Kubernetes Cheat Sheet

Kubernetes manages clusters of containers, providing tools for deploying and scaling applications, managing changes to containerized applications, and optimizing the use of underlying hardware.

kubectl create

Create a resource (pod, service, node, job, and so on) referred to by YAML or JSON file, or by name.

`create deployment foo --image=<image>`	Deploy pod **foo** based on **image**
`create job foo \` `--image busybox -- echo "example"`	Create job **foo** from image **busybox** to echo "example" once

`create cronjob foo \`
`--image=busybox \`
`--schedule="*/1 * * * *" \`
`-- date`
	Create cronjob **foo** from image **busybox** to print the date every minute

`create node --register-node=false f.json`	Create a node from foo.json
`expose deployment foo \` `--type=LoadBalancer --name=foo-service`	Create a service to expose **foo**

kubectl run

`run --stdin --tty foo \` `--image=busybox -- /bin/sh`	Start a pod called foo with an interactive shell, based on the **busybox** image
`run foo --image=busybox \` `--namespace=foospace`	Start **foo** in **foospace** namespace

Interacting

`attach foo -i`	Attach to running container **foo**
`exec --stdin --tty foo -- /bin/sh`	Open a shell in **foo**
`exec foo -- ls`	Run a single command in **foo**
`exec foo -c bar -- ls`	Run the **ls** command in **bar** (in pod **foo**)

Supported by
Red Hat

Kubernetes Cheat Sheet

Scaling

Images serve as blueprints for any number of replicas, which can be created as needed.

`scale deployment foo --replicas=3`	Scale deployment **foo** to 5
`autoscale deployment foo \` `--max 6 --min 3 --cpu-percent 50`	Autoscale deployment **foo**

kubectl get

Get information about a resource.

`get services`	List all services in current namespace
`get services --sort-by=metadata.name`	List all services, sorted by name
`get pv`	List persistent volumes
`get pods --all-namespaces`	List all pods in all namespaces
`get pods \` `--fieldselector=status.phase=Running`	Get all running pods in current namespace
`get deployment foo`	List deployment **foo**
`describe deployment foo`	Get verbose information about **foo**
`logs foo`	Dump logs for pod **foo** to stdout

kubectl delete

`delete pod foo`	Delete the pod named **foo**
`--namespace=foospace delete pod,svc --all`	Delete all pods and services in namespace **foo**