Session-

Struts ActionForm & DynaActionForm

Contents

- ActionForm
  - Understanding the Life Cycle of an ActionForm
  - What an ActionForm Is
  - What an ActionForm Is Not
  - Reducing the Number of ActionForms
- DynaActionForm

ActionForm

- An **ActionForm** is an object representation of an HTML form.
- An ActionForm is not part of the Model.
- An ActionForm sits **between** the **View** and **Controller** acting as a transfer object between the two layers.
- An ActionForm represents not the just the data, but the data entry form itself.

ActionForm

- ActionForms have JavaBean properties to **hold fields from the form**.
- can consist of **master detail relationships** and/or can have dynamic properties.
- ActionForms are configured to be stored by Struts in either session or request scopes. **Session** scope is the **default scope**.
- Struts automatically populate the ActionForm's JavaBean properties from corresponding request parameters, performing type conversion into primitive types (or primitive wrapper types) if needed.

Life Cycle of an ActionForm

Request received by controller

Create or recycle ActionForm

Call reset()

Store ActionForm in desired scope

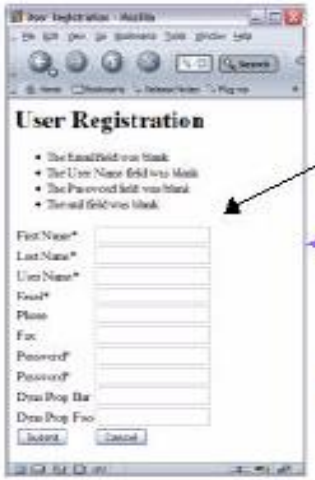
Populate ActionForm from request

Call validate() on ActionForm
if validate attribute = 'true'

Validation Failed

Error messages

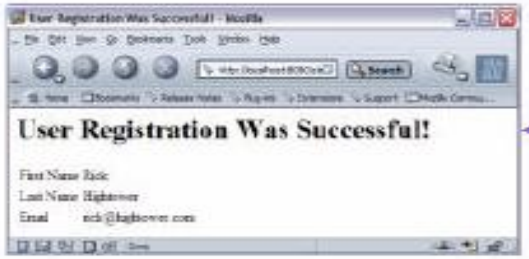
Forward to input



Validation Succeeded

Call execute() on Action

Forward to "success"



reset() Method

- set properties to default values.

- don’t initialize properties for an update operation in the reset() method.

validate() Method

- to check for
  - **field validation and**

    i.e., a field is in a certain range, a certain length, etc..

  - **relationships between fields**

    i.e., Checking to see if the **start date** is before the **end date**, checking to see if the **password** and the **retyped password** field are **equal**, etc..

What an ActionForm Is

- **Data Supplier**: Supplies Data to html:form
- **Data Collector**: Processes Data from html:form
- **Action Firewall**: Validates Data before the Action Sees It

What an ActionForm is Not

- Not Part of the Model or Data Transfer Object
- Not an Action, Nor Should It Interact with the Model

Issues with using ActionForm

- ActionForm class has to be created in Java programming language
- For **each HTML** form page, a **new ActionForm class** has to be created
- Every time HTML form page is modified (a property is added or remove), ActionForm class has to be modified and recompiled

Reducing the Number of ActionForms

- Essentially, you have to create an ActionForm for each HTML/JSP form.
- There are many strategies to get around this.
  - **Super ActionForms**
    - use a super class ActionForm that has many of the fields that each of the other HTML forms need.
  - **Mapped Back ActionForms**
    - ActionForms can be mapped back.
  - **DynaActionForms**
    - instead of creating an ActionForm for each HTML form, you instead **configure** an ActionForm for each HTML form.

What is DynaActionForm?

- **org.apache.struts.action.DynaActionForm**
  - extends **ActionForm** class

- In DynaActionForm scheme,
  - Properties are **configured in configuration file** rather than coding
  - reset() method resets all the properties back to their initial values
  - You can still subclass DynaActionForm to override **reset()** and/or **validate()** methods
    - Version exists that works with Validator framework to provide automatic validation

How to Configure DynaActionForm?

- Configure the properties and their types in your **struts-config.xml** file.
  - add one or more <form-property> elements for each <form-bean> element

```
<form-beans>
      <form-bean  name="userForm"
                  type="org.apache.struts.action.DynaActionForm" >
          <form-property name="name" type="java.lang.String" />
          <form-property name="age" type="java.lang.Integer" />
          <form-property name="email" type="java.lang.String" />
      </form-bean>
</form-beans>
```

Difference between **ActionForm** and **DynaActionForm**

1) For a **DynaActionForm**, the **type** attribute of the form-bean is always **org.apache.struts.action.DynaActionForm**.

2) A regular ActionForm is developed in Java and declared in the *struts-config.xml* as follows:

```
<form-beans>
      <form-bean  name="userForm"
                  type="examples.simple.SimpleActionForm" >
      </form-bean>
</form-beans>
```

Types Supported by DynaActionForm

- java.lang.BigDecimal, java.lang.BigInteger
- boolean and java.lang.Boolean
- byte and java.lang.Byte
- char and java.lang.Character
- java.lang.Class, double and java.lang.Double
- float and java.lang.Float
- int and java.lang.Integer
- long and java.lang.Long
- short and java.lang.Short
- java.lang.String
- java.sql.Date, java.sql.Time, java.sql.Timestamp

Issues with writing **validate()** method in **ActionForm**

- DynaActionForm does not provide **default behavior for validate()** method.
- You have to write validate() method for each ActionForm class, which results in redundant code.
  - It is not recommended.
- Changing validation logic requires recompiling.

How to perform **Validation** with DynaActionForm?

- Use **Validator** Framework
  - Use **DynaValidatorForm** class (instead of DynaActionForm class)
  - DynaValidatorForm class **extends** **DynaActionForm** and provides basic field validation based on configuration file.

```
<form-beans>
    <form-bean name="logonForm"
            type="org.apache.struts.validator.DynaValidatorForm">
            <form-property name="username“ type="java.lang.String"/>
            <form-property name="password“ type="java.lang.String"/>
    </form-bean>
<form-beans>
```

Creating Action Class

```
public class CreateAction extends Action
{
    public ActionForward execute(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    {
     DynaActionForm customerForm = (DynaActionForm) form;
     String name = (String) customerForm.get("username");
     String pass = (String) customerForm.get(“password");
     …….
     }
}
```

Thank you!