

第1学时 Visual Basic for Application是什么

欢迎使用Excel和Visual Basic for Application开发应用程序！作为已经能够熟练使用 Excel 的用户，你一定对 Excel 强大的功能感到满意。现在，你将逐步提高到更高的水平。即使你以前从来没有编写过程序，也能够使用 Visual Basic for Application 开发出解决方案。Visual Basic for Application 是现在可用的最容易学习、最容易使用同时也是最复杂的应用程序自动化语言（过去常常称为宏语言）之一。在这个学时中，通过熟悉宏录制器，将开始学习有关的基础知识。

本学时的重点包括：

- Visual Basic for Application是什么
- Excel环境中基于应用程序的自动化的优点
- 录制一个简单的宏
- 执行宏
- 编辑宏
- 宏录制器的局限性

1.1 Visual Basic for Application是什么

直到90年代早期，使应用程序自动化还是充满挑战性的领域。对每个需要自动化的应用程序，人们不得不学习一种不同的自动化语言。例如，可以使用Excel的宏语言使Excel自动化，使用Word Basic使Microsoft Word自动化，等等。Microsoft决定让它开发出来的应用程序共享一种通用的自动化语言——Visual Basic for Application (VBA)，而不是使用不同的自动化语言。可以认为Visual Basic for Application是非常流行的应用程序开发语言——Visual Basic的子集。实际上，VBA是“寄生于”Visual Basic应用程序的版本。VBA与Visual Basic的区别包括如下几个方面：

- Visual Basic是设计用于创建标准的应用程序，而 VBA是用于使已有的应用程序自动化。
- Visual Basic 具有自己的开发环境，而 VBA必须“寄生于”已有的应用程序。
- 要运行Visual Basic开发的应用程序，用户不用在他的系统上访问 Visual Basic，因为 Visual Basic开发出的应用程序是可执行的。而由于 VBA应用程序是寄生性的，执行它们要求用户访问“父”应用程序，例如 Excel。

尽管存在这些不同，Visual Basic和VBA在结构上仍然非常相似。事实上，如果你已经了解了Visual Basic，会发现学习VBA非常快。相应地，学完VBA会给Visual Basic的学习打下坚实的基础。而且，当学会在Excel中用VBA创建解决方案后，你就已经具备了在Word、Project、Access、Outlook、FoxPro和PowerPoint中用VBA创建解决方案的大部分知识。

VBA的一个关键性特征是从一种 Microsoft产品或者 Visual Basic中学到的知识可以相互转化。

VBA究竟是什么？更确切地讲，它是一种自动化语言，可以用它使常用的过程或者进程自动化，可以创建自定义的解决方案，此外，如果你愿意，还可以将 Excel用做开发平台实现应用程序。

1.2 Excel环境中基于应用程序的自动化的优点

你也许希望知道可以用 VBA干什么。使用VBA可以实现的功能包括：

- 使重复性的任务自动化。
- 自定义Excel中工具栏、菜单和窗体的界面。
- 简化模板的使用。
- 为Excel环境添加额外的功能。
- 创建报表。
- 对数据执行复杂的操作和分析。

你以前也许没有想到将应用程序用作开发平台。大多数人考虑开发应用程序时，想到的都是像Visual Basic或者C++这样的语言。你希望采用 Excel作为开发平台有许多原因，这些原因包括：

- Excel的应用程序功能强大，包括打印、文件处理、格式化和文本编辑。
- Excel具有大量可供选择的内置函数。
- Excel提供熟悉的界面。
- 可连接到多种格式的数据库。

如果以前曾经用某种语言编写过程序，你就会知道，一半的工作不得不用来完成一些基本的功能，包括文件的打开和保存，以及剪贴板操作，例如拷贝和粘贴，等等。而这带来了使用应用程序开发解决方案的一个主要的优点——寄主应用程序已经具备了各种基本功能。需要做的只是使用它，必须使用 Excel中包括文件处理、文本编辑和格式化在内的各种功能。

因为是在Excel中开发解决方案，所以也必须访问Excel的扩展函数库。作为Excel用户时所熟悉的所有函数（包括SUM、IRR、MAX、FV、PMT和AVG），在作为Excel开发者时都是可用的。

从解决方案的最终用户的角度看，他们是在已经知道如何使用的应用程序上进行工作，因此他们可从中受益。他们对Excel的菜单系统、工具栏和工作表区域都很熟悉。正因为如此，他们会立刻对你的自动化解决方案感到满意。

用Excel开发解决方案的其他一些优点不是十分明显，例如 Excel连接数据库的特征。如果在解决方案的窗体中需要对数据库（例如 Microsoft SQL Server或者Microsoft Access）进行操作，由于Excel可以很容易地做到，所以你也可以很容易地做到。

1.3 录制简单的宏

在介绍学习VBA代码之前，应该花几分钟录制一个宏。Excel的宏录制器允许记录一系列的操作，并且将这些操作转换为 VBA代码。即使当你对编写 VBA代码已经完全熟练时，也会在工作时使用宏录制器。作为 VBA开发者，使用宏录制器有两个原因。一个原因是因为使用

宏录制器可以节省时间，开发者通常用它来建立应用程序的基础。另一个原因是宏录制器可以用作教学工具，如果你不能确定如何编写一系列的步骤，可以进行录制，再查看代码。

新术语 宏指一系列以Excel能够执行的名字保存的命令。

以下将要录制的宏非常简单，只是改变单元格的字体和颜色。虽然有其他方法可以实现这种类型的任务（例如风格、自动套用格式等等），但是这一系列步骤为宏录制器提供了很好的示例。请完成如下步骤：

- 1) 打开新的工作簿，确认所有其他工作簿已经关闭（如果它们包含宏或者其他VBA代码），以便能够很容易地对录制的宏进行定位和处理。
- 2) 在单元格A1中输入你的名字，在单元格B1中输入你的姓，在单元格C1中输入你居住的城市名，在单元格D1中输入你居住的国家名。以上操作为下面的练习提供用来处理的数据。
- 3) 选中单元格A1。
- 4) 选择“工具”、“宏”、“录制新宏”，显示如图1-1所示的“录制新宏”对话框。

图1-1 “录制新宏”对话框

允许你为将要录制的宏取名



- 5) 输入“BigFont”作为宏名，键入回车开始录制宏。注意此时Excel应用程序窗口的状态栏中显示“录制”，特别是“停止录制”工具栏也显示出来。

宏的名字最多可为255个字符，并且必须以字母开始。宏名中可用的其他字符包括字母、数字和下划线。宏名中不允许出现空格。通常用下划线代表空格。

要显示“停止录制”对话框，可用鼠标右键单击工具栏，并选择“停止录制”。只有在录制宏时，“停止录制”工具栏才可用。

- 6) 选择“格式”，“单元格”，会显示“单元格格式”对话框，选择“字体”选项卡。
- 7) 将字体大小设为16，将字体颜色设为红色，单击“确定”按钮。
- 8) 单击“停止录制”工具栏按钮，结束宏录制过程。

如果“停止录制”工具栏没有显示出来，请选择“工具”、“宏”、“停止录制”。

录制完一个宏后，就可以执行它了。

1.4 执行宏

当执行一个宏时，它按照录制宏时相同的步骤进行操作。要执行一个宏，可按照如下步骤：

- 1) 选择单元格B1。
- 2) 选择“工具”、“宏”、“宏”，显示“宏”对话框，如图1-2所示。

按下Alt+F8键也可以显示“宏”对话框。

图1-2 “宏”对话框用来
选择将要运行或编
辑的宏



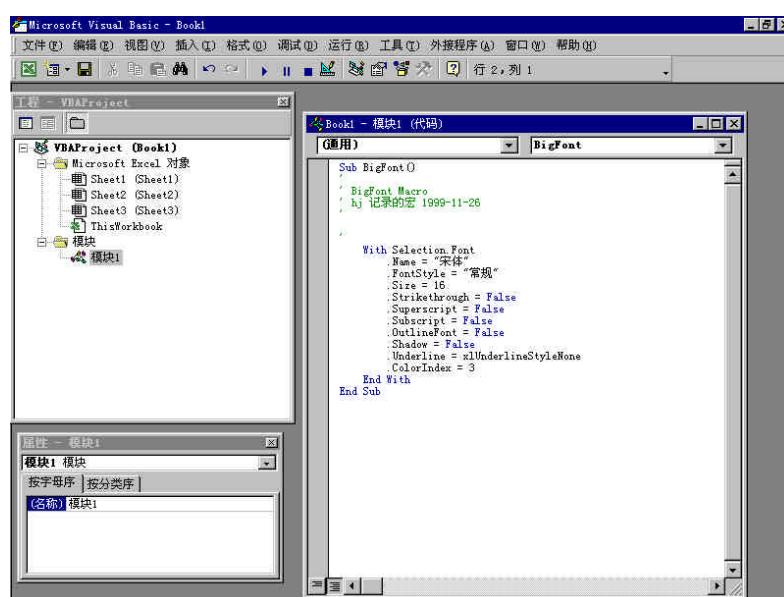
- 3) 选择“BigFont”，选择“执行”，则单元格B1中的字体变为16磅和红色。
- 4) 选择单元格C1和D1，再次运行宏“BigFont”。尽管最初录制宏时只改变了一个单元格的字体，此时两个单元格的字体都变为16磅和红色。

1.5 查看录制的代码

当执行希望保存到宏中的步骤时，Excel将操作步骤转化为VBA代码。要查看生成的代码，可按照如下步骤：

- 1) 选择“工具”、“宏”、“宏”，显示“宏”对话框。
- 2) 选择“BigFont”，单击“编辑”，此时会打开Microsoft Visual Basic编辑器窗口，如图1-3所示。

图1-3 Microsoft Visual Basic编辑器用来查
看和编辑VBA代码



Microsoft Visual Basic编辑器包括几部分组件。在第4学时中，将学习到更多有关Visual Basic组件的知识。现在应该将注意力集中到显示的代码上。显示的代码应该和程序清单1-1相似。

程序清单1-1 BigFont过程

```
Sub BigFont()
    ' BigFont Macro
    ' hj 记录的宏 1999-11-26
    ' 

    With Selection.Font
        .Name = "宋体"
        .FontStyle = "常规"
        .Size = 16
        .Strikethrough = False
        .Superscript = False
        .Subscript = False
        .OutlineFont = False
        .Shadow = False
        .Underline = xlUnderlineStyleNone
        .ColorIndex = 3
    End With
End Sub
```

代码的第一行“Sub BigFont()”表示了宏的起点和名字，接下来的以单引号开头的行为注释行，表示与宏有关的说明，在本例中包括宏的名字、录制时间、录制人。

宏实际工作的部分是从“with”开始的。注意单词“Selection”，Selection在VBA中用来表示突出显示的部分，这就是不论选择一个或者多个单元格宏都能正常工作的原因。你可能注意到的另一点是录制的内容比执行的操作要多得多。你仅仅改变了字体的大小和颜色，但是所有的字体信息都从“单元格格式”对话框的“字体”选项卡中录制下来。

1.6 编辑录制的代码

编辑代码可在Visual Basic编辑器中直接进行。可添加代码行、删除行或者修改行。要做的第一件事是删除录制下来的多余的行，可按照如下步骤：

- 1) 突出显示以“.Name”开始的行。
- 2) 删除该行。不用担心产生的空行，VBA忽略空行。
- 3) 继续删除多余的行，直至过程和下面的代码相同：

```
Sub BigFont()
    ' 
    ' BigFont Macro
    ' hj 记录的宏 1999-11-26
    ' 

    With Selection.Font
        .Size = 16
```

```
.ColorIndex = 3  
End With  
End Sub
```

- 4) 关闭Visual Basic编辑器窗口，返回工作簿。
- 5) 在单元格E1中输入“test”。
- 6) 选中单元格E1，运行宏“BigFont”。注意宏运行的结果与删除宏中的多余的代码行之前完全相同。
- 7) 选择“工具”、“宏”、“宏”。
- 8) 选择“BigFont”，单击“编辑”按钮。
- 9) 现在，当运行这个宏时，字体大小设置为16。编辑这个宏将字体大小设置为24。完成后的宏应该和如下代码相似。

```
Sub BigFont()  
'  
' BigFont Macro  
' hj 记录的宏 1999-11-26  
'
```

```
With Selection.Font  
    .Size = 24  
    .ColorIndex = 3  
End With  
End Sub
```

- 10) 关闭Visual Basic编辑器窗口。
- 11) 选中单元格A1，运行“BigFont”宏。现在该单元格的字体会变得更大。
- 12) 将工作簿保存为“Hour 1”。

现在可以看到编辑录制下来的宏非常简单。需要对宏进行编辑是因为以下两个原因。其一是在录制宏时出错；另一个原因是希望对宏的功能进行改变。不论是因为哪个原因，总是可用Visual Basic编辑器对宏进行编辑。

1.7 宏录制器的局限性

希望自动化的许多Excel的操作过程都可以通过对操作进行录制来完成。但是宏录制器确实具有一定的局限性。通过宏录制器不能完成的工作包括：

- 当宏运行时向用户提示信息。
- 根据用户的输入或者单元格的数值执行不同的操作。
- 相似Excel的对话框，例如“另存为”对话框。
- 显示和使用自定义的用户窗体。

这些局限性仅仅是需要编写你自己的VBA代码的原因的一部分。在下一学时中，将更多地使用宏录制器。

1.8 学时小结

通过本学时，你可以很快地学会使用宏录制器。可学到如何录制、执行和编辑宏，甚至

可以修改部分VBA代码。现在你已经为更详细地学习宏环境打下了基础。

下一学时的重点是有关宏的高级主题。理解宏“怎么样”和“为什么”是使用VBA开发应用程序的第一步。

1.9 专家答疑

问题：用VBA编程为什么需要了解宏录制器？

解答：了解宏录制器有两个主要的好处：你掌握的有关宏录制器的知识可以成为学习VBA的基础；此外，当开始开发应用程序时，你会发现，可以先录制尽可能多的宏，然后再对录制下来的代码进行修改。

问题：如果我从未编写过程序，学习VBA是否有困难？

解答：没有！开始用VBA进行编程只需了解Excel。在开发解决方案时将用到在Excel方面的知识。本书将提供你需要的所有其他知识。

1.10 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容。答案请参考附录。

1.10.1 思考题

- 1) VBA只能用于Excel吗？
- 2) VBA是基于哪种语言？
- 3) 判断题：在VBA应用程序中可以使用Excel的内置函数。
- 4) 对宏代码进行编辑时，工作在_____。
- 5) 列举宏录制器的两个局限性。

1.10.2 练习题

创建一个新的名为“Title”的宏，在单元格A1中输入你的名字，在B1中输入今天的日期。宏的功能是将A1和B1的字体大小设置为14磅。录制完成宏后，复习生成的代码。

第2学时 处理录制的宏

在第1学时中，重点讲述了录制宏的基础知识。本学时则着重讲述有关宏的高级主题。在本学时中学到的知识将有助于编写VBA代码。当开始编写VBA过程时，你将用本学时中学到的技术来决定将代码保存到何处，以及如何使用快捷键和按钮来执行过程。

本学时的重点包括：

- 为宏指定快捷键
- 属于宏的位置的作用
- 创建个人宏工作簿
- 为宏指定按钮或者图形

2.1 指定快捷键

新术语 你也许希望为经常使用的宏指定快捷键。快捷键是指键的组合，当按下时执行一条命令。你也许对拷贝命令的快捷键Ctrl+C很熟悉，Excel允许为宏指定快捷键。快捷键必须是Ctrl键和选择的某个字母键的组合。当为宏指定了快捷键后，就可以用快捷键来执行宏，而不用通过“工具”菜单。

当包含宏的工作簿打开时，为宏指定的快捷键会覆盖Excel默认的快捷键。这意味着，如果将Ctrl+C指定给某个宏，那么Ctrl+C不再执行拷贝命令，这可能会让你或者你的用户感到迷惑。在为宏指定快捷键之前，最好打印一张Excel快捷键的程序清单作为参考。要打印Excel快捷键的程序清单可按照如下步骤：

- 1) 显示Excel在线帮助系统的“目录”选项卡。
- 2) 从“使用快捷键”文件夹中选择“快捷键”标题。
- 3) 用鼠标右键单击“快捷键”，从菜单中选择“打印”。
- 4) 选择“打印所选标题和所有子主题”，单击“确定”按钮。

在创建宏时可以指定快捷键，也可以创建完成后再指定。要在创建宏时指定快捷键，可在输入宏的名字后，在“快捷键”文本框中输入相应的键。录制完一个宏后，也可返回为宏指定快捷键。例如，要为BigFont宏指定快捷键，可按照如下步骤：

- 1) 如果没有打开“Hour1”工作簿，则将它打开。

当打开这个工作簿时，可能看到一个消息框，这取决于宏的安全等级。
如果出现这个消息框，请选择“启用宏”。

- 2) 选择“工具”、“宏”、“宏”，显示“宏”对话框。
- 3) 选择“BigFont”，再单击“选项”按钮，显示“宏选项”对话框如图2-1所示。

图2-1 “宏选项”对话框
用来为宏指定快捷键



4) 输入字母“ b ”作为快捷键。

快捷键必须是Ctrl和一个字母键的组合。

5) 单击“确定”按钮，关闭“宏”对话框。

6) 在单元格A3中输入“ Test ”。

7) 选中单元格A3，在按下Ctrl+B时会执行相应的宏，可以看到该单元格字体的大小和颜色都改变了。

如果对Excel的快捷键比较熟悉，你应该知道，通常情况下，Ctrl+B会将选中的内容以粗体显示。因为你将Ctrl+B指定为执行宏的快捷键，所以就覆盖了Excel的内置函数。只要该工作簿打开，这种情况就会一直保持。注意Excel不会试图阻止这样做。你甚至看不到警告信息，Excel假设你知道这样做的后果。

2.2 决定宏的保存位置

当创建第一个宏时，你接收了将该宏存放在默认的位置。总的来说，可将宏存放在三种可能的位置：

- 本工作簿。
- 新的工作簿。
- 个人宏工作簿。

如果选择将宏保存在本工作簿中，则宏驻留在当前的工作簿中。这意味着只有当该工作簿打开时，该宏才可用。也可选择将宏保存在新的工作簿中，如果这样选择，会自动创建新的工作簿。最后的选择是将宏保存在个人宏工作簿中。

2.3 创建个人宏工作簿

新术语 个人宏工作簿是为宏保留的一种特别的隐藏工作簿。第一次选择将宏保存到个人宏工作簿时，会创建名为PERSONAL.XLS的新文件。如果这个文件存在，当打开Excel时它会自动打开。因为个人宏工作簿始终打开，所以保存在其中的所有宏都是可用的。这意味着当创建可用于多个工作簿的通用的宏时，最好将它保存在个人宏工作簿中。

在Macintosh上，个人宏工作簿的名字为 PERSONAL MACRO WORK BOOK。

个人宏工作簿保存在 XLSTART文件夹中。

2.3.1 保存宏到个人宏工作簿中

将宏保存到个人宏工作簿中基本上和将宏保存到本工作簿中一样。在本练习中，将创建一个非常简单的宏，将文本变为斜体字并加上下划线。要将宏保存到个人宏工作簿中，可按照如下步骤操作：

- 1) 选择“工具”、“宏”、“录制新宏”，显示“录制新宏”对话框。
- 2) 输入“FormatText”作为宏名。
- 3) 从“保存在”下拉式列表框中选择“个人宏工作簿”。
- 4) 单击“确定”按钮。现在进入录制模式。
- 5) 单击“斜体”工具栏按钮。一段时间内，鼠标可能显示为沙漏形状，特别是当第一次将宏保存到个人宏工作簿时，因为Excel需要创建个人宏工作簿文件。
- 6) 单击下划线工具栏按钮。
- 7) 停止录制宏。

2.3.2 使用个人宏工作簿中的宏

现在，已经将一个宏保存到了个人宏工作簿中，可在任何工作簿中使用它。要证明这一点，可按照如下步骤操作：

- 1) 关闭所有打开的工作簿。通过这种方法，可以知道使用的宏是保存在个人宏工作簿中的。
- 2) 打开一个新的工作簿。
- 3) 在单元格A1中输入你的名字。
- 4) 选中单元格A1。
- 5) 选择“工具”、“宏”、“宏”，显示“宏”对话框。可在 PERSONAL.XLS!FormatText的列表框中看到“FormatText”宏，如图2-2所示。

图2-2 保存到个人宏工作簿
中的宏可用于所有的
工作簿



- 6) 选择“FormatText”宏并单击“执行”按钮，选中的单元格的文本现在变为斜体字并有下划线。

2.3.3 编辑个人宏工作簿中的宏

对个人宏工作簿中的宏进行编辑与编辑其他工作簿中的宏在处理上有细微的不同。个人

宏工作簿是一个隐藏的工作簿，在能编辑它的内容前，必须将它转变为非隐藏的工作簿。要使个人宏工作簿成为非隐藏的工作簿并编辑其中的宏，可按照如下步骤：

- 1) 选择“窗口”、“取消隐藏”，显示“取消隐藏”对话框，如图2-3所示。

图2-3 编辑个人宏工作簿中
的宏的第一步是将该
工作簿取消隐藏



- 2) 选择“Personal”并单击“确定”按钮，个人宏工作簿就取消了隐藏属性，并成为当前活动的工作簿。当前工作簿窗口的标题显示为“Personal”。
- 3) 选择“工具”、“宏”、“宏”，显示“宏”对话框。
- 4) 选择“FormatText”并单击“编辑”按钮。如果没有取消个人宏工作簿的隐藏属性而试图编辑该宏，将看到先要取消该工作簿隐藏属性的信息。
- 5) 对宏进行修改，关闭Microsoft Visual Basic编辑器窗口。

2.3.4 从个人宏工作簿中删除宏

要从个人宏工作簿中删除宏，同样应该将该工作簿取消隐藏属性。因为你可能并不希望“FormatText”宏始终存放在个人宏工作簿中。当该工作簿已经取消隐藏属性后，可按照如下步骤删除该宏：

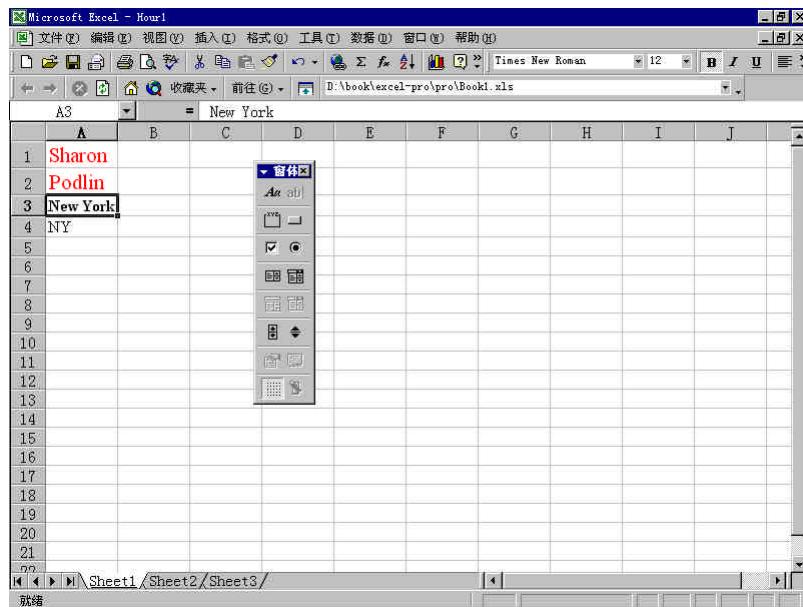
- 1) 选择“工具”、“宏”、“宏”，显示“宏”对话框。
- 2) 选择“FormatText”宏并单击“删除”按钮。
- 3) 将看到询问是否希望删除“FormatText”宏的消息框。单击“确定”，则删除了该宏。
- 4) 保存工作簿。
- 5) 最后需要隐藏个人宏工作簿。选择“窗口”、“隐藏”，则该工作簿成为隐藏工作簿。

2.4 将宏指定给按钮

作为Excel开发者，一个主要目标是为自动化任务提供易于使用的界面。一种最为可视的实现方法是在用户将要使用的工作簿中提供命令按钮。通过使用“窗体”工具栏，可为工作簿中的工作表添加按钮。在创建完按钮后，可为它指定宏，然后你的用户就可以通过单击按钮来执行宏。在本练习中，将创建一个按钮，并为该按钮指定一个宏，然后再用该按钮来执行宏。具体可按照如下步骤：

- 1) 打开“Hour1”工作簿。
- 2) 选择“视图”、“工具栏”、“窗体”，显示如图2-4所示的“窗体”工具栏。
- 3) 单击“窗体”工具栏中的“按钮”，此时鼠标变为交叉线，看上去像个加号。

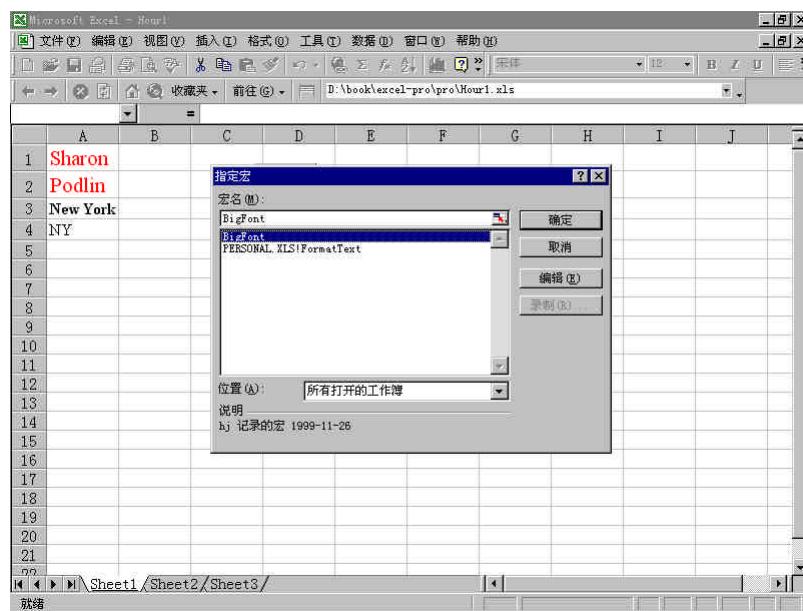
图2-4 “窗体”工具栏中包括多种可用于自动化处理的控件



4) 在希望放置命令按钮的地方按下鼠标左键，拖动鼠标画一个矩形，这个矩形代表了命令按钮的大小和形状。当对命令按钮的大小满意时松开鼠标左键，这样一个命令按钮就添加到了工作表中，显示如图2-5所示的“指定宏”对话框。当添加命令按钮到工作表时，Excel会自动提示为该按钮指定宏。

- 5) 选择“BigFont”，单击“确定”按钮，这个宏就被指定给相应的按钮。
- 6) 在按钮上的字母B前单击鼠标左键，按下Delete键直至删除了所有文本。
- 7) 输入“BigFont”，单击“确定”按钮。
- 8) 在按钮之外单击鼠标左键，使按钮不再被选中。

图2-5 “指定宏”对话框用以选择相应按钮要执行的宏



9) 在单元格 A6 中输入今天的日期，并按回车键。

10) 选中单元格 A6，单击“BigFont”按钮，执行“BigFont”宏。

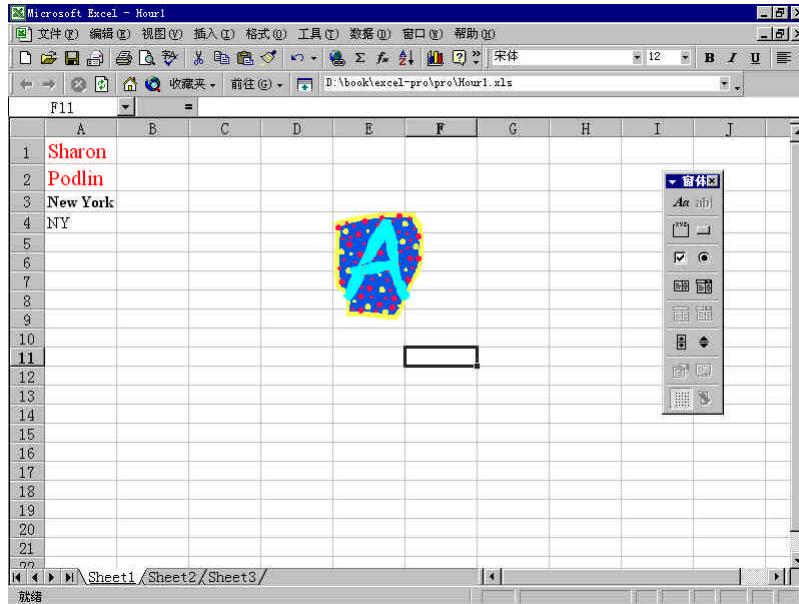
使用命令按钮是为工作表添加功能的一种非常好的方法。按钮既可见，用户又熟悉，并且不要求用户知道要执行的宏的名字。

2.5 将宏指定给图片

要执行宏并非一定要用按钮，任何可放置到工作表上的图片都可用来执行宏。当希望创建非常图形化的友好界面时，这项技术非常有用。例如，如果创建为四个不同的公司分析数据的应用程序，可用公司的标志作为按钮。要将宏指定给图片，可按照如下步骤：

- 1) 选中单元格 G3。
- 2) 选择“插入”、“图片”。
- 3) 选择“剪贴画”或者“来自文件”。
- 4) 选择要插入的图片。
- 5) 当图片显示在工作表上之后，改变图片的大小使它差不多为 2cm × 2cm 大。
- 6) 用鼠标右键单击该图片并选择“指定宏”，显示“指定宏”对话框。
- 7) 选择“BigFont”宏，并单击“确定”按钮。
- 8) 在图片之外单击鼠标左键，使图片不再被选中。
- 9) 在单元格 A9 中输入“100”。
- 10) 选中单元格 A9，并单击该图片执行宏。带有图片“按钮”的工作表如图 2-6 所示。

图2-6 可用任何剪贴画或者图片文件来执行宏



当真正开始编写 VBA 过程时，可用这种技术将它们指定给按钮或者图片。

2.6 将宏指定给工具栏按钮

如果不希望用在工作表上添加按钮或者图片的方式来执行宏，还有另一种可选的方法：

可将宏指定给工具栏按钮。Excel允许通过在工具栏中添加按钮进行自定义。当在工具栏中添加按钮后，可将一个宏指定给它。要将宏指定给工具栏按钮，可按照如下步骤：

- 1) 选择“工具”、“自定义”，显示“自定义”对话框。

也可在工具栏上单击鼠标右键，并从弹出菜单中选择“自定义”。

- 2) 选择“命令”选项卡，如图2-7所示。

图2-7 “自定义”对话框允

许向已有的工具栏添
加按钮或者创建新的
工具栏



- 3) 从“类别”下拉式列表框中选择“宏”。
- 4) 从“命令”列表框中选择“自定义按钮”。
- 5) 将自定义按钮拖动到工具栏。
- 6) 用鼠标右键单击新添加的按钮。
- 7) 选择“改变按钮图像”，为按钮选择一幅图像。
- 8) 用鼠标右键单击新添加的按钮并选择“指定宏”，显示“指定宏”对话框。
- 9) 选择“BigFont”并单击“确定”按钮。
- 10) 单击“关闭”按钮，关闭“自定义”对话框。
- 11) 在单元格A11中输入“200”。
- 12) 选中单元格A11，并单击新添加的按钮，执行“BigFont”宏。

2.7 学时小结

第一学时着重讲述录制宏的基础知识，而本学时有两个重点：其一是学习可以存放宏的不同位置；其二是学习执行宏的不同方式。最初你只能通过使用“工具”、“宏”、“宏”菜单来执行宏，现在知道了如何为宏指定快捷键，以及如何将宏指定给按钮、图片或者工具栏按钮。不要忘了，当编写VBA过程时，可以使用这些技术。

2.8 专家答疑

问题：学习宏对成为Excel开发者有什么帮助？

解答：学习宏一方面能够减少开发时间。录制宏比从头开始编写代码要快得多。另一方面是通过对将宏指定给不同对象（按钮、图片和工具栏按钮）的理解，可以学到更多设计应用程序界面时所需的基础知识。

问题：可以将VBA代码指定给按钮、图片和工具栏按钮吗？

解答：可以，具体技术和本学时中讲述的一样。

2.9 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容。答案请参考附录。

2.9.1 思考题

- 1) 判断题：快捷键只有在第一次录制宏时才能指定。
- 2) 可以存放宏的三个位置是什么？
- 3) 个人宏工作簿位于何处？
- 4) 判断题：启动Excel时个人宏工作簿会自动打开。
- 5) 判断题：Excel不允许将宏指定给已经定义的快捷键。
- 6) 将宏指定给图片的基本步骤是什么？

2.9.2 练习题

将在第一学时的练习题中创建的宏“Title”指定给一个按钮、一幅图片以及一个工具栏按钮。

第3学时 学习控件

在上一学时中，不论是否意识到，你已经开始学习用户界面设计的一些组成部分了。你学到了为应用程序设计添加按钮、图片和工具栏按钮的有关知识。在本学时中将以控件为重点对所学知识进行扩展。

本学时的重点包括：

- Excel开发过程的简要介绍
- 不同控件类型的讨论
- 向工作表添加控件
- 设置控件的格式，将控件和工作表单元格链接起来
- 在应用程序中使用用户窗体

3.1 Excel开发过程的简要介绍

不论创建什么样的应用程序，总是有多种方法可以实现。本学时将要介绍的是一种已经成功应用多年的方法。建立应用程序的第一步是了解尽可能多的与该应用程序有关的内容。这意味着需要找到如下问题的答案：

- 谁将使用这个应用程序？
- 应用程序将要使用的数据来源于何处？
- 应用程序保存什么样的数据？
- 应用程序对数据进行什么样的操作？
- 应用程序应当产生什么样的输出结果？

3.1.1 谁将使用这个应用程序

知道你的应用程序的用户是谁，将有助于你决定应用程序应当采用什么样的外观和给人什么样的总体感觉。例如，如果设计将由熟练的 Excel用户使用的应用程序，那么，你也许希望用工作表作为基本的数据输入机制。另一方面，如果你的用户对 Excel并不熟悉，那么你也许会使用窗体来进行数据输入。

3.1.2 应用程序将要使用的数据来源于何处

数据是否已经保存在 Excel工作簿中？还是从头开始输入新的数据？或者从其他数据源导入数据？应用程序的数据当前保存在什么位置将成为影响开发应用程序方法的一个重要因素。

3.1.3 应用程序保存什么样的数据

数据将保存在和应用程序相同的工作簿中还是保存到别的工作簿中？或者数据将保存为其他类型的数据文件，例如 Microsoft Access、Microsoft SQL Server等等？如果数据将保存在工作簿中，那么在开始编写代码之前就应该创建列数差不多够用的工作表。

3.1.4 应用程序对数据进行什么样的操作

换句话说，应用程序是否需要对数据进行处理和分析？是否需要根据数据绘制图表？是否需要对数据进行计算？是否要建立数据透视表？是否需要对数据进行分类？通过提出类似这些问题，你会明白需要创建什么类型的工作表以及工作表需要什么样的公式。你也可以决定可用宏录制器录制什么类型的操作。

3.1.5 应用程序应当产生什么样的输出结果

这个问题会导致你设计和创建另一组工作表。可能需要为将要输出的每个报表创建工作表。

回答这些问题实际上是在对如何设计应用程序进行引导。了解上面所述的信息将使你作为Excel开发者的工作变得更容易。

3.2 不同类型的控件

新术语 你可以向工作表或者用户窗体添加控件。用户窗体实际上是创建的窗口或者对话框，是用户界面的一部分。对能够添加到工作表中的控件的限制比添加到用户窗体中的控件要严格。首先要讨论的控件是能够添加到工作表中的控件，这些控件也可以添加到用户窗体中。

开始前请关闭Excel中所有已经打开的工作簿，打开一个新的工作簿。在工具栏上单击鼠标右键，从菜单中选择“窗体”，显示“窗体”工具栏如图3-1所示。这个工具栏包括16个控件，但是只有9个可用，这9个控件可以放置到工作表上。

图3-1 “窗体”工具栏包括多种可
放置到工作表上的控件

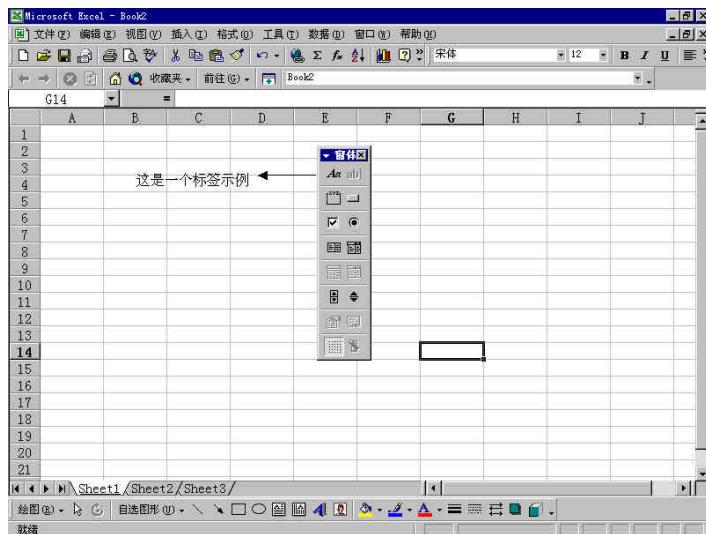


新术语 “窗体”工具栏上的第一个控件是标签。标签是静态的文本区域，用于标识其他界面元素或者提供信息。因为标签是静态控件，所以用户不能改变它们的内容。标签的例子如图3-2所示。

在图3-2中，箭头从“窗体”工具栏上的标签控件指向工作表上的标签对象。

下一个可用于工作表的是分组框控件。分组框控件用于将其他控件进行组合。分组框控件提供可见的方框，以便用户知道哪些是与同一内容有关的组合起来的控件。要查看分组框的例子，请选择“文件”、“打印”。“打印内容”分组框包括一系列用于选择要打印的工作表区域的选项按钮。

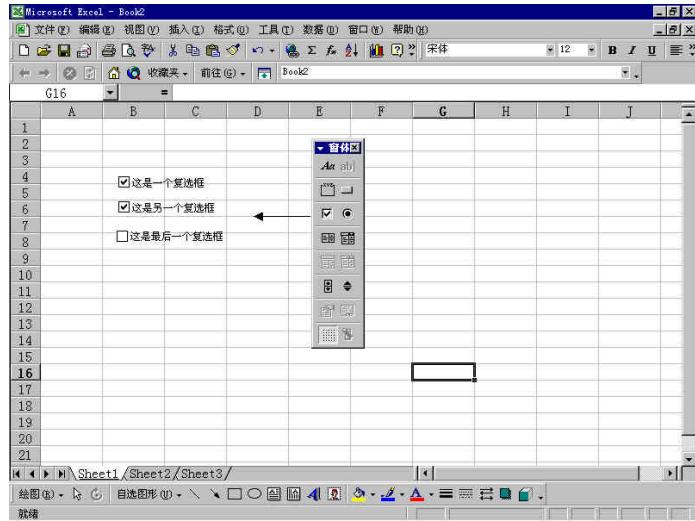
图3-2 “标签”允许向工作表中放置静态文本，不用放置到单元格中



“窗体”工具栏中分组框之后的控件是在第二学时“对记录下来的宏进行处理”中已经熟悉的按钮控件。因为用户对按钮控件很熟悉，所以适于在应用程序中使用。用户在 Windows 中到处都能见到诸如“确定”、“取消”、“是”、“否”等按钮。

下面两个控件是复选框和选项按钮，它们有一个共同的目的：允许用户在各种选项中进行选择，这是它们的相同点。复选框如图 3-3 所示，是一个开关控件，这意味着在该控件上重复地单击将不断地打开和关闭它。如果多个复选框控件组合在一起，那么用户可以选中一个、几个或者全部复选框。复选框类似 A 与/或 B 选择（可以选中复选框 1，与/或复选框 2，与/或复选框 3，等等）。

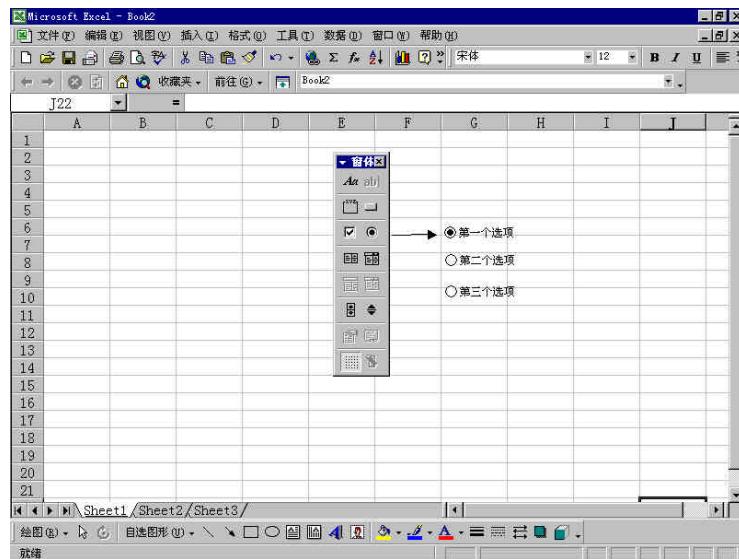
图3-3 复选框使用户易于从多个选项中进行选择



另一方面，如果将一组选项按钮组合在一起（如图 3-4 所示），那么只能选中其中的一个。选项按钮类似或者 A 或者 B 的选择（选中选项按钮 1，或者选项按钮 2，或者选项按钮 3，等等）。

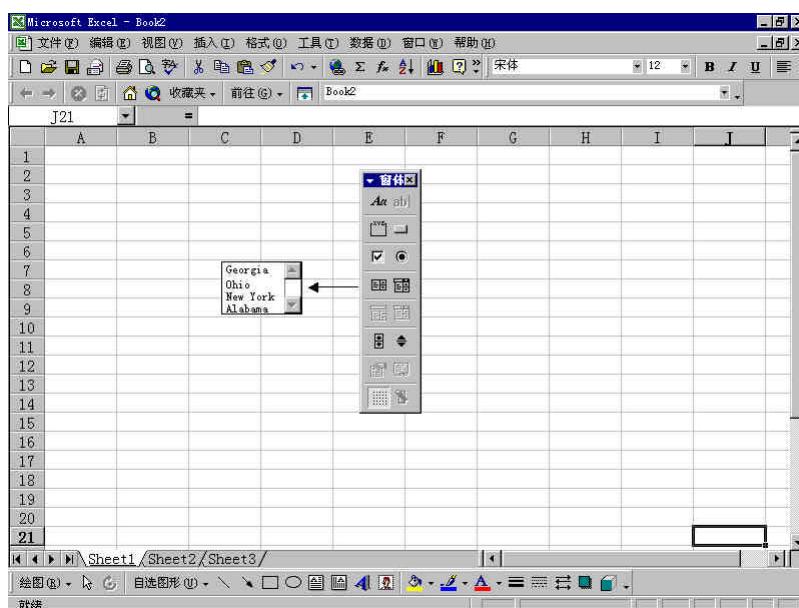
通常情况下，可用分组框对复选框和选项按钮进行组织。

图3-4 选项按钮使用户知道他们只能选择一个



列表框和组合框控件用来显示从多个选项中进行选择。例如，如果希望用户从一系列区域中进行选择，可以把各个区域放置在列表框中。列表框（如图 3-5 所示）只允许用户进行单选。而组合框（如图 3-6 所示）则允许用户从列出的项目中进行选择或者输入一个其他值。组合框的名字来源于它是列表框和文本框功能的组合。

图3-5 列表框控件允许用户从多个选项中选择，如果可用的选择太多，列表框中不能一次显示出来，则可以使用滚动条来查看其他选项

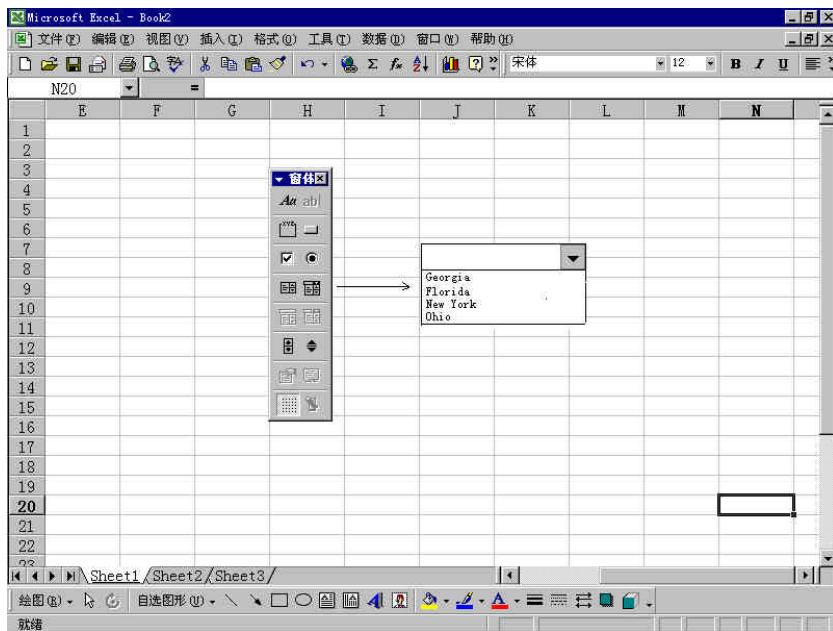


列表框和组合框在允许用户从可能的选项中进行选择的功能上和选项按钮相似。许多开发者都采用如下的使用规则：如果选项多于三个，则使用列表框或者组合框。

图3-5列表框允许用户从多个选项中进行选择。如果选项太多不能在列表框中显示出来，

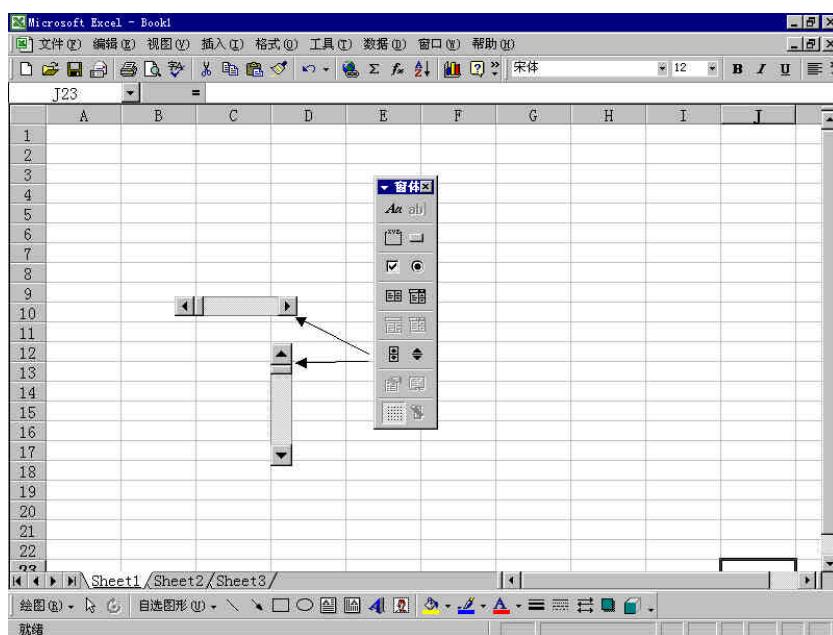
用户可用滚动条查看其他的选项。

图3-6 组合框控件是使用
户从多个选项中进
行选择的另一个解
决途径



可以放置到工作表上的最后两个控件是滚动条控件和微调控件。滚动条控件(如图3-7所示)不是你所见到用来给很长的窗体添加滚动能力的控件,而是一种选择机制。如果你曾经使用过图形程序,那么也许曾经用过滚动条来使画面变得更亮或者更暗,或者从同一颜色的不同数值中进行选择。

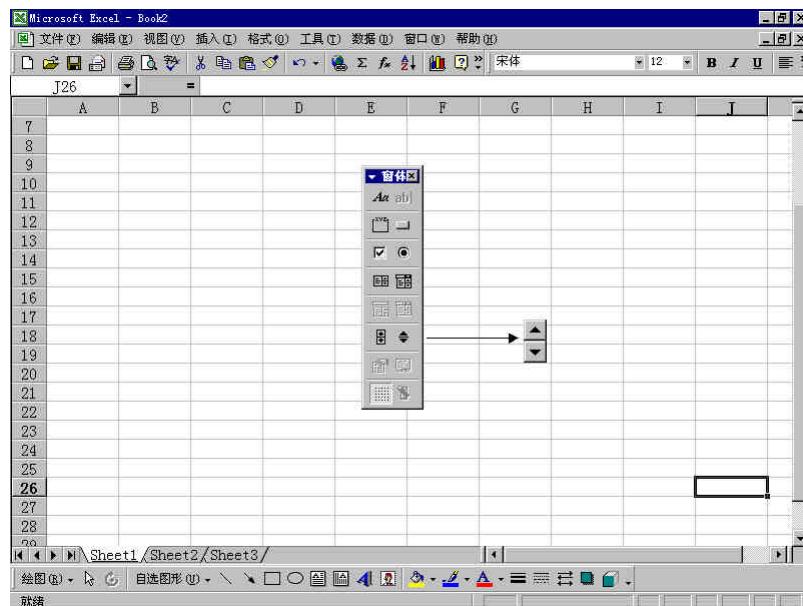
图3-7 滚动条包括水平滚
动条和垂直滚动
条,这取决于如何
设计



微调控件(如图3-8所示)也是一种数值选择机制。如果你曾经改变过Windows系统的日

期或时间，那么就可能使用过微调控件。

图3-8 微调控件是一个复杂的控件，允许用户单击箭头按钮来选择数值



3.3 向工作表添加控件

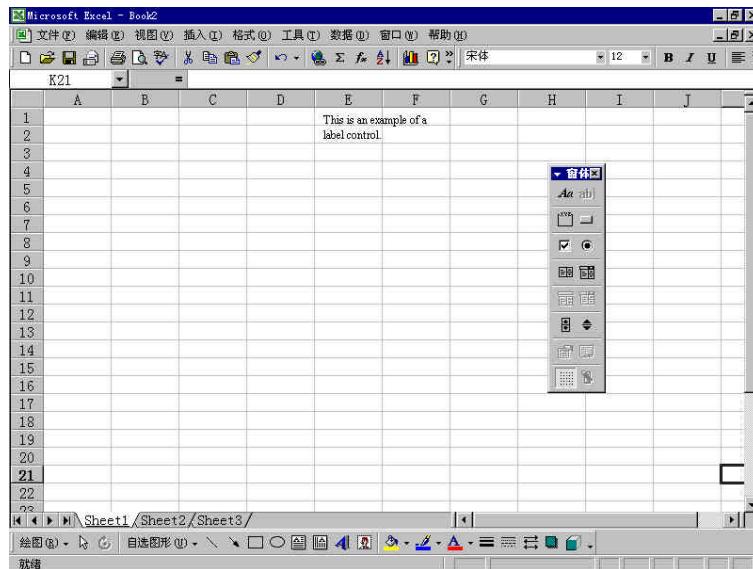
我经常告诉我的学生，如果他们能够在Windows的附件——画图中画矩形，那么他们就具有在Excel环境中设计界面的技巧。这主要是为了说明在Excel中设计界面是如何简单。要将任何可用的控件放置到工作表或者窗体上，可以按照如下步骤进行：

- 1) 从工具栏上选择要用的控件。
- 2) 将鼠标定位到希望该控件所要放置的位置。
- 3) 按下鼠标左键，拖动鼠标直至矩形变为希望的控件的大小。
- 4) 释放鼠标左键，这样控件就添加完毕。

设计界面的过程和绘制一组矩形的过程差不多。要说明这一点，请完成如下步骤：

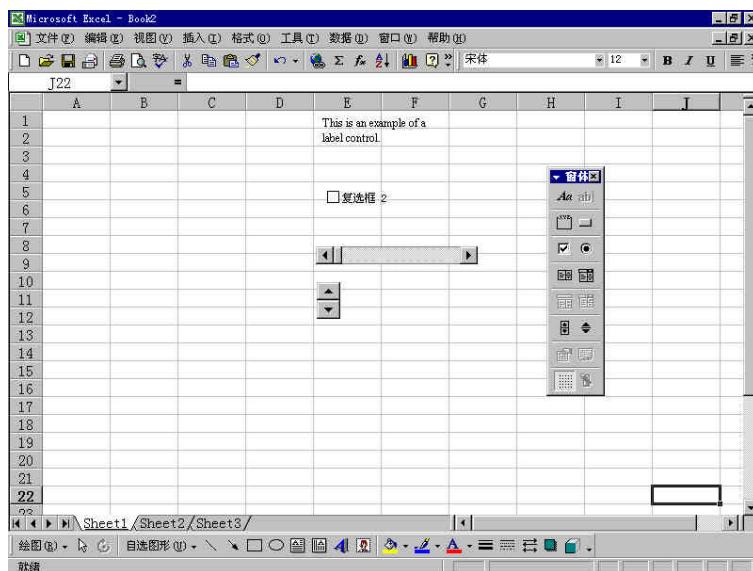
- 1) 打开一个新的工作簿。
- 2) 显示“窗体”工具栏。
- 3) 选择“标签”按钮。
- 4) 将鼠标定位到单元格E1上，此时鼠标看上去像一个加号。
- 5) 按下鼠标左键，拖动鼠标直至矩形变为大约四个单元格，释放鼠标左键。
- 6) 在Label1上的L前单击鼠标左键，按下Delete键直至标签上的文本完全删除。
- 7) 输入“ This is an example of a label control. ”。
- 8) 在标签之外单击鼠标左键，使标签不再被选中。完成后的外观看上去应该和图 3-9相似。
- 9) 从“窗体”工具栏中选择复选框控件。
- 10) 将鼠标定位到单元格E5上，此时鼠标看上去像一个加号。
- 11) 按下鼠标左键，拖动鼠标直至矩形变为大约四个单元格，释放鼠标左键。

图3-9 通过在工作表中添加标签，可以放置不被工作表上的单元格限制的文本



12) 使用与上述相同的步骤，向工作表中添加一个滚动条控件和一个微调控件，直至工作表看上去和图3-10相似。

图3-10 添加了控件后的工
作表



在添加完控件之后，可以调整它们的大小和位置，直到满意为止。要改变控件的大小和位置，首先应当选中该控件。在控件上单击鼠标左键不能选中控件进行编辑，而应当在控件上单击鼠标右键。这使得控件处于一种编辑模式，并且显示出一个菜单。因为不用使用菜单，所以可在选中控件的边框上单击鼠标左键，被选中的控件周围出现灰色的边框。要移动控件，可在控件的灰色边框上移动鼠标，直到鼠标形状变为四个方向的箭头。按下鼠标左键将控件拖动到新的位置。注意被选中的控件的边框上有八个小方块，这些方块是调节控件大小的控制柄。要改变控件的大小，将鼠标移动到尺寸控制柄上，直到鼠标变为双向箭头。按下鼠标左键，拖动尺寸控制柄直至对控件大小满意为止。

如果希望处理、移动多个控件或者调整多个控件的大小等，可选中第一个控件，一直按着Ctrl和Shift键，再用鼠标单击另一个控件，则两个控件都被选中。继续按着Ctrl和Shift键，单击控件，直到选中希望改变的所有控件为止。

3.4 设置控件的格式

你也许想知道怎样使用已经放置在工作表上的控件。可以将宏指定给它们，此外，还有别的使用它们的方法。可用它们来为工作表输入数据，通过设置控件的格式可以做到这一点。设置控件的格式可以控制控件的外观和功能。设置控件的格式可按照如下步骤进行：

- 1) 选中要放置在工作表上的复选框控件。
- 2) 在控件上单击鼠标右键，选择“设置控件格式”，显示“设置控件格式”对话框。
- 3) 选择“控制”选项卡，如图3-11所示。

图3-11 “设置控件格式”

对话框中的“控制”选项卡允许将控件的值和单元格链接起来



- 4) 在“单元格链接”文本框中，输入“A1”并单击“确定”按钮。
- 5) 在复选框之外单击鼠标左键，使复选框不再被选中。
- 6) 用鼠标左键单击复选框将其选中，在单元格A1中显示出“TRUE”，这意味着复选框已经被选中。
- 7) 再次在复选框上单击鼠标左键，清除该复选框。单元格A1显示“FALSE”。

可以用Excel的“IF”函数来检测单元格A1的数值，并根据该单元格的值是“TRUE”还是“FALSE”执行不同的操作或者计算。例如，假设希望选中复选框时将一个数值增加20%，可以使用如下的公式： $= IF (A1 = TRUE , A4 \times 1.2 , A4)$ 。

- 8) 选中刚才创建的滚动条控件。
- 9) 在控件上单击鼠标右键，选择“设置控件格式”，显示“设置控件格式”对话框。
- 10) 选择“控制”选项卡，如图3-12所示。
- 11) 在“单元格链接”文本框中输入“A3”并单击“确定”按钮。

图3-12 选中的控件不同，则“控制”选项卡显示的内容也不相同



12) 在滚动条控件之外单击鼠标左键，使滚动条不再被选中。

13) 用鼠标单击滚动条上向右的箭头，则单元格 A3的数值增加1。继续单击滚动条上向右的箭头增加单元格 A3的数值。

14) 将工作簿保存为“Control”并将其关闭。

现在可以看到，通过使用控件、设置格式和公式，可以执行一些简单的自动化任务（至少对你的用户来说是这样）。

3.5 给控件命名

当创建一个控件时，Excel会为它指定一个不很形象的名字，诸如“复选框 1”或者“滚动条7”等等。当准备用这些控件来编写代码时，你会发现，如果使用 Excel定义的名字，很难记住哪个控件是用来干什么的。大多数开发者在为控件命名时都使用一种命名的惯例。通常情况下，命名的惯例使用名字的前三个字母的前缀表示控件的类别。例如，如果创建一个与货运优先级有关的复选框，那么可以将其命名为“chkPriorityShipping”。在本例中，chk是用来表示复选框的前缀。表 3-1列出了通常用来为控件命名的前缀。

表3-1 推荐的控件命名前缀

控件类型	前 缀
复选框	chk
组合框	cbo
命令按钮	cmd 或者 btn

因为某种原因，Microsoft给命令按钮取了两个名字。在“窗体”工具栏中，称为按钮，而在工具箱中称为命令按钮。在 Visual Basic中，称之为命令按钮。也许这个控件最常用的前缀应该是 cmd。

分组框

grp或者fra

因为某种原因，Microsoft给分组框(Group Box)取了两个名字。在“窗体”工具栏中，称为分组框，而在工具箱中称为框架 (frame)。在 Visual Basic中，称之为框架(frame)。也许这个控件最常用的前缀应该是 fra。

(续)

控件类型	前缀
图像	img
标签	lbl
列表框	lst
多页 (MultiPage)	mul
选项按钮	opt
引用编辑 (RefEdit)	ref
滚动条	hsb或者vsb , 取决于水平的还是垂直的
微调项	spn
选项卡条 (TabStrip)	tab
文本框	txt
切换按钮	tog

为控件取名的方法基本上和给单元格或者区域取名的方法相同。选中某个控件，再用位于公式栏上的“名字”编辑框输入控件的名字。可按照如下步骤进行：

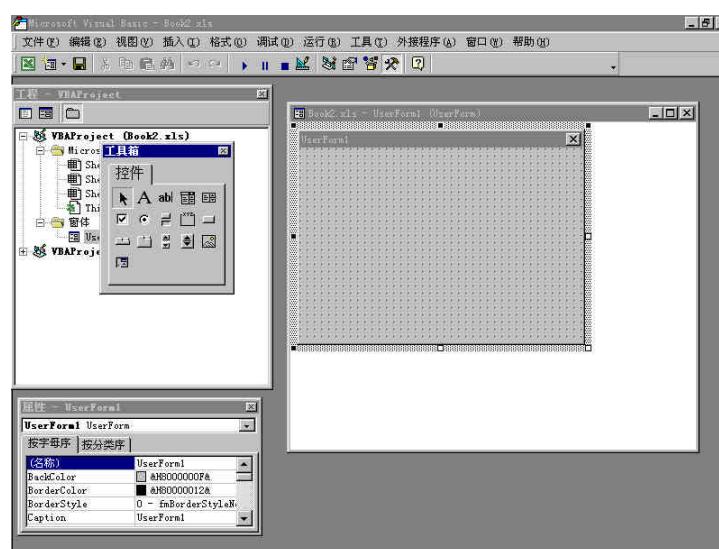
- 1) 在标签上单击鼠标左键将其选中。
- 2) 单击规则栏上的“名字”编辑框（此时也许会看到控件当前的名字是“标签 1”）。
- 3) 输入“lblExample”作为控件的名字并按回车键。这样就为控件更改了名字。

3.6 使用用户窗体

如果希望为应用程序添加专业级的外观，并且使用户能够更容易地输入数据，那么应该使用用户窗体。用户窗体可作为应用程序的对话框和窗口。向用户窗体添加控件和向工作表中添加控件相同。然而，在第一步中，需要为应用程序添加用户窗体。这可以通过 Visual Basic 编辑器实现。具体可以按照如下步骤：

- 1) 创建一个新的工作簿。
- 2) 选择“工具”、“宏”、“Visual Basic 编辑器”，打开 Visual Basic 编辑器。
- 3) 选择工具栏上的“插入用户窗体”按钮或者选择“插入”、“用户窗体”，显示出新添加的用户窗体，如图 3-13 所示。

图3-13 用户窗体允许为应
用程序设计窗口和
对话框



新添加的用户窗体具有一个小小的名为 UserForm1 的标题栏。窗体中显示的网格是用来帮助放置控件的设计工具。当窗体在应用程序中显示出来时不会出现网格。

当用户窗体打开时，“工具箱”自动显示出来，如图 3-14 所示。在工具箱中可以看到许多已经熟悉的控件，包括标签控件、复选框控件、选项按钮控件和命令按钮控件。此外还有几个附加的控件。

图3-14 工具箱中包括了可以放置到用户窗体上的控件



【口】新术语 切换按钮控件如果被选中，那么会保持被按下的状态。如果再次单击它，则恢复为没有按下的状态。几个工具栏按钮实际上是切换按钮。例如，当单击“格式”工具栏上的“加粗”工具栏按钮时，这个按钮就一直保持被按下的状态（显示为凹下），表示打开了“加粗”状态。再次单击“加粗”工具栏按钮可以关闭“加粗”状态，此时按钮看上去恢复原状。

【口】新术语 选项卡条(TabStrip)是包含多个选项卡的控件。选项卡条控件用来将相关的信息进行组织和分组。例如，你也许希望用选项卡条来显示各个区域的销售信息，每个区域都有自己的选项卡。默认设置下，选项卡条包括两页，称为 Tab1 和 Tab2。可以添加更多的选项卡。

【口】多页 多页控件看上去和选项卡控件相似，是包括一页或者多页的控件。选项卡条控件的各个选项卡具有相似的外观，给人以相似的感觉。而多页控件的各页是包含各自控件的窗体，它们都具有各自的布局。多页的例子是“选项”对话框中的多页控件，选择“工具”、“选项”便可以看到。

【图】图像控件 图像控件允许向用户窗体上放置图片。图像控件可以显示的图像文件类型包括：

- .bmp
- .cur
- .gif
- .ico
- .jpg
- .wmf

【图】RefEdit 这是工具箱中的最后一个控件。RefEdit控件看上去和文本框控件相似，不同之处在于RefEdit控件有一个按钮，通过这个按钮，可以将用户窗体折叠起来，以便选择一个区域。图3-15显示了一个包含RefEdit控件的窗体。图3-16显示了这个RefEdit控件的按钮被单击，

用户窗体被折叠起来后的外观。在使用“粘贴函数”对话框时，你也许已经使用过这种类型的控件。

图3-15 在许多Excel对话框中都可以发现，RefEdit增加了区域选择的功能

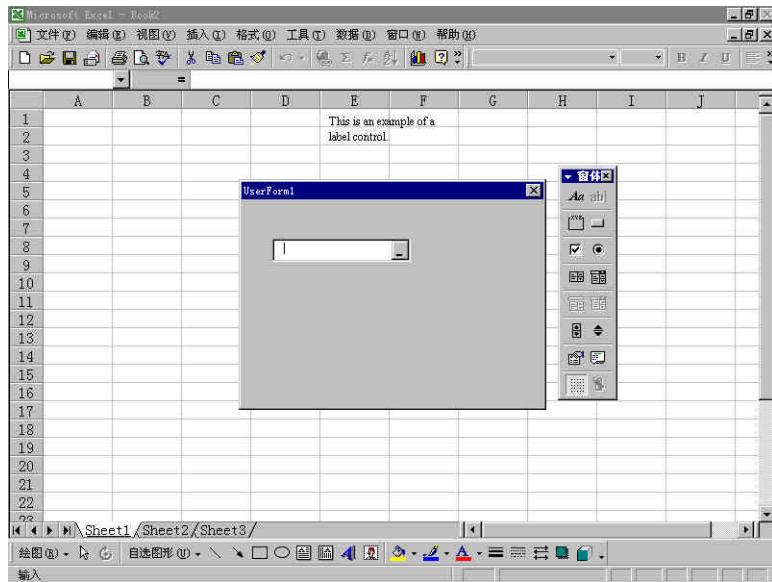
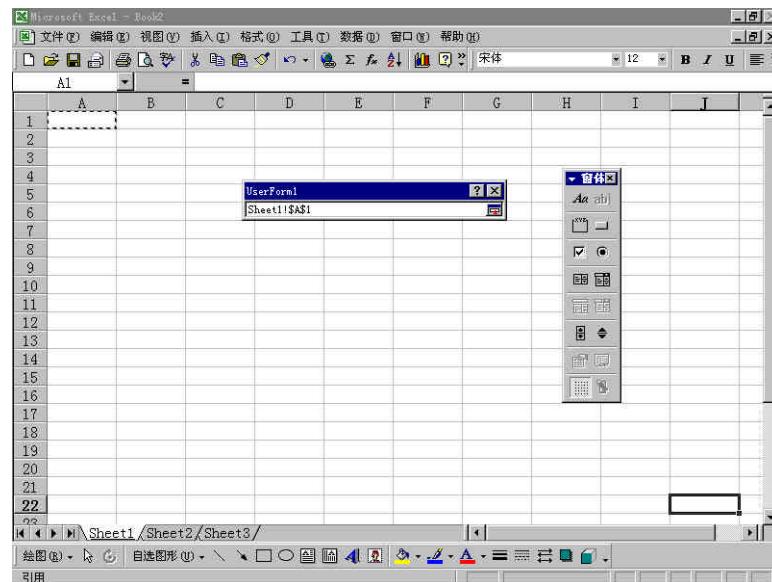


图3-16 使用RefEdit控件可以选择一定区域的单元格

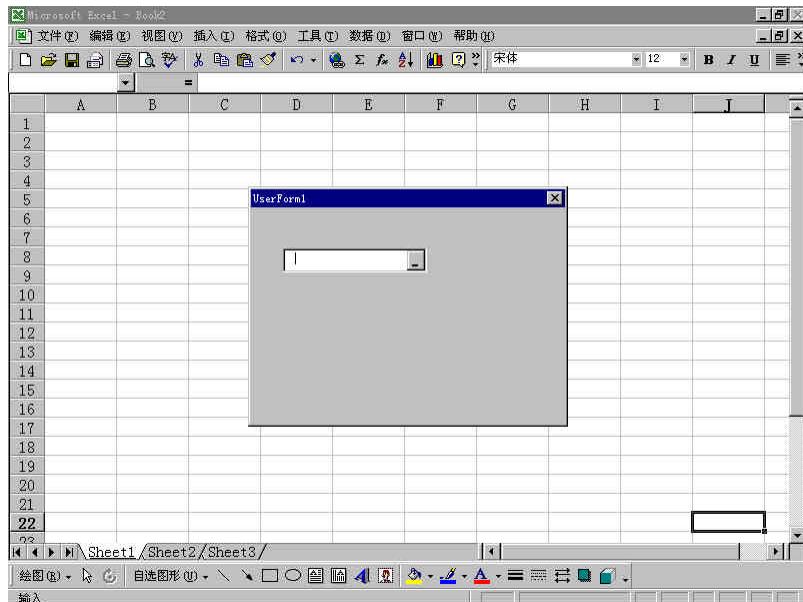


将这些控件添加到窗体上和向工作表添加控件完全一样。与在工作表上操作一样，你也能够移动控件并且改变控件的大小。用户窗体本身的大小也可以通过窗体的尺寸控制柄来调节。

当对用户窗体的外观设计感到满意时，可以对它进行预览，具体方法是选择“运行”、“运行子函数/用户窗体”。这样，这个窗体就会显示在当前工作簿的前面，如图3-17所示。用窗体的关闭按钮（在窗体的右上角可以找到）可以关闭这个窗体，回到Visual Basic编辑器中。在接下来的学时里，我们将学习如何对用户窗体进行自动化。

运行窗体的另一种方法是按下 F5键。

图3-17 运行用户窗体可以预览这个窗体在应用程序中的外观



3.7 学时小结

学完本学时后，你应该具备了为应用程序设计界面的比较全面的基础知识。现在你已经知道如何将控件添加到工作表，也知道如何向应用程序添加用户窗体，以及在用户窗体上放置控件。

此后用来学习Excel应用程序开发的时间将从界面转移到编写自动化解决方案的代码上。请准备好开始编程吧！

3.8 专家答疑

问题：怎样决定放置控件的位置？如何选择使用工作表还是用户窗体？

解答：这完全取决于个人的爱好和应用程序的用户。如果用户对Excel非常熟悉，那么他们也许更愿意以工作表的形式工作。在这种情况下，应该将控件直接放置在工作表上。如果你的用户对Excel的使用还处于初级阶段，或者希望应用程序给人以比较独特的感觉，那么应该使用用户窗体。

问题：什么情况下应当使用选项卡而不是多页控件？

解答：通过询问自己每个选项卡/页是否应该具有相同的布局，就可以知道应该使用哪种控件。如果回答是肯定的，那么就需要选项卡。相反如果回答是否定的，那么就需要多页控件。

3.9 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容。答案请参考附录。

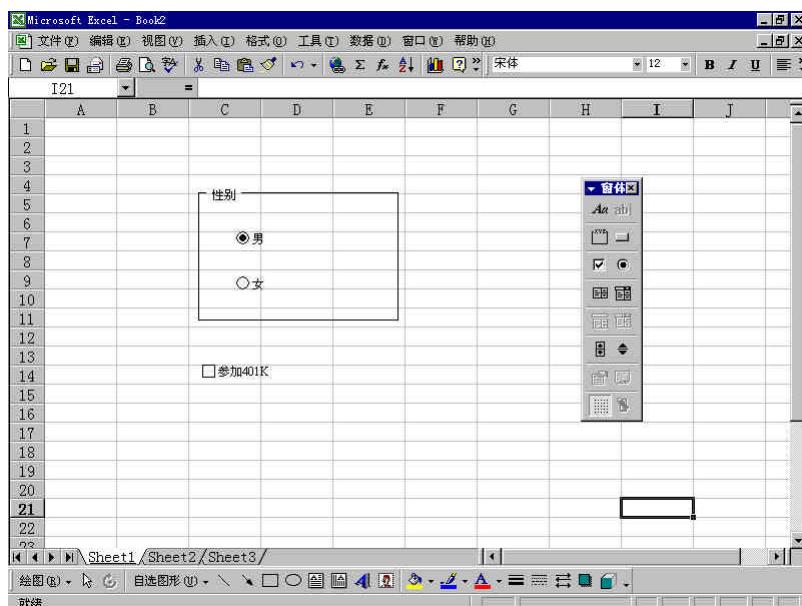
3.9.1 思考题

- 1) 举出两种可以让用户从多个选项中选出某个选项的控件。
- 2) 判断题：只有当 Visual Basic 编辑器激活时才能添加用户窗体。
- 3) 如何将控件和单元格链接起来？
- 4) 判断题：在 Visual Basic 编辑器中看到的用户窗体上的网格在运行时仍然会显示出来。
- 5) _____是显示静态文本的控件。

3.9.2 练习题

本练习主要是给你更多的练习使用控件的机会。打开一个新的工作簿，向工作表中添加必要的控件，使它看上去和图 3-18 相似。将选项按钮命名为 optMale 和 optFemale，将复选框命名为 chk401K。

图3-18 练习中完成后的工
作表



第4学时 理解变量和常量的作用

从本学时开始，将进入VBA程序设计的核心世界。本学时中将介绍五个重要的概念：模块、过程、变量、常量和作用域。

本学时的重点包括：

- 对模块的概览
- 对过程进行讨论，包括如何创建过程
- 如何使用变量
- 如何使用常量
- 作用域如何起作用

4.1 代码存在的基础：模块

VBA代码必须寄存在某个地方，这个地方就是模块。有两种基本类型的模块：类模块和标准模块。模块中的每个过程或者是函数过程，或者是子程序。本学时的后面部分将要讨论函数过程和子程序的区别。

新术语 模块的正式定义是作为一个单元保存在一起的VBA定义和过程的集合。

新术语 VBA允许你创建自己的对象，对象的定义包含在类模块中。

你的大部分工作集中在标准模块上（通常简称为模块）。当录制宏时，如果不存在模块，Excel会自动为你创建。如果愿意，也可以添加附加的模块。Excel和VBA并不关心过程位于哪个模块中，只要过程位于打开的工作簿中即可。

4.2 对模块的概览

新术语 过程被定义为VBA代码的一个单元，过程中包括一系列用于执行某个任务或是进行某种计算的语句。工作簿中的每个过程都有唯一的名字加以区分。

有两种不同的过程：子程序和函数过程。子程序只执行一个或者多个操作，而不返回数值。当录制完宏查看代码时，所看到的就是子程序的例子。宏只能录制子程序，而不能录制函数过程。一个子程序的例子如程序清单4-1所示。

程序清单4-1 子程序的例子

```
1: Sub cmdSmallFont_Click ()  
2:     With Selection.Font  
3:         .Name = "Arial"  
4:         .FontStyle = "Regular"  
5:         .Size = 16  
6:     End With  
7: End Sub
```

上面列出的过程实际上是一个事件过程。通过它的名字，就可以知道这是一个事件过程。这个过程的名字是由一个对象的名字 cmdSmallFont 和一个事件的名字 Click 组成的，两者之间用下划线分开。如果还不明白，可以告诉你，本例中的 cmdSmallFont 是一个命令按钮的名字。这就是说，当单击这个名为 cmdSmallFont 的命令按钮时，就会运行这个事件过程。

函数过程通常情况下简称为函数，要返回一个数值。这个数值通常是计算的结果或者是测试的结果，例如 True 或者 False。正如前面所说，可用 VBA 创建自定义的函数。如果读过 Microsoft 有关 VBA 和 Excel 的材料，那么，你也许看到可以用 VBA 对 Excel 进行扩展的申明。事实上，可在工作表的单元格中使用你创建的函数。程序清单 4-2 包括一个计算价格的 10% 作为运费的简单例子。

程序清单 4-2 简单的用户定义的函数示例

```

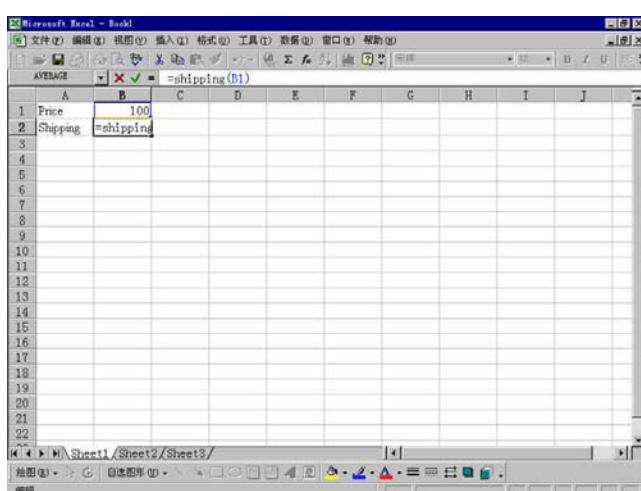
1: Public Function Shipping(Price)
2:     Shipping = Price * 0.1
3: End Function

```

请注意，这个函数使用了一个参数（ Price ）。子程序和函数过程都可以使用参数。不论 Price 的值是多少，它都将决定运费额。 Price 可以是一个数字，也可以是对单元格的引用。函数返回计算出来的运费，这个函数可以用在单元格中，如图 4-1 所示。

图 4-1 用户自定义的函数

Shipping 用来根据价
格计算运费

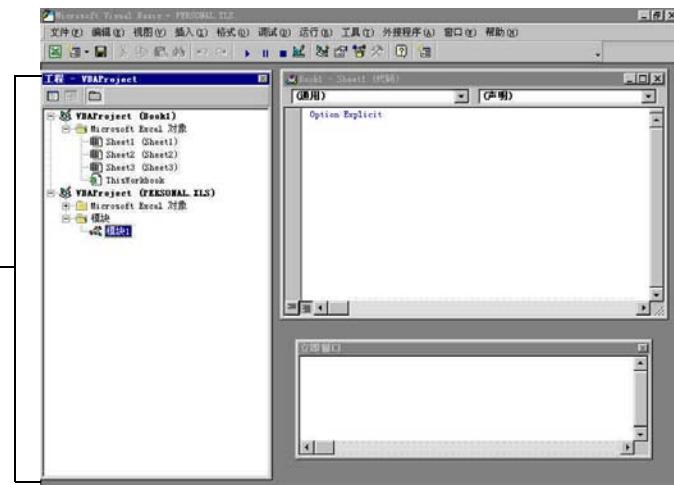


4.2.1 创建过程

创建第一个过程要求两个基本的步骤。首先，需要向工作簿中添加一个模块。接着需要向模块中添加一个工程。对于创建的每个应用程序，只需添加一次模块。可以使用多个模块，但是这不是必要的。某些开发者喜欢使用多个模块，以便根据他们的目的或者窗体等等对过程进行组织。在本练习中，创建的过程只显示一个消息框。在本练习中使用 MsgBox 是为了提供一个可视的例子，虽然我们还没有介绍过 MsgBox 语句，但将在本例中使用它。要创建这个过程，可按照如下步骤：

- 1) 打开一个新的工作簿。
- 2) 选择“工具”、“宏”、“Visual Basic 编辑器”，打开 Visual Basic 编辑器窗口。
- 3) 在 Visual Basic 编辑器的左面，可以看到工程资源管理器窗口（如图 4-2 所示）。在工程资源管理器的“ ThisWorkbook ”上单击鼠标右键，选择“插入”、“模块”，这样就将一个模块添加到应用程序中了。

图4-2 工程资源管理器对组
成应用程序的元素进
行跟踪

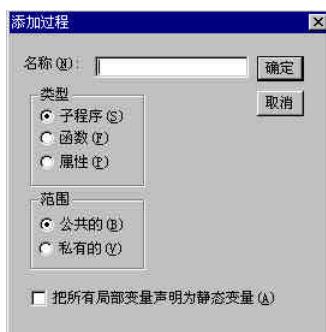


如果你没有看到工程资源管理器窗口，可按 Ctrl+R 键。

在模块的顶部，可能会看到一条“ Option Explicit ”语句。“ Option Explicit ”语句用来要求对所有变量进行明确的定义，具体的定义语句可用Dim、Private、Public、ReDim或者Static。如果试图使用一个没有定义的变量名，那么在编译时就会出现错误。如果在“工具”、“选项”对话框的“模块”选项卡中选中了“要求变量定义”复选框，那么新的模块就会自动添加“ Option Explicit ”语句。

4) 选择“插入”、“过程”，显示“添加过程”对话框，如图 4-3 所示。

图4-3 “添加过程”对话框
可帮助你创建新的子
程序和函数



5) 输入“FirstVBAProc”作为过程名。在“类型”分组框中，确认选择了“子程序”。单击“确定”按钮，这样一个新的过程就添加到模块中了，如图 4-4 所示。

6) 插入点必须是过程空行的所在位置。按一次 Tab 键，再输入如下语句：

MsgBox " This is my 1st VBA procedure. "

输入 MsgBox 后，会弹出一个消息框告诉你有关这条命令的信息，称为自动快速信息技术。

VBA并不要求按下Tab键产生缩进，缩进是为了使代码易于阅读和修改。

7) 按下回车键，完成后的过程如图 4-5 所示。

图4-4 新添加的过程以“Public Sub”语句开始，而以“End Sub”语句结束

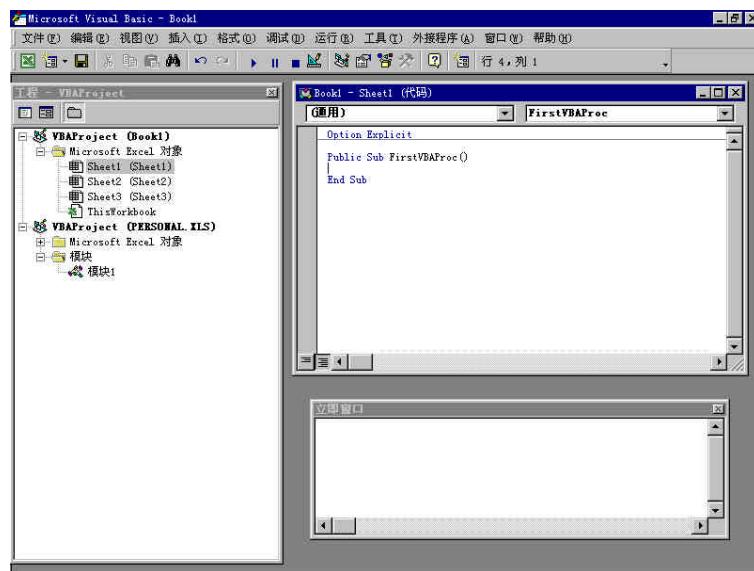
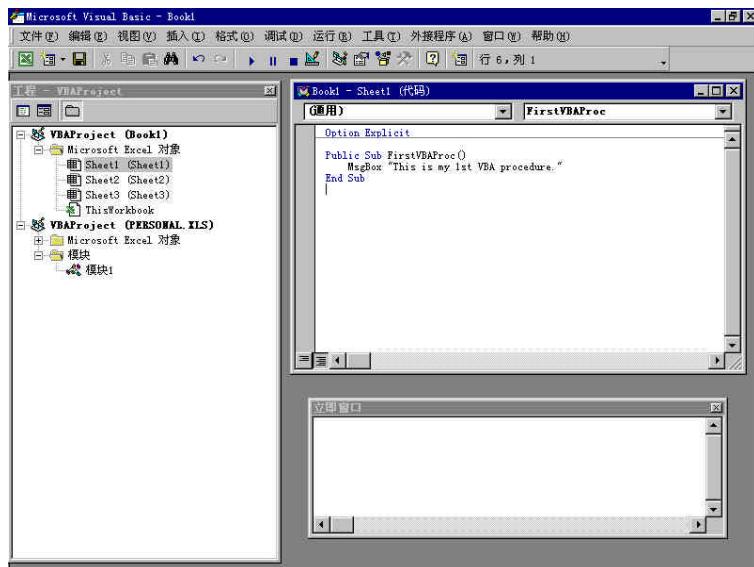


图4-5 第一个VBA过程只有三行VBA代码



VBA对子程序和函数有如下的命名规则：

- 第一个字符必须是字母。
- 名字中可以包含字母、数字和下划线。
- 名字中不能包含空格、句号、惊叹号，也不能包含字符 @、&、\$和#。
- 名字中最多包含 255 个字符。

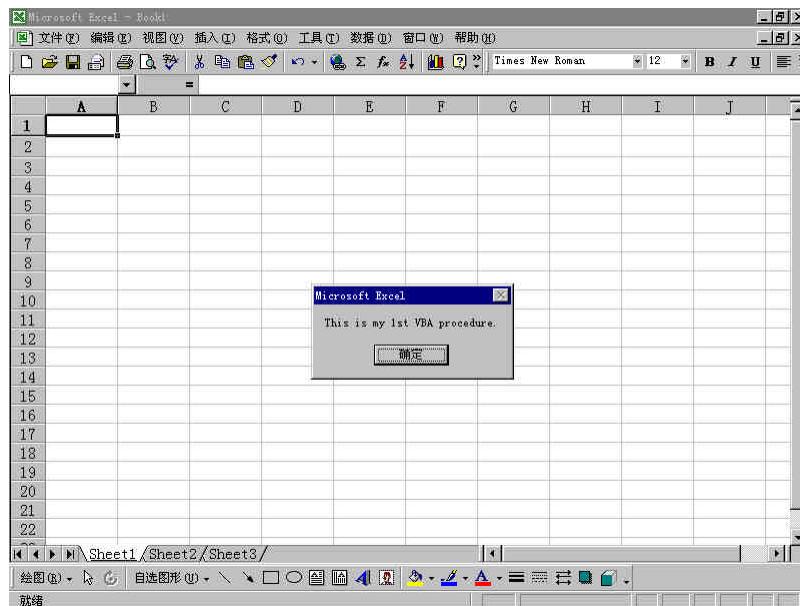
4.2.2 运行宏

创建这个过程后，可以立刻运行它。运行一个过程有几种方法：可以使用“运行”菜单、

“运行子程序/用户窗体”工具栏按钮或者按下F5键。要运行一个过程，可以按照如下步骤：

- 1) 单击“运行子程序/用户窗体”工具栏按钮，过程执行并显示一个消息框，如图4-6所示。

图4-6 在过程中输入的一行代码就可以显示消息框



- 2) 单击“确定”按钮。

4.3 保存对模块做出的改变

要保存新过程，需要保存过程所驻留的工作簿。可以用Visual Basic编辑器保存工作簿。具体步骤如下：

- 1) 选择“文件”、“保存工作簿”。因为本工作簿还没有保存过，所以需要给它取名。
- 2) 输入“First”作为文件名并按回车键，则工作簿、模块和过程都保存下来了。

4.4 变量

新术语 变量是用于临时保存数值的地方。每次应用程序运行时，变量可能包含不同的数值，而在应用程序运行时，变量的数值可以改变。

为了说明为什么需要使用变量，可按照如下步骤创建一个简单的过程：

- 1) 创建一个名为“ WhatsYourName ”的新过程。
- 2) 在过程中输入如下代码：

```
InputBox "Enter your name:"
```

现在不要担心InputBox语句的语法，将在第6学时中了解到有关这条语句的更多信息。

- 3) 按下F5键运行过程，这时会显示一个输入框，要求输入你的名字。

- 4) 输入你的名字并单击“确定”按钮，则过程结束。

你输入的名字到哪里去了？如何找到用户在输入框中输入的信息？在这种情况下，需要使用变量来保存用户的输入结果。

4.4.1 变量数据类型

新术语 使用变量的第一步是了解变量的数据类型。变量的数据类型控制变量允许保存何种类型的数据。表4-1列出了VBA支持的数据类型，还列出了各种数据类型的变量所需要的存储空间和能够存储的数值范围。

表4-1 VBA数据类型

数据类型	存储空间	数值范围
Byte	1字节	0 ~ 255
Boolean	2字节	True或者False
Integer	2字节	- 32 768 ~ 32 767
Long(长整型)	4字节	- 2 147 483 648 ~ 2 147 483 647
Single	4字节	负值范围： - 3.402823E38 ~ - 1.401298E - 45 正值范围： 1.401 298E - 45 ~ 3.402 823E38
Double	8字节	负值范围： - 1.797 693 134 862 32E308 ~ - 4.940 656 458 412 47E - 324 正值范围： 4.940 656 458 412 47E - 324 ~ 1.797 693 134 862 32E308
Currency	8字节	- 922 337 203 685 477.5808 ~ 922 337 203 685 477.580 7
Decimal	14字节	不包括小数时： +/- 79 228 162 514 264 337 593 543 950 335 包括小数时： +/- 7.922 816 251 426 433 759 354 395 033 5
Date	8字节	100年1月1日 ~ 9999年12月31日
Object	4字节	任何引用对象
String(长字符串)	10字节+1字节/字符	0 ~ 约20亿
String(固定长度)	字符串的长度	1 ~ 约65 400
Variant(数字)	16字节	Double范围之内的任何数值
Variant(文本)	22字节+1字节/字符	数据范围和变长字符串相同

作为VBA程序员，一个目标是选择需要存储空间尽可能小的数据类型来保存需要存储的数据，这正是表4-1提供各种数据类型的存储空间的原因。例如，如果要保存诸如班级学生总数这样的小数字，那么Byte数据类型就足够了。在这种情况下，使用Single数据类型只是对计算机存储空间的浪费。

4.4.2 用Dim语句创建变量

现在，你对变量可以使用的数据类型已经比较熟悉了，以下我们将创建变量。创建变量可以使用Dim语句，创建变量通常称为声明变量。

新术语 Dim语句的基本语法如下：

Dim 变量名 As数据类型

这条语法中的变量名代表将要创建的变量名。对变量的命名规则和对过程的命名规则相同。这条语句中的数据类型部分可以是表4-1中的任何一种数据类型。

变量名必须以字母开始，并且只能包含字母、数字和特定的特殊字符，不能包含空格、句号、感叹号，也不能包括字符 @、&、\$和#。名字最大长度为255字符。

在接下来的练习中，我们将要说明如何在 VBA 程序设计中使用变量。你将创建一个过程，其功能是提示用户输入名字，接着在消息框中显示出来。具体步骤如下：

1) 创建新的名为 KnowYourName 的子程序。

2) 输入如下代码：

```
Dim sName As String
```

```
sName = InputBox("Enter your name: ")
```

```
MsgBox "Hi" & sName
```

完成的过程应当与下面的代码相符：

```
Public Sub KnowYourName ()
```

```
    Dim sName As String
```

```
    sName = InputBox("Enter your name: ")
```

```
    MsgBox "Hi" & sName
```

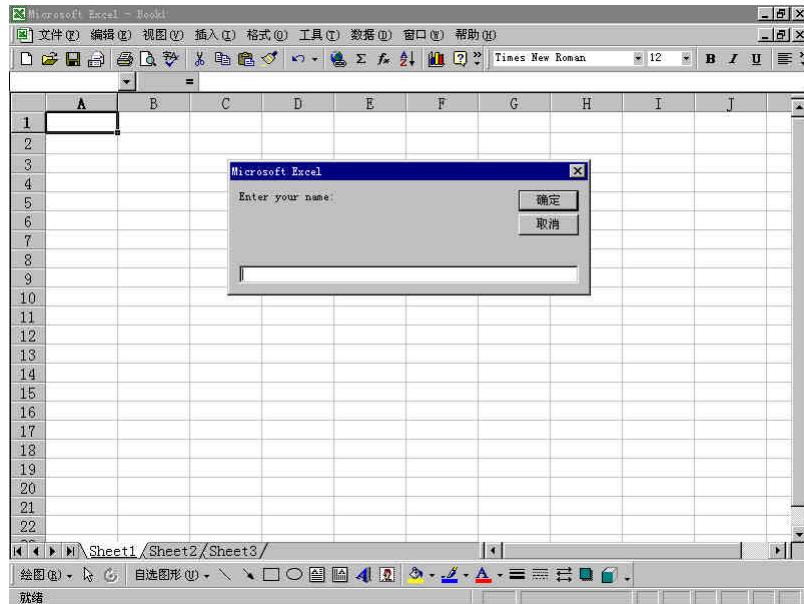
```
End Sub
```

本过程中缩进和空行并不是必须的。之所以强烈推荐缩进和空行，是为了提高代码的可读性。

3) 将鼠标放置到过程中的任何地方，按下 F5 键运行过程，会显示一个输入框，如图 4-7 所示。

图4-7 输入框将用户的输入

保存到变量sName中



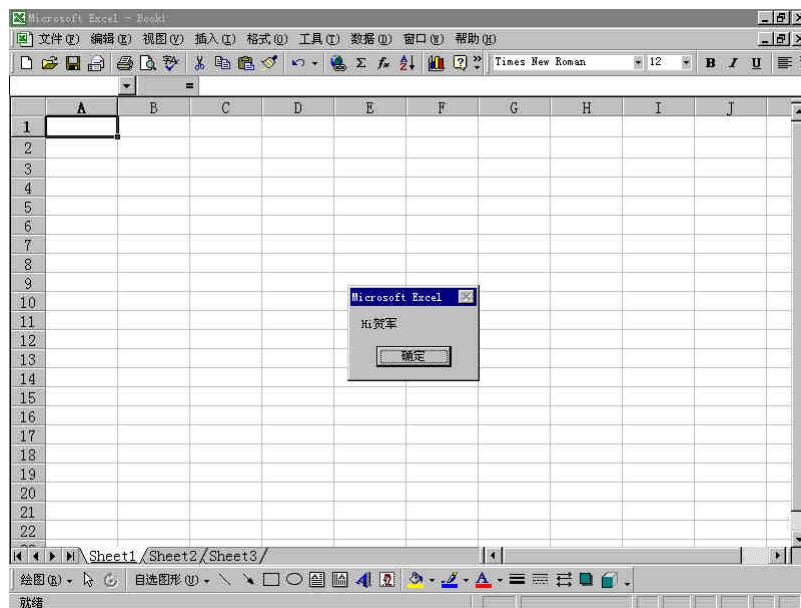
4) 输入你的名字并按回车键，会显示一个消息框，其中包括你刚输入的名字，如图 4-8 所示。

5) 单击“确定”按钮，返回到过程中。

在 Dim 语句中不必提供数据类型。如果没有提供数据类型，变量将被指定为 Variant 类型，因为 VBA 中默认的数据类型是 Variant。你知道这一点后，最初的反应也许是觉得应该不用自

己决定数据类型，而将一切抛给 VBA。这种观点是完全错误的，你必须决定选择何种数据类型。必须指定数据类型的第一个原因是，Variant数据类型占用的存储空间较大，从程序清单4-1中可以看出，即使没有给 Variant类型的变量赋值，它也要占用16个字节或者22个字节。第二个原因是，Variant数据类型将影响应用程序的性能。VBA必须辨认 Variant类型的变量中存储了何种类型的数据。下面的练习说明为什么不希望全都使用 Variant类型：

图4-8 消息框将变量sName
的值作为显示内容的
一部分



1) 创建一个新的名为 VariableExample的过程。

2) 为过程输入如下代码：

```
Dim StartTime
Dim EndTime
Dim i
Dim j

StartTime = Now()
For i = 1 To 5000000
    j = i + 1
Next i

EndTime = Now()
MsgBox "Start Time: " & StartTime & vbCrLf & "end Time: " & EndTime
```

因为没有为Dim语句中的变量提供数据类型，它们被自动指定为 Variant类型。

3) 按下F5键运行过程，其运行事件应该需要几秒钟。运行时间至少应该有4秒钟，由于你使用的计算机可能速度更快，本例中的这个过程可能运行时间太短，而不能说明问题。如果在你的计算机上这个过程的运行时间不到4秒钟，可将数字5 000 000增加到10 000 000。注意过程的运行时间。

4) 单击“确定”按钮，清除消息框。

5) 将过程代码修改为如下所示：

```

Dim StartTime As Date
Dim EndTime As Date
Dim i As Long
Dim j As Long
StartTime = Now()
For i = 1 To 5000000
    j = i + 1
Next i

EndTime = Now()

```

MsgBox "Start Time: " & StartTime & vbCrLf & "End Time: " & EndTime

6) 按下F5键再次运行过程，这次的运行时间大约是第一次运行时间的一半。

7) 单击“确定”按钮，清除消息框。

4.4.3 变量命名的惯例

在第3学时中，我们介绍了给控件命名的惯例。许多开发者对变量的命名也使用一套惯例。

表4-2列出了用于为变量命名的推荐前缀。

表4-2 变量的命名前缀

数据类型	短 前 缀	长 前 缀
Array	a	ary
Boolean	f	bin
Byte	b	bit
Currency	c	cur
Date/Time	dt	dtm 或者 dat
Double	d	dbl
Integer	i	int
Long	l	lng
Object	o	obj
Single		sng
String	s	str
Variant	v	var

程序清单4-3对多种变量声明方式进行了说明。注意对string变量的声明。如果你知道字符串中可能包含的最大字符数，那么可以在变量声明时加以指明。

在本表中，可以看到以单引号开始的行，这些是注释行。注释是用来对代码进行说明，当过程运行时Excel忽略注释。

程序清单4-3 标量声明

- 1: The Following line creates a variant
- 2: Dim StudentID
- 3: Dim iNumberOfStudents as Integer
- 4: Dim dTestDate as Date
- 5: The following line creates a variable length string
- 6: Dim sLastName as String
- 7: The next line creates a 2 char. fixed length string
- 8: Dim sState as String * 2

注意本表中变量的命名方式，它们都使用了短前缀。描述变量类型的前缀用小写方式，而变量名的其余部分则分别使用大小写字母，这是变量命名通常采用的惯例。

4.4.4 使用数组

新术语 如果你曾经使用过其他编程语言，可能对数组已经比较熟悉了。数组是具有相同数据类型并且共享同一个名字的一组变量的集合。数组中的元素通过索引数字加以区分。定义数组的语法如下：

```
Dim array_name (n) As type
```

其中n是数组中的元素的数目。

例如，如果要创建保存10个学生名字的数组，可以使用如下语句：

```
Dim sStudents(9) as string
```

注意，括号中的数字是9而不是10。这是因为在默认情况下，第一个索引数字是0。数组在处理相似信息时非常有用。假设需要处理15门考试成绩，可以创建15个独立的变量，这意味着需要15个独立的Dim语句。也可以创建一个数组来保存考试成绩，具体语句如下：

```
Dim iTestScores(14) As Integer
```

大多数情况下，可以使用像上面例子中那样的一维数组。然而，VBA也支持多维数组。例如，可以认为二维数组和工作表或者表格具有相似的结构。要创建 4×4 的数组，可以使用如下语句：

```
Dim iTable(3,3) As Integer
```

新术语 声明数组时的另一种选择是不给定大小。这样，当程序开始运行时，就具有定义数组大小的灵活性。例如，你的应用程序让用户创建一张表格，可以提示用户确定要创建的表格的行和列的数目。通过创建动态数组就可以做到这样，甚至还可以使用户在创建完表格（实际上是数组）后对行或列进行删除。

如果声明数组时没有给定大小，就成为动态数组。声明动态数组的语法如下：

```
Dim dyn_array() As type
```

对数组进行声明后，可以在运行时用ReDim语句指定数组的大小：

```
ReDim dyn_array(array_size)
```

参数array_size代表数组的新大小。如果要保留数组的数值，请在ReDim语句后使用保留字Preserve，具体语法如下：

```
ReDim Preserve dyn_array(array_size)
```

4.4.5 变量赋值

声明变量后就可以给变量赋值。程序清单4-4列出了一系列给变量赋值的语句。请注意为数值元素赋值时索引数字的使用。

程序清单4-4 给变量赋值

```
1: Dim iNumber As Integer  
2: Dim iTestScore () As Integer  
3: Dim i As Integer  
4:  
5: iNumber = InputBox("Enter the number of students: ")
```

```
6: ReDim Preserve iTestScore(iNumber)
7:
8: For i = 1 To iNumber
9:   iTestScore(i) = InputBox("Enter test score " & i)
10: Next
```

4.5 使用常量

现在，你已经知道变量的作用是非静态信息的存储容器。当需要存储静态信息时，可以创建常量。使用常量有两个原因，其一是常量可以存放数值供程序运行时多次引用而不改变，但是这些数据可能在将来发生变化。一个很好的例子是税率。另一个原因是使用常量可以增加程序的可读性。TAXRATE比.08167要好理解得多。

要声明常量并设定常量的值，需要使用 Const语句。常量声明后，不能对它赋一个新的数值。例如，假设需要声明一个常量来保存销售税率，可以使用如下语句：

```
Const SALESTAX As Long = .06231
```

因为你已经知道常量的值，所以在 Const语句中可以指定数据类型。常量可以声明为如下类型：Boolean、Byte、Integer、Long、Currency、Single、Double、Date、String或者Variant。常用的常量的命名惯例是全部字母都用大写，这样就容易区分代码中的变量和常量。

4.6 作用域

新术语 到目前为止，已经学习了如何定义变量和常量，但是还不知道在何处进行定义。可以在两个位置定义变量和常量：可以在过程中进行定义，也可以在模块顶部一个名为“通用声明”的区域进行定义。定义变量的位置就决定了变量的作用域。当在过程内创建变量时，该变量只能用于这个过程中，其他的过程都不能使用这个变量以及它的数值。这种情况下，变量成为程序级变量或者局部变量，因为相对于定义这个变量的过程而言，变量是局部的。

新术语 如果希望设置变量或者常量的值，然后在模块的其他过程中进行使用的话，应该怎么办呢？这种情况下，变量和常量的作用域发生了改变。如果在模块的“通用声明”区域中定义常量或者变量，那么定义在该模块中的所有过程都可以使用。这种变量称为模块级变量。

作用域的概念也可用于过程。

新术语 还有一种等级的作用域称为公共级。公共级变量可以用在应用程序的任何过程中，不论过程和变量是否定义在同一个模块中。这就使得公共级变量在使用上十分灵活，但是同时这也意味着，当运行应用程序时，它们一直保留在内存中，这样就占用了系统资源。要创建公共级变量，可以使用 Public语句。具体语法如下：

```
Public variablename As datatype
```

要创建要创建公共级常量，具体语法如下：

```
Public Const CONSTANTNAME datatype = value
```

公共变量和常量必须定义在位于模块顶部的通用声明区域中。

4.7 学时小结

本学时的重点是变量和常量。在本学时中，你学会了如何创建变量来保存非静态的数据。作为变量讨论的一部分，我们介绍了选择合适的数据类型的重要性，你也学会了如何创建常量来保存静态数据。本学时的最后一部分介绍了定义变量的位置与变量作用域之间的关系。

在本学时的开始部分，你学会了如何创建过程。在下一学时中，将学习如何从其他过程中调用过程以及如何创建和使用函数。

4.8 专家答疑

问题：如果希望在多个位置使用变量的数值，应该在何处定义变量？

解答：必须在模块的“通用声明”中定义。如何定义取决于将要使用这个变量的过程的位置。如果所有过程都位于同一模块中，那么可以用 Dim语句进行定义。如果它们位于多个模块中，那么应当使用 Public语句。

问题：模块是否独立于工作簿？

解答：不。实际上模块是工作簿的一部分。当保存工作簿时，对模块所做的改变也会保存下来。

问题：为什么不应将所有变量定义为 Variant类型？

解答：这是因为 Variant类型本身的特点，Variant类型占用更多的内存，并且影响应用程序的性能。

4.9 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

4.9.1 思考题

- 1) 作用域的三种级别是什么？
- 2) 如果需要定义一个变量保存大于0小于100的整数，应当使用什么数据类型？
- 3) 过程名、变量名和常量名最多可以包含多少个字符？
- 4) 判断题：过程名的首字符可以是数字。
- 5) 公共级变量定义在什么位置？
- 6) 判断题：变量只能定义在过程中。
- 7) 执行过程的功能键是什么？

4.9.2 练习题

创建新的名为 VarAndConst的过程，创建名为 sTest的数据类型为 String的变量。再创建一个名为 iNumber 的数据类型为 Integer的常量，将其值设置为 2。设置 sTest的值为“ This is a test.”。将如下代码行添加到过程中以显示 sTest和iNumber的值：

```
MsgBox "sTest's value is:" & sTest
```

```
MsgBox "iNumber's value is:" & iNumber
```

运行本过程。

第5学时 用户输入

在前一学时中，一些用来说明不同主题的过程已经使用了 MsgBox和InputBox语句。本学时中，将学习这些语句，并且掌握如何在 VBA代码中使用它们。在第1学时中，我们讨论了录制宏的某些局限性。其中一个局限性是不能向用户进行信息提示。在本学时中，将学习如何对用户进行信息提示，以及如何获取用户对这些提示的响应。

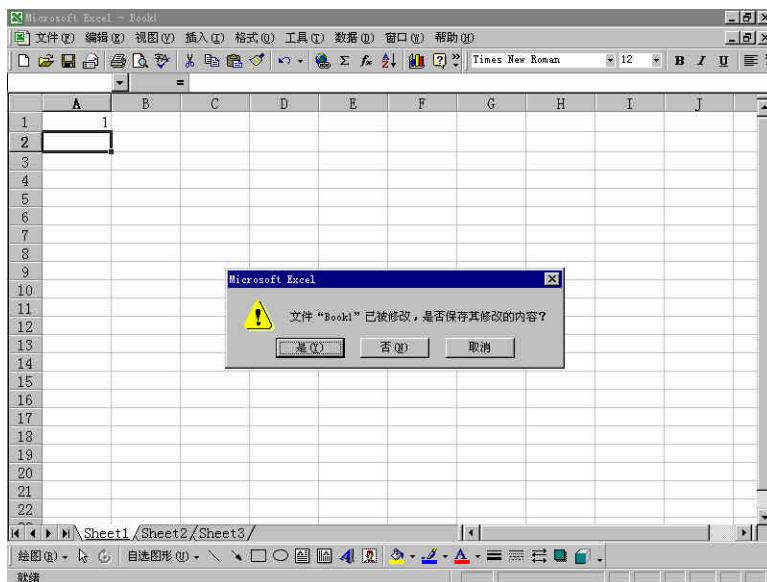
本学时的重点包括：

- 使用MsgBox函数
- 探索InputBox函数
- 使用InputBox方法，并且理解它与InputBox函数的区别
- 对命名参数的讨论
- 如何使用字符串

5.1 MsgBox函数

图5-1显示了当没有保存工作簿而试图退出时的情况。

图5-1 这是应用程序产生的众多的消息框之一



你也许一直认为显示消息框需要长时间的程序设计，这种观点是完全错误的。要创建消息框，可用如下VBA代码：

```
MsgBox "Do you wish to save the changes to '" &_
ThisWorkbook.Name & "' ", vbYesNoCancel + vbExclamation
```

代码中的ThisWorkbook.Name用来获取当前工作簿的名字。

MsgBox函数可用于在对话框中显示信息。一旦显示出来，消息框将一直保持，

直至用户单击某个按钮为止。根据用户单击的按钮，会返回不同的整数值。 MsgBox函数的语法如下：

```
MsgBox(prompt [,buttons][,title][,helpfile,context])
```

这个函数必须具有的参数只有 prompt。prompt的值是将在对话框中作为信息显示的字符串。注意在给出的语法中括号的运用。MsgBox是一个函数，这意味着它将返回一个值。在前面给出的例子中并没有括号。如果去掉函数参数的括号，就是在告诉 VBA不需要函数的返回值。如果希望获得返回值，可使用和下面相似的代码：

```
Dim iResponse As Integer  
iResponse = MsgBox("Do you wish to save the changes to '" & _  
ThisWorkbook.Name & "' ", vbYesNoCancel + vbExclamation)
```

现在MsgBox语句被分成了两行。VBA中的续行号是空格加下划线。

如果没有为 buttons参数提供数值，默认设置下，VBA假设你只希望在创建的对话框中添加“确定”按钮。可选参数 buttons是一个非常有用的参数，它允许你控制：

- 消息框中包括的按钮的数目。
- 消息框中包括的按钮的类型。
- 消息框中显示的图标。
- 消息框的默认按钮。
- 消息框的特征。

表5-1列出了 buttons参数的可能的各种设置。在这张表格中，你会发现参数分成了几组。第一组设置对话框显示的按钮数目和类型，第二组选择图标的风格，第三组设置默认按钮，第四组设置对话框的特征。当添加数值创建最终的 buttons参数时，从各组数值中只能选择一个。

表5-1 可用的MsgBox函数的buttons参数值

参数组	常量	数值	描述
第一组	vbOKOnly	0	只显示“确定”按钮（默认设置）
	vbOKCancel	1	显示“确定”和“取消”按钮
	vbAbortRetryIgnore	2	显示“放弃”、“重试”和“忽略”按钮
	vbYesNoCancel	3	显示“是”、“否”和“取消”按钮
	vbYesNo	4	显示“是”和“否”按钮
	vbRetryCancel	5	显示“重试”和“取消”按钮
第二组	vbCritical	16	显示危险消息图标
	vbQuestion	32	显示警告询问图标
	vbExclamation	48	显示警告消息图标
	vbInformation	64	显示信息消息图标
第三组	vbDefaultButton1	0	第一个按钮为默认按钮
	vbDefaultButton2	256	第二个按钮为默认按钮
	vbDefaultButton3	512	第三个按钮为默认按钮
	vbDefaultButton4	768	第四个按钮为默认按钮
第四组	vbApplication Modal	0	应用程序模式；用户必须对消息框作出响应才能继续使用当前的应用程序
	vbSystemModal	4096	系统模式；所有应用程序都被挂起直至用户对消息框作出响应
附加选项	vbMsgBoxHelpButton	16384	在消息框上添加“帮助”按钮
	vbMsgBoxSetForeground	65536	将消息框设置为前景窗口
	vbMsgBoxRight	524288	显示右对齐的消息框
	vbMsgBoxRtlReading	1048576	指定在希伯莱和阿拉伯系统中显示的文本应当从右向左阅读

要查看MsgBox函数和其他Visual Basic内置常数的列表，可使用Object Browser。寻找以VB开始的类，也可查看Constant类，也可找到Excel的内置常数。

可选的标题常数可以设置对话框标题栏显示的字符串表达式。如果不提供 title常数的值，Microsoft Excel将显示在标题栏中。

helpfile和context都是可选常数，当你为应用程序创建了自己的帮助文件时可以使用它们。

MsgBox是一个函数，这就意味着它将返回一个数值。表 5-2列出了MsgBox函数可能返回的数值。返回值完全取决于用户所选择的按钮。

表 5-2 MsgBox函数的返回值

查看MsgBox函数的返回值，你认为需要创建哪种类型的变量来保存返回值？定义用来保存MsgBox函数的返回值的变量时，可以使用的最好数据类型是 Integer。在接下来的练习中，将创建一个具有多个按钮的消息框，接着用另一个消息框来显示用户选择了哪个按钮，具体步骤如下：

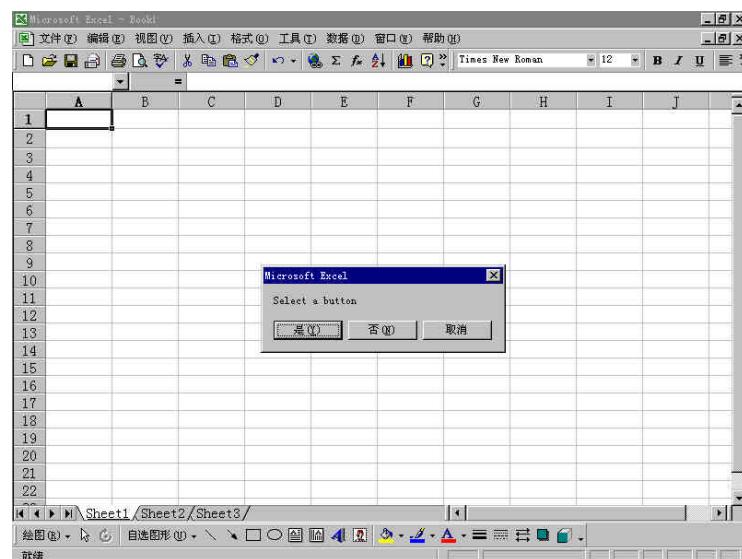
- 1) 打开Visual Basic编辑器。
- 2) 在工程资源管理器中的“ ThisWorkbook ”上单击鼠标右键。
- 3) 选择“插入”、“模块”。
- 4) 选择“插入”、“过程”。
- 5) 输入“ MBExercise ”作为过程名并按回车键。
- 6) 输入如下代码作为MBExercise过程：

```
Dim iResult As Integer
```

```
iResult = MsgBox("Select a button", vbYesNoCancel)
MsgBox iResult
```

- 7) 将鼠标指向该过程，按下F5键运行这个过程，显示如图 5-2所示的对话框。

图5-2 消息框中包括三个由 buttons常数指定的按钮



8) 单击“是”按钮，会显示另一个包括数字“6”的消息框。参考表5-2可以知道6是单击按钮“是”的结果。单击“确定”按钮退出该消息框。

9) 按F5键再次运行该过程。这次单击“否”按钮对第一个消息框进行响应，第二个消息框显示数字“7”。单击“确定”按钮退出该消息框。

10) 按F5键再次运行该过程。这次单击“取消”按钮对第一个消息框进行响应，第二个消息框显示数字“2”。单击“确定”按钮退出该消息框。

现在，你知道了如何用变量来显示消息框的返回值。在第6学时中，将学习如何在VBA代码中使用这种返回值。

5.2 InputBox函数

当只需做出“是”、“否”、“确定”、“取消”等简单的回答时，MsgBox函数非常好用。但是，如果需要其他类型的输入，例如数字和文本时，就需要使用InputBox函数。InputBox函数显示一个对话框，并且提供便于用户输入的文本说明。InputBox函数的语法如下：

```
InputBox(prompt[, title][, default][, xpos][, ypos][, helpfile, context])
```

InputBox函数必须具有的参数是prompt。和MsgBox函数的prompt参数一样，prompt参数的值是将显示在对话框中的字符串。

可选参数title是显示在对话框标题栏中的字符串表达式。和在MsgBox函数中一样，如果没有给title参数设定数值，标题栏将显示“Microsoft Excel”。

为了节省用户的时间，你也许希望给可选参数default设定数值。这个参数的值可设为一个字符串表达式，当用户没有输入任何数据时，其值将显示在文本框中作为默认响应。如果省略了这个参数，文本框将显示为空白。如果要求用户输入邮政编码，而且大部分顾客的邮政编码都相同，那么应当提供一个邮政编码作为默认输入。

xpos和ypos也是可选参数，这两个参数决定输入框在屏幕上的显示位置。

helpfile和context这两个可选参数只有在为应用程序创建了帮助文件时才有用。

InputBox函数的返回值是用户在对话框上的文本框中输入的信息。要练习使用InputBox函数，请完成如下步骤：

1) 在当前模块中插入新的名为IBExercise的过程。

2) 为新的过程输入如下代码：

```
Dim iResult As Integer
```

```
iResult = InputBox("Please enter your favorite number: ")
```

```
MsgBox iResult
```

```
ActiveCell.Value = iResult
```

这些代码提示用户输入一个数字，接着在一个消息框中显示这个数字，同时它也将这个数字显示在工作表上活动的单元格中。

3) 将插入点指向该过程，按下F5键运行这个过程，显示出来的输入框如图5-3所示。

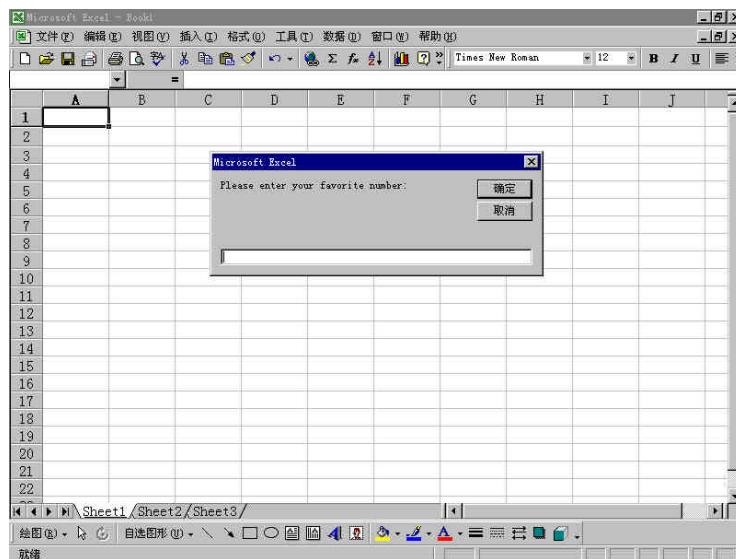
4) 输入一个数字并按回车键，显示出包含刚才输入数字的消息框。

5) 单击“确定”按钮，退出消息框。

6) 返回工作簿，应当在工作簿活动的单元格中看到输入的数字。

图5-3 在输入框中可输入

数字



如果你查看在线帮助文档，就会知道 InputBox函数返回一个字符串。实际上并不完全就是这样。在Visual Basic更新的版本，包括实现VBA的版本中，从输入数据的特征上看，返回值的行为更像是Variant类型。这正是即使iResult变量被设定为Integer数据类型时，过程仍然能够正常工作的原因。关于InputBox函数，需要知道的另一点是，如果用户单击了“取消”按钮会返回什么样的数值。如果用户单击“取消”按钮，InputBox函数将返回一个零长度的字符串”。最后，用户似乎总是想知道如何控制输入框的大小。实际上无法控制输入框的大小，这正是“所见即所得”的一个例子。

5.3 InputBox方法

Excel支持另一种获取用户输入的方式——InputBox方法。看上去InputBox方法和InputBox函数是一样的，实际上InputBox方法具有一些细微的但是非常有用的区别。首先需要了解的是InputBox方法的语法：

```
Application.InputBox(prompt,[Title],[Default],[Left],[Top],[HelpFile],[HelpContextId],[Type])
```

InputBox方法的语法和InputBox函数相似。注意语句以Application开始。这个方法属于Excel，因而它也属于应用程序。

大部分参数看上去都是相同的。细微的区别是Left和Top参数，它们与xpos和ypos参数等价。

应当注意的是InputBox方法的最后一个参数——Type。表5-3列出了参数Type可以使用的数值。通过可选参数Type可以指定返回值的数据类型。表5-3列出了参数Type可以使用的数值。

看过这张表后，你也许希望知道为什么数字之间有间隔，例如从4跳到8、从16跳到64，这是因为可以将这些可用的数值相加后使用。例如，如果你希望接收数字和文本，那么可将Type参数设置为1+2。如果没有设置Type参数的值，InputBox方法将返回文本。为了说明InputBox方法的优点，请完成如下步骤：

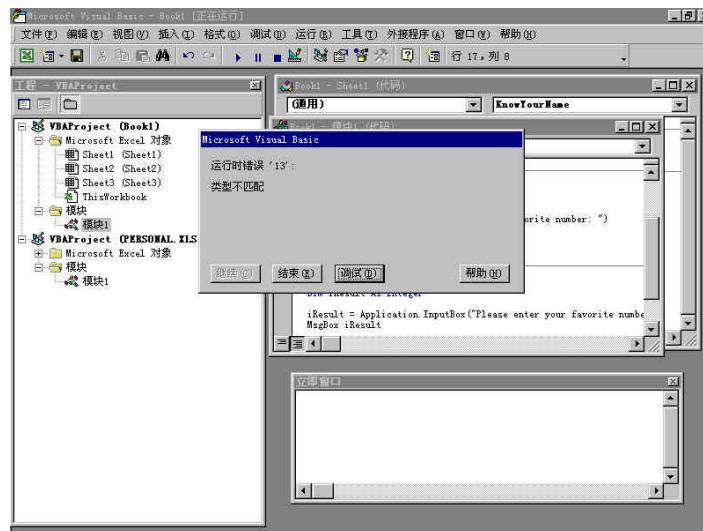
表5-3 参数Type可以使用的数值

值	期望的返回值
0	一个公式
1	一个数字
2	文本(字符串)
4	一个逻辑值，例如true或False
8	一个单元格引用
16	一个错误值
64	一个值的数组

1) 运行IBExercise过程。

2) 在输入框中输入字母“a”作为响应，按下回车键，查看运行的结果。因为输入的是字母而不是数字，过程运行会出错，如图5-4所示。

图5-4 从这个消息框可以知道应用程序出了错。显然你不希望用户看到这样的消息框



3) 单击“结束”按钮，退出消息框。

4) 创建一个新的名为IBMethod的过程，这个新的过程将使用InputBox方法。

5) 为新的过程输入如下代码：

```
Dim iResult As Integer
```

```
iResult = Application.InputBox_
    ("Please enter your favorite number: ", , , , , 1)
MsgBox iResult
ActiveCell.Value = iResult
```

在Application.InputBox语句中使用的逗号相当于一个占位符，用以取代没有输入数值的参数。Type是InputBox方法语法中最后一个参数。将 Type参数设置为1，这意味着只接收数字。注意本过程中这一行代码与IBExercise中的代码行的唯一区别是Application.InputBox语句的使用。

6) 将插入点指向过程IBMethod，按下F5键运行过程，显示出一个输入框。

7) 输入字母“A”并按回车键，此时过程不会出错，而是显示一条解释所出问题的消息，如图5-5所示。

图5-5 用户会注意到他的输入是无效的



8) 按回车键退出该消息框。

9) 输入“7”并按回车键，显示一个消息框，其中有一个7。单击“确定”按钮退出消息框。

现在，可以看到InputBox方法的一个优点是内置的出错处理。InputBox方法和InputBox函数的另一个区别是，当用户单击“取消”按钮时的返回值。如果使用InputBox函数而用户选择了“取消”按钮，则返回一个零字节的字符串。如果使用InputBox方法，则返回False。

5.4 命名参数

在前一练习中，输入过如下语句：

```
iResult = Application.InputBox_("Please enter your favorite number:", , , , , 1)
```

如果少输入一个逗号，当执行这条语句时就有可能出错。按照定义的位置传递参数具有局限性。为了弥补这一缺陷，VBA支持命名参数。使用命名参数可以按照任意次序传递参数。例如，通过使用命名参数，可以将前面的语句替换为：

```
iResult = Application.InputBox("Please enter your favorite number:", Type:=1)
```

要使用命名参数，按照语法输入参数名，接着输入一个冒号和等号，在等号后面输入参数值。当输入一条语句时，语句的语法会显示在QuickInfo框中。通过在线帮助也可以找到语句的语法。

如果没有显示快速信息，可选择“工具”、“选项”。选择“选项”对话框中的“编辑器”选项卡，选中“自动显示快速信息”复选框。

5.5 字符串连接

如果需要合并字符串，那么可以使用字符串连接符&。程序清单5-1说明了字符串连接符的使用方法。

程序清单5-1 字符串连接

```
1: Sub stringstuff ()  
2:     Dim sName As String  
3:     Dim sLongText As String  
4:  
5:     sName = InputBox("Please enter your name: ")  
6:  
7:     'The following line of code combines Hello and the value  
8:     'of sName so that it is displayed as a single string  
9:     'in a message box.  
10:    MsgBox "Hello" & sName  
11:  
12:    sLongText = "This is an example of using string"  
13:    sLongText = sLongText & "concatenation to combine long"  
14:    sLongText = sLongText & "strings." & vbCrLf  
15:    sLongText = sLongText & "The vbCrLf constant allows you"  
16:    sLongText = sLongText & "to add line breaks to your strings."  
17:    MsgBox sLongText  
18: End Sub
```

图5-6显示了代码行MsgBox sLongText显示字符串的结果。通过使用字符串连接符，可以

连接文本字符串和变量、常量中的字符串等。

图5-6 字符串连接符是建立消息框的一个非常有用工具



5.6 学时小结

在本学时中，学习了多种向用户显示提示信息的方法。对于只需获取简单的诸如“是”、“否”、“确定”、“取消”等响应的情况，使用消息框就足够了。输入框允许你的用户在对话框（就像界面一样）中输入响应信息。在下一学时中，将学习检测从消息框和输入框中获取的结果的工具。根据用户的响应，你的代码可以选择合适的操作。

5.7 专家答疑

问题：如何选择使用InputBox函数还是InputBox方法？

解答：InputBox方法具有理想的内置的出错处理特征，通常情况下选择 InputBox方法更好。

问题：输入框的返回值有什么用？

解答：可以在消息框中使用返回值，或者在单元格中进行显示，或者将返回值用于计算，这些只是返回值的众多用途中的几种。

问题：为什么要使用消息框？

解答：因为消息框提供了用户非常熟悉的界面，通过它可以和用户进行交流。

5.8 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容。答案请参考附录 A。

5.8.1 思考题

- 1) 字符串连接符是什么？
- 2) 消息框返回值的数据类型是什么？
- 3) 输入框返回值的数据类型是什么？
- 4) 通过buttons参数可以对消息框进行什么样的设置？
- 5) 判断题：在VBA中，参数只能按位置顺序进行传递。

5.8.2 练习题

创建一个名为YourInfo的过程。这个过程功能如下：

- 1) 显示三个输入框。第一个输入框提示用户输入名字，第二个输入框提示用户输入所居住的城市名，第三个提示用户输入年龄。
- 2) 在一个消息框中显示前三个输入框的结果。

第6学时 条件逻辑

在前一学时中，学习了如何提示用户进行响应并且将用户的响应保存在变量中。这些变量可以用来干什么呢？你将用这些响应来进行判断。你需要做的是如何处理用户对你的提示可能做出的各种响应。通过条件逻辑可以做到这一点，而这正是本学时的重点。

本学时的重点包括：

- 如何控制应用程序的流程
- 使用If语句
- 使用Select Case语句
- 怎样显示Excel内置的对话框

6.1 控制应用程序的流程

新术语 当应用程序显示一个包括“是”和“否”按钮的消息框时，需要决定：如果用户选择了“是”，应用程序应当执行什么操作；如果用户选择了“否”又应当如何。换句话说，应当根据用户的选择执行不同的语句。通过条件语句可以实现这种功能。条件语句可以判断一个条件是True还是False，并且根据判断的结果执行一条或者多条语句。你将发现可能会在你的应用程序中频繁地使用条件语句。

新术语 在条件语句中，你将使用条件逻辑来控制程序流程。条件逻辑可以根据变量值、用户的响应、函数计算或者设置的属性选择不同的程序路径。当使用条件逻辑时，需要先创建一个检测判断，在根据检测的结果让程序执行不同的操作。

新术语 例如，如果过程显示一个输入框，并提示用户输入他的工作区域，通过条件逻辑可以实现：如果用户工作在南部地区，程序执行一组语句；如果用户工作在北部地区，程序会执行另一组语句。另一个例子是，如果你有一张包括所有雇员创造的销售额的工作表，可以创建一个过程，根据雇员在公司工作的时间计算他们的佣金。如果一个雇员的工作时间少于两年，则他可以得到2%的佣金；如果雇员的工作时间大于两年，那么他可以得到4%的佣金。要进行这类判断，需要使用比较运算符。用比较运算符进行检测的结果或者为True（意味着满足测试条件）或者为False（意味着不满足检测条件）。通过对某些Excel函数（例如If函数）的使用，你也许已经对比较运算符很熟悉了。表6-1列出了可用的比较运算符。

表6-1 比较运算符

比较运算符	意 义
=	等于
<>	不等于
>	大于
>=	大于或等于
<	小于
<=	小于等于

有时，需要检测某个数值是否满足多重条件。例如，你也许需要检测一个雇员的工作时间是否超过5年，并且是否是管理人员，以计算他的休假时间。要做到这样，可以用逻辑运算符来合并需要检测的条件。表6-2列出了VBA中的逻辑运算符。

表6-2 逻辑运算符

逻辑运算符	意 义
And	如果两个条件为真，则结果为真
Or	如果两个条件中有一个为真，则结果为真
Not	如果条件表达式为假，则结果为真；如果条件表达式为真，则结果为假
Xor	如果一个且仅有一个条件为真，则结果为真；如果两个条件均为真，或均为假，则结果为假

6.2 If语句

将学习的第一条条件语句是 If...Then...Else...语句。这条语句的语法如下：

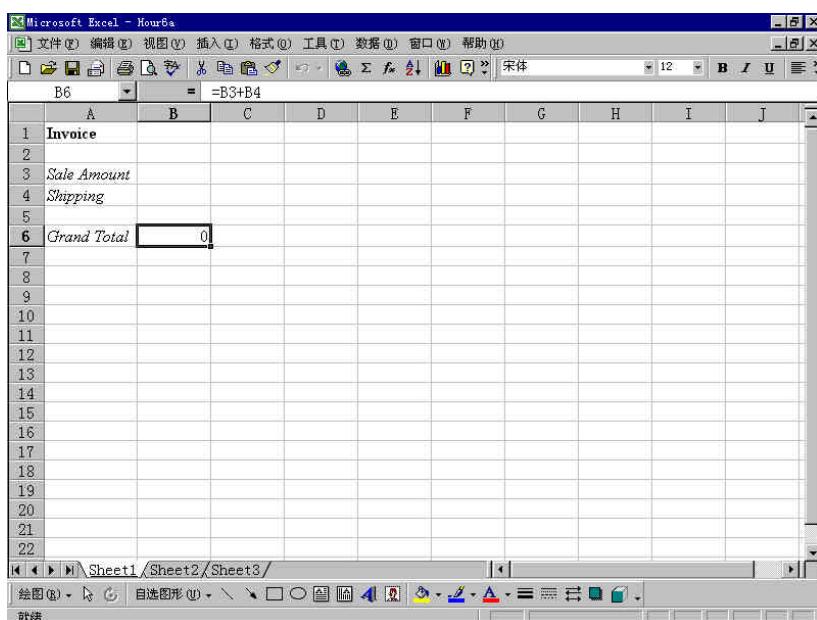
```
If condition Then
[statements]
[Elseif condition-n Then
[elseifstatements]...
[Else
[elsestatements]]
End If
```

condition是语法中必须的一部分，可在此处输入要检测的条件。如果条件检测的结果是 True，将执行在 Then之后的语句。

如果希望在同一条 If语句中检测第二个条件，可以向 If语句中添加一条或者多条 ElseIf从句。VBA首先检测If后的条件，如果检测的结果是 False，VBA将检测ElseIf从句后的条件，并重复检测，直至某个条件检测的结果为 True。如果所有条件检测的结果都是 False，VBA将执行End If语句，除非具有Else从句。如果If语句中所有条件检测的结果都是 False，那么将执行可选的Else从句后的语句。在下面的练习中，将学习使用 If语句：

- 1) 打开一个新的工作簿。
- 2) 创建一个如图6-1所示的工作表。

图6-1 本工作表是新的应
用程序的基础



- 3) 在单元格B6中输入公式B3+B4以计算总额。
- 4) 按下Alt+F11键打开Visual Basic编辑器。
- 5) 用鼠标右键单击工程资源管理器中的“ ThisWorkbook ”。
- 6) 选择“插入”、“模块”向工作簿中添加一个模块。
- 7) 创建一个新的名为Shipping的过程。
- 8) 为Shipping过程输入如下代码：

```
Dim iResponse As Integer
```

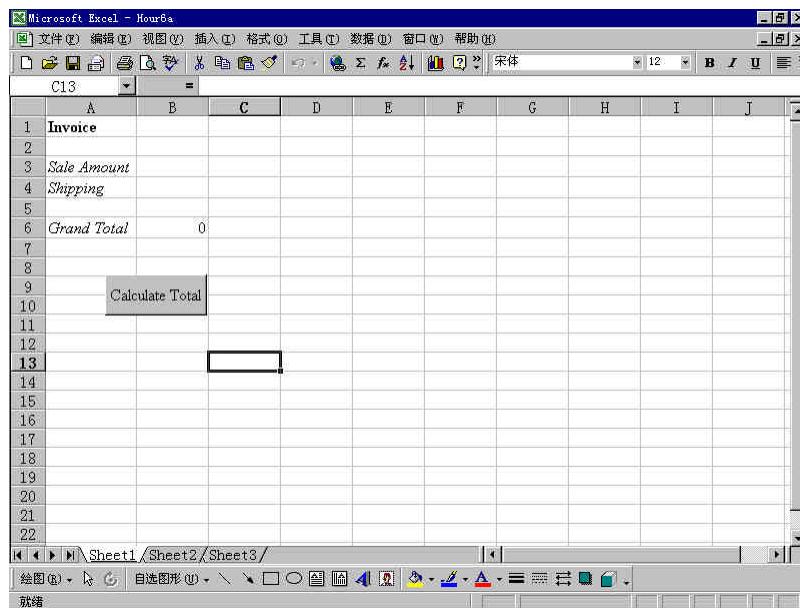
```
iResponse = MsgBox ("Does this sale need to be shipped? ", vbYesNo)
```

```
If iResponse = vbYes Then
    Range("B4").Value = 10
Else
    Range("B4").Value = 0
End If
```

- 9) 回到在第二步中创建工作表。

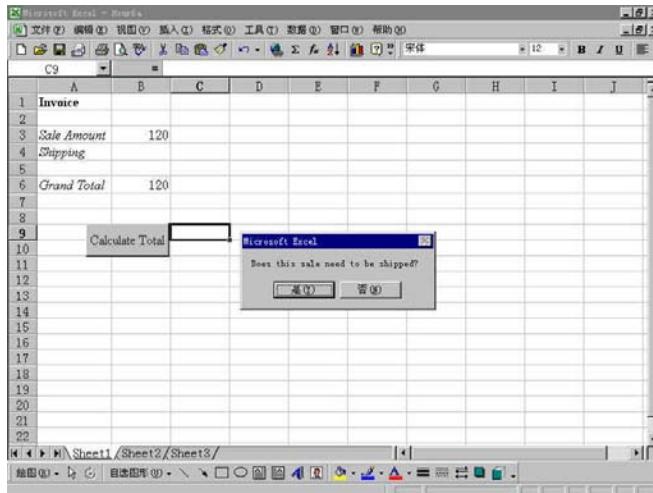
- 10) 向工作表添加一个命令按钮，显示“指定宏”对话框。
- 11) 选择“Shipping”并单击“确定”按钮，这样就将该过程指定给了命令按钮。
- 12) 突出显示新添加的按钮上的文本，按下Delete键删除文本。输入“Calculate Total”作为按钮的标题。
- 13) 在按钮之外单击鼠标左键使按钮不再被选中。完成后的工作表应当和图 6-2 中的工作表相似。

图6-2 完成后的工作表



- 14) 在单元格B3中输入“120”作为销售额。
- 15) 单击“Calculate Total”按钮，显示一个消息框询问这批货物是否需要运送，如图 6-3 所示。

图6-3 消息框询问是否需要
运送



16) 单击“是”按钮，单元格B4显示“10”，这是因为执行了过程中的If语句。

17) 再次单击“Calculate Total”按钮。

18) 单击“否”按钮对消息框作出响应，此时单元格B4显示为“0”。

这个简单的例子说明条件逻辑非常有用。在开始进一步学习之前，请花时间复习一下前面的练习。完成后的过程如程序清单6-1所示。

程序清单6-1 Shipping过程

```

1: Public Sub Shipping ()
2:   Dim iResponse As Integer
3:
4:   iResponse = MsgBox ("Does this sale need to be shipped? ", vbYesNo)
5:
6:   If iResponse = vbYes Then
7:     Range("B4").Value = 10
8:   Else
9:     Range("B4").Value = 0
10: End If
11:
12:End Sub

```

在这个过程中，首先需要声明一个变量：

Dim iResponse As Integer

这个变量在过程中用来保存消息框的响应，如下面的代码所示：

iResponse = MsgBox("Does this sale need to be shipped?",vbYesNo)

这条语句生成的消息框有两个按钮：“是”和“否”。对iResponse变量数值的检测由If语句完成。

```

If iResponse = vbYes Then
  Range ("B4").Value = 10
Else
  Range("B4").Value = 0
End If.

```

如果iResponse值为vbYes，意味着用户单击了“是”按钮，这样单元格B4就显示为

“10”；否则单元格B4显示“0”。

可以创建比前面例子复杂得多的If语句，甚至可以在If语句中嵌套If语句。下面的步骤将指导你创建更为复杂的If语句。在这个例子中，佣金比例需要考虑三个因素。第一个因素是该项目是否销售完毕。如果销售完毕，则所有销售人员获得最少的2%佣金；如果还正在销售，则每人得1%佣金。第二个因素是雇员在公司的工作年限。最后一个因素和雇员的工作部门有关，工作在物资部门的雇员获得附加的1%佣金。

当一条If语句位于另一条If语句中时，称为嵌套的If语句。

1) 向工作簿中添加一个新的工作表。

2) 创建如图6-4所示的工作表。

图6-4 本工作表用来计算佣

金比例

3) 打开Visual Basic编辑器。

4) 插入一个新的名为Commission的过程。

5) 输入如下过程代码：

```

Dim sngCommission As Single
sngCommission = 0.02

If Range("B2") = "No" Then
    sngCommission = 0.02

If Range("B3").Value >= 5 And Range("B3") < 10 Then
    sngCommission = sngCommission + 0.01
ElseIf Range("B3").Value >= 10 Then
    sngCommission = sngCommission + 0.02
End If

If Range("B1").Value="Furniture"Then
    sngCommission=sngCommission+0.01
End If

Else
    sngCommission=0.01
End If

Range("B5").Value=sngCommission

```

- 6) 回到工作表，添加一个命令按钮，并将 Commission 过程指定给该按钮。
- 7) 编辑按钮上的文本，使按钮标题变为“Calculate Comm”。
- 8) 在按钮之外单击鼠标左键，使按钮不再被选中。
- 9) 在单元格 B1 中输入“Furniture”，确认输入的文本完全正确。
- 10) 在单元格 B2 中输入“No”，确认输入的文本完全正确。
- 11) 在单元格 B3 中输入“10”。
- 12) 单击命令按钮。计算出来的佣金比例应该为 .05，因为该项目已经销售完毕，该雇员来自物资部门，而且已经工作了 10 年以上。

在上面的练习中使用的 If 语句要复杂得多，完成后的过程如程序清单 6-2 所示。

程序清单 6-2 过程 Commission

```
1: Public Sub Commission ()  
2:   Dim sngCommission As Single  
3:  
4:   If Range("B2") = "No" Then  
5:     sngCommission=0.02  
6:  
7:     If Range("B3").Value >=5 And Range("B3") < 10 Then  
8:       sngCommission = sngCommission + 0.01  
9:     ElseIf Range("B3").Value >= 10 Then  
10:      sngCommission = sngCommission + 0.02  
11:    End If  
12:  
13:    If Range("B1").Value = "Furniture" Then  
14:      sngCommission = sngCommission + 0.01  
15:    End If  
16:  
17:  Else  
18:    sngCommission = 0.01  
19:  End If  
20:  
21:  Range("B5").Value = sngCommission  
22:  
23: End Sub
```

和通常情况下一样，过程首先声明一个变量，在本例中变量名为 sngCommission。sngCommission 用来保存计算佣金比例的中间值。第一项检测是该项目是否已经销售完毕：

```
If Range("B2") = "No" Then  
  sngCommission = 0.02
```

如果该项目正在销售，则单元格 B2 的值为“Yes”，否则单元格 B2 的值为“No”。如果已经销售完毕，则至少佣金比例为 2%，否则佣金比例为 1%。

```
Else  
  sngCommission = 0.01  
End If
```

如果项目已经销售完毕，则还要进行以下两个检测：

```
If Range ("B3").Value >= 5 And Range("B3") < 10 Then  
  sngCommission = sngCommission + 0.01  
ElseIf Range("B3").Value >= 10 Then
```

```
sngCommission = sngCommission + 0.02  
End If  
If Range("B1").Value = "Furniture" Then  
    sngCommission=sngCommission+0.01  
End If
```

上面代码中的第一个 If 语句是用来检测该雇员在公司中工作了多长时间。第二个检测是决定该项目是否属于物资部门。在本过程中这两个 If 语句都是嵌套在最外层的 If 语句中。

当在创建上面的过程时，特别指出需要输入“ No ”。为了明白为什么，请完成如下步骤：

- 1) 回到具有 Calculate Comm 命令按钮的工作表，注意计算出来的佣金比例是 0.05。
- 2) 在单元格 B2 中输入“ no ”（全部小写）。
- 3) 单击 Calculate Comm 按钮，计算出来的佣金比例结果是 0.01，这是为什么？
- 4) 返回 Visual Basic 编辑器查看 Commission 过程，特别查看第一个 If 语句中的检测是：

```
If Range("B2") = "No"
```

现在你也许猜到了， If 语句是区分大小写的。为了确认这一点，对该过程做出如下改动，具体的改变用粗体字标出：

```
Public Sub Commission ()  
    Dim sngCommission As Single  
    sngCommission = 0.02  
  
    If UCASE (Range("B2")) = "NO" Then  
        sngCommission = 0.02  
  
        If Range ("B3").Value >= 5 And Range("B3") < 10 Then  
            sngCommission = sngCommission + 0.01  
        Elseif Range("B3").Value >= 10 Then  
            sngCommission = sngCommission + 0.02  
        End If  
  
        If UCASE(Range("B1").Value) = "FURNITURE" Then  
            sngCommission = sngCommission + 0.01  
        End If  
  
    Else  
        sngCommission = 0.01  
    End If  
  
    Range("B5").Value = sngCommission  
  
End Sub
```

5) 返回工作表并单击 Calculate Comm 命令按钮。这个过程的执行结果应当是正确的。

UCASE 函数将一个字符串中所有的字符都转换为大写。通过这样，可以避免 If 语句对大小写的敏感性。

6.3 Select Case 语句

请先查看程序清单 6-3 中的代码，其中包括用 If 语句检测数值以决定年级的代码。

程序清单 6-3 具有多重检测的 If 语句

```

1: If Range("A3") >= 90 Then
2:   MsgBox "You got an A on the test! "
3: ElseIf Range("A3") < 90 And Range("A3") >= 80 Then
4:   MsgBox "You got a B on the test. "
5: ElseIf Range("A3") < 80 And Range("A3") >= 70 Then
6:   MsgBox "You got a C on the test. "
7: ElseIf Range("A3") < 70 And Range("A3") >= 60 Then
8:   MsgBox "You got a D on the test. "
9: Else
10:  MsgBox "You failed. "
11: End If

```

这条语句中包括大量的 ElseIf 语句，而且阅读起来有一定困难，可以使用 Select Case 语句来替换 If 语句。Select Case 阅读起来比 If 语句要简单一些，并且更适合于需要设计多重选择的场合。程序清单 6-4 列出了转换为 Select Case 后的代码。

程序清单 6-4 使用 Select Case 语句的例子

```

1: Select Case Range("A3")
2:   Case Is >= 90
3:     MsgBox "You got an A on the test. "
4:   Case 80 To 89
5:     MsgBox "You got a B on the test. "
6:   Case 70 To 79
7:     MsgBox "You got a C on the test. "
8:   Case 60 To 69
9:     MsgBox "You got a D on the test. "
10:  Case Else
11:    MsgBox "You failed. "
12: End Select

```

从这张程序清单中可以看出，在多重选择的情况下，Select Case 更易于阅读。Select Case 语句的语法如下：

```

Select Case testexpression
[Case expressionlist-n
[statements-n]]...
[Case Else
[elsestatements]]
End Select

```

程序清单 6-5 显示了另一个 Select Case 语句的例子。当用逗号将两个数值分开时，实际上隐含实现了“Or”选择。例如，Select Case 语句将“Florida”、“Texas”解释为 Florida 或者 Texas。

程序清单 6-5 另一个使用 Select Case 语句的例子

```

1: 'Select Case to determine shipping cost.
2: Select Case State
3: Case "New York"
4:   cShipping = 5.00
5: Case "Georgia", "South Carolina", "Ohio"
6:   cShipping = 4.00
7: Case "Florida", "Texas"

```

```

8:   cShipping = 3.00
9: Case "Alabama", "Washington", "California", "Illinois"
10:  cShipping = 2.00
11: Case Else
12:  cShipping = 1.00
13: End Select

```

要使用比较运算符，需要使用 Is或者To关键字。Is关键字用来将希望进行比较的检测表达式和Is关键字后列出的表达式进行比较。而 To关键字定义一个范围内的数值。

6.4 使用内置Excel对话框

现在，你学会了如何控制应用程序的流程，根据用户对消息框所做出的响应，可以进行相应的操作。例如，如果要求用户回答他是否希望保存工作，你会希望显示一个“保存”对话框。Excel包括大约200个内置的对话框，通过使用VBA可以访问所有这些内置的对话框。显示内置对话框的语法如下：

```
Application.Dialogs(xlDialogConst).Show
```

要查看xlDialogConst数值的程序清单，可以打开对象浏览器。要找到需要的 xlDialogConst的名字，可以参看对话框的标题栏。如果希望显示“另存为”对话框，可以选择“文件”、“另存为”，可以看到对话框的标题栏显示为“另存为”。xlDialogConst的数值总是以xlDialog开始，并且接着就是对话框的名字，所以在本例中，将使用的是 xlDialogSaveAs。要显示内置对话框，可以按照如下步骤：

- 1) 向模块中插入一个新的名为 SaveNow的过程。

- 2) 为该过程输入如下代码：

```
Dim iResponse As Integer
```

```
iResponse = MsgBox("Do you wish to save your work? ", vbYesNo)
```

```
If iResponse = vbYes Then
```

```
    Application.Dialogs(xlDialogSaveAs).Show
```

```
End If
```

- 3) 返回到工作簿中包括Calculate Comm按钮的工作表。

- 4) 在工作表中添加另一个按钮，将SaveNow过程指定给这个按钮。

- 5) 将按钮的标题改为Save。

- 6) 在按钮之外单击鼠标左键，使该按钮不再被选中。

- 7) 单击Save按钮，显示一个询问是否需要保存工作的消息框。

- 8) 单击“确定”按钮，显示内置的“另存为”对话框，如图 6-5所示。

图6-5 所有Excel内置的对话框都是可用的



9) 单击“取消”按钮。

6.5 学时小结

在本学时中，你学会了控制应用程序流程的方法：If语句和Select Case语句。将这些语句和比较运算符配合使用，可以让应用程序做出各种决定。

本学时也说明了如何显示Excel内置的对话框。Excel提供了200多个内置对话框，既节省了时间，又可以向用户提供他们熟悉的界面。

6.6 专家答疑

问题：如果需要做出的决定有几种可能的答案，那么应当选择If语句还是Select Case语句？

解答：选择Select Case语句更为合适，因为该语句本身就是为多重选择而设计的。

问题：如果我需要一个“保存”、“打印”或者Excel中已经具有的其他类型的对话框，是否需要用VBA创建自定义对话框？

解答：不用，Excel具有的对话框都可以使用。

6.7 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

6.7.1 思考题

- 1) 两种基本的控制程序流程的语句是什么？
- 2) 判断题：If语句和Select Case语句区分大小写。
- 3) 用什么方法显示Excel内置的对话框？
- 4) 如何将字符串全部转化为大写？

6.7.2 练习题

首先，创建一个名为“ClickTest”的过程，要求这个过程显示一个消息框，提示用户“Do you wish to continue?”，并且具有“确定”和“取消”按钮。使用If语句检测用户单击了哪一个按钮，并显示一个消息框告诉用户选择了哪一个按钮。

接着，创建另一个名为“Discount”的过程，显示一个输入框，提示用户输入折扣的类别（可以是1、2、3或者4）。使用Select Case语句在一个消息框中显示各种折扣类别的折扣比例：第1类为5%，第2类为10%，第3类为15%，第4类为20%。

第7学时 循环代码

如果需要进行相同的操作 15次，你希望编写同样的代码行 15次吗？显然答案是否定的，用不着这样做。实际上，用循环语句可用轻而易举地实现：循环运行多次重复相同的代码。在本学时中，将学习 Do循环和For语句。

本学时的重点包括：

- 实现For语句
- 使用Do循环

7.1 For...Next语句

当编写代码时，你会发现经常需要多次重复进行某种操作。在需要重复执行一组语句的次数一定时，可以使用 For...Next循环。For...Next语句的语法如下：

```
For counter = start To end [Step step]  
[statements]  
[Exit For]  
[statements]  
Next [counter]
```

For...Next语句执行一定的次数，这取决于start和end参数的设置。counter是一个整数变量，每次循环增加1，除非设置了可选参数step，此时counter每次以step的数值变化。每次循环都要执行Next语句。当counter的值大于end的数值时，循环结束执行。可选的 Exit For语句通常放置在一条If语句或者Select Case语句中。为了练习使用 For...Next语句，请打开一个新的工作簿，将该工作簿保存为“Hour7”。按下Alt+F11键打开Visual Basic编辑器。现在完成如下步骤，你将看到一个简单的For...Next语句例子。

- 1) 在工程资源管理器中的“ThisWorkbook”上单击鼠标右键，并选择“插入”、“模块”。
- 2) 插入一个新的名为“BeepMe”的子程序。这是一个使计算机发出一定次数蜂鸣声的简单过程。

3) 为BeepMe过程输入如下代码：

```
Dim iCounter As Integer
```

```
For iCounter = 1 To 15  
    Beep  
Next
```

4) 按下F5键运行过程，应当听到很长的蜂鸣声。

如果希望练习使用 For语句，可以改变语句的执行时间，例如将参数 15改为25或者5来观察变化。如果查看For语句的语法，你会注意到一个可选从句 Step。这个从句可以控制For语句如何计数。例如，可以设定让 For语句每次计数为 5、10或者任何你希望的单位。可以通过将step设置为-1使计数每次递减。按照如下步骤创建一个递减计数的 For语句：

- 1) 插入一个新的名为Countdown的子程序，这个过程将显示一系列的消息框以说明 For语

句和可选的Step从句的用法。

2) 为Countdown过程输入如下代码：

```
Dim iCounter As Integer

For iCounter = 1 To 3
    MsgBox "CountUp: " & iCounter
Next

For iCounter = 3 To 1 Step -1
    MsgBox "CountDown: " & iCounter
Next
```

3) 运行该过程，将看到第一个消息框，如图 7-1所示。从消息框中可以看到，iCounter变量的当前值为1。

图7-1 观察本消息框的文本，可以看到计数器的数值是1



4) 单击消息框上的“确定”按钮，直到看到如图7-2所示的消息框，此时iCounter的数值为3。

图7-2 在每个消息框显示之前，
计数器不断递增



5) 单击“确定”按钮，注意计数器的值开始递减。

6) 继续单击“确定”按钮对消息框做出响应，直到过程运行完毕。

现在，你对For语句已经有了一定的了解，接下来将创建一个过程用 For语句来计算。如果将钱存放在利息为10%的银行，你可以得到多少钱。完成如下步骤：

1) 插入一个新的名为“HowMuchMoney”的过程。

2) 为该过程输入如下代码：

```
Dim iNumberOfYears As Integer
Dim cSavings As Currency
Dim iCounter As Integer

cSavings = InputBox("Enter amount you are placing in the account: ")
iNumberOfYears = InputBox("Enter number of years you are saving the money: ")

For iCounter = 1 To iNumberOfYears
    cSavings = cSavings * 1.1
Next

MsgBox "In" & iNumberOfYears & "years you'll have" &
    Format(cSavings, "0.00") & "dollars."
```

- 3) 运行 HowMuchMoney 过程。
- 4) 输入 1000 作为存钱的数额。
- 5) 输入 10 作为存钱的年数。此时显示一个消息框，消息框中包括了 10 年后你将得到的钱的数额。令人难以置信的是，10 年后你将获得 2593.74 元钱。
- 6) 单击“确定”按钮，退出消息框。

使用变量 iNumberOfYears 可控制循环执行的次数，最后的计算结果保存在变量 cSavings 中。你可能已经注意到，在 HowMuchMoney 过程中 Format 语句的用法。语句 Format(cSavings, "0.00") 指定将保存在 cSaving 中的结果显示两位小数。

在第 9 学时中，将看到 HowMuchMoney 过程被转变为可以在工作表上使用的函数。

7.2 Do 循环

只有当需要执行一系列语句的确定次数时，For 语句才便于使用。为了克服这种局限性，VBA 为 For 语句提供了另一种称为 Do 循环的语句。Do 循环是条件循环。共有两种 Do 循环语句：Do While 和 Do Until。Do While 语句在某个特定的条件为 True 时重复执行一组语句，而 Do Until 则重复执行一组语句直到某个特定的条件变为 True 为止。Do 循环语句的语法如下：

语法 1：

```
Do [{While|Until} condition]
    [statements]
    [Exit Do]
    [statements]
Loop
```

语法 2：

```
Do
    [statements]
    [Exit Do]
    [statements]
Loop [{While|Until} condition]
```

在 Do 循环语句的两种语法之间有一个细微的区别。语法 1 将测试条件放在循环的开始部分，这意味着，如果条件不满足，Do 循环中的语句一次都不会执行。而语法 2 将测试条件放在循环的结束部分，这意味着，Do 循环中的语句至少执行一次。在两种语句中，都可以用可选的 Exit Do 语句在必要的时候退出循环。和 Exit For 从句一样，通常将 Exit Do 语句放置在 If 或者 Select 语句中。

请完成如下步骤，创建一个使用 Do 循环的过程：

1) 插入一个新的名为“EnterName”的子程序。该过程将提示用户输入名字，直到用户输入了名字或者选择退出才结束执行。

2) 为 EnterName 过程输入如下代码：

```
Dim sName As String
Dim iResponse As Integer
```

```
SName = ""  
  
Do While sName = ""  
    sName = InputBox("Please enter your name: ")  
    If sName = " " Then  
        iResponse = MsgBox("Do you wish to quit? ", vbYesNo)  
        If iResponse = vbYes Then  
            Exit Do  
        End If  
    End If  
Loop
```

值得提醒的是，当对用 InputBox函数创建的输入框单击“取消”按钮进行响应时，返回值为空字符串。如果对用 Application.InputBox方法创建的输入框单击“取消”按钮进行响应时，返回值为 False。

3) 运行该过程。

4) 为了响应消息框，按下空格键或者不在输入框中输入任何字符就单击“取消”按钮，显示一个询问是否希望退出的消息框。

5) 单击“否”按钮，再次显示输入框，输入你的名字，过程结束。

在第4学时中，你已经学习了数组的概念。在当时的讨论中，谈到了动态数组。动态数组是大小可变的数组。将动态数组和 Do循环结合起来使用非常有用。完成如下步骤，创建一个和Do循环结合使用的动态数组：

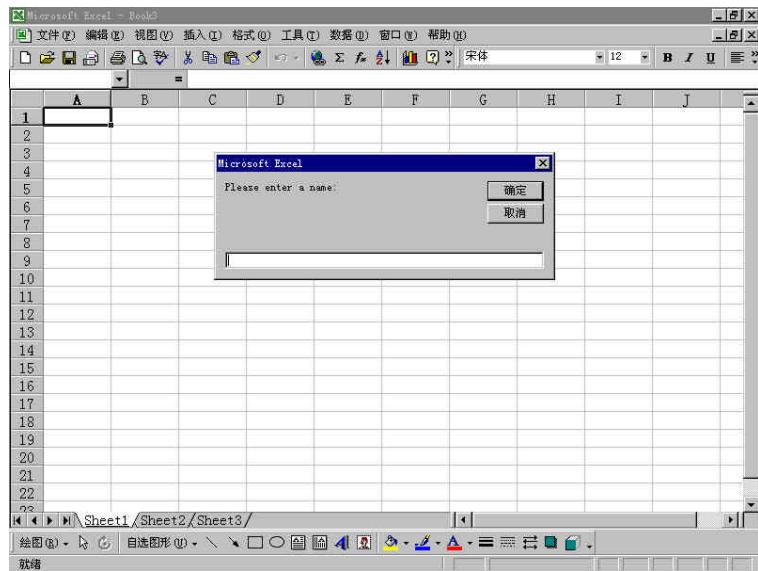
1) 插入一个新的名为“ListOfNames”的过程。该过程将让用户输入名字，直到用户选择不再继续为止。每次用户输入一个新的名字，动态数组都增加一个元素来保存这个名字。

2) 为ListOfNames过程输入如下代码：

```
Dim iCount As Integer  
Dim sNames() As String  
Dim iResponse As Integer  
Dim i As Integer  
  
iResponse = vbYes  
  
Do While iResponse = vbYes  
    iCount = iCount + 1  
    ReDim Preserve sNames(iCount) As String  
    sNames(iCount) = InputBox("Please enter a name: ")  
    If sNames(iCount) = " " Then  
        iResponse = MsgBox("Do you wish to continue?", vbYesNo)  
        If iResponse = vbYes Then  
            sNames(iCount) = InputBox("Please enter a name: ")  
        End If  
    End If  
Loop  
  
For i= 1 To iCount - 1  
    MsgBox("Name #" & i & " is" sNames(i))  
Next
```

3) 运行该过程，显示的第一个消息框如图 7-3 所示。

图7-3 本输入框将一直显示，直到选择“取消”按钮为止



- 4) 输入“Bob”作为第一个名字并按回车键。
- 5) 输入“Mary”作为第二个名字并按回车键。
- 6) 输入“Tom”作为第三个名字并按回车键。
- 7) 单击“取消”按钮，显示一个消息框询问是否希望继续。
- 8) 单击“否”，显示一个包含第一个名字 Bob 在内的消息框。
- 9) 单击“确定”按钮，显示名字 Mary。
- 10) 单击“确定”按钮，显示名字 Tom。
- 11) 单击“确定”按钮，过程结束。

过程ListOfNames的强大之处在于，在过程开始运行之前，不用知道将要输入多少个名字。

完成的过程如程序清单 7-1 所示。

程序清单 7-1 ListOfNames 过程

```

1: Sub ListOfNames ()
2:   Dim iCount As Integer
3:   Dim sNames() As String
4:   Dim iResponse As Integer
5:   Dim i As Integer
6:
7:   iResponse = vbYes
8:
9:   Do while iResponse = vbYes
10:    iCount = iCount + 1
11:    ReDim Preserve sNames(iCount) As String
12:    sNames(iCount) = InputBox("Please enter a name: ")
13:    If sNames(iCount) = " " Then
14:      iResponse = MsgBox("Do you wish to continue?",vbYesNo)
15:      If iResponse = vbYes Then
16:        sNames(iCount) = InputBox("Please enter a name: ")

```

```
17:     End If  
18: End If  
19: Loop  
20:  
21: For i = 1 To iCount - 1  
22:     MsgBox "Name # " & i & "is " & sNames(i)  
23: Next  
24: End Sub
```

本过程中首先需要查看的是变量的声明：

```
Dim iCount As Integer  
Dim sNames() As String  
Dim iResponse As Integer  
Dim i As Integer
```

变量iCount在Do循环的开始部分不断递增，并且用来控制数组的大小和对数组元素的选择。sName数组是一个动态数组，每次执行Do循环数组大小都会改变。iResponse变量跟踪用户是否希望继续输入名字。iResponse同时也用来控制循环是否继续执行。整数i用来控制显示名字的For语句。声明变量之后，首先应当对iResponse变量进行初始化：

```
iResponse = vbYes
```

现在，该变量已经初始化了，接下来开始执行循环语句：

```
Do While iResponse = vbYes
```

因为iResponse已经初始化为vbYes，所以循环开始执行。循环所做的第一件事是将iCount加1。循环第一次运行时，iCount的值变为1；第二次运行时，iCount的值变为2；等等。iCount增加后，就可用来重新设置数组sNames的大小。新增加的元素用于保存新的名字：

```
iCount = iCount + 1  
ReDim Preserve sNames(iCount) As String  
sNames(iCount) = InputBox("Please enter a name: ")
```

接下来，需要做的是测试用户是否选择了“取消”按钮或者是否忘记了输入一个新的名字：

```
If sNames(iCount) = "" Then  
    iResponse = MsgBox("Do you wish to continue? ", vbYesNo)  
    If iResponse = vbYes Then  
        sNames(iCount) = InputBox("Please enter a name: ")  
    End If  
End If
```

当循环执行完毕后，可以用For语句显示用户输入的名字。在For语句中，可用语句的计数器使用数组中相应的元素：

```
For i = 1 To iCount - 1  
    MsgBox "Name # " & i & "is " & sNames(i)  
Next
```

注意，For语句中的计数器是从i=1到iCount-1，需要将iCount减1，以补偿变量在Do循环顶部的加1。如果希望看到不减1的结果，可以从该语句中删除-1，并再次运行，你将看到一个多余的显示空名字的消息框。现在你不仅明白了Do循环的作用，同时也看到了动态数组的作用。

7.3 学时小结

在本学时中，你学到了VBA中的循环机制。通过使用For语句，掌握了如何多次执行相同的语句。通过使用Do循环，掌握了如何有条件地多次执行相同的语句。本学时也扩展了你的有关数组，特别是动态数组的知识。将动态数组和循环结合起来是一种非常有效的创建动态变量的方式。

7.4 专家答疑

问题：使用循环的目的是什么？

解答：是为了多次执行一组语句。

问题：什么时候应该将条件放在Do循环的开始部分，而不是将条件放在Do循环的结束部分？

解答：通过询问自己循环需要执行的最少次数就可以知道。如果答案是1，那么将条件放在循环的结束部分；如果答案是0，那么将条件放在循环的开始部分。

7.5 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

7.5.1 思考题

- 1) VBA中两种主要的循环类型是什么？
- 2) 从循环中跳出来的语句是什么？
- 3) Do循环的两种类型是什么？
- 4) 判断题：Do循环的条件必须放在循环的开始部分。

7.5.2 练习题

首先，创建一个名为EnterHours的过程，用For语句让用户输入雇员每周五天的工作时间，并计算出总的工作时间，用消息框显示总的工作时间。

接下来，创建另一个名为Salary的过程。提示用户输入雇员每小时的工资。该公司每小时的最低工资为6元。用Do循环不断显示输入框，直到用户输入了所需的数据。用这些数据根据EnterHours计算出来的工作时间算出总的工资，并用消息框显示。

提示：将保存总的工作时间的变量声明为公共变量，先运行EnterHours，再运行Salary。

第8学时 对象介绍

不论你是否已经意识到，到目前为止，你已经为用 VBA语言开发应用程序打下了基础。已经学会了创建变量、常量、消息框、输入框、条件逻辑和循环。缺乏的是直接用 Excel开展工作的经验。本学时中这种情况会得到改变，你将开始在 Excel的特定环境中开展工作。

本学时的重点包括：

- 讨论VBA中的对象究竟是什么
- 讨论对象、属性和方法
- Excel的对象模型概览
- 在VBA代码中使用范围（ Range）对象
- 对已知的和未知的范围进行处理
- 改变用代码进行操作的范围的大小和位置
- 在范围内输入数值

8.1 对象是什么

新术语 在开始讨论对象之前，需要先指出一些也许不太明白的情况。 VBA是一种语言，因此它具有一定的构造。当学习一种语言时，学到的大多数知识是如何使用该语言来描述某种事物的行为、操作或者外观，也就是英语中的名词或者主语。而在 VBA中被描述的对象称为对象。对象就是在编程环境中希望控制的目标。

请举出Excel中元素的名字。你想到的也许是工作簿、工作表、单元格、范围和图表，而这些实际上只是一些Excel对象。

8.2 对象、属性和方法

新术语 通过属性和方法可以控制对象。属性是指对象的特征，改变属性的值可以改变对象的行为或者外观。例如，使用属性，可以改变某个范围内的单元格的颜色、数值、字体或者格式。

新术语 另一方面，方法是对象可以执行的操作。范围对象的 Clear方法就是一个例子。

现在，回到对 VBA和英语的比较。对象相当于 VBA语言中的名词，属性相当于形容词，而方法则相当于动词。

可以用对象、属性和方法这样的术语对任何事物进行描述。例如你可以描述自己。你可以称为一个“ human ”对象，你的属性包括名字、身高、体重、眼睛的颜色、头发的颜色和年龄等，你的一些方法可以是睡觉、吃饭、跑步和编程等。在这里不是试图将对象、属性和方法的概念复杂化，实际上，它们处理起来非常简单。

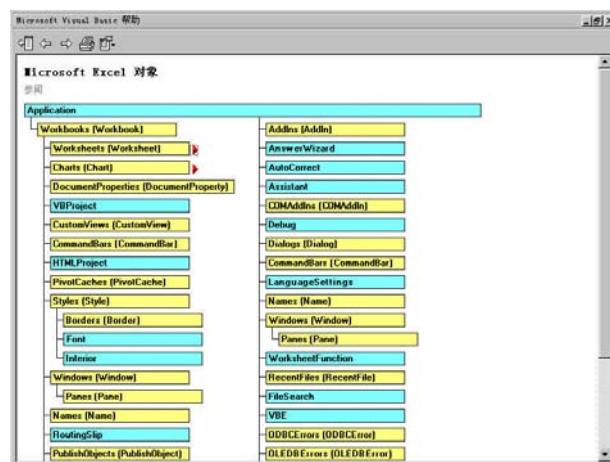
8.3 Excel的对象模型

新术语 开始在Excel（或者任何支持 VBA的应用程序）中用 VBA进行编程之前，必须先查看Excel的对象模型。对象模型是用来描述对象之间的联系的。

Excel的对象模型中有100多个对象，但是，不要感到惊慌，你不用学习它们中的全部。也许在编程时，你只会用到其中的20个对象或者更少。要查看Excel中对象的列表，请完成如下步骤：

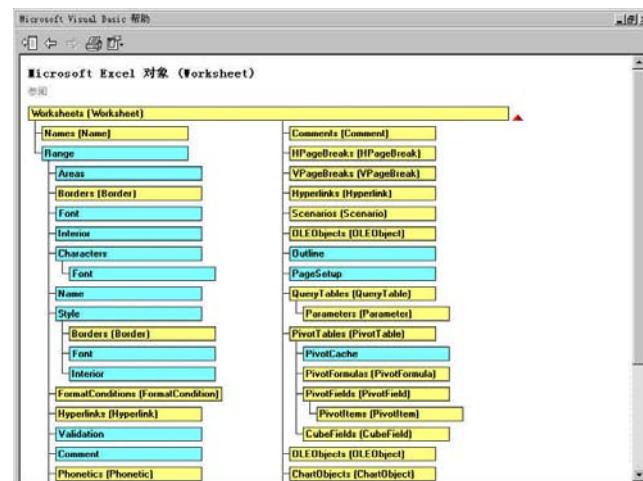
- 1) 关闭所有打开的工作簿，这样可以使工作环境更简洁。
- 2) 打开一个新的工作簿。
- 3) 按下Alt+F11键，打开Visual Basic编辑器。
- 4) 按下F1键，激活“帮助”。
- 5) 输入问题“什么是对象”。
- 6) 按回车键，从列出的主题中选择“Microsoft Excel对象”，此时会显示一张详细的对象模型图，如图8-1所示。

图8-1 在线帮助系统十分详尽地描述了Excel的对象模型



- 7) 单击“Worksheets(Worksheet)”右面的箭头，将扩展这一级对象模型，如图8-2所示。
- 查看这个模型，在层次结构的顶部，可以看到“Application”对象。在第5学时中，你已经用过Application对象了。在层次结构中，可以看到的下一个对象是“Workbook”对象，它和Excel文件等价。如果你对Excel环境已经比较熟悉，对在工作簿(Workbook)中发现工作表(Worksheets)对象不应感到惊奇，而在工作表中有包括范围(Range)对象。现在，你已经知道了在Excel中进行程序设计是最常用的五个对象中的四个！

图8-2 在Worksheet对象中包括大量对象



8.4 最常用的5个对象

尽管在Excel的对象模型中包括100多个对象，但你会发现程序设计主要集中在如下五个对象上：

- Application
- Workbook
- Worksheet
- Range
- Chart

这并不是说用不着使用其他对象，只是说明这五个对象最为常用。Application对象代表Excel。使用Application对象可以控制应用程序级的设置、内置的Excel函数以及高级方法，例如InputBox方法。

Workbook对象是指Excel中的工作簿，即是说Excel文件。在VBA环境中，不说打开一个文件，而称为打开一个工作簿；也不说保存一个文件，而称为保存工作簿。

学习Excel时最先了解到的其中一点就是，Workbook中包括Worksheet。Worksheet是Workbook中独立的页，数据就保存在Worksheet中。

Worksheet中包括单元格（Cell）。你也许会认为不得不编写大量的代码以对单元格对象进行控制，但是，实际上没有单元格这样的对象。有单元格属性，将在后面的学时中学到。相反，你将对范围（Range）对象进行处理。范围对象是指一个或者多个单元格。

大多数Excel用户都使用Excel的图表功能，所以你经常需要处理图表（Chart）对象。用“图表向导”创建图表时所做的一切都可以通过VBA代码做到。

你也许还没有意识到，在第3学时中学习的控件也是对象，包括命令按钮、选项按钮、复选框以及标签。在第15学时中，将进一步学习控件对象。

8.5 对象的层次结构意味着什么

新术语 当查看对象模型时，会发现它看上去像一幅层次结构图。Application处于层次结构的顶部。在Application对象下面，可以找到Workbook对象。用VBA的术语来说，Workbook对象包含在Application对象中。更进一步，Worksheet对象包含在Workbook对象中。包含（Containment）意味着一个对象可以包含其他对象。

将对象模型想象成俄罗斯嵌套木偶，当打开第一个木偶时，可以发现包含在它内部的另一个较小的木偶；而打开第二个木偶，又可以看到包含在它内部的另一个木偶，等等。对象的层次结构与此类似。

8.6 在代码中引用对象

你也许希望知道为什么必须掌握包含的概念。当在代码中引用对象时，就要用到包含的概念。在代码中要限定一个对象名，必须通过对象模型中的层次结构。例如，要引用工作簿Book1中的工作表Sheet1上的范围A1，必须使用如下代码：

```
Application.Workbooks("Book1").Worksheets("Sheet1").Range("A1")
```

实际上，在大多数情况下，可以省略对Application对象的引用（Application.InputBox是一个例外）。可以通过如下代码引用范围A1：

```
Workbooks("Book1").Worksheets("Sheet1").Range("A1")
```

在代码中，并不总是需要使用完全限定的对象名。在某些情况下，例如在代码中某个工作表刚被激活时，可以使用如下较短的引用方式：

```
Range("A1")
```

如何限定对象取决于你的经验。限定一个对象必须可以将它和同名的对象区分开来。例如，你可能有两个工作簿，每个工作簿中都有一个名为 Sheet1的工作表，在这种情况下，需要使用完整的限定名称以免混淆。

8.7 在代码中使用对象

当在代码中使用对象时，实际上是在做以下三种事情：

- 设置对象的属性。
- 获取对象的属性。
- 执行对象的方法。

在下一部分中，将在这方面进行有一点深度的探索。

8.7.1 使用属性

要设置对象的属性，可以使用如下的基本语法：

```
Object.propertyname = value
```

Object是对象名，propertyname是指需要改变的对象的属性名字，而 value是该属性可以设置的数值。对象的名字和属性之间用“.”分开。例如，要设置范围对象的 Value属性，可以使用如下代码：

```
Range("A1").Value = 100
```

获取对象的属性和设置对象的属性相似，基本的语法是：

```
var_name = Object.propertyname
```

在上面的语法中，将对象某个属性的设置读入一个变量或者其他存储位置中，诸如另一个属性。如果要获取范围对象的 Value属性，可以使用如下代码：

```
Dim sngValue As Single  
SngValue = Range("A1").Value
```

可以用其他多种方式来使用属性的设置。例如，可以使用如下代码在消息框中显示某个属性设置：

```
MsgBox "The range's value is " & Range("A1").Value
```

8.7.2 使用方法

通过如下语法可以实现对象的方法：

```
Object.Method
```

当执行一个对象的方法时，需要用“.”将对象和方法的名字分开。例如，如果需要执行工作簿的“Open”方法，可以使用如下代码：

```
Workbooks("VBA Example").Open
```

某些方法是带参数的，而参数又可能是必须的或者可选的。例如，以下代码将工作簿保存为“Current Budget”：

```
ThisWorkbook.SaveAs Filename := "Current Budget"
```

你也许已经猜到，有时对方法的引用就像对专门为某个对象设计的过程和函数的使用。这也解释为什么有的方法具有参数，有的方法甚至还有返回值。

8.7.3 获取有关属性和方法的信息

VBA的帮助系统为提供有关属性和方法的信息做了调整。通过如下步骤，可以看到这方面的一个例子：

- 1) 在当前工作簿中插入一个模块。
- 2) 在Visual Basic编辑器中，输入“worksheet”。
- 3) 将插入点指向“worksheet”。
- 4) 按下F1键，显示“帮助主题”对话框，如图8-3所示。

图8-3 “帮助主题”对话框
常常显示多个选项



5) 选择项目“Worksheet(o...)”，并单击“帮助”按钮，显示“Worksheet对象”帮助主题如图8-4所示。

图8-4 帮助主题“Worksheet
对象”不仅描述该对象，
还提供该对象可用的属
性和方法的信息



- 6) 查看帮助主题的顶部，可以看到列出的属性和方法。单击“属性”，显示“已找到的

主题”对话框，如图8-5所示。

图8-5 因为选择了“属性”，
所以列出了工作表的
所有属性



7) 此时，如果希望获得某个属性的详细信息，可以选中该属性，并单击“显示”按钮。而为了本学时的目的，请单击“取消”按钮返回“Worksheet对象”帮助主题。

8) 单击“方法”，在“已找到的主题”对话框中显示所有可用的方法。

9) 单击“取消”按钮返回“Worksheet对象”帮助主题。

10) 最小化帮助窗口。本学时的后面部分还要使用帮助系统。

你会发现，作为一个初级的VBA程序员，你知道希望做些什么，但是不知道如何去做。对于这种情况，帮助系统非常有用。找到对象的帮助主题并首先查看对象的属性，通常能够找到需要的属性。如果没有找到，可以在对象的方法中试一试。

8.8 使用对象变量

新术语 在学习变量的数据类型时，有一种类型没有讨论。这种类型就是对象变量，它是指向某个对象的变量。因为对象变量指向一个对象，所以它们可以使用相应用对象的属性和方法。

创建对象变量和其他变量完全类似，也通过使用Dim语句。可以使用通用的对象类型，也可以使用特定的对象类型。程序清单8-1显示了对象变量声明的一些例子。

程序清单8-1 对象变量的声明

```

1: Dim BudgetSheet As Object
2: Dim AnotherBudget As Worksheet
3: Dim WorkingFile As Workbook
4: Dim DeptCodes As Range

```

第一条Dim语句使用通用的对象数据类型。通常情况下，不要使用这种对象变量的声明方法。其他的Dim语句显示了更好的声明对象变量的方法。如果知道将要创建的对象的类型，那么最好在定义对象变量时加以指明。对象变量声明后，可以用Set语句将一个对象指定给该变量。程序清单8-2显示了Set语句的用法。

程序清单8-2 Set语句

```

1: Set BudgetSheet = Workbooks("Finance").Worksheets("Budget")
2: Set AnotherBudget = Workbooks("MIS").Worksheets("Budget")
3: Set WorkingFile = WorkBooks("Finance")
4: Set DeptCodes =Workbooks ("Budget").Worksheets ("Category"). Range ("A1:A12")

```

设置了变量所引用的对象之后，就可以在代码中像对象名一样使用它们。程序清单8-3举

出了一个使用对象变量的例子。

程序清单 8-3 使用对象变量

```
1: Sub ObjectvarExample ()  
2:   Dim WorkingRange As Range  
3:  
4:   Set WorkingRange = Workbooks ("Hour8").Worksheets ("Sheet1").Range ("A1:D1")  
5:  
6:   WorkingRange.Font.Bold = True  
7:   WorkingRange.Font.Italic = True  
8:   WorkingRange.Font.Name = "Courier"  
9:  
10: End Sub
```

在这个例子中，可以看到变量 WorkingRange 用在对象名的位置上。正如你所见，这可以使你避免输入一个很长的名字全称。当进一步学习本书时，将发现对象变量的其他用途。

在进行程序设计时，采用这样的原则：如果需要输入同样的对象的名字全称两次以上，就创建一个对象变量以节省输入时间。

8.9 集合

请先花点时间看一看曾经用过的对象名字的全称：

Application.Workbooks("Book1").Worksheets("Sheet1").Range("A1")

请注意单词的复数形式——Workbooks 和 Worksheets，这就是集合。一个集合是指一组相似的对象。Workbooks 是一个集合，而 Worksheets 是另一个集合。在前面的例子中，Book1 是集合 Workbooks 的一个元素。

在大多数情况下，集合都是复数形式的单词。

没有 Ranges 对象，可以添加一个这样的对象。但是不能添加更多的 Ranges，因为 Excel 已经做了定义和限制。

要查看 Excel 中的一些可用的集合，请按照如下步骤：

- 1) 恢复帮助窗口。
- 2) 选择“目录”选项卡。
- 3) 打开“Microsoft Excel Visual Basic 参考”主题。
- 4) 选择“Microsoft Excel 对象”。显示“Microsoft 对象模型”。
- 5) 滚动到“Microsoft Excel 对象”主题的底部，将看到对象模型的代码是有颜色的。黄色的项是对象和集合，而青色的项只是对象。

对象模型图提供了理解和区分 Excel 中的集合的非常好的工具。看完对象模型之后，请关闭“帮助”窗口。

8.9.1 Add方法

所有集合的一个共有特点是可以添加项目。通过添加可以在集合中创建新的元素。要在

一个集合中创建新的元素，可以使用 Add方法。例如，要添加一个新的工作簿，可以使用如下代码：

```
Workbooks.Add
```

这行代码等价于打开Excel的文件菜单并添加一个新的工作簿。要添加一个新的工作表到工作簿中，可以使用如下代码：

```
Worksheets.Add
```

8.9.2 Count属性

集合支持一个非常有用的名叫 Count的属性。Count属性保存集合中元素的数目。如果希望知道一个工作簿中包括多少张工作表，可以使用如下代码：

```
Dim iWSCount As Integer  
iWSCount = Worksheets.Count
```

你也许想知道为什么要使用 Count属性。设想你正在创建一个应用程序，其中包含与一周的各个工作日相对应的工作表，当工作簿完成之后，可以通过使用 Count属性来检测工作簿是否包括五张工作表。在这种情况下，可以使用类似程序清单 8-4中的代码。

程序清单 8-4 使用Count属性

```
1: Sub CountWorkSheets ()  
2:   Dim iWSCount As Integer  
3:   Dim sMessage As String  
4:  
5:   iWSCount = Worksheets.Count  
6:  
7:   If iWSCount <> 7 Then  
8:     sMessage = "The workbook contains "& iWSCount  
9:     sMessage = sMessage & ". It should contain 7 worksheets."  
10:    MsgBox sMessage  
11:   End If  
12: End Sub
```

8.10 学时小结

在本学时中，学习了VBA中一些最重要的概念：对象、属性和方法。现在，你掌握了如何使用方法和属性来控制对象的功能和外观。

从现在开始，将学习新的对象以及它们的属性和方法。有了这方面的知识，就能够为你的应用程序添加越来越多的功能。

8.11 专家答疑

问题：Excel中的对象有什么共有的特征？

解答：所有的对象都具有属性和方法。属性控制对象的特征、外观以及行为，而方法是指对象所能执行的操作。

问题：Excel的对象模型显示为树状或者为层次结构，这是为什么？

解答：这是因为包含的概念，一些对象包含在其他对象中，这就决定了对象模型的外观。

8.12 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

8.12.1 思考题

- 1) 怎样设置属性？
- 2) 怎样调用方法？
- 3) 将一个对象指定给一个对象变量用什么语句？
- 4) 判断题：只有对象才有属性和方法，而集合没有。
- 5) 在集合中怎样创建一个新的元素？

8.12.2 练习题

使用对象浏览器或者在线帮助，找到下面的属性。

对于Application对象：

- Excel安装的目录路径。
- 使用的操作系统。
- Excel注册用户的名字。

对于Workbook对象：

- Workbook的保存位置。

使用这些信息，你将编写一个新的过程。在当前工作簿中插入一个新的模块。创建一个新的名为TellMeMore的过程。该过程需要使用你找到的属性显示一系列的消息框。第一个消息框应当和图8-6相似。运行你的过程。

提示：Application有一个名为ThisWorkbook的属性，其中包含当前工作簿的名字。当生成消息框显示工作簿是否已经保存时，可以使用这个属性。

第9学时 常用的对象

现在是真正学习 Excel 对象的时候了，本学时主要讲述三个常用的对象——Application、Workbook 和 Worksheet。将学习如何使用这些对象的属性和方法。

本学时的重点包括：

- Application 对象在 VBA 中的作用
- 创建和控制 Workbook 对象
- 使用 Worksheet 对象

9.1 Application 对象

在前一学时中，已经知道 Application 对象处于 Excel 对象层次结构的顶层。这对于作为一个程序员的你究竟意味着什么？这意味着你可以使用 Application 对象来控制应用程序级的设置和选项，诸如在“工具”、“选项”菜单中可以找到的项目一样。要查看一个这样的例子，首先请关闭所有打开的工作簿并打开一个新的工作簿。接着完成如下步骤：

- 1) 选择“工具”、“宏”、“录制新宏”，打开“录制宏”对话框。输入“AppSettings”作为宏名，宏的保存位置选择“This Workbook”。
- 2) 单击“确定”按钮，开始录制宏。
- 3) 选择“工具”、“选项”菜单，显示“选项”对话框。选择“视图”选项卡，清除“状态栏”复选框。
- 4) 选择“常规”选项卡，选中“R1C1引用风格”复选框，单击“确定”按钮。
- 5) 停止录制宏。
- 6) 选中“工具”、“选项”菜单，选择“视图”选项卡，选中“状态栏”复选框。
- 7) 选择“常规”选项卡，清除“R1C1引用风格”复选框，单击“确定”按钮。
- 8) 选择“工具”、“宏”、“宏”，显示“宏”对话框。
- 9) 选择“编辑”，查看宏的代码。打开 Visual Basic 编辑器。

程序清单 9-1 中显示了所录制的 AppSettings 宏的代码。

程序清单 9-1 AppSettings 过程

```
1: Sub AppSettings ()  
2:   With Application  
3:     .ReferenceStyle = xlR1C1  
4:     .UserName = "Sharon Podlin"  
5:     .StandardFont = "Arial"  
6:     .StandardFontSize = "10"  
7:     .DefaultFilePath = "C:\WINNT\Profiles\Administrator\Personal"  
8:     .EnableSound = False  
9:     .RollZoom = False  
10:    End With  
11:    Application.DisplayStatusBar = False  
12: End Sub
```

对于上面的代码，需要注意的主要是操作的对象，即Application对象。通过上面的说明，你可以在一定程度上了解这个对象的重要性。

9.1.1 使用Excel的内置函数

Application对象并不仅限于应用程序级的设置和选项，它还是 Excel的内置函数的拥有者。如果希望在你的 VBA过程中使用 SUM、AVERAGE、MAX、IRR或者任何其他的内置函数，必须使用Application对象。

请记住，采用 Excel作为开发平台的优点是 Excel具有大量的内置函数。在你的 VBA代码中为什么不利用 Excel的这个特征呢？请完成如下步骤，在 VBA代码中使用 AVERAGE和SUM函数：

1) 创建一个新的名为“ BuiltIns ”的过程。

2) 为该过程输入如下代码：

```
Dim sngAnswer As Single
```

```
sngAnswer = Application.Average(Worksheets("Sheet1").Range("A1:A4"))
MsgBox "The average for this range is " & sngAnswer
```

```
sngAnswer = Application.Sum(Worksheets("Sheet1").Range("A1:A4"))
MsgBox "The sum of this range is " & sngAnswer
```

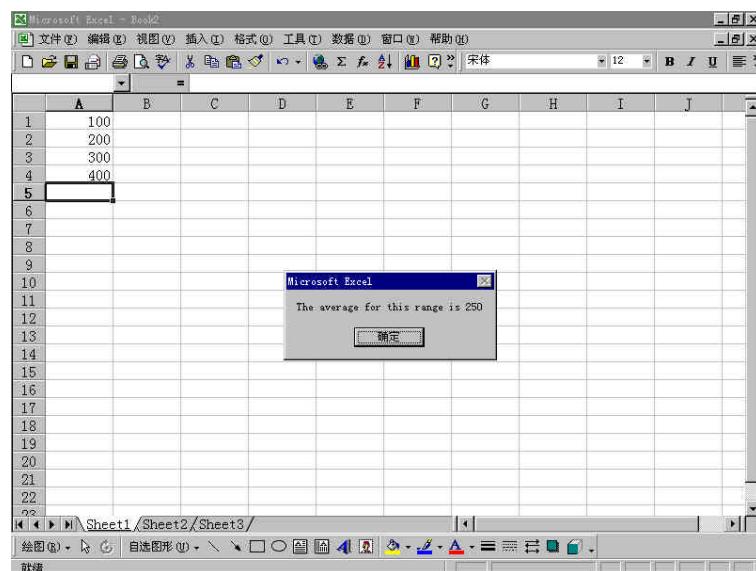
3) 回到工作簿中的Sheet1。

4) 在单元格 A1中输入 100，在单元格 A2中输入 200，在单元格 A3中输入 300，在单元格 A4中输入 400。

5) 运行BuiltIns过程。第一个消息框显示输入数字的平均值，如图 9-1所示。

可按下 Alt+F8键打开“宏”对话框并从中运行该过程。

**图9-1 消息框中显示的结果
是用Excel的内置函
数计算出来的**



6) 单击“确定”按钮，退出该消息框。第二个消息框显示输入数字的总和。

7) 单击“确定”按钮，退出消息框，过程运行结束。

正如你在这个例子中看到的那样，通过在VBA过程中用Application对象限定函数名并提供任何必须的参数值，就可以使用任何Excel的内置函数。

9.1.2 Application对象有用的属性和方法

你将发现Application对象有几种非常有用的属性和方法。其中一些属性如下：

- ActiveWorkbook 返回当前活动的工作簿。
- ActiveSheet 返回当前活动的工作簿中活动的工作表。返回的工作表可以是Excel支持的任何工作表类型，包括工作表和图表工作表。
- ActiveCell 返回当前活动的工作簿中活动的工作表中活动的单元格。
- ThisWorkbook 返回正在执行的过程所驻留的工作簿。
- MailSystem 返回本系统所采用的邮件系统。当进行邮件自动化时本属性非常有用。
- MailSession 用于检测用户是否登录了电子邮件。
- OperatingSystem 当开发由Windows和Mac用户使用的VBA应用程序时，这个属性非常有用。可以用这个属性来决定所使用的操作系统，并且做出必要的改变。
- Selection 用于决定当前选中了什么，可以是单元格、图表、图形对象等等。

你已经使用过一个Application对象的方法——InputBox。InputBox方法可以显示一个输入框，并且可以指定返回值的数据类型。Application对象的其他有用的方法包括：

- MailLogon和MailLogoff 和MailSystem和MailSession属性配合使用，这两个方法可以登录和退出电子邮件系统。
- Quit 用于退出Excel。
- Run 用来执行Excel4.0宏。

9.2 Workbook对象

现在，你已经知道Workbook对象代表Excel文件。正因为如此，你也许可以立刻想到需要使用Workbook对象来处理的各种类型的事件：打开、保存、打印和关闭。以下首先学习Workbook对象的一些最常用的方法，而不是从这个对象的属性开始学习。

- Activate 本方法用来激活工作簿。
- Close 用于关闭工作簿。
- Save 用于保存工作簿。

如果在使用Save方法之前没有保存过该工作簿，该工作簿将以它当前的名字保存。这意味着它将被保存为诸如Book1这样的名字。

- SaveAs 用于保存工作簿。本方法和Save方法的区别在于本方法有几个有用的可选参数，包括Filename、FileFormat、Password、WriteResPassword和ReadOnlyRecommended。
- PrintOut 本方法用于打印整个工作簿。
- PrintPreview 本方法用于在打印预览中显示工作簿。

还将使用几个 Workbook 对象的属性。和 Application 对象一样，Workbook 对象支持 ActiveSheet 属性。如果需要找到保存工作簿的目录，可以使用 Path 属性。Saved 是 Workbook 对象的一个特别有用的属性，如果工作簿已经保存，这个属性的返回值为 True，这意味着所有的变更都已经得到了保存。如果改变了工作簿并且还没有保存，则返回 False。

在前面的讨论中还没有提及的是如何创建一个新的工作簿对象。换句话说，和单击 Excel 的“新建”工具栏按钮等价的 VBA 代码是什么？其实，通过第 8 学时的学习，你已经知道了这个问题的答案：需要使用 Add 方法。请记住，当向集合中添加一个新的对象时，都要使用 Add 方法。在下面的练习中，将创建一个工作簿，然后进行打印、保存和关闭。

1) 创建一个新的名为“WorkbookExample”的过程。

2) 为该过程输入如下代码：

```
Dim wbNewWorkbook As Workbook  
Set wbNewWorkbook = Workbooks.Add  
wbNewWorkbook.Worksheets("Sheet1").Range("A1").Value = 100  
wbNewWorkbook.SaveAs "Hour 9"  
wbNewWorkbook.Close  
MsgBox "The workbook is closed."
```

3) 运行程序，你也许看不到进行了什么操作。

4) 当显示提示工作簿已经关闭的消息框时，请单击“确定”按钮。

5) 打开该工作簿（如果你还在 Visual Basic 编辑器中工作，需要回到 Excel 应用程序窗口，并通过“文件”菜单打开该工作簿）。

6) 在单元格 A1 中，可以看到数值 100，这可以使你知道这个工作簿正是用你的过程创建的。

在练习完上面的过程后，你应当感觉到自己已经成为一个真正的 Excel 开发者了。现在将对组成该过程的代码进行检查。程序清单 9-2 显示了完整的过程。

程序清单 9-2 WorkbookExample 过程

```
1: Sub WorkbookExample ()  
2:   Dim wbNewWorkbook As Workbook  
3:   Set wbNewWorkbook = Workbooks.Add  
4:   wbNewWorkbook.Worksheets("Sheet1").Range("A1").Value = 100  
5:   wbNewWorkbook.SaveAs "Hour 9a"  
6:   wbNewWorkbook.Close  
7:   MsgBox "The workbook is closed."  
8: End Sub
```

这个过程从创建对象变量开始：

```
Dim wbNewWorkbook As Workbook
```

在创建新的工作簿时会用到这个变量：

```
Set wbNewWorkbook = Workbooks.Add
```

这条语句通过向 Workbook 集合中添加一个新的元素而创建了一个新的工作簿。在创建工作簿时，Excel 为它指定诸如 Book1 或者 Book2 这样的名字。你不能确保工作簿的名字会是什么。为了避免这个问题，可以将工作簿指定给一个对象变量，并且在代码中通过对象变量对新的工作簿进行引用。

接下来该过程设置了单元格 A1 的数值，以便工作簿有一些数据可用：

```
wbNewWorkbook.Worksheets("Sheet1").Range("A1").Value = 100
```

最后执行Workbook对象的方法SaveAs和Close。添加一个消息框是一种知道过程已经执行完毕的简单方法：

```
wbNewWorkbook.SaveAs "Hour 9a"  
wbNewWorkbook.Close  
MsgBox "The workbook is closed."
```

现在，你已经完成了这个练习，已经用VBA代码创建、保存并且关闭了第一个Excel工作簿！

当将工作簿保存为某个名字后，可以用这个名字对工作簿进行引用，也可以用对象变量对它进行引用，这完全取决于个人的兴趣。

9.3 Worksheet对象

现在，你已经掌握了如何使用 Workbook对象，接下来将学习使用 Worksheet对象。和学习 Workbook对象一样，请先花一点时间想一想，作为一个 Excel用户，可以用 Worksheet做些什么。你可能会选择工作表、给工作表取名，或者在工作簿中插入新的工作表、复制工作表和删除工作表。现在，你已经知道可以用 Worksheet对象来做些什么，还需要知道的是完成这些任务的属性和方法。

当对工作表进行处理时，常常要用到两个属性。其中一个是 Name属性，在VBA中设置 Name属性等价于双击一个工作表的标签并输入一个新的名字。例如，要将一张工作表的名字改为“Budget”，可以使用如下代码：

```
ActiveSheet.Name = "Budget"
```

如果希望对整个工作表进行某些格式设置，怎样才能选中工作表中的所有单元格呢？通过使用 Cells属性可以做到。如果希望设置一张工作表上所有单元格的字体，可以使用如下代码：

```
Worksheets("Sheet1").Cells.Font.Name = "Arial"
```

也许会用到以下三个Worksheet对象的方法：

- Activate 激活工作表。
- CheckSpelling 用于检查工作表中内容的拼写。
- Delete 用于删除某个工作表。

现在到了测试的时候了。怎样向工作簿中插入工作表？如果回答是使用 Add方法，那么你就过关了！要插入一张工作表并改变它的名字，可按照如下步骤：

1) 创建一个新的名为“ MyNewWorksheet ”的过程。

2) 为该过程输入如下代码：

```
Dim wsNewWorksheet As Worksheet
```

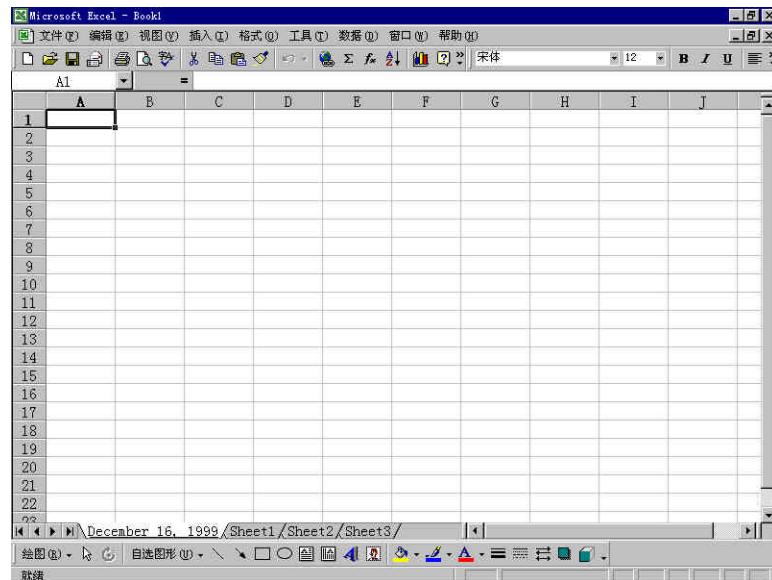
```
Set wsNewWorksheet = Worksheets.Add  
wsNewWorksheet.Name = Format(Date, "mmmm d, yyyy")
```

3) 运行该过程。

4) 如果是在Visual Basic编辑器中运行该过程，那么请回到 Excel应用程序窗口查看新添加的工作表，如图9-2所示。

5) 将工作簿保存为“ Hour9 ”，并关闭该工作簿。

图9-2 注意添加的工作表的名字应当和今天的日期相同



这部分代码和用于创建一个工作簿时的代码比较相似。和创建工作簿时一样，首先创建一个对象变量，再将它和 Add方法配合使用，以便可以在代码中用对象变量的名字来引用该对象：

```
Dim wsNewWorksheet As Worksheet
Set wsNewWorksheet = WorkSheets.Add
```

在插入工作表之后，就可以使用 Name属性了：

```
wsNewWorksheet.Name = Format(Date, "mmmm d,yyyy")
```

这行代码使用一个设置名字的有趣方式——采用当前的日期作为名字。通过Format函数可以控制日期的格式和显示方式，日期以月、日、年的方式显示。

许多开发者喜欢在文件名或者工作表名中使用当前的日期以避免名字重复。

9.4 学时小结

在本学时中，你在处理Excel的一些基本对象——Application、Workbook和Worksheet方面获得了大量的经验。通过使用Application对象，你掌握了如何进行应用程序级的设置和选项，以及使用Excel内置的函数。通过使用Workbook对象，你可以创建新的Excel工作簿文件，并且可以添加和操作工作簿中的工作表。在本学时中，你也有了更多的机会学习属性和方法。

在接下来的学时中，将学习另一个关键的Excel对象——Range对象。当使用Range对象时，你会发现这是作为Excel开发者的核心工作。

9.5 专家答疑

问题：在能够使用VBA进行工作之前，是否不得不学习大量的对象？

解答：不是。实际上，你也许只会用到少于20个对象。如果需要对不熟悉的对象进行处

理，不要忘了两个关键：在线帮助和录制宏。

问题：为什么要将Excel本身作为应用程序对待？

解答：只有将Excel作为一个应用程序才能对它进行某些操作。例如，可以对整个 Excel环境进行设置，可以查看公式栏或者控制重新计算。正因为如此，以及其他一些原因，Excel本身也是一个对象，形式上就是 Application对象。

问题：为什么创建工作簿和工作表时都要使用 Add方法？

解答：因为这是在向集合中添加元素。添加工作簿实际上是在向 Workbooks集合中添加 Workbook对象，而添加工作表则是在向 Worksheets集合中添加 Worksheet对象。

9.6 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

9.6.1 思考题

- 1) 如何使用 VBA中的MAX函数决定范围A1:C5中最大的数字？
- 2) 处于对象层次结构最外层的对象是什么？
- 3) 创建新的工作簿或者新的工作表需要使用什么方法？
- 4) 如何用VBA代码从工作簿中删除工作表？
- 5) 什么属性返回当前过程所驻留的工作簿？这个属性属于什么对象？
- 6) 判断题：在VBA中不能运行Excel4.0的宏。

9.6.2 练习题

创建一个新的名为“Hour9Lab”的过程。本过程必须完成如下任务：

- 创建一个新的工作簿。
- 向工作簿中添加一个新的工作表。
- 用你的名字为新添加的工作表命名。
- 将工作簿保存为“Hour9Lab”。

运行该过程。回到 Hour9Lab工作簿，并在添加到工作簿中的工作表上添加一些数据。创建另一个名为“SaveHour9”的过程，这个过程需要检测工作簿是否已经保存。如果还没有保存，那么就保存该工作簿并显示一个消息框。如果工作簿已经保存过了，则显示一个消息框提示用户已经保存了。运行该过程。

第10学时 Range对象

在VBA代码中，用得最多的单独的对象可能就要算 Range对象了。Range对象是Excel的对象世界中真正的工作对象。作为 Excel用户，你将把大部分时间花在对 Range对象的处理上。在本学时中，将学习多种对 Range对象进行处理的技术。

本学时的重点包括：

- 理解Range对象的作用
- 使用With语句
- 使用For Each语句
- 分析几个使用Range对象的过程

10.1 Range对象

作为Excel用户，你基本上是和工作表上的单元格打交道。而在 VBA中，则称为和Range对象打交道。对于一个程序员来说，在所有的 Excel对象中，对Range对象的处理是最频繁的。正因为如此，在本学时中，你将学习有关Range对象的各种知识。

一个Range对象可以是：

- 一个单独的单元格。
- 对单元格的选择。
- 多个选择。
- 一行或者一列。
- 一个3D区域。

10.1.1 Range对象的属性

和学习其他对象一样，首先需要学习 Range对象的一些属性和方法。Range对象最有用的一些属性如下：

- Address 本属性返回Range的当前位置。
- Count 本属性用于决定Range中单元格的数目。
- Formula 本属性返回用于计算显示值的公式。
- Offset 本属性对于从一个Range移动到另一个Range非常有用。
- Resize 通过它可设置当前选中的Range的大小。
- Value 返回Range的数值。

在上面的程序清单中列出的属性只是 Range对象属性很小的一部分，实际上，Range对象的属性多达几十个。在下面的练习中，将熟悉Range对象的一些属性。在开始前，关闭所有的工作簿，并打开一个新的工作簿。接着从创建一个新的工作表开始：

- 1) 在单元格B1中输入“100”，在单元格B2中输入“200”，在单元格B3中输入“300”。

3) 选中单元格B4，并输入“=SUM(B1:B3)”。

3) 按下Alt+F11键打开Visual Basic编辑器，在当前工作簿中插入一个新的模块。

4) 创建一个名为“RangeProperties”的过程。

5) 为该过程输入如下代码：

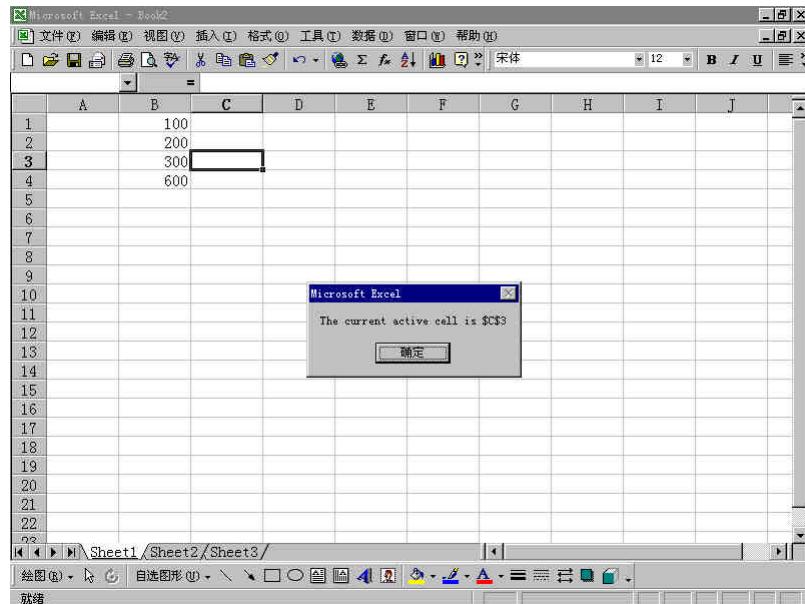
```
ThisWorkbook.Worksheets("Sheet1").Range("A1").Activate
ActiveCell.Offset(2,2).Activate
MsgBox"The current active cell is "& ActiveCell.Address
```

```
MsgBox"The value of B4 is "& Range("B4").Value
```

```
MsgBox"The formula of B4 is "& Range("B4").Formula
```

6) 回到工作簿的Sheet1，运行RangeProperties过程。显示的第一个消息框告诉你激活的单元格为C3，如图10-1所示。

图10-1 将Address属性用作创建消息框文本的内容



7) 单击“确定”按钮，下一个消息框显示单元格B4的数值。

8) 单击“确定”按钮，最后一个消息框显示单元格B4的计算公式。

9) 单击“确定”按钮，退出消息框。

熟悉上面例子中显示信息的细微区别了吗？为了确保对本例中代码的理解，请花一点时间复习一下。程序清单10-1显示了完成后的过程。

程序清单10-1 RangeProperties过程

```
1: Sub RangeProperties ()
2:   ThisWorkbook.Worksheets("Sheet1").Range("A1").Activate
3:   ActiveCell.Offset(2,2).Activate
4:   MsgBox "The current active cell is "& ActiveCell.Address
5:
6:   MsgBox "The value of B4 is "& Range("B4").Value
7:   MsgBox "The formula of B4 is "& Range("B4").Formula
8: End Sub
```

在本过程中，做的第一件事是激活单元格 A1：

```
ThisWorkbook.Worksheets("Sheet1").Range("A1").Activate
```

激活单元格 A1之后，用 Offset 方法移动到单元格 C3，并且在消息框中显示新激活的单元格的地址：

```
ActiveCell.Offset(2,2).Activate  
MsgBox "The current active cell is " & ActiveCell.Address
```

通过 Offset 属性，可以移动到另一个 Range 位置，这个属性的语法如下：

```
rangename.Offset(RowOffset,ColumnOffset)
```

可选参数 RowOffset 和 ColumnOffset 控制移动的方向。在本过程中，将这两个参数均设置为 2，这就将 Range 移动到了 C3。

接下来是显示单元格 B4 的数值和计算公式：

```
MsgBox "The value of B4 is " & Range("B4").Value  
MsgBox "The formula of B4 is " & Range("B4").Formula
```

这两行代码很好地说明了 Range 对象的 Value 属性和 Formula 属性的区别。Value 属性返回显示在单元格中的数值，而 Formula 属性则返回单元格中实际输入的公式。

10.1.2 Range 对象的方法

Range 对象同样具有多种多样的方法，包括：

- Activate 激活一个 Range。
- Clear 清除一个 Range 的内容。
- Copy 将一个 Range 的内容复制到剪贴板。
- Cut 将一个 Range 的内容剪切到剪贴板。
- PasteSpecial 将剪贴板的内容粘贴到 Range 中。
- Select 选择一个 Range。

10.2 使用 With 语句

现在，你对 Excel 中最常用的对象已经比较熟悉了。接下来，将学习一个结构，通过这个结构可以使对对象的处理更为容易。当对 Range 对象进行处理时，一个最普通的任务是设置各种各样的属性。程序清单 10-2 显示了用于对 Range 对象执行各种格式设置的代码的例子。

程序清单 10-2 设置 Range 对象格式的代码

```
1: Range("A1:A6").NumberFormat = "#,##0.00"  
2: Range("A1:A6").Font.Name = "Courier New "  
3: Range("A1:A6").Font.FontStyle = "Regular"  
4: Range("A1:A6").Font.Size = 11  
5: Range("A1:A6").Font.Strikethrough = False  
6: Range("A1:A6").Font.Superscript = False  
7: Range("A1:A6").Font.Subscript = False  
8: Range("A1:A6").Font.OutlineFont = False  
9: Range("A1:A6").Font.Shadow = False  
10: Range("A1:A6").Font.Underline = xlUnderlineStyleNone  
11: Range("A1:A6").Font.ColorIndex = xlAutomatic
```

如果需要输入程序清单 10-2 中的代码，你很快就会对重复地输入 Range("A1:A6") 感到

厌烦。可以用对象变量来引用 Range("A1:A6")，但是仍然需要重复地输入对象变量的名字。VBA提供了一种避免这样的方式——With语句。With语句用于设置同一对象的多个属性，或者执行同一对象的多个方法。程序清单 10-3 中显示了使用 With语句完成同样功能的代码。

程序清单 10-3 With语句的例子

```

1: With Range ("A1:A6")
2:     .NumberFormat = "#, ##0.00"
3:     With.Font
4:         .Name = "Courier New"
5:         .FontStyle = "Regular"
6:         .Size = 11
7:         .Strikethrough = False
8:         .Superscript = False
9:         .Subscript = False
10:        .OutlineFont = False
11:        .Shadow = False
12:        .Underline = xlUnderlineStyleNone
13:        .ColorIndex = xlAutomatic
14:    End With
15: End With

```

虽然程序清单中的 With语句并没有减少代码的行数，但是减少了输入量。 With语句的语法如下：

```

With object
[statements]
End With

```

语法中的 object是 statements部分中的属性和方法的操作对象。从程序清单 10-3 中可以看出，statements中的每一行都以符号“.”开始。同样需要注意的是，可以将 With语句嵌套使用。在程序清单 10-3 中，针对 Font对象的 With语句嵌套在针对 Range对象的 With语句中。程序清单 10-4 举例说明了如何在同一个 With语句中使用属性和方法。

程序清单 10-4 包含属性和方法的 With语句

```

1: Sub WithWorksheet ()
2:     With ThisWorkbook
3:         .SaveAs "WithExample"
4:         MsgBox "Save Status: "&.Saved
5:     End With
6: End Sub

```

10.3 使用For Each语句

With语句是用来执行同一个对象的多个语句，而 For Each语句则是用来对多个对象执行同一条语句。通过 For Each语句，可以对集合中的每个元素重复执行同样的语句。

For Each语句也可用于数组。

For Each语句的语法如下：

For Each element In group

```
[Statements]
[Exit For]
[Statements]
```

Next

注意，该语法支持 Exit For 从句。和其他 Exit 从句一样，Exit For 从句通常位于一条 If 语句中。如果需要改变 Range 中每个单元格的数值，可以使用程序清单 10-5 中的代码。

程序清单 10-5 For Each 语句的例子

```
1: Sub ForExample ()
2:   Dim x As Range
3:
4:   For Each x In ThisWorkbook.Worksheets("Sheet1").Range("A1:A6")
5:     x.Value = x.Value + 10
6:   Next
7: End Sub
```

10.4 Range 对象编程示例

在本部分中，你将看到多个使用 Range 对象的代码示例。这些代码示例都经过了精挑细选，是程序员最常用的。

第一个代码示例是改变一个 Range 中每行的格式设置。对于一张很长的工作表，为了使它易于查看和打印，常常需要这样做。在本例中，假设已经有了一张和图 10-2 相似的工作表，注意第一行是标题。需要将从第三行开始的内容隔行加粗。完成这一任务的过程请参看程序清单 10-6。

图 10-2 将用 BoldEveryOther

过程对本工作表进行
处理

	A	B	C	D	E	F	G	H	I	J
1	Last Name	First Name	Region	Sales						
2	Baker	Bob	South	10000						
3	Smith	Judy	South	25500						
4	Taylor	Alice	North	32500						
5	Wilson	Tom	North	75250						
6	Anders	John	West	10500						
7	Jones	Mike	West	175520						
8	Carter	Jim	East	98500						
9	Davis	Roger	East	111250						
10	Donner	Nancy	South	35450						
11	Felix	Bill	North	12350						
12	Grange	Diane	South	65500						
13	Lane	Ann	East	78500						
14	William	Barbara	East	25350						
15	Moss	Brent	West	45550						
16	Rogers	Larry	North	65450						
17	Kane	Mark	West	14550						
18	Kjames	Cathy	South	22250						
19	Brady	Sally	North	65950						
20	Redmond	Sandra	East	73350						

程序清单 10-6 使用 Row 对象和 For Next 语句

```
1: Sub BoldEveryOther ()
2:   Dim iCounter As Integer
```

```

3:
4:     For iCounter = 3 To ThisWorkbook.Worksheets ("Sheet1")._
Range("A1:C25").Rows.Count Step 2
5:         ThisWorkbook.Worksheets("Sheet1").Range("A1:C25")._
Rows(iCounter).Font.Bold = True
6:     Next
7:
8: End Sub

```

本过程中主要的关键在于For Next语句。注意iCounter设置的初始值为3，即从第三行开始隔行加粗。Step的值为2决定了是隔行加粗。图10-3显示了该过程运行后的工作表。

**图10-3 BoldEveryOther过
程运行后，Range
对象具有了理想的
格式**

The screenshot shows a Microsoft Excel spreadsheet titled 'Microsoft Excel - Book3'. The data is organized into four columns: Last Name, First Name, Region, and Sales. The rows are numbered from 1 to 20. Rows 1 through 4 are regular, while rows 5, 7, 9, 13, and 15 are bolded. The bolded rows correspond to the odd-numbered rows in the original code. The Excel interface includes a toolbar at the top, a ribbon menu, and a formula bar.

	A	B	C	D	E	F	G	H	I	J
1	Last Name	First Name	Region	Sales						
2	Baker	Bob	South	10000						
3	Smith	Judy	South	25500						
4	Taylor	Alice	North	32500						
5	Wilson	Tom	North	75250						
6	Anders	John	West	10500						
7	Jones	Mike	West	175520						
8	Carter	Jim	East	98500						
9	Davis	Roger	East	111250						
10	Donner	Nancy	South	35450						
11	Felix	Bill	North	12350						
12	Grange	Diane	South	65500						
13	Lane	Ann	East	78500						
14	William	Barbara	East	25350						
15	Moss	Brent	West	45550						
16	Rogers	Larry	North	65450						
17	Kane	Mark	West	14550						
18	Kjames	Cathy	South	22250						
19	Brady	Sally	North	65950						
20	Redmond	Sandra	East	73350						

你也许已经意识到，如果知道需要处理的Range对象的地址，那么本例中的代码非常适用。但是如果不知道Range对象的大小时，又应该怎么办呢？有关这方面的一个经典的例子是，当从其他应用程序（例如数据库）向工作表中导入数据时的情况，你也许并不知道会返回多少行数据。程序清单10-7中的代码显示了选中一个不知道大小的Range的例子。

程序清单10-7 选中一个不知道大小的Range

```

1: Sub SelectRange ()
2:     ThisWorkbook.Worksheets("Sheet1").Range("A1").Activate
3:     ActiveCell.CurrentRegion.Select
4:     MsgBox "The address of the selected range "& Selection.Address
5: End Sub

```

SelectRange的关键之处在于CurrentRegion属性的使用，本属性返回第一个空行和空列所包围的区域。通过选中CurrentRegion，用不着知道Range的大小。

最后一个过程将执行复制/粘贴操作。在程序清单10-8的代码中，对选中的区域进行复制，并且将复制的内容粘贴到工作表的另一位置。

程序清单 10-8 复制和粘贴一个区域

```
1: Sub CopyAndPaste ()  
2:     Selection.Copy  
3:     Range("F3").Select  
4:     ActiveSheet.Paste  
5:     Application.CutCopyMode = False  
6: End Sub
```

语句 Selection.Copy 将选中的区域放到剪贴板上。下一步是移动到粘贴所复制的内容的理想位置。接着用 Paste 方法将所复制的区域从剪贴板复制到新的位置上。最后，将 Application 对象的 CutCopyMode 属性设置为 False。如果不这样做，源文本周围的选取框（移动的虚线）将一直保持，并且状态栏上将一直显示移动目标的指令。

10.5 学时小结

到目前为止，你对 Range 对象的处理应该是得心应手了，因为你已经在多个过程中练习了 Range 对象的几个属性和方法。

你还学习了两个 VBA 的结构，通过它们可以使对象的处理更加简化。 With 语句用于为同一对象设置多个属性或者执行多个方法，而 For Each 语句则为多个对象执行同样的语句。

在本学时的最后一部分，学习了几个对 Range 对象进行处理的过程，这些过程为 Range 对象的用法提供了示例。

10.6 专家答疑

问题：为什么没有 Range 集合？

解答：集合的一个必要条件是可以添加元素。工作表上的单元格（也就是 range）的数目是由 Microsoft 预先定义的，没有办法向工作表添加更多的单元格，正因为如此，没有 Range 集合。

问题：For 语句和 For Each 语句有什么不同？

解答：首先，讨论一下这两条语句的共同点：它们都是多次执行相同的语句。而问题在于，它们有什么不同？对于 For 语句而言，语句执行的次数是由一个起始数字和一个结束数字所控制，而 For Each 语句中语句执行的次数是由所操作的对象的数目或者数组的元素的数目所控制。

10.7 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

10.7.1 思考题

- 1) 判断题：在 VBA 中 Range（区域）总是指多个单元格。
- 2) 根据一个 Range 对象的地址访问另一个 Range 对象时，要用到 Range 对象的哪一个属性？
- 3) 如果需要增加 Range 对象中每个单元格的数值，使用什么语句可以用最少的代码实现？
- 4) 什么属性可以根据一个 Range 对象的位置选中另一个未知的 Range 对象？
- 5) 怎样才能知道一个 Range 对象中有多少个单元格？

6) 删除Range对象中的内容需要使用什么属性?

7) 你必须设置同一对象的多个属性,完成这样的任务的最有效方式是什么?

10.7.2 练习题

打开一个新的工作簿,在工作簿的第一张工作表中输入如下数据:

单元格	数值	单元格	数值
A1	Item	B1	Price
A2	Widget	B2	15
A3	Gidget	B3	5
A4	Gizmo	B4	3
A5	Junk	B5	7
A6	Stuff	B6	9

创建一个名为ReducePrices的过程。本过程必须将Range(B2:B6)中每个单元格的数值减5。如果价格小于或者等于0,将价格和项目名用红色粗体字显示。如果Range中单元格的数值有小于或者等于0的,则显示一个消息框,让用户在结束过程前知道出了问题。运行并测试该过程。

第11学时 使用Visual Basic编辑器

现在，你对VBA语言的语法已经比较熟悉了，但是对 Visual Basic编辑器的某些特征可能还比较好奇。Visual Basic编辑器有多种多样的特征，这些特征使得编写代码更容易、速度更快。在本学时中，将学习这些特征，包括工具栏、在线帮助和对象浏览器。

本学时的重点包括：

- 使用Visual Basic编辑器的工具栏
- 浏览代码的提示
- 充分利用在线帮助
- 对象浏览器的使用
- 设置Visual Basic编辑器的选项

11.1 使用Visual Basic编辑器的工具栏

到目前为止，你仅仅使用了 Visual Basic编辑器的标准工具栏上的几个按钮。该工具栏上包括最常用的执行命令的按钮。当你第一次看到工具栏时也许并没有意识到，工具栏按照命令的类型进行了分割。

按钮的第一部分（如图 11-1所示）是和工作簿有关的，使用它们可以返回 Excel，可以向当前工程中添加项目，可以保存你的工作。

图11-1 Visual Basic编辑器的标准工具栏

上的“视图 Microsoft Excel”、
“插入”、“保存”按钮



下一部分按钮（如图 11-2所示）和编辑功能有关。使用这些按钮可以剪切、复制和粘贴文本。在本部分中还包括一个查找文本的按钮。

图11-2 Visual Basic编辑器的标准工具

栏上的“剪切”、“复制”、“粘贴”
和“查找”按钮



工具栏的第三部分（如图 11-3所示）包括两个按钮，“撤销删除”和“重复删除”。当进行了错误操作时可能会用到。

图11-3 Visual Basic编辑器的标准工具

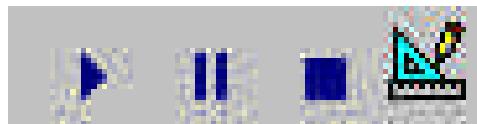
栏上的“撤销删除”和“重复
删除”按钮



接下来，是称为测试和设计按钮的部分，如图 11-4所示。这部分的前三个按钮允许运行、暂停和终止过程的执行，最后一个按钮将用户窗体转到设计模式。

图11-4 Visual Basic编辑器的标准工具

栏上的“运行”、“暂停”、“终止”和“设计”按钮



Visual Basic编辑器的标准工具栏上的下一部分按钮（如图 11-5所示）是用于查看 Visual Basic 编辑器的不同部分。通过这些按钮可以显示“工程资源管理器”、“属性”窗口、“对象浏览器”和工具栏。

图11-5 Visual Basic编辑器的标准工

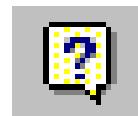
具栏上的“工程资源管理器”、“属性窗口”、“对象浏览器”和“工具箱”按钮



工具栏上的最后一个按钮是“帮助”按钮，如图 11-6所示。本“帮助”按钮和所有 Microsoft 应用程序中的帮助按钮的作用是一样的。

图11-6 Visual Basic编辑器的标准工具

栏上的“帮助”按钮



在使用 Visual Basic 编辑器时，另一个有用的工具栏是“编辑”工具栏。该工具栏上的工具可以影响和增强 Visual Basic 编辑器环境的功能。“编辑”工具栏上的按钮包括：

- 属性/方法列表 单击本按钮将显示一个属性和方法的列表。
- 常数列表 显示可用于当前常数的系统常数。
- 快速信息 显示提供语法信息的快速信息框。
- 参数信息 显示可用的参数信息。
- 自动完成关键字 可激活VBA的自动完成关键字功能，这样可以只输入保留字的一部分而让编辑器输入剩余的部分。
- 缩进 将选中的文本缩进显示。
- 凸出 将选中的文本凸出显示。
- 切换断点 在所选代码行设置断点。

有关断点的更多的信息，可参看第 12 学时。

- 设置注释 将选中的文本转变为注释。在应用程序开发的调试和测试阶段，可用这种方式跳过某部分代码的执行。
- 解除注释 将选中的文本转变为代码。
- 切换书签 在当前行标注书签。书签是为了方便引用而希望标注的代码行。
- 下一书签 移动到下一个书签。
- 上一书签 移动到前一个书签。

-  • 清除所有书签 从文本中删除所有的书签。

11.2 浏览代码

用Visual Basic 编辑器进行工作有一个很好的特征，即如果你对 Microsoft Word的导航键比较熟悉，则可以加以利用。表11-1中列出了部分导航键。

如果需要寻找一个特定的单词或者词组，可以使用 Visual Basic编辑器的“查找”功能。在“编辑”菜单中，可以找到“查找”命令，也可以单击标准工具栏上的“查找”按钮，或者按下Ctrl+F键。“替换”是对查找功能的扩展，“替换”是用于找到一个单词或者词组的位置，并将其替换为别的文本。

表11-1 导航键

要 到	按 此 键
行的开始	Home
行的末尾	End
模块的开始	Ctrl+Home
模块的末尾	Ctrl+End
下一个词	Ctrl+
上一个词	Ctrl+
下一个过程	Ctrl+
上一个过程	Ctrl+

按下F3键可以执行替换命令。

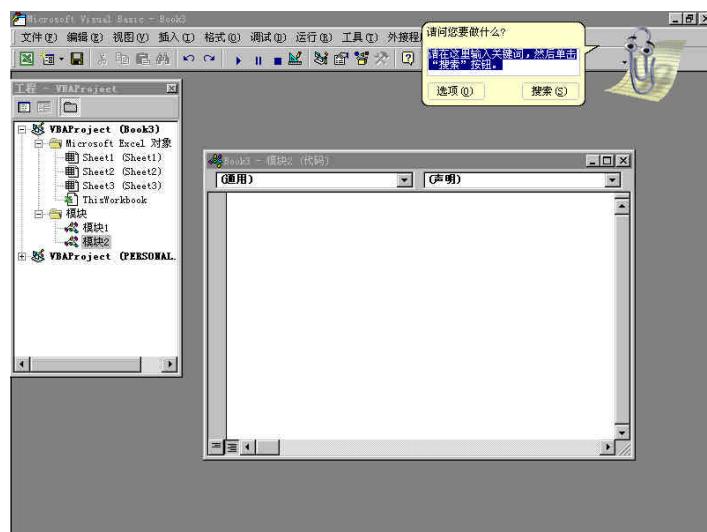
编辑器的另一项浏览功能是书签。通过书签，可以对文本进行标注以便查找，当对代码进行调试和测试时非常有用。如果估计代码中有较多的错误，可用书签对它们进行标注。也可用书签来标注需要由同组的其他开发人员添加的代码区域，或者用书签标注需要更新或升级的代码区域。

11.3 使用在线帮助

你已经使用了VBA帮助的两项功能。现在请花一点时间，按照如下步骤操作，以便更熟悉VBA的帮助特征：

1) 选择“帮助”、“Microsoft Visual Basic帮助”。启动Excel助手并提示输入一个问题，如图11-7所示。

图11-7 Excel助手的形状多种多样，它是Visual Basic帮助的初始界面



- 2) 输入问题“如何创建新工作簿”并按回车键。
- 3) 选择“创建新工作簿”，显示“创建新工作簿”主题，如图 11-8所示。

图11-8 从Excel助手中进行

选择可引导你选择
合适的帮助主题



Visual Basic的帮助提供多个代码的示例，可以从帮助中复制这些示例代码，并粘贴到自己的过程中，具体步骤如下：

1) 单击“Add”(位于主题的顶部并以下划线显示)。带下划线的文本链接到其他主题，在本例中，链接到Add方法。

2) 单击位于帮助主题顶部的“Example”，显示示例代码，如图11-9所示。

图11-9 示例代码是帮助学习
的非常有用的工具



3) 突出显示代码行“Workbooks.Add”。

4) 用鼠标右键单击突出显示的代码行，从显示的菜单中选择“复制”。

5) 关闭帮助窗口。

6) 在当前工作簿中添加一个新的模块。

7) 创建一个新的名为“NewBook”的过程。

8) 将示例代码粘贴到新的过程中，这也是一个非常节省时间的工具。

尽管在本练习中所用到的示例代码不够复杂，但是应当理解这种方法。如果在帮助中找到需要的代码，则应当使用它！

上下文相关帮助是VBA提供的最好的帮助功能。在编辑器中输入希望得到帮助的对象、属性、方法或者函数的名字，并按下F1键，就可以激活合适的帮助主题。

还有几种功能你也许并不会立即认为属于帮助功能，这些功能包括：

- 自动列出成员 本功能可显示一个列表，其中包括完成当前输入的语句逻辑上还需要的部分有关信息。
- 自动显示快速信息 在输入函数时显示一个小的弹出式消息框，其中包括输入的函数及其参数的信息。
- 自动显示数据提示 在一个小的消息框中显示当前光标所在处变量的数值，只有在中断模式下本功能才有用。

中断模式将在第12学时中加以讨论。

11.4 使用对象浏览器

对象浏览器为浏览工程中所有的可用对象提供了一个易于使用的界面。我们可以查看对象的属性、方法和事件，也可以查看工程所引用的对象库中可用的所有过程和常数。使用对象浏览器有几种方式：

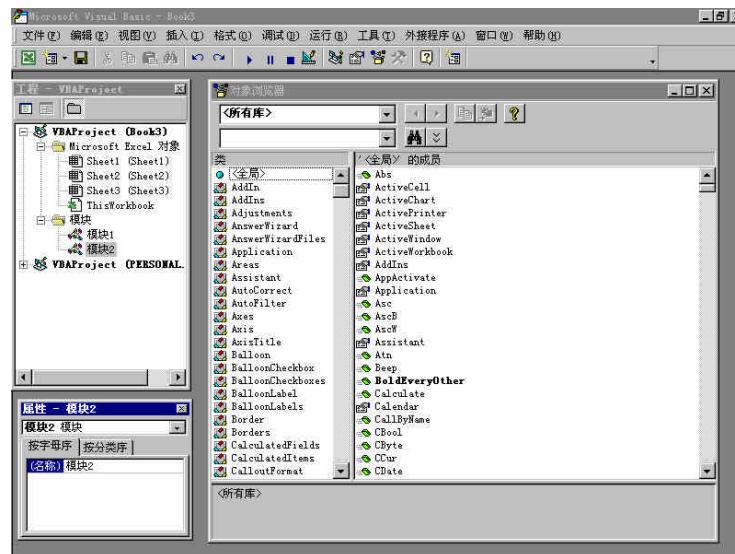
- 单击标准工具栏上的“对象浏览器”按钮。
- 选择“视图”、“对象浏览器”。
- 按下F2键。

通过对象浏览器，可以很容易熟悉对象的属性和方法，也可以通过对象浏览器获取帮助。

为了熟悉对象浏览器的使用，请完成如下步骤：

- 1) 按下F2键显示对象浏览器，如图11-10所示。

图11-10 对象浏览器分类显示对象、属性、方法和事件



- 2) 找到Range对象并选中它。

- 3) 滚动 Range 对象可用的方法（在对象浏览器中以函数的方式引用）和属性。
- 4) 选择 Activate。
- 5) 按下 F1 键。显示“Activate 帮助”主题。
- 6) 关闭帮助窗口。
- 7) 关闭对象浏览器。

11.5 设置编辑器选项

通过“选项”对话框可以对 Visual Basic 编辑器环境进行自定义。通过该对话框，可以控制编辑器的设置、编辑器的格式、常规设置和可连接设置。为了熟悉对“选项”对话框进行的设置，请完成如下步骤：

- 1) 选择“工具”、“选项”，显示“选项”对话框。
- 2) 选择“编辑器”选项卡。“编辑器”选项卡用于对代码和窗口的设置，从其中可以打开和关闭诸如“自动显示快速信息”和“自动语法检测”等选项。
- 3) 选择“编辑器格式”选项卡。“编辑器格式”选项卡可控制编辑器中不同类型文本的颜色，也可以选择不同的字体风格和字体大小。如果你是在笔记本电脑上工作，也许希望改变编辑器使用的某些颜色，因为在较旧的笔记本电脑上蓝色和黑色的区别不明显。如果你的视力不够好，可以将显示的文本放大。
- 4) 选择“通用”选项卡。本选项卡包括影响整个编辑器环境的设置。“窗体网格设置”是用来控制设计用户窗体时是否显示网格的，也可以控制“编辑并继续”、“错误捕获”和“编译”部分的设置。
- 5) 选择“可连接的”选项卡。该选项卡用于控制 Visual Basic 编辑器中各个部分的可连接特性。
- 6) 单击“取消”按钮，关闭对话框。

11.6 学时小结

现在，你已经花了几个小时的时间进行 VBA 代码的编写，已经可以更好地理解本学时中讲述的工具和各种功能了。正如你现在所知道的那样，在线帮助系统不仅解释语句的语法，还提供各种实际的示例。你也对查看对象、属性、方法和事件的非常好用的工具——对象浏览器有了一定的了解。

有时哪怕很小的细节都可以给你带来很大的方便。通过导航键、工具栏和选项，可以对工作所用的 Visual Basic 编辑器环境进行控制。

11.7 专家答疑

问题：如果我不知道要查找的属性的名字，应当怎样使用帮助系统？

解答：完全没有问题！你可以在帮助中查找所处理的对象，接着在它的所有属性中查找。也可以在对象浏览器中找到该对象，并从中查找合适的属性。

问题：我不喜欢编辑器所用的 Courier 字体，难道我只能使用这种字体吗？

解答：不是，通过选择“工具”、“选项”，可以显示“选项”对话框，从中选择“格式”选项卡，并选择其他字体。

11.8 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

11.8.1 思考题

- 1) 举出访问在线帮助系统的三种方法。
- 2) 对象浏览器是列举对象、事件、属性和_____的非常好用的工具。
- 3) 要跳到模块的开始，需要按什么键？
- 4) _____是一个小的弹出事对话框，当输入函数时，显示与该函数及其参数有关的信息。
- 5) 在何处可以对诸如“自动显示快速信息”和“自动语法检测”功能的使用进行控制？
- 6) 判断题：在线帮助系统所提供的例子只能查看，即不能对它们进行复制并放置到自己的代码中。

11.8.2 练习题

使用任何你愿意采用的工具（帮助系统或者对象浏览器），找到如下信息：

用来执行拼写检查的方法是（提示：查看 Application对象）：_____。

包含Excel安装位置的属性是_____。

标识一个工作簿是否已经保存的属性是_____。

强制进行手动计算的方法是_____。

用来隐藏（使之不可见）工作表的属性是_____。

用来清空Range对象内容的方法是_____。

第12学时 调试VBA代码

以前用宏进行开发时，调试过程可以描述为运行宏。如果运行失败，则估计错误出现在何处。现在则大不相同。Visual Basic编辑器提供了具有多种功能的开发环境，其中包括许多与更高级的开发工具中相同的调试工具。

本学时的重点包括：

- 理解对应用程序的调试和测试的意义
- 暂停应用程序的执行
- 使用立即窗口与应用程序进行交互
- 逐行执行过程
- 使用Watches跟踪变量和属性的数值

12.1 应用程序开发的测试和调试阶段

新术语 测试就是当编写完过程之后，运行该过程，并试图猜测用户可能对该过程做出的各种响应的阶段。例如，在测试过程中，应当输入各种各样的数值，单击不同的按钮，并且做出各种不同的选择。测试的要点在于确保出现预期的结果。你也应当查找是否出现了意料之外的结果，例如某个计算是否返回了错误的结果。你的目标是判断是否输入一个特定的数值，或者执行某个特定的操作会导致过程异常终止。当某个过程出现意料之外的结果或者出现非理想的情况时，我们称之为错误。

作为程序员，你可以采用各种各样的方法尽可能地减少过程中这些类型的错误。每个学习编程的人都希望获得突飞猛进的进步，都希望立即开始实际的编程。但是如果花一部分时间进行仔细的设计，并对应用程序的目标和要求进行详细的分析和记录，可以消除应用程序中可能出现的一些问题。通过对应用程序的目标和功能的理解，你对应用程序会有一个清晰的整体印象。如果你是为顾客开发应用程序，而不是为自己开发，这就尤为重要。显然，你不希望将时间浪费在开发顾客并未做出要求的功能上。在开始开发应用程序之前先花一点时间对应用程序进行分析，最终可以节省大量的时间。

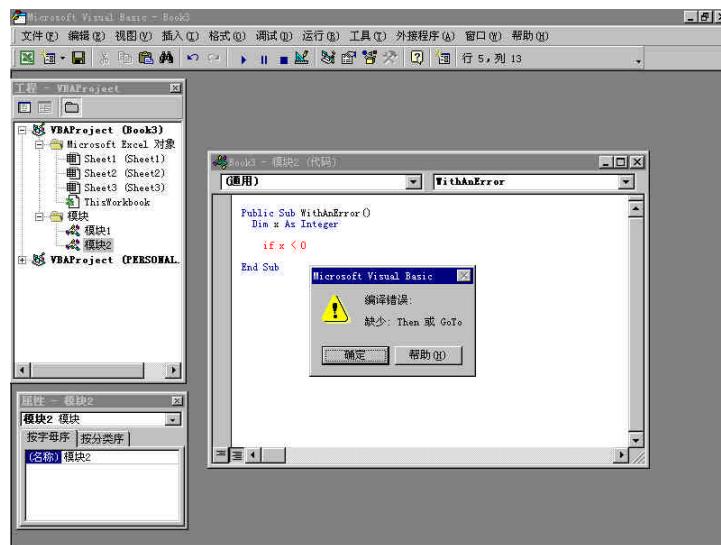
另一个可以采用的工具是命名约定。对变量和对象采用固定的命名约定不仅有助于调试应用程序，还使应用程序更加易于维护。在本书中，我们讲述了各种命名约定。

有关对控件的命名约定请参见第3学时，有关变量和常量的命名约定请参阅第4学时。

当谈论对代码的支持和维护时，不要忘了对代码进行注释。通过对代码进行注释，当编写完代码一段时间后，你对代码的理解会更为容易。你也许想知道怎样决定注释的内容。对此，推荐的做法是：假设你不是将来对应用程序进行支持和维护的人员，添加你作为应用程序的维护人员而不是编写人员时所希望看到的那些信息。

当完成本书中的某些练习后，你也许偶然地出现过输入错误。如果存在这种情况，当你按下回车键时会显示一个消息框。消息框会显示出现了语法错误，如图 12-1 所示。这是因为在默认设置下，Visual Basic 编辑器在用户按下回车键时自动检查每个语句是否存在语法错误。通过“选项”对话框中的“编辑器”选项卡可以打开或者关闭“自动语法检查”功能。

图12-1 编辑器会随时向用户反馈语法错误



新术语 可能遇到的另一类型的错误是运行错误。运行错误是指造成应用程序停止运行的任何错误。有时这是由于程序员犯了错误，例如，你可能在过程中输入了错误的对象名字。VBA不会检查这种类型的错误，除非运行该过程。运行错误也可能是用户的操作所引起的，而用户的操作不是你所能控制和预测的。例如，如果用户不给你编写的函数提供数据，就有可能产生运行错误。

最后一种可能出现的错误是逻辑错误。Visual Basic 编辑器不会为你指出这种类型的错误，因为这种错误在 VBA 语言的语句上没有任何错误。问题在于代码的执行结果和预期的结果不同，这就意味着语句的逻辑出了问题。以后，你会发现大部分的调试时间都用在对逻辑错误的处理上。

12.2 调试

新术语 查找错误的过程就称为调试。因为 VBA 是一个功能强大的开发环境，所以提供了几种调试工具供使用。这些工具包括：

- 立即窗口。
- Watch 表达式。
- 断点。
- 单步执行代码。

在开始进行调试之前，需要先编写部分代码。请按照如下步骤创建一个包含一个错误

的过程：

- 1) 关闭所有已经打开的工作簿。
- 2) 打开一个新的工作簿。
- 3) 按下 Alt+F11 键，打开 Visual Basic 编辑器。
- 4) 在工作簿中插入一个新的模块。
- 5) 创建一个新的名为 Buggy 的过程。
- 6) 为该过程输入如下代码：

```
Dim response
```

```
response = Application.InputBox("Enter your name: ")  
If response = " " Then  
    MsgBox "Procedure cancelled."  
    Exit Sub  
Else  
    MsgBox "Your name is " & response  
End If
```

7) 运行该过程。当显示提示信息时，输入你的名字并按回车键。显示一个消息框，其中包括你的名字。到目前为止，一切似乎都和预期的一样。

- 8) 单击“确定”按钮，退出消息框。
- 9) 按下 F5 键再次运行该过程。
- 10) 当显示输入框时，单击“取消”按钮进行响应。一个消息框显示你的名字是“False”！而你预期的是显示一个表示过程被取消的消息框。
- 11) 单击“确定”按钮，退出消息框。

这个过程运行的结果和预期的不同。你也许已经猜到了错误出在何处，但是现在不要改正这个错误。在下面的学习中，将使用该过程练习调试技术。

12.3 将过程设置为中断模式

新术语 当对过程进行调试时，需要对过程中的每一行代码进行处理。为了便于将错误的区域分离出来，VBA 提供了多种方式，以便在某个特定的地方暂停过程的执行，称为将过程设置为中断模式。中断模式暂时挂起过程的执行。通过将某个过程设置为中断模式，可以检查当前变量和属性的数值，也可以运用其他调试技术。当过程处于中断模式时，可以执行如下操作：

- 编辑应用程序的代码。
- 查看变量、属性的数值和语句。
- 为变量和属性输入不同的数值。
- 通过使用立即窗口运行 Visual Basic 语句。

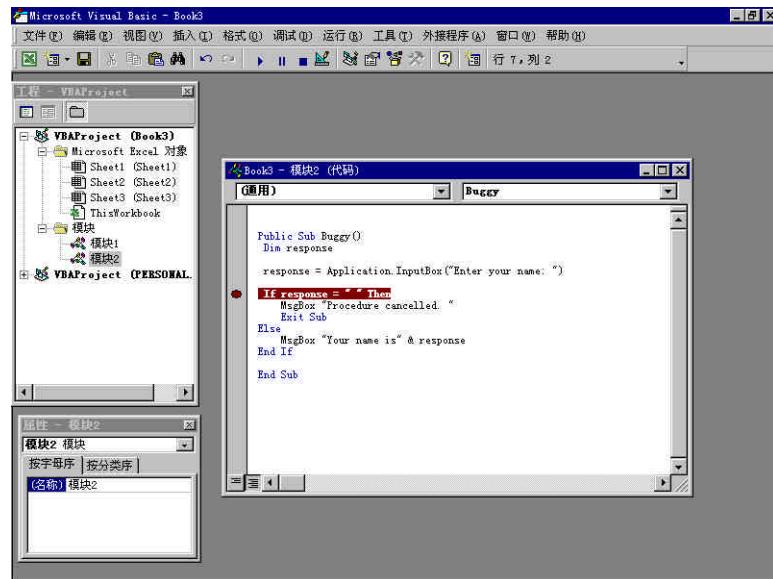
新术语 在 Buggy 过程中，你知道错误出现在 InputBox 语句后面的某处。这样，包含 InputBox 语句的代码行就很适合用来暂停过程。在特定的位置暂停过程的方式称为设置断点。将某行代码设置为断点有以下几种方法：

- 选择要设置为断点的语句，再选择从菜单中选择“调试”、“断点开关”。
- 选择要设置为断点的语句，再按下F9键。
- 选择要设置为断点的语句，再单击“代码”窗口边缘的指示器栏。

只能将断点设置在可执行的代码行，而不能将断点设置在不可执行的代码行，其中包括注释、变量和常数的声明语句以及空行。如果某行代码已经设置为断点，该行代码的颜色会改变，具体变为什么颜色取决于“编辑器格式”当前的设置。此外，该行边缘的指示器栏还显示一个圆点，如图12-2所示。

图12-2 代码窗口中的断点

很容易辨认



上面讲述的三种操作也可以用来删除断点，因为断点是一个开关项。如果在过程中设置了多个断点并希望将它们全部删除，可以选择“调试”、“清除全部断点”，或者按下Ctrl+Shift+F9键。

现在，我们需要在Buggy过程中使用InputBox语句的代码行后设置断点，具体步骤如下：

- 1) 将鼠标指向使用InputBox语句的代码行下一行中的任何位置。
- 2) 单击“切换断点”按钮（位于“编辑”工具栏或者“调试”工具栏）或者按下F9键。可以看到该行代码的颜色变了，并且边缘的指示器栏显示一个圆点。

设置完断点后，将使用其他调试工具来解决过程中的错误。顺便指出，过程中可以设置多个断点，只要需要，可以设置任意数目的断点。现在运行该过程，并且观察设置断点后产生什么影响，具体步骤如下：

- 1) 运行Buggy过程。
- 2) 在输入框中输入你的名字并按回车键。显示Visual Basic编辑器，并且突出显示当前的代码行，如图12-3所示。看一看Visual Basic编辑器窗口的标题栏，会发现标题栏显示“中断”，这样便于知道当前处于中断模式。

现在过程处于中断模式，可以查看变量response的数值，以确定是否和预期的一样。方法是将鼠标移动到相应的变量名上，弹出一个小的消息框，其中包括变量的当前值，如图12-4

所示。你可以看到变量的数值和预期的相同，要找到错误，还需要进一步调试。

图12-3 当运行到断点时，
Visual Basic 编辑器显示相应的代码

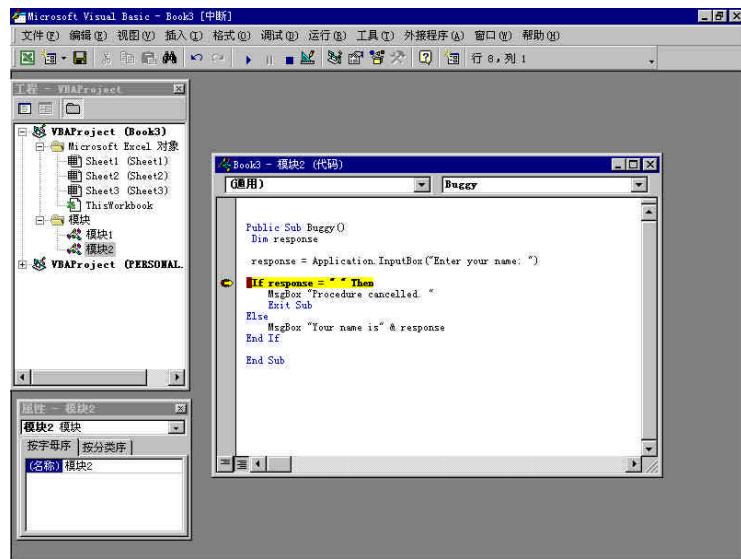
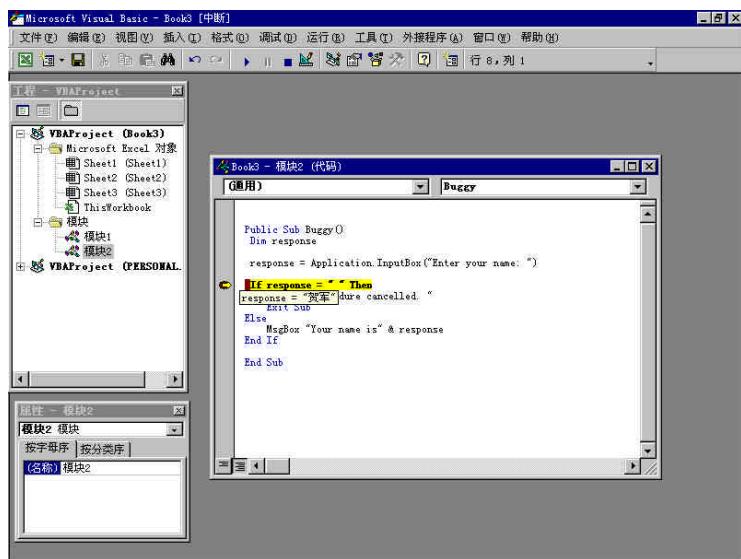


图12-4 Visual Basic 编辑器即时反馈变量的数值或者属性设置



要确认变量或者属性的数值还有其他方法。在本学时的下一部分，将学习使用立即窗口，通过它可以直接和变量以及属性设置进行交互。在本学时的后面部分，还将学习使用 watch 来跟踪变量和属性设置。

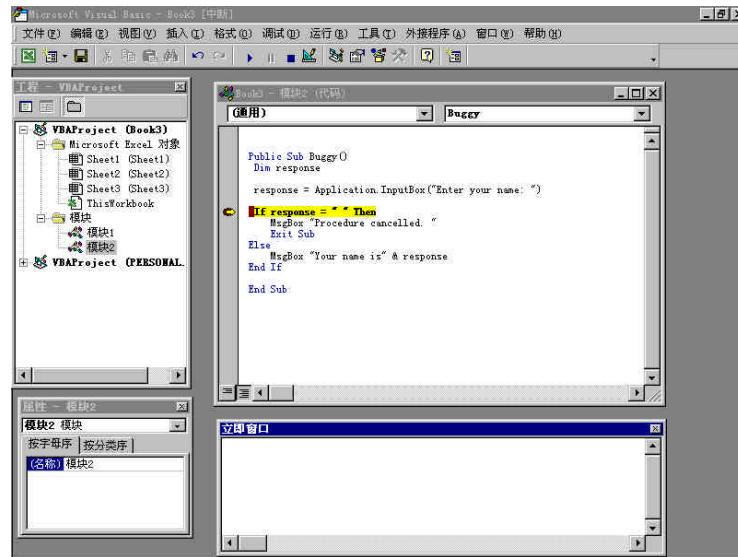
12.4 使用立即窗口

在图12-5的底部，就可以看到立即窗口。通过立即窗口，可以进入过程属性以及变量数值的世界。立即窗口允许完成多种任务，具体包括：

- 当过程运行时打印变量和属性的数值。

- 当过程运行时改变变量和属性的数值。
- 当应用程序运行时查看调试输出。
- 和在程序代码中一样调用过程。
- 输入新的语句并立即执行，这样可以试验不同的代码。
- 将已有的代码行复制和粘贴到立即窗口中并加以执行。

图12-5 立即窗口可能出现的次数不多，但它的确是一个非常有用的调试工具



在立即窗口中显示数值

和你早些时候看到的那样，当运行一个应用程序时，在中断模式下，通过将鼠标指向变量名就可以知道该变量的数值。另一种知道变量（或者属性）数值的方法是使用立即窗口。现在将要完成的是在立即窗口中显示希望得到的数值。在立即窗口中显示数值共有三种方法。可以在立即窗口中使用 Print方法；也可以在立即窗口中使用“？”、“？”实际上是Print方法的速记形式；另一种在立即窗口中显示数值的方法是直接在过程中添加Debug.Print语句。

通过Print方法（或者“？”），可以用另一种方式测试变量和属性的数值。要在立即窗口中打印Response变量的数值，请按照如下步骤：

1) 如果过程处于中断模式，可通过“运行”菜单或者单击“重新设置”工具栏按钮来重新设置过程。

2) 按F5键运行应用程序，对于显示出来的输入框，单击“取消”按钮。

3) 返回到Visual Basic编辑器，如果没有看到立即窗口，请按Ctrl+G键，这样就会显示出立即窗口。通过“视图”菜单也可以显示立即窗口。

4) 为了在立即窗口中打印Response的数值，请单击立即窗口以激活这个窗口。输入如下代码行：

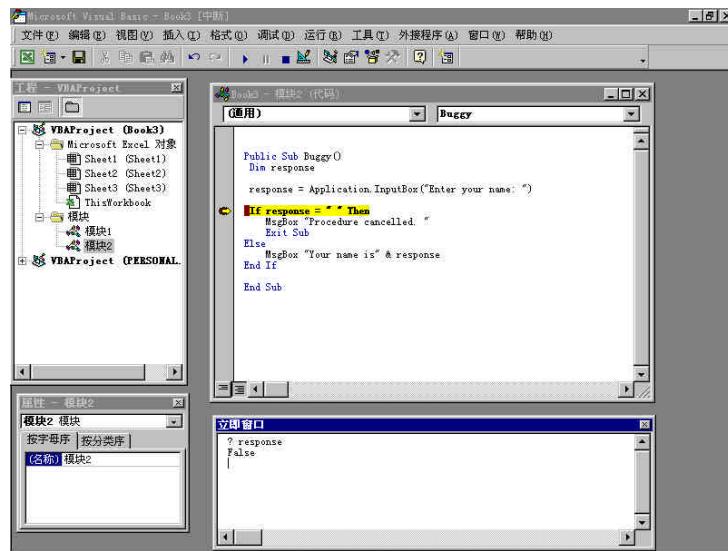
?Response

也可以输入“Print Response”，这和“?Response”是等价的。

如果你希望执行以前在立即窗口中输入的语句，只需将鼠标移动到相应语句的位置并按回车键。

5) 按回车键，Response的数值显示在立即窗口中的下一行。本例中，你看到的数值是“False”，如图12-6所示。

图12-6 立即窗口是另一种
显示数值的工具



6) 在本学时的早些时候，你知道了可以用立即窗口为变量或者属性输入某个数值。此时，你可能希望测试过程的逻辑。因为你的过程中需要的是空字符串，下面可以看一看如果提供一个空字符串会出现什么情况，请在立即窗口中输入如下代码：

```
Response = ""
```

7) 按回车键，将变量Response的数值重置为空字符串。现在你应当知道问题出在何处了。

8) 选择“运行”、“重新设置”结束菜单的执行。

这种方法的一个缺点在于，必须先找到应当在过程执行到何处时才使用立即窗口。在前面的练习中，具体做法是通过设置断点。如果不希望暂停过程的执行，同时又希望知道变量或者属性的数值，那么可以使用 Debug.Print语句。直接将这条语句添加到过程中，该语句会将文本发送到立即窗口。可以在运行完应用程序后再查看输出的调试文本。要在 Buggy过程中使用Debug.Print语句，请完成如下步骤：

- 1) 选择“调试”、“清除全部断点”，删除当前设置的全部断点。
- 2) 删除立即窗口中的文本内容。
- 3) 修改过程 Buggy，使得该过程和下面的代码一致（修改处显示为粗体）：

```
Public Sub Buggy()
Dim response

response = Application.InputBox("Enter your name: ")
Debug.Print "Value of response is " & response
If response = "" Then
    MsgBox "Procedure cancelled."
```

```

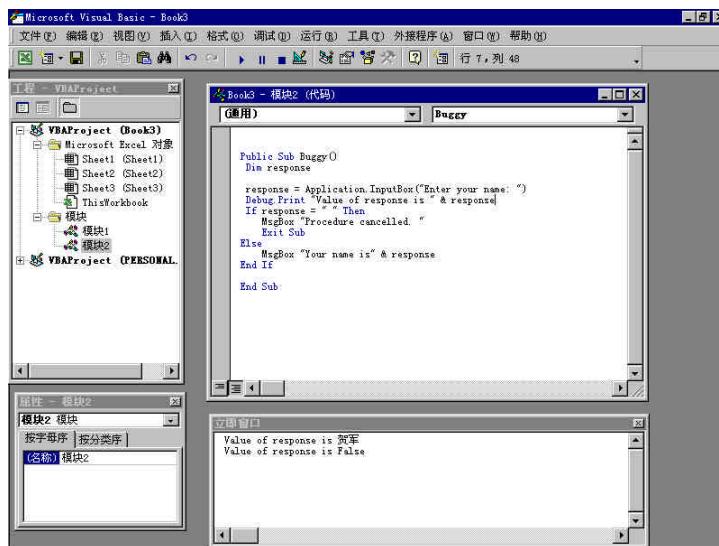
    Exit Sub
Else
    MsgBox "Your name is" & response
End If
End Sub

```

- 4) 按F5键运行该过程。
- 5) 在显示出来的输入框中输入你的名字并按回车键。
- 6) 单击“确定”按钮，退出消息框。
- 7) 再次运行Buggy过程。
- 8) 对显示出来的输入框单击“取消”按钮，再单击“确定”按钮退出消息框。
- 9) 返回Visual Basic编辑器。

当查看Visual Basic编辑器的内容时，在立即窗口中会看到两行文本，如图 12-7所示。第一行文本是第一次运行过程时创建的，第二行文本是第二次运行过程时创建的。现在，可以看到使用Debug.Print语句的优点是，在过程运行完毕后，可以得到可见的结果。

图12-7 Debug.Print在立即窗口中显示某个变量(或者属性)的历史值



12.5 单步执行代码

在大多数情况下，都需要调试过程的逻辑和流程。在这种情况下，知道过程中代码行的执行顺序很有帮助。当对If语句和Select语句进行调试时更是特别有用。要做到这样，可以使用单步执行代码，这意味着可以观察每行代码的执行。

单步执行有两种类型：逐语句和逐过程。通过单步执行方法，可以观察应用程序的逐行执行，这样就可以跟踪逻辑语句的流程并查看各个变量和属性的数值。

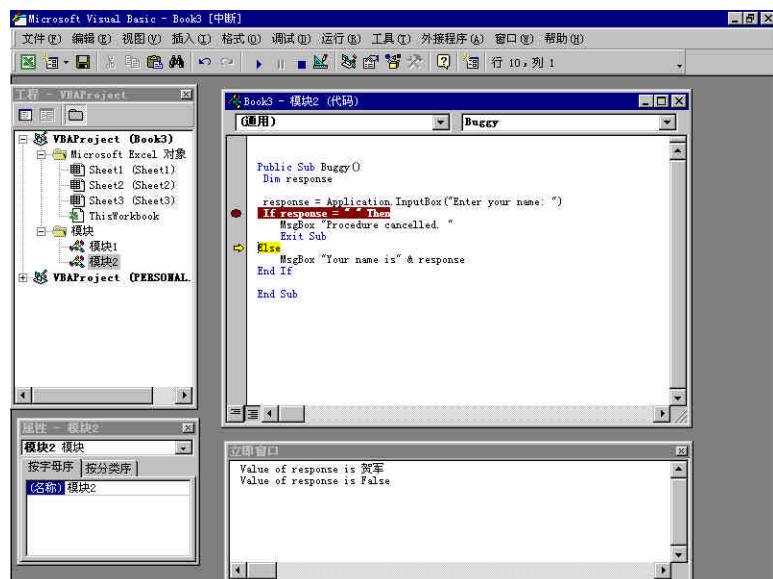
这两种单步执行方法只有一个区别。如果在调试的过程中调用了一个或者多个其他过程，并且不希望进入其他过程，那么应该使用逐过程的方式。逐过程不会单步执行被调用的过程。另一方面，如果希望观察所有语句（包括任何被调用过程中的语句）的单步执行，那么请使用逐语句的方法。

通常情况下，用不着单步执行整个过程，而是在过程中有问题的部分设置断点，接着从

断点处开始单步执行。要单步执行过程，请完成如下步骤：

- 1) 从过程中删除Debug.Print语句。
- 2) 在InputBox语句后面的代码行设置断点，按F5键运行过程。
- 3) 在显示出来的输入框中单击“取消”按钮，因为设置了断点，所以会返回到过程中。过程接下来将执行If语句，你期望发生的是第一个条件判断的结果为True。
- 4) 按F8键单步执行到下一行代码，注意执行的下一行代码是Else从句，而不是显示“Procedure Cancelled.”消息框，如图12-8所示。这对于调试而言是一个非常重要的线索！
- 5) 按F5键继续运行该过程。单击“确定”按钮，退出消息框。

**图12-8 当单步执行某个过
程时，当前的代码
行在边缘的指示器
栏中用一个箭头加
以区分**



12.6 使用监视

新术语 现在你也许已经知道了问题所在。然而，在改正这个错误之前，还需要尝试另一种称为监视表达式的调试技术。监视表达式是用户定义的表达式，通过它可以查看为变量或者表达式设置的数值。监视表达式可以是任何有效的Visual Basic表达式。

监视表达式共有三种类型，具体如下

- 监视表达式 用于在不影响应用程序执行的情况下监视某个表达式。
- 当监视值为真时中断 用于希望查看值为真时会发生什么时的情况，例如，如果希望查看Response变为与空字符串“”相等时会出现什么结果，就可以使用这种类型。
- 当监视值改变时中断 如果使用本选项，则当表达式的数值发生改变时，过程执行就会中断。

要用监视表达式来提供response的数值，请按照如下步骤：

- 1) 删除所有的断点。
- 2) 将Buggy过程中的response突出显示。
- 3) 选择“调试”、“添加监视”，显示“添加监视”对话框，如图12-9所示。

图12-9 通过“添加监视”对话框可以创建三种类型的监视



4) 因为当选择“调试”、“添加监视”时变量名处于突出显示状态，所以变量名已经显示在“表达式”文本框中。

通过“上下文”框中的项，可以控制变量将在何处作为监视表达式。因为变量 response 的作用域为过程内部，所以不用对这些项目进行改变。

5) 选择“当监视值改变时中断”选项按钮。

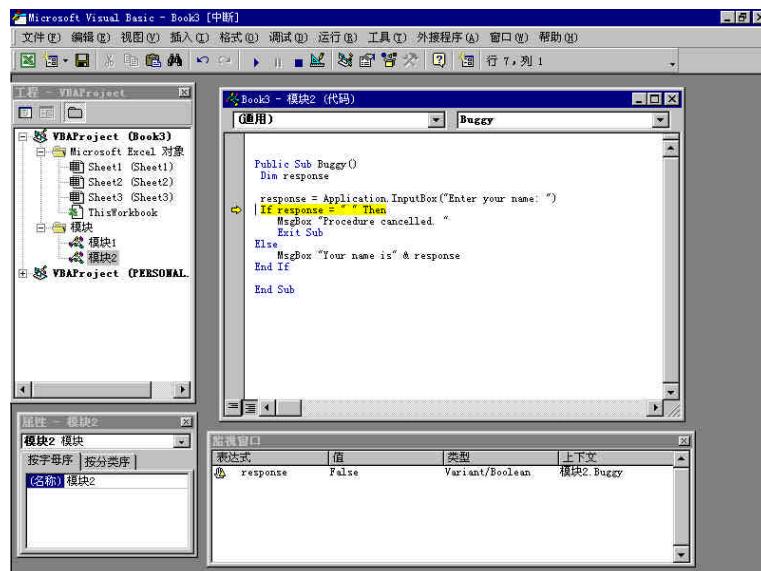
6) 单击“确定”按钮。

如果希望快速创建监视表达式，可在“代码”查看中选中相应的表达式，再选择“调试”、“快速监视”或者按F9键。用这种方法只能创建监视表达式，即是说不能用这种技术创建“当监视值改变时中断”或者“当监视值为真时中断”选项。

7) 按F5键运行这个应用程序。

8) 在显示出来的输入框中单击“取消”按钮。因为设置了监视表达式，所以这个过程处于中断模式。在Visual Basic编辑器中显示出“监视窗口”，如图12-10所示。

图12-10 “监视窗口”中列出定义的所有监视表达式以及它们的数值



9) 按F5键继续执行该应用程序，单击“确定”按钮，退出消息框。

现在，已经结束了对监视窗口的使用，接下来就应该改正过程中的错误了。要删除监视，

可以按照如下步骤：

- 1) 选择“调试”、“编辑监视”，显示“编辑监视”对话框，如图 12-11 所示。

图12-11 使用“编辑监视”

对话框可以修改、
删除已有的监视



- 2) 在“表达式”文本框中列出了唯一的监视表达式——response，单击“删除”按钮，这样就删除了该监视表达式。

12.7 改正代码中的错误

到现在为止，你已经完全为改正 Buggy 过程中的错误做好了准备。错误出在：如果选择“取消”按钮，InputBox 方法的返回值是“False”，而不是像 InputBox 函数那样返回空字符串。正确的过程如程序清单 12-1 所示。

程序清单 12-1 改正后的 Buggy 过程

```

1: Public Sub Buggy ()
2:   Dim response
3:
4:   response = Application.InputBox("Enter your name: ")
5:   If response = False Then
6:     MsgBox "Procedure cancelled. "
7:     Exit Sub
8:   Else
9:     MsgBox "Your name is" & response
10:  End If
11: End Sub

```

通过测试 response 是否等于“False”，解决了程序中的问题。

12.8 学时小结

长时间以来，专业开发人员瞧不起应用程序自动化语言，认为这是属于用户的领域。其中的一个原因是调试工具的缺乏。而现在，VBA 对这一问题相当重视，它功能强大的调试工具能够满足普通的和专业的开发人员的需要。

12.9 专家答疑

问题：如果我只希望运行过程中的某个部分，应当怎么办？

解答：可在你希望中断或者暂停的地方设置断点，这正是断点的用途。

问题：为什么我应当使用监视窗口，而不是在立即窗口中使用 Print 语句？

解答：使用监视窗口相对于使用立即窗口的优点在于，它可以在过程运行之前设置，并可在过程运行的过程中连续地跟踪，监视窗口也可用来中断过程以便测试。

12.10 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

12.10.1 思考题

- 1) 当运行到断点时，进入什么模式？
- 2) 怎样才能知道过程中代码执行的顺序？
- 3) 在立即窗口中怎样显示变量和属性的数值？
- 4) 请指出除了断点之外的另一种暂停应用程序的方法。
- 5) 判断题：当某个过程处于中断模式时，不能查看过程中变量的数值。
- 6) 指出单步执行代码的两种方式。
- 7) 判断题：Watches永远不会影响程序的运行。

12.10.2 练习题

创建如下过程：

```
Sub Hour12Exercise ()  
    Dim sWhichState As String  
  
    sWhichState = InputBox ("Enter the state for shipping: ")  
  
    Select Case sWhichState  
        Case "FL"  
            MsgBox "Shipping is 3.50."  
        Case "NY"  
            MsgBox "Shipping is 5.00."  
        Case "OH"  
            MsgBox "Shipping is 2.00."  
        Case "CA"  
            MsgBox "Shipping is 6.00."  
        Case Else  
            MsgBox "We don't ship there."  
    End Select  
  
End Sub
```

运行该过程，在输入框中输入“Ny”，得到的运行结果和预期的不同，请使用本学时中学到的调试技术找出原因。

第13学时 错误处理

在前一学时里，我们已经学习了有关调试方面的内容，所谓调试，就是对可以预测的问题进行处理并进行纠正的过程。但是，如果问题是不可控制和不可预测的话，应该怎么处理呢？除非程序员开发的应用程序很小，否则，就没有办法预测到应用程序运行时可能发生的所有问题。开发人员不能够控制操作系统、也不能够控制硬件平台的选配，而且肯定不能够控制用户的操作。

这个学时中要学习的重点包括：

- 错误处理是什么
- 设置错误捕获
- 编写错误处理实用程序
- 提供从错误处理程序跳出的出口
- 创建集中的错误处理程序

13.1 错误处理

新术语 为了处理突发事件，开发人员需要编写错误处理程序。所谓错误处理程序，就是应用程序中用来捕获和处理错误的实用程序。

错误处理程序的开发过程可以分为下面三个步骤：

- 1) 设置错误捕获 当设置错误捕获时，就是告诉程序当错误发生时，到什么地方去捕获错误。
- 2) 编写错误处理实用程序 错误处理实用程序就是当错误发生时程序要跳转到的地方。
- 3) 提供从错误处理程序跳出的出口 换句话说，就是当错误处理完毕时，需要程序做的事情。

13.2 步骤1：设置错误捕获

设置错误捕获就是告诉VBA捕获错误的位置，是通过On Error语句来实现的。在一个给定的程序中，任何一个时刻只能够启用一个错误捕获。当然，这不是意味着过程就只能够拥有一个On Error语句，而是说，如果过程中有多个On Error语句的话，只有最近正在执行的那个捕获陷阱才是起作用的。

新术语 处理错误有两种不同的方式，其中之一是执行内联错误处理。内联错误处理在On Error语句中有一些指令来处理错误。要执行内联错误处理的话，可以使用下面语句中的任何一句：

- On Error Resume 如果有运行时刻的错误发生，那么程序将从导致错误发生的语句处重新开始执行。
- On Error Resume Next 如果有运行时刻的错误发生，那么程序就从导致错误发生的语句的下一句继续执行下去。

要禁止错误的处理程序，可以在程序中初始化 On Error 语句以后，使用 On Error GoTo 0 语句。在测试程序或过程并且不想启用错误处理时，禁止错误处理程序是非常有用的。

错误处理的两种不同处理方式中，不推荐使用内联错误处理方法，最好是采用错误捕获的处理方式，采用错误捕获能够跳转到错误处理实用程序，这样的方式使开发人员能够对各种各样的错误进行灵活的处理。

为了设置能够跳转到错误处理实用程序的错误捕获，可以使用 On Error GoTo “行” 语句，这里“行”代表的是位于错误处理代码前面的行标号。要创建行标号的话，只要为该行输入一个名称，后面跟一个冒号就可以了。VBA中的行标号需要独占一行。程序清单 13-1给出了包含有错误处理实用程序的过程的基本框架。

程序清单 13-1 包含错误处理实用程序的过程的基本框架

```
1: Sub WithErrorHandler ()  
2:   On Error GoTo ErrorHandler  
3:   'The body of the procedure goes here.  
4:   .  
5:   .  
6:   .  
7:   'The next statement, Exit Sub or Exit Function (whatever is  
8:   'appropriate), goes before the line label for the error handler.  
9:   'This is done so that if there are no errors, the error-  
10:  'handling routine is skipped.  
11:  Exit Sub  
12: 'The next line is the line label for the error-handling  
13: 'routine.  
14: ErrorHandler:  
15:   'The code for the error handler goes here.  
16:   .  
17:   .  
18:   .  
19: End Sub
```

13.3 步骤2：编写错误处理实用程序

当错误发生时，VBA就查找程序中的行标号，并开始跳转到行标号所在的位置继续执行。错误处理实用程序的代码评估所发生的错误并采取相应的措施。评估处理过程要么是采用 If 语句、要么是采用 Select 语句来完成的，在这两个语句中，应该总是包括 Else 子句，用它来处理那些所有没有预料到的错误。

13.4 步骤3：提供从错误处理程序跳出的出口

你也许想知道错误处理程序的 If 或者 Select 语句中那些条件的使用方法，下面简单介绍。你将测试 Err 对象的 Number 属性的值，Number 属性是该对象的默认属性。Err 对象包含了有关运行时刻错误的信息，它的 Number 属性可以用来返回由发生错误所指定的一个数值（每个错误对应一个数值），一旦明确了所发生的错误以后，就有如下四种选择：

- Resume 返回到导致错误发生的语句。
- Resume Next 返回到导致错误发生的语句的下一行语句。
- Resume “行” 跳转到程序中行标号标明的行。
- End 结束过程或者整个应用程序。

13.5 综合

既然已经学习了创建错误处理程序的所有理论，下面就可以开始把理论应用到实际中了：在工作簿中添加一个新模块。要创建一个包含有错误处理程序的过程的话，请执行下面的步骤：

1) 创建一个新过程，命名为 ErrorExample。

2) 在过程中输入下面的代码：

```
Dim sngValue As Single, sngDivideBy As Single, sngAnswer As Single
Dim iResponse As Integer
On Error GoTo ErrorZone
sngValue = InputBox("Enter the number you wish to divide: ")
sngDivideBy = InputBox("Enter the number you wish to divide by: ")
sngAnswer = sngValue / sngDivideBy
MsgBox "The answer is "& sngAnswer

Exit Sub
ErrorZone:
Select Case Err
Case 7
    MsgBox "Out of Memory" & _
        Chr(13) & "Close nonessential applications. "
    Resume
Case 35 To 51
    MsgBox "Contact the Help Desk. "
Case 11
    MsgBox "You can't divide by zero. "
Case Else
    MsgBox "Unrecoverable error. Exiting application. "
End
End Select
```

3) 运行该过程，在第一个输入框中输入 10。

4) 在第二个输入框中输入 0，当按下回车键以后，就可以看见一个错误消息显示出来。这是因为试图被0除，从而导致错误发生。

5) 单击“确定”按钮来关闭消息对话框。

可以使用 Error 函数来显示与发生的错误相关联的文本信息。要修改过程来使用 Error 函数的话，请完成下面的步骤：

1) 修改 ErrorExample 过程，使它跟下面的代码一样（修改的部分用黑粗体字标出）：

```
Private Sub ErrorExample ()
    Dim sngValue As Single, sngDivideBy As Single, sngAnswer As Single
    Dim iResponse As Integer
    On Error GoTo ErrorZone
```

```

sngValue = InputBox("Enter the number you wish to divide: ")
sngDivideBy = InputBox("Enter the number you wish to divide by: ")
sngAnswer = sngValue / sngDivideBy
MsgBox "The answer is " & sngAnswer

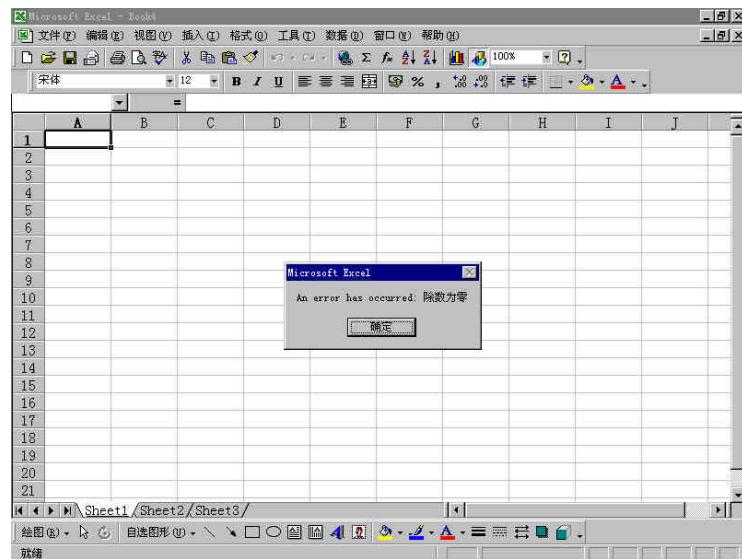
Exit Sub
ErrorZone:
    Select Case Err
        Case 7
            MsgBox "Out of Memory" & _
                    Chr(13) & "Close nonessential applications."
            Resume
        Case 35 To 51
            MsgBox "Contact the Help Desk."
            Exit Sub
        Case 11
            MsgBox "An error has occurred: "& Error
        Case Else
            MsgBox "Unrecoverable error.Exiting application."
            End
    End Select
End Sub

```

2) 运行该过程，在第一个输入框中输入 5。

3) 在第二个输入框中输入 0，当按下回车键时，就可以看到一条错误消息显示出来，该消息显示了跟错误消息相关联的文本，如图 13-1 所示。

图13-1 这个消息对话框使用由 Error 函数产生的文本



4) 单击“确定”按钮关闭该消息对话框。

你也许在想“我实在不想为每个过程都编写一个错误处理实用程序”，实际上也没有这个必要。在下面的介绍中，将学习怎样创建一个集中的错误处理程序。

13.6 创建集中的错误处理程序

新术语 可以创建集中的错误处理程序来代替在每个过程中创建一个错误处理实用程序。集中式错误处理程序是一个函数，它对所发生的每个错误进行处理，然后基于不同的错误号码采取不同的措施。程序中需要进行错误处理的每个过程中仍然需要有 On Error语句，该语句跳转到拥有Select Case语句的应用程序某个区域中，Select Case语句能够处理有集中式错误处理程序返回的结果。Select Case语句有五种可能的情况：

- 执行Resume。
- 执行Resume Next。
- 执行Resume “行”。
- 退出该过程。
- 结束整个应用程序。

对应于每个错误，从集中式错误处理程序会返回一个相应的值到局部过程中，这个值的范围是1~5。局部过程就使用这个返回值初始化相应措施。理解这个过程的最好方式就是实践。要添加集中式错误处理程序的话，请执行下面的步骤：

1) 添加一个模块到工作簿中，创建一个函数，命名为 HandleErrors，要确保它是一个函数。

2) 为该函数输入下面的代码：

```
Function HandleErrors(iErrNum) As Integer
    Select Case iAction
        Case 5
            'Invalid procedure call
            MsgBox Error(iErrNum) & " Contact Help Desk."
            iAction = 2

        Case 7
            'Out of memory
            MsgBox "Close all unnecessary applications. "
            iAction = 1

        Case 11
            'Division by zero
            MsgBox "Zero is not a valid value. "
            iAction = 1

        Case 48, 49,51
            'Error in loading DLL
            MsgBox iErrNum & "Contact Help Desk. "
            iAction = 5

        Case 57
            'Device I/O error
            MsgBox "Insert Disk in Drive A."
            iAction = 1

        Case Else
            MsgBox "Unrecoverable Error. "
            iAction = 5
    End Select
```

```
ErrorHandler = iAction
End Function

3) 找到ErrorExample过程(它在另外一个模块中,而不是在当前模块中)。
4) 修改该过程,使它同下面的代码一样(修改部分用黑粗体字标出):

Private Sub ErrorExample()
    Dim sngValue As Single, sngDivideBy As Single, sngAnswer As Single
    Dim iResponse As Integer
    On Error GoTo ErrorZone
    sngValue = InputBox("Enter the number you wish to divide: ")
    sngDivideBy = InputBox("Enter the number you wish to divide by: ")
    sngAnswer = sngValue / sngDivideBy
    MsgBox "The answer is " & sngAnswer
Exit Sub
ErrorZone:
'This Select statement uses the value returned
'from the HandleErrors function for its condition.
    Select Case HandleErrors(Err)
        Case 1
            Resume
        Case 2
            Resume Next
'This procedure doesn't need case 3 which is resuming to a line.
        Case 4
            Exit Sub
        Case 5
            End
    End Select
End Sub
```

5) 运行该过程。

6) 在第一个输入框中输入8。

7) 在第二个输入框中输入0,按回车键后将发生错误。

8) 单击“确定”按钮关闭该消息对话框。

通过跳转到集中式错误处理函数、然后执行局部过程中的 Resume、Resume Next或者其他必要的语句,这样就可以确保VBA在错误发生以后能够从合适的位置继续执行下去。

该方法也允许在局部过程的错误处理实用程序中放置自定义代码,比如放置某个消息等,需要在想要支持错误处理的所有过程中都放置自定义代码。

只需要一次创建集中式错误处理程序,在将来的应用程序中,通过复制和粘贴该函数就可以重复使用它了。

13.7 学时小结

调试只能够发现可以预测的错误,要处理不可预测的和不可避免的错误时,就必须使用错误处理。通过启用错误处理,就可以使应用程序更稳定、更健壮。如果应用程序中包含了好几个过程,那么可以考虑采用集中式错误处理程序。

13.8 专家答疑

问题：如果需要强制一个错误发生，该怎么实现呢？

解答：可以使用Raise函数来强制发生错误，这在测试处理过程中很有用，另外在最终应用程序的某些情况下也可能有用。

问题：有一个过程要被另外一个过程调用，该过程没有错误处理程序，但是主调过程有，那么当错误发生时，将发生什么事情呢？

解答：如果过程被另外一个过程调用、并且没有错误处理程序的话，VBA将在主调过程中查找错误处理程序。

13.9 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容。答案请参考附录。

13.9.1 思考题

- 1) 创建错误处理程序的三个主要步骤是什么？
- 2) 用来返回错误号码的对象及其属性的名称是什么？
- 3) 哪条语句能够把流程跳转到导致错误发生的代码行？
- 4) 为了把某行变为行标号，该行最后要放置哪个字符？
- 5) 哪条语句能够把流程跳过导致错误发生的代码行？
- 6) 判断题：每个过程都必须有自己的错误处理实用程序。
- 7) 当创建错误处理程序时，哪种逻辑结构最好？

13.9.2 练习题

创建下面的过程：

```
Sub ProcWithError()  
    Workbooks.Open " C:\nosuchfile.wkb"  
End Sub
```

添加代码来实现一个错误处理程序，该错误处理程序哪个显示一条消息，并且从导致错误发生的代码行的下一行继续执行。

第14学时 使用用户窗体

到现在为止，已经学习了许多语法，知道怎样去测试和调试应用程序，知道怎样去处理错误，那么接下来还要学习什么内容呢？实际上，你将返回去学习用户界面。现在，我们准备开始学习用户窗体，用户窗体是自定义应用程序界面的方式之一。

你将要创建一个应用程序来跟踪旅馆客人的花费，当客人使用高尔夫球场或者网球场，或者需要在游泳池使用额外的毛巾（如此等等）时，该应用程序都有对应的处理入口。在该应用程序中，旅馆客人不会直接同工作簿交互，相反，他们将通过你在本章中要创建的一个自定义对话框来输入信息。

这个学时中要学习的重点包括：

- 添加和运行用户窗体
- 在用户窗体上放置控件
- 控件Tab键切换顺序的重要性
- 实现加速键

14.1 为应用程序添加用户窗体

在第3学时里，学习了把控件直接放置到工作簿上的方法，在想要用户直接同工作簿进行交互时，这种放置控件的方法是很好的。但是，如果想把用户同 Excel环境隔离开来的话，那么需要采取什么方法呢？那就是使用用户窗体。

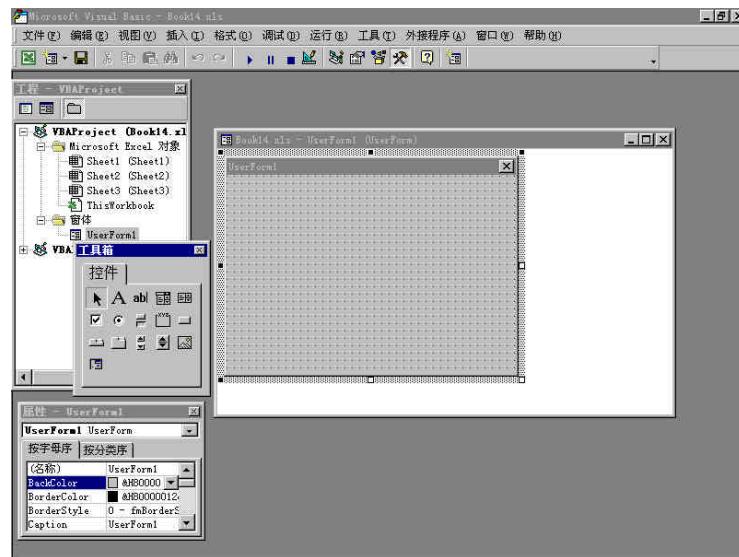
新术语 用户窗体是应用程序的自定义窗口和对话框，用户窗体包含了一些控件，利用这些控件可以从用户那里获取信息。要给应用程序添加用户窗体的话，请执行下面的步骤：

在更早版本的Excel中，用户窗体叫做对话表。

- 1) 关闭所有打开的工作簿。
- 2) 打开一个新的工作簿。
- 3) 按下Alt+F11来打开Visual Basic编辑器。
- 4) 用鼠标右键单击工程资源管理器中的ThisWorkbook。
- 5) 从菜单中选择“插入”、“用户窗体”，于是用户窗体就添加到工作簿中了，如图 14-1 所示。

新添加的用户窗体是一个带有标题栏的灰色框，它就是创建自定义对话框的基石，接下来将看到它的一些属性，并学习怎样去运行它。

图14-1 新添加的用户窗体
看起来就好像空的
对话框



14.1.1 设置用户窗体的属性

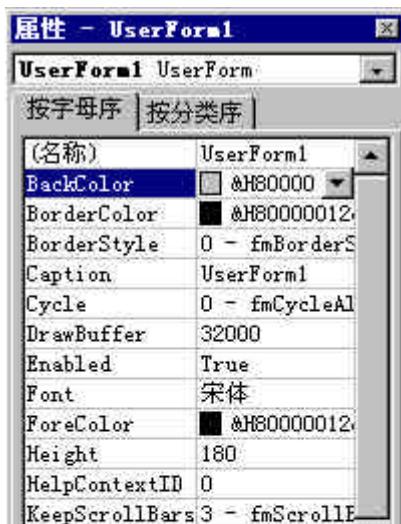
使用用户窗体的第一步就是设置某些属性值，首先要设置的用户窗体的属性值或者对象就是Name（名称）属性，Name属性控制了代码中引用用户窗体的方式。当用户窗体第一次创建时，Excel就为它分配了一个名称，例如像UserForm1。如果接连创建了好几个用户窗体，而让Excel命名为UserForm1、UserForm2、UserForm3的话，能够想象那种混乱情形么？

用户窗体推荐的命名前缀是frm，frm就是form（窗体）的缩写。

设置了Name属性以后，接下来要做的事情就是设置用户窗体的Caption（标题）属性。Caption属性控制了窗体标题栏中所显示的文本。要设置这些属性的话，请执行下面的步骤：

1) 如果“属性”窗口没有显示出来的话，请按F4键，图14-2给出了“属性”窗口，其中列出了用户窗体的属性。

图14-2 浏览属性有两种选
择：按照字母顺序
和按照类别浏览



- 2) 选择Name属性，该属性可以在按照字母顺序列表的最上头找到，显示的是“(名称)”。
- 3) 输入frmGuestExpenses。
- 4) 选择Caption属性。
- 5) 输入Guest Expenses。注意，当输入Guest Expenses到Caption属性中时，用户窗体的标题栏就随之改变。

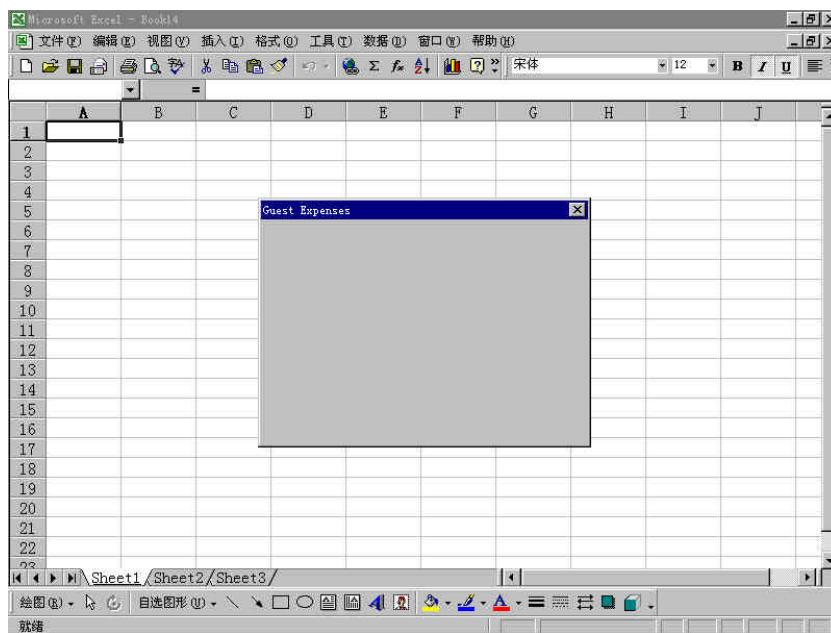
到现在为止，窗体可以看到的唯一改变就是标题栏。还可以为用户窗体设置各种各样其他属性，不过这里只需要用到Name和Caption这两个属性就够了。

14.1.2 运行用户窗体

即使没有任何代码同这个窗体关联，仍然可以运行它。运行用户窗体意味着可以看见该窗体在应用程序中显示出来。执行下面的步骤来运行用户窗体：

- 1) 单击用户窗体以选中它。
- 2) 按下F5键，用户窗体就以“运行”模式显示出来，如图 14-3所示。注意，在Visual Basic编辑器中使用用户窗体时，可以看到的网格在运行窗体时并不显示出来。

图14-3 运行用户窗体的一个例子



- 3) 使用窗体标题栏上的关闭按钮来关闭用户窗体。
 - 4) 返回到Visual Basic编辑器。
- 从这些步骤中可以看到，使用用户窗体可以获得意外的收获：不需要为窗体的关闭按钮编写任何代码，Excel已经处理好了这一点。

14.2 添加控件到用户窗体上

添加控件到用户窗体上跟向工作表上放置控件非常相似，只要从工具箱中选择想要使用

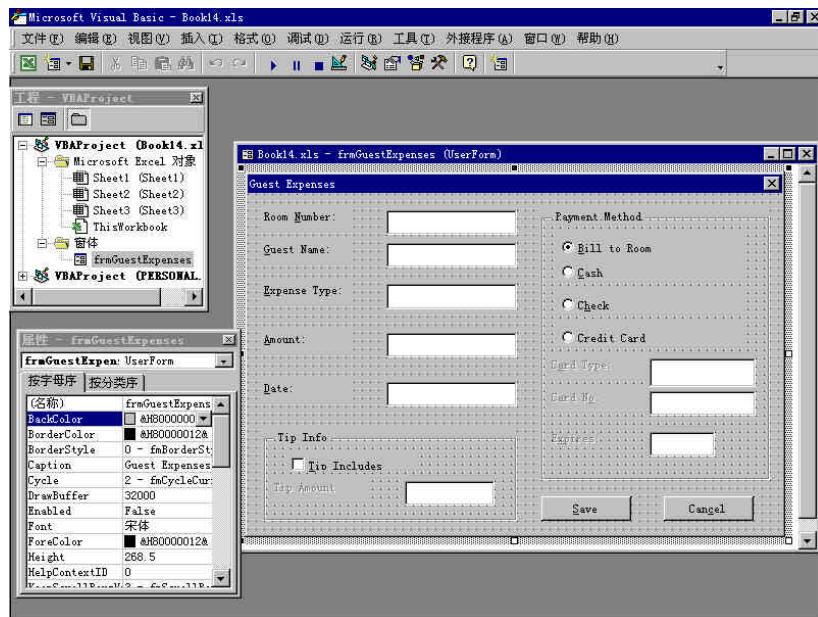
的控件，然后使用鼠标把控件拖到窗体上就可以了。

当向工作表上放置控件时，使用的是“窗体”工具栏，这里将使用工具箱。

请参阅第3学时来获取有关控件不同类型及其使用方法的更多信息。

现在，将要创建一个窗体来跟踪旅馆中各种各样的旅客花费，当这个窗体完成时，看起来就如图14-4所示的那样。

图14-4 完成了的“Guest Expenses”窗体



从图14-4中可以看到，该窗体使用了许多控件，现在准备把这些控件放置到窗体上，请执行下面的步骤：

- 1) 如果工具箱没有显示出来的话，选择“视图”、“工具箱”。
- 2) 从工具箱中选择标签控件。
- 3) 以图14-4为参照，把标签控件放置到窗体的左上角。
- 4) 如果“属性”窗口没有打开的话，请按 F4键，把标签控件的 (Name)属性设置为 lblRoomNumber，把控件的Caption属性设置为 Room Number。
- 5) 从工具箱中选择文本控件，把文本控件放置在标签控件旁边，并把它的 (Name)属性设置为 txtRoomNumber。
- 6) 以图14-4为参照，继续使用表 14-1 中的那些属性来创建下面这些控件。当使用框架时，首先拖放框架，然后把控件放置到框架中。另外，不要过分考虑把控件的对齐方式和尺寸调整得非常完美，稍后将学习到一些技术，使得控件对齐和尺寸调整变得更加简单。

表14-1 frmGuestExpenses窗体上的控件一览

控件类型	Name属性	Caption属性
标签	lblGuestName	Guest Name:
文字框	txtGuestName	
标签	lblExpenseType	Expense Type:
列表框	lstExpenseType	
标签	lblAmount	Amount:
文字框	txtAmount	
标签	lblDate	Date:
文字框	txtDate	
框架	fraPayment	Payment Method
选项按钮	optBillToRoom	Bill to Room
选项按钮	optCash	Cash
选项按钮	optCheck	Check
选项按钮	optCreditCard	Credit Card
标签	lblCardType	Card Type:
列表框	lstCardType	
标签	lblCardNumber	Card No.:
文字框	txtCardNumber	
标签	lblExpires	Expires:
文字框	txtExpires	
命令按钮(命令按钮)	cmdSave	Save
命令按钮	cmdCancel	Cancel
框架	fraTips	Tip Info
复选框	chkTipIncluded	Tip Included
标签	lblTipAmount	Tip Amount:
文字框	txtTipAmount	

7) 把工作簿保存为 Guest Expenses。

创建控件有些单调无聊，但是这显然是必要的。在本章剩下的时间里，将使用其他一些属性和技术来使窗体和控件的外观更好看。

设置其他属性

在前面的实践中，已经为所有的控件都设置了（名称）属性，为其中的一些控件设置了 Caption，现在将要学习怎样设置其他一些属性。

1. 命令按钮属性——Default（默认）和Cancel（取消）

准备为控件设置其他属性的第一种控件是命令按钮。刚才的窗体上有旅馆命令按钮：Save（保存）和Cancel（取消）。Save按钮最终完成的工作是验证数据，如果数据合法的话，就把数据写入到工作表中。你可以想一想自己是怎样输入数据的：很可能是逐个字段地输入，当输入完成时，就按下回车键。在这个窗体上按下回车键不会完成任何事情，必须设置一个属性，使得当按下回车键时，等同于单击了 Save按钮。为了实现这一点，就必须把 Save按钮的Default属性设置为 True。一个窗体上只有一个命令按钮的 Default属性可以设置为 True。

在大多数窗体上当按下回车键时，窗体将执行取消操作。在刚才的窗体上，必须把 Save按钮的Cancel属性设置为 True。

另外，你还想把这两个按钮的尺寸调整为一致，要完成这一点的话，可以使用“宽度相

同”工具栏按钮，通过执行下面的步骤就可以完成：

- 1) 单击Save按钮以选中它。
- 2) 按下Ctrl键，并单击Cancel按钮，这样就同时选中了这两个按钮。
- 3) 从用户窗体的工具栏上选择“宽度相同”按钮旁边的向下箭头。

如果没有看见用户窗体工具栏的话，在工具栏上单击鼠标右键，然后从菜单中选择“用户窗体”。

- 4) 选择“两者都相同”。

这两个按钮将按照Cancel按钮的标准调整到相同的尺寸。最后选中的按钮将作为其他选中按钮进行尺寸调整的标准。

2. 设置单选按钮的Value属性并对齐控件

你要使用的下一种控件是选项按钮，注意到窗体上选项按钮的顺序：首先是 Bill to Room，然后是 Cash，如此等等。选择这样的顺序是因为假定大多数人是把花费支出在房间上，其次最常用的支付方法是现金，顺序依此递减。现在可以发现在设计使用了选项按钮的窗体时，总是试图把选项按钮按照从使用次数最多到最少的顺序来排列组织。另外还需要把其中的一个选项按钮设置为默认选项，为了实现这一点，需要把 optBillToRoom 选项按钮的 Value 属性设置为 True。

另外，你想要把自己的选项按钮按照左边对齐，可以使用下面的步骤来实现：

- 1) 单击 Bill to Room 选项按钮。
- 2) 按下 Ctrl 的同时，再单击其余的选项按钮。
- 3) 单击用户窗体工具栏上“对齐”工具栏按钮旁边的向下箭头。
- 4) 选择“左对齐”，现在控件就左对齐了。

如果 Payment Method 框架中的其他控件也需要对齐的话，这时就可以进行类似的操作了。

3. 禁用控件

还有一些控件只有在其他控件被选中时才需要用到，例如，除非 Tip Included 复选框已经被选中，否则，就不需要使用 Tip Amount 文本框。另外，除非 Credit Card 选项按钮被选中，否则就不需要同信用卡关联的那些控件。为了使用户更明显地看到这一点，就可以禁用那些暂时用不着的控件。当控件禁用时，它们将以灰色显示。之后，如果选中了合适的控件时，可以重新启用那些被禁用的控件。把 Enabled (启用) 属性设置为 False 就可以禁用控件。

4. 稍微再改进一下窗体的外观

现在可以花一些时间，使用前面介绍过的技术来对齐控件，调整控件的尺寸，可以把图 14-5 作为一个参照。

在使用控件时，可能想要设置的另外一个属性是 ControlTipText，如果为这个属性输入一个值，那么当用户把鼠标的指针定位在该控件上时，该属性的值就作为工具提示显示出来。

14.3 为控件分配Tab键切换顺序

现在运行用户窗体，按下 Tab 键可能会发现，Tab 键选中控件的顺序跟希望的顺序不一样。

例如，使用Tab键在选中tip控件之前将先选中payment method控件。又比如说，Save和Cancel按钮不是最后选中的控件等，这显然不是希望的顺序。原因在于：默认情况下，控件的Tab键切换顺序是按照创建它们的先后顺序来分配的。所以，如果窗体上控件的Tab键切换顺序跟自己希望的不一样时，就应该使用TabIndex属性来校正切换顺序。TabIndex属性能够控制窗体上控件的Tab键切换顺序，该属性的起始值是0。

设置TabIndex属性值的第一种方法是把第一个控件的TabIndex值设置为0，第二个控件设置为1，依此类推。还有另外一种设置方法，该方法从希望Tab键最后要选中的那个控件开始，把该控件的TabIndex值设置为0，然后把Tab倒数第二个被选中控件的TabIndex值设置为0，如此操作，按照Tab键选中控件的逆顺序来把每个控件的TabIndex设置为0。要注意，控件要包括标签和框架。能够这样逆顺序操作的原因在于：当把某个控件的TabIndex属性值设置为0时，其他所有控件的TabIndex属性值都会加1。现在，可以花几分钟时间来把刚才示例中的Tab键切换顺序纠正过来，把控件的TabIndex属性值设置完毕后，再次运行用户窗体来查看控件的Tab键切换顺序的设置是否跟希望的一样。

14.4 为控件分配加速键

到现在为止，窗体的界面部分基本上就完成了，运行窗体，将发现还有什么地方需要改进。如果使用键盘而不是使用鼠标来操作的话，很可能就会发现有需要改进的地方：这个窗体没有任何加速键（也就是通过键盘来访问控件的按键）。要添加加速键的话，需要使用Accelerator属性，在控件的标题字母中找出一个作为该控件的加速键，把该字母赋值给控件的Accelerator属性，当设置完成以后，控件标题中该字母下会有下划线，如图14-5所示。表14-2中给出了示例窗体中各控件Accelerator属性的推荐取值。

新术语 加速键是同PC平台上的Alt键或者Macintosh平台上的Command键结合使用来访问/选中某个控件的，例如，如果某个控件用字母B作为它的加速键，那么，就可以按下Alt+B组合键来访问/选中该控件。

图14-5 愿意使用键盘来代替鼠标进行操作的用户对加速键情有独钟

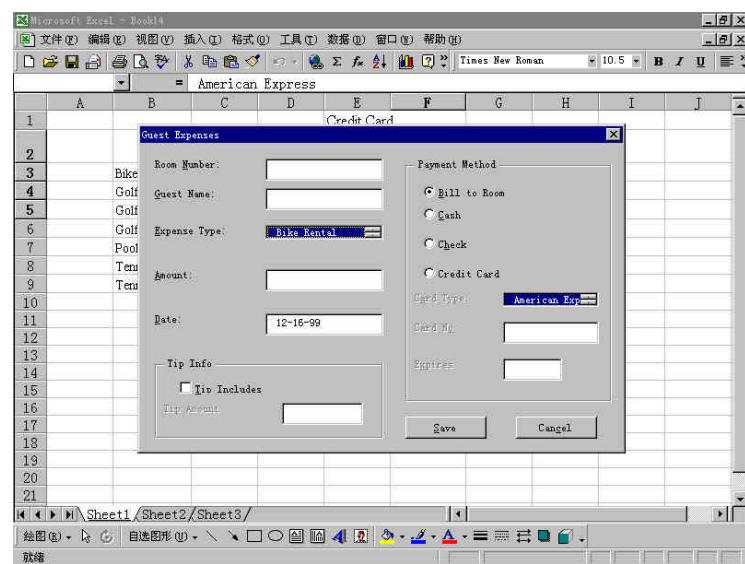


表14-2 示例窗体中控件的加速键一览表

控 件	加 速 键
lblRoomNumber	N
lblGuestName	G
lblExpenseType	E
lblAmount	A
lblDate	D
chkTipIncluded	T
lblTipAmount	i
optBillToRoom	B
optCash	C
optCheck	h
optCreditCard	r
lblCardType	y
lblCardNumber	o
lblExpires	x
cmdSave	S

14.5 学时小结

在这个学时中，创建了一个用户窗体，以实现一定程度的结算自动化。这个实现过程大致是这样的：把用户窗体添加到到应用程序中，然后向窗体上添加控件，最后设置控件的各项属性，最终的结果就是得到具有专业水准的自定义对话框。在下一个学时里，将编写必要的代码来实现该窗体的自动功能。我们将要编写一些代码来验证用户输入到窗体的数据是否合法有效，同时编写一些代码来把窗体上表单中的数据写入到工作表中去。

14.6 专家答疑

问题：为什么要使用用户窗体来放置控件而不是把控件直接放置到工作表上呢？

解答：如果只需要少数几个控件时，是可以把控件直接放置到工作表上的。如果需要的控件数目多一些的时候，最好的解决方案是使用用户窗体。使用用户窗体的另外一个优点是能够提供一个更为专业的界面，直接把控件放置到工作表上使得应用程序好象有点“作坊化”。

问题：每个应用程序仅限于使用一个用户窗体吗？

解答：不是。如果需要的话，可以把多个用户窗体添加到应用程序中，显然，应用程序越复杂，需要使用的用户窗体数量也就越多。

14.7 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容。答案请参考附录。

14.7.1 思考题

- 1) 判断题：当运行用户窗体时，用户窗体上的网格就显示出来。
- 2) 哪个属性是用来设置控件加速键的？
- 3) 当按下回车键时，怎样识别要执行的是哪个命令按钮？

- 4) 设置哪个属性来选中选项按钮呢？
- 5) 判断题：对齐控件需要设置一些属性。
- 6) 当把命令按钮的Cancel属性值设置为True时，意味着什么呢？
- 7) 窗体上控件的初始Tab键切换顺序是由什么来决定的？

14.7.2 练习题

在这个练习中，需要创建如图 14-6 所示的窗体，把该窗体命名为 frmSplash，把它的标题设置为 Welcome to Guest Expenses!，表 14-3 给出了一些需要的属性设置。

图14-6 练习完成的结果窗体

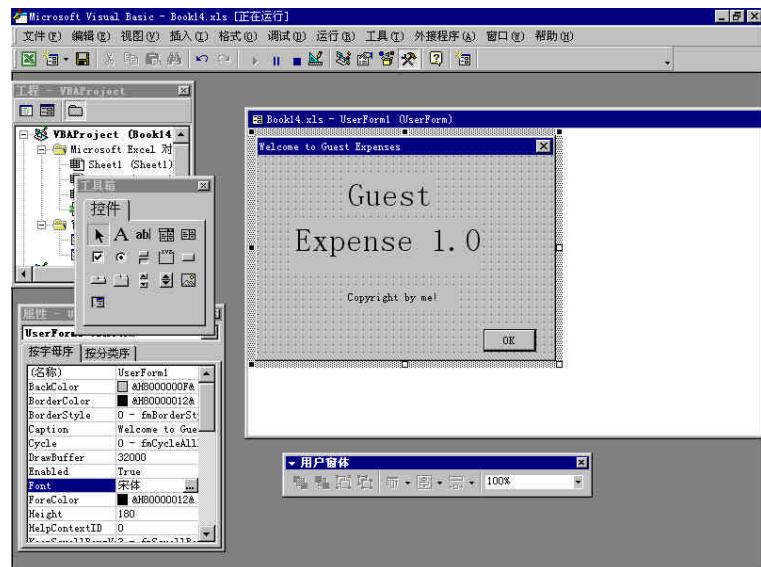


表14-3 属性设置一览

控件类型	名 称	Caption属性取值	其他属性取值
标签	lblTitle	Guest Expenses 1.0	字体大小：26磅 TextAlign : fm.TextAlignCenter
标签	lblCopyRight	Copyrighted by Me!	ForeColor : blue
按钮	cmdOK	OK	Default : True Cancel : True

要设置标签的字体信息的话，找到并选中 Font 属性，单击“选择”按钮，使用“字体”对话框来选择要使用的字体。

第15学时 实现用户窗体的自动功能

实现用户窗体的自动功能可以分为几个部分：首先必须确定窗体在装载时的外观，其次要确定控件的行为，再次要测试用户输入的数值是否可以接受，最后是把数据写入到工作表中。这就是在这个学时中要完成的任务。

这个学时的重点包括：

- 在什么位置怎样去初始化用户窗体的值
- 使用VBA代码来显示用户窗体，在用户窗体运行时控制它的行为
- 执行数据合法有效性检查
- 把窗体上的数据移动到工作表中

15.1 初始化用户窗体中的值

当显示窗体时，在装载它时肯定需要某些初始化工作。例如，如果在 Date文本框中显示当天的日期的话，用户很可能就欣赏这个界面了。为了实现这一点，需要把一些代码放置到用户窗体_Activate过程中，该过程在窗体被激活时执行，也就是在这个过程中编写合适的代码来初始化窗体上控件的值。将要在这个过程中完成的一项任务是使用一些单元格来填充列表框，Expense Type列表框和Card Type列表框需要一些值，要输入这些值的话，请执行下面的步骤：

1) 切换到当前工作簿，进入到工作簿的 Sheet2工作表，把Sheet2改名为Lists。

2) 把光标移动到单元格 A1，输入Expense Categories。

3) 从单元格B2开始，输入下面的值：

单元格B2: Beach Umbrellas

单元格B3: Bike Rental

单元格B4: Golf Lesson

单元格B5: Golf, 18 holes

单元格B6: Golf, 9 holes

单元格B7: Pool Towels

单元格B8: Tennis Court

单元格B9: Tennis Lesson

4) 选择范围B2:B9，把这个范围命名为Expenses。

要为一个范围的单元格命名的话，选择公式栏左端的“名称”框，输入这些单元的名称，按下回车键就可以了。

5) 把光标移动到单元格 E1，输入Credit Card Types。

6) 从单元格F2开始，输入下面的值：

单元格F2: American Express

单元格F3: Diner's Club

单元格F4: Mastercard

单元格F5: Visa

7) 选择范围 F2:F5 , 把它命名为CardType。

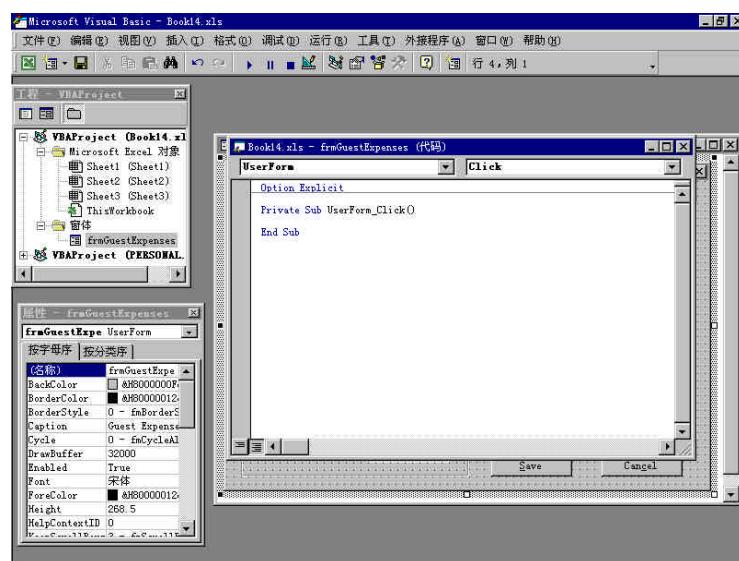
8) 保存工作簿。

9) 按下 Alt+F11 组合键返回到 Visual Basic 编辑器中。

在这个过程中要完成的下一个任务是为窗体上的某些控件提供初始化值和初始化设置，要初始化用户窗体中的值的话，请执行下面的步骤：

1) 双击用户窗体的灰色背景，打开代码窗口，如图 15-1 所示，该窗口中显示了用户窗体_Click 过程。

图 15-1 代码窗口是用来创建和修改用户窗体代码的编辑器



2) 不需要在用户窗体_Click 过程中编写代码，而是要选择在 Activate 事件中编写代码。要在代码窗口中显示 Activate 事件的话，单击代码窗口右上端列表框的向下箭头，这个列表框叫做事件框，从中找到并选择 Activate，用户窗体_Activate 过程就显示出来了。

3) 在这个过程中输入下面的代码：

```

With lstExpenseType
    .RowSource = "Expenses"
    .ListIndex = 0
End With

txtDate.Text = Format(Now, "mm/dd/yy")

With lstCardType
    .RowSource = "CardType"
    .ListIndex = 0
End With

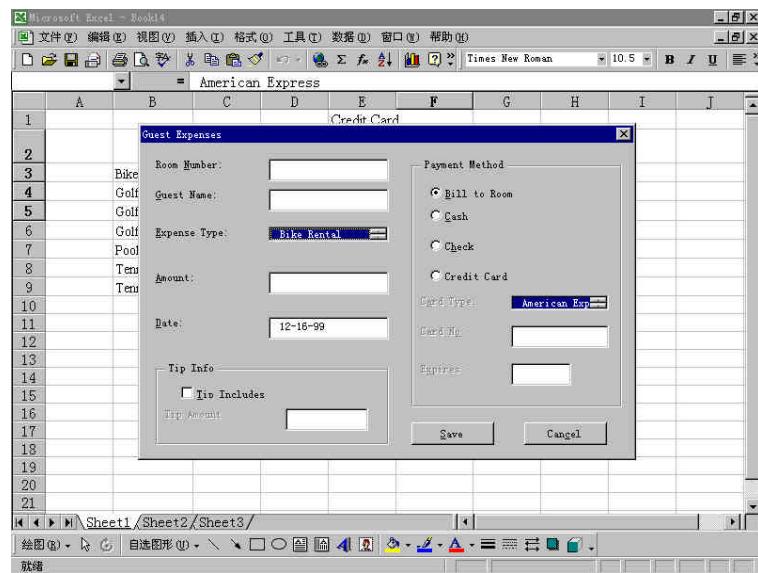
```

4) 按下 F5 键来运行该过程，用户窗体将如图 15-2 所示显示出来，注意到列表框中填充了

一些值，而且在Date文本框中显示了当天的日期。

图15-2 使用窗体的Activate

事件能够为用户自动地提供一些初始化值和初始化设置



5) 关闭该窗体，返回到Visual Basic编辑器。

对窗体中的某些值进行初始化的关键就是使用窗体的Activate事件，如上面步骤介绍的那样。使用类似的代码，就能够初始化选项按钮、复选框的值，也能够初始化列表框、文本框的初始值，初始化标签的标题等等。在用户窗体_Activate过程中，使用了几个列表框和一个文本框，程序清单15-1给出了这个过程的完整代码。

程序清单15-1 用户窗体_Activate过程的完整代码

```

1: Private Sub UserForm_Activate ()
2:   With lstExpenseType
3:     .RowSource = "expenses"
4:     .ListIndex = 0
5:   End With
6:
7:   txtDate.Text = Format(Now, "mm/dd/yy")
8:
9:   .With lstCardType
10:    .RowSource = "CardType"
11:    .ListIndex = 0
12:  End With
13: End Sub

```

在程序清单中，两个With语句都是用来处理列表框的。以第一个With语句为例，可以看到设置了两个属性：

```

With lstExpenseType
  .RowSource = "expenses"
  .ListIndex = 0
End With

```

RowSource属性引用了一个单元格范围，就是在这个范围中存放了用来填充到列表框的值。

当使用单元格的值来填充列表框时，最好使用命名了的单元格范围，因为这对于开发人员来说更容易。

ListIndex属性决定了列表框中哪一个值显示出来，列表框中值的索引是从0开始计数的。

该过程所初始化的另外一个对象类型是文本框：

```
txtDate.Text = Format( Now, "mm/dd/yy")
```

Now函数是用来返回当前日期的，Format函数把日期按照“mm/dd/yy”（月/日/年）的格式来显示。

15.2 显示用户窗体

你也许在考虑不从Visual Basic编辑器中运行用户窗体的话，该怎样来显示它呢？下面就将编写一个简单的过程来实现用户窗体的显示。可以把这个过程分配给各种不同的对象，比如分配给命令按钮、工具栏按钮和菜单命令项等等。

有关使用工具栏的更多信息，请参阅第16学时。有关使用菜单的更多信息，请参阅第17学时。

因为把过程分配给工具栏和菜单命令项在本书中还没有讨论到，所以现在准备把用来显示窗体的过程分配给命令按钮。用来显示对话框的过程只需要一行代码：

```
frmGuestExpenses.Show
```

用户窗体的Show方法把窗体装载到内存中并显示它。要创建这个过程的话，请执行下面的步骤：

1) 如果还没有准备好，那么请切换到Visual Basic编辑器，添加一个新模块到工作簿中。

2) 创建一个新过程，把它命名为ShowGuestExpenses。

3) 在这个过程中输入下面的代码：

```
frmGuestExpenses.Show
```

4) 切换到工作簿的Sheet1工作表，添加一个命令按钮到这个工作表上。

要添加命令按钮的话，必须让“窗体”工具栏显示出来。

5) 当“指定宏”对话框显示出来时，选择ShowGuestExpenses。

6) 把命令按钮的标题设置为Show Guest Expenses，然后在命令按钮之外的任意位置单击以不选中该命令按钮。

7) 要显示窗体的话，单击该命令按钮，窗体就显示出来了，注意到两个列表框控件和Date文本框控件都已经初始化了。

8) 关闭窗体并保存工作簿。

现在你知道了，使用Show方法来显示窗体是很容易的。在本书以后的章节里，将介绍从工具栏按钮和菜单命令来使用ShowGuestExpenses过程显示窗体的方法。

15.3 控制窗体的行为

在显示窗体以后，可能需要窗体具有不同的行为。例如，当选中Tip Included复选框时，

就需要启用 Tip Amount 文本框供用户使用。又比如，如果 Credit Card 选项按钮被选中的话，那么就应该启用跟信用卡信息相关联的所有控件供用户使用。要实现窗体的不同行为的话，需要在合适控件的 change 事件过程中编写代码。当控件的 Value 属性改变时，Change 事件就触发了。因为将使用选项按钮（Credit Card）和复选框（Tip Included），所以当 Value 属性为 True 时，就需要启用相关的控件供用户使用。要输入代码以启用控件的话，请执行下面的步骤：

- 1) 如果还没有准备好，那么请切换到 Visual Basic 编辑器，打开 frmGuestExpenses 窗体。
- 2) 双击 Tip Included 复选框以打开代码窗口。
- 3) 从代码窗口的“事件”列表框中选择 Change，在 chkTipIncluded_Change 过程中输入下面的代码：

```
If chkTipIncluded.Value = True Then  
    lblTipAmount.Enabled = True  
    txtTipAmount.Enabled = True  
Else  
    lblTipAmount.Enabled = False  
    txtTipAmount.Enabled = False  
End If
```

- 4) 位于代码窗口左上角的是一个列表框，该列表框中当前选中的是 chkTipIncluded。单击列表框的向下箭头，查找到 optCreditCard 并选中它。
- 5) 从事件列表框中选择 Change 事件，于是 optCreditCard_Change 过程就在代码窗口中显示出来了。

6) 在 optCreditCard_Change 过程中输入下面的代码：

```
If optCreditCard.Value = True Then  
    lblCardType.Enabled = True  
    lstCardType.Enabled = True  
    lblCardNumber.Enabled = True  
    txtCardNumber.Enabled = True  
    lblExpires.Enabled = True  
    txtExpires.Enabled = True  
Else  
    lblCardType.Enabled = False  
    lstCardType.Enabled = False  
    lblCardNumber.Enabled = False  
    txtCardNumber.Enabled = False  
    lblExpires.Enabled = False  
    txtExpires.Enabled = False  
End If
```

- 7) 切换到工作簿的 Sheet1 工作表，单击该工作表上的命令按钮，窗体就显示出来了，如图 15-3 所示。

- 8) 选中 Tip Included 复选框，启用 Tip Amount 控件。
- 9) 选择 Credit Card 选项按钮，启用信用卡控件，如图 15-4 所示。
- 10) 关闭窗体并保存该工作簿，然后返回到 Visual Basic 编辑器。

图15-3 当窗体开始显示时，
有些控件是禁用的

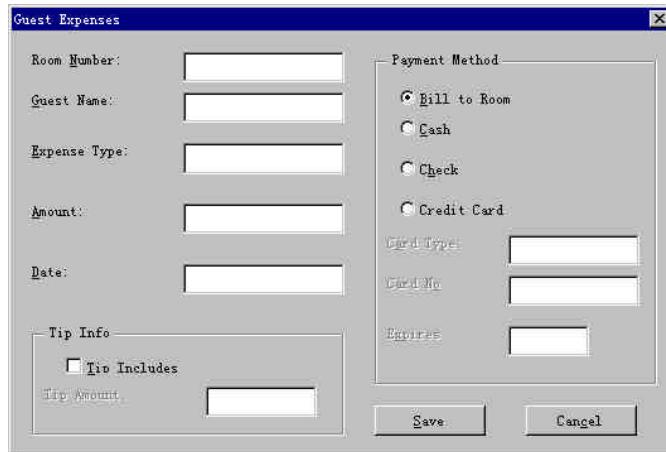
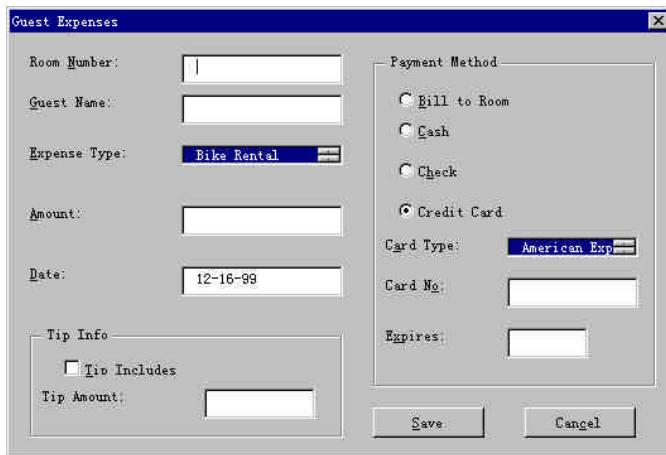


图15-4 使用某个控件的
Change事件就能
够启用其他控件



从根本上看，刚才在 Change事件过程中所做的其实就是测试 Value属性，如果该属性设置为True，那么就把合适控件的Enabled(启用)属性设置为True。如果Value属性设置为False的话，那么就把Enabled属性设置为False。

在运行时并不只限于使用控件的 Enabled属性，还可以设置任意属性，比如 Text、Value、Caption等等。

15.4 数据有效性验证

用户在输入信息到窗体后，就准备单击 Save按钮了。单击按钮后需要完成哪些任务呢？跟Save按钮关联的代码需要完成两件事情：首先是确保输入数据的有效性。其次，如果数据是合法有效的话，就把数据写入到工作表中。现在就来介绍数据有效性验证。

进行数据有效性验证，就是确保数据在应用程序环境中是合法有效的。文本框可以接受从键盘输入的任何内容，但是，这并不意味着输入的数据就是有效的。以 Guest Expenses窗体为例，只有一个控件需要进行数据有效性验证，就是 txtRoomNumber控件。出于举例说明的目的，假定旅馆的房间编号从 101开始，到 730号结束。这并不意味着在有

效性验证实用程序中不再需要任何的代码来进行验证，仍然必须添加代码，以确保必须输入数据的字段都有数据输入。在这个示例中，必须输入数据的字段是 Room Number、Expense Type、Amount和Date。Payment Method也必须输入数据，不过这个字段是通过选择选项按钮来输入数据的。

但是还有更多有效性验证任务需要完成。如果选中 Tip Included复选框的话，就需要向 Tip Amount中输入数据。另外，如果选中 Credit Card选项按钮的话，就需要输入跟信用卡相关的信息。

既然知道了跟 Save按钮相关联的代码需要完成的工作，那么现在就开始编写该过程的代码。请执行下面的步骤：

- 1) 在Visual Basic编辑器中显示用户窗体。
- 2) 双击Save按钮，cmdSave_Click过程就显示出来了。
- 3) 在cmdSave_Click过程中输入下面的代码：

```
If Val(txtRoomNumber.Text) <101 Or Val(txtRoomNumber.Text) > 730 Then
    MsgBox "Invalid room number. "
    txtRoomNumber.SetFocus
    Exit Sub
End If

If txtGuestName.Text = " " Then
    MsgBox "Please enter guest's name. "
    txtGuestName.SetFocus
    Exit Sub
End If

If chkTipIncluded.Value = True Then
    If txtTipAmount = " " Then
        MsgBox "If tip is included you must enter amount. "
        txtTipAmount.SetFocus
        Exit Sub
    End If
End If

If txtAmount.Text = " " Then
    MsgBox "Please enter amount. "
    txtAmount.SetFocus
    Exit Sub
End If

If IsNumeric(txtAmount.Text) = False Then
    MsgBox "Amount must be a number. "
    txtAmount.SetFocus
    Exit Sub
End If

If txtDate.Text = " " Then
    MsgBox "You must enter a date. "
```

```

txtDate.SetFocus
Exit Sub
End If

If optCreditCard.Value = True Then
  If txtCardNumber = " " Then
    MsgBox "Please enter a card number"
    txtCardNumber.SetFocus
    Exit Sub
  End If

  If txtExpires.Text = " " Then
    MsgBox "Please enter the card's expiration date."
    txtExpires.SetFocus
    Exit Sub
  End If
End If

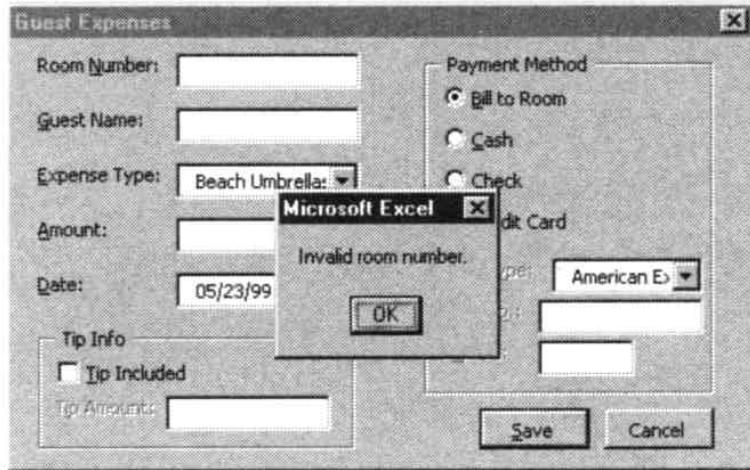
Unload Me

```

4) 切换到Sheet1工作表，单击命令按钮，窗体就显示出来了。

5) 单击Save按钮，一个消息对话框显示出来，告诉用户必须输入房间编号，如图 15-5 所示。

图15-5 cmdSave_Click过
程可以生成多种可
能的错误消息对话
框，图中显示的是
其中的一种



6) 单击“OK”按钮关闭对话框，输入房间编号 121，单击Save按钮。将显示另外一个消息对话框，单击“OK”按钮关闭对话框。

7) 在Guest Name文本框中输入自己的名字，在Amount文本框中输入20。

8) 单击Tip Included复选框，显示的消息对话框告诉用户必须输入小费的数量。单击“OK”按钮关闭对话框。

9) 在Tip Amount文本框中输入2，单击Save按钮，该窗体就不再显示任何对话框了。

你很可能已经看到为cmdSave_Click过程创建代码的一种明确形式，实际上就是测试了控件的Text或者Value属性。程序清单 15-2 给出了在这个过程中使用到的一条 If 语句。

程序清单 15-2 测试房间号的输入是否有效

```
1: If Val(txtRoomNumber.Text) < 101 Or Val(txtRoomNumber.Text) > 730 Then  
2:   MsgBox"Invalid room number."  
3:   txtRoomNumber.SetFocus  
4:   Exit Sub  
5: End If
```

这条If语句就确保了输入的房间号必须是101~730之间的数字。注意，程序清单中使用了Val函数。文本框控件的Value属性的内容实际上是一个字符串，Val函数能够把字符串转换为数字。你是否已经注意到并没有编写代码来检查txtRoomNumber.text是否为空，这是因为如果该文本框为空的话，那么其值就小于101，If语句已经把这种情况包括进来了。

SetFocus方法把命名的控件设置为窗体上的活动对象。在合适控件激活以后，使用了Exit Sub语句来退出过程，这是因为如果发生错误的话，就没有必要继续执行过程了。

程序清单中的If语句给出了对输入控件的数据进行有效性验证的一个基本模板。如果没有任何非法无效的输入数据的话，就可以使用Unload语句把窗体从内存中清除，不再显示了。

15.5 使用返回值

cmdSave_Click过程还有一个问题要解决：怎样从窗体的输入得到数据供写入工作表呢？下面将准备使用窗体上控件的Text和Value属性的内容，然后把这些属性的内容写入到工作表。执行下面的步骤来修改cmdSave_Click过程：

1) 切换到工作簿的Sheet3（如果工作簿还没有Sheet3的话，就添加一个），把Sheet3改名为Guest Expenses。

2) 切换到Visual Basic编辑器，显示用户窗体。

3) 双击Save按钮以显示代码窗口。

4) 在cmdSave_Click过程中的最后一条语句之前插入下面的代码。一定要把这些代码放置在Unload Me语句的前面，这一点很重要。

```
Worksheets("Guest Expenses").Activate  
Range("A2").Select  
If Range("A2").Value = " " Then  
  Range("A2").Activate  
Else  
  Range("A2").CurrentRegion.Select  
  ActiveCell.Offset(Selection.Rows.Count,0).Activate  
End If  
With ActiveCell  
  .Value = txtRoomNumber.Text  
  .Offset(0,1).Value = txtGuestName.Text  
  .Offset(0,2).Value = lstExpenseType.Text  
  .Offset(0,3).Value = txtAmount.Text  
  .Offset(0,4).Value = txtDate.Text  
  If chkTipIncluded.Value = True Then  
    .Offset(0,5).Value = txtTipAmount.Text  
  End If  
  If optBillToRoom.Value = True Then  
    .Offset(0,6).Value = "Room"
```

```

ElseIf optCash.Value = True Then
    .Offset(0,6).Value = "Cash"
ElseIf optCheck.Value = True Then
    .Offset(0,6).Value = "Check"
ElseIf optCreditCard.Value = True Then
    .Offset(0,6).Value = "Credit card"
    .Offset(0,7).Value = lstCardType.Text
    .Offset(0,8).Value = txtCardNumber.Text
    .Offset(0,9).Value = txtExpires.Text
End If
End With

```

5) 切换到工作簿的Sheet1。

6) 保存工作簿，在运行需要输入数据的过程之前保存工作簿总是一个好主意。

7) 单击 Guest Expenses按钮，输入房间号 430，输入自己的名字到 Guest文本框中，选择 Golf Lesson作为消费类型，在 Amount文本框中输入 100，选中 Tip Include复选框，在 Tip Amount文本框中输入 10，选择 Credit Card选项按钮，选择 Visa作为信用卡类型，卡号输入 112233，当信用卡到期时就输入 1/1/00。单击Save按钮。

窗体上的这些输入数据都写入到 Guest Expenses工作表中了，如图 15-6所示。现在花一些时间来浏览一下所输入的代码。刚才示例中所做的第一件事情是确定在工作表上开始输入数据的位置：

```

Worksheets("Guest Expenses").Activate
Range("A2").Select
If Range("A2").Value = " " Then
    Range ("A2").Activate
Else
    Range("A2").CurrentRegion.Select
    ActiveCell.Offset(Selection.Rows.Count,0).Activate
End If

```

Guest Expenses工作表被激活了，然后是区域 A2被激活了。区域 A2就是用户开始输入数据的位置，因为这张工作表的第一行包含了标题。接下来一步是测试以查看区域 A2中是否有值，如果没有的话，A2就是输入数据的开始位置了。否则的话，就必须找到最后那个单元格，这就是CurrentRegion开始活动的位置。通过选择当前的区域，可以确定已经输入了数据的行数，利用这个数字就可以找到用户开始输入数据的位置。

在获得了输入数据的开始位置以后，使用 Offset属性在行之间输入数据。如果 Credit Card选项按钮被选中的话，还将有额外输入要处理。

窗体还有一件事情要做，就是必须为 Cancel按钮编写代码，请执行下面的步骤：

1) 切换到 Visual Basic编辑器，双击 Cancel按钮以打开代码窗口。

2) 在cmdCancel_Click过程中输入下面的代码：

Unload Me

3) 保存工作簿。

如果用户单击了 Cancel按钮，那么就把窗体卸载，使得窗体不再显示，并且从内存中清除了。在实际应用中，很可能显示一个消息对话框来询问用户是否真的要取消。

15.6 学时小结

现在已经创建了一个功能完整的对话框，该对话框能够执行数据有效性验证，并且能够把有效的数据返回到工作表中。

在前一个学时中，我们集中介绍了用户窗体的设计。这个学时中集中介绍的是实现窗体的自动化功能，你已经从头到尾学习了整个窗体自动化的过程，知道怎样显示窗体、懂得怎样初始化窗体的值、怎样对窗体的输入进行有效性验证、怎样把用输入的数据写入到工作表中。

15.7 专家答疑

问题：怎样知道要把控件的哪个属性写入到工作表中呢？

解答：下面是总结的内容：

控件	属性
标签	Caption
文字框	Text
列表框	Text
组合框	Text
选项按钮	Value
复选框	Value
微调按钮	Value

问题：把列表框使用到的值放置到工作表上，如果不想用户修改或者浏览这些值的话，该怎么办呢？

解答：有几种选择。其一是通过“格式”、“工作表”菜单来隐藏工作表。另外一种选择是在工作簿的Auto_Open过程中把工作表的Visible属性设置为False。

15.8 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

15.8.1 思考题

- 1) 怎样禁用控件？
- 2) 怎样显示窗体？
- 3) 在哪个过程中放置窗体的初始化代码？
- 4) 怎样把窗体从内存中清除掉？
- 5) 如果要通过代码返回到控件，应该采用什么方法？
- 6) 控件使用的值（例如列表框等使用）应该存储在什么地方？
- 7) 列表框的哪个属性控制了列表框中显示的项？
- 8) 列举出放置数据有效性验证代码的几个位置。

15.8.2 练习题

创建一个命名为ShowSplash的过程，该过程需要显示frmSplash窗体。

为frmSplash上的cmdOK按钮输入必要的代码，以显示 frmGuestExpenses窗体。

切换到Guest Expenses工作簿的Sheet1，把ShowSplash分配给那个窗体上的命令按钮。运行并测试应用程序。

第16学时 工 具 栏

这个学时将介绍两个主题：手工使用工具栏和通过代码来使用工具栏。工具栏作为绝大多数用户访问Excel命令的界面是很重要的。你可以创建自己的工具栏按钮，然后把不同的过程分配给不同的工具栏按钮，从而为执行自己的宏提供一个熟悉和方便的访问方式。

这个学时的重点是：

- 手工添加按钮到工具栏上
- 不使用代码来创建新工具栏
- 通过代码来使用工具栏

16.1 手工修改工具栏

笔者发现：只有很少的Excel用户懂得怎样创建新工具栏、懂得怎样把按钮添加到工具栏上。工具栏显然是Excel应用程序窗口中最显眼的部分了。到现在为止，你已经使用了两种方法来执行自己的过程（不包括按下F5键和从“运行”菜单来执行过程的方法）。当宏分配给某个控件后，可以执行它。也可以通过选择“工具”、“宏”、“宏”来执行宏。这两种方法都已经使用过。如果不想使用控件来执行宏命令，也不准备通过“工具”菜单来运行宏命令的话，可以把宏过程分配给工具栏上的某个按钮。

下面要做的第一件事情是向现成的工具栏上添加一个工具栏按钮，Microsoft提供了几个图像，可以作为工具栏按钮的图标，另外也可以自己创建图像作为按钮的图标。要把工具栏按钮添加到工具栏上、并且把宏分配给按钮的话，请执行下面的步骤：

1) 打开Guest Expenses工作簿。

2) 用鼠标右键单击工具栏，从弹出的菜单中选择“自定义”，“自定义”对话框就显示出来，如图16-1所示。

图16-1 “自定义”对话框

可以用来自定义工
具栏和菜单



3) 从“命令”标签上的“类别”列表框中选择“宏”，在“命令”列表框中将看见“自定义按钮”这个短语。

4) 把“自定义”按钮拖放到现成的一个工具栏上。当移动到工具栏上时，可以发现有一个黑条跟着工具栏上的拖动按钮的移动而移动。这个黑条就表示当释放鼠标按钮时放置工具栏按钮的位置。

5) 当找到了新工具栏按钮的满意位置时，就可以释放鼠标按钮，于是按钮就添加到了工具栏上，如图16-2所示。

6) 用鼠标右键单击这个新的工具栏按钮，从弹出的菜单中选择“命名”，输入Guest Expenses作为该按钮的名称，输入的名称将用来作为按钮的“工具栏提示”。

7) 再次用鼠标右键单击这个新工具栏按钮，选择“改变按钮图标”，这时将显示出许多小图像供选择，如图16-3所示。选择中意的图像，按钮的图标就改变成为选中的图像了。

图16-2 新的宏按钮的默认图标是一个笑脸图像

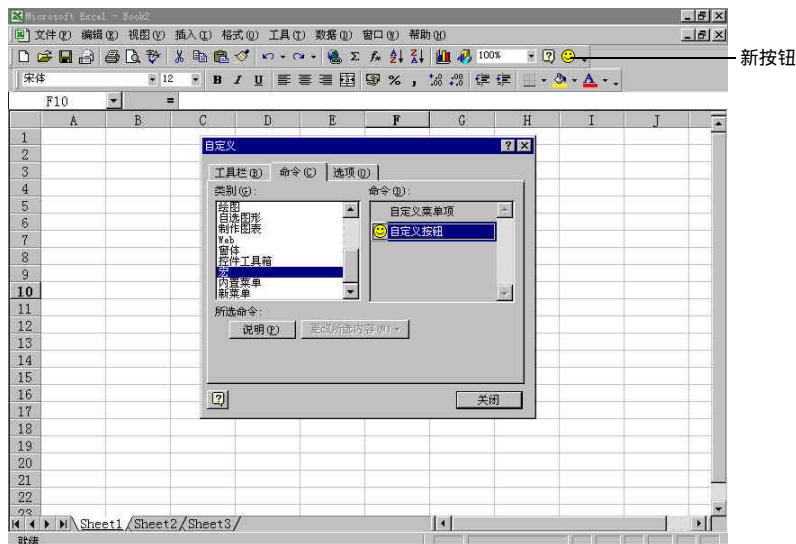
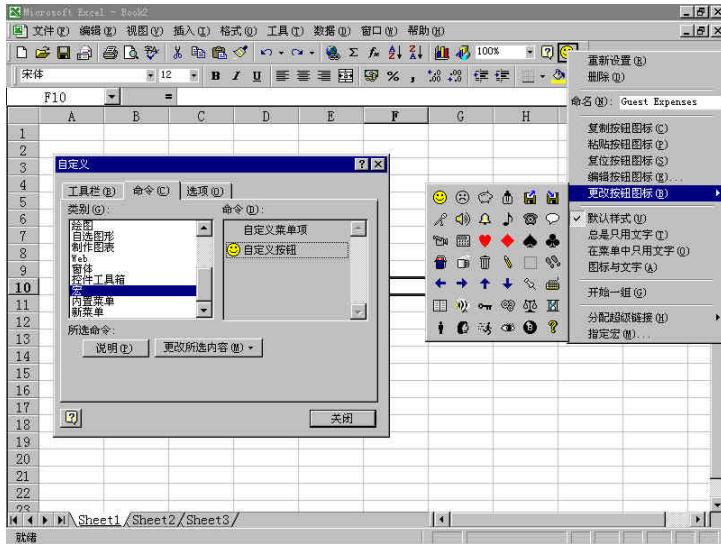


图16-3 Excel提供了一些图像供选择用来作为宏按钮的图标



8) 再次用鼠标右键单击新按钮，选择“指定宏”，把ShowGuestExpenses分配给这个新按钮。

9) 关闭“自定义”对话框。

10) 单击新的Guest Expenses工具栏按钮，Guest Expenses窗体就显示出来。

11) 单击“取消”按钮以关闭窗体。

现在所有工作簿都可以使用这个新工具栏按钮，这就意味着，如果在另外一个工作簿中单击了这个按钮的话，Guest Expenses对话框就会显示出来。实际上，使用刚才介绍的技术来把宏过程分配给工具栏按钮的方法只适用于在所有的工作簿中都使用宏过程的应用场合。例如，位于个人工作簿中的宏和过程就很适合分配给工具栏按钮。

因为上面这个缘故，你很可能想要删除这个工具栏按钮，要实现这一点的话，请执行下面的步骤：

1) 用鼠标右键单击工具栏，从弹出的菜单中选择“自定义”。

2) “自定义”对话框打开以后，就能够删除按钮了。把Guest Expenses工具栏按钮拖回到“自定义”对话框，这样就把按钮从工具栏上删除了。

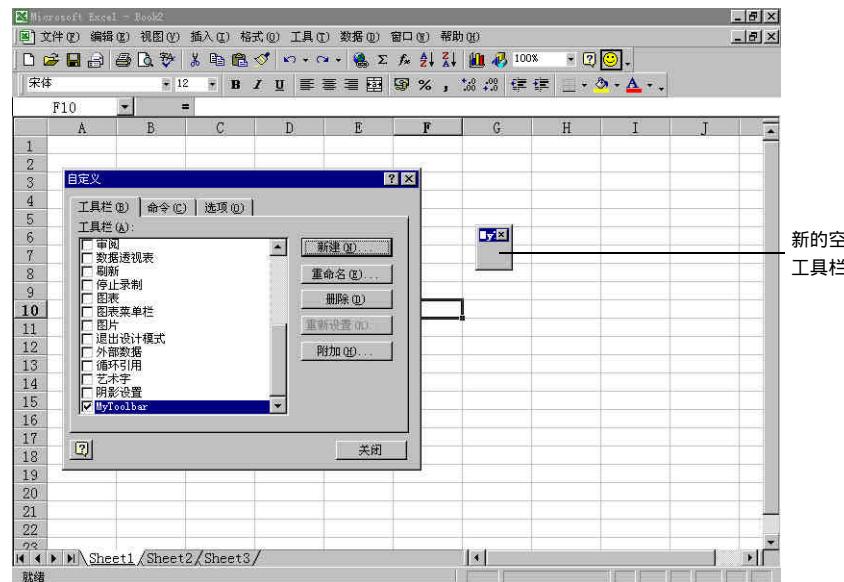
3) 关闭“自定义”对话框。

如果你已经对内置的工具栏做出了一些改变，而现在又想撤销全部改变的话，那么请切换到“自定义”对话框的“工具栏”选项卡，从“工具栏”列表框中选择要撤销所有改变的工具栏，单击“重新开始”按钮，工具栏就重置了，又跟刚安装完Excel时的工具栏完全一样了。

16.2 创建自定义工具栏

并不是只限于往现成的工具栏上添加工具栏按钮，还可以创建自定义工具栏。例如，也许需要创建一个自定义工具栏，可以用它来执行应用程序的所有宏命令和过程。在这个学时的后面部分，将学习到怎样显示和隐藏工具栏，这意味着可以控制能够使用工具栏的时机，从而可以控制工具栏按钮可以使用的时机。要创建新的自定义工具栏的话，请执行下面的步骤：

图16-4 当工具栏刚创建时，工具栏是空的



- 1) 用鼠标右键单击工具栏。
- 2) 从弹出的菜单中选择“自定义”。
- 3) 选择“工具栏”选项卡。
- 4) 单击“新建”，“新建工具栏”对话框显示出来。
- 5) 输入MyToolbar作为新工具栏的名称，然后按回车键。
- 6) 单击“关闭”按钮关闭该对话框。

新工具栏显示时是一个小的灰色框，如图 16-4所示。当向新工具栏上添加按钮时，工具栏的大小会随之增加。可以使用前面介绍过的技术向自定义工具栏上添加工具栏按钮。

16.3 通过代码来使用工具栏

通过使用VBA，能够完成创建工具栏、向工具栏上添加按钮、显示/隐藏工具栏和按钮、删除工具栏和按钮等所有功能。通过代码来创建和维护工具栏有一个优势：通过代码允许在工作簿打开的时候创建工具栏，而在工作簿关闭的时候删除工具栏。这意味着工具栏和工具栏上的按钮只有在工作簿打开的时候才可以使用。

16.3.1 创建工具栏

你想到使用VBA代码来创建工具栏的第一个方案可能是使用 Add方法在 Toolbars集合中创建新的工具栏，这个方案对了一半。我们是准备使用 Add方法，但是这里并没有像 Toolbars集合这样的对象存在。工具栏跟菜单项和加速键菜单组成了一个组，是作为 CommandBars（命令项）引用的。可以把 CommandBars看作命令的集合，这样你就能够理解工具栏和菜单为什么能够位于一个组了。因此，要创建工具栏的话，就要使用 CommandBars集合的Add方法。

创建了一个CommandBar对象以后，下一步就是添加工具栏按钮了，要实现这一点，就要使用Controls（控件）集合的Add方法。要创建带有按钮的新工具栏，请执行下面的步骤：

- 1) 打开Guest Expenses工作簿。
- 2) 按下Alt+F11组合键以打开Visual Basic 编辑器。
- 3) 创建一个命名为CustomTB的新过程。
- 4) 在这个新过程中输入下面的代码：

```
Dim ctlGEButton As Object  
ThisWorkbookActivate  
  
Application.CommandBars.Add Name:="GuestTB"  
CommandBars("GuestTB").Visible = True  
Set ctlGEButton = Application.CommandBars("GuestTB").Controls.Add_  
(Type:=msoControlButton, ID:=2950, Before:=1)  
With ctlGEButton  
    .FaceId = 2141  
    .OnAction = "ShowGuestExpenses"  
End With
```

- 5) 运行这个过程，就创建了新的工具栏。
- 6) 单击GuestTB工具栏上的按钮，Guest Expenses对话框就显示出来了。
- 7) 单击“取消”按钮关闭这个对话框。

如果要再次运行这个过程，就必须删除 GuestTB工具栏，可以用鼠标右键单击工具栏、然后选择“自定义”，从“工具栏”选项卡中把工具栏删除。

现在，我们来对刚才的代码进行分析。代码所做的第一件事情是创建一个对象变量：

```
Dim ctlGEButton As Object
```

接下来激活工作簿，因为要创建的工具栏是供工作簿使用的：

```
ThisWorkbook.Activate
```

接下来创建工具栏，并且把工具栏的 Visible属性设置为 True，以便工具栏能够显示出来：

```
Application.CommandBars.Add Name := " GuestTB"  
CommandBars(" GuestTB"). Visible = True
```

之后，代码在窗体上创建了一个按钮，并且把这个按钮设置成为控件按钮，使用了 ID参数来应用这个按钮，“自定义宏”按钮的ID是2950。

```
Set ctlGEButton = Application. CommandBars (" GuestTB"). Controls.Add _  
( Type := msoControlButton, ID := 2950
```

Type参数有几个不同的值可供选择。下面每个参数值都对应于可以创建的工具栏按钮的一种类型：

- msoControlButton
- msoControlEdit
- msoControlDropdown
- msoControlGomboBox
- msoControlPopup

接下来，代码设置了按钮的一些属性，尤其是设置了 FaceID和OnAction属性的值。FaceID设置的是按钮的图标，OnAction属性把宏分配给按钮：

```
With ctlGEButton  
.Faceld = 2141  
.OnAction = "ShowGuestExpenses"  
End With
```

你也许想知道笔者是怎样找到 FaceID属性所需要的号码，当然是使用代码来实现的。笔者首先在GuestTB工具栏上创建了一个要使用的带有图标的按钮，然后运行了如程序清单 16-1 中所示的过程。

程序清单 16-1 FindOut过程的代码

```
1: Sub FindOut ()  
2:   MsgBox CommandBars("GuestTB").Controls(1).Faceld  
3: End Sub
```

如果想在打开工作簿时创建工具栏，就需要把代码放置在 Auto_Open过程中。可以在第24学时中查找到有关Auto_Open过程的更多信息。

16.3.2 删除工具栏

如果不再需要工具栏作为 Excel环境下的一个组成部分时，就可以使用 Delete方法把工具栏删除。下面的步骤创建了用来删除 GuestTB 工具栏的过程：

1) 创建一个命名为DeleteGuestTB的新过程。

2) 在这个过程中输入下面的代码：

```
ThisWorkbookActivate  
CommandBars("GuestTB").Delete
```

3) 运行这个过程，就删除了 GuestTB工具栏。

16.4 学时小结

工具栏是用户同 Excel和应用程序进行交互的界面中最受欢迎的方式之一，既可以使用 Excel的内置特性来创建自定义工具栏，也可以使用 VBA来创建、修改和删除为自己的应用程序方案专门设计的工具栏。

在下一个学时中，将使用 Excel界面的另外一个重要部分，即菜单系统，将学习用来操作内置菜单和从一开始就使用代码来创建菜单的几项技术。

16.5 专家答疑

问题：当浏览“对象浏览器”时，没法找到命名为 Toolbar（工具栏）的对象，这是为什么？

解答：工具栏实际上是一种叫做 CommandBar类型的对象，另外一个 CommandBar对象的例子是菜单项。

问题：如果工具栏不想使用预先定义的图像做图标时该怎么办？我能够创建自己的图像做图标吗？

解答：可以，Excel提供了一个按钮编辑器。要访问按钮编辑器的话，用鼠标右键单击工具栏，从弹出菜单中选择“自定义”。当“自定义”对话框打开时，用鼠标右键单击要修改图标的那个按钮，选择“编辑按钮图标”。

16.6 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

16.6.1 思考题

- 1) 工具栏属于什么集合？
- 2) 使用哪个属性来把宏分配给工具栏按钮？
- 3) 请说出工具栏按钮所属的集合。
- 4) 使用什么方法可以删除工具栏？
- 5) 怎样使用VBA代码来显示工具栏？
- 6) 哪个属性可以用来控制工具栏按钮上显示的图标？

7) 判断题：工具栏在创建时就自动显示出来。

16.6.2 练习题

创建一个命名为 Hour16Toolbar的过程，该过程需要创建和显示一个命名为 Hour16的新工具栏，该工具栏上有 3个按钮：“新建（ID是2520）”，“打开（ID是23）”和“保存（ID是3）”。

创建另外一个命名为 DeleteTB的过程，该过程删除 Hour16工具栏。运行并测试这两个过程。

第17学时 菜单

在前一个学时中，已经学习到工具栏是定义为执行菜单项命令的一种替代方法。既然已经知道了怎样去修改工具栏，你很可能想进一步弄清楚怎样修改菜单。在这个学时中，将学习通过手工和编程方式来使用菜单。

这个学时的重点包括：

- 学习手工修改菜单
- 使用VBA代码来创建菜单栏、菜单和菜单项
- 显示和删除菜单栏

17.1 手工修改菜单

当使用菜单时，实际上使用了三个对象：容器、菜单和菜单项。容器跟工具栏中的一样，叫做命令栏，也叫做菜单栏，位于命令栏中的就是菜单。菜单有“文件”、“编辑”、“帮助”菜单等等，每个菜单中内容的就是菜单项，比如，“文件”菜单包含了像“打开”、“保存”和“打印”之类的菜单项（或者叫做菜单命令）。

当采用手工方式来自定义菜单时，可以添加菜单和菜单项。可以添加预先定义好的菜单和菜单项，也可以创建自己的菜单词条。要创建能够用来执行宏命令的新菜单和菜单项，请执行下面的步骤：

- 1) 打开Guest Expenses工作簿。
- 2) 用鼠标右键单击菜单栏，从弹出的菜单中选择“自定义”，“自定义”对话框显示出来。
- 3) 选择“命令”选项卡，就显示了可以使用的菜单类别和命令的列表，如图 17-1所示。

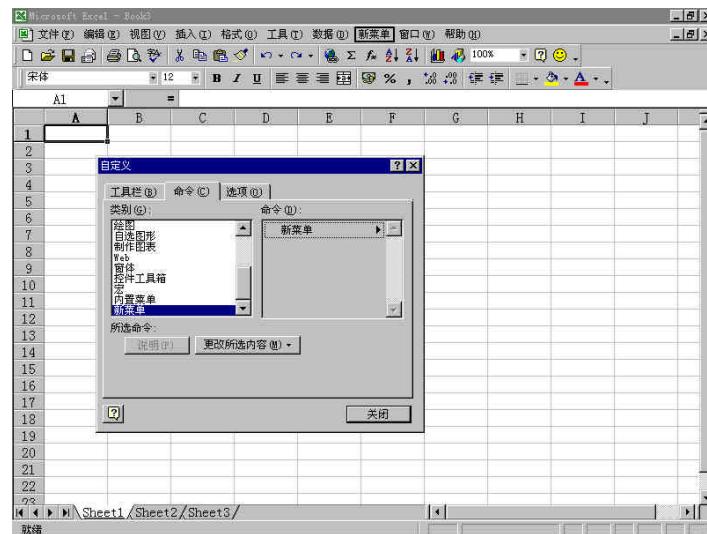
图17-1 在“类别”列表框中，可以看到列出了一些熟悉的菜单



- 4) 在“类别”列表框中，找到并选择“新菜单”，在“命令”列表框中，就将看到“新菜单”。
- 5) 从“命令”列表框中拖拉“新菜单”。我们准备把“新菜单”放置在“数据”和“窗口”菜单之间，当定位合适时，就能够看到这两个菜单之间将出现黑色的竖条。

6) 释放鼠标键，把菜单放置到新位置上，新的菜单如图 17-2所示。

图17-2 当新菜单最初添加
到菜单栏上时，它
的标题是“新菜单”

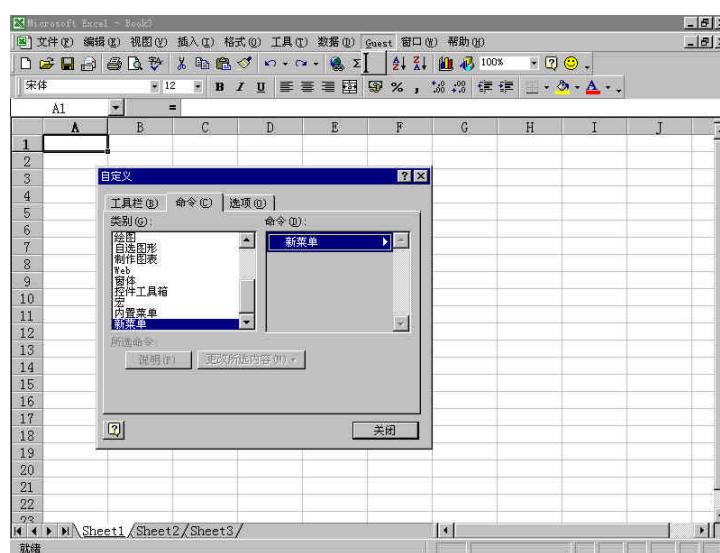


7) 添加了这个菜单以后，下一步是重新给新菜单命名。用鼠标右键单击“新菜单”，选择“名称”，输入&Guest作为该菜单的名称。字母 G前面的&就把该字母设置为这个菜单的加速键。

8) 下一步是添加一个新的菜单项到 Guest菜单中。再次从“命令”列表框中选择“新菜单”，把它拖到Guest菜单。当拖到该菜单上时，Guest菜单下面就显示出一个灰色框，把新菜单拖到这个框中。当定位合适时，这个框中将显示一条竖直方向的黑线，如图 17-3所示。释放鼠标键来创建这个新的菜单项。

从技术上来说，可以给 Guest菜单指定宏，不过，界面设计的通常标准是把宏分配给菜单项而不是菜单，通过菜单项来执行宏。

图17-3 在Guest菜单下面
显示的框表明菜单
现在还是空的



- 9) 用鼠标右键单击“新菜单”，选择“重新命名”。
- 10) 输入&Expenses作为名称。
- 11) 现在可以准备给Expenses菜单项指定宏了。用鼠标右键单击Expenses，从弹出的菜单中选择“指定宏”，“指定宏”对话框显示出来。
- 12) 选择ShowSplash宏，然后单击“确定”按钮。
- 13) 单击“关闭”按钮来关闭“自定义”对话框。

关闭“自定义”对话框以后，就可以准备使用新菜单了。选择Guest, Expenses, Welcome对话框就显示出来，单击“确定”按钮，然后单击“取消”按钮，于是现在又有了一种启动应用程序的方式。

17.2 通过编程方式来使用菜单

使用VBA代码来添加菜单跟添加工具栏非常相似。实际上是把菜单项添加到跟工具栏共用的集合，也就是叫做CommandBars的集合中。添加菜单栏的基本语法如下：

```
CommandBars.Add ( Name, Position, MenuBar, Temporary )
```

Name参数指定了命令栏的名称。令人惊奇的是，Name参数居然是可选的。如果没有给这个参数提供值的话，系统就会给这个命令栏分配一个默认的名称，比如Custom 1。

Position（位置）是另外一个可选的参数，这个参数允许选择新命令栏的位置或者类型。

因为下面要创建的是菜单栏，所以需要把MenuBar参数设置为True。把这个参数设置为True就用新命令栏取代了当前活动的菜单栏。该参数的默认值是False。

最后一个参数是Temporary（临时的），它也是可选的。把这个参数设置为True就使得新命令栏成为临时的。临时命令栏在包容器应用程序关闭时被删除。这个参数的默认值是False。在你的应用程序中，也许想要把这个参数设置为True，这样就不会把新增加的临时菜单栏遗留在Excel环境中。

现在开始学习菜单：我们即将创建新菜单，并显示新菜单来取代工作表菜单。要创建自定义菜单的话，关闭所有打开的工作簿，然后打开一个新工作簿，按下Alt+F11切换到Visual Basic编辑器，在这个新工作簿中插入一个模块，接下来请执行下面的步骤：

- 1) 创建一个新过程，命名为MyFirstMenubar。

- 2) 在这个新过程中输入下面的代码：

```
Dim mybar As CommandBar
```

```
Set mybar = CommandBars.Add(Name: ="Hour17", _  
    Position:=msoBarTop, MenuBar:=True, temporary:=True)  
mybar.Visible = True  
CommandBars("Worksheet Menu Bar").Visible = False
```

- 3) 另外创建一个过程，命名为UndoMyMenu。

- 4) 在这个过程中输入下面的代码：

```
CommandBars ("Hour 17"). Delete
```

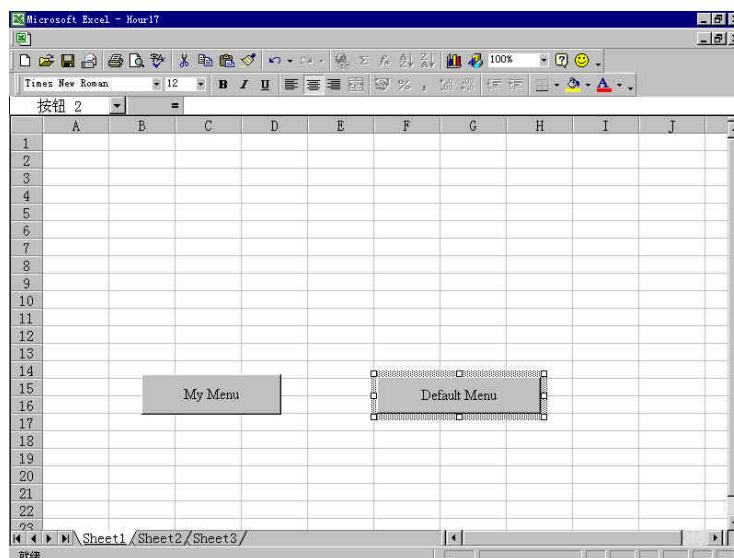
- 5) 切换回工作簿，在Sheet1上创建一个命令按钮，把宏过程MyFirstMenubar分配给这个命令按钮，并把按钮的标题设置为My Menu。

- 6) 另外创建一个命令按钮，把宏过程UndoMyMenu分配给它，并把它的标题设置为

Default Menu。

7) 单击My Menu按钮，就显示出一个空菜单来取代默认的工作表菜单，如图 17-4所示。

图17-4 这个窗口上部的灰色条实际上是一个空的菜单栏



8) 单击Default Menu按钮，工作表菜单又恢复了。

完成了上面介绍的这些步骤以后，你已经知道怎样显示自定义的和内置的菜单了。刚才使用VBA代码不但显示了一个自定义菜单，也创建了该自定义菜单。程序清单 17-1给出了 MyFirstMenu过程的完整代码。

程序清单 17-1 MyFirstMenu过程代码

```

1: Sub MyFirstMenu ()
2:   Dim mybar As CommandBar
3:
4:   Set mybar = CommandBars.Add(Name:= "Hour17",_
      Position:=msoBarTop,MenuBar:=True,temporary:=True)
5:   mybar.Visible = True
6:   CommandBars("Worksheet Menu Bar").Visible = False
7:
8: End Sub

```

在这个过程中所做的第一件事情是创建了一个对象变量，这个对象变量在用来创建菜单栏的Add方法时进行赋值：

```
Dim mybar As CommandBar
```

```
Set mybar = CommandBars. Add ( Name := " Hour 17",_
Position := msoBarTop,MenuBar := True, temporary := True )
```

菜单栏创建以后，通过把 Visible属性设置为True来显示它，同时也把工作表菜单栏的这个属性设置为False，以便不显示工作表的菜单栏。

```
mybar.Visible = True
CommandBars ( " Worksheet Menu Bar " ).Visible = False
```

以上就是创建菜单栏的方法。当不再需要某个菜单栏时，就可以使用 Delete方法来删除它。下一步是向菜单栏上添加菜单。

添加菜单和菜单项

创建菜单的下一步是添加菜单和菜单栏。要实现这一点的话，将使用 Controls集合的 Add方法：

```
NameOfMenu.Controls.Add( Type, Id, Parameter, Before, Temporary )
```

你很可能已经猜测到，NameOfMenu需要用要添加的菜单名称来替换，而 Type参数对于菜单而言，取值应该是 msoControlPopup。

参数Id的取值取决于要添加的菜单类型。如果添加的是自定义菜单，那么 Id取值应该为1。如果是内置菜单，那么 Id取值应该是一个整数，这个整数指定了所希望的菜单。那么，怎样去找到哪个菜单对应哪个整数呢？实际上，最简单的方式是开始录制一个宏，把需要向菜单栏添加的菜单添加进去，记录的菜单就拥有了所需要的那个整数值。

要添加内置菜单到菜单栏的话，用鼠标右键单击菜单栏，从弹出的菜单中选择“自定义”，切换到“命令”选项卡页。在“类别”列表框中找到并选择“内置菜单”，选中以后，内置菜单就在“命令”列表框中显示出来。把所要添加的菜单拖到菜单栏上，该菜单就添加到了菜单栏中，并且其中包括了该菜单的所有菜单项。

Parameter参数可选，它的取值取决于要使用的菜单类型。如果是内置菜单，那么很可能就不需要提供该参数的值。如果是自定义菜单，就可以使用这个参数来把信息发送到 Visual Basic过程中。还可以利用这个参数来存储控件的有关信息。

Before参数也可选，它的取值是一个代表菜单栏上新控件位置的数字。菜单（或者菜单项）将插入到位于所提供的那个控件的前面。如果没有给该参数提供值的话，菜单或者菜单项就添加到最后。

如果想要菜单或者菜单项成为临时的，那么就应该把可选参数 Temporary的值设置为True。该参数的默认值是False。

如果想要把菜单添加到活动的菜单栏上，那么可以使用 CommandBars集合的 ActiveMenuBar属性来返回活动菜单栏的名称，应用举例如下：

```
Set CurrMenuBar = CommandBars.ActiveMenuBar
```

有时候，可能想要自定义自己的菜单，以便能够启动简单的过程，下面的练习将介绍怎样实现这一点，将添加“文件”菜单和一个自定义菜单到 Hour 17菜单栏上。首先，需要修改 MyFirstMenu过程以添加菜单到菜单栏上，然后将添加菜单项到菜单中。现在修改 MyFirstMenu，使得它跟程序清单17-2一样。注意，修改的部分用黑粗体表示。

程序清单17-2 修改后的MyFirstMenu过程代码

```
1: Sub MyFirstMenu ()  
2:   Dim mybar As CommandBar  
3:   Dim mymenu As Object
```

```

4: Dim mymenuItem As Object
5:
6: Set mybar = CommandBars.Add(Name:="Hour17",_
    Position:=msoBarTop,MenuBar:=True,_
    Temporary:=True)
7:
8:
9: mybar.Controls.Add Type:=msoControlPopup, ID:=30002,Before:=1
10:
11: Set mymenu = mybar.Controls.Add(Type:=msoControlPopup,_
    Temporary:=True)
12: mymenu.Caption = "Hour 17"
13:
14:
15: Set mymenuItem = mymenu.Controls.Add(Type:=msoControlButton, ID:=1)
16: mymenuItem.Caption = "Macro Demo"
17: mymenuItem.Style = msoButtonCaption
18: mymenuItem.OnAction = "ShowMe"
19:
20: mybar.Visible = True
21: CommandBars("Worksheet Menu Bar").Visible = False
22: End Sub

```

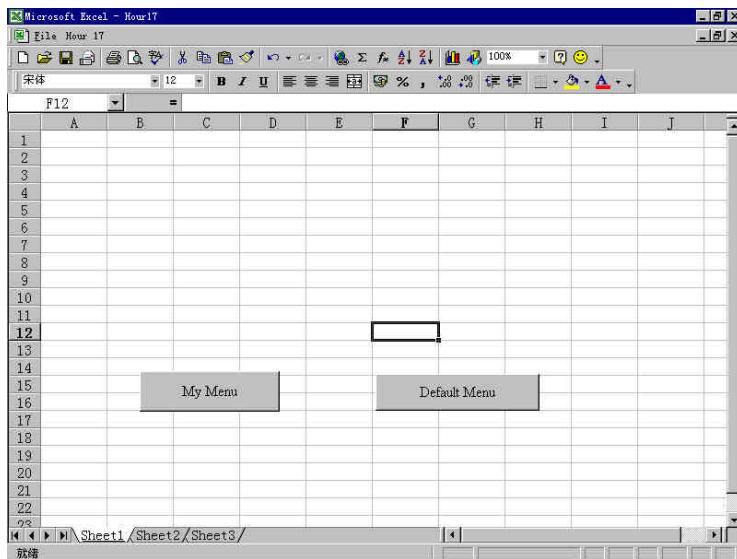
我们需要创建一个新过程，在这个例子中，把它命名为 ShowMe。在这个新过程中输入下面的代码：

```
MsgBox "It Works!"
```

现在，返回到工作簿的 Sheet1，单击 My Menu 按钮来运行程序，现在菜单栏中就有菜单了，如图 17-5 所示。

图 17-5 菜单栏不再是空

的了



选择 Hour 17 菜单，然后选择 Macro Demo（宏演示），将显示消息对话框，单击“确定”按钮以关闭该对话框。接下来单击 Default Menu 按钮以返回到工作表菜单栏，把工作簿另存为 Hour 17。

现在，我们花一些时间来分析对 MyFirstMenu 过程所做出的修改。可以看到代码首先创建

了两个新的对象变量：

```
Dim mymenu As Object  
Dim mymenuitem As Object
```

过程中下一行新修改的代码添加了“文件”菜单。为了获得该菜单 ID属性的取值，笔者录制了一个添加“文件”菜单的宏。“文件”菜单是这个菜单栏上的第一个菜单，所以 Before 参数的取值是1：

```
mybar.Controls.Add Type := msoControlPopup, ID := 30002, Before :=1
```

下一个添加的菜单是自定义类型的：

```
Set mymenu = mybar.Controls.Add ( Type := msoControlPopup, _Temporary := True)  
mymenu.Caption = "Hour 17"
```

添加了自定义菜单以后，又添加了一个自定义菜单项：

```
Set mymenuitem = mymenu.Controls.Add ( Type := msoControlButton, ID :=1)
```

自定义菜单项需要设置它的几个属性，Caption属性的取值就是显示的菜单项文本，Style（样式）设置为msoButtonCaption是因为这个菜单项只有显示的文本。菜单项是没有与之相关的图标的。OnAction属性设置了当菜单项选中时需要运行的那个过程：

```
mymenuitem.Caption = " Macro Demo"  
mymenuitem.Style = msoButtonCaption  
mymenuitem.OnAction = " ShowMe"
```

现在，你已经知道怎样向菜单栏上添加菜单和菜单项了，不管是添加菜单还是菜单项，都要使用Controls集合，关键是要明白把控件添加到哪种类型的对象中去。添加菜单时，是把控件添加到菜单栏上。而添加菜单项时，是把控件添加到菜单中。

17.3 学时小结

通过VBA代码来使用菜单的关键是Add方法，要创建菜单栏、菜单或者菜单项，都使用到Add方法。在这个学时中，我们使用了两个集合：CommandBars集合和Controls集合。现在已经知道，使用菜单的方法跟使用工具栏的方法非常相似，所不同的是参数的取值和属性的设置有区别而已。

17.4 专家答疑

问题：使用工具栏时，使用的是CommandBars集合和Controls集合，现在使用菜单时，仍然使用了这两个同样的集合，这是为什么？

解答：在Excel（和Office）环境中，菜单栏和工具栏两者都是CommandBars集合的成员。菜单、菜单栏和工具栏按钮也都是Controls集合的成员。如果理解了这一点，那么问题就解决了。菜单栏和工具栏是那些用来执行命令的控件的容器。

问题：如果想要创建在应用程序运行时不可使用的菜单或者菜单项，应该怎么办呢？怎样具体实现？

解答：把菜单或者菜单项的Enabled属性设置为False就可以实现。

17.5 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

17.5.1 思考题

- 1) 创建菜单时使用的是什么方法？
- 2) 过程是分配给自定义控件（菜单项）的哪个属性呢？
- 3) 菜单的控件类型是什么？
- 4) 怎样使用VBA代码来删除菜单？
- 5) 修改菜单时，必须打开哪个对话框？
- 6) 当使用Add方法来创建命令栏时，哪个属性是用来在容器应用程序关闭时自动地删除命令栏？
- 7) 怎样显示菜单栏？

17.5.2 练习题

使用VBA代码，创建命名为Exercise17的自定义菜单栏，其中包含“文件”、“编辑”和“帮助”菜单，并且使用VBA代码来显示新创建的菜单栏。另外创建一个过程来删除Exercise17菜单，必要时，可以向工作簿中另外添加一个工作表。在Sheet2上，添加两个命令按钮：一个用来执行创建菜单栏的过程，另外一个用来执行删除菜单栏的过程。运行并测试这两个过程。

第18学时 图 表

Excel最受欢迎的一个特性就是它强大的图表功能，你在自己的应用程序中显然也要用到这个特性。在这个学时中，将录制创建图表的步骤，然后使用 Visual Basic 编辑器来修改所生成的代码。

这个学时的重点包括：

- 录制图表的创建过程
- 编辑录制的代码
- 执行修改过的代码
- 创建一个更复杂的图表示例

18.1 创建图表

作为Excel的用户，你对创建图表可能很熟悉。如果知道怎样创建图表，就具备了自动创建图表时所需要的技能了。你也许会问：“为什么这样说？”因为录制了创建图表的绝大多数处理过程，然后根据需要对录制的代码进行稍微修改就可以了。

18.2 使用图表向导和宏录制器

知道了要制成图表的内容以后，要做的第一件事情是打开宏录制器，现在准备创建希望的图表、并作出任何必要的格式修改，然后关闭宏录制器，浏览录制下来的代码。下面我们来创建一个饼图。首先确保关闭了所有的工作簿，然后打开一个新工作簿，输入表 18-1中的数据。

表18-1 用来制图的数据

单 元 格	值
A1	Item
A2	Apples
A3	Bananas
A4	Oranges
A5	Lemons
A6	Pears
B1	Sales
B2	1250
B3	795
B4	1400
B5	1000
B6	1550

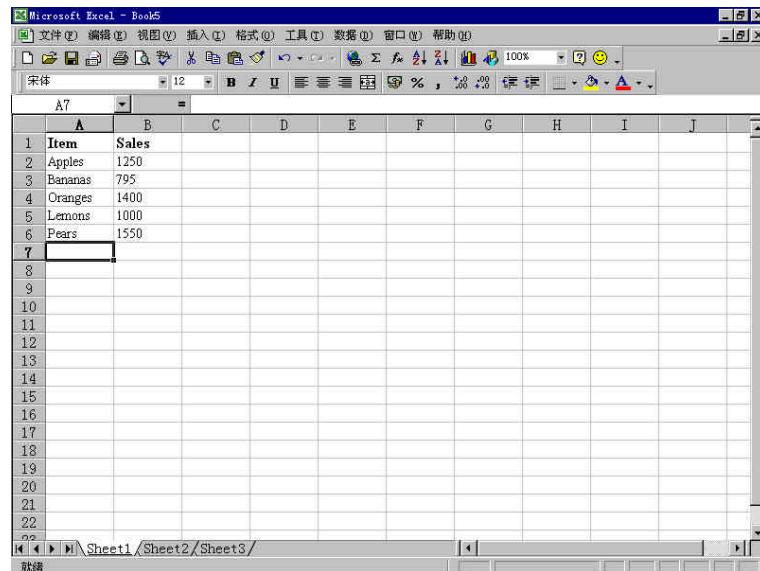
把单元格 A1 到 B1 改成粗体，完成后的工作表如图 18-1 所示。

接下来，选择区域 A1: B6，选择菜单命令“工具”、“宏”、“录制新宏”，把宏命名为 MyPieChart，并把这个宏存储在这个工作簿中，现在准备开始录制宏。

首先，启动“图表向导”，选择“饼图”作为“图表”类型，第2步和第3步都单击“下一步”按钮，然后选择“数据标志”选项卡，选择“显示百分比”，单击“下一步”以显示“图表向导”中的最后一个对话框。

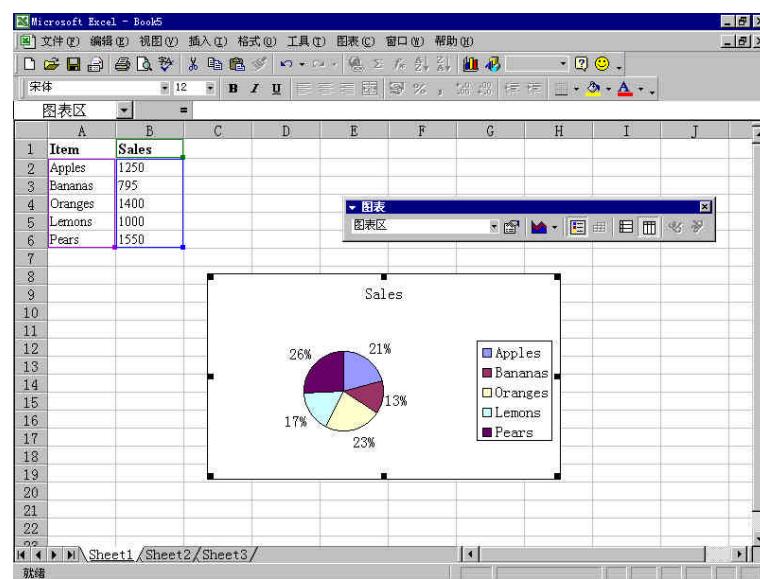
饼图现在就完成了，但是，应该自定义和重新命名这个图，可以通过选择饼图的标题（现在是Sales）来完成。用鼠标右键单击标题，从弹出的菜单中选择“格式化图表标题”。

图18-1 工作表中的数据将
用来创建图表



选择“字体”选项卡，把字体选择为14磅、粗体、斜体。关闭“格式”对话框以后，就停止录制宏。这时应该看到完成的饼图如图18-2所示。

图18-2 这个饼图是使用“图
表向导”来创建的



这个过程很容易吧！现在需要学习怎样编辑刚才录制下来的代码，以便最大限度地利用宏的功能。

18.3 编辑录制的代码

现在，可以按下 Alt+F11 组合键切换到 Visual Basic 编辑器中，浏览刚才录制下来的代码。可能需要在“工程资源管理器”中展开当前的工作簿，然后打开“模块”文件夹来找到这个录制下来的过程，程序清单 18-1 给出了这个录制宏的完整代码：

程序清单 18-1 MyPieChart 过程的完整代码

```
1: Sub MyPieChart ()  
2:   Charts.Add  
3:   ActiveChart.ChartType = xlPie  
4:   ActiveChart.SetSourceData_  
5:     Source:=Sheets("Sheet1").Range("A1:B6"), PlotBy:= xlColumns  
6:   ActiveChart.Location Where:=xlLocationAsObject,Name:= "Sheet1"  
7:   ActiveChart.ApplyDataLabels_  
     Type:=xlDataLabelsShowPercent,LegendKey:=False _  
8:     ,HasLeaderLines:=True  
9:   ActiveChart.ChartTitle.Select  
10:  Selection.AutoScaleFont = True  
11:  With Selection.Font  
12:    .Name = "Arial"  
13:    .FontStyle ="Bold Italic"  
14:    .Size = 14  
15:    .Strikethrough = False  
16:    .Superscript = False  
17:    .Subscript = False  
18:    .OutlineFont = False  
19:    .Shadow = False  
20:    .Underline = xlUnderlineStyleNone  
21:    .ColorIndex = xlAutomatic  
22:    .Background = xlAutomatic  
23:  End With  
24: End Sub
```

录制下来的第一个操作转换成了 Charts 集合上的 Add 方法：

Charts. Add

这立即让你知道刚才创建的集合和对象类型。创建了图表以后，录制宏的其余代码设置了各种属性，执行了一些方法。过程中引用的 ActiveChart 是 Application 对象的一个属性，它返回当前活动的图表。 ChartType 属性设置为 xlPie（第 3 行），这个值是 Excel 中代表饼图的一个常量。接下来，在代码中的第 4 行执行了 SetSourceData 方法，把数据源设置为所选中的区域。例子中的 Location 方法是用来把饼图放置到当前工作表上。最后执行了图表的 ApplyDataLabels 方法，并且传递了一些必要的参数值用来显示饼图中的百分数。

这个过程中使用到的另外一个对象是 ChartTitle。首先选中了 ChartTitle，然后改变了它的字体。注意到即使只改变了字体的大小以及把字体设置为粗体和斜体，但是对话框上的所有字体选项都录制在宏过程的第 9 ~ 22 行中。

18.4 执行修改后的图表代码

现在，我们已经录制和修改了代码，要看宏再次执行后的效果，请执行下面的步骤：

1) 选择区域A1: B3。

2) 运行MyPieChart宏过程，你是否注意到即使刚才只选择了三行数据，但是饼图中却包括了所有六行数据。

3) 删除工作表上的所有饼图。

4) 查看MyPieChart过程。不管选择了几行数据，饼图中始终包括六行数据的原因在于下面的语句：

```
ActiveChart. SetSourceData_
    Source := Sheets (" Sheet1"). Range (" A1: B6"), PlotBy := _
    XlColumns
```

5) 在MyPieChart过程的上部输入下面的代码行：

```
Dim rCurrentRange As Range
Set rCurrentRange = Selection
```

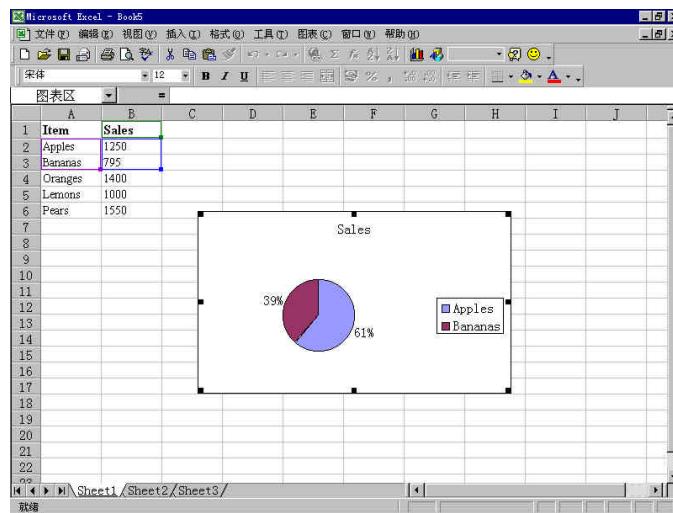
6) 把步骤4中的Sheets (" Sheet1"). Range (" A1: B6")用rCurrentRange来替代，修改后的代码行应该是下面这样的：

```
ActiveChart. SetSourceData Source := rCurrentRange, PlotBy :=_
    XlColumns
```

7) 返回到工作簿。

8) 选择区域A1: B3，运行MyPieChart宏过程，结果饼图就只有两个部分了，如图18-3所示。

图18-3 通过使用 Selection 属性，可以使过程变得更加灵活



18.5 更复杂的图表示例

下面的图表代码示例将分析选中的数据源，从而确定图形类型应该是饼图还是条形图。如果选中的数据只有两列的话，就准备绘制饼图；如果选中的数据超过两列，就绘制条形图。请执行下面的步骤：

1) 在工作表中输入表18-2中的数据。

表18-2 饼图数据源

单 元 格	值
C1	2nd Qtr
C2	1000
C3	1150
C4	875
C5	1270
C6	1395

2) 把单元格C1中的字体设置为粗体，另外把单元格B1中的值修改成1st Qtr。

3) 删除工作表上的所有图形，选择区域A1:C6。

4) 选择“工具”、“宏”、“录制新宏”菜单命令，把新宏命名为MyColumnChart，并把它存储在当前工作簿中。单击“确定”按钮开始录制宏。

5) 单击“图表向导”工具栏按钮，“图表向导”就打开了。

6) 选择“条形图”作为图表类型，单击“完成”按钮来创建条形图。

7) 停止录制宏。

现在有两个录制的宏：MyPieChart和MyColumnChart。接下来，把这两个宏过程合并到另外一个新过程中，这个新过程将根据所选中区域中的列数目来绘制合适的图形类型。按下Alt+F11组合键切换到Visual Basic编辑器中，创建一个命名为MyChart的新过程，在这个新过程中输入下面的代码：

```
Dim rCurrentRange As Range
Set rCurrentRange = Selection

If Selection.Columns.Count = 2 Then
```

为了节省一些时间，可以切换到MyPieChart过程，复制下面的代码行：

```
Charts.Add
ActiveChart.ChartType = xlPie
ActiveChart.SetSourceData Source:=rCurrentRange, PlotBy:= _
    XlColumns
ActiveChart.Location Where:=xlLocationAsObject, Name:="Sheet1"
ActiveChart.ApplyDataLabels Type:=xlDataLabelsShowPercent, LegendKey:=False _,HasLeaderLines:=True
```

别忘了，Visual Basic编辑器中剪切的标准加速键是Ctrl+X组合键，复制的加速键是Ctrl+C，粘贴的加速键是Ctrl+V。

把复制的这些代码行粘贴到MyChart过程中If语句下面的空行中，在MyChart过程中所粘贴的代码下面另起一行，输入下面的代码：

```
ElseIf Selection.Columns.Count > 2 Then
```

切换到MyColumnChart过程，复制下面的代码行：

```
Charts.Add
ActiveChart.ChartType = xlColumnClustered
ActiveChart.SetSourceData Source:=Sheets("Sheet1").Range("A1:C6")
ActiveChart.Location Where:=xlLocationAsObject, Name:="Sheet1"
```

把这些复制的代码行粘贴到 MyChart过程中 ElseIf语句下面的空行中，在刚才粘贴的代码行 ActiveChart. SetSourceData 中，用 rCurrentRange 来取代 Sheets (“ Sheet1 ”). Range (“ A1: C6 ”)，在 MyChart 过程中所粘贴的代码下面另起一行，输入下面的代码：

```

Else
    MsgBox "Selection is not suitable for a chart."
    Exit Sub
End If

With ActiveChart
    .HasTitle = True
    .ChartTitle.Characters.Text = "Sales"
    .ChartTitle.Select
End With

```

现在，从 MyPieChart 过程中复制下面的代码行：

```

Selection.AutoScaleFont = True
With Selection.Font
    .Name = "Arial"
    .FontStyle = "Bold Italic"
    .Size = 14
    .Strikethrough = False
    .Superscript = False
    .Subscript = False
    .OutlineFont = False
    .Shadow = False
    .Underline = xlUnderlineStyleNone
    .ColorIndex = xlAutomatic
    .Background = xlAutomatic
End With

```

把这些复制的代码行粘贴到 MyChart 过程中 End With 语句下面的空行中，MyChart 过程的完整代码如程序清单 18-2 所示。

程序清单 18-2 完整的 MyChart 过程

```

1: Sub MyChart ()
2:     Dim rCurrentRange As Range
3:     Set rCurrentRagne = Selection
4:
5:     If Selection.Columns.Count = 2 Then
6:         Charts.Add
7:         ActiveChart.ChartType = xlPie
8:         ActiveChart.SetSourceData Source:=rCurrentRange, PlotBy:= _
9:             xlColumns
10:        ActiveChart.Location Where:=xlLocationAsObject, Name:="Sheet1"
11:        ActiveChart.ApplyDataLabels _
12:            Type:=xlDataLabelsShowPercent, LegendKey:=False _
13:            ,HasLeaderLines:=True
14:        Elseif Selection.Columns.Count > 2 Then

```

```
14: Charts.Add
15: ActiveChart.ChartType = xlColumnClustered
16: ActiveChart.SetSourceData Source:=rCurrentRange
17: ActiveChart.Location Where:=xlLocationAsObject,Name:="Sheet1"
18: Else
19:   MsgBox"Selection is not suitable for a chart."
20: Exit Sub
21: End If
22:
23: With ActiveChart
24:   .HasTitle = True
25:   .ChartTitle.Characters.Text = "Sales"
26:   .ChartTitle.Select
27: End With
28:
29: Selection.AutoScaleFont = True
30: With Selection.Font
31:   .Name = "Arial"
32:   .FontStyle ="Bold Italic"
33:   .Size = 14
34:   .Strikethrough = False
35:   .Superscript = False
36:   .Subscript = False
37:   .OutlineFont = False
38:   .Shadow = False
39:   .Underline = xlUnderlineStyleNone
40:   .ColorIndex = xlAutomatic
41:   .Background = xlAutomatic
42: End With
43:
44: End Sub
```

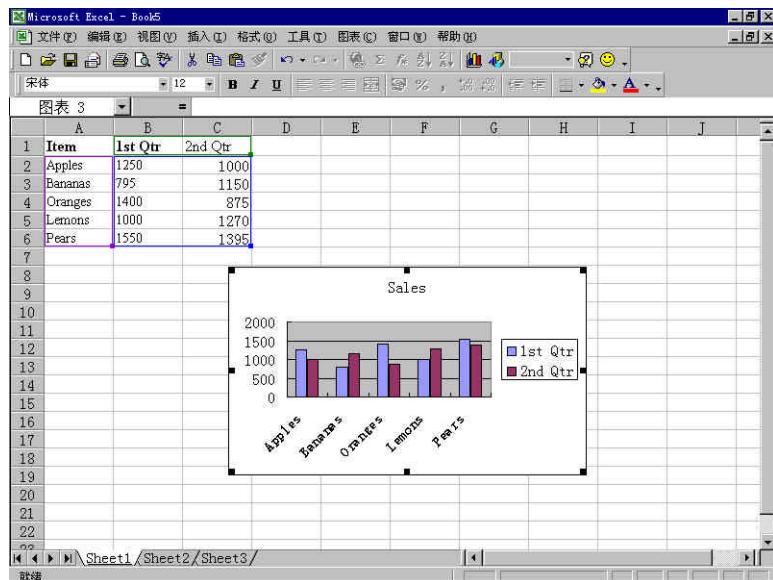
使用复制和粘贴就把两个过程合并到一个新过程中了，这个新过程中唯一的真正的关键是If语句。If语句测试所选中区域的Count属性的取值来确定要绘制图形的数据列数目。如果选中的数据列数目是两列的话，就创建饼图。如果选中的数据列超过两列的话，就创建条形图。如果只选中了一列数据的话，就显示一条消息：

```
If Selection.Columns.Count = 2 Then
  'code for pie chart
ElseIf Selection.Columns.Count > 2 Then
  'code for column chart
Else
  MsgBox"Selection is not suitable for a chart. "
  Exit Sub
End If
```

当图形创建以后，这个过程的其余代码就添加一个标题到图形中，并且设置好标题的格式。要试验这个过程的话，请切换到工作簿中，删除 Sheet1上的所有图形。选择区域 A1: C6，运行MyChart宏过程，绘制的图形如图 18-4所示。

现在删除该图形，选择区域 A1: B5，运行MyChart宏过程，这一次绘制的是饼图。把工作簿另存为Hour 18。

图18-4 因为选中的数据列超过两列，所以绘制的是条形图



18.6 学时小结

这个学时重点介绍了图表的创建，我们学习了 Charts集合以及 Chart对象，还学习了它们的一些属性和方法。

在这个学时中，你自己的编程工作量很小，只是录制了几个宏过程，然后对录制的代码做了一些修改。实际上，这种方式就是许多 VBA开发人员采用的方法。许多开发人员录制尽可能多的宏过程，然后对录制的代码做出必要的修改，这样使得 VBA的开发过程变得快速、简单。

18.7 专家答疑

问题：必须通过录制宏来生成创建图表所需要的代码吗？可以从一开始就自己编程来创建图表吗？

解答：如果愿意，可以从一开始就自己编程来创建图表，这样做更自由、更灵活。这个学时中介绍的技术只是一个很简单的方式，该方式可以使开发人员的工作量最小。

问题：我注意到录制宏有一个缺点：录制宏生成的代码比我需要的要多。例如修改字体设置时，宏录制器包括了对话框中“字体”选项卡上的所有选项。这样难道不会导致低效率的代码吗？

解答：可以说宏录制器并不总是能够提供最有效率的代码，但是要认识到，开发人员总是可以根据需要来编辑和删除任意的代码行。在录制的代码中，可以把认为不需要的代码行注释掉，然后在实际删除这些代码行之前运行这个过程来查看注释掉这些代码行后过程的运行效果。

18.8 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

18.8.1 思考题

- 1) 创建图表时要使用什么 VBA语句 ?
- 2) Application对象的什么属性能够返回当前的活动图表 ?
- 3) Chart对象的什么属性控制了显示的图表类型 ?
- 4) Chart对象的什么属性控制了图形绘制的内容 ?
- 5) 下面哪一个不是Chart对象的属性 : ChartArea, ChartType, ChartLocation, ChartTitle ?
- 6) 判断题 : 创建图表可以通过宏录制器来记录整个处理过程。
- 7) 填空 : 工作簿对象的_____属性返回活动图表。

18.8.2 练习题

创建有5个选项按钮的用户窗体 , 选项按钮的名称和标题分别如下 :

- Pie。
- 3D Pie。
- Clustered Column。
- Stacked Column。
- 3D Stacked Column。

创建一个命为 PickAChartType的过程 , 根据选中的用来绘制图形的区域中数据列数目 , 启用饼图类型中的任意一种或者条形图类型中的任意一种来绘制图形。当这个过程运行时 , 用户窗体应该显示出来 , 用户能够选择需要的图形类型 , 根据选中的区域 , 应该绘制出相应的图形。

提示 : 把用来存储选中区域的变量声明成公共类型。

第19学时 数据透视表

某些工作表拥有数百行的数据和数十个列，处理这种情况的一个方式是固定标题。但是固定标题对于分析大量的数据并没有多少帮助。Excel提供了一个功能强大的工具，能够解决浏览和分析大量数据时需要解决的问题。

这个学时的重点包括：

- 为什么需要使用数据透视表
- 数据透视表概念概述
- 录制宏来创建数据透视表
- 修改录制的数据透视表代码

19.1 使用数据透视表

Excel中最没有得到充分利用的数据分析特性很可能就是数据透视表。数据透视表通过把数据用三维格式表示出来的方式来对大量的数据进行总结。数据显示的格式允许用户把信息筛选到不同层次的细节。图19-1给出了一个数据透视表的示例。

图19-1 图中的数据透视表
用来查看和分析销售信息

求和项: Sales				
Region	Category	Automotive	Large Appliance	Sporting Goods
East	(全部)	167959	145830	123830
North		134412	115764	109885
South		133232	163742	73466
West		148557	115588	83350
总计		584160	540924	390531

数据透视表拥有几个特性，正是这些特性使得应用透视表来分析数据对用户形成了很大的吸引力。

- 内置筛选 数据透视表自动地内置了筛选，从而允许用户只浏览自己关心的细节。
- 动态布局 数据透视表具有动态布局功能，只要通过拖放数据透视表中的字段到数据透视表的其他区域，就能够轻易地改变数据的显示格式。
- 自动汇总报表数据 不管用户决定采用什么字段作为总结，数据都能够自动汇总。用户还能够改变数据汇总的计算类型。
- 支持各种数据源 用户能够基于各种各样的数据源创建数据透视表，这些数据源包括

Excel程序清单、任何带有标签列的工作表区域和外部的数据库文件等等。

19.2 数据透视表概述

当创建数据透视表时，系统会要求用户为下面列出的四个区域选择字段：

- **页字段** 页字段是数据表中主要层次的筛选。当用户选择页字段时，一定要弄清楚到底想怎样组织自己的数据。
- **行字段** 行字段是更低一级的细节筛选，也是第二个层次的细节筛选。
- **列字段** 列字段的筛选层次跟行字段一样。
- **数据项** 数据项就是想要汇总的字段，通常比较适合用来作为数据项的字段，比如销售额、费用、库存数量等等。

现在，我们来看图19-2中给出的工作表，在这张工作表中，能够作为页字段、行字段或者列字段的有三个候选参量：Year，Region和Category。用户只需要决定自己到底最关心的是哪个参量就可以了。数据项的最佳候选参量是Sales字段。

图19-2 区域中的数据即将用
来形成数据透视表

The screenshot shows a Microsoft Excel window titled "Microsoft Excel - Sales". The data is organized into columns labeled "Years", "Region", "Category", and "Sales". The "Sales" column contains numerical values representing sales figures. The rows are numbered from 1 to 20. The "Category" column includes entries like "Automotive", "Large Appliance", and "Sporting Goods". The "Region" column includes "East", "North", and "South". The "Years" column includes "1997" and "1998". The "Sales" column has values such as \$47,634.00, \$34,534.00, \$12,534.00, etc.

图19-3 选定了各级的筛选
字段后所得到的数
据透视表

The screenshot shows a Microsoft Excel window titled "Microsoft Excel - Sales". A PivotTable is being set up with the following structure:

	A	B	C	D	E	F	G
1	Region	(全部)					
3	求和项: Sales	Category					
4	Years	Automotive	Large Appliance	Sporting Goods	总计		
5	1997	171952	115067	59110	346129		
6	1998	210224	149786	166804	526814		
7	1999	201984	276071	164617	642672		
8	总计	584160	540924	390531	1515615		
9							
10							
11							
12							
13							
14							
15							
16							
17							
18							
19							
20							
21							
22							

如果已经决定把Region作为页字段、Year作为行字段、Category作为列字段的话，这样得到的数据透视表如图19-3所示。

如果需要的话，可以拥有多个页/行/列字段，这在处理具有很多行和列的数据时非常有用，因为多个筛选字段能够提供多个筛选选项。

19.3 使用代码来创建数据透视表

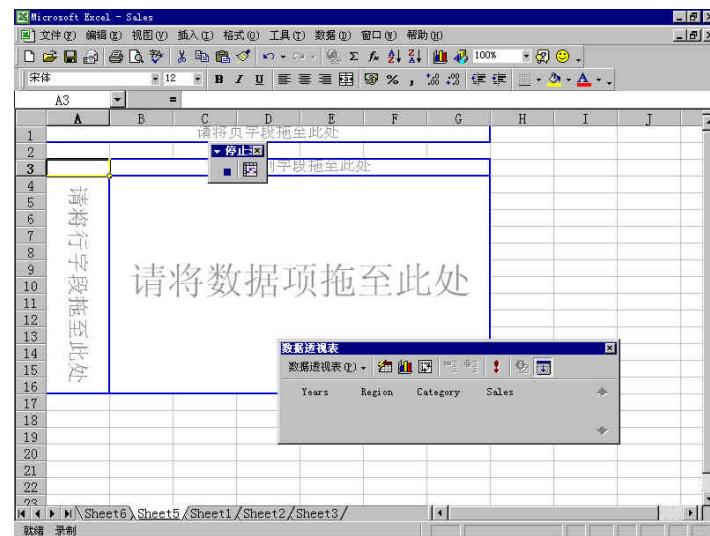
在第18学时中，我们使用了宏录制器来生成创建图表的代码，现在准备学习类似的内容，我们将录制一个宏来生成创建数据透视表的基本代码。

打开本书的Web站点上的Sales.xls文件，你将注意到这个工作簿中的工作表有四列：Year，Region，Category和Sales。

因为想要Excel把这个工作表上的数据当作数据程序清单来处理，所以工作表上的标题行采用了与其他数据行不同的字体，以示区别。

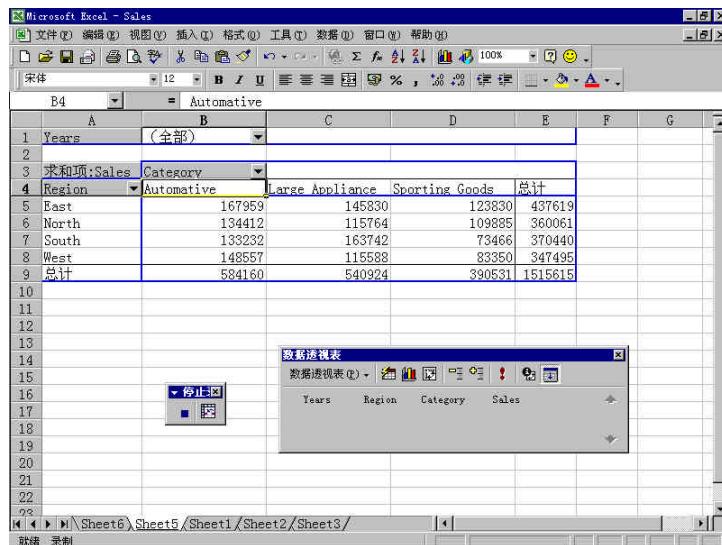
我们准备创建一个基于数据清单的数据透视表，所以应该选择“Microsoft Excel数据清单或数据库”选项按钮。因为正在创建数据透视表，所以“数据透视表”选项按钮也应该选中。单击“下一步”按钮以进入到数据透视表创建中的下一步，确认选中的数据区域是A1:D37后，单击“下一步”按钮。因为准备把新创建的数据透视表放置到新工作表上，所以选择“新建工作表”选项按钮，然后单击“完成”按钮。新建的工作表就添加到了工作簿中，而且一个空数据透视表也显示出来了，如图19-4所示。

图19-4 当向导完成时，就生成了一个空数据透视表，现在准备创建布局了



从“数据透视表”工具栏上把Year字段拖到数据透视表上标记有“请将页字段拖至此处”的地方，接下来，从“数据透视表”工具栏上把Region字段拖到数据透视表上标记有“请将行字段拖至此处”的地方，再从“数据透视表”工具栏上把Category字段拖到数据透视表上标记有“请将列字段拖至此处”的地方，创建数据透视表布局最后要做的一件事情是从“数据透视表”工具栏上把Sales字段拖到数据透视表上标记有“请将数据项拖至此处”的地方。完成后的数据透视表如图19-5所示，现在停止录制宏。

图19-5 在数据透视表中放置 Sales 字段后，
数据就显示出来了



在查看录制的代码之前，先来试验一下数据透视表的功能。Year字段当前显示的是“全部”，在这个字段旁边，有一个带有向下箭头的按钮，单击这个按钮，然后选择1997，单击“确定”按钮，现在数据透视表中就只显示1997年的销售额信息。单击数据透视表上部Category字段旁边的带有向下箭头的按钮，取消选中Automotive和Large Appliance复选框，然后单击“确定”按钮，现在数据透视表中就只显示1997年的Sporting Goods的数字。再次单击Category字段旁边的向下箭头按钮，选中Automotive和Large Appliance复选框，单击“确定”按钮，再从Year字段中选择“全部”，现在所有的信息都再次显示出来了。

19.4 查看录制的代码

按下Alt+F11切换到Visual Basic编辑器中，从“工程资源管理器”中打开“模块”文件夹，然后打开Modules，就可以看到录制的宏代码程序清单。程序清单19-1给出了完整的宏过程：

程序清单19-1 MyPivotTable过程代码

```

1: Sub MyPivotTable()
2:   ActiveWorkbook.PivotCaches.Add(SourceType:=xlDatabase, SourceData:=_
3:     "Sheet1!R1C1:R37C4").CreatePivotTable _
4:     TableDestination:= "", TableName:= _
5:     "PivotTable6"
6:   ActiveSheet.PivotTableWizard TableDestination:=ActiveSheet.Cells(3, 1)
7:   ActiveSheet.Cells(3,1).Select
8:   ActiveSheet.PivotTables("PivotTable6").SmallGrid = False
9:   With ActiveSheet.PivotTables("PivotTable6").PivotFields("Year")
10:    .Orientation = xlPageField
11:    .Position = 1
12:  End With
13:  With ActiveSheet.PivotTables("PivotTable6").PivotFields("Region")
14:    .Orientation = xlRowField
15:    .Position = 1
16:  End With
17:  With ActiveSheet.PivotTables("PivotTable6").PivotFields("Category")

```

```

17:     .Orientation = xlColumnField
18:     .Position = 1
19: End With
20: With ActiveSheet.PivotTables("PivotTable6").PivotFields("Sales")
21:     .Orientation = xlDataField
22:     .Position = 1
23: End With
24: End Sub

```

这个宏过程中的第一条语句执行了 PivotCaches 集合上的 Add 方法，有趣的是这条语句实际上执行了两个方法，同 Add 方法一起执行的还有 CreatePivotTable 方法。注意 TableDestination 参数设置为空字符串，这就告诉 Excel 是在新工作表上创建数据透视表。

```

ActiveWorkbook. PivotCaches. Add (SourceType := xlDatabase, SourceData := _
"Sheet1!R1C1: R37C4"). CreatePivotTable TableDestination := " ", TableName := _
"PivotTable1"

```

新工作表和数据透视表创建以后，代码设置了一系列的属性：

```

ActiveSheet. PivotTableWizard TableDestination := ActiveSheet. Cells(3, 1)
ActiveSheet. Cells(3, 1). Select
ActiveSheet. PivotTables ("PivotTable6"). SmallGrid = False

```

过程中其余的代码跟设置数据透视表的字段有关：

```

With ActiveSheet.PivotTables("PivotTable6").PivotFields("Year")
    .Orientation = xlPageField
    .Position = 1
End With
With ActiveSheet.PivotTables("PivotTable6").PivotFields("Region")
    .Orientation = xlRowField
    .Position = 1
End With
With ActiveSheet.PivotTables("PivotTable6").PivotFields("Category")
    .Orientation = xlColumnField
    .Position = 1
End With
With ActiveSheet.PivotTables("PivotTable6").PivotFields("Sales")
    .Orientation = xlDataField
    .Position = 1
End With

```

如果准备再次运行这个宏过程的话，将得到一个错误，这是因为这个录制的宏在数据透视表创建的时候传递了一个值给 TableName 参数。为了使这个过程更具通用性，请按照程序清单 19-2 对代码进行修改（要修改的代码用粗体标出）。

程序清单 19-2 MyPivotTable 过程的通用版本

```

1: Sub MyPivotTable ()
2:     Dim ptSales As PivotTable
3:
4:     Set ptSales = ActiveWorkbook.PivotCaches.Add(SourceType:=xlDatabase, _
SourceData:= _
5:         "Sheet1!R1C1:R37C4").CreatePivotTable(TableDestination:= " ")
6:     ActiveSheet.PivotTableWizard TableDestination:=ActiveSheet.Cells(3, 1)

```

```
7: ActiveSheet.Cells(3,1).Select
8: ptSales.SmallGrid = False
9: With ptSales.PivotFields("Year")
10:    .Orientation = xlPageField
11:    .Position = 1
12: End With
13: With ptSales.PivotFields("Region")
14:    .Orientation = xlRowField
15:    .Position = 1
16: End With
17: With ptSales.PivotFields("Category")
18:    .Orientation = xlColumnField
19:    .Position = 1
20: End With
21: With ptSales.PivotFields("Sales")
22:    .Orientation = xlDataField
23:    .Position = 1
24: End With
25: End Sub
```

现在，在创建新数据透视表时，就不用担心名称问题了。如果现在已经切换到了工作簿中，那么就可以以任意次数运行这个过程了。

19.5 学时小结

现在，你已经知道怎样使用数据透视表来更简单地处理大量数据了。为了进一步学习VBA，你录制了一个用来创建数据透视表的宏，接着对录制的宏代码进行了修改，以便能够多次使用这个宏过程。

19.6 专家答疑

问题：当改变用来创建数据透视表的数据时，数据透视表能够自动刷新吗？

解答：不能。用户可以通过单击“数据透视表”工具栏上的“刷新”按钮，也可以通过执行RefreshTable方法来刷新数据透视表。

问题：如果每次创建某个数据透视表时都想使用同一个名称的话，需要怎样做呢？

解答：处理这种情况的最简单方式是在用户退出工作簿时删除数据透视表所驻留的工作表，然后在工作簿的Auto_Open过程中放置代码来创建数据透视表，以便当用户打开工作簿时就能够生成数据透视表。

问题：可以使用PivotTables集合上的Add方法来创建数据透视表对象，以取代结合使用PivotCaches集合上的CreatePivotTable方法和Add方法来创建数据透视表吗？

解答：可以。首先执行PivotCaches集合上的Add方法，然后把得到的对象作为PivotTables集合上Add方法中PivotCache参数的值传递过去就可以了。

19.7 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

19.7.1 思考题

- 1) 判断题：只能够创建基于 Excel 数据清单或区域的数据透视表。
 - 2) 列举出创建数据透视表时使用到的两个集合的名称。
 - 3) PivotField 对象的哪个属性分配了字段的位置（就是页、行、列字段和数据项的位置）？
 - 4) 判断题：只能够拥有一个行字段。
 - 5) 列举出数据透视表四个区域的名称。
 - 6) 哪个方法创建了实际的数据透视报表？
 - 7) 判断题：采用宏录制器可以录制创建数据透视表的处理过程。

19.7.2 练习题

打开本书Web站点上的InStock工作簿，开始录制一个命名为 InStockPivot的宏。以图19-6为参照，创建一个数据透视表，选择 Housewares作为Dept.，选择Atlanta作为Location，停止录制宏。修改录制的代码，使代码能够把数据透视表命名为 Instock，并且能够删除工作簿中除Stock Info工作表之外的所有其他工作表。

图19-6 InStockPivot数据
透视表示例

Microsoft Excel - Book7

文件(F) 编辑(E) 视图(V) 插入(I) 格式(O) 工具(T) 数据(D) 窗口(W) 帮助(H)

宋体 12 B Z U EΣ A D F G H I %

D24

	Description	Actual	Target	Actual Total	Target Total	Delta
1	Dept					
2	Customer					
3						
4						
5	Return of the Month	Description	Drive Home			
6		Actual	Target	Actual Total	Target Total	Delta
7						
8						
9	12" Round					
10	Blue Double Crème w/But					
11	Average Wm Margar					
12	Blue Imported Rose					
13	Blue Lemonade					
14	Colossal Doughnut 12 pk					
15	French Tast Compote					
16	Française Truffles					
17	Pista Kiwi-Mango Wreath					
18	Pizzazz Dough					
19	The Lushie Scone					
20	Vegan Cappuccino Red					
21	Watermelon Kefir					
22	Grand Total	18	20	24	20	-16
23						
24						

Sheet5 / Sheet1 /

退出

第20学时 数据访问介绍

在这个学时中，我们将准备使用叫做 MS Query的加载宏，使用MS Query能够创建从外部数据源导入数据的查询。MS Query这个工具的最大优势在于可以使用 Excel的宏录制器来录制整个处理过程。

这个学时的重点包括：

- 对能够检索数据的数据库格式的一些讨论
- 能够检索外部数据的一些方法概述
- 怎样使用MS Query来检索数据
- 修改录制的MS Query宏

20.1 可用的数据库

Excel能够使用几乎所有类型的数据库，能够对数据库中的数据进行访问。下面列出了其中能够使用的一些数据库格式：

- Microsoft Access
- Microsoft SQL Server
- Microsoft FoxPro
- Oracle
- Paradox
- dBASE
- ASCII 文本文件
- SYLK

你也许在考虑，为什么需要把数据库数据导入到 Excel。在处理某些事情时，Excel比数据库能够做得更好，这些事情包括计算、分析和绘制图表等。通过把数据导入到 Excel中以后，就可以利用Excel的这些优势来处理数据。

20.2 数据访问方法

实际上，你已经应用了 Excel中可以使用的一种数据库访问方法，数据透视表可以用来把外部数据导入到Excel。把外部数据导入到Excel的其他方法包括：

- MS Query
- ADO (ActiveX Data Objects , ActiveX数据对象)
- DAO (Data Access Objects , 数据访问对象)
- ODBC (Open Database Connectivity , 开放数据库连接)

快速有效地访问Microsoft Access和其他数据库中数据的最好方法是采用 ADO。支持DAO主要为了向后兼容早期版本的 Access。ODBC可以用来连接各种各样的数据库，包括Microsoft SQL和Oracle等。这个学时将重点介绍使用 Microsoft Query来把外部数据导入到

Excel中的方法。

20.3 使用MS Query来访问数据

从Excel访问外部数据的最简单方法是使用叫做 MS Query的加载宏。MS Query是能够帮助用户连接到数据源并从数据源检索数据的工具，检索到的数据就放置到工作表中。对于Excel开发人员来说，MS Query有一个主要好处：就是可以采用宏录制器来录制整个处理过程，这样就能够大大节省时间，但是同样也得付出代价。你很可能已经猜测到，MS Query在数据访问的性能方面是速度最慢的方法。你将要决定：速度、简单的开发过程以及更好的应用程序性能这三个因素中到底哪个对自己来说最重要。

如果决定使用MS Query作为应用程序解决方案中一个组成部分的话，就需要把MS Query安装到用户的计算机上，在Excel的典型安装时是不会安装MS Query部件的。当选择完全安装时，就会安装MS Query。也可以在Excel安装以后的任何时间再安装MS Query。

我们准备在这个学时中使用的数据库是Northwind，它是同Microsoft Access一起安装的一个示例数据库。要创建从Northwind数据库检索数据的宏，请执行下面的步骤：

- 1) 关闭所有工作簿，打开一个新工作簿。
- 2) 启动录制命名为MSQueryExample的宏，把这个宏存储在当前工作簿中。
- 3) 选择“数据”、“获取外部数据”、“新建数据库查询”，“选择数据源”对话框显示出来，如图20-1所示。用户既可以从已经定义好的数据源中选择，也可以创建新的数据源。

图20-1 “选择数据源”对话框

框列出了系统上已经
定义好的数据源



- 4) 选择Northwind，然后单击“确定”按钮，“查询向导—选择列”对话框显示出来，如图20-2所示。

图20-2 当连接到数据源以
后，下一步工作是
选择查询的列



- 5) 各种查询和表的列表显示出来，要看到可用列的话，请单击“可用的表和列”列表框中选项旁边的“+”标记（加号），找到并选择“产品”。单击产品旁边的“+”标记来查看

列的列表。选择“产品ID”，然后单击“添加”按钮（>）把列添加到“查询结果中的列”列表框中。再添加“产品名称”、“单价”、“库存量”，单击“下一步”按钮，“查询向导—筛选数据”对话框就显示出来了，如图 20-3 所示。

图20-3 筛选能够让用户只检索自己想要的记录



6) 从“待筛选的列”列表框中选择“单价”，从第一个下拉列表框中选择“大于或等于”，在下一个列表框中输入 20，这样就创建了一个筛选，它只显示价格超过 20 美元的那些项。单击“下一步”按钮，“查询向导—排序顺序”对话框显示出来，如图 20-4 所示。

图20-4 选择了筛选以后，就可以选择排序的依据了



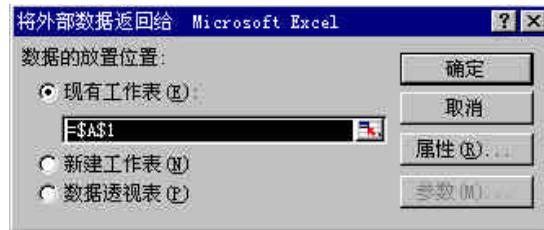
7) 从“主要关键字”下拉列表框中选择“单价”，单击“下一步”按钮，“查询向导—完成”对话框显示出来，如图 20-5 所示。

图20-5 MS Query在处理检索的数据时，为用户提供了几个选项



8) 单击“完成”按钮以返回到Excel，“将外部数据返回给Microsoft Excel”对话框显示出来，如图20-6所示。

图20-6 把外部数据导入Excel
之前的最后一个步骤
是告诉Excel放置数据
的位置



9) 准备把数据放置在当前工作表的A1单元格中。如果A1没有在这个对话框中列出来的话，请选中它，单击“确定”。稍微过一会儿，数据就显示到工作表中，可以供用户使用了。如图20-7所示。

10) 停止录制宏。

现在，我们准备查看录制下来的代码。查看Visual Basic编辑器中MSQueryExample过程，可能让你感到惊奇的是，代码是一个庞大的With语句。程序清单20-1给出了完整的过程代码。

图20-7 检索的数据返回到
了用户的工作簿中

程序清单20-1 MSQueryExample过程

```

1: Sub MSQueryExample ()
2:
3:     With ActiveSheet.QueryTables.Add(Connection:=Array(Array( _
4:         "ODBC; _"
        DBQ=C:\Program Files\Microsoft Office\Office\Samples\
        Northwind.mdb; DefaultDir=C:\Program Files\
        Microsoft Office\Office\Samples; "),
        Array("Driver={Microsoft Access Driver
        (*.mdb)};DriverId=281;FIL=MS Access;
        ImplicitCommitSync=Yes;MaxBufferSize=512;
        MaxScanRows=8;Page"),Array("Timeout=5;
        SafeTransactions=0;Threads=3; UID=admin;

```

```

UserCommitSync=Yes; ")),Destination:=Range
("A1")).CommandText = Array("SELECT
Products.ProductID, Products.ProductName
Products.UnitPrice,Products.UnitsInStock"
& Chr(13)& " "& Chr(10) &"FROM 'C:\Program
Files\Microsoft Office\Office\Samples\
Northwind'.Products Products"& Chr(13)
&" " & Chr(10)& "WHERE(Products.UnitPri"
5: "ce>=15)" & Chr(13) & " " & Chr(10) &
"ORDER BY Products.UnitPrice")
6: .Name = "Query from Northwind"
7: .FieldNames = True
8: .RowNumbers = False
9: .FillAdjacentFormulas = False
10: .PreserveFormatting = True
11: .RefreshOnFileOpen = False
12: .BackgroundQuery = True
13: .RefreshStyle = xlInsertDeleteCells
14: .SavePassword = True
15: .SaveData = True
16: .AdjustColumnWidth = True
17: .RefreshPeriod = 0
18: .PreserveColumnInfo = True
19: .Refresh BackgroundQuery:=False
20: End With
21: End Sub

```

执行的第一个主要语句是 Add方法，这条语句包含了有关连接数据源的信息，其中包括文件位置和驱动程序信息：

```

ActiveSheet.QueryTables.Add(Connection:=Array(Array(_
"ODBC;DBQ=C:\Program Files\Microsoft Office\Office\Samples\
Northwind.mdb;DefaultDir=C:\Program Files\
Microsoft Office\Office\Samples;"),Array(_
"Driver={Microsoft Access Driver (*.mdb)}";
DriverId=281; FIL=MS Access;
ImplicitCommitSync=Yes; MaxBufferSize=512;
MaxScanRows=8;Page"),Array("Timeout=5;
SafeTransactions=0; Threads=3; UID=admin;
UserCommitSync=Yes; ")),Destination:=Range("A1"))

```

当建立连接以后，接着设置了 QueryTable的CommandText属性，这实际上是一个 Select语句，它根据用户在“查询向导”中的选择来筛选合适的数据：

```

.CommandText = Array (
"SELECT Products.ProductID, Products.ProductName,Products.UnitPrice,
Products.UnitsInStock" & Chr(13) & " " & Chr(10) &
"FROM ' C:\Program Files\Microsoft Office\Office\Samples\Northwind'
.Products Products" & Chr(13) & " " & Chr(10)&
"WHERE(Products.UnitPrice>=15)" & Chr(13)& " " &
Chr(10) &"ORDER BY Products.UnitPrice")

```

过程中其余代码设置了各种各样的属性，现在已经可以看到笔者谈到的 MS Query的主要

要好处了：通过录制宏，开发工作就是“小菜一碟”！

你很可能会问怎样修改这种类型的宏，修改这种类型宏的一个完美示例就是设置宏来提示用户正在查找的是什么价格。要实现这一点，请按照程序清单 20-2 中给出的代码来修改这个过程，修改的代码用黑粗体标出。

程序清单 20-2 修改后的 MSQueryExample 过程

```
1: Sub MSQueryExample ()  
2:   Dim sngPrice As Single  
3:   Dim sMessage As String  
4:  
5:   Worksheets.Add  
6:   sMessage = "You wish to see prices greater than: "  
7:   sngPrice = Application.InputBox(sMessage, "Enter Price", Type:=1)  
8:  
9:   With ActiveSheet.QueryTables.Add(Connection:=Array(Array(_  
10:    "ODBC; DBQ=C:\Program Files\Microsoft Office\Office\Samples\  
     Northwind.mdb; DefaultDir=C:\Program Files\Microsoft  
     Office\Office\Samples; "), Array("Driver=  
     {Microsoft Access Driver(*.mdb)}; DriverId=281;  
     FIL=MS Access; ImplicitCommitSync=Yes;  
     MaxBufferSize=512; MaxScanRows=8; Page"), Array  
     ("Timeout=5; SafeTransactions=0; Threads=3; UID=admin;  
     UserCommitSync=Yes; ")), Destination:=Range("A1"))  
11:   .CommandText = Array ("SELECT Products.ProductID,  
     Products.ProductName, Products.UnitPrice,  
     Products.UnitsInStock" & Chr(13) & " " & Chr(10) &  
     "FROM'C:\Program Files\Microsoft Office\Office\  
     Samples\Northwind'.Products Products" & Chr(13) &  
     " " & Chr(10) & "WHERE (Products.UnitPrice>=" &  
     & sngPrice & ")" & Chr(13) & " " & Chr(10)  
     & "ORDER BY Products.UnitPrice")  
12:   .Name = "Query from Northwind"  
13:   .FieldNames = True  
14:   .RowNumbers = False  
15:   .FillAdjacentFormulas = False  
16:   .PreserveFormatting = True  
17:   .RefreshOnFileOpen = False  
18:   .BackgroundQuery = True  
19:   .RefreshStyle = xlInsertDeleteCells  
20:   .SavePassword = True  
21:   .SaveData = True  
22:   .AdjustColumnWidth = True  
23:   .RefreshPeriod = 0  
24:   .PreserveColumnInfo = True  
25:   .Refresh BackgroundQuery:=False  
26: End With  
27: End Sub
```

在修改后的示例中，代码首先创建了几个变量，第一个变量将存储用户输入的值，第二个变量 sMessage 只是简单地用来保存输入框的文本。

真正的过程是从添加一个空白工作表开始的，这样允许用户确保数据不会意外地覆盖任何内容。接下来，过程提示用户输入筛选条件，然后在 Select语句中使用这个输入值来代替原来数字15所在的位置。

运行修改后的过程，当出现提示时，在价格输入框中输入 20，可以看到工作簿中添加了一个工作表，而且所返回的记录的价格都大于或者等于 20美元。这就是对录制的代码进行的一个简单而又有用的修改。

MS Query并不只限于使用单张表，它也可以基于多张链接表创建查询。

使用MS Query作为开发工具的另外一种方式是使用 MS Query来创建所有需要的查询，然后把所有查询保存到一个查询文件中，这样在过程中就将使用保存的数据库查询。要看使用保存的数据库查询的示例，请选择返回数据中的 A1单元格。如果该单元格还没有显示的话，请显示出“外部数据”工具栏，单击该按钮，单击“查询向导”显示的“下一步”按钮，直到出现“完成”按钮。单击“保存查询”，输入Prices作为该查询的名称并单击“保存”按钮。然后单击“完成”按钮来关闭“查询向导”对话框。

现在，我们准备使用存储查询。添加一张新工作表到工作簿中，录制一个命名为 PricesExample的宏，选择“数据”、“获取外部数据”、“运行保存的查询”，从“运行查询”对话框中选择Prices，并单击“获取数据”，当提示出有关返回数据的位置时，单击“确定”按钮。当数据检索完成以后，停止录制宏。切换到 Visual Basic编辑器中查看录制下来的代码。程序清单20-3给出了录制的宏代码：

程序清单20-3 PricesExample过程

```
1: Sub PricesExample ()  
2:  
3:   With ActiveSheet.QueryTables.Add(Connection:= _  
4:     "FINDER; C:\WINNT\Profiles\Administrator\Application  
      Data\Microsoft\Queries\Prices.dqy" _  
5:     ,Destination:=Range("A1"))  
6:     .Name = "Prices"  
7:     .FieldNames = True  
8:     .RowNumbers = False  
9:     .FillAdjacentFormulas = Falsa  
10:    .PreserveFormatting = True  
11:    .RefreshOnFileOpen = False  
12:    .BackgroundQuery = True  
13:    .RefreshStyle = xlInsertDeleteCells  
14:    .SavePassword = True  
15:    .SaveData = True  
16:    .AdjustColumnWidth = True  
17:    .RefreshPeriod = 0  
18:    .PreserveColumnInfo = True  
19:    .Refresh BackgroundQuery:=False  
20:  End With  
21: End Sub
```

这个过程和MSQueryExample过程之间的主要区别在于，Add方法以及PricesExample过程没有设置CommandText属性。这个方法的缺点是不能够提示用户向查询中插入信息。但是，如果有不需要用户输入信息的标准查询，这仍然不失为一个好方法。

20.4 学时小结

现在，我们已经知道了使用外部数据的两种方式，正如第 19 学时中讨论的那样，使用数据透视表能够检索外部数据。在这个学时中，学习了怎样使用 MS Query 来检索数据的方法。这两种方式都可以使用宏录制器来录制整个处理过程，这样对于有时间压力的 Excel 开发人员来说是一个福音。

在下一个学时中，将学习到使用 ADO 来访问数据，这项技术是本书中所讨论的方法中能够提供最多控制和最好性能的方法。

20.5 专家答疑

问题：可以通过 MS Query 来使用用 Access 创建的查询吗？

解答：不一定。我们可以使用 MS Query，在 Access 现存查询的基础上创建新的查询。但是，不能够从“数据”、“获取外部数据”、“运行保存的查询”菜单命令来运行 Access 查询。

问题：如果想要修改使用 MS Query 创建的查询时，该怎么办？

解答：在“查询向导”的“完成”对话框上有一个选项“查看数据或者编辑 Microsoft Query 中的查询”，这个选项将把用户带到“查询”的界面，这个界面跟 Access 中使用查询的界面很相似。

问题：要把 Access 数据读取到 Excel 工作簿中，系统必须安装 Access 吗？

解答：不必要，只需要安装 MS Query 加载宏。

20.6 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

20.6.1 思考题

- 1) 当使用 MS Query 时，查询所加入的是哪个集合呢？
- 2) 用来存储所创建查询的 Select 语句的属性名称是什么？
- 3) 判断题：在录制宏时如果使用 MS Query，必须对生成的代码做出一些修改。
- 4) 判断题：MS Query 只能够适用于 Access 数据库。
- 5) 判断题：MS Query 能够从多个表中检索数据。
- 6) 使用 MS Query 来检索数据的主要好处是什么？
- 7) 说出把外部数据导入到 Excel 中的四种不同方式。

20.6.2 练习题

创建 MSQueryExample 过程的副本，把副本命名为 PriceAndMore，创建一个简单的用户窗体，其上放置四个选项按钮，选项按钮的标题如下：

- Product ID
- Product Name
- Unit Price
- Units In Stock

修改 PriceAndMore 过程，使得过程能够根据用户在用户窗体上作出的筛选条件正确地检索并保存数据。

第21学时 使用ADO访问数据

在这之前，你已经访问了外部数据，但是是依靠录制的代码来实现外部数据访问的。如果用户不可选用MS Query加载宏时，该怎么办呢？如果要获得比使用 MS Query更多的控制和更好的性能时，该怎么办呢？那么就需要使用 ADO。

这个学时的重点包括：

- 讨论什么是 ADO
- ADO 对象概述
- 怎样添加到 ADO 库的引用
- 使用 ADO 来导入数据到工作表中

21.1 ADO是什么

ActiveX Data Object (ActiveX 数据对象，缩写为 ADO) 能够让开发人员开发通过 OLE DB 提供者来访问和操作数据的应用程序。我们不用使用数据透视表和 MS Query，而可以直接使用 ADO 同数据进行交互，这意味着只需要更少的系统开销，就能够对数据进行更多的控制。

新术语 OLE DB 是一套 Component Object Model (组件对象模型，缩写为 COM) 接口，这套接口提供了访问存储在各种信息源中数据的统一接口。

ADO 的目的是获取对数据源的访问、编辑和更新，这意味着可以使用 ADO 把数据导入到工作表中，然后更新数据，再把数据返回到表中。

21.2 访问数据库的对象

正如 ADO 这个名称所隐含的，ActiveX 数据对象为开发人员在访问和操作数据时提供了一个可以使用的对象模型。在使用 ADO 这个模型时，将使用到下面这些关键的对象：

- Connection (连接) 对象 这个对象代表了到达要使用的数据源的连接。有了到达数据源的连接以后，就能够对那个数据源执行命令了。
- Command (命令) 对象 这个对象代表命令，命令就是能够被数据源处理的查询和语句。
- Parameter (参数) 对象 这个对象需要同 Command 对象一起使用，它代表的是命令的参数。
- Recordset (记录集) 对象 这个对象是 ADO 对象模型的真正核心。使用“记录集”对象来通过连接对数据进行操作。这个对象代表了来自表和查询结果的所有记录集合。
- Field (字段) 对象 这个对象代表了“记录集”中的列。
- Error (错误) 对象 这个对象代表了从数据源返回的错误。包含 ADO 对象的任何操作能够生成一个或者多个提供者错误。当每个错误发生时，就有一个或者多个“错误”对象放置到“连接”对象的 Errors 集合中。

21.3 使用ADO

绝大多数Excel开发人员在使用ADO时有一个目的：从数据库获取数据导入到工作簿中。实现这个目的的基本步骤如下：

- 1) 建立到数据源的连接。
- 2) 获取对记录集的访问。
- 3) 从记录集得到记录。
- 4) 关闭到数据源的连接。

21.4 添加到ADO库的引用

在开始编写代码之前，需要创建一个到ADO库的引用，使用下面的步骤来实现这一点：

- 1) 选择“工具”、“引用”，“引用”对话框显示出来。
- 2) 找到Microsoft ActiveX Data Objects 2.0 Library复选框并选中它。
- 3) 单击“确定”按钮，对ADO库的引用就添加到应用程序中了。

21.5 编写从数据库到工作表的数据

因为编写从数据库到工作表的数据是绝大多数Excel开发人员想做的关键事情之一，所以下面将开始这项工作来学习ADO。在新建的工作簿中，创建程序清单21-1中给出的过程。

程序清单21-1 GoGetProducts过程

```
1: Sub GoGetProducts ()  
2:     Dim rsProducts As ADODB.Recordset  
3:     Set rsProducts = New ADODB.Recordset  
4:     rsProducts.Open Source:="Products", _  
5:         activeconnection:="Provider=Microsoft.Jet.OLEDB.4.0;Data Source= _  
6: C:\Program Files\Microsoft Office\Office\Samples\Northwind.mdb", _  
7:         CursorType:=adOpenStatic, _  
8:         LockType:=adLockOptimistic, _  
9:         Options:=adCmdTable  
10:    With Worksheets("Sheet1")  
11:        .Range("A1").CurrentRegion.Clear
```

代码中的长语句分为几行是出于本书每行长度的限制。笔者建议你在自己的过程中输入虚线作为一个长行。

```
12:    Application.Intersect(.Range(.Rows(1),  
13:        .Rows(rsProducts.RecordCount)),  
14:        .Range(.Columns(1),.Columns(rsProducts.Fields.Count))).  
15:        Value = TransposeArray(rsProducts.GetRows  
16:        (rsProducts.RecordCount))  
13:    End With  
14:  
15:    rsProducts.Close  
16: End Sub
```

这个过程从一开始就使用记录集对象，现在，我们来看一看记录集对象的Open方法。在

这个示例中，它的 Source参数设置为表的名称，activeconnection参数的值提供了必要的连接信息。现在连接的是叫做 Northwind的Access数据库。

```
Dim rsProducts As ADODB.Recordset
Set rsProducts = New ADODB.Recordset
rsProducts.Open Source:="Products", _
    activeconnection:="Provider=Microsoft.Jet.OLEDB.4.0; Data Source= _ 
C:\Program Files\Microsoft Office\Office\Samples\Northwind.mdb", _
    CursorType:=adOpenStatic, _
    LockType:=adLockOptimistic, _
    Options:=adCmdTable
```

一旦能够访问记录集以后，就可以把记录集中的数据导入到工作表中，这时可以看到一个稍微有点儿奇怪的现象：ADO记录集中的数据是采用二维表形式来组织存放的，不过却是用每列来代表记录集中的一个记录，每行代表一个字段，这跟通常的表现形式不一样。实际上，仍然可以行列颠倒过来看待数据就跟通常的一样了。

正因为这样，所以在把记录集中的记录复制到工作表之前，需要把记录集的行列互换，让每行代表一个记录，每列代表一个字段。如果你是一个经验丰富的 Excel用户，可能会想到使用Excel的Transpose函数。这样做可能产生的问题在于对于大量的数组，Transpose函数可能导致失败。为了彻底解决行列互换的问题，下面将创建自己的 Transpose函数版本，如程序清单21-2中所示。在GoGetProducts过程中，首先清空了单元格 A1所在的当前区域，然后结合使用了Intersect方法和自己版本的Transpose过程（MyTranspose）。Intersect方法返回了一个代表两个或者更多个区域矩形交集的 Range（区域）对象，使用这个方法是因为想要在记录集所代表的数组和工作表上区域之间创建一个交集：

```
With Worksheets("Sheet1")
    .Range("A1").CurrentRegion.Clear
    Application.Intersect(.Range(.Rows(1), .Rows(rsProducts.RecordCount)),_
        .Range(.Columns(1), .Columns
            (rsProducts.Fields.Count))).Value = _
        MyTranspose(rsProducts.GetRows
            (rsProducts.RecordCount))
End With
```

这个过程完成的最后一件事情是关闭到记录集的连接：

rsProducts.Close

早先，笔者提到过需要创建一个你自己的 Transpose函数版本。请继续学习，输入这个过程的代码。

程序清单21-2 MyTranspose函数

```
1: Function MyTranspose (ByRef ArrayOriginal As Variant)As Variant
2:   Dim x As Integer
3:   Dim y As Integer
4:   Dim i As Integer
5:   Dim j As Integer
6:   Dim ArrayTranspose () As Variant
7:
```

```

8:  x = UBound(ArrayOriginal, 1)
9:  y = UBound(ArrayOriginal, 2)
10:
11: ReDim ArrayTranspose(y,x)
12:
13: For i = 0 To x
14:   For j = 0 To y
15:     ArrayTranspose (j, i) = ArrayOriginal (i, j)
16:   Next
17: Next
18:
19: MyTranspose = ArrayTranspose
20:
21: End Function

```

这个函数所做的工作就是把一个二维数组的行元素和列元素互换。

现在准备运行GoGetProducts过程，切换到工作簿中，运行该过程。在很短的时间内，数据就导入到工作簿中，如图 21-1 所示。

图21-1 使用ADO把数据导入到工作表中

	A	B	C	D	E	F	G	H	I	J	K	L
1	1	Chai	1	10 boxes ×	\$18.00	39	0	10	FALSE			
2	2	Chang	1	1.24 - 12 oz	\$19.00	17	40	25	FALSE			
3	3	Aniseed S	1	2 12 - 550 m	\$10.00	13	70	25	FALSE			
4	4	Chef Antor	2	2 48 - 6 oz	\$22.00	53	0	0	FALSE			
5	5	Chef Antor	2	2 36 boxes	\$21.35	0	0	0	TRUE			
6	6	Grandma's	3	2 12 - 8 oz	\$25.00	120	0	25	FALSE			
7	7	Uncle Bob	3	7 12 - 1 lb	\$30.00	15	0	10	FALSE			
8	8	Northwood	3	2 12 - 12 oz	\$40.00	6	0	0	FALSE			
9	9	Mishi Kobe	4	6 18 - 500 g	\$97.00	29	0	0	TRUE			
10	10	Ikura	4	8 12 - 200 m	\$31.00	31	0	0	FALSE			
11	11	Queso Cat	5	4 1 kg pkg	\$21.00	22	30	30	FALSE			
12	12	Queso Ma	5	4 10 - 500 g	\$38.00	86	0	0	FALSE			
13	13	Konbu	6	8 2 kg box	\$6.00	24	0	5	FALSE			
14	14	Tofu	6	7 40 - 100 g	\$23.25	35	0	0	FALSE			
15	15	Genen Shr	6	2 24 - 250 m	\$15.50	39	0	5	FALSE			
16	16	Pavlova	7	3 32 - 500 g	\$17.45	29	0	10	FALSE			
17	17	Alice Mutt	7	6 20 - 1 kg	\$39.00	0	0	0	TRUE			
18	18	Carnarvon	7	6 16 kg pkg.	\$62.50	42	0	0	FALSE			
19	19	Teatime Cl	8	3 10 boxes	\$9.20	25	0	5	FALSE			
20	20	Sir Rodney	8	3 30 gift box	\$81.00	40	0	0	FALSE			
21	21	Sir Rodney	8	3 24 pkgs	\$10.00	3	40	5	FALSE			
22	22	Gustaf's K	9	5 24 - 500 g	\$21.00	104	0	25	FALSE			
23	23	Tunbrid	9	5 12 - 250 g	\$9.00	61	0	25	FALSE			
24	24	Guarana F	10	1 12 - 355 m	\$4.50	20	0	0	TRUE			
25	25	NuNuCa N	11	3 20 - 450 g	\$14.00	76	0	30	FALSE			
26	26	Groundnut G	11	3 100 - 700 g	\$31.25	16	0	0	FALSE			

21.6 学时小结

在这个学时中，我们快速介绍了 ADO 概念的总体情况，然后，举例说明了怎样使用 ADO 把数据从表导入到你自己的工作表中。

在下一个学时中，将建立 ADO 的知识体系，将创建一个用户窗体，这个用户窗体能够允许用户浏览和更新记录。

21.7 专家答疑

问题：使用 ADO 只能够访问 Access 这一种数据库格式吗？

解答：实际上，可以访问好几种其他格式的数据库，其中包括 SQL Server 和 FoxPro。

问题：记录集总是表吗？

解答：不是，记录集也可以基于一个命令，也就是说，记录集可以是一个查询的结果集。

21.8 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

21.8.1 思考题

- 1) Open方法的哪个参数用来传递用作记录集的对象的名称？
- 2) 判断题：要通过 ADO 去访问数据的话，只要编写合适的代码就可以了。
- 3) 哪个对象存储了有关在使用 ADO 时所发生的问题的信息？
- 4) 判断题：使用 ADO 只能够访问 Microsoft Access 数据库中的数据。
- 5) 哪个对象代表了通过 ADO 来使用的那些记录？
- 6) 怎样终止到 ADO 数据源的连接？
- 7) 请说出使用 ADO 的四个步骤。

21.8.2 练习题

创建一个带有 3 个选项按钮的用户窗体，选项按钮的标题分别是 Suppliers、Products、Orders。编写必要的代码，使得用户能够选择从什么表把数据导入到用户的工作簿中去。

第22学时 ADO应用提高篇

在这个学时中，将进一步学习有关 ADO应用的知识，主要集中学习 Recordset（记录集）对象及其属性和方法。通过使用记录集对象的各种属性和方法，将创建一个用户窗体，这个用户窗体显示使用 ADO从数据库导入的数据。

这个学时的重点包括：

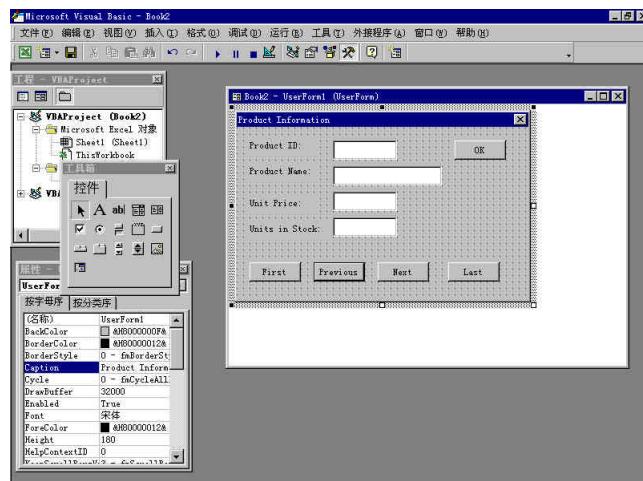
- 怎样从记录集中获取字段显示到窗体上
- 添加记录浏览到窗体上
- 怎样更新记录集的成员

22.1 获取字段以显示到用户窗体

我们不是把数据直接导入到工作表中，而想把数据显示在用户窗体中。这样能够为用户提供一个使用数据的友好界面。你将发现，使用用户窗体的方法跟使用工作表的方法有些区别。现在，我们开始创建用户窗体：

- 1) 添加用户窗体到工作簿中。
- 2) 以图22-1作为参照，添加这些控件到窗体上。

图22-1 这个窗体将用来显示从Access数据库导入的数据



- 3) 设置表22-1中的控件属性。

表22-1 控件属性

控件名称	属性
txtProductID	
txtProductName	
txtUnitPrice	
txtUnitsInStock	
cmdOK	Default = True

4) 把窗体的Caption属性设置为 Product Information，并把窗体命名为 frmProductInfo。

下一步是使用“工具”菜单来添加到 Microsoft ActiveX Data Objects 2.0库的引用。现在准备输入第一个过程的代码。因为想要数据在窗体显示时就显示出来，所以在用户窗体的Activate过程中输入程序清单 22-1中的代码。

程序清单 22-1 用户窗体_Activate过程

```

1: Private Sub UserForm_Activate ()
2:   cnnProduct.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data _"
3:   Source=C:\Program Files\Microsoft Office\Office\Samples\Northwind.mdb"
4:
5:   rstProduct.Open _
"Select ProductID, ProductName,UnitPrice, UnitsInStock from Products", _
6:   cnnProduct,adOpenKeyset, _
7:     adLockOptimistic, adCmdText _
txtProductID.Text = rstProduct.Fields(0).Value
8:   txtProductName.Text = rstProduct.Fields(1).Value
9:   txtUnitPrice.Text = rstProduct.Fields(2).Value
10:  txtUnitsInStock.Text = rstProduct.Fields(3).Value
11:
12: End Sub

```

这个过程开始时打开到数据库的连接，然后使用 Select语句来创建记录集。接下来，使用了字段的取值来设置窗体上那些文本框的 Text属性。请注意，字段索引的取值是从 0开始的。我们还需要把下面的代码行放置到窗体的通用声明区域：

```

Dim cmmProduct As New ADODB. Connection
Dim rstProduct As New ADODB. Recordset

```

在cmdOK_click过程中输入下面的代码行：

```

frmProductInfo. Hide
cnnProduct. Close

```

现在运行窗体，我们将看到记录集的第一个记录的各个取值显示在窗体中。下一步是实现指针在记录之间的移动。

22.2 添加记录浏览

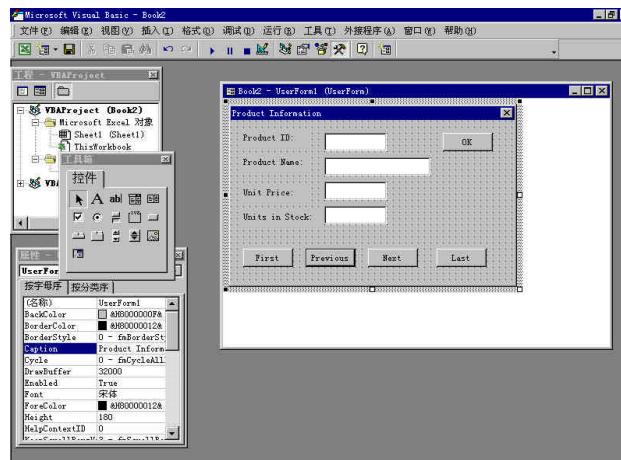
记录集拥有五个移动方法，允许移动浏览记录集中的所有记录：Move、MoveFirst、MoveNext、MovePrevious和MoveLast。Move方法移动一定数量的行，其他方法移动到特定的记录。添加四个命令按钮(如表22-2所示)到用户窗体的底部，如图 22-2 所示。

现在，准备对这四个按钮使用四个移动方法。但是，在添加代码到这些命令按钮之前，需要编写一个过程来把字段的 Value属性值设置成为文本框Text属性的值，因为每个按钮对应的过程都需要使用该过程。需要创建的过程代码如程序清单 22-2所示。

表22-2 命令按钮

Caption	Name
First	cmdFirst
Previous	cmdPrevious
Next	cmdNext
Last	cmdLast

图22-2 增强窗体的功能，使它包括浏览记录的功能



程序清单22-2 FillFields过程

```

1: Sub FillFields ()
2:   txtProductID.Text = rstProduct.Fields(0).Value
3:   txtProductName.Text = rstProduct.Fields(1).Value
4:   txtUnitPrice.Text = rstProduct.Fields(2).Value
5:   txtUnitsInStock.Text = rstProduct.Fields(3).Value
6: End Sub

```

现在，为浏览记录用的命令按钮输入代码，程序清单 22-3 给出了需要的代码。

程序清单22-3 Move过程

```

1: Private Sub cmdFirst_Click ()
2:   rstProduct.MoveFirst
3:   FillFields
4:
5: End Sub
6:
7: Private Sub cmdLast_Click ()
8:   rstProduct.MoveLast
9:   FillFields
10: End Sub
11:
12: Private Sub cmdNext_Click ()
13:
14:   rstProduct.MoveNext
15:   If rstProduct.EOF Then
16:     rstProduct.MoveLast
17:   End If
18:   FillFields
19: End Sub
20:
21: Private Sub cmdPrevious_Click ()
22:   rstProduct.MovePrevious
23:   If rstProduct.BOF Then
24:     rstProduct.MoveFirst
25:   End If

```

26: FillFields

27: End Sub

最后，我们需要为OK按钮输入代码，程序清单22-4给出了OK按钮Click事件过程的完整代码。

程序清单 22-4 cmdOK_Click过程

```
Private Sub cmdOK_Click ()
    Dim iNumRows As Integer

    Worksheets("Product Requests").Activate
    Range("A1").Select
    Selection.CurrentRegion.Select
    iNumRows = Selection.Rows.Count
    Range("A1").Select
    Selection.Offset(iNumRows, 0).Value = txtProductID.Text
    Selection.Offset(iNumRows, 1).Value = txtProductName.Text
    Selection.Offset(iNumRows, 2).Value = txtUnitPrice.Text

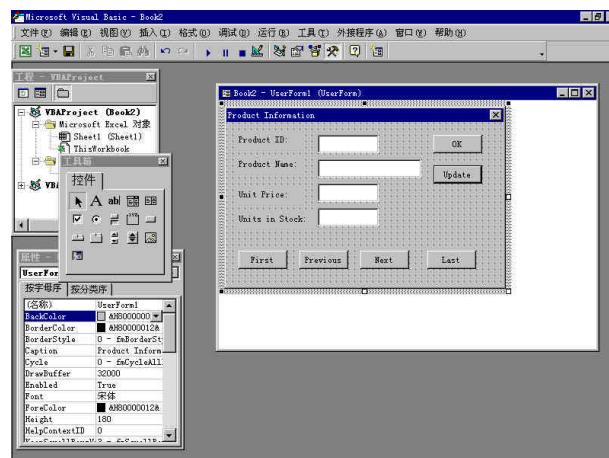
    frmProductInfo.Hide
    cnnProduct.Close
End Sub
```

运行窗体，现在可以使用按钮在记录中前后移动了。查看按钮对应的过程，可以看到，这些过程都很简单，cmdFirst和cmdLast过程又是最简单的，cmdNext和cmdPrevious过程稍微复杂一些。在使用这些Move方法时，并不知道什么时候到达记录集的第一个记录或者最后一个记录，这就需要一些检查工作。记录集对象有BOF和EOF属性，用这两个属性来测试移动是否已经超过记录集的边界。我们可以使用If结构来测试。如果超出了边界，代码能够应用合适的MoveFirst或者MoveLast方法来纠正这种情况。

22.3 更新数据

你也许想让用户能够修改显示在窗体中的数据，并且所做出的修改能够在数据库中得到反映。这可以使用Update方法来实现。在OK按钮下面添加一个命令按钮，命名为cmdUpdate，它的Caption属性设置为Update，如图22-3所示。在这个按钮的Click事件过程中输入程序清单22-5给出的代码。

图22-3 对窗体的下一个增强
是使得窗体能够
更新产品信息



程序清单 22-5 cmdUpdate_Click过程

```

1: Private Sub cmdUpdate_Click()
2:     rstProduct.Fields("ProductName").Value = txtProductName.Text
3:     rstProduct.Fields("UnitPrice").Value = txtUnitPrice.Text
4:     rstProduct.Fields("UnitsInStock").Value = txtUnitsInStock.Text
5: End Sub

```

注意，这个过程没有设置 ProductID 字段的 Value 属性，这是因为 ProductID 字段是主关键字，不应该被修改。要测试过程的话，请运行窗体，把第一个字段的 Unit Price 修改为 19，单击 Update，单击 Next，然后单击 Previous 以返回到这个记录，更新过的价格仍然显示着。

22.4 添加搜索功能

我们准备为窗体添加的最后一个特性是能够搜索某个特定 Product ID 的功能。要实现这一点，将要使用到记录集的 Find 方法，这个方法总是从当前记录开始查找匹配的记录。正因为这样，所以应该通过记录集的 Bookmark 属性把当前记录的位置存储起来，然后执行 MoveFirst 方法把记录指针移动到记录集的第一个记录。如果没有找到匹配的记录，那么，通过把 Bookmark 属性值设置回原来的值，就能够返回到搜索之前所显示的那个记录。添加一个命令按钮到 Update 按钮的下面，把它命名为 cmdFind，把它的 Caption 属性设置为 Find，为这个按钮输入程序清单 22-6 给出的代码。

程序清单 22-6 cmdFind_Click 过程

```

1: Private Sub cmdFind_Click ()
2:     Dim varBookmark
3:     varBookmark = rstProduct.Bookmark
4:     Dim strLookup As String, strFind As String
5:     strLookup = InputBox("Enter product ID", "Locate Product ID")
6:     If strLookup = "" Then Exit Sub
7:     rstProduct.MoveFirst
8:     strFind = "[ProductID] ='" & strLookup & "'"
9:     rstProduct.Find strFind, 0, adSearchForward,rstProduct.Bookmark
10:    If rstProduct.EOF Then
11:        MsgBox "Product not found.", vbInformation, "Find Failure"
12:        rstProduct.Bookmark = varBookmark
13:        Exit Sub
14:    End If
15:    FillFields
16:
17: End Sub

```

运行窗体，单击 Find 按钮，输入 17，当单击 OK 按钮时，将能够看到要查找的字段。

22.5 学时小结

在这个学时中，我们学习使用了记录集对象的一些方法。如果需要从另外的数据源中把数据导入到 Excel，肯定是需要使用到在这里所学到的这些技巧。现在你已经知道怎样浏览、更新、搜索记录集中的记录了，而且還知道怎样把数据库中的数据显示到用户窗体中。如果支持数据库集成的话，那么这些操作类型对于应用程序来说是很重要的。

22.6 专家答疑

问题：为什么不使用Database（数据库）或者Table（表）对象来取代记录集对象呢？

解答：如果在“对象浏览器”中查看ADODB库，你将发现，只要连接着ADO，那么就没有数据库或者表对象可供直接进行数据操作。所有的数据操作都通过记录集对象来完成。

问题：ADO是使用外部数据的最好方式吗？

解答：这取决于具体的需要。如果想花费最小的编程工作量来访问外部数据的话，就可以使用MS Query或者数据透视表了。另外一个替代方法是使用自动化或者ODBC。开发人员最清楚自己的需要，因此能够做出最佳方法选择。

22.7 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

22.7.1 思考题

- 1) 采用什么方法来移动到记录集中的最后一个记录？
- 2) 怎样测试以判断出已经到达记录集的最开头？
- 3) 使用什么方法来修改记录集中的记录？
- 4) 记录集对象的什么属性存储了当前记录的位置？
- 5) 使用什么方法来搜索记录？
- 6) 字段对象的什么属性用来获取字段的内容？
- 7) 判断题：记录集信息可以导入到工作表中，也可以导入到用户窗体中。

22.7.2 练习题

添加一个工作表到工作簿中，并把它命名为Product Requests。在单元格A1中输入Product ID，在单元格B1中输入Product Name，在单元格C1中输入Unit Price。

修改cmdOK过程来把窗体中信息写入到Product Requests工作表中。

第23学时 使用自动化

尽管Excel能够做很多事情，并且能够很好地完成，但是，它并不是万能的。例如，Excel显然不适合写信（尽管听说有人这么用，但是笔者不推荐这么做！）。Microsoft Office已经提供了一整套各种用途的应用软件来满足各种不同的需要。在这个学时中，将学习在Excel中使用VBA来控制其他的应用程序。

这个学时的重点包括：

- 讨论什么是自动化
- 自动化基本情况概述
- 怎样添加到自动化服务器的引用
- 使用“对象浏览器”来查看对象库的信息
- 怎样创建自动化服务器的实例
- 使用自动化来控制Microsoft Word应用程序

23.1 什么是自动化

新术语 自动化也就是早先的OLE自动化，该技术允许用户把Windows应用程序的功能合并到VBA代码中。自动化就是通过一个应用程序来控制另外一个应用程序的处理过程。任何用Word、PowerPoint、Outlook、Access、Project以及其他许多应用程序能够完成的工作，都可以通过使用自动化在Excel VBA应用程序中完成。把这些应用程序作为工具来使用，就能够为用户的应用程序增添新的功能。如果没有自动化这项技术，要想在Excel中获取这些新功能的话，就需要编写大量的代码，或者干脆就是不可能得到。如果想要把Project中工作几个学时的成果到Excel中做成图形的话，该怎么办呢？如果需要对Excel工作表中的数据生成幻灯片时，又该怎么办呢？听起来倒像是PowerPoint的工作。

新术语 要使用自动化的话，需要把另外一个应用程序的功能合并到自己的应用程序中，那么就必须拥有被合并功能的那个应用程序的副本，而且被合并功能的应用程序还必须支持自动化。完全支持自动化的、基于Windows的应用程序显露它们的对象、属性、方法集合，使得这些对象能够被别的应用程序使用。用户使用这些显露的对象，就能够使用应用程序的特性和功能了。如果开发人员以及用户已经拥有了Microsoft Office和其他的OLE兼容的应用程序的话，为什么不使用呢？这些应用程序为用户提供了相似的界面。

23.2 自动化的基本情况

新术语 当使用自动化时，需要懂得每个应用程序的作用。当使用自动化时，需要拥有一个控制器应用程序和一个服务器应用程序。控制器应用程序控制自动化服务器应用程序，Visual Basic就是控制器应用程序的一个非常优秀的例子。服务器应用程序把能够被其他应用程序操

作的对象显露出来。在你的应用程序中，Excel将成为控制器应用程序。

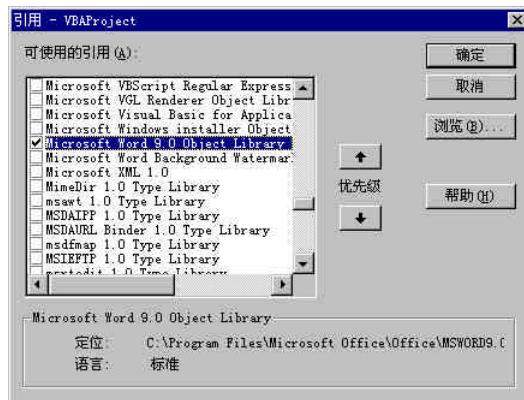
Dynamic Data Exchange(动态数据交换，缩写为 DDE)和SendKeys方法在使用不支持自动化特性的应用程序时可以代替自动化，实现类似的功能。

23.3 引用自动化服务器

新术语 绝大多数支持自动化的应用程序提供了一个对象库。对象库为控制器应用程序提供了服务器应用程序中可以使用的对象的有关信息。要使用对象库信息的话，就需要从控制器应用程序Excel引用对象库。在这个学时中，将通过自动化技术来控制 Microsoft Word，这意味着Word就是服务器应用程序，因而需要引用 Word。在VBA编辑器中，选择“工具”、“引用”菜单命令，“引用”对话框显示出来，如图 23-1 所示。选中 Microsoft Word 9.0 Object Library 复选框后单击“确定”按钮，对 Word 的引用就添加到工程中了。

如果还没有安装 Word 2000，那么“引用”对话框中就不会列出这个对象库，因此就不能够使用 Word 来执行自动化。如果安装了低版本的 Word(Word 8.0)，那么这个学时中介绍的内容仍然适用，只要选中 Microsoft Word 8.0 Object Library 复选框就可以了。如果系统上安装的 Word 最高版本是 Word 95，那么后面介绍的示例就不可使用，因为 Word 95 不支持 VBA。Word 97 (Word 8.0) 是最早使用 VBA 的版本。在这以前的 Word 版本使用 WordBasic 作为其自动化语言。

图23-1 “引用”对话框用来选中自动化使用的对象库

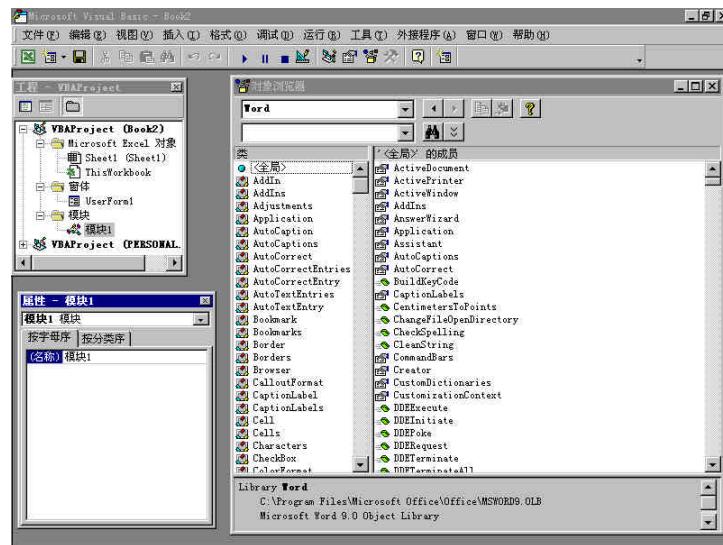


23.4 浏览对象库的内容

“对象浏览器”允许浏览对象库的那些显露出来的对象。“对象浏览器”列出了显露对象的所有属性、事件、方法。要查看 Microsoft Word 的对象库信息，选择“视图”、“对象浏览器”菜单命令来显示“对象浏览器”窗口，如图 23-2 所示，从工程/库列表框（第一个下拉式列表框）中选择 Word 来显示 Word 对象库的内容。可以选择一个对象，查看它的属性和方法。当完

成以上操作时，关闭对象浏览器。

图23-2 对象浏览器是列出特定库中对象的一个很有用的工具



23.5 创建自动化服务器的实例

添加自动化服务器的引用到工程中只是使得自动化能够使用，并不意味就能够实际使用自动化了。要使用对象库，必须创建自动化服务器的实例，这是通过代码来完成的，使用 CreateObject 语句就能够实现。CreateObject 用来创建服务器的新实例，这就意味着在自己的过程中能够使用服务器应用程序的对象、方法和属性了。程序清单 23-1 给出了创建到 Word 的引用时所需要的代码。

程序清单 23-1 创建 Word 的实例

```
1: Dim y As Word.Application
2: Set y = CreateObject("Word.Application")
```

在创建了服务器应用程序的实例以后，就可以使用该实例的所有对象、方法和属性了，就跟使用 Excel 一样。

好了，现在是指出细节的时候了。Word 使用 VBA 作为其自动化语言，Excel 使用 VBA 作为其自动化语言。Word 有宏录制器，Excel 也有宏录制器。这些事实给你什么启示呢？这意味着可以在 Word 中录制宏，然后可以把生成的代码复制到 Excel 过程中！当然需要对代码做细微的修改，稍后再具体说明这一点。

23.6 使用自动化来控制 Microsoft Word

我们准备创建一个简单的应用程序，它把单元格区域中的值插入到一封信中。在开始录制宏之前，需要准备好几件事情，首先需要在 Word 中创建这封信（实际上，可以使用代码从 Excel 中创建信，但是现在不准备这么做），启动 Word 来创建下面的信：

To Sales Manager:

Below you'll find the latest sales figures. If you have any questions please do not hesitate

to contact me.

Sincerely,

The Boss

新术语 把信保存为 Sales，现在，我们准备通过添加书签来对信做一些修改。书签是Word文档中的命名位置。要添加需要的书签，请执行下面的步骤：

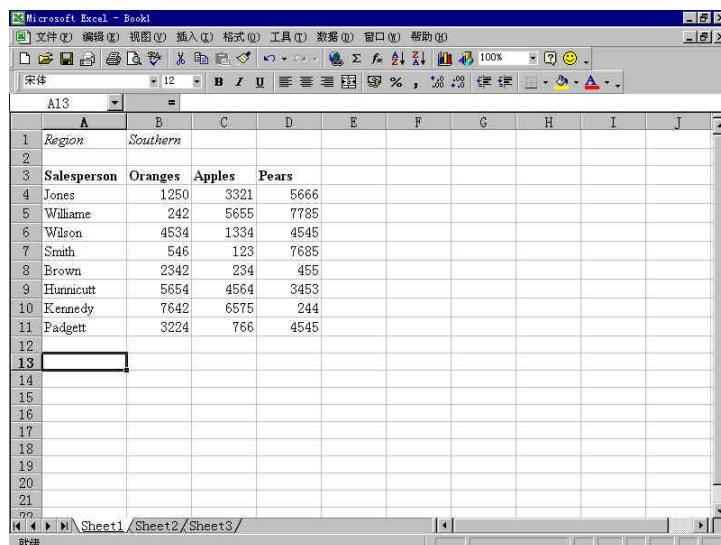
- 1) 在文本 Sales Manager 的开头定位插入点。
- 2) 选择“插入”、“书签”，“书签”对话框显示出来。
- 3) 输入 Region 作为书签的名称，单击“添加”，这就创建了第一个书签。
- 4) 在文本 Sincerely 之前添加一个空行，在空行上定位插入点。在这个位置添加书签，把它命名为 SalesInfo。
- 5) 保存文档并关闭它，最小化 Word。

下一步将在 Excel 中完成。创建图 23-3 所示的工作表，该工作表的信息最终将复制并粘贴到 Sales 文档中。

返回到 Word，开始录制命名为 SalesStuff 的宏（需要打开一个空文档）。

要在 Word 中录制宏的话，使用跟 Excel 中一样的步骤：选择“工具”、“宏”、“录制新宏”菜单命令。

图23-3 图中的销售信息将
复制粘贴到 Sales
文档中



A screenshot of Microsoft Excel showing a sales data table. The table has columns labeled 'Region', 'Salesperson', 'Oranges', 'Apples', and 'Pears'. The data rows show sales figures for various regions and salespeople. Row 13 is highlighted in yellow.

	A	B	C	D	E	F	G	H	I	J
1	Region	Southern								
2										
3	Salesperson	Oranges	Apples	Pears						
4	Jones	1250	3321	5666						
5	William	242	5655	7785						
6	Wilson	4534	1334	4545						
7	Smith	546	123	7685						
8	Brown	2342	234	455						
9	Hunicutt	5654	4564	3453						
10	Kennedy	7642	6575	244						
11	Padgett	3224	766	4545						
12										
13										
14										
15										
16										
17										
18										
19										
20										
21										

要完成录制宏，请执行下面的步骤：

- 1) 选择“文件”、“打开”，打开 Sales 文档。
- 2) 选择“编辑”、“转到”，从“转到”列表框中选择“书签”，选择 Region 作为要跳转到的第一个书签，单击“转到”按钮。
- 3) 选择 SalesInfo 作为下一个要跳转到的书签，单击“转到”按钮。
- 4) 单击“关闭”按钮来关闭对话框。

5) 停止录制宏。

选择“工具”、“宏”、“宏”来访问“宏”对话框，选择 SaleStuff，单击“编辑”来查看录制下来的代码。这个过程中有许多额外的代码，编辑该过程，使得跟程序清单 23-2中给出的一样。

程序清单 23-2 编辑后的SaleStuff过程

```
1: Sub SaleStuff ()  
2:  
3:   Documents.Open FileName:="sales.doc"  
4:   Selection.GoTo What:=wdGoToBookmark, Name:="Region"  
5:   Selection.GoTo What:=wdGoToBookmark, Name:="SalesInfo"  
6:  
7: End Sub
```

这些代码将作为在 Excel中创建过程的基础。剪切这个过程的代码，关闭 Word，返回到 Excel。打开 Visual Basic 编辑器，添加模块到该工作簿中，把剪切的代码粘贴到这个模块中。下一步是编辑这些代码：首先需要创建 Word 的实例，然后需要对粘贴的代码做一些修改。每一行代码需要以变量开头，该变量在创建 Word 的实例时设置。实现这一点的最简单方式是采用 With语句，修改过程使得跟程序清单 23-3中给出的代码一样。

程序清单 23-3 添加实例到Word中

```
1: Sub SaleStuff ()  
2:   Dim y As Word.Application  
3:   Set y = CreateObject("Word.Application")  
4:  
5:   With y  
6:     .Documents.Open Filename:="sales.doc"  
7:     .Selection.GoTo What:=wdGoToBookmark, Name:="Region"  
8:     .Selection.GoTo What:=wdGoToBookmark, Name:="SalesInfo"  
9:   End With  
10:  
11: End Sub
```

我们就快要完成所有工作了，现在，只需要添加代码来把信息从 Excel 复制并粘贴到 Word 中。程序清单 23-4给出了完整的过程，修改的部分用黑粗体标出。

程序清单 23-4 SaleStuff过程的最终版本

```
1: Sub SaleStuff ()  
2:   Dim y As Word.Application  
3:   Set y = CreateObject("Word.Application")  
4:  
5:   With y  
6:     .Visible = True  
7:     'The path listed in the next step may be different  
8:     'on your own machine.Change as needed.  
9:     .Documents.Open FileName:="d:\excel vba\sales.doc"  
10:    Worksheets("Sheet1").Range("B1").Copy  
11:    .Selection.GoTo What:=wdGoToBookmark, Name:="Region"  
12:    .Selection.Paste
```

```
13: Application.CutCopyMode = False
14: Worksheets("Sheet1").Range("A3:D11").Select
15: Selection.Copy
16: .Selection.GoTo What:=wdGoToBookmark, Name:="SalesInfo"
17: .Selection.Paste
18: Application.CutCopyMode = False
19: End With
20:
21: End Sub
```

我们做出的第一个修改是设置 Word的Visible属性为True，这就在前台显示 Word，以便能够看到将要发生的事情：

```
.Visible = True
```

接下来，打开了 Sales文档。文档打开以后，复制了区域 B1，它包含了区域的名称。然后跳转到命名为 Region的书签，把区域 B1的内容粘贴到书签所在的位置。另外还把 Excel的CutCopyMode属性设置为 False，以消除显示在已经剪切或者复制的区域周围的选取框：

```
.Documents.Open Filename:="d:\excel vba\sales.doc"
Worksheets("Sheet1").Range("B1").Copy
.Selection.GoTo What:=wdGoToBookmark, Name:="Region"
.Selection.Paste
Application.CutCopyMode = False
```

对下一个单元格区域重复相同的基本操作：

```
Worksheets("Sheet1").Range("A3:D11").Select
Selection.Copy
.Selection.GoTo What:=wdGoToBookmark, Name:="SalesInfo"
.Selection.Paste
Application.CutCopyMode = False
```

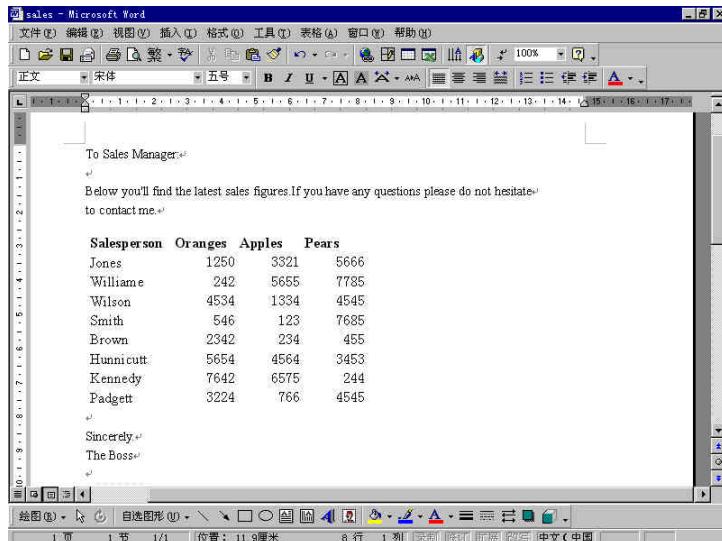
归纳起来，使用自动化时，需要：

- 在服务器应用程序中建立文件。
- 在Excel中建立工作簿。
- 在服务器应用程序中录制尽可能多的宏代码。
- 在Excel中录制尽可能多的宏代码。
- 从服务器应用程序中剪切录制下来的代码，并把它同 Excel中录制的代码合并起来。
- 在Excel过程中添加必要的语句（CreateObject语句）来创建服务器应用程序的实例。
- 对来自服务器应用程序的粘贴代码做出必要的修改。

把工作簿保存为 Hour23，如果Word仍然是打开的话，就关闭它。运行 SaleStuff宏，将看到Word启动，Sales文档打开。在很短的时间内，从 Excel复制的信息就粘贴到文档中了，如图23-4所示。这是一个简单的示例，在实际生活中，很可能需要对信的格式进行更多的调整，但是从示例获得了使用自动化的思想。

要再次最好地运行这个宏过程，关闭 Word，不要保存对 Sales文档所做出的修改。

图23-4 完成后的Sales信



23.7 学时小结

如果最大限度地利用 Excel，就可以获得Excel最擅长完成的工作：数字处理、图表、报告等等。在这些工作之外，还可以看到许多其他应用程序也能够相当出色地完成许多方面的工作，比如创建文档和生成幻灯片等。你现在知道怎样通过使用 Excel VBA应用程序来合并自动化特性，以增强应用程序的功能。通过使用自动化，实际上就添加了 Excel所缺乏的所有功能特性。这样就真正扩展了作为程序员所能够完成的事情。

23.8 专家答疑

问题：所有的Windows应用程序都支持自动化吗？

解答：不是，只有那些设计用来实现自动化的应用程序支持自动化，并不是所有的应用程序制造商都这么做。

问题：基于DOS的应用程序支持自动化吗？

解答：不是，基于DOS的应用程序不能够支持自动化。要控制基于 DOS的应用程序，可以尝试使用SendKey方法，该方法在VBA的在线帮助中有说明文档。

23.9 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

23.9.1 思考题

- 1) 使用什么语句来添加实例到自动化服务器应用程序中？
- 2) 判断题：除了 Excel和VBA的局部对象之外，在“对象浏览器”中只能够查看引用的库。
- 3) 在这个学时创建的应用程序中，Excel是_____应用程序。
- 4) 判断题：Excel只能够充当控制器应用程序。

5) 怎样添加到对象库的引用？

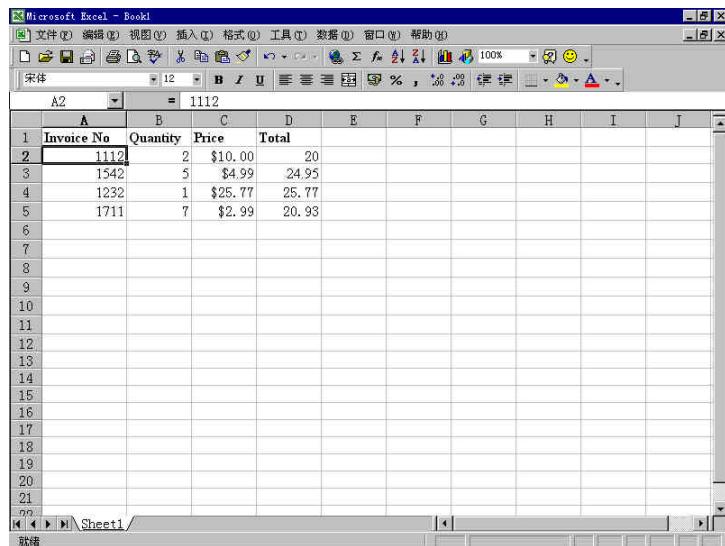
6) 判断题：要在自动化中使用Word，就必须能够从系统访问Word。

7) 判断题：多数Office应用程序支持某种层次的自动化。

23.9.2 练习题

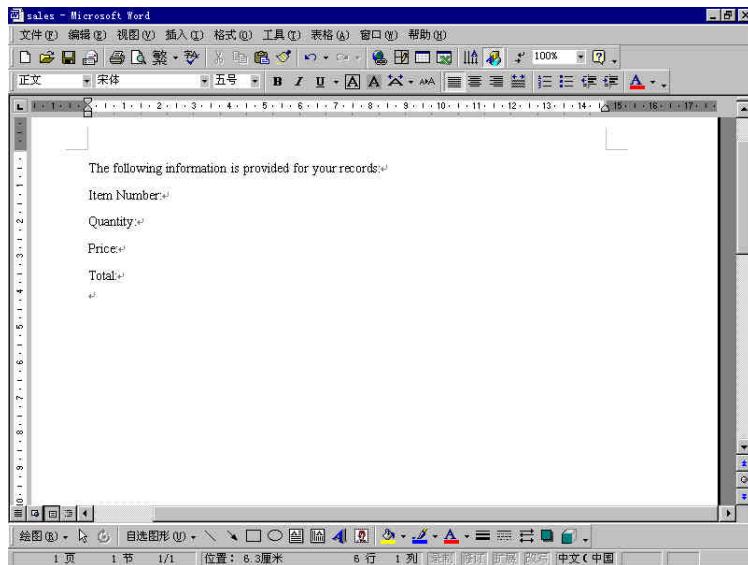
创建一个新工作簿，把它命名为Orders。创建如图23-5所示的工作表。在Word中创建命名为Invoice Letter的文档，如图23-6所示。添加需要的书签。创建一个过程，该过程把选中行中的信息添加到Invoice Letter文档中的适当位置。

图23-5 这个工作表中的数据通过自动化来填写的内容



	A	B	C	D	E	F	G	H	I	J
1	Invoice No.	Quantity	Price	Total						
2	1112	2	\$10.00	20						
3	1542	5	\$4.99	24.95						
4	1232	1	\$25.77	25.77						
5	1711	7	\$2.99	20.93						
6										
7										
8										
9										
10										
11										
12										
13										
14										
15										
16										
17										
18										
19										
20										
21										

图23-6 图中显示的是自动化练习中使用的信



第24学时 运行过程

在任何时候当打开或者关闭某个特定的工作簿时，如果想要运行某个特定的过程，那么该怎么办呢？在用户激活工作簿中的工作表时，想要运行某个过程，又该怎么办呢？Excel提供了实现这些任务和其他任务的方法，这个学时就重点介绍这方面的内容。

这个学时的重点包括：

- 使用Excel的Auto_Open和Auto_Close过程
- 实现事件过程
- 创建加载宏

24.1 自动运行过程

当用户打开或者关闭工作簿时，可能需要某些过程能够自动运行。Excel提供的方法使得这项任务实现起来非常简单。如果想要某个过程在工作簿打开时运行，那么把这个过程命名为Auto_Open就可以了。

Auto_Open对于很多任务来说都非常有用：

- 初始化变量的值。
- 添加菜单和工具栏按钮。
- 初始化工作表上的值。
- 显示应用程序的欢迎界面。
- 改变某些选项，比如选择网格线显示还是直接显示的选项等。
- 插入工作表。
- 执行格式设置任务。
- 从工作簿删除不需要的数据。

如果在关闭工作簿时需要运行某个过程，那么可以把这个过程命名为 Auto_Close，需要在Auto_Close过程中执行的动作的可能类型包括：

- 删除不需要的工作表。
- 把工作簿以其他名称保存，以进行备份。
- 返回到已经关闭的一些显示设置，比如状态栏、网格线、工具栏等等。

每个工作簿只能有一个Auto_Open过程和一个Auto_Close过程。

这两个过程的名称不区分大小写。

如果从VBA打开工作簿，那么工作簿的两个自动过程（Auto_Open和Auto_Close）不会运行。如果想要运行这两个自动过程，请使用RunAutoMacros方法。

要看使用自动过程示例，请创建一个新工作簿，在新模块中创建名为 Auto_Open的过程。在这个自动过程中输入程序清单 24-1 中所给出的代码。

程序清单 24-1 Auto_Open 过程示例

```

1: Sub Auto_Open ()
2:   Range("A1").Value = "Date:"
3:
4:   Range("B1").FormulaR1C1 = "=NOW()"
5:   Columns("B:B").EntireColumn.AutoFit
6:
7:   With ActiveWindow
8:     .DisplayHorizontalScrollBar = False
9:     .DisplayVerticalScrollBar = False
10:  End With
11:  With Application
12:    .DisplayFormulaBar = False
13:    .DisplayStatusBar = False
14:  End With
15: End Sub

```

这个过程将输入日期，关闭滚动条、公式编辑栏和状态栏。当退出工作簿时，应该把这些初始设置恢复过来。创建 Auto_Close 过程，它的代码在程序清单 24-2 中给出。

程序清单 24-2 Auto_Close 过程示例

```

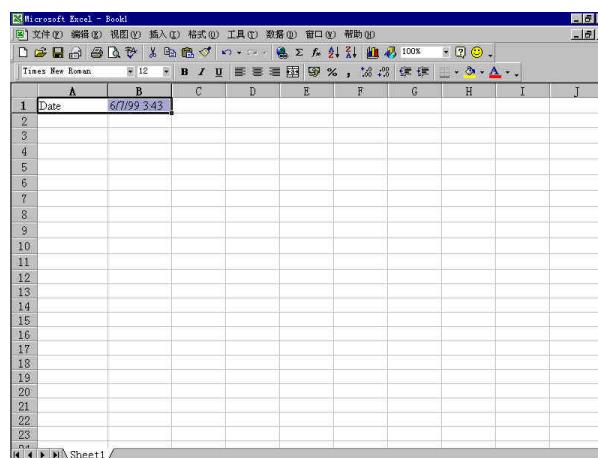
1: Sub Auto_Close ()
2:
3: With ActiveWindow
4:   .DisplayHorizontalScrollBar = True
5:   .DisplayVerticalScrollBar = True
6: End With
7: With Application
8:   .DisplayFormulaBar = True
9:   .DisplayStatusBar = True
10: End With
11: End Sub

```

要测试这两个过程，先保存工作簿为 Hour24。关闭工作簿，然后再次打开它，我们将看到日期已经添加到单元格 B1 中，并且滚动条、公式编辑栏和状态栏这三项设置都没有显示出来，如图 24-1 所示。关闭这个工作簿以后，这些设置又都恢复显示了。

图24-1 自动过程对于自定义

Excel环境非常有用



24.2 运行事件过程

我们可以用编写的过程来响应某些特定的事件，比如：双击、激活工作表、切换到某个窗口、在单元格中输入值等等，这些都是事件。如果想要把宏分配给一个事件 / 动作的话，就需要使用 On 过程。Excel 支持 12 种类型的事件过程（On 过程）：

- OnAction 当单击某个特定对象时触发。
- OnCalculate 在工作表重新计算以后触发。
- OnData 当数据从非 Excel 应用程序中导入时触发。
- OnDoubleClick 当双击特定对象时触发。
- OnEntry 当用户在工作表中输入值时触发。该过程一直要到用户按下回车键或者移动到另外一个单元格时才触发。
- OnKey 当用户按下某个特定组合键时触发。
- OnRepeat 当用户执行 Repeat 命令时触发。
- OnSheetActivate 当从另外一个工作表切换到某个特定的工作表时触发。
- OnSheetDeactivate 当焦点从某个特定的工作表切换到另外一个工作表时触发。
- OnTime 如果 Excel 正在运行并且 OnTime 过程所在的工作簿已经加载，该过程就在某个特定的日期和时间到达时触发。
- OnUndo 当用户撤销一个命令时触发。
- OnWindow 当用户切换到某个特定窗口时，或者当 Excel 应用程序被激活或打开时触发。

许多这样的事件在在线帮助中称作 hidden（隐藏的），表示这些事件不在“对象浏览器”中显示，这并不意味着这样的事件不可用。

典型情况下，这些 On 过程是在工作簿的 Auto_Open 过程中设置的。有各种方式可以使用这些属性。例如，任何时候当 OnData 事件发生时，也许想要执行保存操作。另外当 OnEntry 事件发生时，也许想要进行数据有效性验证。要看一个说明事件怎样工作的简单示例，请执行下面的步骤：

1) 打开一个新工作簿，该工作簿中至少需要有两个工作表。

2) 使用下面的代码来创建一个 Auto_Open 过程：

```
Worksheets("Sheet1").OnSheetActivate = "TryIt"
```

3) 创建命名为 TryIt 的过程，在该过程输入下面的代码：

```
MsgBox "You're back!"
```

4) 把工作簿保存为 Event。

5) 打开工作簿，切换到 Sheet2。

6) 现在切换到 Sheet1，“消息”对话框就显示出来。

7) 单击“确定”按钮以关闭对话框。

为了进一步实验这些属性，请添加下面的代码到 Auto_Open 过程中：

```
Worksheets("Sheet1").OnEntry = "ValidateA1"
```

接下来，创建如程序清单 24-3 中给出的过程。

程序清单 24-3 ValidateA1过程

```
1: Sub ValidateA1 ()  
2:   If Range("A1").Value < 5 Then  
3:     MsgBox "Invalid value"  
4:     Range("A1").Clear  
5:     Range("A1").Activate  
6:   End If  
7:  
8: End Sub
```

现在切换到 Sheet1，选择单元格 A1，键入 4，按下回车键。这样将得到一个消息对话框告诉用户刚才输入了一个无效的值。单击“确定”按钮以关闭该消息对话框。

24.3 创建加载宏

你已经看到好几种让过程自动运行的方法。但是，如果想要在 Excel 启动时就自动加载所开发的应用程序，该怎么办呢？如果确实有这种需要，可以把工作簿保存下来，作为加载宏发布就可以了。加载宏就是不能够读取和修改的工作簿，但是，作为加载宏的工作簿能够包含像自定义工作表布局和过程之类的自定义信息。加载宏能够操作自身、并且能够保存所做出的改变。想要保存在加载宏中的信息类型包括下面的例子：

- 用户自定义的函数。
- 自定义对话框。
- 自定义菜单。
- 自定义工具栏。

在下面的步骤中，要创建一个带有函数的工作簿，这个函数用来计算佣金。之后把工作簿保存为加载宏来使用。完成下面的步骤：

1) 创建一个新工作簿，在该工作簿中创建一个命名为 CalcComm 函数。

2) 在这个函数过程中输入下面的代码：

```
Function CalcComm (x As Variant)  
  CalcComm = x * 0.3  
End Function
```

3) 把工作簿保存为 MyFunctions。

4) 选择“文件”、“另存为”，从“文件类型”中选择“Microsoft Excel 加载宏”，所保存的目录自动变成为 AddIns 文件夹。单击“保存”按钮以创建加载宏。

5) 关闭所有工作簿。

6) 创建新的工作簿。

7) 选择“工具”、“加载宏”，“加载宏”对话框显示出来。

8) 单击“浏览”按钮，选择 MyFunction，单击“确定”按钮。单击“确定”按钮以关闭“加载宏”对话框。

9) 如果查看“窗口”菜单的话，将看到唯一打开的工作簿就是刚创建的那个新工作簿。

10) 在单元格 A1 中输入 100，在单元格 B1 中输入 =calcomm(A1)，然后按下回车键，佣金就计算完了。

24.4 学时小结

在这个学时中，我们学习了用来自动运行过程时所需要的一些技巧。把一个过程命名为 Auto_Open，这个过程就在工作簿打开时运行；把过程命名为 Auto_Close，那么过程就在工作簿关闭时运行。还学习了有关 On过程和怎样使用 On过程来执行所开发宏的方法。最后学习了怎样把自动化解决方案打包到加载宏中，使得它能够在 Excel运行时自动加载。

24.5 专家答疑

问题：为了使得一个过程能够自动运行，除了把该过程命名为 Auto_Open之外，还需要做其他工作吗？

解答：不用，名称本身就是过程自动运行的触发器。

问题：如果想要某个加载宏总是可以使用，那么应该把加载宏存储在什么地方呢？

解答：把它保存到自己的 PERSONAL（个人）工作簿中。

24.6 课外作业

思考题和练习题是为了使你能够进一步地理解所学内容，答案请参考附录。

24.6.1 思考题

- 1) 如果想要过程在使用 MS Query检索到数据时运行某个过程的话，应该使用哪个事件属性（或者说事件过程）呢？
- 2) 判断题：在同一个工作簿中可以拥有多个 Auto_Open过程。
- 3) 至少说出两个能用来分配过程以执行数据有效性验证的事件属性。
- 4) 判断题：每次想要使用自定义的加载宏时，就必须手工加载这些加载宏。
- 5) 在关闭某个工作簿时，需要保存一个外部的 Word文件，这时应该怎么做？
- 6) 至少说出通过加载宏能完成的两件事情。

24.6.2 练习题

创建一个新工作簿，创建命名为 Auto_Open的过程，这个过程需要：

- 在从单元格 A1到C1中依次输入 Date:, Name:, Company Name:
- 使用Now函数以在单元格 B1中输入系统日期。
- 使用 Application（应用程序）对象的UserName属性在单元格 B2中插入自己的名字。
- 使用 Application对象的OrganizationName属性在单元格 B3中插入公司的名称。
- 把A列和B列的尺寸调整到最合适填充的大小。

测试所做的工作。

附录

1. 第1学时答案

1) 只有Excel中才有VBA吗？

不是，其他的Microsoft Office应用程序，包括Word和Access中都可以发现有Excel。

2) VBA基于什么语言？

Visual Basic。

3) 判断题：在VBA应用程序中可以使用Excel的内置函数。

正确。

4) 当编辑宏代码时，是在_____中进行的。

Visual Basic 编辑器。

5) 说出宏录制器的两个局限性。

使用宏的话，在宏运行时不能够给用户提示信息。另外，不能够根据用户的输入和单元格中的值执行不同的动作。

2. 第2学时答案

1) 判断题：加速键只能在最初创建宏时分配。

错误。

2) 能够保存宏的三个地方是什么？

工作簿、新工作簿和个人宏工作簿。

3) 个人宏工作簿保存在什么地方？

\XLSTART。

4) 判断题：个人宏工作簿在启动Excel时自动打开。

正确。

5) 判断题：Excel不允许把宏分配给已经定义了加速键的按键。

错误。

6) 把宏分配给图形的基本步骤是什么？

添加图形到工作表中。

用鼠标右键单击该图形，然后从弹出的菜单中选择“指定宏”。

3. 第3学时答案

思考题

1) 说出两个允许用户从多个可能选项中选择一个可能选项的控件。

选项按钮和列表框。

2) 判断题：用户窗体只有在 Visual Basic 编辑器活动时才能够添加。

正确。

3) 怎样把控件连接到单元格？

用鼠标右键单击控件，选择“设置控件格式”，切换到“控件”选项卡，在“单元格链接”框中输入合适的单元格。

4) 判断题：在 Visual Basic 中可以看见的在用户窗体上显示的网格，当运行窗体时仍然显示出来。

错误。

5) _____是显示静态文本的控件。

标签。

练习题

没有答案。

4. 第4学时答案

1) 作用域有哪三种级别？

过程级别（也叫做局部变量）、模块级别和公共级别。

2) 需要一个变量来存储从 0 ~ 100 的数字时，应该选择什么数据类型？

最好的选择是整型。

3) 过程、变量或者常量名称中最多的字符数是多少？

255。

4) 判断题：过程名称可以以数字开头。

错误。

5) 公共变量在什么地方声明？

在模块的通用声明部分声明。

6) 判断题：常量只能在过程中定义。

错误。常量也可以在模块的 General Declaration 部分定义。

7) 要执行过程时，是按下哪个功能键？

F5。

下面是练习题的完整过程：

```
Public Sub VarAndConst ()  
    Dim sTest As String  
    Const iNumber As Integer = 2  
  
    sTest = "This is a test. "  
  
    MsgBox "sText's value is: " & sTest  
    MsgBox "iNumber's value is: " & iNumber  
End Sub
```

5. 第5学时答案

思考题

1) 用来连接字符串的字符是哪个？

&。

2) 从消息对话框返回的值是什么数据类型？

整型。

3) 从InputBox方法返回的值是什么数据类型？

这取决于类型参数所提供的值。

4) 通过buttons参数给消息对话框设置了哪些内容？

按钮的类型、显示的图标以及指定了默认按钮。

5) 判断题：在VBA中，只能够通过位置传递参数。

错误。也可以使用命名的参数。

练习题

下面是练习题的完整过程：

```
Public Sub YourInfo ()  
    Dim sName As String  
    Dim sCity As String  
    Dim sAge As String  
  
    sName = Application.InputBox("Enter your name: ", Type:=2)  
    sCity = Application.InputBox("Enter your home city: ", Type:=2)  
    sAge = Application.InputBox("Enter your age: ", Type:= 1+2)  
  
    MsgBox sName & "lives in " & sCity & "and is "& sAge & ". "  
  
End Sub
```

6. 第6学时答案

思考题

1) 两种主要的流程控制语句是什么？

If语句和Select Case语句。

2) 判断题：If语句和Select Case语句是对大小写敏感的。

正确。

3) 使用什么方法来显示内置的Excel对话框？

Show方法。

4) 怎样把字符串中的所有字符都转换成大写？

使用UCase函数。

练习题

下面是练习题的完整过程：

```
Sub ClickTest ()  
    Dim iResponse As Integer  
  
    iResponse = MsgBox("Do you wish to continue? ",vbOkCancel)  
  
    If iResponse = vbOK Then  
        MsgBox "OK was clicked."  
    Else  
        MsgBox "Cancel was clicked."  
    End If  
End Sub  
  
Sub Discount ()  
    Dim iDiscountCategory As Integer  
  
    iDiscountCategory = InputBox("Enter discount code: ")  
    Select Case iDiscountCategory  
        Case 1  
            MsgBox "Discount is 5%."  
        Case 2  
            MsgBox "Discount is 10%."  
        Case 3  
            MsgBox "Discount is 15%."  
        Case 4  
            MsgBox "Discount is 20%."  
        Case Else  
            MsgBox "Invalid discount code."  
    End Select  
  
End Sub
```

7. 第7学时答案

思考题

1) VBA中两种主要类型的循环是什么？

For语句和Do语句。

2) 什么语句允许流程跳出循环？

Exit For和Exit Do。

3) Do循环的两种类型是什么？

Do While和Do Until。

4) 判断题：Do循环的条件必须放置在循环的上端。

错误。

练习题

首先，创建命名为 EnterHours 的过程。使用 For 语句来允许用户输入 5 天中职员的学时数，然后计算学时总数。在消息框中显示这个总数。

接下来，创建另外一个命名为 Salary 的过程，给用户提示职员单位学时的报酬。公司中单位学时最少的报酬是 6 美元。使用 Do 循环，继续显示输入框，直到最小数量输入完毕。使用这个比率，根据 EnterHours 过程所计算出的学时数来计算报酬的数量。在消息框中显示报酬总数。

提示：把用来保存学时数目的变量声明成为公共变量。

下面是练习题的完整代码：

```
Public sngNumberOfHours As Single

Public Sub EnterHours()
    Dim iCounter As Integer

    For iCounter = 1 To 5

        sngNumberOfHours = sngNumberOfHours + InputBox
        ("Please number of hours for day " & iCounter)
    Next

    MsgBox "Total of hours: " & sngNumberOfHours

End Sub

Public Sub Salary ()
    Dim sngRate As Single
    Dim sngSalary As Single

    Do While sngRate < 6
        sngRate = InputBox ("Enter the employee's hourly rate: ")
        If sngRate < 6 Then
            MsgBox "Minimum rate is 6.00"
        End If
    Loop

    MsgBox "Salary is: " & sngRate * sngNumberOfHours
End Sub
```

8. 第8学时答案

思考题

1) 怎样设置属性？

Object.propertyname = value (对象. 属性名称 = 值)

2) 怎样调用方法？

Object. Method (对象. 方法)

3) 使用什么语句来把对象分配给对象变量？

Set语句。

4) 判断题：只有对象拥有属性和方法，集合没有。

错误。集合也有属性和方法，例如，集合有Add方法和Count属性。

5) 怎样在集合中创建一个新元素？

使用Add方法。

练习题

对于Application对象来说：

- Excel安装的目录路径是：Path。
- 使用的操作系统是：OperatingSystem。
- Excel这个副本的注册用户名是：UserName。

对于Workbook对象来说：

- 工作簿是否已经保存：Saved。

这个练习题创建的过程应该跟下面的代码类似：

```
Sub TellMeMore ()  
    MsgBox "Excel is installed at " & Application.Path  
    MsgBox "Excel is installed on the " & Application.OperatingSystem  
    MsgBox Application.UserName & "is the registered user."  
    MsgBox "This workbook has " & Worksheets.Count  
    MsgBox "Workbook saved? " & ThisWorkbook.Saved  
End Sub
```

9. 第9学时答案

思考题

1) 怎样使用VBA中的MAX函数来确定区域A1: C5中的最大值？

```
SngResult = Application.Max(Range("A1: C5"))
```

2) 对象层次中最外层的对象是哪个？

Application对象。

3) 创建新工作簿对象或者新工作表对象使用的方法是什么？

Add方法。

4) 使用VBA代码怎样从工作簿中删除某工作表？

使用Worksheet对象的Delete方法。

5) 哪个属性返回当前过程所在的工作簿？这个属性属于哪个对象？

Application对象的ThisWorkbook属性返回当前正在执行的过程所在的工作簿。

6) 判断题：在VBA中不能够运行Excel 4.0的宏。

错误。使用Application对象的Run方法可以运行Excel 4.0的宏。

练习题

下面是练习题完整的过程：

```
Sub Hour9Lab ()  
    Dim wbH9Workbook As Workbook  
    Dim wsH9Worksheet As Worksheet  
  
    Set wbH9Workbook = Workbooks.Add  
    Set wsH9Worksheet = wbH9Workbook.Worksheets.Add  
  
    wsH9Worksheet.Name = "Sharon"  
  
    wbH9Workbook.SaveAs ("Hour9Lab")  
  
End Sub  
  
Sub SaveHour9 ()  
  
    If Workbooks("Hour9Lab").Saved = True Then  
        MsgBox "This workbook has already been saved."  
    Else  
        Workbooks("Hour9Lab").Save  
        MsgBox "The workbook has been saved."  
    End If  
End Sub
```

10. 第10学时答案

思考题

- 1) 判断题：在VBA中，区域总是指多个单元格。
错误。单个的单元格也可以看作区域。
- 2) 根据一个区域的地址来访问另外一个访问时，应该使用 Range对象的哪个属性？
Offset属性。
- 3) 如果想要增加区域中每个单元格的值，并且使得代码最少时，应该使用什么语句？
For Each语句。
- 4) 哪个属性能够根据一个区域的位置去选择另外一个未知的区域？
CurrentRegion属性。
- 5) 怎样确定某个区域中的单元格数目？
使用Count属性。
- 6) 使用什么属性来删除区域的内容？
Clear方法。
- 7) 同一个对象必须设置好几个属性，设置属性时最有效率的方式是什么？
使用With语句。

练习题

下面是练习题的完整过程：

```
Sub ReducePrices ()
```

```
Dim x As Range
Dim bProblems As Boolean

'The For Each statement lets you work with
'the cells in the range.
For Each x In Range("B2:B6")
    x.Value = x.Value - 5
    'Test to see if the prices are now
    'zero or less.
    If x.Value <= 0 Then
        'Because you want to make the price and
        'the item name bold and red you need to
        'start by selecting the item name.
        'Offset(0, -1).Select
        'Resize is used to select both the price
        'and the item name.
        Selection.Resize(1, 2).Select
        With Selection
            .Font.Bold = True
            .Font.Color = vbRed
        End With
        'This variable is used to determine
        'if there was a problem
        bProblems = True
    End If

Next
If bProblems = True Then
    MsgBox "Some of the prices are zero or less!"
End If

End Sub
```

11. 第11学时答案

思考题

1) 说出三种获得在线帮助的方式。

通过“对象浏览器”、在代码窗口按下F1键或者使用“帮助”菜单。

2) “对象浏览器”是用来列出对象、事件、属性和_____的地方。
方法。

3) 跳转到模块的开头时需要按什么键？

Ctrl+Home。

4) _____是一个小的弹出框，这个框在键入时显示有关函数和函数参数的信息。
快速信息。

5) 在什么地方可以控制使用像自动显示快速信息和自动语法检测这样的功能？
“选项”对话框。

6) 判断题：在线帮助提供的示例只可以浏览。这就是说，不能够把这些示例复制并放置到自己的代码中。

错误。从“帮助”系统中可以很容易地复制并粘贴示例代码，这就是帮助的目的所在。

练习题

选择自己编好的任何工具（“帮助”菜单或者“对象浏览器”），查找到下面的信息：

用来执行拼写检查的方法（提示：选中 Application 对象）：

CheckSpelling_____

包含 Excel 安装位置的属性：Path_____

告诉工作簿是否已经保存的属性：Saved_____

用来强制手工计算的方法：Calculate_____

用来隐藏（显示）工作表的属性：Visible_____

用来清空区域内容的方法：Clear_____

12. 第12学时答案

思考题

1) 当运行到断点时将进入什么模式？

中断模式，表示过程的执行暂停了。

2) 怎样确定过程代码执行的顺序？

通过跟踪代码。

3) 怎样在立即窗口中显示属性和变量的值？

使用 Print 或者“？”。

4) 说出除了断点之外用来暂停应用程序的另外一种方法。

使用监视表达式。

5) 判断题：当过程处于中断模式时，查看不到变量的值。

错误。让过程处于中断模式的一个主要原因就是要查看变量的当前状态。

6) 说出跟踪代码运行的两种形式。

逐过程和逐语句。

7) 判断题：观察不会影响程序执行。

错误。观察可以用来让过程进入到中断模式。

练习题

这个过程中的问题是 Select Case 语句是区分大小写的。校正后的过程程序清单如下（校正的部分用黑粗体标出）：

Sub Hour12Exercise ()

Dim sWhichState As String

sWhichState = InputBox("Enter the state for shipping: ")

Select Case **Ucase(sWhichState)**

```
Case "FL"
    MsgBox "Shipping is 3.50. "
Case "NY"
    MsgBox "Shipping is 5.00. "
Case "OH"
    MsgBox "Shipping is 2.00. "
Case "CA"
    MsgBox "Shipping is 6.00. "
Case Else
    MsgBox "We don't ship there. "
End Select

End Sub
```

13. 第13学时答案

思考题

1) 创建错误处理程序的三个主要步骤是什么？

设置错误捕获、编写错误处理实用程序、提供从错误处理实用程序跳出的出口。

2) 用来返回错误号码的对象及其属性的名称是什么？

Err对象的Number属性。实际上只需要引用 Err对象，因为 Number属性是它的默认属性。

3) 哪条语句能够把流程跳转到导致错误发生的代码行？

On Error Resume。

4) 为了把某行变为行标号，该行最后要放置哪个字符？

冒号 (:)。

5) 哪条语句能够把流程跳过导致错误发生的代码行？

On Error Resume Next。

6) 判断题：每个过程都必须有自己的错误处理实用程序。

错误。可以采用集中的错误处理程序。

7) 当创建错误处理程序时，哪种逻辑结构最好？

Select Case语句。

练习题

创建下面的过程：

```
Sub ProcWithError()
    Workbooks. Open "C:\nosuchfile.wkb"
End Sub
```

添加代码以实现错误处理程序，该程序显示一条消息，并从过程中导致错误的下一行继续执行。

下面是练习题的完整过程：

```
Sub ProcWithError ()
    On Error GoTo MyErrorHandler
```

```
Workbooks.Open "C:\nosuchfile.wkb"  
Exit Sub  
MyErrorHandler:  
    MsgBox"This file does not exist."  
    Resume Next  
End Sub(c)Answers for Hour 14
```

14. 第14学时答案

思考题

1) 判断题 : 当运行用户窗体时 , 用户窗体上的网格就显示出来。

错误。只有当窗体处于“设计”模式时网格才显示出来。

2) 哪个属性是用来设置控件加速键的 ?

Accelerator属性。

3) 当按下回车键时 , 怎样识别要执行的是哪个命令按钮 ?

通过设置按钮的Default属性为True。

4) 设置哪个属性来选中选项按钮呢 ?

Value属性。

5) 判断题 : 对齐控件需要设置一些属性。

错误。可以使用“格式”菜单或者位于用户窗体工具栏上的“对齐”按钮来对齐控件。

6) 当把命令按钮的Cancel属性值设置为True时 , 意味着什么呢 ?

如果用户按下了Esc键的话 , 该按钮就执行。

7) 窗体上控件的初始Tab键切换顺序是由什么来决定的 ?

窗体上控件的创建顺序。

练习题

在这个练习中 , 准备创建如图 14-6所示的窗体。把窗体命名为 frmSplash , 并把它的Caption属性设置为 Guest Expenses!。表14-3给出了需要用到的属性设置。

创建窗体以后 , 也许你想要运行它。对于一个简单窗体 , 比如在这个练习中所创建的 , 通过运行窗体以查看窗体集成到完成后的应用程序的效果 , 这总是一个好主意。

15. 第15学时答案

思考题

1) 怎样禁用某个控件 ?

通过把Enabled属性设置为False。

2) 怎样显示窗体 ?

使用窗体的Show方法。

3) 在哪个过程中放置窗体的初始化代码 ?

在用户窗体_Activate过程中。

4) 怎样把窗体从内存中清除掉？

执行Unload方法。

5) 如果要通过代码返回到控件，应该采用什么方法？

使用控件的SetFocus方法。

6) 控件使用的值（例如列表框等使用）应该存储在什么地方？

最简单的地方是工作表上的命名区域。

7) 列表框的哪个属性控制了列表框中显示的项？

ListIndex属性决定了列表框中所列出值的哪一个显示出来。

8) 列举出放置数据有效性验证代码的几个位置。

在Click事件过程的“保存”或者“确定”按钮中。

练习题

创建命名为ShowSplash过程，该过程需要显示frmSplash。

为frmSplash上的cmdOK按钮创建必要的代码以显示frmGuestExpenses窗体。

切换到Guest Expenses工作簿的Sheet1，把ShowSplash分配给窗体上的命令按钮。运行并测试这个应用程序。

下面是练习题的答案：

要显示frmSplash，将使用下面的代码（这就是当单击Sheet1上的命令按钮时将执行的代码。把这些代码放置到模块中）。

```
Sub ShowSplash()
    frmSplash. Show
End Sub
```

要显示frmGuestExpenses的话，将要使用到下面的代码（这就是当单击frmSplash上的“确定”按钮时将执行的代码）。

```
Private Sub cmdOK_Click ()
    frmGuestExpenses. Show
    Unload Me
End Sub
```

16. 第16学时答案

思考题

1) 工具栏属于什么集合？

CommandBars集合。

2) 使用哪个属性来把宏分配给工具栏按钮？

OnAction属性。

3) 请说出工具栏按钮所属的集合。

Controls集合。

4) 使用什么方法可以删除工具栏？

Delete方法。

5) 怎样使用VBA代码来显示工具栏？

通过设置工具栏的Visible属性为True。

6) 哪个属性可以用来控制工具栏按钮上显示的图标？

FaceID设置了按钮上的图标。

7) 判断题：工具栏在创建时就自动显示出来。

错误。必须把工具栏的Visible属性设置为True才能够显示它。

练习题

创建命名为Hour16Toolbar的过程，这个过程需要创建和显示一个命名为 Hour16的新工具栏，其上有三个按钮：“新建（ID是2520）”、“打开（ID是23）”和“保存（ID是3）”。

创建另外一个命名为DeleteTB的过程，这个过程用来删除 Hour16工具栏。运行并测试这两个过程。

下面是练习题的完整过程：

```
Sub Hour16Toolbar ()  
    Dim cbHour16 As CommandBar  
  
    Set cbHour16 = CommandBars.Add(Name:="Hour16")  
  
    With cbHour16  
        .Visible = True  
        .Controls.Add Type:=msoControlButton, ID:=2520, Before:=1  
        .Controls.Add Type:=msoControlButton, ID:=23, Before:=2  
        .Controls.Add Type:=msoControlButton, ID:=3, Before:=3  
    End With  
End Sub  
  
Sub DeleteTB ()  
    CommandBars("Hour16").Delete  
End Sub
```

17. 第17学时答案

思考题

1) 创建菜单时使用的是什么方法？

Add方法。

2) 过程是分配给自定义控件（菜单项）的哪个属性呢？

OnAction属性。

3) 菜单的控件类型是什么？

msoControlPopup。

4) 怎样使用VBA代码来删除菜单？

使用Delete方法。

5) 修改菜单时，必须打开哪个对话框？

“自定义”对话框，该对话框可以通过用鼠标右键单击菜单栏或者用鼠标右键单击工具栏然后选择“自定义”命令来打开。

6) 当使用Add方法来创建命令栏时，哪个属性是用来在容器应用程序关闭时自动地删除命令栏？

设置可选的Temporary参数为True来使新命令栏成为临时的。临时命令栏在容器应用程序关闭时自动删除。该参数的默认值是False。

7) 怎样显示菜单栏？

通过设置菜单的Visible属性为True。

练习题

下面是练习题的完整过程：

```
Sub Exercise17 ()  
  
    Dim Ex17menubar As CommandBar  
    Dim mymenu As Object  
    Dim mymenuItem As Object  
  
    Set Ex17menubar = CommandBars.Add _  
        (Name:="Exercise17", Position:=msoBarTop,MenuBar:=True,  
        Temporary:=True)  
  
    With Ex17menubar  
        .Controls.Add Type:=msoControlPopup, ID:=30002, Before:=1  
        .Controls.Add Type:=msoControlPopup, ID:=30003, Before:=2  
        .Controls.Add Type:=msoControlPopup, ID:=30010, Before:=3  
        .Visible = True  
    End With  
  
    CommandBars("Worksheet Menu Bar").Visible = False  
End Sub  
  
Sub DeleteExercise17 ()  
    CommandBars("Exercise17").Delete
```

18. 第18学时答案

思考题

1) 创建图表时要使用什么VBA语句呢？

Charts. Add。

2) Application对象的什么属性能够返回当前的活动图表？

ActiveChart属性。

3) Chart对象的什么属性控制了显示的图表类型？

ChartType属性。

4) Chart对象的什么方法控制了图形所绘制的内容？

SourceData方法。

5) 下面哪一个不是 Chart对象的属性： ChartArea、 ChartType、 ChartLocation、 ChartTitle？

ChartLocation不是。

6) 判断题：创建图表可以通过宏录制器来记录整个处理过程。

正确。可以很简单地录制图表的创建过程。

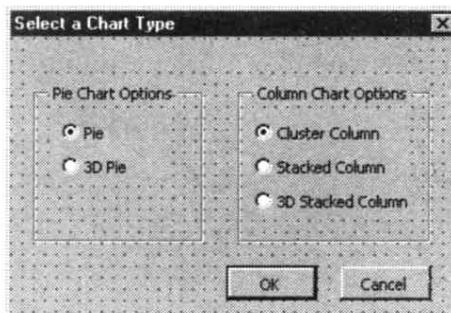
7) 填空：工作簿对象的_____属性返回活动图表。

ActiveChart。

练习题

创建的用户窗体应该跟图 18-5所示的类似。

图18-5 完成后的用户窗体



在模块的通用声明部分，将需要输入下面的代码：

```
Dim rCurrRange As Range
```

下面的过程设置了rCurrRange变量，显示了用户窗体：

```
Sub PickAChartType ()
    Set rCurrRange = Selection
    frmChartType.Show
End Sub
```

下面的代码是同用户窗体关联的：

```
1: Private Sub cmdCancel_Click ()
2:   frmChartType.Hide
3:
4: End Sub
5:
6: Private Sub cmdOK_Click ()
7:
8:   If optPie.Value = True Then
9:     Charts.Add
10:    ActiveChart.ChartType = xlPie
11:    ActiveChart.SetSourceData Source:=rCurrRange, PlotBy:= _
12:      xlColumns
13:    ActiveChart.Location Where:=xlLocationAsObject, Name:="Sheet1"
14:    ActiveChart.ApplyDataLabels _
```

```
Type:=xlDataLabelsShowPercent, LegendKey:=False _
15:     , HasLeaderLines:=True
16: ElseIf opt3DPie.Value = True Then
17:     Charts.Add
18:     ActiveChart.ChartType =xl3DPie
19:     ActiveChart.SetSourceData Source:=rCurrRange,PlotBy:= _
20:         xlColumns
21:     ActiveChart.Location Where:=xlLocationAsObject, Name:= "Sheet1"
22:     ActiveChart.ApplyDataLabels _
23:         Type:=xlDataLabelsShowPercent,LegendKey:=False _
24:             ,HasLeaderLines:=True
25: ElseIf optStackedColumn.Value = True Then
26:     Charts.Add
27:     ActiveChart.ChartType = xlColumnStacked
28:     ActiveChart.SetSourceData Source:=rCurrRange
29:     ActiveChart.Location Where:=xlLocationAsObject,Name:= "Sheet1"
30: ElseIf optClusterColumn.Value = True Then
31:     Charts.Add
32:     ActiveChart.ChartType = xlColumnClustered
33:     ActiveChart.SetSourceData Source:=rCurrRange
34:     ActiveChart.Location Where:=xlLocationAsObject,Name:= "Sheet1"
35: ElseIf opt3DStackedColumn.Value = True Then
36:     Charts.Add
37:     ActiveChart.ChartType = xl3DColumnStacked
38:     ActiveChart.SetSourceData Source:=rCurrRange
39:     ActiveChart.Location Where:=xlLocationAsObject, Name:= "Sheet1"
40: End If
41: With ActiveChart
42:     .HasTitle = True
43:     .ChartTitle.Characters.Text ="Sales"
44:     .ChartTitle.Select
45: End With
46:
47: Selection.AutoScaleFont = True
48: With Selection.Font
49:     .Name ="Arial"
50:     .FontStyle = "Bold Italic"
51:     .Size = 14
52:     .Strikethrough = False
53:     .Superscript = False
54:     .Subscript = False
55:     .OutlineFont = False
56:     .Shadow = False
57:     .Underline = xlUnderlineStyleNone
58:     .ColorIndex = xlAutomatic
59:     .Background = xlAutomatic
60: End With
61:
```

```
62: frmChartType.Hide
63:
64: End Sub
65:
66: Private Sub UserForm_Click ()
67:
68: End Sub
69:
70: Private Sub UserForm_Initialize ()
71: If rCurrRange.Columns.Count = 2 Then
72:     optPie.Enabled = True
73:     optPie.Value = True
74:     opt3DPie.Enabled = True
75:     opt3Dpie.Value = False
76:     optClusterColumn.Enabled = False
77:     optClusterColumn.Value = False
78:     optStackedColumn.Enabled = False
79:     optStackedColumn.Value = False
80:     opt3DStackedColumn.Enabled = False
81:     opt3DStackedColumn.Value = False
82:     lblInfo.Caption = "The recommended type for your data is a pie. "
83: ElseIf rCurrRange.Columns.Count > 2 Then
84:     optClusterColumn.Enabled = True
85:     optClusterColumn.Value = True
86:     optStackedColumn.Enabled = True
87:     optStackedColumn.Value = False
88:     opt3DStackedColumn.Enabled = True
89:     opt3DStackedColumn.Value = False
90:     optPie.Enabled = False
91:     optPie.Value = False
92:     opt3DPie.Enabled = False
93:     opt3Dpie.Value = False
94:     lblInfo.Caption = "The recommended type for your data is a column. "
95: Else
96:     MsgBox "Selection is not suitable for a chart. "
97:     frmChartType.Hide
98: End
99: End If
100:
101: End Sub
```

19. 第19学时答案

思考题

1) 判断题 : 只能够创建基于 Excel 数据表单或区域的数据透视表。

错误。可以根据其他各种各样的数据源来创建数据透视表 , 包括外部数据库和其他的数据透视表。

2) 列举出创建数据透视表时使用到的两个集合的名称。

PivotCaches集合和PivotTables集合。

3) PivotField对象的哪个属性分配了字段的位置（就是页、行、列字段和数据项的位置）？
Orientation属性。

4) 判断题：只能够拥有一个行字段。

错误。可以拥有多个行字段，也可以拥有多个列字段、页字段和数据字段。

5) 列举出数据透视表四个区域的名称。

页字段、行字段、列字段和数据项。

6) 哪个方法创建了实际的数据透视报表？

CreatePivotTable方法。

7) 判断题：采用宏录制器可以录制创建数据透视表的处理过程。

正确。录制数据透视表的创建过程是生成必要的VBA代码的最简单方法。

练习题

下面是练习题的完整宏代码：

```
1: Sub InStockPivot ()  
2:   Dim x As Worksheet  
3:  
4:   'The following line has been added so that the  
5:   'message box with the delete sheets alert  
6:   'won't display.  
7:   Application.DisplayAlerts = False  
8:  
9:   For Each x In Worksheets  
10:    If x. Name <> "Sotck Info" Then  
11:      x.Delete  
12:    End If  
13:  Next x  
14:  
15:  Application.DisplayAlerts = True  
16:  
17:  ActiveWorkbook.PivotCaches.Add(SourceType:=xlDataBase, SourceData:= _  
18:    "Stock Info!R1C1:R45C6").CreatePivotTable  
    TableDestination:=" ", TableName:=_  
19:    "InStock"  
20:  ActiveSheet.PivotTableWizard TableDestination:=ActiveSheet.Cells(3,1)  
21:  ActiveSheet.Cells(3,1).Select  
22:  ActiveSheet.PivotTables("InStock").SmallGrid = False  
23:  With ActiveSheet.PivotTables("InStock").PivotFields("Category")  
24:    .Orientation = xlPageField  
25:    .Position = 1  
26:  End With  
27:  With ActiveSheet.PivotTables("InStock").PivotFields("Dept. ")  
28:    .Orientation = xlPageField  
29:    .Position = 2
```

```
30: End With
31: With ActiveSheet.PivotTables("InStock").PivotFields("Store Num. ")
32:     .Orientation = xlColumnField
33:     .Position = 1
34: End With
35: With ActiveSheet.PivotTables("InStock").PivotFields("Location")
36:     .Orientation = xlColumnField
37:     .Position = 1
38: End With
39: With ActiveSheet.PivotTables("InStock").PivotFields("Description")
40:     .Orientation = xlRowField
41:     .Position = 1
42: End With
43: With ActiveSheet.PivotTables("InStock").PivotFields("In Stock")
44:     .Orientation = xlDataField
45:     .Position = 1
46: End With
47: ActiveSheet.PivotTables("InStock").PivotFields("Dept.").CurrentPage = _
48:     "Housewares"
49: With ActiveSheet.PivotTables("InStock").PivotFields("Location")
50:     .PivotItems("Dallas").Visible = False
51:     .PivotItems("Orlando").Visible = False
52: End With
53: End Sub
```

20. 第20学时答案

思考题

1) 当使用MS Query时，查询所加入的是哪个集合？

QueryTables集合。

2) 用来存储所创建查询的Select语句的属性名称是什么？

CommandText属性。

3) 判断题：在录制宏时如果使用MS Query，必须对生成的代码做出一些修改。

错误。除非想要添加自定义的内容，比如用户提示之类的，否则根本不需要修改录制下来的代码。

4) 判断题：MS Query只能适用于Access数据库。

错误。MS Query能够应用于各种格式的数据库。

5) 判断题：MS Query能够从多个表中检索数据。

正确。MS Query并不只限于使用单个表，可以根据多个链接的表来创建查询。

6) 使用MS Query来检索数据的主要好处是什么？

这个处理过程是可以录制下来的。

7) 说出把外部数据导入到Excel中的四种不同的方式。

MS Query、ADO (ActiveX数据对象)、DAO (数据访问对象) 和 ODBC (开放数据库连接)。

练习题

创建MSQueryExample过程的副本，把副本命名为PriceAndMore，创建一个简单的用户窗体，其上有四个选项按钮。这些选项按钮的标题应该如下：

- Product ID
- Product Name
- Unit Price
- Units In Stock

修改PriceAndMore过程，使得该过程能够根据用户从用户窗体上所作出的选择对数据进行正确地排序。

下面是练习题完整的过程：

```
1: Public sSortChoice As String
2:
3: 'Changes are in bold.
4: Sub PriceAndMore ()
5:   Dim sngPrice As Single
6:   Dim sMessage As String
7:
8:
9:   Worksheets.Add
10:  sMessage = "You wish to see prices greater than: "
11:  sngPrice = Application.InputBox (sMessage, "Enter Price", Type:=1)
12:  frmSort.Show
13:  With ActiveSheet.QueryTables.Add(Connection:=Array ( _
14:    "ODBC; DBQ=C:\Program Files\Microsoft Office\Office\Samples\
      Northwind.mdb;DefaultDir=C:\Program Files\Microsoft
      Office\Office\Samples;"),Array("Driver=
      {Microsoft Access Driver(*.mdb)};DriverId=281;
      FIL=MS Access; ImplicitCommitSync=Yes; MaxBufferSize=512;
      MaxScanRows=8;Page),Array("Timeout=5;SafeTransactions=0;
      Threads=3; UID=admin;UserCommitSync=Yes; "),
      Destination:=Range("A1"))
15:    .CommandText = Array(_
16:      "SELECT Products.ProductID,Products,ProductName,
        Products.UnitPrice,
        Products.UnitsInStock" & Chr(13) & " & Chr(10) &
      "FROM 'C:\Program Files\Microsoft Office\Office\Samples\Northwind'
      .Products Products" & Chr(13) & " & Chr(10) &
      "WHERE(Products.UnitPrice>=" & sngPrice & ")" & Chr(13) &
      " " & Chr(10) & "ORDER BY Products. " & sSortChoice)
17:    .Name ="Query from Northwind"
18:    .FieldNames = True
19:    .RowNumbers = False
20:    .FillAdjacentFormulas = False
21:    .PreserveFormatting = True
22:    .RefreshOnFileOpen = False
23:    .BackgroundQuery = True
24:    .RefreshStyle = xlInsertDeleteCells
```

```
25: .SavePassword = True  
26: .SaveDate = True  
27: .AdjustColumnWidth = True  
28: .RefreshPeriod = 0  
29: .PreserveColumnInfo = True  
30: .Refresh BackgroundQuery:= False  
31: End With  
32: End Sub
```

下面是同用户窗体关联的代码：

```
Private Sub cmdCancel_Click ()  
    frmSort.Hide  
End  
End Sub  
  
Private Sub cmdOK_Click ()  
    If optProductID = True Then  
        sSortChoice = "ProductID"  
    ElseIf optProductName =True Then  
        sSortChoice = "ProductName"  
    ElseIf optUnitPrice = True Then  
        sSortChoice = "UnitPrice"  
    Else  
        sSortChoice = "UnitsInStock"  
    End If  
    frmSort.Hide  
End Sub
```

21. 第21学时答案

思考题

1) Open方法的哪个参数用来传递用作记录集的对象的名称？

Source参数。

2) 判断题：要通过 ADO去访问数据，只要编写合适的代码就可以了。

错误。还必须通过Visual Basic编辑器的“工具”菜单来添加到 ADO的引用。

3) 哪个对象存储了有关在使用 ADO时所发生的问题的信息？

Error对象。

4) 判断题：使用 ADO只能访问 Microsoft Access数据库中的数据。

错误。还可以使用其他各种数据源，包括 Microsoft FoxPro和Microsoft SQL Server。

5) 哪个对象代表了通过 ADO来使用的那些记录？

使用Recordset对象通过连接来操作数据。 Recordset对象代表了来自表或者查询结果的所有记录的集合。

6) 怎样终止到 ADO数据源的连接？

使用Close语句。

7) 请说出使用 ADO的四个步骤。

建立到数据源的连接、获取对记录集的访问、从记录集获取记录和关闭到数据源的连接。

练习题

创建带有三个选项按钮的用户窗体，它们的标题分别为 Suppliers、Products和Orders。编写必要的代码以允许用户选择从哪个表把数据导入到用户的工作簿中。

下面是答案：

用户窗体的代码如下：

```
Private Sub cmdCancel_Click ()  
    frmPickADatabase.Hide  
  
End Sub  
  
Private Sub cmdOK_Click()  
    If optSuppliers.Value = True Then  
        sUserChoice ="Suppliers"  
    Elseif optProducts.Value = True Then  
        sUserChoice = "Products"  
    Else  
        sUserChoice = "Orders"  
    End If  
    frmPickADatabase.Hide  
    GoGetData  
  
End Sub
```

其他过程需要的代码如下，这些过程必须驻留在同一个模块中（如果在这个学时的其他地方也使用到这个模块的话，就不需要再次输入 MyTranspose函数的代码了）。

```
1: Public sUserChoice As String  
2:  
3: Function MyTranspose (ByRef ArrayOriginal As Variant)As Variant  
4:     Dim x As Integer  
5:     Dim y As Integer  
6:     Dim i As Integer  
7:     Dim j As Integer  
8:     Dim ArrayTranspose () As Variant  
9:  
10:    x = UBound(ArrayOriginal, 1)  
11:    y = UBound(ArrayOriginal, 2)  
12:  
13:    ReDim ArrayTranspose(y,x)  
14:  
15:    For i = 0 To x  
16:        For j = 0 To y  
17:            ArrayTranspose(j, i) = ArrayOriginal(i,j)  
18:        Next  
19:    Next  
20:  
21:    MyTranspose = ArrayTranspose  
22:  
23: End Function
```

```
24:  
25:  
26: Sub GoGetData()  
27:   Dim rsData As ADODB.Recordset  
28:   Set rsData = New ADODB.Recordset  
29:   rsData.Open Source:=sUserChoice,_  
30:     activeconnection:="Provider=Microsoft.Jet.OLEDB.4.0;  
31:     Data Source=C:\Program Files\Microsoft Office\Office\  
32:     Samples\Northwind.mdb",_  
33:     CursorType:=adOpenStatic,_  
34:     LockType:=adLockOptimistic,_  
35:     Options:=adCmdTable  
36:   With Worksheets("Sheet1")  
37:     .Range("A1").CurrentRegion.Clear  
38:   Application.Intersect(.Range(.Rows(1),  
39:     .Rows(rsData.RecordCount)),_  
40:     .Range(.Columns(1),  
41:     .Columns(rsData.Fields.Count))).Value =  
42:     MyTranspose(rsData.GetRows(rsData.RecordCount))  
43:  
44:   rsData.Close  
45:  
46: End Sub
```

22. 第22学时答案

思考题

1) 采用什么方法来移动到记录集中的最后一个记录？

MoveLast方法。

2) 怎样测试以判断出已经到达记录集的最开头？

通过测试BOF属性是否为True来判断。

3) 使用什么方法来修改记录集中的记录？

Update方法。

4) 记录集对象的什么属性存储了当前记录的位置？

Bookmark属性。

5) 使用什么方法来搜索记录？

Find方法。

6) 字段对象的什么属性用来获取字段的内容？

Value属性。

7) 判断题：记录集信息可以导入到工作表中，也可以导入到用户窗体中。

正确。可以把数据从记录集导入到工作表或者用户窗体中。

练习题

下面是练习题的答案，修改的部分用黑粗体标出：

```
1: Private Sub cmdOK_Click()
2:   Dim iNumRows As Integer
3:
4:   Worksheets("Product Requests").Activate
5:   Range("A1").Select
6:   Selection.CurrentRegion.Select
7:   iNumRows = Selection.Rows.Count
8:   Range("A1").Select
9:   Selection.Offset(iNumRows, 0).Value = txtProductID.Text
10:  Selection.Offset(iNumRows, 1).Value = txtProductName.Text
11:  Selection.Offset(iNumRows, 2).Value = txtUnitPrice.Text
12:
13:  frmProductInfo.Hide
14:  cnnProduct.Close
15:
16: End Sub
```

23. 第23学时答案

思考题

1) 使用什么语句来添加实例到自动化服务器应用程序中？

CreateObject语句。

2) 判断题：除了Excel和VBA的局部对象之外，在“对象浏览器”中只能够查看到引用了的库。

正确。只有局部的和已经引用了的对象信息可以在“对象浏览器”中查看到。

3) 在这个学时中创建的应用程序中，Excel是_____应用程序。

控制器。

4) 判断题：Excel只能够充当控制器应用程序。

错误。Excel既能够充当控制器应用程序，也能够充当服务器应用程序。

5) 怎样添加到对象库的引用？

可以在Visual Basic编辑器中，使用“工具”、“引用”菜单来添加。

6) 判断题：要在自动化中使用Word的话，就必须能够从系统访问Word。

正确。要使用Word自动化的话，就必须能够访问它。

7) 判断题：多数Office应用程序支持某种层次的自动化。

正确。多少Office应用程序支持自动化，要么是可以充当控制器应用程序，要么是可以充当服务器应用程序，要么两种都能够充当。

练习题

下面是练习题的完整过程：

```
1: Sub Hour23exercise ()
```

```
2: 'This code assumes the active cell
3: 'is in column A.
4:
5: Dim y As Word.Application
6: Set y = CreateObject("Word.Application")
7:
8: With y
9:   .Visible = True
10:  'Your document location may be different from the
11:  'one shown below. Make necessary changes.
12:  .Documents.Open Filename:= "d:\excel vba\Invoice Letter.doc"
13:  ActiveCell.Copy
14:  .Selection.GoTo What:=wdGoToBookmark, Name:= "InvoiceNumber"
15:  .Selection.Paste
16:  Application.CutCopyMode = False
17:  ActiveCell.Offset(0, 1).Select
18:  Selection.Copy
19:  .Selection.GoTo What:=wdGoToBookmark, Name:= "Quantity"
20:  .Selection.Paste
21:  Application.CutCopyMode = False
22:  Selection.Offset(0, 1).Select
23:  Selection.Copy
24:  .Selection.GoTo What:=wdGoToBookmark, Name:= "Price"
25:  .Selection.Paste
26:  Application.CutCopyMode = False
27:  Selection.Offset(0, 1).Select
28:  Selection.Copy
29:  .Selection.GoTo What:=wdGoToBookmark, Name:= "Total"
30:  .Selection.Paste
31:  Application.CutCopyMode = False
32: End With
33:
34: End Sub
```

24. 第24学时答案

思考题

1) 如果想要过程在使用 MS Query 检索到数据时运行某个过程的话 , 应该使用哪个事件属性 (或者说事件过程) ?

OnData 属性。

2) 判断题 : 在同一个工作簿中可以拥有多个 Auto_Open 过程。

错误。每个工作簿只能有一个 Auto_Open 过程和一个 Auto_Close 过程。

3) 至少说出两个能用来分配过程以执行数据有效性验证的事件属性。

可以使用 OnCalculate、 OnEntry、 OnSheetDeactivate、 Onkey 等等。

4) 判断题 : 每次想要使用自定义的加载宏时 , 就必须手工加载这些加载宏。

错误。通过 “ 工具 ” 菜单选择加载宏后 , 以后每次在 Excel 启动时 , 加载宏就能够自动

加载。

5) 在关闭某个工作簿时，需要保存一个外部的 Word文件，这时应该怎么做？

编写过程，使用自动化来保存外部的 Word文件，并且把这个过程命名为 Auto_Close。

6) 至少说出通过加载宏能完成的两件事情。

存储用户自定义的函数、自定义的对话框、自定义的菜单和自定义的工具栏。

练习题

下面是练习题的完整过程：

```
Sub Auto_Open ()  
    With ActiveSheet  
        .Range("A1").Value = "Date: "  
        .Range("B1").FormulaR1C1 ="=NOW()"  
        .Range("A2").Value = "Name: "  
        .Range("B2").Value = Application.UserName  
        .Range("A3").Value = "Company Name: "  
        .Range("B3").Value = Application.OrganizationName  
        .Columns("A:A").EntireColumn.AutoFit  
        .Columns("B:B").EntireColumn.AutoFit  
    End With  
End Sub
```