Last update: 4 April 2009

JSP Custom Tags Lecture

Today we will look at custom tags.

JSTL Class Demo

Purpose: The purpose of this demo is to get hands-on experience with a typical JSTL action tag, the <c:forEach> tag to support looping without scripting. When we work with JSF we will rarely use JSTL tags. Instead we will use special JSF action tags. So, the important point of this demo is not so much to really learn how to use the <c:forEach> tag, but just to see a tag in use, which will then make it easier to understand the rest of the lesson on how to write the code behind a tag.

Here is how to create a table using the JSTL forEach custom tag

0. Create a web application project name ForEachDemo

1. Right-click the Libraries folder in the ForEachDemo project, choose Add Library

2. Add JSTL 1.1

3. Create a JSP page named table.jsp
Key tag lines are in red

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@ taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core' %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<table>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Lecture 5</title>
</head>
<body>
<table>

        <c:forEach var="student" items="${students}">
                <tr>
                        <td>${student.name}</td>
                        <td>${student.age}</td>
```

```
                </tr>
        </c:forEach>

</table>
</body>
</html>
```

4. Create a Student bean. Note that it has two constructors. One of them has no parameters
so it mets the requires of being a bean. The second one is used for convenience. See below

```
package net.mumde.cs545;

public class Student
{
        private String name;
        private int age;

        public Student()
        {
        }

        public Student(String name, int age)
        {
                this.name = name;
                this.age = age;
        }

        public void setName(String name)
        {
                this.name = name;
        }

        public String getName()
        {
                return name;
        }


        public void setAge(int age)
        {
                this.age = age;
        }

        public int getAge()
        {
                return age;
        }
}
```

5. Create a servlet named TableServlet.
Line relating to the custom tag are in red.

```
package net.mumde.cs545;

import javax.servlet.*;
import javax.servlet.http.*;
```

```
import java.io.*;

public class TableServlet extends HttpServlet
{
        static final long serialVersionUID = 0;

        public void doGet(HttpServletRequest request, HttpServletResponse
response)
                throws IOException, ServletException
        {

                Student[] table = new Student[]
                {
                                new Student("bob", 23),
                                new Student("jill", 33),
                                new Student("kim", 18)
                };

                request.setAttribute("students", table);

                RequestDispatcher dispatcher =
request.getRequestDispatcher("table.jsp");
                dispatcher.forward(request, response);
        }

        public void doPost(HttpServletRequest request, HttpServletResponse
response)
        throws IOException, ServletException
        {
                RequestDispatcher dispatcher =
request.getRequestDispatcher("table.jsp");
                dispatcher.forward(request, response);
        }
}

6. Add the following link to index.jsp
<a href='TableServlet'>View student table</a>
```

Creating a Simple tag handler

Here is how to write a custom Label tag that is reminiscent of the one in ASP.NET.

Purpose: This is a hello-world example of how to write a custom tag. It shows the basic steps involved in writing custom tags.

```
<aspx:Label foreColor='red' text='Hello'/>
the above will generate the following HTML:
<span style='color:red'>Hello</span>
```

1. Create a new Tag Library Descriptor named TldDemo.tld. Accept defaults. [Note: in Netbeans, this means create a new TLD file, and name it TldDemo, NOT TldDemo.tld. Netbeans will add the .tld suffix.] To

inspect the tld you created, look in WEB-INF/tlds (Netbeans might automatically open this for you.)

2. Create a new java class named Label that will be your tag handler class. Copy the following code into Label.java

```
package net.mumde.cs545;
import java.io.IOException;

import javax.servlet.jsp.*;
import javax.servlet.jsp.tagext.*;

public class Label extends SimpleTagSupport
{
        String foreColor;
        String text;

//This is called from JSP servlet to render custom tag
        public void doTag() throws JspException, IOException
        {
                JspWriter out = getJspContext().getOut();
                if (foreColor != null)
                        out.write(String.format("<span
style='color:%s'>%s</span>", foreColor, text));
                else
                        out.write(String.format("<span>%s</span>",
text));
        }

//Need a setter for each attribute of custom tag

        public void setForeColor(String foreColor)
        {
                this.foreColor = foreColor;
        }

        public void setText(String text)
        {
                this.text = text;
        }
}
```

3. Insert the following <tag> element into the tld file you created in step 1:

```
  <tag>
    <description>Generates a label</description>
    <name>Label</name>
    <tag-class>net.mumde.cs545.Label</tag-class>
    <body-content>empty</body-content>
    <attribute>
      <name>foreColor</name>
      <required>false</required>
      <rtexprvalue>true</rtexprvalue>
    </attribute>
    <attribute>
```

```
      <name>text</name>
      <required>true</required>
      <rtexprvalue>true</rtexprvalue>
    </attribute>
  </tag>
```

4. Create a JSP file named label.jsp that uses the tag

[Note: Netbeans might give you a cannot load Label file error until after you have run the jsp file.]

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@ taglib prefix='aspx' uri='/WEB-INF/tlds/TldDemo'%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>JSTL and Custom Tags Lecture</title>
</head>
<body>

<aspx:Label foreColor='red' text='hello'/>

</body>
</html>
```

5. Right-click label.jsp in the projects window and run it.