

 案例素材和源代码、电子课件、习题参考答案

JSP

应用与开发技术

(第3版)

马建红 李学相 主 编
韩 颖 王瑞娟 张 晗 副主编



清华大学出版社

JSP 应用与开发技术

(第3版)

马建红 李学相 主 编
韩 颖 王瑞娟 张 晗 副主编

清华大学出版社

北 京

内 容 简 介

JSP(Java Server Pages)是由 Sun Microsystems 公司倡导的、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 章和第 2 章是第 1 部分,介绍了 JSP 与 Web 技术概论和开发环境的搭建;第 3~5 章是第 2 部分,详细讲解了 JSP 技术的基本语法;第 6 章和第 7 章是第 3 部分,是 JSP 应用开发的进阶,讲述了 JavaBean、表单处理及文件的操作;第 8 章和第 9 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 10 章和第 11 章是第 5 部分,详细讲述了 Servlet 技术;第 12 章和第 13 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 14~16 章是第 7 部分,以 3 个实例讲述了 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。

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 应用与开发技术 / 马建红, 李学相 主编. —3 版. —北京: 清华大学出版社, 2019
ISBN 978-7-302-51373-5

I. ①J… II. ①马… ②李… III. ①JAVA 语言—程序设计 IV. ①TP312.8

中国版本图书馆 CIP 数据核字(2018)第 232137 号

责任编辑: 王 定
封面设计: 孔祥峰
版式设计: 思创景点
责任校对: 成凤进
责任印制: 丛怀宇

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 装 者: 三河市金元印装有限公司

经 销: 全国新华书店

开 本: 185mm×260mm

印 张: 32.5

字 数: 811 千字

版 次: 2011 年 2 月第 1 版

2019 年 1 月第 3 版

印 次: 2019 年 1 月第 1 次印刷

印 数: 26501~30000

定 价: 78.00 元

产品编号: 079635-01



P R E F A C E

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互联网上的主流开发技术,已经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应用程序难以维护。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,除了基本语法介绍外,本书同时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3版)》仍保持了第2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、EL、标签库、MVC等技术。本书每一章节的实例读者可以直接使用,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2版相比,本书不仅采用了最新版本的开发工具,而且增加了最新Servlet版本中出现的新特性,调整了相关实例,竭尽所能展现本书的实用性。另外,对第2版的章节进行了细致的修订,对部分内容进行了重大改动。

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,包括第1章和第2章:第1章,JSP与Web技术概论;第2章,JSP的开发和运行环境。和第2版相比,增加了最新版的Eclipse Java EE、MyEclipse集成开发环境,以及其他JSP开发环境。新版本删除了第2版的第2章HTML和JavaScript技术的介绍。

第2部分介绍了JSP应用开发基础,包括第3~5章:第3章,JSP基本语法;第4章,JSP内置对象;第5章,Cookie及会话追踪。和第2版相比,增加了session会话与会话追踪相关内容。

第3部分介绍了JSP应用开发进阶,包括第6章和第7章:第6章,JavaBean和表单处理;第7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2版的不同之处是,增加了Apache Commons FileUpload文件的上传。

第4部分介绍了JSP数据库编程基础,包括第8章和第9章:第8章,应用JDBC进行数据库开发,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；第 9 章，JSP 与 JavaBean 应用实例。和第 2 版相比，本书删除了 MySQL 数据库安装配置等基本操作，并且更换了第 2 版第 11 章中的电商管理实例，改为一个简洁的商品管理系统，对 Model1 模式的应用进行详细的讲解。

第 5 部分讲述了 Servlet 技术，包括第 10 章和第 11 章：第 10 章，Servlet 基础；第 11 章，使用 Servlet 过滤器和监听器。新版本中增加了 Servlet 3.0 程序的步骤，监听器新增@WebListener 注解。

第 6 部分讲述了标签库的应用，包括第 12 和第 13 章：第 12 章，JSTL 标准标签库；第 13 章，自定义标签库。新版本中对每部分内容都进行了细致的修订。

第 7 部分即第 14~16 章，讲解了 Web 开发的具体应用。和第 2 版不同的是，新版中更新了部分实例，运用 MVC 技术，采用三层架构，并采用了自定义标签和 EL 等技术展示了三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。

本书由浅入深、循序渐进，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，可以帮助读者很快地进行实际开发。在本书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，并巩固所学到的知识。

本书由马建红、李学相任主编，韩颖、王瑞娟、张晗任副主编。参与编写的人员还有卫权岗、郭红艳、谷保平、何保锋、程凤娟、尹辉、赵玉娟等。此外，代瑾、张文亮、王若楠、任东冉、杨卢阳、张天壮、石刘帅、申清源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。

本书每章的案例源代码可通过扫描对应章节前的二维码下载，课件、实验参考答案及课后习题参考答案下载如下：



课件



实验参考答案



课后习题参考答案

编 者
2018 年 9 月



C O N T E N T S

第 1 章 JSP 与 Web 技术概论	1
1.1 Web 应用程序开发基础知识	2
1.1.1 HTTP 通信协议概述	2
1.1.2 传统的 Web 应用程序	3
1.1.3 Web 结构	3
1.2 静态网页与 HTML	5
1.3 交互式动态网页技术	6
1.3.1 动态网页的程序语言	6
1.3.2 在客户端执行的网页语言	6
1.3.3 在服务器端执行的网页语言	7
1.4 网站数据库技术	8
1.5 JSP 与相关技术	9
1.5.1 什么是 JSP	10
1.5.2 JSP 技术原理	11
1.5.3 JSP 本质与 Servlet	12
1.5.4 JSP 与 JavaBean	12
1.6 软件编程体系简介	13
1.6.1 C/S 结构编程体系	14
1.6.2 B/S 结构编程体系	14
1.7 企业应用开发架构	15
1.7.1 两层架构	15
1.7.2 三层架构	15
1.7.3 N 层架构	15
1.7.4 开发架构比较	15
1.7.5 J2EE 简介	16
1.8 JSP 知识体系及学习之路	18
1.8.1 JSP 知识体系	18
1.8.2 JSP 程序员学习路径	18
1.9 小结	20

1.10 习题	20
第 2 章 JSP 的开发和运行环境	22
2.1 JSP 的开发和应用平台介绍	23
2.1.1 Caucho 公司的 Resin 平台	23
2.1.2 Apache 公司的 Tomcat 平台	23
2.1.3 BEA 公司的 WebLogic 平台	24
2.1.4 IBM WebSphere Application Server 平台	24
2.2 Eclipse Java EE 集成开发环境	25
2.2.1 安装和配置 JDK	25
2.2.2 Tomcat 服务器	27
2.2.3 Eclipse Java EE 开发环境 搭建	32
2.3 MyEclipse 开发环境	38
2.3.1 MyEclipse 简介与下载	38
2.3.2 MyEclipse 安装与使用	39
2.4 其他 JSP 开发环境	45
2.4.1 IDEA 开发环境	45
2.4.2 Adobe Dreamweaver	46
2.4.3 UltraEdit 编辑器	47
2.5 小结	47
2.6 习题	48
第 3 章 JSP 基本语法	50
3.1 JSP 文件的结构	51
3.1.1 创建第一个 JSP 文件	51
3.1.2 分析 JSP 文件的组成元素	52
3.2 JSP 的脚本元素	53
3.2.1 隐藏注释	53

3.2.2	HTML 注释	53	4.5.5	其他 session 对象的常用方法	110
3.2.3	声明	55	4.5.6	session 对象应用实例	110
3.2.4	脚本代码	57	4.6	application 内置对象	116
3.2.5	表达式	58	4.6.1	存取 application 中的数据	116
3.3	JSP 指令元素	59	4.6.2	使用 application 对象取得信息	117
3.3.1	page 指令	59	4.6.3	application 对象应用实例	118
3.3.2	include 指令	64	4.7	其他 JSP 内置对象	122
3.3.3	taglib 指令	66	4.7.1	pageContext 对象	122
3.4	JSP 动作元素	66	4.7.2	config 对象	124
3.4.1	<jsp:include>	67	4.7.3	page 对象	127
3.4.2	<jsp:forward>	70	4.7.4	exception 对象	127
3.4.3	<jsp:param>	72	4.8	小结	128
3.4.4	<jsp:useBean>、<jsp:setProperty>和<jsp:getProperty>动作	73	4.9	习题	129
3.4.5	<jsp:plugin>	81	第 5 章	Cookie 及会话追踪	132
3.5	小结	83	5.1	Cookie 的概念和特性	133
3.6	习题	83	5.1.1	什么是 Cookie	133
第 4 章	JSP 内置对象	86	5.1.2	Cookie 的常见用途	134
4.1	JSP 内置对象概述	87	5.2	在 JSP 中使用 Cookie	135
4.2	request 对象	88	5.2.1	创建 Cookie	135
4.2.1	request 对象常用方法	88	5.2.2	读写 Cookie	136
4.2.2	request 对象应用实例	89	5.2.3	Cookie 中的主要方法	137
4.3	response 对象	95	5.2.4	操作 Cookie 的常用方法	137
4.3.1	response 对象常用方法	96	5.3	Cookie 对象的应用实例	139
4.3.2	response 对象应用实例	96	5.4	会话与会话追踪	146
4.4	out 对象	103	5.5	实现会话追踪的 4 种方式	147
4.4.1	out 对象方法成员与数据输出	104	5.5.1	Cookies 和 URL 重写	147
4.4.2	缓冲区操作	105	5.5.2	隐藏表单字段	148
4.4.3	out 对象应用实例	105	5.5.3	使用内建 session 对象	148
4.5	session 对象	107	5.6	内建 session 对象的生命周期	150
4.5.1	session 的概念	108	5.6.1	在 web.xml 中配置内建 session 对象的过期时间	150
4.5.2	session 对象的 ID	108	5.6.2	设置内建 session 对象的过期时间	150
4.5.3	session 的有效期限	109			
4.5.4	访问 session 中的数据	109			

5.7 利用 session 存取功能实现简单购物车	152	7.4 创建和删除目录	190
5.8 小结	158	7.5 文件的上传和下载	191
5.9 习题	159	7.6 使用 jspSmartUpload 上传包	195
第 6 章 JavaBean 和表单处理	162	7.7 使用 Commons FileUpload 上传包	197
6.1 非 MVC 模式(Model1)	163	7.8 小结	199
6.1.1 Model1 的特点	163	7.9 习题	199
6.1.2 Model1 的应用范围	164	第 8 章 应用 JDBC 进行数据库开发	202
6.2 MVC 编程模式(Model2)	164	8.1 JDBC 概述	203
6.2.1 什么是 MVC 模式	164	8.1.1 JDBC 的用途	203
6.2.2 MVC 模式在 Web 编程中的应用	165	8.1.2 JDBC 的典型用法	203
6.3 剖析 JavaBean	165	8.1.3 JDBC 的体系结构	204
6.3.1 什么是 JavaBean	166	8.1.4 驱动器类型	204
6.3.2 JavaBean 的特征	167	8.1.5 安装驱动器	206
6.3.3 创建一个 JavaBean	168	8.2 JDBC 连接数据库的方法	206
6.4 在 JSP 中使用 JavaBean	169	8.3 使用 JDBC 操作数据库	207
6.4.1 调用 JavaBean	170	8.3.1 使用 JDBC 访问数据库的过程	208
6.4.2 访问 JavaBean 属性	170	8.3.2 使用 Statement 执行 SQL 语句	210
6.4.3 设置 JavaBean 属性	170	8.3.3 PreparedStatement 接口	217
6.4.4 JavaBean 的生命周期	170	8.3.4 CallableStatement 对象	220
6.4.5 类型自动转换规则	174	8.3.5 使用 ResultSet 处理结果集	225
6.5 使用 JavaBean 处理表单数据	174	8.4 Java 与 SQL 的数据类型转换	229
6.5.1 JSP 处理与 form 相关的常用标签简单实例	175	8.5 使用 JDBC 连接不同的数据库	231
6.5.2 设置中文编码	181	8.5.1 连接 Oracle 数据库	231
6.6 小结	181	8.5.2 连接 DB2 数据库	231
6.7 习题	181	8.5.3 连接 SQL Server 数据库	231
第 7 章 JSP 中的文件操作	184	8.5.4 连接 Sybase 数据库	232
7.1 数据流和 File 类	185	8.5.5 连接 Access 数据库	232
7.1.1 数据流	185	8.6 连接池	232
7.1.2 File 类	185	8.6.1 连接池的实现原理	232
7.2 读写文本文件	187	8.6.2 在 Tomcat 上配置数据源与连接池	233
7.3 文件的浏览	189		

8.6.3 配置连接池时需要注意的问题.....	235	10.3.1 GenericServlet	293
8.7 存取二进制文件.....	236	10.3.2 HttpServlet.....	294
8.7.1 图像文件存取到数据库的过程.....	236	10.3.3 Servlet 实现相关实例.....	295
8.7.2 声音文件存取到数据库的过程.....	240	10.4 Servlet 请求和响应相关.....	298
8.7.3 视频文件存取到数据库的过程.....	244	10.4.1 HttpServletRequest 接口.....	298
8.7.4 保存图片文件路径到数据库的存取图片过程.....	248	10.4.2 HttpServletResponse 接口.....	300
8.8 实现分页显示.....	251	10.4.3 Servlet 请求和响应相关实例.....	301
8.8.1 分页显示技术的优劣比较.....	251	10.5 Servlet 配置相关.....	303
8.8.2 分页显示的 JavaBean 实现.....	252	10.5.1 ServletConfig 接口.....	303
8.9 小结.....	258	10.5.2 获取 Servlet 配置信息的例子.....	303
8.10 习题.....	258	10.6 Servlet 中的会话追踪.....	307
第 9 章 JSP 与 JavaBean 应用实例.....	261	10.6.1 HttpSession 接口.....	307
9.1 需求和设计.....	262	10.6.2 HttpSession 应用实例.....	309
9.1.1 功能介绍.....	262	10.7 Servlet 上下文.....	311
9.1.2 文件结构.....	262	10.7.1 ServletContext 接口.....	311
9.1.3 数据库设计.....	263	10.7.2 ServletContext 接口的应用实例.....	312
9.2 使用 JavaBean 封装数据库的访问.....	263	10.8 Servlet 协作.....	313
9.3 项目页面实现.....	265	10.8.1 RequestDispatcher.....	313
9.4 小结.....	282	10.8.2 forward()控制页面跳转.....	314
第 10 章 Servlet 基础.....	283	10.8.3 include()控制页面包含.....	315
10.1 Servlet 介绍.....	284	10.9 Servlet 异常相关.....	316
10.1.1 什么是 Servlet.....	284	10.9.1 声明式异常处理.....	316
10.1.2 Servlet 技术特点.....	285	10.9.2 程序式异常处理.....	319
10.1.3 JSP 与 Servlet 的关系.....	285	10.10 Servlet 3.0 注解.....	322
10.1.4 Servlet 的工作原理.....	286	10.10.1 开发 Servlet 3.0 程序所需要的环境.....	322
10.1.5 Servlet 常用接口和类.....	286	10.10.2 开发 Servlet 3.0 程序.....	322
10.2 开发部署一个简单的 Servlet.....	288	10.11 Servlet 应用实例.....	325
10.2.1 创建 Servlet 文件.....	289	10.12 小结.....	333
10.2.2 Servlet 的配置文件.....	291	10.13 习题.....	333
10.3 Servlet 实现相关的接口和类.....	292	第 11 章 使用 Servlet 过滤器和监听器.....	337
		11.1 过滤器在 Web 开发中的应用.....	338

11.1.1 过滤器概述.....	338	12.3 设置 JSTL 运行环境.....	375
11.1.2 Filter API.....	338	12.3.1 JSTL 的安装.....	375
11.1.3 Filter 接口.....	339	12.3.2 JSTL 应用示例.....	375
11.1.4 FilterConfig 接口.....	340	12.4 使用核心标签.....	376
11.1.5 FilterChain 接口.....	340	12.4.1 表达式操作.....	377
11.1.6 编写过滤器类.....	340	12.4.2 建立 URL.....	380
11.1.7 过滤器的部署.....	342	12.4.3 条件控制.....	383
11.1.8 对请求数据进行处理 的过滤器.....	345	12.4.4 迭代—运行循环.....	385
11.1.9 过滤器新增@WebFilter 注解.....	350	12.5 使用 JSTL 的数据库标签.....	387
11.2 Servlet 监听器.....	351	12.5.1 指定数据源.....	388
11.2.1 监听器接口.....	351	12.5.2 进行查询或更新操作.....	388
11.2.2 ServletRequestListener 接口.....	352	12.5.3 对返回的结果进行处理.....	391
11.2.3 ServletRequestAttributeListener 接口.....	353	12.5.4 其他 SQL 标签库的标签.....	392
11.2.4 ServletContextListener 接口.....	353	12.6 i18n 与国际化.....	393
11.2.5 ServletContextAttributeListener 接口.....	355	12.6.1 国际化设置标签.....	393
11.2.6 HttpSessionAttributeListener 接口.....	357	12.6.2 消息标记库.....	394
11.2.7 HttpSessionBindingListener 接口.....	359	12.6.3 数字、日期格式化.....	396
11.2.8 监听器新增@WebListener 注解.....	364	12.7 函数标签.....	399
11.3 小结.....	365	12.8 小结.....	399
11.4 习题.....	365	12.9 习题.....	400
第 12 章 JSTL 标准标签库.....	368	第 13 章 自定义标签库.....	403
12.1 EL 表达式语言.....	369	13.1 自定义标签体系介绍.....	404
12.1.1 EL 与 EL 隐含对象.....	369	13.1.1 标签的形式.....	404
12.1.2 在 EL 中访问 JSP 隐含对象 的 getXXX() 方法.....	373	13.1.2 标签类相关接口和类.....	405
12.1.3 用 EL 访问 JavaBean 中的 属性.....	373	13.1.3 标签库描述文件.....	409
12.2 JSTL 标签库简介.....	374	13.1.4 在 Web 部署描述符中引入 标签库文件.....	411
		13.1.5 在页面中使用标签.....	411
		13.1.6 标签在 Web 页面中的 作用.....	412
		13.2 传统标签的开发.....	412
		13.2.1 带属性标签的开发.....	412
		13.2.2 带 Body 标签的开发.....	415
		13.2.3 嵌套标签的开发.....	417
		13.2.4 迭代标签的开发.....	420
		13.3 Simple 标签的开发.....	423

13.3.1 SimpleTag 接口.....	423	15.5.4 用户管理.....	448
13.3.2 Simple 标签的开发示例.....	424	15.5.5 调查问卷管理.....	454
13.4 小结	425	15.5.6 问题管理.....	460
13.5 习题	425	15.6 小结.....	467
第 14 章 网上书店	427	第 16 章 Web 应用开发实践	468
14.1 快乐购书网介绍.....	428	16.1 系统介绍.....	469
14.2 系统需求 and 设计.....	428	16.2 系统需求分析.....	469
14.3 数据库表设计.....	428	16.3 系统功能结构.....	469
14.4 JSP 页面开发.....	429	16.4 系统功能描述.....	470
14.5 JavaBean 开发	431	16.4.1 游客用户浏览模块.....	470
14.5.1 使用 Java Bean 封装数据库的 访问	431	16.4.2 管理员登录模块.....	473
14.5.2 购物车 JavaBean.....	432	16.4.3 管理员管理模块.....	473
14.6 Servlet 开发	432	16.5 数据库设计.....	477
14.6.1 普通 Servlet 开发.....	433	16.5.1 数据库逻辑结构设计	477
14.6.2 Listener 开发.....	434	16.5.2 数据库表的设计	478
14.6.3 Filter 开发.....	435	16.5.3 数据库相关脚本	479
14.7 其他 bean 类——Util.....	435	16.6 系统实现.....	480
14.8 部署描述符.....	436	16.6.1 模块公用类.....	480
14.9 小结	438	16.6.2 JavaBean.....	481
14.10 习题	439	16.6.3 Servlet.....	484
第 15 章 调查问卷管理系统	440	16.6.4 自定义标签.....	485
15.1 系统介绍	441	16.6.5 前台界面的实现	490
15.2 系统需求分析.....	441	16.6.6 后台管理页面的实现	493
15.3 系统功能结构.....	442	16.7 小结.....	494
15.4 数据库设计.....	442	16.8 习题.....	494
15.4.1 数据库逻辑结构设计	442	附录 实验	496
15.4.2 数据库表的设计	443	实验一 JSP 应用开发基础(一).....	496
15.4.3 数据库相关脚本	444	实验二 JSP 应用开发基础(二).....	498
15.5 系统实现	445	实验三 JSP 应用开发进阶.....	502
15.5.1 JavaBean.....	445	实验四 JSP 数据库编程基础.....	504
15.5.2 Servlet.....	445	实验五 Servlet 技术实验.....	506
15.5.3 系统界面.....	447	实验六 Web 应用开发.....	508

第 1 章

JSP与Web技术概论

本章主要对 JSP 与 Web 技术进行概要介绍，并为读者提出了一些学习 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 Web 应用程序开发基础知识，还介绍了静态网页和动态网页技术，通过 JSP 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；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；最后介绍了软件编程体系和企业应用开发架构。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 技术有一个总体了解。



本章学习目标

- ◎ 掌握 Web 应用程序开发基础知识
- ◎ 了解静态网页与动态网页技术
- ◎ 了解网站数据库技术
- ◎ 掌握 JSP 的基本概念
- ◎ 掌握 JSP 的知识体系
- ◎ 了解软件编程体系
- ◎ 了解企业应用开发架构
- ◎ 了解 JSP 的学习之路

1.1

Web 应用程序开发基础知识

WWW(World Wide Web)即全球广域网,也称为万维网。它是一种基于超文本和 HTTP 的、全球性的、动态交互的、跨平台的分布式图形信息系统,是建立在 Internet 上的一种网络服务,为浏览者在 Internet 上查找和浏览信息提供了图形化的、易于访问的直观界面,其中的文档及超级链接将 Internet 上的信息节点组织成一个互为关联的网状结构。万维网并不等同互联网,万维网只是互联网提供的服务之一。

从表现形式上来看,基于 Web 的应用程序是许多网页的集合,这些网页可以与其他网页进行交互,也可以与 Web 服务器上的其他各种资源(如数据库或 Web 服务)进行交互。

为了规范这种 Web 页面之间的交互流程,保证客户端和服务端之间能很好地通信,在网络上必须采用一个规范性的协议(HTTP 通信协议)来定义各种 Web 应用上的服务细节。

1.1.1 HTTP 通信协议概述

HTTP(Hypertext Transfer Protocol)通信协议是目前在 Internet 上应用最广泛的通信协议之一。HTTP 通信协议允许客户端向服务器提出基于 HTTP 格式的“请求”(Request),而服务器解析请求和完成请求的处理后,将根据实际的处理结果向请求端传回基于 HTTP 的“回应”(Response)。根据 HTTP 通信协议,客户端和服务端端的交互主要由以下四个步骤组成。

- (1) 当客户端向 Web 服务器发出请求时,Web 服务器将会为该客户开启一个新的连接。
- (2) 通过这个连接,用户(服务请求端)可以将 HTTP 请求通过网络,传送给 Web 服务器。
- (3) 当 Web 服务器收到 HTTP 请求时,将根据请求内容进行相应的处理,并将处理结果包装成 HTTP 回应。
- (4) 服务器会将 HTTP 回应传送给用户。只要用户接收到 HTTP 回应,Web 服务器就会关闭同客户端的连接,从而结束本次通信。

图 1-1 说明了客户端与 Web 服务器通信的整个流程,从图中可以看到,HTTP 通信协议用来规范在客户端和服务端之间的数据传输格式,更重要的是,HTTP 通信协议定义了从客户端发送 HTTP 请求到服务器端返回 HTTP 响应的整个流程。

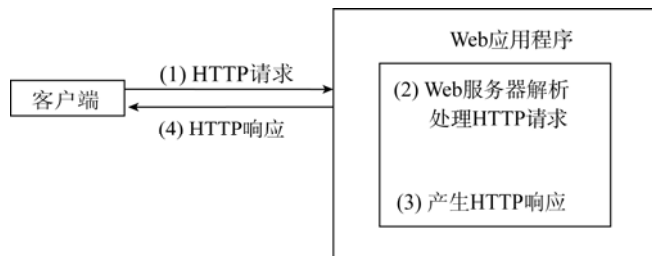


图 1-1 客户端(客户)与服务器端(Web 服务器)之间的交互关系

1.1.2 传统的 Web 应用程序

目前,大多数的 Web 应用程序都是基于 HTTP 通信协议的,即一个典型的基于 Web 的服务器和客户端应用程序必须具备“接受客户端的请求”和“将处理结果回应给客户端”这两种能力。

在 Web 开发的起步阶段,程序员是用 CGI(Common Gateway Interface, 公共网关接口)程序来编写 Web 应用程序组件的,即 Web 服务器之间及客户端和服务器之间是通过 CGI 来互相交换信息。当客户端向 Web 服务器送出一个 HTTP 请求时,Web 服务器将根据其中的 CGI 内嵌服务程序,执行如下动作:

- 如果客户端请求的内容是静态的 HTML 网页数据(即这些数据不是从数据库等数据源动态获取生成),Web 服务器将会根据事先编写好的处理程序自行处理,产生回应信息并返回。
- 如果涉及动态数据(如查询在服务器上的数据),则由内嵌的 CGI 程序负责处理,处理结束后,CGI 程序会先把结果回送给 Web 服务器,再由 Web 服务器传回用客户端。

整个流程如图 1-2 所示。

从实际的运行效果来看,虽然 CGI 程序能在一定程度上解决客户端与服务器端之间的交互问题,但是基于 CGI 程序本身的特点,这样的开发模式隐含了一些可能导致系统效率降低和项目可维护性变差的缺点。

- 从运行方式上来看,CGI 程序属于“操作系统进程”,因此对于每个 HTTP 的请求,服务器端都必须对应地开启一个 CGI 的服务,这加重了系统整体资源负担。
- CGI 程序通常不具有“平台独立性”,它甚至可以用多种语言来编写,如果转换到其他系统平台,程序可能要做适当的改写或重写。
- 最为重要的是,CGI 无法很好地形成一种通用的规范。事实上,不同类型的服务器 CGI 代码往往从请求风格到服务方式,都是不同的。

在实际的应用中,以上三个特点确实影响到 Web 应用程序的执行性能,因此需要一个能够取代 CGI 的 Web 开发技术,来满足“低负载、高重用和规范统一”等通用要求。

JSP 技术恰好可以完美地取代 CGI 程序,Java 的“平台无关性”可以保证用 JSP 编写出来的代码能够在多个平台之间很好地迁移。同时,JSP 语言是基于 Java 虚拟机的,不会对系统产生严重的资源负载。而且,不同平台上的 Java 语言及其虚拟机均采用了同一类标准(基于 Java 的标准),因此,JSP 可以满足“规范统一”这个要求。

1.1.3 Web 结构

Web 主要由两个部分组成,其结构如图 1-3 所示:提供 Web 信息服务的服务器端网站及向网站提出信息内容浏览要求的客户端浏览器。服务器端网站存放包含各种形态的多媒体信息网页,客户端主要包含各种可以浏览网页内容的浏览器软件,目前

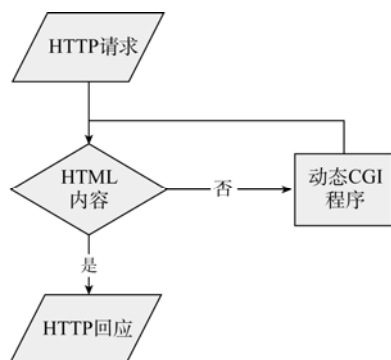


图 1-2 CGI 程序基本运行流程

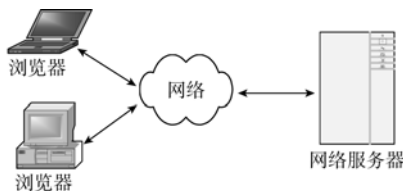


图 1-3 Web 结构

通用的浏览软件为 Windows 操作系统的 Internet Explorer(IE)浏览器。

了解 Web 结构的初步概念后,下面继续针对其中几个重要的技术内容进行讲解。

1. 客户端与浏览器

要打开和浏览网络上的网页文件,必须通过浏览器程序,如 IE、Netscape 等。而使用浏览器打开网页的这一端,称为客户端。因为网站为上线用户提供打开网页的服务,因此用户也可称为客户。

浏览器的功能主要是解释 HTML 文件中的内容,若 HTML 中同时含有客户端执行的描述语言,例如 VBScript 或 JavaScript,则浏览器同样会对其进行解释的操作,最后将整份网页的执行结果呈现在用户的浏览器窗口中。

2. 服务器端与服务器

与客户端相比,提供浏览网页服务的一方称作“服务器端”,而用来放置这些网页信息的计算机,则称为服务器。

服务器的功能并不只是单纯地存放网页信息,任何可提供网络服务的计算机都是服务器。例如,提供网页信息的称为网页服务器,而提供文件上传与下载功能的则称为文件服务器。

3. 通信协议

在网络上要能彼此互通信息就必须遵循一定的沟通方式,这些沟通方式即所谓的通信协议,表 1-1 所示为 WWW 中较为常用的通信协议。

表 1-1 WWW 常用通信协议

通 信 协 议	说 明
http	最常用的通信协议之一,主要用来传送文字、图片、声音等多媒体类型的数据,也是 Web 信息网传输网页所使用的通信协议
ftp	可用来上传文件至远程主机,或从指定的远程主机下载文件至本地计算机
mail	主要用来建立邮件发送服务
telnet	使用于远程登录,例如登录 BBS 系统
news	用于取得网络新闻论坛的协议

只有双方都使用相同的通信协议,才能建立起连接通道。

4. 全球资源定位器——URL

URL 的英文全称为 Uniform Resource Locator(全球资源定位器),是对可以从互联网上得到的资源的位置和访问方法的一种简洁的表示,是互联网上标准资源的地址。互联网上的每个文件都有一个唯一的 URL,它包含的信息指出文件的位置及浏览器应该怎么处理它。当用户想要打开位于远程网站主机上的网页时,必须指定其 URL 位置,也就是通常所讲的网址,它从左到右由下述部分组成。

- Internet 资源类型(scheme): 指出 WWW 客户程序用来操作的工具。如“http://”表示 WWW 服务器,“ftp://”表示 FTP 服务器,“gopher://”表示 Gopher 服务器,而“new:”表示 Newgroup 新闻组。
- 服务器地址(host): 指出 WWW 页所在的服务器域名。
- 端口(port): 有时需要,对某些资源的访问来说,需给出相应的服务器提供端口号。
- 路径(path): 指明服务器上某资源的位置(其格式与 DOS 系统中的格式一样,通常由“目

录/子目录/文件名”这样的结构组成)。与端口一样,路径并非总是需要的。

URL 地址格式排列为: scheme://host:port/path。例如, http://www.oracle.com/technetwork/java/index.html 就是一个典型的 URL 地址。当浏览器接收了用户输入的一个网址后,便会根据其中所提供的信息,向服务器提出网页浏览请求。

1.2 静态网页与 HTML

HTML 称为超文本标记语言,是一种用于开发静态网页的技术语言,主要提供创建网页文件所需的标准语法,以及描述网页数据的呈现方式,其中包含超级链接、图形或声音影像等多媒体内容。

静态网页主要由各种规范数据展现格式的标签组成,图 1-4 所示为显示“JSP 动态网页技术”的 HTML 网页文件 jsp.html 在浏览器上的显示结果。



图 1-4 静态网页在浏览器中的显示示例

使用记事本打开并查看这份网页的 HTML 如下:

```
<html>
<head>
<title>JSP</title>
</head>

<body>
  <b><font size="6"><i>JSP 动态网页技术</i></font></b>
</body>

</html>
```

可以看到,其内容由各种 HTML 标签组成,这些标签各有其特定的意义。例如, ... 表示其中包含的文字要以粗体的外观显示, <i>...</i> 则是将其中包含的文字设置为斜体格式。

由于 HTML 大量使用于网页的建构当中,因此编写 HTML 的相关软件也发展得相当成熟,网页设计人员几乎可以不用了解 HTML 标签即能设计出各种出色的网页。当然,学习 JSP 网页程序设计,了解 HTML 是不可缺少的必备知识。在本书后面的内容中,对于 HTML 与 JSP 的相关应用,将会有详细的介绍。

静态网页的 HTML 技术对于文件内容的展示,表现得相当称职。用户在浏览网页的时候,可以轻松查看网页设计者预先设计好的内容,网页一旦设计完成,内容就不会再变动,无论使用的网页浏览器、浏览用户的身份如何不同,他们所见到的内容都是相同的。

在 Web 发展初期,HTML 所建构的静态网页的确达到了信息传递的主要目的。然而随着 Web 网站各种应用服务的快速增长,静态网页单向呈现信息的特性,很快便无法满足实际的应用需求。尤其是越来越多的商业活动在网络上进行,网站与用户的互动需求开始浮现,如何创建出更加吸引用户的网站内容,这对静态网页的发展提出了新的要求。

一个购物网站根据到访的用户身份显示出不同的信息,以提供个性化的网页内容;一个门户网站也可能根据用户的习惯,将其喜好的信息内容显示在每次访问的首页。为了满足这些需

求, 具备动态特性的交互式网页开始出现, CGI、ASP 及 JSP 便是用于建构动态网页的相关技术, 在下一节的内容当中, 我们继续来介绍这一方面的相关概念。

1.3 交互式动态网页技术

由于静态 HTML 网页不能与网站用户进行互动, 因此出现了各种提供交互式功能的网页技术。本节将介绍交互式网页的概念, 同时说明交互式网页技术领域的主流技术 JSP 与其他动态网页技术。

1.3.1 动态网页的程序语言

HTML 网页只能提供各种静态的多媒体信息, 却不能与联机用户进行数据交换等动态交流。为了让网页根据用户不同的行为作出动态响应, 因此产生了内嵌于 HTML 网页的程序语言, 这一类程序语言依解释方式的不同, 可以区分为两类: 客户端 Script 语言和服务器端网页语言。下面分别对这两种网页语言进行说明。

1.3.2 在客户端执行的网页语言

在客户端执行的网页语言内嵌在 HTML 中, 而包含这类客户端执行程序的网页扩展名同样是 .htm。当浏览器向服务器请求打开网页时, 服务器会将整份网页传送至客户端, 由浏览器进行网页程序解释操作, 并且将结果显示在浏览器窗口中, 其过程如图 1-5 所示。

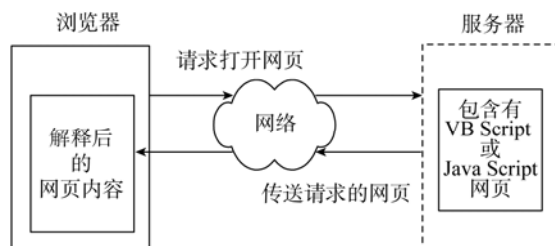


图 1-5 在客户端执行的网页语言

该过程说明如下:

- (1) 用户通过浏览器指定 URL, 向网页服务器请求特定的网页内容。
- (2) 服务器加载指定的 HTML 网页, 返回至客户端的网页浏览器进行解释。
- (3) 浏览器取得 HTML 文件, 解释其中的 HTML 及 Script 网页程序代码。
- (4) 显示 HTML 网页。

在客户端执行的网页语言可在网页中产生动态的效果, 如各类网页特效, 同时也能够在客户端与服务器端之间作数据交换时, 先行处理一些事前的准备操作。例如, 一般提供会员登录功能的网页, 通常都会利用客户端的 Script 提供输入数据的校验功能, 当用户输入不正常的数据(如不合法的身份证信息)时, 登录网页的操作将会失败, 相关的资料则无法返回至服务器端进行处理。

客户端的 Script 语言, 可以直接在浏览器这一端完成一些工作, 而不需要将所有的工作都返回至服务器端, 这样可以降低服务器的负担并提高执行的效率, 为一些大型网站减轻负担提供了很好的解决方案。

目前可以提供动态网页的 Script 语言有两种, 分别是 JavaScript 和 VBScript。其中, JavaScript

最初是由 Netscape 发展出来的一种 Script。用户需特别注意的是, JavaScript 是一种独立使用在前端网页的脚本语言, 与本书后面内容中所要探讨的结合 JSP 的 Java 语言完全没有关系, 只是后来 Netscape 与 Sun(Java 与 JSP 的创造者) 合作之后, 这种网页语言才被改名为 JavaScript, 尽管在语法上与 Java 类似, 但两者不能混为一谈。

JavaScript 内嵌于网页上, 与 HTML 相互混用创造动态的网页效果。目前, 普遍使用的浏览器均支持这一种网页语言, 它是一种很简单的程序语言, 可以很有效地完成一些在客户端进行的网页工作。该网页语言的另一种用途在于提供网页动画效果, 用户可以在许多网站看到由 JavaScript 创建的动态网页。

VBScript 是另一种被广泛使用的网页语言, 与 JavaScript 进行同样的客户端操作。这种语言以 Visual Basic 为基础发展而来, 只有微软本身的浏览器支持这种脚本语言。尽管目前用户所使用的浏览器几乎都是微软的 IE, 但是由于 JavaScript 已是客户端网页语言的通用标准, 同时也具备较强大的功能, 因此建议读者还是以 JavaScript 为主要的学习对象。

Script 虽然能够达到与用户互动的目的, 但是在功能上却有非常大的限制, 其中最大的缺陷在于其无法集成服务器上的资源, 如文件操作与数据库存取等。

具备 Script 的网页只能算是单纯的动态网页, 在客户端浏览器进行动态效果, 服务器一旦将网页送出, 就无法再与其沟通, 因此无法达到真正的互动行为。同时, 基于安全上的考虑, 用户也无法通过客户端 Script 进行各种服务器的操作, 于是服务器端网页语言便被开发出来以解决相关的问题。

1.3.3 在服务器端执行的网页语言

在服务器端执行的网页语言特点为必须由服务器中的解释器来做解释的操作, 最后再将解释后的结果以 HTML 的格式传送至客户端, 直接显示在浏览器中。服务器端网页语言的出现, 真正实现了让用户通过网络与网站进行沟通的目的, 整个服务器端网页的运行过程如下:

- (1) 用户通过浏览器指定 URL, 向网页服务器请求特定的网页内容。
- (2) 服务器加载指定的 JSP 网页, 通过解释器解读其中的 JSP 程序代码, 创建 HTML 网页, 然后返回至客户端的网页浏览器进行解释。
- (3) 浏览器取得 JSP 文件, 解释其中的 HTML 及 Script 网页程序代码。
- (4) 显示网页。

在上述过程中, 用户可以发现, 服务器端的网页语言与 HTML 静态网页最大的差异在于第(2)步, 这个部分通过执行 JSP 程序代码创建 HTML 网页, 因此在不同条件的情况下, 网站得以利用同一份 JSP 网页文件, 输出完全不同的网页内容, 甚至动态产生 Script 程序代码, 其步骤流程如图 1-6 所示。

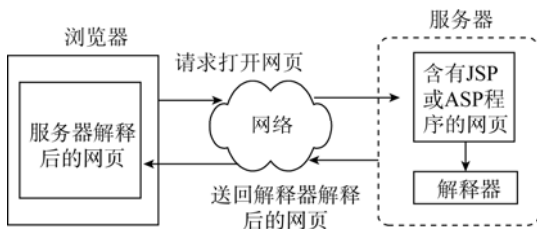


图 1-6 在服务器端执行的网页语言

可以用于产生交互式动态网页的技术有好几种, 比较著名的有 CGI、微软的 ASP、新版的 ASP.NET 和本书所要探讨的主题——JSP。其中, CGI 属于比较早期的服务器端动态技术, 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 和 JSP 等技术出现之后, 已逐渐淡出用

户的视线。

ASP 是微软开发的新一代服务器端网页语言, 其英文全称为 Active Server Pages(动态服务器网页), 因其简单易用的优点吸引了不少网站建构技术人员采用, 然而由于其功能过于单一, 只能使用简单的内置对象, 因而尽管结合 COM 技术得以延伸其结构格局, 但程序设计的复杂度也因此增加。

ASP.NET 是微软集成 .NET 平台发展而来的服务器端网页语言, 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, 不仅功能强大, 紧密结合 .NET 平台, 而且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。

JSP 是一种以 Java 为基础的服务器端网页语言, 本书对于这一技术将有详细的介绍。

无论何种服务器端的网页技术, 都有其学习的价值。而在几种主流技术之中, ASP.NET 无疑是 JSP 最为强大的竞争技术, 尤其是近几年 ASP.NET 因其集成微软 .NET 平台与强大的对象导向网页开发技术, 更将动态网页的发展推向了一个崭新的高度。然而 JSP 除了集成 Java 平台, 同样具备高性能与功能强大的特色之外, 其另外一项优势在于 Java 具有强大的跨平台能力。

网络本身是一种混杂各种技术平台的开放式环境, 无论何种网络平台, JSP 几乎都可以在其上不受制约地顺畅运行。虽然已有一些机构致力于将 ASP 移植到其他非 Windows 平台, 然而在短期内, ASP 还只能被限制在 Windows 的 .NET 平台上运行。同时, Java 语言的成功与强大的功能, 使得 JSP 能够提供比 ASP 更为广泛的网页功能。

1.4 网站数据库技术

JSP 网站开发离不开数据库编程, 与数据库交互几乎是每个大型动态网站必备的功能。通过访问数据库, 可以使一个动态网站所能呈现的内容范围比静态网站广泛得多。所以, 掌握数据库编程技术是网站开发的基础。

JSP 页面访问数据库操作基本流程是用户浏览器向 Web 服务器发出 HTTP 请求, 例如请求读取某个 JSP 页面, Web 服务器在接收到请求后解析该 JSP 文件, 如果其中含有数据库操作的代码, 则 Web 服务器会连接相应的数据库, 并执行需要的 SQL 语句, 如插入、修改或删除等。数据库操作完成后, 数据库服务器会将结果返回给 Web 服务器(如果需要的话), Web 服务器再根据这些结果生成相应的页面, 最后将这个页面及其他一些必要的文件, 如图片文件、CSS 文件等, 一起发送给用户端, 这一过程称为 HTTP 响应。要注意的是, 这里的 Web 服务器和数据库服务器在物理上可以是同一台服务器, 也可以是相距非常远的两台或多台服务器, 这要根据具体的应用来决定。图 1-7 所示是这一过程的形象描述。

一般情况下, 网站开发所需要了解的数据库技术也不是很多, 因为网站只是在运行过程中需要访问相应的数据库, 进行插入、修改及删除等基本操作, 对数据库更复杂的维护通常不是通过网页来完成的, 数据库本身有自己的维护方法和操作界面。

JSP 网页主要提供用户与服务器端数据库的操作存取功能, 同时也是数据库内容的展示界面。客户端浏览器、服务器 JSP 网页与数据库三者形成一个交互式的数据库网站, 其流程说明如下:

- (1) 用户通过浏览器指定 URL，向网页服务器请求特定的网页内容。
- (2) 服务器加载指定的 JSP 网页，通过解释器解读其中的 JSP 程序代码。
- (3) JSP 根据其运算逻辑，存取数据库内容，建立所需的内容信息。
- (4) 处理好的数据用以创建 HTML 网页，返回至客户端的网页浏览器进行解释。
- (5) 浏览器取得 JSP 文件，解释其中的 HTML 及 Script 网页程序代码。
- (6) 显示网页。

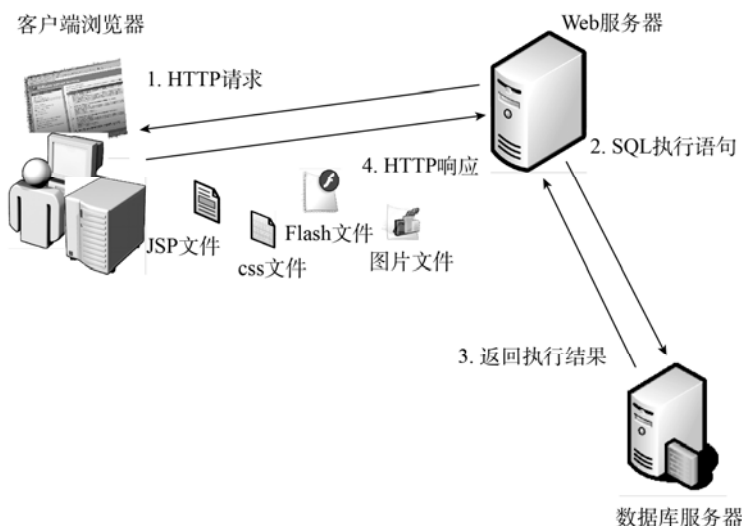


图 1-7 JSP 页面访问数据库的过程

当服务器里的网页包含存取数据库的程序代码时，服务器先完成数据库的存取工作，然后再交由解释引擎将 JSP 程序作解释操作，并将解释后的结果送回给浏览器。通过以上介绍，用户可以了解到，仅仅通过客户端技术并没有办法创建真正的交互式网页，只有使用服务器端执行的网页语言，并与数据库系统的结合，才能真正将请求付诸实现。

1.5 JSP 与相关技术

JSP 是由 Sun Microsystems 公司倡导的、许多公司参与建立的一种动态网页技术标准。JSP 技术用 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 Java Applet，甚至无 Frame。

1.5.1 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 Web 和 Java 功能的双重特性。

JSP 1.0 规范是 1999 年 9 月推出的,当年 12 月又推出了 1.1 规范。此后 JSP 又经历了几个版本,本书介绍的技术基于 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。程序运行效果如图 1-8 所示。

```
<%@ 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。

同 HTML 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 1-9 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。



图 1-8 HelloWorld.jsp 运行效果

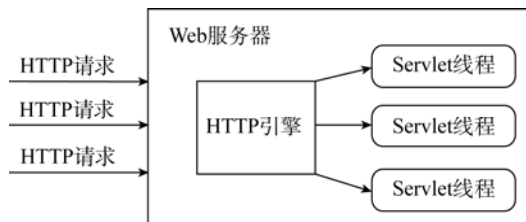


图 1-9 Web 服务器使用 Servlet 提供服务的示意图

- 编写简单。由于 JSP 是一项基于 Java 语言和 HTML 元素的技术,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。
- 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。

1.5.2 JSP 技术原理

JSP 文件的执行方式是“编译式”，并非“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(通常被译为 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 1-10 显示了整个 JSP 的运行流程。每次 Servlet 容器接收到一个 JSP 请求时，都会遵循以下步骤。

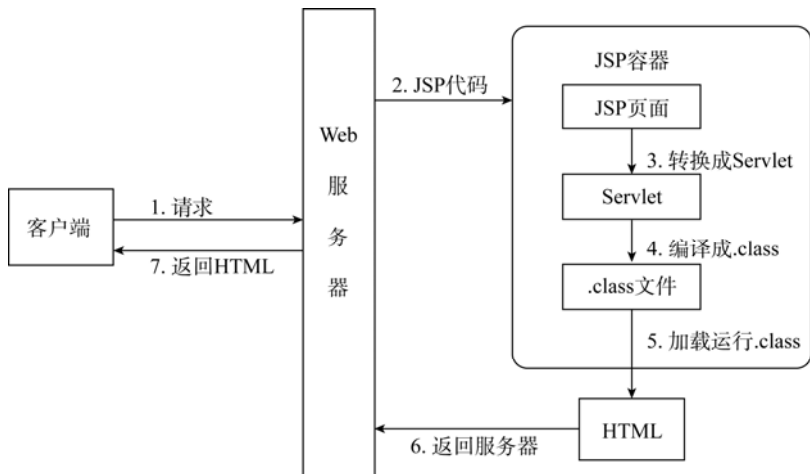


图 1-10 JSP 运行流程

(1) Web 容器接收 JSP 页面的访问请求时，它将把该访问请求交给 JSP 引擎去处理。每个 JSP 页面在第一次被访问时，JSP 引擎先将它翻译成一个 Servlet 源程序，接着再把这个 Servlet 源程序编译成 Servlet 的 class 类文件，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) Web 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。

(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 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, 又可以处理基于 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, 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。

1.5.3 JSP 本质与 Servlet

严格来说, JSP 是一种集成技术, 以 Java 平台为基础, 提供一种简便的方式, 让 Java 技术能够轻易运用于构建动态网页内容。

Java 为了产生与用户互动的动态网页内容, 开发出一种名为 Servlet 的应用技术。Servlet 是一种纯粹以 Java 语言编写, 符合标准规范的 Java 应用程序, 在网站服务器运行的时候被加载, 在客户端浏览器针对服务器提出内容要求时, 作出动态响应。

尽管 Servlet 应用程序产生动态网页的功能相当强大, 但是却不容易开发。JSP 就是为了克服这个障碍而发展出来的一种服务器端网页技术, 其本身是一个内容混杂 HTML 标签与 Java 语言的纯文本文件。当浏览器请求 JSP 网页时, 文本文件在网站服务器被编译成为 Servlet 加载, 然后产生动态的网页内容, 如图 1-11 所示。

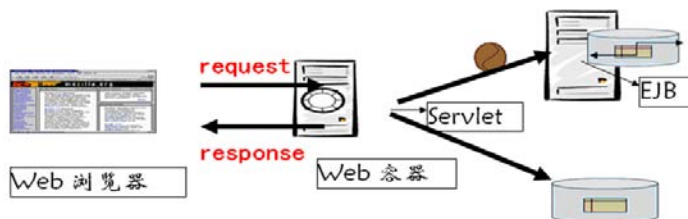


图 1-11 Servlet 执行过程

由于 JSP 本身被设计得非常容易学习与使用, 因此 Java 动态网页技术得以快速发展并且被广泛应用, 程序开发人员只需学习基础 Java 语法及相关的 JSP 对象, 便可以轻松完成动态网页的创建与设计。

1.5.4 JSP 与 JavaBean

让 JSP 如此出色的原因, 除了 Servlet 之外, 另外一点便是其运用 JavaBean 的能力。JavaBean 是一种组件技术, 为 Java 应用程序提供一种极具灵活性的设计模型, 它将程序依功能分开, 包装成为一种称为 JavaBean 的程序组件, 让其可以被外部 JSP 网页重复使用以提供所需的功能, 同时还可以扩充 JSP 网页的功能, 维持简洁易于维护的 JSP 网页程序代码。

使用 JSP 设计动态网页并不困难, 然而当网站的复杂度开始增加时, 用户在必须面对如何

有效管理与切割程序内容等，以及在提供强大功能的同时，保有网页程序设计的灵活性与容易维护的程序代码等。

如图 1-12 所示，JavaBean 用来包装网页程序逻辑及其相关功能，并且通过网页作引用，而网页本身则负责数据收集和-content 展示。

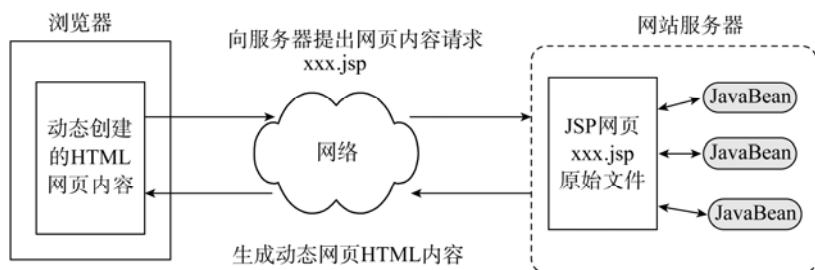


图 1-12 JavaBean 流程

JavaBean 本身是一种包含特定功能的 Java 程序组件，当一个 JSP 网页需要此项功能时，直接引用相关的 JavaBean 组件即可；而 JavaBean 最重要的概念在于将程序逻辑与创建网页界面的程序代码分开，网页只负责数据的收集及页面元素放置的处理，而 JavaBean 组件则是包含重要的程序逻辑运算。如此一来，JSP 网页将更容易维护与建构，JavaBean 同时也是建构商业级 JSP 网页程序最为重要的核心。

1.6 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 1-13 所示。

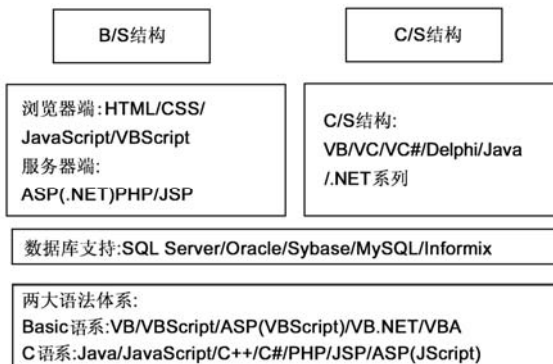


图 1-13 应用程序开发体系

开发基于 C/S 结构项目，其传统的开发环境有 VB、VC 及 Delphi 等，随着 Java 体系及 .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

(Personal Home Page)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 体系及 ASP.NET+SQL Server 体系。

软件开发涉及的语言虽然很多,但学习起来是有规律可循的。图 1-13 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。

1.6.1 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 结构的应用大概占到 40%的份额。C/S 结构应用程序的最大特点是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;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,比较烦琐,而且不能很好地跨操作系统平台。

C/S 结构通常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,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。随着时间的发展及版本的更新,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 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。

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。

1.6.2 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,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,在这三种语言中是使用最为广泛的一种。JSP 是 Sun 公司推出的 J2EE (Java 2 Enterprise Edition, Java2 企业版) 核心技术中重要的一种。PHP 在 1999 年下半年和 2000 年用得非常广泛,因为 Linux+PHP+MySQL (一种中小型数据库管理系统) 构成了全免费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。

数据库支持是必需的,目前应用领域的数据库系统全部采用关系型数据库 (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。

1.7 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，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。

1.7.1 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 1-14 所示。

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。

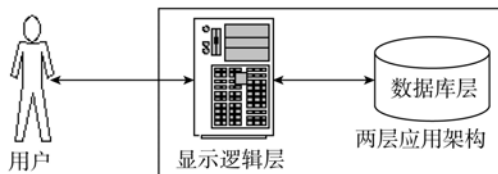


图 1-14 两层应用架构

1.7.2 三层架构

在两层应用架构中，应用程序直接同数据库进行对话。三层架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，通常这个层叫作“商务逻辑层”，如图 1-15 所示。

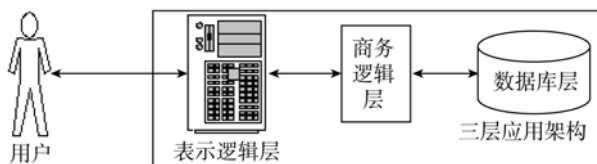


图 1-15 三层应用架构

1.7.3 N 层架构

如果某个应用超过 3 个独立的代码层，那么这个应用叫作 N 层应用，而不再叫四层或者五层等名称，而是统称为 N 层，如图 1-16 所示。

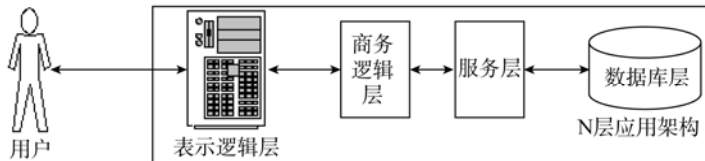


图 1-16 N 层应用架构

1.7.4 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，而且执行效率也有所提高，但是相对部署起来比较困难。

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。

1.7.5 J2EE 简介

目前，Java2 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 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 公司推出 J2EE 的目的是克服传统 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。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。由于它们都遵循了 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。目前在市场上可以看到很多实现了 J2EE 的产品，如 BEA WebLogic、IBM WebSphere 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成组件，各个应用组件根据它们所在的层分布在不同的机器上。事实上，Sun 公司设计 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，第一次部署的时候比较容易，但难以升级或改进，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)，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不同的服务提供一个独立的层，图 1-17 所示是 J2EE 典型的四层结构。J2EE 的四层结构的分别是：

- 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 EIS(Enterprise Information System, 企业信息系统)层组件。

下面分别对图 1-17 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 J2EE 组件: 应用客户端程序和 Applet 是客户层组件; Java Servlet 和 JavaServer Pages(JSP)是 Web 层组件; Enterprise Java Beans(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 Servlet。按照 J2EE 规范, 静态的 HTML 页面和 Applet 不算是 Web 层组件。如图 1-18 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。

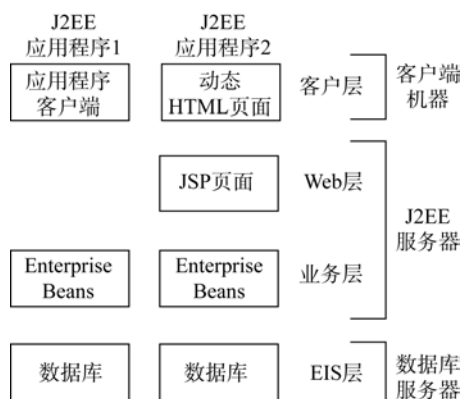


图 1-17 J2EE 四层模型

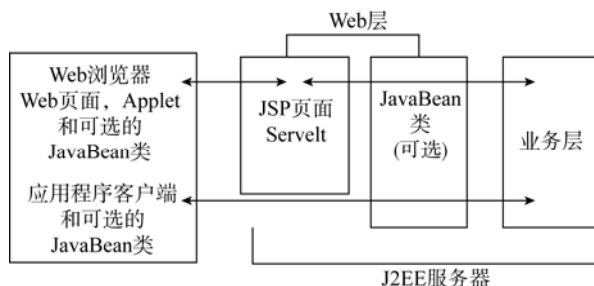


图 1-18 Web 层

(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 1-19 所示表明了一个 Enterprise Bean 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 层存储, 这个过程也可以逆向进行。

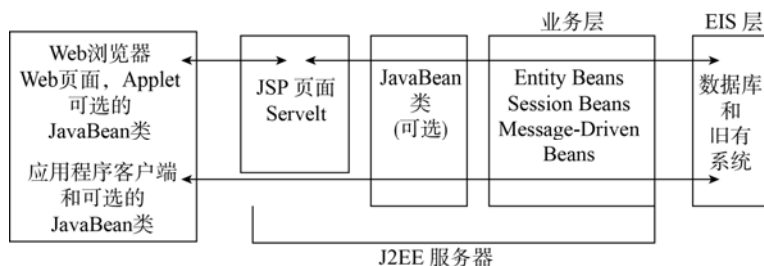


图 1-19 J2EE 服务层

有三种企业级的 Bean: 会话(Session)Bean, 实体(Entity)Bean 和消息驱动(Message-Driven)Bean。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 表示数据库表中一行永久的记录。当客户端程序中止或服务关闭时, 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) 企业信息系统层。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、数据库系统以及其他的遗留信息系统。例如, 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。

1.8 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。

1.8.1 JSP 知识体系

JSP 的知识体系如图 1-20 所示。

Java 和 HTML 是 JSP 学习中非常重要的基础, 如果仅仅懂得 JSP 的一些语法但对 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。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, 意味着 JSP 的学习之路要轻松很多。如果没有这些基础, 就需要花一些时间来学习这些基础知识。

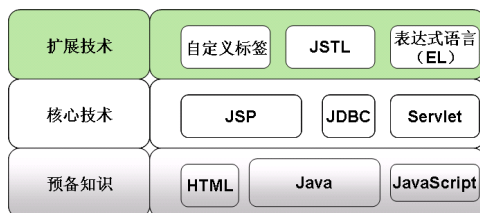


图 1-20 JSP 知识体系图

1.8.2 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, 这意味着学习 JSP 将会比较简单。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，特别是在 HTML 布局中 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会产生混乱的 HTML 语法，所以花时间学习手工写作 HTML 是很有必要的。因为我们经常会使用 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。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 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，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，因此建议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，在遇到问题时，将容易获得帮助。

(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 大多数应用将使用数据库，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 的细节很重要。

(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 的应用很多，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。简单说就是别人搭好舞台，你来表演。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)和高层业务逻辑之间。

此时，你已经成为了熟练的 JSP 程序员。但仍然有很多知识需要学习，如 DHTML、XML、Java 证书、JSP Tag Libraries 或表达式语言，这根据想要建造什么类型的网站来决定。

这些训练是 JSP 的核心。读者不必学完上面所有的内容，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些。

1.9 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。本章介绍了 Web 应用程序开发基础知识,动态网站开发技术的现状,JSP、ASP 和 PHP 等几种主要动态网页技术,以及软件编程系统,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。

1.10 习题

一、选择题

- 通过 Internet 发送请求消息和响应消息使用()网络协议。
A. FTP B. TCP/IP C. HTTP D. DNS
- Web 应用程序使用的三层体系结构包括()。
A. 表示层、逻辑层和业务层 B. 表示层、逻辑层和数据层
C. 逻辑层、业务层和数据层 D. 表示层、业务层和数据层
- 以下关于 HTML 说法正确的是()。
A. HTML 是一种 Web 客户和 Web 服务器之间的通信协议
B. HTML 是一种标记语言
C. HTML 文件能被 Windows 的文本编辑器解析
D. 浏览器不可以解析 HTML
- 以下()不是 HTML 的标记?
A. <html> B. <body> C. <% %> D.

- 与 JSP 同类型的技术有()。
A. JavaScript B. Java C. ASP.NET D. C#
- 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
- 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

8. JSP 页面经过转译之后, 将创建一个()文件。
A. applet B. servlet C. application D. server
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

二、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 是解释型语言。 ()
5. JSP 是以 Servlet 程序方式运行的, 而 ASP 是由 ASP 引擎解释执行的。 ()

三、填空题

1. 组成 Web 结构的两个主要部分, 分别是_____及_____。
2. 试举例目前网络上通用的三种通信协议名称: _____、_____、_____。
3. 目前在客户端运行的两种主要动态 Script 语言分别是_____、_____。
4. JSP 的发展, 主要在于提供一种简化服务器端程序_____的开发技术。
5. JSP 网页程序, 通过_____的引用, 以组件化的方式进行建构。
6. JSP 页面由_____和嵌入其中的_____所组成。
7. MVC 设计模式将应用程序分为模型、_____和_____。
8. JSP 的实质就是_____。

四、问答题

1. 说明客户端与服务器端在网络结构中所扮演的角色。
2. 解释超链接的定义。
3. 解释 URL 与通信协议的差异。
4. 解释 HTML 的定义。
5. 简述客户端与服务器端网页语言的不同。
6. 描述 HTML 网页、JavaScript 网页及 JSP 网页运作上的差异。
7. 说明 Java 相关技术——Servlet、JavaBean 及 JSP 之间的关联与运作。
8. 什么是 B/S 模式? 什么是 C/S 模式? 试举例说明。

第 2 章

JSP的开发和运行环境

开发基于 JSP 的动态网站，首先需要建立 JSP 的开发和运行环境。MyEclipse 是在 Eclipse 基础上加上自己的插件开发而成的功能强大的企业级集成开发环境，主要用于 Java、Java EE 及移动应用的开发。MyEclipse 的功能非常强大，支持也十分广泛，尤其是对各种开源产品的支持相当有力。Java EE 是 Enterprise Edition 的企业版本。这些工具系统是理想的完整 Java 开发 IDE，可以用来开发 HTML、JSP 等 web 程序。目前流行的 JSP 运行服务器有 Apache Tomcat、WebLogic 和 Resin 等，由于 Tomcat 是常见的开源免费软件，因此本书的案例全部使用 Tomcat 来测试运行。



本章学习目标

- ◎ 了解 JSP 的开发和应用平台
- ◎ 掌握 Eclipse Java EE 开发环境的搭建
- ◎ 掌握 MyEclipse 的开发环境
- ◎ 了解 IDEA 等 JSP 开发环境