Spring Boot MicroServices in Docker and Kubernetes

Introduction to MicroServices with Spring Boot in Docker and Kubernetes Environment

Service Oriented Architecture defines data sharing between the applications in language and platform independent applications as services.
SOAP Web services provides communication and data sharing across the applications in a language neutral way.

MicroServices is a variant of the service-oriented architecture (SOA) architectural style that defines the application as a composition of loosely coupled services.
In MicroServices applications, fine-grained services are supported by lightweight protocols.
The decomposing of an application into different smaller services defines MicroServices and it improves modularity and makes the application easier to understand, develop and test. It enables isolated parallel development by small distributed teams in develop, deploy and scaling the respective services independently.

Spring Boot is a higher level spring framework which simplifies build and deployments of Spring based applications and MicroServices.
These Spring Boot applications get featured support on application health monitoring, diagnosis and a host of other features that make building, deploying, and running the Spring applications far easier.
Spring Boot supports Spring Data, JPA, Web, Services, JDBC, Integration and security flavors to Spring application with minimum programming and configuration.

Docker is an open-source tool that automates the deployment of applications inside software containers, by providing an additional layer of abstraction and automation of operating-system-level virtualization on Linux. The Docker supports independent "containers" to run within a single Linux instance, avoiding the overhead of starting and maintaining virtual machines.

Kubernetes is an open source container orchestration engine designed by Google team to provide a platform for automating deployment, scaling, and operations of application containers across clusters of hosts.

This course imparts the complete practical learning and hands-on experience to the team to jump start to develop MicroService with Spring boot and deploy in Docker and Kubernetes environment to manage as containerized applications.

Course Objectives

This course imparts practical introduction to software developers on MicroService applications with Spring Boot framework with Netflix components and Docker.

The attendees:

- Understand the basics of MicroServices architecture and patterns.

Spring Boot MicroServices in Docker and Kubernetes

- Understand the basics of Spring Boot Framework for MicroServices.
- Build and deploy MicroService applications with Spring Boot
- Utilize the Netflix infrastructure components in MicroService deployments.
- Understand the deployments in Docker and Kubernetes environment.

Prerequisites
The participants must be well versed in java EE web application development and basic understanding about Spring Framework, exposure to Soap and REST services and basic understanding about Linux command line is mandatory.
The participants lacking these skills reduce the pace of class and impact others learning speed.

Training Methodology
Each topic is supplemented with practical demonstrations and exercises for the participants.
The theoretical topics are discussed interactively and technical details are discussed with practical demonstrations and followed by the participants. The participants work on the hands on exercises which strengthen the concepts learned.
The training material including documents, presentations with demo exercises and sample case studies are shared with the participants in soft format.

Software Installations
The Intel dual core compatible CPU with minimum 16GB RAM and 500 GB HDD with **Windows 10 64** bit Adobe Acrobat reader with, JDK1.8 64 bit, WinZip, MySQL 5.x server and Workbench client latest versions and Docker for Windows to be installed.
The Zip distributions for Spring Tool Suite to be made available.

The download installation links
Spring Tool Suite 3.9.11
https://dist.springsource.com/release/STS/3.9.11.RELEASE/dist/e4.14/spring-tool-suite-3.9.11.RELEASE-e4.14.0-win32-x86_64.zip

MySQL community Server-5.6.48
https://dev.mysql.com/get/Downloads/MySQLInstaller/mysql-installer-web-community-5.6.48.0.msi

MySQL Workbench Client ver. 8.0.19
https://cdn.mysql.com/archives/mysql-workbench/mysql-workbench-community-8.0.19-winx64.msi

Docker Desktop for Windows 10
https://download.docker.com/win/stable/Docker%20Desktop%20Installer.exe

The participants must have admin rights on their systems.

Spring Boot MicroServices in Docker and Kubernetes

Live internet connection with reasonable speed and download permissions is required to download the dependencies and plug-ins.

Training duration: Five days. (5 hrs per day)

Course outline

Day1 and Day2

Introduction to MicroServices

- Service Oriented Architecture
- Overview on Soap and REST services
- Monolithic applications and challenges
- What is MicroService?
- Microservices Architecture
- Independent services
- Loosely coupled service applications
- Breaking the monolith into MicroServices
- MicroServices with Spring Boot
- MicroServices pros and Cons

MicroServices Patterns

- Service aggregation and splitting
- API Documentation (Swagger)
- Service Discovery (Eureka)
- Gateway and Config Service
- Message Broker and Asynchronous process
- Event Sourcing & CQRS (Kafka / RabbitMQ)
- Distributed Logging (ELK)
- Circuit Breaker (Hystrix)
- Hands on exercises

Containerization with Docker

- Containers vs. Virtual Machine
- Docker Container Engine architecture
- Docker daemon and docker client cli
- Docker-hub the image registry
- Docker CLI commands overview
- Docker container and image
- Create and run the Docker containers
- Inspect docker process and containers

- Start, Stop and remove containers
- Interactive and detached containers
- Manage the docker images
- Docker container Logging and monitoring
- Commit the local modified container as image
- Define your own image with Dockerfile
- Dockerfile directives
- Build the image from Dockerfile
- Push and pull images from Docker-Hub
- Hands on exercises

Day3

Docker container State management

- Docker container state
- Sharing the data across containers with volume
- Make the Volumes data persistent across container restarts
- Mount volumes to containers from host system
- Copy files/dirs from container to host and vice versa
- Clean the docker environment

MicroService deployments in Docker

- Configure and create the database container
- Define the volume with database container
- Deploy the front-end application with link to database container
- Install and Use local Docker Registry
- Monitoring services
- Host network mapping
- Host Port Mapping
- Dynamic port mapping
- Default bridge network

Docker Compose Usage

- Container configuration with YML files
- Docker-compose commands
- Configure and Deploy micro-service with database backend

Day4 and Day5

Docker Container Orchestration requirements

- High availability with Cluster
- Load balancing and instance management
- Run the application across multiple host nodes

- Application updates (rolling updates) and roll back
- Scaling the containers
- Monitor the containers
- Inspect the services

Kubernetes for Container orchestration

- Introduction and features
- Kubernetes architecture and features
- Kubernetes with container engine
- Kubernetes Core Concepts
- Kubeadm and kubectl tools
- Start the basic single node cluster with kubeadm
- The kubectl commands overview
- Basic objects of deployment
    - Pod
    - Deployment
    - Service
    - Volume
    - Namespace
    - Job
- The pod configuration in YML file
- Create and manage pods with kubectl
- The host network and host port mode
- Run and monitor the pods with logs
- Inspect the pods
- Interact with pods
- Pods with multiple containers: inspect, interact and logs
- Linking the containers in pod
- Monitor and manage the cluster with dashboard application
- The logging drivers

Manage the container state with volume

- The pods with volume
- Sharing the data across containers within POD.
- Making the volume data persistent by mounting from host system.
- Configure PersistentVolume and PersistentVolumeClaim objects
- Using the PersistentVolumeClaim in the container

Service Deployments

- Deployment : to group multiple pod instances
- Replication : To manage multiple replicas of pods
- Staefulset: To manage the Stateful applications
- DaemonSet : To manage the group of pods across hosts
- Job :Grouping of pods
- Create deployments for MicroServices in cluster

Spring Boot MicroServices in Docker and Kubernetes

- The deployment options
- The labels and selectors in deployment
- The service types
    - ClusterIP
    - NodePort
    - LoadBalancer
    - ExtrenalName
- Expose the deployment with service
- Service port mapping : fixed and dynamic
- Inspect and log the service
- Scale the service with pods
- Service Discovery in Kubernetes
- Default services and namespaces
- Communication with outside world

****