第 3 章

JSP 内置对象

本章主要内容

- 请求对象 request
- 响应对象 response
- 会话对象 session
- 全局应用程序对象 application

有些对象在 JSP 页面中不需要声明和实例化，可以直接在 Java 程序片和 Java 表达式部分使用，这些对象就是 JSP 的内置对象。内置对象由 Web 服务器负责实现和管理。

常见的 JSP 内置对象有 request、response、session、application 以及 out。在本章中将主要学习这几种内置对象的使用方法。

在本章中，将新建一个 Web 工程 ch3，本章例子中涉及的 JSP 页面保存在 ch3 的 WebContent 目录中。

3.1 请求对象 request

3.1.1 核心知识

request 内置对象是实现了 javax. servlet. ServletRequest 接口的一个实例。当用户请求一个 JSP 页面时，JSP 页面所在的服务器将用户发出的所有请求信息封装在内置对象 request 中，使用该对象就可以获取用户提交的信息。

request 对象获取客户提交信息的两个常用方法如下。

1. public String getParameter(String name)

该方法以字符串的形式返回客户端传来的某个参数的值，该参数名由 name 指定。例如：

```
<form action="getValue.jsp">
    <input type="text" name="userName"/>
    <input type="submit" value="提交"/>
</form>
```

单击“提交”按钮向 JSP 页面 getValue. jsp 提交信息后就可以获得文本框中输入的信息，代

码如下：

```
String name=request.getParameter("userName");
```

2. public String[] getParameterValues(String name)

该方法以字符串数组的形式返回客户端向服务器端传递的指定参数名的所有值。例如：

```
<form action="getValues.jsp">
    选择你去过的城市:<br/>
    <input type="checkbox" name="cities" value="Beijing"/>北京
    <input type="checkbox" name="cities" value="Shanghai"/>上海
    <input type="checkbox" name="cities" value="Hong Kong"/>香港
    <input type="submit" value="提交"/>
</form>
```

如果选择了北京和上海两个城市，提交表单后可以用 getParameterValues 方法，以复选框的 name 属性值"cities"为参数，获取到一个数组，其中元素分别对应北京和上海两个选项的 value 值"Beijing"和"Shanghai"。代码如下：

```
String yourCities[]=request.getParameterValues("cities");
```

3.1.2 能力目标

能够灵活使用 request 内置对象获取客户提交的信息。

3.1.3 任务驱动

1. 任务的主要内容

编写 example3_1.jsp 和 getValue.jsp 两个 JSP 页面，example3_1.jsp 通过表单向 getValue.jsp 提交输入的姓名和选择的城市，getValue.jsp 负责获得表单中提交的信息并显示。页面运行效果如图 3.1 所示。

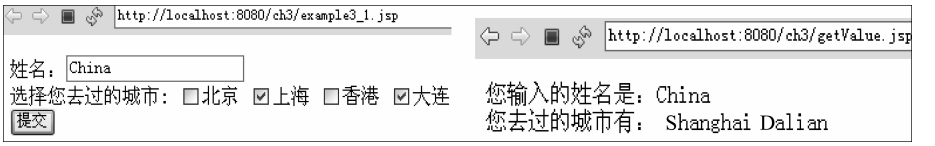


图 3.1 example3_1.jsp 的效果图

2. 任务的代码模板

将下列 example3_1.jsp 中的【代码】替换为真正的 JSP 的代码。

example3_1.jsp

```
<%@page language="java" contentType="text/html; charset=GBK" pageEncoding=
"GBK"%>
<html>
<head>
<title>example3_1.jsp</title>
```

```
</head>
<body>
    <form action="getValue.jsp">
        姓名:<input type="text" name="userName"/>
        <br>
        选择您去过的城市:
        <input type="checkbox" name="cities" value="Beijing"/>北京
        <input type="checkbox" name="cities" value="Shanghai"/>上海
        <input type="checkbox" name="cities" value="Hong Kong"/>香港
        <input type="checkbox" name="cities" value="Dalian"/>大连
        <br>
        <input type="submit" value="提交"/>
    </form>
</body>
</html>
```

getValue.jsp

```
<%@page language="java" contentType="text/html; charset=GBK" pageEncoding=
"GBK"%>
<html>
<head>
<title>getValue.jsp</title>
</head>
<body>
<%
    String name=【代码 1】                    //获得 example3_1.jsp 页面中输入的姓名
    String cities[]=【代码 2】                //获得 example3_1.jsp 页面中选择的城市
%>
    您输入的姓名是:<%=name %><br>
    您去过的城市有:
    <%
        for(int i=0;i<cities.length;i++){
            out.print(cities[i]+"  ");
        }
    %>
</body>
</html>
```

3. 任务小结或知识扩展

1）NullPointerException 异常

如果不选择 example3_1.jsp 页面的城市，而直接单击“提交”按钮，那么 getValue.jsp 页面就会提示出现 NullPointerException 异常。为了避免在运行时出现 NullPointerException 异常，我们在 getValue.jsp 页面中使用如下代码。

```
if(cities!=null){
    for(int i=0;i<cities.length;i++){
        out.print(cities[i]+"  ");
    }
}
```

2）中文乱码问题

如果在 example3_1.jsp 页面的文本框中输入中文姓名，那么 getValue.jsp 页面获得的姓名是乱码（由很多“?”组成）。此时必须对含有汉字字符的信息进行特殊的处理。乱码解决常用的方法有如下两种。

（1）使用 setCharacterEncoding(String code)设置统一字符编码

request 对象提供了方法 setCharacterEncoding(String code)设置编码，其中参数 code 以字符串形式传入要设置的编码格式，但这种方法仅对于提交方式是 post 的表单（表单默认的提交方式是 get）有效。例如，我们使用该方法解决 example3_1.jsp 和 getValue.jsp 页面中出现的中文乱码问题，需要完成如下两件事。

首先，将 example3_1.jsp 中的表单提交方式改为“post”，具体代码如下：

```
<form action="getValue.jsp" method="post">
```

其次，在 getValue.jsp 中获取表单信息之前设置统一编码，具体代码如下：

```
request.setCharacterEncoding("GBK");
```

（2）对获取的信息进行重新编码

通过内置对象 request 获取到字符串的值后，对该字符串使用 ISO-8859-1 重新编码，并把编码的结果存放到一个字节数组中，然后再把这个字节数组转化为字符串。例如，我们使用该方法解决 example3_1.jsp 和 getValue.jsp 页面中出现的中文乱码问题，具体代码如下：

```
String name=request.getParameter("userName");
byte b[]=name.getBytes("ISO-8859-1");
name=new String(b);
```

3）字符集

为了更好地理解中文乱码的解决方案，需要了解几种常用的字符集。

（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）GB2312。GB2312 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，通行于中国内地和新加坡。

（4）GBK。GBK 编码规范，除了完全兼容 GB2312 外，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。

4）request 的常用方法

request 的常用方法除了核心知识中的两个外，还有如下几个。

(1) public void setAttribute(String name,Object obj)

该方法可以将某个参数和它的值与当前的 request 对象绑定。name 为参数的名称,obj 为对应的参数值,必须是复合类型的对象。

(2) public Object getAttribute(String name)

该方法返回之前调用 setAttribute 方法时所设置参数 name 对应的属性值,如果对应的属性值不存在,则会返回 null;如果对应的属性值存在,则返回一个 Object 对象,所以需要进行强制类型转换。

(3) public void removeAttribute(String name)

该方法从 request 对象中删除参数 name 所对应的属性。

4. 代码模板的参考答案

```
【代码 1】:request.getParameter("userName");
【代码 2】:request.getParameterValues("cities");
```

3.1.4　实践环节

使用两种方法(设置统一编码和重新编码)解决 example3_1.jsp 和 getValue.jsp 页面中出现的中文乱码问题。

3.2　响应对象 response

3.2.1　核心知识

当用户请求服务器的一个页面时,会提交一个 HTTP 请求,服务器收到请求后,返回 HTTP 响应。request 对象对请求信息进行封装,与 request 对象对应的对象是 response 对象。response 对象对用户的请求作出动态响应。动态响应通常有如下 3 个。

1. 动态改变 contentType 属性值

JSP 页面用 page 指令标记设置了页面的 contentType 属性值,response 对象就按照这种属性值的方式对客户作出响应。在 page 指令中只能为 contentType 属性指定一个值。如果想动态改变 contentType 属性值,换一种方式来响应客户,可以让 response 对象调用 setContentType(String s)方法来重新设置 contentType 的属性值。

2. 设置响应表头(HTTP 文件头)

response 对象可以通过方法 setHeader(String name,String value)设置指定名字的 HTTP 文件头的值,以此来操作 HTTP 文件头。response 对象设置的新值将会覆盖原值。如果希望某页面每 3 秒钟刷新一次,那么在该页面中添加如下代码。

```
response.setHeader("refresh","3");
```

3. response 重定向

在需要将用户引导至另一个页面时,可以使用 response 对象的 sendRedirect(String url)方法实现用户的重定向。例如,用户输入的表单信息不完整,应该再次被重定向到输入

页面。

3.2.2 能力目标

能够灵活使用 response 内置对象动态响应用户的请求。

3.2.3 任务驱动

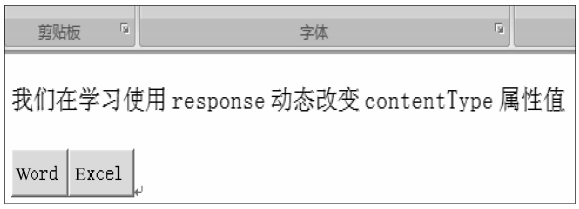
本节有以下 3 个任务。

1. 任务 1——动态改变 contentType 属性值

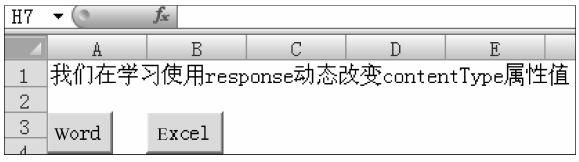
(1) 任务 1 的主要内容

编写一个 JSP 页面 example3_2_1.jsp,客户通过单击页面上的不同按钮,可以改变页面响应的 MIME 类型。当单击 word 按钮时,JSP 页面动态地改变 contentType 的属性值为 application/msword,客户浏览器启用本地的 Word 软件来显示当前页面内容;当单击 excel 按钮时,JSP 页面动态地改变 contentType 的属性值为 application/vnd.ms-excel,客户浏览器启用本地的 Excel 软件来显示当前页面内容。效果如图 3.2(a)～图 3.2(c)所示。

(a) text/html响应方式



(b) application/msword响应方式



(c) application/vnd.ms-excel响应方式

图 3.2　example3_2_1.jsp 的效果图

(2) 任务 1 的代码模板

将下列 example3_2_1.jsp 中的【代码】替换为真正的 JSP 的代码。

example3_2_1.jsp

```
<%@page language="java" contentType="text/html; charset=GBK" pageEncoding=
"GBK"%>
<html>
<head>
<title>example3_2_1.jsp</title>
```

```
</head>
<body>
  <form action="" method="post">
      <p>我们在学习使用 response 动态改变 contentType 属性值
      <p>
      <input type="submit" value="word" name="submit">
      <input type="submit" value="excel" name="submit">
      <%
          String str=request.getParameter("submit");
          if ("word".equals(str)) {
          【代码 1】
            //response 调用 setContentType 方法设置 MIME 类型为 application/msword
          }else if ("excel".equals(str)) {
          【代码 2】
          //response 调用 setContentType 方法设置 MIME 类型为 application/vnd.ms-excel
          }
      %>
  </form>
</body>
</html>
```

(3) 任务 1 小结或知识扩展

response 对象调用 setContentType(String s)方法来重新设置网页响应的 MIME 类型。常见的 MIME 类型有 text/html、application/msword、application/vnd. ms-excel、image/gif、image/jpeg、application/vnd. ms-powerpoint、application/x-shockwave-flash、application/pdf 等。

(4) 任务 1 代码模板的参考答案

```
【代码 1】:response.setContentType("application/msword");
【代码 2】:response.setContentType("application/vnd.ms-excel");
```

2. 任务 2——设置响应表头

(1) 任务 2 的主要内容

编写一个 JSP 页面 example3_2_2.jsp,在该页面中使用 response 对象设置一个响应头“refresh”,其值是“3”。那么用户收到这个头之后,该页面会每 3 秒钟刷新一次。

(2) 任务 2 的代码模板

将下列 example3_2_2.jsp 中的【代码】替换为真正的 JSP 的代码。

example3_2_2.jsp

```
<%@page language="java" contentType="text/html; charset=GBK" pageEncoding=
"GBK"%>
<%@page import="java.util.*" %>
<html>
<head>
<title>example3_2_2.jsp</title>
</head>
<body>
    <h2>该页面每 3 秒钟刷新 1 次</h2>
```

```
<p>现在的秒钟时间是:
<%
    Date d=new Date();
    out.print(""+d.getSeconds());
    【代码】            //使用 response 对象设置一个响应头"refresh",其值是"3".
%>
</body>
</html>
```

(3) 任务 2 小结或知识扩展

有时候希望从当前页面几秒钟后自动跳转到另一个页面。比如,打开 a.jsp 页面 3 秒钟后,自动跳转到 b.jsp 页面(a.jsp 与 b.jsp 在同一个 Web 服务目录下)。这该如何实现呢? 我们只需为 a.jsp 设置一个响应头即可,也就是在 a.jsp 页面中添加如下代码。

```
response.setHeader("refresh","3;url=b.jsp");
```

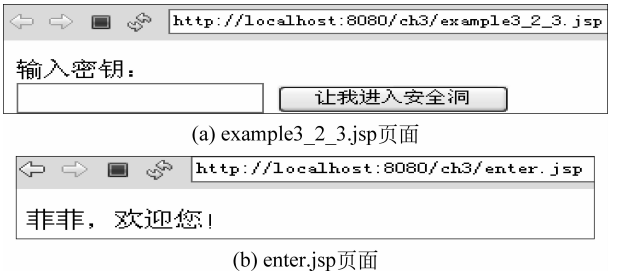
(4) 任务 2 代码模板的参考答案

【代码】:`response.setHeader("refresh","3");`

3. 任务 3——重定向

(1) 任务 3 的主要内容

编写 example3_2_3.jsp 和 enter.jsp 两个 JSP 页面,如果在页面 example3_2_3.jsp 中输入正确的密码"kazhafei",单击"让我进入安全洞"按钮后提交给页面 enter.jsp;如果输入不正确,重定向到 example3_2_3.jsp 页面。先运行 example3_2_3.jsp 页面,页面效果如图 3.3(a)、图 3.3(b)所示。



(a) example3_2_3.jsp页面

(b) enter.jsp页面

图 3.3　example3_2_3.jsp 的效果图

(2) 任务 3 的代码模板

将下列 example3_2_3.jsp 中的【代码】替换为真正的 JSP 的代码。

example3_2_3.jsp

```
<%@page language="java" contentType="text/html; charset=GBK" pageEncoding=
"GBK"%>
<html>
<head>
<title>example3_2_3.jsp</title>
</head>
<body>
```

```
<form action="enter.jsp" method="post" name=form>
    <p>
        输入密钥:
    <br>
    <input type="password" name="pwd"/>
    <input type="submit" value="让我进入安全洞">
</form>
</body>
</html>
```

enter. jsp

```
<%@page language="java" contentType="text/html; charset=GBK" pageEncoding=
"GBK"%>
<html>
<head>
<title>enter.jsp</title>
</head>
<body>
<%
    String str=request.getParameter("pwd");
    if (!"kazhafei".equals(str)) {
        【代码】//重定向到 example3_2_3.jsp 页面重新输入密码
    } else {
        out.print("菲菲,欢迎您!");
    }
%>
</body>
</html>
```

(3) 任务 3 小结或知识扩展

response 对象的 sendRedirect 方法是在用户的浏览器端工作,Web 服务器要求浏览器重新发送一个到被定向页面的请求。在浏览器地址栏上会出现重定向页面的 URL,且为绝对路径。

forward 动作标记也可以实现页面的跳转,如<jsp:forward page="info. jsp"/>。但使用 forward 动作标记与 response 对象调用 sendRedirect 不同。对两者的比较如下:

- forward 为服务器端跳转,浏览器地址栏不变;sendRedirect 为客户端跳转,浏览器地址栏改变为新页面的 URL。
- 执行到 forward 动作标记出现处停止当前 JSP 页面的继续执行,而转向标记中 page 属性指定的页面;sendRedirect 所有代码执行完毕之后再跳转。
- 使用 forward,通过 request 请求信息能够保留在下一个页面;使用 sendRedirect 不能保留 request 请求信息。

forward 传递参数的格式如下:

```
<jsp:forward page="info.jsp">
    <jsp:param name="no" value="001"/>
    <jsp:param name="age" value="18"/>
</jsp:forward>
```

response 对象的 sendRedirect 传递参数的方式如下：

```
response.sendRedirect("info.jsp? no=001&age=18") ;
```

(4) 任务 3 代码模板的参考答案

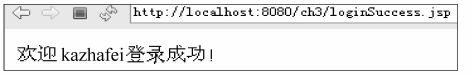
【代码】:`response.sendRedirect("example3_2_3.jsp");`

3.2.4 实践环节

编写 login. jsp、server. jsp 和 loginSuccess. jsp 3 个 JSP 页面，如果在页面 login. jsp 中输入正确的用户名“kazhafei”和正确的密码“aobama”，单击“登录”按钮后提交给页面 server. jsp。在 server. jsp 页面中进行登录验证：如果输入正确，提示“成功登录，3 秒钟后进入 loginSuccess. jsp 页面”；如果输入不正确，重定向到 login. jsp 页面。先运行 login. jsp 页面，页面运行效果如图 3.4(a)～图 3.4(c)所示。

(a) login.jsp页面

(b) server.jsp页面



(c) loginSuccess.jsp页面

图 3.4　页面效果图

3.3 会话对象 session

浏览器与 Web 服务器之间使用 HTTP 协议进行通信。HTTP 是一种无状态协议，客户向服务器发出请求(request)，服务器返回响应(response)，连接就被关闭了，在服务器端不保留连接的相关信息。所以服务器必须采取某种手段来记录每个客户的连接信息。Web 服务器可以使用内置对象 session 来存放有关连接的信息。session 对象指的是客户端与服务器端的一次会话，从客户端连到服务器端的一个 Web 应用程序开始，直到客户端与服务器端断开为止。

3.3.1 核心知识

1. session 对象的 ID

Web 服务器会给每一个用户自动创建一个 session 对象，为每个 session 对象分配一个唯一标识的 String 类型的 session ID，这个 ID 用于区分其他用户。这样每个用户都对应着一个 session 对象(该用户的会话)，不同用户的 session 对象互不相同。session 对象调用 getId()方法就可以获取当前 session 对象的 ID。