# Training Catalog

@mjbright CONSULTING

training@mjbright.net

**Adaptation**

All trainings can be delivered in English or in French and adapted to specific needs in terms of:

- subjects covered, content
- duration, time zone (max 8 hours duration/day between 7h to 23h59 CET for virtual trainings)
- delivery as half or full days
- training duration can be reduced (& content) or extended (more time for labs)
- English or French delivery (some supporting materials may still be in English)
- Virtual training preferred, or on your premises or in a rented meeting room
- Pre-/post-training support can be added (a la French Qualiopi)

**Legend**

| |
|---|
| Available |
| On demand |
| Later |

## Trainings Index

| Category | Training |
|---|---|
| **Linux** | [1d] BELOW LIN-001-EN Linux Introduction - command line / shell skills |
| | |
| **General** | [1d] BELOW GEN-001-EN Containers for Managers |
| | |
| **Kubernetes** | [2d] BELOW K8S-001-EN Kubernetes Introduction |
| | [4d] BELOW K8S-002-EN Kubernetes Administration |
| | [3d] BELOW K8S-003-EN Kubernetes Application Development |
| | [4d] BELOW K8S-004-EN Kubernetes Security |
| | |
| | [2d] BELOW K8S-010-EN Advanced Kubernetes |
| | [1d] BELOW K8S-011-EN Introduction to Helm |
| | [1d] BELOW K8S-012-EN Kubernetes Policy Management (OPA) |
| | [3d] BELOW K8S-013-EN Kubernetes Policy Management (Kyverno) |

| | |
|---|---|
| | [2d] BELOW K8S-014-EN Deploying Kubernetes with Istio Service Mesh |
| | [2d] BELOW K8S-015-EN Deploying Kubernetes with Linkerd Service Mesh |
| | |
| | [1d] BELOW K8S-020-EN CKA exam preparation |
| | [1d] BELOW K8S-021-EN CKAD exam preparation |
| | [2d] BELOW K8S-022-EN KCNA exam preparation |
| | [4d] BELOW K8S-023-EN CKS exam preparation |
| | |
| | |
| **Docker** | [2d] BELOW DO-001-EN Introduction to Docker |
| | [2d] BELOW DO-002-EN Avanced Docker |
| | |
| **Terraform / OpenTofu** | [2d] BELOW TF-000-EN Terraform Introduction (CORE) |
| | [1d] BELOW TF-001-EN Terraform Introduction (AWS) |
| | [1d] BELOW TF-002-EN Terraform Introduction Terraform (Azure) |
| | [1d] BELOW TF-003-EN Terraform Introduction Terraform (Google Cloud) |
| | [1d] BELOW TF-004-EN Terraform Introduction Terraform (Oracle Cloud) |
| | [1d] BELOW TF-005-EN Terraform Introduction Terraform (OpenStack) |
| | [1d] BELOW TF-006-EN Terraform Introduction Terraform (Proxmox) |
| | |
| | [2d] BELOW TF-011-EN Advanced Terraform (AWS) |
| | [2d] BELOW TF-012-EN Advanced Terraform (Azure) |
| | [2d] BELOW TF-013-EN Advanced Terraform (Google Cloud) |
| | [2d] BELOW TF-014-EN Advanced Terraform (Oracle Cloud) |
| | [2d] BELOW TF-015-EN Advanced Terraform (OpenStack) |
| | [2d] BELOW TF-016-EN Advanced Terraform (Proxmox) |
| | |
| | [2d] BELOW TF-021-EN OpenTofu Introduction (AWS) |
| | |
| | |
| Configuration Management | |
| | [2d] BELOW AN-002-EN Introduction to Ansible |

| | |
|---|---|
| | [3d] BELOW AN-003-EN Introduction to Ansible |
| | |
| | |
| Policy Management | [2d] BELOW POL-001-EN Introduction to Policy Management with OPA/Rego |
| | [1d] BELOW K8S-010-EN Kubernetes Policy Management (OPA) |
| | [3d] BELOW K8S-011-EN Kubernetes Policy Management (Kyverno) |
| | |
| | |
| WebAssembly | [2d] BELOW WASM-001-FN Introduction to WebAssembly |
| | |
| | |

# Trainings

## TRAININGS: Kubernetes

### [4d] K8S-002-EN Kubernetes Administration

#### Objectives

The objective of the training is to acquire Kubernetes Administration basics:

* Learn about container orchestration
* Master the installation, configuration of a Kubernetes platform
* Understand the main types of objects managed by Kubernetes
* Know how to deployer applications on Kubernetes
* Be able to debug and observe applications and the Kubernetes cluster itself

The training consists of about 50% theory and 50% hands-on labs.

**Note:** Although the training is a good preparation for the CKA certification, it can also be combined with the *"K8S-004-EN CKA Exam Preparation"* training.
This day is dedicated to practical exercises to help pass the certification.
It is recommended to take the preparation training a week later to properly fix the learnings.

#### Training Outcomes

The training participant will acquire Kubernetes Administration basics, in particular:

* Installation, Configuration, Upgrading of a Kubernetes Cluster
* Deploying and upgrading applications on Kubernetes
* How to expose applications
* Use of external data volumes
* Observation, debugging of applications & of the cluster
* Security & Best practices

#### Target Audience

This training is ideal for engineers with little or no experience with Kubernetes administration.

Participants must provide their own PC with internet access, with the ability:

* to use a browser to connect to sites hosted on AWS
* to connect to AWS EC2 VMs using SSH: alternative connection methods can be provided

#### Pre-requisites

To take full advantage of this training, participants:

* should be at ease working at the command line
* have basic notions of Linux processes
* have basic notions of container engines such as Docker
* Be able to use an SSH client such as openssh on Linux, macOS, WSL or Putty on Windows

If lacking command-line skills it is advised to first follow the training *“LIN-001-EN - Introduction to Linux - command-line / shell”* or equivalent.

## Evaluation

At the beginning of the training we will verify the domain experience, if any, and any expectations of each participant.

An accompaniment can be proposed at extra cost after the formation.

## Programme

Proposed as a 4 day training, this may be extended to 5 days to have more time to complete the hands-on labs and thus better assimilate the concepts & practical skills.

Working from concrete examples, the following aspects will be covered

**MODULE: Kubernetes Principles & Concepts**

* Review of container principles
* Why Pods - comparison with Containers, VMs ?
* Launch and interact with a Pod (ports, exec, logs)
* “Reliability at scale” with Kubernetes
* Basic principles: loose coupling, desired state, namespaces, Pods, Controllers, Services
* Installation of a Kubernetes cluster (kubeadm) & CNI (Cilium)
* The Kubernetes network model
* Kubeconfig
* The Kubernetes bootstrap process, static Pods

**MODULE: Deploying applications on Kubernetes**

* Kubernetes architecture
* Pods - init containers, multi-container patterns
* Resource management by Container, by Namespace
* API Kubernetes - API groups, explain, API access
* Workload controllers
* Application upgrades
* Storage: basic, PV/PVC, dynamic provisioners

**MODULE: More storage, exposing applications on Kubernetes**

* ConfigMaps: creation, usage, updating
* Secrets: creation, usage, updating
* Extending Kubernetes - Helm, CRDs, Operators
* Exposing applications:
  * Services
  * Ingress controller
  * Gateway API
* ServiceMesh

**MODULE: Observability, Debugging, Sécurity, HA**

* Observability
    * MetricsServer, Dashboard
    * Logging - EFK, Loki
    * Metrics - Prometheus/grafana
* Troubleshooting
    * Application troubleshooting with challenge
* Security
    * SecurityContext & PodSecurityStandard
    * APIServer stages - Authentication, Authorisation, Admission Control
    * Policy Engines: OPA, Kyverno
    * Networkpolicies
* Cluster HA
* Advice for taking the CKA exam

# [3d] K8S-003-EN Kubernetes Application Development

## Objectives

The objective of the training is to acquire basic skills of Application Development for Kubernetes:

* Learn about container orchestration
* Master the creation and hosting of application container images
* Learn to develop applications following Cloud Native Principles
* Know the principle types of objects managed by Kubernetes
* Know how to deploy applications on Kubernetes
* Be able to debug and observe applications on Kubernetes
* How to secure applications, best practices

The training consists of about 50% theory and 50% hands-on labs.

**Note:** Although the training is a good preparation for the CKAD certification, it can also be combined with the *"K8S-005-EN CKAD Exam Preparation"* training.
This day is dedicated to practical exercises to help pass the certification.
It is recommended to take the preparation training a week later to properly fix the learnings.

## Training Outcomes

The training participant will acquire basics of Application Developement for Kubernetes, in particular:

* How to create container images for applications
* How to host container images in a registry
* Deploying and upgrading applications on Kubernetes
* How to expose applications
* Use of external data volumes
* Observation, debugging of applications
* Security & Best practices

## Target Audience

This training is ideal for engineers with little or no experience with Kubernetes application development.

Participants must provide their own PC with internet access, with the ability:

* to use a browser to connect to sites hosted on AWS
* to connect to AWS EC2 VMs using SSH: alternative connection methods can be provided

## Pre-requisites

To take full advantage of this training, participants:

* should be at ease working at the command line
* have basic notions of Linux processes
* have basic notions of container engines such as Docker
* Be able to use an SSH client such as openssh on Linux, macOS, WSL or Putty on Windows

If lacking command-line skills it is advised to first follow the training *"LIN-001-EN - Introduction to Linux - command-line / shell"* or equivalent.

## Evaluation

At the beginning of the training we will verify the domain experience, if any, and any expectations of each participant.

An accompaniment can be proposed at extra cost after the formation.

## Programme

Proposed as a 3 day training, this may be extended to 4 days to have more time to complete the hands-on labs and thus better assimilate the concepts & practical skills.

Working from concrete examples, the following aspects will be covered

**MODULE: Kubernetes Principles & Concepts:**

* Review of container principles
* Why Pods - comparison with Containers, VMs ?
* Launch and interact with a Pod (ports, exec, logs)
* “Reliability at scale” with Kubernetes
* Basic principles: loose coupling, desired state, namespaces, Pods, Controllers, Services
* Scripted installation of a Kubernetes cluster (kubeadm) & CNI (Cilium)
* The Kubernetes network model
* Kubeconfig
* The Kubernetes bootstrap process, static Pods

**MODULE: Deploying applications on Kubernetes:**

* Kubernetes architecture
* Pods - init containers, multi-container patterns
* Resource management by Container, by Namespace
* API Kubernetes - API groups, explain, API access
* Workload controllers
* Application upgrades
* Storage: basic, PV/PVC, dynamic provisioners
* ConfigMaps: creation, usage, updating
* Secrets: creation, usage, updating
* Installing applications with Helm

**MODULE: Exposing applications on Kubernetes, Observability, Debugging, Sécurity:**

* Exposing applications:
  * Services
  * Ingress controller
  * Gateway API
* Observability:
  * MetricsServer
  * Logging - EFK, Loki
  * Metrics - Prometheus/grafana
* Troubleshooting:
  * Application troubleshooting with challenge
* Security:
  * SecurityContext & PodSecurityStandard
  * APIServer stages - Authentication, Authorisation, Admission Control

    * Policy Engines: OPA, Kyverno
    * Networkpolicies
* Advice for taking the CKAD exam:

# [1d] K8S-020-EN CKA exam preparation

## Objectives

The objective of this single day training is to prepare to take the Linux Foundation *"CKA - Certified Kubernetes Administrator"* exam.

Participants will be challenged with various exam-like tasks to perform - see below for more details.

The training consists of about 90% hands-on labs.

## Training Outcomes

The training participant will be well prepared to take the certification

## Target Audience

This training is ideal for engineers who already have some experience with Kubernetes administration.

**Note:** Following K8S-002-EN & K8S-004-EN trainings alone is not sufficient to succeed in certification - much practice is necessary. This training will provide tips, practical exercises and resources to help prepare for the exam.

## Pre-requisites

To take full advantage of this training, participants:

* should be at ease working at the command line
* have basic notions of Linux processes
* have basic notions of container engines such as Docker
* Be able to use an SSH client such as openssh on Linux, macOS, WSL or Putty on Windows

If lacking command-line skills it is advised to first follow the training *"LIN-001-EN - Introduction a Linux - en ligne de commande / shell"* or equivalent.

Participants must provide their own PC with internet access, with the ability:

* to use a browser to connect to sites hosted on AWS
* to connect to AWS EC2 VMs using SSH: alternative connection methods can be provided

Participants should already have a first experience with the theory and practice of administering Kubernetes - having followed the *"K8S-002-EN Administration Kubernetes"* training for example or équivalent.

## Evaluation

At the beginning of the training we will verify the domain experience, if any, and any expectations of each participant.

## Programme

During the day, participants will be challenged with practical tasks to accomplish similar to those encountered in the exam, for example:

* Implement a scénario:
    * a Deployment where the Pods access a ConfigMap mounted as a Volume
    * a DaemonSet where the Pods access a Secret via environment variables
    * prevent Pods in the ‘frontend’ Namespace ‘frontend’ being able to communicate with Pods in the ‘backend’ Namespace
* Debug scenarios:
    * a Service which fails to respond to requests, or is intermittent
    * a Pod which fails to start, or a Container which continually restarts
    * kubectl timing out

# [1d] K8S-021-EN CKAD exam preparation

## Objectives

The objective of this single day training is to prepare to take the Linux Foundation *"CKAD - Certified Kubernetes Application Developer"* exam.

Participants should already have a first experience with the theory and practice of developing for Kubernetes - having followed the *"K8S-003-EN Development with Kubernetes"* training for example or équivalent.

Participants will be challenged with various exam-like tasks to perform - see below for more details.

The training consists of about 90% hands-on labs.

## Training Outcomes

The training participant will be well prepared to take the certification

## Target Audience

This training is ideal for engineers who already have some experience with development for Kubernetes administration.

**Note:** Following K8S-003-EN & K8S-005-EN trainings alone is not sufficient to succeed in certification - much practice is necessary. This training will provide tips, practical exercises and resources to help prepare for the exam.

Participants must provide their own PC with internet access, with the ability:

* to use a browser to connect to sites hosted on AWS
* to connect to AWS EC2 VMs using SSH: alternative connection methods can be provided

## Pre-requisites

To take full advantage of this training, participants:

* should be at ease working at the command line
* have basic notions of Linux processes
* have basic notions of container engines such as Docker
* Be able to use an SSH client such as openssh on Linux, macOS, WSL or Putty on Windows

If lacking command-line skills it is advised to first follow the training *"LIN-001-EN - Introduction a Linux - en ligne de commande / shell"* or equivalent.

## Evaluation

At the beginning of the training we will verify the domain experience, if any, and any expectations of each

participant.

## Programme

During the day, participants will be challenged with practical tasks to accomplish similar to those encountered in the exam, for example:

* Implement a scénario:
  * a Deployment where the Pods access a ConfigMap mounted as a Volume
  * a DaemonSet where the Pods access a Secret via environment variables
  * prevent Pods in the ‘frontend’ Namespace ‘frontend’ being able to communicate with Pods in the ‘backend’ Namespace

* Debug scenarios:
  * Correct a Dockerfile
  * Make sure that best practices are adhered to for a Dockerfile
  * a Service which fails to respond to requests, or is intermittent
  * a Pod which fails to start, or a Container which continually restarts

# [1d] K8S-011-EN Introduction to Helm

## Objectives

The objective of the training is to acquire basic skills of using Helm to create and install application Charts for Kubernetes.  The principal objectives are :

* Facilitate application life-cycle management on Kubernetes
* Customize applications at installation time

The training consists of about 50% theory and 50% hands-on labs.

## Training Outcomes

The training participant will acquire the basics of Application life-cycle management using Hem, in particular :

* Installation, configuration, upgrading of an application on Kubernetes
* Creation & test of a Helm chart

## Target Audience

This training is ideal for engineers with some experience with Kubernetes application development.

Participants must provide their own PC with internet access, with the ability :

* to use a browser to connect to sites hosted on AWS
* to connect to AWS EC2 VMs using SSH: alternative connection methods can be provided

## Pre-requisites

To take full advantage of this training, participants :

* should be at ease working at the command line
* have basic notions of Linux processes
* have basic notions of container engines such as Docker
* Be able to use an SSH client such as openssh on Linux, macOS, WSL or Putty on Windows

If lacking command-line skills it is advised to first follow the training *“LIN-001-EN - Introduction a Linux - en ligne de commande / shell”* or equivalent.

## Evaluation

At the beginning of the training we will verify the domain experience, if any, and any expectations of each participant.

An accompaniment can be proposed at extra cost after the formation.

## Programme

Proposed as a single day training, this may be extended to 2 days to going into more depth and with more time to complete the hands-on labs and thus better assimilate the concepts & practical skills.

Working from concrete examples, the following aspects will be covered

**MODULE: Introduction to Helm & Concepts:**

* Introduction to Helm : What is it, why do we need it ?
* Comparison of Helm Charts with alternatives (k8s yaml, kustomize)
* Charts : Structure & composition
* Helm Repositories : ArtifactHub, Chart Museum, Repositories
* How to install an application with Helm

**EXERCISE: Installation:**

* Creation of a Kubernetes cluster via a simplified script
* Helm Installation
* Searching for a repository and a chart via the ArtifactHub & the Helm CLI
* Installation of a Chart on a Kubernetes cluster

**MODULE: Helm Commands & Workflow:**

* Chart Structure
* Helm Commands : helm install, helm upgrade, helm rollback
* Customizing by overriding installation values
* Updating an application
* Template functions, variables, conditionals

**EXERCISE: Creation of a simple Helm Chart:**

* Recuperating a chart to examine it's structure
* Creation of a new Chart
* Customized Chart installation using the '--set' option
* Customized Chart installation using the '-f values.yaml' option
* Use of the 'dry-run' option

**MODULE: Advanced Helm:**

* Upgrading an application
* Some useful helm commands
* Helm Best Practices
* Use of pre-/post-operation hooks
* Testing Helm Charts
* Packaging & dependencies

**EXERCISE: Upgrades, Rollback:**

* Release management with Helm.
* Updating an application using new values
* Rolling back to aTesting Helm Charts
* Tests des Charts

**MODULE: Questions & Answers, Feedback:**

* Q & A session
* Course evaluation

# TRAININGS: Terraform

## [2d] TF-000-EN Terraform Introduction (CORE)

### Description

This training, proposed as a 2-day training, or 4 half-days, introduces students to the advantages of developing “Infrastructure as Code” with Terraform.

The intention is to follow this 2-day training with a separate 1-day training which applies the concepts learnt to the customers preferred Cloud Provider (AWS, Azure, Google Cloud, Oracle) or other virtualized environment (Proxmox, OpenStack).

Note: These trainings are proposed as Terraform, but can equally be delivered as OpenTofu trainings

This initial 2-day training is the recommended way to learn Terraform basics:
* It uses mainly Docker or local Providers for the base exercises - This allows to assimilate important Terraform concepts at an accelerated pace

Terraform allows to manage the deployment of different infrastructure types via the appropriate “Provider” plugin - this course uses the Docker provider to be able to experiment ten times faster than when using Cloud Providers or even Hypervisors.

Students will appreciate the ease with which infrastructure resources are defined in a declarative manner - using HCL v2 - “*HashiCorp Configuration Language*” - allowing resources to be created, updated or destroyed in an idempotent manner.

It is recommended to combine this 2-day training with one or more of the following 1-day Provider specific trainings:
* TF-001-EN Terraform Introduction (AWS)
* TF-002-EN Terraform Introduction Terraform (Azure)
* TF-003-EN Terraform Introduction Terraform (Google Cloud)
* TF-004-EN Terraform Introduction Terraform (Oracle Cloud)
* TF-005-EN Terraform Introduction Terraform (OpenStack)
* TF-006-EN Terraform Introduction Terraform (Proxmox)

### Pre-requisites

* Be at ease working at the command-line, editing files
* Basic notions of Cloud, Linux, Containers
* Use of an ssh Client, e.g. openssh on Linux, macOS or Windows (or Putty)

### Included

* Course materials and labs: 50% hands-on, 50% presentation & demos
* Access to a temporary lab environment
* Access to an evolutive document covering various Terraform learning resources

### Objectives

* Learn to use Terraform to stand up various resources, in a declarative manner
* Learn the many intricacies of Terraform/HCLv2 to write quality declarative configs

- Know where to find information about other Providers, Modules for Google Cloud, Azure etc …

## Programme

**MODULE: Introduction to Infrastructure as Code:**
* HashiCorp “free to use” ecosystem
* Infrastructure as Code, Config Management, Idempotence
* Terraform
* Installation

**MODULE: Terraform Workflow:**
* The plan
* Applying and re-applying plans
* Destroying resources
* Various sub-commands

**MODULE: HCL Configurations:**
* Providers
* Variables
* Resources

**MODULE: Variable types:**
* Variables,passing values to the configuration, Locals
* Basic and complex types
* Functions

**MODULE: Control Structures:**
* Count, ternary, for_in, for_each
* Dynamic Blocks
* Templates

**MODULE: Terraform Registry:**
* Provider Data Sources
* Modules
* Using Modules
* Writing your own Modules

**MODULE: In Practice:**
* Variable validation
* Debugging
* 3rd-party tools

**MODULE: State:**
* Local State
* Using “remote state” for working in teams

**MODULE: Importation of foreign ressources:**
Importation of resources created outside Terraform:
* terraform import
* Import blocks

**MODULE: In Production:**
* Provisioners (Local-exec, File, Remote-exec)
* Terraform Best practices
* Tooling: Linters, scanners, testers
* Terraform test

**MODULE: HashiCorp Terraform Eco-system:**

* HCP Terraform & Enterprise
* Terraform CDK
* Waypoint, Boundary
* Terraform Certification

# [1d] TF-001-EN Terraform Introduction (AWS)

## Description

This training proposed as a 1-day training, or 2 half-days, applies to the AWS Cloud the principles of Terraform already learnt in the “TF-000-EN Terraform Introduction (CORE)” 2-day training to resources.

This training will be oriented more toward hands-on than theory - it is not intended to teach about AWS, but to experience the application of Terraform to AWS resources

**Note:** Taking “TF-000-EN Terraform Introduction (CORE)” is a prerequisite for this cloud specific training.

Terraform allows to manage the deployment of different infrastructure types via the appropriate “Provider” plugin - this course uses the AWS provider allowing to manage many AWS cloud resources.

Students will appreciate the ease with which infrastructure resources are defined in a declarative manner allowing resources to be created, updated or destroyed in an idempotent manner.

Terraform uses HCL v2 - “*HashiCorp Configuration Language*” - to define resources to be created for 1 or more providers.

Note: the same course is available for other Providers

## Pre-requisites

- Be at ease working at the command-line, editing files
- Basic notions of Cloud, Linux, Containers
- Use of an ssh Client, e.g. openssh on Linux, macOS or Windows (or Putty)

## Included

- Course materials and labs: 50% hands-on, 50% presentation & demos
- Access to a temporary lab environment
- Access to an evolutive document covering various Terraform learning resources

## Objectives

- Learn to use Terraform for standing up various AWS resources, in a declarative manner
- Learn to use AWS specific modules, data sources & tooling

**Note:** the following AWS specific resources are proposed, but can be adapted to customer needs

## Programme

**MODULE: Review**
* Infrastructure as Code principles
* Terraform & OpenTofu workflows

**MODULE: Working with Containers**
* Managing AWS ECS containers with Terraform
* Using Data Sources with AWS ECS

**MODULE: Working with VMs**
* Managing AWS EC2 virtual machines with Terraform
* Using Data Sources with AWS EC2

**MODULE: Working with modules**
* Terraform registry: Working with existing terraform modules for AWS
* Writing modules for AWS: Creating clusters of VMs

**MODULE: Templates**
* Creating useful templates (ssh_config, ansible inventory, reports)

**MODULE: In Practice**
* Variable validation
* Debugging
* 3rd-party tools

**MODULE: State**
* Local State
* Using AWS/S3+DynamoDB for “remote state”

**MODULE: Importation of foreign ressources**
* Importation of AWS resources
* Move of AWS resources

**MODULE: Auto-scaling & Load-Balancing**
* AWS EC2 ASG - Autoscaling Groups
* AWS EC2 ALB - Application Load Balancer

**MODULE: Other AWS resources**
* Lambda, VPC, EIP, S3, EBS, IAM, RDS

**MODULE: In Production**
* Provisioners (Local-exec, File, Remote-exec)
* Provider Aliases

# [1d] TF-002-EN Terraform Introduction (Azure)

## Description

This training proposed as a 1-day training, or 2 half-days, applies to the Azure Cloud the principles of Terraform already learnt in the “TF-000-EN Terraform Introduction (CORE)” 2-day training to resources.

This training will be oriented more toward hands-on than theory - it is not intended to teach about Azure, but to experience the application of Terraform to Azure resources

**Note:** Taking “TF-000-EN Terraform Introduction (CORE)” is a prerequisite for this cloud specific training.

Terraform allows to manage the deployment of different infrastructure types via the appropriate “Provider” plugin - this course uses the Azure provider allowing to manage many Azure cloud resources.

Students will appreciate the ease with which infrastructure resources are defined in a declarative manner allowing resources to be created, updated or destroyed in an idempotent manner.

Terraform uses HCL v2 - “*HashiCorp Configuration Language*” - to define resources to be created for 1 or more providers.

Note: the same course is available for other Providers

## Pre-requisites

- Be at ease working at the command-line, editing files
- Basic notions of Cloud, Linux, Containers
- Use of an ssh Client, e.g. openssh on Linux, macOS or Windows (or Putty)

## Included

- Course materials and labs: 50% hands-on, 50% presentation & demos
- Access to a temporary lab environment
- Access to an evolutive document covering various Terraform learning resources

## Objectives

- Learn to use Terraform for standing up various Azure resources, in a declarative manner
- Learn to use Azure specific modules, data sources & tooling

**Note:** the following Azure specific resources are proposed, but can be adapted to customer needs

## Programme

**MODULE: Review**
* Infrastructure as Code principles
* Terraform & OpenTofu workflows

**MODULE: Working with Containers**
* Managing Azure ACI containers with Terraform
* Using Data Sources with Azure ACI

**MODULE: Working with VMs**
* Managing Azure virtual machines with Terraform
* Using Data Sources with Azure VMs

**MODULE: Working with modules**
* Terraform registry:: Working with existing terraform modules for Azure
* Writing modules for Azure: Creating clusters of VMs

**MODULE: Templates**
* Creating useful templates (ssh_config, ansible inventory, reports)

**MODULE: In Practice**
* Variable validation
* Debugging
* 3rd-party tools

**MODULE: State**
* Local State
* Using Azure Blob Storage for “remote state”

**MODULE: Importation of foreign ressources**
* Importation of Azure resources
* Move of Azure resources

**MODULE: Auto-scaling & Load-Balancing**
* Azure VM ScaleSets
* terraform-azurerm-loadbalancer

**MODULE: Other Azure resources**

* Azure functions, Azure VPC, Azure Public IP, Blob Storage, AKS

**MODULE: In Production**
* Provisioners (Local-exec, File, Remote-exec)
* Provider Aliases

# TRAININGS: Ansible

## [2d] AN-002-EN Introduction to Ansible

### Description

This training proposed as a 2-day training, or 4 half-days, provides a comprehensive introduction to Configuration Management using Ansible.

**Note:** This training also exists as a 3-day training "*AN-003-EN Introduction to Ansible*" allowing more time for labs and also the addition of "*Ansible Vault*" and "*Network Automation*" modules.

This training includes much hands-on as well as theory.
All hands-on exercises will be performed on Linux Virtual Machines & Containers.

Students will appreciate the ease with which software can be installed & maintained in a reliable, repeatable and reusable manner across teams thanks to the largely declarative nature of Ansible Playbooks & Roles. Playbooks can be re-run guaranteeing repeatable outcomes due to the idempotent nature of Ansible..

### Pre-requisites

* Be at ease working at the command-line, editing files
* Basic notions of Cloud, Linux, Containers
* Use of an ssh Client, e.g. openssh on Linux, macOS or Windows (or Putty)

### Included

* Course materials and labs: 50% hands-on, 50% presentation & demos
* Access to a temporary lab environment
* ccess to an evolutive document covering various learning resources

### Objectives

* Learn to use Ansible for performing software configuration & maintenance of machines/devices
* Learn about the many builtin modules, collections of modules & roles for Ansible
* Learn best practices for working with Ansible

### Programme

**MODULE: Introduction**
* Why Ansible?
* Concepts & Terms
* Agentless Architecture
* Inventory:
  * Static & Dynamic
  * Host & Group Patterns

**MODULE: Deploying with Ansible**
* Installing Ansible
* YAML Configuration Files
* Playbooks
  * Ansible Modules
  * Ad-Hoc Commands

    * Dynamic Inventory:
        * Scripted
        * Plugins

**EXERCISE**
* Deploying Ansible
* Ad-Hoc Commands
* Dynamic Inventories

**MODULE: Playbooks**
* Playbook structure
* Host and Task Execution Order
* Module Categories
    * Command
    * File Manipulation
    * Network Modules
    * Packaging Modules
    * System Storage
    * Account Management
    * Security
    * Services

**EXERCISE**
* Playbook Basics
* Playbooks: Command Modules
* Playbooks: Common Modules

**MODULE: Variables**
* at runtime
* in Playbooks
* register task output, debug module
* in inventory: host_vars, global_vars
* Scope & precedence
* Facts & magic variables
* In included files, roles
* Output: callback plugins

**EXERCISE**
* Variables and Facts
* Inclusions

**MODULE: Jinja Templating**
* Jinja Expressions
* Builtin.template module
* Filters
* Methods
* Conditionals
* Lookup plugins
* Control Structures

**EXERCISE**
* Jinja Templates

**MODULE: Control Structures**
* Loops & Variables (with)
* Conditionals (when)
* Handlers
* Tags
* Error Handling

**EXERCISE**
* Task Control

**MODULE: Roles**
* Usage
* Creating Roles
* Deploying Roles with Ansible Galaxy

**EXERCISE**
* Converting Playbooks to Roles
* Creating Roles from Scratch
* Ansible Galaxy Roles

# [3d] AN-003-EN Introduction to Ansible

## Description

This training proposed as a 3-day training, or 6 half-days, provides a comprehensive introduction to Configuration Management using Ansible.

**Note:** This training also exists as a 2-day training "*AN-002-EN Introduction to Ansible*" excluding "*Ansible Vault*" and "*Network Automation*" modules.

This training includes much hands-on as well as theory.
All hands-on exercises will be performed on Linux Virtual Machines & Containers.

Students will appreciate the ease with which software can be installed & maintained in a reliable, repeatable and reusable manner across teams thanks to the largely declarative nature of Ansible Playbooks & Roles. Playbooks can be re-run guaranteeing repeatable outcomes due to the idempotent nature of Ansible..

## Pre-requisites

- Be at ease working at the command-line, editing files
- Basic notions of Cloud, Linux, Containers
- Use of an ssh Client, e.g. openssh on Linux, macOS or Windows (or Putty)

## Included

- Course materials and labs: 50% hands-on, 50% presentation & demos
- Access to a temporary lab environment
- Access to an evolutive document covering various learning resources

## Objectives

- Learn to use Ansible for performing software configuration & maintenance of machines/devices
- Learn about the many builtin modules, collections of modules & roles for Ansible
- Learn best practices for working with Ansible

## Programme

**MODULE: Introduction**

* Why Ansible?
* Concepts & Terms
* Agentless Architecture
* Inventory:
  * Static & Dynamic
  * Host & Group Patterns

**MODULE: Deploying with Ansible**
* Installing Ansible
* YAML Configuration Files
* Playbooks
  * Ansible Modules
  * Ad-Hoc Commands
  * Dynamic Inventory:
    * Scripted
    * Plugins

**EXERCISE**
* Deploying Ansible
* Ad-Hoc Commands
* Dynamic Inventories

**MODULE: Playbooks**
* Playbook structure
* Host and Task Execution Order
* Module Categories
  * Command
  * File Manipulation
  * Network Modules
  * Packaging Modules
  * System Storage
  * Account Management
  * Security
  * Services

**EXERCISE**
* Playbook Basics
* Playbooks: Command Modules
* Playbooks: Common Modules

**MODULE: Variables**
* at runtime
* in Playbooks
* register task output, debug module
* in inventory: host_vars, global_vars
* Scope & precedence
* Facts & magic variables
* In included files, roles
* Output: callback plugins

**EXERCISE**
* Variables and Facts
* Inclusions

**MODULE: Jinja Templating**
* Jinja Expressions
* Builtin.template module
* Filters
* Methods

* Conditionals
* Lookup plugins
* Control Structures

**EXERCISE**
* Jinja Templates

**MODULE: Control Structures**
* Loops & Variables (with)
* Conditionals (when)
* Handlers
* Tags
* Error Handling

**EXERCISE**
* Task Control

**MODULE: Roles**
* Usage
* Creating Roles
* Deploying Roles with Ansible Galaxy

**EXERCISE**
* Converting Playbooks to Roles
* Creating Roles from Scratch
* Ansible Galaxy Roles

**MODULE: Optimisation**
* Connection Types
* Delegation
* Parallelism
* Callback Plugins

**EXERCISE**
* Optimizing Ansible

**MODULE: Secrets with Ansible Vault**
* Configuring Vault
* Vault IDs
* Working with Vault

**EXERCISE**
* Ansible Vault

**MODULE: Network Automation**
* Simple Network Module Examples
* Debugging Network Modules
* ios Modules
* Simple IOS Modules Examples