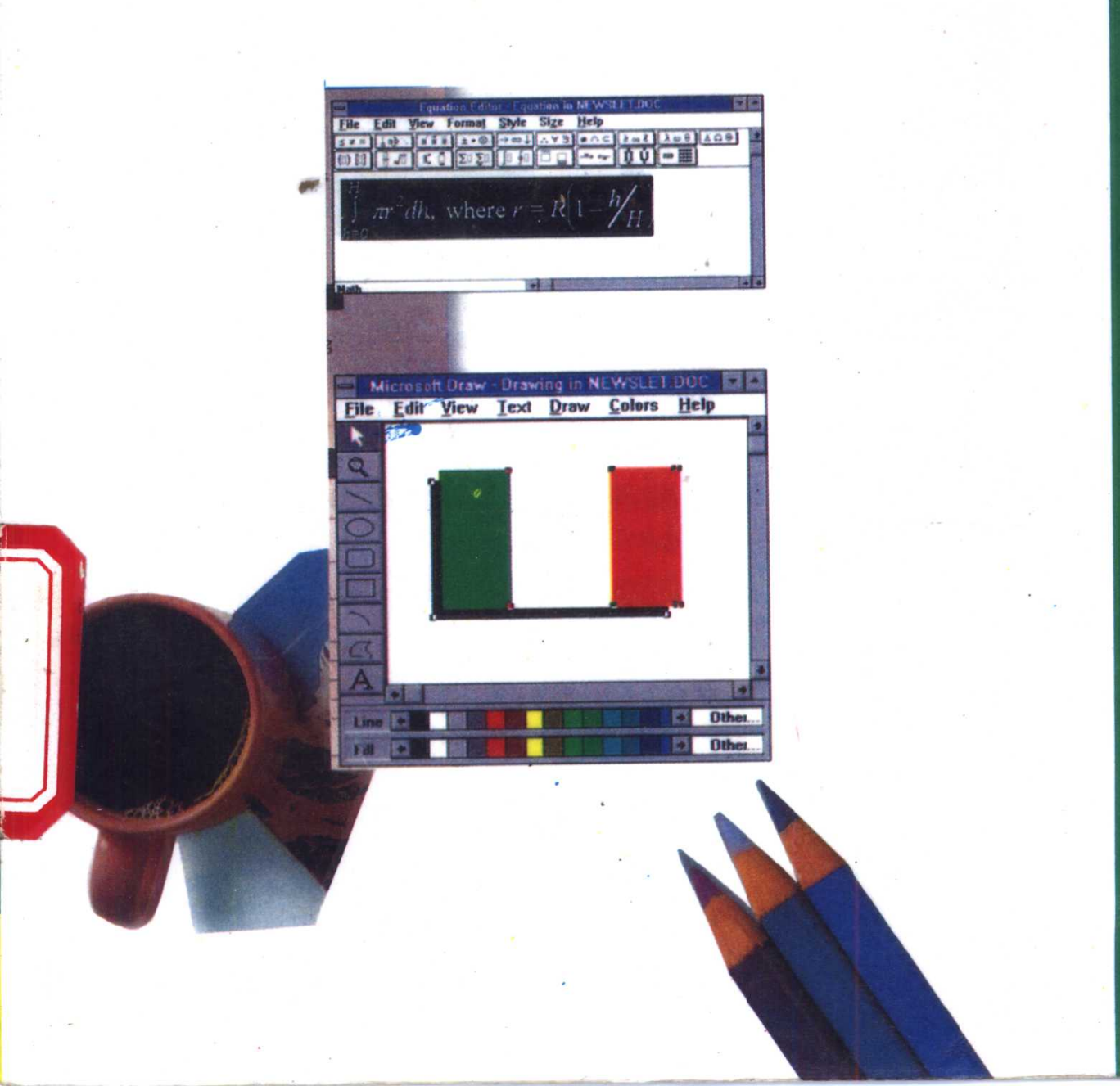
Windows技术丛书之一

高级编程指南

李　力　　编著

海洋出版社

1



Windows 技术丛书之一

高级编程指南

李　力　　编著
吴　强　　审校

海洋出版社

1993 年·北京

内容提要

本书介绍了 Windows 数据类型、消息、结构、宏指令。Microsoft Windows 操作系统下的打印机转义，以及动态数据交换事务(DDE)、文件管理事件、光栅操作码、虚拟键代码。

欲购本书的用户可直接与北京 8721 信箱联系，邮编：100081，电话：2562329。

（京）新登字 087 号

责任编辑　闫世尊

Windows 软件技术丛书之一

高级编程指南

[illegible]　编著

吴　强　审校

海洋出版社出版(北京复兴门外大街 1 号)

海洋出版社发行　兰空印刷厂印刷

开本 787×1092 1/16　印张：24.5　字数：573 千字

1993 年 12 月第一版　1993 年 12 月第一次印刷

印数 1—5000 册

*

ISBN 7—5027—3821—5/TP. 234　　定价：193.00 元/套(6 册)

前　言

本书由十章和两个附录组成，具体内容包括：

第一章"数据类型"介绍了定义窗口应用程序接口中出现的参数、返回值的大小和意义的关键词。

第二章"消息"介绍了格式化的窗口消息，通过这种消息，窗口操作系统和应用程序得以交流。还有通知消息，它控制着发生在控制器内的父窗口的操作。

第三章"结构"定义了 Windows 应用程序接口(API)中部分函数的数据结构。

第四章"宏指令"定义了窗口应用中控制数据的宏指令参数及目标。

第五章"打印机转义"列出了窗口操作系统的打印机转义。

第六章"动态数据交换事务"描述了由动态数据交换管理库发给应用程序的动态数据交换(DDE)调回函数的事务，它阐明了 DDE 活动对应用程序的影响。

第七章"文件管理和消息"提供了文件管理与文件管理分程序动态连接程序进行联系时发送的事件和菜单命令。描述了 DLL 发给文件管理程序检索信息的消息。

第八章"控制板消息"列出了控制板发出的与 DLL 控制板进行联络的消息。

第九章"公共对话框消息"描述了由公共对话框发出的，用来通知用户已经在对话框内作出选择改变选择。

第十章"可安装驱动程序消息"列出了 Windows 操作系统发送的，用来通知可安装驱动器特殊事件的消息。

附录 A"二元和三元光栅操作码"描述图象设备窗口(GDI)所使用的二元和三元光栅操作码。

附录 B"虚拟键代码"显示了 Windows 虚拟键代码的符号常量名、十六进制值和等效键盘符。

目　录

第一章　数据类型

本章中所列出的数据类型是用来定义大小、参数意义和 Windows 函数返回值的关键字，下表所列数据类型适用于 Microsoft Windows 操作系统版本 3.1。表中包括字符、整数、布尔型、指针型及指针。字符、整数和布尔型与编译程序大致相同，大部分指针类型都是以 P（短指针）、N（短指针）或 LPI（长指针）做前缀开头的。短指针指向当前数据段内的数据，长指针包括一个 32 位数据段／偏移量。Windows 应用程序使用指针访问内存的资源，Windows 通过一些表提供对这些资源的访问，这些表中包括每个指针的专用项，每个专用项包括资源的地址和标识资源类型的方法。

Windows 数据类型定义如下表：

类型	定义
ABORTPROC	指向AbortProc退出函数的32位指针。
ATOM	作为原子指针的16位值。
BOOL	16位布尔值。
BYTE	8位无符号整数，使用LPBYTE创建32位；使用PBYTE 创建匹配编译器内存模型的指针。
CATCHBUF9	Catch函数使用的18位缓存器。
COLORREF	作为颜色值使用的32位值。
DLGPROC	对话框函数的32位指针。
DWORD	无符号32位整数或一个分段／偏移地址。使用PDWORD 创建 32 位指针；使用 PDWORD 创建匹配编译器内存模型的指针。
FARPROC	指向函数的指针。
FNCALLBACK	标识函数 DdeCallback的 32位值。使用 PFNCALLBACK 创建匹配编译器内存模型的指针。
FONTENUMPROC	指向EnumFontsProc函数的32位指针。
GLOBALHANDLE	整个内存的指针，是由系统堆栈中分配出来的内存块的 16 位变址。
GNOTIFYPROC	指向函数NotifyProc的32位指针。
GOBJENUMPROC	指向EnumObjectsProc函数的32位指针。
GRAYSTRINGPROC	指向函数GrayStringProc的32位指针。
HANDLE	一般指针，表示一个判定程序数据的16位变址。使用 LPHANDLE 创建 32 位指针，使用 SPHANDLE 创建 16 位指针，使用 PHANDLE 创建匹配编译器内存模式的指针。

HCURSOR	光标指针，是资源表项的16位变化。
HFILE	文件指针的16位变址。
HGDIOBJ	图形设备接口(GDI)目标指针的16位变址。
HGLOBAL	全局内存目标指针的16位变址。
HHOOK	钩指针的32位变址。
HKEY	登记数据库中键指针的32位变址。使用PHKEY创建32位指针。
HLOCAL	局部内存目标指针的16位变址。
HMODULE	模块指针的16位变址。
HOBJECT	OLE对象指针的16位变址。
HWND	窗口指针的16位变址。
HOOKPROC	钩函数的32位指针。
HRSRC	资源指针的16位变址。
LHCLIENTDOC	OLE用户文档指针的的32位变址。
LHSERVER	OLE服务器指针的32位变址。
LHSERVERDOC	OLE服务器文档指针的32位变址。
LINEDDAPROC	LINEDDAProc函数的32位指针。
LOCALHANDLE	局部内存目标指针的16位指针。
LONG	32位无符号整数。
LPABC	ABC结构的32位指针。
LPARAM	32位有符号值，作为参数传给窗口函数或返回函数。
LPBI	指向结构BANDINFOSTRUCT的32位指针。
LPBITMAP	BITMAP结构的32位指针。使用NPBITMAP创建16位指针，使用PBITMAP创建匹配编译内存模型的指针。
LPBITMAPCOREHEADER	指向BITMAPCOREHEADER数据结构的长指针。
LPBITMAPCOREINFD	指向BITMAPCOREINFO数据结构的长指针。
LPBITMAPFILEHEADER	指向LPBITMAPFILEHEADER数据结构的长指针。
LPBITMAPINFO	指向BITMAPINFO数据结构的32位长指针。使用PBITMAPINFOHEADER创建匹配编译器内存模型的指针。
LPCATCHBUF	CATCHBUF队列的32位指针。
LPCBT_CREATEWND	CBT_CREATEWND数据结构的32位指针。
LPCHOOSECOLOR	CHOOSECOLOR数据结构的32位指针。
LPCHOOSEFONT	CHOOSEFONT数据结构的32位指针。
LPCLIENTCREATESTRUCT	CLIENTCREATESTRUCT数据结构的32位指

	针。
LPCOMPAREITEMSTRUCT	COMPAREITEMSTRUCT数据结构的32位指针。
LPCPLINFO	CPLINFO数据结构的32位指针。
LPCREATESTRUCT	CREATESTRUCT数据结构的32位指针。
LPCSTR	不可改变字符串的32位指针。
LPCTLINFO	CTLINFO数据结构的32位指针。
LPDCB	DCB数据结构的32位指针。
LPDEBUGHOOKINFO	DEBUGHOOKINFO数据结构的32位指针。
LPDELETEITEMSTRUCT	DELETEITEMSTRUCT数据结构的32位指针。使用 PDELETEITEMSTRUCT 创建匹配编译器内存模型的指针。
LPDEVMODE	DEVMODE的数据结构的32位指针。使用 NPDEVMODE 创建 16 位指针，使用 PDEVMODE 创建匹配编译器内存模型的指针。
LPDEVNAMES	DEVNAMES数据结构的32位指针。
LPDOCINFO	DOCINFO数据结构的32位指针。
LPDRAWITEMSTRUCT	DRAWITEMSTRUCT数据结构的32位指针，使用 PDRAWITEMSTRUCT 创建匹配编译器内存模型的指针。
LPDEVCONFIGINFO	DEVCONFIGINFO数据结构的32位指针。使用 PDEVCONFIGINFO 创建匹配编译器内存模型的指针。
LPEVENTMSG	EVENTMSG数据结构的32位指针。使用 NPEVENTMSG 创建 16 位指针，使用 PEVENTMSG 创建匹配编译器内存模型的指针。
LPDRIVERINFOSTRUCT	DRIVERINFOSTRUCT数据结构的32位指针。
LPFINDREPLACE	FINDREPLACE数据结构的32位指针。
LPFMS__GETDRIVEINFO	FMS__GETDRIVEINFO数据结构的32位指针。
LPFMS__GETFILESEL	FMS__GETFILESEL数据结构的32位指针。
LPFMS__LOAD	FMS__LOAD数据结构的32位指针。
LPHANDLETABLE	HANDLETABLE数据结构的32位指针。使用 PHANDLETABLE 创建匹配编译器内存模型的指针。
LPHELPWININFO	HELPWININFO数据结构的32位指针。使用 PHELPWININFO 创建匹配编译器内存模型的指针。
LPINT	指向16位有符号值的32位指针。
LPKERNINGPAIR	KERNINGPAIR数据结构的32位指针。

LPLOGBRUSH	指向LOGBRUSH数据结构的32位指针。使用NPLOGBRUSH创建16位指针，及用PLOGBRUSH创建匹配编译器内存模型的指针。
LPLOGFONT	LOGFONT数据结构的32位指针。
LPLOGPALETTE	LOGPALETTE数据结构的32位指针。
LPLOGPEN	LOGPEN数据结构的32位指针。
LPLONG	指向32位有符号整数的32位指针。使用PLONG创建匹配编译器内存模型的指针。
LPMAT2	MAT2数据结构的32位指针。
LPMDICREATESTRUCT	MDICREATESTRUCT数据结构的32位指针。
LPMETAFILEPICT	METAFILEPICT数据结构的32位指针。
LPMETARECORD	METARECORD数据结构的32位指针。
LPMOUSEHOOKSTRUCT	MOUSEHOOKSTRUCT数据结构的32位指针。
LPMSG	MSG数据结构的32位指针。
LPNCCALCSIZE__PARAMS	NCCALCSIZE__PARAMS数据结构的32位指针。
LPNEWCPLINFO	NEWCPLINFO数据结构的32位指针。
LPNEWTEXTMETRIC	NEWTEXTMETRIC数据结构的32位指针。
LPOFSTRUCT	OFSTRUCT数据结构的32位指针。
LPOLECLIENT	OLECLIENT数据结构的32位指针。
LPOLEOBJECT	OLEOBJECT数据结构的32位指针。
LPOLEOBJECTVTBL	OLEOBJECTVTBL数据结构的32位指针。
LPOLESERVER	OLESERVER数据结构的32位指针。
LPOLESERVERDOC	OLESERVERDOC数据结构的32位指针。
LPOLESERVERDOCVTBL	OLESERVERDOCVTBL数据结构的32位指针。
LPOLESERVERVTBL	OLESERVERVTBL数据结构的32位指针。
LPOLESTREAM	OLESTREAM数据结构的32位指针。
LPOLESTREAMVTBL	OLESTREAMVTBL数据结构的32位指针。
LPOLETARGETDEVICE	OLETARGETDEVICE数据结构的32位指针。
LPOPENFILENAME	OPENFILENAME数据结构的32位指针。
LPOUTLINETEXTMETRIC	OUTLINETEXTMETRIC数据结构的32位指针。
LPPAINTSTRUCT	PAINTSTRUCT数据结构的32位指针。
LPPALETTEENTRY	PALETTEENTRY数据结构的32位指针。
LPPOINT	POINT数据结构的32位指针。
LPPOINTFX	POINTFX数据结构的32位指针。
LPPRINTDLG	PRINTDLG数据结构的32位指针。
LPRASTERIZER__STATUS	RASTERIZER__STATUS数据结构的32位指针。
LPRECT	RECT数据结构的32位指针。
LPRGBQUAD	RGBQUAD数据结构的32位指针。

LPRGBTRIPLE	RGBTRIPLE数据结构的32位指针。
LPSEGINFO	SEGINFO数据结构的32位指针。
LPSIZE	SIZE结构的32位指针。
LPSTR	指向字符串的32位指针，使用MPSTR创建16位指针，使用 PSTR 创建匹配编译器内存模型的指针。
LPTEXTMETRIC	TEXTMETRIC数据结构的32位指针。
LPTTPOLYCURVE	TTPOLYCURVE数据结构的32位指针。
LPTTPOLYGONHEADER	TTPOLYGONHEADER数据结构的32位指针。
LPVOID	未指定类型的32位指针。
LPWINDOWPLACEMENT	WINDOWPLACEMENT数据结构的32位指针。
LPWINDOWPOS	WINDOWPOS数据结构的32位的指针。
LPWNDCLASS	WNDCLASS数据结构的32位指针。
LPWORD	10位无符号值的32位指针。
LRESULT	窗口函数或回调函数返回的32位有符号值。
MFENUMPROC	函数EnumMetaFileProc的32位指针。
NEARPROC	函数的16位指针。
OLECLIPFORMAT	标准剪裁板格式的16位值。
PATTERN	等同于LOGBRUSH结构，使用LPPATTERN创建 32 位指针，使用 NPPATTERN 创建 16 位指针，使用 PPATTERN 创建匹配编译器内存模型的指针。
PCONVCONTEXT	CONVCONTEXT结构的32位指针。
PCONVINFO	CONVINFO结构的32位指针。
PHSZPAIR	HSZPAIR结构的32位指针。
PROPENUMPROC	EnumPropFixedProc或 EnumPropMovableProc函数的 32 位指针。
RSRCHDLRPROC	LoadProc函数的32位指针。
TIMERPROC	TimerProc函数的32位指针。
UINT	16位无符号值。
WNDENUMPROC	EnumWindowsProc函数的32位指针。
WNDPROC	窗口函数的32位指针。
WORD	16位无符号值。
WPARAM	16位有符号值，作为参数传给窗口函数或退出函数。

第二章 消 息

Microsoft Windows 通过格式化的窗口消息在应用程序中通讯。这些消息被送到应用程序的窗口函数处理。

有些消息的返回值包含应用程序需要的有关访问其它消息或其它数据的消息。为了得到返回值，应用程序必须调用 SendMessage，发送消息到窗口。此函数值直到消息得到处理后才返回。若应用程序不需要消息的返回值，则它可以调用 PostMessage 发送信息。此函数把消息放在窗口应用程序队列中，然后立即返回。对于没有返回值的消息，应用程序可以任意使用这两种函数发送，除非在消息描述中另有规定。

一个消息有三部分组成：消息号、字参数和一个长整型参数。消息号由预定义的消息名标识。消息名以字母开始，表达消息的意义和源渊。字参数和长整形参数分别为 wParam 和 lParam，包含各消息号相应的值。

lParam 参数常常包含多种类型的信息。例如高位字表示一个窗口的句柄，而低位字表示一个整数值。HIWORD 和 LOWORD 实用宏抽取 lParam 的高位字和低位字。HIBYTE 和 LOBYTE 实用宏与 HIWORD 和 LOWORD 一起使用便可访问任何字节。

消息号有四个范围，如下所示：

范围	意义
0到WM_USER-1	保留给Windows用
WM_USER到0x7FFF	应用程序可用的整数消息。
0x8000到0xBFFF	保留给Windows用。
0xC000到0xFFFF	应用程序可用的字符串消息。

第一个范围(0 到 WM_USER-1)内的消息号是 Windows 定义的。其中有些值没有显示定义，是留给 Windows 将来用的。本章描述此范围内的消息。

第二个范围(WM_USER 到 7FFF)内的消息号可被一个应用程序定义，用于在应用程序内发送消息，这些消息不应被送到别的应用程序，除非这些应用程序设计成交换消息，因而与消息号有相同的意义的时候。

第三个范围(8000 到 8FFF)内的消息号保留给 Windows 将来使用。

第四个范围(C0000 到 FFFF)内的消息号是在一个应用程序调用 RegisterWindowMessage 函数获取一个字符串的消息号时定义的。登记相同字符串的所有应用程序可用相同的消息号彼此交换信息。然而实际消息号不是常数，不能假定在不同窗口操作中是相同的。

2.1 窗口消息

下面以字母顺序描述了窗口消息

BM__GETCHECK **2.x**

```
wParam = 0;     / * not  used,must  be  zero * /
lParam = 0L;    / * not  used,must  be  zero * /
```

本消息确定按键或检查框是否被检查过。

参　数: wParam,lParam未用

返回值: 由 BS__ AUTOCHECKBOX、 BS__ AUTORADIOBUTTON、BS__ AUTO3STATE、 BS__ CHECKBOX、 BS__ RADIOBUTTON、BS__3STATE 格式创建的钮返回值表示如下:

值	意义
0	按钮未检查
1	按钮已检查
2	按钮状态不确定

如按钮为其它格式，则返回值为零。

实　例: 下例确定ID__MYCHECKBOX控制是否被检查过。

```
int checked;
checked = (int) SendDlgItemMessage(hwndDlg,ID__MYCHECKBOX,
BM__GETCHECK,0,0L
```

参　考: BM__GETSTATE、BM__SETCHECK

BM__GETSTATE **2.x**

```
BM__GETSTATE
wParam = 0;  / * not used,must be zero * /
lParam = 0L / * not used,must be zero * /
```

参　数: 未用

返回值: 出现下列情况之一返回非0

- 按钮是高亮度的
- 按一按键是输入高亮条时，用户按了一鼠标键或空格
- 当光标在一按钮上时，用户按了鼠标键。

实　例: 下例判断当前按钮是否有亮度条。

```
#define  BFFOCUS  0x0008
DWORD  dwResult;
dwResult = SendDlgItemMessage(hdlg,ID__MYBUTTON,BM__GETS
TATE,0,0L); if(dwResult & BFFOCUS)
/ * button  has  the  focus * /
```

参　考: BM__GETCHECK、BM__SETSTATE

BM__SETCHECK **2.x**

wParam = (WPARAM)fCheck; / * check state * /

lParam = 0L; / * not used,must be zero * /

应用程序发送BM__SETCHECK消息检查按钮状态。

参　数: fCheck

是wParam的值，指明检查状态，可为如下值:

值	意义
0	设置按钮至未检查状态。
1	置按钮为检查状态
2	置按钮为不确定状态。只当按钮拥有BS__3STATE或BS__AUTO8STATE格式时方可。

返回值: 为零

说　明: BM__SETCHECK消息对按钮无任何影响。

实　例: 下例将一点置入无线按钮中

SendDlgItemMessage(hdlg,ID__MYRADIOBUTTON,BM__SETCHECK,TRUE,0L);

参　考: BM__GETCHECK、BM__GETSTATE、BM__SETSTATE

BM__SETSTATE **2.x**

wParam = (WPARAM)fstate; / * highlight state * /

lParam = 0L; / * not used,must be zero * /

应用程序发送BM__SETSTATE消息设置按钮的高亮条状态。

参　数: fState为wParam的值，指示按钮是否已经高亮显示。非零值显示高亮条，零值消除高亮条。

返回值: 零

说　明: 高亮显示影响按钮的外部，它对无线按钮及检查框状态无影响。当用户压住鼠标的左键时按钮自动高亮显示。当松开该鼠标键就消除高亮条，以模拟用户按动按钮的视觉效果:

```
/ *
 * Perform some action; then remove the highlighting,
 * thereby returning it to its normal state.
 * /
```

参　考: BM__GETSTATE、BM__SETCHECK

BM__SETSTYLE **2.x**

wParam = (WPARAM) LOWORD (dwStyle); / * style * /

lParam = MAKELPARAM(fRedraw,0); / * redraw flag * /

本消息用来改变按钮的格式。

参　数: wParam值为dwstyle，表示格式值，详见下表lParam低位值为fRedraw，表示按钮是否要再画。TRUE值再画按键，若为FALSE，则不再画。

返回值: 零

说　明: 下表所列为按键格式:

值	意义
BS__3STATE	创建一个与检查框相同的按键，只是框被检查而且变灰，灰色标志一般用于显示一个检查已关闭。
BS__AUTO3STATE	与BS__3STATE相同，只是当用户按它时，按键自动触发其状态。
BS__AUTOCHECKBOX	与BS__CHECKBOX相同，只是当用户按它时，按键自动触发其状态。
BS__AUTORADIOBUTTON	与 BS__RADIOBUTTON 相同，只是按键被检查时，由 BN__CLICKEN 通知应用程序，并且从所有收音按键中删去检查标志。
BS__CHECKBOX	创建一个小四方形，右边显示文本。
BS__DEFPUSHBUTTON	指定有一黑边的一个按键。此按键表示缺省用户响应。任何正文均在按键内显示。用户按此键时，Windows发送一个消息到父窗口。
BS__GROUPBOX	指定一个矩形，其它按键组织于其中，任何正文均显示于矩形的左上角。
BS__LEFTTEXT	使正文出现于收音按键或检查框按键的左边，可以将此方式与 BS__ CHECKBOX、BS__ RADIOBUTTON 或 BS__3STATE 方式一起使用。
BS__OWNERDRAW	指定一所有者绘制的按键。按此按键通知父窗口。通知包括父窗口的一个请求、转换、并关闭此按键。
BS__PUSHBUTTON	指定包含给定正文的一个按键，当用户按此按键时，此控制发送一个消息到其父窗口。
BS__RADIOBUTTON	指定能被检查的一个小图形按键，用户按此按键后其边框变黑。任何正文均出现于按键的右边。一般两个或多个收音按钮组成一组以表达互相排斥的选择，因而一次不会检查一组中的多个按键。

应用程序不能改变键类型。

实　例:　下面例子发送BM__SETSTYLE消息使按键变成缺省按键。

CB__ADDSTRING **3.0**

CB__ADDSTRING

wParam = 0;/ * 没有用，必须为零 * /

lParam = (LPARAM) (LPCSTR) LPSZ;/ * 要增加字符串的地址 * /

应用程序发送一个CB__ADDSTRING消息，以便增加一个字符串到一个组合框的表框中。如果表框不是 CBS__SORT 格式，则字符串加到表尾，否则，字符串插入表中并对该表排序。

参　数: LPSZ

lParam的值。指向要被加入的以null结尾的字符串。如果组合框是以自画格式建立，没有 CBS__HASSTRINGS 格式，则 LPSZ 参数的值被存储而不是指向字符串。

返回值: 返回值是指向表框中字符串的基于零的索引。如果出错，返回值为 CB__ ERR; 如果没有足够的空间来存储新字符串，则返回值为 CB__ERRSPACE。

注意: 如果一个自画组合框是用 CBS__ SORT格式建立的，而不是用 CBS__HASSTRINGS 格式建立的，则 WM__COMPAREITEM 消息被一次或多次送到组合框的所有者，以便新项能被合适地放到表框中。

为了插入一个字符串到表中的特定位置，使用CB__INSERTSTRING消息。实例增加字符串"mystring"到一个表框:

```
DWORD dwIndex;
dwIndex = SendDlgItemMessage(hdlg,ID__MYCOMBOBOX,CB__AD
DSTRING,0,(LPARAM)((LPCSTR)"my string"));
```

参　考: CB__INSERTSTRING, WM__COMPAREITIM

CB__DELETESTRING **3.0**

CB__DELETESTRING

wParam = (WPARAM)indcx;/ * 要删除的项 * /

lParam = 0L;/ * 没有用，必须为零 * /

应用程序发送一个CB__DELETESTRING消息，以删除一个组合框表框中的一个字符串。

参　数: indcx

wParam的值，规定要被删除字符串的基于零的索引。

返回值: 返回值为留在表中字符串的个数。如果indcx参数指定一个大于表中项数的索引，则返回值为 CB__ERR。

说　明: 如果组合框是用自画格式建立的，因而没有CBS__HASSTRINGS格式，则一个 WM__DELETEITEM 消息被送到组合框所有者，以便应用程序能够释放任何与此项关联的附加数据。

实　例: 本实例删除某组合框中的第一个字符串。

```
dword DWREMAINING;
dwRemaining = SendDlgItemMessage(hdlg,ID__MYCOMBOBOX,
```

CB__DELETESTRING,0,0L);

参　考:　WM__DELETEITEM

CB__DIR **3.0**

CB__DIR

wParam=(WPARAM) (UINT) uAttris;/ * 文件属性 * /

lParam=(LPARAM) (LPCSTR) LpszFileSpec;/ * 文件名地址 * /

应用程序发送一个CB__DIR消息，以便增加一组文件名到一个组合框的表框中.

参　数:　uAttris

wParam的值. 说明被加入文件的属性. 它可以是下列值的任意组合:

值	意义
0x0000	能读取或写入的文件
0x0001	能读取但不能写入的文件
0x0002	文件被隐藏而不是显示在目录表中
0x0004	文件是一个系统文件
0x0010	lpszFileSpec参数所指向的名字指定一个目录
0x0020	已归档的文件
0x4000	包括所有与由lpszFileSpec参数指定名字相匹配的驱动器
0x8000	专用标志. 如果专用标志被设置，则只列出特定类型的文件，否则列出所有文件.

lpszFileSpec

lParam的值，指向以null结尾的字符串，该字符串指明要加入表中的文件名. 如果文件名包含通配符(如 * .*)，则所有与之匹配并具有由 uAttris 参数说明的属性的文件都将被加入到表中去.

返回值:　返回值是被加入的最后一个文件名的基于零的索引. 如果发生错误则返回值为 CB__ERR; 如果没有足够的空间存储新字符串，则返回值为 CB__ERRSPACE.

实　例:　本实例把所有可能的驱动器名字增加到一个组合框中.

dwIndexLastItem = SendDlgItemMessage(hdlg,ID__MYCOMBOBOX, CB__DIR,0x4000 |0x8000,(lparam) ((lpcstr)" * :));

参　考:　DlgDirList

CB__FINDSTRING **3.0**

CB__FINDSTRING

wParam=(WPARAM) indexstart;/ * 开始检索项的前一项 * /

lParam=(LPARAM) (LPCSTR) lpszFind;/ * 前缀字符的地址 * /

应用程序发送一个CB__FINDSTRING消息，以便在组合框的表框中找到包含指定前缀的第一个字符串.

参　数:　indexstart

wParam的值，为要被检索的第一项的前一项的索引，索引从零开始. 当检

索到框底时，它继续从框顶开始检索，直到找到由 indexstart 参数指定的项。如果 indexstart 为-1，则整个表框从头到尾均检索。

lpszFind

lParam的值，指向含有被检索前缀的以null结尾的字符串。检索不区分大小写，所以该字符可以包含大小写字母的任意组合。

返回值: 返回值是匹配的基于零的索引，如果检索不成功则返回值为CB_ERR.

说　明: 如果组合框的格式是自画的而不是CB_HASSTRING和CBS_SQRT，则使用 CB_FINDSTRING；如果格式是自画的和 CBS_SORT 而不是 CB_HASSTRING，则发送 WM_COMPAREITEM 消息。

实　例: 本实例在组合框中检索字符串"mystring"。如果找到则把它拷贝到szBuf缓冲区。

```
Char szBuf[20];
DWORD dwIndex;
dwIndex = SendDlgItemMessage(hdlg,ID_ MYCOMBOBOX,CB_ FI
NDSTRING,0,(LPARAM)((LPCSTR)"my string"));
if (dwIndex != CB_ERR)
SendDlgItemMessage(hdlg,ID_MYCOMBOBOX,
CB_ GETLBTEXT,(WPARAM)dwIndex,(LPARAM)((LPCSTR)szBu
f));
```

参　考: CB_FINDSTRINGEXACT，CB_SETCURSEL

CB_FINDSTRINGEXACT　　3.1

CB_FINDSTRINGEXACT

wParam = (WPARAM) indexstart / * 开始检索项的前一项 * /

lParam = (LPARAM)(LPCSTR)lpszFind; / * 前缀字符串的地址 * /

应用程序发送一个CB_FINDSTRINGEXACT消息，以便在组合框中找到与由 lpszFind 参数说明的字符串相匹配的第一个表框字符串。

参　数: indexStart

wParam的值，为要被检索的第一项的前一项的索引，索引从零开始。当检索到框底时，它继续从框顶开始检索，直到找到由 indexStart 指定的项。如果 indexStart 为-1，则整个表框从头到尾均检索。

lpszFind

lParam的值，指向以null结尾的被检索字符串。该字符串可能包含一个完整的文件名，其中包括扩展名。检索不区分大小写，所以字符串可以包含大小写字母的任意组合。

返回值: 返回值是匹配项的基于零的索引。如果检索失败，则返回值为CB_ERR。

说　明: 如果组合框的格式是自画式而不是CB_HASSTRING或者CBS_SORT，则使用 CB_FINDSTRING；如果格式是自画式和 CBS_SORT 而不是 CBS_HASSTRING，则发送 WM_COMPAREITEM 消息。

参　考: CB_FINDSTRING，CB_SETCURSEL