Cheatography

Kubernetes - CKAD Cheat Sheet

by Mansour Javaher (mansourj) via cheatography.com/153462/cs/32989/

Helm

helm repo add <repo-name> <repo-url>	Adds a new Helm chart repository to your local setup.
helm repo add bitnami https://charts.bitnami.com/bitnami	Adds the Bitnami Helm repository.
helm repo list	Lists all configured Helm repositories.
helm repo update	Updates the local cache of Helm repositories to fetch the latest charts.
helm search repo <chart-name>	Searches for a Helm chart in the locally added repositories.
helm search repo bitnami	Lists all charts available in the Bitnami repository.
helm search hub <app-name>	Searches for a Helm chart across multiple repositories in Helm Hub.
helm install <release-name> <repo-name>/<chart-name>	Deploys a Helm chart from a repository into Kubernetes.
helm install my-wordpress bitnami/wordpress	Installs the WordPress chart from Bitnami into Kubernetes.
helm uninstall my-release	Removes the specified Helm release from Kubernetes.
helm env	Displays Helm-related environment variables.
helm install my-release ./my-chart	Installs a Helm chart from a local directory.
helm pull <chart-name> --untar	Downloads and extracts a Helm chart without installing it.

General commands

alias k=kubectl	
k api-resources	List all resource types in the cluster
k explain node	Introspect documentation for a resource
k explain node.spec	Dive deeper into a resource specification
k explain node.spes | less	
kubectl explain --recursive pod.spec	

Pods

k get po	List pods
k get po -o wide	Include additional details
k get po <pod name> -o yaml	Output a specific pod in YAML format
k get po <pod-name> -o yaml > pod.yaml	Output a specific pod in YAML to a file
k get pods -n <pod-name>	Pods in a specific namespace
k get pods --all-namespaces	List pods across all namespaces
kubectl get pods -A	Short version of `--all- nam espaces`
kubectl get events -A | grep error	All events in all namespaces with errors
k get pod --show-labels	Show labels for pods
k get pod -l <key>=<value>	Filter pods by label
k describe po <pod name>	Get details about a pod
k delete po <pod name>	Delete a specific pod
k delete po --all	Delete all pods in the namespace
k apply -f <pod.yaml> -n <namespace-name>	Apply pod configuration
k edit po <pod-name>	Edit a pod
k run nginx --image=nginx --dry-run=client -o yaml > pod.yaml	Generate pod YAML imperatively
k exec [pod-name] -- [command]	Execute a command in a pod
k exec nginx -- ls /	Execute ls command in the root /
k exec -it nginx -- /bin/sh	Get a shell in the container
k get pods --no-headers | wc -l	Get number of running pods
k exec ubuntu-sleeper -- whoami	

Key Notes:
- Pods can have initContainers for pre-task operations.
- Use restartPolicy (Always, OnFailure, Never) in the pod spec.

By **Mansour Javaher** (mansourj)
cheatography.com/mansourj/
www.mansour.co.nz/

Published 26th November, 2024.
Last updated 23rd February, 2025.

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com

Replica set

k apply -f <replicaset-definition.yaml>	Create/Update a ReplicaSet
k get rs	List ReplicaSets
k get rs -o wide	Get ReplicaSet details
k delete rs <replicaset-name>	Delete a ReplicaSet
k scale rs <replicaset-name> --replicas=6	Scale replicas
k edit rs <replicaset-name>	Edit a ReplicaSet
k describe rs <replicaset-name>	Get details about a ReplicaSet
k delete po <rs-po-name>	Delete all pods under ReplicaSet
k get rs <rs-name> -o yaml > rs.yaml	Export a ReplicaSet to a YAML file

Editing ReplicaSet
- Either delete and re-create the ReplicaSet or
- Update the existing ReplicaSet and then delete all PODs, so new ones with the correct image will be created.

The value for labels in spec.selector clause and spec.template.metadata should match in replicaset.

ConfigMap

k create cm <cm-name> --from-literal=key=value	
k create cm <cm-name> --from-file=<path>	
k describe cm <cm-name>	Read config map detals

ConfigMaps in Kubernetes
ConfigMaps can be injected into Pods as:
- Environment variables: Use `config Map KeyRef` (for specific keys) or `config MapRef`/`envFrom` (for the whole ConfigMap).
- Files in a volume: Mount the ConfigMap as a volume for your application to read.

Secret

k get secrets	
k get secret --all-namespaces	
k describe secret <secret-name>	This shows the attributes in secret but hides the values.
k get secret <secret-name> -o yaml	To view the values (values are encoded).
k create secret generic <secret-name> --from-literal=<key1>=<value1> --from-literal=<key2>=<value2>	
k create secret generic <secret-name> --from-file=<path to file>	
echo -n 'string' | base64	How to decode to base64
echo -n 'encoded string' | base64 --decode	how to decode a base64 string

Taints (on nodes) and Tolerations (on pods)

k taint nodes [node_name] <key>=<value>:[NoSchedule|NoExecute|Prefer-NoSchedule]	Taint effect defines what happens to pods that do not tolerate this taint.
k taint no node01 spray=mortein:NoSchedule	
k describe nodes node01 | grep -i "taint"	To check taints on a node

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com

Taints (on nodes) and Tolerations (on pods) (cont)

kubectl taint nodes master node-role.kubernetes.io/master:No-Schedule-	Remove the taint on the master node that has the NoSchedule effect
kubectl taint nodes foo dedicated-	Remove all taints with the key dedicated from the node named foo: bash Copy code

Tolerations in the spec section have properties like key, operator, value, and effect. Their values are written inside quotation marks ("").

Horizontal Pod Auto Scaler

k get hpa

k delete hpa <hpa-name>

k autoscale deploy nginx --min=5 --max=10 --cpu-percent=80

Monitoring

k top no	Resource usage for nodes
k top po	Resource usage for pods

k top pod --namespace=default | head -2 | tail -1 | cut -d " " -f1

k top po --sort-by cpu --no-headers

Events

k get events

k get events -n kube-system

k get events -w

DaemonSets

k get ds

k get ds --all-namespaces

k describe ds <ds-name> -n <ns-name>

k describe ds <ds-name> -o yaml

Environment Variables

k run nginx --image=nginx --env=app=web	nginx pod and with an ENV set to app=web

env and envFrom property is an array.
env property takes two properties - name and value. value takes only string and will always come in double quotes.

Service

k apply -f <svc.yaml>

k get svc

k get svc --show-labels

k get svc -o wide

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com

Service (cont)

k get svc -o yaml	
k describe svc <service-name>	To get to know about port, target port etc.
k delete svc <service-name>	
k expose deploy <deploy-name> --port=80 --type=NodePort	
k expose deploy <deploy-name> --port=80 --type=NodePort --dry-run=client -o yaml > service.yml	

Node

k get no	
k get no -o wide	
k describe no <node-name>	

Network Policy

k get netpol	
k describe netpol <name>	

Note: While creating network policy, make sure that not only network policy is applied to the correct object but also that it allows access from (ingress) / to correct object (egress).

labels and selectors are used.

An empty podSelector selects all pods in the namespace.

Annotations

k annotate po nginx desc="Hello World"	
k annotate po nginx author=Avnish	
k annotate po nginx desc-	Remove this annotation from the pod
k annotate no <node-name>	

Labels & Selectors

k get [pod|deploy|all] --show-labels	Show labels
k get label [node|pod|deploy|etc] <key>=<value>	Syntax
k label pod nginx env=lab	To label pod
k label deploy my-webapp tier=frontend	To label deployment
k label node node01 size=large	To label nodes
k label po nginx <key>-	Remove the label
k label po nginx env=lab1 --overwrite	To overwrite a label
k get po --selector=app=App1	
k get po --selector=app!=App1	
k get all --selector=env=prod	
k get po --selector=env=prod,bu=finance,tier=frontend	Equivalent of && in programming languages

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com

Volumes

k get pv	
k describe pv	
k delete pv <pv-name>	
k get pvc	
k describe pvc	
k delete pvc <pvc-name>	

Ingress

k get ingress	
k describe ingress <ingress name>	
k edit ingress <ingress name>	
k apply -f <ingress.yaml>	

Logging

k logs -f <pod-name> <container-name>	Follow the logs
k logs <pod-name> --previous	Dump pod logs for a previous instantiation of a container
k logs --tail=20 <pod-name>	

Deployments

k create deploy nginx --image=nginx --replicas=2	
k get deploy	
k get deploy -n <namespace-name>	
k get deploy <deployment-name>	
k get deploy <deployment-name> -o yaml | more	
k get deploy -o wide	
k describe deploy <deployment-name>	
k apply -f deploy.yaml	
k apply -f deploy.yaml --record	Record the change-cause in revision history
k rollout status deploy <deployment-name>	
k rollout history deploy <deployment-name>	
k rollout history deploy nginx --revision=2	to get detailed history for a specific revision.T
k rollout undo deploy <deployment-name>	
k rollout undo deploy <deployment-name> --to-revision=1	
k rollout pause deploy <deployment-name>	
k rollout resume deploy <deployment-name>	
k delete deploy <deployment-name>	
k create deploy nginx --image=nginx --dry-run=client -o yaml > deploy.yaml	

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com

Deployments (cont)

k scale deploy <deployment-name> --replicas=3	To scale up / down a deployment. Not recorded in revision history.
k create deploy <deployment-name> --image=redis -n <namespace-name>	
k edit deploy <deployment-name> -n <namespace-name>	
k expose deploy <deployment-name> --port=80 --type=NodePort|ClusterIp	
k autoscale deployment <deployment-name> --max 6 --min 3 --cpu-percent 50	Autoscale deployment

Deployments can be paused and resumed. When paused, no changes are recorded to revision history.

RollingUpdateStrategy in define upto how many pods can be down/up during the update at a time.
spec.strategy.type==RollingUpdate|Recreate
spec.strategy.rollingUpdate.maxUnavailable
spec.strategy.rollingUpdate.maxSurge

Updating images or new deployements, triggers rollout. A new rollout creates a new deployment revision.

Service Account

k create sa <sa-name>	
k get sa	
k get sa <sa-name> -o yaml	
k describe sa <sa-name>	Fetch token: gives secret name
k describe secret <secret-name>	Fetch token: gives token stored in secret
k run nginx --image=nginx --serviceaccount=myuser --dry-run=client -o yaml > pod.yaml	nginx pod that uses 'myuser' as a service account

When we use service account inside the pod, the secret for that service account is mounted as volume inside the pod.

Property to remember: spec -> serviceAccountName # set at pod level
Injected into the Pod.
For a deployment, can set service account in pod template.

A user makes a request to API server through k using user account.
A process running inside a container makes a request to API server using service account.
A service account just like user account has certain permissions.

CronJobs

k get cj	
k create cj busybox --image=busybox --schedule="*/1 *" -- /bin/sh -c "date; echo Hello from Kubernetes cluster"	cron job with image busybox that runs on a schedule and writes to standard output

In a cronjob, there are 2 templates - one for job and another for pod.

In a cronjob, there are 3 spec sections - one for cronjob, one for job and one for pod (in order).

Properties to remember: spec -> successfulJobHistoryLimit, spec -> failedJobHistoryLimit

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com

Jobs

k create job busybox --image=busybox -- /bin/sh -c "echo hello;sleep 30;echo world"	
k get jobs	
k logs busybox-qhcnx	pod under job
k delete job <job-name>	

restartPolicy defaults to Never.

Job has 2 spec sections - one for job and one for pod (in order).

restartPolicy has to be OnFailure or Never. If the restartPolicy is OnFailure, a failed container will be re-run on the same pod. If the restartPolicy is Never, a failed container will be re-run on a new pod.

Job properties to remember: completions, backoffLimit, parallelism, activeDeadlineSeconds, restartPolicy.

By default Pods in a job are created in sequence

Namespace

k get ns	
k get ns -o yaml	
k create ns <namespace-name>	
k apply -f <namespace.yaml>	
k get po -n kube-system	
k describe ns <namespace-name>	
k delete ns <namespace-name>	
k config set-context $(k config current-context) --namespace=dev	To switch to a namespace permanently
k run redis --image=redis -n <namespace-name>	Create/run in a specific ns
k exec -it <pod-name> -n <namespace-name> -- sh	
k create ns <namespace-name> --dry-run=client -o yaml	

By **Mansour Javaher** (mansourj)
cheatography.com/mansourj/
www.mansour.co.nz/

Published 26th November, 2024.
Last updated 23rd February, 2025.

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com