kubectl cluster-info #Kubernetes kurulu mu değil mi

kubectl explain nodes

kubectl explain pods

kubectl run nginx --image=nginx

kubectl get pods

**çıktısı:**



**Yaml ile nginx Pod oluşturma:**

apiVersion: v1

kind: Pod

metadata:

  name: nginx-pod

  labels:

    app: nginx

spec:

  containers:

  - name: mynginx

    image: nginx

    ports:

    - containerPort: 80

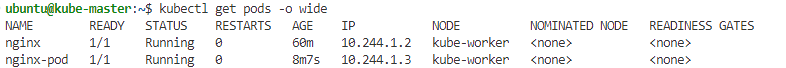
kubectl create -f mypod.yaml

kubectl apply -f mypod.yaml

kubectl get pods #podları görüntülemek için

kubectl get pods -o wide #Podlar hakkında daha geniş bilgi almak için

**Çıktısı:**



kubectl describe pods/nginx-pod #Oluşan pod hakkında daha ayrıntılı bilgi

kubectl get pods/nginx-pod -o yaml #Bilgiyi yaml formatında gösterir.

kubectl delete -f mypod.yaml

# or

kubectl delete pod nginx-pod

kubectl delete pod pod\_adı #Başka konteyner varsa ıd ile silmek gerekir.

**Replika set:**

apiVersion: apps/v1

kind: ReplicaSet

metadata:

name: nginx-rs

labels:

environment: dev

spec:

# modify replicas according to your case

replicas: 3

selector:

matchLabels:

app: nginx

template:

metadata:

labels:

app: nginx

spec:

containers:

- name: mynginx

image: nginx

ports:

- containerPort: 80

kubectl apply -f myreplicaset.yaml

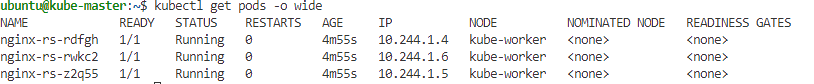
kubectl get replicaset #Kısaltma kullanarakta bu komut çalışır **kubectl get rs**

**Çıktısı:**



kubectl get pods -o wide

**Çıktısı:**



kubectl delete -f myreplicaset.yaml

**Deployment:**

apiVersion: apps/v1

kind: Deployment

metadata:

  name: nginx-deployment

  labels:

    environment: dev

spec:

  replicas: 3

  selector:

    matchLabels:

      app: nginx

  template:

    metadata:

      labels:

        app: nginx

    spec:

      containers:

      - name: nginx

        image: nginx

        ports:

        - containerPort: 80

kubectl apply -f mydeployment.yaml

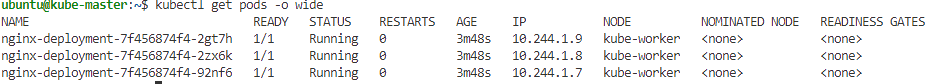
kubectl get deployments

**Çıktısı:**



kubectl get pods -o wide

**Çıktısı:**



kubectl describe deploy/nginx-deployment

kubectl exec pod\_adı -- date

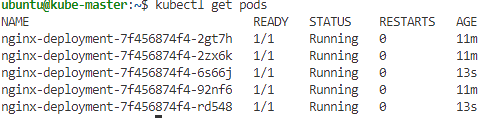
**Çıktısı:**



kubectl exec -it <pod-name> -- bash #Oluşan poda bash ile bağlanılır.

kubectl scale deploy nginx-deployment --replicas=5

**Oluşan podlar:**



kubectl delete -f mydeployment.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

  name: devops-deploy

  labels:

    app: container-info

  annotations:

    kubernetes.io/change-cause: devops-deploy is set as container-info=gluobe/container-info:green

spec:

  replicas: 3

  selector:

    matchLabels:

      app: container-info

  template:

    metadata:

      labels:

        app: container-info

    spec:

      containers:

      - name: container-info

        image: gluobe/container-info:green

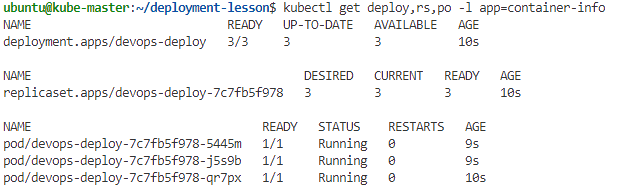
        ports:

        - containerPort: 80

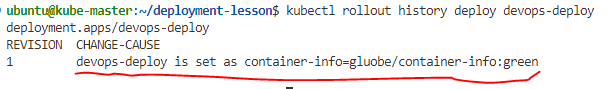
kubectl apply -f devops-deploy.yaml

kubectl get deploy,rs,po -l app=container-info

**Çıktısı:**



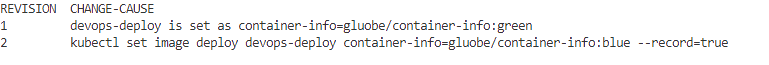
kubectl rollout history deploy devops-deploy #   annotations: kısmındaki bilgiyi getirir.



kubectl set image deploy devops-deploy container-info=gluobe/container-info:blue --record=true #image başlangıçta greendi mavi oldu.

kubectl rollout history deploy devops-deploy #--record=true dediğimiz için komut çalışır.Kayıtları görürüz.

**Çıktısı:**

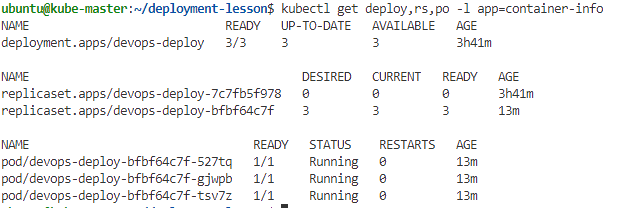


kubectl rollout history deploy devops-deploy --revision=1

kubectl rollout history deploy devops-deploy --revision=2

kubectl get deploy,rs,po -l app=container-info

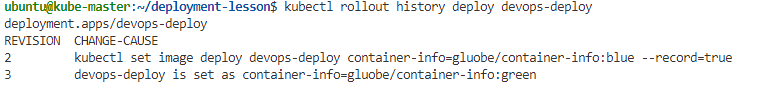
**Çıktısı:**



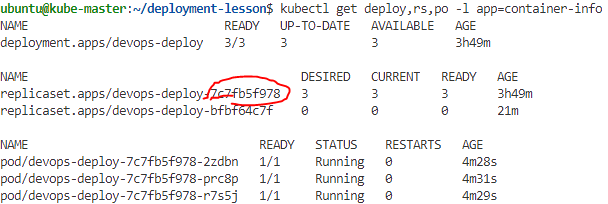
kubectl rollout undo deploy devops-deploy --to-revision=1 #Revizyon 1e tekrar dönmek istenirse

kubectl rollout history deploy devops-deploy

**Çıktı:**



#Tekrar Revizyon bire döndü ancak revizyon 1 ‘i 3 diye yazdı.



Replikat set olarak revision 1dekine(978) 3 3 3 yazıyor ona döndüğünü burdan algılayabiliriz.Önceki sayfada revizyon 2 de iken 978 0 0 0 yazıyordu.

kubectl delete deploy -l app=container-info