mimacom

Kubernetes Cheat Sheet

MIMACOM
Digital. Innovation. Engineers.

Kubernetes, is an open-source system for automating deployment, scaling, and management of containerized applications. It is built for planet scale to run millions of containers, while it's flexibility allows customization to meet any needs so in fact, Kubernetes can run anywhere from on-premises to hybrid or public cloud infrastructure.

AUTHENTICATION

A context connects to a K8s cluster with a specific user.

```
kubectl config get-contexts
```
View all available and configured contexts

```
kubectl config current-context
```
Show the context that is currently being used

```
kubectl config use-context <context-name>
```
Switch to another context and use it for subsequent commands

```
kubectl config view
```
View the merged configuration file

CREATING RESOURCES

Create resources declaratively with a manifest.

```
kubectl apply -f <manifest.yml>
```
Create and update resources to match definition in manifest file.

```
kubectl apply -k <directory>
```
Apply a folder using Kustomize configuration management.

```
kubectl create job <name> --image=busybox -
- echo 'Hello mimacom!'
```
Imperatively create a resource, in this case a job.

```
kubectl create secret generic <name> --from-
literal=<key>=<value>
```
Create a secret from the CLI. Do not store secrets in manifest files.

EXPLORING RESOURCE TYPES

Every cluster may have different resource types.

```
kubectl api-resources
```
Retrieve a list of all available resource types of the cluster.

```
kubectl explain <type>
```
Get the documentation for the resource type.

```
kubectl explain <type>.<property>
```
Get documentation for a specific property.

DELETING RESOURCES

Resources that are deleted are usually gone forever.

```
kubectl delete <type> <name>
```
Delete a specific resource of type identified by its name.

```
kubectl delete -f <manifest.yml>
```
Delete all resources mentioned in the manifest file.

EDITING RESOURCES

For a quick test you can manually edit resources.

```
kubectl edit <type> <name>
```
Edit a specific resource and sync any changes to the cluster.

```
KUBE_EDITOR="nano"
```
Set the editor to use to your liking.

VIEW AND FIND RESOURCES

Explore the resource objects available.

```
kubectl get <type>
```
Get all resources of a type in current namespace, e.g. pods.

```
kubectl get <type> -A
```
Get all resources of a type across all namespaces.

```
kubectl get <type> <name>
```
Get condensed info on a specific resource by name.

```
kubectl describe <type> <name>
```
Get verbose info on a specific resource by name.

```
kubectl get <type> <name> -o yaml
```
Get a specific resource in its complete YAML representation.

```
kubectl get <type> --show-labels
```
Get all resource along with the labels defined on each resource.

```
kubectl get <type> -l <label>=<value>
```
Show only resources that have label set to value.

OPERATING APPLICATIONS

Rollout and scale your applications.

`kubectl rollout history <type> <name>`	View the deployment history of a specific deployment
`kubectl rollout undo <type> <name>`	Rollback to the previous deployment.
`kubectl rollout restart <type> <name>`	Rolling restart of a resource.
`kubectl scale --replicas=3 <type> <name>`	Scale a resource to a number of replicas.

TROUBLESHOOTING

When things go south start investigating.

`kubectl get <type> <name>`	If you identify a faulty resource, start by investigating its details.
`kubectl logs <pod>`	Get logs of a specific pod.
`kubectl logs <pod> -f`	Stream and follow the logs in real-time.
`kubectl logs <pod> --previous`	See the log output of the previous container, if still available.
`kubectl exec <pod> -- curl localhost:8080`	Execute a command on a pod.
`kubectl exec <pod> -it -- /bin/sh`	Start an interactive shell on the pod.
`kubectl port-forward <pod> 8080:6000`	Port-forwarding from a pod to your local machine.
`kubectl get events`	View cluster/namespace events.
`kubectl top pods`	View CPU and memory usage for all pods.

GENERAL OPTIONS

These general options are useful from time to time.

`kubectl --namespace <namespace>`	Apply the command to a specific namespace.
`kubectl --v=<0..9>`	Increase log output to even see HTTP requests starting at level 7.

EXAMPLE MANIFESTS

Use these example manifests as quick reference.

```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mima-deployment
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
  selector:
    matchLabels: {'app': 'mima-app'}
  template:
    metadata:
      labels: {'app': 'mima-app'}
    spec:
      containers:
      - name: mima-app-container
        image: ...
        ports:
        - containerPort: 3000
        resources:
          limits:
            memory: "256M"
            cpu: "500m"
        livenessProbe:
          httpGet:
            port: 3000
            path: '/actuator/info'
```

A deployment manifest with some important properties set.

```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: mima-app-service
spec:
  type: ClusterIP
  ports:
  - port: 3000
  selector: {'app': 'mima-app'}
```

A service exposing the above deployment.