Full stack Microservices by Spring Framework

40 hours

Course Overview:

This training is relevant for java programmers. The course includes 3 modules:

- Spring Core
- Spring Boot
- Writing and testing Microservice

Many years ago Java developers used news in order to create some Java services. They had many handmade configurations mixed with business logic; they even were using copy-paste techniques. They wrote many lines of the same ugly code, which sometime worked, and sometimes not, but they understood everything they did (mostly). They could easily use debug in order to find a problem and solve it.

And then Spring happened and things have changed… We received a lot of magic from Spring black box and our code became much cleaner and more simple. Business aspects became apart from other technical aspects and configuration code, but debug became more complicated. After we learned how Spring is working we know how to debug our code and solve more complicated problems (mostly).

And then Spring Boot happened…

On the one hand, it solved thousands of problems: versions conflict, configuration issues, infrastructure's beans declarations, environments problems, and even running or deploying applications including building jar/war… On the other hand, Spring Boot uses so much magic that mostly we have only two scenarios:

1. Everything is working without any effort.

2. Nothing is working and nobody knows why.

In this training we will write our own spring framework in order to understand both spring concepts and spring internals. During the training we will drill down deep to spring core, we will see how you can write bean post processors and bean factory post processors, how to configure different spring profiles, how to write custom scopes. We will talk about different spring problems and solutions. We will compare strategies of different context types (xml, annotations, java config and groovy config). What are their advantages, and when to use what?

We will reveal Spring Boot magic by discovering Spring Boot concepts, conventions and the way it works. And no matter that after this training you will understand that there is no magic you will enjoy Spring Boot even more, because you will be able to solve Spring Boot problems or conflicts without calling 911.

In the end of the training you will write your own application which will integrate several Microservices.

Who should attend:

Java developers

Course Contents:

Spring Core

- Design patterns
  - Factory
  - Singleton
  - Chain of responsibilities
  - Proxy
- Reflections
  - Annotations
  - Dynamic Proxy
  - CGLib
- Spring concepts
  - IOC
  - Dependency Injection
  - Container
  - Bean
- Spring XML context
- Bean lifecycle
- Init / destroy method
- BeanPostProcessors / BeanFactoryPostProcessors
- ApplicationListeners
- FactoryBeans
- Annotation Config
  - Standard and Spring annotations
  - Qualifiers
- Java Config
  - Bean Declaration
  - Static beans
  - Refreshing prototypes in singletons
- Proxy Mode
- Java Config problems
- What is better (Java Config / Xml / Annotations)
- Spring profiles / Conditional
- Tests with Spring

Spring AOP

- Aspect
- Advice

- JoinPoint
  - @Before / @After
  - @AfterReturning / @AfterThrowing
  - @Around
- Pointcut
- Weaving
- Best practice
  - Custom annotations
  - Exception handling
  - Logging

Spring-ORM

- ORM concept
- JPA and Hibernate
- Entity
- EntityManager
- ORM mismatches
  - Embeded entities
  - One to Many / Many to One
  - Inheritance
- Cascade
- Spring configuration
  - DataSource
  - EntityManagerFactoryBean
  - @PersistenceContext
- Spring TX (Transactions)
  - ACID
  - Transaction Manager
  - Declarative transactions
  - Isolation
  - Propagation

Web & Spring MVC

- Servlet – are they still exists?
- Rest
- Controllers / RestControllers
- RestTemplate
- Tomcat
  - web.xml
  - without web.xml
  - without tomcat
- Async work
  - Java Executers
  - @Async
  - DefferedResult
  - ListennableSettableFuture
- Model
  - Jackson
  - Inheritance hell

- Lombok composition

Spring Boot

- How magic happens
  - Dependency management
  - Spring-boot-maven-plugin
  - autoconfiguration
- Conditional annotations
  - @OnBeanCondition
  - @OnMissingClass
  -
- How to override default configuration
  - application.properties and profiles
  - default beans
  - other techniques
- start.spring.io
- spring boot starters
  - spring.factories
  - spring data
  - spring data rest
  - lets write starter

Testing

- Unit test
  - JUnit
  - Mockito
  - Best practice
- Component Test
  - SpringRunner
  - Active Profiles
  - @ContextConfiguration
  - @SpringBootTest
  - @MockBean
  - Test controlers with MockMvc
- Microservice test
  - TestRestTemplate
  - Web environment
  - Mocking SDK
  - Wire mock