Class 5 – Spring Boot Microservice Enterprise Applications with Java

Contents

Review

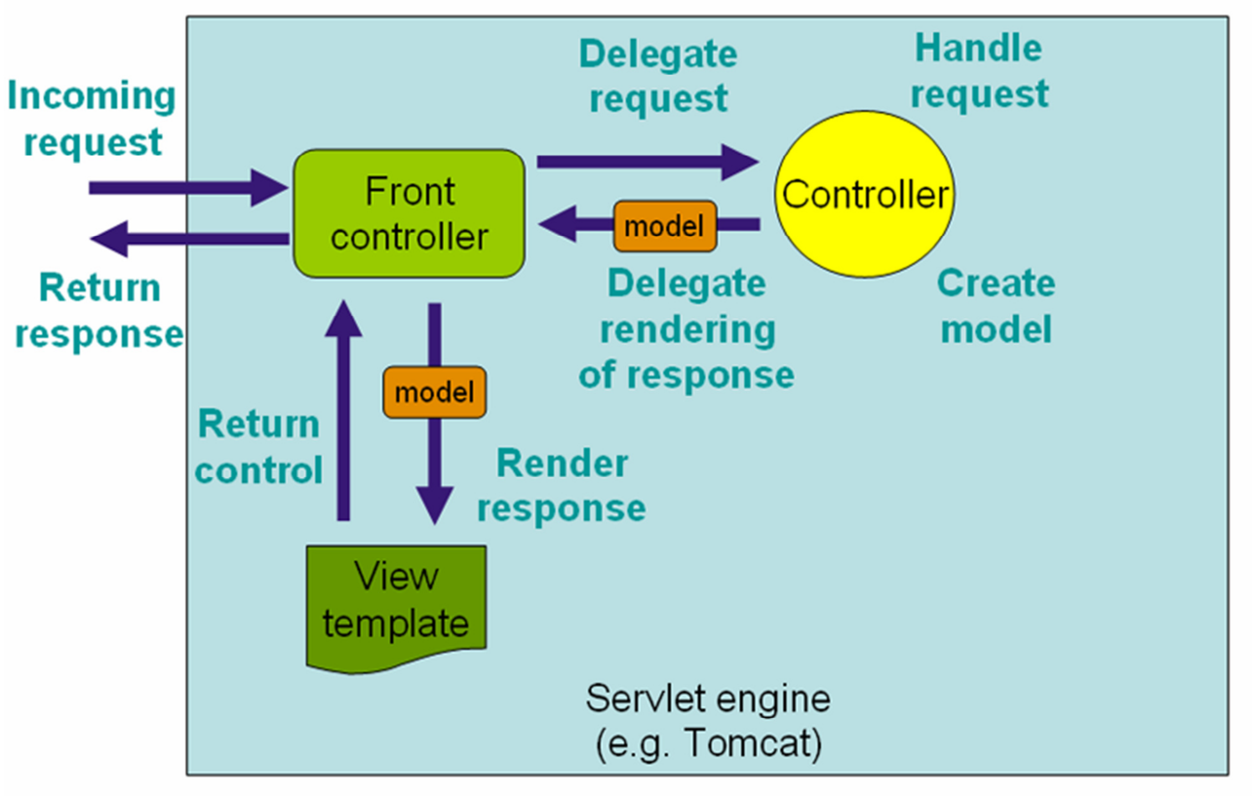
- Class4
- Q&A

Break

Spring MVC[1]

This is the project of Spring that is used to develop web applications – HTML app, REST etc. We will be using it for the REST endpoints we create. It is comparable to the JAX-RS specification but it does not implement it. It uses a similar Dispatcher servlet that handles the incoming requests.

[1] https://docs.spring.io/spring/docs/current/spring-framework-reference/html/mvc.html



We will use this together with Spring Boot which adds some convenience and simplifies configuration as we will see in the next section.

- @RestController
- @RequestMapping

Spring Boot[2]

- "Convention over configuration" for Spring
- Managed dependencies with multiple "starters"
  - Spring Boot parent pom
  - Spring Cloud BOM
- Intelligent defaults so production ready with minimal changes
- Many production ready endpoints in the Actuator project
- Embedded servlet container (default is Tomcat) – no need for separate servers
- Helpful for Microservices as you need to create multiple ones and a lot boilerplate stuff is taken care of
  - Logging config
  - Monitoring config
  - Spring dependencies and config

[2] https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/

    - Managing properties
    - Metrics
    - DB connections
- Multiple ways to get started – CLI, start.spring.io, within IDE
- "Fat Jars" – includes all dependencies
- "Auto configuration" of beans depending on dependencies added into POM

12-factor app[3]

In the modern era, software is commonly delivered as a service: called web apps, or software-as-a-service. The twelve-factor app is a methodology for building software-as-a-service apps that:

- Use declarative formats for setup automation, to minimize time and cost for new developers joining the project;
- Have a clean contract with the underlying operating system, offering maximum portability between execution environments;
- Are suitable for deployment on modern cloud platforms, obviating the need for servers and systems administration;
- Minimize divergence between development and production, enabling continuous deployment for maximum agility;
- And can scale up without significant changes to tooling, architecture, or development practices.

The twelve-factor methodology can be applied to apps written in any programming language, and which use any combination of backing services (database, queue, memory cache, etc).

We will be using these best practices throughout the examples.

Spring Boot example app

See code: https://github.com/rahulaga/hello-spring-boot

- @SpringBootApplication – defines:
    - Configuration
    - Auto configuration - depending on the dependencies you add to your POM spring will auto initialize beans based on a META/spring.factories file. Eg: db connectors. These are all @Conditional so depends on your POM
    - component scan

Run from:

- Within STS
- On command line mvn spring-boot:run
- Run fat jar
    - mvn clean package

[3] https://12factor.net/

- java –jar /path/to/.jar

Load page: http://localhost:8080/hello

Break

TBTF bank example Microservices

Now for some more realistic examples that will also serve as a template for your assignment. See code in File resources in the online portal.

We had defined the APIs in Class4.

Example implementation: user-service1 and checking-account-service1

Various ways to start the Spring Boot app[4]:

- Starting the server outside of Eclipse: `mvn spring-boot:run`
- Start from within STS (View->Boot Dashboard) Easy to debug and auto deploy
- Run fat jar with java -jar

Use Chrome/Firefox plugins for REST Clients. Example Postman Collection in the sample.

Testing REST services – manual

Plugins for your browser – REST Client, Postman etc. On Chrome:

https://chrome.google.com/webstore/detail/postman/fhbjgbiflinjbdggehcddcbncdddomop?hl=en

https://chrome.google.com/webstore/detail/advanced-rest-client/hgmloofddffdnphfgcellkdfbfbjeloo?hl=en-US

Homework for Class 5

Complete Assignment 2, due at next class.

[4] http://docs.spring.io/spring-boot/docs/current/reference/html/using-boot-running-your-application.html