Getting Started

Kubernetes in Docker

Release 2021-11-01-2235

Instructor

Sean P. Kane

@spkane

@superorbital_io

SUPERORBITAL

Follow Along Guide

Textual Slides

Katacoda Online Sandbox

https://learning.oreilly.com/scenarios/devops-tools-sandbox/9781098126469/

Prerequisites (1 of 2)

- A recent computer and OS
  - Recent/Stable Linux, macOS, or Windows 10
  - Reliable and fast internet connectivity
- **Docker Desktop/Community Edition & Docker Compose**
- **kubectl**
- *Optional*: Visual Studio Code
  - w/ Docker, Kubernetes & Kubernetes Kind extensions

Prerequisites (2 of 2)

- A graphical web browser
- A text editor
- A software package manager
- Git client
- General comfort with the command line will be helpful.
- curl

A Note for Powershell Users

Terminal commands reflect the Unix bash shell. PowerShell users will need to adjust the commands.

- Unix Variables
  - `export MY_VAR=test`
  - `echo ${MY_VAR}`
- Windows 10 Variables (powershell)
  - `$env:my_var = "test"`
  - `Get-ChildItem Env:my_var`

Translation Key

/ - Unix Shell Line Continuation
` - Powershell Line Continuation (sort of)

${MY_VAR} - Is generally a place holder in the slides.

Linux Container Mode

- On the Windows platform make sure that you are running in Linux Container mode.

A Note About Proxies

Proxies can interfere with some activities if they are not configured correctly.

- Kind
- Docker
- Docker-Compose

Instructor Environment

- **Operating System**: macOS (v11.2.X+)
- **Terminal**: iTerm2 (Build 3.X.X+) - https://www.iterm2.com/
- **Shell Prompt Theme**: Starship - https://starship.rs/
- **Shell Prompt Font**: Fira Code - https://github.com/tonsky/FiraCode
- **Text Editor**: Visual Studio Code (v1.X.X+) - https://code.visualstudio.com/

Setup Workspace

```
$ mkdir $HOME/class
$ cd $HOME/class
```

Docker Resource Configuration

- It is recommended that you increase the CPU and memory avaliable to Docker, if you are running it in a virtual machine.
  - 4 CPU/core & 8 GB of memory should be reasonable.

Test Docker

```
$ docker image ls
$ docker container run -d --rm -p 18080:8080 --name quantum2 spkane/quantum
$ docker container ls
# Wait for it to finishing starting
$ docker container logs -f quantum2
```

Test the Container

- Find this in the logs

```
...
 DONE  Compiled successfully in 31398ms5:55:24 PM

<s> [webpack.Progress] 100%

  App running at:
  - Local:   http://localhost:8080/
...
```

Navigate to: `http://localhost:18080/`

Stop the Container

```
$ docker stop quantum2
$ docker container ls
$ docker container ls -a
```

Installing KinD - 1 of 2

- `KinD` : Kubernetes IN Docker
- Navigate to:
  - https://github.com/kubernetes-sigs/kind/releases/
- Download most recent release for your system.
  - e.g. `kind-darwin-amd64`
- Rename downloaded file to `kind`

Installing KinD - 2 of 2

- You might need to make the binary executable.
  - e.g. `chmod u+rwx kind`
- Move file to a folder in your $PATH.

macOS Catalina+ Notice 1 of 2

- You may need to whitelist the binary every time you update it, since it is not signed.
  - You might need to do something similar for `kubectl` as well.

```
$ kind
```

- Click `Cancel`

macOS Catalina+ Notice 2 of 2

- Go to `System Preferences` → `Security & Privacy` → `General`
  - Click `Allow Anyway`
- Run `kind`
  - Click `Open`

Test Kind Binary

```
$ kind

kind creates and manages local Kubernetes clusters using
Docker container 'nodes'

Usage:
  kind [command]

...

Use "kind [command] --help" for more information about a command.
```

Unix Shell Completion

```
$ kind completion
```

Spin up a local cluster (single-node)

```
$ kind create cluster --name class
Creating cluster "class" ...
 ✓ Ensuring node image (kindest/node:v1.20.2)
 ✓ Preparing nodes 📦
 ✓ Writing configuration 📜
 ✓ Starting control-plane 🕹️
 ✓ Installing CNI 🔌
 ✓ Installing StorageClass 💾
Set kubectl context to "kind-class"
You can now use your cluster with:
kubectl cluster-info --context kind-class
Have a nice day! 👋
```

Examine the Cluster

```
$ kubectl cluster-info --context kind-class
```

Setting up kubectl

- This is already done for us, by kind, but just for the record:

```
$ kubectl config set current-context --namespace=default kind-class
```

View our Auth info

```
$ kubectl config view --minify
```

What did this get us?

Definitions

- **Note**: Most of the definitions in this class come straight from the Kubernetes documentation.
  - https://kubernetes.io/docs/concepts/

Node

A node typically represents a server or VM that can run workloads in kubernetes. These workloads may be part of the kubernetes control plane or workloads added by users. In our case our node is comprised of a single container in Docker.

- `kind get nodes --name class`
- `kubectl get nodes --show-labels`

kubelet

This is the primary service on each node that registers the node with the cluster and acts like the primary interface between the cluster and each individual node.

- **Unix Shells**:
  - `docker top class-control-plane | grep /usr/bin/kubelet`
- **Powershell**:
  -
```
docker top class-control-plane | Select-String -Pattern
/usr/bin/kubelet
```

API Server (kube-apiserver)

The API server is the primary interface with the cluster. This is used by many components ini the cluster and tools like `kubectl` to interact with the various REST API interfaces that manage the creation, updating and destruction of objects in the system.

- **Unix Shells**:
  - `docker top class-control-plane | grep kube-apiserver`
- **Powershell**:
  - `docker top class-control-plane | Select-String -Pattern kube-apiserver`

And more...

Access the API Server

- In a new shell run:

```
$ kubectl proxy --port=8080
Starting to serve on 127.0.0.1:8080
```

Test the API Server

- In the original terminal run:

```
$ curl http://localhost:8080/api/
$ curl http://localhost:8080/api/v1/nodes/
```

Stop the Proxy

- In the previous shell running `kubectl proxy` type [Control-C]

Definition: Kind

In Kubernetes manifests you will always see a line near the top that starts with `Kind:`. The word kind has a special meaning in Kubernetes and is used to define what type/kind of object we are managing. In the next few slides we will discuss some of most basic object types that developers needs to understand.

- **Note**: This is not the same thing as the tool, `kind`.
  - **Lesson**: Clever naming often leads to confusion.

Definition: Namespaces

Kubernetes supports multiple virtual clusters backed by the same physical cluster. These virtual clusters are called namespaces.

Namespaces are intended for use in environments with many users spread across multiple teams, or projects. Namespaces provide a scope for names. Names of resources need to be unique within a namespace, but not across namespaces. Namespaces cannot be nested inside one another and each Kubernetes resource can only be in one namespace.

Definition: Pods

A Pod is a group of one or more containers, with shared storage/network resources, and a specification for how to run the containers. A Pod's contents are always co-located and co-scheduled, and run in a shared context. A Pod models an application-specific "logical host": it contains one or more application containers which are relatively tightly coupled.

Definition: ReplicaSets

A ReplicaSet's purpose is to maintain a stable set of replica Pods running at any given time. As such, it is often used to guarantee the availability of a specified number of identical Pods.

Definition: Deployments

A Deployment provides declarative updates for Pods and ReplicaSets.

You describe a desired state in a Deployment, and the Deployment Controller changes the actual state to the desired state at a controlled rate. You can define Deployments to create new ReplicaSets, or to remove existing Deployments and adopt all their resources with new Deployments.

Clone Manifests

```
$ git clone https://github.com/spkane/class-kind-manifests.git \
    --config core.autocrlf=input
$ cd class-kind-manifests
```

Launch a Deployment

```
$ kubectl apply -f quantum-game-2/qg2-deployment.yaml
$ kubectl get deployments
$ kubectl get replicasets
$ kubectl get pods
```

Connect to the Deployment

```
$ kubectl describe pod ${POD_NAME}
$ kubectl port-forward ${POD_NAME} 8080
```

- In your web browser navigate to:
  - http://localhost:8080/
- Type [Ctrl]-C to stop port forward.

Delete the Deployment

```
$ kubectl get deployments
$ kubectl delete deployment ${DEPLOYMENT_NAME}
$ kubectl get deployments
$ kubectl get pods
```

Delete the Cluster

- Let's delete the cluster, so that we can customize it just a bit.

```
$ kind delete cluster --name class
```

Definition: Services

An abstract way to expose an application running on a set of Pods as a network service.
With Kubernetes you don't need to modify your application to use an unfamiliar service discovery mechanism. Kubernetes gives Pods their own IP addresses and a single DNS name for a set of Pods, and can load-balance across them.

ServiceTypes

- ClusterIP (*default*)
- NodePort
- LoadBalancer
- ExternalName

Definition: ClusterIP

ServiceType

Exposes the Service on a cluster-internal IP. Choosing this value makes the Service only reachable from within the cluster. This is the default ServiceType.

- These require us to use `kubectl port-forward` to connect to.

Kind Configuration File

- Examine `kind-config/ingress.yaml`
- Copy file to `$HOME/.kind/ingress.yaml`

```
$ mkdir -p $HOME/.kind
$ cp kind-config/ingress.yaml $HOME/.kind/ingress.yaml
```

Create a Customized Cluster

```
$ kind create cluster --name class --config=$HOME/.kind/ingress.yaml
```

Definition: NodePort

ServiceType

Exposes the Service on each Node's IP at a static port (the NodePort). A ClusterIP Service, to which the NodePort Service routes, is automatically created. You'll be able to contact the NodePort Service, from outside the cluster, by requesting **<NodeIP>:<NodePort>**.

NodePort Service

```
$ kubectl apply -f quantum-game-2/qg2-deployment.yaml
$ kubectl apply -f quantum-game-2/qg2-service-nodeport.yaml
```

- Navigate to: http://NodeIP:NodePort (*e.g.* *http://127.0.0.1:30080/*)
  - Due to some known issues on macOS with Docker for Mac. I can not easily demonstrate connecting to this in KinD.
    - https://docs.docker.com/docker-for-mac/networking/#known-limitations-use-cases-and-workarounds

Definition: LoadBalancer

ServiceType

Exposes the Service externally using a cloud provider's load balancer. NodePort and ClusterIP Services, to which the external load balancer routes, are automatically created.

- Not typically used with Kind.

Definition: ExternalName

ServiceType

Maps the Service to the contents of the externalName field (e.g. foo.bar.example.com), by returning a CNAME record with its value. No proxying of any kind is set up.

- Not typically used with Kind.

Definition: Ingress

An API object that manages external access to the services in a cluster, typically HTTP.

Ingress may provide load balancing, SSL termination and name-based virtual hosting.

Install Ingress Operator

Nginx

```
$ kubectl delete service quantum-game-2
# Upstream version:
# https://raw.githubusercontent.com
# /kubernetes/ingress-nginx/master/deploy/static/provider/kind/deploy.yaml
$ kubectl apply -f kind/ingress-nginx.yaml
$ kubectl wait --namespace ingress-nginx \
  --for=condition=ready pod \
  --selector=app.kubernetes.io/component=controller \
  --timeout=90s
$ kubectl  get all --namespace ingress-nginx
```

Add Service w/ Ingress

```
$ kubectl apply -f quantum-game-2/qg2-ingress.yaml
```

- In your web browser navigate to:
  - http://localhost/

Scale the Deployment

```
$ kubectl get pods
$ kubectl scale deployment quantum-game-2 --replicas=3
$ kubectl get pods
```

Install Stern for Logging - 1 of 2

- Install the latest version of Stern:
  - https://github.com/wercker/stern/releases/latest
- Download most recent release for your system.
  - e.g. `stern_darwin_amd64`
- Rename downloaded file to `stern`
- For Apple Silicon (M1) the best option at the moment is to use `brew install stern`, until the developer adds a `darwin_arm64` build to Github.

Install Stern for Logging - 2 of 2

- You might need to make the binary executable.
  - e.g. `chmod u+rwx kind`
- Move file to a folder in your $PATH.
- If required, authorize executable with OS security features.

Watching the Logs

```
$ stern --all-namespaces -l app=quantum-game-2

default quantum-game-2-6554d68fb8-n6nnj quantum-game-2 ...
default quantum-game-2-6554d68fb8-lpgzz quantum-game-2 ...
default quantum-game-2-6554d68fb8-yhwfn quantum-game-2 ...

$ kubectl scale deployment quantum-game-2 --replicas=1
$ kubectl get pods
```

Advanced Kind Features

- https://kind.sigs.k8s.io/docs/user/quick-start/#advanced
  - Multiple worker nodes
  - Control-plane high availability
  - Mapping ports to the host machine
  - Setting the Kubernetes version
  - Enabling feature gates
  - IPv6 networking
  - and more...

k9s

- A terminal UI tool for interacting with Kubernetes clusters.
  - https://github.com/derailed/k9s

IDE Integration - Visual Studio Code

- Docker Extension
  - Explore images & containers
  - View and download files
- Kubernetes & KinD Extension
  - Explore Cluster and YAML manifest

Tear Down Cluster

```
$ kind delete cluster --name class
```

What We Have Learned

- Installing Kubernetes in Docker
- Modifying the Kind/Cluster Configuration
- Primary Components of a Cluster
- Pods, ReplicaSets & Deployments
- ServiceTypes and Ingress
- Scaling, Logs, and IDE Integtation
- Advanced KinD Features

Additional Reading

Kubernetes: Up & Running

Docker: Up and Running

KinD - Kubernetes in Docker

Additional Learning Resources

https://learning.oreilly.com/

Student Survey

Please take a moment to fill out the class survey linked to from the bottom of the ON24 audience screen.

O'Reilly and I value your comments about the class.

Thank you!

Any Questions?

Sean P. Kane

Providing stellar Kubernetes engineering and workshops.

https://superorbital.io/contact/