

第5章 工作簿操作

前面章节已经分别讲解了单元格、工作表对象。在 Excel VBA 中，工作簿操作则是前面各对象操作的基础。例如，用户如果希望对工作表或者单击格进行操作，首先必须打开工作簿对象。在用户对工作表或者单元格对象操作结束后，必须对工作簿进行保存，才能让所有的操作保存。基于工作簿的 VBA 代码主要有新建、打开、保存工作簿，以及工作簿的保护与撤消等相关代码。本章将详细讲解如何使用 VBA 代码操作工作簿。

5.1 操作工作簿

在 Excel VBA 中，单个工作簿对应 Workbook 对象，而多个 Workbook 对象则组成 Workbooks 集合。工作簿的操作包括新建、保存工作簿等。本节介绍常用的操作工作簿的实例。

案例 91 新建工作簿

1. 功能说明

在实际开发中，用户可能需要在特定的时候创建新的工作簿。用户可以使用 VBA 代码实现该功能。

2. 语法说明

在 Excel VBA 中，用户可以使用 Workbooks 集合对象的 Add 方法新建工作簿。该方法的语法格式如下：

表达式.Add(Template)

参数 Template 确定如何创建新工作簿。如果此参数为指定现有 Excel 文件名的字符串，那么创建新工作簿将以该指定的文件作为模板。如果此参数为常量，新工作簿将包含一个指定类型的工作表。可为以下常量之一：

- xlWBATChart：图表；
- xlWBATExcel4IntlMacroSheet：Excel 版本 4 宏；
- xlWBATExcel4MacroSheet：Excel 版本 4 国际宏；
- xlWBATWorksheet：工作表。

如果省略此参数，Excel 将创建包含一定数目空白工作表的新工作簿（该数目由 SheetsInNewWorkbook 属性设置）。Excel 自动为新创建的工作簿命名为“BookN”，其中“N”是下一个可用的数字。新工作簿将成为活动工作簿。创建新工作簿更好的方法是将其分配

给一个对象变量，在程序中可通过该对象变量对工作簿进行设置。使用对象变量可以很容易地控制新工作簿。

3. 案例说明

某公司统计了部分员工的销量数据以及员工所在的地区，为了能够进一步分析对应的数据，用户需要新建工作簿，原始数据如图 5.1 所示。



	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	

图 5.1 原始数据

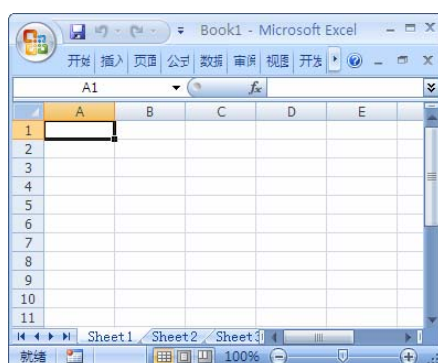
4. 编写代码

新建工作簿的 VBA 代码如下：

```
Sub AddNewWorkbook()
    Dim Wb As Workbook
    Application.DisplayAlerts = False
    Set Wb = Workbooks.Add
    Application.DisplayAlerts = True
End Sub
```

5. 运行结果

运行程序代码，查看新建的工作簿，如图 5.2 所示。



	A	B	C	D	E
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					

图 5.2 新建工作簿

6. 程序分析

在上面的代码中，首先从工作表中获取要创建工作簿的数量，再循环使用 **Add** 方法新建工作簿，保存工作簿后再关闭退出当前工作簿。

案例 92 打开工作簿

1. 功能说明

在用户在使用 VBA 编写数据处理的代码时，经常需要处理多个工作簿中的数据。这个时候，用户需要使用 VBA 代码打开其他的工作簿。

2. 语法说明

在 Excel VBA 中，用户可以使用 **Workbooks** 的 **Open** 方法打开一个工作簿时，该工作簿将成为 **Workbooks** 集合的成员。下述代码打开当前目录中的“×××××.xls”工作簿。

```
Sub OpenUp()  
    Workbooks.Open("×××××.xls")  
End Sub
```

在更多的时候，打开工作簿时需要查找文件所在位置，这时可通过“打开”对话框来进行查找，引用“打开”对话框的语法格式如下：

```
Application.GetOpenFilename(FileFilter, FilterIndex, Title, ButtonText, MultiSelect)
```

使用 **GetOpenFilename** 方法返回选定的文件名或用户输入的名称。返回的名称可能包含路径说明。如果用户取消了对话框，则该值为 **False**。

对打开的工作簿，可使用 **Close** 方法进行关闭，且不退出 Excel 程序。如果某个打开的工作簿有改动，Excel 将显示询问是否保存更改的对话框和相应提示。

提示：Workbooks 集合还提供了 **OpenDatabase**、**OpenText** 和 **OpenXML** 方法，分别用来打开数据库、文本文件和 XML 数据文件。

3. 案例说明

某公司统计了部分员工的销量数据以及员工所在的地区，为了能够进一步分析对应的数据，用户需要打开某各数据工作簿，原始数据如图 5.3 所示。



The screenshot shows an Excel spreadsheet with the following data:

	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	
11					

图 5.3 原始数据

4. 编写代码

打开工作簿的 VBA 代码如下：

```
Sub OpenBooks()  
Workbooks.Open "D:\Excel VBA\MyWorkBook.xlsx"  
End Sub
```

5. 运行结果

运行程序代码，查看打开的工作簿，如图 5.4 所示。其原始工作簿如图 5.5 所示。

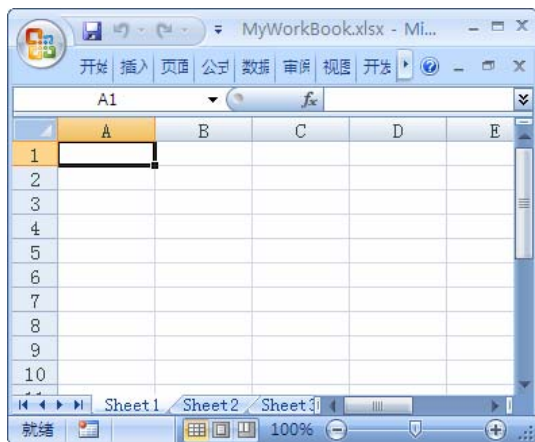


图 5.4 打开工作簿

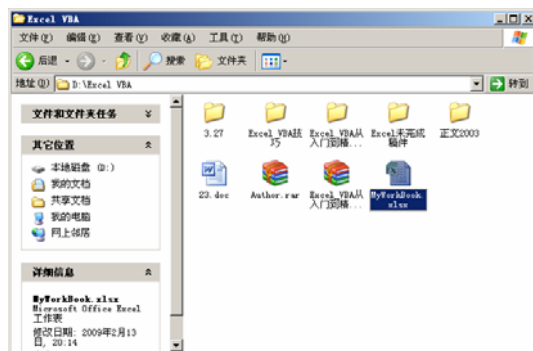


图 5.5 原始工作簿

6. 程序分析

在上面的代码中，首先通过 `GetOpenFilename` 方法显示“打开”对话框，如果用户未选择要打开的工作簿，则将反复显示“打开”对话框。

案例 93 导入文本文件

1. 功能说明

Excel 可以和其他文件格式的数据进行数据交互，在本小节中，将演示如何使用 VBA 代码导入文本文件中的数据。

2. 语法说明

在 Excel VBA 中，用户可以使用 Workbooks 集合对象的 OpenText 方法，可载入一个文本文件，并将其作为包含单个工作表的新工作簿进行分列处理，然后在此工作表中放入经过分列处理的文本文件数据。该方法的语法格式如下：

```
表达式.OpenText(Filename, Origin, StartRow, DataType, TextQualifier, ConsecutiveDelimiter, Tab, Semicolon, Comma, Space, Other, OtherChar, FieldInfo, TextVisualLayout, DecimalSeparator, ThousandsSeparator, TrailingMinusNumbers, Local)
```

该方法中，除了 Filename 为必需的参数之外，其他参数都可省略。各参数的含义如下：

- **Filename**：指定要打开和分列的文本文件的名称。
- **Origin**：指定文本文件来源。可为以下常量 xlMacintosh、xlWindows 或 xlMSDOS。此外，它还可以是一个整数，表示所需代码页的代码页编号。例如，“1256”指定源文本文件的编码是阿拉伯语。如果省略该参数，则此方法将使用“文本导入向导”中“文件原始格式”选项的当前设置。
- **StartRow**：文本分列处理的起始行号。默认值为 1。
- **DataType**：指定文件中数据的列格式。可为常量 xlDelimited 或 xlFixedWidth。如果未指定该参数，则 Excel 将尝试在打开文件时确定列格式。
- **TextQualifier**：指定文本识别符号。
- **ConsecutiveDelimiter**：如果为 True，则将连续分隔符视为一个分隔符。默认值为 False。
- **Tab**：如果为 True，则将制表符用作分隔符（DataType 必须为 xlDelimited）。默认值为 False。
- **Semicolon**：如果为 True，则将分号用作分隔符（DataType 必须为 xlDelimited）。默认值为 False。
- **Comma**：如果为 True，则将逗号用作分隔符（DataType 必须为 xlDelimited）。默认值为 False。
- **Space**：如果为 True，则将空格用作分隔符（DataType 必须为 xlDelimited）。默认值为 False。
- **Other**：如果为 True，则将 OtherChar 参数指定的字符用作分隔符（DataType 必须为 xlDelimited）。默认值为 False。
- **OtherChar**：（如果 Other 为 True，则为必选项）。当 Other 为 True 时，指定分隔符。如果指定了多个字符，则仅使用字符串中的第一个字符而忽略剩余字符。
- **FieldInfo**：包含单列数据相关分列信息的数组。对该参数的解释取决于 DataType 的值。如果此数据由分隔符分隔，则该参数为由两元素数组组成的数组，其中每个两元素数

组指定一个特定列的转换选项。第一个元素为列标（从 1 开始），第二个元素是 `XlColumnDataType` 的常量之一，用于指定分列方式。

- `TextVisualLayout`: 文本的可视布局。
- `DecimalSeparator`: 识别数字时，Excel 使用的小数分隔符。默认设置为系统设置。
- `ThousandsSeparator`: 识别数字时，Excel 使用的千位分隔符。默认设置为系统设置。
- `TrailingMinusNumbers`: 如果应将结尾为减号字符的数字视为负数处理，则指定为 `True`。如果为 `False` 或省略该参数，则将结尾为减号字符的数字视为文本处理。
- `Local`: 如果分隔符、数字和数据格式应使用计算机的区域设置，则指定为 `True`。

3. 案例说明

在本例中，用户需要导入文本文件中的数据。在默认情况下，Excel 文件中不包含任何的数据，如图 5.6 所示。

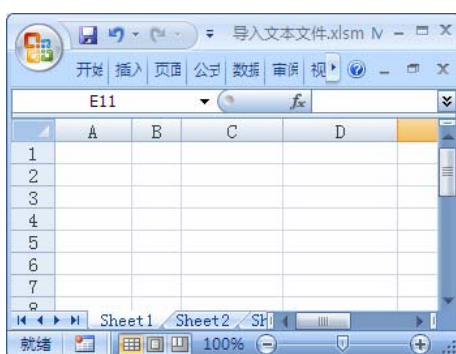


图 5.6 原始数据

4. 编写代码

本例的代码很简单，具体如下：

```
Sub OpenTextFiles()  
    Workbooks.OpenText Filename:="D:\Excel VBA\销售表.txt", _  
        DataType:=xlDelimited, Tab:=True  
End Sub
```

5. 运行结果

运行程序代码，查看打开的工作簿，如图 5.7 所示。

	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	
11					

图 5.7 打开的文本文件

原始的文本文件路径如图 5.8 所示。

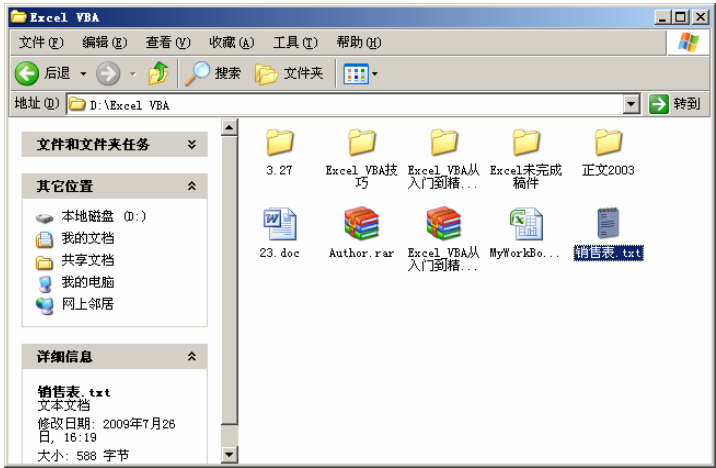
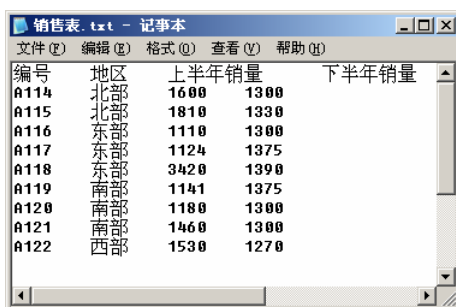


图 5.8 文本文件的路径

默认情况下，文本文件中的数据如图 5.9 所示。



编号	地区	上半年销量	下半年销量
A114	北部	1600	1300
A115	北部	1810	1330
A116	东部	1110	1300
A117	东部	1124	1375
A118	东部	3420	1390
A119	南部	1141	1375
A120	南部	1180	1300
A121	南部	1460	1300
A122	西部	1530	1270

图 5.9 文本文档的数据

6. 程序分析

在上面的程序代码中，主要为了演示如何导入文本文件，从上面的参数设置可以看出，用户可以设置关于导入的各种属性，这里就不重复介绍。

案例 94 保存工作簿

1. 功能说明

保存工作簿是十分常见的操作，用户可以通过 VBA 代码实现该操作。

2. 语法说明

在 Excel VBA 中，当用户需要保存工作簿时，可以使用 Workbook 对象的 Save 方法，该方法将工作簿的修改保存到磁盘中，不需要任何参数。

同时，通过 Workbook 对象的 Path 属性，可获取工作簿的完整路径，不包括末尾的分隔符和应用程序名称。通过 Open 方法打开的工作簿，其 Path 属性不为空。而新建的工作簿，其 Path 属性为空。

3. 案例说明

某公司统计了部分员工的销量数据以及员工所在的地区，为了能够进一步分析对应的数据，需要对工作簿所做的修改进行保存，原始数据如图 5.10 所示。

	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	
11					

图 5.10 原始数据

4. 编写代码

保存工作簿按钮的 VBA 代码如下：

```
Sub SaveAllBooks()

    Dim Wb As Workbook

    For Each Wb In Workbooks
        If Wb.Path <> "" Then
            Wb.Save
        End If
    Next Wb
End Sub
```

5. 运行结果

运行程序代码，查看打开的工作簿，如图 5.11 所示。

	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	
11					

图 5.11 保存工作簿

6. 程序分析

在上面代码中，根据工作簿的 **Path** 参数来判断是否为新建工作簿，若 **Path** 属性值不为空，则保存该工作簿。

案例 95 重命名保存工作簿

1. 功能说明

在用户使用 Excel VBA 处理数据的时候，经常需要在工作簿进行编辑后，重命名进行保存，用户可以使用 VBA 代码实现该功能。

2. 语法说明

在 Excel VBA 中，用户可以使用 **SaveAs** 方法将工作簿换名保存，其语法格式如下：

表达式.SaveAs(FileName, FileFormat, Password, WriteResPassword, ReadOnlyRecommended, CreateBackup, AccessMode, ConflictResolution, AddToMru, TextCodepage, TextVisualLayout, Local)

SaveAs 方法的参数都可以省略。各参数的含义如下：

- **Filename**：表示要保存文件的文件名的字符串。可包含完整路径，如果不指定路径，Excel 将文件保存到当前文件夹中。
- **FileFormat**：保存文件时使用的文件格式。对于现有文件，默认采用上一次指定的文件格式；对于新文件，默认采用当前所用 Excel 版本的格式。
- **Password**：它是区分大小写的字符串，用于指定文件的保护密码。
- **WriteResPassword**：表示文件写保护密码的字符串。如果文件保存时带有密码，但打开文件时不输入密码，则该文件以只读方式打开。
- **ReadOnlyRecommended**：如果为 **True**，则在打开文件时显示一条消息，提示该文件以只读方式打开。
- **CreateBackup**：如果为 **True**，则创建备份文件。
- **AccessMode**：工作簿的访问模式。
- **ConflictResolution**：它确定该方法在保存工作簿时如何解决冲突。如果设为 **xlUserResolution**，则显示冲突解决对话框。如果设为 **xlLocalSessionChanges**，则自动接受本地用户的更改。如果设为 **xlOtherSessionChanges**，则自动接受来自其他会话的更改（而不是本地用户的更改）。如果省略此参数，则显示冲突处理对话框。
- **AddToMru**：如果为 **True**，则将该工作簿添加到最近使用的文件列表中。默认值为 **False**。
- **TextCodepage** 和 **TextVisualLayout**：不在美国英语版的 Excel 中使用。
- **Local**：如果为 **True**，则以 Excel 的语言保存文件。如果为 **False**（默认值），则以 VBA 的宏语言版本的语言保存文件。

3. 案例说明

某公司统计了部分员工的销量数据以及员工所在的地区，为了能够进一步分析对应的

数据，需要重命名保存工作簿，原始数据如图 5.12 所示。



The screenshot shows an Excel spreadsheet with the following data:

	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	

图 5.12 原始数据

4. 编写代码

重命名保存工作簿的 VBA 代码如下：

```
Sub SaveOther()  
  
Dim oldName As String  
Dim newName As String  
Dim FolderName As String  
Dim fname As String  
    oldName = ActiveWorkbook.Name  
    newName = "重命名: " & oldName  
  
    MsgBox "将<" & oldName & ">以<" & newName & ">的名称保存"  
  
    FolderName = Application.DefaultFilePath  
    fname = FolderName & "\" & newName  
  
    ActiveWorkbook.SaveAs fname  
End Sub
```

5. 运行结果

运行程序代码，查看重命名保存的工作簿，如图 5.13 所示。



图 5.13 提示保存信息

单击对话框中的“确定”按钮后，查看保存后的文件，如图 5.14 所示。



The screenshot shows an Excel spreadsheet with the following data:

	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	

图 5.14 查看重命名的文件

该重命名的工作簿保存在默认路径中，如图 5.15 所示。

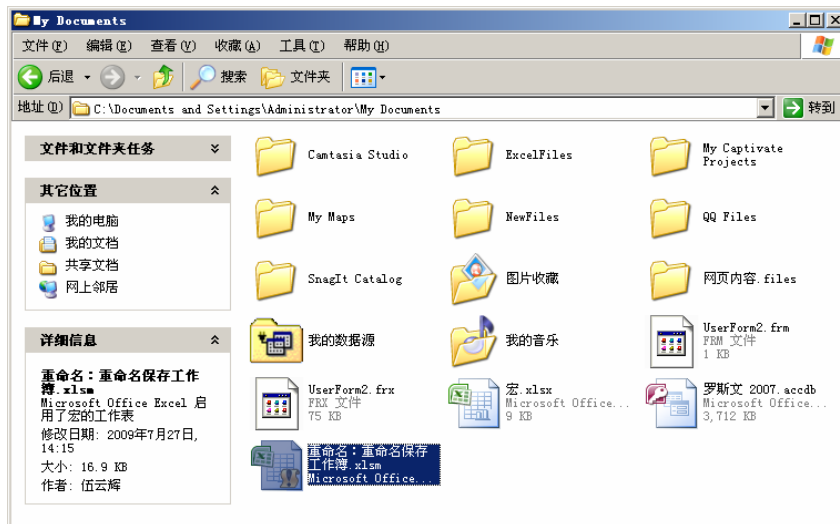


图 5.15 查看文件保存的路径

6. 程序分析

在上面的代码中，通过 `Application.DefaultFilePath` 获取程序代码的默认路径。因此，本程序将新创建的工作簿保存在默认路径中。

案例 96 将工作簿保存为网页

1. 功能说明

用户可以将工作簿保存为各种其他格式，本小节中，将结合具体的例子来演示如何将工作簿保存为网页。

2. 语法说明

在 Excel VBA 中，用户使用 Workbook 的 SaveAs 方法可以将文档保存为 Web 页，有关该方法的语法格式参见上例中的介绍。

在本例中，要将工作簿保存为 Web 页，所以需要将 SaveAs 的参数 FileFormat 设置为常量 xlHtml。保存工作表时的文件格式常量在 xlFileFormat 枚举类型中，有关常量值可通过 Excel 的帮助查找。

3. 案例说明

某公司统计了部分员工的销量数据以及员工所在的地区，为了能够进一步分析对应的数据，需要将工作簿保存为网页格式，原始数据如图 5.16 所示。



	A	B	C	D	E	F
1	编号	地区	上半年销量	下半年销量		
2	A114	北部	1600	1300		
3	A115	北部	1810	1330		
4	A116	东部	1110	1300		
5	A117	东部	1124	1375		
6	A118	东部	3420	1390		
7	A119	南部	1141	1375		
8	A120	南部	1180	1300		
9	A121	南部	1460	1300		
10	A122	西部	1530	1270		

图 5.16 原始数据

4. 编写代码

本例代码很简单，具体如下：

```
Sub SaveAsWeb()  
    ActiveWorkbook.SaveAs Filename:="销售表.htm", FileFormat:=xlHtml  
End Sub
```

5. 运行结果

运行程序代码，查看重命名保存的工作簿，如图 5.17 所示。

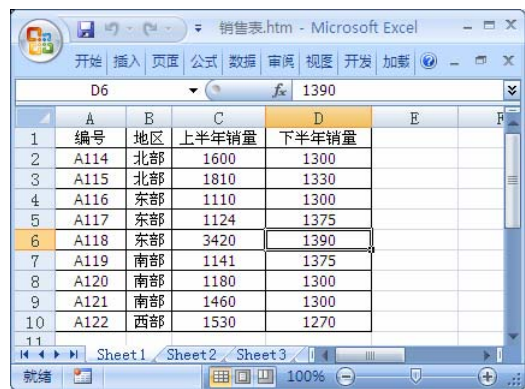


图 5.17 保存的 web 文件

用户可以查看网页格式文件保存的路径，如图 5.18 所示。



图 5.18 文件的保存路径

同时，用户可以通过 IE 打开保存的 web 网页文件，如图 5.19 所示。



图 5.19 用 IE 打开文档

6. 程序分析

和在 Excel 文档中一样，用户可以在 IE 中查看其他工作表的内容，如图 5.20 所示。

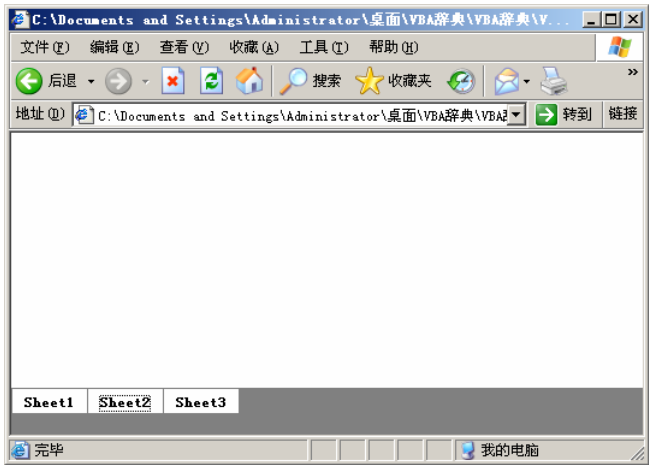


图 5.20 查看其他工作表

案例 97 查看工作簿信息

1. 功能说明

工作簿的信息对于大型软件程序开发是十分重要的信息，用户可以通过 VBA 代码获取工作簿的信息。

2. 语法说明

在 Excel VBA 中，用户可以通过访问 `ThisWorkbook` 的属性获取工作簿的信息。表 5.1 列出工作簿的一些通用属性，使用这些通用属性在组织大的程序，定位程序目录等方面具有非常重要的用途。

表 5.1 工作簿的一些通用属性

序号	名称	类型	意义
1	Name	String	只是返回名称部分
2	Path	String	只返回路径部分
3	FullName	String	返回完整路径名称，包括工作簿文件名
4	CodeName	String	返回对象的代码名。
5	ReadOnly	String	工作簿对象是否以只读方式打开
6	Saved	String	工作簿从上次保存至今是否发生过更改

3. 案例说明

本例的主要功能是在工作表中显示工作簿的基本信息，其中，需要了解其信息的原始数据如图 5.21 所示。

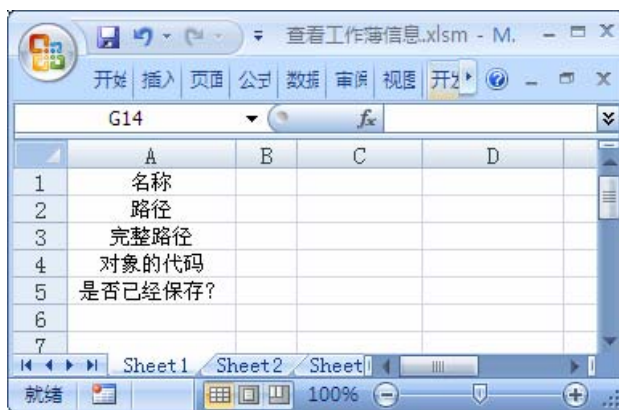


图 5.21 原始表格

4. 编写代码

显示工作簿信息的 VBA 代码如下：

```
Sub ShowInfo()  
With ActiveSheet  
    .Range("B1").Value = ThisWorkbook.Name  
    .Range("B2").Value = ThisWorkbook.Path  
    .Range("B3").Value = ThisWorkbook.FullName  
    .Range("B4").Value = ThisWorkbook.CodeName  
    .Range("B5").Value = ThisWorkbook.Saved  
End With
```


End Sub

5. 运行结果

运行程序代码，查看工作簿的信息，如图 5.22 所示。

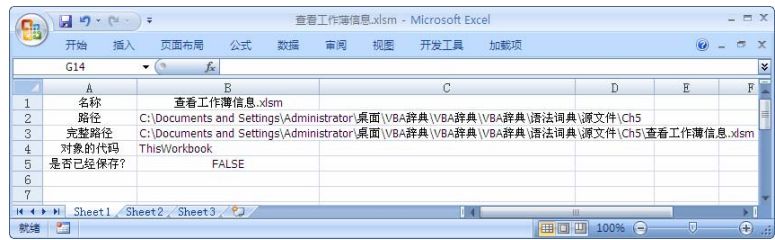


图 5.22 查看工作簿的信息

6. 程序分析

在 Excel VBA 中，还提供了关于工作簿的很多其他属性。用户在编写程序代码的时候，可以根据需要访问和设置这些属性。

案例 98 查看工作簿的内置属性

1. 功能说明

工作簿的内置属性和其普通属性不同，通过访问工作簿的内置属性，用户可以了解到保存时间、用户名等信息。用户可以通过 VBA 获取工作簿的内置属性。

2. 语法说明

在 Excel VBA 中，工作簿的保存时间属于内在信息，和前面例子中的信息不同。用户可以按照下面的步骤查看和修改文档的属性：

(1) 单击“Office 按钮”打开下拉菜单。选择“准备”|“属性”命令，在 Excel 的功能区下方将打开如图 5.23 所示的“文档属性”面板，供用户查看和修改。

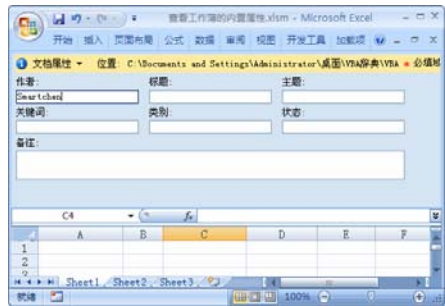


图 5.23 文档属性

(2) 单击左上角“文档属性”标签，打开下拉菜单选择“高级属性”，打开如图 5.24 所示的属性窗口，在其“自定义”选项卡中可定义各种属性值。

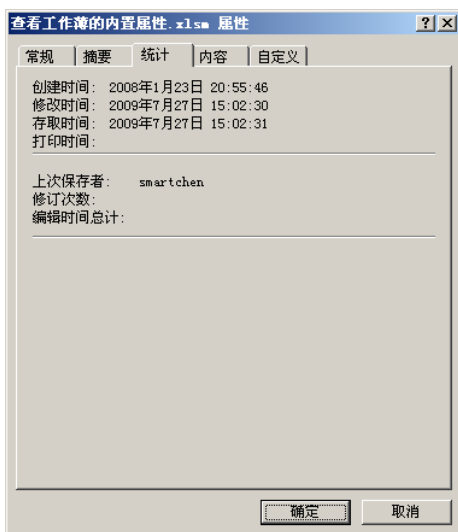


图 5.24 查看内在属性

在 Excel VBA 中，用户可以通过代码来获取并且设置文档的属性值。实际上，使用 **Workbook** 对象的 **BuiltinDocumentProperties** 属性返回 **DocumentProperties** 集合，该集合表示工作簿的所有内置文档属性，通过指定属性的名称或集合索引号，用 **Item** 方法可返回集合中的单个成员 (**DocumentProperty** 对象)。通过 **DocumentProperty** 对象的属性可操作属性，该对象的常用属性值如下：

- **Name** 属性：获取或设置文档属性的名称。
- **Value** 属性：获取或设置文档属性的值。
- **Type** 属性：获取或设置文档属性类型。对于内置文档属性为只读；对于自定义文档属性为可读/写。该属性使用 **msoDocProperties** 枚举类型，具体各枚举常量如下：
- **msoPropertyTypeBoolean**: Boolean。
- **msoPropertyTypeDate**: Date。
- **msoPropertyTypeFloat**: Floating point。
- **msoPropertyTypeNumber**: Integer。
- **msoPropertyTypeString**: String。

3. 案例说明

本例的主要检测运行的工作簿是否保存。如果用户没有保存过工作簿，则系统显示“工作簿未保存”。如果用户在之前某个时刻保存过该工作簿，则显示工作簿的保存信息，原始工作簿文件如图 5.25 所示。

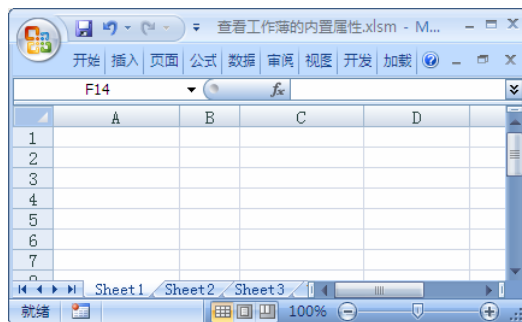


图 5.25 原始表格

4. 编写代码

显示工作簿的保存时间信息的 VBA 代码如下：

```
Sub ShowSaveInfo()  
    Dim SaveTime As String  
  
    SaveTime = ActiveWorkbook.BuiltinDocumentProperties("Last Save Time").Value  
  
    If SaveTime = "" Then  
        MsgBox ActiveWorkbook.Name & "工作簿没有保存."  
    Else  
        MsgBox "本工作簿在" & SaveTime & "保存", , ActiveWorkbook.Name  
    End If  
  
End Sub
```

5. 运行结果

运行程序代码，查看工作簿的信息，如图 5.26 所示。

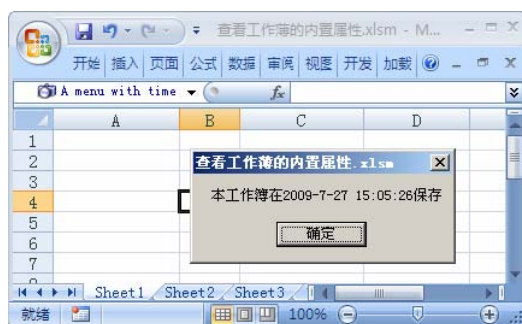


图 5.26 查看工作簿的保存时间

6. 程序分析

在上面的程序代码中，通过 `BuiltinDocumentProperties` 的 `Last Save Time` 成员访问工作

薄的最后修改时间，如果用户没有保存过该工作簿，系统会返回空文本，则显示没有保存。如果已经保存过，则系统返回修改的时间。

案例 99 设置工作簿密码

1. 功能说明

在 Excel VBA 中，为了工作簿的安全，可以为工作簿设置打开权限密码。用户可以通过 VBA 代码设置工作簿的密码。

2. 语法说明

在 Excel VBA 中，使用 Workbook 对象的 Password 属性可获取或设置该密码。要取消工作簿的密码，可设置其 Password 属性为空字符串，代码如下：

```
ActiveWorkbook.Password = ""
```

在 Excel 中，用户也可以直接通过操作来设置文档的密码。单击“Office 按钮”打开下拉菜单，从下拉菜单中选择“准备”|“加密文档”命令，如图 5.27 所示。打开“加密文档”对话框，如图 5.28 所示，在其中可以设置工作簿的密码。

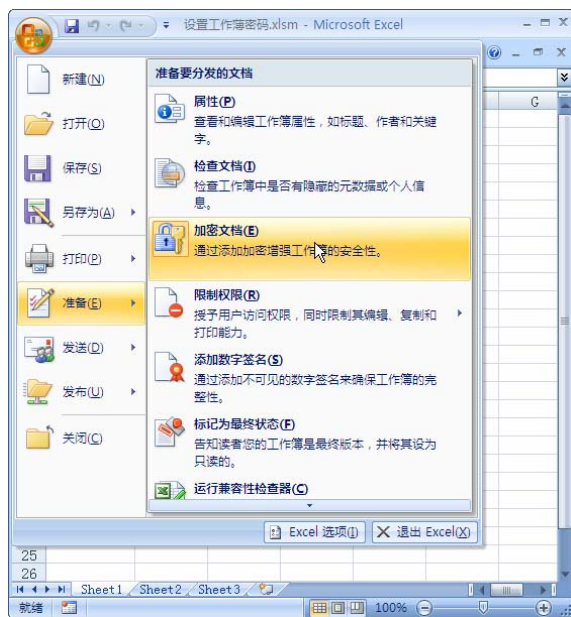


图 5.27 选择加密文档选项

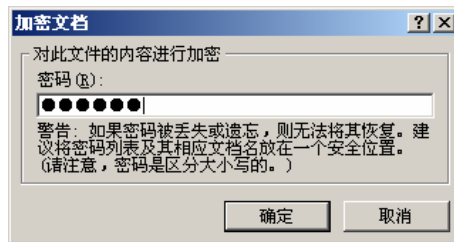



图 5.28 设置对应的密码

3. 案例说明

某公司统计了部分员工的销量数据以及员工所在的地区，为了能够进一步分析对应的数据，用户需要设置工作簿密码，原始数据如图 5.29 所示。



	A	B	C	D	E	F
1	编号	地区	上半年销量	下半年销量		
2	A114	北部	1600	1300		
3	A115	北部	1810	1330		
4	A116	东部	1110	1300		
5	A117	东部	1124	1375		
6	A118	东部	3420	1390		
7	A119	南部	1141	1375		
8	A120	南部	1180	1300		
9	A121	南部	1460	1300		
10	A122	西部	1530	1270		
11						
12						

图 5.29 原始数据

4. 编写代码

设置工作簿密码的具体代码如下：

```
Sub SetBookPassWord()  
  
    With ActiveWorkbook  
        .Password = InputBox("输入工作簿的密码:")  
        .Save  
        .Close True  
    End With  
  
End Sub
```

5. 运行结果

运行程序代码，设置工作簿的密码，如图 5.30 所示。

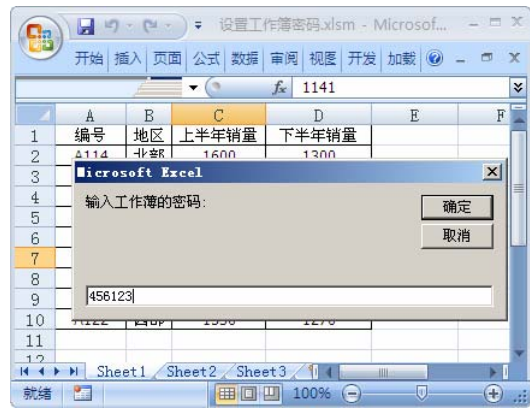


图 5.30 设置工作簿的密码

当用户单击对话框中的“确定”按钮后，系统会关闭工作簿，如图 5.31 所示。

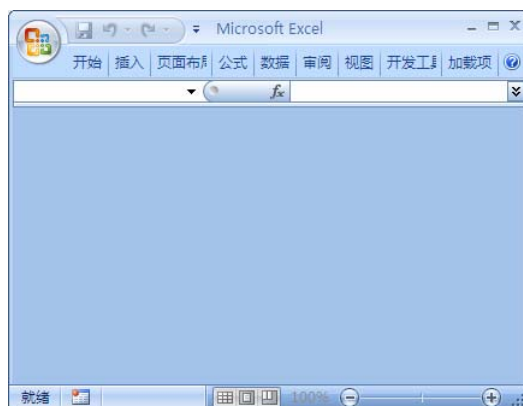


图 5.31 关闭工作簿

当用户再次打开该工作簿的时候，需要输入密码，如图 5.32 所示。

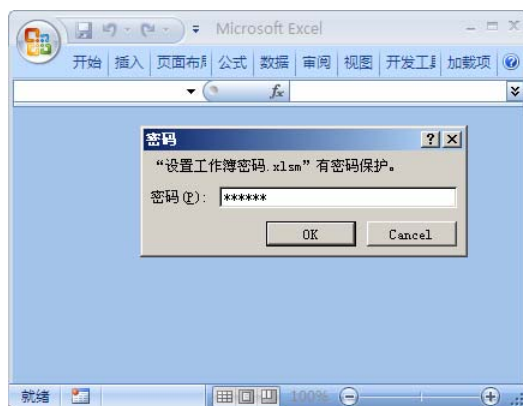


图 5.32 输入密码

当用户输入密码错误的时候，系统会提示密码输入错误，如图 5.33 所示。



图 5.33 密码错误提示信息

6. 程序分析

该程序代码所实现的功能和用户自行设置工作簿的密码类似。

案例 100 查看用户状态信息

1. 功能说明

用户状态是一种特殊的信息,通过用户状态信息,用户可以了解工作簿的创建时间等。用户可以通过 VBA 代码获取用户状态的信息。

2. 语法说明

在 Excel VBA 中,用户可以使用 `Workbook.UserStatus` 属性来获取用户的信息。`UserStatus` 属性返回二维数组,该数组提供有关以共享列表模式打开工作簿的用户的信息。其语法表达式如下:

`表达式.UserStatus`

其中的表达式表示 `Workbook` 对象的变量。其中,数组第二维的第一个元素为用户名,第二个元素是用户打开工作簿的日期和时间,第三个元素是表示清单类型的数字,1 表示独占,2 表示共享。`UserStatus` 属性不返回以只读方式打开指定工作簿的用户的信息。

同时,为了逐个返回数组的信息,需要使用 `UBound` 函数。该函数的主要功能是返回 `Long` 型数据,其值为指定的数组维可用的最大下标。其语法表达式是:

`UBound(arrayname[, dimension])`

其中参数的具体说明如下:

- `arrayname`: 数组变量的名称,遵循标准变量命名约定。
- `dimension`: 指定返回哪一维的上界。1 表示第一维,2 表示第二维,如此等等。如果省略 `dimension`,就认为是 1。

3. 案例说明

本例的主要功能是在工作表中显示工作簿的用户状态信息,其中,需要了解其信息的原始数据如图 5.34 所示。

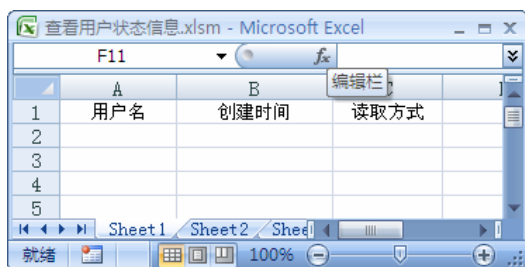


图 5.34 原始表格

4. 编写代码

显示用户状态信息的 VBA 代码如下:

```
Sub ShowUserInfo()
```

```
Dim UserInfo() As Variant
Dim Row As Integer

UserInfo = ActiveWorkbook.UserStatus

With ActiveWorkbook.Sheets(1)
    For Row = 1 To UBound(UserInfo, 1)
        .Cells(Row + 1, 1) = UserInfo(Row, 1)
        .Cells(Row + 1, 2) = UserInfo(Row, 2)
        Select Case UserInfo(Row, 3)
            Case 1
                .Cells(Row + 1, 3).Value = "独占"
            Case 2
                .Cells(Row + 1, 3).Value = "共享"
        End Select
    Next
End With

End Sub
```

5. 运行结果

运行程序代码，查看用户状态的信息，如图 5.35 所示。

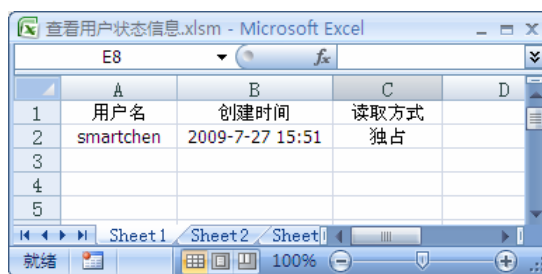


图 5.35 查看用户状态的信息

6. 程序分析

在上面的程序代码中，为了显示读取方式，直接将返回的数值转换为对应的读取方式。因此，在最后的工作簿单元格中显示的是读取方式。

案例 101 激活工作簿

1. 功能说明

在 Excel VBA 中，为用户激活工作簿提供了对应的事件。用户可以在对应事件中编写

代码，来实现激活工作簿时的操作。

2. 语法说明

在 Excel VBA 中，用户可以通过 `Workbook.Activate` 来激活工作簿。当用户激活工作簿、工作表、图表工作表或嵌入式图表时发生此事件。其语法表达式如下：

表达式.Activate

表达式表示 `Workbook` 对象的变量。新建窗口时不发生此事件。切换两个显示同一工作簿的窗口时，将发生 `WindowActivate` 事件，但不发生工作簿的 `Activate` 事件

3. 案例说明

本例的主要功能是依次激活打开的工作簿，直到最后一个打开的工作簿。其中，默认打开的工作簿如图 5.36 所示。

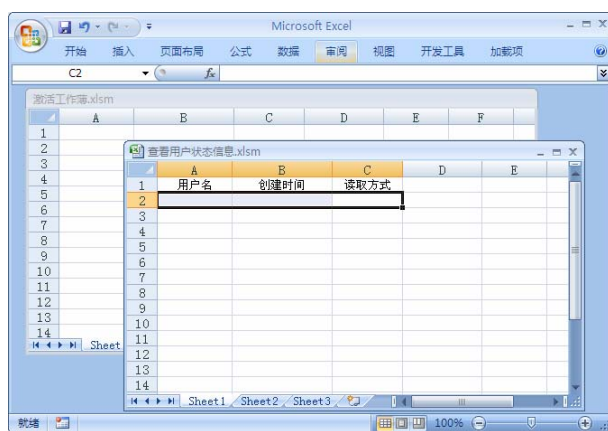


图 5.36 打开的工作簿

4. 编写代码

依次激活工作簿的 VBA 代码如下：

```
Sub ActiveWorkBooks()  
    Dim intNum As Long  
    Dim i As Integer  
    Dim strMsg As String  
  
    intNum = Workbooks.Count  
  
    For i = 1 To intNum  
        Workbooks(i).Activate  
        strMsg = MsgBox("第 " & i & "个工作簿被激活，继续激活？", vbYesNo)  
        If strMsg = vbNo Then  
            Exit Sub  
        End If  
    End For
```

```
If i = intNum Then  
    MsgBox "最后一个工作簿已被激活！"  
End If  
Next i  
  
End Sub
```

5. 运行结果

运行程序代码，查看激活第一个工作簿的情况，如图 5.37 所示。

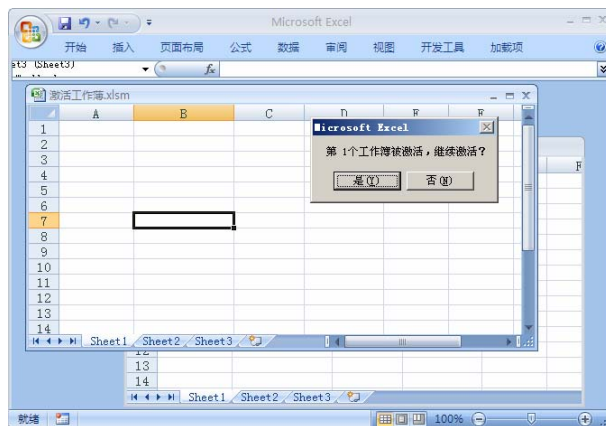


图 5.37 激活第一个工作簿

单击对话框中的“确定”按钮，继续激活第二个工作簿，如图 5.38 所示。

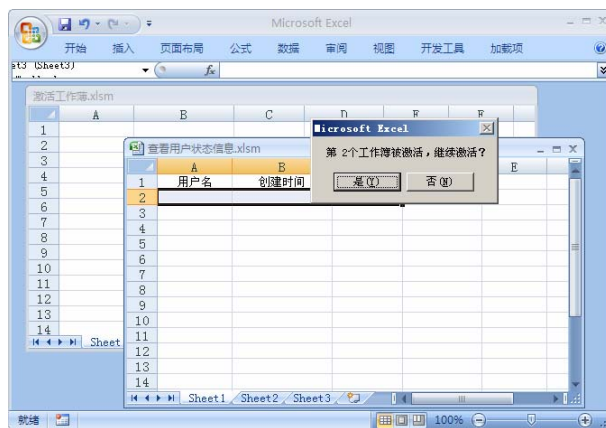


图 5.38 激活第二个工作簿

单击“确定”按钮，显示激活的信息，如图 5.39 所示。



图 5.39 显示激活信息

6. 程序分析

在上面的程序代码中，首先统计工作簿的个数，然后通过循环来依次激活现在已经打开的工作簿。

案例 102 保护工作簿

1. 功能说明

和其他对象类似，用户同样可以使用 VBA 代码来保护工作簿。

2. 语法说明

在 Excel VBA 中，用户可以使用 Workbook 对象的 Protect 方法对工作簿进行保护。其语法格式如下：

`表达式.Protect(Password, Structure, Windows)`

Protect 方法的三个参数都可省略，各参数的含义如下：

- **Password**：一个字符串，该字符串为工作簿的密码。如果省略此参数，不用密码就可以取消对工作簿的保护。否则，必须指定密码才能取消对工作簿的保护。如果忘记了密码，就无法取消对工作簿的保护。
- **Structure**：如果为 True，则保护工作簿结构，此时不能对工作簿中的工作表进行插入、复制、删除等操作。默认值是 False。
- **Windows**：如果为 True，则保护工作簿窗口，此时该工作簿右上角的最小化、最大化和关闭按钮消失。如果省略此参数，则窗口不受保护。默认值是 False。

3. 案例说明

某公司统计了部分员工的销量数据以及员工所在的地区，为了能够进一步分析对应的数据，用户需要保护工作簿，原始数据如图 5.40 所示。



	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	

图 5.40 原始数据

4. 编写代码

保护工作簿的 VBA 代码如下：

```
Sub SetProtectBooks()  
  
Dim pw As String  
    pw = Application.InputBox(prompt:="设置工作簿的密码: ", _  
        Title:="设置密码", Type:=2)  
ActiveWorkbook.Protect Password:=pw, Structure:=True, Windows:=True  
  
End Sub
```

5. 运行结果

运行程序代码，设置保护工作簿的密码，如图 5.41 所示。

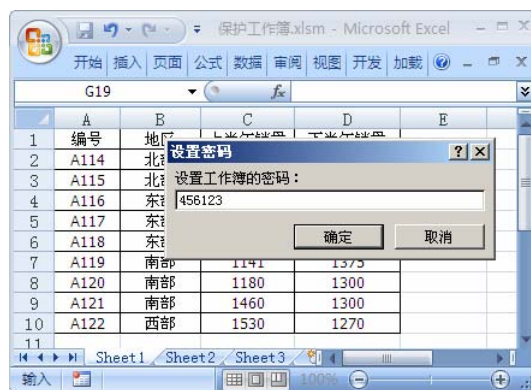


图 5.41 设置工作簿的密码

被保护的工作簿效果如图 5.42 所示，当用户在工作表标签上右击，弹出的快捷菜单中

“插入”、“删除”等相关操作工作表的菜单也变为灰色，不允许用户进行操作了。



图 5.42 保护的结果

为了撤销工作簿的保护，用户可以选择“审阅”|“更改”|“保护工作簿”|“保护结构和窗口”选项，如图 5.43 所示。



图 5.43 选择撤销工作簿的保护

在打开的对话框中输入密码，如图 5.44 所示。



图 5.44 输入密码

单击对话框中的“确定”按钮，查看撤销保护后的工作簿，如图 5.45 所示。

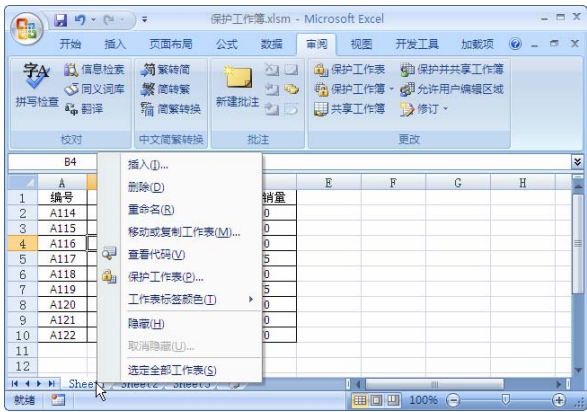


图 5.45 撤销工作簿

6. 程序分析

在 Excel VBA 中，用户使用 `Unprotect` 方法可取消工作簿的保护。如果工作簿不是受保护的，则此方法不起作用。该方法的语法格式如下：

```
表达式.Unprotect(Password)
```

参数 `Password` 指定用于解除工作簿保护的密码，此密码是区分大小写的。如果工作簿不设密码保护，则省略此参数。

案例 103 设置名称

1. 功能说明

名称是用户在处理数据时，经常用到的方法。在 Excel VBA 中，用户可以通过 `Add` 方法添加名称对象。

2. 语法说明

在 Excel VBA 中，使用 `Add` 方法可向 `Names` 集合中添加 `Name` 对象，使用该方法可创建一个新名称，具体的语法格式如下：

```
表达式.Add(Name, RefersTo, Visible, MacroType, ShortcutKey, Category, NameLocal, RefersToLocal, CategoryLocal, RefersToR1C1, RefersToR1C1Local)
```

各参数的含义如下：

- **Name**：如果没有指定 `NameLocal`，则为用作名称的文本（以宏语言表示）。名称不能包括空格，不能与单元格引用相似。
- **RefersTo**：除非指定了其他 `RefersTo` 参数，否则该参数说明名称引用的内容（使用 `A1` 格式表示法以宏语言表示）。如果引用不存在，则返回 `Nothing`。
- **Visible**：如果为 `True`，则用常规方式定义名称。如果为 `False`，则将名称定义为隐藏名

称（即该名称在“定义名称”、“粘贴名称”或“定位”对话框中都不出现）。默认值是 True。

- **MacroType**: 确定宏的类型，可取值为 1~3。1 表示用户定义函数（Function 过程），2 表示宏（也称为子过程），3 表示无（忽略该参数，即该名称不引用用户定义函数或宏）。
- **ShortcutKey**: 宏的快捷键。必须是单个字母，例如“z”或“Z”。只用于命令宏。
- **Category**: 如果 MacroType 为 1 或 2，则该参数为宏或函数的分类。该分类在“函数向导”中使用。可以用数字（从 1 开始）或名称（以宏语言指定）引用现有的分类。如果指定的分类不存在，Excel 将创建新分类。
- **NameLocal**: 如果没有指定 Name，则为用作名称的文本（以用户语言表示）。名称不能包括空格，不能与单元格引用相似。
- **RefersToLocal**: 除非指定了其他 RefersTo 参数，否则该参数说明名称引用的内容（使用 A1 格式表示法以用户语言表示）。
- **CategoryLocal**: 如果未指定 Category，则为以用户语言标识自定义函数分类的文本。
- **RefersToR1C1**: 除非指定了其他 RefersTo 参数，否则该参数说明名称引用的内容（使用 R1C1 格式表示法以宏语言表示）。
- **RefersToR1C1Local**: 除非指定了其他 RefersTo 参数，否则该参数说明名称引用的内容（使用 R1C1 格式表示法以用户语言表示）。

RefersTo 参数必须以 A1 样式表示法指定，包括必要时使用的美元符(\$)。例如，如果在工作表 Sheet1 上选定了单元格 A10，然后将 RefersTo 参数指定为“=Sheet1!A1:B1”而定义了一个名称，那么该名称实际上指向单元格区域 A10:B10（因为指定的是相对引用）。若要指定绝对引用，应当用“=Sheet1!\$A\$1:\$B\$1”。

3. 案例说明

某公司统计了部分员工的销量数据以及员工所在的地区，为了能够进一步分析对应的数据，用户为原始区域设置名称，原始数据如图 5.46 所示。



	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	
11					

图 5.46 原始数据

4. 编写代码

设置名称的 VBA 代码如下：

```
Public Sub AddNames()  
ActiveWorkbook.Names.Add                               Name:="DataSource",  
RefersToR1C1:="=Sheet1!R1C1:Sheet1!R10C4"  
End Sub
```

5. 运行结果

运行程序代码，然后选择“公式”|“定义的名称”|“名称管理器”选项，如图 5.47 所示。

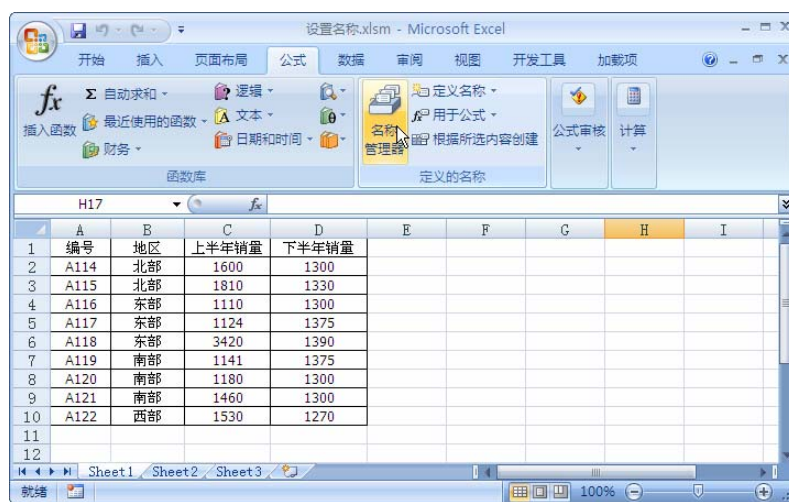


图 5.47 选择名称管理器

在用户选择按钮后，程序打开“名称管理器”对话框，查看添加的数据名称，如图 5.48 所示。

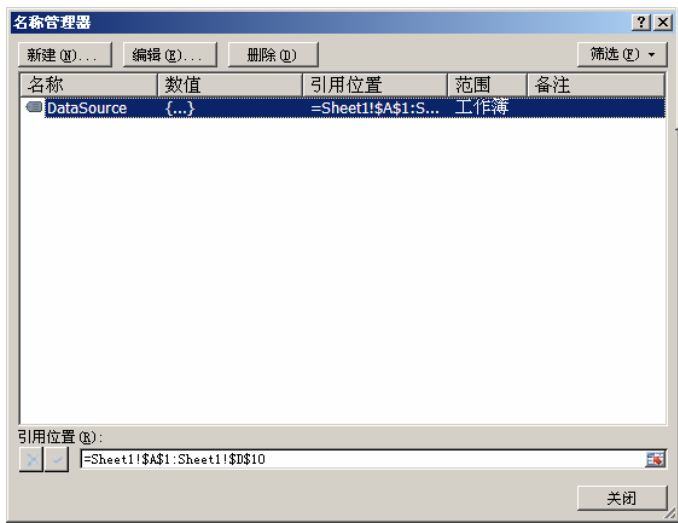


图 5.48 查看添加的数据名称

6. 程序分析

上面的程序代码十分简单，直接使用最基本的语法格式，设置了单元格的名称，然后通过 RefersToR1C1 的参数设置了单元格的区域范围。

案例 104 判断工作簿是否存在

1. 功能说明

在 VBA 程序中，打开不存在的工作簿，将产生错误。因此，在打开工作簿之前，先判断该文件是否存在是很有必要的。在本小节中，将结合具体的例子来演示如何判断工作簿是否存在。

2. 语法说明

在 Excel VBA 中，可以使用 VBA 的 Dir 函数来检查某些文件或目录是否存在。其语法格式如下：

Dir([pathname[, attributes]])

该函数的两个参数都可省略，其中 **pathname** 用来指定文件名的字符串表达式，可能包含文件夹、驱动器。如果没有找到 **pathname**，则会返回零长度字符串("")。Attributes 是一个常数或数值表达式，其总和用来指定文件属性。如果省略，则会返回匹配 **pathname** 但不包含属性的文件。

3. 案例说明

某公司统计了部分员工的销量数据以及员工所在的地区，为了能够进一步分析对应的数据，用户需要首先判断其他工作簿是否存在，原始数据如图 5.49 所示。



The screenshot shows an Excel spreadsheet with the following data:

	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	
11					

图 5.49 原始数据

4. 编写代码

判断工作簿是否存在的 VBA 代码如下：

```
Sub CheckWorkBooks()  
    Dim str As String  
  
    str = Application.InputBox(prompt:="输入工作簿的名称: ", _  
        Title:="文件名", Type:=2)  
  
    If str = "False" Then Exit Sub  
  
    If Not Len(Dir(str)) > 0 Then  
        MsgBox "工作簿" & str & "不存在！"  
    Else  
        Workbooks.Open str  
    End If  
  
End Sub
```

5. 运行结果

运行程序代码，在弹出的对话框中输入工作簿的名称，如图 5.50 所示。



图 5.50 输入文件的名称

单击对话框中的“确定”按钮，查看打开的结果，如图 5.51 所示。

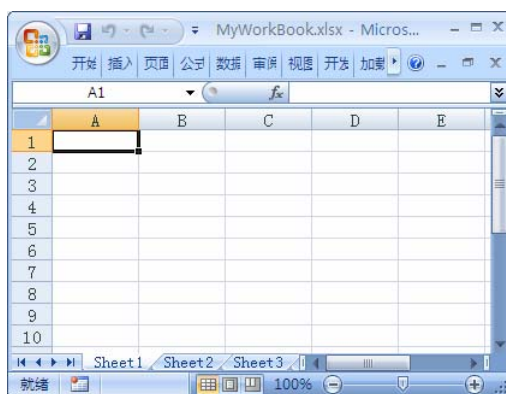


图 5.51 打开的工作簿

6. 程序分析

在上面的代码中，使用 `Dir` 函数判断指定文件是否存在。使用 `Dir` 函数还可以方便的获取指定路径下的文件夹或所有文件名。

案例 105 判断打开的工作簿个数

1. 功能说明

在用户使用 Excel 处理数据时，经常需要在工作簿集合中遍历，然后依次执行操作。要完成遍历的操作，用户需要首先判断打开的工作簿个数。

2. 语法说明

在 Excel VBA 中，`Workbooks.Count` 属性的功能是返回集合中对象的数量。其语法表达式如下：

`表达式.Count`

其中的表达式表示 `Workbooks` 对象的变量。

3. 案例说明

本例的主要功能是通过代码判断打开的工作簿个数，其中，默认打开的工作簿如图 5.52 所示。

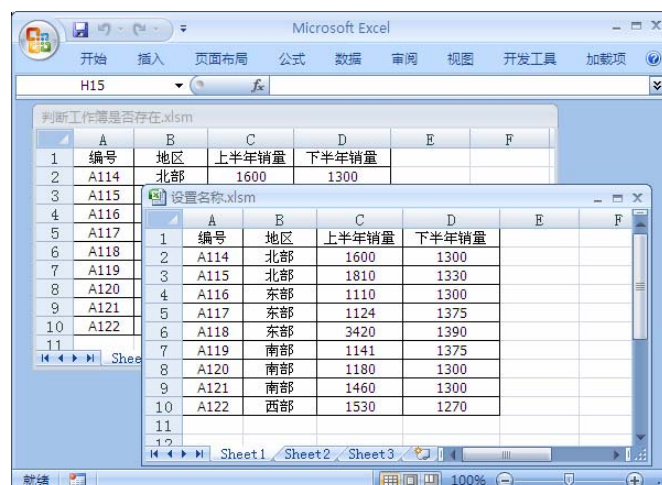


图 5.52 打开的工作簿

4. 编写代码

判断工作簿个数的代码如下：

```
Sub CountNumber()  
Dim Intnumber As Integer  
Intnumber = Workbooks.Count  
  
MsgBox "打开的工作簿有" & Intnumber & "个"  
End Sub
```

5. 运行结果

运行程序代码，查看打开工作簿的个数，如图 5.53 所示。

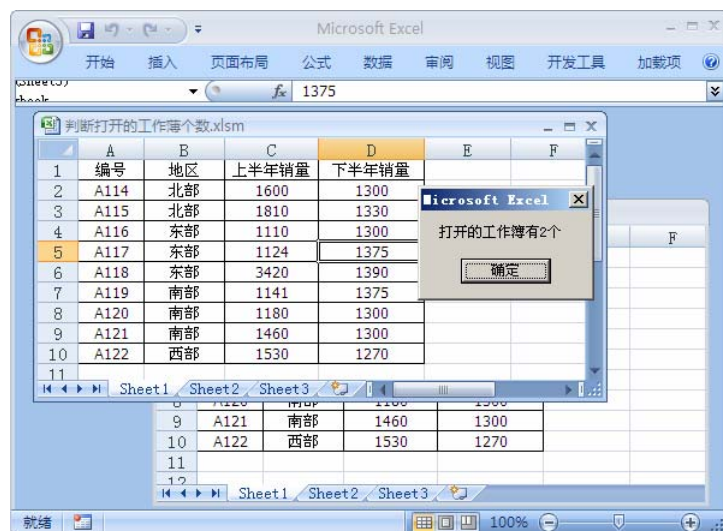


图 5.53 打开的工作簿个数

6. 程序分析

很显然，`Workbooks.Count` 属性返回的是已经打开的工作簿的个数，而不是所有已经存在的工作簿的个数。

案例 106 备份工作簿

1. 功能说明

备份工作簿是经常用到的操作，本例的代码可对当前工作簿进行备份，备份文件与工作簿名称相同，但后缀名为“.bak”，且该备份与当前工作簿在同一文件夹中。

2. 语法说明

在 Excel VBA 中，备份工作簿涉及的技术包括下面的内容：

(1) `Path` 属性：`Workbook` 对象的 `Path` 属性，返回工作簿对象的完整路径。如果其值为空，表示该工作簿为新建工作簿，还未进行保存。

(2) `Dialogs`：`Application` 对象的 `Dialogs` 属性返回一个 `Dialogs` 集合，该集合包含了 Excel 的内置对话框，通过设置集合的常量可调用这些内置对话框。在本例代码中，用来调用“另存为”对话框。

(3) `FullName` 属性：通过 `Workbook` 对象的 `FullName` 属性，可获取工作簿的名称，该名称包括其磁盘路径。

(4) `InStrRev` 函数：该函数返回一个字符串在另一个字符串中出现的位置，从字符串的末尾算起。其语法格式如下：

```
InStrRev(stringcheck, stringmatch[, start[, compare]])
```

函数中各参数的含义如下：

- **Stringcheck**：要执行搜索的字符串表达式（被查找字符串）。
- **Stringmatch**：要搜索的字符串表达式（查找字符串）。
- **Start**：设置每次搜索的开始位置。如果忽略，则使用 -1，它表示从上一个字符位置开始搜索。如果 `start` 包含 `Null`，则产生一个错误。
- **Compare**：指出在判断子字符串时所使用的比较方法。如果忽略，则执行二进制比较。

提示：若需要从字符串首部向后查找子串，可使用 `Instr` 函数。

3. 案例说明

某公司统计了部分员工的销量数据以及员工所在的地区，为了能够进一步分析对应的数据，用户需要将其数据进行备份，原始数据如图 5.54 所示。

	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	
11					

图 5.54 原始数据

4. 编写代码

备份工作簿的 VBA 代码如下：

```
Sub CopyWorkBoChecks()
    Dim wb As Workbook
    Dim FileName As String
    Dim i As Integer
    Dim Check As Boolean

    Set wb = ActiveWorkbook
    If wb.Path = "" Then
        MsgBox "请先保存工作簿"
        Exit Sub
    End If
    FileName = wb.FullName

    i = InStrRev(FileName, ".")
    If i > 0 Then FileName = Left(FileName, i - 1)
    FileName = FileName & "工作簿的备份文件" & ".bak"

    Check = False
    On Error GoTo Finish
    With wb
        .Save
        .SaveCopyAs FileName
        Check = True
    End With

Finish:
    Set wb = Nothing
    If Not Check Then
```

```
MsgBox "备份工作簿失败!", vbExclamation, ThisWorkboCheck.Name  
End If  
  
End Sub
```

5. 运行结果

运行程序代码，查看备份的工作簿文件，如图 5.55 所示。

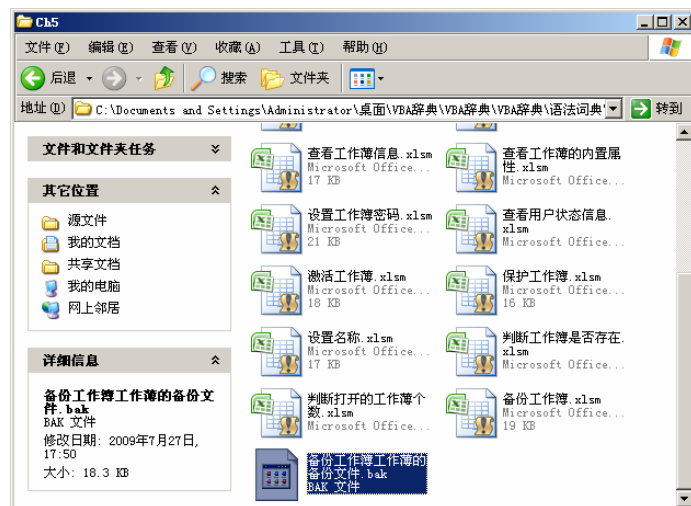


图 5.55 查看备份的文件

6. 程序分析

在上面的程序中，首先获取对当前工作簿的引用，然后获取工作簿的全名，使用 `InStrRev` 函数查找工作簿全名是否有扩展名，若有扩展名，则截取文件名的前面部分（不含扩展），再将文件名后面加上“.bak”，形成备份文件的名称，最后保存当前工作簿，并使用备份文件名另存文件，得到备份文件。

5.2 使用工作簿事件

和其他对象类似，在 Excel VBA 中，工作簿对象也有多个常见的事件。使用这些常见的事件，用户可以对工作簿的各种操作进行自定义。本小节中，将详细讲解各种常见的工作簿事件。

案例 107 强制保存工作簿

1. 功能说明

当用户需要对用户关闭工作簿之前的操作进行设置的时候，可以使用 `BeforeClose` 事件。在该事件中，用户可以设置关闭之前的所有操作。

2. 语法说明

在 Excel 中，在用户关闭 Excel 工作簿之前，将产生 `BeforeClose` 事件。如果工作簿已经更改过，则本事件在弹出对话框之前产生。该事件过程的结构如下所示：

```
Private Sub Workbook_BeforeClose(Cancel As Boolean)

End Sub
```

参数 `Cancel` 为一个逻辑变量，当事件发生时其值为 `False`。如果在事件过程中将此参数设置为 `True`，则停止关闭操作，返回到工作簿中继续操作（工作簿保持打开状态）。

一般情况下，只应该在 `BeforeClose` 事件中加上以下功能，而不应将对系统的设置或者恢复代码放在此处。

（1）不显示是否保存修改对话框而保存工作簿的任何更改。

（2）放弃保存工作簿的任何更改，而直接退出，代码如下：

```
Private Sub Workbook_BeforeClose(Cancel as Boolean)
    Me.Saved = True
End Sub
```

（3）在程序中设置标志变量，控制用户直接按窗口上的“关闭”按钮退出系统，只有通过代码设置标志变量为对应值时才允许退出系统，具体代码如下：

```
Private Sub Workbook_BeforeClose(Cancel as Boolean)
    If bFlag=False then Cancel=True
End Sub
```

3. 案例说明

在 Excel 工作簿中进行修改后，如果没有保存就关闭工作簿，Excel 将弹出如图 5.56 所示的提示窗口，让用户选择是否保存对工作簿的更改，该对话框共有三个选项：

- 单击“是”按钮，保存更改并退出工作簿；
- 单击“否”按钮，不保存并退出工作簿；
- 单击“取消”按钮，不保存工作簿，并返回工作簿继续操作。



图 5.56 保存工作簿提示对话框

本例的主要功能则是，当用户关闭工作簿时，不希望出现上面的对话框，系统直接保存修改，然后关闭工作簿。其中原始数据如图 5.57 所示。

	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	
11					

图 5.57 原始数据

4. 编写代码

本例在 ThisWorkbook 对象的 BeforeClose 事件代码中编写如下 VBA 代码：

```
Private Sub Workbook_BeforeClose(Cancel as Boolean)
    If Me.Saved = False Then Me.Save
End Sub
```

5. 运行结果

在本例中，当用户关闭工作簿时，Excel 不会显示提示保存的对话框，而是直接保存后，直接关闭工作簿。

6. 程序分析

在本例的代码中，Workbook 对象的 Saved 属性用来判断工作簿是否保存。如果指定工作簿从上次保存至今未发生过更改，则该属性值为 True。

如果要关闭某个已更改的工作簿，但又不想保存它或者不想出现保存提示，则可将此属性设为 True。

案例 108 限制打印

1. 功能说明

打印是对 Excel 工作表的特殊操作，VBA 中提供了 BeforePrint 事件来设置打印前的操作。在本小节中，将结合具体的例子来演示如何限制打印。

2. 语法说明

在 Excel VBA 中，通过 Workbook 对象的 BeforePrint 事件可控制工作簿的打印，在打

印指定工作簿之前，将产生 BeforePrint 事件。该事件过程的结构如下所示：

```
Private Sub Workbook_BeforePrint(Cancel As Boolean)

End Sub
```

参数 Cancel 为逻辑型，当事件发生时其值为 False。如果该事件过程将此参数设置为 True，则该过程完成后将不打印工作簿

3. 案例说明

某公司统计了部分员工的销量数据以及员工所在的地区，为了能够进一步分析对应的数据，用户需要限制用户打印该工作簿，原始数据如图 5.58 所示。



	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	

图 5.58 原始数据

4. 编写代码

本例在 Workbook 事件中的 BeforePrint 事件过程中编写如下代码即可：

```
Private Sub Workbook_BeforePrint(Cancel As Boolean)
    MsgBox "不允许打印！", vbCritical + vbOKOnly
    Cancel = True
End Sub
```

5. 运行结果

当用户选择“Office 按钮”菜单中选择“打印”命令，如图 5.59 所示，可打印工作簿中当前工作表或工作簿中所有工作表。在本例中，当用户选择该命令后，将弹出如图 5.60 所示的对话框，提示用户不能打印工作簿中的内容。

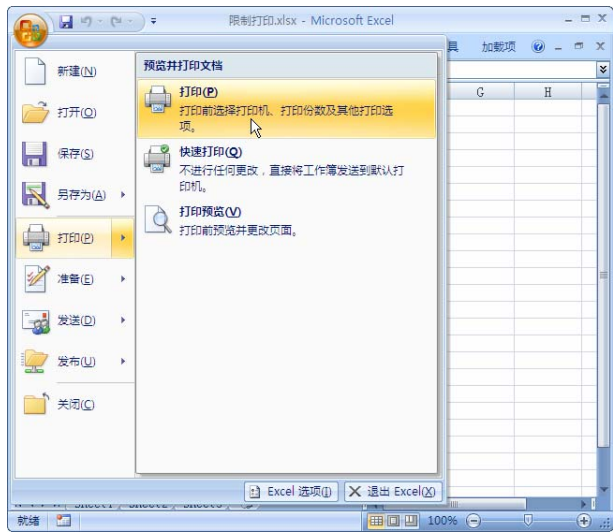


图 5.59 选择打印工作簿



图 5.60 限制打印

6. 程序分析

在实际的程序开发中，当工作簿的内容比较隐秘的时候，需要限制用户随意打印工作簿的内容。

案例 109 限制保存

1. 功能说明

当用户对工作簿进行编辑的时候，在没有进行保存时，所有的修改都将失效。当不希望用户对工作簿进行编辑的时候，可以限制保存。

2. 语法说明

在 Excel VBA 中，在保存工作簿之前将产生 BeforeSave 事件，其事件过程结构如下：

```
Private Sub Workbook_BeforeSave(ByVal SaveAsUI As Boolean, Cancel As Boolean)

End Sub
```

该事件过程有两个参数，其含义如下：

- SaveAsUI：如果将显示“另存为”对话框，则为 True。
- Cancel：当事件发生时该参数的值为 False。如果该事件过程将此参数设置为 True，则该过程完成后将不保存工作簿。

针对该事件的两个参数，该事件一般可以完成以下功能。

- 禁止文件另存，但可对原文件的修改进行保存；
- 禁止保存修改，使保存与另存为功能都失效。

在禁止保存修改时应配合在 BeforeClose 事件中写入代码才能达到完美效果，否则退

出 Excel 时将弹出保存提示对话框。

```
Private Sub Workbook_BeforeClose(Cancel As Boolean)
    Me.Saved=True
End Sub
```

3. 案例说明

某公司统计了部分员工的销量数据以及员工所在的地区,为了能够进一步分析对应的数据,用户需要限制用户保存该工作簿,原始数据如图 5.61 所示。



	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	
11					

图 5.61 限制保存

4. 编写代码

在工作簿的 BeforeSave 事件中编写以下 VBA 代码,禁止 Excel 的“保存”功能。

```
Private Sub Workbook_BeforeSave(ByVal SaveAsUI As Boolean, Cancel As Boolean)

    MsgBox "不允许保存修改!", vbCritical + vbOKOnly
    Cancel = True
End Sub
```

5. 运行结果

完成前面的代码后,单击“保存”按钮,查看保存的结果,如图 5.62 所示。

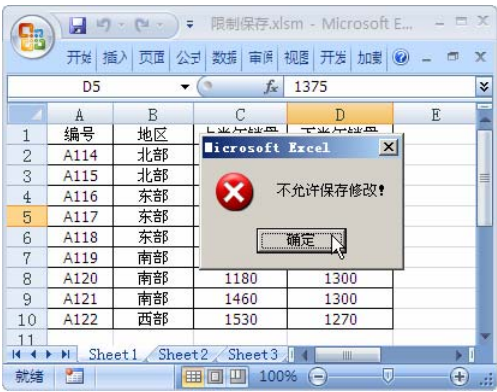


图 5.62 程序运行的结果

6. 程序分析

若要禁止“另存为”功能，可使用以下代码：

```
Private Sub Workbook_BeforeSave(ByVal SaveAsUI As Boolean, Cancel As Boolean)
    If SaveAsUI = True Then Cancel = True           '禁止另存为
    MsgBox "不允许保存修改！", vbCritical + vbOKOnly
End Sub
```

案例 110 显示工作表的插入时间

1. 功能说明

工作表是属于工作簿的对象，用户可以对用户工作表对象的操作编写代码。

2. 语法说明

本例涉及的主要技术是 `Workbook.NewSheet` 事件，当在工作簿中新建工作表时发生此事件。其语法表达式如下：

```
表达式.NewSheet(Sh)
```

表达式表示 `Workbook` 对象的变量。同时，为了将新插入的工作表添加在最后，需要使用 `WorkSheet` 的 `Move` 方法，这个在前面技巧中已经介绍过。

3. 案例说明

某公司统计了部分员工的销量数据以及员工所在的地区，为了能够进一步分析对应的数据，需要添加新的工作表。用户需要编写 `VBA` 代码，显示工作表的插入时间，原始数据如图 5.63 所示。



	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	
11					

图 5.63 显示工作表的插入时间

4. 编写代码

在工作簿的 BeforeSave 事件中编写以下 VBA 代码，禁止 Excel 的“保存”功能。

```
Private Sub Workbook_NewSheet(ByVal Sh As Object)
    Sh.Move After:=Sheets(Sheets.Count)
    If TypeName(Sh) = "Worksheet" Then
        Range("A1").Value = "工作表的插入时间是 " & Now()
    End If
End Sub
```

5. 运行结果

选择工作表，然后在标签处右击，在弹出的快捷菜单中选择“插入”选项，如图 5.64 所示。

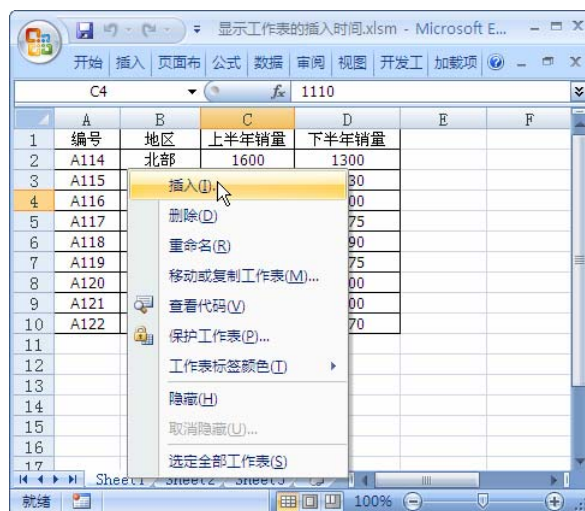


图 5.64 插入新的工作表

Excel 会弹出“插入”对话框，在对话框中选择“工作表”选项，如图 5.65 所示。

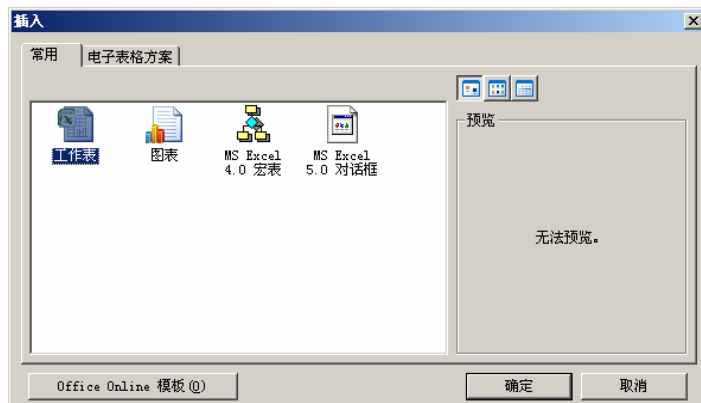


图 5.65 选择插入的是工作表

单击“确定”按钮后，系统显示的结果如图 5.66 所示。

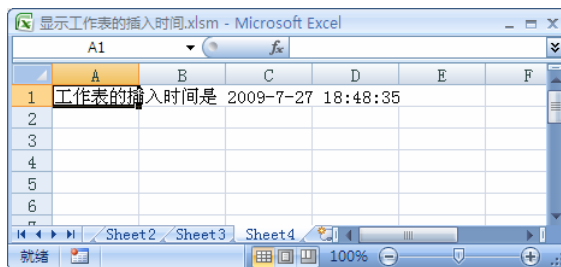


图 5.66 显示插入工作表的时间

6. 程序分析

在上面的案例中，首先使用 `Sheets.Count` 统计工作簿中的工作表个数，然后将新插入的工作表插入到最后一个工作表的后面。

案例 111 设置工作簿的使用时间

1. 功能说明

当用户打开工作簿的时候，会触发 `Open` 事件，这个事件类似初始化的内容。用户可以在该事件中编写对应的代码，实现对工作簿的定制。

2. 语法说明

当用户打开工作簿时，`Workbook` 对象将产生 `Open` 事件。`Workbook_Open` 事件只在工

作簿打开的时候产生，在下次打开之前不再发生此事件。在此事件中可以写入一些只需执行一次的代码，对系统设置进行修改的代码不应该编写在这个事件过程中，而应写入 `Workbook_Activate` 事件中，`Workbook_Activate` 事件紧随在 `Workbook_Open` 事件后面发生。

在 Excel 中，当用户使用 `Application` 对象的 `Quit` 方法，将退出 Excel。使用此方法时，如果未保存的工作簿处于打开状态，则 Excel 将显示一个对话框，询问是否要保存所作更改。要防止发生这种情况，在使用 `Quit` 方法前保存所有工作簿或将 `DisplayAlerts` 属性设置为 `False`。如果该属性为 `False`，则在 Excel 退出时，即使有未保存的工作簿，也不会显示对话框，而且不保存就退出。

3. 案例说明

某公司统计了部分员工的销量数据以及员工所在的地区，为了能够进一步分析对应的数据，设置工作簿的使用说明，原始数据如图 5.67 所示。



	A	B	C	D	E
1	编号	地区	上半年销量	下半年销量	
2	A114	北部	1600	1300	
3	A115	北部	1810	1330	
4	A116	东部	1110	1300	
5	A117	东部	1124	1375	
6	A118	东部	3420	1390	
7	A119	南部	1141	1375	
8	A120	南部	1180	1300	
9	A121	南部	1460	1300	
10	A122	西部	1530	1270	
11					

图 5.67 原始数据

4. 编写代码

在工作簿的 `Open` 事件过程中编写代码实例本例的功能，具体 VBA 代码如下：

```
Private Sub Workbook_Open()  
  
    If Now >= #7/25/2009# Then  
        MsgBox "工作簿只能在 2009 年 7 月 25 日之前打开！"  
        Application.Quit  
    End If  
  
End Sub
```

5. 运行结果

保存程序代码，然后重新打开工作簿，查看提示信息，如图 5.68 所示。

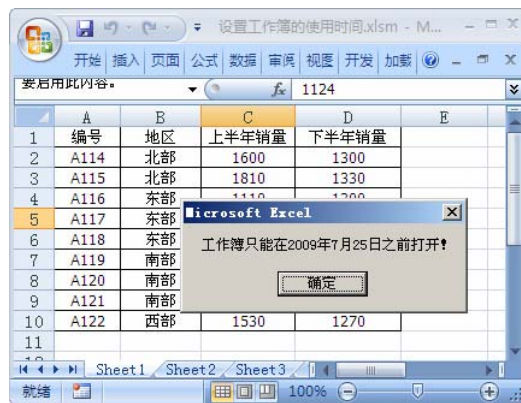


图 5.68 提示信息

6. 程序分析

本案例十分的简单，只是设置了用户使用工作簿的时间。当时间超过限制的时候，将在打开工作簿的时候，退出 Excel。