+33 (0)1 48 10 09 46
formation@ambient-it.net
140 Avenue Jean Lolive 93500 Pantin - France

Mis à jour le 20/06/2025

S'inscrire

Formation Bun

3 jours (21 heures)

Présentation

Notre formation Bun vous permettra de découvrir et maîtriser ce runtime JavaScript ultra-performant, conçu pour remplacer Node.js, npm et même votre bundler préféré. Vous apprendrez à développer des applications fullstack modernes, à tester et déployer efficacement, tout en exploitant les performances exceptionnelles de Bun. Vous commencerez par comprendre l'architecture unique de Bun, son installation, sa CLI minimaliste, ainsi que son moteur d'exécution capable de gérer nativement JavaScript et TypeScript. L'objectif : être rapidement autonome et productif avec un outil rapide et tout-en-un. Nous aborderons ensuite le cœur de Bun : son bundler intégré, sa gestion des dépendances ultra-rapide, et son test runner natif. Vous verrez comment construire, organiser et maintenir des projets robustes sans recourir à Babel, Webpack ou Jest. Un module complet sera dédié à la création de serveurs HTTP, d'APIs REST, et à la manipulation des WebSockets. Vous apprendrez aussi à interagir avec des bases de données, à sécuriser vos applications et à déployer vos projets avec Docker ou Railway. Comme pour toutes nos formations, celle-ci vous sera présentée avec la toute dernière version stable de Bun.

Objectifs

- Comprendre le fonctionnement interne de Bun et ses différences avec Node.js et Deno
- Savoir créer, tester et exécuter des applications TypeScript sans transpilation externe
- Maîtriser la gestion des paquets, le bundling et le hot reload avec les outils intégrés de Bun
- Concevoir des serveurs HTTP performants et des APIs REST sécurisées avec Bun.serve
- Intégrer des bases de données, WebSockets et middlewares dans une architecture fullstack
- Être capable de déployer et superviser une application Bun en production

Public visé

- Développeurs JavaScript
- Développeurs Fullstack

Pré-requis

- Connaissance du langage JavaScript et connaissance d’un framework côté client
- Connaissance d'un langage typé

Programme de la formation Bun

Introduction à Bun

- Comparaison avec Node.js et Deno
- Installation de Bun (Linux, macOS, Windows WSL)
- Initialisation d’un projet (bun init)
- Structure de projet Bun
- bun run, bun install, bun bun
- Exécuter des scripts TypeScript/JavaScript directement Commandes utiles : bun upgrade, bun pm, etc.

Bun comme Runtime JS/TS

- Fichiers .js, .ts, .tsx, .jsx
- Pas de compilation explicite avec tsc
- fetch, Request, Response, WebSocket, FormData
- API Blob et TextEncoder/TextDecoder
- Modules standards (fs, http, crypto, etc.)
- Limitations et suivi de compatibilité
- Modules CommonJS vs ESModules

Gestion de paquets avec Bun

- Utilisation et comparaison avec npm/yarn Génération de bun.lockb Arborescence node_modules standard Gestion des packages natifs ou compilés Limitations et résolution des erreurs Imports URL, packages globaux, polyfills intégrés

bundler et transpileur

- Bundling d’un projet en un fichier
- Minification, tree-shaking et sourcemaps
- Types de fichiers pris en charge (.ts, .jsx, .tsx)
- Pas de Babel ou TypeScript Compiler nécessaires
- Introduction aux futurs plugins de bundling
- Extensibilité du système

Serveurs HTTP et développement web

- Utilisation de Bun.serve
- Gestion des routes, des headers, du JSON
- Création d’une API complète avec Bun
- Paramètres, body, status, JSON
- Utilisation de bun --watch
- Redémarrage automatique des serveurs

Tests avec Bun

- Écriture de tests unitaires avec le framework intégré Syntaxe (test(), expect()) Fichiers *.test.ts Groupes, setup/teardown

Mocking & snapshot testing

- Mock de fonctions et modules
- Support de snapshots (en développement)

Bun + Frontend

- React, Preact, Vue, Svelte (support via bundler)
- Utilisation de JSX/TSX avec Bun
- Fichiers CSS, JSON, SVG
- Comportement de résolution des imports
- Premiers pas avec SSR en Bun
- Limites actuelles et bonnes pratiques

Bun : Backend & Fullstack

- SQLite intégré (via bun:sqlite)
- Utilisation de Prisma ou Drizzle
- Création de serveurs WebSocket
- Communication en temps réel
- Cookies, sessions, headers
- Création de middlewares simples

Packaging et déploiement

- Compilation avec bun build
- Déploiement sur Vercel, Railway, Docker

Monitoring et logging

- Logs natifs et format JSON
- Intégration avec outils externes (Sentry, Logtail)

Performance et benchmarks

- Profilage d’une application Bun
- Analyse comparative vs Node.js

Sociétés concernées

Cette formation s’adresse à la fois aux particuliers ainsi qu’aux entreprises, petites ou grandes, souhaitant former ses équipes à une nouvelle technologie informatique avancée ou bien à acquérir des connaissances métiers spécifiques ou des méthodes modernes.

Positionnement à l'entrée en formation

Le positionnement à l'entrée en formation respecte les critères qualité Qualiopi. Dès son inscription définitive, l'apprenant reçoit un questionnaire d’auto-évaluation nous permettant d’apprécier son niveau estimé sur différents types de technologies, ses attentes et objectifs personnels quant à la formation à venir, dans les limites imposées par le format sélectionné. Ce questionnaire nous permet également d’anticiper certaines difficultés de connexion ou de sécurité interne en entreprise (intraentreprise ou classe virtuelle) qui pourraient être problématiques pour le suivi et le bon déroulement de la session de formation.

Méthodes pédagogiques

Stage Pratique : 60% Pratique, 40% Théorie. Support de la formation distribué au format numérique à tous les participants.

Organisation

Le cours alterne les apports théoriques du formateur soutenus par des exemples et des séances de réflexions, et de travail en groupe.

Validation

À la fin de la session, un questionnaire à choix multiples permet de vérifier l’acquisition correcte des compétences.

Sanction

Une attestation sera remise à chaque stagiaire qui aura suivi la totalité de la formation.