Ambient IT
128 Boulevard Macdonald
75019 Paris

Mail : formation@ambient-it.net
Tél : +33 (0)1 40 34 11 53

AMBIENT-IT

Formation Node.JS

4 jours (28 heures)

Présentation

Node.js est une plateforme logicielle libre et événementielle, basée sur le moteur V8 de Google, qui permet le développement d'application réseau en JavaScript tout en garantissant d'excellentes performances.

L'atout majeur de Node.js® réside dans la possibilité d'employer un unique langage de programmation, le JavaScript, à travers toutes les couches d'une architecture logicielle, facilitant ainsi la rationalisation de la base de code et la communication au sein de l'équipe technique.

L'outil est utilisé en production par un grand nombre d'entreprises technologiques (parmi elles LinkedIn, PayPal et Netflix).

Il est supporté par les principaux fournisseurs de cloud (AWS, Google App Engine, Microsoft Azure).

Comme tous nos programmes, notre formation porte sur la toute dernière version stable en date de cette plateforme (Node 11.11 Current à la date de l'article).

Objectifs

- Savoir utiliser NodeJS, NPM et son écosystème dans les dernières versions
- Développer une application web avec NodeJS et ES2017
- Maîtriser la programmation événementielle & asynchrone
- Sécuriser, industrialiser, tester & déployer son application

Public visé

Développeurs web

Pré-requis

Connaissance du langage JavaScript et connaissance d’un framework côté client seraient un plus ou connaissance d’un autre langage de programmation orienté objet (java, php, etc...)

Programme de la formation Node.JS

Jour 1 - Introduction aux principes fondamentaux

Rappel de Javascript

- L’histoire du langage
- Les principes fondamentaux du langage
- La boucle d’événement ou l’Event Loop
- Les moteurs javascript
- Focus sur le moteur V8 de Google

ECMAScript 8 ou ES2017

- Initiation à ECMAScript 8
- Déclaration de variables et portée
- Littéraux objets
- Le format JSON
- Les classes
- Déstructuration
- Rest et Spread
- Template strings
- Les fonctions fléchées
- Les modules ES
- Compatibilité native Node
- Utiliser la dernière version de javascript grâce à Babel

Programmation asynchrone

- Les callbacks
- Les callbacks selon Node.Js
- Le problème du “callback hell”
- Utiliser async.js pour éviter le callback hell
- Les promesses
- Async control flow avec async / await

Introduction à NodeJs

- La genèse de NodeJs
- Exécuter du javascript côté serveur
- Installation du serveur Node.js
- Un premier programme
- Exécuter un fichier
- Présentation global de l’API de Node.Js
- Comparaison avec d'autres technologies

Les objets globaux

- Focus sur la documentation de l’API de Node
- L’objet global et différence avec window
- Utilisation des fonctions setTimeout, setInterval et setImmediate
- logging sur process.stdout avec console
- Accès au context du fichier avec __dirname et __filename
- Accès à la configuration hardware du server process et os

Jour 2 - Manipulation de l’API de Node

Gestion des modules Node

- Qu’est-ce qu’un module Node ?
- Les modules core
- Import de module avec require et import
- Configuration de module et initialisation de module
- Utilisations des modules utilitaires (util, path, queryString, url)
- Création de module

Découverte de NPM

- Le gestionnaire de paquet
- L’outil en ligne de commande npm
- L’alternative yarn
- Recherche de module en ligne de commande
- Le site npmjs.com
- Recherche de module sur le site
- Installation local ou global
- Packaging de module
- Le fichier package.json
- Déclaration des dépendances
- Gestion des conflits de version
- Gestion de dépendances par environnement

Manipulation de fichier

- Présentation du module fs
- Lecture de fichier synchrone
- Lecture de fichier asynchrone
- Création de fichier asynchrone
- Suppression de dossier asynchrone

Programmation événementielle

- Pourquoi la programmation événementielle
- Présentation du module events
- Utilisation de EventEmitter
- Exemple d’utilisation concret

Jour 3 - Développement d'application web

Accès au réseaux depuis NodeJs

- Rappel de réseau
- Les modules core Node orienté réseaux
- Utilisation des module udp et net
- Utilisation des module http et http2
- Utilisation du module dns
- Zoom sur le protocol HTTP

Création d’un serveur web avec l’api Node.JS

- Qu’est-ce qu’un serveur HTTP ?
- Lancement d’un serveur web Node
- Gestion des requêtes/réponses HTTP
- Mise en place d’un gestionnaire de routes
- Traitement de requête de manière asynchrone

Création d’un serveur web avec Express

- Introduction à Express
- Lancement d’un serveur express
- Configuration d’une application Express avec les middlewares
- Utilisation du gestionnaire de routes d’Express
- Les moteurs de templating
- Création de template Pug et rendering d’une page HTML
- Traitement de formulaire HTML

Connexion à une base de données

- Les base de données compatibles
- Introduction à MongoDB
- Utilisation du package mongoose : création de modèle et requêtage
- Lier une route à un modèle mongoose
- Restifier un modèle de données avec express-restify-mongoose
- Utilisation du package sequelize : création de modèle et requêtage
- Lier une route à un modèle sequelize
- Restifier un modèle de données avec finale-rest

Communication bidirectionnelle temps réel

- Introduction à Websocket
- Présentation de socket.io
- Gestion de la communication côté serveur
- Gestion de la communication côté client

Jour 4 - Industrialisation d’une application Node.Js

Builder votre projet

- Pourquoi builder un projet nodeJs ?
- Les outils de build
- Rédiger ses propres scripts
- Partir d’un projet boilerplate (style Yeoman)

Tester et déboguer

- Les modules Node core pour tester et déboguer (console, debugger, inspector, repl, assert)
- L'écosystème des packages npm orienté testing (unitaire et intégration)
- Modules d'assertion : assert et Chai
- Tester son module avec Mocha

L'écosystème des packages NPM

- Bien choisir un package npm: analyse de viabilité
- Les principaux frameworks de développement d'API
- Qui sont les développeurs de package npm ?
- Comment contribuer à un package npm ?

Sécurisation d'une application Node/Express

- Les modules core de sécurité (crypto, https, tls)
- Encryption de mot de passe avec bcrypt
- Le package helmet
- Authentification avec Passeport

Faciliter le développement d'application Node en équipe

- Versionner proprement votre code avec git
- Documentation du code avec docco
- Documentation d'une API à l'aide de Swagger
- Harmonisation d'une base de code à l'aide de ESLint
- Imposer le typage via Typescript ou Flow

Modules Optionnels (+ 1 jour)

Gestion de streams et buffer

- Qu'est-ce un stream ou flux ?
- Comparaison entre les usages de streams unix et nodejs
- Les types de streams: readable, writable, duplex et transform
- La classe Buffer
- Un exemple d'usage haut niveau

Déploiement d'application Node

- Déploiement du code sur Heroku
- Déploiement du code sur AWS
- Containerisation d'application Node avec Docker
- Gestion de processus en environnement de production avec PM2
- Intégration continue avec Jenkins et TravisCI

Sociétés concernées

Cette formation s'adresse aux entreprises, petites ou grandes, souhaitant former ses équipes à une nouvelle technologie informatique avancée.

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 multiple permet de vérifier l'acquisition correcte des compétences.

Sanction

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