



The Windows[®] Interface Guidelines for Software Design

Microsoft

PUBLISHED BY
Microsoft Press
A Division of Microsoft Corporation
One Microsoft Way
Redmond, Washington 98052-6399

Copyright © 1995 by Microsoft Corporation

All rights reserved. No part of the contents of this book may be reproduced or transmitted in any form or by any means without the written permission of the publisher.

Library of Congress Cataloging-in-Publication Data
The Windows interface guidelines for software design.

p. cm.

Includes index.

ISBN 1-55615-679-0

1. Microsoft Windows (Computer file) 2. User interfaces (Computer systems) 3. Computer software--Development. I. Microsoft Corporation.

QA76.76.W56W553 1995

005.265--dc20

95-330

CIP

Printed and bound in the United States of America.

1 2 3 4 5 6 7 8 9 QEQE 0 9 8 7 6 5

Distributed to the book trade in Canada by Macmillan of Canada, a division of Canada Publishing Corporation.

A CIP catalogue record for this book is available from the British Library.

Microsoft Press books are available through booksellers and distributors worldwide. For further information about international editions, contact your local Microsoft Corporation office. Or contact Microsoft Press International directly at fax (206) 936-7329.

Information in this document is subject to change without notice and does not represent a commitment on the part of Microsoft Corporation. Companies, names, and data used in examples herein are fictitious unless otherwise noted. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents or pending patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. The furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property rights.

Adobe, PostScript, and TIFF are trademarks of Adobe Systems, Inc. Apple and TrueType are registered trademarks of Apple Computer, Inc. Borland and Quattro are registered trademarks of Borland International, Inc. Frutiger is a registered trademark of Eltra Corporation. HP and LaserJet are registered trademarks of Hewlett-Packard Company. Backup was developed for Microsoft by Colorado Memory Systems, Inc., a division of Hewlett-Packard Company. HyperTerminal is a trademark of Hilgraeve, Inc. 1-2-3 and Lotus are registered trademarks of Lotus Development Corporation. Microsoft, Microsoft Press, Microsoft Press logo, MS, MS-DOS, PowerPoint, Visual Basic, Windows, Windows logo, and XENIX are registered trademarks and Windows NT is a trademark of Microsoft Corporation. Arial, Bodoni, Swing, and Times New Roman are registered trademarks of The Monotype Corporation PLC. Paintbrush is a trademark of Wordstar Atlanta Technology Center.

Introduction

Welcome to *The Windows Interface Guidelines for Software Design*, an indispensable guide to designing software that runs with the Microsoft® Windows® operating system. The design of your software's interface, more than anything else, affects how a user experiences your product. This guide promotes good interface design and visual and functional consistency within and across Windows-based applications.

What's New

Continuing the direction set by Microsoft OLE, the enhancements in the Windows user interface provide a design evolution from the basic and graphical to the more object oriented — that is, from an application-centered interface to a more data-centered one. In response, developers and designers may need to rethink the interface of their software — the basic components and the respective operations and properties that apply to them. This is important because, from a user's perspective, applications have become less the primary focus and more the engines behind the objects in the interface. Users can now interact with data without having to think about applications, allowing them to better concentrate on their tasks.

When adapting your existing Windows-based software, make certain you consider the following important design aspects:

- Title bar text and icons
- Property sheets
- Transfer model (including drag and drop)

Introduction

- Pop-up menus
- New controls
- Integration with the system
- Help interface
- OLE embedding and OLE linking
- Visual design of windows, controls, and icons
- Window management
- Presentation of minimized windows

These elements are covered in depth throughout this guide.

How to Use This Guide

This guide is intended for those who are designing and developing Windows-based software. It may also be appropriate for those interested in a better understanding of the Windows environment and the human-computer interface principles it supports. The content of the guide covers the following areas:

- Basic design principles and process — fundamental design philosophy, assumptions about human behavior, design methodology, and concepts embodied in the interface.
- Interface elements — descriptive information about the various components in the interface as well as when and how to use them.
- Design details — specific information about the details of effective design and style when using the elements of the interface.
- Additional information — summary and quick reference information, a bibliography, a comprehensive word list in numerous languages to assist in product localization, and a glossary.

You can choose to display a desktop toolbar when the user runs a specific application, or by creating a separate application and including a shortcut icon to it in the system's Startup folder. Preferably set the initial size and position of your desktop toolbar so that it does not interfere with other desktop toolbars or the taskbar. However, the system does support multiple desktop toolbars to be docked along the same edge of the display screen. When docking on the same edge as the taskbar, the system places the taskbar on the outermost edge.

Your desktop toolbar can include any type of control. A desktop toolbar can also be a drag and drop target. Follow the recommendations outlined in this guide for supporting appropriate interaction.

Full-Screen Display

Although the taskbar and application desktop toolbars normally constrain or clip windows displayed on the screen, you can define a window to the full extent of the display screen. Because this is not the typical form of interaction, only consider using full-screen display for very special circumstances, such as a slide presentation, and only when the user explicitly chooses a command for this purpose. Make certain you provide an easy way for the user to return to normal display viewing. For example, you can display an on-screen button when the user moves the pointer that restores the display when the user clicks it. In addition, keyboard interfaces, like ALT+TAB and ESC, should automatically restore the display.

Remember that desktop toolbars, including the taskbar, should support auto-hide options that allow the user to configure them to reduce their visual impact on the screen. Consider whether this auto-hide capability may be sufficient before designing your application to require a full-screen presentation. Advising the user to close or hide desktop toolbars may provide you with sufficient space without having to use the full display screen.

Recycle Bin Integration



The Recycle Bin provides a repository for deleted files. If your application includes a facility for deleting files, support the Recycle Bin interface. You can also support deletion to the Recycle Bin for nonfile objects by first formatting the deleted data as a file by writing it to a temporary file and then calling the system functions that support the Recycle Bin.



The **SHFileOperation** function supports deletion using the Recycle Bin interface. For more information about this function, see the documentation included in the Win32 SDK.

Control Panel Integration



The Windows Control Panel includes special objects that let users configure aspects of the system. Your application can add Control Panel objects or add property pages to the property sheets of existing Control Panel objects.

Adding Control Panel Objects

You can create your own Control Panel objects. Most Control Panel objects supply only a single secondary window, typically a property sheet. Define your Control Panel object to represent a concrete object rather than an abstract idea.

Every Control Panel object is a dynamic-link library. To ensure that the dynamic-link library can be automatically loaded by the system, set the file's extension to `.CPL` and install it in the Windows System directory.



The system automatically caches information about Control Panel objects in order to provide quick user access, provided that the Control Panel object supports the correct system interfaces. For more information about developing Control Panel objects, see the documentation included in the Win32 SDK.

Adding to the Passwords Object

The Passwords object in Control Panel supplies a property sheet that allows the user to set security options and manage passwords for all password-protected services in the system. The Passwords object also allows you to add the name of a password-protected service to the object's list of services and use the Windows login password for all password-protected services in the system.

When you add your service to the Passwords object, the name of the service appears in the Select Password dialog box that appears when the user chooses Change Other Passwords. The user can then change the password for the service by selecting the name and filling in the resulting dialog box. The name of your service also appears in the Change Windows Password dialog box; the name appears with a check box next to it. By setting the check box option, the user chooses to keep the password for the service identical to the Windows login password. Similarly, the user can disassociate the service from the Windows login password by toggling the check box setting off.

To add your service to the Passwords object, register your service under the **HKEY_LOCAL_MACHINE** key.

HKEY_LOCAL_MACHINE

System

CurrentControlSet

Control

PwdProvider

Provider Name Value Name = Value

You can also add a page to the property sheet of the Passwords object to support other security-related services that the user can set as property values. Add a property page if your application provides security-related functionality beyond simple activation and changing of passwords. To add a property page, follow the conventions for adding shell extensions.



For more information about registering your password service, see the documentation included in the Win32 SDK.

Plug and Play Support

Plug and Play is a feature of Windows that, with little or no user intervention, automatically installs and configures drivers when their corresponding hardware peripherals are plugged into a PC. This feature applies to peripherals designed according to the Plug and Play specification. Supporting and appropriately adapting to Plug and Play hardware change can make your application easier to use. Following are some examples of supporting Plug and Play:

- Resizing your windows and toolbars relevant to screen size changes.
- Prompting users to shut down and save their data when the system issues a low power warning.
- Warning users about open network files when undocking their computers.
- Saving and closing files appropriately when users eject or remove removable media or storage devices or when network connections are broken.

System Settings and Notification

The system provides standard metrics and settings for user interface aspects, such as colors, fonts, border width, and drag rectangle (used to detect the start of a drag operation). The system also notifies running applications when its settings change. When your application starts up, query the system to set your application's user interface to match the system parameters to ensure visual and operational consistency. Also, design your application to adjust itself appropriately when the system notifies it of changes to these settings.



The **GetSystemMetrics**, **GetSysColor**, and **SystemParametersInfo** functions and the **WM_SETTINGSCCHANGE** message are important to consider when supporting standard system settings. For more information about these system interfaces, see the documentation included in the Win32 SDK.