

Neusoft

“十一五”国家重点电子出版物出版规划
东软软件工程师系列教程

Struts框架 应用与开发

李世祥 编著

东软电子出版社

“十一五”国家重点电子出版物出版规划

东软软件工程师系列教程

Struts 框架应用与开发

李世祥 编著

东软电子出版社

• 大连 •

Struts 框架应用与开发

编 著 者:李世祥

责任编辑:周凌波

光盘开发:夏一丁

装帧设计:王 岩

书 号:ISBN 978-7-900680-36-5

出 版 者:东软电子出版社

发 行 者:东软电子出版社

地 址:大连市软件园路 8 号

邮 编:116023

电 话:0411-84835086/84835087

传 真:0411-84835089

电子邮箱:nep@neusoft.edu.cn

出版时间:2007 年 7 月

印 制 者:大连华录影音实业有限公司

大连金华光彩色印刷有限公司

题词

这是一套来自实践者的作品。对中国 IT 教育和软件产业实践应用更紧密结合愿景的期待,驱动了东软的员工组织起来完成了这套作品。他们力图将自己在教育和研发中的收获传播给自己的同事、IT 教育和软件产业的同行,并与在蓬勃发展中的中国软件产业分享;他们企盼着中国未来的软件工程师们在一个更贴近实用化的环境中学习和掌握技术的价值。东软为他们的行动而感动和自豪。

东软集团有限公司董事长,总裁

刘积仁

东软应用型 IT 教育系列教材

总编辑委员会

顾 问	刘积仁					
主 编	温 涛					
编辑委员	张 伟	曹云凤	翟吉方	刘春雨	孙福权	
	齐志儒	刘文涛	张 韬	李旭光	陈曦光	
	孙晓凌					

东软软件工程师系列教程

编辑委员会

主 编	温 涛					
副 主 编	翟吉方					
编辑委员	陆元婕	郑吉军	秦 健	朱 毅	王 凯	
	陈丽萍	姜 英	李世祥	方建平	张晓芳	
	秦建国	吕海东				

东软软件工程师系列教程

审稿委员会

主 任

张 霞 东软软件股份有限公司 首席技术官兼首席知识官
齐志儒 东北大学东软信息学院 顾问； 计算机科学与技术系 主任

委 员

赵立军 东软软件股份有限公司 技术战略与发展部 部长
纪 勇 东软软件股份有限公司 基础软件事业部 部长
于洪勇 东软软件股份有限公司 解决方案技术中心 主任
刘 洋 东软软件股份有限公司 技术战略与发展部 软件工程师
王世龙 东软软件股份有限公司 技术战略与发展部 架构设计师
柴永春 东软软件股份有限公司 基础软件事业部
技术总监 & 软件架构师
王治博 东软软件股份有限公司 基础软件事业部 资深软件工程师
杜田甜 东软软件股份有限公司 基础软件事业部 资深软件工程师
赵 伟 东软软件股份有限公司 解决方案技术中心 资深技术顾问
郭瑞斌 东软软件股份有限公司 解决方案技术中心 资深技术顾问
王 哲 东软软件股份有限公司 解决方案技术中心 系统工程师
滕久华 东软软件股份有限公司 解决方案技术中心 技术经理
郭 权 东北大学东软信息学院 计算机科学与技术系 副主任
张冬青 东北大学东软信息学院 计算机科学与技术系
计算机基础教研室 主任
滕英岩 东北大学东软信息学院 计算机科学与技术系
计算机科学与技术教研室 主任
毛翼飞 东北大学东软信息学院 计算机科学与技术系 资深讲师
闫海珍 东北大学东软信息学院 计算机科学与技术系 资深讲师
樊晓勇 东北大学东软信息学院 计算机科学与技术系 资深讲师

前 言

Jakarta-Struts 是 Apache 软件组织提供的一项开放源代码项目,它为 Java Web 应用提供了模型-视图-控制器(Model-View-Controller, MVC)框架,尤其适用于开发大型可扩展的 Web 应用。随着 Web 技术的成熟和发展,Struts 框架逐渐成为了 Web 开发中的主流框架。Struts 框架为软件开发提供了丰富的资源和软件开发包,大大提高了开发 Web 应用的速度和效率。

“理论+实践”是本书的一大特点。本书中不仅有完整的理论讲解,而且全书由一个完整的实例贯穿始终,让读者在实践中深入掌握 Struts 框架的相关知识。

本书凝聚了作者多年的实践经验,并讲述实际开发中最需要的技术,适合所有准备学习 Struts 框架的读者。

由于时间仓促,书中存在纰漏在所难免,恳请广大读者指正。

编 者

2007 年 3 月

目 录

第 1 章 Struts 与 MVC 框架介绍	1
1.1 基于 J2EE 的 Java Web 技术及应用	1
1.1.1 Java Web 应用	1
1.1.2 Java Web 技术介绍	2
1.2 典型的 JSP 设计模式 Model I 与 Model II	3
1.2.1 JSP 设计模式 Model I	3
1.2.2 JSP 设计模式 Model II	4
1.3 Struts 框架简介	6
1.3.1 Struts 框架的起源	6
1.3.2 Struts MVC 体系结构	6
1.3.3 Struts 框架的工作流程	8
1.4 获取和安装 Struts	10
1.4.1 获取 Struts	10
1.4.2 Struts 框架的安装	10
1.5 本章小结	11
第 2 章 使用 Struts 框架开发第一个应用	12
2.1 sayHello 应用分析	12
2.2 创建视图组件	13
2.3 创建 ActionForm 类	18
2.3.1 ActionForm 类介绍	18
2.3.2 表单验证	20

2.4 创建 Action 类	22
2.5 创建 web.xml 及 struts-config.xml 文件	24
2.5.1 部署描述文件	24
2.5.2 struts-config.xml 文件	25
2.6 在 Web 服务器中部署 sayHello 应用程序	28
2.7 本章小结	29

第 3 章 Struts 核心组件介绍

30

3.1 控制器组件	30
3.1.1 ActionServlet	30
3.1.2 RequestProcessor 类	34
3.2 Action 组件类	37
3.2.1 Action 类介绍	37
3.2.2 Action 组件类的线程问题	38
3.3 ActionForward 与 ActionMapping 类	40
3.3.1 ActionForward 类	40
3.3.2 ActionMapping 类	41
3.4 其他内置 Action 类简介	42
3.4.1 org.apache.struts.actions.ForwardAction	42
3.4.2 org.apache.struts.actions.IncludeAction	42
3.4.3 org.apache.struts.actions.DispatchAction	43
3.5 ActionForm 组件及错误处理	45
3.5.1 创建 ActionForm	45
3.5.2 ActionForm 组件在 struts-config.xml 文件中的配置	48
3.5.3 Struts 框架的错误处理机制	49
3.6 扩展 Struts 框架	52
3.6.1 扩展 ActionServlet 类	52
3.6.2 扩展 RequestProcessor 类	54
3.6.3 扩展 ActionMapping	56
3.7 本章小结	60

第 4 章 Struts 配置	61
4.1 部署描述文件(web.xml)	61
4.2 struts-config.xml 配置文件	63
4.2.1 <data-sources>	64
4.2.2 <form-beans>	65
4.2.3 <global-exceptions>	66
4.2.4 <global-forwards>	67
4.2.5 <action-mappings>	68
4.2.6 <controller>	70
4.2.7 <message-resources>	71
4.2.8 <plug-in>	72
4.3 本章小结	73
 第 5 章 开发国际化的 Struts 应用	75
5.1 国际化与本地化	75
5.1.1 国际化	75
5.1.2 本地化	76
5.2 Java 对国际化的支持	76
5.2.1 Locale 类	76
5.2.2 ResourceBundle 类	82
5.2.3 MessageFormat 类	82
5.3 Struts 对国际化的支持	84
5.3.1 Struts 资源文件	84
5.3.2 使用资源文件	85
5.4 将 sayHello 应用实现国际化	88
5.5 解决中文问题的最佳实践	91
5.6 本章小结	96
 第 6 章 动态表单及 Validator 验证框架	97
6.1 动态表单	97

6.1.1	配置动态表单	98
6.1.2	动态表单中的 validate() 方法	99
6.1.3	动态表单中的 reset() 方法	100
6.2	Validator 验证框架	101
6.2.1	Validator 框架的安装	101
6.2.2	Validator 框架的配置	102
6.2.3	validator-rules.xml 配置文件	103
6.2.4	validation.xml 配置文件	106
6.2.5	Validator 验证框架中的 ActionForm	110
6.3	本章小结	115
第 7 章 Struts 框架中的异常处理		116
7.1	Struts 框架异常处理机制概述	116
7.2	基于配置的异常处理	120
7.3	基于编程的异常处理	122
7.4	本章小结	123
第 8 章 在 Struts 框架中使用数据库连接池		124
8.1	数据库连接池工作原理	125
8.2	Struts 框架中的连接池实现	126
8.2.1	数据库连接池配置	126
8.2.2	数据库连接池的使用	128
8.3	本章小结	133
第 9 章 Struts 自定义标记的使用		134
9.1	HTML 标记库	134
9.1.1	生成基于 HTML 元素的标记	134
9.1.2	显示错误或异常消息的标记	142
9.2	Bean 标记库	143
9.2.1	变量定义标记	143

9.2.2 显示数据标记	144
9.2.3 国际化标记	144
9.2.4 Bean 标记库中的其他标记	145
9.3 Logic 标记库	146
9.3.1 比较运算标记	146
9.3.2 字符串匹配标记	148
9.3.3 判断数据是否存在的标记	148
9.3.4 循环遍历标记	149
9.3.5 请求转发或重定向标记	150
9.4 本章小结	151
参考文献	153

第 1 章 Struts 与 MVC 框架介绍

开发 Web 应用程序是 Java 技术应用最为广泛的领域,而 MVC 框架则是 Java Web 应用中最常见的框架。对于大多数项目来说,如果独立实现一个 MVC 框架,将是一个非常庞大的工程。为了提高生产率,提高开发效率,在 MVC 框架的基础之上又出现了许多基于 MVC 的新的框架,这些框架出现的目的是为了简化 MVC 的开发过程,使开发基于 MVC 框架的 Web 应用程序更加简单。Struts 就是众多实现了 MVC 的开源框架之一。Struts 是目前最为流行的 Web 开发框架,它可以大大提高 Web 开发的效率,使用 Struts 框架让人们不用再花费心思去设计框架,而是考虑如何使用 Struts 框架解决实际业务问题。

1.1 基于 J2EE 的 Java Web 技术及应用

1.1.1 Java Web 应用

图 1-1 是一个典型的 J2EE 框架图。

典型的 J2EE 框架分为 4 个层次:客户端层、Web 服务器层、应用服务器层和数据服务器层。

(1)客户端层:可以采用多种客户端,比如浏览器、Java 应用程序等。但我们推荐使用浏览器作为标准客户端。

(2)Web 服务器层:主要是完成与客户端交互的任务,它能够接受客户端的请求(request),返回给客户端一个响应(response)。并且实现对数据业务处理的调用。在这个层次中,我们主要使用了 Servlet 技术以及 JSP 技术来实现 Web 的交互功能。

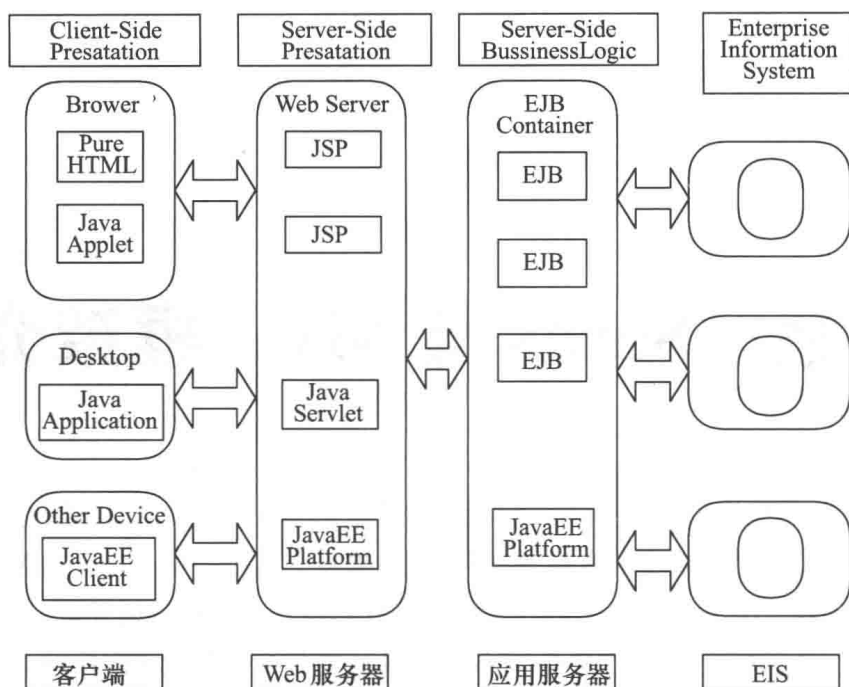


图 1-1 J2EE 框架图

(3)应用服务器层:在这个层次中,主要是对数据业务进行处理,完成复杂的数据计算。并且能够实现与数据库的交互。在 J2EE 框架中,提供了 EJB 技术来实现该层。

(4)数据服务层:主要是对数据进行持久保存,大多是由数据库来实现的。

在构建大多数项目的时候,不需要搭建在这个庞大的 J2EE 架构上,一般会使用 JavaBean 来替代 EJB 技术作为处理数据业务层(因为大多数项目不需要使用具有分布式计算能力的 EJB 技术)。这样就会把 J2EE 架构简化成一个常见的 Web 应用。

1.1.2 Java Web 技术介绍

开发一个 Java Web 应用一般会使用到以下技术:

- (1)Servlet;
- (2)JSP;
- (3)JavaBean;
- (4)XML。

Servlet 是进行 Web 开发的最常用的技术。Servlet 是服务器端组件,处理来自客

户端的请求信息和输出的响应。我们不提倡使用 Servlet 向客户端输出表示层数据。当然可以使用 Servlet 定义一个流(out 对象),将内容添加到 Web 页面中,但这不是一个好方法,因为这样做会使编写的 Servlet 类很复杂。Servlet 更适合使用 Java 语言来对要处理的数据进行逻辑控制。

JSP(JavaServer Pages)是一种基于 Java 语言的脚本技术。在 JSP 众多的优点之中,其中一个最显著的优点就是 JSP 将 HTML 编码从 Web 处理的业务逻辑中分离出来。这样会使编码更加简化(因为 HTML 编码可以直接写在 JSP 文件中),又将 JSP 中的页面信息输出与业务逻辑处理分离开。由于 JSP 编写简单,很快成为 Web 开发中应用最为广泛的技术。但 JSP 也有它的缺点,如果在 JSP 中出现大量的 Java 代码,会使 JSP 页面混乱难以阅读与维护。所以 JSP 中不适用于基于 Java 语言的业务逻辑处理,它应该完全作为客户端的视图,用来显示数据。

JavaBean 技术是 Web 开发中经常使用的技术,我们会定义很多 JavaBean 类,这些 JavaBean 类中封装处理数据的代码。在 Servlet、JSP 中去调用这些处理数据的方法。使用 JavaBean 技术会使代码逻辑更加清晰,可读性更高,代码更加灵活等。

XML 是 Web 开发中非常关键的技术应用,在 Web 开发中典型的 XML 应用就是 web.xml 文件。web.xml 文件又称为部署描述文件,在这个文件中,描述了很多重要信息,比如当前 Web 应用的 Servlet 类信息、Servlet 映射路径等信息。Web 服务器也是通过读取该文件从而获得相关信息。XML 文件的存在使我们对 Web 应用的配置更加方便、灵活。

1.2 典型的 JSP 设计模式 Model I 与 Model II

JSP 设计模式 Model I 与 Model II 是 Web 应用程序开发的最为常用的两种模式,在这一节中,我们一起来探讨这两种模式。

1.2.1 JSP 设计模式 Model I

Model I 模式就是 JSP、JavaBean 两种技术的结合。

JSP 有两个作用,一方面它要能够接受客户的请求信息,并处理相应的业务逻辑;另一方面需要给客户返回响应信息。

JavaBean 的作用就是进行数据的处理。这里也包括对数据库的操作。图 1-2 显示了 Model I 模式的工作方式。

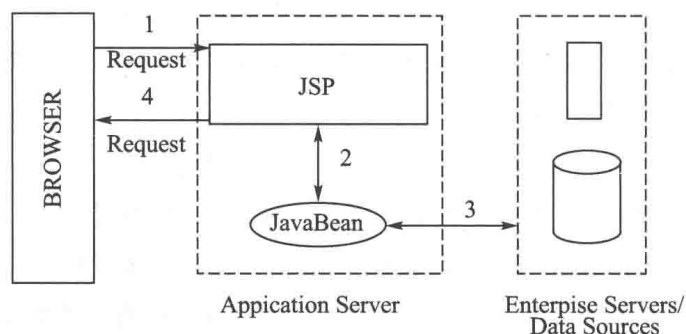


图 1-2 Model I 设计模式

图 1-2 中显示了 Model I 模式的工作流程如下：

- (1) 客户将请求提交给 JSP；
- (2) JSP 调用 JavaBean 组件进行数据处理；
- (3) 如果数据处理需要数据库支持，则使用 JDBC 操作数据库数据；
- (4) 当数据返回给 JSP 时，JSP 组织响应数据，返回给客户端。

由于 Model I 模式的设计流程清晰，以及 JSP 技术所带来的编码的简单性，使得 Model I 模式的应用比较广泛，尤其对于一些小型项目来说，Model I 模式是一个首选的模式。使用 Model I 模式会提高开发效率。

但是，Model I 模式也有其不足的地方。虽然 Model I 模式实现了一定的页面与数据业务逻辑的分离，但是这种逻辑分层并不能满足在大型项目中的要求。很多时候会需要有一个更加细化的逻辑分层结构。

使用 Model I 模式也会导致在 JSP 页面中嵌入大量的 Java 代码，这样会使页面变得庞大，同时也非常复杂。从而难以对 JSP 页面进行维护、升级。所以 Model I 模式对于小型的开发中比较有优势，但针对于中大型项目，它就不是一个非常好的选择。

1.2.2 JSP 设计模式 Model II

Model II 模式就是经典的 MVC 设计模式。Model II 模式主要是针对应用框架从逻辑上做了三层划分。这样层次更加清晰，任务分工更加明晰。所谓的三层划分指的就是：模型层(Model)、视图层(View)和控制层(Controller)三层。这三个层次中所使用的技术分别是 JavaBean、JSP 和 Servlet。图 1-3 显示了 Model II 模式的工作方式。

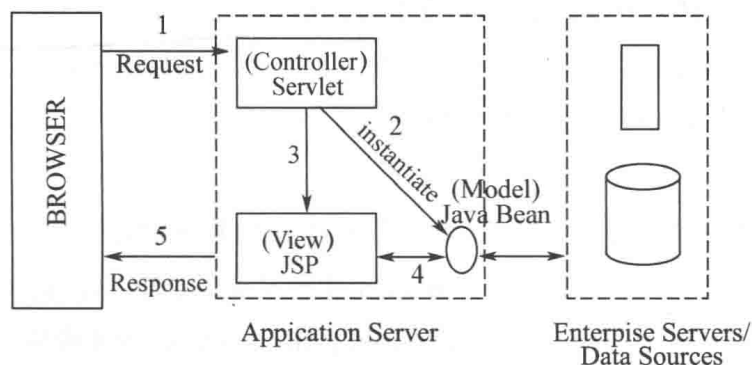


图 1-3 Model II 设计模式

图 1-3 展示了 MVC 的工作流程,具体如下:

(1) 客户的请求信息首先提交给 Servlet。在这里 Servlet 充当控制器的角色。所以对于控制器来说,一定要保证它的唯一性,也就是说,在整个应用中,有且只有一个控制器对象;

(2) 控制器接下来要选择对当前请求进行数据处理的 Model 对象。而 model 对象在 MVC 设计模式中的作用和在 Model I 模式中的作用是一致的,都是进行数据的处理,当然如果在进行数据处理的过程中需要使用数据库,那么它应该具有访问数据库的能力;

(3) 控制器对象还需要选择相应的 view 组件作为客户的响应信息返回。在这里 view 组件是由 JSP 技术来实现的,而 JSP 页面就是一个显示数据的页面,在 JSP 页面中不需要 Java 流程控制或逻辑处理的代码。它就是一个单纯的显示数据的页面;

(4) 对于 JSP 页面来说,它所要显示的数据内容的来源是 JavaBean 组件,也就是说,JSP 显示的是 JavaBean 处理的数据,所以,JSP 在数据显示的过程中需要使用 JavaBean 中的数据;

(5) 最后,JSP 把组织好的数据以响应的方式返回给客户端浏览,从而完成了一次客户访问的交互。

从分析 MVC 设计模式工作流程上,可以看出,原来在 Model I 模式中由 JSP 来完成的工作,现在把它分为了两层。Controller 层负责接受客户请求,View 层负责返回客