

<<Spring 3.x企业应用开发实战>>

图书基本信息

书名：<<Spring 3.x企业应用开发实战>>

13位ISBN编号：9787121152139

10位ISBN编号：7121152134

出版时间：2012-2

出版时间：电子工业出版社

作者：陈雄华

页数：710

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<Spring 3.x企业应用开发实战>>

内容概要

Spring

3.0是Spring在积蓄了3年之久后，隆重推出的一个重大升级版本，进一步加强了Spring作为Java领域第一开源平台的翘楚地位。

Spring

3.0引入了众多Java开发者翘首以盼的新功能和新特性，如OXM、校验及格式化框架、REST风格的Web编程模型等。

这些新功能实用性强、易用性高，可大幅降低Java应用，特别是Java Web应用开发的难度，同时有效提升应用开发的优雅性。

《Spring3.x企业应用开发实战》是在《精通Spring 2.x——企业应用开发详解》的基础上，经过历时一年的重大调整改版而成的，本书延续了上一版本追求深度，注重原理，不停留在技术表面的写作风格，力求使读者在熟练使用Spring的各项功能的同时，还能透彻理解Spring的内部实现，真正做到知其然知其所以然。此外，本书重点突出了“实战性”的主题，力求使全书“从实际项目中来，到实际项目中去”。

作者简介

陈雄华，2002年毕业于厦门大学计算机与信息工程学院，获硕士学位。

是宝宝淘科技有限公司的创始人之一。

这是一个服务于全国母婴用户的综合性网站，作者负责网站整体框架设计性及核心代码开发的工作。技术开发之余，常将经验所得行诸于文字，作者是天极网、IT168的专栏作者，在各大技术网站、报刊杂志发表过数十篇技术文章，广受读者好评。

于2005年出版《精通JBuilder2005》，本书是2005年最畅销技术图书之一。

林开雄，2006年毕业于集美大学软件工程专业，获取学士学位。

资深软件工程师，精通Spring等优秀开源技术在企业的应用开发，主要研究方向包括业务基础平台、BPM、智能报表、分布式等技术。

书籍目录

第1篇 概述

第1章 Spring概述

- 1.1 认识Spring
- 1.2 关于SpringSource
- 1.3 Spring带给我们什么
- 1.4 Spring体系结构
- 1.5 Spring 3.0的新功能
 - 1.5.1 核心API更新到Java 5.
 - 1.5.2 Spring表达式语言
 - 1.5.3 可通过Java类提供IoC配置信息
 - 1.5.4 通用类型转换系统和属性格式化系统
 - 1.5.5 数据访问层新增OXM功能
 - 1.5.6 Web层的增强
 - 1.5.7 其他
- 1.6 Spring对Java版本的要求
- 1.7 如何获取Spring
- 1.8 小结

第2章 快速入门

- 2.1 实例功能概述
 - 2.1.1 比Hello World更适用的实例
 - 2.1.2 实例功能简介
- 2.2 环境准备
 - 2.2.1 创建库表
 - 2.2.2 建立工程
 - 2.2.3 类包及Spring配置文件规划
- 2.3 持久层
 - 2.3.1 建立领域对象
 - 2.3.2 UserDao
 - 2.3.3 LoginLogDao
 - 2.3.4 在Spring中装配DAO
- 2.4 业务层
 - 2.4.1 UserService
 - 2.4.2 在Spring中装配Service
 - 2.4.3 单元测试
- 2.5 展现层
 - 2.5.1 配置Spring MVC框架
 - 2.5.2 处理登录请求
 - 2.5.3 JSP视图页面
- 2.6 运行Web应用
- 2.7 小结

第2篇 IoC和AOP

第3章 IoC容器概述

- 3.1 IoC概述
 - 3.1.1 通过实例理解IoC的概念
 - 3.1.2 IoC的类型

<<Spring 3.x企业应用开发实战>>

- 3.1.3 通过容器完成依赖关系的注入
- 3.2 相关Java基础知识
 - 3.2.1 简单实例
 - 3.2.2 类装载机ClassLoader
 - 3.2.3 Java反射机制
- 3.3 资源访问利器
 - 3.3.1 资源抽象接口
 - 3.3.2 资源加载
- 3.4 BeanFactory和ApplicationContext
 - 3.4.1 BeanFactory介绍
 - 3.4.2 ApplicationContext介绍
 - 3.4.3 父子容器
- 3.5 Bean的生命周期
 - 3.5.1 BeanFactory中Bean的生命周期
 - 3.5.2 ApplicationContext中Bean的生命周期
- 3.6 小结
- 第4章 在IoC容器中装配Bean
 - 4.1 Spring配置概述
 - 4.1.1 Spring容器高层视图
 - 4.1.2 基于XML的配置
 - 4.2 Bean基本配置
 - 4.2.1 装配一个Bean
 - 4.2.2 Bean的命名
 - 4.3 依赖注入
 - 4.3.1 属性注入
 - 4.3.2 构造函数注入
 - 4.3.3 工厂方法注入
 - 4.3.4 选择注入方式的考量
 - 4.4 注入参数详解
 - 4.4.1 字面值
 - 4.4.2 引用其他Bean
 - 4.4.3 内部Bean
 - 4.4.4 null值
 - 4.4.5 级联属性
 - 4.4.6 集合类型属性
 - 4.4.7 简化配置方式
 - 4.4.8 自动装配
 - 4.5 方法注入
 - 4.5.1 lookup方法注入
 - 4.5.2 方法替换
 - 4.6 < bean > 之间的关系
 - 4.6.1 继承
 - 4.6.2 依赖
 - 4.6.3 引用
 - 4.7 整合多个配置文件
 - 4.8 Bean作用域
 - 4.8.1 singleton作用域

<<Spring 3.x企业应用开发实战>>

- 4.8.2 prototype作用域
- 4.8.3 Web应用环境相关的Bean作用域
- 4.8.4 作用域依赖问题
- 4.9 FactoryBean
- 4.10 基于注解的配置
 - 4.10.1 使用注解定义Bean
 - 4.10.2 使用注解配置信息启动Spring容器
 - 4.10.3 自动装配Bean
 - 4.10.4 Bean作用范围及生命过程方法
- 4.11 基于Java类的配置
 - 4.11.1 使用Java类提供Bean定义信息
 - 4.11.2 使用基于Java类的配置信息启动Spring容器
- 4.12 不同配置方式比较
- 4.13 小结
- 第5章 Spring容器高级主题
 - 5.1 Spring容器技术内幕
 - 5.1.1 内部工作机制
 - 5.1.2 BeanDefinition
 - 5.1.3 InstantiationStrategy
 - 5.1.4 BeanWrapper
 - 5.2 属性编辑器
 - 5.2.1 JavaBean的编辑器
 - 5.2.2 Spring默认属性编辑器
 - 5.2.3 自定义属性编辑器
 - 5.3 使用外部属性文件
 - 5.3.1 使用外部属性文件
 - 5.3.2 使用加密的属性文件
 - 5.3.3 属性文件自身的引用
 - 5.4 引用Bean的属性值
 - 5.5 国际化信息
 - 5.5.1 基础知识
 - 5.5.2 MessageSource
 - 5.5.3 容器级的国际化信息资源
 - 5.6 容器事件
 - 5.6.1 Spring事件类结构
 - 5.6.2 解构Spring事件体系的具体实现
 - 5.6.3 一个实例
 - 5.7 小结
- 第6章 Spring AOP基础
 - 6.1 AOP概述
 - 6.1.1 AOP到底是什么
 - 6.1.2 AOP术语
 - 6.1.3 AOP的实现者
 - 6.2 基础知识
 - 6.2.1 带有横切逻辑的实例
 - 6.2.2 JDK动态代理
 - 6.2.3 CGLib动态代理

- 6.2.4 AOP联盟
- 6.2.5 代理知识小结
- 6.3 创建增强类
 - 6.3.1 增强类型
 - 6.3.2 前置增强
 - 6.3.3 后置增强
 - 6.3.4 环绕增强
 - 6.3.5 异常抛出增强
 - 6.3.6 引介增强
- 6.4 创建切面
 - 6.4.1 切点类型
 - 6.4.2 切面类型
 - 6.4.3 静态普通方法名匹配切面
 - 6.4.4 静态正则表达式方法匹配切面
 - 6.4.5 动态切面
 - 6.4.6 流程切面
 - 6.4.7 复合切点切面
 - 6.4.8 引介切面
- 6.5 自动创建代理
 - 6.5.1 实现类介绍
 - 6.5.2 BeanNameAutoProxyCreator
 - 6.5.3 DefaultAdvisorAutoProxyCreator
- 6.6 小结
- 第7章 基于@AspectJ和Schema的AOP
 - 7.1 Spring对AOP的支持
 - 7.2 JDK 5.0注解知识快速进阶
 - 7.2.1 了解注解
 - 7.2.2 一个简单的注解类
 - 7.2.3 使用注解
 - 7.2.4 访问注解
 - 7.3 着手使用@AspectJ
 - 7.3.1 使用前的准备
 - 7.3.2 一个简单的例子
 - 7.3.3 如何通过配置使用@AspectJ切面
 - 7.4 @AspectJ语法基础
 - 7.4.1 切点表达式函数
 - 7.4.2 在函数入参中使用通配符
 - 7.4.3 逻辑运算符
 - 7.4.4 不同增强类型
 - 7.4.5 引介增强用法
 - 7.5 切点函数详解
 - 7.5.1 @annotation ()
 - 7.5.2 execution ()
 - 7.5.3 args () 和@args ()
 - 7.5.4 within ()
 - 7.5.5 @within () 和@target ()
 - 7.5.6 target () 的this ()

7.6 @AspectJ进阶

7.6.1 切点复合运算

7.6.2 命名切点

7.6.3 增强织入的顺序

7.6.4 访问连接点信息

7.6.5 绑定连接点方法入参

7.6.6 绑定代理对象

7.6.7 绑定类注解对象

7.6.8 绑定返回值

7.6.9 绑定抛出的异常

7.7 基于Schema配置切面

7.7.1 一个简单切面的配置

7.7.2 配置命名切点

7.7.3 各种增强类型的配置

7.7.4 绑定连接点信息

7.7.5 Advisor配置

7.8 混合切面类型

7.8.1 混合使用各种切面类型

7.8.2 各种切面类型总结

7.9 JVM Class文件字节码转换基础知识

7.9.1 java.lang.instrument包的工作原理

7.9.2 如何向JVM中注册转换器

7.9.3 使用JVM启动参数注册转换器的问题

7.10 使用LTW织入切面

7.10.1 Spring的LoadTimeWeaver

7.10.2 使用LTW织入一个切面

7.10.3 在Tomcat下的配置

7.10.4 在其他Web应用服务器下的配置

7.11 小结

第3篇 数据访问

第8章 Spring对DAO的支持

8.1 Spring的DAO理念

8.2 统一的异常体系

8.2.1 Spring的DAO异常体系

8.2.2 JDBC的异常转换器

8.2.3 其他持久技术的异常转换器

8.3 统一数据访问模板

8.3.1 使用模板和回调机制

8.3.2 Spring为不同持久化技术所提供的模板类

8.4 数据源

8.4.1 配置一个数据源

8.4.2 获取JNDI数据源

8.4.3 Spring的数据源实现类

8.5 小结

第9章 Spring的事务管理

9.1 数据库事务基础知识

9.1.1 何为数据库事务

<<Spring 3.x企业应用开发实战>>

- 9.1.2 数据并发的问题
- 9.1.3 数据库锁机制
- 9.1.4 事务隔离级别
- 9.1.5 JDBC对事务支持
- 9.2 ThreadLocal基础知识
 - 9.2.1 ThreadLocal是什么
 - 9.2.2 ThreadLocal的接口方法
 - 9.2.3 一个ThreadLocal实例
 - 9.2.4 与Thread同步机制的比较
 - 9.2.5 Spring使用ThreadLocal解决线程安全问题
- 9.3 Spring对事务管理的支持
 - 9.3.1 事务管理关键抽象
 - 9.3.2 Spring的事务管理器实现类
 - 9.3.3 事务同步管理器
 - 9.3.4 事务传播行为
- 9.4 编程式的事务管理
- 9.5 使用XML配置声明式事务
 - 9.5.1 一个将被实施事务增强的服务接口
 - 9.5.2 使用原始的TransactionProxyFactoryBean
 - 9.5.3 基于tx/aop命名空间的配置
- 9.6 使用注解配置声明式事务
 - 9.6.1 使用@Transactional注解
 - 9.6.2 通过AspectJ LTW引入事务切面
- 9.7 集成特定的应用服务器
 - 9.7.1 BEA WebLogic
 - 9.7.2 BEA WebLogic
- 9.8 小结
- 第10章 Spring的事务管理难点剖析
 - 10.1 DAO和事务管理的牵绊
 - 10.1.1 JDBC访问数据库
 - 10.1.2 Hibernate访问数据库
 - 10.2 应用分层的迷惑
 - 10.3 事务方法嵌套调用的迷茫
 - 10.3.1 Spring事务传播机制回顾
 - 10.3.2 相互嵌套的服务方法
 - 10.4 多线程的困惑
 - 10.4.1 Spring通过单实例化Bean简化多线程问题
 - 10.4.2 启动独立线程调用事务方法
 - 10.5 联合军种作战的混乱
 - 10.5.1 Spring事务管理器的应对
 - 10.5.2 Hibernate+Spring JDBC混合框架的事务管理
 - 10.6 特殊方法成漏网之鱼
 - 10.6.1 哪些方法不能实施Spring AOP事务
 - 10.6.2 事务增强遗漏实例
 - 10.7 数据连接泄漏
 - 10.7.1 底层连接资源的访问问题

<<Spring 3.x企业应用开发实战>>

- 10.7.2 Spring JDBC数据连接泄漏
- 10.7.3 通过DataSourceUtils获取数据连接
- 10.7.4 通过DataSourceUtils获取数据连接
- 10.7.5 JdbcTemplate如何做到对连接泄漏的免疫
- 10.7.6 使用TransactionAwareDataSourceProxy
- 10.7.7 其他数据访问技术的等价类
- 10.8 小结
- 第11章 使用Spring JDBC访问数据库
 - 11.1 使用Spring JDBC
 - 11.1.1 JdbcTemplate小试牛刀
 - 11.1.2 在DAO中使用JdbcTemplate
 - 11.2 基本的数据操作
 - 11.2.1 更改数据
 - 11.2.2 返回数据库的表自增主键值
 - 11.2.3 批量更改数据
 - 11.2.4 查询数据
 - 11.2.5 查询单值数据
 - 11.2.6 调用存储过程
 - 11.3 BLOB/CLOB类型数据的操作
 - 11.3.1 如何获取本地数据连接
 - 11.3.2 相关的操作接口
 - 11.3.3 插入Lob类型的数据
 - 11.3.4 以块数据方式读取Lob数据
 - 11.3.5 以流数据方式读取Lob数据
 - 11.4 自增键和行集
 - 11.4.1 自增键的使用
 - 11.4.2 如何规划主键方案
 - 11.4.3 以行集返回数据
 - 11.5 其他类型的JdbcTemplate
 - 11.5.1 NamedParameterJdbcTemplate
 - 11.5.2 SimpleJdbcTemplate
 - 11.6 以OO方式访问数据库
 - 11.6.1 使用MappingSqlQuery查询数据
 - 11.6.2 使用SqlUpdate更新数据
 - 11.6.3 使用StoredProcedure执行存储过程
 - 11.6.4 SqlFunction类
 - 11.7 小结
- 第12章 整合其他ORM框架
 - 12.1 Spring整合ORM技术
 - 12.2 在Spring中使用Hibernate
 - 12.2.1 配置SessionFactory
 - 12.2.2 使用HibernateTemplate
 - 12.2.3 处理LOB类型数据
 - 12.2.4 添加Hibernate事件监听器
 - 12.2.5 使用原生Hibernate API
 - 12.2.6 使用注解配置
 - 12.2.7 事务处理

- 12.2.8 延迟加载的问题
- 12.3 在Spring中使用myBatis
 - 12.3.1 配置SqlMapClient
 - 12.3.2 在Spring配置myBatis
 - 12.3.3 编写myBatis的DAO
- 12.5 DAO层设计
 - 12.5.1 DAO基类的设计
 - 12.5.2 查询接口方法的设计
 - 12.5.3 分页查询接口设计
- 12.6 小结
- 第4篇 业务层及Web层技术
- 第13章 任务调度和异步执行器
 - 13.1 任务调度概述
 - 13.2 Quartz快速进阶
 - 13.2.1 Quartz基础结构
 - 13.2.2 使用SimpleTrigger
 - 13.2.3 使用CronTrigger
 - 13.2.4 使用Calendar
 - 13.2.5 任务调度信息存储
 - 13.3 在Spring中使用Quartz
 - 13.3.1 创建JobDetail
 - 13.3.2 创建Trigger
 - 13.3.3 创建Scheduler
 - 13.4 Spring中使用JDK Timer
 - 13.4.1 Timer和TimerTask
 - 13.4.2 Spring对JDK Timer的支持
 - 13.5 Spring对JDK 5.0 Executor的支持
 - 13.5.1 了解JDK 5.0的Executor
 - 13.5.2 Spring对Executor所提供的抽象
 - 13.6 实际应用中的任务调度
 - 13.6.1 如何产生任务
 - 13.6.2 任务调度对应用程序集群的影响
 - 13.6.3 任务调度云
 - 13.6.4 Web应用程序中调度器的启动和关闭问题
 - 13.7 小结
- 第14章 使用OXM进行对象XML映射
 - 14.1 认识XML解析技术
 - 14.1.1 什么是XML
 - 14.1.2 XML的处理技术
 - 14.2 XML处理利器：XStream
 - 14.2.1 XStream概述
 - 14.2.2 快速入门
 - 14.2.3 使用XStream别名
 - 14.2.4 XStream转换器
 - 14.2.5 XStream注解
 - 14.2.6 流化对象
 - 14.2.7 持久化API

<<Spring 3.x企业应用开发实战>>

- 14.2.8 额外功能：处理JSON
- 14.3 其他常见O/X Mapping开源项目
 - 14.3.1 JAXB
 - 14.3.2 XMLBeans
 - 14.3.3 Castor
 - 14.3.4 JiBX
 - 14.3.5 总结比较
- 14.4 与Spring OXM整合
 - 14.4.1 Spring OXM概述
 - 14.4.2 整合OXM实现者
 - 14.4.3 如何在Spring中进行配置
 - 14.4.4 Spring OXM 简单实例
- 14.5 小结
- 第15章 Spring MVC
 - 15.1 Spring MVC概述
 - 15.1.1 体系结构
 - 15.1.2 配置DispatcherServlet
 - 15.1.3 一个简单的实例
 - 15.2 注解驱动的控制器的
 - 15.2.1 使用@RequestMapping映射请求
 - 15.2.2 请求处理方法签名概述
 - 15.2.3 请求处理方法签名详细说明
 - 15.2.4 使用HttpMessageConverter < T >
 - 15.2.5 处理模型数据
 - 15.3 处理方法的数据绑定
 - 15.3.1 数据绑定流程剖析
 - 15.3.2 数据转换
 - 15.3.3 数据格式化
 - 15.3.4 数据校验
 - 15.4 视图和视图解析器
 - 15.4.1 认识视图
 - 15.4.2 认识视图解析器
 - 15.4.3 JSP和JSTL
 - 15.4.4 模板视图
 - 15.4.5 Excel
 - 15.4.6 PDF
 - 15.4.7 输出XML
 - 15.4.8 输出JSON
 - 15.4.9 使用XmlViewResolver
 - 15.4.10 使用ResourceBundle ViewResolver
 - 15.4.11 混合使用多种视图技术
 - 15.5 本地化解析
 - 15.5.1 本地化概述
 - 15.5.2 使用CookieLocaleResolver
 - 15.5.3 使用SessionLocaleResolver
 - 15.5.4 使用LocaleChangeInterceptor
 - 15.6 文件上传

<<Spring 3.x企业应用开发实战>>

- 15.6.1 配置MultipartResolver
- 15.6.2 编写控制器和文件上传表单页面
- 15.7 杂项
 - 15.7.1 静态资源处理
 - 15.7.2 装配拦截器
 - 15.7.3 异常处理
- 15.8 小结
- 第5篇 测试及实战
- 第16章 实战型单元测试
 - 16.1 单元测试概述
 - 16.1.1 为什么需要单元测试
 - 16.1.2 单元测试之误解
 - 16.1.3 单元测试之困境
 - 16.1.4 单元测试基本概念
 - 16.2 JUnit 4快速进阶
 - 16.2.1 JUnit 4概述
 - 16.2.2 JUnit 4生命周期
 - 16.2.3 使用JUnit
 - 16.3 模拟利器Mockito
 - 16.3.1 模拟测试概述
 - 16.3.2 创建Mock对象
 - 16.3.3 设定Mock对象的期望行为及返回值
 - 16.3.4 验证交互行为
 - 16.4 测试整合之王Unitils
 - 16.4.1 Unitils概述
 - 16.4.2 集成Spring
 - 16.4.3 集成Hibernate
 - 16.4.4 集成Dbunit
 - 16.4.5 自定义扩展模块
 - 16.5 使用Unitils测试DAO层
 - 16.5.1 数据库测试的难点
 - 16.5.2 扩展Dbunit用Excel准备数据
 - 16.5.3 测试实战
 - 16.6 使用unitils测试Service层
 - 16.7 测试Web层
 - 16.7.1 对LoginController进行单元测试
 - 16.7.2 使用Spring Servlet API模拟对象
 - 16.7.3 使用Spring RestTemplate测试
 - 16.7.4 使用Selenium测试
 - 16.8 小结
- 第17章 实战案例开发
 - 17.1 论坛案例概述
 - 17.1.1 论坛整体功能结构
 - 17.1.2 论坛用例描述
 - 17.1.3 主要功能流程描述
 - 17.2 系统设计
 - 17.2.1 技术框架选择

<<Spring 3.x企业应用开发实战>>

- 17.2.2 Web目录结构及类包结构规划
 - 17.2.3 单元测试类包结构规划
 - 17.2.4 系统的结构图
 - 17.2.5 PO的类设计
 - 17.2.6 持久层设计
 - 17.2.7 服务层设计
 - 17.2.8 Web层设计
 - 17.2.9 数据库设计
 - 17.3 开发前的准备
 - 17.4 持久层开发
 - 17.4.1 PO类
 - 17.4.2 DAO基类
 - 17.4.3 通过扩展基类所定义DAO类
 - 17.4.4 DAO Bean的装配
 - 17.4.5 使用Hibernate二级缓存
 - 17.5 对持久层进行测试
 - 17.5.1 配置Unitils测试环境
 - 17.5.2 准备测试数据库及测试数据
 - 17.5.3 编写DAO测试基类
 - 17.5.4 编写BoardDao测试用例
 - 17.6 服务层开发
 - 17.6.1 UserService的开发
 - 17.6.2 ForumService的开发
 - 17.6.3 服务类Bean的装配
 - 17.7 对服务层进行测试
 - 17.7.1 编写Service测试基类
 - 17.7.2 编写ForumService测试用例
 - 17.8 Web层开发
 - 17.8.1 BaseController的基类
 - 17.8.2 用户登录和注销
 - 17.8.3 用户注册
 - 17.8.4 论坛管理
 - 17.8.5 论坛普通功能
 - 17.8.6 分页显示论坛板块的主题帖子
 - 17.8.7 web.xml配置
 - 17.8.8 Spring MVC配置
 - 17.9 对Web层进行测试
 - 17.9.1 编写Web测试基类
 - 17.9.2 编写ForumManageController测试用例
 - 17.10 部署和运行应用
 - 17.11 小结
- 以下内容详见本书配书光盘：
- 附录A JavaMail发送邮件
 - 附录B 在Spring中开发Web Service

章节摘录

版权页：插图：12.1 Spring整合ORM技术 Spring的开放性和扩展性在持久化技术领域得到了充分的证明，Spring不但直接提供了SpringJDBC，还整合了JPA、Hibernate、iBatis、JDO等ORM领域的代表者。

使用者完全可以根据需要作出适合自己的选择。

在第11章中，我们详细讲解了Spring对JDBC的API的封装，其中大部分的思路和整合其他ORM框架是一脉相通的：在资源管理、DAO模板类、事务管理、DAO异常体系等方面，Spring始终保持整合方式上的统一。

在Spring中掌握了一种持久化技术后，切换到另一种持久化技术并不需要花费什么代价。

Spring在集成ORM框架时，始终走民主亲民的路线。

它既提供了方便的模板类对原ORM进行简化封装，以一种更具Spring的风格使用ORM技术。

同时，在保证继续享受Spring通用功能的情况下，开发者还可能使用ORM框架原生的API编写程序。

这让一直使用ORM原生API编程的开发者感到自然，不会存在过渡上的障碍。

当我们使用某种ORM框架时，为了让其更适合项目的需要，降低使用难度，一般情况下，都会在原有ORMAPI的基础上编写一套封装类。

Spring高屋建瓴地为我们做了类似的工作，所以在决定花费力气并冒着风险去构造类似的底层架构之前，最好事先参考一下Spring的解决方案，而不要急着另起炉灶。

使用Spring所提供的ORM整合方案，我们可以获得许多好处。

方便基础设施的搭建 不同的ORM技术都有一套自己的方案以初始化框架、搭建基础设施等。

在搭建基础设施中，数据源是不可或缺的资源，不同ORM框架的实现方式各不相同：如JPA中通过persistence.xml定义数据源，而Hibernate在hibernate.cfg.xml中配置数据源，目的相同但方法迥异。

在数据源基础上，不同的ORM框架拥有自己的基础实例，如Hibernate的SessionFactory、iBatis

的SqlMapClient，它们是ORM程序运行时的底层设施，在程序级别代表着ORM框架本身。

在学习不同的ORM框架知识时，往往都会有一个独立的篇章专门讲解如何初始化这些实例。

在Spring中对于不同的ORM框架，首先，始终可以采用相同方式配置数据源：其次，Spring为不同的ORM框架提供相应的FactoryBean，用以初始化ORM框架的基础设施，可以将它们当成普通Bean对待，唯一的差别只是属性的不同而已。

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>