21世纪 高等学校计算机语言课 进阶辅导

权威推荐

Visual Basic语言程序设计

↘知识点全面覆盖
↘国内外精典实例剖析
↘典型习题拓展训练

《进阶辅导》编委会 组编

超值享受

- 全国计算机等级考试上机考试练习系统
  - 模拟考试，完全贴近真实上机考试环境
  - 仿真练习，全面熟悉考试真题
- 全国计算机等级考试笔试练习系统
  - 模拟考试，充分体验考试氛围
  - 仿真练习，通过练习历年真题掌握考试内容
- 书中涉及的源代码和资源文件

大连理工大学出版社
大连理工大学电子音像出版社

21世纪高等学校计算机语言课进阶辅导

Visual Basic 语言程序设计

组编 《进阶辅导》编委会

编著 黄 明 梁 旭 金 花

刘玉霞 郎大维 汪 静

大连理工大学出版社

大连理工大学电子音像出版社

《进阶辅导》编委会

（按笔画顺序排列）

王民生　任建娅　刘玉霞　刘晓红　李正光
李　昕　汪　静　谷晓琳　郎大维　金　花
闻丽华　梁　旭　梁　霞　黄　明　程智勇

Visual Basic 语言程序设计

《进阶辅导》编委会　组编

大连理工大学出版社
大连理工大学电子音像出版社　出版

地址：大连市凌水河　邮政编码：116024

电话：0411-84708842　传真：0411-84701466　邮购：0411-84707961

E-mail：dzcb@dutp.cn　URL：http://www.dutp.cn

大连业发印刷有限公司印刷　　大连理工大学出版社发行

幅面尺寸：185mm×260mm　印张：13.75　字数：318 千字

2005 年 7 月第 1 版　2005 年 7 月第 1 次印刷

责任编辑：高智银　　责任校对：达　理

封面设计：宋　蕾

ISBN 7-900670-44-0　　定价：23.00 元

前　言

在多年的教学实践中，我们总是听到学生反映学完“计算机语言课”后，实际编程还有很大的困难。我们认为，主要的问题就在于课内教材偏重于语句、语法的教学，实际应用方法的介绍、应用技能的培养非常有限，而练习题较简单，无法满足学生提高技能并达到实际应用的需求。为此，我们编写了这套《21世纪高等学校计算机语言课进阶辅导》丛书。

1. 内容结构

共分为两个部分：

第一部分是程序设计学习指导。按内容分章节讲述，每章中都包含知识点归纳、例题精讲、拓展训练及其参考答案。概念清晰、讲解透彻、重点突出、适用性强，知识点全面覆盖。

第二部分是程序设计上机指导。包括编译环境介绍、编程技巧、应用实例、经典游戏等内容。

2. 特点

(1)以内容难度适中、讲解深入浅出为基础，以切实帮助学生提高能力为出发点和根本目的。

(2)参加编写的人员都是多年从事计算机语言课教学、毕业设计指导、计算机语言培训的教师，他们既有丰富的实际开发经验，又真切地了解大多数学生在日常学习中的不足，所以本套丛书可以帮助学生解决学习中遇到的普遍性问题。

(3)在精选国内外实例的同时，对每个实例的解题思路均进行详细分析，对程序中的重点语句也通过注释的形式加以特殊强调。本套丛书力争做到指导性与可操作性相结合。

(4)书中的程序通过调试，可以直接在实际项目中使用。

3. 适用范围

适用范围广泛，既可以作为计算机语言课的课程设计、毕业设计辅导用书，又可以作

为学习软件开发的辅助用书，还可以作为参加计算机等级考试、软件水平考试的参考用书，同时还可以作为学员培训、程序员入门用书。

4. 配套光盘

与其配套的光盘中附有书中涉及的源代码和资源文件，可供读者调试、运行。另外，还为读者提供了最新的“全国计算机等级考试练习系统”，全真的模拟环境、历年的真题练习可以满足广大全国计算机等级考试考生的要求。

本套丛书由《进阶辅导》编委会组织编写。由于水平有限，书中错误和不妥之处在所难免，请读者和专家批评指正。

读者在使用过程中如有问题，可与编者联系：dlhm@263.net。

编　者

2005 年 7 月

目　录

第一部分 Visual Basic 语言程序设计学习指导

第 1 章

界面外观和菜单设计类

1.1 知识点归纳

Visual Basic 6.0 作为一种可视化计算机编程语言，在对用户的界面外观上日渐趋于友好，更充分地发挥了它简捷、易懂的特点。Visual Basic 6.0 对于界面处理简单且功能齐全使其成为界面设计中的佼佼者。

在 Windows 环境下，几乎所有的应用软件都通过菜单来实现各种操作。而对于 Visual Basic 6.0 应用程序来说，当操作比较简单时，一般通过控件来执行，而当要完成较复杂的操作时，使用菜单具有十分明显的优势。

在这一章里，通过几个经典的实例，讲解在界面外观和菜单程序设计上 Visual Basic 6.0 的处理技巧和方法，使你对于这方面知识的掌握更上一个台阶。

1.1.1 界面外观设计

对于相对大型的应用程序，一般只提供一个应用平台。平台的形式是一个父窗体或容器窗体。在父窗体中可以同时包含多个子窗体。Windows、Word 甚至 Visual Basic 6.0 开发平台本身就是采用的这种结构。

多文档界面(Multi Document Interface，MDI)的涵义就是创建在单个容器窗体中的多文档界面。MDI 应用程序允许用户同时显示多个文档，每个文档显示在它自己的窗口中，文档或子窗口被包含在父窗口中，父窗口为应用程序中所有的子窗口提供工作空间。

1.1.2 菜单程序设计

1. Visual Basic 6.0 中的菜单

菜单的基本作用主要有两个方面：一是提供人机对话的界面，以便让使用者选择应用

系统的各个功能；二是管理应用系统，控制各种功能模块的运行。一个高质量的菜单程序，不仅能使系统美观，而且能为操作者提供方便，并可以避免由于误操作而带来的严重后果。

在实际应用中，菜单可分为两种基本类型，即弹出式菜单和下拉式菜单。用鼠标右键单击某处时所显示的菜单就是弹出式菜单，弹出式菜单是一种小型的菜单，它可以在窗体的某个地方显示出来，对程序事件做出响应。通常用于对窗体中某个特定区域有关的操作或选项进行控制，例如用来改变某个文本区的字体属性等。与下拉式菜单不同，弹出式菜单不需要在窗口顶部下拉打开，而是通过单击鼠标右键在窗口（窗体）的任意位置打开，因而使用方便，具有较大的灵活性。

应用最广泛的就是下拉式菜单，下面就主要介绍一下下拉式菜单。

在下拉式菜单系统中，一般有一个主菜单，其中包括若干个选择项。主菜单的每一项又可以“下拉”出下一级菜单，这样逐渐扩展，用一个个窗口的形式弹出在屏幕上，操作完毕后即从屏幕上消失，并恢复原来的屏幕状态。

在用 Visual Basic 6.0 创建下拉式菜单时，可以把每个菜单项（主菜单或子菜单项）看作一个图形对象，即控件，它具有与某些控件相同的属性。

2. 菜单编辑器

Visual Basic 6.0 通过菜单编辑器来进行菜单的设计。用菜单编辑器可以创建新的菜单和菜单栏、在已有的菜单上增加新命令、用自己的命令来替换已有的菜单命令，以及修改和删除已有的菜单和菜单栏。

Visual Basic 6.0 中的菜单编辑器可以通过以下四种方法进入：

(1)执行“工具”菜单中的“菜单编辑器”命令。

(2)使用热键 Ctrl + E。

(3)单击工具栏中的“菜单编辑器”按钮。

(4)在要建立菜单的窗体上单击鼠标右键，将弹出一个菜单，然后单击“菜单编辑器”命令。

注意 只有当某个窗体为活动窗体时，才能用上面的方法打开菜单编辑器窗口。

打开后的菜单编辑器如图 1-1 所示。

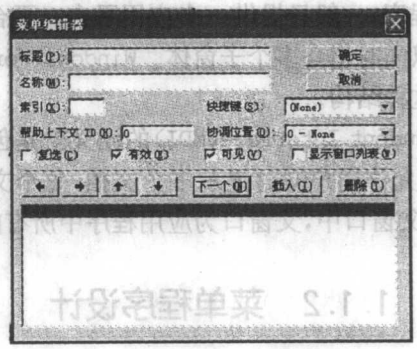

图 1-1 菜单编辑器

从菜单编辑器的界面可知，每一项菜单选项即菜单控件的标题和名称下，都对应自己

的一套属性，下面介绍一下这些属性的名称和功能。

(1)标题　是一个文本框，用来输入出现在控件上的文本(相当于控件中的 Caption 属性)。

(2)名称　是一个文本框，是代码中用来引用菜单控件的名字。

(3)索引　输入整型数据，在创建菜单控件数组时作为索引。

(4)快捷键　是一个列表框，用来设置菜单项的快捷键。

(5)复选　可以把一个复选标志放置在菜单上。

(6)可见　设置此菜单控件是否隐藏。

(7)有效　用来设置菜单项的操作状态，可以使菜单控件失效，颜色变灰。

菜单控件列表框(位于菜单编辑器的下部)列出了当前窗体的所有菜单控件。当在标题文本框中键入一个菜单项时，该项也会出现在菜单控件列表框中。从列表框中选取一个已存在的菜单控件，则关于此菜单控件的所有信息将出现在编辑窗口中，可以方便地编辑该控件的属性。

“插入”按钮用来在列表的当前位置插入一个新的菜单控件；“删除”按钮的功能是用来在列表的当前位置删除一个菜单控件；“下一个”按钮的作用是当添加一个菜单控件后，把菜单编辑器的指针移动到下一个位置，清空文本框，为输入新的菜单控件做准备。上下箭头按钮的作用是在列表框中实现上下移动选中项，调整该控件的位置；左右箭头控件的作用是改变当前菜单控件的级别，使其处于适当的菜单打开层次。

3. 创建菜单控件数组

在程序中创建新菜单是通过菜单控件数组实现的。

菜单控件作为控件的一种特殊的控件对象，同样具有数组的形式。菜单控件数组就是在同一菜单上共享相同名称和事件过程的菜单项目的集合。如果在运行时要创建一个新菜单项，它必须是窗体中现存的一个菜单数组中的成员，即必须先通过设置此菜单数组中一个元素的属性和事件处理过程，然后才能在程序运行中通过代码在菜单中添加新成员。在实际的程序设计中，创建菜单控件数组是经常使用的一种方法。例如，在许多应用程序中的文件菜单常显示当前打开的文件，而文件的数目是不定的，这是 Visible 属性无法实现的，所以必须采用菜单控件数组的方式。

创建菜单控件数组是通过对名称、标题和索引属性的设置来完成的。

4. 创建弹出式菜单

在程序设计中，单击鼠标右键弹出方便的帮助菜单是程序成熟的标志。在这里，将要介绍弹出式菜单的设计方法。

弹出式菜单是独立于菜单栏而显示在窗体上的浮动菜单。在弹出式菜单上显示的项目取决于按下鼠标右键时指针所处的位置，因而，弹出式菜单也被称为上下文菜单。

为了显示弹出式菜单，可使用 PopupMenu 方法。这个方法使用下列语法：

[object.]PopupMenu menuname[,flags[,x[,y[,boldcommand]]]]

其中，object 为当前的对象名称，menuname 为弹出菜单中菜单选项的名称，可以为多组。

5. 多级式菜单和复选菜单

在程序设计中，多级式菜单和复选菜单是经常出现的。

多级式菜单是在一级子菜单下还有二级和二级以下的子菜单。大多数应用程序都只使用一级子菜单，以保证当菜单拉下时所有控件都可见，如果菜单栏中还有空间，最好再创建一个菜单标题而不是子菜单。多级式菜单的应用范围包括：当菜单栏内容过多，必须加以分类；某一特定菜单控件很少被用到等情况。

可以通过菜单编辑器对所有子菜单进行操作。最直观的方法是子菜单的缩进量，不同级别的子菜单的缩进量是不同的，级别越低，缩进量越大。

复选菜单是由菜单控件的 Checked 属性决定的。当控件的 Checked 属性为 True 时，则菜单中对应选项前出现选中标志，否则，当 Checked 属性为 False 时，选中标志消失。通过复选菜单来标识菜单对应功能的打开和关闭状态，这一操作是需要用程序代码来保证实现的，具体过程在这里就不做阐述了。

6. 失效菜单和不可见菜单

在程序设计中，当程序运行处于特殊状态时，菜单中某些菜单的功能将无法发挥作用或根本不出现，此时应该把这些菜单设置成失效菜单或不可见菜单。

失效菜单通常在菜单功能只是暂时失效时使用，例如当剪切板中已经存放剪切内容后，则编辑菜单中的“剪切”选项将失效，这时不应该把“剪切”选项设置成不可见，因为经过“粘贴”操作后“剪切”选项就会重新生效。

不可见菜单使菜单控件不可见或产生使之无效的作用，该控件通过菜单、访问键或者快捷键都无法再访问。如果设定某菜单不可见，则该菜单上所有控件均无效。

设置失效菜单和不可见菜单的方法非常简单。只要把 Enabled 属性设置为 False 即可设置成失效菜单；而设置菜单的可见与否是通过 Visible 属性来实现的。

1.2 例题精讲

【例 1-1】 编写程序，制作一个字体窗口。在窗体中添加一幅图像作为背景，窗体界面效果如图 1-2 所示。程序运行后，将呈现出“VB6”字样的窗口。

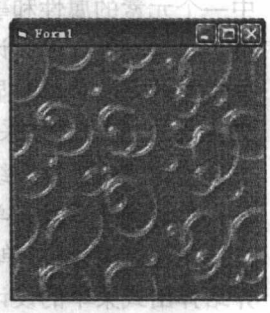

图 1-2 例 1-1 窗体界面

【分析】

本实例中将主要应用 grid32.dll 中的部分 API 函数来实现特殊窗体效果。其中 BeginPath 函数调用启动一个路径分支，使用 SetBkMode 函数设置背景为透明模式，利用 TextOut 函数设置输出文本绘图属性，使用 EndPath 函数结束定义一个路径，利用 PathToRegion 函数把所记录的路径转化为窗体轮廓句柄，使用 SelectObject 函数，利用 CreateFont 函数设置字体。另外，还应用 user32 中的 SetWindowRgn 函数设置新窗口区域。

下面是函数声明及其参数说明：

(1) BeginPath 函数功能是开始记录窗体轮廓路径。

函数声明：

Private Declare Function BeginPath Lib "gdi32" (ByVal hdc As Long) As Long

参数:hdc 参数,窗体轮廓句柄。

返回值:long,返回指定窗体轮廓句柄。

(2) SetBkMode 函数功能是设置背景模式。

函数声明:

Private Declare Function SetBkMode Lib "gdi32" (ByVal hdc As Long, ByVal nBkMode As Long) As Long

参数:hdc 参数,窗体轮廓句柄。

nBkMode 参数,窗体轮廓的背景模式。

返回值:long,返回指定窗体的背景模式。

(3) TextOut 函数功能是文本绘图。

函数声明:

Private Declare Function TextOut Lib "gdi32" Alias "TextOutA" (ByVal hdc As Long, ByVal x As Long, ByVal y As Long, ByVal lpString As String, ByVal nCount As Long) As Long

参数:hdc 参数,设备场景句柄。

x,y 参数:绘图的起点,采用逻辑坐标。

lpString 参数:欲绘制的字符串。

nCount 参数:字符串中要绘制的字符数量,一个汉字的字符数量为 2。

(4) EndPath 函数功能是结束记录窗体轮廓路径。

函数声明:

Private Declare Function EndPath Lib "gdi32" (ByVal hdc As Long) As Long

参数:hdc 参数,设备场景句柄。

返回值:函数调用成功,返回非 0 值;否则返回 0 值。

(5) PathToRegion 函数功能是将当前选定的路径转换到指定区域中。

函数声明:

Private Declare Function PathToRegion Lib "gdi32" (ByVal hdc As Long) As Long

参数:hdc 参数,设备场景句柄。

返回值:函数调用成功,返回非 0 值;否则返回 0 值。

(6) SetWindowRgn 函数功能是设定新的窗口区域。

函数声明:

Private Declare Function SetWindowRgn Lib "user32" (ByVal hWnd As Long, ByVal hRgn As Long, ByVal bRedraw As Boolean) As Long

参数:hWnd 参数,窗体句柄。

返回值:函数调用成功,返回非 0 值;否则返回 0 值。

(7) SelectObject 函数功能是将一个对象放入指定设备场景。

函数声明:

Private Declare Function SelectObject Lib "gdi32" (ByVal hdc As Long, ByVal

hObject As Long) As Long

参数:hdc 参数,设备场景句柄。

hObject 参数,被装入的对象。

返回值:函数调用成功,返回非 0 值;否则返回 0 值。

【操作步骤】

第一步:启动 Visual Basic 6.0 打开一个新的标准工程。如果 Visual Basic 6.0 已经运行,单击"新建工程"命令,创建一个新的标准工程。

第二步:选择窗体 Form1,将其属性窗口中的 Picture 属性设置为一张位图。具体操作是单击位于该属性右侧的按钮,指定位图路径即可。

第三步:选择窗体,在"查看代码"窗口中键入相应代码。

程序代码如下:

```
Private Declare Function BeginPath Lib "gdi32" (ByVal hdc As Long) As Long    '开始记录
Private Declare Function SetBkMode Lib "gdi32" (ByVal hdc As Long, ByVal nBkMode As Long) As
Long    '设置背景模式
Private Declare Function TextOut Lib "gdi32" Alias "TextOutA" (ByVal hdc As Long, ByVal x As
Long, ByVal y As Long, ByVal lpString As String, ByVal nCount As Long) As Long
Private Declare Function EndPath Lib "gdi32" (ByVal hdc As Long) As Long    '结束记录窗体轮
廓路径
Private Declare Function PathToRegion Lib "gdi32" (ByVal hdc As Long) As Long    '将当前选定
的路径转换到指定区域中
Private Declare Function SetWindowRgn Lib "user32" (ByVal hWnd As Long, ByVal hRgn As Long,
ByVal bRedraw As Boolean) As Long    '设置新的窗口区域
Private Declare Function SelectObject Lib "gdi32" (ByVal hdc As Long, ByVal hObject As Long)
As Long    '将一个对象放入指定设备场景
Private Declare Function CreateFont Lib "gdi32" Alias "CreateFontA" (ByVal H As Long, ByVal W
As Long, ByVal E As Long, ByVal O As Long, ByVal W As Long, ByVal I As Long, ByVal u As Long, By-
Val S As Long, ByVal C As Long, ByVal OP As Long, ByVal CP As Long, ByVal Q As Long, ByVal PAF As
Long, ByVal F As String) As Long    '设置字体
Private Const OPAQUE = 2
Private Const TRANSPARENT = 1
Private Const ANSI_CHARSET = 0
Private Const FW_HEAVY = 900
Private Const OUT_DEFAULT_PRECIS = 0
Private Const CLIP_DEFAULT_PRECIS = 0
Private Const DEFAULT_QUALITY = 0
Private Const DEFAULT_PITCH = 0
Private Const FF_SWISS = 32
Private Sub Form_Load()
    Dim dc As Long    '变量声明
    Dim wndRgn As Long
    Dim Font As Long
```

```
    Dim OldFont As Long
    dc = Me.hdc
    m_Font = CreateFont(200, 90, 0, 0, FW_HEAVY, 1, 0, 0, ANSI_CHARSET, OUT_DEFAULT_PRE-
CIS, CLIP_DEFAULT_PRECIS, DEFAULT_QUALITY, DEFAULT_PITCH Or FF_SWISS, "宋体")
    BeginPath dc
    '开始记录窗体轮廓路径
    SetBkMode dc, TRANSPARENT
    '设置背景为透明模式,这是必须有的
    OldFont = SelectObject(dc, Font)
    TextOut dc, 0, 0, "VB6", 3
    SelectObject dc, OldFont
    '将一个 OldFont 对象放入 dc,返回的那个
    EndPath dc
    '结束记录窗体轮廓路径
    wndRgn = PathToRegion(dc)
    '把所记录的路径转化为窗体轮廓句柄
    SetWindowRgn Me.hWnd, wndRgn, True
    '赋予窗体指定的轮廓形状
End Sub
```

图 1-3　例 1-1 效果图

第四步:运行程序,或按 F5 键,运行结果如图 1-3 所示。

【例 1-2】 编写程序,自制一个窗体放大器。创建一个标题为“智能窗体放大器”的窗体,在窗体中包括四个标题分别为“文件”、“放大”、“窗口大小”、“退出”的主菜单(Menu)和一个定时器控件(Timer),在“放大”菜单下包含四个标题为“1 倍”、“2 倍”、“3 倍”、“4 倍”的子菜单项;在“窗口大小”菜单下包含三个标题分别为“100 * 100”、“200 * 200”、“300 * 300”的子菜单项,窗体界面效果如图 1-4 所示。程序运行后,它不仅实现了放大的功能,还可以调整放大倍数、放大镜的大小。移动鼠标,可以将鼠标所处屏幕的内容放大显示。单击放大菜单,选择放大倍数,可以调整放大倍数大小,选择窗口大小,可以调整放大镜的大小。

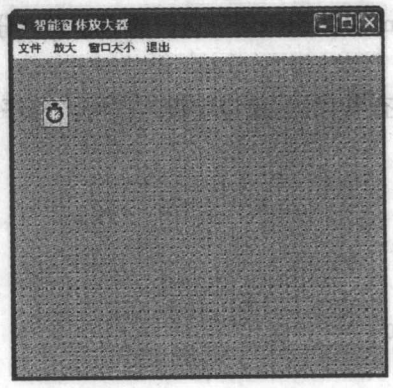

图 1-4　例 1-2 窗体界面

【分析】

本例中运用菜单编辑器建立下拉式菜单,它是一种典型的窗口式窗口。窗口是指屏

幕上一个特定的矩形区域。它可以从屏幕上消失，也可以重新显示在屏幕上，并且各个窗口之间允许覆盖。下拉式菜单自上而下在屏幕上“下拉”一个窗口菜单供用户选择或输入信息。在 Windows 及各种语言软件中，下拉式菜单得到了广泛的应用。

在 Visual Basic 6.0 中，下拉式菜单在一个窗体上设计，窗体被分为三个部分：第一部分为菜单栏(或主菜单行)，它是菜单的常驻行，位于窗体的顶部，由若干个菜单标题组成；第二部分为子菜单区，这一区域为临时性的弹出区域，只有在用户选择了相应的主菜单项后才会弹出子菜单，以供进一步选择菜单的子项，子菜单中菜单命令和分隔条，称为菜单项；第三部分为工作区，程序运行时可以在此区域内进行输出输入操作。

本例题运用了 GetDC 函数来获取窗口工作区的设备环境。下面是 GetDC 函数声明及其参数说明。

函数声明：

```
Private Declare Function GetDC Lib "user 32" (ByVal hWind As Long) As Long
```

参数：hWind 参数，获取设备的窗口句柄。如果为 0，则获取整个屏幕。

返回值：如果调用成功，返回指定窗口工作区的设备环境。否则，返回 null。

【操作步骤】

第一步：启动 Visual Basic 6.0 打开一个新的标准工程。如果 Visual Basic 6.0 已经运行，单击“新建工程”命令，创建一个新的标准工程。

第二步：选择窗体，添加定时器控件(Timer)。

第三步：在窗体中单击鼠标右键，选择菜单编辑器。添加四个主菜单，将它们的名称属性分别设置为“file”、“fdbs”、“cksz”、“exit”；它们的标题属性分别设置为“文件”、“放大”、“窗口大小”、“退出”。

第四步：选择“放大”主菜单，在它下面建立四个子菜单项，将它们的名称属性分别设置为“one”、“two”、“three”、“four”；它们的标题属性分别设置为“1 倍”、“2 倍”、“3 倍”、“4 倍”。

第五步：选择“窗口大小”主菜单，在它下面建立三个子菜单项，将它们的名称属性分别设置为“onewin”、“twowin”、“threewin”；它们的标题属性分别设置为“100 * 100”、“200 * 200”、“300 * 300”。

第六步：选择窗体，在“查看代码”窗口中键入代码。程序代码如下：

```
Option Explicit
Private Type POINTAPI
    a As Long
    b As Long
End Type
Const Srccopy = &HCC0020
Const Swp_nomove = &H2
Const Swp_nosize = &H1
Const Flags = Swp_nomove Or Swp_nosize
Const hwnd_topmost = -1
Private Declare Function SetWindowPos Lib "user32" (ByVal hWnd As Long, ByVal hWndInsertAfter
```

```
As Long, ByVal a As Long, ByVal b As Long, ByVal cx As Long, ByVal cy As Long, ByVal wFlags As
Long) As Long '设置窗体位置
Private Declare Function GetCursorPos Lib "user32" (lpPoint As POINTAPI) As Long '获取鼠标位置
Private Declare Function GetDC Lib "user32" (ByVal hWnd As Long) As Long
Private Declare Function StretchBlt Lib "gdi32" (ByVal hdc As Long, ByVal x As Long, ByVal y As
Long, ByVal nWidth As Long, ByVal nHeight As Long, ByVal hSrcDC As Long, ByVal xSrc As Long,
ByVal ySrc As Long, ByVal nSrcWidth As Long, ByVal nSrcHeight As Long, ByVal dwRop As Long) As
Long '将源窗体位置复制到目标窗体
Private Declare Function CreateEllipticRgn Lib "gdi32" (ByVal X1 As Long, ByVal Y1 As Long,
ByVal X2 As Long, ByVal Y2 As Long) As Long '创建椭圆区域
Private Declare Function SetWindowRgn Lib "user32" (ByVal hWnd As Long, ByVal hRgn As Long,
ByVal bRedraw As Long) As Long
Dim pos As POINTAPI '设置新的窗口区域
Dim s As Long
Dim sx As Long
Dim sy As Long
Private Sub Form_Load()
    setwindow
    s = 2
    sx = 100
    sy = 100
End Sub
Private Sub Form_Click()  '鼠标单击窗体
    Timer1.Interval = 0
    Me.Height = 3000
    Me.Width = 3000
    sx = 100
    sy = 100
    setwindow
    Timer1.Interval = 50
End Sub
Private Sub setwindow()
    SetWindowPos hWnd, hwnd_topmost, 0, 0, 0, 0, Flags
    Dim hr&, dl&
    Dim usewid&, usehig&
    usewid& = Me.Width / Screen.TwipsPerPixelX
    usehig& = Me.Height / Screen.TwipsPerPixelY
    hr& = CreateEllipticRgn(0, 0, usewid, usehig)
    dl& = SetWindowRgn(Me.hWnd, hr, True)
End Sub
Private Sub Timer1_Timer()  '窗体放大效果
    Dim wx As Integer
    Dim wy As Integer
```

```
    GetCursorPos pos
    wx = IIf(pos.x < 50 Or pos.x > 800, IIf(pos.x < 50, 0, 800), pos.x - 50)
    wy = IIf(pos.y < 20 Or pos.y > 800, IIf(pos.y < 20, 0, 800), pos.y - 20)
    Caption = "    坐标" & wx & "," & wy & "放大倍数=" & i
    StretchBlt hdc, 0, 0, sx * i, sy * i, GetDC(0), wx, wy, sx, sy, Srccopy
End Sub
Private Sub one_Click()  '放大1倍
    Timer1.Interval = 0
    s = 1
    Timer1.Interval = 50
End Sub
Private Sub onewin_Click()  '窗口大小
    Timer1.Interval = 0
    Me.Height = 3000
    Me.Width = 3000
    sx = 100
    sy = 100
    setwindow
    Timer1.Interval = 50
End Sub
Private Sub threewin_Click()  '窗口大小
    Timer1.Interval = 0
    Me.Height = 3000 * 3
    Me.Width = 3000 * 3
    sx = 300
    sy = 300
    setwindow
    Timer1.Interval = 50
End Sub
Private Sub two_Click()  '放大2倍
    Timer1.Interval = 0
    s = 2
    Timer1.Interval = 50
End Sub
Private Sub twowin_Click()  '窗口大小
    Timer1.Interval = 0
    Me.Height = 3000 * 2
    Me.Width = 3000 * 2
    sx = 200
    sy = 200
    SetWindowPos hWnd, hwnd_topmost, 0, 0, 0, 0, Flags
    Dim hr&, dl&
    Dim w&, h&
```