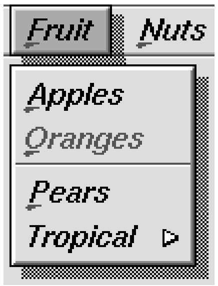
NOTE: This document excerpts Silicon Graphics' User Interface Style Guidelines, in order to provide an overview of a concerted set of style enhancements to baseline OSF/Motif Style, as implemented in release 1.1 from the Open Software Foundation.

Component Specifications

A standard set of user interface components are specified, in order to promote a familiar set of interaction techniques across all applications within the SGI User Environment. While many developers will use available toolkits which provide these components readymade, off the shelf, the following specification provides sufficient guidance for programmers who wish to implement their own style–conforming components.

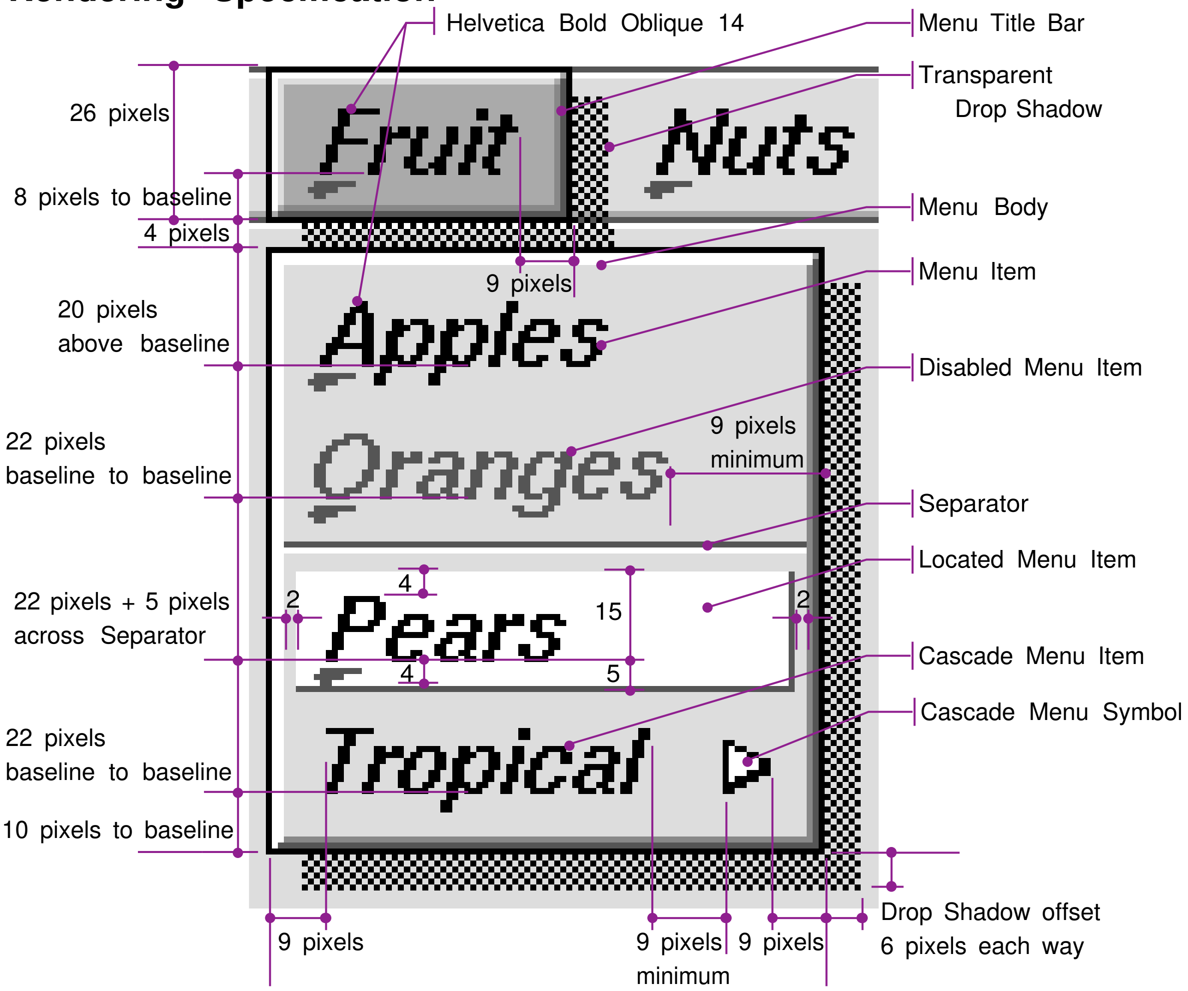
Table of Components [excerpted]

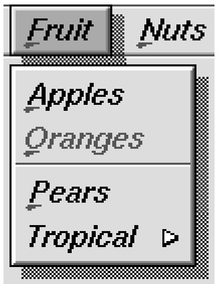


Pull Down Menu

Use Pull Down Menus with Pull Down Menu titles in the Menu Bar, to provide quick access to application commands. See "Menu Bar" for a fuller description.

Rendering Specification



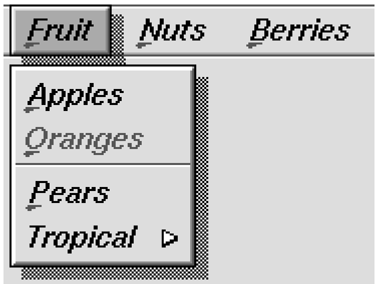


Pull Down Menu

Presentation States

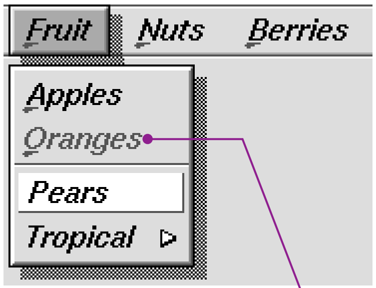
Top–level Pull Down Menu

Neutral



Located Not Applicable

Loc+Sel

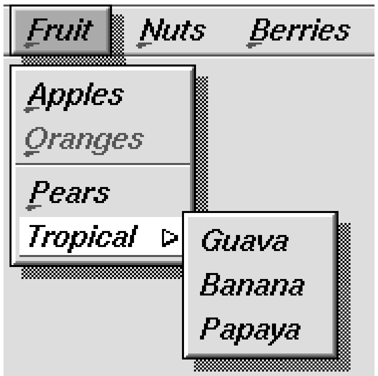


Selected Not Applicable

Disabled Grayed out per Menu Item

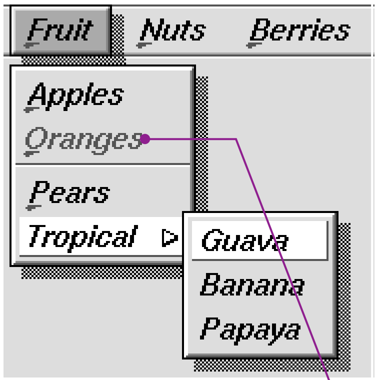
Cascaded Submenu

Neutral



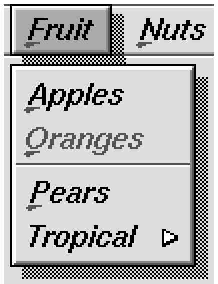
Located Not Applicable

Loc+Sel



Selected Not Applicable

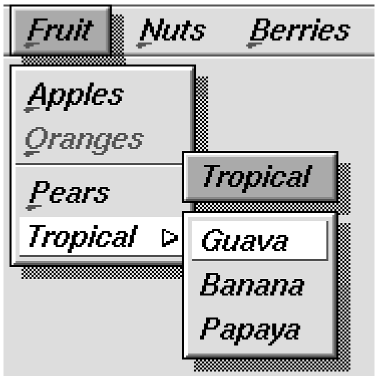
Disabled Grayed out per Menu Item



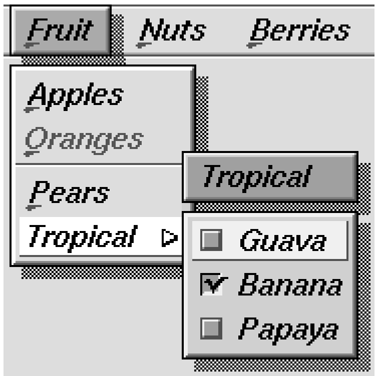
Pull Down Menu

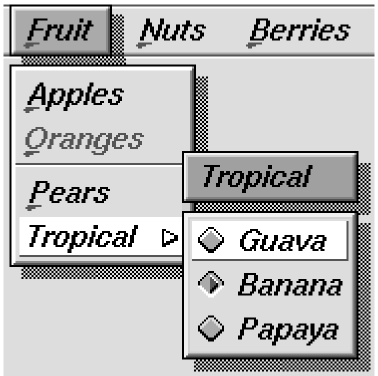
Variations

A Menu Title Bar may be added to the Cascading Submenu for further emphasis.



Check Button Items and Radio Button Items may be used within Pull Down Menus.

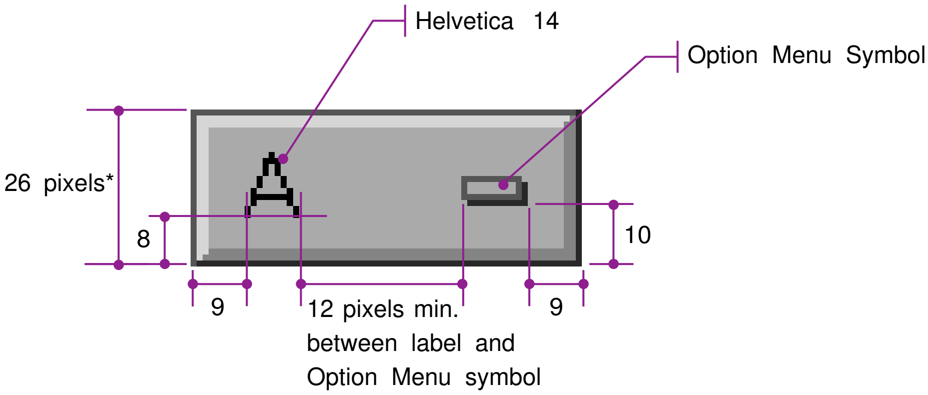




Option Menu

Use an Option Menu to establish a field where users can choose one setting from a limited list of options. The field is represented by a fixed Option Menu Button, with the text label presenting the value of its current effective setting. When the user presses down either the Select Mouse Button (LeftMouse) or the Popup Mouse Button (RightMouse) over the Option Menu Button, a Popup Option Menu appears. The user can make a new selection in the same way as with a standard Popup Menu. When the selection is completed, the Popup Option Menu disappears, and the selected menu item appears on the face of the Option Menu Button.

Rendering Specification



* Height to match Menu Bar

Option Menu

Presentation States

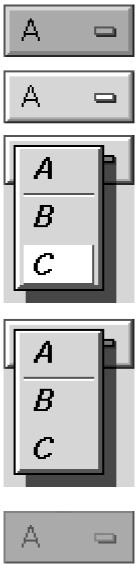
Neutral

Located

Loc+Sel

Selected

Disabled

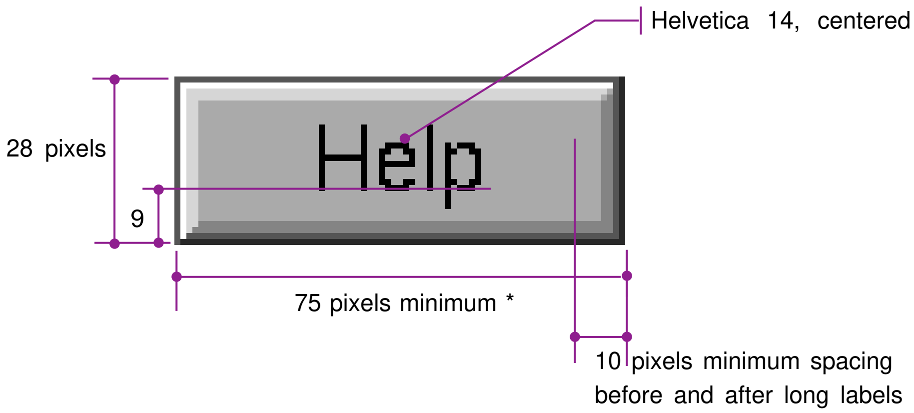


After a new item has been selected from the popup Option Menu, its label is displayed on the Option Menu Button.

Push Button

The Push Button consists of a button and a text label. Fixed Push Buttons are preferable to transient menus when the user needs to make several selections at once or is working primarily in a control panel area.

Rendering Specification

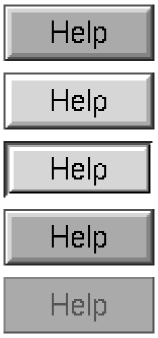


*Button width is the greater of 75 pixels or the actual text label width + 2 * 10 pixel spacing.

Push Button

Presentation States

Neutral
Located
Loc+Sel
Selected
Disabled



State Transitions

[Transitions between the basic presentation states, and any other intermediate states, are outlined against user input, to help describe the behavior of the component.]

Rendering Details

Neutral

Located

Loc + Sel

Text is always Black unless disabled. In disabled state, it is Dark Gray.

Push Button

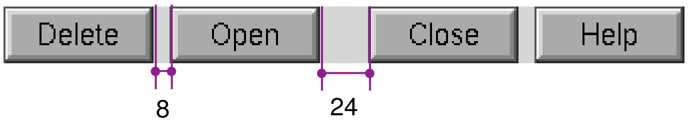
Rendering Details

Selected

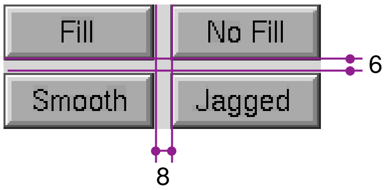
Disabled

Layout Guidelines

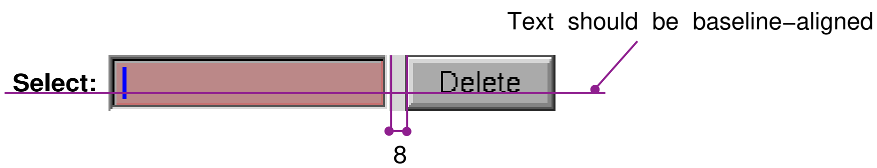
Fixed menus of Push Buttons are suitable for presenting high–visibility commands within a dialog or option box. Related Push Buttons should be grouped into distinct sets, and positioned within the window in a manner which reflects the mainstream workflow. Master commands which affect the entire window context should appear across the very bottom of the window, aligned from right to left starting in the lower right hand corner of the window.



8 pixels horizontally between push buttons,
24 pixels between groups of push buttons.



6 pixels verticaly between groups
of push buttons

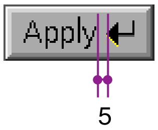


Push buttons and text entry fields are the same height, and can thus be aligned.

Push Button

Variations

One Push Button within a given window can be designated as the "default" Push Button for that window. When the <Return> key is pressed, the default Push Button is executed, providing a speedy shortcut alternative to clicking that Push Button with the Select Mouse Button (LeftMouse). The default Push Button should be clearly labelled with the Return Arrow symbol, to inform the user of the available shortcut.



The button designated as the "default" case shows a Return Arrow symbol to the right of the text label. Leave 5 pixels of space between the text label and the arrow symbol, and center the resultant label–symbol pair on the face of the button.

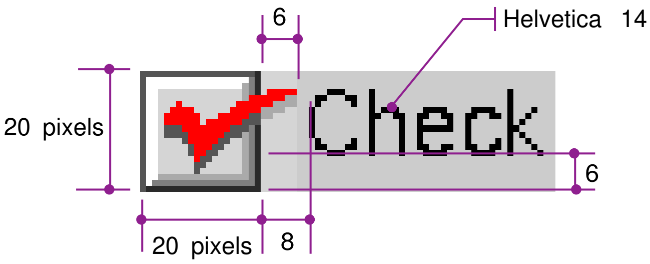
Check Button

Use a Check Button to set or unset the state of a given parameter or option. Each individual Check Button serves as its own field, controlling one independent variable within the application. You may use clusters of Check Buttons to manage a set of related parameters, such as bold, italic, and underline modifiers to a font style, and you may provide any number of these fields within the application.

The Check Button should always be presented with an accurate, up–to–date status (On or Off), whether the related variable has been changed by the user, via the Check Button interface itself, or as a side effect of the underlying application.

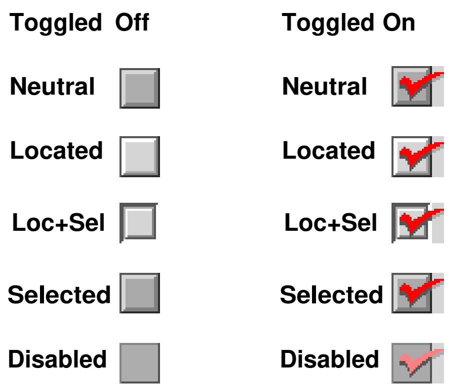
The Check Button is composed of a text or graphic label, preceded by the square check–box device itself.

Rendering Specification



Check Button

Presentation States



State Transitions

[Transitions between the basic presentation states, and any other intermediate states, are outlined against user input, to help describe the behavior of the component.]

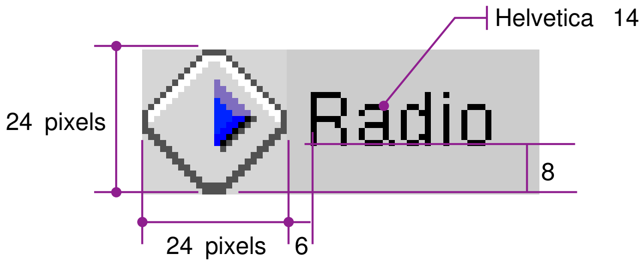
Radio Button

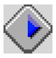
Use a Radio Button to select one option from a group of related, mutually exclusive options. You can provide any number of Radio Buttons within a single field (as a gang of related, alternative settings), and you may provide any number of these fields within the application.

Within a given field, a user can set only one Radio Button at a time to the On state. When users set one Radio Button, they automatically unset all the other Radio Buttons within that field.

The Radio Button is composed of a text or graphic label, preceded by the diamond–shaped button device itself.

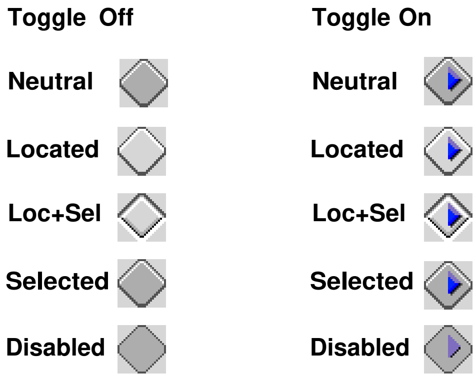
Rendering Specification





Radio Button

Presentation States



State Transitions

[Transitions between the basic presentation states, and any other intermediate states, are outlined against user input, to help describe the behavior of the component.]

Field Title and Label

You use a Field Title to provide primary identification of the role of a given control or working area. It is used to name fields and groups of buttons which act together as a single field. Field Titles are displayed in Helvetica Bold 14, and are always followed by a colon. Abbreviations should never be used in Field Titles.

Labels provide minor text identification, appearing on the faces of Push Buttons, Option Menu Buttons, and opposite individual Check and Radio Buttons. Labels are also used to present output–only values, captioning tic marks on Dials and Valuators, as well as filling in output–only values to the right a Field Title. Labels are displayed in Helvetica 14.

Rendering Specification

Helvetica Bold 14:
Helvetica 14.

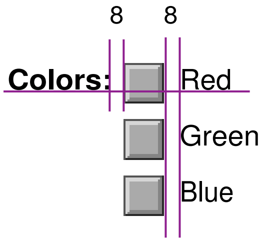
Field Title: Label

Field Title and Label

Layout Guidelines

Field titles are often found above the fixture to be identified. Leave 6 pixels vertical distance in this case. Note Left justification of field title and left edge of scrolling list.

They are also often seen to the left of the fixture.
A field title identifying a text entry field.



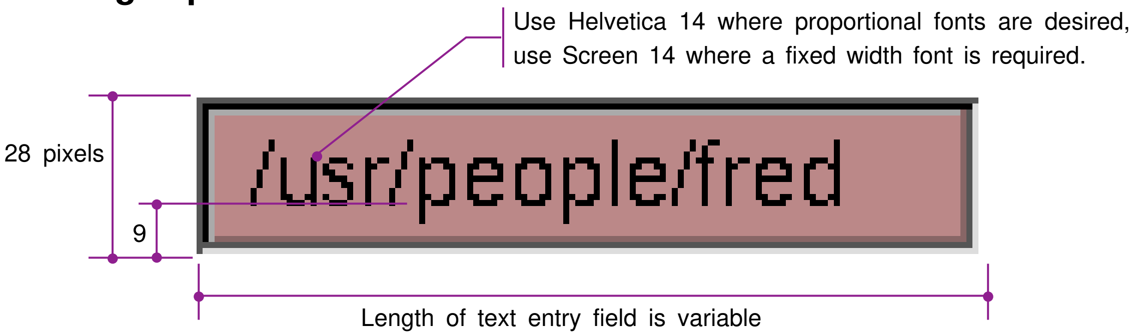
Field title identifies group of check buttons.
Labels identify each individual button.
Note position of labels to right of buttons.
This is true for radio buttons also.

Text Field

Use a Text Field to present text from the application to the user, and to receive text from the user into the application. You can use a Text Field for output only (read–only), or you can use one of the text editing configurations that support input. These include fixed–format text entry fields (for example, numeric–only or restricted character sets), a single–line, horizontally scrolling field suitable for entering filenames and other arbitrarily long "one liners", a compact scrolling field just two lines high, and a full–fledged multi–line text editor.

Use the distinctive "terra cotta" color for any field that supports text entry from the user. This color tells users that they can type input text into the field. Never use this reserved background color for scrolling lists or other output–only fields, as their contents may not be edited free–form directly from the keyboard.

Rendering Specification



Text Field

Presentation States

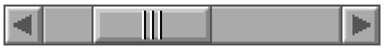
Text Entry Fields do not use locate or select highlighting in the same manner that most other components do. Instead, highlight the Text Field's internal text selection if and only if the field is selected as the one current keyboard input focus.

With Keyboard Focus: show text insertion cursor and sweep–selection

Without Keyboard Focus: hide text insertion cursor and sweep–selection

Do not display the text selection highlight when the Text Field loses input focus.

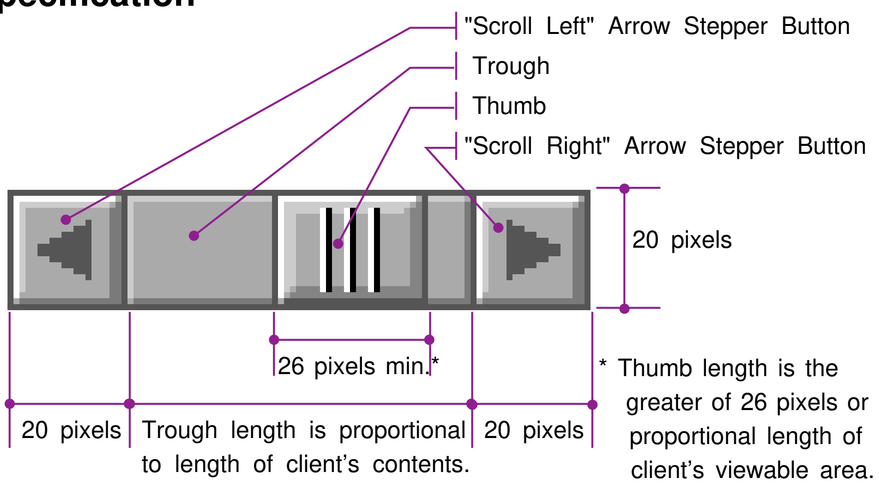
The SGI Style Guide recommends an explicit, "click–to–type" keyboard focus policy. Under this policy, each and every application window can have at most one candidate for the active keyboard input field among its various internal components. As the user makes a given application window the current input focus, that window's candidate for the active keyboard input field gains the keyboard focus. The application may want to assign a convenient text field as its default candidate for input; however, the user can easily assign the keyboard input focus to a new field within that window, by clicking the Select mouse Button (LeftMouse) over the desired field.

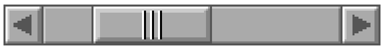


Scroll Bar

You use a Scroll Bar to manage the view of the contents of an associated client component. The standard Scroll Bar consists of a Thumb which slides back and forth within a Trough, accompanied by a pair of Arrow Stepper Buttons. The length of the Trough represents the entire length of the contents of the client, and the proportional position and length of the Thumb represents the subset of the contents currently visible within the client's view. As the Thumb is dragged within the Trough, the view of the client's contents is re–oriented continuously and simultaneously, allowing the user to browse the newly–revealed contents on the fly. The Scroll Bar may be used either horizontally or vertically, or in combination together to scroll the appropriate dimensions of the client's view.

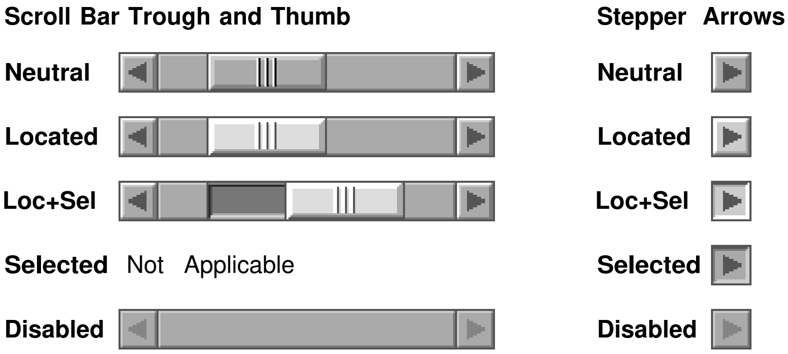
Rendering Specification





Scroll Bar

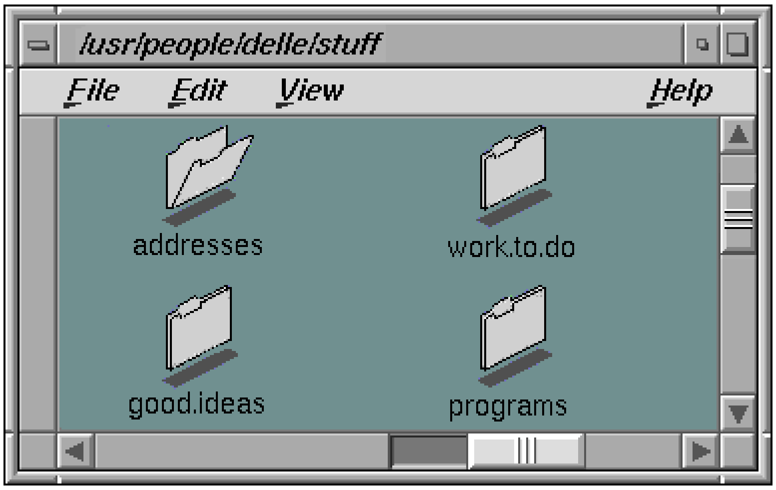
Presentation States



Layout Guidelines

When assembling the Scroll Bar Trough and its surrounding Arrow Stepper Buttons, be sure to overlap their outermost single–pixel outline. This should also be done when abutting the Scroll Bar with its adjoining client Pane, and when abutting the Scroll Bar with the Window Border Frame.

A horizontal Scroll Bar should be presented along the lower edge of its adjoining client Pane. A vertical Scroll Bar should be presented along the right edge of its adjoining client Pane.



Typical example of horizontal and vertical Scroll Bar usage.