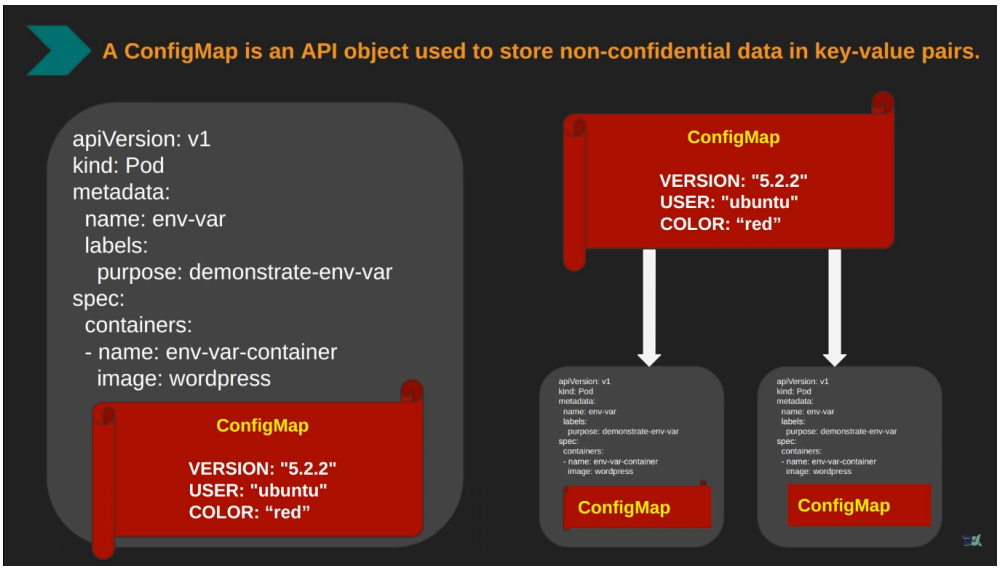
Problems with environmental variables :

- Environment variables are specified in `yaml files of pods`
- If we have multiple pods with the same environment variables, we need to mention them in each and every YAML file. Right >?

This problem is solved by ConfigMaps.

- ConfigMaps are a way to store environment variables in a central location.
- ConfigMaps can be shared by multiple pods.
- This makes it easier to manage environment variables and reduces the risk of errors.

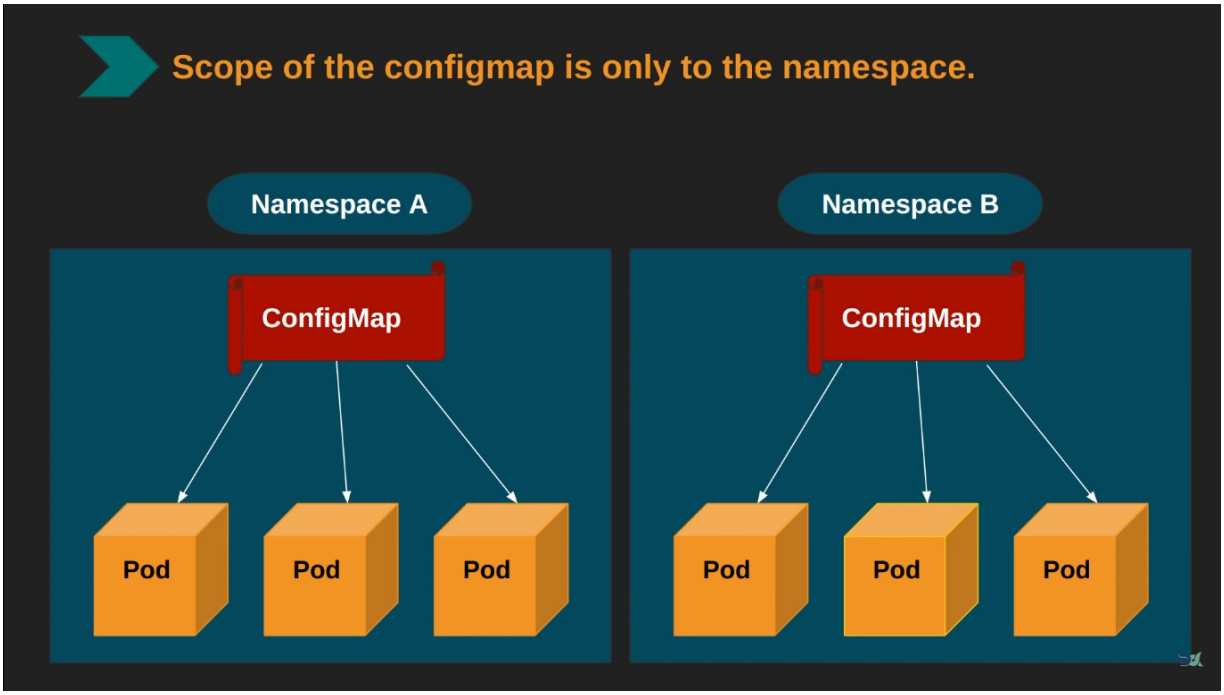


A ConfigMap is an API object used to store non-confidential data in key-value pairs. We dont store any sensitive information in it.

- Configmap is separate entity, it will not delete even if pod is deleted
- Configmap scope is limited to that particular namespace only. We cant use configmap of one name space into another namespace.

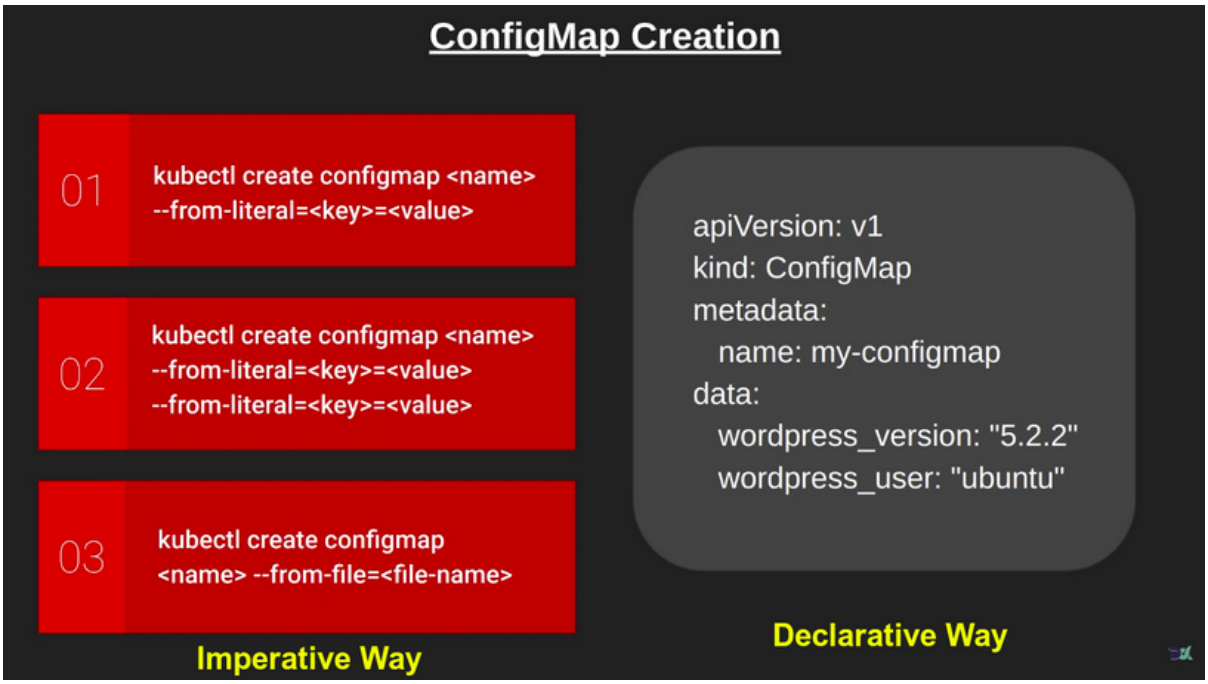
Now what is namespace.?

In Kubernetes, namespaces provides a mechanism for isolating groups of resources within a single cluster.

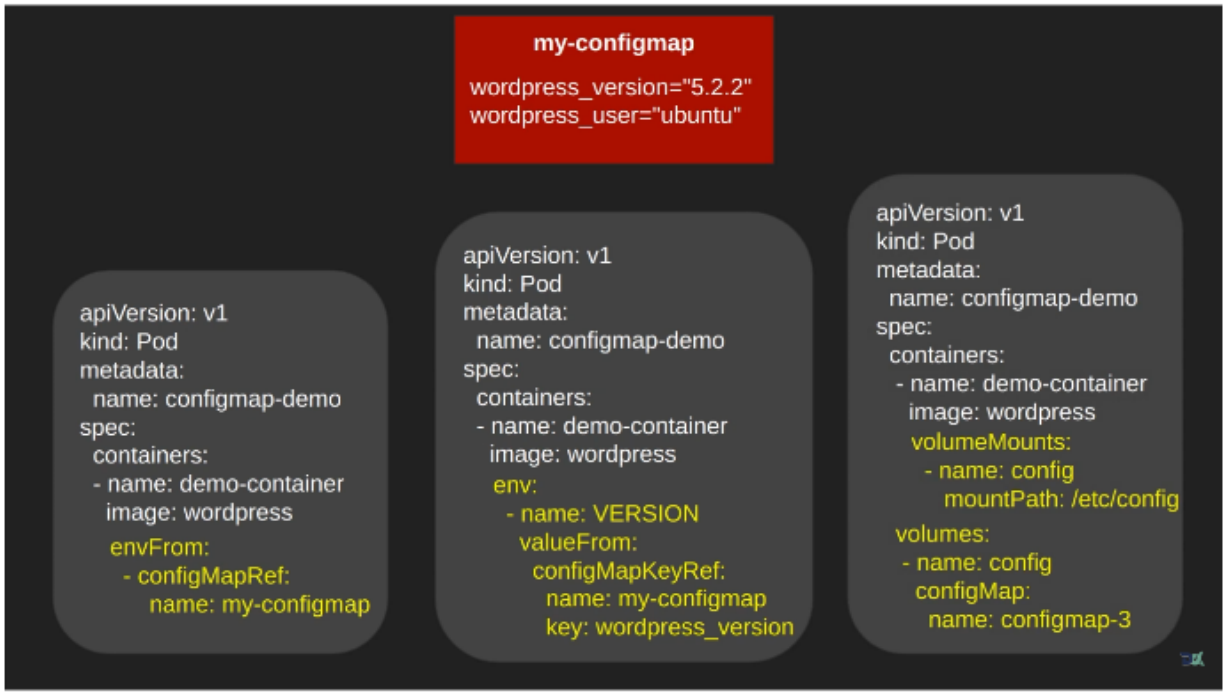


Two ways of specifying configmap :

- **Imperative way (using commands)**
- **Declarative way (using manifest yml files)**



How to attached configmap:



By passing env variable As a configmap | **Only passing specific values From configmap** | **Mounting configmap Using volume**

Imperative Way :

- kubectl create configmap name-of-cm --from-literal=key=value .. **General Command**

I m going to create one like this

- kubectl create configmap vishal-cm --from-literal=name=vishal --from-literal=engg=devops

```
5. vagrant master
vagrant@k8s-master:~$ kubectl get configmap
NAME               DATA   AGE
kube-root-ca.crt   1      18s
vagrant@k8s-master:~$ kubectl create configmap vishal-cm --from-literal=name=vishal --from-literal=engg=devops
configmap/vishal-cm created
vagrant@k8s-master:~$
vagrant@k8s-master:~$ kubectl get configmap
NAME               DATA   AGE
kube-root-ca.crt   1      37s
vishal-cm          2      3s
vagrant@k8s-master:~$
```

Pass this as a env var:

- vi env.yml

```
apiVersion: v1
kind: Pod
metadata:
  name: env-demo
spec:
  containers:
  - name: demo-container
    image: nginx
    envFrom:
    - configMapRef:
       name: vishal-cm
```

- Save , exit and apply

```
vagrant@k8s-master:~$ kubectl create configmap vishal-cm --from-literal=name=vishal --from-literal=engg=devops
configmap/vishal-cm created
vagrant@k8s-master:~$
vagrant@k8s-master:~$ kubectl get configmap
NAME               DATA   AGE
kube-root-ca.crt   1      37s
vishal-cm          2      3s
vagrant@k8s-master:~$ vi env.yml
vagrant@k8s-master:~$
vagrant@k8s-master:~$ kubectl apply -f env.yml
pod/env-demo created
```

```
vagrant@k8s-master:~$ kubectl get pods
NAME       READY   STATUS    RESTARTS   AGE
env-demo   1/1     Running   0          13m
vagrant@k8s-master:~$
```

Check:

- **Print env variables form the container**

```
vagrant@k8s-master:~$ kubectl get pods
NAME       READY   STATUS    RESTARTS   AGE
env-demo   1/1     Running   0          15m
vagrant@k8s-master:~$
vagrant@k8s-master:~$ kubectl exec -it env-demo -- printenv
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
HOSTNAME=env-demo
NGINX_VERSION=1.25.1
NJS_VERSION=0.7.12
PKG_RELEASE=1~bookworm
engg=devops
name=vishal
KUBERNETES_PORT_443_TCP_PORT=443
KUBERNETES_PORT_443_TCP_ADDR=10.96.0.1
KUBERNETES_SERVICE_HOST=10.96.0.1
KUBERNETES_SERVICE_PORT=443
KUBERNETES_SERVICE_PORT_HTTPS=443
KUBERNETES_PORT=tcp://10.96.0.1:443
KUBERNETES_PORT_443_TCP=tcp://10.96.0.1:443
KUBERNETES_PORT_443_TCP_PROTO=tcp
TERM=xterm
HOME=/root
vagrant@k8s-master:~$
```

Get only required values from configmap instead of using whole configmap:

To create a ConfigMap, use the following command:

- kubectl create configmap configmap-2 --from-literal=name=second-configmap --from-literal=color=blue

- vi specific-env.yml

```
apiVersion: v1
kind: Pod
metadata:
  name: configmap-demo-2
spec:
  containers:
  - name: demo-container
    image: nginx
    env:
    - name: COLOR
      valueFrom:
        configMapKeyRef:
          name: configmap-2
          key: color
```

- Save. apply , exit

```
vagrant@k8s-master:~$ vi specific-env.yml
vagrant@k8s-master:~$ kubectl apply -f specific-env.yml
pod/configmap-demo-2 created
vagrant@k8s-master:~$ kubectl get pods
NAME               READY   STATUS    RESTARTS   AGE
configmap-demo-2   1/1     Running   0          4s
env-demo           1/1     Running   0          26m
vagrant@k8s-master:~$ kubectl exec -it configmap-demo-2 -- printenv
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
HOSTNAME=configmap-demo-2
NGINX_VERSION=1.25.1
NJS_VERSION=0.7.12
PKG_RELEASE=1~bookworm
COLOR=blue
KUBERNETES_PORT_443_TCP_ADDR=10.96.0.1
KUBERNETES_SERVICE_HOST=10.96.0.1
KUBERNETES_SERVICE_PORT=443
KUBERNETES_SERVICE_PORT_HTTPS=443
KUBERNETES_PORT=tcp://10.96.0.1:443
KUBERNETES_PORT_443_TCP=tcp://10.96.0.1:443
KUBERNETES_PORT_443_TCP_PROTO=tcp
KUBERNETES_PORT_443_TCP_PORT=443
TERM=xterm
HOME=/root
vagrant@k8s-master:~$
```

Mount ConfigMap as a volume:

Create file , and put values in it and create configmap from this file

- vi data-file

```
username="vishal"
password="1234"
```

kubectl create configmap configmap-3 --from-file=data-file

- vi mount-as-vol.yml

```
apiVersion: v1
kind: Pod
metadata:
  name: configmap-demo-3
spec:
  containers:
  - name: demo-container
    image: nginx
    volumeMounts:
      - name: config
        mountPath: /etc/config
  volumes:
  - name: config
    configMap:
      name: configmap-3
```

- Save , apply , exit

Check :

- To check we need to go that specific location and require to print the contents,
- In above case it has to be in /etc/config dir

Enter in the container

kubectl exec -it configmap-demo-3 -- bash

```
vagrant@k8s-master:~$ kubectl get pods
NAME              READY   STATUS    RESTARTS   AGE
configmap-demo-2  1/1     Running   0          15m
configmap-demo-3  1/1     Running   0          5m17s
env-demo          1/1     Running   0          41m
vagrant@k8s-master:~$ kubectl exec -it configmap-demo-3 -- bash
root@configmap-demo-3:/#
root@configmap-demo-3:/# ls
bin   dev           docker-entrypoint.sh  home  lib32  libx32  mnt  proc
boot  docker-entrypoint.d  etc                  lib   lib64  media   opt  root
root@configmap-demo-3:/# cd /etc/config
root@configmap-demo-3:/etc/config#
root@configmap-demo-3:/etc/config# ls
data-file
root@configmap-demo-3:/etc/config# cat *
username="vishal"
password="1234"
```

We have seen, how to create configmap using imperative way. Now how to do with declarative way:

sample.yml

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: name-of-configmap
data:
  key: value
  name: vishal
```

- vi declarative-way.yml

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-web
data:
  my_website: https://www.vishalk17.com
  color: blue
```

- Save , exit , apply

Rest of the use of configmap in manifest files same as we learned previously.

```
vagrant@k8s-master:~$ kubectl get configmap
NAME               DATA   AGE
configmap-2        2      30m
configmap-3        1      19m
kube-root-ca.crt   1      60m
vishal-cm          2      59m
vagrant@k8s-master:~$
vagrant@k8s-master:~$ ls
data-file  declarative-way.yml  env.yml  mount-as-vol.yml  specific-env.yml
vagrant@k8s-master:~$ kubectl apply -f declarative-way.yml
configmap/my-web created
vagrant@k8s-master:~$
vagrant@k8s-master:~$ kubectl get configmap
NAME               DATA   AGE
configmap-2        2      30m
configmap-3        1      19m
kube-root-ca.crt   1      60m
my-web             2      3s
vishal-cm          2      60m
vagrant@k8s-master:~$
```

My devops repo :

- https://github.com/vishalk17/devops

My telegram channel:

- https://t.me/vishalk17_devops

Contact:

Telegram : t.me/vishalk17

vishalk17 My youtube Channel :

- YouTube https://www.youtube.com/@vishalk17

Ref:

DevOps Pro

- YouTube https://youtu.be/EKDmz49BhX8
-