CLUSTER INFO & EVENTS

```
# list the kube config settings
kubectl config view

# list all the users in the cluster
kubectl config view -o jsonpath='{.users[*].name}'

# find where control plane is running
kubectl cluster-info

# get system health
kubectl get componentstatus

# list all resources available to create
kubectl api-resources

# get the raw metrics for all nodes
kubectl get --raw /apis/metrics.k8s.io/v1beta1/nodes

# get the raw metrics for all pods
kubectl get --raw /apis/metrics.k8s.io/v1beta1/pods

# list all events in the default namespace
kubectl get events

# list all events in all namespaces
kubectl get events --all-namespaces

# list all events in the 'kube-system' namespace
kubectl get events -n kube-system

# watch events in real time in default namespace
kubectl get events -w
```

```
# create namespace 'robot-shop'
kubectl create ns robot-shop

# list all namespaces in the cluster
kubectl get ns

# get the yaml config for all namespaces
kubectl get ns -o yaml

# list all kubernetes resources in all namespaces
kubectl get all --all-namespaces

# describe the namespace configuration
kubectl describe ns

# edit namespace 'robot-shop'
kubectl edit ns robot-shop

# delete namespace 'robot-shop'
kubectl delete ns robot-shop

# list all available contexts from kube config
kubectl config get-contexts

# get the current context for kubectl
kubectl config current-context

# switch context to a cluster named 'gkeCluster'
kubectl config set-context gkeCluster

# change context 'ro-shop' namespace in 'gkeCluster'
kubectl config set-context gkeCluster --namespace ro-shop

# change context to user 'admin' in cluster 'gkeCluster'
kubectl config set-context gkeCluster --user=admin

# set the default context to the cluster 'gkeCluster'
kubectl config use-context gkeCluster
```

NODES

```
# list all nodes in the default namespace
kubectl get no

# same as previous, but with additional info
kubectl get no -o wide

# describe all nodes
kubectl describe no

# label node 'mynode1` with key 'disk' and value 'ssd'
kubectl label no mynode1 disk=ssd

# show labels for nodes in a cluster
kubectl get no --show-labels

# annotate node 'mynode1' with key 'azure' & value 'node'
kubectl annotate no mynode1 azure=node

# get IP addresses of nodes in default namespace
kubectl get nodes \
-o jsonpath='{items[*].status.addresses\
[?(@.type=="ExternalIP")].addresses}'

# view resource utilization of node 'mynode1'
kubectl top node mynode1

# taint 'mynode1',key='node-role.kubernetes.io'
# and effect 'NoSchedule'
kubectl taint no mynode1 node-role.kubernetes.io:NoSchedule

# taint 'mynode1',key 'dedicated',value 'special-user'
# and effect 'NoSchedule'
kubectl taint no mynode1 dedicated=special-user:NoSchedule

# Remove taint from 'mynode1' with key 'dedicated'
# and effect 'NoSchedule'
kubectl taint no mynode1 dedicated:NoSchedule-

# Remove taints with key 'dedicated' from 'mynode1'
kubectl taint no mynode1 dedicated-

# list the taints applied to all nodes
kubectl describe no | grep Taint

# taint nodes with the label 'disk=ssd' with key 'dedicated'
kubectl taint no -l disk=ssd \
dedicated=mynode1:PreferNoSchedule

# taint 'mynode1' with key 'bar' and no value
kubectl taint no mynode1 bar:NoSchedule

# drain node 'mynode1' to remove any scheduled pods
# while ensuring no pods are scheduled to it
kubectl drain mynode1 --ignore-daemonsets --force

# cordon node 'mynode1', ensure no pods are scheduled
kubectl cordon mynode1

# uncordon node 'mynode1', resume scheduling pods
kubectl uncordon mynode1

# delete node 'mynode1' from the cluster
kubectl delete no mynode1

# edit the configuration of 'mynode1'
kubectl edit no mynode1
```

PODS

```
# create pod 'nginx' using the 'nginx' image
kubectl run nginx --image=nginx

# create pod 'busybox', open a shell to container
# delete pod upon exit
kubectl run busybox --image=busybox --rm -it -- sh

# create a pod yaml file named pod.yml
kubectl run nginx --image=nginx \
--dry-run=client -o yaml > pod.yml

# list all pods in the default namespace
kubectl get po

# list all pods in all namespaces
kubectl get po --all-namespaces

# list all kubernetes resources in all namespaces
kubectl get all --all-namespaces

# list all pods, nodes and services in all namespaces
kubectl get po,no,svc --all-namespaces

# same as above but return additional info
kubectl get po -o wide

# describe all pods in default namespace
kubectl describe po

# give pod 'nginx' a label of 'app=prod'
kubectl label nginx app=prod

# show the labels for pods in default namespace
kubectl get po --show-labels

# show pods with a label of 'app=nginx'
kubectl get po -l app=nginx

# annotate 'nginx' with key 'special', value of 'app1'
kubectl annotate po nginx special=app1

# show the yaml output for the pod named nginx
kubectl get po nginx -o yaml

# export yaml of pod 'nginx' to 'podconfig.yml'
kubectl get pod nginx -o yaml --export > podconfig.yml

# list all the pods that are running
kubectl get po --field-selector status.phase=Running

# run 'mongo' command inside terminal in pod 'mongodb'
kubectl exec -it mongodb mongo

# list environment variables in pod 'nginx'
kubectl exec -it nginx env

# open shell to container 'cart' in pod 'mypod'
kubectl exec -it mypod -c cart -- /bin/bash

# get the log output for a pod named 'nginx' in the default
namespace
kubectl logs nginx

# same as above but output to a file named 'pod.log'
kubectl logs nginx > pod.log

# get the last hour of log output for a pod named 'nginx'
kubectl logs nginx --since=1h

# get the last 20 lines of a log output for a pod named
'nginx'
kubectl logs nginx --tail=20

# get the streaming log output for a container named 'log' in
a pod named 'nginx'
kubectl logs -f nginx -c log

# delete pod 'nginx'
kubectl delete po nginx

# edit the configuration of pod 'nginx'
kubectl edit po nginx
```

DEPLOYMENTS & REPLICASETS

```
# create deployment 'nginx' using image 'nginx'
kubectl create deploy nginx --image nginx

# create a deployment yaml file named deploy.yml
kubectl create deploy nginx \
--image nginx --dry-run=client \
-o yaml > deploy.yml

# scale deployment 'nginx' up to 5 replicas
kubectl scale deploy nginx --replicas=5

# edit deployment 'nginx'
kubectl edit deploy nginx

# list deployments in default namespace
kubectl get deploy

# list deployments in all namespaces
kubectl get deploy --all-namespaces

# list kubernetes resources in all namespaces
kubectl get all --all-namespaces

# list pods, nodes and services in all namespaces
kubectl get po,no,svc --all-namespaces

# same as above but get additional info
kubectl get deploy -o wide

# get yaml for deployments in default namespace
kubectl get deploy -o yaml

# describe deployments in default namespace
kubectl describe deploy

# delete deployment 'nginx'
kubectl delete deploy nginx

# list all replicasets in default namespace
kubectl get rs

# same as above but output more info
kubectl get rs -o wide

# output yaml for all replicasets in default namespace
kubectl get rs -o yaml

# describe all replicasets in default namespace
kubectl describe rs
```

SERVICES

```
# create service 'nodeport-svc' in default namespace
kubectl create svc nodeport nodeport-svc \
--tcp=8080:80

# create service 'app-svc' from deployment 'nginx'
kubectl expose deploy nginx --name=app-svc \
--port=80 --type=NodePort

# list services in default namespace
kubectl get svc

# same as above but additional info
kubectl get svc -o wide

# list all resources in all namespaces
kubectl get all --all-namespaces

# list pods, nodes and services in all namespaces
kubectl get po,no,svc --all-namespaces

# show yaml for services in default namespace
kubectl get svc -o yaml

# describe services in the default namespace
kubectl describe svc

# show labels for services in default namespace
kubectl get svc --show-labels

# edit service 'app-svc' in default namespace
kubectl edit svc app-svc

# delete service 'app-svc' in default namespace
kubectl delete svc app-svc
```

ROLES & SERVICE ACCOUNTS

```
# list all roles in 'kube-system' namespace
kubectl get roles -n kube-system

# output yaml for roles in 'kube-system' namespace
kubectl get roles -n kube-system -o yaml

# list all cluster roles
kubectl get clusterroles

# create role 'pod-reader' to get, watch and list pods
kubectl create role pod-reader --verb=get --verb=list
--verb=watch --resource=pods

# create clusterrole 'pod-reader' to get,watch,list pods
kubectl create clusterrole pod-reader \
--verb=get,list,watch --resource=pods

# give 'bob' permission in 'admin', in 'robot-shop' ns
kubectl create rolebinding bob-admin-binding \
--clusterrole=admin --user=bob --namespace=robot-shop

# grant 'bob' permissions in 'admin' ClusterRole
kubectl create clusterrolebinding \
root-cluster-admin-binding \
--clusterrole=admin --user=bob

# list all service accounts in default namespace
kubectl get sa

# view yaml for all service accounts in default namespace
kubectl get sa -o yaml

# output yaml for sa 'default' to a file named 'sa.yml'
kubectl get sa default -o yaml > sa.yml

# replace 'default' with file 'sa.yml'
kubectl replace sa default -f sa.yml

# edit service account 'default' in default namespace
kubectl edit sa default

# delete service account 'default' in default namespace
kubectl delete sa default
```

CONFIGMAPS & SECRETS

```
# list configmaps in default namespace
kubectl get cm

# list configmaps in all namespaces
kubectl get cm --all-namespaces

# output yaml for configmaps in all namespaces
kubectl get cm --all-namespaces -o yaml

# list secrets in default namespace
kubectl get secrets

# list secrets in all namespaces
kubectl get secrets --all-namespaces

# output yaml for secrets in all namespaces
kubectl get secrets --all-namespaces -o yaml
```

DAEMONSETS

```
# list daemonsets in default namespace
kubectl get ds

# list daemonsets in all namespaces
kubectl get ds --all-namespaces

# describe daemonset 'kube-proxy' in 'kube-system'
kubectl describe ds kube-proxy -n kube-system

# output yaml for 'kube-proxy' daemonset
kubectl get ds kube-proxy -n kube-system -o yaml

# edit daemonset 'kube-proxy' in 'kube-system'
kubectl edit ds kube-proxy -n kube-system

# edit daemonset 'kube-proxy' in 'kube-system'
kubectl edit ds kube-proxy -n kube-system

# delete daemonset 'kube-proxy' in 'kube-system'
kubectl delete ds kube-proxy -n kube-system
```

VOLUMES & STORAGE CLASS

```
# list persistent volumes in default namespace
kubectl get pv

# describe persistent volumes in default namespace
kubectl describe pv

# list all persisten volume claims in default namespace
kubectl get pvc

# describe persistent volume claims in default namespace
kubectl describe pvc

# list all storage class resources in default namespace
kubectl get svc

# output yaml for storage class in default namespace
kubectl get sc -o yaml
```