

DVD

含视频讲解
案例源码

布 局

Java EE 企业级开发：

寻觅框架和开发模式的完美整合

全面介绍Java EE技术的核心知识，内容深入、语言通俗
剖析了Struts、Spring、Hibernate、JSF和JNDI的使用
兼顾理论、案例的完美展现

张元亮 编著



清华大学出版社

布局 Java EE 企业级开发： 寻觅框架和开发模式的完美整合

张元亮 编 著

清华大学出版社
北京

内 容 简 介

本书可以帮助读者精心布局 Java EE 企业级开发, 并以此寻觅出框架与开发模式完美整合的真谛。

本书内容新颖、知识全面、讲解详细。全书分为 18 章, 内容都采用了理论加实践的讲述方法, 每个实例先提出制作思路及包含的知识点, 然后力求用最通俗的语言将高深的知识阐述出来。

本书适合 Java 各个级别的程序员、研发人员及在职程序员阅读, 也可以作为培训学校和大专院校相关专业的教学用书。

本书封面贴有清华大学出版社防伪标签, 无标签者不得销售。

版权所有, 侵权必究。侵权举报电话: 010-62782989 13701121933

图书在版编目(CIP)数据

布局 Java EE 企业级开发: 寻觅框架和开发模式的完美整合/张元亮编著. --北京: 清华大学出版社, 2013

ISBN 978-7-302-31204-8

I. ①布… II. ①张… III. ①JAVA 语言—程序设计 IV. ①TP312

中国版本图书馆 CIP 数据核字(2013)第 002461 号

责任编辑: 魏 莹 桑任松

装帧设计: 杨玉兰

责任校对: 周剑云

责任印制: 宋 林

出版发行: 清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址: 北京清华大学学研大厦 A 座 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

课 件 下 载: <http://www.tup.com.cn>, 010-62791865

印 刷 者: 清华大学印刷厂

装 订 者: 三河市新茂装订有限公司

经 销: 全国新华书店

开 本: 185mm×260mm 印 张: 46.25 字 数: 1120 千字

(附 DVD1 张)

版 次: 2013 年 3 月第 1 版

印 次: 2013 年 3 月第 1 次印刷

印 数: 1~4000

定 价: 86.00 元

产品编号: 046881-01

前 言

自从 Java 语言诞生以来，经过十多年的发展和应用，已经成为当今最流行的编程语言之一。在一项权威的编程语言排行榜中，Java 始终居于第一位。现在全球已有超过 15 亿部手机和手持设备应用了 Java 技术。同时，Java 平台因其跨平台特性和良好的可移植性，已成为广大软件开发技术人员的挚爱，是全球程序员的首选开发平台之一。

日益成熟的 Java 语言编程技术目前已经无处不在，使用该编程技术可以进行桌面程序、Web 应用、分布式系统和嵌入式系统应用的开发，并且在信息技术等各个领域得到了广泛的应用。

本书全面介绍了 Java EE 的核心知识，通过实例介绍了 Java EE 编程技术和开发过程。全书内容翔实深入，作者用通俗的语言将大师级的知识展现在读者的面前。

1. 本书内容

本书分为 18 章，具体内容安排如下：第 1 章讲解 Java EE 初体验的基本知识；第 2 章讲解 Struts 2 的基础知识；第 3 章讲解 Struts 2 的核心知识；第 4 章讲解 Struts 2 的高级知识；第 5 章讲解 Hibernate 的基础知识；第 6 章深入分析 Hibernate 映射的核心知识；第 7 章详细讲解 Hibernate 高级知识；第 8 章讲解 Spring 技术的基本知识；第 9 章讲解 Spring 的核心知识；第 10 章讲解 Spring 高级知识；第 11 章开始步入经典级 Java EE 开发的基本知识；第 12 章讲解 JSF 的基础知识；第 13 章讲解 JTA 事务处理的基本知识；第 14 章讲解 JNDI 接口的基本知识；第 15 章讲解使用 JavaMail 发送邮件的基本知识；第 16 章讲解会话 EJB 的基本知识；第 17 章讲解消息驱动 EJB 的基本知识；第 18 章讲解一个基于 MVC 的学校餐费管理系统的开发过程。全书内容都采用理论加实践的讲述方法，每个实例先提出制作思路及包含的知识点，然后力求用最通俗的语言将高深的知识阐述出来。

2. 本书特色

本书内容相当丰富，实例覆盖全面，可满足 Java 程序员成长道路上的多方面需求。我们的目标是通过一本图书提供多本图书的价值，读者可以根据自己的需要有选择地阅读，以完善自己的知识和技能结构。在内容的编写上，本书具有以下特色。

(1) 专家写作

本书是国内一线著名的 Java 专家级作者的力作。为了确保广度和深度，本书并没有将大量篇幅用在规范和基本语法上，而是专注于各个基本知识的具体细节，尽量涉及每个知识点中最为重要的内容，并且讨论了相关的高级用法技术。

本书既是介绍性书籍，又是深入研究的技术性书籍。本书实现了高级技术与介绍性知识并重的效果，为了实现这一目标，作者做过大量的研究。比如，参与论坛讨论，开发大量的实际项目，参加学术会议和研讨会，同时跟制定 Java 规范的专家组进行沟通，同全世界顶级专家进行合作。



(2) 结构合理

本书从用户的实际需要出发,科学安排知识结构,内容由浅入深,叙述清楚,具有很强的知识性和实用性,反映了 Java EE 的核心知识。同时全书精心筛选了最具代表性、读者最关心的典型知识点,几乎包括 Java EE 技术的各个方面。

(3) 易学易懂

本书条理清晰、语言简洁,可帮助读者快速掌握各个知识点;每个部分既相互连贯又自成体系,使读者既可以按照本书编排的章节顺序进行学习,也可以根据自己的需求对某一章节进行有针对性的学习。

(4) 由浅入深

本书从 Java 语言的发展、开发环境及基本语法入手,逐步介绍 Struts 2、Hibernate、Spring、JSF、NetBeans、JTA 和 JNDI 等知识。保证让读者在没有任何编程基础的情况下,也能够很快地掌握 Java EE 编程的各种技术。

(5) 实用性强

本书彻底摒弃枯燥的理论和简单的操作,注重实用性和可操作性,详细讲解各个部分的源码知识,使用户在掌握相关的操作技能的同时,还能学习到相应的基础知识。

3. 致读者

因为篇幅有限,所以实例中的有些代码没有在书中一一列出,给读者带来了不便,为此谨表歉意。希望读者在阅读本书时,参考本书附带光盘中的源代码。

4. 读者对象

本书的读者对象主要包括下列群体:

- | | |
|--------------------------------------|---------------------------------------|
| <input type="checkbox"/> 初学编程的自学者 | <input type="checkbox"/> 编程爱好者 |
| <input type="checkbox"/> 大中专院校的教师和学生 | <input type="checkbox"/> 相关培训机构的教师和学员 |
| <input type="checkbox"/> 搞毕业设计的学生 | <input type="checkbox"/> 初、中级程序开发人员 |
| <input type="checkbox"/> 程序测试及维护人员 | <input type="checkbox"/> 参加实习的初级程序员 |
| <input type="checkbox"/> 在职程序员 | <input type="checkbox"/> 资深程序员 |

在编写本书的过程中,得到了清华大学出版社工作人员的大力支持,在此表示感谢。

由于作者的知识水平有限,书中疏漏和不尽如人意之处在所难免,恳请读者提出宝贵的意见或建议,以便再版时修订并使之更臻完善。另外,为方便读者学习,特开通了技术支持 QQ 群,群号为 75593028,欢迎读者加入。

编 者

目 录

第 1 章 Java EE 初体验.....1	2.2.1 下载并设置 Struts 2..... 26
1.1 学习 Java 的巨大优势.....2	2.2.2 在 Eclipse 中使用 Struts 2..... 28
1.1.1 排名第一的编程语言.....2	2.2.3 开发 Struts 2 应用的基本 步骤..... 29
1.1.2 良好的就业前景.....2	2.3 配置 Struts 2..... 31
1.2 探讨 Java EE 体系.....3	2.3.1 配置常量..... 31
1.2.1 Java EE 的分层模型.....3	2.3.2 包含其他配置文件..... 35
1.2.2 Java EE 应用的组件.....4	2.4 实现 Action..... 36
1.3 为什么使用 Java EE.....5	2.4.1 基础知识..... 36
1.4 Java EE 企业级开发四步走完全攻略....6	2.4.2 标准 Action 类..... 37
1.5 Java EE 的主要框架.....6	2.4.3 ActionSupport 基类..... 38
1.5.1 Struts.....6	2.4.4 访问 Servlet API..... 42
1.5.2 Spring.....7	2.4.5 直接访问 Servlet API..... 47
1.5.3 Hibernate.....8	2.4.6 用 ServletActionContext 访问 Servlet API..... 48
1.5.4 Swing.....8	2.5 配置 Action..... 51
1.5.5 本书的内容.....8	2.5.1 命名空间和包..... 51
1.6 开发前的准备.....9	2.5.2 基本配置..... 54
1.6.1 安装 JDK.....9	2.5.3 使用 Action 的动态方法 调用..... 55
1.6.2 设置环境变量.....13	2.5.4 为 Action 配置 method 属性.... 58
1.6.3 JRE 和 JDK 是有区别的.....13	2.5.5 使用通配符映射方式..... 59
1.6.4 困扰初学者的环境变量 问题.....14	2.6 处理结果..... 59
1.7 安装并配置 Tomcat 服务器.....15	2.6.1 配置处理结果..... 59
1.7.1 获取 Tomcat..... 15	2.6.2 处理结果类型..... 60
1.7.2 配置 Tomcat 的服务端口.....18	2.6.3 动态返回结果..... 60
1.7.3 列出 Web 应用根路径下的 所有页面.....18	2.7 初学者应该明白的几个问题..... 64
1.7.4 登录控制台.....19	2.7.1 MVC 思想和观察者模式..... 64
1.7.5 设置虚拟目录.....21	2.7.2 Action 处理用户请求..... 64
第 2 章 Struts 2 基础.....23	2.7.3 Struts 2 的工作流程..... 64
2.1 MVC 思想.....24	2.7.4 一份完整 struts.xml 文件的 常量配置骨架..... 65
2.1.1 什么是 MVC 思想.....24	第 3 章 Struts 2 核心..... 69
2.1.2 MVC 思想及其优势.....25	3.1 异常处理机制..... 70
2.1.3 Struts 的 MVC 思想.....25	
2.2 下载与安装 Struts 2.....26	



3.2	Struts 2 的国际化	73	4.2.2	实现简单的文件上传	147
3.2.1	为什么需要国际化	73	4.2.3	实现文件过滤	150
3.2.2	认识 Struts 国际化组件	74	4.3	文件下载	153
3.2.3	访问资源包的方式	75	4.3.1	一个误区	153
3.2.4	输出带占位符的国际化消息	78	4.3.2	Struts 2 实现文件下载	154
3.3	OGNL 表达式	80	4.4	拦截器	157
3.3.1	OGNL 的优势	80	4.4.1	拦截器的原理	157
3.3.2	OGNL 的语法	80	4.4.2	配置拦截器<interceptors>	160
3.3.3	OGNL 的集合操作	83	4.4.3	使用拦截器	162
3.3.4	Lambda 表达式	83	4.4.4	自定义拦截器类	163
3.4	Struts 2 的标签库	84	4.4.5	使用拦截器	164
3.4.1	五类标签库	84	4.4.6	使用拦截方法的拦截器	165
3.4.2	控制标签	84	4.4.7	拦截器的执行顺序	167
3.4.3	数据标签	92	4.4.8	Struts 2 中拦截结果的 监听器	170
3.4.4	主题和模板	98	4.4.9	覆盖拦截器栈里特定 拦截器的参数	172
3.4.5	Struts 2 的内建主题	99	4.4.10	实现未登录拦截和权限 控制	173
3.4.6	Struts 2 的表单标签	100	4.5	在 Struts 2 中使用 Ajax	174
3.5	类型转换	107	4.6	初学者应该明白的几个问题	177
3.5.1	Struts 2 的类型转换机制	107	4.6.1	剖析短路校验器	177
3.5.2	Struts 2 内建的类型转换	108	4.6.2	拦截器的机理	178
3.5.3	基于 OGNL 的类型转换	108	第 5 章 Hibernate 基础		183
3.5.4	指定集合元素的类型	112	5.1	Hibernate 基础	184
3.5.5	自定义类型转换器(1)	114	5.1.1	Hibernate 概述	184
3.5.6	注册类型转换器	116	5.1.2	Hibernate API 简介	184
3.5.7	自定义类型转换器(2)	117	5.1.3	Hibernate 的核心接口	185
3.5.8	处理 Set 集合	118	5.1.4	Hibernate 的体系结构	186
3.6	初学者应该明白的几个问题	122	5.2	Hibernate 的下载和安装	186
3.6.1	创建自己的主题	122	5.2.1	下载 Hibernate	186
3.6.2	Action 处理用户请求	123	5.2.2	为 Eclipse 安装插件	188
第 4 章 Struts 2 进阶		125	5.3	认识几种简单配置 Hibernate 的 方式	189
4.1	输入验证	126	5.3.1	配置数据源	189
4.1.1	编写验证规则文件	126	5.3.2	配置 c3p0 连接池	190
4.1.2	国际化提示	129	5.3.3	配置 Proxool 连接池	191
4.1.3	客户端验证	130	5.3.4	MySQL 连接配置	192
4.1.4	配置校验规则	131	5.3.5	SQL Server 连接配置	193
4.1.5	内置校验器	132			
4.1.6	输入验证	141			
4.2	文件上传	146			
4.2.1	Struts 2 文件上传的机理	146			

5.3.6 Oracle 连接配置	193	6.2.3 组件作为 Map 的索引	233
5.4 Hibernate 配置	194	6.2.4 组件作为复合标识符	236
5.4.1 持久化操作实例	194	6.2.5 动态组件	237
5.4.2 创建 Configuration 对象	198	6.2.6 多列映射成联合主键	238
5.4.3 Hibernate 的 JDBC 连接	200	6.3 关联关系映射	239
5.4.4 数据库方言	201	6.3.1 关联关系基础	240
5.4.5 与 Hibernate 相关的常用 属性	202	6.3.2 单向 N-1 的关系映射	241
5.5 SessionFactory 接口	204	6.3.3 单向 1-1 的关系映射	244
5.6 Session 接口	205	6.3.4 单向 1-N 的关联	246
5.6.1 save()方法	205	6.3.5 单向 N-N 的关联	249
5.6.2 get()方法	206	6.3.6 双向 1-N 的关联	250
5.6.3 load()方法	207	6.3.7 双向 N-N 的关联	251
5.6.4 update()方法	207	6.3.8 双向 1-1 的关联	254
5.6.5 delete()方法	208	6.3.9 基于复合主键的关联关系	258
5.7 Transaction 接口	208	6.3.10 传播性持久化	261
5.8 Query 接口	209	6.4 继承映射	262
5.9 Criteria 接口	210	6.4.1 继承关系中每个具体类对应 一个表	262
5.10 持久化对象	210	6.4.2 基类(根类)对应一个表	263
5.10.1 持久化类的规则	211	6.4.3 每个类对应一个表	264
5.10.2 持久化的状态	211	6.5 初学者应该明白的几个问题	265
5.10.3 改变持久化对象状态	212	6.5.1 三种映射方式的比较 和选择	265
5.11 初学者应该明白的几个问题	213	6.5.2 使用 Hibernate 的场合	265
5.11.1 尽量使用数据源方式	213	6.5.3 两边指定值	266
5.11.2 何时使用 Hibernate 作为 持久化框架	214	第 7 章 Hibernate 进阶	267
5.11.3 分析集合性能	214	7.1 批量处理	268
第 6 章 Hibernate 映射	215	7.1.1 认识批量处理	268
6.1 Hibernate 映射文件	216	7.1.2 批量插入	268
6.1.1 映射文件结构	216	7.1.3 批量更新	270
6.1.2 主键映射	218	7.1.4 StatelessSession(无状态会话) 接口	270
6.1.3 普通属性映射	219	7.1.5 DML(数据操纵语言)风格的 操作	271
6.1.4 映射引用属性	220	7.2 HQL 查询	273
6.1.5 映射 Set 集合属性	225	7.2.1 HQL 的基本语法	273
6.1.6 映射 List 集合属性	227	7.2.2 实体对象与动态实例化对象 查询	274
6.1.7 其他属性	227	7.2.3 条件查询与别名的使用	274
6.2 组件映射	228		
6.2.1 组件属性	228		
6.2.2 集合属性的元素为组件	231		



7.2.4	HQL 语句的动态赋值	275	功能定义	315
7.2.5	对象导航查询	275	8.4.3 XmlBeanFactory 的工作原理	316
7.2.6	排序查询	276	8.4.4 ApplicationContext 的国际化	318
7.2.7	聚合函数	276	8.4.5 详解 ApplicationContext 的事件机制	320
7.2.8	分组操作	277	8.4.6 Bean 获取 Spring 容器	324
7.2.9	对日期时间的处理	277	8.5 Spring 中的 Bean	326
7.2.10	联合查询	278	8.5.1 定义 Bean	326
7.2.11	子查询	279	8.5.2 Spring 中的 Bean 的作用域	328
7.3	条件查询(Criteria Query)	279	8.5.3 配置依赖	330
7.3.1	Criteria 对象	279	8.5.4 属性值注入	331
7.3.2	动态关联	282	8.5.5 注入 field 值	334
7.3.3	离线查询和子查询	283	8.5.6 注入方法返回值	335
7.4	Hibernate 的数据过滤查询	284	8.6 Java 配置管理	338
7.5	Hibernate 事务控制	287	8.7 初学者应该明白的几个问题	340
7.5.1	Session 和事务范围	288	8.7.1 代码升级游刃有余	340
7.5.2	数据库事务声明	291	8.7.2 抽象 Bean 和子 Bean 的必要性	341
7.6	乐观并发控制	294	8.7.3 Spring 中 Bean 的继承与 Java 中继承的区别	341
7.6.1	应用程序级别的版本检查	294	第 9 章 Spring 核心	343
7.6.2	悲观锁定	296	9.1 与容器交互	344
7.7	初学者应该明白的几个问题	297	9.1.1 工厂 Bean 简介与配置	344
7.7.1	Hibernate 的性能技巧	297	9.1.2 FactoryBean 接口	346
7.7.2	Criteria 和 DetachedCriteria 的区别	298	9.2 Bean 的生命周期	347
第 8 章	Spring 技术	299	9.2.1 定制 Bean 的生命周期行为	348
8.1	Spring 技术介绍	300	9.2.2 协调不同步的 Bean	351
8.1.1	Spring 的优点	300	9.3 Bean 的继承	353
8.1.2	Spring 将改变 Java EE	300	9.3.1 使用 abstract 属性	353
8.2	下载并安装 Spring	300	9.3.2 定义子 Bean	355
8.2.1	获取 Spring 压缩包	300	9.4 零配置	358
8.2.2	在 Eclipse 中使用 Spring	302	9.4.1 搜索 Bean 类	358
8.2.3	在 Web 项目中使用 Spring	305	9.4.2 使用 @Resource 配置依赖	361
8.3	依赖注入	308	9.4.3 使用 @PostConstruct 和 @PreDestroy 定制生命周期行为	362
8.3.1	依赖注入基础	308	9.5 后处理器	363
8.3.2	设置注入	309		
8.3.3	代码升级游刃有余	311		
8.3.4	构造注入	312		
8.4	Spring 容器	313		
8.4.1	Spring 的 IoC 容器	314		
8.4.2	BeanFactory 对 IoC 容器的			

9.5.1	Bean 后处理器.....	363	10.5.2	Spring 中的 AOP 代理.....	423
9.5.2	容器后处理器.....	367	10.5.3	基于 XML 配置文件的 管理方式.....	424
9.6	Spring 资源访问.....	370	10.5.4	确定是否需要应用服务器 管理事务.....	424
9.6.1	URL 类和 Resource 接口.....	370	10.5.5	编程式还是声明式事务 管理的选择.....	425
9.6.2	Resource 的实现类.....	371	10.5.6	BeanNameAutoProxyCreator 是另一种声明方式.....	425
9.6.3	使用 Resource 作为属性.....	375	10.5.7	如果没有 Spring 框架 该怎么办.....	427
9.6.4	在 ApplicationContext 中 使用资源.....	376	10.5.8	Spring 的异常处理.....	428
9.7	初学者应该明白的几个问题.....	377	10.5.9	Spring 管理 Hibernate 的 SessionFactory 多种配置 方法.....	429
9.7.1	Spring 3.0 新增的 Annotation.....	377			
9.7.2	Bean 后处理器的用处.....	378			
第 10 章	Spring 进阶.....	379	第 11 章	开始步入经典级 Java EE 开发.....	433
10.1	AOP.....	380	11.1	经典级需要什么技术.....	434
10.1.1	AOP 的相关概念.....	380	11.1.1	JSP、Servlet 和 JavaBean	434
10.1.2	AOP 的作用.....	381	11.1.2	JSF.....	434
10.1.3	Spring AOP 的本质.....	383	11.1.3	EJB.....	435
10.1.4	使用 AspectJ.....	384	11.2	JBoss 的下载和安装.....	436
10.1.5	基于 Annotation 的 “零配置”方式.....	390	11.2.1	下载和安装 JBoss 服务器....	436
10.2	事务管理.....	396	11.2.2	配置 JBoss 的客户端.....	438
10.2.1	Spring 事务抽象.....	397	11.2.3	核心管理概念.....	442
10.2.2	事务策略.....	398	11.2.4	管理任务.....	455
10.2.3	编程式事务管理.....	400	11.2.5	JVM 设置.....	463
10.2.4	声明式事务管理.....	402	11.2.6	子系统配置.....	464
10.3	Spring + Struts 2 整合.....	403	11.2.7	在 Eclipse 中集成 JBoss 7.....	467
10.3.1	启动 Spring 容器.....	404	11.2.8	在 JBoss 7 中建立 MySQL 数据源.....	470
10.3.2	Spring 管理 Struts 2 的 控制器.....	405	11.2.9	JBoss 7 配置 Oracle 10g 数据源.....	472
10.4	Spring + Hibernate 整合.....	407	11.3	下载并安装 WebLogic.....	473
10.4.1	DAO 支持.....	407	11.3.1	WebLogic 的特色优势.....	473
10.4.2	管理 Hibernate 的 SessionFactory.....	413	11.3.2	下载并安装 WebLogic.....	474
10.4.3	Spring 对 Hibernate 的 简化.....	414	11.3.3	创建域.....	479
10.4.4	使用 HibernateTemplate.....	416	11.3.4	发布应用程序.....	483
10.4.5	Hibernate 的 DAO 实现.....	420	11.3.5	配置服务.....	486
10.5	初学者应该明白的几个问题.....	423			
10.5.1	Spring AOP 的功能.....	423			

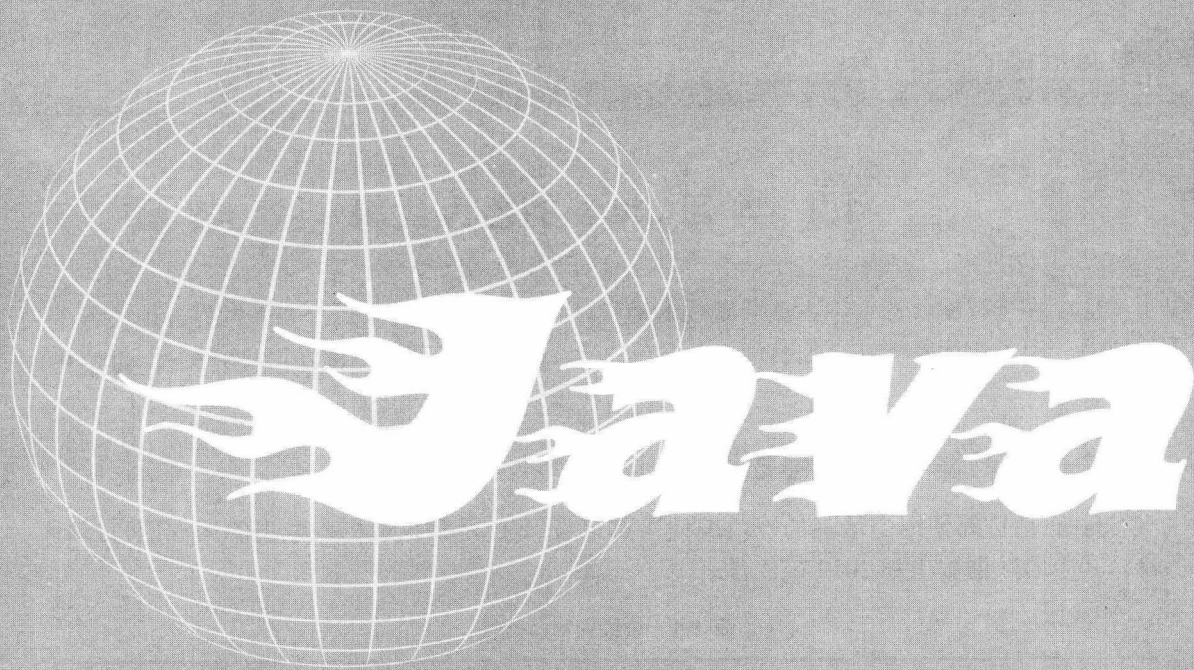


11.4 使用 NetBeans	492	12.7.3 NumberConverter	536
11.4.1 下载 NetBeans	492	12.7.4 Faces 验证系统	536
11.4.2 安装 NetBeans	493	12.8 初学者应该明白的几个问题	538
11.4.3 使用 NetBeans 新建项目	495	12.8.1 解决 JSF 冗余代码和管理	
11.5 初学者应该明白的几个问题	497	混乱的问题	538
11.5.1 用脚本启动和停止		12.8.2 在 JSF 框架中使用的	
JBoss 7	498	设计模式	539
11.5.2 使用 JBoss-native 提高处理		12.8.3 使用 JSF 的原则和技巧	540
静态文件的速度	498		
11.5.3 防止重启 WebLogic 时 JSP		第 13 章 JTA 事务处理	541
重编译	499	13.1 事务的基本概念	542
11.5.4 总结 WebLogic 的配置		13.1.1 Java 中事务处理的由来	542
技巧	499	13.1.2 事务的几个特性	544
第 12 章 JSF 介绍	501	13.1.3 Java EE 中的事务管理器	545
12.1 JSF 简介	502	13.2 分布式事务处理	546
12.2 下载并安装 JSF	503	13.2.1 基本概念	547
12.2.1 配置 JSF	503	13.2.2 两阶段提交	547
12.2.2 JSF 的环境配置	507	13.2.3 XA 规范	548
12.3 详解 JSF 配置文件的说明和常用		13.2.4 为什么使用分布式事务	548
配置元素	509	13.3 全局事务和局部事务	549
12.3.1 在 web.xml 文件中配置		13.4 在同一事务中 JTA 实现 Oracle、	
FacesServlet 核心控制器	510	SQL Server	550
12.3.2 JSF 的配置文件		13.5 在 WebLogic 服务器用 JTA 实现	
faces-config.xml	511	多种数据库的一致性	557
12.4 使用 JSF	511	13.6 EJB 3.0 的事务管理	561
12.5 导航	515	13.6.1 容器管理事务	561
12.5.1 静态导航	515	13.6.2 Bean 管理事务	563
12.5.2 动态导航	516	13.7 初学者应该明白的几个问题	564
12.5.3 通配符	521	13.7.1 关于分布式事务必须清楚的	
12.5.4 使用 from-action	522	几点	564
12.6 JSF 的核心标签	522	13.7.2 使用 JTA 和 JDBC	565
12.6.1 JSF 核心标签概述	522	13.7.3 如何选择最好的 JTA 方式	565
12.6.2 JSF HTML 标签	524	13.7.4 在 JavaBean 中用 JDBC 方式	
12.6.3 表单	530	处理事务	566
12.6.4 文本字段和文本区域	531	第 14 章 JNDI 接口	567
12.6.5 按钮和链接	532	14.1 JNDI 基础	568
12.7 数据转换和数据验证	534	14.1.1 命名服务和目录服务	568
12.7.1 Faces 转换器系统	534	14.1.2 为什么会有 JNDI	569
12.7.2 DateTimeConverter	536	14.2 JNDI 的工作原理	570

14.3	目录操作.....	574	15.5.8	处理 HTML 消息.....	613
14.4	分析主流服务器对 JNDI 的支持.....	577	15.5.9	用 SearchTerm 搜索.....	614
14.4.1	WebLogic 对 JNDI 的支持.....	577	15.6	JavaMail 的常见应用.....	614
14.4.2	JBoss 对 JNDI 的支持.....	583	15.6.1	Gmail 收发信.....	615
14.5	初学者应该明白的几个问题.....	586	15.6.2	JavaMail 收取邮件属性配置.....	617
14.5.1	在 Tomcat 中配置 JNDI 对象.....	586	15.6.3	JavaMail 发送邮件.....	619
14.5.2	在 Servlet 中访问 Tomcat JNDI 资源.....	587	15.6.4	GmailFetch 收取 Gmail 邮件.....	623
14.5.3	在 JSP 中访问 Tomcat JNDI 资源.....	591	15.6.5	Gmail Sender 发送 Gmail 邮件.....	624
第 15 章	使用 JavaMail 发送邮件.....	593	15.7	初学者应该明白的几个问题.....	625
15.1	JavaMail 介绍.....	594	15.7.1	JavaMail 中的常见中文乱码问题与解决办法.....	625
15.2	邮件协议介绍.....	594	15.7.2	如何配置 JavaMail 通过代理服务器工作.....	627
15.2.1	SMTP 协议.....	594	15.7.3	授予特殊的权限.....	627
15.2.2	POP 协议.....	595	第 16 章	会话 EJB.....	629
15.2.3	IMAP 协议.....	595	16.1	EJB 3.0 介绍.....	630
15.2.4	MIME 协议.....	595	16.1.1	长久以来对 EJB 的误区.....	630
15.2.5	NNTP 和其他协议.....	595	16.1.2	EJB 的现状.....	631
15.3	JavaMail 基础.....	596	16.1.3	EJB 的优势.....	631
15.3.1	JavaMail 简介.....	596	16.2	无状态的 Session Bean.....	633
15.3.2	安装 JavaMail.....	596	16.2.1	开发远程调用的无状态 Session Bean.....	634
15.4	JavaMail 核心类详解.....	598	16.2.2	开发本地调用的无状态 Session Bean.....	639
15.4.1	java.util.Properties 类.....	598	16.3	发布 Session Bean.....	642
15.4.2	javax.mail.Session 类.....	599	16.4	编写有状态的 Session Bean.....	642
15.4.3	javax.mail.Authenticator 类.....	600	16.5	使用 Session Bean 的本地接口.....	644
15.4.4	javax.mail.Message 类.....	602	16.6	Session Bean 中的标注方法.....	645
15.4.5	javax.mail.Address 类.....	604	16.7	使用配置文件发布 Session Bean.....	646
15.4.6	javax.mail.Transport 类.....	605	16.8	编写一个实体 Bean 程序.....	647
15.4.7	javax.mail.Store 类和 javax.mail.Folder 类.....	606	16.8.1	配置 JBoss 的数据库连接池.....	648
15.5	使用 JavaMail API.....	606	16.8.2	编写实体 Bean.....	648
15.5.1	发送消息.....	607	16.8.3	配置 persistence.xml 文件.....	649
15.5.2	获取消息.....	607	16.8.4	在 Session Bean 中调用实体 Bean.....	649
15.5.3	删除消息和标志.....	609			
15.5.4	自我验证.....	609			
15.5.5	回复消息.....	610			
15.5.6	转发消息.....	610			
15.5.7	操作附件.....	611			



16.8.5 客户端调用 Session Bean	650	第 18 章 学校餐费管理系统	699
16.9 实现 Entity Bean 的一对一映射	650	18.1 功能模块划分	700
16.10 实现 Entity Bean 的一对多映射	654	18.1.1 需求分析	700
16.11 EJB 拦截器	656	18.1.2 系统功能模块	700
16.12 初学者应该明白的几个问题	658	18.2 系统分析和设计	701
16.12.1 总结 Session Bean 的编程 规则	659	18.2.1 分析、设计数据库	701
16.12.2 Spring 与 EJB 3.0 的对比 ...	659	18.2.2 设计业务逻辑层 和 DAO 层	702
16.12.3 如何在 Spring 中访问 EJB	664	18.2.3 规划系统包	703
第 17 章 消息驱动 EJB	667	18.2.4 构建系统的 MVC 结构	703
17.1 消息服务基础	668	18.3 配置开发环境	705
17.1.1 JMS 模型	668	18.3.1 在 Eclipse 中配置 Struts	705
17.1.2 JMS 编程模型	669	18.3.2 在 Eclipse 中配置 Spring	706
17.1.3 分析 Queue 类型的发送 和接收流程	671	18.3.3 在 Eclipse 中配置 Hibernate	706
17.1.4 分析 topic 类型的消息发送 和接收流程	675	18.3.4 在 Eclipse 中配置 Hibernate Synchronizer	707
17.1.5 实现 Java 消息服务(JMS)	678	18.4 具体编码	707
17.2 MDB 基础	682	18.4.1 建立视图	707
17.2.1 Queue 消息的发送与接收 (PTP 消息传递模型)	683	18.4.2 建立 JSP 页面	709
17.2.2 Topic 消息的发送与接收 (Pub/Sub 消息传递模型)	683	18.4.3 设置固定部分	710
17.2.3 消息选择器	684	18.4.4 实现国际化	711
17.3 使用消息驱动 Bean	686	18.4.5 设置可变部分	712
17.4 使用 NetBeans 开发 EJB 3 会话 Bean	690	18.4.6 建立控制部分	713
17.5 初学者应该明白的几个问题	695	18.4.7 自定义的 Action	716
17.5.1 理解 JMS 和 EJB 的关系	696	18.4.8 错误处理	717
17.5.2 利用 Spring 实现 EJB	696	18.4.9 建立模型部分	718
17.5.3 实现 EJB 与 Spring 的 集成	698	18.4.10 建立业务逻辑类	719
		18.4.11 建立 DAO 类	720
		18.4.12 装配组件	722
		18.5 项目调试	724
		参考文献	726



第 1 章



Java EE 初体验

Java 语言是当今使用率最高的一门编程语言，从它诞生的那一天起，便被广大程序员所接受，并运用到世界各地的项目中。针对企业级应用的开发，Sun 公司专门为程序员推出了 Java EE。到今天为止，Java EE 已经成为 Java 企业级开发的核心。为什么 Java 语言的使用率如此之高？在 Java EE 开发过程中我们应该选择哪一个框架？希望读者仔细阅读本章内容，相信读完之后会对 Java EE 有一个深层次的印象，这些问题也会迎刃而解。





1.1 学习 Java 的巨大优势

Java 在当代编程语言中占据着重要的地位，用 Java 开发的项目分布在世界各地的各个领域。为了让读者能够更加有信心地学好 Java 语言，本节我们将首先介绍学习 Java 语言的优势。

1.1.1 排名第一的编程语言

Java 是目前使用率最高的一门编程语言。表 1-1 是某权威机构做出的编程语言排行榜，其中的数据截止到 2012 年 2 月。

表 1-1 编程语言排行榜

排名	语言	使用率(%)
1	Java	17.050
2	C	16.523
3	C#	8.653
4	C++	7.853
5	Object-C	7.062
6	PHP	5.641

1.1.2 良好的就业前景

由表 1-1 的统计数据可以看出，Java 语言是当今使用率最高的编程语言。这也就决定了 Java 程序员的就业机会要大于使用其他编程语言的开发者。

Java 的功能比较强大，在服务器领域、移动设备、桌面应用和 Web 领域都占据了重要的地位。

- ❑ 服务器领域：Java 在服务器编程方面很强悍，拥有很多其他语言所没有的优势。
- ❑ 移动设备：Java 在手机领域的应用比较广泛，手机 Java 游戏随处可见，当前异常火爆的移动开发平台 Android 上面的应用项目基本上都是用 Java 开发的。
- ❑ 桌面应用：Java 与 C++、.NET 一样重要，影响着桌面程序的发展。
- ❑ Web 领域：Java Web 有着巨大的优势，无论是开发工具还是开发框架都是开源的，并且安全性更强。

正是因为 Java 语言可以在多个领域中开发项目，所以市面中需要多个领域的 Java 程序员。据统计，当前全球有 30 亿个 Java 器件正在运行着 Java 程序，有 500 多万 Java 开发者活跃在全球的各个角落，数以千万计的 Web 用户每次上网都会亲历 Java 的威力。

今天，Java 运行在 9.08 亿部手机、10 亿张智能卡和 10 亿台 PC 机上，并为 28 款可兼容的应用服务器提供了功能强大的平台。这么多应用，彻底地改变了用户的生活。越来越多

的企业因为使用了 Java 而提高了生产效率。在中国,越来越多的用户因为 Java 而降低了成本,享受了生活。

作为唯一在互联网上开发的语言,Java 平台以其移动性、安全性和开放性受到追捧。据 IDC 预计,自 2010 年起的其后 5 年内,采用 Java 的 IT 产品的价值将翻番,在 2015 年将达到 45.53 亿美元,年增长率为 14.9%。截止到 2010 年 5 月,Java 注册开发商超过 1300 万人,对 JRE(Java 运行环境)的下载达 9200 万次。詹姆斯·戈士林博士预计在 3~5 年内 Java 技术开发商将发展到 2000 万,无线 Java 也在迅速攀升。

上述发展趋势在国内更是如此,当前我国对软件人才的需求旺盛,并且以每年 20%左右的速度增长。在未来 5 年内,合格软件人才的需求将远大于供给。在过去的 2012 年,我国软件人才的缺口已达 52.6 万人,其中尤以 Java 人才最为缺乏。

根据 IDC 的统计数字,在所有软件开发类人才的需求中,对 Java 工程师的需求达到全部需求量的 60%~70%。这也就不难解释在智联招聘和 51Job 等主流招聘站点中,为何随处可见 Java 工程师的招聘广告了。

1.2 探讨 Java EE 体系

2006 年, Sun 公司提出了 Java EE 这一概念,并与之同步推出了两个主要规范: JSF 1.2 和 EJB 3.0,但应用依然不如 SSH(Struts + Spring + Hibernate)组合的应用广泛。SSH 组合是一种轻量级的 Java EE 平台,具有高度的实用性和可扩展性。基于轻量级 Java EE 平台的应用可以运行在普通 Web 容器中,无须 EJB 容器的支持,且一样具有稳定的性能和极高的可扩展性、可维护性。

在 Java 体系中,Java Web 和 Java EE 一直是程序员、教师、初学者口中的常说的名词之一。究竟它们二者之间有什么关系呢?本节将简要地介绍一下。

- Java Web: 看名字就知道,是指利用 Java 语言开发 Web 项目的一种技术,与 ASP、ASP.NET、PHP 等功能类似。如果详细一点说,Java Web 主要包括 JDBC、JSP、Servlet、JavaBean、HTML、JavaScript、Session/Cookie、MVC 设计模式、Tomcat、Eclipse 和 MyEclipse 等知识。
- Java EE: 是指企业级的应用开发,是 Java Web 之后的更高级阶段,通常说的 Java EE 是指 SSH,SSH 是 Spring、Struts 和 Hibernate 架构应用整合开发的缩写,另外它还包括 XML、EJB、WebService、UML/Rose、Ajax、Weblogic 及 Oracle 等知识。

从上述两者包含的技术看,差异一目了然,Java Web 是指传统意义上的 Web 开发技术,而 Java EE 是 Java Web 的更高级产物,是企业级的应用开发模式。本书将带领读者详细研究 Java Web 技术的精粹,并且带领读者探索高高在上的 Java EE。

1.2.1 Java EE 的分层模型

经典 Java EE 应用往往以 EJB(企业级 Java Bean)为核心,以应用服务器为运行环境,



所以通常开发、运行成本较高。而当前应用最广泛的是 SSH，通常被称为轻量级 Java EE，SSH 具备了 Java EE 规范的种种特征，例如面向对象建模的思维方式、优秀的应用分层及良好的可扩展性、可维护性。轻量级 Java EE 应用已经超出了 Sun 公司所提出的经典 Java EE 应用规范，是一种更广泛的开发规范。

无论是经典的 Java EE 架构，还是本书所介绍的轻量级 Java EE 架构，大致上都可分为如下几层。

- ❑ Domain Object(领域对象)层：此层由系列的 POJO(Plain Old Java Object，普通的、传统的 Java 对象)组成，这些对象是该系统的 Domain Object，往往包含了各自所需要实现的业务逻辑方法。
- ❑ DAO(Data Access Object，数据访问对象)层：此层由系列的 DAO 组件组成，这些 DAO 实现了对数据库的创建、查询、更新和删除(CRUD)等原子操作。
- ❑ 业务逻辑层：此层由系列的业务逻辑对象组成，这些业务逻辑对象实现了系统所需要的业务逻辑方法。这些业务逻辑方法可能仅仅用于暴露 Domain Object 对象所实现的业务逻辑方法，也可能是依赖 DAO 组件实现的业务逻辑方法。
- ❑ 控制器层：此层由系列控制器组成，这些控制器用于拦截用户请求，并调用业务逻辑组件的业务逻辑方法，处理用户请求，并根据处理结果转发到不同的表现层组件。
- ❑ 表现层：此层由系列的 JSP 页面、Velocity 页面、PDF 文档视图组件组成，负责收集用户请求，并显示处理结果。

上述各层的 Java EE 组件之间以松耦合的方式耦合在一起，各组件并不以硬编码方式耦合，这种方式是为了应用以后的扩展性。从上向下，上面组件的实现依赖于下面组件的功能；从下向上，下面组件支持上面组件的实现。

至于以 EJB 3、JPA 为核心的 Java EE 应用的结构，与上述应用结构大致相似，只是它的 DAO 层(一般称为 EAO 层)组件、业务逻辑层组件都由 EJB 充当。

1.2.2 Java EE 应用的组件

Java EE 应用提供了系统架构上的飞跃，通过 Java EE 架构提供了良好的分离，隔离了各组件之间的代码依赖。一般来说，Java EE 应用主要包括如下 5 大类组件。

- ❑ 表现层组件：主要负责收集用户输入数据，或者向客户显示系统状态。最常用的表现层技术是 JSP，但 JSP 并不是唯一的表现层技术。表现层还可用 Velocity、FreeMarker 和 Tapestry 等技术来完成，或者使用普通的应用程序充当表现层组件，甚至可以是小型智能设备。
- ❑ 控制器组件：对于 Java EE 的 MVC 框架来说，框架提供一个前端核心控制器，而核心控制器负责拦截用户请求，并将请求转发给用户实现的控制器组件。而而这些用户实现的控制器则负责处理调用业务逻辑方法，处理用户请求。
- ❑ 业务逻辑组件：这是系统的核心组件，实现系统的业务逻辑。通常，一个业务逻辑方法对应一次用户操作。一个业务逻辑方法应该是一个整体的，因此我们要求对业务逻辑方法增加事务性。业务逻辑方法仅仅负责实现业务逻辑，不应该进行