第 3 章

JSP 基本语法

本章将 JSP 所包含的各类元素，如声明、注释、程序片、表达式、JSP 指令标记(Directive Elements)、JSP 动作标记(Action Elements)等主要方面来说明 JSP 的基本语法结构。

3.1 JSP 文件的构成

3.1.1 知识要点

在传统的网页即 HTML 文件中插入 Java 动态元素、JSP 标记等，就形成了 JSP 页面。通常，一个 JSP 文件包括如下组成部分：

(1) HTML 标记，这在第 1 章中已有介绍。

(2) Java 动态元素，包括变量和方法的定义、Java 程序片和 Java 表达式。

(3) 注释，包括 HTML 注释和 JSP 特有的注释。HTML 注释格式为：<! --注释内容-->，JSP 特有的注释格式为：

```
<%--注释内容--%>
```

(4) JSP 标记，包括 JSP 指令标记和 JSP 动作标记。

3.1.2 技能操作

识别 JSP 文件中的构成要素，具体任务如下：

- 阅读下面的代码，识别其中的 JSP 构成要素。
- 分别在记事本和 Eclipse 中输入下面的代码，并对 JSP 页面进行调试运行。

1. 阅读下面的代码，识别其中的 JSP 构成要素。

代码模板 eg3_1.jsp 如下：

```
<%@ page language="java" contentType="text/html; charset=GBK"
pageEncoding="GBK"%><!-- JSP 指令标记 -->
<%--下面是变量的声名和方法的定义--%>
<%!
    int i=0; //变量声明
    double mul(double a,double b){ //方法定义
        return a * b;
```

```
    }
%>
<html>
    <head>
        <title>JSP 文件构成</title>
    </head>
     <body bgcolor="cyan"><!--HTML 标记 -->
    <%i++;
    double mulitiple=mul(5.5,6.0); //Java 程序片
   %>
  第
  <%=i %>
  次计算,结果为
  <%=mulitiple %><!--Java 表达式 -->
 </body>
</html>
```

2. 分别在记事本和 Eclipse 中输入上面 eg3_1.JSP 中的代码,并对 JSP 页面进行调试运行。

1) 使用记事本和浏览器完成

按照 2.1 节介绍的方法,打开记事本编写代码,然后将代码保存到自己设定的 Web 服务目录(如 ch03)中,最后在浏览器地址栏输入相应的地址访问该 JSP 页面,完成效果如图 3-1 所示。

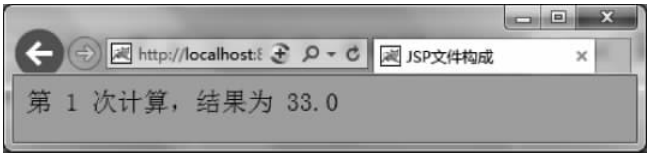

图 3-1　JSP 文件构成页面效果

2) 使用 Eclipse 完成

按照 2.2 节介绍的方法,在 Eclipse 中建立 Web 项目和 JSP 文件,然后编写并保存代码,最后在 Eclipse 中完成运行,效果同图 3-1。

3.1.3　拓展训练

1. 在 JSP 文件中,方法的定义使用(　　)符号。

A. <% %>　　B. <%! %>　　C. <%@ %>　　D. <%@ %>

2. 在 JSP 文件中,Java 程序片使用(　　)符号。

A. <% %>　　B. <%! %>　　C. <%@ %>　　D. <%@ %>

3. 在 Tomcat 服务器下新建服务目录 ch03,打开记事本,输入如下的 Test3_1.jsp 程

序，并启动 Tomcat 服务器，在地址栏中输入 http://localhost:8080/ch03/Test3_1.jsp 访问 JSP 页面。

代码 Test3_1.jsp 如下：

```
<%@ page contentType="text/html;charset=gb2312" %>
<html>
    <body bgcolor="cyan">
    <h3>求两个数之和</h3>
    <!--下面是求两个数之和的方法的声明过程-->
    <%!
        double sum(double a,double b){
            return a+b;
        }
     %>
    <%/*下面是方法的调用过程,并且注释的内容在客户端看不到*/%>
    <%= sum(3.3,4.5)%>
    </body>
</html>
```

3.2 变量和方法的定义

3.2.1 知识要点

1. 变量的定义

JSP 文件中，可以在“<%!”和“%>”符号之间定义成员变量，变量类型可以是 Java 语言所支持的任何数据类型。例如下面的例子中定义了两个普通型变量 i 和 j 初值都为 0，定义了一个双精度实型变量 k 初值为 0.0，定义了一个字符串对象 s 其实体为“hello”。

```
<%!
   int i=0,j=0;
   double k=0.0;
   String s="hello";
%>
```

变量的定义在书写位置上没有固定要求，但习惯上总是将其放在 JSP 文件的前端。成员变量一旦定义就在整个 JSP 页面内都有效，且为多个用户所共享。也就是说，任何用户操作该变量都会对其他用户产生影响，原理如图 3-2 所示。例如 3.1.2 节中的 JSP 文件 eg3_1.jsp，第一个用户访问页面时显示效果为“第 1 次计算，结果为 33.0”，第二个用户再访问页面时显示效果则为“第 2 次计算，结果为 33.0”，以此类推；原因就在于成员变量 i 为多个用户所共享。

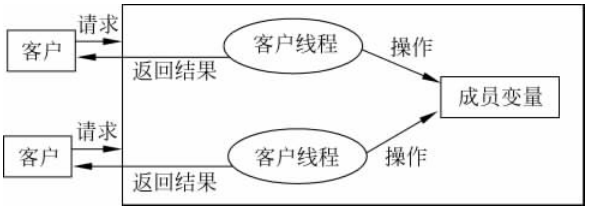

图 3-2　共享成员变量

2. 方法的定义

JSP 文件中，也可以在“<%!”和“%>”符号之间定义方法，这些方法将在 Java 程序片中被调用以完成相应的功能，关于 Java 程序片的具体内容会在下一节中详细介绍。需要强调的是，这些方法在整个 JSP 页面内都有效，但方法内部定义的变量为局部变量只在该方法内部有效。例如下面的例子就是定义了一个名称为 mul 的方法，用于求两个实数的乘积，返回值类型为双精度实型，其中的变量 a 和 b 为局部变量，只在该方法内部有效，离开了这个方法，便不能再被访问。

```
<%!
    double mul(double a,double b){
        return a * b;
%>
```

3.2.2　技能操作

编写一个 JSP 页面文件 eg3_2.jsp，要求在该页面文件中定义四个成员变量：result1、result2、result3 和 result4(初值均为 0)；然后定义四个方法，分别为 add(求两个数的和)、sub(求两个数的差)、mul(求两个数的积)和 div(求两个数的商)。另外，还要求在页面文件中编写一段 Java 程序片，在 Java 程序片中调用上述四个方法以实现两个数的加、减、乘、除运算，并将最终结果在页面上显示出来。

代码模板 eg3_2.jsp 如下：

```
<%@ page language="java" contentType="text/html; charset=GBK"
    pageEncoding="GBK"%>
<html>
<head>
<title>变量和方法的定义</title>
</head>
<%!
    double result1,result2,result3,result4;
    double add(double a, double b){
```

```
    double c=a+b;
    return c;
    }
  double sub(double a, double b){
    double c=a-b;
    return c;
    }
  double mul(double a, double b){
    double c=a*b;
    return c;
    }
  double div(double a, double b){
        double c;
        if(b!=0) {
            c=a/b;
              return c;
              }
        else return 0;
    }
%>
<body bgcolor="cyan">
<%
     result1=add(3.5,1.2);
     result2=sub(3.5,1.2);
     result3=mul(3.5,1.2);
     result4=div(3.5,1.2);
     out.print("两个数的和是:"+result1+",");
     out.print("两个数的差是:"+result2+",");
     out.print("两个数的积是:"+result3+",");
     out.print("两个数的商是:"+result4+".");
%>
</body>
</html>
```

运行效果如图 3-3 所示。

图 3-3　变量和方法的定义页面效果

3.2.3　拓展训练

1. 编写 JSP 页面，完成显示 1 * 2 * 3 * … * 10 乘积的功能，最终效果如图 3-4 所示。

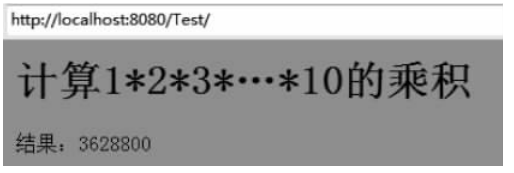

图 3-4　累乘页面效果

要求将如下代码模板补充完整，并进行调试运行。

```
<%@ page contentType="text/html;charset=GB2312"%>
    <html>
    <head>
      <title>练习 1</title>
    </head>
    <body bgcolor="gray">
        【代码段 1】//定义长整型变量 m,初值为 1;定义整型变量 i,初值为 1
      <h1>计算 1 * 2 * 3 * … * 10 的乘积</h1>
      <%
        while(i<=10){
            【代码段 2】//给出累乘的算法
         }
      %>
      结果: <%=m%>
    </body>
</html>
```

2. 编写 JSP 页面，完成求和及求平均值的功能，并在页面中显示，最终效果如图 3-5 所示。

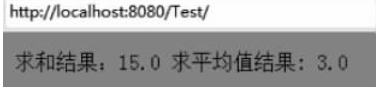

图 3-5　求和及求平均值页面效果

要求将如下代码模板补充完整，并进行调试运行。

```
<%@ page contentType="text/html;charset=GB2312"%>
    <html>
    <head>
      <title>练习 2</title>
    </head>
    <body bgcolor="gray">
    <%!
     float sum(float[] array){
        float sum=0.0f;
        for(int i=0;i<array.length;i++)
```

```
        {
            sum+=array[i];
        }
        return sum;
      }
      【代码段 1】//定义求平均值的方法
    %>
    <%
      float[] a={1,2,3,4,5};
      float s=sum(a);
      float result=aver(s,a.length);
     %>
      求和结果：<%=s%>
      求平均值结果：<%=result%>
    </body>
</html>
```

3.3 Java 程序片

3.3.1 知识要点

JSP 文件中，可以在“<%”和“%>”符号之间插入 Java 程序代码，使得 JSP 页面的功能更加完备，这些程序代码在 JSP 页面中被称为 Java 程序片或 Java 程序段。一个 JSP 页面可以有很多个 Java 程序片，这些程序片将被 JSP 引擎（即 Tomcat 服务器）按顺序执行。在程序片中声明的变量为局部变量，具体的作用范围与其声明位置有关，即程序片中的局部变量从声明之处起，在 JSP 页面后续的所有 Java 程序片及 Java 表达式内部都有效，原理如图 3-6 所示。

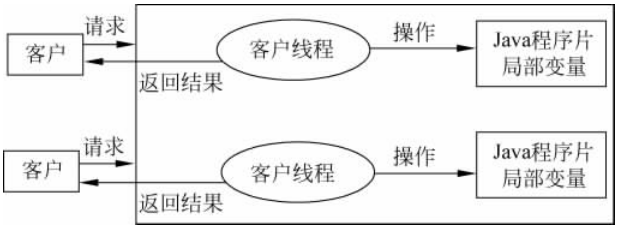

图 3-6 Java 程序片局部变量

3.3.2 技能操作

编写程序产生 1～7 之中的任意一个随机数，判断随机数是几，并相应地输出星期几。具体步骤如下：

- 用 JSP 声明方式产生一个 1～7 的随机数方法。
- 用 JSP 程序片判断产生的随机数是几。
- 判断以后在页面中相应地显示出星期几。

打开记事本，输入下面程序，并保存到相应的 Web 服务目录。

代码模板 eg3_3.jsp 如下：

```
<%@ page language="java" import="java.util.*" pageEncoding="GB2312"%>
<html>
  <body bgcolor="cyan">
  <h1>
  <!--声明 random 方面，产生一个 1 到 7 的随机数 -->
  <%!
    int random(){
     int i;
     Random r = new Random();
     i=r.nextInt(7)+1;
     return i;
  }
  %>
 <!--下面程序是对 random 方法的调用，并判断输出星期几 -->
  <%
   int i = random();
   out.print("本次访问产生的随机数是:"+i+"<br>");
   switch (i){
       case 1:
              out.print("星期一");
              break;
       case 2:
              out.print("星期二");
              break;
       case 3:
              out.print("星期三");
              break;
       case 4:
              out.print("星期四");
              break;
       case 5:
              out.print("星期五");
              break;
       case 6:
              out.print("星期六");
              break;
       case 7:
              out.print("星期日");
```

```
                break;
        }
        %>
        </h1>
    </body>
</html>
```

运行效果如图 3-7 所示。

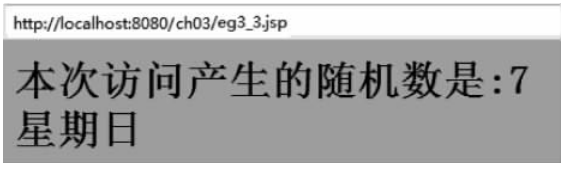

图 3-7　显示星期页面效果

3.3.3　拓展训练

1. 声明一个求梯形面积的方法，给定上底、下底和高求梯形面积，最终运行效果如图 3-8 所示。

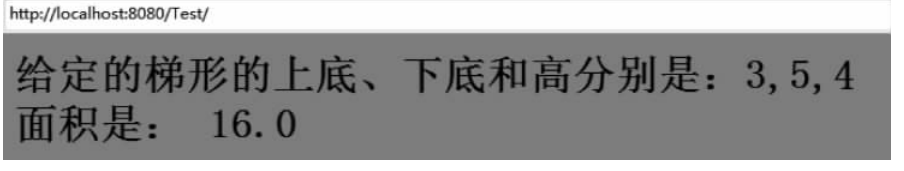

图 3-8　求梯形面积页面效果

要求将如下代码模板补充完整，并进行调试运行。

```
<%@ page contentType="text/html;charset=gb2312" %>
<html>
<body bgcolor="gray">
    <h1>
        <%!
        double area;
        double getArea(double a,double b,double c){
            【代码段 1】//求梯形的面积公式
            return area;
        }
        %>
    给定的梯形的上底、下底和高分别是: 3,5,4
    <br>
    面积是:
```

```
    <%=【代码段 2】//获得梯形面积
    %>
    </h1>
  </body>
  </html>
```

2. 模仿上面的例题完成求三角形面积的功能(注意：前提是给定的三条边能够构成三角形)。

3. 模仿上面的例题完成求圆形面积的功能。

3.4　Java 表达式

3.4.1　知识要点

JSP 文件中，可以在“<%=”和“%>”符号之间插入 Java 表达式，这个表达式必须拥有确定的值，其值的计算由 Web 服务器完成，计算结果会以字符串的形式发送到客户端浏览器进行显示。需要注意的是，“<%=”和“%>”符号之间不能插入语句只能插入表达式，而且“<%=”是一个完整的符号，其间不能有空格，“%>”符号也是如此。

3.4.2　技能操作

利用 Java 表达式在 JSP 页面中输出相应的内容。具体任务如下：

- 借助 Math 类中提供的方法，输出正弦值、输出立方值、输出平方根值；
- 输出算术表达式的值、输出逻辑表达式的值。

代码模板 eg3_4.jsp 如下：

```
<%@ page language="java" contenttype="text/html; charset=gbk"
    pageencoding="gbk"%>
<html>
<head>
    <title>java 表达式</title>
</head>
<body bgcolor="cyan">
    <p>sin(3.14/6)等于
      <%=Math.sin(3.14/6)%>
    <p>8 的立方是:
      <%=Math.pow(8,3)%>
    <p>196 的平方根等于
      <%=Math.sqrt(196)%>
    <p>123456 乘以 2 等于
      <%=123456 * 2%>
    <p>10000 大于 9999 吗?答案:
```

```
        <%=10000>9999%>
</body>
</html>
```

运行效果如图 3-9 所示。

图 3-9 Java 表达式页面效果

3.4.3 拓展训练

1. 利用 Java 程序片和 Java 表达式，在 JSP 页面中输出一个 10 行 10 列的表格，每一个单元格中显示行、列坐标值，最终运行效果如图 3-10 所示。

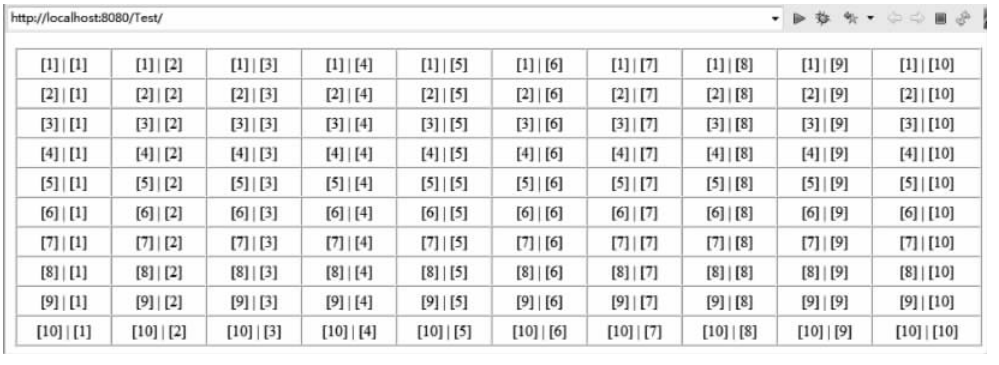

图 3-10 表格页面效果

要求编写、调试、运行下面给出的模板代码，体会 html 语言和 Java 程序片以及 Java 表达式相互之间的协作关系。

```
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
<html>
    <head>
      <title>输出 10 行 10 列的表格</title>
    </head>
    <body>
```

```
        <table cellpadding="0" cellspacing="0" border="1" width="100%">
        <%
          int rows = 10;          // 多少行
          int cols = 10;          // 多少列
          for(int i = 0; i < rows; i++ ){
      %>
       <tr align="center" height="30">
      <%
          for(int j = 0; j < cols; j++ ){
      %>
        <td>[<%=i+1 %>] | [<%=j+1 %>]</td>
      <%
          }
        }
      %>
    </table>
      </body>
</html>
```

2. 综合运用变量的定义、Java 程序片和 Java 表达式等所学内容，完成在 JSP 页面中输出九九乘法表的功能，最终效果如图 3-11 所示。

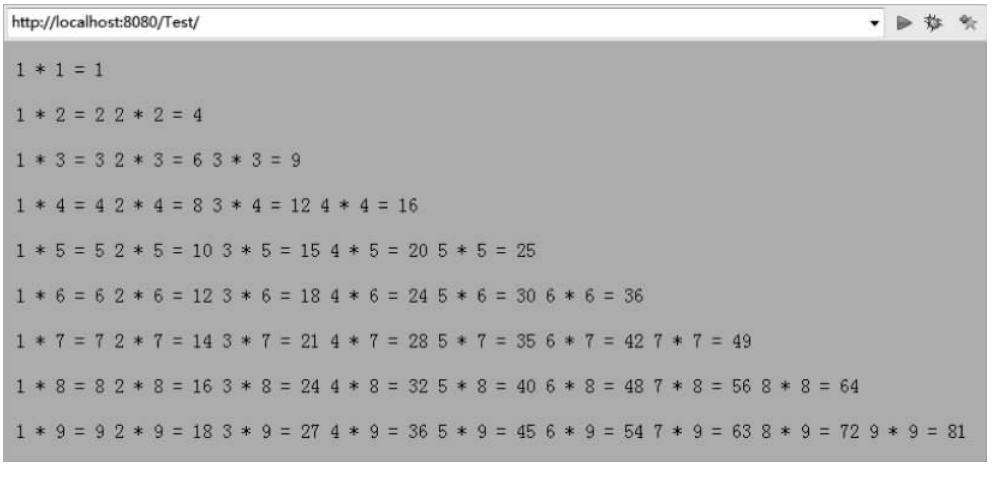

图 3-11　九九乘法表页面效果

要求将如下代码模板补充完整，并进行调试运行。

```
<%@ page contentType="text/html;charset=gb2312" %>
<html>
【代码段 1】//定义整型变量 i 和 j
<body bgcolor="gray">
  <%
```

```
        for(i=1;i<=9;i++){
            for(j=1;j<=i;j++){
    %>
    【代码段 2】//输出乘法表内容
      <%}%>
      <p>
    <%}%>
</body>
</html>
```

3.5 JSP 指令标记

为了更好地完成 Web 页面功能，JSP 引入了指令标记。本节将介绍常用的两个指令标记：page 指令标记和 include 指令标记。

3.5.1 知识要点

1. page 指令标记

page 指令标记用来定义 JSP 页面中的一些属性及其属性值。具体包括 import 属性、contentType 属性、pageEncoding 属性、language 属性、session 属性、isELIgnored 属性（只限 JSP 2.0）、buffer 属性、autoFlush 属性、info 属性、errorPage 属性、isErrorPage 属性、isThreadSafe 属性、extends 属性。在使用时，采用如下格式：

```
<%@page 属性 1="属性 1 的值"属性 2="属性 2 的值"…属性 n="属性 n 的值" %>
```

或者

```
<%@page 属性 1="属性 1 的值" %>
<%@page 属性 2="属性 2 的值" %>
⋮
<%@page 属性 n="属性 n 的值" %>
```

在本书中只介绍 import、contentType、pageEncoding 三个常用属性。

1）import 属性

使用 page 指令标记的 import 属性可以为 JSP 页面指定应该引入的包及包中的类，以便 JSP 页面在变量和方法的定义、Java 程序片、Java 表达式中正常使用相应包中的类。

在同一个 JSP 文件中，page 指令标记的 import 属性可以多次出现，也就是说，可以指定多个 import 属性值，导入多个包中的多个类。例如，下面的指令：

```
<%@ page import="java.util.*, china.dalian.*" %>
```

表示 java.util 包和 china.dalian 包中的所有类在使用时无须再给出明确的包标识符。

另外，import 属性的书写位置没有特定要求，但通常情况下，习惯将 import 语句放在 JSP 文件顶部附近或是放在相应的包首次使用之前。

2）contentType 属性

使用 page 指令标记的 contentType 属性可以用来设置 contentType 响应报头，指明即将发送到客户端的文档的 MIME 类型（MIME 类型是设定某种文件用匹配的一种应用程序来打开的方式类型）和 JSP 页面字符的编码。例如，如果希望客户端浏览器启用 HTML 解析器来解析执行所接收到的信息，就可以按照如下方式设置 contentType 属性值：

```
<%@ page contentType="text/html;charset=GB2312" %>
```

如果希望客户端浏览器启用本地的 MS-Excel 应用程序来解析执行所接收的信息，则可以按照如下方式设置 contentType 属性值：

```
<%@ page contentType="application/vnd.ms-excel" %>
```

如果不使用 page 指令标记为 contentType 指定属性值，那么 contentType 属性就取默认值，即"text/html;charset=ISO-8859-1"。

与 import 属性不同，在 JSP 文件中，page 指令标记只能为 contentType 属性指定一个值。下面的用法是错误的：

```
<%@ page contentType="application/vnd.ms-excel" %>
<%@ page contentType="text/html;charset=GB2312" %>
```

可以利用 page 指令标记为 contentType 属性指定的 MIME 类型有 text/html（超文本标记语言文本）、text/plain（普通文本）、application/rtf（rtf 文本）、image/gif（gif 图形）、image/jpeg（jpeg 图形）、audio/basic（au 声音文件）、audio/midi（midi 音乐文件）、audio/x-pn-realaudio（realaudio 音乐文件）、video/mpeg（mpeg 文件）、video/x-msvideo（avi 文件）、application/x-gzip（gzip 文件）、application/x-tar（tar 文件）、application/vnd.ms-excel（excel 文件）、application/msword（word 文件）。

3）pageEncoding 属性

如前所述，如果希望同时设置所接收到信息的 MIME 类型和 JSP 页面字符编码，可以使用 contentType 属性来完成。但是，如果只想更改字符集，使用 pageEncoding 属性更为简单。例如，中文 JSP 页面可以使用下面的语句：

```
<%@ page pageEncoding="GBK" %>
```

来设置。

关于前面提到的字符集，经常使用的有如下几种：

（1）ASCII。

ASCII（American Standard Code for Information Interchange，美国信息互换标准代

码),是基于常用英文字符的一套编码。

(2) ISO-8859-1。

ISO-8859-1 编码通常叫做 Latin-1,除收录 ASCII 字符外,还增加了其他一些语言和地区需要的字符。该编码是 Tomcat 服务器默认采用的字符编码。

(3) GB 2312。

GB 2312 码是中华人民共和国国家标准汉字信息交换用编码,简称国标码,是由国家标准总局发布的关于汉字的编码,通行于中国大陆和新加坡。

(4) GBK。

GBK 编码规范,除了完全兼容 GB 2312,还对繁体中文和一些不常用的字符进行了编码。GBK 是现阶段 Windows 和其他一些中文操作系统的默认字符集。

(5) Unicode。

Unicode 为统一的字符编码标准集,为地球上几乎所有地区每种语言中的每个字符设定了统一并且唯一的编码,以满足跨语言、跨平台进行文本转换、处理的要求。

(6) UTF-8。

UTF-8 是 Unicode 的一种变长字符编码。用在网页上可以在同一页面显示中文和其他语言。当处理包含多国文字的信息页面时一般选择用 UTF-8。

2. include 指令标记

网站中的多个 JSP 页面有时需要显示同样的信息,如该网站的 Logo 或是导航等,为了便于网站的管理与维护,通常在这些 JSP 页面的适当位置嵌入一个相同的文件以完成相应的功能。include 指令标记的用途就在于此,可以借助该标记在当前 JSP 页面中整体嵌入另一个文件。语法格式为:

```
<%@ include file="文件的地址" %>
```

其中被嵌入的文件必须是可访问和可使用的。

3.5.2 技能操作

使用 JSP 指令标记,完成相应的功能。具体任务如下:

- 利用 page 指令标记的 import 属性,导入 java.util 包中的类,完成在当前页面显示系统时间的功能。
- 利用 page 指令标记的 contentType 属性,指定客户端浏览器启用本地的 Microsoft Word 应用程序来解析执行所接收的信息。
- 利用 include 指令标记,在同一网站的不同页面顶端显示相同的 Logo 图片。

1. 利用 page 指令标记的 import 属性,导入 java.util 包中的类,完成在当前页面显示系统时间的功能。

代码模板 eg3_5.jsp 如下:

```
<%@ page language="java" contentType="text/html; charset=GBK"%>
<%@ page import="java.util.*"%>
<html>
<head>
    <title>JSP 指令标记</title>
</head>
<%!
 String s=null ;
%>
<body bgcolor="cyan">
    当前的日期和时间是:
    <%
     Date date=new Date();
     s = date.toString();
    %>
    <%= s%>
</body>
</html>
```

运行效果如图 3-12 所示。

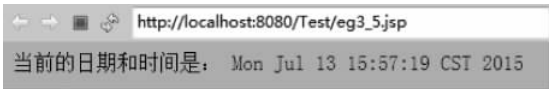

图 3-12　显示日期时间页面效果

2. 利用 page 指令标记的 contentType 属性，指定客户端浏览器启用本地的 Microsoft Word 应用程序来解析执行所接收的信息。

代码模板 eg3_6.jsp 如下：

```
<%@ page language="java" contentType="application/msword; charset=GBK"%>
<html>
<head>
    <title>JSP 指令标记</title>
</head>
<body bgcolor="cyan">
  <P>启用 Microsoft Word 应用程序处理所接收到的信息.
  <input type="text" size="12">
</body>
</html>
```

运行上述页面时会弹出如图 3-13 所示的“文件下载”对话框，单击“保存”按钮弹出如图 3-14 所示的对话框，选择相应的路径保存后，就会启用本地的 Microsoft Word 应用程序来显示当前页面内容，如图 3-15 所示。

图 3-13 “文件下载”对话框

图 3-14 选择文件保存路径

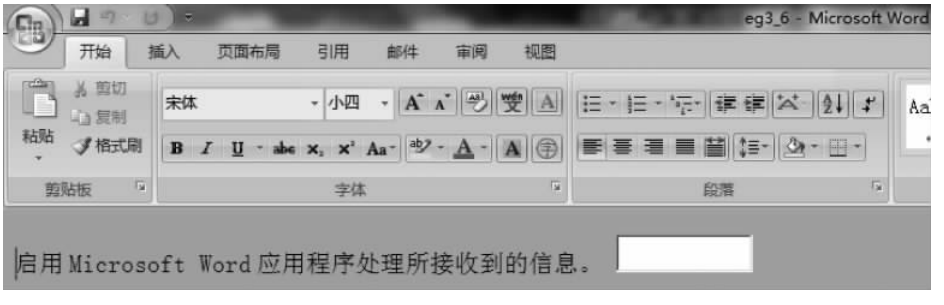

图 3-15 Microsoft Word 解析页面内容

3. 利用 include 指令标记，在同一网站的不同页面顶端显示相同的 logo 图片。

代码模板 eg3_7.jsp 如下：

```
<%@ page language="java" contentType="text/html; charset=GBK"%>
<html>
<head>
    <title>Logo 图片</title>
</head>
<body>
    <img src=" logo.jpg" >
</body>
</html>
```

代码模板 eg3_7_1.jsp 如下：

```
<%@ page language="java" contentType="text/html; charset=GBK" %>
<%@ include file="eg3_7.jsp" %>
<html>
<head>
    <title>include 指令标记</title>
</head>
<body>
  <p>
  <font size="20" color="red" >
  立警为公　执法为民
  </font>
</body>
</html>
```

运行上述代码，显示效果如图 3-16 所示。其中上半部分的图片来自于被包含进来的 eg3_7.jsp 页面效果，下半部分的文字“立警为公　执法为民”来自于 eg3_7_1.jsp 本身页面效果。

图 3-16　include 指令标记应用之一

代码模板 eg3_7_2.jsp 如下：

```
<%@ page language="java" contentType="text/html; charset=GBK" %>
<%@ include file="logo.jsp" %>
<html>
<head>
    <title>include 指令标记</title>
</head>
<body>
  <p>警察(武警或人民警察)
  <p>中国的警察包括武警和人民警察两大类。
  <p>“公安”广义上是指人民警察,分为公安警察、国家安全警察及司法警察。
  <p>人民警察是国家公务员,实行警监、警督、警司、警员的警衔制度,服装以藏黑为主色调。
  <p>武警全称中国人民武装警察部队,是中华人民共和国武装力量的一部分,是担负国家赋予的安全保卫任务的部队。
</body>
</html>
```

运行上述代码,显示效果如图 3-17 所示。

图 3-17　include 指令标记应用之二

与前面的应用类似,图 3-17 中上半部分的图片来自于被包含进来的 eg3_7.jsp 页面效果,下半部分的文字来自于 eg3_7_2.jsp 本身页面效果。

3.5.3　拓展训练

1. 编写一个 JSP 页面，利用 page 指令标记的 import 属性将 java.util 包中的类、java.io 包中的类导入，然后根据实际需求进行使用。调试运行程序后观察页面效果。

2. 把 3.5.2 节任务 2 中 eg3_6.jsp 页面的 contentType 属性值修改为 application/vnd.ms-powerpoint，指定客户端浏览器启用本地的 Microsoft PowerPoint 应用程序来解析执行所接收的信息。运行修改后的页面观察页面效果。

3. 使用“记事本”编写一个文本文件 included.txt。included.txt 的每行有若干个英文单词，这些单词之间用空格分隔，每行之间用＜br＞分隔，具体如下：

included.txt 代码：

```
packag apple void back public
<br>
private throw class hello welcome
```

编写三个 JSP 页面：first.jsp、second.jsp 和 third.jsp，要求每个页面都包含 included.txt 文件，其余页面内容自行设计。调试运行程序后观察页面效果。

3.6　JSP 动作标记

动作标记是一种特殊的标记，它们将影响 JSP 运行时的功能。JSP 有 7 个动作标记，分别是：

jsp:include——用于动态引入一个 JSP 页面。

jsp:param——用于传递参数，必须与其他支持参数的标签一起使用。

jsp:forward——执行页面转向，将请求的处理转发到下一个页面。

jsp:plugin——用于下载 JavaBean 或 Applet 到客户端执行。

jsp:useBean——使用 JavaBean。

jsp:setProperty——修改 JavaBean 实例的属性值。

jsp:getProperty——获取 JavaBean 实例的属性值。

本节着重介绍前 4 个动作标记，其余的动作标记将在第 5 章详细介绍。

3.6.1　知识要点

1. include 动作标记

include 动作标记用于在当前 JSP 页面中动态包含另一个文件，即将当前 JSP 页面、被包含的文件各自独立地翻译为字节码文件。当前 JSP 页面执行到该动作标记时，才加载执行被包含文件的字节码。语法格式如下：

```
<jsp:include page="文件的地址"/>
```

或是

```
<jsp:include 动作标记 page= "文件的地址">
param 子标记
</jsp:include>
```

需要注意的是，当 include 动作标记不需要子标记时，必须使用第一种形式。在 3.5 节中介绍的指令标记与本节的 include 动作标记功能相似，都是将另一个文件嵌入到当前 JSP 页面中，但是其处理 include 指令标记方式和处理时机不同。include 指令标记是静态嵌入，即在编译阶段就处理嵌入的文件，被嵌入的文件在语法上和逻辑上依赖于当前 JSP 页面，其优点是执行速度快；而 include 动作标记是动态嵌入，即在 JSP 页面运行时才处理嵌入的文件，被嵌入的文件在语法上和逻辑上相对独立，其优点是可以借助 param 子标记灵活处理所包含的文件，与此同时也会造成执行速度减慢。

2. param 动作标记

param 动作标记用于设置参数值，这个动作标记本身不能独立使用，独立的 param 动作标记没有实际意义，需作为 jsp:include、jsp:forward、jsp:plugin 标记的子标记来使用，以完成相应的功能。例如当该标记与 jsp:include 动作标记一起使用时，可以将 param 标记中的值传递到 include 动作标记要加载的文件中，被加载的 JSP 文件可以使用 Web 服务器提供的 request 内置对象获取 include 动作标记的 param 子标记中属性所提供的值。

语法格式如下：

```
<jsp:param name="paramName" value="paramValue">
```

3. forward 动作标记

forward 动作标记用于将页面响应转发给另外的页面，既可以转发给静态的 HTML 页面，也可以转发到动态的 JSP 页面。换言之，就是从该动作标记处停止执行当前 JSP 页面，而转向指定的 HTML 页面或 JSP 页面去执行。语法格式如下：

```
<jsp:forward page="将转向的页面"/>
```

或是

```
<jsp:forward page="将转向的页面">
        param 子标记
</jsp:forward>
```

需要注意的是，当 forward 动作标记不需要子标记时，必须使用第一种形式。

4. plugin 动作标记

plugin 动作标记用于执行一个 applet(Java 小应用程序)，因为有些浏览器并不支持 Java applet 程序，所以可以借助 plugin 动作标记保证客户端浏览器能够顺利执行 Java

applet 程序。在具体执行过程中，plugin 动作标记指示 JSP 页面加载 Java plugin，该插件由客户负责下载，并使用该插件来运行 Java applet 程序。语法格式如下：

```
<jsp:plugin
    type="applet "
    code="字节码文件"
    codebase="字节码文件路径"
    [ height="小程序高度值" ]
    [ width="小程序宽度值" ]
    [ jreversion="JRE 版本号" ]
    [ <jsp:fallback>提示用户的文本信息</jsp:fallback> ]
</jsp:plugin>
```

下面的例子代码，表明小应用程序的字节码文件是 A.class，且该字节码文件存放在当前 Web 服务目录中；小应用程序显示高度为 180 像素，宽度为 200 像素，JRE 版本号为 1.1；当插件不能启动时显示给用户一段提示性文字“Unable to load applet”。

```
<jsp:plugin
    type="applet "
    code="A.class"
    codebase="."
    height="180"
    width="200"
    jreversion="1.1"
    <jsp:fallback>Unable to load applet.</jsp:fallback>
</jsp:plugin>
```

需要注意的是，上面语法格式中被“[]”括起来的部分为可选项，可以根据实际情况来决定是否对其进行设置。标记<jsp:fallback>是<jsp:plugin>动作标记的一部分，并且只能在<jsp:plugin>动作中使用。

3.6.2　技能操作

使用 JSP 动作标记，完成相应的功能。具体任务如下：

- 利用 include 动作标记和 param 动作标记完成简单的欢迎登录页面。
- 利用 forward 动作标记和 param 动作标记实现奇偶页的不同跳转页面。

1. 利用 include 动作标记和 param 动作标记完成简单的欢迎登录页面。

代码模板 eg3_8.jsp 如下：

```
<%@ page language="java" contentType="text/html; charset=GB2312" %>
<html>
    <head>
        <title>JSP 动作标记</title>
    </head>
```

```
    <body bgcolor="cyan">
    <%
    String user="sa",password="123456";
    %>
    <jsp:include page="success.jsp">
      <jsp:param name="text1" value="<%=user %>"/>
      <jsp:param name="text2" value="<%=password %>"/>
    </jsp:include>
  </body>
</html>
```

代码模板 success.jsp 如下：

```
<%@ page language="java" contentType="text/html; charset=GBK"%>
登录成功,欢迎
<br>
您的用户名是:
<%=request.getParameter("text1")%>
<br>
您的密码是:
<%=request.getParameter("text2")%>
```

最终运行结果如图 3-18 所示。可见，eg3_8.jsp 中通过下面两行语句：

```
<jsp:param name="text1" value="<%=user %>"/>
<jsp:param name="text2" value="<%=password %>"/>
```

设定了传递的参数列表，即名称为 text1 的参数其值为 user 变量的值，也就是"sa"字符串；名称为 text2 的参数其值为 password 变量的值，也就是"123456"字符串。

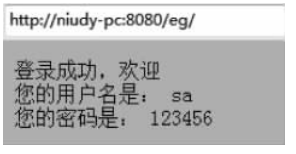

图 3-18 欢迎登录页面效果

2. 利用 forward 动作标记和 param 动作标记实现奇偶页的不同跳转页面。

代码模板 eg3_9.jsp 如下：

```
<%@ page language="java" contentType="text/html; charset=GBK"%>
<html>
  <body>
    <%
    int number=(int)(Math.random()*100+1);
    if(number%2!=0){
    %>
    <jsp:forward page="eg3_9_1.jsp">
    <jsp:param name="text" value="<%=number%>"/>
    </jsp:forward>
    <%
```

```
        }
        else
        {
        %>
        <jsp:forward page="eg3_9_2.jsp">
        <jsp:param name="text" value="<%=number%>"/>
        </jsp:forward>
        <% }
        %>
  </body>
</html>
```

代码模板 eg3_9_1.jsp 如下：

```
<%@ page language="java" contentType="text/html; charset=GBK"%>
<html>
<body bgcolor="cyan">
传递过来的随机数为:
<%=request.getParameter("text")%>
<p>
该奇数页图片为:
<p>
<img src="D:\picture1.jpg">
</body>
```

代码模板 eg3_9_2.jsp 如下：

```
<%@ page language="java" contentType="text/html; charset=GBK"%>
<html>
<body bgcolor="cyan">
传递过来的随机数为:
<%=request.getParameter("text")%>
<p>
该偶数页图片为:
<p>
<img src="D:\picture2.jpg">
</body>
</html>
```

具体运行效果如图 3-19 和图 3-20 所示。从图中可以看出，当 eg3_9.jsp 中产生的随机数为奇数时，则跳转到 eg3_9_1.jsp 页面，显示图像为人物左脸；当 eg3_9.jsp 中产生的随机数为偶数时，则跳转到 eg3_9_2.jsp 页面，显示图像为人物右脸。

图 3-19 奇数页跳转页面效果

图 3-20 偶数页跳转页面效果

3.6.3 拓展训练

1. 编辑下面给出的代码模板，调试运行并观察页面效果，体会 include 指令标记和 include 动作标记的联系与区别。

代码模板 static.html 如下

```
<%@ page language="java" contentType="text/html; charset=GBK"%>
<html>
    <head>
        <title>JSP 动作标记</title>
    </head>
    <body bgcolor="cyan">
        name:<input type="text" value="123">
        password:<input type="text" value="123">
        <input type="button" value="login" name="button">
    </body>
</html>
```

代码模板 dynamic.jsp 如下：

```
<%@ page language="java" contentType="text/html; charset=GBK"%>
<html>
    <head>
        <title>JSP 动作标记</title>
    </head>
    <body>
```

```
        我来自第 2 个文件.
    </body>
</html>
```

代码模板 index.jsp 如下：

```
<%@ page language="java" contentType="text/html; charset=GBK"%>
<html>
    <head>
        <title>JSP 动作标记</title>
    </head>
    <body bgcolor="cyan">
        这是 include 指令标记举例.
        <br><%@ include file="static.html" %>
        <br>这是 include 动作标记举例.
        <br><jsp:include page="dynamic.jsp" flush="true"/>
    </body>
</html>
```

2. 模仿 3.6.2 节中的例题，使用 forward 动作标记和 param 动作标记实现页面跳转功能。要求：编写一个静态 HTML 文件 decide.html，在该文件中定义一个变量 i 并为其赋初值，当 i 的值大于 0 时，跳转到页面 positive.jsp；否则跳转到页面 negative.jsp。positive.jsp 和 positive.jsp 页面内容可以自行设计。

3.7　小　　结

- 一个 JSP 页面通常由普通的 HTML 标记、JSP 注释、Java 动态元素（包括变量和方法的声明、Java 程序片、Java 表达式）以及 JSP 标记（包括指令标记、动作标记和自定义标记）组成。
- 在一个 Java 程序片中声明的变量称为 JSP 页面的局部变量，它们在 JSP 页面后续的所有程序片部分以及表达式部分有效。一个用户对 JSP 页面局部变量操作的结果，不会影响到其他用户。
- JSP 页面成员变量是被所有用户共享的变量，任何用户对 JSP 页面成员变量操作的结果，都会影响其他用户。
- page 指令标记用来定义整个 JSP 页面的一些属性以及这些属性的值。page 指令只能为 contentType 属性指定一个值，但可以为 import 属性指定多个值。
- include 指令标记是先将当前 JSP 页面与要嵌入的文件合并成一个新的 JSP 页面，然后再由 JSP 引擎将新页面转化为 Java 文件处理并运行。而 include 动作标记在把 JSP 页面转译成 Java 文件时，并不合并两个页面；而是在 Java 文件的字节码文件被加载和执行时，才去处理 include 动作标记中引入的文件。