

**Microservices with Spring Boot and Spring Cloud Training**

**Course Number:** SPRG-216WA
**Duration:** 4 days

**Overview**

Spring Boot is a platform for building secure and scalable applications that can be deployed independently with an embedded web server. When combined with Spring Cloud, developers can quickly develop robust microservices-based architectures without compromising security or scalability.

This Spring Boot and Spring Cloud with Microservices training course teaches developers how to use Spring Boot, Spring Cloud, and the Netflix OSS suite to deploy highly resilient and scalable RESTful services and web applications.

**Prerequisites**

All attendees must have Java development experience.

**Materials**

All Spring Boot and Spring Cloud training attendees receive comprehensive courseware covering all topics in the course.

**Software Needed on Each Student PC**

For in-person deliveries, attendees require computers with the Java Development Kit version 8 and an internet connection. Students also require a Java Integrated Development Environment like Eclipse or IntelliJ. We will provide full classroom setup instructions that will include seating in small groups, with supplies such as flipcharts, sticky notes, markers, and pens for the attendees and a projector and Internet connection for the instructor's laptop.

Online deliveries for this interactive training will use an online meeting platform (such as Zoom, WebEx, GoTo, or Teams) to have face-to-face contact online, including use of breakout rooms for group activities.

**Objectives**

* Create Spring Boot projects
* Persist data in SQL and non-SQL databases using 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

**Outline**

* Introduction to the Spring Framework
	+ What is the Spring Framework?
	+ Spring Philosophies
	+ Why Spring?
	+ Spring Modules
	+ Requirements and Supported Environments
	+ Using Spring with Servers
	+ Role of Spring Container
	+ Spring Example
	+ Avoiding Dependency on Spring
	+ Additional Spring Projects/Frameworks
* Spring Annotation Configuration
	+ Spring Containers
	+ Annotation-based Spring Bean Definition
	+ Scanning for Annotation Components
	+ Defining Component Scope Using Annotations
	+ JSR-330 @Named Annotation
	+ JSR-330 @Scope
	+ Annotation-based Dependency Injection
	+ Wiring Bean using @Inject
	+ @Autowired
		- @Autowired – Constructor
		- @Autowired – Field
		- @Autowired – method
		- @Autowired – Collection
		- @Autowired – Maps
		- @Autowired & @Qualifier with Constructors, Fields, and Methods
		- @Autowired & Custom Qualifiers
		- @Autowired & Simple Custom Qualifier Field
		- @Autowired & Simple Custom Qualifier Method
		- @Autowired & CustomAutowireConfigurer
	+ Dependency Injection Validation
	+ @Resource
	+ @PostConstruct and @PreDestroy
* Spring Framework Configuration
	+ Java @Configuration Classes
	+ Defining @Configuration Classes
	+ Loading @Configuration Classes
	+ Modularizing @Configuration Classes
	+ Qualifying @Bean Methods
	+ The Trouble with Prototype Scope
	+ Configuration with Spring Expression Language
	+ Resolving Text Messages
	+ Spring Property Conversion
	+ Spring Converter Interface
	+ Using Custom Converters
	+ Spring PropertyEditors
	+ Registering Custom PropertyEditors
* Introduction to Spring Boot
	+ What is Spring Boot?
	+ Spring Framework
	+ How is Spring Boot Related to Spring Framework?
	+ Spring Boot 2
	+ Spring Boot Main Features
	+ Spring Boot on the PaaS
	+ Understanding Java Annotations
	+ Spring MVC Annotations
	+ Example of Spring MVC-based RESTful Web Service
	+ Spring Booting Your RESTful Web Service
	+ Spring Boot Skeletal Application Example
	+ Converting a Spring Boot Application to a WAR File
	+ Externalized Configuration
	+ Starters
	+ Maven - The 'pom.xml' File
	+ Spring Boot Maven Plugin
	+ Gradle - The 'build.gradle' File
	+ Spring Boot Maven Plugin
	+ Create a Spring Boot Application
* Spring MVC
	+ Spring MVC
	+ Spring Web Modules
	+ Spring MVC Components
	+ DispatcherServlet
	+ Spring WebFlux Module
	+ Spring WebFlux
	+ Template Engines
	+ Spring Boot MVC Example
	+ Spring MVC Mapping of Requests
	+ Advanced @RequestMapping
	+ Composed Request Mappings
	+ Spring MVC Annotation Controllers
	+ Controller Handler Method Parameters
	+ Controller Handler Method Return Types
	+ View Resolution
	+ Spring Boot Considerations
* Overview of Spring Boot Database Integration
	+ DAO Support in Spring
	+ Spring Data Access Modules
	+ Spring JDBC Module
	+ Spring ORM Module
	+ DataAccessException
	+ @Repository Annotation
	+ Using DataSources
	+ DAO Templates
	+ DAO Templates and Callbacks
	+ ORM Tool Support in Spring
* Using Spring with JPA or Hibernate
	+ Spring JPA
	+ Benefits of Using Spring with ORM
	+ Spring @Repository
	+ Using JPA with Spring
	+ Configure Spring Boot JPA EntityManagerFactory
	+ Application JPA Code
	+ "Classic" Spring ORM Usage
	+ Spring JpaTemplate
	+ Spring JpaCallback
	+ JpaTemplate Convenience Features
	+ Spring Boot Considerations
	+ Spring Data JPA Repositories
	+ Database Schema Migration
	+ Database Schema Migration for CI/CD using Liquibase
	+ How Liquibase Works?
	+ Changelogs in Liquibase
	+ Preconditions in Changelogs
	+ Sample Empty Changelog
	+ Sample Precondition in Changelog
	+ Sample Changeset in Changelog
	+ Running Liquibase
	+ Liquibase Commands
* Introduction to MongoDB
	+ MongoDB Features
	+ MongoDB’s Logo
	+ Positioning of MongoDB
	+ MongoDB Applications
	+ MongoDB Data Model
	+ MongoDB Limitations
	+ MongoDB Use Cases
	+ MongoDB Query Language (QL)
	+ The CRUD Operations
		- The
		- find
		- Method
		- The
		- findOne
		- Method
	+ A MongoDB QL Example
	+ Data Inserts
	+ MongoDB vs. Apache CouchDB
* Working with Data in MongoDB
	+ Reading Data in MongoDB
	+ The Query Interface
	+ Query Syntax is Driver-Specific
	+ Projections
	+ Query and Projection Operators
	+ MongoDB Query to SQL Select Comparison
	+ Cursors
	+ Cursor Expiration
	+ Writing Data in MongoDB
	+ An Insert Operation Example
	+ The Update Operation
	+ An Update Operation Example
	+ A Remove Operation Example
	+ Limiting Return Data
	+ Data Sorting
	+ Aggregating Data
	+ Aggregation Stages
	+ Accumulators
	+ An Example of an Aggregation Pipe-line
	+ Map-Reduce
* Spring Data with MongoDB
	+ Why MongoDB?
	+ MongoDB in Spring Boot
	+ Pom.xml
	+ Application Properties
	+ MongoRepository
	+ Custom Query Methods
	+ Supported Query Keywords
	+ Complex Queries
	+ Create JavaBean for Data Type
	+ Using the Repository
* Spring REST Services
	+ Many Flavors of Services
	+ Understanding REST
	+ RESTful Services
	+ REST Resource Examples
	+ @RestController Annotation
	+ Implementing JAX-RS Services and Spring
	+ JAX-RS Annotations
	+ Java Clients Using RestTemplate
	+ RestTemplate Methods
* Spring Security
	+ Securing Web Applications with Spring Boot 2
	+ Spring Security
	+ Authentication and Authorization
	+ Programmatic vs. Declarative Security
	+ Getting Spring Security Gradle or Maven
	+ Spring Security Configuration
	+ Spring Security Configuration Example
	+ Authentication Manager
	+ Using Database User Authentication
	+ LDAP Authentication
	+ What is Security Assertion Markup Language (SAML)?
	+ What is a SAML Provider?
	+ Spring SAML2.0 Web SSO Authentication
	+ Setting Up an SSO Provider
	+ Adding SAML Dependencies to a Project
	+ Dealing with the State
	+ How Can I Maintain State?
	+ SAML vs. OAuth2
	+ OAuth2 Overview
		- OAuth – Facebook Sample Flow
		- OAuth Versions
		- OAuth2 Components
		- OAuth2 – End Points
		- OAuth2 – Tokens
		- OAuth – Grants
	+ Authenticating Against an OAuth2 API
		- OAuth2 using Spring Boot – Dependencies
		- OAuth2 using Spring Boot – application.yml
		- OAuth2 using Spring Boot – Main Class
		- OAuth2 using Spring Boot – SPA Client
	+ JSON Web Tokens
	+ How JWT Works
		- JWT Header
		- JWT Payload
		- JWT Example Payload
		- JWT Example Signature
		- How JWT Tokens are Used
		- Adding JWT to HTTP Header
		- How The Server Makes Use of JWT Tokens
	+ What are “Scopes”?
	+ JWT with Spring Boot – Dependencies
	+ JWT with Spring Boot – Main Class
* Spring JMS
	+ JmsTemplate
	+ Connection and Destination
	+ JmsTemplate Configuration
	+ Transaction Management
	+ Example Transaction Configuration
	+ Producer Example
	+ Consumer Example
	+ Converting Messages
	+ Message Listener Containers
	+ Message-Driven POJO's Async Receiver Example
	+ Message-Driven POJO's Async Receiver Configuration
	+ Spring Boot Considerations
* Microservices
	+ What is a “Microservice”?
	+ One Helpful Analogy
	+ SOA – Microservices Relationship
	+ ESB – Microservices Relationship
	+ Traditional Monolithic Designs and Their Role
	+ Disadvantages of Monoliths
	+ Moving from a Legacy Monolith
	+ When Moving from a Legacy Monolith
	+ The Driving Forces Behind Microservices
	+ How Can Microservices Help You?
	+ The Microservices Architecture
	+ Utility Microservices at AWS
	+ Microservices Inter-connectivity
	+ The Data Exchange Interoperability Consideration
	+ Managing Microservices
	+ Implementing Microservices
	+ Embedding Databases in Java
	+ Microservice-Oriented Application Frameworks and Platforms
* Spring Cloud Config
	+ The Spring Cloud Configuration Server
	+ Why Configuration Management is Important
	+ Configuration Management Challenges in Microservices
	+ Separation of Configuration from Code
	+ Configuration Service
	+ How the Configuration Service Works
	+ Cloud Config Server Properties File
	+ Git Integration
	+ Properties
	+ Configuration Client
	+ Sample Client Config File
	+ Sample Client Application
	+ Dynamic Property Updates
* Service Discovery with Netflix Eureka
	+ Service Discovery in Microservices
	+ Load Balancing in Microservices
	+ Netflix Eureka
	+ Eureka Architecture
	+ Communications in Eureka
	+ Time Lag
	+ Eureka Deployment
	+ Peer Communication Failure between Servers
	+ Eureka Server Configuration
	+ Eureka Client/Service
	+ Eureka Client Properties
	+ Spring Cloud DiscoveryClient Interface
	+ ServiceInstance JSON
	+ ServiceInstance Interface
	+ What about Services
	+ Eureka and the AWS Ecosystem
* Load-Balancing with Netflix Ribbon
	+ Load Balancing in Microservices
	+ Netflix Ribbon
	+ Server-side load balance
	+ Client-side Load Balance
	+ Architecture
	+ Load Balance Rules
	+ RoundRobinRule
	+ AvailabilityFilteringRule
	+ WeightedResponseTimeRule
	+ RandomRule
	+ ZoneAvoidanceRule
	+ IPing Interface (Failover)
	+ Using Ribbon
	+ YAML Configuration
	+ Configuration Class
	+ Client Class
	+ Client Class Implementation
	+ Integration with Eureka (Service Discovery)
	+ Using Ribbon in the Amazon AWS Cloud
* Application Hardening with Netflix Hystrix
	+ Netflix Hystrix
	+ Design Principles
	+ Design Principles (continued)
	+ Cascading Failures
	+ Bulkhead Pattern
	+ Circuit Breaker Pattern
	+ Thread Pooling
	+ Request Caching
	+ Request Collapsing
	+ Fail-Fast
	+ Fallback
	+ Using Hystrix
	+ Circuit Breaker Configuration
	+ Fallback Configuration
	+ Collapser Configuration
	+ Rest Controller and Handler
	+ Collapser Service (Part 1)
	+ How the Collapser Works
	+ Hystrix Monitor
	+ Enable Monitoring
	+ Turbine
	+ The Monitor
	+ Monitor details
* Edge Components with Netflix Zuul
	+ Zuul is the Gatekeeper
	+ Request Handling
	+ Filters
	+ Filter Architecture
	+ Filter Properties
	+ filterType()
	+ filterOrder()
	+ shouldFilter()
	+ Run()
	+ Cancel Request
	+ Dynamic Filter Loading
	+ Filter Communications
	+ Routing with Eureka and Ribbon
* Distributed Tracing with Zipkin
	+ Zipkin Features
	+ Architecture
	+ The Collector
	+ Storage
	+ API
	+ GUI Console
	+ Zipkin Console Homepage
	+ View a Trace
	+ Trace Details
	+ Dependencies
	+ Dependency Details
	+ Zipkin in Spring Boot
	+ Zipkin Configuration
* Conclusion