# ConfigMap

Try the following case:

1. Consume ConfigMap as a *folder*.
2. Run Pod that uses the ConfigMap with the following command:

while true; do cat /path/to/file; sleep 5; done

1. Check how ConfigMap is represented on a disk. It could be done by execution command in the container via kubectl exec.
2. Update ConfigMap and wait about 2 minutes.
3. Do you see new value. What has been changed?

Try the following case:

1. Consume ConfigMap as a *environment variables*.
2. Run Pod that uses the ConfigMap. Pod may run busybox image with sleep 1000 command. Thus you can attach to it while it is running.
3. Check values of environment variables related to ConfigMap.
4. Update ConfigMap and wait about 2 minutes:

while true; do echo $ENV\_VAR; sleep 5; done

1. Check values of environment variables related to ConfigMap again. What has been changed?
2. Restart Pod.
3. Check values of environment variables related to ConfigMap again. What has been changed?

## Solution

### Case A

Try the following case:

1. Consume ConfigMap as a *folder*.

$ mkdir tmp-configmap  
  
$ cat <<"EOF" > tmp-configmap/config.txt  
key1=value1  
key2=value2  
EOF  
  
$ echo debug > tmp-configmap/mode  
  
$ kubectl create configmap myconfigmap --from-file tmp-configmap  
configmap/myconfigmap created  
  
$ kubectl describe configmaps myconfigmap  
Name: myconfigmap  
Namespace: msuslov  
Labels: <none>  
Annotations: <none>  
  
Data  
====  
config.txt:  
----  
key1=value1  
key2=value2  
  
mode:  
----  
debug  
  
Events: <none>

1. Run Pod that uses the ConfigMap with the following command:

$ cat <<"EOF" | kubectl create -f -  
apiVersion: v1  
kind: Pod  
metadata:  
 name: demo-configmap  
spec:  
 containers:  
 - name: busybox  
 image: busybox  
 imagePullPolicy: IfNotPresent  
 command:  
 - /bin/sh  
 - -c  
 - |  
 while true; do  
 echo $(date +'%Y.%m.%d %H:%M:%S')  
 cat /etc/config/mode  
 sleep 5  
 done  
 volumeMounts:  
 - name: config  
 mountPath: /etc/config  
 readOnly: true  
 volumes:  
 - name: config  
 configMap:  
 name: myconfigmap  
 restartPolicy: Never  
EOF  
  
$ kubectl get pod demo-configmap  
NAME READY STATUS RESTARTS AGE  
demo-configmap 1/1 Running 0 23s

1. Check how ConfigMap is represented on a disk. It could be done by execution command in the container via kubectl exec.

$ kubectl exec demo-configmap -- ls /etc/config  
config.txt  
mode  
  
$ kubectl exec demo-configmap -- cat /etc/config/mode  
debug  
  
$ kubectl logs demo-configmap  
debug  
debug  
...

1. Update ConfigMap and wait about 2 minutes.

$ kubectl create configmap myconfigmap --from-literal=mode=release -o yaml --dry-run=client | kubectl replace -f -  
configmap/myconfigmap replaced  
  
$ kubectl describe configmaps myconfigmap  
Name: myconfigmap  
Namespace: msuslov  
Labels: <none>  
Annotations: <none>  
  
Data  
====  
mode:  
----  
release  
Events: <none>

1. Do you see new value. What has been changed?

kubectl logs demo-configmap -f  
2021.06.13 21:13:08  
debug  
2021.06.13 21:13:13  
debug  
...  
2021.06.13 21:14:23  
release

Clean up:

$ kubectl delete pod/demo-configmap  
pod "demo-configmap" deleted

### Case B

Try the following case:

1. Consume ConfigMap as a *environment variables*.

$ cat <<"EOF" > tmp-configmap/pod.yaml  
apiVersion: v1  
kind: Pod  
metadata:  
 name: demo-configmap  
spec:  
 containers:  
 - name: busybox  
 image: busybox  
 imagePullPolicy: IfNotPresent  
 command: ["/bin/sh", "-c", "sleep 1000"]  
 envFrom:  
 - configMapRef:  
 name: myconfigmap  
 restartPolicy: Never  
EOF

1. Run Pod that uses the ConfigMap. Pod may run busybox image with sleep 1000 command. Thus you can attach to it while it is running.

$ kubectl create -f tmp-configmap/pod.yaml  
pod/demo-configmap created

1. Check values of environment variables related to ConfigMap.

$ kubectl exec demo-configmap -- printenv | grep mode  
mode=debug

1. Update ConfigMap and wait about 2 minutes:

$ kubectl create configmap myconfigmap --from-literal=mode=release -o yaml --dry-run=client | kubectl replace -f -  
configmap/myconfigmap replaced

1. Check values of environment variables related to ConfigMap again. What has been changed?

$ sleep 120  
  
$ kubectl exec demo-configmap -- printenv | grep mode  
mode=debug

1. Restart Pod.

$ kubectl delete -f tmp-configmap/pod.yaml  
pod "demo-configmap" deleted  
  
$ kubectl create -f tmp-configmap/pod.yaml  
pod/demo-configmap created

1. Check values of environment variables related to ConfigMap again. What has been changed?

$ kubectl exec demo-configmap -- printenv | grep mode  
mode=release  
  
$ kubectl delete pod/demo-configmap configmap/myconfigmap  
pod "demo-configmap" deleted  
configmap "myconfigmap" deleted  
  
$ rm -rf tmp-configmap