Java Backend Development Live-85

lecture-10

Agenda

- Controller vs RestController
- Important Annotations: @Autowired, @Bean, @Qualifier, @Value, @Primary
- Lombok
- RESTFul Services
- CRUD application

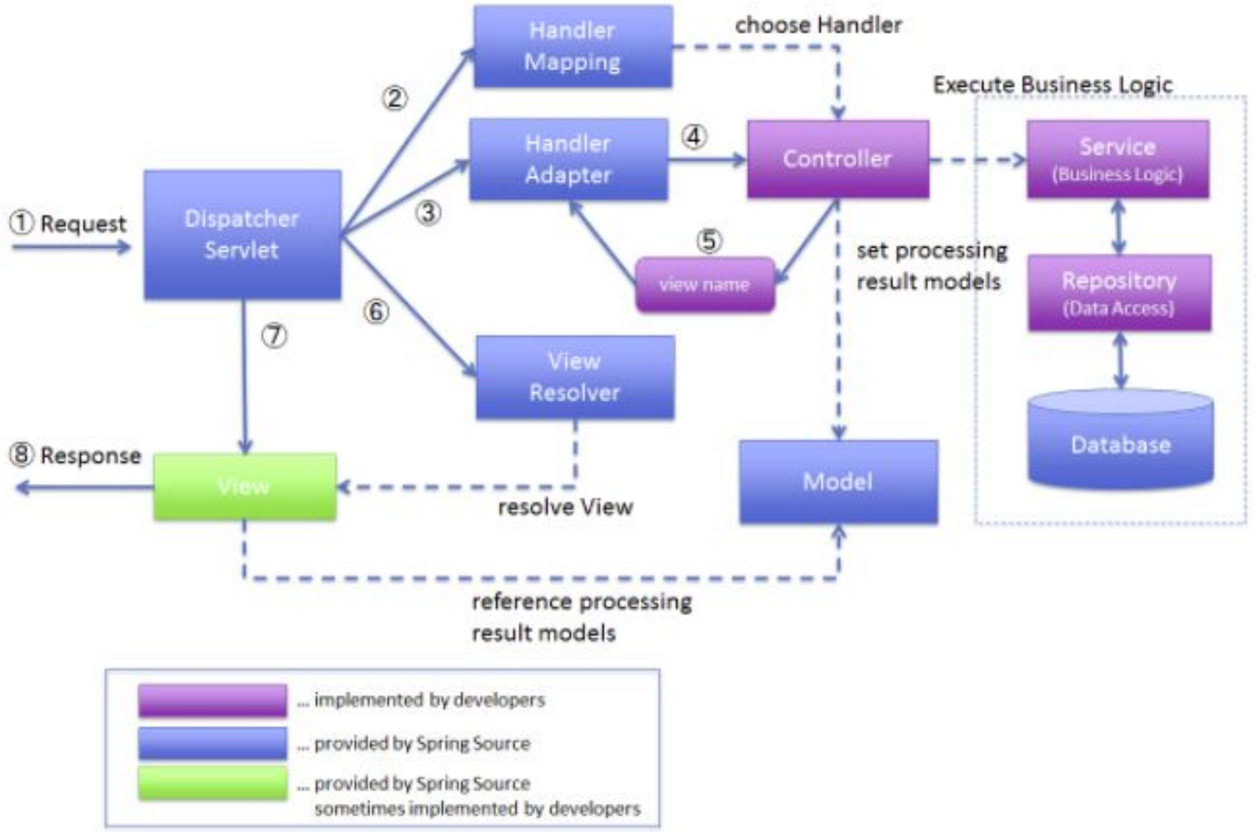
Spring MVC (Model-View-Controller)

- Spring MVC is a web framework that simplifies the development of web application by separating the application into three components.
  - The **Model** encapsulates the application/business data.
  - The **View** is responsible for rendering the model data and it generates HTML output that the client's browser can interpret.
  - The **Controller** is responsible for processing user requests and building an appropriate model and passes it to the view for rendering.
- Spring Boot enhances Spring MVC by providing auto-configuration, embedded servers, and starter dependencies.

Key Concepts of Spring MVC

- **DispatcherServlet**: The central component of Spring MVC. It acts as a front controller that handles all HTTP requests and responses.
- **Handler Mapping**: Maps incoming requests to the appropriate controller methods.
- **Controller**: Contains business logic and processes user requests.
- **View Resolver**: Resolves view names into actual view implementations (e.g., Thymeleaf templates, JSP).

Spring MVC Architecture



Controller vs RestController

@Controller create a Map of Model Object and find a view(jsp,html).

@RestController simply return object and object data directly written into http response as String, JSON or XML.

@Controller is used to serve HTML, JSP, etc pages. We can pass data as a Model to be filled in these pages (views).

Nowadays frontend code (HTML, js, etc.) is served by different servers (e.g nginx). Backend APIs are used to get data only.

Important Annotations

@Autowired: to mark a dependency which Spring is going to resolve and inject. @Autowired has a boolean argument called required with a default value of true.

@Bean: @Bean marks a factory method which instantiates a Spring bean. All methods annotated with @Bean must be in @Configuration classes.

@Qualifier: We use @Qualifier along with @Autowired to provide the bean id or bean name we want to use in ambiguous situations.

@Value: We can use @Value for injecting property values into beans.

@Primary: We mark the most frequently used bean with @Primary, it will be chosen on unqualified injection points

Lombok

- Lombok library aims to reduce boilerplate code in Java classes.
- It provides annotations that automatically generate code for common tasks such as getter and setter methods, constructors, equals and hashCode methods, and more.
- Commonly used annotations provided by Lombok:
  - @Setter, @Getter, @ToString, @EqualsAndHashCode
  - @NoArgsConstructor, @RequiredArgsConstructor, @AllArgsConstructor
  - @Data

RESTFul Services

REST stands for Representational State Transfer.

It is an **architecture style** for designing **loosely coupled** applications over **HTTP**, that is often used in the development of web services.

REST does **not enforce** any rule regarding how it should be implemented at the lower level, it just put high-level design guidelines and leaves us to think of our own implementation.

HTTP Methods

GET: GET APIs should be **idempotent**. Making multiple identical requests must produce the same result every time until another API (POST or PUT) has changed the state of the resource on the server. Response Codes: 200, 404, 400

POST: POST methods are used to create a new resource into the collection of resources.Please note that POST is **not idempotent**, Response Codes: 201 (Created), 200(OK), 204(No Content)

PUT: Use PUT APIs primarily to update an existing resource (if the resource does not exist, then API may decide to create a new resource or not).201 (Created), 200 (OK) , 204 (No Content), 404 (NOT FOUND)

DELETE: DELETE APIs delete the resources (identified by the Request-URI). 200 (OK), 202 (Accepted), 204 (No Content), 404 (NOT FOUND)

An **idempotent** HTTP method is a method that can be invoked many times without the different outcomes. It should not matter if the method has been called only once, or ten times over. The result should always be the same.

POST is **NOT idempotent.**

GET, PUT, DELETE, HEAD, OPTIONS and TRACE are idempotent.