Introduction to Node.js (TT4153)

Overview

Node.js is a server-side JavaScript platform using an event-driven, non-blocking I/O model allowing users to build fast and scalable data-intensive applications running in real time. This fast-paced hands-on course provides the core skills required to develop web applications with Node.js. You will progress from a rudimentary knowledge of JavaScript and server-side development to being able to create, maintain and test your own Node.js applications. You will explore the importance of transitioning to functions that return Promise objects, and the difference between fs, fs/promises and fs-extra, as well as how to use the HTTP Server and Client objects, and data storage with both SQL and MongoDB databases.

Prerequisite Comments

• Introduction to HTML5, CSS3 and JavaScript
• Introduction to JavaScript

• Introduction to HTML5, CSS3 and JavaScript
• Introduction to JavaScript

Target Audience

Incoming attendees are required to have current, hands-on experience in developing basic web applications. Student should have some experience with HTML and CSS and be well versed in JavaScript. Experience with coding for the server side would be helpful.

Course Objectives

This skills-focused course is approximately 50% hands-on. Our engaging instructors and mentors are highly experienced practitioners who bring years of current "on-the-job" experience into every classroom.
Working in a hands-on learning environment, guided by our expert team, attendees will learn to:
Learn server-side JavaScript coding through Node.js
Explore the latest JavaScript features, and ECMAScript modules
Walk through different stages of developing robust applications using Node.js
Install and use Node.js for development
Use the Express application framework
Work with REST service development using the Restify framework
Use data storage engines such as MySQL, SQLITE3, and MongoDB

Course Outline

1 - Overview of Node.js

The capabilities of Node.js
Why should you use Node.js?
The Node.js event-driven architecture
Embracing advances in the JavaScript language
Developing microservices or maxiservices with Node.js

2 - Setting Up Node.js

System requirements
Installing Node.js using package managers
Installing from the source on POSIX-like systems
Installing multiple Node.js instances with nvm
Requirements for installing native code modules
Choosing Node.js versions to use and the version policy
Choosing editors and debuggers for Node.js
Running and testing commands
Advancing Node.js with ECMAScript 2015, 2016, 2017, and beyond
Using Babel to use experimental JavaScript features

3 - Exploring Node.js Modules

Defining a Node.js module
Finding and loading modules using require and import
Using npm – the Node.js package management system
The Yarn package management system

4 - HTTP Servers and Clients

Sending and receiving events with EventEmitter
Understanding HTTP server applications
HTTP Sniffer – listening to the HTTP conversation
Web application frameworks
Getting started with Express
Creating an Express application to compute
Fibonacci numbers
Making HTTPClient requests
Calling a REST backend service from an Express application

5 - Your First Express Application

Exploring Promises and async functions in Express router functions
Architecting an Express application in the MVC paradigm
Creating the Notes application
Theming your Express application
Scaling up – running multiple Notes instances

6 - Implementing the Mobile-First Paradigm

Understanding the problem – the Notes app isn't mobile friendly
Learning the mobile-first paradigm theory
Using Twitter Bootstrap on the Notes application
Flexbox and CSS Grids
Mobile-first design for the Notes application
Using third-party custom Bootstrap themes

7 - Data Storage and Retrieval

Remembering that data storage requires asynchronous code
Logging and capturing uncaught errors
Storing notes in a filesystem
Storing notes with the LevelDB datastore
Storing notes in SQL with SQLite3
Storing notes the ORM way with Sequelize
Storing notes in MongoDB