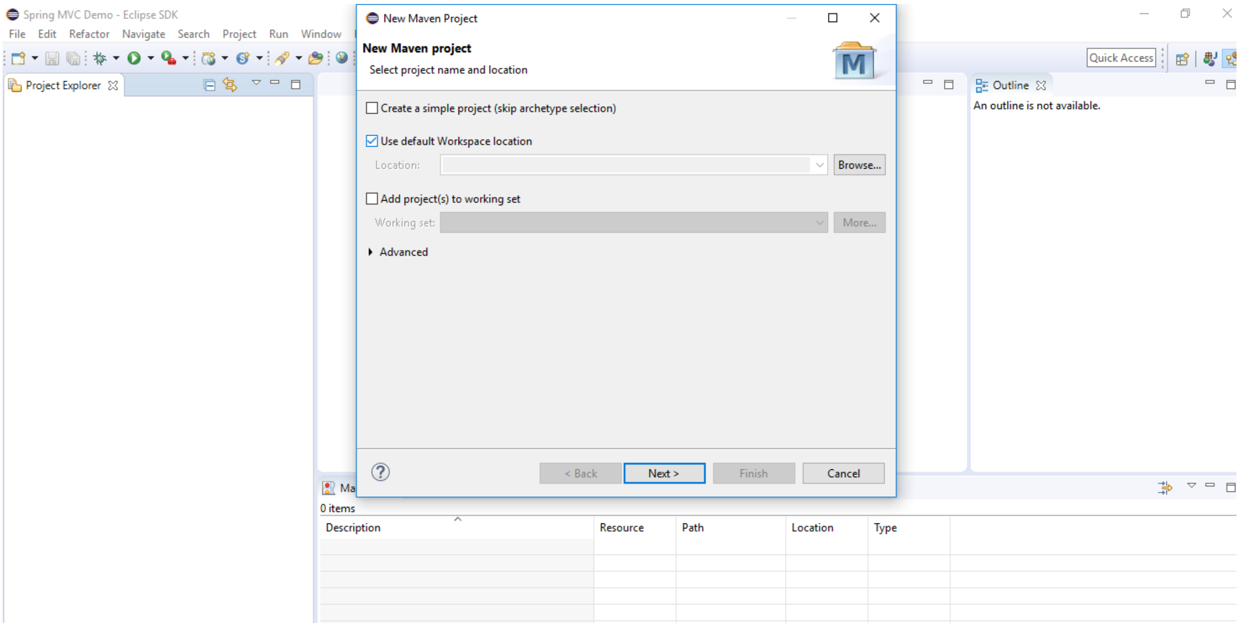
Spring MVC Project

1. Create Maven Project and choose use default workspace location
    a. Click Next



New Maven project

Select project name and location

☐ Create a simple project (skip archetype selection)

☑ Use default Workspace location

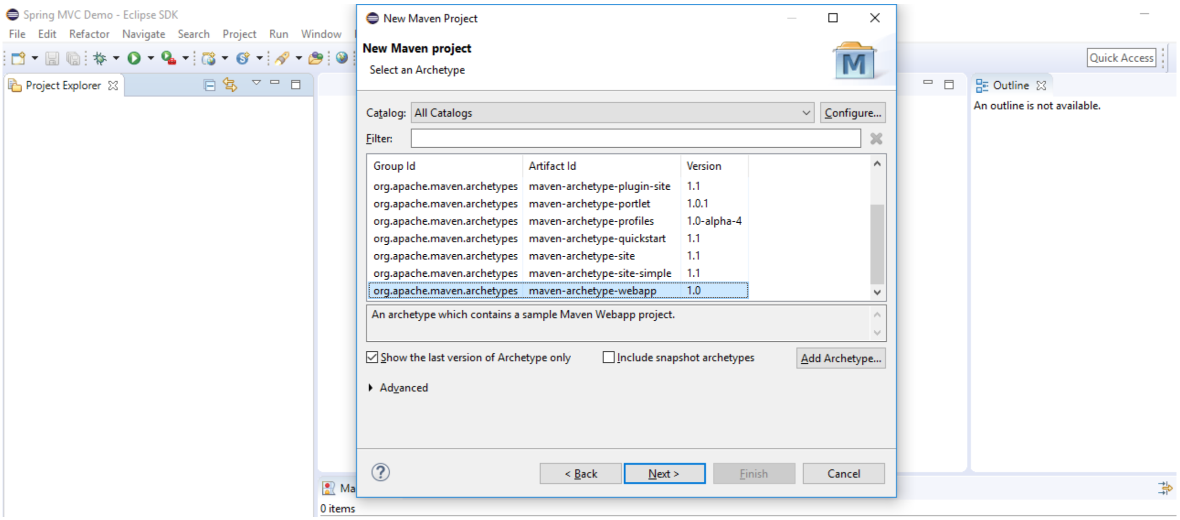
Location:

☐ Add project(s) to working set

Working set:

▸ Advanced

2. Under Archetype choose **maven-archetype-webapp** from below given group Id and version
    a. Group Id : **org.apache.maven.archetypes**
    b. Version can be any latest version. Here It is taken 1.0
        i. You can also pick a specific version by checking the ***Include snapshot archetypes***
    c. Click Next



New Maven project

Select an Archetype

Catalog: All Catalogs

Filter:

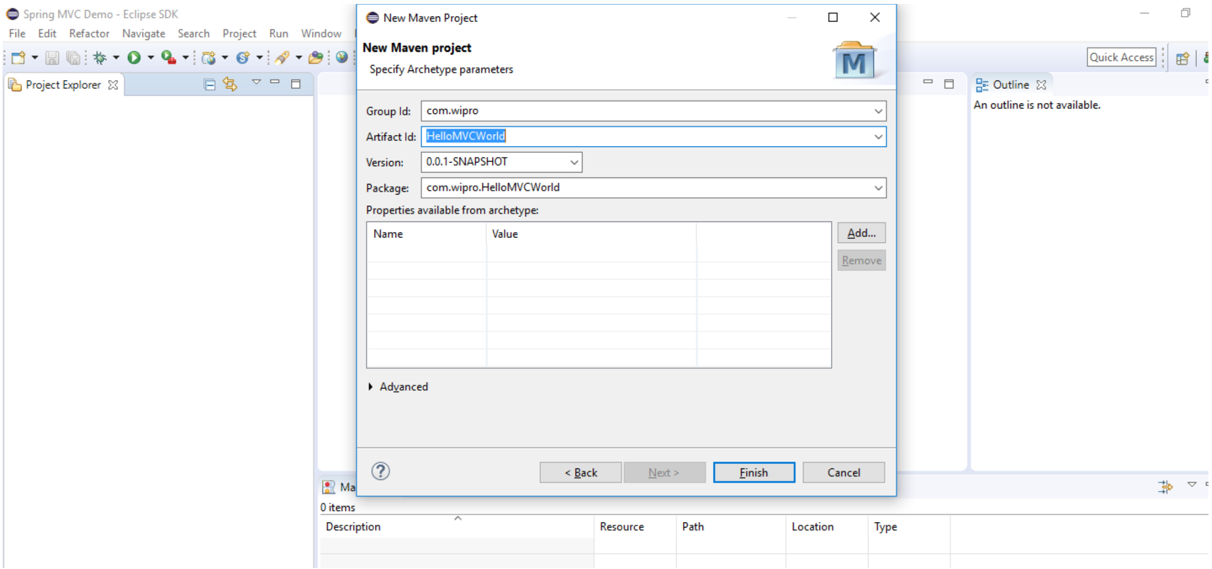
Group Id	Artifact Id	Version
org.apache.maven.archetypes	maven-archetype-plugin-site	1.1
org.apache.maven.archetypes	maven-archetype-portlet	1.0.1
org.apache.maven.archetypes	maven-archetype-profiles	1.0-alpha-4
org.apache.maven.archetypes	maven-archetype-quickstart	1.1
org.apache.maven.archetypes	maven-archetype-site	1.1
org.apache.maven.archetypes	maven-archetype-site-simple	1.1
org.apache.maven.archetypes	maven-archetype-webapp	1.0

An archetype which contains a sample Maven Webapp project.

☑ Show the last version of Archetype only ☐ Include snapshot archetypes

▸ Advanced

3. Fill the Maven Project details
    a. Group ID :: com.wipro
    b. Artifact ID :: HelloMVCWorld
    c. Click finish



Spring MVC Demo - Eclipse SDK
File Edit Refactor Navigate Search Project Run Window
Project Explorer
Quick Access
Outline
< Back
Next >
Finish
Cancel
Add...
Remove
Advanced

New Maven Project

New Maven project

Specify Archetype parameters

Group Id: com.wipro

Artifact Id: HelloMVCWorld

Version: 0.0.1-SNAPSHOT

Package: com.wipro.HelloMVCWorld

Properties available from archetype:

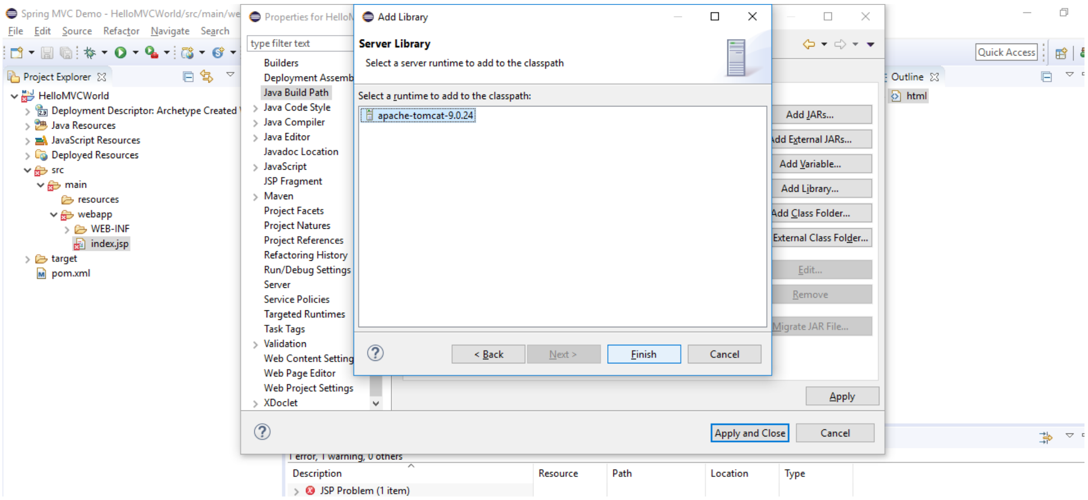
Name	Value

An outline is not available.

0 items

Description	Resource	Path	Location	Type

4. This generates Spring Web Project. With the initial Error
    a. This is because the build path is not aware of servlets and jsp APIs.
    b. Add Tomcat server
        i. Click on window menu -> preferences -> server -> Runtime Environment
            1. Search the unzipped folder of your Apache Tomcat Server
    c. right click on project -> Properties -> Java Build Path -> Add Library
    d. Server Runtime -> Apache Tomcat -> Finish



Add Library

Server Library

Select a server runtime to add to the classpath

Select a runtime to add to the classpath:

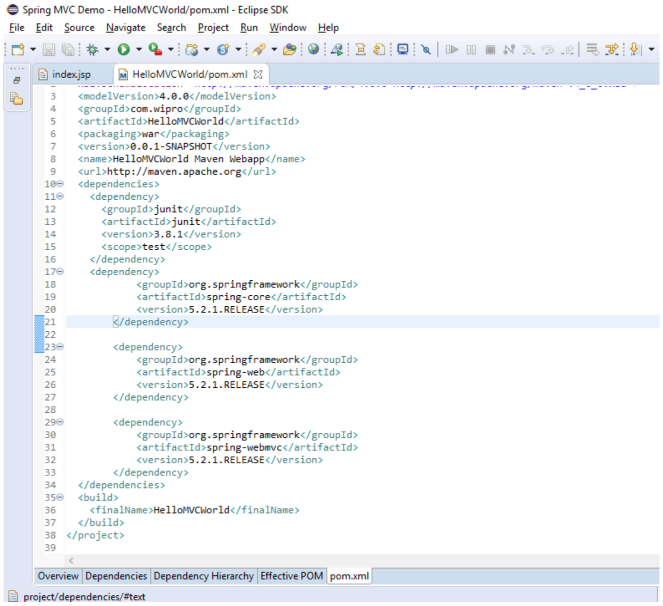
apache-tomcat-9.0.24

1 error, 1 warning, 0 others

Description	Resource	Path	Location	Type
JSP Problem (1 item)				

5. Set the spring dependencies.
    a. Open POM file
    b. Add the following spring dependencies under the tag <dependencies>

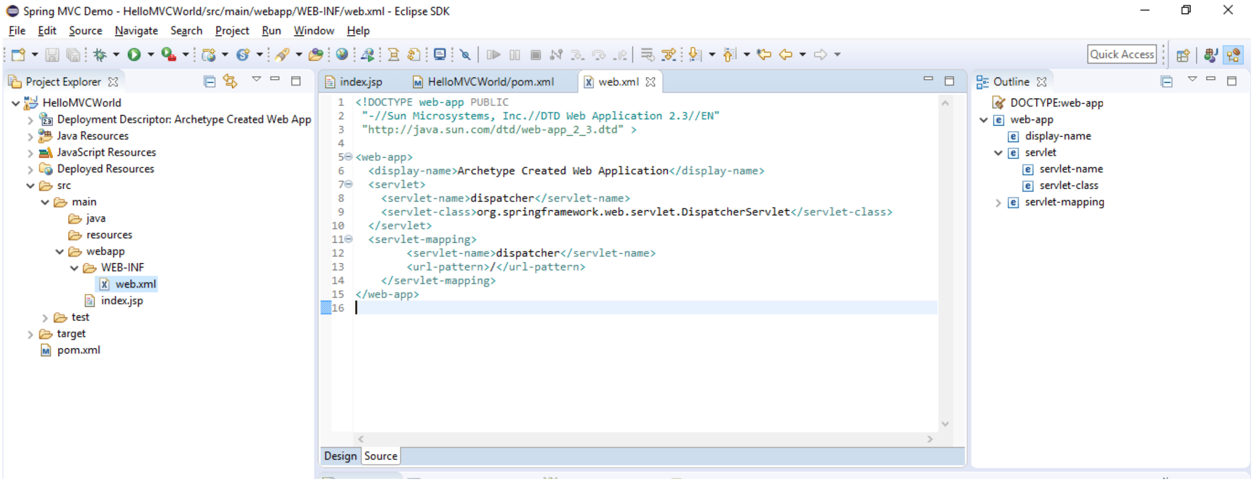
```
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-core</artifactId>
        <version>5.2.1.RELEASE</version>
</dependency>
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-web</artifactId>
        <version>5.2.1.RELEASE</version>
</dependency>
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-webmvc</artifactId>
        <version>5.2.1.RELEASE</version>
</dependency>
```


Spring MVC Demo - HelloMVCWorld/pom.xml - Eclipse SDK
File Edit Source Navigate Search Project Run Window Help
index.jsp
HelloMVCWorld/pom.xml
Overview
Dependencies
Dependency Hierarchy
Effective POM
pom.xml
project/dependencies/#text

```
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.wipro</groupId>
  <artifactId>HelloMVCWorld</artifactId>
  <packaging>war</packaging>
  <version>0.0.1-SNAPSHOT</version>
  <name>HelloMVCWorld Maven Webapp</name>
  <url>http://maven.apache.org</url>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>
    <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-core</artifactId>
            <version>5.2.1.RELEASE</version>
        </dependency>

        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-web</artifactId>
            <version>5.2.1.RELEASE</version>
        </dependency>

        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-webmvc</artifactId>
            <version>5.2.1.RELEASE</version>
        </dependency>
  </dependencies>
  <build>
    <finalName>HelloMVCWorld</finalName>
  </build>
</project>
```

6. Set the dispatcher Servlet.
    a. Edit the web.xml file
    b. Include <servlet> and <servlet-mapping>

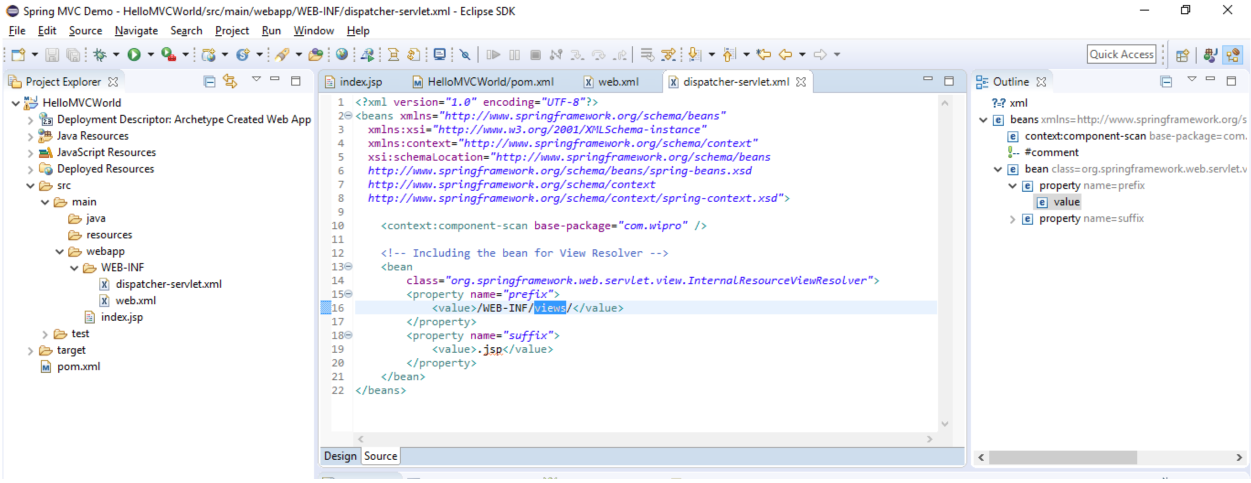
```
    <servlet>
       <servlet-name>dispatcher</servlet-name>
      <servlet-class>org.springframework.web.servlet.DispatcherServlet
       </servlet-class>
       </servlet>
       <servlet-mapping>
              <servlet-name>dispatcher</servlet-name>
              <url-pattern>/</url-pattern>
      </servlet-mapping>
```



```
<!DOCTYPE web-app PUBLIC
 "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
 "http://java.sun.com/dtd/web-app_2_3.dtd" >

<web-app>
  <display-name>Archetype Created Web Application</display-name>
  <servlet>
    <servlet-name>dispatcher</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
  </servlet>
  <servlet-mapping>
        <servlet-name>dispatcher</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>
</web-app>

```

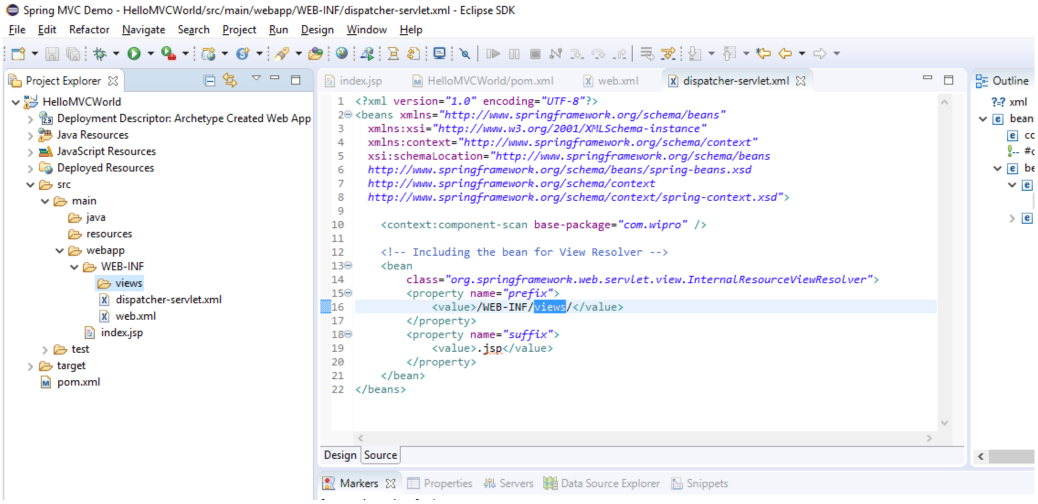
7. Create the Spring configuration file named **dispatcher-servlet.xml**

```
<?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"
  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">

       <context:component-scan base-package="com.wipro" />

       <!-- Including the bean for View Resolver -->
<bean
class="org.springframework.web.servlet.view.InternalResourceViewResolver">
              <property name="prefix">
                     <value>/WEB-INF/views/</value>
              </property>
              <property name="suffix">
                     <value>.jsp</value>
              </property>
       </bean>
</beans>
```


Spring MVC Demo - HelloMVCWorld/src/main/webapp/WEB-INF/dispatcher-servlet.xml - Eclipse SDK
File Edit Source Navigate Search Project Run Window Help
Project Explorer
HelloMVCWorld
Deployment Descriptor: Archetype Created Web App
Java Resources
JavaScript Resources
Deployed Resources
src
main
java
resources
webapp
WEB-INF
dispatcher-servlet.xml
web.xml
index.jsp
test
target
pom.xml
index.jsp
HelloMVCWorld/pom.xml
web.xml
dispatcher-servlet.xml
Design
Source
Outline
xml
beans xmlns=http://www.springframework.org/s
context:component-scan base-package=com.
#comment
bean class=org.springframework.web.servlet.v
property name=prefix
value
property name=suffix
Quick Access

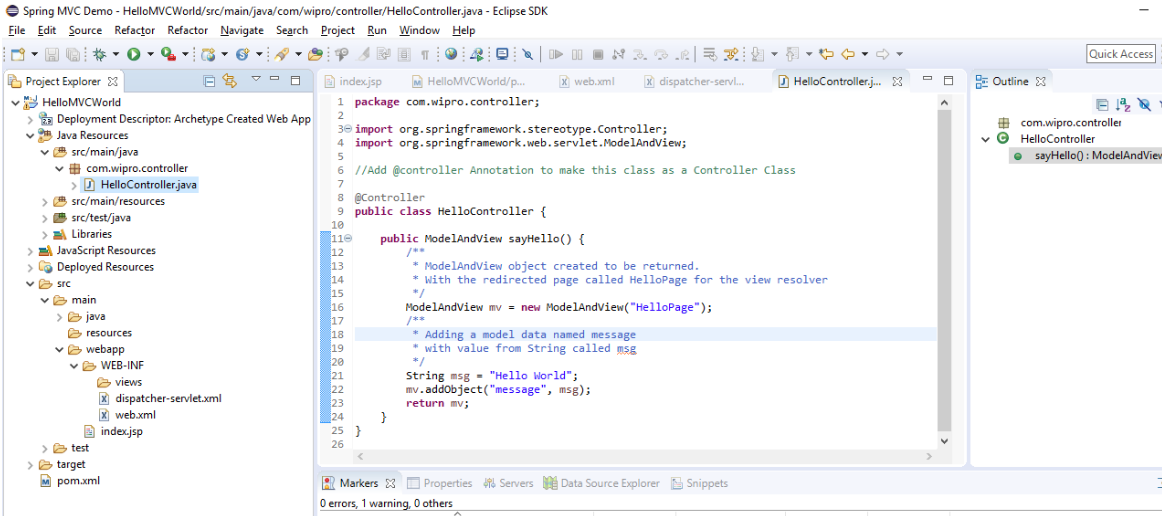
```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:context="http://www.springframework.org/schema/context"
  xsi:schemaLocation="http://www.springframework.org/schema/beans
  http://www.springframework.org/schema/beans/spring-beans.xsd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

    <context:component-scan base-package="com.wipro" />

    <!-- Including the bean for View Resolver -->
    <bean
        class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix">
            <value>/WEB-INF/views/</value>
        </property>
        <property name="suffix">
            <value>.jsp</value>
        </property>
    </bean>
</beans>
```

8. Create the views folder under WEB-INF for defining the views
    a. Right click on WEB-INF folder NEW -> Folder (Enter Name as **views**) - > finish.



```xml
<?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"
  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">

    <context:component-scan base-package="com.wipro" />

    <!-- Including the bean for View Resolver -->
    <bean
        class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix">
            <value>/WEB-INF/views/</value>
        </property>
        <property name="suffix">
            <value>.jsp</value>
        </property>
    </bean>
</beans>
```

9. Create the controller class named HelloController
    a. Right click on Java Resources and create a package called com.wipro.controller
    b. Under the package Create a class called HelloController



```
package com.wipro.controller;

import org.springframework.stereotype.Controller;
import org.springframework.web.servlet.ModelAndView;

//Add @controller Annotation to make this class as a Controller Class

@Controller
public class HelloController {

    public ModelAndView sayHello() {
        /**
         * ModelAndView object created to be returned.
         * With the redirected page called HelloPage for the view resolver
         */
        ModelAndView mv = new ModelAndView("HelloPage");
        /**
         * Adding a model data named message
         * with value from String called msg
         */
        String msg = "Hello World";
        mv.addObject("message", msg);
        return mv;
    }
}

```

0 errors, 1 warning, 0 others

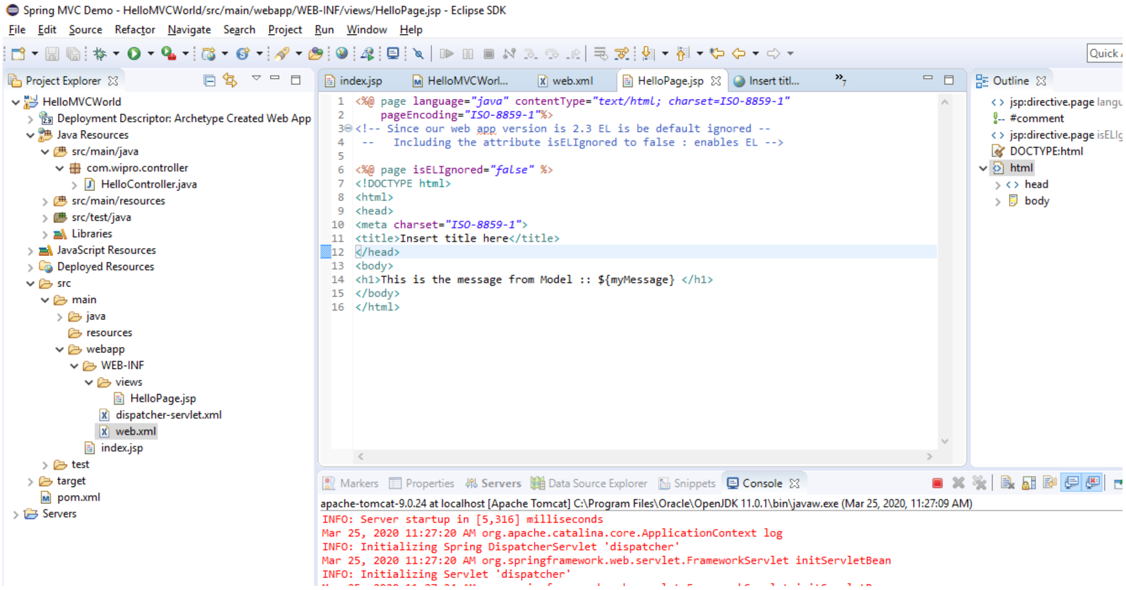
```
package com.wipro.controller;

import org.springframework.stereotype.Controller;
import org.springframework.web.servlet.ModelAndView;

//Add @controller Annotation to make this class as a Controller Class

@Controller
public class HelloController {
    @RequestMapping("/Hello")
    public ModelAndView sayHello() {
            /**
             * ModelAndView object created to be returned.
             * With the redirected page called HelloPage for the view resolver
             */
            ModelAndView mv = new ModelAndView("HelloPage");
            /**
             * Adding a model data named message
             * with value from String called msg
             */
            String msg = "Hello World";
            mv.addObject("message", msg);
            return mv;
    }
}
```

10. Edit the Index.jsp page to initiate the call to HelloPage

```
<html>
<body>
<h2>Hello World!</h2>
<a href="Hello">Click here </a>
</body>
</html>
```

11. Create a new jsp page under **WEB-INF -> views** called HelloPage.jsp
    a. Use ${message} for presenting the data stored in message model value from ModelAndView object.

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!-- Since our web app version is 2.3 EL is be default ignored --
 --   Including the attribute isELIgnored to false : enables EL -->

<%@ page isELIgnored="false" %>

<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<h1>This is the message from Model :: ${message }</h1>
</body>
</html>
```



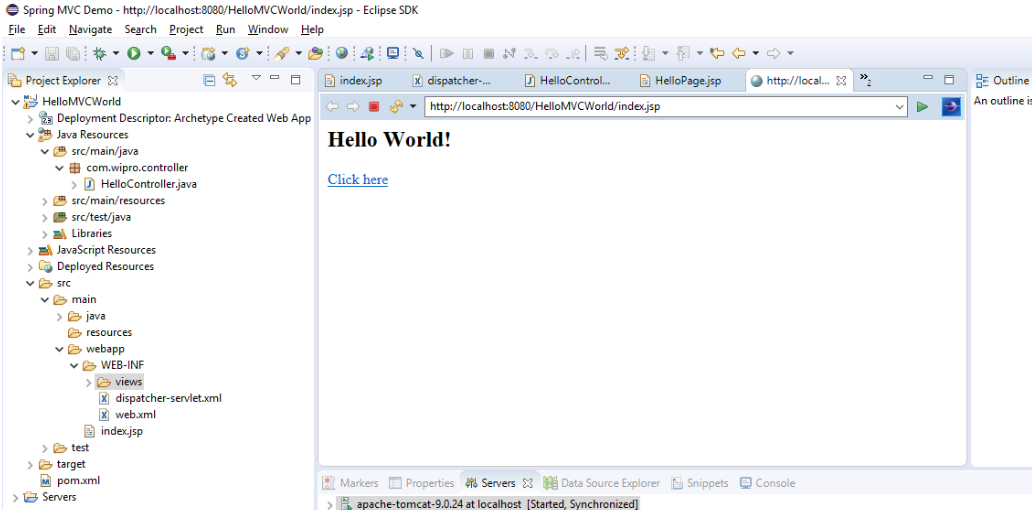
Spring MVC Demo - HelloMVCWorld/src/main/webapp/WEB-INF/views/HelloPage.jsp - Eclipse SDK

```
HelloMVCWorld
  Deployment Descriptor: Archetype Created Web App
  Java Resources
    src/main/java
      com.wipro.controller
        HelloController.java
    src/main/resources
    src/test/java
    Libraries
  JavaScript Resources
  Deployed Resources
  src
    main
      java
      resources
      webapp
        WEB-INF
          views
            HelloPage.jsp
          dispatcher-servlet.xml
          web.xml
        index.jsp
    test
  target
  pom.xml
Servers
```

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!-- Since our web app version is 2.3 EL is be default ignored --
 --   Including the attribute isELIgnored to false : enables EL -->

<%@ page isELIgnored="false" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<h1>This is the message from Model :: ${myMessage} </h1>
</body>
</html>
```

```
apache-tomcat-9.0.24 at localhost [Apache Tomcat] C:\Program Files\Oracle\OpenJDK 11.0.1\bin\javaw.exe (Mar 25, 2020, 11:27:09 AM)
INFO: Server startup in [5,316] milliseconds
Mar 25, 2020 11:27:20 AM org.apache.catalina.core.ApplicationContext log
INFO: Initializing Spring DispatcherServlet 'dispatcher'
Mar 25, 2020 11:27:20 AM org.springframework.web.servlet.FrameworkServlet initServletBean
INFO: Initializing Servlet 'dispatcher'
```

12. Run the Application from index.jsp Page

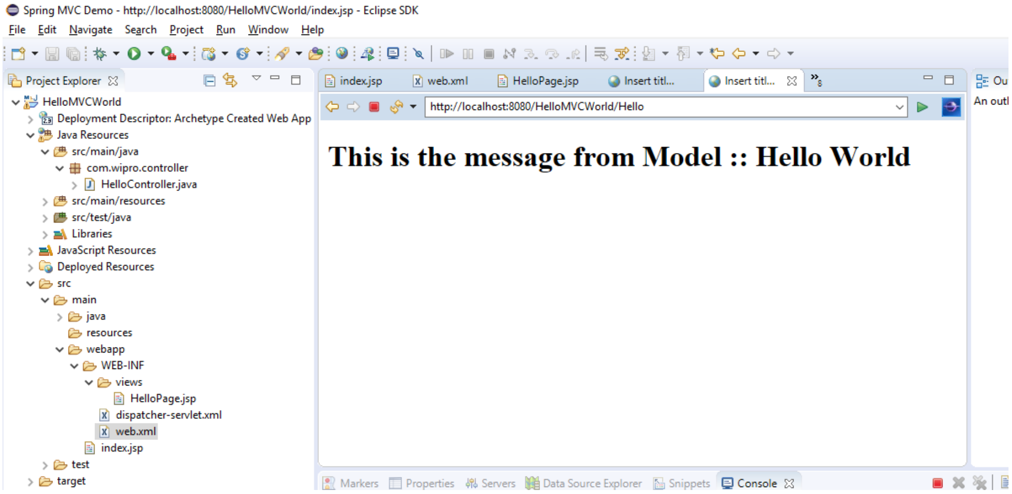


http://localhost:8080/HelloMVCWorld/index.jsp

Hello World!

Click here

apache-tomcat-9.0.24 at localhost [Started, Synchronized]



http://localhost:8080/HelloMVCWorld/Hello

This is the message from Model :: Hello World