Course Duration: 5 Days

Course Overview

Learn to create a flexible development architecture using Agile deployment methods Our Microservices with Docker Training Course will guide you through the steps needed to create flexible and scalable services for your enterprise applications.

You'll learn how to design, build and implement streamlined services, in the form of lightweight containers (essentially Virtual Machines) which will allow you to modularize an otherwise monolithic architecture and take a more Agile approach to your enterprise / deployment architecture.

The course is suitable for all comers, with example code available in Java, Node.js, Golang, and .NET Core - we cover software installation during the course (either natively or on a VM - your choice) so you just need to bring yourself and your laptop.

Course Objectives

By the end of the course, you will have learnt about:

- Microservices Fundamentals
- The architecture of a Microservice
- Obtaining and Deploying the Docker platform
- How to model Microservices
- Integrating multiple Microservices
- Testing Microservices
- Going live
- Maintaining healthy Microservices
- Microservices Security
- Scaling up your Microservices

Who should attend

This course is aimed at SysAdmins and Developers who wish to develop and deploy Microservices.

Prerequisites

Delegates should have some understanding of systems architecture and design. Experience of administering Linux or Windows servers may also be useful. Some basic editing of code will happen during the course, so experience with a language such as Java, C#, C++, Node JS would be advantageous.

Course Syllabus

Microservices Fundamentals

- How did we get here
- What's in a name
- What is a Microservice Architecture
- What is a Microservice
- Benefits of Microservices
- Downsides of Microservices
- Use Cases for Microservices

Introducing Docker

- What is a Container
- What is Docker
- Alternatives to Docker
- The Evolution of Containers
- Hypervisor Virtual Machines
- How containers work
- Containers and Microservice Architectures

Getting Started with Docker

- Prerequisites
- Installing Docker – Native Linux
- Installing Docker – Other Operating Systems
- Docker Toolbox
- Docker Machine Basics
- Running your first Container

Developing a Microservice

- Spring Boot
- Template Microservice
- Play Framework
- Docker

Dockerfile

- Dockerfile
- Instructions and images
- FROM
- RUN
- Building Images
- The Build Context
- Adding files to an Image

- Executing Commands
- Specifying an Entrypoint

Docker Port Mapping

- Multi Container Hosting
- Automatic Port Mapping
- Specific Port Mapping

Deployment Patterns

- Service instance per host
- Multiple service instances per host
- Service instance per VM
- Service instance per container

Communication Patterns

- API Gateway
- Partial Failures
- Circuit Breaker

Service Discovery Patterns

- Client-Side Discovery
- Server-Side Discovery
- Service Registry
- Self-Registration
- 3rd Party Registration

Data Management Patterns

- Shared Database
- Database per Service
- Event Driven Architecture
- Event Sourcing
- Transaction Log Tailing
- Database Triggers
- Application Events
- CQRS (Command Query Responsibility Segregation)

Docker Hub

- What is Docker Hub
- Creating an account
- Creating a Repository
- Markdown Format
- Pushing an Image

Integrating Multiple Microservices

- Keep it simple
- Avoid Breaking Changes
- Sync and Async
- Logging
- Docker Logging

Microservices Security

- Kerckhoff's Principle
- Shannon's Maxim
- Security through obscurity
- General Security Considerations
- Middleware Security Considerations
- Edge Services Security Considerations
- Web & Other Client Security Considerations
- People and Process Security Considerations

REST Interfaces to Microservices

- What is a RESTful Web service?
- HTTP verbs
- HTTP response codes
- Versioning Strategy
- Richardson Maturity Model
- Example scenario
- Key Principles of RESTful Web Services
- Using JAX-RS

Other Interfaces to Microservices

- Protocol Buffers
- Web Sockets

Asynchronous Requests

- Message Brokers
- JMS
- AMQP
- STOMP
- Point to Point
- Publish – Subscribe

Software Development Process

- Speed over Efficiency
- Test Your Assumptions
- High Freedom, High Responsibility
- Conway's Law
- Continuous Integration
- Continuous Delivery / Deployment
- DevOps
- Deliver Early, Deliver Often
- Immutable Infrastructure