

Visual Basic 程序设计 与数据库应用

V i s u a l B a s i c

黄保和 李更明 编著



厦门大学出版社

XIAMEN UNIVERSITY PRESS

Visual Basic 程序设计 与数据库应用

黄保和 李更明 编著

厦门大学出版社

图书在版编目(CIP)数据

Visual Basic 程序设计与数据库应用/黄保和,李更明编著. —厦门:厦门大学出版社,2009.12
ISBN 978-7-5615-3385-7

I. V… II. ①黄…②李… III. BASIC 语言-程序设计-高等学校-教材 IV. TP312

中国版本图书馆 CIP 数据核字(2009)第 181741 号

厦门大学出版社出版发行

(地址:厦门市软件园二期望海路 39 号 邮编:361008)

<http://www.xmupress.com>

xmup@public.xm.fj.cn

沙县方圆印刷有限公司印刷

2009 年 12 月第 1 版 2009 年 12 月第 1 次印刷

开本:787×1092 1/16 印张:18

字数:457 千字 印数:1~3000 册

定价:28.00 元

如有印装质量问题请与承印厂调换

内容简介

本书以 Visual Basic 6.0 为编程环境,通过大量实用案例由浅入深地介绍了面向对象的程序设计方法,特别是数据库应用技术。本书主要读者是非计算机专业的学生,因此内容的组织从实用角度出发,突出应用,不面面俱到。重点介绍基本控件的使用、基本程序设计结构的使用和数据库编程技术,使读者能够容易入门,学以致用。

本书内容图文并茂、实用性强,适合作为高等院校非计算机专业学生的教材,也适合作为程序设计爱好者自学用书。

前 言

Visual Basic 6.0 是微软公司推出的系列可视化开发工具 Visual Studio 6.0 的组件之一,它具有简单易学、功能强大、软件费用支出低、见效快等特点,为用户开发 Windows 应用程序提供了最迅速和简捷的方法。同时它也提供了访问数据库的强大功能,为用户开发数据库应用程序创造了条件。在众多的新一代编程语言中,Visual Basic 以其独特的优势赢得了广大用户喜爱,成为当前最流行的程序设计语言之一。

本书针对非计算机专业学生的特点,提供了较多的示例及联系实际的演练习题。在体系结构和内容上尽量做到由简及繁、由浅入深、循序渐进、深入浅出、理论与实践相结合,使读者能够快速入门,较好地掌握本书的内容。

在数据库应用方面,本书全面介绍了 Visual Basic 6.0 访问数据库的各种方法和常用的编程技术,提供了大量实际应用的例子,使读者能快速掌握数据库应用和开发技术。本书假定读者对 Access 2003 有一定的了解。

本书作者是多年从事计算机应用基础教学及相关研究的教师,对非计算机专业学生的计算机教学有较深刻的理解和体会,具有丰富的教学经验,充分了解学生本身的计算机知识结构,能较好地把握本书各章节的内容。

本书由黄保和、李更明合作编写,其中黄保和编写前 5 章,李更明编写后 5 章。

本书在编写过程中,参阅了一些专家同行的研究资料,在此表示感谢!由于编者水平有限,书中难免有不足之处,欢迎读者批评指正(huangbh@xmu.edu.cn)。

作 者

2009.9

目 录

前言

第 1 章 概论	(1)
1.1 程序设计概念	(1)
1.1.1 程序设计	(1)
1.1.2 程序设计语言	(3)
1.1.3 程序设计方法	(4)
1.2 Visual Basic 概述	(5)
1.2.1 Visual Basic 的发展	(5)
1.2.2 Visual Basic 的特点	(6)
1.3 Visual Basic 的集成开发环境	(7)
1.3.1 集成环境窗口	(7)
1.3.2 工程的操作	(11)
习题	(12)
第 2 章 简单应用程序设计	(13)
2.1 应用程序的建立过程	(13)
2.1.1 建立可视化的用户界面	(13)
2.1.2 设置对象属性	(14)
2.1.3 编写代码	(15)
2.1.4 程序的测试和调试	(16)
2.1.5 保存文件	(17)
2.2 VB 的相关概念	(17)
2.2.1 对象	(17)
2.2.2 窗体	(19)
2.2.3 控件	(22)
2.3 VB 最常用控件	(23)
2.3.1 标签	(23)
2.3.2 文本框	(24)
2.3.3 命令按钮	(26)
习题	(29)
第 3 章 VB 语言基础	(32)
3.1 VB 代码书写规则	(32)
3.2 数据类型	(33)

3.2.1 数值	(33)
3.2.2 字符串型(String)	(34)
3.2.3 布尔型(Boolean)	(34)
3.2.4 日期型(Date)	(34)
3.2.5 对象型(Object)	(35)
3.2.6 变体型(Variant)	(35)
3.3 常量	(35)
3.3.1 文字常量	(35)
3.3.2 符号常量	(37)
3.4 变量	(37)
3.4.1 定义变量	(37)
3.4.2 静态变量	(39)
3.5 表达式	(40)
3.5.1 算术运表达式	(40)
3.5.2 字符串表达式	(41)
3.5.3 逻辑表达式	(42)
3.5.4 表达式的运算顺序	(43)
3.6 常用内部函数	(43)
3.6.1 数学函数	(44)
3.6.2 字符串函数	(44)
3.6.3 日期时间函数	(45)
3.6.4 转换函数	(46)
3.6.5 Shell 函数	(46)
习题	(47)
第4章 程序控制结构	(50)
4.1 顺序结构程序设计	(50)
4.1.1 赋值语句	(50)
4.1.2 数据输出	(51)
4.1.3 数据输入	(54)
4.1.4 其他语句	(60)
4.2 选择结构程序设计	(62)
4.2.1 单行结构 IF 语句	(62)
4.2.2 块结构 IF 语句	(64)
4.2.3 多分支 If 结构	(69)
4.2.4 Select Case 语句	(71)
4.2.5 条件函数	(74)
4.3 循环结构程序设计	(75)
4.3.1 For—Next 循环	(76)
4.3.2 While—Wend 循环	(80)
4.3.3 Do—Loop 循环	(83)

4.3.4 多重循环	(85)
4.4 程序调试	(90)
4.4.1 错误类型	(90)
4.4.2 调试方法	(92)
习题	(93)
第5章 常用控件	(100)
5.1 复选框、单选按钮和框架	(101)
5.1.1 复选框	(101)
5.1.2 单选按钮	(102)
5.1.3 框架	(103)
5.2 列表框和组合框	(106)
5.2.1 列表框	(106)
5.2.2 组合框	(110)
5.3 滚动条、滑动器和进度条	(113)
5.3.1 滚动条	(114)
5.3.2 滑动器	(115)
5.3.3 进度条	(117)
5.4 图形控件	(119)
5.4.1 图片框	(119)
5.4.2 图像框	(122)
5.4.3 线条和形状	(122)
5.5 计时器	(123)
5.6 事件与焦点	(126)
5.6.1 鼠标事件	(126)
5.6.2 键盘事件	(128)
5.6.3 焦点	(129)
习题	(131)
第6章 数组	(134)
6.1 数组的定义	(134)
6.1.1 静态数组的定义	(134)
6.1.2 动态数组的定义	(136)
6.1.3 数组初始化	(137)
6.2 数组的操作	(138)
6.2.1 数组的引用	(138)
6.2.2 For Each...Next 语句	(139)
6.3 数组应用举例	(140)
6.3.1 一维数组的应用	(140)
6.3.2 二维数组的应用	(145)
6.4 控件数组	(149)
6.4.1 控件数组的创建	(149)

6.4.2 控件数组的应用	(150)
习题	(154)
第7章 过程	(158)
7.1 Sub 过程	(158)
7.1.1 Sub 过程的定义	(158)
7.1.2 Sub 过程的创建	(159)
7.1.3 Sub 过程的调用	(160)
7.2 Function 过程	(162)
7.2.1 Function 过程的定义	(162)
7.2.2 Function 过程的调用	(163)
7.3 事件过程	(165)
7.4 Sub Main 过程	(166)
7.5 参数传递	(167)
7.5.1 按值传递	(167)
7.5.2 按地址传递	(168)
7.5.3 指名传递	(169)
7.5.4 可选参数和可变参数	(170)
7.5.5 数组参数传递	(172)
7.6 变量与过程的作用域	(173)
7.6.1 VB 应用程序结构	(173)
7.6.2 变量的作用域	(174)
7.6.3 过程的作用域	(178)
7.7 过程应用举例	(179)
习题	(182)
第8章 菜单与对话框	(186)
8.1 多重窗体	(186)
8.2 菜单设计	(189)
8.2.1 菜单编辑器	(189)
8.2.2 菜单数组	(192)
8.2.3 弹出式菜单	(193)
8.3 通用对话框	(195)
8.3.1 基本属性与方法	(195)
8.3.2 “打开”对话框	(196)
8.3.3 “另存为”对话框	(198)
8.3.4 “颜色”对话框	(198)
8.3.5 “字体”对话框	(199)
8.3.6 “打印”对话框	(201)
8.3.7 “帮助”对话框	(202)
习题	(202)
第9章 数据库应用	(205)

9.1 数据库概述	(205)
9.1.1 数据库的基本概念	(205)
9.1.2 VB 数据库访问技术	(206)
9.2 可视化数据管理器	(207)
9.2.1 建立数据库	(207)
9.2.2 建立数据表	(208)
9.2.3 建立查询	(209)
9.3 ADO Data 控件	(210)
9.3.1 常用属性、方法和事件	(210)
9.3.2 Recordset 对象	(215)
9.3.3 Field 对象	(217)
9.4 数据绑定控件	(218)
9.4.1 基本概念	(218)
9.4.2 主要属性和使用步骤	(219)
9.4.3 应用举例	(220)
9.5 SQL 简介	(232)
9.5.1 数据查询语句	(232)
9.5.2 数据操作语句	(234)
9.5.3 数据定义语句	(234)
9.5.4 数据查询应用举例	(235)
9.6 ADO 对象模型编程	(240)
9.6.1 常用 ADO 对象简介	(240)
9.6.2 ADO 对象编程操作形式	(242)
9.6.3 ADO 对象编程应用举例	(244)
9.7 数据环境设计器及其应用	(255)
9.7.1 用数据环境设计器连接数据源	(255)
9.7.2 用数据环境设计器编程	(257)
9.7.3 SQL 生成器及其应用	(258)
习题	(261)
第 10 章 编译与发布应用程序	(267)
10.1 应用程序的编译和运行	(267)
10.1.1 生成可执行文件	(267)
10.1.2 运行可执行文件	(268)
10.2 应用程序的发布	(268)
10.2.1 概述	(269)
10.2.2 打包和展开向导的使用	(269)
参考文献	(273)

第1章

概论

Visual Basic 是一种可视化的、面向对象和采用事件驱动方式的程序设计语言,可用于开发 Windows 环境下的各类应用程序。它简单易学,效率高且功能强大。

在介绍 VB 语言及 VB 语言程序设计之前,首先对程序设计的基本概念做简单介绍。同时介绍 Visual Basic 6.0 的集成开发环境,为后面的程序设计提供实验平台。

1.1 程序设计概念

1.1.1 程序设计

程序是对解决问题的方案和步骤的描述。要用计算机解决问题,就要设计解决问题的程序,计算机通过运行程序而获得问题的解决。

程序设计是一门实用技术,程序设计要遵循一定的规则和方法。然而就像一般应用文写作一样,仅仅记住文章格式和语法规则是不够的,要不断地实践并且大量阅读他人成功的作品,才能驾轻就熟。

程序设计的结果是程序,程序设计是一个系统工程,所以程序设计的结果也称为程序系统(或称软件系统),如财务管理系统,人事管理系统等。程序系统设计一般包括以下步骤:

1. 系统分析

用计算机解决问题,首先要通过对问题的分析,确定这个问题要计算机做些什么?如果要解决什么问题都没搞清楚,如何解决问题?因此,分析问题,弄清楚要解决的问题并给出问题的明确定义是解决问题的第一步。

2. 系统设计

在弄清要解决的问题之后,就要考虑如何解决它,即如何做。

系统设计一般包括总体设计、数据结构设计、模块设计(或称算法设计)等内容。

总体设计。要解决的问题如果比较复杂,规模庞大,就要采用科学的分析方法,把原问题分解成若干个相对独立的小问题,即把系统划分为若干个模块。系统功能的细分有助于问题的解决。

数据结构设计。用计算机解决问题实际上就是数据处理。所以必须搞清楚问题的结果如何表达,为了得到结果应该收集哪些已知数据。一般的,系统中需要处理的数据是大批量的。数据结构设计就是找出数据之间的内在联系,即数据的逻辑关系,并把数据的逻辑关系转化为有效的存储结构。数据结构设计包括数据库设计、文件设计等。

界面设计。程序运行过程中一般需要用户干预。比如需要为程序提供运行数据,需要根据实际需要确定下一步该做什么。程序运行中也需要把运行结果通知用户。界面设计就是设计用户操作界面,提供一个人机交流的平台。

算法设计。一般把系统划分为若干的模块,每个模块解决系统中的一个子问题。模块设计(算法设计)就是给出解决问题的数学模型。或者说给出问题的具体处理步骤。数据结构设计和算法设计是一个问题的两个方面,数据结构设计解决的是数据表达方式,而算法设计解决的是数据处理方法。

3. 编码

在进行数据结构设计和算法设计时,往往采用某种与具体程序设计语言无关的语言(如流程图、自然语言等)来描述数据结构和算法。这样做的目的是为了从一开始就陷入程序设计语言的具体细节中。因为过多地涉及实现细节,不利于从较高抽象层次对问题本质进行考虑,并造成对设计过程难以把握和理解。当然,用其他方式描述的算法是不能被计算机执行的,计算机执行的程序必须用计算机能识别的某种程序设计语言描述。所谓编码就是把系统设计的结果变成计算机可执行的程序。

4. 测试与调试

程序编写完成之后需要进行程序的测试和调试,程序才能正式运行。

所谓测试,就是通过精心设计一批测试用例(包括输入数据和与之相应的预期输出结果),然后分别用这些测试用例运行程序,检验程序的运行结果与预期输出结果是否一致。可见,测试的目的是为了检验程序是否达到预期的功能。

所谓调试,就是找出程序中的错误并改正错误。因此,调试又称查错。调试和测试往往是交替进行的,通过测试发现程序中的错误,通过调试进一步找出错误的位置并改正错误。这个过程往往需要重复多次,直至达到满意的结果。

程序的错误通常有三种:语法错误、逻辑错误和运行异常错误。

语法错误是指源程序中某语句存在违反 VB 语法规则的地方,有些语法错误在语句输入时 VB 系统会自动发现,而有些语法错误要在程序编译时才会被 VB 系统发现,VB 系统发现语法错误时会给出“出错信息提示”并给出出错位置,用户可以根据提示信息修改源程序。

逻辑错误是指程序没有完成预期的功能,如源程序中错将 $c=a \setminus b$ 写成 $c=a/b$ 等。具有逻辑错误的程序仍然可以运行,但得不到正确的结果,或根本就无法获得结果。遇到此类错误,用户往往需要通过设置断点,跟踪程序的运行过程等调试手段,才能发现错误所在。逻辑错误可能是编码阶段产生的,也可能是系统设计阶段就埋下的隐患,所以排除逻辑错误可能要追溯到系统设计阶段。

运行异常错误是指程序对运行环境考虑不周而导致的程序运行异常终止。如异常的数据输入没有有效地筛选,逻辑判断的临界状态考虑不周等。

需要注意的是,无论你采用何种测试手段,都只能发现程序有错,而不能证明程序无错。几乎可以肯定,即使程序通过了良好的测试,总还是会有一些隐蔽的错误。作为一个程序设计者,你的任务就是尽可能地对程序进行彻底测试,使错误尽可能减少。

5. 整理文档资料

在程序系统设计的各个阶段都会产生相应的文档,如任务说明书、系统结构图、源程序代码清单等。这些资料对今后系统的运行和维护至关重要,必须加以整理并完整保存。最后还要编写系统的安装、使用和维护说明书,供程序系统的使用者参考。

6. 运行与维护

程序通过测试后就可交付使用了,但程序在使用中还需要不断得到维护。程序维护可分为三种:正确性维护、完善性维护和适应性维护。

正确性维护。虽然经过测试,程序依然可能存在逻辑错误,这些错误会在程序的使用过程中不断暴露,发现错误就得改正。

完善性维护。异常错误在系统测试阶段不容易发现,在系统实际运行时暴露会使系统运行失败。所以需要修补完善系统的运行环境。

适应性维护。随着时间的推移,用户要求修改或增加系统的功能,对于用户提出的修改和扩充程序功能的要求,程序开发者应考虑响应。如果对程序的功能进行了较大的更改,还应发布程序的新版本。

1.1.2 程序设计语言

人们写文章需要语言,如中文、英文等。人们写计算机程序也需要语言,写计算机程序的语言就称为计算机语言或称为程序设计语言。程序设计语言是一种人造语言,人们发明程序设计语言的目的就是编写程序。

1. 机器语言

程序的基本成分是指令,指令也是计算机最基本的操作步骤,描述指令需要使用二进制代码,因为计算机内部采用的是二进制。

表示各种指令的二进制代码称为机器语言。机器语言由计算机的 CPU 结构确定,即不同结构的 CPU 有不同的机器语言。用机器语言编写的程序,机器可以直接执行。机器语言编写的程序是由二进制代码组成的代码序列,这种程序可读性差,而且由于不同机器的机器指令不同,因此程序的可移植性差。用机器语言所编写的程序只能在相同的硬件环境下使用,大大地限制了程序的通用性。用机器语言描述问题处理的方式与人们习惯的思维方式有较大差距。使用机器语言进行程序设计,要求程序设计者具有深入的计算机专业知识,对机器的硬件有充分的了解。

2. 高级语言

用机器语言虽然可以编写程序,但程序开发效率低。20 世纪 50 年代中期以后,人们陆续推出了各种“高级程序设计语言”,简称高级语言。高级语言是与“低级语言”相对应的,因为机器语言与计算机结构关系密切,与人类自然语言差别极大,故称机器语言为低级语言。

与低级语言相比,高级语言的表达方式更接近人类自然语言,或者说高级语言是自然语言的一个子集。高级语言不依赖于计算机的具体型号,用高级语言编写的程序具有良好的可移

植性,同时具有良好的可读性。

历史上先后流行的高级语言有:Fortran、Algol、COBOL、Basic、Pascal、C、VB、VC、Java等。

用高级语言编写的程序称为“源程序”。源程序并不能直接在计算机上运行,需要通过系统软件把他转换成机器语言,计算机才能运行。把高级语言转换成机器语言的软件称为语言处理程序,各种高级语言都有相应的语言处理程序。语言处理程序一般有两种工作方式,即编译方式和解释方式。

编译方式。编译方式需要两个步骤:

(1)执行语言处理程序(这时也称编译系统)把源程序“翻译”成对应的机器语言程序(通过编译而获得的对应机器语言程序也称为目标程序);

(2)执行目标程序,获得问题的解。

解释方式。解释方式并没有产生源程序对应的目标程序。语言处理程序(这时也称解释系统)对源程序中的每个语句逐句解释成二进制代码并立即执行该代码,即“边翻译,边执行”。

VB 源程序可以采用编译方式运行,也可采用解释方式运行。一般在系统开发的程序调试测试阶段,采用解释方式运行;程序开发完成后,把源程序编译成目标程序,以后就直接运行目标程序。

1.1.3 程序设计方法

随着计算机技术的不断发展,人们对程序设计方法的研究也在不断的深入。早期衡量程序设计质量的主要标准是程序的运行时间多少和程序运行时占有内存空间大小。然而随着计算机硬件技术的不断进步,计算机运算速度不断提高,存储容量不断扩大,良好的程序结构已成为衡量程序质量的主要标准。一个结构良好的程序易于阅读和理解,便于测试和调试,便于修改和扩充。

早期受计算机功能的限制,程序规模不大,程序设计一般采用“面向过程的程序设计”方法。20 世纪 90 年代以后,应用程序的规模越来越大,“面向对象的程序设计”方法便应运而生。

1. 面向过程的程序设计

程序是对问题处理过程的描述,程序设计就是用计算机语言把问题的详细求解步骤描述出来。支持面向过程程序设计的语言有 Fortran、Basic、Pascal、C 等。

20 世纪 60 年代末出现了所谓“软件危机”,迫切需要对程序设计进行规范,因此提出了“结构化程序设计”方法。

结构化程序设计采用“自顶向下、逐步求精和模块化”的分析方法,有效地把一个复杂问题分解成若干个易于控制和处理的子问题,即把一个大型程序设计任务分解成一个个小程序,这样便于开发和维护。

自顶向下是指对设计的系统进行全面分析和理解,从问题的全局入手,把一个复杂问题分解成若干个相对独立的子问题(也称模块),然后再对各子问题作进一步的分解,如此重复,直到每个问题都容易解决为止。

逐步求精总是和自顶向下结合使用的,逐步求精步骤是自顶向下设计的具体体现。模块逐步细化为更小的模块是逐步求精。把模块中的功能逐步分解细化成一系列基本操作步骤,以致能用计算机语言的基本语句描述也是逐步求精。可见逐步求精把程序设计过程看作是一个渐进的过程。

模块化是结构化的重要原则。所谓模块化就是把大程序按照功能划分为较小的程序,每个小程序就是一个模块。

结构化程序设计还要求模块内部只能使用三种基本的控制结构,即顺序结构、分支结构和循环结构。

2. 面向对象的程序设计

随着硬件性能的提高和图形用户界面的推广,应用软件的规模迅速膨胀。由此引起的程序复杂性、数据安全性和代码可重用性等问题,面向过程的结构化程序设计方法已经很难解决了。

面向过程的程序设计是基于求解过程来组织程序流程的。在这类程序中,数据和施加于数据的操作是独立设计的,以对数据进行操作的过程作为程序的主体,程序的执行流程是人们事先设计的。

面向对象程序设计方法既继承了结构化程序设计的一切优点,又考虑到现实世界与计算机解空间的影射关系。面向对象的程序设计以“对象”作为程序的主体。对象是数据和数据操作的封装体,封装在对象内部的程序(操作)是通过“消息”驱动运行的。在图形用户界面上,消息可通过鼠标或键盘的操作传递。

支持面向对象程序设计方法的语言有 Visual Basic、Visual C++、Delphi、Java 等。

专业软件机构(如 Microsoft)已经把一些常用对象精心设计成各种“类”,然后把他们组织和存放在程序设计环境的类库中。由于这些类的存在,使程序设计过程中能很方便地把类实例化成为对象,一般程序设计者能分享专业人员的成果(也称为代码共享),使程序设计就变得简单了。

1.2 Visual Basic 概述

1.2.1 Visual Basic 的发展

Visual Basic 是在 Basic 语言基础上发展起来的。

Basic 是 Beginners All Purpose Interchange Code(初学者通用符号代码)的缩写。与其他高级语言相比,他语法规则简单,容易理解,容易掌握且能编写具有实用价值的程序,被认为最理想的入门语言。

Basic 自 1964 年问世以来长盛不衰,在广泛应用中与时俱进,不断发展。20 世纪 80 年代,为了结构化程序设计的需要,新版本的 Basic 语言在功能上进行了较大的扩充,增加了数据类型和程序控制结构,成为结构化程序设计语言。当时较有影响的版本有 True Basic、Quick Basic 和 Turbo Basic 等。

随着 Windows 操作系统全面替代 MS-DOS 操作系统,图形用户界面(Graphic User In-

terface, GUI) 必然代替字符用户界面, 但编写图形界面的应用程序工作量极大。于是可视化程序设计语言应运而生, 有了可视化程序设计语言, 程序设计者不用编写代码就可以设计出具有实用价值的图形用户界面。1991 年 Microsoft 公司推出了 Basic 的可视化版本 Visual Basic。

因为 Basic 语言的易用性和普及性, Visual Basic 同样受到广大专业和业余程序设计者的欢迎。从 1991 年到 1998 年, Microsoft 公司先后推出了 6 个 Visual Basic 版本。其中 1998 年推出的 Visual Basic 6.0 提供了更多、功能更强的用户控件, 并增强了多媒体、数据库和网络功能。为了适应网络技术的快速发展, 2002 年 Microsoft 公司推出了 Visual Basic. Net。Visual Basic. Net 功能更强, 而且已经演化成完全面向对象的程序设计语言。

笔者把本书的读者定位为程序设计的初学者, 所以以 Visual Basic 6.0 (以后简称 VB) 作为蓝本介绍, 并以 Visual Basic 6.0 作为实验平台。Visual Basic 6.0 包含 3 种版本, 即学习版、专业版和企业版。学习版包含了 Visual Basic 6.0 的基本功能, 而企业版功能最强。本书提供的程序在 3 种版本上均可运行。

1.2.2 Visual Basic 的特点

VB 是一种可视化的、面向对象的、事件驱动的、结构化程序设计语言。

1. 可视化

VB 语言用于开发在 Windows 操作系统上运行的图形用户界面的应用程序。图形用户界面包含窗口、菜单、对话框、按钮等各种对象, 如果这些对象都由开发者自己编写代码实现, 工作量十分庞大, 对初学者而言更是不可能的。VB 把各种对象的数据和代码“封装”起来, 作为可视化设计工具提供给程序开发人员, 开发人员就可以根据自己的设计思想, 用鼠标和键盘制作出 Windows 风格的图形用户界面, 大大提高了程序设计的效率。

2. 面向对象

构成 VB 应用程序的基本单元是对象, 即 VB 应用程序由一组对象组成。

把相关的数据和代码封装起来作为一个对象。在 VB 中大部分的对象是可见的 (如窗体、命令按钮等), 而且是由系统提供的。程序设计者不用像 C++ 语言一样自己定义对象的程序代码。当然如果需要, 程序设计者也可以自己定义对象。

3. 事件驱动

设计 VB 应用程序, 不用事先设计好整个程序的流程, 而只是编写一些相对短小的独立小程序, 即所谓“过程”, 这些过程分别属于各对象。过程可以由用户操作引发的事件驱动, 也可以由系统产生的事件驱动。

一个对象可能会产生多个事件, 每个事件都可以通过一段程序来响应。例如当鼠标单击某按钮, 该按钮对象就发生一个单击事件, 如果事先已经为该按钮的鼠标单击事件编写了一段程序, 鼠标单击事件就会驱动该程序的执行。

4. 访问数据库

Visual Basic 系统具有很强的数据库管理功能, 可以直接建立和管理 Microsoft Access 数

数据库。通过数据控件可以简单地和多种数据库建立连接,查询并操作本地和网络数据库。

1.3 Visual Basic 的集成开发环境

VB 系统集成了应用程序的用户界面设计、源程序代码的编辑,编译和连接、源程序的测试和调试等功能,故称为集成开发环境。

启动 VB 后,显示如图 1-3-1 所示的“新建工程”对话框。对话框有三张选项卡,其中“新建”选项卡用于创建新工程,而“现存”和“最新”选项卡用于打开已存在的工程。

用 VB 开发应用程序,每个应用程序对应一个工程或一个工程组。

通过“新建工程”对话框可以创建不同类型的工程,本书着重讨论“标准 EXE”工程。在对话框中选择“标准 EXE”,单击“打开”按钮,即进入如图 1-3-2 所示的 VB 集成环境。如用户主要使用“标准 EXE”工程,希望启动 VB 后就直接进入 VB 集成开发环境,可以在图 1-3-1 对话框中勾选“不再显示这个对话框(U)”选项,以后每次启动 VB 就直接打开图 1-3-2 所示的集成环境窗口。



图 1-3-1 新建工程对话框

1.3.1 集成环境窗口

VB 集成开发环境除了主窗口以外,还包含很多子窗口,图 1-3-2 中的子窗口已经过重新排列,和你打开的集成开发环境可能不一样。你可以通过主窗口的“视图”菜单打开或关闭各子窗口,并通过鼠标对子窗口进行缩放和移动。

1. 主窗口

启动 VB 后,主窗口位于集成环境的顶部,该窗口由标题栏、菜单栏、工具栏组成。

(1)标题栏。除显示工程名称外,还显示 VB 三种工作模式中的当前工作模式。三种工作