

Module 3b: Kubernetes Objects

Kubernetes objects are persistent entities in the Kubernetes system that represent the state of your cluster. They describe what containerized applications are running (and on which nodes), the resources available to those applications, and the policies for how they operate. This lab will delve into three specific types of Kubernetes objects: **namespaces** for organizing resources, **jobs** for running finite tasks, and **cronjobs** for scheduling recurring tasks.



There are 2 ways to create a kubernetes object:

1. Ad-hoc using kubectl create
2. Declarative approach using a file. kubectl applycreate -f FILE

Namespaces

01

Create two namespaces: testing and dummy

```
kubectl create namespace testing  
kubectl create ns dummy
```

02

List all the namespaces

```
kubectl get ns
```

03

Find out which is the current namespace for the user

```
kubectl config get-contexts
```

04

Change it to testing namespace

```
kubectl config set-contexts --current --namespace  
testing  
kubectl config get-contexts
```

05

Try listing pods from kube-system namespace

Why can't you see the pods in your namespace?

```
kubectl get po  
kubectl get po -n kube-system
```

06

Delete the dummy namespace

```
kubectl delete ns dummy  
kubectl get ns
```

Jobs

We will be creating the job using the declarative approach.

1

Create a job that will end in a few seconds

Make sure the container name is lazy.

```
kubectl create job short --image=busybox --dry-run=client -o yaml -- sleep 3 > short.yaml
vi short.yaml
...
spec:
  template:
    spec:
      containers:
      - command:
        - sleep
        - "3"
        image: busybox
        name: lazy # Change this line
...
```

```
kubectl create -f short.yaml
```

2

Check the status of the job and pod

```
kubectl describe job short
kubectl get job,po
```

3

Set the job to run 5 times and 2 in parallel each time

Make sure the job does not take 15 seconds to finish.

```
vi short.yaml
...
kind: Job
name: short
spec:
  completions: 5      # Add this line
  parallelism: 2      # Add this line
  activeDeadlineSeconds: 15 # Add this line
  template:
    spec:
    ...
```

4

Open another terminal and monitor live events

```
kubectl get events -w
```

5

Back in the first terminal, recreate the job

```
kubectl delete job short
kubectl apply -f short.yaml
```

6

Monitor the events in the second terminal

Type the following commands in the first terminal:

```
kubectl get job,po
```

7

Clean up

Delete the job and exit the live event monitoring.

```
kubectl delete job short
Ctrl + C # exit the kubectl get events -w
```

CronJobs



Step 1

Create a cronjob using the 3s-cronjob.yaml file. Correct any errors encountered in the file.

```
kubectl apply -f 3s-  
cronjob.yaml
```

Step 2


Monitor the job. The job will only run after every 2 minutes.

```
kubectl describe cronjob  
repeat  
kubectl get cj,job,po
```

Step 3

Clean up. Delete the CronJob.

```
kubectl delete cj repeat
```

 **Note:** CronJobs in Kubernetes follow the standard cron format for scheduling. The job will execute at the specified intervals and create new pods for each run.