

Course duration

- 4 days

Course Benefits

- Create Spring Boot projects.
- Use databases and JPA in Spring Boot.
- Create RESTful services with Spring Boot.
- Deploy services that use Netflix Eureka, Hystrix, and Ribbon to create resilient and scalable services.

Available Delivery Methods

Public Class

Public expert-led online training from the convenience of your home, office or anywhere with an internet connection. Guaranteed to run .

Private Class

Private classes are delivered for groups at your offices or a location of your choice.

Course Outline

1. Introduction to the Spring Framework
 1. What is the Spring Framework?
 2. Spring Philosophies
 3. Why Spring?
 4. Spring Modules
 5. Requirements and Supported Environments
 6. Using Spring with Servers
 7. Role of Spring Container
 8. Spring Example
 9. Avoiding Dependency on Spring
 10. Additional Spring Projects/Frameworks
 11. Summary
2. Spring Annotation Configuration
 1. Spring Containers
 2. Annotation-based Spring Bean Definition
 3. Scanning for Annotation Components

4. Defining Component Scope Using Annotations
5. JSR-330 @Named Annotation
6. JSR-330 @Scope
7. Annotation-based Dependency Injection
8. Wiring Bean using @Inject
9. @Autowired - Constructor
10. @Autowired – Field
11. @Autowired - method
12. @Autowired – Collection
13. @Autowired – Maps
14. @Autowired & @Qualifier with Constructors, Fields, and Methods
15. @Autowired & Custom Qualifiers
16. @Autowired & Simple Custom Qualifier Field
17. @Autowired & Simple Custom Qualifier Method
18. @Autowired & CustomAutowireConfigurer
19. Dependency Injection Validation
20. @Resource
21. @PostConstruct and @PreDestroy
22. Summary
3. Spring Framework Configuration
 1. Java @Configuration Classes
 2. Defining @Configuration Classes
 3. Loading @Configuration Classes
 4. Modularizing @Configuration Classes
 5. Qualifying @Bean Methods
 6. Trouble with Prototype Scope
 7. Configuration with Spring Expression Language
 8. Resolving Text Messages
 9. Spring Property Conversion
 10. Spring Converter Interface
 11. Using Custom Converters
 12. Spring PropertyEditors
 13. Registering Custom PropertyEditors
 14. Summary
4. Introduction to Spring Boot
 1. What is Spring Boot?
 2. Spring Boot Main Features
 3. Spring Boot on the PaaS
 4. Understanding Java Annotations
 5. Spring MVC Annotations
 6. Example of Spring MVC-based RESTful Web Service
 7. Spring Booting Your RESTful Web Service
 8. Spring Boot Skeletal Application Example
 9. Converting a Spring Boot Application to a WAR File
 10. Externalized Configuration
 11. Starters
 12. The 'pom.xml' File

13. Spring Boot Maven Plugin
14. HOWTO: Create a Spring Boot Application
15. Summary
5. Spring MVC
 1. Spring MVC
 2. Spring Web Modules
 3. Spring MVC Components
 4. DispatcherServlet
 5. Template Engines
 6. Spring Boot MVC Example
 7. Spring MVC Mapping of Requests
 8. Advanced @RequestMapping
 9. Composed Request Mappings
 10. Spring MVC Annotation Controllers
 11. Controller Handler Method Parameters
 12. Controller Handler Method Return Types
 13. View Resolution
 14. Spring Boot Considerations
 15. Summary
6. Overview of Spring Boot Database Integration
 1. DAO Support in Spring
 2. Spring Data Access Modules
 3. Spring JDBC Module
 4. Spring ORM Module
 5. DataAccessException
 6. @Repository Annotation
 7. Using DataSources
 8. DAO Templates
 9. DAO Templates and Callbacks
 10. ORM Tool Support in Spring
 11. Summary
7. Using Spring with JPA or Hibernate
 1. Spring JPA
 2. Benefits of Using Spring with ORM
 3. Spring @Repository
 4. Using JPA with Spring
 5. Configure Spring Boot JPA EntityManagerFactory
 6. Application JPA Code
 7. "Classic" Spring ORM Usage
 8. Spring JpaTemplate
 9. Spring JpaCallback
 10. JpaTemplate Convenience Features
 11. Spring Boot Considerations
 12. Spring Data JPA Repositories
 13. Summary
8. Introduction to MongoDB
 1. MongoDB

2. MongoDB Features
3. MongoDB's Logo
4. Positioning of MongoDB
5. MongoDB Applications
6. MongoDB Data Model
7. MongoDB Limitations
8. MongoDB Use Cases
9. MongoDB Query Language (QL)
10. The CRUD Operations
11. The
12. find
13. Method
14. The
15. findOne
16. Method
17. A MongoDB QL Example
18. Data Inserts
19. MongoDB vs Apache CouchDB
20. Summary
9. Working with Data in MongoDB
 1. Reading Data in MongoDB
 2. The Query Interface
 3. Query Syntax is Driver-Specific
 4. Projections
 5. Query and Projection Operators
 6. MongoDB Query to SQL Select Comparison
 7. Cursors
 8. Cursor Expiration
 9. Writing Data in MongoDB
 10. An Insert Operation Example
 11. The Update Operation
 12. An Update Operation Example
 13. A Remove Operation Example
 14. Limiting Return Data
 15. Data Sorting
 16. Aggregating Data
 17. Aggregation Stages
 18. Accumulators
 19. An Example of an Aggregation Pipe-line
 20. Map-Reduce
 21. Summary
10. Spring Data with MongoDB
 1. Why MongoDB?
 2. MongoDB in Spring Boot
 3. Pom.xml
 4. Application Properties
 5. MongoRepository

6. Custom Query Methods
7. Supported Query Keywords
8. Complex Queries
9. Create JavaBean for Data Type
10. Using the Repository
11. Summary
11. Spring REST Services
 1. Many Flavors of Services
 2. Understanding REST
 3. RESTful Services
 4. REST Resource Examples
 5. REST vs SOAP
 6. REST Services With Spring MVC
 7. Spring MVC @RequestMapping with REST
 8. Working With the Request Body and Response Body
 9. @RestController Annotation
 10. Implementing JAX-RS Services and Spring
 11. JAX-RS Annotations
 12. Java Clients Using RestTemplate
 13. RestTemplate Methods
 14. Summary
12. Spring Security
 1. Securing Web Applications with Spring Security 3.0
 2. Spring Security 3.0
 3. Authentication and Authorization
 4. Programmatic v Declarative Security
 5. Getting Spring Security from Maven
 6. Spring Security Configuration
 7. Spring Security Configuration Example
 8. Authentication Manager
 9. Using Database User Authentication
 10. LDAP Authentication
 11. Summary
13. Spring JMS
 1. Spring JMS
 2. JmsTemplate
 3. Connection and Destination
 4. JmsTemplate Configuration
 5. Transaction Management
 6. Example Transaction Configuration
 7. Producer Example
 8. Consumer Example
 9. Converting Messages
 10. Message Listener Containers
 11. Message-Driven POJO's Async Receiver Example
 12. Message-Driven POJO's Async Receiver Configuration
 13. Spring Boot Considerations

14. Summary

14. Microservices

1. What is a "Microservice"?
2. One Helpful Analogy
3. SOA - Microservices Relationship
4. ESB - Microservices Relationship
5. Traditional Monolithic Designs and Their Role
6. Disadvantages of Monoliths
7. Moving from a Legacy Monolith
8. When Moving from a Legacy Monolith
9. The Driving Forces Behind Microservices
10. How Can Microservices Help You?
11. The Microservices Architecture
12. Utility Microservices at AWS
13. Microservices Inter-connectivity
14. The Data Exchange Interoperability Consideration
15. Managing Microservices
16. Implementing Microservices
17. Embedding Databases in Java
18. Microservice-Oriented Application Frameworks and Platforms
19. Summary

15. Spring Cloud Config

1. The Spring Cloud Configuration Server
2. Why Configuration Management is Important
3. Configuration Management Challenges in Microservices
4. Separation of Configuration from Code
5. Configuration Service
6. How the Configuration Service Works
7. Cloud Config Server Properties File
8. Git Integration
9. Properties
10. Configuration Client
11. Sample Client Config File
12. Sample Client Application
13. Dynamic Property Updates – Server
14. Dynamic Property Update – Client
15. Dynamic Property Update – Execute
16. Summary

16. Service Discovery with Netflix Eureka

1. Service Discovery in Microservices
2. Load Balancing in Microservices
3. Netflix Eureka
4. Eureka Architecture
5. Communications in Eureka
6. Time Lag
7. Eureka Deployment
8. Peer Communication Failure between Servers

9. Eureka Server Configuration
10. Eureka Client/Service
11. Eureka Client Properties
12. Spring Cloud DiscoveryClient Interface
13. ServiceInstance JSON
14. ServiceInstance Interface
15. What about Services
16. Eureka and the AWS Ecosystem
17. Summary
17. Load-Balancing with Netflix Ribbon
 1. Load Balancing in Microservices
 2. Netflix Ribbon
 3. Server-side load balance
 4. Client-side Load Balance
 5. Architecture
 6. Load Balance Rules
 7. RoundRobinRule
 8. AvailabilityFilteringRule
 9. WeightedResponseTimeRule
 10. RandomRule
 11. ZoneAvoidanceRule
 12. IPing Interface (Failover)
 13. Using Ribbon
 14. YAML Configuration
 15. Configuration Class
 16. Client Class
 17. Client Class Implementation
 18. Integration with Eureka (Service Discovery)
 19. Using Ribbon in the Amazon AWS Cloud
 20. Summary
18. Application Hardening with Netflix Hystrix
 1. Netflix Hystrix
 2. Design Principles
 3. Design Principles (continued)
 4. Cascading Failures
 5. Bulkhead Pattern
 6. Circuit Breaker Pattern
 7. Thread Pooling
 8. Request Caching
 9. Request Collapsing
 10. Fail-Fast
 11. Fallback
 12. Using Hystrix
 13. Circuit Breaker Configuration
 14. Fallback Configuration
 15. Collapser Configuration
 16. Rest Controller and Handler

- 17. Collapser Service (Part 1)
- 18. How the Collapser Works
- 19. Hystrix Monitor
- 20. Enable Monitoring
- 21. Turbine
- 22. The Monitor
- 23. Monitor details
- 24. Summary
- 19. Edge Components with Netflix Zuul
 - 1. Zuul is the Gatekeeper
 - 2. Request Handling
 - 3. Filters
 - 4. Filter Architecture
 - 5. Filter Properties
 - 6. filterType()
 - 7. filterOrder()
 - 8. shouldFilter()
 - 9. Run()
 - 10. Cancel Request
 - 11. Dynamic Filter Loading
 - 12. Filter Communications
 - 13. Routing with Eureka and Ribbon
 - 14. Summary
- 20. Distributed Tracing with Zipkin
 - 1. Zipkin
 - 2. Zipkin Features
 - 3. Architecture
 - 4. The Collector
 - 5. Storage
 - 6. API
 - 7. GUI Console
 - 8. Zipkin Console Homepage
 - 9. View a Trace
 - 10. Trace Details
 - 11. Dependencies
 - 12. Dependency Details
 - 13. Zipkin in Spring Boot
 - 14. Zipkin Configuration
 - 15. Summary

Class Materials

Each student will receive a comprehensive set of materials, including course notes and all the class examples.

Class Prerequisites

Experience in the following *is required* for this Microservices class:

- Experience with Java development.