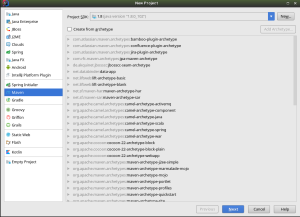
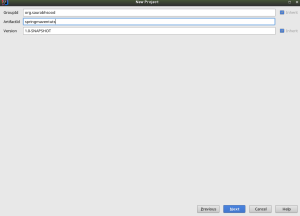
**With Maven**

1.) Create New Project -> Maven Project. Do not select an archetype right now. In the next screen, add an artifact ID and Group ID

[](https://iamsaurabh.files.wordpress.com/2017/02/1.png) [](https://iamsaurabh.files.wordpress.com/2017/02/2-create.png)

2.) When the project gets created, the pom.xml file gets opened. Intellij also gives a message saying ‘Maven projects need to be imported’. Select ‘Enable Auto-Import’.

3.) Add the following to the pom.xml file inside the <project> tag:

<packaging>war</packaging>

<properties>  
<spring.version>4.3.0.RELEASE</spring.version>  
</properties>

<dependencies>  
<dependency>  
<groupId>org.springframework</groupId>  
<artifactId>spring-core</artifactId>  
<version>${spring.version}</version>  
</dependency>

<dependency>  
<groupId>org.springframework</groupId>  
<artifactId>spring-web</artifactId>  
<version>${spring.version}</version>  
</dependency>

<dependency>  
<groupId>org.springframework</groupId>  
<artifactId>spring-webmvc</artifactId>  
<version>${spring.version}</version>  
</dependency>  
</dependencies>

Maven will auto import the modules for you

4.) Create a webapp/WEB-INF directory inside ‘main’  
5.) Create a web.xml file in WEB-INF

6.) Add the following code to the web.xml file:  
  
<?xml version="1.0" encoding="ISO-8859-1" ?>

<web-app xmlns=”<http://java.sun.com/xml/ns/j2ee&#8221>;  
xmlns:xsi=”<http://www.w3.org/2001/XMLSchema-instance&#8221>;  
xsi:schemaLocation=”<http://java.sun.com/xml/ns/j2ee><http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd&#8221>;  
version=”2.4″>

<display-name>HelloWorld Application</display-name>  
<description>  
This is a simple web application with a source code organization  
based on the recommendations of the Application Developer’s Guide.  
</description>

<context-param>  
<param-name>contextConfigLocation</param-name>  
<param-value>/WEB-INF/dispatcher-servlet.xml</param-value>  
</context-param>  
<listener>  
<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>  
</listener>  
<servlet>  
<servlet-name>dispatcher</servlet-name>  
<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>  
<load-on-startup>1</load-on-startup>  
</servlet>  
<servlet-mapping>  
<servlet-name>dispatcher</servlet-name>  
<url-pattern>/</url-pattern>  
</servlet-mapping>

</web-app>

7.) In WEB-INF, create a file called dispatcher-servlet.xml. Add the following code

<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
xmlns:context="http://www.springframework.org/schema/context"  
xmlns:mvc="http://www.springframework.org/schema/mvc"  
xsi:schemaLocation="http://www.springframework.org/schema/beans<http://www.springframework.org/schema/beans/spring-beans.xsd><http://www.springframework.org/schema/context><http://www.springframework.org/schema/context/spring-context.xsd><http://www.springframework.org/schema/mvc><http://www.springframework.org/schema/mvc/spring-mvc.xsd>">

<mvc:annotation-driven />  
<context:component-scan base-package=”org.saurabhsood.controllers” />

<bean class=”org.springframework.web.servlet.view.InternalResourceViewResolver”>  
<property name=”prefix” value=”/WEB-INF/views/” />  
<property name=”suffix” value=”.jsp” />  
</bean>

</beans>

**Notice the base-package attribute for context. You should change this to the package that contains your controllers (which you create in the next step)**

8.) Create the package which you specified in the dispatcher-servlet.xml. I created a class called IndexController. I had the following code:

package org.saurabhsood.controllers;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.ui.Model;

/\*\*

\* Created by saurabh on 2/10/17.

\*/

@Controller

public class IndexController {

@GetMapping("/")

public String index(Model m) {

m.addAttribute("someAttribute", "someValue");

return "index";

}

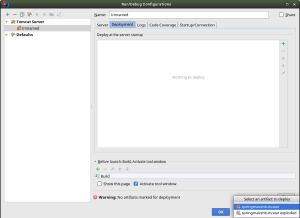
}

9.) Create a directory called views in WEB-INF. In that directory, add a file called index.jsp. Add the following line in <body>, to display the model attribute:

<p>${someAttribute}</p>

10.) Select Edit Run Configurations -> Tomcat -> Local.

11.) Look at the warnings. Click on Fix, and select the option with ‘war-exploded’ when asked to select the artifact to deploy.

[](https://iamsaurabh.files.wordpress.com/2017/02/3-fix-problems.png)

12.) Run the application. Hopefully you should get the output of the model information.

[](https://iamsaurabh.files.wordpress.com/2017/02/output-maven.png)

Hopefully this long post helps kick start the development, and spares the pain that I had to go through over the period of 4 days

<https://iamsaurabh.wordpress.com/2017/02/11/configure-a-spring-mvc-project-with-intellij-idea/>

**Include TOMCAT7-MAVEN-PLUGIN without installing local TOMCAT Server**

