**Struts Declarative Support**

Declarative Validation Support

Declarative validation support is provided by the struts classes given below.

1. ValidatorForm
2. DynaValidataorForm
3. ValidatorActionForm
4. DynaValidatorActionForm

These four struts classes provides following features:

1. Declarative Validation support(both Client side and Application side)
2. Form Backup Support
3. Localization support(II8N)

ValidatorForm and DynaValidataorForm provides us single form handling support ie It can handle a single form validation, form backup or localization support. For multiform handling we need to use either ValidatorActionForm or DynaValidatorActionForm.

In programmatic approach the validations run through programs written for the same while in declarative approach the validations executes through xml files where validations are written. In programmatic approach the ActionServlet calls the requestprocessor for creating a bean object and checks validation through validate() method of ActionForm class while in decalarative approach we need to write validations in xml files.

For each input form we create a validation.xml file and predefined rules are given under validationrules.xml. To read this validations they given a plug in class called validator plug-in. validator plug in will read the xml files and this validator plug in class will run by ActionServlet init() method. Plug in class is configured under the plug in tag of struts-config.xml. ActionServelet init() method will create a validator plug in class object which will invoke the init() method validator plug in class which will read our validation.xml files.

ActionServlet

Struts-config.xml

<plug-in>

Validationrule.xml

Validation.xml

Init()

Init()

Validator plug in Class

Under Validationrule.xml we already have many types of validations like email, Strings, Alphanumeric, numeric etc. so that we don’t require to write any types of validations. We just have to reference the various validations according to our needs from validation-rule.xml. Total 23 validation rules are written in validation-rule.xml. It also has 13 error message and a plug in configuration. Validator class also supports for localization support i.e. if a number is being entered then its format may be different for different country. So struts provides us with validator class which validates the mob number or phone number according to the user’s country or language.

For each and every form’s validations we can write in validation.xml file

An Example of Declarative Validation Approach

1. Index file

<%@taglib uri=*"http://struts.apache.org/tags-html"* prefix=*"html"*%>

<h1>Reg Form</h1>

<html:form action=*"reg"*>

<pre>

ID: <html:text property=*"id"*/><html:errors property=*"id"*/>

Name: <html:text property=*"name"*/><html:errors property=*"name"*/>

Email: <html:text property=*"email"*/><html:errors property=*"email"*/>

Mob: <html:text property=*"mob"*/><html:errors property=*"mob"*/>

Address: <html:text property=*"address"*/><html:errors property=*"address"*/>

Salary: <html:text property=*"salary"*/><html:errors property=*"salary"*/>

Credit Card:<html:text property=*"creditcard"*/><html:errors property=*"creditcard"*/>

<html:submit value=*"Register"*/>

</pre>

</html:form>

1. Validation.xml

<!DOCTYPE form-validation PUBLIC

"-//Apache Software Foundation//DTD Commons Validator Rules Configuration 1.3.0//EN"

"http://jakarta.apache.org/commons/dtds/validator\_1\_3\_0.dtd">

<form-validation>

<!-- Common validations for all locales -->

<formset>

<form name=*"RF"*>

<field property=*"id"* depends=*"intRange"*>

<arg key=*"id"* resource=*"false"*/>

<arg key=*"${var:min}"* resource=*"false"*/>

<arg key=*"${var:max}"* resource=*"false"*/>

<var>

<var-name>min</var-name>

<var-value>111</var-value>

</var>

<var>

<var-name>max</var-name>

<var-value>999</var-value>

</var>

</field>

<field property=*"name"* depends=*"required"*>

<arg key=*"name"* resource=*"false"*/>

</field>

<field property=*"email"* depends=*"required,email"*>

<arg key=*"email"* resource=*"false"*/>

</field>

<field property=*"creditcard"* depends=*"creditCard"*>

<arg key=*"creditcard"* resource=*"false"*/>

</field>

<field property=*"address"* depends=*"required"*>

<arg key=*"address"* resource=*"false"*/>

</field>

</form>

</formset>

<!-- Indian Mobile validation and salary validations -->

<formset language=*"en"* country=*"IN"*>

<form name=*"RF"*>

<field property=*"salary"* depends=*"floatRange"*>

<arg key=*"id"* resource=*"false"*/>

<arg key=*"${var:min}"* resource=*"false"*/>

<arg key=*"${var:max}"* resource=*"false"*/>

<var>

<var-name>min</var-name>

<var-value>10000.5</var-value>

</var>

<var>

<var-name>max</var-name>

<var-value>200000.5</var-value>

</var>

</field>

<field property=*"mob"* depends=*"mask"*>

<arg key=*"mob"* resource=*"false"*/>

<var>

<var-name>mask</var-name>

<var-value>^[7-9]\d{9}$</var-value>

</var>

</field>

</form>

</formset>

<formset language=*"en"* country=*"US"*>

<form name=*"RF"*>

<field property=*"salary"* depends=*"floatRange"*>

<arg key=*"id"* resource=*"false"*/>

<arg key=*"${var:min}"* resource=*"false"*/>

<arg key=*"{var:max}"* resource=*"false"*/>

<var>

<var-name>min</var-name>

<var-value>100000.5</var-value>

</var>

<var>

<var-name>max</var-name>

<var-value>2000000.5</var-value>

</var>

</field>

<field property=*"mob"* depends=*"mask"*>

<arg key=*"mob"* resource=*"false"*/>

<var>

<var-name>mask</var-name>

<var-value>^[2-4]\d{9}$</var-value>

</var>

</field>

</form>

</formset>

</form-validation>

1. Struts-config.xml

<!DOCTYPE struts-config PUBLIC

"-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"

"http://struts.apache.org/dtds/struts-config\_1\_3.dtd">

<struts-config>

<form-beans>

<form-bean name=*"RF"* type=*"beans.RegForm"*></form-bean>

</form-beans>

<action-mappings>

<action path=*"/reg"* name=*"RF"* validate=*"true"* scope=*"session"* input=*"/index.jsp"*/>

</action-mappings>

<message-resources parameter=*"Messages.properties"*></message-resources>

<plug-in className=*"org.apache.struts.validator.ValidatorPlugIn"*>

<set-property property=*"pathnames"*

value=*"/WEB-INF/validator-rules.xml,*

*/WEB-INF/validation.xml"*/>

</plug-in>

</struts-config>

1. Web.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<web-app xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns=*"http://xmlns.jcp.org/xml/ns/javaee"* xsi:schemaLocation=*"http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd"* id=*"WebApp\_ID"* version=*"3.1"*>

<display-name>RegForm\_VF</display-name>

<welcome-file-list>

<welcome-file>index.jsp</welcome-file>

</welcome-file-list>

<servlet>

<servlet-name>action</servlet-name>

<servlet-class>org.apache.struts.action.ActionServlet</servlet-class>

<init-param>

<param-name>config</param-name>

<param-value>/WEb-INF/struts-config.xml</param-value>

</init-param>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>action</servlet-name>

<url-pattern>\*.do</url-pattern>

</servlet-mapping>

</web-app>

1. RegForm.java bean class

**package** beans;

**import** org.apache.struts.validator.ValidatorForm;

**public** **class** RegForm **extends** ValidatorForm {

**private** **int** id;

**private** **long** mob, creditcard;

**private** **float** salary;

**private** String name,email, address;

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** **long** getMob() {

**return** mob;

}

**public** **void** setMob(**long** mob) {

**this**.mob = mob;

}

**public** **long** getCreditcard() {

**return** creditcard;

}

**public** **void** setCreditcard(**long** creditcard) {

**this**.creditcard = creditcard;

}

**public** **float** getSalary() {

**return** salary;

}

**public** **void** setSalary(**float** salary) {

**this**.salary = salary;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getEmail() {

**return** email;

}

**public** **void** setEmail(String email) {

**this**.email = email;

}

**public** String getAddress() {

**return** address;

}

**public** **void** setAddress(String address) {

**this**.address = address;

}

}

1. Messages.properties

errors.range=**{0}** is not in the range **{1}** through **{2}**.

errors.required=**{0}** is required.

errors.email=**{0}** is an invalid e-mail address.

errors.creditcard=**{0}** is an invalid credit card number.