实例源代码 · 程序软件包 · 行业规范

JSP 应用与开发技术(第2版)

马建红　李占波　主　编
韩　颖　王瑞娟　姬莉霞　副主编

清华大学出版社

JSP 应用与开发技术

(第 2 版)

主　编　马建红　李占波
副主编　韩　颖　王瑞娟　姬莉霞

清华大学出版社
北　京

内 容 简 介

JSP(Java Server Pages)是由 Sun Microsystems 公司(已被甲骨文公司收购)倡导的、许多公司参与建立的一种动态网页技术标准。JSP 被赋予了 Java 技术的强大功能，能够为用户提供功能强大的技术支持，同时，JSP 继承了 Java 的优势，可以建立安全的、跨平台的动态网站。

本书详细讲解了 JSP 的基本语法和 Web 程序设计方法。全书共 16 章，可分为 7 个部分。第 1～3 章是第 1 部分，介绍了 JSP 应用技术的前导知识和开发环境的搭建；第 4～6 章是第 2 部分，详细讲解了 JSP 技术的基本语法；第 7 章和第 8 章是第 3 部分，是 JSP 应用开发的进阶，讲述了 JavaBean、表单处理以及文件的操作；第 9～11 章是第 4 部分，以 MySQL 数据库为例详细讲解了 JSP 中使用数据库的操作；第 12 章和第 13 章是第 5 部分，详细讲述了 Servlet 技术；第 14 章和第 15 章是第 6 部分，介绍了表达式语言(EL)、标准标签库(JSTL)及自定义标签库；第 16 章是第 7 部分，通过实例讲述了 Web 开发的实际应用。另外，本书还给出了 6 个实验，以指导读者进行上机操作。

本书所附光盘中收录了相关实例运行的开源程序、实例源代码，读者可参照光盘说明进行调试运行。此外，读者还可以通过 www.tupwk.com.cn 下载本书的电子课件。

本书语言简洁，内容丰富，既可作为 JSP 初学者的入门教材，也可作为高等院校相关专业的教材和辅导用书，而且对 JSP 开发人员的自学也具有较高的参考价值。

图书在版编目(CIP)数据

JSP应用与开发技术/马建红，李占波 主编. —2版. —北京：清华大学出版社，2014

ISBN 978-7-302-35687-5

Ⅰ. ①J… Ⅱ. ①马… ②李… Ⅲ. ①JAVA语言－主页制作工具 Ⅳ. ①TP312 ②TP393.092

中国版本图书馆 CIP 数据核字(2014)第 180832 号

责任编辑：王　定
封面设计：杜丽雅
版式设计：孔祥峰
责任校对：成凤进
责任印制：

出版发行：清华大学出版社　　**地　　址：**北京清华大学学研大厦
http://www.tup.com.cn　　**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：
装 订 者：
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张：**37.75　　**字　　数：**942 千字
(附光盘 1 张)
版　　次：2011 年 2 月第 1 版　2014 年 9 月第 2 版　　**印　　次：**2014 年 9 月第 1 次印刷
印　　数：13201~17200
定　　价：59.80 元

产品编号：

前言

PREFACE

JSP(Java Server Pages)是目前动态网站开发技术中最典型的一种，它继承了 Java 语言的优势，是一种与平台无关的开发技术，而 Java 技术也赋予了 JSP 为用户提供强大功能的技术支持。JSP 实现了动态页面与静态页面的分离，脱离了硬件平台的束缚，提高了执行效率而成为因特网上的主流开发技术，越来越受到编程者的关注和喜爱。

JSP 虽然综合性地包括了 Java 和 HTML 这两类语法，但不能通过简单地使用 JSP，让它集显示、业务逻辑和流程控制于一身，因为用这种方式开发出来的 Web 应用程序是非常难以维护的。所以对 JSP 使用观念的建立，以及 JavaBean、数据库、Servlet 等技术的了解运用是利用 JSP 开发复杂的商业级网站的重点。为了让读者在学习的过程中能够彻底掌握相关概念，除了对基本语法的介绍外，本书同时也将重点集中在面向对象的观点和 JSP 程序架构方面。

《JSP 应用与开发技术(第 2 版)》仍保持了第 1 版实用、新颖和经验总结的特点，从基本的语法和规范入手，以经验为后盾，以实用为目标，以实例为导向，以实践为指导，深入浅出地讲解了 JSP 开发中的种种问题，以精简的内容介绍了 JSP 的语法和 Servlet、JDBC、MySQL、EL、标签库、MVC 等技术。本书每一章节的实例读者都可以直接使用，实例讲解过程条理清晰、循序渐进，符合程序设计的自然思路，读者学完一个章节，也就相应地掌握了相关的 JSP 开发思想和技术，并且通过几个较完整的综合实例，让读者对 JSP+Servlet+JavaBean+JDBC 这样的 Web 架构有一个整体认识。和第 1 版相比，本书不仅采用了最新版本的开发工具，还添加了 JSP 和 MySQL 最新版本中出现的新特性，调整了相关实例，补充了大量的习题，进一步充实了上机实验部分，尽可能地展现出本书的实用性。另外，原有的章节全部进行了细致的修订，部分内容还进行了重大改动。

本书共 16 章，可以分成 7 个部分。

第 1 部分介绍了 JSP 编程基础与环境配置，包括 3 章：第 1 章，JSP 概述；第 2 章，JSP 动态网页设计基础，和第 1 版相比使用了 HTML 最新版本 HTML5，新增了 CSS 知识；第 3 章，JSP 的开发和运行环境，和第 1 版相比增加了最新版的 Eclipse Java EE 集成开发环境和 MyEclipse 开发环境。

第 2 部分介绍了 JSP 应用开发基础，包括 3 章：第 4 章，JSP 基本语法；第 5 章，JSP 内置对象；第 6 章，使用 Cookie 记录信息。

第 3 部分介绍了 JSP 应用开发进阶，包括 2 章：第 7 章，JavaBean 和表单处理；第 8 章，JSP 中的文件操作。这部分主要介绍了 JavaBean 分离表示与实现，使用 JSP 处理 HTML 表单，使用 Java 类库里的 I/O 类，开发具备文件存取功能的网页程序。和第 1 版不同的是，更换了容

易使用的 JspSmartUpload 包实现文件的上传。

第 4 部分介绍了 JSP 数据库编程基础，包括 3 章：第 9 章，数据库操作基础；第 10 章，应用 JDBC 进行数据库开发；第 11 章，JSP 与 JavaBean 应用实例。和第 1 版一样介绍了 MySQL 数据库的无界面基本操作，但不同的是，第 2 版中 MySQL 的版本升级到 mysql-5.1.25-rc-win32，并且添加了和版本相匹配的 MySQL 带界面的基本操作，以 MySQL 数据库为例详细讲解了 JSP 中使用数据库的操作，包括 JDBC、连接池、分页处理等技术；更换了第 1 版第 11 章中简单的留言板实例，改为一个完整的电商系统，还对 Model1 模式的应用给予详细的讲解，项目模块功能更强大。

第 5 部分讲述了 Servlet 技术，包括 2 章：第 12 章，Servlet 基础；第 13 章，使用 Servlet 过滤器和监听器。第 2 版中重点调整了监听器接口，增加了新实例。

第 6 部分讲述了标签库的应用，包括 2 章：第 14 章，表达式语言(EL)和标准标签库(JSTL)；第 15 章，自定义标签库。第 2 版中对每部分内容都进行了细致的修订，增加了 EL，更加完善了标签库的知识体系结构。

第 7 部分即第 16 章，Web 应用开发实践。和第 1 版不同的是第 2 版中更新了实例，运用 MVC 技术，采用三层架构，并采用了自定义标签和 EL 等技术展示了一个门户网站综合实例，使读者对这些技术的使用有更深刻的了解。

本书采用由浅入深、循序渐进的方法，全面、系统地介绍了 JSP 程序设计的原理、方法和技术，还提供了大量的 JSP 应用开发实例，给出了相应的实用技巧、操作步骤及优化思路，使读者可以很快地进行实际开发。每章的最后还提供了习题，让读者能够检验自己对各章内容的学习、消化程度，并巩固所学的知识。

本书由马建红、李占波任主编，韩颖、王瑞娟、姬莉霞任副主编，参与编写的人员还有张晗、卫权岗、宰光军、何保锋、尹辉、程凤娟、秦兴桥、赵玉娟、王永皎等。此外，林楠、李翠霞、史苇航在整理材料方面给予了编者很大的帮助，在此对他们表示衷心的感谢。

由于时间仓促，加之水平有限，书中不足之处在所难免，敬请读者批评指正。

编　者

目录

CONTENTS

第 1 章 JSP 概述 …… 1
1.1 软件编程体系简介 …… 2
1.1.1 C/S 结构编程体系 …… 2
1.1.2 B/S 结构编程体系 …… 3
1.2 企业应用开发架构 …… 3
1.2.1 两层架构 …… 3
1.2.2 三层架构 …… 3
1.2.3 N 层架构 …… 4
1.2.4 开发架构比较 …… 4
1.3 JSP 概述 …… 4
1.3.1 什么是 JSP …… 5
1.3.2 JSP 技术原理 …… 6
1.3.3 JSP 和其他动态网站开发技术 …… 7
1.3.4 J2EE 简介 …… 10
1.4 JSP 知识体系及学习之路 …… 13
1.4.1 JSP 知识体系 …… 13
1.4.2 JSP 程序员学习路径 …… 13
1.5 小结 …… 15
1.6 习题 …… 15
1.6.1 选择题 …… 15
1.6.2 判断题 …… 16
1.6.3 填空题 …… 16
1.6.4 简答题 …… 16
1.6.5 拓展实践题 …… 16
第 2 章 JSP 动态网页设计基础 …… 17
2.1 HTML 技术 …… 18
2.1.1 HTML5 基本结构 …… 18
2.1.2 HTML 常用标签 …… 20
2.1.3 表单 …… 28
2.1.4 XML 与 XHTML …… 36
2.2 CSS 技术 …… 38
2.2.1 CSS 基本语法 …… 38
2.2.2 在 HTML 文档中使用 CSS 的方法 …… 40
2.2.3 常用 CSS 属性 …… 42
2.3 JavaScript 技术 …… 50
2.3.1 JavaScript 语法 …… 50
2.3.2 JavaScript 使用方式 …… 51
2.3.3 JavaScript 代码实例 …… 52
2.4 小结 …… 57
2.5 习题 …… 57
2.5.1 选择题 …… 57
2.5.2 判断题 …… 58
2.5.3 填空题 …… 58
2.5.4 简答题 …… 58
第 3 章 JSP 的开发和运行环境 …… 59
3.1 JSP 的开发和应用平台介绍 …… 60
3.1.1 Caucho 公司的 Resin 平台 …… 60
3.1.2 Apache公司的Tomcat平台 …… 60
3.1.3 BEA公司的WebLogic平台 …… 61
3.1.4 IBM WebSphere Application Server 平台 …… 61
3.2 Eclipse Java EE集成开发环境 …… 62
3.2.1 安装和配置 JDK …… 62
3.2.2 Tomcat 服务器 …… 64
3.2.3 Eclipse Java EE开发环境搭建 …… 68
3.3 Eclipse 集成开发环境配置 …… 73

3.4 MyEclipse 开发环境 75
3.4.1 MyEclipse 简介与下载 76
3.4.2 MyEclipse 安装与使用 76
3.5 小结 82
3.6 习题 82
3.6.1 选择题 82
3.6.2 判断题 83
3.6.3 填空题 83
3.6.4 简答题 83
第 4 章 JSP 基本语法 85
4.1 JSP 文件的结构 86
4.1.1 创建第一个 JSP 文件 86
4.1.2 分析 JSP 文件的组成元素 87
4.2 JSP 的脚本元素 88
4.2.1 隐藏注释(Hidden Comment) 88
4.2.2 HTML 注释 89
4.2.3 声明语句(Declaration) 91
4.2.4 脚本段(Scriptlets) 94
4.2.5 表达式(Expression) 95
4.3 JSP 指令元素 96
4.3.1 page 指令 96
4.3.2 include 指令 101
4.3.3 taglib 指令 103
4.4 JSP 动作元素 104
4.4.1 <jsp:include> 104
4.4.2 <jsp:forward> 107
4.4.3 <jsp:param> 110
4.4.4 <jsp:useBean>、<jsp:setProperty>和<jsp:getProperty>动作 111
4.4.5 <jsp:plugin> 120
4.5 小结 122
4.6 习题 123
4.6.1 选择题 123
4.6.2 判断题 124
4.6.3 填空题 125
4.6.4 简答题 125
4.6.5 编程题 125
第 5 章 JSP 内置对象 127
5.1 JSP 内置对象概述 128
5.2 request 对象 129
5.2.1 request 对象常用方法 129
5.2.2 request 对象应用实例 130
5.3 response 对象 137
5.3.1 response 对象常用方法 137
5.3.2 response 对象应用实例 138
5.4 out 对象 146
5.4.1 out 对象方法成员与数据输出 146
5.4.2 缓冲区操作 147
5.4.3 out 对象应用实例 148
5.5 session 对象 151
5.5.1 session 的概念 151
5.5.2 session 对象的 ID 152
5.5.3 session 的有效期限 152
5.5.4 访问 session 中的数据 152
5.5.5 其他 session 对象的常用方法 153
5.5.6 session 对象应用实例 154
5.6 application 内置对象 160
5.6.1 存取 application 中的数据 161
5.6.2 使用 application 对象取得信息 162
5.6.3 application 对象应用实例 162
5.7 其他 JSP 内置对象 164
5.7.1 pageContext 对象 164
5.7.2 config 对象 166
5.7.3 page 对象 169
5.7.4 exception 对象 169
5.8 小结 169
5.9 习题 170
5.9.1 选择题 170
5.9.2 判断题 172

5.9.3 填空题 172
5.9.4 简答题 172
5.9.5 编程题 172
第 6 章 使用 Cookie 记录信息 175
6.1 Cookie 的概念和特性 176
6.1.1 什么是 Cookie 176
6.1.2 Cookie 的常见用途 177
6.1.3 对 Cookie 进行适当设置 178
6.2 在 JSP 中使用 Cookie 179
6.2.1 创建 Cookie 179
6.2.2 读写 Cookie 179
6.2.3 Cookie 中的主要方法 180
6.2.4 几个操作 Cookie 的常用方法 181
6.3 Cookie 对象的应用实例 183
6.4 Cookie 的安全问题 192
6.5 小结 193
6.6 习题 194
6.6.1 选择题 194
6.6.2 判断题 195
6.6.3 填空题 196
6.6.4 简答题 196
6.6.5 编程题 196
第 7 章 JavaBean 和表单处理 197
7.1 非 MVC 模式(Model1) 198
7.1.1 Model1 的特点 198
7.1.2 Model1 的应用范围 199
7.2 MVC 编程模式(Model2) 199
7.2.1 什么是 MVC 模式 199
7.2.2 MVC 模式在 Web 编程中的应用 200
7.3 剖析 JavaBean 201
7.3.1 什么是 JavaBean 202
7.3.2 JavaBean 的特征 203
7.3.3 创建一个 JavaBean 204
7.4 在 JSP 中使用 JavaBean 206
7.4.1 调用 JavaBean 206
7.4.2 访问 JavaBean 属性 206
7.4.3 设置 JavaBean 属性 206
7.4.4 JavaBean 的生命周期 207
7.4.5 类型自动转换规则 211
7.5 使用JavaBean处理表单数据 211
7.5.1 JSP 处理与 form 相关的常用标签简单实例 212
7.5.2 设置中文编码 217
7.5.3 Post 与 Get 的差异 218
7.6 小结 219
7.7 习题 219
7.7.1 选择题 219
7.7.2 判断题 220
7.7.3 填空题 221
7.7.4 简答题 221
7.7.5 编程题 221
第 8 章 JSP 中的文件操作 223
8.1 数据流和 File 类 224
8.1.1 数据流 224
8.1.2 File 类 224
8.2 读写文本文件 227
8.3 文件的浏览 229
8.4 创建和删除目录 230
8.5 文件的上传和下载 231
8.6 使用jspSmartUpload上传包 235
8.7 小结 238
8.8 习题 238
8.8.1 选择题 238
8.8.2 判断题 239
8.8.3 填空题 239
8.8.4 简答题 239
8.8.5 编程题 239
第 9 章 数据库操作基础 241
9.1 关系数据库及 SQL 242
9.2 在Windows上安装MySQL 243

9.3 MySQL 的常用操作…… 247
9.3.1 设置环境变量…… 247
9.3.2 启动 MySQL 数据库…… 248
9.3.3 连接 MySQL…… 248
9.3.4 退出 MySQL…… 248
9.3.5 增加用户…… 249
9.3.6 删除授权…… 249
9.3.7 备份数据库…… 250
9.3.8 恢复数据库…… 250
9.3.9 备份表…… 250
9.3.10 恢复表…… 251
9.3.11 查看、创建、删除和选择数据库命令…… 251
9.3.12 导入命令…… 252
9.4 常用查询的例子…… 252
9.4.1 查询时间…… 253
9.4.2 查询当前用户…… 253
9.4.3 查询数据库版本…… 253
9.4.4 查询当前使用的数据库…… 253
9.4.5 使用AUTO_INCREMENT…… 254
9.4.6 列的最大值…… 254
9.4.7 拥有某个字段的组间最大值的行…… 256
9.4.8 使用用户变量…… 256
9.5 MySQL GUI Tools…… 256
9.5.1 MySQL GUI Tools 安装…… 256
9.5.2 MySQL GUI Tools 基本使用方法…… 258
9.6 小结…… 260
9.7 习题…… 261
9.7.1 选择题…… 261
9.7.2 填空题…… 261
9.7.3 编程题…… 262

第 10 章 应用 JDBC 进行数据库开发…… 263
10.1 JDBC 概述…… 264
10.1.1 JDBC 的用途…… 264
10.1.2 JDBC 的典型用法…… 264
10.1.3 JDBC 体系结构…… 265
10.1.4 驱动器类型…… 265
10.1.5 安装驱动器…… 267
10.2 JDBC 连接数据库的方法…… 267
10.3 使用 JDBC 操作数据库…… 269
10.3.1 使用 JDBC 访问数据库的过程…… 269
10.3.2 使用 Statement 执行 SQL 语句…… 272
10.3.3 PreparedStatement 接口…… 281
10.3.4 CallableStatement 对象…… 285
10.3.5 使用 ResultSet 处理结果集…… 287
10.4 Java 与 SQL 的数据类型转换…… 292
10.5 使用 JDBC 连接不同的数据库…… 294
10.5.1 连接 Oracle 数据库…… 294
10.5.2 连接 DB2 数据库…… 294
10.5.3 连接 SQL Server 数据库…… 294
10.5.4 连接 Sybase 数据库…… 295
10.5.5 连接 Access 数据库…… 295
10.6 连接池…… 295
10.6.1 连接池的实现原理…… 295
10.6.2 在 Tomcat 上配置数据源与连接池…… 296
10.6.3 配置连接池时需要注意的问题…… 299
10.7 存取二进制文件…… 299
10.7.1 图像文件存取到数据库的过程…… 300
10.7.2 声音文件存取到数据库的过程…… 304

10.7.3 视频文件存取到数据库的过程 309
10.8 实现分页显示 313
10.8.1 分页显示技术的优劣比较 313
10.8.2 分页显示的 JavaBean 实现 314
10.9 小结 322
10.10 习题 322
10.10.1 选择题 322
10.10.2 判断题 323
10.10.3 填空题 323
10.10.4 简答题 324
10.10.5 编程题 324
第 11 章 JSP 与 JavaBean 应用实例 325
11.1 需求和设计 326
11.1.1 功能介绍 326
11.1.2 文件结构 327
11.1.3 数据库设计 328
11.2 使用 JavaBean 封装数据库的访问 330
11.3 项目页面实现 335
11.3.1 用户模块设计与实现 335
11.3.2 管理员模块设计与实现 353
11.4 小结 366
第 12 章 Servlet 基础 367
12.1 Servlet 介绍 368
12.1.1 什么是 Servlet 368
12.1.2 Servlet 技术特点 369
12.1.3 JSP 与 Servlet 的关系 369
12.1.4 Servlet 的工作原理 370
12.1.5 Servlet 常用接口和类 371
12.2 开发部署一个简单的 Servlet 372
12.2.1 创建 Servlet 文件 373
12.2.2 Servlet 的配置文件 375
12.3 Servlet 实现相关的接口和类 377
12.3.1 GenericServlet 378
12.3.2 HttpServlet 378
12.3.3 Servlet 实现相关实例 379
12.4 Servlet 请求和响应相关 383
12.4.1 HttpServletRequest 接口 383
12.4.2 HttpServletResponse接口 385
12.4.3 Servlet 请求和响应相关实例 386
12.5 Servlet 配置相关 388
12.5.1 ServletConfig 接口 389
12.5.2 获取 Servlet 配置信息的例子 389
12.6 Servlet 中的会话追踪 394
12.6.1 HttpSession 接口 394
12.6.2 HttpSession 应用实例 396
12.7 Servlet 上下文 398
12.7.1 ServletContext 接口 398
12.7.2 ServletContext 接口的应用实例 399
12.8 Servlet 协作 401
12.8.1 RequestDispatcher 401
12.8.2 forward()控制页面跳转 401
12.8.3 include()控制页面包含 403
12.9 Servlet 异常处理 404
12.9.1 声明式异常处理 404
12.9.2 程序式异常处理 408
12.10 Servlet 应用实例 411
12.11 小结 421
12.12 习题 421
12.12.1 选择题 421
12.12.2 判断题 423
12.12.3 填空题 423
12.12.4 简答题 424

12.12.5 编程题 424

第 13 章 使用 Servlet 过滤器和监听器 425

13.1 过滤器在 Web 开发中的应用 426

13.1.1 过滤器概述 426

13.1.2 Filter API 427

13.1.3 Filter 接口 427

13.1.4 FilterConfig 接口 428

13.1.5 FilterChain 接口 428

13.1.6 编写过滤器类 429

13.1.7 过滤器的部署 430

13.1.8 对请求数据进行处理的过滤器 434

13.1.9 对响应内容进行压缩的过滤器 440

13.2 Servlet 监听器 445

13.2.1 监听器接口 445

13.2.2 ServletRequestListener 接口 446

13.2.3 ServletRequestAttributeListener 接口 448

13.2.4 ServletContextListener 接口 448

13.2.5 ServletContextAttributeListener 接口 449

13.2.6 HttpSessionAttributeListener 接口 452

13.2.7 HttpSessionBindingListener 接口 454

13.3 小结 460

13.4 习题 461

13.4.1 选择题 461

13.4.2 判断题 462

13.4.3 填空题 462

13.4.4 简答题 462

13.4.5 编程题 463

第 14 章 JSTL 标准标签库 465

14.1 EL 表达式语言 466

14.1.1 EL 与 EL 隐含对象 466

14.1.2 在 EL 中访问 JSP 隐含对象的 get×××( )方法 471

14.1.3 用 EL 访问 JavaBean 中的属性 472

14.2 JSTL 标签库简介 473

14.3 设置 JSTL 运行环境 473

14.3.1 JSTL 的安装 474

14.3.2 JSTL 应用示例 475

14.4 使用核心标签 475

14.4.1 表达式操作 476

14.4.2 建立 URL 480

14.4.3 条件控制 484

14.4.4 迭代-运行循环 486

14.5 使用 JSTL 的数据库标签 490

14.5.1 指定数据源 490

14.5.2 进行查询或更新操作 491

14.5.3 对返回的结果进行处理 494

14.5.4 其他SQL标签库的标签 495

14.6 i18n 与国际化 496

14.6.1 国际化设置标签 496

14.6.2 消息标签库 497

14.6.3 数字、日期格式化 499

14.7 函数标签 503

14.8 小结 503

14.9 习题 504

14.9.1 选择题 504

14.9.2 判断题 505

14.9.3 填空题 506

14.9.4 简答题 506

14.9.5 编程题 506

第 15 章 自定义标签库 507

15.1 自定义标签体系介绍 508

15.1.1 标签的形式…… 509
15.1.2 标签类相关接口和类…… 509
15.1.3 标签库描述文件…… 514
15.1.4 在 Web 部署描述符中引入标签库文件…… 515
15.1.5 在页面中使用标签 …… 516
15.1.6 标签在 Web 页面中的作用…… 517
15.2 传统标签的开发…… 517
15.2.1 带属性标签的开发 …… 517
15.2.2 带 Body 标签的开发 …… 521
15.2.3 嵌套标签的开发…… 524
15.2.4 迭代标签的开发…… 527
15.3 Simple 标签的开发…… 531
15.3.1 SimpleTag 接口…… 531
15.3.2 Simple标签的开发示例 …… 532
15.4 小结…… 533
15.5 习题…… 534
15.5.1 选择题…… 534
15.5.2 判断题…… 534
15.5.3 填空题…… 534
15.5.4 简答题…… 534
15.5.5 编程题…… 534
第 16 章 Web 应用开发实践 ……537
16.1 信息发布平台…… 538
16.2 系统需求分析…… 538
16.3 系统功能结构…… 539
16.4 系统功能描述…… 539
16.4.1 游客用户浏览模块…… 539
16.4.2 管理员登录模块…… 541
16.4.3 管理员管理模块…… 542
16.5 数据库设计…… 545
16.5.1 数据库逻辑结构设计 …… 545
16.5.2 数据库表的设计…… 545
16.5.3 数据库相关脚本…… 546
16.6 系统实现 …… 548
16.6.1 模块公用类 …… 548
16.6.2 JavaBean …… 553
16.6.3 Servlet …… 557
16.6.4 自定义标签 …… 561
16.6.5 前台界面的实现…… 567
16.6.6 后台管理页面的实现 …… 571
16.7 小结…… 576
16.8 习题…… 576
附录 实验……577
实验一 JSP 应用开发基础 …… 577
实验二 JSP 应用开发基础 …… 580
实验三 JSP 应用开发进阶 …… 585
实验四 JSP 数据库编程基础 …… 587
实验五 Servlet 技术实验…… 589
实验六 Web 应用开发 …… 591

第1章

JSP 概 述

本章主要对 JSP 技术进行概要介绍，并给出了一些学习 JSP 的建议。为了让读者在开始学习之前能对 JSP 技术有一个清晰与完整的概念，本章首先介绍了软件编程体系、企业应用开发架构，然后通过编写一个简单的 JSP 页面实例让读者对 JSP 技术有一个直观的感性认识。接着通过介绍 JSP 的技术原理以及与其他主流动态网页技术的比较，进一步了解 JSP 技术是一种功能强大、可以实现跨平台操作的动态网页开发技术。本章对 JSP 知识体系的剖析有助于读者学习和掌握 JSP 知识体系中的各个模块，对 JSP 技术有一个总体性的了解。

本章学习目标

- ◎ 了解软件编程体系
- ◎ 了解企业应用开发架构
- ◎ 掌握 JSP 的基本概念
- ◎ 掌握 JSP 的知识体系
- ◎ 了解 JSP 的学习之路

1.1 软件编程体系简介

目前，在应用开发领域中主要分为两大编程体系，一种是基于浏览器的 B/S(Browser/Server)结构，另一种是 C/S(Client/Server)结构。应用程序开发体系如图 1-1 所示。

开发基于 C/S 结构项目，传统的开发环境有 Visual Basic、Visual C++以及 Delphi 等，随着 Java 体系以及.NET 体系的普及，目前更流行.NET 编程体系和 Java 编程体系。

开发基于 B/S 结构项目，目前主要采用三种服务器端语言：JSP(Java Server Pages)、PHP(Hypertext Preprocessor)和 ASP.NET。这三种语言构成三种常用应用开发组合：JSP+Oracle 体系、PHP+MySQL 体系以及 ASP.NET+SQL Server 体系。

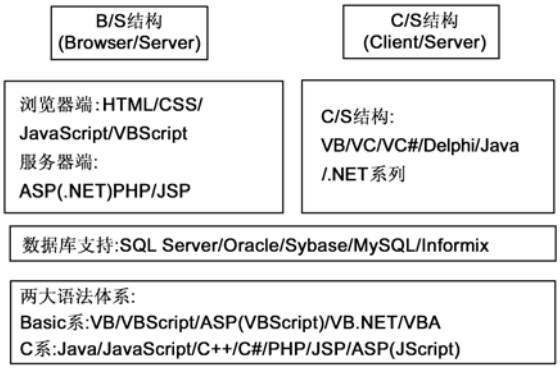

图 1-1　应用程序开发体系

软件开发涉及的语言很多，学习起来也是有规律可循的。图 1-1 最下面的方框将目前常用的开发语言分成两大语系：Basic 语系和 C 语系，语系中的语言所有的流程控制语句都是一样的，常用的函数也大同小异。所以只要精通其中任何一门语言，该语系中的其他语言也就比较容易掌握了。

1.1.1 C/S 结构编程体系

2000 年以前，C/S 结构占据开发领域的主流，随着 B/S 结构的发展，C/S 结构主流地位已经逐步被 B/S 结构取代，目前在整个开发领域中，C/S 结构的应用大概能占到 40%的份额。C/S 结构应用程序最大特点是在每个用户端需要安装程序，所有用户端程序和中心服务器进行信息交互；这种结构优点是用户端程序一致，比较方便控制，服务器端和用户本地的数据很容易进行交互，通信速度比较快；缺点是每个用户都需要安装客户端，比较繁琐，而且不能很好地跨操作系统平台。

C/S 结构通常适用于具有固定的用户端或者少量的用户端，并且是对安全性要求比较高的应用。比如银行信息管理系统、邮局信息管理系统和飞机票火车票售票系统等。

传统的 C/S 结构通常使用 PowerBuilder、Delphi、Visual Basic、Visual C++、JBuilder 作为开发环境，使用 SQL Server、Oracle 或者 DB2 作为数据库支持。随着时间的发展、版本的更新，主流的 C/S 开发环境开始向.NET 和 Java 两大主流体系转变，目前大部分 C/S 结构应用都使用 VB.NET、VC#.NET 以及 Java 开发，其中，VB.NET 和 VC#.NET 只是描述的语言不一样，设计思想和开发环境全部一样，因此只要掌握其中一个，就可以满足开发要求了。

VB.NET 是从 Visual Basic 发展而来的，Visual Basic 曾经具有开发领域世界第一的程序员数量，因此非常多的 C/S 应用采用 VB.NET 开发环境。

1.1.2 B/S 结构编程体系

B/S 结构编程语言分成浏览器端编程语言和服务器端编程语言。

浏览器端包括 HTML (HyperText Markup Language，超文本标记语言)、CSS(Cascading Style Sheets，层叠样式表单)、JavaScript 语言和 VBScript 语言。所谓浏览器端编程语言就是这些语言都是被浏览器解释执行的。HTML 和 CSS 是由浏览器解释的，JavaScript 语言和 VBScript 语言也是在浏览器上执行的。

为了实现一些复杂的操作，比如连接数据库、操作文件等，需要使用服务器端编程语言。目前主要是 3P(ASP.NET、JSP 和 PHP)技术。ASP.NET 是微软公司推出的，在这三种语言中是用得最为广泛的一种。JSP 是 SUN 公司(已被甲骨文公司收购)推出的 J2EE(Java 2 Enterprise Edition，Java2 企业版)核心技术中重要的一种。PHP 在 1999 年的下半年和 2000 年应用得非常广泛，因为 Linux+PHP+MySQL(一种中小型数据库管理系统)构成全免费的而且非常稳定的应用平台，这三种语言是目前应用开发体系的主流。

数据库支持是必须的，目前应用领域的数据库系统全部采用关系型数据库系统(Relation Database Management System，RDBMS)。在企业级的开发领域中，主要采用三大厂商的关系数据库系统：微软公司的 SQL Server、Oracle 公司的 Oracle 和 IBM 公司的 DB2。

1.2 企业应用开发架构

在构建企业级应用时，通常需要大量的代码，而且这些代码一般分布在不同的计算机上，划分代码运行在不同计算机上的理论就是多层设计理论。

企业级应用系统通常分为两层、三层和 N 层架构。

1.2.1 两层架构

传统的两层应用包括用户接口和后台程序，后台程序通常是一个数据库，用户接口直接同数据库进行对话。实现上，通常使用 JSP、ASP 或者 VB 等技术编写这类软件，结构如图 1-2 所示。

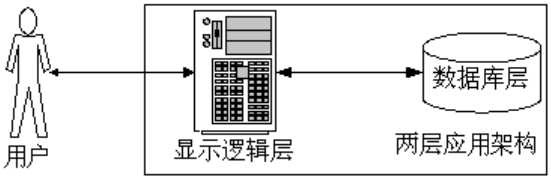

图 1-2 两层应用架构

两层应用架构显示逻辑层一般由 HTML、JSP、ASP 实现，通过 JSP 和 ASP 直接和数据库相连。

1.2.2 三层架构

在两层应用中，应用程序直接同数据库进行对话。三层结构在用户接口代码和数据库中间加入了一个附加的逻辑层，通常这个层称为“商务逻辑层”，如图 1-3 所示。

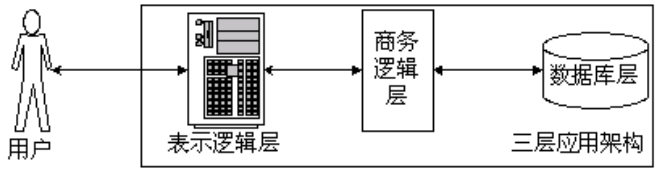

图 1-3　三层应用架构

1.2.3　N 层架构

如果某个应用超过 3 个独立的代码层，就不再称四层或者五层等名称，而是统称为 N 层，那么这个应用称为 N 层应用，如图 1-4 所示。

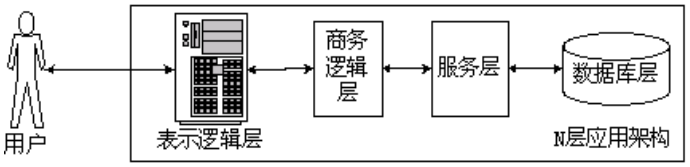

图 1-4　N 层应用架构

1.2.4　开发架构比较

两层架构的优点是开发过程比较简单，利用服务器端的程序直接访问数据库，部署起来比较方便；缺点是程序代码维护起来比较困难，程序执行的效率比较低，用户容量比较少。

三层架构基本解决了两层架构的缺点，将显示部分和逻辑流程控制分开，利用服务器应用程序实现显示部分，利用商务逻辑层实现程序的流程控制，分层使维护变得方便一些，而且执行效率也会有所提高，但是相对部署起来就比较困难一些。

根据实际需要，会进一步细化每一层，或者添加一些层，就形成了 N 层架构。和三层架构一样，组件化的设计使维护相对容易，但是部署相对困难。

1.3　JSP 概述

JSP(Java Server Pages)是由 Sun Microsystems 公司(现已被甲骨文公司收购)倡导的、许多公司参与一起建立的一种动态网页技术标准。JSP 技术是用 Java 语言作为脚本语言，JSP 网页为整个服务器端的 Java 库单元提供了一个接口来服务于 HTTP 的应用程序。

在传统的网页 HTML 文件(*.htm，*.html)中加入 Java 程序片段(Scriptlet)和 JSP 标记(Tag)，就构成了 JSP 网页(*.jsp)。Web 服务器在遇到访问 JSP 网页的请求时，首先执行其中的程序片段，然后将执行结果以 HTML 格式返回给客户。程序片段可以操作数据库、重新定向网页以及发送 E-mail 等，这就是建立动态网站所需要的功能。所有程序操作都在服务器端执行，网络上传送给客户端的仅是得到的结果，对客户浏览器的要求最低，可以实现无 Plugin、无 ActiveX、无 JavaApplet，甚至无 Frame。

1.3.1 什么是 JSP

JSP 是基于 Java 的技术，用于创建可支持跨平台及 Web 服务器的动态网页。从构成情况上来看，JSP 页面代码一般由普通的 HTML 语句和特殊的基于 Java 语言的嵌入标记组成，所以它具有了 Web 和 Java 功能的双重特性。

JSP 1.0 规范是 1999 年 9 月推出的，12 月又推出了 1.1 规范。此后 JSP 又经历了几个版本，最新版本是 2003 年发布的 JSP 2.0。本书介绍的技术都基于 JSP 2.0 规范。

为了让读者对 JSP 技术有一个直观的认识，先来看一个非常简单的 JSP 页面及其运行效果。以下是 helloWorld.jsp 的源代码。

```
<%@ page language="java" contentType="text/html; charset=gbk"%>
<html>
  <head>
    <title>Hello World!</title>
  </head>
  <body bgcolor="#FFFFFF">
    <h3>
    <%
      out.println("JSP Hello World!");
    %>
    </h3>
  </body>
</html>
```

JSP 是一种动态网页技术标准。可以将网页中的动态部分和静态的 HTML 相分离。用户可以使用平常得心应手的工具并按照平常的方式来书写 HTML 语句。然后，将动态部分用特殊的标记嵌入即可，这些标记常常以“<%”开始并以“%>”结束。

程序运行效果如图 1-5 所示。

同 HTML 以及 ASP 等语言相比，JSP 虽然在表现形式上同它们的差别并不大，但是它却提供了一种更为简便、有效的动态网页编写手段，而且，由于 JSP 程序同 Java 语言有着天然的联系，所以在众多基于 Web 的架构中，都可以看到 JSP 程序。

由于 JSP 程序增强了 Web 页面程序的独立性、兼容性和可重用性，所以，与传统的 ASP、PHP 网络编程语言相比，它具有以下特点：

- JSP 的执行效率比较高。由于每个基于 JSP 的页面都被 Java 虚拟机事先解析成一个 Servlet，服务器通过网络接收到来自客户端 HTTP 的请求后，Java 虚拟机解析产生的 Servlet 将开启一个“线程(Thread)”来提供服务，并在服务处理结束后自动销毁这个线程，如图 1-6 所示，这样的处理方式将大大提高系统的利用率，并能有效地降低系统的负载。
- 编写简单。由于 JSP 是基于 Java 语言和 HTML 元素的一项技术，所以只要熟悉 Java 和 HTML 的程序员都可以开发 JSP。
- 跨平台。由于 JSP 运行在 Java 虚拟机之上，所以它可以借助于 Java 本身的跨平台能力，在任何支持 Java 的平台和操作系统上运行。

图 1-5　helloWorld.jsp 运行效果

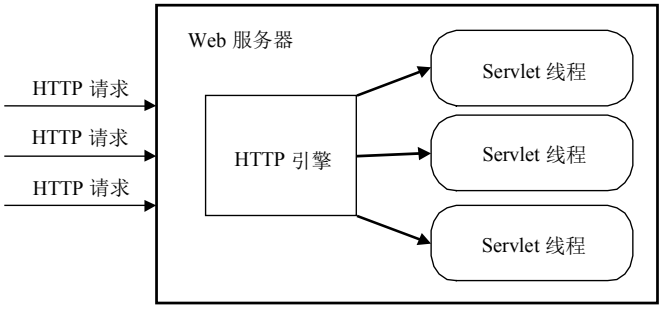

图 1-6　Web 服务器使用 Servlet 提供服务的示意图

- JSP 可以嵌套在 HTML 或 XML 网页中。这样不仅可以降低程序员开发页面显示逻辑效果的工作量，更能提供一种比较轻便的方式来同其他 Web 程序交互。

1.3.2　JSP 技术原理

JSP 文件的执行方式是“编译式”，而不是“解释式”，即在执行 JSP 页面时，是把 JSP 文件先翻译为 Servlet 形式的 Java 类型的字节码文件，然后通过 Java 虚拟机来运行。所以从本质上来讲，运行 JSP 文件最终还是要通过 Java 虚拟机，不过根据 JSP 技术的相关规范，JSP 语言必须在某个构建于 Java 虚拟机之上的特殊环境中运行，这个特殊环境就是 Servlet Container(通常译为 Servlet 容器)，而且，每个 JSP 页面在被系统调用之前，必须先被 Servlet 容器解析成一个 Servlet 文件。

图 1-7 显示了整个 JSP 的运行流程。每次 Servlet 容器接受到一个 JSP 请求时，都会遵循以下步骤：

(1) Servlet 容器查询所需要加载的 JSP 文件是否已经被解析成 Servlet 文件。如果没有，在 Servlet 容器里找到对应的 Servlet 文件，容器将根据 JSP 文件新创建一个 Servlet 文件；反之，如果在容器里有此 Servlet 文件，容器则比较两者的时间，如果 JSP 文件的时间要晚于 Servlet 文件，则说明此 JSP 文件已被重新修改过，需要容器重新生成 Servlet 文件，反之容器将使用原先的 Servlet 文件。

(2) 容器编译好的 Servlet 文件被加载到 Servlet 容器中，执行定义在该 JSP 文件里的各项操作。

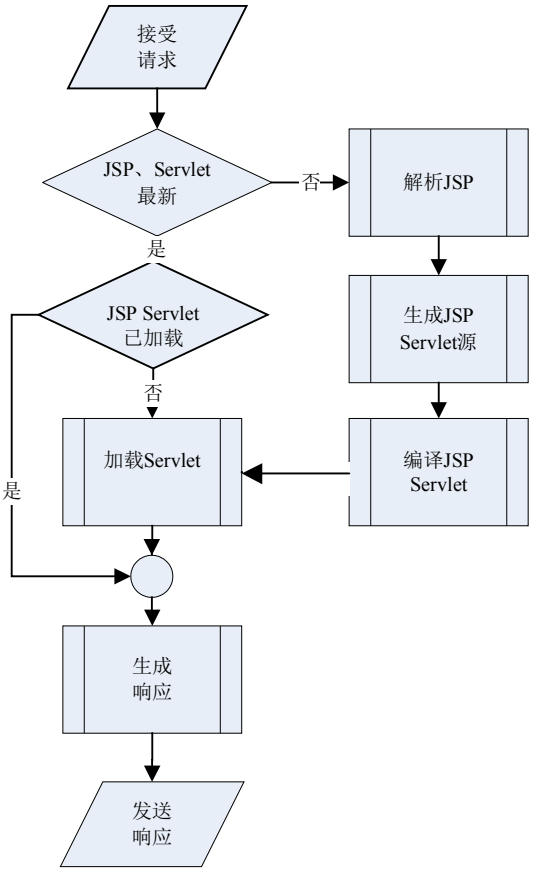

图 1-7　JSP 运行原理

(3) Servlet 容器生成响应结果，并返回给客户端。

(4) JSP 文件结束运行。

从这个 JSP 的工作原理和运作流程上来看，JSP 程序既能以 Java 语言的方式处理 Web 程序里的业务逻辑，更可以处理基于 HTML 协议的请求，它是集众多功能于一身的。

不过，在编写程序的过程中，不能过多地在 JSP 代码里混杂提供显示功能和提供业务逻辑的代码，而是要把 JSP 程序定位到“管理显示逻辑”的角色上。

当服务器第一次接收到对某个页面的请求时，JSP 引擎就开始进行上述的处理过程，将被请求的 JSP 文件编译成 Class 文件。在后续对该页面再次进行请求时，若页面没有进行任何改动，服务器只需直接调用 Class 文件执行即可。所以当某个 JSP 页面第一次被请求时，会有一些延迟，而再次访问时会感觉快了很多。如果被请求的页面经过修改，服务器将会重新编译这个文件，然后执行。

1.3.3 JSP 和其他动态网站开发技术

动态网页是与静态网页相对应的，也就是说，网页 URL 不固定，能通过后台与用户交互，完成用户查询、提交等动作。所谓“动态”，并不是指放在网页上的 GIF 图片，动态网页技术有以下几个特点：

- 交互性：即网页会根据用户的要求和选择而动态改变和响应，将浏览器作为客户端界面，这将是今后 Web 发展的大趋势。
- 自动更新：即无须手动地更新 HTML 文档，便会自动生成新的页面，可以大大节省工作量。
- 因时因人而变：即当不同的时间、不同的人访问同一网址时会产生不同的页面，可以大大减少工作量。

除了早期的 CGI 外，目前主流的动态网页技术有 ASP、PHP、JSP 等，下面分别介绍这些动态网页技术。

1. CGI

早期的动态网站开发技术是基于 CGI(Common Gateway Interface，公共网关接口)的，功能主要是客户端向服务器发送请求，Web 服务器接收到请求后启动所指定的 CGI 程序来完成诸如对数据库进行访问、存储信息等操作，最后将处理的结果反馈给客户端。CGI 程序包括两个主要部分，一个是程序代码，一个是 HTML 代码。由于每次修改 HTML 页面代码都必须重新编译 CGI 程序，以至于最后在 CGI 程序调试中，调试 HTML 代码的工作量可能超过调试 CGI 程序代码的工作量。CGI 属于比较早期的服务器端动态技术，但由于其发展的历史较早，目前使用此项技术所构建的网站依然不在少数，然而由于其不易学习和效率不高的特性，在 ASP 及 JSP 等技术出现之后，已逐渐淡出用户的视线。

2. ASP

ASP(Active Server Pages)是 Microsoft 公司开发的一种类似 HTML、Script(脚本)与 CGI 的结合体，可以结合 HTML 网页、ASP 指令和 ActiveX 元件建立动态、交互且高效的 Web 服务器应用程序。ASP 允许用户使用包括 VBScript 和 JavaScript 等在内的许多已有的脚本语言编写 ASP 应用程序。ASP 程序的编制比 HTML 更方便且更有灵活性。它是在 Web 服务器端运行，

运行后再将运行结果以 HTML 格式传送至客户端的浏览器。更精确来说 ASP 是一个中间件，这个中间件将 Web 上的请求转入到一个解释器中，在这个解释器中将所有的 ASP 的 Script 进行分析，再执行，而这时可以在这个中间件中去创建一个新的 COM 对象，对这个对象中的属性和方法进行操作和调用，同时再通过这些 COM 组件完成更多的工作。所以说，ASP 强大的不在于它的 VBScript， 而在于它后台的 COM 组件，这些组件无限地扩充了 ASP 的能力。

ASP 具有以下优势：

- 简单易学。BASIC 语言有着很大的用户群，开发基础是最广的。Microsoft 有一项做得非常好的联机手册，又有本地化语言的帮助，同时迅速出现教学书籍，这样大大方便了开发人员的学习和对技术的深入研究。
- 安装使用方便。装好一个 Windows 操作系统，只要安装了 IIS(Internet Information Server)，那么 ASP 就可以使用了，不需要花心思去配置。
- 开发工具强大而多样、易用、简单、人性化，这本就是微软的强项。
- 效率高。在低的访问量下，ASP 能体现出一定的效率，这时它对机器的要求并不高。

ASP 具有以下弱势：

- Windows 本身的所有问题都会一成不变地累加到它的身上。安全性、稳定性、跨平台性的不足都会因为与 Windows 的捆绑而显现出来。
- ASP 由于使用了 COM 组件，所以它变得十分强大，但是这样的强大由于 Windows NT 系统最初的设计问题而会引发大量的安全问题。只要在这样的组件或是操作中稍不注意，那么外部攻击就可以取得相当高的权限而导致网站瘫痪或者数据丢失。
- 由于 ASP 还是一种 Script 语言，所以除了大量使用组件外，没有办法提高其工作效率。它必须面对即时编译的时间考验。
- 无法实现跨操作系统的应用。ASP 基本上只能局限于 Microsoft 公司的操作系统平台之上，例如 IIS 和 PWS(Personal Web Server)等。
- 无法完全实现一些企业级的功能，如无法实现完全的集群、负载均衡。

3. ASP.NET

ASP.NET 是 ASP 在微软的.NET 平台上的升级版本，是微软集成.NET 平台发展而来的服务器端网页语言，用户可以使用任何.NET 兼容的语言来编写 ASP.NET 应用程序。使用 Visual Basic.NET、C#、J#、ASP.NET 页面(Web Forms)进行编译可以提供比脚本语言更出色的性能表现。Web Forms 允许用户在网页基础上建立强大的窗体。当建立页面时，可以使用 ASP.NET 服务器端控件建立常用的 UI 元素，并对它们编程来完成一般的任务。这些控件允许用户使用内建可重用的组件和自定义组件快速建立 Web Forms，使代码简单化。

ASP.NET 使用.NET 提供的类别库与对象导向理论建构的服务器端动态网页，提供了一种编程模型和结构，不仅功能强大，紧密结合.NET 平台，而且对比原来的 Web 技术来说，它能更快速、容易地建立灵活、安全和稳定的应用程序，在性能上也有相当出色的表现，近年来已经成为最热门的动态网页技术之一。

4. PHP

PHP(Hypertext Preprocessor)是一种 HTML 内嵌式的语言(类似于 IIS 上的 ASP)，是一种用

于创建动态 Web 页面的服务器端脚本语言。而 PHP 独特的语法混合了 C、Java、Perl 以及 PHP 式的新语法。它可以比 CGI 或者 Perl 更快速地执行动态网页。同样 PHP 也可以混合使用 PHP 和 HTML 编写 Web 页面，当客户端访问某页面时，服务器端会首先对页面中的 PHP 命令进行处理，然后把处理后的结果连同 HTML 内容一起传送到客户端的浏览器。

PHP 能够支持诸多数据库，如 Microsoft SQL Server、MySql、Sybase、Oracle 等。

PHP 与 HTML 语言具有非常好的兼容性，用户可以直接在脚本代码中加入 HTML 标签，或者在 HTML 标签中加入脚本代码从而更好地实现页面控制。PHP 提供了标准的数据库接口，数据库连接方便，兼容性强，扩展性强，可以进行面向对象编程。

另外，PHP 是一种开源程序，拥有很好的跨平台兼容性。用户可以在 Windows 系列以及许多版本的 UNIX 和 Linux 系统上运行 PHP，而且可以将 PHP 作为 Apache 服务器的内置模块或 CGI 程序运行。

PHP 具有以下优势：

- PHP 是一种能快速学习、跨平台、有良好数据库交互能力的开发语言。ASP 比不上它的就是这种跨平台能力，而正是它的这种能力让 UNIX、Linux 有了一种与 ASP 媲美的开发语言。PHP 语法简单、书写容易，现在市面上也有大量的相关书籍，同时 Internet 上也有大量的代码可以共享。
- 与 Apache 及其他扩展库结合紧密。PHP 可以与 Apache 以静态编绎的方式结合起来，与其他的扩展库也可以用这样的方式结合(Windows 平台除外)。该方式的最大好处就是最大化地利用了 CPU 和内存，同时极为有效地利用了 Apache 高性能的吞吐能力。同时外部的扩展也是静态联编，从而达到了最快的运行速度。由于与数据库的接口也使用了这样的方式，所以使用的是本地化的调用，这也让数据库发挥了最佳效能。
- 良好的安全性。由于 PHP 本身的代码开放，所以它的代码在许多工程师手中进行了检测，同时与 Apache 编译在一起的方式也可以让它具有灵活的安全设定。所以到现在为止，PHP 具有了公认的安全性能。

PHP 具有以下弱势：

- 数据库支持的极大变化。由于 PHP 的所有扩展接口都是独立团队开发完成的，同时在开发时为了形成相应数据的个性化操作，所以 PHP 虽然支持许多数据库，可是针对每种数据库的开发语言都完全不同。这样形成针对一种数据库的工发工作，在数据库进行升级后需要开发人员进行几乎全部的代码更改工作。而为了让应用支持更多种的数据库，就需要开发人员将同样的数据库操作使用不同的代码写出 n 种代码库，让程序员的工作量大大增加。
- 安装复杂。由于 PHP 的每一种扩充模块并不是完全由 PHP 本身来完成，需要许多外部的应用库，如图形需要 GD 库、LDAP 需要 LDAP 库等，这样在安装完成相应的应用后，再联编进 PHP 中来。只有在这些环境下才能方便地编译对应的扩展库。这些都是一般开发人员在使用 PHP 前所先要面对的问题。
- 缺少企业级的支持。没有组件的支持，那么所有的扩充就只能依靠 PHP 开发组所给出的接口，事实上这样的接口还不够多。同时难以将集群、应用服务器这样的特性加入到系统中去，而一个大型的站点或是一个企业级的应用一定需要这样的支持。注意：

在 PHP4.0 版本以后加入了对 Servlet、JavaBean 的支持，也许这样的支持会在以后的版本中更加增强，也许这样的支持会是 PHP 以后的企业级支持的起点。

- 缺少正规的商业支持。这也是自由软件一贯的缺点。
- 无法实现商品化应用的开发。由于 PHP 没有任何编译性的开发工作，所有的开发都是基于脚本技术来完成的。所有的源代码都无法编译，所以完成的应用只能是自己或是内部使用，无法实现商品化。

5. JSP

JSP 不像 CGI、ISAPI 和 NSAPI 那样难于编写和维护，不像 PHP 那样只能适应中小流量的网站，也不像 ASP 那样受到跨平台的限制(只能运行于 Microsoft 公司开发的 IIS 和 PWS 上)。JSP 体现了当今最先进的网站开发思想，和其他 Web 开发工具相比，JSP 有着强大的优势：

- 程序可以跨平台执行。JSP 可以让开发人员在任意环境中进行开发，在任意环境中进行系统部署，在任意环境中扩展应用程序。
- 多样化和功能强大的开发工具支持。Java 有许多非常优秀的开发工具，而且有的工具可以免费得到，其中许多工具已经可以顺利地运行于多种平台之下。
- 强大的可伸缩性。从只有一个小的 JAR 文件就可以运行 Servlet/JSP，到由多台服务器进行集群和负载均衡，到多台 Application 进行事务处理，一台服务器到无数台服务器，Java 显示了巨大的生命力。

当然，JSP 也有不足，Java 的一些优势也是它致命的问题所在。

- 跨平台的功能和极度的伸缩能力极大地增加了产品的复杂性。也就是说，它在扩展时需要分成多少块，那么 Java 系统中就有多少种产品，所以用户可能会看到 JRE、JDK、JSDK，JSWDK 等，而实际上它们是密不可分的。只要将它们有效地搭配在一起，就可以产生强大的效能。当然，这也使应用程序变得非常复杂。
- JSP 运行是用 class 常驻内存来完成的，虽然提高了响应速度，但要占用内存。 Java 的运行速度是用 class 常驻内存来完成的，所以它在一些情况下所使用的内存比起用户数量来说确实是“最低性价比”了。另一方面，它还需要硬盘空间来储存一系列的.java 文件和.class 文件，以及对应的版本文件。
- JSP 程序调试也不是很方便。JSP 页面首先被转化为一个.java 文件(Servlet)，然后再被编译。这样，出错信息实际上指向的是经过转化的那个.java 文件而不是.jsp 本身。

1.3.4 J2EE 简介

目前，Java 2 平台有 3 个版本，它们是适用于小型设备和智能卡的 Java 2 平台 Micro 版(Java 2 Platform Micro Edition，J2ME)、适用于桌面系统的 Java 2 平台标准版(Java 2 Platform Standard Edition，J2SE)、适用于创建服务器应用程序和服务的 Java 2 平台企业版(Java 2 Platform Enterprise Edition，J2EE)。本节主要介绍 J2EE。

J2EE 是一种利用 Java 2 平台来简化企业解决方案的开发、部署和管理相关复杂问题的体系结构。J2EE 技术的基础就是核心 Java 平台或 Java 2 平台的标准版。J2EE 不仅巩固了标准版中的许多优点，例如“编写一次、随处运行”的特性、方便存取数据库的 JDBC API、CORBA

技术以及能够在 Internet 应用中保护数据的安全模式等，同时还提供了对 EJB(Enterprise JavaBeans)、Java Servlets API、JSP(Java Server Pages)以及 XML 技术的全面支持。其最终目的就是成为一个能够使企业开发者大幅缩短投放市场时间的体系结构。

需要指出的是，J2EE 并非一个产品，而是一系列的标准。因此，从整体上讲，J2EE 是使用 Java 技术开发企业级应用的一种事实上的工业标准(Sun 公司出于其自身利益的考虑，至今没有将 Java 及其相关技术纳入标准化组织的体系)，它是 Java 技术不断适应和促进企业级应用过程中的产物。Sun 推出 J2EE 的目的是为了克服传统Client/Server模式的弊病，迎合 Browser/Server 架构的潮流，为应用 Java 技术开发服务器端应用提供一个平台独立的、可移植的、多用户的、安全的和基于标准的企业级平台，从而简化企业应用的开发、管理和部署。J2EE 是一个标准，而不是一个现成的产品。各个平台开发商按照 J2EE 规范分别开发了不同的 J2EE 应用服务器，J2EE 应用服务器是 J2EE 企业级应用的部署平台。由于它们都遵循 J2EE 规范，因此，使用 J2EE 技术开发的企业级应用可以部署在各种 J2EE应用服务器上。目前市场上可以看到很多实现了 J2EE 的产品，如 BEA WebLogic、IBM WebSphere 以及开源的 JBoss 等。

J2EE 使用多层的分布式应用模型，应用逻辑按功能划分为组件，各个应用组件根据他们所在的层分布在不同的机器上。事实上，Sun 设计 J2EE 的初衷正是为了解决两层模式(Client/Server)的弊端，在传统模式中，客户端担当了过多的角色而显得臃肿，在这种模式中，第一次部署的时候比较容易，但难于升级或改进，可伸展性也不理想，而且经常基于某种专有的协议(通常是某种数据库协议)。这使得重用业务逻辑和界面逻辑非常困难。现在 J2EE 的多层企业级应用模型将两层化模型中的不同层面切分成许多层。一个多层化应用能够为每种不同的服务提供一个独立的层，图 1-8 所示是 J2EE 典型的四层结构。

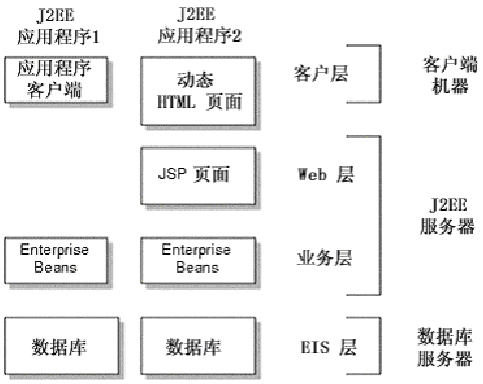

图 1-8　J2EE 四层结构模型

J2EE 的四层结构分别是：

- 运行在客户端机器上的客户层组件。
- 运行在 J2EE 服务器上的 Web 层组件。
- 运行在 J2EE 服务器上的业务层组件。
- 运行在数据库服务器上的企业信息系统(Enterprise Information System，EIS)层软件。

下面分别对图 1-8 中的四层结构进行说明：

(1) J2EE 应用程序组件。J2EE 组件是具有独立功能的软件单元，它们通过相关的类和文件组装成 J2EE 应用程序，并与其他组件交互。J2EE 说明书中定义了以下的 J2EE 组件：应用客户端程序和 Applets 是客户层组件；Java Servlet 和 Java Server Pages(JSP)是 Web 层组件；Enterprise JavaBeans(EJB)是业务层组件。

(2) 客户层组件。J2EE 应用程序可以是基于 Web 方式的，也可以是基于传统方式的。

(3) Web 层组件。J2EE Web 层组件可以是 JSP 页面或 Servlets，按照 J2EE 规范，静态的 HTML 页面和 Applets 不算是 Web 层组件。如图 1-9 所示，Web 层可能包含某些 JavaBean 对象来处理用户输入，并把输入发送给运行在业务层上的 Enterprise Bean 来进行处理。

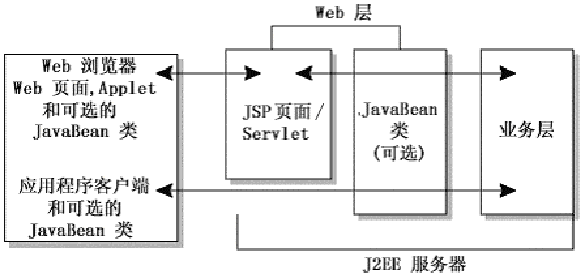

图 1-9　Web 层

(4) 业务层组件。业务层代码的逻辑用来满足银行、零售、金融等特殊商务领域的需要，由运行在业务层上的 Enterprise Bean 进行处理。图 1-10 表明了一个 Enterprise Bean 是如何从客户端程序接收数据，进行处理(如果必要的话)，并发送到 EIS 层储存的，这个过程也可以逆向进行。

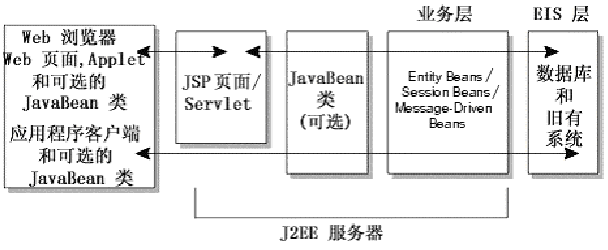

图 1-10　J2EE 服务层

有三种企业级的 Bean：会话(Session) Beans、实体(Entity) Beans 和消息驱动(Message-Driven) Beans。会话 Bean 表示与客户端程序的临时交互。当客户端程序执行完后，会话 Bean 和相关数据就会消失。相反，实体 Bean 表示数据库表中的一行永久的记录。当客户端程序中止或服务器关闭时，就会有潜在的服务保证实体 Bean 的数据得以保存。消息驱动 Bean 结合了会话 Bean 和 JMS 的消息监听器的特性，允许一个业务层组件异步接收 JMS 消息。

(5) EIS 层。企业信息系统(EIS)层处理企业信息系统软件包括企业基础建设系统，如企业资源计划(ERP)、大型机事务处理、数据库系统，和其他的遗留信息系统，如 J2EE 应用组件可能为了数据库连接需要访问企业信息系统。

注意

有些书中也把J2EE分为客户端机器、J2EE服务器和数据库服务器三层。J2EE Web层组件可以是 JSP 页面或 Servlet。按照 J2EE 规范，静态的 HTML 页面和 Applet 不算是 Web 层组件。

1.4 JSP 知识体系及学习之路

JSP 技术本身并不复杂，但是由于 JSP 是一种综合技术，它涉及了许多其他的技术，这些技术组合起来形成了 JSP 知识体系，整个的 JSP 知识体系是比较庞大的。

1.4.1 JSP 知识体系

JSP 的知识体系如图 1-11 所示。

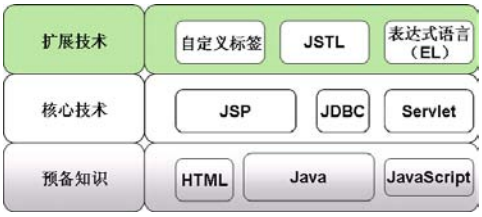

图 1-11 JSP 知识体系图

Java 和 HTML 是 JSP 学习中非常重要的基础，如果仅仅懂得 JSP 的一些语法而对 Java 的基础知识不了解，那么要开发一个高级的动态网站也是相当困难的。JSP 之所以被越来越多的人接受，一个很重要的原因是它依靠 Java 的强大优势。可以说，如果只是使用了 JSP 的基本功能来制作一个网站，那么这个 JSP 网站也许跟 ASP 网站十分类似了。前面讲过，JSP 最终是要编译成 Java Servlet 来执行的，而 Servlet 从本质上说就是一个 Java 类，整合内部逻辑的 JavaBean 也是一个 Java 类，所以，了解 Java 语言对开发一个动态网站至关重要。当然，网站开发也只是使用 Java 语言中的部分内容，像 Swing 和 Applet 等知识就会用得特别少，用户也不需要对其进行了解，但是熟悉基本的语法、逻辑控制以及面向对象等概念还是很有必要的。

如果读者已经掌握这些基础知识，这就意味着 JSP 的学习之路要轻松很多。如果没有这些基础，那么就需要花一些时间来学习这些基础知识。

1.4.2 JSP 程序员学习路径

如何成为一个成功的 JSP 程序员？一个常见的错误是把 JSP 当作简化的 Java(事实上，JSP 是简化的 Servlet)。JSP 是一个衔接技术，并且成功地连接用户需要理解的其他技术。如果已经知道 Java、HTML 和 JavaScript，这意味着 JSP 将确实是简单的。要成为一个成功的 JSP 程

序员可参考下面的步骤：

(1) 保证理解 HTML / XHTML。用户需要了解 HTML 基础，特别是 HTML 布局中的 Form 和 Table 的使用。XHTML 不久将代替 HTML，学习 XHTML 的基础是一个好主意。许多程序员通过集成开发环境学习 HTML。因为大多数集成开发环境产生混乱的 HTML 语法，所以花时间学习手工写作 HTML 是很有必要的。因为使用 JSP 和 HTML 混合编程，精通 HTML 语法是重要的。所以，必须能流利地写 HTML。

(2) 开始学习 Java。开始学习 Java，理解 Java 基础是很重要的。不用担心学习 Swing 或 Java 的图形方面，因为在 JSP 中不会使用这些特征。集中精力在 Java 的工作细节，学习 Java 的逻辑，也在 JavaBean 上花时间。学习 Applet 是好的，但是就像 Swing，JSP 的大多数应用将不使用小程序。

(3) 学习 JavaScript。学习怎么用 JavaScript 在 HTML 中验证输入的 Form 元素，也学习 JavaScript 怎么能在 HTML 页内修改 Form 的元素。最后要求从 HTML 页内的事件中触发 JavaScript Function。JavaScript 是一种基于网页的客户端脚本技术，这种技术的核心思想是通过这种技术，来增加用户与浏览器的交互，增加用户在使用网页应用时的体验。

(4) 学习安装和配置一种 Servlet 容器。推荐以 Tomcat 开始，它可以很好地运行 JSP 程序。学习技术的最好方法就是一边学习一边实践。为了运行开发的 JSP 和 Servlet 实例，当然要建立一个测试和运行环境。Tomcat 是 JSP 规范和 Servlet 规范的参考实现，因此也推荐读者在学习阶段使用它作为运行环境。另外，许多 JSP 程序员也使用 Tomcat，因此当遇到问题时，将容易寻得帮助。

(5) 开始学习 JSP 基本语法。JSP 的基本语法包括 JSP 脚本元素、JSP 指令元素、JSP 动作元素等几个基本的组成部分，这一部分知识是 JSP 区别于其他技术的主要内容。

(6) 学习 JDBC。JSP 大多数应用都使用数据库，JDBC 被用于数据库连接。经常忽略的一个事实就是，每个 JDBC Driver 所支持的东西是相当不同的。了解并熟悉在 JSP 工程上被使用的 JDBC Driver 的细节是很重要的。

(7) 学习 Servlet。JSP API 是建立在 Servlet API 基础之上的，为了更深入地理解 JSP，需要学习 Servlet。另外，在高级的 JSP 应用开发中，Servlet 的应用也是很多的，因此作为一个高级的 JSP 程序员，Servlet 的知识是必备的。通过全面深入地学习 Servlet，将会真正理解 JSP 应用在 Servlet 容器上的运行原理，理解 JSP 页面和 Servlet 响应客户端请求的整个过程，此时会将产生一种豁然开朗的感觉。

(8) 学习开源框架。框架(Framework)是一个可复用的设计，它是由一组抽象类及其实例间协作关系来表达的。其实框架就是某种应用的半成品，就是一组组件，供用户选用完成自己的系统。简单说就是使用别人搭好的舞台，你来做表演。而且，框架一般是成熟的、不断升级的软件。框架一般处在低层应用平台(如 J2EE)和高层业务逻辑层之间。

到这里，你已经成为了熟练的 JSP 程序员，但仍然有很多需要学习，可以考虑扩展自己的知识，比如 DHTML、XML、Java 证书、JSP Tag Libraries 或表达式语言，根据想要建造什么类型的网站而决定了。

这些训练是 JSP 的核心。读者不必学完上面所有的，这取决于在工程中分配到什么任务和已经有什么知识。但要成为一个资深的 Web 程序员，所学的东西远远不止这一些。

1.5 小 结

JSP 是目前最为流行的基于 Java Servlet 的 Web 开发技术之一，其底层以 Java 语言为支撑，基于 Servlet 技术，具有很好的开放性、可移植性和可扩展性。本章简单介绍了软件编程系统，动态网站开发技术的现状及 JSP、ASP 和 PHP 等几种主要动态网页技术，并概括了开发一个动态网站所需要掌握的一些必备知识，为后续章节的学习打下基础。

1.6 习 题

1.6.1 选择题

1. JSP 文件在第一次运行的时候被 JSP 引擎编译为(　)文件。

A. Servlet　　B. Class　　C. HTML　　D. XML

2. MVC 设计模式将应用程序分为(　)部分。

A. 2　　B. 3　　C. 4　　D. 5

3. 与 JSP 同类型的技术有(　)。

A. JavaScript　　B. Java　　C. ASP.NET　　D. C#

4. 以下文件名后缀中，只有(　)不是动态网页的后缀。

A. .jsp　　B. .html　　C. .aspx　　D. .php

5. 以下关于 Servlet 和 JSP 的叙述中，正确的是(　)。

A. JSP 和 Servlet 都是 Java

B. Servlet 是 Java 平台下实现的基本技术

C. 在 Servlet 中，需要用 Java 代码向客户端输出返回信息

D. 以上都不对

6. JSP 页面经过转译之后，将创建一个(　)文件。

A. Applet　　B. Servlet　　C. Application　　D. Server

7. 下面关于 JSP 的说法不正确的是(　)。

A. 将内容的生成和显示进行分离　B. 能够跨平台

C. 可以直接在浏览器端解释执行　D. 采用标签简化页面开发

8. JSP 在转译阶段生成文件的扩展名是(　)。

A. class　　B. java　　C. exe　　D. bin

9. HTTP 协议默认使用(　)。

A. 8080 端口　　B. 7001 端口　　C. 80 端口　　D. 25 端口

10. 在 JDK 的工具包中用来编译 Java 源文件的工具是(　)。

A. Javac　　B. Javap　　C. Java　　D. Javah

1.6.2 判断题

1. 静态网页*.htm 中可以嵌入脚本代码，如 JavaScript、VBScript 程序段等，但这些程序段不可能在服务器端运行，只能在客户端运行。 ( )

2. 动态网页是在服务器端被执行，其中嵌入的代码只能在服务器端运行，不能在客户端浏览器中运行。 ( )

3. JSP 文件可以单独运行。 ( )

4. JSP 是解释性语言。 ( )

5. JSP 是以 Servlet 程序方式运行的，而 ASP 是由 ASP 引擎解释执行的。 ( )

1.6.3 填空题

1. JSP 的英文全称是_____，中文全称是_____。

2. JSP 页面由_____和嵌入其中的_____所组成。

3. Tomcat 是 JSP 运行的_____。

4. MVC 设计模式将应用程序分为模型、_____和_____。

5. JSP 的实质就是_____。

1.6.4 简答题

1. 什么是静态网页？什么是动态网页？两者的区别是什么？试举例说明。

2. 什么是 B/S 模式？什么是 C/S 模式？试举例说明。

3. 什么是 JSP？与 ASP、PHP、ASP.NET 相比，JSP 有哪些优点？

4. JSP、Java 和 JavaScript 有什么区别与联系？

1.6.5 拓展实践题

1. 体验静态网页和动态网页的特点。

在客户端浏览器地址栏中输入 http://www.sina.com/进入新浪网主页，查看新闻内容，体验静态网页的特点。在客户端浏览器地址栏中输入 http://www.csdn.net/index.htm，进入中国程序员网，注册会员，体会动态网页特点。在“程序员网”注册成功后，使用用户名登录程序员网，查询相关 JSP 的学习资料。

2. 通过网上书店购书，体验 Web 应用程序特点。

第2章

JSP动态网页设计基础

要使用 JSP 开发专业的动态网站，首先必须熟练掌握静态网站的制作技术。HTML 是在学习 JSP 之前必须了解的基础知识，很多 JSP 语法的使用都是建立在 HTML 文档的基础上。实际开发中，一般都是使用现成的 HTML 文档来添加 JSP 的动态脚本并做适当修改，除了特殊的应用，很少从零开始写一个 JSP 页面，所以读懂 HTML 文档并了解 HTML 语言中的技巧，可为更快地上手 JSP 提供很大的帮助。

JavaScript 是 JSP 知识体系中一个可选的知识模块。也就是说，不了解 JavaScript 知识不会影响 JSP 的应用开发，但是如果掌握了 JavaScript 的知识，将可以更加方便地解决网页开发中的某些特定问题，例如经常使用 JavaScript 判断用户在表单中输入数据的合法性。

本章集中介绍一些实际开发中经常用到的 HTML 标签，以及简单的 JavaScript 应用。

本章学习目标

- ◎ 了解 HTML 与 JSP 的关系
- ◎ 掌握 HTML 的常用标签
- ◎ 能够使用 HTML 设计基本网页
- ◎ 能够使用 HTML 设计网络中常用的表单
- ◎ 能够使用简单的 CSS 控制页面样式
- ◎ 了解 JavaScript 在 JSP 学习中的作用
- ◎ 了解 JavaScript 的简单应用
- ◎ 能够读懂 JavaScript 程序，并能通过查阅资料了解更多 JavaScript 应用

2.1 HTML 技术

HTML 是 Hyper Text Markup Language 的缩写，中文一般译为“超文本标记语言”。用 HTML 语言编写的超文本文档称为 HTML 文档，和一般文档的不同之处是，一个 HTML 文件不仅包含文本内容，还包含一些 Tag，中文称为“标签”或者“标记”。HTML 文件的后缀名是.html 或者.htm。

本书简单介绍 HTML 5，HTML 5 是用于取代 HTML 4.01 和 XHTML 1.0 标准的 HTML 标准版本。HTML 5 第一份正式草案已于 2008 年 1 月公布，现在仍处于完善之中，大部分现代浏览器已经部分支持 HTML5。

2.1.1 HTML5 基本结构

HTML 文档的基本结构如下：

```
<!DOCTYPE html>
<html>
<head>
...
</head>
<body>
...
</body>
</html>
```

我们称<html>…</html>、<body>…</body>等为“标签”或“标记”，它是组成 HTML 文件的重要部分，标签是一些字母或单词，放在尖括号内。

- <!DOCTYPE html>表示文档的类型。
- HTML 文档分为文档头和文档体两部分，头部信息和主体部分信息都包含在<html>和</html>标签之间，这一对标签表示包含的内容为 HTML 文档。
- 头部信息放在<head>和</head>之间，用来说明网页的一些基本情况，例如网页的标题等。
- 网页的主体信息包含在<body>和</body>之间，包含网页的正文部分。

【例 2-1】网页的基本结构(first.html)。

```
<!DOCTYPE html>
<html>
<head>
<title>我的第一个网页</title>
</head>
<body>
众里寻他千百度，蓦然回首，那人却在，灯火阑珊处。
</body>
</html>
```

创建 HTML 文档的步骤如下：

(1) 建立站点根目录。新建一个文件夹用于存放网页，比如在 D 盘创建一个文件夹 ch02，在本章我们把它作为站点根目录，读者在做练习时可以自行设定站点根目录。

(2) 编写代码。HTML 可以使用任何字处理软件来编写，最简单的是 Windows 自带的记事本。用记事本编写以上代码，标题和网页主体内容可以适当修改。

(3) 保存文件。文件名命名为 first.html(注意：其中 first 是自己为文件起的名字，可修改，但扩展名必须为 html 或者 htm)，另外注意将“保存类型”改为“所有文件”，如图 2-1 所示。

图 2-1 保存 html 文件

(4) 浏览网页。保存完毕后，在文件夹 ch02 中可以看到 first.html 被浏览器识别，双击即可查看网页效果，在 IE9.0 中浏览效果如图 2-2 所示，如需要继续编辑，则右击选择以记事本打开。

图 2-2 第一个网页浏览效果

注意

记事本默认的扩展名为.txt，在保存 HTML 文件时，最好的方法是在文件名的两侧加上双引号，并将“保存类型”改为“所有文件”，可以保证文件以 HTML 文件格式存储。

我们回过头分析一下第一个网页，其中的代码由四对标签组成。

- <html>…</html>：表示整个 HTML 文档的开始和结束，网页的所有内容都位于这两个标签之间。

- <head>…</head>：头部标签，在例子中，头部之间插入一对标题标签。
- <title>…</title>：标题标签，说明该 HTML 文件的标题是什么，当浏览该文件时，标题显示在浏览器上方的标题栏内。在第一个网页中，我们在<title>…</title>标签之间包含的标题是“我的第一个网页”，在图 2-2 中可以看到它显示在网页的标题栏。
- <body>…</body>：主体标签，页面的主体部分都放在这对标签之间。在第一个网页中，我们在<body>…</body>标签之间包含的内容是“众里寻他千百度，蓦然回首，那人却在，灯火阑珊处。”，在图 2-2 中可以看到这行字显示在网页的主体部分。

2.1.2 HTML 常用标签

前文我们提到 HTML 用于描述功能的符号称为“标签”，也称为“标记”。比如<html>、<head>、<body>等都是标签。标签大多是成对使用，分别表示功能的开始和结束，但也有不需要结束标签的。

标签可以从大体上分为单标签(没有结束标签)和双标签(有结束标签)两种，根据有没有属性，有以下四种显示方式：

- <标签>…</标签>
- <标签 属性 1=值 1 属性 2=值 2 …>...</标签>
- <标签 />
- <标签 属性 1=值 1 属性 2=值 2 … />

HTML 常用的标签有文本相关标签、图像标签、超链接标签、表格类标签、表单标签等。

1. 文本

在网页中添加文本非常简单，只需要把要添加的文字输入到<body>和</body>之间。

【例 2-2】在网页中输入文字(text.html)。

在网页中输入词《青玉案 • 元夕》，代码如下。

```
<!DOCTYPE html>
<html>
<head>
<title>青玉案 • 元夕</title>
</head>
<body>
东风夜放花千树，
更吹落，星如雨。
宝马雕车香满路。
凤箫声动，玉壶光转，一夜鱼龙舞。

蛾儿雪柳黄金缕，
笑语盈盈暗香去。
众里寻他千百度，
蓦然回首，那人却在，灯火阑珊处。
</body>
</html>
```

预览网页，结果如图 2-3 所示，似乎没有我们预期的结果，这是因为，如果没有标签修饰文本，文字将以无格式的形式显示(如果浏览器窗口显示不下，则自动换行)，要实现分段可以使用标签来修饰文字，但是 HTML5 不再有用于修饰文本字体、大小和颜色的<font>标签，而改由 CSS 来实现这些。

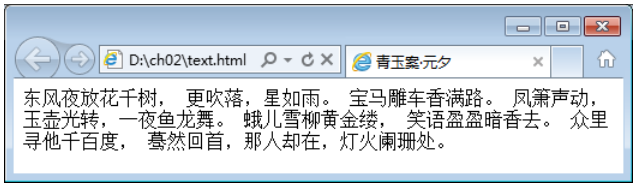

图 2-3 多行文本的预览

(1) 分段排版。分段标签主要有“预先格式化标签”<pre>…</pre>、“换行标签”
和“段落标签” … 。

- 预先格式化标签<pre>…</pre>：添加在文本的开始和结尾处。用这个标签括起来的文本，在网页中会按照输入时的格式显示。
- 换行标签
：换行标签加在需要换行的位置，当浏览器遇到这个标签时，会自动进行换行。
- 段落标签 … ：段落标签添加在段首和段尾。值得注意的是，在 HTML5 中 标签不再保留属性 align。段落标签和
标签的不同之处在于在不同段落之间，多一行空行。

【例 2-3】预先格式化标签的使用(pre.html)。

```
<!DOCTYPE html>
<html>
<head>
<title>青玉案 元夕</title>
</head>
<body>
<pre>
东风夜放花千树，
更吹落，星如雨。
宝马雕车香满路。
凤箫声动，玉壶光转，一夜鱼龙舞。

蛾儿雪柳黄金缕，
笑语盈盈暗香去。
众里寻他千百度，
蓦然回首，那人却在，灯火阑珊处。
</pre>
</body>
</html>
```

预览结果如图 2-4 所示。

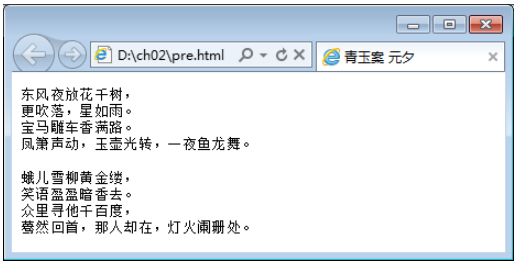

图 2-4　<pre>…</pre>标签的使用

【例 2-4】换行和段落标签的使用(paragraph.html)。

```
<!DOCTYPE html>
<html>
<head><title>换行标签和段落标签</title></head>
<body>
<p>老木匠的房子</p>
<hr />
<p>有个老木匠准备退休，回家与妻子儿女享受天伦之乐。<br />
老板舍不得他的好工人走，问他是否能帮忙再建一座房子，老木匠说可以。但是他的心已不在工作上，他用的是软料，出的是粗活。房子建好的时候，老板把大门的钥匙递给他。<br />
“这是你的房子，”他说，“我送给你的礼物。”</p>
<p>他震惊得目瞪口呆，羞愧得无地自容。如果他早知道是在给自己建房子，他怎么会这样呢？现在他得住在一幢粗制滥造的房子里！我们又何尝不是这样。我们漫不经心地“建造”自己的生活，不是积极行动，而是消极应付，凡事不肯精益求精，在关键时刻不能尽最大努力。等我们惊觉自己的处境，早已深困在自己建造的“房子”里了。</p>
</body>
</html>
```

预览结果如图 2-5 所示。

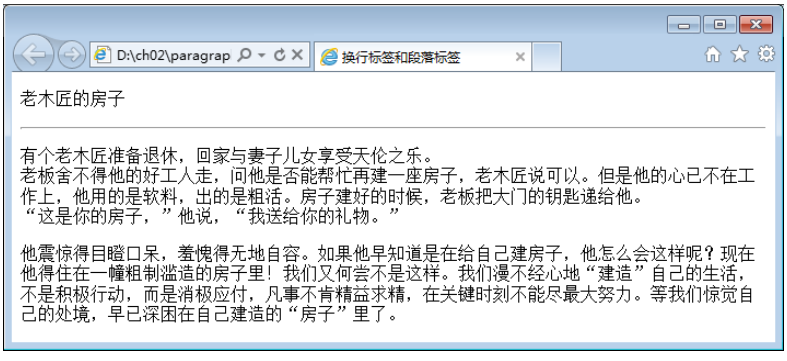

图 2-5　段落标签和换行标签的使用

除了段落标签和换行标签外，该例子第 6 行使用到了水平线标签<hr />，该标签在网页中显示一条水平分割线。

(2) 强调文本。有些时候，我们会希望对某些文本进行必要的强调，或者标记出需要在 CSS 中强调的文本。

- 粗体标签：...
- 斜体标签：...

【例 2-5】强调文本(empha.html)。

```
<!DOCTYPE html>
<html>
<head>
<title>青玉案・元夕</title>
</head>
<body>
<b>《青玉案・元夕》</b>为南宋著名词人<b>辛弃疾</b>所作，词从极力渲染元宵节绚丽多彩的热闹场面入手，反衬出一个<i>孤高淡泊、超群拔俗</i>、不同于金翠脂粉的女性形象，寄托着作者政治失意后，不愿与世俗同流合污的孤高品格。
</body>
</html>
```

在浏览器中结果如图 2-6 所示。

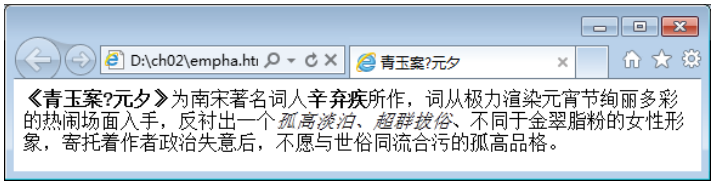

图 2-6 强调文本

(3) 标题文本。一般文章都有标题、章和节等结构，HTML 中提供了标题标签<h1>、<h2>、<h3>、<h4>、<h5>和<h6>，<h*n*>中的 *n* 表示标题级别，*n* 值越小，标题字号越大。标题标签也是成对出现的。

【例 2-6】标题标签的使用(headline.html)。

```
<!DOCTYPE html>
<html>
<head><title>标题标签的使用</title></head>
<body>
<h1>一级标题</h1>
<h2>二级标题</h2>
<h3>三级标题</h3>
<h4>四级标题</h4>
<h5>五级标题</h5>
<h6>六级标题</h6>
正文非标题文字
</body>
</html>
```

在浏览器中结果如图 2-7 所示。

2. 图像

适当的应用图像可以使网页变得赏心悦目、充满吸引力，网页中可以加入的图像格式有 JPG、GIF 和 PNG。

图 2-7 标题标签的使用

网页中插入图像的标签是<img>，它是一个单标签。<img>并不是真正地把图片加入到 HTML 文件中，而是通过一个路径告诉浏览器图像在哪里。其基本语法格式为：

```
<img src=图像路径 />
```

这里图像路径一般也使用相对路径，即图像文件相对于 html 文档的路径，<img>标签的常用属性如表 2-1 所示。

表 2-1 <img>标签的属性

属　性	说　明
src	图像文件的路径，一般使用图像文件相对于网页文档的相对路径
alt	图像的说明文字，在网页不能正常显示时出现该文字，另外在浏览器中，当鼠标经过并短暂停留时，也会显示出该说明文字
width 和 height	图像的高度和宽度，单位是像素，默认为图像的实际大小

下面要在网页中加入图像 flower1.jpg、flower2.jpg 和 flower3.jpg。首先，要将图像置入 D:\ch02\images 中，然后在网页中加入，加入其他几幅图像方法类似。

【例 2-7】在网页中加入图像(image.html)。

```
<!DOCTYPE html>
<html>
<head><title>图像应用</title></head>
<body>
<img src="images/flower1.jpg" alt="牵牛花" width="300" height="225" />
<img src="images/flower2.jpg" alt="牵牛花" width="300" height="225" />
<img src="images/flower3.jpg" alt="牵牛花" width="300" height="225" />
<hr />
牵牛花属于旋花科牵牛属，一年或多年生草本缠绕植物。这一种植物的花酷似喇叭状，因此有些地方叫它做喇叭花。种植牵牛花一般在春天播种，夏秋开花，其品种很多，花的颜色有蓝、绯红、桃红、紫等，亦有混色的，花瓣边缘的变化较多，是常见的观赏植物。果实卵球形，可以入药。牵牛花叶子三裂，基部心形。花呈白色、紫红色或紫蓝色，漏斗状，全株有粗毛。花期以夏季最盛。种子具有药用价值。
```

```
</body>
</html>
```

预览效果如图 2-8 所示。

图 2-8 在网页中加入图像

3. 超链接

超级链接(HyperLink)习惯上称为超链接。给网页上的文本或图像设置超链接，可以使得从源点跳到目标点。

基本语法如下：

```
<a href=链接目标地址>文字或图像</a>
```

语法说明如下：

- 在 HTML 文件中，超链接目标可以分为内部链接和外部链接。所谓内部链接，指网站内部文件之间的链接，此类链接的地址一般使用相对地址，即链接目标文件相对于该网页文件的路径地址。所谓外部链接，指网站内的文件链接到站点外文件的链接，外部链接一般使用绝对地址，比如链接到新浪，地址就是 http://www.sina.com.cn。
- 链接的源点可以是文字或者图像。
- 另外，比较特殊的是在网页上添加一个电子邮件链接。添加电子邮件链接，只需要在希望链接的电子邮件地址的文字前后分别加上和即可。在浏览网页时，单击链接文字就可以打开本机上默认的电子邮件收发软件。

【例 2-8】超链接(link.html)。

```
<!DOCTYPE html>
<html>
<head><title>超链接</title></head>
<body>
<p>
汉宫春  立春日 <br /><br />
春已归来，看美人头上，袅袅春幡。<br />
无端风雨，未肯收尽余寒。<br />
```

```
年时燕子，料今宵梦到西园。<br />
浑未办黄柑荐酒，更传青韭堆盘<br />
<br />
却笑东风从此，便薰梅染柳，更没些闲。<br />
闲时又来镜里，转变朱颜。<br />
清愁不断，问何人会解连环？<br />
生怕见花开花落，朝来塞雁先还。
</p>
<a href="images/han2.jpg">
    <img src="images/han1.jpg" alt="点击查看原图" width="300" target="_blank" />
</a>
<p>文字来源:<a href="http://baike.baidu.com">百度百科</a></p>
<p>辛弃疾的其他作品<br />
<a href="pre.html" target="_blank">青玉案  元夕</a><br />
<a href="chounuer.doc">丑奴儿  书博山道中壁</a> <br />
<a href="images/han2.jpg" target="_blank">文中配图</a>
</p>
<p><a href="mailto:meihao@163.com">联系我们</a></p>
</body>
</html>
```

网页显示效果如图 2-9 所示。

图 2-9　在网页中加入链接

该网页存在站点根目录 D:\ch02，在该网页中加入了 6 个链接：

- 第 1 个链接 ，链接源点为图像 han1.jpg，目标点为图像 han2.jpg，这两幅图像都存储在 D:\ch02\images 中。

- 第 2 个链接百度百科，链接源点为文字“百度百科”，目标点为外部网址 http://baike.baidu.com。
- 第 3 个链接青玉案 元夕，链接源点为文字“青玉案 元夕”，目标点为 D:\ch02 中的内部网页 pre.html。
- 第 4 个链接丑奴儿 书博山道中壁，链接源点为文字“丑奴儿 书博山道中壁”，目标点为 D:\ch02\images 中的 word 文档 chounuer.doc。
- 第 5 个链接文中配图，链接源点为文字“文中配图”，目标点为 D:\ch02\images 中的图像 han2.jpg。
- 第 6 个链接联系我们为电子邮件链接。

链接属性中 target="_blank"表示在新窗口中打开链接目标。如果没有设置目标属性或者设置为 target="_self"，则在当前窗口打开链接。

4. 表格

表格曾经是网页中最常使用的排版方法，DIV+CSS 排版布局方法出现后，表格更多用来显示各种数据，在这里我们只简单介绍一下。

表格所涉及的标签有“表格标签”“行标签”和“单元格标签”。

- <table>…</table>：表格标签，表示一个表格的开始和结束。
- <tr>…</tr>：行标签，成对出现，包含在表格标签之间，有几对行标签说明该表格有几行。
- <td>…</td>：单元格标签，成对出现，包含在行标签之间，有几对单元格标签说明该行有几个单元格，另外<th>是特殊的单元格，表示该单元格为标题，即表头。

表格的 3 个标签在 HTML4 之前有很多属性，在 HTML5 中基本都不再支持，表 2-2 给出了 HTML5 中表格标签的属性。

表 2-2 表格相关标签的属性

属 性	说 明
border	<table>标签的属性，border 属性只允许使用值 1 或 0；表示是否显示表格的边框
colspan	<td>标签的属性，表示跨多列，如 colspan=3 表示该单元格跨 3 列，即横向合并 3 个单元格
rowspan	<td>标签的属性，表示跨多行，rowspan=2 表示该单元格跨 2 行，即纵向合并 2 个单元格

【例 2-9】表格示例(table.html)。

```
<!DOCTYPE html>
<html>
<head><title>表格应用</title></head>
<body>
<table border="1">
    <tr>
    <th colspan="5">学生登记表</th>
</tr>
<tr>
```

```
<th>学号</th><th>姓名</th><th>年龄</th><th>专业</th><th>照片</th>
</tr>
<tr>
<td>201303001</td><td>张小凡</td><td>18</td><td>软件开发</td>
<td><img src="images/photo1.jpg" /></td>
</tr>
<tr>
<td>201303002</td><td>碧瑶</td><td>17</td><td>计算机多媒体技术</td>
<td><img src="images/photo2.jpg" /></td>
</tr>
<tr>
<td>201303003</td><td>陆雪琪</td><td>16</td><td>计算机应用</td>
<td><img src="images/photo3.jpg" /></td>
</tr>
</table>
</body>
</html>
```

预览结果如图 2-10 所示。

图 2-10　表格应用

2.1.3　表单

表单是网页中提供的一种交互式操作手段，在网页中的使用十分广泛。无论是提交搜索的信息，还是网上注册等都需要使用表单。用户可以通过提交表单信息与服务器进行动态交流。表单主要可以分为两部分：一是 HTML 源代码描述的表单；二是提交后的表单处理，需要调用服务器端编写好的 JSP 等代码对客户端提交的信息作出回应。该部分仅讲述 HTML 源代码描述的表单。

在 HTML 中，在需要使用表单的地方插入成对的表单标签<form></form>即可。

基本语法如下：

```
<form name=" "   method=" "   action=" "   enctype=" " >
表单项、文字、图片等
</form>
```

其中，name 属性表示表单的名称；method 属性指定传输方式，可以选择 post 或 get(在 7.5 节中详细介绍两者的区别)；action 属性用来指定接纳表单数据的 JSP 页面或者 Servlet，如果该属性为空则提交给当前页面；enctype 属性指定传送数据的编码方式，默认值是

application/x-url-encoded，利用表单上传文件时，需要改变编码方式，此时需要配合 post 方法。

1. 常用表单项

表单最重要的作用是获取用户信息，这就需要在表单中加入表单项(控件)，例如文本框、单选按钮等，常用的表单项如表 2-3 所示。

表 2-3　常见表单项

表单项	说　　明
<input type="text">	单行文本框
<input type="password">	密码文本框
<input type="submit">	提交按钮，将表单里的信息提交给表单里 action 所指向的地址
<input type="image">	图片提交
<input type="reset">	重置按钮，重设表单内容
<input type="button">	普通按钮
<input type="hidden">	隐藏元素
<input type="radio">	单选按钮
<input type="checkbox">	复选框
<input type="file">	文件域
<select>...</select>	列表框
<textarea>...</textarea>	多行文本框

(1) 单行文本框。单行文本框允许用户输入一些简短的单行信息，例如用户姓名、住址等。其基本语法如下所示：

```
<input type=text name="名称" size="数值"value="预设内容" maxlength="数值" />
```

- name：设定文本框的名称，数据交互中会用到。
- size：设定此文本框显示的宽度。
- value：设定此文本框的预设内容。
- maxlength：设定此文本框输入的最大长度。

(2) 密码文本框。密码文本框主要用于一些保密信息的输入，例如密码。其基本语法如下所示：

```
<input type="password"    name="名称" size="数值" value="预设内容" maxlength="数值" />
```

密码文本框用法基本与单行文本框一样，其属性也类似于单行文本框，在此不予重复。

(3) 提交按钮。通过提交按钮可以将表单里的信息提交给表单 action 所指向的文件地址。其基本语法如下所示：

```
<input type="submit"    name="名称" value="预设内容" />
```

- name：设定提交按钮的名称。
- value：设定按钮上显示的文本，默认为“提交”。

(4) 图片提交。“图片提交”作用和“提交按钮”相同，不同的是“图片提交”以一个图片作为表单的提交按钮，其中 src 属性表示图片的路径。其基本语法如下所示：

```
<input type="image"   src="图片路径" name="名称" alt="替代文本" width="宽度" height="高度" />
```

- name：设定提交图片的名称。
- src：图像路径，一般使用相对路径。
- alt：鼠标经过图像或者图像不显示时的替换文本。
- width：设定图像的宽度。
- height：设定图像的高度。

例如：

```
<input type=" image" src="images/sub.gif" name=" imageField" alt="点此提交" />
```

(5) 重置按钮。重置按钮是表单中另外一个比较常用的按钮，其作用是重置用户填写的信息。其基本语法如下所示：

```
<input type="reset"   name="名称" value="预设内容">
```

- name：设定提交按钮的名称。
- value：设定按钮上显示的文本，默认为“重置”。

(6) 普通按钮。另外，表单中经常用到普通按钮，它没有默认的动作，有时需要利用 JavaScript 来做一些特殊的效果时使用。其基本语法如下所示：

```
<input type="button"   name="名称" value="预设内容">
```

除了按钮的显示文字外，可以在按钮上添加很多效果，特别是单击按钮后发生的事件等，如图 2-11 所示是一个单击后会有对话框提示的按钮，下面是对应的 HTML 源代码：

```
<input type="button" name="button1" value="单击进入" onclick="alert('你单击按钮了')" />
```

图 2-11　按钮示例

(7) 隐藏元素。隐藏元素多用于在提交表单时向服务器传递一些不需要用户设定但程序必需的参数值。这在动态网页中的需求更加明显。基本语法如下所示：

```
<input type="hidden" name="参数名称" value="参数取值">
```

隐藏元素一般位于<form></form>标签内，在表单提交时一同被发送给服务器端，下面是一个隐藏元素的使用示例，功能是在表单提交时，将用户的 IP 地址和用户所在的地区传送给服务器端，这种数据传递用户往往是不会发觉的。

```
<input type="hidden" name="userIP" value="123.123.123.123">
<input type="hidden" name="region" value="northeast">
<input type="submit" value="注册">
```

(8) 单选按钮。单选按钮通常是给出几个选项供用户选择，一次只能从中选一个，应用单选按钮时要确定显示给用户的文字和不同选项的取值。其基本语法如下所示：

```
<input type="radio" name="名称" value="选项内容" checked="checked">
```

- value：设定此单选按钮的选定值。
- checked：当该项默认被选中时设定，否则不设定。

(9) 复选框。复选框通常是给出几个选项供用户选择，并且可以从中选择多个，使用复选框时也要确定显示给用户的文字和不同选项的取值。其基本语法如下所示：

```
<input type="checkbox" name="名称" value="选项内容" checked="checked">
```

- value：设定此复选框的选定值。
- checked：当该项默认被选中时设定，否则不设定。

(10) 文件域。文件域是用来填写文件路径，通过表单上传文件的地方。

```
<input type="file" name="名称">
```

(11) 列表框。列表框是表单中供用户选择的一个表单项，可以显示多个选项供选择，且用户能同时选择其中的一个或多个。列表框中包含<option>标签。其基本语法如下所示：

```
<select name="名称 " size="大小" multiple="multiple">
<option value=" "></option>
<option value=" "></option>
…
</select>
```

- name：设定下拉列表的名称。
- size：设定下拉列表显示选项的个数。
- multiple：设定此下拉列表可多选，如果为单选则省略该项。
- value：<option>的属性，当选择该项时的值。

<select>中 size 属性指整个选项框内显示的选项个数，默认为 1，当 size 值大于 1 时，列表框会改变形式，显示出多个列表项。一般情况下，列表框都是用在单选的情况下，如果要使用多选的功能，只要加上 multiple="multiple"就可以了。

(12) 多行文本框。多行文本框用来输入较多的文字信息，常在新闻发布与论坛等系统中用到。其基本语法如下所示：

```
<textarea name="名称" rows="文本框的显示行数" cols="文本框的显示列数"></textarea>
```

- rows：文本框显示的行数。
- cols：文本框显示的列数。

使用多行文本框时主要是确定它的名称以及大小(行数与列数)，当用户输入的文字超过显示容量时，多行文本框会自动产生滚动条。

【例 2-10】表单应用(form.html)。

```
<!DOCTYPE html>
<html>
<head><title>表单应用</title></head>
<body>
<form action="" method="post" enctype="multipart/form-data" name="form1">
<table border="1">
    <tr>
      <td colspan="2">会员注册</td>
    </tr>
    <tr>
      <td>用户名：</td>
      <td><input type="text" name="userid" /></td>
    </tr>
    <tr>
      <td>密码：</td>
      <td><input type="password" name="pass" /></td>
    </tr>
    <tr>
      <td>确认密码：</td>
      <td><input type="password" name="pass2" /></td>
    </tr>
    <tr>
      <td>性别：</td>
      <td>
        <input type="radio" name="gender" value="male" />男
        <input type="radio" name="gender" value="female" />女
      </td>
    </tr>
    <tr>
      <td>爱好：</td>
      <td><input type="checkbox" name="checkbox" value="001" />体育
        <input type="checkbox" name="checkbox2" value="002" /> 音乐
        <input type="checkbox" name="checkbox3" value="003" /> 文学
        <input type="checkbox" name="checkbox4" value="004" /> 其他</td>
    </tr>
    <tr>
      <td>所在城市：</td>
      <td><select name="select">
        <option value="01">北京</option>
        <option value="02">上海</option>
        <option value="03">天津</option>
        <option value="04">重庆</option>
      </select></td>
    </tr>
    <tr>
      <td>照片：</td>
      <td><input type="file" name="photo"><input type="button" value="上传"></td>
    </tr>
    <tr>
      <td>备注：</td>
```

```
            <td><textarea name="textfield" rows="3"></textarea></td>
        </tr>
        <tr>
            <td> </td>
            <td><input type="submit" name="Submit" value="提交">
                  <input type="reset" name="Submit2" value="重置"></td>
        </tr>
    </table>
    </form>
    </body>
    </html>
```

在浏览器中预览效果如图 2-12 所示。

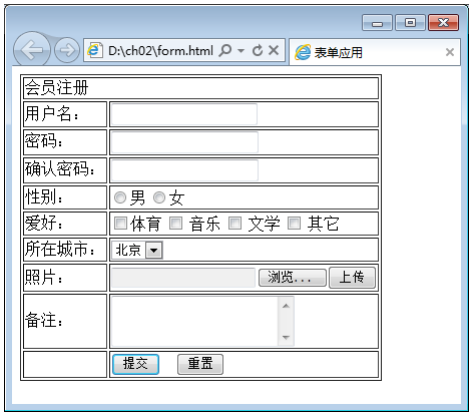

图 2-12 表单示例

2. HTML5 新增表单项

HTML5 中定义了很多全新的表单输入类型，这些新的表单类型为网页设计提供了更好的输入控制和验证方法，表 2-4 列举了一些新的表单项。

表 2-4 HTML5 新增表单项

表单项	说　明
<input type="color">	颜色选择器
<input type="date">	日期选择器
<input type="datetime">	日期时间选择器
<input type="month">	月份选择器
<input type="time">	时间选择器
<input type="week">	周选择器
<input type="email">	邮件输入框
<input type="number">	数字输入框，可以设置输入值的范围
<input type="range">	数字滑动条，通过拖动滑动条改变一定范围内的数字
<input type="search">	搜索输入框
<input type="tel">	电话号码输入框
<input type="url">	输入 URL 地址的文本框

【例 2-11】新增表单项的应用(form2.html)。

```
<!DOCTYPE html>
<html>
<head>
<title>新增的表单项</title>
</head>
<body>
<form>
输入您的出生日期：<input type="date" /> <br /><br />
输入您的 E-mail：<input type="email" name="user_email" /><br /><br />
输入您的身高(cm)：<input type="number" min="50" max="240" /><br /><br />
选择您喜欢的颜色：<input type="color" /><br /><br />
<input type="submit" />
</form>
</body>
</html>
```

新增表单项的浏览器支持性还不太好，在 Opera 12.15 浏览器中显示结果如图 2-13 所示。日期和日期选择项都支持用户直接选取，E-mail 项如果用户输入不正确，在提交表单时会有提示。

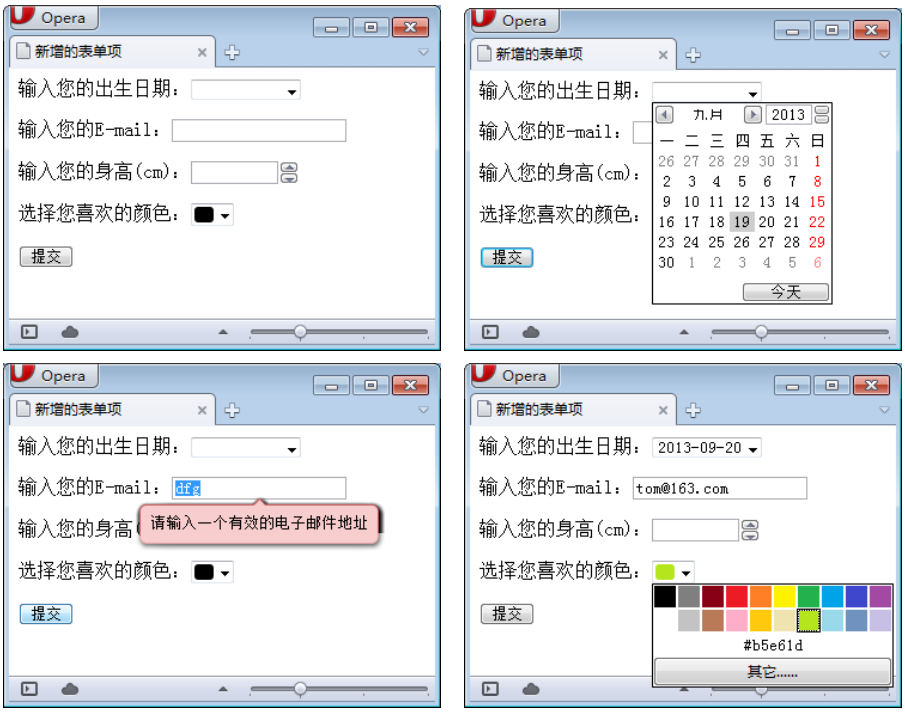

图 2-13 新增表单项示例

3. <input>标签的常用属性

这一节我们讲解几个<input>标签非常有用的属性，如表 2-5 所示。

表 2-5 <input>标签的属性

属　性	说　明
autofocus	用法<input autofocus="autofocus" />，表单项在页面加载后自动地获取焦点
multiple	用法<input multiple="multiple" />，设置输入框可以同时选中多个输入值
placeholder	用法<input placeholder="提示文本" />，可以在输入域内给浏览者显示一段提示语句，当输入域获得焦点时，也就是当用户将光标定位进去后，这种提示就会消失，从而让用户输入自己的内容。placeholder 属性适用于以下几种类型的<input>标签：text、url、telephone、search、email 以及 password
required	用法<input required="required" />，规定其所在的输入域在提交之前不能为空，否则不能提交。
pattern	适用于以下类型的 <input> 标签：text、telephone、url、search、email 以及 password。pattern 属性的作用相当于给 input 输入域加上一个验证模式，这个验证模式(pattern)是一个正则表达式

【例 2-12】<input>标签的属性(form3.html)。

```
<!DOCTYPE html>
<html>
<head>
<title>input 的常用属性</title>
</head>
<body>
<form>
必须输入的项：<input type="text" required="required" /> <br /> <br />
自动获取焦点：<input type="text" autofocus="autofocus" /> <br /> <br />
6~9 个英文字母：<input type="text" pattern="[A-Za-z]{6,9}" /> <br /> <br />
提示文本：<input type="text" placeholder="网页设计" /> <br /> <br />
<input type="submit" />
</form>
</body>
</html>
```

【例 2-12】在 Opera 12.15 浏览器中显示效果如图 2-14 所示。

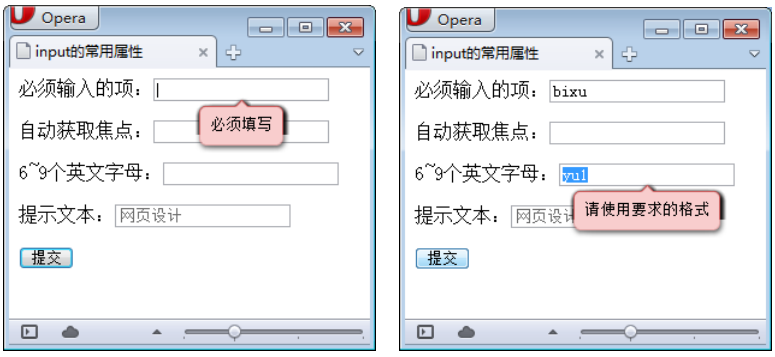

图 2-14　<input>标签的属性示例

2.1.4 XML 与 XHTML

1. XML

XML(Extensible Markup Language，可扩展标签语言)与 HTML 一样，都是 SGML(Standard Generalized Markup Language，标准通用标签语言)。XML 是 Internet 环境中跨平台的、依赖于内容的技术，是当前处理结构化文档信息的有力工具。XML 是一种简单的数据存储语言，使用一系列简单的标签描述数据，而这些标签可以很方便地建立。虽然 XML 比二进制数据要占用更多的空间，但 XML 极其简单并易于掌握和使用。

XML 是一种元标签语言，没有许多固定的标签，为 Web 开发人员提供了更多的灵活性。当使用 HTML 时，标签只是简单地表示内容的显示形式，与表示的内容没有任何关联，会给文档的进一步处理带来极大的不便。比如要表示个人简历，用 HTML 的表示方式如下(文件名 resume.html)：

```
<html>
<body>
<table width="400" border="1" bordercolor="#336666">
  <tr>
    <th>姓名</th><th>性别 </th><th>出生日期</th><th>专业</th>
  </tr>
  <tr>
    <td>张三</td><td>男</td><td>1991.8.12</td><td>软件开发</td>
  </tr>
</table>
</body>
</html>
```

这里无法从<th>、<td>等标签得知其内容表示什么，如果用 XML，相应的文档(文件名 resume.xml)就可以写成如下形式：

```
<?xml version="1.0" encoding="gb2312"?>
<?xml:stylesheet type="text/xsl" href="resume.xsl"?>
<resume>
<name>张三</name>
<sex>男</sex>
<birthday>1991.8.12</birthday>
<major>软件开发</major>
</resume>
```

这里，version 规定了 XML 文档的版本，encoding 规定了 XML 文档的编码类型，此处取值 gb2312 表示简体中文。对比两个例子，使用 XML 可以做到自定义标签，用标签表明内容的含义。这就为在网络上交流资料时用计算机处理文档提供了极大的方便，也增加了源文件的可读性。当然 resume.xml 能以表格形式运行的前提是必须自定义好形式，定义其显示形式的文件如下(文件名 resume.xsl，在 xml 文件的第二行引入了该 xsl 文件)：

```
<?xml version="1.0" encoding="GB2312"?>
<html xmlns:xsl="http://www.w3.org/TR/WD-xsl">
```

```
<body>
<xsl:for-each select="resume">
<table width="400" border="1" bordercolor="#336666">
<tr>
<th>姓名</th><th>性别</th><th>生日</th><th>专业</th>
</tr>
<tr>
<td><xsl:value-of select="name"/></td>
<td><xsl:value-of select="sex"/></td>
<td><xsl:value-of select="birthday"/></td>
<td><xsl:value-of select="major"/></td>
</tr>
</table>
</xsl:for-each>
</body>
</html>
```

XSL 之于XML ，就像 CSS 之于 HTML。它是指可扩展样式表语言 (Extensible Stylesheet Language)，这是一种用于以可读格式呈现 XML 数据的语言。有了 resume.xsl 定义显示形式，文件 resume.xml 在浏览器中显示的形式和 resume.html 形式相同，如图 2-15 所示，不过它给程序员提供了更大的灵活性，也更易于数据交换。

图 2-15　浏览 xml 文件

2. XHTML

可扩展超文本标签语言(Extensible Hypertext Markup Language，XHTML)与 HTML 类似，不过语法上更加严格。HTML 语法要求比较松散，这样对网页编写者来说比较方便，但对于机器来说，语言的语法越松散，处理起来就越困难。传统的计算机还有能力兼容松散语法，但对于许多其他设备，比如手机，难度就比较大。因此产生了由 DTD 定义规则，语法要求更加严格的 XHTML， XHTML 1.0 在 2000 年 1 月 26 日成为 W3C 的推荐标准。当然，HTML5 是用于取代 1999 年所制定的 HTML 4.01 和 XHTML 1.0 标准的 HTML 标准版本，HTML5 在语法上与 XHTML 一样追求严谨。

和 CSS(Cascading Style Sheets，层叠样式表)结合后，XHTML 才发挥真正的威力，在实现样式和内容分离的同时，又能有机地组合网页代码，在另外的单独文件中，还可以混合各种 XML 应用。从 HTML 到 XHTML 过渡的变化比较小，主要是为了适应 XML，最大的变化在于文档必须是良构的。具体变化如下：

(1) 所有的标签都必须要有一个相应的结束标签。在 HTML 中，可以有开放标签，例如 标签可以省略其对应的 标签，但在 XHTML 中这是不合法的。XHTML 要求有严谨的结构，所有标签必须关闭。如果是单独不成对的标签，在标签最后加一个“/”来关闭它，例如
。

(2) 所有标签的元素和属性的名字都必须使用小写。

(3) 所有的属性必须用引号("")括起来。在 HTML 中，属性值可加双引号，也可以不加，但是在 XHTML 中，属性必须加引号。例如<hr width=80%>必须修改为：<hr width="80%">。

(4) 所有的 XHTML 标签都必须合理嵌套。 XHTML 要求所有的嵌套都必须按顺序，也就是说，一层一层的嵌套必须是严格对称，不能出现交叉。例如 ... 就是错误的，而 ... 才是合法的。

(5) 把所有“<”和“&”等特殊符号用编码表示。 任何“<”不是标签的一部分，都必须被编码为“<”，任何“>”不是标签的一部分，都必须被编码为“>”，任何“&”不是实体的一部分的，都必须被编码为“&”。

(6) 给所有属性赋一个值。XHTML 规定所有属性都必须有一个值，没有值的就重复本身。例如<input type="radio" name="gender" value="female" checked> 必须修改为<input type="radio" name="gender" value="female" checked="checked">。

(7) 不能在注释内容中使用“--”。“--”只能发生在 XHTML 注释的开头和结束，也就是说，在内容中它们不再有效。例如<!--这里是注释----------这里是注释-->是无效的，这里必须用等号或者空格等替换内部的虚线，<!--这里是注释==========这里是注释-->是合法的。

(8) 图像必须有替代文字。每个图像标签都必须有 alt 说明文字。例如： 。

2.2 CSS 技术

CSS(Cascading Style Sheet，层叠样式表，简称样式表)是一种用来定义网页外观格式的技术。在网页中引入 CSS，可以方便有效地对页面进行样式设计，更加精确地控制网页的字体、颜色、背景等网页元素的效果，更重要的是 DIV+CSS 技术已经是目前主流的网页布局技术。

CSS3 是 CSS 技术的一个升级版本，CSS3 语言将 CSS 划分为更小的模块，在 CSS3 中有字体、颜色、布局、背景、定位、边框、多列、动画、用户界面等等多个模块。

2.2.1 CSS 基本语法

CSS 由 3 个基本部分组成：样式选择器、属性和属性值。

基本语法如下：

```
样式选择器{属性 1:属性值; 属性 2:属性值;…属性 n:属性值;}
```

这里简单介绍几种常用的选择器类型及其基本语法。

1. 标签选择器

标签选择器可以同时控制所有此类标签的风格控制，达到了站点风格高效统一的目的，例如，需要将所有段落文字的字号改为 14 像素，只需要在 CSS 文件中定义代码：

```
p{font-size:14px;}
```

如果在网页中引用上述样式表，则页面中所有段落文字都将受到这种样式的控制，字号都显示为 14 像素。

2. 类选择器

类选择器的基本语法如下：

```
标签名.类名{属性 1:属性值 1;属性 2:属性 2;…}
```

这里，类名由设计者定义，标签名在使用过程中可以改为*表示全部，也可以省略。例如：

```
p.s1{color:green;}
```

表示样式 s1 仅适用于段落元素，其使用方法如下：

```
<p class="s1">段落文字</p>
```

此引用表示该段落的文字表现为绿色，而未引用样式 s1 的段落则不会受其影响。例如：

```
.s2{color:red;}
```

表示样式 s2 适用于任何元素，其使用方法如下：

```
<p class="s2">段落文字</p>
<h1 class="s2">一级标题</h1>
```

示例中引用了样式 s2 的段落文字和一级标题文字都表现为红色。

3. ID 选择器

ID 选择器和类选择器非常相似，不同的是定义时不使用“.”而使用“#”，引用时不使用 class 属性而使用 id 属性。一般情况页面元素的 id 是唯一的，因此 ID 选择器一般只针对某个特定的元素作用一次，不推荐重复使用。ID 选择器的基本语法如下：

```
标签名#id 名{属性 1:属性值 1;属性 2:属性 2;…}
```

这里，id 名由设计者定义，标签名在使用过程中可以改为*表示全部，也可以省略。例如：

```
#s3{color:red;font-size:24px;}
```

表示样式 s3 适用于任何元素，其使用方法如下：

```
<h2 id="s3">二级标题</h2>
```

示例中引用了样式 s3 的二级标题表现为 24 像素大小的红色字体。

2.2.2 在 HTML 文档中使用 CSS 的方法

在 HTML 文档中引用 CSS 样式表的方法有 4 种：行内样式、内嵌样式、链接外部样式和导入外部样式。

1. 行内样式

行内样式是直接在 HTML 元素中加入了 style 属性，然后把 CSS 代码直接写入其中即可。该方法的优点是使用方法简单，缺点是不能真正实现内容和样式的分离，使用效率低下。其基本语法如下。

```
<标签 style="样式属性:样式属性值;样式属性:样式属性值;... ">
```

例如：

```
<body style="background-color:#909090;">
```

2. 内嵌样式

内嵌样式是一种比较常用的样式，将 CSS 样式直接定义在网页的<head>部分，其基本语法如下：

```
<style type="text/css">
<!--
        选择器 1{样式属性:样式属性值;…}
  选择器 2{样式属性:样式属性值;…}
  ……
  选择器 n{样式属性:样式属性值;…}
-->
</style>
```

这里<style>标签表示声明样式表内容，type 用来指定元素中的文本属性。“<!--”和“-->”是注释标签，作用是当某个浏览器不支持 CSS 时可以将其视为注释，不过目前主流浏览器都对 CSS 有很好的支持，因此，注释标签也可省略。

【例 2-13】行内样式和内嵌样式示例(inline.html)。

```
<!DOCTYPE html>
<html>
<head>
<title>内嵌样式例子</title>
<style type="text/css">
body{
            background-color:#fef273;     /*设置背景色*/
            font-size:18px;
}
</style>
</head>
<body>
<h1 style="background-color:#48480c;color:white;font-size:28px;text-align:center;">青玉案  元夕</h1>
```

```
<p>东风夜放花千树，更吹落，星如雨。宝马雕车香满路。凤箫声动，玉壶光转，一夜鱼龙舞。<br />蛾儿雪柳黄金缕，笑语盈盈暗香去。众里寻她千百度，蓦然回首，那人却在灯火阑珊处。
</p>
</body>
</html>
```

该示例在 head 部分为 body 元素设置了样式，在行内为 h1 定义了样式，在 IE9.0 中的浏览效果如图 2-16 所示。

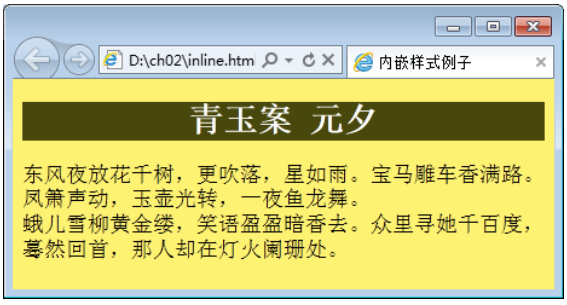

图 2-16　行内样式和内嵌样式示例

3. 链接外部样式

链接外部样式表是在网页中调用已经定义好的外部样式表的方法之一，其基本语法如下：

```
<link href="样式表路径" rel="stylesheet" type="text/css" />
```

这里 href 指出外部样式表存放的路径，rel 用来定义链接的文件与 HTML 之间的关系，stylesheet 值表示指定一个固定或首选的样式。

【例 2-14】链接外部样式示例。

外部样式 CSS 文件(css.css)。

```
.s1{
        background-color:#705968;      /*设置背景色*/
        color:#FFFFFF;                 /*设置前景色*/
        text-align:center;             /*居中*/
}
```

链接外部样式示例 HTML 文件(exter1.html)

```
    <!DOCTYPE html>
<html>
<head>
<link href="css.css" rel="stylesheet" type="text/css" />
<title>链接外部样式表示例</title>
</head>
<body>
<h1 class="s1">链接外部 CSS</h1>
</body>
</html>
```

该示例首先定义好样式表文件 css.css 文件，然后在 HTML 文件的 head 元素内通过 link 元素来调用 css.css 文件，其在 IE9.0 中的浏览效果如图 2-17 所示。

图 2-17　外部链接样式示例

4. 导入外部样式

导入外部样式表与链接外部样式的功能基本相同，都是使用外部样式表文件，只是在语法和运行方式上稍有区别。

基本语法如下：

```
@import url("样式表路径");
```

在以下示例中，我们将【例 2-14】中的 CSS 文件 css.css 导入到【例 2-15】的 html 文件中。

【例 2-15】导入样式示例(exter2.html)。

```
<!DOCTYPE html>
<html>
<head>
<style type="text/css">
@import url(css.css);
</style>
<title>导入样式示例</title>
</head>
<body>
<h1 class="s1">导入外部 CSS</h1>
</body>
</html>
```

该示例在 IE9.0 中的浏览效果与上一个示例相同，不再赘述。

2.2.3　常用 CSS 属性

本节我们简单介绍几个常用的 CSS 属性，如字体、颜色、背景、边框等效果。

1. CSS 字体和段落

字体和段落相关的 CSS 属性很多，最常用的如表 2-6 所示。

表 2-6　常用 CSS 字体和段落属性

基 本 语 法	说　明
font-family :字体 1, 字体 2, ……;	设置字体，可同时设定一个或者多个
font-size: 绝对尺寸 | 相对尺寸 | 长度 | 百分比 ;	设置字体大小，最常用的是用长度值指定文字大小，长度单位有 px(像素)、pt(点)和 em(字体高)等
font-weight: normal | bold | bolder | lighter | 数值;	设置字体粗细，normal、bold、bolder 和 lighter 分别代表正常、粗体、加粗和细体，数值是 100~900 的整百数字
line-height: normal | 长度值 | 百分比 | 数值 ;	设置行高，比如字体大小为 12px 时，值 18px、150%和 1.5 都表示 1.5 倍行高
text-indent: 长度值 | 百分比;	设置文本缩进，如 text-indent:2em 表示缩进 2 个字体高
text-align: left | right | center | justify;	水平对齐方式，常用的有左、中、右和两端对齐
text-decoration:值; 或 text-decoration-line:值;	文本修饰线，常用值有 none(没有修饰线)、underline(下划线)、overline(上划线)和 line-through(删除线)

【例 2-16】CSS 字体和段落示例(font.html)。

```
<!DOCTYPE html>
<html>
<head>
<title>换行标签和段落标签</title>
<style type="text/css">
.headline{
        font-size:28px;          /*字体大小*/
        text-align:center;       /*居中对齐*/
        font-weight:900;         /*字体加粗*/
        font-family:黑体,隶书;    /*设置字体*/
}
.text{
        font-size:16px;
        line-height:1.5;
        text-indent:2em;
}
</style>
</head>
<body>
<p class="headline">老木匠的房子</p>
<hr />
<p class="text">有个老木匠准备退休，回家与妻子儿女享受天伦之乐。<br/>老板舍不得他的好工人走……。
<br/>“这是你的房子，”他说，“我送给你的礼物。”</p>
<p class="text">他震惊得目瞪口呆，……</p>
</body>
</html>
```

该示例在 IE9.0 中的浏览效果如图 2-18 所示。

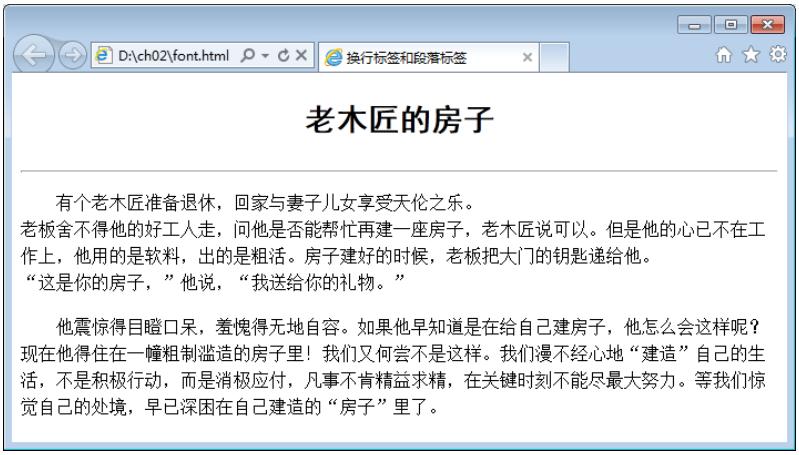

图 2-18　CSS 字体段落示例

2. CSS 颜色和背景

常用的颜色背景属性如表 2-7 所示。

表 2-7　常用 CSS 颜色和背景属性

基 本 语 法	说　　明
color: 颜色值;	设置前景色，常用的颜色表示方式有： ● 颜色英文名称，如 green 表示绿色 ● 十六进制记法，形式为#RRGGBB 或者#RGB，如#00FF00 表示绿色、如果每两位颜色值相同，可以简写为#RGB 形式，如#0F0 也表示绿色 ● RGB 函数，形式为 RGB(R,G,B)，R、G、B 可以为 0~255 的正整数或 0%~100%的百分数。例如 RGB(0,255,0)和 RGB(0%,100%,0%)都表示绿色
background-color: 颜色值; background-image: url(图像路径);	设置背景色 设置背景图像
background-repeat: repeat | no-repeat | repeat-x | repeat-y;	设置背景图像的重复方式，值分别表示重复、不重复、在水平方向重复和在垂直方向重复
background-attachment: scroll | fixed;	设置背景图像是否随背景滚动，scroll 为默认值，fixed 表示背景图像固定，背景图像不会跟随内容滚动
background-position: 关键字 | 百分比 | 长度;	背景图像位置，值常用关键字表示，关键字在水平方向上有 left、center 和 right，垂直方向主要有 top、center 和 bottom，水平方向和垂直方向可以相互搭配使用

除了上述属性外，需要特别注意，background 是其他背景相关属性的快捷属性还是复合属性，举例如下：

```
background:#FFFFCC;
```

可以用来设置背景颜色。

```
background: url(bg.gif) no-repeat fixed;
```

表示设置页面背景为图像 bg.gif，不重复，图像不随内容滚动，相当于如下代码：

```
background-image: url(bg.gif);
background-repeat: no-repeat;
background-attachment: fixed;
```

【例 2-17】CSS 颜色和背景示例(background.html)。

```
    <!DOCTYPE html>
<html>
<head>
<title>换行标签和段落标签</title>
<style type="text/css">
body{
    color:#434305;
    background:#dfd9b3 url(images/house.png) no-repeat right bottom fixed;
}
.headline{
        font-size:28px;          /*字体大小*/
        text-align:center;       /*居中对齐*/
        font-weight:900;         /*字体加粗*/
        font-family:黑体,隶书;    /*设置字体*/
}
.text{
        font-size:16px;
        line-height:1.5;
        text-indent:2em;
}
</style>
</head>
<body>
<p class="headline">老木匠的房子</p>
<hr />
<p class="text">有个老木匠准备退休，回家与妻子儿女享受天伦之乐。<br/>老板舍不得他的好工人走，……
<br/>“这是你的房子，”他说，“我送给你的礼物。”</p>
<p class="text">他震惊得目瞪口呆……</p>
</body>
</html>
```

该示例在 IE9.0 中的浏览效果如图 2-19 所示。

3. CSS 盒子属性

表 2-8 给出了 CSS 盒子相关的边框和宽高属性。

图 2-19　CSS字体段落示例

表 2-8　其他常用 CSS 属性

基 本 语 法	说　　明
width: auto | 长度 | 百分比; height: auto | 长度 | 百分比;	设置盒子大小
float: none | left | right;	浮动属性，改变元素块的显示方式，none 设置元素不浮动，left 设置元素浮在左边，right 设置元素浮在右边
border- style: 样式值{1,4}; border-top-style: 样式值; border-bottom-style: 样式值; border-left-style: 样式值; border-right-style: 样式值;	设置边框样式，样式值可以是 none(无轮廓)、hidden(隐藏边框)、dotted(点状轮廓)、dashed(虚线轮廓)、solid(实线轮廓)、double(双线轮廓)、groove(3D 凹槽轮廓)、ridge(3D 凸槽轮廓)、inset(3D 凹边轮廓)和 outset(3D 凸边轮廓)
border-width: 宽度值{1,4}; border-top-width: 宽度值; border-bottom-width: 宽度值; border-left-width: 宽度值; border-right-width: 宽度值;	宽度值可以是长度或关键字，关键字可以是 medium、thin 和 thick，分别表示默认厚度的边框、细边框和粗边框。border-top-width、border-bottom-width、border-left-width 和 border-right-width 属性分别用于设定上、下、左、右边框宽度，而 border-width 可同时设定 1 个或多个边框宽度
border-color: 颜色值{1,4}; border-top-color: 颜色值; border-bottom-color: 颜色值; border-left-color: 颜色值; border-right-color: 颜色值;	border-top-color 、 border-bottom-color 、 border-left-color 和 border-right-color 属性分别用于设定上、下、左、右边框颜色，而 border-color 可同时设定 1 个或多个边框颜色
border-left: [宽度] [样式] [颜色]; border-right: [宽度] [样式] [颜色]; border-top: [宽度] [样式] [颜色]; border-bottom: [宽度] [样式] [颜色]; border: [宽度] [样式] [颜色];	属性值中“宽度”、“颜色”和“样式”可同时设定 1 个或多个，值之间无特定顺序，用空格分隔

表中 border-color、border-style 和 border-width 可以设置 1~4 个值，如果提供全部 4 个参数值，将按上、右、下、左的顺序作用于 4 个边框；如果只提供 1 个值，将用于全部的 4 个边框；如果提供 2 个值，第 1 个用于上、下边框，第 2 个用于左、右边框；如果提供 3 个，第 1 个用于上边框，第 2 个用于左、右边框，第 3 个用于下边框。

首先我们使用 CSS 属性为前文表格设置表格大小、表格背景色、表格边框，以及图像的边框。

【例 2-18】CSS 盒子属性示例 1(box1.html)。

```
    <!DOCTYPE html>
<html>
<head>
<title>表格应用</title>
<style type="text/css">
body{
        width:500px;                    /*设置页面宽度*/
        margin:0 auto;                  /*设置页面边距*/
}
table{
        color:#434305;
        background-image:url(images/bg01.jpg);
        width:500px;                    /*设置表格宽度*/
        height:260px;                   /*设置表格高度*/
        border:10px outset #4c5d2d;     /*设置表格边框*/
}
img{
        border:white 6px solid;         /*设置图像边框*/
}
.headline{
        font-size:28px;                 /*字体大小*/
        text-align:center;              /*居中对齐*/
        font-weight:900;                /*字体加粗*/
        font-family:黑体,隶书;          /*设置字体*/
}
.text{
        font-size:16px;
        line-height:1.5;
        text-indent:2em;
}
</style>
</head>
<body>
<table border="1">
<tr><th colspan="5">学生登记表</th></tr>
<tr><th>学号</th><th>姓名</th><th>年龄</th><th>专业</th><th>照片</th></tr>
<tr><td>201303001</td><td>张小凡</td><td>18</td><td>软件开发</td>
<td><img src="images/photo1.jpg" /></td></tr>
<tr><td>201303002</td><td>碧瑶</td><td>17</td><td>计算机多媒体技术</td>
<td><img src="images/photo2.jpg" /></td></tr>
<tr><td>201303003</td><td>陆雪琪</td><td>16</td><td>计算机应用</td>
```

```
<td><img src="images/photo3.jpg" /></td></tr>
</table>
</body>
</html>
```

该示例中 margin:0 auto;设置页面上下边距为 0，左右边距自动，页面在 IE9.0 中的浏览效果如图 2-20 所示。

图 2-20　CSS 盒子示例 1

【例 2-19】CSS 盒子属性示例 2(box2.html)。

```
    <!DOCTYPE html>
<html>
<head>
<title> CSS 盒子属性示例 2</title>
<style type="text/css">
body{
        margin:0 auto;                  /*设置网页边距*/
        width:700px;
        font-size:13px;
}
div{
        box-sizing:border-box;          /*设置盒子显示模式*/
        background-color:#d3dec0;
        line-height:1.2;
}
.imgfloat{
        float:right;                    /*设置图像右浮动*/
}
a{
        text-decoration:none;           /*设置链接无下划线*/
        color:#7f5f52;
}
#banner{ width:100%; }                  /*页眉所在 div 样式*/
#left{                                  /*左边 div 样式*/
        width:200px;
        float:left;                     /*浮动*/
        padding:10px;                   /*设置盒子边框距内容的补白*/
```

```
        height:430px;
}
#right{                                    /*右边 div 样式*/
        width:500px;
        height:430px;
        padding:10px;
        float:right;                       /*浮动*/
}
</style>
</head>
<body>
<div id="banner"><img src="images/banner.jpg" /></div>
<div id="left">
<h4>辛弃疾其它作品</h4>
<a href="#">青玉案  元夕</a><br />
<a href="#">丑奴儿  书博山道中壁</a><br />
<a href="#">菩萨蛮  书江西造口壁</a><br />
<a href="#">清平乐  村居</a><br />
<a href="#">永遇乐  京口北固亭怀古</a><br />
<a href="#">西江月  夜行黄沙道中</a><br />
<a href="#">南乡子  登京口北固亭有怀</a>
</div>
<div id="right">
<p>此词上篇通过立春时节景物的描绘，隐喻当时南宋不安定的政局。开头“春已归来”三句，点明立春节候。按当时风俗，立春日，妇女们多剪彩为燕形小幡，戴之头鬓。故欧阳修《春日帖子》中有“共喜钗头燕已来”之句。“无端风雨”两句，既指自然界的气候多变，也暗指南宋最高统治集团惊魄不定、碌碌无为之态，宛如为余寒所笼罩。“年时燕子”三句，作者由春幡联想到这时正在北飞的燕子，可能已经把他的山东家园作为归宿了。“年时”即去年之意，这说明作者作此词时，离别他的家乡才只一年光景。接下去“浑未办”三句，是说作者新来异乡，生活尚未安定，春节到了，连旨酒也备办不起，更谈不到肴馔了。</p>
<p>词的下篇进一步抒发作者自己的忧国怀乡之情。“却笑东风从此”三句，作者想到立春之后，东风就会忙于吹送出柳绿花江的一派春光。“闲时又来镜里，转变朱颜”，语虽虚拟，实际表达了作者初归南宋急欲报国、收复失土的决心，深恐自己磋砣岁月，年华虚度。这里说的“清愁”，实际是作者的忧国忧民的情怀。
<img src="images/flower4.png" class="imgfloat" />
“解连环”，是用《战国策》秦昭王送玉连环给齐国王后，让她解开的故事。当时的齐王后果断机智地把玉连环椎破，使秦的诡计流于破产。但环顾当前，南宋最高统治集团中人，谁是能作出抗金的正确决策的智勇人物呢？“生怕”，即“甚怕”。“生怕见、花开花落，朝来塞雁先还。”表示作者对于恢复事业的担忧，深恐这一年的花由盛开又复败落，而失地却未能收复，有家仍难归去，言语流露出一丝的惆怅。</p>
</div>
</body>
</html>
```

此例中 padding:10px;设置边线距内容的距离，box-sizing:border-box;设置的是盒子在不同浏览器中的显示模式，请感兴趣的读者自行查阅。 标签是一个区块容器标签，即 与 之间相当于一个容器，可以容纳段落、标题、表格、图片各种 HTML 元素。可以通过设置 CSS 样式对 进行相应的控制，其中的各种标签元素会因此改变显示效果。页面显示效果如图 2-21 所示。

图 2-21　CSS 盒子示例 2

2.3 JavaScript 技术

JavaScript 是由 Netscape 公司开发的一种网页的脚本编程语言，它支持客户端与服务器端的应用程序以及构件的开发。JavaScript 是一种解释性的语言，它的基本结构形式与其他编程语言相似，需要先编译后执行。

2.3.1　JavaScript 语法

在这里只作简单介绍，在以后的例子里结合程序再具体解释其作用。

(1) 运算符

运算符就是完成操作的一系列符号，它有 7 类：赋值运算符(=，+=，-=，*=，/=，%=，<<=，>>=，|=，&=)、算术运算符(+，-，*，/，++，--，%)、比较运算符(>，<，<=，>=，==，!=)、逻辑运算符(||，&&，!)、条件运算(?:)、位移运算符(|，&，<<，>>，~，^)和字符串运算符(+)。

(2) 表达式

运算符和操作数的组合称为表达式，通常分为四类：赋值表达式、算术表达式、布尔表达式和字符串表达式。

(3) 语句

Javascript 程序是由若干语句组成的，语句是编写程序的指令。Javascript 提供了完整的基本编程语句，它们是：赋值语句、switch 选择语句、while 循环语句、for 循环语句、do while

循环语句、break 循环终止语句、continue 循环中断语句、with 语句、try...catch 语句、if-else 语句等。

(4) 函数

函数是命名的语句段，这个语句段可以被当作一个整体来引用和执行。使用函数要注意以下几点：

- 函数一般由关键字 function 定义。
- 函数名是调用函数时引用的名称，它对大小写是敏感的。
- 参数表示传递给函数使用或操作的值，它可以是常量，也可以是变量，也可以是函数。
- return 语句用于返回表达式的值。

(5) 对象

JavaScript 的一个重要功能就是面向对象的功能，通过基于对象的程序设计，可以用更直观、模块化和可重复使用的方式进行程序开发。

一组包含数据的属性和对属性中包含数据进行操作的方法，称为对象。比如要设定网页的背景颜色，所针对的对象就是 document，所用的属性名是 bgcolor，如 document.bgcolor="blue"，就是表示使背景的颜色为蓝色。

(6) 事件

用户与网页交互时产生的操作，称为事件。事件可以由用户引发，也可能是页面发生改变，甚至还有你看不见的事件引发。绝大部分事件都由用户的动作所引发，如：用户按鼠标的按钮，就产生 onClick 事件，若鼠标经过某项内容，就产生 onMouseOver 事件等等。在 Javascript 中，事件往往与事件处理程序配套使用。

(7) 变量

变量有它的类型，上例中 username 的类型为 string(字符串)，JavaScript 支持的常用类型还有：object(对象)、array(数组)、number(数)、boolean(布尔值)、null(一个空值)和 undefined(没有定义和赋值的变量)。

例如：var username = "yangli";

实际上 JavaScript 的变量是弱变量类型，赋值给它字符串，它就是 String 类型；赋给它数字，它就是整型；赋给它 true 和 false，它就是 boolean 型。

2.3.2 JavaScript 使用方式

JavaScript 加入网页有两种方法：

(1) 直接加入到网页中

这是最常用的方法，大部分含有 JavaScript 的网页都采用这种方法，如：

```
<script language=" javascript">
<!--
document.write("这是 JavaScript！ ");
//-Javascript 结束-->
</script>
```

在这个例子中，我们可看到一个新的标签<script>...</script>，而<script language="javascript">

用来告诉浏览器这是用 JavaScript 编写的程序，需要调动相应的解释程序进行解释(W3C 已经建议使用新的标准<script type="application/javascript">)。

HTML 的注释标签<!--和-->用来去掉浏览器所不能识别的 JavaScript 源代码，这对不支持 JavaScript 语言的浏览器来说是很有用的。

“//-Javascript 结束”中双斜杠表示 JavaScript 的注释部分。至于程序中所用到的 document.write()函数则表示将括号中的文字输出到窗口中去。另外一点需要注意的是，<script>…</script>的位置并不是固定的，可以包含在<head>...</head> 或<body>...</body>中的任何地方，甚至是<html>...</html>标签之外。

(2) 引用方式

如果已经存在一个 JavaScript 源文件(通常以.js 为扩展名)，则可以采用这种引用的方式，以提高程序代码的利用率。在网页中调用 JavaScript 源文件的基本格式如下：

```
<script src＝"url" type="text/javascript"></script>
```

其中的 url 就是 JavaScript 源文件的地址。同样的，这样的语句可以放在 HTML 文档头部或主体的任何部分。例如要实现前文“直接插入方式”中所举例子的效果，可以首先创建一个 JavaScript 源代码文件 script.js，其内容如下：

```
document.write("这是 Javascript！");
```

如果网页文件和 script.js 文件在同一目录下，那么在网页中可以这样调用程序：<script src="script.js" type="text/javascript"></script>。

2.3.3 JavaScript 代码实例

【例 2-20】在网页主体和标题上分别显示日期(date.html)。

```
<!DOCTYPE html>
<html>
<body>
<script language = "JavaScript">
<!--
var    today = new Date();
var    day    = today.getDate();       //获取日期
var    month = today.getMonth() + 1;   //获取月份
var    year = today.getFullYear();     //获取年份
var    week = today.getDay();           //获取星期几
var    vw = new Array("星期天","星期一","星期二","星期三","星期四","星期五","星期六");
document.write("今天是" +year+ "年" + month + "月" + day+"日"+vw[week]);
document.title="今天是" +year+ "年" + month + "月" + day+"日"+vw[week];
//->
</script>
</body>
</html>
```

该例通过日期类 Date 的对象来获取当前的年月日和星期几信息，对这些信息进行简单加工后，通过 document.write()将日期显示在网页主体部分(可根据需要显示在不同位置，如表格

的单元格中)，通过设置 document.title 将日期显示在标题栏上。程序执行结果如图 2-22 所示。

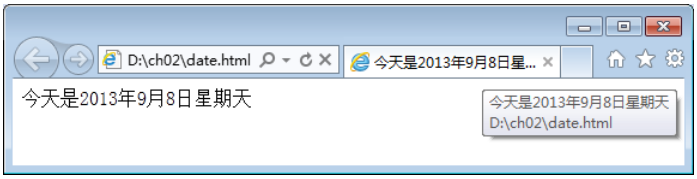

图 2-22　显示日期时间

【例 2-21】简单的表单验证(formcheck.html)。

```
<!DOCTYPE html>
<html>
<body>
<form    name="form1" action="#" onSubmit="return check()">
      用户名：<input type="text" name="user" /><br />
      邮&amp;nbsp;&amp;nbsp;件：<input type="text" name="email" /><br />
      <input type="submit" name="button" value="提交" / >
</form>
</body>
</html>

<script language="JavaScript">
<!--
function check()
{
      if (document.form1.user.value.length = 0)
      {
            alert("请输入用户名！ ");
            return false;
      }
      if (document.form1.email.value.indexOf("@") = -1 ||document.form1.email.value.length = 0)
      {
            alert("请输入一个正确的 email 地址");
            return false;
      }
      return true;
}
//-->
</script>
```

该例是一个简单的表单验证，<form name="form1" action="#" onSubmit="return check()">中的 onSubmit="return check()"表示在提交表单时调用函数 check()，而函数 check()在后面的 JavaScript 中定义，主要用于检验用户名是否为空，邮件是否为空以及邮件名中是否含有@，如果不正确，则使用 alert 弹出警示框。程序运行结果如图 2-23 所示。

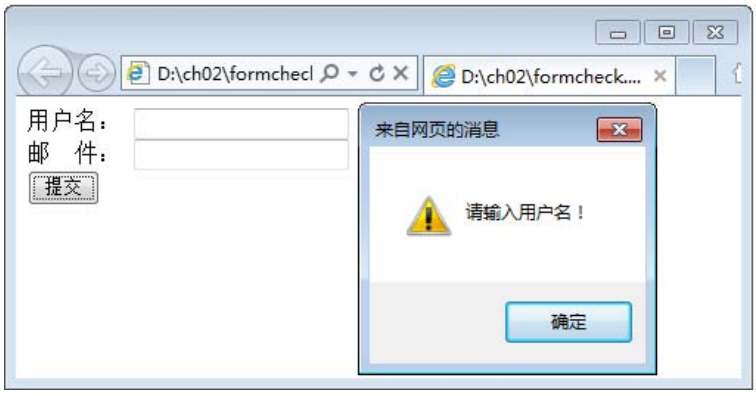

图 2-23　简单表单验证

【例 2-22】验证表单(formtest.html)。

```
<!DOCTYPE html>
<html>
<head>
<title>JavaScript 验证表单字段</title>
<link href="style.css" rel="stylesheet" type="text/css">
</head>
<body>
<form action="#" onSubmit="return check(this);">
<table border="1">
      <tr>
<td colspan="2">请填入个人信息：</td>
</tr>
<tr>
<td>员工号：</td>
<td>
     <input type="text"    name="id" validChar="\d{4}"    isRequired="true" />
     <span class="feedbackHide" id="idMsg">(员工号必须输入四位数字)</span>
     <br /><i>必填，四位数字</i></td>
     </tr>
<tr>
     <td>姓名：</td>
<td>
          <input type="text" name="name" validChar="[\u4E00-\u9FA5]{2,4}" isRequired="true" />
     <span class="feedbackHide" id="nameMsg">(姓名必须输入二到四位汉字)</span>
      <br /><i>必填，二到四位汉字</i></td>
 </tr>
 <tr>
 <td>邮件：</td>
 <td>
          <input type="text" name="mail" validChar="\w+([-+.]\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*"
        isRequired="true" />
      <span class="feedbackHide" id="mailMsg">(请输入正确格式的邮件)</span>
      <br /><i>必填</i>
 </td>
```

```
 </tr>
 <tr>
 <td>费用：</td>
 <td><input type="text" name="expense" validChar="\d+(\.\d{0,2})*" isRequired="false" />
      <span class="feedbackHide" id="expenseMsg">(请输入合法的费用格式，如 1.23)</span>
</td>
     </tr>
<tr>
<td> </td>
<td><input type="submit" value="提交" /></td>
</tr>
</table>
</form>
</body>
</html>

<script LANGUAGE="JavaScript">
<!--
//提交前检查
function check(vform){
    // 遍历表单中每个表单域
    for(var i=0;i<vform.elements.length;i++){
        // 如果表单域是文本框的话,进行定义好的验证
            if(vform.elements[i].type=="text"){
            // 取得验证结果
            var checkResult=checkTextBox(vform.elements[i]);
            // 取得文本框名,getAttribute 获取指定标签属性的值
            var name=vform.elements[i].getAttribute("name");
            // 验证通过
            if(checkResult){
                    //getElementById 通过通过控件 ID 取得元素的值
                document.getElementById(name+"Msg").className="feedbackHide";
            }
            else{
                // 验证不通过，显示提示文字并置焦点
                document.getElementById(name+"Msg").className="feedbackShow";
                vform.elements[i].focus();
                return false;
            }
        }
    }
    return true;
}

// 检查文本框
function checkTextBox(vTextBox){
    // 取得文本框中允许输入的合法文字正则表达式
    var validChar=vTextBox.getAttribute("validChar");
    // 取得文本框中是否必须检查的标志
    var isRequired=vTextBox.getAttribute("isRequired");
```

```
        // 取得文本框的输入
        var inputValue=vTextBox.value;
         // 如果是非必填字段且没有输入则返回真
        if(isRequired!="true" && inputValue.length<1){
              return true;
        }
        // 否则进行正则表达式验证
        else{
              var regexStr="^"+validChar+"$";
              var regex=new RegExp(regexStr);
              return regex.test(inputValue);
        }
    }
    //-->
    </script>
```

该例依然是表单验证，在表单每个<input>项中设置 validChar 属性来记录此文本框中允许输入的字符的正则表达式，设置 isRequired 属性来记录该项是否为必须的，这些都需要在 JavaScript 代码中判断，请大家认真阅读程序以及注释。例如表单项“员工号”<input type="text" name="id" validChar="\d{4}" isRequired="true" />中，validChar="\d{4}"表示员工号必须输入 4 位数字，isRequired="true"用于验证该项必须输入。更多常用正则表达式详如表 2-9 所示。网页浏览结果如图 2-24 所示。

表 2-9　常用正则表达式举例

正则表达式	说　明	正则表达式	说　明
[A-Za-z]	1 位字母	\d	1 位 0~9 的数字
[A-Za-z]{n}	n 位字母	\d{n}	n 位 0~9 的数字
[A-Za-z]{n,}	至少 n 位字母	\d{2,6}	2~6 位数字
[A-Za-z]{2,6}	2~6 位字母	\d{n,}	至少 n 位数字
[A-Za-z]+	1 串字母	\d+	1 串数字
[A-Za-z0-9]	1 位字母或数字	[\u4E00-\u9FA5]	1 位汉字
[A-Za-z0-9]{n}	n 位字母或数字	[\u4E00-\u9FA5]{n}	n 位汉字
[A-Za-z0-9]{n,}	至少 n 位字母或数字	[\u4E00-\u9FA5]{n,}	至少 n 位汉字
[A-Za-z0-9]{2,6}	2~6 位字母或数字	[\u4E00-\u9FA5]{2,6}	2~6 位汉字
[A-Za-z0-9]+	字母、数字组成的字符串	[\u4E00-\u9FA5]+	1 串汉字
-?\d+\.\d+	浮点数(正负均可)	\d+\.\d+	正浮点数
-?\d+\.\d{n}	保留 n 位小数的浮点数	-\d+\.\d+	负浮点数
-?\d+\.\d{n,}	保留至少 n 位小数的浮点数	-?\d+(\.\d{0,2})*	整数或者保留 0 到 2 位小数的浮点数
电子邮件	\w+([-+.]\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*		

备注：“+”元字符规定其前导字符必须在目标对象中连续出现1次或多次，“*”元字符规定其前导字符必须在目标对象中出现0次或连续多次，“?”元字符规定其前导对象必须在目标对象中连续出现0次或1次。

另外，该程序中，一旦表单项内容出错，则在其后出现红色提示文字，提示性红色文字是否显示通过 CSS 文件来设置，相应的文件保存为 Style.css，控制其他样式的代码也在此文件中，文件内容如下。

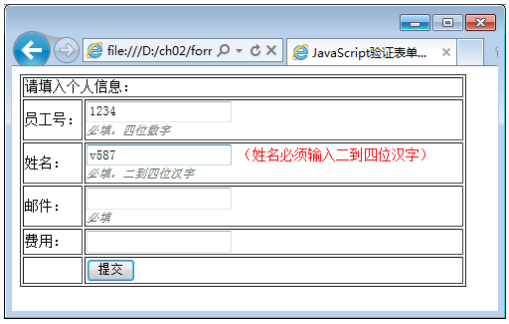

图 2-24　表单验证

```
.feedbackShow{
        visibility: visible;
        color:red;
}
.feedbackHide{
        visibility: hidden;
}
td{
        font-size:14px;
}
td i{
        color:#666666;
        font-size:12px;
}
```

2.4 小结

HTML 和 JavaScript 这两项技术是学习 JSP 的基础，CSS 用于辅助 HTML 设计网页样式，希望读者对其能够有所了解和掌握。其中 HTML 更是写好 JSP 程序最基础和不可或缺的技术，读者必须能够熟练地制作 HTML 静态网页，而不仅仅局限于本书介绍内容。JavaScript 作为客户端的脚本语言，在实际项目中的应用也比较多，不过读者没必要记住 JavaScript 语法的每个细节，能够做到基本读懂就行，在实际需要时可以有目标地在网上查找相关资料。

2.5 习题

2.5.1 选择题

1. 在网页中插入图像使用标签(　)。

A. <img>　　B. <image>　　C. <pic>　　D. <picture>

2. 表格标签是(　)。

A. <table>　　B. <td>　　C. <tr>　　D. <th>

3. 表单项<input>标签中表示插入提交按钮时，type 的属性值需设为(　)。

A. button　　B. submit　　C. reset　　D. btn

4. HTML5 中，<input>标签的属性 type 为(　)时，用户可以通过鼠标在调色板上自由地选择某种颜色。

A. color　　B. check　　C. select　　D. look

5. 关于 CSS，以下叙述错误的是(　)。

A. CSS 的中文意思是层叠样式表，简称样式表

B. CSS 可以精确地控制网页中元素的样式

C. 一个 HTML 网页文件只能应用一个 CSS 文件

D. CSS 文件可以单独保存而不必和 HTML 文件合并在一起

6. 下列选项中，应用了行内样式的是(　)。

A.　<p class="s1">　　B.　<p id="s1">

C.　<p style="color:green;">　　D.　<p class="color:green;">

2.5.2　判断题

1. HTML 的中文含义是超文本标记语言。　(　)

2. colspan 是<td>标签的属性，表示跨越的列数，例如“colspan=4”表示这一格的宽度为 4 个列的宽度。　(　)

3. 表单在网页中用来给访问者填写信息，从而能采集客户端信息，使页面具有交互信息的功能。　(　)

4. 超链接只能在不同的网页之间进行跳转。　(　)

5. 文本链接使用属性 a{text-decoration:none;}可以设置链接加上下划线。　(　)

2.5.3　填空题

1. 表格由 3 个基本组成部分，分别为行、列和_____。

2. CSS 中 ID 选择器在定义的前面要有指示符_____。

3. 段落标签是_____。

4. 当<input>标签的 type 属性值为_____时，代表在网页中加入一个日期表单项。

2.5.4　简答题

1. HTML 文档的基本结构是什么？

2. 完成如图 2-25 所示网页，使用表格布局，使用 CSS 控制样式，表格第一行的日期是使用 JavaScript 显示的系统当前日期，单击“查看谜底”时，弹出谜底。

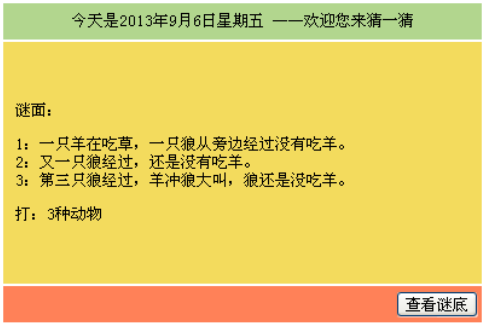

图 2-25　综合练习