Introduction

Hello and welcome to our backend track. In this track we are going to learn how to create a backend server using Node.js, Express and MongoDB. These are the technologies that we are going to use during this track.

Let’s start with Node.js. In this Super Skill, we are going to:

- Discover what Node.js is.
- Learn how to setup our environment and create our first application.
- Understand how to use the REPL terminal.
- Understand the node package module and the concept of modules in Node.js.
- Practice working with event loops, callback functions and file systems.

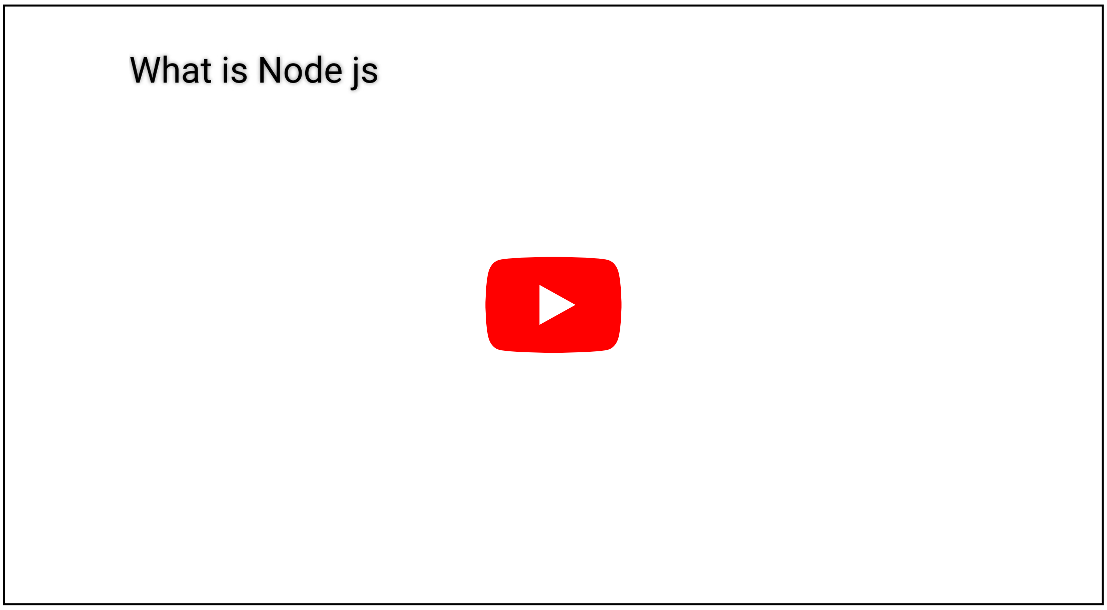
What is Node.js?

Node.js is a server-side platform built on Google Chrome's JavaScript Engine (V8 Engine). Node.js was developed by Ryan Dahl in 2009. The definition of Node.js as supplied by its official documentation is as follows:

> Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

- Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications.
- Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.
- Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

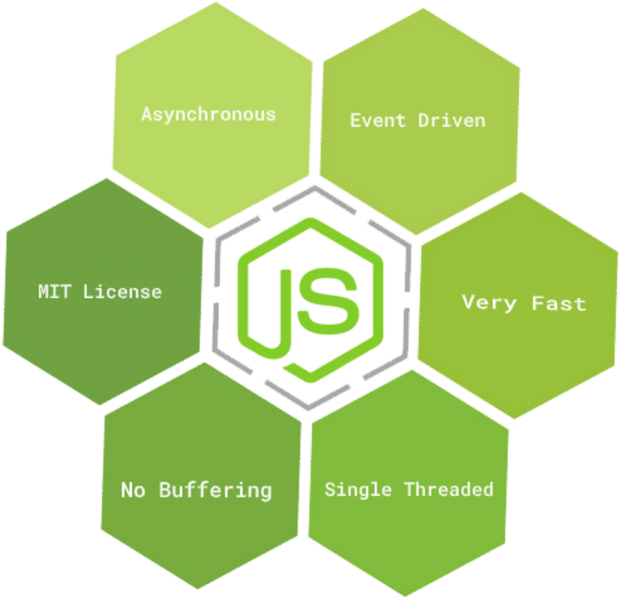
```
Node.js = Runtime Environment + JavaScript Library
```



Features of Node.js.

The Following are some of the important features that make Node.js the software architects' first choice:

- **Asynchronous and Event Driven** – All APIs of Node.js library are asynchronous, that means they are non-blocking. It essentially means a Node.js based server never waits for an API to return data. The server moves on to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call.
- **Very Fast** – Being built on Google Chrome's V8 JavaScript Engine, Node.js library is very fast in code execution.
- **Single Threaded but Highly Scalable** – Node.js uses a single threaded model with event looping. Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers which create limited threads to handle requests. Node.js uses a single threaded program and the same program can provide service to a much larger number of requests than traditional servers like Apache HTTP Server.
- **No Buffering** – Node.js applications never buffer any data. These applications simply output the data in chunks.
- **License** – Node.js is released under the MIT license.



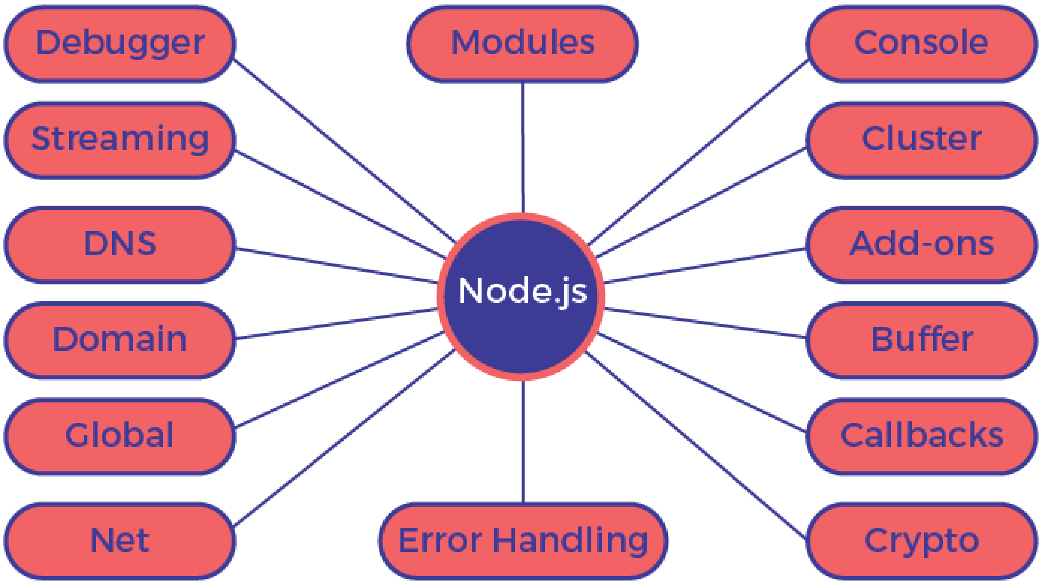
Who Uses Node.js?

It's now nearly impossible to find a complete list of projects, applications and companies that do not use Node.js. Nowadays nearly everyone has adopted it in one way or another. Nonetheless, here's a glimpse of some Top Techs using Node.js.

ebay
NETFLIX
PayPal
in
Uber

Concepts

The following diagram depicts some important parts of Node.js which we will discuss in detail in the subsequent chapters.



When to Use Node.js?

The following areas are the areas where Node.js is proving itself to be a perfect technology partner.

- I/O bound Applications
- Data Streaming Applications
- Data Intensive Real-time Applications (DIRT)
- JSON API-based Applications
- Single-Page Applications

When Not to Use Node.js?

It is not advisable to use Node.js for CPU intensive applications.

‹ Previous

next ›