Kubernetes provides the ability to easily run container workloads in a distributed cluster, but not all workloads need to run constantly. With jobs, we can run container workloads until they complete, then shut down the container. CronJobs allow us to do the same, but re-run the workload regularly according to a schedule. In this lesson, we will discuss Jobs and CronJobs and explore how to create and manage them.

Relevant Documentation

- https://kubernetes.io/docs/concepts/workloads/controllers/jobs-run-to-completion/
- https://kubernetes.io/docs/concepts/workloads/controllers/cron-jobs/
- https://kubernetes.io/docs/tasks/job/automated-tasks-with-cron-jobs/

Lesson Reference

This Job calculates the first 2000 digits of pi.

```
apiVersion: batch/v1
kind: Job
metadata:
  name: pi
spec:
  template:
    spec:
      containers:
      - name: pi
        image: perl
        command: ["perl",  "-Mbignum=bpi", "-wle", "print bpi(2000)"]
      restartPolicy: Never
  backoffLimit: 4
```

You can use `kubectl get` to list and check the status of Jobs.

```
kubectl get jobs
```

Here is a CronJob that prints some text to the console every minute.

```
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: hello
spec:
  schedule: "*/1 * * * *"
  jobTemplate:
    spec:
      template:
        spec:
          containers:
          - name: hello
            image: busybox
            args:
            - /bin/sh
            - -c
            - date; echo Hello from the Kubernetes cluster
          restartPolicy: OnFailure
```

You can use `kubectl get` to list and check the status of CronJobs.

```
kubectl get cronjobs
```