Exercise 4.3: Designing Applications With Duration: 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.

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

cronjob.yaml

```
1  apiVersion: batch/v1
2  kind: CronJob   #<-- Change this line
3  metadata:
4    name: sleepy
5  spec:                        #<-- Remove completions:, parallelism:, and activeDeadlineSeconds:
6    schedule: "*/2 * * * *"    #<-- Add Linux style cronjob syntax
7    jobTemplate:               #<-- New jobTemplate and spec
8      spec:
9        template:   #<-- This and following lines space four to right
10         spec:
11           containers:
12           - name: resting
13             image: busybox
14             command: ["/bin/sleep"]
15             args: ["3"]
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
```

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

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

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

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

```
No resources found in default namespace.
```

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

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

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

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

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

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

```
NAME                COMPLETIONS   DURATION   AGE
sleepy-1539722040   1/1           5s         5m17s
sleepy-1539722160   1/1           6s         3m17s
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
```

cronjob.yaml

```
....
  jobTemplate:
    spec:
      template:
        spec:
          activeDeadlineSeconds: 10  #<-- Add this line
          containers:
          - name: resting
....
            command: ["/bin/sleep"]
            args: ["30"]             #<-- Edit this line
          restartPolicy: Never
```

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
```

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

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

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

```
student@cp:~$ sleep 120 ; kubectl get jobs
```

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

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

```
NAME     SCHEDULE      SUSPEND   ACTIVE   LAST SCHEDULE   AGE
sleepy   */2 * * * *   False     1        72s             94s
```

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

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

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

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

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

```
NAME     SCHEDULE      SUSPEND   ACTIVE   LAST SCHEDULE   AGE
sleepy   */2 * * * *   False     2        31s             2m53s
```

7. Clean up by deleting the `CronJob`.

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

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