第一部分

Visual Basic for Windows 入门

第一章 BASIC 简史

BASIC 程序设计语言由 John Kemeny 和 Thomas Kurtz 于 1963 年在 Dartmouth 学院创立。他们设计 BASIC 是为了程序设计概念的教学，放以速度和效率为代价，着重程序的简洁性。其实现方法利用了作业控制语言以及在其他编程语言，如 FORTRAN 和汇编语言中构造程序时所需的编译/连接步骤(实际上，BASIC 系统本身就是一个编译过的程序。它负责解释用户输入的正文行。该系统每次读入、解释和执行一行程序代码。因此，BASIC 被称为"解释"语言)。BASIC 允许用户集中精力考虑编程任务所需的方法和算法，而不必考虑计算机硬件的具体操作细节。

BASIC 的早期版本具有若干明显的特征。程序的每一行以行号开头，语句通常不缩格(现在，为提高程序的可读性，语句通常缩格)。所有的字符均以大写字母输入和显示。用来转移程序控制权的 GOTO 和 GOSUB 语句均以行号作为目的地。

这些特征鼓励了用户书写难读的"意大利面条式代码"。也就是说，程序的逻辑流程经常迂回和分支，就象柔软的意大利面条搁在碟子上一样。图 1.1 给出了一段典型的意大利面条式代码。即使在这个简短的程序中，沿控制流"走"下去也极易"迷路"。幸运的是，BASIC 从那时起已走过了相当的历程了。

早期的 BASIC 获得了玩具语言的名声，它并不适合现实世界中的编程任务。不过，现在的 BASIC 已经从一种慢速的、非结构化的解释语言进化成一种快速的，结构化的编译语言，适合于建立各类应用程序。Hewlett-Parckard，Microsoft 和其它许多公司已经推出了具有高级功能的 BASIC 增强版本。对于很久不用 BASIC 的用户来说，现在是对该语言刮目相看的时候了！

```
10 REM--PRIME NUMBERS LESS THAN 100
20 N=N+1
30 IF N=100 THEN GOTO 120
40 I=1
50 I=I+1
60 J=N/I
70 IF INT(J)=J THEN GOTO 20
80 IF I>SQR(N) THEN GOTO 100
90 GOTO 50
100 PRINT N,
110 GOTO 20
120 GND
```

图 1.1 意大利面条式代码的早期 BASIC 程序

1.1 BASIC 的进展

BASIC 紧随个人计算机的发展而发展。70 年代，Microsoft 率先为早期的个人计算机

引入了基于 ROM 的解释型 BASIC。例如，Radio Shack TRS-80 向公众介绍了 BASIC(和个人计算机的概念)。Microsoft BASIC 的这个原始版本至今仍以 GW-BASIC 的形式存在，未经多少修改。4.01 版及更早的 MS-DOS 均提供了解释型的 GW－BASIC。

尽管 GW BASIC 能够解决快速计算等简单问题，但它更适合于充当玩具语言。没有一个软件开发者会想用 BASIC 来编写商用软件，就象 MS-DOS 实用程序不会采纳批文件形式一样。原因是程序的执行速度很慢，而且必须把源代码提供给用户。现在已经有了更好的程序开发方法。

1982 年，Microsoft QuickBasic 革新了 Basic，使之真正成为 MS-DOS 环境下的开发语言。QuickBasic 把 GW-BASIC 的交互性和多产性与编译语言的多功能和高速度等特性有效地结合在一起。行号不再是必需的，并增加了现代语言的某些特征(如子程序和用户定义的结构数据类型)。QuickBasic 的图形和声音功能超过了 C、Pascal 和 Fortran。QuickBasic 程序还有一个优点：程序既可以在交互式的解释模式下运行，也可以编译成可独立执行的程序，以供商用。

1.2 Visual Basic

当前正在进行的 Microsoft Windows 革命，提供了功能强大而且是标准化的环境来更充分地发挥 Intel 公司最新微处理器的能力。对用户而言，Windows 是全新的；它使得个人电脑更个人化并且更具友好性。但是对程序设计人员而言，他们必须强迫自己又去学习一整套新的程序设计概念，这样才有可能更有效地开发基于 Windows 的应用程序。但是，Visual Basic 改变了这种情形，它是 Basic 语言的最新最大的一次改进，现在学习给 Windows 开发应用程序已变成了一件激动人心的事情，而程序设计工作也变得更具吸引力，更有效率，甚至更有趣！

过去的十年中，Basic 语言已改进了许多地方，而 Visual Basic for Windows 3 也正是按这种趋势发展的。目前，数以千计的基于 Windows 的应用程序是用 Visual Basic 的老版本开发成功的，而以后更多的应用程序则会用最先进的 Visual Basic 3 开发。在下一章中，我们将研究一下 Visual Basic 的一些独特的功能，这样就会清楚到底它的妙处何在。

第二章 Visual Basic for Windows 的特点和优点

Visual Basic for Windows 的下列特点使其成为 Windows 下的理想开发语言。这些特点不仅加快了程序开发的速度，而且提供了开发复杂应用程序所需的全部工具。本章将着重介绍这些特点。

2.1 对 QuickBasic 的改进

Visual Basic for Window 保留了许多 Microsoft Quick Basic 的优点，同时针对基于 Windows 的应用程序的开发增加了许多功能。例如，用户可以很容易地把图形直接输出到窗口的
任一部分，甚至在打印机上打印出来；可以从超过一千六百万种以上的颜色中选取图形对象的颜色(Windows 可以把像素混合提供近似的颜色，或者在硬件支持下生成一种颜色。无论哪种情形，用户都不必关心这些细节)。

相对于 Quick Basic，Visual Basic 所作的另一个改进是变量局部化的方式。这些规则容易理解和记忆，因为它们是经过改进的，而且说明得较详细。用 Visual Basic 开发的应用程序可以包含两种类型的文件：窗体，其中包含有关窗口和对话框的可见的代码；模块，其中只包含源代码。在一个窗体或模块中的通用定义段中定义的常量和变量，对于其中的任何一个子程序或函数都是可见的。在一个模块中定义为 Global 的变量或常量，对于本项目中的任一个窗体或模块都是可见的。除非在别处已定义成全局量，一般情况下，在子程序和函数中定义的变量和常量都局限于它们所在的例程中(等用户稍微有了一些 Visual Basic 的应用经验后，这些规则就显得更清楚了)。

图 2.1 列出了一个应用程序的代码。当用户在应用程序的窗口中按下鼠标按钮时，这个程序会显示下列信息：

3.141593　　6.283186

注意：常量 PI 在本项目中任何地方都是可见的，TWOPI 在 Form 1 中任何地方也是可见的，但 THREEPI 只在子程序 Form_Load 中可见(从 Visual Basic 2 开始，未定义的变量的类型是 Variant 类型。在下面的例程 Form Clide 中，THREEPI 被解释为一个空变量，也就是如果要显示它的值，将不会产生任何动作。在 Visual Basic 1 中，将显示 0)。

图 2.1 所示的代码看起来可能有些陌生，但不要着急，稍后我们还会讨论这个例子，到时自然就会明白。要记住的重点是我们可以根据变量和常量在项目中的位置，很容易地确定它们的可见性(可访问性)。

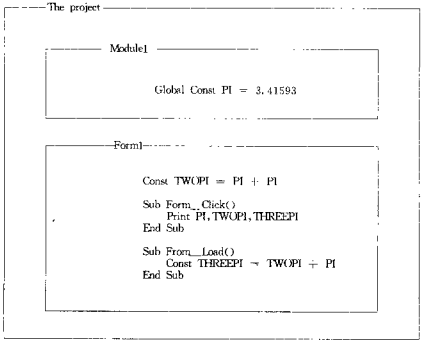

图 2.1 变量范围的例子

Visual Basic for Windows 的优点之一就是用户可以在短时间内成为一名多产的程序员。熟悉 QuickBasic 的用户很快会意识到 Visual Basic for Windows 的这个特点。用其它程序设计语言,如 C,开发基于 Windows 的应用程序时,用户不得不理解和掌握大量的编程方法,而这两种 Visual Basic 版本的交互性和直观性将使用户深受其益。目前,若想加快 Windows 应用程序的开发速度,最快速的方法就是学会用 Visual Basic 编程。

即使是经验丰富的 C 语言程序员也会欣赏 Visual Basic 的交互性界面的开发周期。在这一方面,它优于其它编程语言。例如,运行一两行 GW-BASIC 代码比在文档中找到相应的语句还要快速方便。现在,Visual Basic for Windows 利用这种交互开发概念,为应用程序创建复杂的用户界面。在界面的设计过程中,能够获得快速的视觉反馈是最为重要的。一般来说,很难把用户界面设计过程中的所有细节均记在心中,程序员将会发现,Visual Basic 的确节省时间。

2.2 Windows 的动态链接库

Visual Basic 的另外一个优点是它的可扩展性。虽然用 Visual Basic for Windows 开发的应用程序执行速度快,但在一些特定的领域中,经过优化的用 C 语言开发的程序执行速度更快。在本书的第二部分,用户可以看到一些例子,其中的 Visual Basic 应用程序调用 Windows 的 KERNEL、USER 和 GDI 动态链接库(DLLS)中用优化的 C 语言开发的 Windows 函数。DLLS 中有许多功能强大的函数,而用 Visual Basic 开发的应用程序可以很容易地调用它们。如果用户有 C 语言编译程序,也可以创建自己的有特殊用途的 DLLS。

如果用户已购买了 Windows，就可以使用 DLLS 提供的功能，因为它已集成在 Windows 操作系统中了。Windows 的软件开发工具(SDK)中详细地介绍了这些函数，并且提供了供 C 语言调用的原型定义。Visual Basic for Windows 专业版中有一个特别的帮助文件，其中包含了用 Visual Basic 形式定义的所有标准 DLLS 中的函数。如果用户曾在 Visual Basic 1 中调用过这些函数就会体会到这些定义具有多大价值、能节省多少时间了。

前面已经提到过，用户可以用 C 语言创建自己的 DLLS，其中的可执行代码是经过优化的，可以提高执行的速度。通过这种方法，用户可以利用 Visual Basic 提供的功能快速进行用户界面开发，同时结合使用经过高度优化的 C 代码。这也是 Visual Basic 的优点之一。有关这种技术的例子可在本书的第十七章中找到。通常需要一个 C 语言编译程序才能创建自己的 DLLS，但本身提供的 DLL 范例已包含在随书的软盘中。

2.3 其它的 Windows 功能

Visual Basic for Windows 3 为基于 Windows 的应用程序，它提供便于使用的动态数据交换(DDE)、对象链接和嵌入(OLE)机制。我们将会越来越多地应用 DDE 和 OLE，因为基于 Windows 的应用程序会更多地使用这些强大的功能。第十三章和第十四章提供的例子演示了如何在各自独立的 Windows 应用程序之间进行数据和对象的传递和链接。

2.4 未来展望

现在用 Visual Basic 开发应用程序将使用相当长一段时间。Windows 操作系统设计得很稳定，考虑到了将来计算机硬件功能的变化。例如，计算机硬件的图像分辨率和颜色功能会增强，但基于 Windows 的应用程序并不需要作任何修改，它们只会运行得更快、更好。

第三章　事件驱动的程序设计方法

Visual Basic for Windows 程序是由事件驱动的，这个概念是现代程序设计语言的核心。程序员用过这类编程工具后，就不会再使用原来的编程方法。这种编程工具提供了一套崭新的、富有创造性和高效的技术。

幸运的是，事件驱动的代码与过程式代码在概念上的差异并没有人们开始想象的那么大。Visual Basic for Windows 程序的变化并不大。下面准备了一个小程序的三种版本，体现了从 GW-BASIC 到 QuickBasic，最后到 Visual Basic for Windows 的演化过程。Visual Basic for Windows 版本的扩充部分不多，但已经是一个事件驱动的程序了。

这个程序的三种版本在屏幕的顶端(Visual Basic for Windows 版本在窗口的顶端)显示 Testing……1 2 3。先看看 GW-BASIC 版本(注意其中有老式的行号)：

```
10 REM  GW- BASIC Test Program
20 CLS
30 PRINT "Testing...";
40 FOR I=1 TO 3
50 PRINT I;
60 NEXT I
70 PRINT
80 END
```

这段程序的功能是清除屏幕，在显示器的顶端显示 Testing...，然后进入 FOR 循环，显示 1,2,3。

下面的 QuickBasic 程序显示同样的结果，从这个版本可以看出 QuickBasic 对解释型 Basic 的几个改进之处。

```
'QuickBasic Test Program
CLS
PRINT "Testing...";
FOR i= 1 TO 3
    Print i;
Next i
PRINT
END
```

现在，就轮到我们所要讨论的第一个全部由 Visual Basic for Windows 书写的、事件驱动的应用程序：

```
'Visual Basic for Windows Test Program
Sub Form_Click( )
    Print "Testing..."
```

```
    For i=1 To 3
        Print i;
    Next i
End Sub
```

注意 Visual Basic 代码与 QuickBasic 代码是多么类似。关键字，例如 Print，第一个字母大写，其余字母小写，但语法不变。这种版本与老版本之间的主要差别是 Visual Basic 中已把代码放入一个名为 Form_Click 的事件驱动的子程序中。

注意：这个例子在 Visual Basic 1.2 或 3 中显示的内容是一样的，但在版本 2 和 3 中源代码是彩色显示的，所以操作起来更简单些。注释行是绿色的，关键字是蓝色的，而变量是黑色的。这只是 Microsoft 在对 Visual Basic 所作的两次重大修改中添加的许多漂亮的改进之一。

代码为什么不放在主程序中呢？这个问题的答案非常重要，它体现了 Visual Basic for Windows 编程思想的核心。的确，如果想保持与 QuickBasic 的兼容性，代码可以放在主模块中。但是，为了利用新的窗体、事件驱动代码、以及该语言的其它高级功能，包括与 Visual Basic for Windows 的兼容性，Visual Basic for Windows 程序的所有可执行代码必须放在子程序或函数中。在模块一级不应该有可执行的语句。这些子程序和函数由事件（如鼠标器的单击，按下命令按钮或选择正文框中的表项）驱动，或被其它过程调用（可以追加到一些事件驱动的例程中）。因为 Visual Basic 处理了所有日常琐事，应用程序所要做的就是对所发生的事件作出响应。

记住：Visual Basic 的所有可执行代码都必须放在一个子程序或函数块中！

事件驱动的子程序由 Visual Basic for MS-DOS 提供名字。在上面的测试程序中，Form_Click 表示，当用户在窗体窗口中单击鼠标器时，该子程序就运行。整个过程相当简单：运行程序时，先显示一个空的窗口，在窗口中的任一处单击鼠标器后，Testing……1 2 3 显示在窗口中。

在这个例子中，Click（单击）事件激活了一个显示正文的子程序。同样，代码也可以放在 Form_DblClick 子程序中。当用户在窗体窗口中任一处双单击鼠标器按钮后，Testing……1 2 3 就会出现。代码还可以放在 Form_Paint 子程序中，这些信息在程序运行后立即就会出现。

不难看出，在 Visual Basic for MS-DOS 中，还有许多事件可以用来激活子程序。从本书中的示范程序中还可以看出，事件驱动的程序设计方法是极其灵活的，非常适合于开发功能强，响应速度快的应用程序。

第四章 第一个 Visual Basic for Windows 应用程序

第一次使用 Visual Basic for Windows 时，人们总是想，“噢，从何处入手呢？”本章将介绍一个简单而完整的应用程序的开发步骤，使用户熟悉程序的建造方法。

4.1 WINDCHIL 应用程序

WINDCHIL 程序显示给定风速和气温时的 Windchill 指标。Windchill 指标表示皮肤散热的快慢程度。风速增大时散热就快，Windchill 指标则减小；风速为零时，则根据气温导致的等价散热速率来计算这个指标。

单窗体应用程序的开发步骤是：

1. 启动一个新的项目。
2. 设计窗体：
   - 改变窗体窗口的大小、形状和位置。
   - 建立所需大小、形状和位置的控制元件（简称“控件”）。
   - 改变窗体和控件的其它属性（如果需要）。
3. 添加子程序，用于处理所需的事件。
4. 测试应用程序，必要时，可重复上面的步骤。
5. 存储项目文件。
6. （可选）建立可执行（EXE）文件。

这个程序并不大，但它以简单、明了的方式说明了单个窗体的若干控件。例如，用户可以用一对单选钮来选择温度值的单位类型，即华氏度或摄氏度。用户还可以用滚动条方便地选择风速。

我们将按上面给出的步骤来建造 WINDCHIL 程序。熟悉 Visual Basic for MS-DOS 环境后就会发现，在开发单窗体应用程序时，均可参照上面的步骤。

4.1.1 启动一个新的项目

现在，开始在 Visual Basic for Windows 中建立 WINDCHIL 程序。为了从初始状态开始，先在 File 菜单中选择 New Project。图 4.1 给出了初始屏幕布局（可能在用户的显示器中，有些 Visual Basic 的窗口在其它位置，但可以随意把任何窗口移到任意位置）。

4.1.2 改变窗体标题的属性

本项目只包含一个窗体。窗体是一个应用程序窗口，它具有相关的代码和控件，以便完成一个或多个指定的任务。新窗体窗口的缺省标题是 Form1。要改变标题，可在新窗体的

Properties 窗体中，选取 Caption 属性，然后键入 Windchill Index。(如果看不到 Properties 窗口，可在窗体上单击鼠标，然后在 Window 菜单中选取 Properties 或按 F4。)，这时标题也随之改变了。图 4.2 表示改动后的窗体窗口。

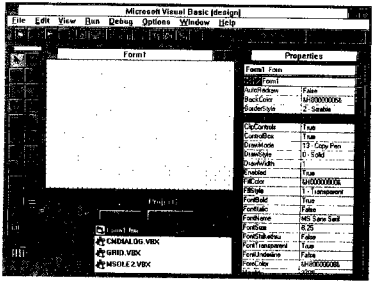

图 4.1 启动 Visual Basic for windows 3 后 屏幕上的内容

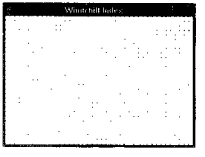

图 4.2 缺省标题 Form1 改成 Windchill Index 后新窗体的窗口

缺省窗体相对于将要放在它上面的控件来说有点小。拖动窗体的边框就可以改变它的大小(拖动窗体的角则可以同时改变窗体的高度和宽度)。拖动标题条则可以移动窗体。移动和放缩窗体，使之如图 4.3 所示。不用担心操作不当，因为以后也很容易改变窗体的形状。

4.1.2.1 为窗体添加菜单

建立窗体的菜单时，在 Window 菜单中选择 Menu Design。Menu Design Window 对话框出现，如图 4.3 所示，Menu Design Window 对话框是为 Visual Basic 程序设计和修改菜单的接口。单击 Caption 正文框并键入&Help。这是 WINDCHIL 窗体唯一的菜单标牌。"&"表示用户按 ALT-H 就可以打开菜单。按 Tab 移到 Name 正文框并键入每个菜单标牌和菜单项都有一个控件名。在窗体的代码部分中，子程序名就是根据控件名来命名的。用户选择菜单标牌或菜单项后就激活相应的子程序。笔者习惯于在这些菜单控件名前面加上 men，以便迅速找到与菜

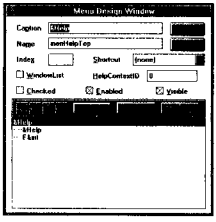

图 4.3　键入与菜单有关的信息后 Menu Design Windows 的布局

单相关连的代码。

为 Help 菜单增加菜单项时，可单击 Next 按钮，使 Caption 名称缩格时，单击对话框中部的向右缩格按钮，高亮条中将出现两个圆点。表项缩格一次表示它以后将成为一个菜单项，而缩格两次则表示它将成为下拉菜单的级联菜单。

单击缩格按纽后，在 Caption 正文框中输入 &Help，在 Name 正文框中输入 menHelp，作为第一个菜单项。对于第二个菜单项，在 Caption 正文框中输入 E&ixt，在 Name 正文框中输入 menExit。注意，第二个菜单项保持前一项的缩格次数，无需再次敲击向右缩格按钮。下表说明了这种菜单标题和菜单项在本书中的排版格式。

Caption	Name	Indentation
&Help	menTop	0
&Help	menHelp	1
E&xit	menExit	1

WINDCHIL 的菜单就做完了。单击 OK，退出 Menu Design Window 对话框。Help 菜单即出现在窗体的菜单条上。单击窗体新的 Help 菜单就可以看见 Help 菜单项，如图 4.4 所示。

4.1.2.2　*增加窗体的控件*

下一步是增加和修改窗体的控件。这些控件可从 Toolbox（工具箱）中选出，如图 4.5 所示。控件包括按钮、标签、正文框以及在窗体上构成用户界面的其它构件。选择哪些控件以及怎样在窗体窗口中排放它们主要取决于窗体的功能。控件可以用来显示正文，提供用户可修改的正文输入字段、定义矩形的图片输出区域、显示列表清单，以及提供切换选项等。每种控件的详细介绍在 Visual Basic for Windows 手册中可以找到。

图 4.6 是放好控件后的 WINDCHIL 窗体。下面将介绍在窗体上放置控件的过程。

框架　在工具箱中单击 Frame 可选择框架。单击窗体的左上角（Help 菜单之下），把鼠标拖到窗体中部附近，这样就画好了第一个框架。在完成后的窗体上，这将成为窗体左边的 Wind

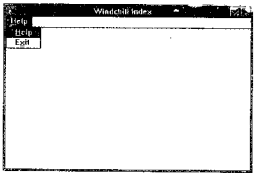

图 4.4 打开 Help 菜单后的 WINDCHIL 窗体

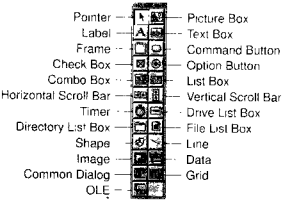

图 4.5 工具框

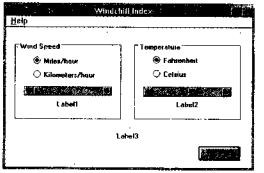

图 4.6 带控件的 WINDCHIL 窗体

Speed(风速)框。以后可以随时放缩和移动这个框架,使它在窗体上的位置完全正确,所以在刚开始时不必担心是否精确。在屏幕顶端的 Properties 条上,从 Property 列表框中选择 Caption(如果还未选择 Caption),然后在 Value 列表框中键入 Wind Speed。框架的标题就变为 Wind Speed。用同样的步骤建立第二个框架,这个新框架放在第一个框架的右边(使用相同的高度和

宽度),并且在 Caption 属性的 Value 列表框中键入 Temperature,作为新框架的标题。

单选钮　单击工具箱中的 Option Btn,在窗体上分别画好四个单选钮。在 Wind Speed 框中用拖动的方法画好前面两个按钮,然后在 Temperature 框中画好另外两个按钮(单选钮的建立次序是很重要的)。建立一个新的单选钮之后,应再次单击 Option Btn。参照图 4.6 设置每个单选钮的 Caption 属性。

在框架内对单选钮分组时,Visual Basic Windows 要求在任何时候,在每一框架内只能选择一个选项按钮。第六章的 FRAMOPT 程序说明了按框架分组的选项按钮是怎样工作的。单击 Miles/hour 和 Fahrenheit 单选钮,便它们在程序运行后处于活动状态。

滚动条　现在,用同样的方法添加两个滚动条。在工具箱中单击 HScrollBar,然后用拖动的方法在每个框架中画好滚动条。首先在 WindSpeed 框中画滚动条,其位置可参照图 4.6。现在修改这些滚动条的五个属性。单击 Wind Speed 框中的滚动条。从 Property 列表框中选择 Large Change,并且在 Value 列表框中输入 10。用同样的方式改变 SallCharge、Max、Min 和 Value 属性,SmallChange 缺省值为 1,具体的数值在下表给出。在本书中,窗体和控件属性均以这种格式列出。

Property	Value
HScroll1:	
LargeChange	10
Max	50
Min	5
Name	Hscroll1
Value	5
HScroll2:	
LargeChange	10
Max	90
Min	-50
Name	Hscroll2
Value	32

标签　在工具箱中单击 Label,并在框架中画好两个标签(画完一个标签后,记住重新在工具箱中单击 Label)。另外在框架的下部画好第三个标签,如图 4.6 所示。把这三个标签的 Alignment 属性都改为 2-Center。

命令按钮　最后添加的控件是命令按钮。单击工具箱中的命令按钮,并在窗体的右下角放置该按钮,见图 4.6。把按钮的 Caption 属性改为 Cancel,把 Cancel 和 Default 属性改为 True。当用户按下 Esc 键时,若 Cancel 属性为 True,Visual Basic for MS DOS 就选择该命令按钮。若 Default 属性为 True,该按钮在窗体被装入之后就成为活动的命令按钮(用户按下 Enter 键时就表示选择活动的命令按钮)。

4.1.3 添加子程序代码

构造该应用程序时，最后一项任务是增加一些事件驱动的代码，用于计算和显示 Windchill 指标。在窗体的窗口中的任意位置(除控件外)双击鼠标，Visual Basic 开发环境即认为用户要编辑窗体中某个事件的子程序(如果在某个控件上双击鼠标，Visual Basic 会跳至与该控件相联的子程序中执行)。Visual Basic 将在代码编辑窗口中建立下面两行代码：

```
SUB Form_Load()
    END SUB
```

子程序所需的代码行将键入到这两行代码之间。

Visual Basic 为每一个窗体、菜单项和控件中的每一个事件创建一个子程序，并给子程序命名。其方法是通过 Name 的属性进行控制。所有事件驱动子程序的名字语法相同：Object_Event。其中的 Object 部分是与 Event 部分限定的事件有关的窗体、菜单项和控件的名字。下划线把事件驱动子程序的名字分成两个部分。在本例中，子程序在 Visual Basic 装入窗体时才激活，因此命名为 From_Load。

用户并不需要给 From_Load 或其它子程序添加代码。举个例子，如果需要程序对双击鼠标按钮而不是单击作出反应，就只需在 From_Click 子程序中添加代码，而 From_Click 子程序仍然为空。

下面开始在 From_Laod 子程序中添加代码。把下列三行程序添加在 From_Laod 子程序中。

注意：随书的软盘中包含了 WINDCHIL 应用程序。如果用户不想键入这段程序，可以打开 Open 菜单，选择 Open Project 命令，然后键入 C:\WORKSHOP\WINDCHIL.MAK。不过建议用户还是自己键入这段程序，这样就可以经历开发应用程序的每一步骤。

```
Sub Form_Load()
Force scrokkll bars to update
HScroll1_Change
HScroll2_Chage
End Sub
```

这些代码所作的全部工作就是调用另外两个事件子程序，即滚动条的 Change 子程序。这样保证了程序开始运行时的显示是正确的。

如果用户了解 Microsoft QuickBasic 环境的编辑功能，就能很快地适应 Visual Basic 编辑器，因为许多命令是一样的。图 4.7 表示键入 Form_Load 代码后，代码编辑窗口中的情形。

现在建立 Command_Click 子程序。它只需一行代码。在 Edit 菜单中选择 Event Procedure 命令，在 Object 列表框中选择 Command，在 Events 列表框中选择 Click，然后单击 Edit In Active 按钮。

```
SUB Command1_Click
    'Cancel button clicked
    Unload Windchil
END SUB
```

图 4.7　Form_Load 子程序

除了与控件事件有关的子程序外，Visual Basic 还可以创建其它名字的子程序，这些子程序可从运行的程序中调用，但不能直接被事件激活。例如，可以在 HScroll_Change 子程序中调用 ChillOut 子程序，修改显示的 WindChill 数据。用户在创建该子程序之前，不能在 Objects 和 Proc 组合方框中调用它。要创建一个像 ChillOut 这样的全新的子程序，可以在 View 菜单中选择 New Procedure。在弹出的对话框中键入新过程的名，Visual Basic 就会创建这个新过程。当然，用户还必须逐行添加代码。现在用户可以试着添加下面几行代码创建 ChillOut 子程序。

```
Sub Chillout()
    'Get working values form scroll bars
    Wind = HScroll1.value
    Temp = HScroll2.value

    'Convert to MPH if KPH selected
    If Option2.Value = True Then
        Wind = Mph(Wind)
    End If
    'Convert to Fahrenheit if Celsius selected
    If Option4.Value = True Then
    Temp = Fahrenheit (Temp)
    End If

    'Calculate windchill index
    x = .303439 * Sqr(Wind) - .0202886 * Wind
    Chill = Int(91.9 - (91.4 - Temp) * (x + .474266))

    'Convert back to Celsius if selected
     If Option4.Value = True Then
        Chill = Celsius(Chill)
    End If

    'Display windchill index
    Y$ = "Windchill Index is " + Str$(CInt(Chill))
```

```
    If Option3.Value = True Then
        Label3.Caption = Y$ + "F"
    Else
        Label3.Caption = Y$ + "C"
    End If
End Sub
```

在任何时候都可以直接进入某个已存在的子程序或函数。在 View 菜单中选择 Code 命令，列出所有可用的子程序和函数。从中选择一个函数、子程序或窗体，然后单击 Edit In Active，就可以编辑该子程序、函数或窗体的代码。

在本书中，每个窗体的源代码清单均按下面的格式给出，其中包括窗体的所有源代码、注释以及其它在窗体一级上出现的正文行。我们已经输入了 Form_Load、Command_Click 子程序以及 FAHRENHEIT 函数的代码。输入程序的其余代码之后，WINDCHIL 程序就开发完毕了。

WINDCHILL 的程序源代码：

```
'WINDCHIL
'Calculates windchill index

Function Celsius(F)
    'Convert Fahrenheit to Celsius
    Celsius = (F+40) * 5/9 - 40
End Function

Sub Chill0ut()
    'Get working values from scroll bars
    Wind = HScroll1.Value
    Temp = HScroll2.Value
'Convert to MPH if KPH selected
If Option2.Value = True Then
    Wind = MPH(Wind)
End If

'Convert to Fahrenheit if Celsius selected
If Option4.Value = True Then
    Temp = Fahrenheit(Temp)
End If

'Calculate windchill index
X=.303139 * Sqr(Wind)-.0202886 * Wind
Chill = Int(91.9 - (91.4-Temp) * (X + .474266))

'Convert back to Celsius if selected
```

```
    If Option4.Value = True Then
        Chill = Celsius(Chill)
    End If

    'Display windchill index
    Y$ = "windchill Index is " + Str$(CInt(Chill))
    If Option3.Value = True Then
        Label3.Caption = Y$ + "F"
    Else
        Label3.Caption = Y$ + "C"
    End If
End Sub

    Sub Command1_Click()
        'Cancel button clicked
        Unload Windchil
    End Sub

    Function Fahrenheit(C)
        'Convert Celsius to Fahrenheit
        Fahrenheit = (C + 40) * 9/5 - 40
    End Function

    Sub Form_Load()
        'Force scroll bars to update
        HScroll1_Change
        HScroll2_Change
    End Sub

    Sub HScroll1_Change()
        'Get wind speed
        Tmp = HScroll1.Value

    'Display using selected units
    If Option2.Value = True Then
        Label1.Caption = Tmp + "KPH"
    Else
        Label1.Caption = Tmp + "MPH"
    End If

        'Calculate windchill index
        ChillOut
    End Sub
```