



JWorld@TW技术论坛版主
Java权威技术顾问与专业讲师

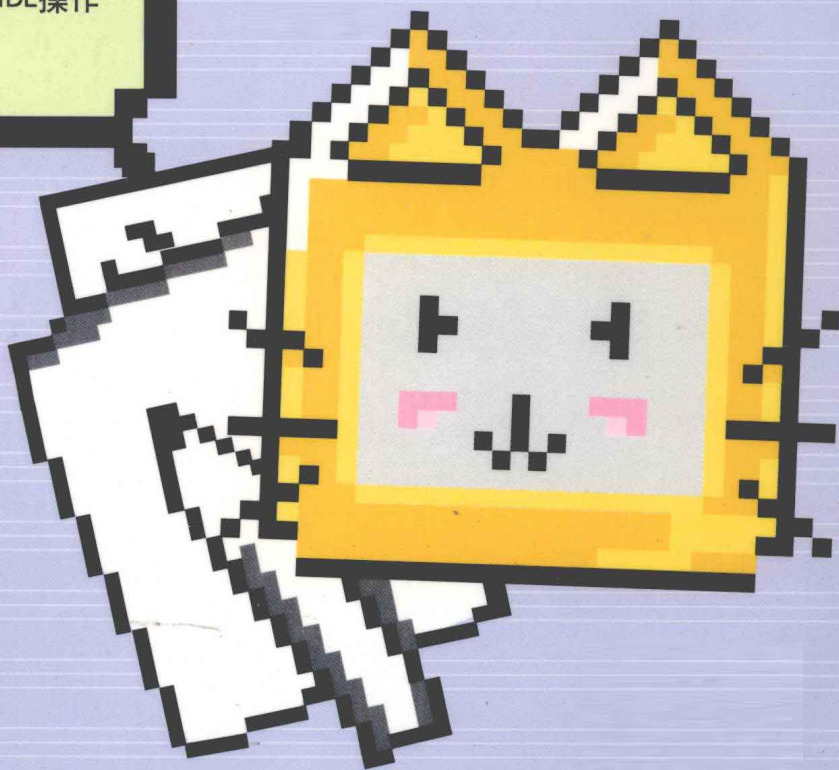
全新改版!

JSP&Servlet

学习笔记(第2版)

- 分享作者学习JSP心得
- 涵盖SCWCD考试范围
- Servlet 3.0新功能介绍
- 全新综合练习/微博开发
- 提供Lab操作案例与IDE操作教学视频

林信良 编著





JWorld@TW技术论坛版主
Java权威技术顾问与专业讲师

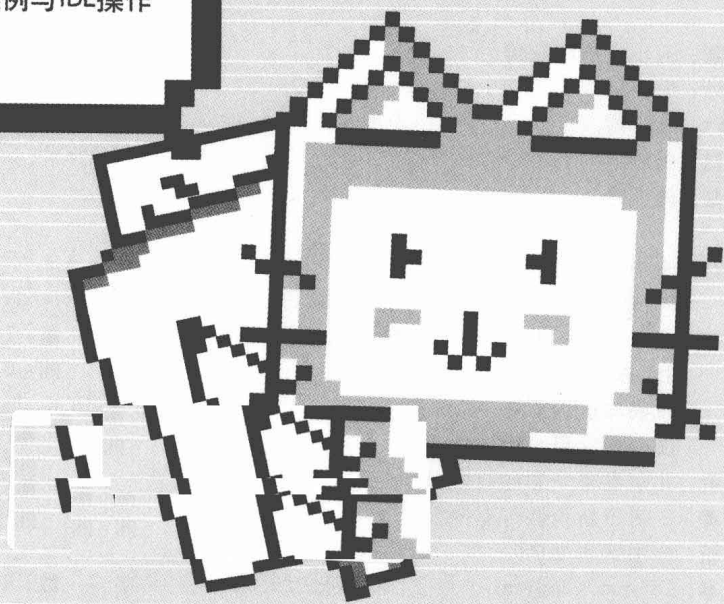
全新改版!

JSP&Servlet

学习笔记(第2版)

- 分享作者学习JSP心得
- 涵盖SCWCD考试范围
- Servlet 3.0新功能介绍
- 全新综合练习/微博开发
- 提供Lab操作案例与IDE操作教学视频

林信良 编著



清华大学出版社

北 京

内 容 简 介

本书是作者多年来教学实践经验的总结,汇集了学生在学习 JSP & Servlet 或认证考试时遇到的概念、操作、应用等各种问题及解决方案。

本书针对 Servlet 3.0 的新功能全面改版,无论章节架构还是范例程序代码,都做了全面更新。书中详细介绍了 Servlet/JSP 与 Web 容器之间的关系,必要时从 Tomcat 源代码分析,了解 Servlet/JSP 如何与容器互动。本书还涵盖了文本处理、图片验证、自动登录、验证过滤器、压缩处理、JSTL 应用与操作等各种实用范例。

本书在讲解的过程中,以“微博”项目贯穿全书,随着每一章的讲述都在适当的时候将 JSP & Servlet 技术应用于“微博”程序之中,使读者能够了解完整的应用程序构建方法。

本书适合 JSP & Servlet 初学者以及广大 JSP & Servlet 技术应用人员。

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 13701121933

图书在版编目(CIP)数据

JSP & Servlet 学习笔记/林信良 编著. —2 版. —北京:清华大学出版社,2012.5

ISBN 978-7-302-28366-9

I. ①J… II. ①林… III. ①JAVA 语言—程序设计 ②JAVA 语言—网页—程序设计

IV. ①TP312 ②TP393.092

中国版本图书馆 CIP 数据核字(2012)第 049933 号

责任编辑:王 定 胡花蕾

封面设计:牛艳敏

版式设计:康 博

责任校对:成凤进

责任印制:杨 艳

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社总机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印 刷 者:北京鑫丰华彩印有限公司

装 订 者:三河市新茂装订有限公司

经 销:全国新华书店

开 本:185mm×260mm 印 张:29.5 字 数:664 千字
附光盘 1 张

版 次:2010 年 4 月第 1 版 2012 年 5 月第 2 版 印 次:2012 年 5 月第 1 次印刷

印 数:1~5000

定 价:58.00 元

产品编号:044950-01

第2版序

“序”应该表达些什么？写一本书的动机？写一本书的过程？写完一本书的感想？

在本书第1版手稿完成后，思考着如何写序的那几天，在整理旧书时从一本书中掉出了一张车票，于是我写了一张车票引发一连串回忆的故事。在本书第2版手稿完成后，思考着如何写序的这几天，我回顾改版的这段漫长日子，想着一脚踏入陌生领域、探索一切未知的过程。

现在的你，也许在某个领域有擅长的事务，有没有想过，或许哪天，你会接触另一个完全未知的世界，到时候，你会怎么办？

我在信息领域的知识，大多都是自学而来，对于信息领域知识的搜寻、过滤、验证与实践，自认为颇有心得，改版过程中，乍然面对一切毫无所知的世界，也曾一度乱了手脚。某个下午带着慌乱的心路过了书局，突然心里有了答案：“我一直认为收集与过滤是我最大的能力，不用在这个时候，那要用在什么时候？”

你有没有听过类似的事呢？某人拥有高学历，却在生完小孩之后，毅然决然在家带小孩，某人在某领域拥有很好的经历，却在大家觉得他即将迈向巅峰时，投入另一个领域重新开始。像这类的情况，旁人通常都会为他们可惜。

我面对着完全未知的世界，开始发挥大量阅读的能力，极尽可能地寻找相关的书籍，在网络上搜寻各种相关资料，逐步勾勒出这个世界应有的方向，就如同当初从电机转换入信息，一切从未知开始累积，一切从头开始建立基础，既然是初学者，那就一切从头开始建构。

高学历带小孩不好吗？也许是自愿或被迫这么做，但如果可以发挥出高学历下该有的学习态度，好好学习如何让小孩子健康、快乐成长，那不也是件好事吗？放弃原有领域的经历不好吗？把建立原有领域经历的方式应用在新领域经验的建立，因此而有所成就的案例也不在少数！

一切都是动心转念之间，无论如何，保有一颗初学者的心，保有一颗赤子之心，放下熟悉领域拥有的一切，重新出发，方向就会逐步建立，所有的基础，后续的成就，就交由时间慢慢验证。

林信良

2011年5月26日

第 1 版序

在完成本书之前，意外翻出了这张车票，94.08.15 从高雄到台北的座位证！一时之间还想起不起这张车票是哪里来的，反倒是我老婆提醒了我，不过我却想起了更久之前的事情……

大学时代参加的社团是社会服务团，寒、暑假时会到一些地方举办营队，在学期中，即将参与营队的队友们必须负责各自的课程、准备教材、设计教具、验收教案等，出营队时则上台实行课程。

除了社团之外，自己平常也爱写些东西。大学时代正值 WWW 兴起之时，自己学会如何写 HTML，也常将学计算机时的心得写下来放到 Web 上，像如何安装 Apache、CGI 留言版之类的，说来写作的习惯应该是从那时养成的。

在大学最后一年考完硕士研究生入学考试之后，我在 BBS 的 Job 版上发现了几个短期打工需求，有一天接到一个电话，问我想不想写书。虽然主题只是网页制作，但第一次要写完整的一本书，合同载明页数必须有 400 页以上，着实有很大的压力，甚至还因此失眠了好几次，所幸在当兵前夕完成了这本书，成了我的第一本著作。事后在市面上发现，这本书重印了 4 次，心里还蛮感到安慰的。

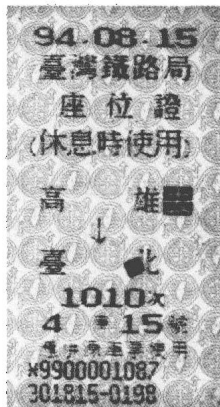
当兵期间所属的单位是学校，平时除了连队勤务或卫哨之外，所做的事就是协助教官编写教材、教案、上课担任助教等。退伍后的第一份工作是在高雄，公司的业务之一是出版计算机图书，因为早有写作及出版图书的经验，自然也在公司的名义之下写了几本书。

2003 年 3 月底，开始将一些东西以“良葛格学习笔记”的名称放在网络上，随着时间的累积，伴随着网络传播的力量，越来越多人的知道了这个网站的存在，我也在网络上结交了许多朋友，并因此得以在 Javaworld@TW 前站长林上杰(Browser)先生的介绍下，认识了碁峰编辑江佳慧(Novia)小姐，出版了第二本有个人名号的书籍。

想到这里，发现在我过去的经验中，怎么都跟上台、写作、课程有关？还有一点不知道是否也有相关，我岳父岳母也都是老师……

这就是看到 94.08.15 从高雄到台北的座位证时，突然涌出的一连串回忆……

1994 年 8 月 15 日是什么日子？隔两天就是“2005 Java TWO 社区大会”！这张车票是为了参加 Java TWO 大会而买的。这是我第一次参加 Java TWO 大会，目的之一是



想看看许多网络上认识的但未曾谋面的朋友，还有一个原因是暮峰也在大会上设摊，其中有卖我的书，想去看看反应如何……在大会上，碰到了王森(Moli)先生，他跟我说：“想要请你帮忙写个教材……”，不过那时场面很混乱，反正就是一堆人哈啦来哈啦去的，话题很难继续，直到后来出现了 Moli 先生想加我的 MSN，哈啦过后，才确定这件事是真的！

之后又因为一连串的因缘机会，开始了我江湖卖艺……呃……讲课的日子！时光匆匆，岁月如梭……转眼来到了 2009 年 3 月，Novia 小姐问我有没有新的写作计划？我想了一下，这些日子以来，许许多多的授课经验积累了不少的想法，也了解了不少学员在实际学习时所遇到的问题，不如写下来吧！而这些写下来的东西就成了你眼前的这本书(篇幅有限，这本书只针对 Servlet / JSP)！

我不太知道人有没有宿命，但回首时总会发现许多的巧合，过去的种种经验，好像是在为了将来的某个事件而准备似的。当然，你也可以说，这是因为回忆是选择性地挑选拼凑而成的。无论如何，这些事情过去总得发生过，未来的你才有的拼凑。

一张车票引发了一连串回忆，也终于知道要怎么写这本书的序了，这本书就是这么来的……

林信良

2009 年 5 月 26 日

导 读

这份导读可以让你了解如何使用本书。

字体

本书内文中与代码相关的文字，都用等宽字体来加以呈现，以与一般名词相区别。例如，JSP 是一般名词，而 `HttpServlet` 类为代码相关文字，使用了等宽字体。

程序范例



本书中许多范例都使用完整的程序实现来展现，如果是用以下方式示范程序代码：

FirstServlet HelloServlet.java

```
package cc.openhome;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/hello.view")
public class HelloServlet extends HttpServlet { ← ❶ 继承 HttpServlet
    @Override
    protected void doGet(HttpServletRequest request, ← ❷ 重新定义 doGet()
        HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8"); ← ❸ 设定响应内容类型
        PrintWriter out = response.getWriter(); ← ❹ 取得回应输出对象
        String name = request.getParameter("name"); ← ❺ 取得请求参数

        out.println("<html>");
        out.println("<head>");
        out.println("<title>Hello Servlet</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1> Hello! " + name + " !</h1>"); ← ❻ 跟用户说 Hello!
        out.println("</body>");
```



```
        out.println("</html>");  
        out.close();  
    }  
}
```

范例开始的左边名称为 `FirstServlet`，表示可以在本书配套光盘的 `samples` 文件夹中查找相应章节目录，即可找到对应的 `FirstServlet` 项目，而右边名称为 `HelloServlet.java`，表示可以在项目中找到 `HelloServlet.java` 文件。如果代码中出现标号与提示文字，表示后续的内文中会有对应于标号及提示的更详细说明。

原则上，建议每个项目范例都亲自动手编写，但如果由于教学时间或实现时间上的考量，本书有建议进行的练习，如果在范例开始前有个  图示，表示建议动手实践，而且在本书配套光盘的 `labs` 文件夹中会有练习项目的基础，可以导入项目后，完成项目中遗漏或必须补齐的代码或设定。

如果文中使用以下的代码形成呈现，则表示它是一个完整的程序内容，不是项目的一部分，主要用来展现一个完整的文件如何编写：

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>  
<html>  
<head>  
    <title>SimpleJSP</title>  
</head>  
<body>  
    <h1><%= new java.util.Date() %></h1>  
</body>  
</html>
```

如果文中使用以下的代码形式呈现，则表示它是一个程序代码片段，主要展现程序编写时需要特别注意的片段：

```
// 略 ...  
public void _jspService(HttpServletRequest request,  
                        HttpServletResponse response)  
    throws java.io.IOException, ServletException {  
    // 略...  
    try {  
        response.setContentType("text/html;charset=UTF-8");  
        //略...  
        out = pageContext.getOut();  
        // 略...  
    } catch (Throwable t) {  
        // 略 ...  
    } finally {  
        // 略 ...  
    }  
}
```

由于受书籍页面宽度的限制，有些过长的程序代码可能会在一行容纳不下，不得不隔行表示，此时会使用箭头符号表示两行实际上是必须连接在一起的。例如：

JDBCDemo context.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<Context antiJARLocking="true" path="/JDBCDemo">
  <Resource name="jdbc/demo"
    auth="Container" type="javax.sql.DataSource"
    maxActive="100" maxIdle="30" maxWait="10000" username="root"
    password="123456" driverClassName="com.mysql.jdbc.Driver"
    url="jdbc:mysql://localhost:3306/demo?
    useUnicode=true&characterEncoding=UTF8"/>
</Context>
```

在上面的程序代码片段中，在 `url` 属性的设定中，完整的设置其实是 `jdbc:mysql://localhost:3306/demo?useUnicode=true&characterEncoding=UTF8` 当中不可以中断。

操作步骤

本书将 IDE 设定的相关操作步骤，也作为练习的一部分，你会看到如下的操作步骤说明：



(1) 选择 File/New/Dynamic Web Project 命令，在出现的 New Dynamic Web Project 对话框的 Project name 文本框中输入 FirstServlet。

(2) 确定 Target runtime 为刚才设置的 Apache Tomcat v7.0，单击 Finish 按钮。

(3) 展开新建项目中的 Java Resources 节点，在 src 上单击鼠标右键，从弹出的快捷菜单中选择 New/Servlet 命令。

(4) 在弹出的 Create Servlet 对话框的 Java package 文本框中输入 cc.openhome，在 Class name 文本框中输入 HelloServlet，单击 Next 按钮。

(5) 选择 URL mappings 中的 HelloServlet，单击右边的 Edit 按钮，将 Pattern 改为 /hello.view 后，单击 OK 按钮。

(6) 单击 Create Servlet 对话框中的 Finish 按钮。

如果操作步骤旁有个  图示，表示光盘的 videos 文件夹中对应的章节文件夹有操作步骤的录像，可打开观看以更了解实际操作过程。

特殊段落

在本书中会出现以下特殊段落：

提示 >>> 针对课程中提到的概念，提供一些额外的资源或思考方向，暂时忽略这些提示对课程进行并没有影响，但有时间的话，针对这些提示多作阅读、思考或讨论是有帮助的。

注意>>> 针对课程中提到的概念，以特殊段落方式特别呈现出必须注意的一些使用方式、陷阱或避开问题的方法，看到这个特殊段落时请集中精神阅读。

综合练习

本书以“微博”项目作为范例贯穿全书，随着每一章的进行，都会在适当的时候将新学习到的技术，应用至“微博”程序之中并作适当的修改，以了解完整的应用程序基本上是如何建构出来的。

附 录

本书配套光盘中包括本书所有的范例，提供 Eclipse 范例项目，附录 A 说明如何使用这些范例项目，本书也说明如何在 Web 应用程序中整合数据库，范例中使用的数据库为 MySQL；附录 B 包括了 MySQL 的入门简介。

关于认证

本书涵盖了 Oracle Certified Professional, Java Platform, Enterprise Edition JavaServer Pages and Servlet Developer 考试范围，也就是原 Sun Certified Web Component Developer (SCWCD)，不过第 9 章整合数据库与第 11 章 JavaMail 入门不在考试范围，只是为了 Web 应用程序相关技术范围完整性而作介绍。

关于 Java 认证介绍，建议直接参考 Oracle University 网站上的认证介绍：

http://education.oracle.com/pls/web_prod-plq-dad/db_pages.getpage?page_id=140

每章最后的“重点复习”是针对该章的重要提示，可作为考前复习时使用。每章都会有“课后练习”，与认证相关的是选择题，分为单选、复选题两种形式，实训题是与每个章节相关的程序练习。

联系作者

若有勘误反馈等相关书籍问题，可通过网站与作者联系：

<http://openhome.cc>

目 录

Chapter 1 Web 应用程序简介	1	2.3.2 Web 目录结构	40
1.1 Web 应用程序基础知识	2	2.3.3 使用 web-fragment.xml	41
1.1.1 关于 HTML	2	2.4 重点复习	44
1.1.2 URL、URN 与 URI	3	2.5 课后练习	45
1.1.3 关于 HTTP	5	Chapter 3 请求与响应	47
1.1.4 有关 URL 编码	8	3.1 从容器到 HttpServlet	48
1.1.5 动态网页与静态网页	10	3.1.1 Web 容器做了什么	48
1.2 Servlet/JSP 简介	12	3.1.2 doXXX()方法	50
1.2.1 何谓 Web 容器	13	3.2 关于 HttpServletRequest	53
1.2.2 Servlet 与 JSP 的关系	15	3.2.1 处理请求参数与标头	53
1.2.3 关于 MVC/Model 2	18	3.2.2 请求参数编码处理	55
1.2.4 Java EE 简介	20	3.2.3 getReader()、getInputStream() 读取 Body 内容	58
1.3 重点复习	21	3.2.4 getPart()、getParts()取得上传 文件	63
1.4 课后练习	22	3.2.5 使用 RequestDispatcher 调派 请求	67
Chapter 2 编写与设置 Servlet	25	3.3 关于 HttpServletResponse	73
2.1 第一个 Servlet	26	3.3.1 设置响应标头、缓冲区	73
2.1.1 准备开发环境	26	3.3.2 使用 getWriter()输出字符	74
2.1.2 第一个 Servlet 程序	28	3.3.3 使用 getOutputStream()输出 二进制字符	78
2.2 在 HelloServlet 之后	31	3.3.4 使用 sendRedirect()、 sendError()	79
2.2.1 关于 HttpServlet	31	3.4 综合练习	81
2.2.2 使用 @WebServlet	33	3.4.1 微博应用程序功能概述	81
2.2.3 使用 web.xml	34	3.4.2 实现会员注册功能	83
2.2.4 文件组织与部署	36		
2.3 进阶部署设置	37		
2.3.1 URL 模式设置	38		

3.4.3 实现会员登录功能	86	5.3.1 过滤器的概念	141
3.5 重点复习	88	5.3.2 实现与设置过滤器	142
3.6 课后练习	89	5.3.3 请求封装器	147
Chapter 4 会话管理	93	5.3.4 响应封装器	151
4.1 会话管理基本原理	94	5.4 异步处理	155
4.1.1 使用隐藏域	94	5.4.1 AsyncContext 简介	155
4.1.2 使用 Cookie	97	5.4.2 模拟服务器推播	157
4.1.3 使用 URL 重写	101	5.4.3 更多 AsyncContext 细节	160
4.2 HttpSession 会话管理	103	5.5 综合练习	162
4.2.1 使用 HttpSession	104	5.5.1 创建 UserService	162
4.2.2 HttpSession 会话管理 原理	107	5.5.2 设置过滤器	167
4.2.3 HttpSession 与 URL 重写	110	5.5.3 重构微博	168
4.3 综合练习	112	5.6 重点复习	173
4.3.1 修改微博应用程序	112	5.7 课后练习	175
4.3.2 新增与删除信息	114	Chapter 6 使用 JSP	179
4.3.3 会员网页显示信息	116	6.1 从 JSP 到 Servlet	180
4.4 重点复习	119	6.1.1 JSP 生命周期	180
4.5 课后练习	119	6.1.2 Servlet 至 JSP 的简单 转换	183
Chapter 5 Servlet 进阶 API、过滤器 与监听器	121	6.1.3 指示元素	187
5.1 Servlet 进阶 API	122	6.1.4 声明、Scriptlet 与表达式 元素	191
5.1.1 Servlet、ServletConfig 与 GenericServlet	122	6.1.5 注释元素	195
5.1.2 使用 ServletConfig	124	6.1.6 隐式对象	195
5.1.3 使用 ServletContext	127	6.1.7 错误处理	198
5.2 应用程序事件、监听器	130	6.2 标准标签	203
5.2.1 ServletContext 事件、 监听器	130	6.2.1 <jsp:include>、<jsp:forward> 标签	203
5.2.2 HttpSession 事件、 监听器	133	6.2.2 <jsp:useBean>、<jsp:setProperty> 与<jsp:getProperty>简介	204
5.2.3 HttpServletRequest 事件、 监听器	139	6.2.3 深入<jsp:useBean>、 <jsp:setProperty>与 <jsp:getProperty>	206
5.3 过滤器	140	6.2.4 谈谈 Model 1	210

6.2.5 XML 格式标签	212	7.5 函数标签库	274
6.3 表达式语言(EL)	213	7.6 综合练习	276
6.3.1 EL 简介	213	7.6.1 修改 register.jsp	276
6.3.2 使用 EL 取得属性	215	7.6.2 修改 member.jsp	277
6.3.3 EL 隐式对象	218	7.6.3 修改 user.jsp	278
6.3.4 EL 运算符	219	7.7 重点复习	279
6.3.5 自定义 EL 函数	220	7.8 课后练习	281
6.4 综合练习	222	Chapter 8 自定义标签	285
6.4.1 改用 JSP 实现视图	222	8.1 Tag File 自定义标签	286
6.4.2 重构 UserService 与 member.jsp	226	8.1.1 Tag File 简介	286
6.4.3 创建 register.jsp、index.jsp、 user.jsp	232	8.1.2 处理标签属性与 Body	289
6.5 重点复习	237	8.1.3 TLD 文件	291
6.6 课后练习	238	8.2 Simple Tag 自定义标签	293
Chapter 7 使用 JSTL	241	8.2.1 Simple Tag 简介	293
7.1 JSTL 简介	242	8.2.2 了解 API 架构与生命 周期	295
7.2 核心标签库	244	8.2.3 处理标签属性与 Body	298
7.2.1 流程处理标签	244	8.2.4 与父标签沟通	302
7.2.2 错误处理标签	247	8.2.5 TLD 文件	306
7.2.3 网页导入、重定向、URL 处理标签	248	8.3 Tag 自定义标签	306
7.2.4 属性处理与输出标签	250	8.3.1 Tag 简介	307
7.3 I18N 兼容格式标签库	252	8.3.2 了解架构与生命周期	308
7.3.1 I18N 基础	253	8.3.3 重复执行标签 Body	310
7.3.2 信息标签	256	8.3.4 处理 Body 运行结果	312
7.3.3 地区标签	258	8.3.5 与父标签沟通	315
7.3.4 格式标签	263	8.4 综合练习	318
7.4 XML 标签库	267	8.4.1 实现首页最新信息	318
7.4.1 XPath、XSLT 基础	267	8.4.2 自定义 Blahs 标签	321
7.4.2 解析、设置与输出标签	270	8.5 重点复习	324
7.4.3 流程处理标签	271	8.6 课后练习	326
7.4.4 文件转换标签	272	Chapter 9 整合数据库	329
		9.1 JDBC 入门	330
		9.1.1 JDBC 简介	330

9.1.2 连接数据库	335	10.1.8 标注访问控制	409
9.1.3 使用 Statement、 ResultSet	341	10.2 综合练习	411
9.1.4 使用 PreparedStatement、 CallableStatement	346	10.2.1 使用容器窗体验证	412
9.2 JDBC 进阶	349	10.2.2 设置 DataSourceRealm	413
9.2.1 使用 DataSource 取得 连接	349	10.3 重点复习	416
9.2.2 使用 ResultSet 滚动、 更新数据	353	10.4 课后练习	417
9.2.3 批次更新	355	Chapter 11 JavaMail 入门	419
9.2.4 Blob 与 Clob	356	11.1 使用 JavaMail	420
9.2.5 事务简介	364	11.1.1 传送纯文字邮件	420
9.2.6 metadata 简介	371	11.1.2 发送多重内容邮件	423
9.2.7 RowSet 简介	374	11.2 综合练习	426
9.3 使用 SQL 标签库	379	11.2.1 实现取回密码功能	427
9.3.1 数据源、查询标签	379	11.2.2 接收重送密码请求	431
9.3.2 更新、参数、事务标签	380	11.3 重点复习	433
9.4 综合练习	382	11.4 课后练习	433
9.4.1 重构 / 使用 DAO	382	Chapter 12 从模式到框架	435
9.4.2 使用 JDBC 实现 DAO	385	12.1 认识设计模式	436
9.4.3 设置 JNDI 部署描述	389	12.1.1 Template Method 模式 (Gof 设计模式)	436
9.5 重点复习	390	12.1.2 Intercepting Filter 模式 (Java EE 设计模式)	437
9.6 课后练习	391	12.1.3 Model-View-Controller 模式(架构模式)	437
Chapter 10 Web 容器安全管理	393	12.2 重构、模式与框架	439
10.1 了解与实现 Web 容器安全 管理	394	12.2.1 Business Delegate 模式	439
10.1.1 Java EE 安全基本概念	394	12.2.2 Service Locator 模式	439
10.1.2 声明式基本身份验证	397	12.2.3 Transfer Object 模式	440
10.1.3 容器基本身份验证原理	400	12.2.4 Front Controller 模式	441
10.1.4 声明式窗体验证	402	12.2.5 库与框架	442
10.1.5 容器窗体验证原理	403	12.3 重点复习	443
10.1.6 使用 HTTPS 保护数据	404	12.4 课后练习	444
10.1.7 程式式安全管理	407	Appendix A 如何使用本书项目	445
		Appendix B MySQL 入门	449

Web 应用程序简介

Chapter

1

学习目标:

- 认识 HTTP 基本特性
- 了解 GET、POST 使用时机
- 了解何为 URL/URI 编码
- 认识 Web 容器角色及其重要性
- 初步了解 Servlet 与 JSP 的关系
- 初步认识 MVC/Model 2

1.1 Web 应用程序基础知识

在正式学习 Servlet/JSP 相关技术之前,要先花点时间了解一些 Web 应用程序基础知识,这些知识虽然基础,但却很重要。在我这几年的教学中,发现有些学员并不具备这些基础,或者忽略了这些基础中的一些细节,如 HTML(HyperText Markup Language)、HTTP(HyperText Transfer Protocol)、URL(Uniform Resource Locator)甚至文字编码的问题等。

当然,谈这些东西并不是要求你成为这几个名词的专家,而是在以后学习 Servlet/JSP 相关技术时,若有这些基础,就能真正理解相关技术背后的原理,而不会沦落到死背 API(Application Programming Interface)的窘境。

这里先从 HTML 开始……

1.1.1 关于 HTML

本书介绍的 Web 应用程序,是由客户端(Client)与服务器端(Server)两个部分组成的,客户端基本是浏览器(Browser),服务器端则是 HTTP 服务器,浏览器会请求服务器上放置的文件或资源。对本书来说,服务器上的文件或资源必须产生 HTML。

HTML 是以标签(Tag)的方式来定义文件结构。例如,一个简单的 HTML 范例如下所示:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>HTML 范例文件</title>
  </head>
  <body>
    哈啰! 请输入...<br><br>
    <form method="get" action="sample.do" name="sample">
      </>名称: <input type="text" name="name"><br><br>
      <button>送出</button>
    </form>
  </body>
</html>
```

HTML 文件的标签通常是成对的,有开头标签与结尾标签(但少数标签例外)。例如,整份 HTML 文件的定义编写在<html>与</html>标签之间。在文件开始呈现之前,浏览器必须先处理编写在<head>与</head>标签之间的元素。显示在浏览器窗口上的标题,就是编写在 HTML 中的<title>与</title>标签之间的内容。

浏览器若要针对文件内容绘制画面与定义行为,相关的信息是定义在<body>标签之中。例如,
告诉浏览器换下一行后再继续绘制文件内容,范例文件中有个代表图

片的标签,告诉浏览器要读取指定的图形文件并绘制在画面上。**HTML** 标签可以拥有属性(attribute),定义该标签的额外信息,如图片来源(src 属性)。**<form>**标签定义了一个窗体,窗体用来让用户填写一些将送至服务器的信息,其中还使用了**<input>**标签分别定义了一个输入字段及发送按钮。

简而言之,当浏览器从服务器取得这份 **HTML** 文件之后,就可以按照其中的结构等信息进行画面的绘制。图 1.1 所示为大致的 **HTML** 标签与对应的画面呈现。

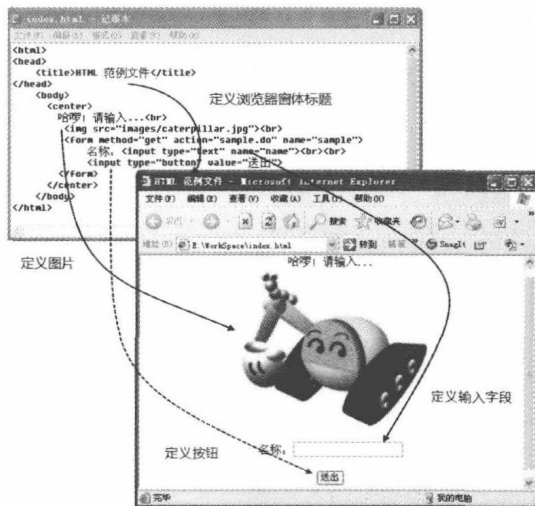


图 1.1 浏览器按 **HTML** 的结构等信息进行画面绘制

以上描述是对你的一种测试,如果连以上的基本 **HTML** 都不甚了解,建议先寻找 **HTML** 相关的文件或书签作大致的了解, **w3schools**(<http://w3schools.com>)的 **HTML Tutorial** 是不错的快速入门文件,足以应付阅读本书所需的 **HTML** 基础,网址是:

<http://www.w3schools.com/html/default.asp>

1.1.2 URL、URN 与 URI

既然 **Web** 应用程序的文件等资源是放在服务器上,而服务器是因特网(Internet)上的主机,当然必须要有个方式,告诉浏览器到哪里取得文件等资源。通常会听到有人这么说:“你要指定 **URL**”,偶而会听到有人说:“你要指定 **URI**”。那么到底什么是 **URL**、**URI**? 甚至你还听过 **URN**。首先,三个名词都是缩写,其全名分别为:

- **URL**: Uniform Resource Locator
- **URN**: Uniform Resource Name
- **URI**: Uniform Resource Identifier

从历史的角度来看, **URL** 的标准最先出现,早期 **U** 代表 **Universal**(万用),标准化之后代表着 **Uniform**(统一)。正如名称所指出, **URL** 的主要目的,是以文字方式来说明因特网上的资源如何取得。一般而言, **URL** 的主要格式为: