Java Spring Boot MicroServices Index

Java Language Fundamentals

Variables
Static variables
Instance variables
Local variables
Reference variables
Instance flow
Static Flow
Interface
default methods in interface (JAVA 1.8)
static methods in interfaces (JAVA 1.8)
private methods in interface (JAVA 1.9)
Anonymous Innerclass
Lambda Expression
- Rules
- Examples

.forEach (JDK 1.8)
Predefined Java Functional interfaces
Predicate
Filter
Consumer
Supplier
Stream API
Comparator
Comparable
Sorting Data using Java 8
Default Sorting
Customized Sorting
Abstract Classes
Use cases
Rules
interfaces vs abstract class vs class
Multithreading Java
- Runnable & Thread
- Executor Framework
- Execute Service, Callable , Future
- ForkJoinPool
- CompletableFuture (Java 1.8)

Runnable + Lambda Expression

Annotation (JDK 1.5)

- Java Predefined Annotations

@override , @Deprecated, @SupressWarnings, @FunctionalInterface(1.8)

- Custom Annotations

Declarations

Useage

Processing

Java Reflection API

Class, Method, Field, Constructor

MAVEN

- What is Build tool
- Maven Installation
- pom.xml
- Repositories (Central / Local / Remote)
- Creating Maven Project From Eclipse
- Archtype
- Group Id
- Artifact Id
- Version (Snapshot vs Release[GA])
- Settings.xml
- Dependencies Management
- Dependency Resolution Process
- Project Structure
- Maven Lifecycle
- Parent POM
- Customer Parent POM like Spring Boot Parent
- Creating Maven Project
- Multi Module Maven project

Spring Core

- History
- Release history
- Java vs Spring
- Java Version Compatibility with Spring

First Spring Example

Configuring Metadata in Spring

XML

Annotations (Component Scan)

Java (@Configuration/ @Bean)

XML vs Annotation vs Java configuration

Ways to create SpringBoot applications

- Spring Initializer (https://start.spring.io)

- Using IDE (Eclipse /STS/ Intellij IDE)
- Converting Maven project into Spring boot project

Spring Boot First Example
Spring Boot Java Configuration
Spring Boot XML Configuration
Duplication Id in Spring configuration
NoUniqueBeanDefination
Inversion Of Control (IOC)
- Dependency LookUP (Dependency Pull / Contextualized DL)
- Dependency Injection

CDI (Constructor Dependency Injection)
SDI (Setter Dependency Injection)
- CDI vs SDI
- CDI Use Cases
- SDI Use Cases

CDI in XML & Java Configuration
SDI in XML & Java Configuration
Circular Dependencies (CDI vs SDI)
Dependency Resolution Process
@Autowired
- by Type
- by Name
- by Constructor

@Qualifier
@Primary
@Autowired
- required false

Bean Lazy Initialization
Bean Dependency
Bean initialization callbacks
- InitializingBean (afterpropertiesSet)
- @PostConstruct
- Java/XML init method configuration

Bean destruction callbacks
- Disposablebean (destroy)
- @PreDestroy
- Java/XML destroy method configuration

Bean Scopes
- Singleton
- Prototype
- Request
- Session
- Application

- Custom Scope

Mixing 2 Scopes (Singleton + Prototype)

- Method Injection
- Java Singleton vs Spring Singleton

Spring Bean Instantiation Types

- XML Configuration
- Java Configuration
- How to Instantiate singleton classes using Spring

Spring Boot

- Maven Spring Boot Configuration
- Running the Example
- Creating an Executable Jar
- Spring Boot Maven Plugin
- Running Spring Boot Application from Command Line

Spring Boot Application startup Failure

Spring Boot Project Structure Best practises

Spring Boot default properties/yml file support (application.properties/ application.yml)

Auto Configuration

How to disable AutoConfiguration

Developer Tools

Externalized Configuration

- Application Property Files
- Accessing Command Line Properties
- Placeholders in Properties
- YML vs Properties
- JavaBean properties binding
- @ConfigurationProperties
- @PropertySource
- @PropertySources
- @Value
- Environment.java
- Binding Relaxation

Profiles

- Spring Boot Profile Management
- How to activate profile dynamically
- profile specific properties configuration

Logging

- Log Format
- Console Output
- Spring Boot Logback
- Profile-specific Configuration

Servlet API

- Http Protocol
- Http Methods
  - GET
  - POST
  - PUT
  - DELETE
  - TRACE
  - HEAD
  - Options etc…
- Http Status Code
  - 100 - 200
  - 200 - 300
  - 300 - 400
  - 400 - 500
- GET vs POST vs PUT vs DELETE
- Http Request Template
- Http Response Template
- Interfaces
- Servlet
  - Life cycle methods
- GenericServlet
- HttpServlet
- **Servlet Life Cycle**
- URL formation

Spring MVC + Boot

- Spring Boot MVC First application
- Spring MVC
  - web.xml + spring XML configuration
  - without web.xml + Spring Java Configuration
  - without web.xml + Spring XML Configuration

Spring Boot Restful Web services

- Spring Boot vs Spring MVC
- Dispatcher Service AutoConfiguration
- mobile-service microservice implementation
  - http://localhost:8080/msk/mobiles → GET all mobiles
  - http://localhost:8080/msk/mobiles/{mobileId} → GET mobile by mobile Id
  - http://localhost:8080/msk/mobiles → POST to save mobile
  - http://localhost:8080/msk/mobiles/{mobileId} → Delete to delete a mobile
  - http://localhost:8080/msk/mobiles → PUT method to update mobile Info

@Controller vs @RestController
@RequestBody vs HttpEntity
@ResponseBody
@PathVariable
@Getmapping
@PostMapping
@PutMapping
@DeleteMapping
@RequestParam
@RequestHeader
@ResponseStatus
@CookieValue
@ModelAttribute
@SessionAttribute
@RequestAttrbute
Single endpoint to give json and xml data
produces and consumes
content-type and accept
Spring MVC Complete Application (Profile Creation + Login + Dashboard + Logout)
Redirect vs Forward
Throwing Custom Exception Handling
ExceptionHandling in Micro services
DefaultExceptionHandler
@ResponseStatus - Exception level Exception Handling
@ExceptionHandler - Controller Level Exception Handling
@ControllerAdvise - Global Exception Handling
Request Pre -Post processing
Interceptors
Filters
Multiple filters configuration
Logging Response time
@CrossOrigin
Spring Boot Swagger Integration (Open API)
Web services, there are two development styles.
- Contract First
- Contract Last

Writing Client service to invoking mobile-service
mobile service client
RestTemplate
- getForObject
- getForEntity
- postForObject
- postForEntity

Spring Boot MVC Functional Endpoints

- RouterFunction
- HandlerFunction
- ServerRequest
- ServerResponse

Implement mobile-service -functional micro service

Spring Jdbc + Boot

Java JDBC API
Jdbc Template
NamedParameterJdbcTemplate
Country-Service Microservice

Spring Data JPA + Boot

Introduction
JDBC vs JPA
Repository
CrudRepository
JPARepository
Native Spring Data Jpa first Example
Spring Data JPA

- default methods in JPARepository & CRUDRepository

findBy - Method support in data jpa
SQL vs JPQL
Data Filtering using @Query
@Query
- Place Holders (?1)
- Named Parameters
- Native Query

Data Filtering in Data Jpa using @Query**
@NamedQuery
@NamedQueries
@NamedNativeQuery
Specifications

- JpaSpecificationExecutor
- Data Filtering With JPA Specifications

Sorting + Pagination

Validation + Boot

Bean Validation for Microservices
Loading Error messages from properties file
Handling Error Messages through Global Exception Handler

Returning Common Response from Micro services (Success & Failure cases)

Spring AOP + Boot

- Aspect
- Advice
    * @Before
    * @AfterReturning
    * @AfterThrowing
    * @After Finally
    * @Around
- PointCut
- JointPoint

@PointCut

Combining Pointcut Expressions

Pointcut Expression Examples

Implementing Logging Functionality in Micro Services using Spring AOP

Annotation Model Spring AOP

Spring Soap WebServices + Boot

- Contract First vs Contract Last
- SOAP vs REST
- First Spring Boot Soap Web services
- @Endpoint , @PayrootLoad, @RequestPayLoad, @ResponsePayLoad
- Exception Handling in Soap Web Services
- Client program to invoke Soap web service (WebServiceTemplate)
- Securing Spring Soap Web Services

Parent module

Common module

- Making Response type as generic
- Moving Common configuration into common module

Spring Reactive Programming + Boot

- What is Blocking API
- Why We need reactive programming
- Reactive stream API
- Reactive Stream API vs Java 9 Flow
- Project reactor vs RXJava
- Publisher , Subscriber , Subscription, Processor
- Mono , Flux introduction
- Project reactor standalone project example
- Operators on Mono
- Operators on Flux
- filter / map/ flatMap
- Creating Mono / Flux
- Subscribing Publisher (subscribe / block)

MongoDB installation on Cloud

Reactive Restful web service first example with Spring Web Flux + Mongo DB

Micro Service Development With Spring Reactive Programming

- CRUD Operations with Spring WebFlux

Spring Webflux Integration

- Open Api
- AOP Integration to log each method execution time
- ResponseTimeFilter
- Validations through properties file

Spring Security + Boot

- Authentication vs Authorization
- Default Security
- Custom User Name & Password with Default Security
- Configuring Users with InMemory Database
- Configuring Authorities with InMemory Database
- Http Basic authentication
- Http Form based authentication
- Spring Security + JDBC
    - DefaultSchema + H2 DB
    - Custom Schema + H2 DB
    - DefaultSchema + MySql
    - Custom Schema + H2 DB
- Spring Security + Data JPA
- JWT token based authentication

Spring Testing + Boot

- Unit Testing With Spring Testing
- @SpringBootTest
- @SpringBootTest vs @Runwith(SpringRunner.class)
- Unit Testing With Spring Testing Mockito
- Integration Testing using @SpringBoot
- @Mock
- @MockingBean

MicroServices + Spring Cloud + Spring Boot

- Advantages of Microservices Architectures
- Spring Boot Actuator API
- Connect Spring Cloud Config Server
    - Installing Git
    - Creating Local Git Repository
    - Connect Spring Cloud Config Server to Local Git Repository
    - Configuration for Multiple Environments in Git Repository

- Client-Side Load-Balancing with Spring Cloud LoadBalancer
- Introduction to API Gateways
    - Setting up Zuul API Gateway
    - Implementing Zuul Logging Filter
    - Executing a request through Zuul API Gateway
    - Setting up Zuul API Gateway between microservice invocations
- Introduction to Distributed Tracing
    - Implementing Spring Cloud Sleuth
    - Introduction to Distributed Tracing with Zipkin
- Understanding the need for Spring Cloud Bus
    - Implementing Spring Cloud Bus
- Fault Tolerance with Hystrix
- **MicroServices Centralized logging with ElasticSearch + Kibana**

100 + Hours Live Training
100% Live Coding