

# Table of Contents

[**Table of Contents 2**](#_y11wb7y15e5v)

[**🔧 Setup & Info 4**](#_e3b8r5ow6kqw)

[Info 4](#_dgifjack9k7d)

[Context & Cluster Management 4](#_cuqqye8kjg3d)

[**🏷️ Namespaces 5**](#_rxu52x50ybbm)

[Basic Namespace Operations 5](#_txpdijgir3ba)

[**🟢 Pods 5**](#_6kjhum14bntf)

[Pod Management 5](#_i92m2dim03ke)

[Pod Logs 5](#_ebnf1vkegdbq)

[Pod Execution & Access 6](#_ijhwmbylhvn)

[Port Forwarding 6](#_31iz81dnxnok)

[Watch & Monitor Pods 7](#_zb3kew9qpp4)

[**🚀 Deployments 7**](#_m9swcuo2qgay)

[Deployment Management 7](#_9gchco5nqxmv)

[**🌐 Services 8**](#_51bwt54a3191)

[Service Management 8](#_xp1oihhsivzf)

[**📦 ConfigMaps & Secrets 9**](#_gqgrdnvuj1n9)

[ConfigMaps 9](#_c8440k3q0fxr)

[Secrets 9](#_5kvgfvt28lwq)

[**🔍 Debugging & Troubleshooting 10**](#_7lde8ue828w8)

[Events & Logs 10](#_qv761fvizqzv)

[Temporary Debugging Pod 10](#_90i7eba11v4x)

[**📊 Resource Management 11**](#_942z14rmjwzo)

[Resource Quotas & Limits 11](#_98mm39wq93s4)

[Horizontal Pod Autoscaler (HPA) 11](#_ze55mvtjo3bf)

[**🛡️ Security & RBAC 12**](#_va55qzohiv0u)

[Service Accounts 12](#_5r4gyxyld7qj)

[RBAC 12](#_8ltchrk84hul)

[**📋 Working with YAML Manifests 12**](#_vk6rx93r12ed)

[Apply & Create 12](#_srvhra11xknz)

[Generate & Export 13](#_xft1uwhm5up1)

[**🔄 Advanced Operations 13**](#_v5b86smngu29)

[Labels & Selectors 13](#_k2hphe5rcxee)

[Annotations 14](#_5o9v5sa9bm84)

[Patch Operations 14](#_yie1hk9ww4cb)

[**📃Output Formats 15**](#_rewzjf2bn5i7)

[**📖 Resource Shortnames 15**](#_57myqpcflfmp)

[**🆘 Emergency Commands 16**](#_yzy39thb4y50)

[Force Operations 16](#_o3qth8macjgc)

[Recovery Operations 16](#_sz92ksby5548)

[**💡Additional Tips 17**](#_3u3gngqu3ptu)

[**Wrap-up & Resources 18**](#_h7mtmdhyqj96)

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 🔧 Setup & Info

## Info

| # Check kubectl version kubectl version --client # Get cluster info kubectl cluster-info # Get cluster info dump kubectl cluster-info dump # Get API versions kubectl api-versions # Get API resources kubectl api-resources # Get API resources with short names kubectl api-resources --namespaced=false |
| --- |

## Context & Cluster Management

| # Get current context kubectl config current-context # List all contexts kubectl config get-contexts # Switch context kubectl config use-context <context-name> # Set default namespace for current context kubectl config set-context --current --namespace=<namespace> # View kubeconfig kubectl config view # Merge kubeconfig files KUBECONFIG=~/.kube/config:~/.kube/config2 kubectl config view --merge --flatten > ~/.kube/merged\_config |
| --- |

# 

# 🏷️ Namespaces

## Basic Namespace Operations

| # List all namespaces kubectl get namespaces kubectl get ns # Create namespace kubectl create namespace <namespace-name> # Delete namespace kubectl delete namespace <namespace-name> # Set default namespace for current context kubectl config set-context --current --namespace=<namespace> # Get resources from all namespaces kubectl get pods --all-namespaces kubectl get pods -A # shorthand |
| --- |

# 🟢 Pods

## Pod Management

| # List pods in current namespace kubectl get pods # List pods with more details kubectl get pods -o wide # List pods with labels kubectl get pods --show-labels # List pods in specific namespace kubectl get pods -n <namespace> # Get pod details kubectl describe pod <pod-name> # Get pod YAML kubectl get pod <pod-name> -o yaml |
| --- |

## Pod Logs

| # Get pod logs kubectl logs <pod-name> # Follow logs (tail -f equivalent) kubectl logs <pod-name> -f # Get logs from specific container in multi-container pod kubectl logs <pod-name> -c <container-name> # Get previous container logs (useful for crashed containers) kubectl logs <pod-name> --previous # Get logs with timestamps kubectl logs <pod-name> --timestamps # Get logs from last 1 hour kubectl logs <pod-name> --since=1h # Get last 100 lines of logs kubectl logs <pod-name> --tail=100 |
| --- |

## Pod Execution & Access

| # Execute command in pod (single container) kubectl exec <pod-name> -- <command> # Interactive shell into pod kubectl exec -it <pod-name> -- /bin/bash kubectl exec -it <pod-name> -- /bin/sh # if bash not available # Execute command in specific container kubectl exec -it <pod-name> -c <container-name> -- /bin/bash # Copy files from/to pod kubectl cp <pod-name>:/path/to/file /local/path kubectl cp /local/path <pod-name>:/path/to/file # Copy files from specific container kubectl cp <pod-name>:/path/to/file /local/path -c <container-name> |
| --- |

## Port Forwarding

| # Forward local port to pod port kubectl port-forward <pod-name> <local-port>:<pod-port> # Forward to service kubectl port-forward service/<service-name> <local-port>:<service-port> # Forward with specific address binding kubectl port-forward --address 0.0.0.0 <pod-name> <local-port>:<pod-port> # Forward multiple ports kubectl port-forward <pod-name> <local-port1>:<pod-port1> <local-port2>:<pod-port2> |
| --- |

## Watch & Monitor Pods

| # Watch pods in real-time kubectl get pods -w # Watch pods with output refreshing every 2 seconds kubectl get pods -w --output-watch-events # Get pod resource usage (requires metrics-server - Metrics API) kubectl top pods # Get pod resource usage for specific namespace kubectl top pods -n <namespace> # Get pod resource usage for all namespaces kubectl top pods --all-namespaces |
| --- |

# 🚀 Deployments

## Deployment Management

| # List deployments kubectl get deployments kubectl get deploy # shorthand # Get deployment details kubectl describe deployment <deployment-name> # Create deployment from image kubectl create deployment <deployment-name> --image=<image-name> # Scale deployment kubectl scale deployment <deployment-name> --replicas=<number> # Update deployment image kubectl set image deployment/<deployment-name> <container-name>=<new-image> # Rollout status kubectl rollout status deployment/<deployment-name> # Rollout history kubectl rollout history deployment/<deployment-name> # Rollback deployment kubectl rollout undo deployment/<deployment-name> # Rollback to specific revision kubectl rollout undo deployment/<deployment-name> --to-revision=<revision-number> # Restart deployment (rolling restart) kubectl rollout restart deployment/<deployment-name> |
| --- |

# 🌐 Services

## Service Management

| # List services kubectl get services kubectl get svc # shorthand # Get service details kubectl describe service <service-name> # Expose deployment as service kubectl expose deployment <deployment-name> --port=<port> --type=<service-type> # Create service from YAML kubectl apply -f service.yaml # Delete service kubectl delete service <service-name> # Get service endpoints kubectl get endpoints <service-name> |
| --- |

# 

# 📦 ConfigMaps & Secrets

## ConfigMaps

| # List configmaps kubectl get configmaps kubectl get cm # shorthand # Create configmap from literal values kubectl create configmap <configmap-name> --from-literal=<key>=<value> # Create configmap from file kubectl create configmap <configmap-name> --from-file=<file-path> # Create configmap from directory kubectl create configmap <configmap-name> --from-file=<directory-path> # Get configmap data kubectl get configmap <configmap-name> -o yaml # Edit configmap kubectl edit configmap <configmap-name> |
| --- |

## Secrets

| # List secrets kubectl get secrets # Create secret from literal values kubectl create secret generic <secret-name> --from-literal=<key>=<value> # Create secret from file kubectl create secret generic <secret-name> --from-file=<file-path> # Create TLS secret kubectl create secret tls <secret-name> --cert=<cert-file> --key=<key-file> # Get secret data (base64 encoded) kubectl get secret <secret-name> -o yaml # Decode secret value kubectl get secret <secret-name> -o jsonpath='{.data.<key>}' | base64 --decode |
| --- |

# 🔍 Debugging & Troubleshooting

| Resource Information # Get all resources in namespace kubectl get all # Get all resources in all namespaces kubectl get all --all-namespaces # Describe any resource kubectl describe <resource-type> <resource-name> # Get resource in JSON format kubectl get <resource-type> <resource-name> -o json # Get resource with custom columns kubectl get pods -o custom-columns=NAME:.metadata.name,STATUS:.status.phase,NODE:.spec.nodeName |
| --- |

## Events & Logs

| # Get events in current namespace kubectl get events # Get events sorted by time kubectl get events --sort-by=.metadata.creationTimestamp # Get events for specific resource kubectl get events --field-selector involvedObject.name=<resource-name> # Watch events kubectl get events -w |
| --- |

## Temporary Debugging Pod

| # Run temporary pod for debugging kubectl run debug-pod --image=busybox --rm -it --restart=Never -- /bin/sh # Run temporary pod in specific namespace kubectl run debug-pod --image=busybox --rm -it --restart=Never -n <namespace> -- /bin/sh # Run temporary pod with specific service account kubectl run debug-pod --image=busybox --rm -it --restart=Never --serviceaccount=<sa-name> -- /bin/sh |
| --- |

# 

# 📊 Resource Management

## Resource Quotas & Limits

| # Get resource quotas kubectl get resourcequotas kubectl get quota # shorthand # Get limit ranges kubectl get limitranges kubectl get limits # shorthand # Get node resource usage kubectl top nodes (Requires Metrics API) # Get pod resource requests and limits kubectl describe pods <pod-name> | grep -A 5 "Requests\|Limits" |
| --- |

## Horizontal Pod Autoscaler (HPA)

| # List HPAs kubectl get hpa # Create HPA kubectl autoscale deployment <deployment-name> --cpu-percent=80 --min=1 --max=10 # Get HPA details kubectl describe hpa <hpa-name> |
| --- |

# 

# 🛡️ Security & RBAC

## Service Accounts

| # List service accounts kubectl get serviceaccounts kubectl get sa # shorthand # Create service account kubectl create serviceaccount <sa-name> # Get service account token kubectl get serviceaccount <sa-name> -o yaml |
| --- |

## RBAC

| # List roles kubectl get roles # List cluster roles kubectl get clusterroles # List role bindings kubectl get rolebindings # List cluster role bindings kubectl get clusterrolebindings # Check permissions for current user kubectl auth can-i <verb> <resource> # Check permissions for specific user kubectl auth can-i <verb> <resource> --as=<user> # Check permissions for service account kubectl auth can-i <verb> <resource> --as=system:serviceaccount:<namespace>:<sa-name> |
| --- |

# 📋 Working with YAML Manifests

## Apply & Create

| # Apply configuration from file kubectl apply -f <file.yaml> # Apply all YAML files in directory kubectl apply -f <directory>/ # Apply from URL kubectl apply -f https://example.com/manifest.yaml # Create resource from file (fails if is already exists) kubectl create -f <file.yaml> # Dry run to validate YAML kubectl apply -f <file.yaml> --dry-run=client # Server-side dry run kubectl apply -f <file.yaml> --dry-run=server |
| --- |

## Generate & Export

| # Generate YAML for deployment kubectl create deployment <name> --image=<image> --dry-run=client -o yaml # Generate YAML for service kubectl expose deployment <deployment-name> --port=80 --dry-run=client -o yaml # Export existing resource to YAML kubectl get deployment <deployment-name> -o yaml --export > deployment.yaml # Generate manifest template kubectl run <pod-name> --image=<image> --dry-run=client -o yaml > pod.yaml |
| --- |

# 

# 🔄 Advanced Operations

## Labels & Selectors

| # Add label to resource kubectl label pods <pod-name> <label-key>=<label-value> # Remove label from resource kubectl label pods <pod-name> <label-key>- # Select resources by label kubectl get pods -l <label-key>=<label-value> # Select resources by multiple labels kubectl get pods -l <label-key1>=<label-value1>,<label-key2>=<label-value2> # Select resources with label exists kubectl get pods -l <label-key> # Select resources with label not equal kubectl get pods -l <label-key>!=<label-value> |
| --- |

## Annotations

| # Add annotation to resource kubectl annotate pods <pod-name> <annotation-key>=<annotation-value> # Remove annotation from resource kubectl annotate pods <pod-name> <annotation-key>- |
| --- |

## Patch Operations

| # Patch resource with strategic merge kubectl patch deployment <deployment-name> -p '{"spec":{"replicas":5}}' # Patch resource with JSON merge kubectl patch deployment <deployment-name> --type merge -p '{"spec":{"replicas":5}}' # Patch resource with JSON patch kubectl patch deployment <deployment-name> --type json -p='[{"op": "replace", "path": "/spec/replicas", "value": 5}]' |
| --- |

# 

# 

# 

# 📃Output Formats

| # Wide output (more columns) kubectl get pods -o wide # JSON output kubectl get pods -o json # YAML output kubectl get pods -o yaml # LYAML output (requires Kubernetes 1.34+) kubectl get pods -o kyaml # Custom columns kubectl get pods -o custom-columns=NAME:.metadata.name,STATUS:.status.phase # JSONPath output kubectl get pods -o jsonpath='{.items[\*].metadata.name}' # Go template output kubectl get pods -o go-template='{{range .items}}{{.metadata.name}}{{"\n"}}{{end}}' |
| --- |

# 📖 Resource Shortnames

| po # pods svc # services deploy # deployments rs # replicasets ds # daemonsets sts # statefulsets cm # configmaps sa # serviceaccounts ns # namespaces no # nodes pv # persistentvolumes pvc # persistentvolumeclaims ing # ingresses netpol # networkpolicies |
| --- |

# 🆘 Emergency Commands

## Force Operations

| # Force delete pod (when stuck in terminating) kubectl delete pod <pod-name> --grace-period=0 --force  # Force delete stuck pod  kubectl patch pod <pod>-p '{"metadata":{"finalizers":null}}' # Force delete namespace (when stuck in terminating) kubectl delete namespace <namespace-name> --grace-period=0 --force # Force delete all pods in namespace kubectl delete pods --all --grace-period=0 --force -n <namespace> |
| --- |

## Recovery Operations

| # Cordon node (mark as unschedulable) kubectl cordon <node-name> # Uncordon node kubectl uncordon <node-name> # Drain node (evict all pods) kubectl drain <node-name> --ignore-daemonsets --delete-emptydir-data # Restart all pods in deployment (rolling restart) kubectl rollout restart deployment/<deployment-name> # Scale deployment to 0 and back kubectl scale deployment <deployment-name> --replicas=0 kubectl scale deployment <deployment-name> --replicas=3 |
| --- |

# 

# 

# 

# 💡Additional Tips

Here are some extra tips you can follow to get even smoother experience with kubectl:

* Use kubectl explain <resource> to get detailed information about resource fields.
* Use kubectl diff -f <file.yaml> to see what changes will be applied.
* Use kubectl get events --sort-by='.lastTimestamp' for chronological events.
* Use kubectl logs -l <label-selector> to get logs from multiple pods
* Set export KUBE\_EDITOR=nano or vim for your favorite editor when using kubectl edit\*

You can also add these aliases to ~/.bashrc:

| alias k='kubectl' alias kgp='kubectl get pods' alias kgs='kubectl get svc' alias kgd='kubectl get deployment' alias kdp='kubectl describe pod' alias kds='kubectl describe svc' alias kdd='kubectl describe deployment' alias kaf='kubectl apply -f' alias kdel='kubectl delete' alias klog='kubectl logs' alias kex='kubectl exec -it' alias kpf='kubectl port-forward' |
| --- |

Type source ~/.bashrc to activate the aliases.

# 

# 

# Wrap-up & Resources

In this document, you got a list of the most popular kubectl commands you can use everyday.

If you’re just starting out with Kubernetes, and want to learn the fundamentals, check out my ebook, [**Master Kubernetes from Scratch**](https://curiousdevscorner.gumroad.com/l/kubernetes), which is suitable for beginners.