Must to know Kubectl commands for the CKAD

NB : Once the aliases are set you can replace for every command the word **Kubectl** with **K**

- Easily check the format of a yaml section within a K8S object :
  - `Kubectl explain pods --recursive | grep envFrom -A3`
  - `Kubectl explain pods --recursive | less`

- List all K8S objects within the cluster :
  - `Kubectl get all -A`

POD

- Create a pod the fastest way :
  - `Kubectl run nginx --image=nginx --namespace=dev -l app=front`

- Generate a pod and save its definition to a yaml file :
  - `Kubectl run nginx --image=nginx -o yaml > filename.yaml`

- Generate a pod yaml file definition without creating the pod :
  - `Kubectl run nginx --image=nginx --dry-run=client -o yaml > filename.yaml`

- If no yaml file is given for a specific object, we need to extract its definition into a new file using the following command :
  - `Kubectl get pod <pod-name> -o yaml > pod-definition.yaml`
  - Then we can edit the yaml, delete the existing pod and finally create a new one from the edited file.

- Create or edit a pod properties using the yam file :
  - `Kubectl apply -f pod-definition.yaml`

- Edit pod properties we can use directly this commande :
  - `Kubectl edit pod <pod-name>`

- Delete an existing pod :
  - `Kubectl delete pod <pod-name>`

- Delete a pod immediately :
  - `Kubectl delete pod <pod-name> –grace-period=0 –force`

- Another command to create a pod from a yaml file :
  - `Kubectl create -f /tmp/<file-name>.yaml`

- Delete all pods :
  - `Kubectl delete pod --all`
  - Don’t forget to specify a NS if not this will delete all pods in all NS

ReplicaSet

- Scale a pod using a new defined replicaset :
  - `Kubectl scale --replicas=4 -f replicaset-definition.yaml`
  - `Kubectl scale --replicas=5 rs replicas-set`

Deployment

- Creating a deployment the fastest way :
  - `Kubectl create deployment --image=nginx nginx  --replicas=4`

- Only generating the yaml file without creating the deployment :
  - `Kubectl create deployment --image=nginx nginx --dry-run -o yaml`
  - To save the latest def to a yaml file just add **`> deployment.yaml`** to the end of the command

- Scale a deployment :
  - `Kubectl scale deployment.v1.apps/nginx-deployment --replicas=4`

- Edit deployment :
  - `Kubectl edit deployment my-deployment`

NameSpace

- Getting a pod within a specific namespace :
  - `Kubectl get pods --namespace=dev`
- Changing the default namespace to dev :
  - `Kubectl config set-context --current --ns=dev`
- Getting all pod in within all namespaces :
  - `Kubectl get pods --all-namespaces`

Service

- Fastest way to create a pod and exposing it on a specific port :
  - `Kubectl run custom-nginx --image=nginx --port=8080 --expose`
- Exposing an existing pod on port 444 with using "frontend" service :
  - `Kubectl expose pod valid-pod --port=444 --name=frontend`
  - `Kubectl expose deployment <name-deployment> --name=service-name --target-port=8080 --port=80 --type=NodePort --dry-run=client -o yaml > service.yaml`

ConfigMap

- Creating a ConfigMap from literal :
  - `Kubectl create configmap <config-name> --from-literal=<key>=<value> --from-literal=<key>=<value>`
- Creating ConfigMap from a file :
  - `Kubectl create configmap <config-name> --from-file=<file-name>`

- Creating a ConfigMap a declarative way :
  - `Kubectl create -f config-map.yaml`

Secrets

Same as configmap the only diff is that the data need hashed

- Display the value of the secret :
  - `Kubectl get secret app-secret -o yaml`

- Code/Decode secret values :
  - `echo -n "password" | base64`
  - `echo -n "jhk/=" | base64 --decode`

Logging

- Show a pod logs :
  - `Kubectl logs -f <pod-name>`

- If the pod is having more than 1 container then :
  - `Kubectl logs -f <pod-name> <container-name>`