kubectl Kubernetes CLI

START HERE

`kubectl get pods`
List pods

`kubectl describe pod <pod>`
Details and events

`kubectl logs -f <pod>`
Follow logs

`kubectl apply -f <file>`
Create or update

`kubectl exec -it <pod> -- sh`
Shell in a pod

`kubectl config use-context <ctx>`
Switch cluster

`kubectl delete -f <file>`
Remove objects

`kubectl explain <resource>`
Field docs

CONCEPTS

Check the context and namespace before changing anything: the same command hits whichever cluster is current.

Cluster context
Namespace -n
Deployment desired state
Pod containers

Most commands accept TYPE NAME or TYPE/NAME, and -l <selector> to act on many objects. Short names: po, svc, deploy, ns, cm.

Declarative (apply -f) keeps manifests as the source of truth; imperative edits and patches drift from them.

CONTEXT

`kubectl config get-contexts`
List contexts

`kubectl config current-context`
Show active context

`kubectl config use-context <ctx>`
Switch context

`kubectl config set-context --current --namespace=<ns>`
Set default namespace
Saves you from typing -n on every command.

`kubectl config view`
Show merged kubeconfig
Add --raw to include credentials.

`kubectl cluster-info`
Show API server address

`kubectl version`
Client and server versions

`kubectl api-resources`
List resource types

`kubectl explain <resource>.spec`
Show field docs

`kubectl get pods -A`
All namespaces (-A)

CREATE & APPLY

`kubectl apply -f <file>`
Create or update from file

`kubectl apply -f <dir>`
Apply every manifest in a dir

`kubectl create deployment <name> --image=<img>`
New deployment

`kubectl run <name> --image=<img>`
Start a single pod

CREATE & APPLY (cont.)

`kubectl run <name> --image=<img> --dry-run=client -o yaml`
Generate YAML only
Redirect to a file to bootstrap a manifest.

`kubectl create job <name> --image=<img> -- <cmd>`
Run a one-off job

`kubectl create cronjob <name> --image=<img> --schedule="<cron>" -- <cmd>`
Scheduled job

`kubectl expose deployment <name> --port=<port>`
Create a service

`kubectl diff -f <file>`
Preview changes before apply

INSPECT

`kubectl get pods`
List pods

`kubectl get pods -o wide`
Add node and IP

`kubectl get pods -n <ns>`
List in a namespace

`kubectl get all`
Common resources in namespace
Not truly all: skips configmaps, secrets, ingresses and CRDs.

`kubectl get <type> <name> -o yaml`
Full object as YAML

`kubectl describe <type> <name>`
Details and recent events

`kubectl get pods -l <key>=<value>`
Filter by label

`kubectl get pods -w`
Watch for changes

`kubectl get events --sort-by=.metadata.creationTimestamp`
Events by time

`kubectl events --types=Warning`
Only warnings

`kubectl get pods --sort-by=.status.startTime`
Sort by a field

`kubectl get <type> <name> -o jsonpath='{.<path>}'`
Extract one field

`kubectl get pods -o custom-columns=NAME:.metadata.name,IMAGE:.spec.containers[*].image`
Custom table

`kubectl get <type> -o name`
Names only

PODS & DEBUG

`kubectl logs <pod>`
Show pod logs

`kubectl logs -f <pod>`
Follow logs

`kubectl logs <pod> -c <container>`
Logs of one container

`kubectl logs <pod> --previous`
Logs of crashed container

`kubectl logs -l <key>=<value> --all-containers`
Logs by label

`kubectl exec -it <pod> -- sh`
Interactive shell
Everything after -- is the command run in the container.

`kubectl exec <pod> -- <cmd>`
Run one command

`kubectl port-forward <pod> <local>:<remote>`
Forward a local port

`kubectl port-forward svc/<name> <local>:<port>`
Forward to a service

`kubectl cp <pod>:<path> <local>`
Copy file from pod
Needs tar inside the container image.

`kubectl cp <local> <pod>:<path>`
Copy file to pod

`kubectl debug <pod> -it --image=busybox:1.28`
Debug container in a pod

`kubectl debug node/<node> -it --image=busybox:1.28`
Debug on a node

`kubectl top pod`
Pod CPU and memory
Needs metrics-server in the cluster.

UPDATE & ROLLOUT

`kubectl set image deployment/<name> <container>=<img>`
Change an image

`kubectl rollout status deployment/<name>`
Wait for rollout

`kubectl rollout undo deployment/<name>`
Roll back

`kubectl rollout restart deployment/<name>`
Restart all pods

`kubectl scale deployment/<name> --replicas=<n>`
Set replica count

UPDATE & ROLLOUT (cont.)

`kubectl edit <type> <name>`
Edit in $EDITOR
KUBE_EDITOR overrides the editor.

`kubectl patch <type> <name> -p '<json>'`
Patch fields (strategic merge)

`kubectl label <type> <name> <key>=<value>`
Add or change a label
Append - to the key to remove it; add --overwrite to replace.

DELETE

`kubectl delete -f <file>`
Delete objects from file

`kubectl delete pod <pod>`
Delete a pod
A deployment will recreate it.

`kubectl delete pods -l <key>=<value>`
Delete by label

NODES

`kubectl get nodes`
List nodes

`kubectl top node`
Node CPU and memory

`kubectl cordon <node>`
Mark unschedulable

`kubectl drain <node> --ignore-daemonsets`
Evict pods from a node
Usually needs --ignore-daemonsets; add --delete-emptydir-data for emptyDir volumes.

`kubectl uncordon <node>`
Allow scheduling again

RECIPES

Roll out a new image

```
kubectl set image deployment/<name> <container>=<img>:v2
kubectl rollout status deployment/<name>
# if it goes wrong
kubectl rollout undo deployment/<name>
```

Debug a crashing pod

```
kubectl get pods
kubectl describe pod <pod>
kubectl logs <pod> --previous
kubectl get events --sort-by=.metadata.creationTimestamp
```

Draft a manifest

```
kubectl create deployment web --image=nginx \
  --dry-run=client -o yaml > web.yaml
kubectl apply -f web.yaml
```

Shell setup (bash)

```
source <(kubectl completion bash)
alias k=kubectl
complete -o default -F __start_kubectl k
```