第零章 绪 论

仿佛昨天恐龙还在统治着这个世界，这种巨大的生物实在令人惊奇，可惜的是它们适应不了环境。恐龙虽然如此强大有力，但是却由于不够灵活，而终于在不断变化的世界中遭到淘汰。

昨晚在看了"侏罗纪公园"后，我整夜不能入睡，我并不是在思考这些巨大的爬行动物的事，而是在考虑某些"古老"的软件包，这些很快就要淘汰的"怪兽"是计算机时代而非中生代的恐龙。

今天，高级应用程序的新时代已经到来，由 Microsoft 的 Excel 5.0 和 Project 4.0 为开始的一些更快、更敏捷、更灵活的应用程序正在取代那些原始而古老的僵硬的软件包("恐龙")，这些高级应用程序的最重要的特点便是它们具有更强的灵活性。

他们成功的秘诀是什么呢？毕竟，这些应用程序和以往的程序一样，十分庞大并且同样具有数百种功能。这两者的不同点是：新的高级应用程序有着变色龙一样的变化能力以迎合用户的需求，以及和其他高级程序协同工作溶为一体的能力。

"Visual Basic for Applications"(缩写为 VBA)的激动人心的新功能，将使上述应用程序的能力变为可能。

0.1 定 制

VBA 允许我们赋予自己喜欢的程序以新的功能并且能使它们完成一些连编程者自己也想不到的任务。没有自己喜欢的功能吗？那就补充它！是已有的功能不好用吗？那么换掉它！为每月都做着一系列相同而又复杂的事而感到疲倦吗？那么就教会程序如何去做这些工作，让它去做！

0.2 自动化

"Visual Basic for Application"不仅具有上述功能，它还能让你使共同的任务自动完成，VBA 能使 Excel 和 Project 成为你的私人助手——通过接管一些重复、耗时的任务，使你腾出时间去做更重要、更有创造性的工作。

0.3 协作化

如果这样还不够的话，那么 VBA 还可以提供对像 Word for Windows 6.0 和 PowerPoint 3.0 这样的高级应用程序的控制能力。实际上，这些高级程序可以是任何一种遵守新的 OLE2 自动化标准的 Windows 应用程序。现在，复杂的、多用途的任务能在 VBA 控制下自动完成。

0.4 本书读者对象

如果你有 Microsoft Excel 5.0 或 Microsoft Project 4.0 的软件拷贝而且希望充分使用它们的功能，那么你需要本书。你需要从头至尾地阅读本书，尤其是当你属于以下所列出情况中的任何一种时：

- ◆ 想让应用程序更易于使用。
- ◆ 想让应用程序的功能更强。
- ◆ 标准化的应用程序满足不了你的特殊要求。
- ◆ 经常做一些重复并且复杂的工作，这样的工作可以包括一个或更多的应用程序。
- ◆ 一直想学习一些有关计算机怎样工作的知识。
- ◆ 从来不想了解计算机是怎样工作的，而只是想简化你的工作更简单。
- ◆ 老板说你必须学习有关 VBA 的知识。

你不必知道怎样给计算机编程序。实际上，有关编程方面的知识懂得越少越好！本书假定你从来没有编过程序。如果你曾经编过，那么最好把所有的经验全部忘掉，然后重新开始学习。

另一方面，你需要对 Windows 有些熟悉：能够选择菜单，单击和双击鼠标等。如果你还不会这些的话，有必要在阅读本书之前先看一看："Windows 3.1：The Visual learning Guide (初版)"。在开始学习之前，先学会使用 Excel 或 Project。本书没有花很多时间去讨论这些应用程序的基本特征，只是讨论如何用 VBA 去控制和增强它们的功能

0.5 综 述

本书是对"Visual Basic for Applications"的程序语言的介绍。如果你觉得这些内容听起来很可怕或者看起来很乱，别担心，在读完本书后，你将能够像专业人员一样编写 VBA 的程序。

0.5.1 第一部分：什么是程序

程序，我们使用它、喜欢它、憎恨它。但它们是什么呢？它们很神奇吗？你必须很紧张地去弄明白它们吗？本书的这一部分将告诉你所有这一切。

0.5.2 第二部分：编写自己的程序

在知道什么是程序后，你会渴望去写一个 VBA 的程序！在这一部分中，我们将告诉你如何一步一步地去做，你将准确地看到程序是如何构成的和怎样去编写一个自己的程序，你还将学到怎样用 VBA 去修改 Excel 和 Project，以及怎样去加入一些编程者所未曾开发的功能。

0.5.3 第三部分：遥控

在掌握了基本的程序结构之后，就是学习一些高级技巧和窍门的时候了。最古老和最省

时的技巧和窍门是"把难的工作留给别人去做"!在这一部分中,你将学到怎样去选择更新的Windows应用程序的先进功能(那些支持OLE2自动化的)并且让它们为你工作。

0.6 本书约定

本书是在讨论一些一般的书中所没有的东西,因为本书是计算机方面的书籍,所以我们需要一种方法来表示出现在计算机屏幕里的文件信息和你在编程时所输入的东西等。为了使所有的东西更加直观,下面是关于表达方式的一些约定:

- 在计算机键盘上单独的键是用按键的小图片表示的(例如Enter,F1或A)
- 必须同时按下的键是用一个加号把两个按键的符号连结起来的方式来表示的,例如Shift+F1表示"按住Shift键,然后按F1键,并在松开Shift键前松开F1键"
- 在程序里出现的单词和词组是用以下的字体显示出现的:

```
X=X+1 'Add 1 to X
```

- 在创建一个程序时所看到的以黑体字出现的单词,必须按其出现的形式准确地输入。
- 在程序里以斜体字出现的单词是一些可以更改或者是一些将在后文中对其进行详细解释的单词。

EXCEL 这个图标表示在该图标旁边的操作方法仅适用于VBA的Excel版本

PROJECT 这个图标表示在该图标旁边的操作方法仅适用于VBA的MS Project版本

第一章 程 序

“计算机程序”、“宏”、“软件”、“应用程序”，这些词，你可能已经听到过几千次。那么它们究竟是什么意思呢？这些改变了我们的生活乃至于我们的世界的东西究竟是什么呢？下面就让我们对这些概念做进一步的了解。

1.1 什么是程序

早在计算机出现以前就有了程序，或许在1786年莫扎特的“费加罗的婚礼”首次演出前就把其“程序”(也就是节目安排)印在了休息厅里。毫无疑问，在1830年以前朱利叶斯·凯萨便制定了能让他管理好参议员的立法程序。不过这些例子也许太古老，无法回答我们的问题。准确地说，程序到底是什么？

简而言之，一个程序就是一个计划；它是为完成一个特定的目标而罗列出的一组行为。你今天有一些要做的事情吗？如果有，那么就应当制定一个计划，这便是程序。上周你向朋友讲述的如何到达你家的路线也可算是一种程序，如果他按照你告诉的程序去做，那么他会在周末准时到你家来参加舞会。

其实在日常生活中有着各种各样的程序，比如太空程序(指导人们如何到达月球的计划)，保养健康的程序(保持人们身心健康的计划)，以及投资程序(指导人们赚钱的计划)，甚至还有节食的程序(使节食者变得苗条的计划)和锻炼身体的程序(让锻炼者能有健壮的体格)。有了这么多的程序，那么还有没有其他的程序？

当然有，这就是我们还没有提到的一种更好的程序——计算机程序。

1.2 计算机程序

设想一下当世界上第一台电子计算机刚刚组装起来，焦急的科学家和工程师们身着白色的工作服，在计算机旁忙得团团转。一位科学家接上开关，为这台大如一间房子的机器提供上千瓦的电，随着电流的通过，这个身价百万美元的庞然大物开始呻吟起来，最后，它终于有了足够的能量。在场的每一个人都屏息观注着这个巨大的电脑……而最终却什么也做不了。

它只是一堆过热的电线，加上一些真空管和显示灯！的确，它“能”在眨眼的工夫完成两数相加。它也能在一瞬间做完减法、乘法和除法。但事实上它又不能。这台机器发明不久，科学家们便发现它缺少了一些东西。

也许读者已经猜到，这台机器所缺少的便是程序，也就是一个计划需要有人告诉计算机把哪些数相加，哪些数相乘等等，还要告诉计算机是先做加法再做乘法或是反过来。这台计算机所需要的便是一个所要执行任务的清单——一序列所要执行的指令。也就是说它所需要的便是计算机程序。

事实上，很难想象一台没有计算机程序的计算机。一个计算机程序就是一个计划，而计算机便是能执行这一计划的复杂的电子器件。计算机与程序两者之间无论缺少谁，都会失去意义。两者的统一改变了我们的世界。

现代的计算机程序往往复杂得令人难以想象，像 Microsoft Excel 或 Project 这样的大型 Windows 程序可以容纳一百多万条独立的步骤或指令。编写这样大的程序需要数百人努力工作好几年。

但不是所有的程序都是这样，许多有用的程序，以及本书所涉及的绝大多数程序都往往只有三、四十行，个别的会更长一些，但更多的程序则更加短小。读者很快会发现，只要你方法得当，寥寥数语便可完成很多的工作。

1.3 语 言

什么是语言？可以这样给它下定义：语言是表达思想的途径。

如果某种语言是书面语言，那么每一个字或符号都代表某种信息，一些简单的思想（字）按照一定的语法规则可以组成更为复杂的思想（句子或段落）。

如果两个人都知道语法规则，他们便可相互交流，其中一个人用公共的语言写出她的思想，而另一个人阅读它并把它翻译成原先的思想。

1.3.1 人类语言

众所周知，世界上有许多种人类语言。每种语言都有一种不同的字词搭配以及把这些词组织起来的规则（语法），并且每句话都有注音和轻音。假设你想描述雪，那么千万不要用波利尼西亚那边的语言，他们根本没有“雪”这个词，而使用因纽特的语言则是一个很好的选择，那里的阿拉斯加人和北部加拿大人对这一事物是非常熟悉的。

除了特定地区和国家的语言之外，还有一些专业性的“语言”。今天我们有医学方面的专业语言，这种语言具有其独一无二的词汇。假如你参加了物理、经济或是心理学方面的高级课程，你便能听懂这方面的专家所说的本学科的专业术语。远在此之前，海员、农夫和士兵们也都有他们各自的语言。

每一种语言都满足了使用它的人们的需要，让这些人之间能相互交流。

1.3.2 计算机语言

难道语言仅仅是为人类服务的吗？也许从前是这样的，但现在不是了，计算机需要的交流绝不比人类的需要少。

正如人类的语言，计算机语言可以反映出计算机的思想、记忆和它们所想说的东西。由于计算机是用来处理数字的机器，因此它们进行的每一个思维对象也是数字，数字不但是它们思考的对象，而且还是它们进行思维的方式，正如人类用语言、图像、声音来进行思维一样。

为了能了解人类语言和计算机语言之间的基本区别，先让我们来看一个例子：你在训练一个招待员如何数清来看电影的人数，你告诉他每当有一个人进入剧院，就“在总数上加1”，这个语言表达方式他肯定是能懂的。

假设你让计算机做相同的事情，那么又应该怎样让它知道“在总数上加 1”呢？很简单，只要说“10001011 01000110 11111100 00000101 00000001 00000000 10001001 01000110 11111100”。

多么大的区别呀！并且对于像你和我这样的人来讲，如何告诉计算机去做什么事情必然成为一大难题，既然有着如此大的语言差别，那么人们究竟如何与计算机进行通信呢？

1.3.3 程序语言

除非你在语言方面的天赋远远的比我强，否则想通过熟练的掌握机器语言来解决这一问题是办不到的。当然也不可能让计算机学会英语，它们还没有这么聪明，我们所需要的是一种新的语言，一种人和计算机都能懂的语言——为能进行高效通信而设计的计算机程序。

我们所需要的是一种和英语比较接近，人们不用花太大的力气（并且在一本好书的帮助下），便能学会的语言。这种语言应该言简意赅，并能够清楚地翻译成计算机的自然语言（按特定目标编制，计算机所能读懂的语言），我们所需要的便是计算机程序语言。也就是所说的“Visual Basic for Applications”。

“Visual Basic for Applications”（缩写为 VBA）是一种新的为那些使用 Microsoft Excel 5.0 或 Microsoft Project 4.0 的用户（很快也会为使用 Word for Windows 之类的 windows 应用软件的用户）而特别设计的语言。它使你不必去记那些神秘的计算机机器语言，而且能使你用更加自然的方式来表达计划。

VBA 除了能让你命令计算机做传统的加、减、乘、除运算之外，还能让用户以遥控的方式来控制其他的程序运行，并且也能让你通过给程序添加你所喜爱的功能或改变程序内部某些你所不喜欢的功能来改变程序。

本书剩余部分将介绍如何实现上述奇迹般的功能，将学习一些 VBA 中的专用词汇和语法。你还将学会编写计算机程序这门“艺术”。

1.4 宏、软件和应用程序

本章已讲了许多有关计算机程序的问题。但在本章的第一段所提到的另外三个术语“宏”、“软件”和“应用程序”是什么东西呢？简单的回答是：他们仅仅是计算机程序的另一个名字，但每一个又有其自己的发展历史，并且他们和程序之间还有一些细小的差别。

1.4.1 宏

你会经常听到 VBA 被称之为一种“宏语言”。用它来写的程序叫作“宏（macro）”。macro 来自希腊语的单词“makros”，意思是“宏大的”，它能成为一个计算机术语是因为现代程序语言的每一条语句通常要在计算机的自然语言里翻译成好几条语句。

后来宏语言（macro language）这一术语被用来专指为控制某一特殊应用程序而设计的特殊语言。宏便是用一种宏语言编写的任何一个程序。

在这方面来说，VBA 可以被称为一种宏语言，但我们又宁愿说它不是，为什么这样说呢？VBA 不仅仅是一种宏语言，如果这样称呼它，便抹杀了它的真正的功能和潜力。

1.4.2 软件

软件，在某种程度上可以指一些计算机程序，这个术语发明在计算机的早期，用来从计算机本身，即硬件中区别计算机程序，这样称呼它是很合适的。因为程序看起来是软性的。因为它比齿轮和晶体管这样的硬件设备更容易被改变。

1.4.3 应用程序

最后，把一种计算机程序称为应用程序，是为强调编写一个计算机程序的首要用途。

计算机程序用来发挥计算机所具有的功能。每一个程序都描述了一种能够完成某一特定任务的方法。从这种观点上看，程序可作为一个问题的解决方案。

你将会注意到“Visual Basic for Applications”的编者更喜欢最后这一条术语——“应用程序”(Application)。他们想让你把 VBA 想象为一个问题的解决方案。此外，读者还可这样理解 VBA：它能改善这些问题的解决方案，使它们更好地完成我们的工作或者完成我们以前不能完成的工作。本书的目的在于要让读者学会并使用 VBA 来做到这一点。

第二章　编写程序

与英语不一样，VBA 是一种专用的书面语言（虽然现在是，但随着计算机的发展，谁能预测 VBA 今后是不是?）。因此在学习 VBA 语言之前（在第三章开始学习 VBA），让我们先复习一下打字和编辑的技巧。因为在使用 VBA 编程时，要保证所写的程序准确无误地存入计算机。

2.1　创建一个新模块

当你要输入一本伟大的美国小说前，你该如何开始呢？很自然地，你应先把一张新的纸卷进手动打字机中，但这种作法岂不是跟不上时代潮流的发展吗？你完全可以启动自己喜欢的字处理软件，从 File（文件）菜单中选择 New（建新文件），于是便可在全屏编辑的环境下录入自己的文件了。

或者，你准备了一块位置来记录你文章的小片段。在 VBA 的环境下创建的这种片段被称为模块。模块可以想像为用 VBA 语言写的一组语句。一个模块可以包括一个或多个完整的 VBA 程序，也可以只包括某个程序的一部分。VBA 中的每一种应用程序都有建立新模块的方法。

EXCEL　Microsoft Excel 把模块存储在一个特别工作簿表格内，因此在创建一个模块前，应先打开一个工作簿或创建一个新的工作簿，若要打开一个已经存在的工作簿，只要从 Excel 的 File 菜单中选择 Open（打开）或按 Ctrl＋O。若要建立一个新的工作簿，则可从 Excel 的 File 菜单中选择 New 或按 Ctrl＋N，下图便是 File 的下拉菜单。

补充说明：在下面的例子中，我将使用名叫 TYPING.XLS 的新工作簿，并希望读者也能仿效而不要去改变现存的任何一个文本，这样，任何失败的操作都不会对你的重要文件产生影响。

在屏幕上已有了自己所需的工作簿后，可通过在工作簿中某一表格（SHEETS）的标记上单击鼠标去告诉 Excel 应把新的一页放于何处。当新的模块创建之后，它将立即在所选择的任意一个已存在的表格前放置好。

下面是建立新模块的步骤：从 Excel 的 Insert(插入)菜单中选择 Macro(宏)，然后从子菜单中再选择 Module(模块)，屏幕上将出现：

这样便可得到一个空白的模块表格，如图 2.1 所示。

看上去它似乎不是空的，但事实上它是。图中所示的 Visual Basic 的 Toolbar(工具条)不是这张空白模块表格的一部分，它"悬浮"在表的上面。只要所激活的文本表格为一模块，工具条便显示出来。因为它是"悬浮"在表的上面，因此可置于屏幕的任意位置，甚至可把它放到主应用程序的按钮栏中。现在先把它拖到一边去。关于如何使用工具条将在第三章介绍。

若要打开已存在的 project 文件，只要从 Project 中的 File 菜单选择 Open(也可直接按 Ctrl + O)；若要建立一个新的 project 文件，则从 Project 中的 File 菜单中选择 New(也可直接按 Ctrl + N)。

补充说明：在模块表格创建后，可以通过在表格标记上双击鼠标来改变其名称(比如改变默认的名字 Module1)。

PROJECT Microsoft Project 提供两个存储模块的地方：一是用户个人的 project 文件，二是所谓的"GLOBAL"文件，它是一个具有 GLOBAL.MPT 名字的文件，但不可直接打开或使用它，而 Microsoft Project 将使 Global 文件中的内容成为用户创建的每一个 project 文件中的一部分。因此若要访问 Global 文件，只须打开任意的 project 文件，打开它之后，除了可以得到 project 文件中的信息外，还可以访问 Global 文件中的信息。

补充说明：用 Globe 文件去存储信息——特别是被所有或大部分 project 文件所使用的 VBA 模块的信息。被单独的 project 所使用的 VBA 模块和其他信息，应存储于该 project 文件中。

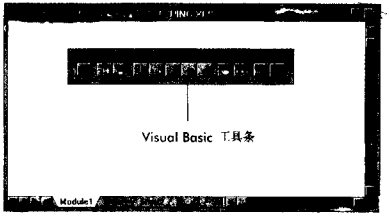


图 2.1 Excel 工作簿中的模块表格

一旦所要的 project 文件出现在屏幕上，为了看一些已包含在该文件中的 VBA 模块并创建新的模块，就应选择 Project 下的 Module Editor(模块编辑器)。很自然地，在 Project 的 View(观看)菜单中做选择便可以改变视窗。由于我们所想要选择的视窗不在 View 本身的菜单上，我们便可从 View 的下拉菜单中选择 More Views 来代替。

选择 More Views 后便出现如图 2.2 中所示的窗口，在这里我们可以选择 Module Editor 的视窗。

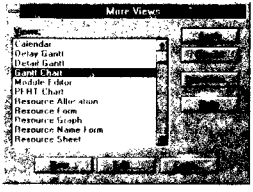


图 2.2 选择项目的 Module Editor 的外观

在显示有 Module Editor 这一行的位置上单击鼠标，然后单击 Apply 按钮，这样便可以通过 Module Editor 的"眼睛"看到自己的 project 文件，并且你还将注意到借助于 Module Editor 可以看到所有在当前 project 文件和 Global 文件中的所有模块。

如果要看一个已存在的模块，你可以使用 Visual Basic 工具条中的 Module 列表框（见图 2.3），只须单击框里右边向下的箭头，Module Editor 将列出所有存在的模块表，用鼠标单击你所选的模块，它将魔术般的出现在屏幕上。

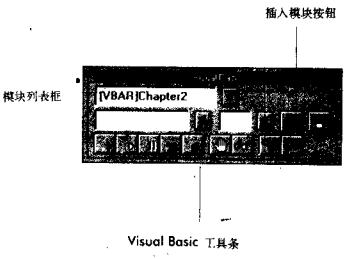


图 2.3 Microsoft Project 中的模块表格

补充说明：如果 Visual Basic 工具条未出现在屏幕上，那就从 Project 的 View 菜单中选择 Toolbars...，在工具条的列表里击一下 Visual Basic 这个词，然后击一下 Show。就像其他 Project 工具条一样，Visual Basic 工具条能在 Project 菜单栏中缩短，或者从缩短的位置用鼠标拖动使你感到它"悬浮"在 Module Editor 的窗口上。

如果要创建一个新的模块，你也可以用 Visual Basic 工具条，方法是在工具条的 Insert Module 按钮上单击鼠标；若你偏爱于菜单操作，也可从 Project 的 Insert 下拉菜单中选择 New Module 来建新的模块，无论你选择哪种方法，都将看到如图 2.4 所示的 Module Editor 的 New Module 窗口，在这里你可以给所创建的模块输入新的名字及描述，如果在窗口中单击 Options 按钮，你还可以决定模块是加在 Global 文件中，还是加在当前的 project 文件上（默认的位置是 Global 文件上）。

补充说明：若要改变当前屏幕上模块的名称，可从 Project 的 Edit 菜单中选 Module，然后从 Module 的下拉子菜单中选择 Rename...，Project 将出现一个用于输入模块的新名字的对话框。另外，从这个子菜单中选择 Delete...，便可删除当前所示的模块。

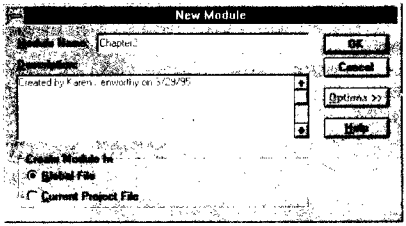
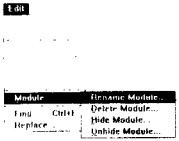


图 2.4 在 Microsoft Project 中建立一个新的模块



2.2 输入文本

在一个新的模块创建后，输入文本便是很容易的。只要在窗口的任意位置(你想让新文本出现的地方)单击鼠标后便可以输入，其工作过程和方式很像 Windows 中的文本编辑器和字处理器，轻松地按下 BackSpace 或 Delete 键便可以更正任何错误的输入，按 Enter 键便开始新的一行。

VBA 是很挑剔的。你输入的任一整行都必须是 VBA 语言里的有效语句。因为你现在还不会 VBA 语法，下面几行文字都是有效的(虽然不一定是有用的)，作为练习，你可以把它们输入到模块中。

```
x = x + 1
The Time Is Now
If Spring Then Fall
```

你也许注意到，在按下回车键后，一些单词变色了或者自动地变成以大写字母开头。这都是正常现象。这实际上是很有用的，你后面将学到这一点，但现在倒不必关心它。

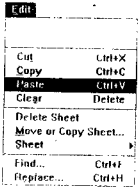
补充说明：试着去构造你自己的句子；输入它，然后按 Enter，如果幸运的话，你会看到一个对话框，这是 VBA 用来告诉你，它不懂你刚才输进去的东西。这些警告是无害的（因为你不过是在练习而已），击一下 OK 按钮，把这个对话框消除。

2.3 更改文本

你可以用所有你喜欢的并熟悉的字处理方法来更改输入的文本。

按 BackSpace 和 Delete 键可以删除一个或多个字符。也可以按 Ctrl + ← 或者 Ctrl + → 来快速左移或右移一个单词。也能用组合键 Ctrl + Home 和 Ctrl + End 移动光标到模块的首端或末端。

在编辑一个模块的文本时，还能够用 Windows 的剪贴板(Clipboard)。通过选择文本(用鼠标单击和拖拽把它放到剪贴板上)然后从应用程序的 Edit 菜单中选择 Copy 或者 Cut。从模块中找到需要粘贴的地方，用鼠标单击，然后从应用程序的 Edit 菜单中选择 Paste，那么粘贴板上的文本将进入模块中。



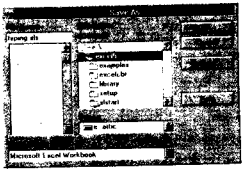
Ins 键用于在 Insert(插入)和 Overstrike(覆盖)方式之间进行切换。在插入方式中，新输入的文本被插入到模块中，原来的文本被向后推开，在覆盖方式中，新输入的文本一个字一个字地代替了原来的文本。

很容易就能分辨哪一种方式在起作用。首先，在 Insert 方式中，光标是很细的一条闪烁的垂直线；在覆盖方式下，光标变成了一个粗一些的闪烁的方块，而且 OVR 这个字将出现在应用程序的状态框内(在主窗口的底部)。如果在插入模式下，它将消失。

2.4 将模块存盘

在创建或更改一个模块后，可以通过存盘将这些改变保存下来，而且只要文本文件或者 project 文件被存盘，模块也跟着存盘。

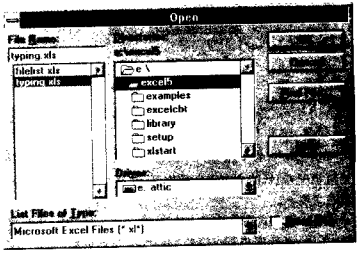
如果要存一个新的工作簿或项目(要把一个原有的工作簿或项目存放在一个新的文件中)，可以从应用程序 File 菜单里选择 Save As 这一项。选择好盘符和路径，还有文件名，然后单击 OK 按钮。



更改模块后，为了对已存在的工作簿或项目作相应的更新，可以从应用程序的 File 菜单里选择 Save，旧版本就会被新版本的文件更新覆盖了。

2.5 从磁盘里调一个模块

检索一个先前存到磁盘里的模块就像调一个磁盘中的工作表或 project 文件一样容易。首先从应用程序的 File 菜单里选择 Open，然后选择文件所在的驱动器名和路径。最后选择文件名并击一下 OK 键。



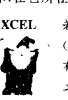
2.6 在文件之间移动和拷贝模块

从一个工作表到另一个或者在一个 project 文件和一个 Global 文件间移动或复制整个模块是很容易做到的。是否能准确地完成这项工作取决于从哪里开始移动或复制和要移动或复制到哪里。

如果要把一个模块从一个 Excel 的工作簿中移到 project 文件中(或做相反的操作)，那么，首先，打开现存模块所在的工作簿或文件，然后将要接收模块的工作簿或文件打开，并在其中按本章前面所叙述的方法创建一个新的模块。

其次，用鼠标或键盘选择所有模块中将要移动的文本，并把它放入 Windows 中剪贴板

里(实现方法为从应用程序中的 Edit 菜单中选择 Cut 或直接按 Ctrl+X)。如果不是移动而是拷贝模块,则从 Edit 菜单中选择 COPY(或按 Ctrl+C 代替),最后,在新模块中单击鼠标,在它所在的应用程序中的 Edit 菜单中选择 Paste(粘贴)(或按 Ctrl+V)。

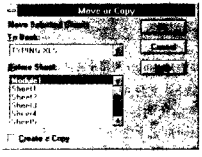


EXCEL 若要从一个 Excel 的工作簿中移动或拷贝一个模块到另一个 Excel 的工作簿中(或到该文本的其他地方),有一个简单的方法和一个复杂的方法,两者都要求所有相应的文本打开(模块所在的文本和要接收模块的文本)。然后通过在模块的工作簿标记处单击鼠标选择好所要移动或拷贝的模块。

下面是容易的部分:只需用鼠标单击要移动的模块的标记。按鼠标左键把模块拖至新的位置。如在鼠标左键被释放前压住 Ctrl 键,模块将被复制而不是被移动。

当你在拖动光标时,模块表格的画面将被改变(在执行拷贝而不是移动时,在图标处会出现一个加号),一个小标志会出现在标记行上面来描述模块的新位置。

另一种不同于点击和拖动鼠标的方法是:在将要移动或拷贝的模块表格的文本标记处单击鼠标,然后从 Excel 的 Edit 菜单中选择 MOVE 或 COPY Sheet,于是便可见看下图所示的对话框。



To Book 列表框将允许你选择将要接收模块的工作簿,有效的选择是那些当前打开的和新建的工作簿。选择 New Book 可创建一全新的文本来存放将要移动或拷贝的模块。如果想用来接收模块的文本不在表列中,则单击 Cancel 按钮,把合适的工作簿打开,然后再重复上述操作。

在选择好合适的目标工作簿后,它的一个列表将出现在 Before Sheet 的列表框中,通过在这些名字中的一个名字上单击鼠标便可选择工作簿内模块的位置。

最后,决定模块是要被拷贝还是被移动,选择 Create a Copy 将给出两个相同的模块表,一个是在原来的位置,另一个在所创建的新位置。如果不选择 Create a Copy 将会使原先的模块表被删除,而只保留新创建的模块表(即完成了移动),做完选择后,单击 OK 按钮,这样一切便都完成了。

PROJECT Project 提供了一称为 Organizer 的工具,该工具能把模块从当前的 project 文件拷贝到 Global 文件中去,或者反过来,它还能把任一文件中的任一个模块重新命名或删除。

Organizer 是通过选择 Project 的 View 菜单中的 More Views... 来激活的，当 More Views 窗口出现后(见图 2.5)，单击 Organizer 按钮，Organizer 将呈现出如图 2.5 所示的窗口(若 Modules 平台未显示，则可通过单击 Modules 边界标记处使它弹出到顶部)。

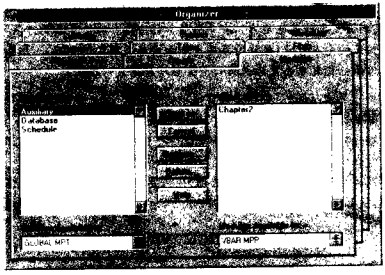


图 2.5 Project 的 Organizer 可用来拷贝、重命令和删除模块

正如你所看到的，存储在 Global 文件中的模块是排列在左边的；当模块存储在当前 project 文件中时，它们却是排列在右边的。从一个文件拷贝一个模块到另一文件时，可在模块的名字上击一下，然后再在 Copy 按钮上击一下。一旦拷贝了，你将有两个模块，它们在不同的文件中，但有相同的内容。如果你只需要其中的一个，可以在拷贝完成后删除原来的模块。

如果要删除一个模块，在两个列表中的任意一个里单击一下它的名字。并击 Delete 按钮，给一个模块改名也一样简单，像刚才删除时那样选择模块的名字，然后击 Rename 按钮，而不是击 Delete 按钮。Organizer 将向你询问模块的新名字，输入新名字后，剩下的工作就全交给它做了。

第三章　着手编程

如果你像我一样，你会迫不及待地想用 Visual Basic 写一个 VBA 程序。如果你已准备打开它并且了解它能做什么，那么就让我们开始吧！你将写你的第一个 VBA 程序了！

在这个过程中，你将了解和学习到第一个 VBA 语句。学习 VBA 就像学习任何新语言一样。有点像婴儿学说话。一开始时，什么都不懂，但过一段时间后，就会从嘈杂的环境中获得少量的词汇。开始时比较慢，然后逐步加快，新掌握的单词组成了短语和句子。然后有一天，你会意识到先前一大串不可理解的废话已变成美妙的可理解的对话。因此，当第一次看到下面的东西时或许不能完全明白，但完全不必为之焦急。如果某些讨论的内容像一个婴儿的乱语一样不知所云，你一定要有耐心。如果一步步地坚持下去的话，你将获得越来越多的信息。最后，一切都豁然开朗。好，坐下来放松一下——我们这就开始。

3.1　MsgBox 语句

先创建一新模块并输入下面这三行语句（如何创建模块并输入，参见第二章）：

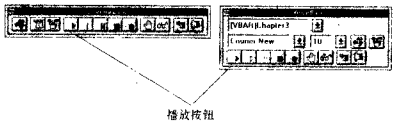
```
Sub MyProgram()
    MsgBox "Hello word"
End Sub
```

补充说明：注意到中间一行是怎样缩进的吗？这些空格并不是必须的，但它能使程序更易读懂。要使一行缩格，可在输入文本前按一次[Tab]键。通过 Project 或 Excel 的 Tools 菜单选择 Option，便可让 VBA 为你自动地在行前留空（见图 3.1）。单击标有 Module General 的标签，然后选中 Auto Indent 复选框。

在学习这些行的意义之前，让我们先看一下它们能干些什么。让我们运行一下你的第一个程序。

首先，在程序的任一地方单击鼠标，这一步告诉 VBA 你想运行哪一个程序。即使你只写了一个程序，VBA 也不会聪明到能想像得出它本身以外的东西。

其次，还记得在前一章时曾把 Visual Basic 的工具条移走这件事吗？现在该重新把它找回来了，它看起来如下面所示：



上面的左起第四个按钮是一个实心的右向小箭头，它就是 Play 按钮。现在单击它，看看