DEVOPS & DÉPLOIEMENT

Linux et ligne de commande pour développeurs

Maîtriser le terminal, SSH, les permissions et l'écriture de scripts bash. Le prérequis implicite de toutes les formations techniques, Git, Docker, CI/CD, déploiement.

DURÉE	TARIF HT	NIVEAU	LANGUE	GROUPE	FORMAT
2j (14h)	~~600.00 €~~ **360.00 €**	**Débutant**	**Français**	**1-6**	**Formation**

1 PUBLIC VISÉ

- Développeurs web débutants ou en reconversion qui travaillent encore exclusivement dans des interfaces graphiques et n'ont jamais vraiment appris le terminal.
- Développeurs frontend (HTML, CSS, JavaScript) souhaitant passer au backend ou au DevOps et qui butent sur les prérequis en ligne de commande.
- Apprenants ayant suivi une formation de développement web et qui ressentent un manque de maîtrise dès qu'il s'agit de déployer, de configurer un serveur ou de travailler sur un VPS.
- Profils en reconversion issus d'autres domaines (gestion, design, marketing) qui entrent dans le développement et n'ont aucun bagage sur les systèmes Unix.
- Développeurs autodidactes qui connaissent quelques commandes par cœur mais sans en comprendre la logique, et qui veulent combler ce trou dans leur formation.

2 PRÉREQUIS

- Savoir utiliser un ordinateur : créer des dossiers, installer des logiciels, naviguer dans un navigateur web.
- Avoir un éditeur de texte installé (VS Code recommandé) et un accès à un terminal — Terminal sur macOS, WSL2 ou Git Bash sur Windows, terminal natif sur Linux.
- Aucune connaissance préalable du terminal, de Linux ou de la programmation shell n'est requise. La formation part de zéro.

3 OBJECTIFS PÉDAGOGIQUES

- Naviguer et manipuler le système de fichiers Linux avec aisance depuis le terminal
- Comprendre et gérer les permissions Unix : utilisateurs, groupes, chmod, chown
- Maîtriser les commandes de traitement de texte : grep, sed, awk, sort, cut
- Gérer les processus, les ports et les services depuis le terminal
- Se connecter à un serveur distant via SSH et gérer ses clés de façon sécurisée
- Configurer et personnaliser son environnement shell : .bashrc, alias, variables, PATH
- Écrire des scripts bash utiles et robustes : variables, conditions, boucles, fonctions
- Automatiser des tâches récurrentes : sauvegardes, logs, déploiements simples, tâches planifiées

4 PROGRAMME DÉTAILLÉ

Jour 1 (7h) - Le terminal, le système de fichiers et les outils essentiels

Module 1 - S'orienter dans Linux et le terminal (1h30)

Linux, Unix et le shell (45min)

- Pourquoi Linux : la quasi-totalité des serveurs web, des conteneurs Docker et des pipelines CI tournent sous Linux
- La famille Unix : macOS, Linux, WSL2 - le même modèle, les mêmes commandes
- Le shell : l'interpréteur de commandes - bash, zsh, fish - différences pratiques
- Le terminal vs le shell vs la console : démêler le vocabulaire une fois pour toutes
- L'invite de commande : décrypter user@host:~/projet$ - utilisateur, machine, répertoire courant, niveau de privilège
- Obtenir de l'aide : man, --help, tldr - lire une page de manuel sans se noyer
- Cas pratique : ouvrir un terminal, identifier son shell, lire la page man de ls, installer tldr et comparer les deux niveaux de documentation

Naviguer dans le système de fichiers (45min)

- L'arborescence Linux : /, /home, /etc, /var, /tmp, /usr - à quoi sert chaque répertoire
- Chemins absolus vs chemins relatifs : /home/adel/projet vs ../projet
- Les raccourcis essentiels : ~ (home), . (répertoire courant), .. (parent), - (répertoire précédent)
- Naviguer : pwd, cd, ls avec les options -l, -a, -h, -t
- L'autocomplétion avec Tab : la commande la plus sous-utilisée du terminal
- L'historique des commandes : history, Ctrl+R pour la recherche incrémentale
- Cas pratique : naviguer dans l'arborescence système uniquement avec le terminal - atteindre /etc, revenir au home, explorer un projet web existant sans jamais utiliser l'explorateur de fichiers

Module 2 - Manipuler les fichiers et les répertoires (2h)

Créer, copier, déplacer, supprimer (1h)

- Créer : touch, mkdir avec -p pour créer des arbres de répertoires en une commande
- Copier : cp avec -r pour les répertoires, -p pour préserver les métadonnées
- Déplacer et renommer : mv - la même commande pour les deux opérations
- Supprimer : rm avec -r et -f - comprendre pourquoi rm -rf / est la commande la plus dangereuse qui soit
- Liens symboliques : ln -s - créer des raccourcis, comprendre leur utilité dans les projets
- Wildcards : *, ?, [abc] - sélectionner des fichiers par pattern
- Cas pratique : structurer un projet web depuis le terminal - créer l'arborescence complète, déplacer des fichiers, créer des liens symboliques pour des assets partagés

Lire et éditer des fichiers (1h)

- Afficher le contenu : cat, less, head, tail - quand utiliser lequel

tail -f : suivre un fichier de log en temps réel - l'outil de débogage numéro un sur un serveur

- Éditer depuis le terminal : nano pour les débutants, introduction à vim - savoir a minima ouvrir, modifier et quitter
- Redirection de la sortie : > (écraser), >> (appender), 2> (stderr), &> (stdout + stderr)
- Les pipes | : enchaîner des commandes - la philosophie Unix "faire une chose, la faire bien"
- Compter les lignes, mots, caractères : wc -l, wc -w
- Cas pratique : analyser un fichier de log Apache ou Nginx - afficher les 50 dernières lignes, suivre les nouvelles entrées en temps réel, rediriger les erreurs dans un fichier séparé

Module 3 - Permissions, utilisateurs et groupes (2h)

Comprendre le modèle de permissions Unix (1h)

- Décrypter la sortie de ls -l : -rwxr-xr-- - type, permissions owner, groupe, autres
- Les trois permissions : lecture (r), écriture (w), exécution (x) - leur signification diffère selon fichier vs répertoire
- Les trois entités : owner, group, others - qui est concerné par chaque permission
- Modifier les permissions : chmod en notation symbolique (u+x, go-w) et octale (755, 644, 600)
- Les valeurs courantes expliquées : 755 pour les répertoires, 644 pour les fichiers, 600 pour les clés SSH
- Changer le propriétaire : chown user:group fichier
- Cas pratique : déploiement simulé - corriger les permissions d'une application web (fichiers de config trop ouverts, scripts non exécutables, répertoire d'uploads mal configuré)

Utilisateurs, groupes et élévation de privilèges (1h)

- Les utilisateurs Linux : /etc/passwd, UID, GID, home, shell - lire le fichier sans l'éditer
- Créer et gérer des utilisateurs : useradd, usermod, passwd, userdel
- Les groupes : groupadd, usermod -aG pour ajouter un utilisateur à un groupe
- sudo : exécuter une commande en tant que root - comprendre le fichier /etc/sudoers et visudo
- su vs sudo : différences, cas d'usage, bonnes pratiques de sécurité
- L'utilisateur root : pourquoi ne jamais travailler en root au quotidien sur un serveur
- Le groupe docker : pourquoi l'ajouter à son utilisateur et ce que ça implique en termes de sécurité
- Cas pratique : créer un utilisateur de déploiement dédié sur un VPS simulé - permissions minimales, ajout au groupe docker, sudo restreint à certaines commandes uniquement

Module 4 - Rechercher et traiter du texte (1h30)

grep, find et les expressions régulières de base (45min)

- find : rechercher des fichiers par nom, type, taille, date de modification, permissions
- find + -exec : appliquer une commande à chaque résultat trouvé
- grep : chercher un pattern dans un fichier ou une sortie - options -r, -i, -n, -l, -v
- Les expressions régulières de base dans grep : ., *, ^, $, [], \b
- grep -E (egrep) pour les regex étendues : +, ?, |, ()
- Cas pratique : dans un projet web, trouver tous les fichiers contenant une clé API en dur, lister tous les .env commités par erreur, chercher les TODO dans le code

sed, awk, sort, cut (45min)

- cut : extraire des colonnes depuis un fichier CSV ou une sortie tabulaire - -d pour le délimiteur, -f pour les champs
- sort : trier des lignes - alphabétique, numérique (-n), inverse (-r), dédoublonner avec -u
- uniq : supprimer les doublons consécutifs, compter les occurrences avec -c
- sed : substitution dans un flux de texte - s/ancien/nouveau/g, supprimer des lignes, éditer un fichier en place avec -i
- awk : traiter des fichiers structurés par colonnes - $1, $NF, filtrer et reformater des lignes
- Cas pratique : pipeline de traitement d'un fichier de log Apache - extraire les IP, compter les requêtes par IP, trier par volume décroissant, identifier les 10 IPs les plus actives

Jour 2 (7h) - SSH, processus, environnement et scripts bash

Module 5 - SSH : se connecter et travailler sur un serveur distant (2h)

Connexion SSH et gestion des clés (1h)

- Comment fonctionne SSH : chiffrement asymétrique, clé publique / clé privée - l'essentiel sans les maths
- Se connecter : ssh user@host, ssh -p 2222 pour un port non standard
- Générer une paire de clés : ssh-keygen -t ed25519 -C "commentaire" - pourquoi ed25519 plutôt que RSA
- Déployer sa clé publique : ssh-copy-id, éditer manuellement ~/.ssh/authorized_keys
- Protéger sa clé privée : permissions 600 sur ~/.ssh/id_ed25519, passphrase, ssh-agent
- Désactiver l'authentification par mot de passe sur le serveur : PasswordAuthentication no dans sshd_config
- Cas pratique : se connecter à un VPS Ubuntu sans mot de passe - générer une clé ed25519, déployer la clé publique, désactiver les connexions par mot de passe, vérifier que la connexion fonctionne uniquement par clé

Productivité SSH et transfert de fichiers (1h)

- Le fichier ~/.ssh/config : définir des alias de connexion, spécifier clé, port et utilisateur par hôte
- Transfert de fichiers : scp pour les copies simples, rsync pour la synchronisation incrémentale
- rsync : options essentielles -avz, --delete, --exclude - déployer un site statique en une commande
- Les tunnels SSH : -L pour le port forwarding local - accéder à une base de données distante comme si elle était locale
- Maintenir une connexion SSH active : ServerAliveInterval dans ~/.ssh/config
- Multiplexage SSH avec ControlMaster : réutiliser une connexion existante pour accélérer les connexions successives
- Cas pratique : configurer ~/.ssh/config avec deux hôtes (staging et production), déployer un site statique avec rsync, ouvrir un tunnel vers une base PostgreSQL distante

Module 6 - Processus, ports et gestion des services (1h30)

Gérer les processus et surveiller le système (45min)

- Lister les processus : ps aux, top, htop - lire les informations affichées
- Trouver le PID d'un processus par son nom : pgrep, pidof
- Envoyer des signaux : kill -15 (SIGTERM, arrêt propre) vs kill -9 (SIGKILL, arrêt forcé) - pourquoi commencer par -15
- Les jobs en arrière-plan : &, jobs, fg, bg, Ctrl+Z

Surveiller la mémoire et le disque : free -h, df -h, du -sh *

Ports, réseau et services systemd (45min)

- Voir les ports en écoute : ss -tlnp - identifier quel processus occupe quel port
- Tester la connectivité : curl, wget, ping, traceroute
- Inspecter les en-têtes HTTP depuis le terminal : curl -I, curl -v
- systemd et systemctl : démarrer, arrêter, redémarrer, activer au boot, consulter les logs d'un service
- Les logs système avec journalctl : filtrer par service, par date, suivre en temps réel avec -f
- Cas pratique : diagnostiquer une application web qui ne démarre pas - inspecter les logs systemd, identifier le port bloqué, tuer le processus qui l'occupe, relancer le service

Module 7 - Environnement shell et personnalisation (1h)

Variables, PATH et fichiers de configuration (1h)

- Les variables shell : déclarer, lire avec $VAR, exporter avec export pour les sous-processus
- Les variables d'environnement système importantes : PATH, HOME, USER, SHELL, EDITOR
- Modifier le PATH : ajouter un répertoire pour rendre un binaire accessible depuis n'importe où
- Les fichiers de configuration du shell : .bashrc vs .bash_profile vs .profile - lequel est lu quand
- Créer des alias utiles : alias ll='ls -lah', alias gst='git status'
- Les fonctions shell dans .bashrc : aller plus loin que les alias pour les séquences de commandes
- Recharger la configuration sans relancer le terminal : source ~/.bashrc
- Cas pratique : configurer un .bashrc de développeur - aliases git, navigation rapide dans les projets, prompt personnalisé avec la branche Git courante, PATH étendu avec les binaires locaux

Module 8 - Scripts bash : automatiser ses tâches (2h30)

Écrire son premier script bash (45min)

- Le shebang : #!/usr/bin/env bash - pourquoi cette ligne, et pas #!/bin/bash
- Rendre un script exécutable : chmod +x
- Les variables dans un script : affectation sans espaces, convention de nommage, ${VAR} vs $VAR
- Les arguments positionnels : $1, $2, $@, $#, $0
- Capturer la sortie d'une commande : $(commande) - la substitution de commande
- Les options de sécurité : set -e (arrêt sur erreur), set -u (variables non définies = erreur), set -o pipefail
- Cas pratique : script setup-project.sh - crée l'arborescence d'un projet web, copie les fichiers de config, affiche un résumé de ce qui a été créé

Conditions, boucles et fonctions (1h)

- Les conditions if/elif/else : syntaxe bash, opérateurs de comparaison numérique (-eq, -lt) et de chaînes (=, !=)
- Les tests sur les fichiers : -f (fichier), -d (répertoire), -e (existe), -r (lisible), -z (chaîne vide)
- La boucle for : itérer sur une liste, sur les fichiers d'un répertoire, sur les arguments

La boucle while : attendre qu'une condition soit vraie - utile pour les retries et les healthchecks

- Les fonctions bash : déclarer, appeler, passer des arguments, retourner une valeur via echo ou le code de sortie
- Le code de sortie $? : 0 = succès, non-zéro = erreur - base de toute gestion d'erreur en bash
- Cas pratique : script de vérification d'environnement - vérifie que Node.js, Docker et Git sont installés avec les bonnes versions, affiche un rapport coloré, retourne un code d'erreur si un prérequis manque

Scripts pratiques pour développeurs et automatisation (45min)

- Script de sauvegarde : archiver un répertoire avec tar, horodater le fichier, supprimer les archives de plus de 7 jours
- Script de déploiement simple : git pull, installer les dépendances, redémarrer le service, vérifier le healthcheck
- Automatiser avec cron : la syntaxe crontab -e, planifier une sauvegarde quotidienne, un rapport hebdomadaire
- Redirection des sorties dans un cron job : logger stdout et stderr dans un fichier pour le débogage
- Les bonnes pratiques de scripting bash : commentaires, noms de variables explicites, messages d'erreur clairs, option --dry-run
- Cas pratique : script de déploiement complet - pull git, rebuild Docker, restart Compose, healthcheck avec retry (5 tentatives, 3 secondes d'intervalle), notification de succès ou d'échec

5 COMPÉTENCES VISÉES

- Naviguer et manipuler le système de fichiers Linux avec aisance depuis le terminal
- Comprendre et corriger les permissions Unix sur un serveur ou un projet
- Se connecter à un serveur distant via SSH avec des clés et sans mot de passe
- Diagnostiquer une application en production : logs, processus, ports
- Écrire des scripts bash utiles : sauvegarde, déploiement, vérification d'environnement
- Automatiser des tâches récurrentes avec cron et des scripts robustes

6 MODALITÉS PÉDAGOGIQUES

Formation délivrée en présentiel ou distanciel, exclusivement dans le terminal — aucune interface graphique n'est utilisée pendant la formation. Le formateur alterne entre méthode démonstrative (live coding dans le terminal avec de vraies situations de développeur), méthode interrogative (analyse des erreurs fréquentes et des messages d'erreur cryptiques) et méthode active (exercices guidés de manipulation et d'écriture de scripts). L'accent est mis sur la compréhension du pourquoi derrière chaque commande, pas uniquement la mémorisation.

7 MOYENS PÉDAGOGIQUES ET TECHNIQUES

- Support de cours numérique mis à disposition
- Scripts bash annotés avec exercices et corrections par module
- Environnement : terminal natif (macOS/Linux) ou WSL2 (Windows) + accès à un VPS Ubuntu pour les exercices SSH
- Pour le distanciel : visioconférence, partage d'écran, chat en direct
- Accès à la plateforme pédagogique LaPolaris (supports, ressources, émargement)

8 MODALITÉS D'ÉVALUATION

- En cours de formation : pipelines construits et testés en direct à chaque module
- En fin de formation : pipeline CI/CD complet sur un projet réel avec déploiement automatique
- Questionnaire d'auto-évaluation des acquis en fin de parcours

9 CRITÈRES D'ÉVALUATION

- Navigation et manipulation du système de fichiers sans recours à l'explorateur graphique
- Permissions Unix correctement appliquées : valeurs adaptées aux fichiers, répertoires et clés SSH
- Connexion SSH fonctionnelle par clé sans mot de passe, fichier ~/.ssh/config configuré
- Traitement de fichiers texte avec les outils grep, sed, awk, cut, sort en les combinant par pipes
- Script bash de synthèse fonctionnel avec gestion des erreurs, conditions, boucles et set -euo pipefail
- Tâche planifiée avec cron opérationnelle avec redirection des sorties dans un fichier de log

10 MODALITÉS DE VALIDATION

Attestation de fin de formation délivrée à l'issue du parcours, conditionnée à une assiduité d'au moins 80 % et à la réalisation du script de synthèse. L'attestation précise les objectifs atteints et les compétences acquises.

11 SUIVI ET ACCOMPAGNEMENT

- Feuilles d'émargement signées par demi-journée (présentiel) ou émargement numérique (distanciel)
- Traçabilité des activités pédagogiques réalisées
- Attestation d'assiduité délivrée en fin de formation
- Suivi individuel via la vérification du dépôt GitHub de chaque apprenant en fin de parcours

12 CONDITIONS D'ACCÈS

Formation accessible sur inscription directe, sans prérequis administratif particulier. Le financement peut être pris en charge par l'employeur dans le cadre d'un plan de développement des compétences, ou en autofinancement avec possibilité de paiement en plusieurs fois.

13 DÉLAIS D'ACCÈS

Inscription possible jusqu'à **5 jours ouvrés** avant le début de la session. Pour toute demande urgente, nous contacter directement.

ACCESSIBILITÉ · HANDICAP

Nos formations sont accessibles aux personnes en situation de handicap. Pour toute situation nécessitant un aménagement (matériel, temporel ou pédagogique), nous vous invitons à nous contacter avant l'inscription afin d'étudier les adaptations possibles. Référent handicap : contact@lapolaris.fr

LaPolaris
TÉL. +33762584798
EMAIL contact@lapolaris.fr
WEB lapolaris.fr