**Microservices using Eclipse MicroProfile**

**Duration: 5 Days**

**Prerequisites:**

1. **Working Java 8 knowledge is required. At least 5-6 years of Java/JEE experience.**
2. **Familiarity with developing RESTful services is required.**
3. **Experience working with Docker containers or other similar containers**
4. **Familiarity with Spring Framework, or similar enterprise Java framework is a plus.**

**What you will learn:**

1. **Understand why microservices are important in the digital economy**
2. **Analyze how MicroProfile addresses the need for enterprise Java microservices**
3. **Test and secure your applications with Eclipse MicroProfile**
4. **Get to grips with various MicroProfile capabilities such as OpenAPI and Typesafe REST Client**
5. **Discover and implement coding best practices using MicroProfile**
6. **Develop, Deploy and monitor a MicroProfile microservice**
7. **Develop, Deploy and monitor a containerized MicroProfile microservice**
8. **Build Cloud Native Microservices using MicroProfile specifications**
9. **Understand difference between Spring Boot and Eclipse MicroProfile**
10. **How to secure a RESTful endpoint**
11. **How to manage failure conditions by applying various fault tolerance patterns**
12. **How to expose application health**
13. **Define and expose application metrics with MicroProfile metrics**
14. **Create an uberjar and native executable microservice**
15. **Prove application portability by deploying the microservice using a different MicroProfile implementation**

**Course Outline**

**Day 1**

**What are Microservices?**

* Introduction to Microservices
* [Monolithic architecture](http://microservices.io/patterns/monolithic.html) v/s [Micro Service architecture](http://microservices.io/patterns/microservices.html)
* What is decomposition and why do I care
* Decompose by business capabilities
* Decompose by subdomain
* I am a developer Why do I care
* I am a Consumer why do I care
* I am an Enterprise why do I care

**How are Microservices deployed?**

* [Multiple service instances per host](http://microservices.io/patterns/deployment/multiple-services-per-host.html)
* [Service instance per host](http://microservices.io/patterns/deployment/single-service-per-host.html)
* [Service instance per VM](http://microservices.io/patterns/deployment/service-per-vm.html)
* [Service instance per Container](http://microservices.io/patterns/deployment/service-per-container.html)
* [Serverless deployment](http://microservices.io/patterns/deployment/serverless-deployment.html)
* [Service deployment platform](http://microservices.io/patterns/deployment/service-deployment-platform.html)-new

**Introduction to Eclipse MicroProfile**

* Enterprise Java Microservices
* What is Eclipse MicroProfile
* How MicroProfile differs from Spring Boot based Microservices
* Why industry is adopting MicroProfile as a first choice
* Introducing Eclipse MicroProfile
* MicroProfile value proposition
* Create BackendService with a JAX-RS endpoint
* Running Microservice on Open Liberty/Apache TomEE/Other Runtime
* Other MicroProfile runtimes available

**MicroProfile Config**

* Introduction to MicroProfile Config
* Reading configuration using MicroProfile Config API
* The Config object
* The @ConfigProperty annotation
* Providing sources of configuration
* Default ConfigSources
* Configure BackendService using MicroProfile config

**MicroProfile REST Client**

* Introduction to the MicroProfile REST Client
* Defining the endpoint Interface
* MicroProfile REST Client programmatic API usage
* MicroProfile REST Client CDI usage
* MicroProfile Config integration
* Simplifying configuration keys
* Dealing with client headers
* Create a FrontendService which invokes BackendService using the MicroProfile Rest Client

**Day 2**

**MicroProfile OpenAPI**

* Introduction to MicroProfile OpenAPI
* Configuration
* Generating the OpenAPI document
* MicroProfile OpenAPI annotations

**MicroProfile Reactive**

* Why reactive?
* MicroProfile relation to reactive programming
* Why and When reactive programming is beneficial
* Reactive Manifesto
* Reactive Pipeline
* Reactive Programming in Microservices With MicroProfile
* MicroProfile Reactive Features
  + Reactive REST client (via JAX-RS)
  + Server-Sent Events (SSE) server and client (via JAX-RS)
  + Asynchronous events (via CDI)
  + Metrics and Health endpoints to react to resource usage and service failures
* MicroProfile brings common APIs
* Collaboration of vendors and library creators
* Reactive APIs
  + Stream processing
  + Message passing
  + Fault Tolerance
* Reactive Streams
* MicroProfile Streams Operators
  + Consuming
  + Filtering
  + Transforming
  + Peeking
  + Connecting
  + Error Handling
* MicroProfile message API
* Fault Tolerance API

**Day 3**

**MicroProfile Fault Tolerance**

* Introduction to MicroProfile Fault Tolerance
* MicroProfile Fault Tolerance in action
* The @Asynchronous policy
* The @Retry policy
* The @Fallback policy
* The @Timeout policy
* The @CircuitBreaker policy
* The @Bulkhead policy
* Handle FrontendService failure conditions when BackendService becomes uncooperative

**MicroProfile Metrics**

* Introduction to MicroProfile Metrics
* Metadata
* Retrieving metrics from the server
* Accessing specific scopes
* Supplying application-specific metrics
* Create FrontendService business metrics

**MicroProfile Distributed Tracing**

* Introduction to MicroProfile OpenTracing API
* Need for Distributed Tracing
* The Importance for Distributed Tracing
* Add OpenTracing to a MicroProfile app
* Connect two MicroProfile applications
* Enable Distributed Tracing

**Day 4**

**MicroProfile Health Check**

* Introduction to MicroProfile Health
* Understanding health checks and how MicroProfile handles them
* The Health Check protocol and wire format
* The Health Check Java API
* Add Health Endpoint to BackendService

**Containerizing, Deploying and Monitoring the services**

* Package applications as jars and native executables
* Developing/Deploying micro profile application in open shift
* View application metrics and monitor application health

**Testing Microservices (Overview)**

* Unit Testing
* Integration Testing
* What is TestContainer
* What is Arquillian
* Integration Tests with Arquillian
* Microservices testing with the Arquillian managed container
* Configuring Arquillian
* Developing Arquillian tests
* Running Arquillian Tests
* Microservices testing with Mock Frameworks
* Tools Available to test microservices

**Day 5**

**Microservices and Cloud**

* Cloud Concepts
* Microservice Architectures
* Understand Cloud architecture
* Compare IaaS and PaaS
* Cloud Native Applications
* Challenges with cloud-native applications
* Reaching for The Clouds
* Embrace "The Cloud"
* Developing a simple microservice system

**Using MicroProfile in Multi-Cloud Environments**

* Using Eclipse MicroProfile for cloud-native application development
* Microservices versus cloud native versus container native
* What about 12-factor applications?
* What about serverless and FaaS?
* Cloud-native application development
* Developing and running MicroProfile applications across clouds
* Bare-metal machines versus VMs versus containers
* Considerations when using MicroProfile in a hybrid cloud deployment

**Microservices Best Practices & Migration**

* Microservice Communication
* Handling Data
* Development to production
* Management and Operations

**Monolith into Microservices**

* Warm up with a Simple and Fairly Decoupled Capability
* Minimize Dependency Back to the Monolith
* Split Sticky Capabilities Early
* Decouple Vertically and Release the Data Early
* Decouple Capability and not Code
* Go Macro First, then Micro

**Conclusion**