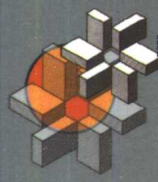
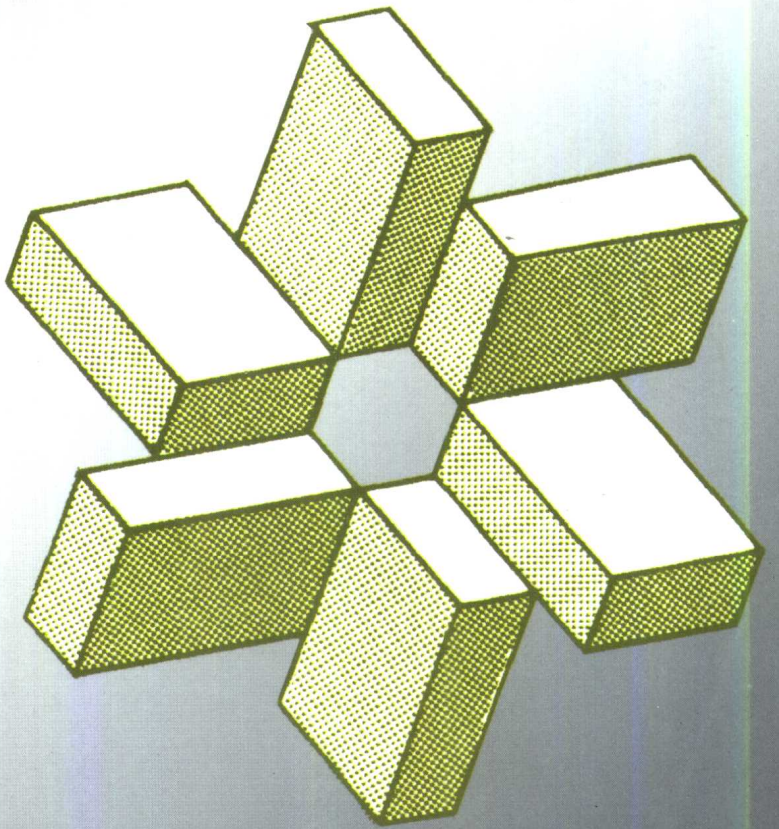


看实例学编程系列丛书

Visual Basic 6.0 API

函数开发实例

同志工作室　编著



人民邮电出版社
www.pptph.com.cn

看实例学编程系列丛书

Visual Basic 6.0 API 函数开发实例

同志工作室　　编著

人 民 邮 电 出 版 社

内 容 提 要

Visual Basic 6.0 是美国 Microsoft 公司开发的 Microsoft Visual Studio 套件的一部分，是运行于 Windows 平台上的交互式的可视化集成开发环境。本书从 API 编程基础开始，以示例的形式全面介绍了 Visual Basic 6.0 支持的 API 函数以及 API 函数的应用，涵盖了字体、文本、图形、高级绘图、图像处理、窗口、菜单、系统信息控制、消息控制等各个方面，揭去了 API 函数的神秘面纱，带领读者进入 Windows 程序开发的内部。

本书通俗易懂，示例丰富，讲解细致，分析透彻，适合于中级程序开发人员学习使用，对于从事 Visual Basic API 函数开发与应用的广大科研人员、高校相关专业的师生也不失为一本有价值的自学和教学的参考书。

看实例学编程系列丛书

Visual Basic 6.0 API 函数开发实例

◆ 编　　著　同志工作室

责任编辑　姚予疆

◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 14 号

邮编　100061　　电子函件　315@ pptph.com.cn

网址　http://www.pptph.com.cn

北京顺义向阳胶印厂印刷

新华书店总店北京发行所经销

◆ 开本：787 × 1092　1/16

印张：27.25

字数：682 千字　　2000 年 12 月第 1 版

印数：5 001 – 8 000 册　　2001 年 2 月北京第 2 次印刷

ISBN 7-115-09030-0/TP·2001

定价：41.00 元

编 者 的 话

面向对象技术近年来发展迅速，它被广泛地应用到计算机研究与应用的各个方面，如文件处理、操作系统设计、多媒体技术、网络与数据库开发等。用面向对象技术进行程序设计、开发软件已经成为一种时尚。这种技术从根本上改变了人们以往设计软件的思维方式，从而使程序设计者可以最大限度地摆脱烦琐的数据格式和冗长的研发过程，将精力集中在对要处理的对象的设计和研究上，大大提高了软件开发的效率。为了满足初中级 Windows 程序开发人员、大专院校相关专业师生及业余爱好者学习和应用各种流行程序设计软件的需求，我们同志工作室经过多方调研，在收集了不同层次读者意见的基础上，经过仔细研讨，于 2000 年 5 月份推出了《看实例学编程》系列丛书的前 5 本。

《看实例学编程》系列丛书介绍的软件都是国内外著名软件公司的知名产品，也是国内应用面最广的软件。本套丛书一改以往计算机编程图书枯燥的风格，将软件开发技术融合到程序示例中，采用了由实际到理论、由具体到抽象的逆向写作思路。丛书一经推出，就立即得到了广大读者的好评，同时，也有不少读者建议，能不能以这种方式更深入地介绍软件开发的各专项领域。为了满足广大读者的需求，我们同志工作室的全体成员经过多方讨论，又精心策划了下面 10 本专题类图书，收入本套丛书内。它们是：《Delphi 5 数据库开发实例》、《Visual Basic 6.0 数据库开发实例》、《Visual C++ 6.0 数据库开发实例》、《C++ Builder 5.0 数据库开发实例》、《Delphi 5 API 函数开发实例》、《Visual Basic 6.0 API 函数开发实例》、《C++ Builder 5.0 API 函数开发实例》、《Delphi 5 多媒体开发实例》、《Visual Basic 6.0 多媒体开发实例》及《C++ Builder 5.0 多媒体开发实例》。

这 10 本书秉承了前 5 本书的特点，但它更侧重于软件开发的具体领域。例如，数据库、多媒体和 API 函数，而不是广泛地学习软件各个方面的知识；不是繁琐冗长的使用手册或枯燥乏味的大本参考书，而是独具实效的实例指南。这 10 本书准确地告诉读者用程序设计软件可以做哪些开发工作以及如何做这些开发工作，内容充实、讲解细致、分析透彻，笔调亲切，绝没有居高临下的架势。而且，我们在编写的过程中尽量省去了枯燥难懂的专业术语，以平和易懂的语言带领大家逐步进入到编程的艺术天堂。这些书以计算机中级程序开发人员为主要的读者对象，为便于读者理解，我们根据自己学习和使用的体会精心挑选了大量的实例，这些实例都是针对程序员在开发过程中最需掌握的技术而特意定制的，能较好地满足读者的需求。

同志工作室

前　言

Visual Basic 6.0 是美国 Microsoft 公司开发的 Microsoft Visual Studio 套件的一部分，是运行于 Windows 平台上的交互式的可视化集成开发环境。像其他的可视化集成开发环境（如 Visual C++、Delphi、C++ Builder）一样，Visual Basic 6.0（为了叙述方便，以下简称为 VB6）集程序的代码编辑、编译、连接和调试于一体，给编程人员提供了一个完整方便的开发界面和许多有效的辅助开发工具。VB6 的应用程序向导可以为很大一部分类型的程序提供框架代码，用户不用书写程序代码，只需按几个按钮就可以生成一些完整的可以运行的程序。

本书从 API 编程基础开始，以示例的形式全面介绍了 Visual Basic 6.0 支持的 API 函数以及 API 函数的应用，这些示例程序都是作者根据自己使用和开发 VB6 程序时的体会精心挑选的，是针对程序员在开发过程中需要最迫切、使用频率最高的内容特意定制的，可以说比较贴切地符合了初级和中级程序员的需求。另外，本书中所有示例程序的代码都经过了严格的调试和测试，读者只要按照书中的步骤做，最终一定能够圆满地完成程序。

第 1 章是本书的基础部分，对 API 函数与 VB 的关系、API 函数的声明和参数、Windows 环境和 VB 环境的不同做了详细的介绍，这是调用 API 函数的基础知识。最后，通过一个简单的字符大小写转换程序，演示了 API 函数的调用方法。API 函数因其声明的复杂性，使没有接触过的读者感到很难，希望本书的第 1 章能使读者消除“API 很神秘”的印象。

第 2 章介绍与字体和文本有关的 API 函数。利用 VB 自身的功能，很难实现添加字体、删除字体和创建字体的功能，但 API 函数提供了这些功能，通过简单的代码即可以实现。利用 API 函数还能实现 VB 无法做到的使一行字符倾斜（不是水平）的功能，在本章的旋转字体程序中，读者可以领略到 API 函数的神奇作用。

在 VB 中提供了一些绘图的方法，如 Line、Circle、Pset 等，API 函数也有相应的绘图函数，并且可以实现更强大的功能。利用 VB6 支持的 API 函数，配合其他的一些函数，完全可以实现像 Windows 的“附件”——画图那样的程序，甚至可以做得更好。第 3 章通过一个示例程序向读者说明了利用 API 函数绘制图形的技术。

在接触 API 函数之前，我们不能够定制绘图环境，仅仅能够绘制基本图形。通过第 4 章的学习，读者可以更加灵活地处理图形应用程序，本章综合利用 API 函数制作了示例程序，希望读者认真体会，以从中体会到 API 函数强大的功能。

在第 5 章中主要介绍了创建位图、装载位图、设置鼠标形状、创建图标、装载图标、图像处理和多媒体函数。其中，图像的处理和用 API 函数播放多媒体文件比较常用，希望读者仔细阅读程序示例，掌握用 API 函数处理图像和播放多媒体文件的技术。

第 6 章介绍了文件的创建、打开、修改、关闭，系统目录和驱动器信息的取得或更改、创建以及有关注册表的操作。本章内容较多，较抽象，可以加深对 Windows 操作系统的了解。

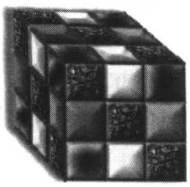
第 7 章中介绍了控制窗体和菜单的 API 函数，主要内容有窗体之间的关系、排列窗口、获得及改变窗口的状态、窗体与矩形、获得菜单属性、添加与删除菜单和设置菜单等。窗口和菜单是 Windows 操作系统中重要的部件，在试着使用这些函数时，是不是有一种恍然大悟的感觉呢？

第 8 章介绍了与 Windows 系统有关的函数和消息，包括鼠标信息的读取和设置、剪贴板操作、系统信息的读取和设置以及消息传递，通过这些函数和消息，读者可以对 Windows 进行一些简单的定制。当然，如果能够把功能强大的 API 函数与简单易用的 VB 结合起来的话，一定能够使编程水平更上一层楼。

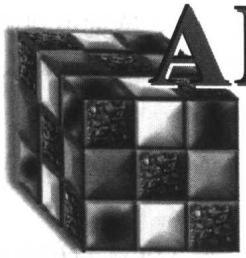
本书由张晓玲、王兴晶、王洪岩、安志敏、王兴艳等同志编写。由于编写时间紧张，作者水平有限，书中难免存在一些不足之处，恳请读者批评指正。

编著者

目　　录



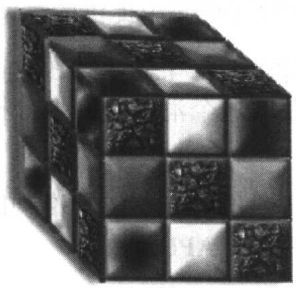
第 1 章
API 编程基础

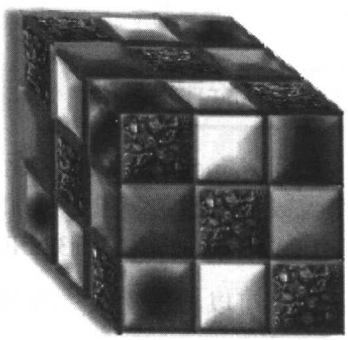


API 简介

API 基础

API 函数示例





什么是 API?

API(Application Programming Interface)即应用程序编程接口，是一些用 C 语言编写由操作系统自身调用的函数，用来控制 Windows 的各个部件外观和行为。

用户的每个动作（单击鼠标、输入文本等等）都会引发一个或几个 API 函数的运行，以此来告诉 Windows 发生了什么事情。

API 与 Visual Basic

在 Visual Basic（以下简称 VB）环境中编程，我们完全感觉不到调用了 API 函数，例如，当用鼠标单击窗体上的一个命令按钮时，会引发这个命令按钮的 Click 事件，执行 Click 事件中的代码完成相应的功能；我们并不需要 API 函数。

其实这是因为 VB 做了很多工作，它完全隐藏了 API，提供了在 Windows 环境下编程的一种完全不同的方法。

这也就是说，用 VB 写的每行代码都会被它转换为 API 函数传递给 Windows。例如：

```
Form1.Print...
```

VB 将会以一定的参数（代码中提供的，或是默认参数）调用 TextOut()这个 API 函数。

同样，当你单击窗体上的一个命令按钮时，Windows 会发送一个消息给窗体（这对于你来说是隐藏的），VB 获取这个消息并经过分析后生成一个特定事件(Button_Click)，并执行你写在事件中的代码。

通常，只使用 VB 提供的定制好的标准功能便可以满足要求，但是当你想做一点标新立异的事情时，就需要“呼叫”API 函数了。

VB 一个强大的特性是它具有调用驻留在动态链接库(DLL)文件中的函数的功能，可以在应用程序中利用 Windows API 提供的数百个 API 接口进行扩充，加速应用程序的建立，减少程序开发的重复性。

API 函数的声明

由于 Windows API 函数不是 VB 内部函数，所以在使用它们之前必须显式地加以声明，API 函数用到的常量也必须加以声明。API 函数包含在 Windows 系统目录下的动态链接库文件(DLL)中，例如，User32.dll、GDI32.dll、Shell32.dll 等。

用户可以自己输入 API 函数的声明，但是，VB 提供了一种更简单的方法，就是使用 API Viewer 工具。

想在工程中声明 API 函数，只需运行 API Viewer，打开 Win32api.txt，选择 Declares（声

明），找到所需函数并选中，单击 Add 按钮，再单击 Copy 按钮，然后将其粘贴到工程里，使用预定义的常量和结构（即类型）也是同样的方法。

在以后的章节中会演示用 API Viewer 声明函数。

这样做，也许遇到一些问题：

- 假设想在窗体模块中声明一个函数，按照上述方法粘贴然后运行，VB 会出现编译错误（Declare 语句不允许作为类或对象模块中的 Public 成员），看起来情况很糟糕，其实你需要做的是在声明前面添加一个 Private，如 Private Declare Function。

如果先把函数声明为 Private，而不是 Public 成员，可以在 API Text Viewer 中在单击 Add 按钮前，选中 Declare Scope 的 Private 项，这样函数的声明就是 Private Declare Function...形式。

- 有时候，可能得到“不明确的名称”这样的提示，这是因为函数或常量与其他的什么共用了一个名称，可以通过 Alias 子句使用其他的而不是原有的名称，这样函数仍然可以正常运行。

API 函数声明详解

先看一个 API 函数的声明。

```
Privat[Public] Declare Function AbortSystemShutdown Lib"advapi32.dll" Alias
 "AbortSystemShutd ownA"
                    (
                     ByVal lpMachineName As String
                     ) As Long
```

下面解释一下这个函数声明的各个部分：

- Public：表示所有模块的所有过程都可以调用这个函数。声明函数时，要在 Private 和 Public 中选择一个，但有时只能将其声明为 Private 成员，在本书中，均将其声明为 Private 成员；
- Declare Function：声明对动态链接库(DLL)中函数的引用；
- AbortSystemShutdown：函数的名称，通常名称可以反映函数的功能；
- Lib "advapi32.dll"：表明这个函数是 advapi32.dll 动态链接库中的；

现在不断有新的 API 出现，处理新的操作系统扩展，比如 E-mail、联网和新的外设。

- Alias "AbortSystemShutdownA"：Alias 表明为函数起个别名，AbortSystemShutdownA 就是这个函数的别名，也可以起别的名称，以避免与已有的函数或变量的名称发生冲突；
- ByVal：函数参数传递的类型，ByVal 表示传递的是参数的值，如果这里是 ByRef，

那么传递的是参数的地址;

- lpMachineName As String: 参数，并且指明参数的数据类型为 String（字符串）;
- As Long: 数据类型定义的关键字 As Long 处在这个位置，表示函数的返回值为 Long 型数据。

常用到的动态链接库有：

- gdi32.dll: 图形显示界面的 API;
- kernel32.dll: 处理低级任务（比如内存和任务管理）的 API;
- user32.dll: 处理窗口和消息（VB 程序员能把其中一些当作事件访问）的 API。

API 函数的参数

如果你使用 API 函数，肯定注意到了函数的返回值或参数有很多奇怪的数据类型，如 VOID、LPCSTR、和 DWORD。在 API 函数中，有如下一些类型的返回值或参数：

- Bool 类型: 它的值为 Long 型而不是 Boolean，0 表示 False（假），其他任何值表示 True（真）;
- hWnd，hDC，hMenu 等类型: 它们都以 h 开头的，表示不同对象的句柄;

例如，**hBitmap** 表示一幅位图的句柄，**hBrush** 表示一个刷子的句柄，**hwnd** 表示窗口的句柄，**hMenu** 表示菜单的句柄等等。它们均为 **Long** 型，而且要按值传递**(ByVal)**。

- 以 LP 开头的类型: 有些类型以 LP 开头，LP 是 Long Pointer 的缩写，例如，LPWORD 实际表示数据所在的内存地址;

不过，并没有必要调用某个函数来获取这个地址，当按引用**(ByRef)**传递参数时,实际上传递的就是它的地址。所以，如果某个参数的类型以**"LP"**开头，应该按引用**(ByRef)**传递。

- NULL 类型: 这是一种奇怪的类型，在大多数情况下，可以用 ByVal 0 或 vbNullString 传递;
- VOID 类型: 是指那些没有返回值的函数。

如果一个函数被声明为 **VOID**，必须在 **VB** 中把它声明为 **Sub** 而不是 **Function**，以此表明该函数没有返回值。

还有一些常见的类型，如 SHORT、INTEGER、LONG，鉴于读者对此应该比较熟悉，就不详细介绍了。

下面主要介绍一下 Any 数据类型。

有些函数的参数声明为 Any 类型，这表示该参数是一种可变的类型（可以以整型,字符串或自定义类型来传递）。

例如，某一参数在函数声明中是 Any 类型，在实际调用函数时，想以 Long 类型来传递这个参数，就在这个参数的值前加上 ByVal。

下面列出了一个 Any 数据类型，在实际调用中使用不同数据类型时的传递方式：

- Long 型：ByVal（按值传递）；
- String 型：ByRef（地址传递）；
- Array 型：ByRef（地址传递）。

Windows 环境与 Visual Basic

在 Visual Basic 中，用户使用 Name 属性来标识每一个控件；而 Windows 通过句柄（Handle）识别每个窗体、控件、菜单、菜单项或其他任何你能想得到的东西。当程序运行时，它所包含的每个部件都有一个唯一确定的句柄用来同其他的部件相区别。例如，某个命令按钮的句柄就与其他所有部件不同，当通过 API 来执行有关该按钮的某种操作时就必须使用这个句柄。

句柄就像每个人的身份证，在一个国家内绝不会重复，是代表这个人的唯一有效证件。

从哪里得到控件的句柄呢？VB 为每个拥有 Windows 句柄的控件都提供了 hWnd 属性来表示其句柄。

在调用 API 函数时，有些函数参数需要控件的 hDC。

稍加注意，就可以发现，在 VB 环境中，每添加一个窗体或控件，系统就会自动为其设置 hDC 属性的值，需要时只需读取即可。例如，在窗体 From1 上有一个图片框控件 Picture1，取得它的 hDC 属性用如下代码：

```
Dim pichdc As Long
pichdc = Form1.Picture1.hDC
```

有一些控件 VB 并没有给提供 hDC 属性，但提供了 hWnd 属性，这时，可以通过调用 GetDC()这个 API 函数得到控件的 hDC 值。

GetDC()函数只需一个参数，即控件的 hWnd 属性值。例如，在窗体 From1 上有一个文本框控件 Text1，取得它的句柄用如下代码：

```
Dim txthdc As Long
txthdc = GetDC(Form1.Text1.hWnd)
```

hWnd 和 hDC 属性在设计阶段不可用，只能在程序运行中使用，且只读。所以在 VB 的设计阶段，不能从属性窗口中看到这两个属性，只能在编写代码时使用。

在本书中，常常用到这样一个词——设备。所谓“设备”，在这里指窗体和各种控件，API 函数常常需要“设备”的句柄，用来确定操作的对象。

Windows 使用像素(Pixel)而不是缇(Twip)来描述屏幕属性，因此，把涉及 API 函数调用控件的 ScaleMode 属性设为 3--(Pixel)是个不错的主意，这样你可以通过 ScaleWidth 等属性得到它们的公制单位值。尽管这样，你可能有时仍需要进行从 Twip 到 Pixel 的转换（反之亦然）。

另外需要提到的是，Windows API 函数中用到了不同的坐标系统，因此需要注意。

VB 并不能识别 API 调用中的错误，因此一旦你的程序出现异常，要先检查 API 调用是否缺少 ByVal，或者是使用了错误的参数类型等。

一旦你使用了 Windows API 函数，VB 就可能不再可靠了，因为 API 调用中一个简单的语法错误就会导致程序崩溃！

1-2 API 函数的一个简单应用

在一般的应用程序设计过程中，经常涉及到字符的大小写转化问题，在程序的设计阶段可以通过直接对属性赋值来完成，但是在程序运行的过程中，如何才能够动态的进行字符串大小写之间的转换呢？通过调用 API 函数可以实现。下面先介绍实现字符大小写转换的四个函数，它们的函数名称和能够实现的功能如下所示：

- CharLower(): 将指定的字符或字符串转换成相应的小写字符;
- CharLowerBuff(): 将指定的字符串的指定个数的字符转换成相应的小写字符;
- CharUpper(): 将指定的字符或字符串转换成相应的大写字符;
- CharUpperBuff(): 把指定的字符串的指定个数的字符转换成相应的大写字符。

值得注意的是，在程序中调用 CharLowerBuff()函数和 CharUpperBuff()函数进行字符串的大小写转换时，要为函数指定两个参数——String 型变量 lpsz、Long 型变量 cchLength，其中前者指定了将要转换的字符串，后者指定进行转换的字符个数。

下面就以一个示例来说明在程序中如何调用 CharLower()函数、CharLowerBuff()函数、CharUpper()函数和 CharUpperBuff()函数进行字符串的大小写转换，具体的程序设计步骤如下所示。

设计界面

首先选择菜单 File 中的 New Project 选项，在屏幕上就会弹出一个如图 1-1 所示的 New Project 对话框。

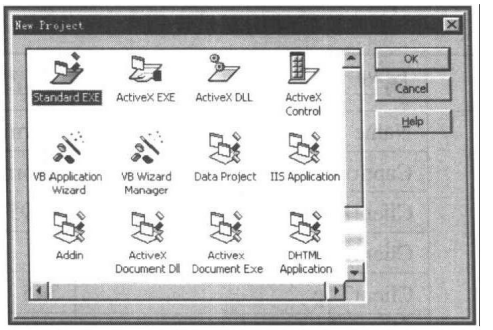


图 1-1　New Project 对话框

在 New Project 对话框中选择 Standard EXE 选项，然后单击 OK 按钮，VB6 的程序设计环境中就新建了一个标准的项目文件，同时在屏幕上就会出现一个空白的窗体，在它的上面用户可以添加自己的控件。

在项目文件所打开的空白的窗体上放置两个 Label 控件、两个 TextBox 控件和四个 ButtonCommand 控件。

添加控件后窗体如图 1-2 所示。

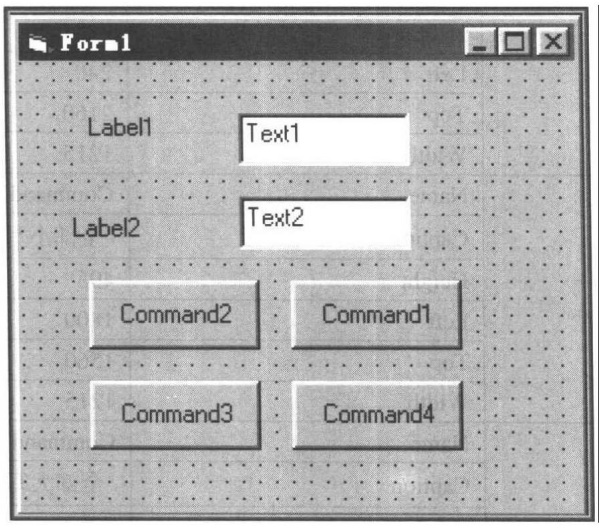


图 1-2　添加控件后的窗体

设置控件属性

在程序的设计过程中，用鼠标的左键选中窗体上的控件，在键盘上按功能键【F4】，这时在屏幕上就会弹出一个控件属性设置窗口，在其中对窗体和控件的属性进行设置，如表 1-1 所示。