

# Exercice

---

Le but de cet exercice est de manipuler les namespaces et de comprendre comment les utiliser pour isoler les ressources

## 1. Création

Créez les 2 namespaces *development* et *production*

## 2. Liste des namespaces

Listez les namespaces présents

## 3. Création de Deployments

Créer le Deployment *www-0* basé sur `nginx:1.14-alpine` dans le namespace par défaut

Créer le Deployment *www-1* basé sur `nginx:1.14-alpine` dans le namespace *development*

Créer le Deployment *www-2* basé sur `nginx:1.14-alpine` dans le namespace *production*

## 4. Répartition des ressources

Listez les Deployments et Pods présents sur le système (l'option `--all-namespaces` permet de prendre en compte l'ensemble des namespaces)

## 5. Suppression

Supprimez les namespaces *development* et *production*

Listez une nouvelle fois les Deployments et Pods présents sur le système. Que constatez-vous ?

Note: comme nous le verrons par la suite, il est possible de définir des règles permettant de donner accès à des actions précises dans un namespace. Cette approche permet d'utiliser les namespaces pour isoler les ressources du cluster entre plusieurs équipes (dev, qa, prod) et/ou plusieurs clients.

---

# Correction

---

## 1. Création

Les commandes suivantes permettent de créer les namespaces *development* et *production*.

```
$ kubectl create namespace development
$ kubectl create namespace production
```

## 2. Liste des namespaces

La commande suivante permet de lister les namespaces présents sur le système.

```
$ kubectl get namespace
NAME          STATUS   AGE
default       Active   10d
development   Active   14m
kube-public    Active   10d
kube-system    Active   10d
production     Active   14m
```

## 3. Création de Deployments

La commande suivante permet de créer le Deployment *www-0* dans le namespace par défaut

```
$ kubectl run www-0 --image nginx:1.14-alpine
```

La commande suivante permet de créer le Deployment *www-1* dans le namespace *development*

```
$ kubectl run www-1 --image nginx:1.14-alpine --namespace development
```

La commande suivante permet de créer le Deployment *www-2* dans le namespace *production*

```
$ kubectl run www-2 --image nginx:1.14-alpine --namespace production
```

## 4. Répartition des ressources

La commande suivante permet de lister l'ensemble des Deployments et Pods

```
$ kubectl get deploy,po --all-namespaces
```

NAMESPACE	NAME	DESIRED	CURRENT	UP-TO-DATE
AVAILABLE	AGE			
default	deploy/www-0	1	1	1
5m				
development	deploy/www-1	1	1	1
11m				
production	deploy/www-2	1	1	1
11m				

  

NAMESPACE	NAME	READY	STATUS
RESTARTS	AGE		
default	po/www-0-7f4988bf4f-kgfm7	1/1	Running 0
5m			
development	po/www-1-857988c875-5vzfb	1/1	Running 0
11m			
production	po/www-2-57676d6dcc-vczc2	1/1	Running 0
11m			
...			

## 5. Suppression

La commande suivante permet de supprimer les namespaces

```
$ kubectl delete ns/development ns/production
namespace "development" deleted
namespace "production" deleted
```

Si nous listons les Deployments et Pods, nous pouvons voir que seuls ceux qui ont été créés dans le namespace *default* sont présents. Les ressources présentes dans les namespaces *development* et *production* ont été supprimées avec la suppression du namespace.