# Mise a jour d'une ConfigMap

Dans cet exercice nous allons créer une ConfigMap et l'utiliser dans un Deployment, nous verrons alors comment faire pour déclencher une mise à jour du Deployment lorsque l'on modifie cette ConfigMap.

## Création d'un configMap

Créez le fichier *nginx.conf* avec le contenu suivant:

```
user nginx;
worker_processes 4;
pid /run/nginx.pid;
events {
   worker_connections 768;
}
http {
  server {
    listen 8080;
    root /usr/share/nginx/html;
  }
}
```

Cette configuration est très simple, elle spécifie simplement que le serveur nginx que nous allons utiliser écoutera sur le port *8080*

A partir de ce fichier, créez une ConfigMap à l'aide de la commande suivante:

```
$ kubectl create configmap www-config --from-file=./nginx.conf
```

## Création d'un Deployment

Créez le fichier *deploy.yaml* avec le contenu suivant:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: www
spec:
```

```
  selector:
    matchLabels:
      app: www
  template:
    metadata:
      labels:
        app: www
    spec:
      containers:
      - name: nginx
        image: nginx:1.14-alpine
        volumeMounts:
        - name: config
          mountPath: "/etc/nginx/"
      volumes:
      - name: config
        configMap:
          name: www-config
```

Ce fichier définit un Deployment qui gère un Pod ayant un unique container basé sur *nginx*. La configuration créée précédemment est montée dans ce container, elle y sera disponible dans */etc/nginx/nginx.conf*.

Créez le Deployment avec la commande suivante:

```
$ kubectl apply -f deploy.yaml
```

```
```

En utilisant la commande ci-dessous, en remplaçant POD_NAME par le nom du Pod qui a été créé, vous allez vous assurez que le serveur web est bien opérationnel sur le port 8080:

```
$ kubectl port-forward POD_NAME 8080:8080
```

Vérifiez cela depuis un autre terminal en lançant la commande suivante, celle-ci devrait vous renvoyer le contenu de la page *index.html* servie par défaut par nginx.

```
$ curl localhost:8080
<!DOCTYPE html>
<html>
<head>
```

```
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

## Mise a jour de la configmap

Vous allez à présent mettre la ConfigMap à jour et changer le port d'écoute de *8080* à *9090*. Pour cela, utilisez l'une des 2 options suivantes:

### Option 1

Modifiez le fichier *nginx.conf* en remplaçant le port d'écoute par 9090 puis mettez la ConfigMap à jour avec la commande suivante:

```
$ kubectl create configmap www-config --from-file=./nginx.conf --dry-run -o yaml
| kubectl apply -f -
```

### Option 2

Utilisez la commande suivante afin de modifier la ConfigMap "on the fly" depuis un éditeur de texte:

```
$ kubectl edit cm/www-config
```

## Impact sur le Deployment

Comme précédemment, utilisez la commande `port-forward` afin de vérifier que le Pod a été redéployé et qu'il écoute maintenant sur le port 9090:

```
$ kubectl port-forward POD_NAME 9090:9090
```

Lancez un curl sur *localhost:9090* depuis un autre terminal, vous devriez noter que cela ne fonctionne pas, la mise à jour de la ConfigMap n'ayant pas déclenché la mise à jour du Deployment.

Cependant, si l'on regarde la configuration qui est montée dans le container nginx, nous pouvons voir que celle-ci correspond bien à la nouvelle version, elle n'a simplement pas été prise en compte par le process nginx.

```
$ kubectl exec -ti POD_NAME -- sh
/ # cat /etc/nginx/nginx.conf
user nginx;
worker_processes 4;
pid /run/nginx.pid;
events {
   worker_connections 768;
}
http {
  server {
    listen 9090;
    root /usr/share/nginx/html;
  }
}
```

Note: il n'y a pas eu de redéploiement car la spécification du Pod géré par le Deployment (clés se trouvant dans .spec.template) n'a pas été mise à jour.

## Modification du Deployment

Nous allons modifiez le Deployment de façon à pouvoir facilement prendre en compte la mise à jour de la ConfigMap utilisée. On modifie pour cela la spécification du container *nginx* en lui ajoutant une variable d'environnement, celle-ci contiendra un hash de la ConfigMap que nous la mettrons à jour à chaque fois que la configuration sera modifiée.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: www
spec:
  selector:
    matchLabels:
      app: www
  template:
    metadata:
      labels:
        app: www
    spec:
      containers:
      - name: nginx
        image: nginx:1.14-alpine
        volumeMounts:
        - name: config
          mountPath: "/etc/nginx/"
        env:
        - name: CONFIG_HASH
          value: ${CONFIG_HASH}
      volumes:
      - name: config
        configMap:
          name: www-config
```

Mettez le fichier *deploy.yaml* à jour avec le contenu ci-dessus et appliquer ces changements au Deployment:

```
$ kubectl apply -f deploy.yaml
```

Etant donné qu'une variable d'environnement a été ajouté au container, un nouveau Pod a été créé, celui-ci utilisant la nouvelle version de la ConfigMap.

Note: nous n'avons pas utilisé la hash de la ConfigMap pour le moment.

## Mise à jour de la ConfigMap

Une nouvelle fois, mettez la ConfigMap à jour en utilisant l'une des 2 options précédentes, remettez par exemple le port d'écoute à la valeur *8080*. Récupérer ensuite le sha256 de celle-ci.

```
export CONFIG_HASH=$(kubectl get cm -oyaml | sha256sum | cut -d' ' -f1)
```

Vous pouvez alors mettre le Deployment à jour en modifiant la variable d'environnement *CONFIG_HASH* du container *nginx*:

```
$ envsubst '${CONFIG_HASH}' < deploy.yaml | kubectl apply -f -
deployment.apps/www configured
```

Note: *envsubst* est un utilitaire très pratique pour effectuer des substitution de variable d'environnement dans des fichiers

Vérifiez qu'un nouveau Pod a été déployé et que celui-ci écoute bien sur le port 8080.