第 5 章 JSP与JavaBean

本章导读

主要内容

- 编写和使用 JavaBean
- 获取和修改 bean 的属性
- bean 的辅助类
- JSP 与 bean 结合的简单例子

难点

- 获取和修改 bean 的属性

关键实践

- 记忆测试

在谈论组件之前让我们看一个常见的事——组装电视机。组装一台电视机时，人们可以选择多个组件，例如电阻、电容、显像管等，一个组装电视机的人不必关心显像管是怎么研制的，只要根据说明书了解其属性和功能就可以了。不同的电视机可以安装相同的显像管，显像管的功能完全相同。如果一台电视机的显像管发生了故障，并不会影响其他的电视机；如果两台电视机安装了一个共享的组件——天线，当天线发生了故障，两台电视机就会受到同样的影响。

按照 Sun 公司的定义，JavaBean 是一个可重复使用的软件组件。实际上 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处理某个业务的对象，简称 bean。由于 JavaBean 是基于 Java 语言的，因此 JavaBean 不依赖平台，具有以下特点：

- 可以实现代码的重复利用。
- 易编写、易维护、易使用。
- 可以在任何安装了 Java 运行环境的平台上的使用，而无须重新编译。

JSP 页面可以将数据的处理过程指派给一个或几个 bean 来完成，即 JSP 页面调用这些 bean 完成数据的处理，并将有关处理结果存放到 bean 中，然后 JSP 页面负责显示 bean 中的数据。例如使用 Java 程序片或某些 JSP 指令标记显示 bean 中的数据（见 5.2 节），即 JSP 页面的主要工作是显示数据，不负责数据的逻辑业务处理，如图 5.1 所示。

图 5.1　JSP＋JavaBean

本章在 webapps 目录下新建一个 Web 服务目录 ch5，除非特别约定，本章例子中涉及的 JSP 页面均保存在 ch5 目录中。创建 bean 的类的字节码文件须按要求存放，因此要在 ch5 目录下建立目录结构\ch5\WEB-INF\classes（WEB-INF 字母大写）。

5.1 编写和使用 JavaBean

▶ 5.1.1 编写 JavaBean


视频讲解

编写 JavaBean 就是编写一个 Java 的类，所以只要会写类就能编写一个 JavaBean。这个类创建的一个对象称为一个 JavaBean，简称 bean，分配给 bean 的变量（成员变量），也称 bean 的属性。为了能让使用 bean 的应用程序构建工具（比如 Tomcat 服务器）使用 JSP 动作标记知道 bean 的属性和方法，在类的命名上需要遵守以下规则：

（1）如果类的成员变量（也称 bean 的属性）的名字是 xxx，那么为了获取或更改 bean 的属性的值，类中必须提供两个方法：

- getXxx()，用来获取属性 xxx。
- setXxx()，用来修改属性 xxx。

也就是方法的名字用 get 或 set 为前缀，后缀是将属性（成员变量）名字的首字母大写的字符序列。

（2）类中定义的方法的访问权限都必须是 public 的。

（3）类中必须有一个构造方法是 public、无参数的。

下面我们编写一个创建 bean 的 Java 类，并说明在 JSP 中怎样使用这个类创建一个 bean。要求创建 bean 的类带有包名，即 Java 源文件须使用 package 语句给出包名，例如：

```
package gping;
```

或

```
package tom.jiafei;
```

以下是用来创建 bean 的 Java 源文件。

Circle.java（负责创建 bean）

```
package tom.jiafei;
public class Circle {
    double radius;
    public Circle() {
        radius = 1;
    }
    public double getRadius() {
        return radius;
    }
    public void setRadius(double newRadius) {
        radius = newRadius;
    }
    public double circleArea() {
        return Math.PI * radius * radius;
    }
    public double circleLength() {
        return 2.0 * Math.PI * radius;
    }
}
```

将上述 Java 文件保存为 Circle.java。注意，保存 Java 源文件时，“保存类型”选择为“所有文件”，将“编码”选择为“ANSI”。

▶ 5.1.2 保存 bean 的字节码


视频讲解

为了使 JSP 页面使用 bean，Tomcat 服务器必须使用相应的字节码文件创建一个对象，即创建一个 bean。为了让 Tomcat 服务器能找到字节码文件，字节码文件必须保存在特定的目录中。

ch5 \WEB-INF\classes 目录下，根据包名对应的路径，在 classes 目录下再建立相应的子目录。例如，包名 tom.jiafei 对应的路径是 tom\jiafei，那么在 classes 目录下建立子目录结构 tom\jiafei，如图 5.2 所示。

图 5.2 字节码文件的存放位置

将创建 bean 的字节码文件，例如 Circle.class，复制到\WEB-INF\classes\tom\jiafei 中。为了调试程序方便，可以直接按照 bean 的包名将 bean 的源文件（例如 Circle.java）保存在\WEB-INF\classes\tom\jiafei 目录中，然后用命令行进入 tom\jiafei 的父目录 classes（不要进入 tom 或 jiafei 目录）编译 Circle.java：

```
classes > javac tom\jiafei\Circle.java
```

▶ 5.1.3 创建与使用 bean


视频讲解

❶ 使用 bean

使用 JSP 动作标记 useBean 加载使用 bean，语法格式是：

```
<jsp:useBean id = "bean 的名字" class = "创建 bean 的类" scope = "bean 有效范围"/>
```

或

```
<jsp:useBean id = "bean 的名字" class = "创建 bean 的类" scope = "bean 有效范围">
</jsp:useBean >
```

例如：

```
<jsp:useBean id = "circle" class = "tom.jiafei.Circle" scope = "page" />
```

需要特别注意的是，其中的“创建 bean 的类”要带有包名，例如：

```
class = "tom.jiafei.Circle"
```

❷ bean 的加载原理

当 JSP 页面使用 JSP 动作标记 useBean 加载一个 bean 时，Tomcat 服务器首先根据 JSP

动作标记 useBean 中 id 给出的 bean 名字以及 scope 给出的使用范围(bean 生命周期),在 Tomcat 服务器管理的 pageContent 内置对象中查找是否含有这样的 bean(对象)。如果这样的 bean(对象)存在,Tomcat 服务器就复制这个 bean(对象)给 JSP 页面,就是常说的 Tomcat 服务器分配这样的 bean 给 JSP 页面。如果在 pageContent 中没有查找到 JSP 动作标记要求的 bean,就根据 class 指定的类创建一个 bean,并将所创建的 bean 添加到 pageContent 中。通过 Tomcat 服务器创建 bean 的过程可以看出,首次创建一个新的 bean 需要用相应类的字节码文件创建对象,当某些 JSP 页面再需要同样的 bean 时,Tomcat 服务器直接将 pageContent 中已经有的 bean 分配给 JSP 页面,从而提高 JSP 页面 bean 的使用效率。

注:如果修改了字节码文件,必须重新启动 Tomcat 服务器才能使用新的字节码文件。

❸ bean 的有效范围和生命周期

scope 的取值范围给出了 bean 的生命周期(存活时间),即 scope 取值决定了 Tomcat 服务器分配给用户的 bean 的有效范围和生命周期,因此需要理解 scope 取值的具体意义。下面就 JSP 动作标记 useBean 中 scope 取值的不同情况进行说明。

(1) page bean。scope 取值为 page 的 bean 称为 page bean,page bean 的有效范围是用户访问的当前页面,存活时间直到当前页面执行完毕。Tomcat 服务器分配给每个 JSP 页面的 page bean 是互不相同的。也就是说,尽管每个 JSP 页面的 page bean 的功能相同,但它们占有不同的内存空间。page bean 的有效范围是当前页面,当页面执行完毕,Tomcat 服务器取消分配的 page bean,即释放 page bean 所占有的内存空间。需要注意的是,不同用户(浏览器)的 page bean 也是互不相同的。也就是说,当两个用户同时访问同一个 JSP 页面时,一个用户对自己 page bean 的属性的改变,不会影响到另一个用户。

(2) session bean。scope 取值为 session 的 bean 称为 session bean,session bean 的有效范围是用户访问的 Web 服务目录下的各个页面,存活时间是用户的会话期(session)间,直到用户的会话消失(session 对象达到了最大生存时间或用户关闭自己的浏览器以及服务器关闭,见 4.3.1 节)。如果用户访问 Web 服务目录多个页面,那么每个页面 id 相同的 session bean 是同一个 bean(占有相同的内存空间)。因此,用户在某个页面更改了这个 session bean 的属性值,其他页面的这个 session bean 的属性值也将发生同样的变化。当用户的会话(session)消失,Tomcat 服务器取消所分配的 session bean,即释放 session bean 所占有的内存空间。需要注意的是,不同用户(浏览器)的 session bean 是互不相同的(占有不同的内存空间)。也就是说,当两个用户同时访问同一个 Web 服务目录,一个用户对自己 session bean 属性的改变,不会影响到另一个用户(一个用户在不同 Web 服务目录的 session bean 互不相同)。

(3) request bean。scope 取值为 request 的 bean 称为 request bean,request bean 的有效范围是用户请求的当前页面,存活时间是从用户的请求产生到请求结束。Tomcat 服务器分配给每个 JSP 页面的 request bean 是互不相同的。Tomcat 服务器对请求作出响应之后,取消分配给这个 JSP 页面的 request bean。简单地说,request bean 只在当前页面有效,直到响应结束。request bean 存活时间略长于 page bean 的存活时间,原因是 Tomcat 服务器认为页面执行完毕后,响应才算结束。需要注意的是,不同用户的 request bean 的也是互不相同的。也就是说,当两个用户同时请求同一个 JSP 页面时,一个用户对自己 request bean 属性的改变,不会影响到另一个用户。

(4) application bean。scope 取值为 application 的 bean 称为 application bean，application bean 的有效范围是当前 Web 服务目录下的各个页面，存活时间直到 Tomcat 服务器关闭。Tomcat 服务器为访问 Web 服务目录的所有用户分配一个共享的 bean，即不同用户的 application bean 也都是相同的一个。也就是说，任何一个用户对自己 application bean 属性的改变，都会影响到其他用户(不同 Web 服务目录的 application bean 互不相同)。

图 5.3 是 bean 的有效期的示意图，图 5.4 是有效范围示意图。

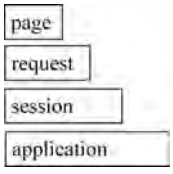


图 5.3 bean 的有效期

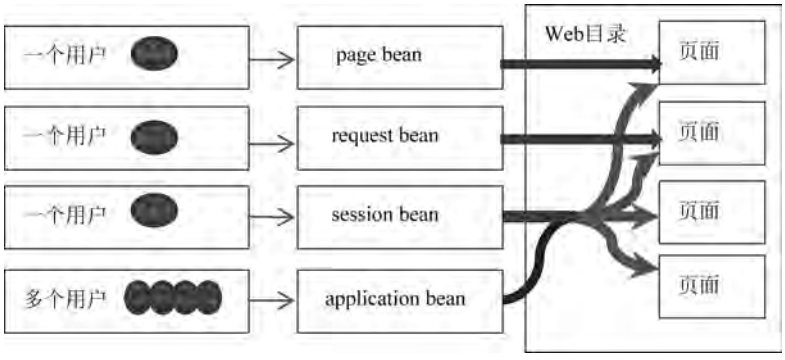


图 5.4 bean 的有效范围

注：当使用 session bean 时，要保证用户端支持 Cookie。

例 5_1 中负责创建 page bean 的类是上述的 Circle 类，page bean 的名字是 circle。

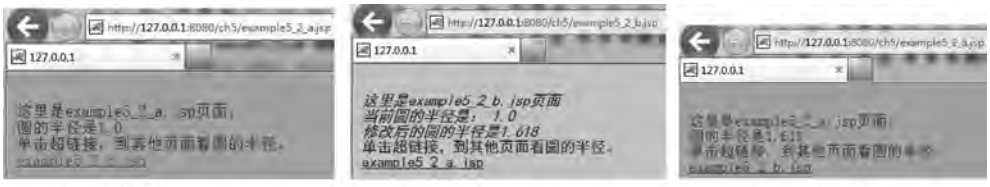
例 5_1

example5_1.jsp(效果如图 5.5 所示)

```
<%@ page contentType = "text/html" %>
<%@ page pageEncoding = "utf - 8" %>
<HTML><body bgcolor = #ffccff>
<style>
    #textStyle{
       font - family:宋体;font - size:36;color:blue
    }
</style>
<HTML><body bgcolor = #ffccff>
<p id = "textStyle">
<jsp:useBean id = "circle" class = "tom.jiafei.Circle" scope = "page" />
<% -- 通过 useBean 标记，获得名字是 circle 的 page bean -- %>
圆的初始半径是: <% = circle.getRadius() %>
<%   double newRadius = 100;
     circle.setRadius(newRadius);              //修改半径
 %>
<br>修改半径为<% = newRadius %>
<br><b>圆的半径是: <% = circle.getRadius() %>
<br><b>圆的周长是: <% = circle.circleLength() %>
<br><b>圆的面积是: <% = circle.circleArea() %>
</p></body></HTML>
```

图 5.5　page bean

例 5_2 使用 id 为 girl 的 session bean，创建 session bean 的类仍然是上述的 Circle.class。在例 5_2 的 example5_2_a.jsp 页面中，session bean 的半径 radius 的值是 1(如图 5.6(a)所示)，然后链接到 example5_2_b.jsp 页面，显示 session bean 的半径 radius 的值，然后将 session bean 的半径 radius 的值更改为 1.618(如图 5.6(b)所示)。用户再刷新 example5_2_a.jsp 或 example5_2_b.jsp 时看到的 session bean 的 radius 的值就都是 1.618 了(如图 5.6(c)所示)。



(a) 初次使用session bean　　(b) 修改bean的属性值　　(c) 同一个session bean

图 5.6　session bean

例 5_2

example5_2_a.jsp(效果如图 5.6(a)(c)所示)

```
<%@ page contentType = "text/html" %>
<%@ page pageEncoding = "utf - 8" %>
<HTML><body bgcolor = cyan>
<p style = "font - family:宋体;font - size:36;color:blue">
<% -- 通过 JSP 标记,用户获得一个 id 是 girl 的 session bean: -- %>
<jsp:useBean id = "girl" class = "tom.jiafei.Circle" scope = "session" />
<br>这里是 example5_2_a.jsp 页面.
<br>圆的半径是<% = girl.getRadius() %>
<br>单击超链接,到其他页面看圆的半径.
<a href = "example5_2_b.jsp"><br> example5_2_b.jsp </a>
</p></body></HTML>
```

example5_2_b.jsp(效果如图 5.6(b)所示)

```
<%@ page contentType = "text/html" %>
<%@ page pageEncoding = "utf - 8" %>
<HTML><body bgcolor = #ffccff>
<p style = "font - family:黑体;font - size:36;color:blue">
<% -- 用户的 id 是 girl 的 session bean: -- %>
<jsp:useBean id = "girl" class = "tom.jiafei.Circle" scope = "session"/>
<i><br>这里是 example5_2_b.jsp 页面
<br>当前圆的半径是: <% = girl.getRadius() %>
<% girl.setRadius(1.618);
```

```
%>
<br>修改后的圆的半径是:<% =girl.getRadius()%></i>
<br>单击超链接,到其他页面看圆的半径.
<a href="example5_2_a.jsp"><br>example5_2_a.jsp</a>
</p></body></HTML>
```

例5_3中使用了id为boy的application bean。当第一个用户访问这个页面时,显示application bean的radius的值,然后把application bean的radius的值更改为2.718(如图5.7(a)所示)。当其他用户访问这个页面时,看到的application bean的radius的值都是2.718(如图5.7(b)所示)。

(a) 第一个用户访问的效果　　(b) 后续用户访问的效果

图5.7　application bean

例5_3

example5_3.jsp(效果如图5.7(a)(b)所示)

```
<%@ page contentType="text/html"%>
<%@ page pageEncoding="utf-8"%>
<HTML><body bgcolor=#ffcfff>
<p style="font-family:宋体;font-size:36;color:blue">
<jsp:useBean id="boy" class="tom.jiafei.Circle" scope="application" />
圆的初始半径是:<% =boy.getRadius()%>
<% boy.setRadius(2.718);
%>
<br><b>修改后的圆的半径是:<% =boy.getRadius()%>
</p></body></HTML>
```

5.2　获取和修改bean的属性

使用useBean动作标记获得一个bean后,在Java程序片或表达式中bean就可以调用方法产生行为,如前面的例5_1～例5_3所示,这种情况下,不要求创建bean的类遵守setXxx和getXxx等规则(见5.1.1节)。获取或修改bean的属性还可以使用JSP动作标记getProperty、setProperty,这种情况下,要求创建bean的类遵守setXxx和getXxx等规则,当JSP页面使用getProperty、setProperty标记获取或修改属性xxx时,必须保证bean有相应的getXxx和setXxx方法,即对方法名字的命名有特殊的要求。下面讲述怎样使用JSP的动作标记getProperty、setProperty去获取和修改bean的属性。

▶ 5.2.1　getProperty动作标记


视频讲解

使用getProperty动作标记可以获得bean的属性值,并将这个值用串的形式发送给用户的浏览器。使用getProperty动作标记之前,必须使用useBean动作标记获得相应的bean。

getProperty 动作标记的语法格式是：

```
<jsp:getProperty name = "bean 的 id " property = "bean 的属性" />
```

或

```
<jsp:getProperty name = "bean 的 id "property = "bean 的属性">
</jsp:getProperty>
```

其中，name 取值是 bean 的 id，用来指定要获取哪个 bean 的属性的值，property 取值是该 bean 的一个属性的名字。

注：让 request 调用 setCharacterEncoding 方法设置编码为 UTF-8，以避免显示 bean 的属性值出现乱码现象。

▶ 5.2.2 setProperty 动作标记


视频讲解

使用 setProperty 动作标记可以设置 bean 的属性值。使用这个标记之前，必须使用 useBean 标记得到一个相应的 bean。

setProperty 动作标记可以通过两种方式设置 bean 属性值。

(1) 将 bean 属性值设置为一个表达式的值或字符序列。

```
<jsp:setProperty name = "bean 的 id " property = "bean 的属性"
value = "<% = expression %>"/>
<jsp:setProperty name = "bean 的 id " property = "bean 的属性"
value = "字符序列"/>
```

value 给出的值的类型要和 bean 的属性的类型一致。

(2) 通过 HTTP 表单的参数值来设置 bean 的相应属性值。

① 用 form 表单的所有参数值设置 bean 相对应属性值的使用格式如下：

```
<jsp:setProperty name = "bean 的 id 的名字"property = " * " />
```

在 setProperty 标记的上述用法中不具体指定 bean 属性值将对应 form 表单中哪个参数指定的值，系统会自动根据名字进行匹配对应，但要求 bean 属性的名字必须在 form 表单中有名称相同的参数名字相对应，Tomcat 服务器会自动将参数的字符串值转换为 bean 相对应的属性值

② 用 form 表单的某个参数的值设置 bean 的某个属性值的使用格式如下：

```
<jsp:setProperty name = "bean 的名字"property = "属性名"param = "参数名" />
```

setProperty 标记的上述用法具体指定了 bean 属性值将对应表单中哪个参数名(param)指定的值，这种设置 bean 的属性值的方法，不要求 property 给出的 bean 属性的名字和 param 给出的参数名一致，即不要求 bean 属性的名字必须和表单中某个参数名字相同。

当把字符序列设置为 beans 的属性值时，这个字符序列会自动被转化为 bean 的属性类型。Java 语言将字符序列转化为其他数值类型的方法如下。

- 转化到 int：Integer. parseInt(Sting s)

- 转化到 long：Long. parseLong(Sting s)
- 转化到 float：Float. parseFloat(Sting s)
- 转化到 double：Double. parseDouble(Sting s)

这些方法都可能发生 NumberFormatException 异常，例如，当试图将字符序列 ab23 转化为 int 型数据时就发生了 NumberFormatException。

注：用 form 表单设置 bean 的属性值时，只有提交了表单，对应的 setProperty 标记才会被执行。

例 5_4 使用 Goods 类创建 request bean。example5_4_a. jsp 通过 form 表单指定 example5_4_a. jsp 和 example5_4_b. jsp 中的 request bean 的 name 和 price 属性值。example5_4_a. jsp 和 example5_4_b. jsp 使用 getProperty 动作标记以及 bean 调用方法两种方式显示 request bean 的 name 和 price 属性值。

例 5_4

➢ **JavaBean**

将 Goods. java 保存在\ch5\WEB-INF\classes\tom\jiafei 中，用命令行进入 tom\jiafei 的父录 classes，按如下格式编译 Goods. java：

```
classes > javac tom\jiafei\Goods. java
```

Goods. java(负责创建 bean)

```
package tom. jiafei;
public class Goods  {
   String name = "无名";
   double price = 0;
   public String getName() {
      return name;
   }
   public void setName(String newName){
      name = newName;
   }
   public double getPrice() {
      return price;
   }
   public void setPrice(double newPrice) {
      price = newPrice;
   }
}
```

➢ **JSP 页面**

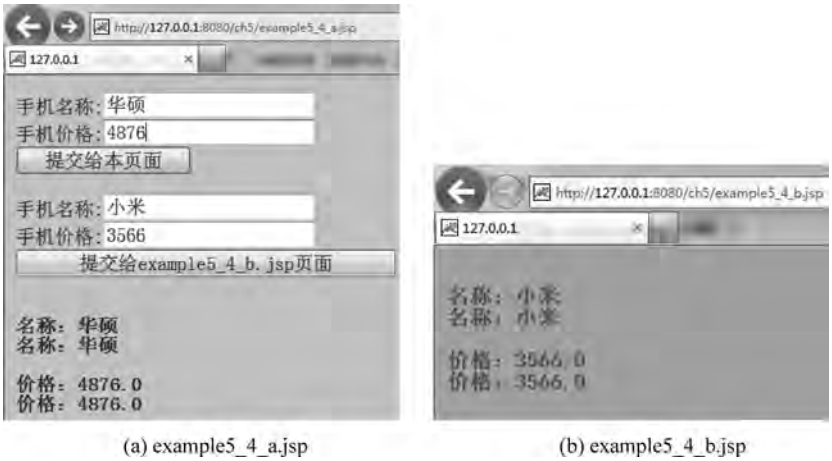
example5_4_a. jsp(效果如图 5. 8(a)所示)

```
<%@ page contentType = "text/html" %>
<%@ page pageEncoding = "utf - 8" %>
<% request. setCharacterEncoding("utf - 8");
 %>
< jsp:useBean id = "phone" class = "tom. jiafei. Goods" scope = "request"/>
< HTML >< body bgcolor = #ffccff >
```

```
<style>
    #textStyle{
       font-family:宋体;font-size:20;color:blue
    }
</style>
<p id="textStyle">
<form action="" Method="post" >
手机名称:<input type=text id=textStyle name="name">
<br>手机价格:<input type=text id=textStyle name="price"/>
<br><input type=submit id=textStyle value="提交给本页面"/>
</form>
<form action="example5_4_b.jsp" Method="post" >
手机名称:<input type= text name="name" id=textStyle>
<br>手机价格:<input type=text name="price" id=textStyle>
<br><input type=submit id=textStyle value="提交给 example5_4_b.jsp 页面">
</form>
<jsp:setProperty name="phone" property="name" param ="name" />
<jsp:setProperty name="phone" property="price" param ="price"/>
<br><b>名称:<jsp:getProperty name="phone" property="name"/>
<br><b>名称:<% = phone.getName() %><br>
<br><b>价格:<jsp:getProperty name="phone" property="price"/>
<br><b>价格:<% =phone.getPrice() %>
</p></body></HTML>
```

example5_4_b.jsp(效果如图 5.8(b)所示)

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding="utf-8" %>
<% request.setCharacterEncoding("utf-8");
 %>
<jsp:useBean id="phone" class="tom.jiafei.Goods" scope="page"/>
<HTML><body bgcolor= cyan>
<p style="font-family:黑体;font-size:20;color:red">
<jsp:setProperty name="phone" property="name" param ="name" />
<jsp:setProperty name="phone" property="price" param ="price"/>
```



(a) example5_4_a.jsp　　(b) example5_4_b.jsp

图 5.8　设置与获取 bean 的属性值

```
<br><b>名称：<jsp:getProperty name="phone" property="name"/>
<br><b>名称：<% = phone.getName() %><br>
<br><b>价格：<jsp:getProperty name="phone" property="price"/>
<br><b>价格：<% =phone.getPrice() %>
</p></body></HTML>
```

注：setProperty 和 getProperty 动作标记适合在处理数据和显示数据不是很复杂的情况下使用，如果业务逻辑或显示数据比较复杂，可能在 Java 程序片中用 bean 调用方法更加方便（创建 bean 的类也不用遵循方法命名的 setXXX 和 getXXX 规则）。

5.3 bean 的辅助类


视频讲解

在写一个创建 bean 的类时，除了需要用 import 语句引入 JDK 提供的类，可能还需要自己编写一些其他的类，只要将这样类的包名和 bean 类的包名一致即可（也可以和创建 bean 的类写在一个 Java 源文件中）。

在下面的例 5_5 中，使用一个 bean 列出 Tomcat 服务器驻留的计算机上某目录中特定扩展名的文件。创建 bean 的 ListFile 类，需要一个实现 FilenameFilter 接口的辅助类 FileExtendName，该类可以帮助 bean 列出指定扩展名的文件（把 ListFile.java 编译生成的字节码 ListFile.class 和 FileExtendName.class 复制到\ch4\WEB-INF\classes\tom\jiafei 中）。

例 5_5 使用 ListFile 类创建 request bean。例 5_5 中用户通过表单设置 request bean 的 extendsName 属性值，request bean 列出目录中由 extendsName 属性值指定的扩展名的文件。

例 5_5

➢ **JavaBean**

用命令行进入 tom\jiafei 的父目录 classes，编译 ListFile.java（约定见 5.1.2 节）：

```
classes > javac tom\jiafei\ListFile.java
```

ListFile.java（负责创建 bean）

```
package tom.jiafei;
import java.io.*;
class FileExtendName implements FilenameFilter {
   String str = null;
   FileExtendName (String s) {
      str = "." + s;
   }
   public boolean accept(File dir,String name) {
      return name.endsWith(str);
   }
}
public class ListFile {
   String extendsName = null;
   String [] allFileName = null;
   String dir = null;
```

```
    public void setDir(String dir) {
       this.dir  = dir;
    }
    public String getDir() {
       return dir;
    }
    public void setExtendsName(String s) {
       extendsName = s;
    }
    public String getExtendsName() {
       return extendsName;
    }
    public String [] getAllFileName() {
       if(dir!= null) {
         File mulu = new File(dir);
         FileExtendName help = new FileExtendName(extendsName);
         allFileName = mulu.list(help);
       }
       return allFileName;
    }
}
```

➢ **JSP 页面**

example5_5.jsp(效果如图 5.9 所示)

```
<%@ page contentType = "text/html" %>
<%@ page pageEncoding = "utf - 8" %>
<% request.setCharacterEncoding("utf - 8");
 %>
< style >
    # textStyle{
       font - family:宋体;font - size:36;color:blue
    }
</style >
< jsp:useBean id = "file" class = "tom.jiafei.ListFile" scope = "request"/>
< HTML >< body id = textStyle bgcolor = # ffccff >
< form action = "" Method = "post">
输入目录名(例如 D:/2000)< input type = text name = "dir" id = textStyle size = 15/>< br >
输入文件的扩展名(例如 java)
< input type = text name = "extendsName" id = textStyle size = 6 >
< input type = submit id = textStyle value = "提交"/>
</form >
< jsp:setProperty name = "file" property = "dir" param = "dir"/>
< jsp:setProperty name = "file" property = "extendsName" param = "extendsName"/>
< br >< b >目录 < jsp:getProperty name = "file" property = "dir"/>中
扩展名是 < jsp:getProperty name = "file" property = "extendsName"/> 的文件有:
<% String [] fileName = file.getAllFileName();
     if(fileName!= null) {
       for(int i = 0;i < fileName.length;i++) {
          out.print("< br >" + fileName[i]);
       }
     }
 %>
</body ></HTML >
```

图 5.9 特定扩展名的文件

5.4 JSP 与 bean 结合的简单例子

JSP 页面中调用 bean 可以将数据的处理从页面中分离出来，实现代码复用，以便更有效地维护一个 Web 应用。本节将结合一些实际问题，进一步熟悉掌握 bean 的使用方法。在本节中，创建 bean 类的包名都是 red.star，使用的 Web 服务目录仍然是 ch5，因此，需要在 ch5 目录下建立目录结构：ch5\WEB-INF\classes\red\star，将创建 bean 的字节码文件都保存在该目录中。为了调试程序方便，可以直接按照创建 bean 类的包名将相应的 Java 源文件保存在 Web 服务目录的相应目录中，例如将 Java 源文件保存在 Web 服务目录 ch5 的 WEB-INF\classes\red\star 目录中，然后使用 MS-DOS 命令行进入\red\star 的父目录 classes，按如下格式编译 Java 源文件：

```
classes>javac red\star\源文件名
```

5.4.1 三角形 bean


视频讲解

例 5_6 使用 request bean(Triangle 类负责创建)完成三角形的有关数据的处理。例子中的 JSP 页面提供一个 form 表单，用户可以通过 form 表单将三角形三边的长度提交给该页面。用户提交 form 表单后，JSP 页面将计算三角形面积的任务交给一个 request bean 去完成。

例 5_6

➢ **JavaBean**

用命令行进入 red\star 的父目录 classes，编译 Triangle.java(约定见 5.1.2 节)：

```
classes>javac red\star\Triangle.java
```

Triangle.java(负责创建 request bean)

```
package red.star;
public class Triangle {
    double sideA=-1,sideB=-1,sideC=-1;
    String area;
    boolean isTriangle;
    public void setSideA(double a) {
        sideA=a;
    }
    public double getSideA() {
```

```
        return sideA;
    }
    public void setSideB(double b) {
        sideB = b;
    }
    public double getSideB() {
        return sideB;
    }
    public void setSideC(double c) {
        sideC = c;
    }
    public double getSideC() {
        return sideC;
    }
    public String getArea() {
        double p = (sideA + sideB + sideC)/2.0;
        if(isTriangle){
            double result = Math.sqrt(p * (p - sideA) * (p - sideB) * (p - sideC));
            area = String.format("%.2f",result);        //保留 2 位小数
        }
        return area;
    }
    public boolean getIsTriangle(){
        if(sideA < sideB + sideC&&sideB < sideA + sideC&&sideC < sideA + sideB)
            isTriangle = true;
        else
            isTriangle = false;
        return isTriangle;
    }
}
```

➢ **JSP 页面**

example5_6.jsp(效果如图 5.10 所示)

```
<%@ page contentType = "text/html" %>
<%@ page pageEncoding = "utf - 8" %>
<style>
    #textStyle{
        font - family:宋体;font - size:36;color:blue
    }
</style>
<% request.setCharacterEncoding("utf - 8");
 %>
<jsp:useBean id = "triangle" class = "red.star.Triangle" scope = "request"/>
<HTML><body id = textStyle bgcolor = #ffccff>
<form action = "" method = "post" >
输入三角形三边:
边 A:<input type = text name = "sideA" id = textStyle value = 0 size = 5/>
边 B:<input type = text name = "sideB" id = textStyle value = 0 size = 5/>
边 C:<input type = text name = "sideC" id = textStyle value = 0 size = 5/>
<br><input type = submit id = textStyle value = "提交"/>
```

```
</form>
<jsp:setProperty name="triangle" property="*"/>
三角形的三边是:
<jsp:getProperty name="triangle" property="sideA"/>,
<jsp:getProperty name="triangle" property="sideB"/>,
<jsp:getProperty name="triangle" property="sideC"/>.
<br><b>这三个边能构成一个三角形吗?<jsp:getProperty name="triangle" property="isTriangle"/>
<br>面积是:<jsp:getProperty name="triangle" property="area"/></b>
</body></HTML>
```

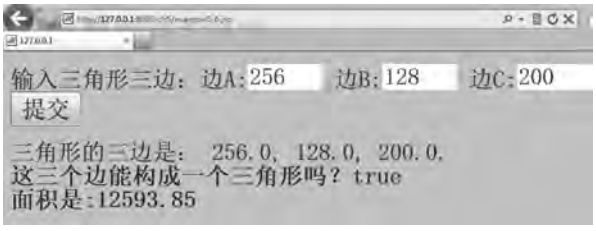


图 5.10 用 bean 计算三角形面积

▶ 5.4.2 四则运算 bean


视频讲解

例 5_7 使用 session bean(ComputerBean 类负责创建)完成四则运算。例子中的 JSP 页面提供一个 form 表单,用户可以通过 form 表单输入两个数,选择四则运算符号提交给该页面。用户提交 form 表单后,JSP 页面将计算任务交给 session bean 去完成。

例 5_7

➢ JavaBean

用命令行进入 red\star 的父目录 classes,编译 ComputerBean.java(约定见 5.1.2 节):

```
javac red\star\ComputerBean.java
```

ComputerBean.java(负责创建 session bean)

```
package red.star;
public class ComputerBean {
    double numberOne,numberTwo,result;
    String operator="+";
    public void setNumberOne(double n) {
        numberOne=n;
    }
    public double getNumberOne() {
        return numberOne;
    }
    public void setNumberTwo(double n) {
        numberTwo=n;
    }
    public double getNumberTwo() {
        return numberTwo;
    }
```

```
    public void setOperator(String s) {
        operator = s.trim();;
    }
    public String getOperator() {
        return operator;
    }
    public double getResult() {
        if(operator.equals("+"))
            result = numberOne + numberTwo;
        else if(operator.equals("-"))
            result = numberOne - numberTwo;
        else if(operator.equals("*"))
            result = numberOne * numberTwo;
        else if(operator.equals("/"))
            result = numberOne/numberTwo;
        return result;
    }
}
```

➢ **JSP 页面**

example5_7.jsp(效果如图 5.11 所示)

```
<%@ page contentType = "text/html" %>
<%@ page pageEncoding = "utf-8" %>
<style>
    #textStyle{
        font-family:宋体;font-size:36;color:blue
    }
</style>
<% request.setCharacterEncoding("utf-8");
 %>
<jsp:useBean id = "computer" class = "red.star.ComputerBean" scope = "session"/>
<HTML><body id = textStyle bgcolor = #ffccff>
<jsp:setProperty name = "computer" property = "*"/>
<form action = "" method = post>
    <input type = text name = "numberOne" id = textStyle size = 6/>
    <select name = "operator" id = textStyle>
        <option value = "+" id = textStyle>+
        <option value = "-" id = textStyle>-
        <option value = "*" id = textStyle>*
        <option value = "/" id = textStyle>/
    </select>
    <input type = text name = "numberTwo" id = textStyle size = 6/>
     <br><input type = "submit" value = "提交" id = textStyle "/>
</form>
<b>
<jsp:getProperty name = "computer" property = "numberOne"/>
<jsp:getProperty name = "computer" property = "operator"/>
<jsp:getProperty name = "computer" property = "numberTwo"/> =
<jsp:getProperty name = "computer" property = "result"/></b>
</body></HTML>
```

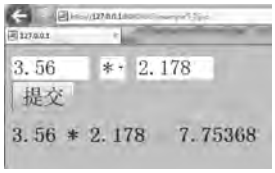


图 5.11 用 bean 完成四则运算

5.4.3 浏览图像 bean


视频讲解

例 5_8 中的 JSP 页面通过单击“下一张”或“上一张”超链接浏览图像，JSP 页面将获取图像名字的任务交给 session bean 去完成，JSP 页面根据 bean 获得的图像名字显示图像。例子中使用的图像文件保存在当前 Web 服务目录的子目录 image 中（图像文件的名字中不能含有空格）。

例 5_8

➢ JavaBean

用命令行进入 red\star 的父目录 classes，编译 Play.java（约定见 5.1.2 节）：

```
classes > javac red\star\Play.java
```

Play.java（负责创建 session bean）

```
package red.star;
import java.io.*;
public class Play {
    String pictureName[];              //存放全部图片文件名字的数组
    String showImage;                  //存放当前要显示的图片
    String webDir = "";                //Web 服务目录的名字，例如 ch5
    String tomcatDir;                  //Tomcat 的安装目录，例如 apache - tomcat - 9.0.26
    int index = 0;                     //存放图片文件的序号
    public Play() {
        File f = new File("");         //该文件认为在 Tomcat 服务器启动的目录中，即 bin 目录中
        String path = f.getAbsolutePath();
        int index = path.indexOf("bin");          //bin 是 Tomcat 的安装目录下的子目录
        tomcatDir = path.substring(0,index);      //得到 Tomcat 的安装目录的名字
    }
    public void setWebDir(String s) {
        webDir = s;
        File dirImage = new File(tomcatDir + "/webapps/" + webDir + "/image");
        pictureName = dirImage.list();
    }
    public String getShowImage() {
       showImage = pictureName[index];
       return showImage;
    }
    public void setIndex(int i) {
       index = i;
       if(index >= pictureName.length)
          index = 0;
       if(index < 0)
```

```
            index = pictureName.length - 1;
        }
        public int getIndex() {
            return  index ;
        }
}
```

➢ **JSP 页面**

example5_8.jsp(效果如图 5.12 所示)

```
<%@ page contentType = "text/html" %>
<%@ page pageEncoding = "utf - 8" %>
<style>
    #textStyle{
       font - family:宋体;font - size:36;color:blue
    }
</style>
<% request.setCharacterEncoding("utf - 8");
%>
<jsp:useBean id = "play" class = "red.star.Play" scope = "session" />
<%
    String webDir = request.getContextPath();      //获取当前 Web 服务目录的名称
    webDir = webDir.substring(1);                   //去掉名称前面的目录符号: /
%>
<jsp:setProperty  name = "play" property = "webDir" value = "<% = webDir %>"/>
<jsp:setProperty  name = "play" property = "index"  param = "index" />
<HTML><body bgcolor = cyan><p id = textStyle>
<image src =
image/<jsp:getProperty name = "play" property =
"showImage"/> width = 300 height = 200 ></image><br>
<a href = "?index = <% = play.getIndex() + 1 %>">下一张</a>
<a href = "?index = <% = play.getIndex() - 1 %>">上一张</a>
</p></body></HTML>
```

图 5.12　浏览图像

视频讲解

▶ 5.4.4　日历 bean

例 5_9 使用 session bean(Calendar 类负责创建)显示某月的日历。用户单击“下一月”“上一月”超链接可以翻阅日历。也可以输入年份,选择月份,单击 form 表单中

的提交键查看日历。

例 5_9

➢ **JavaBean**

用命令行进入 red\star 的父目录 classes，编译 Calendar.java(约定见 5.1.2 节)：

```
classes > javac red\star\Calendar.java
```

Calendar.java(负责创建 session bean)

```
package red.star;
import java.time.LocalDate;
import java.time.DayOfWeek;
public class Calendar {
      int year ,month ;
      String saveCalender;                                          //存放日历
      public Calendar(){
          year = LocalDate.now().getYear();
          month = LocalDate.now().getMonthValue();
      }
      public void setYear(int y){
          year = y;
      }
      public int getYear(){
          return year;
      }
      public void setMonth(int m){
          month = m;
          if(month > 12){
            year++;
            month = 1;
          }
          if(month < 1){
            month = 12;
            year -- ;
          }
      }
      public int getMonth(){
          return month;
      }
      public String getSaveCalender(){
          LocalDate date = LocalDate.of(year,month,1);
          int days = date.lengthOfMonth();                          //得到该月有多少天
          int space = 0;                                            //存放空白字符的个数
          DayOfWeek dayOfWeek = date.getDayOfWeek();                //得到 1 号是星期几
          switch(dayOfWeek) {
             case SUNDAY:     space = 0;
                              break;
             case MONDAY:     space = 1;
                              break;
             case TUESDAY:    space = 2;
                              break;
             case WEDNESDAY: space = 3;
                              break;
```

```
            case THURSDAY:   space = 4;
                             break;
            case FRIDAY:     space = 5;
                             break;
            case SATURDAY:   space = 6;
                             break;
        }
        String [] c = new String[space + days];
        for(int i = 0;i < space;i++)
            c[i] = " -- ";
        for(int i = space,n = 1;i < c.length;i++){
            c[i] = String.valueOf(n) ;
            n++;
        }
        String head =
        "< tr >< th >星期日</th >< th >星期一</th >< th >星期二</th >< th >星期三</th >" +
        "< th >星期四</th >< th >星期五</th >< th >星期六</th > </tr >";
        StringBuffer buffer = new StringBuffer();
        buffer.append("< table border = 0 >");
        buffer.append(head);
        int n = 0;
        while(n < c.length){
            buffer.append("< tr >");
            int increment = Math.min(7,c.length - n);
            for(int i = n;i < n + increment;i++) {
                buffer.append("< td align = center >" + c[i] + "</td >");
            }
            buffer.append("</tr >");
            n = n + increment;
        }
        buffer.append("</table >");
        saveCalender = new String(buffer);
        return saveCalender;
    }
}
```

➢ **JSP 页面**

example5_9.jsp(效果如图 5.13 所示)

```
< % @ page contentType = "text/html" % >
< % @ page pageEncoding = "utf - 8" % >
< style >
    # textStyle{
       font - family:宋体;font - size:18;color:blue
    }
</style >
< % request.setCharacterEncoding("utf - 8");
 % >
< HTML >< body id = textStyle bgcolor = # ffccff >
< jsp:useBean id = "calendar" class = "red.star.Calendar" scope = "session" />
< jsp:setProperty  name = "calendar" property = "year"   param = "year"/>
```

```
<jsp:setProperty name="calendar" property="month"param="month" />
<jsp:getProperty name="calendar" property="year" />年
<jsp:getProperty name="calendar" property="month" />月的日历:<br>
<jsp:getProperty name="calendar" property="saveCalender" />
<br><a href ="?month=<% =calendar.getMonth()+1 %>">下一月</a>
<a href ="?month=<% =calendar.getMonth()-1 %>">上一月</a>
<form action="" method=get>
输入年份<input type=text name="year"id=textStyle size=6 />
选择月份 <select name="month" id=textStyle size =1>
  <option value="1">1 月</option>
  <option value="2">2 月</option>
  <option value="3">3 月</option>
  <option value="4">4 月</option>
  <option value="5">5 月</option>
  <option value="6">6 月</option>
  <option value="7">7 月</option>
  <option value="8">8 月</option>
  <option value="9">9 月</option>
  <option value="10">10 月</option>
  <option value="11">11 月</option>
  <option value="12">12 月</option>
</select><br>
<input type="submit" value="提交" id=textStyle "/>
</form>
</body></HTML>
```

图 5.13 查看日历

▶ 5.4.5 计数器 bean


视频讲解

例 5_10 使用 application bean(ComputerCount 类负责创建)记录 Web 服务目录(通常所说的网站)被访问的次数。只要用户第 1 次访问(用户的 session 被创建)Web 服务目录,那么当前 Web 服务目录的访问计数就增加 1。如果用户的 session 没有消失,用户再访问当前 Web 服务目录,访问的计数不再增 1。

例 5_10

➢ JavaBean

用命令行进入 red\star 的父目录 classes,编译 ComputerCount.java(约定见 5.1.2 节):

```
classes>javac red\star\ComputerCount.java
```

ComputerCount.java(负责创建 application bean)

```
package red.star;
import java.io.*;
public class ComputerCount {
    int number = 0;
    public synchronized void addCount() {
        number++;
    }
    public int getNumber(){
        return number;
    }
}
import java.time.DayOfWeek;
```

➢ **JSP 页面**

example5_10_a.jsp(效果如图 5.14(a)所示)

```
<%@ page contentType = "text/html" %>
<%@ page pageEncoding = "utf-8" %>
<jsp:useBean id = "count" class = "red.star.ComputerCount"
                                    scope = "application"/>
<% if(session.isNew()) {
       count.addCount();
   }
%>
<HTML><body bgcolor = cyan>
<h1>这是网站的 example5_10_a.jsp 页面.
<br>网站访问量:<jsp:getProperty name = "count" property = "number"/>
<br>
<a href = "example5_10_b.jsp">欢迎去 example5_10_b.jsp 参观</a>
</body></HTML>
```

example5_10_b.jsp(效果如图 5.14(b)所示)

```
<%@ page contentType = "text/html" %>
<%@ page pageEncoding = "utf-8" %>
<jsp:useBean id = "count" class = "red.star.ComputerCount"
                    scope = "application"/>
<HTML><body bgcolor = #ffcfff>
<% if(session.isNew()) {
       count.addCount();
   }
%>
<h1>这是网站的 example5_10_b.jsp 页面
<br>网站访问量:
<jsp:getProperty name = "count" property = "number"/>
<br>
<a href = "example5_10_a.jsp">欢迎去 example5_10_a.jsp 参观</a>
</body></HTML>
```

(a) 一个用户看到的计数　　(b) 另一个用户看到的计数

图 5.14　两个用户访问 Web 服务目录

5.5　上机实验

视频讲解

提供了详细的实验步骤要求，按步骤完成，提升学习效果，积累经验，不断提高 Web 设计能力。

▶ 5.5.1　实验 1　小数表示为分数

❶ 实验目的

掌握怎样使用 request bean。

❷ 实验要求

(1) 编写 inputNumber.jsp，该页面提供一个 form 表单，该 form 表单提供一个 text 文本框，用于用户输入一个纯小数(例如 0.618)。用户在 form 表单中输入纯小数后，单击 submit 提交键将纯小数提交给 getFraction.jsp 页面。

(2) getFraction.jsp 使用 request bean，并使用 setProperty 动作标记让 request bean 将纯小数转换为分数，把分子和分母存放在 request bean 的名字是 numerator 和 denominator 的属性(变量)中。然后 getFraction.jsp 使用 getProperty 动作标记获取 request bean 的 numerator 和 denominator 的属性值。

(3) 在 Tomcat 服务器的 webapps 目录下(例如 D:\apache-tomcat-9.0.26\webapps)新建一个名字是 ch5_practice_one 的 Web 服务目录。把 JSP 页面都保存到 ch5_practice_one 目录中。在 ch5_practice_one 下建立子目录 WEB-INF(字母大写)，然后在 WEB-INF 下再建立子目录 classes。将创建 request bean 的类的 Java 源文件保存在 classes 的相应子目录中(见 5.1.2 节)。

(4) 用浏览器访问 JSP 页面 inputNumber.jsp。

❸ 参考代码

参考代码运行效果如图 5.15 所示。

(a) 输入纯小数　　(b) 得到分数

图 5.15　把纯小数表示成分数

JavaBean 用命令行进入 sea\water 的父目录 classes，编译 Fraction.java(约定见 5.1.2 节)：

```
classes > javac sea\water\Fraction.java
```

Fraction.java(负责创建 request bean)

```
package sea.water;
public class Fraction {
    public double number ;                                          //存放小数
    public long numerator ;                                         //存放分子
    public long denominator;                                        //存放分母
    public double getNumber(){
        String numberString = String.valueOf(number);
        String xiaoshuPart =
        numberString.substring(numberString.indexOf(".") + 1);     //得到纯小数部分
        return Double.parseDouble("0." + xiaoshuPart);
    }
    public long getNumerator(){
        return numerator;
    }
    public long getDenominator(){
        return denominator;
    }
    public void setNumber(double number){
        this.number = number;
        String numberString = String.valueOf(number);
        String xiaoshuPart =
        numberString.substring(numberString.indexOf(".") + 1);     //得到小数部分
        int m = xiaoshuPart.length();                               //m 的值就是小数的小数位数
        numerator = Long.parseLong(xiaoshuPart);                    //分子
        denominator = (long)Math.pow(10,m);                         //分母
        long greatCommonDivisor = f(numerator,denominator) ;        //最大公约数
        numerator = numerator/greatCommonDivisor;
        denominator = denominator/greatCommonDivisor;
    }
    private long f(long a,long b) {                                 //求 a 和 b 的最大公约数
      if(a == 0) return 1;
      if(a < b) {
         long c = a;
         a = b;
         b = c;
      }
      long r = a % b;
      while(r!= 0) {
         a = b;
         b = r;
         r = a % b;
      }
      return b;
   }
}
```

➢ **JSP 页面**

inputNumber.jsp(效果如图 5.15(a)所示)

```
<%@ page contentType="text/html"%>
<%@ page pageEncoding="utf-8"%>
<style>
    #tomStyle{
       font-family:宋体;font-size:36;color:blue
    }
</style>
<HTML><body bgcolor=#ffccff>
<form action="getFraction.jsp" id=tomStyle method=post>
输入一个纯小数:<br>
<input type=text name="number" id=tomStyle size=16 value=0.618 />
<br><input type="submit" id=tomStyle  value="提交" /><br>
看小数的分数表示.
</form>
</body></HTML>
```

getFraction.jsp (效果如图 5.15(b)所示)

```
<%@ page contentType="text/html"%>
<%@ page pageEncoding="utf-8"%>
<HTML><body bgcolor=cyan>
<p style="font-family:宋体;font-size:36;color:red">
<jsp:useBean id="fraction" class="sea.water.Fraction" scope="request" />
<jsp:setProperty name="fraction" property="number"param="number" />
<jsp:getProperty name="fraction" property="number" />
表示成分数是:<br>
<jsp:getProperty name="fraction" property="numerator" />
<jsp:getProperty name="fraction" property="denominator" />
</p></body></HTML>
```

5.5.2 实验 2 记忆测试

❶ 实验目的

掌握使用 session bean 存储用户的数据,和 4.6.5 节的记忆测试实验进行比对,体会使用 bean 的方便和好处。

❷ 实验要求

(1) 编写 choiceGrage.jsp,该页面中的 form 表单中使用 radio 标记选择记忆测试级别:初级、中级和高级。初级需要记忆一个长度为 5 个字符的字符序列(例如★■★▲●),中级需要记忆一个长度为 7 个字符的字符序列,高级需要记忆一个长度为 10 个字符的字符序列。在 choiceGrage.jsp 页面选择级别后,单击 form 表单的提交键提交给 giveTest.jsp 页面。

(2) 编写 giveTest.jsp 页面,该页面获取 choiceGrage.jsp 页面提交的级别后,使用 session bean 显示 testString 的属性值(例如属性值是长度为 7 个字符的字符序列),然后提示用户在 5 秒内记住这个字符序列。5 秒后,该页面将自动定向到 answerTest.jsp 页面。

(3) 编写 answerTest.jsp 页面,该页面的 form 表单提供用户给出答案的界面,即使用 radio 标记让用户选择字符序列中的各个字符,以此代表用户认为自己记住的字符序列。单击

提交键，将选择提交给 judgeAnswer.jsp 页面。

(4) 编写 judgeAnswer.jsp 页面，该页面负责判断有户是否记住了 giveTest.jsp 页面给出的字符序列。

(5) 在 Tomcat 服务器的 webapps 目录下新建名字是 ch5_practice_two 的目录，即新建 Web 服务目录 ch5_practice_two。把 JSP 页面都保存到 ch5_practice_two 目录中。在 ch5_practice_two 目录下建立子目录 WEB-INF(字母大写)，然后在 WEB-INF 目录下再建立子目录 classes，即在 ch5_practice_two 目录下建立\WEB-INF\classes 目录结构。将创建 session bean 类的 Java 源文件按照包名保存在 classes 的相应子目录中(见 5.1.2 节)。

(6) 用浏览器访问 JSP 页面 choiceGrage.jsp。

(7) 将本实验与 4.6.5 节的实验进行对比，感受使用 session bean 的好处。

❸ 参考代码

请比较 4.6.5 节的记忆测试实验，体会使用 bean 的方便和好处。参考代码运行效果如图 5.16 所示。

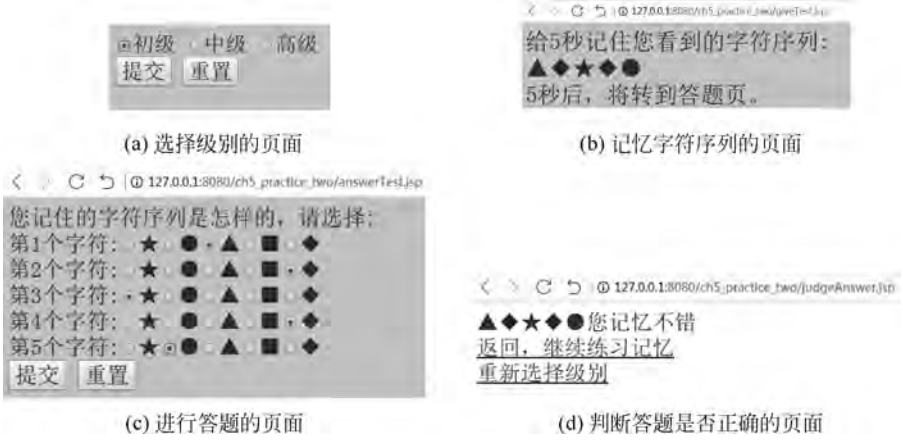


图 5.16 记忆测试

➢ **JavaBean**

用命令行进入 sea\water 的父目录 classes，编译 Memory.java(约定见 5.1.2 节)：

```
classes > javac sea\water\Memory.java
```

Memory.java(负责创建 session bean)

```
package sea.water;
import java.util.ArrayList;
import java.util.Random;
public class Memory {
    static ArrayList < String > list = new ArrayList < String >();
    static {
        list.add("★");
        list.add("●");
        list.add("▲");
        list.add("■");
        list.add("◆");
    }
```

```
    int grade = 5 ;                    //存放级别,例如初级 grade 存放的值是 5,中级是 7,高级是 10
    String testString;                 //存放需要记忆的字符序列,例如,★■★▲●
    boolean isGivenTestString = false;          //存放是否已经给了测试题目
    public void setGrade(int n){
        grade = n;
    }
    public int getGrade(){
        return grade;
    }
    public void giveTestString(){
        StringBuffer buffer = new StringBuffer();
        Random random = new Random();
        for(int i = 0;i < grade;i++) {
          int index = random.nextInt(list.size());
          String str = list.get(index);          //从 list 中得到一个字符,例如★
          buffer.append(str);
        }
        testString = new String(buffer);
    }
    public void setIsGivenTestString(boolean b){
        isGivenTestString  = b;
    }
    public boolean getIsGivenTestString(){
        return isGivenTestString;
    }
    public String getTestString(){
        return testString;
    }
}
```

➢ **JSP 页面**

choiceGrade.jsp(效果如图 5.16(a)所示)

```
<%@ page contentType = "text/html" %>
<%@ page pageEncoding = "utf-8" %>
<HTML><body bgcolor = #ffccff>
<style>
    #textStyle{
       font-family:宋体;font-size:26;color:blue
    }
</style>
<form action = "giveTest.jsp" id  = "textStyle" method = post >
<input type = radio name = "grade"value = "5" />初级
<input type = radio name = "grade"value = "7" checked = "ok" />中级
<input type = radio name = "grade"value = "10" />高级
<br><input type = "submit" id  = "textStyle" value = "提交"/>
<input type = "reset" id  = "textStyle" value = "重置" />
</form>
</body></HTML>
```

giveTest.jsp(效果如图 5.16(b)所示)

```
<%@ page contentType = "text/html" %>
<%@ page pageEncoding = "utf - 8" %>
<%@ page import = "java.util.ArrayList" %>
<%@ page import = "java.util.Random" %>
<jsp:useBean id = "memory" class = "sea.water.Memory" scope = "session" />
<HTML><body bgcolor = #ffccff>
<style>
    #tomStyle{
       font - family:宋体;font - size:36;color:blue
    }
</style>
<%   String grade = request.getParameter("grade");
     String testString  = "";                              //存放测试题目,例如★■★▲●
     if(grade  ==  null){
          memory.setGrade(memory.getGrade());
     }
     else {
          memory.setGrade(Integer.parseInt(grade));
     }
     if(memory.getIsGivenTestString() ==  false) {
          memory.giveTestString();
          testString = memory.getTestString();             //得到测试的题目
          memory.setIsGivenTestString(true);
     }
     else if(memory.getIsGivenTestString() ==  true){
          response.sendRedirect("answerTest.jsp");         //定向到答题页面
     }
%>
<p id = tomStyle>给 5 秒记住您看到的字符序列:<br>
<% = testString %>
<br>5 秒后,将转到答题页.
<%   response.setHeader("refresh","5");
%>
</p></body></HTML>
```

answerTest.jsp(效果如图 5.16(c)所示)

```
<%@ page contentType = "text/html" %>
<%@ page pageEncoding = "utf - 8" %>
<jsp:useBean id = "memory" class = "sea.water.Memory" scope = "session" />
<HTML><body bgcolor = #ffccff>
<style>
    #tomStyle{
       font - family:宋体;font - size:26;color:blue
    }
</style>
<form action = "judgeAnswer.jsp" id = tomStyle method = post>
您记住的字符序列是怎样的,请选择:
<%
  int n = memory.getGrade();
```

```
    memory.setIsGivenTestString(false);
for(int i=1;i<=n;i++){
       out.print("<br>第"+i+"个字符:");
       out.print
(      "<input type=radio id=tomStyle name=R"+i+" value='★'/>★"+
       "<input type=radio id =tomStyle name=R"+i+" value='●'/>●"+
       "<input type=radio id =tomStyle name=R"+i+" value='▲'/>▲"+
       "<input type=radio id =tomStyle name=R"+i+" value='■'/>■"+
       "<input type=radio id =tomStyle name=R"+i+" value='◆'/>◆");
   }
%>
<br><input type="submit"  id=tomStyle value="提交"/>
<input type="reset" id=tomStyle value="重置" />
</form>
</body></HTML>
```

judgeAnswer.jsp(效果如图 5.16(d)所示)

```
<%@ page contentType="text/html"%>
<%@ page pageEncoding="utf-8"%>
<jsp:useBean id="memory" class="sea.water.Memory" scope="session" />
<HTML><body bgcolor=white>
<p style="font-family:宋体;font-size:26;color:blue">
<%   memory.setIsGivenTestString(false);
     request.setCharacterEncoding("utf-8");
     int n=memory.getGrade();
     StringBuffer buffer=new StringBuffer();
     for(int i=1;i<=n;i++){
        buffer.append(request.getParameter("R"+i));          //获取 radio 提交的值
        out.print(""+request.getParameter("R"+i));
     }
     String userAnswer=new String(buffer);
     String testString=memory.getTestString();              //得到测试的题目
     if(testString.equals(userAnswer)){
         out.print("您记忆不错");
     }
     else {
         out.print("您没记忆住!答案是:<br>"+testString);
     }
%>
<br><a href="giveTest.jsp">返回,继续练习记忆</a>
<br><a href="choiceGrade.jsp">重新选择级别</a>
</p></body></HTML>
```

▶ 5.5.3 实验3 成语接龙

❶ 实验目的

掌握使用 application bean 存储所有用户共享的数据。

❷ 实验要求

(1) 编写 inputIdioms.jsp,该页面使用 application bean 显示目前成语接龙的信息。提供 form 表单,用户根据目前成语接龙的信息输入一个成语,单击提交键提交给当前页面,如果输

入成语符合成语接龙规则(例如输入的成语的首字和成语接龙中的最后一个成语的末字相同),application bean 就将用户输入的成语添加到成语接龙中。

(2) 在 Tomcat 服务器的 webapps 目录下新建名字是 ch5_practice_three 的 Web 服务目录。把 JSP 页面都保存到 ch5_practice_three 目录中。在 ch5_practice_three 下建立子目录 WEB-INF(字母大写),然后在 WEB-INF 目录下再建立子目录 classes,即在 ch5_practice_three 目录下建立\WEB-INF\classes 目录结构。将创建 application bean 的类的 Java 源文件按照包名保存在 classes 的相应子目录中(见 5.1.2 节)。

(3) 用浏览器访问 JSP 页面 inputIdioms.jsp。

❸ 参考代码

参考代码运行效果如图 5.17 所示。

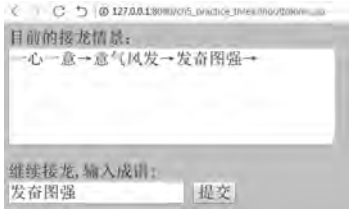


图 5.17 成语接龙

➢ **JavaBean**

用命令行进入 sea\water 的父目录 classes,编译 ContinueIdioms.java(约定见 5.1.2 节):

```
classes > javac sea\water\ ContinueIdioms.java
```

ContinueIdioms.java(负责创建 application bean)

```
package sea.water;
import java.util.LinkedList;
import java.util.Iterator;
import java.util.NoSuchElementException;
public class ContinueIdioms {
    LinkedList < String > listIdioms ;                    //存放成语的链表
    public String nowIdioms;                              //当前参与接龙的成语
    public ContinueIdioms(){
        listIdioms = new LinkedList < String >();
    }
    public synchronized void setNowIdioms(String s){
        nowIdioms = s;
        try{
           String previous = listIdioms.getLast();        //得到上次添加的成语
           //上一个成语的最后一个字符
           char endChar = previous.charAt(previous.length() - 1);
           char startChar = nowIdioms.charAt(0);          //当前成语的第一个字符
           if(startChar == endChar)
                   listIdioms.add(nowIdioms);
         }
         catch(NoSuchElementException exp){
            listIdioms.add(nowIdioms);
            System.out.println(exp);
```

```
            }
        }
        public String getAllIdioms(){
            StringBuffer buffer = new StringBuffer();
            Iterator<String> iterator  = listIdioms.iterator();
            if(iterator.hasNext()  ==  false)
               buffer.append("→");
            while(iterator.hasNext()){
               buffer.append(iterator.next() + "→");
            }
            return new String(buffer);
        }
}
```

➢ **JSP 页面**

inputIdioms.jsp（效果如图 5.17(a)(b)所示）

```
<%@ page contentType = "text/html" %>
<%@ page pageEncoding = "utf-8" %>
<jsp:useBean id = "idioms" class = "sea.water.ContinueIdioms"
                           scope = "application" />
<style>
#tomStyle{
      font-family:宋体;font-size:26;color:blue
   }
</style>
<%  request.setCharacterEncoding("utf-8");
 %>
<jsp:setProperty name = "idioms" property = "nowIdioms" param  = "nowIdioms" />
<HTML><body bgcolor = #ffccff>
<p id = tomStyle>
目前的接龙情景：<br>
<textArea id = tomStyle rows = 5 cols = 38>
<% = idioms.getAllIdioms() %>
</textArea><br>
<form  action = "" id = tomStyle method = post>
继续接龙,输入成语:<text  name = "nowIdioms" value = 10 />
<br><input type = "text" name  = "nowIdioms"  id = tomStyle />
<input type = "submit"  id = tomStyle value = "提交"/>
</form>
</p></body></HTML>
```

5.6 小结

- JavaBean 是一个可重复使用的软件组件，是遵循一定标准、用 Java 语言编写的一个类，该类的一个实例称作一个 JavaBean。
- 一个 JSP 页面可以将数据的处理过程指派给一个或几个 bean 来完成，我们在 JSP 页面中调用这些 bean 即可。在 JSP 页面中调用 bean 可以将数据的处理代码从页面中分离出来，实现代码复用，更有效地维护一个 Web 应用。
- bean 的生命周期分为 page、request、session 和 application。

习 题 5

1. 假设 Web 服务目录 mymoon 中的 JSP 页面要使用一个 bean，该 bean 的包名为 blue.sky。请说明，应当怎样保存 bean 的字节码文件。

2. 假设 Web 服务目录是 mymoon，star 是 mymoon 的一个子目录，JSP 页面 a.jsp 保存在 star 中，并准备使用一个 bean，该 bean 的包名为 tom.jiafei。下列哪个叙述是正确的？

A. 创建 bean 的字节码文件保存在\mymoon\WEB-INF\classes\tom\jiafei 中

B. 创建 bean 的字节码文件保存在\mymoon\star\WEB-INF\classes\tom\jiafei 中

C. 创建 bean 的字节码文件保存在\mymoon\WEB-INF\star\classes\tom\jiafei 中

D. 创建 bean 的字节码文件保存在\mymoon\WEB-INF\classes\start\tom\jiafei 中

3. tom.jiafei.Circle 是创建 bean 的类，下列哪个标记是正确创建 session bean 的标记？

A. `<jsp:useBean id = "circle" class = "tom.jiafei.Circle" scope = "page" />`

B. `<jsp:useBean id = "circle" class = "tom.jiafei.Circle " scope = "request" />`

C. `<jsp:useBean id = "circle" class = "tom.jiafei.Circle " scope = "session" />`

D. `<jsp:useBean id = "circle" type = "tom.jiafei.Circle " scope = "session" />`

4. 假设创建 bean 的类有一个 int 型的属性 number，下列哪个方法是设置该属性值的正确方法？

A.
```
public void setNumber(int n){
    number = n;
}
```

B.
```
void setNumber(int n){
    number = n;
}
```

C.
```
public void SetNumber(int n) {
    number = n;
}
```

D.
```
public void Setnumber(int n){
    number = n;
}
```

5. 假设 JSP 页面使用标记

```
<jsp:useBean id = "moon" class = "tom.jiafei.AAA" scope = "page"/>
```

创建了一个名字为 moon 的 bean，该 bean 有一个 String 类型、名字为 number 的属性。如果创建 moon 的 Java 类 AAA 没有提供 public String getNumber()方法，JSP 页面是否允许使用如下 getProperty 标记获取 moon 的 number 属性值？

```
<jsp:getProperty  name = "moon" property = "number"/>
```

6. 编写一个 JSP 页面，该页面提供一个表单，用户可以通过表单输入梯形的上底、下底和高的值，并提交给本 JSP 页面，该 JSP 页面将计算梯形面积的任务交给一个 page bean 去完成。JSP 页面使用 getProperty 动作标记显示 page bean 中的数据，例如梯形的面积。

7. 编写两个 JSP 页面 a.jsp 和 b.jsp，a.jsp 页面提供一个表单，用户可以通过表单输入矩形的两个边长提交给 b.jsp 页面，b.jsp 调用一个 request bean 去完成计算矩形面积的任务。b.jsp 页面使用 getProperty 动作标记显示矩形的面积。