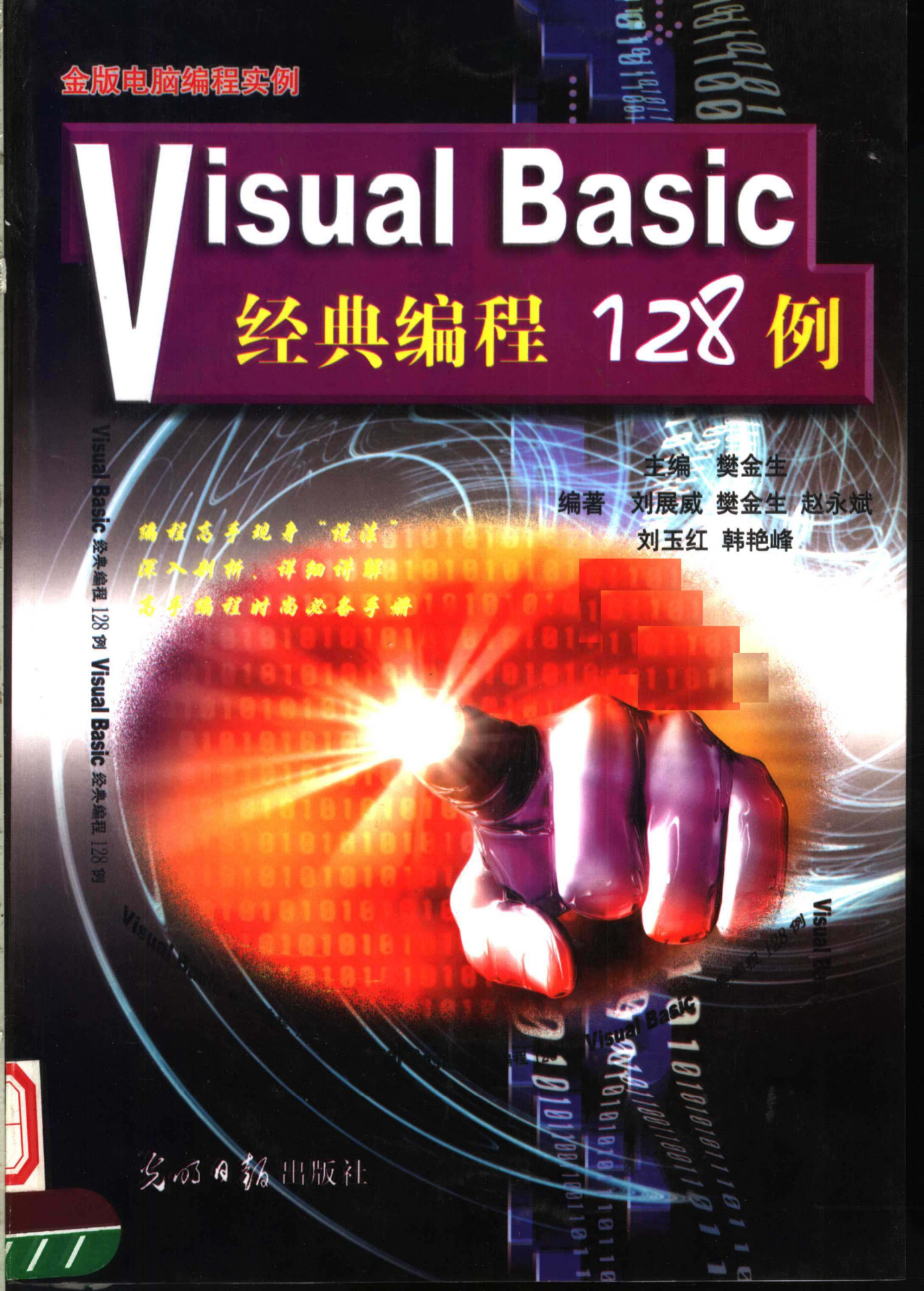
金版电脑编程实例

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



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（CIP）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、风格各异的精彩设计实例，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

实例 1 最小化所有浏览器窗口

范例目的

在上网时，经常会打开很多浏览器窗体，本例介绍如何使使所有浏览器窗体最小化。

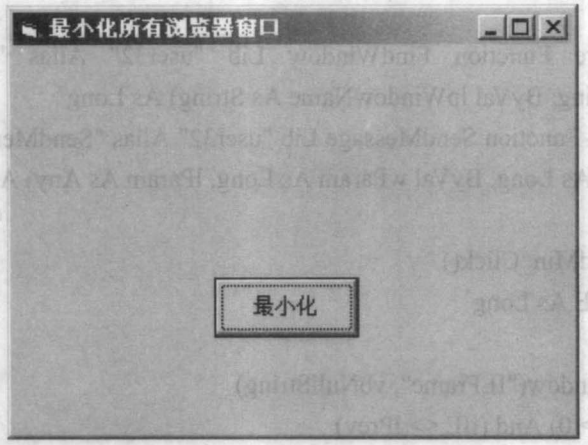


图 1-1 最小化所有浏览器窗口

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 wMsg As Long, ByVal wParam 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）;

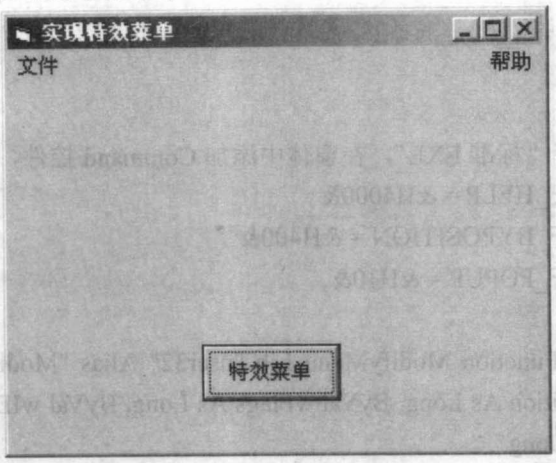


图 2-1 实现特效菜单

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。

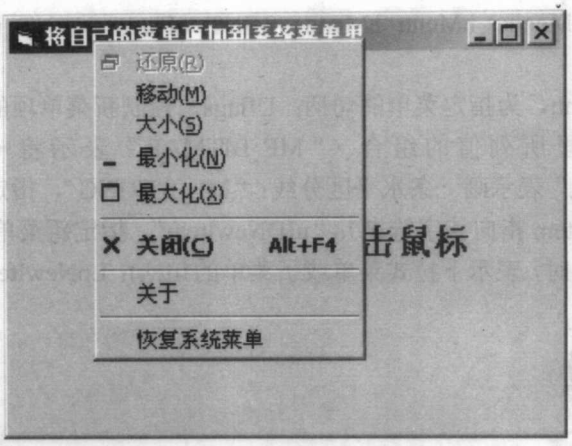


图 3-1 将自己的菜单项加到系统菜单里

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 nlndex）；

其中参数 hWnd，获取信息的窗口的句柄；nlndex，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 nlndex，LONG dwNewLong）；

其中参数 hWnd，窗口句柄；nlndex，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的窗口过程。函数原型为：

LRESULT CallWindowProc（WNDPROC lpPrevWndFunc，HWND hWnd. UINT Msg，WPARAM wParam，LPARAMIParam）；

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；Iparam，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 MainFrm.hwnd, True
            SetWindowLong MainFrm.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。

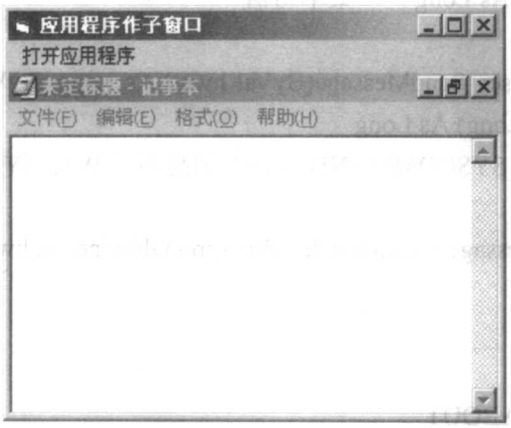


图 4-1 将其他应用程序作为本程序的一个子窗口

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
```