



附赠光盘

老兵新传

Visual Basic

核心编程及通用模块开发

张宁 编著



- 编程“高手”的功力体现在对语言的彻底掌握和丰富的实践经验
- 书稿内容由作者原创编写，直接面向实际应用
- 本书程序代码凝聚了作者十余年的编程经验
- 书中所有通用模块可不加修改直接调用



清华大学出版社



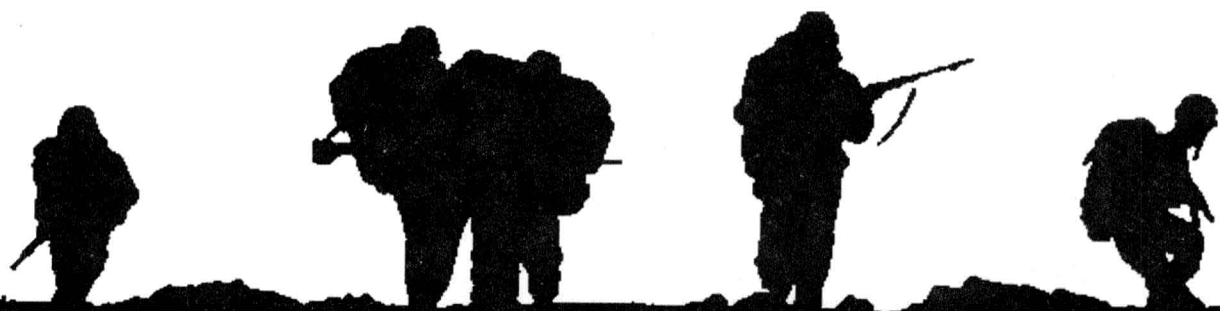
附赠光盘

老兵新传

Visual Basic

核心编程及通用模块开发

张宁 编著



- 编程“高手”的功力体现在对语言的彻底掌握和丰富的实践经验
- 书稿内容由作者原创编写，直接面向实际应用
- 本书程序代码凝聚了作者多年开发经验
- 书中所有通用模块可不

清华大学出版社
北京

内 容 简 介

Visual Basic 作为一种 Windows 软件开发工具,以简单易用和胜任快速开发著称,但 Visual Basic 系统本身提供的功能比较少,很多重要的软件功能不能直接实现。本书正是介绍如何弥补它的这种不足、增强 VB 程序功能的书籍,但增强功能的代码仍不失简单易用的特点,使 Visual Basic 成为既强大又好用的编程工具。书中介绍了使用 Visual Basic 语言进行 Win32 核心编程和高级编程的技术,并面向 Visual Basic 的开发实践和针对编程中的常见问题,编写了调用简单、运行独立、能胜任大型软件开发的可重用的代码模块,不仅大大增强 Visual Basic 的功能,而且减轻编程的工作量,并使程序代码更为简洁。

本书属于 Visual Basic 高级编程的书籍,适合有一定 Visual Basic 基础并想进一步提高 VB 开发技能的读者,可作为使用 Visual Basic 语言编程的软件开发人员、科研人员和广大 VB 爱好者的提高技能的读物。由于书中介绍了大量有关 Windows 运行机理和 Win32 编程的技术,因此对使用其他语言编程的 Windows 程序设计人员也有一定参考价值。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

老兵新传: Visual Basic 核心编程及通用模块开发 / 张宁编著. —北京: 清华大学出版社, 2012
ISBN 978-7-302-28310-2

I. ①老… II. ①张… III. ①BASIC 语言—程序设计 IV. ①TP312

中国版本图书馆 CIP 数据核字(2012)第 043869 号

责任编辑: 夏兆彦

封面设计: 柳晓春

责任校对: 胡伟民

责任印制: 沈 露

出版发行: 清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址: 北京清华大学学研大厦 A 座 邮 编: 100084

社 总 机: 010-62770175 邮 购: 010-62786544

投稿与读者服务: 010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈: 010-62772015, zhiliang@tup.tsinghua.edu.cn

印 刷 者: 清华大学印刷厂

装 订 者: 北京市密云县京文制本装订厂

经 销: 全国新华书店

开 本: 190mm×260mm 印 张: 47 字 数: 1350 千字
(附光盘 1 张)

版 次: 2012 年 8 月第 1 版

印 次: 2012 年 8 月第 1 次印刷

印 数: 1~4000

定 价: 89.00 元

前 言

相信很多读者都有“书本上学到的编程知识与实际开发中的应用相差很大”的感受。在软件的开发实践中，怎样实现一些功能或如何解决一些问题很多时候都是无法从教科书上找到答案的。本书正是面向 Visual Basic 编程和开发实践，直接介绍如何实现软件中的实际功能，直击编程时的各种疑难问题，解决难题，帮你铺路！

谁说 Visual Basic 不如人？

笔者见过不少程序员，他们讨论最多的是使用什么语言来编程，使用什么语言做开发，似乎他们对使用何种语言很敏感：使用“时髦”的语言似乎能提高自己的技术含量，使用“传统”的语言似乎有碍自己的开发水平。然而事实是这样吗？计算机语言的发展相当迅速，许多崭新的程序设计语言不断涌现，现有语言之间的竞争又此起彼伏，今年某某语言比较流行，在排行榜名列前茅，而明年这种语言又可能被其他语言所打败。许多程序员就在语言的选择之间辗转，刚刚接触到一门语言，还没来得及彻底搞懂，就发现这种语言“过时”了，更新的语言出现了，于是就转向另外一种语言；而新的语言没学多久，发现又有了更新的语言……他们总是把焦点放在选用何种语言上，频繁地在语言的切换中追赶所谓的“时髦”，虽然“了解”（不能达到掌握）的语言不少，可是没有一种能够精通，也没有一种能够得心应手地使用，真正地把它应用到解决实际问题的程序设计中。

不可否认，一款好的程序设计语言会提高编程的效率，结构清晰、语法严谨、支持面向对象等的特性不仅可以使程序的开发事半功倍，而且使编译后的可执行程序的运行效率也会提高。然而使用一种“最新”的语言就意味着这个程序员能力很强、很优秀吗？我也见过不少真正很优秀的程序员，或者真正的编程“高手”，他们对待语言种类的选择是很淡漠的。他们经常说使用何种语言开发都是一样的，事实上他们也完全有能力在很短的时间内快速地掌握一门新的语言，并能很快地将这种语言使用到自己的项目开发中。这种能力不是所有程序员都具有的，那么这种能力来自何处？这些编程“高手”都有一个共同点：虽然可以掌握数种语言，但他们都有一种或少数几种是自己最“精通”的，是自己用起来最得心应手的。因此“高手”的能力来自于对一门语言的精通，来自于对一门语言的彻底掌握和大量的实践经验！因为不同的计算机语言虽然在语法上会有这样或那样的不同，但在编程思路和算法设计上都有许多相同之处：当彻底地掌握了一种语言的时候，再遇到一种新的语言，只需要了解一下它的语法，再有一本函数手册，不难立即把它用起来！

因此，笔者这里想建议的是，我们不要把过多的精力放在考察何种语言是最新的、何种语言是最“时髦”的、或者每年排行第几上，选择自己喜欢的一种语言，彻底掌握它，然后把精力放在如何使用它提高自己真正的编程能力。再大量地基于这种语言编程实践，假以时日你的编程能力就会达到一个很高的层次，到时再掌握任何一种新的语言就都不是一件难事！

Visual Basic（以下简称 VB）这种程序设计语言，真可谓是“一波多折”，它受到过许多人的青睐，同时也受到过许多人的“白眼”。VB 具有许多优点：比如简单易学、容易上手、封装性好、安全、胜任快速开发、是个人独立开发的首选。这些优点足以成为我们使用 VB 的理由。因为在当今时代，效率就意味着竞争，一种胜任快速开发的语言就代表着高效率。

比尔·盖茨曾自豪地说：“世界上有 80%的人在用 VB 编程!”，“微软肯定大力发展 VB，VB 当之无愧是 Windows 平台上最好的开发软件”。

然而事物都有两面性，任何一种语言在具有它自身优点的同时，都势必会具有另外的一些缺点，VB 也不例外。于是许多人抓住 VB 的缺点大做文章，不少人把 VB 冠以“弱语言”、“玩具语言”等的帽子，他们认为“编 VB 的人太没水平”、“VB 没有指针，不能实现复杂的数据结构”、“VB 功能太弱，不可能写出系统级的程序”……其实大多这样的人一般都是使用其他语言编程的，对 VB 仅有少数的了解或者根本没有了解，他们并没有深入地去研究过 VB，得出这样的结论是不是过于武断？VB 最大的不足是系统提供的功能少，直接由 VB 开发出的软件不如由 Visual C++等语言开发出的功能强大，这一点笔者也不否认，但这只是表面现象，难道真的很多功能用 VB 做不出来吗？笔者在此要强调的是：VB 同样很强大，其他语言能够开发的功能，VB 照样能做！本书正是为了增强 VB 的功能、弥补 VB 这个所谓“缺点”为目的的，让喜欢 VB 的读者在当前众多语言的竞争中，仍然能够把 VB 用起来，发挥出 VB 应有的生命力！当读完本书后你一定会发现，VB 并不弱！

怎样才能让 Visual Basic 强大起来？

Visual Basic 从诞生之日起已经经历了数个版本的变化，直到现在 Visual Basic.Net 的出现。Visual Basic.Net 具备了面向对象的绝大部分特性，使代码更加方便维护、可扩展性好、支持代码重用等，而且在语法上也做了很多非常有益的改进。因此，有不少 VB 程序员转向 VB.Net。然而，VB.Net 与 VB 相比较有了非常多的变化，并不是 VB 语言的简单升级，而可以看作是一种崭新的语言，新的 VB.Net 中有许多内容需要重新学习。VB.Net 的程序与 VB 的程序也有很多不兼容的成份，很多 VB 程序并不能直接升级为 VB.Net，而需要重新编写，这对于先前使用 VB 开发的大型系统的升级和维护来说无疑是一场灾难！另外，使用 VB.Net 开发的程序需要用户安装 .Net Framework 才能运行，程序发布时就不得不带有比较庞大的安装包，这为使用 Windows XP 或以下操作系统的用户带来许多不便。再有，使用 VB.Net 开发的程序编译后是“中间代码”，而不是传统意义上的标准 Windows 可执行文件，这也不能满足很多的开发需求。因此，在当前阶段，尽管 VB.Net 蓬勃发展，而且代表了未来的一种发展趋势；但就目前形势来看，VB.Net 仍无法取代 VB，VB 语言仍有它的一席之地。总之，要增强 VB 的功能，或者要解决在 VB 程序设计中遇到的问题，仍然需要通过 VB 来解决，而 VB.Net 是帮不上忙的。因此，本书并不打算介绍使用 VB.Net 开发程序，而把主要问题仍然放在 VB 上。

那么怎样才能依靠 VB 自身，增强 VB 的功能呢？VB 支持 Windows API 函数的调用，这就为 VB 具有“无所不能”的本领打下了坚实的基础，因为所有运行在 Windows 系统下的程序包括 Windows 本身的运行，都是依靠调用这些 API 函数的。也就是说，无论使用任何语言编写程序，无论语言本身提供的功能有多少，程序经过编译之后的可执行文件中的机器代码都是用于调用这些 API 函数。例如，Visual C++很强大，其中的 MFC 库就是封装的 API 函数；使用 Win32 汇编语言开发 Windows 程序，仅使用汇编语言的基本指令，甚至可以通过直接调用 API 函数，做出大型的软件！在本书第二章的内容中还要介绍在 VB 中实现指针，有了指针+API 函数，VB 就真的可以无所不能！

本书与其他书籍最大的不同之处在于，本书不仅介绍具体功能的实现技术和具体问题的解决方法，还讲解在实际编程时尤其是在开发大型软件时必须考虑的细节和必须处理的有可能存在潜在隐患的细微问题，使这些知识直接可以应用于开发实践，减少理论与实践的“脱

节”。尤其要介绍如何将实现这些技术的程序代码“封装”成通用的模块，使这些通用模块在以后的编程中可以得到重用。这样，有关 API 函数的调用和程序的实现细节只需要考虑一次，在以后编程中需要某项功能时，只要把通用模块添加到工程中，然后直接调用模块中的 1~2 个函数即可轻松实现。通过编写通用模块，不仅大大增强了 VB 程序的功能，而且减少了以后编程的工作量，程序代码也更为简洁、便于维护。

本书所介绍的通用模块大多数具有如下特点：

(1) 封装性强：无论标准模块还是类模块都自成一體，具有较强的独立性和内聚性。主要表现在使用局部变量和局部 API 声明、合理设计模块内函数的功能和参数、对其他模块的依赖很少，像子类操作、函数指针、对象地址、对象关联、资源释放等复杂的操作都被封装在模块内部。今后在开发新的程序时，根本不必考虑这些细节，也不必干预模块内的代码，而只需把模块当做“黑匣子”添加进工程，再通过几个简单的函数调用即可实现功能。

(2) 使用方便：通用模块在实现功能时，几乎不需要添加任何引用或任何的 ActiveX 控件，而仅依靠 VB 基本的功能或 API 函数实现。因此模块对各种组件或第三方控件几乎没有依赖，它们只是纯粹的文本代码；也就是说，只要在工程中添加几个纯文本的代码模块即可实现 VB 程序功能的强大提升。此外，模块的设计本着让主调程序调用负担最低的原则，模块编写好后，主调程序为调用模块功能而编写的代码极少。

(3) 功能强大：模块中大量调用了 Windows 的 API 函数，并且使用了指针，很多程序还涉及了 Windows 的核心。可以说 Windows 能做的，VB 都能做！

(4) 可重用性强：模块可以被添加到任何工程，又可被用于工程中的任何窗体、模块和类模块中。

(5) 便于开发绿色软件：本书所介绍的通用模块一般属于类模块和标准模块，还有少量窗体模块，而本书不准备介绍任何 ActiveX 控件的开发。因此使用本书所介绍的通用模块编写 VB 程序，很少或不需任何额外的引用或 ActiveX 控件的支持，在程序发布和安装时也不必包含过多的系统文件或注册组件。实际上使用这些通用模块所开发的应用程序在 Windows 2000 及以上系统可以直接运行，根本不需要打包安装程序和制作 setup。

开发通用模块，实际上是丰富了 VB 程序员可以使用的函数库和类库，这相当于是在 VB 的基础上搭建了一套自己的开发环境。而这一套开发环境提供的功能，有很多比其他语言提供的还要强大，但使用起来却比其他语言还要容易。这就使得用 VB 语言开发的程序不仅不逊色于其他语言，而且更有着 VB 简单易学、容易上手的特点作为支持，在开发速度上也会比使用其他语言还要快。

本书适合的读者

本书是介绍使用 Visual Basic 做高级开发的书籍，适合对 Visual Basic 已经有了基本的掌握，想进一步提高 Visual Basic 编程能力，以及在编程中遇到一些令人头疼的问题而正在寻找解决方案的读者。对使用其他语言开发 Windows 程序的读者来说，书中的许多方法和编程技巧也会对他们有一定的参考价值，因为无论用什么语言开发，Windows 的运行机制是不变的，程序的设计思路也都是可以借鉴的。本书假定读者已具有以下知识：

- 对 Visual Basic 已经有了基本的掌握，如常用控件的使用、程序结构、变量、语句、函数、对象的属性、事件、方法等，已经能够用 Visual Basic 编写简单的程序，因为本书不会涉及这些基本的 Visual Basic 编程方法；
- 计算机基础知识的一般了解，如进制转换、内存、指针等；

- 最基本的 C 语言的基础知识，因为 Windows API 函数是用 C 语言编写的，为了能够读懂参考资料，读者需要了解一些最基本的 C 语言知识；但读者不需要完全掌握 C 语言，只需要了解可能用到的 C 语言常量、变量、数组、结构体、指针等数据类型和简单的函数结构。

关于本书的内容

本书分为三个篇章：

第一篇：高级技术篇

本篇介绍使用 Visual Basic 进行高层次软件开发和解决实际问题的技术，这些技术在本书后续章节的内容中也会被大量用到。例如：如何在 VB 中使用指针、开发常用的数据结构模块、开发实现子类技术的通用模块、Windows 窗口、控件、消息的运行机制、如何通过 API 函数创建控件、如何开发支持多国语言版的程序等。在本篇章中还将介绍如何解决 VB 编程时一些常见的并很棘手的问题。这些技术大多是以编写为“通用模块”的形式实现的，使技术细节只需考虑一次，在以后的编程中可以直接把“通用模块”当做“黑匣子”来调用。

第二篇：功能模块篇

本篇介绍如何在 VB 中编写一些通用的功能模块，以帮助开发人员实现 VB 系统不曾提供的、但开发软件时却常需要实现的重要功能，并胜任快速开发：如增强图形界面、菜单高级控制、文件操作、注册表操作、系统热键、系统托盘图标等。这些模块大多也可以看作是对 API 函数的封装，实现这些功能的 API 函数都被自己编写的、调用起来更为简单的简单函数所替代。这在以后的编程中，再不必考虑细节，并简化程序设计，大大减少编程工作量，提高开发效率。

第三篇：应用模块篇

有一句格言讲到：“一个具体的问题胜过数千个抽象的方案”。本篇就针对在软件开发时可能遇到的具体编程问题，每章开发一个通用模块，实现软件中一个常见的功能。限于篇幅，本篇只介绍有代表性的几种功能的通用模块实现，如智能组合框、定时器、进度指示工作助手、控件的边框调整、气泡提示、表达式计算等。希望读者能够举一反三，并自己开发出更多的通用模块，使这些模块能够像“积木”一样被积累，在需要时可随时被添加到 VB 工程中使用，为自己的 VB 编程实践和软件开发添砖加瓦。

读过本书后，相信广大的 Visual Basic 编程人员和 VB 爱好者，尤其是使用 Visual Basic 语言做软件项目开发的读者，一定会有所收获。

致谢

首先要感谢南开大学生命科学学院张涛教授、数学院阮吉寿教授以及医学院杨卓教授对本书写作的大力支持，感谢三位恩师多年来在学习和生活上无微不至的关心和帮助，他们活跃的科研思维和严谨的工作态度一直深深影响着我。

感谢我的父母、岳父岳母和家人，他们总是默默地支持我，生活上悉心地照顾我，为我分担压力。特别要感谢我的妻子佳佳，为了能按时完稿，她牺牲了逛街的时间，放弃了蜜月旅行，在写作的过程中一直陪伴着我，给我鼓励、勇气和信心。

由于笔者水平有限，错谬之处在所难免，恳请专家和广大读者不吝赐教、批评指正。笔者的 E-mail 是：zhni2011@163.com。

目 录

第一篇 高级技术篇

第 1 章 万丈高楼平地起，一劳永逸 打地基——知识准备	2
1.1 使用 Windows API 函数	2
1.1.1 API 函数的声明	2
1.1.2 使用 API 函数的注意事项	6
1.2 模块化编程	7
1.2.1 我能在一天之内做出一个软件吗	7
1.2.2 用标准模块还是类模块	7
1.2.3 怎样创建类模块	8
1.3 编程风格	12
1.3.1 变量在使用前一定要定义	12
1.3.2 尽量不要用 Variant 数据类型	12
1.3.3 代码的缩进和换行	13
1.3.4 避免重复用对象名称做一串调用	13
1.3.5 Boolean 型条件表达式的写法	15
1.3.6 字符串为空串的判断	15
1.3.7 能用常量就不要用函数求值	15
1.3.8 数组下标从 1 开始	16
1.3.9 用 Optional 选项定义函数的参数	16
1.3.10 不能滥用 Goto 语句不代表不能用 Goto 语句	17
1.3.11 尽量少用 ActiveX 控件	17
1.4 条件编译	18
1.4.1 什么是条件编译	18
1.4.2 条件编译有什么用	18
1.5 获取资料	20
1.5.1 MSDN	20
1.5.2 借助 Visual C++ 的头文件	22
1.5.3 其他资料	23
第 2 章 这个可以有——在 VB 中使用指针	24
2.1 在 VB 中使用指针变量	24
2.1.1 什么是指针	24
2.1.2 如何实现指针变量	24
2.2 VarPtr, StrPtr, ObjPtr 函数和 AddressOf 运算符	25
2.2.1 内联类型和指针类型	25

2.2.2	VarPtr, StrPtr 和 ObjPtr 函数	27
2.2.3	AddressOf 运算符	27
2.3	CopyMemory 函数	28
2.3.1	CopyMemory 函数的声明和功能	28
2.3.2	CopyMemory 函数用于内联类型的数据传递	28
2.3.3	CopyMemory 函数用于字符串类型的数据传递	29
2.3.4	对象的弱引用	30
2.4	在 VB 中使用指针程序举例	32
2.4.1	使用指针实现不同类型数据的交换	32
2.4.2	使用指针实现不同窗体模块的数据交换	33
2.4.3	不使用数组类型的参数向函数传递数组	34
2.4.4	用一个 Long 型参数向函数传递任意类型任意大小的数据	35
2.5	部分字节运算的问题和解决办法	36
2.5.1	Or 按位或运算	36
2.5.2	取长整数的高位和低位	40
第 3 章	不做数据的奴隶——常用数据结构	42
3.1	何谓数据结构	42
3.2	堆栈	43
3.2.1	什么是堆栈	43
3.2.2	堆栈的基本算法	44
3.2.3	堆栈的通用模块实现	44
3.2.4	堆栈通用模块的使用	51
3.3	哈希表	53
3.3.1	什么是哈希表	53
3.3.2	哈希表的基本算法	54
3.3.3	哈希表的通用模块实现	55
3.3.4	哈希表通用模块的使用	75
3.3.5	哈希表类模块与 VB 的 Collection 对象的效率比较	77
3.4	带“回收站”的数组	79
3.4.1	基本思想	79
3.4.2	程序实现	79
第 4 章	青出于蓝胜于蓝——子类技术	84
4.1	Windows 窗口和消息驱动机制	84
4.1.1	窗口	84
4.1.2	窗口的消息驱动机制	85
4.1.3	消息的组织和发送	87
4.1.4	Visual Basic 事件与消息	89
4.2	子类技术及其常规实现	89
4.2.1	子类技术简介	89

4.2.2	子类技术的常规实现	91
4.2.3	子类技术常规实现的问题	96
4.3	开发实现子类技术的通用模块	97
4.3.1	通用模块的目标	97
4.3.2	封装子类信息的类模块	98
4.3.3	实现子类技术的标准模块	106
4.3.4	子类技术通用模块小结	118
4.4	子类技术通用模块的用法举例	119
4.4.1	标准模块 Module1 的设计	120
4.4.2	窗体 Form1 的设计	121
4.4.3	程序的运行效果	122
第 5 章	想解雇 ActiveX 通用控件吗——使用 API 函数创建控件	124
5.1	控件的深入理解	124
5.1.1	ActiveX 控件	124
5.1.2	dll 通用控件库	125
5.1.3	控件的运行机制	128
5.1.4	简单实例——用 API 函数创建 Animation 控件	136
5.2	用 API 函数创建控件的模块划分	140
5.2.1	实现控件的类模块	140
5.2.2	支持的标准模块	143
5.2.3	其他“黑匣子”模块	144
5.2.4	模块划分小结	144
5.3	mdlAPIControlsSupport 标准模块	145
5.3.1	公有常量定义	146
5.3.2	控件类型的枚举类型定义	147
5.3.3	管理已创建控件	149
5.3.4	子类处理的自定义窗口程序	151
5.3.5	标准模块的其他内容	156
5.4	clsControlBase 基类模块	157
5.4.1	API 函数声明和常量、类型定义	157
5.4.2	创建控件	159
5.4.3	实现控件的共有属性	161
5.4.4	实现控件的共有方法	176
5.4.5	类模块的 Initialize 和 Terminate 事件过程	181
5.5	实现跳格表控件的控件类模块 clsAPITabs	182
5.5.1	API 函数声明和常量、类型定义	183
5.5.2	继承的“变通”实现	184
5.5.3	创建跳格表控件	185
5.5.4	实现跳格表控件的属性	188
5.5.5	实现跳格表控件的方法	190

5.5.6	实现跳格表控件的事件	193
5.5.7	clsAPITabs 控件类模块的应用实例	196
5.6	实现状态栏控件的控件类模块 clsAPIStaBar	198
5.6.1	API 函数声明和常量、类型定义	199
5.6.2	继承的“变通”实现	200
5.6.3	创建状态栏控件	200
5.6.4	状态栏分栏和分栏属性	202
5.6.5	状态栏的其他功能	210
5.6.6	实现状态栏的事件	211
5.6.7	clsAPIStaBar 控件类模块的应用实例	212
5.7	实现丰富格式文本框控件的控件类模块 clsAPIRichTextBox	215
5.7.1	RichTextBox 控件加载和继承的“变通”实现	215
5.7.2	创建 RichTextBox 控件	216
5.7.3	管理文本和 RTF 代码	219
5.7.4	选择区域	227
5.7.5	文本的剪切、复制、粘贴	229
5.7.6	设置文本格式	229
5.7.7	实现 RichTextBox 的事件	232
5.7.8	实现 RichTextBox 的其他功能	233
5.7.9	clsAPIRichTextBox 控件类模块的应用实例	234
第 6 章	路见不平一声吼, 强大功能我也有 —— 常见 VB 系统功能问题和解决	237
6.1	长字符串处理	237
6.1.1	VB 系统中的字符串连接运算及问题	237
6.1.2	解决方案——clsLongString 类模块	240
6.1.3	改进字符串连接效率测试	250
6.2	多种分行格式的纯文本文件的按行读取	251
6.2.1	VB 按行读取文本文件的常规方法及问题	251
6.2.2	二进制读取文本文件	253
6.2.3	设计类模块支持多种分行格式	255
6.2.4	类模块的使用	274
6.3	部分字符串函数的增强	275
6.3.1	Split 函数的增强	275
6.3.2	Trim 函数的增强	281
6.3.3	判断 Collection 对象的键是否存在	283
6.4	解决控件不支持鼠标滚轮的问题	284
6.4.1	鼠标滚轮消息 WM_MOUSEWHEEL	285
6.4.2	窗口滚动消息 WM_HSCROLL 和 WM_VSCROLL	285
6.4.3	解决鼠标滚轮问题的通用模块	287
6.5	增强 VB 的滚动条控件	289
6.5.1	基本思路	290

6.5.2	实现技术	291
6.5.3	类模块的完整代码	297
6.5.4	增强滚动条通用模块使用举例	303
第 7 章	让你的程序在哪都不 out——使程序支持多国语言版	308
7.1	多国语言支持的主要问题	308
7.2	字符串资源和资源加载	309
7.2.1	字符串资源	309
7.2.2	VB 的资源编辑器	310
7.2.3	解决不同语言语法元素顺序问题	311
7.3	编写支持多国语言版程序的通用模块	313
7.3.1	公有函数	313
7.3.2	条件编译的全局元素	314
7.3.3	开发多国语言版的程序小结	316
7.4	支持多国语言版的程序实例	317
7.4.1	简单实例	317
7.4.2	改造 clsReadLines 类模块	319

第二篇 功能模块篇

第 8 章	程序骨感没人爱——Windows 绘图和图形界面增强	324
8.1	GDI 的基本原理	324
8.1.1	色彩和坐标	324
8.1.2	设备环境简介	327
8.1.3	GDI 对象	330
8.2	高级文本描绘的类模块 clsDrawText	335
8.2.1	DrawText 函数	335
8.2.2	clsDrawText 类模块的开发	338
8.2.3	clsDrawText 使用的实例程序	349
8.3	绘制 Windows 图形元素	352
8.3.1	绘制蚀刻线	352
8.3.2	绘制 3D 效果的边框	354
8.3.3	绘制 Windows 按钮元素	356
8.3.4	绘制首尾移动样式的按钮	361
8.4	绘制透明位图	363
8.4.1	位图简介	364
8.4.2	光栅操作	366
8.4.3	绘制透明位图	367
8.4.4	程序实例	371
8.5	为静态图形添加图形热区	372
8.5.1	图形热区实现的基本思路	372
8.5.2	开发图形热区管理类模块 clsGraphCoords	373

8.5.3	图形热区编程实例	381
8.6	改变窗体透明度	383
8.6.1	设置窗体透明度的 API 函数	384
8.6.2	编写类模块封装设置窗体透明度的 API 函数	384
8.6.3	程序实例	388
8.7	使控件具有 Windows XP 风格的外观	391
8.7.1	XP 风格外观的控件探秘	391
8.7.2	使应用程序中的控件具有 XP 风格	393
8.7.3	程序实例	397
第 9 章	搞定这个怕什么? 你懂我的——菜单高级控制和菜单选择助手	398
9.1	编写菜单控制标准模块 mdlMenuFuncs	399
9.1.1	菜单的句柄	399
9.1.2	获取菜单项信息	402
9.1.3	设置菜单项信息	406
9.1.4	设置菜单项高亮状态	412
9.1.5	返回或设置子菜单默认项	412
9.1.6	获得菜单中的菜单项总数	413
9.1.7	系统菜单控制	414
9.2	菜单选择助手	415
9.2.1	基本思路	416
9.2.2	在 mdlMenuFuncs 中添加的完整代码	419
9.2.3	clsMenuSelAssist 的完整代码	420
9.3	程序实例	423
9.3.1	菜单选择助手功能演示	423
9.3.2	标准模块 mdlMenuFuncs 部分功能演示	423
第 10 章	玩转注册表, 换汤不换药——简化 ini 文件和注册表编程	426
10.1	ini 文件和使用 ini 文件保存信息	426
10.1.1	ini 文件的结构	426
10.1.2	ini 文件读写的主要 API 函数	427
10.1.3	封装 ini 文件的读写操作	428
10.1.4	ini 文件读写的实例程序	429
10.2	注册表结构简介	429
10.2.1	控制项	430
10.2.2	子项	432
10.2.3	键值	432
10.3	编写通用模块封装常用注册表读写操作	433
10.3.1	项的句柄和项的打开、关闭	433
10.3.2	固定子项位置	435
10.3.3	创建或删除子项	436

10.3.4	读取子项键值	438
10.3.5	设置子项键值	440
10.3.6	删除子项键值	442
10.3.7	读取注册表任意位置数据	443
10.3.8	设置文件关联	444
10.4	注册表编程实例	448
10.4.1	记忆窗体位置	448
10.4.2	在固定子项下读写数据	448
10.4.3	建立和删除文件关联	449
第 11 章	真的很给力——文件系统编程	451
11.1	文件系统操作	451
11.1.1	列文件目录	451
11.1.2	判断文件（夹）存在	457
11.1.3	文件名操作函数	461
11.1.4	创建文件夹	466
11.1.5	文件的复制、移动和删除	467
11.1.6	获得驱动器信息	470
11.1.7	获得临时文件	475
11.2	获得文件属性	477
11.2.1	文件名属性	477
11.2.2	文件大小和文件属性	478
11.2.3	文件时间属性	480
11.2.4	文件属性对话框	483
11.2.5	文件类型字符串和文件图标	484
11.3	打开/保存文件对话框的通用模块	491
11.3.1	打开/保存文件对话框的相关 API 函数	491
11.3.2	编写通用模块	494
11.3.3	通用模块使用实例	503
11.4	浏览文件夹对话框的通用模块	505
11.4.1	浏览文件夹对话框的相关 API 函数	505
11.4.2	编写通用模块	507
11.4.3	通用模块使用实例	509
11.5	制作自己的简易资源管理器	509
第 12 章	非“程”勿扰——其他常见系统功能	518
12.1	运行应用程序	518
12.1.1	运行程序或打开文档	518
12.1.2	运行程序并等待程序结束	522
12.1.3	程序实例	528
12.2	监视 Windows 剪贴板	530

12.2.1	剪贴板查看器和查看器链	530
12.2.2	使 VB 程序具有剪贴板监视功能的通用模块	532
12.2.3	剪贴板监视程序举例	536
12.3	设置系统热键	537
12.3.1	设置系统热键的有关知识	538
12.3.2	实现系统热键通用模块的完整代码	541
12.3.3	程序实例	548
12.4	添加系统托盘图标	550
12.4.1	相关 API 函数	550
12.4.2	实现系统托盘图标通用模块的完整代码	552
12.4.3	程序实例	557

第三篇 应用模块篇

第 13 章	我是勤奋的“猪”，我很乖——智能组合框	562
13.1	智能列表项维护	562
13.1.1	类模块对象与组合框控件的关联	562
13.1.2	不重复地添加列表项	563
13.1.3	列表项的保存和加载	565
13.2	实现按汉字拼音字头的中文自动输入	568
13.2.1	实现原理	568
13.2.2	获得汉字的拼音字头	570
13.2.3	查找拼音字头	572
13.2.4	实现按拼音字头自动输入	573
13.2.5	自动提示	575
13.3	智能组合框的其他功能	578
13.3.1	返回或设置下拉列表是否被拉下	578
13.3.2	设置下拉列表的高度	579
13.3.3	获取或设置下拉列表的最小宽度	579
13.4	智能组合框应用实例	580
第 14 章	“时”上编程——用 API 函数实现定时器	583
14.1	用 API 函数创建定时器的通用模块	583
14.1.1	定时器简介	583
14.1.2	使用 API 函数创建定时器	584
14.1.3	定时器支持模块 mdlTimerSupport	585
14.1.4	定时器类模块 clsTimer	587
14.2	定时器通用模块应用实例	590
14.2.1	类模块内的定时器使用	590
14.2.2	标准模块内的定时器使用	591
14.3	通过定时器变通实现多线程	592

第 15 章 谁说这事不能说太细——进度指示	595
15.1 编写长时间运行程序需注意的主要问题.....	595
15.2 使程序在运行途中响应用户按下【取消】按钮.....	599
15.3 开发工作助手类模块 clsOperAssistant.....	601
15.3.1 数据定义.....	601
15.3.2 更新进度指示.....	606
15.3.3 显示简短提示文本.....	610
15.3.4 响应用户中途取消.....	611
15.3.5 出错提示.....	613
15.3.6 程序开始和结束.....	613
15.3.7 清除方法 Clear.....	615
15.3.8 类模块的 Initialize 和 Terminate 事件过程.....	616
15.4 工作助手类模块的应用实例.....	616
15.5 开发指示进度窗体.....	618
15.5.1 指示进度窗体的运行效果和调用.....	618
15.5.2 在工作助手类模块中添加驱动窗体的代码.....	619
15.5.3 开发 frmWorkingAvi 窗体.....	621
第 16 章 边界条约用户定——实现运行时控件的边框调整	629
16.1 解决边框调整问题的基本思路.....	629
16.1.1 通用模块的使用.....	629
16.1.2 Frame 控件的作用.....	630
16.1.3 控件大小位置的重新调整.....	631
16.1.4 上下型和左右型风格.....	632
16.2 开发边框调整的通用模块.....	633
16.2.1 控件关联.....	633
16.2.2 有关的数据定义.....	635
16.2.3 重新安排控件大小和位置.....	642
16.2.4 处理分隔线拖动事件.....	644
16.2.5 类模块的 Initialize 和 Terminate 事件过程.....	648
16.3 边框调整通用模块应用实例.....	648
第 17 章 “泡泡”几时有，自己编程瞅——自己编程实现气泡提示框	653
17.1 制作气泡型窗体.....	653
17.1.1 制作气泡型窗体的基本思路.....	653
17.1.2 制作气泡型窗体实例.....	656
17.2 气泡提示框的通用模块.....	658
17.2.1 窗体的界面设计.....	659
17.2.2 气泡样式和图标类型的枚举类型.....	659
17.2.3 属性和常量定义.....	662
17.2.4 显示气泡提示.....	664

17.2.5	指向控件的气泡提示	676
17.2.6	气泡提示的关闭	677
17.2.7	窗体的加载和卸载	677
17.3	气泡提示框程序实例	678
第 18 章	乱“式”英雄——具有高级功能的表达式计算	680
18.1	表达式计算的算法简介	680
18.2	clsCacuExp 类模块的数据定义	682
18.2.1	数据类型	682
18.2.2	表达式“解析”与表达式元素	684
18.2.3	表达式中的常量元素	686
18.2.4	表达式中的变量元素	687
18.2.5	运算符和函数的常量定义	690
18.2.6	表达式字符串属性	696
18.2.7	计算结果属性	697
18.2.8	错误信息属性	698
18.3	clsCacuExp 类模块的功能代码	699
18.3.1	表达式解析	699
18.3.2	表达式计算	713
18.4	clsCacuExp 类模块的使用实例	725
索引		727
参考文献		734