kubectl_cheatsheet

kubectl is the command line client for Kubernetes: it talks to the cluster in your current context to inspect, create and change resources. These are the commands you reach for every day, grouped by task. Switch to the k alias and every command updates. Click a command to copy it.

:setup — 4 cmds

Shell completion and the k alias. Add these lines to ~/.bashrc to keep them.

Turn on bash completion

```
$ source <(kubectl completion bash)
```

Turn on zsh completion

```
$ source <(kubectl completion zsh)
```

Use k as an alias, with completion

```
$ alias k=kubectl
$ complete -o default -F __start_kubectl k
```

Show client and server versions

```
$ kubectl version
```

:config — 7 cmds

Clusters, contexts and the active namespace.

Show the merged kubeconfig

```
$ kubectl config view
```

List contexts

```
$ kubectl config get-contexts
```

Show the current context

```
$ kubectl config current-context
```

Switch to another context

```
$ kubectl config use-context my-cluster
```

Set the default namespace

```
$ kubectl config set-context --current --namespace=my-ns
```

Create a context for a namespace

```
$ kubectl config set-context dev --cluster=my-cluster --user=me --namespace=dev
```

Show cluster endpoints

```
$ kubectl cluster-info
```

:get — 14 cmds

List and inspect resources.

List pods in the current namespace

```
$ kubectl get pods
```

List pods in every namespace

```
$ kubectl get pods -A
```

Show node and IP for each pod

```
$ kubectl get pods -o wide
```

Watch pods change live

```
$ kubectl get pods -w
```

List pods by label

```
$ kubectl get pods -l app=web
```

List services, endpoints and ingresses

```
$ kubectl get svc,endpoints,ingress
```

List the common resources

```
$ kubectl get all
```

Print a resource as YAML or JSON

```
$ kubectl get pod my-pod -o yaml
$ kubectl get pod my-pod -o json
```

Print one field with JSONPath

```
$ kubectl get pods -o jsonpath='{.items[*].metadata.name}'
```

Print chosen columns

```
$ kubectl get pods -o custom-columns=NAME:.metadata.name,NODE:.spec.nodeName
```

Sort pods by age

```
$ kubectl get pods --sort-by=.metadata.creationTimestamp
```

List only running pods

```
$ kubectl get pods --field-selector=status.phase=Running
```

List every resource type

```
$ kubectl api-resources
```

Explain a field of a resource

```
$ kubectl explain pod.spec.containers
```

:pods

17 cmds

Logs, shells, port forwards and debugging.

Describe a pod and its events

```
$ kubectl describe pod my-pod
```

Print pod logs

```
$ kubectl logs my-pod
```

Stream pod logs

```
$ kubectl logs -f my-pod
```

Logs from the last hour, last 100 lines

```
$ kubectl logs my-pod --since=1h --tail=100
```

Logs from one container

```
$ kubectl logs my-pod -c app
```

Logs from the previous crash

```
$ kubectl logs my-pod --previous
```

Logs from every pod with a label

```
$ kubectl logs -l app=web --all-containers
```

Open a shell in a pod

```
$ kubectl exec -it my-pod -- sh
```

Forward a local port to a pod

```
$ kubectl port-forward pod/my-pod 8080:80
```

Forward a local port to a service

```
$ kubectl port-forward svc/web 8080:80
```

Copy a file out of a pod

```
$ kubectl cp my-pod:/tmp/dump.sql ./dump.sql
```

Start a throwaway debug pod

```
$ kubectl run debug --image=busybox -it --rm -- sh
```

Debug a running pod with an ephemeral container

```
$ kubectl debug -it my-pod --image=busybox --target=app
```

Wait until a pod is ready

```
$ kubectl wait --for=condition=Ready pod/my-pod --timeout=60s
```

Show pod CPU and memory (needs metrics-server)

```
$ kubectl top pod
```

Delete a pod

```
$ kubectl delete pod my-pod
```

Force delete a stuck pod

```
$ kubectl delete pod my-pod --grace-period=0 --force
```

:deploy

10 cmds

Scale, update and roll back workloads.

Create a deployment

```
$ kubectl create deployment web --image=nginx
```

Scale a deployment

```
$ kubectl scale deploy/web --replicas=3
```

Update the container image

```
$ kubectl set image deploy/web nginx=nginx:1.27
```

Watch a rollout finish

```
$ kubectl rollout status deploy/web
```

Show rollout history

```
$ kubectl rollout history deploy/web
```

Roll back to the previous version

```
$ kubectl rollout undo deploy/web
```

Roll back to a specific revision

```
$ kubectl rollout undo deploy/web --to-revision=2
```

Restart every pod in a deployment

```
$ kubectl rollout restart deploy/web
```

Autoscale on CPU

```
$ kubectl autoscale deploy/web --min=2 --max=10 --cpu-percent=80
```

Expose a deployment as a service

```
$ kubectl expose deploy/web --port=80 --target-port=80
```

:jobs

5 cmds

One-off and scheduled work.

Run a one-off job

```
$ kubectl create job hello --image=busybox -- echo hello
```

Create a CronJob

```
$ kubectl create cronjob nightly --image=busybox --schedule='0 3 * * *' -- echo hi
```

Run a CronJob now

```
$ kubectl create job nightly-manual --from=cronjob/nightly
```

List jobs and CronJobs

```
$ kubectl get jobs,cronjobs
```

Wait for a job to finish

```
$ kubectl wait --for=condition=complete job/hello --timeout=120s
```

:apply

9 cmds

Create and change resources from manifests.

Apply a manifest

```
$ kubectl apply -f app.yaml
```

Apply a kustomize directory

```
$ kubectl apply -k ./overlays/prod
```

Preview changes before applying

```
$ kubectl diff -f app.yaml
```

Delete what a manifest created

```
$ kubectl delete -f app.yaml
```

Edit a live resource

```
$ kubectl edit deploy/web
```

Generate YAML without creating

```
$ kubectl create deploy web --image=nginx --dry-run=client -o yaml > web.yaml
```

Add a label

```
$ kubectl label pod my-pod env=prod
```

Add an annotation

```
$ kubectl annotate deploy/web owner=team-a
```

Patch a field

```
$ kubectl patch deploy/web -p '{"spec":{"replicas":2}}'
```

:rbac

5 cmds

Check and grant permissions.

Check if you can do something

```
$ kubectl auth can-i create deployments
```

List everything you can do

```
$ kubectl auth can-i --list
```

Check as a service account

```
$ kubectl auth can-i get pods --as=system:serviceaccount:dev:app
```

Create a role

```
$ kubectl create role pod-reader --verb=get,list,watch --resource=pods
```

Bind a role to a user

```
$ kubectl create rolebinding read-pods --role=pod-reader --user=jane
```

:nodes

7 cmds

Cluster capacity and maintenance.

List nodes

```
$ kubectl get nodes
```

Show node CPU and memory (needs metrics-server)

```
$ kubectl top node
```

Stop scheduling on a node

```
$ kubectl cordon node-1
```

Evict pods before maintenance

```
$ kubectl drain node-1 --ignore-daemonsets
```

Allow scheduling again

```
$ kubectl uncordon node-1
```

Taint a node

```
$ kubectl taint nodes node-1 dedicated=gpu:NoSchedule
```

Recent events, newest last

```
$ kubectl get events --sort-by=.metadata.creationTimestamp
```

:secrets

5 cmds

Configuration and isolation.

Create a secret from a value

```
$ kubectl create secret generic db --from-literal=password=s3cret
```

Read and decode a secret

```
$ kubectl get secret db -o jsonpath='{.data.password}' | base64 -d
```

Create a ConfigMap from a file

```
$ kubectl create configmap app --from-file=config.yaml
```

Create a namespace

```
$ kubectl create namespace my-ns
```

List namespaces

```
$ kubectl get ns
```

:short 14 cmds

Type less. Every resource has a short name.

pods

`po`

services

`svc`

deployments

`deploy`

replicasets

`rs`

statefulsets

`sts`

daemonsets

`ds`

cronjobs

`cj`

namespaces

`ns`

nodes

`no`

configmaps

`cm`

persistentvolumeclaims

`pvc`

ingresses

`ing`

horizontalpodautoscalers

`hpa`

serviceaccounts

`sa`

Updated Nov 8, 2026 · checked against kubectl 1.30+ · sources: kubectl quick reference, kubectl command reference

Made by Lucas Bittencourt. Free, no ads, no tracking. Spotted a mistake? Tell me.