Specialized - Microservices Training: Technical Introduction

Code:	INTRO-MS
Length:	2 days
URL:	View Online

This Introduction to Microservices training course will help you understand the value proposition and technical aspects of microservices, a new and rather fuzzy concept used to describe rapidly provisionable, independently deployable services with narrow and distinct functionality. For IT professionals, developers, software engineers, and DevOps practitioners – our microservices training provides the technical practices and tooling fundamentals necessary to begin realizing the benefits of microservices as a foundation for IT architecture, software engineering, and service/release delivery.

Skills Gained

Upon completion of this Microservices training course, students will have fundamental understanding of microservices and practical experience in implementing microservices using different technology stacks.

Who Can Benefit

IT Architects, Software Designers, and Developers

Prerequisites

Foundational knowledge of programming and software design principles.

Course Details

Outline of Technical Introduction To Microservices Training

Chapter 1. Breaking Up Monoliths – Pros and Cons

- Traditional Monolithic Applications and Their Place
- Disadvantages of Monoliths
- Developer’s Woes
- Architecture Modernization
- Architecture Modernization Challenges
- Microservices Architecture is Not a Silver Bullet!
- What May Help?
- In-Class Discussion
- Summary

Chapter 2. 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
- Microservices Architecture – Cons
- Docker and Microservices
- Microservice Deployment with Docker – Workflow
- Writing Dockerfile
- Kubernetes
- What is OpenShift
- OpenShift Architecture
- Microservices and Various Applications
- Web Applications
- Web Applications – Reference Architecture
- Web Applications – When to use?
- Single Page Applications
- Single Page Applications – Benefits
- Traditional Enterprise Application Architecture
- Sample Microservices Architecture
- Serverless & Event-driven Microservice – AWS Lambda
- Summary

Chapter 3. Twelve-factor Applications

- Twelve-factor Applications
- Twelve Factors, Microservices, and App Modernization
- The Twelve Factors
- Categorizing the 12 Factors
- 12-Factor Microservice Codebase
- 12-Factor Microservice Dependencies
- 12-Factor Microservice Config
- 12-Factor Microservice Backing Services
- 12-Factor Microservice Build, Release, Run
- 12-Factor Microservice Processes
- 12-Factor Microservice Port Binding
- 12-Factor Microservice Concurrency
- 12-Factor Microservice Disposability
- 12-Factor Microservice Dev/Prod Parity

Chapter 5. 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 ExampleThe MEAN Stack
- Summary

Chapter 6. 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

Chapter 7. 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 AnnotationImplementing JAX-RS Services and Spring
- JAX-RS Annotations
- Java Clients Using RestTemplate
- RestTemplate Methods
- Summary

Chapter 8. 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 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
- JSON Web Token Architecture
- How JWT Works
- JWT Header
- JWT PayloadJWT Example Payload
- JWT Example Signature
- How JWT Tokens are Used
- Adding JWT to HTTP Header
- How The Server Makes Use of JWT TokensWhat are “Scopes”?
- JWT with Spring Boot – Dependencies
- JWT with Spring Boot – Main Class
- Summary

Chapter 9. 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
- Summary

Chapter 10. Consuming REST Services from a Client

- Accessing REST Services using jQuery – GET Example
- Accessing REST Services using jQuery – GET Example (Contd.)
- Accessing REST Services using jQuery – POST Example
- Accessing REST Services in React – Component
- Accessing REST Services in React – componentDidMount
- Accessing REST Services in React – render

- Accessing REST Services in React – POST Method
- 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
- Summary

Chapter 11. 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 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
- Summary
- Chapter 12. Introduction to Kubernetes
- What is Kubernetes
- What is a Container
- Container – Uses
- Container – ProsContainer – 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
- Summary

Chapter 13. CI/CD with OpenShift, Jenkins, and Blue Ocean

- What is OpenShift
- OpenShift Online
- OpenShift Origin
- OpenShift Architecture
- OpenShift Origin Installation
- OpenShift CLI
- OpenShift CLI (Contd.)
- 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 Subversion (cont'd)
- 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

Chapter 14. 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
- Prometheus Architecture
- Service Discovery
- File-based Service Discovery
- Istio and Prometheus
- Exposing Metrics in Services
- Querying in Prometheus

- Circuit Breaker Solutions
- Load Balancing in Microservices
- Server-side load balance
- Client-side Load Balance
- Architecture
- Service Mesh
- Service Mesh (Contd.)
- Service Mesh Solutions
- Content Delivery Network (CDN)
- How does a CDN Work?
- Benefits of using a CDN
- CDN Solutions
- Summary

Lab Exercises

- Lab 1. Monolith vs Microservices Design
- Lab 2. Getting Started With Node.js
- Lab 3. Getting Started with Spring Boot
- Lab 4. Enable Basic Security
- Lab 5. Using AWS Lambda
- Lab 6. Getting Started with Docker
- Lab 7. Getting Started with Docker Compose
- Lab 8. Getting Started with Kubernetes
- Lab 9. CI/CD with Jenkins, Docker, and OpenShift

Schedule (as of 4)

Date	Location