



# ORGANISME DE FORMATION AUX TECHNOLOGIES ET METIERS DE L'INFORMATIQUE

## Formation Node.js

*Des bases solides pour développer des applications back-end en JavaScript*

N° ACTIVITÉ : 11 92 18558 92

TÉLÉPHONE : 01 85 77 07 07

E-MAIL : inscription@hubformation.com

Node.js est un ensemble de technologies permettant d'exécuter du code JavaScript mais aussi de gérer son projet logiciel et ses dépendances avec d'autres bibliothèques. Node.js s'est imposé comme une technologie majeure du développement JavaScript fullstack grâce à son approche pragmatique et performante. Une grande majorité des projets utilisant du JavaScript mettent en oeuvre Node.js que ce soit pour créer des sites web, développer des web services ou des API REST, ou comme socle de construction d'applications front-end ou back-end.

Référence	NODEJS
Durée	3 jours (21h)
Tarif	à partir de 1 590 €HT

### Objectifs

- | Identifier l'architecture de Node.js
- | Installer, configurer et déployer des applications JavaScript s'exécutant sous Node.js
- | Développer des applications asynchrones non-bloquantes avec Node.js
- | Mettre en oeuvre les principaux modules built-in de Node.js: système de fichiers, serveurs réseau et http(s)
- | Développer des applications web dynamique et des API REST avec Node.js

### Public

- | Développeurs, intégrateurs, architectes logiciel, chefs de projet technique

### Prérequis

- | Avoir suivi le cours JavaScript: Les fondamentaux du langage ou avoir des connaissances équivalentes

### Programme de la formation

#### Introduction à Node.js

- | Qu'est-ce que Node.js ?
- | Les outils: node et npm
- | Environnement de développement Node.js
- | Notions d'architecture Node.js: IO, asynchrone, non bloquant, concurrence, event-loop, scalability
- | Ateliers: Installation de Node.js et d'un environnement de développement /
- Démonstration de la event loop à travers un premier programme asynchrone / non-bloquant / REPL

#### Évènements

- | Architecture de Node.js basée sur les évènements
- | Event et EventEmitter
- | Synchron vs Asynchrone
- | Gestion des évènements multiples ou unique
- | Gestion des erreurs
- | Ateliers: Développement d'EventEmitter personnalisé

#### Streams

- | Exemples de données streaming gérées par Node.js
- | Les différents types de Streams

## PROCHAINES SESSIONS

Pour connaître les prochaines dates ou organiser un intra-entreprise, contactez-nous, nous vous répondrons sous 72 heures.

- | Buffers internes
- | Pipelines
- | Ateliers:
- | Consommer une stream Readable (http, fs)
- | Ecrire dans une stream Writable (fs)
- | Exploiter une stream Duplex (net.Socket)
- | Utiliser une stream Transform (zlib)

### **Gestion d'erreurs**

- | Les différents types d'erreurs
- | Gestion d'erreur dans les APIs synchrones
- | Gestion d'erreur dans les APIs asynchrones
- | Assertions
- | Logging via l'API Console
- | Debugger un programme Node.js
- | Ateliers:
- | Gestion d'erreur dans les APIs synchrones
- | Gestion d'erreur dans les APIs asynchrones
- | Débugger un programme Node.js dans son IDE ou dans Chrome

### **Modularité**

- | Object Global et scope inter-modules
- | Objects accessibles globalement
- | Modules, scopes et gestion des fichiers
- | Core modules
- | Module main
- | Algorithme de chargement des modules et cache de modules
- | NPM et modules tiers
- | Ateliers:
- | Utilisation de l'objet Global
- | Objets accessibles globalement: process
- | Usage de core modules: timer, OS, Util, Path, ...
- | Création de module interne à l'application
- | Création d'un projet Node.js via NPM et installation de modules populaires

### **Clients/Serveurs TCP et UDP**

- | Module Net pour création de serveurs et clients TCP
- | Module UDP/Datagram pour création serveurs et clients UDP/Datagram
- | Ateliers: Création de clients / serveur TCP / Création de clients / serveur UDP/Datagram

### **Clients/Serveurs http, https et http/2**

- | Modules http, https et http/2 pour la création de serveurs et clients http, https et http/2
- | Module url pour exploiter les requêtes http
- | Ateliers:
- | Création de serveurs et clients http, https et http/2 dans le cadre d'un site web
- | Création de serveurs et clients http, https et http/2 dans le cadre d'un webservice RESTful

### **Interactions avec le système de fichiers**

- | Le module fs et sa proximité avec POSIX
- | APIs synchrones vs APIs asynchrones
- | Ateliers: Interactions avec le système de fichiers via APIs synchrones et asynchrones

### **Développement de back-end web**

- | Gestion des données avec bases de données Relationnelles (Mysql) ou NoSQL (Mongo, Redis)
- | Gestion du middleware (connect, expressjs)
- | Gestion de templates (ejs)
- | Ateliers: Création d'un back web statique et dynamique avec express.js / Création d'une API RESTful avec express.js

### **Déployer une application Node.js**

- | NPM: les principales commandes
- | NPM: packager un module
- | Différents modes de déploiement d'un projet Node.js
- | Modules Forever et PM2
- | Ateliers: Packager une librairie / Dockerizer une application web Node.js

## Méthode pédagogique

Chaque participant travaille sur un poste informatique qui lui est dédié. Un support de cours lui est remis soit en début soit en fin de cours. La théorie est complétée par des cas pratiques ou exercices corrigés et discutés avec le formateur. Le formateur projette une présentation pour animer la formation et reste disponible pour répondre à toutes les questions.

## Méthode d'évaluation

Tout au long de la formation, les exercices et mises en situation permettent de valider et contrôler les acquis du stagiaire. En fin de formation, le stagiaire complète un QCM d'auto-évaluation.

---

## Accessibilité



Les sessions de formation se déroulent sur des sites différents selon les villes ou les dates, merci de nous contacter pour vérifier l'accessibilité aux personnes à mobilité réduite.

Pour tout besoin spécifique (vue, audition...), veuillez nous contacter au 01 85 77 07 07.