

develop mentor series

DON BOX, SERIES EDITOR

高级 Visual Basic 编程

Advanced Visual Basic

Matthew Curland 著

涂翔云 刘玉印 刘岩 译

Power Techniques for Everyday Programs



本书附赠光盘一张



Addison-Wesley



中国电力出版社

www.infopower.com.cn

高级 Visual Basic 编程

Advanced Visual Basic

Matthew Curland 著

涂翔云 刘玉印 刘岩 译

Power Techniques for Everyday Programs

中国电力出版社

内 容 提 要

本书是微软 Visual Studio 的资深专家 Matthew Curland 多年 VB 开发经验的总结, 着重于解决 VB 程序员编程时所遇到的困难。由浅入深, 循序渐近地讲解了如何开发标准的 VB 代码、高级编程技巧、以及如何通过一些新算法的使用提高编码效率和性能。附带的光盘提供了书中内容的相关代码, 稍加扩展即可获得用户自定义类型、轻量对象系统、创建定制窗口以及函数重载等功能。盘中还提供了三个功能强大的类型库, 对 VB 中使用的和由 VB 产生的类型库进行了修正。

本书适合高级程序员阅读, 也可供专业计算机人士参考。

图书在版编目 (CIP) 数据

高级 Visual Basic 编程/(美)柯兰德 (Curland.) 著; 涂翔云等译.

-北京: 中国电力出版社, 2001.5

ISBN 7-5083-0662-7

I.高… II.①柯…②涂… III.Basic 语言-程序设计 IV.TP312

中国版本图书馆 CIP 数据核字 (2001) 第 034248 号

北京版权局著作权登记号 图字 01-2001-2232

本书英文版原名: Advanced Visual Basic 6

Published by arrangement with Addison Wesley Longman, Inc.

All rights reserved.

本书中文版由美国培生集团授权出版, 版权所有。

中国电力出版社出版、发行

(北京 甲河路 6 号 100044 <http://www.infopower.com.cn>)

河市实验小学印刷厂印刷

各地新华书店经售

*

2001 年 7 月第一版 2001 年 7 月北京第一次印刷

787 毫米×1092 毫米 16 开本 27 印张 612 千字

定价 55.00 元

版 权 所 有 翻 印 必 究

(本书如有印装质量问题, 我社发行部负责退换)

To Lynn and Elizabeth,
the two biggest blessings in my life.
I've missed you.

译者序

本书的原作者 Matthew 是 VB 编程方面的资深专家，是微软 Visual Studio 开发组的长期成员。他曾经参与了 VB 的设计和改进工作，多年来发表了许多有关 VB 编程的论文和专著。他在这本书中对 VB 编程的一些高级概念进行了深入的探讨，如聚合、单元线程和绑定等。这是一本有关 VB 编程的书，但作者并不是简单地介绍 VB 的特性，而更多的是从 VB、COM 和 OLE Automation 的关系这一更高的层次来探讨如何最优地解决问题的。这正适用于那些在 VB 编程方面有一定基础，希望能深入地理解使用 VB 编程的优点和局限，从而最大限度地发挥 VB 的能力，编写出高质量的专业水平的软件产品的读者。

本书由涂翔云完成了第一、二、六、七、八、九、十六章及附录部分，刘玉印完成了第三、四、五、十三、十四、十五章，刘岩小姐完成了第十、十一、十二章的翻译和校对工作。从本书的篇幅中读者也许能够体会到译者的艰辛，在翻译行将完成之际，我们看着彼此日渐憔悴的面庞，不禁感慨万千。但愿本书给广大读者带来的益处能使心血得以补偿。

在此我们要感谢家人给予我们的支持与理解，翻译本书适逢传统的春节，为了让本书能早日与读者见面，我们不得不放弃与家人团聚的机会。

在此对微软亚洲技术中心的沈小兵先生表示由衷的感谢，他给本书的翻译工作提供了大量技术上的支持，使得我们的专业不足得以弥补。

同时，还衷心感谢周郭飞先生、冯琳小姐、涂辉云先生还有姚欣小姐，他们做了大量的审阅稿件和校对文字工作，并提出了许多宝贵的意见，是大家共同的努力，促成了本书中文版的诞生。

由于我们对 Visual Basic 使用的时间不长，掌握的程度也有限，书中翻译错误难免，恳请各位同仁和广大读者批评指正。

涂翔云 刘玉印

于 2001 年 2 月 5 日凌晨

前言

给一本书写前言是一种荣幸，特别当书的作者是一位著名作家而且这本书很有可能会成为 VB 高级编程人员的“圣经”时更是如此。但是，当让我写这几页前言时我的感觉要简单得多：“我终于有机会在 Matt 的书躺在书店里数月之前拜读它了！”，而我感觉到这更是一种荣幸。现在您应该体会到我是怀着怎样迫切的心情来阅读这本您也在阅读的著作了吧。

我甚至不用介绍 Matthew Curland。如果您曾经阅读过他发表在 Visual Basic Programmer's 期刊上的加黑标题的编程专栏或者在 VBITS 以及其他的会议上听过他的报告的话，您不可能不产生一种要把 Visual Basic 推向它能力的极限的冲动。即使您是 VB 的新手，当键入“.”时看到一大串控件或对象的方法和属性时，您也会感谢他。（Matthew 是给 Visual Basic 增加智能提示特性的开发小组的成员。）

基于下面的一些原因，我特别高兴 Matthew 完成了这本书。他在最近几年发表了许多文章，但那些文章由于篇幅所限，不得不将许多高级概念压缩到几页文字中，相比而言，您会更愿意阅读厚厚的一本著作。我在开始阅读本书时，迫不及待的想了解后面的内容，略过了一些我认为已经很好的掌握了的专题，但到后来我发现自己不得不回过头来认真的弥补知识上的漏洞。

第二个原因是我现在能够只享受我对 VB 的既爱又恨的感情的好的方面。正如作者在序中所说的那样，许多高级 VB 编程人员喜欢 VB 是因为 VB 简单，但因为 VB 功能有限他们也憎恶它。这本书告诉我们使用 VB 可以做任何事情。同样重要的是，作者还告诉我们许多编写健壮、高效的应用程序的技巧，您在日常编程中会使用到这些技巧。实际上，在我发现这本书中包含了许多我以前的工程中出现的的问题的解决方法时，不禁感到十分吃惊。我相信您也会有同样的感觉。

我喜欢这本书的最后一个原因是 Matthew 写了一本很有层次性的、可重用的书。作为一个程序员，您对代码重用这一概念应该不会陌生，但是书的重用对您来说可能是一个新事物。简单地说，当您可以多次阅读一本书并且每一次阅读都带有不同的目的的时候，这本书就是可重用的。一段时间之前我就认识到可重用的书确实存在，但实际的例子却不太好找。因为我已经说服您购买了这本书，我现在就要告诉您如何最大限度地利用这本《高级 Visual Basic 编程》。

第一次阅读时您不必仔细阅读书中的每一个字。只是对这本书作大概的了解，熟悉一下 Matthew 面对问题和解决问题的方法并且记下一些 VB 的未公开的细节。即使不想在不久的将来使用他介绍的那些编程技巧，这些细节也能省去您不少麻烦。您可以在几小时内

完成这一步，我想即使您三岁的小孩在旁边玩，这些时间也足够了。

现在要重用这本书了。这次您要坐在计算机旁，把您的小孩关在屋中。返回第一页，好了，略过前言，然后集中精力阅读书中的所有东西。这一步可能要花好几天的时间，但是您得到的回报是您可以借助 Matthew 提供的工具（也可不借助工具）使用您心爱的编程语言来创造奇迹。

您要勇敢地回到重用这本书的另一个层次。将这本书摊开在书桌上，浏览光盘中的代码然后搞明白每一行代码完成的任务。您会发现许多至今仍未公开的 VB 的秘密，您就会学会编写那些只有真正的权威才能讲授的代码。这一次您要关上您的 CD 播放器，摆上一杯咖啡并且可能数天不能去上班。

最后给您一个忠告，这来自于我的亲身经历：这本书可能使人上瘾。不要在睡觉之前看这本书，否则您不可能安然入睡。

Francesco Balena 简介

WWW.vb2themax.com的创始人

Visual Basic Programmer's 期刊的编辑

Programming Microsoft's Visual Basic 6 的作者

序

Microsoft Visual Basic 被微软定位为一种高级软件开发工具，它使无经验的程序员能够创建 Windows 应用程序。VB 中包含了丰富的工具包以支持快速应用程序开发，所以 VB 是一种真正的高级语言。除此之外，VB 也被越来越多的高级程序员们所利用。他们选择 VB 作为开发工具，主要是因为 VB 相对于其他应用程序开发平台而言，工作效率高一些，错误少一些。

高级 VB 程序员，对 VB 可以说是既爱又恨。他们爱 VB 是因为 VB 对底层细节进行了充分考虑，从而可以让他们将精力放在一些用户事务问题上。而他们恨 VB 则是因为当系统需要充分控制时，在 VB 中不能做到在更低层次上对程序进行完全的控制，从而也就不能控制整个系统。由于低层次的代码在一个大的应用中通常只占一小部分，所以 VB 的优点显得更为突出，对于一些存在的问题程序员们也可以忍受。绝大多数应用程序只需要少量的低层编码，例如：用它们来产生特殊的 UI 效应，消除性能瓶颈，减少内存的使用以及解决其他一些问题。这些考虑在一般的应用中可能只占很小的比例，但是他们对于一个专业化的软件产品来说却是绝对必要的。

本书着重于排除许多 VB 程序员在试图完善他们的应用时所遇到的困难。这些问题也是一个管理者在对一个新的软件工程项目中将要用到的技术进行评估时需要考虑的。管理者可能经常会选择使用低级语言进行编码，因为他们害怕整个工程不能在 VB 中完成。实际上，这种选择经常是不适宜的，因为低级语言已经不能适用于当今大量应用的发展。仅仅为了防止日后在工程中需要加入一些低层编码而花费几个星期甚至几个月的时间辛辛苦苦地在 VB 中去构筑传统的 C++ 代码似乎有些浪费。

消除 VB 中存在的障碍，最简单的方法便是利用 VB 中的隐藏数据类型。从创建对象到管理数组和字符串，几乎做任何事，VB 都始终与 COM 和 OLE 自动联系得十分紧密。尽管在 VB 中我们不能直接看到隐藏的 COM 类型，但实际上它们总是潜伏于表面之下：我们并不需要为了寻找相应的 IUnknown、BSTR、SAFEARRAY 和 VARIANT 等数据类型而去挖掘 VB 中对象、字符串、数组及可变类型变量下面深层的东西。COM 是基于二进制上的，因此，所有 COM 类型的内存布局都易于描述，也易于复制。

通过了解在 VB 中如何使用 COM 类型，我们将打开一个全新的 VB 世界。通过对这些数据类型进行认真考察，本书将打破 VB 表层，从而向读者展示位于标准 VB 代码表层之下的东西，同时也将向读者展示如何来创建自己的定制化对象。一旦让 VB 认识到一个定制化内存块实际上是一个数组或者一个对象，我们便能使用标准的 VB 代码来操作数据。

我并不打算教读者在 VB 中如何编写 C++ 代码，相反，我所要教读者的是如何去提高 VB，从而使 VB 代码运行得更好，并支持一些其他环境下不可实现的功能。

对于本书的读者，我在此提出三个主要的目标。

(1) 学会如何写出更好的标准 VB 代码，即使在读完本书后，并不打算去使用那些高级技巧，读者也仍然应该能够编写出更为健壮的 VB 代码，并使代码具有更加出色的性能。

(2) 学会一些高级 VB 编程技巧。

(3) 通过最小的代码重用，及一些新算法的使用，读者可以使用这些编程技巧来提高日常的编码效率和性能。与 VB 的指导方针一致，在本书中我也力图将一些底层技术封装，从而使读者可以在无需完全理解其底层结构的情况下很容易地使用这些技巧。

本书所面向的读者

本书特别适用于高级 VB 程序员，同时也适用于那些想进一步了解 VB 工作机理的 VB 和 COM 程序员。我强烈建议阅读本书的人应具备一定的 VB 和 COM 工作经验。如果读者不具备这方面知识和经验的话，那么最好买一本别的书与本书配套使用。

即使读者并不认为自己是一个高级的 VB 程序员，而且也并不像我一样对底层操作有较好的理解，那么通过学习本书，读者也将受益非浅。本书附带的光盘还提供了部分相关的代码，这些代码分别实现了子类化、对象聚合结构、函数指针调用、直接内存共享以及工作线程的创建等一些较常用的功能。只要再添入一些自己的代码，读者即可获得诸如用户自定义类型、轻量对象系统、定制化创建窗口以及函数重载之类的扩展功能。并且，读者还可获得三个功能强大的类型库，这些库对 VB 中使用的以及由 VB 产生的类型库进行了修正。

尽管本书中的许多代码都是底层代码，但本书的重点并不在于如何编写这一层次的代码。本书的目的是：从对整个接口实现的重用到优良的层次化对象模型，为用户提供一整套实用性的工具。读者可以利用聚合技术，并通过使用以普通接口实现为主体的组合对象来提高代码的重用。并且，使用这一技术的时间越长，我就越发现它能够简化我的一些日常代码。事实上，每一位 VB 程序员都能够从本书中找出对自己有用的代码并将其应用到日常工作中去。

未来的可兼容性

VB 7 的发行已开始，所以读者可能已经注意到：我所提供的一些技术在 VB 7 中是很有用的。例如，VB 7 支持具有继承性的重载函数、函数指针调用以及为类函数提供指针。如果读者现在利用本书所提供的技术来实践这些设计原则的话，那么，只要通过在 VB 7 中等价地替代这些框架调用，就可以将应用程序安全地转移到 VB 7 中。这里需要改变的只是框架代码：整个设计结构和代码逻辑都不需要作任何变化。在 <http://www.PowerVB.com>

网站上我尽我全力向读者提供了关于 VB7 兼容性的一些信息，另外，用户还可以通过 Matt@PowerVB.com 与我联系。

本书内容摘要

本书共有十六章和一个附录，主要是从底层察看了隐藏于数组、对象及字符串类型之下的一些数据，并对这些结构的实际应用进行了讨论。

1. 创建块

本书中的许多技巧在很大程度上都依赖于对 VB 中数组和对象类型下的内存进行操作的能力。对底层数据的读和写需要直接对内存进行访问，或者称为“指针操作”。VB 中并没有为指针操作提供太明确的支持，但对于启动我们的旅行来说已经足够了。在第一章中不仅讲述了 VB 是如何稳定地来处理指针的，而且说明了在 VB 中如何对实际的指针进行访问。另外，从本章中读者还将了解到对支持性 VBoost 对象的有关介绍。本书中的许多样例程序都需要用到 VBoost 对象，所以读者在运行样例程序之前需要知道如何对 VBoost 进行初始化。

2. 使用数组

我以数组作为开始，原因并不在于我认为数组是用得最为普遍的一种结构，而是因为数组允许我们采用标准代码来使 VB 对任意内存位置进行修正。VB 中的数组变量实际上是指向一个被称为 `SAFEARRAY` 的数组描述符结构的指针。VB 自动地允许对数组的数据部分进行写操作，但除此之外，我们还可以对描述符、数组变量进行读写，从而进行直接内存访问。另外，读者还可以看到在简单的 VB 代码中如何来最好地利用数组。本章所讲述的技巧将贯穿全书，并且还可用于其他技术的优化中。

3. IUnknown:未知量

VB 要不断地与 COM `IUnknown` 接口打交道，但在通常的 VB 代码中我们是不会看到这个接口的。`IUnknown` 调用在运行时间和代码产生上的花费很大，因此理解 `IUnknown` 接口是很有必要的。即使读者自己不调用 `IUnknown`，VB 也会频繁地调用它。从后续章节中读者将看到，熟悉 `IUnknown` 有助于在最底层上构建 COM 对象。如果读者不具备 VB 与 `QueryInterface`、`AddRef`、和 `Release` 函数相交互的知识，就不能充分用户化 VB 的 COM 对象。

4. 绑定函数到对象上

“绑定”是用来描述一个函数是如何调用另一个函数的代码的一般术语。绑定到对象上意味着将与对象类有关的代码应用到这一类的一个特定实例上。读者将看到 VB 编译器如何去确定什么时候、如何去绑定以及如何在运行时间动态的进行绑定。读者也将看到如何绕过 VB 放置用户控件的层来直接与 VB 的固有 VTable 接口进行对话，从而将运行时间减至最少。

5. 对象的设计结构

一个设计良好的结构，不管采用何种技术，都是程序成功的最重要的因素。读者可以参考面向对象、基于接口、重用优化以及其他一些程序设计原理。无论读者的程序设计的原理是什么，目的都是写出稳定的、易于维护的程序。本章着重介绍可插入组件的 **try-and-true** 原理、抽象和代码重用原理在 VB 中的实现。VB 本身提供了“**Implements**”关键字来为对象提供一个雏形，这一关键字在读者使用基于接口的语言来达到设计目标的时候起着很大作用。但是，读者会发现自己不能容易地重用实现代码。我将带读者摆脱 VB 的限制，通过聚合现在的对象以充分地重用实现代码。这给予读者将多个实现合并到一个对象中去的能力。

6. 循环引用

每次设计一个对象模型的时候，几乎都会遇到循环引用的问题。如果读者事先知道如何通过强引用（标准对象变量）和弱引用（对象指针）来处理这种现象的话，那么便可以利用所需工具来设计对象系统并保证不会在系统拆分时产生问题。为了帮助读者对比各种不同的方法，我将向读者说明，在无需使用指针的情况下，解决循环引用问题所需的一些步骤（包括解决方案不够完善或过于繁琐的情况）。最后，我将把弱引用这项技术应用于集合、对象所有权以及层次对象模型中。

7. 外部对象的创建

可将所有 COM 对象分为两大类：一类为工程内部对象，另一类为工程外部对象。COM 对象不能直接创建，相反，COM 使用注册表来创建类工厂对象，其中类工厂对象用来创建实际的类实例。因为根据应用程序路径而非注册表来直接从 ActiveX DLL 和 OCX 组件中载入对象，直接对类工厂进行处理将十分简单。在本章中，读者还将看到如何利用聚合及类工厂装载来动态地将控件指定为 MDI 页。本章的最后，将向读者讲述如何使用 **CoRegisterClassObject** 和 **CoGetClassObject** API 函数来构筑自己的应用程序级对象，并创建一个跨多组件的且易于取得的上下文对象。

8. 轻量 COM 对象

即使并不需要用到所有的 COM 支持, VB 仍然会创建一个完善的 COM 对象。轻量 COM 对象已经具备被 VB 识别为一个对象的功能, 但没有 VB 所创建对象那么大的内存开销。令人激动的 COM 对象和令人生厌的用户自定义类型 (UDT) 之间的唯一区别便是 COM 对象中具有 VTable 指针。因此, 可以为 UDT 提供一个自己的 VTable 从而把 UDT 转换成 COM 对象。在本章中, 我们将看到几个简单的轻量对象, 并展示了基于堆栈的局部对象以及局部或模块级结构中的终止代码, 并且这些对象还具有实现任意可聚合到标准 VB 对象中的接口的能力。接下来, 读者还将看到创建任意对象的所有步骤: VTable 布局、内存分配选择、引用计数、利用 QueryInterface 进行接口识别以及用于产生和处理错误的一系列可选项。

9. 大型多对象系统

对于创建少量的十分复杂的对象来说, 采用 VB 类模块是完美的选择。但是如果以同样的方法去创建大量的简单对象, 那么创建每一对象所花费的开销合在一起, 其总开销将变得十分惊人。从日程安排工具到编译器, 大量算法通常需要的都是成千上万个对象而非少量的大对象。读者将在没有使用这些对象进行编码的经验的情况下, 学习如何使用一个定制化内存管理器来分配轻量对象。当使用完这些对象之后, 可以一个个地将它们释放或者利用单个调用来回收整个对象系统所使用的内存。

10. VB 对象和运行对象表

我们不能使用 GetObject 关键字来取得 VB 对象, 因为 VB 并没有提供一种在运行对象表 (ROT) 中注册公有对象的机制。本章将细致地阐述轻量对象的使用和实现, 从而使得可将对象放在 ROT 中, 并且在对象终止后会自动地将其移除。关于实现细节, 读者可以参看高级轻量对象专题, 例如: 在单个轻量对象中对多个接口的支持以及如何安全地保持对辅助接口的弱引用。

11. 调用函数指针

在 VB 的新版本中, 除保留了 VB5 中的 AddressOf 操作之外, 还提供了对函数指针的支持。其主要目的是提供对从 VB 中的 Addressof 出发来调用大量的 Win32 API 函数, 另外也是为了能够为轻量对象建立 VTable。由于我们可以创建一个函数指针并调用它, 所以本章中定义了 VB 上的一个转换 VTable 的轻量对象。函数指针为许多可能性提供了一个入口, 例如: 动态地装载 (或卸载) DLL; 编写任意的排序程序; 在 VB 中使用显式堆栈分配; 调用内联汇编代码。因为在调用标准函数指针方面仍然存在一些漏洞, 所以我在本章中还介绍了如何将一个对象实例与函数指针 (极大地简化了标准操作, 例如子类化窗口) 相联

系起来，并同时介绍了如何来调用 `cdecl` 函数，从而使得读者能够对 `MSVCRT` 中的入口项和其他的 `cdeclDLL` 进行调用。

12. 重载函数

在完全没有提供重用的实现与提供了大量实现重用的聚合之间，还存在着一种部分实现重用，并且部分实现重用是继承性所提供的一个主要的优点。继承的优越性在很大程度上依赖于派生类为使对基类中函数的调用能够首先到达继承类而对某个函数的实现进行重载的能力。本章将说明如何来对函数进行重载，一是通过使用函数指针来重定向基类中的调用，二是使对单个对象的聚合定制化从而在无需与基类协作的情况下重载函数。为了在 `VB` 中使用继承，读者还必须编写一些附加的代码。不管怎样，这总比完全不能实现要好得多。

13. VB 中的线程

“`VB` 支持多线程组件”是一句简单的话，但它却意味着许多东西。本章试图弄明白多线程这一术语，并向读者展示如何充分利用 `VB` 的线程的能力。读者将看到 `VB` 如何在 `COM` 单元内运行以在线程层次上完成初始化的。读者还将会看到如何在一个 `VB ActiveX EXE` 工程中 `spin` 其他的线程。本书也将向读者展示如何在具有跨线程对象支持（提供优化的编程灵活性）或没有对象支持（提供优化的性能）的 `DLL` 中，启动一个工作线程。

14. VB 中的字符串

在 `VB` 中字符串操作是十分容易的。读者具有大量的操作符和字符串函数可供使用。但是，这并不意味着字符串操作是无代价的。每一个字符串是一块内存单元，越大的字符串需要越大的内存开销。自动字符缓冲可给读者有一种虚假的安全感，当字符串的大小大于缓冲区时这种感觉会马上消失。读者应结合 `VB` 字符串类型去了解处理字符串的真实开销以及如何使开销最小化，其中包括如何将一个字符串当作数值数组来处理以便提高处理速度。

15. 类型库和 VB

尽管 `VB` 依赖类型库去编辑组件并向外部应用提供这些组件，但产生和使用类型库在日常工作中却不易碰到。读者和类型库的唯一直接联系是通过工程/参数对话框。产生完整的类型库几乎总是正确的，但在一些大的工程中，当读者迫切需要控制类型库时情况就不同了。本章将向读者展示如何创建 `VB` 使用类型库，以及读者编辑二进制兼容性文件和在 `ActiveX` 工程中创建的类型库的原因。本章没有详细介绍 `PowerVB` 类型库修改器、`PowerVB` 二进制兼容性编辑器、`PowerVB` 后期构建类型库修改器插件，这些都包括在本书的光盘中。相反，本章展示了让读者能使用这些工具来完成任务的步骤。

16. 控制 Windows

绝大多数客户端的 VB 程序都建立于窗口基础之上。事实上，VB 之所以能在众多的程序开发工具中占有一席之地，是因为它使得创建一个 Windows 应用变得十分简单，并免除了和各种复杂的窗口对象进行交互的麻烦。但是，如果试图在创建及窗口过程的层面上与窗口进行交互，这一保护层却带来了许多的麻烦。本章还将前几章中所讲到的函数指针、直接内存访问、轻量对象及函数重载等技术应用到一般的窗口对象和特殊的定制化控件上。读者还将了解到一种轻量的子类化方法，它用来控制即将到来的窗口消息。实现子类化之后，本书将着重于讲述定制化窗口的创建以及怎样来创建无窗口控件并使之如同一个带有窗口的控件一样发生作用。

附录：VBoost 参考

VBoost 对象是一个小的函数库，它为本章许多的技巧提供了基础。本书中包含了一个 VBoost 的 C++实现(VBoost6.DLL)和一个 VB 实现 (VBoost.bas)，这些实现使我们可以去除对外部 VBoost 依赖。并且它们提供了在执行聚合、定制化 IUnknown 钩入和函数重载中所需的任何东西。从这里还可以得到两个优化的内存管理器变量和在 VB 中大大简化指针操作的一系列赋值和算术函数。除了代码以外，VBoost 还包括一系列按功能分类的类型库。这些类型库将使得我们可以很容易地编译光盘中 Code 目录下的文件。

致 谢

首先我想对我那才华横溢、美丽的妻子 Lynn 致以最深切的谢意，因为当这本书象蘑菇一样长成，开始呈现它的生命的时候，她只能忍耐我的日常时间的压缩。还要感谢我的女儿 Elizabeth。她与此书共同成长，把我从电脑旁拉开，告诉我是该休息的时候了。小孩总是知道什么是最重要的。

感谢那些审阅过本书的许多技术细节的人们：Troy Cambra、Dan Fergus、Michael Kaplan、Karl Peterson 和 Ben Wulfe，特别是 Brian Harris，他在这本书讲述的许多技术还没有修改完善时就已经使用过了。

特别要感谢为这一项目勾画雏形的 Bill Storage。他作为我的好朋友，一直热心地支持我的工作。Bill，我十分高兴和您一起探讨文章并且一起定义 VB black-belt 空间。

感谢 Glenn Hackney，他在这些年中忍受了许多不期的打扰，他是我新思想的知音和技术指导的源泉。。

感谢我的意大利朋友 Giovanni Librando 在两本书的项目中对我的信任。感谢 Francesco Balena，他为我写了绝妙的前言。与这两位绅士共事实在是我的荣幸。

感谢 Addison-Wesley 的 Kristin Erickson 、Rebecca Bence 和 Jacquelyn Doucette，当临近交稿时，他们表现得十分友好而有耐心，同样感谢 Gary Clarke，就是他在一年前用一本小书诱惑我进入到这个项目中来。现在那本小书变成了这本厚书，他知道一定会这样的。感谢上个月黎明时提醒我睡觉的小鸟们。我会想念黎明的宁静，但是我不会忘掉我在深夜中对睡眠的渴望。

目 录

译者序
前言
序
致谢

第一章 构建块	1
1.1 虚指针	1
1.2 活指针	4
1.3 VBoost 对象	8
第二章 使用数组	9
2.1 数组描述符	10
2.2 读取数组变量	13
2.3 写入到数组变量	15
2.4 数组选项：超出固定或可变字长	23
2.5 使用数组的一些小提示	32
第三章 IUnknown 接口：一个未知量	36
3.1 VB 和 IUnknown 接口	37
3.2 声明 IUnknown 并调用它的函数	42
第四章 绑定函数到对象上	45
4.1 何时绑定对象	47
4.2 运行时间的名字绑定	50
4.3 VTable 绑定用户定制控件接口	55
第五章 对象的设计结构	61
5.1 使用 Implements 来实现抽象	62
5.2 调用代码的可插入性	64
5.3 实现和实现重用	67
5.4 聚合	74
5.5 聚合现存的对象	80

第六章 循环引用	84
6.1 中间对象解决方案	87
6.2 弱引用和集合	92
6.3 转移对象所有权	93
6.4 层次化对象模型	94
第七章 外部对象的创建	99
7.1 使用类工厂进行对象的创建.....	100
7.2 直接加载 DLL 对象.....	104
7.3 自定义加载定制化控件.....	108
7.4 定制类对象	115
第八章 轻量 COM 对象	119
8.1 关于轻量的基础知识.....	120
8.2 结构终止代码	128
8.3 LastIID 的轻量版本.....	131
8.4 ArrayOwner 的轻量版本.....	133
8.5 接口位于何处	136
8.6 错误的产生及避免	137
8.7 从轻量对象返回错误.....	139
8.8 聚合轻量对象	147
8.9 编制 Query Interface 函数	152
第九章 大型多对象系统	155
9.1 使用定长内存管理器.....	158
9.2 Scribble 示例.....	159
第十章 VB 对象和运行对象表.....	172
10.1 在 ROT 中注册 VB 对象.....	173
10.2 ROTHook 实现细节	178
第十一章 函数指针的调用.....	194
11.1 示例：调用 DLLRegister Server	198
11.2 示例：QuickSort，一劳永逸	201
11.3 Alpha 中的 VB 函数指针.....	206
11.4 堆栈分配.....	208
11.5 产生自己的内联汇编.....	213