Overview of Struts Forms

This tutorial gives an overview of the different form classes available in Struts. It explains the features and differences.

General

Autor:
Sebastian Hennebrüder
http://www.laliluna.de/tutorials.html – Tutorials für Struts, EJB, xdoclet und eclipse.
Datum:
September, 13th 2004
Software:
Struts Framework 1.2.6
PDF Download
http://www.laliluna.de/download/struts-forms-overview-en.pdf

What is a Java Bean

A Java Bean is a simple class following some conventions of Sun. The properties of this class have to be private. The access to the properties is realized with so called “Getter” and “Setter” methods.

Example
```
private String name;

public String getName(){
        return name;
}
```

The value is set with a “setter” and read with a “getter”.

What is an ActionForm class

An ActionForm class is a Java Bean saving data, that is used in the struts application.

The ActionForm can be assigned to an action in the configuration file struts-config.xml. There are two possibilites to use an action form.

- Data can be saved to be shown as default values in a web form. (Example: you want to edit data) or to be displayed in a JSP.
- Struts saves data of a submitted form in an action form.

In the second case, the developer have to validate the data if it has the correct type and value area (Example: email address).

The developer can validate data manually in the action class or he puts the validation in the *validate(..)* method of the action form. This method is called after the data has been submitted.

Another feature is to specify validation rules in an xml file.

Types of ActionForm classes

There are different types of action form classes which can be separated in two major groups.

ActionForms (here you have to create a java bean on your own) and dynamic forms (here Struts will create the action form for you).

ActionForm (do it yourself)

You must create a class which extends the class ActionForm.

Below you can see an example for an ActionForm class with the *validate(..)* method.

```
public class myActionForm extends ActionForm {

      private String text = "Hello World!"

      public String getText(){
            return text;
      }

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

      public ActionErrors validate(
            ActionMapping mapping,
            HttpServletRequest request) {

            ActionErrors actionErrors = new ActionErrors();
            if(!text.equals("Hello World!"))
                  actionErrors.add("form", new ActionMessage("Error !!"));

            return actionErrors;
      }

}
```

Definition of the form beans in the struts-config.xml:

```
<form-beans>
    <form-bean name="myForm" type="my.package.myActionForm" />
</form-beans>
```

Accessing the form bean in the action class:

```
public class myAction extends Action {

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



      ActionForm myForm = (ActionForm) form;

      //reading values
      System.out.println("TEXT=" + myForm.getItem());

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

Dynamic ActionForms (Generated by Struts)

You do not create a class when using a dynamic ActionForm (DynaActionForm) but define the class and the properties in the struts-config.xml. Struts will generate the class from the configuration.

Definition of Form-Beans in the struts-config.xml:

```
<form-beans>
 <form-bean name="myForm" type="org.apache.struts.action.DynaActionForm">
```

```
   <form-property name="text" type="java.lang.String" initial="Hello World!"/>
 </form-bean>
</form-beans>
```

Accessing a dynamic form bean in the action class:

```
public class myAction extends Action {

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



      DynaActionForm myForm = (DynaActionForm) form;

      //reading values
      System.out.println("TEXT=" + myForm.get("text"));

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

Validation with XML rules

A feature existing for dynamic and non dynamic action forms is validation. Examples are ValidatorForm and DynaValidatorForm classes. The rules are configured in the struts-config.xml.

Important: When using Validator Forms you must configure the Validator-Plugin in the struts-config.xml.

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

Example of a non dynamic ValidatorForm class:

```
public class myValidatorForm extends ValidatorForm {

      private String text = "Hello World!"

      public String getText(){
            return text;
      }

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

}
```

Definition of a form bean in the struts-config.xml:

```
<form-beans>
    <form-bean name="myForm" type="my.package.myValidatorForm" />
</form-beans>
```

Access of a form bean in an action class:

```
public class myAction extends Action {

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


      //cast the form to type  ValidatorForm
      ValidatorForm myForm = (ValidatorForm) form;

      //reading values
      System.out.println("TEXT=" + myForm.getItem());

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

Definition of a dynamic ValidatorForm in the struts-config.xml:

```
<form-beans>
 <form-bean name="myDynaForm"
type="org.apache.struts.validator.DynaValidatorForm">
  <form-property name="item" type="java.lang.String" />
 </form-bean>
</form-beans>
```

Example for XML Validation (validation.xml):

```
<form-validation>
  <formset>
    <form name="myDynaForm">
      <field property="item"
             depends="required">
        <arg0 key="prompt.item"/>
      </field>
    </form>
  </formset>
</form-validation>
```

There is yet another feature when using Validation. You can assign a validation to a action path. This is useful, when you use the same form bean for multiple actions (Multiple pages with one form bean). The *path* attribute must be specified in the struts-config.xml.

Action forms with this feature are ValidatorActionForm and DynaValidatorActionForm.

Special types of the ValidatorForm class

There are two types of ValidatorForm class, the BeanValidatorForm (only from Struts version 1.2.4) and the LazyValidatorForm (only from Struts version 1.2.6).
Both classes support XML based validation like the ValidatorForm class.

The BeanValidator class can be used when you have a simple class (POJO = Plain Old Java Object).

Example of a simple class:

```
public class myPOJOClass {

      private String text = "Hello World!"

      public String getText(){
            return text;
```

```
      }

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

}
```

Definition of the Form-Beans in the struts-config.xml

```
<form-beans>
      <form-bean name="myForm" type="my.package.myPOJOClass" />
</form-beans>
```

Access to the form bean in the action class

```
public class myAction extends Action {

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


      //cast the form to the type BeanValidatorForm
      BeanValidatorForm myForm = (BeanValidatorForm) form;

      //reading values
      System.out.println("TEXT=" + myForm.get("text"));

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

The LazyValidatorForm is a very dynamic type of the ValidatorForm. You do not need to specify any properties at all. They are created from the submitted parameters of a form.

Definition of the form bean in the struts-config.xml

```
<form-beans>
    <form-bean name="testForm"
type="org.apache.struts.validator.LazyValidatorForm" />
</form-beans>
```

Extract from a JSP:

```
<html:form action="/test">
    <html:text property="text" />
    <br>
    <html:submit />
</html:form>
```

Accessing the form bean in an action class:

```
public class myPOJOAction extends Action {

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

```
      //casten der form als LazyValidatorForm
      LazyValidatorFormtestForm = (LazyValidatorForm) form;

      //wert auslesen
      System.out.println("NAME=" + testForm.get("text"));

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

Overview of ActionForm classes

Form Typ	Features	Validation	Dynamic
ActionForm	Basic ActionForm class, all action form classes extend this class.	No	No
DynaActionForm	Dynamic type of ActionForm class. No Java bean must be created. Definition in the struts-config.xml	No	Yes
ValidatorForm	XML based Validation	Yes	No
DynaValidatorForm	Dynamic type of ActionForm class. No Java bean must be created. Definition in the struts-config.xml. XML based Validation.	Yes	Yes
ValidatorActionForm	XML based Validation. Assignment of the validation rules to the *path* attribute of the action. You can use different validations for the same form bean.	Yes	No
DynaValidatorActionForm	Dynamic type of ActionForm class. XML based Validation. Assignment of the validation rules to the *path* attribute of the action. You can use different validations for the same form bean.	Yes	Yes
BeanValidatorForm	XML based Validation. Can be used when you want to have a simple object class as a from bean.	Yes	No
LazyValidatorForm	Dynamic type of ActionForm class. No Java bean must be created. Definition in the struts-config.xml. XML based Validation. You do not need to specify any attributes.	Yes	Yes