

Microsoft[®] Windows[™] 3.1

Programmer's Reference

Volume 3

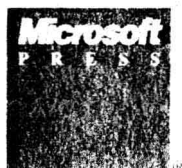
Messages, Structures, and Macros

Microsoft WindowsTM 3.1

Programmer's Reference

Volume 3

Messages, Structures, and Macros



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

Copyright ©1987-1992 Microsoft Corporation. All rights reserved.

Information in this document is subject to change without notice and does not represent a commitment on the part of Microsoft Corporation. The software, which includes information contained in any databases, described in this document is furnished under a license agreement or nondisclosure agreement and may be used or copied only in accordance with the terms of that agreement. It is against the law to copy the software except as specifically allowed in the license or nondisclosure agreement. No part of this manual may be reproduced in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose without the express written permission of Microsoft Corporation.

Library of Congress Cataloging-in-Publication Data

Microsoft Windows programmer's reference.

p. cm.

Includes indexes.

Contents: v. 1. Overview -- v. 2. Functions -- v. 3. Messages, structures, macros -- v. 4. Resources.

ISBN 1-55615-453-4 (v. 1). -- ISBN 1-55615-463-1 (v. 2). -- ISBN 1-55615-464-X (v. 3). -- ISBN 1-55615-494-1 (v. 4)

1. Microsoft Windows (Computer program) I. Microsoft Corporation.

QA76.76.W56M532 1992

005.43--dc20

91-34199

CIP

Printed and bound in the United States of America.

1 2 3 4 5 6 7 8 9 MLML 7 6 5 4 3 2

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

Distributed to the book trade outside the United States and Canada by Penguin Books Ltd.

Penguin Books Ltd., Harmondsworth, Middlesex, England
Penguin Books Australia Ltd., Ringwood, Victoria, Australia
Penguin Books N.Z. Ltd., 182-190 Wairau Road, Auckland 10, New Zealand

British Cataloging-in-Publication Data available.

ITC Zapf Chancery and ITC Zapf Dingbats fonts. Copyright ©1991 International Typeface Corporation. All rights reserved.
Copyright ©1981 Linotype AG and/or its subsidiaries. All rights reserved. Helvetica, Palatino, New Century Schoolbook, Times, and Times Roman typefont data is the property of Linotype or its licensors.
Arial and Times New Roman fonts. Copyright ©1991 Monotype Corporation PLC. All rights reserved.

Adobe® and PostScript® are registered trademarks of Adobe Systems, Inc. Apple® Macintosh® and TrueType® are registered trademarks of Apple Computer, Inc. PANOSF™ is a trademark of ElseWare Corporation. Epson® and FX® are registered trademarks of Epson America, Inc. Hewlett-Packard® HP® LaserJet® and PCL® are registered trademarks of Hewlett-Packard Company. IBM® is a registered trademark of International Business Machines Corporation. ITC Zapf Chancery® and ITC Zapf Dingbats® are registered trademarks of International Typeface Corporation. Helvetica® New Century Schoolbook® Palatino® Times® and Times Roman® are registered trademarks of Linotype AG and/or its subsidiaries. CodeView® Microsoft® MS® MS-DOS® and QuickC® are registered trademarks and QuickBasic™ and Windows™ are trademarks of Microsoft Corporation. Arial® and Times New Roman® are registered trademarks of Monotype Corporation PLC. Okidata® is a registered trademark of Oki America, Inc.

The Symbol fonts provided with Windows version 3.1 are based on the CG Times font, a product of AGFA Compugraphic Division of Agfa Corporation.

U.S. Patent No. 4974159

Document No. PC28917-0492

Introduction

This manual, *Microsoft Windows Programmer's Reference, Volume 3*, describes the data types, messages, structures, macros, and printer escapes supported by the Microsoft® Windows™ operating system. In addition, dynamic data exchange (DDE) transactions, File Manager events, raster-operation codes, virtual-key codes, and character tables are presented.

Organization of This Manual

Following are brief descriptions of the chapters and appendixes in this manual:

- Chapter 1, "Data Types," describes the keywords that define the size and meaning of parameter and return values associated with the Windows application programming interface (API).
- Chapter 2, "Messages," describes formatted window messages, through which the Windows operating system communicates with applications, and notification messages, which notify a control's parent window of actions that occur within the control.
- Chapter 3, "Structures," defines the data structures associated with the functions that are part of the Windows API.
- Chapter 4, "Macros," describes the purpose and defines the parameters of macros used to help manipulate data in Windows applications.
- Chapter 5, "Printer Escapes," lists printer escapes for the Windows operating system.
- Chapter 6, "Dynamic Data Exchange Transactions," describes the transactions sent by the Dynamic Data Exchange Management Library (DDEML) to an application's dynamic data exchange (DDE) callback function. The transactions notify the application of DDE activity that affects the application.
- Chapter 7, "File Manager Events and Messages," provides descriptions of the events and menu commands File Manager sends to communicate with a File Manager extension dynamic-link library (DLL). The chapter also describes messages the DLL can send File Manager to retrieve information.
- Chapter 8, "Control Panel Messages," lists the messages Control Panel sends to communicate with a Control Panel DLL.

- Chapter 9, “Common Dialog Box Messages,” describes the messages a common dialog box can send to notify applications that the user has made or changed a selection in the dialog box.
- Chapter 10, “Installable Driver Messages,” lists the messages the Windows operating system sends to notify installable drivers about specific events.
- Appendix A, “Binary and Ternary Raster-Operation Codes,” lists and describes the binary and ternary raster operations used by the graphics device interface (GDI).
- Appendix B, “Virtual-Key Codes,” shows the symbolic constant names, hexadecimal values, and keyboard equivalents for Windows virtual-key codes.
- Appendix C, “Character Tables,” illustrates the Windows character set, the Symbol character set, and the OEM character set used by the Windows operating system.

Document Conventions

The following conventions are used throughout this manual to define syntax:

Convention	Meaning
Bold text	Denotes a term or character to be typed literally, such as a resource-definition statement or function name (MENU or CreateWindow), a command, or a command-line option (/nod). You must type these terms exactly as shown.
<i>Italic text</i>	Denotes a placeholder or variable: You must provide the actual value. For example, the statement SetCursorPos (<i>X</i> , <i>Y</i>) requires you to substitute values for the <i>X</i> and <i>Y</i> parameters.
[]	Enclose optional parameters.
	Separates an either/or choice.
...	Specifies that the preceding item may be repeated.
BEGIN	Represents an omitted portion of a sample application.
.	
.	
END	

In addition, certain text conventions are used to help you understand this material:

Convention	Meaning
SMALL CAPITALS	Indicate the names of keys, key sequences, and key combinations—for example, ALT+SPACEBAR.
FULL CAPITALS	Indicate filenames and paths, type names and most structure names (which are also bold), and constants.
monospace	Sets off code examples and shows syntax spacing.

Contents

	Introduction	v
	Organization of This Manual.....	v
	Document Conventions	vi
Chapter 1	Data Types	1
Chapter 2	Messages	11
	2.1 Window Messages.....	14
	2.2 Notification Messages	213
Chapter 3	Structures	229
Chapter 4	Macros	429
Chapter 5	Printer Escapes	449
Chapter 6	Dynamic Data Exchange Transactions	513
Chapter 7	File Manager Events and Messages	529
	7.1 File Manager Events.....	531
	7.2 File Manager Messages	534
Chapter 8	Control Panel Messages	541
Chapter 9	Common Dialog Box Messages	551
Chapter 10	Installable Driver Messages	559
Appendix A	Binary and Ternary Raster-Operation Codes	571
	A.1 Binary Raster Operations	573
	A.2 Ternary Raster Operations	576

Appendix B Virtual-Key Codes	587
Appendix C Character Sets	593
C.1 ANSI Character Set	596
C.2 Symbol Character Set	597
C.3 OEM Character Set	598
Index	599

Data Types

Chapter 1

Alphabetic Reference 3

The data types in this chapter are keywords that define the size and meaning of parameters and return values associated with functions for the Microsoft Windows operating system, version 3.1. The following table contains character, integer, and Boolean types; pointer types; and handles. The character, integer, and Boolean types are common to most C compilers. Most of the pointer-type names begin with a prefix of P, N (for near pointers), or LP (for long pointers). A near pointer accesses data within the current data segment, and a long pointer contains a 32-bit segment:offset value. A Windows application uses a handle to refer to a resource that has been loaded into memory. Windows provides access to these resources through internally maintained tables that contain individual entries for each handle. Each entry in the handle table contains the address of the resource and a means of identifying the resource type.

The Windows data types are defined in the following table:

Type	Definition
ABORTPROC	32-bit pointer to an AbortProc callback function.
ATOM	16-bit value used as an atom handle.
BOOL	16-bit Boolean value.
BYTE	8-bit unsigned integer. Use LPBYTE to create 32-bit pointers. Use PBYTE to create pointers that match the compiler memory model.
CATCHBUF[9]	18-byte buffer used by the Catch function.
COLORREF	32-bit value used as a color value.
DLGPROC	32-bit pointer to a dialog box procedure.
DWORD	32-bit unsigned integer or a segment:offset address. Use LPDWORD to create 32-bit pointers. Use PDWORD to create pointers that match the compiler memory model.
FARPROC	32-bit pointer to a function.
FNCALLBACK	32-bit value identifying the DdeCallback function. Use PFNCALLBACK to create pointers that match the compiler memory model.
FONENUMPROC	32-bit pointer to an EnumFontsProc callback function.
GLOBALHANDLE	16-bit value used as a handle to a global memory object.
GNOTIFYPROC	32-bit pointer to a NotifyProc callback function.
GOBJENUMPROC	32-bit pointer to a EnumObjectsProc callback function.
GRAYSTRINGPROC	32-bit pointer to a GrayStringProc callback function.

Type	Definition
HANDLE	16-bit value used as a general handle. Use LPHANDLE to create 32-bit pointers. Use SPHANDLE to create 16-bit pointers. Use PHANDLE to create pointers that match the compiler memory model.
HCURSOR	16-bit value used as a cursor handle.
HFILE	16-bit value used as a file handle.
HGDIOBJ	16-bit value used as a graphics device interface (GDI) object handle.
HGLOBAL	16-bit value used as a handle to a global memory object.
HHOOK	32-bit value used as a hook handle.
HKEY	32-bit value used as a handle to a key in the registration database. Use PHKEY to create 32-bit pointers.
HLOCAL	16-bit value used as a handle to a local memory object.
HMODULE	16-bit value used as a module handle.
HORJECT	16-bit value used as a handle to an OLE object.
HWND	16-bit value used as a handle to a window.
HOOKPROC	32-bit pointer to a hook procedure.
HRSRC	16-bit value used as a resource handle.
LHCLIENTDOC	32-bit value used as a handle to an OLE client document.
LHSERVER	32-bit value used as a handle to an OLE server.
LHSERVERDOC	32-bit value used as a handle to an OLE server document.
LINEDDAPROC	32-bit pointer to a LineDDAProc callback function.
LOCALHANDLE	16-bit value used as a handle to a local memory object.
LONG	32-bit signed integer.
LPABC	32-bit pointer to an ABC structure.
LPARAM	32-bit signed value passed as a parameter to a window procedure or callback function.
LPBI	32-bit pointer to a BANDINFOSTRUCT structure.
LPBITMAP	32-bit pointer to a BITMAP structure. Use NPBITMAP to create 16-bit pointers. Use PBITMAP to create pointers that match the compiler memory model.

Type	Definition
LPBITMAPCOREHEADER	32-bit pointer to a BITMAPCOREHEADER structure. Use PBITMAPCOREHEADER to create pointers that match the compiler memory model.
LPBITMAPCOREINFO	32-bit pointer to a BITMAPCOREINFO structure. Use PBITMAPCOREINFO to create pointers that match the compiler memory model.
LPBITMAPFILEHEADER	32-bit pointer to a BITMAPFILEHEADER structure. Use PBITMAPFILEHEADER to create pointers that match the compiler memory model.
LPBITMAPINFO	32-bit pointer to a BITMAPINFO structure. Use PBITMAPINFO to create pointers that match the compiler memory model.
LPBITMAPINFOHEADER	32-bit pointer to a BITMAPINFOHEADER structure. Use PBITMAPINFOHEADER to create pointers that match the compiler memory model.
LP_CATCHBUF	32-bit pointer to a CATCHBUF array.
LP_CBT_CREATEWND	32-bit pointer to a CBT_CREATEWND structure.
LPCHOOSECOLOR	32-bit pointer to a CHOOSECOLOR structure.
LPCHOOSEFONT	32-bit pointer to a CHOOSEFONT structure.
LPCLIENTCREATESTRUCT	32-bit pointer to a CLIENTCREATESTRUCT structure.
LPCOMPAREITEMSTRUCT	32-bit pointer to a COMPAREITEMSTRUCT structure. Use PCOMPAREITEMSTRUCT to create pointers that match the compiler memory model.
LP_CPLINFO	32-bit pointer to a CPLINFO structure. Use PCPLINFO to create pointers that match the compiler memory model.
LP_CREATESTRUCT	32-bit pointer to a CREATESTRUCT structure.
LPCTSTR	32-bit pointer to a nonmodifiable character string.
LPCTLINFO	32-bit pointer to a CTLINFO structure. Use PCTLINFO to create pointers that match the compiler memory model.
LPCTLSTYLE	32-bit pointer to a CTLSTYLE structure. Use PCTLSTYLE to create pointers that match the compiler memory model.
LPDCB	32-bit pointer to a DCB structure.
LPDEBUGHOOKINFO	32-bit pointer to a DEBUGHOOKINFO structure.

Type	Definition
LPDELETEITEMSTRUCT	32-bit pointer to a DELETEITEMSTRUCT structure. Use PDELETEITEMSTRUCT to create pointers that match the compiler memory model.
LPDEVMODE	32-bit pointer to a DEVMODE structure. Use NPDEVMODE to create 16-bit pointers. Use PDEVMODE to create pointers that match the compiler memory model.
LPDEVNAMES	32-bit pointer to a DEVNAMES structure.
LPDOCINFO	32-bit pointer to a DOCINFO structure.
LPDRAWITEMSTRUCT	32-bit pointer to a DRAWITEMSTRUCT structure. Use PDRAWITEMSTRUCT to create pointers that match the compiler memory model.
LPDRIVERINFOSTRUCT	32-bit pointer to a DRIVERINFOSTRUCT structure.
LPDRVCONFIGINFO	32-bit pointer to a DRVCONFIGINFO structure. Use PDRVCONFIGINFO to create pointers that match the compiler memory model.
LPEVENTMSG	32-bit pointer to a EVENTMSG structure. Use NPEVENTMSG to create 16-bit pointers. Use PEVENTMSG to create pointers that match the compiler memory model.
LPDRIVERINFOSTRUCT	32-bit pointer to a DRIVERINFOSTRUCT structure.
LPFINDREPLACE	32-bit pointer to a FINDREPLACE structure.
LPFMS_GETDRIVEINFO	32-bit pointer to a FMS_GETDRIVEINFO structure.
LPFMS_GETFILESEL	32-bit pointer to a FMS_GETFILESEL structure.
LPFMS_LOAD	32-bit pointer to a FMS_LOAD structure.
LPHANDLETABLE	32-bit pointer to a HANDLETABLE structure. Use PHANDLETABLE to create pointers that match the compiler memory model.
LPHELPWININFO	32-bit pointer to a HELPWININFO structure. Use PHELPWININFO to create pointers that match the compiler memory model.
LPINT	32-bit pointer to a 16-bit signed value. Use PINT to create pointers that match the compiler memory model.
LPKERNINGPAIR	32-bit pointer to a KERNINGPAIR structure.

Type	Definition
LPLOGBRUSH	32-bit pointer to a LOGBRUSH structure. Use NPLOGBRUSH to create 16-bit pointers. Use PLOGBRUSH to create pointers that match the compiler memory model.
LPLOGFONT	32-bit pointer to a LOGFONT structure. Use NPLOGFONT to create 16-bit pointers. Use PLOGFONT to create pointers that match the compiler memory model.
LPLOGPALETTE	32-bit pointer to a LOGPALETTE structure. Use NPLOGPALETTE to create 16-bit pointers. Use PLOGPALETTE to create pointers that match the compiler memory model.
LPLOGPEN	32-bit pointer to a LOGPEN structure. Use NPLOGPEN to create 16-bit pointers. Use PLOGPEN to create pointers that match the compiler memory model.
LPLONG	32-bit pointer to a 32-bit signed integer. Use PLONG to create pointers that match the compiler memory model.
LPMAT2	32-bit pointer to a MAT2 structure.
LPMDICREATESTRUCT	32-bit pointer to an MDICREATESTRUCT structure.
LPMEASUREITEMSTRUCT	32-bit pointer to a MEASUREITEMSTRUCT structure. Use PMEASUREITEMSTRUCT to create pointers that match the compiler memory model.
LPMETAFILEPICT	32-bit pointer to a METAFILEPICT structure.
LPMETARECORD	32-bit pointer to a METARECORD structure. Use PMETARECORD to create pointers that match the compiler memory model.
LPMOUSEHOOKSTRUCT	32-bit pointer to a MOUSEHOOKSTRUCT structure.
LPMSG	32-bit pointer to an MSG structure. Use NPMSG to create 16-bit pointers. Use PMSG to create pointers that match the compiler memory model.
LPNCCALCSIZE_PARAMS	32-bit pointer to an NCCALCSIZE_PARAMS structure.
LPNEWCPINFO	32-bit pointer to an NEWCPINFO structure. Use PNEWCPINFO to create pointers that match the compiler memory model.

Type	Definition
LNEWTEXTMETRIC	32-bit pointer to a NEWTEXTMETRIC structure. Use NPNEWTEXTMETRIC to create 16-bit pointers. Use PNEWTEXTMETRIC to create pointers that match the compiler memory model.
LPOFSTRUCT	32-bit pointer to an OFSTRUCT structure. Use NPOFSTRUCT to create 16-bit pointers. Use POFSTRUCT to create pointers that match the compiler memory model.
LPOLECLIENT	32-bit pointer to OLECLIENT structure.
LPOLECLIENTVTBL	32-bit pointer to OLECLIENTVTBL structure.
LPOLEOBJECT	32-bit pointer to OLEOBJECT structure.
LPOLEOBJECTVTBL	32-bit pointer to OLEOBJECTVTBL structure.
LPOLESERVER	32-bit pointer to OLESERVER structure.
LPOLESERVERDOC	32-bit pointer to OLESERVERDOC structure.
LPOLESERVERDOCVTBL	32-bit pointer to OLESERVERDOCVTBL structure.
LPOLESERVERVTBL	32-bit pointer to OLESERVERVTBL structure.
LPOLESTREAM	32-bit pointer to OLESTREAM structure.
LPOLESTREAMVTBL	32-bit pointer to OLESTREAMVTBL structure.
LPOLETARGETDEVICE	32-bit pointer to OLETARGETDEVICE structure.
LPOPENFILENAME	32-bit pointer to OPENFILENAME structure.
LPOUTLINETEXTMETRIC	32-bit pointer to an OUTLINETEXTMETRIC structure.
LPPAINTSTRUCT	32-bit pointer to a PAINSTRUCT structure. Use NPPAINTSTRUCT to create 16-bit pointers. Use PPAINTSTRUCT to create pointers that match the compiler memory model.
LPPALETTEENTRY	32-bit pointer to a PALETTEENTRY structure.
LPPPOINT	32-bit pointer to a POINT structure. Use NPPOINT to create 16-bit pointers. Use PPOINT to create pointers that match the compiler memory model.
LPPPOINTFX	32-bit pointer to a POINTFX structure.
LPPRINTDLG	32-bit pointer to a PRINTDLG structure.
LPRASTERIZER_STATUS	32-bit pointer to a RASTERIZER_STATUS structure.
LPRECT	32-bit pointer to a RECT structure. Use NPRECT to create 16-bit pointers. Use PRECT to create pointers that match the compiler memory model.

Type	Definition
LPRGBQUAD	32-bit pointer to a RGBQUAD structure.
LPRGBTRIPLE	32-bit pointer to a RGBTRIPLE structure.
LPSEGINFO	32-bit pointer to a SEGINFO structure.
LPSIZE	32-bit pointer to a SIZE structure. Use NPSIZE to create 16-bit pointers. Use PSIZE to create pointers that match the compiler memory model.
LPSTR	32-bit pointer to a character string. Use NPSTR to create 16-bit pointers. Use PSTR to create pointers that match the compiler memory model.
LPTEXTMETRIC	32-bit pointer to a TEXTMETRIC structure. Use NPTEXTMETRIC to create 16-bit pointers. Use PTEXTMETRIC to create pointers that match the compiler memory model.
LPTTPOLYCURVE	32-bit pointer to a TTPOLYCURVE structure.
LPTTPOLYGONHEADER	32-bit pointer to a TTPOLYGONHEADER structure.
LPVOID	32-bit pointer to an unspecified type.
LPWINDOWPLACEMENT	32-bit pointer to a WINDOWPLACEMENT structure. Use PWINDOWPLACEMENT to create pointers that match the compiler memory model.
LPWINDOWPOS	32-bit pointer to a WINDOWPOS structure.
LPWNDCLASS	32-bit pointer to a WNDCLASS structure. Use NPWNDCLASS to create 16-bit pointers. Use PWNDCLASS to create pointers that match the compiler memory model.
LPWORD	32-bit pointer to a 16-bit unsigned value. Use PWORD to create pointers that match the compiler memory model.
LRESULT	32-bit signed value returned from a window procedure or callback function.
MFENUMPROC	32-bit pointer to an EnumMetaFileProc callback function.
NEARPROC	16-bit pointer to a function.
OLECLIPFORMAT	16-bit value used as a standard clipboard format.
PATTERN	Equivalent to the LOGBRUSH structure. Use LPPATTERN to create 32-bit pointers. Use NPPATTERN to create 16-bit pointers. Use PPATTERN to create pointers that match the compiler memory model.
PCONVCONTEXT	32-bit pointer to a CONVCONTEXT structure.
PCONVINFO	32-bit pointer to a CONVINFO structure.