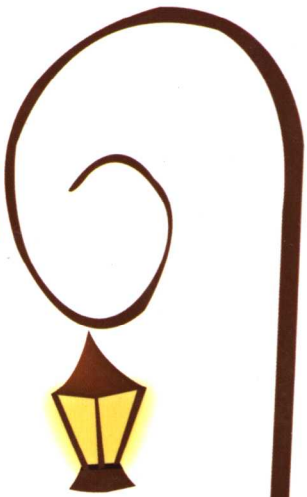
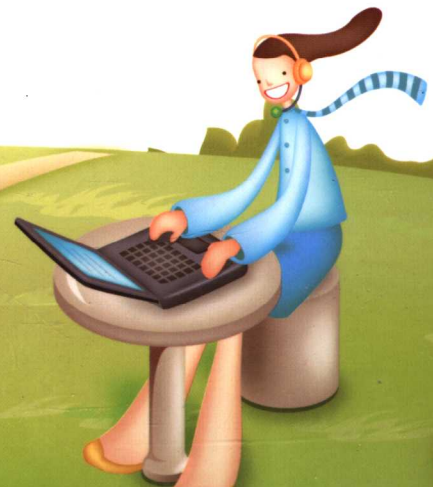
Struts与Hibernate实用教程

——构建基于MVC模式的高效Java Web应用

邬继成　编著

随书附带光盘
包含本书重点范例
源代码和所用到开源
软件的最新版本安装文件



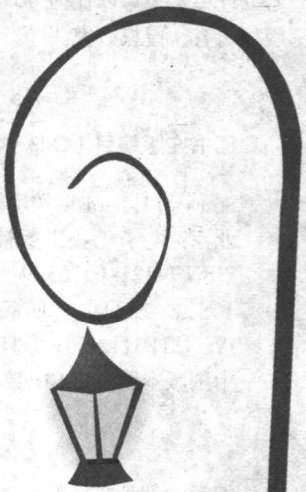


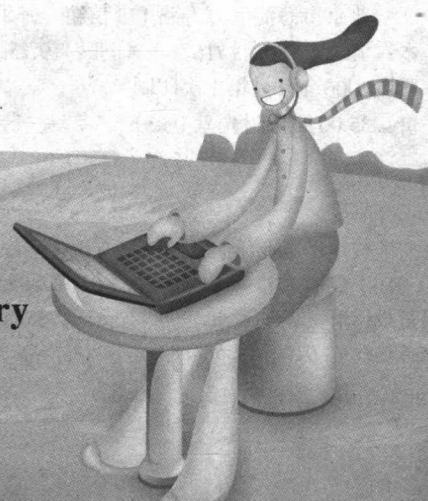
電子工業出版社
Publishing House of Electronics Industry
http://www.phei.com.cn

Struts与Hibernate实用教程

—— 构建基于MVC模式的高效Java Web应用

邬继成　编著





電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内容简介

Struts 和 Hibernate 是当前非常流行的 Java Web 应用框架，由于它们很好地实现了 MVC 设计模式以及具有使用简便、开源免费的特点，在国内外获得越来越广泛的应用。本书将结合实际例子由浅入深地介绍 Struts 和 Hibernate 的基本原理和应用方法，内容包括 Java Web 应用基础，Struts 和 Hibernate 的基本原理，实例讲解 Struts 和 Hibernate 应用程序的开发方法，一些相对高级但实用的 Struts 和 Hibernate 技术，以及 Struts 和 Hibernate 结合一起使用来构建一个完整的基于 MVC 模式的 Java Web 应用程序。

本书内容安排采用实用人才培训的思路，由浅入深，实用为主，既可以供广大工程技术人员参考，也是各类院校学生学习 Java Web 开发的绝佳入门读本。

图书在版编目（CIP）数据

Struts 与 Hibernate 实用教程：构建基于 MVC 模式的高效 Java Web 应用 / 邬继成编著.
北京：电子工业出版社，2006.9
ISBN 7-121-03163-9
Ⅰ.S... Ⅱ.邬... Ⅲ.①软件工具－程序设计－教材 ②Java 语言－程序设计－教材
Ⅳ.①TP311.56 ②TP312
中国版本图书馆 CIP 数据核字（2006）第 106778 号

责任编辑：贺瑞君
印　　刷：北京市天竺颖华印刷厂
装　　订：三河市金马印装有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编：100036
开　　本：787 × 980　1/16　　印张：19.75　　字数：506 千字
印　　次：2006 年 9 月第 1 次印刷
定　　价：35.00 元（附光盘 1 张）

凡所购买电子工业出版社的图书有缺损问题，请向购买书店调换；若书店售缺，请与本社发行部联系。联系电话：（010）68279077。邮购电话：（010）88254888。
质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。
服务热线：（010）88258888。

反侵权盗版声明

前　言

随着互联网和浏览器技术的普及，Web编程已变得越来越重要，各种Web编程技术和框架也随之涌现出来。在Web编程中，MVC（Model-View-Controller，即模型、视图和控制器）设计模式已经成为一种最主要的设计模式。Struts作为第一个基于MVC设计模式的开源Java Web编程框架，在业界受到了广泛的尊崇，得到了普遍应用，已经成为一个主流的Web编程框架。

任何应用软件都要与数据库连接，这就存在一个持久性问题，即数据在数据库中的存取问题。为解决这个问题，J2EE推出了实体Bean（Entity Bean）。但在后来的应用中，人们发现实体Bean存在许多问题，使用很不方便，不容易掌握，而且一旦程序出现Bug，很难找到原因，不容易调试。而Hibernate成功地解决了持久性问题。作为一个开源软件，它非常容易使用，可以根据用户的需要进行修改。所以它一经推出，立刻受到了业界空前的欢迎。

目前市面上已有一些关于Struts和Hibernate的图书，但它们许多都是大部头专著，过于理论化和系统化，其中很多内容应用很少。初学者要读完这样的书，需要耗费很长的时间和很多的精力，而且读完后往往还不知道如何着手编写Struts和Hibernate程序。因此，许多这样的大部头并不适合初学者。

本书内容来自作者多年的教学和软件开发经验，主要面对初学者，特别是针对面对就业压力、需要实践技能的大学生和年轻的程序开发爱好者。本书也可用做大专院校和计算机培训学校的指导图书。

本书的内容可分为三部分：第一部分讲解Struts的原理及其应用；第二部分讲解Hibernate的原理和应用；第三部分是提高篇，介绍如何用Struts和Hibernate一起构建Java Web应用，包括在Eclipse中开发Struts和Hibernate应用的方法，同时涉及一些常用开源软件工具的用法。

本书将结合应用例子由浅入深地介绍Struts和Hibernate的基本原理，着重于它们的实际应用。掌握本书内容后，读者将能够自己编写Struts和Hibernate的应用程序。因为把Struts和Hibernate结合在一起使用，可以构建出高效、完整的Java Web应用，所以作者把它们写入一本书中，这样可以突出它们之间的密切联系，有助于培养读者把这两种工具结合使用的良好习惯。如果将它们分别写成专著，则容易割裂它们之间的关系。一句话，该书是一本强调Struts和Hibernate开发技能的实用指导图书。我深信，该书会对将要走上工作岗位的广大学子和程序开发爱好者有实质性的帮助。如对本书有任何建议和批评，欢迎与我联系。我的电子邮箱是jcwu167@126.com。

邬继成

2006年9月于北京

目　录

第1章　Java Web编程基础

随着互联网和浏览器技术的日益普及，Web应用编程的重要性逐渐凸现出来，各种Web编程技术和框架也随之涌现出来。本章将简单介绍一些主流的Java Web编程技术、设计模式和框架。

本章的主要内容如下：

- 简单介绍Java Web编程的主要组件技术。
- 介绍MVC设计模式。
- 结合实例介绍如何构建一个基于MVC模式的Java Web应用程序。

通过本章的学习，大家应该对Java Web应用开发有一个比较全面的了解，并掌握如何开发一个简单的基于MVC设计模式的Java Web应用。

1.1　Java Web编程的主要组件技术

一个完整的Java Web应用软件通常是由多种组件构成的，一般由表示层组件、控制层组件、业务逻辑层组件及数据访问层组件组成。表示层组件通常由HTML和JSP页面构建，控制层组件一般是Servlet，业务逻辑层组件是JavaBeans或EJB，数据访问层组件是JDBC，Hibernate或CMP。此外，Java Web应用的各个组件需要在XML格式的配置文件中进行声明，然后打包，部署到Java Web服务器（例如Tomcat）中运行。下面我们将分别介绍Servlet，JSP，JavaBeans，JDBC，XML和Tomcat等主要的Java Web组件技术。

1.1.1　Servlet

Servlet（Java服务器小程序）是用Java编写的服务器端程序，是由服务器端调用和执行的、按照Servlet自身规范编写的Java类。

Servlet可以处理客户端传来的HTTP请求，并返回一个响应。它是一个Java类，具有可移植、功能强大、安全、集成、模块化和可扩展性好等特点。代码清单1.1所示为一个简单的小服务器程序。

代码清单1.1　一个简单的Servlet例子

```
-------------------------------------------------------------------------
import javax.servlet.*;
import java.io.* ;

public class HelloWorld extends GenericServlet
```

```
{
  public void service(ServletRequest request, ServletResponse response)
    throws IOException
    {
    //设定Web服务器的响应方式是HTML
    response.setContentType("text/html");

    //获得PrintWriter对象，以打印输出响应
    PrintWriter out = response.getWriter();

    //把HTML形式的响应显示在浏览器上
    out.println("<HTML>");
    out.println("<BODY>");
    out.println("<H1>Hello World!<H1>");
    out.println("</BODY>");
    out.println("</HTML>");
    }
  }
-------------------------------- End ----------------------------------
```

Servlet需要部署在Web容器中，它的生命周期由Web容器管理，可以分为以下几个阶段：

- 装载Servlet：这项操作一般是动态执行的。有些服务器提供了相应的管理功能，可以在启动的时候就装载Servlet，并能够初始化特定的Servlet。
- 创建一个Servlet实例。
- 调用Servlet的init( )方法。
- 服务：如果Web容器接收到对此Servlet的请求，那么它调用Servlet的service( )方法。
- 销毁：销毁实例，通过调用Servlet的destroy( )方法来销毁Servlet。

Java对Servlet的支持是通过javax.servlet和javax.servlet.http两个软件包实现的，其中javax.servlet.http是javax.servlet的一个子软件包。

javax.servlet软件包包含Servlet常用的接口和类，如GenericServlet，ServletConfig，ServletRequest，ServletResponse和RequestDispatcher等。

javax.servlet.Servlet接口提供了创建Servlet的一般框架。Servlet可直接实现此接口，或者间接地通过扩展javax.servlet.GenericServlet类或javax.servlet.http.HttpServlet类来实现此接口。javax.servlet.GenericServlet类可用于创建与任何协议一起使用的Servlet。

而javax.servlet.http软件包除了继承以上接口和类外，还包含一些针对HTTP协议的方法、特殊请求和响应对象，如HttpServlet，HttpServletRequest，HttpServletResponse，HttpSession和HttpDispatcher等。下面我们对它们进行简单介绍。

- HttpServlet类提供了处理不同HTTP请求类型（有GET，POST和PUT等）的方法，其中两种最常见的HTTP请求类型是GET和POST，它们分别是由doGet( )和doPost( )方法进行处理的：

```
protected void doGet(HttpServletRequest request,
                     HttpServletResponse response)
              throws ServletException, java.io.IOException;

protected void doPost(HttpServletRequest request,
                     HttpServletResponse response)
              throws ServletException, java.io.IOException;
```

doGet( )和doPost( )方法与代码清单1.1中的service( )方法具有相同的形式。HttpServlet类的doGet( )与doPost( )方法检查HTTP请求，然后调用相应的处理方法处理此请求。

- HttpServletRequest接口提供了处理客户请求的方法，如String getParameter（String name），String[ ] getParameterValues（String name）等方法。使用这些方法，Servlet可从客户页面窗体中取得数据。例如，从客户页面窗体取得用户名（username）的方法如下：

```
HttpServletRequest request;
String name = request.getParameter("username");
```

- HttpServletResponse接口提供了以HTML页面形式把请求发送给客户的方法，主要方法有setContentType( )，sendRedirect( )等。例如，把请求转发给下一个页面next.jsp的实现方法如下：

```
HttpServletResponse response;
response.sendRedirect("next.jsp");
```

- HttpSession接口用来记录当前Servlet中的用户会话。例如，在Web站点上注册的每个用户自动与HttpSession对象关联。Servlet可用此对象来存储关于用户会话的信息。可使用HttpSession接口的putValue( )和getValue( )等方法把数据存入HttpSession对象中，或从该对象中取出数据。通过HttpServletRequest对象的getSession()方法可生成一个HttpSession对象，如下面的代码所示：

```
HttpServletRequest request;
HttpSession session = request.getSession(true);
```

- RequestDispatcher接口可以把Servlet的请求提交或委派给另一个资源，如Servlet，HTML文件或JSP页面。在这种情况下，源Servlet进行同样处理并把请求委派给另一个Servlet。RequestDispatcher接口封装指向特定资源（Servlet，HTML文件或JSP页面）的URL。用此接口的forward（HttpServletRequest request, HttpServletResponse response）方法委派任务到

特定资源。RequestDispatcher对象可通过HttpServletRequest对象的getRequestDispatcher( )方法生成。例如：

```
RequestDispatcher rd = request.getRequestDispatcher("reply.jsp");
rd.forward(request,response);
```

代码清单1.2是一个典型的HttpServlet应用例子。

代码清单1.2 HttpServlet的应用

```
-------------------------------------------------------------------------
import javax.servlet.http.*;
import javax.servlet.*;
import java.io.*;
import java.util.*;

public class MyFirstServlet extends HttpServlet{

  public void doPost(HttpServletRequest request,HttpServletResponse response){

    String name = request.getParameter("username");
    RequestDispatcher rd;

    try {
      try {
       if(name.equals("Tom")) {
        HttpSession session = request.getSession(true);
        session.setAttribute("nm",name);
       response.sendRedirect("reply.jsp");
    }

     else {
rd = request.getRequestDispatcher("error.jsp");
       rd.forward(request,response);
     } }catch(ServletException e) { System.out.println(e);}
    } catch(IOException e) { System.out.println(e);}
  }

  public void doGet(HttpServletRequest request,HttpServletResponse response){
     doPost(request, response);
  }
}
-----------------------------------End ---------------------------------
```

在 Java Web 应用程序中，Servlet 常用做控制器组件。当 Servlet 部署到 Web 服务器中时，需要在 Web 配置文件 web.xml 中进行声明，如代码清单 1.3 所示。

代码清单 1.3　web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application
2.3//EN" "http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>

  <servlet>
    <servlet-name>myFirstServlet</servlet-name>
    <servlet-class>MyFirstServlet</servlet-class>
  </servlet>

<servlet-mapping>
    <servlet-name>myFirstServlet</servlet-name>
    <url-pattern>/myFirstServlet</url-pattern>
  </servlet-mapping>

</web-app>
```

---------- End ----------

1.1.2　JSP

JSP（Java Server Page）页面由 HTML 代码和嵌入其中的 Java 代码组成。在页面被客户端请求后，服务器对这些 Java 代码进行处理，然后将生成的 HTML 页面返回客户端的浏览器。Java 技术简单易用、完全面向对象、平台无关、安全可靠和面向 Internet 等特点，都在 JSP 中得到了体现。一个简单的 JSP 页面如代码清单 1.4 所示。

代码清单 1.4　一个简单的 JSP 页面

```
<html>
<body>
<%
   out.println("<H1>Hello World!</H1>");
%>
</body>
</html>
```

---------- End ----------

JSP 页面一般包含 JSP 指令、JSP 脚本元素、JSP 标准动作及 JSP 隐式对象。

1.1.2.1 JSP指令

JSP页面中的指令元素提供了特定JSP页面的全局信息，可分为三种类型： page指令，include指令及taglib（标记库）指令。把指令加入JSP页面的语法如下：

```
<%@ directive (attribute="attribute value") %>
```

1) page指令

page指令定义了一些属性，它们用来通知Servlet引擎有关JSP页面的一般设置。这些属性包括language，contentType，import，session，autoFlush，errorPage及isErrorPage等。它们的用法如表1.1所示。

表1.1 page指令

属性	描述和用法
language	在编写JSP页面时要用的脚本语言，例如Java语言。 用法：<%@ page language="java" %>
contentType	应答MIME类型（多用途Internet邮件扩充类型）。属性的默认值为text/html。例如： <%@ page contentType="text/html; charset=GB2312" %>
import	可用来在JSP页面中引入软件包或类，例如： <%@ page import="java.io.*; java.util.*" %>
session	用来指出关于特定JSP页面的会话数据的可用性。此属性的默认值为true。例如： <%@ page session="false" %>
autoFlush	自动刷新选项，如果是true则页面自动刷新，否则页面不自动刷新
errorPage	错误页面的URL，用来显示错误。例如： <%@ page errorPage="/error.jsp" %>
isErrorPage	指定是否将当前页面用做JSP错误页面。如果是true，则将当前页面用做JSP错误页面；如果是false，则否。例如： <%@ page isErrorPage="true" %>

2) include指令

include指令用来指出编译JSP页面时要插入的文件名（以相对URL形式），所以被包括的文件内容成为JSP页面的一部分。include指令也可以用来插入为多页面所公用的部分代码。例如，在一个JSP页面中包含头文件（header.html）的方法如下：

```
<%@ include file ="header.html" %>
```

3) taglib（标记库）指令

taglib指令，或称为标记库指令，用来在JSP页面中引入该页面所要用到的标记库资源。例如，在Struts中页面一般要用到标记库struts-html.tld中的标记，因此要在这些页面中引入该标记库，方法如下：

```
<%@ taglib prefix="html" uri=" /WEB-INF/struts-html.tld" %>
```