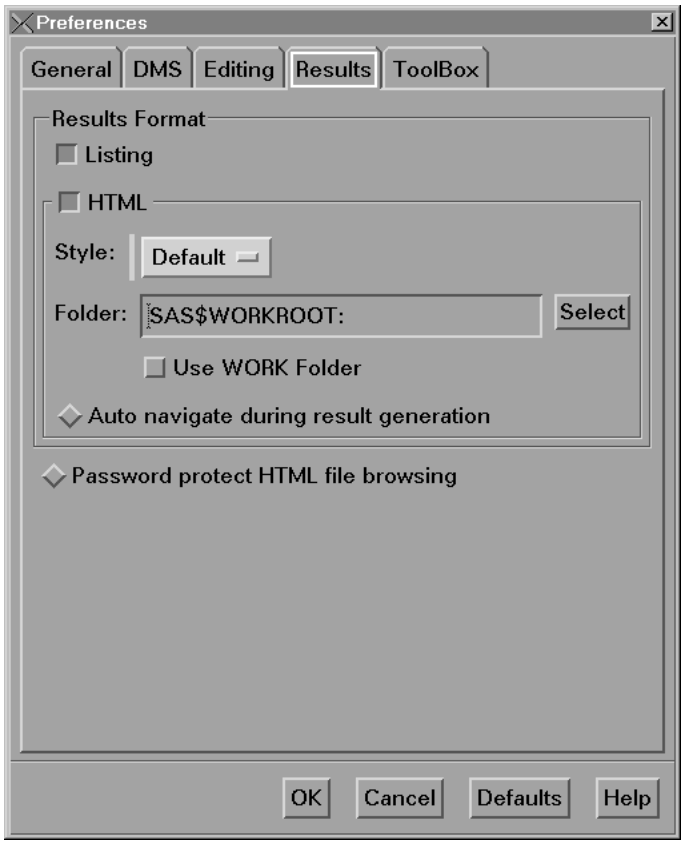
Cursor
controls the editing mode in SAS text editor windows. The **Insert** and **Overtype** check boxes set the **SAS.insertModeOn** resource to **True** and **False**, respectively.

After you have changed the settings, click on OK to implement your choices or click on Cancel to cancel your choices. Click on Defaults to return to the default settings. Click on Help for help about the Preferences dialog box.

Modifying the Results Settings

The items in the Results area of the Preferences dialog box control the format of the results that you produce. Display 4.5 on page 80 shows the default results settings.

Display 4.5 Results Tab in the Preferences Dialog Box



The following is an explanation of the settings:

Listing specifies that your output is in text format. You can select both **Listing** and **HTML** format.

HTML specifies that your output is in HTML format. You can select both **Listing** and **HTML** format.

Style specifies the HTML style to use when generating output in HTML format.

Folder specifies the folder in which to store your HTML results. Click Select to choose a folder. Select **Use WORK Folder** to designate your SAS WORK folder as the repository for your HTML results.

Auto navigate during result generation specifies that HTML output is automatically displayed when it is generated. This value cannot be selected if you have selected **Password protect HTML file browsing**.

Password protect HTML file browsing requires a password when sending HTML files to a browser. This value cannot be selected if you have selected **Auto navigate during result generation**.

Modifying the Toolbox and Command Window Settings

The items in the ToolBox area of the Preferences dialog box affect both the toolbox and the command window. To modify these settings, select the ToolBox tab in the Preferences dialog box. Display 4.6 on page 81 shows the default ToolBox settings.

Display 4.6 ToolBox Tab in the Preferences Dialog Box

The following is an explanation of the settings:

Display tools window
determines whether to display the default toolbox. This check box sets the **SAS.defaultToolBox** resource.

Display command window
determines whether to display the command window. This check box sets the **SAS.defaultCommandWindow** resource.

Auto Complete Commands
specifies that commands will be automatically completed as you type the first letters of the command.

Save Commands
specifies the number of commands to be saved. You can change the number by clicking on the up or down arrow next to the number of commands. The default is 25.

Combine windows
combines the toolbar and command window into one window. The toolbar and command window are combined by default. This check box sets the **SAS.useCommandToolBoxCombo** resource.

Use arrow decorations
adds left and right arrows to either end of the combined toolbox/command window. This check box sets the **SAS.useShowHideDecorations** resource.

Always on top
keeps the toolbox or the combined toolbar/command window on top of the window stack. This check box is selected by default, which could cause problems with window managers that are not Motif window managers or other applications that want to be on top of the window stack. If you have such a situation, turn off this feature. This check box sets the **SAS.toolBoxAlwaysOnTop** resource.

Toolbox Persistent
specifies whether the toolbox that is associated with the Program Editor window stays open when you close the window. By default, the Program Editor toolbox stays open whenever you close the Program Editor window. If you deselect this check box, then the toolbox will close if you close the Program Editor window. This check box sets the **SAS.isToolBoxPersistent** resource.

Tools
The items in the Tools area affect the individual tools in the toolbox:

Use large tools
controls whether tool icons are displayed as 24x24 or 48x48 pixels. The default is 24x24. This check box sets the **SAS.useLargeToolBox** resource.

Use tip text
specifies whether tool tip text is displayed when you position your cursor over a tool in the toolbox. Some window managers may place the toolbox tip behind the toolbox. If this happens in your environment, deselect this check box. This check box sets the **SAS.useToolBoxTips** resource.

delay
controls the delay in milliseconds before popping up the toolbox tip. This field sets the **SAS.toolBoxTipDelay** resource. You can enter a value directly into the field or use the arrows to the right of the field to change the value.

After you have changed the settings, click on OK to implement your choices or click on Cancel to cancel your choices. Click on Defaults to return to the default settings. Click on Help for help about the Preferences dialog box.

Using the SAS ToolBox

For all of the operating environments that the SAS System runs under, you can enter commands from many SAS windows. For information about commands that are available under OpenVMS, see Chapter 11, “Commands,” on page 219 and the SAS commands section in the SAS online Help. On many hosts, including OpenVMS, you can use the SAS ToolBox command window to enter commands.

The SAS ToolBox has two parts, as illustrated in Display 4.7 on page 82:

- A command window that enables you to quickly enter any command.
- A toolbar that contains several tool icons. When you select a tool icon, SAS immediately executes the command that is associated with that icon. The toolbar and the tool icons are completely customizable. (See “Using the Tool Editor” on page 84.)

Display 4.7 SAS ToolBox

Under OpenVMS, the default ToolBox automatically appears at the bottom of the SAS windows by default. To control its configuration, you use the Preferences dialog box. (See "Modifying the Toolbox and Command Window Settings" on page 80.)

The name of the active window is displayed in the title bar of the toolbox. For example, if the Log window were active, the title bar would say ToolBox: Log instead of ToolBox: Program Editor.

If the toolbox is not displayed automatically when the SAS System initializes, it is due to one of the following reasons:

- You executed your SAS job in a nonwindowing mode such as batch.
- The **SAS.defaultToolBox** resource is set to **False**. The default value is True. For more information about the resources that control the toolbox, see "Specifying Toolbox X Resources" on page 83.
- You deselected **Display tools window** in the Preferences dialog box. (See "Using the Preferences Dialog Box to Modify X Resource Settings" on page 75.)

If you invoke an application that does not have a toolbox that is supplied by the SAS System, then the default toolbox is displayed for that application. If you then customize the toolbox for that application, the customized toolbox is stored in SASUSER.PROFILE.<*entry*>.TOOLBOX, where < *entry*> is the same entry name as the PMENU entry for the application window.

Customizing Toolboxes and Toolsets

You can customize toolboxes

- through the Preferences dialog box. The Preferences dialog box allows you to customize the appearance and behavior of toolboxes. For information about using the Preferences dialog box, see "Using the Preferences Dialog Box to Modify X Resource Settings" on page 75 and "Specifying Toolbox X Resources" on page 83.
- by specifying X resources in your resource file. For more information about X resources that affect toolboxes, see "Specifying Toolbox X Resources" on page 83.
- through the Tool Editor dialog box. The Tool Editor allows you to customize the individual tools in a toolbox. For more information, see "Using the Tool Editor" on page 84.

The Tool Editor also allows you to create custom toolsets for your SAS applications. A *toolset* is a set of predefined tools that is associated with an application. Toolsets make it easier for individual users to customize their application toolboxes. If you create a toolset for an application, users can simply choose the tools they want to appear in their toolboxes and do not have to define the icons, commands, tip text, and IDs for those tools. For example, you can define a default toolbox for your application that includes tools for opening files, cutting, copying, and pasting text, and saving files. You can define a toolset that includes those tools and tools for opening the Preferences dialog box, opening the Replace dialog box, and entering the RECALL command. These additional tools will not appear in the users' toolbox unless a user adds them to their toolbox with the Tool Editor. You can view the TOOLSET contents by opening the Tool Editor dialog box and clicking on Actions..., provided that the TOOLSET name has the same name as the TOOLBOX you are editing. For more information, see "Changing an Existing Tool" on page 85 and "Creating or Customizing an Application- or Window-Specific Toolset" on page 88.

Specifying Toolbox X Resources

You can control the behavior of toolboxes with the following resources:

SAS.defaultToolBox: [True | False]
controls opening the default toolbox when the SAS System is invoked. The default is True.

SAS.isToolBoxPersistent: [True | False]
controls whether the toolbox stays open when you close the Program Editor window. The default value is True, which means that the toolbox remains open whenever you close the Program Editor window.

SAS.toolBoxAlwaysOnTop: [True | False]
controls whether the toolbox is always on top of the window stack. The default value is True, which may cause problems with window managers that are not Motif window managers or other applications that want to be on top of the window stack. If you have such a situation, set this resource to False.

SAS.toolBoxTipDelay: ***delay-in-milliseconds***
sets the delay in milliseconds before displaying the toolbox tip. The default is 750.

SAS.useCommandToolBoxCombo: [True | False]
controls whether the command window and toolbox are joined or separated. The **SAS.defaultToolBox** and **SAS.defaultCommandWindow** resources control whether the toolbox and command window are displayed. If both are displayed, this resource controls whether they are joined or separated. The default is True.

SAS.useLargeToolBox: [True | False]
controls whether tool icons in the toolbox are displayed as 48x48 pixels or 24x24 pixels. The default is False (24x24 pixels).

SAS.useShowHideDecorations: [True | False]
controls whether the combined command/toolbox window has arrows at the left and right. You can use these arrows to hide or show portions of the window as they are needed. The default is False.

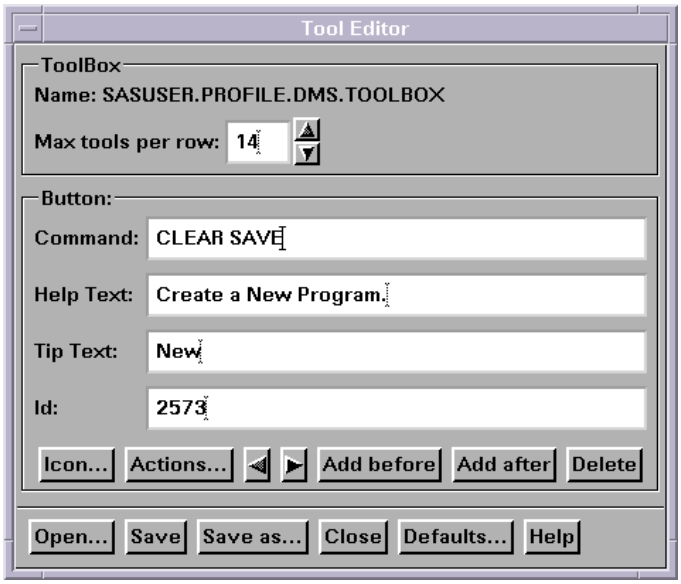
SAS.useToolBoxTips: [True | False]
determines if toolbox tip text is displayed. Some window managers, such as **tvtwm**, may place the toolbox tip behind the toolbox. If this happens in your environment, set this resource to False. The default is True.

Using the Tool Editor

The Tool Editor dialog box enables you to modify the contents and appearance of your toolboxes, as shown in Display 4.8 on page 85. To invoke the Tool Editor dialog box, do one of the following:

- From the **Tools** menu in the active window, select **Options**, and then select **Edit Toolbox...**.
- Issue the TOOLEDIT command in the command window (described in the command “TOOLEDIT” on page 232).
- Click on the window menu button in the upper-left corner of the toolbox and select **Edit Toolbox...**. The **Edit Toolbox...** item is displayed when you use the Motif window manager. Contact your system administrator for more information about the Motif window manager.

Display 4.8 Tool Editor Dialog Box



By default, the Tool Editor dialog box edits the current toolbox. To edit a different toolbox, invoke the Tool Editor dialog box and click on Open... . Then specify the libref, catalog, and entry name for the toolbox that you want to edit. Click on OK .

After you invoke the Tool Editor dialog box, the toolbox goes into "preview mode." In preview mode, clicking on a tool icon makes that icon the current icon and displays its associated commands in the **Command:** field. The current icon always appears selected.

Changing the Appearance of the Entire ToolBox

The items in the **ToolBox** area of the Tool Editor dialog box affect the entire toolbox:

Name:
displays the catalog entry that you are editing. The default toolbox is named SASUSER.PROFILE.DMS.TOOLBOX.

Max tools per row:
specifies how the icons in the toolbox are arranged. The default size creates a horizontal toolbox. One tool per row creates a vertical toolbox.

Changing an Existing Tool

When you open the Tool Editor dialog box, the first icon in the toolbox is the current icon (the icon appears selected), and its information appears in the **Button:** area of the dialog box. To change an existing tool, you can select a tool from the toolset displayed by clicking on Actions... or you can modify the fields individually.

Note: Clicking on Actions... will display a toolset only if a toolset is associated with (has the same entry name as) the toolbox that you are editing. For more information, see "Saving Changes to the Toolbox or Toolset" on page 87. △

To use Actions... , select the tool that you want to change, and then select Actions... . The Tool Editor displays the toolset associated with the toolbox. If you select a tool from this toolset, the Tool Editor enters the appropriate information into the button fields for you.

To modify the fields individually, perform the following steps:

1 Select the icon you want to change.
2 Click in and change the **Button:** area fields as appropriate:

Command:
specifies the SAS windowing environment command or commands that you want executed when you select the icon. You can use any SAS command that is available under OpenVMS. For information about commands, see Chapter 11, "Commands," on page 219 and the SAS commands section in the SAS online Help. For example, you could create an icon to open the Change Directory dialog box by using the DLGCDIR command, which is described in "DLGCDIR" on page 222.
Separate commands with a semicolon (;).

Help Text:
is used for applications that are designed to be run under Microsoft Windows or OS/2. The help text is displayed in the SAS windowing environment status line on Windows and OS/2 when a toolbox is ported to and loaded on those operating environments.

Tip Text:
specifies the text that is displayed when you position the cursor over the icon.

Id:
is useful if you are creating toolboxes for SAS/AF applications. The ID is the identifier of the corresponding menu item in the application. This number is the value assigned to the item in the ID option of the ITEM statement in PROC PMENU. If you specify an ID, then the application can set the state of the PMENU item to match the state of the tool in the toolbox, and it can make the PMENU item active or inactive to match whether the PMENU item is active or inactive. If you do not specify an ID, the ID defaults to 0.

3 Change the icon if necessary:
   a Click on Icon... or double-click on an icon in the preview toolbox. The Select a Pixmap dialog box opens, which displays the icons provided with the SAS System. These icons are divided into several categories such as SAS windows; data; analysis; numbers and symbols; files, folders, and reports; and so on. To change categories, click on the arrow to the right of the **Icon Category** field and select a new category.
   b Select the icon you want to use and then click on OK.
   c Save your changes as described in "Saving Changes to the Toolbox or Toolset" on page 87.

Adding Tool Icons to the Toolbox

To add an icon to the toolbox, perform the following steps:
1 Select the icon that is adjacent to where you want to add the new tool icon.
2 Click on Add before or Add after, whichever is appropriate. The Toolbox Editor adds a new icon (labeled SAS) to the toolbox and clears the **Button:** area fields.
3 Enter the appropriate information in the **Button:** area fields. These fields are described in "Changing an Existing Tool" on page 85.
4 Change the icon, if desired, as described in "Changing an Existing Tool" on page 85.
5 Save your changes as described in "Saving Changes to the Toolbox or Toolset" on page 87.

Changing the Order of the Icons in the Toolbox

To change the position of a tool in the toolbox, select the icon, and then click on the left or right arrows in the **Button:** area to move the icon.

Deleting Tool Icons from the Toolbox

To delete a tool icon from the toolbox:

1. Select the icon that you want to delete.
2. Click on Delete in the **Button:** area.
3. Save your changes, as described in "Saving Changes to the Toolbox or Toolset" on page 87.

Returning to the Default Settings

To return all tools in the current toolbox to their default settings, click on Defaults... at the bottom of the Tool Editor dialog box. When you make this selection, any customizations that you have made are lost. It also deletes any icons that you have added to the toolbox. The Tool Editor dialog box asks you to verify your request. Click on Yes to restore the default settings, No to keep your customizations, or Cancel to do neither.

Saving Changes to the Toolbox or Toolset

You can save the changes to the catalog entry that is displayed in the **Name:** field of the Tool Editor dialog box, or you can create a new toolbox with a different name.

Note: If you are customizing a window-specific or application-specific toolbox for your personal use, save the customized toolbox in your SASUSER.PROFILE catalog by using the same entry name as the PMENU entry for the window or application. The SAS System searches for toolboxes first in SASUSER.PROFILE and then in the application catalog. △

Click on Save or Save as ... to perform the following tasks:

Save
: saves the toolbox information to the catalog entry shown in the **Name:** field.

Save as...
: causes the Tool Editor dialog box to prompt you to enter a different libref, catalog, and entry name. The entry type for a toolbox is always TOOLBOX.

  You can also choose to save the toolbox as a toolset. If you save the toolbox as a toolset, the entry type will be TOOLSET; otherwise, the entry type is always TOOLBOX. Saving a set of tools as a TOOLSET does not change your TOOLBOX entry. For more information about toolsets, see "Customizing Toolboxes and Toolsets" on page 83 and "Creating or Customizing an Application- or Window-Specific Toolset" on page 88.

If you click on Close or Open... without first saving your changes, the Tool Editor dialog box prompts you to save the changes to the current toolbox before continuing. Click on Yes to save your changes; click on No to ignore any changes you made; or click on Cancel to do neither.

After you save the toolbox or toolset, the Tool Editor remains open for additional editing, and the **Name:** field changes to the name of the new entry (if you entered a new name).

Note: To SAS/AF application developers or site administrators: If you are editing a window-specific or application-specific toolbox that you want to be accessible to all users, you must save the TOOLBOX entry with the same library, catalog, and entry name as the PMENU entry for the window or application. You need WRITE access to the appropriate location. For example, to store a customized toolbox for the graphics editor, the site administrator would store the toolbox in SASHELP.GI.GEDIT.TOOLBOX. △

Creating a New ToolBox

To create a new toolbox, you do one of the following:

- Edit an existing toolbox by invoking the Tool Editor dialog box (from the **Tools** menu in the active window, select **Options**, and then select **Edit Toolbox...**) and making your changes. Then save the edited toolbox, as described in "Saving Changes to the Toolbox or Toolset" on page 87.
- Copy an existing TOOLBOX entry. You can copy a toolbox by opening the CATALOG window, specifying the appropriate catalog, and then entering the COPYITEM command. For example, to copy the default toolbox, enter the following commands in the command window:

```
catalog sasuser.profile
   copyitem dms.toolbox
            newtools.toolbox
```

Next, edit the new toolbox by using the Tool Editor dialog box (from the **Tools** menu in the active window, select **Options**, and then select **Edit Toolbox...**). Click on Open... . Then specify the libref, catalog, and entry name of the new toolbox. Click on OK to create the new toolbox.

Loading a Different ToolBox

To load a toolbox, issue the TOOLLOAD command in the command window:

```
TOOLLOAD <libref.catalog.entry>
```

Creating or Customizing an Application- or Window-Specific ToolBox

If you are an application developer and want to create or edit an application TOOLBOX, follow these steps:

1. Delete any existing TOOLBOX entry in your SASUSER.PROFILE for the window or application that you want to customize. When you delete the copy of the toolbox in your SASUSER.PROFILE catalog, you automatically receive a copy of the toolbox that is supplied with the SAS System when you invoke the Tool Editor dialog box.
2. Create or edit the application toolbox, as described in "Creating a New ToolBox" on page 88 or "Using the Tool Editor" on page 84.
3. Save the edited toolbox, as described in "Saving Changes to the Toolbox or Toolset" on page 87.
4. Inform your users that you have changed the window or application toolbox. If they want to use the new toolbox, they must delete the corresponding TOOLBOX entry from their SASUSER.PROFILE. The new toolbox will then be automatically loaded when the window or application is invoked. If users do not delete the corresponding TOOLBOX entry from their SASUSER.PROFILE catalog, that copy of the toolbox will be loaded instead of the new toolbox.

Creating or Customizing an Application- or Window-Specific Toolset

You define application- or window-specific toolsets in the same way that you create an application- or window-specific toolbox. There are only two differences:

- To create a new toolset, start by defining a toolbox as described in "Creating a New ToolBox" on page 88.
- After you have defined the toolbox, save it as a TOOLSET entry, not as a TOOLBOX entry.

Note: If you are an application developer, make sure that you delete an existing TOOLSET entry for your application as described in "Creating or Customizing an Application- or Window-Specific ToolBox" on page 88 before you modify your application's toolset. △

Customizing Key Definitions

You can define most of the keys on your keyboard. However, a few keys have dedicated functions that are associated with them. For example, the mouse buttons are dedicated to the cursor and cut-and-paste operations and are not available for user customization. For more information, see "Default Keyboard Actions" on page 97.

You can customize your key definitions by using one of the following methods:

- Through the KEYS window. To open the KEYS window, do one of the following:
  - Issue the KEYS command.
  - From the **Tools** menu, select **Options**, and then select **Keys**.

  If you change any key definitions through the KEYS window for the SAS windowing environment windows, the definitions are stored in the catalog SASUSER.PROFILE in the entry DMKEYS.KEYS. Key definitions for other SAS windows are stored in catalog entries named BUILD.KEYS, FSEDIT.KEYS, and so on.

  For more information about the KEYS command and the KEYS window, refer to the SAS online Help.
- With the KEYDEF command. The KEYDEF command allows you to redefine individual function keys:

  `keydef` *`keyname <command | ~text-string>`*

  For example, if you specify **`keydef F8 dlgpref`**, then the F8 key will issue the DLGPREF command that opens the Preferences dialog box.

  For more information about the KEYDEF command, refer to the SAS online Help.
- By defining the **SAS.keyboardTranslations** and **SAS.keysWindowLabels** resources in your resources file as described in "Defining Key Translations" on page 89.

Defining Key Translations

To customize a key for the X Window System, you define a key sequence and specify an action to be executed when that key sequence is typed on the keyboard. This is known as *binding* keys to *actions*; together they are referred to as a *translation*.

The **SAS.keyboardTranslations** resource specifies the set of key bindings that the SAS System uses in all SAS windows. The default value for the **SAS.keyboardTranslations** resource is determined at run time based on the vendor identification string reported by the X server that you are using as the display. These default settings are listed in the **SAS$ROOT:[TOOLS]** directory. To modify the default

bindings supplied by the SAS System, you must modify the **SAS.keyboardTranslations** resource.

Note: The X Toolkit Intrinsics translations specified in this resource apply to both the user area and the command line of all SAS windows that are affected by this resource. This resource does not affect windows that are controlled by Motif interface resources, such as the Command window, the Open or Import dialog boxes, and some other menu dialog boxes. △

To create a key definition, follow these steps:

1. Determine the keysyms for the keys that you want to define. *Keysyms* are the symbols recognized by the X Window System for each key on a keyboard. For more information, see "Determining Keysyms" on page 90.
2. Modify/add the **keyboardTranslations** resource in your resource file to include the definitions of the keys that you want to define. Use a keyboard action routine to define which action you want the key to perform. The definition in the right column in the KEYS window will no longer control the function of any keys that are defined with a keyboard action routine other than **sas-function-key()**.
3. Modify/add the **SAS.keysWindowLabels** resource in your resource file. The **SAS.keysWindowLabels** resource specifies the set of valid labels that will appear in the KEYS window. Modify this resource only if you want to add new labels or modify existing labels in the left column in the KEYS window.

   The **SAS.keysWindowLabels** resource defines only the mnemonics used in the KEYS window. For a specific key to perform an action, you must specify a **SAS.keyboardTranslations** definition for the key. For more information, see "Modifying the SAS.keyboardTranslations Resource" on page 91.
4. Start a SAS session and open the KEYS window.
5. In the right-hand column in the KEYS window, type a command name or other description of each key that you have defined.

Determining Keysyms

You can use the xev utility to determine the keysyms associated with the keys on your keyboard. Refer to your X Window documentation for more information about the xev utility.

The xev utility prints a message for each X event that occurs. The **KeyPress** event specifies the keysym for each key that is pressed.

Use the following steps to determine keysyms:

1. Start the xev utility on the X server for which you want to define keys. The xev client displays a small Event Tester window that lists the X events that occur. (The xev client generates a large amount of output, so you may want to save the output to a file for later review. You can issue the **script** command to save the output to a file.)
2. Give keyboard focus to the Event Tester window by clicking the mouse pointer on the window, if necessary.
3. Press the key that you want to define, and watch for the **KeyPress** event to be listed. The listing has a number of items that are separated by commas. One of the fields in the **KeyPress** event lists the keysym name that is associated with the key that was pressed.

   For example, when the 0 key on the keyboard is pressed, it could generate output similar to the following:

```
KeyPress event, serial 14, synthetic NO,
  window 0x4400001,root 0x23, subw 0x4400002,
```

```
time 507920400, (54,37),root:(67,66),
state 0x0, keycode 30 (keysym 0xffb0, KP_0),
same_screen YES,
XLookupString gives 1 characters: ''0''
```

In this example the keysym name is **KP_0**.

Modifying the SAS.keyboardTranslations Resource

The **SAS.keyboardTranslations** resource specifies the default set of key bindings for the SAS System to use in all SAS windows. User-specified translations for this resource may use the **#augment** or **#override** modifiers to define a previously undefined key or to override the default translation string with user preferences. This resource does not affect windows that are controlled by Motif resources, such as the Command window, the Open dialog box, and some other menu dialog boxes.

Note: Most SAS documentation uses angle brackets (<>) to indicate optional syntax. However, in this topic optional syntax is shown with square brackets ([ ]). The angle brackets that are shown are part of the syntax and should be entered exactly as shown. △

Use the following syntax for the **SAS.keyboardTranslations** resource:

SAS.keyboardTranslations: #override \

[*modifier*] <Key>*keysym* : *action-routine* \n\

[*modifier*] <Key>*keysym* : *action-routine*

where

#override
indicates that this definition should override any existing bindings for the specific keys that you define without affecting any other keys. If you omit the **#override** directive, the new bindings replace all of the default bindings, and none of the other keys on the keyboard will be available to the SAS interface to Motif. For information on the **#augment** and **#replace**, see refer to the documentation for the X Window System.

modifier
can be **Alt**, **Ctrl**, **Meta**, **Shift**, **Lock**, **Mod1**, **Mod2**, **Mod3**, **Mod4**, **Mod5**, **None**, or a blank space. The list of valid modifiers varies depending on your keyboard.

<Key>
is required. It signals the beginning of the keysym.

keysym
is the key symbol recognized by X for the key that you are defining. For more information, see “Determining Keysyms” on page 90.

action-routine
is what you want the key to do. You can specify any action routine described in “SAS Keyboard Action Names” on page 93. Some action routines require parameters.

\n
allows the X translation manager to determine where one translation sequence ends and the next one begins. Do not enter **\n** after the end of the last translation.

\
prevents the new-line character at the end of the line from being interpreted as part of the definition. This is a stylistic convention that allows each translation to

be listed on a separate line. Do not enter a backslash after the end of the last translation.

Note: The SAS System does not prevent you from specifying invalid keys in the **SAS.keyboardTranslations**resource. In some case, invalid keys will produce warnings in the terminal window. △

The following example overrides the default settings, binds the key sequence CTRL-K to the KEYS command, and binds CTRL-D to delete the character under the cursor:

```
SAS.keyboardTranslations: #override
    Ctrl<Key>k: sas-do-command(keys)\n
    Ctrl<Key>d: sas-delete-char()
```

Modifying the SAS.keysWindowLabels Resource

The **SAS.keysWindowLabels** resource specifies the set of valid labels that will appear in the KEYS window. This resource defines KEYS window internal and external mnemonics only. It is still necessary to have **SAS.keyboardTranslations** definitions for the specific X Windows keysym and modifier combinations that are bound to the **sas-function-key()** action, using the *InternalKeyName* as the **sas-function-key()** action routine parameter.

Note: Most SAS documentation uses angle brackets (<>) to indicate optional syntax. However, in this topic optional syntax is shown with square brackets ([ ]). △

Use the following syntax for the **SAS.keysWindowLabels** resource:

SAS.keyWindowLabels: \

KeyWindowLabel [(*InternalKeyName*)] \n\

KeyWindowLabel [(*InternalKeyName*)]

where

KeyWindowLabel
is the label that you want to appear in the KEYS window. This label must be one to eight characters long.

InternalKeyName
is the character string that is passed to the **sas-function-key()** action routine in the corresponding **keyboardTranslations** key binding. *InternalKeyName* is used by the SAS System to correlate KEYS window entries to key definitions in the KEYS modules loaded from SAS catalogs or defined in the KEYS window. This name must be one to eight characters long. If the *InternalKeyName* is not specified, then SAS uses the *KeyWindowLabel* as the *InternalKeyName*.

\n
allows the X translation manager to determine where one translation sequence ends and the next one begins. Do not enter **\n** after the end of the last translation.

\
prevents the new-line character at the end of the line from being interpreted as part of the definition. This is a stylistic convention that allows each translation to be listed on a separate line. Do not enter a backslash (\) after the end of the last translation.

SAS Keyboard Action Names

The SAS System declares a set of keyboard actions during X initialization. You can think of these keyboard actions as dimple functions. When the actions are executed, they act on the window that currently has keyboard input focus.

Keyboard action names represent X Toolkit action routines that are registered by the SAS interface to Motif for use with X Toolkit keyboard-event translations. A set of default keyboard actions is supplied as part of the interface. You can override and augment these actions by supplying a suitably formatted X Toolkit translation string in the **SAS.keyboardTranslations** resource (see the description of the **SAS.keyboardTranslations** resource in "Modifying the SAS.keyboardTranslations Resource" on page 91).

Note: Most SAS documentation uses angle brackets (<>) to indicate optional syntax. However, in this topic optional syntax is shown with square brackets ([ ]). The angle brackets that are shown are part of the syntax and should be entered exactly as shown. △

The following list of keyboard actions represents action routines registered by the Motif interface for use with X Toolkit keyboard event translations:

sas-cursor-down()
moves the cursor down one line in a SAS window. The cursor does not wrap when it reaches the bottom of the SAS window interior.

sas-cursor-left()
moves the cursor left one character in a SAS window. The cursor does not wrap when it reaches the left side of the SAS window interior.

sas-cursor-right()
moves the cursor right one character in a SAS window. The cursor does not wrap when it reaches the right side of the SAS window interior.

sas-cursor-up()
moves the cursor up one line in a SAS window. The cursor does not wrap when it reaches the top of the SAS window interior.

sas-delete()
deletes all text in the current field.

sas-delete-begin()
deletes all text from the current cursor position to the beginning of the current text field.

sas-delete-char()
deletes the character under the text cursor and leaves the cursor in place.

sas-delete-end()
deletes text from the current cursor position to the end of the current text field.

sas-delete-prev-char()
deletes the character to the left of the text cursor and moves the cursor back one space.

sas-delete-prev-word()
deletes text from the current cursor position to the start of the previous word. If the cursor is in the interior of a word when the action is invoked, the text from the cursor position to the beginning of the word is deleted.

sas-delete-word()
deletes text from the current cursor position to the end of the current or next word.

sas-do-command()
accepts one or more text-string parameters that are interpreted as SAS commands to be executed when the action is invoked. The action can be invoked with multiple parameters, separated by commas. The parameters are concatenated, with semicolon delimiters that are supplied by the **sas-do-command()** action. The assembled SAS command string is then submitted for execution. For example, the following translation can be used to define a global HOME, SUBMIT key sequence for all SAS windowing environment windows:

```
SAS.keyboardTranslations: <Key>KP_F3:
_sas-do-command(HOME,SUBMIT)
```

sas-function-key("*InternalKeyName*")
invokes the SAS commands associated with the function key identified by the *InternalKeyName* label. *InternalKeyName* is the character string (one to eight characters long) that is passed to the **SAS.keysWindowLabel** resource. You must enclose *InternalKeyName* in double quotation marks. For a description of internal key names, see "Defining Key Translations" on page 89. For a list of function-key parameters that SAS recognizes, refer to the sample SAS$XDEFAULTS.DAT file.

For example, the following keyboard translation designates the physical key KP_1 as a function key and associates the "Keypd 1" parameter with KP_1. The SAS System recognizes the "Keypd 1" parameter as being associated with the command LEFT (that is, scroll left) by default. Thus, it assigns the LEFT command to KP_1.

```
SAS.keyboardTranslations: ~Ctrl<Key>KP_1:
_sas-function-key("Keypd 1")
```

If you specify a function-key parameter that SAS does not recognize, the function-key command is initially left unspecified.

sas-home-cursor()
is the equivalent of the HOME command. This action is provided for convenience so that the HOME action can be defined globally.

sas-insert-char(["*InsertionString*"])
inserts or overwrites the character typed into the input field under the text cursor. Insert or overstrike behavior is determined by the **sas-toggle-insert()** action, which has a mode that is reflected by the text cursor style displayed: the block cursor indicates overstrike mode, and the underline cursor indicates insert mode. Normally, **sas-insert-char** translates the XKeyEvent into the appropriate character and inserts it at the SAS text cursor location. If you specify the parameter, the text string represented by this parameter is inserted at the SAS text cursor location. Any spaces in the string are interpreted by the X Toolkit as a parameter delimiter unless you enclose the string in double quotation marks. For information about embedding quotation marks in the string parameter, refer to your X Window System documentation. To include an escaped quotation mark, use the following syntax:

```
Shift<Key>KP_1:\
   sas-insert-char("One\\"1\\" ")
```

This produces the text string **One"1"** at the SAS text cursor location.

Note: Most SAS documentation uses angle brackets (< >) to indicate optional syntax. However, in this topic optional syntax is shown with square brackets ([ ]). The angle brackets that are shown are part of the syntax and should be entered exactly as shown. △

sas-kp-application()
sets the workstation's numeric keypad to allow function key translations to be reinstated. This toggle only works for those keypad keys that are bound to **sas-function-key()** actions. Keypad bindings to other actions are not affected by this translation.

sas-kp-numeric()
sets the workstation's keypad to generate numeric characters instead of its previous function key assignment. Note that this toggle only works for keypad keys that are bound to **sas-function-key()** actions. Keypad bindings to other actions are not affected by this translation.

sas-move-begin()
moves the cursor to the beginning of the current text field.

sas-move-end()
moves the cursor to the end of the current text field.

sas-new-line()
generates an end-of-line event when invoked. This is a context-sensitive action. If you are typing in the SAS command line, the text that you enter is submitted for execution. If you invoke the **sas-new-line** action in the SAS application client-area, then the action depends on the attributes of the text area that is under the cursor. In simplest terms, this action is the general line terminator for an input field.

sas-next-field()
advances the cursor to the next field in the SAS window client-area.

sas-next-word()
moves the cursor forward to the beginning of the next word in the current text field. Wrapping is supported; that is, if the **sas-next-word()** action does not find the beginning of a word in the current text field, it advances to the next SAS application field. However, if you are typing in the SAS command-line area of the window, the cursor will not wrap into the SAS window client-area.

sas-page-down()
scrolls the current window contents forward by one page.

sas-page-end()
moves the text cursor to the end of the current page.

sas-page-top()
moves the text cursor to the top of the current page.

sas-page-up()
scrolls the current window contents backward by one page.

sas-prev-field()
returns the cursor to the previous field in the SAS window client-area.

sas-prev-word()
moves the cursor backward to the start of the next word in the current text field. Wrapping is supported; that is, if the **sas-prev-word()** action does not find the beginning of a word in the current text field, then it returns to the end of the previous SAS application field. However, if you are typing in the SAS command-line area of the window, the cursor does not wrap into the SAS window client-area.

sas-to-bottom()
moves the text cursor to the absolute bottom of the window's text range.

sas-to-top()
moves the text cursor to the absolute top of the window's text range.

sas-toggle-insert()
toggles the line-editing behavior of the associated window between insert and overstrike modes. This action applies only to the SAS command line and to the SAS window client-area. The current mode is indicated by the style of the cursor:

block cursor
indicates overstrike mode.

underline cursor
indicates insert mode.

sas-xattr-key(<*KeyType*>[,<*KeyParam*>])
defines the extended attributes that are associated with extended-attribute keys. This action accepts two text-string parameters.

Note: Most SAS documentation uses angle brackets (< >) to indicate optional syntax. However, in this topic optional syntax is shown with square brackets ([ ]). The angle brackets that are shown are part of the syntax and should be entered exactly as shown. △

The **<*KeyType*>** parameter must be one of the following:

XAATTR
sets a display attribute. If XAATTR is specified, then the second parameter must be one of the following:

HIGHLIGHT
turns on bold.

UNDERLINE
turns on underlining.

REVERSE
turns on reverse video.

BLINK
turns on blinking.

Note: The BLINK attribute is not supported in the Motif interface. However, if you specify the BLINK attribute, it will be displayed when the catalog is ported to other operating environments. △

XACLEAR
clears all attributes. If XACLEAR is specified, then you must specify NULL as the second parameter.

XACOLOR
sets a color attribute. If XACOLOR is specified, then the second parameter must be one of the following:

BLACK
sets the text color to black.

BLUE
sets the text color to blue.

BROWN
sets the text color to brown.

CYAN
sets the text color to cyan.

GRAY
sets the text color to gray.

GREEN
sets the text color to green.

MAGENTA
sets the text color to magenta.

ORANGE
sets the text color to orange.

PINK
sets the text color to pink.

RED
sets the text color to red.

WHITE
sets the text color to white.

YELLOW
sets the text color to yellow.

For example, the following resource definition defines the CTRL-B key sequence to set the extended attribute for the color BLACK:

```
sas.keyboardTranslations:#override \
    Ctrl<Key>b:sas-xattr-key(XACOLOR,BLACK)
```

Default Keyboard Actions

Some keyboard-action routines are assigned to certain keys by default. Table 4.1 on page 97 shows the default keyboard actions, which are defined by the **SAS.keyboardTranslations** resource. For more information about this resource, see "Defining Key Translations" on page 89.

Note: Most SAS documentation uses angle brackets (< >) to indicate optional syntax. However, in this topic the angle brackets that are shown are part of the syntax and should be entered exactly as shown. △

Table 4.1 Default Key Actions

Key Name	Keyboard Action Routine
<Key>Home	**sas-home-cursor()**
<Key>osfUp	**sas-cursor-up()**
<Key>osfDown	**sas-cursor-down()**
<Key>osfRight	**sas-cursor-right()**
<Key>osfLeft	**sas-cursor-left()**
<Key>Return	**sas-new-line()**
Shift<Key>Tab	**sas-prev-field()**
<Key>Tab	**sas-next-field()**
<Key>osfBackSpace	**sas-delete-prev-char()**

Key Name	Keyboard Action Routine
<Key>osfDelete	**sas-delete-prev-char()**
<Key>	**sas-insert-char()**
Shift<Key>	**sas-insert-char()**

Extended-Attribute Key Resources

The SAS interface to Motif supports the use of attributes such as bold, reverse video, and underline. You can use the **SAS.keyboardTranslations** resource to control this feature.

Table 4.2 on page 98 summarizes the functions that are provided through the SAS System extended-attribute keys.

Note: Most SAS documentation uses angle brackets (< >) to indicate optional syntax. However, in this topic the angle brackets that are shown are part of the syntax and should be entered exactly as shown. △

Table 4.2 Functions Provided through the SAS System Extended-Attribute Keys

Keyboard Chord	Character Attribute Selected
Mod1<Key>b	Blue
Mod1<Key>r	Red
Mod1<Key>p	Pink
Mod1<Key>g	Green
Mod1<Key>c	Cyan
Mod1 <Key>y	Yellow
Mod1<Key>w	White
Mod1<Key>m	Magenta
Mod1<Key>o	Orange
Mod1<Key>k	Black
Mod1<Key>n	Brown
Mod1<Key>a	Gray
Mod1<Key>0	Clear extended attributes
Mod1<Key>1	Set highlight (bolding) attribute
Mod1<Key>2	Set underline attribute
Mod1<Key>3	Set reverse-video attribute

Customizing Fonts

You can change the SAS windowing environment font either through the Fonts dialog box or by specifying the resources in your resources file. The windowing environment font must be a fixed font.

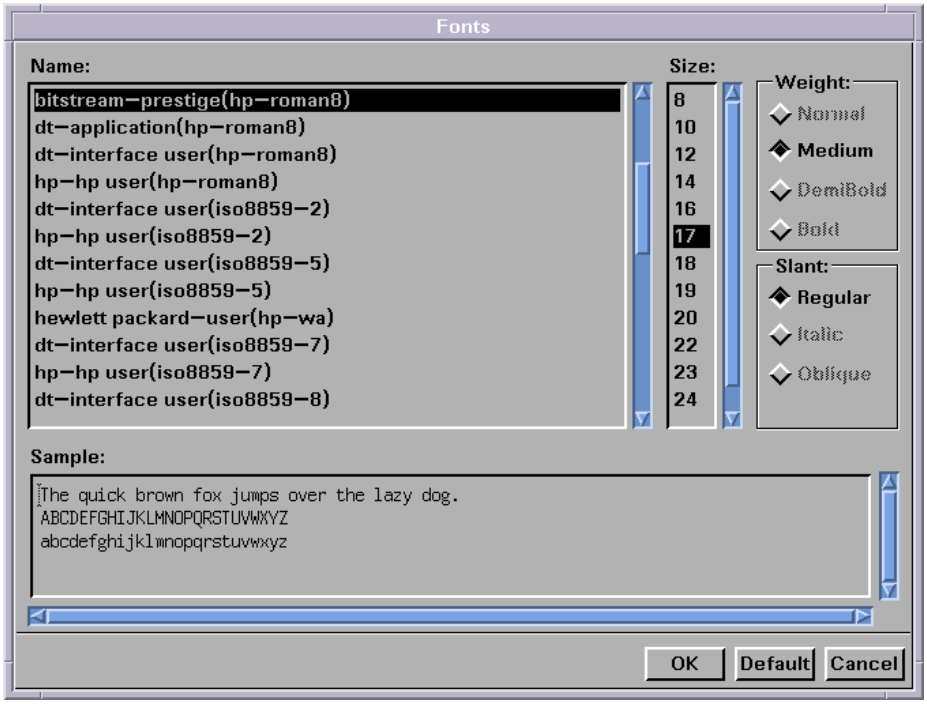
CAUTION:
It is best to change fonts before invoking any applications. Changing fonts while applications are running can result in unexpected behavior. △

Note: The SAS windowing environment font is used in SAS windows. The system font is used in most dialog boxes and menus. SAS inherits the system font defined by the **`*.systemFont`** resource. If this resource is not defined, SAS uses a Helvetica font. △

Using the Fonts Dialog Box

The Fonts dialog box (shown in Display 4.9 on page 99) enables you to change windowing environment fonts for the entire SAS session. To access this dialog box, issue the DLGFONT command in the command window, or from the **`Tools`** menu in the active window, select **`Options`** and then **`Fonts`**.

Display 4.9 Fonts Dialog Box



The default windowing environment font is the font that the X resource **`SAS.DMSfont`** specifies. For more information about the **`SAS.DMSfont`** resource, see "Specifying Font Resources" on page 100.

To change the font, select a font name and, if you want, a size, weight, and slant from the appropriate areas in the dialog box. Not all fonts are available in all sizes, weights, or slants. The **`Sample:`** field shows what the selected font looks like. After selecting a name, size, weight, and slant, click on OK to change the existing font to the selected font.

To return to the default font, click on Default.

If you are changing the font, the font that you select is stored in SASUSER.PROFILE.DMSFONT.VMSPREFS and will be used in future SAS sessions.

To cancel any changes and exit the Fonts dialog box, click on Cancel.

Specifying Font Resources

When the SAS System is initialized, it checks to see whether you have specified a font in your X resource files. If so, SAS uses that font. If SAS finds no font specification in the resource files, then it queries the server in an attempt to determine which fonts are available and which one it should use. If, based on this query, SAS cannot determine which font to use, it defaults to the fixed font.

You can customize the fonts used in the SAS windowing environment with the following resources:

SAS.DMSFont: ***font-name***
specifies the font that you want to be used as the default normal font. The font must be a monospace font. The default is dynamic, which means that the default value is determined at run time. If you change the value for the **SAS.DMSFont** resource, then you must also change the **SAS.DMSboldFont** resource to one that has the same style, set width, font size, point size, spacing, and number of pixels (or dots) per inch.

SAS.DMSboldFont: ***font-name***
specifies the font that you want to be used as the default bold font. The default is dynamic, which means that the default value is determined at run time. If you change the value for the **SAS.DMSboldFont** resource, then you must also change the **SAS.DMSFont** resource to one that has the same style, set width, font size, point size, spacing, and number of pixels (or dots) per inch.

SAS.DMSDBFont: ***font-name***
specifies the bold double-byte character-set font used by the SAS windowing environment for platforms that support the SAS DBCS user interface. The default is dynamic, which means that the default value is determined at run time.

DMSDBboldFont: ***font-name***
specifies the bold double-byte character-set font used by the SAS windowing environment for platforms that support the SAS DBCS user interface.

SAS.DMSfontPattern: ***XLFD-pattern***
specifies an X Logical Font Description (XLFD) pattern that you want SAS to use to determine the windowing environment font. Most fonts in the X Window System are associated with an XLFD, which contains a number of different fields delimited by a dash (-) character. The fields in the XLFD indicate properties such as the font family name, weight, size, resolution, and whether the font is proportional or monospaced. For more information about the XLFD and font names used with X, refer to your X Window documentation.

The *XLFD-pattern* that you specify for **SAS.DMSfontPattern** must contain the same number of fields as an XLFD. An asterisk (*) character means that any value is acceptable for that particular field. For example, the following pattern (which is the default) matches any font that has a regular slant, is not bold, is monospaced, and is an ISO 8859 font:

```
SAS.DMSfontPattern: \
  -*-*-*-r-*--*-*-*-*-m-*-iso8859-1
```

SAS uses the *XLFD-pattern* to choose a font as follows:

1. SAS queries the X server for the list of fonts that match the **SAS.DMSfontPattern** resource.
2. SAS excludes all fonts that have X and Y resolution values different from the current X display, all fonts that have variable character cell sizing (such as proportional fonts), and all fonts that have point sizes smaller than 8 points

or larger than 15 points. If this step results in an empty list, SAS chooses a generic (and usually fixed) font.

3 The font with the largest point size is chosen from the remaining list.

Given an XLFD font for the DMS font, the matching XLFD bold font name is derived and loaded. If the SAS interface to Motif cannot automatically select or load a DMS bold font, the DMS font is also used for the DMS bold font. The auto-selection of the DMS bold font is independent of the auto-selection of the DMS font. If the font resources are explicitly specified, then the auto-selection processing is not invoked. Explicitly specified values for the **SAS.DMSFont** and **SAS.DMSboldFont** resources take precedence over automatic selection.

If you have not specified a value for the **SAS.DMSboldFont** resource, then the SAS interface to Motif chooses a value by using the current DMS font as a reference point. The current DMSFont must have an XLFD name associated with it. The **SAS.DMSfontPattern** resource will have no effect if a ***Font** or ***FontList** resource is defined.

In many cases font names are aliased so that a shorter name can be used to refer to a font that has an XLFD name associated with it. The name used in determining a **SAS.DMSboldFont** is based on the *XA_FONT* font property for the DMS font.

SAS.fontPattern: ***XLFD-pattern***
specifies an XLFD font pattern that describes the candidate fonts used to resolve SAS graphics font requests. This allows the user to optimize or control the use of X fonts within the context of various SAS graphics applications. The default value of an asterisk (*****) usually does not affect performance to a significant degree. Cases where it may be appropriate to restrict the font search include X servers with an excessive number of fonts or X servers operating on performance-limited environments.

systemFont
specifies the system font. The SAS windowing environment font is used in SAS windows. The system font is used in most dialog boxes and menus. SAS inherits the system font defined by the ***.systemFont** resource. If this resource is not defined, SAS uses a Helvetica font.

How SAS Determines Which Font to Use

The SAS System determines the normal (not bold) default windowing environment font as follows:

1. If you have saved a font in SASUSER.PROFILE.DMSFONT.VMSPREFS through the Fonts dialog box, this font is used as the default normal font.
2. If you have not saved a font using the Fonts dialog box, but you have set the **SAS.DMSFont**resource, SAS uses the font specified by this resource as the default font.
3. If you have not set the **SAS.DMSFont** resource, SAS uses any ***Font** resources that you have defined.
4. If you have not set the ***Font** resources, but you have set the **SAS.DMSfontPattern** resource, SAS uses this resource to determine which font to use. The **SAS.DMSfontPattern** resource will have no effect if a ***Font** resource is defined.
5. If no resources have been set, SAS chooses a font from the fonts that are available on your server.

If you have not specified a value for the **SAS.DMSboldFont** resource, SAS uses the default normal font to determine the default bold font. If the normal **SAS.DMSFont**

resource has an XLFD name associated with it, then SAS selects the matching bold font and loads it. If SAS cannot automatically select or load a bold font, the normal font is also used for the bold font.

In many cases, font names are given aliases so that a shorter name can be used to refer to a font that has an XLFD name associated with it. The name used in determining a bold font is based on the *XA_FONT* font property for the normal font.

Specifying Font Aliases

If your server does not provide fonts to match all of those that the SAS System supplies, you can use font-alias resources to substitute the fonts that are available on your system. Use the following syntax to specify font aliases in your resource file:

```
SAS.supplied-fontAlias: substitute-fontfamily
```

where *supplied-font* is the name of the font that the SAS System supplies and *substitute-fontfamily* is the family name of the font that you want to substitute. For example, suppose that your system lacks a Palatino font, but has the following Lucinda font:

```
b&h-lucinda-bold-r-normal-sans-
    10-100-75-75-p-66-iso8859-1
```

To substitute Lucinda for Palatino, include the following line in your resource file:

```
SAS.palatinoAlias: lucinda
```

CAUTION:

Do not specify a SAS font as a font alias. A conflict can occur if you specify a font that is supplied by the SAS System as a font alias, as in the following example:

```
SAS.timesRomanAlias: symbol
```

Assigning this value to a font alias prevents the selection of any symbol fonts through the font selection dialog box, because they are specified as the Times Roman alias. △

Table 4.3 on page 102 lists the SAS font-alias resource names. All of the resources listed are of the type String and have a default of NULL.

Table 4.3 SAS Font-Alias Resources

Resource Name	Class Name
SAS.timesRomanAlias	TimesRomanAlias
SAS.helveticaAlias	HelveticaAlias
SAS.courierAlias	CourierAlias
SAS.symbolAlias	SymbolAlias
SAS.avantGardeAlias	AvantGardeAlias
SAS.bookmanAlias	BookmanAlias
SAS.NewCenturySchoolbookAlias	newCenturySchoolbookAlias
SAS.palatinoAlias	PalatinoAlias

Resource Name	Class Name
SAS.zapfChanceryAlias	ZapfChanceryAlias
SAS.zapfDingbatsAlias	ZapfDingbatsAlias

Customizing Colors

The SAS system ships to all sites a default set of colors and attribute settings for the elements of all SAS windows. Although the same setting for each element is shipped to all sites, the implementation of that setting is determined by device type, logically and physically. Thus, settings may vary among sites and across operating environments.

Optionally, the SAS Installation Coordinator can create the catalog entry SASHELP.BASE.SAS.CPARMS. The color and attribute settings in this catalog become default for the site.

As an individual user, you can customize the colors in your SAS session by using one of the following methods:

- Through the SASCOLOR window, as described in "Using the SASCOLOR Window" on page 104. You can customize any window element for most SAS windows with the SASCOLOR window.
- With the COLOR command using the following syntax:

  `color` *`field-type`* `<`*`color`* `|` `NEXT` `<`*`highlight`*`>>`

  For a complete description of the COLOR command, see "COLOR" on page 221. The COLOR command affects only the specified element of the active window. Changes made with the COLOR command override changes entered through any of the other methods described here. To save your changes beyond your current session, do one of the following:

  - Issue the WSAVE command. The changes are saved to SASUSER.PROFILE.*window*.WSAVE.

    Note: Issuing the WSAVE ALL command will save all windows that support the WSAVE command. △

  - From the **View** menu, select **Change display**, and then select **Save attributes**.

  The WSAVE command is not available for all SAS windows. For example, with SAS/FSP or SAS/CALC software, changes are saved either through the EDPARMS or the PARMS window. To determine whether WSAVE is available for a SAS window that is not part of the SAS windowing environment, refer to the product documentation.

  Both the COLOR command and the WSAVE command override actions in the SASCOLOR window. That is, COLOR and WSAVE override the use of CPARMS colors for that particular window without affecting the CPARMS values for other SAS windows.

- By entering the color resource specifications yourself. You can enter specific RGB values or color names for any of the X resources that control color. For more information, see "Defining Color Resources" on page 104.

Using the SASCOLOR Window

You can use the SASCOLOR window to change the color and highlighting of specific elements of SAS windows. To open the SASCOLOR window, do one of the following:

- From the **Tools** menu in the active window, select **Options**, and then select **Colors...**.
- Issue the SASCOLOR command in the command window.

To change a color for a window element, select the element name, and then select the color and attribute that you want assigned to the element.

Click on OK to make the desired color change. You will not see your changes in windows that are already open. To see the new settings you must close the windows and re-open them.

Click on Cancel to ignore any changes you made and close the SASCOLOR window. To return all tools in the current toolbox to their default settings, click on Defaults. When you click on Save, your changes are saved to the catalog entry SASUSER.PROFILE.SAS.CPARMS.

For more information about the SASCOLOR window, refer to the SAS online Help.

Defining Color Resources

You can define window colors and attributes by assigning values to SAS resources known as CPARMS. The CPARMS resources are divided into classes, each of which defines the color and attribute of a specific window element, such as the background in a secondary window or the border of a primary window.

Color resources fall into two categories:

foreground and background definitions
: These resources allow you to customize the RGB values that are used to define the 12 DMS colors. Since each color could be used as either a background or a foreground color, you can specify different RGB values or color names for each color for each usage. For example, you can specify that when blue is used as a foreground color that color #0046ED is used, and when blue is used as a background that CornflowerBlue is used. For more information, see “Specifying RGB Values or Color Names for Foreground and Background Resources” on page 104.

window element definitions
: These resources, which are referred to as CPARMS resources, allow you to specify which of the 12 DMS colors you want to use for each window element. For example, you can specify that message text is displayed in magenta. For more information, see “Defining Colors and Attributes for Window Elements (CPARMS)” on page 106.

The two types of resources work together. The CPARMS color values use the current foreground and background definitions. For example, the following resources specify that the background of your primary windows will be CornflowerBlue:

```
SAS.blueBackgroundColor: CornflowerBlue
SAS.cparmBackground: DmBlue
```

Specifying RGB Values or Color Names for Foreground and Background Resources

The SAS system uses **SAS.systemBackground**, **SAS.systemForeground**, and the resources listed in Table 4.4 on page 105 to determine the colors to be used in the SAS windows:

SAS.systemForeground
specifies the color for the foreground system color in the SASCOLOR window.

SAS.systemBackground
specifies the color for the background system color the SASCOLOR window.

SAS.systemSecondaryBackground
sets the system secondary background color and specifies the color for the secondary background system color in the SASCOLOR window.

You can specify color names such as MediumVioletRed or RGB values such as #0000FF for all of the foreground and background resources. For more information about RGB color values, refer to your X Window System documentation.

Table 4.4 on page 105 lists all of the foreground and background color resources and their class names. All of these resources are of the type String.

Table 4.4 Foreground and Background Color Resources

Resource Name	Class Name
SAS.systemForeground	SystemForeground
SAS.systemBAckground	SystemBackground
SAS.systemSecondaryBackground	Background
SAS.blackForegroundColor	BlackForegroundColor
SAS.blueForegroundColor	BlueForegroundColor
SAS.brownForegroundColor	BrownForegroundColor
SAS.cyanForegroundColor	CyanForegroundColor
SAS.grayForegroundColor	GrayForegroundColor
SAS.greenForegroundColor	GreenForegroundColor
SAS.magentaForegroundColor	MagentaForegroundColor
SAS.orangeForegroundColor	OrangeForegroundColor
SAS.pinkForegroundColor	PinkForegroundColor
SAS.redForegroundColor	RedForegroundColor
SAS.whiteForegroundColor	WhiteForegroundColor
SAS.yellowForegroundColor	YellowForegroundColor
SAS.blackBackgroundColor	BlackBackgroundColor
SAS.blueBackgroundColor	BlueBackgroundColor
SAS.brownBackgroundColor	BrownBackgroundColor
SAS.cyanBackgroundColor	CyanBackgroundColor
SAS.grayBackgroundColor	GrayBackgroundColor
SAS.greenBackgroundColor	GreenBackgroundColor
SAS.magentaBackgroundColor	MagentaBackgroundColor
SAS.orangeBackgroundColor	OrangeBackgroundColor
SAS.pinkBackgroundColor	PinkBackgroundColor
SAS.redBackgroundColor	RedBackgroundColor

Resource Name	Class Name
SAS.whiteBackgroundColor	WhiteBackgroundColor
SAS.yellowBackgroundColor	YellowBackgroundColor

Defining Colors and Attributes for Window Elements (CPARMS)

You specify CPARMS resources just as you would specify any other SAS X resource in an X resource file. You specify CPARMS values either in your resource file or with the **-xrm** option, which can be used as a system option or on the SAS command line. (The **-xrm** option is also described with the SAS system option "XRESOURCES=" on page 456.)

Use the following syntax to specify CPARMS values:

```
SAS.cparmResource: DmColorName|DmAttrName\
[+ DmColorName|DmAttrName]
```

Resource can be any of the CPARMS resources listed in "Defining the CPARMS Resource Set" on page 106 All of these resources are of the type DmColor, and their default values are dynamic—that is, the default values are determined at run time.

DmColorName can be any one of the following colors:

DmBLACK

DmBLUE

DmBROWN

DmCYAN

DmGRAY

DmGREEN

DmMAGENTA

DmORANGE

DmPINK

DmRED

DmWHITE

DmYELLOW

DmAttrName can be any one of the following attributes:

DmHIGHLIGHT

DmUNDERLINE

DmREVERSE

You can specify attribute names and color names in any order. If you specify more than one color name for a single class, then SAS uses the last color that you specified.

Defining the CPARMS Resource Set

The CPARMS resource set contains the following values:

SAS.cparmBackground
specifies the color and attribute settings for backgrounds within all primary windows displayed in a SAS session. The default is DmWhite.

SAS.cparmBanner
specifies the color and attribute settings for a banner within a window. The banner is the command line prompt. The default is DmBlack.

SAS.cparmBorder
specifies the color and attribute settings for the border of a primary window. The default is DmBlack.

SAS.cparmByline
specifies the color and attribute settings for BY lines written to the Output window. The default is DmBlue.

SAS.cparmColumn
specifies the color and attribute settings for text labels for column information. You can use this resource within the SAS editor to identify editing lines and in spreadsheet windows to label spreadsheet. The default is DmBlue/Underline.

SAS.cparmCommand
specifies the color and attribute settings for the command data entry field when menus are disabled. The default is DmBlack.

SAS.cparmData
specifies the color and attribute settings for general lines written to the Log window or the Output window. The default is DmBlack.

SAS.cparmError
specifies the color and attribute settings for ERROR: lines that are written to the Log window or the Output window. The default is DmRed.

SAS.cparmFootnote
specifies the color and attribute settings for FOOTNOTE lines written to the Output window. The default is DmBlue.

SAS.cparmForeground
specifies the color and attribute settings for all text fields within a SAS windowing environment window that can be edited. The default is DmBlack.

SAS.cparmHeader
specifies the color and attribute settings for HEADER lines written to the Output window. The default is DmBlue.

SAS.cparmHelpLink
specifies the color and attribute settings for links to additional levels of information in the HELP system. The default is DmGreen/Underline.

SAS.cparmHelpMainTopic
specifies the color and attribute settings for topic words or phrases in the HELP system. The default is DmBlack.

SAS.cparmHelpSubTopic
specifies the color and attribute settings for topic words or phrases in the HELP system. The default is DmBlack.

SAS.cparmInfo
specifies the color and attribute settings for text that is displayed in a window as an aid to the user, for example:

```
Press Enter to continue
```

The default is DmBlack.

SAS.cparmLabel
specifies the color and attribute settings for text that precedes a widget. For example, the text **Name:** in the following example is a label:

```
Name: _____________________
```

The default is DmBlack.

SAS.cparmMark
specifies the color and attribute settings for areas that have been selected for operations such as FIND, CUT, and COPY. The default is DmBlack/DmReverse.

SAS.cparmMessage
specifies the color and attribute settings for the message field. The default is DmRed.

SAS.cparmNote
specifies the color and attribute settings for NOTE: lines that are written to the Log window or the Output window. The default is DmBlue.

SAS.cparmNumber
specifies the color and attribute settings for line numbers. The default is DmBlue.

SAS.cparmSecondaryBackground
specifies the color and attribute settings for backgrounds in secondary windows. Secondary windows are dialog boxes or requestor windows that prompt the user. The default is DmGray.

SAS.cparmSecondaryBorder
specifies the color and attribute settings for the border of a secondary window. The default is DmBlack.

SAS.cparmSource
specifies the color and attribute settings for SAS source lines that are written to the Log window. The default is DmBlack.

SAS.cparmText
specifies the color and attribute settings for text labels for row information. You can use this resource within the SAS editor to identify editing lines and in spreadsheet windows to label spreadsheet rows. The default is DmBlue.

SAS.cparmTitle
specifies the color and attribute settings for TITLE lines written to the Output window. The default is DmBlue.

SAS.cparmWarning
specifies the color and attribute settings for WARNING lines written to the Log window or the Output window. The default is DmGreen.

Table 4.5 on page 108 lists the resource name, SASCOLOR window element, class name, and default for each CPARMS resource. All the resources are of the type DmColor and their default values are dynamic—that is, the default values are determined at run time.

Table 4.5 SAS CPARMS Resources

Resource Name	SASCOLOR Window Element	Class Name	Default Color
SAS.cparmBackground	Background	CparmBackground	DmWhite
SAS.cparmBanner	Banner	CparmForeground	DmBlack
SAS.cparmBorder	Border	CparmBackground	DmBlack
SAS.cparmByline	Byline	CparmForeground	DmBlue

Resource Name	SASCOLOR Window Element	Class Name	Default Color
SAS.cparmColumn	Column Label	CparmForeground	DmBlue/ Underline
SAS.cparmCommand	Command	CparmForeground	DmBlack
SAS.cparmData	Data	CparmForeground	DmBlack
SAS.cparmError	Error	CparmForeground	DmRed
SAS.cparmFootnote	Footnote	CparmForeground	DmBlue
SAS.cparmForeground	Foreground	CparmBackground	DmBlack
SAS.cparmHeader	Header	CparmForeground	DmBlue
SAS.cparmHelpLink	Help Link	CparmForeground	DmGreen/ Underline
SAS.cparmHelpMainTopic	Help Main Topic	CparmForeground	DmBlack
SAS.cparmHelpSubTopic	Help Subtopic and Syntax	CparmForeground	DmBlack
SAS.cparmInfo	Informational Text	CparmForeground	DmBlack
SAS.cparmLabel	Label	CparmForeground	DmBlack
SAS.cparmMark	Selected Area	CparmForeground	DmBlack/ DmReverse
SAS.cparmMessage	Message	CparmForeground	DmRed
SAS.cparmNote	Note	CparmForeground	DmBlue
SAS.cparmNumber	Line Numbers	CparmForeground	DmBlue
SAS.cparmSecondaryBackground	Secondary Background	CparmForeground	DmGray
SAS.cparmSecondaryBorder	Secondary Border	CparmForeground	DmBlack
SAS.cparmSource	Source	CparmForeground	DmBlack
SAS.cparmText	Row Label	CparmForeground	DmBlue
SAS.cparmTitle	Title	CparmForeground	DmBlue
SAS.cparmWarning	Warning	CparmForeground	DmGreen

Example: Defining CPARMS

The following example specifies that all background colors are gray and all foreground colors are black:

```
SAS.cparmBackground: DmGRAY
SAS.cparmForeground: DmBLACK
```

You can add to these settings to make errors red with reverse video and warnings yellow with bolding and reverse video:

```
SAS.cparmError: DmRED + DmREVERSE
SAS.cparmWarning:
    DmHIGHLIGHT + DmYELLOW + DmREVERSE
```

You can further add to these settings by making the background of secondary windows white:

```
SAS.cparmSecondaryBackground: DmWHITE
```

Controlling Color Contrast

During interactive move/stretch operations, such as rubberbanding and dragging rectangles in SAS/INSIGHT software, you may find it hard to see the outline of the graphics primitive because of the lack of contrast between the primitive and the background. The XCONTRAST command makes the primitive visible against the background. The rendering performance and the aesthetic appearance of the primitive is compromised for the sake of visibility. You can enter XCONTRAST to act as a toggle, or you can specify XCONTRAST ON or XCONTRAST OFF.

With some color combinations, text fields, push buttons, and check boxes and other foreground categories may not be visible. The following resource makes these categories legible:

SAS.dmsContrastCheck: [True | False]
controls whether contrast mapping is applied to non-graphic foreground colors in a SAS window. The value True specifies that DMS foreground colors will be remapped if necessary to produce a contrast. The default value is False, which disables contrast checking. Some color usage based on graphic operations is not affected by this resource.

Controlling Menus

Pull-down menus are controlled by the following resources:

SAS.pmenuOn: [True | False]
turns the menus on for the current SAS session, regardless of the information stored by the WSAVE command. You can use this resource to override the saved settings in your SASUSER.PROFILE catalog. The WSAVE state of an individual window takes precedence over the global state. The default is True. You can also use the PMENU ON and PMENU OFF commands to turn menus on and off.

SAS.usePmenuMnemonics: [True | False]
specifies whether mnemonics are attached to the menus for the current SAS session. The default is True.

Customizing Cut-and-Paste Operations

The SAS interface to Motif uses the SAS concepts of marks and the paste buffer to provide the normal X Window System cut-and-paste behavior. The mark defines and highlights a region of text. The paste buffer retains a collection of text lines. In the SAS System, you must associate marks with a paste buffer by issuing a command such as STORE or CUT. This is different from other X applications, in which marks and paste are generally synonymous with each other.

You can do any of the following tasks with a marked area of text in many SAS windows:

- delete the text.
- delete and retain the text in a paste buffer.
- search the text for words or phrases.
- store the text in a paste buffer.
- use the text in an application such as SAS/CALC software to define a set of values that are operated on by a specified command.

□ submit the marked text only.

Usually, a mark defines the text to be operated on by a STORE or CUT command that places the text into a named SAS paste buffer.

The default definitions enable you to cut and paste between the SAS interface to Motif and other X window clients or between the SAS interface to Motif and the XPRIMARY buffer in OpenVMS.

The following sections explain the parameters that are associated with cut-and-paste operations and their default definitions. You can use these parameters to customize cut-and-paste operations on your system.

Marking Text

The SAS System supports the character mark and the block mark. The behavior of the character mark resembles text marking in most X Window System terminal emulators. The range of the *character mark* spans whole interior lines between the starting point and the ending point of the marked text. Interior lines are those that are between the lines containing the starting point and the ending point. By contrast, a *block mark* is a rectangular region that includes one corner at the starting point and one corner at the ending point of the mark.

The SAS interface to Motif supports two methods of marking text. One method uses the MARK command. The other method uses the mouse to click and drag; the marks that are generated by this second method are called *drag marks*.

Using the MARK Command to Mark Text

The MARK command establishes a mode in the SAS window that lasts until you issue another MARK command.

Position the cursor by clicking at the beginning of the text that you want to mark. Issue the MARK command. Then, move the cursor to the end of the text that you want to mark and click. Issue the MARK command a second time. Before you issue the second MARK command, the end point changes as you move the window's text cursor by using the keyboard arrow keys, tabbing, clicking the mouse, or scrolling through the window's contents. Mark highlighting also changes as you change the cursor position.

You can issue the MARK command from the command line, or you can assign it to a function key. If you want to select a rectangular block of text instead of a string of text, add the BLOCK argument to the MARK command. With the MARK command, you can select more than one area of text in the same window at the same time. To unmark the selected text, issue the UNMARK command.

Using the Mouse to Mark Text

You can also use the mouse to select text by clicking and dragging the left mouse button (MB1) inside the SAS window. You can use keyboard modifiers to change the behavior of the marking. When you end a drag mark by releasing MB1, the SAS System performs an end-of-mark action that may generate a STORE command to save the contents of the mark into a SAS paste buffer. This feature is controlled by the **`SAS.markPasteBuffer`** resource. You can clear marks in a SAS window by clicking MB1 inside the SAS window or by starting a new drag mark in the SAS window.

The SAS interface to Motif supports the following mouse button behavior:

MB1 press and move
makes a SAS character mark starting at the point where the mouse button is depressed and marks an area with the mouse. The area that you mark is highlighted. Release MB1 to complete the mark.

Note: Any existing marks in the same window are deleted when you press MB1. △

MB1 press and release
frees existing drag marks in the window except those that were created by the MARK command. You can use MB1 to position the text cursor inside a SAS mark.

MB2 press
generates a PASTE command at the location of the click, using the **SAS.markPasteBuffer** value as the name of the paste buffer if the resource is defined. If **SAS.markPasteBuffer** is not defined, press MB2 to generate a PASTE command with BUFFER=DEFAULT.

There are three modifier keys that may be used in conjunction with MB1 to produce the following results:

Unmodified
character mark and end-of-mark action

Shift
extend mark and end-of-mark action

CTRL
block mark and end-of-mark action

Mod1
alters end-of-mark action to the opposite of normal behavior.

The normal end-of-mark action depends on the setting of the **SAS.markPasteBuffer** resource. If this resource is defined, the normal action is to generate a STORE BUFFER=<*name*> command to store the newly created mark to the named paste buffer. If this resource is not defined, the normal action is to suppress the generation of the STORE command; the marked area remains highlighted.

Paste Buffers

SAS *paste buffers* are named objects that retain a copy of selected text. Each buffer is identified by a name of up to eight characters; the name is not case sensitive. Most commands operating on a SAS paste buffer support the BUFFER= option that enables you to identify the paste buffer. This paste buffer name directs the SAS interface to Motif to interact with the standard X inter-client data exchange mechanisms.

Paste buffers that are not associated with an X inter-client mechanism are called *local paste buffers* because their contents are known only within the scope of the SAS session. Paste buffers associated with X inter-client data exchange mechanisms are called *extended paste buffers*.

The SAS interface to Motif enables you to use X cut buffers and X selections to exchange information with other X clients. The paste buffer name determines whether the buffer has extended semantics in the context of the X data exchange mechanisms. The following list describes paste buffer names and their associations:

XPRIMARY
is the paste buffer associated with the X primary selection (PRIMARY).

XSCNDARY
is the paste buffer associated with the secondary selection (SECONDARY).

XCLIPBRD
is the paste buffer associated with the clipboard selection (CLIPBOARD). This paste buffer allows you to use the MIT X Consortium **xclipboard**client with the SAS System.

XTERM
is the paste buffer associated with the exchange protocol used by the *xterm* client. XTERM is the default buffer. DEFAULT is an alias for XTERM. If you copy or cut text into the XTERM buffer, the text is actually copied or cut into all four of the past buffers. When you paste text from the XTERM buffer, the text is pasted from the XPRIMARY buffer.

XCUT*n*
is the paste buffer associated with the X cut buffer *n*, where the range of *n* is 0 to 7.

You can use an alias to associate the DEFAULT paste buffer to another SAS paste buffer name. In this way, operations that would otherwise occur in the context of the local DEFAULT paste buffer can use a paste buffer that is specific to X. This is most useful in the context of SAS menu operations that generate explicit STORE, CUT, and PASTE commands that make use of the DEFAULT paste buffer. Use the **SAS.defaultPasteBuffer** resource to define an alias for the DEFAULT paste buffer.

Resources can be defined to associate SAS paste buffers with the X11R5 version of xterm, a terminal emulator client based on X. To define an alias that associates the SAS DEFAULT paste buffer with the XTERM paste buffer, declare the following X resource for the SAS application:

```
SAS.defaultPasteBuffer: XTERM
```

Defining an alias that associates the DEFAULT paste buffer with the XTERM paste buffer enables you to copy and paste text between xterm windows and SAS windows. You can select the range of text in the xterm window with MB1 and use MB2 to paste the text into the SAS window. You can use MB1 to select a range of text in a SAS window and use MB2 to paste the text into an xterm window. When marked text in the SAS System is stored to a SAS paste buffer, the mark is released, and the text highlighting clears. This behavior is different from most X clients, in which the highlighting is retained as long as the client continues to own the associated X selection.

When one SAS session communicates with another SAS session, the text and the original text attributes, such as color, are preserved where possible when transfers are made via an X selection conversion. For example, if you have associated an XTERM alias with the **SAS.markPasteBuffer** and **SAS.defaultPasteBuffer** resources, and then you copy and paste text between two SAS sessions, the color and character attributes are preserved. However, if you copy and paste the same text into an xterm window while using the vi editor, you will not retain the SAS text color and attribute information. A subsequent paste of this selection into a SAS session window will contain text only with a default text color of WHITE and no additional attributes. Extended paste buffers that use X selection transfers include XPRIMARY, XSCNDARY, and XTERM. If you change the definition for **SAS.defaultPasteBuffer** and **SAS.markPasteBuffer** to XCUT0, you will not retain the text and color attributes when copying and pasting text between two SAS sessions.

If you use the XCLIPBRD extended paste buffer, you can use the MIT xclipboard client with the SAS System for retained-mode cut-and-paste conversions through the ICCCM CLIPBOARD selection protocol. Define one or both of these SAS resources as follows:

```
SAS.defaultPasteBuffer: XCLIPBRD
SAS.markPasteBuffer: XCLIPBRD
```

SAS commands that reference the DEFAULT paste buffer are associated with the X CLIPBOARD selection; mouse-based SAS marks will automatically be stored in the XCLIPBRD paste buffer.

If you want to use the CLIPBOARD selection only for some cut-and-paste interactions, you can issue the following KEYS window commands for selected windows:

```
STORE BUFFER=XCLIPBRD
PASTE BUFFER=XCLIPBRD
```

Note: When you use the xclipboard client, SAS text attributes are not preserved in exchanges made between SAS sessions. However, when you use the SAS XCLIPBRD paste buffer without a clipboard manager such as the xclipboard client, SAS text attributes are preserved in exchanges between SAS sessions. △

Using Paste Buffers to Manipulate Text

If you want SAS to automatically copy selected text into your paste buffer every time you mark a region of text with the mouse, you should specify your paste buffer name in the **SAS.markPasteBuffer** resource. To generate a STORE command every time you mark a region of text with the mouse, define the following X resource for the SAS application:

```
SAS.markPasteBuffer: XTERM
```

Since the DEFAULT paste buffer is aliased to XTERM, you could also make the following declaration for **SAS.markPasteBuffer** to produce the same result:

```
SAS.markPasteBuffer: DEFAULT
```

The **markPasteBuffer** definition causes SAS to automatically issue a STORE command whenever you select text.

The STORE command, as well as the CUT and PASTE commands, support a BUFFER= option that specifies which buffer to use. When these commands are issued from function keys or menus whose definitions do not include the BUFFER= option, if the **SAS.markPasteBuffer** resource is not defined, these commands use BUFFER=DEFAULT. If this resource is defined, these commands use BUFFER=*buffer-name*.

You can customize your normal cut, copy, or paste keys to issue any of these commands with the BUFFER= option. For example, you can define a SAS **SAS.keyboardTranslations** binding for the **sas-do-command()** action that will be valid for the same set of operations in every SAS window and override the SAS **SAS.keyboardTranslations** definition for the **osfCopy** and **osfPaste** keys with the following specifications:

```
SAS.keyboardTranslations: #override
<Key>osfCopy:
     sas-do-command("STORE BUFFER=XCLIPBRD") \n
<Key>osfPaste:
     sas-do-command("PASTE BUFFER=XCLIPBRD")
```

For more information about customizing keys, see “Customizing Key Definitions” on page 89.

When you cut or copy and paste text between SAS session using the XTERM, XPRIMARY, or XSCNDARY paste buffers, the color and attribute information is preserved. However, if you copy and paste the same text into an xterm window while using the vi editor, the color and attribute information is lost. If you change the definition for **SAS.defaultPasteBuffer** and **SAS.markPasteBuffer** to XCUT0, then you will not retain the text and attributes when you copy and paste text between two SAS sessions.

Note: When you use the xclipboard client, SAS text attributes are not preserved in exchanges made between SAS sessions. However, when you use the SAS XCLIPBRD paste buffer without a clipboard manager such as the xclipboard client, SAS text attributes are preserved in exchanges between SAS sessions. △

Using Paste Buffers for Information Exchange

You can use X Window paste buffers and X selections to exchange information with other X Window clients. The SAS System paste-buffer interface allows this interaction for all paste-buffer interaction commands and operations. With the SAS interface to Motif, you can use the following paste buffers:

- XPRIMARY
- XSCNDARY
- XCLIPBRD
- XTERM
- XCUT*n*

For more information about these paste buffers, see "Paste Buffers" on page 112.

If you are not sure which X data exchange protocols your other X clients are using, you should use the XTERM paste buffer. You can specify your default paste buffer with the **SAS.defaultPasteBuffer** resource:

```
SAS.defaultPasteBuffer: XTERM
```

If you know that the X clients in your workstation environment all use the X PRIMARY selections to exchange data, you should use the XPRIMARY paste buffer:

```
SAS.defaultPasteBuffer: XPRIMARY
```

This specification uses both SAS and X resources more efficiently and provides for the on-demand transfer of data between clients.

You can also use the SAS XCLIPBRD paste buffer to interact with Motif clients that use the Motif clipboard mechanism for text exchanges. This clipboard mechanism makes it unnecessary to have a dedicated client such as xclipboard. For example, you can use XCLIPBRD to exchange text directly with the Motif xmeditor application when you select the **Cut**, **Copy**, or **Paste** items from the xmeditor **Edit** menu.

Customizing Session Workspace, Session Gravity, and Window Sizes

The SAS System uses the following resources to determine the size of the session workspace, the gravity of the workspace, and the size of the windows.

Note: SAS requires at least 20 rows and 78 columns of internal window space. If you specify smaller dimensions for the **SAS.maxWindowHeight**, **SAS.maxWindowWidth**, **SAS.windowHeight**, or **SAS.windowWidth** resource, then SAS defaults to these minimum values. △

SAS.awsResizePolicy
controls the policy for resizing the application workspace (AWS) windows as interior window are added and removed. Possible values include the following:

grow
the AWS window will attempt to grow any time an interior window is grown or moved, in order to show all interior windows, but it will not shrink to removed dead areas.

fixed
the AWS window will attempt to size itself to the size of the first interior window and will not attempt any further size changes.

The default is grow.

SAS.maxWindowHeight
specifies the maximum height of a window. The unit of measure is specified by the **SAS.windowUnitType** resource. The default is 95.

SAS.maxWindowWidth
specifies the maximum width of a window. The unit of measure is specified by the **SAS.windowUnitType** resource. The default is 95.

SAS.noAWS: [True | False]
controls whether each of your application's windows appears in its own native window rather than in an application workspace (AWS). The default is False, which confines all windows displayed by an application to a single AWS window.

SAS.scrollBarSize
specifies the size of the scroll bars in pixels. The default is 17.

SAS.sessionGravity
specifies in which region of the screen the SAS System attempts to place its windows. Possible values include the following:

NorthWestGravity

NorthGravity

NorthEastGravity

WestGravity

CenterGravity

EastGravity

SouthWestGravity

SouthGravity

SouthEastGravity

The default is dynamic, which means that the default value is determined at run time. For example, for the first SAS session that you invoke the default is NorthWestGravity. However, if you invoke a second SAS session while the first session is still running, the default for the second session is NorthEastGravity.

SAS.sessionGravityXOffset
specifies an X (horizontal) offset to be added when the SAS System attempts to place a window in the gravity region. The default is 0.

SAS.sessionGravityYOffset
specifies a Y (vertical) offset to be added when the SAS System attempts to place a window in the gravity region. The default is 0.

SAS.windowHeight
specifies the default height of a window. The unit of measure is specified by the **SAS.windowUnitType** resource. The default is 50.

SAS.windowUnitType
specifies the unit of measure for the **SAS.windowWidth**, **SAS.windowHeight**, **SAS.maxWindowWidth**, and **SAS.maxWindowHeight** resources. Possible values include the following:

character
specifies the number of rows and columns.

pixel
specifies the number of pixels.

percentage
specifies the percentage of the display.

The default is percentage.

SAS.windowWidth
specifies the default width of a window. The unit of measure is specified by the **SAS.windowUnitType** resource. The default is 67.

Specifying User-Defined Icons

You can add your own icons to those icons that are supplied with the SAS System. For example, if you want to use your own color icons in the ToolBox, define the **SAS.colorUiconPath**, **SAS.colorUiconCount**, and **SAS.sasUiconx** resources. then, when you are defining tools in the Tool Editor, the Tool Editor will include your icons in the display of icons that you can choose for each tool.

SAS.colorUiconCount
specifies the number of user-defined color icons that are available for the SAS System to use. The default is 0.

SAS.colorUiconPath
specifies the file search path for locating user-defined color icon files. The default is NULL. This string resource may contain multiple directory paths to be searched for a SAS icon file stored in X PixMap (.XPM) format. Use a comma to separate individual directory path elements in the icon path string. For example, the following string first searches for icon files in the **MYDISK:[MYDIR1.PIXMAPS]** directory, then searches in the **MYDISK:[MYDIR2.PIXMAPS]** directory:

```
SAS.colorUiconPath : MYDISK:[MYDIR1.PIXMAPS],
                     MYDISK:[MYDIR2.PIXMAPS]
```

SAS.uiconCount
specifies the maximum number of user-defined icons that are available for use in the SAS System. The default is 0.

SAS.uiconPath
specifies the search path for locating bitmap files for user-defined icons. The default is NULL. This string resource can contain multiple directory paths to be searched for the existence of a particular icon file. These files are assumed to be in the X11 XYBitmap format. A comma delimits individual directory-path elements in the icon-path string.

For example, the following string first searches for bitmap files in the MYDISK:[MYDIR1.BITMAPS] directory, then in the MYDISK:[MYDIR2.BITMAPS] directory:

```
SAS.uiconPath: MYDISK:[MYDIR1.BITMAPS],
               MYDISK:[MYDIR2.BITMAPS]
```

The directory-path elements must contain the complete path syntax.

SAS.sasUicon*x*
associates a value with the filename of an X bitmap or pixmap file. The value of *x* is a number assigned to the file. The default is NULL. A file extension of .XBM or .XPM is automatically supplied.

The resource name used to locate the icon bitmap filename for user icon number *x* is **SAS.sasUicon*x***. For example, to define the filename MYICON for the user icon 1, you should define the resource:

```
SAS.sasUicon1: myicon
```

If the resource name is not defined, SAS generates a filename of the form **sasui*nnn*.xbm**or **sasui*nnn*.xpm**. The path elements from the **SAS.uiconPath** resource are searched in sequence until the icon file is found or until the search path is exhausted.

For example, the following set of X resources defines a collection of color icons:

```
SAS.colorUiconPath: MYDISK:[MYDIR1.PIXMAPS]
SAS.colorUiconCount: 7
SAS.sasUicon1: adsetup
SAS.sasUicon2: adverse
SAS.sasUicon3: altmenu
SAS.sasUicon4: batch
SAS.sasUicon5: is
SAS.sasUicon6: patgrps
SAS.sasUicon7: pctchg
```

The Motif interface will search for icon **sasUicon1**in a file named MYDISK:[MYDIR1.PIXMAPS]ADSETUP.XPM.

Miscellaneous Resources

You can also customize the following resources:

SAS.altVisualId: ***ID***
specifies a visual type ID. The default is NULL.

SAS.autoSaveInterval: ***minutes***
specifies how often (in minutes) that the data from the Program Editor window should be saved. The default is 10.

SAS.autoSaveOn: [True | False]
specifies that data from the Program Editor window should be saved to a file at intervals specified by the **SAS.autoSaveInterval** resource. The default is True.

SAS.confirmSASExit: [True | False]
controls whether SAS displays the Exit dialog box when you issue the DLGENDR command or when you select Exit from the File menu. The default is True.

SAS.defaultCommandWindow: [True | False]
specifies whether the command window is opened when you start your SAS session. The default is True.

SAS.directory: ***directory-pathname***
specifies the directory that you want when you first open a file selection dialog box. By default, the Open dialog box uses the current directory.

SAS.graphicsClipboardPath: ***directory-name***
specifies which directory to place the GSTORE temporary file in. The default is **SAS$WORKLIB:**. This directory is usually the best place to store the temporary

file. To cut and paste graphs and images between SAS sessions, the setting of this resource must be identical in both sessions, and the path must be accessible to the systems on which the sessions are running.

SAS.helpBrowser: ***pathname***
specifies the pathname of the World Wide Web browser that you want to use for viewing the online help. The default browser is Netscape.

SAS.insertModeOn: [True | False]
controls the editing mode in SAS editor windows, either insert or overtype. The default is False (overtype).

SAS.noDoCommandRecall
specifies that SAS commands submitted through the **sas-do-command()** Xt action routine should not be recorded in the command recall buffer. The default value, True, causes these commands to be recorded.

SAS.pattern: ***default-pattern***
specifies the default pattern that you want to be used as the file filter when you first invoke the Open dialog box. This pattern is displayed in the text field at the top of the dialog box. By default, the Open dialog box uses the first filter in the **File type** list. The **SAS.pattern** resource has no effect on the **File type** field.

SAS.selectTimeout
specifies how long (in seconds) the SAS System waits for the completion of a request to convert an X Toolkit selection. The default value, 60, is adequate in most cases.

SAS.startupLogo: [*xpm-filename* | None | ""]
specifies the .XPM file that you want the SAS System to display when it is initialized. If the string is empty (" ", which is the default), the SAS System uses the default logo.

SAS.webBrowser: ***pathname***
specifies the pathname of the web browser that you want invoked when the WBROWSE command is issued. The default browser is Netscape.

SAS.wsaveAllExit: [True | False]
specifies whether SAS should issue the WSAVE ALL command when you end your session. This command saves the global settings, such as window color and window position, that are in effect for all windows that are currently open. The default is False.

Summary of X Resources for the SAS System

Table 4.6 on page 120 lists the resource and class names, type, and statically defined default values for many of the SAS X resources. For additional information about specific types of resources, see the following tables:

- "SAS Font-Alias Resources," Table 4.3 on page 102
- "Foreground and Background Color Resources," Table 4.4 on page 105
- "SAS CPARMS Resources," Table 4.5 on page 108

If no X resource value overrides the resource and class name, then the static default is chosen. Some X resources have a default of *dynamic* listed. This means that the default value is determined at run time.

An online example resource file is available for use with Version 8 of the SAS System. The filename is SAS$ROOT:[TOOLS]SAS$XDEFAULTS.DAT.

Note: X resource names are case-sensitive. When you specify them in your resource files, be sure to use the correct capitalization. △

Table 4.6 X Resources Used by the SAS Interface to Motif

Resource Name	Class Name	Type	Default
SAS.altVisualId	AltVisualId	int	NULL
SAS.autoSaveInterval	AutoSaveInterval	int	10
SAS.autoSaveOn	AutoSaveOn	Boolean	True
SAS.awsResizePolicy	AWSResizePolicy	String	grow
SAS.colorUiconCount	UiconCount	int	0
SAS.colorUiconPath	UiconPath	String	NULL
SAS.confirmSASExit	ConfirmSASExit	Boolean	True
SAS.defaultCommandWindow	DefaultCommandWindow	Boolean	True
SAS.defaultPasteBuffer	DefaultPasteBuffer	String	XPRIMARY
SAS.defaultToolBox	DefaultToolBox	Boolean	True
SAS.directory	Directory	String	NULL
SAS.DMSboldFont	Font	String	*dynamic*
SAS.dmsContrastCheck	DmsContrastCheck	Boolean	False
SAS.DMSDBboldFont	DBFont	String	*dynamic*
SAS.DMSDBFont	DBFont	String	*dynamic*
SAS.DMSFont	Font	String	*dynamic*
SAS.DMSfontPattern	DMSFontPattern	String	-*-*-*-r-*–*-*-*-*-m-*-iso8859-1
SAS.fontPattern	FontPattern	String	*
SAS.graphicsClipboardPath	GraphicsClipboardPath	String	SAS$WORKLIB:
SAS.helpBrowser	helpBrowser	String	netscape
SAS.insertModeOn	InsertModeOn	Boolean	False
SAS.isToolBoxPersistent	IsToolBoxPersistent	Boolean	True
SAS.keyboardTranslations	KeyboardTranslations	Translation	*dynamic*
SAS.keysWindowLabels	KeysWindowLabels	String	*dynamic*
SAS.markPasteBuffer	MarkPasteBuffer	String	NULL
SAS.maxWindowHeight	WindowHeight	Dimension	95
SAS.maxWindowWidth	WindowWidth	Dimension	96
SAS.noAWS	NoAWS	Boolean	False
SAS.noDoCommandRecall	NoDoCommandRecall	Boolean	True
SAS.pattern	Pattern	String	NULL
SAS.pmenuOn	PmenuOn	Boolean	True
SAS.scrollBarSize	ScrollBarSize	Dimension	17

Resource Name	Class Name	Type	Default
SAS.selectTimeout	SelectTimeout	int	60
SAS.sessionGravity	SASGravity	String	*dynamic*
SAS.sessionGravityXOffset	SASGravityOffset	int	0
SAS.sessionGravityYOffset	SASGravityOffset	int	0
SAS.startupLogo	StartUpLogo	String	" "
SAS.SystemFont	SystemFont	String	-adobe-helvetica-medium-r-normal–12–*–*–*–*–*–*–*
SAS.toolBoxAlwaysOnTop	ToolBoxAlwaysOnTop	Boolean	True
SAS.toolBoxTipDelay	ToolBoxTipDelay	int	750
SAS.uiconCount	UiconCount	int	0
SAS.uiconPath	UiconPath	String	NULL
SAS.useCommandToolBoxCombo	UseCommandToolBoxCombo	Boolean	True
SAS.useLargeToolBox	UseLargeToolBox	Boolean	False
SAS.usePmenuMnemonics	UsePmenuMnemonics	Boolean	True
SAS.useShowHideDecorations	UseShowHideDecorations	Boolean	False
SAS.useToolBoxTips	UseToolBoxTips	Boolean	True
SAS.webBrowser	WebBrowser	String	Netscape
SAS.windowHeight	WindowHeight	Dimension	50
SAS.windowUnitType	WindowUnitType	String	percentage
SAS.windowWidth	WindowWidth	Dimension	67
SAS.wsaveAllExit	wsaveAllExit	Boolean	False

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS® Companion for the OpenVMS Environment, Version 8,* Cary, NC: SAS Institute Inc., 1999. 518 pp.

SAS® Companion for the OpenVMS Environment, Version 8

1–58025–526–4

1st printing, October 1999

The Institute is a private company devoted to the support and further development of its software and related services.

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