PROGRAMMER TO PROGRAMMER™

Professional Refactoring in Visual Basic

代码重构（Visual Basic版）

MVP Microsoft® Most Valuable Professional

(美) Danijel Arsenovski 著
冯 飞 译

wrox™
WILEY
清华大学出版社

代码重构(Visual Basic 版)

(美) Danijel Arsenovski 著

冯 飞 译

清华大学出版社

北 京

Danijel Arsenovski
Professional Refactoring in Visual Basic
EISBN：978-0-470-17979-6

北京市版权局著作权合同登记号　图字：01-2009-2542

图书在版编目(CIP)数据

代码重构(Visual Basic 版)/ (美) 阿瑟诺维斯基(Arsenovski，D.) 著；冯飞 译. —北京：清华大学出版社，2009.5
书名原文：Professional Refactoring in Visual Basic
ISBN 978-7-302-20084-0
Ⅰ.代… Ⅱ.①阿…②冯… Ⅲ. BASIC 语言—程序设计 Ⅳ.TP312
中国版本图书馆 CIP 数据核字(2009)第 064228 号

责任编辑：王　军　郑雪梅
装帧设计：康　博
责任校对：胡雁翎
责任印制：何　芊
出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**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：北京密云胶印厂
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：185×260　**印　张**：31.5　**字　数**：767 千字
版　　次：2009 年 5 月第 1 版　　**印　　次**：2009 年 5 月第1 次印刷
印　　数：1～4000
定　　价：68.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。联系电话：(010)62770177 转 3103　　产品编号：028900-01

序

“不成熟的优化是所有不幸的根源。”

——Charles Antony Richard Hoare 先生，英国计算机科学家，后来 Donald Knuth 在其著作 *The Art of Computer Programming* 中进一步解释了这句话。

前几天，我观看了一部关于南美电影导演 Fabián Bielinsky 的纪录片，我特别注意到他在一次访谈中与观众共同分享的一些想法。他说在拍摄场景时，通常发生的情况是摄制的场景可能并不是导演原来想象希望能得到的。有时候，需要重新拍摄场景直到导演获得最初想要的场景为止，而在其他一些情况下，有些场景可能是唾手可得的，甚至认为这可能比最初的构思要更好。因此，他总结道，制作电影的真正艺术在于明智地决定什么时候需要重做，什么时候只要使用第一次获得的场景即可。

令人意外的是，虽然这是电影制作的情形，但编写组件的代码与摄制场景也是非常相似的。总是可能获得比给定算法更好的方法。因此，出于时间安排、预算、交付责任日期以及通常的用户满意度的考虑，有着管理人员头衔的人必须决定在什么时候冻结所有可能的改进以及什么时候继续并获得更多的改进。

因此，这里将说明重构的需求问题。首先定义重构在软件行业术语中的含义(非正式的，作者将更好地定义)：重构是一系列用于改善代码片断质量(可理解性、可维护性、模块性、可扩充性等)的技术和机理，方法是以保持通常行为固定不变的方式来重新组织语句。换言之，受影响组件的行为不应该随着过程的后果而是随着它们的质量而变化，并希望增加其寿命。

经验表明，一些代码迟早都是重构的候选，理由有很多：

- 从用户的角度讲，直到看到安装和运行了原来需要的东西之后才能确定确切的应用程序。这并不是玩笑话！从一开始，他们开始需要可以预见的东西，但是很不明确，一切都很自然——我无论如何都不会指责他们。这样一来一回，不希望出现的副作用是代码开始失去一定的凝聚力；由于面市时间的压力，最后的修改结果就是不同的模块可能开始在可接受的层次之上耦合。
- 从开发的角度来看，我们对在建模时编码可能呈现的情况并没有清晰的想法。和用户的表现一样，我们也认为我们有很好的想法(或最后会有很好的想法)，直到我们试图将它们付诸实施时为止。失败并不是一件坏事。不好的是只是因为不想承认没有考虑出一个很好的想法，所以拒绝更改我们的想法，而事实上实现起来是很容易的。同样的，由于时间压力，那么以最快的速度交付一个组件代码也可能损害到它的质量。

- 最后，从技术的角度讲，与行业趋势伴随的是一个看不见的(有点无所不在的)压力。典型示例是演化的.NET 或 Java API——AJAX、Web Services 等——这将让以前版本或通常的策略(如 Service-Oriented Architecture (SOA)、Model-View-Controller (MVC)、Object/Relational Mapping(O/R-M)等)变得陈旧。同样的，在对这些趋势作出反应的同时，已有代码需要作一些处理，这些处理随着时间推移会侵蚀质量。

同时，现实世界告诉我们，尽力提高代码质量是我们天生应该做的事情。要考虑到，重构技术是这类自发尝试的最高级别，并通过一些可供使用的支持工具来确保这一过程的成功实现。

关于这一问题的好消息是可以在局部运用重构，即在组件或方法的层次上，这也是运用任何其他修改的地方；也可以在全局运用重构，即在模块或应用程序的层次上，赋予它整个项目的范围。确定重构是否适当的标准取决于其弥补的质量缺陷、剩余的时间、可供使用的预算等因素。与在没有任何更新请求的地方相比，在必须作出处理的地方总是可以更容易地判断重构的应用。

在本书中，作者将深入讨论重构的话题，包括从预见其优点到目前将重构付诸实施的方法。Danijel Arsenovski 从.NET 和 Java 平台最早的版本开始就已经开始从事重构技术的研究。他在很多会议上做过报告，并创立了这一主题的工作室，同时还推动银行业中重构项目的成功实现。

作为开发工具的领导者之一，Microsoft 完成了将最佳资源提供给那些每天需要处理编码行为和软件项目的人。通过其成功推出的 Visual Studio IDE，Microsoft 使得重构一个现成工具只需要用鼠标键右击代码即可。在这些页面上，Danijel 将展示在 Visual Basic 中重构如何可以像复制/粘贴或其他编辑操作一样的简单。

亲爱的读者，我敢保证，您将获得关于这些技术最可靠的、最基本的指导。请尽情享受阅读本书的乐趣吧！

Diego Dagum
Microsoft 公司的技术作家
2007 年冬天

前　言

感谢您选择本书，欢迎进入到重构的神奇世界中来。笔者希望，随着日常编程工作的进行，您在与同行讨论不同的设计方案时，或是在准备解决一些模糊的遗留代码时，或甚至晚上在头脑中琢磨代码行时，能发现本书的用途。如果这是您第一次接触到重构，那么希望本书能深刻地改变您编程和思考代码的方式。然而，完成这项任务并不是一件简单的事情，最终将通过这些来判断成功与否。

在阅读完 Martin Fowler 编著的 *Refactoring: Improving the Design of Existing Code*(Addison-Wesley，1999)之后，笔者决定以一种系统化的方式采用重构。事实证明，该书是一本很实用的著作，它将帮助学会一些可立刻运用于实际项目的、独立的技术。该书不是建立在复杂理论的基础之上，也没有包含任何复杂的数学公式。任何编写代码的人都能够立刻理解语言。在阅读完该书之后，笔者注意到自己在编程方式上的变化：

- 能够检测到更多确定的问题代码和设计流程。
- 能够为这些问题思考解决方案并通过重构有效地解决它们。
- 在与同事讨论时，能够以一种清晰且精确的方式阐述自己的决策。

最后，笔者不再将这些固定的结构看作不随时间变化的，并将其看作是一个塑性的、可模化的窗体，可以根据自己的需求和爱好来定制该窗体的风格。这样就以处理代码的方式引发了一种基本的修改。笔者意识到存在一种方法，通过该方法，可以以一种高效、可预知的方式来修改代码并改进设计过程。

很快，关于重构的言论在笔者共事的团队中流传开来，笔者看到越来越来多的同事从书架上取下这本书，仔细阅读。甚至有些人还自己购买了该书。这样，笔者就可以使用重构的术语来与同事们交谈，并将重构作为软件构建过程中不可缺少的部分介绍给大家。甚至证明管理层在这一方面是有远见的。

一部分是因为笔者对于学习不同的语言和技术感兴趣，一部分是因为笔者必须常常在自己的工作场所陈述观点，所以笔者开始与不同的团队一起工作并以不同的语言来编程。在与用 Visual Basic 编程的团队一起工作时，笔者尽量区分不同的重构，其方式类似于笔者与使用 Java 或 C#语言的团队一起共事时一样。事实证明这一过程并不是那么的一帆风顺。很快，笔者意识到对于 Visual Basic 的编程人员而言，几乎没有什么关于重构的信息可供使用。虽然在任何面向对象的编程语言中，可以以相同的方式使用大部分的重构，但是仍然存在一些细微的区别。在编程人员阅读完以其自己偏好的语言编写的代码示例之后，他不学习重构的理由将不复存在。

这种情况激励笔者思考 Visual Basic 中的重构问题，并最终编著了本书。笔者认为本书确实有存在的需要，对于使用 Visual Basic 的编程人员来说将会发现本书是非常实用的。

希望本书将帮助读者更快更好地编写代码。笔者希望本书能提高编程人员的生产率并

对编码的性能产生积极的影响。笔者也期待能够创建更好、更精巧的设计。更为重要的是，笔者希望本书能帮助编程人员从日常的工作负担中解脱出来，并将快乐建立在所希望看到的地方：即编写伟大的程序代码。

本书的读者对象

本书的读者对象是经验丰富的(中高级的)Visual Basic .NET 开发人员，他们希望被引入到重构世界中。为了充分利用本书，读者应该很好地掌握 Visual Basic .NET，尤其是通常的面向对象的编程。

如果您还只是一个编程方面的初学者，那么无法将本书用作初级入门书。本书的目的不是传授用 Visual Basic .NET 编程的基本方法。然而，在职业生涯中不尽早地熟悉重构，是没有任何理由的。随着学会编写第一个类，您就可以使用本书来学会如何正确地设计并纠正可能引入到设计中的任何错误。

如果您是一个 VB 6 的编程人员，那么因为 Visual Basic 6(或之前的)编程语言与 Visual Basic .NET 之间存在差距，以致将无法成功地运用本书中讨论的很多重构。然而，在这种情况下，可以尽量将 VB 6 的代码升级到 VB .NET 中，此时您就会发现本书将非常有用。笔者花费了整整一章的篇幅(本书最后一章)来讨论将 VB 6 代码升级到.NET 的话题。然而，该章本身就是建立在本书前面部分所讨论主题的基础上的。

此外，可能还会发现本书在从 VB 6 到 Visual Basic .NET 转换的时候也非常有用。本书介绍新手在从 VB 6 转到 VB .NET 时可能会犯的错误，并给出了解决这些问题的方法。

本书没有对应用程序的类型或域做出任何假设。所以，这个应用程序可能是通常的 Web 应用程序、Web 服务、架构、组件、购物车应用程序、新的 Facebook 小部件或射击游戏，但是不管是什么应用程序，只要其中存在 VB 代码，那么都将会发现本书所阐释的技术是很有价值的。

本书所讨论的大部分重构都是适用于所有完全面向对象语言中的标准重构。这意味着如果使用其他某种面向对象的语言(如 C#)进行编程，那么只要熟悉 VB .NET 语法且能够阅读代码示例，就能够使用和运用本书所提供的信息。

本书的主要内容

本书将详尽地介绍 Visual Basic 中的重构。本书将介绍如下基本的重构概念：

- 代码味道
- 重构代码转换
- 一些基本的面向对象的原则
- 可以自动化重构过程的重构工具的使用

本书只使用了一个示例研究(对于本书的示例研究而言，该示例的规模相对较大)来阐述重构在较大代码基上的实际应用。

除了适用于任何面向对象语言的标准重构之外，本书还介绍了一些 VB .NET 专有的重

构。此外，本书还介绍了重构的一些特殊用法。例如：

- 作为 VB 6 到 VB .NET 升级不可分割的一部分的重构
- 用来将 VB .NET 2005 代码升级到 VB .NET 2008 的重构
- 用来实现设计模式的重构

本书包含了大量重要的味道和重构。然而，本书并没有给出完整的重构目录；由于时间和篇幅的限制，一些重要的重构并没有介绍。例如，本书并没有讨论诸如 Simplify Conditional 或 Reverse Conditional 之类的重构，而这些用于自动化的重构已经在 Developer Express 的 Refactor!插件中可供使用。此外，本书也没有讨论很多“逆向”重构，如 Inline Method 或 Inline Class。

本书的首要目标是介绍重构。笔者的经验表明编程人员刚开始接触重构时首先需要处理的问题是结构化很差的代码问题，而诸如 Extract Method 或 Extract Class 之类的重构将帮助解决这一问题。与其相对的，Inline Method 和 Inline Class 重构将帮助处理过分结构化的代码。这些重构将帮助消除不再需要的构造(如方法或类)。这种情况常常在对代码基运用了广泛的重构之后才发生。

然而，这并不意味着“逆向”重构不重要。一旦已经掌握了基本的重构之后，对这些重构的需要可能就在了解重构之后出现。其中部分学习过程是让人认识到学无止境。例如，通过每个新版本的 Refactor!插件，就将新的重构添加到一批被支持的重构中。人们不断地发明和修改重构。随着逐渐熟悉该技术，读者将会发明自己的重构并可能最终决定与其他人共享这些重构。

因此，笔者建议读者跳出本书的范围；不要局限于本书中所罗列的重构。寻找和发明新的重构。这样，将能真正掌握持续的代码改进的艺术。

本书的组织结构

因为本书是第一本专门处理 Visual Basic 中重构的书籍，也可能是读者阅读的第一本关于重构的书，笔者希望本书完成如下多个目标：

- 详尽地介绍 Visual Basic 中的重构。
- 详尽地介绍日常编程会议中可能会咨询到的重构技术和代码味道。
- 阐述如何通过一个示例的学习，在现实情形中运用重构技术。这个示例就是 Rent-a-Wheels 应用程序，在整本书的每一章中都将分析和修改该应用程序。

为了达到上述目标，本书就像其他的技术书籍一样叙述了各个主题。在介绍和详细阐述新的概念时，也是从逻辑上由基本到复杂逐步推进。为了达到简化的目的，本书还为每个概念给出了用于举例的代码示例。通过这种方法，可以从头到尾按照一种逻辑顺序来阅读本书。在得到本书并首次阅读之后，可能会发生这种情况。

除了主要的叙述部分之外，还将看到本书中散布着很多味道、重构和面向对象设计原则的所谓的“定义框”。这些定义的目的是提供相应主题的综述。例如，对于味道而言，其定义包含了关于如何发现该味道的试探法。对于重构的情况而言，有一个部分称作“技巧(Mechanics)”，它包含了特定的方法，并指明了为高效的执行特定重构而必须完成的步骤。

在日常工作期间，您就可以通过考虑这些技巧来提醒自己如何完成该重构、如何发现某种味道以及使用什么重构来消除味道等。

大部分章节的末尾都将讨论重构、味道以及原则是如何反映到本书所包括的示例研究中。读者可以下载每一章中的代码并查看当前一章中的示例发生了何种变化。示例研究的目的是展示重构一个更现实的应用程序。在很多技术性的书籍中，常常会发现一些选中的代码样本，为了论证观点，已经对该样本进行了不太实际的简化。虽然这样的处理让示例更加清楚且更易于编者证明其观点，但是这常常意味着读者将碰到现实世界中更复杂的情形，而且如果可以在生产代码上运用特定的技术，则可能会产生无法预计的障碍。本书试图通过 Rent-a-Wheels 示例研究来向读者呈现一个更为贴近实际的情形。

本书分成了 5 个部分，这些部分将引导读者按照逻辑顺序从较为基本的概念进入到较为复杂的概念中。

- 第 I 部分“重构的介绍”为本书奠定了基础。例如，在第 1 章中，从一般的角度讨论了有关重构的内容。本章还澄清了一些关于重构的错误理解。第 2 章则立即提供了在实践中重构的初次尝试。随后，第 3 章仔细介绍了为自动化重构工作而必不可少的工具。第 4 章则提供了整本书中使用的一个示例研究。
- 第 II 部分“VB 重构的初步”涵盖了一些初步的重构(如无用代码的消除)以及其他一些帮助您为大修准备代码的重构。此外，还将讨论一些只有 VB 中才出现的问题，如严格类型化与弱类型化、遗留的错误处理与结构化错误处理等。
- 第 III 部分“标准重构转换的初步介绍”主要讨论了一些核心的标准重构。本部分介绍了为代码构造选择名称的重要性以及重复代码可能带来的毁灭性的后果。本部分还讨论了诸如 Extract Method 之类的标准重构并详细介绍了方法层次上的代码结构化。
- 第 IV 部分“高级重构”主要讨论了一些高级重构，这些重构最大限度地利用了编程环境中面向对象的能力。对于本书这部分而言，优秀的面向对象的技术是至关重要的。这一部分将介绍如何发现类、创建继承的层次结构以及重新组织大规模的代码等。
- 第 V 部分“重构的运用”将介绍为了达到具体的目标如何成功地运用重构。例如，如何将重构与设计模式配合使用以生成更为精密的设计方案。本部分将展示一些与 VB 的 Visual Studio 2008 版本一起装载的新功能以及如何才能使用重构来最大化地利用它们的功能。最后，将简单回顾 Visual Basic .NET 的前驱者——VB 6，并介绍如何运用重构将 VB 6 代码升级到 Visual Basic .NET。

使用本书的条件

为了能成功地使用本书，需要如下的软件：

- Visual Studio .NET 2008——为了能利用 Refactor!插件，至少需要安装 Professional Edition。在 Visual Basic 2008 Express Edition 中，仍然可以调试、执行代码，并手动地完成重构。如果使用的是 Visual Studio 2005，那么也可以利用本书大部分的

内容，但第 15 章除外，因为这一章专门讨论了只有 Visual Basic 9 和 Visual Studio 2008 才提供的功能。

- Developer Express 中的 Refactor! for VB 插件——从 Developer Express 站点上下载免费的 Visual Studio 插件的最新版本，网址为：www.devexpress.com/vbrefactor/。
- Microsoft SQL Server——为了运行本书中包含的样本示例研究，需要安装 Microsoft SQL Server。安装 MS SQL Server 2000 或 2005，对于本书介绍的应用程序而言，Express 版本已经足够。可以使用任何可以运行上述软件的操作系统，如 Windows XP、Windows Server 2003 或 Windows Vista。

味道、重构以及面向对象设计原则的方框

在本书中，将碰到 3 种位于方框中的文本：味道、重构以及面向对象的设计原则。

- 味道——这些方框包含了关于代码味道概括性定义。代码味道是一个非常重要的重构概念，每个方框包含如下 3 个部分：
  - 检测该味道——详细介绍一些简单的试探法来检测代码中的代码味道。
  - 相关的重构——列出为消除该味道可以使用的重构。
  - 基本原理——以更理论化的术语来解释代码味道的消极效果。
- 重构——这些方框给出的内容包括重构的基本内容。每个方框包括如下部分：
  - 动机——解释该重构带给代码的好处以及如何改进应用程序的设计。
  - 相关的味道——列出该重构可以帮助消除的味道。
  - 技巧——提供完成该重构的逐步方法。
- 面向对象的设计原则——在这些方框中，定义了一些重要的面向对象设计原则并常常使用一些简短的代码样本来阐释它们。

源代码

在读者学习本书中的示例时，可以选择手动输入代码或者使用本书附带的源代码文件。本书中用到的所有源代码也可以从 www.tupwk.com.cn/downpage 和 www.wrox.com 下载。进入站点 http://www.worx.com 后，只需找到本书的书名(使用 Search 框或者书名列表)，单击本书详细信息页面上的 Download Code 链接，就可以得到本书所有的源代码。

注意：

因为很多书的书名都很相似，所以用 ISBN 搜索更为容易。本书英文版的 ISBN 是 978-0-470-17979-6。

下载了代码后，用您喜欢的压缩工具把它解压缩。此外也可以去 Wrox 的主下载页面 www.worx.com/dynamic/books/download.aspx 找到本书或 Wrox 出版的其他书的代码。

勘误表

尽管我们竭尽所能来确保在正文和代码中没有错误，但人无完人，错误难免会发生。如果您在 Wrox 出版的书中发现了错误(如拼写错误或者代码错误)，我们将非常感谢您的反馈。发送勘误表将节省其他读者的时间，同时也会帮助我们提供更高质量的信息。

请给 wkservice@vip.163.com 发电子邮件，我们会检查您的反馈信息。如果是正确的，我们将在本书的后续版本中使用。

如果您在本书的勘误表页面上没有看到您发现的错误，可以到 www.wrox.com/contact/techsupport.shtml 上填写表单，把您发现的错误发给我们。我们会检查这些信息，如果属实就把它添加到本书的勘误表页面上，并在本书随后的版本中更正错误。

p2p.wrox.com

如果想和笔者或者其他人讨论，请加入 http://p2p.worx.com 上的 P2P 论坛。该论坛是基于 Web 的系统，您可以发布关于 Wrox 出版的图书和相关技术的消息，与其他读者或技术人员交流。该论坛有预定功能，当您感兴趣的主题有新帖子发布时，系统会邮件通知。Wrox 的作者、编辑、其他业界专家和像您一样的读者都会出现在这些论坛中。

在 http://p2p.wrox.com，您会找到很多不同的论坛，它们不但有助于您阅读本书，还有助于您开发自己的应用程序。加入论坛的步骤如下：

(1) 进入 http://p2p.wrox.com，单击 Register 链接。

(2) 阅读使用说明，并单击 Agree 按钮。

(3) 填写加入该论坛必需的信息和其他您愿意提供的信息，单击 Submit 按钮。

(4) 您将收到一封电子邮件，描述如何验证您的账户和完成加入过程。

注意：

不加入 P2P 也可以阅读论坛里的消息。但是如果要发布自己的消息，就必须加入。

加入之后，就可以发布新的消息和回复其他用户发布的消息。可以随时在 Web 上阅读论坛里的消息。如果想让某个论坛的新消息以电子邮件的方式发给您，可以单击论坛列表中论坛名称旁边的 Subscribe to this Forum 图标。

要了解如何使用 Wrox P2P 的更多信息，请阅读 P2P FAQ，其中回答了论坛软件如何使用的问题，以及许多与 P2P 和 Wrox 出版的图书相关的问题。要阅读 FAQ，单击任何 P2P 页面里的 FAQ 连接即可。

目　录

第Ⅰ部分　重构的介绍