

《精通Spring 4.x》

图书基本信息

书名：《精通Spring 4.x》

13位ISBN编号：9787121304430

出版时间：2017-1-1

作者：陈雄华

页数：799

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：www.tushu000.com

内容概要

Spring 4.0是Spring在积蓄4年后，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**开源平台的翘楚地位。Spring 4.0引入了众多Java开发者翘首以盼的基于Groovy Bean的配置、HTML 5/WebSocket支持等新功能，全面支持Java 8.0，*低要求是Java 6.0。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。本书是在《精通Spring 3.x 企业应用开发详解》的基础上，历时一年的重大调整改版而成的，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的各项功能的同时透彻理解Spring的内部实现，真正做到知其然并知其所以然。此外，本书重点突出了“实战性”的主题，力求使全书内容体现“从实际项目中来，到实际项目中去”的写作原则。

作者简介

陈雄华，现担任厦门中软海晟信息技术有限公司技术总监，.已出版作品《精通JBuilder 2005》《精通Spring 2.x》《精通Spring 3.x》。

书籍目录

第1篇 基础篇

第1章 Spring概述\t2

- 1.1 认识Spring\t2
- 1.2 关于SpringSource\t4
- 1.3 Spring带给我们什么\t5
- 1.4 Spring体系结构\t6
- 1.5 Spring对Java版本的要求\t8
- 1.6 Spring 4.0新特性\t8
 - 1.6.1 全面支持Java 8.0\t9
 - 1.6.2 核心容器的增强\t11
 - 1.6.3 支持用Groovy定义Bean\t12
 - 1.6.4 Web的增强\t12
 - 1.6.5 支持WebSocket\t12
 - 1.6.6 测试的增强\t13
 - 1.6.7 其他\t13
- 1.7 Spring子项目\t13
- 1.8 如何获取Spring\t15
- 1.9 小结\t16

第2章 快速入门\t17

- 2.1 实例概述\t17
 - 2.1.1 比Hello World更适用的实例\t18
 - 2.1.2 实例功能简介\t18
- 2.2 环境准备\t20
 - 2.2.1 构建工具Maven\t20
 - 2.2.2 创建库表\t22
 - 2.2.3 建立工程\t23
 - 2.2.4 类包及Spring配置文件规划\t28
- 2.3 持久层\t29
 - 2.3.1 建立领域对象\t29
 - 2.3.2 UserDao\t30
 - 2.3.3 LoginLogDao\t33
 - 2.3.4 在Spring中装配DAO\t34
- 2.4 业务层\t35
 - 2.4.1 UserService\t35
 - 2.4.2 在Spring中装配Service\t37
 - 2.4.3 单元测试\t38
- 2.5 展现层\t40
 - 2.5.1 配置Spring MVC框架\t40
 - 2.5.2 处理登录请求\t42
 - 2.5.3 JSP视图页面\t44
- 2.6 运行Web应用\t46
- 2.7 小结\t48

第3章 Spring Boot\t49

- 3.1 Spring Boot概览\t49
 - 3.1.1 Spring Boot发展背景\t50
 - 3.1.2 Spring Boot特点\t50
 - 3.1.3 Spring Boot启动器\t50

- 3.2 快速入门\t52
- 3.3 安装配置\t54
 - 3.3.1 基于Maven环境配置\t54
 - 3.3.2 基于Gradle环境配置\t56
 - 3.3.3 基于Spring Boot CLI环境配置\t57
 - 3.3.4 代码包结构规划\t58
- 3.4 持久层\t59
 - 3.4.1 初始化配置\t59
 - 3.4.2 UserDao\t61
- 3.5 业务层\t62
- 3.6 展现层\t64
 - 3.6.1 配置pom.xml依赖\t64
 - 3.6.2 配置Spring MVC框架\t65
 - 3.6.3 处理登录请求\t65
- 3.7 运维支持\t67
- 3.8 小结\t70
- 第2篇 核心篇
- 第4章 IoC容器\t72
 - 4.1 IoC概述\t72
 - 4.1.1 通过实例理解IoC的概念\t73
 - 4.1.2 IoC的类型\t75
 - 4.1.3 通过容器完成依赖关系的注入\t77
 - 4.2 相关Java基础知识\t78
 - 4.2.1 简单实例\t78
 - 4.2.2 类装载机ClassLoader\t80
 - 4.2.3 Java反射机制\t83
 - 4.3 资源访问利器\t85
 - 4.3.1 资源抽象接口\t85
 - 4.3.2 资源加载\t88
 - 4.4 BeanFactory和ApplicationContext\t91
 - 4.4.1 BeanFactory介绍\t92
 - 4.4.2 ApplicationContext介绍\t94
 - 4.4.3 父子容器\t103
 - 4.5 Bean的生命周期\t103
 - 4.5.1 BeanFactory中Bean的生命周期\t103
 - 4.5.2 ApplicationContext中Bean的生命周期\t112
 - 4.6 小结\t114
- 第5章 在IoC容器中装配Bean\t115
 - 5.1 Spring配置概述\t116
 - 5.1.1 Spring容器高层视图\t116
 - 5.1.2 基于XML的配置\t117
 - 5.2 Bean基本配置\t120
 - 5.2.1 装配一个Bean\t120
 - 5.2.2 Bean的命名\t120
 - 5.3 依赖注入\t121

- 5.3.1 属性注入\t121
- 5.3.2 构造函数注入\t124
- 5.3.3 工厂方法注入\t128
- 5.3.4 选择注入方式的考量\t130
- 5.4 注入参数详解\t130
 - 5.4.1 字面值\t130
 - 5.4.2 引用其他Bean\t131
 - 5.4.3 内部Bean\t133
 - 5.4.4 null值\t133
 - 5.4.5 级联属性\t134
 - 5.4.6 集合类型属性\t134
 - 5.4.7 简化配置方式\t138
 - 5.4.8 自动装配\t141
- 5.5 方法注入\t142
 - 5.5.1 lookup方法注入\t142
 - 5.5.2 方法替换\t143
- 5.6 <bean> 之间的关系\t144
 - 5.6.1 继承\t144
 - 5.6.2 依赖\t145
 - 5.6.3 引用\t146
- 5.7 整合多个配置文件\t147
- 5.8 Bean作用域\t148
 - 5.8.1 singleton作用域\t148
 - 5.8.2 prototype作用域\t149
 - 5.8.3 与Web应用环境相关的Bean作用域\t150
 - 5.8.4 作用域依赖问题\t152
- 5.9 FactoryBean\t153
- 5.10 基于注解的配置\t155
 - 5.10.1 使用注解定义Bean\t155
 - 5.10.2 扫描注解定义的Bean\t156
 - 5.10.3 自动装配Bean\t157
 - 5.10.4 Bean作用范围及生命过程方法\t162
- 5.11 基于Java类的配置\t164
 - 5.11.1 使用Java类提供Bean定义信息\t164
 - 5.11.2 使用基于Java类的配置信息启动Spring容器\t167
- 5.12 基于Groovy DSL的配置\t169
 - 5.12.1 使用Groovy DSL提供Bean定义信息\t169
 - 5.12.2 使用GenericGroovyApplication Context启动Spring容器\t171
- 5.13 通过编码方式动态添加Bean\t172
 - 5.13.1 通过DefaultListableBean Factory\t172
 - 5.13.2 扩展自定义标签\t173
- 5.14 不同配置方式比较\t175

- 5.15 小结\t177
- 第6章 Spring容器高级主题\t178
 - 6.1 Spring容器技术内幕\t178
 - 6.1.1 内部工作机制\t179
 - 6.1.2 BeanDefinition\t182
 - 6.1.3 InstantiationStrategy\t183
 - 6.1.4 BeanWrapper\t183
 - 6.2 属性编辑器\t184
 - 6.2.1 JavaBean的编辑器\t185
 - 6.2.2 Spring默认属性编辑器\t188
 - 6.2.3 自定义属性编辑器\t189
 - 6.3 使用外部属性文件\t192
 - 6.3.1 PropertyPlaceholderConfigurer
属性文件\t192
 - 6.3.2 使用加密的属性文件\t195
 - 6.3.3 属性文件自身的引用\t198
 - 6.4 引用Bean的属性值\t199
 - 6.5 国际化信息\t201
 - 6.5.1 基础知识\t201
 - 6.5.2 MessageSource\t206
 - 6.5.3 容器级的国际化信息资源\t209
 - 6.6 容器事件\t210
 - 6.6.1 Spring事件类结构\t211
 - 6.6.2 解构Spring事件体系的具体
实现\t213
 - 6.6.3 一个实例\t214
 - 6.7 小结\t215
- 第7章 Spring AOP基础\t216
 - 7.1 AOP概述\t216
 - 7.1.1 AOP到底是什么\t217
 - 7.1.2 AOP术语\t219
 - 7.1.3 AOP的实现者\t221
 - 7.2 基础知识\t222
 - 7.2.1 带有横切逻辑的实例\t222
 - 7.2.2 JDK动态代理\t224
 - 7.2.3 CGLib动态代理\t228
 - 7.2.4 AOP联盟\t229
 - 7.2.5 代理知识小结\t230
 - 7.3 创建增强类\t230
 - 7.3.1 增强类型\t230
 - 7.3.2 前置增强\t231
 - 7.3.3 后置增强\t235
 - 7.3.4 环绕增强\t236
 - 7.3.5 异常抛出增强\t237
 - 7.3.6 引介增强\t239
 - 7.4 创建切面\t243
 - 7.4.1 切点类型\t243
 - 7.4.2 切面类型\t244
 - 7.4.3 静态普通方法名匹配切面\t246

- 7.4.4 静态正则表达式方法匹配切面\t248
- 7.4.5 动态切面\t251
- 7.4.6 流程切面\t254
- 7.4.7 复合切点切面\t256
- 7.4.8 引介切面\t258
- 7.5 自动创建代理\t259
 - 7.5.1 实现类介绍\t259
 - 7.5.2 BeanNameAutoProxyCreator\t260
 - 7.5.3 DefaultAdvisorAutoProxyCreator\t261
 - 7.5.4 AOP无法增强疑难问题剖析\t262
- 7.6 小结\t267
- 第8章 基于@AspectJ和Schema的AOP\t269
 - 8.1 Spring对AOP的支持\t269
 - 8.2 Java 5.0注解知识快速进阶\t270
 - 8.2.1 了解注解\t270
 - 8.2.2 一个简单的注解类\t271
 - 8.2.3 使用注解\t272
 - 8.2.4 访问注解\t273
 - 8.3 着手使用@AspectJ\t274
 - 8.3.1 使用前的准备\t275
 - 8.3.2 一个简单的例子\t275
 - 8.3.3 如何通过配置使用@AspectJ切面\t277
 - 8.4 @AspectJ语法基础\t278
 - 8.4.1 切点表达式函数\t278
 - 8.4.2 在函数入参中使用通配符\t279
 - 8.4.3 逻辑运算符\t280
 - 8.4.4 不同增强类型\t281
 - 8.4.5 引介增强用法\t282
 - 8.5 切点函数详解\t283
 - 8.5.1 @annotation()\t284
 - 8.5.2 execution()\t285
 - 8.5.3 args()和@args()\t287
 - 8.5.4 within()\t288
 - 8.5.5 @within()和@target()\t289
 - 8.5.6 target()和this()\t290
 - 8.6 @AspectJ进阶\t291
 - 8.6.1 切点复合运算\t292
 - 8.6.2 命名切点\t292
 - 8.6.3 增强织入的顺序\t294
 - 8.6.4 访问连接点信息\t294
 - 8.6.5 绑定连接点方法入参\t295
 - 8.6.6 绑定代理对象\t297
 - 8.6.7 绑定类注解对象\t298
 - 8.6.8 绑定返回值\t299

- 8.6.9 绑定抛出的异常\t299
- 8.7 基于Schema配置切面\t300
 - 8.7.1 一个简单切面的配置\t300
 - 8.7.2 配置命名切点\t302
 - 8.7.3 各种增强类型的配置\t303
 - 8.7.4 绑定连接点信息\t305
 - 8.7.5 Advisor配置\t306
- 8.8 混合切面类型\t307
 - 8.8.1 混合使用各种切面类型\t308
 - 8.8.2 各种切面类型总结\t308
- 8.9 其他\t309
 - 8.9.1 JVM Class文件字节码转换基础知识\t309
 - 8.9.2 使用LTW织入切面\t311
- 8.10 小结\t314
- 第9章 Spring SpEL\t316
 - 9.1 JVM动态语言\t316
 - 9.2 SpEL表达式概述\t318
 - 9.3 SpEL核心接口\t319
 - 9.3.1 EvaluationContext接口\t320
 - 9.3.2 SpEL编译器\t321
 - 9.4 SpEL基础表达式\t323
 - 9.4.1 文本字符解析\t323
 - 9.4.2 对象属性解析\t323
 - 9.4.3 数组、集合类型解析\t324
 - 9.4.4 方法解析\t326
 - 9.4.5 操作符解析\t327
 - 9.4.6 安全导航操作符\t329
 - 9.4.7 三元操作符\t330
 - 9.4.8 Elvis操作符\t331
 - 9.4.9 赋值、类型、构造器、变量\t332
 - 9.4.10 集合过滤\t335
 - 9.4.11 集合转换\t335
 - 9.5 在Spring中使用SpEL\t336
 - 9.5.1 基于XML的配置\t336
 - 9.5.2 基于注解的配置\t337
 - 9.6 小结\t338
- 第3篇 数据篇
- 第10章 Spring对DAO的支持\t340
 - 10.1 Spring的DAO理念\t340
 - 10.2 统一的异常体系\t341
 - 10.2.1 Spring的DAO异常体系\t341
 - 10.2.2 JDBC的异常转换器\t343
 - 10.2.3 其他持久化技术的异常转换器\t344
 - 10.3 统一数据访问模板\t344
 - 10.3.1 使用模板和回调机制\t345
 - 10.3.2 Spring为不同持久化技术所提供的模板类\t347

- 10.4 数据源\t348
 - 10.4.1 配置一个数据源\t348
 - 10.4.2 获取JNDI数据源\t352
 - 10.4.3 Spring的数据源实现类\t353
 - 10.5 小结\t353
- 第11章 Spring的事务管理\t355
 - 11.1 数据库事务基础知识\t355
 - 11.1.1 何为数据库事务\t356
 - 11.1.2 数据并发的问题\t357
 - 11.1.3 数据库锁机制\t359
 - 11.1.4 事务隔离级别\t360
 - 11.1.5 JDBC对事务的支持\t361
 - 11.2 ThreadLocal基础知识\t362
 - 11.2.1 ThreadLocal是什么\t363
 - 11.2.2 ThreadLocal的接口方法\t363
 - 11.2.3 一个ThreadLocal实例\t364
 - 11.2.4 与Thread同步机制的比较\t366
 - 11.2.5 Spring使用ThreadLocal解决线程安全问题\t366
 - 11.3 Spring对事务管理的支持\t368
 - 11.3.1 事务管理关键抽象\t369
 - 11.3.2 Spring的事务管理器实现类\t371
 - 11.3.3 事务同步管理器\t374
 - 11.3.4 事务传播行为\t375
 - 11.4 编程式的事务管理\t376
 - 11.5 使用XML配置声明式事务\t377
 - 11.5.1 一个将被实施事务增强的服务接口\t379
 - 11.5.2 使用原始的TransactionProxyFactoryBean\t379
 - 11.5.3 基于aop/tx命名空间的配置\t382
 - 11.6 使用注解配置声明式事务\t385
 - 11.6.1 使用@Transactional注解\t385
 - 11.6.2 通过AspectJ LTW引入事务切面\t389
 - 11.7 集成特定的应用服务器\t390
 - 11.7.1 BEA WebLogic\t390
 - 11.7.2 WebSphere\t390
 - 11.8 小结\t390
- 第12章 Spring的事务管理难点剖析\t392
 - 12.1 DAO和事务管理的牵绊\t393
 - 12.1.1 JDBC访问数据库\t393
 - 12.1.2 Hibernate访问数据库\t395
 - 12.2 应用分层的迷惑\t398
 - 12.3 事务方法嵌套调用的迷茫\t401
 - 12.3.1 Spring事务传播机制回顾\t401
 - 12.3.2 相互嵌套的服务方法\t402
 - 12.4 多线程的困惑\t405
 - 12.4.1 Spring通过单实例化Bean

- 简化多线程问题\t405
- 12.4.2 启动独立线程调用事务方法\t405
- 12.5 联合军种作战的混乱\t408
- 12.5.1 Spring事务管理器的应对\t408
- 12.5.2 Hibernate+Spring JDBC混合框架的事务管理\t408
- 12.6 特殊方法成漏网之鱼\t412
- 12.6.1 哪些方法不能实施Spring AOP事务\t412
- 12.6.2 事务增强遗漏实例\t413
- 12.7 数据连接泄露\t416
- 12.7.1 底层连接资源的访问问题\t416
- 12.7.2 Spring JDBC数据连接泄露\t417
- 12.7.3 事务环境下通过DataSource Utils获取数据连接\t420
- 12.7.4 无事务环境下通过DataSource Utils获取数据连接\t422
- 12.7.5 JdbcTemplate如何做到对连接泄露的免疫\t424
- 12.7.6 使用TransactionAwareDataSourceProxy\t425
- 12.7.7 其他数据访问技术的等价类\t426
- 12.8 小结\t426
- 第13章 使用Spring JDBC访问数据库\t428
- 13.1 使用Spring JDBC\t428
- 13.1.1 JdbcTemplate小试牛刀\t429
- 13.1.2 在DAO中使用JdbcTemplate\t429
- 13.2 基本的数据操作\t431
- 13.2.1 更改数据\t431
- 13.2.2 返回数据库的表自增主键值\t434
- 13.2.3 批量更改数据\t436
- 13.2.4 查询数据\t437
- 13.2.5 查询单值数据\t440
- 13.2.6 调用存储过程\t442
- 13.3 BLOB/CLOB类型数据的操作\t444
- 13.3.1 如何获取本地数据连接\t445
- 13.3.2 相关的操作接口\t446
- 13.3.3 插入LOB类型的数据\t448
- 13.3.4 以块数据方式读取LOB数据\t450
- 13.3.5 以流数据方式读取LOB数据\t451
- 13.4 自增键和行集\t452
- 13.4.1 自增键的使用\t452
- 13.4.2 如何规划主键方案\t454
- 13.4.3 以行集返回数据\t456

13.5 NamedParameterJdbcTemplate

模板类\t456

13.6 小结\t459

第14章 整合其他ORM框架\t460

14.1 Spring整合ORM技术\t460

14.2 在Spring中使用Hibernate\t462

14.2.1 配置SessionFactory\t462

14.2.2 使用HibernateTemplate\t465

14.2.3 处理LOB类型的数据\t469

14.2.4 添加Hibernate事件监听器\t470

14.2.5 使用原生的Hibernate API\t471

14.2.6 使用注解配置\t472

14.2.7 事务处理\t474

14.2.8 延迟加载问题\t475

14.3 在Spring中使用MyBatis\t476

14.3.1 配置SqlMapClient\t476

14.3.2 在Spring中配置MyBatis\t478

14.3.3 编写MyBatis的DAO\t479

14.4 DAO层设计\t482

14.4.1 DAO基类设计\t482

14.4.2 查询接口方法设计\t484

14.4.3 分页查询接口设计\t486

14.5 小结\t487

第4篇 应用篇

第15章 Spring Cache\t490

15.1 缓存概述\t490

15.1.1 缓存的概念\t490

15.1.2 使用Spring Cache\t493

15.2 掌握Spring Cache抽象\t498

15.2.1 缓存注解\t498

15.2.2 缓存管理器\t504

15.2.3 使用SpEL表达式\t506

15.2.4 基于XML的Cache声明\t506

15.2.5 以编程方式初始化缓存\t507

15.2.6 自定义缓存注解\t509

15.3 配置Cache存储\t509

15.3.1 EhCache\t510

15.3.2 Guava\t510

15.3.3 HazelCast\t511

15.3.4 GemFire\t511

15.3.5 JSR-107 Cache\t511

15.4 实战经验\t513

15.5 小结\t514

第16章 任务调度和异步执行器\t516

16.1 任务调度概述\t516

16.2 Quartz快速进阶\t517

16.2.1 Quartz基础结构\t518

16.2.2 使用SimpleTrigger\t520

16.2.3 使用CronTrigger\t522

- 16.2.4 使用Calendar\t526
- 16.2.5 任务调度信息存储\t527
- 16.3 在Spring中使用Quartz\t530
 - 16.3.1 创建JobDetail\t530
 - 16.3.2 创建Trigger\t533
 - 16.3.3 创建Scheduler\t534
- 16.4 在Spring中使用JDK Timer\t536
 - 16.4.1 Timer和TimerTask\t536
 - 16.4.2 Spring对Java Timer的支持\t539
- 16.5 Spring对Java 5.0 Executor的支持\t540
 - 16.5.1 了解Java 5.0的Executor\t541
 - 16.5.2 Spring对Executor所提供的抽象\t543
- 16.6 实际应用中的任务调度\t544
 - 16.6.1 如何产生任务\t545
 - 16.6.2 任务调度对应用程序集群的影响\t547
 - 16.6.3 任务调度云\t547
 - 16.6.4 Web应用程序中调度器的启动和关闭问题\t549
- 16.7 小结\t552
- 第17章 Spring MVC\t553
 - 17.1 Spring MVC体系概述\t554
 - 17.1.1 体系结构\t554
 - 17.1.2 配置DispatcherServlet\t555
 - 17.1.3 一个简单的实例\t560
 - 17.2 注解驱动的控制器的\t565
 - 17.2.1 使用@RequestMapping映射请求\t565
 - 17.2.2 请求处理方法签名\t569
 - 17.2.3 使用矩阵变量绑定参数\t570
 - 17.2.4 请求处理方法签名详细说明\t571
 - 17.2.5 使用HttpMessageConverter < T > \t575
 - 17.2.6 使用@RestController和AsyncRestTemplate\t584
 - 17.2.7 处理模型数据\t586
 - 17.3 处理方法的数据绑定\t591
 - 17.3.1 数据绑定流程剖析\t592
 - 17.3.2 数据转换\t592
 - 17.3.3 数据格式化\t598
 - 17.3.4 数据校验\t602
 - 17.4 视图和视图解析器\t611
 - 17.4.1 认识视图\t611
 - 17.4.2 认识视图解析器\t612
 - 17.4.3 JSP和JSTL\t613
 - 17.4.4 模板视图\t618
 - 17.4.5 Excel\t621
 - 17.4.6 PDF\t623

- 17.4.7 输出XML\t625
- 17.4.8 输出JSON\t626
- 17.4.9 使用XmlViewResolver\t626
- 17.4.10 使用ResourceBundleView
Resolver\t627
- 17.4.11 混合使用多种视图技术\t628
- 17.5 本地化解析\t630
 - 17.5.1 本地化的概念\t630
 - 17.5.2 使用CookieLocaleResolver\t631
 - 17.5.3 使用SessionLocaleResolver\t632
 - 17.5.4 使用LocaleChange
Interceptor\t632
- 17.6 文件上传\t633
 - 17.6.1 配置MultipartResolver\t633
 - 17.6.2 编写控制器和文件上传表单
页面\t633
- 17.7 WebSocket支持\t634
 - 17.7.1 使用WebSocket\t634
 - 17.7.2 WebSocket的限制\t638
- 17.8 杂项\t639
 - 17.8.1 静态资源处理\t639
 - 17.8.2 装配拦截器\t643
 - 17.8.3 异常处理\t644
 - 17.8.4 RequestContextHolder的
使用\t646
- 17.9 小结\t646
- 第18章 实战案例开发\t648
 - 18.1 论坛案例概述\t648
 - 18.1.1 论坛整体功能结构\t648
 - 18.1.2 论坛用例描述\t649
 - 18.1.3 主要功能流程描述\t651
 - 18.2 系统设计\t655
 - 18.2.1 技术框架选择\t655
 - 18.2.2 采用Maven构建项目\t656
 - 18.2.3 单元测试类包结构规划\t657
 - 18.2.4 系统架构图\t658
 - 18.2.5 PO类设计\t658
 - 18.2.6 持久层设计\t659
 - 18.2.7 服务层设计\t660
 - 18.2.8 Web层设计\t661
 - 18.2.9 数据库设计\t662
 - 18.3 开发前的准备\t663
 - 18.4 持久层开发\t664
 - 18.4.1 PO类\t664
 - 18.4.2 DAO基类\t666
 - 18.4.3 通过扩展基类定义DAO类\t670
 - 18.4.4 DAO Bean的装配\t672
 - 18.4.5 使用Hibernate二级缓存\t673
 - 18.5 对持久层进行测试\t675

- 18.5.1 配置Unitils测试环境\t675
 - 18.5.2 准备测试数据库及测试数据\t676
 - 18.5.3 编写DAO测试基类\t677
 - 18.5.4 编写BoardDao测试用例\t678
 - 18.6 服务层开发\t680
 - 18.6.1 UserService的开发\t680
 - 18.6.2 ForumService的开发\t681
 - 18.6.3 服务类Bean的装配\t683
 - 18.7 对服务层进行测试\t684
 - 18.7.1 编写Service测试基类\t685
 - 18.7.2 编写ForumService测试用例\t685
 - 18.8 Web层开发\t687
 - 18.8.1 BaseController的基类\t687
 - 18.8.2 用户登录和注销\t689
 - 18.8.3 用户注册\t691
 - 18.8.4 论坛管理\t692
 - 18.8.5 论坛普通功能\t694
 - 18.8.6 分页显示论坛版块的主题帖子\t696
 - 18.8.7 web.xml配置\t700
 - 18.8.8 Spring MVC配置\t702
 - 18.9 对Web层进行测试\t703
 - 18.9.1 编写Web测试基类\t703
 - 18.9.2 编写ForumManageController测试用例\t704
 - 18.10 开发环境部署\t705
 - 18.11 项目配置实战经验\t708
 - 18.11.1 “传统的”Web项目属性文件\t708
 - 18.11.2 如何规划便于部署的Web项目属性文件\t709
 - 18.11.3 数据源的配置\t710
 - 18.12 小结\t712
- 第5篇 提高篇
- 第19章 Spring OXM\t714
- 19.1 认识XML解析技术\t714
 - 19.1.1 什么是XML\t714
 - 19.1.2 XML的处理技术\t715
 - 19.2 XML处理利器：XStream\t717
 - 19.2.1 XStream概述\t717
 - 19.2.2 快速入门\t718
 - 19.2.3 使用XStream别名\t720
 - 19.2.4 XStream转换器\t721
 - 19.2.5 XStream注解\t723
 - 19.2.6 流化对象\t725
 - 19.2.7 持久化API\t726
 - 19.2.8 额外功能：处理JSON\t727

- 19.3 其他常见的O/X Mapping开源项目\t729
 - 19.3.1 JAXB\t729
 - 19.3.2 Castor\t733
 - 19.3.3 JiBX\t738
 - 19.3.4 总结比较\t741
- 19.4 与Spring OXM整合\t742
 - 19.4.1 Spring OXM概述\t742
 - 19.4.2 整合OXM实现者\t744
 - 19.4.3 如何在Spring中进行配置\t744
 - 19.4.4 Spring OXM简单实例\t747
- 19.5 小结\t749
- 第20章 实战型单元测试\t750
 - 20.1 单元测试概述\t751
 - 20.1.1 为什么需要单元测试\t751
 - 20.1.2 单元测试之误解\t752
 - 20.1.3 单元测试之困境\t754
 - 20.1.4 单元测试基本概念\t755
 - 20.2 TestNG快速进阶\t757
 - 20.2.1 TestNG概述\t757
 - 20.2.2 TestNG生命周期\t758
 - 20.2.3 使用TestNG\t758
 - 20.3 模拟利器Mockito\t763
 - 20.3.1 模拟测试概述\t763
 - 20.3.2 创建Mock对象\t763
 - 20.3.3 设定Mock对象的期望行为及返回值\t764
 - 20.3.4 验证交互行为\t766
 - 20.4 测试整合之王Unitils\t767
 - 20.4.1 Unitils概述\t767
 - 20.4.2 集成Spring\t770
 - 20.4.3 集成Hibernate\t773
 - 20.4.4 集成DbUnit\t774
 - 20.4.5 自定义扩展模块\t775
 - 20.5 使用Unitils测试DAO层\t776
 - 20.5.1 数据库测试的难点\t776
 - 20.5.2 扩展DbUnit用Excel准备数据\t776
 - 20.5.3 测试实战\t779
 - 20.6 使用Unitils测试Service层\t789
 - 20.7 测试Web层\t794
 - 20.7.1 对LoginController进行单元测试\t794
 - 20.7.2 使用Spring Servlet API模拟对象\t795
 - 20.7.3 使用Spring RestTemplate测试\t797
 - 20.8 小结\t798

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:www.tushu000.com