

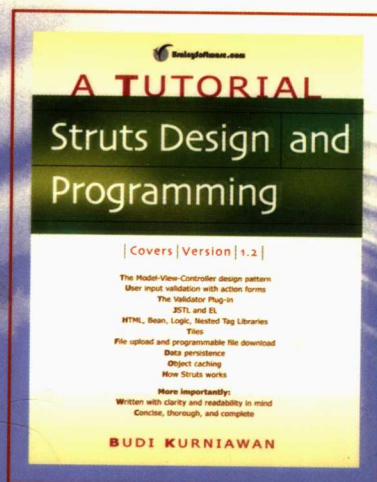
Struts Design and Programming: A Tutorial

深入浅出 Struts

[加] Budi Kurniawan 著

王建桥 杨晓云 韩 兰 等译

- Amazon 畅销书
- 深入全面地阐释 Struts 的方方面面
- 专门章节剖析 Struts 源代码



图书在版编目 (CIP) 数据

深入浅出 Struts / (加) 库尼亚瓦姆 (Kurniawan, B.) 著; 王建桥等译. —北京: 人民邮电出版社, 2007.9

(图灵程序设计丛书)

ISBN 978-7-115-16407-0

I . 深... II . ①库... ②王... III . 软件工具-程序设计 IV . TP311.56

中国版本图书馆 CIP 数据核字 (2007) 第 088600 号

内 容 提 要

Struts 是 Java Web 应用首选的 MVC 框架。本书是世界畅销的 Struts 著作之一, 深入浅出地探讨了许多能帮助程序员们编写高效 Struts 应用程序的技巧, 主要内容包括利用动作表单进行输入验证、HTML 标签库、输入验证和数据转换、Validator 插件、表达式语言、JSTL 库、Bean 标签库、Logic 标签库、消息处理与国际化、Tiles 框架等 Struts 功能, 最后还直接剖析源代码, 深入阐述了 Struts 的工作机理。书中概念清晰, 环环相扣, 便于读者学习。

本书适合希望使用 Struts 开发应用程序的读者阅读。

图灵程序设计丛书

深入浅出 Struts

-
- ◆ 著 [加] Budi Kurniawan
 - 译 王建桥 杨晓云 韩兰 等
 - 责任编辑 傅志红
 - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
三河市海波印务有限公司印刷
新华书店总店北京发行所经销
 - ◆ 开本: 800 × 1000 1/16
印张: 21.5
字数: 503 千字 2007 年 9 月第 1 版
印数: 1—4 000 册 2007 年 9 月河北第 1 次印刷

著作权合同登记号 图字: 01-2006-5782 号

ISBN 978-7-115-16407-0/TP

定价: 45.00 元

读者服务热线: (010)88593802 印装质量热线: (010)67129223

版 权 声 明

Original English language edition, entitled *Struts Design and Programming: A Tutorial* by Budi Kurniawan.

Copyright © 2005 by Budi Kurniawan. Simplified Chinese-language edition copyright © 2007 by Posts & Telecom Press. All rights reserved.

本书中文简体字版由 Apress L.P. 授权人民邮电出版社独家出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，侵权必究。

前言

Servlet¹技术和JSP（JavaServer Pages）是利用Java语言开发Web应用程序的两种主要技术。Sun公司于1996年首次推出Servlet技术的时候，人们认为这种技术远优于当时占主导地位的公共网关接口（Common Gateway Interface, CGI）。这是因为某个servlet（服务器端Java程序）在应用户的请求而首次调入内存执行之后将一直驻留在内存里，对同一个servlet的后续请求不用再对这个servlet的类进行实例化，因此响应时间更快。

可是，servlet也存在一个严重问题，因为所有的HTML输出必须像下面这段代码那样封装在String对象里，所以servlet将HTML标签发送给浏览器时既繁琐又容易出错：

```
PrintWriter out = response.getWriter();
out.println("<html><head><title>Testing</title></head>");
out.println("<body bgcolor=\"\#ffdddd\">");
...
```

这大大增加了编程的难度，而且即便是网页在表现方面的细微改动，例如改变网页的背景颜色，也需要重新编译整个servlet。

Sun公司意识到了这个问题，并提出了JSP技术。JSP允许Java代码和HTML标签混杂在一起以简化页面的编辑工作，所有的改动无需重新进行编译：某个页面修改后，将在第一次被调用时自动编译。JSP里的Java代码称为scriptlet²。

允许Java代码和HTML混杂在一起的办法乍看起来很实用，但实际上并不好。正是因为意识到了这一点，JSP技术的设计者允许开发人员把有关的业务逻辑放到相应的Java类里，并从JSP页面去调用这些类里的方法。把Java代码和HTML混杂在一起的做法普遍认为不好，理由如下。

- ❑ 把Java代码和HTML混杂在一起会让应用程序变得难以阅读和维护。
- ❑ 把代码编写在JSP页面里的做法使代码很难再重用。当然，你完全可以把所有的Java方法都放到一个JSP页面里，并把这个页面包含到需要用到这些方法的其他JSP页面里，但这种做法显然偏离了面向对象范型。别的先不说，至少继承机制的好处是无从谈起了。
- ❑ 在JSP页面里编写Java代码要比在Java类里编写Java代码困难得多。我们必须面对这样一个

1. Servlet首字母大写是指这种技术的概念和API，小写时指某个具体的servlet类、对象等。——编者注

2. 直译为“小脚本”。——编者注

事实：各种各样的IDE都是为了分析Java类里的Java代码而设计的。

- ❑ 如果把代码封装在Java类里，调试起来会更容易。
- ❑ 封装在Java类里的业务逻辑更容易测试。
- ❑ Java类里的Java代码更容易重构。

事实上，把业务逻辑（Java代码）和页面表现（HTML标签）分开是非常重要的，JSP的设计者从JSP的第一个版本开始就一直在鼓励程序员采用这种做法。

在JSP 1.0里，JSP的设计者向程序员提供了使用JavaBeans¹技术来封装Java代码的机制，其目的就是为了能够让程序员能够把有关代码和页面表现分离开来，并最大限度地提高那些代码的可重用性。从本质上讲，JavaBean就是一个Java类，程序员可以通过<jsp:useBean>和<jsp:setProperty>来对它进行操作。

令人遗憾的是，JavaBeans技术并没有为代码和页面表现的分离问题提供一个完美的解决方案。首先，在创建JavaBean的时候，对方法的命名必须遵守JavaBean的命名约定，所以偶尔会出现某个方法的名字非常冗长难记的情况。其次，在使用JavaBean的时候，如果不求助于scriptlet，我们就无法把有关的参数传递给相应的方法。

为了使代码和HTML标签分离更容易实现，JSP 1.1定义了几个自定义标签（custom tag）库，比JavaBeans更加灵活易用的。但这又引出了一个新问题：自定义标签很难编写，而JSP 1.1中的自定义标签又都有非常复杂的生命期。

为了减轻那些想把代码和页面表现分离开来的开发人员们编写自定义标签的工作量，人们开始给有关的标签添加一些特定的常用功能。这些标签编译为几个库文件，这些库文件统称为JSTL（JavaServer Pages Standard Tag Libraries，JSP标准标签库）。比如说，JSTL提供了一些可以用来处理各种作用域对象（scoped object）、遍历集合、进行条件测试、对数据进行解析和格式化等的标签。

尽管有了JavaBeans、自定义标签和JSTL等多种选择，还是有不少人因为以下几个理由仍在JSP页面里使用scriptlet。

- ❑ 贪图方便。把所有的东西都集中放在JSP页面里非常方便。如果应用程序是只包含一两个页面并且不可能再变得更加复杂的小型应用程序的话，这种做法还是很有诱惑力的。
- ❑ 目光短浅。乍看起来，把代码和HTML集中编写在JSP页面里似乎是一种能够缩短软件开发周期的好办法。但从长远看，采用这一思路开发软件是要付出代价的。代码的维护工作和可读性是两个主要的问题。
- ❑ 缺乏知识。

此外，如果参与项目开发工作的程序员水平参差不齐，想把全部的Java代码都井井有条地收

1. JavaBeans（复数形式）是指技术概念或API，它同时也是Sun公司的一个商标，JavaBean（单数形式）则是指某个具体的JavaBean类、对象等。——编者注

录到各有关Java类里将更加难以保证。要知道，即使是一位天才的架构师也无法做到对其团队里的每一个程序员的具体工作情况都了如指掌。

JSP 2.0增加了一项能够让软件架构师在JSP页面里统一禁用scriptlet的功能，而这意味着架构师可以强制地要求程序员必须把代码和HTML分开，从而更容易达到不在JSP页面里掺杂scriptlet的目的。换句话说，只要架构师有足够的知识和经验，就完全可以做到让开发人员必须借助JavaBeans、自定义标签、JSTL和其他开发库等实现必要的业务逻辑。在此基础上，为了简化自定义标签本身的编写工作，JSP 2.0还提供了一个更简单的自定义标签并支持把标签单独保存在标签文件里进行编译的做法。

为什么说 Servlet 还有生命力

随着JSP的到来，有许多人认为Servlet已行将就木，可实际情况并非如此。JSP没能完全取代Servlet。事实上，许多新近推出的应用程序都同时使用了Servlet和JSP。Servlet在JSP出现之后仍有生命力的原因是什么？要想了解其中的奥妙，必须先把用Java语言开发Web应用程序时的两种设计模型弄清楚。

第一种设计模型简称为Model 1，它紧随着JSP的出现而诞生。Servlet通常不用在这种模型中，从一个JSP页面浏览到另一个页面是通过点击该页面里的链接做到的。第二种设计模型简称为Model 2。下面马上就会介绍为什么Model 1不值得推荐，以及为什么Model 2才是应该遵循的设计思路。

Model 1 设计模型的弊病

Model 1设计模型是以页面为中心的。按照这一模型实现的应用程序由一系列JSP页面构成，用户就在这些页面间转来转去。这种模型既易于理解又易于实现，所以很多程序员在刚开始学JSP时都会采用它。按Model 1模型开发的应用程序的主要弊病是，它们既难以维护又很不灵活。这种架构不利于网页设计师和Web开发人员之间的劳动分工：开发人员既要参与页面的开发，又要参与业务逻辑的编码。

Model 1设计模型不值得推荐的原因主要有以下几点。

- ❑ 浏览问题。如果你打算改变某个JSP页面的名字，就必须在所有引用了这个页面的其他页面里把这个页面的名字全都改过来。
- ❑ 在这种模型下，因为JavaBeans的能力有限，而自定义标签又比较难以编写，所以在JSP页面里嵌入scriptlet的做法对程序员更有诱惑力。但正如我们刚才解释的那样，让Java代码和HTML混杂在JSP页面里并不是件好事。
- ❑ 如果不让Java代码出现在JSP页面里，那么你必须为大多数业务逻辑编写自定义标签，因此你需要花费更多的时间开发应用程序。相比之下，把Java代码编写在Java类里更节约时间。

Model 2 设计模型

第二种设计模型简称为Model 2。如果打算使用Java语言开发Web项目，这一模型应该是首选的架构。Model 2是MVC（Model-View-Controller，模型-视图-控制器）设计模式的另一个名字。按照Model 2模型开发的应用程序由三个主要部分组成：模型、视图和控制器。第1章将对MVC模式做进一步讨论。

注解 Model 2这个术语最早出现在0.92版本的JSP规范（JavaServer Pages Specification）里。

在Model 2模型里，所有的页面都有一个共同的入口点，通常是由一个servlet来充当主控制器，JSP页面负责实现页面表现。与Model 1相比，基于Model 2的应用程序主要有以下几个优点：

- 开发速度快；
- 易于测试；
- 易于维护；
- 易于扩展。

现在，你应该知道为什么servlet和JSP会同时被程序员使用了。

注解 有个问题常被问及：如果整个应用程序只有一个servlet而这又是整个应用程序的唯一入口，代码的性能会不会受到影响？servlet的容器会不会“繁殖”出该servlet的许多个实例？答案是：这个容器将只创建一个servlet实例——除非你实现了`javax.servlet.SingleThreadModel`接口（这种做法不推荐，Servlet 2.4已经淘汰了它），代码的性能不会受到影响。对这个问题的详细讨论见我的另一本书*How Tomcat Works*。

Struts 的优点

理解MVC模式（Model 2）为什么是用Java语言开发Web应用程序时的首选设计模型之后，接下来的问题就是：怎样才能加快Model 2应用程序的开发速度？

这也正是Craig McClanahan在创建Struts之前想到的问题。Struts是一种用来开发Model 2应用程序的框架。这个框架之所以能加快开发速度，是因为它提供的下面这些功能解决了Web应用程序开发过程中的一些常见问题：

- 页面导航管理；
- 用户输入验证；
- 一致的布局；
- 可扩展性。

因为Struts是一个Model 2框架，所以在使用Struts的时候必须严格遵守以下几条不成文的规定：

- 不要在JSP页面里嵌入Java代码，所有的业务逻辑都驻留在称为动作类（action class）的Java

类里。

- ❑ 在JSP页面里通过表达式语言（Expression Language, EL）（如果你使用的是JSP 2.0的话）或JSTL访问有关的业务模型。
- ❑ 尽量避免编写自定义标签（因为自定义标签的代码比较难以编写）。

各章内容概述

本书的目标读者是那些希望学习如何开发Struts应用程序的人。不过，我们想通过本书达到的目的并不仅限于此。在本书里，我们还花一些篇幅教授编写高效Struts应用程序的方法。为了使本书从头到尾环环相扣，我们在编写本书时一直把“概念清晰，便于学习”摆在首位。

下面是本书各章内容的概述。

第1章：Model 2 与 Struts

该章分析了Model 2的架构，并提供了两个Model 2应用程序的示例。这章还介绍了用来快速构建Model 2应用程序的一个主流编程框架——Struts。

第2章：利用动作表单验证输入

该章讨论了Struts的一个主要功能——帮助程序员对用户输入进行验证。输入验证是Web应用程序开发工作中的一项重要任务，它主要包括以下几项：必填字段里是否有值，日期或数值的输入格式是否正确等。Struts动作表单（action form）不仅能让输入验证工作变得非常容易实现，还可以在无需编写Java代码的情况下让应用程序显示出错消息，并把用户刚才输入的值重新显示在有关字段里。

第3章：HTML 标签库

该章解释了HTML标签库中的各种标签，这个库是Struts自带的3个核心标签库之一。如果你想最大限度地利用Struts，就应该在JSP页面里只使用这些标签来生成HTML表单和其他HTML元素。

第4章：输入验证和数据转换

该章探讨了输入验证工作和数据转换工作的细节问题。我们知道，在进行输入验证之前几乎总是要对有关数据进行这样或那样的格式转换。这一章为大家准备了3个示例，还对Jakarta Commons项目的成果之一（BeanUtils工具类）进行了讨论。

第5章：Validator 插件

该章讨论了Struts中最有用的一个特性：Validator插件。利用这个插件，你可以在无需编写任何Java代码的情况下对用户输入进行验证。这一章也探讨了使用和扩展这个插件的每一个细节。

第6章：EL 语言

本书的第6章~第9章解释了如何从JSP页面访问MVC模式中的各种模型对象。这一章重点介绍了JSP 2.0中的EL（表达式语言）。EL语言是JSP 2.0中最重要的特性之一，它不仅可以帮助你编写出更简短、更高效的JSP页面，还可以帮助你编写出不掺杂scriptlet的页面。

第7章：JSTL

该章介绍了JSTL中用来访问各种作用域对象（scoped object）和作用域对象的属性，遍历集合中的各个成员，进行条件测试，对日期和数值进行解析和格式化的各种标签，以及JSP 2.0 EL表达式中的常用函数。

第8章：Bean 标签库

该章介绍了Bean标签库，这个库也是Struts自带的三个核心标签库之一。这个库里的标签主要用于访问各种作用域对象和包容在HttpServletRequest对象里的各种对象。

第9章：Logic 标签库

该章介绍了Logic标签库。这个库里的标签可以用来完成以下工作：进行值的比较，对字符串进行匹配，对用户进行转发和重定向，对集合进行遍历，等等。

第10章：Struts-EL 标签库、Nested 标签库和 selectLabel 标签

该章介绍了Struts-EL标签库。Nested标签库和名为selectLabel的自定义标签。后者用来显示某个选中选项的文本。

第11章：消息处理与国际化

该章讨论了消息处理。消息处理也是应用程序开发工作最重要的任务之一。时至今日，能否用多种语言显示消息已经成为评价应用程序优劣的一项必备条件。在Struts的设计过程中，国际化和本地化一直是工作重点之一，所以它的三个核心标签库里的各有关标签都支持对多语种消息的处理。此外，本章还向大家展示了如何利用JSTL库里的几个标签显示和格式化一条消息的方法。

第12章：Tiles 框架

该章讨论了如何使用Tiles框架让Struts应用程序获得一致的视觉效果。Tiles远优于JSP包含文件，程序员可以利用Tiles布局和编写各种定义页面。定义页面的类型有两种：一种是JSP页面，另一种是XML页面。定义页面里的定义元素或标签允许扩展。

第 13 章：Struts 应用程序的安全性

这一章将介绍如何配置部署描述文件（deployment descriptor）才能对应用程序里的部分或全部资源的访问加以限制。配置在这里的含义是你只需要修改部署描述文件——不需要进行任何编程。此外，还将介绍如何在Struts配置文件里使用action元素的roles属性。当然，你也可以利用javax.servlet.http.HttpServletRequest接口提供的getRemoteUser、getUserPrincipal、getAuthType和isUserInRole方法编写Java代码来保护Web应用程序的安全。

第 14 章：Config 对象

该章介绍了Config对象的实现方法。这种对象可以把键/值对保存起来供整个应用程序使用。键/值对的来源可以是部署描述文件也可以是一个属性文件，可以使用一个应用程序侦听器（application listener）来填充Config对象。

第 15 章：持久层

该章讨论了存储各种值对象所需要的持久层（persistence layer）。持久层可以让其客户（就本书讨论的问题而言，持久层的客户主要是有关的Struts动作对象）无需关心访问数据库时的复杂性。持久层可以实现为实体bean、DAO（Data Access Object，数据访问对象）模式，或者使用Hibernate，等等。这一章将向大家详细演示如何实现DAO模式。这类模式有许多种变体，你可以根据项目的具体情况进行选择。最灵活的DAO模式当然是最佳的选择，这可以让你在日后需要修改应用程序时更易扩展。

第 16 章：对象缓存

与其他操作相比，创建Java对象的开销相对较大，而创建某种特定类型的对象（比如数据库连接）对计算机来说也是累活。这一章演示了如何才能高效率地缓存对象。

第 17 章：文件的上传与下载

该章讨论了现有的Web编程书普遍没有给予足够重视的两个重要论题。Struts通过与Jakarta Commons FileUpload库的无缝集成来支持文件上传。这一章的第一节讨论如何使用Struts实现这一编程任务，第二节讨论文件下载的实现问题，并演示了如何把二进制数据流发送给浏览器。

第 18 章：分页与排序

所谓分页是指把有关数据显示在多个页面上。当需要处理的数据量比较大时，经常需要使用分页。如果你想拿出一个高效的解决方案，首先要判断操作检索出来的数据是一个大的集合还是一个超大的集合。如果是前一种情况，基于缓存的分页解决方案往往效率更高；如果是后一种情况，基于查询的分页往往更好。这一章还将介绍显示标签库（Display Tag Library），这个库默认支持分页和排序。

第 19 章：防止重复提交

该章解释了如何利用Struts的内建功能防止用户重复提交请求。重复提交有时属于用户的错误操作，有时是因为表单处理时间过长而让用户不知应该如何是好。

第 20 章：早期会话验证

该章介绍了一个技巧，即如果用户关闭了其浏览器，该技巧将提前结束相应的HttpSession对象的生命，从而有效地释放宝贵的内存空间。

第 21 章：装饰 HttpServletRequest 对象

该章介绍了利用Decorator设计模式来打包HttpServletRequest对象。这一章的一个示例里实现了一个去空格过滤器。

第 22 章：Struts 的工作原理

作为全书的总结，这一章介绍了Struts的内部工作原理。具体地说，解释了Struts中的三个重要类型：ActionServlet、RequestProcessor和ConfigModule。理解Struts的工作原理可以帮助我们编写出更高效的应用程序。

附录 A：Struts 配置

这个附录对Struts配置文件进行了汇总介绍。

代码下载

本书的特色之一是每个主题都配有大量的示例。相关的程序代码可以从以下站点免费下载：

<http://www.brainysoftware.com/books/struts/mgm/StrutsBookApps.zip>

各章的配套应用程序按以下格式命名：

appXXy

其中xx是一个用两位数表示的章号，y代表对应那一章里的第几个示例程序。比如，第1章里的第2个示例程序将是app01b。

本书中的所有应用程序示例都已经在Tomcat 5环境下通过了测试。它们在我的机器上都运行在8080号端口上。因此，所有的示例程序的URL都需要以http://localhost:8080开头，然后写出示例程序的名字和servlet的路径。

其他学习资源

与本书有关的资源有一些可以在本书英文版出版公司的Web站点（www.brainysoftware.com）

com) 和其他站点上找到, 还有一些则已经收录在了可下载的ZIP文件里¹。下面是这些资源的清单。

- *Introduction to XML*: Struts和Tiles配置文件都是些XML文档。因此, 你们需要了解什么是XML文档以及如何使用。你们可以在www.brainysoftware.com上找到一篇关于XML的文章。千万记得还要阅读关于DTD文件的有关章节。²
- *The complete reference on JSTL*: 第7章对与Struts编程工作有关的部分标签进行了讨论。如果你们想了解更多情况, 请从<http://java.sun.com/products/jsp/jstl/>下载这份文献。
- *Digester*: Digester是一个开源Java库, 它可以帮助我们吧XML文档转换为Java对象。你可以在我为大家准备的可下载ZIP文件里找到一章关于Digester的内容。

1. 该文件也可以从图灵网站下载。——编者注

2. 想了解更多有关XML知识的读者, 可以阅读《XML基础教程》(人民邮电出版社)。——编者注

目 录

第 1 章 Model 2 与 Struts	1
1.1 传输对象	3
1.2 简单的 Model 2 应用程序	4
1.2.1 ProductTO 类	6
1.2.2 ControllerServlet 类	6
1.2.3 视图	8
1.2.4 部署描述文件	11
1.2.5 运行示例程序	11
1.3 视图管理器和动作管理器	11
1.4 Struts 入门	14
1.4.1 Struts 配置文件	14
1.4.2 Action、ActionForward 和 ActionMapping	16
1.4.3 编辑部署描述文件 (web.xml 文件)	17
1.5 Struts 应用程序示例	18
1.5.1 部署描述文件和 Struts 配置 文件	19
1.5.2 动作类	20
1.5.3 运行示例程序	21
1.6 Struts 的 MVC 特性	21
1.7 小结	22
第 2 章 利用动作表单验证输入	23
2.1 动作表单概述	23
2.2 ActionForm 类	25
2.3 使用动作表单	27
2.4 使用 HTML 标签库	34
2.4.1 重新显示表单里的值	37
2.4.2 出错处理	37
2.5 另一个示例程序	38
2.5.1 目录结构	39
2.5.2 Struts 配置文件	39
2.5.3 动作类	40
2.5.4 JSP 页面	42
2.5.5 运行 app02c 应用程序	43
2.6 多页表单	44
2.6.1 目录结构	45
2.6.2 Struts 配置文件	45
2.6.3 动作表单	46
2.6.4 动作类	47
2.6.5 运行 app02d 应用程序	48
2.7 动态动作表单	48
2.7.1 目录结构	49
2.7.2 Struts 配置文件	49
2.7.3 动作类	50
2.7.4 运行 app02e 应用程序	50
2.8 小结	51
第 3 章 HTML 标签库	52
3.1 html 标签	54
3.2 base 标签	55
3.3 form 标签	55
3.4 text、password、hidden、textarea 标签	56
3.5 submit 和 cancel 标签	60
3.6 reset 标签	61
3.7 button 标签	61
3.8 checkbox 标签	61
3.9 multibox 标签	62
3.10 radio 标签	64
3.11 select 标签	66
3.12 option 标签	67
3.13 options 标签	68
3.13.1 options 标签的用法: 与包含 集合的对象配合使用	69

3.13.2 options 标签的用法: 与作用域集合配合使用	70	5.2.1 编写动作表单	102
3.14 optionsCollection 标签	71	5.2.2 编写验证规则	103
3.14.1 optionsCollection 标签的用法: 与包含集合的对象配合使用	71	5.2.3 运行 app05a 应用程序	104
3.14.2 optionsCollection 标签的用法: 与集合配合使用	72	5.3 验证文件	104
3.15 link 标签	72	5.3.1 global 元素及 validator 元素	106
3.15.1 添加一个请求参数	74	5.3.2 formset 元素及 form 子元素	106
3.15.2 添加多个请求参数	75	5.3.3 向验证器传递参数	108
3.16 img 标签	75	5.3.4 Validator 的出错处理机制	109
3.17 rewrite 标签	77	5.4 使用客户端验证	110
3.18 frame 标签	77	5.5 使用 DynaValidatorForm 类	112
3.19 image 标签	77	5.6 Validator 与自定义验证的结合	112
3.20 xhtml 标签	78	5.7 声明基于动作的验证规则	114
3.21 小结	78	5.8 编写自定义验证器	118
第 4 章 输入验证和数据转换	79	5.8.1 编写自定义验证器: MyCustomValidator 类	119
4.1 为动作表单的属性选择数据类型	79	5.8.2 注册自定义验证器	120
4.2 输入验证和数据转换示例	80	5.8.3 验证文件	121
4.3 使用 BeanUtils 类	87	5.8.4 注册 Validator 插件	121
4.4 提高数据转换的效率	89	5.8.5 运行 app05f 应用程序	122
4.5 小结	95	5.9 支持客户端验证的自定义验证器	122
第 5 章 Validator 插件	96	5.9.1 编写 JavaScript 函数	122
5.1 Validator 插件的默认验证器	96	5.9.2 注册改进版的自定义验证器	124
5.1.1 required 验证器	97	5.9.3 JSP 页面	124
5.1.2 minlength 验证器	97	5.9.4 运行 app05g 应用程序	125
5.1.3 maxlength 验证器	98	5.10 使用 validwhen 验证器	125
5.1.4 byte、short、integer、long、float、double 验证器	98	5.11 小结	126
5.1.5 intRange 验证器	98	第 6 章 EL 语言	127
5.1.6 floatrange 验证器	98	6.1 EL 语言的语法	127
5.1.7 date 验证器	98	6.1.1 EL 语言中的保留字	128
5.1.8 mask 验证器	98	6.1.2 [] 和 . 操作符	128
5.1.9 email 验证器	99	6.1.3 EL 表达式求值规则	129
5.1.10 url 验证器	99	6.2 访问 JavaBean	130
5.1.11 creditCard 验证器	99	6.3 EL 语言的隐含对象	130
5.1.12 validwhen 验证器	99	6.3.1 pageContext 对象	131
5.2 使用 Validator 插件	100	6.3.2 initParam 对象	132
		6.3.3 param 对象	132
		6.3.4 paramValues 对象	132
		6.3.5 header 对象	132

6.3.6 headerValues 对象	132	7.6.4 escapeXml 函数	156
6.3.7 cookie 对象	133	7.6.5 indexOf 函数	156
6.3.8 applicationScope、session- Scope、requestScope 和 pageScope 对象	133	7.6.6 join 函数	156
6.4 其他 EL 操作符	133	7.6.7 length 函数	156
6.4.1 算术操作符	133	7.6.8 replace 函数	157
6.4.2 关系操作符	134	7.6.9 split 函数	157
6.4.3 逻辑操作符	134	7.6.10 startWith 函数	157
6.4.4 条件操作符	134	7.6.11 substring 函数	157
6.4.5 empty 操作符	135	7.6.12 substringAfter 函数	158
6.5 JSP 2.0 中的 EL 语言	135	7.6.13 substringBefore 函数	158
6.5.1 无脚本 JSP 页面	135	7.6.14 toLowerCase 函数	158
6.5.2 禁用 EL 求值	135	7.6.15 toUpperCase 函数	158
6.6 小结	137	7.6.16 trim 函数	158
第 7 章 JSTL	138	7.7 小结	159
7.1 JSTL 简介	138	第 8 章 Bean 标签库	160
7.2 通用动作	139	8.1 write 标签	162
7.2.1 out 标签	139	8.2 cookie 标签	163
7.2.2 set 标签	140	8.3 header 标签	164
7.2.3 remove 标签	142	8.4 parameter 标签	164
7.3 条件动作	143	8.5 define 标签	164
7.3.1 if 标签	143	8.6 page 标签	165
7.3.2 choose、when 和 otherwise 标签	144	8.7 include 标签	165
7.4 遍历动作	145	8.8 message 标签	166
7.4.1 forEach 标签	145	8.9 resource 标签	167
7.4.2 forTokens 标签	147	8.10 size 标签	167
7.5 格式化动作	148	8.11 struts 标签	168
7.5.1 formatNumber 标签	148	8.12 小结	168
7.5.2 formatDate 标签	150	第 9 章 Logic 标签库	169
7.5.3 timeZone 标签	151	9.1 值比较标签	170
7.5.4 setTimeZone 标签	152	9.1.1 present 和 notPresent 标签	171
7.5.5 parseNumber 标签	152	9.1.2 equal、notEqual、lessThan、 lessEqual、greaterThan 和 greaterEqual 标签	171
7.5.6 parseDate 标签	153	9.2 子串匹配标签	171
7.6 函数	155	9.3 表现定位标签	173
7.6.1 contains 函数	155	9.3.1 forward 标签	173
7.6.2 containsIgnoreCase 函数	155	9.3.2 redirect 标签	173
7.6.3 endsWith 函数	155	9.4 集合处理标签	174
		9.5 小结	176

第 10 章 Struts-EL 标签库、Nested 标签库和 selectLabel 标签.....177

- 10.1 Struts-EL 标签库.....177
- 10.2 Nested 标签库.....179
- 10.3 selectLabel 标签.....182
- 10.4 小结.....186

第 11 章 消息处理与国际化.....187

- 11.1 在 Struts 中处理地区.....187
- 11.2 注册属性文件.....190
- 11.3 HTML 标签库中的 errors 标签.....190
- 11.4 HTML 标签库中的 messages 标签.....191
- 11.5 JSTL `intl` 标签库.....192
 - 11.5.1 setLocale 标签.....192
 - 11.5.2 bundle 标签.....193
 - 11.5.3 setBundle 标签.....193
 - 11.5.4 message 标签.....194
 - 11.5.5 param 标签.....195
 - 11.5.6 requestEncoding 标签.....195
- 11.6 小结.....195

第 12 章 Tiles 框架.....196

- 12.1 JSP 包含的不足.....197
- 12.2 初识 Tiles.....198
- 12.3 使用 JSP 定义文件.....200
- 12.4 使用 XML 定义.....201
- 12.5 XML 定义的继承.....204
- 12.6 Tiles 标签库里的标签.....205
 - 12.6.1 add 标签.....206
 - 12.6.2 definition 标签.....207
 - 12.6.3 getAsString 标签.....207
 - 12.6.4 importAttribute 标签.....207
 - 12.6.5 insert 标签.....208
 - 12.6.6 put 标签.....209
 - 12.6.7 putList 标签.....209
 - 12.6.8 useAttribute 标签.....210
- 12.7 编写 XML 格式的 Tiles 定义文件.....210
 - 12.7.1 tiles-definition 元素.....211
 - 12.7.2 definition 元素.....211
 - 12.7.3 put 元素.....211

- 12.7.4 putList 元素.....212
- 12.7.5 add 元素.....212
- 12.7.6 bean 元素.....212
- 12.7.7 item 元素.....213
- 12.7.8 set-property 元素.....213
- 12.8 小结.....213

第 13 章 Struts 应用程序的安全性.....214

- 13.1 用户和角色.....214
- 13.2 编写安全策略.....215
 - 13.2.1 保护资源.....215
 - 13.2.2 指定登录方法.....216
- 13.3 身份验证方法.....217
 - 13.3.1 使用基本身份验证方法.....218
 - 13.3.2 使用基于表单的身份验证方法.....218
- 13.4 隐藏资源.....220
- 13.5 Struts 安全配置.....221
- 13.6 以编程方式提高安全性.....222
 - 13.6.1 getAuthType 方法.....222
 - 13.6.2 isUserInRole 方法.....222
 - 13.6.3 getUserPrincipal 方法.....223
 - 13.6.4 getRemoteUser 方法.....223
- 13.7 小结.....223

第 14 章 Config 对象.....224

- 14.1 使用 web.xml 文件或属性文件.....224
- 14.2 Config 对象.....225
 - 14.2.1 Singleton 模式简介.....225
 - 14.2.2 实现 Singleton Config 对象.....226
 - 14.2.3 运行 appl4a 应用程序.....230
- 14.3 小结.....230

第 15 章 持久层.....231

- 15.1 DAO 模式.....232
 - 15.1.1 DAO 模式的最简单实现.....232
 - 15.1.2 使用 DAO 接口的 DAO 模式.....232
 - 15.1.3 使用 Abstract Factory 模式的 DAO 模式.....233
- 15.2 实现 DAO 模式.....234
 - 15.2.1 DAO 接口和 DAOBase 类.....234
 - 15.2.2 Config 类.....237

15.2.3 CustomerDAO 接口	238	第 20 章 早期会话验证	284
15.2.4 CustomerDAOMySQLImpl 类	239	20.1 在 Struts 应用程序中让 HttpSession	
15.2.5 DAOFactory 类	244	对象失效	284
15.2.6 客户组件	245	20.2 让 HttpSession 对象尽早失效	285
15.2.7 运行 app15a 应用程序	245	20.3 小结	287
15.3 复杂的数据结构	246	第 21 章 装饰 HttpServletRequest	
15.4 Hibernate	247	对象	288
15.5 小结	247	21.1 Decorator 模式	288
第 16 章 对象缓存	248	21.2 去空格过滤器	290
16.1 实现缓存	248	21.3 小结	292
16.2 小结	250	第 22 章 Struts 的工作原理	293
第 17 章 文件的上传与下载	251	22.1 动作 servlet	293
17.1 文件上传	251	22.2 创建 ModuleConfig 对象	295
17.1.1 文件上传概述	251	22.3 请求处理器	298
17.1.2 Struts 中的文件上传	252	22.3.1 processMultipart 方法	299
17.2 文件下载	256	22.3.2 processPath 方法	300
17.3 小结	261	22.3.3 processLocale 方法	301
第 18 章 分页与排序	262	22.3.4 processContent 方法	302
18.1 大数据集/超大数据集	262	22.3.5 processNoCache 方法	302
18.2 Display 标签库简介	263	22.3.6 processPreprocess 方法	302
18.2.1 table 标签	264	22.3.7 processCachedMessages	
18.2.2 column 标签	265	方法	303
18.2.3 setProperty 标签	265	22.3.8 processMapping 方法	303
18.3 处理大数据集	266	22.3.9 processRoles 方法	305
18.4 处理超大数据集	269	22.3.10 processActionForm 方法	305
18.4.1 searchCustomers 方法	270	22.3.11 processPopulate 方法	306
18.4.2 CustomerSearchCriteria-		22.3.12 processValidate 方法	307
Form 动作表单	272	22.3.13 processForward 方法	308
18.4.3 SearchCustomerAction 类	273	22.3.14 processInclude 方法	309
18.4.4 displaySearchCustomer-		22.3.15 processActionCreate	
Form.jsp 页面	276	方法	309
18.4.5 运行 app18b 应用程序	278	22.3.16 processActionPerform	
18.5 小结	279	方法	310
第 19 章 防止重复提交	280	22.3.17 processForwardConfig	
19.1 Struts 中的标记管理	280	方法	311
19.2 使用标记	281	22.4 小结	312
19.3 小结	283	附录 A Struts 配置	313