Node

Naveen Pete

What is Node.js?

- Node.js, or Node for short is a runtime for JavaScript.
  - It can compile and execute JavaScript scripts
  - It does this using the same V8 JavaScript engine that executes JavaScript in the Chrome browser

- It provides libraries written in a mix of JavaScript and C++ (called modules) that provide access to filesystem, spawn web servers, read platform (OS) and memory information, perform cryptographic operations, etc.

- Thus, Node can be used to:
  - Build backend web applications
  - Host front-end web applications (like Angular, React apps)
  - Build command-line utilities
  - Even build desktop apps (combining it with a tool like Electron)

Features of Node

- Relies on asynchronous APIs for achieving I/O– unlike traditional languages like Java which rely on multi-threading
  - *HTTP requests are handled by callback functions executed on a single thread (called the main thread)*
  - *DB requests over the network are again non-blocking asynchronous requests*

- Event-driven
  - *Many of Node's objects like web servers, request/response objects, file streams trigger events to communicate with their consumers*

- Coding in JavaScript
  - *Since V8 is the engine, ES2015+ features can be used (post Node v6). V8 is also reputed to be very fast*

Features of Node

- Built with scalability in mind
  - *Easy to scale both horizontally (add more machines to a cluster), and also vertically (have many cores and create a cluster of servers)*

- No multi-threading (in any of the JavaScript code, application code or otherwise)
  - *Many of Node's objects like web servers, request/response objects, file streams trigger events to communicate with their consumers*
  - *Multi-threading is heavy when compared to non-blocking asynchronous APIs – results in more HTTP connections per server instance*

- Comes with Node Package Manager (npm) and npmjs.com
  - *A robust package manager and an ecosystem of Node modules (packages)*

Advantages

- Performs well for I/O bound applications
  - *Web apps, chat applications, apps for collecting analytics data etc. are a good fit*
- No need for multithreading - less buggy and simpler code
- Built with scalability in mind – easy to create a cluster of Node servers
- JavaScript for the full-stack (especially when combined with a DB like MongoDB)
  - *Unopinionated, Easy to learn, easy to find developers, server-side rendering is easy*
- Less boilerplate code leads to faster development cycles
- Active community – easy to find support, third-party modules, and other resources

Disadvantages

- Performance is not good for CPU-intensive applications - e.g. for a web server, one long-running operation on a request, will block all other requests
- Callback hell makes code difficult to reason about at times
- Not easy to build apps standalone - Need a framework like Express
- Does not have as well-equipped standard libraries as in other languages
  - *Forced to choose a third-party module for many common tasks, which may be hard to choose*