Struts Code Pieces – ValidatorForm

This tutorial explains the ValidatorForm class using a working example.

Generals

Autor:
Sebastian Hennebrüder
http://www.laliluna.de/tutorials.html – Tutorials for Struts, EJB, xdoclet und eclipse.

Datum:
February, 16th 2005

Dependencies
Struts 1.1
Jboss, Tomcat, Jetty etc
PDF download: http://www.laliluna.de/download/struts-validator-form-en.pdf
Source download: http://www.laliluna.de/download/struts-validator-form-source.zip

ValidatorActionForm class

The form bean class ValidatorForm extends the ActionForm class.

It has the capacity to validate fields of a form using validation rules which are defined in an XML file.

Example:

```
public class ExampleForm extends ValidatorForm {}
```

Define a name for the form bean class in the Struts Config.

Example:

```
<form-beans >
      <form-bean name="exampleForm" type="my.package.ExampleForm" />
</form-beans>
```

The form bean can than be reused in an action definition. Below you see an example for an ActionMapping in the Struts Config.

Example:

```
<action attribute="exampleForm"
        input="/form/example.jsp"
        name="exampleForm"
        path="/example"
        scope="request"
        type="my.package.ExampleAction" />
```

Validation of properties

The form bean ValidatorForm uses the Struts validation capabilities to use validation rules defined in XML files. Struts offers a wide choice of rules, you can all find in the file validator-rules.xml.

You configure the rules for each property of a FormBean. These validations have to be written in the XML file (validation.xml)

Example validation file validation.xml:

```
<form-validation>
 <formset>
        <!-- validation mapping für example form -->
        <form name="exampleForm">
            <field
                  property="name"
```

```
                depends="required, minlength">
                  <arg0 key="exampleForm.name" />
                  <arg1 key="${var:minlength}" resource="false" />
                          <var>
                            <var-name>minlength</var-name>
                            <var-value>3</var-value>
                          </var>
            </field>
       </form>
    </formset>
</form-validation>
```

Initializing the properties

The ValidatorForm class offers a reset method which is called automatically by Struts. In this method you can initialize properties.

Example:

```
public void reset(ActionMapping mapping,
                  HttpServletRequest request) {

       //Initialisieren der Eigenschaft text
       text = "Hello World";
}
```

Working example of a ValidatorForm Beans

We will show you the usage of an ValidatorForm Bean using a simple working example.

Creating a ValidatorActionForm class

Create a bean form class *ExampleForm* in a package.
(Example*: de.laliluna.tutorials.validatorform.form*).

The class extends the *ValidatorForm*.

Create two properties name of type String and age of type int.

Create getters and setters for your properties.

The class looks like the following.

```
public class ExampleForm extends ValidatorForm {

      //Eigenschaften der Klasse
      private String name;
      private int age;

      //Getter und Setter Methoden
      public int getAge() {
            return age;
      }
      public void setAge(int age) {
            this.age = age;
      }
      public String getName() {
            return name;
      }
      public void setName(String name) {
            this.name = name;
      }
}
```

Creating an Action class

Create a new class of type ExampleAction in the package *de.laliluna.tutorial.validatorform.action.*

The class extends the Action class of Struts.

Implement the *execute(..)* method and output the variables.

The following is the source code of the class *ExampleAction*

```
public class ExampleAction extends Action {

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

 //cast to ValidatorForm
 ExampleForm exampleForm = (ExampleForm) form;

 //accessing the properties of ValidatorForm
 //Klasse innerhalb der Action Klasse
 System.out.println(exampleForm.getName());
 System.out.println(exampleForm.getAge());

 return mapping.findForward("success");
 }

}
```

Creating a JSP

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

Below you can see the 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"%>

<html>
      <head>
            <title>JSP for exampleForm</title>
      </head>
      <body>
            <html:form action="/example">
                  <html:errors />
                  Name: <html:text property="name" /> <br>
                  Age: <html:text property="age" /> <br>
                  <html:submit value="Send"/>
            </html:form>
      </body>
</html>
```

Configure the form bean (struts-config.xml)

Open the *struts-config.xml* and add your form bean definition between the *<form-bean>,*

The attribute *name* defines the name of the form bean. Other action can refer to the form bean using this name.

type defines the form bean class. In our case it is ValidatorForm.

```
<form-beans >
      <form-bean name="exampleForm"
```

```
type="de.laliluna.tutorial.validatorform.form.ExampleForm" />
</form-beans>
```

Creating action mapping (struts-config.xml)

We will create an action mapping to show the usage of the ValidatorForm class. Both actions will use the same form bean.

As we are lazy we will also use the same action class *ExampleAction*.

Create the action mappings in between the <*action-mappings*> tags. Use the form bean *exampleForm* and forward to the JSP files.

The action mapping has the path */example.*

name is the name of the form bean

input defines the JSP which is shown when an error occurred during the validation. In our case this is the same JSP we used to show the form.

type defines the action class which is called for the action

<forward ...> defines the forwards of the action. Here the JSP file is *example.jsp.*

```
<action-mappings>
      <action
         attribute="exampleForm"
         input="/form/example.jsp"
         name="exampleForm"
         path="/example"
         scope="request"
         type="de.laliluna.tutorial.validatorform.action.ExampleAction">

            <forward name="showExample" path="/form/example.jsp" />

      </action>
</action-mappings>
```

Initializing the ValidatorForm class

Add a reset method in the **ValidatorForm** class *ExampleForm* . Initialize the form properties in this method.

```
public void reset(ActionMapping mapping,
      HttpServletRequest request) {

      //Initialisieren der Eigenschaften
      name = "Adam Weisshaupt";
      age = 23;
}
```

Validating properties with XML validation rules

To validate the user input, if a name's length is greater than 3 character or the age is between 0 and 150, you have to configure this validations in an XML file.

Create the XML file *validation.xml* in the directory */WebRoot/WEB-INF/*.

<form name=".."> defines the Form Bean to which the validations are applied.

<field property=".."> defines a property of a form bean. The attribute *depends* configures the used rule from the Struts rule set. (All rules are defined in the validator-rules.xml).

<arg0 key=".."> defines a parameter which is passed to the error message. In the error message for intRange, there is one parameter expected. (more informations at MessageResouce).

<var-name> sets the name of the variable used in the validation rule and *<var-value>* the value of the variable.

We will create validation rules for each mapping.

```
<form-validation>
<formset>
      <!-- validation mapping für example form -->
      <form name="exampleForm">
          <field
            property="name"
            depends="required, minlength">
                  <arg0 key="exampleForm.name" />
                  <arg1 key="${var:minlength}" resource="false" />
                        <var>
                              <var-name>minlength</var-name>
                              <var-value>3</var-value>
                        </var>
            </field>
            <field
              property="age"
              depends="intRange">
                  <arg0 key="exampleForm.age"/>
                   <arg1 name="intRange" key="${var:min}" resource="false" />
                  <arg2 name="intRange" key="${var:max}" resource="false" />
                        <var>
                              <var-name>min</var-name>
                              <var-value>1</var-value>
                        </var>
                        <var>
                              <var-name>max</var-name>
                              <var-value>150</var-value>
                        </var>
            </field>
        </form>
    </formset>
</form-validation>
```

Activate the ValidatorPlugins in the Struts Config

You must configure the ValidatorPlugin and the XML Dateien in the Struts Configuration to use struts validation.

Open the file struts-config.xml and add the following properties in the tag *<struts-config>* .

The tag *<set-property ...>* defines the XML files which are used by the ValidatorPlugin.

```
<plug-in className="org.apache.struts.validator.ValidatorPlugIn">
    <set-property
        property="pathnames"
        value="/WEB-INF/validator-rules.xml,/WEB-INF/validation.xml"/>
</plug-in>
```

Creation of a Message Resource file

The Message resource file is needed for the output of error messages. It provides all messages.

Create a new file named *ApplicationResources.properties* in the package *de.laliluna.tutorial.validatorform.*

More Information to Message Resource Files can be found in the free MessageResource Tutorial http://www.laliluna.de/struts-message-resources-tutorial-en.html

Add the following lines to the file:

```
errors.suffix=<br>
# -- default error messages for struts validator
errors.required='{0}' is required.
errors.minlength='{0}' can not be less than {1} characters.
errors.range='{0}' is not in the range {1} through {2}.
# -- field names
exampleForm.name=Name
exampleForm.age=Age
```

Open the *struts-config.xml* and add the following:

```
<message-resources
parameter="de.laliluna.tutorial.validatorform.ApplicationResources" />
```

Test the example

You have finished your application and can test it now. Open your browser and call your application with the link below. (We expect a standard installation of a java application server like Jboss or Tomcat.)

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