**SpringMVC Guide**

- First of all prepare your development environment either by downloading [STS](https://spring.io/tools/sts/all) or download [Eclipse](http://www.eclipse.org/downloads/eclipse-packages/)  then install [spring plugin](https://marketplace.eclipse.org/content/spring-ide) into it

- Download Maven

- Download apache tomcat 7 and configure tomcat users to have a user tomcat/password

- configure Maven to point to tomcat using MAVEN\_HOME\conf\settings.xml add the following configurations under servers section:

<server>

<id>tomcat7</id>

<username>tomcat</username>

<password>password</password>

</server>

**Target of this workshop:**

* Quick overview on spring mvc
* Create a hello world xml configuration based spring mvc application
* Create a Spring form to submit a bean and display it

**Start:**

* Start eclipse and create a new project
* To create a new Spring project you can do it in at least three ways
  1. Create a new dynamic web project and convert it into maven then add Spring dependencies and required configurations
  2. Create a new maven project then add spring dependencies and required configurations
  3. Create Spring Legacy project and select Simple Spring Web Maven
* We will take 2nd way so from file menu select New -> Maven Project
* Skip archetype selection and press Next
* Input Maven configuration as follows:

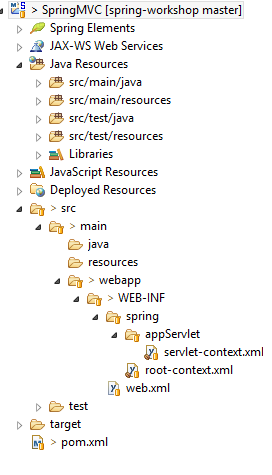
<groupId>com.ihorizons.spring.mvc</groupId>

<artifactId>SpringMVC</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>war</packaging>

* Replace pom.xml content with this  content which contains:
  + Spring MVC dependencies
  + Maven Eclipse plugin configurations
  + Tomcat deployment configurations
* Under src/main/webapp create WEB-INF folder and create web.xml inside it
* Replace web.xml content with  content which contains :
  + Context configurations path
  + Spring context listener
  + DispatcherServlet mapping
* Now create src\main\webapp\WEB-INF\spring\root-context.xml  and src\main\webapp\WEB-INF\spring\appServlet\servlet-context.xml 
* Content of servlet-context.xml is as follows:
  + Enabling annotations in Spring
  + Static resources mapping
  + InternalResourceViewResolver mapping to define views path
  + Base package definition to define where to start scanning for annotated components
* Now you have a Spring mvc xml configuration based project skeleton which looks as follows:



* Create com.ih.spring.HomeController

**package** com.ih.spring;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.servlet.ModelAndView;

@Controller

**public** **class** HomeController {

@RequestMapping(value = "/home")

**public** ModelAndView goHome() {

ModelAndView model = **new** ModelAndView("home");

model.addObject("msg", "Welcome to Spring FrameWork");

**return** model;

}

}

* Create a new View src\main\webapp\WEB-INF\views\home.jsp

<%@ page language=*"java"* contentType=*"text/html; charset=windows-1256"*

pageEncoding=*"windows-1256"*%>

<!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=windows-1256"*>

<title>Spring Welcome</title>

</head>

<body>

<b>${msg}</b>

</body>

</html>

* Now create a new Run Configurations using clean tomcat7:deploy
* Hit <http://localhost:8080/SpringMVC/home>

------- To this point we are done with the first two targets of the workshop --------

Now we will create a form view to input data and a result view to display the input data

* So in HomeController add the following two methods

@RequestMapping(value = "/form", method = RequestMethod.***GET***)

**public** String home(Locale locale, Model model) {

model.addAttribute("user",**new** User());

**return** "form";

}

@RequestMapping(value = "/result", method = RequestMethod.***GET***)

**public** String welcome(Locale locale,

Model model,@ModelAttribute("user") User user) {

model.addAttribute("userData", user);

**return** "result";

}

* Create two Views  and 

Run the application and check the result