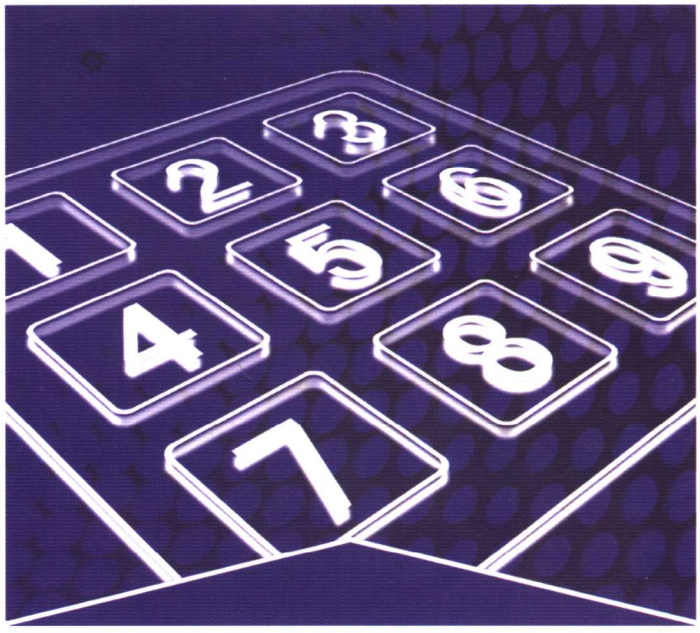
电脑编程实例导航丛书



Visual Basic 编程

典型实例解析

《电脑编程技巧与维护》杂志社 编著

中国水利水电出版社
www.waterpub.com.cn

电脑编程实例导航丛书

Visual Basic 编程典型实例解析

《电脑编程技巧与维护》杂志社　编著

中国水利水电出版社

内 容 提 要

本书是 Visual Basic 编程人员的经验之作，通过一个个典型实例解析总结了 Visual Basic 编程难点和关键技术，详尽而具体地解说了 Visual Basic 编程的思路、方法和技巧，浓缩了Visual Basic应用程序设计的精华。

本书根据 Visual Basic 的不同应用对象，将精选的 64 个应用实例分为 3 章。第 1 章为基础与应用编程实例，通过 27 个典型实例引导初学 Visual Basic 编程和应用的读者能够快速掌握Visual Basic 的编程方法和技巧；第 2 章为图形图像与数据库编程实例，通过 17 个典型实例为编程人员提供使用Visual Basic图形图像与数据库的编程方法和技巧；第 3 章为网络与通信及计算机安全与维护编程实例，通过 20 个典型实例介绍用 Visual Basic 实现网络与通信及计算机安全与维护编程方法和技巧。全书每个实例都按照设计思路、编程原理以及实例详解方法和步骤来解说。尤其是对实例的核心源代码进行了注释，每一个实例都具有很强的实用性、针对性和代表性。书中实例源程序代码可到网上下载。

全书结构清晰、层次分明、语言浅显、通俗易懂，实例典型而实用，适合于初、中级读者学习使用，对于有一定编程经验的读者也具有很好的参考价值。

本书实例源代码可以从中国水利水电出版社网站免费下载，网址为：http://www.waterpub.com.cn/softdown/。

图书在版编目（CIP）数据

Visual Basic 编程典型实例解析/《电脑编程技巧与维护》杂志社编著 .—北京：中国水利水电出版社，2007

（电脑编程实例导航丛书）

ISBN 978-7-5084-4240-2

Ⅰ.V… Ⅱ.电… Ⅲ.BASIC 语言—程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2006）第 143332 号

书　　名	Visual Basic 编程典型实例解析
作　　者	《电脑编程技巧与维护》杂志社　编著
出版　发行	中国水利水电出版社（北京市三里河路 6 号　100044） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）63202266（总机）、68331835（营销中心）、82562819（万水）
经　　售	全国各地新华书店和相关出版物销售网点
排　　版	北京万水电子信息有限公司
印　　刷	北京蓝空印刷厂
规　　格	787mm×1092mm　16 开本　22.25 印张　674 千字
版　　次	2007 年 1 月第 1 版　2007 年 1 月第 1 次印刷
印　　数	0001—4000 册
定　　价	36.00 元

丛 书 序

《电脑编程技巧与维护》杂志是为从事电脑编程、系统应用开发的人员创办的专业性和实用性都很强的技术刊物。自1994年创刊以来，始终以“实用第一，智慧密集”为宗旨，坚持“质量第一”、“读者第一”的原则，为广大电脑编程爱好者、软件开发人员和专业计算机系统维护人员提供第一手技术资料、编程技巧和维护经验；紧紧跟踪计算机软硬件技术发展和应用趋势，不断求变创新，针对软件开发过程中的许多关键技术问题着重提供各类解决方案，在业内获得一致好评，是广大编程和维护人员的首选刊物。在栏目内容上，选题覆盖面广，涉及技术领域宽、信息量大，帮助程序员开阔视野；在技术水平上，始终把握计算机技术发展的大方向，提供先进、详尽、准确的技术指导，并在长期工作中与国际性大公司建立了良好的合作关系，为读者提供全球最新、最全的实用信息；在实用性上，稿源来自于专业开发和维护人员的实践经验，是普通书籍难以获得的编程心得、体会与技巧。

2006年是《电脑编程技巧与维护》创刊十二周年，为了最大限度地开发和利用本刊宝贵而丰富的资源，更好地服务和真诚回报多年来一直关爱和支持本刊的广大读者，《电脑编程技巧与维护》杂志社和中国水利水电出版社共同策划出版了这套“电脑编程实例导航丛书”。

这套丛书包括《Visual C/C++系统开发典型实例解析》、《Visual C/C++图形图像与游戏编程典型实例解析》、《Visual Basic编程典型实例解析》、《Delphi编程典型实例解析》、《C#编程典型实例解析》、《Java编程典型实例解析》、《计算机系统安全与维护编程典型实例解析》、《计算机网络与通信编程典型实例解析》、《编程疑难问题解析126例》，一套9册共684个典型实例。每册书的编程实例均依不同的编程应用分成若干章，条目清晰可查，使用极为方便。

这套丛书选编了《电脑编程技巧与维护》杂志近两年发表的和一部分尚未发表而又极为实用、精彩的典型编程实例。该套书的特点是：其各册内容来自编程高手的智慧和经验总结，其中不少文章的作者是业界资深程序员和技术专家，内容有深度、思路有新意、讲解深入浅出，编程技巧新颖实用，构思巧妙；丛书中的实例都是作者从实际项目提炼出的开发范例，实例讲解部分先给出设计目标，然后介绍实现目标的基本思想和方法，最后详细给出其核心程序的源代码，对程序的关键部分进行讲解并给出程序的运行效果；丛书中每一个实例

的程序源代码都经过上机调试通过，对编程中的疑难问题进行了深入解答，给程序开发人员移植源代码和学用编程带来了方便，加快了编程应用的步伐。全套书既讲究内容的深入性、专业性、权威性和实用性，同时兼顾轻松、通俗易懂、时效性强的特点。

这套丛书是《电脑编程技巧与维护》资源的二次深入开发，浓缩了当前主流编程语言Visual C/C++、Visual Basic、Delphi、Java 、C#等程序设计的精华，其目的是力求为读者建造一个真正的知识整合、编程思想、编程技术、技巧交流的平台，让读者从中学习到编程高手的诀窍，丰富读者的编程技巧，拓宽读者的编程思路，迅速提升读者的程序开发能力。对电脑编程人员来说，程序开发能力的提高，除了对语言和算法的不断钻研学习、不断实践、不断总结提高，练好基本功，打好基础外，还要集思广益，善于学习，善于借鉴参考别人的经验，深入透彻地理解其中的精髓，然后溶入到自己的设计方案中去，这无疑是一条有效的学习途径，对于自身编程能力的增强和编程水平的迅速提高十分重要，这也正是我们编写这套丛书想要达到的目的。

这套丛书可作为高等院校学生进行课程项目开发、毕业项目设计的参考书，也可作为软件从业人员及编程爱好者的珍藏宝典，还可作为高等培训学校的实例教程。

《电脑编程技巧与维护》杂志社

中 国 水 利 水 电 出 版 社

2006 年 5 月

前　言

Visual Basic（VB）是 Microsoft 公司推出的一个集成开发环境，是 Microsoft Visual Studio 系列产品之一。Visual Basic 继承了早期 Basic 语言的优点，采用面向对象的程序设计技术，为我们提供了开发 Windows 应用程序最迅速、最简捷的方法。不论是 Microsoft Windows 应用程序的资深专业开发人员还是初学者，Visual Basic 都为他们提供了整套工具以方便开发应用程序。

对编程者来说，Visual Basic 虽然很容易上手，但要深入、灵活地驾驭它还要下一番功夫。平时除了对语言和算法的深入学习和透彻的理解，不断超越自我外，还需多钻研、多学习一些成功的编程案例，集思广益，充分学习借鉴别人的长处，这样无疑能较快地提高自己的编程能力和水平。

为了让更多的电脑编程人员比较集中地学习和参考 Visual Basic 应用编程的实践经验、心得体会和技巧，在《电脑编程实例导航丛书》中，《Visual Basic 编程典型实例解析》一书精选了《电脑编程技巧与维护》杂志近两年共 24 期已发表的精彩编程实例 64 例。根据 Visual Basic 的不同应用对象，将精选的 64 个应用实例分为 3 章。第 1 章为基础与应用编程实例，通过 27 个典型实例引导初学 Visual Basic 编程和应用的读者能够快速掌握 Visual Basic 的编程方法和技巧；第 2 章为图形图像与数据库编程实例，通过 17 个典型实例为编程人员提供使用 Visual Basic 图形图像与数据库的编程方法和技巧；第 3 章为网络与通信及计算机安全与维护编程实例，通过 20 个典型实例介绍用 Visual Basic 实现网络与通信及计算机安全与维护编程方法和技巧。全书每个实例都按照设计思路、编程原理以及实例详解方法和步骤来解说。尤其是对实例的核心源代码进行了注释，每一个实例都具有很强的实用性、针对性和代表性。

本书的显著特色是所选的每一个实例都是从事 Visual Basic 应用编程人员的经验总结，源于实践，其中很多编程方法和技巧可供借鉴；书中每一个实例的源代码都经过上机调试通过，给程序开发人员移植源代码带来了方便；对个别版本和开发环境稍微低一些的经典实例进行点评和分析，起到触类旁通的作用。本书的内容涵盖了 Visual Basic 的初级和高级程序设

计，读者只需要举一反三，对本书的实例稍做修改，就可以完成自己的开发任务。书中实例源代码可到网上下载。

本书是《电脑编程技巧与维护》资源的二次开发，浓缩了 VisualBasic 程序设计的精华。全书结构清晰、层次分明、语言浅显、通俗易懂，实例典型而实用，适合于初、中级读者学习使用，对于有一定编程经验的读者也具有很好的参考价值。但不足甚至疏漏之处在所难免，恳请广大读者批评指正。

《电脑编程技巧与维护》杂志社

2006 年 5 月

目　录

第 3 章　网络与通信及计算机安全与维护编程实例

第 1 章

基础与应用编程实例

例题 1　在 VB 中调用 API 函数操作指针变量

一、前言

我们用 VB 编程时会发现，VB 中没有指针变量，当需要用到指针变量的特殊处理时，VB 就显得不够用。但通过与 API 的结合，可以在 VB 中用指针变量解决问题。

在程序设计中，链表是一种重要的数据结构。要编写链表，需要用指针来完成，因此链表操作总是利用能提供指针变量的语言，例如 C、C + +、Pascal 等语言来完成。本文就以单链表的基本操作为例，介绍在 VB 应用程序中如何创造使用指针变量。

二、主要函数

1. VB 函数

VB 中有这几个函数：VarPtr(), StrPtr(), ObjPtr(), 分别用来取得变量的地址、字符串地址和对象地址，返回值均为 Long 型，例如：

```
Dim x As Integer
Dim p As Long
p = VarPtr (x)
```

称变量 p 为指针变量，p 指向变量 x。

2. API 函数

实现链表操作，用到两个 API 函数：

函数一：CopyMemory()：

```
Private Declare Sub CopyMemory Lib"kernel32 " " Alias "RtlMoveMemory " (Destination As Any, Source As Any, ByVal Length As Long)
```

该函数用来向一段内存地址复制一段数据，在传递变量时只需按值传递变量的地址即可。参数 Desination 表示目的地址，Source 表示源数据地址，Length 表示被复制数据的字节数（可用 VB 函数 LenB() 求出）。

说明：由于 CopyMemory() 在具体使用过程中不需要返回值，因此可将其声明为过程。

函数二：GlobalAlloc()：

```
Private Declare Function GlobalAlloc Lib"kernel32" Alias "GlobalAlloc " (ByVal wFlags As Long, ByVal dwBytes As Long) As Long
```

该函数用来分配一段内存区。参数 wFlags 表示分配方式，一般令 wFlags = 0，表示分配的内存区是非活动性的，dwBytes 表示要分配的内存区的字节数。

函数 CopyMemory()、GlobalAlloc() 和 VarPtr() 三者结合使用，可向分配的内存区中存放数据。如下语句可实现分配内存区 p，在内存区 p 中存入整数 5：

```
Dim x As Integer
Dim p as Long
x = 5
p = GlobalAlloc (0, LenB(x))
CopyMemory ByVal p, ByVal VarPtr(x), LenB(x)
```

要查看内存区 p 中数据，可用如下语句：

```
Dim m As Integer
```

```
CopyMemory ByVal VarPtr(m), ByVal p, LenB(m)
Print m
```

三、VB 编写链表

设链表结点结构如下（结点类型名为 node）：

其中 data 表示数据域，next 部分存储后继结点的地址。

设有两个结点 p1，p2，分配内存区如下：

```
Dim y As node
Dim p1 As Long
Dim p2 As Long
p1 = GlobalAlloc(0, LenB(y))
p2 = GlobalAlloc(0, LenB(y))
```

对结点 p1 操作如下：

```
Dim m As node
Dim x As Long
x = Val(InputBox("请输入一个数 (Long 型)", "提示", 0 + 0 + 0))
m. data = x
m. next = p2
CopyMemory ByVal p1, ByVal VarPtr(m), LenB(m)
```

对结点 p2 操作如下：

```
    y = Val(InputBox("请输入一个数 (Long 型)", "提示", 0 + 0 + 0))m. data = y
m. next = 0
CopyMemory ByVal p2, ByVal VarPtr(m), LenB(m)
```

经上述操作，可实现 p2 结点为 p1 结点的后继结点，p2 结点无后继结点，即两个结点连接起来。循环操作，即可建立整个链表。

下面的源程序可实现建立单链表、在链表中任意结点的位置插入一个结点、删除任意结点和输出链表中每个结点。

四、单链表操作的步骤

启动 VB 6.0 中文版，选择“新建/标准 EXE”，单击“打开”，进入窗体设计器，增加一个命令按钮 Command1，设置其属性 Caption 为“单击，根据提示操作”。

程序代码如下：

```
Private Declare Sub CopyMemory Lib "kernel32" Alias "RtlMoveMemory" (Destination As Any, Source As Any, By-
Val Length As Long)
Private Declare Function GlobalAlloc Lib "kernel32" (ByVal wFlags As Long, ByVal dwBytes As Long) As Long
Private Type node   '结点类型定义
data As Long
next As Long
End Type
Dim n As Integer 'n 为全局变量，表示链表中数据个数
'建立链表的函数
Private Function creat_link(n As Integer) As Long
Dim y As node
Dim x As node
Dim L As Long
```

```
Dim m As node
Dim q As Long
Dim p As Long
Dim a(1 To 100)
y. next = 0
L = GlobalAlloc(0, LenB(y)) '链表头结点
q = L
CopyMemory ByVal L, ByVal VarPtr(y), LenB(y) '头结点的 next 域为 0
For i = 1 To n '本循环正序建立链表并使最后一个结点的 next 域为 0
x. data = Val(InputBox("请输入链表中数据（整数）", "输入数据", 0))
a(i) = x. data '本语句与建立链表无关，仅是为了保存输入过的数据
x. next = 0
p = GlobalAlloc(0, LenB(x))
CopyMemory ByVal p, ByVal VarPtr(x), LenB(x)
CopyMemory ByVal VarPtr(m), ByVal q, LenB(x)
m. next = p
CopyMemory ByVal q, ByVal VarPtr(m), LenB(m)
q = p
If i = n Then MsgBox "注意：数据输入完成！请注意下面的操作提示信息!", 0 + 0 + 0, "提示"
Next
Print "请注意：你输入的数据分别是";
For i = 1 To n
 Print a(i);
 Next
creat_link = L '函数返回头结点指针
End Function
'输出链表中每一结点过程
Private Sub output(L As Long)
Dim p As Long
Dim m As node
CopyMemory ByVal VarPtr(m), ByVal L, LenB(m)
p = m. next
Do While p <>0 '因最后一个结点的指针域为0，故循环条件为：指针变量的值不为0
CopyMemory ByVal VarPtr(m), ByVal p, LenB(m)
Print m. data;
p = m. next
Loop
End Sub
'在链表 L 的第 pos 个结点的位置插入新结点的过程
Private Sub insert_link(L As Long, ByVal pos As Long)
Dim x As node
Dim p As Long
Dim m As node"
Dim q As Long
If pos < 1 Or pos >n Then
 MsgBox "注意：输入的插入位置不正确，本次插入操作无效！", 0 + 0 + 0, "提示"
Exit Sub
End If
x. data = Val(InputBox("请输入插入的数据（整数）", "输入数据", 0))
CopyMemory ByVal VarPtr(m), ByVal L, LenB(m)
```

```
q = L
i = 0
Do While q <>0 And i < pos - 1 '查找指定位置的前趋结点 q
CopyMemory ByVal VarPtr(m), ByVal q, LenB(m)
q = m. next
i = i + 1
Loop
p = GlobalAlloc(0, LenB(x))  '申请新结点 p
CopyMemory ByVal VarPtr(m), ByVal q, LenB(m) '在结点 q 的后面插入 p
x. next = m. next
CopyMemory ByVal p, ByVal VarPtr(x), LenB(x)
CopyMemory ByVal VarPtr(m), ByVal q, LenB(m)
m. next = p
CopyMemory ByVal q, ByVal VarPtr(m), LenB(m)
Print"在第"; pos; "个位置插入"; x. data; "这一结点后链表为："
output L '输出
MsgBox "注意：插入操作完成！请注意下面的提示信息！", 0 + 0 + 0, "提示"
End Sub
'删除链表 L 中第 pos 个结点的过程
Private Sub delete_link(L As Long, ByVal pos As Integer)
Dim q As Long
Dim x As node
Dim m As node
If pos < 1 Or pos>n + 1 Then
MsgBox "注意：输入的删除位置不正确，本次删除操作无效！", 0 + 0 + 0, "提示"
Exit Sub
End If
q = L
i = 0
CopyMemory ByVal VarPtr(m), ByVal L, LenB(m)
Do While m. next <>0 And i < pos - 1 '查找被删除结点的前趋结点 q
q = m. next
i = i + 1
CopyMemory ByVal VarPtr(m), ByVal q, LenB(m)
Loop
CopyMemory ByVal VarPtr(m), ByVal q, LenB(m) '以下语句删除结点 q
CopyMemory ByVal VarPtr(x), ByVal m. next, LenB(m)
m. next = x. next
CopyMemory ByVal q, ByVal VarPtr(m), LenB(m)
Print"在新链表中删除第"; pos; "个结点后链表为："
output L '输出
End Sub
'命令按钮 Command1 实现对上述过程的调用
Private Sub Command1_Click()
Dim L As Long
n = Val(InputBox("请输入链表中数据个数", "数据个数", 0))
Print"请注意：你打算输入的数据个数 N ="; n
L = creat_link(n)
Print
Print
```

```
Print"你建立的链表为："
output L
Print
Print
pos = Val(InputBox("请输入要插入结点的序号，例如输入3，表示在第3个结点的位置插入一个新结点",
"插入位置", 0))
insert_link L, pos
Print
Print
pos = Val(InputBox("请输入被删结点的序号，例如输入3，表示删除第3个结点", "删除位置", 0))
delete_link L, pos
MsgBox "全部操作完成,请退出！", 0 + 0 + 0, "提示"
End Sub
```

（李晓杰）

例题2　用函数指针访问钩子函数

一、引言

在Windows系统中，Hook(钩子)为应用程序提供了强大的系统开发功能，程序员不仅可以利用它实时监听操作系统发送给应用程序的各种消息，而且可以截取这些消息，根据用户需要编写自己的代码来改变或扩展某一特定控件或窗体的行为，从而实现对操作系统或其他应用程序的控制，完成较复杂的程序设计。

VB中，由于没有提供指针数据类型，不能直接引用回调函数，因此在使用钩子时遇到很大困难，这就使得大部分VB程序员为了实现钩子函数等复杂应用程序开发时，放弃了VB而不得不求助于VC。其实这样做大可不必，微软为了弥补VB没有指针而造成系统程序开发困难的缺陷，已在VB5尤其是VB6中提出了一套解决办法：即利用AddressOf运算符来模仿指针操作，将一个函数(主要是指回调函数)指针传入API函数中，实现对某一特定函数的引用，从而解决了长期以来VB程序员不能使用回调函数而不得不使用第三方控件的致命缺点。

下面本文结合实例，简要介绍AddressOf函数指针的用法，并结合具体的钩子API函数、AddressOf运算符和回调函数在VB中实现了访问和操作WH_CBT钩子；并用它完成了对操作系统默认消息对话框的操作，改变了消息对话框在屏幕中弹出时的显示位置。

二、基本函数和方法

1. AddressOf运算符

AddressOf是一个一元运算符，它将其后面的函数或过程的地址传递给一个API函数，并且该API函数的参数表对应位置中需要一个函数指针。

使用语法：

AddressOf FunctionName/ProcedureName

其中参数必需，用来指定要传递的地址是哪一个函数的地址，并且这个函数必须是发出调用命令的工程中的一个标准模块里的一个过程。在使用 AddressOf 关键字声明函数时，必须注意以下事项：

（1）AddressOf 只能紧接在参数列表中的参数前；该参数可以是自定义的过程、函数或属性的名字。

（2）写在 AddressOf 后面的过程、函数、属性必须与有关的声明和过程在同一个工程中。

（3）AddressOf 只能用于自定义的过程、函数和属性，不能将其用于 Declare 语句声明的外部函数，也不能将其用于类型库中的函数。

（4）在声明的 Sub、Function 或自定义的类型定义中，可以将函数指针传递到 As Any 或 As Long 类型的参数。

（5）要使用 VC（或类似的工具）编译的 DLL 中的回调函数时，DLL 中回调函数原型必须使用 _stdcall 调用约定，以使它能被 AddressOf 声明使用。

2. Hooks 介绍

在微软操作系统环境下，钩子是一种机制。通过这种机制，钩子函数不仅能够监听和拦截系统中的各种消息和事件，而且还可以改变和扩展系统中消息和事件的流程和行为。

钩子函数的原理是：当用 SetWindowsHookEx 函数创建一个钩子后，系统就在 Windows 的消息处理链中插入一个函数（通常称为过滤函数或钩子函数），钩子一旦安装成功，对应的钩子函数就可以实时监控系统消息。此时，系统中发向应用程序的消息都会先经过钩子函数，然后再交给系统处理。钩子函数共有 12 种，其中较常用的钩子主要有 WH_CALLWNDPROC、WH_GETMESSAGE、WH_CBT 等；本文程序主要用到 WH_CBT 钩子（关于钩子的详细资料，读者可以参考 MSDN）。

3. 主要 API 函数

（1）SetWindowsHookEx 函数

```
Public Declare Function SetWindowsHookEx Lib "user32" Alias _"SetWindowsHookExA" (ByVal idHook As Long,
        ByVal lpfn As Long,
        ByVal hmod As Long,
        ByVal dwThreadId As Long) As Long
```

功能：为一特定的钩子添加过滤函数。主要有 4 个参数，其中：

idHook 参数指定安装的钩子类型；

lpfn 参数为过滤函数的地址，在 VB 中此过滤函数必须在模块文件或 DLL 中定义，并且必须通过 AddressOf 函数指针来引用；

hmod 为包含过滤函数模块的实例句柄，在 Win32 中，当钩子属于特定特定线程时，此值为 0；当钩子为系统或为另一个进程的线程安装的钩子时，此值不能为 0；

dwThreadId 为将要被安装钩子的线程号；如果此值为 0，则钩入的过滤函数是系统钩子，并且能被整个系统中的所有线程调用；如果此值不为 0，则此钩子只能被某一特定线程使用。

（2）UnhookWindowsHookEx

```
Public Declare Function UnhookWindowsHookEx Lib "user32" ( _ByVal hHook As Long) As Long
```

功能：用来从钩子链中卸载过滤函数，它的参数为从 SetWindowsHookEx 函数返回的钩子句柄；返回值确定钩子是否被移去，此函数只能和 SetWindowsHookEx 函数一起使用。

（3）GetWindowLong

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

功能：获得指定窗口的信息。主要有两个参数，其中：

hwnd 指定窗口的句柄；

nIndex 可取 7 个值，函数获得的窗口信息随此值不同而变。以下是一些取值情况：

GWL_EXSTYLE：函数返回扩展窗口风格；

GWL_STYLE：函数返回窗口风格；

GWL_HINSTANCE：函数返回应用程序实例句柄；

GWL_ID：函数返回窗口的资源号。

（4） GetCurrentThreadId

Public Declare Function GetCurrentThreadId Lib "kernel32" () As Long

功能：返回当前线程的 ID 值。

三、程序实现

本例中，应用程序包括一个简单的窗体，其中只有 3 个命令按钮。代码的作用是为应用程序安装一个 WH_CBT 钩子，用来截取发送到消息对话框窗体的 HCBT_ACTIVATE 消息，在消息对话框激活显示时改变它的系统默认位置。

本程序在 Windows 98 环境下采用 VB 6.0 编程语言，具体步骤如下：

（1）启动 VB 6.0，建立一个标准 EXE 工程，Form1 采用缺省设置。

（2）为工程添加一个标准模块(MyHook.bas)；在模块中加入如下代码：

```
'定义一个 RECT 结构
 Type RECT
    Left As Long
    Top As Long
    Right As Long
    Bottom As Long
  End Type
'在模块中声明如下 API 函数
  Public Declare Function GetWindowLong Lib "user32"
Alias "GetWindowLongA" (ByVal hwnd As Long, ByVal
nIndex As Long) As Long
  Public Declare Function GetCurrentThreadId Lib "
kernel32" () As Long
  Public Declare Function SetWindowsHookEx Lib "user32"
 Alias "SetWindowsHookExA" (ByVal idHook As Long, ByVal lpfn As Long, ByVal hmod As Long, ByVal
dwThreadId As Long)
  As Long
  Public Declare Function UnhookWindowsHookEx Lib "
user32" ( ByVal hHook As Long) As Long
  Public Declare Function SetWindowPos Lib "user32" (
  ByVal hwnd As Long, ByVal hWndInsertAfter As Long,
  ByVal x As Long, ByVal y As Long, ByVal cx As Long,
  ByVal cy As Long, ByVal wFlags As Long) As Long
  Public Declare Function GetWindowRect Lib "user32"
(ByVal hwnd _ As Long, lpRect As RECT) As Long
'定义 API 函数中用到的所有常量
  Public Const GWL_HINSTANCE = (-6)
  Public Const SWP_NOSIZE = &H1
  Public Const SWP_NOZORDER = &H4
  Public Const SWP_NOACTIVATE = &H10
  Public Const HCBT_ACTIVATE = 5
```