Workloads

Kubernetes Objects - apiVersion, Kind, metadata, and Spec

Pod

alias k=kubectl

k api-resources

k explain pod.spec.containers

k explain pod.spec.containers [--]recursive

One-container-per-Pod

Create a file - pod.yaml

apiVersion: v1

kind: Pod

metadata:

  name: nginx

spec:

  containers:

  - name: nginx

    image: nginx:1.14.2

    ports:

    - containerPort: 80

kubectl apply -f pod.yaml

kubectl get pods

kubectl describe pod nginx

# Looks for events and lifecyle

Multi-Container-Pod

apiVersion: v1

kind: Pod

metadata:

  name: multicontainer-pods

  labels:

    app: httpd

    tier: frontend-backend

    version: v1

spec:

  containers:

  #Container 01

  - name: web

    image: httpd

    ports:

    - containerPort: 80

  #Container 02

  - name: redis

    image: redis

Init-Container

apiVersion: v1

kind: Pod

metadata:

  name: purple

spec:

  containers:

    - command:

        - sh

        - -c

        - echo The app is running! && sleep 3600

      image: busybox:1.28

      name: purple-container

  # Adding 2 init containers to execute sleep commands

  initContainers:

    - command:

        - sh

        - -c

        - sleep 60

      image: busybox:1.28

      name: warm-up-1

    - command: ["sh", "-c", "sleep 120"]

      image: busybox:1.28

      name: warm-up-2

kubectl get pods -w

# Both the init containers will get executed before the main container is started

# NAME     READY   STATUS     RESTARTS   AGE

# purple   0/1     Init:1/2   0          2m41s

# After 3mins(60+120seconds), the output will be

# NAME     READY   STATUS    RESTARTS   AGE

# purple   1/1     Running   0          3m7s

Static-Pod

In worker01, Create a yaml file in

apiVersion: v1

kind: Pod

metadata:

  name: static-web

  labels:

    role: myrole

spec:

  containers:

    - name: web

      image: nginx

      ports:

        - name: web

          containerPort: 80

          protocol: TCP

In the master node,

k get pods -A

The pod will appear in default ns

Delete the static pod file in worker01

In the master node,

k get pods -A

The pod will Disappear in default ns

Resource Limits

apiVersion: v1

kind: Pod

metadata:

  name: rl-pod

spec:

  containers:

  - name: nginx

    image: nginx:1.14.2

    ports:

    - containerPort: 80

    resources:

      requests: # Minimum Value

        memory: "100Mi"

        cpu: "250m" # 1 core = 1000m

      limits:  # Maximum Value

        memory: "128Mi"

        cpu: "300m"

## Deployment

Diagram

Description automatically generated

apiVersion: apps/v1

kind: Deployment

metadata:

  name: nginx-deployment

  labels:

    app: nginx

spec:

  replicas: 3

  selector:

    matchLabels:

      app: nginx

  template:

    metadata:

      labels:

        app: nginx

    spec:

      containers:

      - name: nginx

        image: nginx:1.14.2

        ports:

        - containerPort: 80

kubectl get deployments

kubectl rollout status deployment/nginx-deployment

kubectl get rs

kubectl get pods --show-labels

### Update Deployment

kubectl set image deployment/nginx-deployment nginx=nginx:1.16.1

kubectl rollout status deployment/nginx-deployment

kubectl get rs

kubectl get pods | grep nginx-deployment

kubectl describe deployment nginx-deployment

Setting wrong image

kubectl set image deployment/nginx-deployment nginx=nginx:1.161

kubectl rollout status deployment/nginx-deployment

Waiting for rollout to finish: 1 out of 3 new replicas has been updated...

kubectl get rs

kubectl get pods | grep nginx-deployment

kubectl describe deployment

kubectl rollout history deployment/nginx-deployment

kubectl rollout history deployment/nginx-deployment --revision=2

### Rolling Back to a Previous Revision

kubectl rollout undo deployment/nginx-deployment

kubectl rollout history deployment/nginx-deployment

kubectl rollout history deployment/nginx-deployment --revision=4

kubectl get deployment nginx-deployment

kubectl describe deployment nginx-deployment

# Check container image version/tag

k rollout undo deployment/nginx-deployment --to-revision=1

### Scaling deployment

kubectl scale deployment/nginx-deployment --replicas=5

kubectl get deployment nginx-deployment

kubectl get rs

kubectl get pods | grep nginx-deployment

kubectl describe deployment nginx-deployment

## Daemon Set

apiVersion: apps/v1

kind: DaemonSet

metadata:

name: fluentd-elasticsearch

namespace: kube-system

labels:

k8s-app: fluentd-logging

spec:

selector:

matchLabels:

name: fluentd-elasticsearch

template:

metadata:

labels:

name: fluentd-elasticsearch

spec:

tolerations:

# this toleration is to have the daemonset runnable on master nodes

# remove it if your masters can't run pods

- key: node-role.kubernetes.io/master

operator: Exists

effect: NoSchedule

containers:

- name: fluentd-elasticsearch

image: quay.io/fluentd\_elasticsearch/fluentd:v2.5.2

resources:

limits:

memory: 200Mi

requests:

cpu: 100m

memory: 200Mi

volumeMounts:

- name: varlog

mountPath: /var/log

- name: varlibdockercontainers

mountPath: /var/lib/docker/containers

readOnly: true

terminationGracePeriodSeconds: 30

volumes:

- name: varlog

hostPath:

path: /var/log

- name: varlibdockercontainers

hostPath:

path: /var/lib/docker/containers

kubectl apply -f https://k8s.io/examples/controllers/daemonset.yaml

kubectl get ds

kubectl describe ds fluentd-elasticsearch

kubectl get pods -o wide | grep fluentd

## Jobs

apiVersion: batch/v1

kind: Job

metadata:

name: pi

spec:

template:

spec:

containers:

- name: pi

image: busybox:1.28

imagePullPolicy: IfNotPresent

command:

- /bin/sh

- -c

- date; echo Hello from the Kubernetes cluster

restartPolicy: Never

backoffLimit: 4

kubectl apply -f job.yaml

kubectl describe jobs/pi

pods=**$(**kubectl get pods --selector=job-name=pi --output=jsonpath='{.items[\*].metadata.name}'**)**

echo $pods

kubectl logs $pods

## CronJobs

wget <https://k8s.io/examples/application/job/cronjob.yaml>

Note: Change version in yaml

kubectl create -f cronjob.yaml

kubectl get cronjob hello

kubectl get jobs -w

kubectl delete -f cronjob.yaml

## Configuration basics

Introduction:

Graphical user interface

Description automatically generated

[All JAVA Spring Configuration item](https://docs.spring.io/spring-boot/docs/current/reference/html/application-properties.html#application-properties.data.spring.datasource.username)

[MySQL DB Configuration](https://github.com/spring-guides/gs-accessing-data-mysql/blob/main/complete/src/main/resources/application.properties)

a[pplication.properties](https://github.com/spring-guides/gs-accessing-data-mysql/blob/main/complete/src/main/resources/application.properties)

spring.datasource.url=jdbc:mysql:// \

${MYSQL\_HOST:localhost}:3306/db\_example

spring.datasource.username=springuser

spring.datasource.password=ThePassword

spring.datasource.driver-class-name =com.mysql.jdbc.Driver

For each environment, we define the individual application.properties

Graphical user interface, text, application, chat or text message

Description automatically generated

In application.properties, provide default profile

spring.profiles.active=dev

While deploying the jar at different environments,

java -jar app.jar -Dspring.profiles.active=prod

## Env

apiVersion: v1

kind: Pod

metadata:

name: envar-demo

labels:

purpose: demonstrate-envars

spec:

containers:

- name: envar-demo-container

image: gcr.io/google-samples/node-hello:1.0

env:

- name: DEMO\_GREETING

value: "Hello from the environment"

- name: DEMO\_FAREWELL

value: "Such a sweet sorrow"

kubectl apply -f https://k8s.io/examples/pods/inject/envars.yaml

kubectl get pods -l purpose=demonstrate-envars

kubectl exec envar-demo -- printenv

## ConfigMaps

apiVersion: v1

kind: ConfigMap

metadata:

name: game-demo

data:

*# property-like keys; each key maps to a simple value*

player\_initial\_lives: "3"

ui\_properties\_file\_name: "user-interface.properties"

*# file-like keys*

game.properties: |

*enemy.types=aliens,monsters*

*player.maximum-lives=5*

user-interface.properties: |

*color.good=purple*

*color.bad=yellow*

*allow.textmode=true*

K describe cm game-demo

**Using Env**

apiVersion: v1

kind: Pod

metadata:

name: dapi-test-pod01

spec:

containers:

- name: test-container

image: k8s.gcr.io/busybox

command: [ "/bin/sh", "-c", "env" ]

**env:**

- name: LIVES

valueFrom:

configMapKeyRef:

name: game-demo

key: player\_initial\_lives

- name: FILE\_NAME

valueFrom:

configMapKeyRef:

name: game-demo

key: ui\_properties\_file\_name

restartPolicy: Never

**Using EnvFrom**

apiVersion: v1

kind: Pod

metadata:

name: dapi-test-pod02

spec:

containers:

- name: test-container

image: k8s.gcr.io/busybox

command: [ "/bin/sh", "-c", "env" ]

**envFrom:**

- configMapRef:

name: game-demo

restartPolicy: Never

## Secrets

apiVersion: v1

kind: Secret

metadata:

name: mysecret

type: kubernetes.io/basic-auth

stringData:

username: admin

password: t0p-Secret

---

apiVersion: v1

kind: Pod

metadata:

name: secret-env-pod

spec:

containers:

- name: mycontainer

image: redis

env:

- name: SECRET\_USERNAME

valueFrom:

secretKeyRef:

name: mysecret

key: username

- name: SECRET\_PASSWORD

valueFrom:

secretKeyRef:

name: mysecret

key: password

restartPolicy: Never

kubectl exec -it secret-env-pod env

## Labels & Selectors

Check Labels of Nodes and Workloads

kubectl get pods --show-labels

Diagram

Description automatically generated

wget https://raw.githubusercontent.com/sparkmbt/sparkmbt/main/kubesample.yaml

kubectl apply -f kubesample.yaml

Using Selectors,

kubectl get pods -l tier=backend

kubectl get pods -l tier=frontend

### Set-based selectors

kubectl get pods -l 'tier notin (backend, frontend)'

kubectl get pods -l 'tier in (backend), role notin (slave)'