Session 25

Spring Controller

1

Reading & Reference

- Reading

  www.tutorialspoint.com/spring/spring_web_mvc_framework.htm

  https://springframework.guru/spring-requestmapping-annotation/

- Reference
  - Spring home page

    spring.io/

  - Spring 4.3 Reference Documentation

    docs.spring.io/autorepo/docs/spring/4.3.0.RELEASE/spring-framework-reference/pdf/spring-framework-reference.pdf

2

Lecture Objectives

- Understand the structure of the Spring controller
- Become familiar with Spring controller annotation

3

CSE336 Recap

View

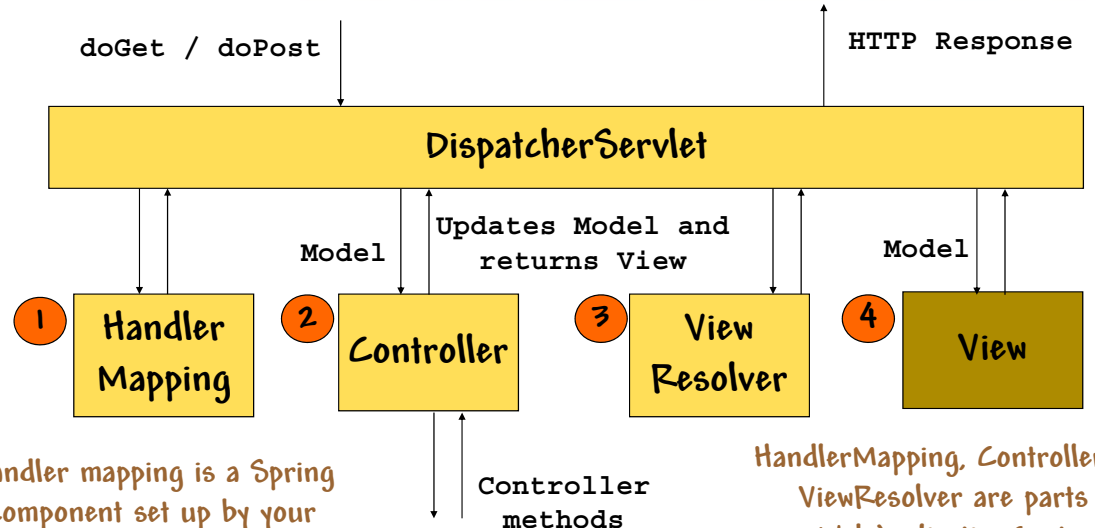
- Html
- CSS
- JSP
- DOM
- jQuery
- JavaScript

Model / Controller

- Http
- Servlet API
- JAX-RS
- Session
- ServletContext
- RequestDispatcher

4

Spring MVC Control Flow



Handler mapping is a Spring component set up by your controller annotation

HandlerMapping, Controller, and ViewResolver are parts of WebApplicationContext

Diagram from tutorialspoint.com

5

Model

- The Model object is a generalized version of the shared scopes you have used (Session, ServletContext, etc.)
- Each contains name/value pairs
- Usually accessed as a ModelMap

6

ViewResolver

- Allows you to render models in the browser
- Compatible with multiple view technologies
- Maps view names to actual views
- Spring contains multiple view resolvers, but you can add one of your own

7

View

- Html
- PDF
- XML
- Templates
  - JSP
  - ThymeLeaf
  - Velocity

8

Spring Controllers

- In the last session, we reviewed a very simple example of a Controller
- One class, one method
- All request return a simple string (RestController)
- Today we look at variations on the simple controller

```
package com.example.demo.controller;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.*;
@RestController
public class HelloController {
    @GetMapping
    public String getHello() {
        return "hello";    } }
```


9

Request Mapping Basics

- RequestDispatcher servlet routes HTTP requests to handler methods of controllers
- You specify the mapping of requests to handler methods using Spring annotation
- @RequestMapping - applied to classes and/or methods

The examples in this session use a RestController since it does not involve views

10

@RequestMapping With Multiple URIs

- Reference states that you can provide an array of URI mappings
- Requires full specification of annotation
- Better to associate individual mappings with separate methods with common logic abstracted into a private method

```
@RestController
@RequestMapping("/home")
public class Cse336Controller {

  @RequestMapping(
     value={"", "view*", "page*, **/msg"})
  String indexMultipleMapping(){
  return "Hello from multiple mapping";
    }
}
```

Does not work correctly

Notice the use of wild cards

11

@RequestMapping with @RequestParam

- @RequestParam used with @RequestMapping to bind a request parameter to the parameter of the handler method
- Note the differences in the two methods

```
@RequestMapping(value="/id")
  String getIdByValue(
    @RequestParam("id") String personId){
    return "Get parameter of id = " + personId;
  }
  @RequestMapping(value="/personId")
  String getId(@RequestParam String personId) {
    System.out.println("ID is " + personId);
    return "Get parameter of personId = " + personId;
  }
```

Used when variable name matches html form componenet name

12

@RequestMapping - Required Element

- You can declare whether or not a parameter is required
- If required is false, you can specify a defaultValue as an annotation element

```
@RequestMapping("/name") String
  getName(@RequestParam(value="person",
    required= true) String personName) {
      return "Required name is " +
        personName;
  }
```

Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Mon Nov 26 15:19:01 EST 2018
There was an unexpected error (type=Bad Request, status=400).
Required String parameter 'person' is not present

This Spring error message is returned when form parameter is not present

13

@RequestMapping With HTTP Method

- You can send different http requests with the same URI to different server methods.
- Use the RequestMethod enum
- Supported methods - GET, HEAD, POST, PUT, PATCH, and DELETE
- @GetMapping acts as a shortcut for @RequestMapping(method = RequestMethod.GET)
- Similar shortcuts for POST, PUT, DELETE, and PATCH

```
@RequestMapping(
method = RequestMethod.POST)
```


14

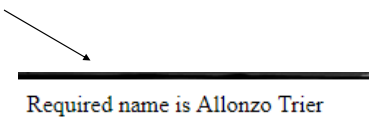
Other Mappings

- You can specify mappings with Producible and Consumable
- You can specify methods that respond to headers present in the request

15

Are We on Track?

- Build a simple html page that will contain links for some examples shown in these slides
- Build a Spring application that contains methods to respond to each
- Test each link by displaying each response in a new browser tab
- Sample response



Required name is Allonzo Trier

Click a link below to Test Your Spring Controller

- /home
- /home/view
- localhost:8080/home/id?id=5
- localhost:8080/home/personId?PersonId=5
- localhost:8080/home/name?person=Allonzo Trier
- localhost:8080/home/prod

The last block of code not in the slides. You should look up the annotation for produces

16

Were We on Track?

```
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.bind.annotation.RequestParam;

@RestController
@RequestMapping("/home")
public class Cse336Controller {

    @RequestMapping(value={"", "view*", "page*, **/msg"})
    String indexMultipleMapping(){
    return "Hello from index multiple mapping";
    }

    @RequestMapping(value="/id")
    String getIdByValue(@RequestParam("id") String personId){
        return "Get parameter of id = " + personId;
    }
...
```

17