

CD-ROM ©

Java EE

Web开发实例

精解

■ 郝玉龙 尹建平 编著

清华大学出版社

● 北京交通大学出版社



TP312/2793D

2008

Java EE Web 开发实例精解

郝玉龙 尹建平 编著

实体 EJB

例程 6-1

数据库

例程 1-6

例程 2-5

例程 4-2

清华大学出版社

北京交通大学出版社

责任编辑：李文静
出版发行：清华大学出版社
地址：北京交通大学出版社
邮编：100044
电话：010-62770175
http://www.tup.com.cn
http://press.bjtu.edu.cn

ISBN 978-7-81123-512-8
定价：42.00元
E-mail: press@bjtu.edu.cn

内 容 简 介

本书采用工程实例的形式对 Java EE Web 编程技术进行深入系统的讲解。Java EE Web 应用系统从逻辑上可划分为表现层、业务层和持久层,为了使读者对 Java EE 编程技术获得全面系统的了解,本书以 Java EE Web 应用系统的逻辑架构为主线,通过多个典型工程实例对上述三个层次的开发技术和编程技巧进行讲解,并对同一实例的多个解决方案进行对比分析,切实帮助开发人员提高 Java EE Web 开发水平。

除了对 Java EE Web 编程基本技术的讲解演示外,还针对 Web 应用系统开发中如打印、图表、日志、上传、下载和国际化等常见功能特性的实现,综合运用多种 Java EE 开发技术,提出多种解决方案,并深入讨论分析,对开发人员动手开发实际项目具有借鉴意义。

最后,通过三个综合实例,演示如何利用 EJB 及 Struts、Hibernate、Spring 等常见的 Java EE 架构开发企业 Web 应用系统。

本书适用于对 Java EE 编程有一定了解,但不知道如何开发实际 Web 应用系统的初学者,本书也适合希望掌握 Java EE 编程高级技巧,提高 Java EE 编程能力的中高级初学者。本书也可作为 Java EE 编程技术的培训教材。

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

Java EE Web 开发实例精解 / 郝玉龙,尹建平编著. —北京:清华大学出版社;北京交通大学出版社, 2008.3

ISBN 978-7-81123-215-8

I. J… II. ①郝… ②尹… III. JAVA 语言-程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(2007)第 199587 号

责任编辑:谭文芳

出版发行:清华大学出版社 邮编:100084 电话:010-62776969 <http://www.tup.com.cn>

北京交通大学出版社 邮编:100044 电话:010-51686414 <http://press.bjtu.edu.cn>

印刷者:北京鑫海金澳胶印有限公司

经 销:全国新华书店

开 本:203×280 印张:20.25 字数:650 千字 附光盘 1 张

版 次:2008 年 3 月第 1 版 2008 年 3 月第 1 次印刷

书 号:ISBN 978-7-81123-215-8/TP·396

印 数:1~5 000 册 定价:45.00 元(含光盘)

本书如有质量问题,请向北京交通大学出版社质监组反映。对您的意见和批评,我们表示欢迎和感谢。

投诉电话:010-51686043, 51686008; 传真:010-62225406; E-mail: press@bjtu.edu.cn。

前言

随着网络的日益普及和社会信息化程度的提高,越来越多的软件开发人员需要开发 Web 应用程序。Java EE Web 开发技术以其开放性、灵活性、安全性和技术成熟度,赢得了广大编程爱好者的青睐,并且在目前企业级 Web 应用系统开发领域占领了很大的市场,取得了许多成功的案例。

本书起源

尽管 Java EE 开发人员的队伍在不断壮大,但是 Java EE 开发目前不得不面临这样的困境:经过高校或者培训机构的教育学习,大部分开发人员都能够熟练掌握 Java EE 技术的基本语法,但是在解决实际项目开发中遇到具体问题时仍然感到无从下手,力不从心。另外,Java EE 技术是个充满活力的技术,新的技术规范和标准不断推出,刚刚熟悉了 EJB2 的你,不得不马上了解新推出的 EJB3;才搞懂 JSF 技术内幕,Hibernate3 又在向你招手。如何应对这些层出不穷的新技术带来的挑战和机遇,往往令 Java EE 初学者顾此失彼,心神疲惫。

造成上述现象的原因是由 Java EE Web 应用开发自身的特点决定的。企业系统的开发,本身就是一个复杂的工程,涉及需求分析、数据建模、系统架构、开发技术选型等多个方面,这些都要求开发人员必须在实践中不断学习体会,逐步提高。

Java 技术的开放性决定了 Java EE 编程技术的更新速度较其他开发技术要快得多,这也是 Java 技术自身的优势和魅力所在。所谓“万变不离其宗”,无论 Java Web 应用开发技术如何推陈出新,但 Web 应用的基本逻辑架构是不变的,即整个 Web 应用体系分为表现层、业务层和持久层,不同的 Java 技术分别服务于不同的 Web 应用结构层面,共同实现 Web 应用系统。把握住这个本质,开发人员便可以“以不变应万变”,从容面对新技术的挑战。

本书以 Java EE Web 应用系统的逻辑架构为主线,通过多个典型工程实例分别对 Java Web 应用各个层次上的开发技术和编程技巧进行讲解,并对同一实例的多个解决方案进行对比分析,切实帮助开发人员提高 Java EE Web 开发水平。

本书主要内容

本书各章节的内容安排如下。

第 1 章 构建动态 Web 页面:通过多个示例演示构建 Java Web 应用表现层的各种技术。

第 2 章 服务器与客户端的交互:演示客户端与服务器之间进行交互的基本方法和技术。

第 3 章 管理和维护应用程序的状态:通过多个示例演示 Java EE Web 开发中如何维护应用程序状态信息。

第 4 章 访问企业信息资源:演示 Java Web 应用如何与邮件服务器、关系数据库等外部信息系统交互。

第 5 章 开发 Web 高级功能特性:演示如何实现 Web 应用中常见的打印、图表、上传、下载、日志、国际化等高级功能特性。

第 6 章 应用架构:演示如何基于 Java 重量级架构及 Struts、Hibernate、Spring 等轻量级架构开发企业应用系统。

本书特色

与市场上的同类图书相比，本书具有以下特色：

1. 贴近实战，指导性强

书中所有示例都是作者结合多年教学实践和实际工程项目经验严格挑选的，力求简洁明了，切中要害，使读者能够快速理解并运用到实践中去。全部示例代码都经过严格测试，确保能够正确运行。

2. 思路清晰，结构严谨

书中内容遵循由浅入深、循序渐进的原则，分别按照 Web 应用特定层次（包括表现层、业务层和持久层）实现技术，高级开发特性及系统架构的顺序组织内容，便于开发人员学习理解。对于每个示例，按照学习目的、背景问题、解决方案、知识链接、实现步骤、运行结果、讨论分析的顺序进行讲解，使得初学者能够迅速掌握开发要领。

3. 多案并举，深入分析

避免一般实例图书中仅仅围绕实例实现堆砌代码的弊端，而是从整个 Java EE 编程技术的高处出发，对同一实例提出多个解决方案，并对不同解决方案各自的优缺点进行讨论分析，深入点评，使得开发人员能够尽快领会各种 Java EE 编程技术的实质和适用场景，从而切实提高解决实际问题的能力。

适用读者

本书适用于对 Java EE 编程有一定了解，但不知道如何开发实际 Web 应用系统的初学者，本书也适合希望掌握 Java EE 编程高级技巧，提高 Java EE 编程能力的中高级初学者。本书中的每个实例都按照知识点进行了索引，因此，本书也可以作为开发人员案头的参考手册，随时进行查阅。

使用与反馈

为方便广大读者使用本书学习 Java EE Web 编程，本书附赠的光盘中包含了所有例程的源代码及关于如何使用这些源代码的详细指导和说明。

由于作者水平有限，加之编写时间仓促，书中难免出现错误和不足。对于书中的任何问题，请发 E-mail 至邮箱：haoyulongsd@163.com。

致谢

在本书的编写过程中，得到众多老师的指导和帮助。感谢解放军理工大学的程宝义教授、张宏军教授、吴耀平高工，他们为本书提供了良好的技术支持。感谢周旋、尹鹏飞、周铭、张志杰、姜波，他们参与了本书的部分编辑修改工作，并对本书的内容组织提供了建设性的意见。感谢本书的编辑，北京交通大学出版社的谭文芳老师，没有她的辛勤劳动，本书不可能出版。感谢我的父母，在我写作的过程中给我无微不至的关怀。还要特别感谢我的女友季平，正是她的关心和鼓励，使我能够克服各种困难，确保本书能够顺利完成。

郝玉龙 敬

2008 年 1 月

目 录

第 1 章 构建动态 Web 页面	1
例程 1-1: 利用 Servlet 显示动态日期	1
解决方案	1
讨论与思考	3
知识点索引	3
例程 1-2: 利用 JSP 表达式显示动态格式文本	3
解决方案	3
讨论与思考	5
知识点索引	5
例程 1-3: 利用 JSP 标准标记库和表达式语言显示动态新闻	5
解决方案	5
讨论与思考	8
知识点索引	9
例程 1-4: 利用自定义标记控制网页图片显示	9
解决方案	9
讨论与思考	15
知识点索引	16
例程 1-5: 利用 Applet 在 Web 页面实现动画时钟	17
解决方案	17
讨论与思考	28
知识点索引	28
例程 1-6: 利用 JSP 与 Flash 实现用户登录和注册模块	28
解决方案	29
讨论与思考	37
知识点索引	38
例程 1-7: 利用 JavaScript 脚本实现奥运倒计时日历	38
解决方案	38
讨论与思考	42
知识点索引	43
例程 1-8: 利用 XML、CSS 和 XSL 显示食谱信息	43
解决方案 1: 利用 CSS 显示食谱 XML 文件的内容	43
解决方案 2: 利用 XSL 显示食谱 XML 文件的内容	45
解决方案 3: 利用 xalan 动态绑定 XML 和 XSL	47
讨论与思考	48
知识点索引	49
例程 1-9: 利用标准标记库显示本地化信息	49

解决方案	49
讨论与思考	51
知识点索引	51
例程 1-10: 在 Web 页面中引入版权信息声明	51
解决方案 1: 使用 include 指令在 JSP 中包含版权信息	51
解决方案 2: 使用动作组件在 JSP 中包含版权信息	53
解决方案 3: 使用标准标记<c:import>在 JSP 中包含版权信息	54
解决方案 4: 在 Servlet 响应中包含版权信息	56
讨论与思考	57
知识点索引	58
例程 1-11: 实现带图形验证码的用户登录	58
解决方案	58
讨论与思考	62
知识点索引	62
例程 1-12: 利用 Ajax 实现网上智能订餐	62
解决方案	62
讨论与思考	71
知识点索引	73
本章小结	73
第 2 章 服务器与客户端的交互	74
例程 2-1: 奥运网上问卷调查	74
解决方案	74
讨论与思考	78
知识点索引	78
例程 2-2: 发送 PDF 文件到客户端浏览器	79
解决方案	79
讨论与思考	82
知识点索引	82
例程 2-3: 客户信息显示栏	82
解决方案	82
讨论与思考	83
知识点索引	84
例程 2-4: 获取服务器基本信息	84
解决方案	84
讨论与思考	86
知识点索引	86
例程 2-5: 横幅广告系统	86
解决方案	87
讨论与思考	97
知识点索引	98
例程 2-6: 利用过滤器限制客户端访问	98
解决方案	98

讨论与思考	103
知识点索引	103
例程 2-7: 多组件协作实现用户登录验证	103
解决方案	104
讨论与思考	108
知识点索引	108
本章小结	108
第 3 章 管理和维护应用程序状态	110
例程 3-1: 购物车	110
解决方案 1: 利用隐藏字段实现购物车	111
解决方案 2: 利用 URL 重写实现购物车	115
解决方案 3: 利用 Cookie 实现购物车	119
解决方案 4: 利用 Session 实现购物车	123
讨论与思考	127
知识点索引	128
例程 3-2: 聊天室	128
解决方案	128
讨论与思考	143
知识点索引	144
例程 3-3: 网站计数器	145
解决方案	145
讨论与思考	149
知识点索引	149
本章小结	150
第 4 章 访问企业信息资源	151
例程 4-1: 发送接收 E-mail	151
解决方案	151
讨论与分析	159
知识点索引	159
例程 4-2: 访问数据库	159
解决方案 1: 直接使用 JDBC 驱动访问数据库	159
解决方案 2: 利用 JDBC-ODBC 桥访问数据库	162
解决方案 3: 利用数据源和连接池技术访问数据库	165
讨论与思考	170
知识点索引	171
例程 4-3: 创建基于 XML 的网上论坛	171
解决方案	171
讨论与思考	191
知识点索引	192
例程 4-4: 访问体重检测 Web 服务	192
解决方案	192
讨论与思考	200

801	····· 知识点索引	202
801	····· 本章小结	202
第5章 开发 Web 高级功能特性		203
401	····· 例程 5-1: 在 Web 页面显示统计图表	203
801	····· 解决方案 1: 使用 Applet 显示统计图表	203
801	····· 解决方案 2: 服务器端的图表解决方案	209
801	····· 讨论与思考	214
011	····· 知识点索引	214
011	····· 例程 5-2: 为 Web 应用添加打印功能	215
111	····· 解决方案 1: 利用 iText 组件打印 Web 表格	215
211	····· 解决方案 2: 利用 JavaScript 脚本打印 Web 报表	218
911	····· 讨论与思考	224
821	····· 知识点索引	225
721	····· 例程 5-3: 创建国际化的 Web 应用	225
821	····· 解决方案 1: 为不同地区创建单独的页面资源	225
821	····· 解决方案 2: 利用标准标记库自动绑定地区属性资源	229
821	····· 讨论与思考	233
841	····· 知识点索引	234
441	····· 例程 5-4: 在 Web 应用中实现文件上传	234
241	····· 解决方案 1: 利用流操作实现文件上传	234
241	····· 解决方案 2: 利用 jspSmartUpload 组件实现上传	238
941	····· 解决方案 3: 利用 common-upload 组件实现上传	241
941	····· 讨论与思考	248
021	····· 知识点索引	248
121	····· 例程 5-5: 在 Web 应用中控制文件下载	248
121	····· 解决方案 1: 利用文件流操作实现文件下载	249
121	····· 解决方案 2: 利用 RequestDispatcher 实现文件下载	254
921	····· 讨论与思考	255
921	····· 知识点索引	256
921	····· 例程 5-6: 为 Web 应用添加日志功能	256
921	····· 解决方案 1: 利用服务器自身的日志功能	256
521	····· 解决方案 2: 利用 log4j 实现日志功能	257
201	····· 知识点链接	257
071	····· 讨论与思考	260
171	····· 知识点索引	261
171	····· 本章小结	261
第6章 应用架构		262
191	····· 预备知识: 软件架构基础	262
591	····· 例程 6-1: 利用 EJB 实现公告发布系统	263
591	····· 解决方案	263
591	····· 讨论与思考	275
005	····· 知识点索引	276

例程 6-2: 基于 Struts 构建新闻发布系统	276
解决方案	277
讨论与思考	289
知识点索引	289
例程 6-3: 基于 Struts、Spring 和 Hibernate 构建学生信息管理系统.....	290
解决方案	290
讨论与思考	303
知识点索引	303
本章小结	303
附录 A 开发环境的搭建	306
附录 B 知识点索引	311
参考文献	314

第 1 章 构建动态 Web 页面

条条大路通罗马

——谚语

动态 Web 页面是应用程序与用户进行动态交互的场所，因此构建动态 Web 页面是开发 Web 应用程序最基本的工作之一。本章将通过具体的示例演示 Java EE 体系中构建动态 Web 页面的各种基本方法和技巧，并对这些方法和技巧进行深入剖析和讨论，帮助 Java EE 开发人员彻底理清它们之间的脉络关系，从而为在实际的项目开发中具体采用何种技术构建 Web 页面奠定基础。

例程 1-1：利用 Servlet 显示动态日期

目的

- (1) 演示如何利用 Servlet 技术创建动态 Web 页面；
- (2) 日期函数的使用。

问题

如何将当前日期信息显示到 Web 页面，并且要求如果当前日期是周六、周日，则用红色字体显示日期信息。

解决方案

在 Servlet 负责处理客户端请求的方法(如 `doGet()` 或 `doPost()`)中，参数 `HttpServletResponse` 代表 Servlet 对客户端的响应。`HttpServletResponse` 的 `PrintWriter` 对象代表输出到客户端的显示信息。获取参数 `HttpServletResponse` 的 `PrintWriter` 对象，调用 `PrintWriter` 对象的 `println()` 方法，将要显示的日期信息作为参数，即可将动态日期信息输出到客户端的 Web 页面。

示例代码

本例程的所有示例代码均在 netbeans 的工程 `ServletOut` 内。

程序 1-1: `ShowDate.java`

```
package com.example;

import java.io.*;
import java.net.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.Date;
```

```

public class ShowDate extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse
    response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        Date date=new Date();
        int day=date.getDay();
        String color="black";
        if(day==0|day==6)color="red";
        out.println("<html>");
        out.println("<head>");
        out.println("<title>显示动态日期</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<font color="+color+">");
        out.println("<h1>你好! 今天是" + date + "</h1>");
        out.println("</body>");
        out.println("</html>");
        out.close();
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        processRequest(request, response);
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        processRequest(request, response);
    }

    public String getServletInfo() {
        return "Short description";
    }
}

```

程序说明：首先调用 `HttpServletResponse` 对象的 `setContentType("text/html; charset=UTF-8")` 方法设置 Servlet 对客户端响应的类型和编码格式，然后调用 `HttpServletResponse` 对象的 `getWriter()` 方法获得代表 Servlet 到客户端输出显示内容的 `PrintWriter` 对象，最后调用 `PrintWriter` 对象的 `println()` 方法将要显示的 HTML 页面一行行输出。

运行结果

说明：实例程序的开发运行环境为：Windows XP SP2 + JDK6+NetBeans 5.5+Sun Application Server PE 9+MySQL 5.0。其中 Sun Application Server PE 9 为 NetBeans 5.5 内置绑定的应用服务器。运行环境的安装配置过程可参见附录 A。

注意：如果不特别说明，本书中所有实例的运行环境均采用上述配置。

程序发布成功后，在浏览器地址栏输入“`http://localhost:8080/ServletOut/ShowDate`”，将得到如图 1-1 所示的运行结果。

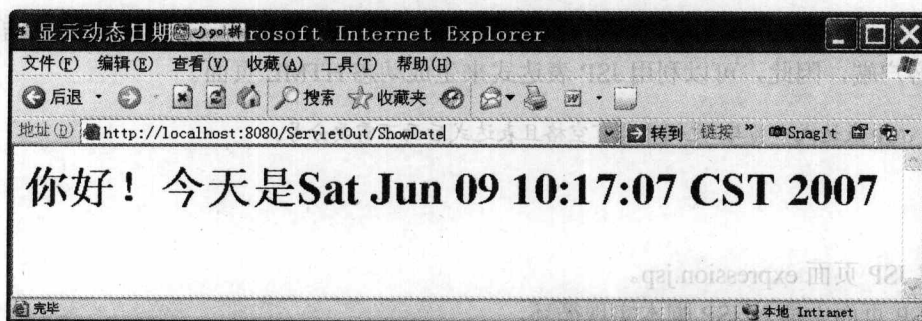


图 1-1 利用 Servlet 显示动态日期

讨论与思考

Servlet 的工作模式是对客户端的请求进行处理，并生成响应结果页面发送到客户端。利用 Servlet 构建动态页面的基本思路就是将 Servlet 对客户端请求的响应动态结果页面输出到客户端。其基本步骤如下：获取代表 Servlet 到客户端输出显示内容的 `PrintWriter` 对象，然后调用 `PrintWriter` 对象的 `println()` 方法将要显示的 HTML 页面逐行输出。

从例程 1-1 不难看出，由于将 HTML 标记大量嵌入到 Java 代码中，大大降低了程序的可读性，同时要求程序开发人员必须熟练掌握 HTML 标记语言的使用。另外，利用 `println()` 方法输出字符串的过程中，对于一些特殊字符（如“\”等）必须采用转义字符标志进行标注。因此页面显示内容一旦复杂，则给程序开发人员带来无尽的痛苦。因此，Servlet 主要完成应用程序后台的业务逻辑处理，它仅仅用来输出比较简单的内容，显示动态页面的任务一般由 JSP 来完成。

知识点索引

Servlet；日期函数。

例程 1-2：利用 JSP 表达式显示动态格式文本

目的

演示使用 JSP 表达式输出动态信息。

问题

如何在 Web 页面显示字体从大到小渐变的文本“Java EE 编程技术实例精解”。

解决方案

通过 JSP 表达式，动态输出定制的 HTML 标记，从而在 Web 页面实现动态格式文本的显示。

知识链接

JSP 表达式是一种特殊的 JSP 脚本。JSP 脚本是 `<% 与 %>` 之间用 Java 语言编写的代码块。如果 JSP

脚本以“=”开头，如`<%=Java 代码%>`形式，称为 JSP 表达式，它将 Java 代码的计算结果转换为字符串形式输出到客户端，因此，可以利用 JSP 表达式来生成动态 HTML 页面。

注意：JSP 表达式中 % 与 = 之间不能有空格且表达式后面不需要分号。

实现步骤

- (1) 创建 JSP 页面 `expression.jsp`。
- (2) 在 JSP 页面中插入 JSP 脚本实现循环。
- (3) 在循环中插入 HTML 标记来输出文本“Java EE 开发实例精解”。
- (4) 利用 JSP 表达式动态生成的 HTML 标记。

示例代码

本例程的所有示例代码均在 netbeans 的工程 ShowDate 内。

程序 1-2: `expression.jsp`

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<html>
<head>
<title>在 JSP 页面显示动态格式文本</title>
</head>
<body bgcolor="#FFFFFF">
<% for(int i=1;i<=5;i++){%>
<h<%=i%>>Java EE 开发实例精解</h<%=i%>><br>
<%}%>
</body>
</html>
```

运行结果

程序发布成功后，在浏览器地址栏输入“`http://localhost:8080/ ShowDate /expression.jsp`”，将得到如图 1-2 所示的运行画面。

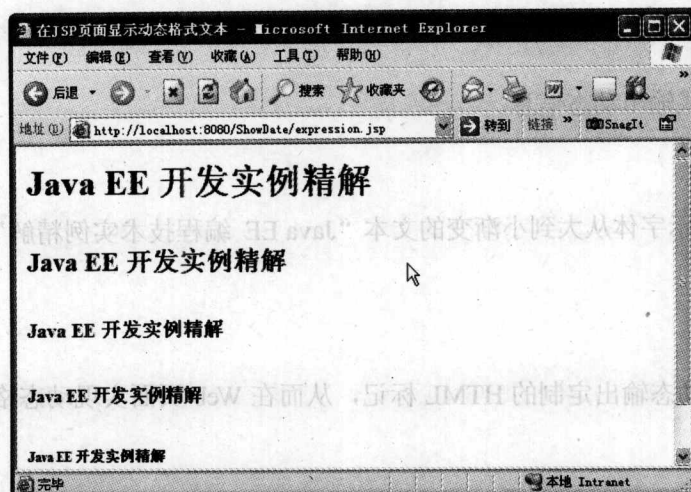


图 1-2 利用 JSP 表达式显示动态文本

讨论与思考

通过 JSP 脚本进行动态计算, 利用输出表达式将动态变量信息输出到页面生成动态标记是 JSP 页面编程的基本思路。JSP 脚本输出的动态内容可以是 HTML 页面中的任何内容, 包括 HTML 标记的名称、属性及标记体内容等信息。

由于 JSP 页面中可以直接嵌入 HTML 代码, 这样就避免了 Servlet 编程中大量的 `println()` 语句。但是使用 JSP 脚本生成动态内容仅适合于简单的动态 Web 页面实现, 它的缺点在于将 Java 代码与 HTML 混杂在一起, 如果页面的逻辑过于复杂, 则页面内容过于杂乱, 不利于提高页面开发效率, 也不便于开发过程中前端美工设计人员与后台业务逻辑人员的合理分工。最重要的, 这种开发模式将表现层的逻辑与业务逻辑紧密耦合在一起, 违背了 Java EE 多层体系架构的基本模式。

知识点索引

JSP 脚本; JSP 表达式。

例程 1-3: 利用 JSP 标准标记库和表达式语言显示动态新闻

目的

- (1) 演示 JSP 标准标记库的使用;
- (2) 演示表达式语言的使用;
- (3) JSP 与 JavaBean 的结合。

问题

网上新闻是网站常见的组件。每一条新闻由分类信息、标题和内容三部分组成。那么如何创建一个网上新闻组件实现在 Web 页面显示动态新闻信息?

解决方案

创建一个 JavaBean 来代表发布的新闻信息, 利用标准标记库的 `<c:out>` 将新闻内容输出到页面。

知识链接

1. JSP 标准标记库

JSP 标准标记库 (JSP Standard Tag Library, JSTL) 是一个实现 Web 应用程序中常见的通用功能的定制标记库集, 这些功能包括迭代和条件判断、数据管理格式化、XML 操作及数据库访问。通过为上述典型表现层任务提供标准实现, JSTL 使 JSP 开发人员可以专注于特定于应用程序的开发需求。

2. JSP 表达式语言

自 JSP2.0 以后, JSP 提供了一种新特性——JSP 表达式语言 (Expression Language, EL)。表达式语言允许开发人员使用简单的语法方便地从 JSP 页面访问数据, 而不需要使用 JSP 脚本或者 Java 表达式。

表达式语言定义了一些隐含对象以支持开发人员访问需要的应用程序数据。表达式语言定义的隐含

对象如表 1-1 所示。

表 1-1 表达式语言中的隐含对象

隐含对象	内 容
applicationScope	应用程序范围内的 scoped 变量组成的集合
cookie	所有 cookie 组成的集合
header	HTTP 请求头部, 字符串
headerValues	HTTP 请求头部, 字符串集合
initParam	全部应用程序参数名组成的集合
pageContext	当前页面的 javax.servlet.jsp. PageContext 对象
pageScope	页面范围内所有对象的集合
param	所有请求参数字符串组成的集合
paramValues	所有作为字符串集合的请求参数
requestScope	所有请求范围的对象的集合
sessionScope	所有会话范围的对象的集合

表达式语言支持“.”运算符来访问对象的属性信息, 同时还支持算术运算符、关系运算符和逻辑运算符, 以完成大多数的数据处理操作。此外, 它还提供了一个用于测试一个对象是否为空的特殊运算符。

实现步骤

(1) 将 JSTL 相关的类库 (jstl.jar 和 standard.jar) 添加到项目的类路径中。具体操作为: 在项目文件夹 jstlsample 下选中子文件夹【库】, 单击右键, 在弹出的快捷菜单中选择【添加库】选项, 弹出的对话框如图 1-3 所示。在对话框的列表中选择【JSTL1.1】, 单击【添加库】按钮, 则成功地将 JSTL 相关的类库添加到项目的类路径中。

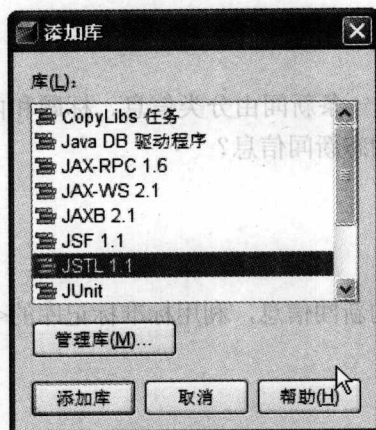


图 1-3 为 Web 工程添加 JSTL 库

- (2) 生成代表新闻内容的 JavaBean News。
- (3) 生成显示动态信息内容 JSP 页面 news.jsp。
- (4) 将下面的代码行添加到 news.jsp 的第一行以便在页面中使用标准标记库中的标记。

```
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
```

- (5) 在 news.jsp 中利用标准标记库和表达式语言显示动态新闻内容。

示例代码

本例程的所有示例代码均在 netbeans 的工程 jstlsample 内。

程序 1-3: News.java

```
package com.example;

import java.util.HashMap;
import java.util.Map;
public class News extends Object implements java.io.Serializable {
    private String topic;
    private HashMap values;
    public News() {
        topic="热烈欢迎进入 Java 开发人员阵营!";
        values = new HashMap();
        values.put("netbeans", "netbeans6.0 正式发布!");
        values.put("Java", "Java 开发成员超过 3000 万!");
        values.put("北京", "奥运会准备一切就绪!");
    }
    public String getTopic(){
        return topic;
    }
    public Map getValues() {
        return this.values;
    }
}
```

程序说明：程序用来实现一个代表动态新闻的 JavaBean。其中动态新闻的内容采用一个 HashMap 对象来表示。

程序 1-4: news.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<jsp:useBean id="mynews" scope="page" class="com.example.News" />
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>通过表达式语言显示动态新闻</title>
</head>
<body>
<c:if test="${param.sayHello}">
<!-- Let's welcome the user ${param.name} -->
Hello ${param.name}!
</c:if>
<h2>${mynews.topic}</h2>
<font color="blue">
<h3>
```