

Spring MVC

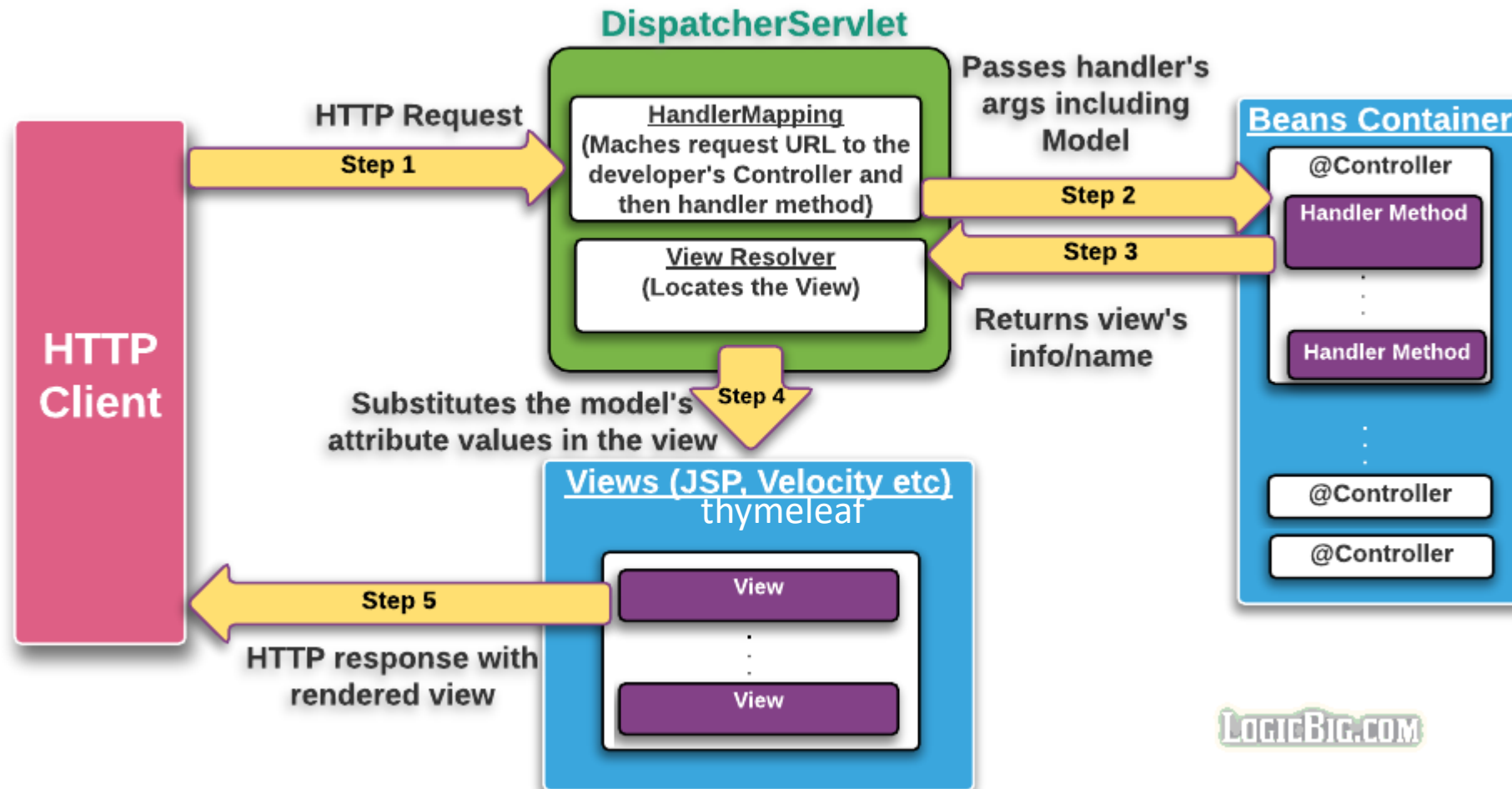
An intro to Spring MVC with template engine

A spring application

- Controllers
 - Handle the logic
 - Receive HTTP requests
 - Manipulate data/objects and store them (session, cookies, database...)
 - Inject data into a view
 - Returns the view (response)
- Views
 - Html pages
 - Augmented html: special tags to insert dynamic content (the data injected from the controllers)
- Model
 - The objects used to pass data to the view

Request / response

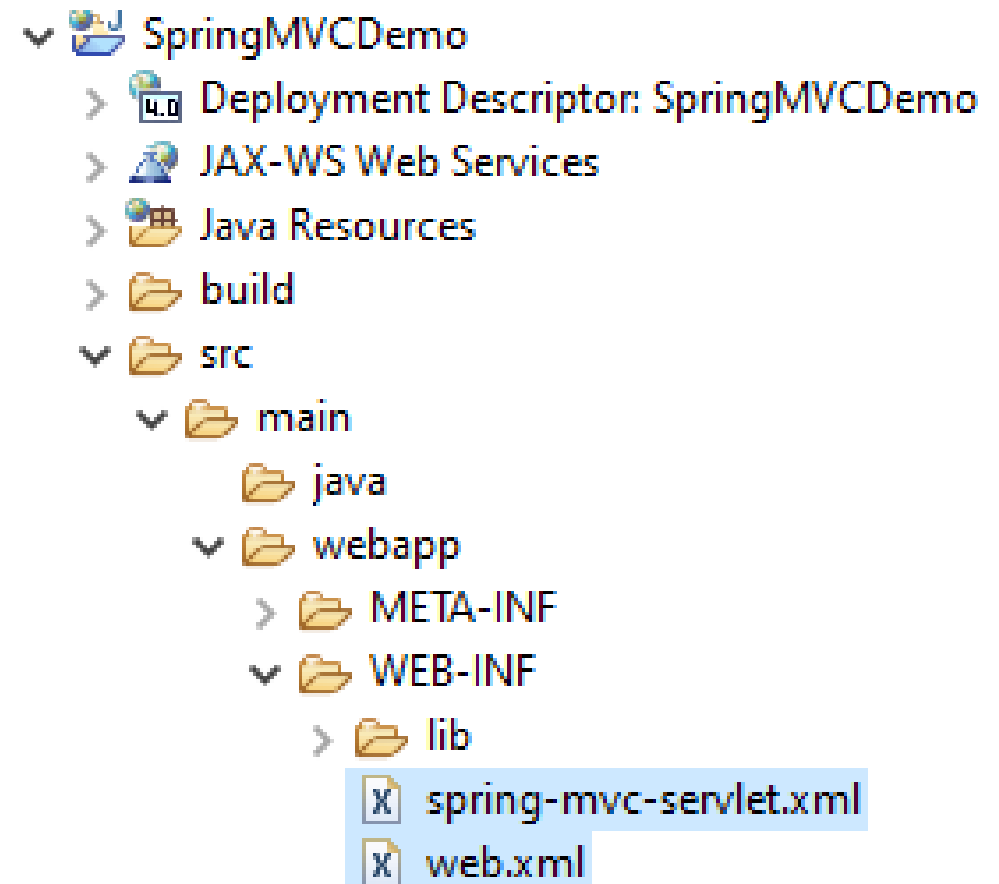
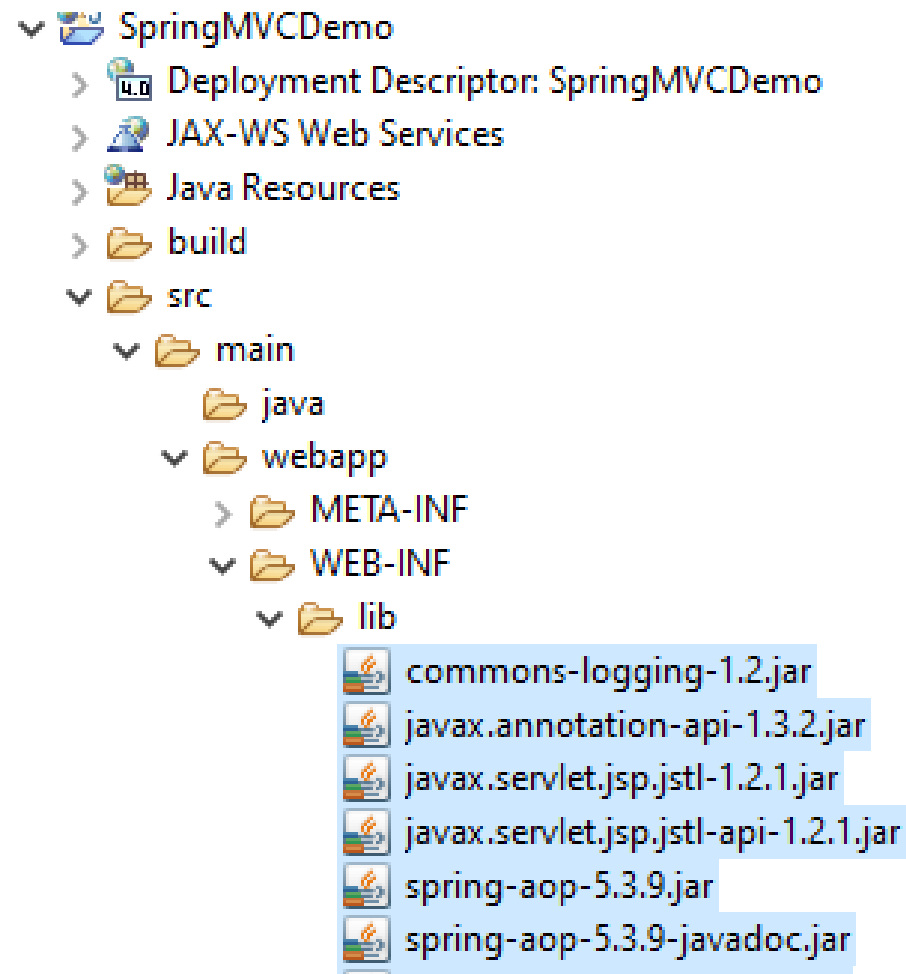
High level Spring MVC



Project Setup

- Create a Dynamic Web Project
- Add spring jars in WEB-INF/lib
- Add java-servlet-api jars also in WEB-INF/lib
 - Javax.servlet.jsp.jstl-1.2.1.jar
 - Javax.servlet.jsp.jstl-api-1.2.1.jar
- Add Configuration files in WEB-INF
 - spring-mvc-servlet.xml and web.xml

Project Setup with spring jars and config files



Project Setup :

Instead of adding jars, we can do it with maven as follows:

- Create a Dynamic Web Project
- Rclick Project->Configure->Convert To maven
- Include following dependencies
 - spring-webmvc
 - jakarta.servlet-api
 - javax.annotation-api
 - commons-logging
 - jakarta.servlet.jsp.jstl-api
- Add Configuration files in WEB-INF
 - spring-mvc-servlet.xml and web.xml

Alternative Maven
Project Setup

Maven dependencies

Alternative Maven
Project Setup

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-web</artifactId>
  <version>6.0.11</version>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-webmvc</artifactId>
  <version>6.0.11</version>
</dependency>
<dependency>
  <groupId>jakarta.servlet</groupId>
  <artifactId>jakarta.servlet-api</artifactId>
  <version>6.0.0</version>
  <scope>provided</scope>
</dependency>
```

```
<dependency>
  <groupId>javax.annotation</groupId>
  <artifactId>javax.annotation-api</artifactId>
  <version>1.3.2</version>
</dependency>
<dependency>
  <groupId>commons-logging</groupId>
  <artifactId>commons-logging</artifactId>
  <version>1.1.2</version>
</dependency>
<dependency>
  <groupId>jakarta.servlet.jsp.jstl</groupId>
  <artifactId>jakarta.servlet.jsp.jstl-api</artifactId>
  <version>2.0.0</version>
</dependency>
```

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4       xmlns:context="http://www.springframework.org/schema/context"
5       xmlns:mvc="http://www.springframework.org/schema/mvc"
6       xsi:schemaLocation="
7         http://www.springframework.org/schema/beans
8         http://www.springframework.org/schema/beans/spring-beans.xsd
9         http://www.springframework.org/schema/context
10        http://www.springframework.org/schema/context/spring-context.xsd
11        http://www.springframework.org/schema/mvc
12        http://www.springframework.org/schema/mvc/spring-mvc.xsd">
13
14     <!-- Step 3: Add support for component scanning -->
15     <context:component-scan base-package="com.rit.mvc" />
16
17     <!-- Step 4: Add support for conversion, formatting and validation support -->
18     <mvc:annotation-driven/>
19
20     <!-- Step 5: Define Spring MVC view resolver -->
21     <bean
22         class="org.springframework.web.servlet.view.InternalResourceViewResolver">
23         <property name="prefix" value="/WEB-INF/view/" />
24         <property name="suffix" value=".jsp" />
25     </bean>
```



```
5     id="WebApp_ID" version="4.0">
6
7     <display-name>SpringMVCDemo</display-name>
8     <absolute-ordering />
9
10    <!-- Step 1: Configure Spring MVC Dispatcher Servlet -->
11    <servlet>
12        <servlet-name>dispatcher</servlet-name>
13        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
14        <init-param>
15            <param-name>contextConfigLocation</param-name>
16            <param-value>/WEB-INF/spring-mvc-servlet.xml</param-value>
17        </init-param>
18        <load-on-startup>1</load-on-startup>
19    </servlet>
20
21    <!-- Step 2: Set up URL mapping for Spring MVC Dispatcher Servlet -->
22    <servlet-mapping>
23        <servlet-name>dispatcher</servlet-name>
24        <url-pattern>/</url-pattern>
25    </servlet-mapping>
26
27    <welcome-file-list>
28        <welcome-file>index.jsp</welcome-file>
29    </welcome-file-list>
30 </web-app>
```

Project Setup

- Create a package `com.rit.mvc` as defined in servlet config file

```
<context:component-scan base-package="com.rit.mvc" />
```

- We will create all controller, model & service classes here
- Create a folder **view** in WEB-INF folder as defined in servlet config file

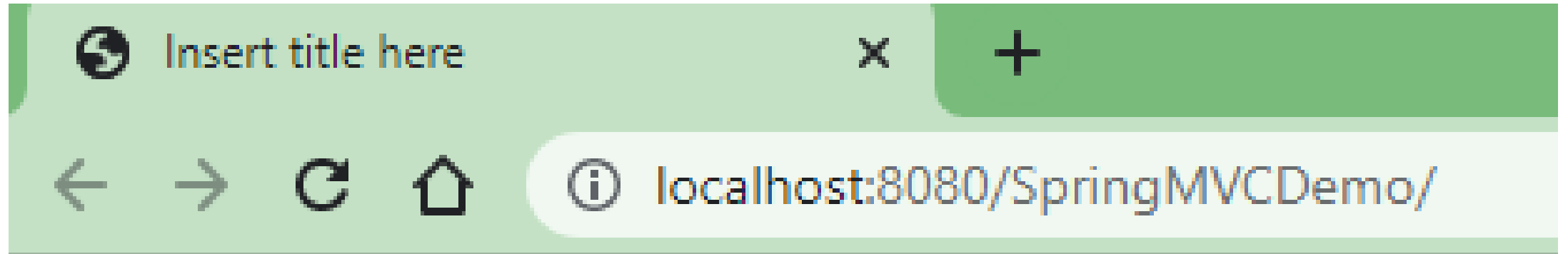
```
<bean  
    class="org.springframework.web.servlet.view.InternalResourceViewResolver">  
    <property name="prefix" value="/WEB-INF/view/" />  
    <property name="suffix" value=".jsp" />  
</bean>
```

- We will create all the jsp files within this folder

Package & View folder created

- SpringMVCDemo
 - Deployment Descriptor: SpringMVCDemo
 - JAX-WS Web Services
 - Java Resources
 - src/main/java
 - com.rit.mvc
 - Libraries
 - build
 - src
 - main
 - java
 - webapp
 - META-INF
 - WEB-INF
 - lib
 - view
- SpringMVCDemo
 - Deployment Descriptor: SpringMVCDemo
 - JAX-WS Web Services
 - Java Resources
 - src/main/java
 - com.rit.mvc
 - Libraries
 - build
 - src
 - main
 - java
 - webapp
 - META-INF
 - WEB-INF
 - lib
 - view

Lets Develop this

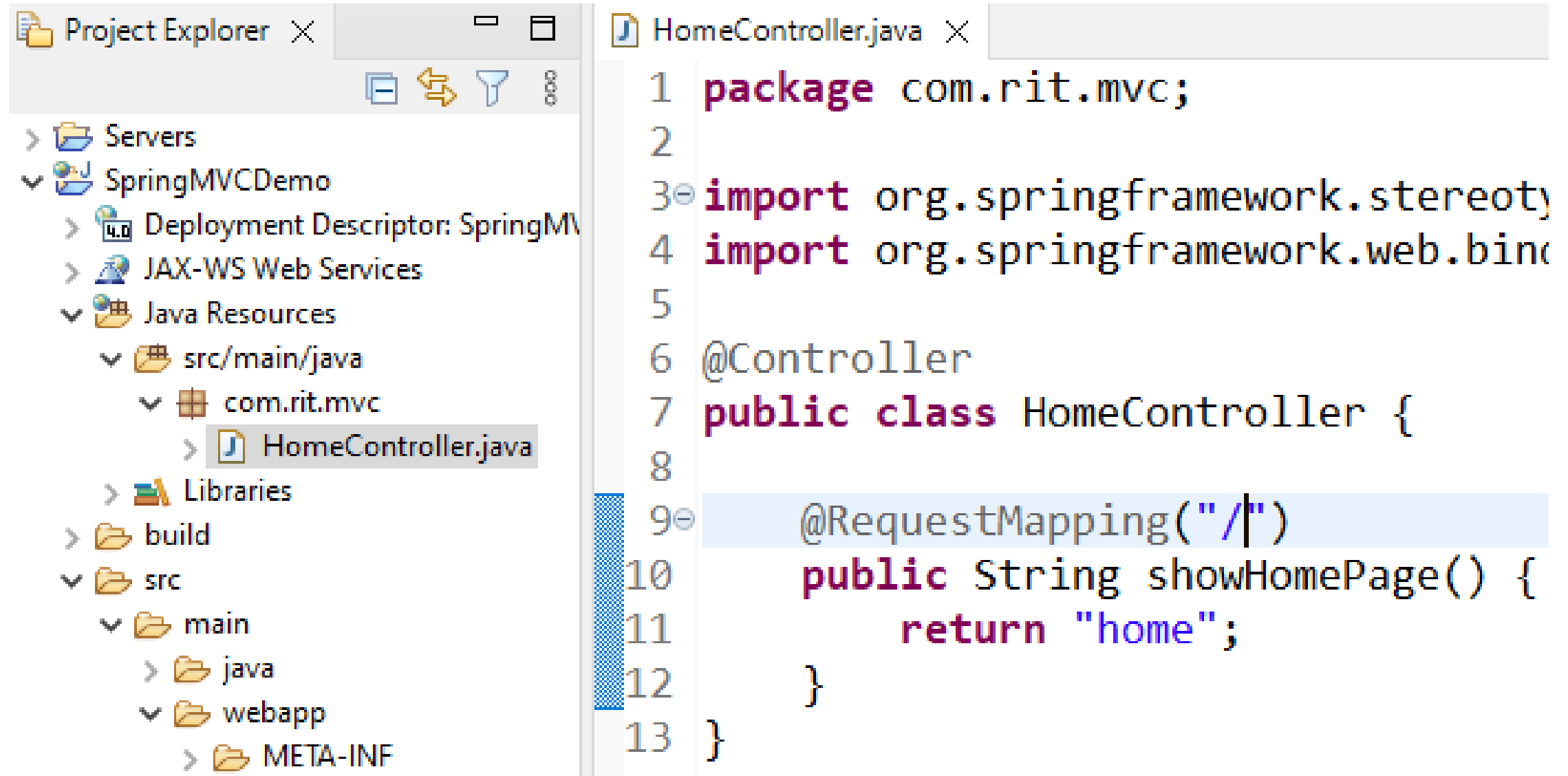


Welcome Home Page

Development Process : First Controller

- Create a Controller Class
- Create a Controller Method
- Add Request Mapping to the Controller Method
- Return View Name
- Develop View Page

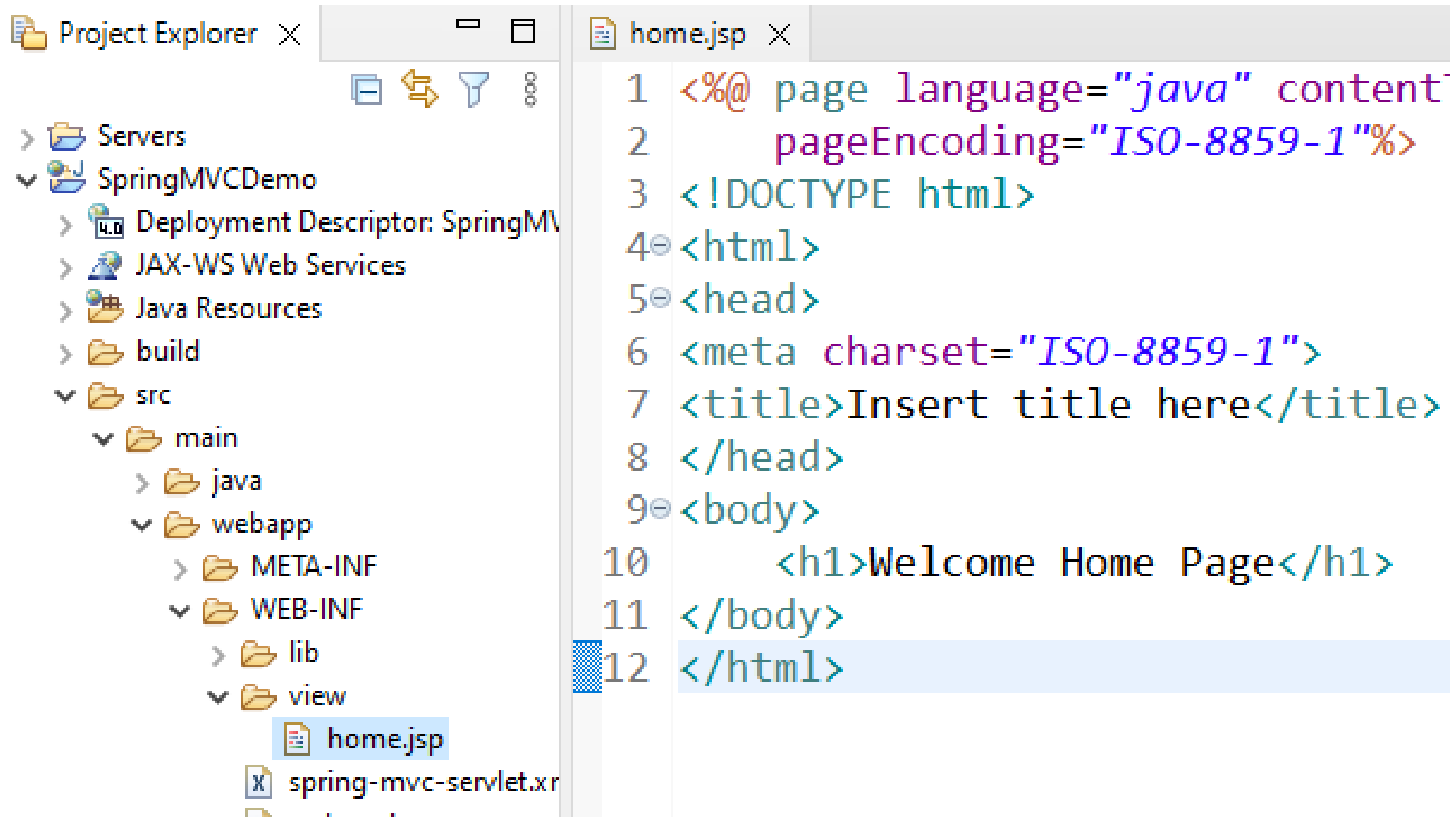
Controller Class



The image shows a screenshot of an IDE with two panels. The left panel is the Project Explorer, showing a project named 'SpringMVCDemo'. Under 'Java Resources', there is a package 'com.rit.mvc' containing a file 'HomeController.java'. The right panel is the code editor, displaying the source code of 'HomeController.java'.

```
1 package com.rit.mvc;
2
3 import org.springframework.stereotype.*;
4 import org.springframework.web.bind.annotation.*;
5
6 @Controller
7 public class HomeController {
8
9     @RequestMapping("/")
10    public String showHomePage() {
11        return "home";
12    }
13 }
```

View Page



The image shows a screenshot of an IDE interface. On the left is the 'Project Explorer' panel, and on the right is the 'home.jsp' file editor.

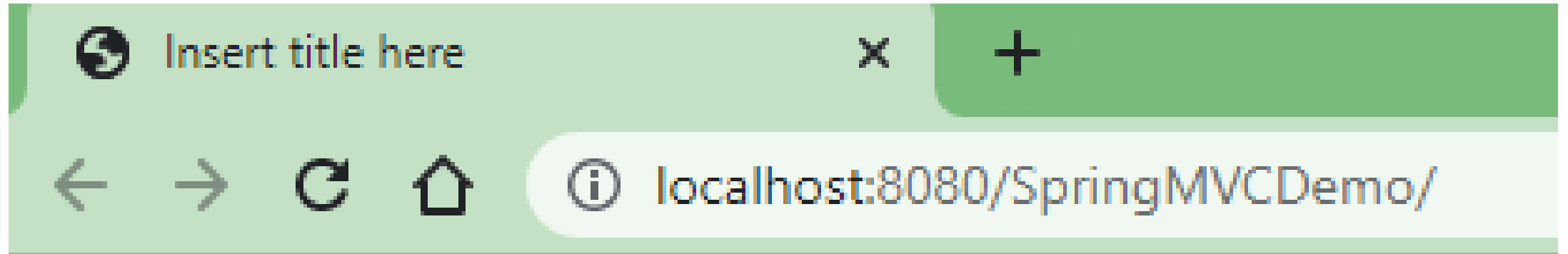
Project Explorer:

- Servers
- SpringMVCDemo
 - Deployment Descriptor: SpringMV
 - JAX-WS Web Services
 - Java Resources
 - build
 - src
 - main
 - java
 - webapp
 - META-INF
 - WEB-INF
 - lib
 - view
 - home.jsp
 - spring-mvc-servlet.xr

home.jsp File Editor:

```
1 <%@ page language="java" content
2     pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="ISO-8859-1">
7 <title>Insert title here</title>
8 </head>
9 <body>
10     <h1>Welcome Home Page</h1>
11 </body>
12 </html>
```

Project -> Run as Server



Welcome Home Page

Controller Level Request Mapping

Lets develop it with controller level ReqMap...



← → ↻ 🏠 ⓘ localhost:8080/SpringMVCDemo/stud

Welcome to Student Home Page



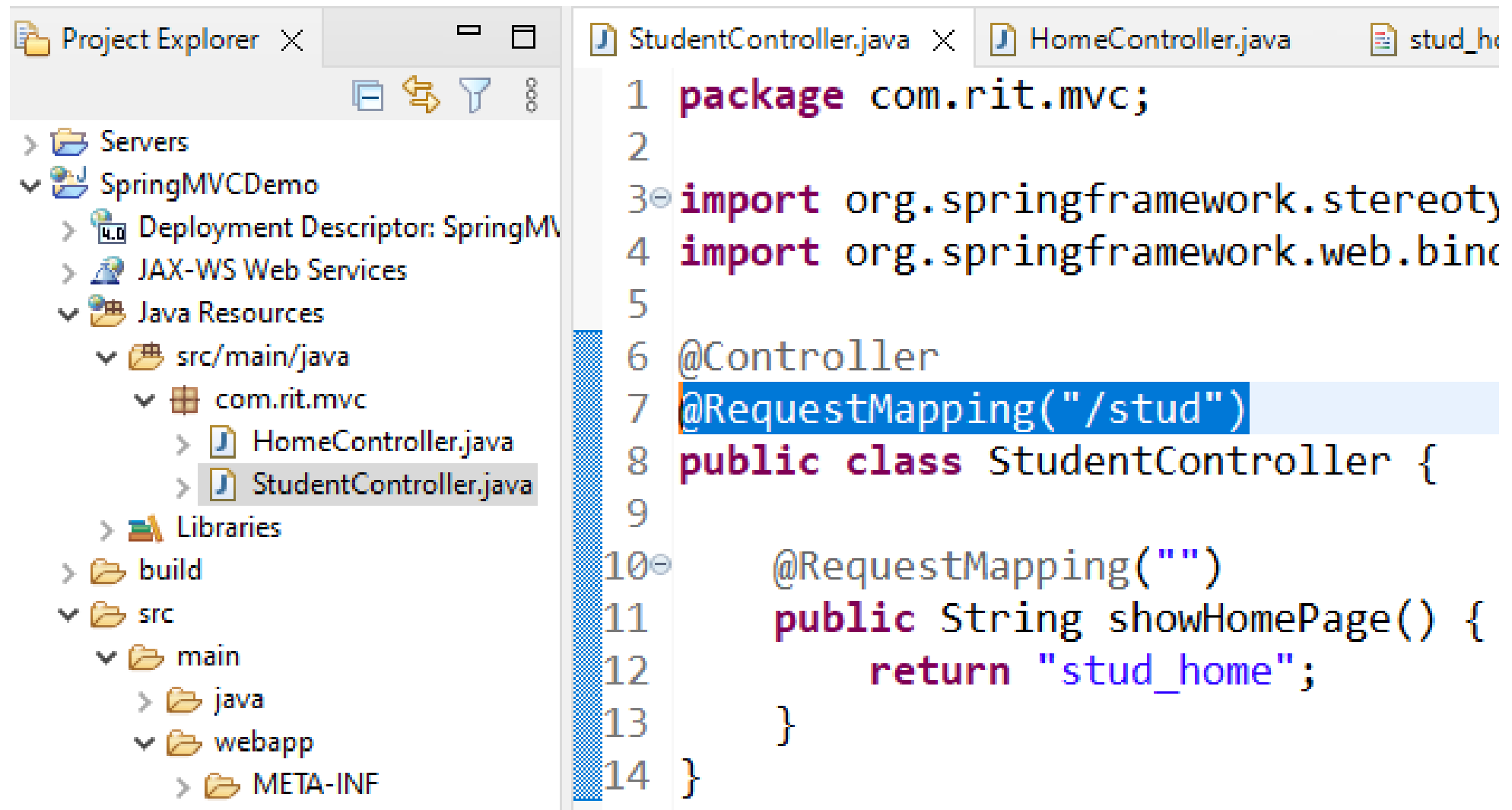
← → ↻ 🏠 ⓘ localhost:8080/SpringMVCDemo/stud/

Welcome to Student Home Page

Development Process : First Controller

- Create a Controller Class
- Add Request Mapping to the Controller Class
- Create a Controller Method
- Add Request Mapping to the Controller Method
- Return View Name
- Develop View Page

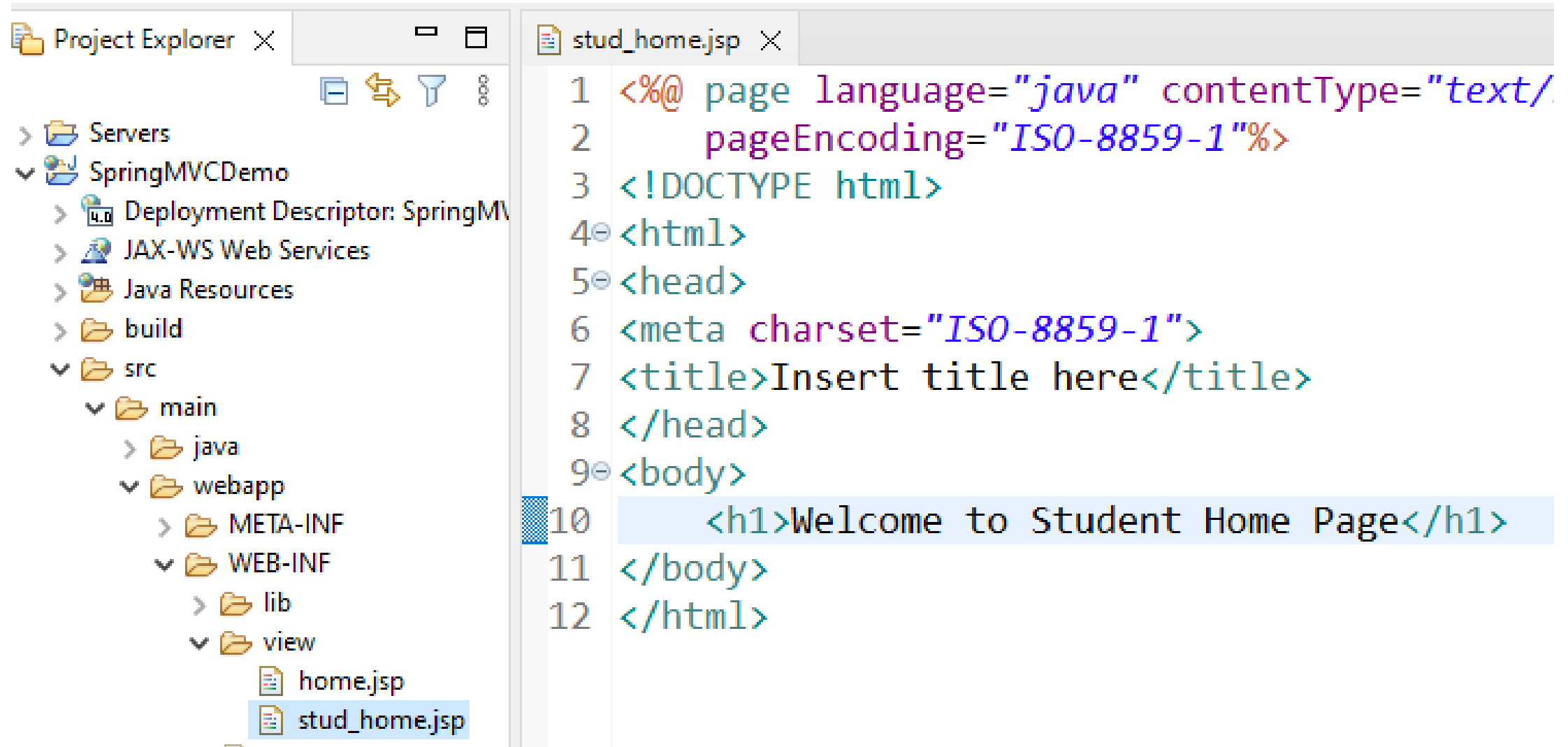
Controller Class



The screenshot displays an IDE interface with a Project Explorer on the left and a code editor on the right. The Project Explorer shows a project named 'SpringMVCDemo' with a package structure 'com.rit.mvc' containing 'HomeController.java' and 'StudentController.java'. The code editor shows the 'StudentController.java' file with the following code:

```
1 package com.rit.mvc;
2
3 import org.springframework.stereotype.*;
4 import org.springframework.web.bind.annotation.*;
5
6 @Controller
7 @RequestMapping("/stud")
8 public class StudentController {
9
10     @RequestMapping("")
11     public String showHomePage() {
12         return "stud_home";
13     }
14 }
```

Stud view



The image shows a screenshot of an IDE interface. On the left is the 'Project Explorer' panel, and on the right is the 'stud_home.jsp' file editor.

Project Explorer:

- Servers
- SpringMVCDemo
 - Deployment Descriptor: SpringMV
 - JAX-WS Web Services
 - Java Resources
 - build
 - src
 - main
 - java
 - webapp
 - META-INF
 - WEB-INF
 - lib
 - view
 - home.jsp
 - stud_home.jsp (selected)

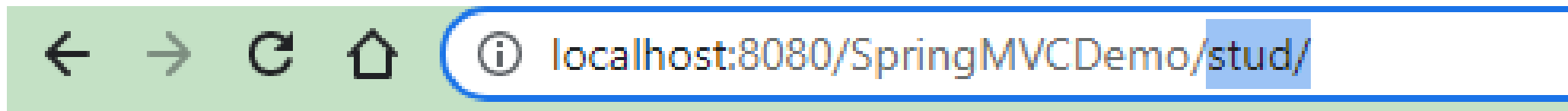
stud_home.jsp:

```
1 <%@ page language="java" contentType="text/  
2     pageEncoding="ISO-8859-1"%>  
3 <!DOCTYPE html>  
4 <html>  
5 <head>  
6 <meta charset="ISO-8859-1">  
7 <title>Insert title here</title>  
8 </head>  
9 <body>  
10 <h1>Welcome to Student Home Page</h1>  
11 </body>  
12 </html>
```

Output



Welcome to Student Home Page



Welcome to Student Home Page

Reading HTML Form data

student/save?roll=101&firstName=Anand&mark1=70&mark2=80&mark3=90

← → ↻ ⓘ localhost:8080/Spring4MVC/student/insert

Add Student Details

Roll :

FirstName :

Mark1 :

Mark2 :

Mark3 :

View Student

Roll : 101

FirstName : Anand

Mark1 : 70

Mark2 : 80

Mark3 : 90

Development Process : First Controller

- Create a Controller Class
- Create Two Controller Methods
 - Insert()
 - Save()
- Develop View Pages
- Access the form data using `${param.<name>}` in jsp

Controller

```
@Controller
@RequestMapping("/student")
public class StudentController {

    @RequestMapping("/insert")
    public String insert() {
        return "student-insert";
    }

    @RequestMapping("/save")
    public String save() {
        return "student-view";
    }
}
```

Student-insert.jsp

```
student-insert.jsp ×
8 </head>
9 <body>
10     <h1> Add Student Details</h1>
11     <form action="save" method="get">
12         Roll : <input type="text" name="roll"/> <br><br>
13         FirstName : <input type="text" name="firstName"/> <br><br>
14         Mark1 : <input type="text" name="mark1"/> <br><br>
15         Mark2 : <input type="text" name="mark2"/> <br><br>
16         Mark3 : <input type="text" name="mark3"/> <br><br>
17         <input type="submit" value="Insert"/>
18     </form>
19 </body>
20 </html>
```

Student-view.jsp

```
student-view.jsp X
7 <title>Insert title here</title>
8 </head>
9 <body>
10 <div style="padding-left:450px">
11 <h1> View Student </h1>
12 Roll : ${param.roll} <br><br>
13 FirstName : ${param.firstName} <br><br>
14 Mark1 : ${param.mark1} <br><br>
15 Mark2 : ${param.mark2} <br><br>
16 Mark3 : ${param.mark3} <br><br>
17 </div>
18 </body>
19 </html>
```

HttpRequest & Model

In Header -> Get data from VIEW FORM and set data to VIEW PAGE

ident/save?roll=102&firstName=Gayathri&mark1=80&mark2=85&mark3=90

← → ↻ ⓘ localhost:8080/Spring4MVC/student/insert

Add Student Details

Roll :

FirstName :

Mark1 :

Mark2 :

Mark3 :

View Student

Roll : 102

FirstName : Gayathri

Mark1 : 80

Mark2 : 85

Mark3 : 90

Total : 255

Average : 85.0

Result : Pass

```
17 // @RequestMapping( /save )
18 // public String save() {
19 //     return "student-view";
20 // }
21 @RequestMapping("/save")
22 public String save(HttpServletRequest req, Model model) {
23     int m1 = Integer.parseInt(req.getParameter("mark1"));
24     int m2 = Integer.parseInt(req.getParameter("mark2"));
25     int m3 = Integer.parseInt(req.getParameter("mark3"));
26
27     int tot = m1 + m2 + m3;
28     double avg = tot / 3.0;
29     String res = (m1>=35 && m2>=35 && m3>=35) ? "Pass" : "Fail";
30
31     model.addAttribute("total",tot);
32     model.addAttribute("average",avg);
33     model.addAttribute("result",res);
34
35     return "student-view";
36 }
```

StudentController.java

student-view.jsp

```
7 </title></head></html>
8 </head>
9 <body>
10 <div style="padding-left:450px">
11 <h1> View Student </h1>
12 Roll : ${param.roll} <br><br>
13 FirstName : ${param.firstName} <br><br>
14 Mark1 : ${param.mark1} <br><br>
15 Mark2 : ${param.mark2} <br><br>
16 Mark3 : ${param.mark3} <br><br>
17 <hr/>
18 Total : ${total} <br><br>
19 Average : ${average} <br><br>
20 Result : ${result} <br><br>
21 </div>
22 </body>
23 </html>
24
```


RequestParam

Instead of HttpServletRequest

@RequestParam instead of HttpServletRequest

```
StudentController.java × student-view.jsp
2 // public String save(HttpServletRequest req, Model model) {
3 //     int m1 = Integer.parseInt(req.getParameter("mark1"));
4 //     int m2 = Integer.parseInt(req.getParameter("mark2"));
5 //     int m3 = Integer.parseInt(req.getParameter("mark3"));
6
7 @RequestMapping("/save")
8 public String save(@RequestParam("mark1") int m1, @RequestParam("mark2")
9                   int m2, @RequestParam("mark3") int m3, Model model ) {
10     int tot = m1 + m2 + m3;
11     double avg = tot / 3.0;
12     String res = (m1>=35 && m2>=35 && m3>=35) ? "Pass" : "Fail";
13
14     model.addAttribute("total",tot);
15     model.addAttribute("average",avg);
16     model.addAttribute("result",res);
17
18     return "student-view";
19 }
```

@RequestParam

- When we use @RequestParam we must pass the data for the param
- By default the required attribute is “true”
- We can set it false to make the param optional
- We can also set default value for the param



localhost:8080/Spring4MVC/student/save?roll=103&firstName=Siva&mark1=&mark2=70&mark3=60

HTTP Status 400 – Bad Request

Type Status Report

Description The server cannot process the request because of a client error (e.g., malformed request syntax, timing that is perceived to be a client error (

Apache Tomcat/10.1.28

Add Student Details

Roll :

FirstName :

Mark1 :

Mark2 :

Mark3 :

required & defaultValue attributes

```
@RequestMapping("/save")
public String save(
    @RequestParam(name="mark1", required=false, defaultValue="0") int m1,
    @RequestParam("mark2") int m2,
    @RequestParam("mark3") int m3, Model model ) {
    int tot = m1 + m2 + m3;
    double avg = tot / 3.0;
    String res = (m1>=35 && m2>=35 && m3>=35) ? "Pass" : "Fail";

    model.addAttribute("total",tot);
    model.addAttribute("average",avg);
    model.addAttribute("result",res);

    return "student-view";
}
```

e?roll=103&firstName=Siva&mark1=&mark2=70&mark3=60

Add Student Details

Roll :

FirstName :

Mark1 :

Mark2 :

Mark3 :

View Student

Roll : 103

FirstName : Siva

Mark1 :

Mark2 : 70

Mark3 : 60

Total : 130

Average : 43.33333333333333336

Result : Fail

Spring MVC Form Tags

Spring MVC Form Tags

- Spring MVC Form Tags are the building blocks for a web page
- Form Tags are configurable and reusable for a web page
- Spring MVC Form Tags can make use of databinding
- Automatically Setting/Retrieving data from a java object/bean

Form Tags

- Form Tags will generate HTML for us.

Form Tag	Description
form:form	main form container
form:input	text field
form:textarea	multi-line text field
form:checkbox	check box
form:radiobutton	radio buttons
form:select	drop down list
<i>more</i>	

Development Process

- Create a bean class (Employee)
- Create a Controller (EmpController)
 - Create a form method with model attribute
 - Create a process method with ModelAttribute
- Create a Form Jsp with formtags and path
- Create a Process Jsp

```
1 package com.rit.mvc;
2
3 public class Employee {
4
5     public String firstname;
6     public String lastname;
7
8     public Employee() {
9     }
10    public String getFirstname() {
11        return firstname;
12    }
13    public void setFirstname(String firstname) {
14        this.firstname = firstname;
15    }
16    public String getLastname() {
17        return lastname;
18    }
19    public void setLastname(String lastname) {
20        this.lastname = lastname;
21    }
22 }
```

```
1 package com.rit.mvc;
2
3+ import org.springframework.stereotype.Controller;
4
5
6
7
8 @Controller
9 @RequestMapping("/emp")
10 public class EmployeeController {
11
12     @RequestMapping("/signup")
13     public String showSignupForm(Model model) {
14         Employee emp = new Employee();
15         model.addAttribute("emp", emp);
16         return "emp_signup";
17     }
18
19     @RequestMapping("/signupprocess")
20     public String processSignupForm(@ModelAttribute("emp") Employee employee) {
21         return "emp_signup_resp";
22     }
23 }
```

Employee.java

EmployeeController.java

emp_signup.jsp

emp_signup_resp.jsp

```
1 <%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>
2 <!DOCTYPE html>
3 <html>
4 <body>
5
6     <form:form action="signupprocess" modelAttribute="emp">
7         FirstName : <form:input path="firstname" /> <br><br>
8         lastName : <form:input path="lastname" /> <br><br>
9         <input type="submit" />
10    </form:form>
11
12 </body>
13 </html>
```

Employee.java

EmployeeController.java

emp_signup.jsp

*emp_signup_resp.jsp X

```
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2     pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html>
4 <html>
5 <body>
6     <h1>Employee Details</h1>
7     <hr>
8     First Name : ${emp.firstname} <br><br>
9     Last Name : ${emp.lastname} <br><br>
10 </body>
11 </html>
```

← → ↻ 🏠 ⓘ localhost:8080/SpringMVCDemo/emp/signup

FirstName :

lastName :

← → ↻ 🏠 ⓘ localhost:8080/SpringMVCDemo/emp/signupprocess

Employee Details

First Name : Anand

Last Name : Krishnan

MVC Form Tag - Select

Development Process

- Add a property in Employee (bean) class
- Generate Getter and Setter
- Add the dropdown element
 <form:select..> tag in emp_signup.jsp
- Get the value in the response page
 \${emp.-----}

```
private String qual;

public String getQual() {
    return qual;
}

public void setQual(String qual) {
    this.qual = qual;
}
```

Qualification:

```
Qualification:
<form:select path="qual">
    <form:option value="BCA" label="BCA" />
    <form:option value="BSc" label="BSc" />
    <form:option value="BE" label="BE" />
    <form:option value="Others" label="Others" />
</form:select><br><br>
```

Qualification : \${emp.qual}

Qualification : BSc

MVC Form Tag - Radio

```
private String gender;

public String getGender() {
    return gender;
}
public void setGender(String gender) {
    this.gender = gender;
}
```

Gender :

```
<form:radio button path="gender" value="Male"/> Male
<form:radio button path="gender" value="Female"/> Female
```

Gender : \${emp.gender}

Gender : ☐ Male ☒ Female

Submit

Gender : Female

@GetMapping & @PostMapping

```
1 package com.rit.mvc;
2 import org.springframework.stereotype.Controller;
3
4
5
6 @Controller
7 @RequestMapping("/user")
8 public class UserController {
9
10     @RequestMapping(value="/login", method=RequestMethod.GET)
11     public String userLogin() {
12         return "login";
13     }
14
15     @RequestMapping(value="/login", method=RequestMethod.POST)
16     public String userLoginPrc() {
17         return "login_success";
18     }
19 }
```

UserController.java

login.jsp

login_success.jsp

```
1 <%@ page language="java" contentType="text/html; charset=ISO-8859-1
2     pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html>
4 <html>
5 <body>
6     <h1>User Login</h1>
7     <hr>
8     <form action="login" method="post">
9         Username : <input type="text" name="uname" /><br><br>
10        Password : <input type="password" name="upass" /><br><br>
11
12        <input type="submit" value="Login" />
13    </form>
14 </body>
15 </html>
```


UserController.java login.jsp login_success.jsp ×

```
1 <%@ page language="java" contentType="text/html" %>
2     pageEncoding="ISO-8859-1"%>
3 <!DOCTYPE html>
4 <html>
5 <body>
6     <h1>login Success</h1>
7     <hr>
8     Welcome ${param.uname}
9     </form>
10 </body>
11 </html>
```

← → ↻ 🏠 ⓘ localhost:8080/SpringMVCDemo/user/login

User Login

Username :

Password :

Login

← → ↻ 🏠 ⓘ localhost:8080/SpringMVCDemo/user/login

login Success

Welcome Anand

```
1 package com.rit.mvc;
2 import org.springframework.stereotype.Controller;
6
7 @Controller
8 @RequestMapping("/user")
9 public class UserController {
10
11     //@RequestMapping(value="/login", method=RequestMethod.GET)
12     @GetMapping("/login")
13     public String userLogin() {
14         return "login";
15     }
16
17     //@RequestMapping(value="/login", method=RequestMethod.POST)
18     @PostMapping("/login")
19     public String userLoginPrc() {
20         return "login_success";
21     }
22 }
```

Thank you