**Experiment No. 1**

**Title: Demonstrate the struts2 architecture through an application to display any text message.**

**Batch:B1 RollNo.:1514033 ExperimentNo.:1**

### Aim: Demonstrate the struts2 architecture through an application to display any text message.

### --------------------------------------------------------------------------------------------

**Resources needed:**Eclipse,Tomcat,Struts2.

### Theory:

### Download struts2 from <https://struts.apache.org/download.cgi#struts258>

### Create new dynamic Web application in Eclipse IDE

### Give any project name of your choice. Change Project Location if you want.

### Set target runtime environment as ‘Apache Tomcat v7.0’ -> Next->Next

### Context root gives the project location

### Rename content directory as web

### Check Generate web.xml deployment descriptor.

### Click on Finish

### Right click on the project -> properties -> java build path -> add libraries -> user library -> Click next

### Click on User Libraries to create a new Library. A window gets open click on New

### Give a suitable name for library

### Add following jar files in that library by clicking ‘Add jars’ and click OK

### commons-fileupload-1.3.1.jar

### commons-io-2.2.jar

### commons-lang3-3.2.jar

### commons-logging-api-1.1.jar

### freemarker-2.3.19.jar

### javassist-3.11.0.GA.jar

### ognl-3.0.6.jar

### struts2-core-2.3.20.1.jar

### xwork-core-2.3.20.1.jar

### Right click on the project name -> properties ->deployment assembly ->add -> java build path entries ->next-> select the created library ->Apply->OK.

### Create new Class by right clicking on src folder, under Java Resources.

### Enter the code for execute method in it.

### Create.xml file of any name, in the src folder, by right clicking on src, and clicking on New-> XML File.

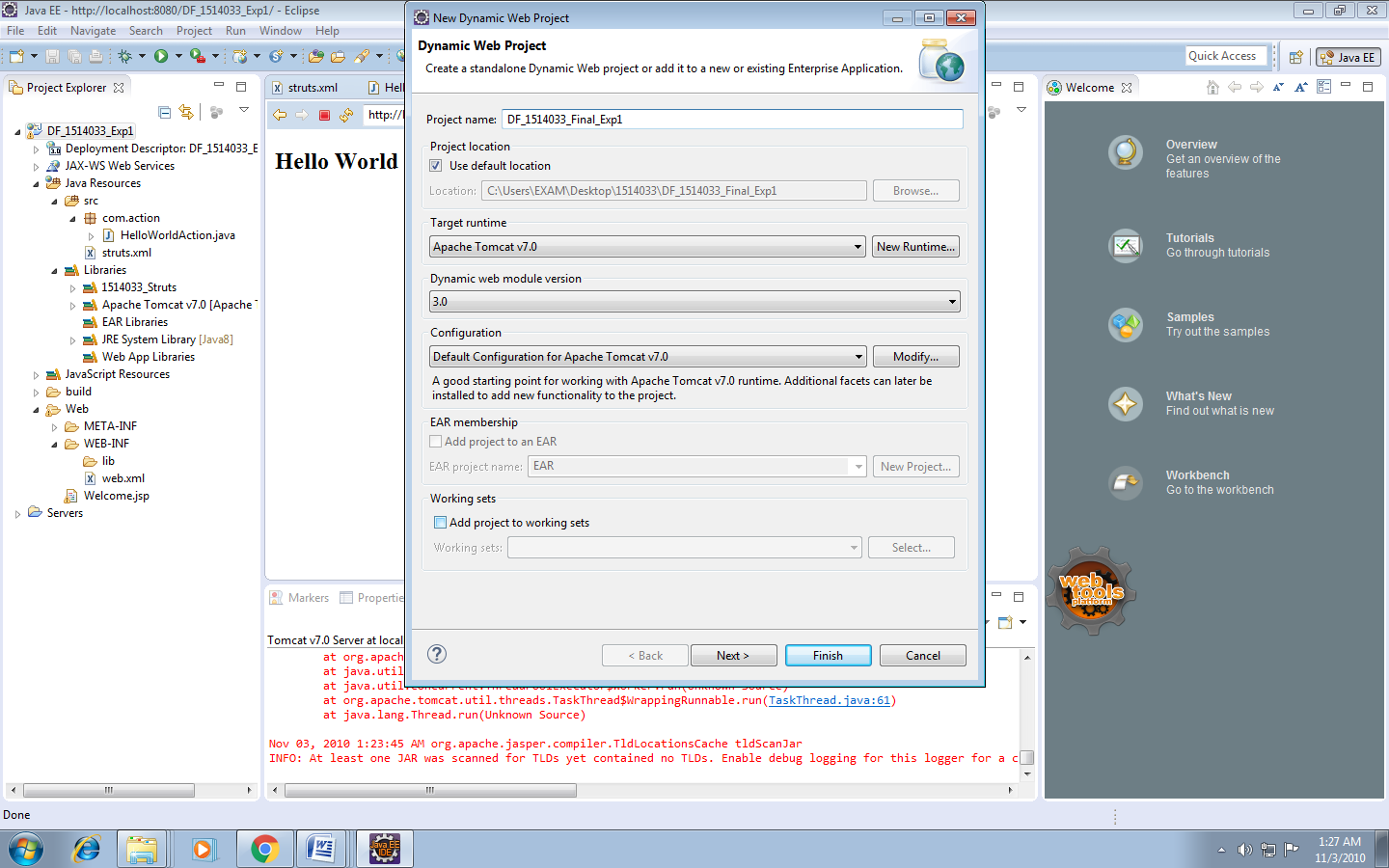
### enter the code in the web.xml file.

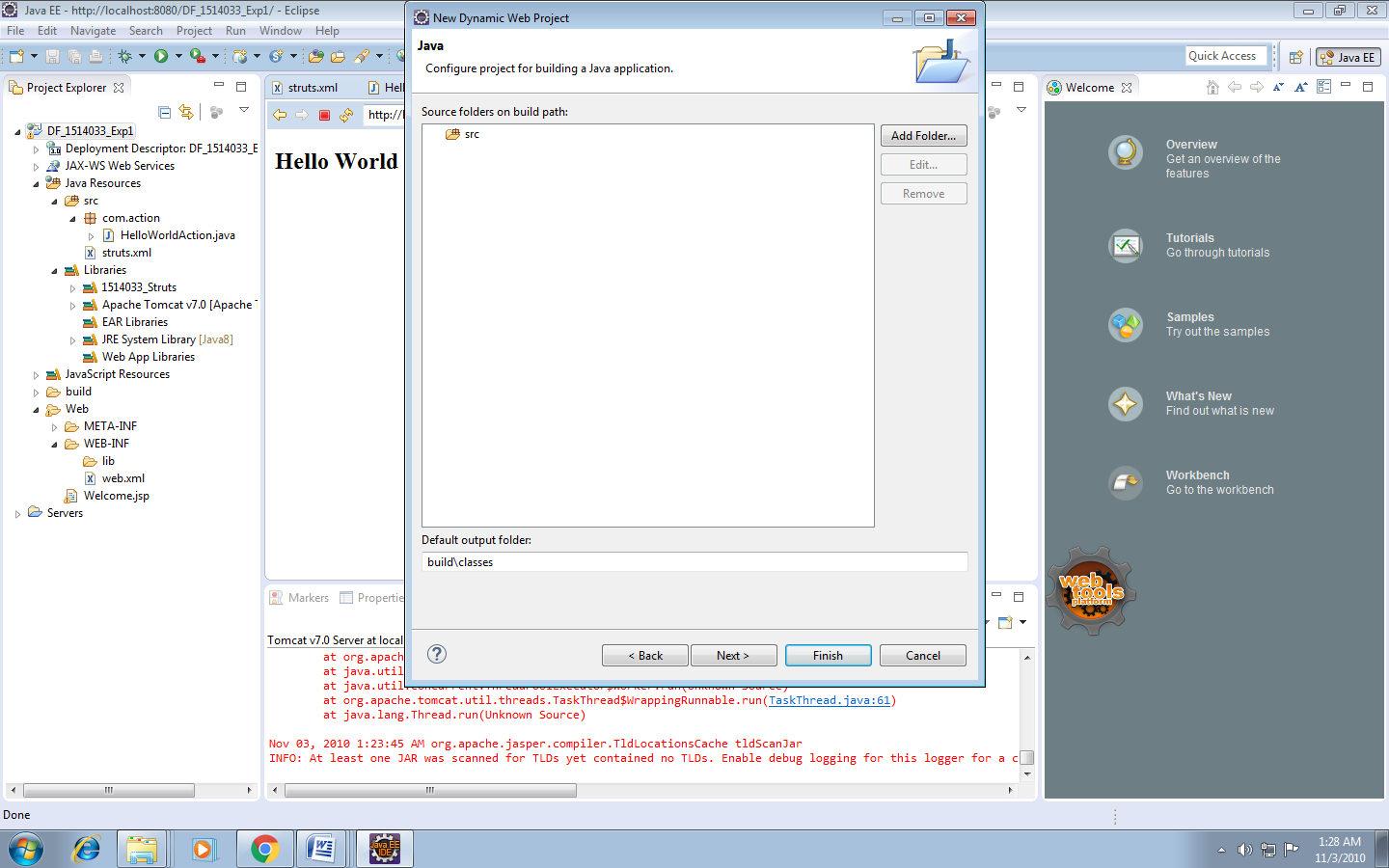
### Create new JSP file in WEB-INF under WebContent, by right clicking on WEB-INF, New->JSP file, and enter the code.

### : Run the project on Tomcat server.

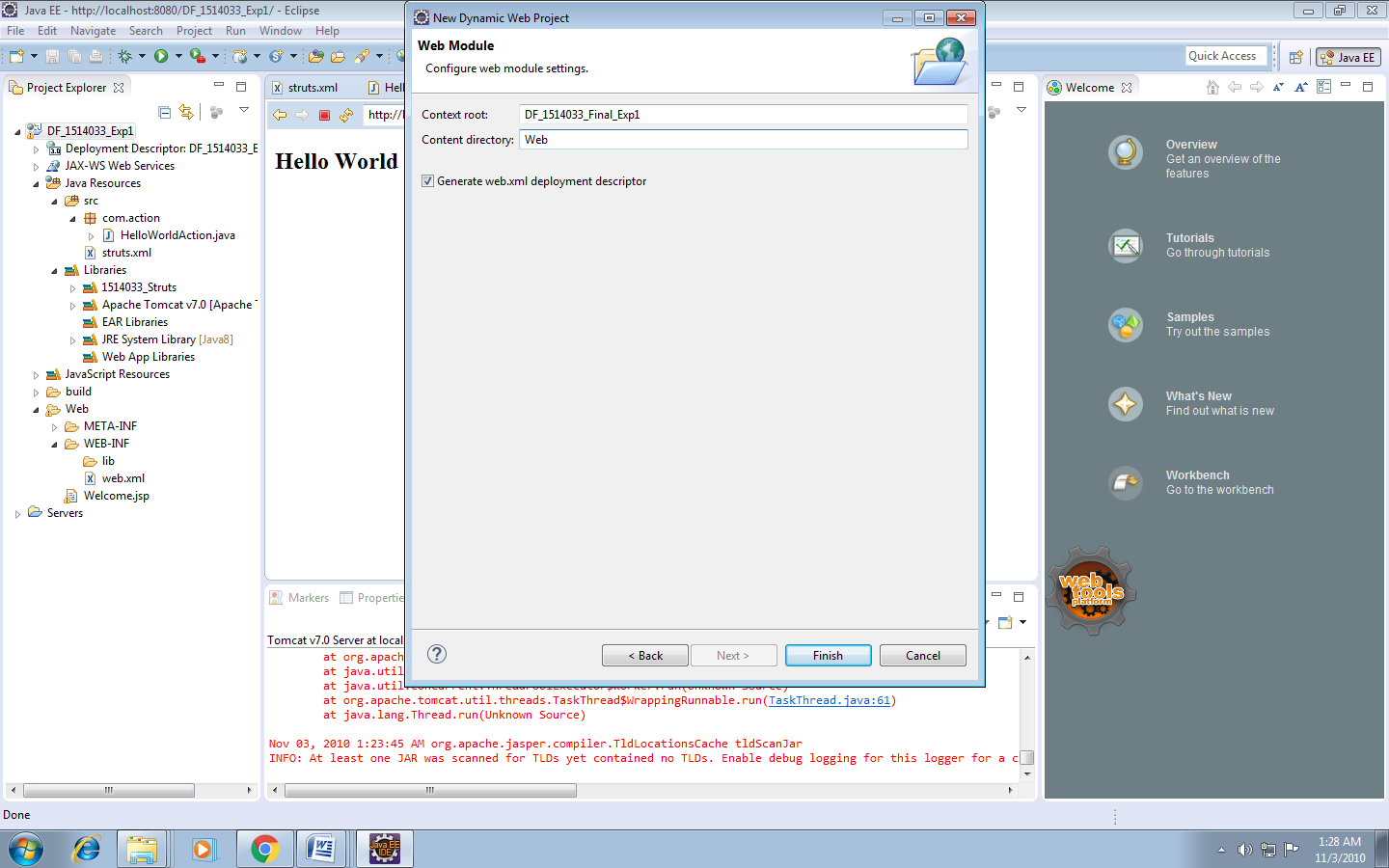
### Results: (Screen shots of application development steps, program code and web browser displaying the specified message.)

### Create new dynamic web project. Give any name and add server to it.

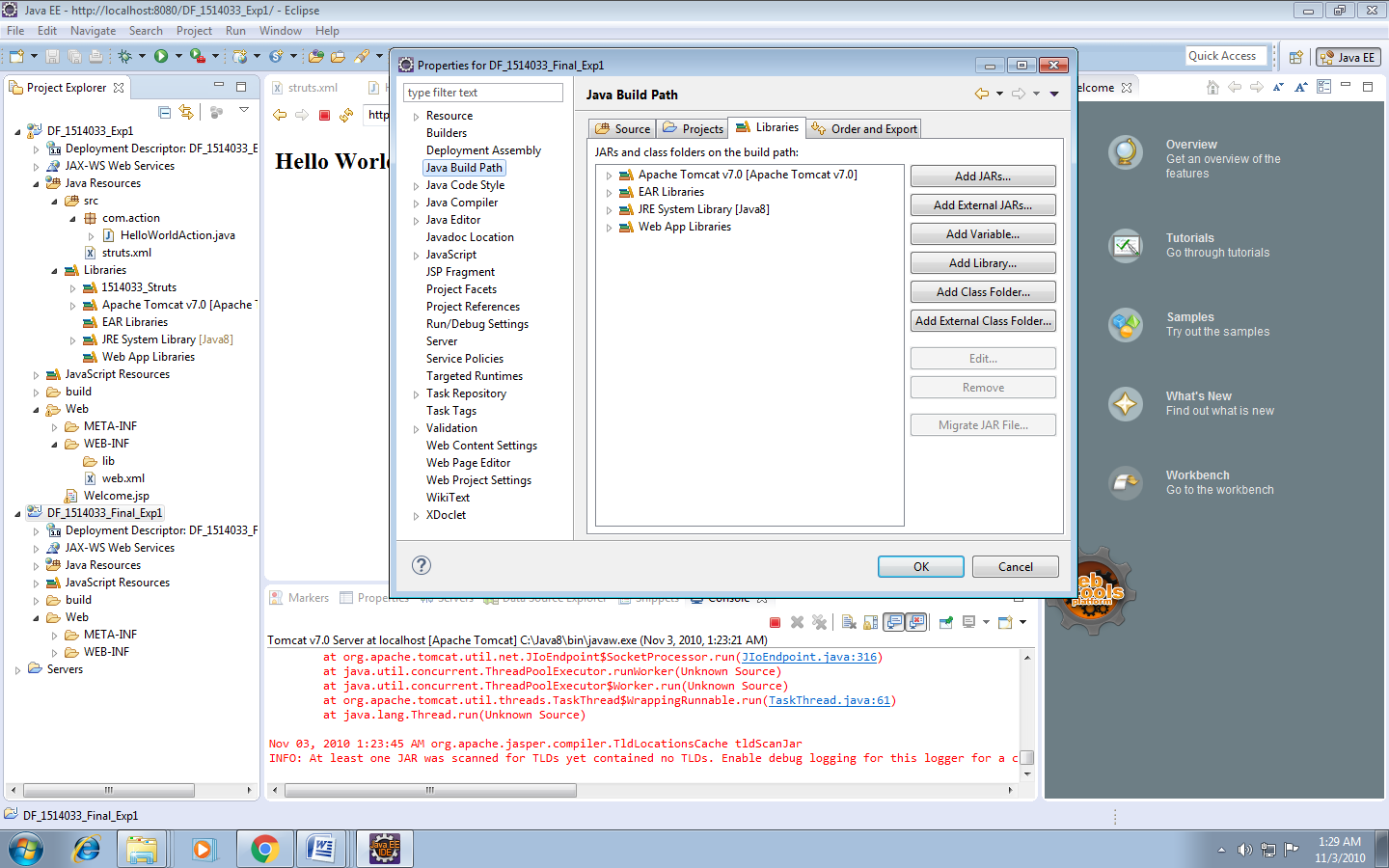




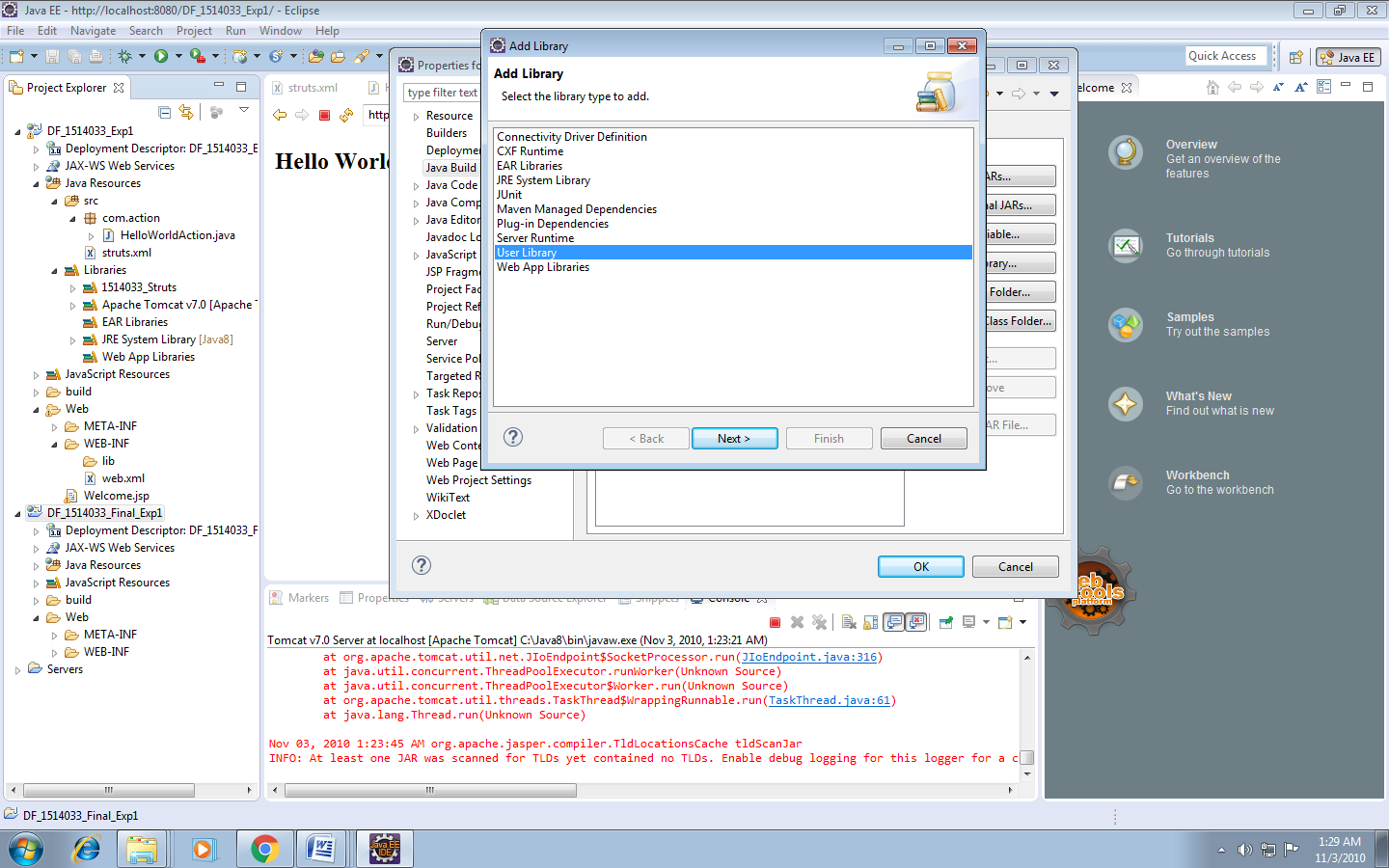
Tick mark on generate web deployment descriptor and give name of content directory as “Web”



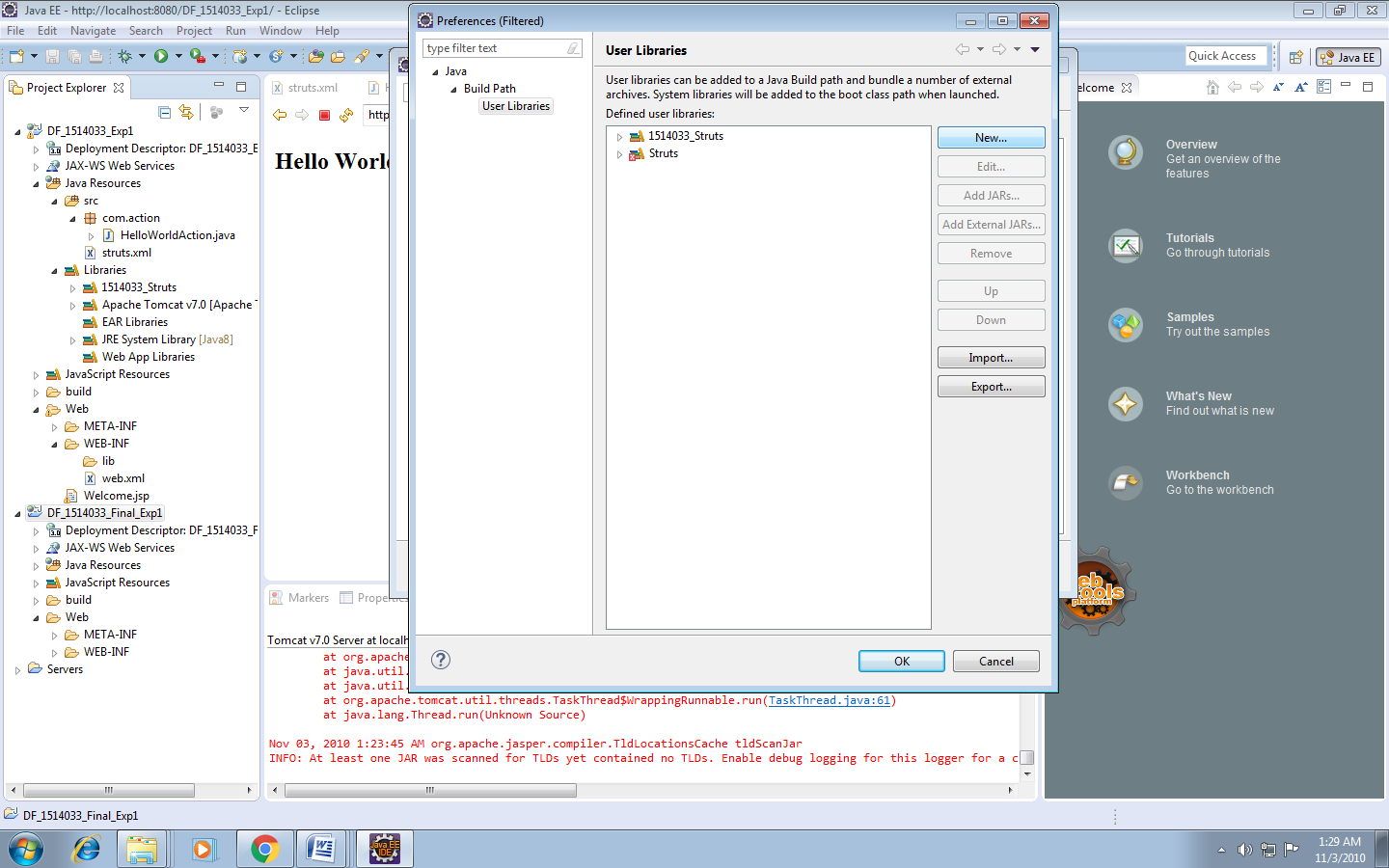
Right click on project .Go to properties. Go to Java Build Path. Go to add library.



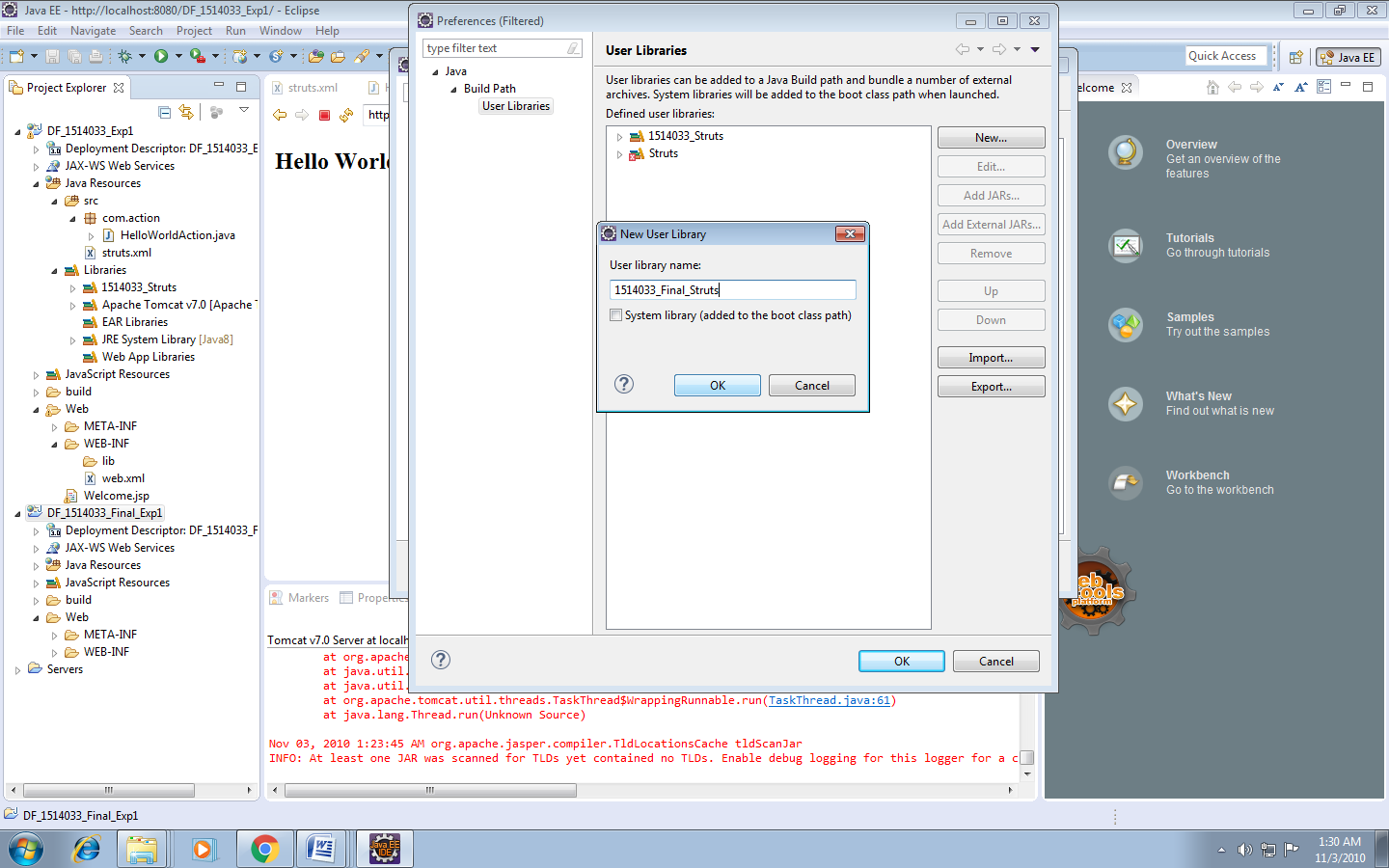
Select “User Library”



Select new



Give library name to library



Add jar to this library

### commons-fileupload-1.3.1.jar

### commons-io-2.2.jar

### commons-lang3-3.2.jar

### commons-logging-api-1.1.jar

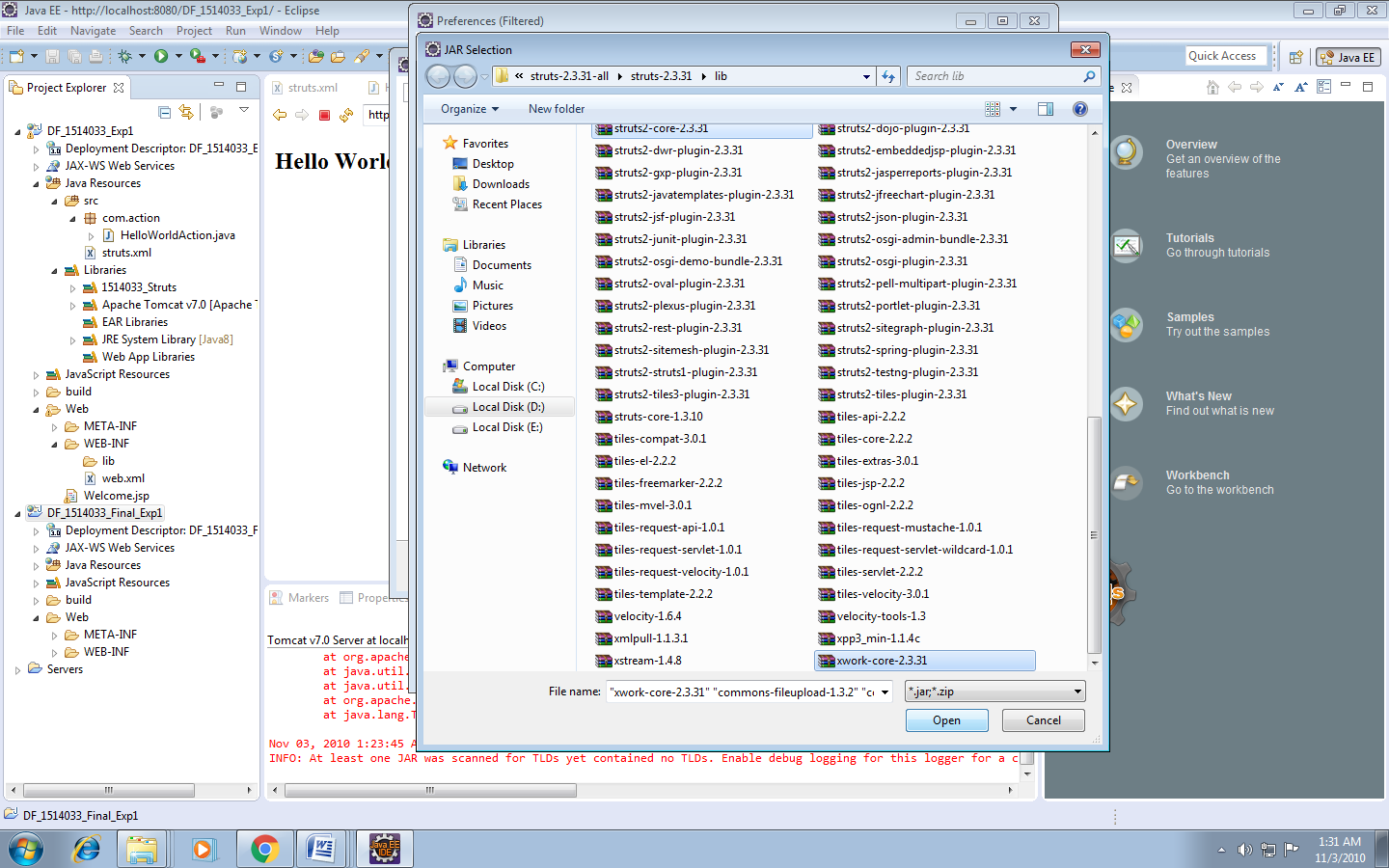
### freemarker-2.3.19.jar

### javassist-3.11.0.GA.jar

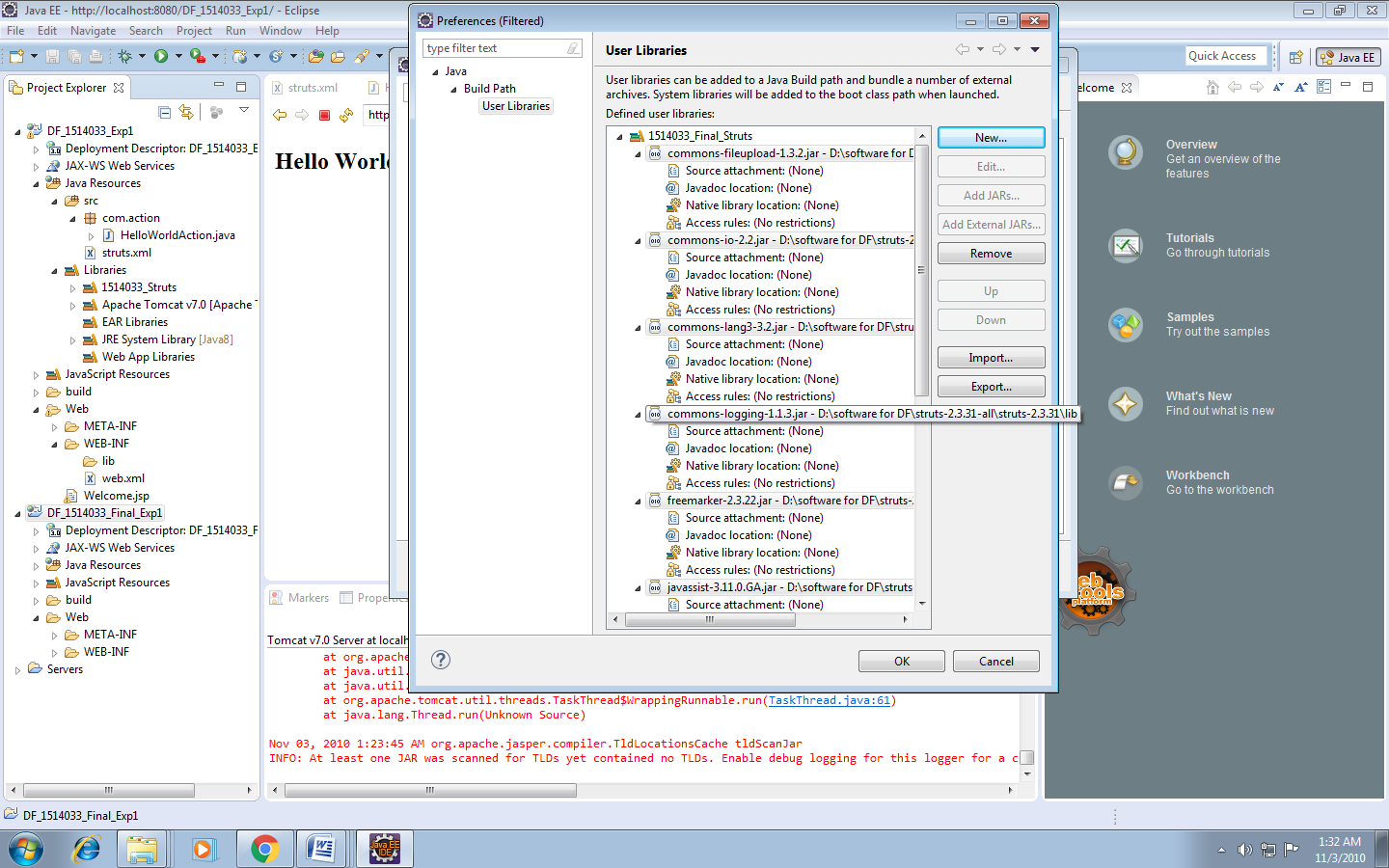
### ognl-3.0.6.jar

### struts2-core-2.3.20.1.jar

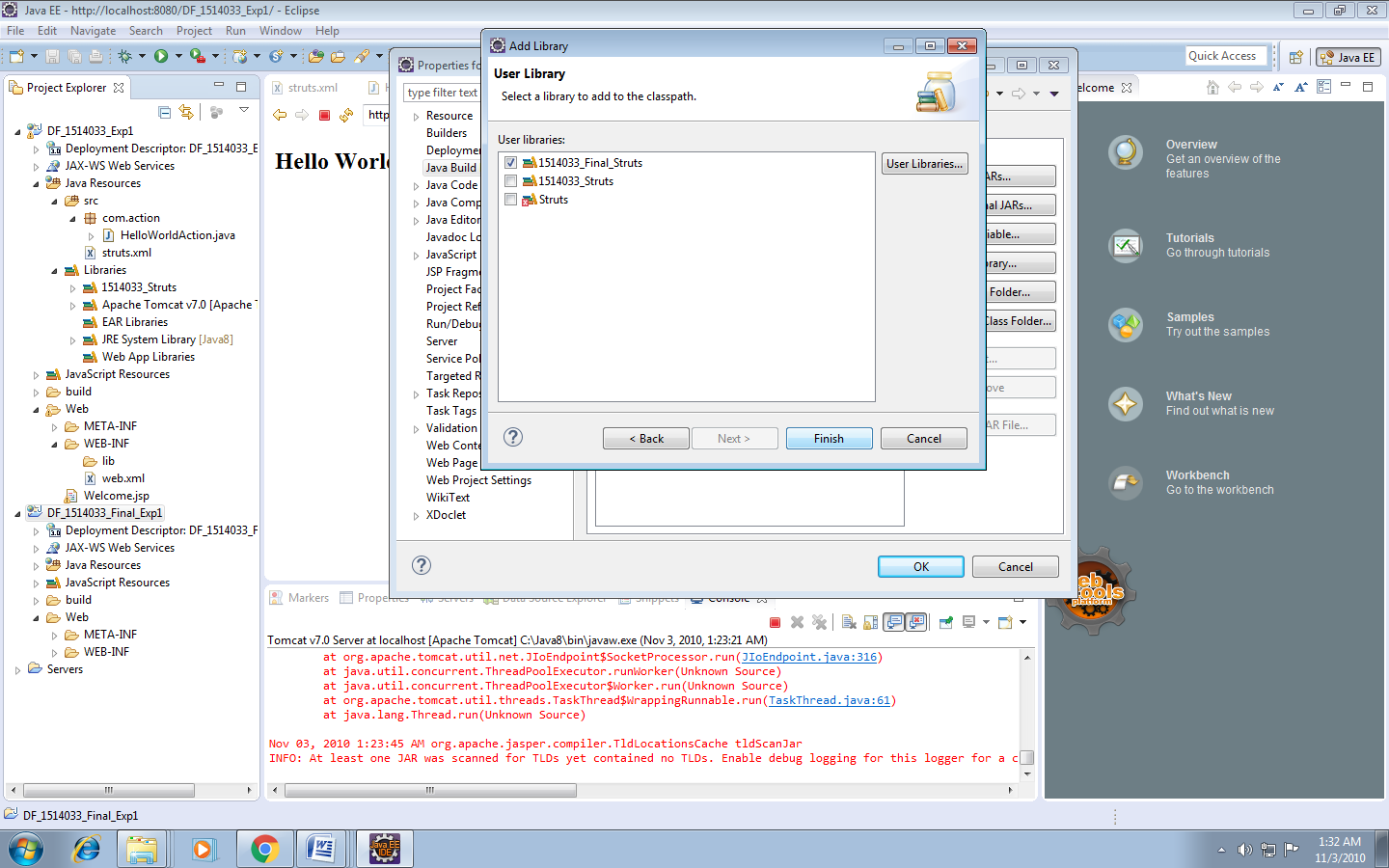
### xwork-core-2.3.20.1.jar



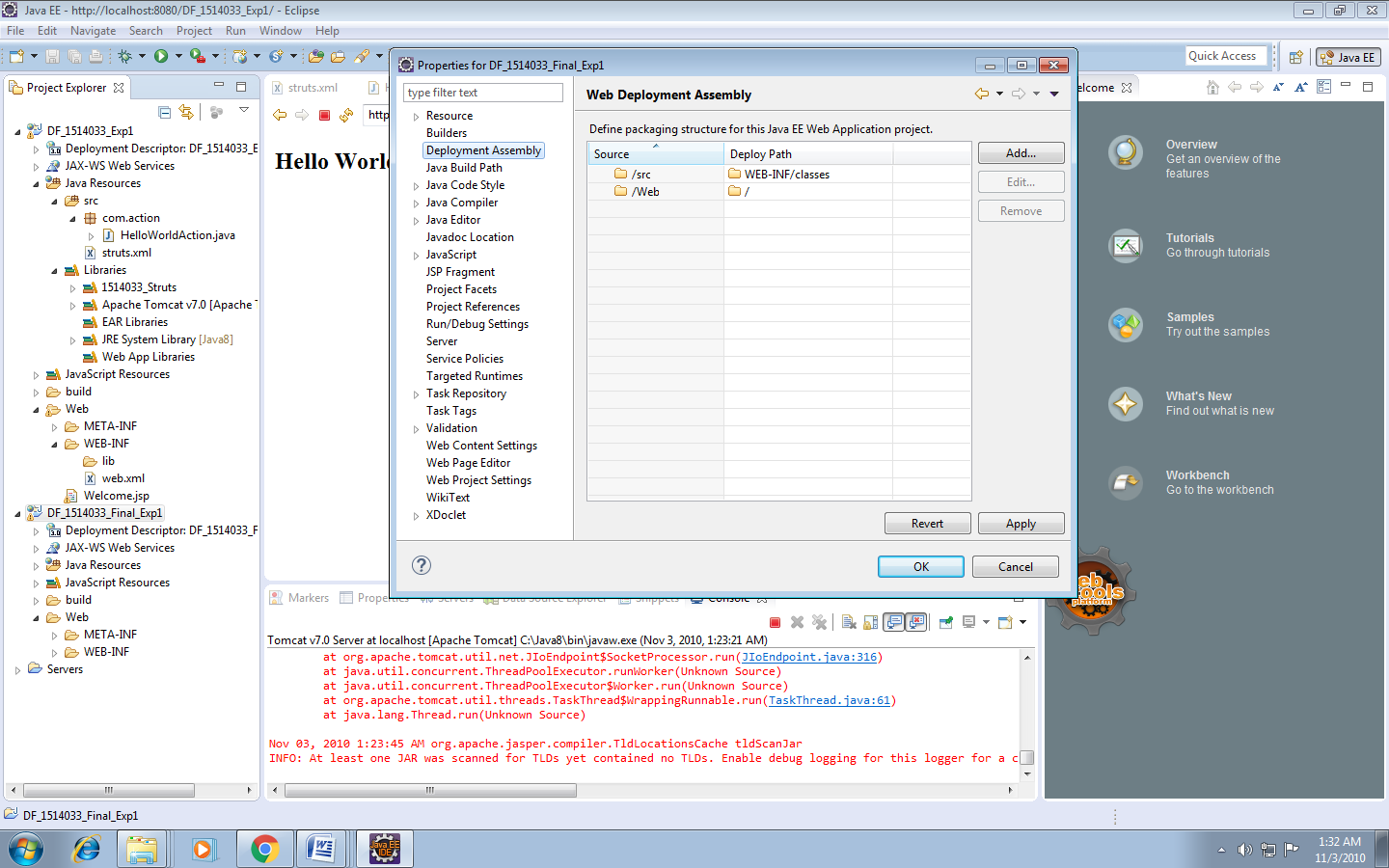
Click on OK



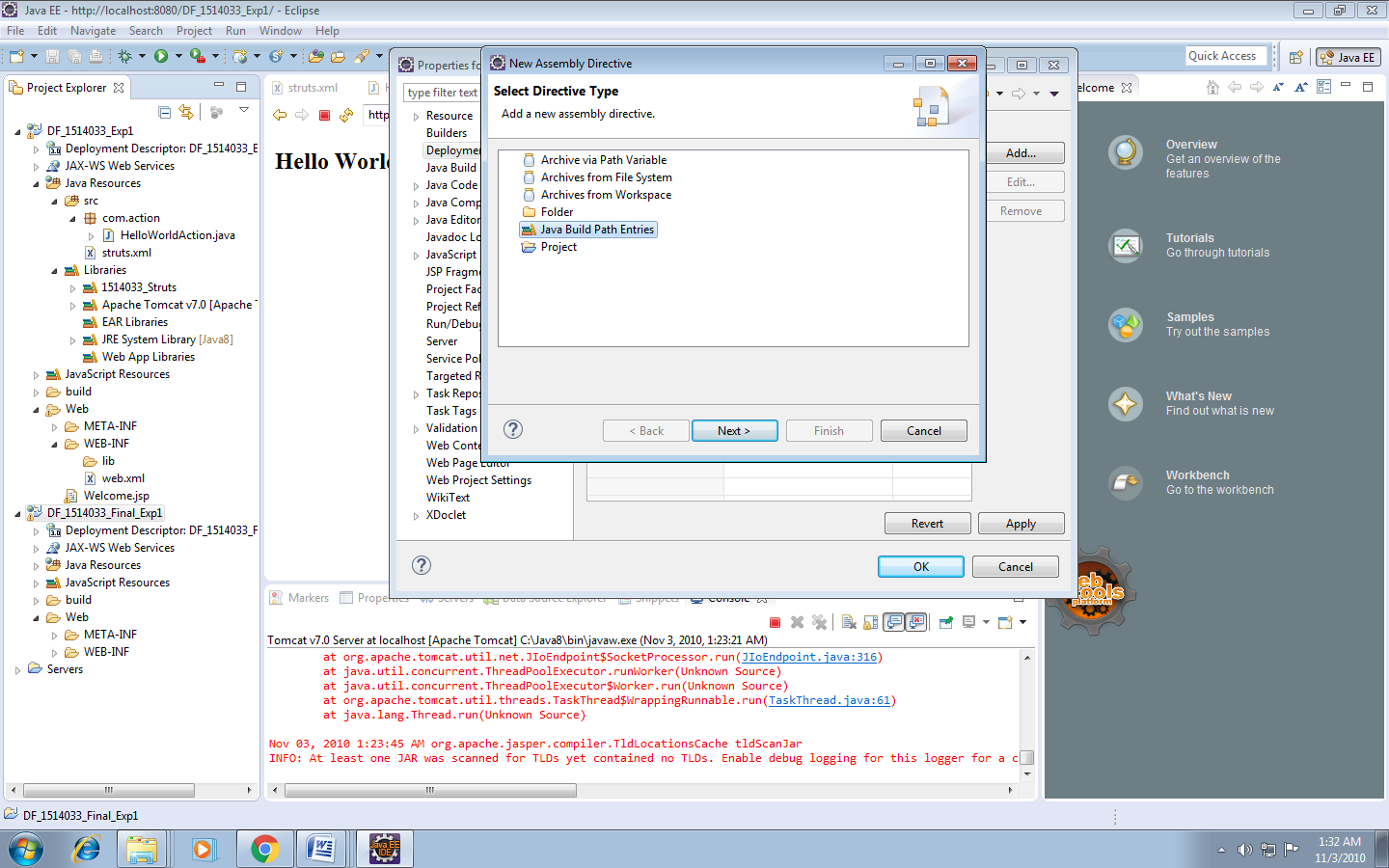
Mark your library and click on finish



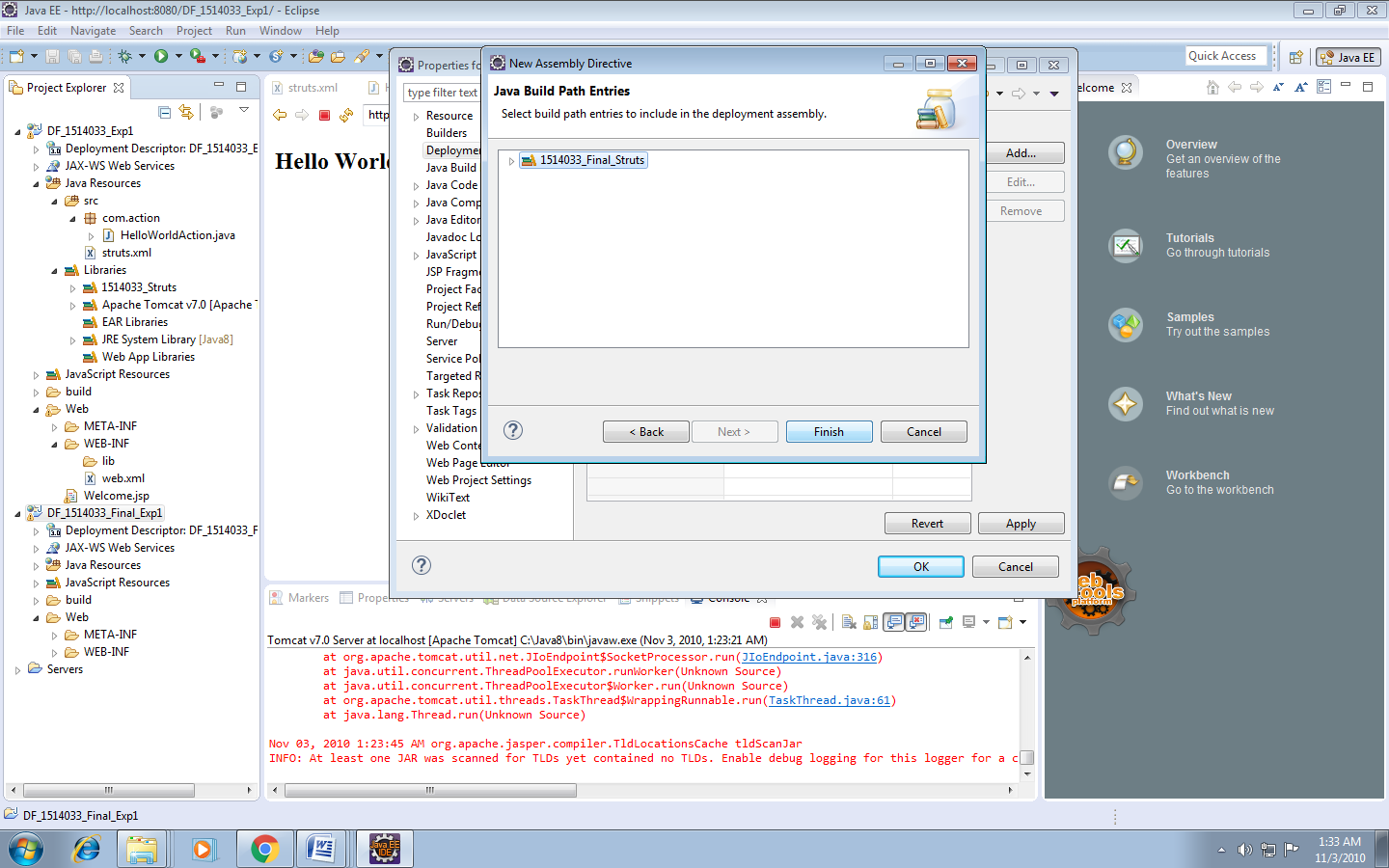
Right click on project name. Go to properties. Go to Deployment assembly. Click on Add



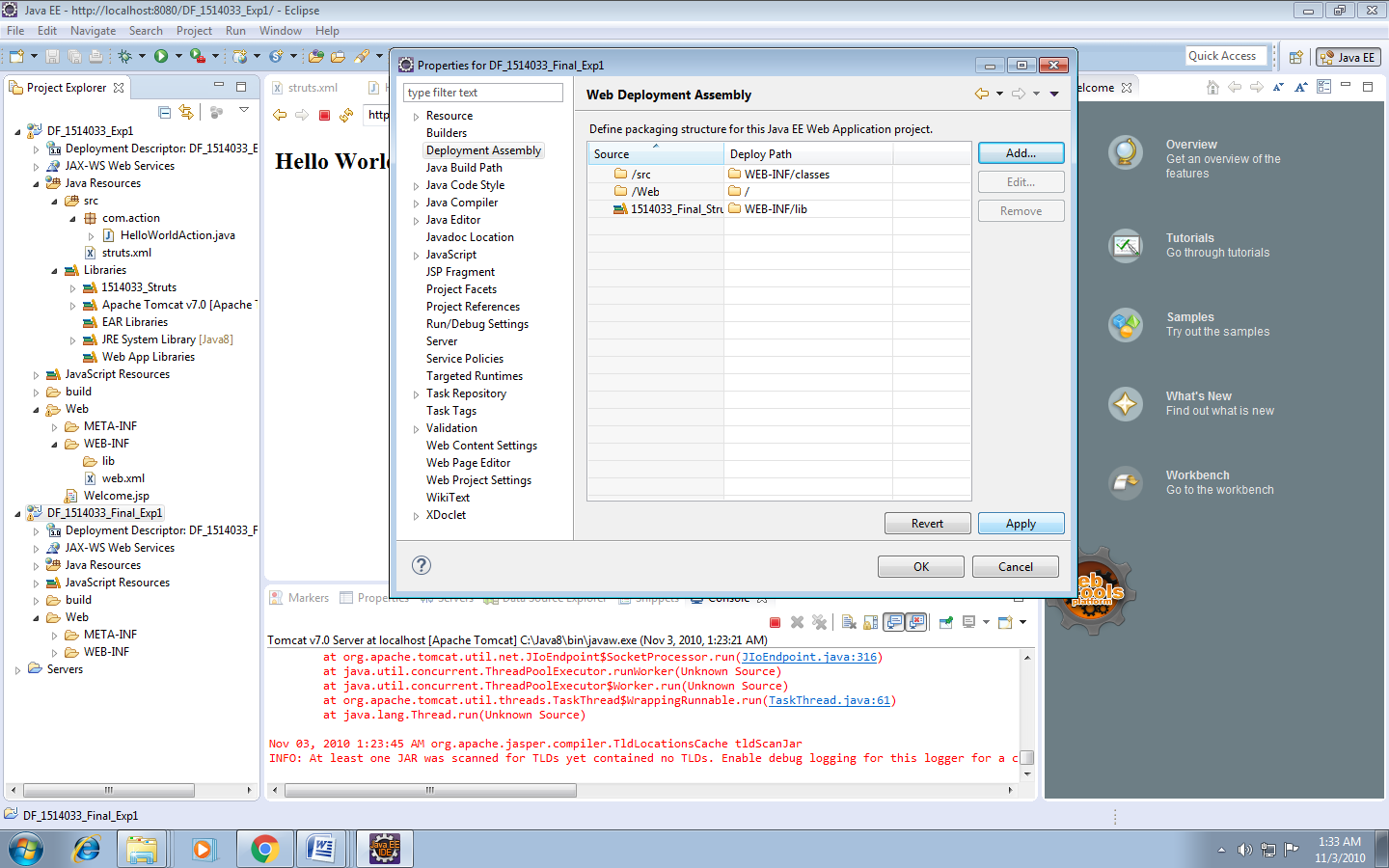
Click on Java Build Path Libraries.



Select library and click on finish



Click on apply and then click on ok.



Web.xml

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

<web-app xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns=*"http://java.sun.com/xml/ns/javaee"* xsi:schemaLocation=*"http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd"* id=*"WebApp\_ID"* version=*"3.0"*>

<display-name>DF\_1514033\_Final\_Exp1</display-name>

<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.ng.filter.StrutsPrepareAndExecuteFilter

</filter-class>

</filter>

<filter-mapping>

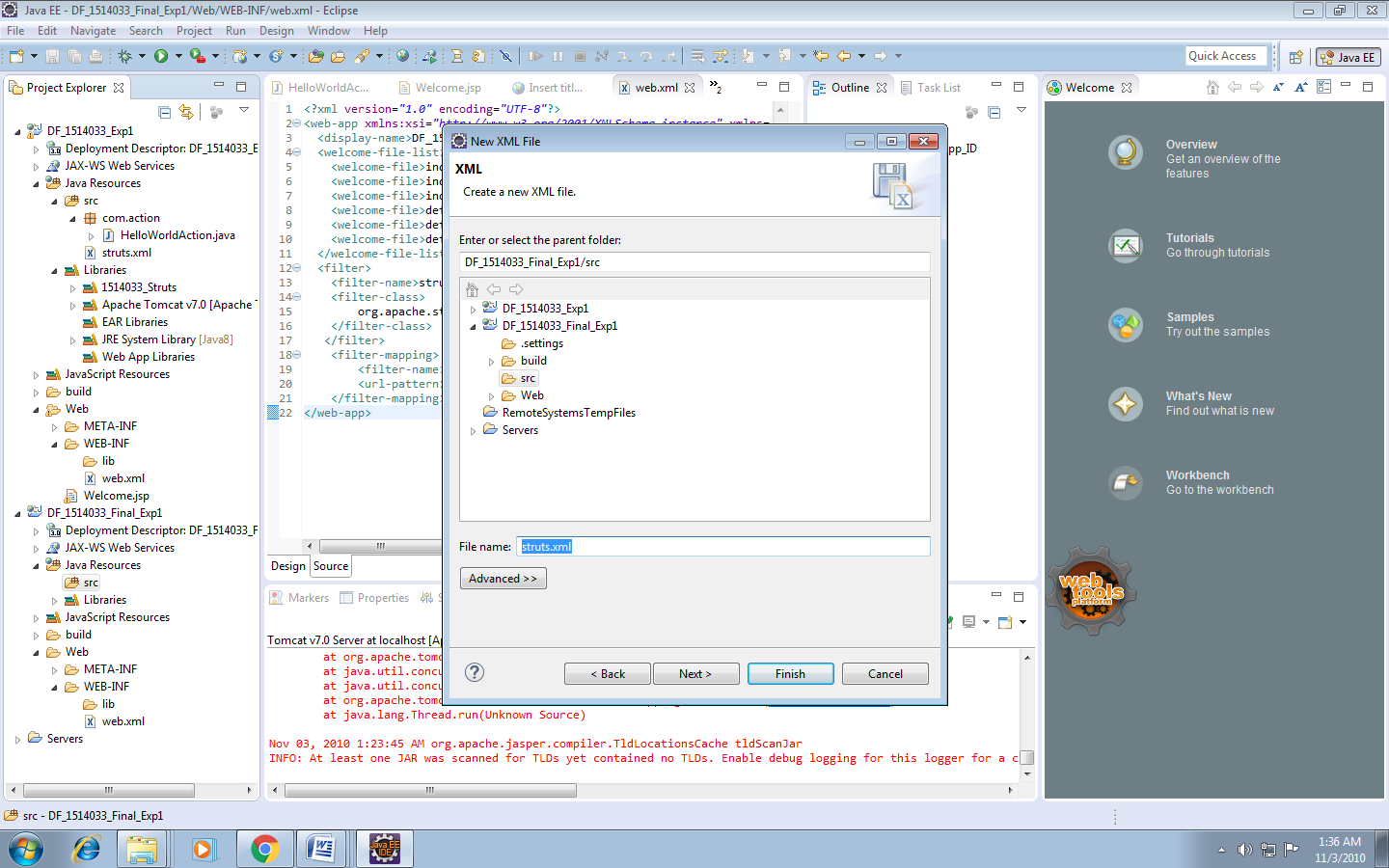
<filter-name>struts2</filter-name>

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

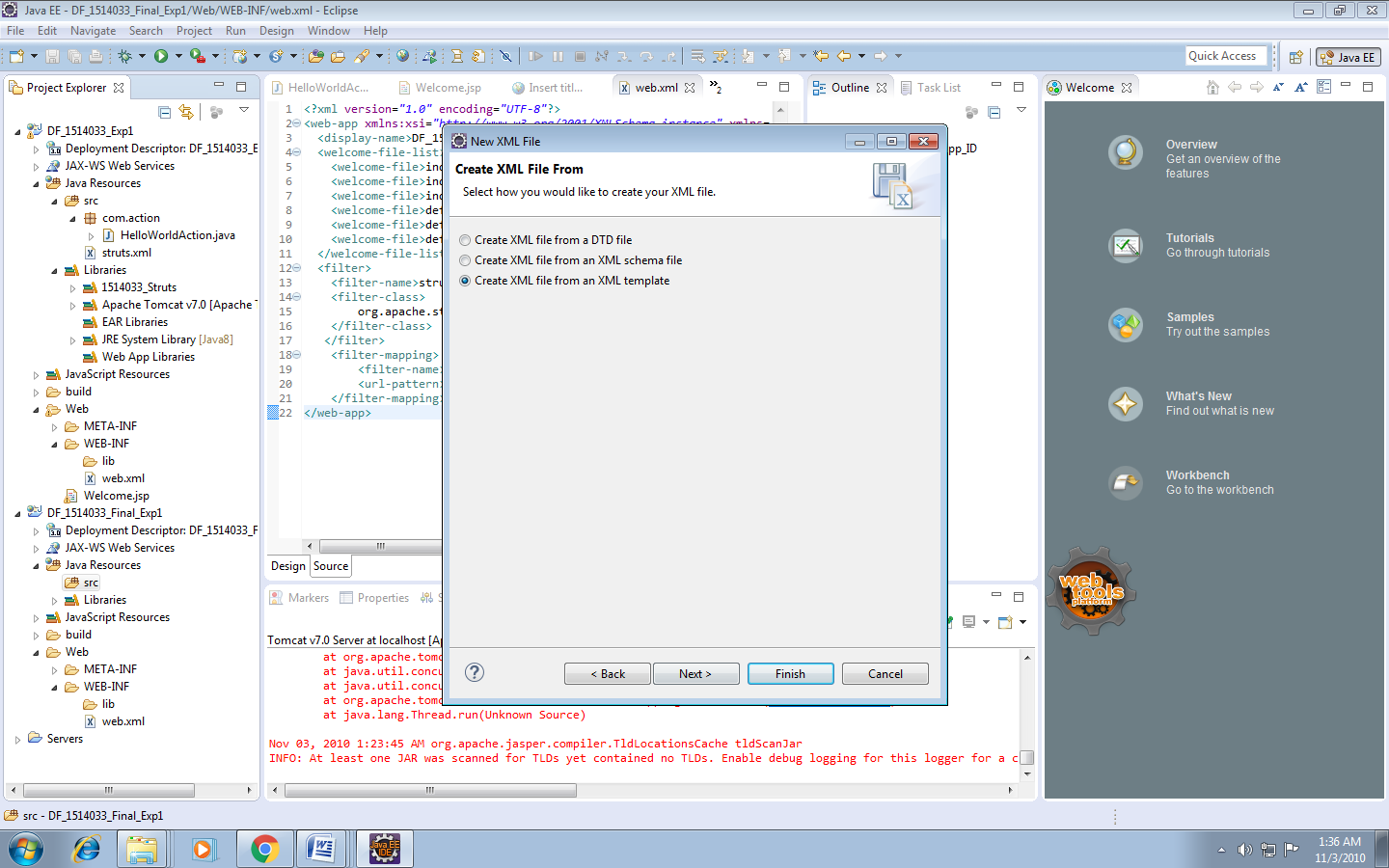
</filter-mapping>

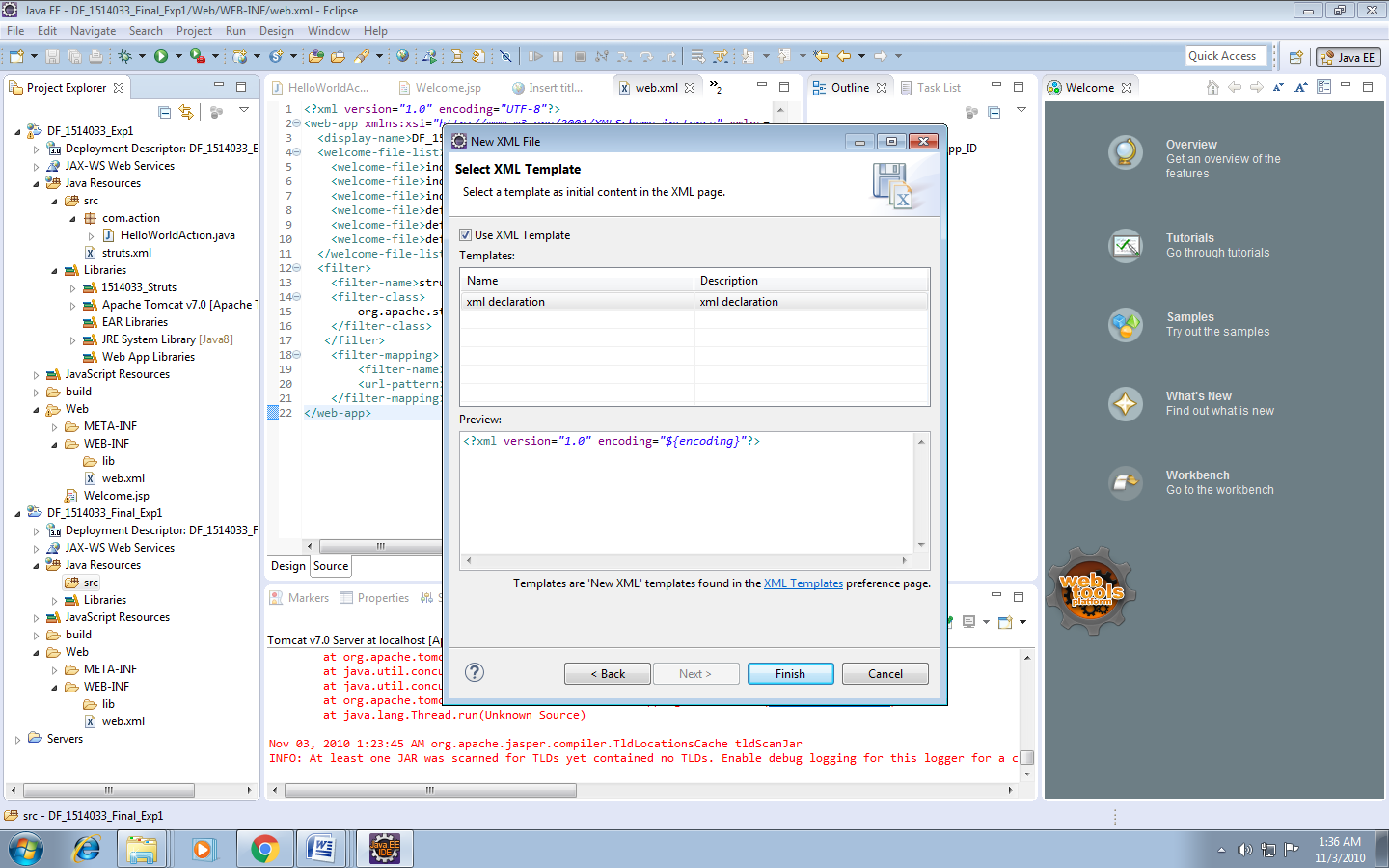
</web-app>

Create new xml file in java->src



Click on finish.





struts.xml

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

<!DOCTYPE struts PUBLIC "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"

"http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>

<package name=*"default"* extends=*"struts-default"*>

<action name=*"getHello"* class=*"com.action.HelloWorldAction"*>

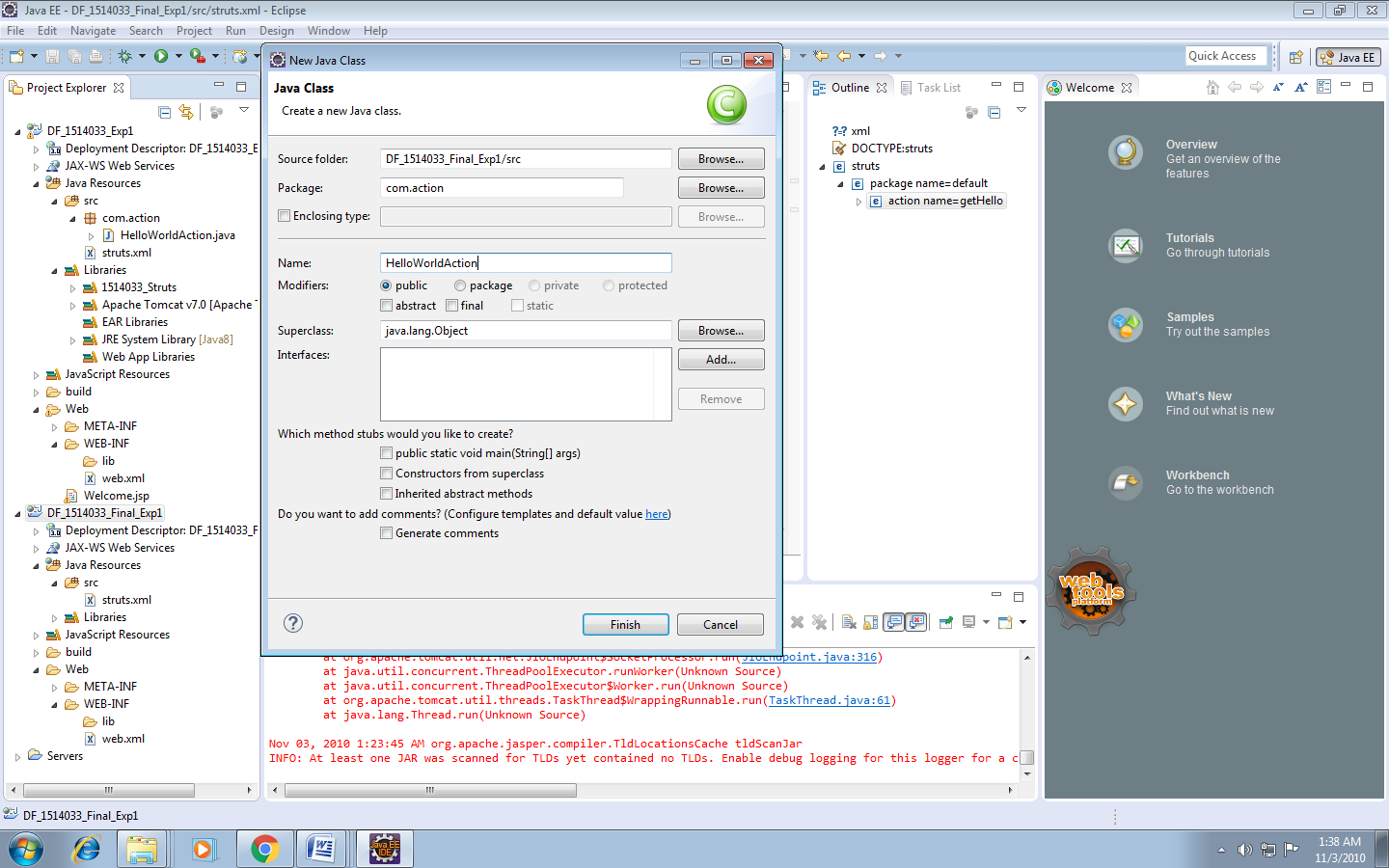
<result name=*"SUCCESS"*>/Welcome.jsp</result>

</action>

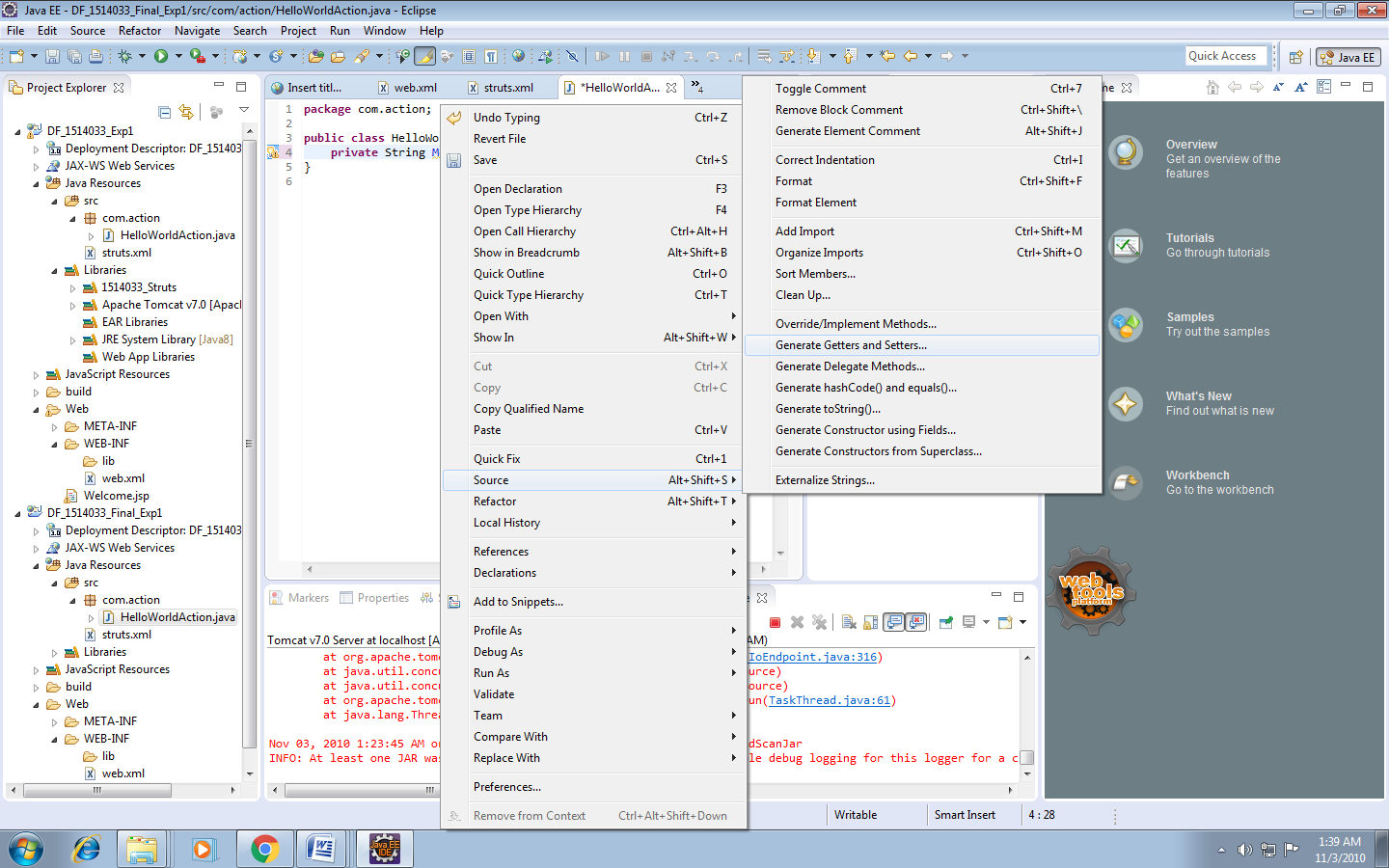
</package>

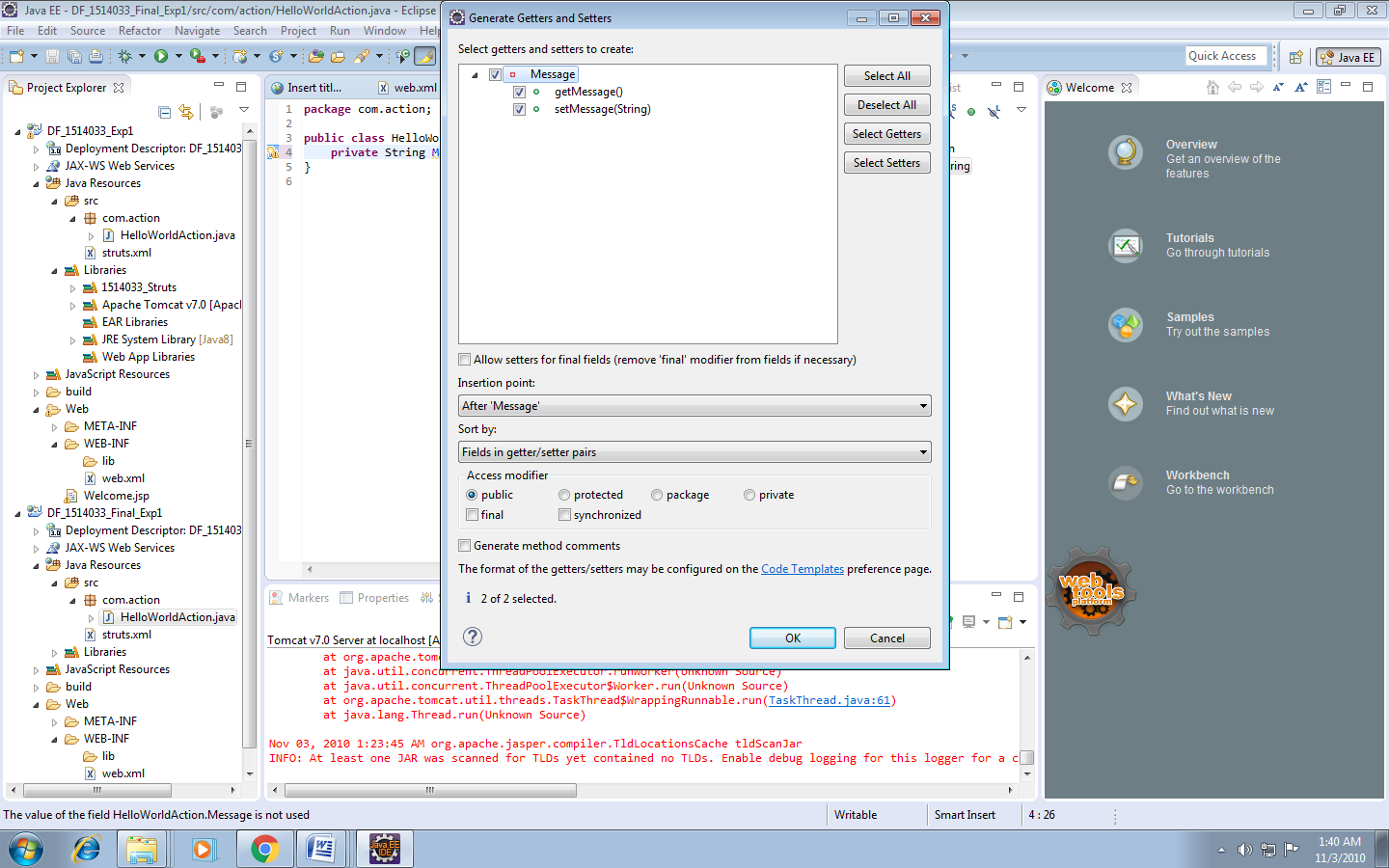
</struts>

Create class file in java->src with package as desired.



For getter setter method for message variable. Right click on message. Go to source. Go to generate getter and setter method.





com.action.HelloWorldAction.java

**package** com.action;

**public** **class** HelloWorldAction {

**private** String Message;

**public** String execute()

{

setMessage("Welcome to DF Lab");

**return** "SUCCESS";

}

**public** String getMessage() {

**return** Message;

}

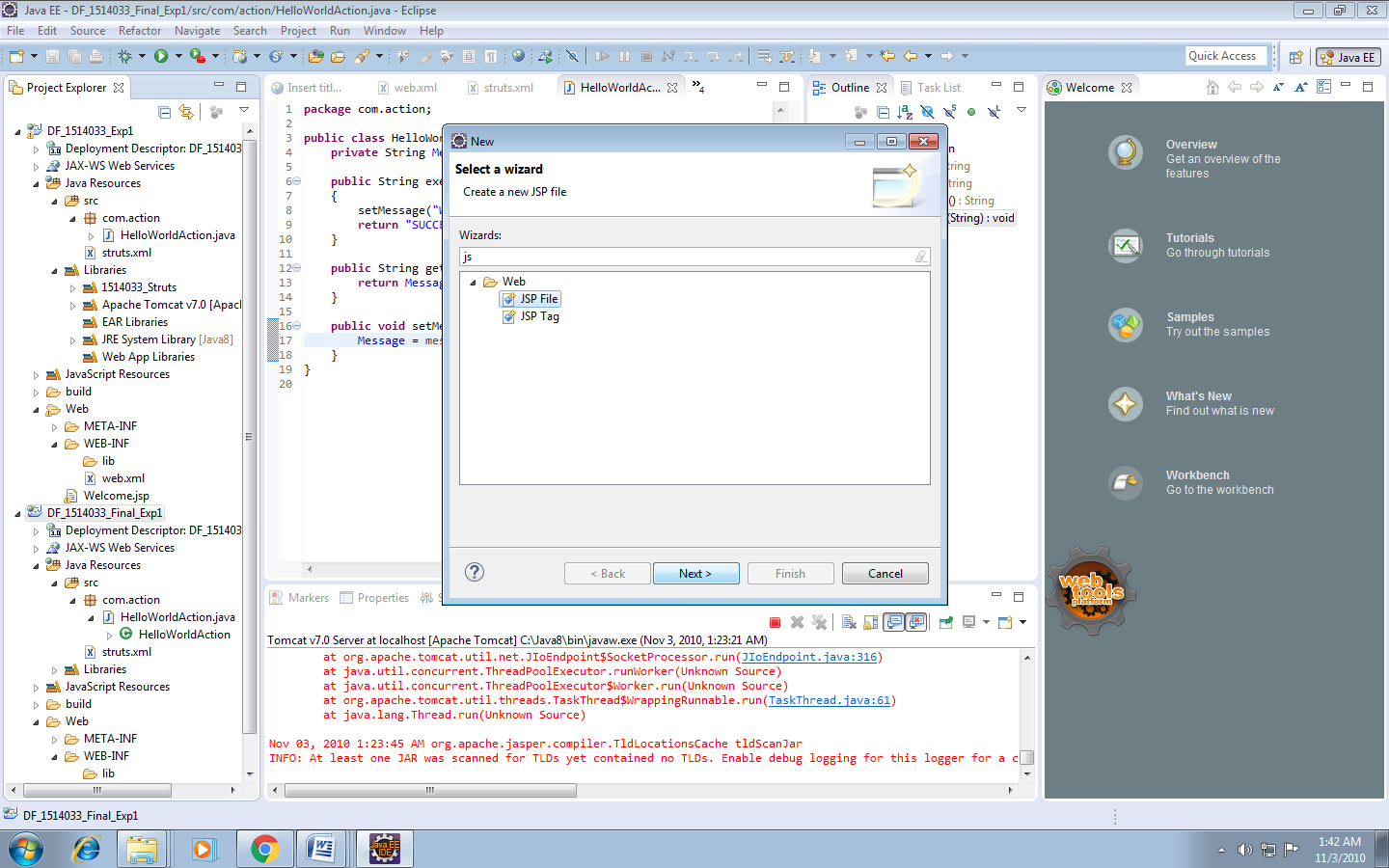
**public** **void** setMessage(String message) {

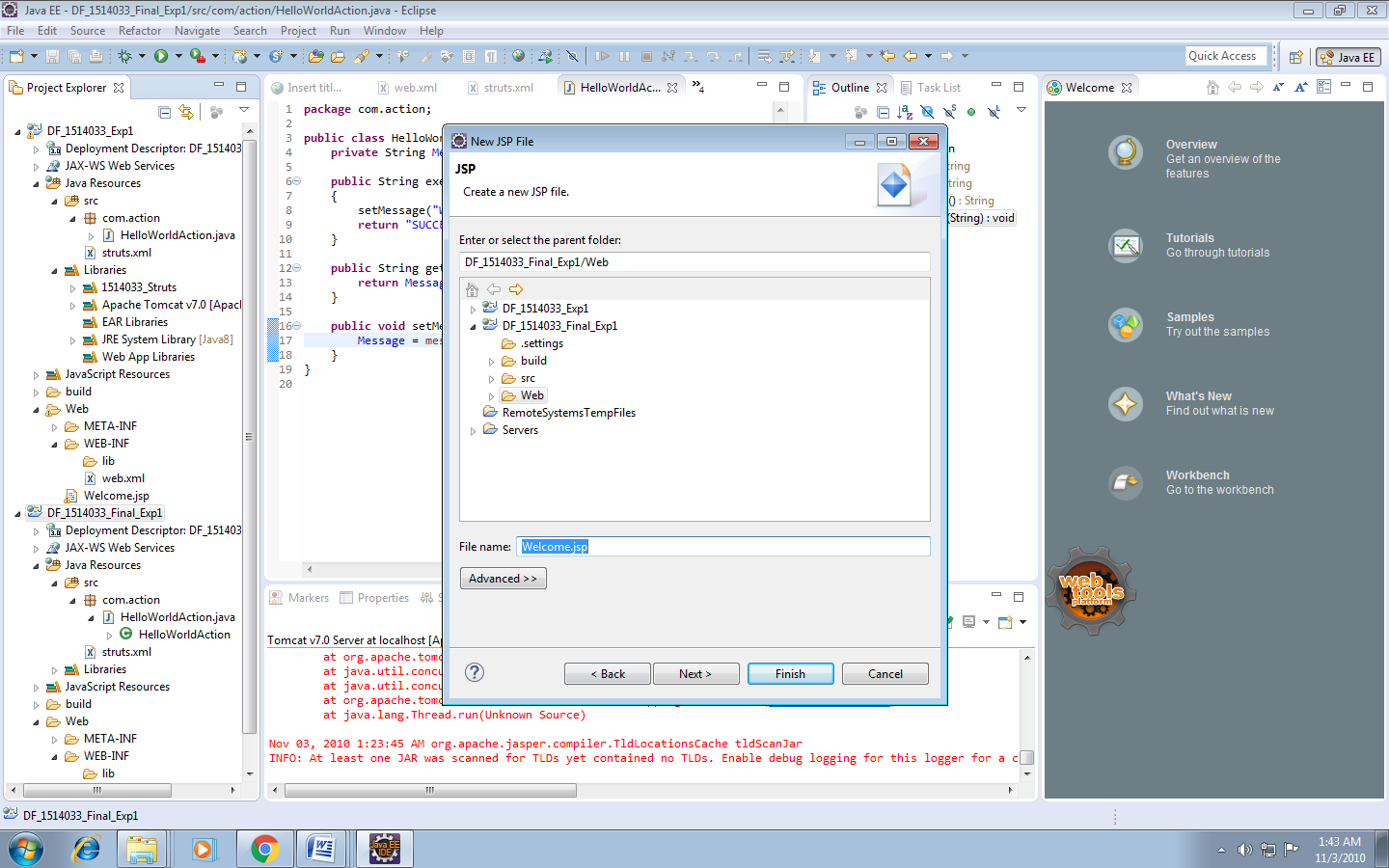
Message = message;

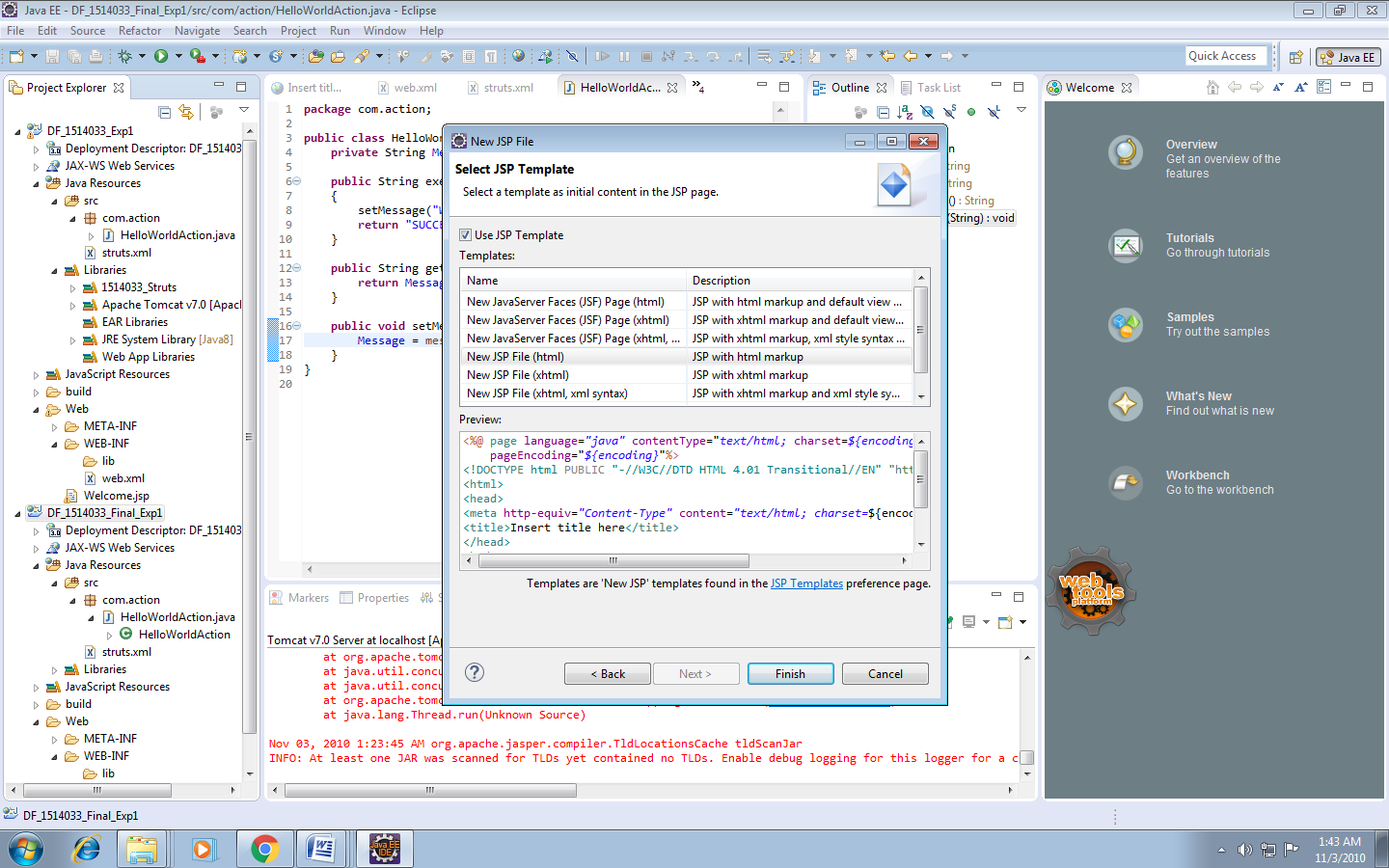
}

}

Create JSP File







Welcome.jsp

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<%@ taglib prefix=*"s"* uri=*"/struts-tags"* %>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Insert title here</title>

</head>

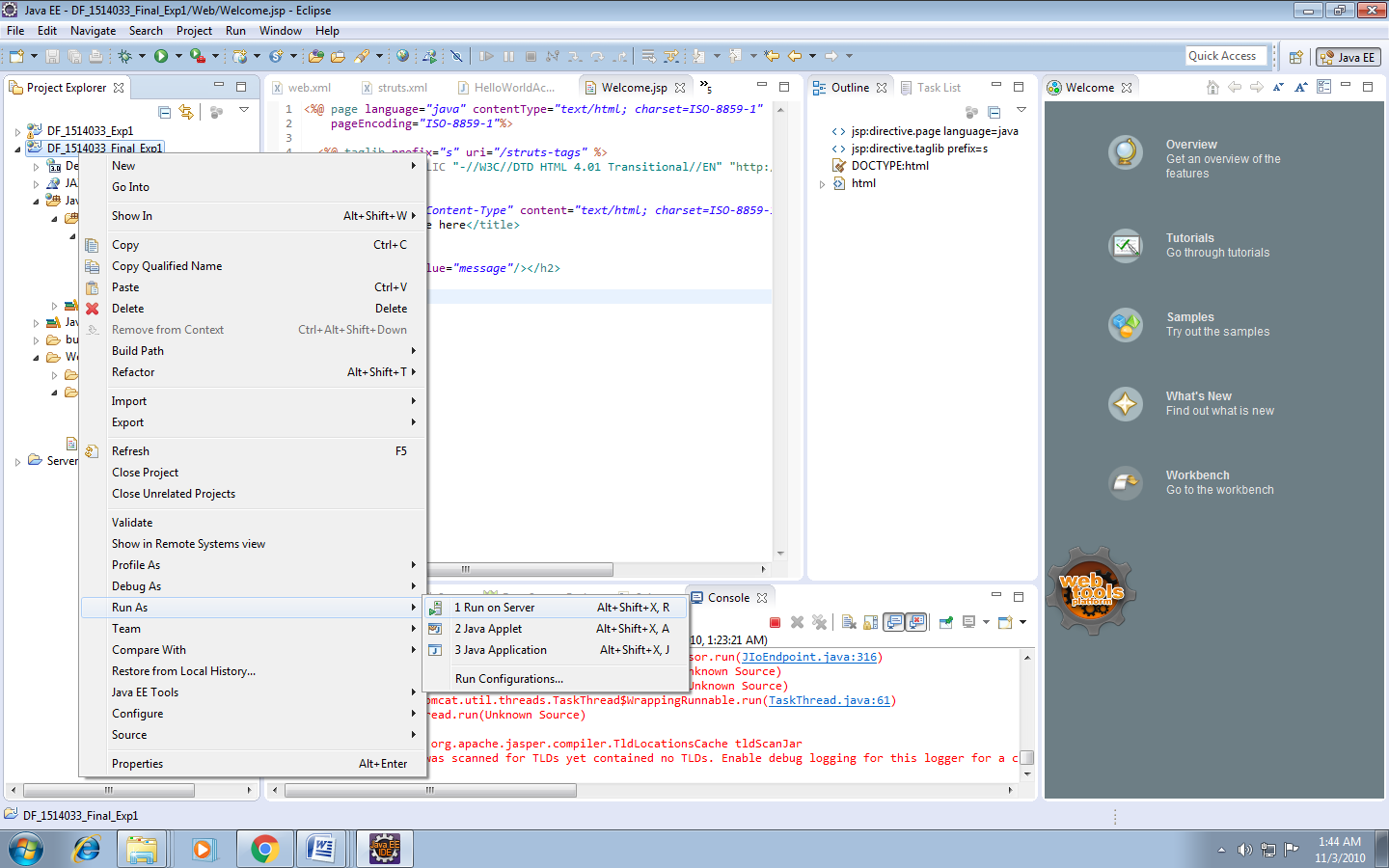
<body>

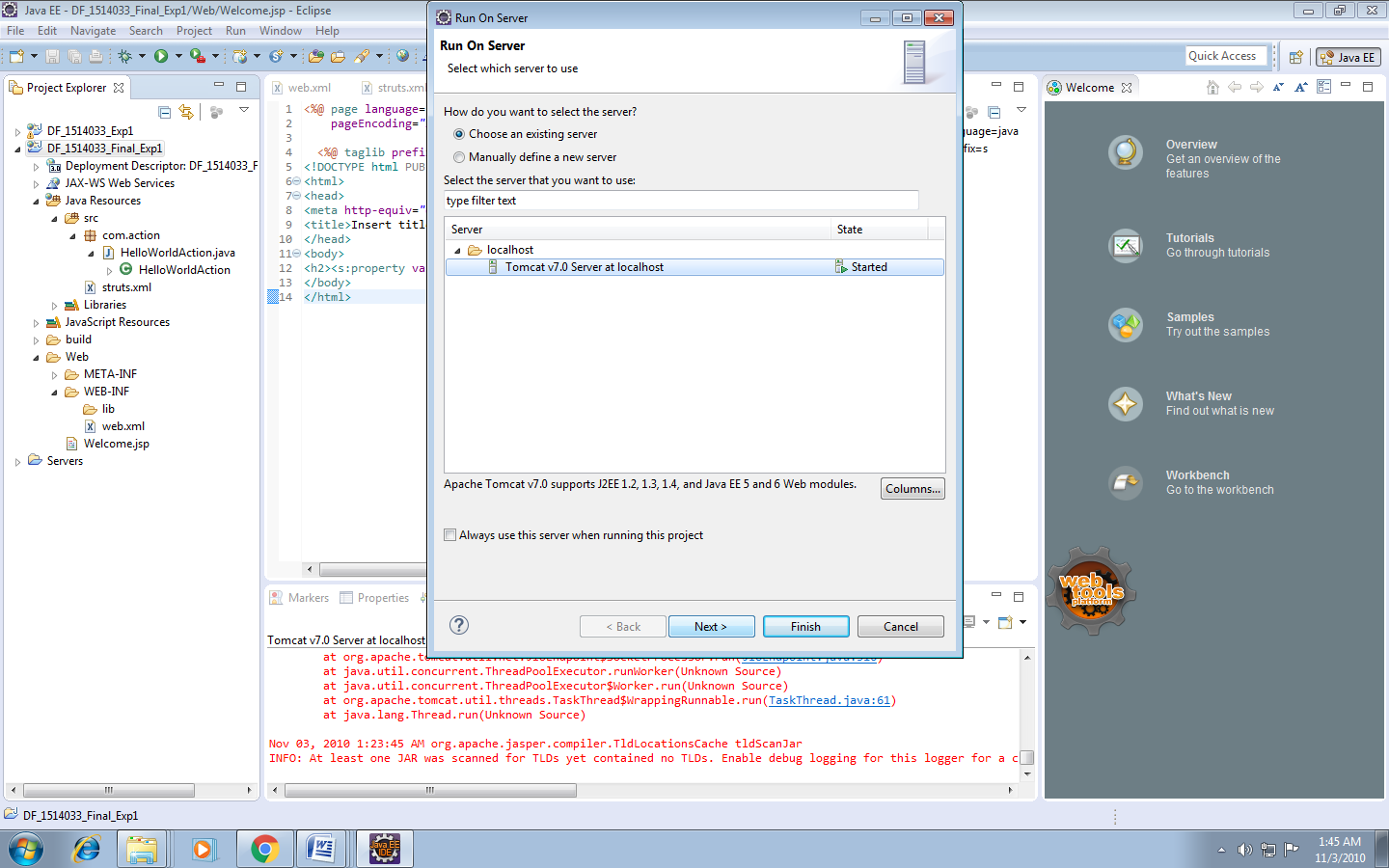
<h2><s:property value=*"message"*/></h2>

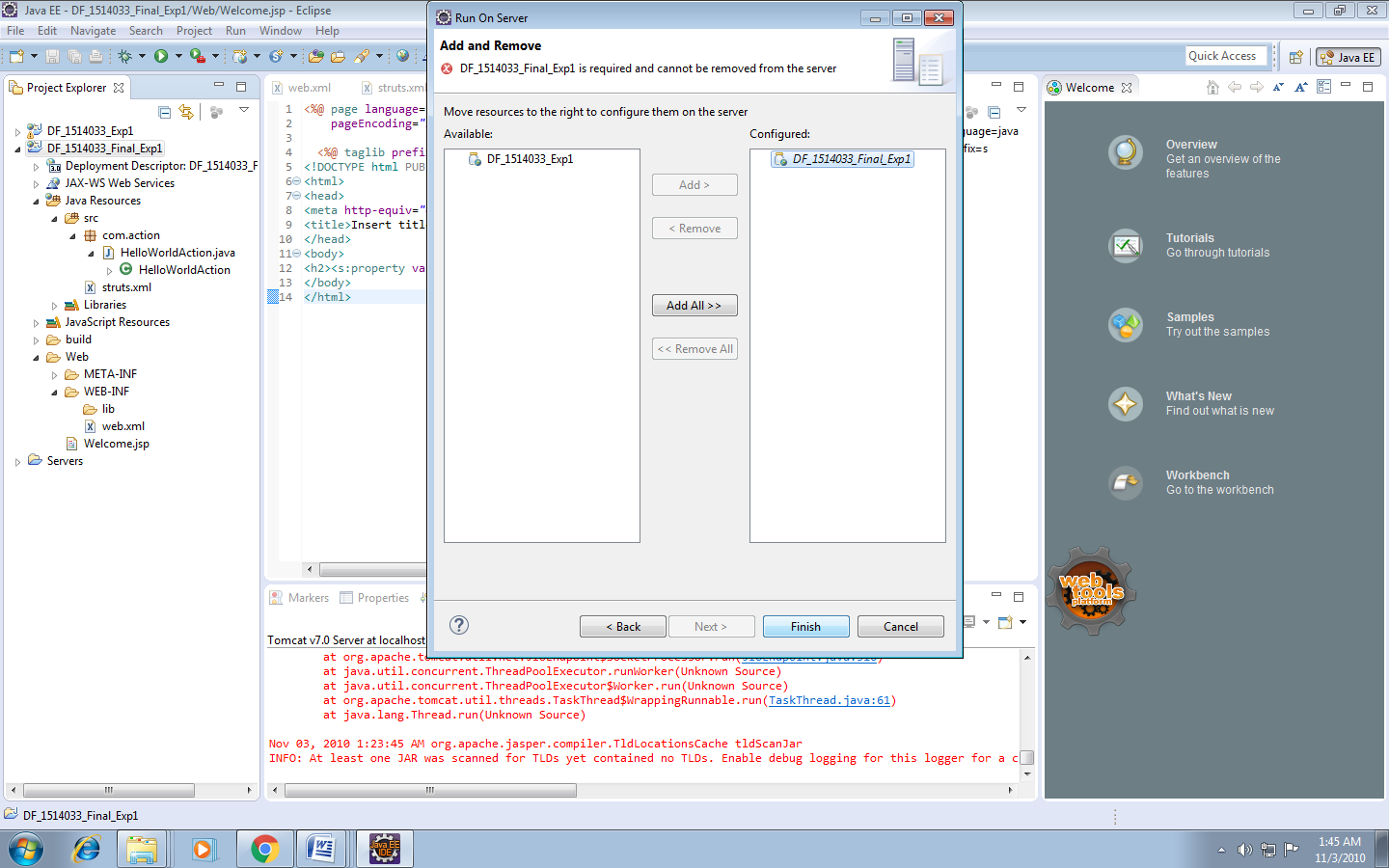
</body>

</html>

Run Project

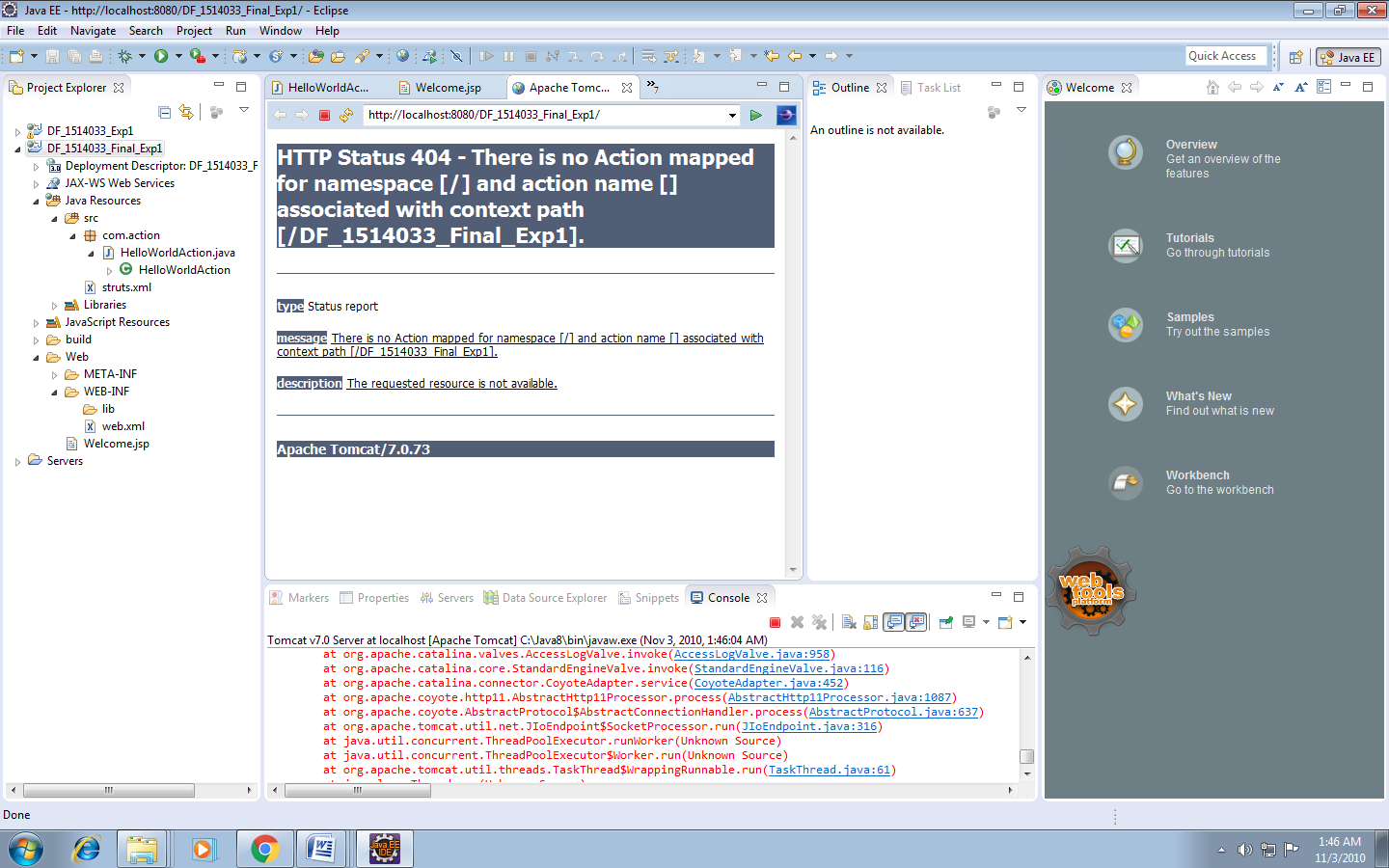




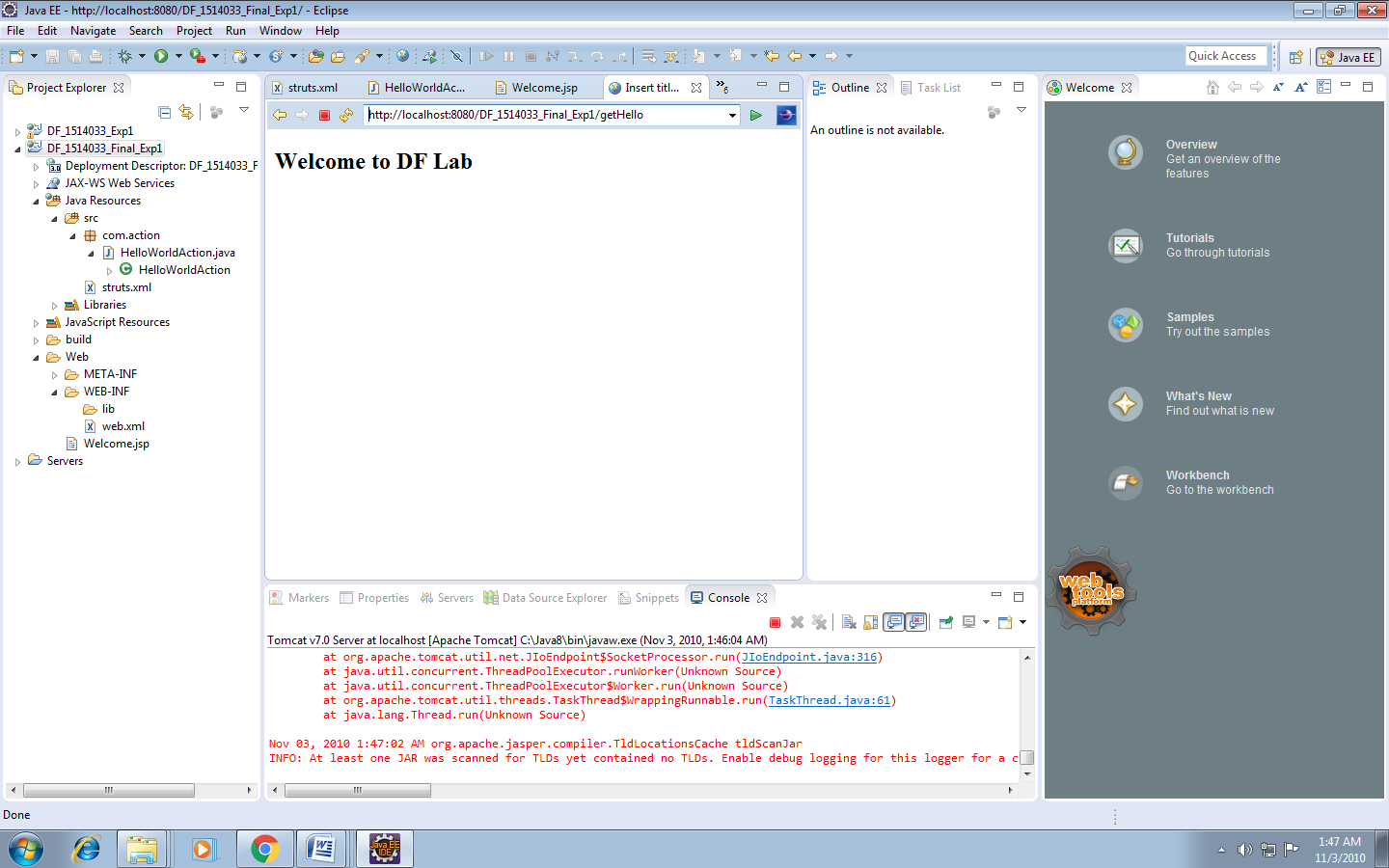


OUTPUT:

Namespace error



Add name getHello



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Questions:**

1. Explain the flow of execution for struts program.

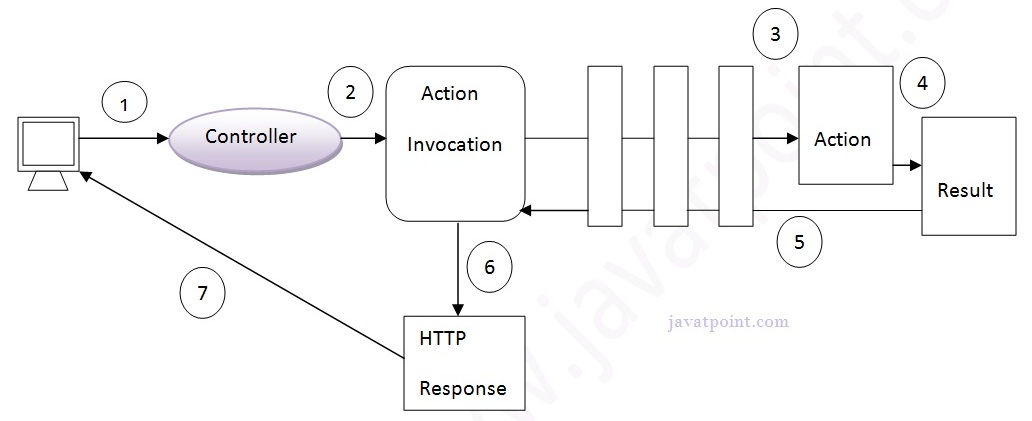
The **architecture and flow of struts 2 application**, is combined with many components such as Controller, ActionProxy, ActionMapper, Configuration Manager, ActionInvocation, Inerceptor, Action, Result etc.

Here, we are going to understand the struts flow by 2 ways:

1. struts 2 basic flow
2. struts 2 standard architecture and flow provided by apache struts

#### Struts 2 basic flow

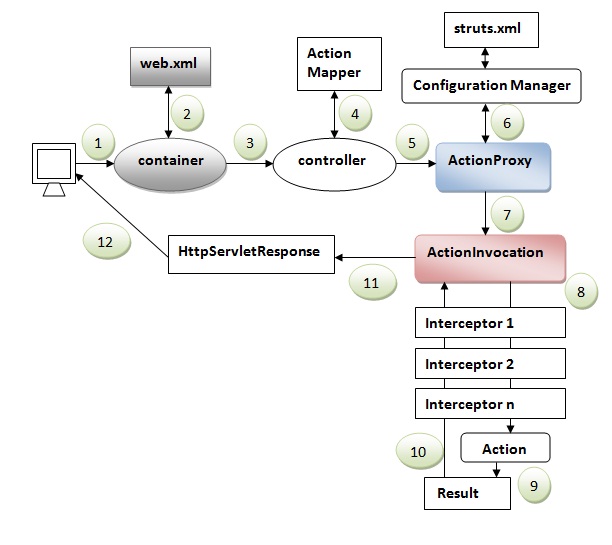
Let's try to understand the basic flow of struts 2 application by this simple figure:



1. User sends a request for the action
2. Controller invokes the ActionInvocation
3. ActionInvocation invokes each interceptors and action
4. A result is generated
5. The result is sent back to the ActionInvocation
6. A HttpServletResponse is generated
7. Response is sent to the user

#### Struts 2 standard flow (Struts 2 architecture)

Let's try to understand the standard architecture of struts 2 application by this simple figure:



1. User sends a request for the action
2. Container maps the request in the web.xml file and gets the class name of controller.
3. Container invokes the controller (StrutsPrepareAndExecuteFilter or FilterDispatcher). Since struts2.1, it is StrutsPrepareAndExecuteFilter. Before 2.1 it was FilterDispatcher.
4. Controller gets the information for the action from the ActionMapper
5. Controller invokes the ActionProxy
6. ActionProxy gets the information of action and interceptor stack from the configuration manager which gets the information from the struts.xml file.
7. ActionProxy forwards the request to the ActionInvocation
8. ActionInvocation invokes each interceptors and action
9. A result is generated
10. The result is sent back to the ActionInvocation
11. A HttpServletResponse is generated
12. Response is sent to the user

Outcomes: Understand the need, features, architecture and applications of frameworks

**Conclusion: (Conclusion to be based on the objectives and outcomesachieved)**

Hence we successfully demonstrated a simple program using the Struts2 architecture.

**Grade: AA / AB / BB / BC / CC / CD/DD**

**Signature of faculty in-charge withdate**

**References:**

1.By James Holmes ; ”Struts: The Complete Reference”; 2nd Edition ; McGraw-Hill

Publication

2.ByChuckCavaness; “Programming Jakarta Struts” ; 2nd Edition ; O'Reilly Media

Publication; 2004