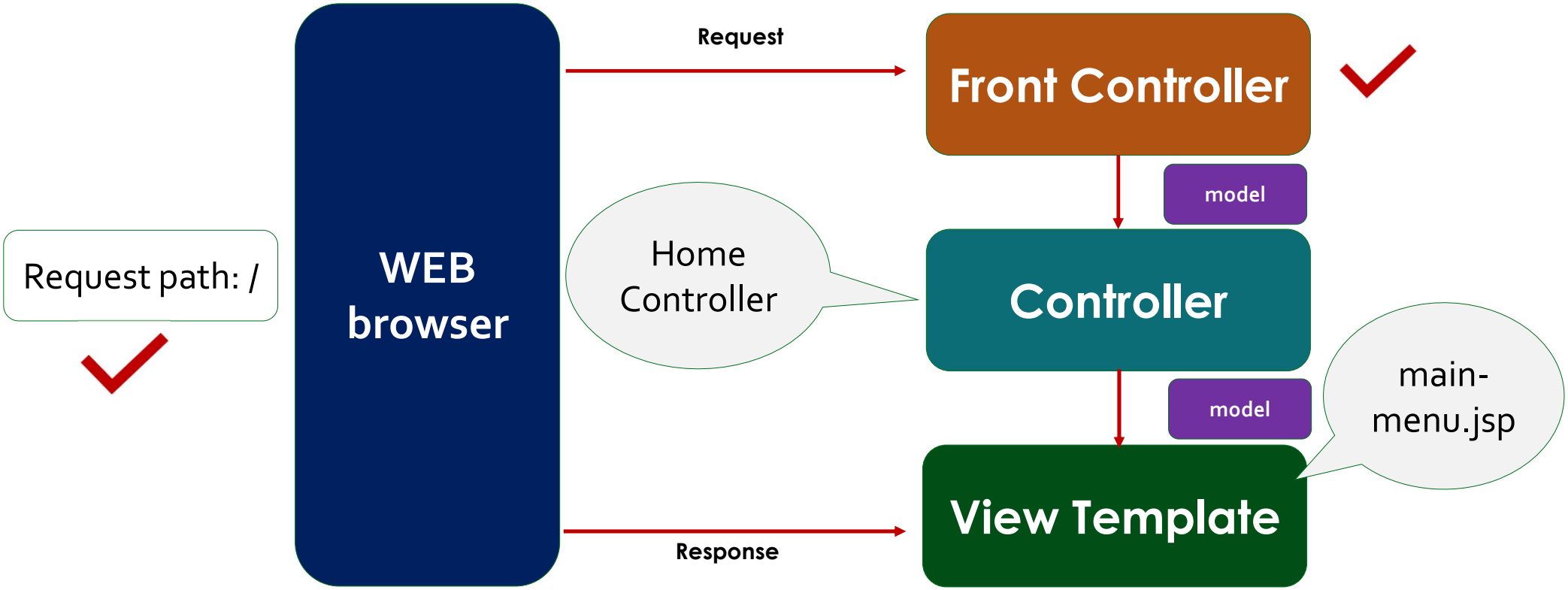
Bộ môn: Kỹ Thuật Phần Mềm

Lập trình WWW *Java*

.

Spring MVC – Controllers and Views

First Spring MVC Example



Development Process

- Create **Controller class**
- Define **Controller method**
- Add **Request mapping** to **Controller method**
- Return the **View Name**
- Develop **View Page**

Step 1: Create Controller class

- **Annotate class** with **@Controller** (@Controller inherits from @Component and supports scanning)

File: HomeController.java (project: spring-mvc-demo)

```
package com.se.springmvc.demo;
import org.springframework.stereotype.Controller;
@Controller
public class HomeController {
}
```

Step 2: Define Controller method

File: HomeController.java

```
@Controller
public class HomeController {
    public String showMyPage(){
        return "main-menu";
    }
}
```

Step 3: Add Request mapping to Controller method

File: HomeController.java

```
@Controller
public class HomeController {
    @RequestMapping("/")
    public String showMyPage(){
        return "main-menu";
    }
}
```

Step 4: Return the View Name

File: HomeController.java

```
@Controller
public class HomeController {
    @RequestMapping("/")
    public String showMyPage() {
        return "main-menu";
    }
}
```

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

Finding a view page

/WEB-INF/view/main-menu.jsp

Step 5: Develop View Page

File: WEB-INF/view/main-menu.jsp

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <h1>Spring MVC Demo - Home page</h1>
    </body>
</html>
```

Reading Form Data

Hight Level View

helloworld-form.jsp

What your name?

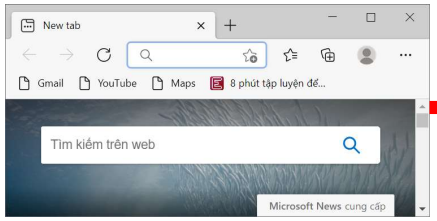
Submit Query

helloworld.jsp

Hello world of Spring!

Student Name: John Doe

Application Flow



Request mapping

/showform

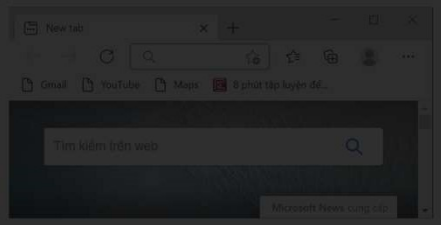
HelloWorld Controller

helloworld-form.jsp

What your name?

Submit Query

Application Flow



Request mapping

/showForm

HelloWorld Controller

helloworld-form.jsp

What your name? Submit Query

HelloWorld Controller that's going to have two request mappings.

helloworld-form.jsp

What your name? Submit Query

Request mapping

/processForm

HelloWorld Controller

helloworld.jsp

Hello world of Spring!

Student Name: John Doe

Development Process

- Create **Controller class**
- Show **HTML Form**
  - Create **controller** method to **show HTML form**
  - Create **view page** for HTML form
- Process **HTML Form**
  - Create **controller** method to **show HTML form**
  - Develop **View Page for Confirmation**

Controller Class

File: HelloWorldController.java

```
@Controller
public class HelloWorldController {
    @RequestMapping("/showForm")
    public String showForm() {
            return "helloworld-form";}
    @RequestMapping("/processForm")
    public String processForm() {
            return "helloworld";}
}
```

Show HTML Form
(helloworld-form.jsp)

Process **HTML Form**
(helloworld.jsp)

Create view page for HTML form

File: helloworld-form.jsp

```
<html>
<head>
<meta charset="ISO-8859-1">
<title>Hello World - Input Form</title>
</head>
<body>
    <form action="processForm" method ="post">
            <input type= "text" name="studentName"
            placeholder="What's your name? "/>
            <input type="submit" />
    </form>
</body>
</html>
```

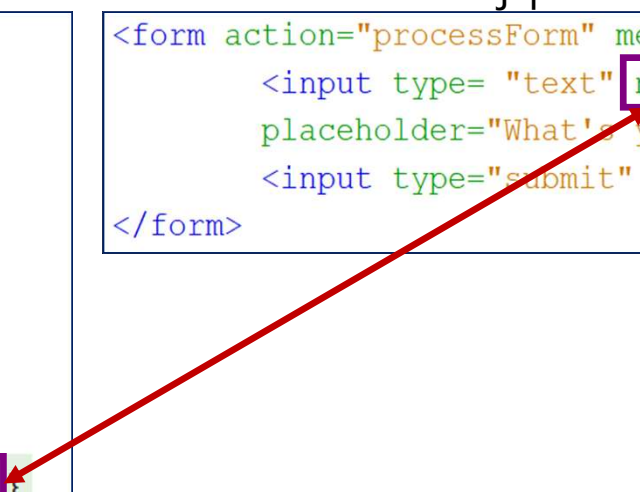
Develop View Page for Confirmation

File: helloworld.jsp

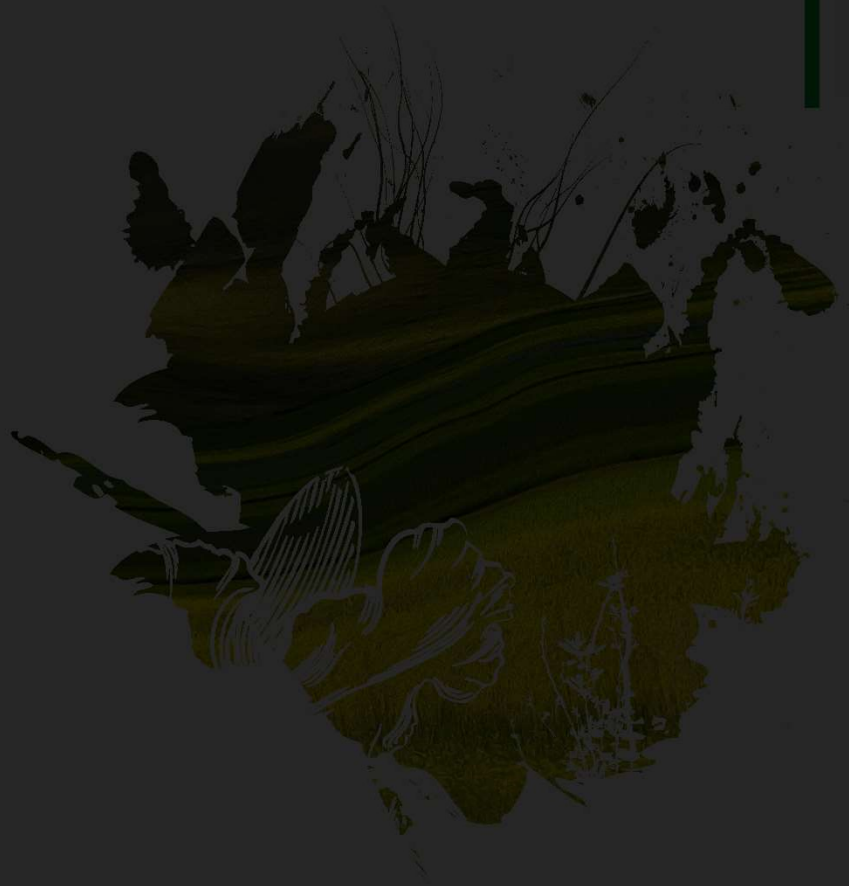
```
<html>
    <head>
        <meta charset="ISO-8859-1">
    <title>Hello World of Spring</title>
    </head>
    <body>
        Hello World of Spring!
        <br><br>
        Student name : ${param.studentName}
   </body>
</html>
```

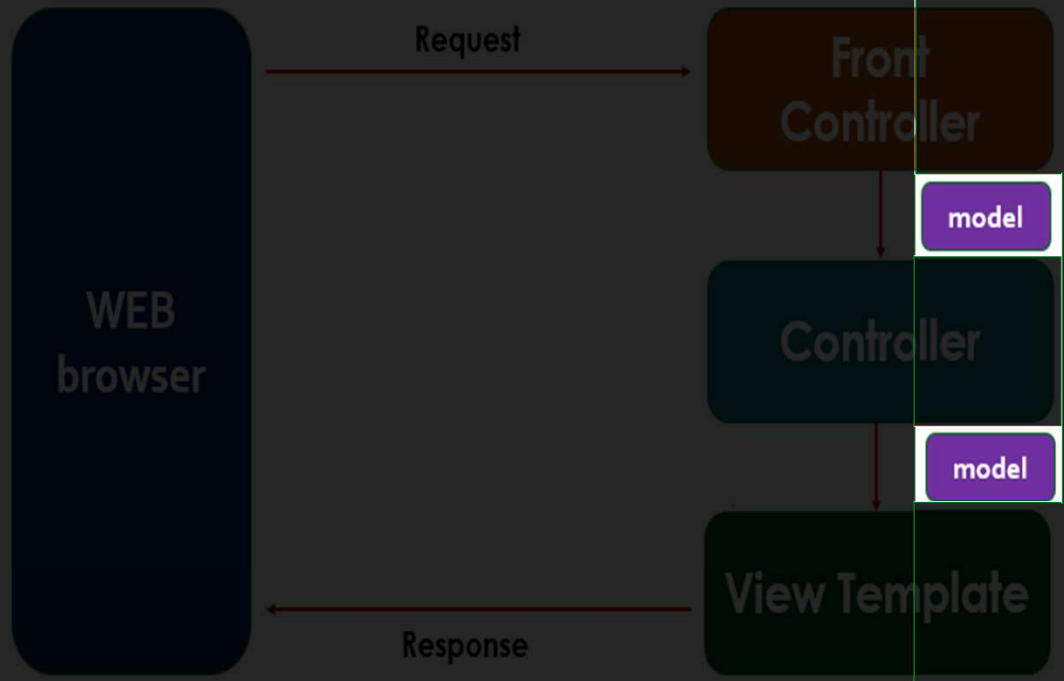
File: helloworld-form.jsp

```
<form action="processForm" method ="post">
        <input type= "text" name="studentName"
        placeholder="What's your name? "/>
        <input type="submit" />
</form>
```



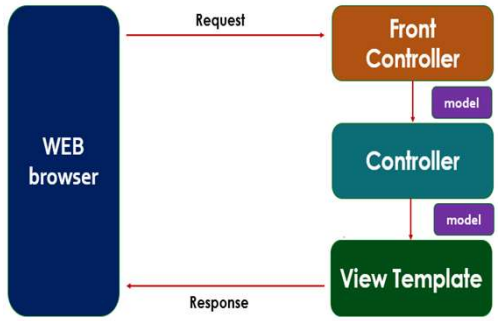
Adding Data to Spring Model





Spring Model

- The **Model** is a **container** for **your application data**
- **In controller:** You can put **any things** to **model: strings, objects, info from the database, etc…**
- **View page (JSP)** can **access data** from **model**



Code Example

- We want to create a new method to process form data
- Read the form data: Student's name
- Convert the name to upper case
- Add the upper-case version to model

Passing Model to Controller

model is a container that can hold your form data

File: HelloWorldController.java

```
@RequestMapping("processFormVersionTwo")
public String letShoutDude(HttpServletRequest request, Model model) {
    //read the request parameter form html form
    String theName = request.getParameter("studentName");
    //convert data to all caps
    theName= pUpperCase();
    //create
    String res o! " + theName;
    //add message to Model
    model.addAttribute("message", result);
    //return the name f view (jsp page)
    return "helloworld"
```

if you need to read form data in your controller →you pass in the HttpServletRequest

Name of Attribute

Name of HTML form field

Value of Attribute

add something to the model→model.addAttribute

View Template - JSP

File: helloworld.jsp

```
<html>
    <head>
        <meta charset="ISO-8859-1">
    <title>Hello World of Spring</title>
    </head>
    <body>
        Hello World of Spring!
        <br><br>
        Student name : ${param.studentName}
        <br><br>
        The message: ${message}
    </body>
</html>
```

Get Attribute value of model

View Template - JSP

File: helloworld-form.jsp

```
<form action="processFormVersionTwo" method ="post">
        <input type= "text" name="studentName"
        placeholder="What's your name? "/>
        <input type="submit" />
</form>
```

Read HTML Form data with @RequestParam

File: HelloWorldController.java

```
@RequestMapping("processFormVersionThree")
public String processFormVersionThree(
    @RequestParam("studentName") String theName, Model model) {
    theName= theName.toUpperCase();
    //create the message
    String result = "Hey My Friend from " + theName;
    //add message to MOdel
    model.addAttribute("message", result);
    //return the name of view (jsp page)
    return "helloworld";}
```

Spring will read the request parameter, and bind it to variable theName

```
@RequestMapping("processFormVersionTwo")
public String letShoutDude(HttpServletRequest request, Model model) {
    //read the request parameter form html form
    String theName = request.getParameter("studentName");
```

Add Controller RequestMapping

- Serves as **parent mapping** for **controller**
- **All request mappings** on the **methods** in the controller are **relative**
- Similar to **directory structure**
- The **way** to **group your request mappings** together
- A **technique** for **resolving any problems or conflicts** with **other request mappings** (in other controller).

Add Controller RequestMapping

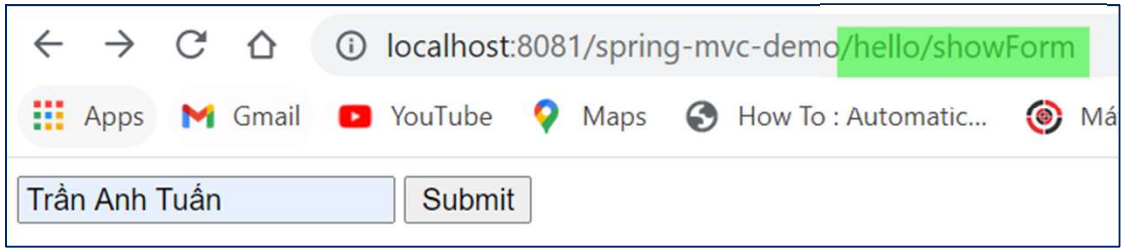
File: HelloWorldController.java

```
@Controller
@RequestMapping("hello")
public class HelloWorldController {
    @RequestMapping("/showForm")
    public String showForm() {
            return "helloworld-form";}
    @RequestMapping("/processForm")
    public String processForm() {
            return "helloworld";}
}
```

File: main-menu.jsp

```
<body>
    <h1>Spring MVC Demo - Home page</h1>
    <a href="hello/showForm">Plain Hello Wor
</body>
```

Web Browser



File: HelloWorldController.java

```
<form action="processForm" method ="post">
        <input type= "text" name ="studentName"
        placeholder="What's your name? "/>
        <input type="submit" />
</form>
```

How does "processForm" work for "/hello"?

Spring MVC Form Tags

Spring MVC Form Tags

- Spring has **support** for **form tags – The Building block for a web page**
- **Form Tags** are **configurable**, and **reusable** for a **webpage**
- Can **make use of Data binding → automatically set data** and **retrieve data** from **Java objects** and **beans**

Web Page Structure

» JSP with **special Spring MVC Form tags**

<html>

- Regular HTML
- Spring MVC form tags
- more HTML

</html>

Reference to Spring MVC Form Tags

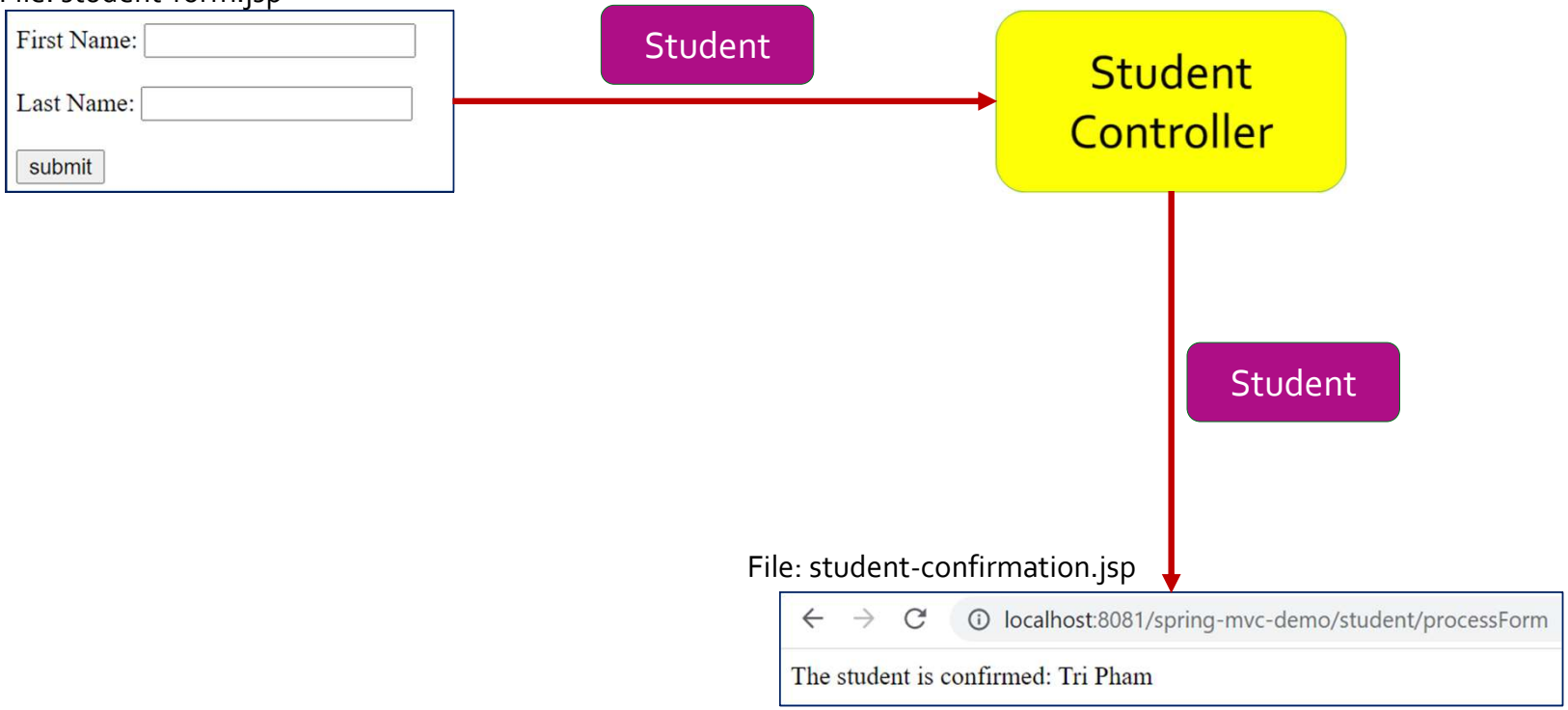
- Specify the **Spring namespace** at the **beginning** of **JSP file**

```
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>
<!DOCTYPE html>
<html>
```

Spring MVC Form Tags – Text Field

Code Example

File: student-form.jsp



Development Process

- Create Student class
- Create **Student Controller** class
- Create **HTML form**
- Create **form processing code**
- Create **confirmation page**

Step 1: Create Student class

File: Student.java

```
public class Student {
    private String firstName;
    private String lastName;
    public String getFirstName() {
        return firstName;     }
    public void setFirstName(String firstName)
        this.firstName = firstName;   }
    public String getLastName() {
        return lastName;   }
    public void setLastName(String lastName) {
        this.lastName = lastName; }
}
```

Step 2: Create Student Controller class

File: StudentController.java

```
@Controller
@RequestMapping("/student")
public class StudentController
    @RequestMapping("/showForm")
    public String showForm(Model theModel) {
        //Create the Student
        Student theStudent = new Student();
        //add student to the Model
        theModel.addAttribute("student", theStudent);
        return "student-form";}
    @RequestMapping("/processForm")
    public String processForm(@ModelAttribute("student") Student theStudent) {
        System.out.println("theStudent: " + theStudent.getFirstName()
        + " " + theStudent.getLastName());
        return "student-confirmation"; }
}
```

Model is used to pass data between controllers and views

File: student-form.jsp

```
<form:form action="processForm" modelAttribute="student" >
    First Name: <form:input path="firstName"/>
    <br><br>
    Last Name: <form:input path="lastName"/>
    <br><br>
   <input type="submit" value="submit"/>
</form:form>
```

Step 3: Create HTML form - Databiding

File: student-form.jsp

```
<form:form action="processForm" modelAttribute="student" >
    First Name: <form:input path="firstName"/>
    <br><br>
    Last Name: <form:input path="lastName"/>
    <br><br>
    <input type="submit" value="Submit"/>
</form:form>
```

When form is loaded, Spring MVC will call **student.getFirstName()**

When form is submitted, Spring MVC will call **student.setFirstName()** **and use whatever data the user entered there in the form field**

Step 4: Create form processing code

File: StudentController.java

```
@Controller
@RequestMapping("/student")
public class StudentController {
    @RequestMapping("/showForm")
    public String showForm(Model theModel) {
        //Create the Student
        Student theStudent = new Student();
        //add student to the Model
        theModel.addAttribute("student", theStudent);
        return "student-form";}
    @RequestMapping("/processForm")
    public String processForm(@ModelAttribute("student") Student theStudent) {
        System.out.println("theStudent: " + theStudent.getFirstName()
        + " " + theStudent.getLastName());
        return "student-confirmation"; }
}
```

Step 5: Create confirmation page

File: student-confirmation.jsp

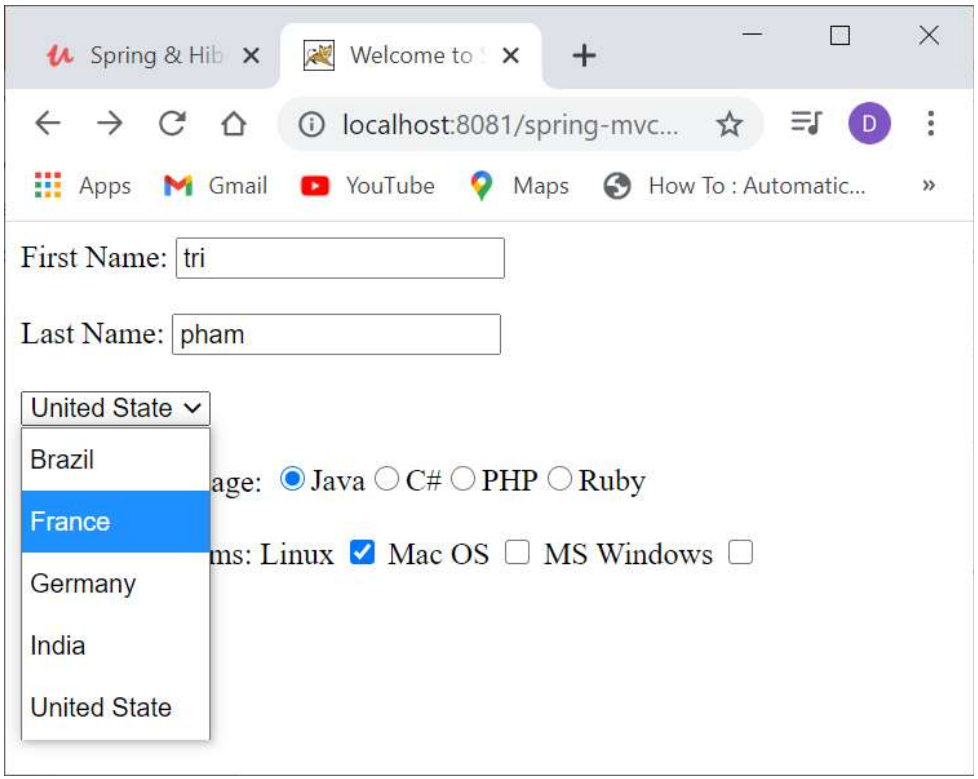
```
<body>
The student is confirmed: ${student.firstName} ${student.lastName}
<br><br>
```

File: StudentController.java

```
@RequestMapping("/processForm")
public String processForm(@ModelAttribute("student") Student theStudent) {
    System.out.println("theStudent: " + theStudent.getFirstName()
    + " " + theStudent.getLastName());
    return "student-confirmation"; }
```

Spring MVC Form Tags – Drop Down List

Review HTML <select> Tag




```
<select name="country">
    <option>Brazil</option>
    <option>France</option>
    <option>Germany</option>
   <option>India</option>
...
</select>
```

Spring MVC Tag

Drop Down List is presented by the tag:

<form:select>

Spring MVC Tag – Code Example

File: Student.java

```
private LinkedHashMap<String, String> countryOptions;
public Student() {
    //populate country options: use ISO country code
    countryOptions = new LinkedHashMap<String, String>();
    countryOptions.put("BR", "Brazil");
    countryOptions.put("FR", "France");
    countryOptions.put("DE", "Germany");
    countryOptions.put("IN", "India");
    countryOptions.put("US", "United State");
```

File:StudentController.java

```
@Controller
@RequestMapping("/student")
public class StudentController {
    @RequestMapping("/showForm")
    public String showForm(Model theModel) {
        //Create the Student
        Student theStudent = new Student();
        //add student to the Model
        theModel.addAttribute("student", theStudent);
        return "student-form";}
```

File: student-form.jsp

```
<form:select path="country">
    <form:options items="${student.countryOptions}" />
</form:select>
```

or

File: student-form.jsp

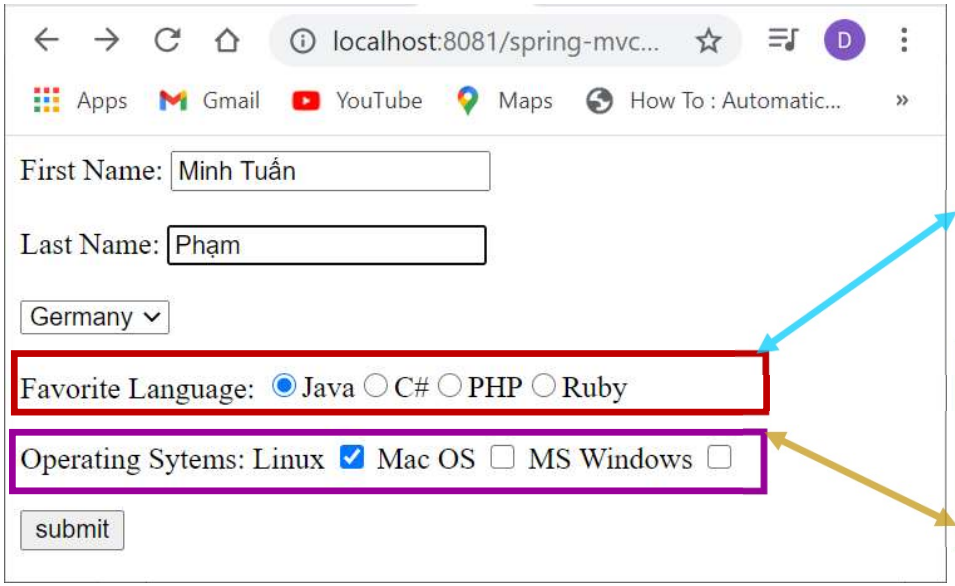
```
<form:select path="country">
    <form:option value="Brazil" label="Brazil"/>
    <form:option value="France" label="Frane"/>
    <form:option value="Germany" label="Germany"/>
    <form:option value="India" label="India"/>
</form:select>
```

File: studentconfirmation.jsp

```
<body>
    The student is confirmed:
    ${student.firstName} ${student.lastName}
    <br><br>
    Country: ${student.country}
    <br><br>
```

Spring MVC Form Tags – Check Box and Radio Button

Spring MVC Tags



A Radio Button is presented by the tag

<form:radiobuttons>

A Check Box is presented by the tag

<form:checkbox>

Spring MVC Tag – Check Box and Radio Button - Code Example

File: Student.java

```
public class Student {
 private String firstName;
 private String lastName;
 private String country;
 private String favoriteLanguage;
 private String[] operatingSystems;

 private LinkedHashMap<String, String> countryOptions;
 private LinkedHashMap<String, String> favoriteLanguageOptions;
 private ArrayList<String> operatingSystemOptions;
```

```
public Student() {
     // populate favorite language options
    favoriteLanguageOptions = new LinkedHashMap<>();
    // parameter order: value, display label
    favoriteLanguageOptions.put("Java", "Java");
    favoriteLanguageOptions.put("C#", "C#");
    favoriteLanguageOptions.put("PHP", "PHP");
    favoriteLanguageOptions.put("Ruby", "Ruby");

    operatingSystemOptions= new ArrayList<>();
    operatingSystemOptions.add("Linux");
    operatingSystemOptions.add("Mac OS");
    operatingSystemOptions.add("MS Windows");
```

File:StudentController.java

```
@RequestMapping("/showForm")
public String showForm(Model theModel) {
  Student theStudent = new Student();
  theModel.addAttribute("student", theStudent);
  return "student-form";}
```

Spring MVC Tag – Check Box and Radio Button - Code Example

File: student-form.jsp

```
Operating Systems:
Linux <form:checkbox
    path="operatingSystems" value="Linux" />
Mac OS <form:checkbox
    path="operatingSystems" value="Mac OS" />
MS Windows <form:checkbox
    path="operatingSystems" value="MS Windows" />
```

```
Or
```

```
<form:checkboxes path="operatingSystems"
                 items="${student.operatingSystemOptions}" />
```

```
Favorite Language:
Java <form:radiobutton path="favoriteLanguage" value="Java"/>
C# <form:radiobutton path="favoriteLanguage" value="CSharp"/>
PHP <form:radiobutton path="favoriteLanguage" value="PHP"/>
Python <form:radiobutton path="favoriteLanguage" value="Python"/>
Ruby <form:radiobutton path="favoriteLanguage" value="Ruby"/>
```

```
Or
```

```
<form:radiobuttons path="favoriteLanguage" items="${student.favoriteLanguageOptions}"
```

[items] is mandatory

Spring MVC Tag – Check Box and Radio Button - Code Example

File:StudentController.java

```
@RequestMapping("/processForm")
public String processForm(@ModelAttribute("student") Student theStudent)
    return "student-confirmation"; }
```

File: studentconfirmation.jsp

```
<body>
    The student is confirmed:
    ${student.firstName} ${student.lastName}
    <br><br>
    Country: ${student.country}
    <br><br>
    Favorite Language: ${student.favoriteLanguage}
    <br><br>
    Operating Systems:
    <ul>
        <c:forEach var="temp" items= "${student.operatingSystems}">
            <li> ${temp}  </li>
        </c:forEach>
    </ul>
</body>
```

Question?