**javaBean总结**

**第一节：简单的JavaBean**

Javabean本身就是一个类，属于java的面向对象编程。

Javabean中至少要存在一个无参构造方法。

范例：一个简单的javaBean

**package** myfirstweb;

**public** **class** SimpleJavaBean {

**private** String name;

**private** **int** age;

**public** SimpleJavaBean(){

System.***out***.println("-------------------一个新的实例化对象产生----------------------");

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

}

Lib文件夹一般存放第三方的jar文件。

WEB-INF文件夹的安全级别最高。

在这里需要说明的是，对于这样的简单JavaBean也有几个名词：

1. VO：值对象，存放所有的传递数据的操作上；
2. POJO:简单java对象
3. TO：传输对象，一个类必须实现Serializable接口才称为传输对象。

范例：第一种javaBean使用方法（古老）：

<?xml version=*"1.0"* encoding=*"GBK"* ?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns=*"http://www.w3.org/1999/xhtml"*>

<head>

<%@ page language=*"java"* contentType=*"text/html; charset=GBK"*

pageEncoding=*"GBK"*%>

<%@ page import = *"myfirstweb.\*"* %>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=GBK"* />

<title>第一个javabean</title>

</head>

<body>

<%

SimpleJavaBean simple = **new** SimpleJavaBean();

simple.setName("文攀");

simple.setAge(20);

%>

<h3>姓名：<%=simple.getName() %></h3>

<h3>年龄：<%=simple.getAge() %></h3>

</body>

</html>

范例：使用<jsp:useBean/>指令完成JavaBean的使用

<?xml version=*"1.0"* encoding=*"GBK"* ?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns=*"http://www.w3.org/1999/xhtml"*>

<head>

<%@ page language=*"java"* contentType=*"text/html; charset=GBK"*

pageEncoding=*"GBK"*%>

<jsp:useBean id = *"simple"* scope = *"page"* class = *"myfirstweb.SimpleJavaBean"*/> <!-- 使用jsp中的标签指令完成对javaBean的调用 -->

<!--id 表示要实例化对象的名称，scope表示Javabean的保存范围，class表示包.类名 用来替代new对象-->

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=GBK"* />

<title>第一个javabean</title>

</head>

<body>

<%

//实际上是利用反射生成一个名为ID的实例化对象。

simple.setName("文攀");

simple.setAge(20);

%>

<h3>姓名：<%=simple.getName() %></h3>

<h3>年龄：<%=simple.getAge() %></h3>

</body>

</html>

**<jsp:useBean/>指令其实是通过反射实现了一个对象的实例化。**

**当一个JavaBean修改之后一定要重新启动服务器之后才会出效果。**

**第二节：通过JavaBean自动设置与取得内容**

<jsp:setProperty name = “” property = “”/>标签可以完成所有的请求参数自动设置，而且可以自动的将输入的字符串内容转换为整型，或其他类型,name表示使用JavaBean中的ID的定义，property表示设置里面的属性内容。

使用这种操作，参数名称必须与属性名称一致。

范例：自动取得和设置内容

用户输入界面

<?xml version=*'1.0'* encoding=*'GBK'* ?>

<!DOCTYPE html PUBLIC '-//W3C//DTD XHTML 1.0 Strict//EN' 'http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd'>

<html xmlns=*'http://www.w3.org/1999/xhtml'*>

<head>

<meta http-equiv=*'Content-Type'* content=*'text/html; charset=GBK'* />

<title>javaBeanTest</title>

</head>

<body>

<form action = *"input\_bean01.jsp"* method = *"post"*>

姓名：<input type = *"text"* name = *"name"*></input><br></br>

年龄：<input type = *"text"* name = *"age"*></input><br></br>

<input type = *"submit"* value = *"提交"*></input>

<input type = *"reset"* value = *"重置"*></input>

</form>

</body>

</html>

表单提交后的页面（古老的方式）

<?xml version=*"1.0"* encoding=*"GBK"* ?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns=*"http://www.w3.org/1999/xhtml"*>

<head>

<%@ page language=*"java"* contentType=*"text/html; charset=GBK"*

pageEncoding=*"GBK"*%>

<%@ page import = *"myfirstweb.\*"* %>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=GBK"* />

<title>第一个javabean</title>

</head>

<%request.setCharacterEncoding("GBK"); %> <!-- 解决提交乱码 -->

<body>

<%

SimpleJavaBean simple = **new** SimpleJavaBean();

//实际上是利用反射生成一个名为ID的实例化对象。

simple.setName(request.getParameter("name")); //接收得到表单提交过来的表单内容参数名称必须与属性名称一致。即对应的文本框名称必须要跟JavaBean类里的属性名称一致。

simple.setAge(Integer.parseInt(request.getParameter("age")));

%>

<h3>姓名：<%=simple.getName() %></h3>

<h3>年龄：<%=simple.getAge() %></h3>

</body>

</html>

表单提交后的页面（简单的方式）

<?xml version=*"1.0"* encoding=*"GBK"* ?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns=*"http://www.w3.org/1999/xhtml"*>

<head>

<%@ page language=*"java"* contentType=*"text/html; charset=GBK"*

pageEncoding=*"GBK"*%>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=GBK"* />

<title>第一个javabean</title>

</head>

<%request.setCharacterEncoding("GBK"); %> <!-- 解决提交乱码 -->

<body>

<jsp:useBean id = *"simple"* scope = *"page"* class = *"myfirstweb.SimpleJavaBean"*/> <!-- 定义一个javaBean -->

<jsp:setProperty name = *"simple"* property = *"\*"*/> <!-- 用表单提交过来的内容自动为simple对象匹配内容 ，且是

按照控件名称和simple对象的属性名称来匹配的 .此方法可以自动将表单提交过来的字符串表示的年龄转换为int型-->

<h3>姓名：<jsp:getProperty property=*"name"* name=*"simple"*/></h3> <!-- 利用<jsp:getProperty>来代替getter方法 -->

<h3>年龄：<jsp:getProperty property=*"age"* name=*"simple"*/></h3>

</body>

</html>

**四种设置属性格式：**

**<jsp:setProperty name = “实例化对象名称” property = “\*”/>**

**<jsp:setProperty name = “实例化对象名称” property = “属性名称”/>**

**<jsp:setProperty name = “实例化对象名称” property = “属性名称” param = “参数名称”/>**

**<jsp:setProperty name = “实例化对象名称” property = “属性名称” value = “值”/>**

**取得属性格式：**

**<jsp:getProperty name = “实例化对象名称” property = “属性名称”/>**

以后只要在标签上出现了id属性，那么一定是表示一个实例化对象。如果看见了name肯定找的是属性范围中的内容。就是ID指定的，如果有了property则表示的就是类中的一个属性。