En-lighten Technology Ltd

Kubernetes and Helm For Java Developers: Designing, Deploying and Managing Cloud Native Applications

Overview

This course is for Java developers who want to learn how to design, deploy and manage cloud native applications on Kubernetes. The course provides in depth coverage of how to design, configure, package and deploy Java applications and services to a Kubernetes cluster. The course starts with simple Java services and incrementally builds to show how to package, deploy, observe and troubleshoot applications. The networking is introduced in detail and various aspects of securing applications discussed. The key Kubernetes concepts and tools are introduced all from an application developer viewpoint, not a system administrator. The lifecycle of individual services and of applications is covered and upgrade strategies detailed. Pacakging and deploying using Helm is a major part of the course. How to observe and monitor services is detailed and troubleshooting methods introduced. Throughout best practices are emphasized.

Attendees will learn:

- How to design, deploy and manage applications on Kubernetes
- The Kubernetes architecture and core components
- Package services and applications for Kubernetes deployment
- The Kubernetes networking model and exposing services
- How to perform zero downtime deployments and upgrades
- Secure applications
- Packaging and deploying services and applications using Helm
- Observe and monitor deployed microservices and applications
- Troubleshoot applications and containers

Audience and Recommended Background

The course is a hands-on course with practical exercises. Attendees should have a working knowledge of Docker to the level of the Docker Fundamentals course and of packaging Java applications in containers.

Course Approach

Taught material will be delivered in short sessions and will be followed by the opportunity to practice what has been taught and reflect on how to put it into practice.

Course Duration: 5 days

Course Content

Kubernetes Core Concepts

- What is Kubernetes?
- Kubernetes concepts
    - Pods

    - Namespaces
    - Replication controllers
    - Replica sets
    - Deployments
    - Services
    - Jobs
    - Volumes
- Kubernetes architecture

Cloud Native Applications

- Architecting cloud native applications
- Microservices and Kubernetes
- Domain driven design and service independence
- Designing stateful applications
- Deploying an application to Kubernetes
- Best practices for development and testing

A First Deployment to Kubernetes

- A single service deployment with run
- Accessing the service
- Pod names and port forwarding
- Gaining access to container logs
- Kubernetes declarative infrastructure
- Declaring your first application
- Deploying your first application

Kubernetes Configuration: ConfigMaps and Secrets

- What are ConfigMaps and why are they useful?
- Creating ConfigMaps
- Using a ConfigMap
- What are secrets?
- Creating secrets
- Consuming secrets
- Naming constraints
- Managing ConfigMaps and secrets
    - Listing
    - Creating
    - Updating

Managing State and Stateful Applications

- Persistent volumes and volume mounts
- Persistent volume claims
- Using remote disks
- Storage classes
- Kubernetes storage best practices
- Stateful applications
    - SatefulSets

    - Operators
    - StatefulSet and operator best practices

Services and Networking

- Services
    - ClusterIP
    - NodePort
    - ExternalName
    - LoadBalancer
- Ingress and Ingress controllers
- Services and Ingress controller best practices
- Isolated and non-isolated pods
- NetworkPolicy resource
- To and from selectors

Up and Running with Helm

- Installing Helm
- The role of Helm
- Helm architecture
- Using Helm
    - Packages, repositories and releases
    - Simple Helm commands
- Creating a Helm chart
- Deploying application to Kubernetes
- Finding charts with helm search
- Customising charts before installing
- Creating image pull secrets

Helm Charts in Detail

- Chart file structure
- The Chart.yaml file
- Versioning charts
- Dependencies and dependency management
- Tags and condition fields in dependencies
- Importing child values via dependencies
- Chart repositories
- Signing charts
- Chart tests
- Chart hooks
- Library charts
- Provenance and integrity
- Helm upgrade and helm rollback
- Automatically rolling deployments
- Instructing helm to not uninstall resources
- Complex charts with many dependencies

Developing Templates

- Introduction to Helm chart templates

En-lighten Technology Ltd

- Built-in objects
- Values files
- Template functions and pipelines
- Flow control
- Variables
- Named templates
- Using partials and template includes
- Accessing files inside templates
- Notes.txt files
- Subcharts and global values
- .Helmignore
- Debugging templates

Helm Best Practices

- Chart names
- Integration charts
- General conventions
- Values
- Templates
- Requirements
- Labels and Annotations
- Pods and Pod Templates
- Custom Resource Definitions
- Role based access control
- Helm plugins
- Composing applications
- Helm lint

Monitoring and Deployment Strategies

- Monitoring for uptime
- Liveness probes
- Readiness probes
- Using labels and selectors
- Versioning, releases and rollouts
- Scaling and autoscaling
- Executing zero downtime deployments
- Application deployment
- Individual service deployments
- Releasing features
    - Canary deployments
    - Blue-green deployments
- Controlling the rate of rollout
- Preparing for rollbacks
- Best practices for versioning, releases and rollouts

Securing Deployments

- Securing clusters with role-based access control

En-lighten Technology Ltd

- Identity in Kubernetes
- Understanding roles and role bindings
- Users and roles and service accounts
- Techniques for managing RBAC
- Network security policies
- Isolating the pod network

Observability

- The need for observability
- The three pillars of observability:
    - Logs, metrics, traces
- Key microservice metrics:
    - Four golden signals
    - Apdex score
    - RED
    - USE
- Exposing key metrics from a microservice
- Introducing Prometheus for metrics gathering
- Gathering metrics data with Prometheus
- Visualising metrics with Grafana
- Logging and aggregating logs
- Best practices for logging
- Alerting on logs
- Distributed tracing with Jaeger
- Spans and traces
- Making use of traces

Troubleshooting

- Predicting problems before they occur
- A structured approach to troubleshooting
- Logs should be the last resort
- Interacting with code in Kubernetes with debuggers
- Common errors and how to find them
- Missing endpoints
- Failed deployments
- Resource capacity problems