精通AspectJ

WILEY

面向方面的Java编程指南

Joseph D. Gradecki 著
Nicholas Lesiecki
王欣轩 吴东升 等 译

清华大学出版社

精通 AspectJ

Joseph D. Gradecki
Nicholas Lesiecki
著

王欣轩　吴东升 等　译

清华大学出版社
北京

Joseph D. Gradecki & Nicholas Lesiecki

Mastering AspectJ

EISBN: 0-471-43104-4

图书在版编目(CIP)数据

精通 AspectJ/格雷德斯基(Gradecki, J.D.),莱丝斯基(Lesiecki, N.)著;吴东升,王欣轩等译.—北京:清华大学出版社,2005.1

书名原文:Mastering AspectJ

ISBN 7-302-10161-2

Ⅰ.精… Ⅱ.①格…②莱…③吴…④王… Ⅲ.JAVA 语言—程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2004)第 141816 号

出 版 者: 清华大学出版社 **地 址**: 北京清华大学学研大厦

http: //www.tup.com.cn **邮 编**: 100084

社总机: 010-62770175 **客户服务**: 010-62776969

责任编辑: 常晓波

封面设计: 立日新

印 刷 者: 清华大学印刷厂

装 订 者: 北京鑫海金澳胶印有限公司

发 行 者: 新华书店总店北京发行所

开 本: 185 × 230 **印张**: 24.5 **字数**: 548 千字

版 次: 2005 年 1 月第 1 版 2005 年 1 月第 1 次印刷

书 号: ISBN 7-302-10161-2/TP·1073

印 数: 1 ~ 3000

定 价: 48.00 元

导　言

大多数应用程序(特别是企业级的应用程序)都不仅仅是存在于某个文件中的单一代码模块。应用程序是模块的集合,这些模块相互配合以提供某些预期功能,而这些功能是由一组需求定义的。从理论上讲,开发人员可以构建含多个分散功能的模块。这样,一个安全模块和一个登录模块就可与一个 HTTP 模块结合在一起,用以创建出 Web 服务器应用程序。

然而,由于面向对象工具(及语言)的特性,实际上很少能实现这种理想的模块化编程。相反,开发人员常常必须创建多个含有混合目标的模块。像用户登录这样的一项简单功能,实际上会分配到应用程序的多个模块中。Apache Web 服务器就是这种情况,例如 Apache 有 43 个模块,而其中 37 个模块都含有处理用户登录的代码。这种做法导致了纠缠不清的代码,从而代码易出错也难以调试、重构、文档化和进行技术支持。

AOP(Aspect-Oriented Programming, 面向方面的编程)的目标就是解决这类开发问题。AOP 强调 aspect 的构建,aspect 就是将分散式功能集中起来的模块。为了在实际中运用 AOP 理论,Polo Alto Research Center(PARC)创建了 AspectJ。AspectJ 是一种作为 Java 扩展的开放源代码语言。AspectJ 与 Java 完全兼容:AspectJ 中的 aspect 与 Java 中的类互相配合,以实现综合的应用程序。通过 Java 使用 AspectJ 的优势包括以下内容。

- 使纠缠不清的代码更少
- 使代码更短
- 使应用程序的维护及发展更容易
- 使应用程序更容易调试、重构及修改
- 使代码更容易复用——开发人员可通过与以前创建 OOP 对象库类似的方法来创建 aspect库

本 书 内 容

本书着眼于 AOP 范例的全面介绍,可作为一本完整的 AspectJ 语言手册,以及针对客户当前和未来项目的 AOP 和 AspectJ 使用指导。下面是阅读本书所要了解的一些概念。

- **连接点(join point)**——执行应用程序过程中的一个可预见的点
- **pointcut**——一种专门用来识别并选择 AspectJ 程序中的连接点的结构
- **advice**——到达应用程序代码中的连接点处时要执行的代码

- **类型间声明(inter-type declaration)**——一种向以前建立的类中添加属性及方法的强大机制
- **aspect**——一种类似于 Java 类的结构,对连接点、pointcut、advice 及类型间声明进行封装

通过代码片断和全面解释,对所有的 aspect 概念进行说明。掌握了AspectJ 的概念之后,我们再给出大量的例子来阐述如何利用 AOP 解决实际问题。

本书的 Web 站点提供了书中的所有代码,网址是 www.wiley.com/compbooks/gradecki。网站上有一个包含每个章节的代码的压缩文件。本书中的示例是通过 Java SDK 1.4 版本创建的(必须安装 Java SDK)。除了 Java 之外,还要有 AspectJ 编译器及运行时组件(这些组件可以在 www.eclipse.org/aspectj 上找到)。

本书适用的读者

若你目前正在设计、开发软件应用程序,就应该阅读本书。所有软件都包含一些要求,这些要求超出了应用程序的最基本用途,并使应用程序的设计及编码复杂化。由于 AspectJ 构建于 Java 之上,因此我们假定你在 Java 语言方面拥有很多知识。

本 书 结 构

本书分成 3 部分。第 1 部分全面概述了 AOP 以及一种支持 AOP 的语言的发展。第 2 部分讨论了 AspectJ 的主要概念,解释这些概念时我们使用了短小的代码示例。第 3 部分则利用第 1 部分及第 2 部分的概念,将这些概念实际应用到软件项目的开发之中。以下是各章的介绍。

第 1 部分:AOP 初探

第 1 章:AOP 简介

该章全面概述了什么是 AOP、AOP 起源于何处、如何利用 AOP 等问题。该章首先完整地讨论了面向对象编程以及有关的一些问题(这些问题是目前开发过程中无法解决的)。之后,该章又简要分析了 AOP,以及如何应用 AOP 来解决关注点分离的问题。

第 2 章:实现 AOP

第 2 章阐述了如何把 AOP 概念从理论转化为具体的编译器支持。这方面的重点在于代码交织,涉及编译时在连接点处添加 aspect 代码。

第 2 部分:AspectJ 简介

第 3 章:AspectJ 的获取和安装

通过讨论如何获取并在系统上安装 AspectJ,第 3 章首先分析了 AOP 的实际特性。该章给出了一个简单的示例,它有助于判定系统安装是否正确。由于 AspectJ 构建在 Java 之上并与 Java 一起工作,因此该章对 Windows 安装以及 Linux 安装都进行了说明。

第 4 章:实现 AspectJ

对于学习任何一种新语言或新规范来说,利用示例可大大简化学习过程。该章讨论了 Hello World 应用程序。这个程序涵盖了主要的概念和编译机制,并对本书后面的其他内容进行了简要说明。

第 5 章:AspectJ 连接点

AspectJ 中最重要的概念就是连接点(join point)。连接点是应用程序中定义明确的一个点,可作为 AspectJ 代码的触发器。该章首先分析了 AspectJ 中使用的"动态连接点"(Dynamic Join Point)模型。AspectJ 提供了多种不同的连接点类型,以处理方法、构造函数、属性获取、属性设置等结构。签名(signature)是连接点的定义部分。指定签名时可使用纯文本、模式或通配符。该章用了很多代码片断来说明各种连接点。

第 6 章:AspectJ pointcut

pointcut 是一种选择连接点集合的语言结构。第 6 章利用你在编写连接点时学到的知识,示范了 pointcut 的构建过程,这些 pointcut 将实现涉及主要应用程序的各种要求。在每个用户定义的 pointcut 中,有很多不同的指示符,这些指示符规定了什么时候应该匹配连接点。该章还讨论了反射(reflection)的用法,反射允许 AspectJ 代码查看连接点的上下文。

第 7 章:advice

一旦用 pointcut 选出了程序中所关心的点(比如,以 set 开头命名的方法的所有调用点),你就可以用 AspectJ 定义在连接点之后、之前运行或替代该连接点运行的 advice 代码。比如,可以对所有设定方法的执行进行记录,或者检查参数的合法性。该章说明了如何定义 advice,并且讨论了 advice 的不同类型和用途。

第 8 章:类型间声明

除了提供连接点及 advice 之外,AspectJ 还通过支持类型间声明扩展了 Java 类型系统。类型间声明使你可以向该类型以外的类(甚至接口)添加方法或属性。这种向接口添加具体功能的能力提供了一种多继承类型。类型间声明也使得 aspect 可以将添加继承层次的新接口或父类型转变成系统中的任何类型。该章讨论了 AspectJ 类型间声明的机制及用法,并说明了以前结构的大规模能力拓展方式。该章还谈到了两种不太引人注意(但仍然重要)的功能:异常软化以及自定义编译错误。

第 9 章:aspect

AspectJ 代码是连接点、pointcut、advice 及类型间声明的结合。将这种新式代码扔到你正设法从混乱代码纠缠中保存下来的同一应用程序中,将可能会产生相反的效果。aspect 是 AspectJ语言提供的一种语言结构,用于封装 AOP 所需的复合结构。该章综合分析了 aspect,并通过许多不同示例和情形说明如何构建 aspect。

第 3 部分:使用 AspectJ

第 10 章:AspectJ 的开发应用

该章首先展示了 AspectJ 的一些应用。而此章讨论的主题包括 AspectJ 应用、测试应用、常见关注点、生成 aspect 以及性能调试。

第 11 章:使用 AspectJ 工具

AspectJ 的部分功能在于它对通用集成开发环境(Integrated Development Environment,IDE)

的辅助性扩展。目前的系统包括了对 JBuilder、Forte、Eclipse 及 Emacs 等的扩展，该章还对这些扩展的应用进行了详细说明。此外，AspectJ 也提供了一种结构浏览器，利用它可全面了解应用程序及 aspect 代码。该章还讨论了 AspectJ 代码的调试以及 Ant 在构建、编译应用程序中的应用。

第 12 章：错误处理和一般问题

该章分析了使用 AspectJ 时常见的编译错误和运行时错误，既详细阐述了错误的根源，也提供了相应的解决方案。

第 13 章：面向 Aspect 例子：模式与重用

该章通过两个规模适中的示例说明了如何利用 aspect 将连贯的功能层添加到应用程序中。这两个示例结合了 pointcut、advice 及类型间声明，说明了如何在域对象中增加持续性、自动缓存无效性等特性，而不用修改这些域对象。第 2 个例子展示了 Observer 模式的一个可复用 aspect 版本。该章通过分析这些例子的含义对组件复用进行了总结。

第 14 章：AspectJ 在现实中的应用

该章综合了在整本书中学到的所有概念，将这些概念应用到两个实际方案中。第 1 个方案涉及 AspectJ 和 AOP 在添加新应用程序功能时的应用，主题包括：aspect 设计理念、文档处理、aspect 编码。第 2 个方案涉及一个开放源代码项目的重构，以使用 aspect 实现计时功能和日志功能。

附录 A：AspectJ API

附录 A 给出了 AspectJ 语言的完整应用程序编程接口（Application Programming Interface，API）。AspectJ 1.1 版支持这个 API。

附录 B：有帮助的 Web 站点

附录 B 提供了包括 AOP 和 AspectJ 的重要 Web 站点列表。

附录 C:其他 AOP 语言绑定

Java 并不是惟一一种受益于应用 AOP 概念的语言。开放源代码社区中有很多项目对 C、C++ 、Ruby 及其他语言提供了 AOP 支持。附录 C 给出了其他语言项目的概要,并说明了参考信息的出处。

献　辞

谨以此书献给我主(基督)、我妻以及我们的儿子。

——Joseph D. Gradecki

谨以此书献给 S.H.。

——Nicholas Lesiecki

致　谢

感谢 Tim Ryan、Tiffany Taylor、Liz Welch、Nick Lesiecki,以及许多花费大量时间审阅本书手稿的人。

——Joseph D. Gradecki

首先,要感谢我的妻子 Suzanne,在创作本书的几个月中,身为丈夫的我却一直未能陪伴在你身边。这本技术手册前面的寥寥数语难以报答你所做的默默牺牲。其次,要感谢 Arno Schmidmeier 和 Andrew Barton,你们审阅了本书的手稿,在那段紧张的日子里你们表现出很大的耐心,也为这本书的手稿提供了宝贵的建议和技术反馈。正是由于你们的付出,这本书才有了明显的改观(对于书中还存在的任何错误我们负有责任)。第三,我要感谢 AspectJ 小组(不管是过去的还是现在的),正是因为你们,这本书的 raison d'etre(存在的理由)才得以发表,我还要感谢你们耐心地回答我的问题(并采纳我的建议)。在此,我要特别感谢 Jim Hugunin、Erik Hilsdale、Ron Bodkin、Gregor Kiczales,尤其要感谢 Wes Isberg,感谢你们的帮助与支持。另外再提一句,我要感谢 Jan Hannemann 和 Vincent Massol,写作本书时用到了你们投给 AspectJ 社区的稿子。最后,要感谢我的小猫 Juno,感谢你在我创作时乖乖地趴在我的大腿上。

感谢阿姆斯特学院(Amherst College)英语系:

此书不愧是一本精品。

——Nicholas Lesiecki

关 于 作 者

Joseph D. Gradecki 是 Comprehensive Software Solutions 公司的一位软件工程师，从事于 SABIL 产品方面的工作（SABIL 产品是一种企业级的安全处理系统）。Gradecki 利用 Java、AspectJ、Servlet、JSP、Resin、MySQL、BroadVision、XML 等构建了大量动态的企业级应用程序。Gradecki 是 Mastering JXTA 的作者，同时还是 MySQL and Java Developer's Guide 的合著者（另两位合著者是 Mark Matthews 与 Jim Cole）。Gradecki 持有计算机科学专业的学士学位及硕士学位，目前正在进修计算机科学的博士学位。

Nicholas Lesiecki 喜欢在"域"（in the zone）中编码——在这个"域"中，代码就像水一样从手中流过，整个团队共享一种理念，做出的结果让自己和客户都皆大欢喜。对实现这种"域"的向往促使 Lesiecki 采用"极限编程"（Extreme Programming）这类技术和 AspectJ 类的工具。Lesiecki 是最畅销的 *Java Tools for Extreme Programming* 一书的合著者（另一位合著者是 Rick Hightower）。Lesiecki 还致力于 Cactus 单元测试框架的工作。最近，他写了有关 AspectJ 和 IBM 的 developer-Works 测试的文章。Lesiecki 效力于 eBlox 公司，是亚利桑那州图森（Tucson）市的首席软件工程师。

目　录