Prise en main de Terraform

Objectifs

À la fin de cette séance, vous serez capable de :

- Installer et vérifier Terraform
- Comprendre la structure d'un projet Terraform
- Lire et écrire une configuration simple
- Comprendre le cycle de vie Terraform
- Identifier le rôle du state

1. Mise en situation

Vous devez créer :

- un environnement reproductible
- sans passer par Docker CLI ou console

Objectif :

- décrire l'infrastructure dans un fichier

2. Installation et vérification

A partir du terminal :

tfenv

tfenv simplifie l'installation de Terraform, et permet de switcher entre versions.

```
# Installation
git clone --depth=1 https://github.com/tfutils/tfenv.git ~/.tfenv

# Ajouter au PATH
echo 'export PATH="$HOME/.tfenv/bin:$PATH"' >> ~/.bashrc
source ~/.bashrc

# Installer Terraform 1.14.0
tfenv install 1.14.0
tfenv use 1.14.0
```

```
terraform version
```

Question :

- Pourquoi vérifier la version est important en équipe ?

3. Structure d’un projet Terraform

Créer un dossier :

```
mkdir terraform-intro
cd terraform-intro
```

Créer un fichier :

Fichier : `main.tf`

```
# fichier principal Terraform
```

Question :

- Une suite de commandes permet d’obtenir exactement le même résultat. Pourquoi les entreprises utilisent quand même Terraform ?

4. Première configuration

Objectif :

- définir un provider

Fichier : `main.tf`

```
terraform {
  required_providers {
    docker = {
      source = "kreuzwerker/docker"
    }
  }
}

provider "docker" {}
```

Initialiser :

Dans le terminal

```
terraform init
```

Questions :

- Que fait réellement `terraform init` ?
- Pourquoi télécharger un provider ?

5. Déclarer une ressource

Objectif :

- créer un conteneur nginx

Fichier : `main.tf`

```
resource "docker_container" "web" {
  name  = "mon_nginx"
  image = "nginx:latest"
}
```

Planifier :

Dans le terminal

```
terraform plan
```

Questions :

- Que montre le plan ?
- Terraform exécute-t-il déjà quelque chose ?

6. Mise en pratique

Appliquer :

Dans le terminal

```
terraform apply
```

Validation :

- vérifier que le conteneur existe
- docker ps

7. Erreur volontaire

Modifier :

Fichier : main.tf

```
image = "nginx:fake"
```

Relancer :

```
terraform apply
```

Questions :

- Que se passe-t-il ?
- À quel moment l'erreur apparaît-elle ?
- Pourquoi Terraform ne bloque pas avant ?

8. Correction

Remettre :

Fichier : `main.tf`

```
image = "nginx:latest"
```

Relancer apply

9. Comprendre le cycle Terraform

Cycle :

- init → prépare

- plan → prévisualise
- apply → exécute

Question :

- Pourquoi séparer plan et apply ?

10. Introduction au state

Observer :

Dans le terminal

```
ls -l
```

Fichier attendu :

- terraform.tfstate

Questions :

- À quoi sert ce fichier ?
- Que se passe-t-il si on le supprime ?

11. Extension

Ajouter un port :

Fichier : `main.tf`

```
ports {
  internal = 80
  external = 8082
}
```

Appliquer et tester

12. Modules (introduction)

Constat :

- le code va se répéter

Solution :

- créer des modules réutilisables

Exemple (conceptuel) :

- module “nginx”
- module “database”

Question :

- Pourquoi les modules sont essentiels en entreprise ?

13. À retenir

- Terraform décrit l’infrastructure
- init / plan / apply = cycle central
- le state garde la mémoire
- tout passe par des fichiers