

## What is Struts 2?

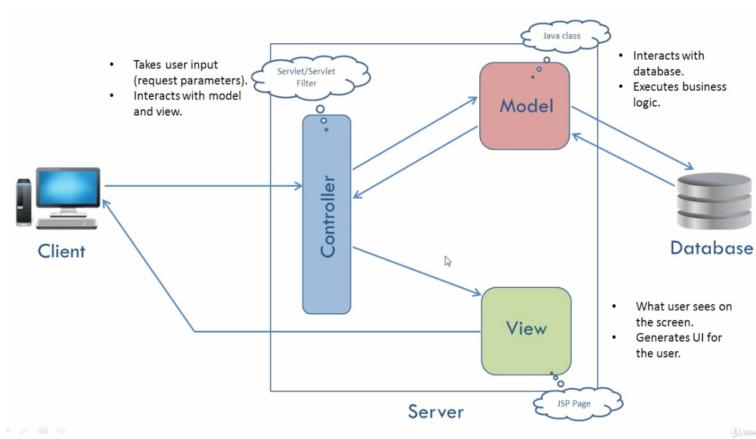
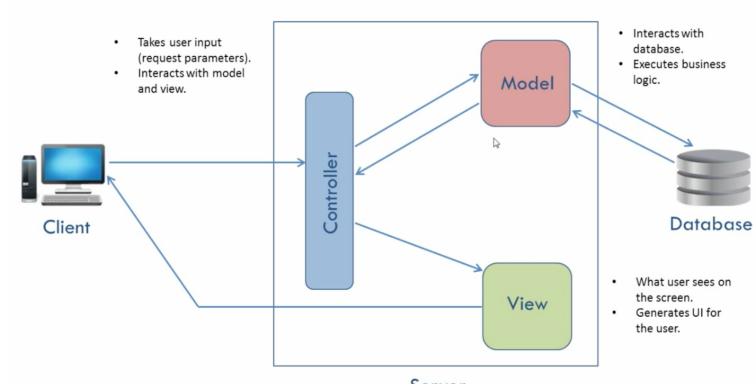
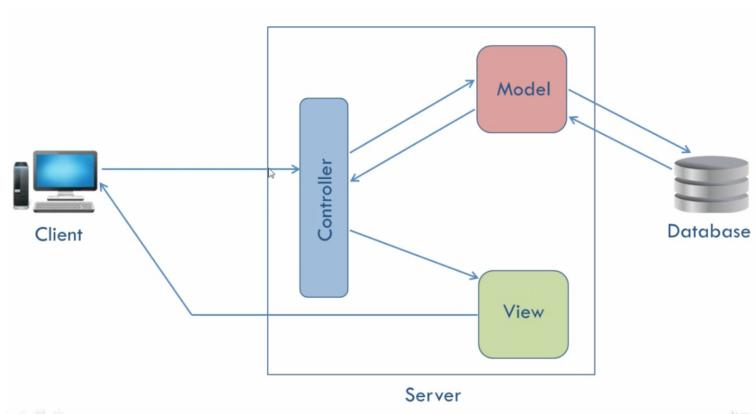
---

- ❑ Apache Struts 2 is a web application framework used for developing Java EE web applications.
- ❑ It implements MVC pattern.
- ❑ It is one of the popular Java web framework.

## Overview

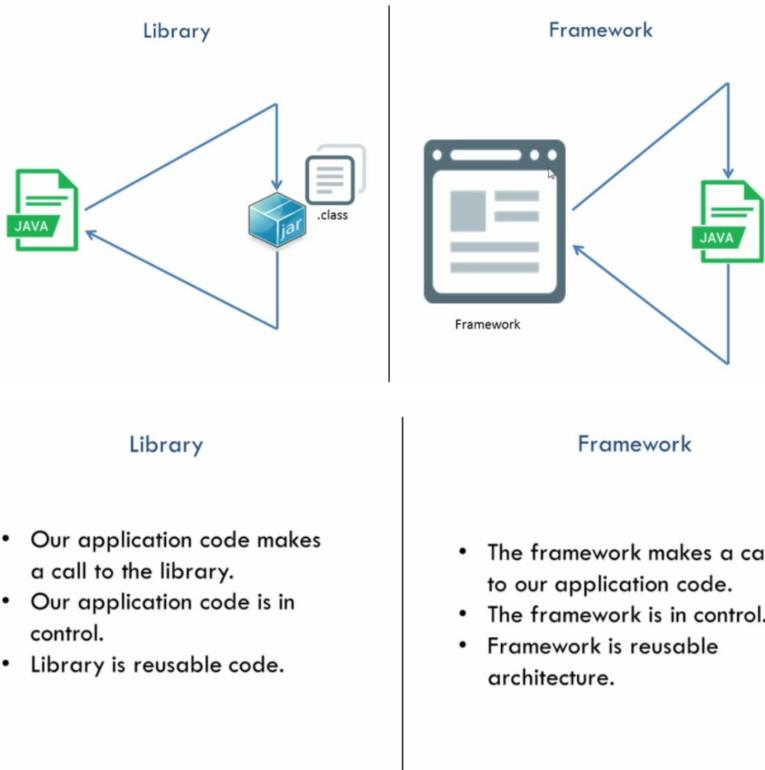
---

- ❑ What is a Framework?
  - ❑ What is MVC?
  - ❑ Struts 2 Architecture
  - ❑ Registration Application
  - ❑ ValueStack and OGNL
  - ❑ Interceptors
  - ❑ Struts 2 Tags
  - ❑ Internationalization
  - ❑ Integration with Database
- 
- **MVC stands for Model View Controller.**
  - **It is an architectural pattern.**
  - **MVC divides a software application in 3 parts:- model, view and controller.**



What is the framework and its benefits?

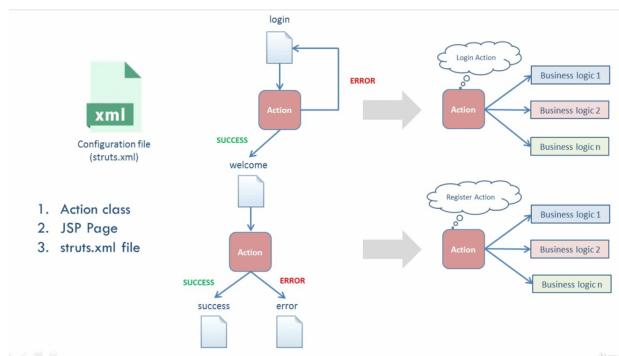
- A framework is a collection of some foundation classes and libraries which makes application development easier.



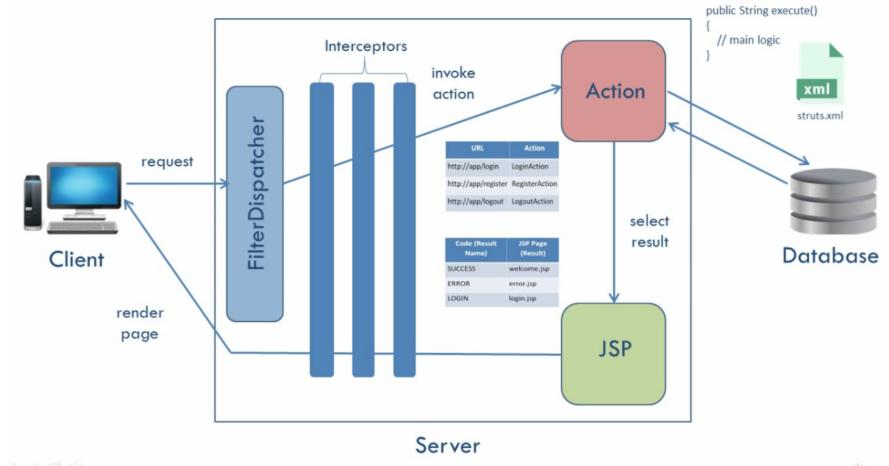
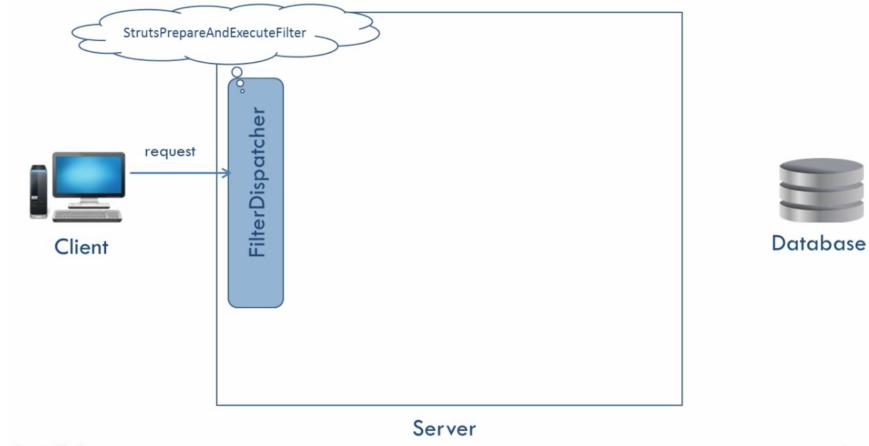
- Our application code makes a call to the library.
  - Our application code is in control.
  - Library is reusable code.
- The framework makes a call to our application code.
  - The framework is in control.
  - Framework is reusable architecture.

## Benefits of a Framework

- Framework makes us write our application by following a pattern.
- We don't have to worry about the design of our application.
- Makes our application maintainable in future.



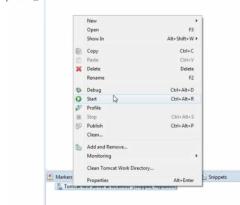
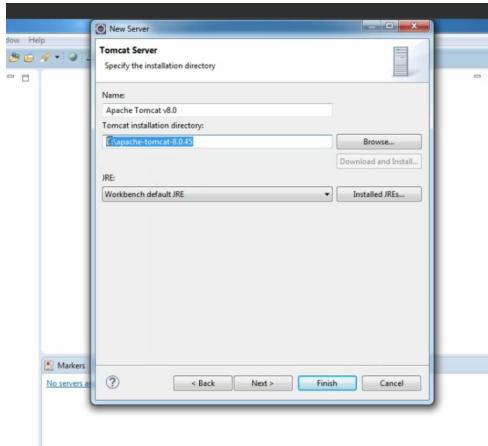
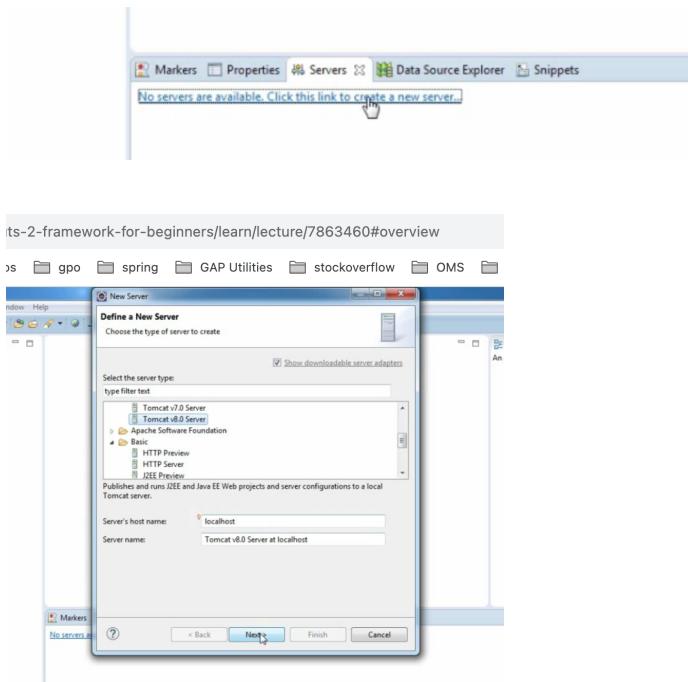
Each action will have an execute method.

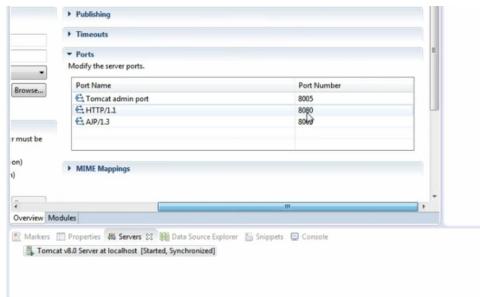


Every action is an execution method.

## Setup Tomcat Server

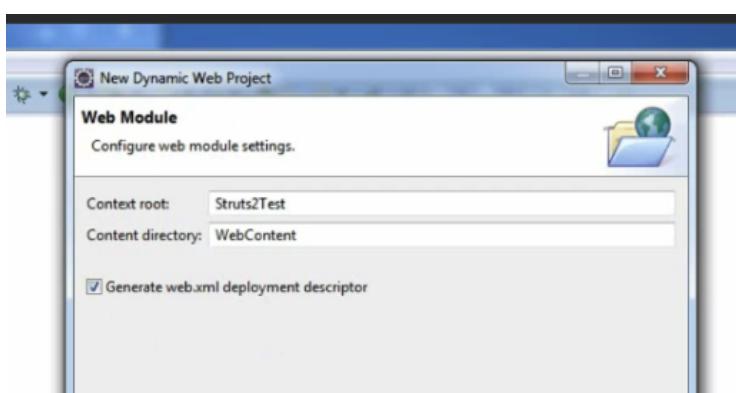
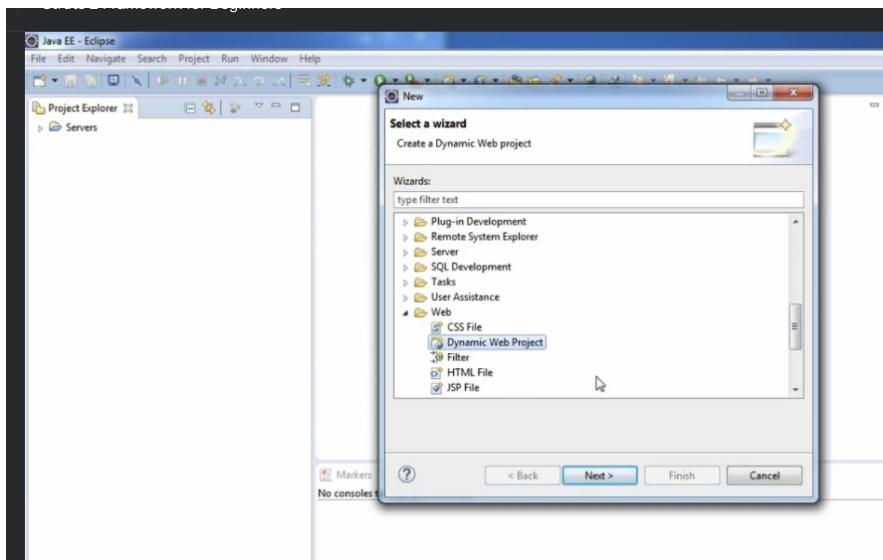


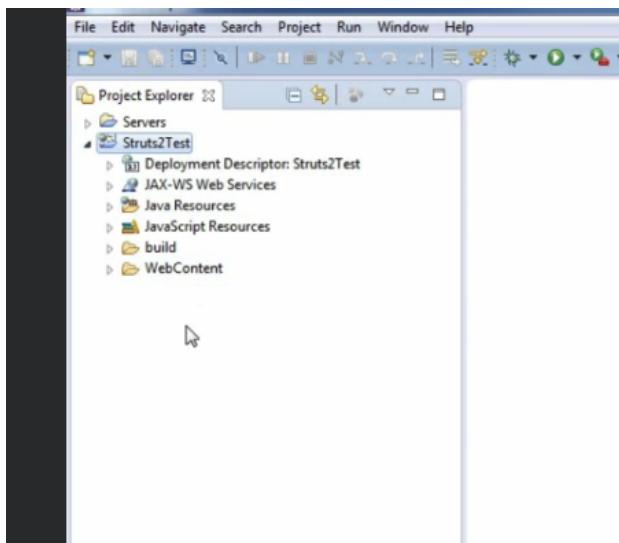




Double click if you want to change the port number

## Setup Struts 2 work environment



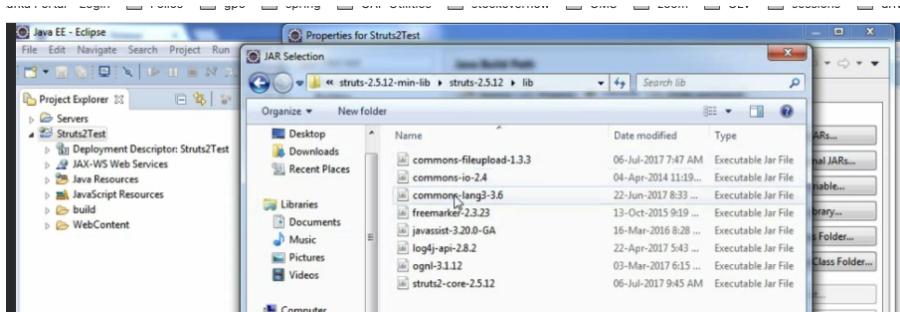


## Download struts jar

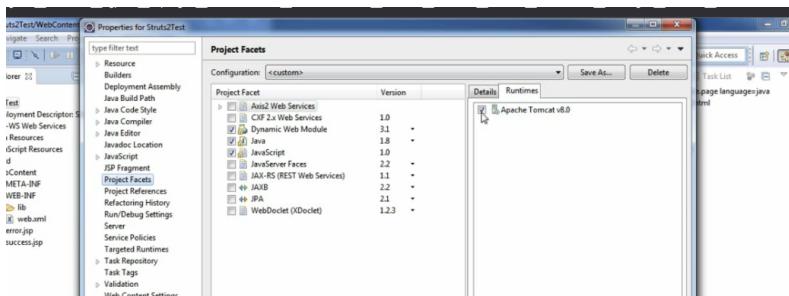
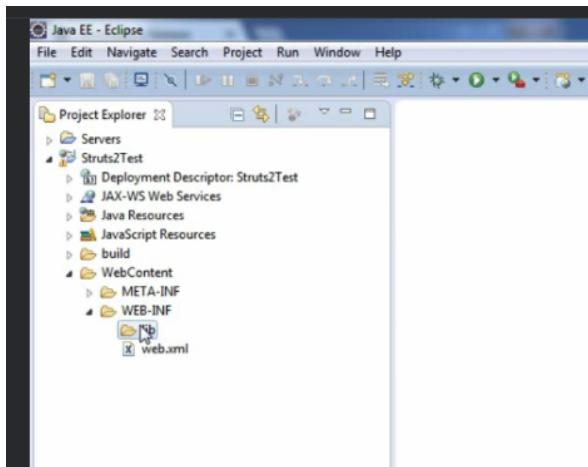
A screenshot of the Apache Struts official website at https://struts.apache.org. The page has a blue header with the Apache logo and navigation links for HOME, SUPPORT, DOCUMENTATION, and CONTRIBUTING. The main title is 'Apache Struts'. Below the title, a brief description states: 'Apache Struts is a free, open-source, MVC framework for creating elegant, modern Java web applications. It favors convention over configuration, is extensible using a plugin architecture, and ships with plugins to support REST, AJAX and JSON.' At the bottom of the page are two buttons: a dark blue 'Download' button and a white 'Technology Primer' button.

A screenshot of the 'Download a Release' page for Struts 2.5.12 at https://struts.apache.org/download.cgi#struts2512. The page title is 'STRUTS'. It displays a list of download options for Struts 2.5.12, including 'Version Notes', 'Full Distribution' (with a link to struts-2.5.12-all.zip), 'Example Applications' (with a link to struts-2.5.12-apps.zip), 'Essential Dependencies Only' (with a link to struts-2.5.12-min-lib.zip), 'All Dependencies' (with a link to struts-2.5.12-lib.zip), and 'Documentation' (with a link to struts-2.5.12-docs.zip). A cursor arrow points to the 'struts-2.5.12-min-lib.zip' link.

Download and add to the proj



Add same thing to below lib folder as well.



Ensure that check box enabled from project folder

Create success and error jsp. And create an action class. Each action class contains an execute method.

Each execution method contains business logic as shown below

```

1  public class TestAction {
2      public String execute()
3      {
4          System.out.println("execute() method called");
5          return "success";
6      }
7  }

```

Now we need struts xml in the src folder as shown below. With line 2, 3, 4 this xml will be treated as struts.xml.

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE struts PUBLIC
3   "-//Apache Software Foundation//DTD Struts Configuration 2.5//EN"
4   "http://struts.apache.org/dtds/struts-2.5.dtd">
5
6<struts>
7  <package name="test" extends="struts-default">
8  </package>
9 </struts>

```

Package groups actions and result types into logical unit as shown below.

```

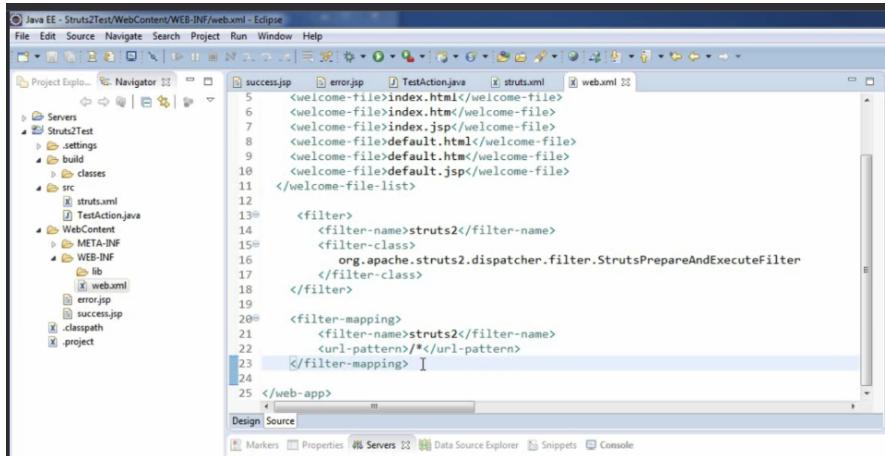
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE struts PUBLIC
3   "-//Apache Software Foundation//DTD Struts Configuration 2.5//EN"
4   "http://struts.apache.org/dtds/struts-2.5.dtd">
5
6<struts>
7  <package name="test" extends="struts-default">
8    <action name="testAction" class="TestAction">
9      <result name="success">/success.jsp</result>
10     <result name="error">/error.jsp</result>
11   </action>
12 </package>
13 </struts>

```

<action name="testAction" is path url, class=TestAction which java class execute method should execute. Here TestAction.java class execute method will be executed.

Each action will have <result name = what execute method returns that should match here to /success.jsp

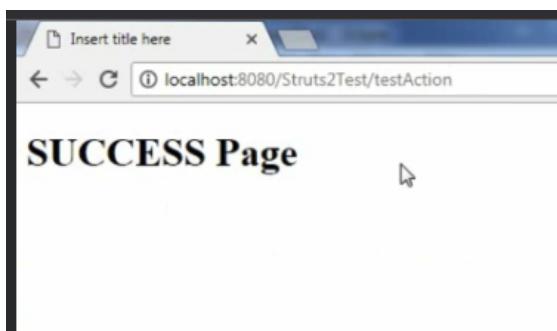
So far we did model and view. Now the controller filter. So everything will come first to filter and redirect to based on url pattern



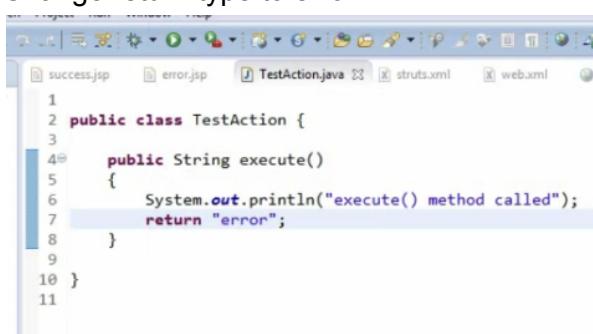
The screenshot shows the Eclipse IDE interface with the 'web.xml' file open in the editor. The code defines welcome files and a filter mapping for the 'struts2' filter.

```
<welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
</welcome-file-list>
<filter>
    <filter-name>struts2</filter-name>
    <filter-class>
        org.apache.struts2.dispatcher.filter.StrutsPrepareAndExecuteFilter
    </filter-class>
</filter>
<filter-mapping>
    <filter-name>struts2</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>

```

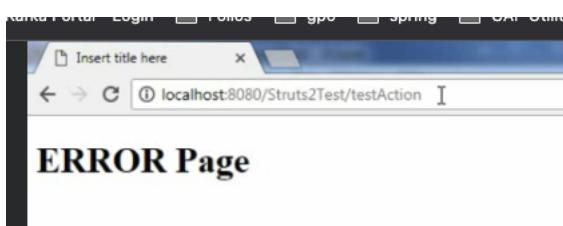


Change return type to error



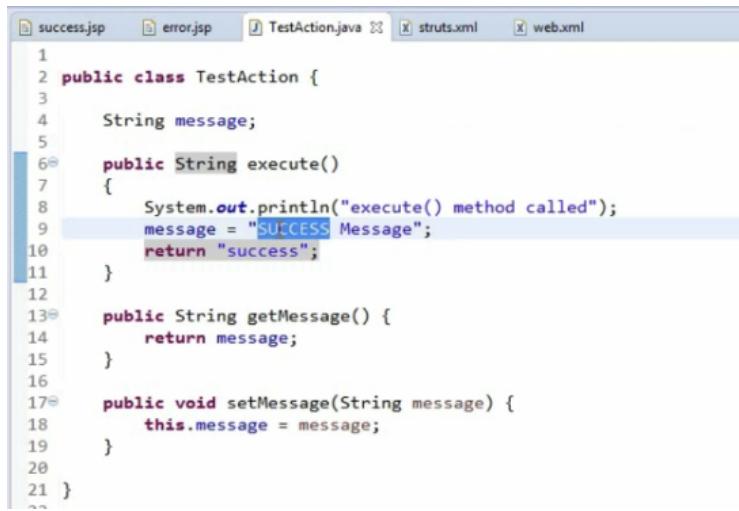
The screenshot shows the Eclipse IDE interface with the 'TestAction.java' file open in the editor. The 'execute()' method now returns the string "error".

```
public class TestAction {
    public String execute()
    {
        System.out.println("execute() method called");
        return "error";
    }
}
```



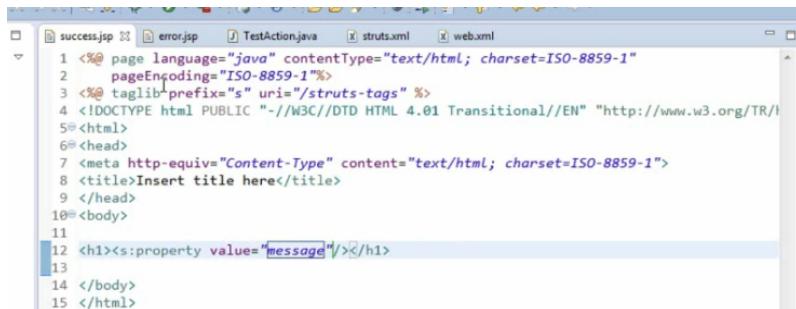
## 10. Passing data from Action to JSP

In our action class introduce a variable message and getter and setter. And logic in execute method as shown below.



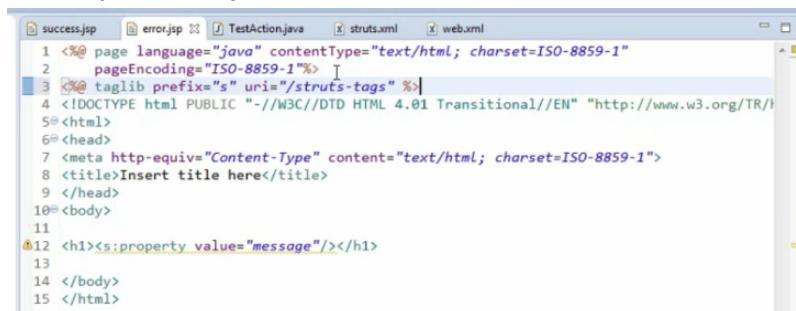
```
1 public class TestAction {
2     String message;
3
4     public String execute()
5     {
6         System.out.println("execute() method called");
7         message = "SUCCESS Message";
8         return "success";
9     }
10
11     public String getMessage()
12     {
13         return message;
14     }
15
16     public void setMessage(String message)
17     {
18         this.message = message;
19     }
20 }
21 }
```

Add struts-tags to the jsp page. And introduce a property tag at line 12. We are given a value as "message" which is used in our action class that will return here.

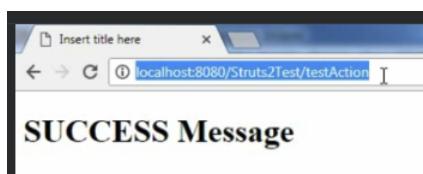


```
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2     pageEncoding="ISO-8859-1"%>
3 <%@ taglib prefix="s" uri="/struts-tags" %>
4 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/I
5<html>
6<head>
7 <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8 <title>Insert title here</title>
9 </head>
10<body>
11
12 <h1><s:property value="message"/></h1>
13
14 </body>
15 </html>
```

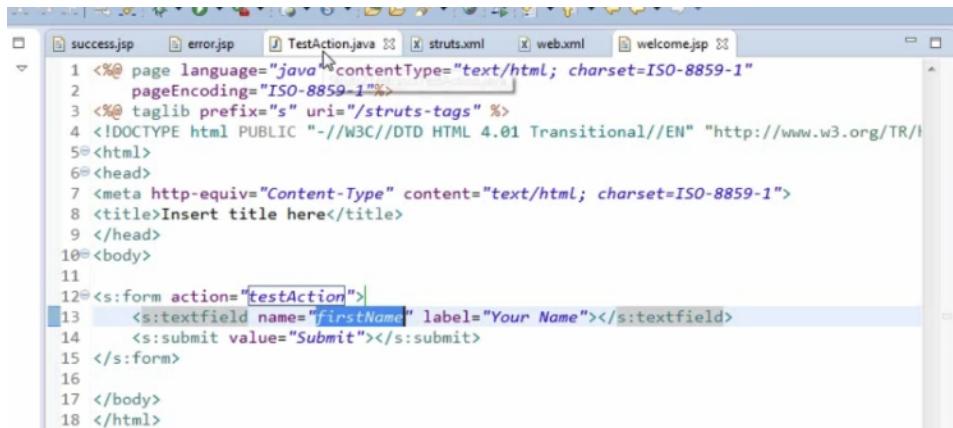
Similarly for error.jsp



```
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2     pageEncoding="ISO-8859-1"%>
3 <%@ taglib prefix="s" uri="/struts-tags" %>
4 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/I
5<html>
6<head>
7 <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8 <title>Insert title here</title>
9 </head>
10<body>
11
12 <h1><s:property value="message"/></h1>
13
14 </body>
15 </html>
```



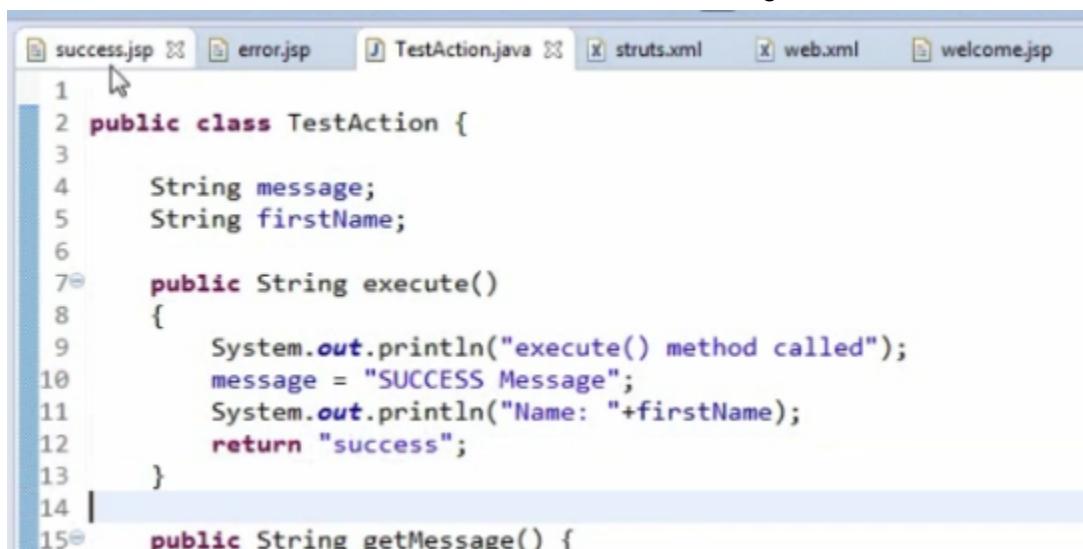
Create a welcome page which contains a text field and submit button.



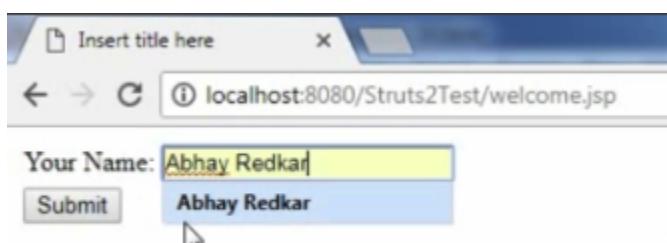
```
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2     pageEncoding="ISO-8859-1"%>
3 <%@ taglib prefix="s" uri="/struts-tags" %>
4 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/I
5<html>
6<head>
7 <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8 <title>Insert title here</title>
9 </head>
10<body>
11
12<s:form action="testAction">
13     <s:textfield name="firstName" label="Your Name"></s:textfield>
14     <s:submit value="Submit"></s:submit>
15 </s:form>
16
17 </body>
18 </html>
```

On submit it will call testAction.java class.

Create same name variable in action class with setter and getters



```
1
2 public class TestAction {
3
4     String message;
5     String firstName;
6
7     public String execute()
8     {
9         System.out.println("execute() method called");
10        message = "SUCCESS Message";
11        System.out.println("Name: "+firstName);
12        return "success";
13    }
14
15    public String getMessage() {
```

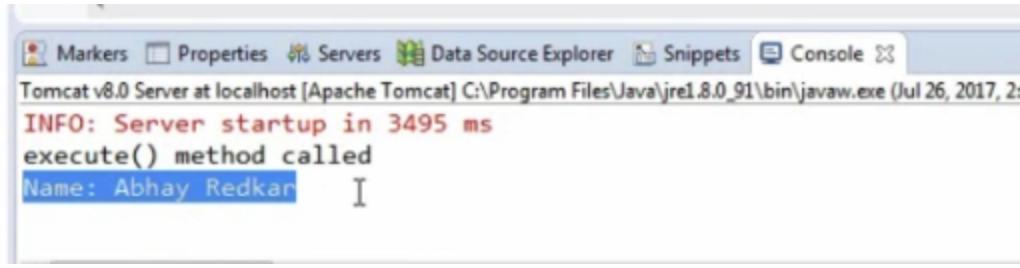


Your Name: Abhay Redkar

Submit



Those two print statements printed



Action interface and its fields. We can use them in return statements.

Modifier and Type	Field and Description
static String	ERROR The action execution was a failure.
static String	INPUT The action execution require more input in order to succeed.
static String	LOGIN The action could not execute, since the user most was not logged in.
static String	NONE The action execution was successful but do not show a view.
static String	SUCCESS The action execution was successful.

Here we are calling the SUCCESS field.

```

1 import com.opensymphony.xwork2.Action;
2
3 public class RegisterAction implements Action{
4
5     String firstName;
6     String lastName;
7     String gender;
8     String age;
9     String email;
10
11    public String execute()
12    {
13        System.out.println("execute() method called");
14        return SUCCESS;
15    }

```

We also have an ActionSupport class to extend our action class that will give access to more useful methods. Can be used a interceptor

**Class ActionSupport**

java.lang.Object  
com.opensymphony.xwork2.ActionSupport

All Implemented Interfaces:  
Action, ValidationAware, LocaleProvider, TextProvider, Validateable, Serializable

Direct Known Subclasses:  
DefaultActionSupport

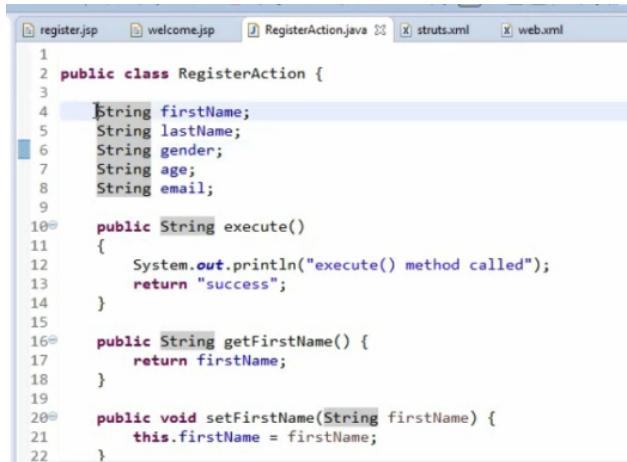
public class ActionSupport



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE struts PUBLIC
3     "-//Apache Software Foundation//DTD Struts Configuration 2.5//EN"
4     "http://struts.apache.org/dtds/struts-2.5.dtd">
5
6<struts>
7    <package name="register" extends="struts-default">
8        <action name="registerAction" class="RegisterAction">
9            <result name="success"/>/welcome.jsp</result>
10       </action>
11   </package>
12 </struts>
```

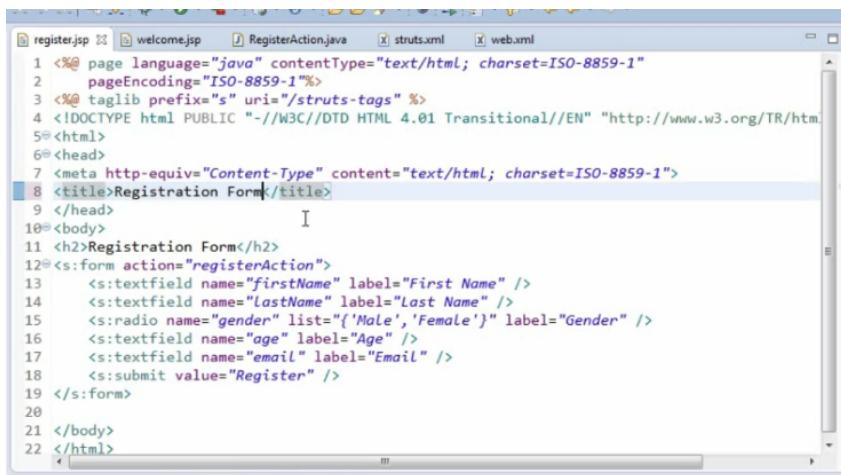
Struts package update registration action should return welcome page on success.

### Different UI components in RegistrationAction.java file



```
1
2 public class RegisterAction {
3
4     String firstName;
5     String lastName;
6     String gender;
7     String age;
8     String email;
9
10    public String execute()
11    {
12        System.out.println("execute() method called");
13        return "success";
14    }
15
16    public String getFirstName() {
17        return firstName;
18    }
19
20    public void setFirstName(String firstName) {
21        this.firstName = firstName;
22    }
```

### Different UI components in registration form - Registration.jsp



```
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2 pageEncoding="ISO-8859-1"%>
3 <%@ taglib prefix="s" uri="/struts-tags" %>
4 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/strict.dtd">
5 <html>
6 <head>
7 <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8 <title>Registration Form</title>
9 </head>
10<body>
11 <h2>Registration Form</h2>
12<s:form action="registerAction">
13     <s:textfield name="firstName" label="First Name" />
14     <s:textfield name="lastName" label="Last Name" />
15     <s:radio name="gender" list="{'Male','Female'}" label="Gender" />
16     <s:textfield name="age" label="Age" />
17     <s:textfield name="email" label="Email" />
18     <s:submit value="Register" />
19 </s:form>
20
21</body>
22</html>
```

And welcome page

```

<%@page contentType="text/html; charset=ISO-8859-1"%>
<%@taglib uri="http://struts.apache.org/tags-bean" prefix="bean"%>
<%@taglib uri="http://struts.apache.org/tags-html" prefix="html"%>
<%@taglib uri="http://struts.apache.org/tags-logic" prefix="logic"%>
<%@taglib uri="http://struts.apache.org/tags-nested" prefix="nested"%>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
        <title>Welcome</title>
    </head>
    <body>
        <h2>Welcome</h2>
        <bean:form>
            <bean:label value="First Name:" />
            <bean:property value="firstName"/><br/>
            <bean:label value="Last Name:" />
            <bean:property value="lastName"/><br/>
            <bean:label value="Gender:" />
            <bean:property value="gender"/><br/>
            <bean:label value="Age:" />
            <bean:property value="age"/>
            <bean:label value="Email:" />
            <bean:property value="email"/>
        </bean:form>
    </body>
</html>

```

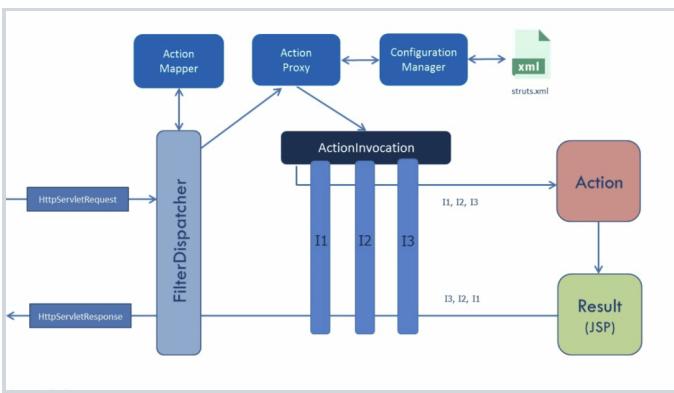
**Registration Form**

First Name: Abhay  
 Last Name: Redkar  
 Gender:  Male  Female  
 Age: 30  
 Email: abc@gmail.com

**Welcome**

First Name: Abhay  
 Last Name: Redkar  
 Gender: Male  
 Age: 30 Email: abc@gmail.com

## 15. Struts 2 Architecture in detail

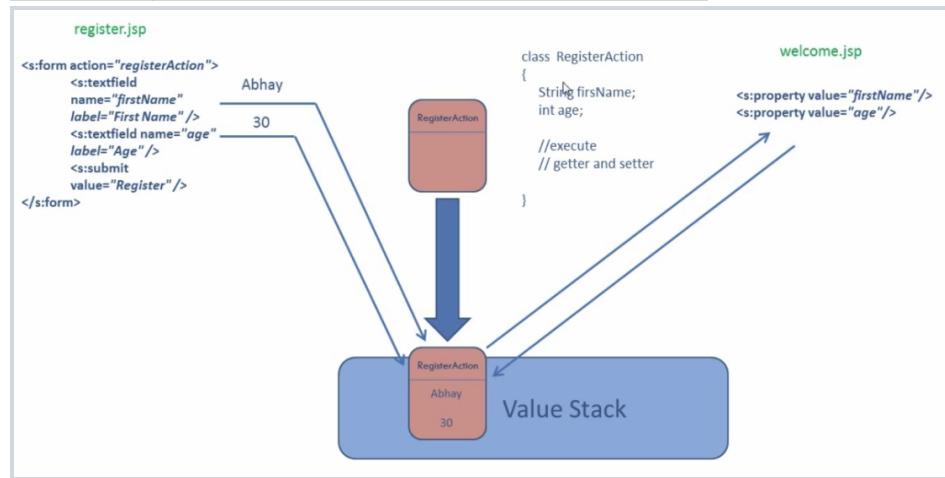


## 16. ValueStack and OGNL

## How values are going from jsp to action java and vice versa

- ValueStack is a storage area which holds all application specific data or data which is associated with the processing of a request.
- OGNL (Object-Graph Navigation Language) is an expression language which is used to access and manipulate data stored on the ValueStack.

All values are stored in the value stack. Welcome jsp to get values from the value stack using OGNL.  
Notice everywhere we maintained same variable names.



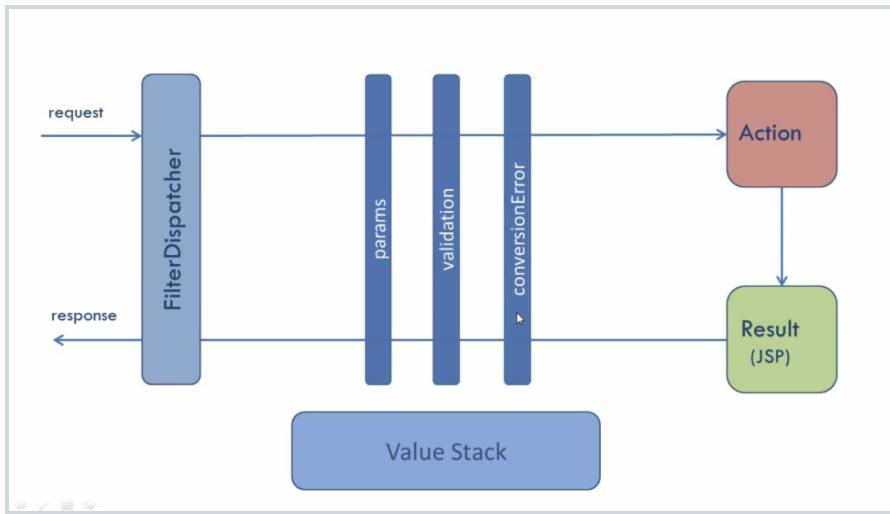
In case of objects - OGNL helps in the below form.

```
class User
{
    String firstName;
    int age;
    // getter and setter
}

class RegisterAction
{
    User user;
    //execute
    // getter and setter
}

<s:property value="#user.firstName"/>
<s:property value="#user.age"/>
```

17. Interceptors : Addresses the cross cutting concerns - which are not related to business.



A screenshot of an IDE showing the code for `RegisterAction.java`. The code extends `ActionSupport` and defines fields for `firstName`, `lastName`, `gender`, `age`, and `email`. It contains an `execute` method that prints "execute() method called" and returns "success". It also has `getFirstName` and `setFirstName` methods.

```

import com.opensymphony.xwork2.ActionSupport;
public class RegisterAction extends ActionSupport{
    String firstName;
    String lastName;
    String gender;
    Integer age;
    String email;
    public String execute() {
        System.out.println("execute() method called");
        return "success";
    }
    public String getFirstName() {
        return firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
}

```

Extend `ActionSupport` class.

And override validate method as shown below.

```

public void validate()
{
    if (firstName.equals(""))
       addFieldError("firstName", "First name is required.");
    if (lastName.equals(""))
       addFieldError("lastName", "Last name is required.");
    if (gender == null)
       addFieldError("gender", "Gender is required.");
}

```

```
66
67     if (gender == null) {
68         addFieldError("gender", "Gender is required.");
69     }
70
71     if (age == null) {
72         addFieldError("age", "Age is required.");
73     }
74     else if(age <= 18)
75     {
76         addFieldError("age", "Age should be above 18.");
77     }
78
79     if (email.equals(""))
80         addFieldError("email", "Email is required.");
81 }
```

```
register.jsp    welcome.jsp    RegisterAction.java    *struts.xml    web.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE struts PUBLIC
3   "-//Apache Software Foundation//DTD Struts Configuration 2.5//EN"
4   "http://struts.apache.org/dtds/struts-2.5.dtd">
5
6<struts>
7  <package name="register" extends="struts-default">
8    <action name="registerAction" class="RegisterAction">
9      <result name="success"/>/welcome.jsp</result>
10     <result name="input"/>/register.jsp</result>
11   </action>
12 </package>
13 </struts>
```

In case of any error page should be in same register.jsp only

The screenshot shows a web browser window titled "Registration Form". The URL in the address bar is "localhost:8080/RegistrationApplication/registerAction.action". The page displays a registration form with several validation errors:

- First name is required.** (highlighted in red)
- Last name is required.** (highlighted in red)
- Gender is required.** (highlighted in red)
- Age is required.** (highlighted in red)
- Email is required.** (highlighted in red)

The form fields include:  
First Name: (empty)  
Last Name: (empty)  
Gender: Male (radio button selected)  
Age: (empty)  
Email: (empty)  
A "Register" button is at the bottom right.

Registration Form

*First Name:* Abhay

*Last Name:* Redkar

**Gender is required.**

*Gender:*  Male  Female

**Age is required.**

*Age:*

**Email is required.**

*Email:*

**Register**

Registration Form

*First Name:* Abhay

*Last Name:* Redkar

*Gender:*  Male  Female

**Age should be above 18.**

*Age:* 13

*Email:* abhay@gmail.com

**Register**

Something with validation xml. The file name must match with the action class name and followed by validation.xml as shown below.

```

File Edit Source Navigate Search Project Run Window Help
Project... Navigator... Properties...
register.jsp welcome.jsp RegisterAction.java struts.xml web.xml RegisterAction-validation.xml
register.jsp welcome.jsp RegisterAction.java struts.xml web.xml RegisterAction-validation.xml
1 <xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE validators PUBLIC "-//Apache Struts//XWork Validator 1.0.3//EN"
3 "http://struts.apache.org/dtds/xwork-validator-1.0.3.dtd">
4
5<validators>
6   <validator type="requiredstring">
7     <param name="fieldName">firstName</param>
8     <message>First name is required</message>
9   </validator>
10  <validator type="requiredstring">
11    <param name="fieldName">lastName</param>
12    <message>Last name is required</message>
13  </validator>
14  <validator type="required">
15    <param name="fieldName">gender</param>
16    <message>Gender is required</message>
17  </validator>
18  <validator type="required">
19    <param name="fieldName">age</param>
20    <message>Age is required</message>
21  </validator>
22  <validator type="int">
23    <param name="fieldName">age</param>
24  </validator>

```

```

14<validator type="required">
15    <param name="fieldName">gender</param>
16    <message>Gender is required</message>
17</validator>
18<validator type="required">
19    <param name="fieldName">age</param>
20    <message>Age is required</message>
21</validator>
22<validator type="int">
23    <param name="fieldName">age</param>
24    <param name="min">18</param>
25    <message>Age should be above ${min}</message>
26</validator>
27<validator type="requiredstring">
28    <param name="fieldName">email</param>
29    <message>Email is required</message>
30</validator>
31<validator type="email">
32    <param name="fieldName">email</param>
33    <message>Must provide a valid email</message>
34</validator>
35</validators>

```

Registration Form

First Name: Abhay  
Last Name: Redkar  
Gender:  Male  Female  
Age: 20  
**Must provide a valid email**  
Email: abhay@ @gmail.com  
Register

Welcome

First Name: Abhay  
Last Name: Redkar  
Gender: Male  
Age: 20 Email: abhay@gmail.com

```

26</validator>
27<validator type="requiredstring">
28    <param name="fieldName">email</param>
29    <message>Email is required</message>
30</validator>
31<validator type="email">
32    <param name="fieldName">email</param>
33    <message>Must provide a valid email</message>
34</validator>
35<field name="email">
36    <field-validator type="requiredstring">
37        <message>Email is required</message>
38    </field-validator>
39    <field-validator type="email">
40        <message>Must provide a valid email</message>
41    </field-validator>
42</field>
43</validators>

```

Field validator will help to group multiple validators into groups as shown above. In the above case from line number 27 to 34 not required. Lines 35 to 42 covered indirectly.