Programme de cours
Terraform – Infrastructure as Code

Durée :	3 jours soit 21 heures	Référence :	IF-TFRM

Public visé :

Cette formation s'adresse aux :

- Administrateurs systèmes et cloud
- Ingénieurs systèmes / DevOps
- Architectes techniques
- Développeurs amenés à gérer des infrastructures

Pré-requis :

- Connaissances de base en programmation ou en scripting
- Connaissances générales des environnements systèmes ou cloud
- Aucune connaissance préalable de Terraform n'est requise

Objectifs pédagogiques :

À l'issue de la formation, les apprenants seront en mesure de :

- Comprendre et appliquer les principes de l'Infrastructure-as-Code et de la culture DevOps
- Utiliser Terraform pour déployer et maintenir une infrastructure de manière fiable
- Gérer les states Terraform en environnement individuel et collaboratif
- Structurer un projet Terraform maintenable et réutilisable
- Créer et utiliser des modules Terraform
- Appliquer les bonnes pratiques actuelles de développement Terraform en équipe
- Automatiser les déploiements Terraform de façon sécurisée

Compétences acquises à l'issue de la formation :

- Comprendre les principes de l'Infrastructure-as-Code et leur application dans une démarche DevOps.
- Installer et utiliser Terraform pour déployer et gérer des infrastructures de manière déclarative.
- Configurer et gérer les providers, ressources, variables et outputs Terraform.
- Gérer et sécuriser les states Terraform en environnement individuel et collaboratif.
- Structurer des projets Terraform maintenables et adaptés à plusieurs environnements.
- Concevoir et utiliser des modules Terraform réutilisables et versionnés.
- Appliquer les fonctionnalités avancées du langage Terraform pour écrire un code flexible et lisible.
- Automatiser et fiabiliser les déploiements Terraform en appliquant les bonnes pratiques de validation, de tests et de collaboration.

Modalités pédagogiques :

Session dispensée en présentiel ou téléprésentiel, selon la modalité inter-entreprises ou intra-entreprises sur mesure.
La formation est animée par un(e) formateur(trice) durant toute la durée de la session et présentant une suite de modules théoriques clôturés par des ateliers pratiques validant l'acquisition des connaissances. Les ateliers peuvent être accompagnés de Quizz.
L'animateur(trice) présente la partie théorique à l'aide de support de présentation, d'animation réalisée sur un environnement de démonstration.
En présentiel comme en téléprésentiel, l'animateur(trice) accompagne les participants durant la réalisation des ateliers.

Moyens et supports pédagogiques :

Cadre présentiel
Salles de formation équipées et accessibles aux personnes à mobilité réduite.
- Un poste de travail par participant
- Un support de cours numérique ou papier (au choix)
- Un bloc-notes + stylo
- Vidéoprojection sur tableau blanc
- Connexion Internet
- Accès extranet pour partage de documents et émargement électronique

Cadre téléprésentiel
Session dispensée via notre solution iClassroom s'appuyant sur Microsoft Teams.
- Un compte Office 365 par participant
- Un poste virtuel par participant
- Un support numérique (PDF ou Web)
- Accès extranet pour partage de documents et émargement électronique

Informations sur l'accessibilité :

Description / Contenu

Jour 1 — Fondamentaux Terraform & Infrastructure as Code

Module 1 : DevOps et Infrastructure as Code

- Évolution des pratiques d'exploitation et de déploiement
- Principes de la culture DevOps
- Infrastructure-as-Code : définition, objectifs et limites
- Différences entre :
  - Gestion de configuration
  - Orchestration
  - Provisioning d'infrastructure
- Outils couramment utilisés :
  - Terraform
  - Ansible
  - Docker
  - Packer
- Approche d'infrastructure immuable
- Langage déclaratif : principes et implications

Exemples d'excercices pratiques :

- *Identifier, à partir d'un schéma fourni, ce qui relève de :* provisioning, *configuration, orchestration.*
- *Comparer un déploiement manuel (documentation écrite) vs Infrastructure-as-Code.*
- Lire un fichier Terraform simple et identifier : ressources, provider, état désiré.
- Associer des outils à leurs usages réels (Terraform, Ansible, Docker, Packer).
- Identifier les avantages et limites de l'approche infrastructure immuable dans un cas concret.

Module 2 : Présentation de Terraform

- Rôle et positionnement de Terraform
- Architecture générale de Terraform
- Fonctionnement des providers
- Cycle de vie d'une ressource Terraform
- Documentation officielle et bonnes pratiques de lecture

Exemples d'excercices pratiques :

- *Analyser l'architecture d'un projet Terraform simple*
- *Identifier le rôle exact :* du provider, des ressources, du state
- *Lire la documentation officielle d'un provider et retrouver :* une ressource, ses arguments obligatoires
- *Identifier le cycle de vie d'une ressource à partir d'un terraform plan*
- *Comprendre les dépendances implicites entre ressources*

Module 3 : Premiers pas avec Terraform

- Installation de Terraform
- Structure minimale d'un projet Terraform
- Vue d'ensemble du langage HCL
- Commandes principales de la CLI :
  - terraform init
  - terraform plan
  - terraform apply
  - terraform destroy
- Déploiement d'une infrastructure simple :
  - Ressource unique
  - Serveur applicatif
- Introduction aux variables et outputs

Exemples d'excercices pratiques :

- *Initialiser un projet Terraform (terraform init)*
- *Déployer une ressource simple via un provider cloud ou local*
- *Utiliser des variables pour paramétrer une ressource*
- *Générer et analyser un terraform plan*
- *Détruire proprement l'infrastructure (terraform destroy)*

Jour 2 — State, collaboration et structuration des projets

Module 4 : Gestion des states Terraform

- Rôle et fonctionnement du state Terraform
- Différence entre state local et state distant
- Problématiques liées au state :
  - Concurrence
  - Sécurité
  - Perte de cohérence
- Stockage distant du state :
  - S3
  - Terraform Cloud
  - Consul
- Verrouillage du state (state locking)
- Lecture seule du state (data sources et remote state)

Exemples d'excercices pratiques :

- *Observer le contenu d'un fichier terraform.tfstate*
- *Passer d'un state local à un backend distant*
- *Tester le verrouillage du state en situation de concurrence*
- *Utiliser un terraform_remote_state pour lire des outputs*
- *Identifier les risques liés à une mauvaise gestion du state*

Module 5 : Organisation des projets Terraform

- Structuration des répertoires Terraform
- Séparation des environnements (dev / test / prod)
- Bonnes pratiques pour :
  - Variables
  - Fichiers .tf
  - Outputs
- Gestion des dépendances entre composants
- Introduction aux workspaces Terraform (cas d'usage et limites)

Exemples d'excercices pratiques :

- *Réorganiser un projet Terraform mal structuré*
- *Séparer des environnements (dev / prod) par fichiers ou dossiers*
- *Utiliser des variables par environnement*
- *Tester l'impact d'un changement d'organisation sur le state*
- *Comparer workspaces vs séparation par projets*

Module 6 : Collaboration et contrôle de version

- Terraform et Git
- Bonnes pratiques de versionnement du code Terraform
- Gestion des changements d'infrastructure
- Revue de code Terraform
- Bonnes pratiques pour travailler à plusieurs sur un même projet

Exemples d'excercices pratiques :

- *Mettre un projet Terraform sous Git*
- *Simuler une modification concurrente du code*
- *Analyser un diff Terraform avant application*
- *Appliquer une modification incrémentale contrôlée*
- *Identifier les bonnes pratiques de revue de code Terraform*

Jour 3 — Modules, bonnes pratiques avancées et automatisation

Module 7 : Modules Terraform

- Principe et intérêt des modules Terraform
- Différence entre modules racine et modules enfants
- Création d'un module simple
- Variables d'entrée et de sortie
- Modules configurables
- Versionnement des modules
- Réutilisation et maintenabilité du code
- Bonnes pratiques actuelles de conception de modules

Exemples d'excercices pratiques :

- *Créer un module Terraform simple*
- *Définir des variables d'entrée et de sortie*
- *Appeler un module depuis un module racine*
- *Versionner un module*
- *Refactoriser un projet existant en modules*

Module 8 : Langage Terraform – fonctionnalités avancées

- Expressions conditionnelles
- count et for_each
- Boucles et itérations
- Fonctions intégrées couramment utilisées
- Gestion des dépendances explicites et implicites
- Bonnes pratiques de lisibilité du code

Exemples d'excercices pratiques :

- *Utiliser count pour créer plusieurs ressources*
- *Remplacer count par for_each lorsque pertinent*
- *Implémenter une condition avec une expression conditionnelle*
- *Utiliser des fonctions natives Terraform*
- *Comprendre et corriger des dépendances explicites*

Module 9 : Qualité, automatisation et bonnes pratiques

- Validation et formatage du code :
  - terraform validate
  - terraform fmt
- Tests Terraform :
  - Tests natifs Terraform (terraform test)
- Introduction aux pipelines d'automatisation Terraform
- Intégration de Terraform dans un workflow CI/CD
- Gestion sécurisée des variables sensibles
- Bonnes pratiques globales pour des projets Terraform en production

Exemples d'excercices pratiques :

- *Formatter et valider un projet Terraform*
- *Écrire un test Terraform simple avec terraform test*
- *Identifier et corriger des erreurs de configuration*
- *Simuler l'intégration de Terraform dans un pipeline CI/CD*
- *Gérer des variables sensibles de manière sécurisée*