第 3 章

零基础开始学习——快速认识 JSP

JSP(Java Server Pages)是由 Sun Microsystems 公司倡导、许多公司参与一起建立的一种动态网页技术标准。JSP 技术自诞生到现在，已经成为流行技术的一种，尤其是在开发电子商务类的网站方面。JSP 以其安全性高、支持多线程、跨平台等特性占领了 Web 开发的中、高层领域。为此，本章将开始认识 JSP 的入门知识。

本章要点(已掌握的在方框中打钩)

- ☐ 了解 JSP 概述
- ☐ 了解 JSP 的形成历史
- ☐ 熟悉 JSP 的优势
- ☐ 理解 JSP 运行机制
- ☐ 理解 JSP 的开发模式
- ☐ 掌握如何创建和运行 JSP 页面

3.1 JSP 概述

JSP(Java Server Pages)中文名叫 Java 服务器页面，其根本是一个简化的 Servlet 设计，它是一种动态网页技术标准。JSP 技术有点类似 ASP 技术，它是在传统的网页 HTML 文件(*.htm,*.html)中插入 Java 程序段和 JSP 标记，从而形成 JSP 文件，后缀名为(*.jsp)。用 JSP 开发的 Web 应用是跨平台的，既能在 Linux 下运行，也能在其他操作系统上运行。

JSP 实现了以<%, %>形式在 HTML 中插入 Java 代码。它是一种 Servlet，在服务器端执行，主要用于实现 Java Web 应用程序的用户界面部分。通常返回给客户端的就是一个 HTML 文本，因此客户端只要有浏览器就能浏览。

JSP 是一种动态页面技术，其主要目的是将表示逻辑从 Servlet 中分离出来。Java Servlet 是 JSP 的技术基础，而且大型的 Web 应用程序的开发需要 Java Servlet 和 JSP 配合才能完成。JSP 具备了 Java 技术的简单易用、完全的面向对象、具有平台无关性且安全可靠、主要面向因特网的所有特点。

3.2 JSP 形成历史

基于浏览器客户端的应用程序相比传统的基于客户端服务器的应用程序的优势在于：几乎没有限制的客户端访问和极其简化的应用程序部署和管理(要更新一个应用程序，管理人员只需要更改一个基于服务器的程序，而不是成千上万的安装在客户端的应用程序)。这样，软件工业正迅速地向基于浏览器客户端的多层次应用程序迈进。

这些快速增长的、基于 Web 的精巧应用程序要求开发技术上的改进。静态 HTML 对于显示相对静态的内容是不错的选择，新的挑战在于创建交互的、基于 Web 的应用程序。在这些程序中，页面的内容是基于用户的请求或者系统的状态，而不是预先定义的文字。

对于这个问题的一个早期解决方案是使用 CGI-BIN 接口：开发人员编写与接口相关的单独程序，以及基于 Web 的应用程序，后者通过 Web 服务器来调用前者。但这个方案有着严重的扩展性问题——每个新的 CGI 都要求在服务器上新增一个进程。如果多个用户并发地访问该程序，这些进程有可能会消耗掉该 Web 服务器所有的可用资源，并且系统性能降低到极其低下的地步。

某些 Web 服务器供应商已经尝试通过为其服务器提供插件和 API 来简化 Web 应用程序的开发。这些解决方案与特定的 Web 服务器相关，但不能解决跨平台操作的问题。例如，微软的 ASP 技术使得在 Web 页面上创建动态内容更加容易，但是也只能工作在微软的 IIS 和 Personal Web Server 上。

当然，还存在其他的解决方案。但是它们都不能使一个普通的页面设计者能够轻易地掌握。例如，Java Servlet 技术就可以使得用 Java 语言编写交互的应用程序的服务器端的代码变得容易，一个 Java Servlet 就是一个基于 Java 技术的运行在服务器端的程序(与 Applet 不同，后者运行在浏览器端)。开发人员需要编写出这样的 Servlet，以接收来自 Web 浏览器的

HTML 请求，动态地生成响应(可能要查询数据库来完成这项请求)，然后发送包含 HTML 或 XML 文档的响应到浏览器。

采用这种方法，整个网页必须都在 Java Servlet 中制作。如果开发人员或者 Web 管理人员想要调整页面显示，他们就不得不编辑并重新编译该 Java Servlet，即使该 Java Servlet 在逻辑上已经能够运行。采用这种方法，生成带有动态内容的页面仍然需要应用程序的开发技巧。

很显然，目前所需要的是一个业界范围内的创建动态内容页面的解决方案。这个方案将解决当前方案所解决不了的问题，例如：

- 能够在任何 Web 或应用程序服务器上运行；
- 将应用程序逻辑和页面显示分离；
- 能够快速地开发和测试；
- 简化开发基于 Web 的交互式应用程序的过程。

JSP 技术就是设计用来满足这些要求的解决方案。JSP 规范是 Web 服务器、应用服务器、交易系统以及开发工具供应商间广泛合作的结果。Sun Microsystems 开发出这个规范来整合及平衡已经存在的对 Java 编程环境(例如，Java Servlet 和 JavaBean)进行支持的技术和工具，其结果是产生了一种新的、开发基于 Web 应用程序的方法，给予使用基于组件应用逻辑的页面设计者以强大的功能。

所谓的 JSP 网页(*.jsp)．就是在传统的网页 HTML 文件(*.htm 或*.html)中加入 Java 程序片段(Sriptlet)和 JSP 标记(Tag)而构成的。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 Java Applet，甚至无 Frame。JSP 在动态网页的建设中有其强大而特别的功能。

在 Sun 正式发布 JSP 之后，这种新的 Web 应用开发技术很快引起了人们的关注。JSP 为创建高度动态的 Web 应用提供了一个独特的开发环境。JSP 使得我们能够分离页面的静态 HTML 和动态部分。HTML 可以用任何通常使用的 Web 制作工具编写，编写方式也和原来的一样；动态部分的代码放入特殊标记之内，大部分以“<%”开始，以“%>”结束。

3.3 JSP 的优势

JSP 技术是由 Servlet 技术发展起来的，自从有了 JSP 后，在 Java 服务器端编程中普遍采用的就是 JSP，而不是 Servlet。因为 JSP 在编写表示页面时远远比 Servlet 简单，并且不需要手工编译(由 Servlet 容器自动编译)，目前 Servlet 主要用做视图控制器、处理后台应用等。由于 JSP 构建在 Servlet 上，所以它有 Servlet 所有强大的功能。

在开发 JSP 规范的过程中，Sun 公司与许多主要的 Web 服务器、应用服务器和开发工具供应商积极进行合作，不断完善技术。

JSP 基于强大的 Java 语言，具有良好的伸缩性，与 Java Enterprise API 紧密地集成在一起，在网络数据库应用开发领域具有得天独厚的优势，基于 Java 平台构建网络程序已经被越

来越多的人认为是未来最有发展前途的技术。

从 JSP 这几年的发展来看，已经获得巨大的成功，它通过和 EJB 等 J2EE 组件进行集成，可以编写出处理具有大的伸缩性、高负载的企业级应用。JSP 技术在多个方面加速了动态 Web 页面的开发。

JSP 在跨平台、执行速度等特性上具有很大的技术优势，主要体现在以下方面。

(1) 将内容的生成和显示进行分离。

使用 JSP 技术，Web 页面开发人员可以使用 HTML 或者 XML 标识来设计和格式化最终页面。使用 JSP 标识或者小脚本来生成页面上的动态内容。生成内容的逻辑被封装在标识和 JavaBean 组件中，并且捆绑在脚本中，所有的脚本在服务器端运行。如果核心逻辑被封装在标识和 Bean 中，那么其他人，如 Web 管理人员和页面设计者，就能够编辑和使用 JSP 页面，而不影响内容的生成。

在服务器端，JSP 引擎解释 JSP 标识和小脚本，生成所请求的内容(例如，通过访问 JavaBean 组件，使用 JDBCTM 技术访问数据库或者包含文件)，并且将结果以 HTML 或者 XML 页面的形式发送回浏览器。这有助于作者保护自己的代码，而又保证任何基于 HTML 的 Web 浏览器的完全可用性。

(2) 生成可重用的组件。

绝大多数 JSP 页面信赖于可重用的、跨平台的组件(JavaBean 或者 Enterprise JavaBean 组件)来执行应用程序所要求的更为复杂的处理。开发人员能够共享和交换执行普通操作的组件，或者使得这些组件为更多的使用者或客户团体使用。

(3) 采用标识简化页面。

Web 页面开发人员不一定都是熟悉脚本语言的编程人员。JSP 技术封装了许多功能，这些功能是在易用的、与 JSP 相关的 XML 标识中进行动态内容生成时所需的。标准的 JSP 标识能够访问和实例化 JavaBean 组件、设置或者检索组件属性、下载 Applet，以及执行其他更难于编码或耗时的功能。通过开发定制标识库，JSP 技术是可扩展的。今后，第三方开发人员和其他人员可以为常用功能创建自己的标识库。这使得 Web 页面开发人员能够使用熟悉的工具和如同标识一样执行特定功能的构件来工作。

(4) JSP 能提供所有的 Servlet 功能。

与 Servlet 相比，JSP 能提供所有的 Servlet 功能，它比用 Println 书写和修改 HTML 更方便。此外，可以更明确地进行分工，Web 页面设计人员编写 HTML，只需要留出空间让 Servlet 程序员插入动态部分即可。

(5) 健壮的存储管理和安全性。

由于 JSP 页面的内置脚本语言是基于 Java 语言编写的，而且所有的 JSP 页面都被编译成为 Java Servlet，JSP 页面具有 Java 技术的所有优点，包括健壮的存储管理和安全性。

(6) 一次编写，随处运行。

作为 Java 平台的一部分，JSP 拥有 Java 编程语言“一次编写，随处运行”的特点。越来越多的供应商将 JSP 支持添加到其产品中，用户可以使用自己所选择的服务器和工具，但并不影响当前的应用。

(7) JSP 的平台适应性更广。

这是 JSP 相比于 ASP 的优越之处。几乎所有平台都支持 Java、JSP+JavaBean，它们可以

在任何平台下通行无阻。Windows NT 下的 IIS 通过一个插件就能支持 JSP，使用 JRUN 或者 ServletExec，著名的 Web 服务器 Apache 已经能够支持 JSP。由于 Apache 广泛应用在 Windows NT、UNIX 和 Linux 上，因此 JSP 有更广泛的运行平台。虽然现在 Windows NT 操作系统占了很大的市场份额，但是在服务器方面，UNIX 的优势仍然很大，而新崛起的 Linux 更是来势不小。从一个平台移植到另一个平台，JSP 和 JavaBean 甚至不用重新编译，因为 Java 字节码都是标识的字节码，与平台无关。

(8) Java 中连接数据库的技术是 JDBC(Java Database Connectivity)。

很多数据库系统都带有 JDBC 驱动程序，Java 程序通过 JDBC 驱动程序与数据库相连，执行查询、提取数据等操作。Sun 公司还开发了 JDBC-ODBC Bridge，用此技术，Java 程序就可以访问带有 ODBC 驱动程序的数据库，目前大多数数据库系统都带有 ODBC 驱动程序，所以 Java 程序能访问诸如 Oracle、Sybase、Microsoft SQL Server 和 Microsoft Access 等类型的数据库。

(9) 简单易学。

随着 JSP 中标签语言的出现，即使不懂 Java 的程序员也能编写出功能完善的 JSP 应用。

3.4 JSP 运行机制

JSP 文件在用户第一次请求时，会被编译成 Servlet，再由这个 Servlet 处理用户的请求，如图 3-1 所示。

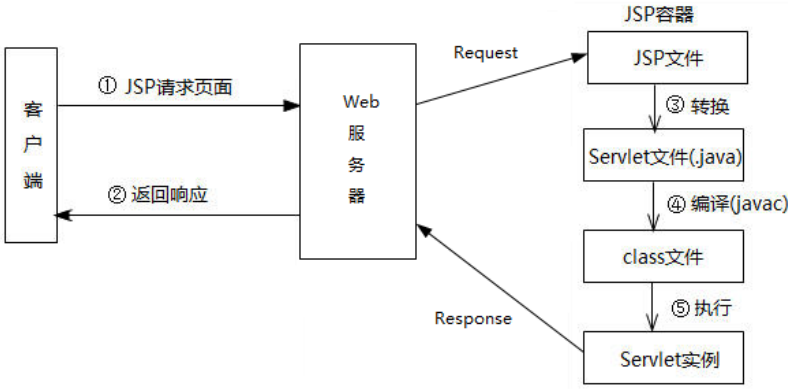

图 3-1 JSP 运行机制

JSP 容器管理 JSP 页面生命周期分为两个阶段：转换阶段和执行阶段。当有一个对 JSP 页面的客户请求到来时，JSP 容器将 JSP 页面转换为 Servlet 源文件，然后调用 javac 工具编译源文件，生成字节码文件，这是转换阶段。接下来，Servlet 容器加载转换后的 Servlet 类，实例化一个对象处理客户端的请求，请求处理完成后响应对象被 JSP 容器接收，容器将 HTML 格式的响应信息发送给客户端，这是执行阶段。

3.5 JSP 开发的两种模式

使用 JSP 技术开发 Web 应用程序，有两种架构模式可供选择，通常称为模式 1 和模式 2。下面将对这两种模式进行介绍，并讲解两种架构模式各自的优缺点及其应用场合。

3.5.1 JSP+JavaBean 模式

模式 1 使用 JSP+JavaBean 技术将页面显示和业务逻辑处理分开。JSP 实现页面的显示，JavaBean 对象用来保存数据和实现商业逻辑。在模式 1 中，JSP 页面独自响应请求并将处理结果返回给客户，所有的数据通过 JavaBean 来处理，并实现页面的显示。模式 1 的结构如图 3-2 所示。

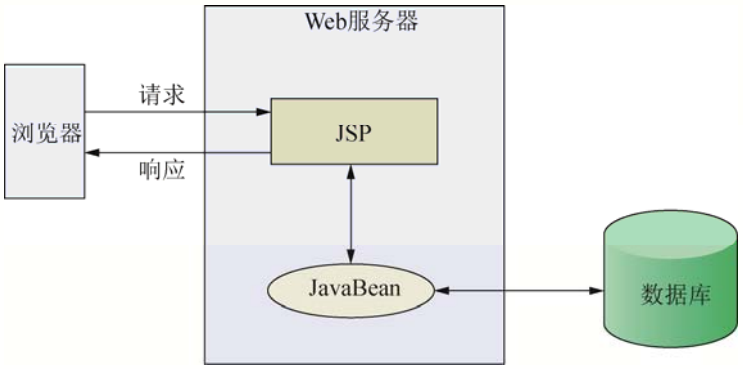

图 3-2 模式 1 的 JSP 架构

3.5.2 JSP+JavaBean+Servlet 模式

在模式 1 中，JSP 页面嵌入了流程控制代码和部分的逻辑处理代码，我们可以将这部分代码提取出来，放到一个单独的角色中，这个角色就是控制器角色，而这样的 Web 架构就是所谓的模式 2。它符合 MVC 架构模式。

在 MVC 架构中，一个应用被分成三个部分，下面分别进行简要介绍。

(1) 模型代表应用程序的数据以及用于访问控制和修改这些数据的业务规则。当模型发生改变时，它会通知视图，并为视图提供查询模型相关状态的能力。同时，它也为控制器提供访问封装在模型内部的应用程序功能的能力。

(2) 视图用来组织模型的内容。它从模型那里获得数据并指定这些数据如何表现。当模型变化时，视图负责维护数据表现的一致性。同时视图会将用户的请求通知控制器。

(3) 控制器定义了应用程序的行为。它负责对来自视图的用户请求进行解释，并把这些请求映射成相应的行为，这些行为由模型负责实现。在独立运行的 GUI 客户端，用户的请求可能是一些鼠标单击或是菜单选择操作。在一个 Web 应用程序中，它们的表现形式可以是一些来自客户端的 GET 或 POST 的 HTTP 请求。模型所实现的行为包括处理业务和修改模型的状态。根据用户请求和模型行为的结果，控制器选择一个视图作为对用户请求的响应。图 3-3

描述了在 MVC 应用程序中模型、视图、控制器三部分的关系。

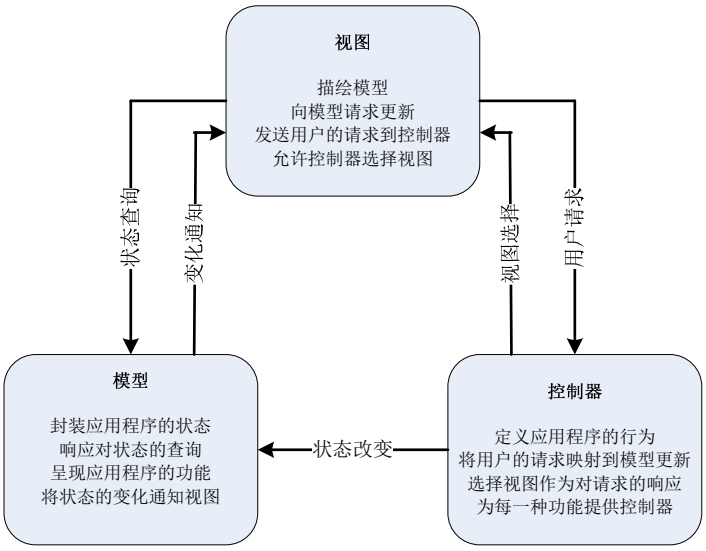

图 3-3　MVC 模型、视图、控制器的关系

在模型 2 中，控制器的角色由 Servlet 来实现，视图的角色由 JSP 页面来实现，模型的角色由 JavaBean 来实现。模型 2 的架构如图 3-4 所示。

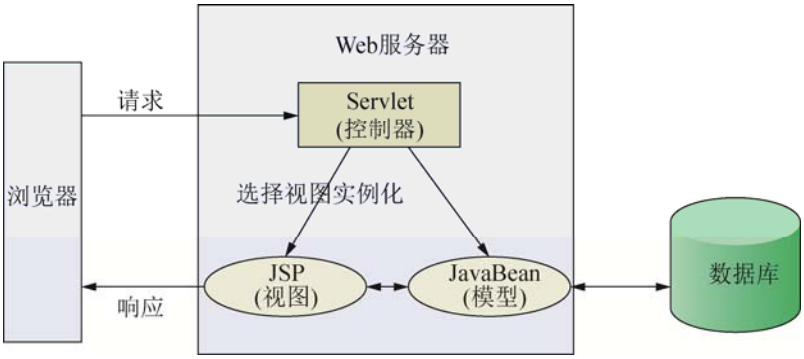

图 3-4　模型 2 架构示意图

Servlet 充当控制器的角色，它接收请求，并且根据请求信息将它们分发给适当的 JSP 页面来产生响应。Servlet 控制器还根据 JSP 视图的需求生成 JavaBean 的实例并输出经 JSP 环境。JSP 视图可以通过直接调用 JavaBean 实例的方法或使用<jsp:useBean>和<jsp:getProperty>动作元素来得到 JavaBean 中的数据。

3.6　第一个 JSP 页面

在介绍 JSP 的基本语法前，首先来认识一下 JSP 页面及其创建，并在 JSP 页面中显示“你好，这是 JSP 页面”信息。具体操作步骤如下。

step 01 在 MyEclipse 的 Package 窗口中，右击空白处，在弹出的快捷菜单中选择 New→Web Project 命令，打开 New Web Project 对话框，输入要创建的 Web 项目名“MyWeb”，选择 Java version 是 1.8 版本，如图 3-5 所示。

step 02 单击 Finish 按钮，创建完成。在 MyWeb 项目中，右击 WebRoot 文件夹，在弹出的快捷菜单中选择 New→JSP(Advanced Templates)命令，如图 3-6 所示。

step 03 打开 Create a new JSP page 对话框，在 File Name 文本框中修改 JSP 页面的名称，如图 3-7 所示。

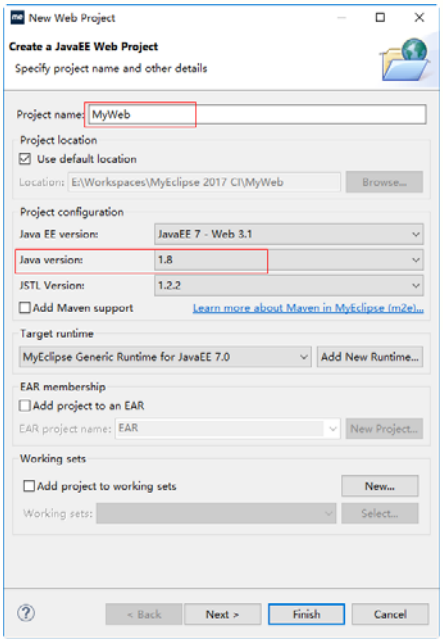

图 3-5　创建 Web 项目

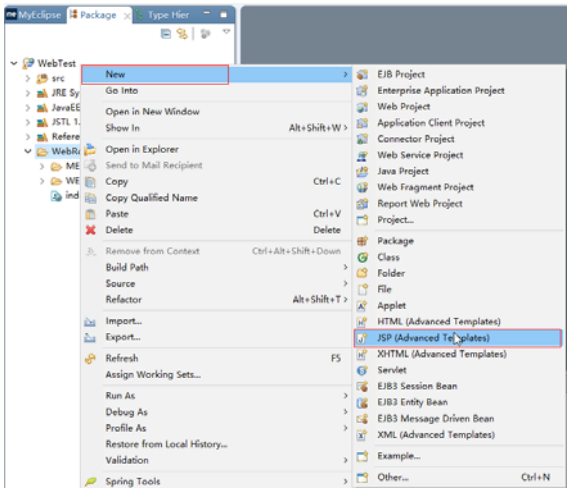

图 3-6　创建 JSP 页面

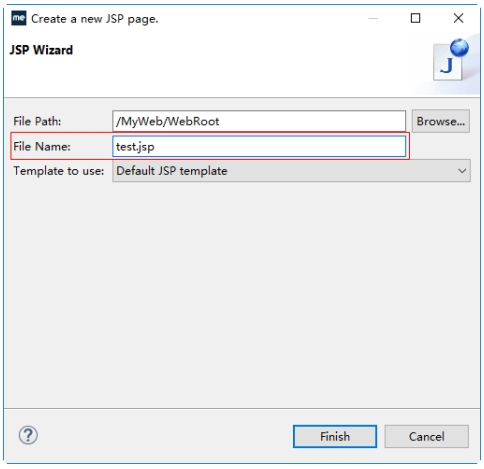

图 3-7　修改 JSP 文件名

step 04 单击 Finish 按钮，创建完成。在打开的 JSP 页面中，修改<body></body>之间的

内容为要显示的信息。JSP 页面代码具体如下：

```
<%@ page language="java" import="java.util.*" pageEncoding="ISO-8859-1"%>
<%
String path = request.getContextPath();
String basePath = request.getScheme()+"://"+request.getServerName()+":
"+request.getServerPort()+path+"/";
%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <base href="<%=basePath%>">
    <title>第一个 JSP 程序</title>
    <meta http-equiv="pragma" content="no-cache">
    <meta http-equiv="cache-control" content="no-cache">
    <meta http-equiv="expires" content="0">
    <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
    <meta http-equiv="description" content="This is my page">
    <!--
    <link rel="stylesheet" type="text/css" href="styles.css">
    -->
  </head>
  <body>
        你好，这是 JSP 页面。 <br>
  </body>
</html>
```

step 05 修改完 JSP 页面内容，保存文件时出现编码错误提示，如图 3-8 所示。单击 OK 按钮，修改 JSP 页面中第一句代码 pageEncoding 的值为“UTF-8”。

step 06 将 MyWeb 项目部署到 Tomcat 上，运行项目。在浏览器中输入运行页面的地址“http://127.0.0.1:8888/MyWeb/test.jsp”，运行效果如图 3-9 所示。

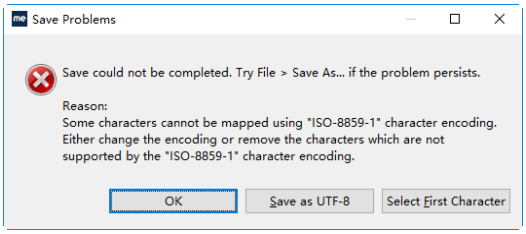

图 3-8 编码错误

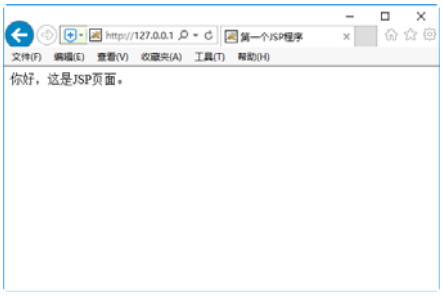

图 3-9 运行效果

3.7 大 神 解 惑

小白：JSP 和 Servlet 的区别是什么？

大神：JSP 经编译后生成 Servlet，JSP 的本质就是 Servlet。JVM 只能识别 Java 类，不能识别 JSP。Web 容器将 JSP 的代码编译成 JVM 能够识别的 Java 类。

JSP 一般用于页面显示，而 Servlet 一般用于逻辑控制。Servlet 中没有内置对象。JSP 中的内置对象必须通过 HttpServletRequest 对象、HttpServletResponse 对象及 HttpServlet 对象获得。

JSP 是 Servlet 的一种简化，使用 JSP 只需要完成程序员要输出到客户端的内容。JSP 中的 Java 脚本如何镶嵌到一个类中，由 JSP 容器完成。而 Servlet 是个完整的 Java 类，这个类的 Service()方法用于生成对客户端的响应。

3.8 跟我学上机

练习 1：掌握 JSP 的运行机制。

练习 2：创建 Web 项目，在项目中创建 test.jsp 页面。结合 HTML 语言在 JSP 页面上显示一个 3 行 4 列的表格。