

第一步

你应该了解真相，真相会使你自由。
——《圣经》

Hello, World

JSF 第一步

——JSF+Spring+Hibernate+AJAX编程实践

罗会波 著

- 为什么是**JSF**而不是Struts
- 释放**轻量级Java EE**应用三剑客（JSH）的力量
- 利用**AJAX**来改善JSF应用程序的用户体验
- 对**概念**的地道讲解+对**疑难点**的透彻剖析
- 从**Hello World**开始，直至完整**项目实例**
- 本书基于**最新版JSF 1.2**

● 附源代码光盘

清华大学出版社



原创经典

JSF 第一步

——JSF+Spring+Hibernate+AJAX 编程实践

罗会波 著

清华大学出版社
北 京

内 容 简 介

JSF 是表示层框架的标准, Hibernate 是一个比较完善的对象关系映射工具, Spring 则提供了一个 Web 应用的轻量级的解决方案。在开发一个多层的 Java EE 应用程序时, 这些框架可谓是相辅相成、相得益彰, 可以称得上是开发轻量级 Java EE 应用的三剑客。另外, AJAX 是一种非常流行的改善用户体验的技术, 但目前国内外还没有一本完整地介绍组合这三种框架及 AJAX 来开发应用程序的书籍, 本书是在这方面的一个尝试。

本书详尽地介绍了 JSF 的基本结构和工作原理、标签、验证与转换、事件处理、国际化编程及 IoC; 对象关系映射的概念、Hibernate 的结构及与 JSF 的集成; Spring 的结构、AOP、Spring 与 JSF 和 Hibernate 的集成; 常用 IDE (Eclipse 和 MyEclipse) 的使用方法等内容。

本书的特点主要是从编程实践的角度来介绍这些框架的基本原理, 以及用示例演示如何具体应用这些框架来开发一个 Web 应用程序。在介绍各种框架时, 对涉及到的 Web 开发中容易遇到的疑难问题尽可能给出浅显易懂的解释, 其目的是帮助想利用这些框架进行开发的人员尽快进入实战状态。

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 13501256678 13801310933

图书在版编目 (CIP) 数据

JSF 第一步: JSF+Spring+Hibernate+AJAX 编程实践 / 罗会波著. —北京: 清华大学出版社, 2007.10
ISBN 978-7-302-15973-5

I. J… II. 罗… III. JAVA 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2007) 第 128839 号

责任编辑: 陈 冰

责任校对: 张 剑

责任印制: 王秀菊

出版发行: 清华大学出版社 地 址: 北京清华大学学研大厦 A 座

<http://www.tup.com.cn> 邮 编: 100084

c-service@tup.tsinghua.edu.cn

社 总 机: 010-62770175 邮购热线: 010-62786544

投稿咨询: 010-62772015 客户服务: 010-62776969

印 刷 者: 清华大学印刷厂

装 订 者: 三河市金元印装有限公司

经 销: 全国新华书店

开 本: 203×260 印 张: 35.75 字 数: 1030 千字

版 次: 2007 年 10 月第 1 版 印 次: 2007 年 10 月第 1 次印刷

印 数: 1~5000

定 价: 65.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题, 请与清华大学出版社出版部联系调换。联系电话: (010)62770177 转 3103 产品编号: 024354-01

About the Author

[作者简介]



罗会波 高级工程师，IBM认证DB2数据库系统管理员。早年从事中学物理教学工作，后在亿利达（Elite）集团明泰R/D部（深圳）从事通信产品研发工作。1996年底进入税务系统从事信息化工作至今。

在亿利达工作期间开始接触汇编语言编程；在税务系统主持或参与了“湖北省国税局纳税大厅Web查询系统”、“全国税务征管软件Ctais纳税人查询系统”等数个省局及市局税务信息化项目的开发工作；参与了湖北省国税局内部教材《国税系统办公自动化实用指南》一书的编撰工作；曾在《计算机世界》、《电脑爱好者》及《电脑编程技巧与维护》等报刊杂志及赛迪网上发表过包括《Struts原理与实践》系列讲座（该系列曾长期占据赛迪网技术文章排行榜TOP10）在内的多篇原创及翻译的IT技术文章。文风严谨，又不失幽默，受到广大读者的一致好评。

前言

为什么要写这本书

早在 2004 年,我根据在项目中使用 Struts 的经验,写出了一篇名为“Struts 的原理与实践”的文章,投到某杂志社,该杂志的编辑觉得文章还过得去,只是认为仅一篇文章会给人一种意犹未尽的感觉,他建议我写一个系列。后来,该杂志因故停刊,杂志编辑说:“其实,可以将这些文章写成一本书。”还热心地说,如果要出书,他可以帮助推荐给出版社的编辑。

由于当时我忙于其他事务,没有时间写书。那时恰逢赛迪网征集这方面的文稿,就把这些写成了的文章寄给赛迪网,文章在赛迪网上发表后,引起了较大的反响,在相当长的一段时间里排在该网站的技术应用栏目的技术文章排行榜的前列,并被许多网站转载。

后来,当有空的时候,就想起了那位编辑关于写书的建议。我就在想,写一本什么样的书呢?光阴荏苒,很快,两年的时间就悄无声息地过去了。大家不要忘了,这个时代对 Java 来说是一个礼崩乐坏的时代。历史反复证明,这样的时代也是牛人辈出的时代。随之,各种新的思想、新的技术井喷一样地呈现在人们的面前。

在这期间,Java 世界也已经和正在发生剧烈的变化。轻量级的 J2EE 应用程序开发已经成为一种时尚。框架也不再是 Struts 的一统天下,各种新的框架层出不穷。在这些框架中最引人注目的是 JSF,居然成了 Java EE 规范的一部分,这在框架的发展史上好像是绝无仅有的。

在选择该书的表现层框架时,我是这样考虑的,虽然目前国内 SSH 方兴未艾,但 JSF 不仅是标准,有众多大公司的支持,而且在实践中也日益成熟起来,如常被人们诟病的早期版本中的那些错误在 JSF 1.2 版中都得到了很好的解决。它的组件模型、事件驱动及依赖注入等比老式的框架如 Struts 的确提供了更高的抽象和更强大的功能。国外的调查也表明它的市场占有率在不断扩大。从论坛的帖子也可以看出,如果是做一个新项目一般也是建议使用 JSF。不过,总的说来,与国外人们对 JSF 的关注程度相比,国内就显得冷清多了。表现在论坛中的帖子不多,市面上关于它的图书更是少之又少。这种局面是很不正常的,是不是又要等到落后一大截后,才想起来追赶呢?到那时可能就悔之晚矣。我也希望本书在改变这种不相称的局面方面起到些许作用。

而且,框架也不再局限于表示层,而是向 J2EE 多层应用程序的各个层面发展,如集成层就出现了 Hibernate (这里要说明一下,有的人不认为 Hibernate 是框架)。Spring 就更是涉及到应用程序的多个层。这些框架的涌现,一方面给了开发者更多的选择,另一方面也增加了开发人员的学习量。特别是由于各种框架都有其自己最擅长的一面,实际应用中往往会组合使用多个框架,以便充分发挥它们的特长,在实践中形成了一些常见的组合,如 Struts+Spring+Hibernate (即人们常说的 SSH)、JSF+Spring+Hibernate 等组合,这就更进一步地加大了初学者的学习难度。

反观市面上有关这方面的一些书籍。它们大多是分门别类地介绍单一的技术。比如专门介绍 Struts 的书、专门介绍 JSF 的书、专门介绍 Hibernate 的书以及专门介绍 Spring 方面的书。这些书的写法往往是这样的,即除了介绍某种单一的技术外,其他部分可能就是根据作者的特长随意使用一些其他的技术,比如介绍 Struts 的书,它的数据访问部分不是直接使用 JDBC 的 API 就是使用 EJB,而不会采用编程实践中采用得较多的 SSH 组合。偶尔看到一些书介绍与其他框架组合使用的书,如一些 Spring 书籍中介

绍 Spring 与 Struts 的组合、Spring 与 Hibernate 的组合，往往是蜻蜓点水般的一般性介绍。可以说，到目前为止，无论是国内还是国外，都还没有一本真正意义上的较全面地介绍用 JSF+Spring+Hibernate 框架进行开发的书籍，而实际的开发过程中却是综合应用这些技术（例如，使用 AJAX 来改善用户体验），这样就难免学用脱节。有没有办法帮助初学者尽快掌握这些技术呢？想到这里眼前豁然一亮。

何不写这样一本书呢？从编程的实践角度出发，比较全面地介绍 JSF 框架的基础知识及其他框架最基本、最常用的知识，以及在一个真实的开发过程中如何综合地应用它们。以便尽快引领初学者综合应用这些框架。本书就是这种想法的一个尝试。

为本书的问世做出贡献的人们

2006 年国庆节前的一天，我在一个编程论坛中看到一个清华大学出版社征集书稿的帖子，就按照帖子给出的邮件地址给发帖子的编辑“冰冰”发了一个 Email，在该 Email 中谈了写这本书的一些初步打算。本没有抱很大的希望，出乎意料的是很快就收到编辑的来信，编辑的真名叫陈冰，他本人原来就是一位计算机图书作者。他让我写一份简历，连同以前发表过的一些文章一起发给他。我就简单地罗列了一下自己的学习和工作经历，并收集了一下以前发表的文章给他寄去。因为我以前从未写过书，更担心自己才疏学浅，因此把自己的一些顾虑也写在 Email 中。这里不得不提的是，他的回信中的一句鼓励的话：“英雄不问出处！”让我至今难忘，好一个英雄不问出处，正是这份信任，让我们开始了接下来的坦诚合作。坦率地说，没有陈冰先生和出版社领导们的这份信任就不会有该书的问世。

几封 Email 往来，一个初步的目录和样章就出来了，书名也商量好了。国庆节后不久正式的出版合同就签订了。拿到合同后，才发现自己正处于一个多事之秋。工作上，金税工程系统升级、综合征管系统数据省级集中等工作接踵而至；家庭方面则因为要迁居而装修房子。写书稿只有晚上和节假日才有时间。

凡事空想容易，实干难。写书尤其如此。一方面因为时间紧张，进度很难保证。另一方面又往往为找不到好的思路和合适的例子来把问题讲得尽量浅显易懂而着急，深感压力沉重。着急就容易上火，说起话来分贝值就免不了比平时大了许多，我的妻子在这方面的感受可能最深，她为此受了不少委屈。此外，在写稿期间，她承担了所有的家务、悉心照料我和儿子的饮食起居。为我分担了不少压力。平心而论，本书也凝聚着她的不少心血。

能够顶住压力完成本书，还要感谢我的父母，是他们的言传身教使我懂得了一分耕耘，一分收获；不怕慢，只怕站等这些质朴而让人受用终身的道理；是他们的淳朴善良、乐于助人的品格，让我体会到予人玫瑰、手有余香的真谛，愿意将自己在学习和工作中的一些心得拿出来与大家分享。

能够完成本书，还要感谢教育过我的老师们，他们不仅传授给我知识，培养了我的技能，还让我懂得了业精于勤荒于嬉，行成于思毁于随的道理。

能够完成本书，还要感谢领导的关怀和同事们的帮助。是领导们为我提供了不少学习的机会，使得我有幸接触到一些新的技术，同时也给我提供了主持或参与一些项目的机会，使得我可以在实践中应用所学的知识。在该书的书写期间我的同事们主动分担了更多的工作任务，同事杨蜜还帮助校对了部分章节的文稿，找出了其中的一些笔误及标点符号错误。在此，再一次对所有为本书做出贡献的人们表示诚挚的感谢！

本书的结构与写法

第 1 章概括地介绍了 J2EE 应用程序的基本知识，以及一些常用框架的产生。

第 2 章介绍了 Web 应用程序的一些基础知识，为正式介绍 JSF 做一些必要的铺垫。

第3~15章介绍了JSF的一些基本知识。包括JSF的请求处理生命周期、基本构成、JSF标签、表达式语言与托管Bean、转换与验证，国际化编程，以及与Tile的集成和访问数据库等。

第16章介绍J2EE应用程序的异常处理。

第17章介绍了应用程序中的日志记录方法。

第18章介绍JSF与AJAX的集成，其内容是如何利用AJAX来改善JSF应用程序的用户体验。

第19~20章主要介绍Hibernate的基本知识及其与JSF的集成。其中，第19章介绍了对象关系映射的基本概念和面临的一些问题，以及Hibernate的基本构成，和一些常见的对象关系映射。还介绍了检索对象的方法。第20章则介绍Hibernate和JSF的集成。

第21章主要介绍Spring的一些基本知识及与JSF和Hibernate的集成。

第22章介绍常见的IDE（Eclipse和MyEclipse组合）的使用。

第23章对面向方面编程（AOP）做了一个简单的介绍，并介绍了事务管理的有关内容。

第24~26章是综合应用示例。其中，第24章介绍的是一个基于角色的访问控制系统，在该示例中，以一个程序员的眼光来看JSF、Hibernate和Spring在一个比较真实的应用中各自所处的位置及它们间的相互关系。第25章介绍的是一个名为办公用品管理系统的示例应用程序，在该示例中，从软件工程的角度出发，来演示一个项目的需求分析、设计、编码、测试等各个阶段。第26章是讲解应用程序的安全性。主要介绍了Java安全架构的一些基本知识。结合示例应用介绍了如何防止用户绕过登录环节；并介绍了口令加密的基本方法。

本书在写法上也做了一些尝试，既不像有的书高高在上地专门讨论一些高深的概念，使读者看完全书后都不容易做出一个简单的应用程序；也不像有的书全部是一些示例的杂乱堆砌，搞得一本书没有主线、没有灵魂。本书采用的方式是对那些最基本的概念先做一些解释，用它们去指导实践，然后用一些简单的示例来检验这些理论方面的东西。采用这种学用结合的方式可能更有助于读者对那些基本知识的理解。

本书在写法上的另一个特点，就是不平均使用力量。对那些在编程实践中经常容易遇到的，也是论坛中被频繁问及的一些问题，会花费更多的笔墨去解释。这方面的例子很多，比如，讲MVC时就有很多人对耦合与解耦的概念产生模糊认识；再如，关于回调的理解、关于乱码的问题，以及Java中的日期时间问题等等，我都会不惜笔墨，设法尽可能把它们讲得浅显易懂，为读者克服这些障碍提供一些切切实实的帮助。相反，对那些用得不是很普遍的东西，以及理解起来不会有什么困难的东西则会一笔带过，或者是完全不提及。如在介绍HttpServlet时对不常用的方法doHead()等就没有提及。否则，这本书的篇幅将会急剧膨胀，这就会与我的初衷相悖。

在介绍的这些框架的版本选择上，考虑到大多数读者从学到真正使用有一个过程，因此尽可能采用比较新的版本。这样，当他们真正使用的时候，这些版本就已经成为稳定版本了。为此，本人付出不少额外的劳动，原来本书是以JSF 1.1为基础来介绍的，在本书快要完稿之际，JDK 1.6及Tomcat 6的相继发布为采用JSF 1.2奠定了基础。因此，在修改稿件时，使本书的版本基本建立在JSF 1.2上，对绝大部分示例应用也用JSF 1.2重写了一遍。

欢迎提出反馈意见

尽管我费了很大的工夫来写本书，比如，大年初一就在准备书中的综合示例。但限于我的学识水平，又加上时间仓促，很多地方没有工夫来仔细推敲，书中可能有不少缺点和错误。尽管如此，如果本书能对初学者有些许帮助，我的心血就没有白费，我的内心也将会感到非常愉悦和满足。

幸好，人的学习过程正如范增先生的一幅画上的题字那样——“师古人亦师造化”。人们从书本上

学习知识时也不会拘泥于书本，还要向自然学习。对应用程序开发人员来说还要向社会学习。

对书中存在的缺点和错误，欢迎各位批评指正。人性中有一个弱点，就是喜欢听表扬的话，不喜欢听批评的话。闻过则喜可能只有理想中的圣人能做到，我当然远没有达到这种境界。但无论您如何评价本书，只要您说得有理有据，即使是批评，我也将真诚接受，因为这对于今后的工作与学习都是有利的。另一方面，我相信人的内心深处都会是向往真、善、美的。对“真”的追求应该是人生中的一个不变的追求。

如果您在阅读本书的过程中有任何的意见或建议，请给我发Email，我的Email地址是 lhbf@sina.com。

罗会波

2007年6月19日

目 录

第 1 章 为什么要学习框架及学习什么样的框架

1.1 Java 的发展与变化	1
1.1.1 版本与称谓的变化	1
1.1.2 应用程序开发方式的变迁	1
1.2 两种形式的多层结构的 J2EE 应用程序	2
1.2.1 胖客户端 J2EE 应用程序	2
1.2.2 Web 应用程序	3
1.3 Web 应用的两种实现方式	4
1.3.1 传统的重量级实现方式	5
1.3.1.1 运行环境	5
1.3.1.2 EJB	5
1.3.1.3 其他特点	6
1.3.1.4 适应的场合	6
1.3.2 方兴未艾的轻量级实现方式	6
1.3.2.1 运行环境	7
1.3.2.2 POJO	7
1.3.2.3 其他特点	7
1.3.2.4 适应场合	7
1.4 风云际会的框架世界	7
1.4.1 框架的出现与流行	8
1.4.2 大器晚成展新颜——JSF	8
1.4.3 层林深处好冬眠——Hibernate	9
1.4.4 七音合奏春光曲——Spring	9
1.5 为什么是 JSF 而不是 Struts	9
1.5.1 比 Struts 更高的抽象	9
1.5.2 早先暴露的缺陷不断被弥补	11
1.5.2.1 解决了与 JSP 集成中的一些突出问题	11
1.5.2.2 统一表达式语言与 JSTL 迭代标签的支持	11
1.5.2.3 其他新的特征	12
1.5.3 与客户端技术的完美结合	12
1.5.3.1 可以借助 JavaScript 改善用户体验	12

1.5.3.2 可与 AJAX 无缝集成	12
----------------------	----

第 2 章 JSF 与 Web 应用程序

2.1 JSF 的设计目标及与 Web 应用的关系	13
2.2 Web 应用程序概览	13
2.2.1 基本的 Web 组件	13
2.2.2 Web 部署描述器文件	13
2.2.3 Web 应用程序的物理结构	14
2.3 Web 应用程序的开发环境配置	14
2.3.1 Java 2 SDK 的下载与安装	14
2.3.2 Web 容器的下载与安装	15
2.3.3 环境变量的设置	16
2.3.4 其他准备工作	16
2.4 Servlet	18
2.4.1 Servlet 的基本概念	18
2.4.2 Servlet 的体系结构	18
2.4.3 你的第一个 Servlet 应用程序——Hello World	19
2.4.3.1 建立示例应用目录结构	19
2.4.3.2 编写 Servlet 类文件	19
2.4.3.3 编译 Servlet 类文件	20
2.4.3.4 编写部署描述器文件 web.xml	20
2.4.3.5 部署与运行示例应用	21
2.4.3.6 原理分析	22
2.4.4 Servlet 的生命周期	22
2.4.4.1 初始化	23
2.4.4.2 处理请求	23
2.4.4.3 销毁	24
2.4.5 Servlet 的作用域对象	24
2.4.6 过滤请求与响应	24
2.5 JSP	24
2.5.1 JSP 如何表示静态内容	25
2.5.2 JSP 如何表示动态内容	25

2.5.3 JSP 的注释	25	2.5.11.4 一个简单的 JavaBean	48
2.5.3.1 HTML 注释	26	2.5.11.5 在 JSP 中使用 JavaBean	50
2.5.3.2 JSP 注释	26	2.5.12 JavaBean 在 JSP 中的有效范围	50
2.5.4 JSP 示例	26	2.5.12.1 演示 Page 范围的 Bean	50
2.5.4.1 建立应用程序目录	26	2.5.12.2 演示 Request 范围的 Bean	52
2.5.4.2 编写 JSP 示例代码	26	2.5.12.3 演示 Session 范围的 Bean	53
2.5.4.3 部署和运行 JSP	26	2.5.12.4 演示 Application 范围的 Bean	54
2.5.5 JSP 与 Servlet 的关系	27	2.6 早期框架 Struts	55
2.5.5.1 JSP 转换为 Servlet 后执行	27	2.6.1 应用程序的分层	55
2.5.5.2 JSP 转化后的 Servlet 的 简要分析	27	2.6.2 表单的处理与验证	55
2.5.5.3 JSP 为 Servlet 提供了模板 机制	29	2.6.3 类型转换	56
2.5.6 JSP 的元素	29	2.6.4 导航	56
2.5.6.1 指令	29	2.6.5 资源管理	56
2.5.6.2 动作	30	2.6.6 国际化编程	56
2.5.6.3 脚本元素	30	2.6.7 异常处理的加强	56
2.5.7 JSP 的标签库	32	2.6.8 模板的重用管理与布局	57
2.5.7.1 标签库的组成	32	2.7 JSF	57
2.5.7.2 自定义标签库	33	2.7.1 具有早期框架的一些特性	57
2.5.7.3 JSP 标准标签库	35	2.7.2 有状态的用户界面组件模型	57
2.5.8 JSP 的内建对象	35	2.7.3 用户界面事件的服务器端的处理	58
2.5.8.1 out	36	2.7.4 自动生成各种标志语言	58
2.5.8.2 request	37	2.7.5 与业务对象的声明式集成	58
2.5.8.3 session	39	第 3 章 JSF 应用程序开发入门	59
2.5.8.4 application	41	3.1 JSF 开发环境的配置	59
2.5.8.5 config	41	3.1.1 JSF 参考实现的下载与安装	59
2.5.8.6 pageContext	41	3.1.2 ANT 的下载与安装	60
2.5.8.7 page	41	3.1.3 环境变量的设置	60
2.5.8.8 response	41	3.2 开发您的第一个 JSF 应用程序	60
2.5.8.9 exception	41	3.2.1 Hello World 应用程序的需求分析	61
2.5.9 JSP 中的异常处理	42	3.2.2 应用程序的目录结构	61
2.5.10 Web 组件间的关联关系	42	3.2.3 应用程序的页面文件	61
2.5.10.1 包含	42	3.2.3.1 index.jsp	61
2.5.10.2 请求转发	43	3.2.3.2 hello.jsp	62
2.5.10.3 重定向	45	3.2.3.3 howDoYouDo.jsp	65
2.5.11 JavaBean	45	3.2.3.4 howAreYou.jsp	66
2.5.11.1 JavaBean 的概念	45	3.2.4 应用程序的 Backing Bean	66
2.5.11.2 为什么要引入 JavaBean	46	3.2.4.1 Backing Bean 及其作用	66
2.5.11.3 JavaBean 的编码约定	46	3.2.4.2 示例中的 Backing Bean 类文件	67

3.2.4.3 Backing Bean 的特点	69	4.1.5.3 调用请求处理生命周期的 render()方法	82
3.2.5 JSF 的配置文件 faces-config.xml	69	4.1.5.4 释放 Faces 上下文对象	83
3.2.5.1 示例应用的导航规则	70	4.1.5.5 与普通 Servlet 对请求处理 的比较	83
3.5.5.2 示例应用的托管 JavaBean 配置	71	4.1.6 处理完成后的资源释放	83
3.2.6 应用程序的配置文件 web.xml	71	4.2 Faces 上下文	83
3.2.7 手工编译应用程序	72	4.2.1 Faces 上下文简介	83
3.2.8 应用程序上下文	73	4.2.2 Faces 上下文的主要方法	84
3.2.9 建立应用程序上下文及部署应用 程序	73	4.2.2.1 获取应用程序实例	84
3.2.9.1 直接修改 Tomcat 的配置 文件 server.xml	73	4.2.2.2 获取外部上下文	84
3.2.9.2 使用 Tomcat 页面形式的 管理工具	73	4.2.2.3 获取与设置视图根	85
3.2.10 运行应用程序	74	4.2.2.4 处理消息队列	85
3.3 用 ANT 工具构建应用程序	75	4.2.2.5 获取呈现器工具包	86
3.3.1 ANT 工具简介	75	4.2.2.6 流程控制方法	86
3.3.2 创建 build.xml 文件	76	4.2.2.7 获取表达式语言上下文	87
3.3.2.1 设置属性	76	4.2.2.8 响应	88
3.3.2.2 设置 classpath	77	4.2.2.9 获取 Faces 上下文的当前 实例	88
3.3.2.3 定义 clean 目标	77	4.3 请求处理的生命周期	88
3.3.2.4 定义 prepare 目标	77	4.3.1 获得生命周期对象	88
3.3.2.5 定义 deploy.copyJars 目标	77	4.3.2 生命周期的场景	89
3.3.2.6 定义 build 目标	78	4.3.2.1 非 Faces 请求生成 Faces 响应	89
3.3.2.7 定义 build.war 目标	79	4.3.2.2 Faces 请求生成非 Faces 响应	90
3.3.2.8 定义 deploy 目标	79	4.3.2.3 Faces 请求生成 Faces 响应	90
3.3.3 用 ANT 工具自动构建和部署应用 程序	79	4.3.3 标准的 JSF 请求处理生命周期	90
第 4 章 JSF 中的请求处理	80	4.3.3.1 重建视图	92
4.1 JSF 处理请求的基本过程	80	4.3.3.2 应用请求值	93
4.1.1 将请求映射到 FacesServlet	80	4.3.3.3 处理验证	94
4.1.2 FacesServlet 类	80	4.3.3.4 更新模型值	95
4.1.3 获得配置 Servlet 的配置信息	81	4.3.3.5 调用应用程序	95
4.1.4 初始化时获得 Faces 上下文工厂和 生命周期实例	81	4.3.3.6 呈现响应	95
4.1.5 FacesServlet 对请求的处理过程	81	4.4 Application 及相关的类	96
4.1.5.1 获得 Faces 上下文并用它 处理当前请求	82	4.4.1 Application 简介	96
4.1.5.2 调用请求处理生命周期的 execute()方法	82	4.4.2 Application 的主要属性	97
		4.4.2.1 动作监听器	97
		4.4.2.2 导航处理器	98
		4.4.2.3 视图处理器	98

4.4.2.4 缺省呈现包标识	98	5.3.8 验证器	120
4.4.2.5 状态管理器	98	5.3.9 消息	120
4.4.2.6 表达式语言解析器	98	第 6 章 标准用户界面组件	122
4.4.3 Application 动作	98	6.1 标准组件概述	122
4.4.4 Application 作为组件工厂	99	6.1.1 组件的类型	122
4.4.5 Application 支持国际化编程的方法	99	6.1.2 组件的家族	122
第 5 章 JSF 的体系结构	101	6.1.3 组件的呈现器类型	123
5.1 MVC 架构模式	101	6.2 JSF 的呈现模型	123
5.1.1 耦合与解耦	101	6.2.1 直接实现方式	123
5.1.2 MVC 架构模式	102	6.2.2 委托实现方式	123
5.1.3 Web 应用中的 MVC——Model2	104	6.2.3 呈现包工厂	124
5.2 MVC 架构模式的实现策略	105	6.2.4 呈现包	124
5.2.1 Struts 的实现策略	105	6.2.5 呈现器	125
5.2.2 JSF 的实现策略	105	6.3 JSF 实现的缺省呈现器类型与标准组件	
5.2.2.1 JSF 的控制器部分	106	HTML 具体类的关系	126
5.2.2.2 JSF 的视图部分	107	6.4 组件在 JSP 页面上的表示	127
5.2.2.3 JSF 的模型部分	107	6.4.1 在 JSP 页面中使用用户界面组件的	
5.3 JSF 的主要组成部分	108	定制动作	128
5.3.1 JSF 主要组成部分及其关系	108	6.4.2 核心标签库	128
5.3.1.1 组成部分概览	108	6.4.3 HTML 标签库	128
5.3.1.2 各部分之间的关系	109	6.4.4 组件的 HTML 属性	130
5.3.2 用户界面组件	109	6.5 标准组件数据模型	130
5.3.2.1 对组件的直观认识	109	6.5.1 标准组件与呈现无关的属性	130
5.3.2.2 JSF 的组件模型	109	6.5.2 标准组件模型 Bean	131
5.3.2.3 组件模型中的接口	110	6.5.2.1 SelectItem	131
5.3.2.4 组件模型中的类	111	6.5.2.2 SelectItemGroup	131
5.3.2.5 组件的作用	112	6.5.2.3 DataModel	131
5.3.2.6 组件的属性	112	6.6 标准组件及使用方法	132
5.3.2.7 组件在服务器和客户端的		6.6.1 用 UIViewRoot 控制页面	132
表现形式	113	6.6.1.1 UIViewRoot 简介	132
5.3.2.8 组件的标识	114	6.6.1.2 <f:view>在 JSP 页面上	
5.3.2.9 通过标识引用组件	115	的应用	133
5.3.2.10 组件是有状态的	116	6.6.1.3 <f:subview>的使用	134
5.3.2.11 标准组件与自定义组件	116	6.6.2 用 UIForm 处理表单	136
5.3.3 呈现器	116	6.6.3 用 UIParameter 传递参数	137
5.3.4 Backing Bean	117	6.6.4 用 Facet 制作表头、表脚及列头	137
5.3.5 事件与事件监听器	118	6.6.5 用 UIInput 处理输入	138
5.3.6 导航系统	119	6.6.5.1 用 HtmlInputText 输入单行	
5.3.7 类型转换器	120	文本	139

6.6.5.2 用 <code>HtmlInputSecret</code> 输入 密码.....	139	6.6.15 用 <code>UIGraphic</code> 显示图片.....	148
6.6.5.3 用 <code>HtmlInputTextarea</code> 输入 多行文本.....	139	6.6.16 用 <code>UIPanel</code> 分组与布局组件.....	148
6.6.5.4 用 <code>HtmlInputHidden</code> 作为隐含字段.....	140	6.6.16.1 用 <code>HtmlPanelGroup</code> 分组 组件.....	148
6.6.6 用 <code>UIOutput</code> 处理输出.....	140	6.6.16.2 用 <code>HtmlPanelGrid</code> 布局 组件.....	148
6.6.6.1 用 <code>HtmlOutputText</code> 输出简单文本.....	140	6.6.17 用 <code>UIColumn</code> 表示数据列.....	149
6.6.6.2 用 <code>HtmlOutputFormat</code> 输出格式化文本.....	141	6.6.18 用 <code>UIData</code> 显示数据集中的数据.....	149
6.6.6.3 用 <code>HtmlOutputLabel</code> 作为 组件的标注.....	142	6.6.19 到哪里获得组件更详细的信息.....	150
6.6.6.4 用 <code>HtmlOutputLink</code> 作为 超链接.....	142	第 7 章 表达式语言与托管 Bean	151
6.6.7 用 <code>UISelectedItem</code> 提供单个选项.....	142	7.1 表达式语言的出现.....	151
6.6.8 用 <code>UISelectItems</code> 提供多个选项.....	143	7.1.1 统一表达式语言出现的背景.....	151
6.6.9 用 <code>UISelectBoolean</code> 表示逻辑选项.....	143	7.1.1.1 JSP 的表达式语言.....	151
6.6.10 用 <code>UISelectOne</code> 作为单选组件.....	144	7.1.1.2 JSF 自己的表达式语言.....	152
6.6.10.1 单选列表框 <code>HtmlSelectOneListbox</code>	144	7.1.1.3 JSP 与 JSF 表达式语言的 冲突.....	153
6.6.10.2 单选下拉菜单 <code>HtmlSelectOneMenu</code>	144	7.1.2 统一表达式语言.....	153
6.6.10.3 单选钮 <code>HtmlSelect- OneRadio</code>	144	7.1.2.1 值表达式.....	153
6.6.11 用 <code>UISelectMany</code> 作为复选组件.....	145	7.1.2.2 方法表达式.....	154
6.6.11.1 复选列表框 <code>HtmlSelectManyListbox</code>	145	7.1.2.3 解决原来的 JSP 与 JSF 表达式的冲突问题.....	154
6.6.11.2 复选下拉菜单 <code>HtmlSelectManyMenu</code>	145	7.2 托管 Bean.....	155
6.6.11.3 复选钮 <code>HtmlSelectManyCheckbox</code>	145	7.2.1 托管 Bean 创建工具与托管 Bean.....	155
6.6.12 用 <code>UICommand</code> 调用应用程序.....	146	7.2.2 托管 Bean 的配置.....	155
6.6.12.1 命令按钮 <code>HtmlCommand Button</code>	146	7.2.2.1 基本的配置元素.....	156
6.6.12.2 命令超链接 <code>HtmlCommandLink</code>	146	7.2.2.2 配置托管 Bean 的属性.....	156
6.6.13 用 <code>UIMessage</code> 显示特定组件的 消息.....	147	7.2.2.3 托管 Bean 属性的初始化与 JSF 的 IoC.....	157
6.6.14 用 <code>UIMessages</code> 显示多条消息.....	147	7.2.3 List 配置为托管 Bean.....	158
		7.2.4 Map 配置为托管 Bean.....	158
		7.3 托管 Bean 的有效范围.....	159
		7.3.1 请求范围.....	159
		7.3.2 会话范围.....	160
		7.3.3 应用程序范围.....	160
		7.3.4 NONE 范围.....	160
		7.3.5 不同范围间的托管 Bean 的引用.....	161
		第 8 章 JSF 中的导航	163
		8.1 JSF 导航概述.....	163

8.1.1 应用程序动作与导航	163	9.2.8 在 Faces 配置文件中配置导航 规则	180
8.1.2 导航处理器的行为是可以配置的	163	9.2.9 在 Faces 配置文件中配置托管 Bean	180
8.2 JSF 支持的导航形式	164	9.2.10 部署和运行示例应用	181
8.2.1 动态导航	164	9.3 JSF 中的控制反转 (IoC)	182
8.2.1.1 何为动态导航	164	9.3.1 登录示例应用的不足	182
8.2.1.2 简单动态导航示例回顾	164	9.3.2 控制反转 (IoC) 及依赖注入 (DI)	182
8.2.2 静态导航	165	9.3.2.1 控制反转及依赖注入简介	182
8.2.2.1 何为静态导航	165	9.3.2.2 依赖注入的形式	183
8.2.2.2 静态导航示例	165	9.4 在 JSF 应用程序中使用依赖注入	185
8.3 导航规则	167	9.4.1 在 Backing Bean 使用 type2 型 依赖注入	185
8.3.1 导航规则需要配置的元素	167	9.4.2 在 Faces 配置文件中配置依赖关系	186
8.3.2 导航规则的匹配算法	167	9.4.3 部署和运行示例应用	187
8.3.3 复杂导航规则的配置示例	168		
8.4 动作事件监听器方法对导航的影响	169	第 10 章 JSF 的事件处理	188
8.4.1 动作事件监听器方法间接影响 导航	169	10.1 JSF 事件及事件监听器模型	188
8.4.2 动作事件监听器方法对导航 影响的示例	169	10.1.1 事件类	188
8.4.2.1 导航出发页面	169	10.1.2 监听器类	189
8.4.2.2 包含动作事件监听器方法和 动作方法的 Backing Bean	170	10.1.3 事件广播及处理	189
8.4.2.3 导航目的地页面	171	10.2 JSF 中的事件分类	190
8.4.2.4 配置导航规则	172	10.2.1 值改变事件	190
8.4.2.5 部署和运行示例应用程序	172	10.2.2 动作事件	190
		10.2.3 阶段事件	190
第 9 章 JSF 与应用程序模型层的集成	174	10.3 事件处理示例	190
9.1 JSF 作为表现层框架	174	10.3.1 值改变事件的处理示例	190
9.1.1 JSF 结构回顾	174	10.3.1.1 示例应用的页面文件	191
9.1.2 Backing Bean 扮演的角色	174	10.3.1.2 在 Backing Bean 中编写处理 值改变事件方法	192
9.1.3 编写 Backing Bean 代码时的 不良实践	174	10.3.1.3 在 Faces 配置文件中配置 Backing Bean	192
9.2 登录示例应用程序	175	10.3.1.4 部署和运行	192
9.2.1 示例登录应用的需求分析	175	10.3.1.5 用专门的监听器来处理值改 变事件	193
9.2.2 登录示例应用的界面	175	10.3.2 动作事件的处理示例	194
9.2.3 目录结构	176	10.3.2.1 在页面中使用命令超链接	194
9.2.4 业务服务接口	176	10.3.2.2 在 Backing Bean 中编写事 件处理方法	195
9.2.5 业务服务实现	176		
9.2.6 示例应用的 Backing Bean	178		
9.2.7 示例中的页面文件	179		

10.3.2.3 部署和运行示例应用	196	11.7.2 新建一个可以输入日期的页面	213
10.3.3 Immediate 组件使用示例	197	11.7.3 在页面上使用显示日期的组件	213
10.3.3.1 在页面上使用 Immediate 组件	197	11.7.4 Backing Bean 中的日期型属性	213
10.3.3.2 在 Backing Bean 中编写 事件监听器方法	199	11.7.5 业务对象中的日期属性	214
10.3.3.3 观察示例应用的运行效果	199	11.7.6 业务服务接口中的日期型参数	215
第 11 章 JSF 中的转换	201	11.7.7 在业务服务实现中处理日期型 数据	215
11.1 JSF 的转换模型	201	11.7.8 更改示例应用的 Faces 配置	216
11.1.1 什么是 JSF 中的转换	201	11.7.9 日期转换的效果	217
11.1.2 JSF 中为什么需要转换	201	11.8 数字转换器	218
11.1.3 JSF 中何时发生转换	202	11.9 使用自定义转换器	218
11.1.4 JSF 的转换器接口	202	11.9.1 创建自定义转换器	218
11.2 使用 JSF 的标准转换器	203	11.9.2 注册自定义转换器	220
11.2.1 JSF 的标准转换器	203	11.10 自定义的转换器示例	220
11.2.2 使用标准转换器的一般方式	203	11.10.1 建立示例应用的目录结构	220
11.3 自动转换方式示例	204	11.10.2 示例应用的 Backing Bean	220
11.3.1 在注册页面增加一个表示年龄 的文本输入框	204	11.10.3 新建自定义格式的业务对象	222
11.3.2 在注册成功的页面上显示年龄	204	11.10.4 修改原业务对象	223
11.3.3 Backing Bean 年龄属性的数据 类型为 int	204	11.10.5 在业务服务接口的方法中 加入新参数	223
11.3.4 示例应用的业务数据对象	205	11.10.6 更改业务服务实现类	224
11.3.5 示例应用中的自定义异常	206	11.10.7 在 Faces 中配置自定义转换器	225
11.3.6 示例应用的业务服务	206	11.10.8 在用户注册页面上使用自定义 转换器	225
11.3.7 示例应用的 Faces 配置	206	11.10.9 在注册成功页面上使用自定义 转换器	226
11.3.8 自动转换的效果	207	11.10.10 检验自定义转换器的使用 效果	226
11.3.8.1 自动转换失败的效果	207	第 12 章 JSF 中的验证	227
11.3.8.2 自动转换成功的效果	207	12.1 JSF 的验证模型	227
11.4 使用组件标签的属性	208	12.1.1 验证器类	227
11.5 使用转换器标签	208	12.1.2 验证器的注册	227
11.6 Java 中的日期与时间	209	12.1.3 验证处理过程	228
11.6.1 Java 中的 Date 类	209	12.1.4 标准验证器实现	228
11.6.2 Calendar 类	210	12.2 使用标准验证器示例	229
11.6.3 GregorianCalendar 类	210	12.2.1 建立示例应用目录结构	229
11.6.4 日期、时间的格式化	211	12.2.2 在页面中使用标准验证器	229
11.6.5 访问数据库时使用的日期类	211	12.2.3 部署和运行应用程序	230
11.7 日期时间转换器示例	212	12.3 正则表达式简介	231
11.7.1 建立示例应用的目录结构	212		

12.4 使用自定义验证器示例.....	233	13.7.2 制作缺省资源文件.....	255
12.4.1 建立示例应用目录.....	233	13.7.3 改编页面文件.....	256
12.4.2 创建自定义验证器类.....	234	13.7.4 编写和配置 Servlet 过滤器.....	258
12.4.3 在 Faces 配置文件中注册验证器.....	235	13.7.5 构建和运行应用程序.....	259
12.4.4 在页面中使用自定义验证器.....	235	13.7.6 编写中文资源文件.....	259
12.4.5 自定义验证器的使用效果.....	235	13.7.7 检验应用程序的运行效果.....	260
12.5 使用 Backing Bean 的验证方法.....	237	13.8 图片的国际化.....	261
12.5.1 建立示例应用目录结构.....	237	13.8.1 在项目中建立存放图片的目录.....	262
12.5.2 在 Backing Bean 中编写验证 方法.....	237	13.8.2 制作对应各种不同语言的图片.....	262
12.5.3 在 Faces 配置中删除自定义验 证器.....	238	13.8.3 在页面文件上添加图片.....	262
12.5.4 在页面中引用验证方法.....	238	13.8.4 在资源束文件中加上对应图片 键-值对.....	263
12.5.5 运行应用程序查看验证效果.....	239	13.8.5 在 build.xml 中添加拷贝图片的 内容.....	263
第 13 章 JSF 的国际化编程.....	240	13.8.6 观察应用程序的运行结果.....	264
13.1 问题的由来.....	240	13.9 异常消息的国际化.....	264
13.2 字符集的基础知识.....	240	13.9.1 异常消息的种类.....	265
13.2.1 计算机表示字符的方式与字 符集.....	240	13.9.2 标准异常消息.....	265
13.2.2 字符集的发展.....	241	13.9.3 自定义异常消息的处理过程.....	266
13.2.3 UTF-8 字符集.....	242	13.10 异常消息国际化示例.....	266
13.3 Java 及 Web 中的编码.....	243	13.10.1 在资源文件中加上消息的 键-值对.....	266
13.3.1 Java 中的编码.....	243	13.10.2 创建消息工厂.....	267
13.3.2 Web 中的编码.....	243	13.10.3 业务模型中的消息处理方式.....	267
13.4 中文乱码的几种表现形式及成因浅析.....	244	13.10.4 Backing Bean 中的消息处理 方式.....	267
13.4.1 字符变问号.....	244	13.10.5 示例应用的运行效果.....	269
13.4.2 部分汉字是乱码.....	247	13.11 日期的国际化示例.....	269
13.4.3 显示乱码（有些是汉字但并 不是你预期的）.....	248	13.11.1 示例应用要实现的效果.....	269
13.4.4 读写数据库时出现乱码.....	250	13.11.2 建立示例应用的目录结构.....	269
13.5 JSF 国际化编程的思路.....	250	13.11.3 修改页面文件.....	270
13.6 JSF 国际化编程的实现.....	251	13.11.4 准备资源文件.....	270
13.6.1 支持国际化编程的应用的基本 特征.....	251	13.11.5 修改业务对象.....	271
13.6.2 Java 平台的本地化类.....	251	13.11.6 修改业务模型中的类文件.....	271
13.6.3 资源束.....	252	13.11.7 修改 Backing Bean.....	272
13.6.4 日期与数字的格式化.....	253	13.11.8 国际化编程后的效果检验.....	274
13.7 登录示例程序的国际化与本地化.....	254	第 14 章 JSF 与 Tiles 框架的集成.....	275
13.7.1 收集区域敏感数据.....	255	14.1 页面布局的几种可选方案.....	275
		14.1.1 使用 JSP 指令的方法.....	275

14.1.2 使用 JSP 标签	279	15.2.6.3 数据源层	302
14.1.3 使用 Tiles 框架	279	15.3 数据访问对象模式简介	302
14.2 Tiles 框架简介	280	15.4 让登录程序访问数据库中用户信息	302
14.3 用 Tiles 布局页面的示例	280	15.4.1 建立数据库	302
14.3.1 建立示例应用的目录结构	280	15.4.2 建立数据库表	303
14.3.2 下载和安装与 Tiles 有关的库 文件	280	15.4.3 在表中输入实验用的数据	304
14.3.3 编写 tiles 的配置文件	281	15.4.4 下载和安装 JDBC 驱动程序	304
14.3.4 在 web.xml 文件中配置 tiles	281	15.4.5 建立应用程序的目录结构	305
14.3.5 编写布局模板文件	282	15.4.6 增加应用程序访问数据库的 部分	305
14.3.6 编写布局的各组成部分的页面 文件	283	15.4.7 修改应用程序的业务逻辑部分	307
14.3.7 编写应用模板的页面文件	284	15.4.8 修改 backing bean 使其使用 新的 service	308
14.3.8 示例应用的运行效果	284	15.4.9 创建与数据访问相关的自定义 异常类	309
14.4 用 Tiles 布局 Frame 页面	285	15.4.10 修改资源文件, 添加新的 键-值对	311
14.4.1 Frame 页面的基础知识	285	15.4.11 配置应用程序	311
14.4.2 一个简单的 Frame 页面例子	286	15.4.12 运行应用程序	313
14.4.3 让 Frame 页面居中显示	288		
14.5 用 Tiles 布局 Frame 页面的示例	289	第 16 章 应用程序中的异常处理	315
14.5.1 建立示例应用的目录结构	289	16.1 Java 异常的分类	315
14.5.2 新建图片文件	289	16.2 Java 处理异常的一般方法	317
14.5.3 更改 tiles 配置	289	16.3 创建自己的异常子类	320
14.5.4 新建 frame 布局模板页面	289	16.4 成链异常	321
14.5.5 新建用于导航的菜单页面	290	16.5 JSF 应用程序的异常处理	322
14.5.6 新建带 frame 的页面	291		
14.5.7 应用程序的运行效果	291	第 17 章 应用程序的日志	324
第 15 章 JSF 访问数据库	293	17.1 为什么要记日志	324
15.1 MS SQL Server 数据库的安装	293	17.2 用什么记日志	324
15.1.1 SQL Server 数据库的安装过程	293	17.3 怎么记日志	324
15.1.2 数据库补丁包的安装	296	17.4 在登录示例程序中记日志	326
15.2 JDBC 简介	297	17.4.1 建立示例应用目录结构	326
15.2.1 JDBC 的发展历程	297	17.4.2 下载通用日志包	326
15.2.2 JDBC 的结构和工作原理	298	17.4.3 下载日志器实现 Log4J	327
15.2.3 JDBC 驱动器	300	17.4.4 写日志有关的配置文件	327
15.2.4 JDBC 的使用场景	300	17.4.5 创建一个空的日志文件	329
15.2.5 两层模式	300	17.4.6 编写访问日志接口的应用程序 代码	329
15.2.6 三层模式	301	17.4.7 部署和运行应用程序	331
15.2.6.1 客户层	301		
15.2.6.2 中间层	301		