

Microsoft[®] Windows[™] 3.1

Programmer's Reference

Volume 4

Resources

MicrosoftTM Windows 3.1

Programmer's Reference

Volume 4

Resources



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, 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. TrueType® is a registered trademark of Apple Computer, Inc. Epson® is a registered trademark of Epson America, Inc. Hewlett-Packard®, HP®, and LaserJet® are registered trademarks of Hewlett-Packard Company. ITC Zapf Chancery® and ITC Zapf Dingbats® are registered trademarks of International Typeface Corporation. CodeView®, Microsoft®, MS®, and MS-DOS® are registered trademarks and Windows™ is a trademark of Microsoft Corporation. Arial® and Times New Roman® are registered trademarks of Monotype Corporation PLC.

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. PC30211-0492

Introduction

The Microsoft® Windows™ operating system is a single-user personal-computer operating system that employs a graphical user interface. This graphical interface uses a variety of resources that must be constructed in specific formats. This manual, the *Microsoft Windows Programmer's Reference, Volume 4*, describes these resource formats and executable-file headers.

Part 1, "File Formats," describes the formats for the principal types of files used by Windows applications. The chapters in this part provide detailed information about the file formats, as well as about the MS-DOS® and Windows executable-file headers and resource formats within executable files. Topics include the formats for the following types of files: graphics, clipboard, font, group, calendar, object-module, library, symbol, and metafile.

Part 2, "Tools Reference," provides detailed reference information about the statements, commands, and macros for tools used to create and maintain Windows resources. Topics include resource-definition statements, assembly-language macros, and help statements and macros. Each entry in this section gives the purpose of the command or macro; its complete syntax, parameters, and return values; and cross-references to related commands or macros. Many entries also include expanded comments on the use of the command or macro.

How to Use This Manual

This manual describes the Windows resource-file formats in individual chapters. Each chapter describes the format that should be used for the type of file associated with a specific resource or activity. For example, the chapter on graphics file formats describes the formats used with bitmap, icon, and cursor resource files.

Each chapter has two parts: a general description of the file type and a detailed presentation of the format. Chapters in Part 2, "Tools Reference," describe only the file format and not the tool. For more information about the associated tools, see *Microsoft Windows Programming Tools*.

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), an MS-DOS 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, most type and structure names (which are also bold), and constants.
monospace	Sets off code examples and shows syntax spacing.

Contents

Introduction	ix
How to Use This Manual.....	ix
Document Conventions	x

Part 1 File Formats

Chapter 1 Graphics File Formats	3
1.1 Bitmap-File Formats.....	5
1.1.1 Bitmap-File Structures.....	5
1.1.2 Bitmap Compression	6
1.1.3 Bitmap Example	9
1.2 Icon-Resource File Format.....	10
1.2.1 Icon Directory	10
1.2.2 Icon Image	11
1.2.3 Windows Icon Selection.....	12
1.3 Cursor-Resource File Format.....	13
1.3.1 Cursor Directory	13
1.3.2 Cursor Image	14
1.3.3 Windows Cursor Selection	16
Chapter 2 Clipboard File Format	17
2.1 Clipboard-File Header.....	19
2.2 Clipboard-File Structure.....	19
Chapter 3 Metafile Format	21
3.1 Metafile Header	23
3.2 Typical Metafile Record.....	24
3.3 Placeable Windows Metafiles	26
3.4 Guidelines for Windows Metafiles	27
3.5 Sample of Metafile Program Output	28
3.6 Function-Specific Metafile Records	29

Chapter 4	Font File Format	47
4.1	Organization of a Font File.....	49
4.2	Font-File Structure.....	49
4.3	Version-Specific Glyph Tables	56
Chapter 5	Group File Format	59
5.1	Organization of a Group File.....	61
5.2	Group-File Structures	61
5.2.1	Group-File Header	61
5.2.2	Item Data.....	63
5.2.3	Tag Data	64
Chapter 6	Executable-File Header Format	67
6.1	MS-DOS Header	69
6.2	Windows Header	70
6.2.1	Information Block	71
6.2.2	Segment Table	74
6.2.3	Resource Table.....	75
6.2.4	Resident-Name Table	78
6.2.5	Module-Reference Table	78
6.2.6	Imported-Name Table.....	78
6.2.7	Entry Table.....	78
6.2.8	Nonresident-Name Table.....	80
6.3	Code Segments and Relocation Data	80
Chapter 7	Resource Formats Within Executable Files.....	83
7.1	Icon Resource	85
7.2	Icon-Directory Resource	85
7.3	Cursor Resource	86
7.4	Cursor-Directory Resource.....	86
7.5	Menu Resource	87
7.5.1	Menu Header.....	87
7.5.2	Pop-up Menu Item	88
7.5.3	Normal Menu Item.....	88
7.5.4	Combined Menu Items	89
7.6	Dialog Box Resource.....	90
7.6.1	Dialog Box Header	90
7.6.2	Control Data.....	92

7.7	Bitmap Resource	93
7.8	Font Resource	94
7.8.1	Font-Directory Data	94
7.8.2	Font-Component Data	95
7.9	String-Table Resources	96
7.10	Accelerator Resource	96
7.11	Name-Table Resource	97
7.11.1	Name-Table Entry	97
7.12	Version-Information Resource	98
7.12.1	Root Block	99
7.12.2	Variable Information Block	100
7.12.3	String Information Block	102
7.12.4	Language-Specific Blocks	102
Chapter 8	Write File Format	105
8.1	Write-File Header	107
8.2	Text and Pictures	108
8.2.1	Text	108
8.2.2	Pictures	108
8.3	Formatting	110
8.3.1	Characters and Paragraphs	110
8.3.2	Footnotes	113
8.3.3	Sections	113
8.3.4	Font Table	115
Chapter 9	Calendar File Format	117
9.1	Calendar-File Header	119
9.2	Date Descriptors	120
9.3	Day-Specific Information	121
9.4	Appointment-Specific Information	121
Chapter 10	Windows Object-Module Format	123
10.1	Object-Module Format Records	125
10.2	Record Reference	126
Chapter 11	Library and Import-Library Formats	133
11.1	Organization of Libraries	135
11.2	Dictionary	135
11.2.1	Collision Resolution	136
11.3	Record Reference	137

Chapter 12	Symbol File Format	141
12.1	Map Definitions	143
12.2	Segment Definitions	145
12.3	Symbol Definitions	147
12.4	Constant Definitions	148
12.5	Line Definitions	148
12.5.1	LINEDEF Structure	148
12.5.2	LINEINF Structure	150

Part 2 Tools Reference

Chapter 13	Resource-Definition Statements	153
13.1	Alphabetic Reference	155
Chapter 14	Assembly-Language Macros	223
14.1	Creating Assembly-Language Windows Applications	225
14.1.1	Specifying a Memory Model	226
14.1.2	Selecting a Calling Convention	227
14.1.3	Enabling the Windows Prolog/Epilog Option	227
14.1.4	Including the CMACROS.INC File	228
14.1.5	Creating the Application Entry Point	228
14.1.6	Declaring Callback Functions	229
14.1.7	Linking with Libraries	229
14.1.8	Enabling Stack Checking	229
14.2	Cmacro Groups	230
14.2.1	Segment Macros	230
14.2.2	Storage-Allocation Macros	231
14.2.3	Function Macros	231
14.2.4	Call Macros	231
14.2.5	Special-Definition Macros	232
14.2.6	Error Macros	232
14.3	Using the Cmacros	233
14.3.1	Overriding Types	233
14.3.2	Symbol Redefinition	233
14.3.3	Sample Cmacros Function	234
14.4	Alphabetic Reference	235

Chapter 15	Windows Help Statements and Macros	253
15.1	Help Statement Syntax	255
15.2	Help Macro Syntax	256
15.3	Help Statement Reference	257
15.4	Help Macro Reference	302
Index		331

File Formats

Part 1

Graphics File Formats

Chapter 1

1.1	Bitmap-File Formats	5
1.1.1	Bitmap-File Structures	5
1.1.2	Bitmap Compression	6
1.1.2.1	Compression of 8-Bits-per-Pixel Bitmaps	7
1.1.2.2	Compression of 4-Bits-per-Pixel Bitmaps	8
1.1.3	Bitmap Example	9
1.2	Icon-Resource File Format	10
1.2.1	Icon Directory	10
1.2.2	Icon Image	11
1.2.3	Windows Icon Selection	12
1.3	Cursor-Resource File Format	13
1.3.1	Cursor Directory	13
1.3.2	Cursor Image	14
1.3.3	Windows Cursor Selection	16

This chapter describes the graphics-file formats used by the Microsoft Windows operating system. Graphics files include bitmap files, icon-resource files, and cursor-resource files.

1.1 Bitmap-File Formats

Windows bitmap files are stored in a device-independent bitmap (DIB) format that allows Windows to display the bitmap on any type of display device. The term “device independent” means that the bitmap specifies pixel color in a form independent of the method used by a display to represent color. The default filename extension of a Windows DIB file is .BMP.

1.1.1 Bitmap-File Structures

Each bitmap file contains a bitmap-file header, a bitmap-information header, a color table, and an array of bytes that defines the bitmap bits. The file has the following form:

```

BITMAPFILEHEADER bmfh;
BITMAPINFOHEADER bmih;
RGBQUAD          aColors[];
BYTE              aBitmapBits[];

```

The bitmap-file header contains information about the type, size, and layout of a device-independent bitmap file. The header is defined as a **BITMAPFILEHEADER** structure.

The bitmap-information header, defined as a **BITMAPINFOHEADER** structure, specifies the dimensions, compression type, and color format for the bitmap.

The color table, defined as an array of **RGBQUAD** structures, contains as many elements as there are colors in the bitmap. The color table is not present for bitmaps with 24 color bits because each pixel is represented by 24-bit red-green-blue (RGB) values in the actual bitmap data area. The colors in the table should appear in order of importance. This helps a display driver render a bitmap on a device that cannot display as many colors as there are in the bitmap. If the DIB is in Windows version 3.0 or later format, the driver can use the **biClrImportant** member of the **BITMAPINFOHEADER** structure to determine which colors are important.

The **BITMAPINFO** structure can be used to represent a combined bitmap-information header and color table.

The bitmap bits, immediately following the color table, consist of an array of **BYTE** values representing consecutive rows, or “scan lines,” of the bitmap. Each scan line consists of consecutive bytes representing the pixels in the scan line, in left-to-right order. The number of bytes representing a scan line depends on the

color format and the width, in pixels, of the bitmap. If necessary, a scan line must be zero-padded to end on a 32-bit boundary. However, segment boundaries can appear anywhere in the bitmap. The scan lines in the bitmap are stored from bottom up. This means that the first byte in the array represents the pixels in the lower-left corner of the bitmap and the last byte represents the pixels in the upper-right corner.

The **biBitCount** member of the **BITMAPINFOHEADER** structure determines the number of bits that define each pixel and the maximum number of colors in the bitmap. These members can have any of the following values:

Value	Meaning
1	Bitmap is monochrome and the color table contains two entries. Each bit in the bitmap array represents a pixel. If the bit is clear, the pixel is displayed with the color of the first entry in the color table. If the bit is set, the pixel has the color of the second entry in the table.
4	Bitmap has a maximum of 16 colors. Each pixel in the bitmap is represented by a 4-bit index into the color table. For example, if the first byte in the bitmap is 0x1F, the byte represents two pixels. The first pixel contains the color in the second table entry, and the second pixel contains the color in the sixteenth table entry.
8	Bitmap has a maximum of 256 colors. Each pixel in the bitmap is represented by a 1-byte index into the color table. For example, if the first byte in the bitmap is 0x1F, the first pixel has the color of the thirty-second table entry.
24	Bitmap has a maximum of 2^{24} colors. The bmiColors (or bmcColors) member is NULL, and each 3-byte sequence in the bitmap array represents the relative intensities of red, green, and blue, respectively, for a pixel.

The **biClrUsed** member of the **BITMAPINFOHEADER** structure specifies the number of color indexes in the color table actually used by the bitmap. If the **biClrUsed** member is set to zero, the bitmap uses the maximum number of colors corresponding to the value of the **biBitCount** member.

An alternative form of bitmap file uses the **BITMAPCOREINFO**, **BITMAPCOREHEADER**, and **RGBTRIPLE** structures.

For a full description of the bitmap structures, see the *Microsoft Windows Programmer's Reference, Volume 3*.

1.1.2 Bitmap Compression

Windows versions 3.0 and later support run-length encoded (RLE) formats for compressing bitmaps that use 4 bits per pixel and 8 bits per pixel. Compression reduces the disk and memory storage required for a bitmap.

1.1.2.1 Compression of 8-Bits-per-Pixel Bitmaps

When the **biCompression** member of the **BITMAPINFOHEADER** structure is set to **BI_RLE8**, the DIB is compressed using a run-length encoded format for a 256-color bitmap. This format uses two modes: encoded mode and absolute mode. Both modes can occur anywhere throughout a single bitmap.

Encoded Mode A unit of information in encoded mode consists of two bytes. The first byte specifies the number of consecutive pixels to be drawn using the color index contained in the second byte.

The first byte of the pair can be set to zero to indicate an escape that denotes the end of a line, the end of the bitmap, or a delta. The interpretation of the escape depends on the value of the second byte of the pair, which must be in the range 0x00 through 0x02. Following are the meanings of the escape values that can be used in the second byte:

Second byte	Meaning
0	End of line.
1	End of bitmap.
2	Delta. The two bytes following the escape contain unsigned values indicating the horizontal and vertical offsets of the next pixel from the current position.

Absolute Mode Absolute mode is signaled by the first byte in the pair being set to zero and the second byte to a value between 0x03 and 0xFF. The second byte represents the number of bytes that follow, each of which contains the color index of a single pixel. Each run must be aligned on a word boundary.

Following is an example of an 8-bit RLE bitmap (the two-digit hexadecimal values in the second column represent a color index for a single pixel):

Compressed data	Expanded data
03 04	04 04 04
05 06	06 06 06 06 06
00 03 45 56 67 00	45 56 67
02 78	78 78
00 02 05 01	Move 5 right and 1 down
02 78	78 78
00 00	End of line
09 1E	1E 1E 1E 1E 1E 1E 1E 1E 1E
00 01	End of RLE bitmap