Cloud Native and Microservices

Dominic Duggan
Stevens Institute of Technology

1

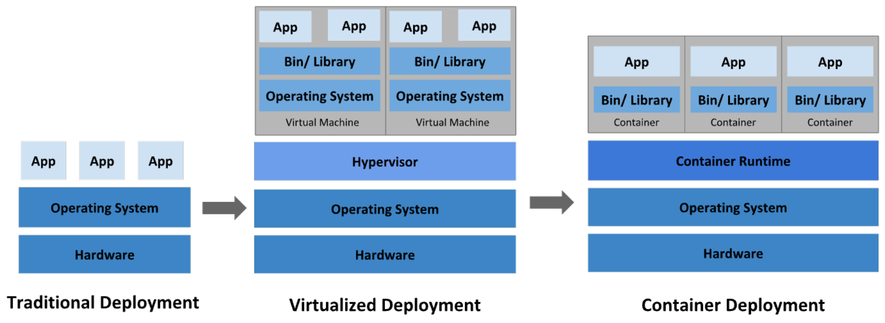
1

CONTAINERS ORCHESTRATION FRAMEWORKS

2

2

Background



3

Containers

- Agile (compared to VMs)
- Continuous development, integration, deployment
- Separation of concerns
  - Create container image at build/release time
- Observability (Health)
- Environmental consistency
- Loose coupling
- Resource isolation

4

Container Orchestration Frameworks

- Connect multiple containers together
- Scale up number of container instances

5

5

Container Orchestration Frameworks

- Service discovery & load balancing
- Storage orchestration
- Automated rollouts and rollbacks
- Automated bin packing
- Self healing
- Secret and configuration management

- **Not a PaaS!**

6

6

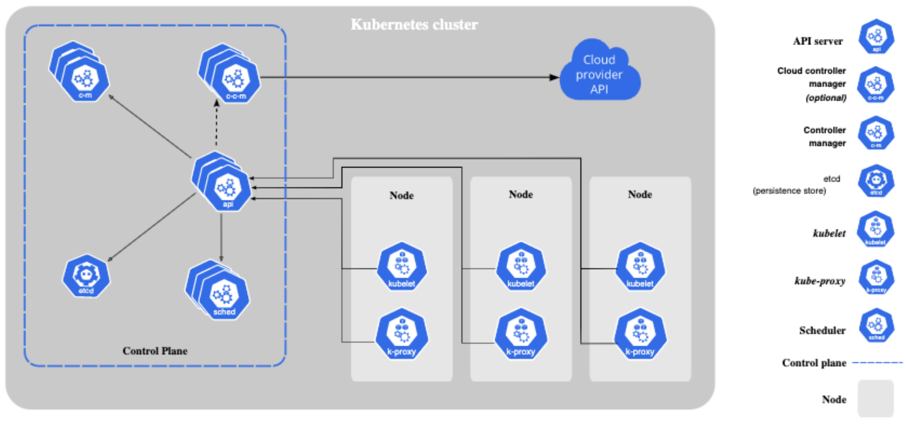
Example: Kubernetes

- Pod: atomic workload unit (Linux container)
- Deployment: manages running pods
  - Scaling up/down
  - Rolling updates
- Service: Logical abstraction for API
  - Direct client requests to pool of pods

7

7

Kubernetes Cluster



8

8

Kubernetes Control Plane

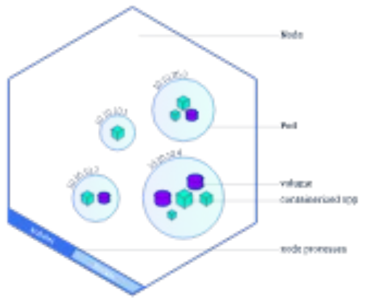
- API server
  - Scales horizontally
- Etcd
  - Highly available key-value store
  - Backing store for clusters
- Schedule *pods* to run on *nodes*
- Controller managers

9

9

Kubernetes: Node

- Worker machine
- Master schedules Pods on Nodes
- Node runs:
  - Docker runtime
  - Kubelet
    - Communication with Master
    - Manages Pods and Containers



10

10

Node Components

- Kubelet
  - Agent that runs on each node
  - Manage containers
- Kube-proxy
  - Network proxy
  - Packet filtering
- Container runtime

11

11

Node Status

- Addresses
  - External and internal IP address
- Conditions

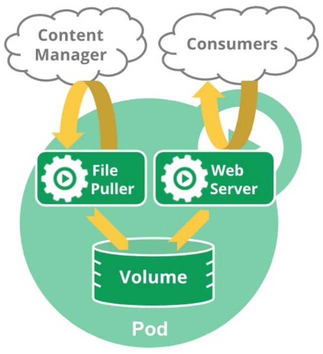
```
"conditions": [
  {
    "type": "Ready",
    "status": "True",
    "reason": "KubeletReady",
    "message": "kubelet is posting ready status",
    "lastHeartbeatTime": "2019-06-05T18:38:35Z",
    "lastTransitionTime": "2019-06-05T11:41:27Z"
  }
]
```

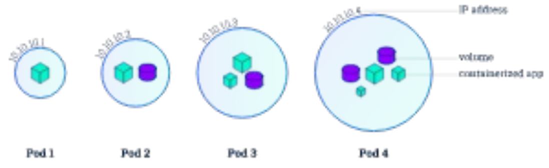
12

12

Kubernetes: Pod

- Abstraction that groups:
  - Application containers
    - Shared context
  - Resources
    - Volumes
    - Network IP address
    - Configuration





13

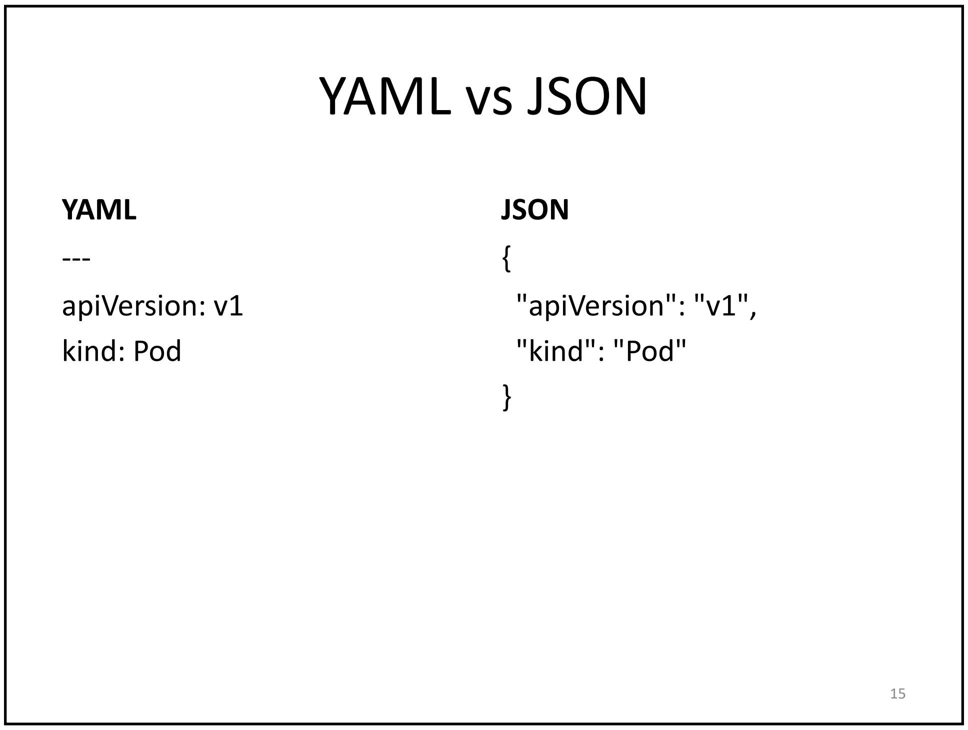
13

YAML

- Yet Another Markup Language
- Basically alternative syntax for JSON
- Configuration language
  - Kubernetes, OpenStack, etc

14

14



YAML vs JSON

YAML

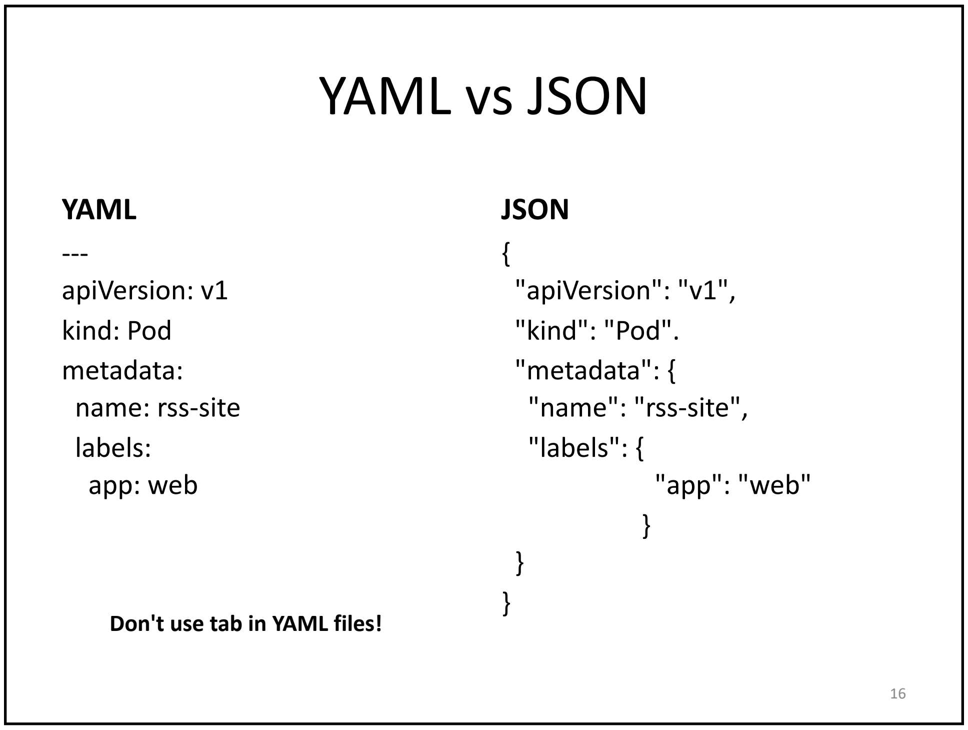
```yaml
---
apiVersion: v1
kind: Pod
```

JSON

```json
{
  "apiVersion": "v1",
  "kind": "Pod"
}
```

15

15



YAML vs JSON

YAML

```yaml
---
apiVersion: v1
kind: Pod
metadata:
  name: rss-site
  labels:
    app: web
```

Don't use tab in YAML files!

JSON

```json
{
  "apiVersion": "v1",
  "kind": "Pod".
  "metadata": {
    "name": "rss-site",
    "labels": {
                   "app": "web"
                 }
  }
}
```

16

16

YAML vs JSON

YAML

```
args:
  - sleep
  - "1000"
  - message
  - "Bring back Saul!"
```

JSON

```
{
  "args": [
    "sleep",
    "1000",
    "message",
    "Bring back Saul!"
  ]
}
```

17

17

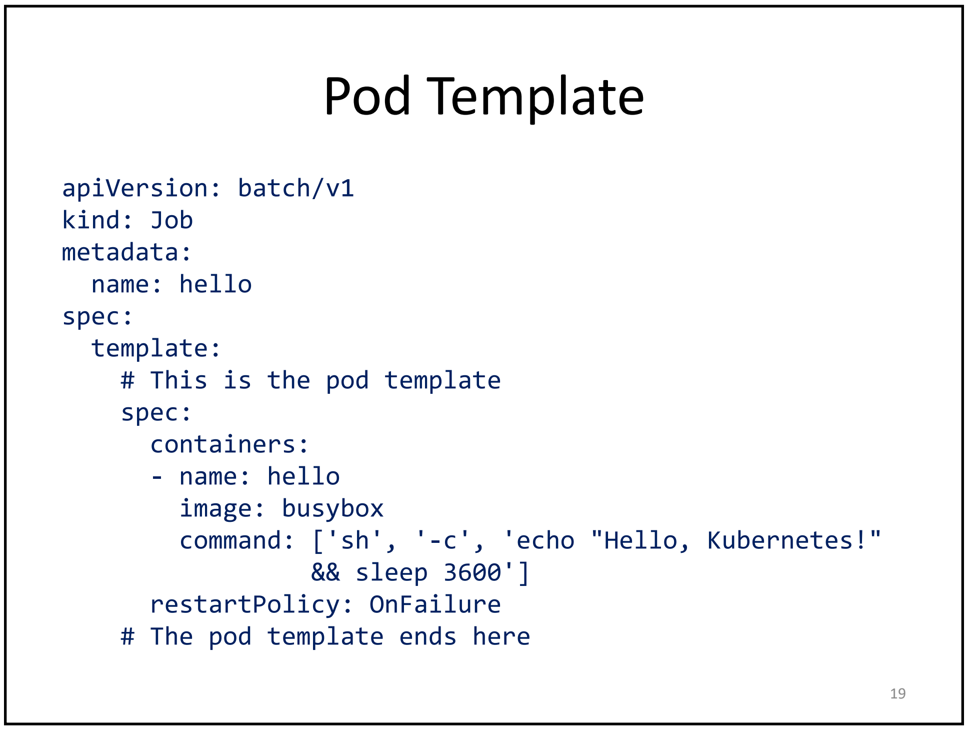
Pod in YAML

```
---
apiVersion: v1
kind: Pod
metadata:
  name: rss-site
  labels:
    app: web
spec:
  containers:
```

```
  - name: front-end
    image: nginx
    ports:
      - containerPort: 80

  - name: rss-reader
    image: rdr/php-nginx
    ports:
      - containerPort: 88
```

18

18

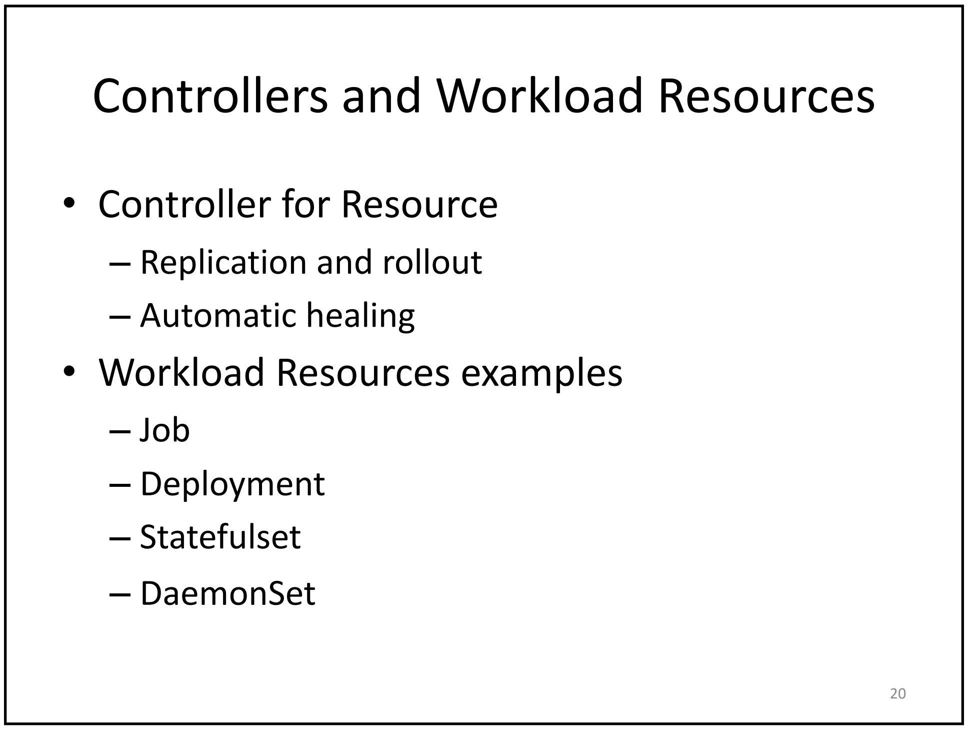


Pod Template

```
apiVersion: batch/v1
kind: Job
metadata:
  name: hello
spec:
  template:
    # This is the pod template
    spec:
      containers:
      - name: hello
        image: busybox
        command: ['sh', '-c', 'echo "Hello, Kubernetes!"
                 && sleep 3600']
      restartPolicy: OnFailure
    # The pod template ends here
```

19

19



Controllers and Workload Resources

- Controller for Resource
  - Replication and rollout
  - Automatic healing
- Workload Resources examples
  - Job
  - Deployment
  - Statefulset
  - DaemonSet

20

20

Example Job

```
apiVersion: batch/v1
kind: Job
metadata:
  name: hello
spec:
  template:
    # This is the pod template
    spec:
      containers:
      - name: hello
        image: busybox:1.28
        command: ['sh', '-c', 'echo "Hello!" && sleep 3600']
      restartPolicy: OnFailure
    # The pod template ends here
```

21

21

Kubernetes: Deployment

- Declarative update (desired state)
- Use cases
  - Ensure availability of a workload
  - Scale up for higher load
  - Change state of Pods
  - Roll back to earlier Deployment
  - Pause the Deployment
  - Check status of Deployment
  - Expose workload outside the cluster

22

22

Example Deployment

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: rss-site
spec:
  replicas: 2
  selector:
    matchLabels:
      app: web
```

```
template:
    metadata:
      labels:
        app: web
    spec:
      containers:
        - name: front-end
          image: nginx
          ports:
            - containerPort: 80
        - name: rss-reader
          image: rdr/php-nginx
          ports:
            - containerPort: 88
```

23

23

Example Deployment

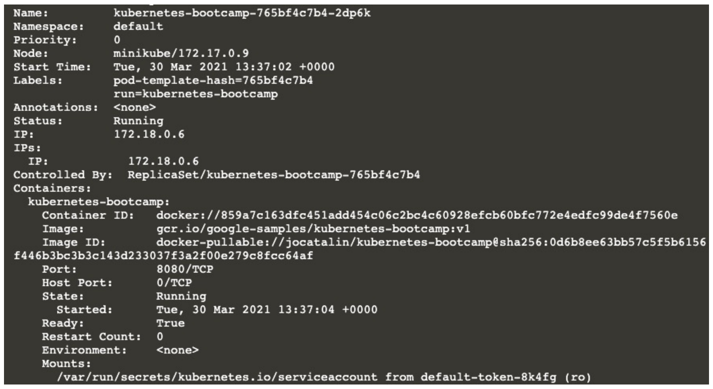
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
        - containerPort: 80
```

24

24

Kubernetes: CLI

```
$ kubectl get pods
$ kubectl describe pods
```



```
Name:         kubernetes-bootcamp-765bf4c7b4-2dp6k
Namespace:    default
Priority:     0
Node:         minikube/172.17.0.9
Start Time:   Tue, 30 Mar 2021 13:37:02 +0000
Labels:       pod-template-hash=765bf4c7b4
              run=kubernetes-bootcamp
Annotations:  <none>
Status:       Running
IP:           172.18.0.6
IPs:
  IP:           172.18.0.6
Controlled By:  ReplicaSet/kubernetes-bootcamp-765bf4c7b4
Containers:
  kubernetes-bootcamp:
    Container ID:   docker://859a7c163dfc451add454c06c2bc4c60928efcb60bfc772e4edfc99de4f7560e
    Image:          gcr.io/google-samples/kubernetes-bootcamp:v1
    Image ID:       docker-pullable://jocatalin/kubernetes-bootcamp@sha256:0d6b8ee63bb57c5f5b6156
f446b3bc3b3c143d233037f3a2f00e279c8fcc64af
    Port:           8080/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Tue, 30 Mar 2021 13:37:04 +0000
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-8k4fg (ro)
```

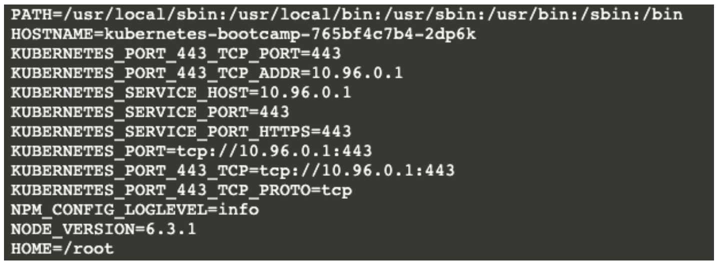
```
$ kubectl logs pod-name
```

25

25

Kubernetes: CLI

```
$ kubectl exec pod-name env
```



```
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
HOSTNAME=kubernetes-bootcamp-765bf4c7b4-2dp6k
KUBERNETES_PORT_443_TCP_PORT=443
KUBERNETES_PORT_443_TCP_ADDR=10.96.0.1
KUBERNETES_SERVICE_HOST=10.96.0.1
KUBERNETES_SERVICE_PORT=443
KUBERNETES_SERVICE_PORT_HTTPS=443
KUBERNETES_PORT=tcp://10.96.0.1:443
KUBERNETES_PORT_443_TCP=tcp://10.96.0.1:443
KUBERNETES_PORT_443_TCP_PROTO=tcp
NPM_CONFIG_LOGLEVEL=info
NODE_VERSION=6.3.1
HOME=/root
```

```
$ kubectl exec --ti pod-name /bin/bash
# curl localhost:8080
```

26

26

SERVICES

27

27

Kubernetes Networking

- Goal: Frictionless migration from VMs to containers
- Within a pod:
  - Containers communicate via loopback (localhost)
- Pods communicate in cluster without NAT
- Service resource
  - Expose app outside cluster
  - Ingress: specific for HTTP

28

28

Kubernetes Services

- Pods
  - Interchangeable among Pods in a ReplicaSet
  - Unique IP address
- Service
  - Discovery and routing among Pods
  - Loose coupling
  - Type:
    - ClusterIP: only within cluster
    - NodePort: each selected node in cluster
    - LoadBalancer: fixed external IP
    - ExternName: DNS

29

29

Kubernetes Services

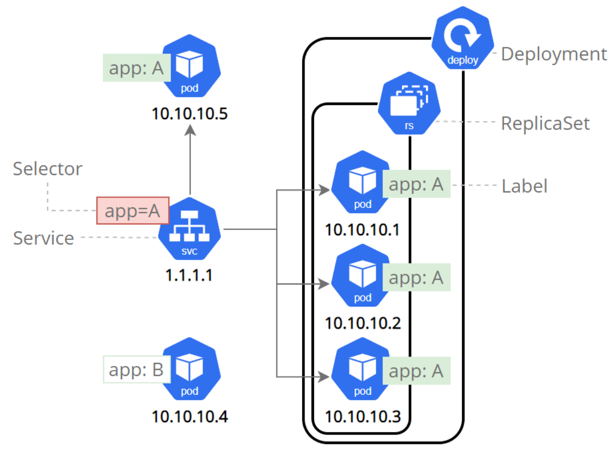
```
kind: Service
apiVersion: v1
metadata:
   name: hello-service
spec:
   selector:
      app: hello-pod
   ports:
      - port: 8080
```

30

30

Kubernetes Services

- Match Pods using Labels and Selectors

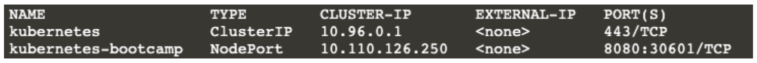


31

31

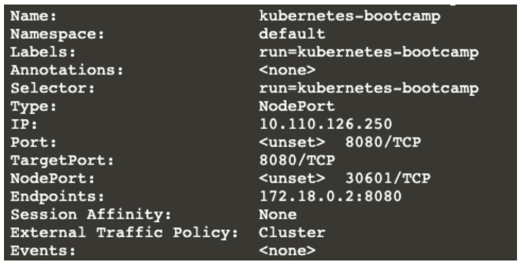
Kubernetes: CLI

```
$ kubectl expose deployment/service-name
      --type="NodePort" --port 8080
$ kubectl get services
```



```
NAME                  TYPE       CLUSTER-IP      EXTERNAL-IP   PORT(S)
kubernetes            ClusterIP  10.96.0.1       <none>        443/TCP
kubernetes-bootcamp   NodePort   10.110.126.250  <none>        8080:30601/TCP
```

```
$ kubectl describe services/service-name
```



```
Name:                     kubernetes-bootcamp
Namespace:                default
Labels:                   run=kubernetes-bootcamp
Annotations:              <none>
Selector:                 run=kubernetes-bootcamp
Type:                     NodePort
IP:                       10.110.126.250
Port:                     <unset>  8080/TCP
TargetPort:               8080/TCP
NodePort:                 <unset>  30601/TCP
Endpoints:                172.18.0.2:8080
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
```

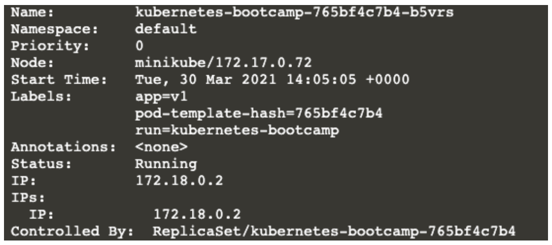
```
$ curl node-ip-address:30601
```

32

32

Kubernetes: Labels

```
$ kubectl get pods –l label
$ kubectl get services –l label
$ kubectl label pod pod-name app=v1
$ kubectl describe pods pod-name
```



```
Name:         kubernetes-bootcamp-765bf4c7b4-b5vrs
Namespace:    default
Priority:     0
Node:         minikube/172.17.0.72
Start Time:   Tue, 30 Mar 2021 14:05:05 +0000
Labels:       app=v1
              pod-template-hash=765bf4c7b4
              run=kubernetes-bootcamp
Annotations:  <none>
Status:       Running
IP:           172.18.0.2
IPs:
  IP:           172.18.0.2
Controlled By:  ReplicaSet/kubernetes-bootcamp-765bf4c7b4
```

```
$ kubectl get pods –l app=v1
```

33

33

Example Deployment

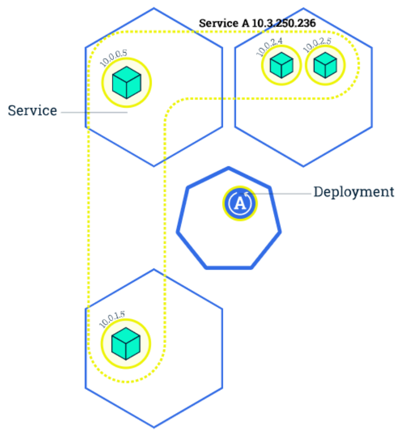
```
kind: Deployment
apiVersion: apps/v1
metadata:
  name: hello-deploy
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: hello-pod
    spec:
```

```
spec:
  containers:
  - name: hello-container
    image: ex.com/hello-cloud:1
  imagePullPolicy: IfNotPresent
  livenessProbe:
    httpGet:
      path: /
      port: 8080
  readinessProbe:
    httpGet:
      path: /resources/hello
      port: 8080
  restartPolicy: Always
```

34

34

Kubernetes: Scaling

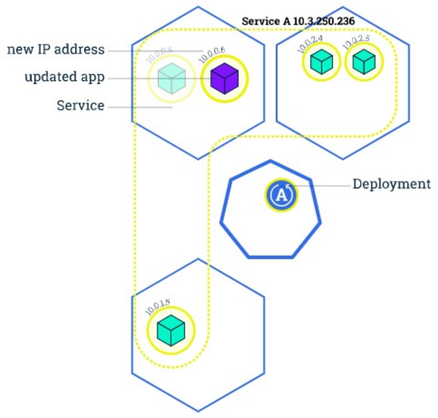


```
kubectl scale deployments/name –replicas=4
```

35

35

Kubernetes: Rolling Updates



```
kubectl scale deployments/name –replicas=4
```

36

36

Exposing External IP Address

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app.kubernetes.io/name: load-balancer-example
  name: hello-world
spec:
  replicas: 5
  selector:
    matchLabels:
      app.kubernetes.io/name: load-balancer-example
  template:
    metadata:
      labels:
        app.kubernetes.io/name: load-balancer-example
    spec:
      containers:
      - image: gcr.io/google-samples/node-hello:1.0
        name: hello-world
        ports:
        - containerPort: 8080
```

37

37

Exposing External IP Address

- Deploy

```
kubectl apply –f load-balance-example.yaml
```

- Expose with a Service

```
kubectl expose deployment hello-world
   --type=LoadBalancer --name=my-service
kubectl get services my-service
```

- Output

```
NAME       TYPE         EXTERNAL-IP    PORT(S)  AGE
my-service LoadBalancer 104.198.205.71 8080/TCP 54s
```

38

38

Exposing External IP Address

- Detailed descrption

  ```
  kubectl describe services my-service
  ```

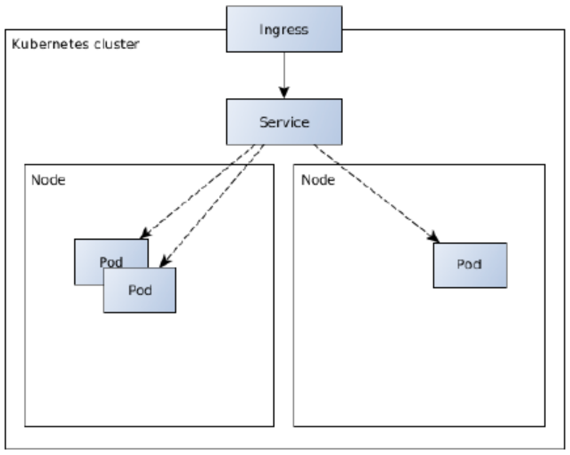
- Output

  ```
  Name:             my-service
  Namespace:        default
  Labels:           app.kubernetes.io/name=load-balancer-example
  Annotations:      <none>
  Selector:         app.kubernetes.io/name=load-balancer-example
  Type:             LoadBalancer
  IP:               10.3.245.137
  LoadBalancer Ingress:   104.198.205.71
  Port:             <unset> 8080/TCP
  NodePort:         <unset> 32377/TCP
  Endpoints:        10.0.0.6:8080,10.0.1.6:8080,10.0.1.7:8080 +
  2 more...
  Session Affinity:   None
  Events:           <none>
  ```

39

39

Kubernetes Ingress



40

40

Kubernetes Ingress

```
kind: Ingress
apiVersion: extensions/v1beta1
metadata:
  name: hello-ingress
spec:
  rules:
  - host: hello.example.com
    http:
      paths:
      - path: /
        backend:
          serviceName: hello-service
          servicePort: 8080
```

41

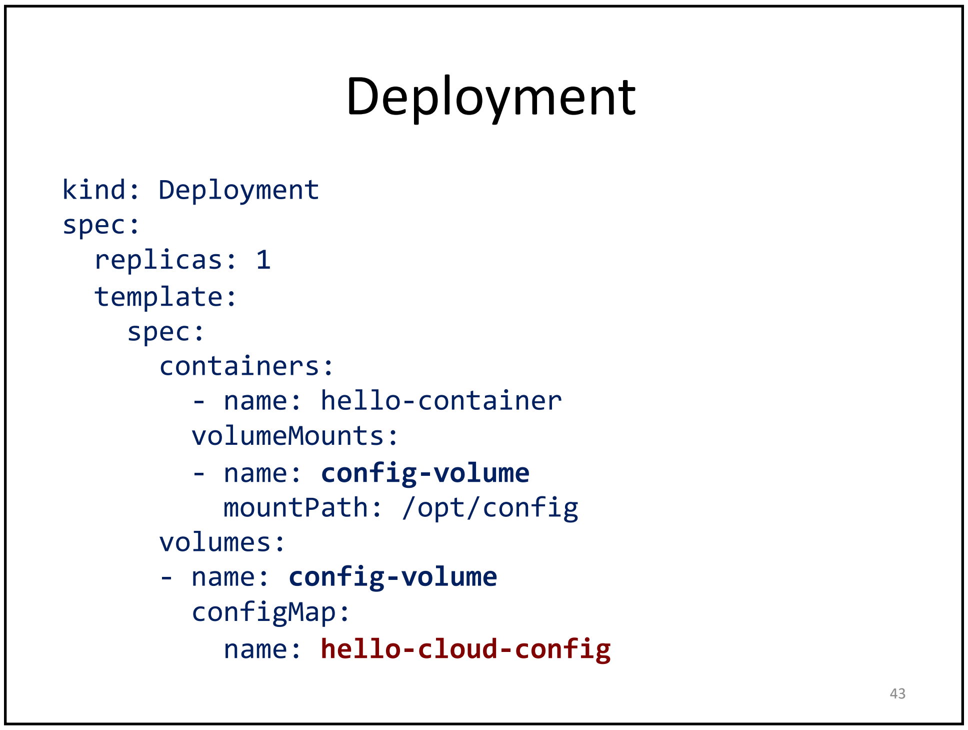
41

ConfigMap

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: hello-cloud-config
data:
  application.properties: |
    hello.greeting=Hello from Kubernetes
    hello.name=Java EE
```

42

42

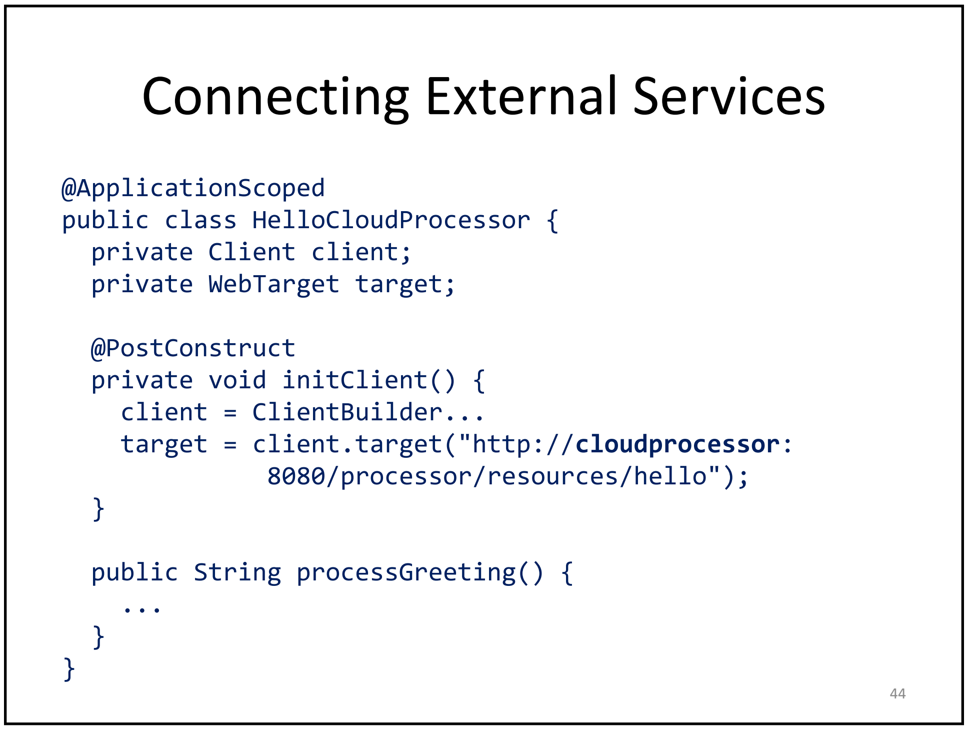


Deployment

```
kind: Deployment
spec:
  replicas: 1
  template:
    spec:
      containers:
        - name: hello-container
        volumeMounts:
        - name: config-volume
          mountPath: /opt/config
      volumes:
      - name: config-volume
        configMap:
          name: hello-cloud-config
```

43

43

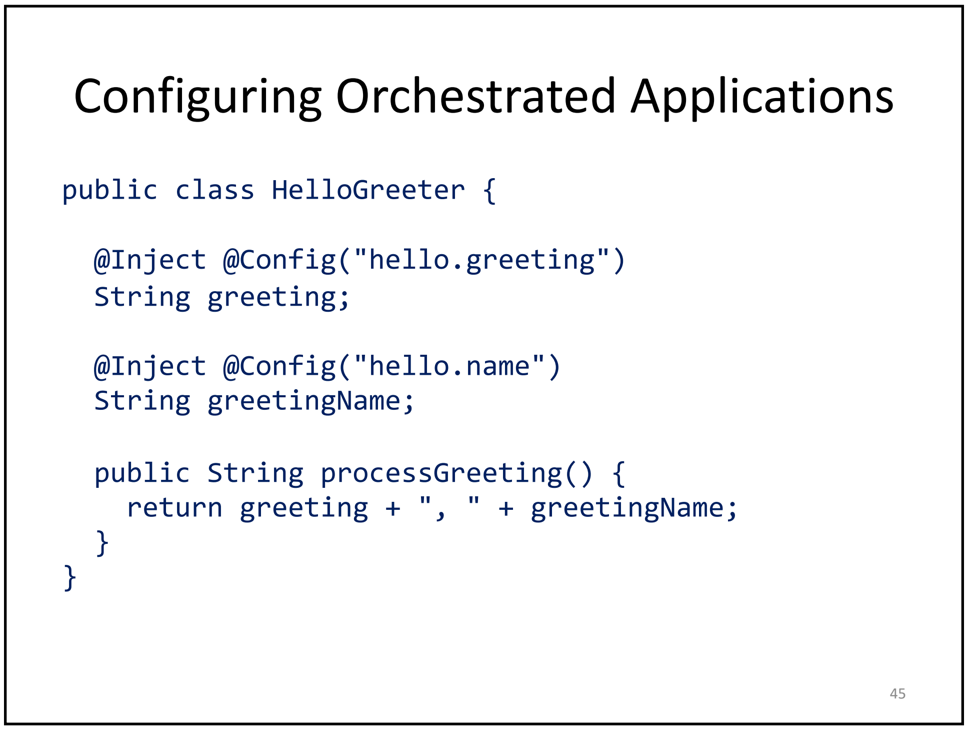


Connecting External Services

```
@ApplicationScoped
public class HelloCloudProcessor {
  private Client client;
  private WebTarget target;

  @PostConstruct
  private void initClient() {
    client = ClientBuilder...
    target = client.target("http://cloudprocessor:
              8080/processor/resources/hello");
  }

  public String processGreeting() {
    ...
  }
}
```

44

44

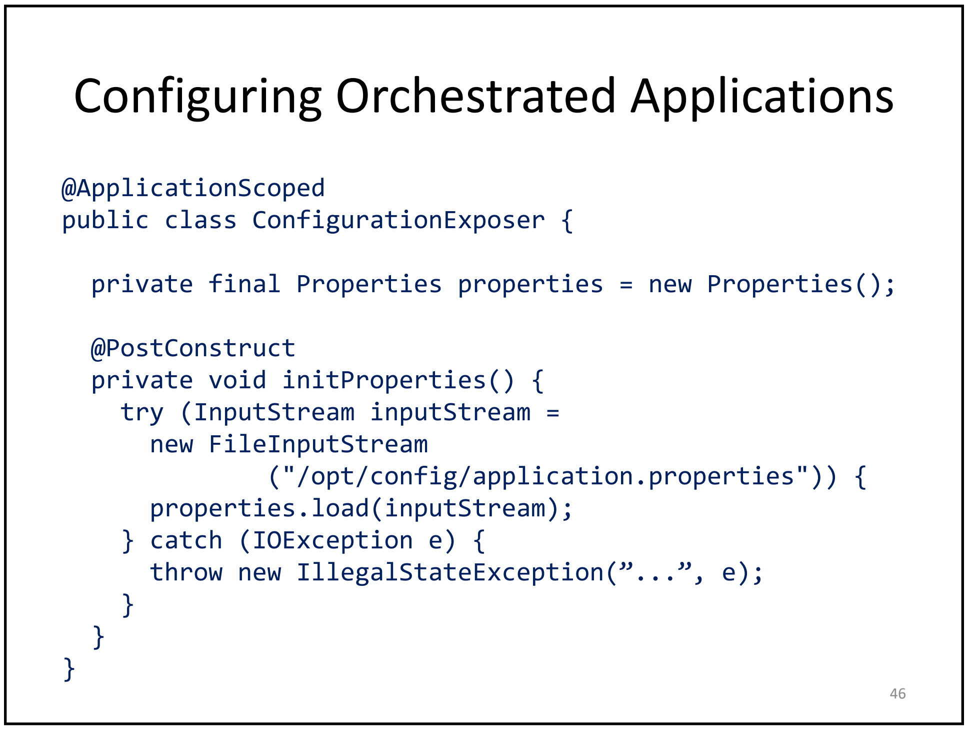


Configuring Orchestrated Applications

```
public class HelloGreeter {

  @Inject @Config("hello.greeting")
  String greeting;

  @Inject @Config("hello.name")
  String greetingName;

  public String processGreeting() {
    return greeting + ", " + greetingName;
  }
}
```

45

45

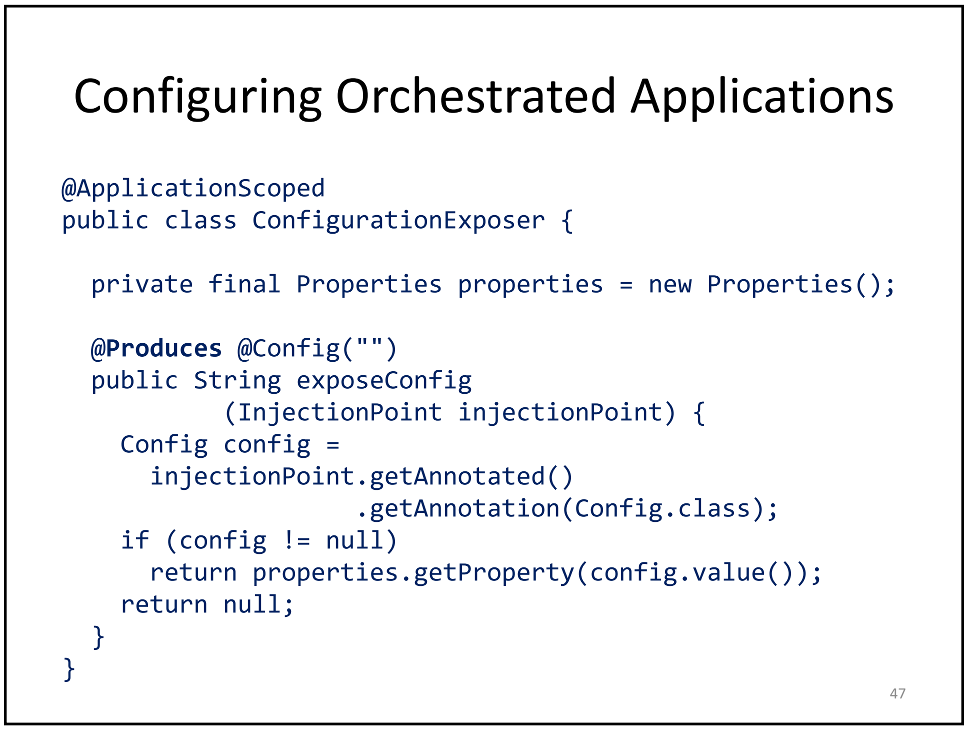


Configuring Orchestrated Applications

```
@ApplicationScoped
public class ConfigurationExposer {

  private final Properties properties = new Properties();

  @PostConstruct
  private void initProperties() {
    try (InputStream inputStream =
      new FileInputStream
              ("/opt/config/application.properties")) {
      properties.load(inputStream);
    } catch (IOException e) {
      throw new IllegalStateException(”...”, e);
    }
  }
}
```

46

46



Configuring Orchestrated Applications

```
@ApplicationScoped
public class ConfigurationExposer {

  private final Properties properties = new Properties();

  @Produces @Config("")
  public String exposeConfig
           (InjectionPoint injectionPoint) {
    Config config =
      injectionPoint.getAnnotated()
                    .getAnnotation(Config.class);
    if (config != null)
      return properties.getProperty(config.value());
    return null;
  }
}
```

47

47