kubernetes

LABELS & ANNOTATIONS

❖ LABELS AND SELECTORS:

➢ LABELS:

- Labels are key/value pairs that are attached to objects, such as pods.
- It is used to specify identifying attributes of objects that are meaningful and relevant.
- Labels can be used to organize and to select subsets of objects.

```
"metadata": {
  "labels": {
    "key1" : "value1",
    "key2" : "value2"
  }
}
```

- These are examples of commonly used labels.

```
"release" : "stable" , "release" : "canary"

"environment" : "dev" , "environment" : "qa" , "environment" : "production"

"tier" : "frontend" , "tier" : "backend" , "tier" : "cache"

"partition" : "customerA" , "partition" : "customerB"

"track" : "daily" , "track" : "weekly"
```

EXAMPLE: Configuration file for a Pod that has two labels environment: production and app: nginx:

```
apiVersion: v1
kind: Pod
metadata:
  name: label-demo
  labels:
    environment: production
    app: nginx
spec:
  containers:
  - name: nginx
    image: nginx:1.14.2
    ports:
    - containerPort: 80
```

➢ LABEL SELECTORS:

- Unlike names and UIDs, labels do not provide uniqueness. In general, we expect many objects to carry the same label(s).
- Via a label selector, the client/user can identify a set of objects. The label selector is the core grouping primitive in Kubernetes.
- It supports two types of selectors: **equality-based and set-based.**

EQUALITY-BASED REQUIREMENT:

- Equality- or inequality-based requirements allow filtering by label keys and values. Matching objects must satisfy all of the specified label constraints, though they may have additional labels as well.
- Three kinds of operators are admitted **=, ==, !=** .

```
environment = production
tier != frontend
```

$kubectl get pods --show-labels

$kubectl get pods -l environment=production

$kubectl get pods -l 'environment in (production)'

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

$kubectl get pods -l environment!=production

SET-BASED REQUIREMENT:

- Set-based label requirements allow filtering keys according to a set of values. Three kinds of
- operators are supported: in, notin and exists (only the key identifier).

environment in (production, qa)
environment notin (backend, frontend)
partition

$kubectl get pods -l environment=production,tier=frontend
$kubectl get pods -l 'environment in (production),tier in (frontend)'
$kubectl get pods -l 'environment in (production, qa)'
$kubectl get pods -l 'environment,environment notin (frontend)'

❖ ANNOTATIONS:

- Annotations are also key-value pairs for connecting non-identifying metadata with objects.
- These are not used to identify and select objects.

```
"metadata": {
  "annotations": {
    "key1" : "value1",
    "key2" : "value2"
  }
}
```

NOTE:

- The keys and the values in the map must be strings. In other words, you cannot use numeric, boolean, list or other types for either the keys or the values

EXAMPLE: The configuration file for a Pod that has the annotation imageregistry: https://hub.docker.com/:

```
apiVersion: v1
kind: Pod
metadata:
  name: annotations-demo
  annotations:
    imageregistry: "https://hub.docker.com/"
spec:
  containers:
  - name: nginx
    image: nginx:1.14.2
    ports:
    - containerPort: 80
```

$kubectl get po

$kubectl describe pod annotations-demo

$kubectl delete pod annotations-demo