Getting Started with Microservices and Kubernetes: Overview

What We'll Learn?

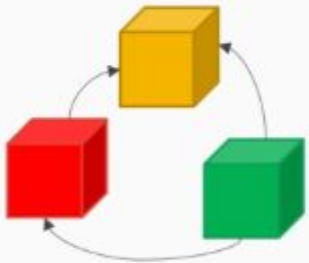
Prerequisites

What do you need to know?

Working knowledge
of Docker

Prerequisites

What do you need to know?



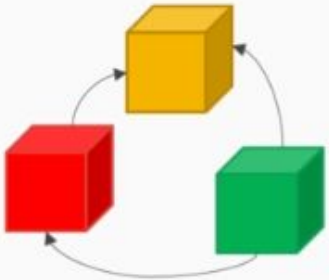
Microservice basics

docker

Working knowledge of Docker

Prerequisites

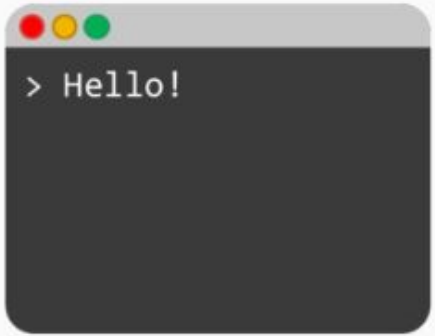
What do you need to know?



Microservice basics

docker

Working knowledge of Docker



Some command-line skills

Required Software

(Don‘t Worry, It‘s All Free!)

kubectl

Command-Line Client for Kubernetes

- Needs to run on your local machine
- Installation instructions: https://kubernetes.io/docs/tasks/tools/install-kubectl/
- Varies by operating system

Minikube

Kubernetes Running Locally

- Download from: https://github.com/kubernetes/minikube/releases
- Requires Oracle VirtualBox https://www.virtualbox.org/wiki/Downloads
- Alternative: Running Kubernetes cluster in cloud, for example, GKE or AKS

Your Favorite Code Editor

Vim or Emacs?

- Recommendation for the indecisive: Visual Studio Code
- https://code.visualstudio.com/download

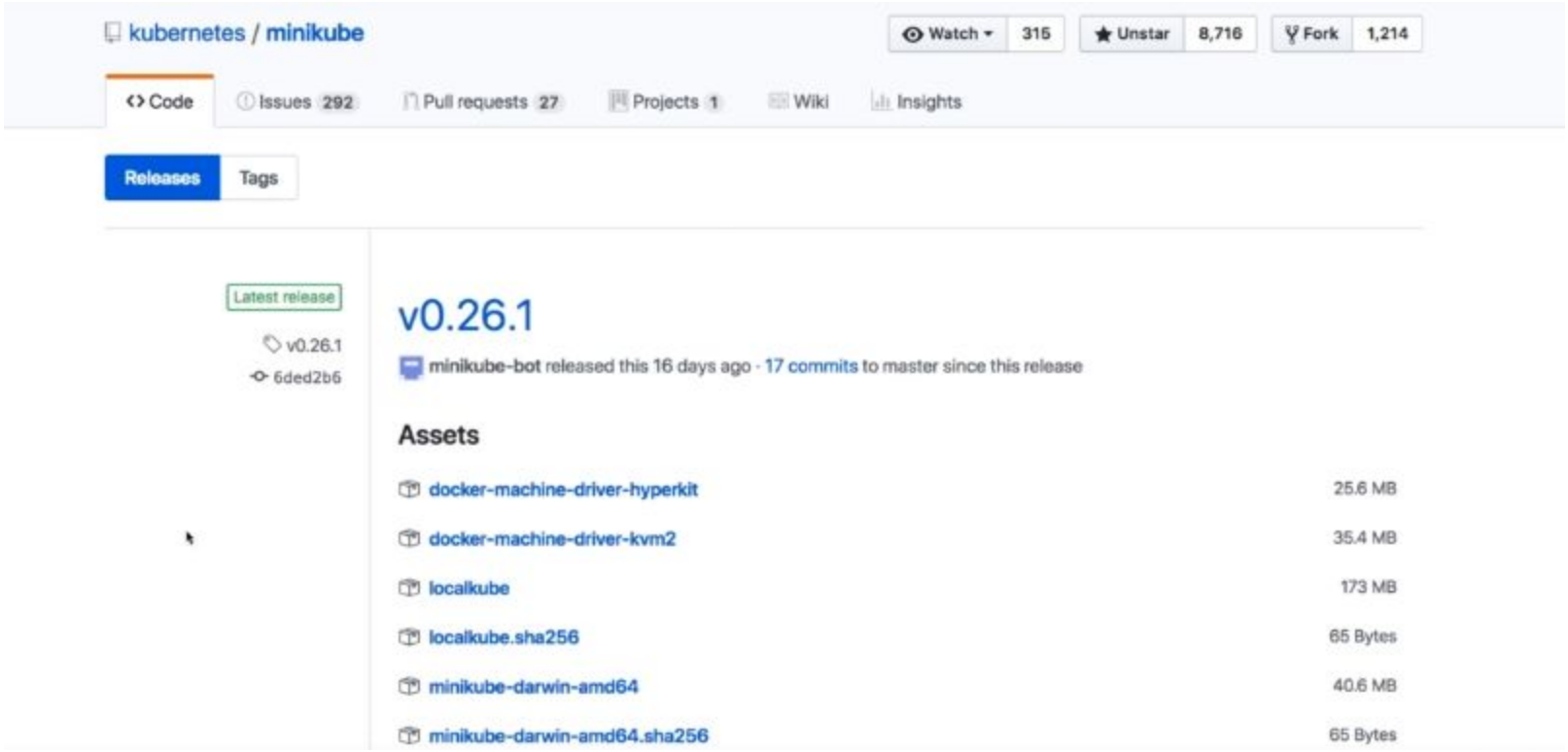
Setting up with Minikube

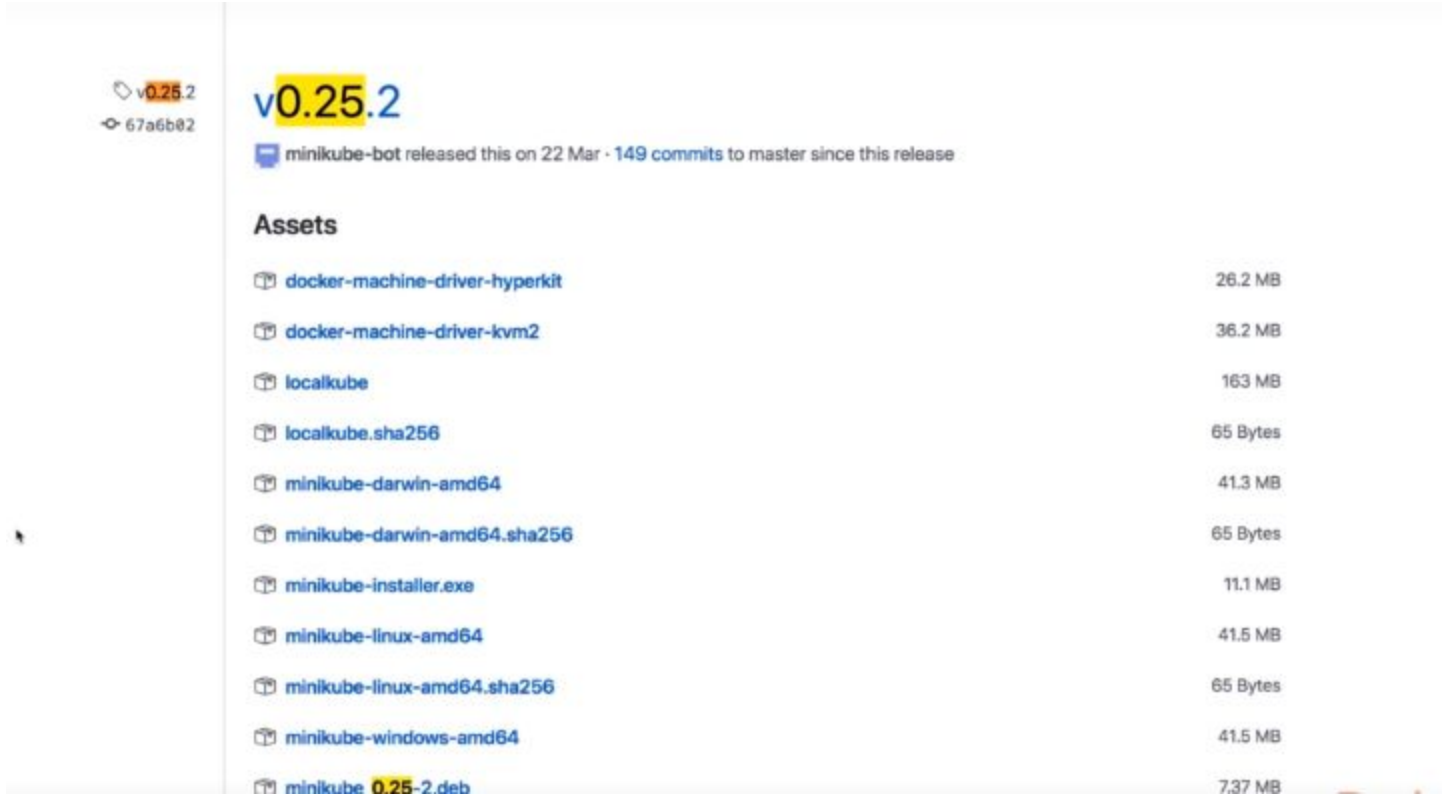
Prerequisite – Oracle VirtualBox

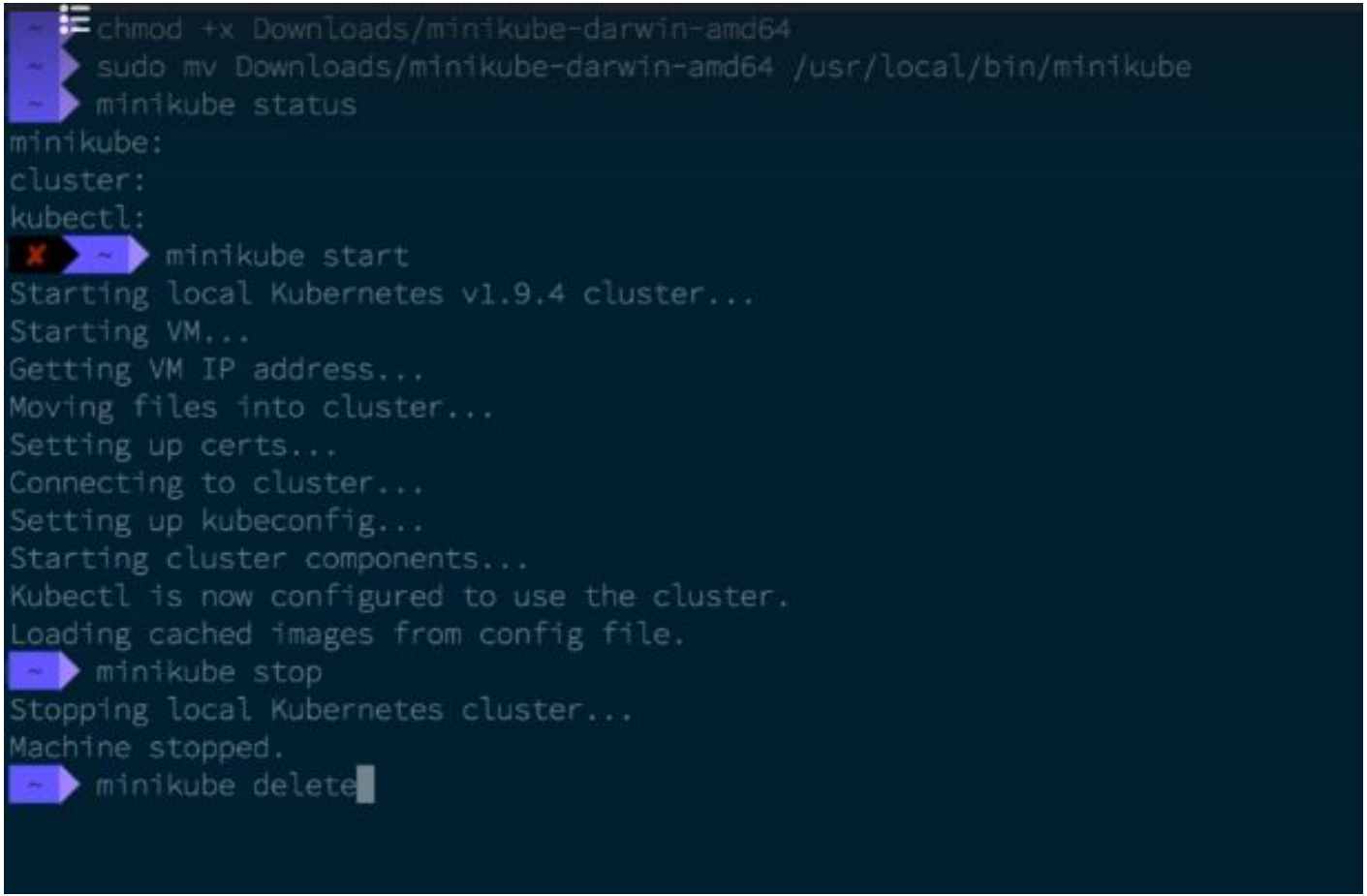
- Download at: https://www.virtualbox.org/wiki/Downloads

Minikube Download

https://github.com/kubernetes/minikube/releases


kubernetes / minikube
Watch 315
Unstar 8,716
Fork 1,214
Code
Issues 292
Pull requests 27
Projects 1
Wiki
Insights
Releases
Tags
Latest release
v0.26.1
6ded2b6
v0.26.1
minikube-bot released this 16 days ago · 17 commits to master since this release
Assets
docker-machine-driver-hyperkit 25.6 MB
docker-machine-driver-kvm2 35.4 MB
localkube 173 MB
localkube.sha256 65 Bytes
minikube-darwin-amd64 40.6 MB
minikube-darwin-amd64.sha256 65 Bytes


v0.25.2
67a6b02
v0.25.2
minikube-bot released this on 22 Mar · 149 commits to master since this release
Assets
docker-machine-driver-hyperkit 26.2 MB
docker-machine-driver-kvm2 36.2 MB
localkube 163 MB
localkube.sha256 65 Bytes
minikube-darwin-amd64 41.3 MB
minikube-darwin-amd64.sha256 65 Bytes
minikube-installer.exe 11.1 MB
minikube-linux-amd64 41.5 MB
minikube-linux-amd64.sha256 65 Bytes
minikube-windows-amd64 41.5 MB
minikube_0.25-2.deb 7.37 MB

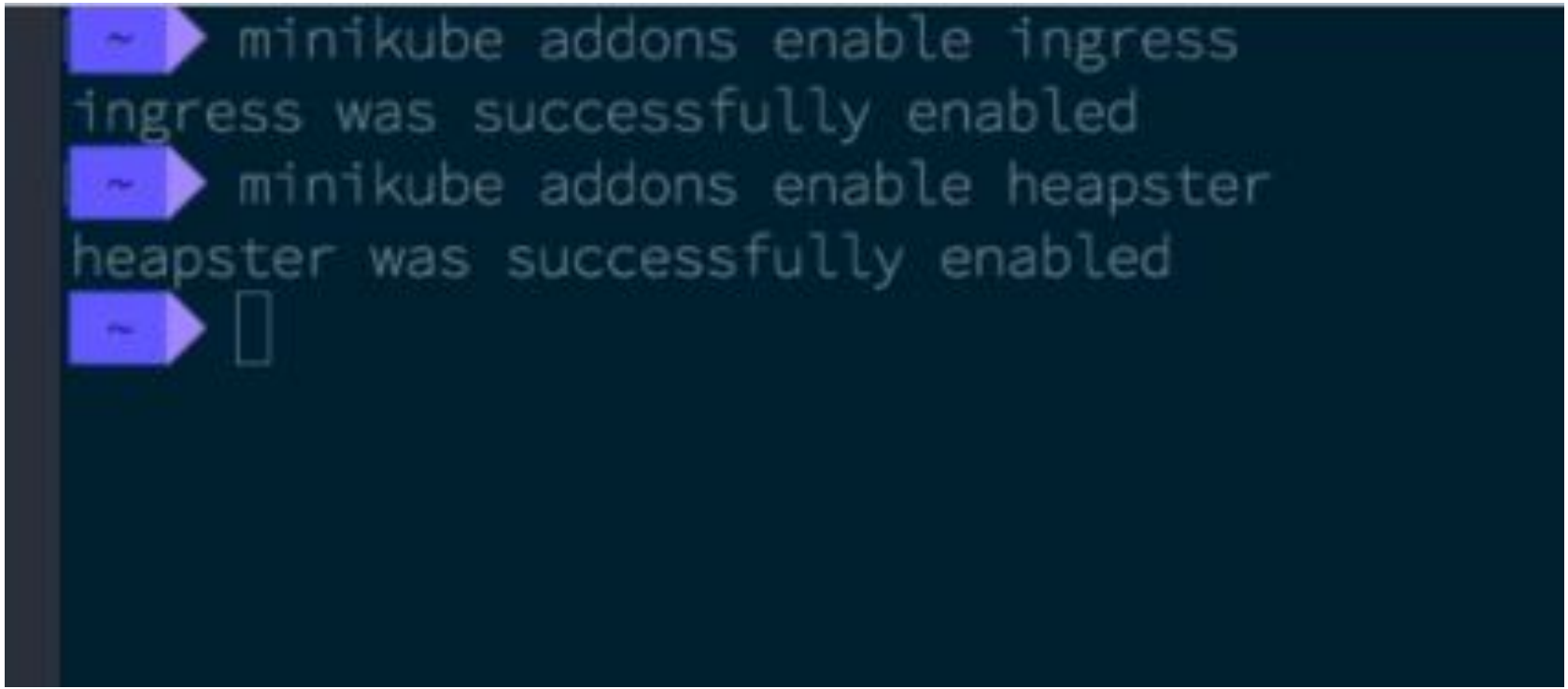

~ chmod +x Downloads/minikube-darwin-amd64
~ sudo mv Downloads/minikube-darwin-amd64 /usr/local/bin/minikube
~ minikube status
minikube:
cluster:
kubectl:
~ minikube start
Starting local Kubernetes v1.9.4 cluster...
Starting VM...
Getting VM IP address...
Moving files into cluster...
Setting up certs...
Connecting to cluster...
Setting up kubeconfig...
Starting cluster components...
Kubectl is now configured to use the cluster.
Loading cached images from config file.
~ minikube stop
Stopping local Kubernetes cluster...
Machine stopped.
~ minikube delete

Minikube Add-Ons

- Enabled per default:
  - DNS
  - Dashboard
  - Storage provisioner

Minikube Add-Ons

- Enabled per default:
  - DNS
  - Dashboard
  - Storage provisioner
- Useful to enable:
  - Ingress
  - Heapster


~ minikube addons enable ingress
ingress was successfully enabled
~ minikube addons enable heapster
heapster was successfully enabled
~

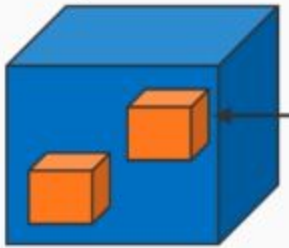
Deploying a Microservice

Kubernetes Core Concepts

Pods – The Basic Deployment Unit

Pod

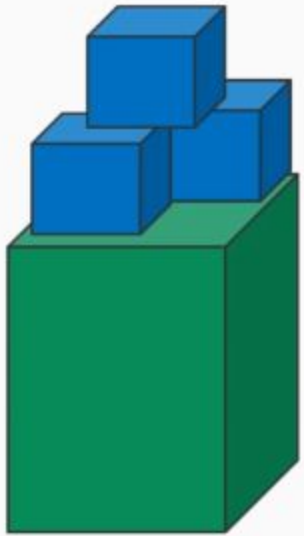
- Multiple containers (in practice seldom more than 1 or 2)
- Scheduled on **one** node



Container

- (Mostly) regular Docker (or rkt or CRIO) container
- Shared network namespace for all containers in Pod

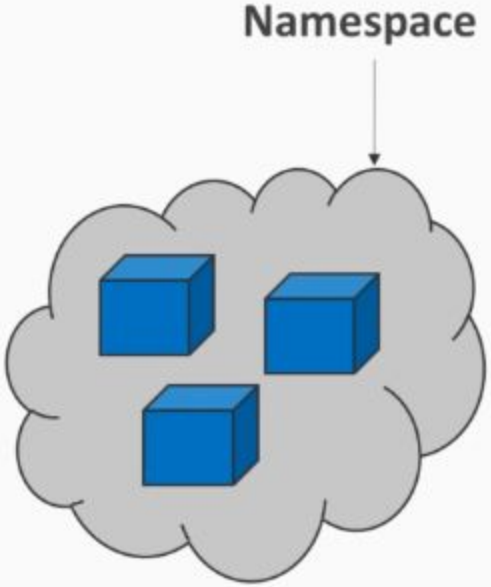
Nodes – It's Just Servers, After All



Node

- Individual server (virtualized or bare metal)
- Runs Docker engine
- Pods/containers started by Kubernetes

Namespaces – Resource Grouping and Isolation



- Groups resources (like Pods)
- In a fresh Kubernetes cluster: default and kube-system
- Purely virtual
- No additional isolation at container level

Defining a Pod

Assumptions

- You have a container image for your microservice available
  (In this example, we will use the nginx image as a dummy image)
- Your Kubernetes cluster is authorized to pull that image
  (Not an issue with the public nginx image; use imagePullSecret option otherwise)


~ kubectl get pods
No resources found.
~

Pod Example

- Full documentation: https://kubernetes.io/docs/concepts/workloads/pods/pod-overview/

```
apiVersion: v1
kind: Pod
metadata:
  name: test-pod
  namespace: default
spec:
  containers:
    - name: service
      image: nginx
      ports:
        - containerPort: 80
      env:
        - name: SOME_ENV_VAR
          value: Hello World
```

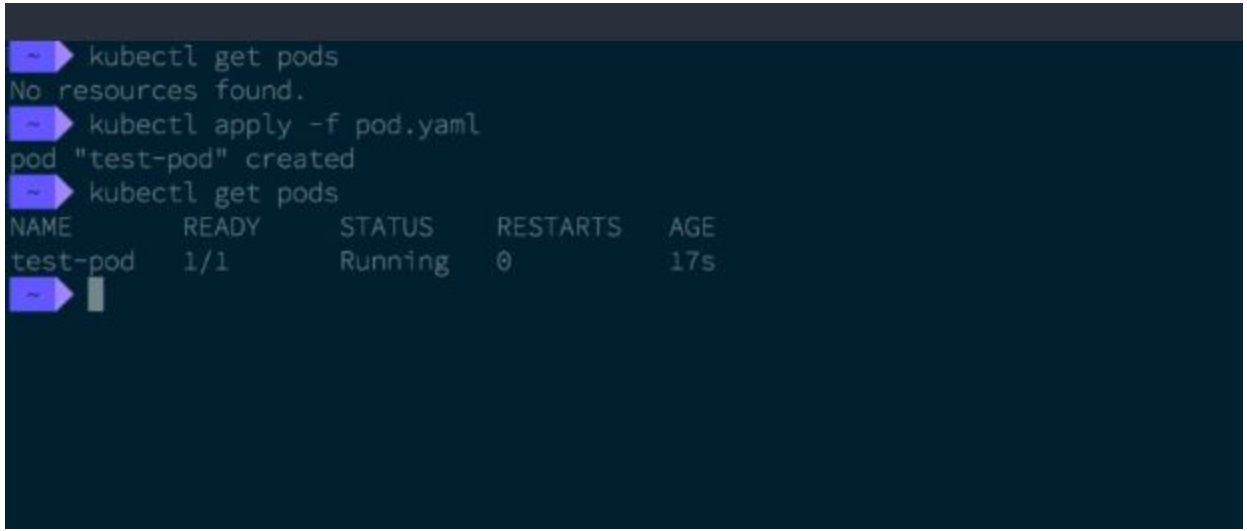
Pod name

Pod Example

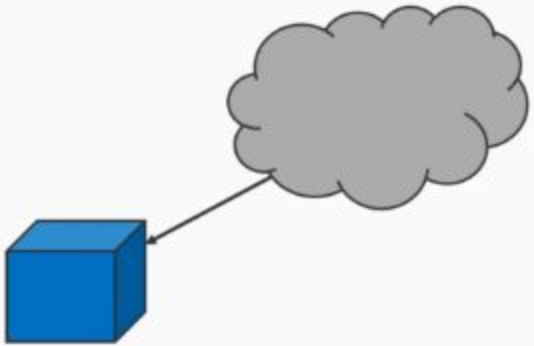
- Full documentation: https://kubernetes.io/docs/concepts/workloads/pods/pod-overview/

```
apiVersion: v1
kind: Pod
metadata:
  name: test-pod
  namespace: default
spec:
  containers:
    - name: service
      image: nginx
      ports:
        - containerPort: 80
      env:
        - name: SOME_ENV_VAR
          value: Hello World
```

Container specification

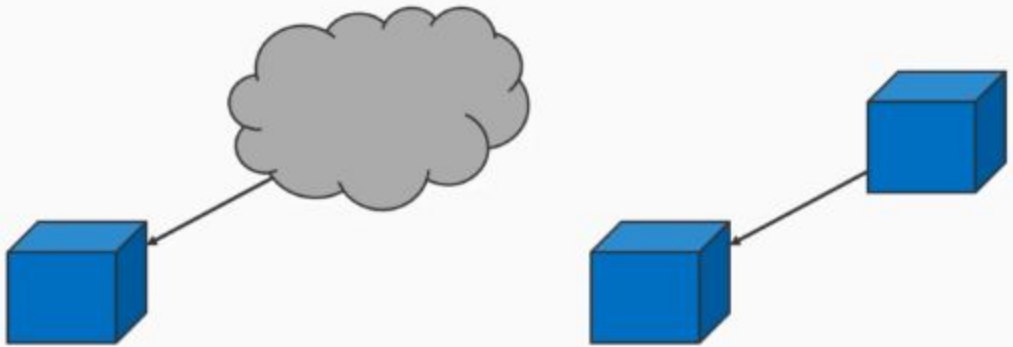

~ kubectl get pods
No resources found.
~ kubectl apply -f pod.yaml
pod "test-pod" created
~ kubectl get pods
NAME READY STATUS RESTARTS AGE
test-pod 1/1 Running 0 17s
~

Open Issues



Network connectivity
from external sources

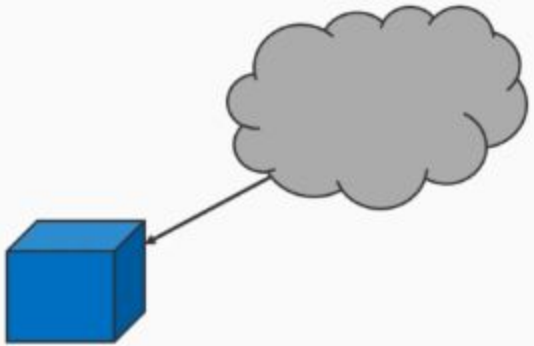
Open Issues



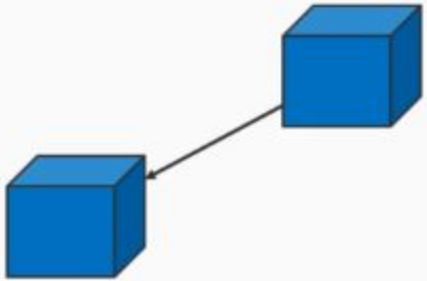
Network connectivity
from external sources

Network connectivity
to/from other services

Open Issues



Network connectivity
from external sources

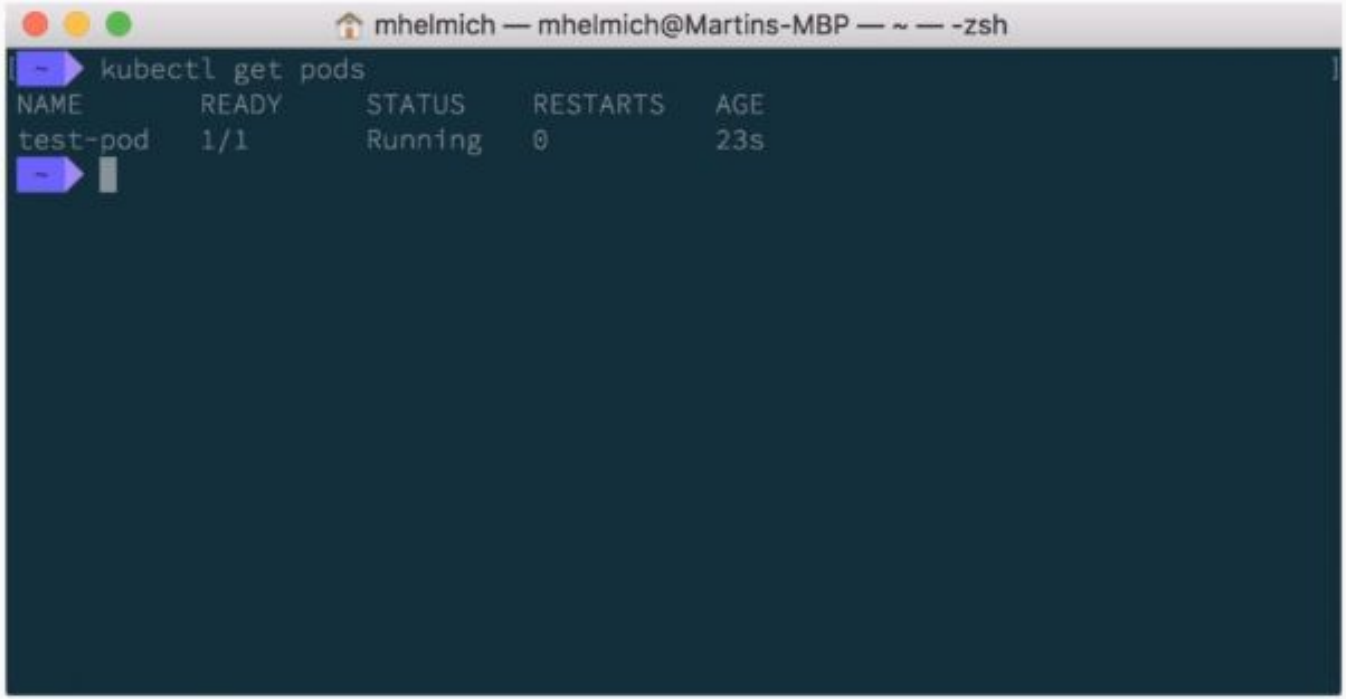


Network connectivity
to/from other services

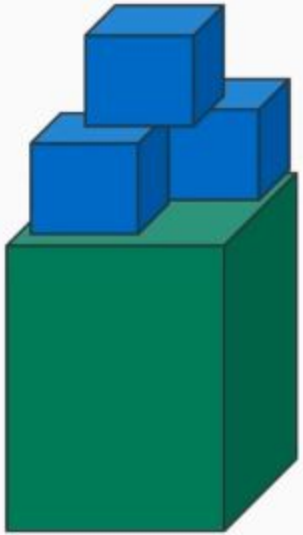
Solution
Service and Ingress resources, covered in Section 2 of this course

Service Resiliency and Scalability

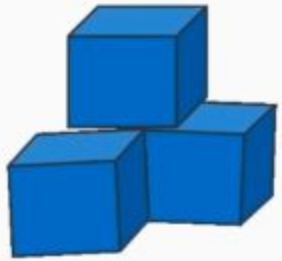
What Happened So Far?

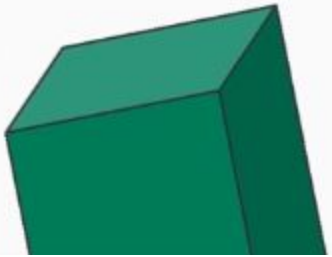


Pod Limitations



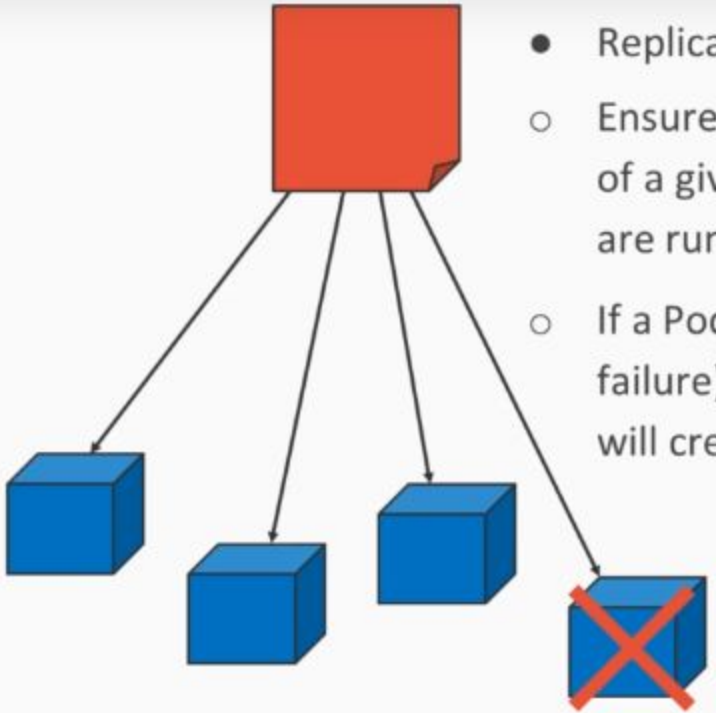
- Each Pod is assigned to a specific node in your cluster
  - If a Pod crashes, it will be restarted
  - If a Node crashes, the Pods running on it will be lost, as well



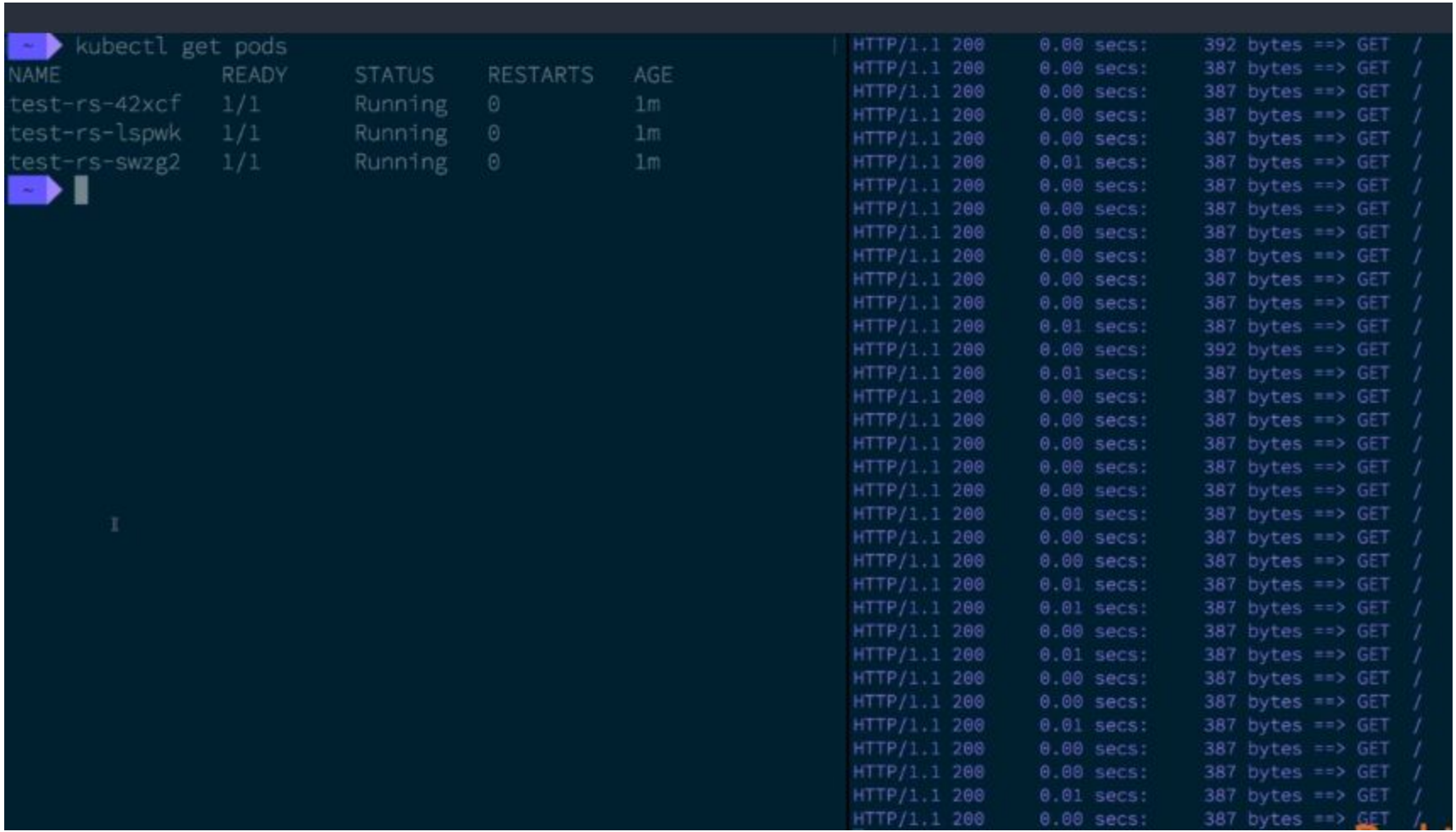


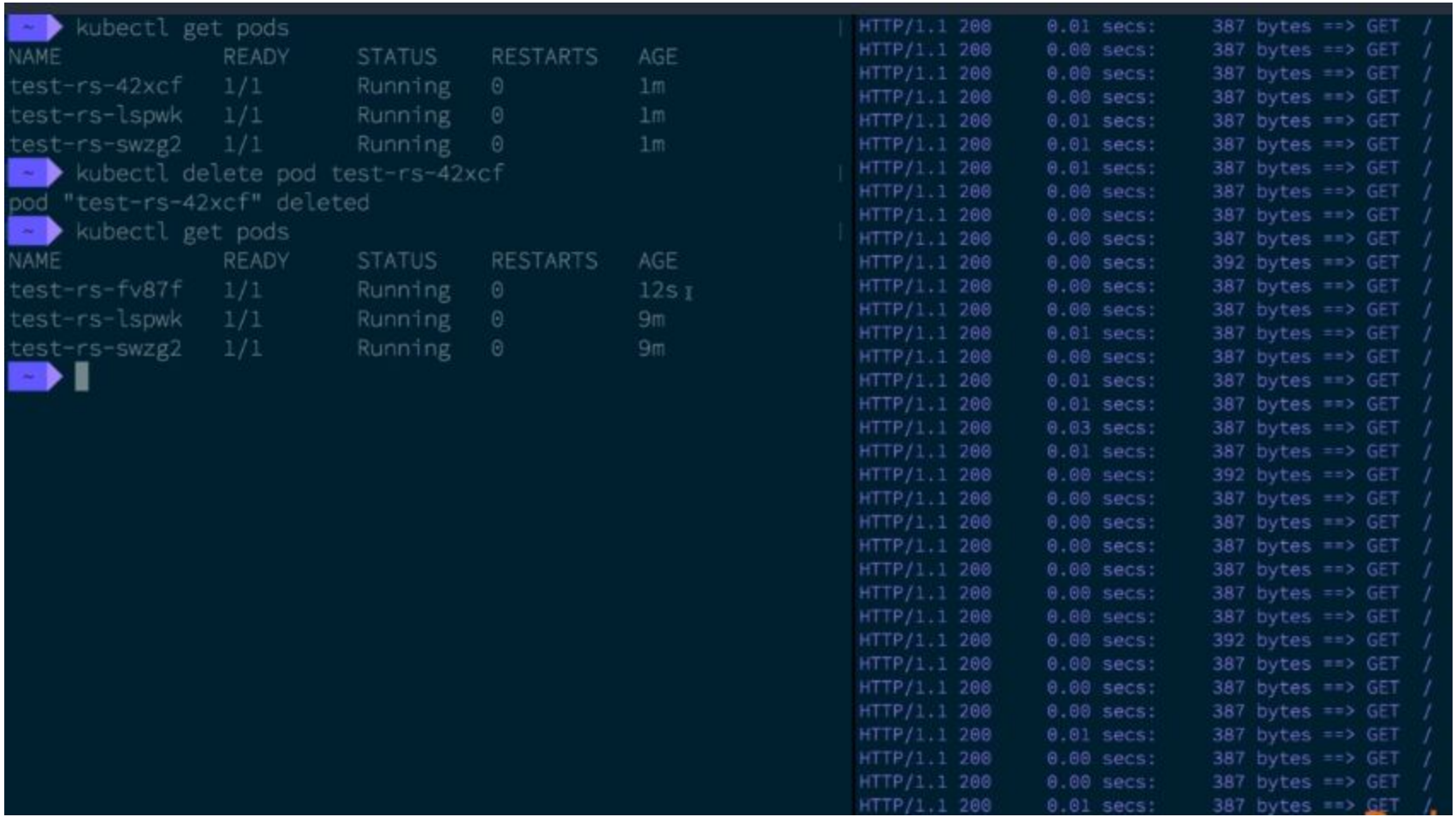
- Each Pod is assigned to a specific node in your cluster
  - If a Pod crashes, it will be restarted
  - If a Node crashes, the Pods running on it will be lost, as well

High-Level Controllers



- ReplicaSet
  - Ensure that N instances of a given type of Pod are running at all times
  - If a Pod is lost (Node failure), the ReplicaSet will create a new one


~ kubectl get pods
NAME READY STATUS RESTARTS AGE
test-rs-42xcf 1/1 Running 0 1m
test-rs-lspwk 1/1 Running 0 1m
test-rs-swzg2 1/1 Running 0 1m
~
HTTP/1.1 200 0.00 secs: 392 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 392 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /

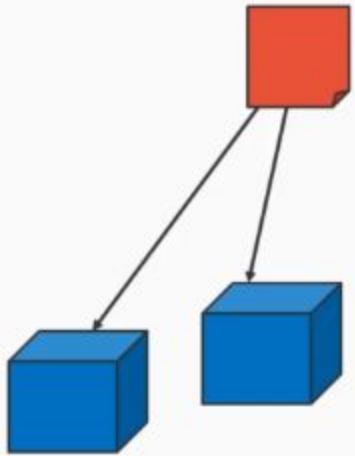

~ kubectl get pods
NAME READY STATUS RESTARTS AGE
test-rs-42xcf 1/1 Running 0 1m
test-rs-lspwk 1/1 Running 0 1m
test-rs-swzg2 1/1 Running 0 1m
~ kubectl delete pod test-rs-42xcf
pod "test-rs-42xcf" deleted
~ kubectl get pods
NAME READY STATUS RESTARTS AGE
test-rs-fv87f 1/1 Running 0 12s
test-rs-lspwk 1/1 Running 0 9m
test-rs-swzg2 1/1 Running 0 9m
~
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 392 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.03 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 392 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 392 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.00 secs: 387 bytes ==> GET /
HTTP/1.1 200 0.01 secs: 387 bytes ==> GET /

Conclusion

Single Pod

- disposable
- not resilient by itself
- not scalable by itself

Conclusion



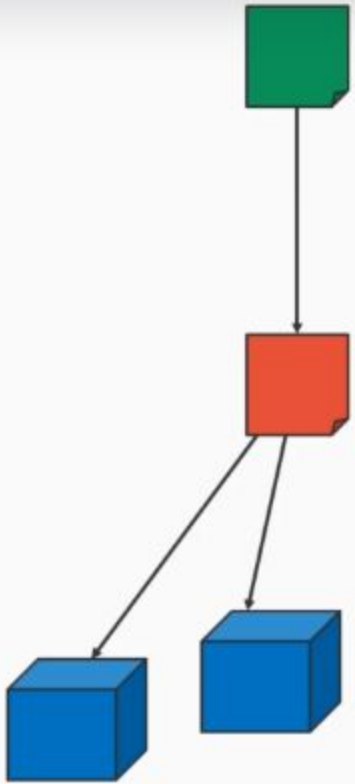
ReplicaSet

- Manages Pods
- Self-healing and scalable

Single Pod

- disposable
- not resilient by itself
- not scalable by itself

Conclusion



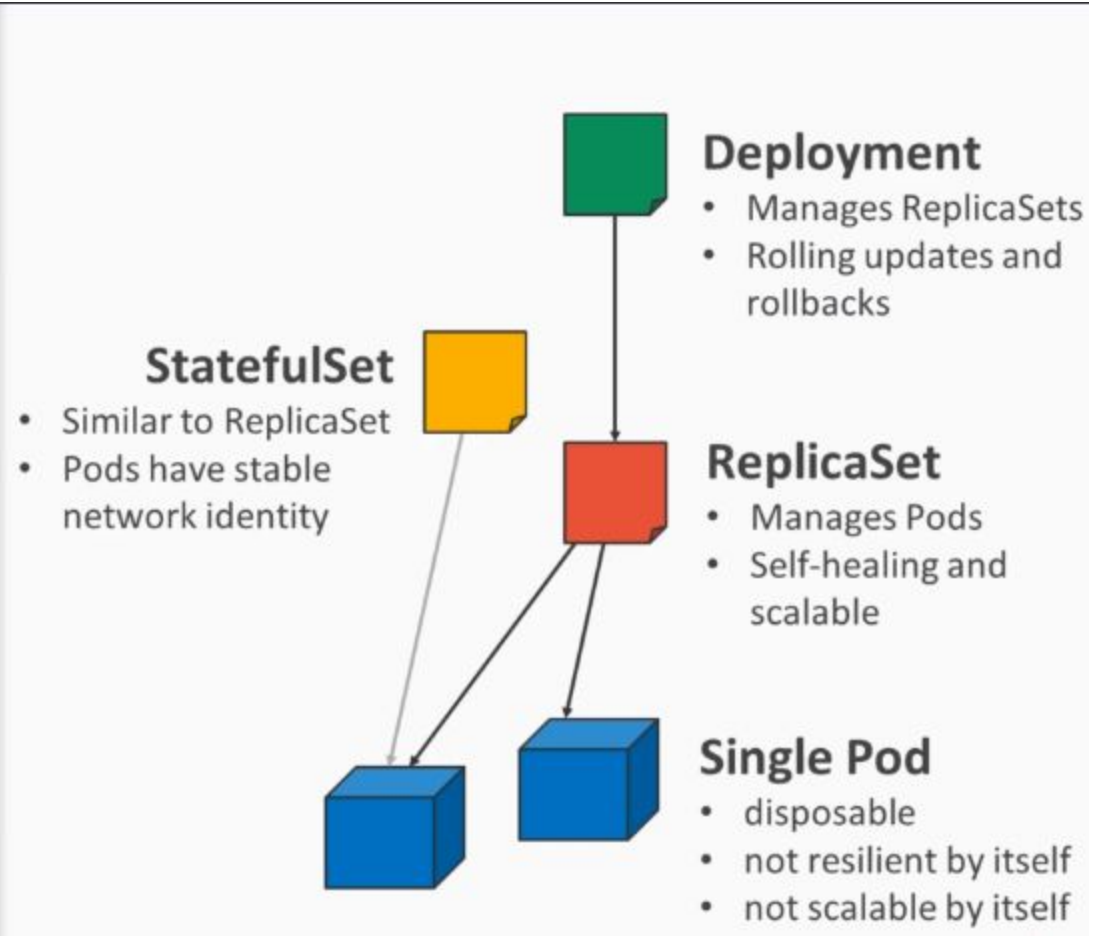
Deployment
- Manages ReplicaSets
- Rolling updates and rollbacks

ReplicaSet
- Manages Pods
- Self-healing and scalable

Single Pod
- disposable
- not resilient by itself
- not scalable by itself

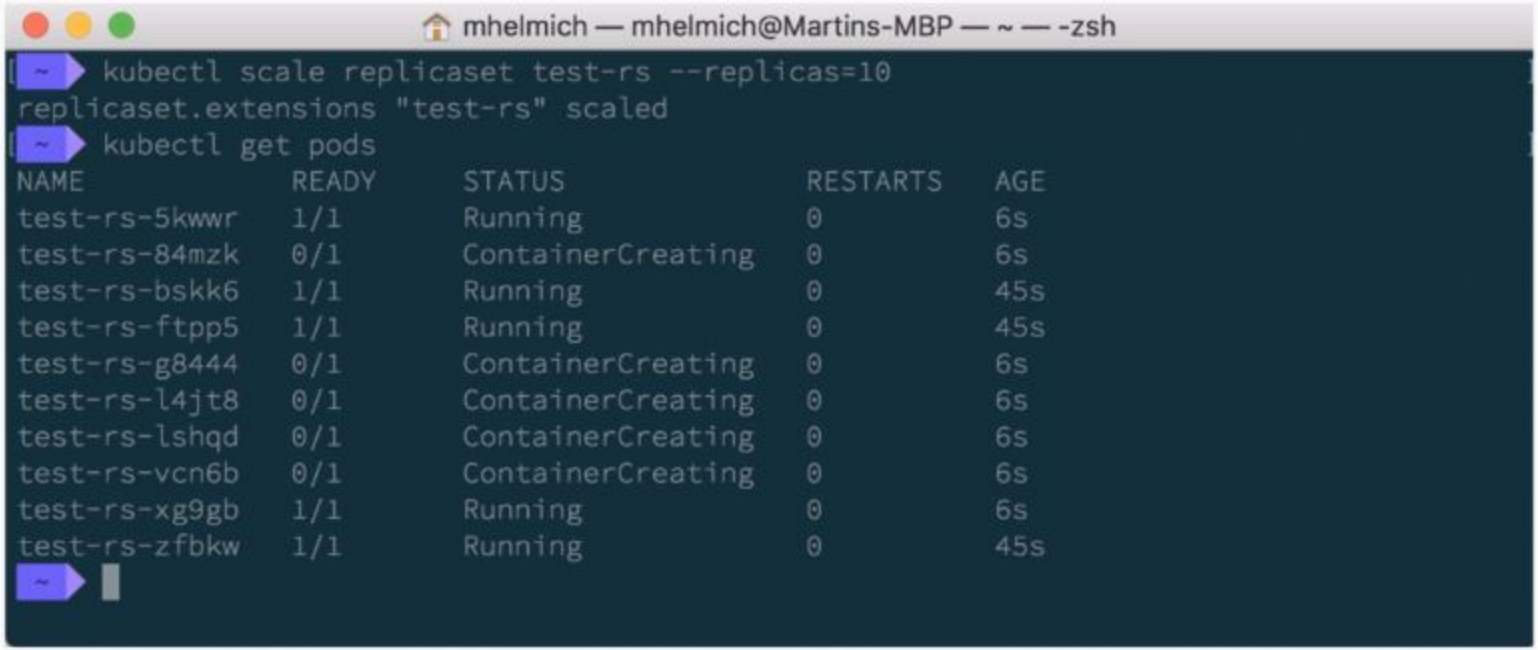
Conclusion

- Rule of thumb: Never manually create a Pod
- Instead rely on controllers (ReplicaSet, Deployment, StatefulSet) to manage Pods for you
- More about ReplicaSets and Deployments in Section 2
- More about StatefulSets in Section 3



The Kubernetes Deployment Model

What Happened So Far?



Labels

Label All the Things!

- In Kubernetes, every object can have arbitrary labels:

```
apiVersion: v1
kind: Pod
metadata:
  name: some-pod
  labels:
    app: my-great-microservice
    stage: production
```

Defining a ReplicaSet

ReplicaSet properties

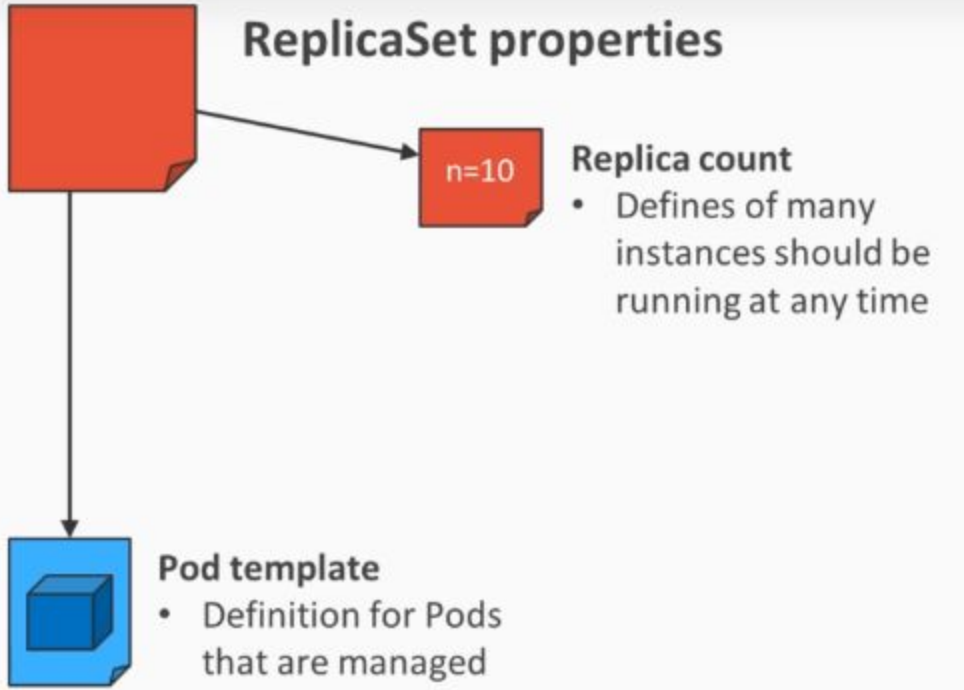
Defining a ReplicaSet

ReplicaSet properties

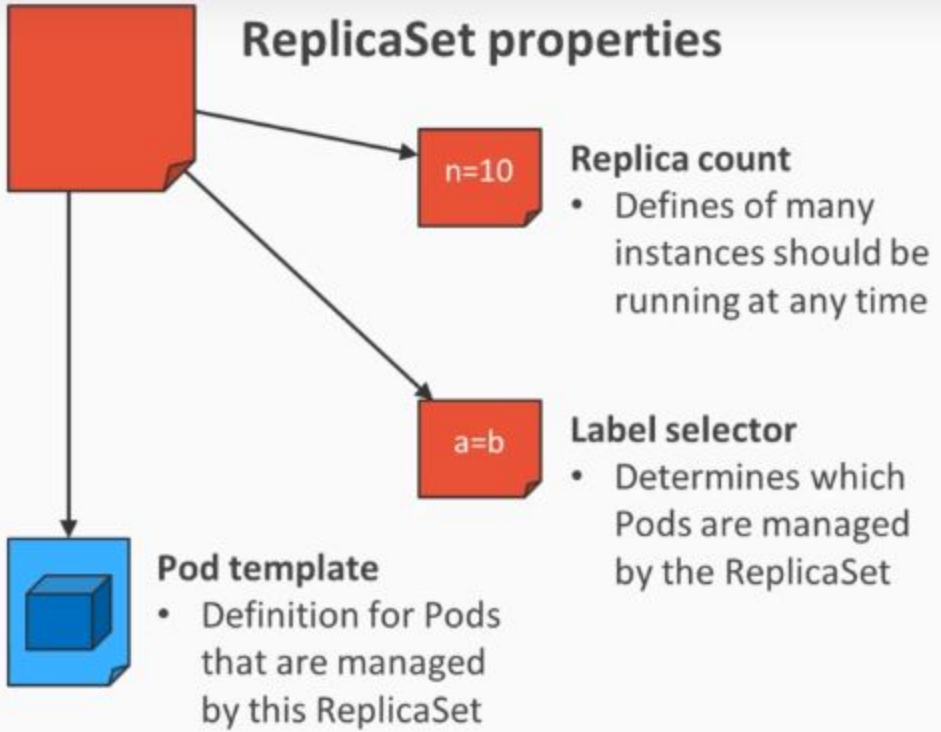
Pod template

- Definition for Pods that are managed by this ReplicaSet

Defining a ReplicaSet



Defining a ReplicaSet



ReplicaSet Example

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: my-service
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-service
  template:
    metadata:
      labels:
        app: my-service
    spec:
      containers:
      - name: service
        image: nginx
```

ReplicaSet Example

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: my-service
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-service
  template:
    metadata:
      labels:
        app: my-service
    spec:
      containers:
      - name: service
        image: nginx
```

ReplicaSet name

ReplicaSet Example

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: my-service
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-service
  template:
    metadata:
      labels:
        app: my-service
    spec:
      containers:
      - name: service
        image: nginx
```

Replica count

ReplicaSet Example

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: my-service
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-service
  template:
    metadata:
      labels:
        app: my-service
    spec:
      containers:
      - name: service
        image: nginx
```

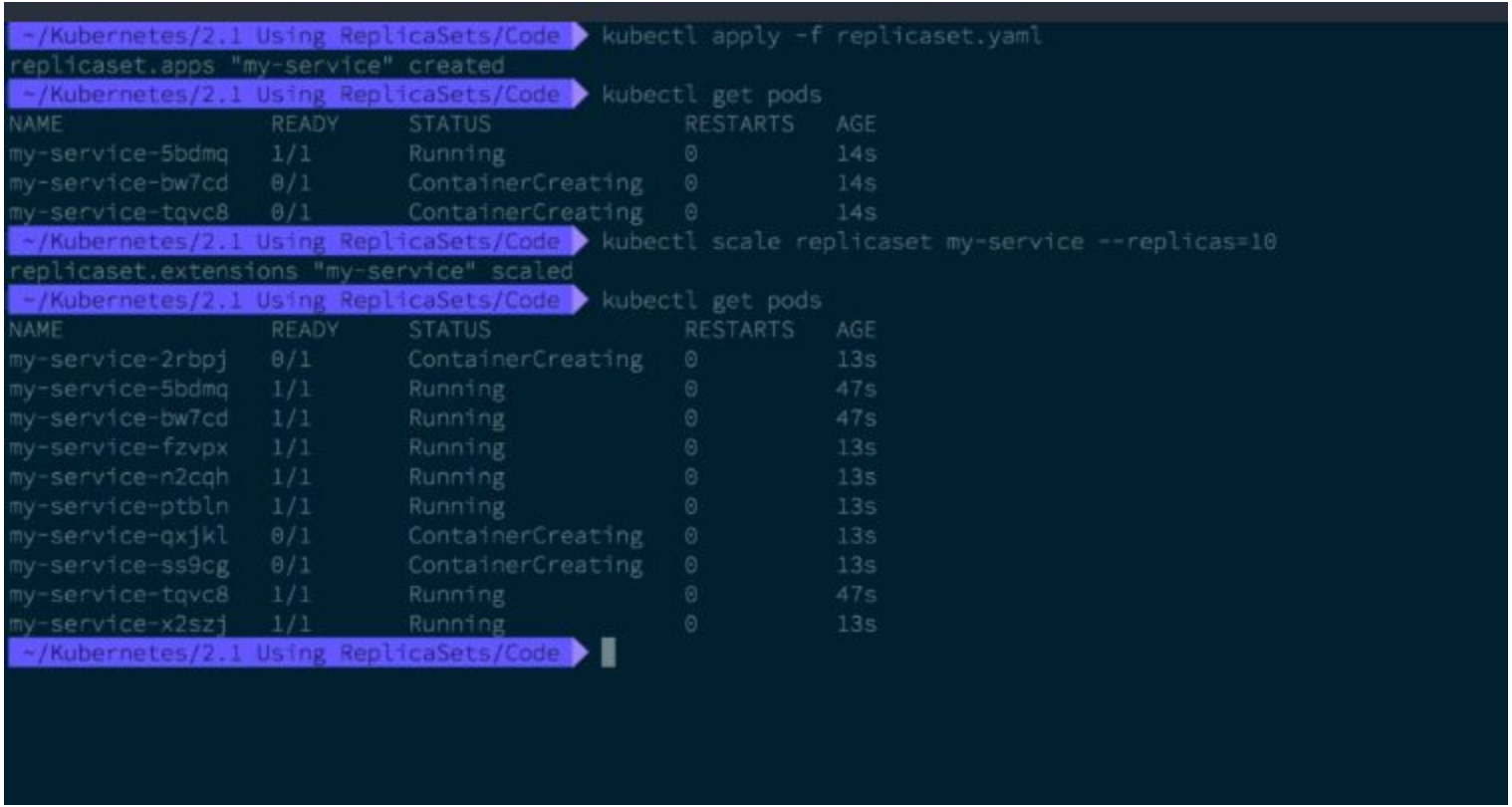
Template used for creating Pods (same Pod specification as in previous video)

eplicaSet Example

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: my-service
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-service
  template:
    metadata:
      labels:
        app: my-service
    spec:
      containers:
      - name: service
        image: nginx
```

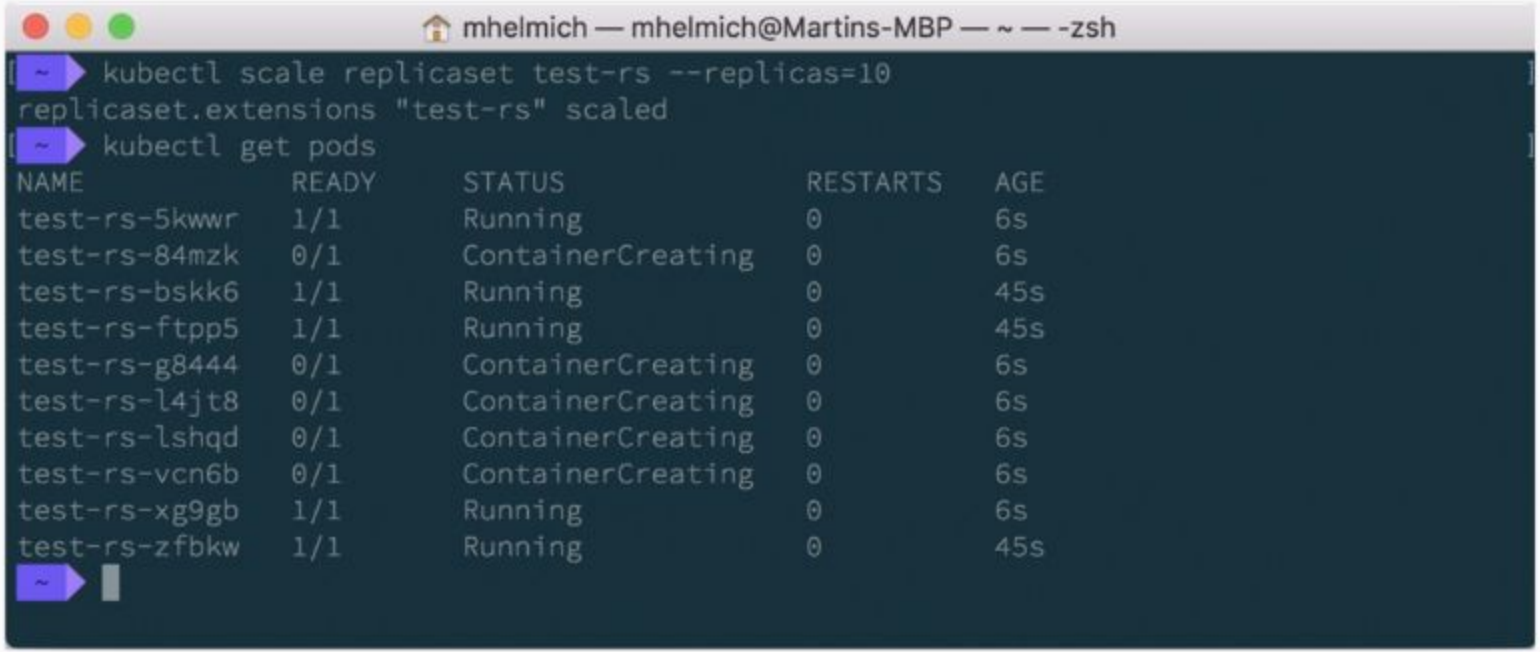
Label selector
Used by the ReplicaSet to determine which Pods are managed by this ReplicaSet

Pod metadata
MUST contain the selected labels from the selector


~/Kubernetes/2.1 Using ReplicaSets/Code kubectl apply -f replicaset.yaml
replicaset.apps "my-service" created
~/Kubernetes/2.1 Using ReplicaSets/Code kubectl get pods
NAME READY STATUS RESTARTS AGE
my-service-5bdmq 1/1 Running 0 14s
my-service-bw7cd 0/1 ContainerCreating 0 14s
my-service-tqvc8 0/1 ContainerCreating 0 14s
~/Kubernetes/2.1 Using ReplicaSets/Code kubectl scale replicaset my-service --replicas=10
replicaset.extensions "my-service" scaled
~/Kubernetes/2.1 Using ReplicaSets/Code kubectl get pods
NAME READY STATUS RESTARTS AGE
my-service-2rbpj 0/1 ContainerCreating 0 13s
my-service-5bdmq 1/1 Running 0 47s
my-service-bw7cd 1/1 Running 0 47s
my-service-fzvpx 1/1 Running 0 13s
my-service-n2cqh 1/1 Running 0 13s
my-service-ptbln 1/1 Running 0 13s
my-service-qxjkl 0/1 ContainerCreating 0 13s
my-service-ss9cg 0/1 ContainerCreating 0 13s
my-service-tqvc8 1/1 Running 0 47s
my-service-x2szj 1/1 Running 0 13s
~/Kubernetes/2.1 Using ReplicaSets/Code

Connecting Services

What Happened So Far?



Pod Networking

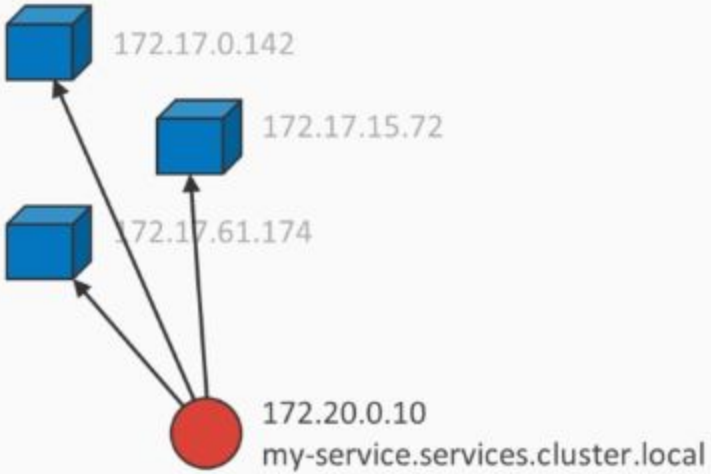
Pods

- Have IP addresses
- These are routed within the cluster (no external connectivity)

But

- Pods are short-lived
- Addresses may change frequently
- No DNS for Pod IPs
- No load balancing

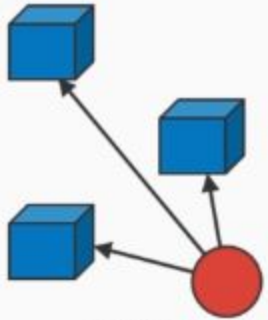
Services to the Rescue!



Services

- Selects a set of Pods by label
- One stable IP and DNS name for that group of Pods
- Round-robin load balancing
- Automatically adds/removes Pods (for example, managed by ReplicaSet)

Service Types

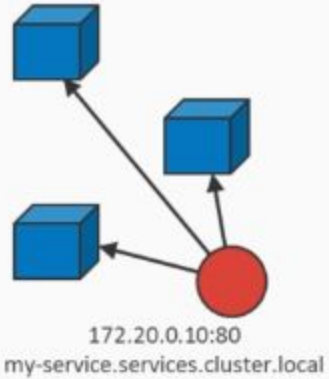


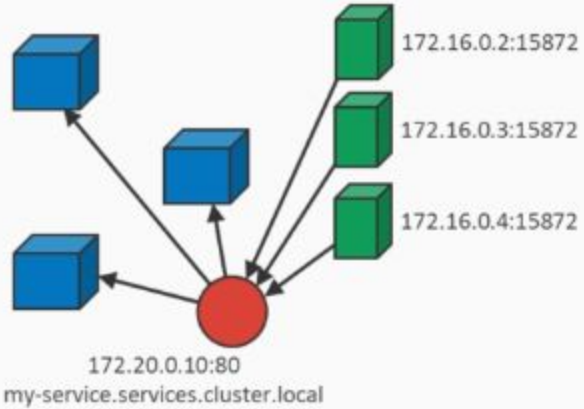
172.20.0.10:80
my-service.services.cluster.local

ClusterIP

- Stable internal IP
- Stable internal DNS

Service Types





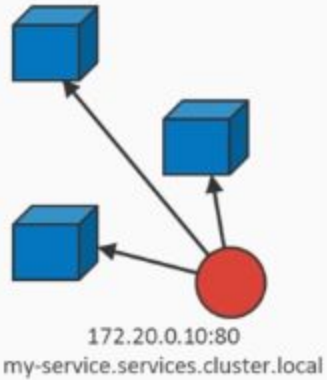
ClusterIP

- Stable internal IP
- Stable internal DNS

NodePort

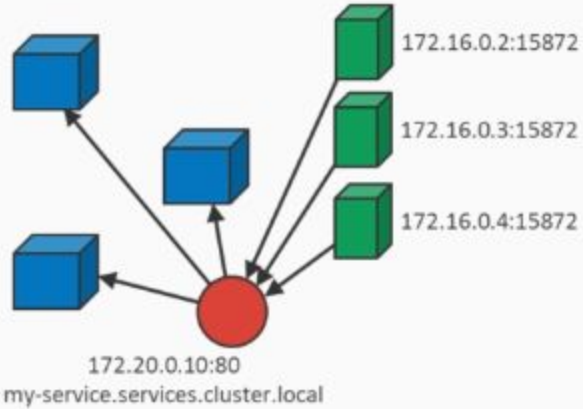
- As ClusterIP, but additionally
- Every Node gets a public TCP port forwarding to that service

Service Types



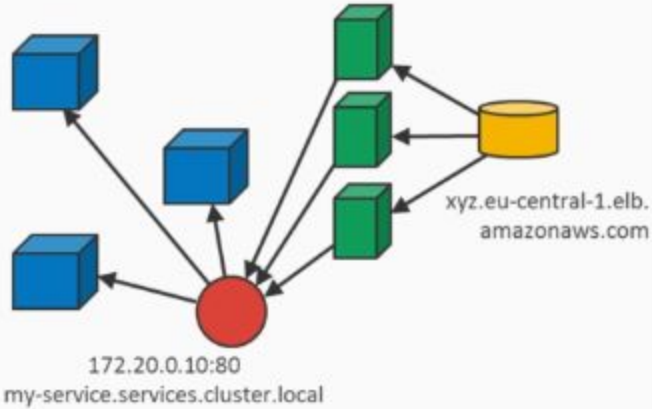
ClusterIP

- Stable internal IP
- Stable internal DNS



NodePort

- As ClusterIP, but additionally
- Every Node gets a public TCP port forwarding to that service



LoadBalancer

- As NodePort, but additionally
- A load balancer (must be supported by cloud provider) is created to allow external incoming traffic

Service Example

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  type: ClusterIP
  selector:
    app: my-service
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

Service Example

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  type: ClusterIP
  selector:
    app: my-service
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

Service name; will also be the DNS name

Service Example

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  type: ClusterIP
  selector:
    app: my-service
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

Service type

Service Example

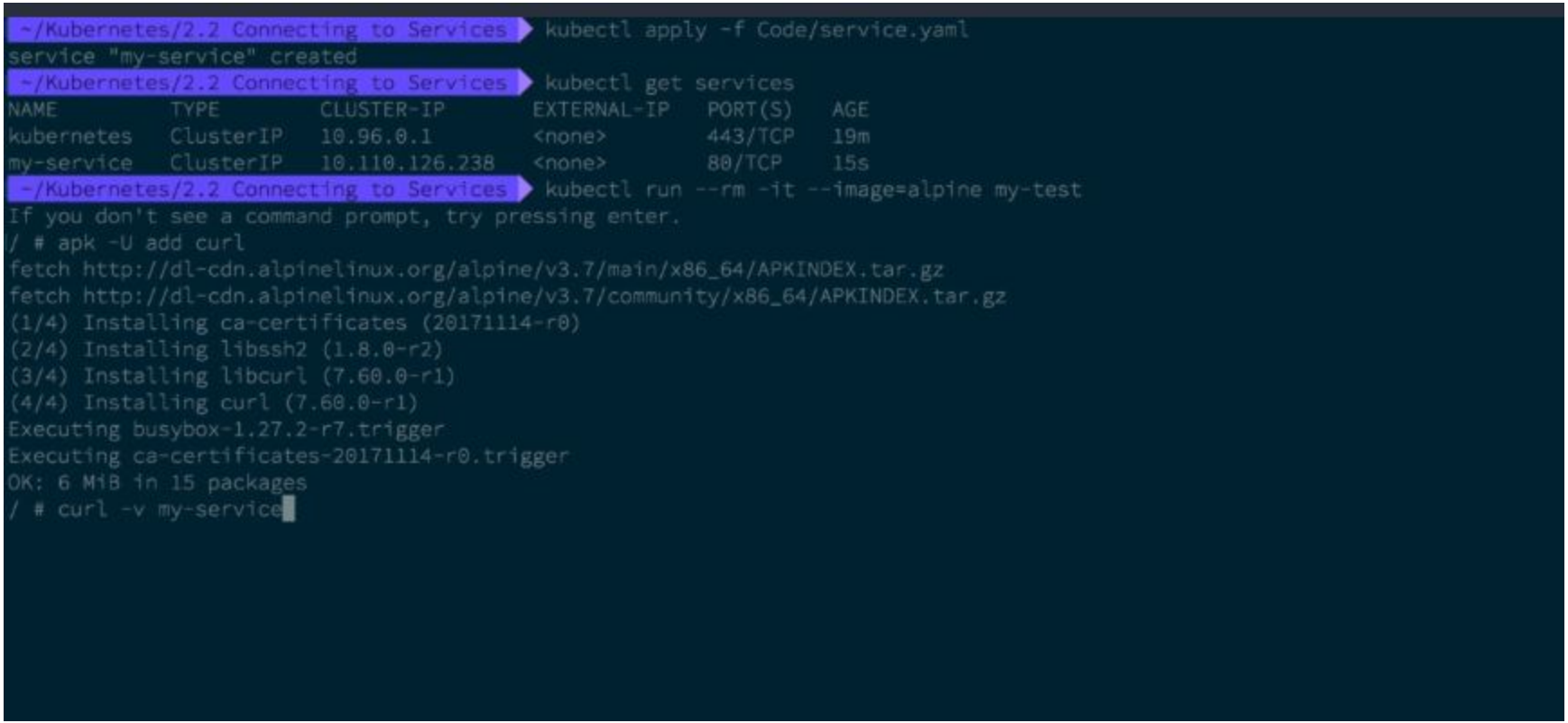
```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  type: ClusterIP
  selector:
    app: my-service
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

Label selector
Used to determine which Pods traffic will be forwarded to

Service Example

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  type: ClusterIP
  selector:
    app: my-service
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

Ports that should be forwarded by the service (supports TCP+UDP)


~/Kubernetes/2.2 Connecting to Services kubectl apply -f Code/service.yaml
service "my-service" created
~/Kubernetes/2.2 Connecting to Services kubectl get services
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 19m
my-service ClusterIP 10.110.126.238 <none> 80/TCP 15s
~/Kubernetes/2.2 Connecting to Services kubectl run --rm -it --image=alpine my-test
If you don't see a command prompt, try pressing enter.
/ # apk -U add curl
fetch http://dl-cdn.alpinelinux.org/alpine/v3.7/main/x86_64/APKINDEX.tar.gz
fetch http://dl-cdn.alpinelinux.org/alpine/v3.7/community/x86_64/APKINDEX.tar.gz
(1/4) Installing ca-certificates (20171114-r0)
(2/4) Installing libssh2 (1.8.0-r2)
(3/4) Installing libcurl (7.60.0-r1)
(4/4) Installing curl (7.60.0-r1)
Executing busybox-1.27.2-r7.trigger
Executing ca-certificates-20171114-r0.trigger
OK: 6 MiB in 15 packages
/ # curl -v my-service

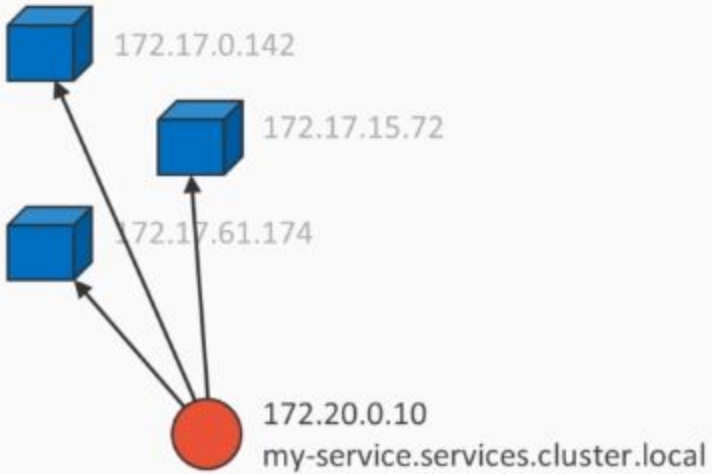
```
< Content-Length: 612
< Last-Modified: Mon, 09 Apr 2018 16:01:09 GMT
< Connection: keep-alive
< ETag: "5acb8e45-264"
< Accept-Ranges: bytes
<
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
* Connection #0 to host my-service left intact
/ #
```

Service Example

```
apiVersion: v1
 kind: Service
 metadata:
   name: my-service
 spec:
-  type: ClusterIP
+  type: LoadBalancer
   selector:
     app: my-service
   ports:
     - protocol: TCP
       port: 80
       targetPort: 80
```

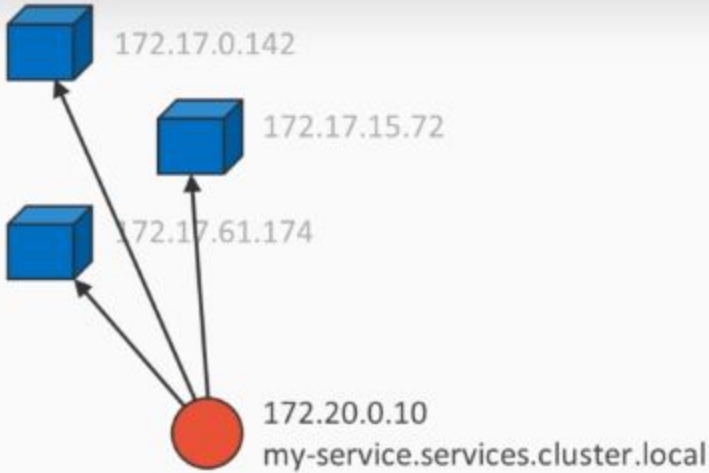
HTTP Connectivity with Ingress

Previously on



Services

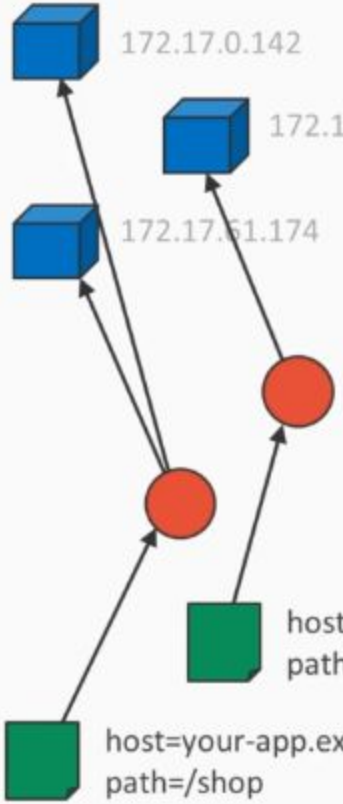
Service Limitations



Services

- Do raw TCP/UDP forwarding
- Do not know about application procotols like HTTP

Ingress Resources



Ingress

- Defines request routing for HTTP requests to services
- Works at application layer
- Supports path and host-based routing, caching, authentication, and more

Ingress Example

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: my-ingress
spec:
  rules:
  - host: your-app.example
    http:
      paths:
      - path: /shop
        backend:
          serviceName: my-service
          servicePort: 80
      - path: /home
        backend:
          serviceName: my-other-
service
          servicePort: 80
```

Ingress Example

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: my-ingress
spec:
  rules:
  - host: your-app.example
    http:
      paths:
      - path: /shop
        backend:
          serviceName: my-service
          servicePort: 80
      - path: /home
        backend:
          serviceName: my-other-
service
          servicePort: 80
```

Host rules; one per HTTP host name

Ingress Controllers

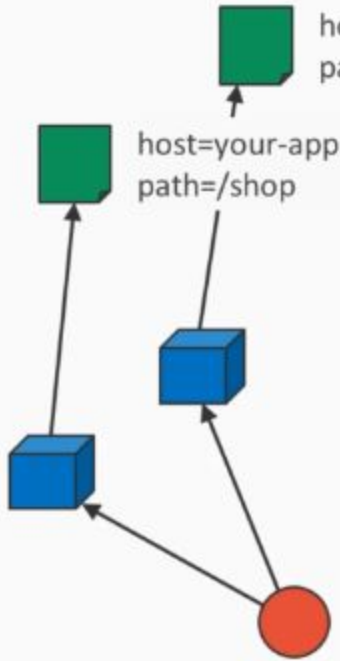
host=your-app.example
path=/home

host=your-app.example
path=/shop

Ingress Controllers

- Ingress definitions are virtual; they require an **ingress controller** that accepts HTTP connections and forwards requests where they should go

Ingress Controllers



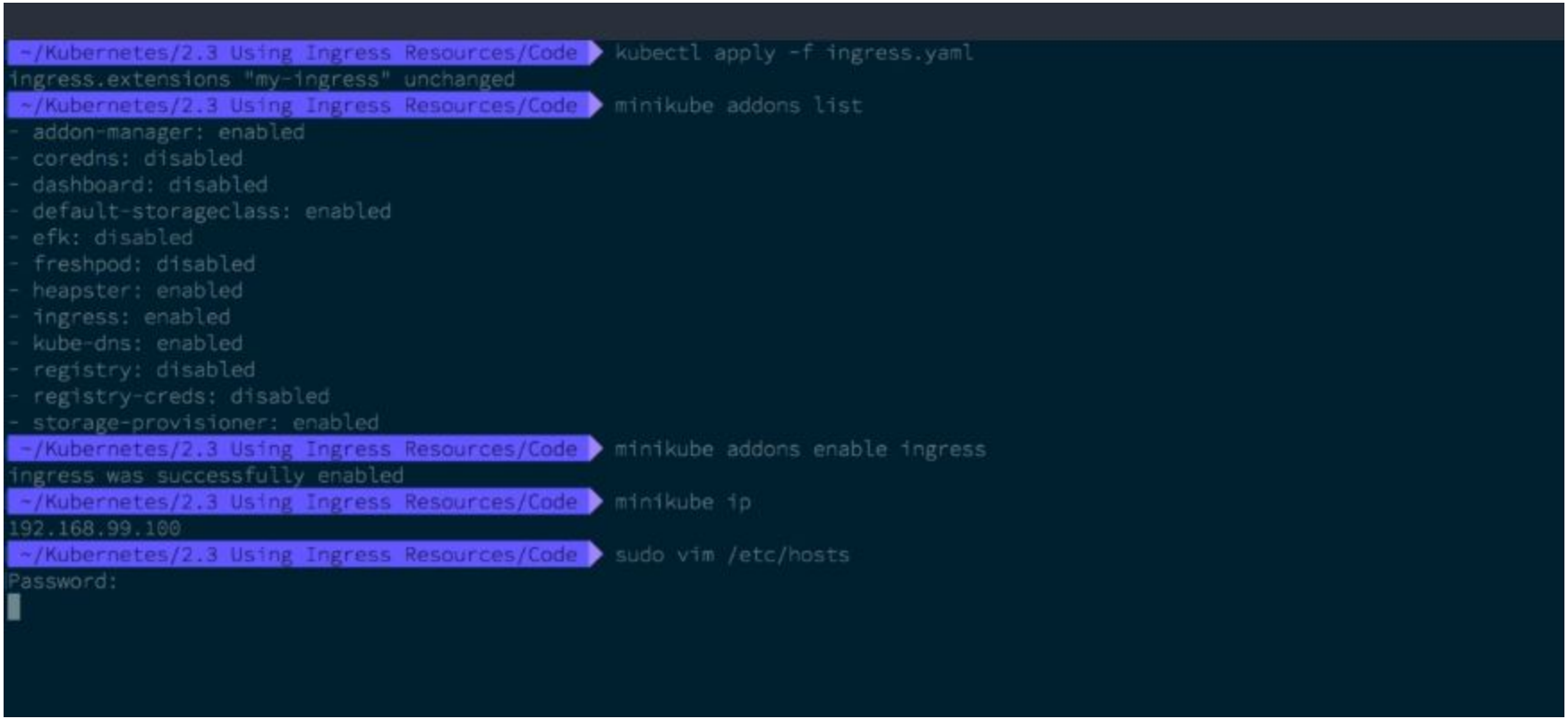
Ingress Controllers

- Ingress definitions are virtual; they require an **ingress controller** that accepts HTTP connections and forwards requests where they should go
- You can deploy an Ingress controller in your Kubernetes cluster, using regular Deployment and Service definitions (have a look at the “NGINX ingress controller“)

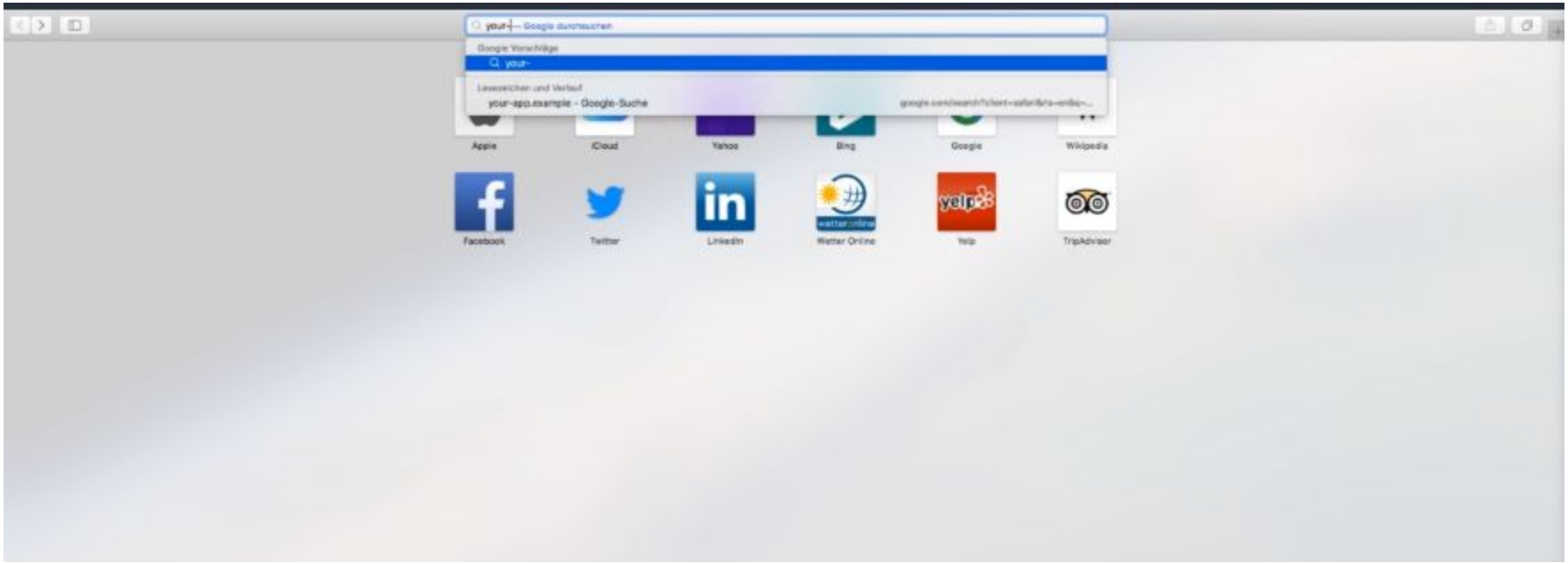
Ingress Controllers

Ingress Controllers

- Ingress definitions are virtual; they require an ingress controller that accepts HTTP connections and forwards requests where they should go
- You can deploy an Ingress controller in your Kubernetes clusters, using regular Deployment and Service definitions (have a look at the "NGINX ingress controller")
- Some cloud providers (like GKE) also provide their own ready-to-use Ingress controllers for you


~/Kubernetes/2.3 Using Ingress Resources/Code kubectl apply -f ingress.yaml
ingress.extensions "my-ingress" unchanged
~/Kubernetes/2.3 Using Ingress Resources/Code minikube addons list
- addon-manager: enabled
- coredns: disabled
- dashboard: disabled
- default-storageclass: enabled
- efk: disabled
- freshpod: disabled
- heapster: enabled
- ingress: enabled
- kube-dns: enabled
- registry: disabled
- registry-creds: disabled
- storage-provisioner: enabled
~/Kubernetes/2.3 Using Ingress Resources/Code minikube addons enable ingress
ingress was successfully enabled
~/Kubernetes/2.3 Using Ingress Resources/Code minikube ip
192.168.99.100
~/Kubernetes/2.3 Using Ingress Resources/Code sudo vim /etc/hosts
Password:

```
#
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1       localhost
255.255.255.255 broadcasthost
::1             localhost

192.168.99.100  your-app.example
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"/etc/hosts" 11L, 247C
```


your-
Google durchsuchen
Google Vorschläge
your-
Lesezeichen und Verlauf
your-app.example - Google-Suche
Apple
iCloud
Yahoo
Bing
Google
Wikipedia
Facebook
Twitter
LinkedIn
Wetter Online
Yelp
TripAdvisor

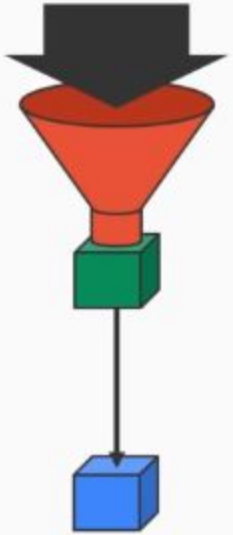

your-app.example
404 Not Found
nginx/1.13.12

Ingress Controller Features

TLS Offloading

Ingress Controller Features

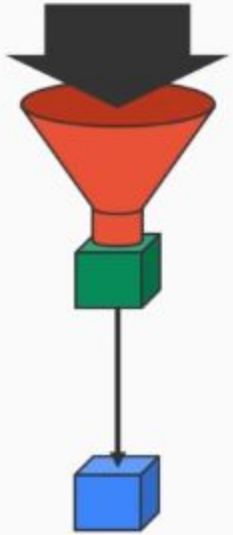
TLS Offloading



Rate Limiting

Ingress Controller Features

TLS Offloading

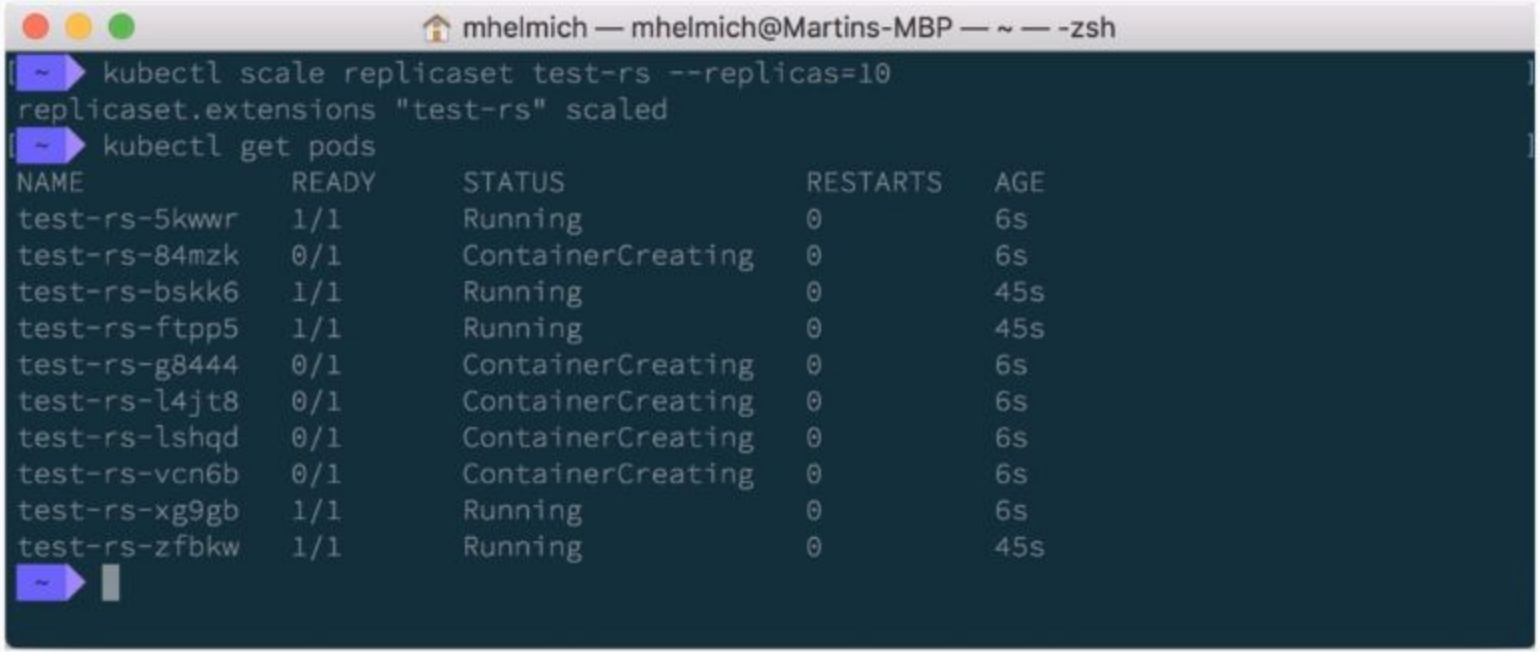


Rate Limiting

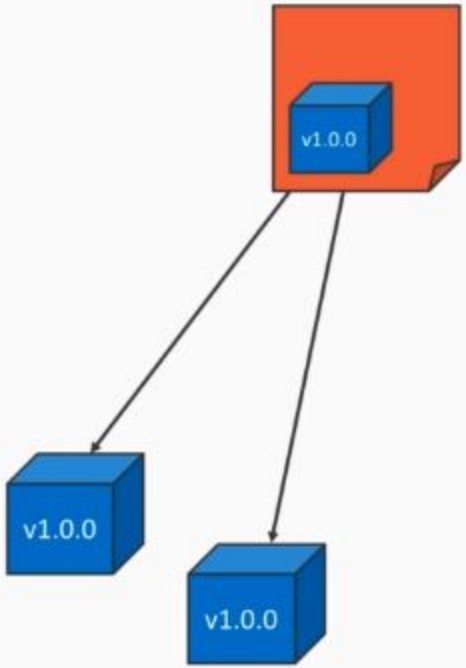
Authentication

Managing the Application Lifecycle with Deployments

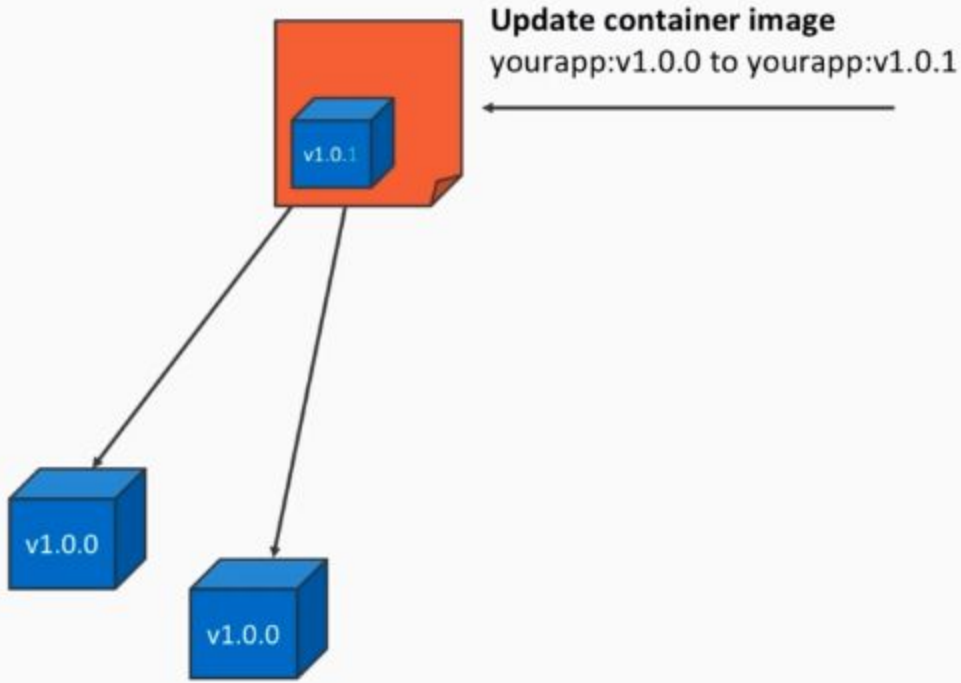
Previously on



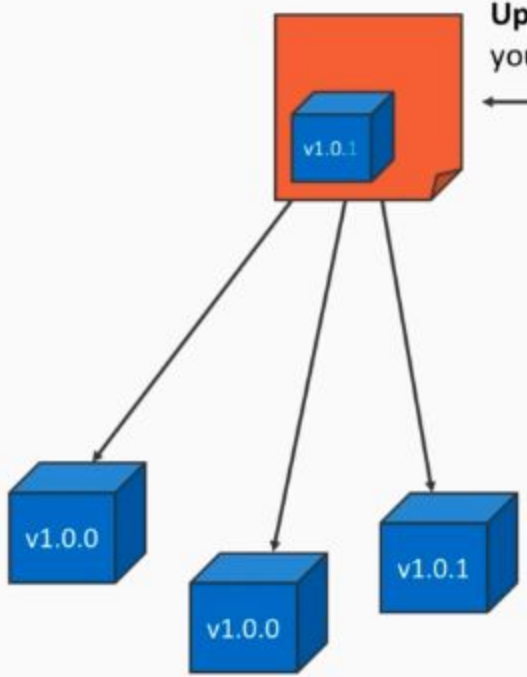
ReplicaSet Limitations



ReplicaSet Limitations



ReplicaSet Limitations

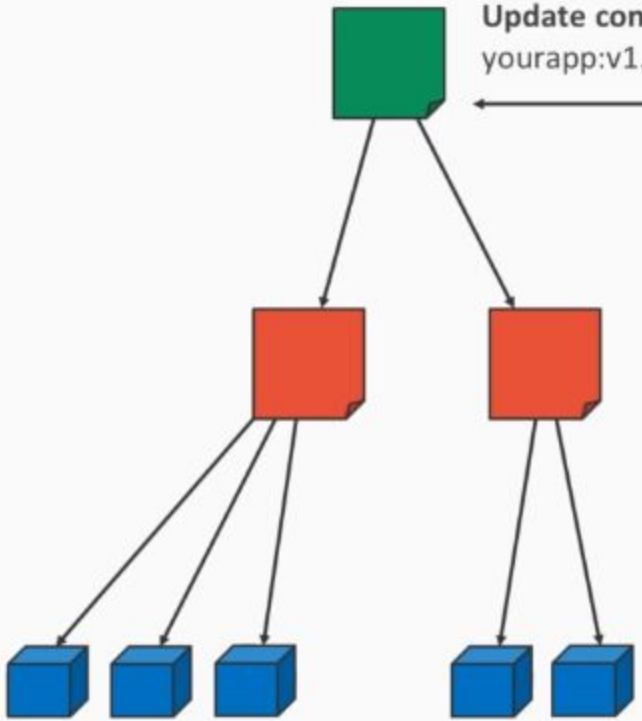


Update container image
yourapp:v1.0.0 to yourapp:v1.0.1

Consequence

- New containers are created with new image
- Existing ones are not updated

Solution: Deployments



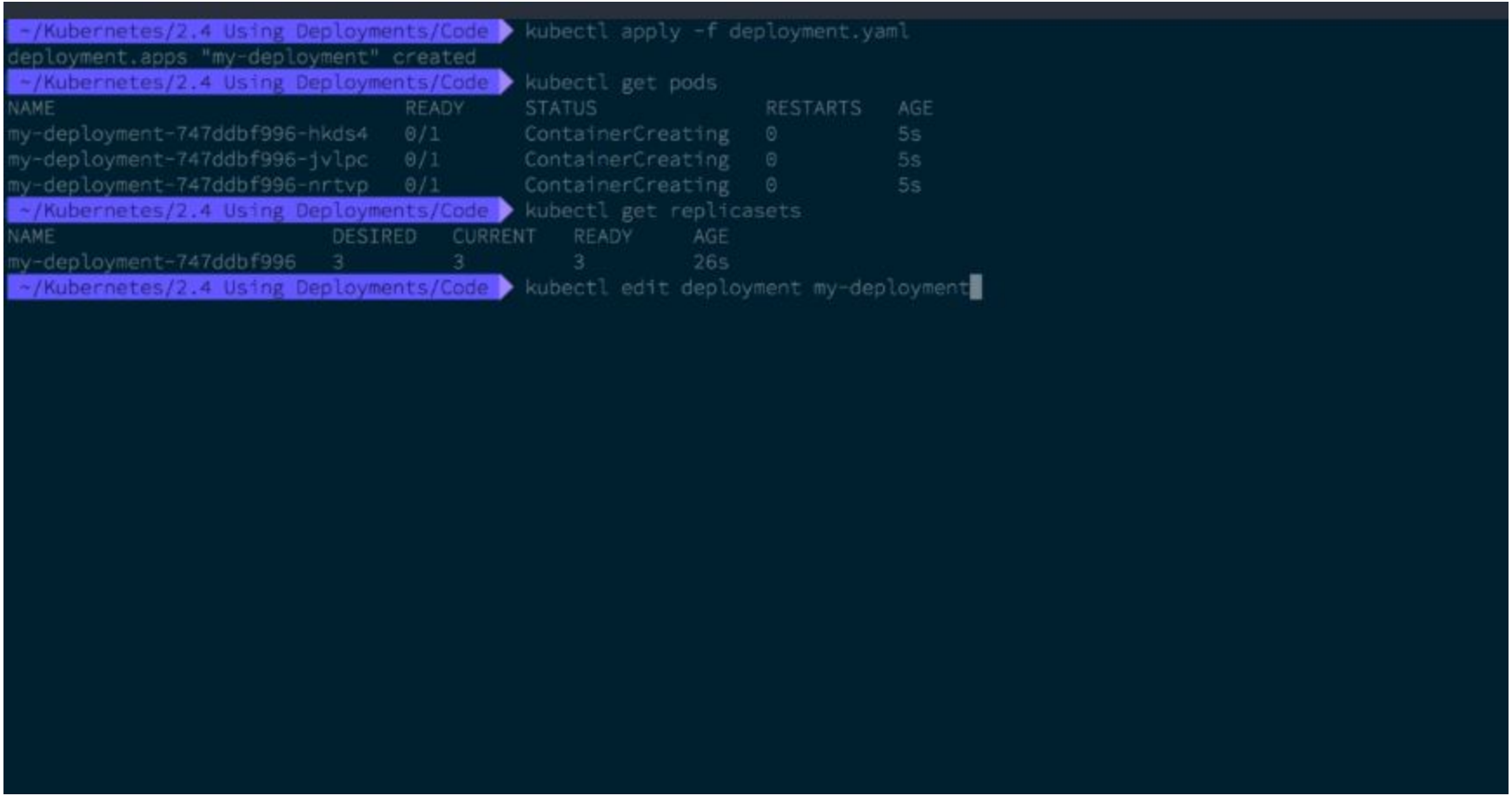
Rolling Updates

- Deployments manage multiple versions of a ReplicaSet
- On update, a new ReplicaSet is created and gradually scaled up
- The old ReplicaSet is gradually scaled down

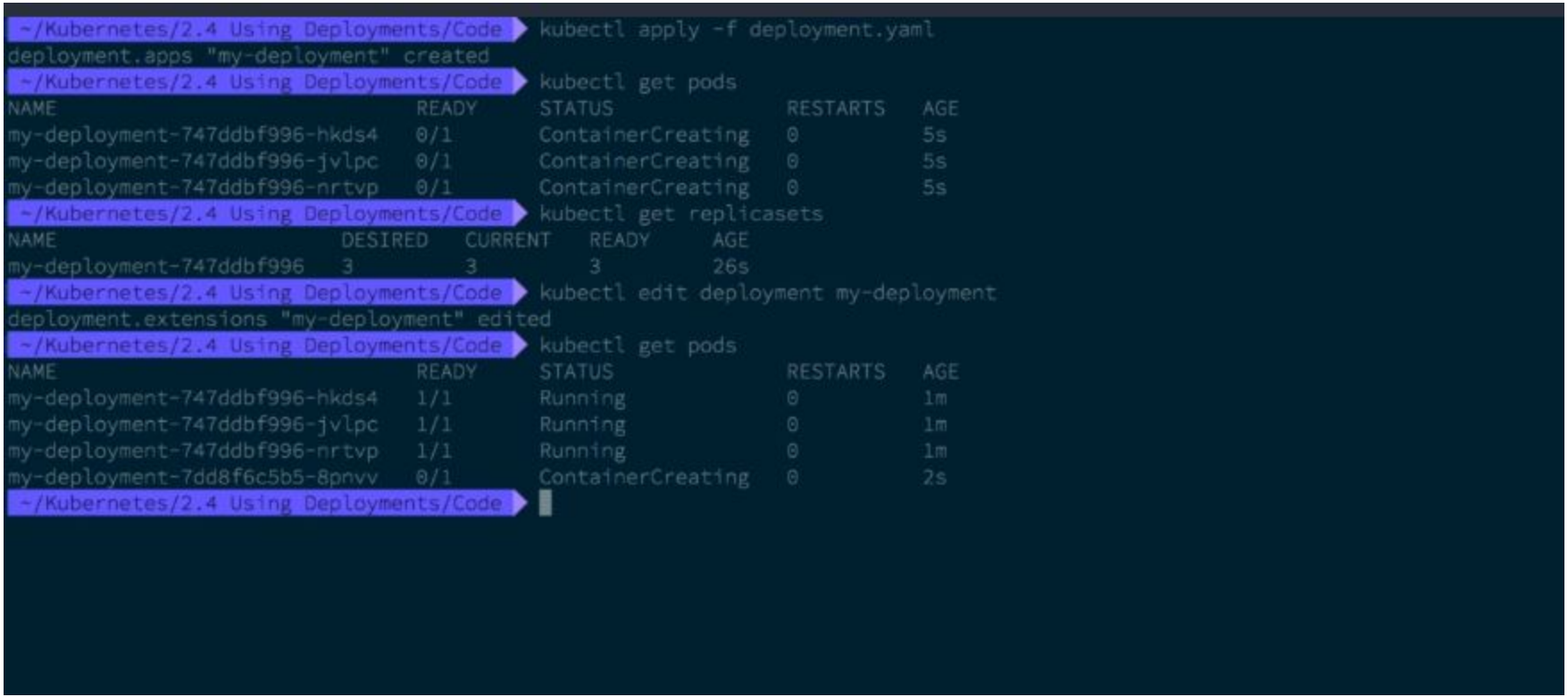
Deployment Example

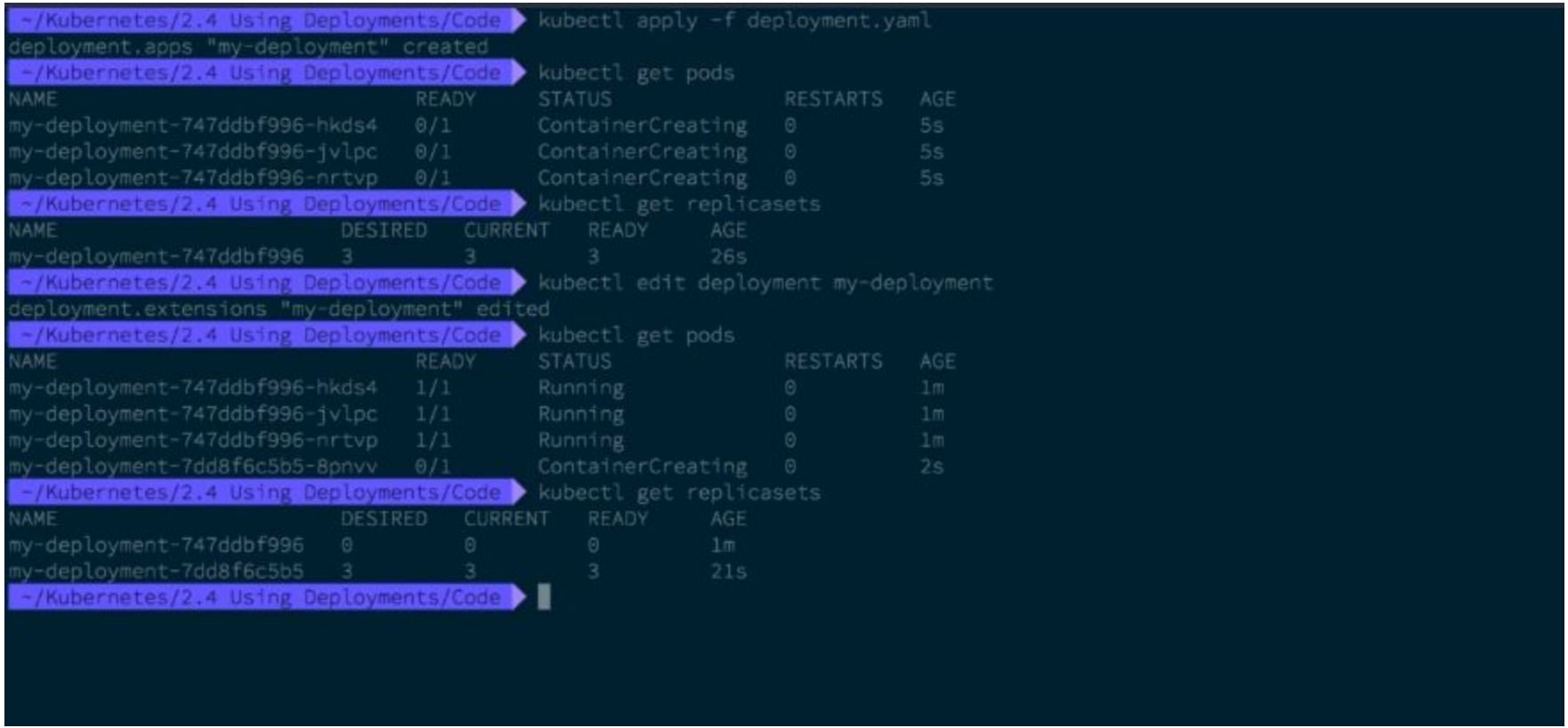
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-service
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-service
  template:
    metadata:
      labels:
        app: my-service
    spec:
      containers:
      - name: service
        image: nginx:1.13.11
```

Hey! This looks just like a ReplicaSet!


~/Kubernetes/2.4 Using Deployments/Code kubectl apply -f deployment.yaml
deployment.apps "my-deployment" created
~/Kubernetes/2.4 Using Deployments/Code kubectl get pods
NAME READY STATUS RESTARTS AGE
my-deployment-747ddbf996-hkds4 0/1 ContainerCreating 0 5s
my-deployment-747ddbf996-jvlpc 0/1 ContainerCreating 0 5s
my-deployment-747ddbf996-nrtvp 0/1 ContainerCreating 0 5s
~/Kubernetes/2.4 Using Deployments/Code kubectl get replicasets
NAME DESIRED CURRENT READY AGE
my-deployment-747ddbf996 3 3 3 26s
~/Kubernetes/2.4 Using Deployments/Code kubectl edit deployment my-deployment

```
  creationTimestamp: 2018-05-31T11:22:14Z
  generation: 1
  labels:
    app: my-service
  name: my-deployment
  namespace: default
  resourceVersion: "9864"
  selfLink: /apis/extensions/v1beta1/namespaces/default/deployments/my-deployment
  uid: de6cb09b-64c4-11e8-8174-080027c553e0
spec:
  progressDeadlineSeconds: 600
  replicas: 3
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: my-service
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: my-service
    spec:
      containers:
      - image: nginx:1.13.12
        imagePullPolicy: IfNotPresent
        name: nginx
        resources: {}
:
```


~/Kubernetes/2.4 Using Deployments/Code kubectl apply -f deployment.yaml
deployment.apps "my-deployment" created
~/Kubernetes/2.4 Using Deployments/Code kubectl get pods
NAME READY STATUS RESTARTS AGE
my-deployment-747ddbf996-hkds4 0/1 ContainerCreating 0 5s
my-deployment-747ddbf996-jvlpc 0/1 ContainerCreating 0 5s
my-deployment-747ddbf996-nrtvp 0/1 ContainerCreating 0 5s
~/Kubernetes/2.4 Using Deployments/Code kubectl get replicasets
NAME DESIRED CURRENT READY AGE
my-deployment-747ddbf996 3 3 3 26s
~/Kubernetes/2.4 Using Deployments/Code kubectl edit deployment my-deployment
deployment.extensions "my-deployment" edited
~/Kubernetes/2.4 Using Deployments/Code kubectl get pods
NAME READY STATUS RESTARTS AGE
my-deployment-747ddbf996-hkds4 1/1 Running 0 1m
my-deployment-747ddbf996-jvlpc 1/1 Running 0 1m
my-deployment-747ddbf996-nrtvp 1/1 Running 0 1m
my-deployment-7dd8f6c5b5-8pnvv 0/1 ContainerCreating 0 2s
~/Kubernetes/2.4 Using Deployments/Code


~/Kubernetes/2.4 Using Deployments/Code kubectl apply -f deployment.yaml
deployment.apps "my-deployment" created
~/Kubernetes/2.4 Using Deployments/Code kubectl get pods
NAME READY STATUS RESTARTS AGE
my-deployment-747ddbf996-hkds4 0/1 ContainerCreating 0 5s
my-deployment-747ddbf996-jvlpc 0/1 ContainerCreating 0 5s
my-deployment-747ddbf996-nrtvp 0/1 ContainerCreating 0 5s
~/Kubernetes/2.4 Using Deployments/Code kubectl get replicasets
NAME DESIRED CURRENT READY AGE
my-deployment-747ddbf996 3 3 3 26s
~/Kubernetes/2.4 Using Deployments/Code kubectl edit deployment my-deployment
deployment.extensions "my-deployment" edited
~/Kubernetes/2.4 Using Deployments/Code kubectl get pods
NAME READY STATUS RESTARTS AGE
my-deployment-747ddbf996-hkds4 1/1 Running 0 1m
my-deployment-747ddbf996-jvlpc 1/1 Running 0 1m
my-deployment-747ddbf996-nrtvp 1/1 Running 0 1m
my-deployment-7dd8f6c5b5-8pnvv 0/1 ContainerCreating 0 2s
~/Kubernetes/2.4 Using Deployments/Code kubectl get replicasets
NAME DESIRED CURRENT READY AGE
my-deployment-747ddbf996 0 0 0 1m
my-deployment-7dd8f6c5b5 3 3 3 21s
~/Kubernetes/2.4 Using Deployments/Code

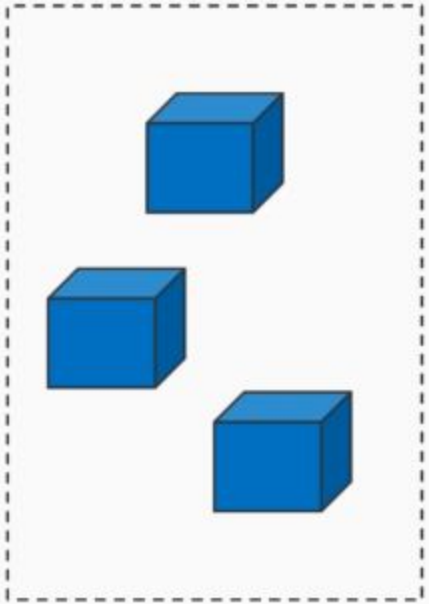
```
deployment.apps "my-deployment" created
~/Kubernetes/2.4 Using Deployments/Code > kubectl get pods
NAME                             READY     STATUS              RESTARTS   AGE
my-deployment-747ddbf996-hkds4   0/1       ContainerCreating   0          5s
my-deployment-747ddbf996-jvlpc   0/1       ContainerCreating   0          5s
my-deployment-747ddbf996-nrtvp   0/1       ContainerCreating   0          5s
~/Kubernetes/2.4 Using Deployments/Code > kubectl get replicasets
NAME                       DESIRED   CURRENT   READY     AGE
my-deployment-747ddbf996   3         3         3         26s
~/Kubernetes/2.4 Using Deployments/Code > kubectl edit deployment my-deployment
deployment.extensions "my-deployment" edited
~/Kubernetes/2.4 Using Deployments/Code > kubectl get pods
NAME                             READY     STATUS              RESTARTS   AGE
my-deployment-747ddbf996-hkds4   1/1       Running             0          1m
my-deployment-747ddbf996-jvlpc   1/1       Running             0          1m
my-deployment-747ddbf996-nrtvp   1/1       Running             0          1m
my-deployment-7dd8f6c5b5-8pnvv   0/1       ContainerCreating   0          2s
~/Kubernetes/2.4 Using Deployments/Code > kubectl get replicasets
NAME                       DESIRED   CURRENT   READY     AGE
my-deployment-747ddbf996   0         0         0         1m
my-deployment-7dd8f6c5b5   3         3         3         21s
~/Kubernetes/2.4 Using Deployments/Code > kubectl rollout undo deployment/my-deployment
deployment.apps "my-deployment"
~/Kubernetes/2.4 Using Deployments/Code > kubectl get replicasets
NAME                       DESIRED   CURRENT   READY     AGE
my-deployment-747ddbf996   3         3         3         2m
my-deployment-7dd8f6c5b5   0         0         0         1m
~/Kubernetes/2.4 Using Deployments/Code > kubectl get pods
NAME                             READY     STATUS    RESTARTS   AGE
my-deployment-747ddbf996-6hdpn   1/1       Running   0          15s
my-deployment-747ddbf996-8jjlh   1/1       Running   0          16s
my-deployment-747ddbf996-m8prz   1/1       Running   0          18s
~/Kubernetes/2.4 Using Deployments/Code >
```

Stateful Services and Storage Drivers

Up Until Now

Stateless Pods

- Pods that do not have any kind of state that needs to be stored
- Data is stored/retrieved from external storages

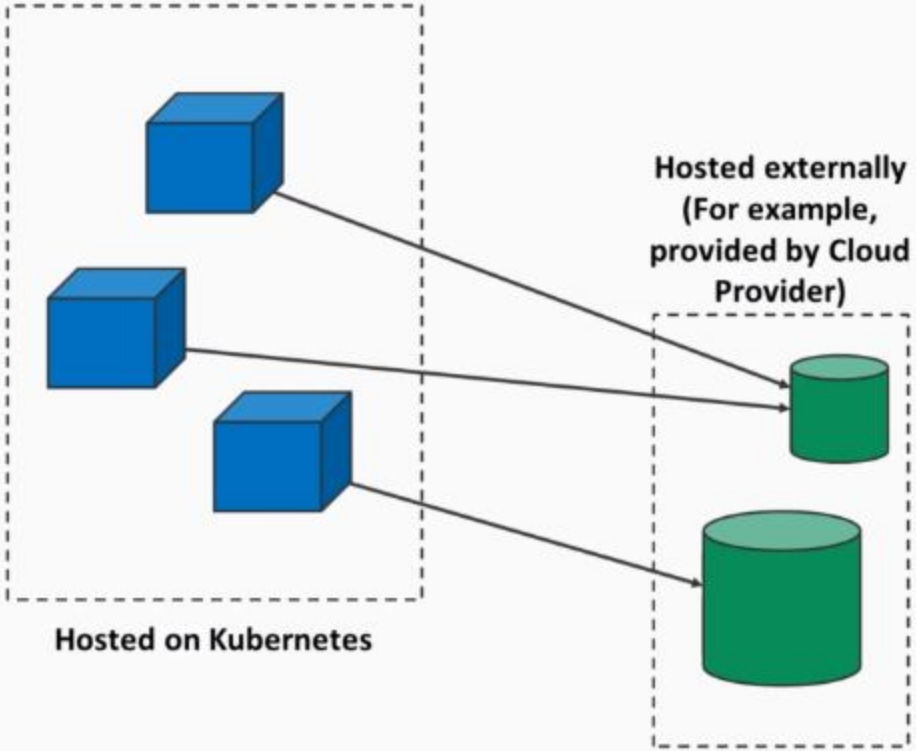


Hosted on Kubernetes

Up Until Now

Stateless Pods

- Pods that do not have any kind of state that needs to be stored
- Data is stored/retrieved from external storages



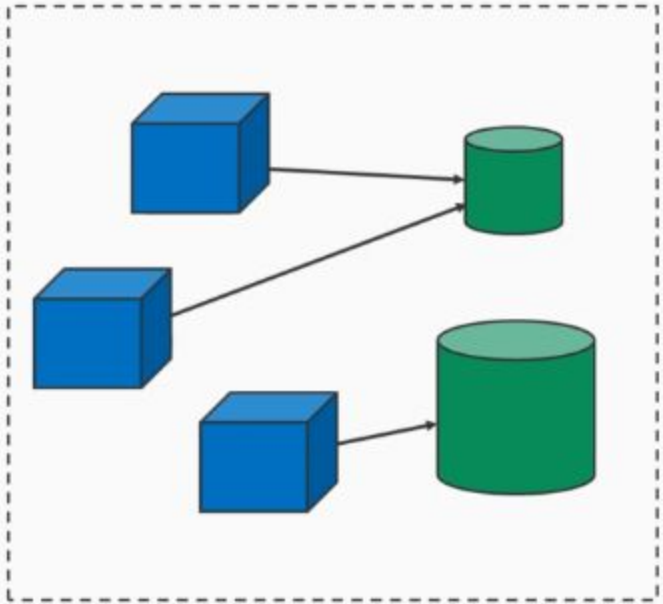
External Storage Services

- Databases
- Object Stores (like AWS S3 and similar)

Upcoming

Stateless Pods

- Pods that do not have any kind of state that needs to be stored
- Data is stored/retrieved from external storages

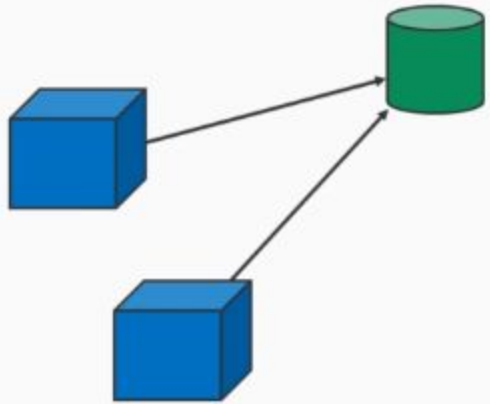


Hosted on Kubernetes

Stateful Pods

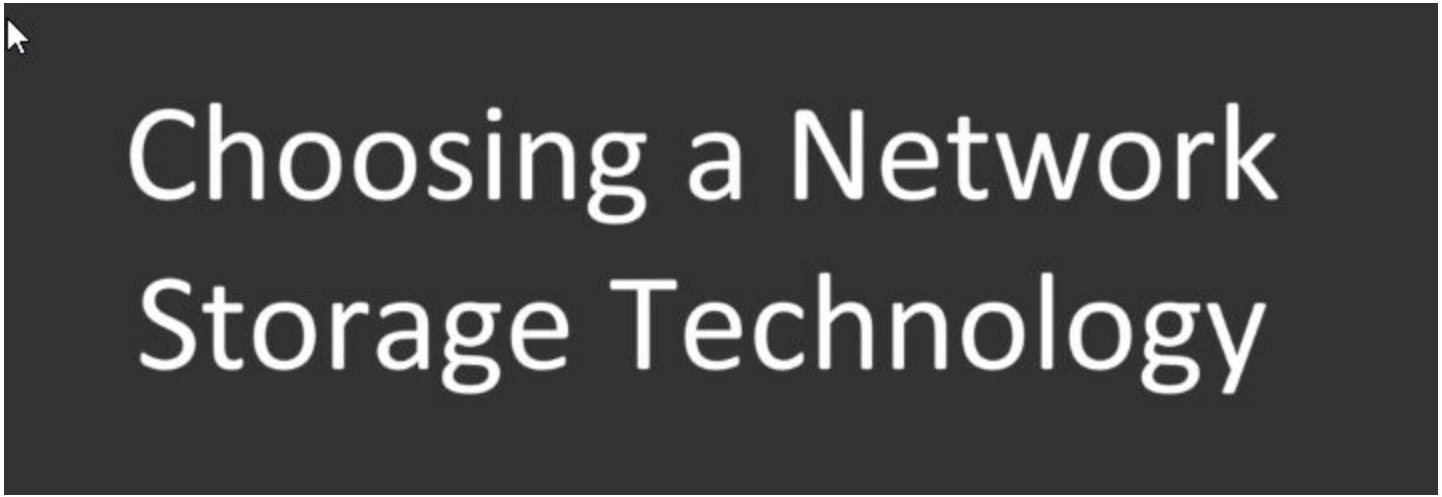
- Pods that have state/need to store data
- Require persistent file system across Pod termination/recreation

Persistent Volumes (Short: PV)



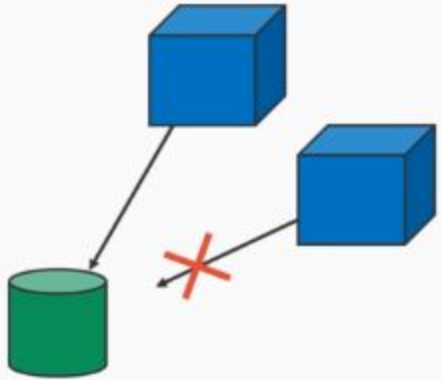
Persistent Volume

- Can be provided as a file system to Kubernetes Pods
- Lifecycle is independent from Pods
- Can be accessed by multiple Pods at once (dependent on storage engine)


Choosing a Network
Storage Technology

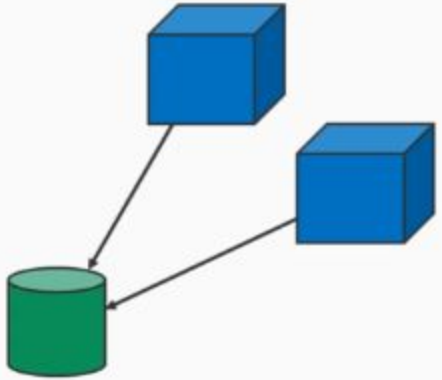
Network Storage Technologies

- Network Block Devices
  - Examples (cloud providers):
    - AWS Elastic Block Devices
    - GCE Persistent Disk
    - AzureDisk
    - OpenStack Cinder
  - Examples (self-hosting):
    - Rados Block Devices (Ceph)
    - iSCSI
  - Key features
    - Fast
    - Fixed-size
    - Can (usually) be used by one Pod at a time



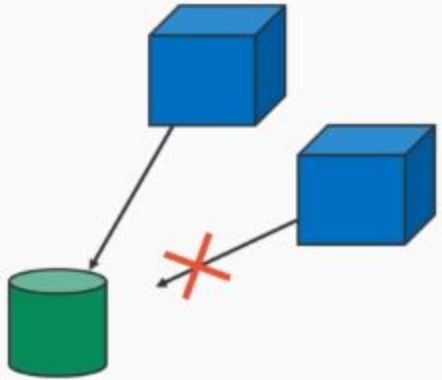
Network Storage Technologies (Continued)

- Network File Systems
  - Examples (cloud providers):
    - AWS Elastic File System (actually NFS)
    - Azure Files (actually SMB)
  - Examples (self-hosting):
    - CephFS
    - NFS
    - GlusterFS
  - Key features
    - Slower than Block Devices
    - Can be used by many Pods at a time



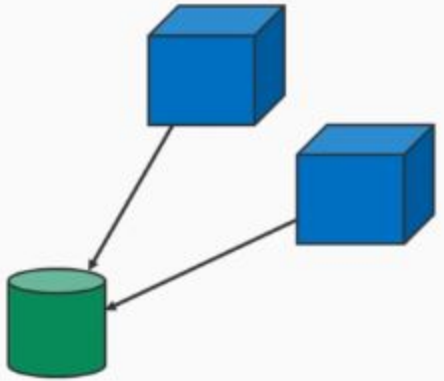
Network Storage Technologies

- Network Block Devices
  - Examples (cloud providers):
    - AWS Elastic Block Devices
    - GCE Persistent Disk
    - AzureDisk
    - OpenStack Cinder
  - Examples (self-hosting):
    - Rados Block Devices (Ceph)
    - iSCSI
  - Key features
    - Fast
    - Fixed-size
    - Can (usually) be used by one Pod at a time



Network Storage Technologies (Continued)

- Network File Systems
  - Examples (cloud providers):
    - AWS Elastic File System (actually NFS)
    - Azure Files (actually SMB)
  - Examples (self-hosting):
    - CephFS
    - NFS
    - GlusterFS
  - Key features
    - Slower than Block Devices
    - Can be used by many Pods at a time



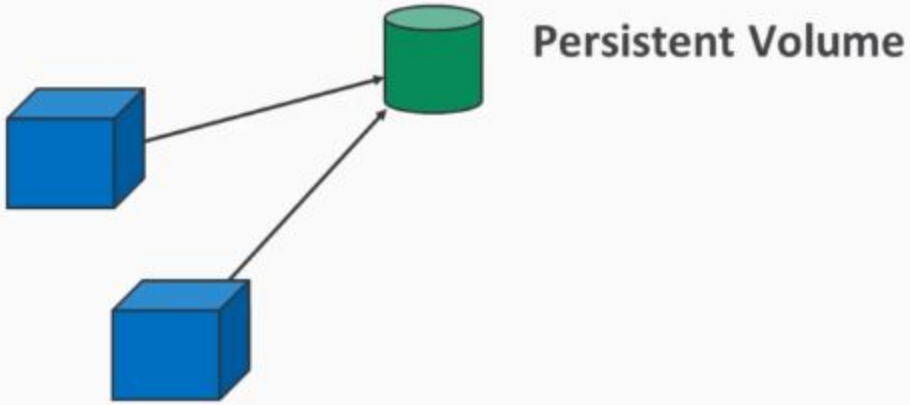
Final Notes

Storage Is ~~Easy~~ Hard

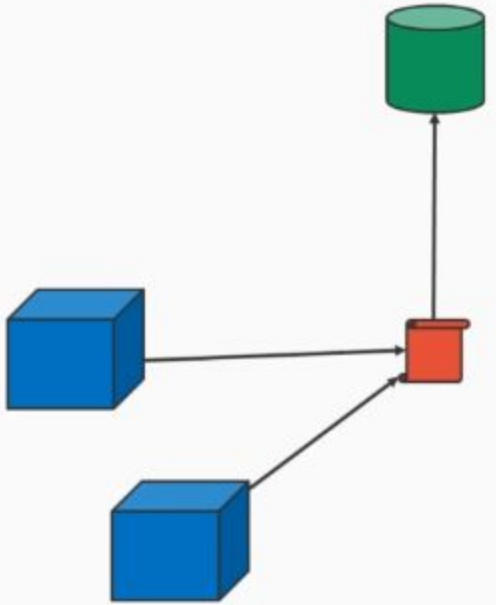
- Setting up Ceph or other self-hosted solutions is out-of-scope for this course
- Have a look at Learning Ceph by Anthony D'Atri, Vaibhav Bhembre and Karan Singh (https://www.packtpub.com/virtualization-and-cloud/learning-ceph-second-edition)
- Mixed operation between storage engines is possible

Working with Persistent Volumes

What Happened Previously?



Volumes and Claims



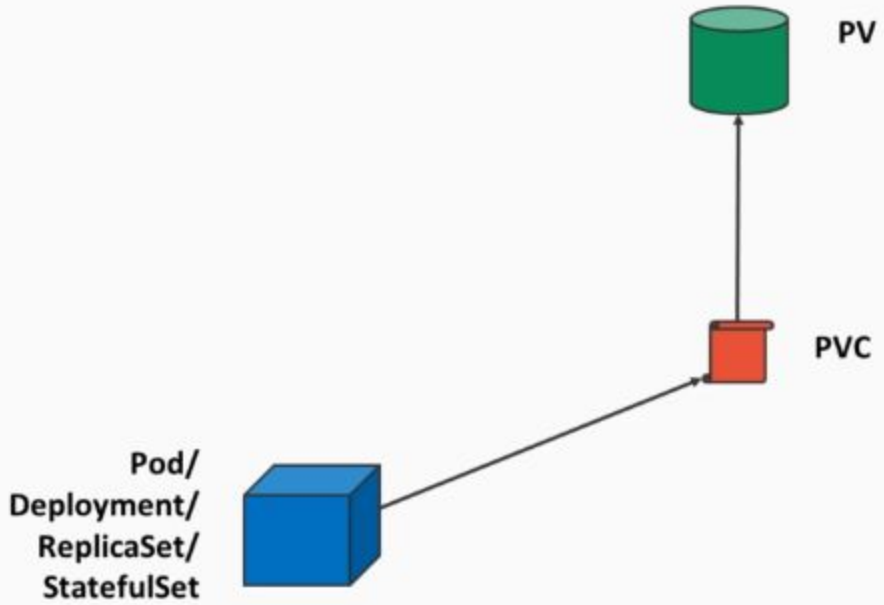
PersistentVolume

- Refers to a specific network block device or file system

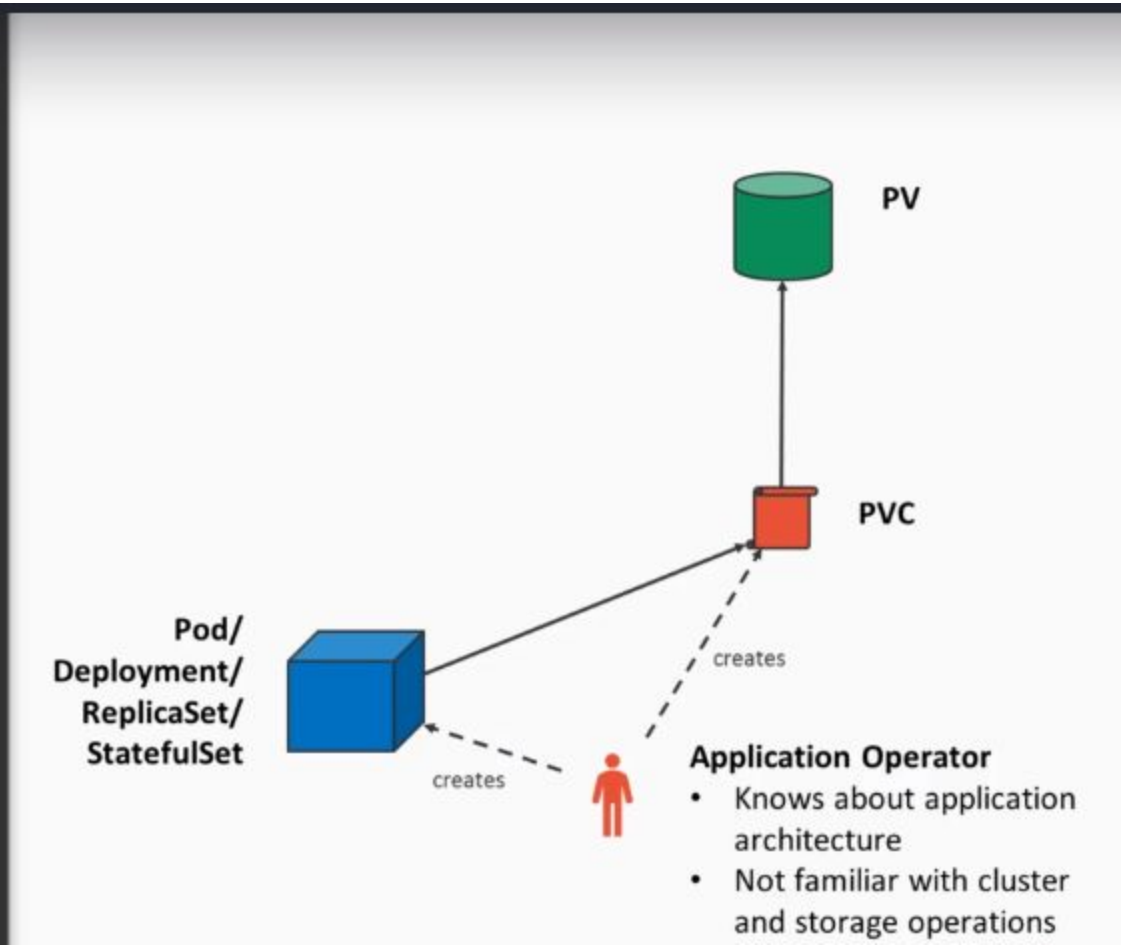
PersistentVolumeClaim

- Claims a PV for use in one or more Pods

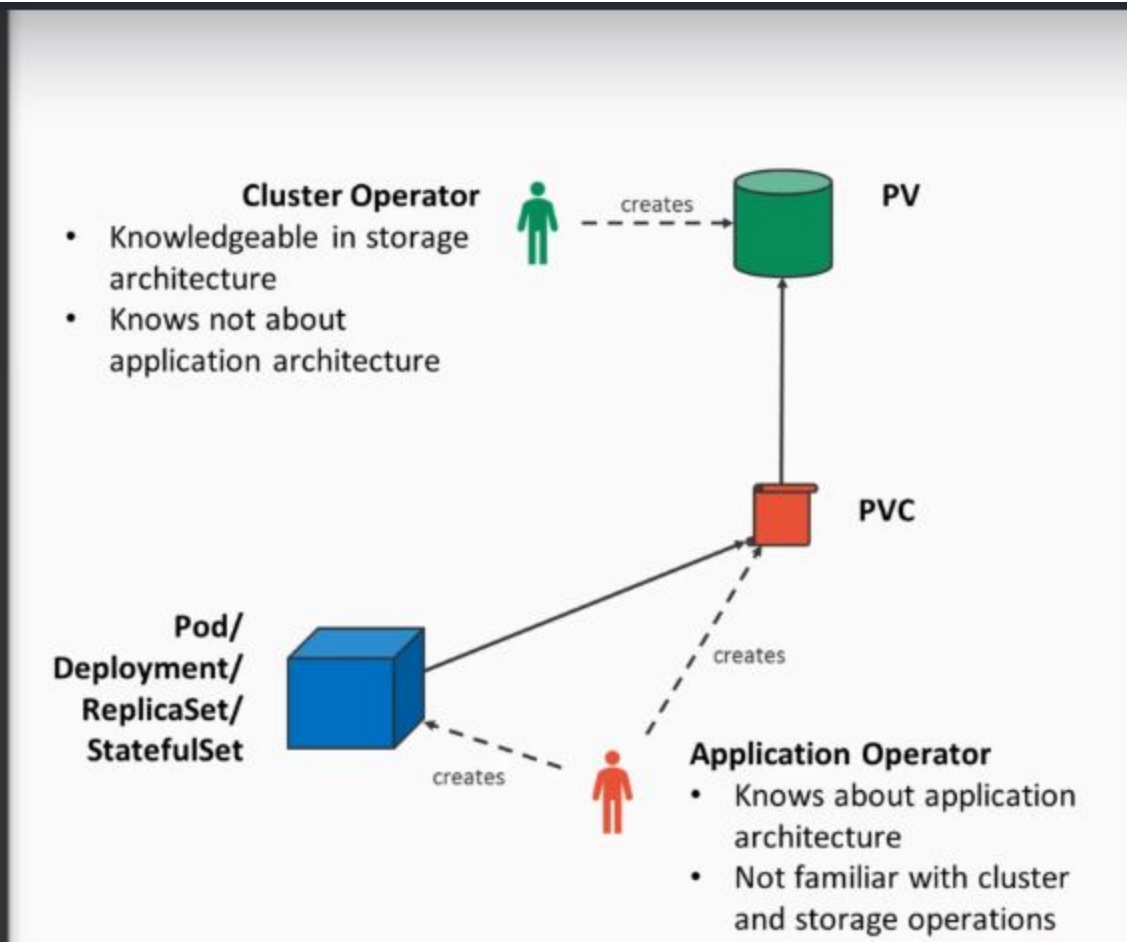
Why? Separation of Concerns!



Why? Separation of Concerns!



Why? Separation of Concerns!



PersistentVolume Example

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: my-volume
spec:
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  awsElasticBlockStore:
    volumeID: vol-08374e22abe6cb9b6
    fsType: ext4
```

Volume capacity

PersistentVolume Example

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: my-volume
spec:
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  awsElasticBlockStore:
    volumeID: vol-08374e22abe6cb9b6
    fsType: ext4
```

Access modes:

- ReadWriteOnce – One Pod at once
- ReadWriteMany – Many Pods at once

PersistentVolume Example

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: my-volume
spec:
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  awsElasticBlockStore:
    volumeID: vol-08374e22abe6cb9b6
    fsType: ext4
```

Actual volume location; depends on volume driver (AWS in this case)

PersistentVolume Example (For Minikube Users)

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: my-volume
spec:
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: /data/my-volume
```

Caution!
Never use a hostPath volume in a clustered environment!

PersistentVolumeClaim Example

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-volume-claim
spec:
  resources:
    requests:
      storage: 10Gi
  accessModes:
    - ReadWriteOnce
```

Required capacity

PersistentVolumeClaim Example

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-volume-claim
spec:
  resources:
    requests:
      storage: 10Gi
  accessModes:
    - ReadWriteOnce
```

Required access mode

Using Volumes in Pods

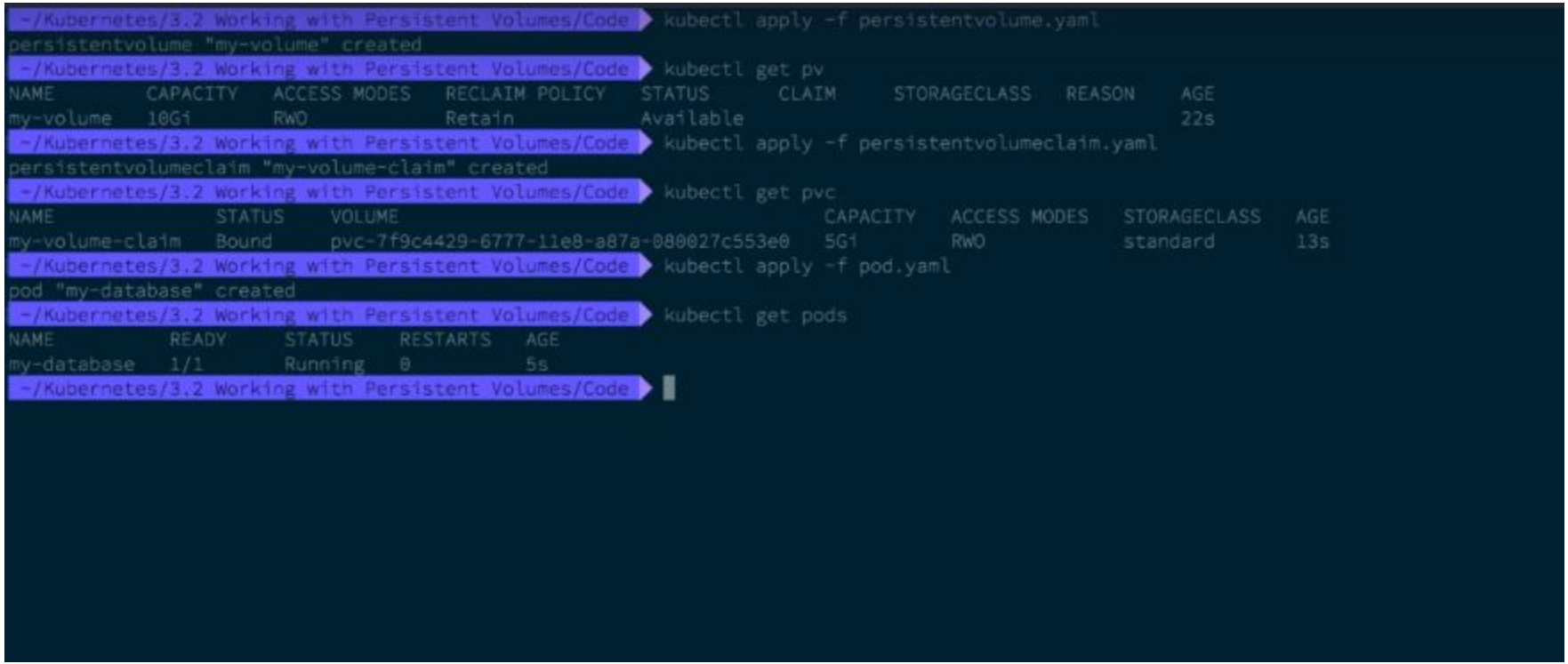
```
apiVersion: v1
kind: Pod
metadata:
  name: my-database
spec:
  containers:
    - name: database
      image: mysql:5.7
      env:
        # …
+     volumeMounts:
+       - mountPath: /var/lib/mysql
+         name: data
+ volumes:
+   - name: data
+     persistentVolumeClaim:
+       claimName: my-volume-claim
```

Volume claims tha
a Pod should use

Using Volumes in Pods

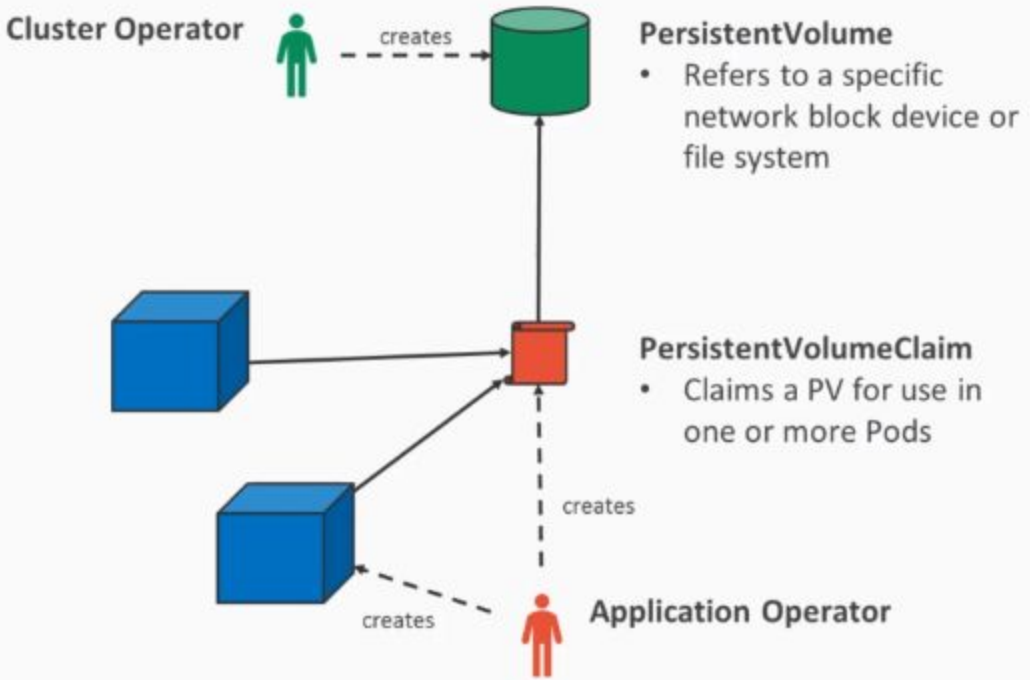
```
apiVersion: v1
kind: Pod
metadata:
  name: my-database
spec:
  containers:
    - name: database
      image: mysql:5.7
      env:
        # …
+     volumeMounts:
+       - mountPath: /var/lib/mysql
+         name: data
+   volumes:
+     - name: data
+       persistentVolumeClaim:
+         claimName: my-volume-claim
```

Where to mount the volume within the Pod

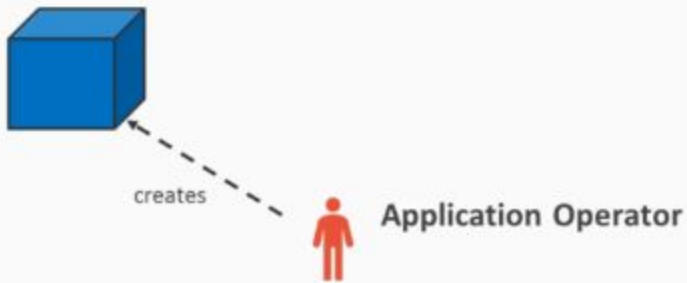

~/Kubernetes/3.2 Working with Persistent Volumes/Code kubectl apply -f persistentvolume.yaml
persistentvolume "my-volume" created
~/Kubernetes/3.2 Working with Persistent Volumes/Code kubectl get pv
NAME CAPACITY ACCESS MODES RECLAIM POLICY STATUS CLAIM STORAGECLASS REASON AGE
my-volume 10Gi RWO Retain Available 22s
~/Kubernetes/3.2 Working with Persistent Volumes/Code kubectl apply -f persistentvolumeclaim.yaml
persistentvolumeclaim "my-volume-claim" created
~/Kubernetes/3.2 Working with Persistent Volumes/Code kubectl get pvc
NAME STATUS VOLUME CAPACITY ACCESS MODES STORAGECLASS AGE
my-volume-claim Bound pvc-7f9c4429-6777-11e8-a87a-080027c553e0 5Gi RWO standard 13s
~/Kubernetes/3.2 Working with Persistent Volumes/Code kubectl apply -f pod.yaml
pod "my-database" created
~/Kubernetes/3.2 Working with Persistent Volumes/Code kubectl get pods
NAME READY STATUS RESTARTS AGE
my-database 1/1 Running 0 5s
~/Kubernetes/3.2 Working with Persistent Volumes/Code

Automatic Volume Provisioning

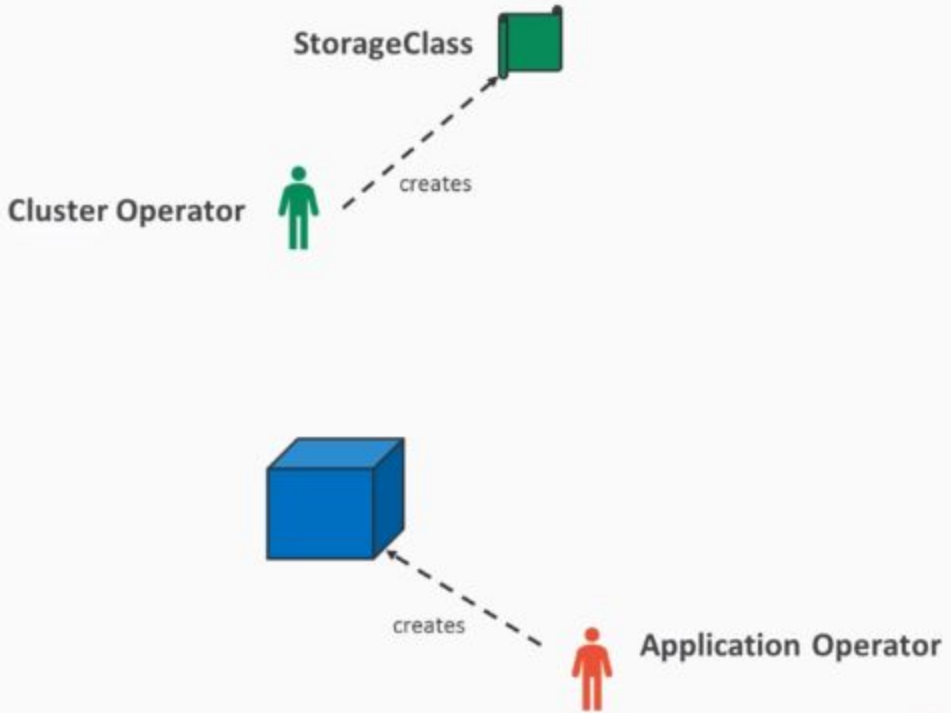
What Happened Previously?



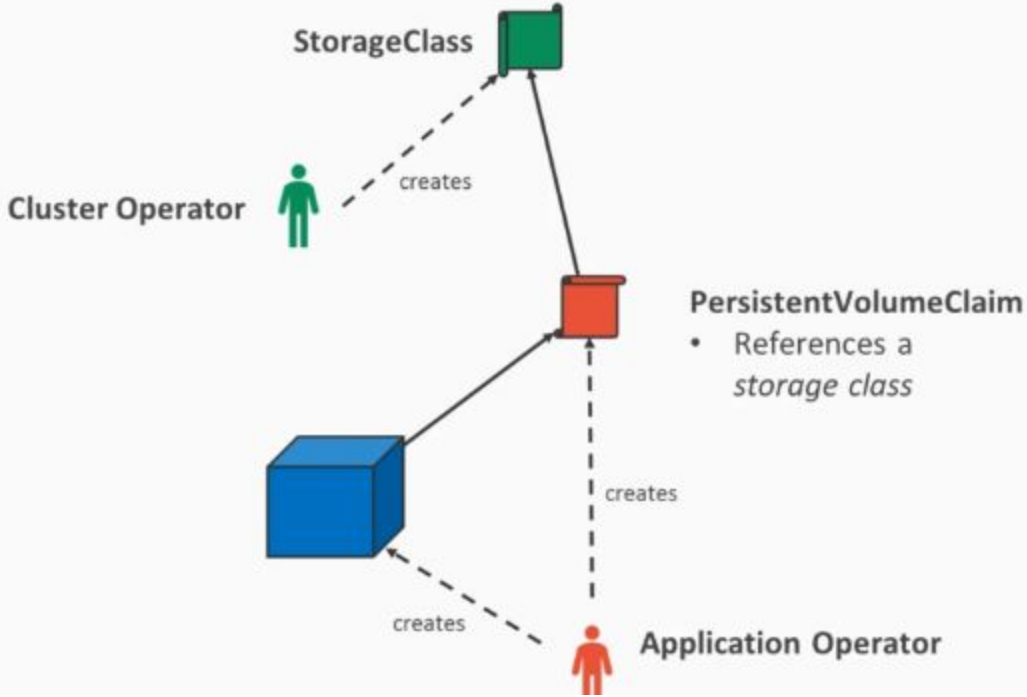
Storage Classes



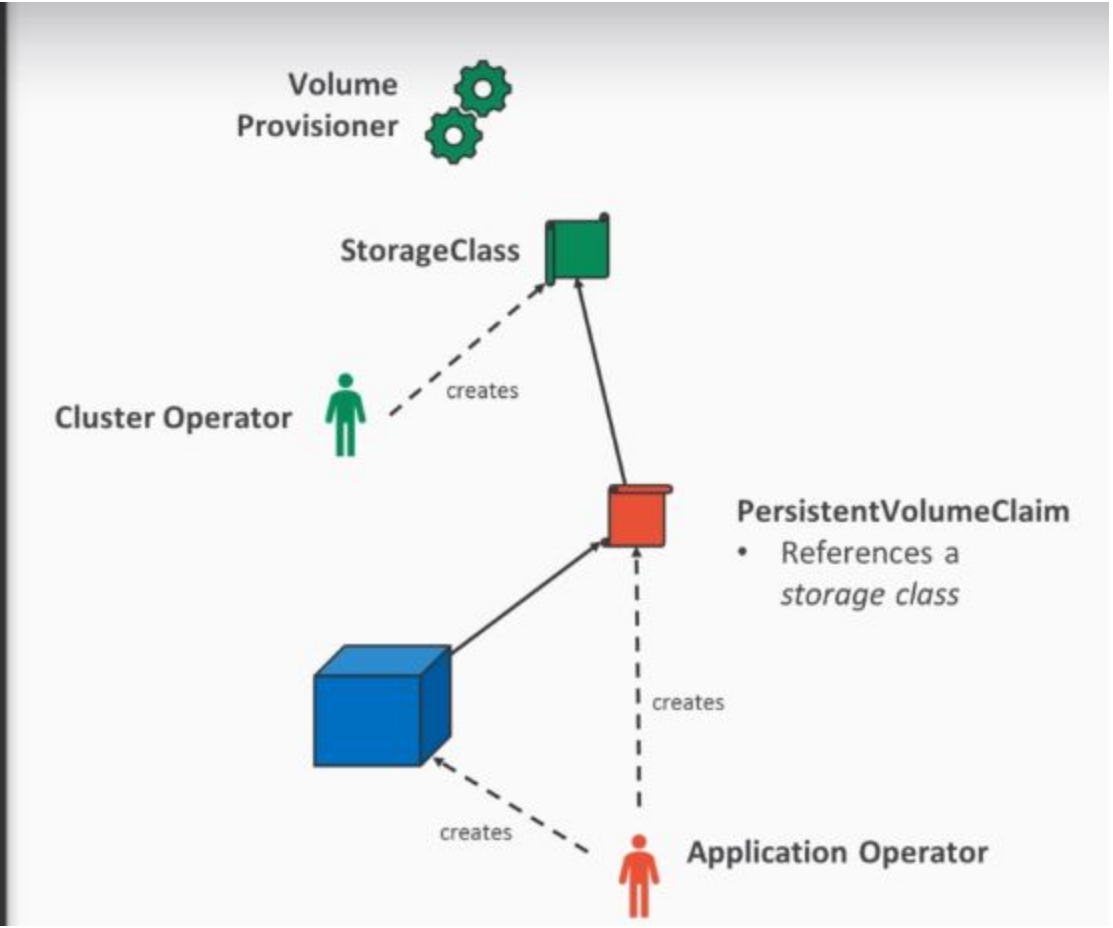
Storage Classes



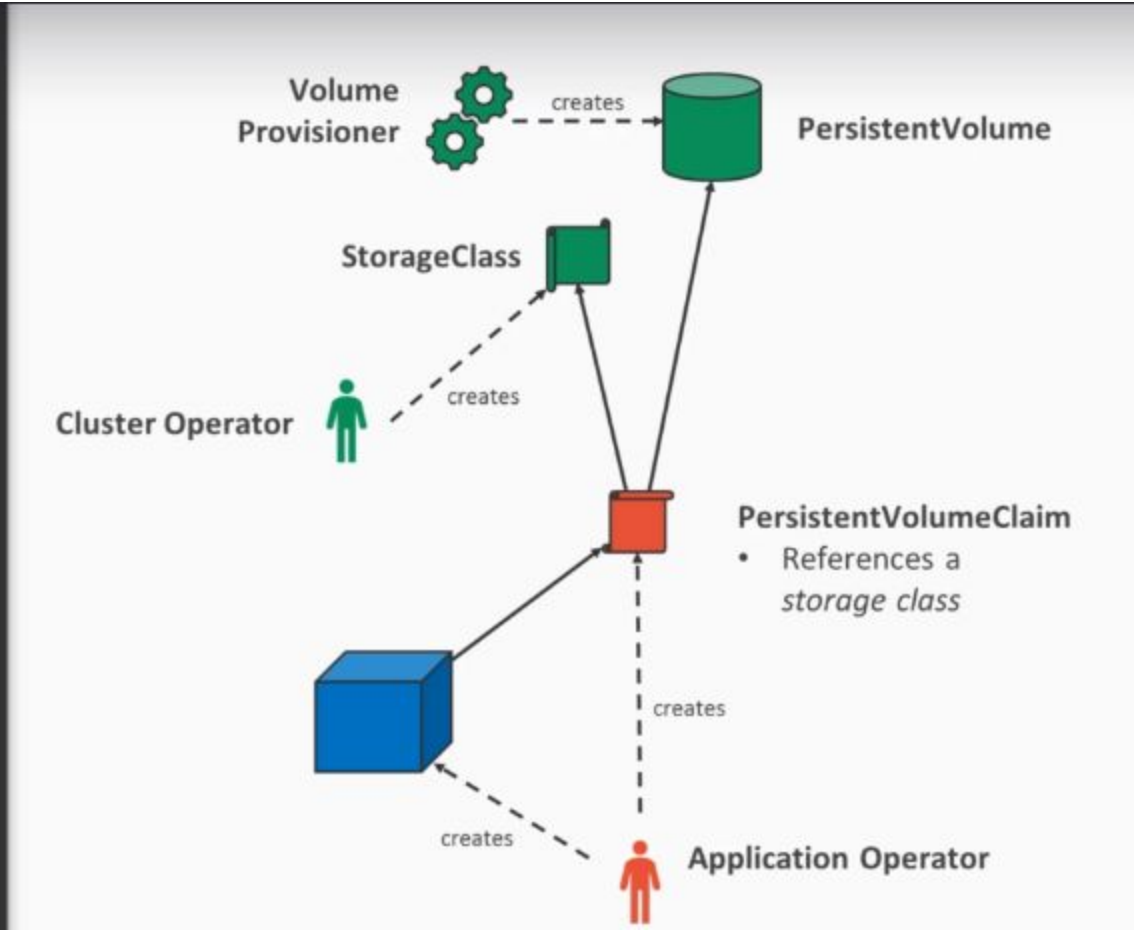
Storage Classes



Storage Classes



Storage Classes



StorageClass Example

```
apiVersion: storage.k8s.io/v1
 kind: StorageClass
 metadata:
   name: ssd-fast
 provisioner: kubernetes.io/aws-ebs
 parameters:
   type: io1
   zones: eu-central-1
   iopsPerGB: "20"
```

Provisioner name Examples:

- kubernetes.io/aws-ebs
- kubernetes.io/gce-pd
- kubernetes.io/azure-disk
- kubernetes.io/azure-file
- kubernetes.io/glusterfs

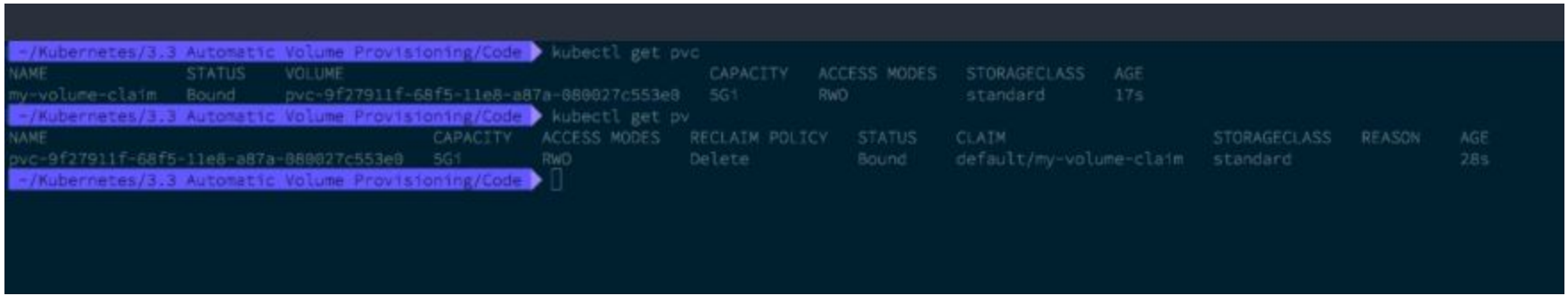
StorageClass Example

```
apiVersion: storage.k8s.io/v1
 kind: StorageClass
 metadata:
   name: ssd-fast
 provisioner: kubernetes.io/aws-ebs
 parameters:
   type: io1
   zones: eu-central-1
   iopsPerGB: "20"
```

Provisioner-dependent parameters

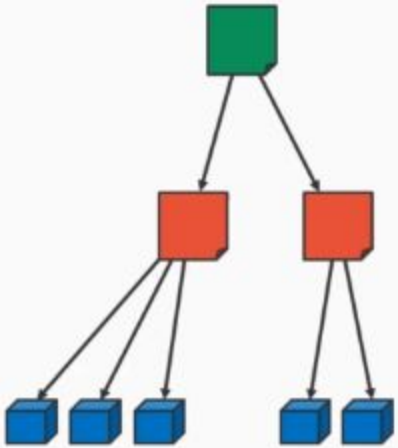
PersistentVolumeClaim Example with StorageClass

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-volume-claim
spec:
+ storageClass: ssd-fast
  resources:
    requests:
      storage: 10Gi
  accessModes:
    - ReadWriteOnce
```


~/Kubernetes/3.3 Automatic Volume Provisioning/Code kubectl get pvc
NAME STATUS VOLUME CAPACITY ACCESS MODES STORAGECLASS AGE
my-volume-claim Bound pvc-9f27911f-68f5-11e8-a87a-080027c553e0 5Gi RWO standard 17s
~/Kubernetes/3.3 Automatic Volume Provisioning/Code kubectl get pv
NAME CAPACITY ACCESS MODES RECLAIM POLICY STATUS CLAIM STORAGECLASS REASON AGE
pvc-9f27911f-68f5-11e8-a87a-080027c553e0 5Gi RWO Delete Bound default/my-volume-claim standard 28s
~/Kubernetes/3.3 Automatic Volume Provisioning/Code

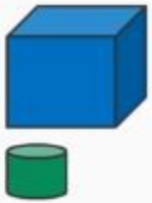
Using Stateful Sets

What Happened Previously?



Deployments and
RollingUpdates

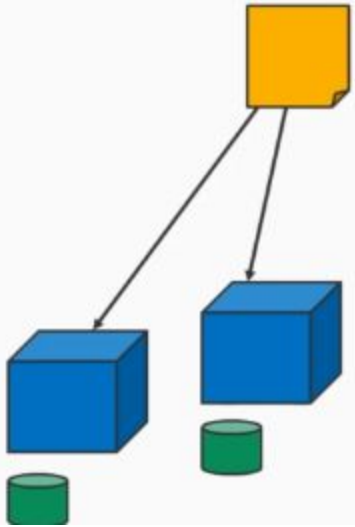
StatefulSets



Single Pod (stateful)

- Disposable
- Not resilient by itself
- Not scalable by itself

StatefulSets



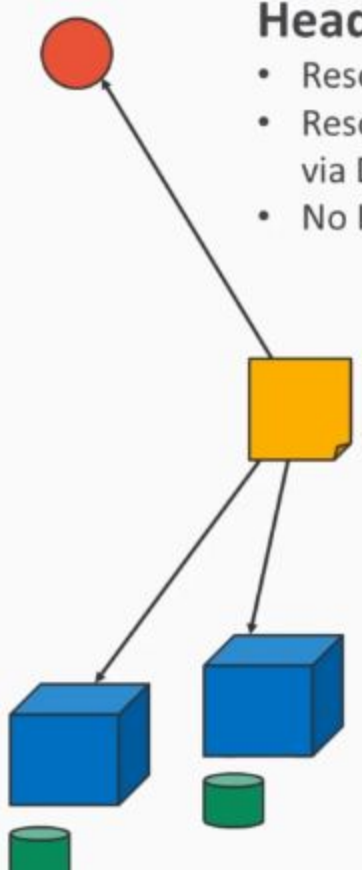
StatefulSet

- Manages Pods and volumes
- Self-healing and scalable
- Ideal for clustered apps

Single Pod (stateful)

- Disposable
- Not resilient by itself
- Not scalable by itself

StatefulSets



Headless Service

- Resolve StatefulSet via DNS
- Resolve individual members via DNS
- No IP loadbalancing, DNS only

StatefulSet

- Manages Pods and volumes
- Self-healing and scalable
- Ideal for clustered apps

Single Pod (stateful)

- Disposable
- Not resilient by itself
- Not scalable by itself

StatefulSet Example

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: my-database
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-database
  service: my-db-service
  template:
    metadata:
      labels:
        app: my-database
    spec:
      containers:
      - name: db
        image: mongo:3.7.9
        ports:
          # Omitted for brevity; see code samples
        volumeMounts:
        - name: data
          mountPath: /data
  volumeClaimTemplates:
  - metadata:
      name: data
    spec:
      accessModes: [ReadWriteOnce]
      storageClassName: ssd-fast
      resources:
        requests:
          storage: 10Gi
```

Replica count

StatefulSet Example

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: my-database
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-database
  service: my-db-service
  template:
    metadata:
      labels:
        app: my-database
    spec:
      containers:
      - name: db
        image: mongo:3.7.9
        ports:
          # Omitted for brevity; see code samples
        volumeMounts:
        - name: data
          mountPath: /data
  volumeClaimTemplates:
  - metadata:
      name: data
    spec:
      accessModes: [ReadWriteOnce]
      storageClassName: ssd-fast
      resources:
        requests:
          storage: 10Gi
```

Assigned service name

StatefulSet Example

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: my-database
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-database
  service: my-db-service
  template:
    metadata:
      labels:
        app: my-database
    spec:
      containers:
      - name: db
        image: mongo:3.7.9
        ports:
          # Omitted for brevity; see code samples
        volumeMounts:
        - name: data
          mountPath: /data
volumeClaimTemplates:
- metadata:
    name: data
  spec:
    accessModes: [ReadWriteOnce]
    storageClassName: ssd-fast
    resources:
      requests:
        storage: 10Gi
```

Template for volume claims
(ideally with automatic provisioning)

StatefulSet Example

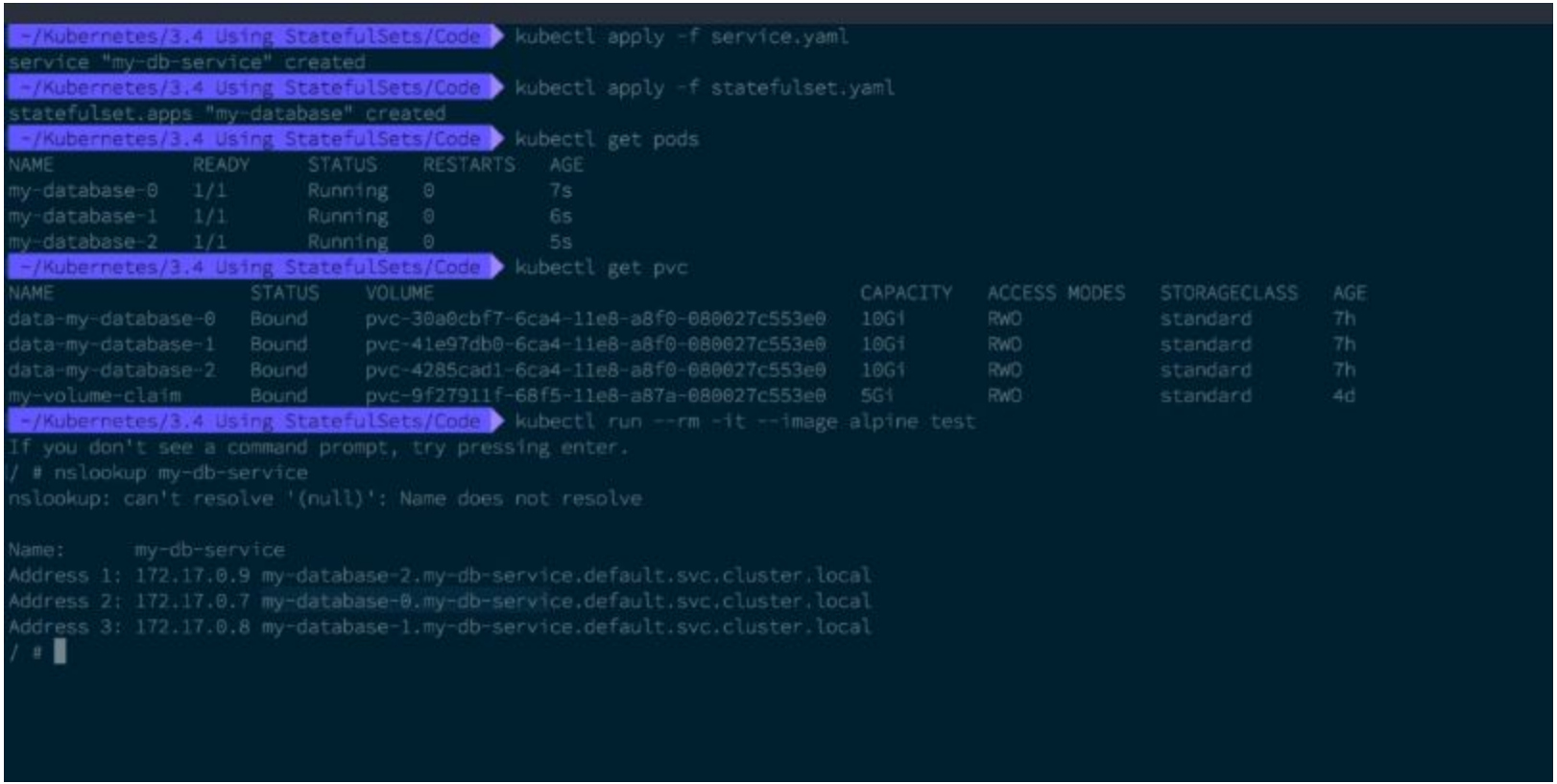
```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: my-database
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-database
  service: my-db-service
  template:
    metadata:
      labels:
        app: my-database
    spec:
      containers:
      - name: db
        image: mongo:3.7.9
        ports:
          # Omitted for brevity; see code samples
        volumeMounts:
        - name: data
          mountPath: /data
  volumeClaimTemplates:
  - metadata:
      name: data
    spec:
      accessModes: [ReadWriteOnce]
      storageClassName: ssd-fast
      resources:
        requests:
          storage: 10Gi
```

Using the volume claim in the Pod

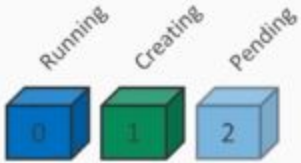
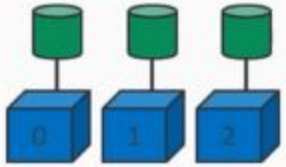
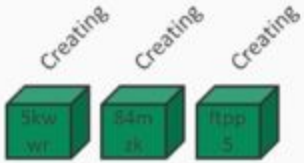
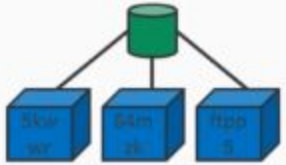
Headless Services

```
apiVersion: v1
kind: Service
metadata:
  name: my-db-service
spec:
  ports:
  - port: 27017
    name: mongodb
  clusterIP: None
  selector:
    app: my-database
```

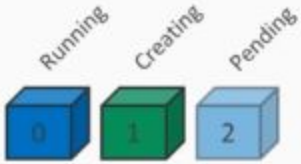
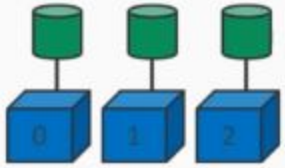
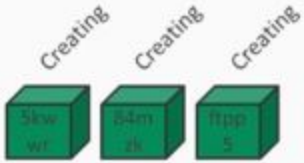
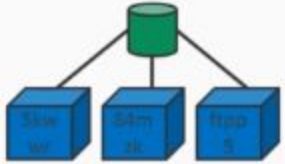
No IP load balancing;
DNS only


~/Kubernetes/3.4 Using StatefulSets/Code kubectl apply -f service.yaml
service "my-db-service" created
~/Kubernetes/3.4 Using StatefulSets/Code kubectl apply -f statefulset.yaml
statefulset.apps "my-database" created
~/Kubernetes/3.4 Using StatefulSets/Code kubectl get pods
NAME READY STATUS RESTARTS AGE
my-database-0 1/1 Running 0 7s
my-database-1 1/1 Running 0 6s
my-database-2 1/1 Running 0 5s
~/Kubernetes/3.4 Using StatefulSets/Code kubectl get pvc
NAME STATUS VOLUME CAPACITY ACCESS MODES STORAGECLASS AGE
data-my-database-0 Bound pvc-30a0cbf7-6ca4-11e8-a8f0-080027c553e0 10Gi RWO standard 7h
data-my-database-1 Bound pvc-41e97db0-6ca4-11e8-a8f0-080027c553e0 10Gi RWO standard 7h
data-my-database-2 Bound pvc-4285cad1-6ca4-11e8-a8f0-080027c553e0 10Gi RWO standard 7h
my-volume-claim Bound pvc-9f27911f-68f5-11e8-a87a-080027c553e0 5Gi RWO standard 4d
~/Kubernetes/3.4 Using StatefulSets/Code kubectl run --rm -it --image alpine test
If you don't see a command prompt, try pressing enter.
/ # nslookup my-db-service
nslookup: can't resolve '(null)': Name does not resolve
Name: my-db-service
Address 1: 172.17.0.9 my-database-2.my-db-service.default.svc.cluster.local
Address 2: 172.17.0.7 my-database-0.my-db-service.default.svc.cluster.local
Address 3: 172.17.0.8 my-database-1.my-db-service.default.svc.cluster.local
/ #

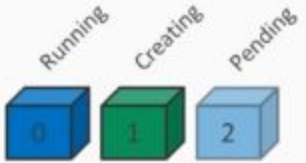
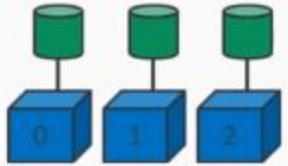
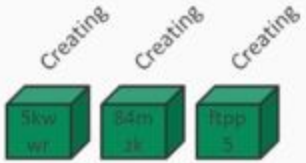
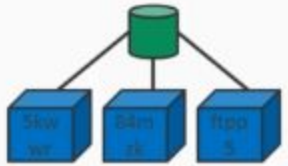
StatefulSets Versus Deployments

StatefulSet	 Stable identity, DNS-addressable	 Started/stopped in order	 Volume provisioning for new members
ReplicaSet/ Deployment	 No stable identity	 Started/stopped all at once	 Manually configured shared volumes
	Identity	Startup and shutdown order	Persistence

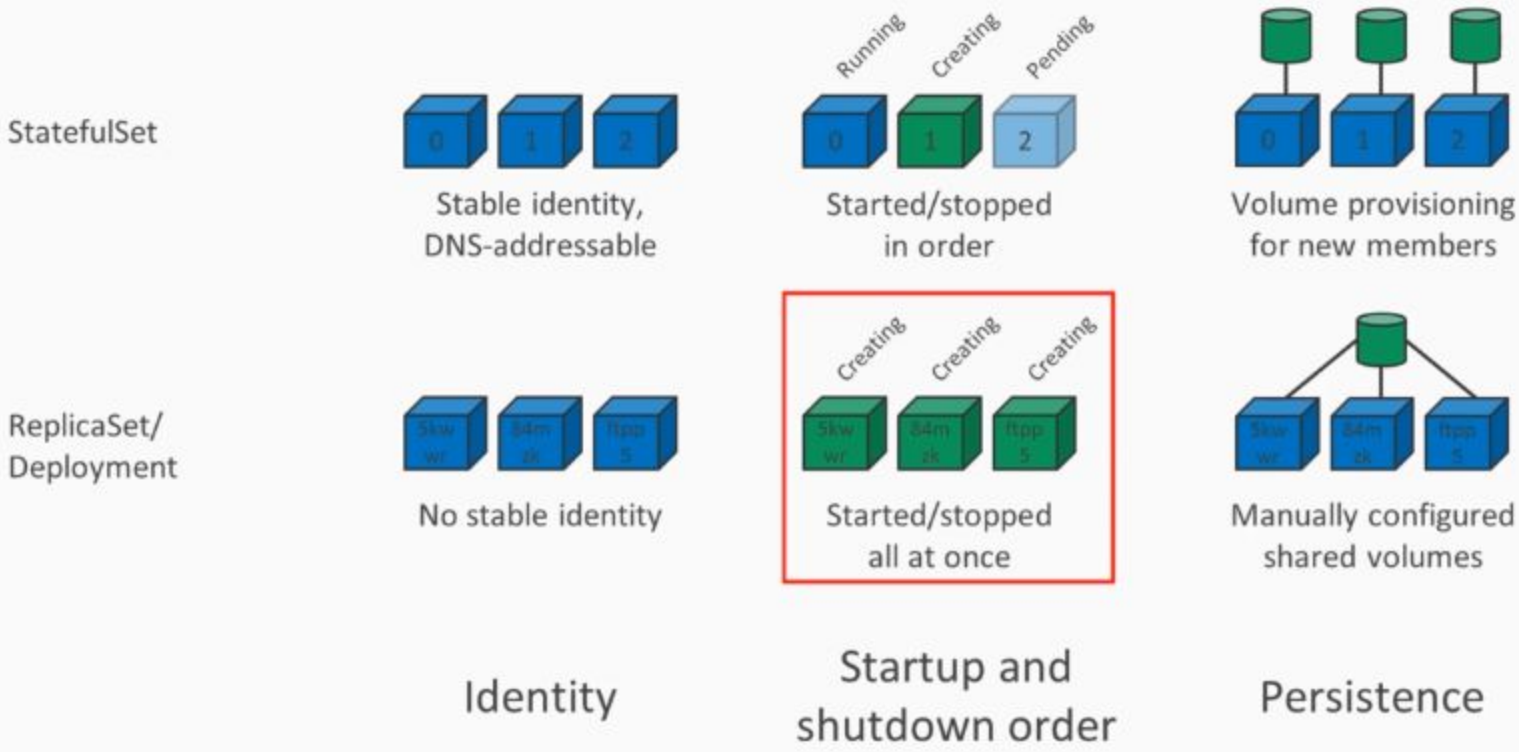
StatefulSets Versus Deployments

	Identity	Startup and shutdown order	Persistence
StatefulSet	 Stable identity, DNS-addressable	 Started/stopped in order	 Volume provisioning for new members
ReplicaSet/ Deployment	 No stable identity	 Started/stopped all at once	 Manually configured shared volumes

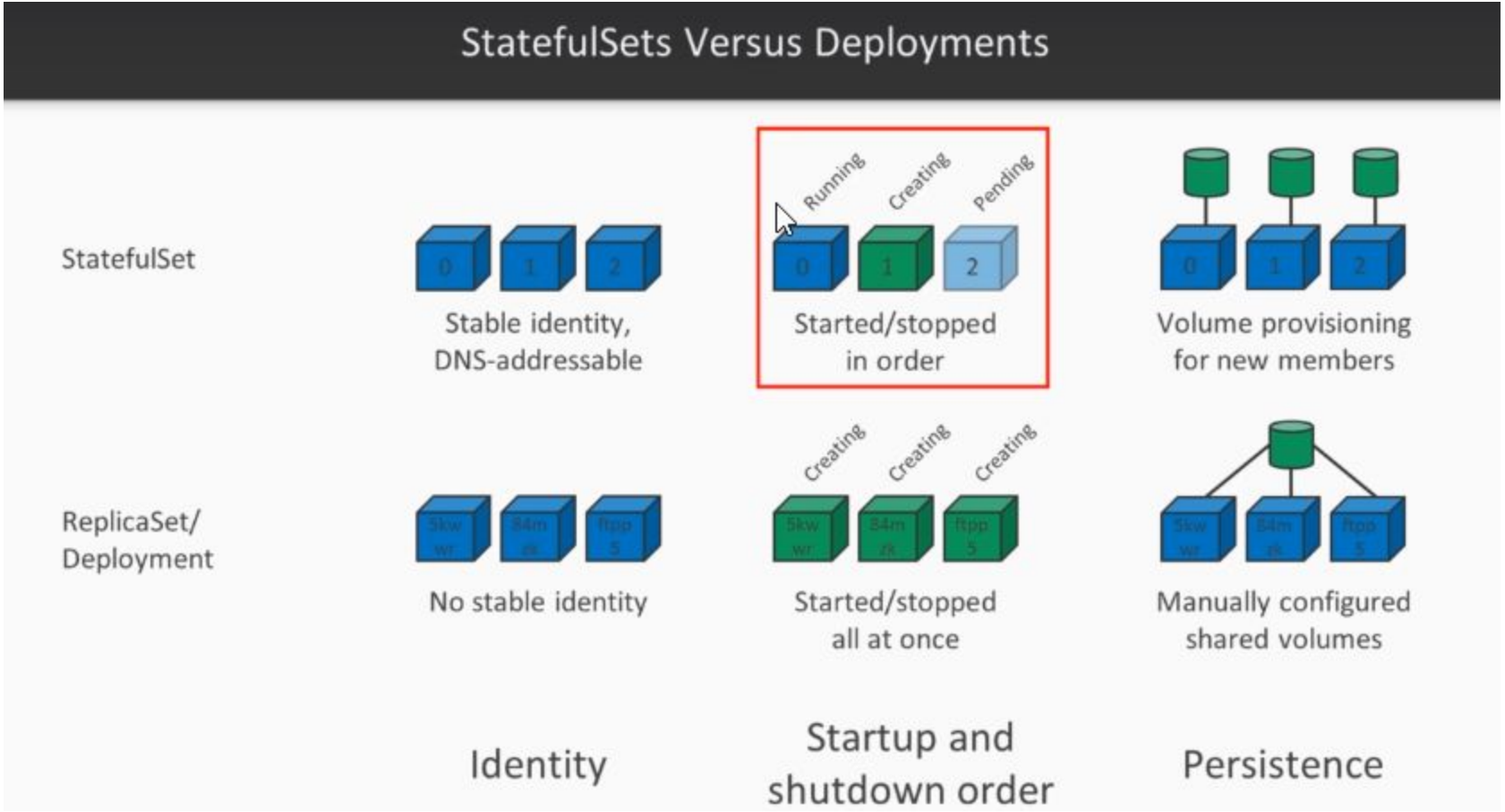
StatefulSets Versus Deployments

	Identity	Startup and shutdown order	Persistence
StatefulSet	 Stable identity, DNS-addressable	 Started/stopped in order	 Volume provisioning for new members
ReplicaSet/ Deployment	 No stable identity	 Started/stopped all at once	 Manually configured shared volumes

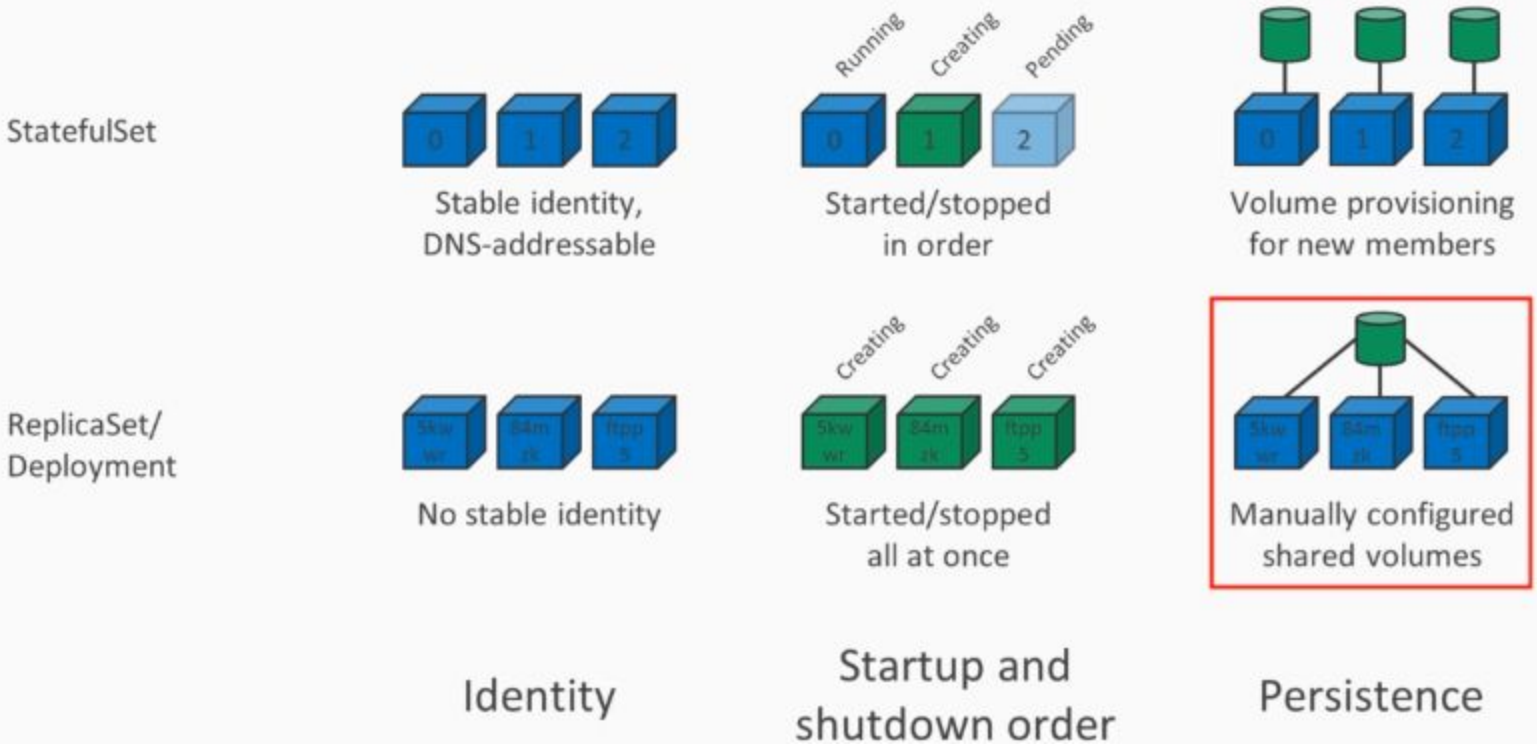
StatefulSets Versus Deployments



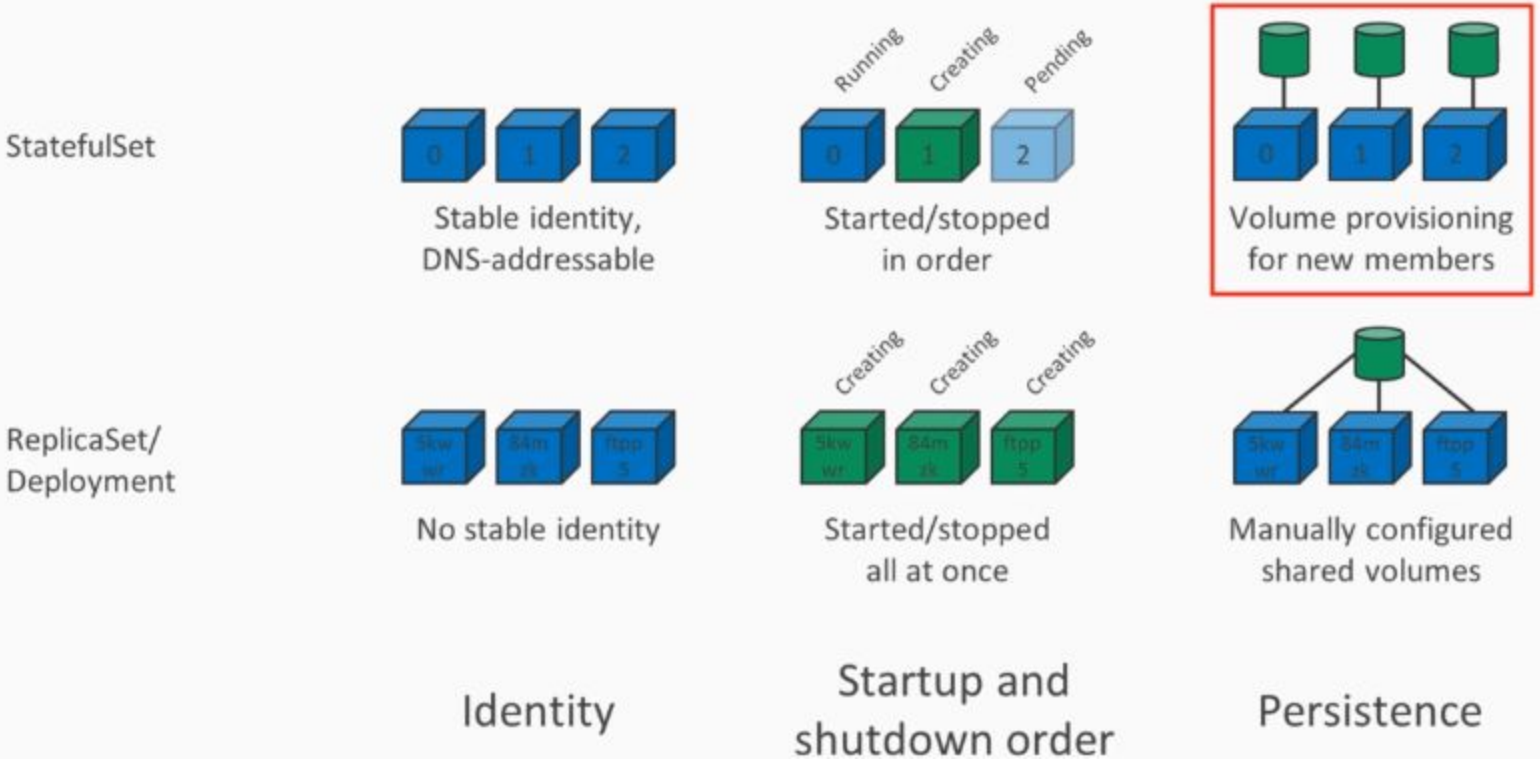
StatefulSets Versus Deployments



StatefulSets Versus Deployments



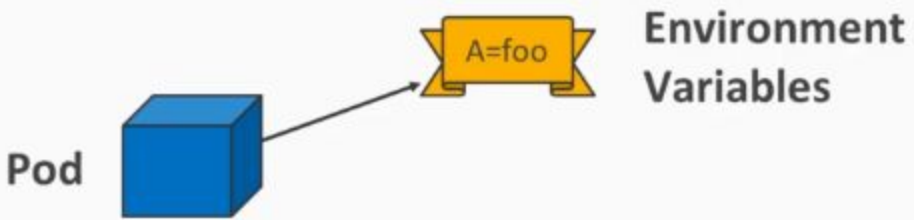
StatefulSets Versus Deployments



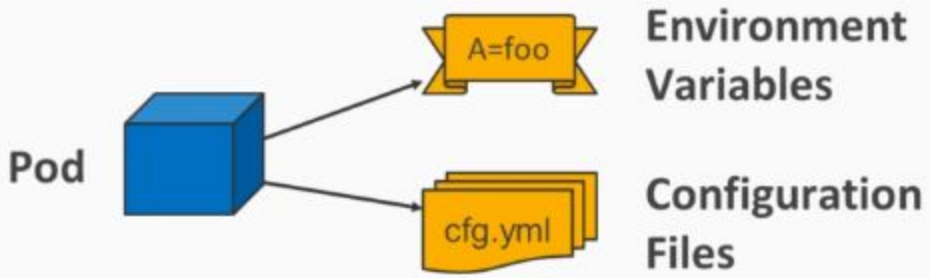
Managing Configuration Data

Pod Configuration

Pod Configuration



Pod Configuration

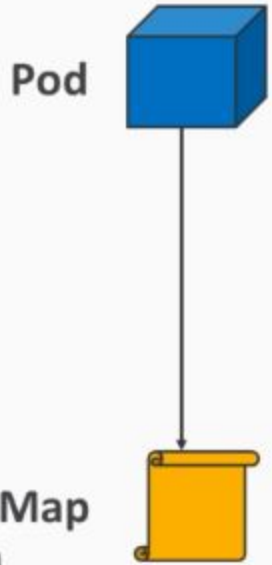


Pod Configuration

Environment Variables:

```
env:
- name: SOME_VARIABLE
  value: "some value"
```

Pod Configuration



Environment Variables:

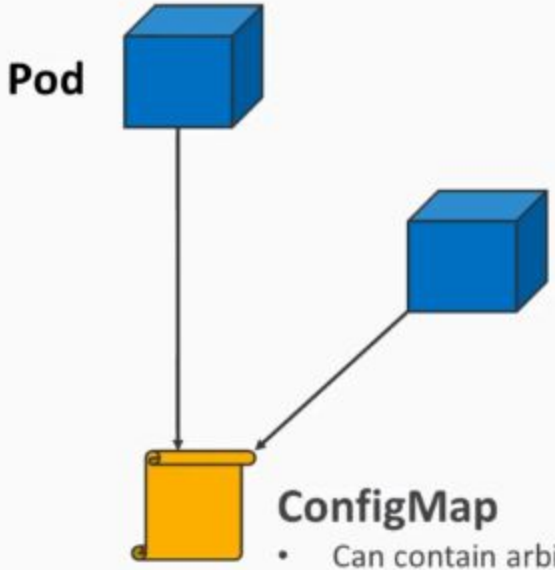
```
env:
- name: SOME_VARIABLE
  valueFrom:
    configMapKeyRef:
      name: my-configmap
      key: someVariable
```

- Can contain arbitrary key/value pairs

ConfigMap contents:

```
data:
  someVariable: "some value"
```

Pod Configuration



ConfigMap

- Can contain arbitrary key/value pairs
- Can be used by multiple Pods at once
- Can be updated independently of the Pod

ConfigMap Example

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-configmap
data:
  someVariable: "some value“
  someBoolean: "true"
```

ConfigMap Example

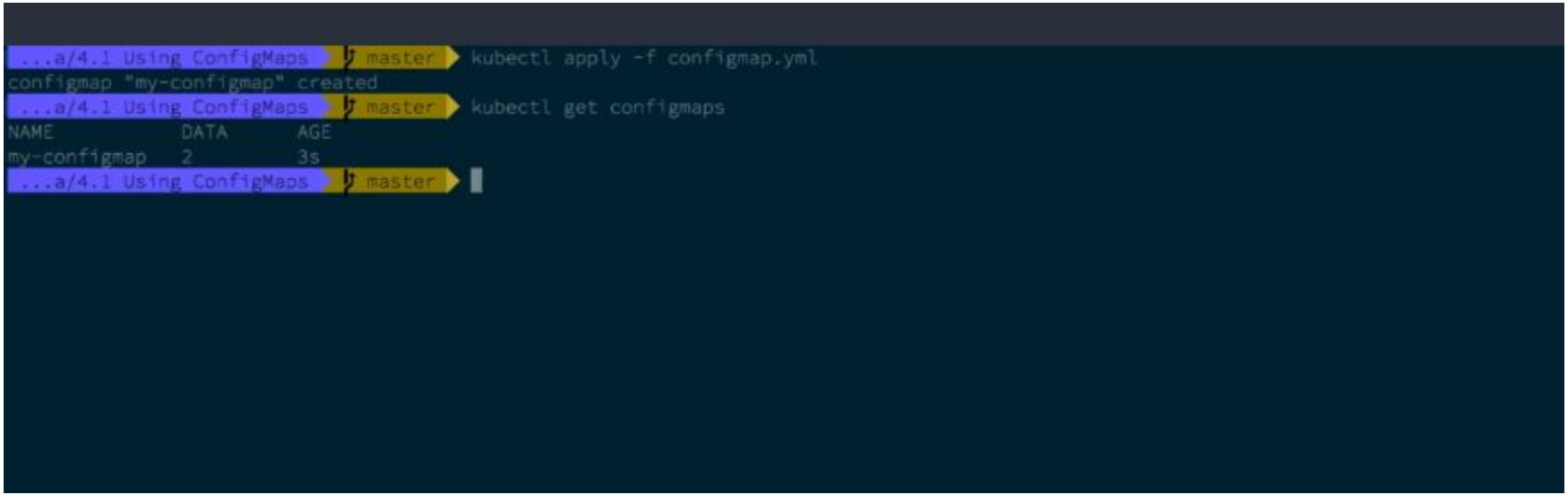
```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-configmap
data:
  someVariable: "some value“
  someBoolean: "true"
```

Key/value map

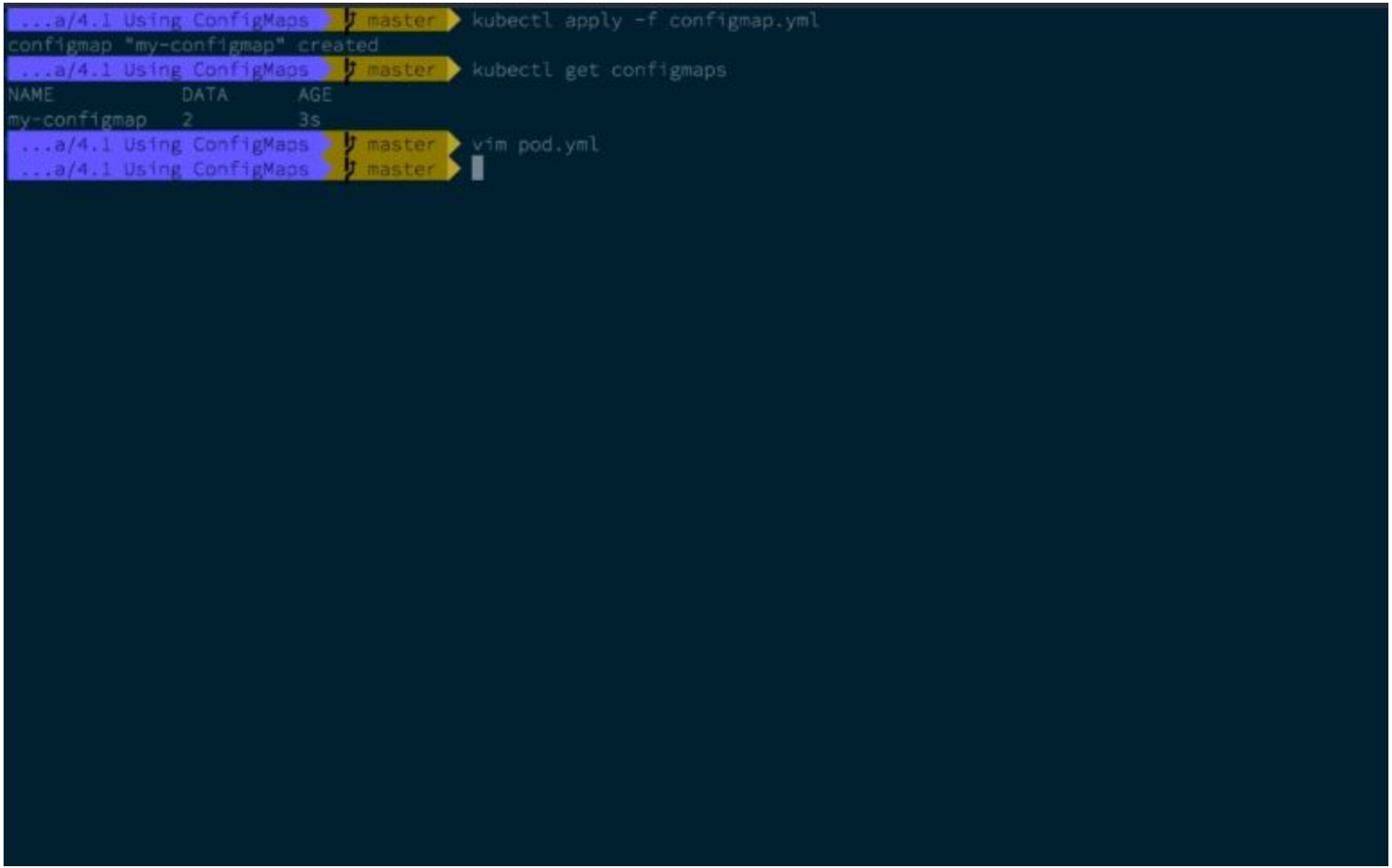
Caution:
Values must
be strings!

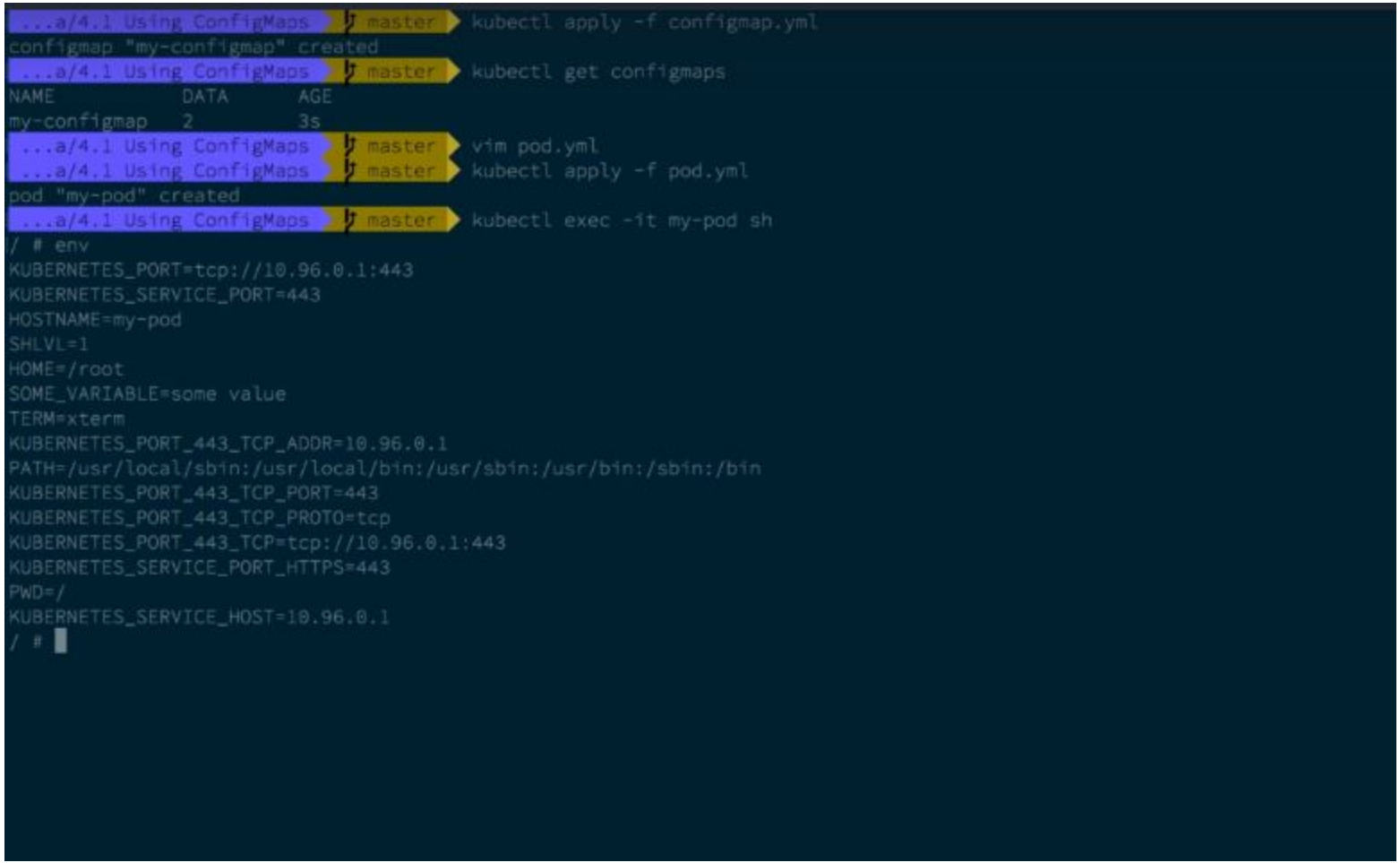
ConfigMap Usage Example

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  containers:
  - name: my-service
    image: nginx:1.3.12
    env:
    - name: SOME_VARIABLE
-     value: some value
+     valueFrom:
+       configMapKeyRef:
+         name: my-configmap
+         key: someVariable
```


...a/4.1 Using ConfigMaps master kubectl apply -f configmap.yml
configmap "my-configmap" created
...a/4.1 Using ConfigMaps master kubectl get configmaps
NAME DATA AGE
my-configmap 2 3s
...a/4.1 Using ConfigMaps master

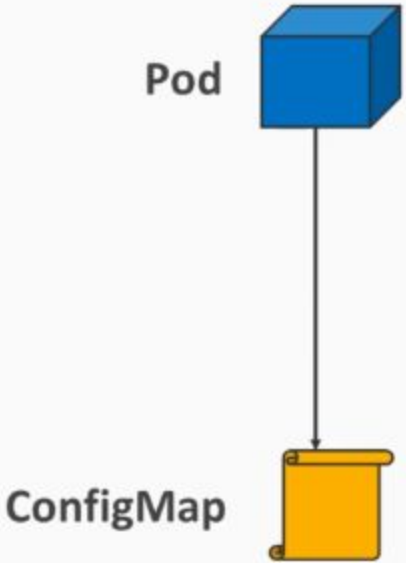
```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  containers:
  - name: app
    image: alpine
    command: ["sleep", "3600"]
    env:
    - name: SOME_VARIABLE
      valueFrom:
        configMapKeyRef:
          name: my-configmap
          key: someVariable
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"pod.yml" 15L, 267C
```


...a/4.1 Using ConfigMaps master kubectl apply -f configmap.yml
configmap "my-configmap" created
...a/4.1 Using ConfigMaps master kubectl get configmaps
NAME DATA AGE
my-configmap 2 3s
...a/4.1 Using ConfigMaps master vim pod.yml
...a/4.1 Using ConfigMaps master


...a/4.1 Using ConfigMaps master kubectl apply -f configmap.yml
configmap "my-configmap" created
...a/4.1 Using ConfigMaps master kubectl get configmaps
NAME DATA AGE
my-configmap 2 3s
...a/4.1 Using ConfigMaps master vim pod.yml
...a/4.1 Using ConfigMaps master kubectl apply -f pod.yml
pod "my-pod" created
...a/4.1 Using ConfigMaps master kubectl exec -it my-pod sh
/ # env
KUBERNETES_PORT=tcp://10.96.0.1:443
KUBERNETES_SERVICE_PORT=443
HOSTNAME=my-pod
SHLVL=1
HOME=/root
SOME_VARIABLE=some value
TERM=xterm
KUBERNETES_PORT_443_TCP_ADDR=10.96.0.1
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
KUBERNETES_PORT_443_TCP_PORT=443
KUBERNETES_PORT_443_TCP_PROTO=tcp
KUBERNETES_PORT_443_TCP=tcp://10.96.0.1:443
KUBERNETES_SERVICE_PORT_HTTPS=443
PWD=/
KUBERNETES_SERVICE_HOST=10.96.0.1
/ #

Inject Configuration Files Pods

Previously



Environment Variables:

```
env:
- name: SOME_VARIABLE
  valueFrom:
    configMapKeyRef:
      name: my-configmap
      key: someVariable
```

ConfigMap contents:

```
data:
  someVariable: "some value"
```

ConfigMap Example

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-configmap
data:
  config.ini: |
    some_variable = foo
    some_other_variable = baz
  database.ini: |
    uri = mongodb://my-
database:27017/…
```

YAML multiline string

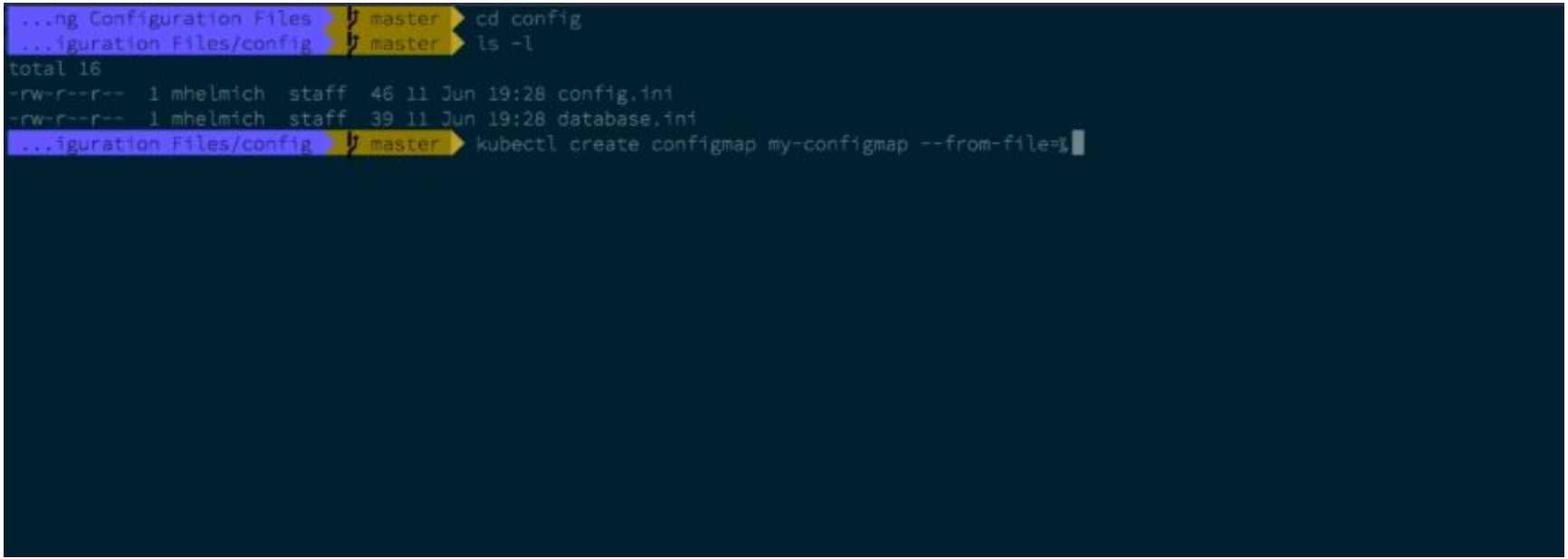
ConfigMap Usage Example

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  containers:
  - name: my-service
    image: nginx:1.3.12
+   volumeMounts:
+   - name: config
+     mountPath: /etc/config
+ volumes:
+ - name: config
+   configMap:
+     name: my-configmap
```

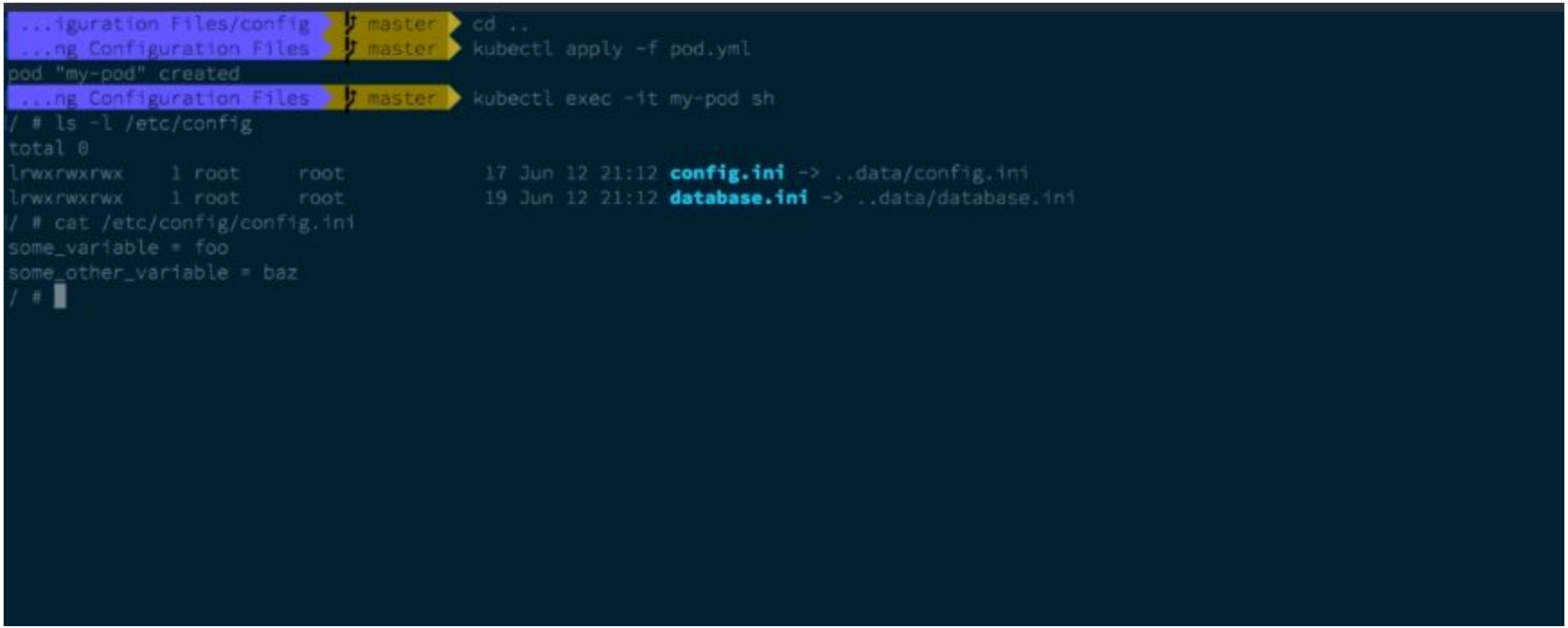
ConfigMap Usage Example

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  containers:
  - name: my-service
    image: nginx:1.3.12
+   volumeMounts:
+   - name: config
+     mountPath: /etc/config
+ volumes:
+ - name: config
+   configMap:
+     name: my-configmap
```

This mount is automatically updated when the ConfigMap changes!

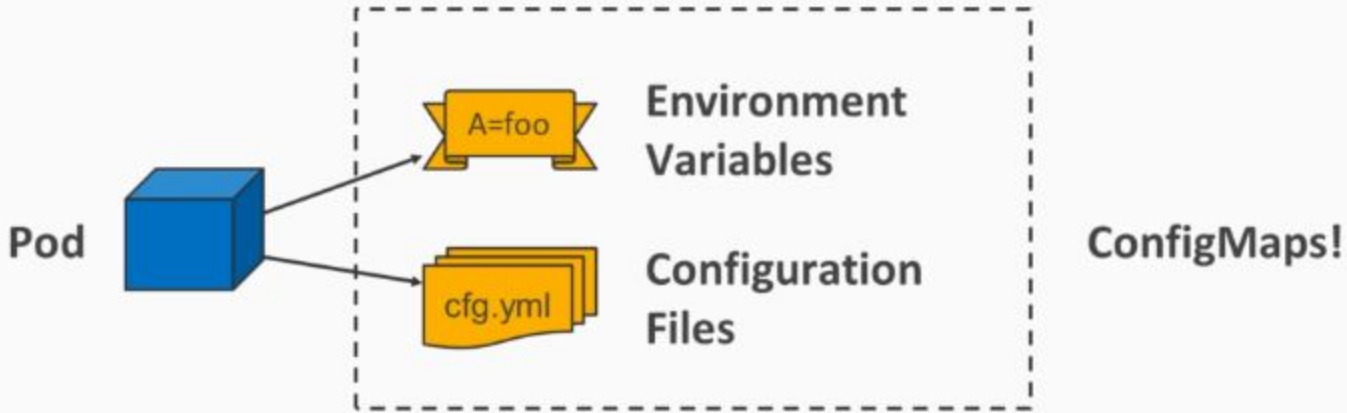

...ng Configuration Files master cd config
...iguration Files/config master ls -l
total 16
-rw-r--r-- 1 mhelmich staff 46 11 Jun 19:28 config.ini
-rw-r--r-- 1 mhelmich staff 39 11 Jun 19:28 database.ini
...iguration Files/config master kubectl create configmap my-configmap --from-file=%

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: v1
data:
  config.ini: |
    some_variable = foo
    some_other_variable = baz
  database.ini: |
    uri = mongodb://my-database:27017/test
kind: ConfigMap
metadata:
  creationTimestamp: 2018-06-12T21:10:52Z
  name: my-configmap
  namespace: default
  resourceVersion: "58518"
  selfLink: /api/v1/namespaces/default/configmaps/my-configmap
  uid: 1698f66b-6e85-11e8-b829-080027c553e0
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"/var/folders/1z/z3f_tz6953jchlbnkswp0zy00000gn/T/kubectl-edit-llczx.yaml" 19L, 604C
```

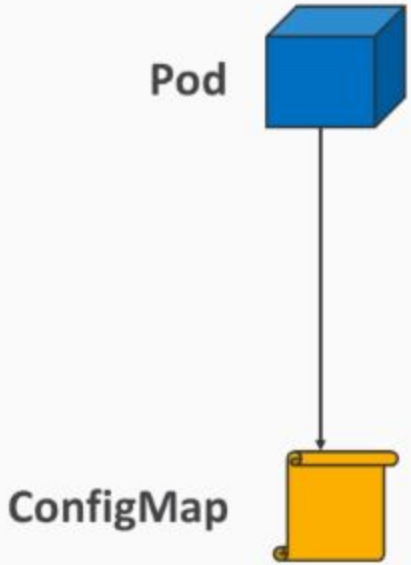

...iguration Files/config master cd ..
...ng Configuration Files master kubectl apply -f pod.yml
pod "my-pod" created
...ng Configuration Files master kubectl exec -it my-pod sh
/ # ls -l /etc/config
total 0
lrwxrwxrwx 1 root root 17 Jun 12 21:12 config.ini -> ..data/config.ini
lrwxrwxrwx 1 root root 19 Jun 12 21:12 database.ini -> ..data/database.ini
/ # cat /etc/config/config.ini
some_variable = foo
some_other_variable = baz
/ #

Keeping Secrets

Previously


Pod
A=foo
Environment Variables
cfg.yml
Configuration Files
ConfigMaps!

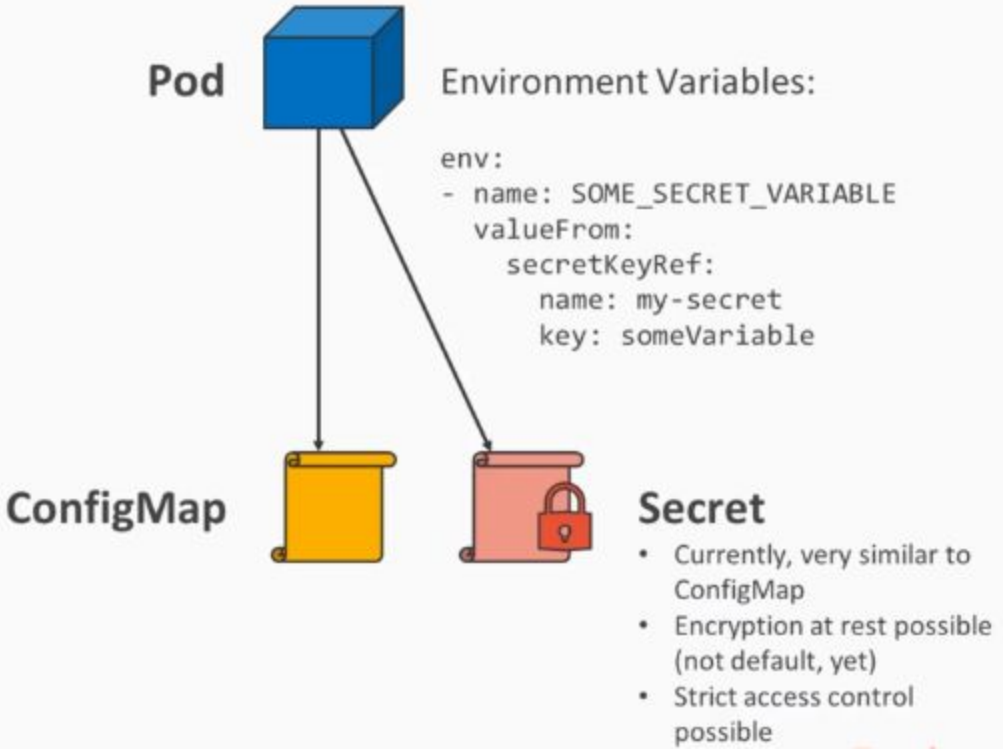
Pod Secrets



Environment Variables:

```
env:
- name: SOME_VARIABLE
  valueFrom:
    configMapKeyRef:
      name: my-configMap
      key: someVariable
```

Pod Secrets


Pod
Environment Variables:
env:
- name: SOME_SECRET_VARIABLE
valueFrom:
secretKeyRef:
name: my-secret
key: someVariable
ConfigMap
Secret
• Currently, very similar to ConfigMap
• Encryption at rest possible (not default, yet)
• Strict access control possible

Secret Example

```
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
data:
  test: VGVzdA==
```

Secret Example

```
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
data:
  test: VGVzdA==
```

Arbitrary key/value pairs
Values are base64 encoded

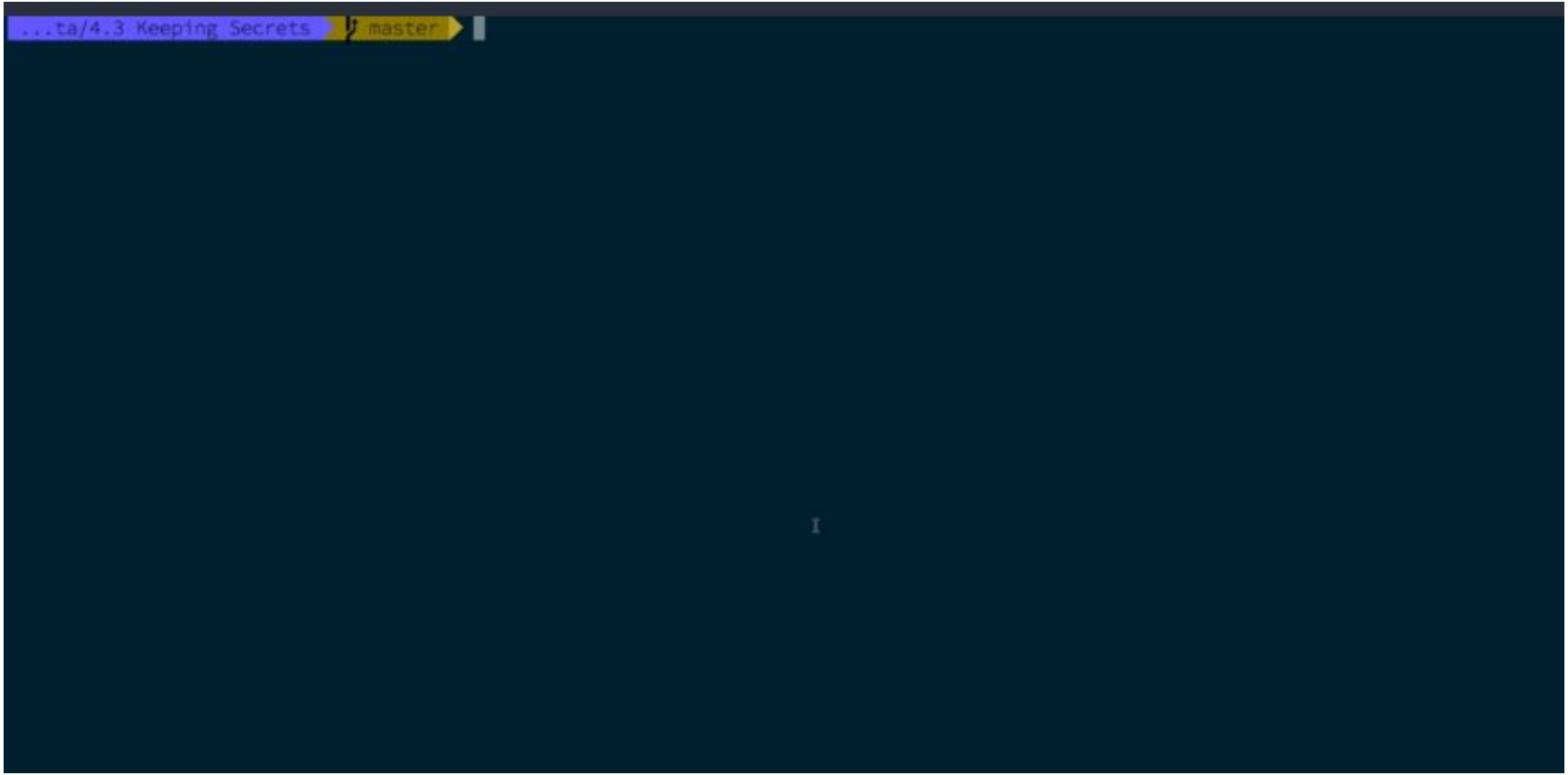
base64 is not encryption!
base64 is not encryption!
base64 is not encryption!
base64 is not encryption!
base64 is not encryption!
base64 is not encryption!
base64 is not encryption!

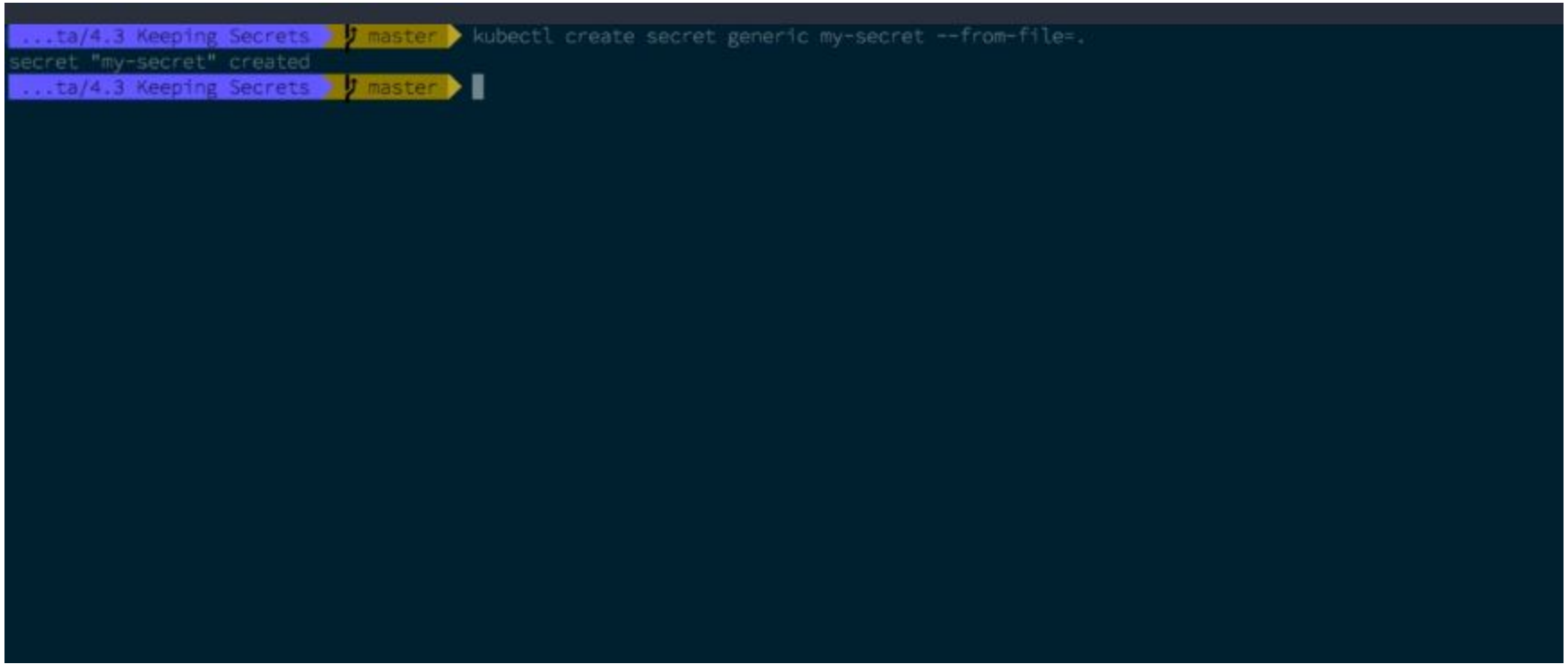
Secret Usage in Environment Variables

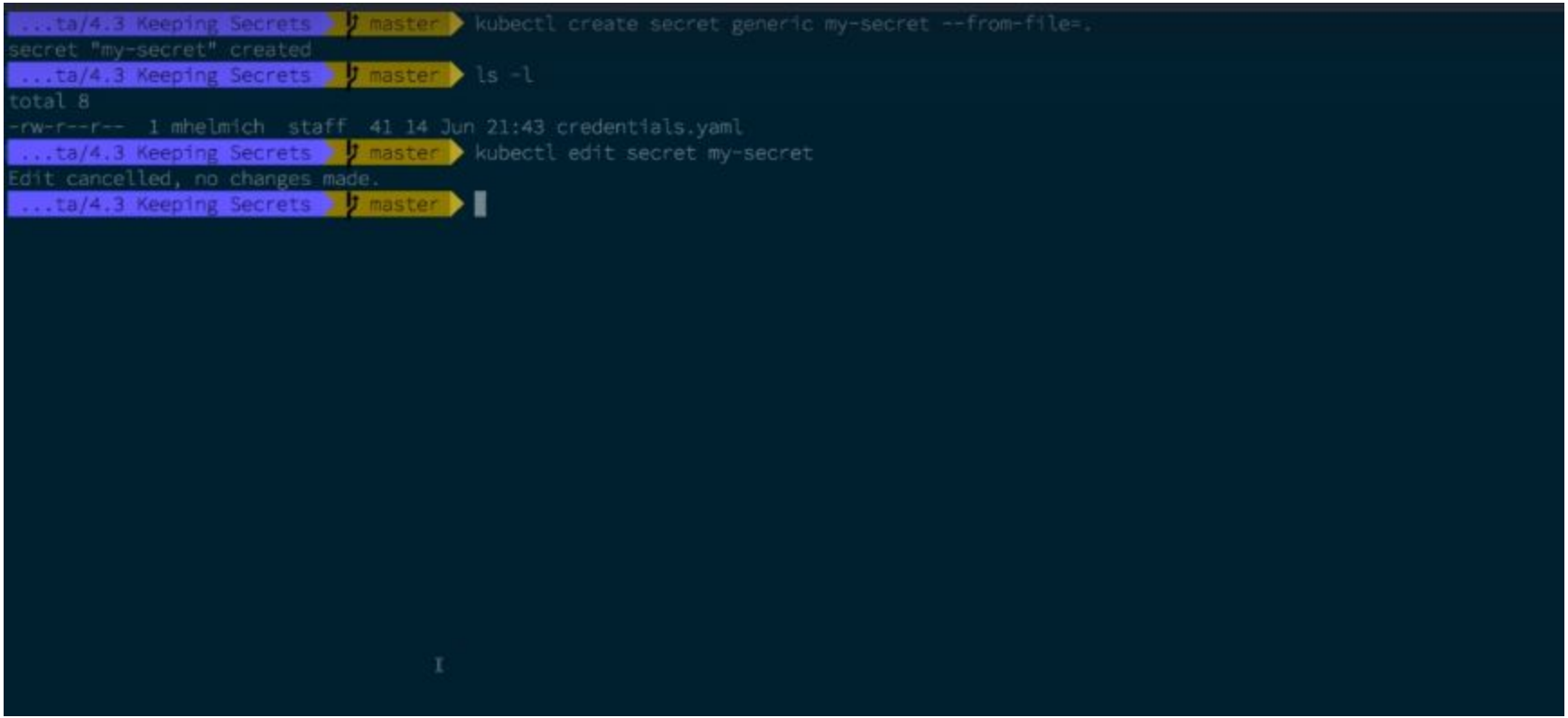
```
 apiVersion: v1
 kind: Pod
 metadata:
   name: my-pod
 spec:
   containers:
   - name: my-service
     image: nginx:1.3.12
+    env:
+    - name: SOME_SECRET_VARIABLE
+      valueFrom:
+        secretKeyRef:
+          name: my-secret
+          key: someVariable
```

Secret Usage in Volumes

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  containers:
  - name: my-service
    image: nginx:1.3.12
+   volumeMounts:
+   - name: secret
+     mountPath: /etc/secret
+ volumes:
+ - name: secret
+   secret:
+     name: my-secret
```


...ta/4.3 Keeping Secrets
master


...ta/4.3 Keeping Secrets master kubectl create secret generic my-secret --from-file=.
secret "my-secret" created
...ta/4.3 Keeping Secrets master


...ta/4.3 Keeping Secrets master kubectl create secret generic my-secret --from-file=.
secret "my-secret" created
...ta/4.3 Keeping Secrets master ls -l
total 8
-rw-r--r-- 1 mhelmich staff 41 14 Jun 21:43 credentials.yaml
...ta/4.3 Keeping Secrets master kubectl edit secret my-secret
Edit cancelled, no changes made.
...ta/4.3 Keeping Secrets master

Composing Environment Variables

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  containers:
  - name: my-service
    image: nginx:1.3.12
    env:
    - name: MONGODB_PASSWORD
      valueFrom:
        secretKeyRef:
          name: mongodb-credentials
          key: password
    - name: MONGODB_URL
      value: "mongodb://user:$(MONGODB_PASSWORD)@mongo/database"
```

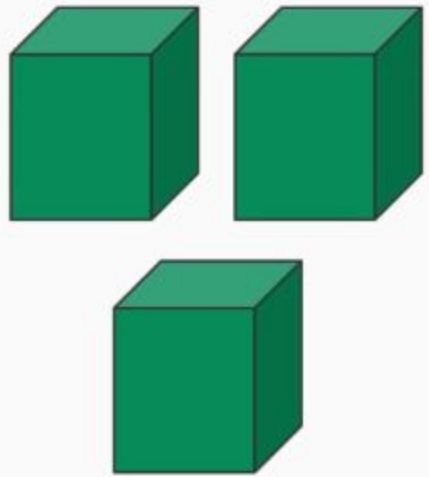
Composing Environment Variables

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  containers:
  - name: my-service
    image: nginx:1.3.12
    env:
    - name: MONGODB_PASSWORD
      valueFrom:
        secretKeyRef:
          name: mongodb-credentials
          key: password
    - name: MONGODB_URL
      value: "mongodb://user:$(MONGODB_PASSWORD)@mongo/database"
```

Continuous Delivery with Kubernetes

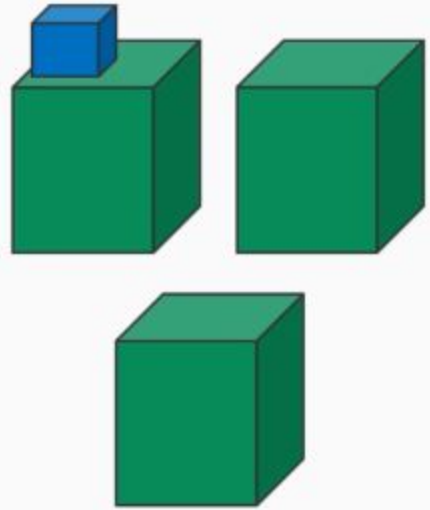
Characteristics of a CI/CD Pipeline

Continuous What?



Your Kubernetes Cluster

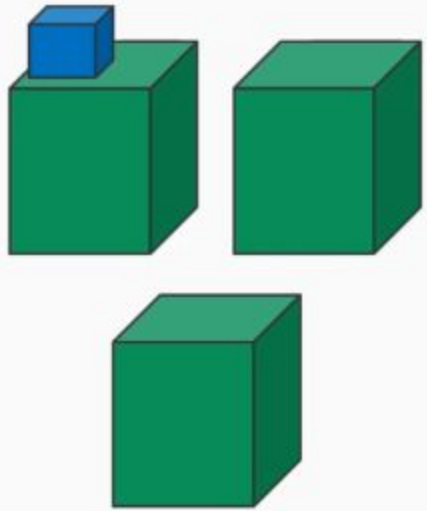
Continuous What?



Your Kubernetes Cluster

Continuous What?

Your Source Code Repository



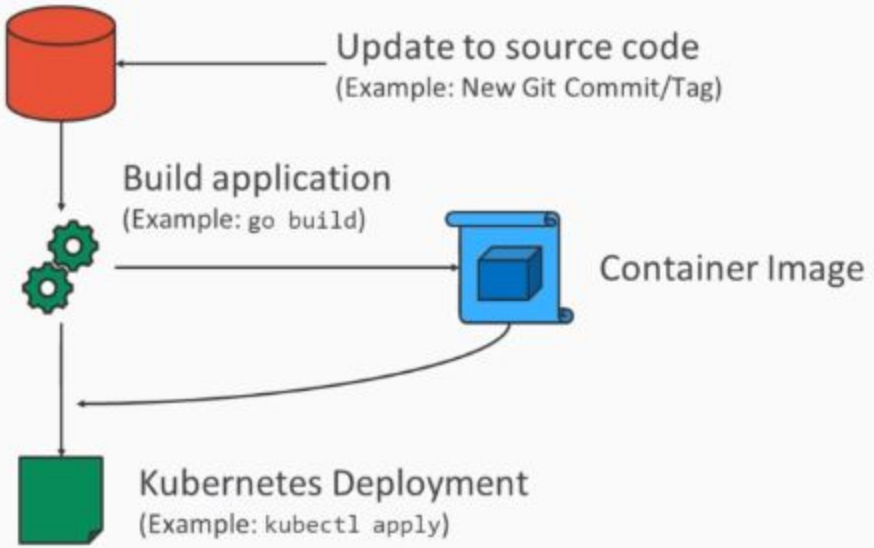
Your Kubernetes Cluster

A Typical Build Pipeline

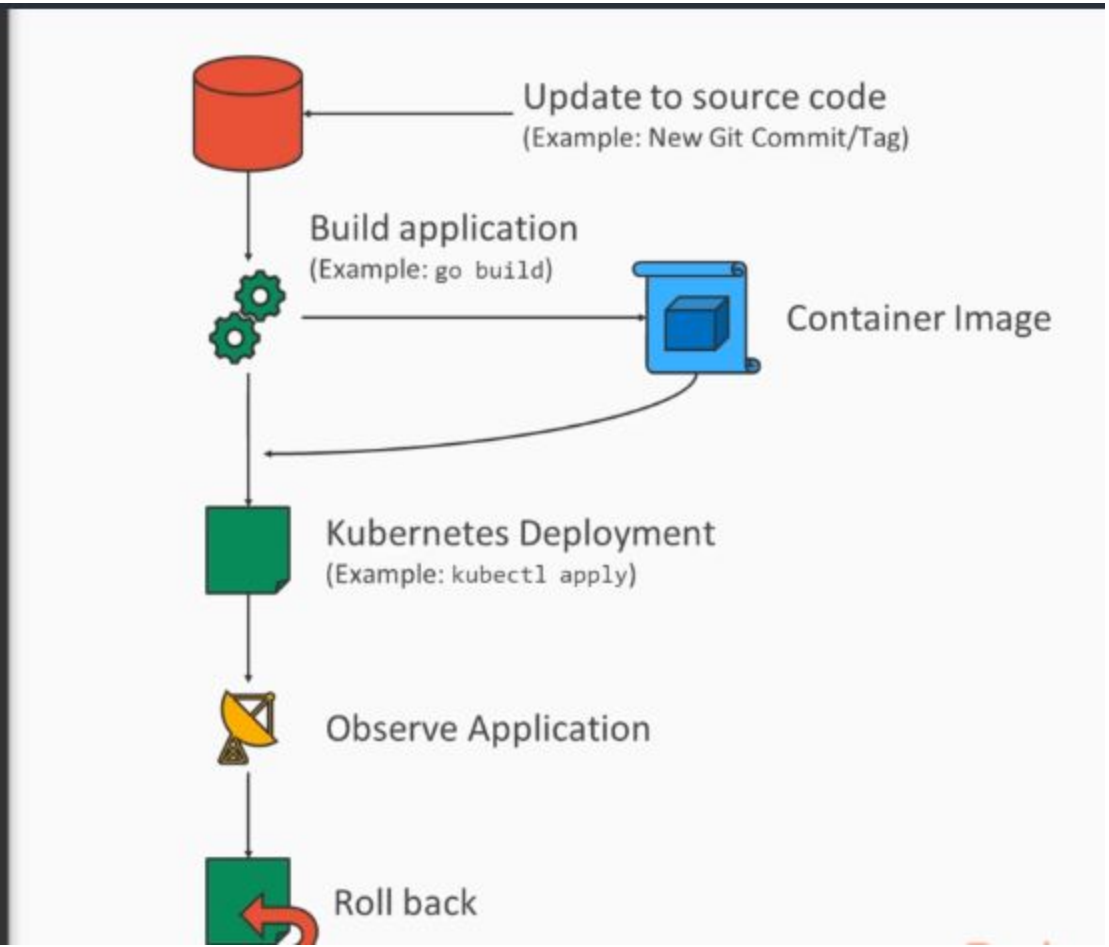
A Typical Build Pipeline

A Typical Build Pipeline

A Typical Build Pipeline



A Typical Build Pipeline



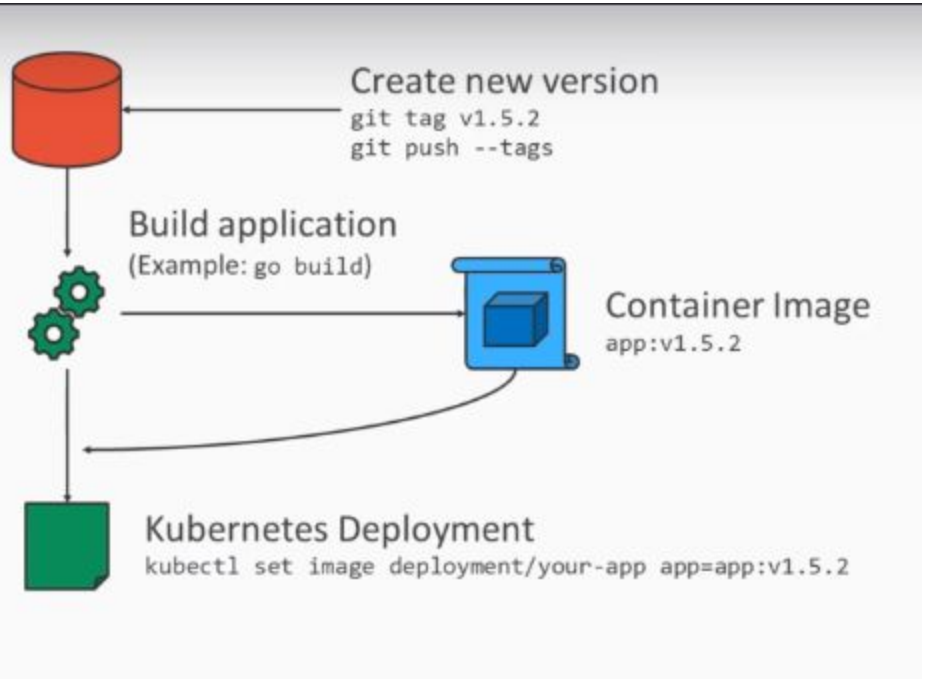
How to Release a Specific Version?

Create new version

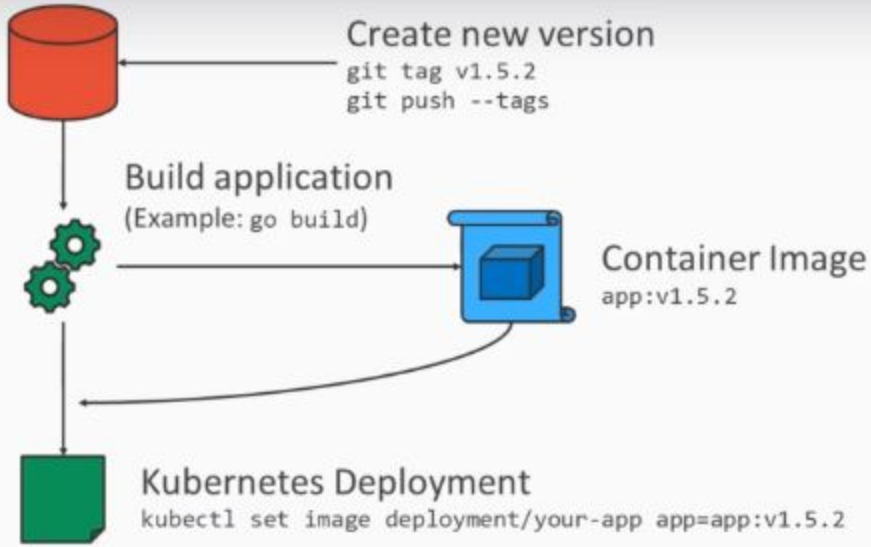
```
git tag v1.5.2
git push --tags
```

How to Release a Specific Version?

How to Release a Specific Version?



How to Release a Specific Version?



Caution!

- Using set image requires that Deployment already exists
- What about Updates to other parts of the Pod spec (environment variables, ports, and so on)?

Later in This Section

GitLab

Configuring CD with GitLab


Why
GitLab ?

Prerequisites

- Running GitLab instance with CI runner (alternatively: hosted GitLab at gitlab.com)
- Running Kubernetes cluster, reachable from CI runner
- Alternatively: Have a look at the example files for this video for a quick and easy (but not production-ready) GitLab setup on Minikube

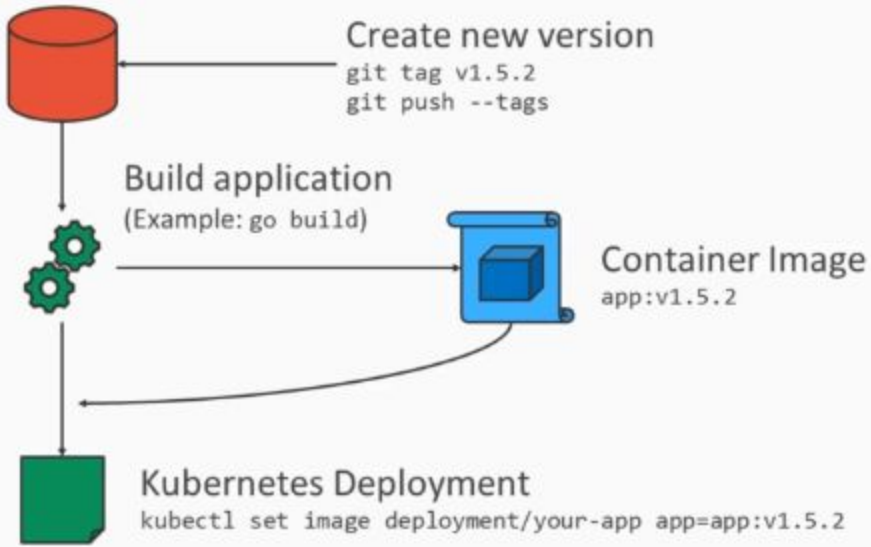
GitLab

kubernetes

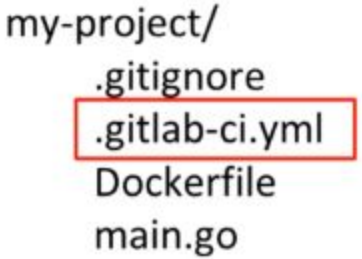
Set Up GitLab on Kubernetes

https://docs.gitlab.com/ce/install/kubernetes/

The Goal


Create new version
git tag v1.5.2
git push --tags
Build application
(Example: go build)
Container Image
app:v1.5.2
Kubernetes Deployment
kubectl set image deployment/your-app app=app:v1.5.2

```
my-project/
        .gitignore
        .gitlab-ci.yml
        Dockerfile
        main.go
```

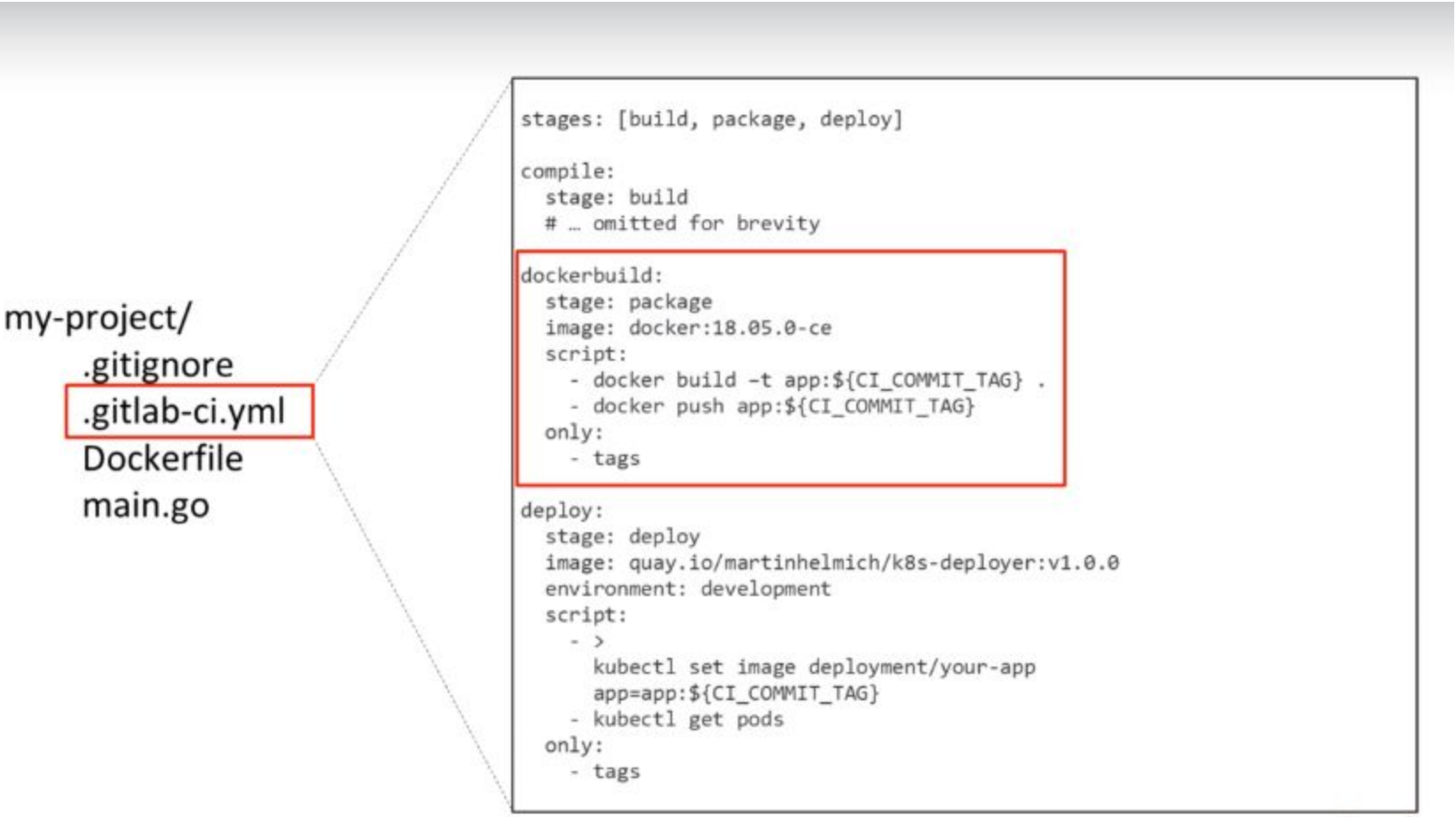

my-project/
.gitignore
.gitlab-ci.yml
Dockerfile
main.go

my-project/

- .gitignore
- .gitlab-ci.yml
- Dockerfile
- main.go

```
stages: [build, package, deploy]

compile:
  stage: build
  # … omitted for brevity

dockerbuild:
  stage: package
  image: docker:18.05.0-ce
  script:
    - docker build -t app:${CI_COMMIT_TAG} .
    - docker push app:${CI_COMMIT_TAG}
  only:
    - tags

deploy:
  stage: deploy
  image: quay.io/martinhelmich/k8s-deployer:v1.0.0
  environment: development
  script:
    - >
      kubectl set image deployment/your-app
      app=app:${CI_COMMIT_TAG}
    - kubectl get pods
  only:
    - tags
```


my-project/
.gitignore
.gitlab-ci.yml
Dockerfile
main.go
stages: [build, package, deploy]
compile:
stage: build
… omitted for brevity
dockerbuild:
stage: package
image: docker:18.05.0-ce
script:
- docker build -t app:${CI_COMMIT_TAG} .
- docker push app:${CI_COMMIT_TAG}
only:
- tags
deploy:
stage: deploy
image: quay.io/martinhelmich/k8s-deployer:v1.0.0
environment: development
script:
- >
kubectl set image deployment/your-app
app=app:${CI_COMMIT_TAG}
- kubectl get pods
only:
- tags


my-project/
.gitignore
.gitlab-ci.yml
Dockerfile
main.go
stages: [build, package, deploy]
compile:
stage: build
… omitted for brevity
dockerbuild:
stage: package
image: docker:18.05.0-ce
script:
- docker build -t app:${CI_COMMIT_TAG} .
- docker push app:${CI_COMMIT_TAG}
only:
- tags
deploy:
stage: deploy
image: quay.io/martinhelmich/k8s-deployer:v1.0.0
environment: development
script:
- >
kubectl set image deployment/your-app
app=app:${CI_COMMIT_TAG}
- kubectl get pods
only:
- tags

my-project/

- .gitignore
- .gitlab-ci.yml
- Dockerfile
- main.go

```
stages: [build, package, deploy]

compile:
  stage: build
  # … omitted for brevity
```

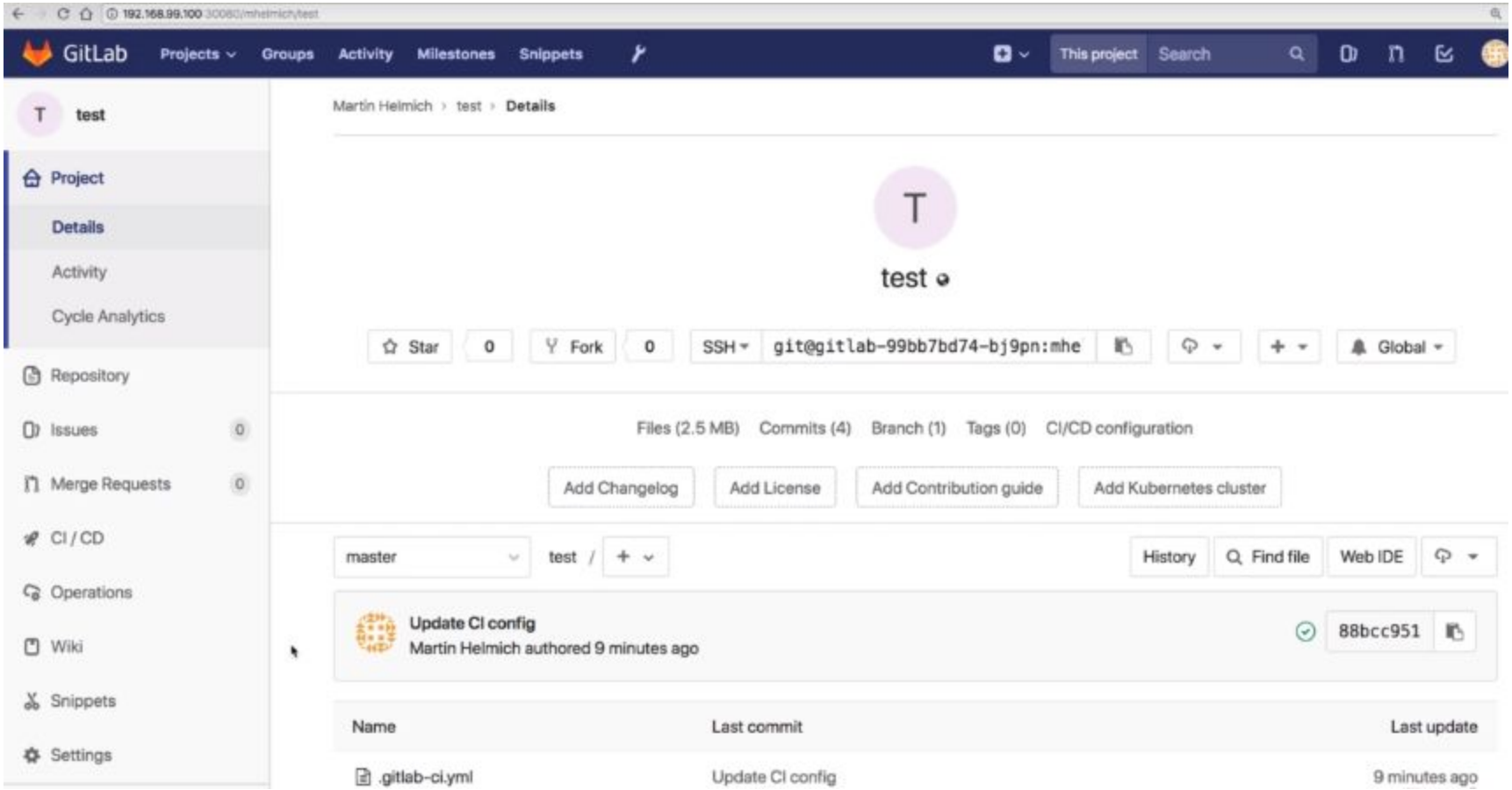
See documentation for more information
https://docs.gitlab.com/ce/ci/docker/using_docker_build.html

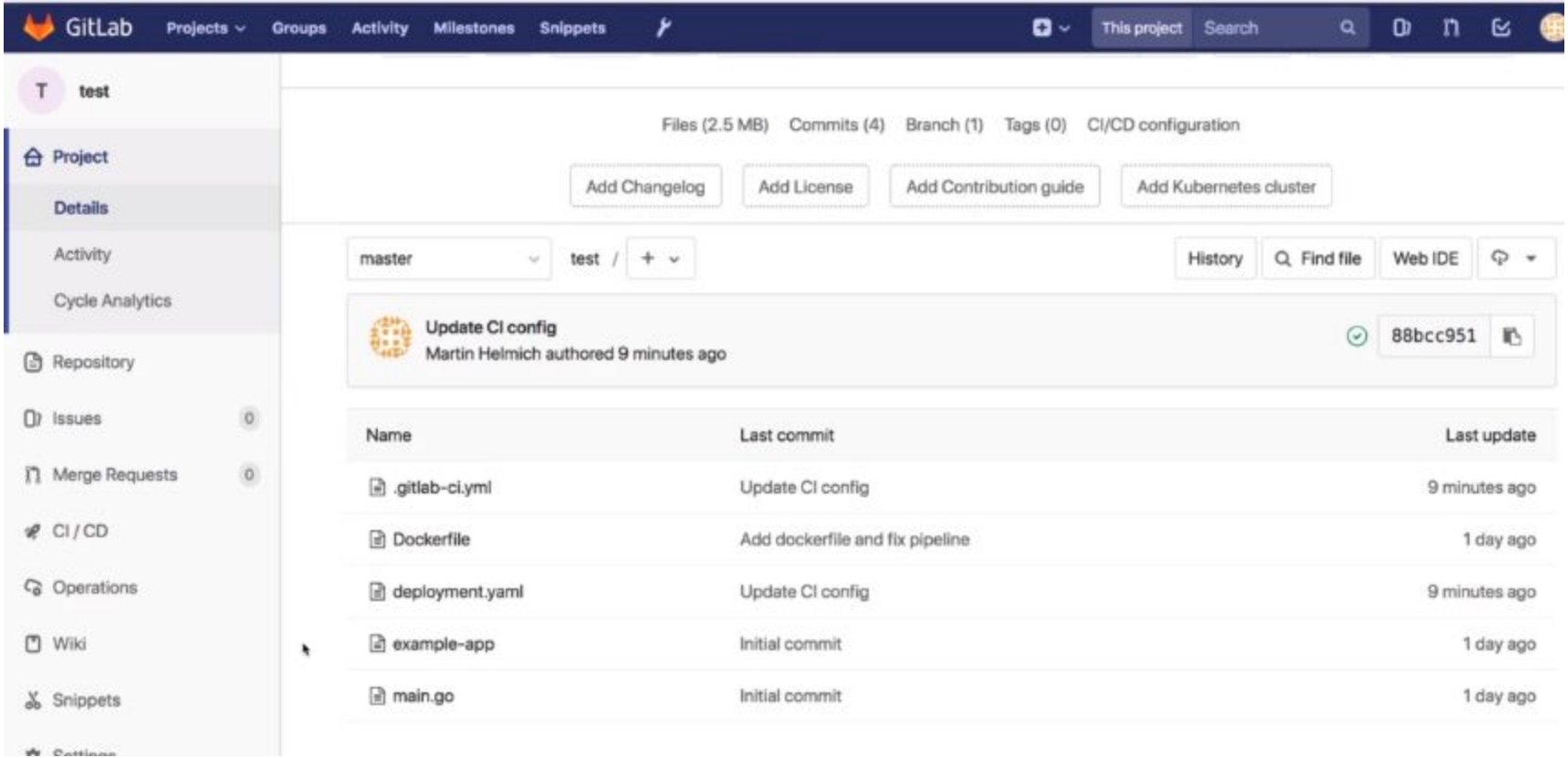
```
dockerbuild:
  stage: package
  image: docker:18.05.0-ce
  script:
    - docker build -t app:${CI_COMMIT_TAG} .
    - docker push app:${CI_COMMIT_TAG}
  only:
    - tags

deploy:
  stage: deploy
  image: quay.io/martinhelmich/k8s-deployer:v1.0.0
  environment: development
  script:
    - >
      kubectl set image deployment/your-app
      app=app:${CI_COMMIT_TAG}
    - kubectl get pods
  only:
    - tags
```

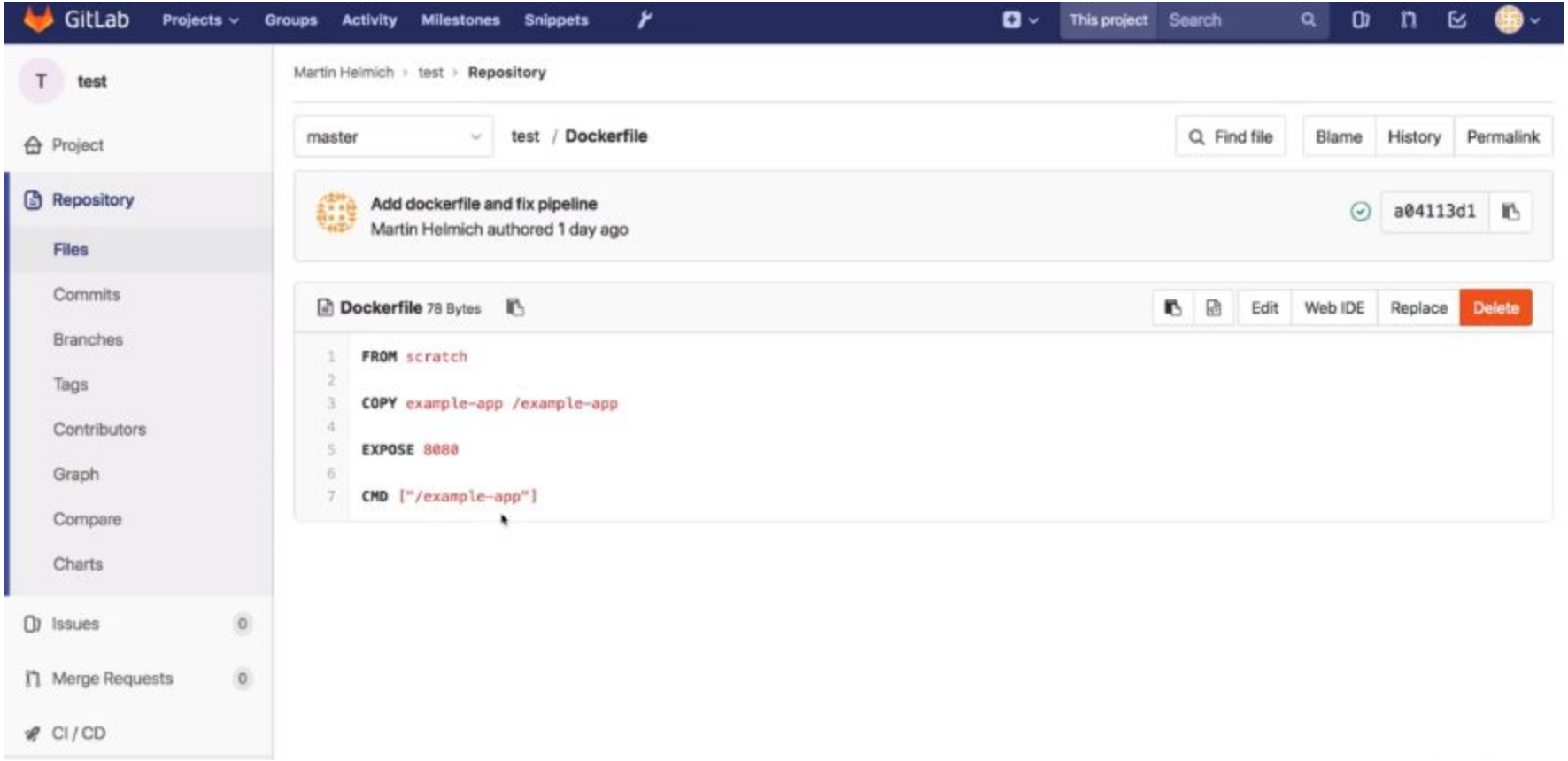
my-project/

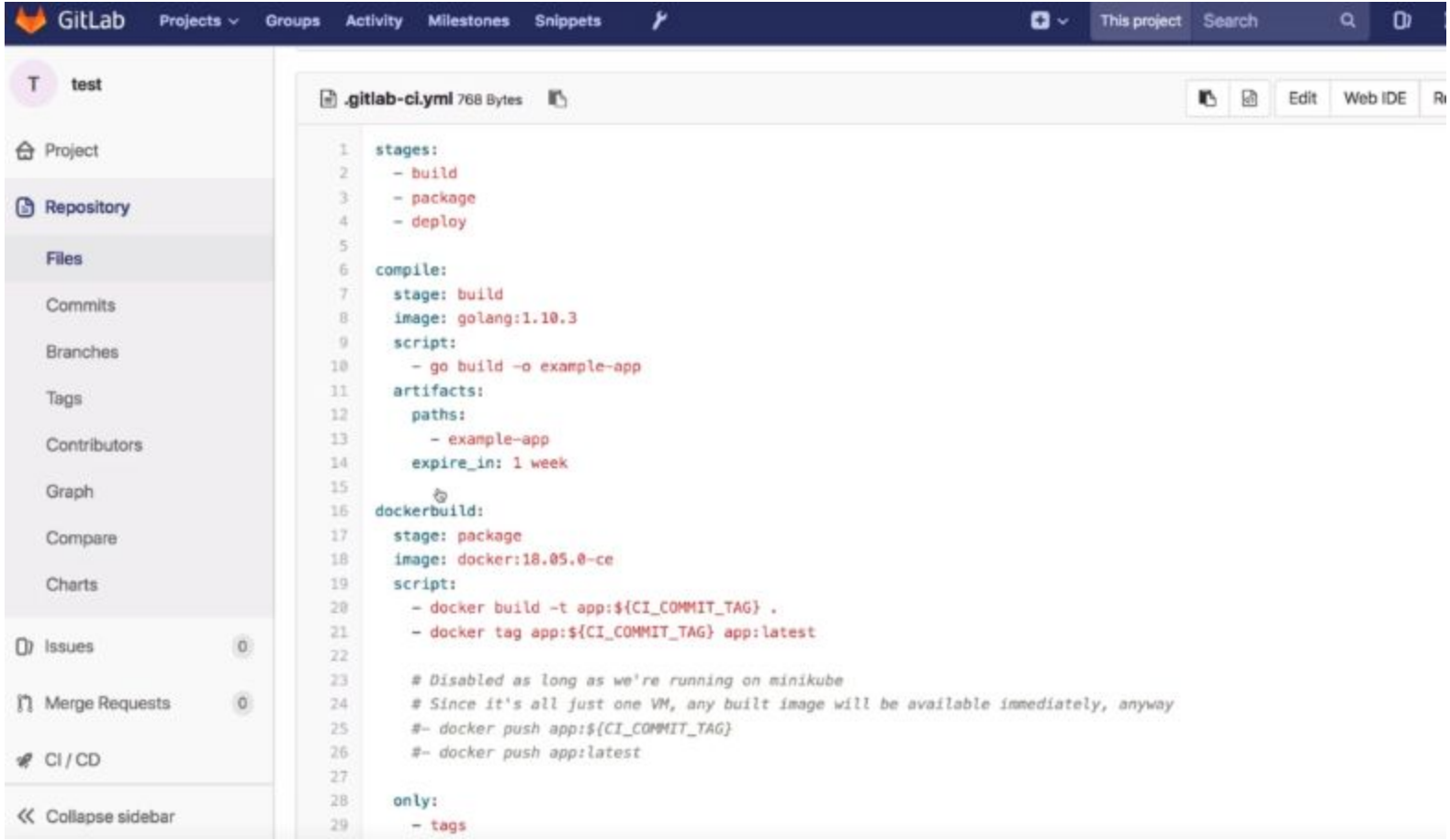
- .gitignore
- .gitlab-ci.yml
- Dockerfile
- main.go

```
stages: [build, package, deploy]

compile:
  stage: build
  # … omitted for brevity

dockerbuild:
  stage: package
  image: docker:18.05.0-ce
  script:
    - docker build -t app:${CI_COMMIT_TAG} .
    - docker push app:${CI_COMMIT_TAG}
  only:
    - tags

deploy:
  stage: deploy
  image: quay.io/martinhelmich/k8s-deployer:v1.0.0
  environment: development
  script:
    - >
      kubectl set image deployment/your-app
      app=app:${CI_COMMIT_TAG}
    - kubectl get pods
  only:
    - tags
```

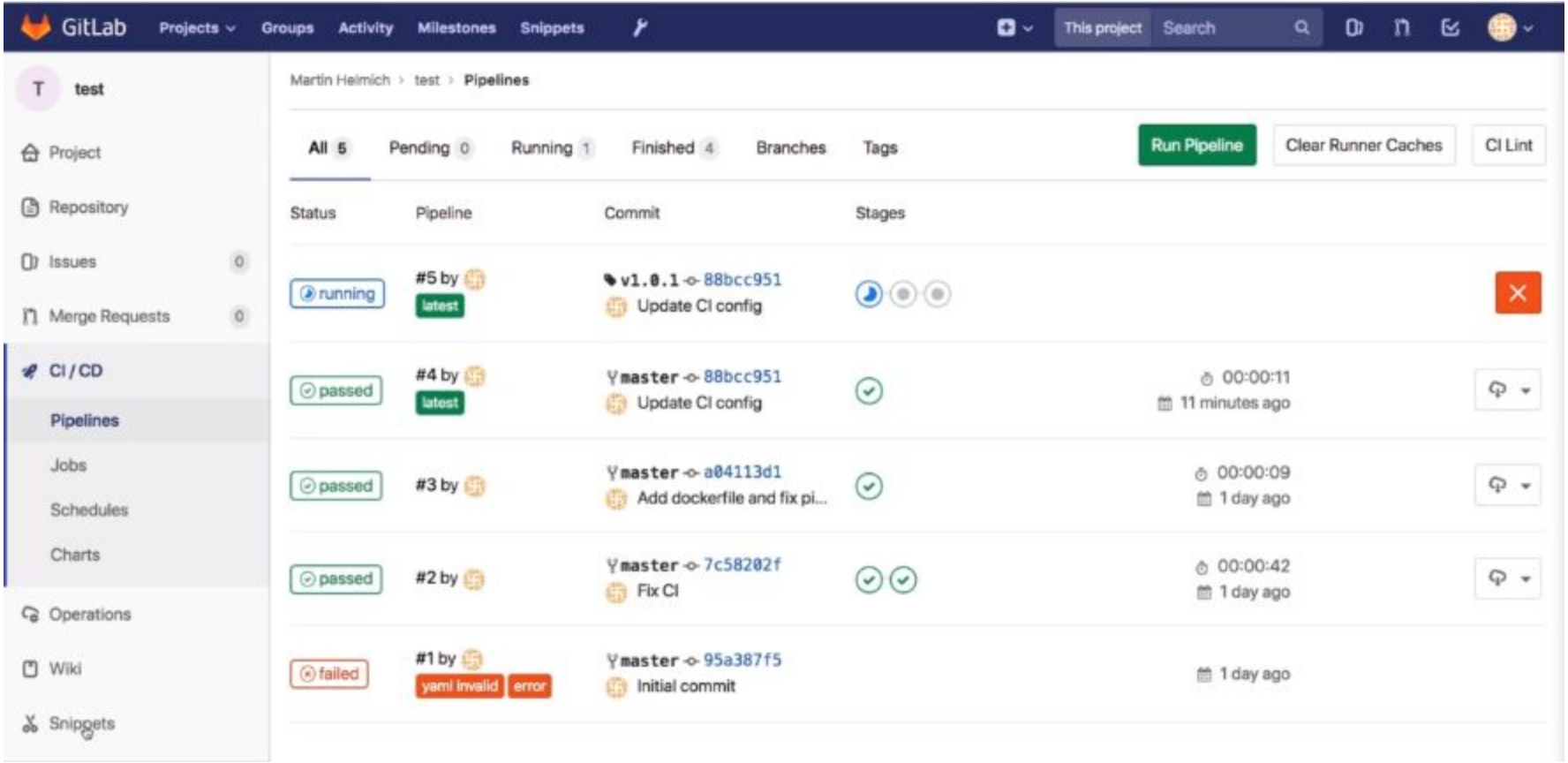

GitLab
Projects
Groups
Activity
Milestones
Snippets
This project
Search
test
Project
Details
Activity
Cycle Analytics
Repository
Issues 0
Merge Requests 0
CI / CD
Operations
Wiki
Snippets
Settings
Martin Helmich › test › Details
T
test
Star 0
Fork 0
SSH
git@gitlab-99bb7bd74-bj9pn:mhe
Global
Files (2.5 MB)
Commits (4)
Branch (1)
Tags (0)
CI/CD configuration
Add Changelog
Add License
Add Contribution guide
Add Kubernetes cluster
master
test /
History
Find file
Web IDE
Update CI config
Martin Helmich authored 9 minutes ago
88bcc951
Name
Last commit
Last update
.gitlab-ci.yml
Update CI config
9 minutes ago

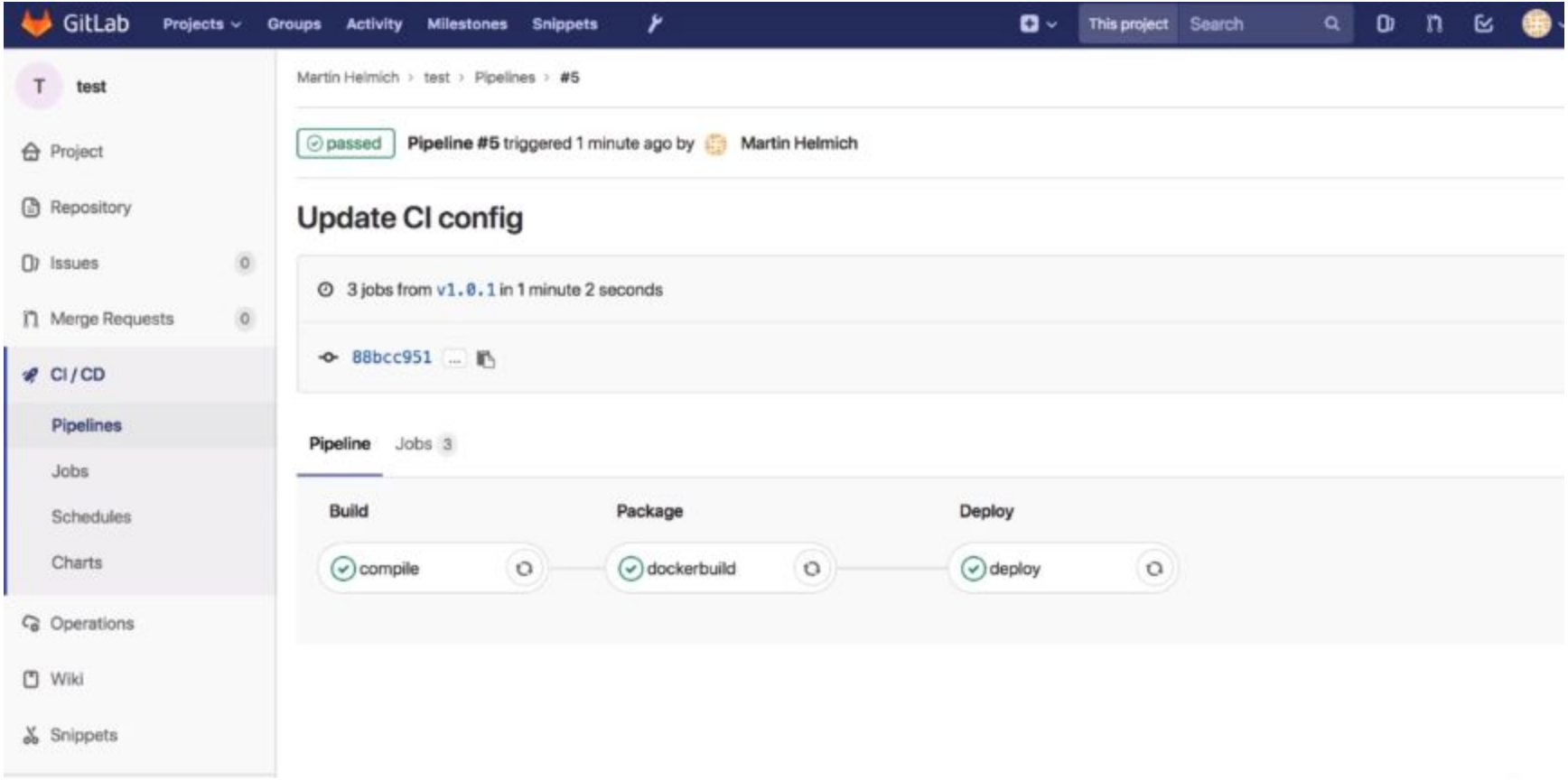

GitLab
Projects
Groups
Activity
Milestones
Snippets
This project
Search
T
test
Project
Details
Activity
Cycle Analytics
Repository
Issues 0
Merge Requests 0
CI / CD
Operations
Wiki
Snippets
Files (2.5 MB)
Commits (4)
Branch (1)
Tags (0)
CI/CD configuration
Add Changelog
Add License
Add Contribution guide
Add Kubernetes cluster
master
test /
History
Find file
Web IDE
Update CI config
Martin Helmich authored 9 minutes ago
88bcc951
Name
Last commit
Last update
.gitlab-ci.yml
Update CI config
9 minutes ago
Dockerfile
Add dockerfile and fix pipeline
1 day ago
deployment.yaml
Update CI config
9 minutes ago
example-app
Initial commit
1 day ago
main.go
Initial commit
1 day ago

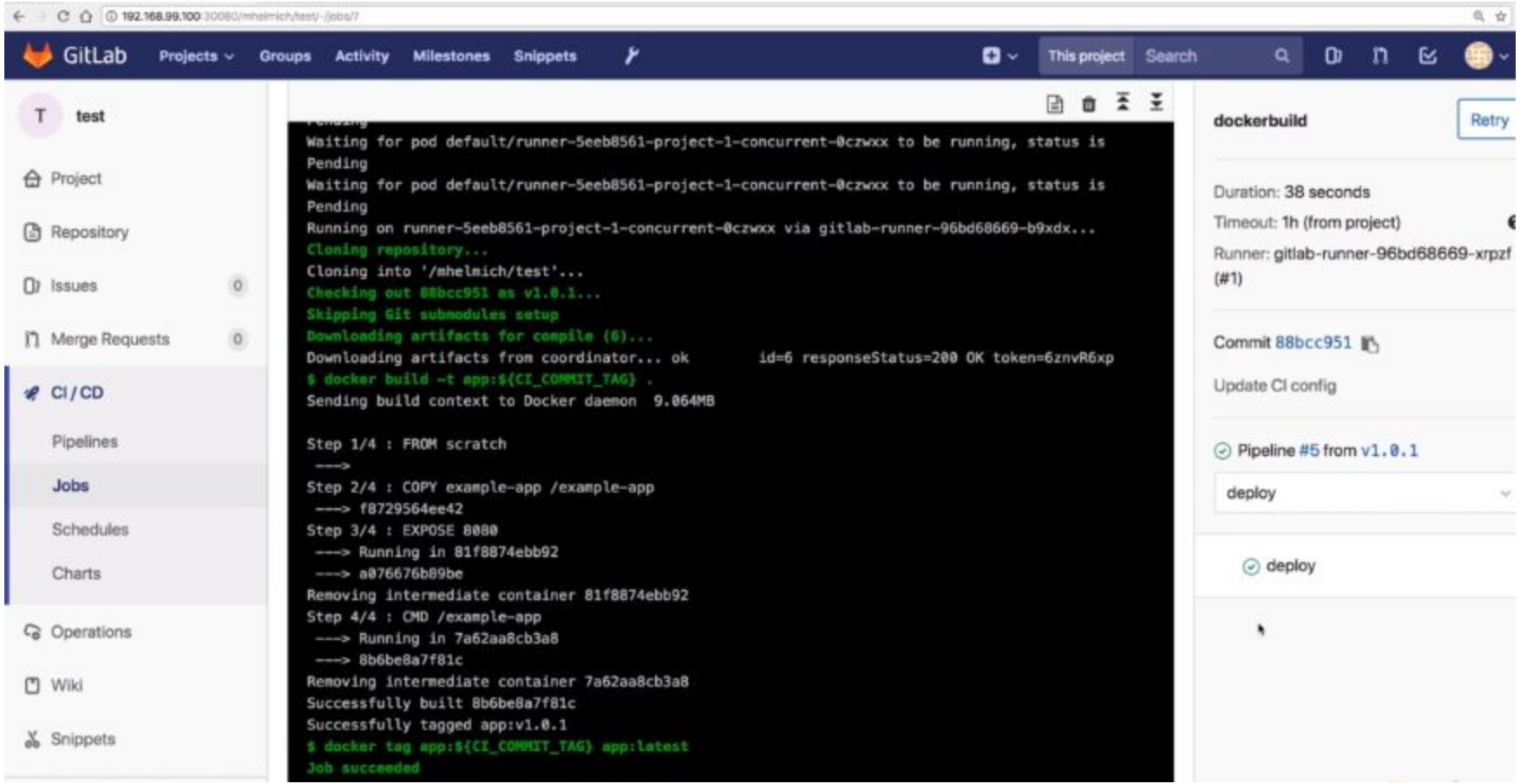

GitLab
Projects
Groups
Activity
Milestones
Snippets
This project
Search
T
test
Project
Repository
Files
Commits
Branches
Tags
Contributors
Graph
Compare
Charts
Issues
0
Merge Requests
0
CI / CD
Martin Helmich › test › Repository
master
test / Dockerfile
Find file
Blame
History
Permalink
Add dockerfile and fix pipeline
Martin Helmich authored 1 day ago
a04113d1
Dockerfile 78 Bytes
Edit
Web IDE
Replace
Delete
1 FROM scratch
2
3 COPY example-app /example-app
4
5 EXPOSE 8080
6
7 CMD ["/example-app"]

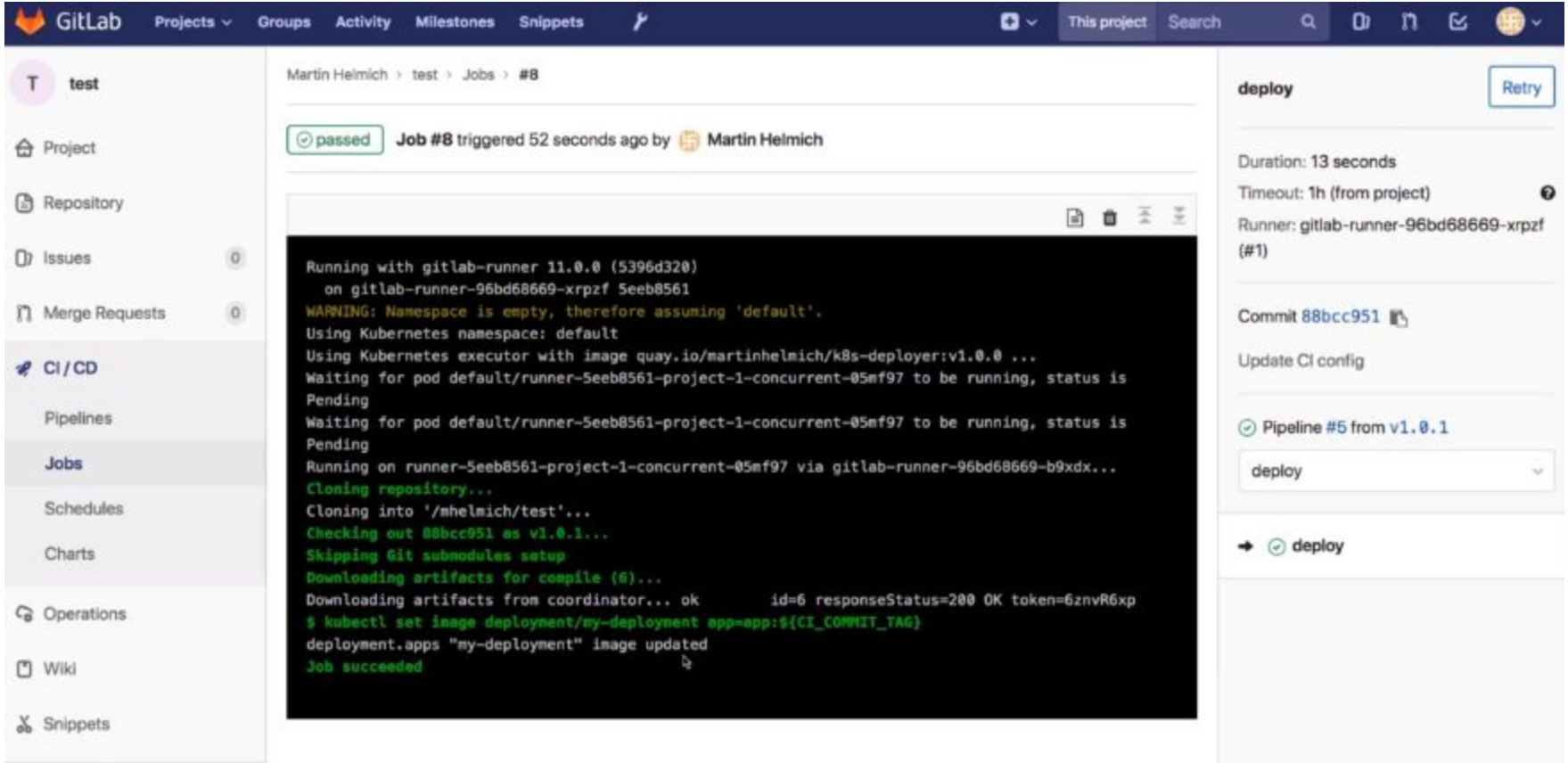

GitLab
Projects
Groups
Activity
Milestones
Snippets
This project
Search
T
test
Project
Repository
Files
Commits
Branches
Tags
Contributors
Graph
Compare
Charts
Issues
0
Merge Requests
0
CI / CD
Collapse sidebar
.gitlab-ci.yml 768 Bytes
Edit
Web IDE
stages:
- build
- package
- deploy
compile:
stage: build
image: golang:1.10.3
script:
- go build -o example-app
artifacts:
paths:
- example-app
expire_in: 1 week
dockerbuild:
stage: package
image: docker:18.05.0-ce
script:
- docker build -t app:${CI_COMMIT_TAG} .
- docker tag app:${CI_COMMIT_TAG} app:latest
Disabled as long as we're running on minikube
Since it's all just one VM, any built image will be available immediately, anyway
#- docker push app:${CI_COMMIT_TAG}
#- docker push app:latest
only:
- tags

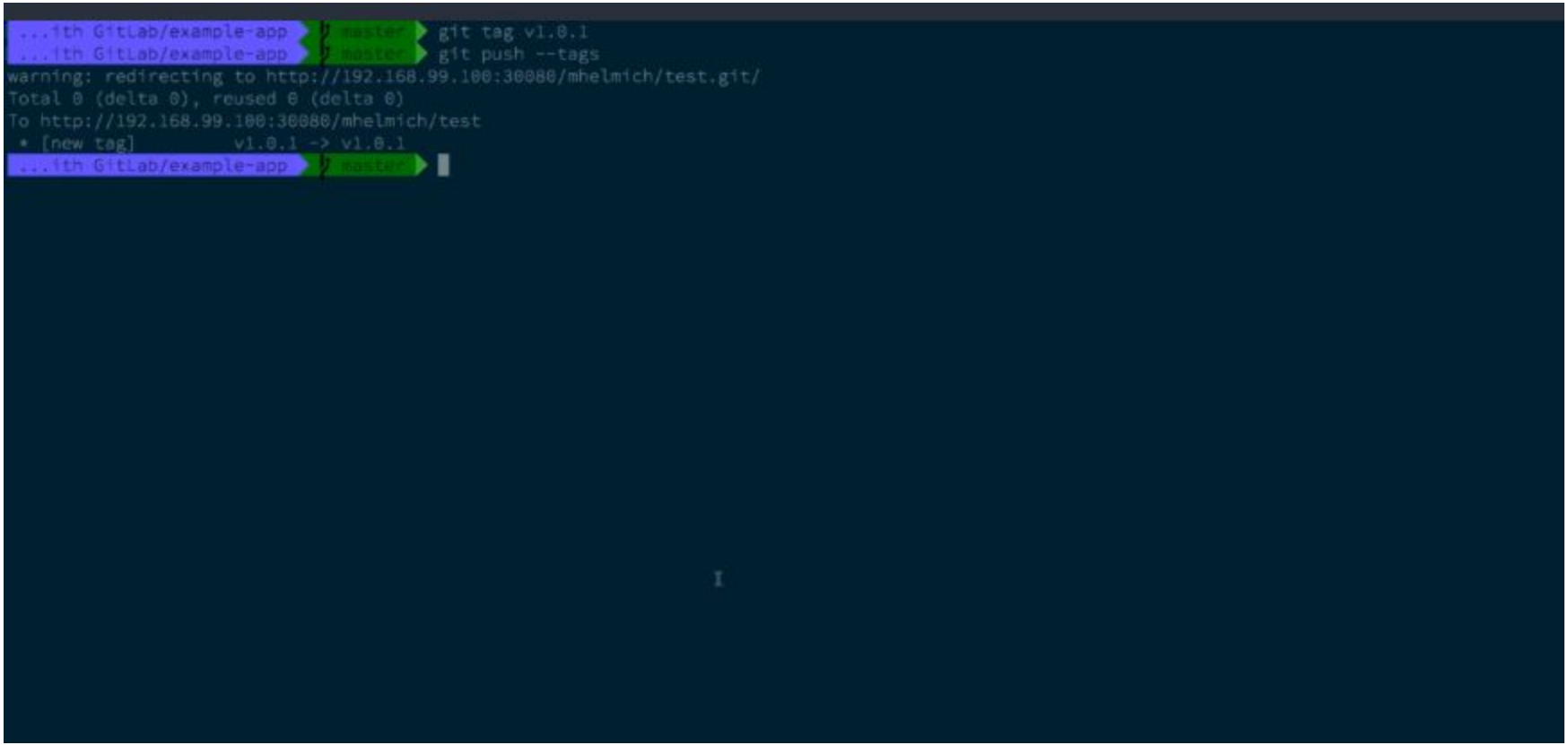

...th GitLab/example-app
master

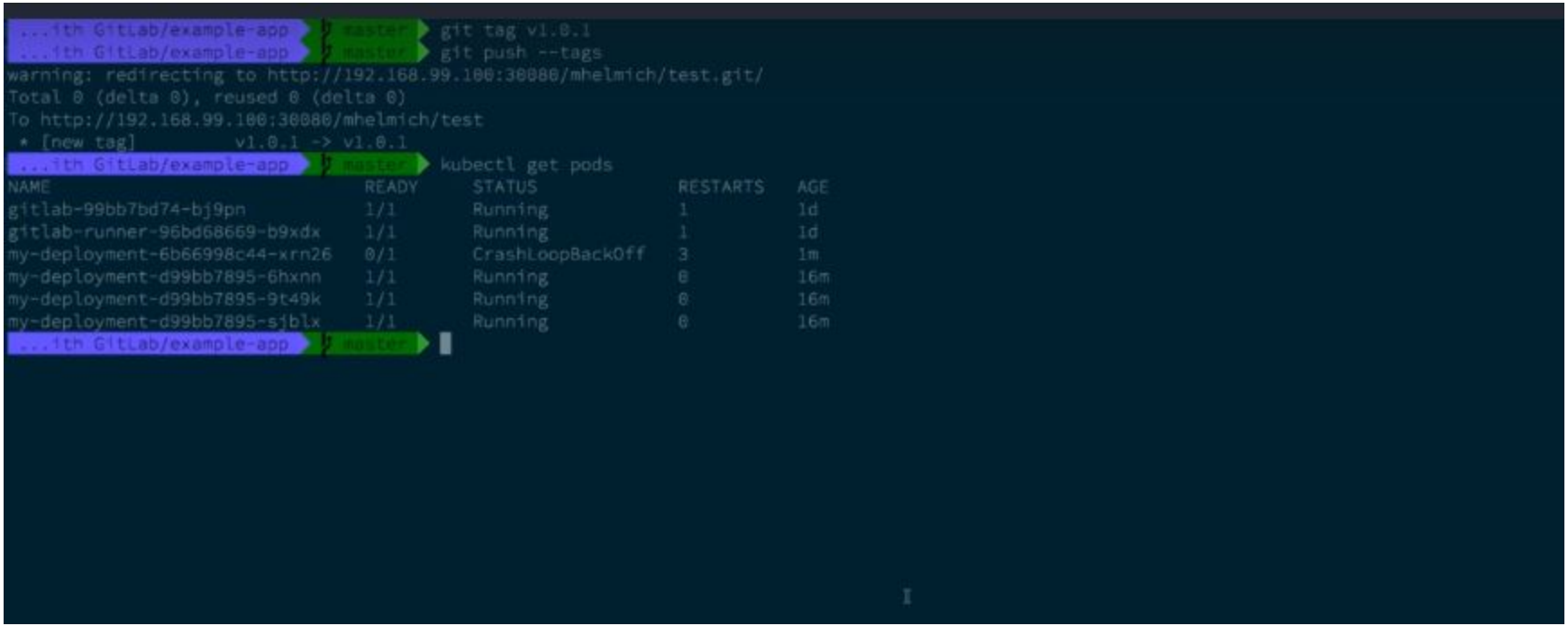

GitLab
Projects
Groups
Activity
Milestones
Snippets
This project
Search
T
test
Project
Repository
Issues 0
Merge Requests 0
CI / CD
Pipelines
Jobs
Schedules
Charts
Operations
Wiki
Snippets
Martin Heimich › test › Pipelines
All 5
Pending 0
Running 1
Finished 4
Branches
Tags
Run Pipeline
Clear Runner Caches
CI Lint
Status
Pipeline
Commit
Stages
running
#5 by
latest
v1.0.1 88bcc951
Update CI config
passed
#4 by
latest
master 88bcc951
Update CI config
00:00:11
11 minutes ago
passed
#3 by
master a04113d1
Add dockerfile and fix pi...
00:00:09
1 day ago
passed
#2 by
master 7c58202f
Fix CI
00:00:42
1 day ago
failed
#1 by
yaml invalid
error
master 95a387f5
Initial commit
1 day ago


GitLab
Projects
Groups
Activity
Milestones
Snippets
This project
Search
T
test
Project
Repository
Issues
0
Merge Requests
0
CI / CD
Pipelines
Jobs
Schedules
Charts
Operations
Wiki
Snippets
Martin Helmich › test › Pipelines › #5
passed
Pipeline #5 triggered 1 minute ago by
Martin Helmich
Update CI config
3 jobs from v1.0.1 in 1 minute 2 seconds
88bcc951
Pipeline
Jobs 3
Build
Package
Deploy
compile
dockerbuild
deploy


192.168.99.100:30080/mhelmich/test/-/jobs/7
GitLab
Projects
Groups
Activity
Milestones
Snippets
This project
Search
T
test
Project
Repository
Issues 0
Merge Requests 0
CI / CD
Pipelines
Jobs
Schedules
Charts
Operations
Wiki
Snippets
Waiting for pod default/runner-5eeb8561-project-1-concurrent-0czwxx to be running, status is Pending
Waiting for pod default/runner-5eeb8561-project-1-concurrent-0czwxx to be running, status is Pending
Running on runner-5eeb8561-project-1-concurrent-0czwxx via gitlab-runner-96bd68669-b9xdx...
Cloning repository...
Cloning into '/mhelmich/test'...
Checking out 88bcc951 as v1.0.1...
Skipping Git submodules setup
Downloading artifacts for compile (6)...
Downloading artifacts from coordinator... ok id=6 responseStatus=200 OK token=6znvR6xp
$ docker build -t app:${CI_COMMIT_TAG} .
Sending build context to Docker daemon 9.064MB
Step 1/4 : FROM scratch
--->
Step 2/4 : COPY example-app /example-app
---> f8729564ee42
Step 3/4 : EXPOSE 8080
---> Running in 81f8874ebb92
---> a076676b89be
Removing intermediate container 81f8874ebb92
Step 4/4 : CMD /example-app
---> Running in 7a62aa8cb3a8
---> 8b6be8a7f81c
Removing intermediate container 7a62aa8cb3a8
Successfully built 8b6be8a7f81c
Successfully tagged app:v1.0.1
$ docker tag app:${CI_COMMIT_TAG} app:latest
Job succeeded
dockerbuild
Retry
Duration: 38 seconds
Timeout: 1h (from project)
Runner: gitlab-runner-96bd68669-xrpzf (#1)
Commit 88bcc951
Update CI config
Pipeline #5 from v1.0.1
deploy
deploy

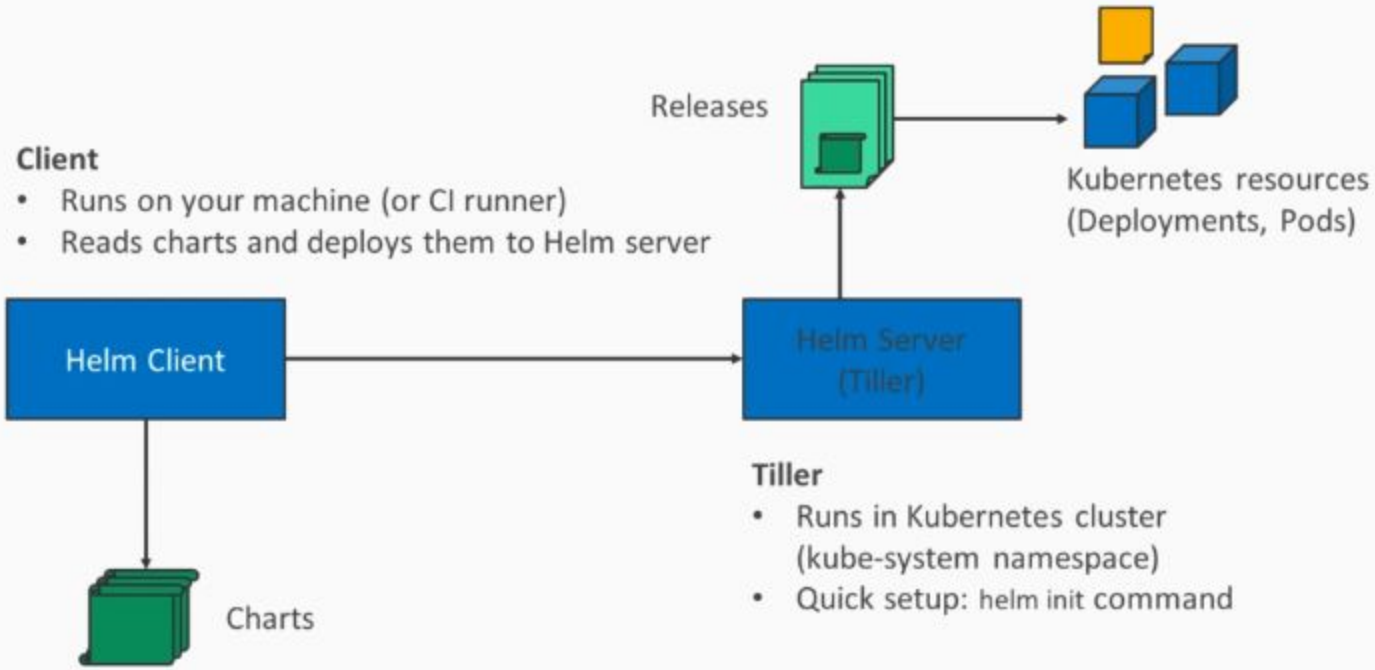

GitLab
Projects
Groups
Activity
Milestones
Snippets
This project
Search
T
test
Project
Repository
Issues
0
Merge Requests
0
CI / CD
Pipelines
Jobs
Schedules
Charts
Operations
Wiki
Snippets
Martin Helmich › test › Jobs › #8
passed
Job #8 triggered 52 seconds ago by Martin Helmich
Running with gitlab-runner 11.0.0 (5396d320)
on gitlab-runner-96bd68669-xrpzf 5eeb8561
WARNING: Namespace is empty, therefore assuming 'default'.
Using Kubernetes namespace: default
Using Kubernetes executor with image quay.io/martinhelmich/k8s-deployer:v1.0.0 ...
Waiting for pod default/runner-5eeb8561-project-1-concurrent-05mf97 to be running, status is Pending
Waiting for pod default/runner-5eeb8561-project-1-concurrent-05mf97 to be running, status is Pending
Running on runner-5eeb8561-project-1-concurrent-05mf97 via gitlab-runner-96bd68669-b9xdx...
Cloning repository...
Cloning into '/mhelmich/test'...
Checking out 88bcc951 as v1.0.1...
Skipping Git submodules setup
Downloading artifacts for compile (6)...
Downloading artifacts from coordinator... ok id=6 responseStatus=200 OK token=6znvR6xp
$ kubectl set image deployment/my-deployment app=app:${CI_COMMIT_TAG}
deployment.apps "my-deployment" image updated
Job succeeded
deploy
Retry
Duration: 13 seconds
Timeout: 1h (from project)
Runner: gitlab-runner-96bd68669-xrpzf (#1)
Commit 88bcc951
Update CI config
Pipeline #5 from v1.0.1
deploy
deploy


...ith GitLab/example-app master git tag v1.0.1
...ith GitLab/example-app master git push --tags
warning: redirecting to http://192.168.99.100:30080/mhelmich/test.git/
Total 0 (delta 0), reused 0 (delta 0)
To http://192.168.99.100:30080/mhelmich/test
* [new tag] v1.0.1 -> v1.0.1
...ith GitLab/example-app master


...ith GitLab/example-app master git tag v1.0.1
...ith GitLab/example-app master git push --tags
warning: redirecting to http://192.168.99.100:30080/mhelmich/test.git/
Total 0 (delta 0), reused 0 (delta 0)
To http://192.168.99.100:30080/mhelmich/test
* [new tag] v1.0.1 -> v1.0.1
...ith GitLab/example-app master kubectl get pods
NAME READY STATUS RESTARTS AGE
gitlab-99bb7bd74-bj9pn 1/1 Running 1 1d
gitlab-runner-96bd68669-b9xdx 1/1 Running 1 1d
my-deployment-6b66998c44-xrn26 0/1 CrashLoopBackOff 3 1m
my-deployment-d99bb7895-6hxnn 1/1 Running 0 16m
my-deployment-d99bb7895-9t49k 1/1 Running 0 16m
my-deployment-d99bb7895-sjblx 1/1 Running 0 16m
...ith GitLab/example-app master

Setting up Helm

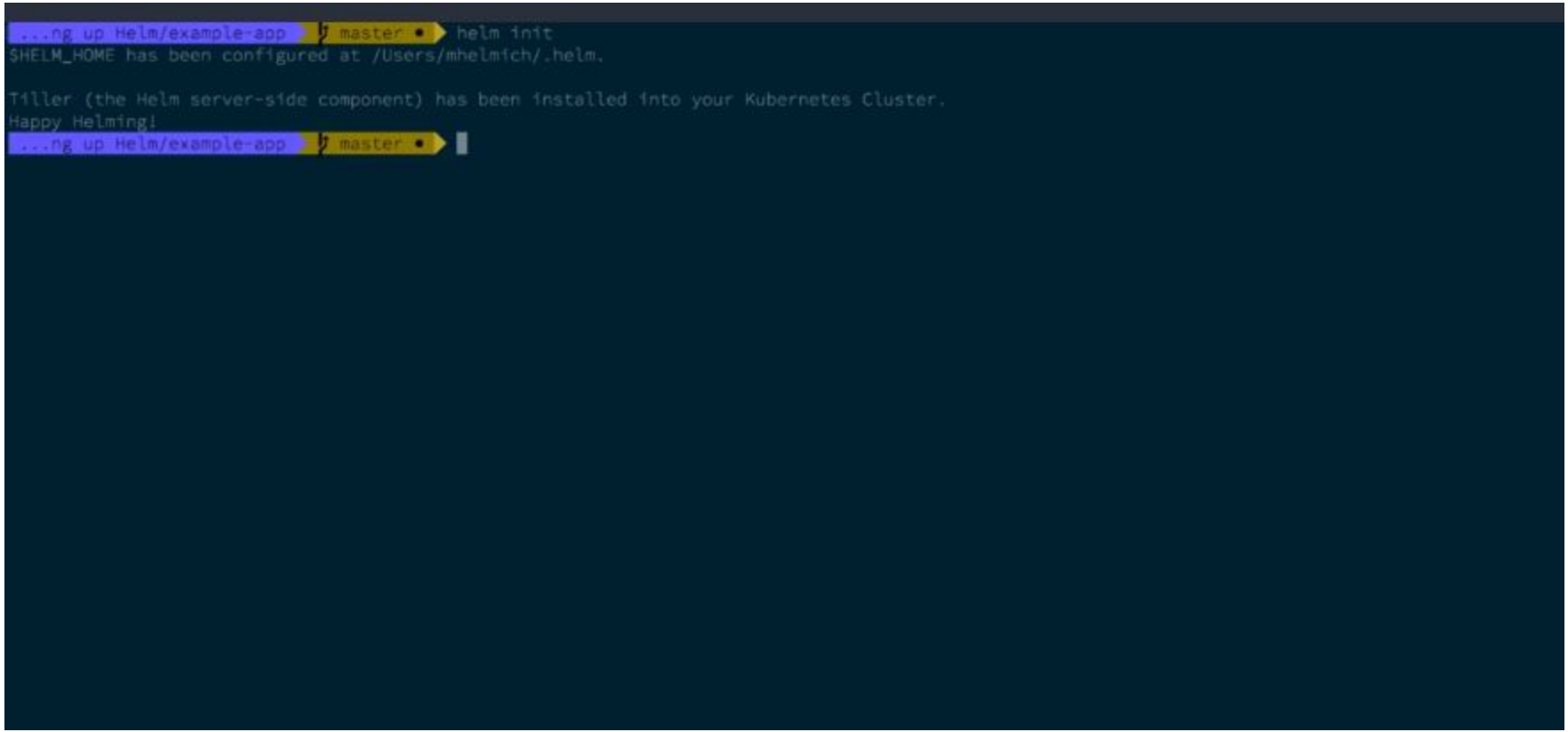
Helm Architecture



Helm Installation and Setup

1. Download and install Helm to your local machine
   https://docs.helm.sh/using_helm/#installing-helm

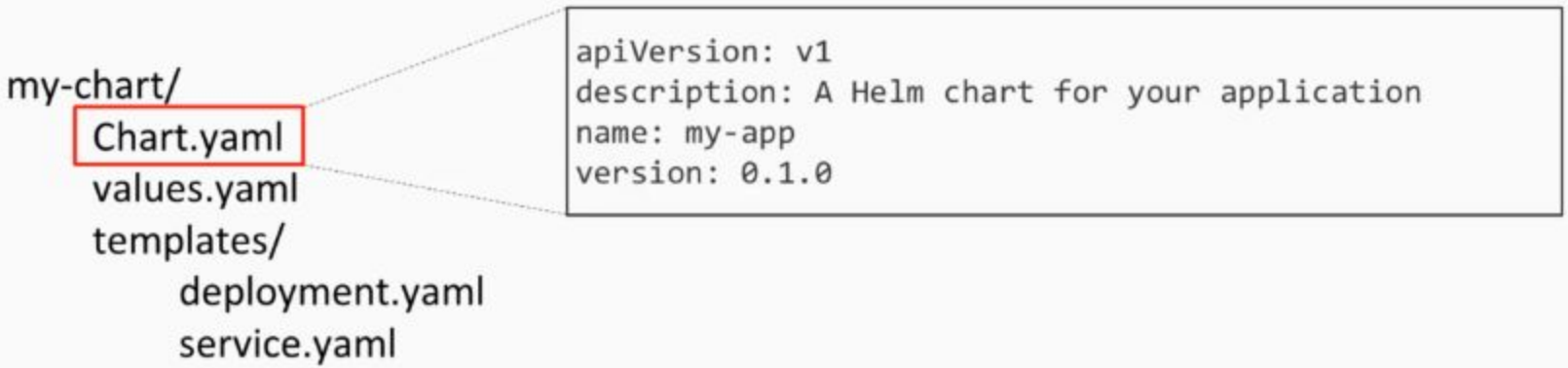
2. Setup local Helm and Tiller server using
   $ helm init


...ng up Helm/example-app master • helm init
$HELM_HOME has been configured at /Users/mhelmich/.helm.
Tiller (the Helm server-side component) has been installed into your Kubernetes Cluster.
Happy Helming!
...ng up Helm/example-app master •

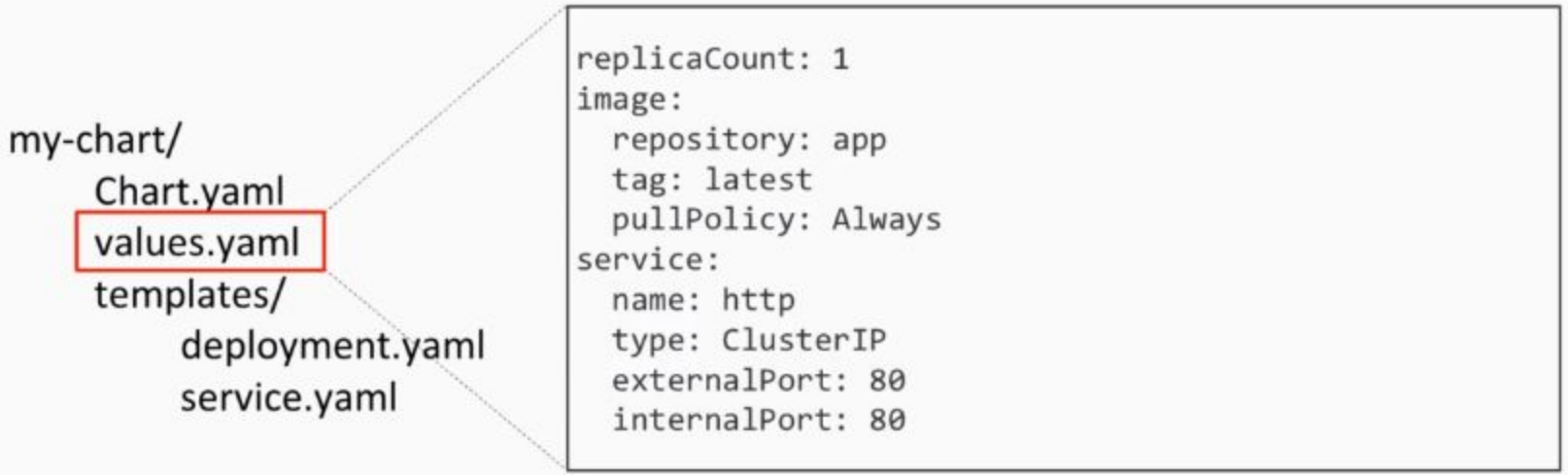
Helm Chart Structure

```
my-chart/
        Chart.yaml
        values.yaml
        templates/
                deployment.yaml
                service.yaml
```

Helm Chart Structure



Helm Chart Structure



Helm Chart Structure

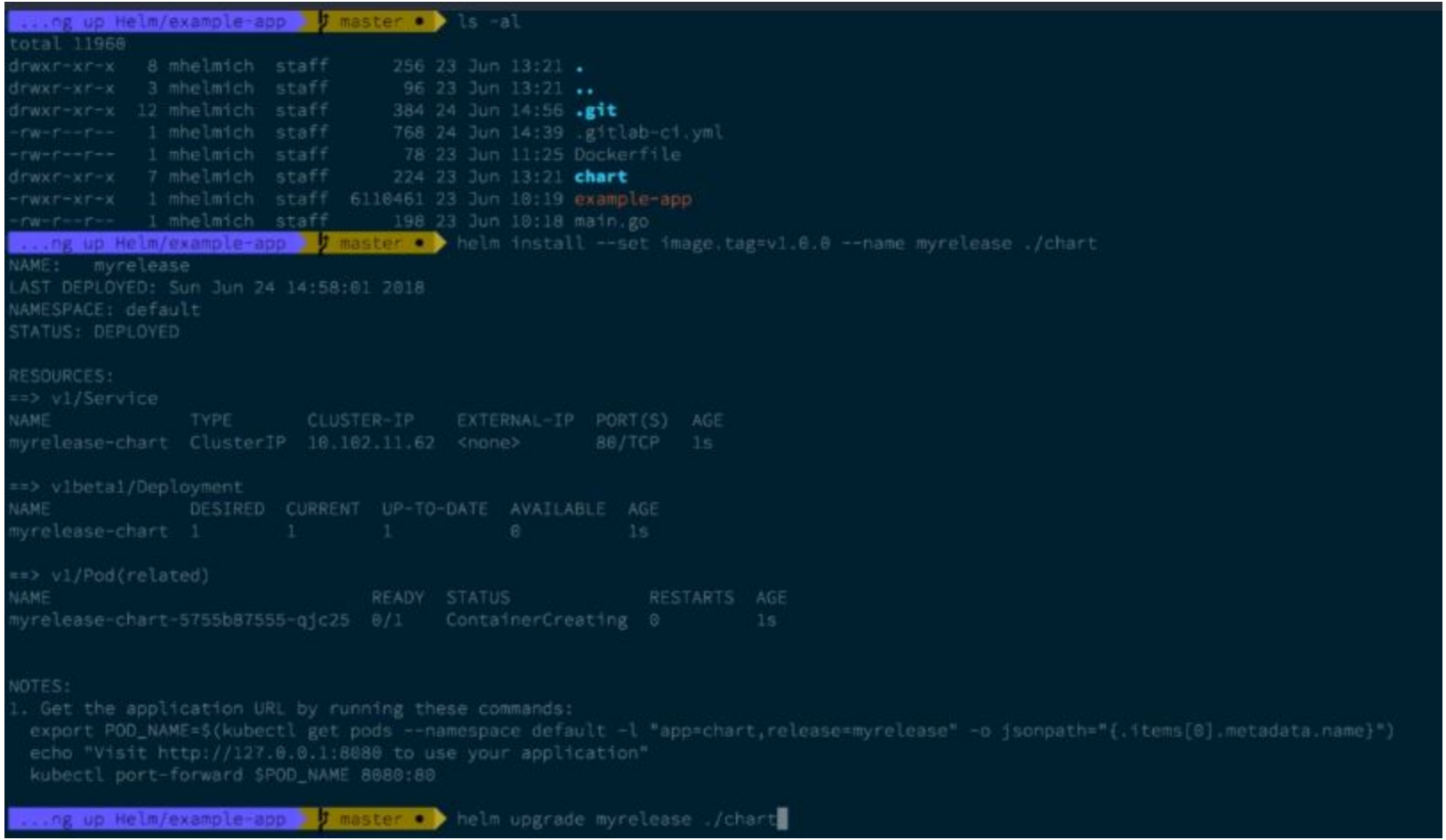
my-chart/
- Chart.yaml
- values.yaml
- templates/
  - deployment.yaml
  - service.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: {{ template "chart.fullname" . }}
  labels:
    app: {{ template "chart.name" . }}
    chart: {{ .Chart.Name }}-{{ .Chart.Version | replace "+" "_" }}
    release: {{ .Release.Name }}
    heritage: {{ .Release.Service }}
spec:
  replicas: {{ .Values.replicaCount }}
  template:
    metadata:
      labels:
        app: {{ template "chart.name" }}
        release: {{ .Release.Name }}
    spec:
      containers:
      - name: {{ .Chart.Name }}
        image: {{ .Values.image.repository }}:{{ .Values.image.tag }}
        imagePullPolicy: {{ .Values.image.pullPolicy }}
        ports:
        - protocol: TCP
          name: http
          containerPort: {{ .Values.service.internalPort }}
```

Helm Chart Structure

```
my-chart/
    Chart.yaml
    values.yaml
    templates/
        deployment.yaml
        service.yaml
```

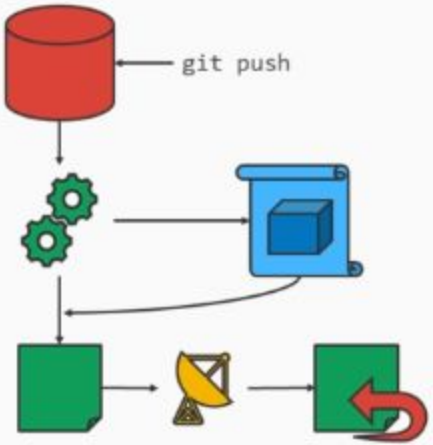
Create these files for a new project quicky using the helm create <name> command


...ng up Helm/example-app master • ls -al
total 11960
drwxr-xr-x 8 mhelmich staff 256 23 Jun 13:21 .
drwxr-xr-x 3 mhelmich staff 96 23 Jun 13:21 ..
drwxr-xr-x 12 mhelmich staff 384 24 Jun 14:56 .git
-rw-r--r-- 1 mhelmich staff 768 24 Jun 14:39 .gitlab-ci.yml
-rw-r--r-- 1 mhelmich staff 78 23 Jun 11:25 Dockerfile
drwxr-xr-x 7 mhelmich staff 224 23 Jun 13:21 chart
-rwxr-xr-x 1 mhelmich staff 6110461 23 Jun 10:19 example-app
-rw-r--r-- 1 mhelmich staff 198 23 Jun 10:18 main.go
...ng up Helm/example-app master • helm install --set image.tag=v1.0.0 --name myrelease ./chart
NAME: myrelease
LAST DEPLOYED: Sun Jun 24 14:58:01 2018
NAMESPACE: default
STATUS: DEPLOYED
RESOURCES:
==> v1/Service
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
myrelease-chart ClusterIP 10.102.11.62 <none> 80/TCP 1s
==> v1beta1/Deployment
NAME DESIRED CURRENT UP-TO-DATE AVAILABLE AGE
myrelease-chart 1 1 1 0 1s
==> v1/Pod(related)
NAME READY STATUS RESTARTS AGE
myrelease-chart-5755b87555-qjc25 0/1 ContainerCreating 0 1s
NOTES:
1. Get the application URL by running these commands:
export POD_NAME=$(kubectl get pods --namespace default -l "app=chart,release=myrelease" -o jsonpath="{.items[0].metadata.name}")
echo "Visit http://127.0.0.1:8080 to use your application"
kubectl port-forward $POD_NAME 8080:80
...ng up Helm/example-app master • helm upgrade myrelease ./chart

```
==> v1/Pod(related)
NAME                              READY  STATUS             RESTARTS  AGE
myrelease-chart-5755b87555-qjc25  0/1    ContainerCreating  0         1s


NOTES:
1. Get the application URL by running these commands:
  export POD_NAME=$(kubectl get pods --namespace default -l "app=chart,release=myrelease" -o jsonpath="{.items[0].metadata.name}")
  echo "Visit http://127.0.0.1:8080 to use your application"
  kubectl port-forward $POD_NAME 8080:80

...ng up Helm/example-app  master ●  helm upgrade --set image.tag=v1.0.1 myrelease ./chart
Release "myrelease" has been upgraded. Happy Helming!
LAST DEPLOYED: Sun Jun 24 14:59:25 2018
NAMESPACE: default
STATUS: DEPLOYED

RESOURCES:
==> v1/Service
NAME             TYPE       CLUSTER-IP    EXTERNAL-IP  PORT(S)  AGE
myrelease-chart  ClusterIP  10.102.11.62  <none>       80/TCP   1m

==> v1beta1/Deployment
NAME             DESIRED  CURRENT  UP-TO-DATE  AVAILABLE  AGE
myrelease-chart  1        1        1           0          1m

==> v1/Pod(related)
NAME                              READY  STATUS             RESTARTS  AGE
myrelease-chart-54f6975cbd-p8bhf  0/1    ContainerCreating  0         1s
myrelease-chart-5755b87555-qjc25  0/1    Terminating        0         1m


NOTES:
1. Get the application URL by running these commands:
  export POD_NAME=$(kubectl get pods --namespace default -l "app=chart,release=myrelease" -o jsonpath="{.items[0].metadata.name}")
  echo "Visit http://127.0.0.1:8080 to use your application"
  kubectl port-forward $POD_NAME 8080:80

...ng up Helm/example-app  master ●  helm upgrade --install --set image.tag=v1.0.1 myrelease ./chart
```

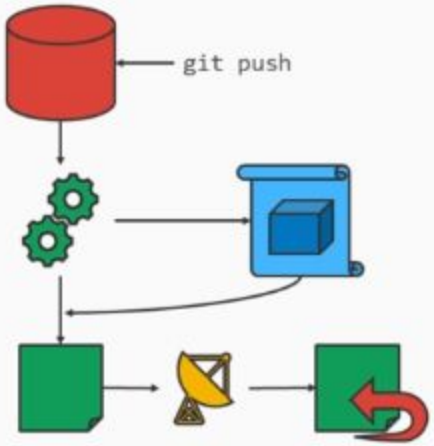
Using Helm in CI

Previously

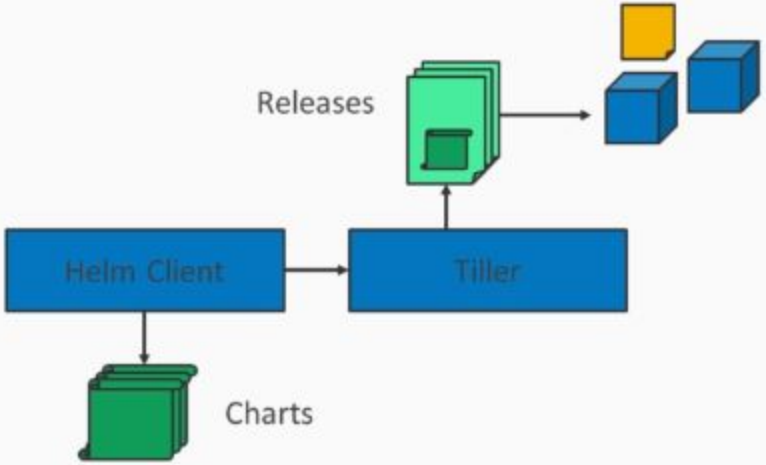


Continuous Delivery
with GitLab

Previously



Continuous Delivery
with GitLab



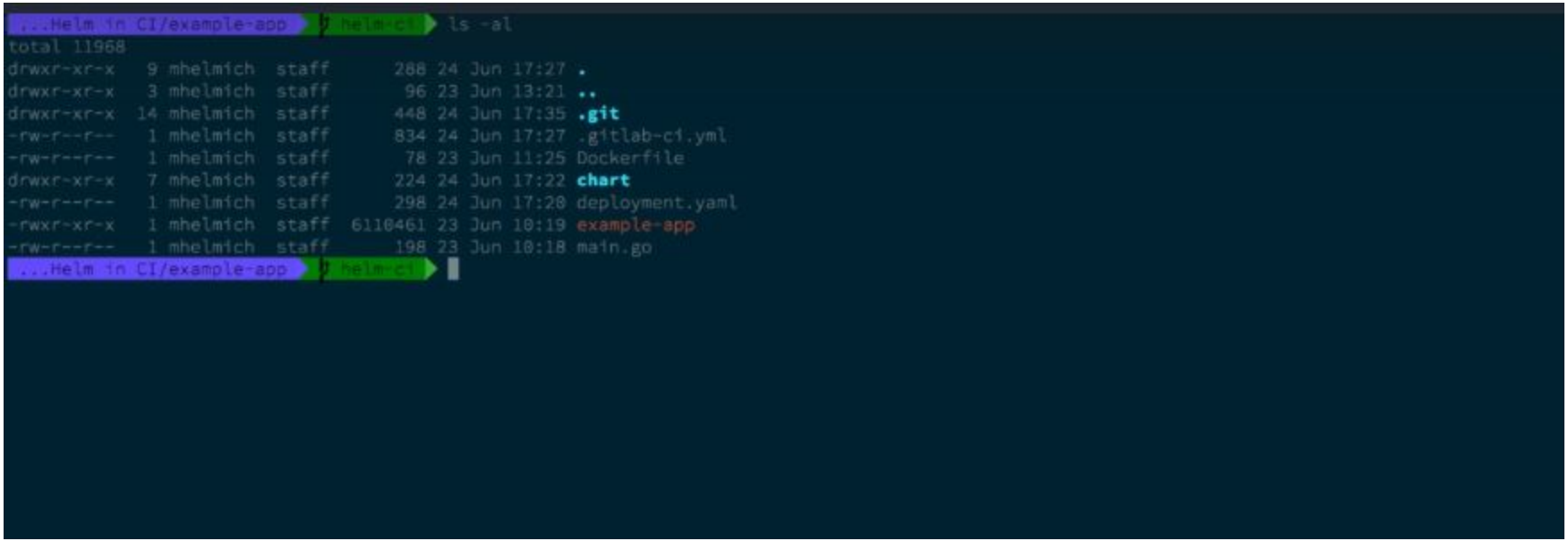
Kubernetes Package Management
with Helm

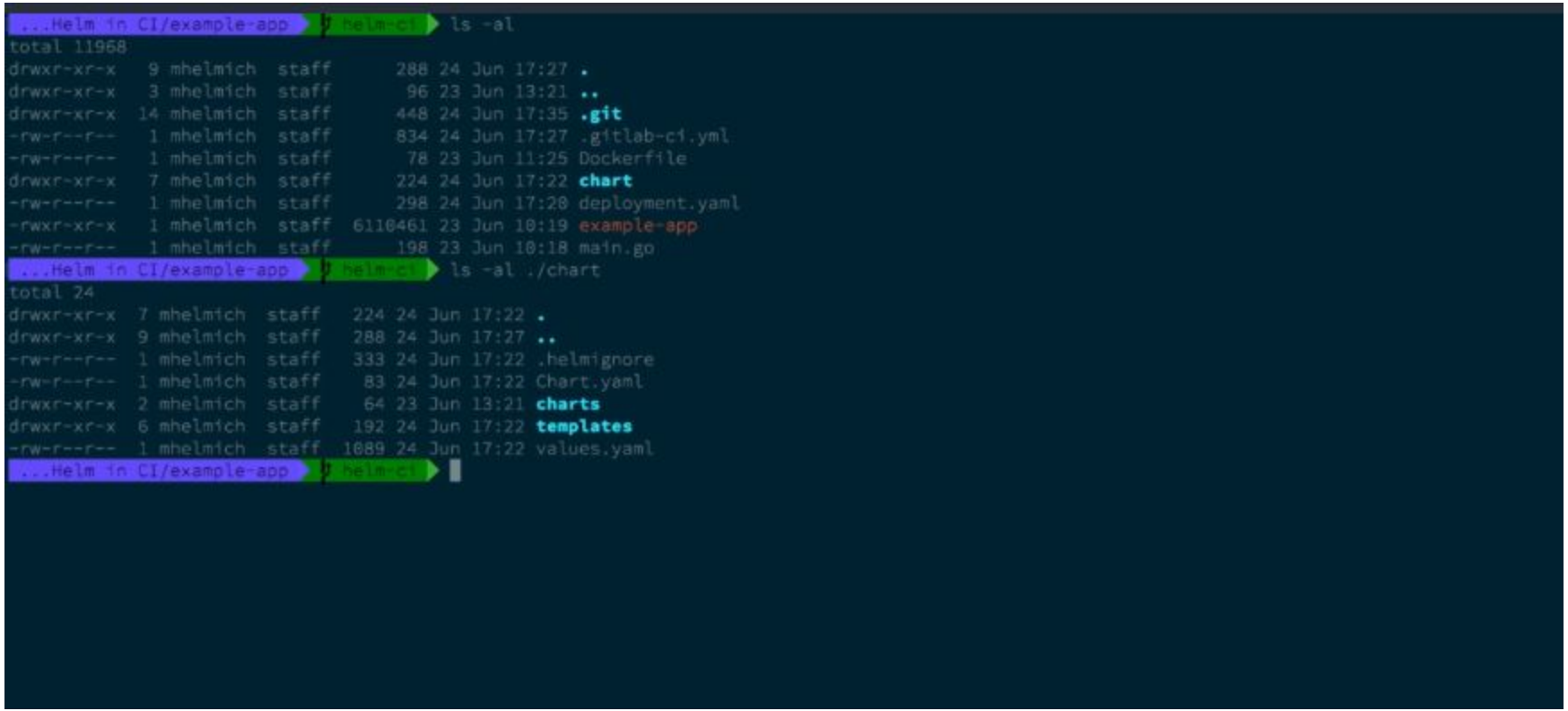
Using Helm in GitLab CI

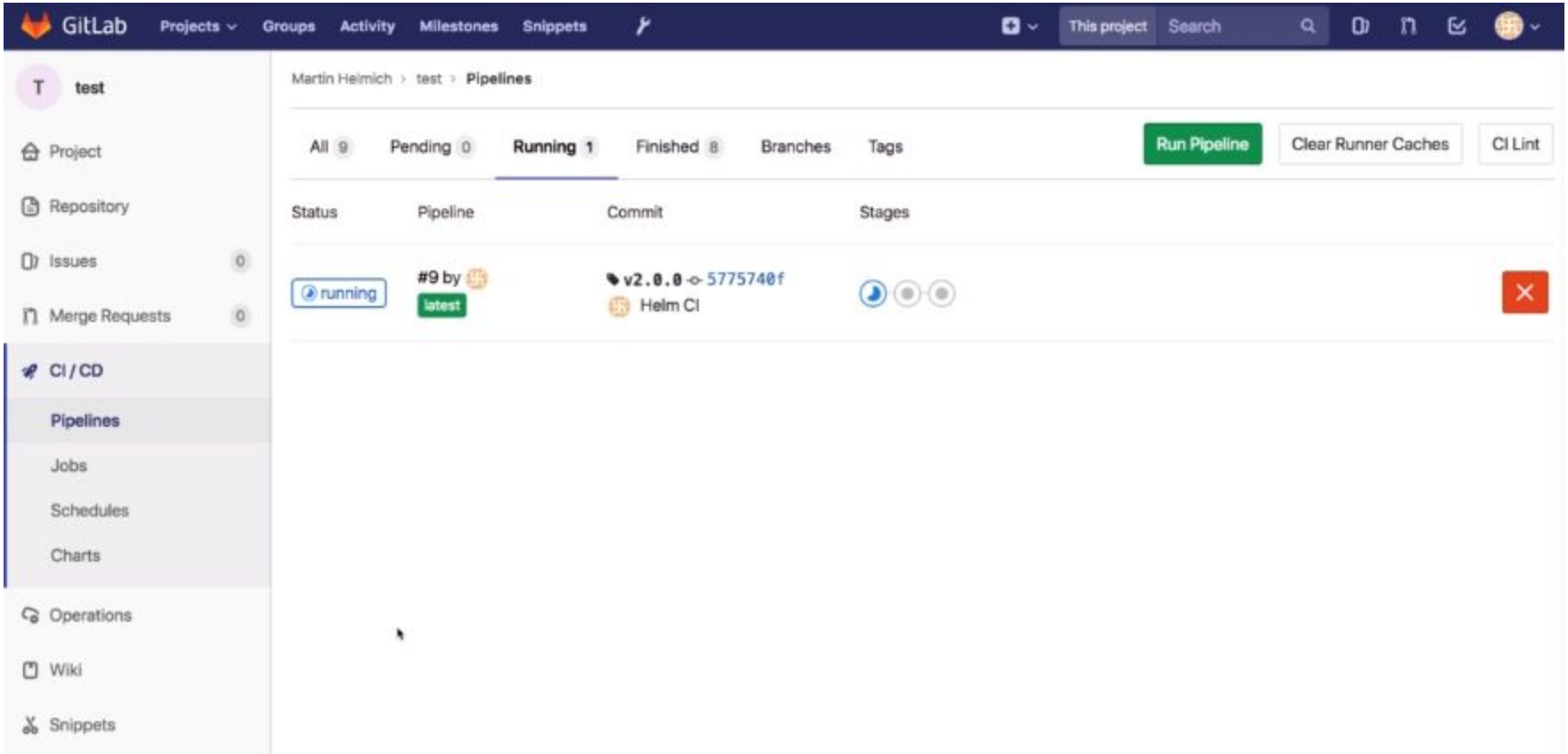
```
deploy:
  stage: deploy
  image: quay.io/martinhelmich/k8s-deployer:v1.0.0
  environment: development
  script:
    - >
      kubectl set image deployment/your-app
      app=app:${CI_COMMIT_TAG}
    - kubectl get pods
  only:
    - tags
```

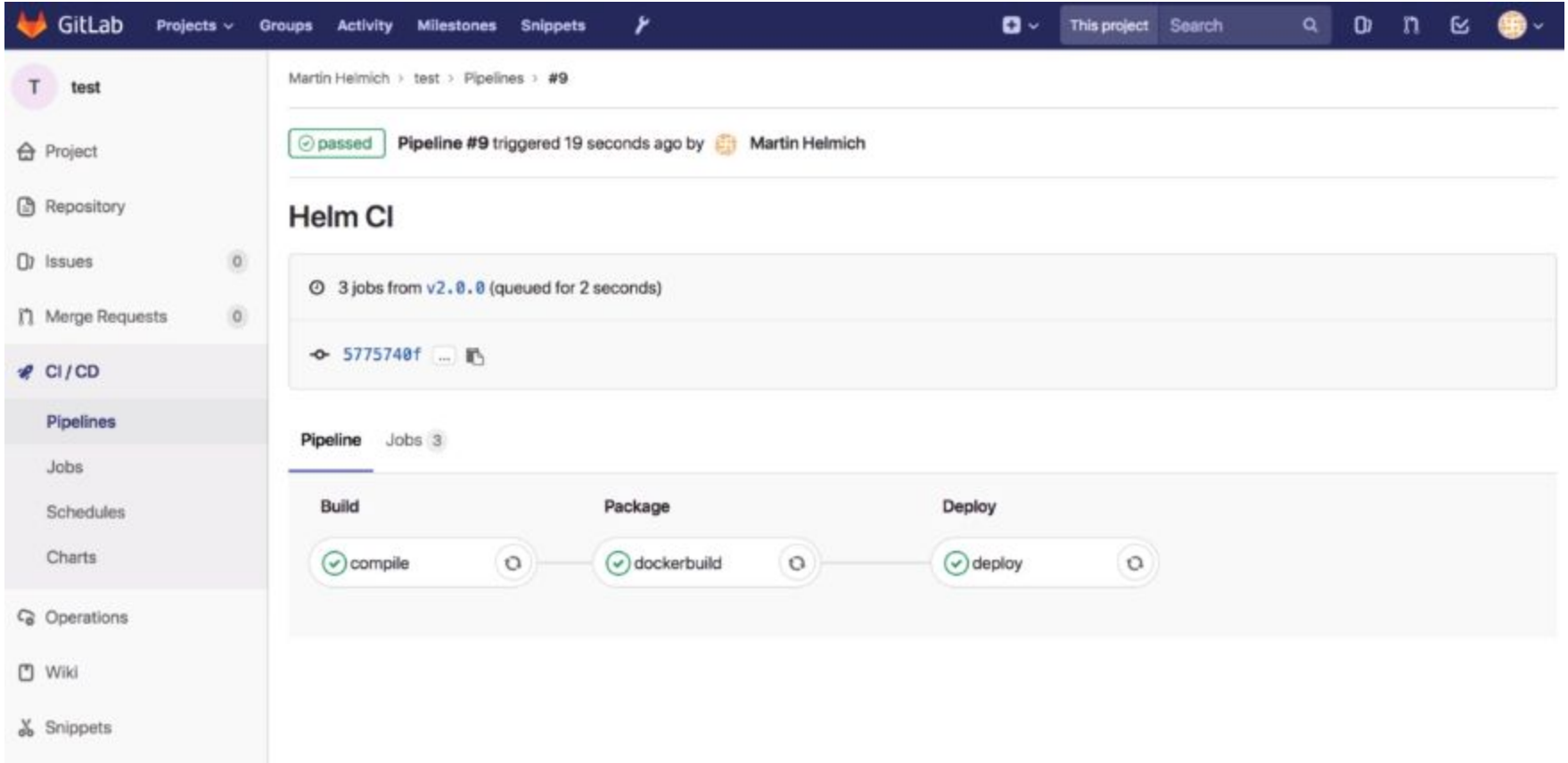
Using Helm in GitLab CI

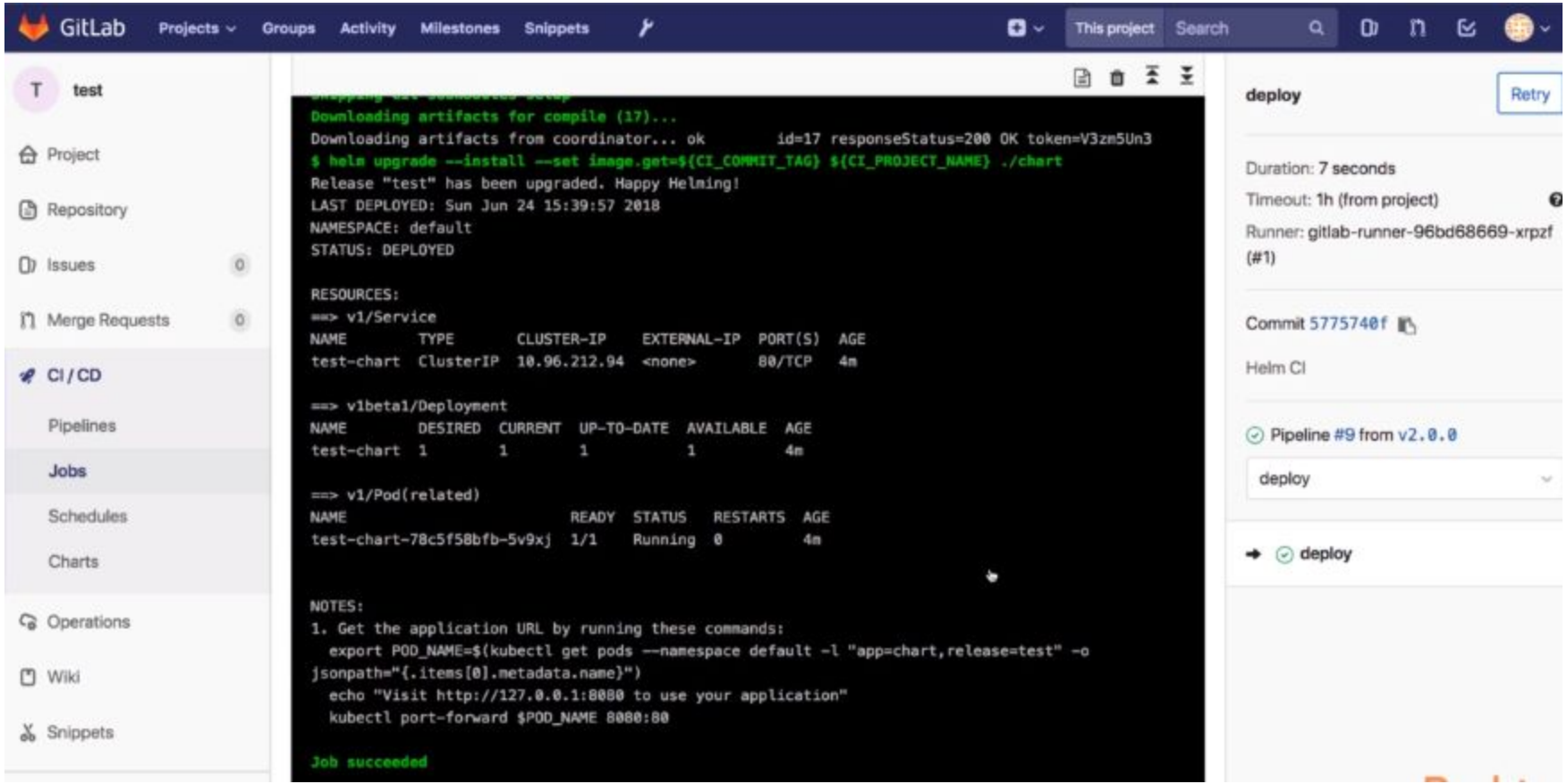
```
 deploy:
   stage: deploy
   image: quay.io/martinhelmich/k8s-deployer:v1.0.0
   environment: development
   script:
-    - >
-      kubectl set image deployment/your-app
-      app=app:${CI_COMMIT_TAG}
+    - >
+      helm upgrade --install
+        --set image.tag=${CI_COMMIT_TAG}
+        ${CI_PROJECT_NAME} ./chart
     - kubectl get pods
   only:
     - tags
```


.../Helm in CI/example-app
helm-ci
ls -al
total 11968
drwxr-xr-x 9 mhelmich staff 288 24 Jun 17:27 .
drwxr-xr-x 3 mhelmich staff 96 23 Jun 13:21 ..
drwxr-xr-x 14 mhelmich staff 448 24 Jun 17:35 .git
-rw-r--r-- 1 mhelmich staff 834 24 Jun 17:27 .gitlab-ci.yml
-rw-r--r-- 1 mhelmich staff 78 23 Jun 11:25 Dockerfile
drwxr-xr-x 7 mhelmich staff 224 24 Jun 17:22 chart
-rw-r--r-- 1 mhelmich staff 298 24 Jun 17:20 deployment.yaml
-rwxr-xr-x 1 mhelmich staff 6110461 23 Jun 10:19 example-app
-rw-r--r-- 1 mhelmich staff 198 23 Jun 10:18 main.go
.../Helm in CI/example-app
helm-ci


...Helm in CI/example-app
helm-ci
ls -al
total 11968
drwxr-xr-x 9 mhelmich staff 288 24 Jun 17:27 .
drwxr-xr-x 3 mhelmich staff 96 23 Jun 13:21 ..
drwxr-xr-x 14 mhelmich staff 448 24 Jun 17:35 .git
-rw-r--r-- 1 mhelmich staff 834 24 Jun 17:27 .gitlab-ci.yml
-rw-r--r-- 1 mhelmich staff 78 23 Jun 11:25 Dockerfile
drwxr-xr-x 7 mhelmich staff 224 24 Jun 17:22 chart
-rw-r--r-- 1 mhelmich staff 298 24 Jun 17:20 deployment.yaml
-rwxr-xr-x 1 mhelmich staff 6110461 23 Jun 10:19 example-app
-rw-r--r-- 1 mhelmich staff 198 23 Jun 10:18 main.go
...Helm in CI/example-app
helm-ci
ls -al ./chart
total 24
drwxr-xr-x 7 mhelmich staff 224 24 Jun 17:22 .
drwxr-xr-x 9 mhelmich staff 288 24 Jun 17:27 ..
-rw-r--r-- 1 mhelmich staff 333 24 Jun 17:22 .helmignore
-rw-r--r-- 1 mhelmich staff 83 24 Jun 17:22 Chart.yaml
drwxr-xr-x 2 mhelmich staff 64 23 Jun 13:21 charts
drwxr-xr-x 6 mhelmich staff 192 24 Jun 17:22 templates
-rw-r--r-- 1 mhelmich staff 1089 24 Jun 17:22 values.yaml
...Helm in CI/example-app
helm-ci


GitLab
Projects
Groups
Activity
Milestones
Snippets
This project
Search
T
test
Project
Repository
Issues
0
Merge Requests
0
CI / CD
Pipelines
Jobs
Schedules
Charts
Operations
Wiki
Snippets
Martin Helmich › test › Pipelines
All 9
Pending 0
Running 1
Finished 8
Branches
Tags
Run Pipeline
Clear Runner Caches
CI Lint
Status
Pipeline
Commit
Stages
running
#9 by
latest
v2.0.0 5775740f
Helm CI


GitLab
Projects
Groups
Activity
Milestones
Snippets
This project
Search
T
test
Project
Repository
Issues
0
Merge Requests
0
CI / CD
Pipelines
Jobs
Schedules
Charts
Operations
Wiki
Snippets
Martin Helmich › test › Pipelines › #9
passed
Pipeline #9 triggered 19 seconds ago by
Martin Helmich
Helm CI
3 jobs from v2.0.0 (queued for 2 seconds)
5775740f
Pipeline
Jobs 3
Build
Package
Deploy
compile
dockerbuild
deploy


GitLab
Projects
Groups
Activity
Milestones
Snippets
This project
Search
T
test
Project
Repository
Issues
0
Merge Requests
0
CI / CD
Pipelines
Jobs
Schedules
Charts
Operations
Wiki
Snippets
Downloading artifacts for compile (17)...
Downloading artifacts from coordinator... ok id=17 responseStatus=200 OK token=V3zm5Un3
$ helm upgrade --install --set image.get=${CI_COMMIT_TAG} ${CI_PROJECT_NAME} ./chart
Release "test" has been upgraded. Happy Helming!
LAST DEPLOYED: Sun Jun 24 15:39:57 2018
NAMESPACE: default
STATUS: DEPLOYED
RESOURCES:
==> v1/Service
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
test-chart ClusterIP 10.96.212.94 <none> 80/TCP 4m
==> v1beta1/Deployment
NAME DESIRED CURRENT UP-TO-DATE AVAILABLE AGE
test-chart 1 1 1 1 4m
==> v1/Pod(related)
NAME READY STATUS RESTARTS AGE
test-chart-78c5f58bfb-5v9xj 1/1 Running 0 4m
NOTES:
1. Get the application URL by running these commands:
export POD_NAME=$(kubectl get pods --namespace default -l "app=chart,release=test" -o jsonpath="{.items[0].metadata.name}")
echo "Visit http://127.0.0.1:8080 to use your application"
kubectl port-forward $POD_NAME 8080:80
Job succeeded
deploy
Retry
Duration: 7 seconds
Timeout: 1h (from project)
Runner: gitlab-runner-96bd68669-xrpzf (#1)
Commit 5775740f
Helm CI
Pipeline #9 from v2.0.0
deploy
deploy

The End