



Exercise 6.3: Working with Jobs

While most API objects are deployed such that they continue to be available there are some which we may want to run a particular number of times called a Job, and others on a regular basis called a CronJob

Create A Job

1. Create a job which will run a container which sleeps for three seconds then stops.

```
student@cp:~$ vim job.yaml
```

YAML

job.yaml

```
1 apiVersion: batch/v1
2 kind: Job
3 metadata:
4   name: sleepy
5 spec:
6   template:
7     spec:
8       containers:
9       - name: resting
10         image: busybox
11         command: ["/bin/sleep"]
12         args: ["3"]
13         restartPolicy: Never
```

2. Create the job, then verify and view the details. The example shows checking the job three seconds in and then again after it has completed. You may see different output depending on how fast you type.

```
student@cp:~$ kubectl create -f job.yaml
```

```
1 job.batch/sleepy created
```

```
student@cp:~$ kubectl get job
```

```
1 NAME      COMPLETIONS  DURATION  AGE
2 sleepy    0/1          3s        3s
```

```
student@cp:~$ kubectl describe jobs.batch sleepy
```

```
1 Name:      sleepy
2 Namespace: default
3 Selector:   controller-uid=24c91245-d0fb-11e8-947a-42010a800002
4 Labels:     controller-uid=24c91245-d0fb-11e8-947a-42010a800002
5             job-name=sleepy
6 Annotations: <none>
7 Parallelism: 1
8 Completions: 1
9 Start Time:  Tue, 16 Oct 2018 04:22:50 +0000
10 Completed At: Tue, 16 Oct 2018 04:22:55 +0000
11 Duration:    5s
12 Pods Statuses: 0 Running / 1 Succeeded / 0 Failed
13 <output_omitted>
```

```
student@cp:~$ kubectl get job
```

```
1 NAME      COMPLETIONS  DURATION  AGE
2 sleepy    1/1           5s        17s
```

3. View the configuration information of the job. There are three parameters we can use to affect how the job runs. Use **-o yaml** to see these parameters. We can see that backoffLimit, completions, and the parallelism. We'll add these parameters next.

```
student@cp:~$ kubectl get jobs.batch sleepy -o yaml
```

```
1 <output_omitted>
2   uid: c2c3a80d-d0fc-11e8-947a-42010a800002
3 spec:
4   backoffLimit: 6
5   completions: 1
6   parallelism: 1
7   selector:
8     matchLabels:
9 <output_omitted>
```

4. As the job continues to AGE in a completion state, delete the job.

```
student@cp:~$ kubectl delete jobs.batch sleepy
```

```
1 job.batch "sleepy" deleted
```

5. Edit the YAML and add the completions: parameter and set it to 5.

```
student@cp:~$ vim job.yaml
```

YAML

job.yaml

```
1 <output_omitted>
2 metadata:
3   name: sleepy
4 spec:
5   completions: 5  #<--Add this line
6   template:
7     spec:
8       containers:
9 <output_omitted>
```

6. Create the job again. As you view the job note that COMPLETIONS begins as zero of 5.

```
student@cp:~$ kubectl create -f job.yaml
```

```
1 job.batch/sleepy created
```

```
student@cp:~$ kubectl get jobs.batch
```

```
1 NAME      COMPLETIONS  DURATION  AGE
2 sleepy    0/5           5s        5s
```

7. View the pods that running. Again the output may be different depending on the speed of typing.

```
student@cp:~$ kubectl get pods
```

```

1 NAME                READY  STATUS      RESTARTS  AGE
2 sleepy-z5tnh         0/1   Completed   0          8s
3 sleepy-zd692         1/1   Running     0          3s
4 <output_omitted>

```

8. Eventually all the jobs will have completed. Verify then delete the job.

```
student@cp:~$ kubectl get jobs
```

```

1 NAME      COMPLETIONS  DURATION  AGE
2 sleepy    5/5          26s      10m

```

```
student@cp:~$ kubectl delete jobs.batch sleepy
```

```
1 job.batch "sleepy" deleted
```

9. Edit the YAML again. This time add in the `parallelism:` parameter. Set it to **2** such that two pods at a time will be deployed.

```
student@cp:~$ vim job.yaml
```

YAML

job.yaml

```

1 <output_omitted>
2   name: sleepy
3   spec:
4     completions: 5
5     parallelism: 2  #<-- Add this line
6     template:
7       spec:
8 <output_omitted>

```

10. Create the job again. You should see the pods deployed two at a time until all five have completed.

```
student@cp:~$ kubectl create -f job.yaml
```

```
1 job.batch/sleepy created
```

```
student@cp:~$ kubectl get pods
```

```

1 NAME                READY  STATUS      RESTARTS  AGE
2 sleepy-8xwpc        1/1   Running     0          5s
3 sleepy-xjqnf        1/1   Running     0          5s
4 <output_omitted>

```

```
student@cp:~$ kubectl get jobs
```

```

1 NAME      COMPLETIONS  DURATION  AGE
2 sleepy    3/5          11s      11s

```

11. Add a parameter which will stop the job after a certain number of seconds. Set the `activeDeadlineSeconds:` to 15. The job and all pods will end once it runs for 15 seconds. We will also increase the sleep argument to five, just to be sure does not expire by itself.

```
student@cp:~$ vim job.yaml
```

YAML

```

1 <output_omitted>
2   completions: 5
3   parallelism: 2
4   activeDeadlineSeconds: 15   #<-- Add this line
5   template:
6     spec:
7       containers:
8         - name: resting
9           image: busybox
10          command: ["/bin/sleep"]
11          args: ["5"]           #<-- Edit this line
12 <output_omitted>

```

12. Delete and recreate the job again. It should run for 15 seconds, usually 3/5, then continue to age without further completions.

```
student@cp:~$ kubectl delete jobs.batch sleepy
```

```
1 job.batch "sleepy" deleted
```

```
student@cp:~$ kubectl create -f job.yaml
```

```
1 job.batch/sleepy created
```

```
student@cp:~$ kubectl get jobs
```

```

1 NAME      COMPLETIONS  DURATION  AGE
2 sleepy    1/5           6s         6s

```

```
student@cp:~$ kubectl get jobs
```

```

1 NAME      COMPLETIONS  DURATION  AGE
2 sleepy    3/5           16s        16s

```

13. View the message: entry in the Status section of the object YAML output.

```
student@cp:~$ kubectl get job sleepy -o yaml
```

```

1 <output_omitted>
2 status:
3   conditions:
4     - lastProbeTime: 2018-10-16T05:45:14Z
5       lastTransitionTime: 2018-10-16T05:45:14Z
6       message: Job was active longer than specified deadline
7       reason: DeadlineExceeded
8       status: "True"
9       type: Failed
10  failed: 2
11  startTime: 2018-10-16T05:44:59Z
12  succeeded: 3

```

14. Delete the job.

```
student@cp:~$ kubectl delete jobs.batch sleepy
```

```
1 job.batch "sleepy" deleted
```

Create a CronJob

A CronJob creates a watch loop which will create a batch job on your behalf when the time becomes true. We Will use our existing Job file to start.

1. Copy the Job file to a new file.

```
student@cp:~$ cp job.yaml cronjob.yaml
```

2. Edit the file to look like the annotated file shown below. Edit the lines mentioned below. The three parameters we added will need to be removed. Other lines will need to be further indented.

```
student@cp:~$ vim cronjob.yaml
```

YAML

```
1 apiVersion: batch/v1beta1      #<-- Add beta1 to be v1beta1
2 kind: CronJob                  #<-- Update this line to CronJob
3 metadata:
4   name: sleepy
5 spec:
6   schedule: "*/2 * * * *"      #<-- Add Linux style cronjob syntax
7   jobTemplate:                  #<-- New jobTemplate and spec move
8     spec:
9       template:                 #<-- This and following lines move
10        spec:                    #<-- four spaces to the right
11          containers:
12            - name: resting
13              image: busybox
14              command: ["/bin/sleep"]
15              args: ["5"]
16          restartPolicy: Never
```

3. Create the new CronJob. View the jobs. It will take two minutes for the CronJob to run and generate a new batch Job.

```
student@cp:~$ kubectl create -f cronjob.yaml
```

```
1 cronjob.batch/sleepy created
```

```
student@cp:~$ kubectl get cronjobs.batch
```

```
1 NAME      SCHEDULE    SUSPEND   ACTIVE   LAST SCHEDULE   AGE
2 sleepy    */2 * * * *   False     0        <none>          8s
```

```
student@cp:~$ kubectl get jobs.batch
```

```
1 No resources found.
```

4. After two minutes you should see jobs start to run.

```
student@cp:~$ kubectl get cronjobs.batch
```

```
1 NAME      SCHEDULE    SUSPEND   ACTIVE   LAST SCHEDULE   AGE
2 sleepy    */2 * * * *   False     0        21s            2m1s
```

```
student@cp:~$ kubectl get jobs.batch
```

```
1 NAME                COMPLETIONS   DURATION   AGE
2 sleepy-1539722040    1/1           5s         18s
```

```
student@cp:~$ kubectl get jobs.batch
```

| | NAME | COMPLETIONS | DURATION | AGE |
|---|-------------------|-------------|----------|-------|
| 1 | sleepy-1539722040 | 1/1 | 5s | 5m17s |
| 2 | sleepy-1539722160 | 1/1 | 6s | 3m17s |
| 3 | sleepy-1539722280 | 1/1 | 6s | 77s |

5. Ensure that if the job continues for more than 10 seconds it is terminated. We will first edit the **sleep** command to run for 30 seconds then add the `activeDeadlineSeconds`: entry to the container.

```
student@cp:~$ vim cronjob.yaml
```

YAML

```
1 ....
2   jobTemplate:
3     spec:
4       template:
5         spec:
6           activeDeadlineSeconds: 10 #<-- Add this line
7           containers:
8             - name: resting
9             ....
10          command: ["/bin/sleep"]
11          args: ["30"] #<-- Edit this line
12          restartPolicy: Never
13 ....
```

6. Delete and recreate the CronJob. It may take a couple of minutes for the batch Job to be created and terminate due to the timer.

```
student@cp:~$ kubectl delete cronjobs.batch sleepy
```

```
1 cronjob.batch "sleepy" deleted
```

```
student@cp:~$ kubectl create -f cronjob.yaml
```

```
1 cronjob.batch/sleepy created
```

```
student@cp:~$ kubectl get jobs
```

| | NAME | COMPLETIONS | DURATION | AGE |
|---|-------------------|-------------|----------|-----|
| 1 | sleepy-1539723240 | 0/1 | 61s | 61s |

```
student@cp:~$ kubectl get cronjobs.batch
```

| | NAME | SCHEDULE | SUSPEND | ACTIVE | LAST SCHEDULE | AGE |
|---|--------|-------------|---------|--------|---------------|-----|
| 1 | sleepy | */* * * * * | False | 1 | 72s | 94s |

```
student@cp:~$ kubectl get jobs
```

| | NAME | COMPLETIONS | DURATION | AGE |
|---|-------------------|-------------|----------|-----|
| 1 | sleepy-1539723240 | 0/1 | 75s | 75s |

```
student@cp:~$ kubectl get jobs
```

| | NAME | COMPLETIONS | DURATION | AGE |
|---|-------------------|-------------|----------|-------|
| 1 | sleepy-1539723240 | 0/1 | 2m19s | 2m19s |
| 2 | sleepy-1539723360 | 0/1 | 19s | 19s |

```
student@cp:~$ kubectl get cronjobs.batch
```

| 1 | NAME | SCHEDULE | SUSPEND | ACTIVE | LAST SCHEDULE | AGE |
|---|--------|-------------|---------|--------|---------------|-------|
| 2 | sleepy | */* * * * * | False | 2 | 31s | 2m53s |

7. Clean up by deleting the CronJob.

```
student@cp:~$ kubectl delete cronjobs.batch sleepy
```

```
1 cronjob.batch "sleepy" deleted
```