Motif and CDE Style Guide Certification Checklist

Preface

- Audience
- Applicability
- Organization
- Related Documents
- Conventions
- Problem Reporting

Certification Checklist

- Action (Choice Type)
- Action Message
- Active Window
- Adjustment Techniques
- Application Design Principles (CDE)
- Area Adjust Click Technique
- Area Adjust Swipe Technique
- Area Click Technique
- Area Swipe Technique
- Browse Technique
- Cancel (Action and Choice)
- Cascading (Choice Type)
- Change View (Choice)
- Check Box (Control)
- Choice
- Clear (Action Choice)

Client Area

Clipboard

Close (Choice)

Combination Box (Control)

Combination Text–List Control (Control Type)

Command Area (Area)

Command Box (Control)

Component Activation (CDE)

Controls, Groups, and Models (CDE)

Container (Control)

Context–Sensitive Help

Control Navigation

Copy To (Dialog Choice)

Cursor

Cut, Copy, Paste (Action Choice)

Data Transfer

Default Action

Delete (Action Choice)

Dialog (Choice Type)

Dialog (Window)

Direct Editing

Direct Manipulation

Drag–and–Drop Transfer

Drag/Move (Action Choice)

Drop–Down Combination Box (Control)

Drop–Down List (Control)

Edit (Menu)

Emphasis (Cue)

Exit (Choice)

File Menu

File Selection (Dialog)

Find (Choice and Dialog)

First–Letter Cursor Navigation

Gauge (Control)

Group Box

Help (Menu/Action Choice)

Icon

In–Progress Message

Include (Choice and Dialog)

Information Message

Input Focus

Input Models (CDE)

Internal Navigation

Keyboard (Device)

Label

List Box (Control)

Margin Selection Techniques

Maximize (Choice)

Menu (Control)

Menu Bar (Menu Type)

Menu Cascade Button (Control)

Menu Guidelines

Message

Minimize (Choice)

Mnemonic

Mouse (Device)

Move To (Dialog Choice)

Multilevel Selection Technique

Multipage Control (Control Type)

Navigation (CDE)

New (Action Choice)

New Window (Choice)

Notebook (Control)

Object

Open (Choice)

Option Menu (Menu Type)

Options (Menu)

Palette Area (Area)

Paned Box (Control)

Persistent Cue

Point Technique

Pointer

Pointer (Predefined)

Pop–Up Menu (Menu Type)

Prefix Completion

Prefix Navigation

Primary Transfer

Primary Window

Properties (Choice)

Push Button (Control)

Push Button (Predefined)

Quick Transfer

Radio Button (Control)

Range Adjust Click Technique

Range Adjust Swipe Technique

Range Click Technique

Range Swipe Technique

Restore (Choice)

Sash (Control)

Save/Save As (Action Choice)

Scroll Bar (Control)

Scrolling

Secondary Window

Selected Menu

Selection

Selection (CDE)

Selection Box (Control)

Selection Dialog (Secondary Window)

Selection Models

Selection Modes

Selection Policies

Selection Techniques

Shortcut Key

Size (Choice)

Size Border (Control)

Slider (Control)

Sort (Choice and Dialog)

Spin Box (Control)

Spring–Loaded (Control Type)

Spring Sensitive (Mode)

Stacking Order

Static Text

Tab Group

Tear–Off Menu (Menu Type)

Text–Display Field (Control)

Text–Entry Field (Control)

Text Field (Abstract Control)

Tool (Value Choice Type)

Touch Adjust Click Technique

Touch Adjust Swipe Technique

Touch Swipe Technique

Undo, Redo, Repeat (Action Choices)

Value (Choice Type)

Value Set (Control)

View

View Menu

Viewing Area (Control)

Warning Signal

Window (Menu)

Window Frame

Window Icon

Window Icon Box

Window Management (CDE)

Window Menu

Window Navigation

Window Title

Workspace

Preface

The Style Guide Certification Checklist is a supplement to the Style Guide and Style Guide Reference. This checklist lists the required guidelines for the topics discussed in the Style Guide and Style Guide Reference.

For specific details on coding an application program, widget, or window manager, refer to Related Documents .

Audience

This document is written for five audiences:

Application designers

Any designer who uses Motif as a basis for building an application

Widget designers

Any designer who adds functionality to the existing Motif software

User interface system designers

Any designer who designs an interface for users with specific cultural and accessibility needs

Window manager designers

Any designer who uses Motif to govern the layout and controls that affect application windows

Common Desktop Environment application designers

Any designer who builds an application for the Common Desktop Environment

You should read through the entire Style Guide once to familiarize yourself with all user interface design concepts. If you are already familiar with graphical user interfaces and their design concepts but need specific guidelines, read the Style Guide Reference. When you are ready to build your application, use the Style Guide Certification Checklist to ensure your application complies with the style guidelines discussed in the Style Guide and Style Guide Reference

Applicability

This is revision 2.1 of this document. It applies to Version 2.1 of the Motif software system and Version 2.1 of the Common Desktop Environment system.

Organization

This document is organized alphabetically into topics that are covered in the Style Guide and Style Guide Reference.

Related Documents

For documentation on Motif and CDE style, see the following documents:

- Motif and Common Desktop Environment: Style Guide
- Motif and Common Desktop Environment: Style Guide Reference

For additional information about Motif, refer to the following documents:

- Motif User's Guide
- Motif Widget Writer's Guide
- Motif Programmer's Guide
- Motif Programmer's Reference
- Motif Release Notes

For additional information about the Common Desktop Environment, refer to the following documents:

- Common Desktop Environment: User's Guide
- Common Desktop Environment: Advanced User's Guide
- Common Desktop Environment: Information Manager Users Guide
- Common Desktop Environment: Application Builder User's Guide
- Common Desktop Environment: ToolTalk Messaging Overview
- Common Desktop Environment: Programmer's Guide
- Common Desktop Environment: Help System Author's and Programmer's Guide
- Common Desktop Environment: Internationalization Programmer's Guide
- Common Desktop Environment: Information System Author's and Programmer's Guide

Conventions

This document uses the following conventions:

Typeface or Symbol	Meaning	Example
`Enter`	A key on the keyboard	When the user presses `Delete`, delete the text.
SELECT, ADJUST, TRANSFER, MENU	A virtual mouse button	The SELECT button selects text.
MB1, MB2, MB3	A physical mouse button (where MB1 is the leftmost button)	Make the MB1 the SELECT button.
Push button	A new term	A *push button* is a control that displays a label or graphic that represents an action.
Control	A reference page	For more information, see the Control reference page.
ls	A command	Use **ls –a** to list all files.

Problem Reporting

If you have any problems with the software or documentation, contact your software vendor's customer service department.

Certification Checklist

The *Certification Checklist* provides a list of requirements for Motif and CDE application−level certification. For your application to be certified Motif or CDE compliant, it must behave according to these requirements. You certify your own application by comparing its behavior with that specified in the checklist.

Check each checklist item only if your application performs exactly as described for that item. If you have not implemented a specified type of behavior in any manner anywhere in your application, you may skip the items pertaining to that behavior.

The checklist uses a model keyboard mechanism to describe keys. For more information on the model keyboard and virtual mouse button mechanisms, see Appendix B and Appendix C.

By default, this checklist assumes that your application is being designed for a left−to−right language environment in an English−language locale. Some sections of the checklist may require appropriate changes for other locales. See Chapter 11 for more information.

As you compare the behavior of your application to the requirements in the checklist, refer also to the corresponding reference page in the Style Guide Reference for recommended and optional guidelines. If you do not understand a particular item, refer to the appropriate section in the Style Guide and check the glossary in the Style Guide for any terms that are unclear.

Action (Choice Type)

[ ]

If an operation or action is to happen immediately after the activation of a choice, provide an action choice.

[ ]

When a user activates an action choice, immediately begin to perform the action.

[ ]

Provide access to all action choices through menus, push buttons, or both.

[ ]

Do not use an action choice to request further parameters in a secondary window; use a dialog choice instead.

[ ]

Do not use an action choice to temporarily display a list or menu; use a cascading choice instead.

[ ]

If an action choice can possibly refer to more than one control in a window, use the following rules to determine which control is affected:

- If the action can be applied to the control that has focus emphasis, apply it to that control.
  For example, if the focus is in a text−entry field and the user uses a shortcut key to invoke the Paste choice, paste to the text−entry field that has focus.
- If the action operates on a selection, apply the action to the primary selection when the following is true:
  - The primary selection is in the window
  - No editing operation has been done in any other control in the window since it was selected
- If the control with interacted emphasis is in the window and the action can be applied to it, apply it.
  The control with interacted emphasis is the last editable control the user interacted with, so, for example, if the user last interacted with a text−entry field, paste to that field.
- If the window has a main control and the action can be applied to it, apply it.
  For example, a drawing program might have a large drawing area and a small text−entry field for entering identification numbers. The drawing area is the main control; if the previous cases do not apply, paste to it.
- If none of the previous cases is true, then invoking the action has no effect except for invoking a warning message.

Action Message

[ ]

Display a question message when the user must respond to a question before continuing the task, but only when the task can be suspended indefinitely without harm and other user activity is unlikely to affect the continuation of the task at a later time.

[ ]

Display a warning message to alert the user to a possible danger and to allow the user to take some alternative action or to withdraw the task request.

[ ]

Display an error message when a response is needed before the task can be continued (or if it is likely that other user actions might make the task impossible to continue) and if it is essential that either the situation be corrected or the task canceled.

[ ]

Display the appropriate symbol to the left of each action message as follows (except in cases of bidirectional language support; see Chapter 11):

Question message

?

Warning message

!

Error message

0

Behavior

[ ]

Display an action message in a secondary window.

Active Window

[ ]

Make the window that can currently receive keyboard input the active window.

[ ]

If an explicit focus policy is in use, and an active secondary window is closed, move focus to the window that the active window is dependent on.

[ ]

When a window receives input focus, provide a visible difference in the window's title bar. For example:

Change the color of the active window's title bar and border to the color the user has specified for the operating environment.

Increase the saturation of the title bar to signal the change for monochrome systems and in situations where the user might have difficulty distinguishing colors.

Adjustment Techniques

[ ]

If you support a group technique, support its corresponding adjustment technique variants.

[ ]

Allow an adjustment technique to be used if and only if the previous selection technique used in the scope was:

- A point technique
- An associated group technique
- Another adjustment technique associated with the same group technique
- A margin or multilevel technique associated with the same group technique (for example, allow only the range adjust click technique to be used after a point, range [click, swipe, click adjust, or click swipe], or a range margin/multilevel technique)

[ ]

When using an adjustment technique, if the current selection region is selected instead of toggled, the following should occur:

- Select all the elements in the new selection region.
- Deselect all other elements in the scope.

[ ]

When using an adjustment technique, if the current selection region is toggled instead of selected, the following should occur:

- Elements added to the selection region become toggled, based on the toggling policy.
- Elements removed from the selection region have their state determined by the toggling removal policy.
- The state of all other elements in the scope are unaffected.

Application Design Principles (CDE)

Main Window Layout

[ ]

Compose your application with at least one main window.

A main window contains a client area and, optionally, a menu bar, a command area, a message area, and scroll bars. The client area contains the framework of the application.

[ ]

The default size of the application's main window must be large enough to accommodate a typical amount of data, but should not fill the entire physical display size to minimize visual conflicts with other applications.

[ ]

Include resize corners in any main window that incorporates a scrolling data pane or list.

[ ]

If your application has multiple main windows that serve the same primary function, close and iconify each window separately.

[ ]

If your application has multiple main windows that serve different primary functions, each window should be able to iconify independently of the other windows.

Menu Bar Layout

[ ]

If your application has a menu bar, use a horizontal bar at the top edge of the application, just below the title area of the window frame. A menu bar organizes the most common features of an application. It contains a list of menu topics in cascading buttons; each button is associated with a distinct pull–down menu that contains commands that are grouped by common functionality.

[ ]

The menu bar should contain only cascading buttons.

File Menu Contents

[ ]

If the user chooses Exit, or in any other manner indicates that the application should be terminated, but if there are changes to the current file that have not been saved, display a dialog box that asks whether the changes should be saved before exiting.

[ ]

If your application uses a File menu, it must include the choices in **File Menu Choices** , with the specified functionality.

Table 1 File Menu Choices

Mnemonic	Menu Choice	Function
N	New...	Creates a new file. If the current client area will be used to display the new file, clear the existing data from the client area. If changes made to the current file will be lost, display a dialog box, asking the user about saving changes.
O	Open...	Opens an existing file by prompting the user for a file name with a dialog box. If changes

		made to the current file will be lost, display a dialog box asking the user about saving changes.
S	Save	Saves the currently opened file without removing the existing contents of the client area. If the file has no name, display a dialog box, prompting the user to enter a file name.
A	Save As...	Saves the currently opened file under a new name by prompting the user for a file name with a dialog box. If the user tries to save the file with an existing name, display a dialog box that warns the user about a possible loss of data. Do not remove the existing contents of the client area.
P	Print (recommended)	Schedules a file for printing. If your application needs specific information to print, it displays a dialog box, requesting the information from the user. In this case, the menu entry is followed by an ellipsis (Print...).
C	Close (recommended)	Closes the current primary window and its associated secondary windows. If your application uses only a single primary window or multiple dependent primary windows, this action is not supplied
X	Exit	Ends the current application and all windows associated with it. If changes made to the current file will be lost, displays a dialog box asking the user about saving changes.

Edit Menu Contents

Recommended

If your application does not provide an <object–type> or Selected menu, but allows the user to select data within the window and manage settings for the selected data, then provide a Properties... choice as the last item in the Edit menu.

Help Menu Contents

[ ]

If your application includes a Help menu, include the set of choices in Help Menu Choices in CDE , with the specified functionality, when the actions are actually supported by your application.

Table 2 Help Menu Choices in CDE

Mnemonic	Menu Choice	Function
V	Overview	Provides general information about the window from which help was accessed or about the application overall. Place a separator after this choice.
I	Index (optional)	Provides an index that lists the topics for all help information available for your application.
C	Table of Contents (recommended)	Provides a table of contents that lists the topics for all help information available for your application.
T	Tasks (recommended)	Provides access to help information that indicates how to perform different tasks within your application.
R	Reference (recommended)	Provides access to reference information.
L	Tutorial (optional)	Provides access to your application's tutorial.
K	Keyboard (optional)	Provides information about your application's use of function keys, mnemonics, and keyboard accelerators. Also provides information on general CDE use of such keys.
M	Mouse (optional)	Provides information about using a mouse with your application.

M	Mouse and Keyboard (optional)	Provides information about your application's use of function keys, mnemonics, keyboard accelerators, and using a mouse with your application. Also provides information on general CDE use of such keys. Use rather than separate mouse and keyboard choices if this information is best presented together.
O	On Item (recommended)	Initiates context−sensitive help by changing the shape of the pointer to the question mark pointer. When the user moves the pointer to a component and presses the SELECT button, any available context−sensitive help for the component is presented. Set off with separators on both sides.
U	Using Help	Provides information on how to use the CDE Help Viewer. Set off with separators on both sides.
A	About *application name*	Displays a dialog box that indicates, minimally, the name and version of your application, and displays its icon or some other signature graphic for your application.

Pop−Up Menus

[ ]

When a pop−up menu is popped up in the context of a selection, any action that acts on elements should also act on the entire selection.

Dialog Boxes

[ ]

Display an information dialog box such that it does not interrupt the user's interaction with your application.

An information dialog box conveys information to the user that does not require immediate attention so it does not need to be modal.

Menu Design

[ ]

Dim (make insensitive) any menu choice that is not currently an appropriate selection.

Dimmed controls cannot be activated by the user and should appear only when the inactive state is short−term (that is, there is something the user can do within the application or the desktop environment to make the control become active). When the control is persistently inactive (because of the current configuration of the application or system, or a particular set of companion software is not currently installed), remove the control rather than dim it.

[ ]

If radio buttons are used in a menu, use separators between each set of radio buttons and other menu items.

[ ]

If your application uses a tear−off choice in a menu, make the tear−off choice the first element in the menu.

[ ]

Make all menus wide enough to accommodate their widest elements.

Dialog Box Design

[ ]

Every dialog box should have at least one button that either performs the dialog box action and dismisses it or dismisses the dialog box without taking any action.

[ ]

If your application provides settings that control the behavior of the application, display these settings in an application properties window that is accessible from an Options menu.

Controls that relate to advanced features should not be displayed with the set of options initially displayed to the user. The typical user should be presented with only those options that are necessary to use the basic functionality of the application. Users looking to access advanced functionality within the dialog box may use the <Category> option button (see Figure 7−1). If the number of advanced controls is few, or the settings for these controls are highly related to the settings of basic controls displayed in the dialog box (that is, the settings of the advanced controls change when the user changes settings for basic controls), you might choose to provide an expandable dialog box.

Property Windows

[ ]

If your application provides settings that control the behavior of the application, display these settings in an application properties window that is accessible from an Options menu.

File Selection Dialog Box

[ ]

The file selection dialog box should recall the directory location that was previously set by the user.

For example, if the user brings up Save As and navigates to /users/jay/letters to save the file, the next time the user brings up Save As, the file selection box should be in the directory /users/jay/letters. The directory, however, should not be retained once the user has closed the primary window. When the user brings up Save As for the first time, it should resort to the default directory.

About Dialog Box

[ ]

Include a Close button in the About dialog box. Other buttons are optional, such as Help and More.

Dialog Box Layout

[ ]

Push buttons that affect the dialog box as a whole, either by modifying its contents or layout, invoking the action of the dialog box, or dismissing the dialog box, should be located at the bottom of the dialog box.

There should be only one row of buttons at the bottom of a dialog box. If your application has dialog boxes that contain several global buttons, you may need to create two or more rows of buttons at the bottom of the dialog box. The last row should contain the standard dialog box buttons (OK, Reset, Cancel, and Help). If a dialog box contains buttons that are not related to the dialog box as a whole, but relate to a specific control within the dialog box, the buttons should be located with the control to which they relate.

[ ]

If your application provides an Apply button within a dialog box, also provide an OK button or <command> button that performs the dialog box action and then dismisses it.

Designing Drag and Drop

[ ]

During a drag operation, change the current pointer to a drag icon.

[ ]

When the user presses `Cancel`, end a drag−and−drop operation by canceling the drag in progress.

[ ]

When the user releases the TRANSFER button (or the SELECT button) when not over a drop target, end a drag−and−drop operation.

[ ]

After a successful transfer, place the data in the drop zone and remove any transfer icon.

[ ]

If your application removes data upon the completion of a drag−and−drop operation, do so only if the drag−and−drop transfer has completed successfully.

If a drag−and−drop operation has been canceled or failed, do not remove the data or object that was the source of the drag.

[ ]

After a failed transfer, keep the data at the drag source and do not place it in the drop zone. Remove any transfer icon.

[ ]

If your application provides any drag−and−drop help dialog boxes, include a `Cancel` button for canceling the drag−and−drop operation in progress.

Installation

[ ]

Install applications in folders in the Application Manager, not directly on the Front Panel or subpanels. For consistency, only CDE desktop components will install in the Front Panel or subpanels. Users may choose to rearrange their Front Panel, but applications should not do this without user consent.

Interaction

[ ]

Display a warning dialog box that allows the user to cancel the destructive action about which the dialog box is providing a warning. The user needs to have a way to cancel an operation that can cause destructive results.

[ ]

When your application displays a dialog box, place the input focus at the first text field into which the user is allowed to type an entry, or at the first control within the dialog box with which the user should interact.

Input focus should always be placed at a predictable and intuitive location. Do not force the user to set focus at the control most likely to be used when the window is displayed.

[ ]

There should always be exactly one control within any window of your application that has the input focus if the window in which it resides has the input focus.

[ ]

Dialog boxes should never block input to other applications within the desktop (that is, they are not system modal) unless it is essential that the user perform no other action on the desktop until the user responds to the dialog box.

[ ]

Dialog boxes should never block access to other functionality within the application (application modal) unless it is essential that the state of the application remains unchanged until the user responds to the dialog box.

[ ]

If your application does not use the values of global environment settings, such as multiclick timeout intervals, drag thresholds, window color settings, mouse left− or right−handedness, and so on, but instead uses its own values for these settings, then provide one or more

Options dialog boxes that allow the user to change the values for these settings.

In general, you should not override the value of settings treated as global environment settings. The user controls these settings through the CDE Style Manager. If you choose to ignore these settings and specify your own settings, then your application will behave inconsistently with other applications in CDE. If you nevertheless choose to provide your own values, then you must provide the user with a way to make your settings consistent with the rest of the desktop.

Toolbars

[ ]

If you use a tool bar, it should appear only in windows with a menu bar.

[ ]

Tool bars should contain only those operations that are already available to the user in your application's menus. All items in a tool bar should be redundant.

[ ]

When an action represented by a tool bar icon is unavailable to the user, make that icon insensitive, with the associated stippled appearance. If a menu item is made insensitive, make the corresponding tool bar item insensitive as well.

[ ]

Place the tool bar container directly under the menu bar. It has the same width as the window, as well as similar height to the menu bar.

Expandable Windows

[ ]

Align command buttons along the bottom of the dialog box. When the window is expanded to show a secondary pane, then move the buttons to the bottom of the secondary pane.

[ ]

Separate the primary pane from the secondary pane with a separator.

[ ]

If a window is resizable, allocate any sizing changes to the pane that contains scrolling lists or text fields whose displayed length is less than their stored length. If both panes contain scrollable controls, distribute size changes evenly between the two panes. If neither pane contains scrollable controls, the window should not be resizable.

[ ]

The expandable window should have one button that changes its label based on the state of the window.

[ ]

The expand button should have two labels that reflect the two states of the expandable window accurately. The current label indicates to the user what will happen if the user clicks the button.

Examples of possible labels are Basic and Options, Expand and Contract, and More and Less.

[ ]

If the window or dialog box contains a scrolling list positioned to the far right side of the pane, do not align the drawn button with the scroll bar. For example, the button should be aligned with the list, not the scroll bar.

[ ]

Applications must remember the state of each window or dialog box (expanded or not expanded) independently (not collectively). The state should be changed only by the user and should always be preserved until explicitly altered by the user.

Area Adjust Click Technique

[ ]

When using the area adjust click technique, and the adjustment policy is to reselect:
- Determine the new selection region to be the area determined by the anchor point and the adjustment point.
- Enlarge the new selection region to include the anchor element or anchor region, if indicated by the anchor inclusion policy.
- Enlarge the new selection region, if indicated by the end−point inclusion policy, to include the element, if any, at the adjustment point.

[ ]

When using the area adjust click technique, and the adjustment policy is to enlarge, the new selection region is larger than the following:
- The current selection region
- The region determined using the reselect adjustment policy

[ ]

When using the area adjust click technique, and the adjustment policy is to balance:
- Move (if necessary) the anchor point so that it is at the corner of the current selection region farthest from the adjustment point.
- Determine the new selection region to be the area determined by the new anchor point and the adjustment point.
- Enlarge the new selection region to include the anchor element or anchor region, if the anchor point is unchanged and if indicated by the anchor inclusion policy.
- Enlarge the new selection region, if indicated by the end−point inclusion policy, to include the element, if any, at the adjustment point.

[ ]

If the current selection region does not include an anchor element, or if the anchor element is no longer in the selection region, identify the anchor element to be the element, if any, in the region nearest the anchor point.

[ ]

After a mouse−based area adjust click technique is used, place the active cursor as follows:
- If it is a text or graphics cursor, at the point at which the ADJUST button was clicked
- If the selection region contains one or more elements, on some element within the selection region
- If neither of the above, on the currently cursored element in the selection scope

Area Adjust Swipe Technique

[ ]

Support the keyboard−based area adjust swipe technique when the last operation within the scope used an area−based selection technique.

[ ]

When using the area adjust swipe technique, and the adjustment policy is to reselect:
Determine the new selection region to be the area determined by the anchor point and the final adjustment point.
Enlarge the new selection region to include the anchor element or anchor region, if indicated by the anchor inclusion policy.
Enlarge the new selection region, if indicated by the end−point inclusion policy, to include the element, if any, at the final adjustment point.

[ ]

When using the area adjust swipe technique, and the adjustment policy is to enlarge, the new selection region is larger than the following:
The current selection region
The region determined using the reselect adjustment policy

[ ]

When using the area adjust swipe technique, and the adjustment policy is to balance:
When using the keyboard−based technique, and the final adjustment point is not within the current selection region, move (if necessary) the anchor point so that it is at the end of the current selection region farthest from either the initial or the final adjustment point. (The use of the final adjustment point is recommended.)
When using the keyboard−based technique, and the final adjustment point is within the current selection region, or when using the mouse−based technique, move (if necessary) the anchor point so that it is at the end of the current selection region farthest from the initial adjustment point.
Determine the new selection region to be the area determined by the new anchor point and the final adjustment point.
Enlarge the new selection region to include the anchor element or anchor region, if the anchor point is unchanged and if indicated by the anchor inclusion policy.
Enlarge the new selection region, if indicated by the end−point inclusion policy, to include the element, if any, at the final adjustment point.

[ ]

If the current selection region does not include an anchor element, or if the anchor element is no longer in the selection region, identify the anchor element to be the element, if any, in the region nearest the anchor point.

[ ]

After a mouse−based area adjust swipe technique is used, place the active cursor as follows:
If it is a text or graphics cursor, at the point at which the ADJUST button was released
If the selection region contains one or more elements, on some element within the selection region
If neither of the above, on the currently cursored element in the selection scope

Area Click Technique

[ ]

Support the area click technique only when a common selection in the scope consists of a set of spatially adjacent elements.

[ ]

Support the area click technique only when discontiguous selections are allowed.

[ ]

Support the keyboard−based area click technique in normal mode only when a text cursor is used for all navigation between the two corners of the area. The result of attempted use in other cases is undefined.

[ ]

Define the current selection region to consist of the specified area as follows:

Enlarged to include the element, if any, on which the technique was initiated

Enlarged, if indicated by the end−point inclusion policy, to include the element, if any, on which the technique finished

Mouse−Based Techniques

[ ]

After a mouse−based area click technique is used:

Identify the anchor point to be the point at which the SELECT button was clicked.

Identify the anchor element to be the element, if any, within the selection region nearest to the anchor point.

[ ]

After a mouse−based area click technique, place the active cursor as follows:

At the point at which the ADJUST button was clicked, if it can be placed there

If using an element cursor, on some element within the region

If there are no elements in the region, where the cursor previously was in the scope

[ ]

When using the area click technique in select mode:

Select all the elements in the selection region.

Deselect all other elements in the scope.

[ ]

If currently in select mode, clicking `Ctrl` SELECT istead of SELECT causes the area click technique to act as if toggle mode were in use.

[ ]

When using the area click technique in toggle mode:

Toggle all elements in the selection region, based on the toggling policy.

Do not change the selection state of the other elements in the scope.

Keyboard−Based Techniques

[ ]

After a keyboard−based area click technique is used:

Identify the anchor point to be the point at which `Select`, `Space` (unless in text), or `Ctrl Space` was pressed.

Identify the anchor element to be the element, if any, within the selection region nearest to the anchor point.

[ ]

When using the keyboard−based area click technique in normal mode:

Select all the elements in the selection region.

Deselect all other elements in the scope.

[ ]

When using the keyboard−based area click technique in add mode:

Toggle all elements in the selection region, based on the toggling policy.

Do not change the selection state of the other elements in the scope.

Area Swipe Technique

[ ]

Support the area swipe technique only when a common selection in the scope consists of a set of spatially adjacent elements.

[ ]

Support the area swipe technique only when discontiguous selections are allowed.

[ ]

Define the current selection region to consist of the specified area as follows:
- Enlarged, if indicated by the end−point inclusion policy, to include the element, if any, at the initial corner of the area
- Enlarged, if indicated by the end−point inclusion policy, to include the element, if any, at the opposite corner of the area

Mouse−Based Techniques

[ ]

After a mouse−based area swipe technique is used:
- Identify the anchor point to be the point at which the SELECT button was pressed.
- Identify the anchor element to be the element, if any, within the selection region nearest to the anchor point.

[ ]

After a mouse−based area swipe technique, place the active cursor as follows:
- At the point at which the SELECT button was released, if it can be placed there
- If using an element cursor, on some element within the region
- If there are no elements in the region, where the cursor previously was in the scope

[ ]

When using the area swipe technique in select mode:
- Select all the elements in the selection region.
- Deselect all other elements in the scope.

[ ]

If currently in select mode, pressing `Ctrl` SELECT instead of SELECT causes the area swipe technique to act as if toggle mode were in use.

[ ]

When using the area swipe technique in toggle mode:
- Toggle all elements in the selection region, based on the toggling policy.
- Do not change the selection state of the other elements.

Keyboard−Based Techniques

[ ]

After a keyboard−based area swipe technique is used:
- Define the current selection region to consist of the identified range.
- Identify the anchor point to be the initial point at which the `Shift` navigation key was pressed.
- Identify the anchor element to be the element, if any, within the selection region nearest to the anchor point.

[ ]

When using the keyboard−based area swipe technique in normal mode:
- Select all the elements in the indicated area.

Deselect all other elements in the scope.

[ ]

When using the keyboard−based range swipe technique in add mode:

Toggle all elements in the indicated range, based on the toggling policy.
Do not change the selection state of the other elements in the scope.

Browse Technique

[ ]

Support the mouse−based browse technique only in select mode, and only when at most one element in a scope can be selected.

[ ]

Support the mouse−based browse technique only in normal mode, and only when an element cursor is being used.

Mouse−Based Browse Technique

[ ]

When using the browse technique in select mode, releasing the SELECT button on a selectable element does the following:

- Selects that element
- Deselects all other elements in the scope
- Places the active cursor on that element

Keyboard−Based Browse Technique

[ ]

Using a navigation key to move the active cursor within a selection scope in normal mode that results in an element cursor being placed on a selectable element does the following:

- Selects that element
- Deselects all other elements in the scope
- Identifies that element as an anchor element for later use in adjustment of the selection
- Defines the current selection region to consist of the area taken up by that element

[ ]

Using a navigation key to move the active cursor within a selection scope in normal mode that results in an element cursor being placed on a selectable element identifies a point as an anchor point for later use in adjustment of the selection. That point is one of the following:

- The pointer position (mouse−based technique)
- The cursor position (keyboard−based technique with a text or graphics cursor)
- The center of the element (keyboard−based technique with an element cursor)

Cancel (Action and Choice)

[ ]

Do not provide a Cancel choice within an in−progress message. Provide a Close or Stop choice instead.

[ ]

If the user invokes the cancel action while a requested task is in progress and is waiting for some input, stop the task and, if possible, undo or do not apply any changes.

[ ]

If the user invokes the cancel action while a requested task is in progress but is not waiting for some input, stop the task and confirm the cancellation with a warning message, unless the operating environment specifies that confirmation is not necessary.

[ ]

If the user invokes the cancel action while some multistep dialog is in progress, terminate the dialog, and do not continue the task that initiated the dialog. If possible, undo any changes to the application state that may have been made during the dialog.

[ ]

If the user invokes the cancel action while some modal interaction is in progress, stop the current interaction, exit from the mode, return to the normal mode of operation specified by the operating environment, and do not apply any changes.

Shortcut Keys

[ ]

Make `Cancel` the shortcut key for the Cancel choice.

Cascading (Choice Type)

[ ]

Use a cascading choice to display a cascaded control.

Activation and Unavailability

[ ]

When a user activates a cascading choice, display its associated cascaded control, even when all choices and controls in the cascaded control are unavailable.

[ ]

Do not display unavailable emphasis on cascading choices.

Pressing a Cascading Choice

[ ]

If a cascading choice is used to display a spring−loaded control, pressing (but not clicking) the SELECT button on the choice activates it and displays the control if it is not already displayed.

Releasing a Cascading Choice

[ ]

If the SELECT or MENU button is pressed, and the pointer is over a cascading choice whose associated cascaded control is not displayed (for example, because the mouse button was pressed elsewhere and moved to the cascading choice), then releasing the mouse button has no effect.

[ ]

Releasing the SELECT button over a cascading choice whose associated cascaded control is currently displayed leaves the cascaded control displayed if it was not displayed when the SELECT button was pressed.

[ ]

Releasing the MENU button over a cascading choice whose associated cascaded menu is currently displayed leaves the cascaded control displayed if it was not displayed when MENU was pressed.

Clicking to Display a Choice

[ ]

Clicking the SELECT button on a cascading choice activates it if its associated cascaded control is not currently displayed.

[ ]

If activating a cascading choice displays a menu, then clicking the MENU button on the choice activates it if the menu is not currently displayed.

Removing a Cascaded Control

[ ]

Clicking the SELECT button on a cascading choice removes the associated cascaded control if it is currently displayed.

[ ]

If a menu is associated with a cascading choice, then clicking the MENU button on the choice removes the associated cascaded menu if it

is currently displayed.

[ ]

Removing a control removes all of its cascaded descendants.

Change View (Choice)

[ ]

Provide a Change View choice in the View menu if your application allows a user to switch among three or more views.

[ ]

Make Change View a cascading choice; its cascaded menu should list the available views.

[ ]

Indicate the current view active in the window by displaying that view's choice in the cascaded menu with unavailable emphasis and, optionally, by including a check graphic next to it.

[ ]

Do not open a new window as a result of activating Change View; instead, display the new view of the element in the current window.

[ ]

Do not change the underlying data being viewed when the user changes the view.

Check Box (Control)

[ ]

Use a check box to display an individual value choice that can be set to on or off.

[ ]

Use a group of check boxes for value choices that are not mutually exclusive and can each be set to on or off.

[ ]

When a check box is initially displayed in a set or unset state, toggle only between those two values.

[ ]

When a check box is displayed in an indeterminate state, toggle from indeterminate to set to unset.

Selection Properties

[ ]

When a check box represents a property shared by more than one selected element:

Show it as set if all of the selected elements have that property turned on. For example, if all of the text in a text selection is bold, and a check box is used to indicate whether the selected text is bold, show the check box as set.

Show it as unset if none of the selected elements have that property turned on. For example, if none of the text in a text selection is bold, show the check box as unset.

Show it as indeterminate if some, but not all, of the selected elements have that property turned on. For example, if only some of the text in a text selection is bold, show the check box as indeterminate.

[ ]

When a check box represents a property shared by more than one selected element, toggling it to a particular state has the following effect:

Toggling it to set ensures that all of the selected elements have that property. For example, toggling the Bold check box to set makes all the selected text bold.

Toggling it to unset ensures that all of the selected elements do not have that property. For example, toggling the Bold check box to unset unbolds all the selected text.

Toggling it to indeterminate gives each selected element the value for the property it had when the check box was displayed as indeterminate.

Visuals

[ ]

A check box consists of a check graphic and a label. The check graphic is displayed to the left of the label. For information on bidirectional and vertical language support, see Chapter 11.

Choice

[ ]

Provide a choice for every action that the user can invoke or for every value that the user can set or unset.

[ ]

Use the predefined label for each predefined choice, except when Yes, No, Apply, and OK are replaced by more descriptive labels.

Behavior

[ ]

Pressing the SELECT button on a choice and then releasing the SELECT button on the same choice activates or toggles it.

[ ]

When focus is on an element that represents a choice, pressing `Select` or `Space` activates or toggles it.

Clear (Action Choice)

[ ]

Include the Clear choice in a pop−up menu on an unselected element only if it can be removed without compressing the visible state it occupied.

[ ]

Include the Clear choice in a pop−up menu on a selected element only if the selected elements can be removed without compressing the visible space it occupied.

[ ]

If only objects are selected, the Clear choice must be unavailable in the Edit menu; use Clear to Trash on the Selected menu instead. If the selection scope includes only objects, then Clear must not be in the Edit menu at all.

Client Area

[ ]

Include a client area when designing primary and secondary windows, dialog boxes, and pop−up menus.

[ ]

Place the client area directly beneath the window menu. For information on designing applications for vertical languages, see Chapter 11.

[ ]

Include scroll bars when the client area is too large to be completely displayed in the window.

Clipboard

[ ]

All selection scopes support transfer to the clipboard.

[ ]

Support clipboard transfer in any selection scope in which the current selection can be deleted or cut.

[ ]

If a control supports text entry or any form of data transfer into the control, it supports Paste.

[ ]

The source of a Copy, Copy Link, Copy Special, or Cut operation are the selected elements of the control for which the operation was invoked.

[ ]

The target of a Copy, Copy Link, Copy Special, or Cut operation is the clipboard.

[ ]

Cut is a move operation, Copy is a copy operation, Copy Link is a link operation. Copy Special results in either a copy or link operation, depending on the format chosen in the dialog.

[ ]

If Paste or Paste Special transfers a data format, it is a copy operation whose source is the clipboard.

[ ]

If Paste, Paste Special, or Paste Link transfers a link format, it is a link operation whose source is that of the operation that placed the link on the clipboard.

[ ]

The target of Paste, Paste Link, or Paste Special is the control for which the action was invoked.

Availability

[ ]

Do not make any Paste choices available when no data is in the clipboard or when the data cannot be pasted.

[ ]

If you provide Paste Special, do not provide Paste Link (the equivalent functionality is available via the Paste Special Dialog).

[ ]

If you provide Copy Special, do not provide Copy Link (the equivalent functionality is available via the Copy Special Dialog).

Close (Choice)

[ ]

Provide the Close choice in the window menu of every window.

Behavior

[ ]

When a user activates the Close choice for the last primary window displayed for an application, remove all dependent secondary windows and exit from the application.

[ ]

When a user activates the Close choice for a window, remove the window and any dependent secondary windows.

[ ]

Use a Close push button to close an in−progress message window without affecting the task in progress. Do not terminate the task when a user presses the Close push button in an in−progress message window.

Confirmation on Close

[ ]

If information in a primary window could be lost when the user activates the Close choice, display a warning message.

Combination Box (Control)

[ ]

Do not allow the list box in a combination box to take focus.

Combination Text−List Control (Control Type)

[ ]

Order the items in the list box in some application−defined way that is meaningful to the user, such as alphabetically, numerically, chronologically, or other such order.

Selection

[ ]

Use either a single or a browse selection model in the list box.

[ ]

Use the browse selection model in the list box if a text−entry field is used and the list box cannot take focus.

[ ]

Use the browse selection model in the list box if prefix completion is in use.

[ ]

Move input focus to the text field if the user presses the SELECT button on an item in the list box and the list box cannot take focus. The item indicated is selected as usual, however.

Navigation

[ ]

If the user presses `[darr ]` when focus is in the text field and the list box is not displayed, do one of the following:

Display the list.

Navigate down the list, select the cursored list item, and display it in the text field without displaying the list.

[ ]

If the user presses keys used for vertical navigation or paging when focus is in the text field and the list is not displayed, do one of the following:

If `[darr ]` would display the list, either display the list or do nothing.

Navigate in the list as indicated, select the cursored list item, and display it in the text field without displaying the list.

[ ]

If the user presses a key used for vertical navigation or paging when focus is in the text field and the list box is displayed, navigate in the list as indicated, select the cursored list item, and display it in the text field.

[ ]

A combination text−list control whose list box cannot take focus is a tab group.

[ ]

When focus is in a text−list control whose list box is temporarily displayed and keyboard navigation (for example, tab group or window navigation) is used to move the focus elsewhere, the list box is removed if currently displayed.

Character Entry

[ ]

When a text−entry field in a combination text−list control has focus, input of printing characters enters them into the text field, with or without prefix navigation.

Behavior

[ ]

When a list item is selected, other than by prefix navigation, place the text of the newly selected list item in the text field.

[ ]

When a newly selected list item is placed in the text field and is selected in the text field, place the cursor at the end of the text field.

Visuals

[ ]

Provide vertical scroll bars if some of the items in the list box are not displayed when the list box is displayed.

[ ]

Make the text field at least as wide as the list box would be if it were displayed.

Visuals When the List Box is Not Permanently Displayed

[ ]

If the list box is not always displayed, include a list cascade element from which the list box can be displayed.

Command Area (Area)

[ ]

If you include a command area, provide a text−entry field or a command box or a group of push buttons or both for common actions.

[ ]

Do not scroll the command area when the viewing areas of the window are scrolled.

Command Box (Control)

[ ]

Either display the list box permanently or do not display it at all.

Behavior

[ ]

When the user invokes the default action of the command box, issue the command in the text field of the command box.

[ ]

If the command is issued successfully, add the command to the end of the list box.

[ ]

If the last command in the list box is selected, and a successfully issued command adds the new command to the end of the list, select the new command.

[ ]

If the command failed, issue a warning signal but do not add the command to the end of the list box, and leave the command unchanged in the text field.

Component Activation (CDE)

Basic Activation

[ ]

Pressing the SELECT button activates a push button.

[ ]

When a push button has focus, pressing `Select` or `Spacebar` activates it.

[ ]

When an activatable menu entry has the focus, pressing `Select`, `Spacebar`, `Enter`, or `Return` activates the entry.

[ ]

When the SELECT button is pressed over a push button, change the appearance of the push button to indicate that releasing the SELECT button will activate the push button. If, while the SELECT button is pressed, the pointer is moved outside of the push button, restore the visual state is. If, while the SELECT button is still pressed, move the pointer back inside of the push button and change the visual state again to indicate the pending activation. If the SELECT button is pressed and released within a push button, activate the button, regardless of whether the pointer has moved out of the push button while it was pressed.

[ ]

If a selectable element of a collection is activatable, clicking the SELECT button, pressing `Select`, or pressing `Spacebar` (except in text) selects it. Double−clicking the SELECT button selects and activates it.

[ ]

The time allowed to detect a double−click *(**doubleClickTime: 500)* should be no less than 500 milliseconds.

Accelerators (Shortcut Keys)

[ ]

If your application uses accelerators (shortcut keys), the component with the accelerator should display the accelerator key or key combination following the label of the component.

[ ]

If a button with an accelerator is within a primary or secondary window, or within a pull−down menu from its menu bar, make it activatable whenever the input focus is in the window or the menu−bar system. If a button with an accelerator is within a pop−up menu, make it activatable whenever the focus is in the pop−up menu or the component with the pop−up menu.

An accelerator must be activatable from the window or component associated with the accelerator.

Mnemonics

[ ]

If your application uses mnemonics, the label for the component with the mnemonic should contain the character that is its mnemonic. If the label does not naturally contain the character, place the mnemonic in parentheses following the label.

[ ]

Mnemonic characters must be chosen for ease−of−location within the text of a label. Wherever possible, use the first character of the label. If that is not possible, use the last character of the label or, if there is more than one word, the first character of the second word. After that, go through the label from the second character on until a unique mnemonic is found.

[ ]

Make all mnemonics case insensitive for activation.

[ ]

When the location cursor is within a menu or a menu bar, pressing the mnemonic key of a component within that menu or menu bar moves the location cursor to the component and activates it. If a mnemonic is used for an option button or for a cascading button in a menu bar, pressing `Alt` and the mnemonic anywhere in the window or its menus moves the cursor to the component with that mnemonic and activates it.

Tear−off Activation

[ ]

When the user activates a tear−off choice, tear off the menu that contains the choice.

[ ]

When a menu with a tear−off choice is posted, pressing the TRANSFER button in the tear−off choice starts a tear−off action. As long as the TRANSFER button is held, a representation of the menu should follow the movement of the pointer. Releasing the TRANSFER button ends the tear−off action by unposting the menu system, creating a new window at the current pointer location that contains the contents of the menu, and giving focus to the new window in explicit pointer mode.

Help Activation

[ ]

When the user presses `Help` on a component. invoke any context−sensitive help for the component or its nearest ancestor with context −sensitive help available.

[ ]

Provide context−sensitive help at all locations.

Never use a "Help not available" message.

Default Activation

[ ]

If your application uses default push buttons in a window, highlight the current default push button. When the focus is on a push button, make its action the default action and highlight the push button. If the default action in a window varies, some push button must always have default highlighting, except when there is no current default action.

[ ]

When focus is in a window with a default action and an activatable menu does not have the focus, pressing `Enter` or `Ctrl Return` invokes the default action. If focus is in a component other than multiline text or an activated menu, `Return` also invokes the default action. These actions should have no other effect on the component with the focus, unless the default action has some effect on that component.

[ ]

Except in the middle of a button motion operation, make pressing `Cancel` anywhere in a dialog box equivalent to activating the Cancel push button in the dialog box.

Expert Activation

[ ]

If your application supports expert activation, expert actions should exist only as shortcuts to application features that are available through another mechanism.

Expert activation, using mouse double−clicking on buttons, provides a convenient way for experienced users to perform certain tasks

quickly. However, new users and keyboard−only users need to be able to perform the same tasks.

[ ]

When the focus is on a button used for expert activation, no default action should be available, unless the default and expert actions are the same.

[ ]

If a component with an expert action is selectable, activating the expert action first selects the component and then performs the expert action.

Previewing and Autorepeat

[ ]

If your application supports activation preview when the user presses the SELECT button, remove the previewing information when the user releases the SELECT button. Activation preview presents the user with additional information that describes the effect of activating a button. This information cannot interfere with the normal operation of the application.

Cancel Activation

[ ]

Pressing the `Cancel` key stops current interaction in the following contexts:

- During a mouse−based selection or drag operation, it cancels the operation.
- During a mouse−based scrolling operation, it cancels the scrolling action and returns the system to its state prior to the start of the scrolling operation.
- Anywhere in a dialog box that has a Cancel push button, it activates that push button, except during a mouse−based selection or drag operation.
- In a pull−down menu, it either dismisses the menu and moves the location cursor to the cascading button used to pull it down, or it unposts the entire menu system. In a pop−up menu, option menu, tear−off menu, or menu bar, it unposts the menu system.
- When the focus is in a torn off menu window, it closes the torn−off menu window.

Controls, Groups, and Models (CDE)

Check Button

[ ]

Check buttons should only select settings that are not mutually exclusive. A check button graphically indicates its state with the presence or absence of a check mark.

[ ]

When the user presses the SELECT button in a check button, display the check button with armed emphasis. If the check button was previously unset, show it in the set state. If the check button was previously set, show in the unset state.

Pressing the SELECT button arms a check button; releasing the SELECT button shows the result of activating it.

[ ]

When the user releases the SELECT button in the same check button in which the press occurred:
If the check button was previously unset, set it.
If the check button was previously set, unset it.

In all cases, disarm the check button and, if the check button is in a menu, unpost the menu. When the user releases the SELECT button, activate the check button.

[ ]

When the user presses `Enter` or `Return` in a check button, if the check button is in a window with a default action, activate the default action. If the check button is in a menu:
If the check button was previously unset, set it.
If the check button was previously set, unset it.
In both cases, disarm the check button and unpost the menu.

[ ]

When the user presses `Select` or `Spacebar` in a check button, if the check button was previously unset, set it. If the check button was previously set, unset it. In both cases, disarm the check button and, if the check button is in a menu, unpost the menu.

Combination Box

[ ]

In a list that can be scrolled, such as a scrollable list box, do not allow the cursor to wrap.

[ ]

Provide vertical scroll bars when some of the data is not visible in the combo box.

Command Box

[ ]

If your application uses a command box, include a text component with a command–line prompt for text input and a list component for a command history area. Use either the single selection or browse selection model.

[ ]

When an element of a command box list is selected, place its contents in the text area.

[ ]

Make the list navigation actions `[uarr ]`, `[darr ]`, `Ctrl Begin`, and `Ctrl End` available from the text component so the user

can move the cursored element within the list and thus change the contents of the text.

[ ]

The default action of the command box should pass the command in the text area to the application for execution and add the command to the end of the list.

File Selection Box

[ ]

If your application uses a file selection dialog box, it should contain the following components:

A directory text component that shows the current directory path. The user can edit the directory text component and press `Return` or `Enter` to change the current directory.

For applications that allow saving to different formats, an option button that allows users to specify the format when saving a file.

A file name text component for displaying and editing a file name. This component is optional when the file selection box is used to choose an existing file or directory.

A group of push buttons, including a command button, and Update, Cancel, and Help buttons. The command button is typically labeled Open or Save, but if there is another label that better describes the resulting action (such as Include), use that label.

Activating the command button carries out the corresponding action and dismisses the file selection box.

[ ]

If the Open button is activated while the appropriate file is selected in the contents list, the file should be utilized by the application and the file selection box dismissed.

[ ]

When the file selection dialog box is used to choose an existing directory, there must also be an additional button, labeled Update, that is enabled whenever a directory is selected in the contents list and that opens the directory. Make this Update button the default action.

[ ]

Make the normal text navigation and editing functions available in the text components for moving the cursor within each text component and changing the contents of the text. These actions provide a convenient way to choose a directory or file name from the corresponding list while focus remains in the text component.

[ ]

When the user double−clicks the SELECT button on an item in the contents list, select that item and activate the default action. In all cases, double−clicking the SELECT button on a directory in the contents list should open that directory and display its contents in the contents list (the default action is Open).

When the file selection box is used to choose an existing file and the user double−clicks the SELECT button on an appropriate file in the contents list, choose that file and dismiss the file selection box (the default action is Open).

When the file selection box is used to choose an existing directory or to specify a new directory or file, the files list should not appear.

[ ]

Make the normal text navigation and editing functions available in the text components for moving the cursor within each text component and for changing the contents of the text.

[ ]

Make use of a selection when one of the following occurs:

The user activates the command push button while an appropriate item is selected in the contents list

The user double−clicks the SELECT button on an appropriate file in the contents list

The user presses `Return` or `Enter` while the file name text component has keyboard focus and contains an appropriate item

[ ]

Make the file selection box display the contents of a directory in the contents list when the file selection box is initialized, when the user

presses Enter or Return in the directory text component, and when the user opens a directory in the contents list. Update the contents list each time the contents of the directory changes.

[ ]

If the user has opened the application with a file name argument, the Open dialog box should default to the directory in which that file resides.

List

[ ]

Within a list component, pressing [uarr] moves the location cursor to the previous item in the list, and pressing [darr] moves the location cursor to the next item in the list. In a scrollable list, pressing [larr] scrolls the list one character to the left, and pressing [rarr] scrolls the list one character to the right.

[ ]

Within a list component, pressing Ctrl Begin moves the location cursor to the first item in the list, and pressing Ctrl End moves the location cursor to the last item in the list. In a scrollable list, pressing Begin moves the horizontal scroll region so that the leftmost edge of the list is visible, and pressing End moves the horizontal scroll region so that the rightmost edge of the list is visible.

[ ]

Within a scrollable list, pressing Page Up moves the location cursor to the item one page up in the list, and pressing Page Down moves the location cursor to the item one page down in the list. In a scrollable list, pressing Page Left (or Ctrl Page Up) scrolls the list one page to the left, and pressing Page Right (or Ctrl Page Down) scrolls the list one page to the right.

[ ]

Within a list component, clicking the SELECT button selects the item that was double−clicked and then initiates any default action for the window.

Option Button

[ ]

If your application uses option buttons, make the label for the button the last selection made from the option button.

[ ]

When the user presses the SELECT button or the MENU button on an option button, post the associated option menu.

[ ]

When the user releases the SELECT button or the MENU button within the same option button that the press occurred in, post the associated option menu if it was not posted at the time of the press. When the user releases the SELECT button or the MENU button outside of the option button, unpost the associated option menu.

[ ]

When the user presses Select or Spacebar in an option button, post the associated option menu.

Paned Window

[ ]

If your application uses paned windows, they should be composed of any number of groups of components, called panes, each separated by a sash and a separator. Group the panes, sashes, and separators linearly, either horizontally or vertically. A sash is the handle on a separator between two panes that is used to adjust the position of the separator.

Panel

[ ]

The [uarr], [darr], [rarr], and [larr] directional keys navigate among components in a panel.

A panel group organizes a collection of basic controls in a horizontal, vertical, or two−dimensional layout. The directional keys are used to navigate among the controls.

Push Button

[ ]

When the user presses the SELECT button in a push button, arm the push button. When the user releases the SELECT button in the same push button where the press occurred, disarm and activate the push button. When the user releases the SELECT button outside the push button, disarm but do not deactivate the push button.

[ ]

When the user presses Enter or Return in a push button that is in a window with a default action, activate the push button. When the user presses Enter or Return in a push button in a menu, activate the push button and unpost the menu.

[ ]

When the user presses Select or Spacebar in a push button, activate the push button is activated. If the push button is in a menu, unpost the menu.

Radio Button

[ ]

If your application uses radio buttons, graphically indicate each button's state.

[ ]

When the user presses the SELECT button in a radio button, the radio button is armed. If the radio button was previously unset, show it in the set state.

[ ]

When the user releases the SELECT button in the same radio button where the press occurred, and the radio button was previously unset, set it and unset any other radio button in the same panel that was previously set. Disarm the radio button and, if the radio button is in a menu, unpost the menu.

[ ]

When the user presses Enter or Return in a radio button, if the radio button is in a window with a default action, activate the default action. If the radio button is in a menu:

If the radio button was previously unset, set it and unset any other radio button in the same panel that was previously set.
Disarm the radio button and the unpost menu.

[ ]

When the user presses Select or Spacebar in a radio button, if the radio button was previously unset, set it and unset any other radio button in the same panel that was previously set. Disarm the radio button and, if the radio button is in a menu, unpost the menu.

Sash

[ ]

Within a paned window, a sash adjusts the position of a separator, which adjusts the sizes of the panes next to it. As a sash is moved, make the pane in the direction of the sash movement get smaller and the opposite pane get larger by an equal amount.

[ ]

Within a sash, pressing the SELECT button or the TRANSFER button and moving the pointer causes the sash to track the movement of

the pointer. In a vertically oriented paned window, the sash tracks the vertical position of the pointer. In a horizontally oriented paned window, the pane tracks the horizontal position of the pointer.

[ ]

Pressing [uarr] or [darr] (for a sash that can move vertically) and [larr] or [rarr] (for a sash that can move horizontally) moves the sash one increment in the specified direction.

[ ]

Pressing Ctrl [uarr] or Ctrl [darr] (for a sash that can move vertically) and Ctrl [larr] or Ctrl [rarr] (for a sash that can move horizontally) moves the sash one large increment in the specified direction.

Scale

[ ]

If a scale has arrow buttons, pressing the SELECT button on an arrow button moves the slider one increment in the direction of the side of the slider on which the button was pressed and autorepeats until the button is released.

[ ]

In a scale trough, if the scale has tick marks, pressing the SELECT button moves the slider one major tick mark in the direction of the side of the slider on which the trough was pressed and autorepeats until the button is released. If the scale does not have tick marks, pressing the SELECT button in the trough moves the slider one large increment in the direction of the side of the slider on which the trough was pressed and autorepeats until the button is released.

[ ]

Within a scale slider, pressing the SELECT button and moving the pointer causes the slider to track the position of the pointer. In a vertical scale, the slider tracks the vertical position of the pointer. In a horizontal scale, the slider tracks the horizontal position of the pointer.

[ ]

Within a scale slider or trough, pressing the TRANSFER button and moving the pointer positions the slider to the point of the button press and then causes the slider to track the position of the pointer. In a vertical scale, the slider tracks the vertical position of the pointer. In a horizontal scale, the slider tracks the horizontal position of the pointer.

[ ]

If a mouse–based sliding action is in progress, pressing Cancel cancels the sliding action and returns the slider to its position prior to the start of the sliding operation.

[ ]

In a vertical scale, pressing [uarr] or [darr] moves the slider one increment in the specified direction. In a horizontal scale, pressing [rarr] or [larr] moves the slider one increment in the specified direction.

[ ]

In a vertical scale, pressing Ctrl [uarr] or Ctrl [darr] moves the slider one large increment in the specified direction. In a horizontal scale, pressing Ctrl [larr] or Ctrl [rarr] moves the slider one large increment in the specified direction.

[ ]

Pressing Begin or Ctrl Begin moves the slider to its minimum value. Pressing End or Ctrl End moves the slider to its maximum value.

Scroll Bar

[ ]

Within a scroll bar, pressing the SELECT button in an arrow button moves the slider one increment in the direction of the side of the slider on which the button was pressed and autorepeats until the button is released.

[ ]

In the trough of a scroll bar, pressing the SELECT button moves the slider one page in the direction of the side of the slider on which the trough was pressed and autorepeats until the button is released.

[ ]

Within a scroll bar slider, pressing the SELECT button and moving the pointer causes the slider to track the position of the pointer. In a vertical scroll bar, the slider tracks the vertical position of the pointer. In a horizontal scroll bar, the slider tracks the horizontal position of the pointer.

[ ]

Within a scroll bar slider or trough, pressing the TRANSFER button and moving the pointer positions the slider to the point of the button press and then causes the slider to track the position of the pointer. In a vertical scroll bar, the slider tracks the vertical position of the pointer. In a horizontal scroll bar, the slider tracks the horizontal position of the pointer.

[ ]

If a mouse–based scrolling action is in progress, pressing the `Cancel` key cancels the scrolling action and returns the slider to its position prior to the start of the scrolling operation.

[ ]

In a vertical scroll bar, pressing `[uarr ]` or `[darr ]` moves the slider one increment in the specified direction. In a horizontal scroll bar, pressing `[larr ]` or `[rarr ]` moves the slider one increment in the specified direction.

[ ]

In a vertical scroll bar, pressing `Ctrl [uarr ]` or `Ctrl [darr ]` moves the slider one large increment in the specified direction. Pressing `Ctrl [larr ]` or `Ctrl [rarr ]` moves the slider one large increment in the specified direction.

[ ]

Pressing `Page Up` or `Page Down` moves the slider in a vertical scroll bar one page in the specified direction. Pressing `Page Left` (or `Ctrl Page Up`) or `Page Right` (or `Ctrl Page Down`) moves the slider in a horizontal scroll bar one page in the specified direction.

[ ]

Pressing `Begin` or `Ctrl Begin` moves the slider to the minimum value. Pressing `End` or `Ctrl End` moves the slider to the maximum value.

Selection Box

[ ]

If your application uses a selection box, include at least a text component for the selected alternative and a list component above the text component for presenting alternatives. Use either the single selection or browse selection model. Selecting an element from the list should place the selected element in the text component.

[ ]

Make the list navigation actions `[uarr ]`, `[darr ] Ctrl Begin`, and `Ctrl End` available from the text component so the user can move the cursored element within the list and thus change the contents of the text.

Spin Box

[ ]

Present the items as a ring of items that wrap. For example, if a user is at the largest number and presses the up arrow, display the smallest number and vice versa so that the user can spin through all the items by pressing the same arrow.

[ ]

Move through the items in a spin box as shown in **Navigation in a Spin Box** .

Table 1 Navigation in a Spin Box

Movement	Keys	Example
Toward the beginning of the list	[larr], [darr]	Chronology: If Tuesday is displayed, move to Monday when the user presses [larr] or [darr] . Magnitude: If 15 is displayed, move to 14 when the user presses [larr] or [darr] .
Toward the end of the list	[rarr], [uarr]	Chronology: If Tuesday is displayed, move to Wednesday when the user presses [rarr] or [uarr] . Magnitude: If 15 is displayed, move to 16 when the user presses [rarr] or [uarr] .

Text

[ ]

In a multiline text component, pressing [uarr] moves the location cursor up one line and pressing [darr] moves the location cursor down one line. In a single−line text component, pressing [uarr] navigates upward to the previous component and pressing [darr] navigates downward to the next component, if the text component is designed to act like a basic control.

[ ]

Pressing [larr] moves the location cursor left one character and pressing [rarr] moves the location cursor right one character.

[ ]

In a text component generally used to hold multiple words, pressing Ctrl [rarr] moves the location cursor to the right by a word and pressing Ctrl [larr] moves the location cursor to the left by a word.

[ ]

In a text component generally used to hold multiple words, pressing Begin moves the location cursor to the beginning of the line and pressing End moves the location cursor to the end of the line.

[ ]

In a multiline text component, pressing Ctrl Begin moves the location cursor to the beginning of the file and pressing Ctrl End moves the location cursor to the end of the file.

[ ]

Pressing Spacebar or Shift Spacebar inserts a space in a text component. Modifying these keys with Ctrl invokes the normal selection function.

[ ]

Pressing Return in a multiline text component inserts a carriage return. Pressing Enter or Ctrl Return invokes the default action.

[ ]

In a multiline text component, Tab is used for tabbing. In a single−line text component, Tab is used either for tabbing or to move to the next field.

[ ]

If a text component supports replace mode, insert toggles between insert mode and replace mode.

By default, the component should start in insert mode, where the location cursor is between two characters. In insert mode, typing a character inserts the character at the position of the location cursor.

In replace mode, place the location cursor on a character. Typing a character replaces the current character with the newly entered character and moves the location cursor to the next character, selecting it.

[ ]

Double−clicking the SELECT button selects text a word at a time.

Gauge

[ ]

A gauge is a display−only device and should have no user interactions. The appearance of a gauge is similar to a scale, but the gauge lacks a scale slider.

Container (Control)

[ ]

Based on a task analysis, allow a container to either support a single, browse, multiple, or extended selection model.

Focus on the Container as a Whole

[ ]

If you design a container so that the container as a whole can take focus, then the following should occur:

If the container has not had focus previously, initially moving focus to the container places the active cursor only on the container as a whole (not on an element within the container).

If the container previously had focus, moving the focus to the container places the active cursor on the element that was last cursored when the container had focus −− either the container as a whole or an element within the container if it still can be cursored.

When focus is on an element within the container and the user presses `Ctrl Home`, move the active cursor to the container as a whole. If normal mode is being used, this should deselect all selected objects in the container.

When the user presses `[darr ]` or `[rarr ]` and the active cursor is on the container as a whole, move focus to the top–left element within the container that can be cursored. For information on bidirectional and vertical language, see Chapter 11.

[ ]

When a container as a whole can take focus, the following should occur:

If the user clicks the SELECT button in the background of the container, move the active cursor to the container as a whole. If select mode is used within the container, all objects should be deselected.

If the user clicks `Ctrl` SELECT in the background of the container, move the active cursor to the container as a whole without affecting the selection state of the objects within the container.

[ ]

When the container as a whole has focus, a transfer operation whose destination depends upon the focus (for example, Paste) transfers into the object represented by the container as a whole. If the elements being transferred are not objects, the application encapsulates them as objects where this is possible.

[ ]

Design a container so that the container as a whole can take focus if support for transferring objects to the container is required, unless the user can switch to a spatial view by using a graphics cursor.

Spatial Views

[ ]

In a spatial view, either use an element cursor or a graphics cursor.

Linear Views

[ ]

In a linear view, use an element cursor.

[ ]

In a linear view, when the active cursor is on an object, pressing `[darr ]` (except on the last object) moves it to the object in the row below.

[ ]

In a linear view, when the active cursor is on an object, pressing `[uarr ]` (except on the first object) moves it to the object in the row above.

[ ]

In a linear view that is neither detailed nor hierarchical, pressing [rarr] has the same effect as [darr], and [larr] has the same effect as [uarr]. For bidirectional and vertical language support, see Chapter 11.

[ ]

In a linear view, to represent an object use a small icon with the text label adjacent to the image if both are present.

[ ]

In a linear view, do not allow the icon's text label to be cursored separately from the icon image. If you want to allow the user to change the label text, support direct editing techniques, property–editing dialogs, or both.

[ ]

In a linear view, only the text label or the image of an icon needs to be shown. However, whichever one is present uniquely identifies the object.

[ ]

Support the range technique when multiple elements can be selected in a container that uses a linear view.

Linear Views with Detail Elements That Take Focus

[ ]

In a linear detailed view, either all detail elements take focus or they all do not take focus. If they do not take focus, and you want to allow the user to change the details, support direct editing techniques, property–editing dialogs, or both. Detail elements are not selectable, even when they can take focus.

[ ]

When multiple elements in a row of a nonhierarchical linear view can be cursored (because detail elements can take focus in addition to the icon in each row), follow the standard rules for directional navigation; that is, [larr] and [rarr] traverse through rows, and [uarr] and [darr] traverse through columns, with the objects treated as a single column.

[ ]

In normal mode, if keyboard navigation is used to move focus to a detail element, deselect all objects in the container.

[ ]

If the user clicks Ctrl SELECT on a detail element in the container that can take focus, move the focus to that element without affecting the selection state of the objects within the container or having any other effect.

[ ]

If the user clicks SELECT on a detail element in the container that can take focus, move the focus to that element. If select mode is used within the container, all objects are deselected as well.

Linear Hierarchical Views

[ ]

In a linear hierarchical view, each object that contains subobjects should be displayed with a display control button to its left. The button should contain a value choice that may be toggled between an *''expanded''* and a *''contracted''* state and should not take focus.

[ ]

When an object's display control button is in the contracted state, the subobjects of the object should not be shown; in the contracted state, they should be shown. All of an object's subobjects should be laid out vertically immediately below the object, but indented to the right (only the object column –– not the detail columns). If any subobject itself contains subobjects, it should also be shown with a display control button that indicates whether its subobjects should be displayed. For information on bidirectional and vertical language support, see

Chapter 11.

[ ]

When focus is on an object with a display control button and the user presses `Ctrl [larr ]`, set the button to its expanded state. If the user presses `Ctrl [rarr ]`, set the button to its contracted state. For information on bidirectional and vertical language support, see Chapter 11.

[ ]

If the user clicks the SELECT button on a display control button, toggle its state and move focus to its corresponding object icon.

[ ]

Toggling a display control button (either with a standard keyboard or mouse) has the following effect:

If the display control button toggles from the contracted to expanded state, the selection state of the objects in the container is unchanged.

If the display control button toggles from the expanded to the contracted state, objects that are no longer displayed are deselected, but the selection state of other objects is unchanged. If the active cursor is on an element that is no longer displayed, move the cursor to the object icon that corresponds to the display control button.

Context–Sensitive Help

[ ]

When the user presses `F1` or `Help`, display context–sensitive help related to the element that has focus.

[ ]

When the user is in help mode (as a result of pressing `Shift F1`, `Shift Help`, or selecting the On Item entry in the Help menu), display context–sensitive help on the element on which the user clicks the SELECT button, if it is in the same window in which help mode was initiated.

[ ]

If help is not available for the element for which it was requested, provide help for the control that contains the element, or for some higher –level grouping of controls, or for the window as a whole.

Control Navigation

[ ]

Allow control navigation to move the active cursor among controls in a tab group.

Horizontal Navigation

[ ]

When unaugmented horizontal navigation is used to navigate among controls in a tab group, the `[rarr ]` and `[larr ]` keys work as follows:

- When the user presses `[rarr ]`, move the active cursor through all the controls that can take focus in a tab group, starting from the upper left and ending at the lower right.
- If the user presses `[rarr ]` when controls are laid out in rows, traverse through one row, then traverse the row below it, and so forth.
- When the user presses `[rarr ]` at the lower right control, wrap back to the control at the upper left if and only if the tab group is not horizontally scrollable; otherwise, it should have no effect.
- When the user presses `[larr ]`, move the active cursor through all the controls that can take focus in a tab group in the exact opposite order as pressing `[rarr ]`.

[ ]

When augmented horizontal navigation is used to navigate among controls in a tab group, pressing `Ctrl [larr ]` and `Ctrl [rarr ]` move the active cursor through all the controls that can take focus in a tab group, following the same set of rules as for unaugmented horizontal navigation described previously. For information on bidirectional and vertical language support, see Chapter 11.

[ ]

Support unaugmented horizontal navigation for control navigation within a tab group, unless controls within the tab group use horizontal navigation internally.

[ ]

If controls within the tab group use unaugmented horizontal navigation internally, support augmented horizontal navigation for control navigation, unless controls within the tab group also use augmented horizontal navigation internally.

Vertical Navigation

[ ]

When unaugmented vertical navigation is used to navigate among controls in a tab group, the `[darr ]` and `[uarr ]` keys work as follows:

- When the user presses `[darr ]`, move the active cursor through the controls that can take focus in a tab group, starting from the upper left and ending at the lower right.
- When the user presses `[darr ]` and the controls are laid out in columns, traverse through one column, then traverse the column to its right, and so forth.
- When the user presses `[darr ]` at the lower right control, wrap back to the control at the upper left if and only if the tab group is not vertically scrollable; otherwise, it should have no effect.
- When the user presses `[uarr ]`, move the active cursor through all the controls that can take focus in a tab group in the exact opposite order as pressing `[darr ]`.

[ ]

When augmented vertical navigation is used to navigate among controls in a tab group, pressing `Ctrl [darr ]` and `Ctrl [uarr ]` move the active cursor through all the controls that can take focus in a tab group, following the same set of rules as for unaugmented vertical navigation described previously.

[ ]

Support unaugmented vertical navigation for control navigation within a tab group, unless controls within the tab group use vertical navigation internally.

[ ]

If controls within the tab group use unaugmented vertical navigation internally, support augmented vertical navigation for control navigation, unless controls within the tab group also use augmented vertical navigation internally.

Navigation

[ ]

If a tab group contains more than one control that can take focus, then either horizontal or vertical navigation should be supported for control navigation within the tab group.

Initial and Final Control

[ ]

The initial control of a tab group (the control that gets focus when the user presses `Tab` or `Ctrl Tab` to navigate to the tab group) should be either the top–left control that can take focus or the control with which the user is most likely to want to interact.

[ ]

In a tab group, if you allow `Tab` to navigate internally among all of its controls, the final control in the tab group should be the final control visited (without repeating) when starting at the initial control.

[ ]

If you do not allow `Tab` to navigate internally among all the controls in the tab group with wrapping, then allow `Shift Tab` or `Ctrl Shift Tab` to place focus on the initial control of the tab group.

[ ]

In a tab group, if you allow `Tab` to navigate internally among all the controls in the tab group, but do not allow warping from the final to the initial control, then allow `Shift Tab` or `Ctrl Shift Tab` to place focus on the final control of the tab group.

[ ]

If a tab group of push buttons contains the push button that has default emphasis when focus initially enters a window, then make that push button the initial control of its tab group.

Tab Groups of Push Buttons

[ ]

When focus is within a tab group of push buttons, support `Tab` in one of the following ways:

When focus is on the final push button in the tab group, move focus to the next tab group.
If the tab group is laid out in columns, but not in rows, navigate vertically to the next push button.
Navigate horizontally to the next push button.

Continue to allow the directional keys to navigate within the tab group as well.

[ ]

When focus is within a tab group of push buttons, support `Shift Tab` in one of the following ways:

When focus is on the initial push button of the tab group, navigate to the previous tab group.
Navigate in the reverse direction of `Tab`.

Copy To (Dialog Choice)

[ ]

When appropriate, the File Selection dialog should allow the user to differentiate between copying a file or object versus copying or creating a link or reference to it.

Cursor

[ ]

When using an explicit focus policy, use an element cursor on a control to indicate focus emphasis, unless a cursor is visible on some element within the control.

[ ]

Use a text cursor at the appropriate point when a control is in a state where characters typed by the user may be inserted in the control or in which characters displayed within the control may be selected.

[ ]

Use a graphics cursor in a control when keyboard users need to be able to place elements at specific (x,y) positions or to select elements within an arbitrary rectangle.

[ ]

Use an element cursor on a control to indicate focus emphasis if a cursor is displayed within that control but is not visible because it is clipped or scrolled out of view.

[ ]

A control that uses a text or graphics cursor must keep track of the cursor position even when the control does not have focus.

Mouse−Based Cursor Placement

[ ]

When the user clicks the SELECT button at a legitimate cursor position within a control that uses keyboard navigation to move the cursor, move the cursor to the location at which SELECT was clicked.

[ ]

When the user presses `Ctrl` SELECT at a legitimate cursor position within a control that uses keyboard navigation to move the cursor, move the cursor to the location at which SELECT was pressed, but do not perform any other action, unless pressing `Ctrl` SELECT already has a predefined meaning in that control.

Pointer Tracking

[ ]

Text and graphics cursors should not track the pointer; they should only move via keyboard navigation or when a mouse button is pressed, even when in a window that uses an implicit focus policy.

[ ]

An element cursor used within a control (other than a spring−sensitive control) should not track the pointer when in a window that uses an explicit focus policy.

Traversal to Obscured Elements

[ ]

When using keyboard navigation, do not skip over an element solely because it is obscured by another window or by an icon in the workspace.

Visual Guidelines

[ ]

If a control keeps track of a cursor position within the control even when the control does not have focus, do one of the following:

- Make the cursor more visible when the control has focus.
- Place an element cursor on the control as a whole when it has focus.

[ ]

If a control hides a cursor but keeps track of its position within the control when the control does not have focus, then when the control gains focus, show the cursor at the position it was in (relative to the underlying data) when the control last had focus, unless the application has changed the data or cursor position in the meantime.

[ ]

Within a selection scope that supports add mode and normal mode, change the appearance of the cursor, depending upon the mode. In particular, when an element cursor is used, draw a solid box around the element in normal mode and a dashed box around the element in add mode.

[ ]

Make sure that a text or graphics cursor used in a control is clearly distinguishable from the shape of the pointer when it is in that control.

[ ]

To make a cursor more visible when the control containing it has focus, use one of the following methods:

- Darken the cursor when the control has focus and dim the cursor when the control does not have focus.
- Use a blinking cursor when the control has focus and use a static cursor when the control does not have focus.
- Show the cursor when the control has focus and hide the cursor when the control does not have focus (except when the cursor is being used to show interacted emphasis).

Text Cursor Visual Guidelines

[ ]

When insert mode is being used, display a text cursor as a vertical bar or I–beam at the cursor position.

[ ]

The size of the I–beam cursor should be proportional to the current font.

[ ]

When replace mode is being used, use a block or underline to make the text cursor appear to be displayed on a character, though it actually is positioned immediately before that character.

Cut, Copy, Paste (Action Choice)

[ ]

Provide a Cut choice for all elements and data.

[ ]

Provide a Copy choice for all elements for which the user can create a duplicate, and for data that the user can duplicate.

[ ]

Provide a Paste choice for all objects or elements that the user can place or insert into other objects or elements.

Cut

[ ]

When the user chooses Cut, remove the objects or data to which the choice applies from their current location and place them on the clipboard.

[ ]

Display the Cut choice in a pull−down menu with unavailable emphasis when the element to which the Cut choice applies cannot be cut.

[ ]

If you provide the Cut choice on an element, place Cut in the element's pop−up menu when it can be cut.

[ ]

Assign `CtrlX` or `Shift Delete` as the shortcut key combination for a Cut choice.

Copy

[ ]

When the user chooses Copy, copy the objects or data to which the choice applies from their current location to the clipboard.

[ ]

Display the Copy choice on a pull−down menu with unavailable emphasis when the element to which the Copy choice applies cannot be copied to the clipboard.

[ ]

If you provide the Copy choice for an element, place Copy in the element's pop−up menu when it can be copied.

[ ]

When the user copies an element to the clipboard, do not change the state of the source element. For example, do not change the selection state of the element as a result of copying it.

[ ]

When the user copies an element to the clipboard, do not change the state of the copied object on the clipboard.

[ ]

Assign `CtrlC` or `Ctrl Insert` as the shortcut key combination for a Cut choice.

Paste

[ ]

When the user chooses Paste, insert the objects or data to which the choice applies from the clipboard to the current cursor location.

[ ]

Display the Paste choice on a pull−down menu with unavailable emphasis when no data is in the clipboard or when the clipboard data cannot be pasted at the cursor's position.

[ ]

If you provide a Paste menu item for an element, place Paste in the element's pop−up menu when the element or data currently in the buffer can be passed into the element at the cursor's location.

[ ]

Assign `CtrlV` or `Shift Insert` as the shortcut key combination for a Paste choice.

Data Transfer

[ ]

Allow any selectable element to be transferred.

[ ]

A copy operation should transfer the source elements to the target without affecting the original source element in any way.

[ ]

A move operation should transfer the source elements to the target and then remove them from the source.

[ ]

A link transfer operation should do one of the following:

Transfer the source element to the target, keeping it linked (in an application−specific manner) to the original source element.

Place a link to the source element (made and displayed in an application−specific manner) in the target without copying the source element.

Insertion

[ ]

The insertion point for an operation that transfers data into a selection scope is determined as follows:

For a mouse−based TRANSFER button primary transfer or drag−and−drop transfer, the insertion point is the pointer position when the TRANSFER button is released.

For a transfer invoked via a menu (popped up via the MENU button) on a selection or object, the insertion point is the pointer position when the MENU button is pressed.

If neither of the previous applies, the pointer position is the cursor position.

If the insertion position is not on an object, then do one of the following:

If insertions can happen only at a fixed place in the selection scope, the insertion point should be placed at that point (for example, append−only text).

If the elements in the selection scope are ordered in some way, the insertion point should be moved (for each element inserted) so that the element may be inserted at the proper position.

[ ]

If the insertion point is on a selection, but not on an object, a transfer operation into the scope should have one of the following effects:

If a link operation, link the selection to the source elements, if such links are supported by your application.

If pending delete behavior is enabled, replace the contents of the selection by the transferred data or link (except possibly for a primary copy at the edge of the primary selection).

Insert the transferred data or link at the insertion point.

[ ]

If the insertion point is neither on a selection nor on an object, a transfer operation should insert the transferred data or link at the insertion point.

[ ]

If the elements are laid out in a linear manner, insert elements immediately prior to the insertion point.

Exclusions

[ ]

If the target insertion point is within the region of the source elements, do not perform the transfer; instead, issue a warning signal.

[ ]

If an operation that transfers data into a scope is selected from a menu popped up (via the MENU button) on neither an object nor a selection, issue a warning message if the insertion point is on an object.

Rules

[ ]

When an object is moved or copied, transfer the contents of the object along with the object itself.

[ ]

When a move operation is performed, remove the source elements only if the data transfer has completed successfully. If the data transfer operation has been canceled or it has failed, do not remove the source elements.

[ ]

Do not select data transferred to a target as a result of a link or copy operation.

[ ]

If the source elements used in a data transfer operation are selected, do not deselect them as a result of the operation.

[ ]

Do not change the source elements used as the source of a copy or link operation.

[ ]

When transferred data is inserted into a selection scope (not into objects within the scope) by using normal mode at a position disjoint from a selection, deselect the selected elements.

[ ]

Inserting or pasting elements at a location disjoint from the selection should have no effect on the selection in add mode.

[ ]

If a transfer operation uses an insertion point based on the position of the pointer (rather than the cursor), place the cursor at the pointer position used.

Data Format

[ ]

During a transfer operation, allow the data transferred to change its appearance or representation to suit the requirements of the target.

Transfer Option

[ ]

Allow the user to override the default operation of a TRANSFER button–based data transfer operation by using a modifier key.

TRANSFER Button Overrides lists these operations and overrides.

Table 1 TRANSFER Button Overrides

Operation	Modifier Override
Move	`Shift`
Copy	`Ctrl`
Link	`Ctrl Shift`

Default Action

[ ]

To support new users and keyboard−only users, provide default actions as shortcuts only for application features that are available by some other method.

[ ]

When a window has focus, show default emphasis on the push button in the window, if any, that performs the current default action for that window.

[ ]

Except when focus is on a tool or object or in a menu, if the default action of a window varies, provide a push button that corresponds to each default action.

Default Activation

[ ]

When the focus is on a push button, make its action the default action.

[ ]

When the focus is on a choice in a menu other than a tear−off menu, make the default action activate or toggle that choice.

[ ]

When focus is on an object icon, the default action should present data of the corresponding object; for a window icon display the corresponding window.

Default Action Behavior

[ ]

When the user presses `keypadEnter`, `Ctrl Enter`, or `Enter` (except in text), invoke the default action. Except possibly when on an element that is both activatable and selectable, it should have no other effect.

[ ]

When the user double−clicks the SELECT button and focus is on a value choice that is part of a mutually exclusive group of value choices (for example, on a radio button, a tool, or in a value set), invoke the default action.

Delete (Action Choice)

[ ]

If only objects are selected, the Delete choice should not be available in the Edit menu. Provide Delete to Trash in the Selected menu instead. If the selection scope includes only objects, then do not provide Delete in the Edit menu.

Region Deletion

[ ]

In an editable selection scope, any operation that deletes a region (not containing objects) has the following effect if something is currently selected in the scope:

In normal mode

Deletes the selection.

In add mode

If the cursor is not outside of the selection, deletes the selection.

In add mode

If the cursor is outside of the selection, deletes the specified region, possibly including part of the selection.

[ ]

In an editable selection scope, pressing `Delete` and `Backspace` have the following effects if the cursor is within text and the selection is not empty and contains no objects:

In normal mode

Deletes the selection.

In add mode

If the cursor is not outside of the selection, deletes the selection.

In add mode

If the cursor is outside of the selection, pressing `Delete` deletes the character following the cursor and pressing `Backspace` deletes the character preceding the cursor.

[ ]

When nothing is selected in an editable selection scope, and there is a character following the text cursor, pressing `Delete` deletes it.

[ ]

When nothing is selected in an editable selection scope, and there is a character preceding the text cursor, pressing `Backspace` deletes it.

Dialog (Choice Type)

[ ]

Use a dialog choice to display a secondary window that contains a dialog from which the user can specify parameters and invoke the action corresponding to the choice or cancel the action.

[ ]

Use ellipses (...) after the textual label of a dialog choice.

[ ]

Use ellipses after a choice in a menu or on a push button that causes a secondary window to be displayed, if the secondary window gathers additional information required to complete the requested task. For example, provide a Print... choice to allow the user to specify print characteristics before starting the print operation.

[ ]

Do not use ellipses if a choice results in immediate action. In some cases, the action may be to display a secondary window that supports interaction with the user. For example, do not use an ellipsis on the Help choice, even though the help information is displayed in a secondary window.

[ ]

Do not use ellipses simply because a confirmation message is displayed.

Dialog (Window)

[ ]

Use a dialog to allow the user to specify parameters to an action that is to be performed.

[ ]

Use a dialog to allow the user to review the parameters to an action before finally approving its invocation.

[ ]

Provide one or both of the following push−button choices in a dialog:

OK

Invokes the action and closes the dialog window.

Apply

Invokes the action and leaves the dialog window open.

You can replace the labels of these choices with ones that are descriptive of the action to be performed.

[ ]

Provide a Cancel choice in a dialog window that closes the dialog without performing the effects of the action.

Direct Editing

Entering Direct Editing Mode

[ ]

If direct editing mode is supported for a textual label, enter the mode when the user clicks the SELECT button on the label or when the user double−clicks the SELECT button, if (single) clicking the SELECT button is already used for some other purpose (for example, margin selection).

[ ]

If direct editing mode is supported for a textual label, provide some means for keyboard users to edit the label as well (for example, by choosing an action from a menu that displays a window in which such labels can be edited via a text−entry field).

[ ]

Use a persistent selection to select the entire text of the label when entering direct editing mode.

Exiting from Direct Editing Mode

[ ]

Exit from direct editing mode when the user presses `Enter` or `keypadEnter`.

[ ]

Exit from direct editing mode and undo any changes made to the text being edited when the user presses `Cancel`.

[ ]

When the user exits from direct editing mode, deselect the text in the label.

[ ]

When the user exits from direct editing mode without explicitly moving focus elsewhere, and the focus remains in the window or is moved back to it, place the active cursor as follows:

If a static text control is being directly edited, place the active cursor on it.
If the label being directly edited is associated with all or part of another element (for example, an icon) that can be cursored with an element cursor, move the active cursor to that element.
If the label being directly edited is a group heading, navigate to the group.
If none of the previous statements apply, the active cursor should return to where it was before entering direct editing mode.

[ ]

During direct editing, mouse buttons have their usual effect, and all operations on other controls except those that transfer data to the label cause direct editing mode to exit.

[ ]

During direct editing, keyboard−based window navigation should exit from direct editing mode and move focus to the window indicated.

[ ]

During direct editing, control navigation should not be supported.

[ ]

During direct editing, tab−group navigation should exit from direct editing mode and allow navigation elsewhere, but only in the following cases:

When a static text control is being directly edited, tab group navigation should allow navigation in the same way as when the control ordinarily has focus.

If the label being directly edited is associated with all or part of another element (for example, an icon) that can be cursored with an element cursor, tab navigation should tab away from that element.
If the label being directly edited is a group heading, allow tab navigation to tab away from the group.

Direct Manipulation

Initiating the Operation

[ ]

Initiate a direct manipulation operation when the user presses a pointer button on an element.

During the Operation

[ ]

Do not change the active window while a direct manipulation operation is in progress.

[ ]

While a direct manipulation operation is in progress, if the action to be performed on termination changes as the pointer is moved or as the state of the keyboard changes, provide appropriate feedback to the user so that the user knows what action will be performed.

Terminating the Operation

[ ]

If a direct manipulation operation requires that buttons be pressed while the operation is in progress, terminate the direct manipulation operation. Alternatively, change its mode when the user releases all pointer buttons and does not press any more buttons or keys within a time period specified by the operating environment.

[ ]

If a direct manipulation operation does not require that any button be pressed while the operation is in progress, allow the user to terminate the direct manipulation operation. Alternatively, change its mode by pressing shortcut keys or by clicking on some combination of pointer buttons.

[ ]

When the user presses `Cancel` or `Esc` while a direct manipulation operation is in progress, cancel the direct manipulation operation.

Drag–and–Drop Transfer

[ ]

Support dragging any object represented as an icon.

[ ]

Support dropping onto objects represented as icons.

[ ]

If a control supports any form of data transfer with its contents used as a source, then those contents can be the source of a drag–and–drop operation.

[ ]

If a control allows any form of data transfer to the control, it can be used as a target for a drag–and–drop operation.

[ ]

Make the source of a drag–and–drop transfer the element or selection on which the drag was initiated.

[ ]

Make the target of a drag–and–drop transfer the control on which the drop was performed; if the control holds a selection scope, then it is the insertion point within that scope.

[ ]

The default operation for a drag–and–drop transfer must be a move, except in the following cases:

When the source is not editable, make copy the default.

When the destination is an external device (for example, a printer, diskette, or mail out basket), make copy the default.

When the destination is an application that manipulates the data (for example, a compiler), consider making copy the default.

[ ]

The default operation for a drag–and–drop transfer may be overridden by `Ctrl` and/or `Shift` modifiers when the TRANSFER button is released.

Separated Transfer

Use the following guidelines when TRANSFER is a separate mouse button from SELECT.

[ ]

Initiate a drag when the user presses the TRANSFER button and then moves the mouse on selections and elements that can be dragged.

[ ]

If the user uses the TRANSFER button to drag an element that has not been selected, drag only that element without affecting any existing selection.

[ ]

Drag all selected elements in the scope when the user presses the SELECT button and then moves the mouse on a selection.

[ ]

Initiate a drag when the user clicks `Alt` TRANSFER on a selection or an element that can be dragged.

Integrated Selection and Transfer

[ ]

Whether the TRANSFER button is integrated with SELECT on MB1 or not, when the user clicks SELECT, optionally augmented with `Shift` and/or `Ctrl`, the effect should be the same.

[ ]

When the user multi−clicks (or multi−presses) the SELECT button, optionally augmented with `Shift Ctrl`, the effect should be the same whether the TRANSFER button is integrated with SELECT on MB1 or not.

Use the following guidelines when the TRANSFER button is the same button as SELECT.

[ ]

When the user presses MB1 and moves the pointer on a selection, drag all selected elements in the scope.

[ ]

When the user presses MB1 on an element that can be activated or toggled (other than a tear−off choice), activate or toggle the element; do not allow dragging.

[ ]

When the user presses MB1 and moves the pointer on an element that cannot be selected (other than a tear−off choice), do not allow dragging.

[ ]

If the MENU button is bound to a mouse button other than MB1, when the user clicks `Alt` MB1, optionally augmented with `Shift` and/or `Ctrl`, on a selection or an element that can be dragged, initiate a drag.

Integrated Selection and Transfer in the Background

[ ]

Whether the TRANSFER button is integrated with SELECT on MB1 or not, support the same effect when the user presses SELECT and moves the pointer, optionally augmented with `Shift` and/or `Ctrl`, in the background (or on unselected text) of a scope that supports area or range selection techniques.

Integrated Selection and Transfer on Unselected Elements

[ ]

When the TRANSFER button is integrated with SELECT on MB1, when the user presses MB1 and moves the pointer, optionally augmented with `Shift` and/or `Ctrl`, on an unselected element for dragging, the effect should be equivalent to the user clicking the SELECT button (without modifiers) on the element (using whatever selection model is in force) and then dragging the selection.

[ ]

When the TRANSFER button is integrated with SELECT on MB1 in a nontextual scope, treat pressing MB1 and moving on an unselected element as a selection technique if the following is true:

Area or range selection techniques are supported and the scope is densely populated (in other words, there is not much background space available)

Different selection techniques are used (for example, touch versus area techniques) depending upon whether MB1 is pressed on an element versus in the background

Otherwise, allow the user to press MB1 and move the pointer on an unselected element for dragging.

Completing the Drag

[ ]

The application that contains the source elements of a drag−and−drop operation may limit drops to windows that it manages. It may also

treat elements of its own or other applications as invisible for the purpose of determining the target.

[ ]

When a drag is completed, a drop should be performed at the pointer location as follows:

Where the pointer is when the TRANSFER button is released, if it was pressed while the drag was in progress
Where the pointer is when the TRANSFER button is clicked, if it was not pressed while the drag was in progress
Where the user presses `Return` or `Enter`

In any case, the state of the `Ctrl` and `Shift` modifiers at the time of the drops determines the operation to be performed.

[ ]

When a drag is in progress and the user presses `Cancel`, cancel the drag.

Dropping an Element

[ ]

When the user attempts to drop an element on an element that does not support the operation, cancel the drag or display a message dialog similar to the one that would be used if `Help` were pressed at that location.

[ ]

When a user drops a source element onto the same position it currently occupies, the application should not perform an action.

[ ]

If the source elements of a drag−and−drop move operation are selected, select the elements transferred to the target scope if the source and target scopes are the same.

[ ]

If the source elements of a drag−and−drop move operation are selected, select the elements transferred to the target scope if the target uses normal mode and allows the number of elements transferred to be selected.

[ ]

Move focus to the target control in a drag−and−drop operation.

Drag/Move (Action Choice)

[ ]

Include Move in the window menu. When used from a window, move the window, and when used from a window icon, move the icon.

[ ]

In the File and Edit menus, use Move if the selected elements can only be moved; use Drag if other operations are possible as well.

[ ]

When the user is moving a window, the only visible target should be the workspace. That is, a window can be dropped anywhere, but should always be treated as being dropped on the workplace, not on some other element.

[ ]

When the user initiates a drag−and−drop operation by selecting Drag or Move from a menu, and the pointer position after choosing the menu entry is outside the bounding box of the element or elements being moved, or pointer warping is not enabled, then do one of the following:

Start the drag at the current pointer position and drag the elements there.

Either wait until the user moves the pointer into the bounding box (within the window) of the elements being moved and start dragging them at that point, or wait until the user presses the directional keys to move the elements.

Drop–Down Combination Box (Control)

[ ]

If the drop–down combination box has focus and the list box is displayed, remove the list box if the user presses `Cancel` or `Escape` and leave focus in the text–entry field.

[ ]

Do not allow the list box in a drop–down combination box to take focus.

Drop–Down List (Control)

[ ]

If the drop–down combination box has focus and the list box is displayed, remove the list box if the user presses `Cancel` or `Escape` and leave focus in the text–display field.

Behavior

[ ]

Do not allow the list box in a drop–down list to take focus.

Edit (Menu)

[ ]

In conjunction with the **Menu Guidelines** reference page, use Edit Menu Choices to decide which choices to include in the Edit menu and how to organize them.

Table 1 Edit Menu Choices

Mnemonic	Menu Choice	Keyboard Function
T	Cut	Cut, `Ctrl X`, `Shift Delete`
C	Copy	Copy, `Ctrl C`, `Ctrl Insert`
P	Paste	Paste, `Ctrl V`, `Shift Insert`
D	Delete	Delete

[ ]

For operations with multiple shortcut keys, show the one that augments a printing character with `Ctrl` (for example, `Ctrl A` for Select All).

[ ]

Do not display the shortcut keys for Primary Copy or Primary Move in their respective menu entries.

Guidelines for Specific Menu Entries

[ ]

When the user selects the Select Pasted choice, select the element last transferred into the selection scope.

[ ]

Allow a Reselect choice only when nothing is selected in the current selection scope. When the user chooses Reselect, reselect the previously selected elements within a selection scope.

[ ]

Provide a Duplicate choice to copy the selected elements in the same selection scope. If the elements are named, selecting Duplicate should automatically generate names for them that do not conflict with the names of other objects in the scope.

[ ]

Provide an Insert choice to allow the user to specify a file or object to be included at the cursor position. Either the name, reference, or contents can be included. If multiple choices are reasonable, the dialog should allow the user to determine the choice. If different kinds of elements can be inserted, then either provide a cascaded choice or make Insert a menu–bar item.

Emphasis (Cue)

In−Use Emphasis

[ ]

Remove in−use emphasis from an icon for an object when all windows that contain a view of that object have been closed and all tasks that use that object have completed.

Selected Emphasis (Cue)

[ ]

Display selected emphasis on the primary selection, whether or not the control containing it has focus.

[ ]

Display selected emphasis on a persistent selection when the control containing it has focus.

Unavailable Emphasis (Cue)

[ ]

Display unavailable emphasis on choices or controls that represent operations that cannot be activated in the current context.

[ ]

If a choice is never available to a particular user, do not display the choice rather than displaying it with unavailable emphasis. For example, if the system administrator assigns read−only access to a user, destructive choices such as Cut or Delete should not be displayed.

[ ]

Do not include unavailable choices in lists and option menus.

Target Emphasis (Cue)

[ ]

During a drag−and−drop operation to a target element in which a drop is likely to succeed, display target emphasis on the target element when the pointer is on that element.

Ready Emphasis

[ ]

Display ready emphasis on a control if the action will be activated or toggled when the user completes the current action.

[ ]

When the pointer is located on an element that represents a choice and the user presses the SELECT button, display ready emphasis to indicate that releasing SELECT activates or toggles the choice. Modify the emphasis as follows if the user moves the pointer:

If the pointer is moved outside of the element while SELECT is pressed, change the element's appearance back to its normal condition.

If the pointer is moved back inside the element while SELECT is still pressed, redisplay the ready emphasis on the element to indicate pending activation or toggling.

[ ]

Remove the ready emphasis from the selected element when the user moves the pointer from the element, whether or not the action that would activate or toggle the choice was completed or when the user cancels the action.

Default Emphasis

[ ]

Display default emphasis on the push button whose action corresponds to the current default action.

Interacted Emphasis (Cue)

[ ]

In explicit mode, display interacted emphasis on the last control within a selection scope that had input focus.

[ ]

In implicit mode, display interacted emphasis on the last control within a selection scope that received input.

Focus Emphasis (Cue)

[ ]

When an explicit focus policy is in use, use focus emphasis to indicate which control has focus.

Exit (Choice)

[ ]

Provide the Exit choice in the File menu. If you do not provide a File menu, make Exit the last choice in the pull−down menu for the first menu−bar item on the menu bar.

[ ]

Provide the Exit choice to allow the user to end the current application.

[ ]

When the user chooses the Exit choice, prompt the user to save any unsaved changes before ending the current application.

File Menu

[ ]

If you provide any of the functionality specified for the choices defined for a File menu, provide a File choice on the menu bar.

[ ]

In conjunction with the **Menu Guidelines** reference page, use File Menu Choices to decide which of the choices listed to include in the File menu and how to organize them.

Table 1 File Menu Choices

Mnemonic	Menu Choice	Keyboard Function
N	New	`Ctrl N`
O	Open ...	`Ctrl O`
S	Save	`Ctrl S`
A	Save As ...	
X	Exit	

Guidelines for Specific Menu Entries

[ ]

When the user chooses Revert, modify the displayed state to revert to the state last saved or, if the state has not been saved, to the initial state.

File Selection (Dialog)

[ ]

Allow the user to continue interacting with the application when a file selection dialog is displayed.

[ ]

The file selection dialog box must be composed of at least the following:

A text component for displaying and editing a directory mask (or filter) used to select the file to be displayed. The directory mask must be a string that specifies the base directory to be examined and a search path.

A list component for displaying file names. The list should display all files and subdirectories in the base directory that match the search pattern. The list must use either the single or browse selection model.

A text component for displaying and editing a file name.

A group of push buttons labeled OK, Filter, Cancel, and Help. If the file selection dialog is used to perform a specific action to the selected files, OK can be replaced by a label that describes the action to be done.

[ ]

The user must be able to select a new directory to examine by scrolling through the list of subdirectories and selecting the desired directory or by editing the directory mask. Selecting a new directory from the directory list must not change the search pattern. A user must be able to select a new search pattern by editing the directory mask.

[ ]

Allow the user to navigate the text within each list component by pressing `[uarr ]`, `[darr ]`, `Ctrl Begin`, and `CtrlEnd`. Allow the navigation actions `PageUp` and `PageDown` to move the cursored element within each list.

[ ]

The contents of the directory in the text−entry field must correspond to the contents of the directory list. The contents of the file name text must also correspond to the contents of the file name list.

[ ]

The user must be able to select a file by scrolling through the list of file names and selecting the desired file or by entering the file name directly into the text−entry field. Selecting a file from the list causes that file name to appear in the file selection text−display field.

[ ]

The user can select a new file as many times as desired. The application must not process the file until one of the following events occurs:

The user activates the OK button.

The user presses `Enter` or `Return` while the file name text−entry field has focus.

The user presses `Enter` or `Return` while the cursor is on an item in the file list.

The user double−clicks the SELECT button on an item in the file list.

[ ]

The file selection dialog must initiate a directory and file search when any of the following occurs:

The file selection dialog box is initialized

The user activates the Filter push button

The user double−clicks the SELECT button or presses `Enter` or `Return` on an item in the directory list

The user presses `Enter` or `Return` while the directory mask text−entry field has focus

Find (Choice and Dialog)

[ ]

Provide a Find choice in the Edit menu to search for elements that match specified criteria. These elements may either be:

- Within the view itself
- Within objects that are represented in the view
- In other objects, with the data in the view used, in part, to identify the other objects and/or the search criteria

[ ]

When Find is chosen, display a Find dialog window that allows a user to specify the parameters of the search. For example, when using the Find choice to search for text, allow a user to specify the text to be found, the direction of the search, sensitivity to text capitalization, and similar parameters.

Finding Matches

[ ]

If Find is used to find one match at a time, and more than one match meets the criteria, keep the Find dialog displayed.

[ ]

When the user specifies new criteria (and matches are to be found one at a time) and chooses the Find push button, find the first match and display it in the dialog.

[ ]

When the Find dialog remains displayed after the user chooses Find within it, find the next match when the user chooses Find again.

First−Letter Cursor Navigation

[ ]

Provide first−letter cursor navigation in a selection scope only if the selection scope does not support a keystroke−based mechanism such as mnemonic navigation.

Behavior

[ ]

Skip white spaces that begin a textual label to determine the first character of a textual label used to match an item.

Key Pressing

[ ]

When a user types a printing character in a scope by using first−letter cursor navigation, move the location cursor in the scope to the next item whose textual label starts with that letter.

[ ]

When a user enters a printing character that has no valid match, do not move the location cursor in the selection scope.

With Combination Text−List Controls

[ ]

When a printing character entered into a text−display field that has focus is used for first−letter cursor navigation of an associated list, select the item matched and place its text contents into the text−display field. If there is no valid match, generate a warning signal and do not make any changes to the appearance of the list or text−display field.

[ ]

When a printing character entered into a text−display field that has focus is used for first−letter cursor navigation of an associated list, and the list is not displayed, do one of the following:

If pressing `[darr ]` would display the list, make entering the character display it as well.

Navigate in the list as indicated and select the cursored list item, displaying it in the text field without displaying the list.

Gauge (Control)

[ ]

When the value of the gauge cannot be directly changed, use a zero−width slider arm. Fill the track between the position of the minimum value and the current value to achieve a thermometer−like effect.

[ ]

When the value of the gauge can be directly changed, do the following:

- Use a slider arm whose appearance provides some indication (such as a thumb or handle) that it can be manipulated.
- Fill the track between the position of the minimum value and the slider arm to achieve a thermometer−like effect.
- Use the same pattern or color in the trough as in the slider arm so that they appear to be one continuous whole.
- Place the slider arm so that the edge opposite to the filled area is at the position of the current value.

[ ]

Do not display arrow buttons in a gauge if the value in the gauge cannot be changed.

[ ]

All appropriate guidelines that apply to sliders apply to gauges whose values can be directly changed by the user.

Group Box

[ ]

When you use a group box, provide a group heading as well.

[ ]

Do not allow a tab group to include elements from both inside and outside a group box.

Help (Menu/Action Choice)

[ ]

Provide a Help menu−bar item in the menu bar of all windows that have a menu bar and provide help information.

[ ]

In conjunction with the **Menu Guidelines** reference page, use Help Menu Choices to decide which choices to include in the Help menu and how to organize them.

Table 1 Help Menu Choices

Mnemonic	Menu Choice	Keyboard Function
U	Using Help	
A	About *application−name*	

[ ]

Do not display the shortcut key for On Item (for example, `Shift F1` or `Shift Help`) in the menu entry if it would significantly increase the width of the Help menu.

On Element

[ ]

When the user chooses On Item, initiate context−sensitive help mode by changing the shape of the pointer to the question mark pointer. When the user moves the pointer to the element for which help is needed and clicks the SELECT button, exit from the mode and present any context−sensitive help available for the element. When the user presses `Cancel` or `Escape`, exit from the mode without presenting help.

Icon

[ ]

Use an icon to represent objects that the user can place in a container.

[ ]

Define an icon to consist of an image and an optional textual label as follows:

Image

Use a common, task−related symbol associated with the object it represents.

Label

If you provide a label, place the label below the image. If the image is small relative to the size of the label, place the label to the right of the image. For information on bidirectional and vertical language support, see Chapter 11.

[ ]

Ensure that the image has a border that prevents the image and the background from merging.

Designing Small Icons

[ ]

Limit the detail in a small icon to avoid a cluttered appearance. For example, do not display a count of the number of objects in the container.

Icon Labels

[ ]

If an icon represents a named object, label the icon with the name of the object.

[ ]

If direct editing can be used to change the textual label of an icon, include a Properties choice in the icon's pop−up menu, and include a text−entry field in the properties window through which the label can be changed.

In−Progress Message

[ ]

An in−progress message in a secondary window should contain at least a graphic image, a textual label or message, and the OK and Close choices.

[ ]

Do not interfere with the user's interaction with an application just because an in−progress message is being displayed; for example, normally make the in−progress message window a modeless window.

[ ]

Close an in−progress message window when a request made by a user completes normally.

[ ]

If information that would have been displayed in an information message is displayed in an in−progress message, do not close the window until the user explicitly requests to close it, such as by activating the Close push button.

Providing Controls in In−Progress Messages

[ ]

Provide a Close push button to allow a user to close the in−progress message window without affecting the task.

Visuals

[ ]

Display the hourglass symbol at the left of an in−progress message. For information on bidirectional and vertical language support, see Chapter 11.

Include (Choice and Dialog)

[ ]

When Include is chosen, display an Include dialog window that allows the user to specify a subset of elements to be displayed within the view.

[ ]

When an element is added to the data on which a view filtered by an include action is displayed, and the element matches the criteria for the view, display the element in the view.

Information Message

[ ]

Display an information message when a situation has occurred that the user may need to know about, but which has no follow−up actions the user can request.

[ ]

Display an information message when there is additional information about the status of normal completion of a task or about the current status of the task.

[ ]

If an in−progress message for the task is currently displayed in its own secondary window, do not use an information message. Instead, augment the in−progress message window with the message information.

[ ]

Display the information symbol appropriate to your operating environment to the left of the information message. For information on bidirectional and vertical language support, see Chapter 11.

[ ]

Design the information message window to be modeless.

Providing Controls in Information Messages

[ ]

Provide an OK push button that allows a user to acknowledge the information displayed in the information message and to remove the message.

Visuals

[ ]

Display the (i) symbol as the graphic image at the left of an information message. For information on bidirectional and vertical language support, see Chapter 11.

Input Focus

[ ]

When the user has set the input focus in the workspace to track the pointer, use an implicit focus policy to move focus from window to window by always ensuring that the window under the pointer has the focus.

[ ]

When the user has set the input focus in the workspace to be explicitly changed, use an explicit focus policy to move focus from window to window.

[ ]

When the user has set the input focus within a window to track the pointer, use an implicit focus policy to move focus from control to control within that window by always ensuring that the control under the pointer has the focus.

[ ]

When the user has set the input focus within a window to be explicitly changed, use an explicit focus policy to move focus from control to control, except within controls that are spring sensitive.

[ ]

When an explicit focus policy is in use in a workspace, and the user presses the SELECT button (optionally augmented by `Shift` and/or `Ctrl`) in a window, move the focus to that window.

[ ]

When an explicit focus policy is in use in a window, and the user presses the SELECT button (possibly augmented by `Shift` and/or `Ctrl`) in a control that can take focus, in addition to performing any action the control binds to the button press, move the focus to the control, unless the control is used to change the visible portion of some area (for example, a scroll bar).

[ ]

When a window that uses an explicit keyboard focus policy acquires focus through either keyboard navigation or a button press (but not by pressing SELECT [possibly augmented by `Shift` and/or `Ctrl`] in a control in that window) causing that control to take focus, then move the focus to the control that last had focus in the window if the following are true:

- The window had the focus at some time in the past (whether or not it has been minimized since then)
- The control that last had focus in the window is still able to receive input focus

Otherwise, move focus to either of the following:

- The control with which the user is most likely to want to interact
- The initial control of the first tab group in the window

[ ]

Do not change the state, appearance, or behavior of a control when the user uses the keyboard to move the input focus to the control, except for showing focus emphasis. For example, when the user moves through a set of radio buttons with the directional keys, do not change the state of any of the radio buttons.

[ ]

When the user presses `Ctrl` SELECT on a control that can take focus, move focus to the control but do not perform any other action unless pressing `Ctrl` SELECT already has a predefined meaning in that control.

Input Models (CDE)

Keyboard Focus Model

[ ]

Only one window at a time can have keyboard focus. Highlight the window that has focus. Within the window that has keyboard focus, only one component at a time can have focus.

The keyboard focus determines which component on the screen receives keyboard events. This rule prevents confusion about which window and component have focus.

[ ]

If your application uses an explicit focus policy, when the user presses the SELECT button, do not move focus to a component that is not traversable or does not accept input.

An explicit focus policy requires the user to explicitly select which window or component receives keyboard focus. Generally, the user gives focus to a window or component by pressing the SELECT button over it. However, this policy must not allow the user to give focus to a component that is not traversable or does not accept input.

[ ]

If your application uses an explicit focus policy, highlight (with a location cursor) the component with the keyboard focus.

The user needs to know the location of the keyboard focus to be able to control an application.

Input Device Model

[ ]

Support methods of interaction for keyboard–only users. All features of your application should be available from the keyboard.

Some users may not have access to a pointing device. These users need to be able to access the full functionality of the application from the keyboard. Also, advanced users may use the keyboard to perform some tasks more quickly than with a pointing device.

[ ]

Use the following bindings for mouse buttons:

SELECT

Used for selection, activation, and setting the location cursor, SELECT is the leftmost button, except for left–handed users, where it can be the rightmost button.

TRANSFER

Used for moving and copying elements, TRANSFER is the middle mouse button, unless dragging is integrated with selection or the mouse has fewer than three buttons.

MENU

Used for popping up menus, MENU is the rightmost button, except for left–handed users, where it can be the leftmost button, or unless the mouse has fewer than three buttons. If the mouse has one button, bind the MENU button to `Alt` SELECT.

[ ]

Do not warp the pointer unless you have given the user a means of disabling the behavior.

The pointer position is intended only as input to applications, not as an output mechanism. An application warps the pointer when it changes the pointer's position. This practice is confusing to users and reduces their sense of control over an application. Warping the pointer can also cause problems for users of absolute location pointing devices (for example, a puck on a graphics tablet).

[ ]

Components or applications developed for CDE must adhere to Motif key bindings (see Appendix A).

Internal Navigation

[ ]

If a control uses internal navigation to move an element cursor among elements within a control, follow all the rules for control navigation, proceeding as if the elements in the control were controls within a tab group.

Graphics Cursor

[ ]

Allow the directional keys to move a graphics cursor one unit in the indicated direction, where a unit is generally a pixel.

[ ]

Allow the directional keys augmented by `Ctrl` to move a graphics cursor multiple units in the indicated direction.

[ ]

Allow `Home` (or `Begin`) and `End` to move a graphics cursor to the leftmost and rightmost ends, respectively, of the data being viewed in the control. For information on bidirectional and vertical language support, see Chapter 11.

[ ]

Allow `Ctrl Home` (or `Ctrl Begin`) and `Ctrl End` to do one of the following:

Move the graphics cursor to the topmost and bottommost ends, respectively, of the data being viewed (recommended).

Move the graphics cursor to the top leftmost and bottom rightmost ends, respectively, of the data being viewed. For information on bidirectional and vertical language support, see Chapter 11.

[ ]

Do not allow a graphics cursor to wrap from one edge to the opposite edge.

Positional Indicator

[ ]

When the purpose of a control is to allow a positional indicator to be set to a value (for example, an arm in a slider), then when the control has focus, allow directional keys to move the positional indicator one unit (not necessarily one pixel) in the corresponding direction.

[ ]

When the purpose of a control is to allow a positional indicator to be set to a value, allow directional keys augmented by `Ctrl` to move the positional indicator multiple units in the indicated direction.

Home and End Keys with Positional Indicators

[ ]

If a control with a positional indicator has only one degree of freedom, then do one of the following:

Make both vertical and horizontal navigation have the same effect.

Support either vertical *or* horizontal navigation.

[ ]

If a control with a positional indicator has only one degree of freedom, then allow `Home` (or `Begin`) to move the positional indicator to its initial position, and `End` to move it to its final position.

[ ]

If vertical and horizontal navigation have different effects on a positional indicator, then allow `Home` (or `Begin`) to move the indicator as far as it can go toward the initial or final horizontal positions. Allow `Ctrl Home` (or `Ctrl Begin`) and `Ctrl End` to move the indicator

as far as it can go towards the initial or final vertical position.

Changing Values

[ ]

If you use horizontal or vertical navigation to switch to a value from among a linear sequence of values, then make `Home` (or `Begin`) switch to the first such value, and `End` switch to the last such value.

Keyboard (Device)

[ ]

Use equivalent, though not necessarily identical, mouse and keyboard techniques to provide access to all functions.

[ ]

Use `Ctrl`, `Shift`, and `Alt` only to modify the function of other keys or key combinations on the keyboard or pointing–device actions.

[ ]

Use `Alt` in conjunction with printing characters only to provide access to mnemonics in an application or for predefined actions. For information on bidirectional and vertical language support, see Chapter 11.

[ ]

If the functions listed in Appendix B are provided, assign them to the keys listed in the table. Do not assign these functions to other keys. Only the user should reassign them.

Label

[ ]

If a label contains a significant amount of text and the user is likely to want to use it as a data transfer source, provide a means for both mouse and keyboard users to do so.

[ ]

Do not treat a static text control as a tab group.

[ ]

Do not support internal navigation or selection within a static text control.

Layout

[ ]

If an element contains a choice described by a textual label, place the label within the element (for example, a push button or radio button) unless some other textual information needs to be placed there (for example, an option menu button).

Quick Transfer

[ ]

If a label supports direct editing, support quick transfer to transfer the range of text identified by the quick transfer operation.

List Box (Control)

Behavior

[ ]

When the cursor is on a list item that labels a viewable object, make presenting that object the default action.

[ ]

Design list boxes with either a single, browse, multiple, or extended selection model.

Navigation

[ ]

In a scrollable list box, do not allow the cursor to wrap.

[ ]

Display the active cursor as an element cursor on the appropriate list item.

[ ]

When the user presses `[uarr ]` or `[darr ]`, navigate to the previous and next items in the list box, respectively.

[ ]

If a list box can be scrolled horizontally, make it a tab group.

[ ]

If a list box can be scrolled horizontally, then when focus is in the list box, allow `[larr ]` and `[rarr ]` to scroll left and right by one character width, and `Ctrl [larr ]` and `Ctrl [rarr ]` to scroll left and right by a larger amount.

Visuals

[ ]

Provide the list box with an area for presenting a list of labels as items.

[ ]

Provide vertical scroll bars when some of the list items are not visible in the list box.

[ ]

Arrange the list items vertically.

[ ]

Use different cursor visuals when the list box is in add mode from the one used in normal mode.

Margin Selection Techniques

[ ]

If margin elements can take focus, place them in a separate selection scope and a separate tab group from the selection scope they control.

[ ]

After using a margin point technique, do the following:

- Define the selection region to consist of the elements corresponding to the margin element that was used.
- Place the anchor point at the point in the region nearest to the margin element.
- Identify the anchor element to be the element, if any, within the selection region nearest to the anchor point.
- Define the anchor region to be the selection region.
- If the margin element cannot take focus, place the active cursor at one of the following:
  - The anchor point, if it can be placed there
  - If an element cursor, on the anchor element
  - If there is no anchor element, where the cursor previously was in the scope

[ ]

When using the margin click or swipe techniques:

- Define the selection region to consist of all the elements that correspond to the region of margin elements identified by the technique.
- Place the anchor point at the point in the region nearest to the position in the margin at which the technique was initiated.
- Define the anchor element to be the element in the selection region nearest to the anchor point.
- Define the anchor region to consist of the elements corresponding to the anchor element of the margin's selection region.
- If the margin elements cannot take focus, place the active cursor at one of the following:
  - The pointer position or element farthest from the margin's anchor element, if it can be placed there
  - Where the cursor previously was in the scope

[ ]

When using the margin point, swipe, or click techniques in select mode:

- Select all the elements in the selection region.
- Deselect all other elements in the scope.

[ ]

When using the margin point, swipe, or click techniques in toggle mode, toggle all elements in the selection region, based on the toggling policy.

Maximize (Choice)

[ ]

Provide a Maximize choice in the window menu when a user can change the size of a window.

[ ]

Enlarge the window to the largest size possible for the current view when the user chooses Maximize from the window menu or its associated window icon.

[ ]

Before maximizing a window, save its state (including its size and position) for use when the window is restored.

[ ]

Do not make the Maximize choice available when the window is maximized.

Maximize Button

[ ]

If you provide a Maximize choice, also provide a maximize button that implements the Maximize choice on the title bar when the window is not currently maximized, and the user has not specified otherwise.

[ ]

Replace the maximize button with a restore button that implements the Restore choice when a window is maximized.

Menu (Control)

[ ]

Place only action, cascading, dialog, and value choices in a menu.

Mnemonics

[ ]

Provide the predefined mnemonic for each predefined textual choice in a menu.

[ ]

Provide a unique mnemonic for each application−specific textual choice in a menu, unless there is no meaningful unique mnemonic.

Availability of Choices

[ ]

Except in the Selected menu or in a pop−up menu (or a menu torn off from them), do not add or remove choices from a menu to indicate unavailability of choices. Instead, display unavailable choices with unavailable emphasis.

[ ]

If a choice is not available to a particular user due to an external factor over which the user has no control, do not include or reserve space for the choice in a menu.

[ ]

Display unavailable emphasis on action, value, and dialog choices in a menu that cannot be activated or toggled in the current context.

Size

[ ]

A cascaded menu should be just large enough so that the labels of all the choices in the menu are fully displayed.

Layout

[ ]

Organize menus in either rows or columns.

[ ]

If you provide cascading choices in a menu other than the menu bar, organize the menu in columns.

Navigation

[ ]

Support the use of directional keys for navigation in a menu system when the menu containing the focus is not spring sensitive.

Navigate Down

[ ]

In a menu organized in columns, make `[darr ]` move the cursor to a choice in the row below, wrapping at the bottom row within the menu.

[ ]

In a menu organized in rows, but not on a cascading choice, make [darr] move the cursor to the choice below, wrapping at the bottom edge within the menu.

Navigate Up

[ ]

In a menu organized in columns, make [uarr] move the cursor to a choice in the row above, wrapping at the top row within the menu.

[ ]

In a menu organized in rows, but not on a cascading choice, make [uarr] move the cursor to the choice above, wrapping at the top edge within the menu.

Navigate Left

[ ]

In a menu organized in rows or columns, whose parent is not a menu, make [larr] move the cursor to the choice to the left, wrapping at the left edge within the menu.

[ ]

In a menu organized in rows or columns, but not at the left edge, make [larr] move the cursor to the choice to the left.

[ ]

At the left edge of a menu organized in rows or columns, whose parent menu is organized in columns, make [larr] remove the menu and move the cursor to its associated cascading choice.

[ ]

At the left edge of a menu organized in rows or columns, cascaded from a menu−bar item, make [larr] remove the menu and activate the menu−bar item to the left, first wrapping within the menu bar, if necessary.

Navigate Right

[ ]

In a menu organized in columns, on a cascading choice, make [rarr] activate the choice.

[ ]

In a menu organized in rows or columns, whose parent is not a menu, but not on a cascading choice, make [rarr] move the cursor to the choice to the right, wrapping at the right edge within the menu.

[ ]

In a menu organized in rows or columns, but not at the right edge, and not on a cascading choice, make [rarr] move the cursor to the choice to the right.

[ ]

At the right edge of a menu organized in rows or columns, whose cascaded menu ancestors are all organized in columns, and which is ultimately descended from a menu−bar item, but not on a cascading choice, make [rarr] remove all the cascaded menus and activate the menu−bar item to the right, first wrapping within the menu bar, if necessary.

Visual Guidelines

[ ]

When a cascading choice is used in a menu whose choices are aligned vertically, display a right−pointing arrow to the right of the choice

and align the right edge of the arrow with the right edge of the menu.

Focus and Default Choices

[ ]

If a spring−loaded menu is displayed and is not spring sensitive, give it focus unless the focus is in a descendant spring−loaded menu.

[ ]

When focus moves to a menu that has just been displayed, place the active cursor on the default choice in the menu. If not determined in any other specified way, make the default choice the first choice in the menu or, if that is a tear−off choice, the following choice.

Placement

[ ]

When a submenu is displayed from a menu−bar item, place it, if possible, immediately below the menu−bar item, with the left edges of the menu and menu−bar item aligned.

[ ]

When a submenu is displayed from a menu organized as a single column, place the submenu, if possible, so that its left edge is aligned with the right edge of the parent menu.

Keyboard Activation and Toggling

[ ]

When the active cursor is on an available choice in a menu and the user presses `Select`, `Space`, `Ctrl Space`, `Enter`, `CtrlEnter`, or `keypadEnter`, activate or toggle the choice.

Menu Bar (Menu Type)

[ ]

Provide a menu bar if you provide any of the action choices included in the File menu, other than Close or Exit.

[ ]

Provide a menu bar if you provide any of the action choices included in the View menu.

[ ]

Place only cascading choices in a menu bar.

[ ]

A cascading choice in a menu bar leads to a pull−down menu.

Menu−Bar Content

[ ]

In conjunction with the Menu Guidelines reference page, use Menu−Bar Choices to decide which choices to include in a menu bar and how to organize them.

Table 1 Menu−Bar Choices

Mnemonic	Item	Keyboard Function
F	File	AltF
E	Edit	AltE
V	View	AltV
W	Window	AltW
H	Help	AltH

Directional Navigation

[ ]

In a menu bar, make [larr] move the cursor to the choice to the left, wrapping at the left edge within the menu bar.

[ ]

In a menu bar, make [rarr] move the cursor to the choice to the right, wrapping at the right edge within the menu bar.

[ ]

In a menu bar, make [darr] activate the cursored cascading choice, display the associated pull−down menu, and move focus to it.

Navigation to and from the Menu Bar

[ ]

When focus is in a control in a window and the user presses F10 or Shift Menu, navigate to the first item in the menu bar.

[ ]

If a window has a menu bar, but focus is in a different spring−loaded system originating from the client area of that window, then make pressing F10 or Shift Menu deactivate the spring−loaded system before navigating to the first item in the menu bar.

[ ]

If the focus is in a menu−bar system, make pressing F10 or Shift Menu deactivate the system.

Placement and Layout

[ ]

Place the menu bar at the top of the client area of a window and extend it the width of the entire window.

[ ]

Align menu−bar items horizontally and lay them out left to right. For information on bidirectional and vertical language support, see Chapter 11.

Visual Guidelines

[ ]

Do not display a graphic (in particular, a down−arrow graphic) along with a cascading choice in a menu−bar item.

Menu Cascade Button (Control)

[ ]

Do not use menu cascade buttons at the top of a window in place of a menu bar.

[ ]

Do not use menu cascade buttons to provide choices that apply to the window or the controls in a window as a whole.

Keyboard Navigation

[ ]

When focus is on a menu cascade button and the user presses `Select`, `spacebar`, or `Ctrl spacebar`, display the associated pull−down menu and move focus to it.

Visual Guidelines

[ ]

Except in a window menu button, display a down−arrow graphic to the right of the cascading choice in a menu cascade button. For information on bidirectional and vertical language support, see Chapter 11.

Menu Guidelines

Inclusion

[ ]

If a choice for a described menu is marked [ ], then your menu should include that choice if that functionality can be reasonably provided by your application.

Usage

[ ]

If your menu includes a choice whose functionality is the same as any choice listed for that menu, use the predefined label specified for the choice.

[ ]

If your menu includes two or more of the choices listed for that menu, they should appear in the order listed.

[ ]

The choice in the described menus are separated into groups. If your menu includes items listed in different groups, use a separator to separate the items.

[ ]

If a choice for a described menu has a mnemonic or shortcut key specified, then if the choice appears in your menu, use that mnemonic or shortcut key.

[ ]

If a choice for a described menu has a mnemonic specified, but that choice does not appear in your menu, do not use the mnemonic for other choices in the menu.

Message

Message Text

[ ]

Phrase message text so that a user clearly understands what caused the message as well as what action, if any, can be taken to correct the situation that caused the message.

[ ]

When wording the messages displayed by your application, do not assume that the user has any expert knowledge about computer systems in general or the operating system in particular. You can assume that the user has knowledge about basic interface terms that could have been learned through tutorials, help information, and user documentation. However, avoid terminology that is typically understood only by an expert or knowledgeable computer user unless the application is specifically targeted at computer professionals. Also, messages returned to your application by the underlying system should not be passed directly to the user, but instead should be interpreted into language the novice user can understand.

Message Windows

[ ]

If a window associated with the task or element for which the task is being performed is open, display the message in a secondary window that depends on the associated window.

[ ]

If no associated window is open for a task being performed and for which an urgent message must be displayed, then display the message in a secondary window and, optionally, display the task's window if such a window exists.

[ ]

If no associated window is open for the task being performed and for which a nonurgent message needs to be displayed, then place the message in a secondary window, but do not display it. Optionally, change the appearance of the icon through which the window can be displayed if such a window exists.

Minimize (Choice)

[ ]

Provide the Minimize choice in the window menu of a primary window.

[ ]

Do not provide the Minimize choice for secondary windows.

[ ]

Remove the window from the display and display its window icon, if not already displayed, when the user chooses Minimize from the window menu or the associated window icon.

[ ]

Before minimizing a window, save its state, including its size and position, along with the state of each of its secondary windows to be used when the window is restored.

[ ]

Do not make the Minimize choice available when the window is minimized.

[ ]

Even if a window has been minimized, design the application to continue to perform any application tasks initiated through that window.

Minimize Button

[ ]

If you provide a Minimize choice and the user has not specified otherwise, also provide a minimize button that implements the Minimize choice on the title bar.

Mnemonic

[ ]

Provide a predefined mnemonic for each predefined textual choice in a menu.

[ ]

Provide a mnemonic for each textual choice in a menu, unless you cannot find a meaningful unique mnemonic.

Uniqueness

[ ]

Make mnemonics unique among all choices in a menu.

[ ]

Make mnemonics unique among all radio buttons and check boxes in a tab group.

[ ]

Make mnemonics unique among all push buttons, group headings, and menu–bar items.

[ ]

If a mnemonic is an alphabetic character, allow the user to press either the uppercase or lowercase character.

Assignment and Display of Mnemonics

[ ]

The character used as a label's mnemonic must appear in the label and be highlighted within it, generally by underlining it.

Mnemonic Behavior

[ ]

When the active cursor is in a menu and the user types the mnemonic of a choice within that menu, move the cursor to the choice and activate or toggle it.

`Alt` Mnemonic Behavior

[ ]

When the user presses `Alt` plus a mnemonic associated with a menu–bar item, activate that menu–bar item and move focus to the menu pulled down from it.

[ ]

When the user presses `Alt` plus a mnemonic associated with an option menu or menu cascade button, pull down the corresponding menu and move focus to it.

Mouse (Device)

[ ]

Provide support for a mouse as a pointing device.

[ ]

Support the following virtual mouse buttons:

SELECT

The virtual pointer button used for selection, activation, toggling, and forms of direct manipulation not assigned to other virtual mouse buttons.

ADJUST

The virtual pointer button used for adjusting a selection.

TRANSFER

The virtual pointer button used for data transfer operations.

MENU

The virtual pointer button used to obtain pop−up menus.

[ ]

The period of time between button release and the subsequent press in a multi−click event is user adjustable, and the default is not less than 100 milliseconds.

[ ]

Do not assign double−click functions to choices on which the user will typically perform multiple single clicks, such as on push buttons.

Three−Button Mouse Bindings

[ ]

Support a 3−button mouse.

[ ]

Treat a 2−button mouse as a 3−button mouse if it has been configured so that the third button is available by chording the other two buttons (for example, with MB1+MB2 treated as MB3).

[ ]

On a 3−button mouse, bind MENU to MB3.

[ ]

Allow the user to choose whether the bindings on a 3−button mouse support one of the following:
SELECT and TRANSFER integrated on MB1 and TRANSFER bound to MB2 (recommended)
SELECT bound to MB1 and TRANSFER bound to MB2
SELECT and TRANSFER integrated on MB1 and ADJUST bound to MB2
SELECT and TRANSFER integrated on MB1 and nothing bound to MB2

[ ]

If you support binding TRANSFER to MB2 and also integrate TRANSFER with SELECT on MB1, then whenever MB1 is pressed and moved, interpret the action as a drag−and−drop operation. Ensure that when MB2 is pressed and the pointer is moved along the same path,

it will have an identical effect.

[ ]

If you support binding ADJUST to MB2, design your application so that using ADJUST is identical to using `Shift` SELECT.

Two−Button Mouse Bindings

[ ]

Support a 2−button mouse.

[ ]

Allow the user to choose whether the bindings on a 2−button mouse support one of the following:

SELECT and TRANSFER integrated on MB1 and MENU on MB2

SELECT and TRANSFER integrated on MB1, TRANSFER bound to MB2, and MENU bound to `Alt` MB1

SELECT bound to MB1, TRANSFER bound to MB2, and MENU bound to `Alt` MB1

Move To (Dialog Choice)

[ ]

When appropriate, the File Selection dialog should allow the user to differentiate between moving the link or reference to a file or object, moving its underlying storage location, or both.

Multilevel Selection Technique

[ ]

After using a multilevel point technique:
- Identify the pointer position as the anchor point.
- Define the selection region to consist of the part of elements (depending on the number of clicks), if any, containing the pointer.
- Define the anchor element to be the element in the region nearest to the pointer.
- Define the anchor region to be the selection region.
- Place the active cursor as follows:
  - At the anchor point, if it can be placed there
  - On the anchor element, if it is an element cursor
  - Where the cursor previously was in the scope, if there is no anchor element

[ ]

When using a multilevel click or swipe technique:
- Identify the pointer position at initiation of the technique as the anchor point.
- Define the selection region to consist of all elements that would be selected when using the corresponding click or swipe technique.
- Expand the selection region if all elements in its part (depending on the number of presses) are also included in the region.
- Define the anchor element to be the element in the selection region nearest to the pointer.
- Define the anchor region to be the selection region that would be obtained if a multilevel point technique had been used instead at the point at which the technique was initiated.
- Place the active cursor as follows:
  - At the pointer position when the technique was finished, if it can be placed there
  - On the element at the other end of the range from the anchor element, if you are using an element cursor, the region is not empty, and range technique is in effect
  - On an element within the selection region, if you are using an element cursor, the region is not empty, and area technique is in effect
  - On the last element touched, if you are using an element cursor, the region is not empty, and touch technique is in effect
  - Where the cursor previously was in the scope, if none of the above are true

[ ]

When using a multilevel point, swipe, or click technique in select mode:
- Select all the elements in the selection region.
- Deselect all other elements in the scope.

[ ]

When using a multilevel point, swipe, or click technique in toggle mode, toggle all elements in the selection region, based on the toggling policy.

[ ]

When using an adjustment technique to adjust a selection initially made using a multilevel technique, expand the selection region so that if an element is in the region, all elements in its part (depending on the number of times the SELECT button was pressed in the initial technique) are also included in the region.

[ ]

When using a multilevel adjust click or swipe technique:
- Identify the pointer position at initiation of the technique as the anchor point.
- Define the selection region to consist of all elements that would be selected when using the corresponding adjust click or swipe technique, but then expanded so that one of the following occurs:
  - If an element is added to the region, also include all elements in its part (depending on the number of times the ADJUST button is pressed) in the region.
  - If an element is removed from the region, place it back into the region, unless all elements in its part (depending on the number of times the ADJUST button was pressed) are also included in the region.
- Define the anchor element to be the element in the selection region nearest to the pointer.

Define the anchor region to be the selection region that would be obtained if a multilevel point technique had been used instead at the point at which the technique was initiated.

Multipage Control (Control Type)

Tab Navigation

[ ]

Organize the controls within each page of a multipage control into one or more tab groups.

[ ]

Do not use tab group navigation to switch pages of a multipage control.

[ ]

When forward tab group navigation is in use, traverse all tab groups on a visible page before navigating off the page.

[ ]

When tab group navigation moves focus into a multipage control, place focus, initially, in one of the visible pages.

Navigation (CDE)

Mouse−Based Navigation

[ ]

When the keyboard focus policy is explicit, allow the user to press the SELECT button on a component to move focus to it, except for components, such as scroll bars, that are used to adjust the size and location of other elements.

[ ]

When the pointer is on a menu, allow the user to press the SELECT button to activate the menu in a spring−loaded manner.

[ ]

When the pointer is on an element with an inactive pop−up menu and the context of the element allows the pop−up menu to be displayed, allow the user to press the MENU button to activate the pop−up menu in a spring−loaded manner.

[ ]

If the user takes an action to display a pop−up menu, and a menu can be displayed for both an inner element and an outer element that contains the inner element, display the pop−up menu for the internal element. Be sure that the pop−up menu for an internal element is always accessible.

[ ]

Once a pop−up menu is displayed, make the MENU button behave just as the SELECT button does for any menu system.

[ ]

Make the SELECT button available from within posted pop−up menus. It should behave just as in any menu system.

[ ]

When a menu is popped up or pulled down, place the location cursor on the menu's default entry or on the first entry in the menu, if there is no default entry.

[ ]

Remove a spring−loaded menu system when the mouse button that activated it is released, except when the button is released on a cascading button in the menu hierarchy.

[ ]

While a spring−loaded menu is popped up or pulled down, moving the pointer within the menu system moves the location cursor to track the pointer.

[ ]

When a spring−loaded menu is popped up or pulled down and the pointer rests on a cascading button, pull down the associated menu and make it traverse. Remove the associated menu, possibly after a short delay, when the pointer moves to a menu item outside of the menu or its cascading button.

[ ]

When a spring−loaded menu that is part of the menu bar is pulled down, moving the pointer to any other element on the menu bar unposts the current menu system and posts the pull−down menu associated with the new element.

[ ]

When a spring−loaded menu is popped up or pulled down, and the button that activated the menu system is released within a component in the menu system, activate that component. If the release is on a cascading button or an option button, activate the associated menu in a posted manner if it was not posted prior to the associated button press.

[ ]

When the pointer is in an area with a pop−up menu, allow the user to click the MENU button to activate the menu in a posted manner if it was not posted prior to clicking the MENU button.

[ ]

Once a pull−down or option menu is displayed, pressing the SELECT button in the menu system causes the menu to behave as a spring −loaded menu. This allows the user to switch easily between using a posted menu and a spring−loaded menu.

[ ]

If a button press unposts a menu and that button press is not also passed to the underlying component, do not pass subsequent events up to and including the button release to the underlying component.

When a button press unposts a menu, the press can be passed to the underlying component. Whether or not it is passed to the underlying component, the press can have additional effects, such as raising and giving focus to the underlying window. If the press is not passed to the underlying component, events up to and including the release must not be passed to that component.

[ ]

Once a pop−up menu is displayed, pressing the SELECT button or pressing the MENU button in the menu system causes the menu to behave as a spring−loaded menu.

[ ]

Pressing or clicking the MENU button on an option button displays the option menu.

[ ]

Pressing the SELECT button on a text field causes the text cursor to be inserted at the mouse cursor position.

Keyboard−Based Navigation

[ ]

In a text component, display the text cursor differently when it has keyboard focus than when it does not. In a text component, the text cursor serves as the location cursor and, therefore, must indicate whether the component has keyboard focus.

[ ]

If a text component indicates that it has lost the keyboard focus by hiding the text cursor and if the component subsequently regains the focus, the cursor should reappear at the same position it had when the component lost focus. The text cursor should not change position when a text component loses and then gains keyboard focus.

[ ]

In a small component, such as a sash, indicate that it has keyboard focus by filling it. No other meaning should be associated with the filled state.

[ ]

Pressing `Ctrl Tab` moves the location cursor to the next field, and pressing `Ctrl Shift Tab` moves the location cursor to the previous field. Unless `Tab` and `Shift Tab` are used for internal navigation within a field, pressing `Tab` also moves the location cursor to the next field, and pressing `Shift Tab` also moves the location cursor to the previous field.

[ ]

Pressing `Tab` (if not used for internal navigation) or `Ctrl Tab` moves the location cursor forward through fields in a window according to the following rules:

If the next field is a control, pressing `Tab` (if not used for internal navigation) or `Ctrl Tab` moves the location cursor to that control.

If the next field is a group, pressing `Tab` (if not used for internal navigation) or `Ctrl Tab` moves the location cursor to a traversable component within the group.

If the next field contains no traversable components, pressing Tab (if not used for internal navigation) or Ctrl Tab skips the field.

[ ]

Pressing Shift Tab (if not used for internal navigation) or Ctrl Shift Tab moves the location cursor backward through fields in the order opposite to that of Tab (if not used for internal navigation) and Ctrl Tab.

[ ]

When a window acquires focus, place the location cursor on the control that last had focus in the window, providing that all the following conditions are met:

The window uses an explicit keyboard focus policy.

The window acquires the focus through keyboard navigation or through a button press other than within the client area of the window.

The window had the focus at some time in the past.

The control that last had focus in the window is still traversable.

[ ]

Wrap field navigation between the first and last fields in the window.

[ ]

When [darr] and [uarr] are used for component navigation within a field, they should behave according to the following rules:

In a left−to−right language environment, pressing [darr] moves the location cursor through all traversable controls in the field, starting at the upper left and ending at the lower right, then wrapping to the upper left. If the controls are aligned in a matrix−like arrangement, [darr] first traverses one column from top to bottom, then traverses the column to its right, and so on. In a right −to−left language environment, pressing [darr] moves the location cursor through all traversable controls, starting at the upper right and ending at the lower left.

Pressing [uarr] moves the location cursor through all traversable controls in the field in the order opposite to that of [darr].

[ ]

When [rarr] and [larr] are used for component navigation within a field, they should behave according to the following rules:

In a left−to−right language environment, pressing [rarr] moves the location cursor through all traversable controls in the field, starting at the upper left and ending at the lower right, then wrapping to the upper left. If the controls are aligned in a matrix−like arrangement, [rarr] first traverses one row from left to right, then traverses the row below it, and so on. In a right−to−left language environment, pressing [rarr] moves the location cursor through all traversable controls, starting at the lower left and ending at the upper right.

Pressing [larr] moves the location cursor through all traversable controls in the field in the order opposite to that of [rarr].

[ ]

If a control uses [rarr] and [larr] for internal navigation, pressing Begin moves the location cursor to the leftmost edge of the data or the leftmost element in a left−to−right language environment. In a right−to−left language environment, pressing Begin moves the location cursor to the rightmost edge of the data or the rightmost element.

[ ]

If a control uses [rarr] and [larr] for internal navigation, pressing End moves the location cursor to the rightmost edge of the data or the rightmost element in a left−to−right language environment. In a right−to−left language environment, pressing End moves the location cursor to the leftmost edge of the data or the leftmost element.

[ ]

If a control uses [darr] and [uarr] for internal navigation, pressing Ctrl Begin moves the location cursor to one of the following:

The first element

The topmost edge of the data

In a left−to−right language environment, the topmost left edge of the data; in a right−to−left language environment, the topmost right

edge of the data

[ ]

If a control uses [darr] and [uarr] for internal navigation, pressing Ctrl End moves the location cursor to one of the following:

The last element

The bottommost edge of the data

In a left−to−right language environment, the bottommost right edge of the data; in a right−to−left language environment, the bottommost left edge of the data

[ ]

Pressing Tab key moves input focus between push buttons within a group.

[ ]

The Ctrl, Shift, and Alt keys should modify only the function of other keys or key combinations.

Menu Traversal

[ ]

If the user traverses to a menu while the keyboard focus policy is implicit, temporarily change the focus policy to explicit and reverts to implicit whenever the user traverses out of the menu system.

Menus must always be traversable, even when the keyboard focus policy is generally implicit.

[ ]

Pressing F10 activates the menu bar−system if it is inactive and the location cursor is placed on the first traversable cascading button in the menu bar. If there are no traversable cascading buttons, the key should do nothing.

[ ]

When the keyboard focus is in an element with an inactive pop−up menu and the context of the element allows the pop−up menu to be displayed, pressing menu activates the pop−up menu and places the location cursor on the default item of the menu or on the first traversable item in the pop−up menu, if there is no default item.

[ ]

When the keyboard focus is in an option button, pressing Select or Spacebar displays the option menu and places location cursor on the previously selected item in the option menu; or, if the option menu has been pulled down for the first time, places the location cursor on the default item in the menu. If there is an active option menu, pressing Return, Select, or Spacebar selects the current item in the option menu, unposts the menu system, and returns the location cursor to the option button.

[ ]

Pressing [darr], [uarr], [larr], and [rarr] traverses through the items in a menu system.

[ ]

When a menu traversal action traverses to the next or previous component in a menu or menu bar, the order of traversal and the wrapping behavior should be the same as that of the corresponding component navigation action within a field.

[ ]

If your application uses any two−dimensional menus, do not include any cascading buttons. Cascading buttons in a two−dimensional menu restricts the user's ability to use the keyboard to navigate to all of the elements of the menu.

[ ]

When focus is on a component in a menu or menu−bar system, pressing [darr] behaves in the following way:

If the component is in a vertical or two−dimensional menu, pressing [darr] traverses down to the next traversable component,

wrapping the component within the menu if necessary.
If the component is in a menu bar, and the component with the keyboard focus is a cascading button, pressing [darr] displays its associated pull–down menu and traverses to the default entry in the menu or, if the menu has no default, to the first traversable entry in the menu.

[ ]

When focus is on a component in a menu or menu–bar system, pressing [uarr] behaves in the following way:
If the component is in a vertical or two–dimensional menu, pressing [uarr] traverses up to the previous traversable component, wrapping within the menu if necessary, and proceeds in the order opposite to that of pressing [darr].

[ ]

When focus is on a component in a menu or menu–bar system, pressing [larr] behaves in the following way:
If the component is in a menu bar or two–dimensional menu, but not at the left edge, pressing [larr] traverses left to the previous traversable component.
If the component is at the left edge of a menu bar, pressing [larr] wraps within the menu bar.
If the component is at the left edge of a vertical or two–dimensional menu that is the child of a vertical or two–dimensional menu, pressing [larr] unposts the current menu and traverses to the parent cascading button.
If the component is at the left edge of a vertical or two–dimensional menu that is the child of a menu bar, pressing [larr] unposts the current menu and traverses left to the previous traversable entry in the menu bar. If that entry is a cascading button, pressing [larr] posts its associated pull–down menu and traverses to the default entry in the menu or, if the menu has no default, to the first traversable entry in the menu.

[ ]

When focus is on a component in a menu or menu–bar system, pressing [rarr] behaves in the following way:
If the component is a cascading button in a vertical menu, pressing [rarr] displays its associated pull–down menu and traverses to the default entry in the menu or, if the menu has no default, to the first traversable entry in the menu.
If the component is in a menu bar or two–dimensional menu, but not at the right edge, pressing [rarr] traverses right to the next traversable component.
If the component is at the right edge of a menu bar, pressing [rarr] wraps within the menu bar.
If the component is not a cascading button and is at the right edge of a vertical or two–dimensional menu, and if the current menu has an ancestor cascading button (typically in a menu bar) from which pressing [darr] displays its associated pull–down menu, pressing [rarr] unposts the menu system pulled down from the nearest such ancestor cascading button and traverses right from that cascading button to the next traversable component. If that component is a cascading button, pressing [rarr] displays its associated pull–down menu and traverses to the default entry in the menu or, if the menu has no default, to the first traversable entry in the menu.

[ ]

Allow all menu traversal actions, with the exception of menu posting, to traverse to tear–off buttons in the same way as for other menu entries. Traversal of tear–off buttons needs to be consistent with traversal of other menu items.

[ ]

If your application uses F10, Menu, or Cancel to unpost an entire menu system and an explicit focus policy is in use, move the location cursor back to the component that had it before the menu system was posted.

Scrollable Component Navigation

[ ]

Any scrollable components within your application should support the appropriate navigation and scrolling operations. The page navigation keys Page Up, Page Down, Control Page Up (for Page Left), and Control Page Down (for Page Right) scroll the visible region by a page increment. The user needs to be able to view and access the entire contents of a scrollable component.

[ ]

When scrolling by a page, leave at least one unit of overlap between the old and new pages.

[ ]

Any keyboard operation that moves the cursor to or in the component, or that inserts, deletes, or modifies items at the cursor location should scroll the component so that the cursor is visible when the operation is complete. The user needs to be able to see the results of moving the location cursor or the effects of operating on the contents of the scrollable component.

[ ]

If a mouse−based scrolling action is in progress, pressing `Cancel` cancels the scrolling action and returns the scrolling device to its state prior to the start of the scrolling operation.

New (Action Choice)

File Menu

[ ]

The New choice in the File menu either replaces the contents of the current window or opens a new window (which is recommended).

[ ]

When the user chooses New from the File menu, and the contents of the window are to be replaced, prompt the user with a message to save the window contents if possible.

Selected Menu

[ ]

The New choice in the Selected menu creates a new object of the single type managed and displayed by the application, and includes it in the selection scope displayed within the window.

[ ]

When the user chooses New from the Selected menu, give the newly created object a default name that does not conflict with the name of an existing object and allow the user to change the name.

New Window (Choice)

[ ]

Provide a New Window choice in the View menu if your application allows a user to display a new window that contains another view of the data viewed in the current window.

[ ]

When New Window is an action choice, choosing it displays a new window that contains the same view of the data as the current window, or a default view.

[ ]

When New Window is a cascading choice, its cascaded menu must list the views available to be shown in the new window.

Multiple Windows for Viewing Data

[ ]

Do not change the underlying data being viewed when the user opens a window that contains a new view.

Notebook (Control)

Notebook Layout

[ ]

Provide a spin box paging control for paging through the notebook.

[ ]

If there is insufficient space to display a group of notebook tabs, the layout is clipped and arrow buttons (action choices that cannot take focus) are displayed on the ends of the group to scroll the visible set of tabs.

Behavior of Notebook Tabs

[ ]

The action associated with a notebook tab displays its corresponding page.

[ ]

Allow directional keys to navigate to all the notebook tabs in a group, including those that are not currently visible (navigating to one makes it visible).

[ ]

When the user uses the keyboard to activate a notebook tab, leave focus in the notebook tab.

Tab Groups

[ ]

Organize the major tabs in a notebook into a single tab group.

[ ]

Organize the minor tabs corresponding to pages within a section associated with a major tab into a single tab group.

[ ]

Organize any paging controls (other than notebook tabs) into one or more sets of tab groups.

[ ]

Organize any bookmark tabs into one or more sets of tab groups.

Tab Navigation Within Single−Page Notebooks

[ ]

Make the minor tabs for a section of a single−page notebook visible only when the currently visible page is within that section.

[ ]

When tab group navigation is used to traverse to a group of major or minor tabs in a single−page notebook, place focus on one of the following:

- The notebook tab for the currently visible page
- The nearest notebook tab preceding the currently visible page
- The first notebook tab in the group

[ ]

Traverse the tab groups within a single−page notebook in the following order:

- Tab group(s) of the visible page
- Any nontab paging controls
- Bookmark tabs
- Major tabs
- Minor tabs
- Traversable status controls

Object

[ ]

In the pop–up menu associated with an object, provide access to the actions the user will need to perform with the task.

[ ]

Make the default action of an object be the action the user is most likely to want to perform with the task.

[ ]

Allow a user to change the default action of an object.

[ ]

Make a default action open a window that displays a view of the object (showing its contents), present an accessible user interface, or both. This is not required if the object is best presented in a nonvisual medium (for example, it contains audio and its default action is Play).

[ ]

Allow an object to be named or otherwise identified uniquely within a scope that contains multiple objects.

[ ]

If the default action opens a view on an object, and the view is already open, use properties that the user can set to determine whether to do one of the following:

- Surface the view (recommended by default)
- Open a new view on the object

[ ]

When the user opens a view of an object, do not remove the icon from which the view was opened.

Open (Choice)

File Menu

[ ]

When the user selects the Open dialog choice from the File menu, either open a new window or replace the contents of the current window.

[ ]

When the user chooses Open from the File menu, and the contents of the window are to be replaced, prompt the user with a message to save the window contents if possible.

Selected Menu

[ ]

When the user chooses Open from the Selected menu, open each selected object in a separate window.

Option Menu (Menu Type)

[ ]

Use an option menu to let the user choose a value from a fixed group of possible values.

[ ]

Place only cascading and value choices in an option menu.

[ ]

A cascading choice in an option menu leads to another option menu.

[ ]

All of the value choices in an option menu and its cascaded descendants are mutually exclusive; the user can choose only one at a time.

Keyboard Navigation

[ ]

When focus is on an option menu button and the user presses `Select`, `Space`, or `Ctrl Space`, display the associated option menu and move focus to it.

Default Choice

[ ]

If a value choice in an option menu corresponds to the value shown in its base option menu button, make it the default choice for that menu.

Size

[ ]

Make the width of the static text area of an option menu button the same as the width of the option menu displayed from it.

Placement

[ ]

When the user chooses an option menu button to display an option menu, chooses an option menu button to the option menu must not cover the graphic that indicates a cascading choice in the option menu button.

Visual Guidelines

[ ]

Display the cascading choice in an option menu button as a horizontal bar graphic.

[ ]

Display the last value chosen in the option menu or its cascaded descendants in the option menu button as static text to the left of the bar graphic. For information on bidirectional and vertical language support, see Chapter 11.

[ ]

Do not place a graphic next to a value choice in an option menu (for example, to indicate that it is the value that was last chosen).

Options (Menu)

[ ]

Provide choices in the Options menu that are specific to the application overall, not just to a particular view.

[ ]

Changes resulting from choices made in the Options menu do not change the underlying data being viewed.

[ ]

Do not use choices in the Options menu to set properties of objects or elements; use a Properties choice instead.

Palette Area (Area)

Layout

[ ]

Lay out the palette area as a control bar (for example, place it just below the window's menu bar) if it can apply to all viewing areas in the window.

[ ]

If a palette area applies to only one viewing area in a window with multiple viewing areas, place the palette area adjacent to the associated viewing area.

[ ]

If a palette area is placed adjacent to its associated viewing area, place it on only one side of the viewing area. Use two sides only when there is a large number of critical action choices that must be placed in a logical relationship to an associated fixed element in the view.

Customizing the Palette Area

[ ]

If the user can customize a palette area or remove the action choices in a palette area, use a menu to provide access to the removable choices.

Paned Box (Control)

[ ]

Support paned boxes that can be split either horizontally or vertically.

[ ]

Provide a split bar between panes of a paned box.

[ ]

If the paned box is used to separate multiple viewing areas in a window, make the window resizable.

Navigation

[ ]

Design each pane in a paned box to consist of one or more tab groups.

[ ]

Make each sash in a paned box a tab group.

[ ]

When the user uses tab−group navigation, traverse through the tab groups in a paned box in the following order:

- All tab groups in the first pane
- The sash on the split bar of the first pane, if any
- All tab groups in the next pane
- The sash on the next split bar

Repeat this pattern to the end of the paned box.

Persistent Cue

[ ]

Provide a persistent cue to distinguish the state of some object represented by an element in the user interface or the status of some task that the user requested.

[ ]

Provide a persistent cue to alert the user about a potentially serious condition in the operating environment.

[ ]

Provide a persistent cue to attract the user's attention to some area of the interface.

[ ]

If the persistent cue is intended to alert the user, supplement it with a warning signal.

[ ]

If a persistent cue represents an internal state that is changing, update the cue as the change happens. If the change is a continuous change, update the cue continuously or at regular intervals.

[ ]

If a persistent cue blinks (for example, it turns on and off repeatedly) use it only if the element it is on has input focus or if the user must be made aware of a critical condition.

[ ]

Do not design a new persistent cue to represent a state that already has one defined for it.

[ ]

If a persistent cue is associated with a task, display the cue while the task is in progress and remove it when the task is done.

Point Technique

Mouse−Based Select Mode

[ ]

When the user is using the point technique in select mode and presses or clicks the SELECT button on a selectable element and then releases SELECT on the same element:

Select that element.

Deselect all other elements in the scope.

[ ]

When using the point technique in select mode and the user clicks the SELECT button in the background of a selection scope, deselect all elements in the scope.

Mouse−Based Toggle Mode

[ ]

When the user is using the point technique in select mode and presses `Ctrl` SELECT in place of SELECT, select as if toggle mode were in use.

[ ]

When the user is using the point technique in toggle mode and presses or clicks the SELECT button on a selectable element and then releases SELECT on the same element, toggle the selection state of that element.

[ ]

Allow the point technique in toggle mode to select an element as follows:

If only one element at a time may be selected in a scope, deselect the currently selected element (if any) in the scope.

If any number of elements can be selected in the scope, do not change the selection state of the other elements.

[ ]

When the user is using the point technique in toggle mode and clicks the SELECT button in the background of a selection scope, it should have no effect on the selection state of the elements in the scope.

Keyboard−Based Normal Mode

[ ]

When the user is using the point technique in normal mode while the active cursor is on a selectable element and presses `Select`, `Space` (unless in text), or `Ctrl Space`:

Select that element.

Deselect all other elements in the scope.

[ ]

When the user is using the point technique in normal mode, if the active cursor is in the background of a selection scope and the user presses `Select`, `Space` (unless in text), or `Ctrl Space`, deselect all elements in the scope.

Keyboard−Based Add Mode

[ ]

When the user is using the point technique in add mode while the active cursor is on a selectable element and presses `Select`, `Space` (unless in text), or `Ctrl Space`, toggle the selection state of that element.

[ ]

Allow the point technique in add mode to select an element as follows:

- If only one element at a time may be selected in a scope, deselect the currently selected element (if any) in the scope.
- If selections are required to be contiguous, deselect all other elements in the scope.
- If any number of elements can be selected in the scope, do not change the selection state of the other elements.

[ ]

When the user is using the point technique in add mode while the active cursor is in the background of a selection scope and presses `Select`, `Space` (unless in text), or `Ctrl Space`:

- If selections are required to be contiguous, deselect all elements in the scope.
- If selections may be discontiguous, do not change the selection state of the elements in the scope.

Selection Region

[ ]

When the user performs the point technique on a selectable element:

- Identify that element as an anchor element for later use in adjustment of the selection.
- Define the current selection region to consist of that element.

[ ]

When the user uses the point technique on a selectable element, identify a point as an anchor point for later use in adjustment of the selection. That point should be:

- The pointer position, for the mouse−based technique
- The cursor position, for the keyboard−based technique with a text or graphics cursor
- The center of the element, for the keyboard−based technique with an element cursor

[ ]

When the user uses the point technique in the background of a selection scope:

- Identify that point as an anchor point for later use in adjustment of the selection.
- Define the current selection region to be empty.

[ ]

After a mouse−based point technique, place the active cursor as follows:

- At the anchor point, if it can be placed there
- On the anchor element, if using an element cursor
- Where the cursor previously was in the scope, if there is no anchor element

Pointer

[ ]

Display a pointer whenever the operating environment allows the user to move it.

[ ]

Use the hot spot to target (point to) the window, object, element, or selection under the pointer. If multiple elements are under the pointer, the topmost element is generally considered the element pointed to by the pointer.

[ ]

When no other element−specific or task−specific pointer is displayed, display the arrow pointer.

[ ]

Do not create new pointer shapes for operations that already have pointer shapes associated with them.

[ ]

If the pointer can be warped (that is, moved by the application without reference to any user control of the pointing device), allow the operating environment to disable this feature and prevent the pointer from being warped.

Definition of the Hot Spot

[ ]

Define a hot spot for each pointer shape.

Using the Keyboard to Move the Pointer

[ ]

For users unable to use a standard pointing device, support a mode whose only effect is to enable the directional keys, possibly modified with `Ctrl`, to act as a pointing device.

[ ]

Support access for users unable to press a pointer button or move the pointer simultaneously or for users who need to perform more precise placement than possible with a standard pointing device.

[ ]

If your interface enters a mode on a pointer button press and supports behavior on pointer motion in that mode, and there is no other efficient alternative means of providing the same effect, then do the following:

Provide a means of entering that mode via the keyboard, clicking a pointer button (possibly augmented by modifier keys), or both.

Support both the pointing device and the directional keys as a means of moving the pointer in that mode, with an unmodified directional key moving the pointer by one pixel in the direction indicated, and a directional key modified by `Ctrl` moving the pointer by a larger increment.

Do not change the effect of other keyboard actions. In particular, support `Enter` to complete the operation and to exit from the mode, `Cancel` to cancel the operation and to exit from the mode, and `Help` to obtain help.

Behavior

[ ]

Use the hot spot position of the pointer to track the movement of the input device and the location where input device actions occur.

[ ]

As the pointer changes shape, indicating a change in the function of the current area, do not change the location of the hot spot on the

screen.

Pointer (Predefined)

[ ]

Display the arrow pointer to indicate the normal or predominant mode of operation of the user interface.

[ ]

Display the cannot pointer to indicate that the element under the pointer is not a valid target for a data transfer.

[ ]

Display the cannot pointer to indicate that action is expected in another visible area before input is accepted in the area under the pointer.

[ ]

Display a wait pointer to indicate that the user cannot currently interact with the element the pointer is over because some action is in progress in that area.

[ ]

Display an arrow pointer or an I−beam pointer when the pointer is in a text field.

[ ]

Display a question pointer when the interface is in context−sensitive help mode.

[ ]

Display the right arrow pointer when a spring−loaded control is displayed and the interface is waiting for a choice to be made.

[ ]

Ignore all mouse button and keyboard actions (except changes in modifier keys) while the cannot pointer or the wait pointer is visible.

[ ]

Use the predefined pointers when a user performs an operation that uses predefined modal interactions or operations.

Pop–Up Menu (Menu Type)

[ ]

Associate a pop–up menu with every object.

Inclusion

[ ]

If a choice is not currently available, do not include it in a pop–up menu (except if a window menu is used as a pop–up menu).

[ ]

If an action choice applies only to objects (for example, choices that also appear on the Selected menu), make the choice available on a pop–up menu.

[ ]

If a choice is not currently available, do not include it in a pop–up menu (except for those used as window menus).

Display

[ ]

Treat a menu popped up on an unselected character in text as if it were popped up in the background.

[ ]

If the user attempts to display a pop–up menu where no menu exists, or in a context in which the menu would be empty, do not display any pop–up menu; initiate a warning signal instead.

[ ]

Keep the relative order of identical choices the same on a pop–up menu as on the pull–down menu. For example, because Paste appears after Copy on the Edit menu, it should appear after Copy on any corresponding pop–up menu.

[ ]

Do not display a shortcut key for a choice on a pop–up menu unless it has the same result as the corresponding choice from a pull–down menu.

[ ]

If you include the Help choice on a pop–up menu, make it the last choice on the menu.

[ ]

On every pop–up menu, provide a title that indicates the function the menu performs or the identity of the element or object associated with it.

[ ]

Describe which elements have a pop–up menu associated with them and how these pop–up menus can be used in the Help choice available through the window’s Help pull–down menu.

Choice Augmentation

[ ]

If you include the Delete or Clear choice in a pop–up menu, augment them with the words "to Trash" if and only if all elements to which the operation applies are objects.

Pop–Up Menus for Selections and Selectable Elements

[ ]

In conjunction with the **Menu Guidelines** reference page, use Pop–Up Menu Choices for Selections and Selectable Elements to decide which choices to include in a pop–up menu associated with a selection or a selectable element and how to organize them.

Table 1 Pop–Up Menu Choices for Selections and Selectable Elements

Mnemonic	Menu Choice
	Drag/Move
	Copy To ...
	Move To ...
D	Delete {to Trash}

[ ]

Make Undo or Repeat/Redo available from a pop–up menu (or a menu cascaded from it) associated with a selectable element only if the corresponding action applies to the selection or the selectable element on which the menu was popped up.

Pop–Up Menus for Selection Scope Backgrounds

[ ]

In conjunction with the Menu Guidelines reference page, use Pop–Up Menu Choices for Selection Scope Backgrounds to decide which choices to include in a pop–up menu associated with the background of a selectable scope and how to organize them.

Table 2 Pop–Up Menu Choices for Selection Scope Backgrounds

Mnemonic	Menu Choice
N	New
D	Delete {to Trash}

[ ]

Include the New choice in a pop–up menu only if it is associated with the background of a selection scope and if New appears in the corresponding Selected menu.

[ ]

Include action choices that act on a selection in a pop–up menu associated with the background only if there are elements in the selection scope that are selected.

Semantics of Pop–up Menu Actions for Selection Scopes

[ ]

If the pop–up menu is associated with a selectable element that is not currently selected and the action label is not augmented with the word "Selected" or "Selection," make the action act as if only the associated element were selected. For example, when the user selects Cut from a pop–up menu on an unselected element, cut just that element to the clipboard.

[ ]

Help on a pop–up menu associated with an element should have the same effect as pressing `Help` or `F1` when the active cursor is on an element.

Pop–Up Menus for Other Elements and Controls

[ ]

Do not change the data or selection state of an element when the pop–up menu associated with it is displayed.

[ ]

If you provide an action choice (other than Help) on a pop–up menu (or a menu cascaded from it) that is associated with a selection scope,

provide access to the same choice from either a menu bar, menu cascade button, push button in the same window, or a shortcut key.

Mouse Display and Deactivation

[ ]

When the user presses (not clicks) the MENU button at a location that allows a pop−up menu, display the associated pop−up menu at that location.

[ ]

When the user clicks the MENU button at a location that allows a pop−up menu and the menu is not currently displayed, display the associated pop−up menu at that location.

[ ]

When the user clicks the MENU button at a location that allows a pop−up menu and the menu is currently displayed, deactivate the associated pop−up menu at that location.

Keyboard Display and Deactivation

[ ]

If a pop−up menu is associated with the location of the active cursor and the user presses `Menu` or `Shift F10`, display the associated pop−up menu at that location.

[ ]

If focus is in a spring−loaded system based on a pop−up menu and the user presses `Menu` or `Shift F10`, deactivate the system.

Prefix Completion

Behavior

[ ]

When the user enters a printing character into a text−display field with prefix completion, move the cursor in the associated list to the next item that matches the prefix. Then, select the item and place its text contents into the text−display field. Leave the text cursor in place after the character that was just typed.

[ ]

When the user enters a printing character into a text−display field with prefix completion, and there is no item matching the prefix, generate a warning signal. Do not make any changes to the appearance of the list or of the text−display field.

[ ]

When the user enters a printing character in a text−display field with prefix completion, and the associated list is not displayed, do one of the following:

Display the list if `[darr ]` also displays the list.

Navigate in the list as indicated and select the appropriate list item, displaying it in the text field without displaying the list.

Prefix Navigation

Behavior

[ ]

When the user types a character in the text−entry field, move the cursor in the associated list to the next item whose label matches the prefix, enter the character into the text−entry field, and place the text cursor after the character. Keep the input focus in the text−entry field.

[ ]

When the user types a character in the text−entry field and there is no item matching the prefix in the list, generate a warning signal.

[ ]

If focus is in a text−entry field that uses prefix navigation, and the associated list box is not displayed, display the list box when the user types a printing character.

[ ]

When prefix navigation is used to move the cursor in a list, do not automatically select the cursored item unless required to do so by the selection model in use.

Primary Transfer

[ ]

If a scope uses primary selection, allow the selection to be used as the source of a primary transfer.

[ ]

If an element can be used as a target of a data transfer operation, support primary transfer to it.

[ ]

The source of a primary transfer is the primary selection.

[ ]

The target of a primary transfer is the control for which the primary transfer was invoked.

[ ]

The default operation for a TRANSFER button–based primary transfer is a copy, though it may be overridden with the `Ctrl` and/or `Shift` modifiers.

TRANSFER Button–Based Initiation

[ ]

When TRANSFER is bound to MB2, allow the user to click the TRANSFER button, optionally augmented with `Shift` and/or `Ctrl`, for primary transfers.

[ ]

If the user presses the TRANSFER button within a primary selection that can be dragged, initiate the drag.

[ ]

If TRANSFER is a separate mouse button from SELECT and the user presses the TRANSFER button outside of a primary selection, but at a position where both a drag can be initiated and data can be pasted by a primary transfer, then use the user–specified timeout and motion threshold in the following ways to distinguish drag initiation from primary transfer:

After the user presses the TRANSFER button, if the motion of the pointer exceeds the motion thresholds, initiate a drag.
If the user presses and releases the TRANSFER button within the timeout period, perform a primary transfer.
When the timeout period expires, initiate a drag. But, if the user releases the TRANSFER button before exceeding the motion threshold, cancel the drag.

Behavior

[ ]

Select the data transferred to a target scope as a result of a primary move operation if the target is using normal mode.

[ ]

A primary transfer must move the focus to the target control.

Primary Window

[ ]

Use a primary window as the main window of an application.

[ ]

Use a primary window to display a view of an object.

[ ]

When a primary window is closed or minimized, remove its secondary windows.

[ ]

When a primary window is closed, close that window and all of its unshared secondary windows.

[ ]

When a primary window that has associated secondary windows is opened or restored, restore the secondary windows that apply to the current conditions.

[ ]

When a primary window is closed, remove its window icon.

[ ]

When the last primary window of an application is closed, close the application.

[ ]

When a primary window that is critically needed to work with an application is closed, close the application, even if there are other displayed primary windows of that application.

Properties (Choice)

[ ]

Use a Properties choice to allow the user to change the properties of a file or object.

[ ]

Include a Properties choice in the File menu when the user can change the attributes of the selected file, for example, to change a read−only file to read−write.

[ ]

Use a multipage control to display properties if they cannot all be displayed at once.

[ ]

If multiple elements are selected, and the user chooses Properties, then for a given property, do the following:

Provide a description that encompasses all selected elements. For example, properties that can take only two values can be displayed as a check box that may either be set, unset, or indeterminate.

Provide a separate page for each element, with each page showing the value for that property.

Push Button (Control)

[ ]

If you do not provide a menu bar in a window with a viewing area, place all action, cascading, or dialog choices on push buttons in that window, except for those choices that appear on the window menu.

Push Button (Predefined)

[ ]

Use a push button with the predefined label when you provide the function defined as follows:

Apply

An action choice that appears in a window and makes the changes indicated in the window without closing it.

Cancel

An action choice that removes a secondary window without applying any changes made in that window.

Close

An action choice that removes a window and all of the windows associated with it from a screen.

Continue

An action choice that resumes a task that has been interrupted by the application operating environment when the user can proceed as originally requested.

Help

An action choice used on push buttons in secondary windows to provide help specific to that window.

A cascading choice that appears as a menu–bar item and provides access to other menu items that contain information related to the use of the application.

More

An action choice that displays additional controls in a separate window.

An action choice that expands the current dialog to show more value choices and replaces itself with the current dialog.

No

An action choice that indicates a negative response to a question presented in a message.

OK

An action choice that accepts the information in a window and closes it. If the window contains changed information, those changes are applied before the window is closed.

Pause

An action choice that temporarily suspends a task without ending it.

RESET

An action choice that resets the values displayed in a dialog or property window to the values they had when the window became displayed or when the values were last saved as defaults, whichever is most recent.

Reset To Default

An action choice that resets the values displayed in a dialog or property window to the values they had when default values were last saved.

Resume

An action choice that resumes a task that the user paused.

Retry

An action choice that attempts to complete an interrupted task.

Save As Default

An action choice that saves the values displayed in a dialog or property window as defaults to be used when the same (or similar) window is subsequently displayed to the same user by the application.

Stop

An action choice that ends a task and removes the message window.

Undo

An action choice that undoes the effect of the last Apply.

Yes

An action choice that indicates a positive response to a question presented in a message by an application or the system.

[ ]

If you provide the Cancel and Help push buttons, place them to the right of all other push buttons. For information on bidirectional and vertical language support, see Chapter 11.

[ ]

Do not use both a Close push button and a Cancel push button in the same window.

[ ]

Provide a Resume push button whenever you provide a Pause push button.

[ ]

When the user makes changes in a dialog or property window and then activates Reset before committing them, return the application to the state that it was in before the user made the changes. Do not reset changes that have been previously committed, for example when the user has selected Apply or OK.

[ ]

When the user selects Reset, restore the settings of values only in the window where Reset was activated.

[ ]

Provide a Reset push button whenever you provide an Apply or OK push button.

Quick Transfer

[ ]

Allow users to use quick transfer when they want to copy, link, or move a temporary (or secondary) selection.

[ ]

Design text controls to support quick transfer as both a source and target.

[ ]

Make the target of a quick transfer the control that has interacted emphasis.

[ ]

Make the source of a quick transfer the elements identified between pressing and releasing the TRANSFER (or SELECT) button.

[ ]

The default operation for a quick transfer should be copy, though it may be overridden using `Ctrl` and/or `Shift` modifiers when the user releases the TRANSFER button.

TRANSFER Button–Based Initiation

[ ]

Allow the user to press `Alt` TRANSFER and move the pointer (optionally augmented with `Shift` and/or `Ctrl`) for quick transfer only when TRANSFER is bound to MB2.

Behavior

[ ]

Do not select the data transferred to a target as the result of a quick cut operation.

[ ]

If the user moves the pointer out of a control that allows scrolling while `Alt` TRANSFER is pressed, use autoscrolling to scroll the control in the direction of the pointer. If TRANSFER is released with the pointer outside of the control, or if `Cancel` (or `Esc`) is pressed while TRANSFER is pressed, then remove the emphasis, do not perform the transfer, and restore the view to the state it was in before scrolling started.

Quick Copy

[ ]

Allow either of the following methods to copy the secondary selection to the insertion point:

Pressing `Alt` TRANSFER and moving the mouse

Pressing `CtrlAlt` TRANSFER and moving the mouse

Quick Cut

[ ]

Move the contents of the secondary selection to the insertion point in the target control when the user presses `Shift Alt` TRANSFER, moves the mouse, then releases the TRANSFER button.

Quick Link

[ ]

Place a link to the secondary selection at the insertion point in the target control when the user presses `CtrlAlt Shift` TRANSFER, moves the mouse to select elements, and then releases the TRANSFER button.

Radio Button (Control)

[ ]

The radio buttons in a radio box are mutually exclusive. When the user sets one radio button, unset any other radio button in the radio box.

Default Radio Buttons

[ ]

If the values represented by the radio buttons in a radio box cover all possibilities, do not allow the user to unset the radio button that is set. Attempting to do so should have no effect.

[ ]

If the values represented by the radio buttons in a radio box do not cover all possibilities, then allow the user to unset the radio button that is set.

Selection Properties

[ ]

When a radio box represents a value shared by multiple selected elements:

Display the radio buttons as selected if that value applies to all of the selected elements.

Do not display any of the radio buttons as selected if not all of the selected elements have a matching value.

[ ]

When a radio box represents a property shared by more than one selected element, ensure that the property of all selected elements corresponds to the radio button that the user sets.

Layout

[ ]

Design a radio button with a graphic that indicates the state of the button and a text label that describes the state it controls.

[ ]

Use at least two radio buttons in a radio box.

Range Adjust Click Technique

[ ]

When the user is using the range adjust click technique, and the adjustment policy is to reselect:

Determine the new selection region to be the range identified by the new anchor point and the adjustment point.

Enlarge the new selection region to include the anchor region, if indicated by the anchor inclusion policy.

[ ]

When the user is using the range adjust click technique, and the adjustment policy is to enlarge, make the new selection region the larger of the following:

Current selection region

The region determined by the reselect adjustment policy

[ ]

When the user is using the range adjust click technique, and the adjustment policy is to balance:

Move (if necessary) the anchor point so that it is at the end of the current selection region farthest from the adjustment point.

Determine the new selection region to be the range identified by the new anchor point and the adjustment point.

Enlarge the new selection region to include the anchor region, if the anchor point is unchanged and if indicated by the anchor inclusion policy.

[ ]

If the current selection region did not include an anchor element or if the anchor element is no longer in the selection region, identify the anchor element to be the element, if any, at the end of the range nearest the anchor point.

[ ]

After the user uses a mouse−based range adjust click technique, place the active cursor at one of the following:

If it is a text cursor, at the point at which the ADJUST button is released

If it is an element cursor, on the element at that end of the range

Range Adjust Swipe Technique

[ ]

If you are supporting discontiguous selections, support only the keyboard−based range adjust swipe technique when the initial adjustment point is at one end of the current selection region. Otherwise, support the range swipe technique.

[ ]

When discontiguous selections are not supported, and normal mode is in use, support only the keyboard−based range adjust swipe technique when the initial adjustment point is within or at one end of the current selection region. Otherwise, support the range swipe technique.

[ ]

When the user is using the range adjust swipe technique, and the adjustment policy is to reselect:

Determine the new selection region to be the range identified by the anchor point and the final adjustment point.

Enlarge the new selection region to include the anchor region, if indicated by the anchor inclusion policy.

[ ]

When the user is using the range adjust swipe technique, and the adjustment policy is to enlarge, make the new selection region the larger of the following:

The current selection region

The region determined using the reselect adjustment policy

[ ]

When the user is using the keyboard−based area adjust swipe technique, and the adjustment policy is to balance:

If the final adjustment point is not within the current selection region, move (if necessary) the anchor point so that it is at the end of the current selection region farthest from either the initial or the final adjustment point. The use of the final adjustment point is recommended.

If the final adjustment point is within the current selection region, or when using the mouse−based technique, move (if necessary) the anchor point so that it is at the end of the current selection region farthest from the initial adjustment point.

[ ]

When the user is using the area adjust swipe technique, and the adjustment policy is to balance:

Determine the new selection region to be the range identified by the new anchor point and the final adjustment point.

Enlarge the new selection region to include the anchor region if the anchor point is unchanged and if indicated by the anchor inclusion policy.

[ ]

If the current selection region does not include an anchor element or if the anchor element is no longer in the selection region, identify the anchor element to be the element, if any, at the end of the range nearest the anchor point.

[ ]

After the user uses a mouse−based range adjust swipe technique, place the active cursor at one of the following:

If it is a text cursor, at the point at which the ADJUST button is released

If it is an element cursor, on the element at that end of the range

Range Click Technique

[ ]

Support use of the range click technique only when the elements in the scope form a natural sequential order, and when the most likely set of elements to be selected is a subsequence of those elements. This is most common in text and sometimes true in lists.

[ ]

Support the mouse−based range swipe technique in toggle mode only when discontiguous selections are supported.

[ ]

Support use of the keyboard−based range click technique in normal mode only when a text cursor is used for all navigation between the two ends of the range. The result of attempted use in other cases is undefined.

Mouse−Based Technique

[ ]

After the user uses a mouse−based range click technique:

- Define the current selection region to consist of the identified range.
- Identify the anchor point to be the point at which the SELECT button was clicked.
- Identify the anchor element to be the element, if any, within the selection region nearest to the anchor point.

[ ]

After the user uses a mouse−based range click technique, place the active cursor at one of the following:

- The point at which ADJUST is clicked, if it can be placed there
- If it is an element cursor, on the element at the end of the range at which the ADJUST button is clicked
- If there are no elements in the range, where the cursor previously was in the scope

[ ]

When the user uses the range click technique in select mode:

- Select all the elements in the indicated range.
- Deselect all other elements in the scope.

[ ]

If currently in select mode and the user uses the range click technique by clicking `Ctrl` SELECT in place of SELECT, activate toggle mode.

[ ]

When the user is using the range click technique in toggle mode:

- Toggle all elements in the indicated range, based on the toggling policy.
- Do not change the selection state of the other elements in the scope.

Keyboard−Based Technique

[ ]

After the user uses a keyboard−based range click technique:

- Define the current selection region to consist of the identified range.
- Identify the anchor point to be the point at which the user pressed `Select`, `Space` (unless in text), or `Ctrl` `Space`.
- Identify the anchor element to be the element, if any, within the selection region nearest to the anchor point.

[ ]

When the user is using the keyboard−based range click technique in normal mode:

Select all the elements in the indicated range.
Deselect all other elements in the scope.

[ ]

When the user is using the keyboard−based range click technique in add mode, and contiguous selections are required:
Select all the elements in the indicated range.
Deselect all other elements in the scope.

[ ]

When the user is using the keyboard−based range click technique in add mode, and discontiguous selections are allowed:
Toggle all elements in the indicated range, based on the toggling policy.
Do not change the selection state of the other elements.

Range Swipe Technique

[ ]

Support use of the range swipe technique only when the elements in the scope form a natural sequential order and when the most likely set of elements to be selected is a subsequence of those elements. This is most common in text and sometimes true in lists.

[ ]

Support the mouse−based range swipe technique in toggle mode only when discontiguous selections are supported.

[ ]

When discontiguous selections are not supported and add mode is in use, do not support the keyboard−based range swipe technique. Use the range adjust swipe technique instead.

[ ]

When discontiguous selections are not supported, and normal mode is in use, support the keyboard−based range swipe technique only when the initial adjustment point is disjoint from the current selection region. Otherwise use the range adjust swipe technique.

[ ]

When discontiguous selections are supported, support the keyboard−based range swipe technique only when the initial adjustment point is not at one end of the current selection region. Otherwise use the range adjust swipe technique.

Mouse−Based Technique

[ ]

After the user uses a mouse−based range swipe technique:
- Define the current selection region to consist of the identified range.
- Identify the anchor point to be the point at which the SELECT button was pressed.
- Identify the anchor element to be the element, if any, within the selection region nearest to the anchor point.

[ ]

After the user finishes a mouse−based range swipe technique, place the active cursor:
- At the point at which the user released the SELECT button, if it can be placed there
- If an element cursor, on the element at the end of the range at which the user released the SELECT button
- If there are no elements in the range, where the cursor previously was in the scope

[ ]

When the user is using the range swipe technique in select mode:
- Select all the elements in the indicated range.
- Deselect all other elements in the scope.

[ ]

If the user is currently operating the range swipe technique in select mode and presses `Ctrl` SELECT in place of SELECT, change to toggle mode.

[ ]

When the user is using the range swipe technique in toggle mode:
- Toggle all elements in the indicated range based on the toggling policy.
- Do not change the selection state of the other elements.

Keyboard−Based Technique

[ ]

After the user uses a keyboard−based range swipe technique:

Define the current selection region to consist of the identified range.

Identify the anchor point to be the initial point at which the user pressed the `Shift` navigation key.

Identify the anchor element to be the element, if any, within the selection region nearest to the anchor point.

[ ]

When the user is using the keyboard−based range swipe technique in normal mode:

Select all the elements in the indicated range.

Deselect all other elements in the scope.

[ ]

When the user is using the keyboard−based range swipe technique in add mode:

Toggle all elements in the indicated range, based on the toggling policy.

Do not change the selection state of the other elements in the scope.

Restore (Choice)

[ ]

If a window can be minimized, provide a Restore choice in the window icon's window menu to allow the user to return the window from the minimized state to its previous size and position.

[ ]

If a window can be maximized, provide a Restore choice to allow the user to return the window from the maximized state to its previous size and position.

[ ]

If a window is maximized and the user chooses Restore from the window menu, restore the window to the size and position it had before it was maximized.

[ ]

If a window is minimized (iconified) and the user chooses Restore from the window icon's window menu, restore the window to the size and position it had before it was minimized.

[ ]

If a primary window is minimized and the user invokes Restore, restore all of the secondary windows that the application has not closed.

[ ]

Do not make the Restore choice available when the window is neither minimized nor maximized.

Visuals

[ ]

Provide a restore button that implements the Restore choice on the title bar when the window is currently maximized.

[ ]

Replace the restore button with a maximize button in the title bar after a window has been restored to a state other than maximized.

[ ]

Replace the maximize button with a restore button in the title bar when the window has been maximized.

Default Action

[ ]

Make the Restore choice the default action associated with the window icon. If the window was maximized before it was minimized, restore it to its maximized state.

[ ]

Allow the user to invoke the default action of the window icon by double−clicking the SELECT button on it, or by pressing `Enter` or `keypadEnter` when it has focus.

Sash (Control)

[ ]

Design a sash to allow the user to adjust the position of the split bar and the size of the panes next to it.

[ ]

If you provide a sash, allow the user to navigate to it by pressing `tab`.

Mouse Behavior

[ ]

Allow the user to adjust the size of the panes in the split window by adjusting the position of the split bar with the sash (if the `Shift` key is not pressed).

[ ]

When a user moves the split bar with the sash and does not press the `Shift` key, shrink the size of the pane in the direction of movement. Enlarge the size of the pane on the opposite edge of the split bar by an equal amount.

[ ]

Allow the user to adjust the position of the split bar by both pointer operations and keyboard methods.

[ ]

When the user drags a sash while pressing the SELECT button or TRANSFER button, have the sash track the movement of the pointer. In a vertically oriented paned window, have the sash track the vertical position of the pointer. In a horizontally oriented paned window, have the sash track the horizontal position of the pointer.

Keyboard Behavior

[ ]

When the user presses the following keys with focus on the sash, move the sash as follows:

`[darr ]` and `[uarr ]`

If the sash can move vertically, move the sash one increment in the specified direction.

`[larr ]` and `[rarr ]`

If the sash can move horizontally, move the sash one increment in the specified direction.

`Ctrl[uarr ]` and `Ctrl[darr ]`

If the sash can move vertically, move the sash one large increment in the specified direction.

`Ctrl[rarr ]` and `Ctrl[larr ]`

If the sash can move horizontally, move the sash one large increment in the specified direction.

Visuals

[ ]

Make the sash look like a handle on the split bar separating two panes of a paned box.

[ ]

If the split bar has a sash, place the sash near the right end of the separators in a vertically split paned box or near the bottom end of the

separators in a horizontally split paned box. Center the sash on the split bar in the other dimension.

Save/Save As (Action Choice)

[ ]

Place the Save and Save As choices in the File menu.

[ ]

Provide a Save choice for each file or object with a specified name that is not automatically saved when changed.

[ ]

Provide a Save As choice for each file or object that the user can save with a new name.

[ ]

Do not close the window or change the appearance of the data when the user chooses Save or Save As.

[ ]

The file or object to be saved should be the last file the user interacted with using Save As or Move To. Otherwise, the file or object to be saved should be the one that the user opened in the current window.

[ ]

If the user attempts to save a new file or object under an existing name, display a warning message stating that a loss of data will occur. For example:

```
myfile already exists.
Choose OK to overwrite.
```

[ ]

If the file or object being saved does not have a user−assigned name, either make Save unavailable or display the Save As dialog.

[ ]

If the user cannot save the data in the current file or object (for example, a file is read−only), display a warning message and allow the user to either cancel the operation or provide a new destination via a Save As dialog.

Save As Dialog

[ ]

Display the Save As dialog in a secondary window that is dependent on the window from which the user chooses Save As.

[ ]

Provide a Save (or OK) push button to allow the user to save the current file or object with the specified name.

[ ]

Provide a Cancel push button to allow the user to close the Save As dialog without saving the file or object.

[ ]

Provide a text−entry field to allow the user to type the name of the new file or object.

[ ]

Provide a drop−down list that allows the user to display a list of file types that can be saved.

[ ]

For objects, provide a list of appropriate containers. Highlight the current container and show the containment hierarchy by indenting

containers that are within other containers.

[ ]

For objects, provide a list box that lists the names of all objects in the specified container.

Mnemonics

[ ]

Assign "S" as the mnemonic for the Save menu item.

[ ]

Assign "A" as the mnemonic for the Save As menu item.

Scroll Bar (Control)

[ ]

Do not use a scroll bar to represent numeric values. Use a slider instead.

[ ]

Adjust the size of a scroll box so that it is proportional to the amount of information that can be visible in the area that can be scrolled in relation to the total amount of information.

[ ]

Adjust the position of a scroll box so that it indicates the position of the information visible in the area that can be scrolled in relation to the total amount of information.

[ ]

Change the size of a scroll box only when one of the following is true:

- Information is added to or removed from data represented in the area that can be scrolled
- The window's size changes

Mouse Behavior

[ ]

When the user presses the SELECT button on the scroll bar's arrow button, scroll the associated scrolling area one unit in the direction of the arrow and repeat until the button is released.

[ ]

When the user presses the SELECT button in the scroll bar's scroll track, scroll the associated scrolling area one page in the direction of the mouse position relative to the scroll box and autorepeat (in the same direction) until the button is released.

[ ]

When the user presses the SELECT button in the scroll box, track the appropriate mouse coordinate with the scroll box, correspondingly scrolling the associated scrolling area, until the button is released. If the user moves the mouse perpendicularly away from the scroll bar (beyond a range specified by the operating environment), snap the scroll box back to its initial position at the time the user presses SELECT. When the user moves the mouse back into the scrolling range, begin tracking the mouse again.

[ ]

When the user presses the TRANSFER button in the scroll box or track, move the scroll box to the pointer position and scroll the corresponding area. While TRANSFER remains pressed, track the mouse with the scroll box in the exact same manner as when the user presses SELECT in the scroll box.

Keyboard Behavior

[ ]

When the user presses a directional key while focus is in the scroll bar, scroll the associated scrolling area one unit in the indicated direction.

[ ]

When the user presses `Ctrl` plus a directional key while focus is in the scroll bar, scroll the associated scrolling area one large scrolling increment in the indicated direction.

[ ]

When the user presses `Begin` (or `Home`) or `End` in a horizontally aligned scroll bar, scroll the associated scrolling area to the leftmost or

rightmost end, respectively.

[ ]

When the user presses `Ctrl Begin` (or `Ctrl Home`) or `Ctrl End` in a vertically aligned scroll bar, scroll the associated scrolling area to the topmost or bottommost end, respectively.

[ ]

When the user presses `PageUp` or `PageDown` in a horizontally aligned scroll bar, scroll the associated scrolling area by one page in the indicated direction.

[ ]

When the user presses `PageLeft` (or `Ctrl PageUp`) or `PageDown` (or `Ctrl PageRight`) in a vertically aligned scroll bar, scroll the associated scrolling area by one page in the indicated direction.

Scrolling

Scrolling Increment

[ ]

Make the unit scrolling increments (for example, the smallest amount a scrolled area can be scrolled when using an associated scroll bar) equal to those shown in Scrolling Increments .

Table 1 Scrolling Increments

Type of Element	Unit Scrolling Increment		
	Horizontal	Vertical	
Text	Width of em (M) character		I c l b c l
Icons or controls	Width of smallest icon or control		I c s i c
Graphic	Five percent of the width of the view, or by units of a scale specified by the user		I p c h t v b c s s b u

Paging

[ ]

When the user presses `PageUp` or `PageDown` and focus is on a scrollable area, scroll up or down by one page.

[ ]

When the user presses `PageLeft` (or `Ctrl PageUp`) or `PageRight` (or `Ctrl PageDown`) and focus is on a scrollable area, scroll left or right by one page.

[ ]

When the user is using the paging keys to scroll (and `ScrollLock` is not used), move the active cursor within the scrollable area so that it is visible, unless the cursor cannot be placed in the visible area after scrolling.

[ ]

Paging keys should apply to the most deeply nested appropriate scrollable area that contains the active cursor.

Navigation and Scrolling

[ ]

If a scrolled control (for example, a text field) maintains a cursor location even when it does not have focus, then when keyboard navigation is used to move focus to that control, scroll it, if necessary, so that the cursor location is visible.

[ ]

When the user navigates to an element within a scrollable area, scroll the area so that the element is visible.

[ ]

When any keyboard operation moves the cursor within a scrolled control, or inserts, deletes, or modifies elements at the cursor position, scroll the control so that the cursor is visible when the operation is complete.

[ ]

Design navigation operations to traverse through an entire scrollable area, not just the visible portion of it.

Scrolling With a Stationary Cursor

[ ]

If the user can change the size of a scrollable control or area, and the cursor is visible in that area, scroll it so that the cursor remains visible.

[ ]

When the user scrolls with a mouse, do not move the cursor in the underlying data; instead, allow it to be scrolled outside of the visible area.

[ ]

If the active cursor is within a scrolled area, but is not visible, indicate focus emphasis by placing an element cursor on the entire scrolled area.

Mouse−Based Scrolling

[ ]

When the user presses `Cancel` (or `Esc`), cancel the mouse−based scrolling action and return the scrolling area to its position prior to the start of the scrolling operation.

Autoscrolling

[ ]

When a mouse−based selection operation is in progress within a control that supports scrolling and the user moves the pointer outside of the control, scroll towards the pointer. This is called "selection autoscrolling."

Secondary Window

[ ]

Display a secondary window when the user activates a dialog choice or, more generally, to allow the user to further specify information that is needed to complete a request.

[ ]

Use a secondary window to display a message.

[ ]

Use a secondary window to show properties associated with an object or other element.

[ ]

Use a secondary window to hold a tear−off control.

[ ]

Use a secondary window when a More choice provides access to additional controls.

[ ]

Do not show unavailable emphasis on a dialog choice that caused the display of a secondary window just because the window is still displayed.

Modality

[ ]

Support the following four modes of interaction with secondary windows:

Modeless

Allows interaction with the secondary window and all other windows, unless the secondary window was activated by a dialog choice in a modal window. In that case, limit interaction to windows displayed subsequent to the activation of the dialog choice.

Primary modal

Does not allow interaction with any ancestor of the window.

Application modal

Does not allow interaction with any window created by the same application except for this window and its secondary descendants, even if the application has multiple primary windows.

System modal

Does not allow interaction with any window on the workspace. This includes windows from all other applications and any window icon view. To indicate a system modal secondary window, the pointer should change shape to a caution pointer whenever it leaves the system modal secondary window.

[ ]

Do not display a secondary window as a system modal window unless your application is a manager application that requires that the user be restrained from performing any other desktop operation.

[ ]

Do not display a secondary window as an application modal window unless the task in progress in your application requires that the user be restrained from changing the state of the application in any way.

[ ]

Do not allow a modal secondary window to be resized or maximized.

Visuals

[ ]

If your application provides an Apply push button with a secondary window, provide an OK push button that, when activated, performs the default action for the secondary window and then closes the secondary window.

[ ]

Place the controls that the user can activate and that affect the contents or layout of the secondary window at the bottom of the secondary window.

Displaying Secondary Windows

[ ]

Remove a secondary window when the user closes or minimizes the primary window it depends on.

[ ]

Do not remove a secondary window just because the primary window it depends on loses input focus, except possibly those containing torn−off controls.

[ ]

If a secondary window is open when its associated primary window is closed or minimized, display that secondary window when its associated primary window is reopened or restored.

[ ]

When a secondary window is closed, close any secondary windows dependent on it. However, do not affect the associated primary window or any window that the secondary window is dependent on.

[ ]

If a secondary window can be stacked below its associated primary window, place it so that it will not be completely covered by the primary window.

Selected Menu

[ ]

In conjunction with the **Menu Guidelines** reference page, use Selected Menu Choices to decide which choices to include in the Selected menu and how to organize them.

Table 1 Selected Menu Choices

Mnemonic	Menu Choice	Keyboard Function
N	New	`Ctrl N`
O	Open	`Ctrl O`
D	Delete to Trash	`Delete`

[ ]

Use `CtrlN` and `Ctrl O` for the New and Open entries on the Selected menu only if your application does not have a state or contents that needs to be named and persistently stored.

[ ]

If your application manages and displays only one type of object that the user can select, replace the Selected label with one naming the type of object.

[ ]

Include type−specific choices at the bottom of the Selected menu that apply to the types of objects selected. Add and remove type−specific choices (rather than making those choices unavailable) as the selection changes.

Guidelines for Specific Menu Entries

[ ]

Print out the contents of each of the selected objects when the user chooses the Print choice on the Selected menu.

Selection

[ ]

A selection scope must use, at any one time, either a primary selection approach or a persistent selection approach.

[ ]

When a scope uses a primary selection approach, deselect the primary selection when the user makes a nonempty primary selection in another scope.

[ ]

Selections in a scope that use a persistent selection approach should be unaffected by selections made in a different scope.

[ ]

If your application selection model allows at most one element at a time to be selected, use a persistent selection approach.

On a Selection

[ ]

Allow a selection if the pointer or cursor is on a selected element that cannot be activated or toggled (for example, a push button within a selected text region).

[ ]

Allow a selection if the pointer or cursor is between elements in a selected text region.

Pending Delete

[ ]

If pending delete is enabled, allow the user to delete the contents of a selection before performing a transfer by inserting or pasting elements into the selection (except possibly on a link operation or, for a primary copy, at the edge of the primary selection).

[ ]

Support pending delete only when a text or graphics cursor is used.

[ ]

Disable pending delete behavior if it would lead to deletion of an object.

Enable pending delete behavior when a text cursor is used.

Disable pending delete behavior when an element cursor is used.

Selection (CDE)

Selection Models

[ ]

There are five selection models: single selection, browse selection, multiple selection, range selection, and discontiguous selection. Each collection should have one or more appropriate selection model. The model limits the kinds of choices the user can make in the collection. Some collections enforce a selection model, while others allow the user or application to change it.

Mouse−Based Single Selection

[ ]

In a collection that uses single selection, when the user clicks the SELECT button on a deselected element, move the location cursor to that element, selected it, and deselect any other selection in the collection. Single selection is the simplest selection model for selecting a single element.

Mouse−Based Browse Selection

[ ]

In a collection that uses browse selection, when the user releases the SELECT button on a selectable element, select that element, and deselect any other selection in the collection. As the pointer is dragged through selectable elements, select each element under the pointer and deselect the previously selected element. The selection should remain on the element where the SELECT button was released, and the location cursor should moved there.

Browse selection selects a single element. It also allows the user to browse through the collection by dragging the pointer.

Mouse−Based Multiple Selection

[ ]

If your application contains collections that follow the multiple selection model make the ADJUST button behave like the SELECT button, when the TRANSFER button is configured to behave as the ADJUST button.

[ ]

Allow the user to change an environment setting indicating that MB2 should be used for the adjust function instead. The ADJUST button can be used to toggle the selection state of elements under the multiple selection model.

[ ]

In a collection that uses multiple selection, clicking the SELECT button or the ADJUST button on an unselected element adds that element to the current selection. Clicking the SELECT button or the ADJUST button on a selected element removes that element from the current selection. Clicking the SELECT button or the ADJUST button moves the location cursor to that element.

Mouse−Based Range Selection

[ ]

In a collection that uses range selection, pressing the SELECT button on an unselected element sets an anchor on the element, or at the position where the SELECT button was pressed, and deselects all elements in the collection. If the SELECT button is released before the drag threshold has been exceeded, then the element under the pointer should be selected. If mouse motion exceeds the drag threshold, then a new selection should begin. The anchor and the current position of the pointer determines the current range. As the pointer is dragged through the collection, highlight the current range. When the SELECT button is released, the anchor should not move and all the elements within the current range should be selected.

[ ]

In a collection that uses range selection, pressing the SELECT button on a currently selected element should not cause all other elements in the selection set to be deselected. If the SELECT button is released before the drag threshold is exceeded, then, at that point, all other elements should be deselected and the element under the pointer should remain selected. If mouse motion exceeds the drag threshold, then no element should be deselected and a drag operation should begin.

[ ]

In a text−like collection that uses range selection, make the anchor point the text pointer position when the SELECT button is pressed. The current range should consist of all elements between the anchor point and the current text pointer position.

[ ]

In text−like collections, order elements linearly, and always consider a text pointer to be between elements at a point near the actual pointer position.

[ ]

In a graphics−like or list−like collection that uses a marquee to indicate the range of a range selection, the current range should consist of those elements that fall completely within the marquee. If there is an anchor element, always make the marquee large enough to enclose it completely. Otherwise, use an anchor point to be the point at which the SELECT button was pressed; the anchor point determines one corner of the marquee. If the collection is not arranged as a list or matrix, extend the marquee to the pointer position. If the collection is arranged as a list or matrix, extend the marquee either to completely enclose the element under the pointer or to the pointer position. Clicking the SELECT button on a selectable element makes it an anchor element, selects it, and deselects all other elements.

[ ]

If your application contains collections that follow the range selection model, make the ADJUST button behave like `Shift` SELECT when the TRANSFER button is configured to behave as the ADJUST button.

[ ]

Allow the user to change an environment setting that indicates MB2 should be used for the ADJUST button function. The ADJUST button can then be used to extend the selection set in the same manner as `Shift` SELECT.

[ ]

In a collection that uses range selection, when the user presses `Shift` SELECT or the ADJUST button, the anchor should remain unchanged and an extended range for the selection is determined, based on one of the extension models.

Mouse−Based Discontiguous Selection

[ ]

In a collection that uses discontiguous selection, make the behavior of the SELECT button exactly the same as in the range selection model. After the user sets the anchor with the SELECT button, `Shift` SELECT should work exactly as in the range selection model.

[ ]

In a collection that uses discontiguous selection, when the current selection is not empty and the user clicks `Ctrl` SELECT, move the anchor and location cursor to that point. If the current selection is not empty and the user clicks `Ctrl` SELECT on an element, toggle the selection state of that element and make it the anchor element.

[ ]

In a collection that uses discontiguous selection, pressing `Ctrl` SELECT and moving the pointer toggles the selection state of a range of elements. The range itself is determined exactly as for the pointer motion. Releasing `Ctrl` SELECT toggles the selection state of the elements in the range according to one of two models:

Anchor toggle

> Toggling is based on an anchor element. If the range is anchored by a point and is not empty, the anchor element is set to the element within the range that is nearest to the anchor point. Toggling sets the selection state of all elements in the range to the inverse of the initial state of the anchor element.

Full toggle

The selection state of each element in the extended range is toggled.

[ ]

In a collection that uses discontiguous selection, after `Ctrl` SELECT toggles a selection, pressing `Shift` SELECT or `Ctrl Shift` SELECT extends the range of toggled elements. The extended range is determined in exactly the same way as when `Shift` SELECT is used to extend a range selection. When the user releases `Ctrl Shift` SELECT, the selection state of elements added to the range is determined by the toggle model in use (either anchor toggle or full toggle). If elements are removed from the range, they either revert to their state prior to the last use of `Ctrl` SELECT or change to the state opposite that of the elements remaining within the extended range.

[ ]

In a collection that uses discontiguous selection, allow the ADJUST button to be used to extend the range of a discontiguous selection. The extended range is determined in exactly the same way as when the ADJUST button is used to extend a range selection.

[ ]

Allow the user to change an environment setting that indicates MB2 should be used for the ADJUST button function. The ADJUST button can then be used to extend the selection set in the same manner as `Shift` SELECT.

Keyboard Selection

[ ]

The selection models should support keyboard selection modes according to the following rules:

Single selection supports only add mode.
Browse selection supports only normal mode.
Multiple selection supports only add mode.
Range selection supports normal mode. If it also supports add mode, normal mode is the default.
Discontiguous selection supports both normal mode and add mode. Normal mode is the default.

[ ]

Selection must be available from the keyboard. In normal mode, which is used for making simple contiguous selections from the keyboard, the location cursor is never disjoint from the current selection. In add mode, which is used for making more complex and possibly disjoint selections, the location cursor can move independently of the current selection.

[ ]

If a collection supports both normal mode and add mode, pressing `Shift F8` switches from one mode to the other. Mouse–based selection should not change when the keyboard selection mode changes. In editable components, add mode is a temporary mode that is exited when the user performs an operation on the selection or deselects the selection.

Keyboard–Based Single Selection

[ ]

In a collection that uses single selection, pressing the navigation keys move the location cursor independently from the selected element. Pressing `Select` or `Spacebar` on an unselected element, selects the element with the location cursor and deselects any other selection in the collection.

Single selection supports only add mode. Pressing `Select` or `Spacebar` is similar to clicking the SELECT button.

Keyboard–Based Browse Selection

[ ]

In a collection that uses browse selection, pressing the navigation keys move the location cursor and select the cursored element, deselecting any other element. If the application has deselected all elements or if the cursor is left disjoint from the selection, pressing the `Select` key or the `Spacebar` selects the cursored element and deselects any other element.

Browse selection supports only normal mode. A navigation operation is similar to dragging the SELECT button.

Keyboard−Based Multiple Selection

[ ]

In a collection that uses multiple selection, the navigation keys move the location cursor independently from the current selection. Pressing `Select` or `Spacebar` on an unselected element adds the element to the current selection. Pressing `Select` or `Spacebar` on a selected element removes the element from the current selection.

Multiple selection supports only add mode. Pressing `Select` or `Spacebar` is similar to clicking the SELECT button.

Keyboard−Based Range Selection

[ ]

In a collection that uses range selection and is in normal mode, pressing the navigation keys move the location cursor and deselect the current selection. If the cursor is on an element, select it and moves the anchor with the location cursor.

[ ]

Text−like collections can use a different model in which the navigation keys leave the anchor at its current location, except that if the current selection is not empty it is deselected and the anchor is moved to the location of the cursor prior to navigation. Range selection supports normal mode; if the collection also supports add mode, make normal mode the default.

[ ]

In a collection that uses range selection, whether in normal mode or add mode, pressing `Select` or `Spacebar` (except in a text component) moves the anchor to the cursor, deselects the current selection, and, if the cursor is on an element, selects the element. Unless the anchor is on a deselected item, pressing `Shift Select` or `Shift Spacebar` (except in text) extends the selection from the anchor to the cursor, based on the extension model that `Shift` SELECT uses (Reselect, Enlarge Only, or Balance Beam).

[ ]

In a range selection, pressing `Select` or `Spacebar` is similar to clicking the SELECT button, and pressing `Shift Select` or `Shift Spacebar` extends the range as with `Shift` SELECT.

[ ]

In a collection that uses range selection and is in normal mode, pressing `Shift` in conjunction with the navigation keys extends the selection, based on the extension model that `Shift` SELECT uses. If the current selection is empty, first move the anchor to the cursor. Then move the cursor according to the navigation keys and extend the selection based on the extension model that `Shift` SELECT uses.

[ ]

In a range selection, make shifted navigation extend the selection in a manner similar to dragging the pointer while holding `Shift` SELECT.

[ ]

In a collection that uses range selection and is in add mode, pressing the navigation keys move the location cursor but leave the anchor unchanged.

Make shifted navigation move the location cursor according to the navigation keys and extend the selection, based on the extension model that `Shift` SELECT uses.

[ ]

Make shifted navigation in add mode similar to shifted navigation in normal mode, except that when the selection is empty the anchor should not move to the cursor prior to navigation.

Keyboard−Based Discontiguous Selection

[ ]

In a collection that uses discontiguous selection and is in normal mode, all keyboard operations should have the same effect as in the range selection model. Do not permit multiple discontiguous selections in normal mode.

[ ]

In a collection that uses discontiguous selection and is in add mode, pressing `Select` or `Spacebar` moves the anchor to the location cursor and initiates toggling. If the cursor is on an element, toggle the selection state of that element, but keep the selection state of all other elements unchanged. Pressing `Shift Select` or `Shift Spacebar` and shifted navigation operations extend the selection between the anchor and the location cursor, based on the toggle mechanism that `Ctrl` SELECT uses (anchor toggle or full toggle). In add mode, permit the use of the keyboard to make multiple discontiguous selections.

Canceling a Selection

[ ]

The `Cancel` key cancels or undoes any incomplete motion operation used for selection. Once the user presses `Cancel` to cancel a motion operation, ignore subsequent key and button releases until after all buttons and keys are released. Pressing `Cancel` while extending a selection or toggling leaves the selection state of all elements as they were prior to the button press.

Autoscrolling and Selection

[ ]

If the user drags the pointer out of a scrollable collection during a motion−based selection operation, use autoscrolling to scroll the collection in the direction of the pointer. If the user presses `Cancel` with the SELECT button pressed, cancel the selection operation.

Selecting and Deselecting All Elements

[ ]

In a collection that uses multiple, range, or discontiguous selection, pressing `Ctrl /` selects all the elements in the collection, places the anchor at the beginning of the collection, and leaves the location cursor at its previous position.

[ ]

In a collection that is in add mode, pressing `Ctrl \` deselects all the elements in the collection. In a collection that is in normal mode, pressing `Ctrl \` deselects all the elements in the collection, except the element with the location cursor if the location cursor is being displayed. In either mode, pressing `Ctrl \` leaves the location cursor at its current position and moves the anchor to the location cursor.

Using Mnemonics for Elements

[ ]

If your application supports mnemonics associated with selectable elements, typing a mnemonic while the collection has keyboard focus should be equivalent to moving the location cursor to the element and pressing `Select` or `Spacebar`.

Selection Actions

[ ]

When the keyboard focus policy is explicit, the destination component should be the editable component that last had keyboard focus. When the keyboard focus policy is implicit, the destination component should be the editable component that last received mouse button or keyboard input.

The destination component is used to identify the component on which certain operations, primarily data transfer operations, act. There can be only one destination component at a time.

[ ]

If the keyboard focus is in a component (or a pop−up menu of a component) that supports selections, operations that act on a selection

should act on the selection in that component.

A selection operation should act on the component that has focus, if that component supports selections.

[ ]

If the keyboard focus is in a component (or a pop–up menu of a component) that supports some operation that does not act on a selection, invoking the operation should act on that component.

An operation that does not act on a selection should act on the component that has focus, if that component supports the operation.

[ ]

Inserting or pasting elements into a selection, except for a primary transfer operation at the bounds of the primary selection, should first delete the selection if pending delete is enabled. (Pending delete controls the conditions under which the selection is deleted. It is enabled by default.)

[ ]

In normal mode, inserting or pasting elements disjoint from the selection should also deselect the selection, except for primary transfer operations whose source and destination are in the same collection. In add mode, the selection should not be deselected.

In add mode, a transfer operation that is disjoint from the selection should not affect the selection.

[ ]

In editable list–like and graphics–like collections, pressing `Delete` deletes the selected elements.

[ ]

In editable text–like collections, pressing `Delete` or `Backspace` behaves as follows:

If the selection is not empty and the control is in normal mode, the selection is deleted.

If the selection is not empty, the control is in add mode, and the cursor is not disjoint from the selection, the selection is deleted.

If the selection is not empty and the control is in add mode, but the cursor is disjoint from the selection, pressing `Delete` deletes one character forward and pressing `Backspace` deletes one character backward.

If the selection is empty, pressing `Delete` deletes one character forward and pressing `Backspace` deletes one character backward.

Transfer Models

[ ]

If the move, copy, or link operation the user requests is not available, the transfer operation should fail.

Three transfer operations are generally available: copy, move, and link. The user requests one of these operations by pressing the buttons or keys appropriate for the type of transfer. In general, for mouse–based operations, pressing `Ctrl` forces a copy, `Shift` forces a move, and `Ctrl Shift` forces a link. However, any requested transfer operation must fail if that operation is not available.

[ ]

If a collection does not have a fixed insertion point or keeps elements ordered in a specific way, determine the insertion position for transferred data as follows:

For the TRANSFER button–based (or the SELECT button) primary and drag transfer operations, except as noted for text collections, the insertion position is the position at which the user releases the TRANSFER button (or the SELECT button).

In a text–like collection, when the user drops selected text, the insertion position is the position at which the user releases the TRANSFER button (or the SELECT button). When the user drops an icon, the insertion position is the text cursor and the data is pasted before it.

In a list–like collection, the insertion position for other transfer operations is the element with the location cursor and the data is pasted before it.

The insertion position is the position in the destination where transferred data is placed. Some mouse–based transfer operations place data at the pointer position, if possible. Other operations, including keyboard–based transfer, generally place the data at the location cursor.

[ ]

Support the use of MB1 to perform drag−and−drop operations.

[ ]

Pressing MB1 (the SELECT button) performs drag−and−drop operations. A drag can be initiated with either MB1 or MB2. This usage is compatible with other graphical user interface (GUI) environments.

[ ]

When MB2 of a 3−button mouse is configured to operate as the ADJUST button, do not perform any TRANSFER button operations when the user clicks MB2.

[ ]

Allow the user can change an environment setting that indicates that MB2 should be used for the ADJUST button function instead of the TRANSFER button. Clicking the ADJUST button should not result in the transfer of any data.

[ ]

Pressing the SELECT button should always initiate a drag if the drag is started on a selected item. The drag starts once the drag threshold has been reached. This is true for text regions, scrolling lists, and other similar elements.

Clipboard Transfer

[ ]

Make keyboard−based clipboard selection actions available in every editable collection in your application. Clipboard selection actions must be available from the keyboard.

[ ]

Pressing `Cut` (or `Shift Delete`) or selecting the Cut entry on the Edit menu cuts the selected elements from an editable component to the clipboard.

[ ]

Pressing `Copy` (or `Ctrl Insert`) or selecting the Copy entry on the Edit menu copies the selected elements to the clipboard.

[ ]

Pressing `Paste` (or `Shift Insert`) or selecting the Paste entry on the Edit menu pastes the contents of the clipboard into an editable component.

[ ]

If Paste or Paste Link is invoked using a component's pop−up menu, paste the data at the insertion position of the component. However, if the pop−up menu is popped up over a selection, first delete the selection, even if pending delete is disabled, and replace with the pasted data if possible.

Popping up a pop−up menu over a selection indicates that a Paste or Paste Link operation should replace the selection.

[ ]

If Paste or Paste Link is invoked from the Edit menu or by a keyboard operation, and the insertion position in the target component is not disjoint from a selection, the pasted data should replace the selection contents if pending delete is enabled.

Pending delete determines whether the selection is deleted when the insertion position is not disjoint from the selection and Paste or Paste Link is invoked from the Edit menu or by a keyboard operation.

Primary Transfer

[ ]

In an editable collection, clicking the TRANSFER button, `Ctrl` TRANSFER, `Alt Copy`, or `Ctrl Alt Insert` copies the primary selection to the insertion position. (Note that the insertion position is usually different for mouse and keyboard operations.)

[ ]

In an editable collection, clicking `Ctrl Shift` TRANSFER places a link to the primary selection at the insertion position.

[ ]

A primary move should move the primary selection as well as the elements selected; that is, the element moved to the destination becomes selected as the primary selection. Primary copy and primary link should not select transferred data at the destination. This is the expected treatment of the selection in a move, copy, and link operation.

Quick Transfer

[ ]

All text components should support quick transfer.

Quick transfer is used to make a temporary selection and then immediately move, copy, or link that selection to the insertion position of the destination component. In text, quick transfer provides a convenient way to move, copy, or link text without disturbing the primary selection.

[ ]

If a component supports quick transfer, pressing `Alt` TRANSFER (or `Ctrl Alt` TRANSFER) and moving the pointer temporarily selects elements in the specified range and, on release, copies them to the insertion position of the destination component.

[ ]

If a component supports quick transfer, pressing `Alt Shift` TRANSFER and moving the pointer temporarily selects elements in the specified range and, on release, moves them to the insertion position of the destination component.

[ ]

If a component supports quick transfer, pressing `Ctrl Alt Shift` TRANSFER and moving the pointer temporarily selects elements in the specified range and, on release, places a link to them at the insertion position of the destination component.

[ ]

Quick transfer should not disturb the primary selection or affect the clipboard, except when the destination of the transfer is within or on the boundaries of the primary selection and pending delete is enabled. In this case, quick transfer should deletes the contents of the primary selection, leaving an empty primary selection, before pasting the transferred elements.

Quick transfer is a secondary selection mechanism so it cannot disrupt the primary selection. When the destination of the transfer is in the primary selection, quick transfer replaces the primary selection with the secondary selection.

[ ]

With quick transfer, determine the range of the temporary selection by using the same model as when pressing the SELECT button and moving the pointer determines the range of a primary selection.

[ ]

If the user drags the pointer out of a scrollable collection while making the temporary selection, use autoscrolling to scroll the collection in the direction of the pointer. If the user releases the TRANSFER button with the pointer outside of the collection, or if the user presses the `Cancel` key with the TRANSFER button pressed, remove the highlighting and do not perform a transfer.

Drag Transfer

[ ]

In a collection that supports selection, releasing `Shift` TRANSFER or `Shift` SELECT forces a drag move operation. If a move is not possible, the operation should fail.

[ ]

In a collection that supports selection, releasing `Ctrl` TRANSFER or `Shift` SELECT forces a drag copy operation. If a copy is not possible, the operation should fail.

[ ]

In a collection that supports selection, releasing `Ctrl Shift` TRANSFER or `Shift` SELECT forces a drag link operation. If a link is not possible, the operation should fail.

[ ]

When a drag move operation moves a selection within the same component, move the selection along with the elements selected.

In other words, when selected elements are moved with a drag operation, they should stay selected after the move.

[ ]

In text−like collections, when the user initiates a drag within a selected region, drag the entire text selection. Drag−and−drop actions need to operate on the entire selection.

[ ]

In list−like and graphics−like collections, when the user initiates a drag with either the SELECT button or the TRANSFER button on a selected element, drag the entire selection. Drag−and−drop actions need to operate on the entire selection.

[ ]

In list−like and graphics−like collections, when the user initiates a drag with the TRANSFER button or the SELECT button on an unselected element, drag just that element and leave the selection unaffected.

Unselected elements can be dragged without affecting the selection.

[ ]

When the user initiates a drag in an unselected region and the pointer is over two possible draggable elements, use the draggable element highest in the stacking order.

[ ]

At the start of a drag operation, replace the pointer with a drag icon.

[ ]

All drag icons should include a source indicator.

[ ]

Pressing the `Cancel` key ends a drag−and−drop operation by canceling the drag in progress.

[ ]

Releasing the TRANSFER button ends a drag−and−drop operation.

[ ]

When the TRANSFER button (or the SELECT button) is released, the drop operation should ordinarily occurs at the location of the hot spot of the drag icon pointer and in the highest drop zone in the stacking order. However, if a drop occurs within a selection and pending delete is enabled, the transferred data should replace the contents of the entire selection.

[ ]

After a successful transfer, place the data in the drop zone and remove any transfer icon that your application used.

You can use a transfer icon to represent the type of data being transferred during a drop operation. A successful drop operation transfers data.

[ ]

After a failed transfer, keep the data at the drag source. Do not place it in the drop zone. Remove any transfer icon that your application used.

A failed drop operation does not transfer data.

Selection Box (Control)

Behavior

[ ]

When the user selects an item from the list, place the item in the text–entry field.

Navigation

[ ]

Allow the user to navigate within the list with `Page Up` and `Page Down` while the cursor is in the text–entry field.

[ ]

Provide an OK push button to allow the user to accept any changes that have been made in the window and to remove the window.

[ ]

Provide a Cancel push button to allow the user to remove the window without applying any changes that were not previously applied in that window.

[ ]

Provide a Help push button to allow the user to display a window that contains context–sensitive help information.

[ ]

Use either the single or browse selection model in the list box.

Selection Dialog (Secondary Window)

Required

Make the list box and the text−entry field traversable tab groups.

Selection Models

[ ]

If you need to provide other models, or variants of these models (for example, because the elements in the scope are arranged densely, or in hierarchies, or in layers or other 3–D arrangements), do not deviate unnecessarily from the standard models.

[ ]

Use the single or browse selection models when at most one element in the scope can be selected.

[ ]

Use the range selection model when only a single contiguous range of elements can be selected.

[ ]

Use the multiple or extended selection models to allow discontiguous groups of elements to be selected.

[ ]

Use the browse selection model instead of the single selection model when you want to provide feedback to the user regarding the effect of the possible selection of an element.

[ ]

Use the modes and techniques with the various selection models described in Keyboard and Mouse Modes and Selection Models .

Table 1 Keyboard and Mouse Modes

Model	Keyboard Mode	Mouse Mode	Technique
Single	Add	Toggle	Point
Browse	Normal	Select	Point, browse
Multiple	Add	Toggle	Point, area and/or touch
Range	Normal	Select	Point, range
Range	Add	Toggle (only for discontiguous ranges)	Point, range
Extended	Normal	Select	Point, range and/or area, touch (when needed), browse (keyboard–only)
Extended	Add	Toggle (only via `Ctrl`)	Point, range and/or area, touch (when needed)

Table 2 Selection Models

Model	Select	Deselect All?	Contiguous?	Technique(s)	Mode(s)
Single	Max of 1	Y	N/A	Only point	Toggle, add
Browse	Max of 1	N	N/A	Browse and point	Select, normal
Multiple	Any	Y	Y	Point. Touch, range, and/or area may also be used.	Toggle, add
Range	Any	Y	Y	Point, browse, range.	Select, normal (default), add
Discontiguous	Any	Y	Y	Point, browse, at least one of touch, range, and/or area.	Select (default), toggle, normal add

[ ]

Make toggle mode available only in the extended selection model through augmentation with the `Ctrl` modifier key. When `Ctrl` is not used, use select mode for mouse–based selection techniques.

[ ]

When using the extended selection model, allow both normal and add modes, but make normal mode the default mode.

[ ]

Group techniques must be available for the range and extended selection models.

Selection Modes

[ ]

If toggle mode is supported, support add mode as well.

[ ]

If toggle mode is the default mode, support only toggle mode and add mode.

[ ]

If select mode is the default mode, and discontiguous selections are allowed, then holding `Ctrl` down while initiating a mouse−based selection technique forces it to operate in toggle mode.

[ ]

When select mode or normal mode is in use, all elements (if any) identified by a selection technique are selected, and all other elements are deselected.

[ ]

When toggle mode or add mode is in use, all elements (if any) identified by a selection technique have their selection state toggled based on the toggling policy; the selection state of the remaining elements is unaffected. However:

If at most one element can be selected, and the technique selects an element, deselect any other element selected.

If selections are required to be contiguous, the identified elements are selected and all other elements are deselected (in add mode only; toggle mode is not supported).

Keyboard−Based Navigation

[ ]

Use of a navigation key to move the active cursor within a selection scope in normal mode, which results in an element cursor being placed on a selectable element, selects that element and deselects all other elements in the scope.

[ ]

Use of a navigation key to move the active cursor within a selection scope in normal mode, which results in an element cursor being placed on a nonselectable element or on the control (as a whole) that contains the scope, deselects all elements in the scope.

[ ]

Use of a navigation key to move the active cursor within a selection scope in normal mode, which results in a text cursor displayed as the active cursor, deselects all elements in the scope.

[ ]

Use of a navigation key to move the active cursor within a selection scope in normal mode, which results in a graphics cursor displayed as the active cursor, has no effect on the selection state of elements within the scope.

[ ]

Use of a navigation key to move the active cursor within a selection scope in add mode has no effect on the selection state of elements within the scope.

Mouse−Based and Keyboard−Based Selection States

[ ]

For mouse−based operations in select mode, select the identified elements.

[ ]

For mouse−based operations in toggle mode, toggle the selection state of the identified elements.

[ ]

For keyboard−based operations in normal mode, select the identified elements.

[ ]

For keyboard−based operations in add mode, toggle the selection state of the identified elements.

Mouse−Based Navigation

[ ]

Clicking the SELECT button in a selection scope, while using select mode on a nonselectable element that can take focus, moves focus to that element and deselects all elements in the scope.

[ ]

Clicking `Ctrl` SELECT in a selection scope, while using select mode on a nonselectable element that can take focus, moves focus to that element but does not affect the selection state of elements in the scope.

Switching Between Add Mode and Normal Mode

[ ]

When both normal mode and add mode are supported, allow a user to press `Shift F8` to switch between the modes.

[ ]

Switching between normal mode and add mode does not change the default mouse−based mode.

[ ]

If a selection scope is editable, automatically switch from add mode to normal mode when an operation is performed on the selection or when it is deselected.

Visuals

[ ]

When both normal mode and add mode are supported, distinguish visually between the two modes.

Selection Policies

Count Policy

[ ]

If a selection model's count policy allows the user to select any number of elements and the user uses a selection technique in add mode or toggle mode, the selection state of elements outside the region identified by the technique should be unaffected.

[ ]

If a selection model's count policy allows at most one element to be selected, the selection model should not support group techniques.

[ ]

If a selection model's count policy allows at most one element to be selected, a technique that selects an element should deselect any other element that is currently selected.

Deselection Policy

[ ]

Allow the user to deselect all elements with mouse–based selection techniques when toggle mode is in use.

[ ]

Allow the user to deselect all elements with keyboard–based techniques or predefined action choices when add mode is in use.

[ ]

Allow the user to deselect all elements when a text or graphics cursor is in use.

[ ]

If the user is not allowed to deselect all elements, do not support add mode.

Contiguity Policy

[ ]

If a selection model requires contiguity, the only group techniques that should be supported are those that select ranges of elements.

[ ]

If a selection model requires contiguity, it should not support use of toggle mode.

[ ]

Require contiguous selections only in text.

Adjustment Policy

[ ]

Use a balance policy in linearly ordered selection scopes that consist entirely or primarily of text.

Toggling Policy

[ ]

Use an anchor toggling policy in text.

Area Inclusion Policy

[ ]

Use an enclose area inclusion policy.

Anchor Inclusion Policy

[ ]

If the anchor point is on an anchor element (for example, after initiating a point technique or an area click technique on an element), extend the selection region to completely include the anchor element, unless a touch inclusion policy is in use.

[ ]

If an anchor region has been defined (for example, via use of a margin or multilevel selection technique), and a reselect adjustment policy is being used, extend the selection to include the anchor region.

End−Point Inclusion Policy

[ ]

When a point technique or a group click (but not an adjust click) technique is initiated on an element, enlarge the selection region to include that element.

Selection Techniques

[ ]

Provide a point technique to allow the user to identify a point as a basis for a subsequent adjustment technique.

[ ]

Provide a point technique to allow the user to select and/or toggle the selection state of any selectable element, except one that may also be activated or have its value toggled.

[ ]

If you support a group technique in select mode, support click and swipe variants of it.

[ ]

When the user double clicks the SELECT button (optionally with `Ctrl` and/or `Shift` modifiers) on a selectable element and in a scope that does not support multilevel selection, it should first have the same effect as clicking SELECT (with the same modifiers) on that element and then performing the default action.

[ ]

If the user may also activate a selectable element or have its value toggled (for example, a push button or radio button), when the user clicks SELECT on it or presses `Select` (or `Space` or `Ctrl Space`) with the active cursor on it, activate or toggle it instead of selecting it. Such elements may be selected by use of a group technique that includes them in the group of elements to be selected.

[ ]

When the user presses `Cancel` or `Escape`, undo any incomplete selection operation. The application must ignore subsequent key and button releases until the user releases all keys and buttons.

[ ]

When the user presses `Cancel` or `Escape` during a range selection, deselect all elements in the range.

[ ]

When the user presses `Cancel` or `Escape` during a touch selection, leave the selection state of all the elements as they were prior to being touched.

Shortcut Key

[ ]

Provide the predefined shortcut key assignment for each predefined choice.

[ ]

When the user presses a shortcut key, perform the same function that would be performed if the user had activated the associated choice.

[ ]

Perform the function assigned to a shortcut key only if it is assigned within the active window.

[ ]

If the shortcut key choice is within a menu displayed from a menu bar, activate the choice if the user presses the shortcut key and the input focus is in the menu−bar system or the window that contains the menu bar.

[ ]

If the shortcut key choice is on a push button, activate the choice if the user presses the shortcut key and the active cursor is in the window.

[ ]

If the shortcut key choice is within a pop−up menu system, activate the choice if and only if that pop−up menu system is active or the pop −up menu is associated with the active cursor.

Visuals

[ ]

Even if the user turns off the display of shortcut keys in a menu, continue to support the shortcut keys that correspond to the menu items.

Assigning Shortcut Keys

[ ]

Do not use a printing character augmented by `Alt` as a shortcut key, as it would conflict with activation that uses mnemonics.

[ ]

Assign the same shortcut key for a choice in all windows that provide that choice.

Changing Shortcut Key Assignments

[ ]

If the user can change shortcut key assignments, display the new assignments in menus and list the new assignments in any Help information provided for the shortcut keys.

Size (Choice)

[ ]

Include the Size choice on the window menu if and only if the size of the window can be changed.

[ ]

Include the Size choice on the window menu if and only if the size of the window can be changed.

Size Border (Control)

Behavior

[ ]

When the user presses the SELECT or TRANSFER button with the pointer on a corner handle and then moves the pointer, change the height and width of the window relative to the moving pointer without changing the position of the diagonally opposite corner.

[ ]

When the user presses the SELECT or TRANSFER button with the pointer on a top or bottom edge and then moves the pointer, change the height of the window relative to the moving pointer without changing the width or the position of the opposite side.

[ ]

When the user presses the SELECT or TRANSFER button with the pointer on a side edge and then moves the pointer, change the width of the window relative to the moving pointer without changing the height or the position of the opposite side.

Slider (Control)

[ ]

Use a slider to allow a user to set a value from a range of numeric values at evenly spaced intervals. The values may either be absolute, such as temperature or size, or relative, such as a percentage.

[ ]

Use a fixed−size slider arm whose center represents the value. Display the current value as a label adjacent to the slider arm, and allow it to be used as the source of a drag−and−drop transfer operation.

[ ]

Do not use a slider if the user cannot change the value. Use a gauge instead.

Behavior

[ ]

When the user manipulates the slider arm with the mouse, either allow the slider to stop only at positions that represent legal values or assign the legal value to the slider nearest to its position.

[ ]

When the slider has focus, allow directional keys to move the slider arm one interval unit (not one pixel) in the corresponding direction. Allow directional keys augmented by `Ctrl` to move the slider arm multiple units in the indicated direction.

[ ]

If arrow push buttons are used, move the slider arm one interval unit (not one pixel) in the corresponding direction when the user clicks on one.

[ ]

The mouse and keyboard behavior of a slider should be identical to that of a scroll bar except that the interval between legal values is used instead of a one−unit scrolling increment.

Sort (Choice and Dialog)

[ ]

When Sort is a cascading choice, list the various ways in which the current view can be sorted in the cascaded menu.

[ ]

When the user chooses Sort as a dialog choice, display a Sort dialog window that allows a user to specify the way in which the elements in the view are to be sorted.

[ ]

When adding new elements to a view, place them in the proper order with respect to the most recent criteria used for sorting the view.

Spin Box (Control)

[ ]

A spin box should consist of a pair of arrow buttons, plus one or more fields, each of which allow a user to choose among one of a mutually exclusive set of values.

[ ]

Each field in a spin box should be either a text−entry field, a text−display field, or a label, and always takes focus (even when it is a label).

[ ]

The arrow buttons in a spin box should represent action choices whose actions spin some field in the spin box. The arrow buttons do not take focus.

[ ]

The pair of arrow buttons in a spin box should be either horizontal (one points left and one points right) or vertical (one points up and one points down).

[ ]

If a spin box contains multiple fields, make either all of them tab groups or none of them.

[ ]

If a spin box is a tab group, then allow the user to use tab group navigation to move focus to the field in the spin box that last had focus.

Activating Arrow Buttons

[ ]

When an explicit focus policy is in use and focus is on the spin box and the user presses the SELECT button on one of the arrow buttons, change the value of the field in the spin box that has focus.

[ ]

When an explicit focus policy is in use and focus is not in the spin box and the user presses the SELECT button on one of the arrow buttons, change the value of the field in the spin box that last had focus and move focus to it.

[ ]

When an implicit focus policy is in use and the user presses the SELECT button on one of the arrow buttons in the spin box, change the value of the field in which the user last pressed SELECT.

[ ]

When the user presses the SELECT button on a left− or upwards−pointing arrow, spin backwards through values in the designated field; on a right− or downwards−pointing arrow, spin forward through the values.

[ ]

Spinning forward through the legal values of a spin box field works in the following ways:

When the values are ordered chronologically, moving forward moves to the next value chronologically. For example, if TUESDAY is currently displayed, display WEDNESDAY when the user presses the right or down arrow.

When the values are ordered alphabetically, moving forward moves to the next value in lexicographic order. For example, for a field used to pick a font from an alphabetically organized list, if CHICAGO is currently displayed, display COURIER when the user presses the right or down arrow.

When the values represent numbers or magnitudes, moving forward moves to a larger value or magnitude. For example, if 15 is currently displayed, display 16 when the user presses the right or down arrow.

Spin Boxes with Horizontal Arrows

[ ]

If a spin box uses horizontal arrows and it has focus, allow [rarr] and [larr] to spin forward and backward, respectively, through the field that has focus.

[ ]

If a spin box uses horizontal arrows and contains a text−entry field, allow the user to enter and delete characters at the end of that field, but do not allow [larr] and [rarr] to move the cursor through the field.

[ ]

If a spin box uses horizontal arrows, then allow Home and End to spin the field that has focus to its first or last legal value, respectively.

[ ]

If a spin box uses horizontal arrows and is a tab group or contains multiple fields (all of which are tab groups), then [uarr] should have the same effect as [larr] and [darr] should have the same effect as [rarr].

[ ]

If a spin box uses horizontal arrows and contains multiple fields (none of which are tab groups), then allow Ctrl [larr] and Ctrl [rarr] to move focus among the fields, wrapping at the edges.

Spin Boxes with Vertical Arrows

[ ]

If a spin button has vertical arrows and contains a text field, it is a tab group.

[ ]

If a spin box uses vertical arrows and contains multiple fields, then either the spin box is a tab group or all of the fields are tab groups.

[ ]

If a spin box uses vertical arrows, then when focus is in the spin box, allow [darr] and [uarr], respectively, to spin forward and backward through the field that has focus.

[ ]

If a spin box uses vertical arrows, then allow Ctrl Home and Ctrl End to spin the field that has focus to its first or last legal values, respectively.

[ ]

If a spin box uses vertical arrows, contains multiple fields (none of which are tab groups), and focus is on a field that is a label, allow [larr] and [rarr] to move focus to the next field to the left or right, respectively, wrapping at the ends of the spin box.

Spring–Loaded (Control Type)

[ ]

Display only a cascaded control as a result of a user action.

[ ]

When displaying a cascaded control, display its parent control as well.

Stacking Rules

[ ]

A cascaded or pop–up control should not be constrained to its originating window, but participate in the stacking order with other windows, other cascaded and pop–up controls, and other elements stacked with them in the workspace.

[ ]

Always stack a cascaded or pop–up control above its originating window and above its parent control.

[ ]

When a spring–loaded control becomes displayed, stack it above all other controls.

Deactivating a Spring–Loaded System

[ ]

When the user deactivates a spring–loaded system, remove all the spring–loaded controls in that system.

[ ]

When the user releases the SELECT or MENU buttons on a dialog or value choice in a spring–loaded control, activate or toggle the choice.

[ ]

When the user activates a dialog or action choice or toggles a value choice in a spring–loaded control, deactivate the associated spring –loaded system.

[ ]

When the user releases the SELECT or MENU button in the background or title of a spring–loaded control, either leave the control displayed or deactivate its associated spring–loaded system.

[ ]

When the user releases the SELECT or MENU button over an unavailable choice in a spring–loaded control, either leave the control displayed or deactivate its associated spring–loaded system.

[ ]

When the user releases the SELECT or MENU button outside of a spring–loaded system, deactivate it.

[ ]

When the user presses a mouse button outside a spring–loaded system, deactivate it and do one of the following:

Allow the mouse press to pass to the element under the pointer and normally direct all subsequent device events (recommended).

Discard all device events between the mouse press and the mouse release. These events should have no effect except to optionally raise and/or move focus to the window over which the mouse button was pressed.

Focus

[ ]

If displaying a spring−loaded control, bring focus to a control in its spring−loaded system.

[ ]

If an explicit focus policy is in use, deactivating a spring−loaded system should have one of the following effects on the focus:

If the base control of the spring−loaded system consists solely of cascading choices, move focus, if possible, to the control in the originating window that last had focus before focus moved into the spring−loaded system.

Move focus to the base control.

If the focus was moved to a different window during deactivation, then give focus to the base control of that window, if possible.

Cancel Behavior

[ ]

When focus is on a cascaded control and the user presses `Cancel` or `Escape`, remove the control.

Placement

[ ]

Place a spring−loaded control so that it fits entirely on the screen. If it is larger, in any dimension, than the screen, fit as much of it on the screen as possible.

[ ]

Do not place a cascaded spring−loaded control in such a way that it completely covers its associated cascading choice.

Spring Sensitive (Mode)

[ ]

When a spring−loaded control becomes spring sensitive, make its parent also spring sensitive if it can be (for example, if it is spring −loaded, a tear−off menu, or a menu bar).

Focus

[ ]

When a control is spring sensitive, have the active cursor track the pointer when it is within the control (for example, as if an implicit focus policy were in force within the control).

[ ]

If a control may be spring sensitive, but is not currently spring sensitive, then do not have the active cursor track the pointer when it is within the control (for example, as if an explicit focus policy were in force within the control).

Pressing a Mouse Button

[ ]

When the user presses (but does not click) a mouse button to display a spring−loaded control, make it spring sensitive.

[ ]

When the user presses the SELECT button on a spring−loaded control, make the control spring sensitive.

[ ]

When the user presses the SELECT button on a menu, make the menu spring sensitive.

[ ]

When the user presses the MENU button on a pop−up menu or any descendant menu, make that menu spring sensitive.

Motion

[ ]

If a cascaded control is spring sensitive and the user moves or places the pointer to the cascaded choice associated with it, then the cascaded control should remain displayed and spring sensitive.

[ ]

If a cascaded control is displayed and the user moves or places the pointer on an ancestor spring−sensitive control other than on the cascading choice associated with the cascaded control, then remove the cascaded control.

[ ]

A spring−sensitive control in a spring−loaded system should remain displayed while the user moves the pointer outside the system with a mouse button pressed.

Releasing a Mouse Button

[ ]

When the user releases the SELECT button over a noncascading choice in a spring−sensitive control, activate or toggle that choice.

Stacking Order

[ ]

Provide automatic stacking order to allow the user to have windows moved to the top of the stacking order when they become active.

[ ]

Make automatic stacking order the default stacking order.

[ ]

Allow the user to choose which stacking order to use.

[ ]

When multiple windows are stacked on the screen and a window in the automatic stacking order gains focus, raise the window and all of its associated windows to the top of the window stack.

[ ]

When using automatic stacking order, place a window on top of all other windows and make it the active window when the user does the following:

- Selects an object in the window
- Selects the title bar or window border of that window
- Uses keyboard navigation to warp the pointer to the window
- Invokes a new window

[ ]

When using manual stacking order, place a window on top of all other windows and make it the active window when the user does the following:

- Selects the title bar or window border of that window
- Explicitly raises the window by an internal keyboard or mouse button binding
- Invokes a new window

Static Text

[ ]

Do not support navigation to a static text element. However, allow the operating environment to specify that large static text can be navigated to so that data transfer can be supported.

Tab Group

[ ]

Configure a group of related push buttons, radio buttons, or check boxes together as a tab group.

Tab Group Order

[ ]

All controls within a window that can take focus, other than the menu bar, should be in one of the tab groups of the window.

[ ]

If a control within a scrolled area is in a tab group, either put the entire tab group within the scrolled area or put the entire scrolled area within the tab group.

Tab Group Navigation

[ ]

When an explicit focus policy is in effect and the user presses `Ctrl Tab` in a window with focus, move the focus to the next tab group in the window.

[ ]

When an explicit focus policy is in effect and the user presses `Tab` in a window with focus (and does not use `Tab` internally), move the focus to the next tab group in the window.

[ ]

When an explicit focus policy is in effect and the user presses `Ctrl Shift Tab` in a window with focus, move the focus to the previous tab group in the window.

[ ]

When an explicit focus policy is in effect and the user presses `Shift Tab` in a window with focus (and does not use `Tab` or `Shift Tab` internally), move the focus to the previous tab group in the window.

[ ]

Allow tab group navigation to wrap between the first and last tab groups of the window.

Tear−Off Menu (Menu Type)

Tear−Off Choice

[ ]

Make a tear−off choice the first choice in a menu.

[ ]

When a menu is torn off, the resulting tear−off menu should contain the same choices, in the same order and with the same availability, as the pull−down menu.

[ ]

A torn−off menu should not contain the tear−off choice from which it was created.

Tear−Off Menu Semantics

[ ]

When the user tears off a pull−down menu, or a menu cascaded from one, ensure that the presence, order, availability, and function of the items in the tear−off menu remain identical to those in the corresponding menu.

[ ]

When the user tears off a menu popped up over a multiple−element selection (or a menu cascaded from one), ensure that the presence, order, availability, and function of the menu items remain consistent with whatever is currently selected in the same scope. Also, ensure that they remain identical to the corresponding menu newly popped up over (or cascaded from) a multiple−element selection.

[ ]

When the user tears off a menu popped up on an object that is unselected, or is the only one selected, ensure that the presence, order, availability, and function of the menu items remain consistent with that object. Also, ensure that they remain identical to the corresponding menu newly popped up over (or cascaded from) the object if it were unselected or the only one selected.

[ ]

When the user tears off a menu popped up in the background (or a menu cascaded from one), ensure that the presence, order, availability, and function of the menu items remain consistent with the state of the selection scope as a whole. Also, ensure that they remain identical to the corresponding menu newly popped up over (or cascaded from) the background.

[ ]

When the user tears off a menu popped up on an element that is not selectable or that does not include a selection scope (or a menu cascaded from one), ensure that the presence, availability, and function of the menu items remain consistent with the state of the element. Also, ensure that they remain identical to the corresponding menu newly popped up over (or cascaded from) the element.

Corresponding Tear−Offs

[ ]

Allow a user to display and interact in the usual way with a menu, even when it corresponds to a currently displayed tear−off menu.

[ ]

The displaying of a menu should have no effect on the display of its corresponding tear−off menu.

[ ]

When a menu is torn off and a corresponding torn−off menu already exists, remove the existing torn−off menu.

Tearing Off to a Secondary Window

[ ]

Always display a tear−off menu in an ordinary modeless secondary window whose parent is the window with the tear−off choice.

[ ]

When the user activates a tear−off choice in a menu, tear off the menu, deactivate the original menu's menu system, and display the resulting tear−off menu in a secondary window at or near the location of the original menu.

[ ]

When the user drags a tear−off choice in a menu while holding the TRANSFER button, tear off the menu and deactivate its menu system. When the drag is completed, place the menu in a secondary window so that its title is at the location of the drop.

[ ]

When the user tears off a menu, move the input focus to the tear−off menu, if an explicit focus policy is in use at the workspace level.

Removal

[ ]

Allow the user to remove a tear−off menu's window by providing the Close choice in its window menu.

[ ]

When focus is on a torn−off menu and the user presses `Cancel` or `Escape`, remove the torn−off menu.

Focus−Based Removal and Display

[ ]

Redisplay a tear−off menu when the user moves focus back to a window that had an applicable tear−off menu that was automatically removed.

Text−Display Field (Control)

[ ]

Allow the user to select the text in a text−display field.

[ ]

Allow the user to scroll the text in a text−display field when more information is available than can be displayed at any one time.

Text−Entry Field (Control)

[ ]

Support insert mode for all entry of text.

[ ]

Do not support replace mode when nontextual elements (for example, icons or choices) are included as elements in text and laid out as characters.

Verification of Entered Text

[ ]

Even if the information entered by the user into a text−entry field is not acceptable to the application and if the focus has been moved out of the text−entry field, do not automatically move focus back to the text−entry field.

Insert Mode

[ ]

In insert mode, position the text cursor between adjacent elements.

[ ]

In insert mode, when the user types a character, insert the character before the cursor.

Replace Mode

[ ]

In replace mode, display the text cursor as if it were on a character. However, with respect to the rules for navigation and selection, treat it as if it were positioned prior to the character on which it is displayed. When the cursor is at the end of a line, it should appear to be on an empty character following the last character on the line.

[ ]

In replace mode, when the cursor is not at the end of a line and the user types a printing character, delete the character on which the cursor appears before inserting the new character. This should have the effect of replacing the character.

[ ]

In replace mode, when the cursor is at the end of the line and the user types a printing character, insert the character before the cursor position.

Behavior

[ ]

When a control supports text entry, support primary, clipboard, and drag−and−drop transfer in the text field.

[ ]

Allow the user to scroll data in a text−entry field when more text can be typed or when more information is available than can be displayed in the field.

[ ]

In text, when the user presses `Space` or `Shift Space`, insert a space.

[ ]

In a text−display field or a single−line text−entry field, when the user presses `Enter`, invoke the default action.

[ ]

In a multiline text−entry field, when the user presses `Enter`, insert a new line. End the current line with the character preceding the current cursor position and start a new line with the following character.

[ ]

In a text−display field, use `Tab` and `Shift Tab` for tab−group navigation.

[ ]

In a multiline text−entry field, use `Tab` and `Shift Tab` for internal navigation. Use `Tab` either to insert a tab or to skip to the next tab stop. `Shift Tab` should have either no effect or should skip back to the previous tab stop.

[ ]

In a single−line text−entry field, use `Tab` and `Shift Tab` either for tab−group navigation or for internal navigation as in a multiline text −entry field.

Text Field (Abstract Control)

[ ]

Provide a text field when you display a sequence of characters that the user can manipulate as a unit or in which you want to allow the user to navigate and manipulate at least one subsequence of characters.

[ ]

Organize elements of a text field into complete words as the first level of a multilevel hierarchy (for example, double−clicking SELECT on a word should select the word). The boundary of a word is defined by a language's characters in the text field.

Selection

[ ]

Support point and range selection techniques and multilevel selection techniques for text.

[ ]

If supporting area techniques, provide a choice that allows the user to switch between use of range and area techniques.

[ ]

When the user presses `Ctrl Space` in a text field, it should have the same effect as a keyboard−based point selection technique.

[ ]

When the user presses `Ctrl Shift Space` in a text field, it should have the same effect as a keyboard−based range adjust click selection technique.

[ ]

If word−wrapping is used and as a result multiple lines are displayed in a text field, treat each visual line as a separate line for navigation operations.

Behavior

[ ]

The directional keys should operate in the following manner:

[uarr]

In a multiline text field, move the cursor up one visually displayed line.

[darr]

In a multiline text field, move the cursor down one visually displayed line.

[larr]

Move the cursor left one character.

[rarr]

Move the cursor right one character.

[ ]

The directional keys augmented by `Ctrl` should operate as follows:

`Ctrl [larr ]`

In a text field with multiple words, move the cursor to the beginning of the word to the left.

`Ctrl [rarr ]`

In a text field with multiple words, move the cursor to the beginning of the word to the right.

[ ]

When the user presses a directional key to move the cursor horizontally, wrap the text cursor from the end of one row to the beginning of the next row (and vice versa). Do not, however, wrap between the beginning and the end of the text field.

[ ]

The navigational keys should operate as follows:

`Home` (or `Begin`)

Place the cursor at the beginning of the current line.

`END`

Place the cursor at the end of the current line.

`Ctrl Home` (or `Ctrl Begin`)

Place the cursor before the first character.

`Ctrl End`

Place the cursor after the last character.

Treating Nontextual Elements as Characters

[ ]

When nontextual elements that have no internal navigation (such as icons or choices) are included as elements in text and laid out as characters, navigation should perform in the following ways:

When the user is navigating through text, a text cursor should always stop in between adjacent characters, between a character and a nontextual element, and between two adjacent nontextual elements.

Use an element cursor to indicate that focus is on a nontextual element.

When a text cursor is immediately to the left or right of a nontextual element, allow `[rarr ]` and `[larr ]` to remove the text cursor and move focus to the nontextual element.

When an element cursor is on a nontextual element, allow `[rarr ]` and `[larr ]` to remove focus from the nontextual element and place the text cursor to the right or left of the element.

Treat nontextual elements as words with regard to multilevel navigation. That is, if the cursor is on the nontextual element or immediately before it and the user presses `Shift [rarr ]`, move the text cursor to the beginning of the word (or nontextual element) following it. If the cursor is on the nontextual element or immediately after it and the user presses `Shift [larr ]`, move the text cursor immediately before it.

[ ]

Treat nontextual elements included with characters in text either as selectable or nonselectable consistently within the scope. If they are treated as nonselectable, exclude them from selections that physically contain them.

Text with Active Regions

[ ]

When active regions (regions of text that act like action choices) are allowed in a text field, use the following rules:

Active regions must be highlighted to indicate their boundaries and must not overlap.

When the user clicks the SELECT button within an active region, invoke the action associated with that region and move the text cursor to the pointer location.

When the user presses the SELECT button within an active region without clicking (including releasing SELECT past the threshold

time), select the active region.
When the user clicks `Ctrl` SELECT within an active region, move the text cursor to the location clicked, but do not invoke the action associated with the region.
When the text cursor is within an active region, the default action should be invoking the action that corresponds to the region.
If the user can edit the text field, all characters typed have their usual effect regardless of whether or not the text cursor is within an active region. For example, when the user presses `Enter` in an editable multiline text control, insert a new line in the active region.
When the user presses `Ctrl Enter`, invoke the default action, which invokes the action that corresponds to the region.

Tool (Value Choice Type)

[ ]

Include tools in a value set to perform specific tasks in an application.

[ ]

When the user selects a tool, do the following:

Change or augment the pointer to indicate which operation the user can perform. For example, if the user chooses a line−drawing tool in a value set, change the pointer to a line−drawing tool pointer in the applications viewing area.

Provide a tool within the value set to allow the user to return to a neutral state, unless completing the activated choice automatically returns the user to a neutral state. For example, if the user chooses a line−drawing tool in a tool palette area and the pointer is not restored to the arrow pointer after the line−drawing operation is complete, provide an arrow−pointer choice on the palette area.

Display the pointer appropriate for the tool mode when the pointer is within the area to which the tool applies.

[ ]

When focus is on a tool, the default action, if any, is an expert action; that is, one which is equivalent to using the tool in some special way. For example, make the default action (such as, expert) for an eraser tool erase the entire contents of the control to which the tool applies.

Touch Adjust Click Technique

[ ]

When the user presses the ADJUST button or presses `Shift Select`, `Shift Space` (unless in text), or `Ctrl Shift Space`, add the adjustment element to the current selection region.

[ ]

If the current selection region did not previously define an anchor point, define it to be the adjustment element.

[ ]

After the user uses a mouse−based touch adjust click technique, place the active cursor on one of the following:

If it is a graphics cursor, at the point at which the ADJUST button was clicked

On the adjustment element

Touch Adjust Swipe Technique

[ ]

After a mouse−based touch swipe technique is used, place the active cursor on one of the following:

- If it is a graphics cursor, at the point at which the ADJUST button is released
- If one or more elements are touched, on the last element touched
- On the currently cursored element in the selection scope

[ ]

After a mouse−based touch adjust swipe technique is used:

- Add all the elements touched to the current selection region.
- If one or more elements are touched, identify the anchor element to be the first element touched.

[ ]

If the current selection region did not include an anchor point, define it to be the first element touched during use of the touch adjust swipe technique.

[ ]

After a mouse−based touch adjust swipe technique is used, place the active cursor on one of the following:

- If it is a graphics cursor, at the point at which the ADJUST button was released
- If one or more elements were touched, on the last element touched
- On the currently cursored element in the selection scope

Touch Swipe Technique

[ ]

Support use of the touch swipe technique only when discontiguous selections are allowed.

[ ]

After a mouse−based touch swipe technique is used:
- Define the current selection region to consist of the elements touched.
- If one or more elements are touched, identify the anchor element to be the first element touched.

[ ]

After a mouse−based touch swipe technique is used, place the active cursor on one of the following:
- If it is a graphics cursor, at the point at which the SELECT button is released
- If one or more elements are touched, on the last element touched
- On the currently cursored element in the selection scope

[ ]

When the user is using the touch swipe technique in select mode:
- Select all the elements touched.
- Deselect all other elements in the scope.

[ ]

If currently in select mode, when the user operates the area swipe technique by pressing `Ctrl` SELECT in place of SELECT, continue as if toggle mode were in use.

[ ]

When the user is using the area swipe technique in toggle mode:
- Toggle all elements touched, based on the toggling policy.
- Do not change the selection state of the other elements.

Undo, Redo, Repeat (Action Choices)

[ ]

Provide the Undo, Redo, and Repeat menu items for each editable element.

[ ]

Include the Undo, Redo, and Repeat choices in the Edit menu or an object's pop−up menu.

[ ]

The Undo choice should undo the last action performed by the user. If only single−level Undo is supported, then after the user chooses Undo, make it unavailable.

[ ]

Replace the Repeat choice with the Redo choice after the user chooses Undo.

[ ]

Replace the Redo choice with the Repeat choice after one of the following:

- A new action is performed
- All actions undone have been redone

Value (Choice Type)

[ ]

Do not change the text label of a value choice when it is set or unset.

Visuals

[ ]

Group related value choices together. If value choices are placed in a menu, use separators to separate groups of related choices.

[ ]

In a control that displays value choices, allow the user to clearly see whether the value choice is set or unset.

[ ]

In a menu other than an option menu, display a graphic next to a value choice's label to indicate its setting.

[ ]

If a value choice does not use a separate graphic to indicate its setting, change the contents (only if it is graphic) or visual aspects of the label to indicate whether it is set or not.

Value Set (Control)

[ ]

Use a value set to allow the user to choose a tool.

[ ]

The value choices in a value set should be mutually exclusive; setting one unsets the choice in the value set that is currently set.

View

[ ]

Display a view in the viewing area of a window.

[ ]

If the application makes changes to one view of user data, simultaneously display the changes in all other views of the same data.

Viewing State

[ ]

When the user opens a minimized window, ensure that the viewing state of all the viewing areas in the window (including the view shown in each, and the cursor and scrolling position within each view) is the same as when the window was last minimized, unless the user or application has made other changes that prevent this.

[ ]

When the user ends a login session (or terminates the window stem) and then later restarts it, any application restarted should display the same views in the same viewing state as when the session ended, unless prevented by changes in underlying persistent data or by user directives.

[ ]

When the user removes all instances of a view (for example, by changing views, combining split views, or closing windows) and then later redisplays the same view in the same instance of the application, display the view with the same viewing state it had when it was removed, unless the user or application has made other changes that prevent this.

View Menu

[ ]

Provide a View choice on the menu bar of a window when more than one view is available for the information displayed in the window.

[ ]

If you provide additional application−specific choices in the View menu, group them with other related choices.

[ ]

When the user selects choices in the View menu, do not change the underlying data, only the view of that data.

Viewing Area (Control)

[ ]

Provide a viewing area when a window is used to display user−provided or specified application−interpreted information. For example, text from a text file would be shown in the viewing area of a word−processing application.

[ ]

If a window contains multiple viewing areas, and the user makes a choice that can apply to more than one view, apply it to the area that has focus. If no area has focus, then apply the choice to the area that has interacted emphasis. Otherwise, issue a warning signal.

[ ]

Provide at least one viewing area in every primary window.

[ ]

Place the viewing areas below the status area and menu bar and above the information area and command area.

[ ]

Support both horizontal and vertical scrolling in each viewing area.

Warning Signal

[ ]

Provide a warning signal when the user types a character that is not supported as input for a text−entry field that has focus. For example, if a text−entry field supports only numeric characters, provide a warning signal when the user types a letter.

[ ]

Provide a warning signal when the user types a character that cannot be used as a mnemonic at the active cursor position or the control on which the cursor is positioned does not accept text input. For example, if mnemonics are supported but no choice has L assigned as the mnemonic, provide a warning signal when the user presses the `L` key.

[ ]

Use a warning signal to supplement a message. Note that in degraded environments, a message may occur in sensory modalities other than the standard visual on−screen mode. See Chapter 12 for more information.

[ ]

If the user has turned off a warning signal option for the operating environment and a situation arises in which the system would normally generate that warning signal, do not generate a warning signal. For example, if the user has turned off audible signals for the operating environment, do not generate an audible signal when an action message is displayed.

Window (Menu)

[ ]

Use the Window menu when your application can edit more than one file or object at the same time.

[ ]

In conjunction with the **Menu Guidelines** reference page, use Window Menu (Menu−Bar) Choices to decide which of the choices listed to include in the Window menu, and how to organize them.

Table 1 Window Menu (Menu−Bar) Choices

Mnemonic	Menu Choice
N	New Window
	Application instance

[ ]

Remove an application instance choice from the Window menu when the user closes it.

[ ]

The application instance choice should be the same name as the file name or object name.

[ ]

When the user chooses New Window from the Window menu, create a duplicate window with the current file or object, if possible.

Window Frame

[ ]

Provide a window frame and title bar in all windows, except those that are part of the workspace.

[ ]

Do not place any window control outside the boundary formed by the sizing border.

Behavior

[ ]

The window menu button should represent a cascading choice that leads to the display of the window menu.

[ ]

Support moving a window when the user presses either the SELECT or TRANSFER button on the title bar and moves the mouse.

Layout

[ ]

Align the bottom of the title bar with the top of the client area.

[ ]

Place the window title in the title bar.

[ ]

If you provide a maximize and a restore button (only one of which is displayed at a time), align the button's right border with the right border of the client area.

[ ]

Place the right border of the minimize button next to the left border of the maximize or restore button.

[ ]

Align the left border of the window menu button with the left border of the client area.

[ ]

If you add more buttons for window manager functions on the window frame, place the buttons directly to the left of the minimize button or directly to the right of the window menu button. Design each button to correspond to a choice in the window menu.

Window Icon

[ ]

Display a window icon for the corresponding window when the window is minimized.

[ ]

When a window icon is placed in a window icon box, continue to display the window icon even when the corresponding window is displayed.

[ ]

Show in−use emphasis on a window icon when its corresponding window is displayed.

[ ]

When a window icon is placed on the workspace, remove it when the corresponding window is displayed.

[ ]

If a window icon's corresponding window is not displayed, display the Minimize choice in the window menu with unavailable emphasis, or do not provide the Minimize choice.

[ ]

Allow the user to change the position of all window icons.

Visuals and Layout

[ ]

If a window is used to view an object, include in its window icon a representation of the type of object being viewed.

Window Icon Box

[ ]

Provide a window icon box for the user to store window icons.

[ ]

Make the window icon box a primary window whose viewing area includes a container for window icons, scrollable if necessary.

[ ]

Do not support the Close choice for the window icon box.

[ ]

Design the window icon of a window placed in the window icon box to be the same as the window icon would be if it were on the workspace.

Arranging Icons

[ ]

Include a Pack Icons choice in the window menu of the icon box.

[ ]

Make the Pack Icons choice the default menu item of the window icon box's window menu.

[ ]

The Pack Icons choice should have the icons arranged in the icon x as closely as possible in the visible area.

Window Management (CDE)

Window Support

[ ]

Application windows should be clearly distinguishable as primary or secondary windows, based on appearance and behavior.

Primary windows include:

- Primary window decoration
- Primary window management
- Window stacking, workspace placement, and minimization, which can be independent of other primary windows

Secondary windows include:

- Secondary window decoration
- Secondary window management
- Window stacking, workspace placement, and minimization tied to the associated primary window

Window Decorations

[ ]

Windows that support particular window management functionality must request the corresponding window decoration (for example, a window that can be minimized should request the minimize button).

[ ]

Windows that support any window management functionality (move, resize, minimize, maximize, close, and others) must have a window menu with items for that functionality.

[ ]

Use the CDE window decoration conventions, as shown in CDE Window Decoration Conventions .

Table 1 CDE Window Decoration Conventions

	Border	Title	Menu	Min	Max	Resize
Primary window:						
Default	Yes	Yes	Yes	Yes	Yes Decorations for resize and maximize should be provided for primary windows if appropriate.	Yes
Front Panel	Yes The Front Panel has custom visuals for the window decorations.	No	Yes	Yes	No	No
Secondary window:						
Default	Yes	Yes	Yes	No	No Secondary windows should be designed such that resizing and maximizing are not necessary or appropriate. If a secondary window must be resizable and mazimizable, the associated	No

					decorations should be displayed.	
Front Panel	No	Yes	Yes	No	No	No

[ ]

Follow the CDE window menu conventions. The following items appear in the window menu if they are applicable to the window or its minimized window icon:

- Restore (R)
- Move (M)
- Size (S)
- Minimize (n)
- Maximize (x)
- Lower (L)
- Occupy Workspace ... (O)
- Occupy All Workspaces (A)
- Unoccupy Workspace (U)
- Close (C) `Alt F4`

Window Management Actions

[ ]

Windows should follow CDE window management functionality conventions, as shown in CDE Window Management Conventions .

Table 2 CDE Window Management Conventions

	Close	**Move**	**Lower**	**Min**	**Max**	**Resize**
Primary window:						
Default	Yes	Yes	Yes	Yes	Yes Resize and maximize functionality should be provided for primary windows if appropriate.	Yes
Front Panel	No	Yes	Yes	Yes	No	No
Secondary window:						
Default	Yes	Yes	Yes	No	No Secondary windows can contain the Maximum and Resize window manager functions, if appropriate.	No
Front Panel	Yes	Yes	Yes	No	No	No

[ ]

Windows that support particular window management functionality should request corresponding window decoration (for example, a window that can be minimized should request the minimize button).

[ ]

Windows that have form−factor constraints should set Window Manager hints for minimum size, maximum size, aspect ratio, and resize increment, as appropriate.

[ ]

Windows that have Close or Exit functionality should support the window management protocol for Close if there is a window menu. In the case of dialog boxes, the Close item on the window menu should correspond to the Cancel functionality or dialog box dismissal with no further action taken.

Session Management Support

[ ]

Support Interclient Communications Conventions Manual (ICCCM) mechanisms for session management of primary windows and key properties.

[ ]

Support ICCCM mechanisms for session management of all associated windows (that is, secondary windows that may include help windows).

Window Menu

[ ]

Place all the choices that allow a user to change the properties of a window, such as its size and position, in the window menu.

[ ]

Provide a window menu for each modeless window unless configured by the user or the application.

[ ]

In conjunction with the **Menu Guidelines** reference page, use Window Menu Choices to decide which of the choices listed to include in the window menu, and how to organize them.

Table 1 Window Menu Choices

Mnemonic	Menu Choice	Keyboard Function
R	Restore	
M	Move	
S	Size	
N	Minimize	
X	Maximize	
C	Close	`Alt F4`

[ ]

If you provide a default menu item for the window menu, make it the Close choice, except for the window icon box's window menu.

[ ]

Display the window menu for a window or for its corresponding window icon when it has input focus and the user presses `Shift Esc` or `AltSpace`. When the user presses these keys, remove the window menu if it is displayed.

[ ]

Make the window menu the pop–up menu of the window icon.

Specific Menu Choices

[ ]

When the user chooses the Refresh choice, update the window or screen to reflect the current state of the underlying data.

[ ]

When the user chooses the Lower choice in a primary window, lower that window to the bottom of the stacking order. When the user chooses the Lower choice in a secondary window, lower the secondary window and it's primary window, when appropriate.

Window Navigation

[ ]

Support window navigation when an explicit focus policy is in effect at the workspace level, that is, at the topmost level of windows on the screen.

Behavior

[ ]

When the user presses `Alt Esc`, move the input focus forward between window families.

[ ]

When the user presses `Alt Shift Esc`, move the input focus backward between window families.

[ ]

When the user presses `Alt F6`, move the input focus forward between windows within a window family.

[ ]

When the user presses `Alt Shift F6`, move the input focus backward between windows within a window family.

[ ]

If there is no window icon box, assign `Alt Esc` and `Alt Shift Esc` to move the input focus among icons as well as windows. Window icons that are not in a window icon box should be treated as a window family consisting of a single window.

Window Title

[ ]

Provide a title for every window.

[ ]

Use a short description of the contents of the window as a title.

[ ]

Place the window title in the title bar of the window frame.

[ ]

If a user sizes a window so that all of the title cannot be displayed in the title bar, clip the window title from the end.

Modifying the Window Title

[ ]

Do not change the name of an object being viewed in the window as a result of the user modifying the window title.

Naming Conventions for Primary Windows

[ ]

Include the name of the application in a primary window's window title if any of the following are true:

- The application does not have a state or contents that needs to be named and persistently stored
- The window is used to view a single file or object and is not the default application used to view that file or object
- The window is used to view multiple files or objects

[ ]

If an application is used to view a single file or object, include the name of the file or object in the primary window's window title, following the name of the application, if necessary.

[ ]

When different primary windows of the same application are used for different views or purposes, append the name of the view or purpose to the window title.

[ ]

For a context−sensitive help window, append the Help choice to the title of the window for which help is being requested.

Naming Conventions for Secondary Windows

[ ]

Include in the title of a secondary window (other than one holding a torn−off control) the title (or a portion of the title likely to be unique) of the window it is dependent upon.

[ ]

Include in the title of a secondary window (other than one holding a torn−off control), one or more of the following:

- The type or purpose of the window (such as dialog or property)
- The name of the action, task, or situation that caused the window to be displayed

Workspace

[ ]

If a menu bar is provided for the workspace, allow the user to turn it on and off through a pop–up menu for the workspace.

[ ]

By default, do not display the menu bar, if one is provided for the workspace. Allow the user to specify the default for the operating environment.

Behavior

[ ]

Assign `AltF10` and `Alt Menu` as the shortcut keys to invoke the pop–up menu for the workspace.

[ ]

If your workspace supports toggling between the current user configuration and the default configuration, assign `Alt Shift Ctrl!` as the shortcut that initiates a configuration toggle.

[ ]

When the user chooses Refresh from the workspace pop–up menu, redraw the contents of the workspace and all windows in it, repairing any visual glitches that might have inadvertently occurred.