```
docker manifest inspect microsoft/dotnet
docker inspect microsoft/dotnedocker run -d -p 8080:80 nginx

-- AKS
%USERPROFILE%\.kube\config

az account set --subscription 7039c21f-5ec5-4d0a-b0ab-cdb574a91063
az aks get-credentials --resource-group CKAD_Workshop --name CKAD

kubectl cluster-info
kubectl api-resources

kubectl proxy
http://127.0.0.1:8001/api/v1




-- POD
kubectl run busybox --image=busybox --command --restart=Never -- env

kubectl run busybox --image=busybox --restart=Never --dry-run=client
-o yaml --command -- env > envpod.yaml
# see it
cat envpod.yaml
kubectl apply -f envpod.yaml


kubectl run nginx --image=nginx --restart=Never --port=80

kubectl run busybox --image=busybox -it --rm --restart=Never -- /
bin/sh -c 'echo hello world'

-- Debugging / Troubleshooting
kubectl get po -o wide # gets more information
kubectl describe po nginx

kubectl logs busybox
kubectl logs nginx --previous

 # SERVER EVENTS
kubectl run busybox --restart=Never --image=busybox -- notexist
kubectl logs busybox # will bring nothing! container never started
kubectl describe po busybox # in the events section, you'll see the
error
# also...
kubectl get events | grep -i error # you'll see the error here as
well
kubectl delete po busybox --force --grace-period=0

kubectl top nodes
```

```
-- working with desired state
kubectl run nginx --image=nginx --restart=Never --dry-run=client -n
mynamespace -o yaml > pod.yaml
kubectl create -f pod.yaml

-- POD DESIGN
kubectl run nginx1 --image=nginx --restart=Never --labels=app=v1
kubectl get po --show-labels

kubectl label po nginx2 app=v2 --overwrite
kubectl get po -l app=v2

kubectl label po -l app app-
kubectl label po nginx1 nginx2 nginx3 app-

kubectl label nodes <your-node-name> accelerator=nvidia-tesla-p100
kubectl get nodes --show-labels

kubectl run nginx --image=nginx --restart=Never --
requests='cpu=100m,memory=256Mi' --limits='cpu=200m,memory=512Mi'

-- JOBS

kubectl create job pi  --image=perl -- perl -Mbignum=bpi -wle 'print
bpi(2000)'

kubectl get jobs -w # wait till 'SUCCESSFUL' is 1 (will take some
time, perl image might be big)
kubectl get po # get the pod name
kubectl logs pi-**** # get the pi numbers
kubectl delete job pi

kubectl wait --for=condition=complete --timeout=300 job pi
kubectl delete job pi


paralleljobs.yaml

kubectl create cronjob busybox --image=busybox --schedule="*/1 * * *
*" -- /bin/sh -c 'date; echo Hello from the Kubernetes cluster'


-- Deployments

kubectl apply -f https://k8s.io/examples/controllers/nginx-
deployment.yaml
kubectl get deployments

kubectl set image deployment/nginx-deployment nginx=nginx:1.16.1 --
record

kubectl edit deployment.v1.apps/nginx-deployment
kubectl rollout status deployment/nginx-deployment
```

```
kubectl get rs
kubectl get pods
kubectl describe deployments

kubectl set image deployment.v1.apps/nginx-deployment
nginx=nginx:1.161 --record=true

kubectl rollout history deployment.v1.apps/nginx-deployment
kubectl rollout undo deployment.v1.apps/nginx-deployment --to-
revision=2

kubectl scale deployment.v1.apps/nginx-deployment --replicas=10
kubectl autoscale deployment.v1.apps/nginx-deployment --min=10 --
max=15 --cpu-percent=80

kubectl rollout pause deployment.v1.apps/nginx-deployment

-- namespaces
kubectl create namespace mynamespace
kubectl run nginx --image=nginx --restart=Never -n mynamespace

kubectl get po --all-namespaces

-- Configuration

kubectl create configmap config --from-literal=foo=lala --from-
literal=foo2=lolo

kubectl describe cm config

kubectl create cm configmap2 --from-file=config.txt
kubectl create cm configmap3 --from-env-file=config.env


kubectl get cm configmap4 -o yaml

kubectl create cm options --from-literal=var5=val5
kubectl create -f configmap.yaml
kubectl exec -it nginx -- env | grep option # will show
'option=val5'


--- SECRETS
kubectl create secret generic mysecret --from-
literal=password=mypass
kubectl create secret generic mysecret2 --from-file=username

$value = kubectl get secret mysecret2 -o jsonpath='{.data.username}
{"\n"}'
[System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64S
tring($value ))

kubectl create -f pod.yaml
kubectl exec -it nginx /bin/bash
```

```
ls /etc/foo  # shows username
cat /etc/foo/username # shows admin


-- SERVICE ACCOUNTS
kubectl create sa myuser


kubectl get sa default -o yaml > sa.yaml
vim sa.yaml

kubectl create -f pod.yaml
kubectl describe pod nginx


-- LIVENESS PROBE

kubectl create -f livenessprobe.yaml
kubectl describe pod nginx
kubectl delete -f livenessprobe.yaml


kubectl create -f delayedliveness.yaml
kubectl describe pod nginx
kubectl delete -f delayedliveness.yaml

-- SERVICES

kubectl run nginx --image=nginx --restart=Never --port=80 --expose

kubectl get svc nginx # services
kubectl get ep # endpoints

kubectl patch svc nginx -p '{"spec":{"type":"NodePort"}}'

kubectl create deploy foo --image=dgkanatsios/simpleapp --port=8080
--replicas=3
kubectl expose deploy foo --port=6262 --target-port=8080

kubectl create -f policy.yaml

-- VOLUMES
volume.yaml

kubectl exec -it busybox -c busybox2 -- /bin/sh
cat /etc/passwd | cut -f 1 -d ':' > /etc/foo/passwd
cat /etc/foo/passwd # confirm that stuff has been written
successfully
exit


kubectl create -f persistentvolume.yaml
# will have status 'Available'
kubectl get pv
```

```
kubectl create -f pvc.yaml
kubectl get pvc # will show as 'Bound'
kubectl get pv # will show as 'Bound' as well
```