What you will learn in Topic 2

Azzeddine
RIGAT

Topic 2: Introduction to Spring MVC

- Spring MVC - Building Spring Web Apps
- Spring MVC - Creating Controllers and Views
- Spring MVC - Request Params and Request Mappings
- Spring MVC - Form Tags and Data Binding
- Spring MVC Form Validation - Applying Built-In Validation Rules
- Spring MVC Form Validation - Validating Number Ranges and Regular Expressions
- Spring MVC Form Validation - Creating Custom Validation Rules

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

What is Spring MVC? ?

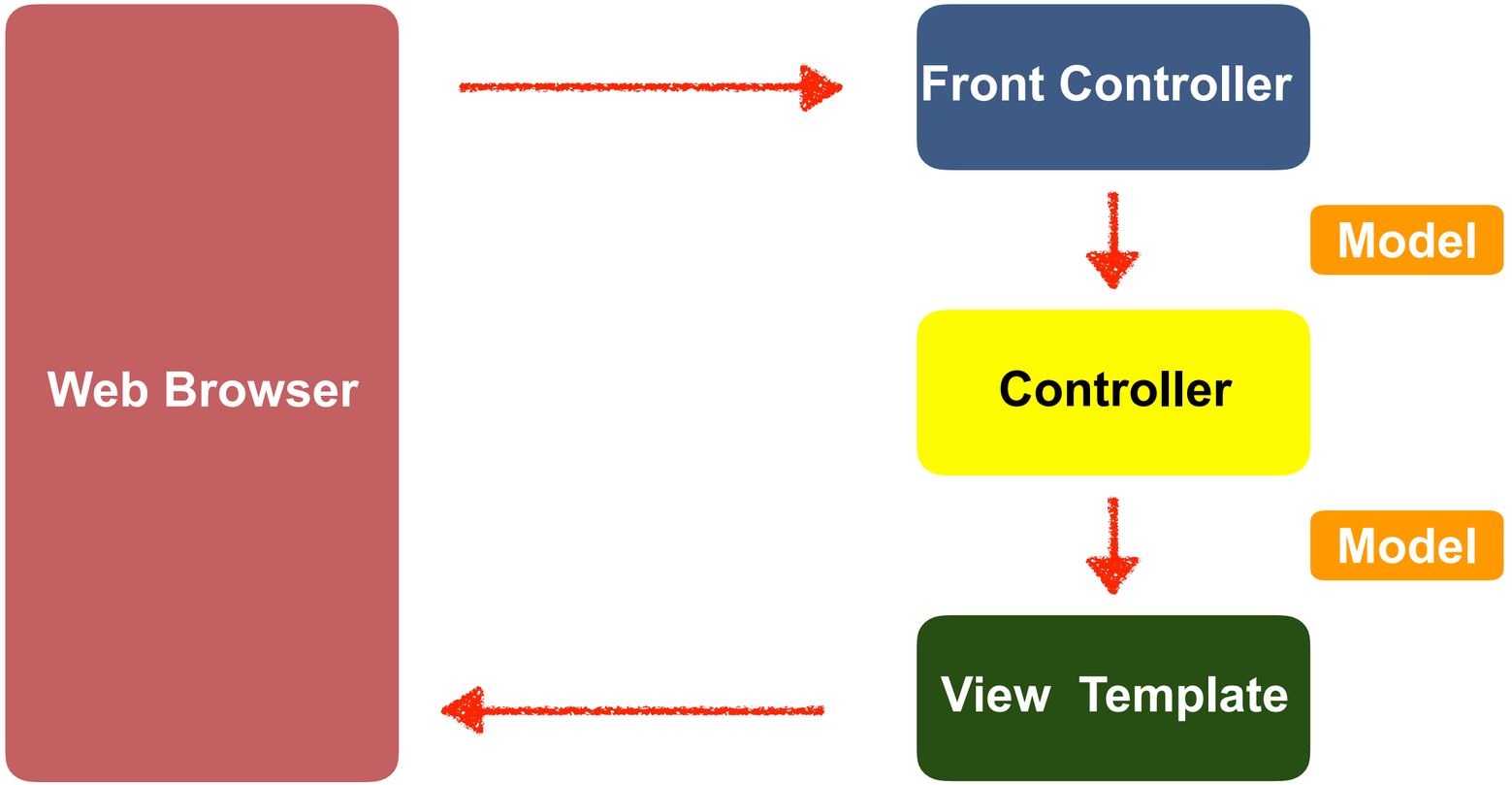
Spring MVC in a Nutshell

- Framework for building web applications in Java
- Based on Model-View-Controller design pattern
- Leverages features of the Core Spring Framework (IoC, DI)

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

Model-View-Controller (MVC)



Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

Spring MVC Benefits

- The Spring way of building web app UIs in Java
- Leverage a set of reusable UI components
- Help manage application state for web requests
- Process form data: validation, conversion etc
- Flexible configuration for the view layer

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

Spring MVC Documentation

https://docs.spring.io/spring/docs/current/spring-framework-reference/web.html#mvc

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

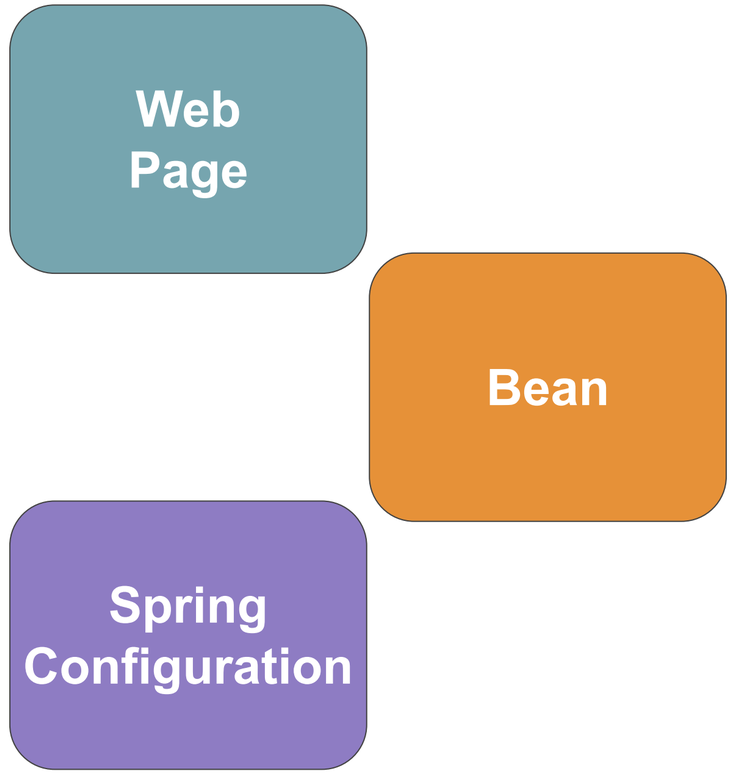
Spring MVC Behind the Scenes

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

Components of a Spring MVC Application

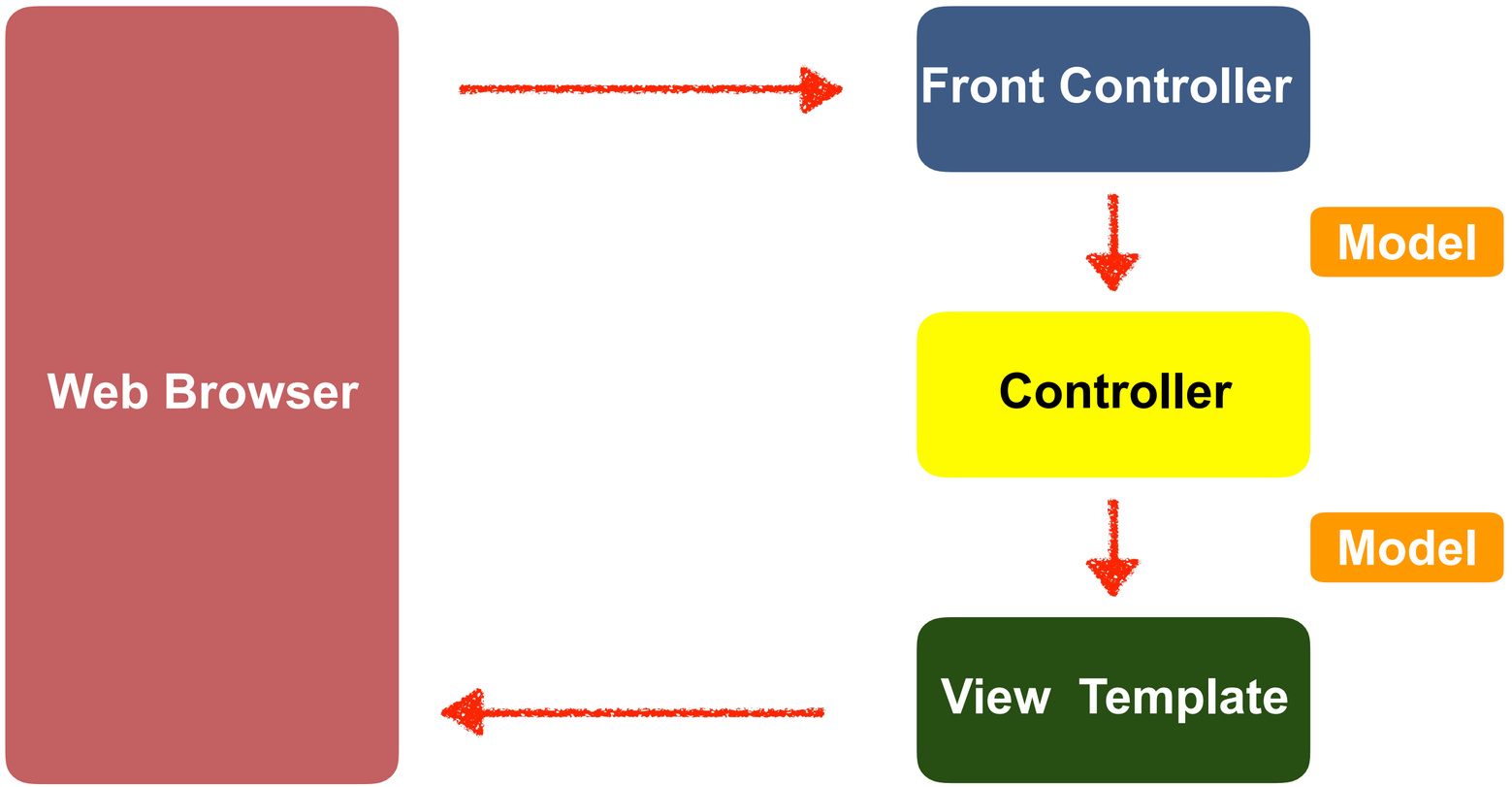
- A set of web pages to layout UI components
- A collection of Spring beans (controllers, services, etc…)
- Spring configuration (XML, Annotations or Java)



Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

How Spring MVC Works Behind the Scenes



Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

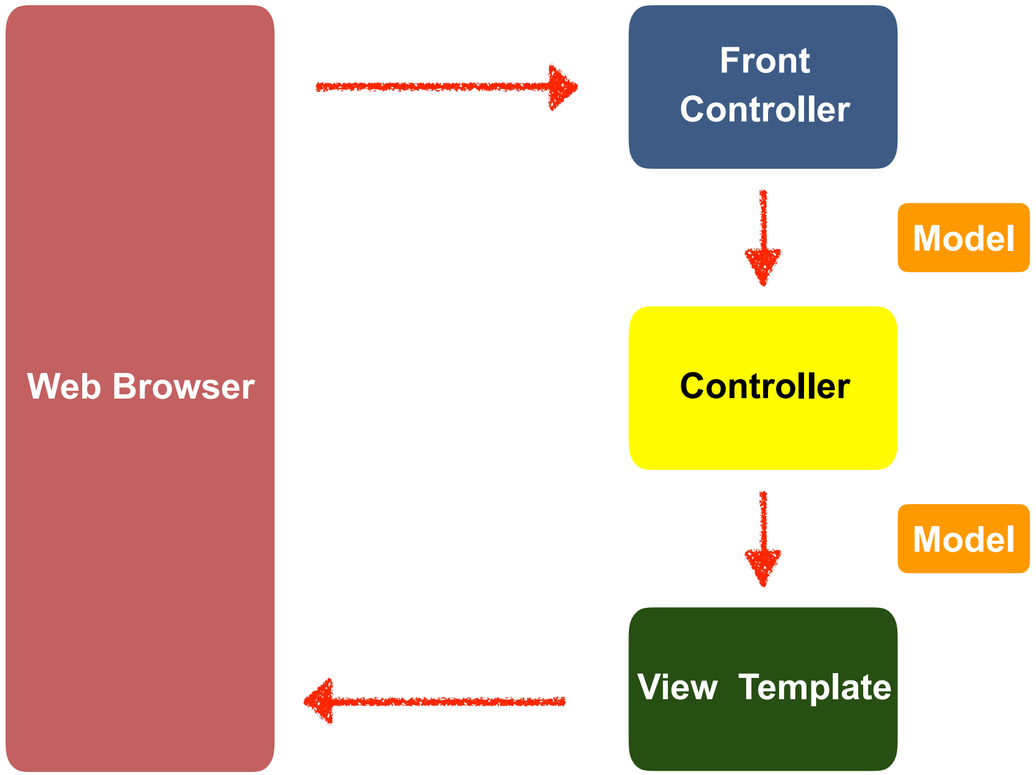
Spring MVC Front Controller

Front controller known as DispatcherServlet

- Part of the Spring Framework
- Already developed by Spring Dev Team

You will create

- **M**odel objects (orange)
- **V**iew templates (dark green)
- **C**ontroller classes (yellow)



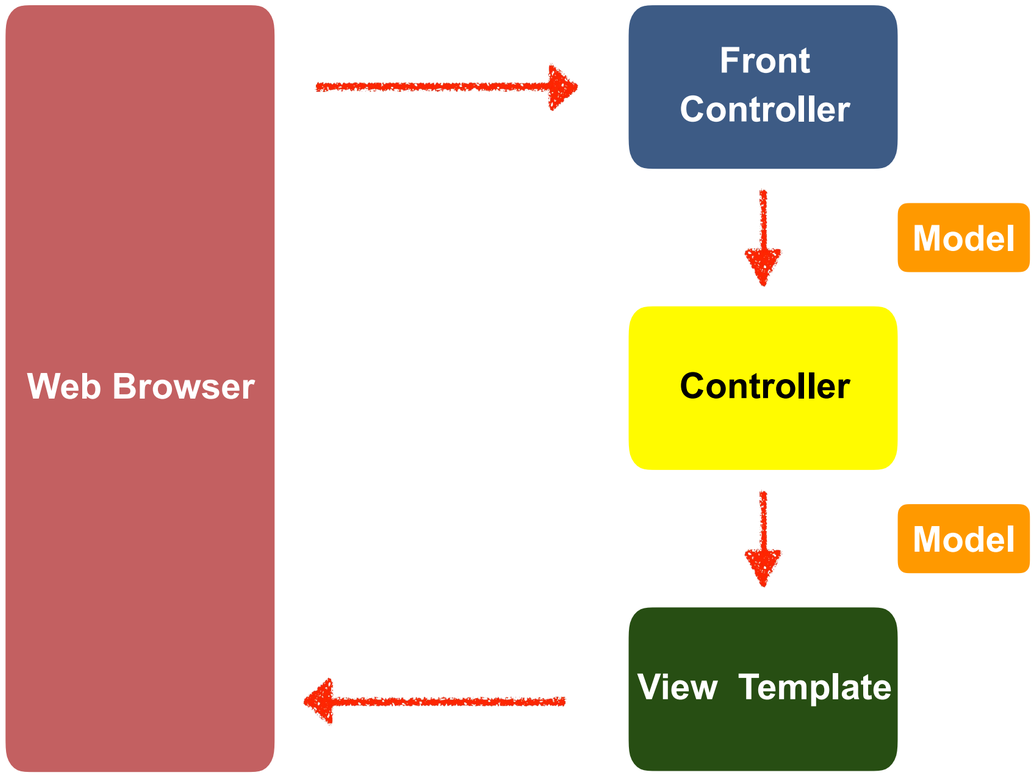
Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

Controller

Code created by developer

- Contains your business logic
- Handle the request
- Store/retrieve data (db, web service…)
- Place data in model
- Send to appropriate view template

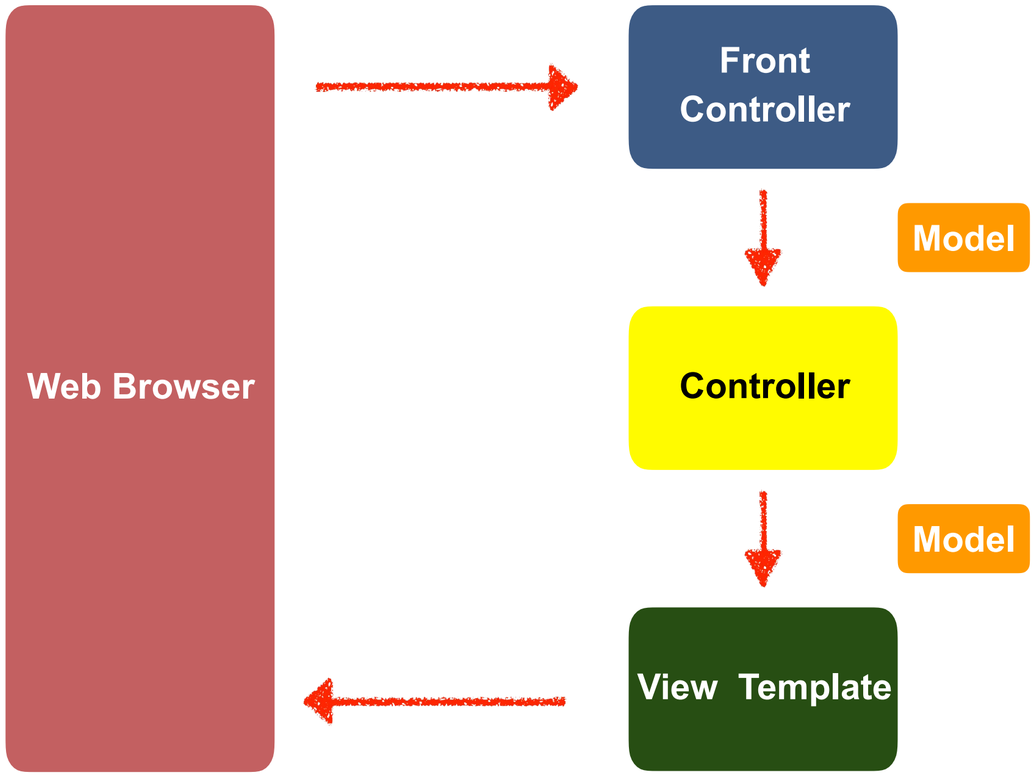


Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

Model

- Model: contains your data
- Store/retrieve data via backend systems
  - database, web service, etc…
  - Use a Spring bean if you like
- Place your data in the model
  - Data can be any Java object/collection

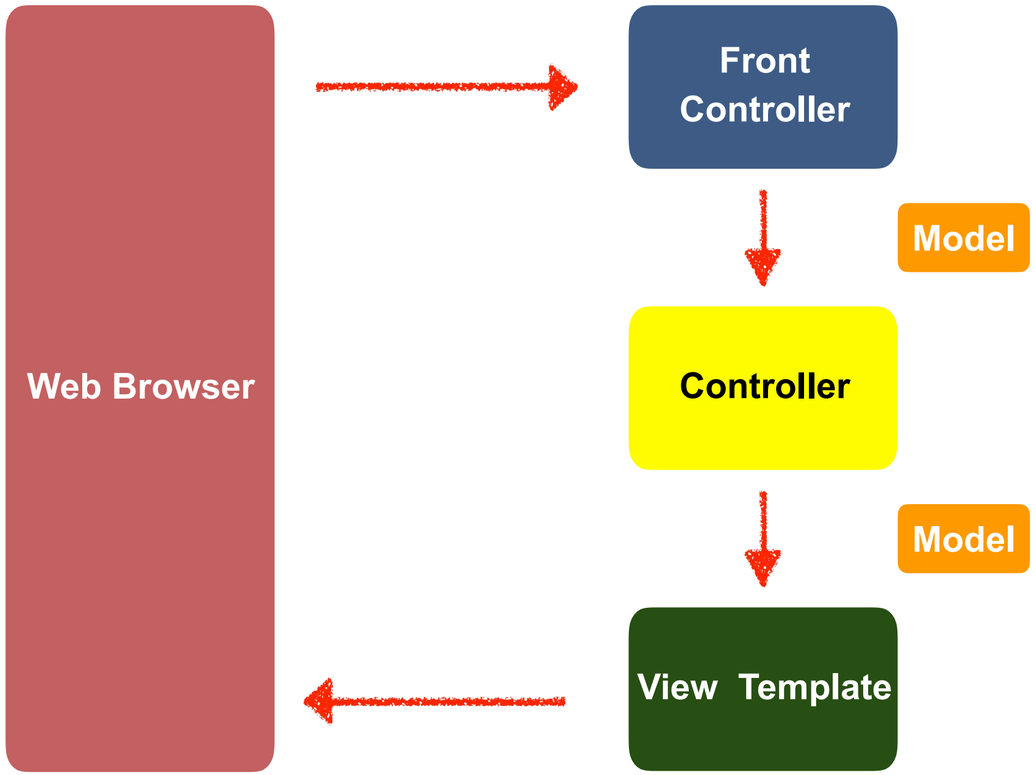


Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

View Template

- Spring MVC is flexible
  - Supports many view templates
- Most common is **JSP** + **JSTL**
- Developer creates a page
  - Displays data



Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

View Template (more)

- Other view templates supported
  - Thymeleaf, Groovy
  - Velocity, Freemarker, etc…
- For details, see:

https://docs.spring.io/spring/docs/current/spring-framework-reference/web.html#mvc-view

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

Dev Environment Check Point

At this point of the course you should have installed:

- Apache Tomcat 8.03
- Eclipse (Java EE version)
- Connected Eclipse to Tomcat

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

Dev Environment Check Point

Additional Things To Do:

- Download the **Jars** necessary for annotations, and validation from Github

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

Spring MVC Configuration

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

Spring MVC Configuration Process - Part 1

Add configurations to file: **WEB-INF/web.xml**

1. Configure Spring MVC Dispatcher Servlet

2. Set up URL mappings to Spring MVC Dispatcher Servlet

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

Spring MVC Configuration Process - Part 2

Add configurations to file: **WEB-INF/spring-mvc-servlet.xml**

3. Add support for Spring component scanning

4. Add support for conversion, formatting and validation

5. Configure Spring MVC View Resolver

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine RIGAT

Step 1: Configure Spring DispatcherServlet

File: web.xml

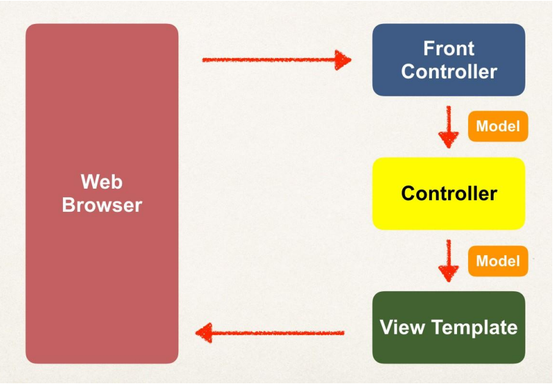
```
<web-app>

    <servlet>
        <servlet-name>dispatcher</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/spring-mvc-servlet.xml</param-value>
        </init-param>

        <load-on-startup>1</load-on-startup>
    </servlet>

</web-app>
```



Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine RIGAT

Step 2: Set up URL mappings to Spring MVC Dispatcher Servlet

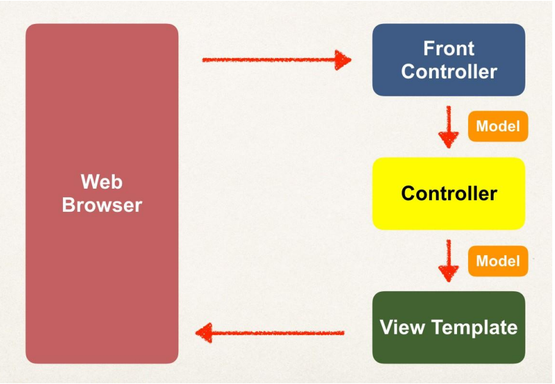
File: web.xml

```
<web-app>

    <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>
```



Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

Step 3: Add support for Spring component scanning

File: spring-mvc-servlet.xml

```
<beans>

    <!-- Step 3: Add support for component scanning -->
    <context:component-scan base-package="edu.javaweb.spring.mvc" />

</beans>
```

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine RIGAT

Step 4: Add support for conversion, formatting and validation

File: spring-mvc-servlet.xml

```
<beans>

    <!-- Step 3: Add support for component scanning -->
    <context:component-scan base-package="edu.javaweb.spring.mvc" />

    <!-- Step 4: Add support for conversion, formatting and validation support -->
    <mvc:annotation-driven/>

</beans>
```

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

Step 5: Configure Spring MVC View Resolver

File: spring-mvc-servlet.xml

```
<beans>

    <!-- Step 3: Add support for component scanning -->
    <context:component-scan base-package="edu.javaweb.spring.mvc" />

    <!-- Step 4: Add support for conversion, formatting and validation support -->
    <mvc:annotation-driven/>

    <!-- Step 5: Define Spring MVC view resolver -->
    <bean
        class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix" value="/WEB-INF/view/" />
        <property name="suffix" value=".jsp" />
    </bean>
</beans>
```

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

View Resolver Configs - Explained

When your app provides a “view” name, Spring MVC will

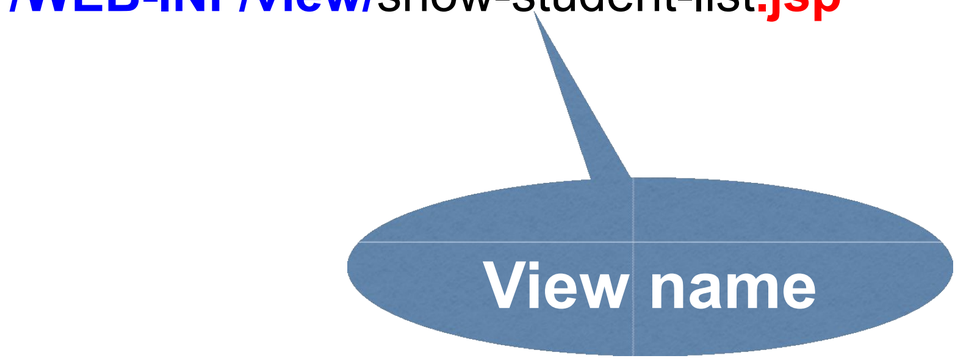
- prepend the prefix
- append the suffix

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

Azzeddine RIGAT

View Resolver Configs - Explained

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



Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine RIGAT

Summary of Spring Config:

- Two steps in **WEB-INF/web.xml**
- Three steps in our xml config

Check **spring-mvc- demo1-config-files**

Topic 2, Spring MVC - Building Spring Web Apps

Azzeddine
RIGAT

Assignment 8

Configure the Spring Dispatcher Servlet using all Java Code (no xml).html

Topic 2, Spring MVC - Creating Controllers and Views

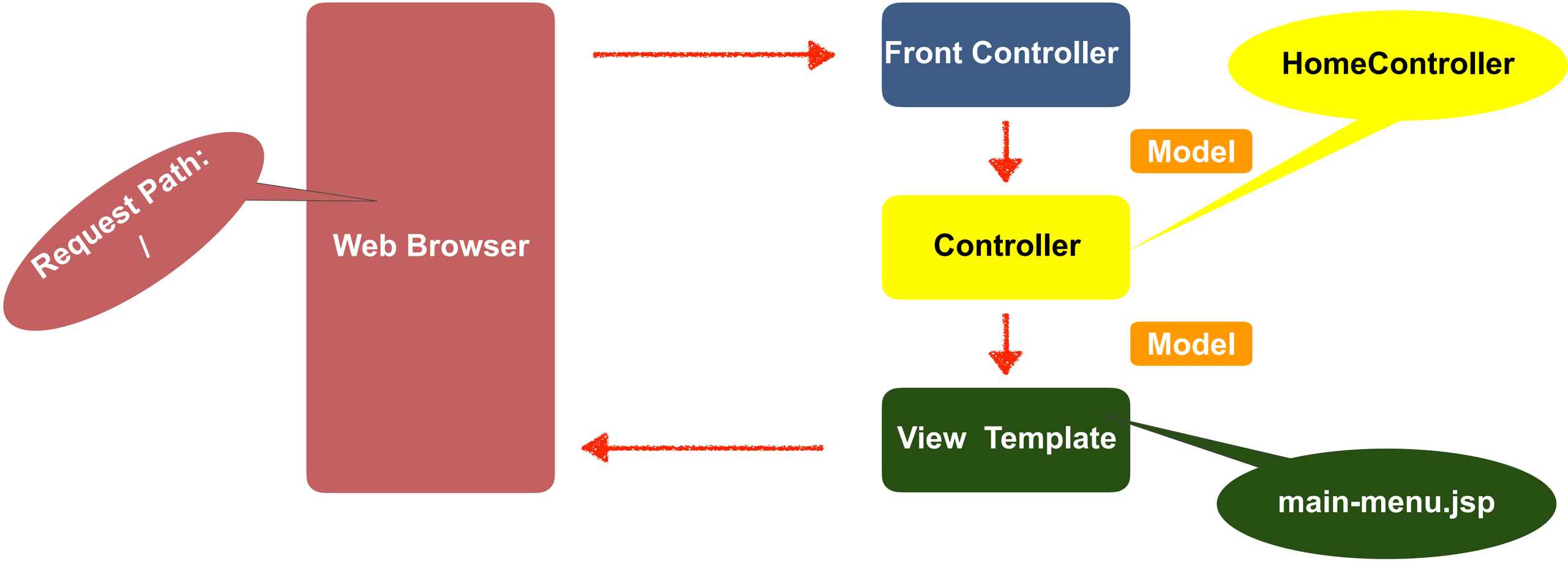
Azzeddine
RIGAT

Creating Controllers and Views

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Our First Spring MVC Example



Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Development Process

1.Create Controller class

2.Define Controller method

3.Add Request Mapping to Controller method

4.Return View Name

5.Develop View Page

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine RIGAT

Step 1: Create Controller class

Annotate class with **@Controller**

- **@Controller** inherits from @Component … supports scanning

```
@Controller
public class HomeController {

                                                    }
```

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Step 2: Define Controller method

```
@Controller
public class HomeController {

    public String showMyPage() {
...
                                    }

}
```

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Step 3: Add Request Mapping to Controller method

```
@Controller
public class HomeController {
    @Request  Mapping("/")
        public String showMyPage() {
                ...
    }

}
```

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Step 4: Return View Name

```
@Controller
public class HomeController {

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

}
```

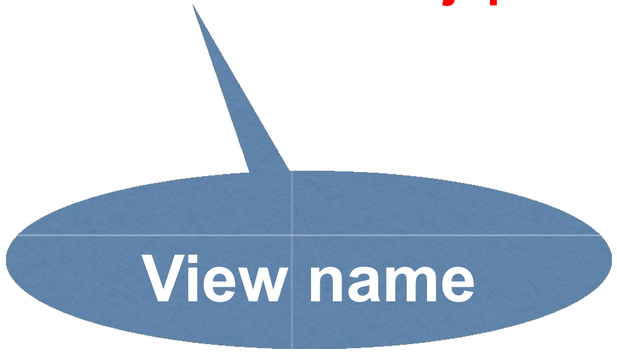
Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Finding the View Page

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

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




```
@Controller
public class HomeController {

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

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Step 5: Develop View Page

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

```
<html>
    <body>

    <h2>Spring MVC Demo - Home Page</h2>

    </body>
</html>
```

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Recap - Development Process

1.Create Controller class

2.Define Controller method

3.Add Request Mapping to Controller method

4.Return View Name

5.Develop View Page

Check **spring-mvc-demo2-create-home-controller-and-view**

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Assignment 9

Use CSS, JavaScript and Images in Spring MVC Web App.html

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Reading Form Data with Spring MVC

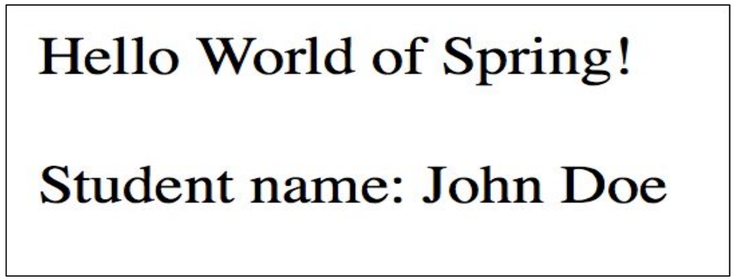
Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

High Level View

helloworld-form.jsp

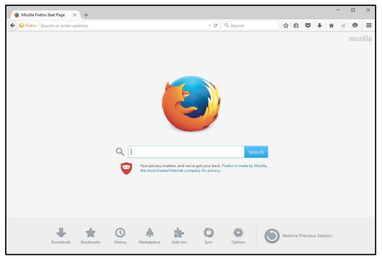
helloworld.jsp



Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Application Flow



Request Mapping
/showForm

HelloWorld Controller

helloworld-form.jsp

What's your name? Submit Query

helloworld-form.jsp

What's your name? Submit Query

Request Mapping
/processForm

HelloWorld Controller

helloworld.jsp

Hello World of Spring!

Student name: John Doe

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Controller Class

```
@Controller
public class HelloWorldController {
    // need a controller method to show the initial HTML form
    @RequestMapping("/showForm")
    public String showForm() {
        return "helloworld-form";
    }
    // need a controller method to process the HTML form
    @RequestMapping("/processForm")
    public String processForm() {
        return "helloworld";
    }
}
```

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Development Process

1. **Create Controller class**

2. **Show HTML form**
   a. Create controller method to show HTML Form
   b. Create View Page for HTML form

3. **Process HTML Form**
   a. Create controller method to process HTML Form
   b. Develop View Page for Confirmation

Check **spring-mvc- demo3-reading-html-form-data**

Topic 2, Spring MVC - Creating Controllers and Views

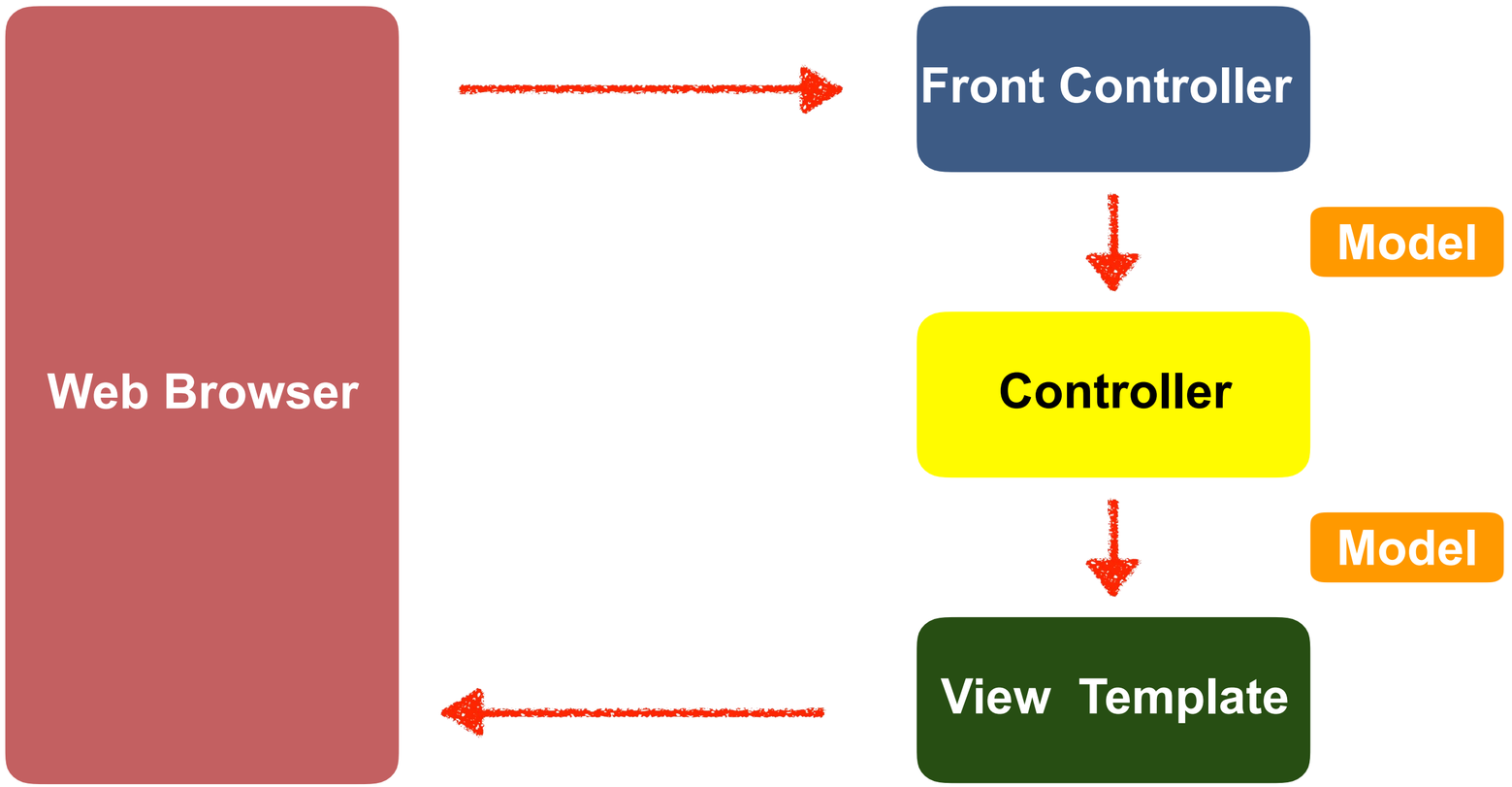
Azzeddine
RIGAT

Adding Data to Spring Model

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Focus on the Model

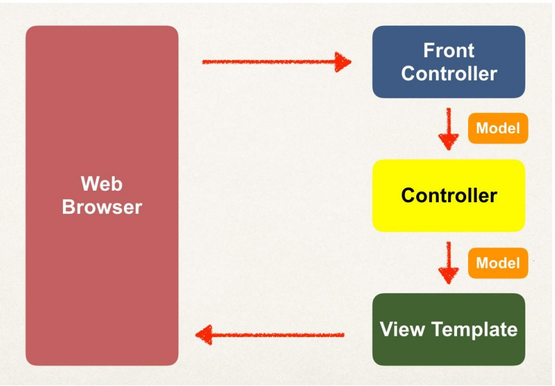


Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Spring Model

- The **Model** is a container for your application data
- In your Controller
  - You can put anything in the **model**
  - strings, objects, info from database, etc…
- Your View page (JSP) can access data from the **model**



Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Code Example

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

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine RIGAT

Passing Model to your Controller

```
@RequestMapping("/processFormVersionTwo")
public String letsShoutDude(HttpServletRequest request, Model model) {

    // read the request parameter from the HTML form
    String theName = request.getParameter("studentName");
    // convert the data to all caps
    theName = theName.toUpperCase();
    // create the message
    String result = "Yo! " + theName;
    // add message to the model
    model.addAttribute("message", result);

    return "helloworld";
}
```

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

View Template - JSP

```
<html>
    <body>

        Hello World of Spring!
        ...

        The message: ${message}

        </body>
</html>
```

Check **spring-mvc- demo4-adding-data-to-the-spring-model**

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Adding more data to your Model

```
// get the data
//
String result = …
List<Student> theStudentList = …
ShoppingCart theShoppingCart = …

// add data to the model
//
model.addAttribute("message", result);
model.addAttribute("students", theStudentList);
model.addAttribute("shoppingCart", theShoppingCart);
```

Topic 2, Spring MVC - Creating Controllers and Views

Azzeddine
RIGAT

Assignment 10

Add a shoppingcart class, with at least attributes: item name and its price
Add a form text that has at least three text inputs then submit them using the the HttpServerletRequest
Finally, display your shopping items, and the sum of the items.

Topic 2, Spring MVC - Request Params and Request Mappings

Azzeddine
RIGAT

Reading HTML Form Data with **@RequestParam** Annotation

Topic 2, Spring MVC - Request Params and Request Mappings

Azzeddine
RIGAT

Instead of using HttpServletRequest

```
@RequestMapping("/processFormVersionTwo")
public String letsShoutDude(HttpServletRequest request, Model model) {

    // read the request parameter from the HTML form
    String theName = request.getParameter("studentName");

    ...
}
```

Topic 2, Spring MVC - Request Params and Request Mappings

Azzeddine
RIGAT

Bind variable using @RequestParam Annotation

```
@RequestMapping("/processFormVersionTwo")
    public String letsShoutDude(
        @RequestParam("studentName") String theName,  Model model) {

    // now we can use the variable: theName

}
```

Behind the scenes:

- **Spring will read param from request: studentName**
- **Bind it to the variable: theName**

Check **spring-mvc- demo5-binding-request-params**

Topic 2, Spring MVC - Request Params and Request Mappings

Azzeddine
RIGAT

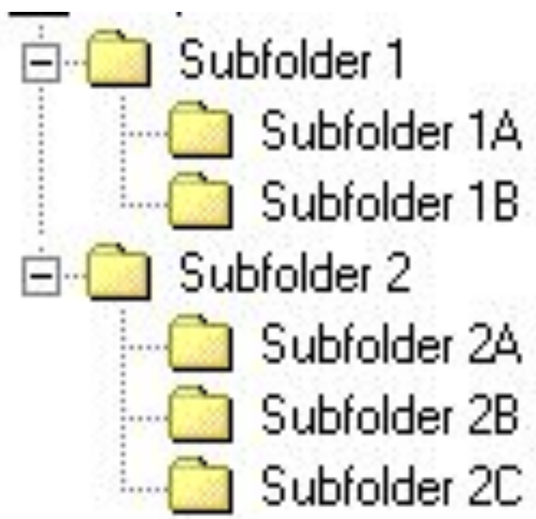
Add **@RequestMapping** to Controller

Topic 2, Spring MVC - Request Params and Request Mappings

Azzeddine
RIGAT

Adding Request Mappings to Controller

- Serves as parent mapping for controller
- All request mappings on methods in the controller are relative
- Similar to folder directory structures



Topic 2, Spring MVC - Request Params and Request Mappings

Azzeddine RIGAT

Controller Request Mapping

```
@RequestMapping("/funny")
public class FunnyController {

    @RequestMapping("/showForm")
    public String showForm() {
    ...
    }

    @RequestMapping("/processForm")
    public String process(HttpServletRequest request, Model model) {
    ...
    }
}
```

Controller Mapping

/funny/showForm

/funny/processForm

Check **spring-mvc- demo6-controller-level-mappings**

Topic 2, Spring MVC - Form Tags and Data Binding

Azzeddine
RIGAT

Spring MVC Form Tags

Topic 2, Spring MVC - Form Tags and Data Binding

Azzeddine
RIGAT

Review HTML Forms

- HTML Forms are used to get input from the user

Topic 2, Spring MVC - Form Tags and Data Binding

Azzeddine
RIGAT

Spring MVC Form Tags

- Spring MVC Form Tags are the building block for a web page

- Form Tags are configurable and reusable for a web page

Topic 2, Spring MVC - Form Tags and Data Binding

Azzeddine
RIGAT

Data Binding

- Spring MVC Form Tags can make use of data binding

- Automatically setting / retrieving data from a Java object / bean

Topic 2, Spring MVC - Form Tags and Data Binding

Azzeddine
RIGAT

Spring MVC Form Tags

- Form tags will generate HTML for you :-)

Form Tag	Description
form:form	Main form container
form:input	Text field
form:textarea	Multi-line text filed
form:checkbox	Check box
form:radiobutton	Radio buttons
Form:select	Drop down list
more….	