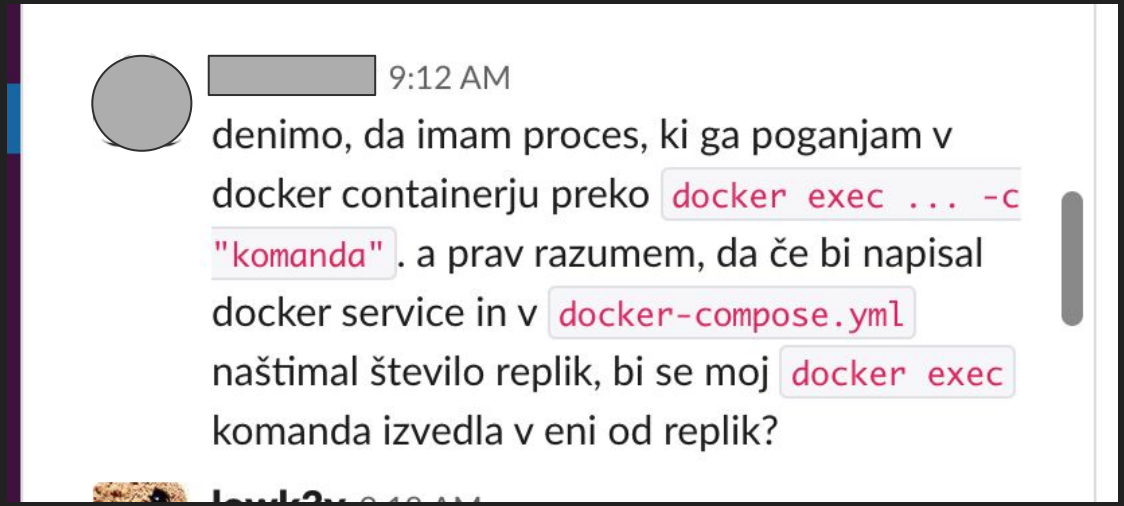
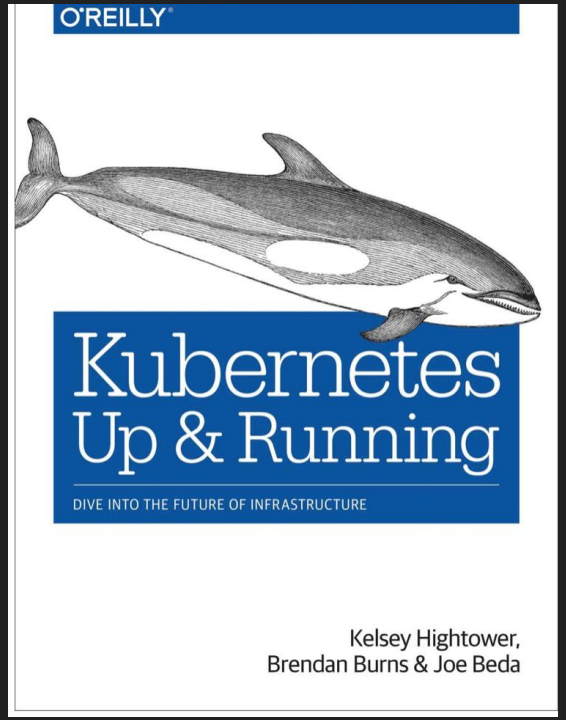
Parallel Python jobs with Kubernetes

Work queues with kubernetes

Martin Cimerman
github.com/CimeM/parallelPythonK8s

When somebody thinks docker is a great idea





Chap

So far v
such as data
workloads r
service is no
processes m
run on a Ku
short-lived,
handling th
A Job c
termination
will continu
useful for th
database m
your databa
continually

fingerprints of the keys they generated. Clean up by deleting the finished `Job` object with `kubectl delete job parallel`.

Work Queues

A common use case for Jobs is to process work from a work queue. In this scenario, some task creates a number of work items and publishes them to a work queue. A worker Job can be run to process each work item until the work queue is empty (Figure 10-1).

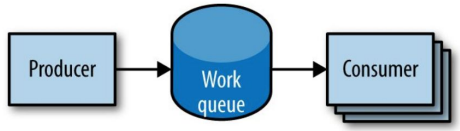


Figure 10-1. Parallel jobs

Job patterns

Type	Use case	behaviour	completions	parallelism
One Shot	Database migrations	A single pod running once until successful termination	1	1
Parallel	multiple pods processing a set of work in parallel	One or more pods running once or more times until reaching a fixed completion count	1+	1+
Work queue	Multiple pods processing from a centralized work queue	One or more pods running once or more times untill successful termination	1	2+


worker
database

worker.py

```
#!/usr/bin/env python

import time
import rediswq

host="redis"

q = rediswq.RedisWQ(name="job2", host="redis")
print("Worker with sessionID: " +  q.sessionID())
print("Initial queue state: empty=" + str(q.empty()))
while not q.empty():
  item = q.lease(lease_secs=10, block=True, timeout=2)
  if item is not None:
    itemstr = item.decode("utf=8")
    print("Working on " + itemstr)
    time.sleep(10) # Put your actual work here instead of sleep.
    q.complete(item)
  else:
    print("Waiting for work")
 print("Queue empty, exiting")
```

Dockerfile

```
FROM python
RUN pip install redis
COPY ./worker.py /worker.py
COPY ./rediswq.py /rediswq.py


CMD  python worker.py
```

```
docker build -t job-wq-2 .

docker tag job-wq-2 <username>/job-wq-2

docker push <username>/job-wq-2
```

```
kubectl run -i -t temp --image redis --labels="job-wq-2" --command "/bin/sh"


#redis-cli -h redis
redis:6379> rpush job2 "apple" "banana" "cherry" "lemon"


redis:6379> lrange job2 0 -1
1) "apple"
2) "banana"
3) "cherry"
4) "lemon"
```


worker
database

job.yaml

```
apiVersion: batch/v1
kind: Job
metadata:
  name: job-wq-2
spec:
  parallelism: 2
  template:
    metadata:
      name: job-wq-2
    spec:
      containers:
      - name: c
        image:<yourusername>/job-wq-2
       restartPolicy: OnFailure
```

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

```
kubectl describe jobs/job-wq-2

Name:             job-wq-2
Namespace:        default
Selector:         controller-uid=b1c7e4e3-92e1-11e7-b85e-fa163ee3c11f
Labels:           controller-uid=b1c7e4e3-92e1-11e7-b85e-fa163ee3c11f
                  job-name=job-wq-2
Annotations:      <none>
Parallelism:      2
Completions:      <unset>
Start Time:       Fri, 19 Apr 2019 16:07:59 -0800
Pods Statuses:    1 Running / 0 Succeeded / 0 Failed
Pod Template:
  Labels:       controller-uid=b1c7e4e3-92e1-11e7-b85e-fa163ee3c11f
                job-name=job-wq-2
  Containers:
   c:
    Image:              <yourusername>/job-wq-2
    Port:
    Environment:        <none>
    Mounts:             <none>
  Volumes:              <none>
Events:
```

```
    Image:              <yourusername>/job-wq-2
    Port:
    Environment:        <none>
    Mounts:             <none>
  Volumes:              <none>
Events:
  FirstSeen    LastSeen    Count    From              SubobjectPath    Type
Reason            Message
  ---------    --------    -----    ----              -------------    --------
------            -------
  33s          33s         1        {job-controller }                  Normal
SuccessfulCreate  Created pod: job-wq-2-lglf8
```

```
kubectl logs pods/job-wq-2-lglf8
Worker with sessionID: bbd72d0a-9e5c-4dd6-abf6-416cc267991f
Initial queue state: empty=False
Working on banana
Working on cherry
Working on lemon
```

Cleanup

```
kubectl delete rs,svc,job -l chapter=job-wk-2
```

github.com/CimeM/parallelPythonK8s