STRUTS 1.3

Rajeev Gupta MTech CS
Rgupta.metch@gmail.com

Agenda

- Why struts1
- Struts 1.x Architecture , Struts components
- Revisiting Hello world with validation and resource bundle
- DynaActionForm
- Struts ForwardAction
- Multiple configuration files example
- DispatchAction
- Using RequestProcessor for ensure loggin user
- Validation Framework

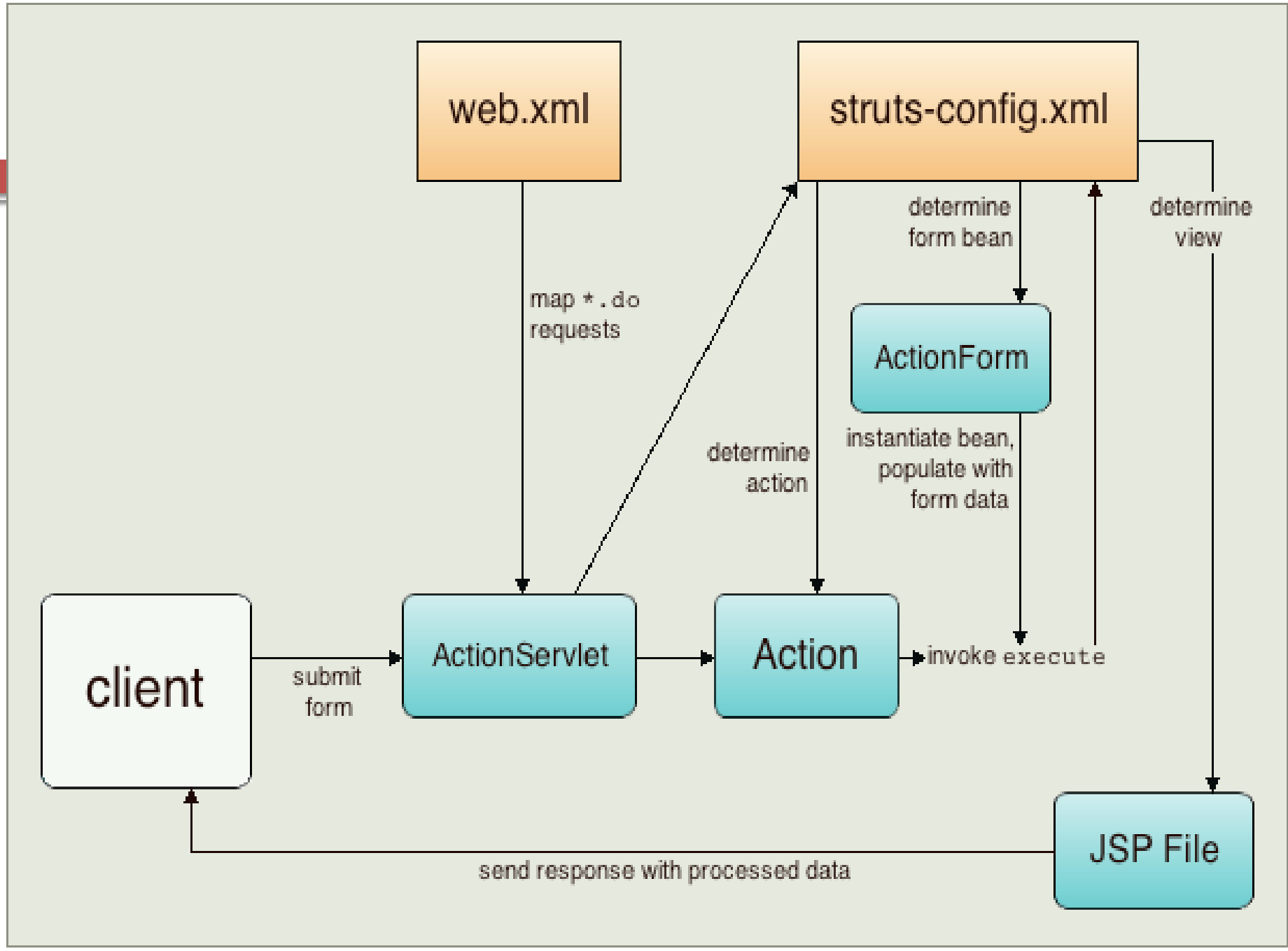
Why Struts1?

Why Struts?

- Apache's open source web application model view controller framework project
- Why Struts ?
  - ilities of SW projects: Flexibility, Maintainability, Extensibility

- Problem with our MVC servlet jsp application
  - A single MVC app will have many models, views, and controllers
  - application grows no of controller increases...problem

Struts 1.x Architecture , Struts components


web.xml
struts-config.xml
map *.do
requests
determine
form bean
determine
view
ActionForm
instantiate bean,
populate with
form data
determine
action
client
submit
form
ActionServlet
Action
invoke execute
JSP File
send response with processed data

Hello World

Steps hello world

- Step 1: Configure Front controller in web.xml, define url pattern
- Step 2: Write the Action class
  - public class LogonAction extends Action {}
- Step 3: Write the matching ActionForm
  - public class LogonForm extends ActionForm {}
- Step 4: Create struts-config.xml map backing bean and back controller

Step 1: Configure Front controller in web.xml, define url pattern

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app id="WebApp_ID" version="2.4"
    xmlns="http://java.sun.com/xml/ns/j2ee" xmlns:xsi="http://www.w3.org/2001/
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xm
    <display-name>struts1-hello-01</display-name>
    <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>2</load-on-startup>
    </servlet>

    <servlet-mapping>
        <servlet-name>action</servlet-name>
        <url-pattern>*.do</url-pattern>
    </servlet-mapping>

    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>
</web-app>
```

- ActionServlet is provided by the framework.
- The Servlet must be mapped in the web.xml file.
  - Must have configuration file mapped
- Lastly, Map the *.do URI to the Action Servlet
- We need to Configure front controller i.e. ActionServlet, we need to provide an param name config that take struts-config.xml

Step 2: Write the Action class

- Step 2: Write the Action class
  - public class LogonAction extends Action {}
- We need to write our action classes, they are back controllers

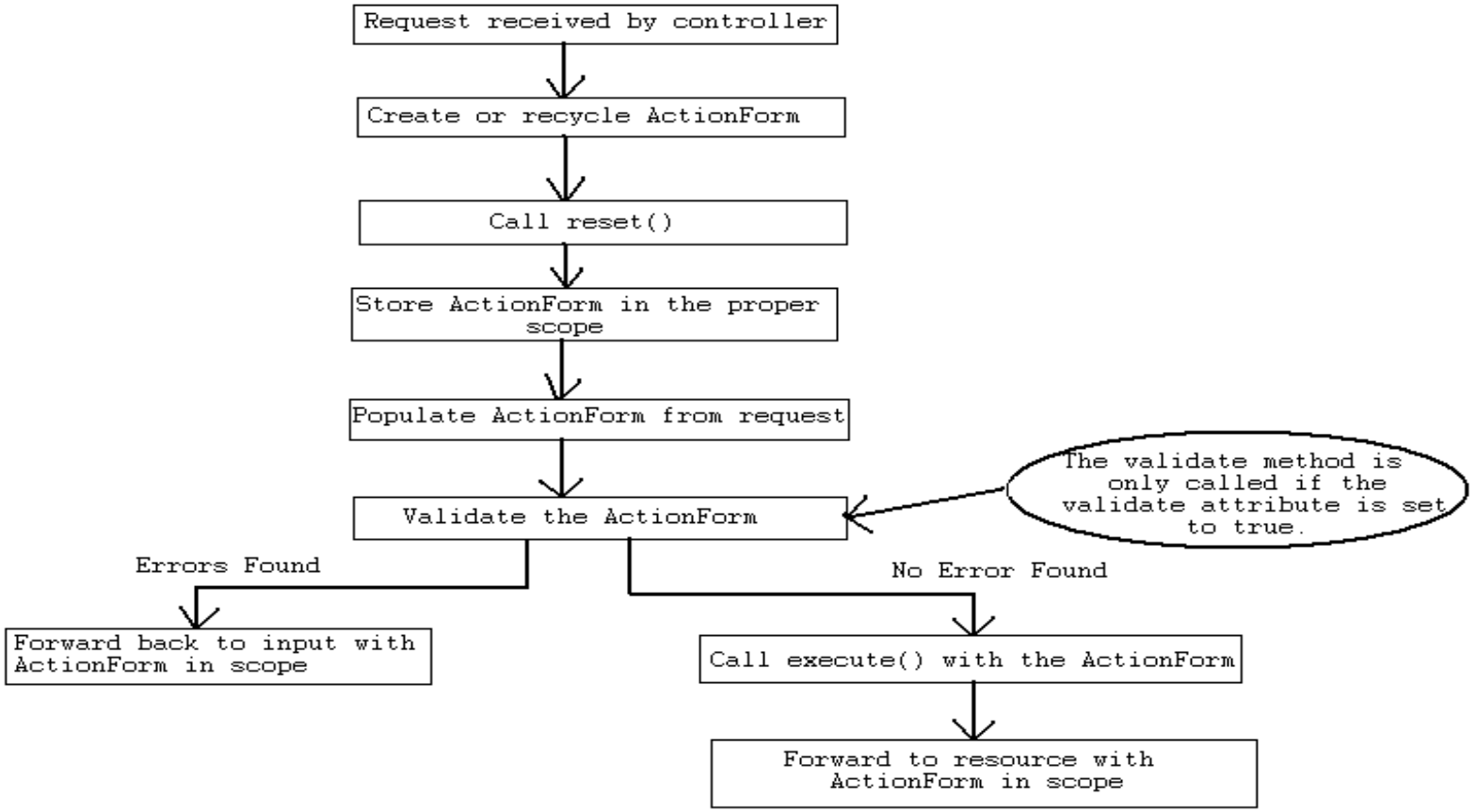
```
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;

import com.forms.LoginForm;

public class LoginAction extends org.apache.struts.action.Action {

    public ActionForward execute(ActionMapping mapping, ActionForm form,
            HttpServletRequest request, HttpServletResponse response)
            throws Exception {
        LoginForm loginForm = (LoginForm) form;
        if (loginForm.getUserName().equals(loginForm.getPassword())) {
            return mapping.findForward("success");
        } else {
            return mapping.findForward("failure");
        }
    }
}
```

Step 3: Write the matching ActionForm

```
import org.apache.struts.action.ActionForm;

public class LoginForm extends ActionForm {
    private static final long serialVersionUID = 1L;

    private String userName;
    private String password;
    public String getUserName() {
        return userName;
    }
    public void setUserName(String userName) {
        this.userName = userName;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }


}
```

- Step 3: Write the matching ActionForm
  - public class LogonForm extends ActionForm {}
- ActionForm is backing form bean
- It is used to collect data from the view and supply it to the back controller
- Different object of Backing form bean is used for each new request, thread safe

The lifecycle of an ActionForm



Step 4: Create struts-config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts-config PUBLIC
          "-//Apache Software Foundation//DTD Struts Configuration
          "http://struts.apache.org/dtds/struts-config_1_3.dtd">

<struts-config>

    <form-beans>
        <form-bean name="LoginForm" type="com.forms.LoginForm" />
    </form-beans>

    <action-mappings>
        <action input="/login.jsp" name="LoginForm" path="/Login"
            scope="request" type="com.actions.LoginAction">
            <forward name="success" path="/success.jsp" />
            <forward name="failure" path="/failure.jsp" />
        </action>
    </action-mappings>

</struts-config>
```

- Step 4: Create struts-config.xml
  - It map backing bean and back controller

Understanding Strut-config.xml

```
<action path="/logon“
  type=“c com.actions LogonAction”

      name="LogonForm"

.
      <forward name="failure"

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

</action>
```

For requests that hit URL=“/logon”
The frame work will invoke execute() on an instance of class com.actions.LogonAction

Store request parameters in form variable “LogonForm” which is defined in another location in the xml document.

If the logical name returned by perform() is “failure” go to page “/failure.jsp”

If the Logical name returned by perform() is “success” go to “/success.jsp”

What one should know in JSP ?

DTO

```
public class Employee {
    private String id;
    private String name;
    private String address;
```

DAO

```
public interface EmployeeDao {
    public List<Employee> getAllEmployee();
    public void addEmployee(Employee employee);
}
```

DAO Implementation

```
public class EmployeeDaoImp implements EmployeeDao{
    private List<Employee>employeeList=new ArrayList<Employee>()

    public EmployeeDaoImp(){
    public List<Employee> getAllEmployee() {

    public void addEmployee(Employee employee) {

}
```

```
public class EmployeeController extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request,
            HttpServletResponse response)
    throws ServletException, IOException {

        EmployeeDao dao=new EmployeeDaoImp();

        List<Employee>employeeList=dao.getAllEmployee();
        request.setAttribute("employeeList", employeeList);

        request.getRequestDispatcher("showAll.jsp").
        forward(request, response);
```

```
All Employee Records :)<br/>
<c:forEach items="${employeeList}" var="emp">
    ${emp.id } , ${emp.name }, ${emp.address }<br/>
</c:forEach>
```

- what is the best way to present data in view layer?
  - Don't use scriptlet, perfer JSTL and custom tag
  - In case of struts use struts tags
  - **Example: Employee Application, developed in Servlet/JSP session**

Using Struts logic and bean tags (Introduction)

```java
import javax.servlet.http.HttpServletRequest;
public class AllEmployeeAction extends Action {

    private List<Employee>employeeList=null;
    @Override
    public ActionForward execute(ActionMapping mapping, ActionForm form,
            HttpServletRequest request, HttpServletResponse response)
            throws Exception {

        EmployeeDao dao=new EmployeeDaoImp();
        employeeList=dao.getAllEmployee();
        request.setAttribute("employeeList", employeeList);

        return mapping.findForward("success");

    }

}
```

```jsp
<body>
    <table>
        <thead>
            <tr>
                <td>Employee ID</td>
                <td>Employee Name</td>
                <td>Employee Address</td>
            </tr>
        </thead>
        <tbody>
                <logic:iterate name="employeeList" id="emp">
                <tr>
                    <td><bean:write name="emp" property="id" /></td>
                    <td><bean:write name="emp" property="name" /></td>
                    <td><bean:write name="emp" property="address" /></td>
                </tr>
                </logic:iterate>
        </tbody>
```

Employee ID	Employee Name	Employee Address
121	gunika	delhi
1	rajiv	delhi
81	keshav	noida

```xml
<action path="/show"
     type="com.actions.AllEmployeeAction">
    <forward name="success" path="/showAllEmployee.jsp" />
</action>
```

Revisiting Hello world with validation and resource bundle

```
import org.apache.struts.action.*;
public class LogonAction extends Action {
    public ActionForward  execute(
                                ActionMapping mapping,
                                 ActionForm form,
                                 HttpServletRequest request,
                                 HttpServletResponse response)
                throws IOException, ServletException
    {
        LogonForm theForm = (LogonForm)form;
        String forward="failure";


        if(theForm.getUserName().equals("borcon"))
        {
          forward="success";
        }
        return mapping.findForward(forward);
    }
}
```

Action class's perform is invoked by Action Servlet

Action Form

```
import org.apache.struts.action.ActionForm;
public class LogonForm extends ActionForm {
     private String userName;
     private String password;


  public String getUserName() {
    return userName;
  }
  public void setUserName(String userName) {
    this.userName = userName;
  }
  public void setPassword(String password) {
    this.password = password;
  }
  public String getPassword() {
    return password;
  }
}
```

- Action Form has properties which map to the HTML page.
- Additionally the form can:
  - Reset
  - Validate

Strut Powered JSP

```
<head>
<title>Logon Form</title>
</head>
<body bgcolor="white">
<html:errors/>
<html:form action="logon" focus="userName">
    User Name:  <html:text property="userName" size="16" maxlength="16"/>
    password:    <html:password property="password" size="16" maxlength="16"
<html:submit property="submit" value="Submit"/>
<html:reset/>
</html:form>
</body>
</html:html>
```

Steps Hello World

- Step 1: Build your JSP in HTML format
  - It's back to editing code
- Step 2: Convert to Struts format
- Step 3: Write the matching ActionForm
  - public class LogonForm extends ActionForm {}
- Step 4: Write the Action class
  - public class LogonAction extends Action {}
- Step 5: Register the entries in struts-config.xml

```
<action path="/logon"
        type="com.codementor.struts.LogonAction"
       name="logonForm"
       input="/Logon.jsp" >
  <forward name="success" path="/Success.jsp" />
  <forward name="failure" path="/Failure.jsp" />
</action>
```

JSP Pages

- Three Pages
  - Login Page
  - Success Page
  - Failure Page

Logon Page – HTML Version

```
<html>
<head>
<title>Login Form</title>
</head>
<body bgcolor="#ffffff">
<form action="logon.do">
      User Name: <input type="text" name="userName" size="16"
    maxlength="16"/><br />
      Password: <input type="text" name="password" size="16"
    maxlength="16"/><br />
<input type="submit" name="Submit" value="Submit">
```

Logon Page – Struts

```
<%@ taglib uri="/WEB-IN                    d"
   prefix="html" %>
<html:html>
<head>
<title>Login Form</title>
</head>
<body bgcolor="#ffffff">
<html:form action="logon.do"
   focus="userName">
<br>
   User Name: <html:text maxlength="16"
```

Added a cool feature

Success and Failure Pages

- Success.jsp

```
<html>
<head>
<title>
Success
</title>
</head>
<body bgcolor="#ffffff">
<h1>
Successful Login
</h1>
</body>
</html>
```

- Failure.jsp

```
<html>
<head>
<title>
Failure
</title>
</head>
<body bgcolor="#ffffff">
<h1>
Failed Login
</h1>
</body>
</html>
```

Create Form Bean Class

```
zpackage com.codementor;
import org.apache.struts.action.ActionForm;

public class LogonForm extends ActionForm {
  private String password;
  private String userName;

  public String getPassword() {
    return password;
  }
  public void setPassword(String password) {
    this.password = password;
```

Form Bean Config

```
<struts-config>
  <form-beans>
    <form-bean      name="logonForm"
                    type="com.codementor.LogonForm" />
  </form-beans>
…
</struts-config>
```

Action Class

```
package com.codementor;
import org.apache.struts.action.*;
import javax.servlet.http.*;


public class LogonAction extends Action {
  public ActionForward execute(ActionMapping mapping,
                               ActionForm form,
                               HttpServletRequest request,
                               HttpServletResponse response) {


    LogonForm logonForm = (LogonForm) form;
    String forward = "failure";


    if(logonForm.getUserName().equals("mentor"))
    {
      forward = "success";
    }
    return mapping.findForward(forward);
  }
}
```

Action Class Config

```
…
 <action-mappings>
    <action             path="/logon"
                         type="com.codementor.LogonAction"
                         name="logonForm"
                        input="/Login.jsp"
                        scope="request" />
  </action-mappings>
</struts-config>
```

Map The Forwards

```
…
 <action-mappings>
    <action       path="/logon"
                   type="com.codementor.LogonAction"
                   name="logonForm"
                  input="/Login.jsp"
                  scope="request" >
     <forward name="success" path="/Success.jsp" />
      <forward name="failure" path="/Failure.jsp" />
    </action>
  </action-mappings>
</struts-config>
```

Struts in web.xml

```
<web-app>
<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>
   <init-param>
     <param-name>debug</param-name>
     <param-value>2</param-value>
   </init-param>
   <load-on-startup>2</load-on-startup>
 </servlet>
 <servlet-mapping>
   <servlet-name>action</servlet-name>
   <url-pattern>*.do</url-pattern>
 </servlet-mapping>
```

Sets the logging level for struts

Loads ActionServlet on startup
** important if there is a JSP page which could be referenced from the Client **

Run The WebApp