You can run an application by creating a Kubernetes Deployment object, and you can describe a Deployment in a YAML file. For example, this YAML file describes a Deployment that runs the nginx:1.7.9 Docker image:

deployment.yaml

```
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
kind: Deployment
metadata:
name: nginx-deployment
spec:
selector:
matchLabels:
app: nginx
replicas: 2 # tells deployment to run 2 pods matching the template
template:
metadata:
labels:
app: nginx
spec:
containers:
- name: nginx
image: nginx:1.7.9
ports:
- containerPort: 80
```

1. Create a Deployment based on the YAML file:

$ kubectl apply -f deployment.yaml
2. Display information about the Deployment:
$ kubectl describe deployment nginx-deployment
The output is similar to this:
```
user@computer:~/website$ kubectl describe deployment nginx-deployment
Name:    nginx-deployment
Namespace:    default
CreationTimestamp:  Tue, 30 Aug 2016 18:11:37 -0700
Labels:    app=nginx
Annotations:    deployment.kubernetes.io/revision=1
Selector:   app=nginx
Replicas:   2 desired | 2 updated | 2 total | 2 available | 0 unavailable
StrategyType:   RollingUpdate
MinReadySeconds:  0
```

RollingUpdateStrategy: 1 max unavailable, 1 max surge
Pod Template:
  Labels: app=nginx
  Containers:
   nginx:
    Image: nginx:1.7.9
    Port: 80/TCP
    Environment: <none>
    Mounts: <none>
   Volumes: <none>
Conditions:
  Type   Status Reason
  ----   ------ ------
  Available   True   MinimumReplicasAvailable
  Progressing   True   NewReplicaSetAvailable
OldReplicaSets: <none>
NewReplicaSet: nginx-deployment-1771418926 (2/2 replicas created)
No events.

3. List the pods created by the deployment:

$ **kubectl get pods -l app=nginx**

The output is similar to this:
NAME   READY   STATUS   RESTARTS AGE
nginx-deployment-1771418926-7o5ns 1/1   Running 0   16h
nginx-deployment-1771418926-r18az 1/1   Running 0   16h

4. Display information about a pod:

$ **kubectl describe pod <pod-name>**

Updating the deployment

You can update the deployment by applying a new YAML file. This YAML file specifies that the deployment should be updated to use nginx 1.8.

application/deployment-update.yaml

```
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
kind: Deployment
metadata:
name: nginx-deployment
spec:
selector:
matchLabels:
app: nginx
replicas: 2
template:
metadata:
labels:
app: nginx
spec:
containers:
- name: nginx
image: nginx:1.8 # Update the version of nginx from 1.7.9 to 1.8
ports:
- containerPort: 80
```

1. Apply the new YAML file:

$ kubectl apply -f deployment-update.yaml

2. Watch the deployment create pods with new names and delete the old pods:
$ kubectl get pods -l app=nginx

$ kubectl get pods -l app=nginx

Scaling the application by increasing the replica count

You can increase the number of pods in your Deployment by applying a new YAML file. This YAML file sets replicas to 4, which specifies that the Deployment should have four pods:

application/deployment-scale.yaml

```
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
kind: Deployment
metadata:
name: nginx-deployment
spec:
```

```
selector:
matchLabels:
app: nginx
replicas: 4 # Update the replicas from 2 to 4
template:
metadata:
labels:
app: nginx
spec:
containers:
- name: nginx
image: nginx:1.8
ports:
- containerPort: 80
```

1. Apply the new YAML file:
$ kubectl apply -f deployment-scale.yaml

2. Verify that the Deployment has four pods:

$ **kubectl get pods -l app=nginx**

The output is similar to this:
```
NAME                          READY   STATUS    RESTARTS   AGE
nginx-deployment-148880595-4zdqq   1/1     Running   0       25s
nginx-deployment-148880595-6zgi1   1/1     Running   0       25s
nginx-deployment-148880595-fxcez   1/1     Running   0       2m
nginx-deployment-148880595-rwovn   1/1     Running   0       2m
```

Deleting a deployment

Delete the deployment by name:
$ kubectl delete deployment nginx-deployment