Broadview
www.broadview.com.cn

Java技术大系


# Spring 2.0 核心技术与最佳实践

廖雪峰 编著

本书的全部源代码和Eclipse的视频教程 1.6GB DVD

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

TP312
2393D
2007

Java技术大系

# Spring 2.0 核心技术与最佳实践

廖雪峰 编著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

## 内 容 简 介

本书注重实践而又深入理论，由浅入深且详细介绍了 Spring 2.0 框架的几乎全部的内容，并重点突出 2.0 版本的新特性。本书将为读者展示如何应用 Spring 2.0 框架创建灵活高效的 JavaEE 应用，并提供了一个真正可直接部署的完整的 Web 应用程序——Live 在线书店。

在介绍 Spring 框架的同时，本书还介绍了与 Spring 相关的大量第三方框架，涉及领域全面，实用性强。本书另一大特色是实用性强，易于上手，以实际项目为出发点，介绍项目开发中应遵循的最佳开发模式。

本书还介绍了大量实践性极强的例子，并给出了完整的配置步骤，几乎覆盖了 Spring 2.0 版本的新特性。

本书适合有一定 Java 基础的读者，对 JavaEE 开发人员特别有帮助。本书既可以作为 Spring 2.0 的学习指南，也可以作为实际项目开发的参考手册。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

**图书在版编目（CIP）数据**

Spring 2.0 核心技术与最佳实践 / 廖雪峰编著. —北京：电子工业出版社，2007.6
（Java 技术大系）
ISBN 978-7-121-04262-1

I. S… II. 廖… III. JAVA 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字（2007）第 056302 号

责任编辑：韩 明
印 刷：北京天宇星印刷厂
装 订：北京牛山世兴印刷厂
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
开 本：787×1092 1/16 印张：32.75 字数：681 千字
印 次：2007 年 6 月第 1 次印刷
印 数：5000 册 定价：59.80 元（含光盘 1 张）

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

# 前　言

Java 开发已经走过十年了！随着因特网的飞速发展，Java 技术获得了前所未有的广泛应用。从桌面系统到企业应用，从手机到智能卡，处处都能看到它的身影。从 1998 年 Sun 公司发布 JavaEE 1.0 版本开始，在此后的短短几年内，JaveEE 获得了巨大的发展，几乎成为企业开发的代名词。

作为 JavaEE 中最核心的 EJB 技术，也一度成为 JavaEE 应用的核心。不幸的是，EJB 在带来了全新的企业级开发模型的同时，也带来了不必要的复杂性：复杂的接口，难于测试和部署。越来越多的开发人员不断反思 EJB 开发的复杂性，并试图以更简单的 Java 技术来简化 JavaEE 应用的开发。Rod Johnson 总结了他数年的 JavaEE 项目经验，在《Expert-One-on-One: JaveEE Design & Development》一书中详细阐述了 EJB 带来的复杂性，并提出了一系列以轻量级框架为核心的全新的 JavaEE 设计思想，阐述了如何组合一系列现有的技术并形成了一个初步的框架，这个框架后来便发展为 Spring Framework。通过 Spring 这个轻量级框架，我们终于可以轻松实现过去必须使用复杂而烦琐的 EJB 才能实现的功能。

Spring 提出了以 JavaBean 为组件模型、针对接口编程、使用依赖注入等许多优秀的设计思想，并且 Spring 可以无缝整合许多流行的框架，如 Struts、Hibernate 等。人们很快意识到以 Spring 框架为基础来开发 JavaEE 应用程序可以大大简化应用程序的设计、调试和部署，并得到一个松散耦合的系统架构。因此，Spring 得到了越来越广泛的应用。随着 Spring 2.0 版本的推出，添加了大量新的功能，进一步强化了 Spring 框架在轻量级 JavaEE 开发领域的主导地位。

## 本书特色

本书以 Spring 2.0 版本为标准，试图向读者展示 Spring 框架的奥秘，引导读者由浅入深、一步一步地掌握 Spring 框架的使用方法和设计思想。此外，本书还特别注重实践，力图给出能够在实际项目中应用的解决方案，并给出完整的示例代码。在本书的最后一章中，还详细介绍了基于 Spring 2.0 框架设计并实现的一个完整的 Web 应用程序——Live 在线书店，并给出了许多有用的设计模式和技巧。

在介绍 Spring 框架的同时，本书也试图介绍与 Spring 相关的大量第三方框架，涉及领域全面，实用性强。例如，作为 ORM 框架的 Hibernate、iBatis、JDO 及最新的 JPA；作为 Web 框架或组件的 Struts、WebWork2、Velocity、FreeMarker、Tiles、JSF；作为 Web 服务框架的 Axis 和 XFire；作为安全框架的 Acegi 等。本书对每个框架的集成都给

出了完整的 Eclipse 工程，这些示例可以直接作为基本的配置模型并应用到实际项目中。

本书另一大特色是实用性强，以实际项目为出发点，介绍项目开发中应遵循的最佳开发模式。例如，在开发 Web 服务时，不是从编写复杂的 WSDL 文件入手，而是首先设计接口，然后采用 Java 5 注解来实现 Web 服务的自动部署；在集成 Hibernate 时，不是从编写配置文件或创建数据库结构入手，而是首先设计 Java 实体对象，然后通过 Java 5 注解并配合 Ant 自动完成数据库表的创建，这些都符合实际项目的开发。

本书还介绍了大量实践性极强的例子，并给出了完整的配置步骤，例如，基于泛型的 DAO 体系设计，结合 Lucene 和 Compass 实现全文搜索功能，利用 CAS 架设单点登录服务器，利用 JMX 实现对应用程序的远程监控，利用 Filter 实现无侵入的页面缓存等，这些都是在实际项目开发中经常需要用的，本书均给出了能够直接运行的配置，并配合屏幕截图尽量详细地给出配置步骤，能够最大限度地让初学者无痛起步。

本书的代码注释也非常详细，并且在书中尽量采用中文注释，便于初学者理解。对于许多复杂的模块设计，本书总是给出流程图或关系图，让读者从设计上能更好地整体把握。

本书还几乎覆盖了 Spring 2.0 版本的新特性，包括使用 AspectJ 5 注解实现 AOP、对 JPA 的完整支持、新的声明式事务配置方式、对动态语言的支持等。在选择某种解决方案时，优先考虑采用 Spring 2.0 的新特性并尽量使用 Java 5 注解进行配置，这也是本书有别于其他介绍 Spring 1.x 书籍的地方。

## 主要内容

本书按照由浅入深、从理论到实践的顺序来安排内容，主要包括以下内容。

**第一部分**：介绍 Spring 的诞生和主要功能，并指导读者在 Eclipse 中编写一个具体的 Spring 应用程序，以便读者能对 Spring 有一个初步认识。

**第二部分**：分别介绍 Spring 的各主要功能模块，按照由浅到深及各模块的依赖关系，首先介绍作为整个 Spring 框架核心基石的 IoC 容器，然后分别介绍 Spring 的 AOP 支持、数据访问策略、事务管理及 Web MVC 模块。紧接着介绍 Spring 框架的一些非核心但是可能在实际项目中应用的模块，包括远程访问、任务调度、JMS 支持、JMX 支持、动态语言支持及 Acegi 安全框架，读者可以根据实际需要有选择地学习。通过第二部分的介绍，读者能全面掌握 Spring 框架的几乎所有内容。

**第三部分**：开发一个完整的基于 Spring 框架的 Web 应用程序——Live 在线书店。这一部分详细介绍了如何利用 Spring 设计并实现一个多层 JaveEE 应用程序。在项目开发中，大量应用实际项目的开发方式，包括使用 Ant 作为构建工具，使用 XDoclet 自动生成配置文件等。在 Live 在线书店的实现细节上，还介绍了许多有用的模式和技巧，包括内存和静态文件的缓存模型、一些 JavaScript 技巧、应用 JMX 检测系统性能等。读者



此为试读，需要完整PDF请访问：www.ertongbook.com

完全可以将其应用到实际的项目开发中。

需要注意的是，本书中的图例并不是完全按照 UML 标准绘制的，图例的设计目的是为了突出问题并试图以最清晰的方式展示给读者，因此，读者不必有 UML 相关知识，只需明白图例的意义即可。

## 读者对象

本书适合有一定 Java 基础的读者，对 JaveEE 开发人员更是特别有帮助。本书既可以作为 Spring 2.0 的学习指南，也可以作为实际项目开发的参考手册。

本书不仅希望读者能掌握 Spring 框架的使用方法，还试图阐述 Spring 框架的实现原理，因此，许多章节都会涉及一些底层实现，例如，AOP 和 MVC 的手动实现方法。不理解这些原理虽然也不会影响 Spring 的学习，但是却失去了了解 Spring 框架底层运行机制的机会，也就无从学习 Spring 框架的设计思想。因此，强烈建议读者在掌握了 Spring 框架的使用方法后，更深入到 Spring 框架内部，最好能结合 Spring 源代码学习 Spring 的设计思想。如果在脱离 Spring 的环境下也能自然而然地应用 Spring 优秀的设计思想，例如，始终坚持针对接口编程，使用依赖注入，那才算真正掌握了 Spring 框架的精髓。

在本书的写作过程中，得到了家人和朋友的大力支持。在此，我要特别感谢我的妻子对我的大力支持，我还要感谢同事李江华，他为本书第 5 章的示例编写了 Swing 界面，最后，我还要感谢为本书提出宝贵意见的朋友和同事。

廖雪峰

2007.2.14

# 作者简介

廖雪峰，具有5年Java/J2EE/J2ME开发经验，早在大学本科时就参与了网易网上商城（http://mall.163.com）的开发，目前在摩托罗拉软件集团担任软件工程师，从事基于Eclipse的可视化建模工具的设计和开发。

目前，廖雪峰创建了国内讨论JavaEE技术的专业网站：JavaEE开发网（http://www.javaeedev.com），读者可以在JavaEE开发网的论坛中对本书提出中肯的批评和意见，作者将尽最大努力回复读者提出的问题。

作者的Blog地址是http://xuefeng.javaeedev.com，欢迎访问。

本书的勘误表也将在JavaEE开发网论坛中不定期发布，请读者关注。

# 目 录 CONTENTS

此为试读，需要完整PDF请访问：www.ertongbook.com

# 第 1 章

# 初识 Spring

## 1.1 JavaEE 平台的诞生和发展

Java 语言从诞生之日起，就受到了广泛关注。这个崭新的语言拥有许多优秀的特性：平台无关、垃圾回收机制、抛弃了 C/C++的指针、强大的网络处理功能和完全面向对象的开发模型。虽然许多特性并非由 Java 首次实现，然而，Java 诞生之日正是因特网开始发展之时。人们很快意识到 Java 语言的这些优秀特性非常适合快速开发基于因特网的健壮的分布式应用，尽管 Java 设计之初的本意是针对嵌入式设备。

今天，越来越多的企业开发人员希望能快速开发安全可靠的、可扩展的分布式企业应用，尤其是以浏览器为前端的 Web 应用，并借助因特网将服务尤其是电子商务扩展到全世界的范围。和过去的客户端/服务器（Client/Server）模式相比，基于浏览器/服务器（Browser/Server）模式的 B/S 应用越来越广泛。随着企业应用规模的快速增长，越来越多的企业将 JavaEE 平台作为企业开发的基础。短短的几年时间里，JavaEE 几乎成了企业开发的代名词。

作为 Java 语言的缔造者，Sun 公司也很快意识到企业对于 Java 的巨大需求，因此，在 1999 年底，一个基于标准 Java 虚拟机的企业级 Java 平台 J2EE——Java 2 Enterprise Edition 正式发布了。

随着 J2EE 1.5 标准的发布，Sun 将 J2EE 正式更名为 JavaEE，与此对应，J2SE 和 JavaME 平台也更名为 JavaSE 和 JavaME。考虑到 JavaEE 这一名称使用越来越广泛，在本书中，一律使用“JavaEE”这一术语，而不再使用“J2EE”这一名称，请读者注意。

值得注意的是，JavaEE 并非是一个产品，而是一系列技术和标准的集合。具体的 JavaEE 平台产品由各厂商实现并遵循同一个标准。JavaEE 平台继承了 Java 语言的安全性和高可移植性，为企业应用的设计、开发、部署和管理提供了一套完善的解决方案，它包括了从前端 Web 界面到中间件，再到后端数据库系统的一系列技术和规范。JavaEE 提供了一套标准的 API 和以组件为基础的企业架构，尤其值得注意的是，JavaEE 提出了一个新的“容器”的概念，通过容器来提供标准的系统底层服务，大大降低了企业级开发的复杂度。

概括来讲，JavaEE 包括了一系列针对组件和服务的平台标准和 API 接口，如图 1-1 所示。

多层的分布式架构是 JavaEE 的典型设计模型。通过将系统分割成多层，以便降低各层的复杂性，并实现一个松耦合的结构以便于扩展和维护。典型的 JavaEE 应用是一个三层结构的系统：表示层、业务层和持久层。表示层负责和用户打交道，接受用户输入并将结果显示给用户；业务层负责实现各种商业逻辑，即企业的业务模型；持久层则负责

将业务层的数据保存到永久的外部存储系统中，或者读取数据以供业务层使用。通常，最常用的数据存储系统是关系数据库，因此，持久层最重要的功能便是完成对象/关系映射（O/R Mapping）的实现，即实现系统中 Java 对象和数据库表记录的双向转换。

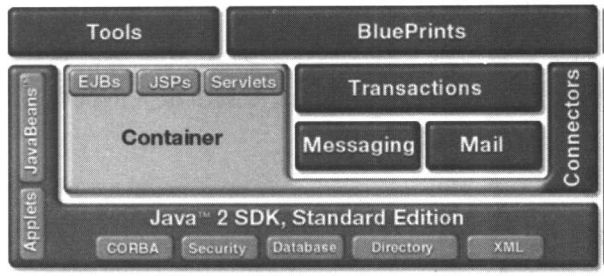


图 1-1

## 1.2 Spring 的起源

在 JavaEE 平台诞生后的日子里，JavaEE 几乎就是企业级开发的代名词，当然，作为 JavaEE 中最核心的 EJB 技术，也一度成为 JavaEE 应用的核心。EJB 第一次提出了声明性事务管理的概念，通过在部署时指定 EJB 组件的声明性事务类型，事务管理被纳入了容器的功能，而不必由开发者来编写冗长的、不易维护的事务代码。这种事务管理模型大大简化了企业应用的开发，也被视为 EJB 最成功的特性。

不幸的是，EJB 在带来了全新的企业级开发模型的同时，也带来了不必要的复杂性：编写 EJB 组件是复杂而困难的。为了实现一个 EJB 组件，除了 Bean 本身的实现类外，还不得不编写额外的 Home 接口和 Remote 接口。大多数时候，为了避免 EJB 远程调用的开销，还常常需要编写 Local 接口。

为每个 EJB 都编写如此复杂的接口需要很多工作量。此外，部署和测试也是令人非常头疼的问题。随着测试驱动开发的流行，人们逐渐意识到单元测试的重要性。实际上，容易被测试的代码往往意味着更容易被维护，也更容易扩展。不幸的是，和普通的 Java 对象相比，EJB 组件更难于测试，因为简单的编码经由测试的开发循环变成了编码，然后进行部署和测试。每次测试时还需要启动 JavaEE 服务器，这中间又加入了一个“等待”的过程。

越来越多的开发人员不断反思 EJB 开发的复杂性，并试图以更简单的 Java 技术来简化 JavaEE 应用的开发。Rod Johnson 总结了他数年的 JavaEE 项目经验，在《Expert-One-on-One: JavaEE Design & Development》一书中详细阐述了 JavaEE 体系尤其是 EJB 带来的复杂性，并提出了一系列以轻量级框架为核心的全新的 JavaEE 设计思想，阐述了如何

组合一系列现有的技术并形成了一个初步的框架，这个框架后来便发展为 Spring Framework。通过 Spring 这个轻量级框架，我们终于可以轻松实现过去必须使用复杂而烦琐的 EJB 才能实现的功能。更重要的是，Spring 抛弃了 EJB 这种重量级组件，以 JavaBean 作为组件实现的一个轻量级框架。由于不再需要 EJB，基于 Spring 的轻量级 JavaEE 应用完全可以抛弃 EJB 服务器，而仅仅需要 Web 服务器即可。更重要的是，以 JavaBean 作为组件模型使得组件的开发和测试得到了极大的简化，而且 Spring 是一个耦合极为松散的框架。

## 1.3 Spring 框架介绍

简单地说，Spring 就是一个实现了 AOP 功能的 IoC 容器，虽然这种说法不太全面，但确实概括了 Spring 框架的核心功能。图 1-2 显示了 Spring 框架包含的 7 个主要模块。

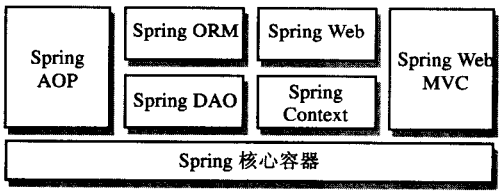


图 1-2

整个 Spring 框架都是轻量级的。和 EJB 这种重量级的组件需要在重量级的 EJB 容器中运行相比，Spring 为普通的 JavaBean 提供了一个轻量级的容器，因此，不需要全功能的 JavaEE 服务器，Spring 容器可以运行在仅支持 Web 容器的 JavaEE 服务器上，或者直接在普通的 main()方法中启动它。

### 1.3.1 Spring 的核心 IoC 容器

如图 1-2 所示，Spring 核心容器是所有其他模块的基础。这个核心的 IoC 容器定义了如何创建、管理和配置 Bean。在一个 Spring 应用程序中，几乎所有的组件都被放到核心 IoC 容器中，并按照某种配置装配起来。作为开发人员，我们需要关心的是组件的编写，以及如何在 Spring 的 IoC 容器中正确地装配它们。

Spring 通过 Bean 工厂来实现基于依赖注入的组件装配。在第 3 章中，将详细介绍如何使用 Spring 的 IoC 容器来装配出应用程序，从而获得强大的灵活性和可维护性。

Spring 提供的应用程序上下文（Application Context）封装了许多基本的系统服务，例如，访问 JNDI、对国际化（I18N）的支持、事件传播、资源装载、电子邮件服务等。

### 1.3.2 Spring 对 AOP 的支持

AOP（Aspect Oriented Programming）即面向切面编程，是近年越来越流行的一种新的编程模式。AOP 的编程思想和 OOP 不同，它是对 OOP 的一种强有力的补充。通过 AOP，在某些情况下能够实现更好的模块化结构，或者可以动态为系统增加新的功能，而不影响原有系统的结构。

Spring 的 AOP 模块提供了 AOP 联盟定义的 AOP 接口的实现，利用 Spring 提供的 AOP 支持，可以简化代码逻辑，分离应用程序关注点。此外，Spring 提供的许多底层服务，例如，对声明式事务管理的支持也是基于 AOP 实现的。

和其他 AOP 框架有所不同，Spring 的 AOP 哲学仍建立在 IoC 之上，这意味着 AOP 也是以 Bean 的方式在 Spring 的 IoC 容器中装配出来的。

### 1.3.3 Spring 对数据访问的封装

Spring DAO 定义了一个访问数据库的一致的接口，对 JDBC 的模板化封装大大简化了 JDBC 代码的编写，并且可解析不同数据库厂商的特定错误代码。使用 Spring DAO 的目的之一就是为了隔离应用程序的业务逻辑和数据访问逻辑，从而获得较高的可扩展性。

对象/关系映射 ORM（Object/Relational Mapping）模块封装了多种 ORM 解决方案。但是，Spring 自身并没有提供任何 ORM 方案，相反，Spring 的 ORM 是为了集成许多流行的 ORM 框架而设计的，例如，Hibernate、iBatis、TopLink、JDO 等。通过 Spring 提供的集成方案，可以非常方便地将其纳入到 Spring 的事务管理中。

### 1.3.4 Spring 的声明式事务

传统的 EJB 开发人员能够从 EJB 中获得的最强大的功能便是声明式事务管理。通过指定事务的属性，开发人员就不必手动编写复杂的事务管理代码，而将事务交给 EJB 容器处理，并且能够获得最高的可移植性。现在，Spring 也提供了完全可媲美 EJB 的声明式事务管理。和 EJB 不同，Spring 的声明式事务管理是建立在轻量级的 AOP 基础之上的，却提供了一致的事务模型。熟悉 EJB 声明式事务管理的开发人员可以立刻转到 Spring 提供的声明式事务管理，这也是整个 Spring 框架最具特色的功能之一。



此为试读，需要完整PDF请访问：www.ertongbook.com