

Visual Basic 4.0

编程技术

Visual Basic

Visual Basic

知音难觅·好书难求

珍藏版

况正谦 编著

西北工业大学出版社

TP312
KZG/1

Visual Basic 4.0 编程技术

况正谦 编著

西北工业大学出版社

1997年4月 西安

(陕)新登字 009 号

【内容简介】 Microsoft 公司的 Visual Basic 是一种面向普通用户的 Windows 应用程序开发平台,其易学易用和功能强大的可视化程序开发方式广受欢迎,一经推出即迅速占领市场,成为最流行的软件开发平台之一。本书以 Microsoft 公司最新推出的 Visual Basic 4.0 版本为准,主要面向广大初学者和普通用户,系统全面地介绍了 Visual Basic 4.0 版的使用和编程。

本书在内容上力求深入浅出,在结构安排上力求循序渐进。为了兼顾具有不同编程背景的读者,本书分为入门、应用和提高三篇。初学者可以从入门篇开始;具有一定 Basic 语言编程知识和 Windows 使用经验的普通用户可以对入门篇大致浏览,把重点放在应用篇上;熟悉 C 语言或 C++ 语言,对 SDK 和 MFC、OWL 有一定了解的程序员也可以从提高篇中汲取营养。

本书所有实例程序均收录在一张软磁盘中,需要者请与编辑部联系。

Visual Basic 4.0 编程技术

况正谦 编著

责任编辑 何格夫

责任校对 雷 军

*

© 1997 西北工业大学出版社出版发行

(710072 西安市友谊西路 127 号 电话 8493844)

全国各地新华书店经销

陕西省咸阳市印刷厂印装

ISBN 7-5612-0924-X/TP·120

*

开本:787×1092 毫米 1/16 印张:30.375 字数:746 千字

1997 年 4 月第 1 版

1997 年 4 月第 1 次印刷

印数:1—6 000 册

定价:39.00 元

购买本社出版的图书,如有缺页、错页的,本社发行部负责调换。

前 言

美国微软(Microsoft)公司于 1992 年推出的 Windows 3.1 为个人计算机带来了深刻的变革。这一变革就是计算机对用户来说更加“友好”了。在 Windows 下,计算机中各种可操作的对象都用形象的图标表示出来,用户使用鼠标单击(click)或拖动(drag)就可以完成各种复杂的操作。这就是可视化操作的概念,各种对象和操作一望而知。用户经过简单的学习就能掌握计算机的使用,不必再像以往那样要记住复杂 DOS 操作命令。

Windows 在给用户带来最大的方便的同时,带给程序员的却是编程的困难。在熟悉 DOS 编程的程序员看来,Windows 编程中的新概念和新机制太多、太复杂,编程的工作量也太大。很多 DOS 程序员试图转向 Windows,结果花了几个月甚至半年,最终不得不半途而废。

自从 Visual Basic(以下简称 VB)出现之后,情况大为不同。VB 把可视化的概念也引入了编程过程,用鼠标点取和拖动同样能完成编程过程中的大部分工作量,程序设计第一次成为一种乐趣。“半个小时学会 Windows 编程”,这在以前也许人们想都不敢想,而 VB 实现了这一神话。Visual Basic 3.0 for Windows(以下简称 VB 3.0)是微软公司推出的面向 Windows 的高级开发系统。它采用的是事件驱动机制(Event-Driven Mechanism)、可视化界面设计和集成化的开发、调试及编译环境,大大降低了开发 Windows 应用程序的难度。开发人员不再要求是 C、C++ 的 Windows 编程专家,特别适合于初学者和普通用户。同时,VB 以其强大的功能也兼顾了高级程序员,利用 VB 同样可以开发出大型、复杂的应用软件。VB 正是以其界面友好、易学易用、功能强大等特点受到 Windows 用户和程序员的青睐,得到了迅速推广,成为首屈一指的 Windows 开发系统。

在 3.0 版之后,为了适应新的 32 位 Windows(Windows NT 和 Windows 95),微软公司又于 1995 年推出了 Visual Basic 4.0 版。4.0 版在 3.0 版的基础上有了重大的改进,除了引入更先进的程序开发方法外,最关键的是它能够兼容于 16 位(Windows 3.x 版)和 32 位的 Windows。

本书由浅入深、循序渐进地介绍了如何在 Microsoft Visual Basic 4.0 下开发 Windows 应用程序,主要面向 Windows 的普通用户和初学 Windows 编程的人员,同时也兼顾了一般的 Windows 开发人员,包括 VB 2.x 和 3.x 版的程序员,另外也可供使用 SDK、MFC 和 OWL 的高级程序员参考。书中所附大量实例都是在实际系统中开发调试成功的,有助于读者更好地掌握 VB 编程中遇到的新概念和新方法。书中的插图是在实际系统的运行过程中用屏幕拷贝功能截取下来,并用 Windows 的画笔程序以位图文件保存下来的。

因时间仓促、水平有限,书中挂漏贻误之处自知难免,诚望广大读者不吝指正。

作 者

1996 年 9 月

目 录

绪论.....	1
---------	---

入门篇

第一章 VB 编程新概念	5
1.1 面向对象编程	5
1.2 事件驱动机制	6
1.3 快速应用开发	6
1.4 语法定义符号	7
第二章 VB 开发环境.....	8
2.1 主窗口	8
2.2 工具箱	14
2.3 窗体	15
2.4 属性窗口	16
2.5 工程窗口	16
第三章 VB 快速入门	17
3.1 VB 编程步骤.....	17
3.2 VB 常用控件.....	23
3.3 制作菜单	48
第四章 VB 编程基础	52
4.1 VB 数据类型.....	52
4.2 结构化程序设计	60
4.3 输入与输出	70
4.4 VB 运算符.....	85
4.5 常用标准函数.....	100
4.6 常用语句	112

应用篇

第五章 多重窗体	117
5.1 窗体对象.....	117

5.2 建立多重窗体	134
第六章 绘图	146
6.1 与绘图有关的控件	146
6.2 在窗体上绘图	182
第七章 鼠标及键盘	206
7.1 鼠标事件	206
7.2 键盘事件	226
第八章 计时器	236
8.1 使用和编程	236
8.2 应用举例	238
第九章 使用 Windows 剪贴板	253
9.1 在程序中访问剪贴板	253
9.2 剪贴板对象的方法	253
9.3 应用实例	256
第十章 加入新的控件	267
10.1 在工具箱中加入新控件	267
10.2 通用对话框	268
10.3 Outrider SpinButton	283
10.4 Microhelp Key State	290
10.5 Microsoft Masked Edit	300
第十一章 文件操作	313
11.1 文件的存储格式	313
11.2 文件访问	313
11.3 应用举例	323
第十二章 多文档界面	338
12.1 多文档界面的组成和特点	338
12.2 VB 中的多文档界面	341
12.3 多文档界面应用实例	356
第十三章 动态数据交换	388
13.1 使用 VB 的 DDE	388
13.2 创建 DDE 源链接和目的链接	388
13.3 与 DDE 有关的属性、事件和方法	389
13.4 应用举例	395
第十四章 对象链接和嵌入	413

14.1 OLE 的术语和概念	413
14.2 Custom Controls 中的嵌入对象	414
第十五章 程序调试及错误处理	422
15.1 程序调试	423
15.2 错误处理	425
第十六章 条件编译	428
16.1 # If - Then - # Else 指令	428
16.2 # Const 指令	429

提 高 篇

第十七章 API 接口	431
17.1 API 的声明和调用	431
17.2 API 应用举例	432
第十八章 VB 与 VC/BC 的接口	444
18.1 在 VC 下生成动态链接库	444
18.2 应用举例	444
附录	453
附录一 VB 关键字	453
附录二 VB 常量	462
附录三 VB 语句	466
附录四 VB 标准函数	467
附录五 VB 对象	469
附录六 VB 属性	470
附录七 VB 事件	473
附录八 VB 方法	475
附录九 VB 可俘获错误(Trappable Error)	476

绪 论

一、Windows 编程和 Visual Basic

早期的 Windows 程序员使用 Windows SDK (Software Development Kit, 软件开发包) 开发 Windows 应用软件。这种开发方式要求程序员精通 C 语言程序设计, 并了解 Windows 的几乎每一技术细节, 因而不只难以掌握, 而且编程工作量极大, 开发过程中极易出错, 还难以调试。随后 Microsoft 公司和 Borland 公司分别推出了基于 C++ 语言的 Microsoft Visual C++ 和 Borland C++, 并在其中加入了支持开发 Windows 应用软件的 MFC (Microsoft Fundamental Class, Microsoft 基本类库) 和 OWL (Object Windows Library, Windows 对象库)。MFC 和 OWL 利用 C++ 语言中对象 (Object) 的概念, 把 SDK 中的绝大部分细节封装 (Encapsulate) 在各种对象中, 程序员只须从库中调出所需的对象即可, 同时还保留了最大的灵活性, 因而 MFC 和 OWL 已成为当今最流行的 Windows 开发系统。与 MFC 和 OWL 相比, VB 更大众化, 倾向于普通的 Windows 用户, 而 MFC 和 OWL 则是高级 Windows 程序员的首选开发系统。与 MFC 和 OWL 不同, VB 不要求用户有深厚的技术储备, 不必去熟悉 C++ 和 SDK, 一般来说, 一个普通用户只须花半个小时即可入门, 这无疑是 VB 最引人之处。当然, 从功能和灵活性上讲, VB 稍逊于 MFC 和 OWL, 但是通过一定的编程技巧也能加以弥补。

二、VB 4.0 版新特色

在 1993 年推出 VB 3.0 版之后, Microsoft 公司又于 1995 年推出 VB 4.0 版。VB 4.0 版在以往版本的基础上有了重大的改进, 加入了很多新的功能和新的特性, 最重要的是 VB 4.0 可以兼容 16 位 Windows (Windows 3.x) 和 32 位 Windows (Windows NT 及 Windows 95)。VB 的新特色包括:

■ 加入了 VB 2.0 应用版作为语言引擎 (Language Engine) VB 2.0 应用版与以前 VB 版本下开发的独立应用程序 (Stand-alone Application) 完全兼容, 并兼容 VB 1.0 应用版 (Microsoft Excel 5.0 版和 Microsoft Project 4.0 版中包含了 VB 1.0 应用版)。有了 VB 2.0 应用版作为语言引擎, 在 VB 4.0 下进行 OLE 自动化对象 (OLE Automation Object, 一种可由其它应用软件和编程工具通过 OLE 自动化接口访问的对象) 的编程更为简单, 同时也意味着 VB 代码更容易在支持 OLE 自动化的模块和应用软件中移植。

■ OLE 自动化 有了 OLE 自动化, 用户可以在 VB 应用程序内部控制其它应用程序中的对象, 实现对其它应用程序的功能借用。如果该对象是一个 OLE 自动化对象, 就可以在 VB 代码中使用该对象的属性 (Property) 和方法 (Method)。OLE 自动化对象好比一个预制件, 用户可以在代码中从其它应用程序搜集数据和函数, 并把它们转化为程序能用的信息和模块。举例来说, 可以构造一个应用程序, 使用 Microsoft Excel 作为计算引擎, 而这个计算引擎的结果又以 Microsoft Word 的文档格式生成。还可以把一个对象 (比如说 Microsoft Excel Chart) 直接加入到 VB 的工具箱中, 以便于直接进行拖放 (Drag-and-Drop) 操作。

■ 可重用的对象和集合 (Collection) 在 VB 4.0 中可以生成可重用的对象, 它们有自己的属性和方法, 可以组装到一个对象模型 (Object Model) 中。这些对象定义为“类 (Class)”。类是对象的正规定义, 它好比是一个模板, 在运行时对象从这个模板生成实例

(Instance)。类定义了对对象的属性和方法,用以控制对象的行为。可重用对象的定义包含在 VB 新的类模块中。在 VB 4.0 中甚至还可以生成自己的集合,这要用到 VB 新的集合对象。

■ **更强大的开发环境** 在 VB 4.0 新的开发环境下,加入了文本泡(Text Bubble)功能,当鼠标光标移动到工具条(Tool Bar)中的某一按钮或工具箱(Tool Box)中的某一控件(Control)上,经过大约一秒钟的延时后,一个包含该按钮或控件的名称的小文本泡会自动浮现出来,提示该按钮或控件的功能。另外,当鼠标光标位于窗体(Form)或控件等对象上时,按下鼠标右键,可以弹出一个与此对象相关的菜单,包含了对该对象的常用操作。这两点小小的改进非常类似于 Windows 95 的某些功能,大大方便了用户的使用和操作。

■ **更严格的作用域限制和面向对象编程概念** 在 VB 4.0 中,所有的声明(Declaration)和定义(Definition)之前都可以冠以 Private(私有)或 Public(公有)关键字,以限定所定义或声明的对象的作用域。这样就更具备了面向对象编程的特性,为我们开发大型程序提供了有力的支持。

■ **属性过程(Property Procedure)** 属性过程为类模块生成和控制属性。用它可以在窗体、标准模块和类模块中加入用户自定义的属性,以及在这些属性被设置或复位时执行相应的代码。例如,用户可以给一个窗体加上一个 Inverted(翻转)属性,当此 Inverted 属性被设置为 True(真)时,与此属性相关联的代码过程激活一个 API(Application Program Interface,应用程序接口,这是 Windows 为应用程序提供的系统服务函数库),把窗体中的位图(Bitmap)翻转过来。

■ **对象浏览器(Object Browser)** 对象浏览器以谱系结构(hierarchical)的形式显示出应用程序可以使用的来自 VB 和其它 OLE 组件中所有的类、属性和方法。这样一来,就可以轻松地选出所需对象,然后把它们剪贴(Cut - and - Paste)到应用程序的代码中。

■ **条件编译(Conditional Compilation)及 32 位支持** 有了条件编译,就可以把与开发平台相关(Platform - Specific)的代码段嵌入到 # If... Then... # Else 结构中。这样,从一套源代码中可以有选择地生成 16 位和 32 位两个版本的应用程序。VB 的 32 位版本支持所有工程组件(Project Component)中的长文件名,还解除了 VB 语言中绝大部分的 64K 容量限制。

三、内容安排

本书以运行于 Windows 3.1 上的 Visual Basic 4.0 版为基础,以入门为宗旨,介绍 VB 4.0 的使用和编程。本书分为入门、应用和提高三篇,由浅入深、循序渐进地介绍了如何使用 VB 4.0 开发 Windows 应用程序。

入门篇分为四章。第一章介绍 VB 引入的编程新概念,帮助初学者建立起 Windows 编程的基本概念。第二章介绍 VB 4.0 的集成化开发环境,帮助读者了解 VB 4.0 开发环境中各种窗口、菜单、按钮以及工具的作用和功能。为避免一开始便强迫初学者去啃冗长乏味的语法定义,第三章安排为快速入门。首先用两个非常简单的例子介绍 VB 编程的三个步骤。通过这两个例子,读者可以立即感受到用 VB 开发 Windows 应用程序的便捷之处,并迅速建立起 VB 编程的感性认识。随后介绍 VB 的常用控件以及如何生成菜单。每个控件以及菜单都带有简单的实例,帮助读者迅速掌握其用法。本篇的最后一章——第四章较为系统地介绍了 VB 的语法规则和编程基础,包括语法定义、语句结构、流程控制、函数和过程、输入输出、常用语句、常用过程和常用函数。初学者学完本篇后,已具备开发普通的 Windows 应用程序的能力。

应用篇包括十二章,分别介绍多重窗体、绘图、鼠标及键盘、记时器、剪贴板的使用、新的

控件、文件操作、多文档界面(MDI)、动态数据交换(DDE)、对象链接和嵌入(OLE)、程序调试及错误处理、条件编译。本篇的内容涵盖了绝大部分实际应用中的编程方法,能够适应绝大多数实际应用。

提高篇包括两章:API接口、VB与VC/BC的接口。这一篇介绍VB程序的高级开发技巧。

另外,因为VB的功能非常强大,内容极其繁多,本书作为一本以入门为主的编程指南,不可能逐一加以详细说明,只能拣最关键最常用的介绍。为了便于读者系统地查阅,本书在最后一部分的附录中汇编了VB的关键字、常量、语句、标准函数、对象及其属性、事件和方法以及可俘获错误(Trappable Error)。

第一章 VB 编程新概念

在开始学习 VB 编程之前,有必要介绍几个新概念。Windows 界面友好、功能强大,但是为了开发和维护这样的界面,必须引入全新的思维方式和编程技术。

1.1 面向对象编程

什么是面向对象编程?简而言之,就是把对象(Object)的概念引入编程的方法和过程中。那么什么是对象呢?对象是一个实体,包含了自身的特性和控制方法,这些特性和方法是以数据和控制过程的形式通过封装的技术捆绑在对象中。这种笼统的解释可能会使初学者迷惑不解,下面举一个例子来进一步阐述对象的概念。

在几年前 Borland 公司的 Turbo C 非常流行的时候,当我们想在 DOS 下用 Turbo C 开发一个菜单程序时,要在屏幕上画出一个菜单,这包括主菜单条、每个主菜单项的文字(即标题)、每个下拉菜单及其选项,还要记录每个菜单项在屏幕上的位置和大小,以便后面检测是否选中此项,然后程序开始循环检测光标,并负责调用与之相关的过程。难度虽然不大,但是工作量却非常大。尤其是当我们要修改扩充或转向开发另一个菜单程序时,又得在此基础上作较大幅度的改动,也就是说这样的程序可维护性和可移植性太差。这种应用程序的开发方式就是胡子眉毛一把抓,所有的编程细节都要求开发者逐一处理。这正是传统的结构化程序设计方法的弊病所在。

在面向对象开发方法中则轻松得多,系统已经为用户准备好了一系列的标准“模板”。菜单无须亲自动手去画,从系统中调出一个标准的菜单模板,在模板上添上实际所需的选项就行了。还可以调整菜单在屏幕上的位置和大小,使它适合实际需要。这些在 Windows 下往往只用鼠标拖动一下就可完成,绝大部分“低级劳动”的工作量已经由系统代劳了。从系统中调出的现成的模板就是一个对象,它是在保留了所有菜单的共性,并留给调用者修改和扩充的余地的基础上抽象而成的。这种方法把大量的细节问题和低级操作隐藏起来,无须用户过问,这样用户就把大部分时间和精力腾给了高级的控制,从而大大提高了编程人员的效率。对象中一切琐碎的东西都被封装起来,对外表现为一个具备一定特性、能完成一定功能的黑匣子。

如果上面的例子仍使你不得要领的话,没有关系,这并不影响你使用 VB。在 VB 中,几乎所有的东西都是对象,所以只要去使用就是了,不必关心它的确切定义以及内部细节。

怎么使用对象这个黑匣子呢?通过对象的属性(Property)和方法(Method)。属性和方法好比黑匣子上的仪表板和控制按钮。

属性是对象对外表现出的特性。属性可以是多种多样的。例如,菜单条的颜色就可以是一种属性,菜单项的字体也可以是一种属性。属性的取值也是多种多样的。比如颜色属性可以取值为一个整数,代表某一颜色代码;字体的取值可以是一个字符串,代表该字体的名称。属性值可由应用程序读取,大多数属性值还可被应用程序随时修改,这就给了编程人员极大的控制余地。例如,下面的语句通过修改宽度属性可以使名为Form1的窗口宽度减为原来的一半:

```
Form1.Width = Form1.Width/2
```

对象的方法是对象提供给应用程序的一个调用接口,通过调用对象的方法,应用程序可以实现与该对象相关的功能或达到控制操作该对象的目的。下面的语句调用对象(窗口)的打印方法 Print,可以在窗口 Form1 中打印“Hello!”:

```
Form1.Print "Hello!"
```

VB 为程序员提供了数量庞大的对象,每个对象又包含大量的属性和方法。我们在编程的时候,主要就是与这些对象、属性和方法打交道。通过各种各样的对象及其属性方法的组合,就能构造出功能强大、变幻莫测的应用程序来。

1.2 事件驱动机制

事件驱动机制(Event - Driven Mechanism)是 Windows 程序设计的另一大特色。与事件驱动相对,以往的 DOS 程序是代码驱动的,也就是说程序的流程由程序自身决定。而事件驱动的程序则不同,程序的流程是由用户的输入等外部事件决定的,而不是以程序事先安排好的顺序去执行。例如,有一个服务程序,界面上有三个按钮或选项,每个按钮或选项对应一项具体的服务。在 DOS 下,该程序的机制是:轮流查询这三个选项,看是否被选中,如果被选中,则调用相应的服务过程,然后继续反复查询。注意,这些操作都是由应用程序自身去控制完成的,也就是说需要程序员亲自编程实现。而在 Windows 下,程序只须做三个按钮,然后编写三个按钮所对应的服务过程,当某按钮被用户鼠标按动时,Windows 系统会感知这一动作,并把它作为一个“事件(Event)”发送给该程序,自动激活相应的服务过程。对于应用程序来说,它不是主动去查询有无事件发生,而是在事件发生后由 Windows 系统“叫醒”。事件驱动机制的思想同样是把一些具有共性的低级操作从程序员身上解脱下来,交给 Windows 系统去做,这样不仅大大减小了程序员的工作量,更重要的是这种处理机制更接近于现实世界。

1.3 快速应用开发

Visual Basic 的“Visual”是“可视化”的意思。VB 的过人之处在于可视化的思想贯穿于程序开发过程的始终。

在非可视化开发系统下开发程序时,程序员按预先构思好的设计,在字符编辑器下闷头编写菜单程序。等程序写好之后,编译链接运行时才能看到自己设计的界面到底是什么样子,是否符合原先的设计。在编程过程中,很可能由于疏忽或考虑不周而使运行时显示的界面与自己原先的设想大相径庭,这时就得费更大的气力去查找错误出在什么地方。运行时期的错误

一般是最难查找的,这就是所谓的“编程容易调试难”。造成这种问题的原因是开发过程和调试过程严重脱节。

在 VB 这样的可视化开发环境中开发界面程序时,程序员用鼠标直接在屏幕上画出自己设计的界面。这样程序员在整个开发过程中都能直接看到程序运行时的样子。而且程序员每次对界面的修改都能立刻反映在屏幕上,程序员能够清楚地看到修改的结果。这就是“所见即所得(WYSIWYG, What You See Is What You Get)”。不会再发生运行后才返回头查找修改界面设计这类低级错误的事了,这类错误在设计过程中就隐含地被排除了。

在 VB 下开发程序时,可以从界面设计开始,先构造一个符合外观要求的空壳,然后逐步扩充内容。在扩充过程中,随时都可以把当前的半成品运行一下,看看效果如何。如果出了问题,错误一般被限制在新扩充的范围内,可以很容易地找出,并及时得到修改,而不是把隐含的错误统统堆积到整个程序编写完,那时再要查错和修改就非常困难了。这种开发方式称为快速应用开发(RAD, Rapid Application Development),或称为快速原型法,其基本思想是把调试过程贯穿在开发过程的始终,从而把错误消灭于萌芽状态。

VB 环境下的程序有三种状态:开发(Design)或称设计、运行(Run)和调试(Break)。程序员只需按动工具条上的一个按钮,就可以使正在开发的程序投入运行。同样也只须一个按钮就可以使正在试运行的程序返回开发状态,或进入调试状态。当运行一个程序时,如果 VB 执行到用户设置的断点处或遇到运行错误时,VB 自动使程序进入调试状态。

此外,VB 还沿用了 Quick Basic 的一大功能,编辑器能够自动对当前行进行语法检查。Visual C++ 和 Borland C++ 只能识别关键字并用不同的颜色标记,而 VB 还能提示程序员该行错在何处,帮助程序员立刻修改。

1.4 语法定义符号

本书中沿用了 Basic 的语法定义符号,这些符号在 VB 的联机帮助文档中也时常见到,此处对常用的符号介绍如下。

- (1) 方括号[]中的内容为可选内容。
- (2) 花括号{ }或方括号中以管道线|分隔开的内容是互斥的,实际的程序中只能从中选择一项。
- (3) 大写字母打头的正体字为关键字,在编辑程序时要原封不动地抄写。
- (4) 斜体字为占位字,程序中要用实际的对象替换。

例如,变量定义的语法为

```
{Dim|Global|Private|Public} varname [As type ][, varname [ As type ][...]]
```

则在实际的程序中,变量定义可以是

```
Dim a As String
```

```
Public Count As Integer, temp, sum As Single
```

等等。

第二章 VB 开发环境

本章介绍 Visual Basic 4.0 的可视化集成开发环境。

在安装了 VB 4.0 之后,如果没出什么错误,VB 4.0 的安装程序 SETUP.EXE 会自动在 Windows 的程序管理器中建立一个名为 Visual Basic 4.0 的程序组。用鼠标双击该程序组,然后再双击名为 Microsoft Visual Basic 4.0 的程序项就启动了 VB 4.0 的可视化集成开发环境。与 Microsoft Visual C++ 和 Borland C++ for Windows 相比,VB 环境下的内容稍多一些,但是都是方便用户的必不可少的工具。这些工具都易学易用,熟悉之后,会使程序开发变得得心应手。

启动 Microsoft Visual Basic 4.0 后,便进入了如图 2-1 所示的开发环境。

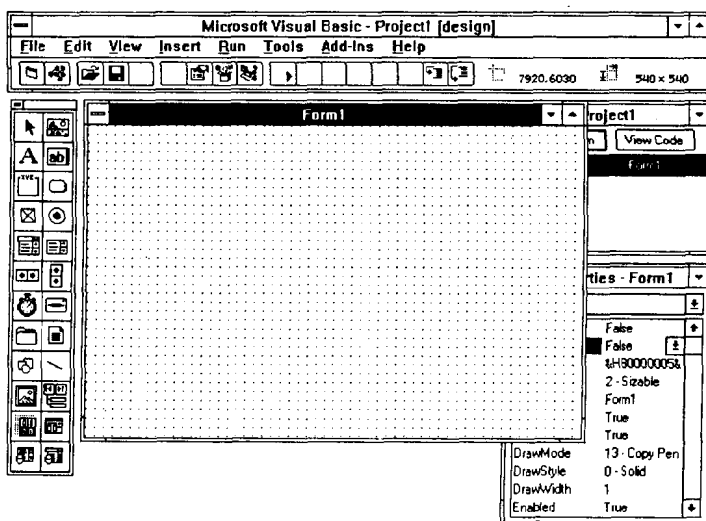


图 2-1 Microsoft Visual Basic 4.0 开发环境

下面逐一介绍各个窗口和工具。

2.1 主窗口

位于屏幕最上方的长条形窗口是 VB 的主窗口,如图 2-2 所示,这是 VB 的总控制台。

2.1.1 标题栏

标题栏中的“Project1”是当前开发的工程文件名,因为刚刚进入 VB 环境,用户还没有指定一个工程文件,故缺省为 Project1。方括号中的“design”标示出当前程序处于开发状态下。当用户运行或调试程序时,方括号中自动变为“run”或“break”。

一个完整的 VB 程序必须有一个工程文件,此外一般还可能有若干个模块文件。工程文件的扩展名一般为“.MAK",它以文本格式记录了本程序的一些重要信息,其中包括本程序包

含的所有模块文件名。当需要调入一个以前的程序时,只需在 VB 环境下打开它的工程文件就行了。VB 会根据此工程文件调入所有的模块文件。因此工程文件 and 应用程序是一一对应的关系。

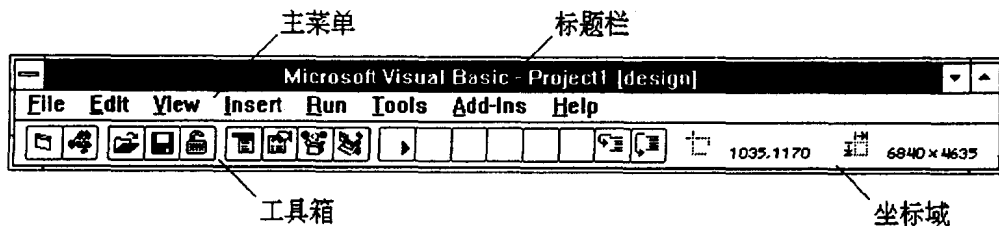


图 2-2 主窗口

应用程序的模块又可分为 5 种:窗体模块、标准模块、多文档窗体(MDI Form)模块、类模块以及资源(Resource)模块。

窗体模块文件以“.FRM”为扩展名,它描述了一个窗体的各种信息,包括:窗口的外观、所包含的控件、窗体和控件的属性设置、本窗体内的常量和变量定义、窗体内的过程和事件等等。应用程序中的每个窗体都对应于一个窗体模块文件。

标准模块文件以“.BAS”为扩展名,从扩展名可以看出,标准模块是一个 Basic 源代码。用户自己编写的过程、函数及声明定义等等都可以放在单独的标准模块中。

多文档窗体模块文件记录了多文档窗体的信息,有关多文档窗体的内容在后面详细讨论。

类模块文件以“.CLS”为扩展名,包含对类的定义(属性和方法)。

资源模块文件以“.RES”为扩展名,包含位图、文本字符串等或其它数据,这些数据被称为资源。把资源放在一个独立的文件中,可以允许用户将来修改这些资源数据,而不用把整个应用程序再编译一遍。

2.1.2 主菜单

主菜单中包括了 VB 开发环境下的所有控制命令。

1. File

File 菜单包含与系统和文件操作有关的命令。

New Project 命令创建一个新的工程文件。

Open Project... 命令打开一个已存在磁盘上的工程文件。

Save File 命令把当前模块文件,即在工程窗口中被加亮显示的模块文件存盘。

Save File As... 命令同 Save File,但要求用户输入一个新的文件名,即换名存储一个模块文件。

Save Project 命令把工程文件存盘。

Save Project As... 命令换名存储工程文件。

Add File... 命令在工程中加入新的模块文件,该文件必须已在磁盘上存在。

Remove File 命令把当前模块文件从工程中删除。如果 VB 发现要删除的文件被修改过但还没存盘,它会提示用户,询问是否要把该模块文件存盘。

Print Setup... 命令设置打印机。

Print... 命令打印模块文件。

Make EXE File... 命令把当前开发的工程编译成可在 Windows 下独立执行的 EXE 文件。这是用 VB 开发 Windows 应用程序的最后一步,生成 EXE 文件后,应用程序就可以直接由 Windows 启动,不必依赖于 VB 环境了。

Exit 命令退出 VB 开发环境,返回 Windows 系统。

如果打开过工程的话,在 Make EXE File... 和 Exit 之间还会有一些内容,这是曾经访问过的工程文件。通过这些项,你可以直接调出最近开发的工程。这一栏最多容纳 4 个最近访问的工程文件,当此栏装满而继续加入时,最早的一个工程文件被淘汰掉。

2. Edit

Edit 菜单中为编辑命令。

Undo... 命令取消上一个操作,恢复到操作前的状态。根据上一个操作的内容,Undo 后面可能是 Typing、Delete、Insert Text、Paste、Cut、Indent、Outdent 等等。

Redo... 命令重复上一个操作。

Cut 命令把选中的文本或控件剪切到剪贴板(Clipboard)上。

Copy 命令把选中的文本或控件拷贝到剪贴板上。

Paste 命令把剪贴板上的内容拷贝过来。

Paste Link 命令把剪贴板上的内容拷贝到目标控件上,同时建立起目标控件与此内容所来自的 Windows 应用程序的链接。这样,以后该应用程序对此内容的任何修改都能立即通知该目标控件并自动刷新。只有当往剪贴板上拷贝此内容的应用程序能够向 VB 传送数据,且目标控件有 DDE(Dynamic Data Exchange, 动态数据交换)链接功能时, Paste Link 才能使用。有关 DDE 的内容将在后面介绍。

Clear 命令删除选中的文本或控件。

Find... 命令搜索字符串。

Replace... 命令搜索字符串,并用新的字符串替换。

Indent 命令当前文本行向右缩进一个 Tab 宽,缺省时一个 Tab 宽为 4 个字符,该值可由用户修改设置。文本行内缩是结构化程序设计的要求,可以使程序的结构更加清晰,可读性更好。

Outdent 命令当前文本行向左移动一个 Tab 宽,这是 Indent 的逆操作。

Bring to Front 命令把选定的控件放在最上层。控件(Control)是加在窗体上的对象,常见的按钮就是一种控件,此外还有文本框、标签等等。在窗体中加入控件时,如果两个控件发生重叠,后加入的总是盖在原先的控件的上面。在 Windows 中,除了表示二维屏幕上位置的 X-Y 坐标外,还有一个表示前后的 Z-坐标。Z-坐标反映了窗口之间的覆盖关系,Z-坐标值大的窗口总是覆盖 Z-坐标值小的窗口。Bring to Front 就是把控件的 Z-坐标值设成最大,使该控件覆盖其它所有的控件。

Send to Back 命令把选定的控件放在最底层,这是 Bring to Front 的逆操作。

Align to Grid 命令把控件与窗体上的栅格对齐。在窗体中的密密麻麻的小点称为栅格,这是用于设置控件时对齐用的。缺省时加入窗体的控件总是自动对齐栅格的,但是在用户修改了这一设定后,控件就可以被移动到任何位置上。如果仍然想使某些控件对齐栅格,就可以选定它们,并执行 Align to Grid。