Struts Code Peaces – <html:radio> element

We explain the struts <html:radio> element and illustrate the usage with some small examples.

Generals

Author:
Sascha Wolski
Sebastian Hennebrueder
http://www.laliluna.de/tutorials.html – Tutorials for Struts, EJB, xdoclet and eclipse.

Date:
February 25th 2005

The <html:radio> element

It renders a HTML <radio> Element. You assign a property of the form-bean to this element. <*html:radio*> elements with the same property will be grouped. It is possible to use a <*html:radio*> element inside an iteration.

The following example shows the source code of the JSP file:

```
<html:radio property="selectedItem" value="Maria" />
<html:radio property="selectedItem" value="Klaus" />
```

The following HTML source code will be rendered.

```
<input type="radio" name="selectedItem" value="Maria">
<input type="radio" name="selectedItem" value="Klaus">
```

Attribute des <html:radio> Elements

Now the most important attribute will be explained. You find a complete list of all available attributes for this tag in the API of the HTML tag library.

http://struts.apache.org/userGuide/struts-html

Name	Beschreibung
disabled	If „true“ disable this element
name	Name of the bean which contain the properties.
property	Name der Eigenschaft oder des Requestparameters das dem Element zugewiesen werden soll.
value	Specify the value, which is submitted to the server.
IdName	Name of the bean which holds the properties. Usually exposed by an iterator. When the idName attribute is present, the value attribute is used as the name of the property on the idName bean that will return the value of the radio tag for this iteration

Usage of <html:radio> element

We illustrate the usage of the <*html:radio*> element with some examples. Create a new project with an action class, an action form class and a JSP file.

Create an object class

Create a new java class customer in the package *de.laliluna.tutorial.radio.object*. This class represents a customer.

Create two properties, *id* of type int and *name* of type String and provide a getter and setter method for each of this properties.

Define a constructor which allows you to set the both properties if you initialilze the class.

The object class looks like the following:

```
public class Customer {

      private int id;
      private String name;

      public Customer(){}

      public Customer(int id, String name){
            this.id = id;
            this.name = name;
      }

      public int getId() {
            return id;
      }
      public void setId(int id) {
            this.id = id;
      }
      public String getName() {
            return name;
      }
      public void setName(String name) {
            this.name = name;
      }
}
```

Create a new FormBean

Create a new action form class *ExampleForm* in the package *de.laliluna.tutorial.option.form*.

Add two properties, selectedItem and selectedItemIterate of type String. This properties hold the value of the associated *<html:radio>* elements.

Add a getter and setter method for each property.

Implement the *reset()* method of the action form class and initialize the properties.

```
public class ExampleForm extends ActionForm {

      private String selectedItem;
      private String selectedItemInterate;

      public String getSelectedItemInterate() {
            return selectedItemInterate;
      }
      public void setSelectedItemInterate(String selectedItemInterate) {
            this.selectedItemInterate = selectedItemInterate;
      }
      public String getSelectedItem() {
            return selectedItem;
      }
      public void setSelectedItem(String selectedItem) {
            this.selectedItem = selectedItem;
      }

      /**
       * Reset method
       * @param mapping
       * @param request
       */
      public void reset(ActionMapping mapping,
                                  HttpServletRequest request) {

            //initial properties
            this.selectedItem = "1";
            this.selectedItemInterate = "2";

      }
}
```

Create a new action class

Create a new class *ExampleAction* in the package *de.laliluna.tutorial.option.action.*
Provide a collection of customers to illustrate the usage of *<html:radio>* elements nested inside a *<html:iteration>* element.

```
public ActionForward execute(
            ActionMapping mapping,
            ActionForm form,
            HttpServletRequest request,
            HttpServletResponse response) {

      ExampleForm selectForm = (ExampleForm) form;

      //define a dummy collection
      Collection customers = new ArrayList();
      customers.add(new Customer(1, "Marie"));
      customers.add(new Customer(2, "Klaus"));
      customers.add(new Customer(3, "Peter"));

      //set the collection in the request
      request.setAttribute("customers", customers);

      return mapping.findForward("success");
}
```

Create the struts-config.xml

Open the struts-config.xml and define the form bean and the action mapping.

```
<struts-config>
   <form-beans>
      <form-bean name="exampleForm"
type="de.laliluna.tutorial.option.form.ExampleForm" />
   </form-beans>

   <action-mappings>
      <action
         name="exampleForm"
         path="/example"
         scope="request"
         type="de.laliluna.tutorial.option.action.ExampleAction">
         <forward name="success" path="/form/example.jsp" />
      </action>
   </action-mappings>
</struts-config>
```

Create a JSP file

Create a JSP file *example.jsp* in the folder */WebRoot/form/*

Open the JSP file and add the following HTML source code.

```
<%@ page language="java"%>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-bean" prefix="bean"%>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-html" prefix="html"%>
<%@ taglib uri="http://jakarta.apache.org/struts/tags-logic" prefix="logic"%>

<html>
      <head>
            <title>example.jsp</title>
      </head>
      <body>
            <html:form action="/example">

            .... sample code ...

            </html:form>
      </body>
</html>
```

Add the first example inside the *<html:form>* element.

Example 1

In the first example we use three *<html:radio>* elements, which are associated with the same property *selectedItem* of the form-bean. All *<html:radio>* elements with the same associated property will be grouped. The attribute *value* is sent to the server and is assigned to the associated property.

```
<h4>Simple use &lt;html:radio&gt; Tag</h4>

Maria <html:radio property="selectedItem" value="1" /> <br />
Klaus <html:radio property="selectedItem" value="2" /> <br />
Peter <html:radio property="selectedItem" value="3" /> <br />

<html:submit/>
```

Example 2

The next example shows the usage of an <html:radio> element nested inside an iteration. We use the collection of customers, we have set in the request before. With the attribute *property* of the *<html:radio>* element we assign the property *selectedItem* of the form bean. All elements inside the iteration with the same associated *property* will be grouped. The attribute *idName* specifes the current element of the iteration, in our case it is *var. It* contains the current customer of the iteration. The attribute *value* specifies the property of this bean which holds the value. This value is transferred to the server when the form is submitted.

```
<h4>Use &lt;html:radio&gt; Tag within an iteration</h4>

<logic:iterate name="customers" id="var">
      <bean:write name="var" property="name" />
      <html:radio property="selectedItemInterate" idName="var" value="id" />
      <br />
</logic:iterate>

<html:submit/>
```

Now you can test the project. Call the project with the following link.

http://localhost:8080/RadioTag/example.do