Spring MVC - Simple Url Handler Mapping Example

Previous Page Next Page

The following example shows how to use Simple URL Handler Mapping using the Spring Web MVC framework. The SimpleUrlHandlerMapping class helps to explicitly-map URLs with their controllers respectively.

```
<beans>
   <bean class = "org.springframework.web.servlet.view.InternalResourceViewResolver">
      <property name = "prefix" value = "/WEB-INF/jsp/"/>
      <property name = "suffix" value = ".jsp"/>
   </bean>
      <property name = "mappings">
         <props>
            <prop key = "/welcome.htm">welcomeController</prop>
            <prop key = "/helloWorld.htm">helloController</prop>
         </props>
      </property>
   </bean>

   <bean id = "helloController" class = "com.tutorialspoint.HelloController" />

   <bean id = "welcomeController" class = "com.tutorialspoint.WelcomeController"/>
</beans>
```

For example, using above configuration, if URI

- /helloWorld.htm is requested, DispatcherServlet will forward the request to the **HelloController**.
- /welcome.htm is requested, DispatcherServlet will forward the request to the **WelcomeController**.

To start with, let us have a working Eclipse IDE in place and consider the following steps to develop a Dynamic Form based Web Application using the Spring Web Framework.

Step	Description

1	Create a project with a name **TestWeb** under a package com.tutorialspoint as explained in the Spring MVC - Hello World chapter.
2	Create Java classes HelloController and WelcomeController under the com.tutorialspoint package.
3	Create view files hello.jsp and welcome.jsp under the jsp sub-folder.
4	The final step is to create the content of the source and configuration files and export the application as explained below.

HelloController.java

```
package com.tutorialspoint;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.web.servlet.ModelAndView;
import org.springframework.web.servlet.mvc.AbstractController;

public class HelloController extends AbstractController{

   @Override
   protected ModelAndView handleRequestInternal(HttpServletRequest request,
      HttpServletResponse response) throws Exception {
      ModelAndView model = new ModelAndView("hello");
      model.addObject("message", "Hello World!");
      return model;
   }
}
```

WelcomeController.java

```
package com.tutorialspoint;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.web.servlet.ModelAndView;
import org.springframework.web.servlet.mvc.AbstractController;

public class WelcomeController extends AbstractController{

   @Override
   protected ModelAndView handleRequestInternal(HttpServletRequest request,
      HttpServletResponse response) throws Exception {
      ModelAndView model = new ModelAndView("welcome");
      model.addObject("message", "Welcome!");
      return model;
   }
}
```

TestWeb-servlet.xml

```
<beans xmlns = "http://www.springframework.org/schema/beans"
   xmlns:context = "http://www.springframework.org/schema/context"
```

Who is Who

```
   xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance
   xsi:schemaLocation = "
   http://www.springframework.org/schema/beans
   http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
   http://www.springframework.org/schema/context
   http://www.springframework.org/schema/context/spring-context-3.0.xsd">

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

   <bean class = "org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
      <property name = "mappings">
         <props>
            <prop key = "/welcome.htm">welcomeController</prop>
            <prop key = "/helloWorld.htm">helloController</prop>
         </props>
      </property>
   </bean>

   <bean id = "helloController" class = "com.tutorialspoint.HelloController" />

   <bean id = "welcomeController" class = "com.tutorialspoint.WelcomeController"/>
</beans>
```

hello.jsp

```
<%@ page contentType = "text/html; charset = UTF-8" %>
<html>
   <head>
      <title>Hello World</title>
   </head>
   <body>
      <h2>${message}</h2>
   </body>
</html>
```

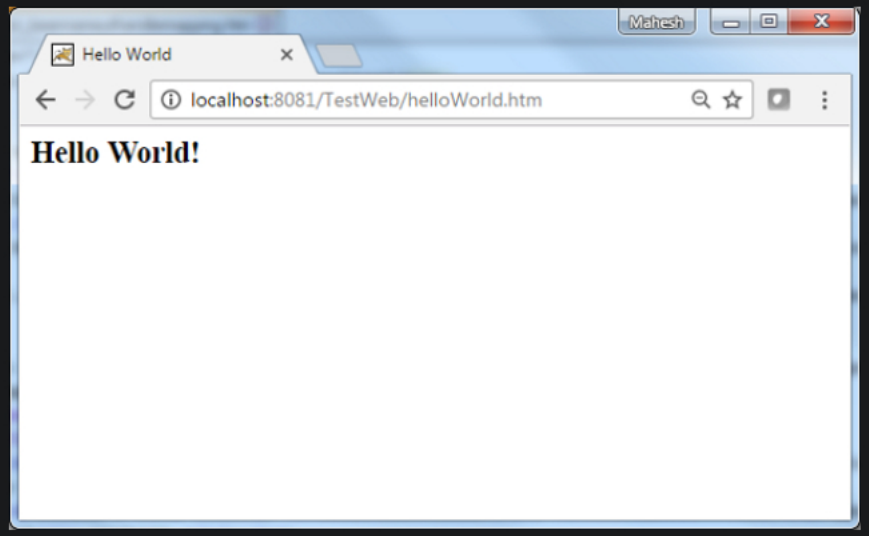
welcome.jsp

```
<%@ page contentType = "text/html; charset = UTF-8" %>
<html>
   <head>
      <title>Welcome</title>
   </head>
   <body>
      <h2>${message}</h2>
   </body>
</html>
```

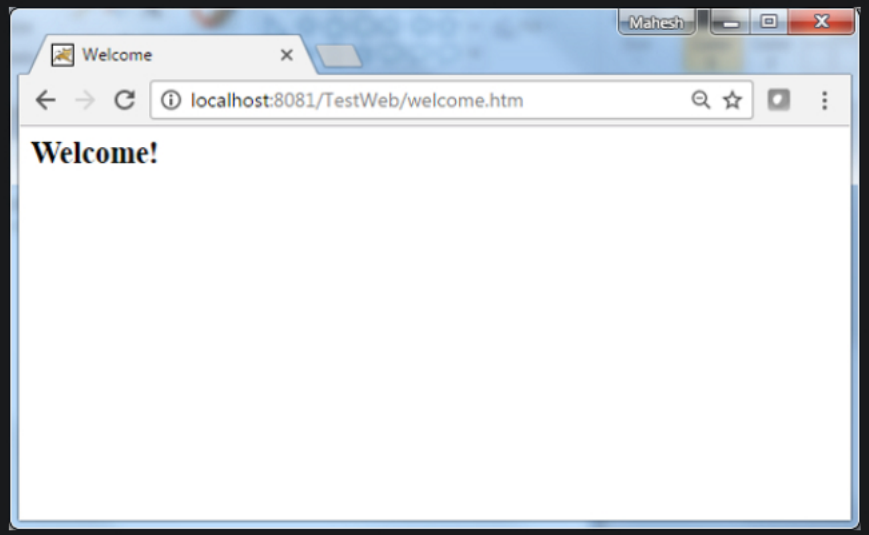
Once you are done with creating source and configuration files, export your application. Right click on your application, use the **Export → WAR File** option and save your **TestWeb.war** file in Tomcat's webapps folder.

Now, start your Tomcat server and make sure you are able to access other webpages from the webapps folder by using a standard browser. Try a URL – **http://localhost:8080/TestWeb/helloWorld.htm** and we will see the following screen, if

everything is fine with the Spring Web Application.



Try a URL **http://localhost:8080/TestWeb/welcome.htm** and you should see the following result if everything is fine with your Spring Web Application.



Previous Page Next Page