**Sample Examples on struts of Hello**

Web Application Directory Stretcher is as follows

1. View Resources (jsp, html files)
2. WEB-INF
3. Classes
4. Lib
5. Web.xml
6. Struts-config.xml

Steps to create a Web Application

1. Create a new Dynamic Web Project using Eclipse IDE and Specify the Application Server (in this case Apache tomcat server) and provide the path for the same.
2. Create a index.jsp file and design form as required. Write code below in index.jsp. Make sure the all struts library jars are imported into the eclipse IDE so as to use the custom tags provided by the struts to make our webpage rapidly. Taglib tag has to specify for the location of custom tags provided by the struts and prefix is compulsory. We can write any prefix. Form html:form tag we to specify our action.

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

<h1>HELLO FORM</h1>

<html:form action=*"hello"* focus=*"name"*>

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

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

</html:form>

1. Now Configure the web.xml file as follows:

<?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>Hello</display-name>

<welcome-file-list>

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

</welcome-file-list>

<servlet>

<servlet-name>actionservlet</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>actionservlet</servlet-name>

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

</servlet-mapping>

</web-app>

In welcome-file-list tag choose index.jsp file and remove other tags from it. Write a servlet tag to configure the struts Front Controller Class called ActionServlet. Similarly configure the same for servlet mapping. It configures the url-pattern for the page.

1. Make a package beans in folder java resources and we write various classes we want to used. Create a Class HelloFormBackup Class and extend it using Class FormAction already given in Struts lib. Then Declare appropriate variable for various inputs that user wants to provide on the form and generate the getters and setters for the same. Override the validate() method of ActionForm if we want to perform simple validations. Return type validate method is ActionEr rors() which stores our error message using object of type ActionMessage() which takes parameter as label following which error message is written in a file . Following is class looks like:

package beans;

import javax.servlet.http.\*;

import org.apache.struts.action.\*;

public class HelloFormBackup extends ActionForm{

private String name;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public ActionErrors validate(ActionMapping mapping, HttpServletRequest request){

ActionErrors ae=new ActionErrors();

if (name.equals(""))

ae.add("name", new ActionMessage("msg"));

return ae;

}

}

1. Create an Error Message file called message.properties which stores index in the form of text followed by errors and in ActionError() we have specify the index we want to display for a particular type of user inputs. Message.properties is as follows

msg=<font color='red'> name is required </font>

1. Write a Controller Class named HelloController Class for action and extend it with struts library class called Action. Override the Action Class Method execute() which has ActionForward as return type. Import the various libraries required. HelloController Class is as Follows:

package beans;

import javax.servlet.http.\*;

import org.apache.struts.action.\*;

public class HelloController extends Action {

public ActionForward execute(ActionMapping mapping, ActionForm from,

HttpServletRequest request, HttpServletResponse response)

throws Exception {

String name = request.getParameter("name");

request.setAttribute("res", "Hello...."+name);

return mapping.findForward("success");

}

}

1. Write h success.jsp page displaying the result.

<%=request.getAttribute(“res”)%>

1. Configure the struts-config.xml file as follows. It has following tags
2. Form-beans tag which specifies the unique label for subclass of ActionForm and type is for subclass of ActionForm.
3. Global exceptions allow you to catch uncaught runtime exceptions that occur in your Action subclasses, displaying them with a custom error message. It has two parameters as Follows:

• key: An error key. When an exception handler is fired, an ActionMessage with this key is created and put on the request. This error message gets pasted on the JSP containing an tag that is finally displayed. Note that key is a required attribute, which you have to specify even if you don’t use it.

• type: Describes the type of error that is caught. Struts will first try to find the error class that matches the declared types. If none matches exactly, then Struts goes up that error’s superclass tree, until it finds a match with a declared exception.

1. global forwards used to define forwarding paths accessible to all Actions or ExceptionHandlers. In fact, all such paths will be accessible as long as we have an ActionMapping instance initialized by Struts. This is certainly the case for the execute() functions in Action or ExceptionHandler.
2. The tag acts as a container for each form handler, described by an tag. The tag contains a few attributes that configure the handler:

• path: Describes the name of the form handler.

• type: The fully qualified class name of the Action subclass that handles business logic.

• name: The name of the form bean associated with this handler.

• validate: Tells Struts whether it should perform simple validation on the form data.

• scope: Sets the scope of the form data. Only request or session scopes are allowed.

• input: The relative path of the page that forms the input of this page.

1. The message resources section declares the location of your properties file that stores the key/value pairs of your application.
2. Plug-ins are custom extensions to Struts. An example is the Tiles framework, which was developed independently of Struts.

<?xml version="1.0" encoding="ISO-8859-1" ?>

<!DOCTYPE struts-config PUBLIC

"-//Apache Software Foundation//DTD Struts Configuration 1.1//EN"

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

<struts-config>

<form-beans>

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

</form-beans>

<global-forwards>

</global-forwards>

<action-mappings>

<action path=*"/ hello"* name=*"RF"* input=*"/index.jsp"* type=*"beans.HelloController"*>

<forward name=*"success"* path=*"/success.jsp"*/>

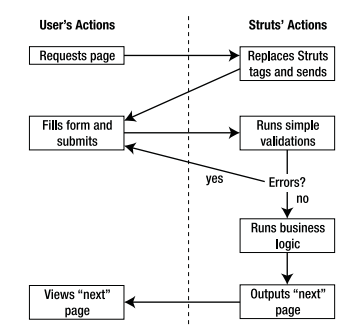
</action>

</action-mappings>

<message-resources parameter=*"beans/Message"*/>

</struts-config>

**Life Cycle of Typical Struts Application**



**Arhitecture of Struts**

**ActionServlet**

Struts-config.xml

**Forms**

Data valid Model

Holds data into beans and does validations

Returns Validation Error

DTO

Data Access Object(DAO)

Business Logic

Controller Class for Every Form

Beans Class for Every Form

Database

Whenever user submits a Form the same request goes to ActionServlet Class also called Front Controller. After getting user inputs the ActionServlet will read the data and hold the same in an object of Bean Class for every form user submits. If a form has 10 parameters then bean will store all the ten parameters. So we don’t need to do read operation like request.getParameter(“attribute value name”). Then Bean will carry out the validations on tha data. If some validation errors are there, the same is returned to ActionServlet. ActionServlet returns the same error to the specific form. So Bean Object provides two things:

1. Form Validations.
2. Form Backup

If there are not any validation errors, the data is sent to the controllers attached to it and cotrollers read the data and sends it to model part of MVC architecture using Data Transfer Object(DTO). The bean data is stored in DTO before sending to the Model Component. The business part of Model component will execute business logic using the data and if it needs to store the data at the backend ie Database. Then it uses Data Access Object to store the same to the Database. If there are a number of forms and number of controllers then how form can interact with the beans and controllers associated with it. So this problem is overcome by the struts-config.xml. Here we need to configure the flow of our Web based applications. ActionServlet Class will understand by reading the struts-config.xml file that which bean class and controller class needed to execute for a particular form.

Controller after performing the task according to the logic will return the success or failure page. Same stuff needs to be configured in the struts.xml file. ActionServlet will forward the success page to the user.

Questions

1. Difference between global forward tags and forward tag in action-mapping