ConfigMap

Utilisation

- Découplage d’une application et de sa configuration
  - pas de configuration dans le code applicatif !
- Assure la portabilité
- Simplifie la gestion par rapport à l’utilisation de variables d’environnement
- Créée à partir d’un fichier, d’un répertoire, ou de valeurs littérales
- Contient une ou plusieurs paires de clé / valeur

Création à partir d'un fichier

```
$ cat nginx.conf
user www-data;
worker_processes 4;
pid /run/nginx.pid;
events {
  worker_connections 768;
}
http {
 server {
   listen *:8000;
   location / {
     proxy_pass http://localhost;
   }
 }
}
```

Les requêtes arrivant sur le port 8000 sont envoyées au container tournant sur le port 80 dans le même Pod

Rappel : la communication entre les différents containers d'un même Pod se fait via l'interface localhost

Création à partir d'un fichier

```
# Création d'une ConfigMap basé sur un fichier
$ kubectl create configmap nginx-config --from-file=./nginx.conf
configmap "nginx-config" created

$ kubectl get cm nginx-config -o yaml
apiVersion: v1
data:
  nginx.conf: |
    user www-data;
    worker_processes 4;
    ...
kind: ConfigMap
metadata:
  creationTimestamp: 2018-03-03T15:35:46Z
  name: nginx-config
  namespace: default
  resourceVersion: "635910"
  selfLink: /api/v1/namespaces/default/configmaps/nginx-config
  uid: 8ac176fc-1ef8-11e8-ba7f-080027f0e385
```

Le contenu du fichier de configuration nginx.conf est disponible sous la clé data de la ConfigMap

Création à partir d'un fichier d'environnement

```
# Fichier d'environnement constitué de couples key=value
$ cat app.env
log_level=WARN
env=production
cache=redis

# Création d'une ConfigMap à partir du fichier .env
$ kubectl create configmap app-config-env --from-env-file=./app.env
configmap "app-config-env" created

$ kubectl get cm app-config-env -o yaml
apiVersion: v1
data:
  cache: redis
  env: production
  log_level: WARN
kind: ConfigMap
metadata:
  creationTimestamp: 2018-03-04T14:45:09Z
  name: app-config-env
  namespace: default
  ...
```

Chaque couple clé=valeur du fichier d'environnement est défini sous le forme clé:valeur dans l'objet ConfigMap

Création à partir de valeurs littérales

```
# Utilisation de l'option --from-literal pour chaque couple clé=valeur
$ kubectl create configmap app-config-lit \
  --from-literal=log_level=WARM \
  --from-literal=env=production \
  --from-literal=cache=redis
configmap "app-config-lit" created

$ kubectl get cm app-config-lit -o yaml
apiVersion: v1
data:
  cache: redis
  env: production
  log_level: WARM
kind: ConfigMap
metadata:
  creationTimestamp: 2018-03-04T14:49:51Z
  name: app-config-lit
  namespace: default
  ...
```

Chaque couple clé=valeur fourni en ligne de commande est défini sous le forme clé:valeur dans l'objet ConfigMap

Utilisation dans un Pod : volume

```
$ cat pod-config-volume.yaml
apiVersion: v1
kind: Pod
metadata:
  name: www
spec:
  containers:
  - name: proxy
    image: nginx:1.12.2
    ports:
    - containerPort: 8000
    volumeMounts:
    - name: config
      mountPath: "/etc/nginx/"
  - name: api
    image: lucj/city:1.0
    ports:
    - containerPort: 80
  volumes:
  - name: config
    configMap:
      name: nginx-config
```

Pod contenant 2 containers
- proxy écoute sur le port 8000
- api écoute sur le port 80

Utilisation dans un Pod : volume

```
$ cat pod-config-volume.yaml
apiVersion: v1
kind: Pod
metadata:
  name: www
spec:
  containers:
  - name: proxy
    image: nginx:1.12.2
    ports:
    - containerPort: 8000
    volumeMounts:
    - name: config
      mountPath: "/etc/nginx/"
  - name: api
    image: lucj/city:1.0
    ports:
    - containerPort: 80
  volumes:
  - name: config
    configMap:
      name: nginx-config
```

Définition d'un volume basé sur la ConfigMap nginx-config

Utilisation dans un Pod : volume

```
$ cat pod-config-volume.yaml
apiVersion: v1
kind: Pod
metadata:
  name: www
spec:
  containers:
  - name: proxy
    image: nginx:1.12.2
    ports:
    - containerPort: 8000
    volumeMounts:
    - name: config
      mountPath: "/etc/nginx/"
  - name: api
    image: lucj/city:1.0
    ports:
    - containerPort: 80
  volumes:
  - name: config
    configMap:
      name: nginx-config
```

Montage du volume dans le container sur le point de montage spécifié

Définition d'un volume basé sur la ConfigMap nginx-config

Utilisation dans un Pod : volume

```
# Création du Pod www
$ kubectl create -f pod-config-volume.yaml
pod "www" created

# Lancement d'un shell interactif dans le container proxy du Pod www
$ kubectl exec -ti www --container proxy -- bash
root@www:/# cat /etc/nginx/nginx.conf
user www-data;
worker_processes 4;
pid /run/nginx.pid;
events {
  worker_connections 768;
}
http {
 server {
   listen *:8000;
   location / {
     proxy_pass http://localhost;
   }
 }
}
```

Utilisation dans un Pod : volume

```
# Lancement d'un shell interactif dans le container api du Pod www
$ kubectl exec -ti www --container api -- sh
/app # apk update && apk add curl
/app # curl localhost
{"message":"www suggests to visit Robjazpaw"}

# Lancement d'un shell interactif dans le container proxy du Pod www
$ kubectl exec -ti www --container proxy -- bash
root@www:/# apt-get update && apt-get install -y curl
root@www:/# curl localhost:8000
{"message":"www suggests to visit Tubogbaj"}
```

Le container *api* écoute sur le port 80 à l'intérieur du Pod

Le container *www* écoute sur le port 8000 à l'intérieur du Pod et formarde les requêtes au container *api*

Utilisation dans un Pod : variable d'environnement

```
$ cat pod-config-env.yaml
apiVersion: v1
kind: Pod
metadata:
  name: w3
spec:
  containers:
  - name: www
    image: nginx:1.12.2
    env:
    - name: LOG_LEVEL
      valueFrom:
        configMapKeyRef:
          name: app_config-lit
          key: log_level
    - name: CACHE
      valueFrom:
        configMapKeyRef:
          name: app-config-env
          key: cache
```

La valeur d'une variable d'environnement est lue depuis la clé correspondante dans la ConfigMap

Utilisation dans un Pod : variable d'environnement

```
# Création du Pod www
$ kubectl create -f pod-config-env.yaml
pod "www" created

# Lancement d'un shell interactif dans le container proxy du Pod www
$ kubectl exec -ti w3 --container www -- bash
# / env
HOSTNAME=w3
NJS_VERSION=1.12.2.0.1.14-1~stretch
NGINX_VERSION=1.12.2-1~stretch
CACHE=production
LOG_LEVEL=WARM
...
```

Les variables d'environnement ont les valeurs spécifiées dans la ConfigMap