**StatefulSet**

Kubernetes statefulset by example

If you have a stateless app you want to use a deployment. However, for a stateful app you might want to use a StatefulSet. Unlike a deployment, the StatefulSet provides certain guarantees about the identity of the pods it is managing (that is, predictable names) and about the startup order. Two more things that are different compared to a deployment: for network communication you need to create a headless services and for persistency the StatefulSet manages a persistent volume per pod.

In order to see how this all plays together, we will be using an educational Kubernetes-native NoSQL datastore.

Let’s start with creating the stateful app, that is, the StatefulSet along with the persistent volumes and the headless service:

$ kubectl apply -f https://raw.githubusercontent.com/mhausenblas/mehdb/master/app.yaml

After a minute or so, you can have a look at all the resources that have been created:

$ kubectl get sts,po,pvc,svc

NAME DESIRED CURRENT AGE

statefulset.apps/mehdb 2 2 1m

NAME READY STATUS RESTARTS AGE

pod/mehdb-0 1/1 Running 0 1m

pod/mehdb-1 1/1 Running 0 56s

NAME STATUS VOLUME CAPACITY ACCESS MODES STORAGECLASS AGE

persistentvolumeclaim/data-mehdb-0 Bound pvc-bc2d9b3b-310d-11e9-aeff-123713f594ec 1Gi RWO ebs 1m

persistentvolumeclaim/data-mehdb-1 Bound pvc-d4b7620f-310d-11e9-aeff-123713f594ec 1Gi RWO ebs 56s

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

service/mehdb ClusterIP None <none> 9876/TCP 1m

Now we can check if the stateful app is working properly. To do this, we use the /statusendpoint of the headless service mehdb:9876 and since we haven’t put any data yet into the datastore, we’d expect that 0 keys are reported:

$ kubectl run -it --rm jumpod --restart=Never --image=quay.io/mhausenblas/jump:0.2 -- curl mehdb:9876/status?level=full

If you don't see a command prompt, try pressing enter.

0

pod "jumpod" deleted

And indeed we see 0 keys being available, reported above.

Note that sometimes a StatefulSet is not the best fit for your stateful app. You might be better off defining a custom resource along with writing a custom controller to have finer-grained control over your workload