Node.js

Duration 3 day(s) (NODEJS-03)

Master the power of JavaScript on the server side

Description

Node.js is a Web platform for developing server-side asynchronous applications in JavaScript languages, offering new possibilities to manage a large volume of users.

Goals

- Know the architecture and operation of Node.js
- Know how to install and configure Node.js and its ecosystem of plugins
- Understand the concepts of an asynchronous architecture
- Know how to create Node.js applications
- Implement best practices for developing and using Node.js

Public

- Developer
- Software Architect

Prerequisites

- Know and master JavaScript fundamentals
- A first experience in server development

Structure

40% Theory, 60% Practice

Program

Review of JavaScript best practices

- Visibility of the variables
- Code structuring
- Closures
- Objects and prototypes
- Lodash

Introduction to Node.js

- Origin of the project
- The Chrome V8 interpretation engine
- The concept of event management
- The different uses
- Overview of the ecosystem of plugins
- Installation
- A very first example

Node.js architecture

- Asynchronism
- Programming by callbacks
- The event loop

Modules and dependency management

- The modular approach
- NPM and manipulation of the modules
- The package.json file in detail
- Modularization of its code
- Core modules: console, process, os, fs, path and util
- Publish a module on NPM

Node and the Web: HTTP, Connect & Express

- Perimeter of the HTTP module
- Connect and its middlewares
- Web server with Express
- Generate an Express generator server
- Router queries
- Managing a request and creating a response
- Use a template engine
- Opening at Passport and Hapi

Asynchronous details

- Callback Hell & Pyramid of Doom
- The async module
- The promises
- Examples of concatenation and parallelization
- Error management

Real-time communication

- Integration of HTML5 WebSockets
- Socket.io
- Transport Management
- Client side and server side integration

The management of the streams

- Description of Streams2
- Buffers
- Type of flow
- Pipelining
- Object mode
- The tools: through2, trumpet, JSONStream ...
- Gulp

Link with persistence of data

- Approach via the driver or an ORM
- Integration with a relational database
- SQL abstraction modules
- The Sequelize module
- Integration with a NoSQL database
- The Mongoose module

Tooling and Software Factory

- Editing tools
- Use of logs
- Debugging tools
- Unit tests with Mocha
- Chai and Otherwise Modules
- Nodemon
- The orchestration of tasks (build) with Grunt
- Integration in Jenkins integration server

Node.js in Cluster mode

- Using the cluster module
- Master and workers
- Messaging
- Error management

Beyond Node.js

- MEAN Stack
- Deploying Node in the Cloud
- Example with AWS, Heroku
- Monitoring with pm2 and NewRelic