

Visual Basic.NET Power Tools

Visual Basic.NET

工具集

[美] Evangelos Petroutsos 著
Richard Mansfield

高春蓉 朱 军 夏永存 等译
李双庆 审校

- 用.NET安全类使你的应用程序更强大
- 掌握先进的用户接口设计原则
- 学习解决有关数据库连接的疑难问题



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>

Visual Basic.NET Power Tools

Visual Basic.NET工具集

[美] Evangelos Petroutsos 著
Richard Mansfield

高春蓉 朱 军 夏永存 等译

李双庆 审校

電子工業出版社.

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书是一本全面介绍VB.NET的书,但又着重介绍了其他VB书中很少涉及的和深奥难懂的内容。它向读者讲述了数据库编程、程序调试、数据加密术或打印硬拷贝中经常会碰到的问题和这些难题的解决方案。对于VB.NET中的新课题,例如异步编程、Web服务、在.NET应用程序中使用Office对象、利用反射和使用.NET小型框架等问题,书中也做了详尽的分析。书中还包括许多范例程序和大量可重复使用的代码,读者稍加修改或不加修改就可方便地用于自己的项目。

本书是专门为专业程序员而写的,使他们能够通过本书的阅读提高水平,以便处理VB.NET程序设计过程中遇到的许多复杂的、边缘的和高深的课题。本书可帮助那些想从VB6过渡到VB.NET的程序员们用较短的时间快速完成语言转换过程。此外,本书也可以供学习VB.NET程序设计的技术人员、大专院校计算机专业的研究生、大学生作为参考读物。



Copyright©2003 SYBEX Inc., 1151 Marina Village Parkway, Alameda, CA 94501.
World rights reserved. No part of this publication may be stored in a retrieval system, transmitted, or reproduced in any way, including but not limited to photocopy, photograph, magnetic or other record, without the prior agreement and written permission of the publisher.

本书英文版由美国SYBEX公司出版,SYBEX公司已将中文版独家版权授予中国电子工业出版社及北京美迪亚电子信息有限公司。未经许可,不得以任何形式和手段复制或抄袭本书内容。

版权贸易合同登记号: 01-2003-8830

图书在版编目(CIP)数据

Visual Basic.NET工具集/(美)佩特罗索斯(Petroutsos, E.)等著;高春蓉等译.一北京:电子工业出版社,2004.5

书名原文: Visual Basic.NET Power Tools

ISBN 7-5053-9780-X

I. V... II. ①佩... ②高... III. BASIC语言—程序设计 IV. TP312

中国版本图书馆CIP数据核字(2004)第022368号

责任编辑:徐云鹏 陈 双

印 刷:北京天竺颖华印刷厂

出版发行:电子工业出版社

北京市海淀区万寿路173信箱 邮编:100036

北京市海淀区翠微东里甲2号 邮编:100036

经 销:各地新华书店

开 本:787×1092 1/16 印张:38.625 字数:985 千字

印 次:2004年5月第1次印刷

定 价:60.00元

凡购买电子工业出版社的图书,如有缺损问题,请向购买书店调换,若书店售缺,请与本社发行部联系。联系电话:010-68279077。质量投诉请发邮件至zltts@phei.com.cn,盗版侵权举报请发邮件至dbqq@phei.com.cn。

致 谢

我们幸运地遇到了几位精明、考虑周详的编辑，帮助我们打磨和推敲本书的原稿。首先，我们要感谢开发编辑**Tom Cirtin**，他敏锐的洞察力和高质量编辑受到人们的信任。他与他的作者们相处得很好，而且同样善于提出重要的问题和改进他们的章节。

技术编辑**Greg Guntle**仔细地审阅了原稿，提出了许多有用的建议，找到了许多前后不一致的地方，并帮助改进了几个代码范例。制作编辑**Leslie Light**保证了本书生产过程的顺利，最有用的是他提出了一些关于图形、绘图和屏幕照片的建议。复制编辑**Suzanne Goraj**细心地整理过我们书稿的每一行，使它从头至尾得到改进。

我们要感谢以上所有的人和**Sybex**公司其他对本书做过贡献的人们，由于他们的聪明智慧和细心照顾才使本书得以出版。此外，作者在此要特别感谢本书的代理商，**Waterside Productions**公司的**Matt Wagner**和**David Fugate**，他们对作者事业的支持远远超越了他们的职责的范围。

关 于 作 者

Evangelos Petroutsos是一位计算机工程师，他在加州理工大学和**MCI**公司工作期间获得了许多宝贵的经验。他写过许多文章并讲授计算机编程课程，他所编写的由**Sybex**公司出版的**Mastering Visual Basic.NET**、**Mastering Visual Basic 6**和**Visual Basic.NET Developer's Handbook**是市场上的畅销书。

Richard Mansfield为各种杂志写过数百篇文章，并写过32本计算机书。**Richard**的书已经被翻译成11种语言，这些书在全世界的销售量已超过50万册。

作者开发的所有代码范例和一些实用程序读者都可以在下列网址下载：

www.sybex.com。

译 者 的 话

近年来，微软推出的.NET系列产品给Windows应用软件的开发带来了巨大的变化。**Visual Basic.NET**是一种真正的面向对象的语言。虽然**VB.NET**的基本语法与以前版本的**Visual Basic**完全一致，但人人都知道，**Visual Basic.NET**与原来版本的**Visual Basic**并不兼容。如果想使在**VB6**中开发的应用程序能在**VB.NET**中运行，必须对原来的程序进行必要的修改，而且这种修改还不是很容易完成的。过去用**VB6**编程的程序员现在如果想用**VB.NET**编程也必须经过一个不太轻松的转换过程。本书正是为了帮助这些程序员从过去的**VB6**平稳过渡到**VB.NET**而写的，书中针对程序员们可能遇到的问题做了详尽的说明。尤其是书中还配有许多实用程序，许多代码稍加修改或不必修改就可以用于读者自己特定的项目中。

本书的两位作者都是闻名的**VB**书的作者，他们针对其他书中很少涉及的**VB.NET**的重要问题做了详尽、透彻的叙述。在书中，他们不仅称赞**VB.NET**对传统程序设计功能的改进，例如流技术和串行化方法，而且也直率地指出**VB.NET**中存在的问题。在翻译本书的过程中，译者对作者实事求是、严肃认真的治学态度感到敬佩。

本书的第1章到第6章由李双庆翻译，第7章到第13章由夏永存翻译，第14章到第17章由高春蓉翻译，第18章到第22章由朱军翻译，全书由李双庆审校。在翻译过程中，彭苏鲁、方志英、刘求真、姚远、龙小波、李春子、许柏庆、刘文林、丘英、卢刚、马文清、张亚宁、李苏云、刘毅兵、甘壁文、张法通、赖庆来、高春波等也做了部分翻译和校对工作，在此对他们表示真挚的谢意。

译 者

前 言

这本书实际上是1993年在希腊的首都雅典开始写的。Evangelos Petroustos为一本关于人们可以用Visual Basic做哪些“吸引人的和复杂深奥的”事情的书写了个非常有趣的提纲和几个样章。我同意出版商的意见，他的想法很有潜力，但Evangelos是第一次写书。但我有写书的经验，出版商说，如果我同意共同编写这本书的话，他们将为这本“功能强大的工具包”的书投资。对于一个出版社来说，即使是一本小的书也需要50 000美元的冒险投资，何况这是一本很大的书。

我高兴地同意了，因为我觉得这个题目很引人注目——不规则碎片、加密术、处理图形、动画的过渡、多媒体、操作颜色调色板、递归和其他课题，这些都是在大多数其他VB的书中被忽视的问题。令我们高兴的是，这本书居然成了1995年最畅销的书。很显然，许多Visual Basic程序员都愿意手头有一本关于高级、最新编程技术的书。

在2002年，我们决定重新回到这个概念上来。几乎已经过去了十年的时间，现在，我们有了一个崭新的Visual Basic语言：VB.NET。我们决定沿着我们十年前走过的路走下去：去探索在其他VB.NET书中被大量忽视的、但又十分有用和有趣的那些问题。

大部分话题是十年以前的那本书中已经涉及过的，现在在这里又重复提到——时代已经变了。但我们感觉到，这本新的书里所探讨的课题引人注目是有它们自己的理由的。

美学

为什么如此吸引人的话题会在计算机书籍中被大量地忽视呢？我们认为主要有两个原因。第一类被忽视话题是一些看起来“价值不高的”或者“边缘的”的话题。换一种方式说，这些题目都涉及到美学。程序设计者们总的说来都宁愿把自己看成科学团体中的一部分，因此研究像美丽形象或好的外观等问题，在许多程序设计者看来好像降低了他们的身份。虽然如此，在本书中有两章大胆地探讨了关于美学的课题。

说实在的，程序设计是一种艺术，不是一门科学。有些教授像变戏法似地变出了理论上的结构和特殊的术语学，但是，空洞的说教和夸张高深的行话并不能自己创造出一门科学——实际上，这一切往往约束了理性的言论。

研究已经表明，最好的程序设计师经常是主修英语或音乐的人。当某些最好的开发者购买了他们自己的第一台Amiga计算机时（一种专门用于创造性计算方面的早期的计算机），他们也进入了编程领域。虽然理论的程序设计通常与数学结合在一起，可数学（或科学）与编程之间几乎没有真正的关系——正像通常许多其他的理论研究与现实世界之间几乎没什么关系一样。

想一想现在主要流行的计算机应用程序：字处理、数据库管理、因特网通信和电子表格。只有电子表格大量地需要用数学。当然，编程可能涉及到数学，但是数学很少是程序员工作的中心。用户即使不知道长除法、知道很少代数学和任何其他数学，也可以写出一个完

整的字处理程序来。

程序设计显然不是一门科学。科学涉及到建立理论和受到实验的控制，科学行为很少与程序设计有关。当然，当纠正程序错误的时候也需要做某种试验，但无论有怎样的想像力也能看出，那不是科学实验。程序调试更接近于搜索一套丢失的钥匙，而不是把风筝送上天空送到暴风雨中去。

程序设计基本上就是沟通——不过是人与机器之间的沟通。但它是一种语言和表现的行为。确切地说，它是不带修辞色彩的（我们不需要说服机器，至少现在还不需要），但它肯定是描述性的、符合语法的和基本的沟通。

本书中被某些人认为是“不科学的”两章是，第21章和第20章，我们同意。可是我们认为整个的程序设计都是非科学的，我们并没有被那个事实所打扰。

复杂性和激进的先锋

本书中剩下的大多数问题都属于第二类：不是非常边缘的就是过份复杂的问题，包含这些问题对许多书都不适合。例如，许多作者没有写到VB.NET杰出的和使用广泛的安全功能——即使安全问题是计算机界面临的一个主要的挑战。

避开与安全性有关的VB.NET编程技术并不是因为涉及的编程方法本身十分困难或过于新颖，而是因为加密术背后的概念和安全性的其他方面相当复杂。许多计算机书的作者对加密术本身就不具备足够的知识，例如，解释在计算机编程技术中安全性的实现。幸好，本书的一位作者长时间以来就对安全性问题有特殊的爱好。

其他的题目也许太新，很难被广大的读者理解或实现。异步编程、Web服务、使用Office对象、使用反射和新的.NET小型框架（怎样把编程和I/O挤压到一个非常受限制的小型便携式设备的平台中，例如个人数字助理和移动电话这类设备），全都属于这个范畴。

我们承认，在本书中有几章涉及的话题是其他书中也广泛涉及的（数据库编程、程序的调试、打印），不过，本书写这些内容是因为我们有新的东西要告诉读者。例如，到目前为止，我们还没有发现任何一本书正确地描述在VB.NET中怎样打印硬拷贝的。我们看到过的所有的编程范例，不是在一些行的末尾切掉半个字母就是在某些页的最后切掉半行。这种现象并不是发生在每一行或者每一页的最后，但你一定会同意，即使这种情况断断续续地出现也是够糟糕的了。如果想寻求解决这个问题的办法，请阅读第7章。

第1章是不寻常的，因为它抓住了一个本质的，但至今人们还在普遍回避的问题：为什么Visual Basic.NET是由C程序员设计的，这意味着什么？这就好像罗马人接受了任务，让他们重新改建底比斯（埃及尼罗河畔的古城）一样——结果可能是令人印象深刻的，但这个印象肯定不会留给埃及人。

第1章是这样开头的：

Visual Basic.NET并不是Visual Basic程序员写的，全部.NET系列语言都是C程序员创建的。C和它的同伴——面向对象的程序设计（OOP）是一种学术语言；而Visual Basic是一种通俗流行的语言。这些事实是有重要意义的。

本书的作者并没有受惠于任何机构和组织。我们既不是为微软出版社（Microsoft Press）写书，也不附属任何公司或学校，我们想，我们不依靠任何人给我们发薪水（除了你，我们亲爱的读者之外），所以我们才能比许多其他的同事更加客观。

我们可以提出异端的问题，例如，为什么像OOP的许多支持者所坚持的那样，OOP应当用于一切程序设计场合。我们可以怀疑，让C程序员写VB.NET帮助系统中的叙述和代码范例，这种做法是否明智。我们可以惊讶，如果OOP专家们坚持用户决不当使用的结构，为什么在VB.NET会包括这种结构。

当VB.NET改进了传统的VB程序设计的功能时（例如，流技术和串行化方法），我们可以为它自由地拍手喝彩；而当VB.NET造成了不必要的混乱时，我们可以直率地指出（在VB.NET中有些集合是基于0的；有些是基于1的。毫无道理，为处理这个问题，没有模式可发现，也没有规律可学习）。

置身于程序设计和学术官僚作风之外的另一个好处是，我们可以保持清醒。人们围绕着编程使用许多难懂的术语，这些术语中的许多除了保护作用外并没有真正的用途。如果其他人读不懂你的源代码，或者甚至不理解你的注释，那他们可能就会尊重你，而你就能保住自己的工作。同样地，如果你追随着同伙的队伍，不断地说着笑话，你就能继续保持在这个队伍里。所以，通常只限于少数人参加的牧师类小团体形成。记住，仅在很短时间之前，人们还用拉丁语说话，这种语言连经常做礼拜的人都不能理解。如果今天去参观大学里面一个学习音乐理论或电影理论的班级，他们说的大部分内容你都不能领会。

在这本书里，当我们偶尔沉湎于技术术语中的时候，我们通常都会用最新的广告用语使读者明白术语的意义。的确，本书使用了“超载签名（overloaded signature）”这个术语，但在它下面紧接着就在括号中解释（多个参数列表），这样读者就会知道，正在讨论的到底是什么东西。当术语（例如“强力键入（strongly typed）”）有多个不同含义的时候，我们会给读者指出。

由于作者独立自主的状况，没有服从某家公司或社会团体的义务，因此还有最后一个好处：我们的书可以是令人愉快的，引人入胜的，或者至少比普通的计算机书更少些无聊的内容。理论文章和书籍（包括许多编程的书）总是故意地回避有趣的和令人感兴趣的写作方式。在某些圈子里的人看来，如果你写的东西不那么难懂和单调乏味，那你一定不是在讨论一些非常严肃的话题。我们所处的位置使我们觉得最好选择坦诚的、容易理解的、直率的和有趣的写作方式。

谁应当阅读本书，为什么

这本书是专门为下一步准备选择更复杂、边缘或高级深奥的课题和程序员们提供解决方案的。如果读者正在从其他的程序设计语言（例如标准的Visual Basic第6版或更早的版本）过渡到VB.NET上来的话，第1章和第2章对你是很很有用的。

这本书对于任何一个具有编程经验的聪明的读者，我们相信，都是容易理解的。在这本书中，我们从头至尾都力图把问题叙述清楚，尽可能直截了当地解释每一件事情，不管各种问题的困难程度如何。

读者在书中会看到各种有用的实用工具，包括第6章中的DES和RSA加密系统，用户可

以直接插入到自己的项目中，用来保护任何一种数据的秘密。关于Office自动化操作的第4章演示了怎样把各种实用程序加到用户的VB.NET程序中；从中提取统计信息，例如字数或段落数；计算数学公式的值；拼写检查；自动检索电子邮件；过滤和显示电子邮件；搜索目录和子目录，从中找出特定的文件；从Word、Excel或Outlook中导入或导出数据；发送传真等。

第5章带领读者漫游了Windows和.NET多层的安全迷宫——即使执行某个单一的功能，也必须首先正确地设置所有各种秘密的锁眼和锁。读者看到了怎样管理基于代码的(.NET)、基于角色的(Windows)和混合型的(IIS, 特殊的数据库等)安全系统。你是否知道，你可以非常确切地指定需要的安全性，可以同意或拒绝执行VB.NET项目中的某个单独的过程——或者甚至某一行代码。

第8章探讨了一种新的叫做反射(reflection)的技术，用户可用这种技术从自描述程序集(包含它们内容描述的代码库)中提取信息。在第8章中向读者展示了怎样在实际的、专业化的应用程序中使用这个信息，甚至怎样更进一步把信息发送出去(编写生成代码的代码)。

窗体设计是第21章讨论的话题：怎样做才能使VB.NET程序的外观更加专业化、更优美和更符合人体工程学。这一章涉及的是程序设计——所谓设计本身的特性，最不为人注意的方面，即所完成的应用程序表面上看起来的样子——编程逻辑出现和被使用者看到的地方。在这一章中要说明的技术有光源、金属表面、字体、分层技术、颜色浓度、取景、阴影、渐变和跃变。在第11章、第13章和18章中，用大量的篇幅分别介绍了数据驱动的Web应用程序、Web服务和XML。

虽然市场上可以找到一本叙述ADO.NET各个方面的书，但是我们只包括两章专门讨论数据库编程的话题：一个带有简单范例的ADO.NET对象模式的概述和带有实用的数据驱动应用程序的一章。这些例子中有一个是开发票的应用程序，它是每个企业最基本的组成部分。除了发送XML数据到日本，或者与驻留在卫星上的数据库交换数据之外，一个公司也许还需要出售产品和服务，因此用户可能需要一种特殊用途的应用程序来准备订单和发票。这是一项琐碎的任务，但又十分重要，在一个介绍数据驱动应用程序的实用章节中不能跳过去不提。读者一定读过许多关于数据驱动应用程序中间层的书了。如果你理解什么是中间层，可以跳过书中关于开发中间层组件的讨论。如果读者需要一个简单商业规则的范例，说明怎样作为中间层组件实现商业规则和怎样开发中间层组件，以便不接触该应用程序的代码就可以修改商业规则，因为该应用程序已经配置到用户工作站了，那么，读者不妨仔细研究一下本书第16章和第18章中的范例。

我们在本书中也包括了一些有用的工具，读者只需稍加修改或不加修改就可以用在自己的项目中。PRNTextBox和PRNListView控件是TextBox和ListView控件的增强型版本，它们提供了打印控件内容的方法。正则表达式并不是典型VB开发者经常使用的，这也是大多数书中完全忽略这个话题的原因。为了在第19章中示范正则表达式，我们介绍了RegExEditor：一个简单的文本编辑器，允许用户用多种搜索模式(例如电子邮件地址、美元数量等)搜索文本文件。第20章中包括两个图形应用程序，一个用于绘图功能(一个实用的控件，用户可用于任何数字处理应用程序)和一个不规则碎片应用程序(一个产生各种供用户消遣的迷人图案的程序)。所有这些实用工具都可以从网址www.sybex.com获得。

目 录

第1章 理解.NET框架	1
为什么要阅读本章	2
帮助	3
抓住框架类的描述	5
搜寻语法	6
理解数据类型	11
开发Framework	18
一个有用的类观看工具	20
简明词典	21
小结	22
第2章 执行传统任务的新方法	23
剪贴板访问	23
用“控件数组”工作	23
使用数组	27
灵活的ArrayList	33
使用HashTables	35
新的日期/时间技术	35
文件I/O（流技术）	37
窗体引用：窗体之间的通信	42
使用Handles命令	43
用LoadingPicture装载图形	45
管理注册表	45
随机数	46
SendKeys	49
串行化技术	50
小结	53
第3章 串行化技术	54
串行化是怎样起作用的	54
创建可串行化对象	60
用户自定义串行化	73
串行化SQL Server数据	75
小结	84

第4章 在用户应用程序中使用Microsoft Office	85
使用Word的功能	85
使用Outlook对象	99
访问Excel	101
小结	106
第5章 理解.NET的安全性	107
安全性概述	107
理解代码访问安全	113
管理.NET安全性政策	116
安全的编程方法	121
小结	124
第6章 加密术、散列法和创建密钥	125
主要问题	125
加密	128
不对称加密术	136
小结	142
第7章 高级打印技术	143
在.NET中打印	143
打印机和页面属性	146
打印相关的对话框	148
页面布局和打印	151
打印普通文本	157
打印表格数据	161
PrintScreen工具	168
小结	172
第8章 反射	173
反射的用处	173
反射	175
用创建实例和调用执行发现的代码	188
发射	192
小结	193
第9章 编写健壮无缺陷的应用程序	194
结构化异常处理程序	194
调试技术	209

小结	218
第10章 配置Windows应用程序	219
安装.NET框架运行时间	219
XCopy配置	220
Internet配置	221
用Windows安装程序进行配置	234
小结	245
第11章 建立数据驱动的Web应用程序	247
ASP.NET的新特点	247
在WebForm上显示数据	248
验证	257
发送图表	261
使用HTML控件	262
小结	263
第12章 层对层的编程	264
Internet编址	264
使用套接字	267
TCPChat应用程序	274
与网络资源交互	281
小结	290
第13章 高级Web服务	291
Web服务是什么	291
创建Web服务	292
Web服务的使用	294
状态保存	296
建立数据库连接	297
WSDL的实现	300
UDDI:注册表	307
测试已发布的Web服务	307
安全问题	309
小结	309
第14章 利用消息队列创建异步应用程序	310
队列和消息	311
MessageQueue类	316

Message 类	317
确认和超时机制	326
处理订单消息	339
消息队列触发器	347
小结	353
第15章 实用的ADO.NET	355
数据库访问	355
DataSet 操作	376
插入和更新操作	388
小结	399
第16章 中间件设计	401
从客户机-服务器方式到多层结构	401
远程商务逻辑	408
在.NET 客户机中使用COM 组件	420
在.NET 中使用COM+ 组件	423
小结	431
第17章 XML 技术	432
SAX 选择	432
DOM 的深入了解	435
在XML 文件中使用名字空间	436
计划书的爆炸	437
可编程的XML 文档	443
XML 和DataSet	448
支持SOAP	450
更灵活的可交换性	457
小结	457
第18章 设计数据驱动的Windows 应用程序	459
数据绑定	459
NWProducts 应用程序	460
开发票应用程序	469
相关应用程序	487
Relations1 项目	492
小结	494

第19章 运用正则表达式	495
编写正则表达式	495
RegularExpressions 类	499
正则表达式的元素	508
正则表达式的高级课题	517
RegularExpressions 项目	529
Visual grep 项目	531
小结	537
 第20章 高级图形技术	538
PlotControl	538
碎片发生器	550
复数运算	566
小结	567
 第21章 设计用户界面	568
使应用程序显得可靠	568
Windows 约定	571
金属阴影	577
滑动和渐变转换	578
小结	583
 第22章 使用.NET小型框架和显示技术	584
删除了什么	584
使用模拟器	586
更新的特性	588
移动的安全性	592
通过跟踪调试	593
提供友好的错误信息	596
设备的特殊性	598
使用仿真程序	599
小结	602

第1章 理解.NET框架

Visual Basic.NET并不是Visual Basic程序员写的，全部.NET系列语言都是C程序员创建的。C和它的同伴——面向对象的程序设计（OOP）是一种学术语言；而Visual Basic是一种通俗流行的语言。

这些事实是有重要意义的。人们在1990年就已经考虑要用Visual Basic来代替C。VB是一种专门用于快速完成应用程序开发的语言——它成功地摆脱了隐语和愚民政策。VB是专门为那种想把小的税款快速计算在一起的小商人创建的，或者是为那个想写几个地理测验题来帮助Billy完成家庭作业的母亲而创建的。VB是为大众而设计的。有几十万人在使用C；而有数百万人在使用Visual Basic。

像许多文明一样（罗马对埃及，美国对法国，城镇对长袍），程序设计语言很快就分成了两大阵营。C和它的后代（C++、Java、C#等）代表一个很大的程序设计师阵营；Visual Basic是另一个阵营。然而，.NET企图把Visual Basic与C语言融合在一起——但仍然尽可能地保留了VB令人满意的标点法（直截了当、条理清楚、简单易懂、类似英语）、语法和句法。

许多教授（祝他们好运！）总是热衷于抽象、分类和精细的定义。其实，这正是使VB.NET在某些方面造成不必要混乱的原因之一。它有许多“可达到的”层（范围），有各种各样组织数据的方法，一些方法比另一些更加有用。它有多种“赋予资格”的方案；相当大的冗余；一个术语可能带有多种意义（例如strong typing）；多个术语可用于一种行为（Imports与Import相比）；而且它自己的规则总是有各种各样的例外。

不过很显然，VB.NET在许多方面都比过去版本的VB有所改进。我们每个人肯定都能够找到从自己当前的状态过渡到分布式程序设计的技术。VB.NET比过去版本的功能有相当大的提高。例如，流技术（streaming）已经取代了传统的文件I/O，可是流技术也可以处理来自多个数据源（不仅仅是硬驱）的数据流。流技术显然扩展了用户数据管理工具的功能。用户可以用一个WebResponse对象来代替FileStream对象，把数据传送到一个Web客户机去。

尽管如此，在人们把所有的计算机语言都合并到.NET的保护伞下所做的努力之中，VB不得不放弃了一些它原有的简洁性和透明性。事实上，VB现在也像所有其他的.NET语言一样，生成了某些编译代码——所以，从表面上看这两种语言已经不再有任何区别了。不过，只是从程序员的工作面上看起来如此，其内部还是存在着很多区别的。

像生物学一样，OOP（Object Oriented Programming，面向对象的程序设计）本身也涉及到一个复杂的分类系统。这意味着使用OOP的人必须花费相当多的时间去完成牧师要做的工作（例如，这个是从哪里来的？这个应当怎样来描述？这个应当归于哪一类？这些文本操作功能是在Text（文本）名字空间还是在String（字符串）名字空间？这个对象必须要用具体例子说明，还是能够不必首先创建对象而直接使用它的方法？）。

为什么要阅读本章

如果像我一样，你是数百万VB程序员中的一个，而且头脑聪明，反应灵敏，那么这一章可能就对你有用。

我并不缺乏才智，想必亲爱的读者也和我一样。但是在VB.NET的世界中苦苦探索的过程看起来真的太漫长了，特别是刚刚开始的时候。这一章会给你一些实实在在的忠告，可免除你许多不必要的困惑和概念混淆。

如果读者有C程序设计语言（以及与C密不可分的面向对象的程序设计）的基础，当然掌握VB.NET会显得容易得多。

数百万VB程序员经过慎重考虑之后决定不使用C。这就是为什么开始时我们会成为VB程序员的原因。因为我们更喜欢VB具有能快速开发应用程序的工具。我们不喜欢那种反方向追溯的语法结构、多余的标点符号（所有的那些分号）和C以及它的子语言的其他方面。

可因特网改变了这一切——我们必须开发新的技能来适应新的程序设计风格。离开已经适应的编程环境（Windows的应用程序，在一台计算机上运行）要求我们掌握新的技术。用户不必转换到C或它的同类，但的确必须扩展自己的VB词汇量和编程技能。

当前的程序有时被分成多个小程序（programlets）（分布式应用程序），驻留在不同硬盘驱动器的不同的位置上，有时这些小程序甚至使用不同的平台或语言。Web Services将来必然会形成高潮，这种计算技术极大地增加了对通信和安全问题的冲击。更不用说压缩代码成为对象的必要性了。

所以要系好安全带，当危险来临时有所防备，准备好对付某些新思想。现在读者已经打算处理一些不同的想法了。

最近几年，我们每个人（作者）都写过好几本关于VB.NET的书（每天在.NET领域工作已经有三年时间了），但现在我们仍然会发现新的工具、概念和特性。这里面有一部分是对巨大.NET框架简单的了解，有一部分是对OOP和其他现在是VB的组成部分类似C的元素进行调整。

有人说他们已经把量子力学应用到了OOP上：全世界大概只有十个人能很好地理解这种说法，没有一个人能完全理解。所以一定要勇敢。你肯定能够学会一些模式和规则，帮助你在.NET中取得成效，你能得到的好处是，VB.NET比传统的VB功能更强大，而且也灵活得多。你的耐心和努力将会得到丰厚的回报。

在本章中，读者会找到一些思路，这会加深读者对大量巨大的.NET环境和框架结构的理解。读者会在本章找到许多有用的信息，这些信息将提高你VB.NET的编程技术——我担保。例如：什么是结构，什么时候应当使用结构？它们是用来替代VB中用户自定义类型的，但是绝对不应当使用它们。OOP专家们说，不论何时，当你想使用结构的时候，都可以创建一个类来代替——这样会有更多的灵活性。当然，其他OOP专家不同意这种说法，于是争论开始了。

帮助

VB与C类型语言的合并所产生的重要后果是，VB.NET Help系统和文件大部分都是由C程序员写的。这些人既不是普通的作家，他们也不很熟悉Visual Basic。这就是为什么人们在Help系统中总能发现一些从技术上说是错误的信息、令人费解的描述和其他模糊不清的补丁程序块。

所以，现在Help的描述已经不像VB受到人们称赞的那样清楚和透明，听起来像是由一个呆笨的美国国税局官僚写的一样。这里有一个例子：

修改与任何一个实例关联的域或属性的值，不会影响该类其他实例相关联的域和属性的值。从另一方面来说，当修改与某个类的实例相关联的共享域和属性的值时，则会改变与该类所有实例相关联的值。

明白我的意思了吧？

不仅是VB.NET文件经常叫人感到莫明其妙，它是由C程序员写的这一事实就意味着，Help的描述甚至包括源代码的范例经常都是些半英语、半C的东西。

许多Help源代码的范例都列在“VB.NET”版本中，事实上，它们都是C程序员写的。VB程序员必须花时间把它们转换成假冒的VB代码。令人高兴的是，现在Help中有很多测试用的无程序错误的范例代码。不过，也许Microsoft最好还是请一个有经验的VB程序员把所有假冒的“Visual Basic”代码范例通读一遍，把它们转换成真正VB风格的程序代码。

例如，请读者看一下Help中用于String.IndexOf的条目。如果向下滚动右边的窗格，可以看出所有范例代码的情况，它们不是典型的VB代码。许多VB程序员都不得不考虑怎样才能使这些代码起作用。不能只是拷贝和粘贴这些代码。

VB程序员可能会对下面代码中，与在Help中许多其他范例代码中会看到的奇怪标点符号以及其他古怪的东西感到困惑不解。虽然在名义上它们是Visual Basic源代码，但很多Help范例代码在许多细节处都是与Visual Basic全然不同的，正如在下面这个描述IndexOf方法的例子中可以看到：

```
Imports System
Class Sample
Public Shared Sub Main()
Dim br1 As String = _
"0----+----1----+----2----+----3----+----4----+----5----+----6----+--"
Dim br2 As String = _
"0123456789012345678901234567890123456789012345678901234567890123456"
Dim str As String = _
"Now is the time for all good men to come to the aid of their party."
Dim start As Integer
Dim at As Integer
```