

金版电脑编程实例

Visual Basic

经典编程 128 例

主编 樊金生

编著 刘展威 樊金生 赵永斌
刘玉红 韩艳峰

编程高手现身“说法”
深入剖析，详细讲解
高手编程时尚必备手册

Visual Basic 经典编程 128 例 Visual Basic 经典编程 128 例

光明日报出版社

金版电脑编程实例

Visual Basic 经典编程 128 例

主编 樊金生

编著 樊金生 赵永斌 刘玉红

韩艳峰 刘展威

光明日报出版社

图书在版编目 (C I P) 数据

Visual Basic 经典编程 128 例/樊金生主编; 樊金生等编著.

—北京: 光明日报出版社, 2004.

(金版电脑编程实例)

ISBN 7-80145-880-X

I. V… II. ①樊…②樊… III. BASIC 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2004) 第 032464 号

内容简介

Visual Basic 是微软公司推出的可视化开发工具, 自其诞生以来, 一直是 Windows 环境下最主要的应用开发系统。本书通过 128 个精心设计、新颖别致、风格各异精彩设计实例, 详细讲解了利用 Visual Basic 的强大功能开发应用程序的方法和技巧。全书包含窗体与界面、控件、系统、文件、图形图像、多媒体、数据库编程、网络应用设计、动态链接库、多线程控制、ActiveX 与自动化编程、其它等 12 篇。内容几乎覆盖了 Windows 编程的各个方面。

本书是广大 Visual Basic 开发人员积累编程经验、拓展视野的得力助手, 既适用于初学者的使用, 也适用于已经初步掌握 Visual Basic 编程概念、方法的读者阅读, 并可以帮助读者迅速掌握实际应用中的各种经验、技巧。

金版电脑编程实例

Visual Basic 经典编程 128 例

※

光明日报出版社出版发行

(北京珠市口东大街 5 号)

邮政编码: 100062

电话: 67078237

全国各地新华书店经销

石家庄市春蕾印刷厂印刷

※

787×1092 1/16 印张 232 字数 5500 千字

2004 年北京第 1 版 2004 年石家庄第 1 次印刷

印数: 1-5000 册 ISBN 7-80145-880-X/TP

定价 326.00 元 (全 9 册)

前言

Visual Basic 自诞生以来,一直是 Windows 环境下最主要的应用开发系统。它不仅是 Basic 语言的集成开发环境,而且与 Win32 紧密相连,所以,利用 Visual Basic 可以完成从底层软件直到上层直接面向用户的软件开发工作;且 Visual Basic 强大的调试功能也为大型复杂软件的开发提供了有效的排错手段。目前 Visual Basic 已成为国内外应用最广泛的开发工具之一。

为了让大家能深入且熟练地掌握使用 Visual Basic 开发程序的方法和技巧,我们编写了《Visual Basic 精彩实效 128 例》一书。本书采用新颖的版式,将知识和实例紧密结合,通过对各种实例的详细讲解,使读者直接从实例的制作过程中体会到 Visual Basic 每项功能的使用方法,并自己做出各种实例效果;这样既节省了大量时间,又使读者有身临其境的感觉,经过反复演练,可以将所学知识运用到工作中去。

本书通过 128 个精心设计、新颖别致、风格各异的精彩设计实例,详细讲解了 Visual Basic 开发应用程序的方法和技巧。本书实例安排上注意由浅入深,在每一部分的开头还注意了相关理论的简要介绍,有助于初学者理解实例内容,随着你技能的不断增长,本书逐渐探讨更深入的内容,如用户界面设计与一般窗口、图形(像)处理与多媒体、数据库、文件处理、网络与通信等,这些话题本书都将以直观的例子和清晰的描述呈现给你,使你能很快进阶到 Windows 编程高手行列。全书共分为 7 篇。

第一篇 窗体与界面设计实例:主要讲述各种程序外观的设计方法,包括窗体、对话框、菜单、工具栏、标题栏、状态栏等。

第二篇 图形(像)处理与多媒体:具体讲述了如何综合使用 Visual Basic 强大的控件资源,结合代码进行图形、图像设计,以及如何使用各种丰富的控件进行多媒体设计。

第三篇 文件处理:讲述了与文件及目录相关的方法与技巧。

第四篇 数据库:主要讲述了在 Visual Basic 中常用的 ODBC、ADO 两种方法进行数据库程序设计的相关概念与技巧。

第五篇 网络与通信:讲述了如何用 Visual Basic 进行各种网络与通信方面的应用设计。

第六篇 操作系统:讲述了与系统设置及系统信息获取相关的方法与技巧。

第七篇 杂项:主要讲述了在 Visual Basic 中如何实现指纹验证等方面的问题。

本书由樊金生主编,樊金生、赵永斌、刘玉红、韩艳峰、刘展威编著,严世强提供了部分应用技术资料和编程素材,刘晓敏、刘晓星在程序录入、文字校对和资料的整理上给予了很大帮助,全书由刘展威统稿,樊金生审校。

作者



目 录

第一章 用户界面设计与一般窗口	1
实例 1 最小化所有浏览器窗口	1
实例 2 实现特效菜单	2
实例 3 将自己的菜单项加到系统菜单里	4
实例 4 将其他应用程序作为本程序的一个子窗口	8
实例 5 窗体标题栏闪烁	11
实例 6 动画光标	12
实例 7 滚动的标题栏文字	14
实例 8 窗体启动特效	15
实例 9 自动隐藏菜单	20
实例 10 带图标的菜单	22
实例 11 软件封面的设计	25
实例 12 显示 About 窗体	27
实例 13 实现颜色渐变的窗体	28
实例 14 特殊窗体	30
实例 15 使窗体始终显示在最前端	32
实例 16 在菜单中显示最近打开的文件记录	34
实例 17 隐藏 Windows 的开始按钮	39
实例 18 自适应表单的实现	41
实例 19 多行文本框的快速操作	43
实例 20 创建快捷方式	45
第二章 用户界面设计与一般窗口	48
实例 21 实现打字效果	48
实例 22 使用 Word 的“艺术字”	51
实例 23 下雪场景的制作	52
实例 24 制作旋转的文字	55
实例 25 翻转图片	58
实例 26 放置透明图片	60
实例 27 图象的放大和缩小	64
实例 28 屏幕抓图方法的实现	67
实例 29 播放 GIF 动画	73
实例 30 制作 AVI 播放器	74
实例 31 旋转图片	76
实例 32 播放 Flash 动画	79



实例 33 在系统托盘中增加一个光驱按钮	81
实例 34 提取可执行文件内部所有图标	84
实例 35 用鼠标旋转立方体	88
实例 36 播放 WAV 和 MDI 文件	91
实例 37 制作 MP3 播放器	95
第三章 文件处理	99
实例 38 如何建立文件关联	99
实例 39 自动打开上次关机时为关闭的应用程序	101
实例 40 获取文件信息	102
实例 41 把数据写入自身 (.exe) 中	106
实例 42 调用 help 文件并实现 winhelp 的关键字、主题等功能	111
实例 43 设置应用程序的启动热键	114
实例 44 按照 16 进制方式查看文件	116
实例 45 监视资源管理器中的操作	120
实例 46 在所有驱动器上查找文件	123
实例 47 提取文件图标	131
实例 48 加密文件	135
实例 49 拷贝文件	137
实例 50 显示文件的属性	138
实例 51 利用递归方法查找文件	141
实例 52 创建文件夹列表框	144
实例 53 文件比较	146
实例 54 txt 文件读取保存	150
实例 55 删除文件到回收站	152
实例 56 清空回收站	153
实例 57 怎样显示文件日期	155
实例 58 读取 INI 文件	159
实例 59 打印文件	160
实例 60 获取目录的大小	164
第四章 数据库	167
实例 61 读取 Access 数据库密码	167
实例 62 在程序运行阶段用代码创建表	169
实例 63 用 VB 控制 Excel	171
实例 64 数据库的自动备份	172
实例 65 简单的数据库查询	174
实例 66 数据库中数据的更新	176



实例 67 将图片（其它）文件存入数据库中	179
实例 68 用数据库保存程序的用户设置	181
实例 69 数据库用户管理	184
实例 70 获取数据库的信息	188
实例 71 数据库的加密和解密	191
实例 72 记录的锁定	194
实例 73 表的锁定	196
实例 74 在数据库中存取图像字段	198
实例 75 用户的数据库权限管理	202
实例 76 代码连接到 ODBC	206
实例 77 将数据库输出到一个文本文件	209
实例 78 打印数据库中的数据	211
实例 79 远程数据库的更新	214
实例 80 简单的 ADO 与 SQL SERVER 连接应用	217
实例 81 用代码向 SQL SERVER 数据库表插入数据	219
实例 82 SQL SERVER 中带条件的模糊查询	221
实例 83 查询 Outlook 数据	223
实例 84 打印 DBGrid 显示的全部数据	225
实例 85 带可变参数的 SQL 查询	227
实例 86 用图表表示数据	229
实例 87 用 ADO 和 DAO 连接数据库	232
第五章 网络与通信	234
实例 88 获取本地网卡 MAC 地址	234
实例 89 获取本地计算机的网上信息	237
实例 90 通过计算机名获得 IP 地址	239
实例 91 获取网络中某台计算机的共享信息	243
实例 92 判断网络连接方式	245
实例 93 实现 Ping 操作	248
实例 94 自动上网计时	253
实例 95 同步网络时间	257
实例 96 端口扫描	258
实例 97 用 UDP 协议实现聊天程序	260
实例 98 连续批量 Ping 测试	262
实例 99 FTP 客户端程序	269
实例 100 自己制作浏览器	276
实例 101 超级链接的制作	278
实例 102 电子邮件发送程序	280



实例 103 电子邮件接收程序	283
实例 104 网上发送消息的实现	285
实例 105 实时监测局域网内部的计算机	286
实例 106 从 VB 应用程序中发送 ICQ 消息	292
实例 107 添加 URL 快捷方式到收藏夹\开始菜单和桌面上	295
第六章 操作系统	299
实例 108 获取系统硬件信息	299
实例 109 编辑注册表信息	302
实例 110 获取 IE 版本号	308
实例 111 获取 CPU 信息	309
实例 112 重新启动和关闭计算机	311
实例 113 改变默认的打印机	314
实例 114 获取内存信息	318
实例 115 显示磁盘剩余空间	321
实例 116 显示磁盘细节	322
实例 117 如何获取系统的版本信息/用户注册信息	325
实例 118 用 Windows API 函数判定文件共享锁定状态	328
实例 119 如何检测 ShiftAlt 和 Ctrl 键是否被按下	331
实例 120 设置 IE 的主页网址	332
实例 121 弹出和关闭光驱	334
实例 122 用户登录时最多可输入 3 次密码	335
实例 123 用 WMI 获取 BIOS 信息	337
实例 124 限定软件使用次数	341
第七章 杂例	343
实例 125 切换中文输入法的实现	343
实例 126 实现系统启动时程序自动被执行	347
实例 127 实现记忆键盘录入	349
实例 128 指纹登记和验证演示程序	355
VB.net 编码规范	361



第一章 用户界面设计与一般窗口

实例 1 最小化所有浏览器窗口

范例目的

在网上时，经常会打开很多浏览器窗体，本例介绍如何使所有浏览器窗体最小化。

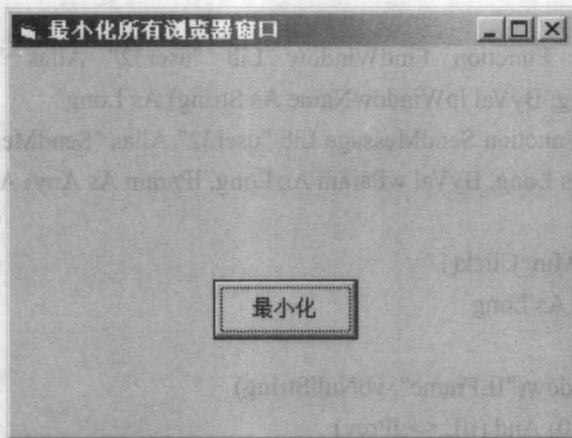


图 1-1 最小化所有浏览器窗口

实现技术

1、使用 FindWindow 函数，寻找窗口列表中第一个符合指定条件的顶级窗口。函数原型为：

HWND FindWindow (LPCTSTR IpClassName, LPCTSTR IpWindowName) ;

其中参数 IpClassName，指向一个指定了类名的空结束字符串，或一个标识类名字符串的成员的指针；IpWindowName，指向一个指定了窗口名（窗口标题）的空结束字符串。如果该参数为空，则为所有窗口全匹配。如果函数成功，返回值为具有指定类名和窗口名的窗口句柄；如果函数失败，返回值为 NULL。

2、调用用 SendMessage 函数将指定的消息发送到一个或多个窗口，只有消息处理完毕，函数才会返回。函数原型为：



LRESULT SendMessage (HWND hWnd, UINT Msg, WPARAM wParam, LPARAM lParam);

其中参数 hWnd, 要接收消息的那个窗口的句柄; Msg, 指定被发送的消息; wParam, 指定附加的消息指定信息; lParam, 指定附加的消息指定信息。

操作和代码

新建工程, 选择“标准 EXE”, 在窗体中添加 Command 控件, 命名为“cmdMin”。

Option Explicit

Private Const WM_SYSCOMMAND = &H112

Private Const SC_MINIMIZE = &HF020&

Private Declare Function FindWindow Lib "user32" Alias "FindWindowA" (ByVal lpClassName As String, ByVal lpWindowName As String) As Long

Private Declare Function SendMessage Lib "user32" Alias "SendMessageA" (ByVal hWnd As Long, ByVal wParam As Long, ByVal lParam As Long, lParam As Any) As Long

Private Sub cmdMin_Click()

Dim lPrev, lIE As Long

lPrev = 0

lIE = FindWindow("IEFrame", vbNullString)

While (lIE <> 0) And (lIE <> lPrev)

SendMessage lIE, WM_SYSCOMMAND, SC_MINIMIZE, 0

lPrev = lIE

lIE = FindWindow("IEFrame", vbNullString)

Wend

End Sub

实例 2 实现特效菜单

范例目的

应用程序中菜单均顺序排列于窗体左侧, 本例介绍如何使具有其它功能的菜单置于窗体右侧。



实现技术

- 1、使用 API 函数 `GetMenu`，取得分配给指定窗口的菜单的句柄，函数原型为：
`HMENU GetMenu (HWND hWnd);`

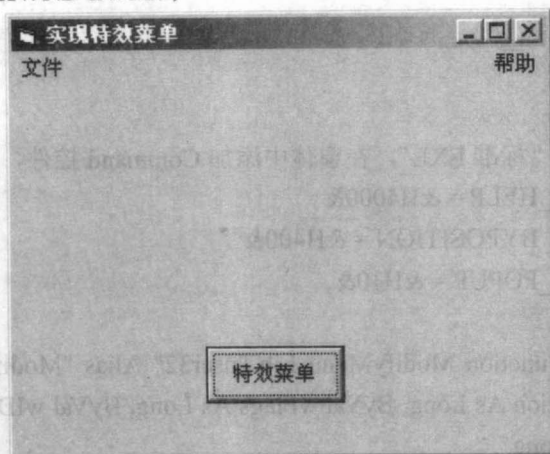


图 2-1 实现特效菜单

- 2、使用 API 函数 `ModifyMenu` 更改菜单，函数原型为：

`BOOL ModifyMenu(HMENU hMnu,UINT uPosition,UINT uFlags,UINT uIDNewItem,LPCTSTR lpNewItem);`

函数功能：该参数修改已存在的菜单项，并指定菜单项的内容、外观和性能。

参数：

Hmnu	将被修改的菜单的句柄。
Uposition	指定将被修改的菜单项，其含义由参数 Uflags 决定。
Uflags	指定控制参数 uPosition 的解释的标志、菜单项的内容、外观和性能。此参数必须为下列值之一和列于备注里的一个值的组合。 MF_BYCOMMAND: 表示 uPostion 给出菜单项的标识符。如果 MF_BYCOMMAND 和 MF_BYPOSITION 都没被指定则 MF_BYCOMMAND 为缺省的标志。 MF_BYPOSITION:表示 uPosition 给出菜单项基于零的相对位置。
uIDNewItem	指定被修改菜单项的标识符，或者当参数 uFlags 设置为 MF_POPUP 时，指定下拉式菜单或子菜单的句柄。
LpNewItem	指定被修改菜单项的内容。其含义依赖于参数 UFlags 是否包含标志 MF_BITMAP,MF_OWNERDRAW 或 MF_STRING。

备注：如果函数 `ModifyMenu` 替换了打开下拉式菜单或子菜单的菜单项，则函数销毁旧



的下拉式菜单或子菜单，并释放它们占用的内存。

3、使用 API 函数 DrawMenuBar 重画指定菜单的菜单条。在系统创建窗口以后如果菜单条被修改，则应调用此函数来画修改了的菜单条。函数原型为：

BOOL DrawMenuBar (HWND hWnd);

其中参数 hWnd 是需要被重画菜单条窗口的句柄。

操作和代码

新建工程，选择“标准 EXE”，在窗体中添加 Command 控件，命名为“cmdMenu”。

Private Const MF_HELP = &H4000&

Private Const MF_BYPOSITION = &H400&

Private Const MF_POPUP = &H10&

Private Declare Function ModifyMenu Lib "user32" Alias "ModifyMenuA" (ByVal hMenu As Long, ByVal nPosition As Long, ByVal wFlags As Long, ByVal wIDNewItem As Long, ByVal lpString As Any) As Long

Private Declare Function DrawMenuBar Lib "user32" (ByVal hwnd As Long) As Long

Private Declare Function GetMenu Lib "user32" (ByVal hwnd As Long) As Long

Private Sub cmdMenu_Click()

Dim hdcMenu As Long

Dim hdcSubMenu As Long

Dim ltemp As Long

hdcMenu = GetMenu(MainFrm.hwnd)

ModifyMenu hdcMenu, 1, MF_BYPOSITION Or MF_POPUP Or MF_HELP, mmHelp, "帮助"

DrawMenuBar (MainFrm.hwnd) '刷新菜单

End Sub

实例 3 将自己的菜单项加到系统菜单里

范例目的

在应用程序中，单击窗体的系统菜单菜单本例简介如何将自己的菜单项加到系统菜单里。

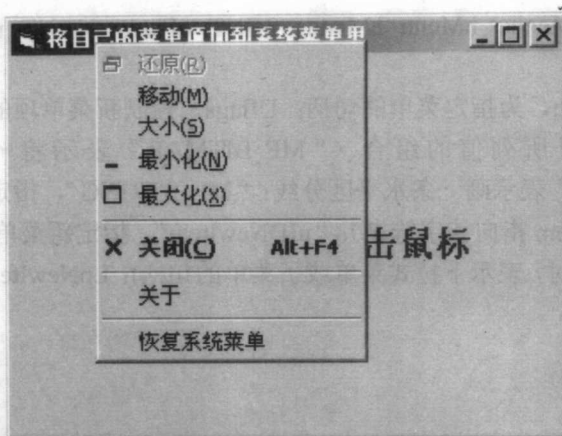


图 3-1 将自己的菜单项加到系统菜单里

实现技术

1、通过 API 函数 `GetWindowLong`，获得有关指定窗口的信息，以便于在恢复设置时使用。函数原型为：

`LONG GetWindowLong (HWND hWnd, int nIndex);`

其中参数 `hWnd`，获取信息的窗口的句柄；`nIndex`，欲取回的信息的值的偏移量，值为 `GWL_WNDPROC` 表示该窗口的窗口函数地址。

2、使用 API 函数 `SetWindowLong`，改变指定窗口的属性，函数原型为：

`LONG SetWindowLong (HWND hWnd, int nIndex, LONG dwNewLong);`

其中参数 `hWnd`，窗口句柄；`nIndex`，参照 `GetWindowLong` 中的 `nIndex`；`dwNewLong`，由 `nIndex` 指定的窗口信息的新值，在本例中为自定义函数入口。

3、`SetWindowLong` 中使用 `Addressof` 运算符，将参数修改为返回函数的地址（此种使用不是返回函数调用的结果）。注意：只有 `.bas` 模块中的 `Visual Basic` 过程名才可以用 `AddressOf` 修改。

4、添加模块文件，将用到的 API 函数、常量和自定义函数。

5、在自定义函数中，如果不是需要的消息使用 API 函数 `CallWindowProc`，将消息信息传送给默认的窗口过程。函数原型为：

`LRESULT CallWindowProc (WNDPROC lpPrevWndFunc, HWND hWnd, UINT Msg, WPARAM wParam, LPARAM lParam);`

其中参数 `lpPrevWndFunc` 是指向前一个窗口过程的指针；`hWnd`，指向接收消息的窗口过程的句柄；`Msg`，指定消息类型；`wParam`：指定其余的、消息特定的信息。该参数的内容与 `Msg` 参数值有关；`lParam`，指定其余的、消息特定的信息。

6、使用 API 函数 `GetSystemMenu`，获得窗口菜单句柄。参照实例 2 中解释。

7、使用 API 函数 `AppendMenu`，在指定的菜单条、下拉式菜单、子菜单或快捷菜单的末



尾追加一个新菜单项。函数原型:

BOOL AppendMenu (hMenu hMenu, UINT uFlags, UINT uIDNewItem, LPCTSTR lpNewItem);

其中参数 hMenu, 为指定菜单的句柄; Uflags, 控制新菜单项的外观和性能的标志。此参数可以是备注里所列值的组合 (“MF_BITMAP” 表示将一个位图用作菜单项; “MF_SEPARATOR” 表示画一条水平区分线; “MF_STRING”, 指定菜单项是一个正文字符串, 此时参数 lpNewItem 指向该字符串); “uIDNewItem”, 指定新菜单项的标识符 (如果 uFlags 设置为 MF_POPUP 时, 表示下拉式菜单或子菜单的句柄); LpNewItem, 指定新菜单项的内容。

操作和代码

1、新建工程, 选择“标准 EXE”, 命名为“MainFrm”。

2、添加模块“Module”, 在其中自定义函数。

MainFrm 中代码:

Option Explicit

Private Sub Form_Load()

 procOldwindow = GetWindowLong(MainFrm.hwnd, GWL_WNDPROC)

 SetWindowLong MainFrm.hwnd, GWL_WNDPROC, AddressOf User_SendMessage

 ' 用 User_SendMessage 代替窗口函数处理消息

 hwndMenu = GetSystemMenu(MainFrm.hwnd, False)

 AppendMenu hwndMenu, MF_SEPARATOR, 1, vbNullString

 AppendMenu hwndMenu, MF_STRING, USER_ABOUT, "关于"

 AppendMenu hwndMenu, MF_SEPARATOR, 2, vbNullString

 AppendMenu hwndMenu, MF_STRING, USER_RESTORE, "恢复系统菜单"

End Sub

Private Sub Form_Unload(Cancel As Integer)

 If procOldwindow <> GetWindowLong(MainFrm.hwnd, GWL_WNDPROC) Then

 SetWindowLong MainFrm.hwnd, GWL_WNDPROC, procOldwindow

 End If

End Sub

Module 中代码:

Option Explicit

Public Const WM_SYSCOMMAND = &H112 ' 单击控制框时产生的消息

Public Const MF_SEPARATOR = &H800& ' 分隔线常数

Public Const MF_STRING = &H0& ' 在菜单中加一个字符串

Public Const GWL_WNDPROC = (-4)



```
Public Const USER_ABOUT = 100
```

```
Public Const USER_RESTORE = 101
```

```
Public Declare Function GetWindowLong Lib "user32" Alias "GetWindowLongA" (ByVal  
hwnd As Long, ByVal nIndex As Long) As Long
```

```
Public Declare Function SetWindowLong Lib "user32" Alias "SetWindowLongA" (ByVal  
hwnd As Long, ByVal nIndex As Long, ByVal dwNewLong As Long) As Long
```

```
Public Declare Function CallWindowProc Lib "user32" Alias "CallWindowProcA" (ByVal  
lpPrevWndFunc As Long, ByVal hwnd As Long, ByVal Msg As Long, ByVal wParam As Long,  
ByVal lParam As Long) As Long
```

```
Public Declare Function GetSystemMenu Lib "user32" (ByVal hwnd As Long, ByVal bRevert  
As Long) As Long
```

```
Public Declare Function AppendMenu Lib "user32" Alias "AppendMenuA" (ByVal hMenu  
As Long, ByVal wFlags As Long, ByVal wIDNewItem As Long, ByVal lpNewItem As Any) As  
Long
```

```
Public procOldwindow As Long ' 默认窗口函数地址
```

```
Public hwndMenu As Long ' 菜单句柄
```

```
Public Function User_SendMessage(ByVal hwnd As Long, ByVal Msg As Long, ByVal wp  
As Long, ByVal lp As Long) As Long
```

```
If Msg <> WM_SYSCOMMAND Then ' 消息不是 WM_SYSCOMMAND, 调用默认的  
窗口函数进行处理
```

```
User_SendMessage = CallWindowProc(procOldwindow, hwnd, Msg, wp, lp)
```

```
Exit Function
```

```
End If
```

```
Select Case wp
```

```
Case USER_ABOUT
```

```
MsgBox "显示关于信息 ", vbOKOnly + vbInformation
```

```
Case USER_RESTORE
```

```
GetSystemMenu MainForm.hwnd, True
```

```
SetWindowLong MainForm.hwnd, GWL_WNDPROC, procOldwindow
```

```
MsgBox "已恢复系统菜单 ", vbOKOnly + vbInformation
```

```
Case Else
```

```
User_SendMessage = CallWindowProc(procOldwindow, hwnd, Msg, wp, lp)
```

```
Exit Function
```

```
End Select
```

```
User_SendMessage = True
```

```
End Function
```




实例 4 将其他应用程序作为本程序的一个子窗口

范例目的

应用程序在运行时均有各自的窗体,而在编程时希望一些应用程序窗体如同 MDI 子窗体一样作为子窗体运行。本例简介如何通过程序实现这一功能。

实现技术

1、通过 Shell 函数,运行一个可执行文件。如果运行成功,返回程序的任务 ID,若不成功,则会返回 0。函数原型为:

Shell(pathname[,windowstyle])

其中参数 pathname,要执行的程序名,以及任何必需的参数或命令行变量,可能还包括目录或文件夹,以及驱动器;Windowstyle,可选参数,表示在程序运行时窗口的样式。

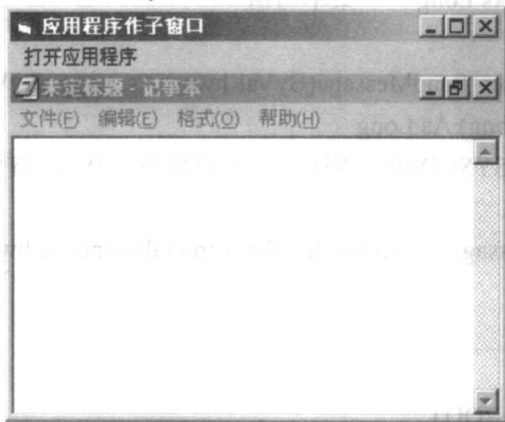


图 4-1 将其他应用程序作为本程序的一个子窗口

2、自定义函数 Getwindowhwnd,通过使用 Shell 获取的 ID,获得应用程序的窗口句柄。

3、在函数 Getwindowhwnd 中使用 API 函数 FindWindow,获得一个顶层窗口的句柄,该窗口的类名和窗口名与给定的字符串相匹配。函数原型为:

HWND FindWindow (LPCTSTR IpClassName, LPCTSTR IpWindowName);

其中参数 IpClassName,指向一个已指定类名的空结束字符串,或一个标识类名字符串的成员的指针,若设为零,表示接收任何类;IpWindowName,指向一个指定了窗口名(窗口标题)的空结束字符串。若设为零,表示接收任何窗口标题。如果函数成功,返回值为具有指定类名和窗口名的窗口句柄;如果函数失败,返回值零。

4、找到窗口句柄后,使用 API 函数 GetParent,获得一个指定子窗口的父窗口句柄。函数原型为:HWND GetParent (HWND hWnd);



5、使用 API 函数 `GetWindowThreadProcessId`，获取与指定窗口关联在一起的一个进程和线程标识符，用于和 `Shell` 函数后的 ID 进行比较。函数原型为：

```
DWORD GetWindowThreadProcessId(HWND hWnd,LPDWORD lpdwProcessId);
```

其中参数 `hWnd`，指定窗口句柄；`lpdwProcessId`，指定一个变量，用于装载拥有那个窗口的一个进程的标识符。

6、使用 API 函数 `SetParent`，改变指定子窗口的父窗口。函数原型为：

```
HWND SetParent (HWND hWndChild,HWND hWndNewParent);
```

其中参数 `hWndChild` 是子窗口句柄；`hWndNewParent`，新的父窗口句柄。如果该参数是零，则桌面窗口就成为新的父窗口。如果函数成功，函数返回值为子窗口的原父窗口句柄，如果函数失败，返回值为零。

操作和代码

新建工程，选择“标准 EXE”，在窗体中新建菜单。

Option Explicit

```
Private Declare Function GetWindowThreadProcessId Lib "user32" (ByVal hWnd As Long,
lpdwProcessId As Long) As Long
```

```
Private Declare Function GetParent Lib "user32" (ByVal hWnd As Long) As Long
```

```
Private Declare Function FindWindow Lib "user32" Alias "FindWindowA" (ByVal
lpClassName As Long, ByVal lpWindowName As Long) As Long
```

```
Private Declare Function GetWindow Lib "user32" (ByVal hWnd As Long, ByVal wCmd As
Long) As Long
```

```
Private Declare Function GetWindowText Lib "user32" Alias "GetWindowTextA" (ByVal
hWnd As Long, ByVal lpString As String, ByVal cch As Long) As Long
```

```
Private Declare Function SetParent Lib "user32" (ByVal hWndChild As Long, ByVal
hWndNewParent As Long) As Long
```

```
Private Const GW_HWNDNEXT = 2
```

```
Private hWndApplicationParent As Long
```

```
Private hWndApplication As Long
```

```
Private Function GetwindowhWnd(ByVal lSourceId As Long) As Long
```

```
Dim hWndTemp As Long
```

```
Dim lProcessId As Long
```

```
Dim lIdTemp As Long
```