Microsoft® Windows

Software Development Kit

Application Style Guide

Version 2.0

Microsoft Corporation

Information in this document is subject to change without notice and does not represent a commitment on the part of Microsoft Corporation. The software described in this document is furnished under a license agreement or nondisclosure agreement. The software may be used or copied only in accordance with the terms of the agreement. The purchaser may make one copy of the software for backup purposes. No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose other than the purchaser's personal use without the written permission of Microsoft Corporation.

© Copyright Microsoft Corporation, 1987. All rights reserved. Simultaneously published in the U.S. and Canada.

Microsoft®, the Microsoft logo, and MS-DOS® are registered trademarks of Microsoft Corporation.

Document No. 050051016-200-I01-1087

Contents

Introduction 1

1 Main Window 3

1.1 Introduction 5
1.2 Icons 6
1.3 Window Captions 6
1.4 Output 7
1.5 Scrolling 8
1.6 Split Windows 10

2 Menus 13

2.1 Introduction 15
2.2 Menu Names 15
2.3 Menu-Item Names 16
2.4 Menu-Item Types 16
2.5 Disabled Menu Items 18
2.6 The Control Menu 19
2.7 The File Menu 20
2.8 The Edit Menu 23
2.9 The Exit Command 24
2.10 The About... Command 24

3 Accelerators 27

3.1 Introduction 29
3.2 Accelerator Rules 29
3.3 Accelerator Names in Menus 29
3.4 Standard Control-Menu Accelerators 30
3.5 Standard Edit-Menu Accelerators 30
3.6 The Help Key 31

4 Selection 33

4.1 Introduction 35
4.2 Selection Highlighting 35
4.3 Text Selection 36
4.4 Editing Text Selections 37
4.5 Non-text Selection 38
4.6 Discontinuous Selection 39

5 Keyboard Interface 41

5.1 Introduction 43
5.2 General Recommendations 43
5.3 Reserved System Keys 44
5.4 Recommended Keyboard Use 46
5.5 Keys Used in Child Windows 49
5.6 Keyboard Input Focus 50

6 Mouse Interface 51

6.1 Introduction 53
6.2 Recommended Mouse Use 53
6.3 Mouse Pointers 55
6.4 Mouse Capture 55

7 Filenames 57

7.1 Introduction 59
7.2 Default Filename Extension 59
7.3 Filename Conventions 59
7.4 Temporary Filenames 60
7.5 Prompting for Directories and Filenames 61

8 Dialog Boxes 63

8.1 Introduction 65
8.2 Dialog-Box Types 65
8.3 Dialog-Box Appearance 66
8.4 Controls 66
8.5 Default Push Buttons 68
8.6 Required Buttons 69
8.7 Dialog Boxes with Multiple Controls 69
8.8 Dialog Boxes Without Push Buttons 70
8.9 Sizing Dialog Boxes 70
8.10 Moving Between Controls 70
8.11 Dialog-Box Appearance 71

9 Message Boxes 73

9.1 Introduction 75
9.2 Message-Box Types 75
9.3 The System Beep 76
9.4 Message-Box Captions 76
9.5 Push Buttons in Message Boxes 76
9.6 Keyboard Use 77

9.7 Flashing for Attention 77
9.8 Sample Message Boxes 78
9.9 Message-Box Text 79

10 Cooperation 81

10.1 Introduction 83
10.2 Loading and Initialization 83
10.3 Yielding and Time-Critical Operations 83
10.4 The System Hourglass 84
10.5 Dead Air 84
10.6 Memory Use 84
10.7 File Use 85
10.8 Color Use 85

11 International Concerns 87

11.1 Introduction 89
11.2 Resource Files 89
11.3 Character Sets and Keyboards 89
11.4 Formats 90
11.5 Additional Software Design Tips 91

12 Help Facility 93

12.1 Introduction 95
12.2 Access to Help 95
12.3 General Help 96
12.4 Context-Sensitive Help 97

13 Multiple-Document Interface 99

13.1 Introduction 101
13.2 The Main Window 101
13.3 The Child Window 103
13.4 Moving Within the Interface 106

Figures

Figure 1.1 Main Window 5
Figure 1.2 Clock Icon 6
Figure 1.3 Window Caption with Filename 7
Figure 1.4 Output in Windows 8
Figure 1.5 Scroll Boxes and Scroll Arrows 9
Figure 1.6 Split Window with Two Child Windows 10
Figure 2.1 Menu Names 15
Figure 2.2 Items in a Menu 16
Figure 2.3 Types of Menu Items 17
Figure 2.4 Mutually Exclusive Toggle Commands 18
Figure 2.5 Grayed Menu Items 19
Figure 2.6 Control Menu 19
Figure 2.7 File Menu 21
Figure 2.8 Dialog Box with Warning 21
Figure 2.9 Save As... Dialog Box 22
Figure 2.10 Save As... Dialog Box with Warning 22
Figure 2.11 Edit Menu with Disabled Commands 24
Figure 2.12 About... Dialog Box 25
Figure 3.1 Accelerator Keynames 30
Figure 4.1 Selected Text in an Inactive Window 36
Figure 4.2 Selected Items in Reverse Video 38
Figure 4.3 Selected Region Enclosed in Rectangle 39
Figure 7.1 Dialog Box Requesting a Filename 61
Figure 7.2 Dialog Box with Separate Disk-Drive List Box 62
Figure 8.1 Controls in a Dialog Box 67
Figure 8.2 Control Labels 68
Figure 8.3 Related Radio Buttons 72
Figure 9.1 Dialog Box with Warning 78

Figure 9.2 Save As... Dialog Box with Warning 78
Figure 9.3 Dialog Box with Warning 79
Figure 9.4 Dialog Box with Print Message 79
Figure 10.1 Hourglass Pointer 84
Figure 12.1 Help Facility as Push-Button Control 95
Figure 12.2 Help Topics Dialog Box 96
Figure 12.3 Help Information Dialog Box 97
Figure 13.1 Multiple-Document Child Windows in a Main Window 101
Figure 13.2 Window Menu 102
Figure 13.3 Child Window's Control Menu 104
Figure 13.4 Maximized Child Window 105

Introduction

The Microsoft® Windows presentation manager is designed to be an interface that makes it easier to use a computer. To fulfill this design, Windows applications should provide a friendly user interface that is consistent with other Windows applications so that the user does not have to relearn basic concepts.

This style guide provides suggestions on how to design a consistent user interface for Windows applications. These are reasonable guidelines, intended to encourage a common user interface for all Windows applications, to benefit the user. Applications that employ these guidelines are easier to learn and use.

This guide does not cover all possible user-interface issues. Applications that have needs not covered by this guide should develop logical extensions of these guidelines. Applications should deviate from the guidelines only when absolutely necessary.

The Microsoft Windows presentation manager runs with the MS-DOS® operating system. MS-DOS may also be known as DOS or PC-DOS. In this style guide, MS-DOS will be referred to as DOS.

Key combinations and sequences are referred to in a particular style in this guide. A plus sign (+) between two keynames indicates that those keys must be pressed at the same time. For example, "Press ALT+ESCAPE" means that users should press and hold down the ALT key while they press and release the ESCAPE key. A comma between two keynames indicates that those keys must be pressed sequentially. For example, "Press ALT, SPACEBAR" means that users should press and release the ALT key, then press and release the SPACEBAR.

Chapter 1
Main Window

1.1 Introduction 5
1.2 Icons 6
1.3 Window Captions 6
1.4 Output 7
1.5 Scrolling 8
1.6 Split Windows 10

1.1 Introduction

Every application developed for Microsoft Windows should have at least one window to serve as the main visual interface for all user interaction with the application. This main window should be an overlapping window with a title bar, Control menu, Minimize and Maximize boxes, and a window border. (Applications that are fixed in size, however, should not have a window border or a Maximize box.) A menu bar is recommended if the application has commands. Scroll bars are recommended if the application contains more information than can fit in a window. The following screen shows the recommended format of the main window and its components:

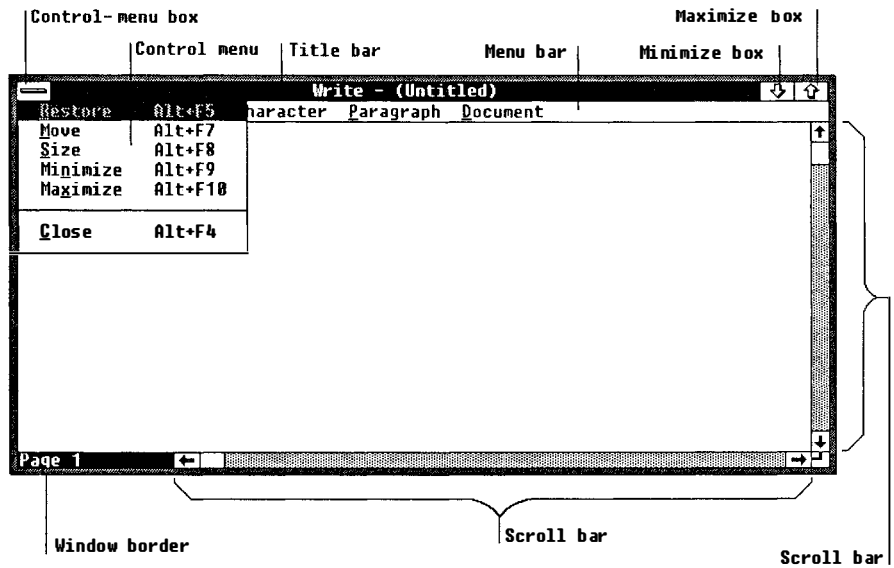


Figure 1.1 Main Window

The application should create and display the main window as its first task to let the user know the application is running. It should destroy the main window as its last task.

1.2 Icons

Every application should have an icon that uniquely identifies the application. Windows uses the icon to symbolize the loaded application when that application's main window is shrunk and not visible.

An icon can be designed and added to an application's executable file as a resource. The application registers the icon with Windows when the main window is created.

When an application is reduced to an icon, it should hide all its associated dialog boxes and child windows. If an application is unable to hide its dialog box (because user input is required), then the application should cause a warning beep to sound. When a window is restored from an icon, it is the application's responsibility to display any dialog boxes or child windows previously displayed.

An application can also choose to draw the icon while the application is running. This permits an application to continue to display some output even though its main window is not visible. This type of output is limited to the space occupied by the icon. The application must be prepared to update the icon when required. An example is Clock, which continues to display the correct time when it is reduced to an icon, shown as follows:

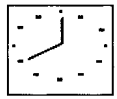


Figure 1.2 Clock Icon

1.3 Window Captions

To identify an application's window, every application should place a unique caption in the title bar of its main window. At a minimum, the caption should contain the application's name, with the first letter capitalized. Additional text may be added.

If an application's window content is always associated with a specific data file, the application should add the name of the data file to the caption in the title bar, separating the application name from the filename with a hyphen. The filename should appear in uppercase letters and should include the filename extension (if any). The file's pathname should not be included.

The following screen shows the recommended format for the title-bar caption of the main window:

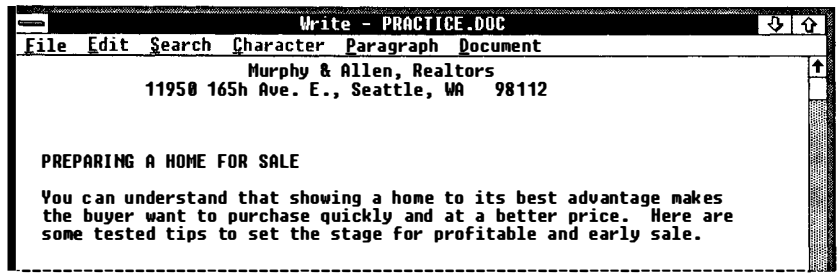


Figure 1.3 Window Caption with Filename

If there is no current file, the string "(Untitled)" should be used, as shown in the following example:

```
Paint - (Untitled)
```

1.4 Output

Applications should display all output, except error messages, warnings, and prompts for additional command information, in the client area of the application's windows. The client area is the rectangle bounded by the menu bar on the top, the scroll bars, if any, on the right side and bottom, and the window border on the left side. The client area is the area the user has access to. Output can be text, graphics, or both. The application must decide the format and draw the actual display. Applications that need to display a complete image (for example, the complete face of a clock) should adjust the image to fit in the available space.

Since client-area size and shape can vary, an application can adjust its display to either resize or clip the image. When an image is clipped, any part of the output that extends beyond the client area is not displayed. Applications that clip output should provide a method of scrolling the output in the window (see Section 1.5). An application must not display any of its output in another application's window.

Applications must be prepared to update the display when requested to by Windows.

The following screen shows two applications with output in their windows:

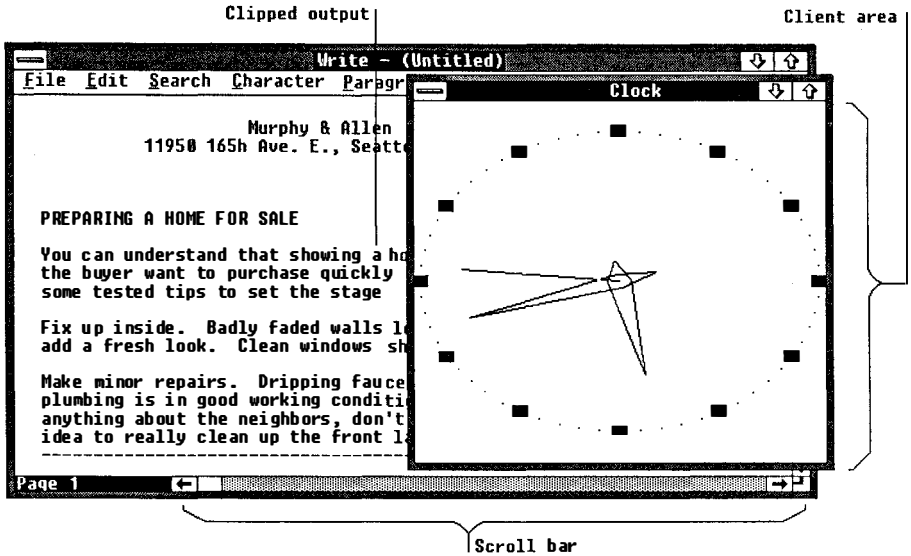


Figure 1.4 Output in Windows

1.5 Scrolling

Applications should provide scrolling for a window's client area whenever the output is too large to display all at once. For example, a window whose output is the contents of a data file can display a portion of the file at a time and let the user scroll the contents of the file as desired.

Applications that scroll should provide ways for scrolling with both the keyboard and the mouse (or other pointing device).

For scrolling with the keyboard, applications should use the following recommended keys:

Key	Action
PAGE UP	Scrolls up one screen.
PAGE DOWN	Scrolls down one screen.
CONTROL+PAGE UP	Scrolls left one screen.
CONTROL+PAGE DOWN	Scrolls right one screen.

For scrolling with the mouse, the application should include a horizontal or vertical scroll bar with the window (it should use both if it permits scrolling in all four directions). Scroll arrows at either end of the scroll bar indicate the direction of scrolling. The rectangular scroll box moves within the scroll bar to indicate the position of the displayed data in the file. The following screen shows the placement of scroll boxes and scroll arrows:

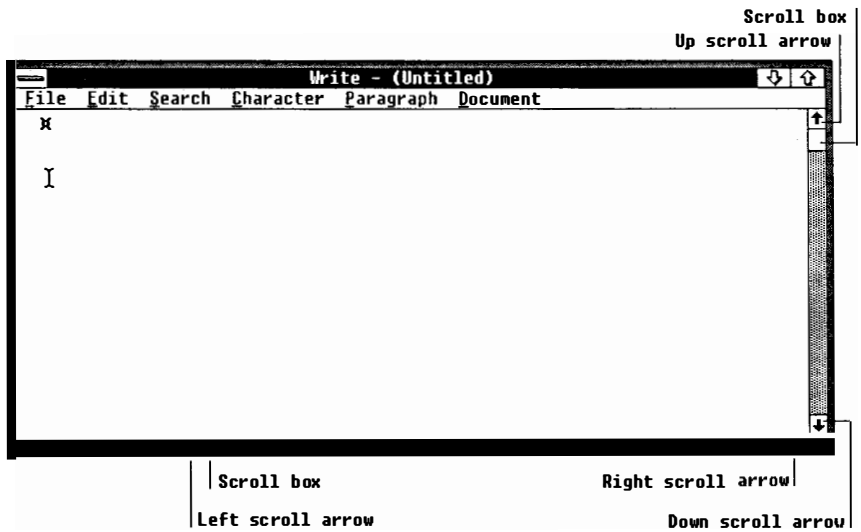


Figure 1.5 Scroll Boxes and Scroll Arrows

The application should respond to user input as follows:

Mouse Movement	Action
Click the up scroll arrow	Scrolls up one unit.
Click the down scroll arrow	Scrolls down one unit.
Click the left scroll arrow	Scrolls left one unit.
Click the right scroll arrow	Scrolls right one unit.

Click the area above or below the scroll box in the vertical scroll bar	Scrolls up or down one screen.
Click the area to the left or right of the scroll box in the horizontal scroll bar	Scrolls left or right one screen.
Drag the scroll box	Scrolls to the approximate location in the file as shown on bar.

1.6 Split Windows

Applications can split the client area of a window into separate viewing areas by creating child windows in the client area. A split window permits different types of output within the same main window. An example might be a spread-sheet application, in which the user needs to see different areas of the data simultaneously.

Applications that split windows in this way are responsible for maintaining each individual window. A child window must reside within the client area of its parent window. Child windows do not usually have icons to represent them.

The following screen shows a split window:

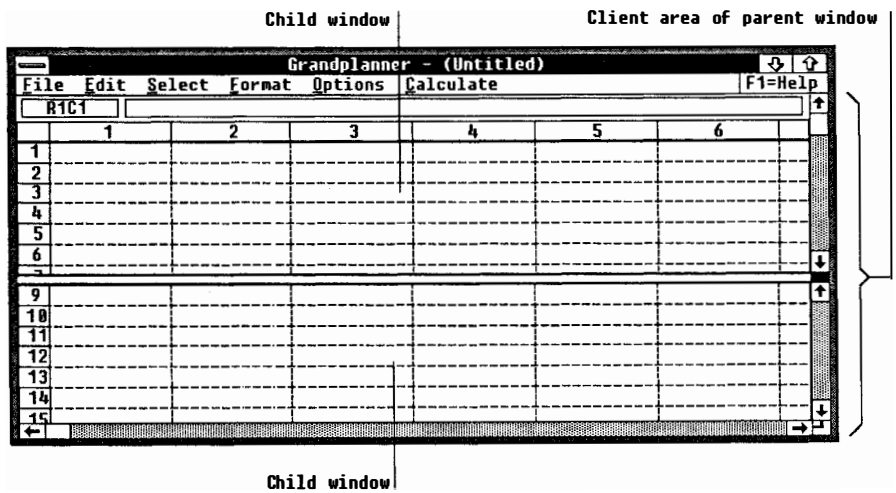


Figure 1.6 Split Window with Two Child Windows

The common use of split windows is also a characteristic of the multiple-document interface provided by Windows. The multiple-document interface supports the viewing of multiple windows within one application—either different views of the same data, or different documents or sets of data. An application that uses the multiple-document interface can contain document windows (child windows within the main window) and subdivisions of windows (child windows within a document window). For more information, see Chapter 13, "Multiple-Document Interface."

Chapter 2
Menus

2.1 Introduction 15
2.2 Menu Names 15
2.3 Menu-Item Names 16
2.4 Menu-Item Types 16
2.4.1 Toggle Commands 17
2.4.2 Menu Items That Use Dialog Boxes 18
2.5 Disabled Menu Items 18
2.6 The Control Menu 19
2.6.1 Disabled Control-Menu Items 19
2.6.2 The Close Command 20
2.7 The File Menu 20
2.7.1 The New and Open... Commands 21
2.7.2 The Save Command 22
2.7.3 The Save As... Command 22
2.8 The Edit Menu 23
2.9 The Exit Command 24
2.10 The About... Command 24

2.1 Introduction

Applications developed for Microsoft Windows that have commands should provide menus to give the user access to the commands. A menu is a list of items, such as commands, from which the user chooses. Each menu is represented by a name in the menu bar of the application's main window. The list of items is displayed when the user selects the corresponding menu name by using the keyboard or the mouse.

2.2 Menu Names

Every menu must have a menu name. The menu name represents the complete menu, so it should always state as clearly as possible the purpose of the items in the menu. Compound names may be used (for example, Fontsize). Numbers should not be used in the menu bar, but are acceptable for use with menu items in drop-down menus. Menu names should not be padded with spaces.

To give keyboard users direct access to menus, menu names that use mnemonics are recommended. A mnemonic is a unique character in the menu name that allows the user to select the menu by first pressing the ALT key, then pressing the key that corresponds to the character. Mnemonics should be, in order of preference, the first letter of the menu name, a consonant in the name, or a letter in the name. Applications that use a non-Latin alphabet (for example, Kanji) and operate on a standard keyboard, such as an IBM-PC keyboard, may use a Latin alphabet character prefix. Underlining indicates the mnemonic character.

The following screen shows a list of menu names in the menu bar:



Figure 2.1 Menu Names

2.3 Menu-Item Names

Menu-item names should specify the action the user will initiate by choosing them. Item names should be unique within a menu, but applications may use the same item name in different menus to represent different actions. Compound names may be used. Mnemonics should be used both on the menu bar and for menu items in a drop-down menu. If numbers are used as mnemonics in menu items, the format of the items should be number, space, then command text. Unlike menus, menu items can be chosen by pressing just the mnemonic key.

In general, items that have similar functions should be placed together. Items that are clearly different should be separated from other items with a line separator. Menu-item names should not be placed in the menu bar. Only menu names should be included in the menu bar. The following figure shows a sample listing of menu items:

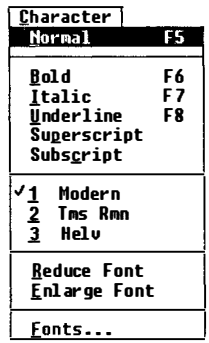


Figure 2.2 Items in a Menu

2.4 Menu-Item Types

Menu items are of several types. They may cause a specific command to be carried out, or they may allow the user to choose from among several features. A menu item may also give the user the option to turn a certain feature on or off (a toggle command). Some menu items request additional information from the user by generating a dialog box.

Menu items include the following:

Item	Description
Command	Carries out actions, such as opening or deleting a file.
State setter	Sets some system state. For example, these may be used to turn a feature on or off (a toggle command) or to choose from several features.
Extended command	Causes a dialog box to appear before the command is carried out. The user then is required to supply additional information, make additional choices, or cancel the request.

The following figure shows several types of menu items:

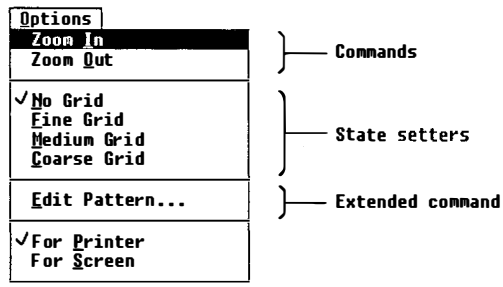


Figure 2.3 Types of Menu Items

2.4.1 Toggle Commands

Applications should always show whether the system state implied by a toggle command is active or not. If the command is active, the application should place a checkmark next to the command name. Otherwise, no checkmark should appear.

Menus may contain any number of toggle commands. Toggle commands are of two basic types: independent and mutually exclusive. If toggle commands are independent, they may be on or off regardless of the state of other commands. Mutually exclusive toggle commands will, when activated, deactivate associated toggle commands.

Toggle commands that affect the same command should be placed together and should be separated from other commands with a line separator. If many toggle commands are in one group, at least one command

should turn off all commands in that group. If two or more toggle commands are mutually exclusive, the application must make sure that the commands cannot be checkmarked at the same time. Mutually exclusive commands should be separated from other menu commands by a line separator.

The following figure shows two groups of mutually exclusive toggle commands with checkmarks that indicate which commands are active:

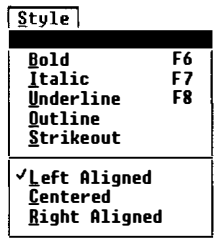


Figure 2.4 Mutually Exclusive Toggle Commands

2.4.2 Menu Items That Use Dialog Boxes

Menu items that require additional information before they can carry out their actions should prompt the user for the information by displaying a dialog box. The dialog box requires action by the user before the application can continue. The menu-item name should end with an ellipsis (...) and the item, when chosen, should display the dialog box.

For more information on dialog boxes, see Chapter 8, "Dialog Boxes."

2.5 Disabled Menu Items

The application must determine when the use of a menu item is appropriate. When the use of an item is not appropriate, the application must disable the item and gray the menu-item name. A grayed item lets the user know the item is disabled. Graying an item is only an indicator, and if the item is not specifically disabled, the user can still choose it. When this happens, a message is sent to the application. It is up to the application to determine how best to respond to the user action.

The following figure shows grayed menu items:

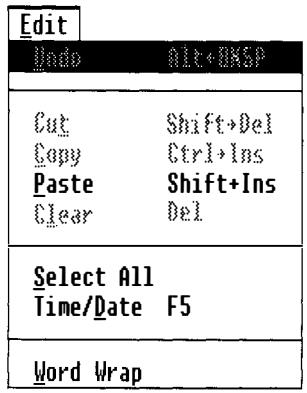


Figure 2.5 Grayed Menu Items

2.6 The Control Menu

The Control menu is predefined by Microsoft Windows. Therefore, the list of menu items, the order in which they appear, and their functions are, for the most part, preset and cannot be changed. Two exceptions are the Close command, which may be removed from the Control menu, and the About... command, which may be appended to the Control menu if there is no other logical place to locate it. (Otherwise, About... is added to the first menu in the application's menu bar. See Section 2.10 for more information on the About... command.) The following figure shows the Control menu:

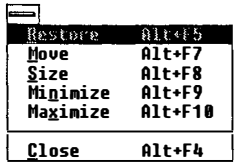


Figure 2.6 Control Menu

2.6.1 Disabled Control-Menu Items

When an application is not using items in the Control menu, the items should be grayed. For example, an application may not allow its window to change sizes. When this application is running, the Size command listed in the Control menu should be grayed.

2.6.2 The Close Command

The Close command is the last command on the Control menu for all windows, including dialog boxes and child windows, unless an About... command is appended to the menu. The Close command is defined to mean "close the window." In the case of the parent window of the application, it may be interpreted to mean "end the application." Double-clicking the Control-menu box with the mouse is an accelerator for Close. However, the recommended way for an application to quit is through the use of the Exit command appended at the bottom of its first drop-down menu. (See Section 2.9 for more information on the Exit command.) If any window within the application should always remain open, the Close command may be removed from the window's Control menu.

If the user chooses the Close command from the Control menu after making changes without saving them, the application should display a dialog box warning of data loss. For more information on dialog boxes and warnings, see Chapter 8, "Dialog Boxes."

2.7 The File Menu

Every application that works with data files should have a File menu. The File menu should provide all commands needed to open and create files and to save data in files. The File menu should be the first menu listed in the menu bar.

It is recommended that the File menu contain the following commands and that they have the following actions:

Command	Action
New	Deletes all existing data, removes the current filename from the title bar, and starts the application at its beginning. If choosing this command could destroy existing data, the command should generate a dialog box asking the user to verify the action before it is carried out. The letter "N" is the recommended mnemonic.
Open...	Prompts the user for the name of the file to be opened, then opens it and uses the file as the current file. The letter "O" is the recommended mnemonic.
Save	Saves the existing data in the current file. If the file is untitled, the application should generate a Save As... dialog box. The letter "S" is the recommended mnemonic.

Save As...	Prompts the user for the name of a file to save the current data in, then opens the file and copies data to it. The letter "A" is the recommended mnemonic for the Save As... command.

The following figure shows the recommended format for the File menu:

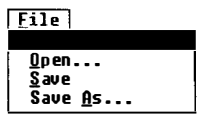


Figure 2.7 File Menu

Menu items that prompt for user input should use application-modal dialog boxes. For more information on dialog boxes, see Chapter 8, "Dialog Boxes."

Note

In all cases, applications should take the safe approach when carrying out default actions with files; that is, they should always attempt to prevent the loss of data from existing files.

2.7.1 The New and Open... Commands

If the user chooses the New or Open... command (or the Close command from the Control menu) after making changes without saving them in a file, the application should display a dialog box with the following format:

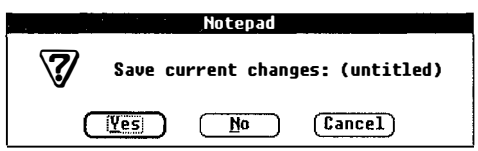


Figure 2.8 Dialog Box with Warning

The application should use the appropriate description of the data file in the first line of the message (in the preceding figure, the date file is untitled). The Yes button is the default button.

2.7.2 The Save Command

The Save command should overwrite the existing file without asking permission; however, if there is no current filename, the application should carry out a Save As... command instead.

2.7.3 The Save As... Command

The Save As... command should display a dialog box with the following format:

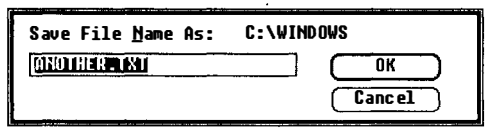


Figure 2.9 Save As... Dialog Box

The Save As... dialog box should propose the existing filename as the default. If there is no existing filename, it should propose a reasonable default. The OK button is the default button.

If choosing the Save As... command would overwrite an existing file, the command should always display a dialog box to ask permission to overwrite, as shown in the following example:

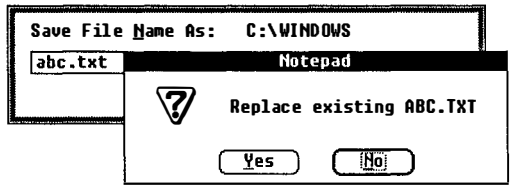


Figure 2.10 Save As... Dialog Box with Warning

If the user selects the No button, the application should return to the Save As... dialog box. Otherwise, the application should attempt to save the file. The No button is the default button.

2.8 The Edit Menu

Every application that lets the user move or copy text or graphics by using Clipboard should have an Edit menu. The Edit menu should provide all the commands needed to carry out Clipboard operations.

The Edit menu should contain at least the following commands:

Command	Action
Undo	At a minimum, reverses the action of the most recently chosen Edit-menu command. You may want to add more information to this command and expand its function. The letter "U" is the recommended mnemonic.
Cut	Copies the selection to the Clipboard, then deletes the selection. The letter "t" is the recommended mnemonic.
Copy	Copies the selection to the Clipboard without deleting the selection. The letter "C" is the recommended mnemonic.
Paste	Copies the contents of the Clipboard to the application's data file. The letter "P" is the recommended mnemonic.
Clear	Deletes the selection, but leaves the Clipboard unchanged. The letter "l" is the recommended mnemonic.

Commands that require a selection should create a selection through the techniques described in Chapter 4, "Selection."

If no current selection exists, the Cut, Copy, and Clear commands should be disabled. If the Clipboard is empty, the Paste command should be disabled. If no Edit-menu command has been chosen, the Undo command should be disabled. The following figure shows disabled Edit-menu commands:

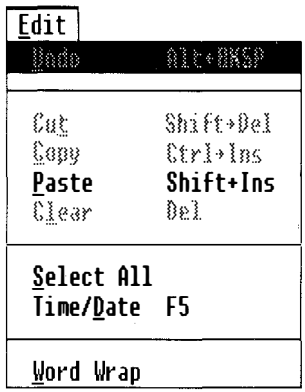


Figure 2.11 Edit Menu with Disabled Commands

2.9 The Exit Command

Each application should add an Exit command for quitting the program. The command should be placed at the bottom of the first menu of the application (unless an About... command is appended to the menu) and should be preceded by a line separator. The mnemonic recommended for access to this command is the letter "x." If the user chooses the Exit command after making changes without saving them, the application should display a dialog box warning of data loss.

The optional Close command on the Control menu is not a substitute for an Exit command unless the application does not have a menu bar.

2.10 The About... Command

When the user chooses the About... command, the application should display a dialog box that contains the application name, the date the application was created, copyright information, and any additional information about the application, such as available workspace or the meaning of specific keys. The following figure shows an example of the About... dialog box:



Figure 2.12 About... Dialog Box

If used, the About... command should be the last menu item on the Help drop-down menu or on the first drop-down menu of the application (following the Exit command). It should be preceded by a line separator. Its recommended mnemonic is the letter "b."

When an application does not have any menus, the About... command may be appended to the Control menu following the Close command. The About... command is the only command that may be added to the Control menu.

Every About... dialog box needs an OK button to allow the user to return to the application. Applications that provide additional help may include other buttons as well.

For more information on dialog boxes, see Chapter 8, "Dialog Boxes."

Chapter 3
Accelerators

3.1 Introduction 29
3.2 Accelerator Rules 29
3.3 Accelerator Names in Menus 29
3.4 Standard Control-Menu Accelerators 30
3.5 Standard Edit-Menu Accelerators 30
3.6 The Help Key 31

3.1 Introduction

Accelerators (shortcut keys) are keys that let the user choose a menu item more easily than the normal method of choosing the item. An accelerator is typically one of the following key combinations: ALT+*key*, CONTROL+*key*, SHIFT+*key*; it can also be a function key that the application has explicitly defined to represent a menu item, or a function key combined with the ALT, CONTROL, or SHIFT key. Pressing the key(s) carries out the item action just as if the user had displayed the menu and chosen the item. It is recommended that the menu name be highlighted when one of its associated accelerators is pressed so that the user can identify the location of the corresponding command.

3.2 Accelerator Rules

Applications developed for Microsoft Windows can provide any number of accelerators but should restrict their use to the menu items used most often. An item should never have more than one accelerator. Applications should avoid using accelerator keys that are reserved for the system; likewise, recommended accelerators should not be used for functions contrary to those described in this chapter and in Chapter 5, "Keyboard Interface."

3.3 Accelerator Names in Menus

Applications that use accelerators should place the accelerator's keyname beside the corresponding menu-item name in the menu. The keyname should appear to the right of the item name.

If an accelerator is an ALT+*key*, CONTROL+*key*, or SHIFT+*key* combination, it should be shown on the menu as Alt+*key*, Ctrl+*key*, or Shift+*key*, where *key* is the keyname. There should be no space between Alt+, Ctrl+, or Shift+ and the keyname. If an accelerator is a function key, it should be shown as F*n*, where F is uppercase and *n* is the digit. There should be no space between the letter and digit.

The following figure shows accelerator keynames listed to the right of the menu items:



Figure 3.1 Accelerator Keynames

3.4 Standard Control-Menu Accelerators

Certain accelerators are assigned to Control-menu commands and may not be used by applications for any other purpose. The reserved accelerators are as follows:

Key	Action
ALT+F5	Invokes the Restore command.
ALT+F7	Invokes the Move command.
ALT+F8	Invokes the Size command.
ALT+F9	Invokes the Minimize command.
ALT+F10	Invokes the Maximize command.
ALT+F4	Invokes the Close command if this command is present on the Control menu.

3.5 Standard Edit-Menu Accelerators

Applications that use accelerators with Edit-menu commands should use the following keys:

Key	Action
ALT+BACKSPACE	Invokes the Undo command.
SHIFT+DELETE	Invokes the Cut command.
CONTROL+INSERT	Invokes the Copy command.

SHIFT+INSERT	Invokes the Paste command.
DELETE	Invokes the Clear command.

The SHIFT+DELETE, CONTROL+INSERT, and DELETE Edit-menu accelerators should not invoke their respective commands if there is no selection. The SHIFT+INSERT accelerator should not invoke the Paste command if the Clipboard is empty. The ALT+BACKSPACE accelerator should not invoke the Undo command if no Edit-menu command has been chosen.

Note

If an application provides an Edit menu, it must not use the standard Edit-menu accelerators for any other purpose.

3.6 The Help Key

In addition to these standard accelerators, the F1 key is recommended for use as the Help key. The user presses F1 to obtain more information about a subject. See Chapter 12, "Help Facility," for more information about on-line help.

Chapter 4
Selection

4.1 Introduction 35
4.2 Selection Highlighting 35
4.3 Text Selection 36
4.4 Editing Text Selections 37
4.5 Non-text Selection 38
4.6 Discontinuous Selection 39

4.1 Introduction

Applications developed for Microsoft Windows that let the user copy information to the Clipboard or that have commands that act on the contents of the screen should provide a way to let the user select the information to be copied or acted on. The selection is one or more items that the user has chosen. These items will be used as input to a subsequent command.

There are two kinds of selections: text and non-text. Text selections are such things as a word or phrase in a manuscript the user is writing. Non-text selections can be either an item, such as a cell on a spread sheet, or a region on the screen, such as part of a drawing the user is making.

Applications that provide selection should let the user select with either the keyboard or the mouse. Applications should use the keys and the mouse buttons recommended in Chapter 5, "Keyboard Interface," and Chapter 6, "Mouse Interface."

4.2 Selection Highlighting

Applications should mark the selection by highlighting it. Highlighting can be achieved by displaying the selection in reverse video (used for text and item selections) or by enclosing it in a rectangle (used for some graphic selections).

When the window containing the selection is no longer active, the application should remove the highlighting. When the window becomes active again, the application should restore the highlighting.

If the window containing the selection is no longer active but the application is still using the selection in some way, the application should keep the selection highlighted. For example, in Microsoft Windows Write, if the user selects a section of text, then chooses the Fonts command from the Character menu, a dialog box appears. The dialog box becomes the active window, but the text remains selected, as shown in the following screen:

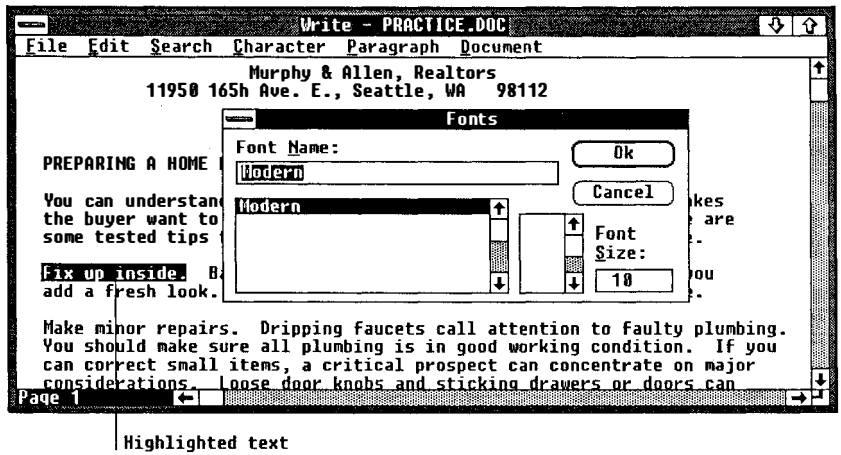


Figure 4.1 Selected Text in an Inactive Window

4.3 Text Selection

When an application permits the user to select text, the following types of selection are recommended:

Selection	Result
Insertion point	User should be able to select a point between characters to insert new characters.
Character	User should be able to select individual characters, including punctuation or a space.
Word	User should be able to select individual words. A word is any contiguous string of characters not containing a word-break character. Word-break characters are such things as new-line characters and tabs.
Line	User should be able to select whole lines. A line is any contiguous string of characters not containing a new-line character.
Extended	User should be able to extend the existing selection to include additional characters. For example, the user should be able to extend a character or word selection to include several more characters.

All text selections, except the insertion-point selection, should be highlighted. The insertion-point selection should be marked with a flashing caret (shown as a flashing bar or bitmap).

An extended selection should always begin at the original selection and continue either to the left or to the right (or up or down). An application should not allow a selection to be extended in both directions at the same time. Instead, there should be one active edge of selection. If, when using the keyboard, the user extends the selection to the beginning or end of a line, the extension should move to the opposite end of the next line up (if moving left) or down (if moving right). Extending the selection up is the same as extending it to the left quickly; extending it down is the same as extending it to the right quickly. If, while extending the selection to the left (or right), the user moves the selection in the opposite direction, the application should remove the extended selection in the reverse order in which it was made.

The application should let the user remove a text selection by making another selection, pressing a key, or clicking with the mouse.

4.4 Editing Text Selections

Applications that permit editing of selected text should follow these rules:

- When text is inserted at an insertion point, the insertion point moves to the right and new characters are placed to the left of the insertion point (it does not put the character on the Clipboard).
- When a character, word, or line is selected, inserting text removes the selected text and replaces the selection with an insertion point. The insertion continues as described in the previous rule.
- When a character, word, or line is selected, deleting text removes the selected text.
- Using the BACKSPACE key at an insertion point deletes the character to the left and moves the insertion point to the left.
- Using the DELETE key at an insertion point deletes the character to the right of the insertion point (it does not put the character on the Clipboard).

4.5 Non-text Selection

Applications that permit the user to select items other than text should provide the following types of selection:

Selection	Result
Item	User should be able to select individual items on the screen, such as filenames from a directory listing, or icons.
Region	User should be able to select the contents of a rectangular region on the screen.

Applications should let the user make any number of item selections. The selections should be highlighted, appearing in reverse video. The user can extend the selection by selecting additional items, as shown in the following figure:

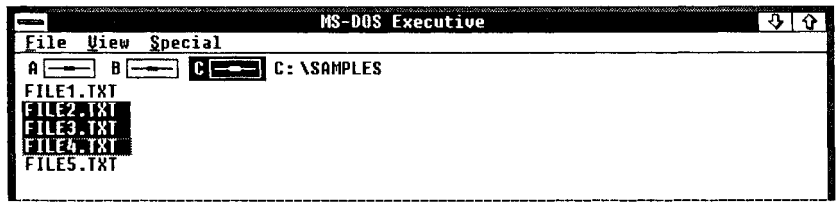


Figure 4.2 Selected Items in Reverse Video

Applications should let the user select any rectangular region on the screen. The user begins the selection at any point and moves it any direction to an end point. The start and end points mark the diagonally opposite corners of the rectangular region. While the user selects the region, the application should highlight the region to mark the user's progress. The highlight can be a reverse-video display or an enclosing rectangle. A selected region is shown highlighted by an enclosing rectangle in the following screen:

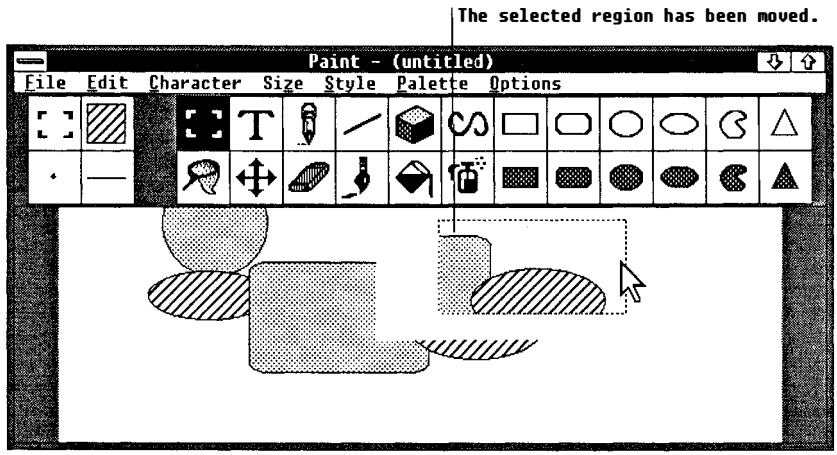


Figure 4.3 Selected Region Enclosed in Rectangle

4.6 Discontinuous Selection

A discontinuous selection is any non-text selection in which two or more non-adjacent items on the screen are highlighted. An application can permit discontinuous selections whenever the user needs to select multiple items from a list of items, such as two or more filenames from a directory listing.

Applications should not permit discontinuous text selections. Instead, each new selection should clear the previous selection.

Chapter 5
Keyboard Interface

5.1 Introduction 43
5.2 General Recommendations 43
5.3 Reserved System Keys 44
5.3.1 Keys Not to Be Redefined 44
5.3.2 Keys Reserved During Menu Use 45
5.3.3 Keys Reserved During Dialog-Box Use 45
5.4 Recommended Keyboard Use 46
5.4.1 General Use Keys 46
5.4.2 Keys Used for Moving and Selecting 46
5.5 Keys Used in Child Windows 49
5.6 Keyboard Input Focus 50

5.1 Introduction

Since Microsoft Windows does not require a mouse or similar pointing device, applications should ensure that every menu item can be selected with the keyboard. (To test this, it is recommended that you unplug the mouse card and try every menu item with the keyboard.)

To assist application design, Windows provides a built-in keyboard interface for accessing an application's menus; for selecting the active application, window, or control; and for moving, sizing, enlarging, shrinking, and closing a window. For other actions, such as scrolling a window and selecting objects in a window, the use of specific keys is recommended, but the application should carry out the actual operations.

5.2 General Recommendations

An application has a wide range of key combinations available in Windows. In selecting keys, use the following guidelines:

- Use single keys for frequently performed, small-scale tasks. For example, in an application that is using split windows, a single function key, F6, can be used to move the input focus clockwise from one subdivision of a window to another.
- Use SHIFT+*key* combinations for actions that are complimentary to the actions of the key used alone. For example, if the F6 key moves the input focus clockwise from one subdivision of a window to another, the combination of SHIFT+F6 would be used to move the input focus counterclockwise.
- Use CONTROL+*key* combinations for less frequent actions, or larger-scale tasks. For example, CONTROL+F6 could be used to move the input focus from one document window to another within the main window.
- Avoid using ALT+*key* combinations since they are generally used for system tasks.

5.3 Reserved System Keys

The reserved system keys are the built-in system keys Windows uses to access an application's menus and dialog boxes. Applications can use some of these keys for normal input and output when the application has the input focus. *Applications must never attempt to trap these keys.*

5.3.1 Keys Not to Be Redefined

The following reserved keys are active at all times and must not be redefined by any application:

Key	Action
ALT+F4	Closes the window.
ALT, SPACEBAR	Selects the Control menu of the active window.
SHIFT+ESCAPE	Selects the Control menu of the active window.
ALT, MINUS	Selects the Control menu of the active document window.
ALT+TAB	Selects the next active window, moving from front to back. If an icon is selected, it is restored to a window.
ALT+SHIFT+TAB	Selects the next active window, moving from back to front.
ALT+ESCAPE	Selects the previously active window in the next active application, moving from front to back.
ALT+SHIFT+ESCAPE	Selects the previously active window in the next active application, moving from back to front.
CONTROL+ESCAPE	Reserved for future use.
ALT+F5	Restores the window to its previous size.
ALT+F7	Moves the window.
ALT+F8	Sizes the window.
ALT+F9	Shrinks the window to an icon.
ALT+F10	Enlarges the window to its maximum size.
F10 or ALT	Sets the menu mode.

5.3.2 Keys Reserved During Menu Use

The following reserved keys are active when a menu has been selected. At other times, they may be redefined by an application:

Key	Action
LEFT	Moves to the next column within a menu, or selects the next menu, moving from right to left.
RIGHT	Moves to the next column within a menu, or selects the next menu, moving from left to right.
UP	Selects the command above the selected command in the selected menu.
DOWN	Selects the command below the selected command in the selected menu, or if no menu is displayed, displays the selected menu.
ENTER	Carries out the selected command.
ESCAPE	Cancels the selected menu.

5.3.3 Keys Reserved During Dialog-Box Use

The following reserved keys are active when a dialog box is displayed. At other times, they may be redefined by an application:

Key	Action
TAB	Selects the next active control, moving from left to right and top to bottom.
SHIFT+TAB	Selects the next active control, moving from right to left and bottom to top.
SPACEBAR	Activates the selected button.
ENTER	Carries out the default action as defined by the default push button.
ESCAPE	Cancels the dialog box.

5.4 Recommended Keyboard Use

Although applications can use all keys on the keyboard for their normal input operations, for the sake of a consistent keyboard interface, it is recommended that applications use the following keys for the actions listed.

5.4.1 General Use Keys

There are a number of common functions in applications. The following key assignments are recommended for these cases:

Key	Recommended Action
F1	Reserved as the Help key.
F6	Selects the next active subdivision of a window (a child window within a document window), moving clockwise.
SHIFT+F6	Selects the next active subdivision of a window, moving counterclockwise.
CONTROL+F6	Selects the next active document window (a child window within the main window), moving from front to back.
SHIFT+CONTROL+F6	Selects the next active document window, moving from back to front.
ALT+F6	Selects the application's next active non-child window (such as a modeless dialog box), moving from front to back. For information on modeless dialog boxes, see Chapter 8, "Dialog Boxes."
SHIFT+ALT+F6	Selects the application's next active non-child window, moving from back to front.
ALT+F1	Same as the F11 key.
ALT+F2	Same as the F12 key.

5.4.2 Keys Used for Moving and Selecting

The following keys are recommended for selecting items, for formatting and selecting text or graphics, and for moving within a window. A key's actions may vary according to whether it affects items, text, or graphics; therefore, some of the keys listed below have several recommended actions:

Key	Recommended Action
TAB	Moves to the next item, moving from left to right and top to bottom; moves text right one tab stop.
SHIFT+TAB	Moves to the next item, moving from right to left and bottom to top; moves text left one tab stop.
ENTER	Confirms or executes the default item; starts a new line of text.
LEFT	Moves to the item to the left of the current item; moves left one character.
RIGHT	Moves to the item to the right of the current item; moves right one character.
UP	Moves to the item above the current item; moves up one line.
DOWN	Moves to the item below the current item; moves down one line.
CONTROL+LEFT	Moves to the left or beginning of a field, group, or section.
CONTROL+RIGHT	Moves to the right or end of a field, group, or section.
CONTROL+UP	Moves to the top or beginning of a field, group, or section.
CONTROL+DOWN	Moves to the bottom or end of a field, group, or section.
SHIFT+CONTROL+LEFT	Moves the input focus and extends the selection to the left or beginning of a field, group, or section.
SHIFT+CONTROL+RIGHT	Moves the input focus and extends the selection to the right or end of a field, group, or section.
SHIFT+CONTROL+UP	Moves the input focus and extends the selection to the top or beginning of a field, group, or section.
SHIFT+CONTROL+DOWN	Moves the input focus and extends the selection to the bottom or end of a field, group, or section.

PAGE UP	Scrolls up one screen.
PAGE DOWN	Scrolls down one screen.
CONTROL+PAGE UP	Scrolls left one screen.
CONTROL+PAGE DOWN	Scrolls right one screen.
SHIFT+PAGE UP	Scrolls and extends the selection up one screen.
SHIFT+PAGE DOWN	Scrolls and extends the selection down one screen.
SHIFT+LEFT	Moves the input focus and extends the selection left one unit.
SHIFT+RIGHT	Moves the input focus and extends the selection right one unit.
SHIFT+UP	Moves the input focus and extends the selection up one unit.
SHIFT+DOWN	Moves the input focus and extends the selection down one unit.
INSERT	Toggles in the insert/replace mode.
SHIFT+INSERT	Inserts (pastes) the contents of the Clipboard.
CONTROL+INSERT	Copies the selection to the Clipboard.
DELETE	Deletes (clears) the selection.
CONTROL+DELETE	Deletes from the current position to the end of the line.
SHIFT+DELETE	Removes (cuts) the selection and puts it on the Clipboard. If there is no selection, it removes the character to the right and puts it on the Clipboard.
BACKSPACE	Deletes the character or space left of the insertion point.
ALT+BACKSPACE	Cancels the last editing action.
HOME	Moves to the beginning of the line.
CONTROL+HOME	Moves to the beginning of the data.
SHIFT+HOME	Moves and extends the selection to the beginning of the line.
SHIFT+CONTROL+HOME	Moves and extends the selection to the beginning of the data.

END	Moves to the end of the line.
CONTROL+END	Moves to the end of the data.
SHIFT+END	Moves and extends the selection to the end of the line.
SHIFT+CONTROL+END	Moves and extends the selection to the end of the data.
CONTROL+DIRECTION	Leaves the selection intact and moves in the indicated direction.
CONTROL+SHIFT+DIRECTION	Adds to a discontinuous selection, moving in the indicated direction.
CONTROL+SPACEBAR	Adds to the current selection (permits discontinuous selection).
SHIFT+SPACEBAR	Extends the selection from its previous end point to the current location.

5.5 Keys Used in Child Windows

In applications that use a multiple-document interface, document windows should use the following key combinations to access Control-menu items:

Key	Action
ALT, MINUS	Selects the Control menu of the active document window.
CONTROL+F5	Restores the document window to its previous size.
CONTROL+F7	Moves the document window.
CONTROL+F8	Sizes the document window.
CONTROL+F9	Shrinks the document window.
CONTROL+F10	Enlarges the document window.
CONTROL+F4	Closes the document window.

5.6 Keyboard Input Focus

An application should acquire the keyboard input focus (current ownership of the keyboard) whenever it is the active window. Since only one window can have the keyboard input focus at a time, having the focus guarantees that the application receives all the input intended for it.

An application should relinquish the keyboard input focus whenever the user moves to another window. This permits the new window to acquire the focus if necessary. The application should reacquire the focus when the user returns to the application's window.

Although Windows provides a mechanism to trap keyboard input (and circumvent the input focus), applications should use this mechanism only when absolutely necessary.

Chapter 6
Mouse Interface

6.1 Introduction 53
6.2 Recommended Mouse Use 53
6.2.1 Text Selection with the Mouse 54
6.2.2 Item Selection with the Mouse 54
6.3 Mouse Pointers 55
6.4 Mouse Capture 55

6.1 Introduction

Although the use of the mouse (or similar pointing device) is optional with Microsoft Windows, applications should use it liberally as an alternate method of input. In particular, applications should always use the mouse wherever it clearly improves the application's performance.

Note

Since the mouse is the most common pointing device, for convenience, the word "mouse" is used in all discussions of pointing devices. However, information about using the mouse pertains to other pointing devices as well.

The following list defines the terms used to describe mouse movement:

Mouse Movement	Definition
Point	Move the mouse until the tip of the pointer rests on the desired item or area.
Click	Quickly press and release the mouse button.
Drag	Press the mouse button and hold it down while moving the mouse.
Double-click	Click the mouse button twice in rapid succession.

6.2 Recommended Mouse Use

As it does with the keyboard, Windows provides a built-in mouse interface for accessing an application's menus; for selecting the active application, window, or control; and for moving, sizing, shrinking, enlarging, and closing a window. Other specific actions should correspond to the mouse buttons and movements listed below. The application is responsible for carrying out the actual operations.

The left mouse button should be used for all of the application's actions. Other buttons may be used for shortcuts but should not be required by the application. Applications should use the mouse as shown in the following lists.

6.2.1 Text Selection with the Mouse

The following list describes the mouse movements recommended to select text:

Mouse Movement	Action
Click	Moves the insertion point to the area being pointed to.
Double-click	Selects a word.
Drag	Selects text from the area where the mouse button is pressed through to the release point.
SHIFT+click	Extends the selection from the insertion point to the area being pointed to.
SHIFT+drag	Extends the selection from the area pointed to when the mouse button is pressed through to the release point.

6.2.2 Item Selection with the Mouse

The following list describes the mouse movements recommended to select items:

Mouse Movement	Action
Click	Selects the item being pointed to.
Double-click	Confirms or executes the item being pointed to.
Drag	Extends the selection from the area pointed to when the mouse button is pressed through to the release point.
SHIFT+click	Extends the selection from the input focus to the area being pointed to.
CONTROL+click	Selects discontinuous items, adding an area or unit to the selection.
CONTROL+drag	Selects discontinuous items, keeping the current selection and creating an additional selection from the area where the mouse button is pressed through to the release point.

6.3 Mouse Pointers

Applications should use mouse pointers that are appropriate for the current action. When the user needs to select items by clicking with the mouse, the application should provide a pointer. A variety of pointers may be needed for the same action when the pointer is in different parts of the window.

Three mouse pointers are shown as follows:

Pointer	Description
	Arrow pointer. Selects menus, items, commands, etc.
	I-beam pointer. Indicates an insertion point.
	Hourglass pointer. Indicates that an operation is taking place even though no output is being sent to the screen.

6.4 Mouse Capture

Under normal operating conditions, an application receives mouse input only when the mouse pointer is in the application's window. Applications that want to receive mouse input from points outside the window can capture the mouse input, but should do so sparingly. When mouse input is captured, the user cannot select menus or change to another application.

Chapter 7
Filenames

7.1 Introduction 59
7.2 Default Filename Extension 59
7.3 Filename Conventions 59
7.4 Temporary Filenames 60
7.5 Prompting for Directories and Filenames 61

7.1 Introduction

Applications developed for Microsoft Windows that use filenames to create and open files should use a consistent approach when forming filenames from user input.

7.2 Default Filename Extension

Applications should use the same extension for all data files whenever possible. In particular, an application should have a default filename extension that it appends to a data filename whenever the user does not supply one. The default extension should be unique so that the user can distinguish one application's data files from another or use the file to invoke the application implicitly. The extension can be one, two, or three characters.

7.3 Filename Conventions

Applications should apply the following rules when forming complete filenames from user input:

- If the user supplies a filename without a pathname, by default, the application should search the current directory. If the file is not found, the PATH variable should be searched.

  If the user is saving a file and the file is not found in the search just described, it should be saved in the current directory. In some cases, a Save As... dialog box will be generated. If the file is in the current directory, the application should show only the filename. If the file is in another directory, the complete pathname should be shown.

- If the user supplies a pathname, the application should decide if it is appropriate. If it is not, the application should display an error message and prompt for a new name.

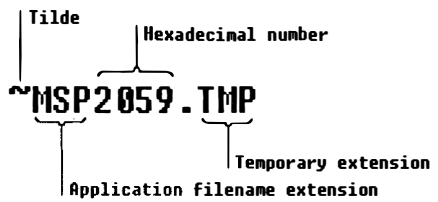
- If the user supplies a name without an extension, the application should append the application's default extension. The user should also log the default extension in the *win.ini* file.

- If the user supplies a null extension (for example, *abc.*), or changes the extension (for example, *abc.new*), the application must not append the default extension. The application may log the user-supplied extension in the *win.ini* file, though this isn't required. If the user changes the extension in *win.ini*, however, the application should accept the new extension as its default.

- If the filename is too long, the application should generate an appropriate error message.

7.4 Temporary Filenames

Applications that create temporary files (files to be used during execution only) should use names as shown in the following example:



The ~ is a required tilde.

The *filename extension* is the 1-, 2-, or 3-character default extension used by the application.

The *hexadecimal number* is unique to each new temporary file. This number is created by the Windows **GetTempFileName** function.

The temporary extension *.tmp* is required.

Since it is possible to run multiple copies of an application simultaneously, each temporary file must have a unique name to prevent one file from overwriting another. The **GetTempFileName** function can be used to assign a unique name to each file.

Temporary files should always be placed in the directory specified by the user's DOS TEMP environment variable, if one is specified. If no directory is specified, the files should be placed in the root directory.

7.5 Prompting for Directories and Filenames

Commands that require a filename or directory name to carry out their actions should prompt for the names using a dialog box with the following format:

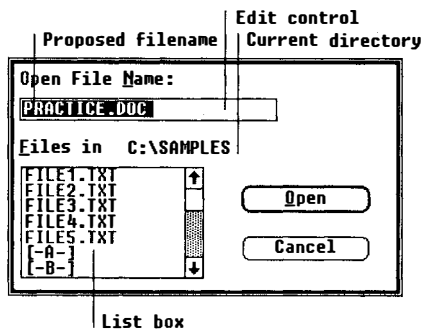


Figure 7.1 Dialog Box Requesting a Filename

The edit control is used to enter the desired filename or directory name. The list box lists the contents of a directory.

The application should place the current filename in the edit control as the proposed filename. If there is no current file, the application should propose an appropriate default name. The application should also select the proposed name for editing. This lets the user delete the name with a single keystroke if another name is desired.

The application should list the partial contents of the current directory in the list box. In particular, it should list all filenames that end with the application's extension and all directory names. Whenever the user selects a filename from the list box, the application should copy this name to the edit control.

In most applications, the available disk drives and subdirectories will be displayed in the list box along with the current directory listing. However, some applications may use separate list boxes for this purpose, as shown in the following example:

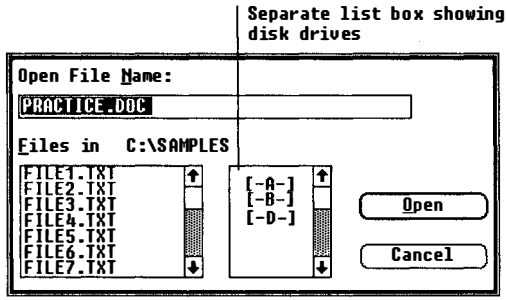


Figure 7.2 Dialog Box with Separate Disk-Drive List Box

For more information on using list boxes, see *Microsoft Windows Programming Tools.*

Chapter 8
Dialog Boxes

8.1 Introduction 65
8.2 Dialog-Box Types 65
8.3 Dialog-Box Appearance 66
8.4 Controls 66
8.4.1 Control Labels 67
8.4.2 Mnemonics in Dialog Boxes 68
8.5 Default Push Buttons 68
8.6 Required Buttons 69
8.7 Dialog Boxes with Multiple Controls 69
8.8 Dialog Boxes Without Push Buttons 70
8.9 Sizing Dialog Boxes 70
8.10 Moving Between Controls 70
8.11 Dialog-Box Appearance 71
8.11.1 Push-Button Design 71
8.11.2 Control Placement 71

8.1 Introduction

Applications developed for Microsoft Windows should use dialog boxes to request input from the user and to display information that is not part of the application's normal output. A special type of dialog box, called the message box, is used to display error messages and warnings. Message boxes are described in Chapter 9, "Message Boxes."

8.2 Dialog-Box Types

Dialog boxes can be modal or modeless. A modal dialog box prevents the user from continuing to work with the application until some action is taken. A modeless dialog box lets the user continue working with the application.

Applications should use modal dialog boxes for requesting command input. In this way, a modal dialog box ensures that the user responds to the input request before continuing to work.

Applications should use modeless dialog boxes for displaying information that does not require immediate attention. Modeless dialog boxes typically display information that the user can use while working with the application (such as a Help dialog box). Unlike modal dialog boxes, modeless dialog boxes can be moved out of the way by the user so that they don't obscure the user's work.

There are three types of dialog boxes:

Dialog Box	Description
Modeless	This dialog box is a separately functioning window. The user can continue to work with the application that called the dialog box or with any other application. The user can move the modeless dialog box if it obscures the work area and can close it at any time by using the dialog box's Control menu.
Application modal	This dialog box requires attention before the user can continue to work with the application that displayed it. This dialog box does not prevent the user from working with other applications.

System modal	This dialog box requires attention before the user can continue to work with *any* application. Use this dialog box only when absolutely needed, such as when warning about an impending fatal system error or an unrecoverable error that could disrupt system operation.

8.3 Dialog-Box Appearance

The appearance of an application's dialog boxes should be defined by the following rules:

- Dialog boxes should not contain menu bars.
- Dialog boxes should disable and gray (or remove) all unused Control-menu commands.
- A modal dialog box should have a framed border.
- A modeless dialog box should have a title bar and a Control menu.
- A modeless dialog box should have a caption in its title bar. The caption should be the name of the command that created the box.
- A modeless dialog box should disable and gray (or remove) all Control-menu commands except Move and Close.

8.4 Controls

Dialog boxes generally contain controls with which the user can supply information for the corresponding menu item. The controls can be any combination of the following:

Control	Description
Push button	A command button, labeled to indicate what it does (for example, OK, Cancel)
Radio button	An option button located in a group of option buttons from which the user may select one option
Check box	An option that is on or off
List box	A box listing available choices
Edit control	A box for entering information

The following dialog box shows several types of controls:

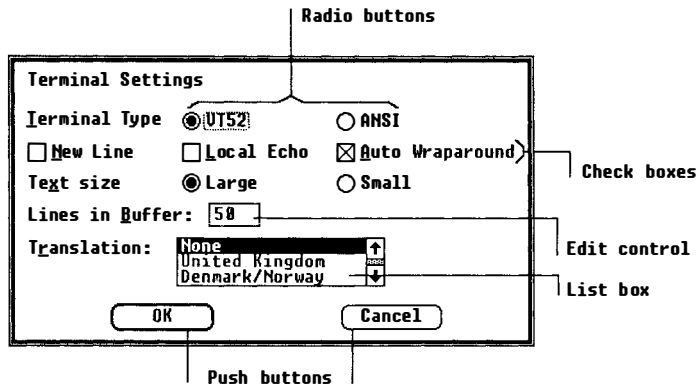


Figure 8.1 Controls in a Dialog Box

The number and type of controls depend on the dialog-box type and the input needs of the corresponding menu item.

The application should always provide proposed values for the information requested in the dialog box and give the user the option of selecting the proposed values or entering new values. Proposed values can be based on previous information or on standard defaults.

8.4.1 Control Labels

Controls should have appropriate labels, as defined by the following rules:

- Control labels should begin with a capital letter.
- All edit controls should have a static-control label that ends with a colon.
- Labels on radio-button groups should not contain a colon.
- Each control label should have a unique mnemonic, or the group label can be given the mnemonic. Visual indication and rules for selection are the same as for mnemonics in menus. For more information on mnemonics in dialog boxes, see Section 8.4.2.
- Inactive controls should be indicated by a grayed label. Inactive controls can still be selected by the user. It is up to the application to decide its response.

The following dialog box shows the recommended format for control labels:

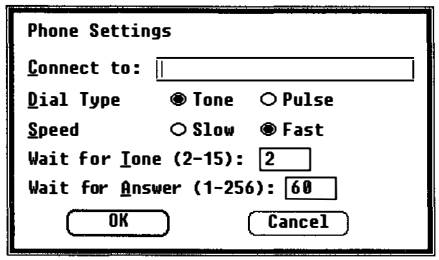


Figure 8.2 Control Labels

8.4.2 Mnemonics in Dialog Boxes

When mnemonics are used in dialog boxes, they are defined by the application in the same way that they are when used in menus.

If the control has a label, the mnemonic may be one of the letters in the label. If the control has a static-control label, as is the case with an edit control or list box, one of the letters of the static-control label should be used as the mnemonic.

If the mnemonic letter associated with the control is unique within the dialog box, the control is automatically selected when the user presses the corresponding key. If not, the input focus moves to the control, but the user must press the SPACEBAR to select it. If the mnemonic letter is associated with a static-control label such as a text field, the input focus automatically moves to the next control in the enumeration sequence, which may be a list box or edit control. If there is no match between any control and ALT+*mnemonic*, pressing such a key combination results in a system beep.

8.5 Default Push Buttons

Every modal dialog box should have a default push button with which the user can provide the most reasonable response to the dialog box. The default button has a bold border. For most dialog boxes, the default button carries out the command. In others, the button cancels the command.

In all cases, the ENTER key should activate the default push button, and the SPACEBAR should activate the selected button. The ESCAPE key should cancel the dialog box (if the dialog box has a Cancel button).

When the user moves between push buttons, the bold outline associated with the default action will move as the input focus is moved, allowing the user to confirm the new selection with the ENTER key or the SPACEBAR. If the user moves by tabs beyond the last push button in the sequence, the bold outline and default action return to the original push button.

8.6 Required Buttons

Dialog boxes that request input should provide at least one push button to carry out the requested action and one to cancel it. In most dialog boxes, the OK and Cancel buttons are used for these purposes. The OK button (the default button) directs the menu item to continue its action with the information supplied. The Cancel button is used to cancel the menu item. In cases where the default or suggested choice is negative, Yes and No buttons are generally used, with the No button as the default. This avoids the confusion that might occur from having the Cancel button be the default in this case. The recommended position for these push buttons is at the upper-right edge of the window with Cancel under OK.

When a button cannot be chosen, the label in the button should be grayed.

8.7 Dialog Boxes with Multiple Controls

Dialog boxes that contain multiple controls should use the controls in the following ways:

- When a push button is chosen, its related action is carried out.
- When check boxes and radio buttons are selected, they change their state (from selected to not selected, or vice versa).
- The TAB key moves the input focus from one section of the dialog box to another.
- The SPACEBAR activates the selected button.

8.8 Dialog Boxes Without Push Buttons

It is recommended that all dialog boxes have push-button controls, though some may not. Applications that use dialog boxes without push-buttons should enable the ENTER and ESCAPE keys to let the user carry out default actions. The application should interpret the ENTER key as confirming or, where appropriate, as executing the actions in the dialog box. Once the action is confirmed, the dialog box should be closed. The application should interpret the ESCAPE key as canceling the dialog box without carrying out an action. In modeless dialog boxes, the ESCAPE key closes the dialog box.

8.9 Sizing Dialog Boxes

Dialog boxes must be sized to fit on all screens that the application will use for display. Calculate the maximum size of a dialog box by using the following formulas:

- Width:

  [(screen width)/(width of system font)]*4=maximum dialog units
- Height:

  [(screen height)/(height of system font)]*8=maximum dialog units

For device-independent applications, the recommended system font is 80 × 24 characters. The corresponding dialog boxes would be a maximum of 320 × 96 dialog units. It is recommended that a dialog box take up no more than 75% of the screen.

8.10 Moving Between Controls

Moving the input focus within a dialog box is accomplished either by the use of general keys such as TAB and SHIFT+TAB (moving backwards by tabs), or by the use of mnemonics.

The normal order of movement within a dialog box should be from left to right and from top to bottom. The following list describes the rules of moving and selecting within a dialog box:

Key	Action
TAB or SHIFT+TAB	Moves the input focus from one control to the next within the dialog box.
ALT+*mnemonic letter*	Moves the input focus directly to a control within a dialog box; if the control is a push button, it activates the button.
Mnemonic letter	Moves the input focus directly to a control within a dialog box, unless the input focus is in an edit control or list box. If the control is a push button, it activates the button.
DIRECTION	Moves the input focus and selects within a group of radio buttons.
DIRECTION; TAB; or SHIFT+TAB	Moves the input focus within a group of check boxes.

8.11 Dialog-Box Appearance

This section contains suggestions for design and text format that you may want to use in creating dialog boxes for your applications. Following these guidelines will help ensure a consistent user interface.

8.11.1 Push-Button Design

Push-button height should be 1-1/2 times as high as the tallest character in the button name and also should include space for the dotted box, which marks the input focus.

The first letter in the push-button name should be a capital. All other characters in the name should be lowercase.

8.11.2 Control Placement

Controls in a dialog box should be located as follows:

- Place horizontal scroll bars below the window to be scrolled.
- Place vertical scroll bars to the right of the window to be scrolled.

- Label edit controls with static-control text.
- Related radio buttons should be grouped in one of two ways. The first method is to use the WS_GROUP window style to set the style for the first and last buttons in a group. This allows the user to move within the group using the DIRECTION keys. The second method is to enclose the related radio buttons in a group box. A group box is a rectangle that groups controls together.

  Examples of both methods are shown in the following figure:

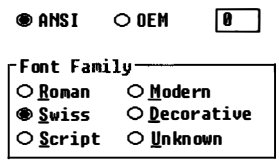


Figure 8.3 Related Radio Buttons

Chapter 9
Message Boxes

9.1 Introduction 75
9.2 Message-Box Types 75
9.3 The System Beep 76
9.4 Message-Box Captions 76
9.5 Push Buttons in Message Boxes 76
9.6 Keyboard Use 77
9.7 Flashing for Attention 77
9.8 Sample Message Boxes 78
9.9 Message-Box Text 79

9.1 Introduction

A message box is a special dialog box through which an application can display error messages and other important information. Message boxes are modal dialog boxes and require immediate attention by the user before work with the application can proceed. Applications developed for Microsoft Windows should use message boxes only when an error occurs or when the application must supply information that the user needs in order to complete an action.

9.2 Message-Box Types

There are four types of message boxes; each type has an appropriate icon that should be located in the upper-left corner of the dialog box, as shown in the following list:

Icon	Message Box	Description
	Status	Informs the user that a task is in progress. It also can specify an action that the user must complete before a task can proceed. This message box contains no choices for the user to make.
	Warning or Note	Warns or reminds the user of the action about to be taken. This message box asks for a user response and may contain choices for the user to make.
	Warning or Caution	Warns the user of the consequences of carrying out an action, and requires the user to give instruction on how to proceed. The box often contains Yes, No, and Cancel push buttons. This message box usually appears when data could be lost as a result of proceeding, for example, when the user deletes a file.
	Stop	Informs the user of a serious system-related deficiency that must be remedied before any action can be carried out.

Only the Stop message box should be system-modal. All other message boxes should be application-modal. See Chapter 8, "Dialog Boxes," for more information.

Note

Applications should use system-modal message boxes only when they encounter a condition that is the result of a system-related problem or when the condition may dramatically affect other applications running on the system.

9.3 The System Beep

Applications may use the system beep to signal a minor and obvious error, such as moving from the client area into a menu while pressing the mouse button.

9.4 Message-Box Captions

At a minimum, every message box should have a title bar and caption. The caption should include only the name of the application. For example, applications should not use "WARNING" or exclamation points in a caption. The text of the message box should clearly specify whether the message is a warning or error message. If the message box indicates a system-related error, the caption "System Error" should be used.

9.5 Push Buttons in Message Boxes

Message boxes should contain only push buttons. Other control types are inappropriate in message boxes. Windows provides the following sets of push buttons for use with message boxes:

Push Button	Description
OK	Used when no explicit action is required.
OK or Cancel	Used when OK is the default and the user has the option of canceling the command.
Yes or No	Used in an interrogatory message or warning when the default is No and the user has the option of proceeding with the command.
Yes, No, or Cancel	Used in an interrogatory message or warning when the user has the Yes and No options within the command, as well as the option to cancel the command and continue working with the application. Yes is the default.
Retry or Cancel	Used when a device error occurs, such as no disk in the disk drive. The user should cancel the command or retry the operation that causes the error. Applications must not let the user ignore any device error. Retry is the default.

The best or safest response should always be shown as the default button. If OK is used, it should be the default button.

Buttons should never be ambiguous or redundant. For example, if an error message requires only acknowledgment, the OK button is sufficient; a Cancel button is unnecessary.

9.6 Keyboard Use

Message boxes use the same keyboard support as other dialog boxes. The TAB key moves between buttons, the SPACEBAR activates the selected button, the ENTER key activates the default button, and the ESCAPE key cancels the message box (when the box contains a Cancel button).

9.7 Flashing for Attention

Any application that wants to display a message box when it or any of its child windows is not active should flash its title bar or icon, and beep to attract the attention of the user. The application should postpone displaying the message box until the user makes the application active.

On system-critical or potentially unrecoverable errors, the application should display a message box. No flashing is recommended.

9.8 Sample Message Boxes

The following are some sample message boxes that can be shared by applications that have standard File menus:

- If the user chooses the New or Open... command in the File menu after changes have been made during the current session, a message box with the following format should be displayed:

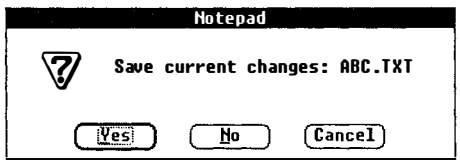


Figure 9.1 Dialog Box with Warning

- If the user chooses the Save As... command and then attempts to save the file under the name of an existing file, a message box with the following format should be displayed:

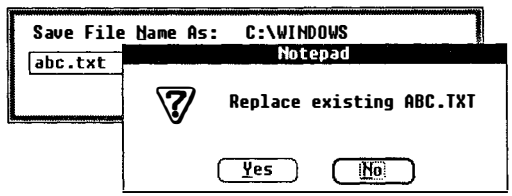


Figure 9.2 Save As... Dialog Box with Warning

- If the user invokes an application with the name of a data file that does not exist, a message box with the following format should be displayed:

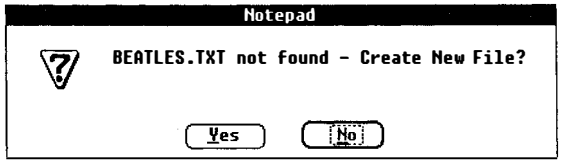


Figure 9.3 Dialog Box with Warning

If the user chooses the Yes button, the application should display the filename in the title bar and as the default field in a Save As... dialog box. If the user chooses the No button, the application should ignore the filename and continue operation as if no file had been specified.

If the user chooses Yes, and the application displays the filename, it should also attach the standard extension unless the user supplies one.

- If an application has a Print command, it should display a print message box whenever it sends a file to the printer. The caption should contain the application's name only. The dialog box should appear in the following format:



Figure 9.4 Dialog Box with Print Message

The name of the file appears in uppercase letters. The dialog box should also have a centered Cancel button.

9.9 Message-Box Text

Message-box text should describe the problem or error as clearly as possible. The message may suggest how to remedy the problem, but it should not be longer than two to three lines of text. Technical descriptions of the problem should not be used.

When writing text for a message box, do the following:

- Use "Cannot" instead of "Can not" or "Can't."
- Use "Not enough...to...." instead of "Disk is full, save not completed." The following example is correct:

  ```
  Not enough disk space to save PICTURE.MSP.
  ```

- Do not put the name of the application in the message.
- Do not imbed colons in the message.

  The following example is correct:

  ```
  Cannot read X.DAT.
  ```

  This example is incorrect:

  ```
  Cannot read: X.DAT.
  ```

Chapter 10
Cooperation

10.1 Introduction 83
10.2 Loading and Initialization 83
10.3 Yielding and Time-Critical Operations 83
10.4 The System Hourglass 84
10.5 Dead Air 84
10.6 Memory Use 84
10.7 File Use 85
10.8 Color Use 85

10.1 Introduction

Cooperation between applications is the key element in making Microsoft Windows a shared environment for applications. Applications should do the following:

- Use the CPU sparingly.

  Applications that perform extended computations should yield control whenever possible to let other applications complete their tasks.

- Use the screen display only when active.

  Inactive applications should alert the user with a flashing title bar or icon if they need to display a dialog box. The user then can activate the application.

- Use only as much memory as needed.

- Use discardable code segments.

  Applications that consist of small, discardable code segments are much easier to move and swap when available memory is short.

- Avoid fixed-memory objects.

10.2 Loading and Initialization

Applications should display their initial screen setup as soon as possible after loading. They should not leave a window empty for a long period of time.

10.3 Yielding and Time-Critical Operations

All applications should yield control whenever possible to allow other applications to access the CPU. For reasonably short operations where control is crucial, such as loading or saving a file, applications should complete the full operation as quickly as possible.

10.4 The System Hourglass

All applications should display the system-hourglass pointer when they begin lengthy or dead-air (pause in activity) operations. This informs the user that interaction with the application is not currently possible. The hourglass should be displayed with both keyboard and mouse interfaces. The hourglass pointer is shown as follows:



Figure 10.1 Hourglass Pointer

10.5 Dead Air

Unless waiting for user input, applications with the input focus must not present any pause in activity (dead air) that lasts longer than five seconds. Applications that require long delays in some operations should signal the user in some visible way that the operation is in progress.

10.6 Memory Use

Applications should cooperate in using memory. Applications should always determine whether there is sufficient space to operate in RAM as well as on the disk. If there is not, the application should display an appropriate message to let the user know that it is not possible to run in the current environment.

Applications should be designed to use only as much memory as is reasonably required to run the application.

10.7 File Use

Applications should always close files located on a floppy disk when not explicitly carrying out disk operations. Leaving these files open can cause a system failure. It is an application's responsibility to ensure that errors, such as the user's removing a disk from the drive without closing a file, do not cause the system to fail.

10.8 Color Use

An application should use color wherever appropriate; however, consideration also must be given to how the program looks on monochrome displays.

The system maintains default color settings that allow the user to adjust the color of various parts of the window. An application should use these default settings, unless it needs to regulate the color of a particular item. The application may override the default settings, but it must not change them.

Chapter 11
International Concerns

11.1 Introduction 89
11.2 Resource Files 89
11.3 Character Sets and Keyboards 89
11.4 Formats 90
11.5 Additional Software Design Tips 91

11.1 Introduction

Software companies are reaching out to international markets for increased sales. Competition from local products has increased dramatically. To reach these markets, applications developed for Microsoft Windows should be fully localized. You should design your product from the start with international markets in mind.

This chapter summarizes the localization concerns you should consider when writing an application. Following these instructions will make your application easy to translate into various languages.

11.2 Resource Files

If the following are used in an application, they should be placed in the application's resource file: menu items, menu bars, dialog boxes, pointers, icons, bitmaps, fonts, strings, and accelerators. This will allow a non-technical person to translate menus and messages and to resize and change the layout of dialog boxes. This person does not have to know anything about the source code of your application. Including this information in the resource file will dramatically cut the time required to bring your application to a new market.

11.3 Character Sets and Keyboards

Foreign countries use characters that are above 127 in the ANSI character set. These characters do not appear on US keyboards, and international keyboard layouts may differ from the layout of US keyboards. These differences can lead to difficulty when selling products in international markets.

Since Windows uses the ANSI character set, you should consider adding a conversion program to your application. In international markets, you will always be using characters above 127 from the 8-bit character set. Below 127, there is no difference between ASCII and ANSI. Above 127, you will encounter problems if you are loading a file created using the ASCII set into your Windows application. You should use the **OemToAnsi** and **AnsiToOem** functions to create a two-way conversion program. The conversion program also should recognize any file that it did not create. It should then invoke a dialog box to ask if the user wants to convert that file to the application's Windows format.

11.4 Formats

Different countries use different formats for numbers, currencies, time, dates, and measures, so porting an application to another country may be difficult. Windows provides you with a way to solve this problem. When requesting any of the above-mentioned values, your application should go to the *win.ini* file. *Win.ini* has an international section that contains all the values for the particular country Windows is installed for. At the time of installation, the **setup** program creates an international table in *win.ini* and puts all these values in it. The values entered are based on the user's choice of a particular country's keyboard. The following is a list of the values set in the [intl] section in *win.ini*:

Setting	Description
`iCountry=`	Country code, see your DOS manual for details.
`iDate=`	0 for mdy (month, day, year), 1 for dmy, 2 for ymd.
`iCurrency=`	0 for currency-symbol prefix, no separation. 1 for currency-symbol suffix, no separation. 2 for currency-symbol prefix, one-character separation. 3 for currency-symbol suffix, one-character separation.
`iDigits=`	Number of significant decimal digits in currency.
`iTime=`	0 for 12-hour clock, 1 for 24-hour clock.
`iLzero=`	0 for no leading zeros, 1 for leading zeros.
`s1159=`	Trailing string from 0:00 to 11:59.
`s2359=`	Trailing string from 12:00 to 23:59.
`sCurrency=`	Currency-symbol string.
`sThousand=`	Thousands-separator string.
`sDecimal=`	Decimal-separator string.
`sDate=`	Date-separator string.
`sTime=`	Time-separator string.
`sList=`	List-separator string.
`dialog=`	Is always yes. This enables the Country Settings command in Control Panel's Preferences menu.

These format settings must not be hardcoded. An application must get these values from the *win.ini* file.

The following are sample US settings from the [intl] section in *win.ini*:

```
[intl]
iCountry=1
iDate=0
iCurrency=0
iDigits=2
iTime=0
iLzero=0
s1159=AM
s2359=PM
sCurrency=$
sThousand=,
sDecimal=.
sDate=-
sTime=:
sList=,
dialog=yes
```

11.5 Additional Software Design Tips

To facilitate their use internationally, applications developed for Windows should do the following:

- An application should include the following in its resource file:

  Any text string that the user may see in the form of a screen message

  All dialog-box text and coordinates

  All child-window text and coordinates

- An application's dialog boxes and child windows should not take up more than 75% of the screen space. This restriction allows the enlargement of the dialog box and/or child window when translating the message strings.

- An application should never rely on the position of a string in a menu, dialog box, or child window. This saves you from having to modify the source code if the position of the string changes when it is translated.

- An application should never rely on the length of a string in a menu, dialog box, or child window. When translated, the length of the string may change.

- If a string contains variables, these variables should be able to be relocated anywhere within the body of the string. In different languages, the position of a word in a string may vary. The recommended way of representing variables is the use of "%%" as a placeholder. The "%%" placeholder then can be relocated by the translator anywhere within the string. The following system message is an example of this:

  ```
  COPYING FOO.DOC
  ```

  The variable is *foo.doc*. In French, this becomes:

  ```
  COPIE DE FOO.DOC EN COURS
  ```

  The variable *foo.doc* had to be relocated in the middle of this string. In the *msdos.rc* file, the string appears as:

  ```
  "COPYING %%"
  ```

- Accelerator keys should not be hardcoded. They should be able to be modified in the application's resource file.

- ALT+*key* combinations should be examined carefully. Some international keyboards use certain ALT+*key* combinations to enter certain characters.

- Application names, as well as filenames and extensions produced by the application, should not be hardcoded. They should be able to be modified in the application's resource file.

- An application should not use any OEM-specific virtual keys, with the exception of alphanumeric keys. Using these OEM-specific virtual keys prevents you from porting the application to international markets that use different keyboard layouts, unless you make substantial code modifications.

- An application should accept and output the full range of the 8-bit ANSI character set. Your application should not be dependent on the US character set or any specific country's character set.

- An application should support both metric and English measurement systems.

- If an application is doing any kind of sorting, it should account for the various international sorting schemes.

Chapter 12
Help Facility

12.1 Introduction 95
12.2 Access to Help 95
12.3 General Help 96
12.4 Context-Sensitive Help 97

12.1 Introduction

To allow users to request on-line information about your Microsoft Windows application, you may decide to include a help facility in the application. Two main types of help may be offered: general information and context-sensitive information. If on-line help is available, the user must be able to access it both from the keyboard and by using the mouse (or similar pointing device).

12.2 Access to Help

The help facility can be provided as either a push-button control in the menu bar, which responds with a dialog box, or a menu name, which causes a menu to appear.

If the help facility is included as a push-button control, a separator is drawn in the menu bar; the push button can be chosen with the mouse or the keyboard. However, the user cannot choose the push button by using the standard menu-selection keys. The application should implement the F1 key as the access key.

If the application includes the help facility as a menu name and provides a menu with it, no separator is drawn in the menu bar. In this case, the help facility can be accessed through the standard menu interface. The application should place the Help-menu name at the far-right side of the menu bar, and the F1 key should be implemented as the accelerator key for selecting the menu.

In the following example, the help facility is provided as a push-button control in the menu bar:

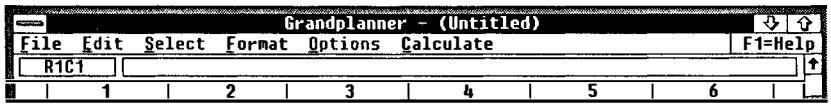


Figure 12.1 Help Facility as Push-Button Control

It is also possible to provide help within an application's dialog boxes. To include this type of help facility, a Help button should be added to the dialog boxes and the F1 key should be supported as the access key.

12.3 General Help

An application can provide help by allowing the user to obtain general help information. When the user chooses the help facility, the application should display a dialog box with the following form:

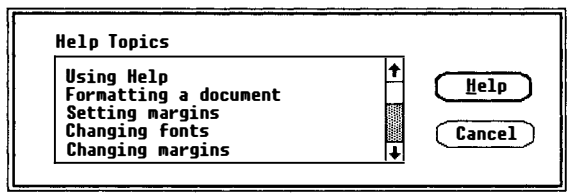


Figure 12.2 Help Topics Dialog Box

The dialog box contains a list box showing all of the topics the user can get information about. The user should be able to choose a topic by any of the following methods:

- Choose the Help button or press the ENTER key to choose the currently highlighted topic.
- Select a topic, then choose the Help button or press the ENTER key.
- Double-click a topic with the mouse.

The Help button should be the default button. If the list box contains more topics than can be shown at one time, the list box should have a scroll bar to allow the user to request additional topics. Choosing the Cancel button should remove the Help Topics dialog box and let the user continue working with the application.

When the user chooses a topic, the application should display a dialog box that contains help information about the selected topic. The dialog box should have the following form:

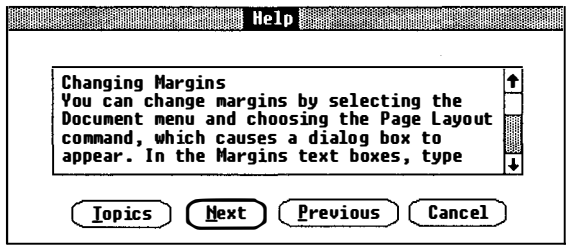


Figure 12.3 Help Information Dialog Box

If the user chooses the Topics button, the application should display the Help Topics dialog box. If the user chooses the Next button, the application should display the Help Information dialog box for the next topic on the list. The Next button is the default button. If the user chooses the Previous button, the application should display the Help Information dialog box for the previous topic. Choosing the Cancel button or pressing the ESCAPE key should return the user to the previous location in the application.

If more information is available than can be displayed in the list box, scroll bars should be provided to allow the user to view the rest of the text.

12.4 Context-Sensitive Help

An application can also provide help by implementing the F1 key as the key that places the application in "help mode." In this mode, the next item the user selects (menu, scroll bar, control, etc.) should display a Help Information dialog box that gives information about the selected item. This is known as context-sensitive help. The dialog box has the same format as the one shown in Figure 12.3 in the preceding section.

When the user presses the F1 key, the mouse pointer changes to a question mark. If no mouse is installed, a question-mark pointer is displayed in the middle of the application window. If the application is displaying a dialog box when the F1 key is pressed, a Help Information dialog box that lists information about the application's dialog box should be displayed.

When the user presses the ESCAPE key, the application should cancel the help mode and display the user's previous location in the application.

Chapter 13
Multiple-Document Interface

13.1 Introduction 101
13.2 The Main Window 101
13.3 The Child Window 103
13.4 Moving Within the Interface 106

13.1 Introduction

Some applications developed for Microsoft Windows may support the viewing of multiple windows from within an application's main window. The windows may provide different views of the same data or may contain different documents or sets of data. Such applications should use a multiple-document interface that allows the user to manage associated windows easily from the main window.

An application with multiple windows should use child windows. The child windows should be clipped so that they can reside inside the application's main window. The following figure shows an application using multiple windows:

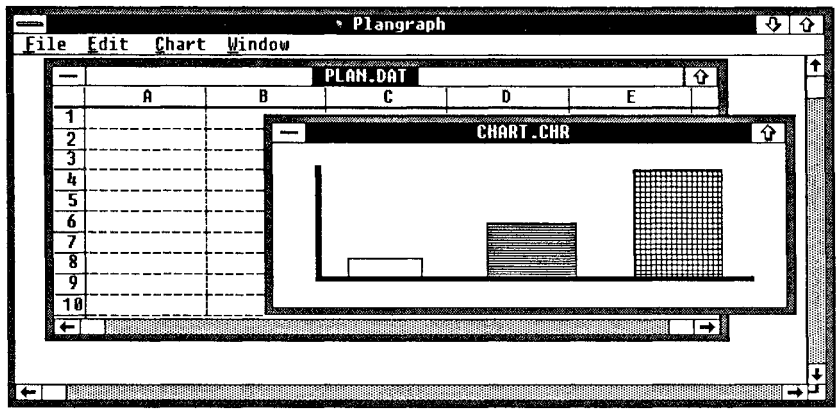


Figure 13.1 Multiple-Document Child Windows in a Main Window

13.2 The Main Window

The main window's Control menu affects the main window (window position and size, for example). The main window's title bar contains the name of the application, but no document name. The main window also contains the application's menus. The menus may vary according to the type of document the active child window contains. For example, if the user moves from a child window that contains a spreadsheet document to a child window that contains a chart document, spreadsheet-specific menus may be replaced with chart-specific menus. The menus can be selected and menu items chosen regardless of which of the application's windows is active.

The last entry on the menu bar is the Window menu. The commands in this menu include any commands that are used to manage child windows and a list of the different child-window names. The main window's name is also included on the Window menu. The user can choose from the list of window titles to select the next active window and bring it to the front. The following figure shows a typical Window menu:

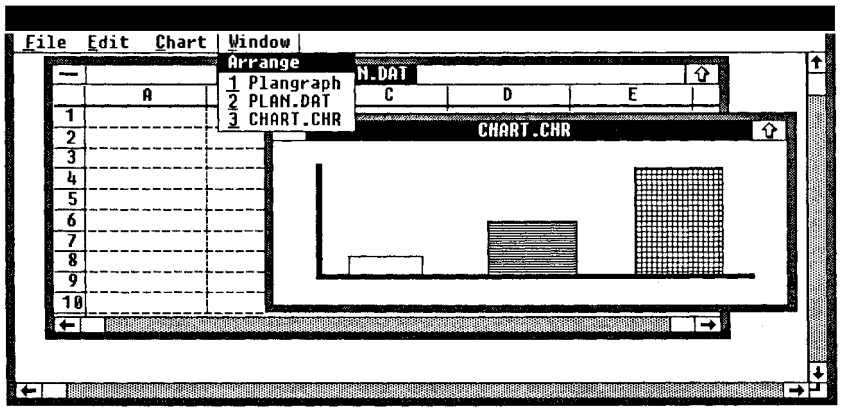


Figure 13.2 Window Menu

Commands that are used to manage child windows should be placed as a group in the Window menu before the list of window names and above a line separator. Such commands might include an Arrange command to organize the placement and size of the child windows within the main window. The resulting arrangement could be either overlapped or tiled, depending on the application. The previous illustration, Figure 13.2, shows a Window menu that contains the Arrange command.

In some applications, commands that are specific to the main window may be available only when that window is selected. Therefore, the main window's name is the first window title listed in the Window menu to make it easy to select regardless of the number of child windows. Applications that do not have commands that are specific to the main window may omit this menu item.

Each window title on the Window menu is preceded by a digit that serves as the menu item's mnemonic. This gives direct and consistent keyboard access to all menu items regardless of the names of the child windows. The application can use letters as the mnemonics if single-digit identifiers are unavailable.

13.3 The Child Window

The title bar of each child window contains the name of the document and a Control menu, but no other menus. If a single document is being viewed in more than one child window, a number is appended to the document name to distinguish between the windows, for example, "SAMPLE.TXT:1" and "SAMPLE.TXT:2."

The active child window is differentiated from other child windows by a change in the color or pattern of the title bar (in the same way that the active main window is differentiated from other main windows). When the application is active, the title bars of the main window and, if appropriate, the selected child window give active indications.

Windows records the active child window when the user begins working with another application. Switching back to the application by using ALT+ESCAPE reactivates that child window as well as the main window.

The Control-menu box in the child window's title bar differs from the main window's Control-menu box. A minus sign in the child window's Control-menu box represents the key combination (ALT, MINUS) used to access the child window's Control menu. The commands on the Control menu are the same for both the child window and the main window; however, the application should modify the corresponding accelerators in the child window, replacing the ALT key with the CONTROL key (see the list included later in this section). Any commands that are not appropriate are disabled (grayed). An example of a Control menu in a child window is shown in the following figure:

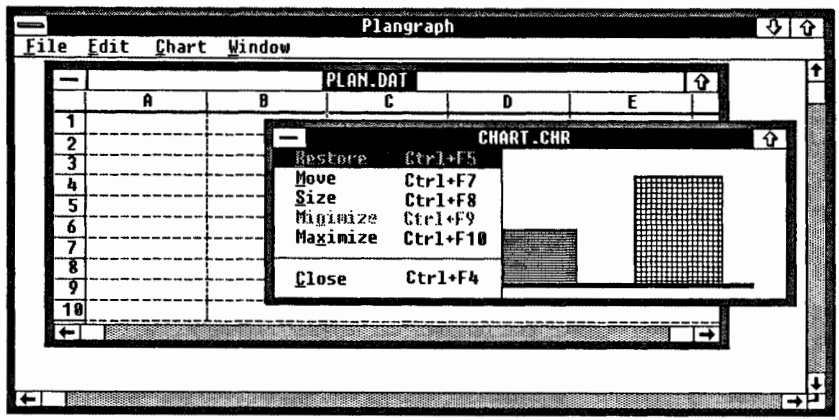


Figure 13.3 Child Window's Control Menu

Child windows should use the following key combinations to access Control-menu items:

Key	Action
CONTROL+F5	Invokes the Restore command.
CONTROL+F7	Invokes the Move command.
CONTROL+F8	Invokes the Size command.
CONTROL+F9	Invokes the Minimize command.
CONTROL+F10	Invokes the Maximize command.
CONTROL+F4	Invokes the Close command.

The size and position of a child window are controlled by commands from the child window's Control menu. These commands parallel functions provided in the main window's Control menu; however, the size and movement of the active child window are restricted to the size of the main window. Child windows also can be moved and sized with the mouse just as main windows can.

The Minimize command is usually disabled in a child window's Control menu, but if the application can create an icon for its child windows, this command may be used. Icons that represent child windows are displayed within the main window.

Choosing the Maximize command causes the document in the child window to fill the main window. (Double-clicking the title bar of the child window is a mouse shortcut for choosing the Maximize command.) The main window itself does not change in size. The title bar of the child

Choosing the Maximize command causes the document in the child window to fill the main window. (Double-clicking the title bar of the child window is a mouse shortcut for choosing the Maximize command.) The main window itself does not change in size. The title bar of the child window "slides under" the menu bar of the main window, and the caption of the main window changes to include the name of the document. The child window's Control menu is placed on the menu bar. (The main window's Control menu controls the operation of the main window even when the child window is enlarged to its maximum size.) The following illustration shows a maximized child window with its Control menu displayed:

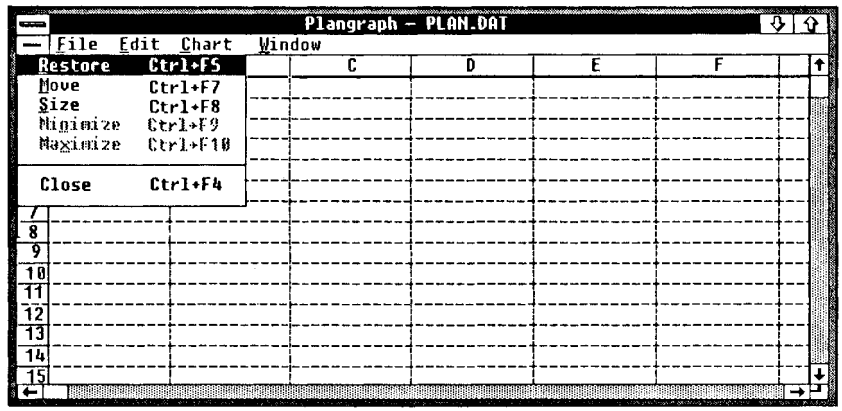


Figure 13.4 Maximized Child Window

If the active child window is enlarged to its maximum size, switching to another child window will automatically enlarge that window to its maximum size. This allows the user to quickly flip through documents. An application may choose to maximize the child window automatically as the default if there is only one open child window, such as when the user starts the application from the MS-DOS Executive window by using the document filename. Documents that were subsequently opened would take on the state of the active child window and, therefore, would be maximized if the previous working document were maximized.

Selecting the Restore command from the child window's Control menu restores the child window to its original size and location among the other child windows.

The child window's Close command destroys the child window after displaying a dialog box to confirm any required save operation for changes to the file. Double-clicking the child window's Control-menu box is a mouse shortcut for choosing the Close command.

13.4 Moving Within the Interface

The following key assignments are recommended for moving within the multiple-document interface:

Key	Recommended Action
F6	Selects the next active subdivision of a window (a child window within a document window), moving clockwise.
SHIFT+F6	Selects the next active subdivision of a window, moving counterclockwise.
CONTROL+F6	Selects the next active document window (a child window within the main window), moving from front to back.
SHIFT+CONTROL+F6	Selects the next active document window, moving from back to front.

Index

... (ellipsis), 18
%% placeholder, 92
~ (tilde), 60

About... command, 24
Accelerators
  ALT key, 29
  coding, 92
  CONTROL key, 29
  Control-menu commands, 30
  definition of, 29
  Edit-menu commands, 30
  function keys, 29
  guidelines, 29, 30
  help facility, 95
  international use, 89
  keynames, 29, 30
  SHIFT key, 29
Active window
  input focus, 43, 50
  multiple-document interface, 46, 101, 103
  selection highlighting, 35
ALT key, 29, 44, 92
ALT, MINUS, 103
ALT, SPACEBAR, 44
ALT+BACKSPACE, 30, 48
ALT+ESCAPE, 44, 103
ALT+F1, 46
ALT+F2, 46
ALT+F4, 30, 44
ALT+F5, 30, 44
ALT+F6, 44
ALT+F7, 30, 44
ALT+F8, 30, 44
ALT+F9, 30, 44
ALT+F10, 30, 44
ALT+*mnemonic*, 15
ALT+*mnemonic letter*, 71
ALT+SHIFT+TAB, 44
ALT+TAB, 44
ANSI character set, 89, 92
AnsiToOem function, 89
Application
  cooperation, 83
  international use, 89
  loading, 83
  names, 92
Application *(continued)*
  quitting, 20, 24
Application-modal dialog box, 65, 76
Arrange Command, 102
Arrow pointer, 55
ASCII character set, 89

BACKSPACE key, 30, 37, 48
Bitmap, 89
Box
  check, 66, 71
  Control menu, 5, 103
  dialog *See* Dialog box
  group, 72
  list, 61, 66
  Maximize, 5
  Minimize, 5
Button
  Cancel, 69, 77
  default, 68
  grayed, 69
  No, 69, 77
  OK, 69, 77
  push *See* Push button
  radio, 66, 67, 71, 72
  required, 69
  Retry, 77
  Yes, 69, 77

Cancel push button, 69, 77
Caption
  child window, 103
  main window, 6
  message box, 76
Caret, 37
Caution message, 75, 78
Character selection, 36, 37
Character sets, 89
Check box, 66, 69, 71
Checkmark, 17, 18
Child window *See also* Multiple-document interface
  control menu, 49
  definition of, 10
  dialog box *See* Dialog box
  flashing message, 77

Child window *See also* Multiple-document interface *(continued)*
illustration of, 10, 101
in multiple-document interface
Control menu, 103-105
icon, 104
illustration of, 101
menus, 101
size, 104
title bar, 103
international use, 91
keyboard interface, 46, 49
strings, 91
Clear command, 23, 31
Client area
definition of, 7
output, 7
parent window, 10, 101
scrolling *See* Scrolling
size, 7, 101
Clipboard
Edit-menu commands, 23
selection for, 35
Clipped output, 7, 8
Clock icon, 6
Close command
main window, 19, 20, 24, 30
child window, 104-105
Code
discardable segments, 83
source, 89, 91
swapping, 83
Color, 85
Command button *See* Push button
Command
About..., 24
accelerators, 29-31
Arrange, 102
button *See* Push button
Clear, 23, 31
Close
main window, 19, 20, 24, 30
child window, 104-105
Copy, 23, 30
Country Settings, 90
Cut, 23, 30
description of, 7
dialog box use, 65, 68, 69
disabled, 18
Exit, 20, 24
extended, 17, 18
grayed, 18
in File menu, 20-22
in Control menu, 19-20, 30 *See also* Control menu
in Edit menu, 23, 30-31
Command *(continued)*
in menu, 15, 17
independent, 17
input requests, 65
Maximize, 19, 30, 104
Minimize, 19, 30, 104
Move, 19, 30, 104
mutually exclusive, 18
New, 20, 21, 78
Open..., 20, 21, 78
Paste, 23, 31
Print, 79
Restore, 19, 30, 104, 105
Save, 20, 22
Save As..., 21, 22, 78
selection for, 23, 35
Size, 19, 30, 104
toggle, 16, 17, 18
Undo, 23, 30
Context-sensitive help, 97
CONTROL key, 29, 43
CONTROL+DELETE, 48
CONTROL+DIRECTION, 49
CONTROL+END, 49
CONTROL+ESCAPE, 44
CONTROL+F4, 49
CONTROL+F5, 49
CONTROL+F6, 43, 46
CONTROL+F7, 49
CONTROL+F8, 49
CONTROL+F9, 49
CONTROL+F10, 49
CONTROL+HOME, 48
CONTROL+INSERT, 30, 48
CONTROL+*mouse movement*, 54
CONTROL+PAGE DOWN, 9, 48
CONTROL+PAGE UP, 9, 48
CONTROL+SHIFT+DIRECTION, 49
Control menu
About... command, 24-25
accelerators, 30
child window, 49, 104
Close command, 20, 21
disabled item, 19
guidelines, 19
illustration of, 19
in dialog box, 65, 66
keyboard selection, 44
main window, 5
multiple-document interface, 103
Control-menu box, 5, 103
Controls
check box, 66, 69
default, 68-69, 70
design, 71
edit control, 61, 66, 72

Controls *(continued)*
grayed labels, 67
help facility, 95
illustration of, 67
inactive, 67
labels, 67
list box, 61-62, 66
message box, 76
mnemonics, 67, 71
moving between, 70
multiple, 69
names, 67
push button *See* Push button
radio button, 66, 69, 71, 72
required buttons, 69
static-control label, 68, 72
Conversion program, character sets, 89
Cooperation between applications, 83
Copy command, 23, 30
Country settings, 90
CPU sharing, 83
Cut command, 23, 30

Data files, 20 *See also* File *and* Filename
Dead-air operations, 84
Default
controls, 68-69, 70, 77
filename extension, 59
maximizing child window, 105
proposed filename, 61
Delayed operation, 84
DELETE key, 31, 37, 48
Device error, 77
Dialog box *See also* Message box
About... command, 24
appearance, 66, 71-72
application-modal, 65
canceling, 69, 70
caption, 66
check box, 66
Close command, 21
Control menu, 65, 66
controls *See* Controls
default button, 68
edit control, 61, 66
Exit command, 24
extended commands, 17, 18
font, 70
framed border, 66
guidelines, 65
help facility, 95
Help Information, 97
Help Topics, 96
illustrations of, 78

Dialog box *See also* Message box *(con inued)*
input focus, 68, 69, 70
international use, 89, 91
keyboard interface, 45, 68-71
list box, 61, 66
menu bar, 66
message box *See* Message box
mnemonics, 67, 71
modal, 65, 68, 75
modeless, 65
multiple controls, 69
multiple-document interface, 105
New command, 21
Open... command, 21
prompting for filename, 61, 62
push button, *See* Push button
radio button, 66, 69, 71, 72
Save As... command, 22, 59
Save command, 20, 59
scrolling, 71
size, 70
static-control label, 72
strings, 91
system-modal, 66, 76
title bar, 66
types, 65
Direct access *See* Mnemonics
DIRECTION keys, 45, 47, 49, 71, 72
Directory
list box, 61-62
prompting for name, 61
temporary files, 60
Disabled menu item, 18, 19, 24, 66
Disabled push button, 69
Discontinuous selection, 39, 54
Disk-drive list box, 60-61
Document window *See also* Child window
caption, 103
definition of, 11
moving input focus, 43

Edit control, 61, 66, 72
Edit menu, 23-24, 30
Editing selections, 37, 46-49
Ellipsis (...), 18
END key, 49
English measurements, 92
ENTER key, 47, 69, 70, 77
Error
device, 77
flashing message, 77
messages, 75, 76, 80
system, 66, 76, 85

Error *(continued)*
system beep, 76
ESCAPE key, 69, 70, 77, 97
Exit command, 20, 24
Extended command, 17, 18
Extended selection, 36, 37, 54
Extension, filename, 59

F1 key, 31, 46
F2 key, 46
F4 key, 30, 44, 49
F5 key, 30, 44, 49
F6 key, 43, 46
F7 key, 30, 44, 49
F8 key, 30, 44, 49
F9 key, 30, 44, 49
F10 key, 30, 44, 49
F11 key, 46
F12 key, 46
File *See also* Filename
international use, 91-92
naming, 6, 78
overwrite message, 78
prompting for name, 61
resource, 89, 91, 92
saving, 59, 78
temporary, 60
system failure, 85
use, 85
win.ini, 59, 90
File menu, 20-22, 78-79
Filename
default extension, 59
extension, 59, 60
guidelines, 59
in title bar, 6-7
international use, 92
null extension, 60
pathname, 59
prompting for, 61
proposed default, 61
searching PATH, 59
temporary files, 60
Fixed-memory objects, 83
Fixed-size window, 5
Flashing
bar, 37
bitmap, 37
caret, 37
error message, 77
icon, 77, 83
message box, 77
title bar, 77, 83
Font, 70
Formats, 90

Function keys *See also specific key*
acclerators, 29-31
recommended use, 43
help facility, 95
Functions
AnsiToOem, 89
GetTempFileName, 60
OemToAnsi, 89

GetTempFileName function, 60
Graphics *See* Non-text selection
Grayed menu item, 18, 19, 24, 66
Grayed push button, 69
Group box, 72
Guidelines
About... command, 24
accelerators, 29, 30
application, 1
child window, 10, 103
Control menu, 19
controls, 66, 67, 69, 71
dialog box, 65-66 *See also herein*
message box
disabled item, 18, 19
Edit menu, 23
File menu, 20, 21
filename, 59, 60, 61
help facility, 95
highlighting, 35
icon, 6
input focus, 50, 55
international use, 89
keyboard interface, 43, 46
main window, 5
menu, 15
menu item, 16
message box, 75-76
mouse interface, 53, 54, 55
multiple-document interface, 101
non-text selection, 38, 46, 54
push button, 69, 71
reserved keys, 44
scrolling, 8
selection, 35, 36, 37
shared environment, 83
split window, 10
temporary file, 60
text selection, 46, 54
toggle command, 17
translating, 89

Help facility
context-sensitive help, 97
F1 key, 31, 95

Help facility *(continued)*
general help, 96
Help Information dialog box, 97
Help mode, 97
Help Topics dialog box, 96
menu, 95
push-button controls, 95
Hexadecimal number, 60
Highlighting, 35, 37, 38
HOME key, 48
Hourglass pointer, 55, 84

I-beam pointer, 55
Icon
child window, 104
Clock, 6
flashing, 77
international use, 89
main window, 6
message box, 75
updating output, 6
Inactive *See* Grayed
Initilization requirements, 83
Input focus
delay in operation, 84
dialog box, 68, 69, 70
keyboard interface, 50
mouse interface, 55
Insertion-point selection, 36, 37
INSERT key, 48
Insert/replace mode, 48
Interface
keyboard *See* Keyboard interface
mouse *See* Mouse interface
multiple-document *See* Multiple-document interface
user, 1, 5
International use
application names, 92
filenames, 92
formats, 90
keyboard interface, 89, 92
measurements, 92
resource file, 89
settings, 90
sorting, 92
Item selection, 38, 46-49
Items in a menu *See* Menu item

Keyboard interface *See also specific key*
built-in, 43
control selection, 68, 71
Control-menu selection, 44
Keyboard interface *See also specific key (continued)*
dialog box, 68, 71
guidelines, 46
help facility, 95, 96, 97
input focus, 50
international use, 89, 92
menu names, 15
message box, 77
mnemonics
dialog box, 67-68, 71
menu item, 16, 20-21, 23, 24, 25
menu name, 15
moving input focus, 44
multiple-document interface, 102, 103, 104, 106
recommended keys, 46-49
reserved keys, 44-45
scrolling, 8, 46-49
selection, 35, 37, 46-49
trapping input, 50
Keys *See also specific key and* Keyboard interface
international use, 89
OEM-specific, 92
recommended use, 46-49
reserved for dialog box, 45
reserved for menu, 45
reserved for system, 44
shortcut *See* Accelerator

Labels on controls, 67
Line selection, 36, 37
Line separator, 16
List box, 61-62, 66-67
Loading applications, 83
Localizing applications, 89

Main window
caption, 6
child *See* Child window
client area *See* Client area
Control menu, 5
creation of, 5
format, 5
guidelines, 5
icon, 6
illustration of, 5, 7
Maximize box, 5
menu bar, 5
Minimize box, 5
multiple-document interface, 101
output *See* Output
scroll bar, 5

Main window *(continued)*
scrolling *See* Scrolling
split *See* Child window
title bar, 6
updating output *See* Output
Maximize box, 5
Maximize command, 30, 104
Measurements, 92
Memory use, 83, 84
Menu bar
definition of, 15
dialog box, 66
help facility, 95
illustration of, 5, 15
international use, 89
main window, 5
multiple-document interface, 101
Menu item
accelerators *See* Accelerators
checkmark, 17
commands, 17 *See also individual command*
Control-menu commands, 19
dialog-box use, 17, 18, 78-79
disabled, 18-19
Edit-menu commands, 23
extended commands, 17, 18
File-menu commands, 20-21
grayed, 18-19
illustration of, 16
international use, 89
keyboard interface, 43
message-box use, 78-79
names, 16, 18
reserved keys, 45
state setters, 17
toggle command, 16, 17, 18
types, 16
Menu mode, 44
Menu names, 15
Menu
accelerators *See* Accelerators
bar *See* Menu bar
checkmark *See* Toggle command
commands, 15 *See also individual command*
Control *See* Control menu
definition of, 15
disabled item, 18, 19
Edit, 23-24, 30
File, 20-22, 78-79
help facility, 95
illustration of, 16
item *See* Menu item
keyboard interface, 43, 45
mode, 44
Menu *(continued)*
names, 15
reserved keys, 45
strings, 91
Window, 102
Message box
application-modal, 76
captions, 76, 79
caution, 75, 78
default push button, 77
definition of, 75
flashing, 77
guidelines, 75, 79
illustrations of, 78
international use, 91
keyboard interface, 77
names, 76
note, 75
print, 79
push buttons, 76-77
save, 78
status, 75
stop, 75
system-modal, 76
text guidelines, 79
title bar, 76
types, 75-76
warning, 75, 76, 78
Messages
caution, 75, 78
error, 75-76, 80
guidelines, 80
international use, 91
note, 75
overwrite, 78
print, 79
save, 78
status, 75
stop, 75
system beep, 76
warning, 75, 78
names, 79
Metric measurements, 92
Minimize box, 5
Minimize command, 30, 104
MINUS key, 103
Mnemonics
definition of, 15
dialog box, 67, 71
menu item, 16
menu name, 15
multiple-document interface, 102
Modal dialog box, 65, 68, 75
Mode
help, 97
insert/replace, 48

Mode *(continued)*
menu, 44
Modeless dialog box, 65
Monochrome, 85
Mouse interface
built-in, 53
guidelines, 53
help facility, 95, 96
input capture, 55
multiple-document interface, 104, 105
pointers, 55
scrolling, 9
selecting, 35, 37, 54
terms, 53
Move command, 30, 104
Multiple-document interface
active window, 103
child window
Control menu, 103-105
icon, 104
illustration of, 101
menus, 101
size, 104
title bar, 103
Control menu, 103
definition of, 11, 101
dialog box, 105
document window *See* Document window
illustration of, 101
keyboard interface, 43, 49, 104, 105
main window, 101
menu bar, 101
mnemonics, 102
mouse interface, 104, 105
moving input focus, 46
moving within, 105
split window, 10
subdivision of a window *See* Subdivision of a window

Naming
accelerators, 29, 30
applications, 6, 92
controls, 67
edit controls, 72
files *See* Filename
menu items, 16
menus, 15
message boxes, 76, 79
push buttons, 71
windows, 6
New command, 20, 78
Next push button, 97
No push button, 69, 77
Non-text selection
definition of, 35
discontinuous, 39
enclosing rectangle, 35, 38
extended, 38
guidelines, 38
highlighting, 35, 38
item, 38
keyboard interface, 46
mouse interface, 54
region, 38
Note message, 75

OemToAnsi function, 89
OEM-specific keys, 92
OK button, 69, 77
On-line help, 95
Open... command, 20, 78
Operation
dead air, 84
delay in, 84
disk, 85
time critical, 83
Option button, 66
Output
clipped, 7
illustration of, 8
in child window, 10
in client area, 7
in split screen, 10
scrolling *See* Scrolling
updating client area, 7
updating icon, 6
Overlapping window, 5
Overwrite message, 78

PAGEDOWN key, 9, 48
PAGEUP key, 9, 48
Parent window *See also* Multiple-document interface
Close command, 20
definition of, 10
illustration of, 10, 101
Paste command, 23, 31
PATH variable, 59
Placeholder, %%, 92
Pointer
arrow, 55
hourglass, 84
I-beam, 55
question mark, 97
Previous push button, 97
Print command, 79

Print message, 79
Program
  conversion, 89
  Setup, 90
Punctuation selection, 36
Push button
  Cancel, 69, 77
  choosing, 69
  default, 68-69
  description of, 66
  design, 71
  disabled, 67, 69
  Help, 95
  in message boxes, 76
  inactive, 67, 69
  Next, 97
  No, 69, 77
  OK, 69, 77
  Previous, 97
  required, 69
  Retry, 77
  Topics, 97
  Yes, 69, 77

Question-mark pointer, 97
Quitting applications, 20, 24

Radio button, 66-67, 69, 72
RAM use, 84
Region selection, 38
Reserved keys
  dialog box, 45
  menu, 45
  system, 44
Resource file, 89, 91, 92
Restore command, 30, 104, 105
Retry push button, 77
Reverse video, 35, 38
Rules *See* Guidelines

Save As... command, 21, 22, 78
Save command, 20, 22
Save message, 78
Screen display, 83, 85
Scroll arrows, 9
Scroll bar
  dialog box, 71
  illustration of, 9
  main window, 5
  mouse interface, 9
Scroll box, 9
Scrolling
  clipped output, 7
Scrolling *(continued)*
  definition of, 8
  dialog box, 71
  guidelines, 8
  keyboard interface, 8, 48
  mouse interface, 9
  scroll bar, 9
Selection
  active window, 35
  character, 36, 37
  Clipboard use, 23, 35
  discontinuous, 39, 54
  editing text, 37
  Edit-menu commands, 23
  extended, 36, 37, 54
  flashing caret, 37
  guidelines, 35, 36, 37, 38
  highlighting, 35
  insertion point, 36, 37
  item, 38
  keyboard interface, 35, 37, 43, 46-49
  line, 36, 37
  mouse interface, 35, 37, 53
  non-text *See* Non-text selection
  punctuation, 36
  region, 38
  text *See* Text selection
  word, 36, 37, 54
Setup program, 90
Shared environment, 83
SHIFT key, 29, 43
SHIFT+CONTROL+END, 49
SHIFT+CONTROL+F6, 46
SHIFT+CONTROL+HOME, 48
SHIFT+DELETE, 30, 48
SHIFT+DIRECTION, 48
SHIFT+END, 49
SHIFT+ESCAPE, 44
SHIFT+F6, 43, 46
SHIFT+HOME, 48
SHIFT+INSERT, 31, 48
SHIFT+*mouse movement*, 54
SHIFT+PAGE DOWN, 48
SHIFT+PAGE UP, 48
SHIFT+TAB, 45, 47, 71
Shortcut keys *See* Accelerators
Size command, 30, 104
Size
  child window, 10, 104
  dialog box, 70
Sorting, 92
Source code, 89, 91
SPACEBAR, 44, 45, 68, 69, 77
Split window, 10 *See also* Child window
State setter, 17

Static-control label, 68, 72
Status message, 75
Stop message, 75
String
  in resource file, 89, 91
  length, 91
  variable, 92
Style guidelines *See* Guidelines
Subdivision of a window *See also* Multiple-document interface
  definition of, 11
  keyboard interface, 106
System
  beep, 68, 76
  error, 66, 76, 85
  failure, 85
  hourglass, 84
  reserved keys, 44
  state, 17
System-modal dialog box, 66, 76

TAB key, 44, 45, 47, 69, 71, 77
TEMP variable, 60
Temporary files, 60
Text editing *See* Text selection
Text selection *See also* Selection
  caret, 37
  character, 36, 37
  definition of, 35
  discontinuous, 39
  editing, 37, 46-49
  extended, 36, 37
  highlighting, 35, 37
  illustration of, 36
  insertion point, 36
  keyboard interface, 37, 46-49
  line, 36, 37
  mouse interface, 54
  punctuation, 36
  removing text, 37
  word, 36, 37
Tilde (~), 60
Time-critical operations, 83
Title bar
  child window, 103
  dialog box, 66
  flashing, 77
  guidelines, 6
  illustration of, 5, 7
  message box, 76
  multiple-document interface, 104
Toggle command, 16, 17, 18
Topics push button, 97
Translating applications, 89

Undo command, 23, 30
User interface, 1, 5

Variable
  international use, 92
  PATH, 59
  TEMP, 60

Warning message, 75, 76, 78
Win.ini file
  country settings, 90
  default filename extensions, 59
Window menu, 102
Window
  border, 5
  caption, 6, 7
  child *See* Child window
  client area *See* Client area
  cooperation, 83
  dialog box *See* Dialog box
  document *See* Document window
  fixed size, 5
  highlighting *See* Selection
  icon, 6
  international use, 89
  main *See* Main window
  naming, 6
  output *See* Output
  overlapping, 5
  parent *See* Parent window
  screen display, 83
  scrolling *See* Scrolling
  split, 10 *See also* Child window
  title bar, 6-7
  updating output *See* Output
Windows style, WS_GROUP, 72
Word selection, 36, 37, 54
WS_GROUP windows style, 72

Yes push button, 69, 77