

**Technical Introduction to Microservices**

**Course Number:** MSV-108WA  
**Duration:** 2 days

**Overview**

This Introduction to Microservices training course teaches attendees the role and mechanics of microservices. Participants gain practical experience implementing microservices using different technology stacks and understand how Docker, RESTful Web Services, Node.js, Spring Boot, Kubernetes, and OpenShift support the deployment of microservices.

**Prerequisites**

All attendees must have basic knowledge of a modern programming language and software design principles.

**Materials**

All Microservices training students receive comprehensive courseware.

**Software Needed on Each Student PC**

Attendees will not need to install any software on their computer for this class. The class will be conducted in a remote environment that Accelebrate will provide; students will only need a local computer with a web browser and a stable Internet connection. Any recent version of Microsoft Edge, Mozilla Firefox, or Google Chrome will be fine.

**Objectives**

* Monolith vs. microservices design
* Work with Node.js and Spring Boot
* Enable basic security
* Use AWS Lambda
* Incorporate Angular communication with REST services
* Create a Docker account and obtain an access token
* Get started with Docker & Kubernetes
* Work with CI/CD using Jenkins, Docker, and OpenShift

**Outline**

* Breaking Up Monoliths – Pros and Cons
  + Traditional Monolithic Applications and Their Place
  + Disadvantages of Monoliths
  + Developer's Woes
  + Architecture Modernization
  + Microservices Architecture is Not a Silver Bullet!
  + What May Help?
  + In-Class Discussion
* Microservice Development
  + What are Microservices?
  + Microservices vs. Classic SOA
  + Principles of Microservices Architecture Design
  + Domain-Driven Design
  + Domain-Driven Design - Benefits
  + Microservices and Domain-Driven Design
  + Designing for failure
  + Microservices Architecture – Pros and cons
  + Docker and Microservices
  + Microservice Deployment with Docker – Workflow
  + Writing Dockerfile
  + Kubernetes
  + What is OpenShift
  + OpenShift Architecture
  + Microservices and Various Applications
  + Web Applications
  + Single Page Applications
  + Traditional Enterprise Application Architecture
  + Sample Microservices Architecture
  + Serverless & Event-driven Microservice – AWS Lambda
* Twelve-factor Applications
  + Twelve-factor Applications
  + Twelve Factors, Microservices, and App Modernization
  + The Twelve Factors
  + Categorizing the 12 Factors
  + Kubernetes and the Twelve Factors
* REST Services
  + Many Flavors of Services
  + Understanding REST
  + Principles of RESTful Services
  + REST Examples
  + SOAP Equivalent Examples
  + JSON
  + Famous RESTful Services
  + Additional Resources
  + What is gRPC?
  + Protocol Buffers
  + REST vs. gRPC
  + Protobuf vs. JSON
  + HTTP/2 vs. HTTP 1.1
  + Messages vs. Resources and Verbs
  + Streaming vs. Request-Response
  + Strong Typing vs. Serialization
  + Web Browser Support
  + REST vs. gRPC – In a Nutshell
* Microservices with Node.js
  + What is Node.js?
  + Node's Value Proposition
  + Example of a Node.js App: A Simple Web Server
  + Node.js Project Types
  + Managing Large Applications
  + Core Modules
  + Why Node.js uses JavaScript?
  + The Traditional Concurrency Support Model
  + Disadvantages of the Traditional Approach
  + Event-Driven, Non-Blocking I/O
  + The Success Callback Function
  + Using Node Package Manager (NPM)
  + NPM Registry (Repository)
  + NPM Enterprise
  + Package Life-Cycle Management
  + Local and Global Package Installation Options
  + Listing and Using Module Versions
  + The Express Package
  + Installing and Using Express
  + Defining Routing Rules in Express
  + Route Path
  + The Response Object
  + A Simple Web Service with Express Example
  + The MEAN Stack
* Introduction to Spring Boot for Non-Java Developers
  + What is Spring Boot?
  + Spring Boot Main Features
  + Spring Boot vs DropWizard
  + 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
  + Summary
* Spring REST Services
  + Many Flavors of Services
  + Understanding REST
  + RESTful Services
  + REST Resource Examples
  + REST vs SOAP
  + REST Services With Spring MVC
  + Spring MVC @RequestMapping with REST
  + Working With the Request Body and Response Body
  + @RestController Annotation
  + Implementing JAX-RS Services and Spring
  + JAX-RS Annotations
  + Java Clients Using RestTemplate
  + RestTemplate Methods
* Spring Security
  + Securing Web Applications with Spring Security 3.0
  + Spring Security 3.0
  + Authentication and Authorization
  + Programmatic v 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
  + Authenticating Against an OAuth2 API
  + OAuth2 using Spring Boot
  + JSON Web Tokens
  + JSON Web Token Architecture
  + 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
* AWS Lambda
  + What is AWS Lambda?
  + Supported Languages
  + Getting Your Code Up And Running in Lambda
  + Examples of the Base Lambda Function
  + Use Cases
  + How It Works
  + Example: Processing S3 Source Events with Lambda
  + The Programming Model
  + Configuring Lambda Functions
  + Configure Triggers Page
  + Lambda Function Blueprints
  + How Do I Troubleshoot and Monitor My Lambda Functions?
  + Developing Lambda in Java
* Consuming REST Services from a Client
  + Accessing REST Services using jQuery
  + Accessing REST Services in React
  + The Angular HTTP Client
  + Using The HTTP Client - Overview
  + Importing HttpClientModule
  + Simple Example
  + Service Using HttpClient
  + ES6 Import Statements
  + Making a GET Request
  + What does an Observable Object do?
  + Using the Service in a Component
  + The PeopleService Client Component
  + Error Handling
  + Making a POST Request
  + Making a PUT Request
  + Making a DELETE Request
* Docker Introduction
  + What is Docker
  + Where Can I Run Docker?
  + Installing Docker Container Engine
  + Docker Machine
  + Docker and Containerization on Linux
  + Linux Kernel Features: cgroups and namespaces
  + The Docker-Linux Kernel Interfaces
  + Docker Containers vs Traditional Virtualization
  + Docker Containers vs Traditional Virtualization
  + Docker Integration
  + Docker Services
  + Docker Application Container Public Repository
  + Competing Systems
  + Docker Command Line
  + Starting, Inspecting, and Stopping Docker Containers
  + Docker Volume
  + Dockerfile
  + Docker Compose
  + Using Docker Compose
  + Dissecting docker-compose.yml
  + Specifying services
  + Dependencies between containers
  + Injecting Environment Variables
  + runC Overview
  + runC Features
  + Using runC
  + Running a Container using runC
* Introduction to Kubernetes
  + What is Kubernetes?
  + What is a Container?
  + Containers – Uses, Pros, and Cons
  + Composition of a Container
  + Control Groups
  + Namespaces
  + Union Filesystems
  + Popular Containerization Software
  + Microservices
  + Microservices and Containers / Clusters
  + Microservices and Orchestration
  + Microservices and Infrastructure-as-Code
  + Kubernetes Container Networking
  + Kubernetes Networking Options
  + Kubernetes Networking – Balanced Design
* CI/CD with OpenShift, Jenkins, and Blue Ocean
  + What is OpenShift?
  + OpenShift Online
  + OpenShift Origin
  + OpenShift Architecture
  + OpenShift Origin Installation
  + OpenShift CLI
  + Jenkins Continuous Integration
  + Jenkins Features
  + Running Jenkins
  + Downloading and Installing Jenkins
  + Running Jenkins as a Stand-Alone Application
  + Running Jenkins on an Application Server
  + Installing Jenkins as a Windows Service
  + Different types of Jenkins job
  + Configuring Source Code Management (SCM)
  + Working with Subversion
  + Working with Git
  + Build Triggers
  + Schedule Build Jobs
  + Polling the SCM
  + Maven Build Steps
  + Jenkins / OpenShift Pipeline
  + Jenkins / OpenShift Pipeline Output
  + Installing Jenkins Plugins
  + The Blue Ocean Plugin
  + Blue Ocean Plugin Features
  + New modern user experience
  + Advanced Pipeline visualizations with built-in failure diagnosis
  + Branch and Pull Request awareness
  + Personalized View
  + OpenShift Pipeline Output
  + Creating OpenShift Blue Ocean Pipeline
  + Summary
* Appendix A: Leading Practices for Microservice Logging
  + Logging Challenges
  + Leading Practices
  + Correlate Requests with a Unique ID
  + Include a Unique ID in the Response
  + Send Logs to a Central Location
  + Structure Your Log Data
  + Add Context to Every Record
  + Examples of Content
  + Write Logs to Local Storage
  + Collecting Logs with Fluentd
  + Leading Practices for Microservice Logging Summary
  + Metrics Using Prometheus
  + Overview
  + Prometheus Architecture
  + Service Discovery
  + File-based Service Discovery
  + Istio and Prometheus
  + Exposing Metrics in Services
  + Querying in Prometheus
  + Grafana
  + Business Metrics
  + Metrics Using Prometheus Summary
  + Tracing Using Jaeger
  + OpenTracing
  + Jaeger Architecture Diagram
  + Jaeger Client Libraries
  + Jaeger Sampling
  + Jaeger Agent
  + Jaeger Collector
  + Query and Ingester Services
  + Jaeger UI Example
  + Jaeger and Prometheus
  + Jaeger and Istio
  + Tracing Using Jaeger Summary
* Appendix B: Traffic Routing Patterns
  + Edge Proxy Server
  + Request Handling
  + Filters
  + Filter Architecture
  + API Gateway for Routing Requests
  + API Gateway – Example
  + Rate Limiting
  + Rate Limiting – Business Cases
  + Configuring Rate Limiting in NGINX
  + Circuit Breaker
  + Design Principles
  + Cascading Failures
  + Bulkhead Pattern
  + Circuit Breaker Pattern
  + Thread Pooling
  + Request Caching
  + Request Collapsing
  + Fail-Fast
  + Fallback
  + Circuit Breaker Solutions
  + Load Balancing in Microservices
  + Server-side load balance
  + Client-side Load Balance
  + Architecture
  + Service Mesh
  + Service Mesh Solutions
  + Content Delivery Network (CDN)
  + How does a CDN Work?
  + Benefits of using a CDN
  + CDN Solutions
* Conclusion