

NODE.JS, PROGRAMMATION JAVASCRIPT CÔTÉ SERVEUR

CODE STAGE : W-NOD

OBJECTIFS

Installer et configurer un serveur Node.js

Mettre en œuvre les concepts de la programmation événementielle et asynchrone

Mettre en place un framework Web

Manipuler l'API de Node.js

Gérer la persistance dans une base de données NoSQL avec un ODM

DURÉE

4 jours

PUBLIC

Développeurs, architectes techniques et chefs de projet Web.

PRÉ-REQUIS

Bonnes connaissances du langage Javascript. Une première approche d'un framework JavaScript (côté client) serait un plus.

PROGRAMME

Du JavaScript côté serveur

Genèse du JavaScript côté serveur et de Node.js.

Le moteur Google V8 utilisé côté serveur.

Pourquoi utiliser la programmation événementielle ?

Rappels JavaScript : callbacks, closures, notion de scope, apply, bind, call.

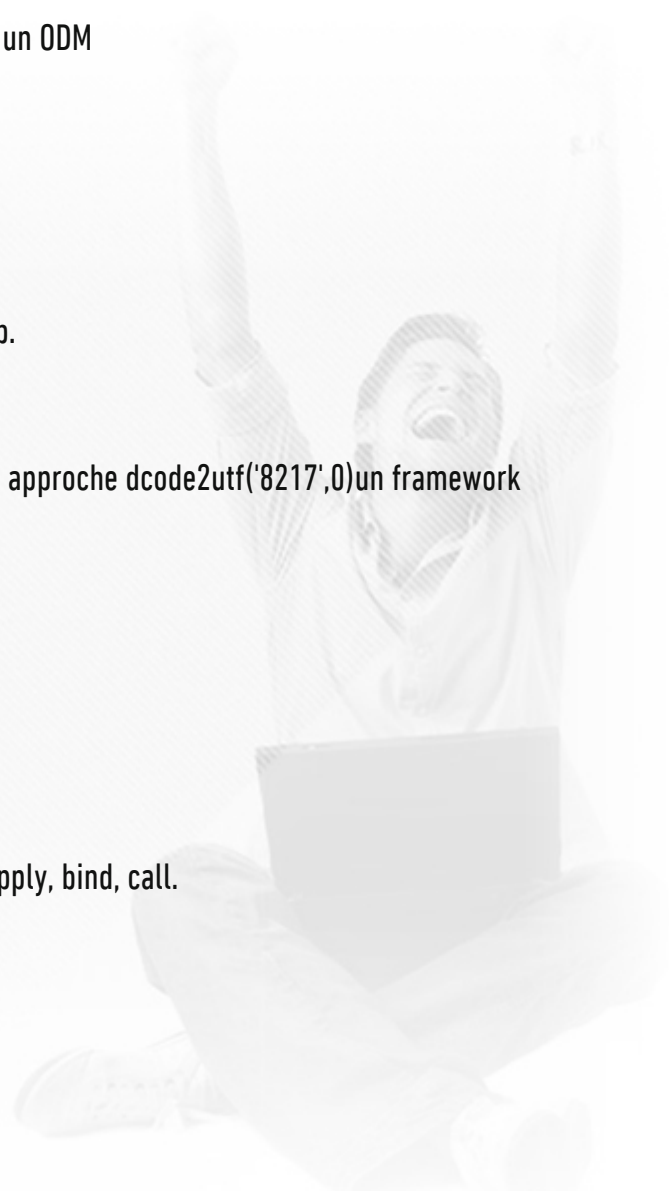
Initiation à ECMAScript 6.

Premiers pas en Node.js

Installation du serveur Node.js.

Le gestionnaire d'extensions NPM.

Approche modulaire de Node.js



Un serveur Web en quelques lignes.

Utilisation de Node.js en REPL.

Travaux pratiques

Usage de `code2utf('8217',0)` utilitaire NPM. Développer une première application.

Les fondamentaux Node.js

Quel intérêt de développer en asynchrone ?

La gestion événementielle : réagir plutôt que `code2utf('8217',0)` attendre.

Principaux modules de `code2utf('8217',0)` API : console, util, file, events `code2utf('038',0)` timer...

Gestion des requêtes/réponses HTTP. HTTPS.

Création de processus fils, https, sockets TCP et UDP...

Travaux pratiques

Lecture de fichier, lecture `code2utf('8217',0)` une ressource en ligne, création `code2utf('8217',0)` un processus fils avec communication IPC.

La gestion de routes

Parsing `code2utf('8217',0)` URL (paramètres, requête...).

Traiter une requête en mode asynchrone.

Mettre en place un gestionnaire de routes.

Travaux pratiques

Mise en place `code2utf('8217',0)` un gestionnaire de routes.

Framework Web

Les concepts fondamentaux `code2utf('8217',0)` Express.

Construction `code2utf('8217',0)` un squelette `code2utf('8217',0)` application.

Configuration `code2utf('8217',0)` Express et de `code2utf('8217',0)` application.

Le rendu de vues avec EJS.

La gestion de formulaires et des uploads de fichiers.

Le routage `code2utf('8217',0)` URL par Express.

Mise en place `code2utf('8217',0)` une API REST.

Travaux pratiques

Mise en place `code2utf('8217',0)` une API REST complète.

Persistence des données

Initiation à une base NoSQL : MongoDB.

Mise en place de CRUD (Create, Read, Update, Delete).



Utilisation dcode2utf('8217',0)un ODM : Mongoose.

Travaux pratiques

Gestion dcode2utf('8217',0)une persistance au travers dcode2utf('8217',0)un ODM avec création des modèles.

Test dcode2utf('8217',0)une application Node.js

Premiers pas avec Mocha.

Les assertions et le test synchrone et asynchrone.

Différentes méthodes de tests (inclusifs, exclusifs...).

