**Namespaces**

Kubernetes namespaces by example

Namespaces provide for a scope of Kubernetes resource, carving up your cluster in smaller units. You can think of it as a workspace you’re sharing with other users. Many resources such as pods and services are namespaced, while some, for example, nodes are not namespaced (but cluster-wide). As a developer you’d usually use an assigned namespace, however admins may wish to manage them, for example to set up access control or resource quotas.

Let’s list all namespaces (note that the output will depend on the environment you’re using,

$ kubectl get ns

NAME STATUS AGE

default Active 13d

kube-system Active 13d

namingthings Active 12d

openshift Active 13d

openshift-infra Active 13d

You can learn more about a namespace using the describe verb, for example:

$ kubectl describe ns default

Name: default

Labels: <none>

Status: Active

No resource quota.

No resource limits.

Let’s now create a new [namespace](https://github.com/openshift-evangelists/kbe/blob/master/specs/ns/ns.yaml) called test now:

$ kubectl apply -f https://raw.githubusercontent.com/openshift-evangelists/kbe/master/specs/ns/ns.yaml

namespace "test" created

$ kubectl get ns

NAME STATUS AGE

default Active 13d

kube-system Active 13d

namingthings Active 12d

openshift Active 13d

openshift-infra Active 13d

**test** Active 3s

Alternatively, we can could have created the namespace using the kubectl create namespace test command.

To launch a [pod](https://github.com/openshift-evangelists/kbe/blob/master/specs/ns/pod.yaml) in the newly created namespace test, do:

$ kubectl apply --namespace=**test** -f https://raw.githubusercontent.com/openshift-evangelists/kbe/master/specs/ns/pod.yaml

Note that using above method the namespace becomes a runtime property, that is, you can deploy the same pod or service, etc. into multiple namespaces (for example: dev and prod). Hard-coding the namespace directly in the metadata section like shown in the following is possible but causes less flexibility when deploying your apps:

apiVersion: v1

kind: Pod

metadata:

name: podintest

namespace: **test**

To list namespaced objects such as our pod podintest, run the following command:

$ kubectl get pods --namespace=**test**

NAME READY STATUS RESTARTS AGE

podintest 1/1 Running 0 16s

You can remove the namespace (and everything inside) with:

$ kubectl delete ns **test**

If you’re an admin, you might want to check out the [docs](https://kubernetes.io/docs/tasks/administer-cluster/namespaces/) for more info how to handle namespaces.