第 5 章

JSP 中使用 JavaBean

JSP 最强有力的一个方面就是能够使用 JavaBean 组件。JavaBean 是一个可重复使用的软件组件。JavaBean 是一种 Java 类，通过封装属性和方法成为具有某种功能或者能够处理某些业务的对象，简称 Bean。

一个基本的 JSP 页面由静态的 HTML 标签和 Java 脚本组成，如果 Java 脚本和 HTML 标签大量掺杂在一起，就显得页面混杂，不易维护。JSP 页面可以将数据的处理过程指派给一个或者几个 JavaBean 来完成，而在 JSP 页面中调用 JavaBean。不提倡大量的数据处理都用页面中的 Java 脚本来完成。JSP 页面中调用 JavaBean，可有效地分离静态工作部分和动态工作部分，实现业务逻辑和表现形式的分离。JavaBean 负责业务逻辑的处理，JSP 负责页面的展示，如图 5-1 所示。

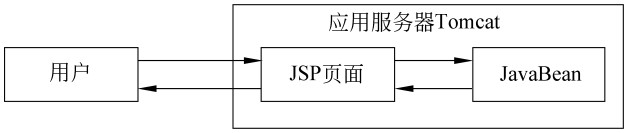

图 5-1　JSP＋JavaBean 模型

5.1　JavaBean 介绍

JavaBean 体系结构是全面基于组件的标准模型。JavaBean 是 Java 描述的软件组件模型，有点类似于 Microsoft 的 COM 组件概念。JavaBean 组件是 Java 类，这些类遵循特定的接口格式，以便于容器按照标准的方式使用方法命名、底层行为来构造和访问 JavaBean 对象。

5.1.1　JavaBean 简介

1. JavaBean 的特点

- 可以实现代码的重复利用。
- 易编写、易维护、易使用。
- 可以在任何支持 Java 的平台上工作，而不需要重新编译。
- 可以通过网络传输。
- 可以与其他 Java 类同时使用。

2. JavaBean 的应用范围

JavaBean 传统的应用在于可视化领域，如 AWT（抽象窗口工具集）和 Swing 下的应用。现在，JavaBean 更多的应用在于非可视化领域，它在服务器端应用方面表现出了越来越强的生命力。非可视化的 JavaBean 和可视化的 JavaBean 同样使用属性和事件。非可视化的 JavaBean 在 JSP 程序中常用来封装业务逻辑、数据库操作等，可以很好地实现业务逻辑和前台页面的分离，使得系统具有更好的健壮性和灵活性。

注意：JavaBean 和 EJB（Enterprise JavaBean）的概念是完全不同的。EJB 分为三类：会话 Bean（Session Bean）、实体 Bean（Entity Bean）、消息驱动 Bean（Message Driven Bean）。EJB 很复杂，而 JavaBean 是非常简单朴素的。

5.1.2 编写 JavaBean 遵循的原则

编写 JavaBean 就是编写一个 Java 的类，所以只要学会写类就能编写一个 JavaBean，这个类创建的一个对象称之为 JavaBean。为了让使用 JavaBean 的应用程序构建工具（比如 JSP 引擎）知道这个 Bean 的属性和方法，JavaBean 的类需要遵守以下规则：

- 必须具备一个零参数的构造方法，显式地定义这样一个无参构造方法或者省略所有的构造方法都能满足这项要求。
- 成员变量也称为属性，JavaBean 不应该有公开的成员变量，需使用存取方法读取和修改属性，不允许对字段直接访问。属性的名字建议以小写英文字母开头。
- 属性的值通过 getXxx()和 setXxx()方法来访问。例如，如果类有 String 类型的属性 title，读取 title 的方法是返回 String 的 getTitle()，修改 title 的方法是 setTitle(String title)。
- 布尔型属性的读取方法可以使用 getXxx()，也可以使用 isXxx()。
- JavaBean 需要定义在包里面，package 保留字给出类所在的包。

具有两个属性的 JavaBean：User 类

```
package cn.edu.uibe.model;
public class User {
    private String userName;
    private String password;
    public String getUserName() { return userName; }
    public void setUserName(String userName) { this.userName = userName; }
    public String getPassword() { return password; }
    public void setPassword(String password) { this.password = password; }
}
```

【技巧】 Eclipse 可以帮助我们生成 Getter 和 Setter 方法，依次选择 Eclipse→Source，打开 Generate Getters and Setters 对话框，如图 5-2 所示。

5.1.3 JavaBean 的属性

属性是 JavaBean 组件内部状态的表示，JavaBean 的属性可以分为以下 4 类：

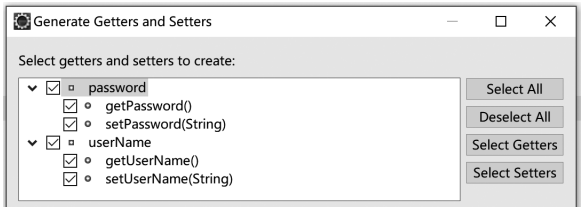

图 5-2　使用 Eclipse 生成 Getter 和 Setter 方法

- Simple(简单属性)。
- Indexed(索引属性)。
- Bound(绑定属性)。
- Constrained(约束属性)。

可以在 JSP 页面中使用的属性是简单属性和索引属性,而绑定属性和约束属性是在图形界面开发时才会用到,下面介绍简单属性和索引属性。

1. 简单属性

```
package cn.edu.uibe.model;
public class SimpleAttrBean {
    private String attr ;                          //JavaBean 的属性
    public SimpleAttrBean(){ }                     //零参数的构造方法
    public String getAttr(){                       //读取属性的方法
        return attr;
    }
    public void setAttr(String attr){              //设置属性的方法
        this.attr = attr;
    }
}
```

2. 索引属性

索引属性是一个数组,读取和写入属性也是使用 Getter 方法和 Setter 方法。

```
package cn.edu.uibe.model;
public class IndexedAttrBean {
    private int[] dataSet = {1,2,3,4,5};           //索引属性
    public IndexedAttrBean(){}                     //零参数构造方法
    public int[] getDataSet(){                     //返回整个数组
        return dataSet;
    }
    public int getDataSet(int index){              //返回数组中的一个元素
        return dataSet[index];
    }
    public void setDataSet(int[] x){               //设置整个数组
        dataSet = x;
    }
```

```
    public void setDataSet(int index, int x){   //设置数组中的一个值
        dataSet[index] = x;
    }
}
```

5.2 <jsp:useBean>

在 JSP 页面中<jsp:useBean>动作元素可以使用 JavaBean 对象，这样就可以将大部分业务处理逻辑封装在 JavaBean 里面。本节介绍<jsp:useBean>的基本语法、JavaBean 的条件化操作、JavaBean 的存放位置和 JavaBean 的作用范围。

5.2.1 <jsp:useBean>的基本语法

<jsp:useBean>动作元素用来在 JSP 页面中获取或者创建一个 JavaBean 对象，并指定它的名字和作用范围。JSP 容器确保 JavaBean 对象在指定的范围内可以使用。<jsp:useBean>的语法格式如下：

```
<jsp:useBean id="beanName" class="package.BeanClassName"
                scope="page | request | session | application"/>
<jsp:useBean id="beanName" class="package.BeanClassName"
                scope="page | request | session | application">
</jsp:useBean>
<jsp:useBean id="beanName" class="package.BeanClassName"
             scope="page | request | session | application">
    //仅当 JavaBean 实例化时才执行
    <jsp:setProperty name="propertyName" value="propertyValue"/>
    ...
</jsp:useBean>
```

当服务器上某个含有<jsp:useBean>动作元素的 JSP 页面被加载执行时，JSP 引擎首先根据 ID 给出的 JavaBean 的名字(beanName)，在 scope 范围对应的 JSP 内部对象上查找是否有这个名字的属性(Attribute)。如果在指定范围内找到了与 beanName 同名的属性，JSP 引擎返回属性对应的对象引用。如果在指定的范围内没有找到与 beanName 同名的属性，JSP 引擎根据 class 属性给出的包名和类名，创建一个该类的对象，并将该对象作为属性名为 beanName 的属性值，绑定到 scope 范围对应的 JSP 内部对象上。创建 JavaBean 对象时，JSP 引擎使用无参数的构造方法。由于 pageContext 对象可以操作各个范围的属性，实际上 JavaBean 的查找和绑定都是通过 pageContext 对象来完成的。如果没有指定 JavaBean 的范围，默认范围是 page。

<jsp:useBean>的含义可以通过下面的代码来理解：

```
Object beanName = pageContext.getAttribute("beanName", SCOPE);
if(beanName==null){
    Object bean = new package.BeanClassName();
```

```
        pageContext.setAttribute("beanName", bean, SCOPE);
    }
```

在pageContext类的定义中，SCOPE的取值有：PAGE_SCOPE、REQUEST_SCOPE、SESSION_SCOPE和APPLICATION_SCOPE，对应的JSP内部对象分别是pageContext、request、session和application。

例如，将User.java中定义的类cn.edu.uibe.model.User定义为名字为user，范围是request的JavaBean。

```
<jsp:useBean id="user" class="cn.edu.uibe.model.User" scope="request"/>
```

或者：

```
<jsp:useBean id="user" class="cn.edu.uibe.model.User" scope="request">
</jsp:useBean>
```

5.2.2 JavaBean的条件化操作

使用<jsp:useBean>在JSP页面中操作JavaBean对象时，bean对象不一定是新创建的，<jsp:useBean>和</jsp: useBean>之间的语句也不一定执行。

1. bean的条件化创建

- 仅当找不到相同id和scope的bean时，<jsp:useBean>才会引发bean新实例的创建。
- 如果找到相同id和scope的bean，则仅仅是将已经存在的bean赋值给由id指定的变量。

HelloBean.java中定义了一个JavaBean的类HelloBean，它只有一个属性message，bean提供了getMessage()和setMessage()方法来访问属性message。

```
package cn.edu.uibe.model;
public class HelloBean {
    private String message;
    public String getMessage() {
        return message;
    }
    public void setMessage(String message) {
        this.message = message;
        System.out.println(message);
    }
}
```

request_bean1.jsp中使用<jsp:useBean>动作元素创建了一个HelloBean类的JavaBean对象，范围是request，bean的名字为hello。页面的小脚本中的调用hello.setMessage()方法将message属性值设置为字符串"Hello JavaBean!"。页面又使用<jsp:include>动作元素包含了另外一个JSP页面request_bean2.jsp。这两个JSP页面对

应的是同一个 request 对象，可以共享范围是 request 的 JavaBean 对象。

```
<%@ page contentType="text/html; charset=GB18030" %>
<!DOCTYPE html><html><head><meta charset="GB18030">
<title>Bean 的条件化创建</title></head><body>
<jsp:useBean id="hello" class ="cn.edu.uibe.model.HelloBean"
             scope="request"/>
<%
  //可以在脚本中直接使用 JavaBean 对象
  hello.setMessage("Hello JavaBean!");
%>
<jsp:include page="request_bean2.jsp"/>
</body></html>
```

在被包含的 JSP 页面 request_bean2.jsp 中，也有一个<jsp:useBean>动作元素，id 为 hello，class 为 cn.edu.uibe.model.HelloBean，scope 为 request，但这并不会导致重新实例化一个 JavaBean 对象，因为 request 范围内已经有了一个名字为 hello 的 JavaBean，这里仅仅是引用那个 JavaBean 对象而已。页面中调用 hello.getMessage()将得到在第一个页面中设置的属性值，即"Hello JavaBean!"。如果在第二个页面中是重新实例化的 JavaBean 对象，那么 getMessage()的输出将是空值 null。

```
<%@ page contentType="text/html; charset=GB18030" %>
<jsp:useBean id="hello" class="cn.edu.uibe.model.HelloBean"
             scope="request"/>
<%
    String message = hello.getMessage();
%>
<p><%=message%></p>
```

使用浏览器访问 request_bean1.jsp，运行结果如图 5-3 所示。

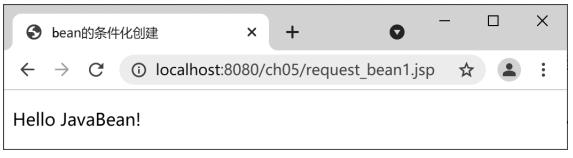

图 5-3　bean 的条件化创建

2. bean 属性的条件化设置

- 将<jsp:useBean ... />替换为<jsp:useBean ...>　</jsp: useBean>。
- 标签内部的语句仅当创建新的 bean 时才执行，如果找到已有的 bean，则不执行。

request_bean3.jsp 中使用<jsp:useBean>动作元素创建了一个 HelloBean 类的 JavaBean 对象，范围为 request，bean 的名字为 hello，属性 message 的值设置为字符串"好消息!"。页面又使用<jsp:include>动作元素包含了另外一个 JSP 页面 request_bean4.jsp。

```
<%@ page contentType="text/html; charset=GB18030" %>
<!DOCTYPE html><html><head><meta charset="GB18030">
<title>Bean 的条件化创建</title></head><body>
<jsp:useBean id="hello" class="cn.edu.uibe.model.HelloBean"
             scope="request">
<%
    hello.setMessage("好消息!");
%>
</jsp:useBean>
<jsp:include page="request_bean4.jsp"/>
</body></html>
```

在被包含的 JSP 页面 request_bean4.jsp 中，也有一个<jsp:useBean>动作元素，id 为 hello，class 为 cn.edu.uibe.model.HelloBean，scope 为 request。在<jsp:useBean>和</jsp:useBean>之间，使用 hello.setMessage()重新设置属性的值为"坏消息!"。请读者思考：这行语句会执行吗？输出是“好消息!”还是“坏消息!”？

```
<%@ page contentType="text/html; charset=GB18030" %>
<jsp:useBean id="hello" class="cn.edu.uibe.model.HelloBean"
             scope= "request">
<%
    hello.setMessage("坏消息!");
%>
</jsp:useBean>
<p><%=hello.getMessage()%></p>
```

使用浏览器访问 request_bean3.jsp，运行结果如图 5-4 所示。

图 5-4　bean 属性的条件化设置

5.2.3　JavaBean 存放的位置

为了在 JSP 页面中使用 JavaBean，应用服务器需要使用字节码文件(扩展名为.class)来创建 Java 对象，这就要求应用服务器能找到字节码，因此字节码文件需要位于特定的目录中。

1. 零散存放

Java 的每个类都对应一个.class 文件，零散存放是指将这些.class 文件存放在 Web 应用的/WEB-INF/classes 目录下包名对应的子目录中。例如，在 JSP 页面中有：

```
<jsp:useBean id="hello" class="cn.edu.uibe.model.HelloBean" />
```

这个 bean 的名字是 hello，类是 cn.edu.uibe.model.HelloBean，生成的字节码文件是 HelloBean.class，它存放的位置是：

```
/WEB-INF/classes/cn/edu/uibe/model/HelloBean.class
```

2. 打包存放

Java 允许将多个.class 文件打包成一个扩展名为.jar 的压缩文件，实际上采用的压缩格式就是 ZIP 格式。多个 JavaBean 可以打包成一个.jar 文件，然后将打包后的.jar 文件存放在 Web 应用的/WEB-INF/lib 目录下。

Eclipse 提供了将.class 文件打包的工具：依次单击 Eclipse→File→Export→Java→JAR File，导出时选择要导出哪些 package 并给出生成的 jar 文件的文件名。在压缩文件内部，扩展名为.class 的文件存放在包名对应的子目录中。

例如将 cn.edu.uibe.model.HelloBean 的字节码文件 HelloBean.class 打包在文件名是 hello.jar 的压缩文件中，那么 hello.jar 压缩文件内部的目录结构是 cn/edu/uibe/model/HelloBean.class，而文件 hello.jar 存放的位置是 Web 应用的/WEB-INF/lib 目录。

5.2.4 JavaBean 的作用范围

JSP 中使用 JavaBean 实际上是将 JavaBean 对象作为一个属性（Attribute）绑定到了 pageContext 对象、request 对象、session 对象或者 application 对象上。scope 对应的取值分别为 page、request、session 和 application，JavaBean 的作用范围分别对应一个页面、一次请求、一个用户的会话和一个 Web 应用。

1. page 范围

- 指定为 page 范围的 JavaBean 仅在定义的 JSP 页面内有效。
- page 范围的 JavaBean 实际上是绑定在 pageContext 对象上的一个属性，可以通过 pageContext.getAttribute("beanName")来得到这个 JavaBean 对象。
- 如果没有指定 JavaBean 的范围，默认范围是 page。

2. request 范围

- 指定为 request 范围的 JavaBean 在客户端的一次请求期间有效。
- request 范围的 JavaBean 实际上是绑定在 request 对象上的一个属性，可以通过 request.getAttribute("beanName")来得到这个 JavaBean 对象。
- request 范围的 JavaBean 可以在客户端的一次请求期间共享和传递数据。

一次请求会经过多个处理环节的情况有：

- 当使用<jsp:include>、<jsp:forward>时，包含或者转向的页面与当前页面对应一个 request 对象。
- 当使用 RequestDispatcher 分发请求时，请求会转发到其他页面。
- 当使用过滤器 Filter 时，请求对象 request 会被过滤器截获。
- 当使用请求相关的监听器时，请求对象 request 可在监听器内部获得。

3. session 范围

- 指定为 session 范围的 JavaBean 在用户的一个会话期间有效。一个会话对应一

段时间内客户端的多次HTTP请求。

- session范围的JavaBean实际上是绑定在session对象上的一个属性，可以通过session.getAttribute("beanName")来得到这个JavaBean对象。
- session范围的JavaBean可以在同一个客户端的多次请求期间共享和传递数据。

session范围的JavaBean一般用于：

- 用户登录后，在session上绑定JavaBean来保存用户信息。
- session上绑定存储商品列表的JavaBean来实现购物车。

4. application范围

- 指定为application范围的JavaBean在Web应用停止之前一直有效。
- application范围的JavaBean实际上是绑定在application对象上的一个属性，程序可以通过application.getAttribute("beanName")来得到这个JavaBean对象。
- application范围的JavaBean的作用域是所在的Web应用。通过application范围的JavaBean，可以在不同客户端的不同次请求间共享和传递数据。

5.3　获取JavaBean的属性

在JSP页面中可以使用<jsp:getProperty>动作元素获取并输出JavaBean的属性，也允许使用表达式语言EL获取并输出JavaBean的属性。

5.3.1　<jsp:getProperty>

<jsp:getProperty>动作元素用来访问JavaBean的属性。访问的属性值将被转化成字符串，然后发送到输出流中。如果属性是一个对象，将调用该对象的toString()方法。<jsp:getProperty>动作元素是通过调用JavaBean的Getter方法获取属性值的。

```
<jsp:getProperty name="beanName" property="propertyName"/>
```

或者：

```
<jsp:getProperty name="beanName" property="propertyName"></jsp:getProperty>
```

特别需要注意的是，<jsp:getProperty>使用name属性给出JavaBean的名字，而<jsp:useBean>使用id属性给出JavaBean的名字，实际上它们是一致的，都是指绑定在特定范围对象的一个属性(Attribute)名。

1. 读取属性　getproperty.jsp

示例getproperty.jsp文件内容如下，运行结果如图5-5所示。

```
<%@ page contentType="text/html; charset=GB18030" %>
<!DOCTYPE html><html><head>
<meta charset="GB18030">
<title>获取并输出属性的值</title>
<jsp:useBean id="hello" class="cn.edu.uibe.model.HelloBean"/>
<%  hello.setMessage("Hello JavaBean Again!");  %>
```

```
</head><body>
<p><jsp:getProperty name="hello" property="message"/></p>
</body></html>
```

图 5-5　使用＜jsp:getProperty＞读取 JavaBean 的属性

2. 深刻理解＜jsp:getProperty＞：getproperty2.jsp

JSP 中使用＜jsp:getProperty＞读取 JavaBean 的属性时，实际是通过调用 Getter 方法完成的，而不管 JavaBean 中是否真的有与属性名对应的成员变量的定义。下面示例中，JSP 内读取 message 属性时，实际上是调用 getMessage()方法，即使 message 属性不存在，也能通过调用 getMessage()方法得到一个字符串，运行结果如图 5-6 所示。

```
package cn.edu.uibe.model;
public class MyBean {
    public String getMessage(){   return "重大利好!";   }
}
```

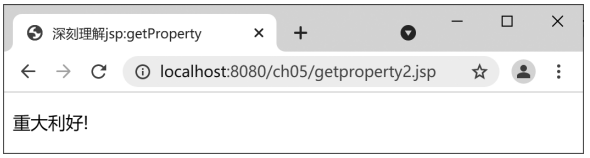

图 5-6　<jsp:getProperty>是调用 getMessage()方法，而不是直接访问属性

getproperty2.jsp 中实例化一个 MyBean 类的对象，并使用<jsp:getProperty>读取属性 message。而在 MyBean 类的定义中，是没有名字为 message 的成员变量的，但仍然能调用 getMessage()方法返回结果。

```
<%@ page contentType="text/html; charset=GB18030" %>
<!DOCTYPE html><html><head><meta charset="GB18030">
<title>深刻理解 jsp:getProperty</title>
<jsp:useBean id="mybean" class="cn.edu.uibe.model.MyBean"/>
</head><body><p>
<jsp:getProperty name="mybean" property="message"/>
</p></body></html>
```

5.3.2　使用 EL 获取 JavaBean 属性

在 JSP 2.0 及之后的版本中，可以使用表达式语言 EL 获取并输出 JavaBean 的属性，它与<jsp:getProperty>动作元素的功能相同，但语法更加简洁。EL 实际上也是调用