

## Closer inspection

1. To observe what's in our cluster use: `kubectl get all`
2. To observe deployments use: `kubectl get deployment`
3. To introspect into a deployment use: `kubectl get deployment/[NAME] -o yaml`
4. To observe services use: `kubectl get services`
5. To introspect into services use: `get service/[NAME] -o yaml`

```
PS C:\WINDOWS\system32> kubectl get deployment
NAME          READY  UP-TO-DATE  AVAILABLE  AGE
helloworld    1/1    1            1           13s
PS C:\WINDOWS\system32> kubectl get deployment/helloworld -o yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
  creationTimestamp: "2022-02-20T22:16:51Z"
  generation: 1
  name: helloworld
  namespace: default
  resourceVersion: "614"
  uid: 28cf2a8b-8a7b-4929-ba4d-b07364318b9e
spec:
  progressDeadlineSeconds: 600
  replicas: 1
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: helloworld
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: helloworld
    spec:
      containers:
      - image: karthequian/helloworld:latest
        imagePullPolicy: Always
        name: helloworld
        ports:
        - containerPort: 80
          protocol: TCP
        resources: {}
        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
      dnsPolicy: ClusterFirst
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext: {}
      terminationGracePeriodSeconds: 30
status:
  availableReplicas: 1
  conditions:
  - lastTransitionTime: "2022-02-20T22:16:58Z"
    lastUpdateTime: "2022-02-20T22:16:58Z"
    message: Deployment has minimum availability.
    reason: MinimumReplicasAvailable
    status: "True"
  - lastTransitionTime: "2022-02-20T22:16:32Z"
    lastUpdateTime: "2022-02-20T22:16:58Z"
    message: ReplicaSet "helloworld-47c6d56" has successfully progressed.
    reason: NewReplicaSetAvailable
    status: "True"
  observedGeneration: 1
  readyReplicas: 1
  replicas: 1
  updatedReplicas: 1
PS C:\WINDOWS\system32>
```

```
Administrator: Windows PowerShell
PS C:\WINDOWS\system32> kubectl get service
NAME          TYPE          CLUSTER-IP    EXTERNAL-IP  PORT(S)          AGE
helloworld    NodePort      10.104.15.147 <none>        80:31268/TCP     32m
kubernetes    ClusterIP     10.96.0.1      <none>        443/TCP           41m
PS C:\WINDOWS\system32> kubectl get service/helloworld -o yaml
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: "2022-02-20T22:21:02Z"
  name: helloworld
  namespace: default
  resourceVersion: "854"
  uid: 47f5db5e-24a5-447a-9b89-f8f990d7e5ca
spec:
  clusterIP: 10.104.15.147
  clusterIPs:
  - 10.104.15.147
  externalTrafficPolicy: Cluster
  internalTrafficPolicy: Cluster
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - nodePort: 31268
    port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: helloworld
  sessionAffinity: None
  type: NodePort
status:
  loadBalancer: {}
PS C:\WINDOWS\system32>
```

## Making deployment efficient

>To print the content of a file into the standard output stream use: `cat [document name]`

>We can deploy apps individually using: `kubectl create -f [filename]`

To make deployment more efficient we can combine deployment and services into a singular document separating them with 3 dashes ---

1. To deploy combined yml file use: `kubectl create -f [docname]`

You should observe a list of deployment/services created.

```
Administrator: Windows PowerShell

PS C:\Users\msuban01\Desktop\kube\Exercise\03_05> cat helloworld-all.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: helloworld-all-deployment
spec:
  selector:
    matchLabels:
      app: helloworld
  replicas: 1 # tells deployment to run 1 pods matching the template
  template: # create pods using pod definition in this template
    metadata:
      labels:
        app: helloworld
    spec:
      containers:
        - name: helloworld
          image: karthequian/helloworld:latest
          ports:
            - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: helloworld-all-service
spec:
  type: NodePort
  ports:
    - port: 80
      protocol: TCP
      targetPort: 80
  selector:
    app: helloworld
PS C:\Users\msuban01\Desktop\kube\Exercise\03_05> kubectl create -f helloworld-all.yml
deployment.apps/helloworld-all-deployment created
service/helloworld-all-service created
PS C:\Users\msuban01\Desktop\kube\Exercise\03_05> kubectl get deployment
NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
helloworld                          1/1     1             1           88m
helloworld-all-deployment          1/1     1             1           34s
helloworld-deployment               1/1     1             1           5m27s
PS C:\Users\msuban01\Desktop\kube\Exercise\03_05> kubectl get services
NAME                                TYPE               CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
helloworld                          NodePort           10.104.15.147    <none>         80:31268/TCP     84m
helloworld-all-service              NodePort           10.105.148.2     <none>         80:31164/TCP     62s
helloworld-service                   NodePort           10.98.241.152    <none>         80:30209/TCP     5m31s
kubernetes                           ClusterIP          10.96.0.1        <none>         443/TCP          93m
PS C:\Users\msuban01\Desktop\kube\Exercise\03_05>
```

```
Administrator: Windows PowerShell

PS C:\Users\msuban01\Desktop\kube\Exercise\03_05> cat helloworld-deployment.yml
# Helloworld application- just the deployment
apiVersion: apps/v1
kind: Deployment
metadata:
  name: helloworld-deployment
spec:
  selector:
    matchLabels:
      app: helloworld
  replicas: 1 # tells deployment to run 1 pods matching the template
  template: # create pods using pod definition in this template
    metadata:
      labels:
        app: helloworld
    spec:
      containers:
        - name: helloworld
          image: karthequian/helloworld:latest
          ports:
            - containerPort: 80
PS C:\Users\msuban01\Desktop\kube\Exercise\03_05> cat helloworld-service.yml
# Helloworld service for nodeports
apiVersion: v1
kind: Service
metadata:
  name: helloworld-service
spec:
  type: NodePort
  ports:
    - port: 80
      protocol: TCP
      targetPort: 80
  selector:
    app: helloworld
PS C:\Users\msuban01\Desktop\kube\Exercise\03_05> kubectl create -f helloworld-deployment.yml
deployment.apps/helloworld-deployment created
PS C:\Users\msuban01\Desktop\kube\Exercise\03_05> kubectl create -f helloworld-service.yml
service/helloworld-service created
PS C:\Users\msuban01\Desktop\kube\Exercise\03_05> kubectl get all
NAME                                READY   STATUS    RESTARTS   AGE
pod/helloworld-d7c6dd56-6rww         1/1     Running   0           84m
pod/helloworld-deployment-d7c6dd56-wg4x9  1/1     Running   0           59s
NAME                                TYPE               CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
service/helloworld                  NodePort           10.104.15.147    <none>         80:31268/TCP     79m
service/helloworld-service          NodePort           10.98.241.152    <none>         80:30209/TCP     35s
service/kubernetes                   ClusterIP          10.96.0.1        <none>         443/TCP          88m
NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/helloworld          1/1     1             1           84m
deployment.apps/helloworld-deployment 1/1     1             1           59s
NAME                                DESIRED   CURRENT   READY   AGE
replicaset.apps/helloworld-d7c6dd56 1         1         1       84m
replicaset.apps/helloworld-deployment-d7c6dd56 1         1         1       59s
PS C:\Users\msuban01\Desktop\kube\Exercise\03_05>
```