Why use Node.js – look behin es of web development

Alexander Sokhanych

I have founded company in 2011 with mission to provide IT & Software experience worldwide.

We may receive compensation when you click on links but we are comm to editorial standards and review process.

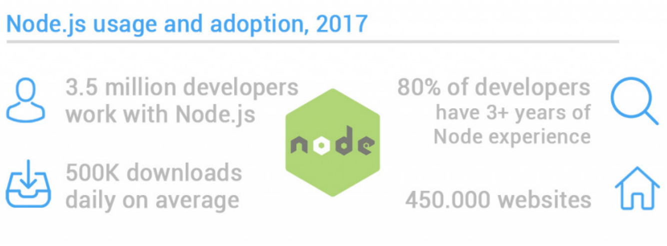
Created: *June 2, 2017* Updated: *August 6, 2021*

JavaScript as programming language and data format (JSON) has changed web development drastically. Integrating Node.js with it to do things on the server as well as in browser is a trend lately. These two sentences, we feel, have to be illuminated and explained for everyone to grasp. So in this article, we are going to talk about why use Node.JS, what is Node.js used for and top examples of Node JS use.

Surely, this is technological stuff for programmers/coders, and some might find the language not quite apprehendable. With this in mind, we will try to make it as light-some as possible and speak in more human terms. We just want to explain **what makes Node.js great** and what is all the hype about.

First off, the praise is well deserved, as Node decisively eased the work of anyone building web applications. After decades of web request/response paradigm, having **real-time 2-way communication** is a bliss. That is a communication between the server and the client. And that, in turn, is a model to distribute workloads among service providers (server) and service requests (clients).

On board so far? Great, now some statistical info about the demand for Nodejs. It is by far the fastest growing language in use, and it ranks in Top-10 most wanted developer skills. The use of Node.js is mainly for full stack, front-end, and back-end.



What is Node js?

As we've previously touched on this in Best Node.js examples, **Node JS is a JavaScript runtime environment**. But what is that, one might ask. By run-time environment, the infrastructure to build and run software applications is meant. To build applications in JavaScript, in this case. Let's see what are the Node JS definition versions.

The company itself describes Node as a “Javascript runtime built on Chrome V8 engine”. Wikipedia states, that Node.js is an open-source and cross-platform environment to execute code. According to Techtarget, it is a development platform aimed at building server-side applications. And PCMag tells us that Node is a platform with its own web server for better control. That is certainly enough to grasp the main idea.

A brief summary would be as goes:

- Node JS is a server framework, and is free
- It runs on Windows, Linux, Mac OS, etc.
- Node utilizes JavaScript on the server

How does Node JS work? Taking a simple task of opening a file on a server, the sequence would be:

- A task goes to the file system
- The system is ready for next requests
- When a file is opened and read, the system sends the content to the client

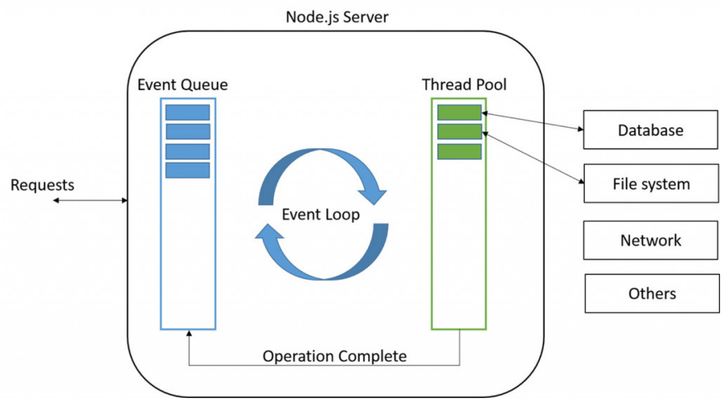
In other words, with Node you do not have to wait and can go on with next tasks. This is one of the reasons it is so efficient. Now, what is a Node JS file:

- It contains tasks and executes them upon set events

- An event is when someone tries to access the server
- A file should be initiated on the server
- Files have ‘.js’ extension

And last but not least, what you can do with Node.js?

- Generate dynamic content
- Create, open and read, or delete files on the server
- Gather and modify data in the database



Why use Node.js

Now it's time to play a quiz of why, who, when and what for. So why use Node JS? Probably, its creator and founder **Ryan Dahl** can shed a light. The major advantage, he tells, is that this JavaScript language doesn't block I/O – meaning input/output communication method. Here, though, the developer community has two views. Some argue, that applications with many CPU cycles can crash then. Others say it's not a big deal at all, as Node code works in small processes.

Another benefit is **single-threaded event loop**, that is responsible for abstracting I/O from external requests. Speaking plainly, this means that Node initiates the event loop at the start, processes the input, and begins the order of operations. Dev geeks interested in exploring it can read Node.js event loop.

Jim Hirschauer of Crafter Software made **few realizations on why use Node JS**. We think they highlight the essence and what is Node.js good for:

1. Google JavaScript engine. Translation: fast and scalable web apps in a result.
2. For server-side applications. Meaning, Node is an event-driven model of programming, where the flow is determined by certain events (user actions, messages, etc.).
3. Easier and scalable. That is, to make apps like Uber or Trello and scaling out on multi-CPU servers.
4. Per-process and across servers. Translation: Node can scale on individual process basis spreading out the load across multi-core servers.

This all seems a bit tough, we realize. So behold a summary of coherent benefits to using Node.js.

10 main reasons to use Node.js

Good for beginner developers, JavaScript is simple to learn, rich framework (Angular, Node, Backbone, Ember)

It is fast, due to Google innovative technologies and the event loop

Ability to keep data in native JSON (object notation) format in your database

Multiple modules (NPM, Grunt, etc.) and supportive community

Good to create real-time apps, such as chats and games

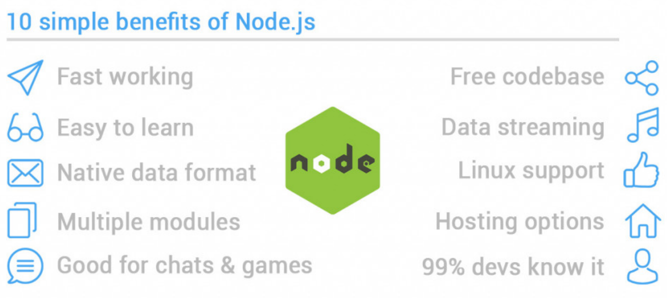
Single free codebase

Good for data streaming, thus for audio and video files, as example

Sponsored by Linux Foundation, as well as PayPal, Joylent, Microsoft, Walmart

Wide range of hosting options

JS is the longest running language, 99% of developers know some of it



Well, this should clear the picture for you a bit more. But wait, did you know NASA also uses Node.js?

Who works with it: Node cases

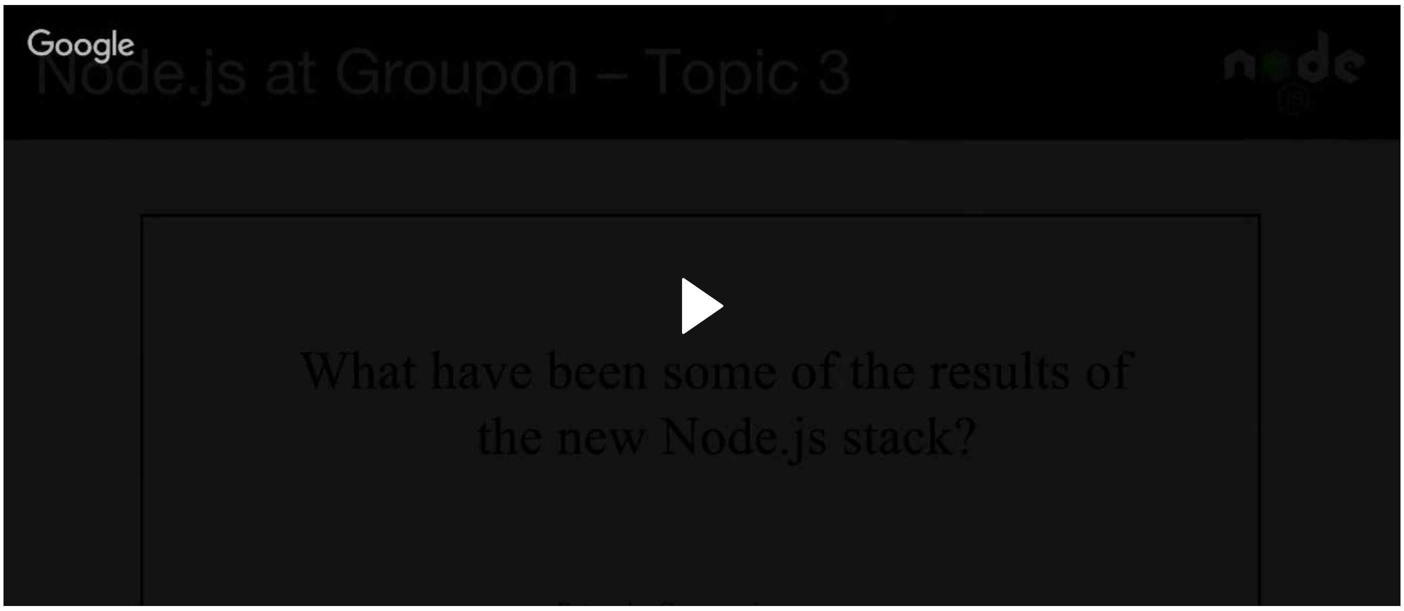
Showing incredible pace (close to 100% growth in use every year), Node JS has become a universal platform for web apps. Companies like PayPal, Walmart use Node for enterprise applications too. Trends building up within Node community are micro-services, real-time applications, and Internet of Things (IoT). But more on that later.

With almost **4 million users** by early 2017, Node JS surely does not lack top-level companies that work with it. For instance, what was our previous reference to NASA all about? Well, this is the truth. The agency in partnership with UTC Aerospace Systems has designed end-to-end system for live data processing. It is used in astronauts' spacesuits and has been build with Node.js.

If you've read our previous post about top companies using Node, you already know about Netflix, Microsoft, Uber and more. Though that is far from all great examples. Capital One, a huge financial corporation, runs numerous projects on because of **short NodeJS development cycles**. Advertising agencies, like Fusion Marketing, create interactive customer experiences. Walmart in retail, Uber in transportation, Google, Twitter, GoDaddy, Skycatch… it may take hours to cover them all.

Manufacturing
BMW
GE
General Motors
Johnson Controls
SIEMENS
Financial
ADP
citigroup
Goldman Sachs
PayPal
WELLS FARGO
eCommerce
amazon.com
BEST BUY
ebay
TARGET
Zappos
Media
beatsMUSIC
Condé Nast
DOWJONES
The New York Times
SONY
Technology
salesforce.com
box
intel
YAHOO!

There's also a series of Node Enterprise conversations, where each episode is devoted to a separate Node.js use case. Like this one, where Adam Geitgey, director of software engineering at Groupon, talks about how the platform helped them to expand.



Node.js at Groupon – Topic 3

What have been some of the results of the new Node.js stack?

One of the results of Node JS use at Groupon was **50% reduction of page load times**. How about that! Let's talk a bit more about Node.js success cases.

Node JS success stories: Groupon, Skycatch & Lowe's

Groupon rebuilt its own website with Node.js moving from Ruby on Rails. Despite Ruby was ok, over time it became harder to maintain the website with each new update. Node.js was chosen for a bunch of reasons: it supported scaling projects, provided better performance and tolerated old Ruby code. In the end of the process, team even released a few own-made JavaScript libraries: Gofer and Node cached. As a bottom line, Groupon is using Node in many ways now:

- for back-end services
- for API's integration layer
- for client's applications and websites

for about 70 own Groupon’s apps

Skycatch is a data company working with commercial drones data. And while creating raw SQL is hard and long, Skycatch allows to do it in an easier way and simplify data extraction from websites. **Andre Deutmeyer** and the team of 20 developers had the task to architect and deliver data to customers quickly. They chose Node.js and basked in wins, like:

Scalable options are better because a hurdle between front-end and back-end is gone;

Node.js back-end services are tolerated by front-end language on servers;

As AWS Lambda is using Node.js too, it allowed focusing on developing apps, not on infrastructure.

Rick Adam, a team leader of 25 developers at **Lowe’s**, had a task to manage apps at presentations tiers. In order to rebuild a monolithic app back in 2007 company chose Node.js as in those days even minor changes in an app text required the whole app to be patched, while Node provided flexibility in that regard. This choice resulted in the following:

positive brokering of web and API’s requests (plus large growth potential)

Node’s asynchronous model gave an opportunity to advance app functionality and better UX

excellent performance

Some of the front-end skills also used in back-end programming

What is Node.js used for

As with Node.js one can use JavaScript on the server, this means one can write JS outside the browser. Additionally, Node.js has the same strength as JavaScript. And it is based on events. These are the 3 whales Node firmly stands on. We can build fast real-time apps like a chat, or an upload system, or any app that has to respond to big number of requests. And we already knew that by now, right?)

So what can be Node JS used for actually? When to use Node and what is it good for? Well, here are few uses that you can name to clients, and examples of what could Node.js be used for.

Streaming data
E.g., file uploading in real time, file encoding while uploading, building proxies between data layers.

Single page apps
Modern web applications, heavy on processing on the client side. Positive response times and sharing data between the server and the client make a good fit for such apps.

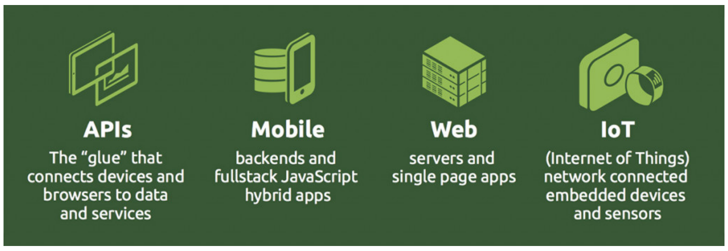
Web applications
Classic web apps on the server side, using Node JS to carry HTML. One of the main benefits in this regard is more SEO-friendly content.

Chats / RTAs
Lightweight real-time applications, like messaging apps interfaces, Twitter, chat software. A classic chat would be a great example of Node use. Simple, intensive on data and across multiple devices.

APIs
REST / JSON programming interfaces and exposing databases or web services through it. No worries about conversion between systems.

Proxy
To deploy Node as proxy to handle connections in non-blocking way. Great for app working with external services, exporting and importing lots of data.

Dashboards
Web application or system monitoring dashboards, enabling tracking user actions. Node also can visualize such interactions for you in real-time.


APIs
The "glue" that connects devices and browsers to data and services
Mobile
backends and fullstack JavaScript hybrid apps
Web
servers and single page apps
IoT
(Internet of Things) network connected embedded devices and sensors

5 less known Node uses

Yet, Node.js is evolving fast and not only web application building is possible. Check out these alternative Node projects that are still in making.

1. NodeOS: an operating system built on top of Linux, with JavaScript as primary runtime and NPM as packaging manager.
2. Node-Webkit: a runtime environment for Node applications. Simple app packaging process – zip it, add information and deploy cross-platform.
3. Log.io: a log monitoring tool, using Socket.io library. All changes you are following, you can track in real-time and in browser.
4. Nodecast: an application that sends images and videos from your mobile phone to PC. Inspired by Google Chromecast.
5. Nexe: a utility for Node app distribution by creating a single executable. Though it only works on Linux and MacOS X so far.

And that is not all, dear stoic readers who are still reading this. Enterprise-scale businesses and projects are also embracing Node.js.

Node for enterprise

We have already mentioned Walmart, Paypal and Netflix. **Fast delivery and iterations** are what Node.js enables and what makes it stand out. Top developers, loving all things new, and who can do anything in JavaScript, like the high performance of Node. One famous example was the choice of Bill Scott, who is now VP at PayPal, when faced with a career choice.

So why enterprises alike go with Node? **Reduced page load times, ease of maintaining, the number of servers reduction** might hold some answers. In addition, a new architecture type of Node, called **micro-services**, helps in handling numerous changes to enterprise software. Under this approach, you can create applications from smaller pieces, and develop those pieces separately. No harm to overall functioning.

There are also developers who prefer full-stack unified solutions. In practice, top 4 technologies used along with Node.js are:

Express

Mongo

jQuery

Angular JS

What are the potential application areas of Node.js?

Media

Payment gateways

Ecommerce

Social media

Enterprise web apps

Backend/API for mobile apps

Basically, any business using Node can: employ less developers, use less servers, decrease page load times. For more thoughts on this, you may watch the next video, where CTO and architect manager of Nodesource talk about Node.

Node in production, dos & don’ts

One final thing in our trip to Nodeland are some practical advises about running this toolset. We should start with process managers to deploy applications. To make your life easier, use NPM, PM2, Adios, Strongloop or any other Node production manager.

Don’t do apps heavy on CPU with Node. Programming things like artificial intelligence (AI), video encoding software, and such software that loads the processor, better use another solution. Node.js has a 1.5 Gb memory limit, though you can apply clustering to fork processes into smaller ones.

Node servers are not great for computational and data-intensive tasks. Thus, it is better to split such tasks into **micro-services** and deploy separately.

Don’t run a Node app **through port 80**. Use a reverse proxy in front of the app, like Nginx for example. In this way you protect servers from internet traffic and spread the load balance.

Install SSL for security reasons. Always use a reverse proxy, check vulnerabilities in SSL and fix the possible issues. Do basic security checks from time to time. Do not use outdated versions of Node and Express.

Think of infrastructure and architecture ahead of app deployment. Experts recommend developing an application **within a private network (VPN)**, so you can allow only trusted connections.

Conclusion

You probably ask a rhetorical question: why on Earth have we poured our souls out about Node? The answer is simple – we love both Node.js and JavaScript.

So, now you’re ready to start your own project. The last step is to hire NodeJS developers right now.