**Labels**

Kubernetes labels by example

Labels are the mechanism you use to organize Kubernetes objects. A label is a key-value pair with certain [restrictions](https://kubernetes.io/docs/concepts/overview/working-with-objects/labels/#syntax-and-character-set) concerning length and allowed values but without any pre-defined meaning. So you’re free to choose labels as you see fit, for example, to express environments such as ‘this pod is running in production’ or ownership, like ‘department X owns that pod’.

Let’s create a [pod](https://github.com/openshift-evangelists/kbe/blob/master/specs/labels/pod.yaml) that initially has one label (env=development):

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

$ kubectl get pods --show-labels

NAME READY STATUS RESTARTS AGE LABELS

labelex 1/1 Running 0 10m env=development

In above get pods command note the --show-labels option that output the labels of an object in an additional column.

You can add a label to the pod as:

$ kubectl label pods labelex owner=michael

$ kubectl get pods --show-labels

NAME READY STATUS RESTARTS AGE LABELS

labelex 1/1 Running 0 16m env=development,owner=michael

To use a label for filtering, for example to list only pods that have an owner that equals michael, use the --selector option:

$ kubectl get pods --selector owner=michael

NAME READY STATUS RESTARTS AGE

labelex 1/1 Running 0 27m

The --selector option can be abbreviated to -l, so to select pods that are labelled with env=development, do:

$ kubectl get pods -l env=development

NAME READY STATUS RESTARTS AGE

labelex 1/1 Running 0 27m

Oftentimes, Kubernetes objects also support set-based selectors. Let’s launch [another pod](https://github.com/openshift-evangelists/kbe/blob/master/specs/labels/anotherpod.yaml) that has two labels (env=production and owner=michael):

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

Now, let’s list all pods that are either labelled with env=development or with env=production:

$ kubectl get pods -l 'env in (production, development)'

NAME READY STATUS RESTARTS AGE

labelex 1/1 Running 0 43m

labelexother 1/1 Running 0 3m

Other verbs also support label selection, for example, you could remove both of these pods with:

$ kubectl delete pods -l 'env in (production, development)'

Beware that this will destroy any pods with those labels.

You can also delete them directly, via their names, with:

$ kubectl delete pods labelex

$ kubectl delete pods labelexother

Note that labels are not restricted to pods. In fact you can apply them to all sorts of objects, such as nodes or services.