****

**Label and Selectors**

**Selectors is Kubernetes Object used to select the pod by using specific labels**

**Labels**

Labels are key-value pairs which are attached to pods, replication controller and services.

They are used as identifying attributes for objects such as pods and replication controller.

They can be added to an object at creation time and can be added or modified at the run time.

### Labels and Selectors

Labels are often reffered to as tags in order to differentiate more easily the Pods running in our system. Because you might run some thousands of Pods in your system, so it makes sense to apply some common naming scheme to know how you can sort the information that is available. All that a label does is to give us a easy-readable plain text to something we can refer to later. It is a key value that we can also define in our YAML file, and we can also use it later to get or set information.

**Selectors**

Labels do not provide uniqueness. In general, we can say many objects can carry the same labels. Labels selector are core grouping primitive in Kubernetes. They are used by the users to select a set of objects.

Label Selectors represent queries that are made against those labels. They resolve to the corresponding matching objects and will show when we are managing our Pods in our cluster how we use the built-in API and tools for Kubernetes in order to get a selection of objects based on these label selectors.

These two items (Labels and Selectors) are the primary way that grouping is done in Kubernetes and determine which components that a given operation applies to when indicated.

Kubernetes API currently supports two type of selectors −

* Equality-based selectors
* Set-based selectors

**Equality-based Selectors**

They allow filtering by key and value. Matching objects should satisfy all the specified labels.

**Set-based Selectors**

Set-based selectors allow filtering of keys according to a set of values.

apiVersion: v1

kind: Service

metadata:

name: tomcatservice

spec:

ports:

- port: 8081

name: testenv

type: NodePort

selector:

app: webserver ---------> 1

env: testenv -----------> 2

In the above code, we are using the label selector as **app: webserver** and env as **env: testenv**.

Once we run the file using the **kubectl** command, it will create a service with the name tomcatservice which will communicate on port 8081. The yype is **NodePort** with the new label selector as **app: webserver** and **env: testenv**.