kubectl

Cheat Sheet

Kubernetes CLI Reference · minikube & production clusters

Free resource by **juanpablo.tech**

Sections: Viewing · Debugging · Manifests · Networking · Contexts · Flags · minikube

1 · Viewing Resources

Basic listing

```
# List pods in the app namespace
kubectl get pods -n app

# Watch pods in real time (Ctrl+C to stop)
kubectl get pods -n app -w

# See everything in a namespace at once
kubectl get all -n app

# More detail — node name, pod IP, etc.
kubectl get pods -n app -o wide
```

Output formats

```
# YAML — great for copying to a new manifest
kubectl get pod postgres-0 -n app -o yaml

# JSON — pipe to jq for filtering
kubectl get pod postgres-0 -n app -o json | jq '.status.phase'

# Custom columns — pick exactly what you need
kubectl get pods -n app -o custom-columns='NAME:.metadata.name,STATUS:.status.phase'

# List across all namespaces
kubectl get pods --all-namespaces
kubectl get pods -A          # shorthand
```

Filtering with labels

```
# Show pods matching a label
kubectl get pods -n app -l app=postgres

# Multiple label selectors
kubectl get pods -n app -l app=api,env=prod

# Show labels as columns
kubectl get pods -n app --show-labels
```

Quick reference | get targets

Command / Flag	What it does
`kubectl get pods`	List pods in current namespace
`kubectl get deployments`	List Deployments
`kubectl get statefulsets`	List StatefulSets
`kubectl get services`	List Services (ClusterIP, NodePort…)
`kubectl get configmaps`	List ConfigMaps
`kubectl get secrets`	List Secrets (values are base64)
`kubectl get pvc`	List PersistentVolumeClaims
`kubectl get nodes`	List cluster nodes and their status
`kubectl get events -n app`	Show recent events (great for debugging)

2 · Debugging Pods

Inspect & describe

```
# Full details: events, resource limits, volumes, image pull errors
kubectl describe pod postgres-0 -n app

# Describe a Deployment
kubectl describe deployment my-app -n app

# Recent events — first stop when something breaks
kubectl get events -n app --sort-by='.lastTimestamp'
```

Logs

```
# Tail live logs from a specific pod
kubectl logs -n app postgres-0 -f

# Logs from a Deployment (picks any matching pod)
kubectl logs -n app -l app=api -f

# Last 100 lines only
kubectl logs -n app postgres-0 --tail=100

# Logs from a crashed container (previous run)
kubectl logs -n app postgres-0 -p

# Multi-container pod — specify which container
kubectl logs -n app my-pod -c sidecar-container
```

Shell access

```
# Open bash inside a running pod
kubectl exec -it postgres-0 -n app -- bash

# Use sh if bash isn't available (alpine images)
kubectl exec -it postgres-0 -n app -- sh

# Run a one-off command without interactive shell
kubectl exec -n app postgres-0 -- env | grep POSTGRES

# Copy a file out of a pod
kubectl cp app/postgres-0:/var/lib/postgresql/data/pg_hba.conf ./pg_hba.conf
```

Pod status (common states)

Command / Flag	What it does
`Running`	All containers up and healthy
`Pending`	Waiting for a node — check events for resource/PVC issues
`CrashLoopBackOff`	Container keeps crashing — check logs -p for previous run
`ImagePullBackOff`	Image name wrong or registry unreachable
`OOMKilled`	Container exceeded memory limit — increase resources
`Terminating (stuck)`	Finalizer blocking — use --force --grace-period=0

3 · Applying & Managing Manifests

Apply / create / delete

```
# Apply a manifest (create or update)
kubectl apply -f postgres.yaml

# Apply everything in a directory
kubectl apply -f ./k8s/

# Delete resources defined in a file
kubectl delete -f postgres.yaml

# Delete a specific resource by name
kubectl delete pod postgres-0 -n app

# Force-delete a stuck pod (use with care)
kubectl delete pod postgres-0 -n app --force --grace-period=0
```

Rollouts & scaling

```
# Check rollout status
kubectl rollout status deployment/my-app -n app

# Rollback to previous version
kubectl rollout undo deployment/my-app -n app

# Scale a deployment
kubectl scale deployment my-app --replicas=3 -n app

# Rolling restart — cycles all pods without downtime
kubectl rollout restart deployment/my-app -n app
```

Quick edits

```
# Edit a live resource in your $EDITOR
kubectl edit deployment my-app -n app

# Patch a specific field (JSON merge patch)
kubectl patch deployment my-app -n app -p '{"spec":{"replicas":2}}'

# Set a new image on a deployment
kubectl set image deployment/my-app app=my-image:v2 -n app
```

4 · Networking & Port Forwarding

Port forwarding — access services locally

```
# Forward local port 5432 → postgres pod port 5432
kubectl port-forward pod/postgres-0 5432:5432 -n app

# Forward via Service (survives pod restarts)
kubectl port-forward svc/postgres 5432:5432 -n app

# Forward Kubernetes Dashboard
kubectl port-forward svc/kubernetes-dashboard 8443:443 -n kubernetes-dashboard
```

Services & DNS

```
# List services and their ports
kubectl get svc -n app

# Describe a service (shows endpoints, selectors)
kubectl describe svc postgres -n app

# DNS name pattern inside the cluster:
# <service>.<namespace>.svc.cluster.local
# e.g.: postgres.app.svc.cluster.local

# Run a temp pod to test DNS / connectivity
kubectl run tmp --image=busybox -it --rm -- nslookup postgres.app.svc.cluster.local
```

5 · Context & Namespaces

Managing contexts (clusters)

```
# Show all contexts
kubectl config get-contexts

# Switch to minikube
kubectl config use-context minikube

# Show current context
kubectl config current-context

# Set default namespace — no more -n on every command
kubectl config set-context --current --namespace=app
```

Namespaces

```
# List all namespaces
kubectl get namespaces

# Create a namespace
kubectl create namespace staging

# Delete a namespace (deletes EVERYTHING inside it!)
kubectl delete namespace staging
```

6 · Useful Flags Reference

Command / Flag	What it does
`-n <namespace>`	Target a specific namespace
`-A / --all-namespaces`	Query across all namespaces
`-o wide`	Extra columns: node, IPs, nominated node
`-o yaml`	Output as YAML (useful for diffing / saving)
`-o json`	Output as JSON — pipe to jq for filtering
`-w / --watch`	Stream resource updates in real time
`-f / --follow`	Follow log stream (kubectl logs)
`-p / --previous`	Logs from previous container run
`--tail=N`	Show last N log lines only
`-l key=value`	Filter resources by label selector
`--dry-run=client -o yaml`	Preview what would be created (no apply)
`--force --grace-period=0`	Force-delete stuck or terminating resources
`-it`	Interactive TTY — for exec / run commands
`--rm`	Auto-delete pod when command exits (run)

7 · minikube Quick Reference

Cluster lifecycle

```
minikube start                  # Start cluster (Docker driver by default)
minikube stop                   # Suspend cluster
minikube delete                 # Wipe cluster entirely
minikube status                 # Health check
minikube dashboard              # Open Kubernetes Dashboard in browser
minikube ip                     # Get the IP of your minikube node
minikube ssh                    # SSH into the node
```

Build images directly into minikube

Skips the registry push step — image is available to the cluster immediately.

```
# Point your shell at minikube's Docker daemon
eval $(minikube docker-env)

# Build — the image lands inside minikube, not your local Docker
docker build -t k8s-lab-app:latest .

# Restore your local Docker when done
eval $(minikube docker-env -u)
```

Enable addons

```
minikube addons enable metrics-server
minikube addons enable dashboard
minikube addons enable ingress
```

⚠️ After eval $(minikube docker-env), all docker commands target minikube's daemon — not your local Docker. Run eval $(minikube docker-env -u) to restore your local context.

Happy shipping! ☸

More free resources at juanpablo.tech