

21世纪高等职业教育计算机系列规划教材

# 基于Java技术的 Web应用开发

孙璐 主编

罗国涛 项巧莲 戴上平 副主编

- 以一个项目贯穿教材的各个章节，理论适度、重在应用、重在会用
- 注重实用，培养应用能力和岗位工作能力

配备课件



电子工业出版社  
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>

21 世纪高等职业教育计算机系列规划教材

# 基于 Java 技术的 Web 应用开发

孙 璐 主 编

罗国涛 项巧莲 戴上平 副主编

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

## 内 容 简 介

本书以 ServletAPI2.3 和 JSP1.2 规范为基础,介绍了应用 Java 技术实现 Web 应用开发的相关技术及编程方法,是一本实用教程。

全书共 9 章,内容包括 Web 应用体系架构、HTML、Servlet 技术、JavaScript 技术、JSP 技术、JavaBean 组件技术、数据库应用等。学生在掌握这些知识后,将具有开发电子商务网站、电子政务、企业信息等项目的能力。本书较全面地体现了应用 Java 技术实现 Web 应用开发的发展特性,涉及当前应用广泛的开发规范,结构清晰,应用实例丰富,实现了理论学习和具体应用的充分结合。

本书可作为各大专院校、高等职业院校计算机专业的教材,也可供从事基于 Java 技术的 Web 应用开发的技术人员学习参考。本书同时提供免费下载的教学课件和所有程序源代码。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

图书在版编目(CIP)数据

基于 Java 技术的 Web 应用开发/孙璐主编. —北京:电子工业出版社,2009.8

(21 世纪高等职业教育计算机系列规划教材)

ISBN 978-7-121-09286-2

I. 基… II. 孙… III. ①JAVA 语言—程序设计—高等学校:技术学校—教材 ②主页制作—程序设计—高等学校:技术学校—教材 IV. TP312 TP393.092

中国版本图书馆 CIP 数据核字(2009)第 123061 号

策划编辑:徐建军

责任编辑:韩玉宏

印 刷:北京市天竺颖华印刷厂

装 订:三河市鑫金马印装有限公司

出版发行:电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本:787×1 092 1/16 印张:14 字数:358 千字

印 次:2009 年 8 月第 1 次印刷

印 数:4 000 册 定价:26.00 元

凡所购买电子工业出版社图书有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系,联系及邮购电话:(010) 88254888。

质量投诉请发邮件至 zlts@phei.com.cn,盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线:(010) 88258888。

# 前 言

基于 Java 技术的 Web 应用开发是 Java 技术教学的第三阶段,该课程的先导课程包括网页制作、Java 语言程序设计、Java 网络编程、数据库应用等。本书介绍了 Web 应用体系架构、HTML、Servlet 技术、JavaScript 技术、JSP 技术、JavaBean 组件技术及数据库应用。学生在掌握这些知识的基础上,将具有利用 Java 开发各种基于 B/S 架构的电子商务与信息化项目的能力。

本书各章相对独立,按进行 B/S 架构项目开发所应有的知识来组织章节。本着“理论适度、重在应用、重在会用”的原则来确定各章内容的深度和广度。理论叙述尽量简单,形象;例子既注重简洁,又注重实用及代码的完整性。并且用一个项目贯穿本书的各个章节,有利于学生提高项目开发的能力,使其能与项目开发无缝接轨。

本书详细讲解了进行 Web 应用开发的所需知识。针对较难理解的问题,从简单到复杂,逐步深入,便于学生掌握。本书分为 9 章。第 1 章和第 2 章介绍 Web 应用体系架构和环境变量的设置。第 3 章介绍贯穿全书的项目的框架及功能模块的划分。第 4 和第 6 章讲述项目前台开发,包括 HTML 和 JavaScript 技术。第 5 章和第 7~9 章讲述项目后台开发,包括 Servlet 技术、JSP 技术、JavaBean 组件技术和数据库应用。

本书面向的读者是具有一定 Java 语言基础的人。本书可作为各大专院校、高等职业院校计算机专业的教材,也可供从事基于 Java 技术的 Web 应用开发的技术人员学习参考。

本书由孙璐、罗国涛组织编写。在编写过程中得到了四川托普信息技术职业学院领导的指导和支持,特别是得到了刘勇军的大力帮助。参加编写工作的还有中南民族大学的项巧莲和华中师范大学的戴上平等。全书由孙璐统稿和审读。在本书的编写过程中得到了各方面的大力支持,在此一并表示感谢。

为了方便教学,请登录[www.huaxin.edu.cn](http://www.huaxin.edu.cn)或[www.hxedu.com.cn](http://www.hxedu.com.cn)免费下载与本书配套的教学资料。需要课件的读者也可以与编者联系(sunlugogo@163.com)。

由于编者水平有限,加上时间仓促,书中难免有不当之处,敬请各位同行批评指正,以便在今后的修订中不断改进。

编 者



# 目 录

|                  |      |
|------------------|------|
| 第1章 Web 应用体系架构   | (1)  |
| 1.1 Web 应用的发展历史  | (1)  |
| 1.2 HTTP 请求响应模型  | (6)  |
| 1.3 动态网页技术介绍与比较  | (7)  |
| 习 题              | (8)  |
| 第2章 环境变量的设置及开发实例 | (9)  |
| 2.1 环境变量的设置      | (9)  |
| 2.1.1 Tomcat 的安装 | (9)  |
| 2.1.2 JDK 的安装    | (10) |
| 2.1.3 设置环境变量     | (11) |
| 2.2 开发实例及发布运行    | (11) |
| 2.2.1 HTML 程序    | (11) |
| 2.2.2 Servlet 程序 | (12) |
| 2.2.3 JSP 程序     | (13) |
| 习 题              | (14) |
| 第3章 人力资源项目概述     | (15) |
| 3.1 系统概述         | (15) |
| 3.1.1 现状         | (15) |
| 3.1.2 建设目标       | (16) |
| 3.2 解决方案         | (16) |
| 3.3 系统架构         | (16) |
| 3.4 系统功能设计       | (17) |
| 3.4.1 系统管理       | (18) |
| 3.4.2 人员管理       | (19) |
| 3.4.3 培训管理       | (19) |
| 3.4.4 补贴统计       | (21) |
| 3.4.5 人员考核       | (21) |
| 3.4.6 事务管理       | (22) |
| 3.4.7 制度管理       | (26) |
| 3.4.8 消息管理       | (26) |
| 3.5 开发和部署        | (26) |
| 习 题              | (27) |
| 第4章 HTML         | (28) |
| 4.1 HTML 概述      | (28) |
| 4.1.1 HTML 的基本格式 | (28) |
| 4.1.2 HTML 的常用标记 | (29) |
| 4.2 文本处理         | (30) |

# 目 录

|       |                               |      |
|-------|-------------------------------|------|
| 4.2.1 | 标题处理                          | (30) |
| 4.2.2 | 字体设置                          | (31) |
| 4.2.3 | 列表                            | (32) |
| 4.2.4 | 特殊符号                          | (33) |
| 4.3   | 超链接处理                         | (33) |
| 4.4   | 图像处理                          | (34) |
| 4.4.1 | 图像的插入                         | (34) |
| 4.4.2 | 设置图像的属性                       | (34) |
| 4.5   | 表单设计                          | (35) |
| 4.5.1 | 表单的基本结构                       | (35) |
| 4.5.2 | 文本框和密码框                       | (36) |
| 4.6   | 表格设计                          | (36) |
| 4.7   | HTML 在项目中的应用                  | (37) |
| 4.8   | 实训操作                          | (40) |
| 习 题   |                               | (40) |
| 第 5 章 | Servlet 技术                    | (42) |
| 5.1   | Servlet 简介                    | (42) |
| 5.1.1 | Servlet 概述                    | (42) |
| 5.1.2 | Servlet 的处理流程                 | (43) |
| 5.1.3 | Servlet 的基本结构                 | (43) |
| 5.1.4 | 生成 HTML 的 Servlet             | (45) |
| 5.1.5 | Servlet 的生命周期                 | (46) |
| 5.2   | Servlet API                   | (48) |
| 5.2.1 | 通过继承 GenericServlet 类         | (48) |
| 5.2.2 | 通过继承 HttpServlet 类            | (50) |
| 5.3   | Servlet 对表单的处理                | (52) |
| 5.4   | 使用 Cookie                     | (57) |
| 5.4.1 | Cookie 的概念                    | (57) |
| 5.4.2 | 检测浏览器是否支持 Cookie              | (58) |
| 5.4.3 | Cookie 的使用                    | (58) |
| 5.5   | Servlet 实现在页面之间的跳转            | (61) |
| 5.5.1 | forward()方法                   | (61) |
| 5.5.2 | include()方法                   | (62) |
| 5.6   | Servlet 在项目中的应用               | (64) |
| 5.7   | 实训操作                          | (74) |
| 习 题   |                               | (75) |
| 第 6 章 | JavaScript 技术                 | (78) |
| 6.1   | JavaScript 概述                 | (78) |
| 6.1.1 | 如何将 JavaScript 脚本嵌入到 HTML 文档中 | (79) |
| 6.1.2 | JavaScript 脚本在 HTML 中的位置      | (82) |

|       |                        |       |
|-------|------------------------|-------|
| 6.1.3 | 事件处理机制                 | (82)  |
| 6.2   | JavaScript 的基本语法       | (82)  |
| 6.2.1 | 数据类型及变量                | (82)  |
| 6.2.2 | 运算符与表达式                | (85)  |
| 6.2.3 | 程序结构                   | (86)  |
| 6.3   | JavaScript 的对象         | (90)  |
| 6.3.1 | Array, Date, String 对象 | (91)  |
| 6.3.2 | 窗口对象                   | (92)  |
| 6.3.3 | 文档对象                   | (96)  |
| 6.3.4 | 表单对象                   | (98)  |
| 6.3.5 | 历史对象                   | (100) |
| 6.4   | 个人用户注册页面的实现            | (101) |
| 6.4.1 | 验证方法                   | (101) |
| 6.4.2 | 实训: 仿照上例, 完成注册页面的验证    | (102) |
| 6.5   | JavaScript 在项目中的应用     | (103) |
| 6.5.1 | 登录实现                   | (103) |
| 6.5.2 | 用户管理实现                 | (105) |
| 6.6   | 实训操作                   | (112) |
| 习 题   |                        | (112) |
| 第 7 章 | JSP 技术                 | (115) |
| 7.1   | JSP 简介                 | (115) |
| 7.1.1 | 一个简单的 JSP 程序           | (116) |
| 7.1.2 | 运行方式                   | (117) |
| 7.1.3 | JSP 的生命周期              | (117) |
| 7.1.4 | 处理汉字信息                 | (118) |
| 7.2   | JSP 元素                 | (118) |
| 7.2.1 | JSP 脚本                 | (118) |
| 7.2.2 | JSP 指令                 | (122) |
| 7.2.3 | JSP 动作                 | (128) |
| 7.3   | JSP 的内置对象              | (135) |
| 7.3.1 | request 对象             | (135) |
| 7.3.2 | response 对象            | (140) |
| 7.3.3 | session 对象             | (142) |
| 7.3.4 | application 对象         | (145) |
| 7.3.5 | page 对象                | (145) |
| 7.3.6 | out 对象                 | (145) |
| 7.3.7 | pageContext 对象         | (145) |
| 7.3.8 | exception 对象           | (145) |
| 7.4   | JSP 在项目中的应用            | (145) |
| 7.5   | 实训操作                   | (152) |

|                                             |       |
|---------------------------------------------|-------|
| 习 题                                         | (153) |
| 第 8 章 JavaBean                              | (155) |
| 8.1 JavaBean 概述                             | (155) |
| 8.2 JavaBean 的编写和使用                         | (156) |
| 8.2.1 JavaBean 的编写                          | (156) |
| 8.2.2 JavaBean 的使用                          | (157) |
| 8.3 获取和设置 JavaBean 的属性值                     | (159) |
| 8.3.1 获取 JavaBean 的属性值                      | (159) |
| 8.3.2 设置 JavaBean 的属性值                      | (161) |
| 8.4 JavaBean 在项目中的应用                        | (167) |
| 8.5 实训操作                                    | (172) |
| 习 题                                         | (172) |
| 第 9 章 数据库应用                                 | (174) |
| 9.1 数据源与 JDBC                               | (174) |
| 9.1.1 JDBC 概述                               | (174) |
| 9.1.2 JDBC API                              | (175) |
| 9.1.3 在 JSP 或 Servlet 中访问 SQL Server 数据库的步骤 | (176) |
| 9.2 查询记录                                    | (181) |
| 9.2.1 根据条件查询记录                              | (181) |
| 9.2.2 对查询的记录进行排序输出                          | (185) |
| 9.2.3 通配符查询                                 | (187) |
| 9.2.4 preparedStatement()方法的应用              | (190) |
| 9.3 修改记录                                    | (191) |
| 9.4 添加记录                                    | (196) |
| 9.5 删除记录                                    | (199) |
| 9.6 数据库连接池                                  | (201) |
| 9.7 数据库在项目中的应用                              | (206) |
| 9.8 实训操作                                    | (211) |
| 习 题                                         | (211) |



## 第 1 章 Web 应用体系架构

### 📖 知识点

- B/S 架构
- HTTP 请求响应模型

### 🔊 难点

- B/S 架构

### 🔍 掌握

- Web 应用的发展历史
- HTTP 请求响应模型

### 任务引入

本章主要介绍 Web 应用的体系架构, 主要包括 Web 应用的发展历史、HTTP 请求响应模型及动态网页技术介绍与比较。

### 1.1 Web 应用的发展历史

早期的 Web 应用仅仅是对静态信息的访问, 服务器根据用户的请求把相应的静态的 HTML 页面传送给客户端进行显示。应用程序不会对请求的信息进行任何的处理。

例 1\_1: 静态页面的实现。

Ex1\_1\_1.html

```
<html>
<head>
  <title>Web 应用-----静态显示</title>
</head>
<body>
  <a href="Ex1_1_2.html">本书的写作特点</a>
</body>
</html>
```

Ex1\_1\_2.html

```
<html>
<head>
  <title>显示静态内容</title>
</head>
<body>
```

本书立足于高职高专, 语言浅显易懂, 对进行 Web 开发的各个环节都予以介绍, 并用一个项目贯穿本书的各个章节。理论知识不求全面, 以必需、够用为度。重点在于实用技术。

```
</body>
```

```
</html>
```

后来出现了 Java 的 Applet 小应用程序。当用户发出请求后, Applet 随着页面一起被下载到客户端, Applet 在浏览器的 JVM 中运行, 能够提供动态的页面内容。从某种程度说, Applet 扩展了 Web 服务器的功能。

例 1\_2: Applet 的应用。

HelloApplet.java

```
import java.awt.*;
import java.applet.*;

public class HelloApplet extends Applet {

    public void init() {
    }

    public void paint(Graphics g) {
        g.drawString("Welcome to Java!!", 50, 60);
    }
}
```

HelloTest.html

```
<HTML>
<HEAD>
</HEAD>
<BODY>
<CENTER>
<APPLET
    code = "HelloApplet.class"
    width= 500
    height= 300
>
</APPLET>
</CENTER>
</BODY>
</HTML>
```

由于 Applet 不能对数据库进行访问, 慢慢就出现了在服务器端运行的 Servlet。Servlet 实际上也是一段 Java 程序, 它会根据用户提交数据的不同从而响应不同的页面, 实现动态 Web 应用。

Servlet 是运行在服务器端的 Web Container 的 Web 容器中, 当一个 HTTP 请求传到服务器, Web Server Plugin 会检测这次请求的是静态资源还是动态资源。如果是静态资源, Web Server Plugin 会将请求传递给 Web Server, Web Server 将请求直接映射到某个页面,

不加任何改变地将页面传回给客户端。但如果请求的是动态资源, Web Server Plugin 会将请求传递给 Web Container, Web Container 会调用相应的 Servlet, 根据用户提交数据的不同, 将动态内容传回给客户端。

### 例 1\_3: Servlet 的应用。

#### Ex1\_3.html

```
<html>
<body bgcolor="cyan" >
<font color="" size="8">
<form action="Ex1_3" method="get">
    <input type="text" name="paraTxt">
    <input type="submit" value="提交">
</form>
</body>
</html>
```

#### Ex1\_3.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;

public class Ex5_5 extends HttpServlet
{
    public void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException
    {
        response.setContentType("text/html;charset=GB2312");
        PrintWriter out=response.getWriter();
        out.println("<html>");
        out.println("<head><title>Servlet 的应用</title></head>");
        out.println("<body>");
        out.println("<hr width=80% align=left color=blue>");

        String para=request.getParameter("paraTxt");
        out.print("<br>"+para);
        out.println("<br><br><hr width=80% align=left color=blue>");
        out.println("</body>");
        out.println("</html>");
    }
}
```

在 Ex1\_3.html 文件中定义了一个文本框和一个提交按钮, 当用户在文本框中输入内容, 单击提交按钮后, 服务器端的 Servlet 会得到用户的提交信息, 并将其显示出来。根据用户



输入的内容不同, 显示不同的信息, 完成动态页面的显示。

但 Servlet 的缺点在于, 如果仅仅使用 Servlet 往往会把业务逻辑和显示逻辑混合到一起。例如, 上面的 Servlet 程序将用户信息的获取和显示放到一起, 程序的可读性变得很差。这时就出现了 JSP。

JSP 可以用来实现更好的 MVC 模式, 也就是分离视图、控制和业务逻辑, 这样可以使应用程序的结构更加清晰。视图层一般由界面设计人员和界面程序员来完成, 随着用户需求的变化而变化; 模型层则由业务逻辑程序员来完成, 根据流程的变化而变化; 控制层则一般由负责整体控制的程序员来完成, 这部分代码比较稳定, 一般会实现一个通用的架构。这种模块功能的划分有利于在代码修改过程中进行模块的隔离, 而不需要把具有不同功能的代码混杂在一起造成混乱。用 JSP 页面来显示给用户的数据, 用 Servlet 控制页面的流程。Servlet 会根据请求信息的不同或应用程序逻辑的设定而调用相应的 JSP 页面。

#### 例 1\_4: JSP 页面的实现。

input.html

```
<html>
<head>
<title>input.html</title>
</head>
<body>
<form action="FirstServlet" method="get">
    <input type="text" name="paraTxt">
    <input type="submit" value="提交">
</form>
</body>
</html>
```

FirstServlet.java

```
package com;
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;

public class FirstServlet extends HttpServlet{
    public void doGet(HttpServletRequest request,HttpServletResponse response) throws
ServletException, IOException
    {
        HttpSession session=request.getSession();
        String para=request.getParameter("paraTxt");
        System.out.println(para);
        session.setAttribute("para", para);
        response.sendRedirect("showInfo.jsp");
    }
}
```

```
}
```

### showInput.jsp

```
<%@ page language="java" import="java.util.*" pageEncoding="GB2312"%>
<%
String path = request.getContextPath();
String basePath=request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";
%>
<html>
<head>
<title>My JSP 'showInfo.jsp' starting page</title>
</head>
<body>
    用户输入的信息为: <br>
    <%
        String para=(String)session.getAttribute("para");
    %>
    <%=para %>
</body>
</html>
```

EJB 是企业级的 JavaBean，它提供了对业务逻辑数据的封装。JavaBean 可以通过 Web Container 来访问 EJB。Java 应用程序客户端也可以访问 EJB。

### 例 1\_5：简单的 EJB 应用。

#### HelloWorld.java

```
package com.foshanshop.ejb3;

public interface HelloWorld {
    public String SayHello(String name);
}
```

#### HelloWorldBean.java

```
package com.foshanshop.ejb3.impl;
import com.foshanshop.ejb3.HelloWorld;
import javax.ejb.Remote;
import javax.ejb.Stateless;
public class HelloWorldBean implements HelloWorld {

    public String SayHello(String name) {
        return name +"说： 你好!世界,这是我的第一个 EJB3 哦。";
    }
}
```



## HelloWorldTest.java

```

package junit.test;
import static org.junit.Assert.*;
import java.util.Properties;
import javax.naming.InitialContext;
import org.junit.BeforeClass;
import org.junit.Test;
import com.foshanshop.ejb3.HelloWorld;
/**
 * 单元测试用例，运行该用例时，你需要把[Jboss 安装目录]/client 下的 jar 加入项目的类路径
 *
 */
public class HelloWorldTest {
    protected static HelloWorld helloworld;

    @BeforeClass
    public static void setUpBeforeClass() throws Exception {
        Properties props = new Properties();
        props.setProperty("java.naming.factory.initial",
            "org.jnp.interfaces.NamingContextFactory");
        props.setProperty("java.naming.provider.url", "localhost:1099");
        InitialContext ctx = new InitialContext(props);
        helloworld = (HelloWorld)ctx.lookup("HelloWorldBean/remote");
    }

    @Test
    public void testSayHello() {
        String result = helloworld.SayHello("佛山人");
        //System.out.println(result);
        assertEquals("佛山人说：你好!世界,这是我的第一个 EJB3 哦.", result);
    }
}

```

## 1.2 HTTP 请求响应模型

Web 应用是基于 B/S 架构（也就是浏览器/服务器架构）的。应用程序部署在服务器端，客户端通过浏览器访问应用程序。客户端发送 HTTP 请求消息给服务器，服务器将请求传递给 Web 应用程序，Web 应用程序处理请求，并把响应的 HTML 页面传回给客户端，所以说 HTTP 协议基于请求响应模型。

一个完整的 HTTP 会话过程包括：首先，客户端与 Web 服务器建立连接，客户端向

Web 服务器发送 HTTP 请求消息，Web 服务器处理请求，并将响应消息传送给客户端，这样一个来回后，这个连接就关闭了。

HTTP 超文本传输协议是一个无状态的协议。也就是说，每当客户端访问 Web 服务器上的某个 Web 页面时，都要建立与服务器的一个独立的连接。服务器不保留前一次访问的任何信息。Web 服务器将客户端对某个页面的每次访问都当做相互无关的访问来处理；服务器不会自动保留用户的状态信息。

HTTP 消息包括起始行、题头域和信息体 3 部分。

起始行通常是请求消息的首行，包含 3 个域：HTTP 方法、通用资源标识符和 HTTP 协议版本。尽管有几种 HTTP 方法可以从服务器中检索数据，但是最常用的方法只有 get 和 post 方法。例如，get 请求的请求行如下。

```
get/first.html HTTP/1.0
```

HTTP 请求的题头域可以没有或有多个。请求题头域允许客户端向服务器传递有关请求和客户端本身的一些附加信息。请求消息和响应消息的题头域是相同的。首先是题头域的名称，接着是冒号和值。如果对同一个题头域规定了多个值，则必须用逗号隔开。例如：

```
hosting:localhost:8080
```

```
accept-encoding:gzip,deflate
```

一旦服务器接收并处理了请求消息，它就必须向客户端返回一条响应消息。响应消息包含状态行、0 个或多个题头域，空行后是一个消息体。状态行包含响应消息所采用的 HTTP 协议版本，之后是响应状态码和状态描述，中间用空格隔开。响应状态码是 3 位数字，用于描述服务器的响应状态。

1yy（以 1 开头的 3 位数）：主要是实验性质的。

2yy：表明请求成功。例如，200 表明已成功取得了请求页面。

3yy：表明在请求成功之前应该采取进一步的行动。

4yy：表明浏览器无法满足请求。例如，404 表示请求的页面不存在。

5yy：表明服务器出现问题。例如，500 说明服务器内部发生错误。

### 1.3 动态网页技术介绍与比较

常用的动态网页技术有 ASP (ASP.NET)，PHP，CGI，JSP。在早期，动态网页主要采用 CGI (Common Gateway Interface 公用网关接口) 技术。CGI 可以用不同的语言编程，如 VB 和 C++ 等。然后将写好的程序放在服务器上运行，再将其运行结果通过服务器传送到客户端的浏览器上。虽然 CGI 技术发展成熟且功能强大，但编程复杂，效率低下，修改复杂。编写 CGI 页面最常用的语言是 Perl，文件后缀为 pl。

ASP 是 Active Server Page (动态服务器网页) 的简称。它将 Web 上的请求转入一个解释器中，在这个解释器中将所有的 ASP 的 Script 进行分析，再执行。ASP 可以使用 COM 组件，从而达到无限扩展。ASP 编程简单，容易学习。但 ASP 绝大部分在服务器上运行，导致安全性、稳定性、跨平台性受到影响。

PHP 是 Hypertext Preprocessor 的简称。它是一种 HTML 内嵌式的语言,混合了 C、Java、Perl 及 PHP 的新语法。它有比 CGI 更快的执行动态网页的速度,并且对多数数据库提供支持,如 SQL Server, MySQL, Sybase 等。

ASP.NET 的前身是 ASP。ASP 提出了新的不同于传统 CGI 的解决方法,能处理 Web 表单,并产生动态内容。通过特定的 HTML 标记,可以在网页中加入 JavaScript 程序代码,在服务器端执行。ASP.NET 是建立在微软 .NET 平台上的网络技术,它可以将页面表现和程序代码分离,并且可以使用多种语言,如 VB, NET, C# 等。

JSP 是使用 Java 语言的 Web 服务器技术,它也提供了在 HTML 代码中混合程序代码的能力。在 JSP 环境下,HTML 负责描述信息的显示样式,而程序代码则用来描述处理逻辑。JSP 是面向 Web 服务器的技术,客户端浏览器不需要任何附加软件的支持。在 JSP 下,代码被编译成 Servlet 并由 Java 虚拟机执行,这种编译操作仅在对 JSP 页面的第一次请求时发生。

## 习 题

### 一、单项选择题

以下有关 C/S 和 B/S 架构的说法错误的是 ( )。

- A. 在 C/S 架构(即客户端/服务器架构)中,有专门的数据库服务器,但客户端还要运行客户端应用程序,这也叫做胖客户端。
- B. 在 B/S 架构中,客户端在浏览器中只负责表示层逻辑的实现,业务逻辑和数据库都在服务器端运行。也就是说,应用程序部署在服务器端,客户端通过浏览器访问应用程序。
- C. 通常在 B/S 架构中,客户端发送 HTTP 请求消息给服务器,服务器将请求传递给 Web 应用程序,Web 应用程序处理请求,并把响应的 HTML 页面传回给客户端。
- D. Web 应用是基于 C/S 架构的,C/S 架构就是客户端/服务器架构。

### 二、多项选择题

1. 以下对 HTTP 状态码的含义描述正确的是 ( )。

- A. 200 表示请求成功。
- B. 400 表示服务器未发现与请求 URL(统一资源定位)匹配的内容。
- C. 404 表示由于语法错误而导致服务器无法理解请求消息。
- D. 500 表示服务器内部错误,无法处理请求。

2. MVC 模式的优势有哪些? ( )

- A. MVC 模式使应用程序的结构更加清晰,通过将代码按照层次划分为业务逻辑/数据层、用户界面和应用流程控制这 3 个层次,能够增强代码的稳定性。
- B. MVC 模式实现了很好的分布式组件架构。
- C. 对于项目开发而言,MVC 3 层的分离有利于在项目小组内按照小组成员各自的擅长方面进行分工,有利于 3 个部分并行开发,加快项目进度。
- D. MVC 模块功能的划分有利于在代码修改过程中进行模块的隔离,而不需要把具有不同功能的代码混杂在一起造成混乱。

## 第2章 环境变量的设置及开发实例

### 📖 知识点

- 如何搭建 Web 应用程序的开发环境
- 了解开发环境所必备的硬件和软件

### 🔊 难点

- 如何搭建 Web 应用程序的开发环境

### 🔧 掌握

- 以 Tomcat 服务器为中心配置 JSP 和 Servlet 的运行环境

### 任务引入

要学习 Web 应用程序的开发，首先需要搭建开发环境。所有程序都要在一定的环境下运行。本书的所有例子都是在 MyEclipse6.0+Tomcat6.0+JDK1.6 环境下调试运行的。其中，MyEclipse6.0 为开发平台，表示所有项目都在此平台上运行；Tomcat6.0 为 Java Web 服务器；JDK1.6 为 Java 虚拟机环境设置。

本章主要讲解 Tomcat6.0 的安装步骤、JDK1.6 的安装、环境变量的设置和开发实例。

## 2.1 环境变量的设置

要学习 Web 应用程序的开发，必须先搭建一个开发环境。本章以 Tomcat 服务器为中心详细介绍如何配置 JSP 和 Servlet 的运行环境。

### 2.1.1 Tomcat 的安装

Tomcat 服务器是一种 Servlet/JSP 容器。它可以用做小型独立服务器，用于测试 Servlet 和 JSP 页面。与所有 Apache 产品一样，Tomcat 是完全免费的。其安装步骤如下。

(1) 运行 jakarta-tomcat-6.0.30 按照提示安装。

(2) 在如图 2.1 所示窗口中，可以单击“Browse...”按钮选择安装路径，然后单击“Next>”按钮，得到如图 2.2 所示窗口。

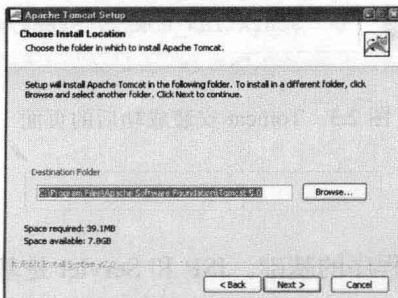


图 2.1 选择安装路径