Assignment No. 07

AIM:

Build a dynamic web application using PHP and MySQL.

a. Create database tables in MySQL and create connection with PHP.
b. Create the add, update, delete and retrieve functions in the PHP web app interacting with MySQL database

LEARNING OBJECTIVES:

1. To understand the principles and methodologies of PHP web based applications development process
2. Having the knowledge of SQL query to create the database

SOFTWARE REQUIREMENTS:

1 Operating System: Windows 7/8/10/Ubuntu
2 Browser: Firefox/Google Chrome/ Microsoft Edge etc.
3 Software/Editor : Sublime Editor/Notepad/Notepad++
4 Any Operating System
5 XAMPP Server

HARDWARE REQUIREMENTS:

1 Processor: Minimum 1 GHz.
2 Ethernet connection (LAN) OR a wireless adapter (Wi-Fi)
3 Hard Drive: Minimum 32 GB.
4 Memory (RAM): Minimum 1 GB

THEORY:

PHP:

The PHP Hypertext Preprocessor (PHP) began as a little open source venture that advanced as an ever increasing number of individuals discovered how valuable it was. Rasmus Lerdorf released the principal form of PHP route in 1994. PHP is a recursive acronym for "PHP: Hypertext Pre-processor".

PHP is a server side scripting dialect that is installed in HTML. It is utilized to oversee dynamic substance, databases, session following, even form whole internet business locales. It is incorporated with various prevalent databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.

PHP is pleasingly zippy in its execution, particularly when gathered as an Apache module on the Unix side. The MySQL server, once began, executes even extremely complex questions with colossal outcome sets in record-setting time. PHP bolsters a substantial number of real conventions, for example, POP3, IMAP, and LDAP. PHP4 included help for Java and conveyed question designs (COM and CORBA), making n- level improvement a plausibility out of the blue. PHP is excusing: PHP dialect tries to be as pardoning as would be prudent. PHP Syntax is C-Like. PHP performs framework capacities, i.e. from documents on a framework it can make, open, read, compose, and close them. PHP can deal with frames, i.e. accumulate information from records, spare information to a document; through email you can send information, return information to the client.

You include, erase, and adjust components inside your database through PHP. Access treats factors and set treats. Utilizing PHP, you can confine clients to get to a few pages of your site. It can encode information.

Basic PHP Syntax:

A PHP script can be placed anywhere in the document. A PHP script starts with <?php and ends with ?>:

```
<?php
        //PHP_code_goes_here
?>
```

The default file extension for PHP files is ".php".
A PHP file normally contains HTML tags, and some PHP scripting code.

Below, we have an example of a simple PHP file, with a PHP script that uses a built-in PHP function "echo" to output the text "Hello World!" on a web page:

Example:

```
<!DOCTYPE html>
<html>
<body>
<h1>My first PHP page</h1>
<?php
echo "Hello World!";
?>
</body>
</html>
```

Note: PHP statements end with a semicolon (;).

Web Server:

PHP is server side scripting language so it requires Web server to execute (eg. Apache Tomcat) Database − PHP will work with for all intents and purposes all database programming, including Oracle and Sybase yet most regularly utilized is uninhibitedly accessible MySQL database.

PHP Parser:

Keeping in mind the end goal to process PHP content directions a parser must be introduced to create HTML yield that can be sent to the Web Browser. This instructional exercise will manage you how to introduce PHP parser on your PC.

MySQL:

MySQL is the most famous Open Source Relational SQL Database Management System. MySQL is outstanding amongst other RDBMS being utilized for creating different online programming applications. MySQL is created, advertised and upheld by MySQL AB, which is a Swedish organization. This instructional exercise will give you a fast begin to MySQL and make you OK with MySQL programming.

TECHNOLOGY:

1 Technology is to be used is PHP (PHP Hypertext Preprocessor) and tool XAMPP server is to be used to execute PHP web application.

2 XAMPP servers embeds the PHP, MySQL and phpmyadmin, these three tools must be required to run php web application.

DESIGN/EXECUTION STEPS:

For the design purpose html and CSS is to be used. For this design part contains the GUI of web applications, how its looks like? When users going to use the web application.

Steps to install XAMPP and configure the PHP, MYSQL server:

1 Download the XAMPP using following link (download latest version as per your Operating system Windows/ Linux).

https://www.apachefriends.org/download.html

2 Install XAMPP by running .exe file

3 Go to start->Xampp-> Xampp control panel. Start Apache and Mysql.

4 Goto Web browser(eg. Firefox) and write localhost and see Xampp has been started or not..

5 By Clicking on PhpMyAdmin you can create database, table and insert values in MySQL.

6 Open a note pad write a PHP code and save this file in Xampp->htdoc->create a folder here(eg.PHP1) and save file in this folder with name index.php

7 To run the code goto Firefox or any web browser and type in address bar- localhost/PHP1 your code will get run.

8 If file name is other that index.php then to run type in address bar – localhost/PHP1/hello.php (file name)

CONCLUSION:

In this assignment, we have studied how to design and develop small web application using PHP script, XAMPP server with apache server and MySQL as backend.

Assignment No. 08

AIM:

Design a login page with entries for name, mobile number email id and login button. Use struts and perform following validations

a. Validation for correct names
b. Validation for mobile numbers
c. Validation for email id
d. Validation if no entered any value
e. Re-display for wrongly entered values with message
f. Congratulations and welcome page upon successful entries

LEARNING OBJECTIVES:

To understand Basics of Struts

THEORY:

INTRODUCTION TO STRUTS:

Apache Struts is an elegant, extensible framework for creating enterprise-ready Java web applications. This framework is designed to streamline the full development cycle from building, to deploying and maintaining applications over time. Apache Struts was originally known as Web Work 2.

BASIC MVC ARCHITECTURE:

Model **V**iew **C**ontroller or **MVC** as it is popularly called, is a software design pattern for developing web applications. A Model View Controller pattern is made up of the following three parts –

- **Model** − The lowest level of the pattern which is responsible for maintaining data.
- **View** − This is responsible for displaying all or a portion of the data to the user.
- **Controller** − Software Code that controls the interactions between the Model and View. MVC is popular as it isolates the application logic from the user interface layer and supports separation of concerns. Here the Controller receives all requests for the application and then works with the Model to prepare any data needed by the View. The View then uses the data prepared by the Controller to generate a final presentable response. The MVC abstraction can be graphically represented as follows.

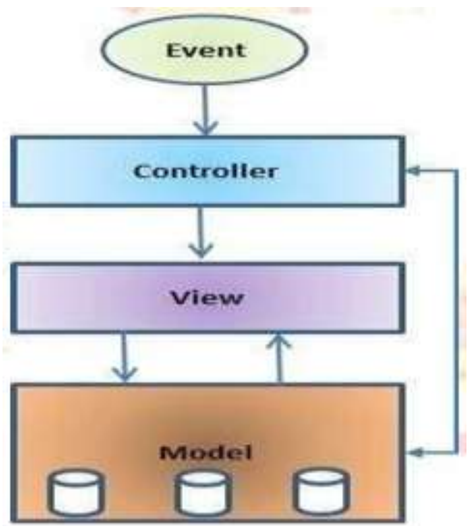


Fig. 01. MVC Architecture

THE MODEL:

The model is responsible for managing the data of the application. It responds to the request from the view and it also responds to instructions from the controller to update itself.

THE VIEW:

It means presentation of data in a particular format, triggered by a controller's decision to present the data. They are script-based templating systems like JSP, ASP, PHP and very easy to integrate with AJAX technology.

THE CONTROLLER:

The controller is responsible for responding to the user input and performs interactions on the data model objects. The controller receives the input; it validates the input and then performs the business operation that modifies the state of the data model

OVERVIEW:

Struts2 is a popular and mature web application framework based on the MVC design pattern. Struts2 is not just a new version of Struts 1, but it is a complete rewrite of the Struts architecture. The Webwork framework initially started with Struts framework as the basis and its goal was to offer an enhanced and improved framework built on Struts to make web development easier for the developers.

After a while, the Webwork framework and the Struts community joined hands to create the famous Struts2 framework.

STRUTS FRAMEWORK FEATURES:

Here are some of the great features that may force you to consider Struts2 –

- **POJO Forms and POJO Actions** − Struts2 has done away with the Action Forms that were an integral part of the Struts framework. With Struts2, you can use any POJO to receive the form input. Similarly, you can now see any POJO as an Action class.
- **Tag Support** − Struts2 has improved the form tags and the new tags which allow the developers to write less code.
- **AJAX Support** − Struts2 has recognized the takeover by Web2.0 technologies, and has integrated AJAX support into the product by creating AJAX tags, this function is very similar to the standard Struts2 tags.
- **Easy Integration** − Integration with other frameworks like Spring, Tiles and SiteMesh is now easier with a variety of integration available with Struts2.
- **Template Support** − Support for generating views using templates.
- **Plugin Support** − The core Struts2 behavior can be enhanced and augmented by the use of plugins. A number of plugins are available for Struts2.
- **Profiling** − Struts2 offers integrated profiling to debug and profile the application. In addition to this, Struts also offers integrated debugging with the help of built in debugging tools.
- **Easy to Modify Tags** − Tag markups in Struts2 can be tweaked using Free marker templates. This does not require JSP or java knowledge. Basic HTML, XML and CSS knowledge is enough to modify the tags.
- **Promote fewer configurations** − Struts2 promotes fewer configurations with the help of using default values for various settings. You don't have to configure something unless it deviates from the default settings set by Struts2.

- **View Technologies** − Struts2 has a great support for multiple view options (JSP, Free marker, Velocity and XSLT)

Listed above are the Top 10 features of **Struts** which makes it as an Enterprise ready framework.

STRUTS DISADVANTAGES:

Though Struts comes with a list of great features, there are some limitations of the current version - Struts which needs further improvement. Listed are some of the main points –

- **Bigger Learning Curve** − To use MVC with Struts, you have to be comfortable with the standard JSP, Servlet APIs and a large & elaborate framework.
- **Poor Documentation** − Compared to the standard servlet and JSP APIs, Struts has fewer online resources, and many first-time users find the online Apache documentation confusing and poorly organized.
- **Less Transparent** − With Struts applications, there is a lot more going on behind the scenes than with normal Java-based Web applications which makes it difficult to understand the framework.

ENVIRONMENT SETUP:

STEP 1 - SETUP JAVA DEVELOPMENT KIT (JDK)

You can download the latest version of SDK from Oracle's Java site − Java SE Downloads. You will find instructions for installing JDK in downloaded files, follow the given instructions to install and configure the setup. Finally, set PATH and JAVA_HOME environment variables to refer to the directory that contains java and javac, typically java_install_dir/bin and java_install_dir respectively.

If you are running Windows and installed the SDK in C:\jdk1.5.0_20, you should be inputting the following line in your C:\autoexec.bat file.
set PATH = C:\jdk1.5.0_20\bin;%PATH% set JAVA_HOME = C:\jdk1.5.0_20 Alternatively, on Windows NT/2000/XP –

- You can right-click on My Computer, Select Properties, then Advanced, then Environment Variables. Then, you would update the PATH value and press the OK button.
- On Unix (Solaris, Linux, etc.), if the SDK is installed in /usr/local/jdk1.5.0_20 and you use the C shell, you would put the following into your .cshrc file.

On Unix (Solaris, Linux, etc.), if the SDK is installed in /usr/local/jdk1.5.0_20 and you use the C shell, you would put the following into your .cshrc file.

setenv PATH /usr/local/jdk1.5.0_20/bin:$PATH setenv JAVA_HOME /usr/local/jdk1.5.0_20 Alternatively, if you use an Integrated Development Environment (IDE) like Borland JBuilder, Eclipse, IntelliJ IDEA, or Sun ONE Studio, compile and run a simple program to confirm that the IDE knows where you installed Java, otherwise do proper setup as per the given document of IDE.

STEP 2 - SETUP APACHE TOMCAT

You can download the latest version of Tomcat from https://tomcat.apache.org/. Once you downloaded the installation, unpack the binary distribution into a convenient location.

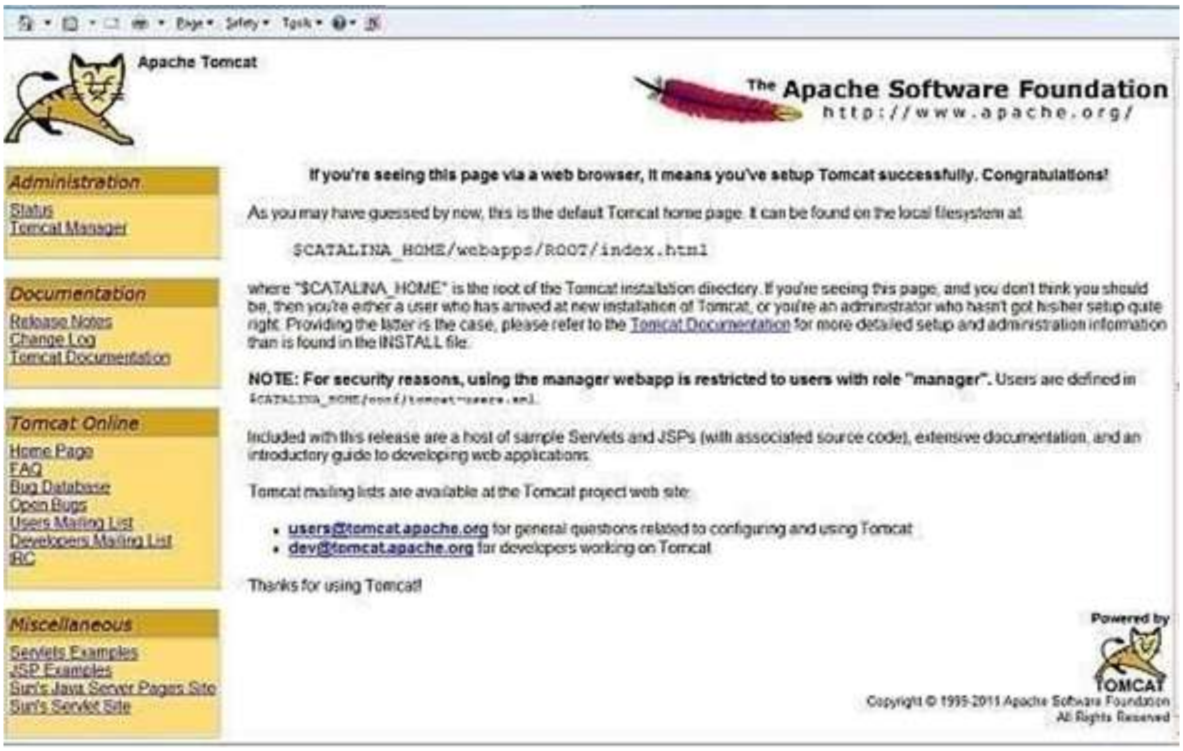
For example in C:\apache-tomcat-6.0.33 on windows, or /usr/local/apachetomcat-6.0.33 on Linux/Unix and create CATALINA_HOME environment variable pointing to these locations. You can start Tomcat by executing the following commands on windows machine, or you can simply double click on startup.bat

```
%CATALINA_HOME%\bin\startup.bat or
C:\apache-tomcat-6.0.33\bin\startup.bat
```

Tomcat can be started by executing the following commands on Unix (Solaris, Linux, etc.) machine –

```
$CATALINA_HOME/bin/startup.sh or
/usr/local/apache-tomcat-6.0.33/bin/startup.sh
```

After a successful startup, the default web applications included with Tomcat will be available by visiting **http://localhost:8080/**. If everything is fine, then it should display the following result –



If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!

As you may have guessed by now, this is the default Tomcat home page. It can be found on the local filesystem at:

```
$CATALINA_HOME/webapps/ROOT/index.html
```

where "$CATALINA_HOME" is the root of the Tomcat installation directory. If you're seeing this page, and you don't think you should be, then you're either a user who has arrived at new installation of Tomcat, or you're an administrator who hasn't got his/her setup quite right. Providing the latter is the case, please refer to the Tomcat Documentation for more detailed setup and administration information than is found in the INSTALL file.

NOTE: For security reasons, using the manager webapp is restricted to users with role "manager". Users are defined in `$CATALINA_HOME/conf/tomcat-users.xml`.

Included with this release are a host of sample Servlets and JSPs (with associated source code), extensive documentation, and an introductory guide to developing web applications.

Tomcat mailing lists are available at the Tomcat project web site:

- **users@tomcat.apache.org** for general questions related to configuring and using Tomcat
- **dev@tomcat.apache.org** for developers working on Tomcat

Thanks for using Tomcat!

Copyright © 1999-2011 Apache Software Foundation
All Rights Reserved

Further information about configuring and running Tomcat can be found in the documentation included here, as well as on the Tomcat website: https://tomcat.apache.org/

Tomcat can be stopped by executing the following commands on windows machine –

```
%CATALINA_HOME%\bin\shutdown or
C:\apache-tomcat-5.5.29\bin\shutdown
```

Tomcat can be stopped by executing the following commands on Unix (Solaris, Linux, etc.) machine –

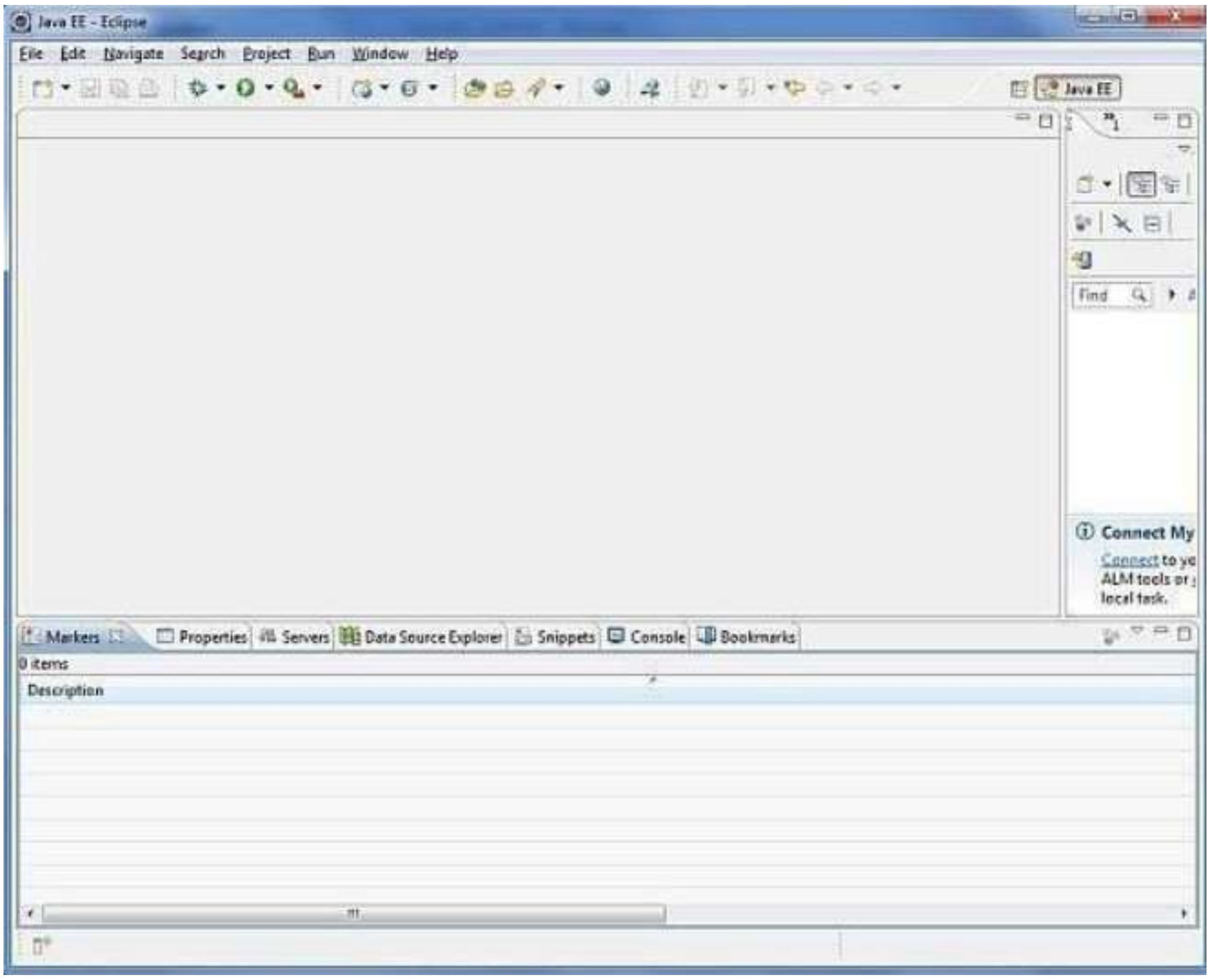
```
$CATALINA_HOME/bin/shutdown.sh
or

/usr/local/apache-tomcat-5.5.29/bin/shutdown.sh
```

STEP 3 - SETUP ECLIPSE (IDE)

All the examples in this tutorial are written using Eclipse IDE. I suggest that, you have the latest version of Eclipse installed in your machine.
To install Eclipse Download the latest Eclipse binaries from https://www.eclipse.org/downloads/. Once you download the installation, unpack the binary distribution into a convenient location.

For example in C:\eclipse on windows, or /usr/local/eclipse on Linux/Unix and finally set PATH variable appropriately. Eclipse can be started by executing the following commands on windows machine, or you can simply double click on eclipse.exe
%C:\eclipse\eclipse.exe

Eclipse can be started by executing the following commands on Unix (Solaris, Linux, etc.) machine −
$/usr/local/eclipse/eclipse
After a successful startup, if everything is fine, it should display the following result –

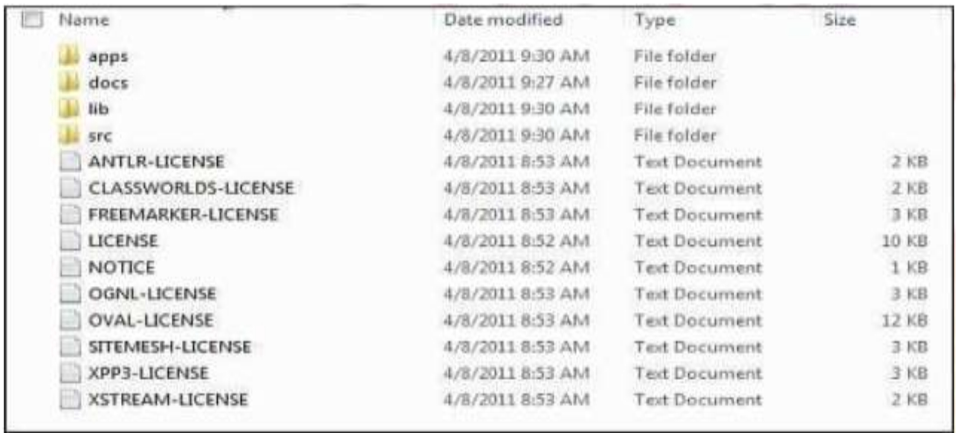


STEP 4 - SETUP STRUTS2 LIBRARIES

Now if everything is fine, then you can proceed to setup your Struts2 framemwork. Following are the simple steps to download and install Struts2 on your machine.

- Make a choice whether you want to install Struts2 on Windows, or Unix and then proceed to the next step to download .zip file for windows and .tz file for Unix.

- Download the latest version of Struts2 binaries from https://struts.apache.org/download.cgi.
- At the time of writing this tutorial, I downloaded **struts-2.0.14-all.zip** and when you unzip the downloaded file it will give you directory structure inside C:\struts-2.2.3 as follows.



Name	Date modified	Type	Size
apps	4/8/2011 9:30 AM	File folder	
docs	4/8/2011 9:27 AM	File folder	
lib	4/8/2011 9:30 AM	File folder	
src	4/8/2011 9:30 AM	File folder	
ANTLR-LICENSE	4/8/2011 8:53 AM	Text Document	2 KB
CLASSWORLDS-LICENSE	4/8/2011 8:53 AM	Text Document	2 KB
FREEMARKER-LICENSE	4/8/2011 8:53 AM	Text Document	3 KB
LICENSE	4/8/2011 8:52 AM	Text Document	10 KB
NOTICE	4/8/2011 8:52 AM	Text Document	1 KB
OGNL-LICENSE	4/8/2011 8:53 AM	Text Document	3 KB
OVAL-LICENSE	4/8/2011 8:53 AM	Text Document	12 KB
SITEMESH-LICENSE	4/8/2011 8:53 AM	Text Document	3 KB
XPP3-LICENSE	4/8/2011 8:53 AM	Text Document	3 KB
XSTREAM-LICENSE	4/8/2011 8:53 AM	Text Document	2 KB

Second step is to extract the zip file in any location, I downloaded & extracted **struts-2.2.3-all.zip** in **c:\** folder on my Windows 7 machine so that I have all the jar files into **C:\struts-2.2.3\lib**. Make sure you set your CLASSPATH variable properly otherwise you will face problem while running your application.

ARCHITECTURE:

From a high level, Struts2 is a pull-MVC (or MVC2) framework. The Model-View Controller pattern in Struts2 is implemented with the following five core components –

- Actions
- Interceptors
- Value Stack / OGNL
- Results / Result types
- View technologies

Struts are slightly different from a traditional MVC framework, where the action takes the role of the model rather than the controller, although there is some overlap.

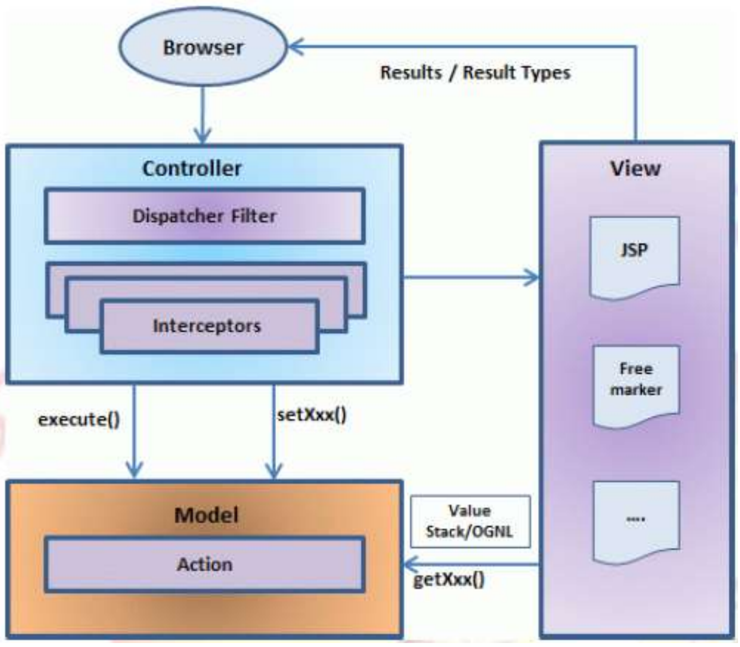


Fig. 02. Architecture

The above diagram depicts the **M**odel, **V**iew and **C**ontroller to the Struts2 high level architecture. The controller is implemented with a **Struts2** dispatch servlet filter as well as interceptors, this model is implemented with actions, and the view is a combination of result types and results. The value stack and OGNL provides common thread, linking and enabling integration between the other components. Apart from the above components, there will be a lot of information that relates to configuration. Configuration for the web application, as well as configuration for actions, interceptors, results, etc.

This is the architectural overview of the Struts MVC pattern. We will go through each component in more detail in the subsequent chapters.

REQUEST LIFE CYCLE:

Based on the above diagram, you can understand the work flow through user's request life cycle in **Struts** as follows −

- User sends a request to the server for requesting for some resource (i.e. pages).
- The Filter Dispatcher looks at the request and then determines the appropriate Action.
- Configured interceptor functionalities applies such as validation, file upload etc.
- Selected action is performed based on the requested operation.
- Again, configured interceptors are applied to do any post-processing if required.
- Finally, the result is prepared by the view and returns the result to the user.

HELLO WORLD EXAMPLE:

As you have already learnt from the Struts architecture, when you click on a hyperlink or submit an HTML form in a Struts web-application, the input is collected by the Controller which is sent to a Java class called Actions. After the Action is executed, a result selects a resource to render the response. The resource is generally a JSP, but it can also be a PDF file, an Excel spreadsheet, or a Java applet window.

Assuming that you already have built your development environment. Now, let us proceed for building our first **Hello World Struts2** project. The aim of this project is to build a web application that collects the user's name and displays "Hello World" followed by the user name. We would have to create following four components for any Struts project –

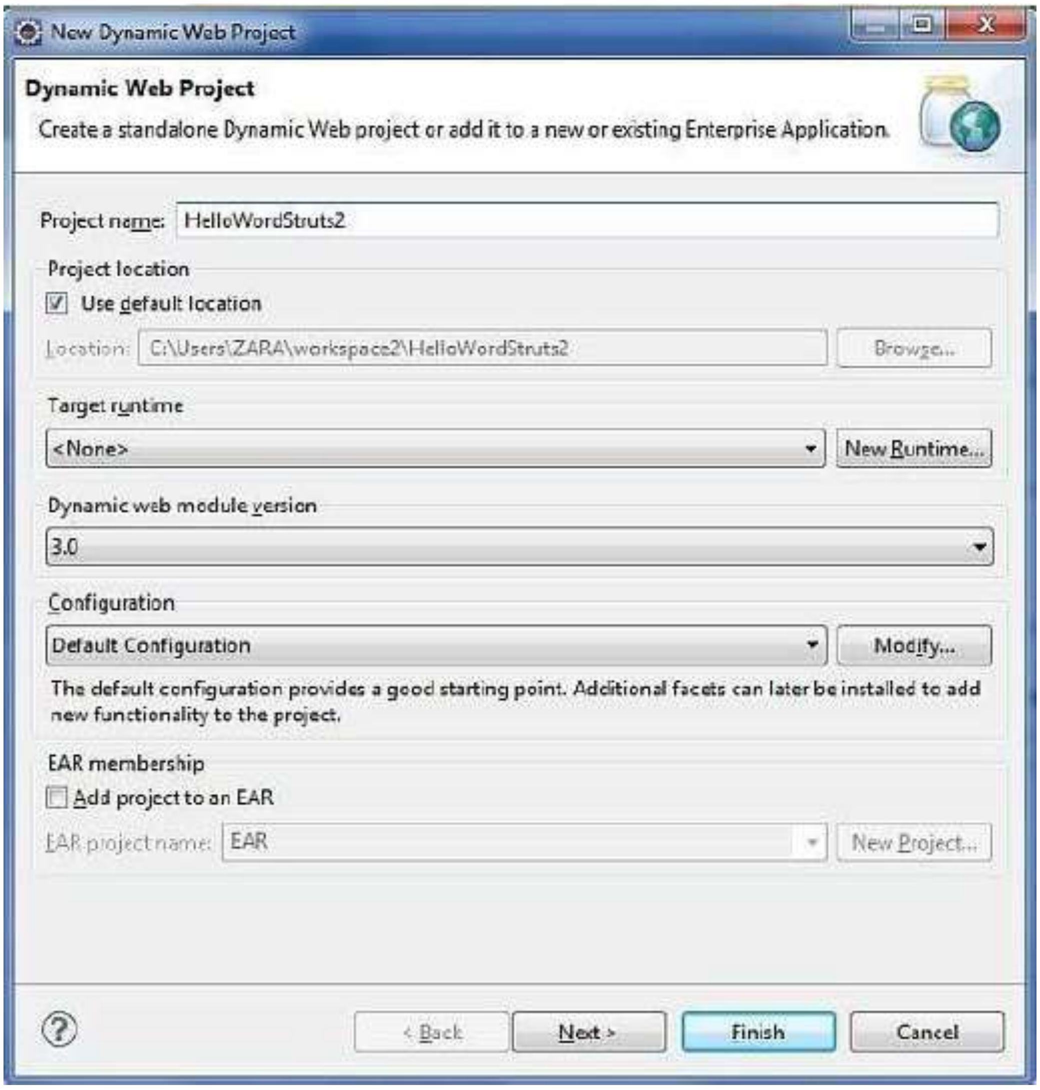
Table 1. Components of Struts

Sr. No.	Components & Description
1	**Action** Create an action class which will contain complete business logic and control the interaction between the user, the model, and the view.
2	**Interceptors** Create interceptors if required, or use existing interceptors. This is part of Controller.
3	**View** Create a JSPs to interact with the user to take input and to present the final messages.
4	**Configuration Files** Create configuration files to couple the Action, View and Controllers. These files are struts.xml, web.xml, struts.properties.

I am going to use Eclipse IDE, so that all the required components will be created under a Dynamic Web Project. Let us now start with creating Dynamic Web Project.

CREATE A DYNAMIC WEB PROJECT:

Start your Eclipse and then go with **File > New > Dynamic Web Project** and enter project name as **HelloWorldStruts2** and set rest of the options as given in the following screen –



New Dynamic Web Project

Dynamic Web Project

Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name: HelloWordStruts2

Project location

☑ Use default location

Location: C:\Users\ZARA\workspace2\HelloWordStruts2

Target runtime: <None>

Dynamic web module version: 3.0

Configuration: Default Configuration

The default configuration provides a good starting point. Additional facets can later be installed to add new functionality to the project.

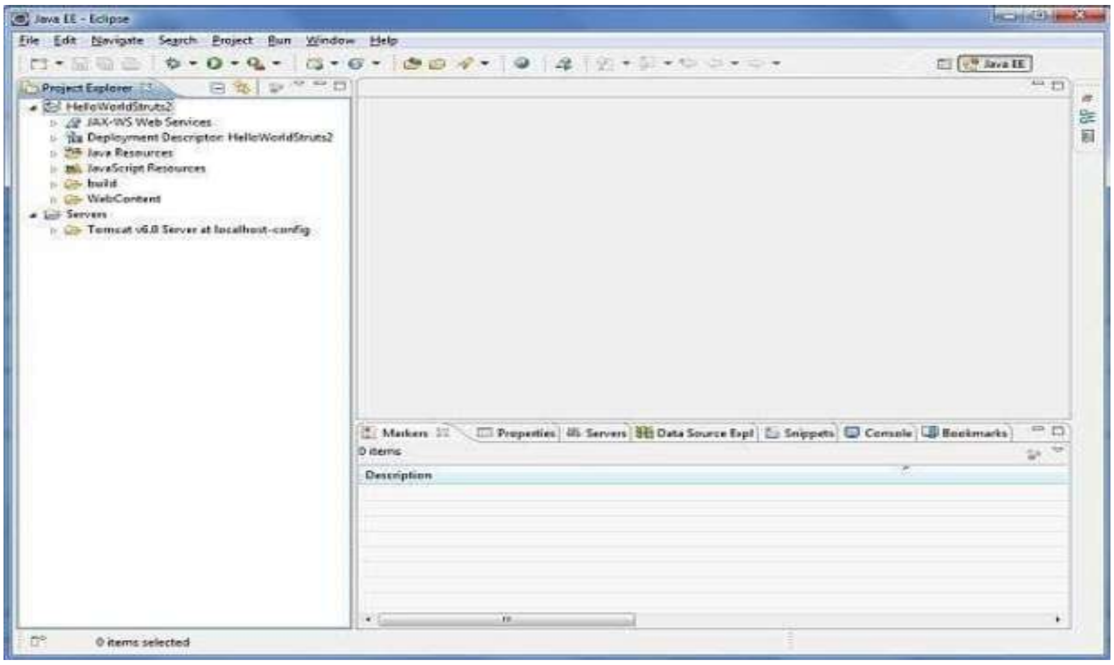
EAR membership

☐ Add project to an EAR

EAR project name: EAR

Select all the default options in the next screens and finally check **Generate Web.xml deployment descriptor** option. This will create a dynamic web project for you in Eclipse.

Now go with **Windows > Show View > Project Explorer**, and you will see your project window something as below –



Now copy following files from Struts lib folder **C:\struts-2.2.3\lib** to our project's **WEB-INF\lib** folder. To do this, you can simply drag and drop all the following files into WEB-INF\lib folder.

- commons-fileupload-x.y.z.jar
- commons-io-x.y.z.jar
- commons-lang-x.y.jar
- commons-logging-x.y.z.jar
- commons-logging-api-x.y.jar
- freemarker-x.y.z.jar
- javassist-.xy.z.GA
- ognl-x.y.z.jar
- struts2-core-x.y.z.jar
- xwork-core.x.y.z.jar

CREATE ACTION CLASS:

Action class is the key to Struts application and we implement most of the business logic in action class. So let us create a java file HelloWorldAction.java under **Java Resources > src** with a package name **com.tutorialspoint.struts2** with the contents given below.

The Action class responds to a user action when user clicks a URL. One or more of the Action class's methods are executed and a String result is returned. Based on the value of the result, a specific JSP page is rendered.

```
package com.tutorialspoint.struts2;
public class HelloWorldAction {
private String name;
public String execute() throws Exception { return "success";
}
public String getName() { return name;
}
public void setName(String name) { this.name = name;
}
}
```

This is a very simple class with one property called "name". We have standard getters and setter methods for the "name" property and an execute method that returns the string "success".

The Struts framework will create an object of the **HelloWorldAction** class and call the executed method in response to a user's action. You put your business logic inside this method which finally returns the String constant. In other words, for each URL, you would have to implement one action class and either you can use that class name directly as your action name or you can map to some other name using struts.xml file as shown below.

CREATE A VIEW:

We need a JSP to present the final message, this page will be called by Struts framework when a predefined action will happen and this mapping will be defined in struts.xml file. So let us create the below jsp file **HelloWorld.jsp** in the WebContent folder in your eclipse project. To do this, right click on the WebContent folder in the project explorer and select **New >JSP File**.

```
<%@ page contentType = "text/html; charset = UTF-8" %>
<%@ taglib prefix = "s" uri = "/struts-tags" %>
<html>
        <head>
            <title>Hello World</title>
        </head>
        <body>
            Hello World, <s:property value = "name"/>
        </body>
</html>
```

The taglib directive tells the Servlet container that this page will be using the **Struts** tags and that these tags will be preceded by **s**.

The s:property tag displays the value of action class property "name> which is returned by the method **getName()** of the HelloWorldAction class.

CREATE MAIN PAGE:

We also need to create **index.jsp** in the WebContent folder. This file will serve as the initial action URL where a user can click to tell the Struts framework to call a defined method of the HelloWorldAction class and render the HelloWorld.jsp view.

```
<%@ 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>
            <title>Hello World</title>
        </head>
        <body>
            <h1>Hello World From Struts2</h1>
            <form action = "hello">
                <label for = "name">Please enter your name</label><br/>
                <input type = "text" name = "name"/>
                <input type = "submit" value = "Say Hello"/>
            </form>
        </body>
</html>
```

The **hello** action defined in the above view file will be mapped to the HelloWorldAction class and its execute method using struts.xml file. When a user clicks on the Submit button it will cause the Struts framework to run the execute method defined in the HelloWorldAction class and based on the returned value of the method, an appropriate view will be selected and rendered as a response.

CONFIGURATION FILES:

We need a mapping to tie the URL, the HelloWorldAction class (Model), and the HelloWorld.jsp (the view) together. The mapping tells the Struts framework which class will respond to the user's action (the URL), which method of that class will be executed, and what view to render based on the String result that method returns.

So let us create a file called **struts.xml**. Since Struts requires struts.xml to be present in the classes folder. Hence, create struts.xml file under the WebContent/WEB-INF/classes folder. Eclipse does not create the "classes" folder by default, so you need to do this yourself. To do this, right click on the WEB-INF folder in the project explorer and select **New > Folder**. Your struts.xml should look like –

```
<?xml version = "1.0" Encoding = "UTF-8"?>
<!DOCTYPE struts PUBLIC
"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
"http://struts.apache.org/dtds/struts-2.0.dtd">
<struts>
        <constant name = "struts.devMode" value = "true" />
        <package name = "helloworld" extends = "struts-default">
        <action name = "hello" class =
        "com.tutorialspoint.struts2.HelloWorldAction" method = "execute">
        <result name = "success">/HelloWorld.jsp</result>
        </action>
        </package>
</struts>
```

Few words which need to be understood regarding the above configuration file. Here, we set the constant **struts.devMode** to **true**, because we are working in development environment and we need to see some useful log messages. Then, we define a package called **helloworld**.
Creating a package is useful when you want to group your actions together. In our example, we named our action as "hello" which is corresponding to the URL **/hello.action** and is

backed up by the**HelloWorldAction.class**. The **execute** method of **HelloWorldAction.class** is the method that is run when the URL **/hello.action** is invoked. If the outcome of the execute method returns "success", then we take the user to **HelloWorld.jsp**.

Next step is to create a **web.xml** file which is an entry point for any request to Struts. The entry point of Struts2 application will be a filter defined in deployment descriptor (web.xml). Hence, we will define an entry of org.apache.struts2.dispatcher.FilterDispatcher class in web.xml. The web.xml file needs to be created under the WEB-INF folder under WebContent. Eclipse had already created a skeleton web.xml file for you when you created the project. So, lets just modify it as follows –

```
<?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"
xmlns:web = "http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" 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>Struts</display-name>
<welcome-file-list>
<welcome-file>index.jsp</welcome-file>
</welcome-file-list>
<filter>
<filter-name>struts2</filter-name>
<filter-class> org.apache.struts2.dispatcher.FilterDispatcher
</filter-class>
</filter>
<filter-mapping>
<filter-name>struts2</filter-name>
<url-pattern>/*</url-pattern>
</filter-mapping>
</web-app>
```

We have specified index.jsp to be our welcome file. Then we have configured the Struts2 filter to run on all urls (i.e, any url that match the pattern /*)

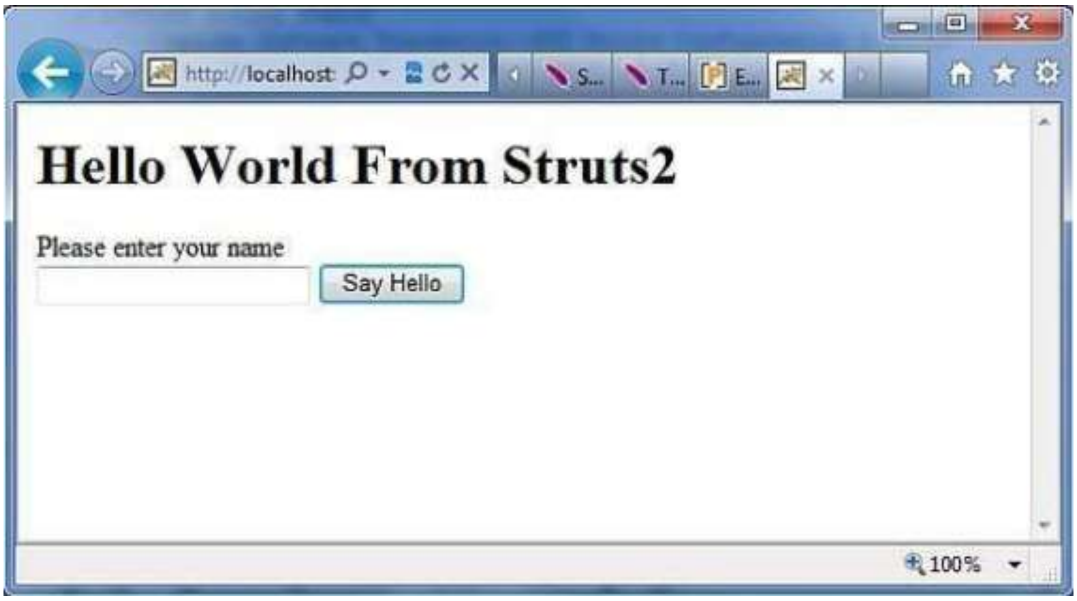
TO ENABLE DETAILED LOG:

You can enable complete logging functionality while working with Struts by creating **logging.properties** file under **WEB-INF/classes** folder. Keep the following two lines in your property file –

```
org.apache.catalina.core.ContainerBase.[Catalina].level = INFO
org.apache.catalina.core.ContainerBase.[Catalina].handlers = \
java.util.logging.ConsoleHandler
```

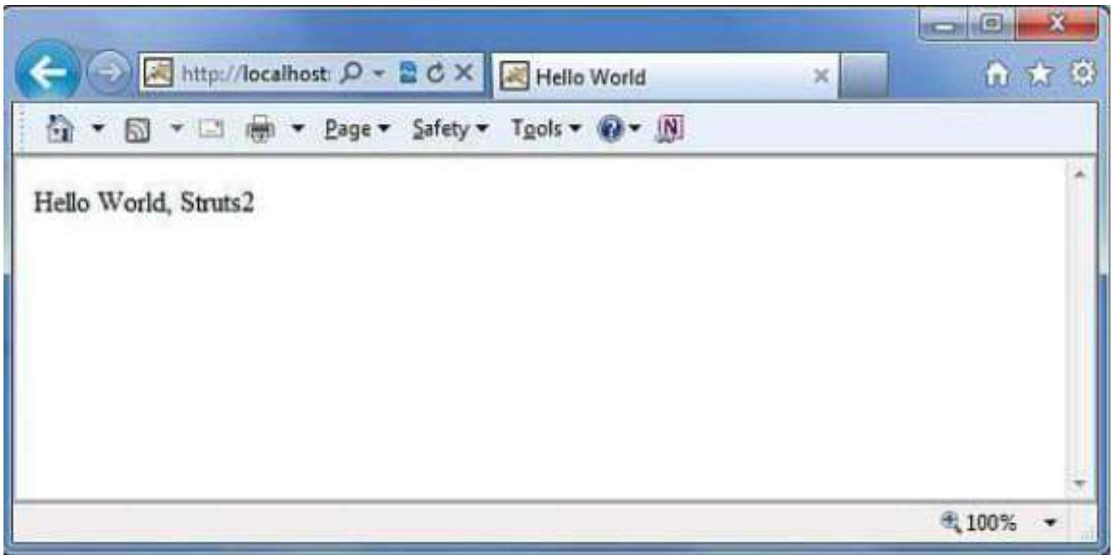
The default logging.properties specifies a ConsoleHandler for routing logging to stdout and also a FileHandler. A handler's log level threshold can be set using SEVERE, WARNING, INFO, CONFIG, FINE, FINER, FINEST or ALL.

PROCEDURE FOR EXECUTING THE APPLICATION:
Right click on the project name and click **Export > WAR File** to create a War file. Then deploy this WAR in the Tomcat's webapps directory.
Finally, start Tomcat server and try to access URL
http://localhost:8080/HelloWorldStruts2/index.jsp. This will give you following screen –


Hello World From Struts2

Please enter your name

Enter a value "Struts2" and submit the page. You should see the next page


Hello World, Struts2

Note that you can define **index** as an action in struts.xml file and in that case you can call index page as **http://localhost:8080/HelloWorldStruts2/index.action**. Check below how you can define index as an action –

```
<?xml version = "1.0" Encoding = "UTF-8"?>
<!DOCTYPE struts PUBLIC
"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
"http://struts.apache.org/dtds/struts-2.0.dtd">
<struts>
```

```
<constant name = "struts.devMode" value = "true" />
<package name = "helloworld" extends = "struts-default">
<action name = "index">
<result >/index.jsp</result>
</action>
<action name = "hello"
class = "com.tutorialspoint.struts2.HelloWorldAction" method = "execute">
<result name = "success">/HelloWorld.jsp</result>
</action>
</package>
</struts>
```

CONCLUSION: Hence we created dynamic website using Struts.