

高等院校计算机应用技术系列教材

程志艳
张 亮 编著
马建红

JSB 实用简明教程

- ◆ 赠送教师完整的电子教案
- ◆ 提供本书程序代码下载
- ◆ 提供课后习题参考答案



清华大学出版社

高等院校计算机应用技术系列教材

JSP 实用简明教程

程志艳 张 亮 马建红 编著

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、许多公司共同参与建立的一种动态网页技术标准,它可以无缝地运行在 Unix, Linux, Windows 等操作平台上,从而建立安全的、跨平台的动态网站。

本书详细讲解了 JSP 语法和基本的 Web 程序设计方法。共分 11 章,内容包括 JSP 概述、Java 语言基础、JSP 应用开发体系和环境配置、JSP 动态网页基础、JSP 内置对象、JSP 中的文件操作、在 JSP 中访问数据库、JSP 与 Servlet 编程、JSP 与 JavaBean 编程等重要内容,最后一章通过一个网络在线书店系统介绍了使用 JSP 开发 Web 应用系统的详细过程,通过这个实例将前面介绍的基础知识应用到具体的程序开发中。

本书语言通俗易懂,结构清晰,突出了 JSP 在开发 Web 动态网站方面的强大功能及在开发商务网站方面的应用,以使读者快速掌握和运用 JSP 的编程技巧,适于初学者和有一定 JSP 基础的读者使用。

读者在阅读本书时,可以通过访问 <http://www.tupwk.com.cn/downpage> 下载本书程序代码或反馈相应意见。

版权所有,翻印必究。举报电话:010-62782989 13501256678 13801310933

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。

本书防伪标签采用特殊防伪技术,用户可通过在图案表面涂抹清水,图案消失,水干后图案复现;或将表面膜揭下,放在白纸上用彩笔涂抹,图案在白纸上再现的方法识别真伪。

图书在版编目(CIP)数据

JSP 实用简明教程/程志艳,张亮,马建红 编著. —北京:清华大学出版社,2005.12

(高等院校计算机应用技术系列教材)

ISBN 7-302-12054-4

I. J… II. ①程… ②张… ③马… III. JAVA 语言—主页制作—程序设计—高等院校—教材
IV. TP393.092

中国版本图书馆 CIP 数据核字(2005)第 126048 号

出 版 者: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 客户服务:010-62776969

组稿编辑:王 定

文稿编辑:鲍 芳

封面设计:康 博

版式设计:康 博

印 装 者:三河市春园印刷有限公司

发 行 者:新华书店总店北京发行所

开 本:185×260 印张:25.25 字数:583 千字

版 次:2005 年 12 月第 1 版 2005 年 12 月第 1 次印刷

书 号:ISBN 7-302-12054-4/TP·7803

印 数:1~5000

定 价:35.00 元

前 言

JSP(Java Server Pages)是由 Sun 公司倡导的、许多公司共同参与建立的一种动态网页技术标准,其设计目的使构造基于 Web 的应用程序更加容易和快捷,而这些应用程序又能够与各种 Web 服务器、应用服务器、浏览器和开发工具共同工作。目前,它已经成为开发动态网页的主流技术之一。

全书共分 11 章,从 Java 语言基础开始,系统、全面地介绍了使用 JSP 开发 Web 程序的方法和技巧,具体内容安排如下。

第 1 章介绍了 Java 技术基础、JSP 技术、使用 JSP 开发 Web 站点的几种重要方式和 JSP 的两种体系结构,使读者对 JSP 有一个整体的了解。

第 2 章主要介绍了 Java 语言的一些基础知识,包括 Java 的数据类型、运算符、条件语句、循环语句、强制类型转换、Java 类的技术基础、Java 高级类技术、多线程技术等,为后面 JSP 的学习提供了基础知识。

第 3 章主要介绍了 JSP 应用开发体系与环境配置,包括 Web 应用程序理论基础、JSP 的实现原理和运行机制、JDK+Tomcat 开发环境的配置和 JBuilder 2005 的开发环境,使读者可以自己搭建 JSP 的开发环境。

第 4 章主要介绍了 JSP 动态网页设计基础,包括 HTML 语法、CSS 语法、JSP 页面的基本结构、JSP 的注释、JSP 变量和方法的声明、JSP 的指令、JSP 的动作和 JSP 的脚本,为以后开发 Web 应用程序打下基础。

第 5 章主要介绍了 JSP 内置对象中的 Request 对象、Response 对象和 Session 对象,并通过多个实例介绍了它们的具体应用,使读者更加深刻地理解这些内置对象。

第 6 章主要介绍了 JSP 内置对象中的 Out 对象、PageContext 对象、Config 对象和 Exception 对象,并通过多个实例介绍了它们的具体应用,使读者更加深刻地理解这些内置对象。

第 7 章主要介绍了 JSP 中的文件操作,包括文件操作相关异常处理、JSP 的输入和输出流、File 类、在页面中读写文件中的数据等,并通过几个文件操作实例讲解了文件操作的应用。

第 8 章主要介绍了 JSP 访问数据库的知识,包括 JDBC 概述、使用 JDBC-ODBC 桥访问数据库、使用 JDBC 驱动程序访问数据库、对数据库的各种操作(查询、更新及添加记录等)、分页显示记录和查询 Excel 电子表格,最后给出了一个数据库访问实例,将本章前面所讲的基础知识综合运用到一起。

第 9 章主要介绍了 Servlet 的相关知识,包括 Servlet 的一些基本概念、主要方法、生命周期、类和接口等,并通过几个实例讲解了 Servlet 编程,使读者能更深刻地理解 Servlet 的应用。

第 10 章主要介绍了 **JavaBean** 的基本概念、属性、作用域、通过 **JavaBean** 实现数据库的访问等，并通过几个实例讲解了 **JavaBean** 的使用方法，使读者能更深刻地理解 **JavaBean** 的应用。

第 11 章通过一个网络书店综合实例，详细介绍了使用 **JSP** 开发 **Web** 应用程序的步骤和过程。

本书内容翔实、结构紧凑、条理清晰、知识点全面、语言通俗易懂，并配有很多实例和插图。每一章最后都有小节和习题，使读者对每一章所讲述的内容都有深刻地理解并加以巩固，十分适合初学者和有一定 **JSP** 基础的人员使用。

本书由程志艳、张亮、马建红编著，此外，武思宇、牟博超、李彬、付鹏程、高翔、朱丽云、崔凌、张巧玲、李辉、李欣、柏宇、郭强、金春范、程梅、黄霆、钟华、高海峰、王建胜、张浩和刘湘等同志在整理材料方面给予了作者很大的帮助。在此，作者对他们表示衷心地感谢。

由于时间仓促，加之水平所限，书中不足之处在所难免，敬请读者批评指正。

作 者

2005 年 11 月

目 录

第 1 章 概述	1	2.2.6 条件运算符	37
1.1 Java 技术简介	1	2.2.7 运算符优先级	37
1.1.1 JavaBean	1	2.3 条件语句	38
1.1.2 JDBC	2	2.4 循环语句	39
1.1.3 J2EE	3	2.4.1 while 语句	40
1.1.4 EJB	5	2.4.2 do...while 语句	40
1.1.5 Java Servlet	8	2.4.3 for 循环	40
1.2 JSP 技术	10	2.4.4 循环退出语句	41
1.2.1 JSP 技术概述	10	2.5 分支语句	43
1.2.2 JSP 技术的优势和不足	11	2.6 Java 中的强制类型转换	44
1.2.3 Java Server Pages 白皮书	12	2.6.1 简单数据类型的强制转换	44
1.3 用 JSP 开发 Web 站点的几种 主要方式	18	2.6.2 简单数据类型和对象之间 的强制转换	44
1.4 理解 JSP 的两种体系结构	19	2.7 Java 类技术基础	45
1.4.1 Model 1	20	2.7.1 类的声明和类体	45
1.4.2 Model 2	20	2.7.2 成员的声明	46
1.5 小结	21	2.7.3 方法调用	47
1.6 习题	22	2.7.4 构造函数	48
1.6.1 填空题	22	2.7.5 重载	49
1.6.2 选择题	22	2.7.6 继承	50
1.6.3 简答题	22	2.8 Java 高级类技术	51
第 2 章 Java 语言基础	23	2.8.1 抽象类	52
2.1 Java 的数据类型	23	2.8.2 匿名类	53
2.1.1 简单类型	23	2.8.3 包和接口	53
2.1.2 数组	27	2.9 多线程技术	58
2.2 运算符	30	2.9.1 多线程的定义	58
2.2.1 算术运算符	30	2.9.2 线程优先级	58
2.2.2 整数位运算符	31	2.9.3 线程同步	59
2.2.3 关系运算符	34	2.10 小结	59
2.2.4 布尔逻辑运算符	35	2.11 习题	60
2.2.5 赋值运算符	36	2.11.1 填空题	60
		2.11.2 选择题	60
		2.11.3 上机题	60

第3章 JSP 应用开发体系与环境配置	61	4.4.2 HTML 注释	127
3.1 Web 应用程序理论基础	61	4.4.3 Java 注释	128
3.1.1 Internet 技术概述	61	4.5 变量、方法的声明	129
3.1.2 HTTP 工作原理	64	4.5.1 变量、方法的声明和 表达式	129
3.2 JSP 实现原理和运行机制	65	4.5.2 表达式	130
3.2.1 JSP 实现原理	65	4.6 JSP 指令	131
3.2.2 JSP 运行机制	67	4.6.1 include 指令	131
3.3 JDK 的安装	70	4.6.2 Page 指令	133
3.3.1 安装 JDK	70	4.6.3 taglib 指令	135
3.3.2 设置 JDK 环境变量	73	4.7 JSP 动作(Action)语法	136
3.4 安装和配置 Tomcat 服务器	76	4.7.1 <jsp:include>动作	136
3.5 JBuilder 开发环境	77	4.7.2 <jsp:useBean>动作	137
3.5.1 使用 JBuilder 2005 创建 JSP 程序	77	4.7.3 <jsp:setProperty>动作	139
3.5.2 JSP 程序的编译和运行	85	4.7.4 <jsp:getProperty>动作	140
3.6 小结	86	4.7.5 <jsp:forward>动作	141
3.7 习题	86	4.7.6 <jsp:plugin>	142
3.7.1 填空题	86	4.8 JSP 脚本	144
3.7.2 选择题	87	4.9 小结	145
3.7.3 上机题	87	4.10 习题	145
第4章 JSP 动态网页设计基础	88	4.10.1 填空题	145
4.1 HTML 语法	88	4.10.2 选择题	146
4.1.1 什么是 HTML 语言	88	4.10.3 上机题	146
4.1.2 HTML 的基本标签	88	第5章 JSP 主要内置对象	147
4.1.3 创建列表	94	5.1 Request 对象	147
4.1.4 图像、多媒体和超链接	97	5.1.1 Request 对象方法	147
4.1.5 表格	100	5.1.2 Request 对象程序应用 实例	149
4.1.6 框架	103	5.2 Response 对象	159
4.1.7 表单	107	5.2.1 Response 对象方法	159
4.2 CSS 语法	113	5.2.2 Response 对象程序 应用实例	161
4.2.1 CSS 简介	113	5.3 Session 对象	164
4.2.2 CSS 基本语法	115	5.3.1 Session 对象的方法	164
4.2.3 CSS 基本属性	119	5.3.2 Session 对象程序应用实例	165
4.3 JSP 页面的基本结构	124	5.4 小结	172
4.4 JSP 注释	126		
4.4.1 隐藏注释	126		

5.5 习题.....	172	7.3 File 类.....	213
5.5.1 填空题.....	172	7.3.1 目录.....	214
5.5.2 选择题.....	172	7.3.2 创建目录.....	216
5.5.3 上机题.....	173	7.3.3 获取信息.....	217
第 6 章 JSP 其他内置对象.....	174	7.4 在页面中读写文件中的数据.....	218
6.1 Out 对象.....	174	7.4.1 读取操作.....	218
6.1.1 Out 对象的方法.....	174	7.4.2 写入操作.....	221
6.1.2 Out 对象程序应用实例.....	175	7.4.3 追加操作.....	224
6.2 PageContext 对象.....	177	7.4.4 在任意位置读取、写入 操作.....	226
6.3 Application 对象.....	178	7.5 文件操作实例.....	227
6.3.1 Application 对象的方法.....	178	7.5.1 文件的创建、检查与删除.....	227
6.3.2 Application 对象程序 应用实例.....	179	7.5.2 复制文件.....	230
6.4 Config 对象.....	184	7.5.3 是否为空白文件.....	232
6.4.1 Config 对象的方法.....	184	7.5.4 获取文件属性.....	233
6.4.2 Config 对象程序应用实例.....	184	7.5.5 保存网页.....	235
6.5 Exception 对象.....	185	7.6 小结.....	236
6.5.1 Exception 对象的方法.....	186	7.7 习题.....	237
6.5.2 Exception 对象程序 应用实例.....	186	7.7.1 填空题.....	237
6.6 小结.....	189	7.7.2 选择题.....	237
6.7 习题.....	189	7.7.3 上机题.....	238
6.7.1 填空题.....	189	第 8 章 在 JSP 中访问数据库.....	239
6.7.2 选择题.....	190	8.1 JDBC 概述.....	239
6.7.3 上机题.....	190	8.1.1 什么是 JDBC.....	239
第 7 章 JSP 中的文件操作.....	191	8.1.2 JDBC 与 ODBC.....	240
7.1 文件操作相关异常处理.....	191	8.1.3 JDBC 两层模型和三层 模型.....	241
7.1.1 异常处理概述.....	191	8.1.4 JDBC 接口.....	242
7.1.2 为什么要使用异常处理.....	192	8.2 使用 JDBC-ODBC 桥访问 数据库.....	245
7.1.3 异常类的层次.....	192	8.2.1 创建一个 Access 型数据库.....	245
7.1.4 异常的处理.....	192	8.2.2 创建 ODBC 数据源.....	245
7.1.5 自定义异常.....	199	8.2.3 编写 JSP 程序.....	247
7.2 JSP 的输入和输出流.....	200	8.2.4 JDBC-ODBC 桥访问 Access 数据库分析.....	249
7.2.1 字节流.....	200		
7.2.2 字符流.....	209		

8.3 使用 JDBC 驱动程序访问数据库	251
8.3.1 下载并安装 JDBC 驱动程序	252
8.3.2 创建 SQL Server 数据库	253
8.3.3 编写 JSP 程序	254
8.3.4 JDBC 直接访问 SQL Server 数据库分析	256
8.4 查询记录	258
8.4.1 SQL 的查询语句	258
8.4.2 JSP 发送执行查询语句	260
8.5 更新记录	262
8.5.1 SQL 的更新语句	262
8.5.2 JSP 发送执行更新语句	263
8.6 添加记录	263
8.6.1 SQL 的插入语句	264
8.6.2 发送执行插入语句	264
8.7 删除记录	265
8.7.1 SQL 的删除语句	265
8.7.2 JSP 发送执行删除语句	265
8.8 分页显示记录	265
8.8.1 分页显示实例	266
8.8.2 代码实现分析	268
8.9 查询 Excel 电子表格	271
8.9.1 创建 Excel 文件	271
8.9.2 创建 Excel 数据源	271
8.9.3 编写 JSP 文件	271
8.9.4 访问 Excel 表格数据实现分析	273
8.10 JSP 数据库应用实例	274
8.10.1 操作选择页面	274
8.10.2 添加学生信息	276
8.10.3 删除学生信息	280
8.10.4 查询学生信息	284
8.10.5 修改学生信息	286
8.11 小结	286
8.12 习题	286

8.12.1 填空题	286
8.12.2 选择题	287
8.12.3 上机题	287
第 9 章 JSP 与 Servlet 编程	288
9.1 Servlet 概述	288
9.1.1 什么是 Servlet	288
9.1.2 Servlet 的优点	288
9.1.3 Servlet 的用途	289
9.1.4 Servlet 与 JSP 的区别	290
9.1.5 Servlet 的方法	290
9.1.6 Servlet 的生命周期	294
9.1.7 Servlet 的类和接口	298
9.1.8 状态代码	301
9.2 编写一个简单的 Servlet	304
9.3 处理客户端提交的数据	306
9.4 返回数据到客户端	307
9.5 JSP+Servlet 编程实例	309
9.6 小结	316
9.7 习题	316
9.7.1 填空题	316
9.7.2 选择题	317
9.7.3 上机题	317
第 10 章 JSP 与 JavaBean 编程	318
10.1 为什么要使用 JavaBean	318
10.1.1 什么是 JavaBean	318
10.1.2 JavaBean 的种类	319
10.2 实现简单的 JavaBean	319
10.3 在页面中使用 JavaBean	323
10.3.1 JavaBean 的属性	324
10.3.2 在 JSP 中使用 JavaBean	327
10.4 通过 JavaBean 实现数据库的访问	337
10.4.1 连接数据库	337
10.4.2 数据查询	339
10.5 JSP 与 JavaBean 编程实例	341

10.6	小结	345
10.7	习题	345
10.7.1	填空题	345
10.7.2	选择题	346
10.7.3	上机题	346
第 11 章	在线书店	347
11.1	需求分析	347
11.2	系统分析设计	348
11.2.1	功能模块分析	348
11.2.2	数据库结构设计	349
11.2.3	界面设计	354
11.3	代码实现	360
11.3.1	通用模块	360
11.3.2	用户注册	364
11.3.3	用户登录	367
11.3.4	用户在线购书	368
11.3.5	用户查看订单	375
11.3.6	管理员登录	377
11.3.7	图书管理	378
11.3.8	订单管理	381
11.3.9	管理用户	385
11.4	程序的运行	389
11.5	小结	390
11.6	习题	391
11.6.1	填空题	391
11.6.2	上机题	391

第1章 概 述

JSP(Java Server Pages)是由 Sun Microsystems 公司倡导的、许多公司共同参与建立的一种动态网页技术标准。JSP 技术的设计目的是使得构造基于 Web 的应用程序更加容易和快捷,而这些应用程序能够与各种 Web 服务器、应用服务器、浏览器和开发工具共同工作。在 Sun 正式发布 JSP 之后,这种新的 Web 应用开发技术很快就引起了人们的关注,目前已经成为开发动态网页的主流技术之一。

1.1 Java 技术简介

1995 年,美国 Sun 公司倡导和推出了 Java 技术。Java 是一种简单的、完全面向对象的、跨平台且安全可靠的、主要面向 Internet 的开发工具。Java 技术包括 Java 语言和 Java Media APIs、Security APIs、Management APIs、Java Applet、Java RMI、Javabeans、Java OS、Java Servlet、JDBC、JNDI 等。自从 1995 年正式问世以来,Java 的快速发展已经极大地影响了整个 Web 世界的发展。随着 Java Servlet 的推出,Java 在电子商务方面开始崭露头角。Java Server Pages 技术的推出,更是让 Java 成为基于 Web 的应用程序的首选开发工具。

1.1.1 JavaBean

JavaBean 是 Java 的可重用组件。众所周知,ASP 通过 COM 来扩充复杂的功能,例如文件上传、发送 E-mail 以及将业务处理或复杂计算分离出来成为独立可重复使用的模块,而 JSP 通过 JavaBean 实现了类似的功能扩充。JSP 对于在 Web 应用中集成 JavaBean 组件提供了很好的支持,例如程序员可以直接使用经测试和可信任的已有组件,避免了重复开发,这样既节省了开发时间,也为 JSP 应用带来了更多的可伸缩性。JavaBean 组件还可以用来执行复杂的计算任务,或负责与数据库的交互以及数据提取等工作。

JavaBean 是一个十分完整的组件模型,它能够支持标准组件技术中的各种特性:属性(properties)、事件(events)、方法(method)和持续性(persistence),这和“对象”是十分相似的。设计一个 JavaBean 分为以下几个步骤:

- (1) 指定 Bean 的属性。
- (2) 指定 Bean 所产生或响应的事件。
- (3) 定义 Bean 应公开给其他 Bean 的属性、方法和事件。

JavaBean 开发起来很简单,且可以使用 Java 语言的强大功能。许多动态页面处理过程实际上被封装到了 JavaBean 中,所以 JSP 页面比传统的 ASP 或 PHP 页面更加简洁。

1.1.2 JDBC

JDBC 是用于执行 SQL 语句的 Java 应用程序接口，由一组用 Java 语言编写的类与接口组成。在 JSP 中将使用 JDBC 来访问数据库。JDBC 是一种规范，它让各数据库厂商为 Java 程序员提供标准的数据库访问类和接口，这样就使独立于 DBMS 的 Java 应用程序的开发工具和产品成为可能。一般来说，JDBC 类文件和 Java 小应用程序是存放在客户端，用来访问位于远程服务器上的数据库。

JDBC 可以通过数据库的 JDBC 驱动程序来访问数据库，当然就需要数据库的支持，例如 Oracle 就有 JDBC 驱动程序。JDBC 通过数据库的 JDBC 驱动程序来访问数据库，如图 1-1 所示。

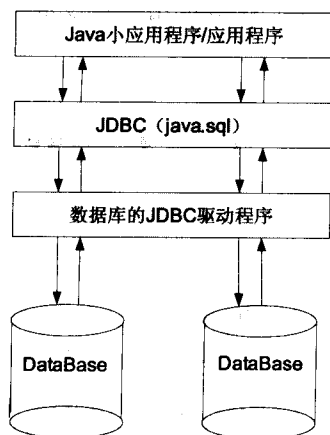


图 1-1 通过数据库的 JDBC 驱动程序来访问数据库

一般的 Java 开发工具都带有 JDBC-ODBC 桥驱动程序，这样只要是能够使用 ODBC 访问的数据库系统，也就能够使用 JDBC 访问，如图 1-2 所示。

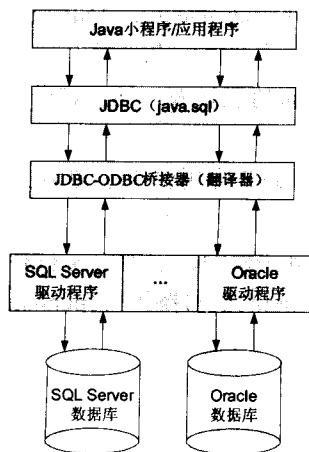


图 1-2 通过 JDBC-ODBC 桥驱动程序访问数据库

1.1.3 J2EE

随着电子商务和信息技术的快速发展,应用程序开发人员面临着许多新的压力:必须用比以前更少的金钱、更少的资源来更快地设计、建立和生产企业应用程序。

J2EE(Java 2 Platform Enterprise Edition)平台规范是一个由 Sun 公司定义的、用于简化分布式企业级应用开发与部署的基于、组件的模式。为了降低成本,加快企业应用程序的设计和开发, J2EE 平台提供了一个基于组件的方法来设计、开发、装配及部署企业应用程序,并提供了多层的分布式应用模型、组件再用、一致化的安全模型以及灵活的事务控制。因此,程序员可以以更快的速度向市场推出创造性的客户解决方案,并且独立的、基于组件的 J2EE 解决方案不会被任何一个厂商的产品和 API 所束缚。

J2EE 平台由一整套服务(Services)、应用程序接口(APIs)和协议构成,它对开发基于 Web 的多层应用提供了功能支持。

1. J2EE 的层和组件

J2EE 使用多层的分布式应用模型,其应用逻辑按功能划分为组件,各个应用组件根据它们所在的层分布在不同的机器上。J2EE 分为 4 个层:客户层(Client Tier)、Web 层(Web Tier)、业务层(Business Tier)和企业信息系统层(Enterprise Information System Tier)。

客户层运行在客户端机器上。应用于客户层的组件是“应用的客户组件(Application Client Components)”,它可以是基于 Web 方式的,即作为 Web 服务器的浏览器,也可以是基于传统方式的(非基于 Web 方式),即独立的应用程序,可以完成瘦客户机无法完成的任务。

J2EE 应用程序可以是基于 Web 的,也可以是不基于 Web 的。在一个基于 Web 的 J2EE 应用程序中,用户的浏览器在客户层中运行,并从一个 Web 服务器上下载 Web 层中的静态 HTML 页面或由 JSP(或 Servlets)生成的动态 HTML 页面。

Web 层运行在 Web 服务器上,Java Servlet 和 Java Server Pages(JSP)是 Web 层组件。调用 Servlet 或者 JSP 页面的 HTML 页面在应用程序组装时与 Web 组件打包在一起。与客户层相似,Web 层可能包括一个 JavaBean 类来管理用户输入,并将输入发送到在业务层中运行的 Enterprise Bean 类来进行处理。

业务层运行在 EJB 服务器上,Enterprise Java Bean(EJB)是业务层组件。一个 Enterprise Bean 从客户程序处接收数据,对数据进行处理(如果需要),再将数据发送到企业信息系统层中存储。一个 Enterprise Bean 还可以从存储中检索数据,并将数据送回客户程序。运行在业务层的 Enterprise Bean 依赖于容器来为诸如事务、生命期、状态管理、多线程及资源存储池提供通常都非常复杂的系统级代码。

Enterprise Bean 分为会话(Session)Bean、实体(Entity)Bean 和消息驱动(Message-driven)Bean 3 类。会话 Bean 表示与客户端程序的临时交互;实体 Bean 表示数据库的表中一行永久的记录;消息驱动 Bean 结合了会话 Bean 和 JMS 消息监听器的特性,允许一个业务层组件异步接收 JMS 消息。

业务层经常被称作 Enterprise JavaBean (EJB)层。业务层和 Web 层一起构成了 3 层 J2EE 应用的中间层,而其他两层是客户层和企业信息系统层。

企业信息系统层运行在 EIS 服务器上,这一层上运行企业信息系统软件。

2. 设计用户界面和引擎

在为 J2EE 应用程序设计用户界面和后端引擎时,需要考虑该程序是否要基于 Web,还要考虑平台配置、下载速度、安全、网络流量和网络服务等。例如,下载一个包含有用用户界面并且经常被大量用户访问的 applet 可能需要花很长的时间,然而如果知道该 applet 要运行在一个公司的内部网内的受控环境中,那么,在这种情况下,该 applet 将拥有一个完全可接受的下载速度。

此外,设计应用程序时,还要考虑繁重的处理应当在哪儿执行。例如,如果客户程序在一个蜂窝电话或呼机中执行,服务器应当完成尽量多的计算和数据处理,而客户程序只应显示结果就可以了。然而,设计在一个强大的台式机平台上运行的大型财务分析系统则应当在客户机上完成其复杂计算。应用的客户程序和 applet 用户界面通常都是用 Swing API 创建的,该 API 可从标准版 Java 2 平台(Java 2 Platform, Standard Edition)中获得。Swing API 提供了一整套 GUI 组件(表格、树形结构、按钮等),Swing 也支持 HTML 文本组件,这个组件可以用来显示来自一个服务器的响应。客户程序可以直接访问 Enterprise Bean 层或企业信息系统层,但应谨慎实现这种程序。绕过 EJB 层的程序可以使用 JDBC API 来访问一个关系型数据库,但应被限制在对数据库表格进行维护等这样的管理任务上。

3. 设计基于 Web 的应用程序

基于 Web 的应用程序是基于浏览器的,如果它们运行在 Internet 上,可能会被全世界的人访问。当设计一个基于 Web 的应用程序时,不仅需要决定用什么来处理内容和应用逻辑(HTML, XML, JSP 页面及 Servlet),还应当考虑如何使该应用程序国际化。

可以在 J2EE 平台上实现的基于 Web 的应用程序有 4 种。从简单到复杂排列,它们是:

- 基本 HTML
- 带基本 JSP 页面或 Servlet 的 HTML
- 带 JavaBean 类的 JSP 页面
- 将应用逻辑根据功能划分成区域的高度结构化的应用

当设计一个基于 Web 的应用程序时,需要决定用什么来建立它。假设是从建立一个简单的应用程序开始着手,如果认为以后会给该应用程序添加功能,那么当初的设计应当适应今后发展的需要。

Servlet 是实现动态内容的一种简便的、平台独立、Web 服务器独立的方式。JSP 页面是开发 Servlet 的一种基于文本的、以显示为中心的方式。JSP 页面提供了 Servlet 所有的优点,并且当与一个 JavaBean 类结合在一起时,提供了一种使内容和显示逻辑分开的简单方式。分开内容和显示逻辑的好处是,更新页面外观的人员不必懂得 Java 代码,而更新 JavaBeans 类的人员也不必是设计网页的行家里手。JSP 页面和 Servlet 都比公共网关接口

(CGI)更可取, 因为 CGI 不是平台独立的, 它使用了更多系统开销, 而且没有一个简单的方法来访问参数数据, 并将这些数据发送给一个程序。可以用带 **JavaBean** 类的 **JSP** 页面来定义 **Web** 模板, 以建立一个由具有相似外观的页面组成的网站。**JavaBean** 类可以完成数据提供, 这样在模板中就没有 **Java** 代码, 这意味着这些模板可以由一个 **HTML** 编写人员来维护。

在选择使用一个 **Servlet** 还是一个 **JSP** 页面时, 要注意的是, **Servlet** 是一个程序设计工具, 它最适用于不需要频繁修改的低级应用功能; 而 **JSP** 页面则通过以显示为中心的描述性的方法将动态内容和逻辑结合在一起。对于一个使用 **JSP** 页面的, 简单的, 基于 **Web** 的应用程序而言, 可以使用定制标记或者 **Scriptlets**(**Scriptlets** 是直接嵌入在 **JSP** 页面中的很小的 **Java** 代码段), 而不是使用 **JavaBean** 类来将内容与应用逻辑结合起来。定制标记被打包到一个标记库中, 并被引入到一个 **JSP** 页面中。

一个国际化的基于 **Web** 的应用程序首先要让用户选择一种语言, 然后, 根据该选定语言加载应用正文方式。对被支持的每种语言而言, 应用正文都被存储在一个外部文件中, 并且与另外一个文件的关键词相对应, 应用代码使用这些关键词以及选定的语言来加载正确的文本。国际化 **API** 还提供类来根据选定的语言格式化日期和货币符号。一旦制订了使应用程序国际化的细节, 就可以决定用什么来实现它了。

概括起来, 一个基于 **Web** 的应用程序使用 **HTML** 来显示数据; 使用 **XML** 来定义数据以使其可被另一个程序读取并处理; 使用 **JSP** 页面或 **Servlets** 来管理用户与业务层或存储层之间的数据流。

1.1.4 EJB

EJB(**Enterprise JavaBean**)是由 **Sun** 公司提出的基于 **Java** 的面向对象的组件标准, 和原来的 **JavaBean** 不同, **EJB** 组件包含一定的业务规则, 运行在服务器端。在目前的企业计算环境中, **EJB**、**COM** 和 **CORBA** 并列为三大组件标准。

EJB 是用于开发和部署多层结构的、分布式的、面向对象的 **Java** 应用系统跨平台构件体系结构。采用 **EJB** 可以使得开发商业应用系统变得容易。应用系统可以在一个支持 **EJB** 的环境中开发, 开发完之后部署在其他环境中。随着需求的改变, 应用系统可以不加修改地迁移到其他功能更强、更复杂的服务器上。

EJB 简化了多层体系结构应用系统的开发过程。在分布式应用系统的开发中, 采用多层体系结构的方法有很多优点, 例如增加了应用系统的可伸缩性、可靠性、灵活性等。服务器端构件可以根据应用需求迅速地加以修改, 且构件在网络中的位置和应用无关, 因此系统管理员可以很容易地重新配置系统的负载。

EJB 通过使用 **RMI** 支持远程的客户端存取。**RMI** 是一个高级程序设计接口, 它使得服务器的位置对于客户机来说是透明的。**RMI** 产生一个对象, 这个对象被安装在客户机系统中, 作为存取服务器对象的代理对象, **EJB** 开发人员为每一个可存取的接口定义一个 **Java** 远程接口。

EJB 通过使用 IIOP 协议(Internet Inter ORB Protocol)也可以和其他的非 Java 客户机进行通信。IIOP 允许 EJB 系统和 CORBA 的集成, EJB 可以存取 CORBA 服务器, CORBA 客户机也可以存取 EJB 服务器。

1. EJB 的体系结构

EJB 的体系结构如图 1-3 所示。

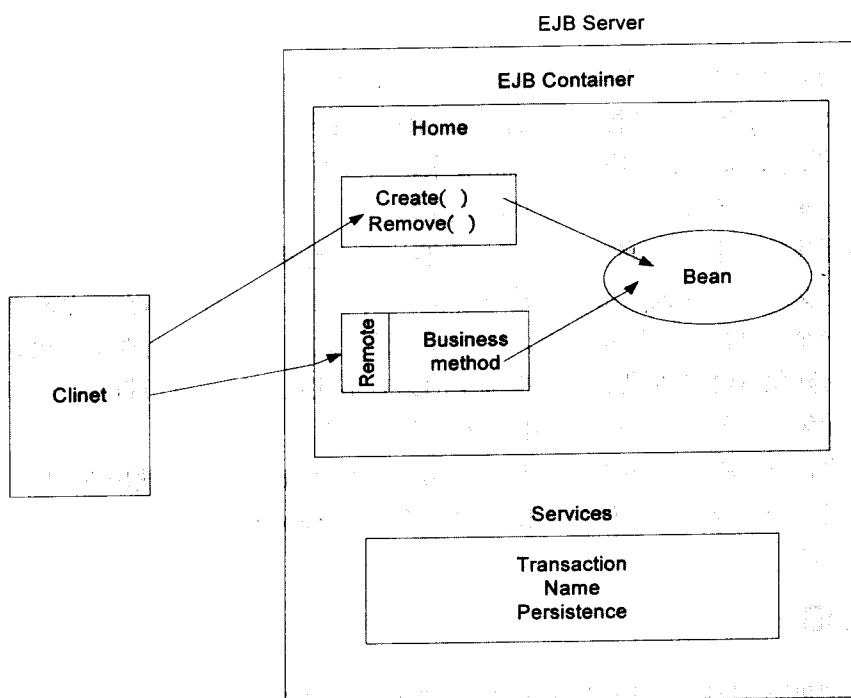


图 1-3 EJB 的体系结构

- **EJB 服务器:** EJB 服务器作为容器和低层平台的桥梁管理着 EJB 容器和函数, 是管理 EJB 容器的高端进程或应用程序, 它向 EJB 容器提供了访问系统服务的能力。例如, 数据库的管理和事务的管理, 或者对于其他 Enterprise 的应用服务器。EJB 服务器也可以提供厂商自己的特性, 如优化的数据库访问接口。一个 EJB 服务器必须提供对可访问 JNDI 的名字服务和事务服务的支持。
- **EJB 容器:** 用于管理 EJB 组件, 主要负责对象生命周期的管理。所有的 EJB 实例都运行在 EJB 容器中。容器提供了系统级的服务, 控制了 EJB 的生命周期, 它通过规范中定义的接口使 EJB 类访问所需的服务。容器厂商也可以在容器或服务器中提供额外服务的接口。
- **远程接口(Remote Interface):** 列出了 EJB 类中的商业方法。
- **Home 接口:** 列出了所有定位、创建、删除 EJB 类实例的方法。Home 对象是 Home 接口的实现, EJB 类开发者必须定义 Home 接口, 容器厂商应该提供从 Home 接口中产生 Home 对象实现的方法。

- **EJB Object:** 实现远程接口, 并且客户端通过它访问 EJB 实例的商业方法。EJB 开发者定义远程接口, 容器开发商提供产生相应的 EJB Object 的方法。客户端不能得到 EJB 实例的引用, 只能得到它的 EJB Object 实例的引用。当客户端调用一个方法, EJB Object 接受请求并把它传给 EJB 实例, 同时提供进程中必要的包装功能。客户端应用程序通过 Home 对象来定位、创建、删除 EJB 类的实例, 通过 EJB Object 来调用实例中的商业方法。

2. EJB 的组件

EJB 定义了一组可重用的组件 Enterprise Bean。开发人员可以利用这些组件, 像搭积木一样建立分布式应用程序。当编辑好这些之后, 这些组件就被组合到特定的文件中。每个文件有一个或多个 Enterprise Bean, 再加上一些配置参数。最后, 这些 Enterprise Beans 被配置到一个装载了 EJB 容器的平台上, 客户能够通过这些 Bean 的 Home 接口, 定位到某个 Bean, 并产生这个 Bean 的一个实例。这样, 客户就能够调用 Bean 的应用方法和远程接口了。EJB 主要有以下 2 种组件。

- **Session Bean (会话 Bean)**

Session Beans 是一种作为单个的 client(客户程序)执行的对象。作为对远程的任务请求的响应, 容器产生一个 Session Bean 的实例。一个 Session Bean 对应一个 client, 从某种程度上来说, 一个 Session Beans 对于服务器来说就代表了它的那个 client。Session Bean 也能用于事务, 它能够更新共享的数据, 但它不直接描绘这些共享的数据。

Session Bean 的生命周期相对较短。典型的是, 只有当 client 保持会话的时候, Session Bean 才是活动的; 一旦 client 退出了, Session Bean 就不再与 client 相联系了。Session Bean 是瞬时的, 因为如果容器崩溃了, 那么 client 必须重新建立一个新的 Session Bean 对象来继续会话。

Session Beans 可以有状态的, 也可以是无状态的。一个有状态的 Session Beans 包含了代表 client 的会话状态, 该会话状态是该 Session Beans 实例的域值和这些域值所引用到的所有对象。有状态 Session Beans 并不代表在一个持久数据存储中的数据, 但是, 它可以代表客户程序访问和更新数据。

无状态 Session Beans 不包含用于某个特定客户程序的任何状态信息, 它们通常用于提供不保持任何特定状态的服务器端行为。client 能够调用 Bean 的方法来完成一些操作, 但是 Bean 只是在方法调用的时候才知道 client 的参数变量。当方法调用完成以后, Bean 并不继续保持这些参数变量。所以, 所有的无状态 Session Bean 的实例都是相同的, 除非它正处在方法调用期间。这样, 无状态的 Session Bean 就能够支持多个 client, 容器能够声明一个无状态的 Session Bean, 并将其指定给任何 client。

- **Entity Bean(实体 Bean)**

Entity Bean 代表一个数据库中的数据及作用于该数据的方法。例如, 一个 Entity Bean 能够代表数据库表中的一行相关的记录。多个 client 能够共享访问同一个 Entity Bean, Entity Bean 通过事务的上下文来访问或更新下层的数据, 这样就能够保证数据的完整性。