

Session 12

JSP and JavaBeans

1

Reading & Reference

■ Reading

- Head First Chapter 8
(pages 343-351)

- Optional reading

www.javaworld.com/javaworld/jw-01-2001/jw-0119-jspframe_p.html

© Robert Kelly, 2001-2009

2

Lecture Objectives

- Understand how to set up visibility to a Java bean from a JSP
- Understand how the useBean scope attribute determines where the bean handle can be found
- Understand how to access bean properties from a JSP
- Know how to handle special form elements (i.e., radio buttons, checkboxes and menus)

© Robert Kelly, 2001-2009

3

JSP Recap

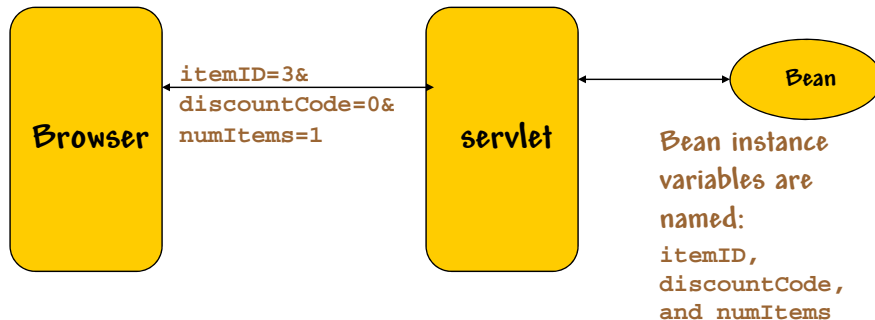
- JSP looks like HTML, but includes the ability to insert dynamic content
 - Java scriptlets (<% ...)
 - Directives (<%@ ...)
- However, mixing presentation with business logic is not a good approach to software structure, so

Don't use scriptlets in your JSP

© Robert Kelly, 2001-2009

4

Setting all Bean Values From the Form

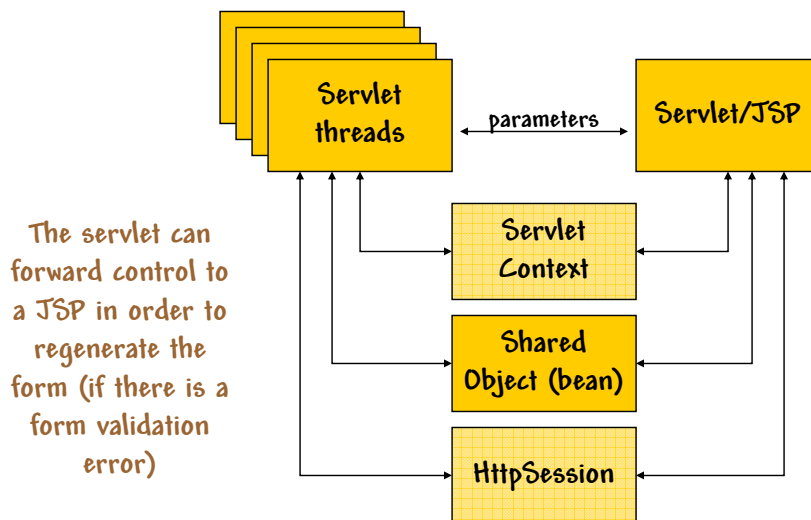


- A Web module (e.g., servlet) will usually read the form data set and set the values of the form in a bean so that they can be used by other Web modules

© Robert Kelly, 2001-2009

5

Servlet/JSP Data Sharing



The servlet can forward control to a JSP in order to regenerate the form (if there is a form validation error)

© Robert Kelly, 2001-2009

6

Model / View / Controller Pattern

- Web systems are usually decomposed into MVC components

- JSPs - view
- Servlets - controller
- Java Beans - model

Data for a Web application are usually stored in objects that are visible to servlets and JSPs

The handle to a bean is usually stored in the relevant shared object (e.g., session or ServletContext)

© Robert Kelly, 2001-2009

7

Bean Class Review

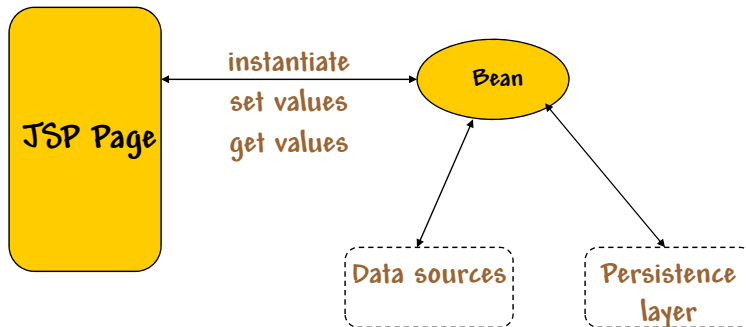
- A bean is an instance of a Java class that:
 - Must have a zero argument constructor
 - Should have no public instance variables
 - Should have get and set methods for any instance variables that are to be accessed (setter argument type and getter return type must be identical)
 - Should support persistence (the bean is serializable)
 - Usually supports events either by firing events when some properties change or listening for events

For use within JSPs, the property type should be either String or a primitive

© Robert Kelly, 2001-2009

8

How do we Access a Bean from a JSP?



- Before we do anything we need to get the handle to the bean

© Robert Kelly, 2001-2009

9

JSP Counter



© Robert Kelly, 2001-2009

10

CountBean

```
public class CountBean implements Serializable
{
    private int count = 0;

    public int getCount() {
        return (count);
    }

    public int fetchAndAdd() {
        int temp=count;
        count++;
        return (temp);
    }

    public void setCount(int newCount) {
        this.count = count;
    }
}
```

Notice that fetchAndAdd returns the pre-incremented value of the counter

Notice that the bean is a standard Java class, but contains the features of a bean (constructor, private instance variable, and properly named methods

© Robert Kelly, 2001-2009

11

Example-Counter (First try)

```
<html><head><title>JSP Counter</title>
<%@ page import="lectures.CountBean" %>
</head>
<body>
<%
    ServletContext sc = this.getServletContext();
    CountBean b = (CountBean) sc.getAttribute("b");
    if (b != null) {}
    else {
        b = new CountBean();
        sc.setAttribute("b", b);}
%>
<h1>JSP Counter</h1>
<p>This JSP will print and increment the value of the counter</p>
<p>The counter is initially: <%= b.getCount() %> </p>
<p>The counter is now: <%= b.fetchAndAdd() %> </p>
<p>The counter is now: <%= b.getCount() %> </p>
<br/>
<a href="http://localhost:8080/CodeCSE336/JSPCounter.jsp">
    Re-count</a>
</body></html>
```

But, we are not supposed to use scriptlets

© Robert Kelly, 2001-2009

12

Example-Counter (Second try)

```

<%@ page language="java" contentType="text/html" %>
<html>
<head>
<title>JSP Counter-2</title>
<%@ page import="lectures.CountBean" %>
</head>
<body>
  <jsp:useBean id="b" class="lectures.CountBean"
    scope="application" />
  <h1>JSP Counter</h1>
  <p>This JSP will print and increment the value of the counter</p>
  <p>The counter is initially: <%= b.getCount() %> </p>
  <p>The counter is now: <%= b.fetchAndAdd() %> </p>
  <p>The counter is now: <%= b.getCount() %> </p>
  <br/>
  <a href="http://localhost:8080/CodeCSE336/JSPs/JSPCounter2.jsp">
    Re-count</a>
</body>
</html>

```

This tag has the same effect as the scriptlet

Remember: application scope means ServletContext

These statements use the bean that was declared above

© Robert Kelly, 2001-2009

13

Generated Code

```

lectures.CountBean b = null;
synchronized (application) {
  b = (lectures.CountBean)
    __jspx_page_context.getAttribute("b",
    PageContext.APPLICATION_SCOPE);
  if (b == null){
    b = new lectures.CountBean2();
    __jspx_page_context.setAttribute("b", b,
    PageContext.APPLICATION_SCOPE);
  }
  out.write("\n");
  out.write("  <h1>JSP Counter</h1>\n");
  out.write("  <p>This JSP will print and increment the
value of the counter</p>\n");
  out.write("  <p>The counter is initially: ");
  out.print( b.getCount() );      out.write("</p>\n");
  out.write("  <p>The counter is now: ");
  out.print( b.fetchAndAdd() );  out.write("</p>\n");
  out.write("  <p>The counter is now: ");
  out.print( b.getCount() );
  ...
}

```

Notice that b is both a name in the PageContext attribute map and an identifier in the generated servlet

Instantiates the bean, if it does not already exist

© Robert Kelly, 2001-2009

14

Loading a Bean...

- To instantiate a bean (or bind an existing bean to your JSP variable):

```
<jsp:useBean id="b" class="lectures.CountBean" scope="application" />
```

If the bean does not already exist, this statement is equivalent (almost) to

```
<% lectures.CountBean b =  
    new lectures.CountBean(); %>
```

Bean class

The scope attribute specifies where a handle to the bean is stored

JSP variable name

© Robert Kelly, 2001-2009

15

JSP XML Syntax

- Bean access is an example of JSP action - JSP tags that transfer control, create and modify objects, etc. at run-time
- Syntax for action tags (e.g., bean usage) within JSPs is XML, so
 - Attribute names are case sensitive
 - Quotes must be used for attributes
 - Empty elements must use the empty tag syntax

```
<jsp:useBean id="name" ... />
```

© Robert Kelly, 2001-2009

16

XML Namespace

- A W3C scheme for building documents from fragments defined in different domains
<http://www.w3.org/TR/1999/REC-xml-names-19990114/>
- A single DTD is considered to own a namespace
- A namespace is an environment in which
 - All element names are unique
 - All attribute names are unique
- URLs are often used as a namespace identifier
- A qualified name for an element or attribute is a combination of namespace and element/attribute name
prefix:name

© Robert Kelly, 2001-2009

17

Obtaining a Handle to the Bean

```
<jsp:useBean id="b" class="lecturecode.CountBean" />
```

- `jsp:usebean` action specifies that a new object is instantiated if there is no existing bean with the same id (name) and scope, otherwise the existing bean is used

Use the `type` attribute if you want polymorphic access (i.e., access the bean as a superclass or interface)

© Robert Kelly, 2001-2009

18

Sharing Beans

- `jsp:useBean` has a scope attribute - defines where the handle to the bean is stored
- Scope values
 - `page` - stored in the `PageContext` object (this is the default scope)
 - `request` - stored in the `ServletRequest` object (very similar to page scope, but useful if the request is forwarded)
 - `session` - stored in the `HttpSession` object
 - `application` - stored in the shared `ServletContext`

© Robert Kelly, 2001-2009

19

Example - Counter (Third try)

```
<%@ page language="java" contentType="text/html" %>
<html><head>
<title>JSP Counter</title>
<%@ page import="lecturecode.CountBean" %>
</head>
<body>
  <jsp:useBean id="b" class="lectures.CountBean"
    scope="application" />
  <h1>JSP Counter</h1>
  <p>This JSP will print and increment the value of the counter</p>
  <p>The counter is initially:
    <jsp:getProperty name="b" property="count" /></p>
  <p>The counter is now:
    <%= b.fetchAndAdd() %> </p>
  <p>The counter is now:
    <jsp:getProperty name="b" property="count" /> </p>
  <br/>
  <a href="http://localhost:8080/CodeCSE336/JSPs/JSPCounter3.jsp">
    Re-count</a>
</body></html>
```

This is only slightly better than the use of a JSP expression

© Robert Kelly, 2001-2009

20

Getting Bean Properties

- Once a bean is initialized/bound, you can access its properties using:

```
<jsp:getProperty name="b" property="count" />
```

or

```
<%= b.getCount() %>
```

These 2 statements are equivalent if the shared scope contains an attribute (name/value pair) of "b"/b

Notice the difference in property name - the JSP variable is lower case, the accessor method uses a capital letter after the get

EL is actually a better approach to accessing bean properties, so we'll cover this material lightly

© Robert Kelly, 2001-2009

21

Setting Bean Properties

- You can set a bean's properties using

```
<jsp:setProperty name="b" property="count" value="336" />
```

or

```
<%= b.setCount(336) %>
```

- Value and name attributes are permitted to be request-time expressions

This can be a powerful programming approach

© Robert Kelly, 2001-2009

22

Dynamic Attribute Values

- Most attributes associated with JSP tags do not support dynamic values (i.e., values cannot be resolved at request time)
- However, the following tags contain attributes that can be resolved at request time
 - include
 - forward
 - useBean
 - setProperty, getProperty
 - param

© Robert Kelly, 2001-2009

23

Associating All Bean Properties

- You can access the form data set within a JSP - and set the bean attributes
- You can associate all bean properties with form data set values
- Association is based on the name
 - Name of form element must agree with bean property name
 - Example

```
<jsp:setProperty name="entry" property="*" />
```

However, for proper MVC design,
you would not access the form
parameters from the JSP

© Robert Kelly, 2001-2009

24

Conditional Bean Creation

- `jsp:useBean` results in a new bean being instantiated only if no bean with the same id and scope can be found
- If a matching bean is found, that bean is bound to the identifier
- In many situations, you can use conditional statements (only executed if the bean is instantiated)

```
<jsp:useBean ...  
    statements  
</useBean>
```

© Robert Kelly, 2001-2009

25

Example: Conditional Attributes

```
<jsp:useBean id="b" class="lecturecode.CountBean"  
    scope="application">  
    <jsp:setProperty name="b" property="count" value="336">  
</jsp:useBean>
```

What are the other ways you
could initialize the bean?

© Robert Kelly, 2001-2009

26

General Forms Processing

- Define form in HTML (or JSP)
 - Specify initial values of elements
- Transmit form to server
- Validate form content (within a bean method)
- Select page to be displayed, based on validation
 - If form content contains errors
 - Repopulate form with already entered values
 - Include error messages
 - Send form (as HTML) to browser
 - If no errors, display next page in sequence

© Robert Kelly, 2001-2009

27

Special Form Components

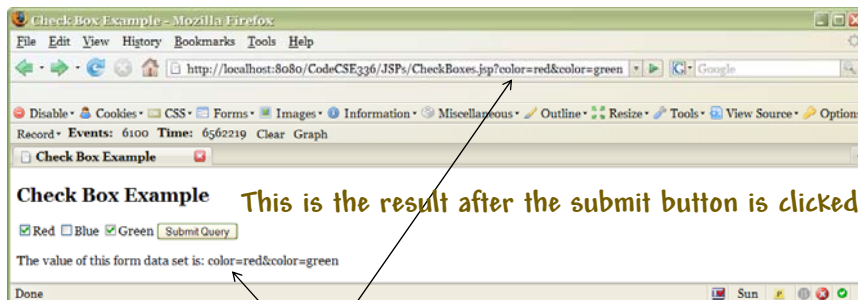
- Html forms for check boxes, radio buttons, and menus are allowed for selection
 - One element in a radio button group can be selected
 - A check box can be either selected or not
 - One or more members of a drop-down can be selected
- An attribute of selected or checked indicates which is selected
- The name/value of the selected component is included in the form data set (not the unselected components)

```
<input type="checkbox" value="13274"  
      checked name="part_updates" />  
Would you like to receive more info from our partners?
```

© Robert Kelly, 2001-2009

28

Example - Check Box



- Note the form dataset uses the check box value, not the browser display (i.e., "red" instead of "Red")
- The name/value of the selected component is included in the form data set (not the unselected components)

© Robert Kelly, 2001-2009

29

CheckBoxes.jsp

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html;
      charset=UTF-8">
    <title>Check Box Example</title>
  </head>
  <body><h2>Check Box Example</h2>
  <form
    action="http://localhost:8080/CodeCSE336/JSPs/CheckBoxes.jsp">
    <input type="checkbox" name="color" value="red"
      checked="checked" />Red
    <input type="checkbox" name="color" value="blue" />Blue
    <input type="checkbox" name="color" value="green"
      checked="checked" />Green
    <input type="submit">
  </form>
  <p>The value of this form data set is:
  <%=request.getQueryString() %> </p>
</body></html>
```

© Robert Kelly, 2001-2009

30

How Do You Set a Radio Button?

```
<input type='radio' name='credit' value='visa'  
<%= form.creditSelectionAttr("visa") %> >visa  
<input type='radio' name='credit' value='mc'  
<%= form.creditSelectionAttr("mc") %> >master  
card<br>  
<input type='radio' name='credit' value='disc'  
<%= form.creditSelectionAttr("disc") %> >discovery  
<input type='radio' name='credit' value='amex'  
<%= form.creditSelectionAttr("amex") %> >american  
express
```

Checked attribute is used in html check boxes and radio buttons

One of these method calls will return the 'checked="checked" ' string

© Robert Kelly, 2001-2009

31

FormBean.java

```
public class FormBean {  
    String name, comments = "Enter comments",  
    String credit;  
    public void setName(String s) { name = s; }  
    public String getName() {  
        return name != null ? name : ""; }  
    public void setComments(String s) { this.comments = s; }  
    public String getComments() { return comments; }  
    public void setCredit(String s) { credit = s; }  
    public String getCredit() {  
        return credit != null ? credit : ""; }  
    }  
    public String creditSelectionAttr(String creditName) {  
        if(credit != null) {  
            return credit.equals(creditName) ? "checked" : ""; }  
        }  
        return "";  
    }  
}
```

© Robert Kelly, 2001-2009

32

Form - Drop Down Components

- Drop down components are created with HTML `select` and `option` elements, as in:

```
<select multiple size="3" name="industry">
  <option value="none" selected="selected"> none
  selected...
</option>
  <option value="2008"> Wireless </option>
  <option value="2009"> Home </option>
  <option value="2010"> DTV </option>
  <option value="2011"> Financial </option>
</select>
```

- If DTV is selected, the form data set will contain:

```
industry=2010
```

© Robert Kelly, 2001-2009

33

Have You Achieved the Lecture Objectives?

- Understand how to set up visibility to a Java bean from a JSP
- Understand how the `useBean` scope attribute determines where the bean handle can be found
- Understand how to access bean properties from a JSP
- Know how to handle special form elements (i.e., radio buttons, checkboxes and menus)

© Robert Kelly, 2001-2009

34

Assignment 4a

- Project 1 - Mets Form Processing
- Regenerate the form with the values contained in your Java bean
 - Populate the fields with values from the bean
 - Include error messages for elements that failed validation
 - Add a congratulations message to form if all the data is correct (and complete)
- For now, you can invoke the JSP with a redirect from the servlet (later you will use a servlet forward)
- Hint: for testing, it will be good to use two versions of the form: HTML and JSP