
Node.js at a Glance

Technology, Market and Ecosystem

Joseph Delaney, Colin Gu

Whale Path, Inc. - P.O.Box 390834, Mountain View, CA 94309

Email: info@whalepath.com

Executive Summary

For over 2 decades, the web has been based on the stateless request-response paradigm of HTTP. Since 2005, the mass adoption of AJAX has made the web more dynamic, however communications is still steered by clients. The advent of social media, interactive gaming and collaboration has driven the web towards a new direction, providing real-time and two-way communications. Node.js is uniquely capable of handling this shift in paradigm.

Backed by Joyent and other leading technology organizations, Node.js is an up and coming software platform for the next generation of web development. It is a tool that will truly revolutionize the playing field for entrepreneurs and developers around the world; the software platform builds highly scalable, server-side applications with unprecedented levels of efficiency. Non-blocking I/O and a single-threaded event loop in combination with a built-in HTTP server library allow developers to achieve high throughput without external software (i.e. Apache), putting true control over the web server back in the hands of the developer.

This report, which includes a compilation and synthesis of research conducted by Whale Path, Inc., serves as a fresh primer on Node.js to the developer community and those that have yet to be familiarized with the platform's architecture, pros and cons, state of the market, primary uses, and resources for further exploration.

Report Highlights

- Node.js architecture is uniquely prepared to handle large-scale, highly interactive web servers
 - Node intelligently pushes information from the server side with discretion
 - Leverages Google's V8 to translated JavaScript into machine language for pushing on the server-side
 - Uses an "Event Loop" to run asynchronous tasks while Node.js runs the server
 - Node-based applications achieve high throughput and real-time responsiveness
 - An ecosystem that is experiencing rapid, opens-source based growth; several frameworks detailed in this report
 - Precedence set; several big-box software companies have already recognized the practicality of Node-based applications
 - Features case study material regarding successfully implemented Node.js platforms and details the specific scenarios where a developer would benefit from using Node.js as opposed to alternative technologies such as Ruby on Rails
-

Node.js SWOT Analysis

STRENGTHS

S

- Low Opportunity Cost, Easy to Learn
- Highly Innovative Open Source
- Backed by Joyent
- Event Loop → High Scalability
- Precedence in Big Name Corporations
- JavaScript Centric
- Highly Efficient, Speedy I/O
- Built in HTTP

WEAKNESSES

W

- Ecosystem in Development
- Dynamic API
- Yet to Gain Widespread Industry Exposure
- Few(er) Resources for Beginners
- Unique Characteristics
- CPU Heavy Tasks

OPPORTUNITIES

O

- Untapped market share, development community
- Low barriers to entry
- Word of mouth marketing driven by performance gains
- GitHub Community
- Ecosystem and Framework Development
- Regional Meetups and Workshops

THREATS

T

- Lack of Adoption
- Loss of Corporate Stewardship
- Improper Application → Reputation Damage

Contents

Executive Summary 2

Report Highlights	2
Node.js SWOT Analysis	3
Contents	4

Background 6

Technicals	Error! Bookmark not defined.
Architecture	6
Competitive Positioning	9
Benefits	10
Performance	10
Scalability	11
Weaknesses	11
The Bittersweet Symphony that is JavaScript	11

Market Analysis 12

Usage Statistics and Market Shares	12
Companies, Applications using Node	13
Growth and Trends	14
Node.js Job Trends	14
Node.js Momentum on Google	15
Node.js Meetup Groups Around the Globe	17

Case Study: LinkedIn 19

Role of Node.js	19
Suggestions from the Development Team	19

Ecosystem: In Development, But A Lot to Offer	21
Frameworks	21
Express	21
Compound.js (formally Railway.js)	22
Sails.js	22
Meteor	23
Derby	24
Web Hosting	28
Managed Providers (Partial List)	28
Self-Managed Preconfigured	28
Self-Managed	29
DIY Platforms	29
Node.js Top Contributors	29
Top 5 Individual Contributors	29
Top 5 Contributing Companies	30
Suggestions for Further Research	31
References	33

Background

Node.js was created by Ryan Dahl in 2009 and sponsored by Joyent, one of the top cloud infrastructure companies. It takes advantage of an efficient and under-utilized technological infrastructure, using an “event-loop” to speed up the web server's operation in an easily understood and easily implemented manner that most developers will find friendly. Indeed, it is something new and will require acquisition of new knowledge and techniques, but with opportunity abound. Node-based systems feature a variety of benefits, for example their high scalability. There are weaknesses with Node as there are with any other system implementation, including a still developing ecosystem, requisite knowledge of OO-programming methods and “events,” as well as a highly dynamic API, however, the performance gains achievable through Node-based systems for the right applications make the few downfalls well worth the challenges.



Technical

Before going into the details of Node.js and trying to figure out whether or not it will be to your benefit to learn the new technology, we should develop an understanding of the technical basis of the platform.

Architecture

JavaScript based, in a similar fashion to Tornado and Twisted for Python or Event Machine, Node.js features event-driven, non-blocking I/O, which keeps itself lightweight and efficient in the face of data-intensive real-time applications that run across distributed devices.

Traditional web-serving techniques require each connection (request) creates a new thread, taking up system RAM and eventually maxing-out at the amount of RAM available. Node.js is different. It operates on a single-thread, using non-blocking I/O calls, allowing it to support tens of thousands of concurrent connections, without worrying about RAM limitations and the cost of context-switching between threads.

Node.js brings a new element to the mix in the form of an event-driven focus on the server-side. Client-side operational methods simply aren't efficient enough for the current and forthcoming generations of web systems. Node.js pushes data to clients if and only when appropriate. The scalability of Node-based systems comes from the fundamental separation of tasks; the application handling client communications lets the event loop take on the time-intensive I/O management and computations. The following diagram visualizes this process thoroughly.

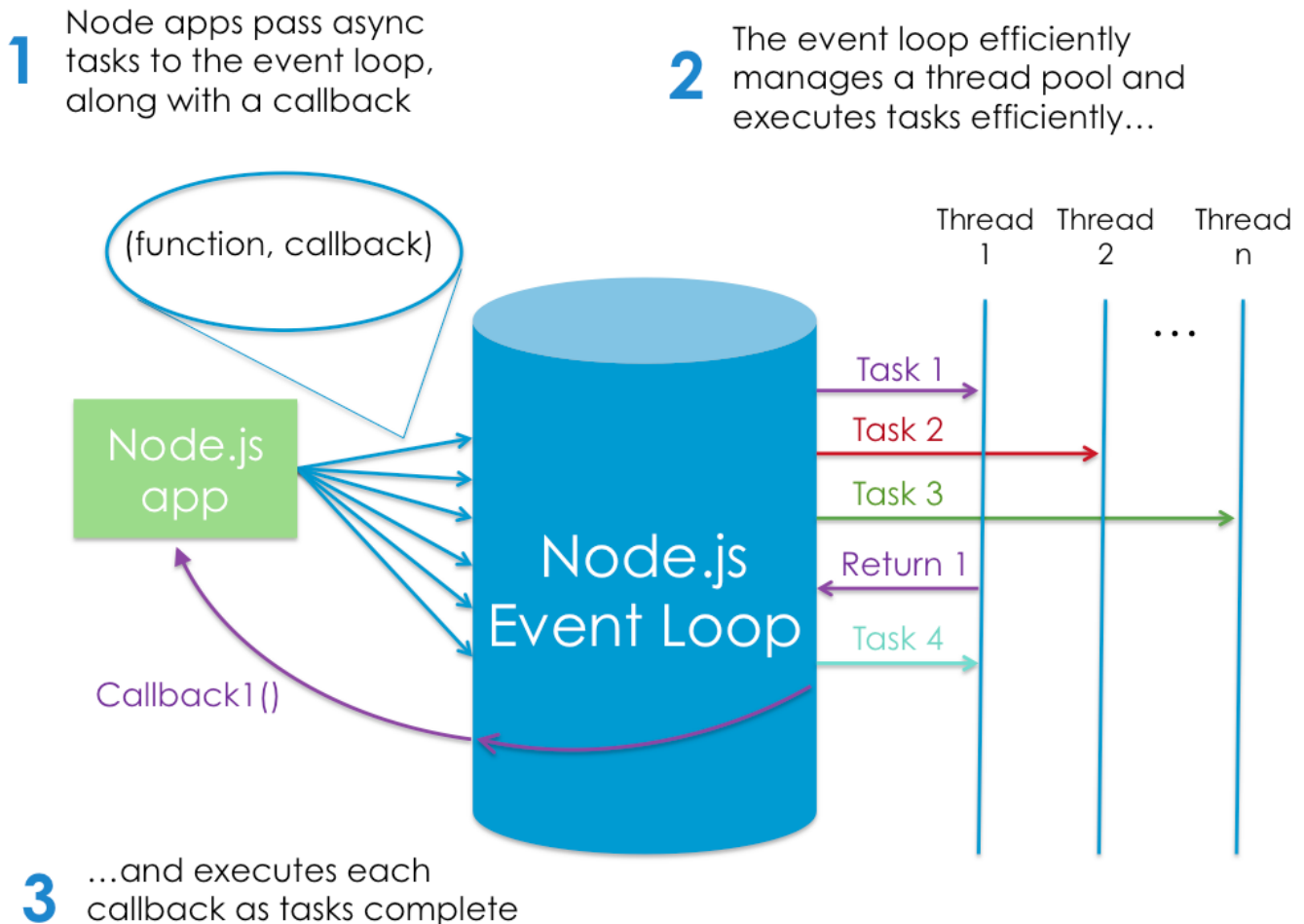


Figure 1 Node.js Architecture

Node's design of sharing a single-thread between all the requests means it's good at building highly concurrent applications. But if you have a computation intensive program, it will choke up the single thread and block all the incoming requests until the computation is completed. A blog article written by Tomislav Capan highlights types of applications that Node.js should and should not be used:

APPLICATIONS WHERE NODE.JS SHOULD BE USED

CHAT	"The most typical real-time, multi-user application, a sweet-spot for Node.js: it's a lightweight, high traffic, data-intensive (but low processing/computation) application that runs across distributed devices."
DATA STREAMING	"In more traditional web platforms, HTTP requests and responses are treated like isolated event; in fact, they're actually streams. This observation can be utilized in Node.js to build some cool features. For example, it's possible to process files while they're still being uploaded, as the data comes in through a stream and we can process it in an online fashion."
PROXY	"Node.js is easily employed as a server-side proxy where it can handle a large amount of simultaneous connections in a non-blocking manner. It's especially useful for proxying different services with different response times, or collecting data from multiple source points. An example: consider a server-side application communicating with third-party resources, pulling in data from different sources, or storing assets like images and videos to third-party cloud services."
APPLICATION MONITORING DASHBOARD	"Tracking website visitors and visualizing their interactions in real-time....You could be gathering real-time stats from your user, or even moving it to the next level by introducing targeted interactions with your visitors by opening a communication channel when they reach a specific point in your funnel."

APPLICATIONS WHERE NODE.JS CAN BE USED

SERVER-SIDE WEB APP	<p>Pros:</p> <ul style="list-style-type: none"> - If your application doesn't have any CPU intensive computation, you can build it in Javascript top-to-bottom, even down to the database level if you use JSON storage Object DB like MongoDB. This eases development (including hiring) significantly. - Crawlers receive a fully-rendered HTML response, which is far more SEO-friendly than, say, a Single Page Application or a websockets app run on top of Node.js. <p>Cons:</p> <ul style="list-style-type: none"> - Any CPU intensive computation will block Node.js responsiveness, so a threaded platform is a better approach. Alternatively, you could try scaling out the computation [*]. - Using Node.js with a relational database is still quite a pain (see below for more detail). Do yourself a favor and pick up any other environment like Rails, Django, or ASP.Net MVC if you're trying to perform relational operations."
---------------------	---

APPLICATIONS WHERE NODE.JS SHOULD NOT BE USED

SERVER-SIDE WEB APP WITH RELATIONAL DB	"Relational DB tools for Node.js are still in their early stages; they're rather immature and not as pleasant to work with. On the other hand, Rails automatically provides data access setup right out of the box together with DB schema migrations support tools and other Gems (pun intended). Rails and its peer frameworks have mature and proven Active Record or Data Mapper data access layer implementations, which you'll sorely miss if you try to replicate them in pure JavaScript"
HEAVY SERVER-SIDE COMPUTATION OR	"In general, any CPU intensive operation annuls all the throughput benefits Node offers with its event-driven, non-blocking I/O model because any incoming requests will be blocked while the

PROCESSING

thread is occupied with your number-crunching... Node.js is single-threaded and uses only a single CPU core. When it comes to adding concurrency on a multi-core server, there is some work being done by the Node core team in the form of a cluster module. With clustering, you should still offload all heavy computation to background processes written in a more appropriate environment for that, and having them communicate via a message queue server like RabbitMQ. Even though your background processing might be run on the same server initially, such an approach has the potential for very high scalability. Those background processing services could be easily distributed out to separate worker servers without the need to configure the loads of front-facing web servers.”

Competitive Positioning

w3tech.com compares all the most popular web servers in the market and shows the market position of Node.js by popularity and website traffics.

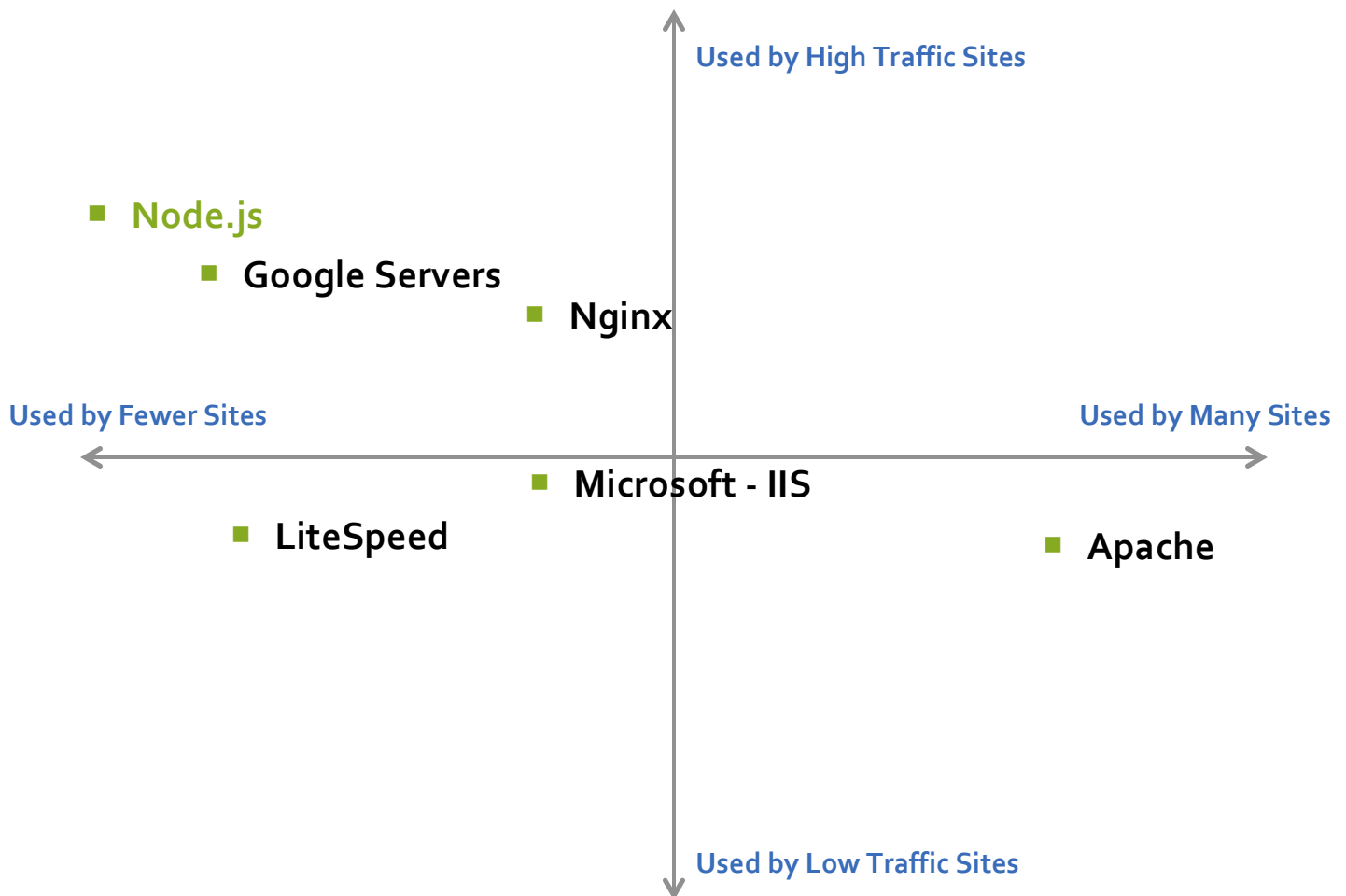


Figure 2 Node.js Market Positioning

Benefits

The benefits of Node really are numerous and they make it well worth learning the platform. First of all, it is relatively easier to learn than many of the other server-side platforms currently in use. JavaScript's developer base and familiarity with the development community at large is a benefit to Node by proxy, as the platform is written and built to run in a JS environment. This, in combination with scalability and pure speed is where Node.js truly excels.

Performance

Performance is always one of the excitements over Node. A series of comparative benchmarks of similar applications show that Node.js is having tremendous performance gains. We need to caution however against using pure benchmarks to base one's decisions around. Clearly, for different unique purposes different languages have better performance. For example, Node-based systems wouldn't perform as well in environments that are more CPU-heavy and less I/O based. Video encoding and AI are examples of such a scenario.

Node.js vs. Java EE

Marc Fasel, Senior Consultant at Shine Technologies in Australia did a performance comparison between Node.js and Java EE for reason JSON data from CouchDB and came to the ultimate conclusion that Node is roughly 20% faster.

Data source: <http://java.dzone.com/articles/performance-comparison-between>

Node.js vs. Nginx

Ryan Dahl (Node's original author) showed a simple benchmark which returned a 1 megabyte binary buffer; Node gave 822 req/sec, while nginx gave 708 req/sec. He also noted that nginx peaked at 4 megabytes memory, while Node peaked at 64 megabytes.

Data source: http://nodejs.org/cinco_de_node.pdf

Node.js vs. PHP

An open-source project hosted on Google does concurrency benchmarks between Node.js and PHP. They tested the scenarios with different amount of requests (ranging from 10,000 to 100,000) and concurrent requests (ranging from 10 to 1,000), and the result shows Node.js is about 100% faster than Apache PHP in all these cases.

Data source: <https://code.google.com/p/node-js-vs-apache-php-benchmark/wiki/Tests>

Scalability

Scalability is key in a situation where you're developing a very consumer oriented system. Node can handle massively distributed environments, with no hesitation for higher server quantities. In terms of pure speed, Node has the potential to improve load times by high marginal percentages; the platform simply offers a better, more fluid web experience. Part of the speed of Node is derived from the speed of V8, an advantage that Node has over many others in the arena.

Weaknesses

Part of the beauty of Node.js is simply that there aren't too many weaknesses in the typical sense of the word. Most if not all of the ailments currently afflicting the platform are simply part of the nature of where Node.js is at in the development life-cycle. Node's founder, Ryan Dahl, has already hinted at future further development of API in a web-worker manner as well as functional modularity and shared objects. In the future, libraries for common databases will ideally be featured in distributions. Perhaps the largest problem with the platform at its current state in time is the lesser developed ecosystem, but that too of course will only appreciate with time. None of the frameworks available for Node currently can match Rails or Django.

The Bittersweet Symphony that is JavaScript

As previously mentioned, there are numerous benefits to Node-based reliance on JavaScript. It is largely beneficial to have the same language for use on the server-side and the client-side. While this long sought dream fell by the wayside over the years, Node brings back the possibility. This would improve costs, as the same staff could work on both ends of the business, and code would be easily migrated. Common data formats and software/testing tools would be a useful feature as well. Furthermore, the familiarity of JS with the developer community, as already pointed out, makes Node a very friendly platform.

However, JS presents several moot points. [5] For one, the Global Object. With all top-level variables tossed together, modularity can be more difficult to navigate. Luckily, Node.js makes use of the CommonJS module system, which lessens the hassle by making local objects truly local.

Market Analysis

Since its initial release on March 27, 2009, Node.js has gained tremendous momentum around the globe. It's the second most popular project on GitHub, and within 3 months after its version 0.10 release, there have been an average of 35,000 downloads per day and over 1 million of downloads from nodejs.org alone.

Let's take a closer look at how popular Node.js is.

Usage Statistics and Market Shares

To start with, the percentage of websites that run Node.js has jumped roughly 5 times to 0.05% in the last year, according to the statistic data from w3techs.com. More and more startup companies and even big brands have started leveraging Node to power their businesses and technologies.

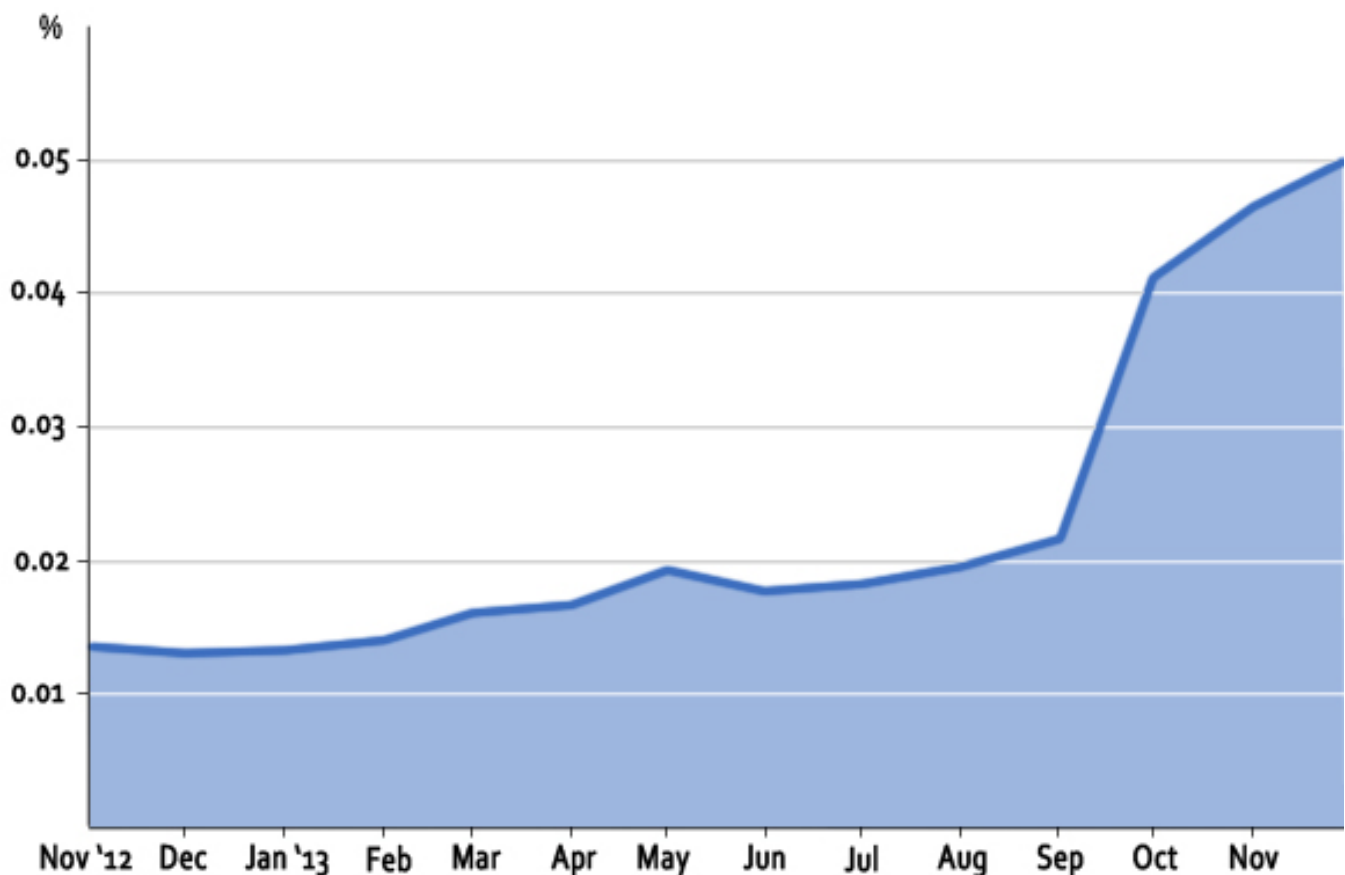









Figure 3 Usage of Node.js for Websites within a year

Companies, Applications using Node

NAME	USAGE
	Use node.js as the server interface for mobile applications
	WSJ Mobile, Portfolio, Streaming Stories, DJ X, and many other tools and services at Dow Jones run completely on node
	The Yammer Application Platform heavily uses node.js
	Created the KrakenJS framework, which is now used in the development of all new web applications.
	Re-architect the entire front-end to small pieces and rebuilt each major section of the website as an independent Node.js application.
	Node.js is the basis of Yahoo! Manhattan, part of the “Cocktails” platform
	Mobile website.

Complete List of Frameworks is available at <https://github.com/joyent/node/wiki/Projects,-Applications,-and-Companies-Using-Node>

Growth and Trends

Node.js Job Trends

The need for Node.js developers is exploding. According to the job trend report from Indeed.com, Node.js jobs have grown over 40,000% in less than 2 years, compare to Ruby on Rails' 4,500% and PHP's 200% during the same period of time.

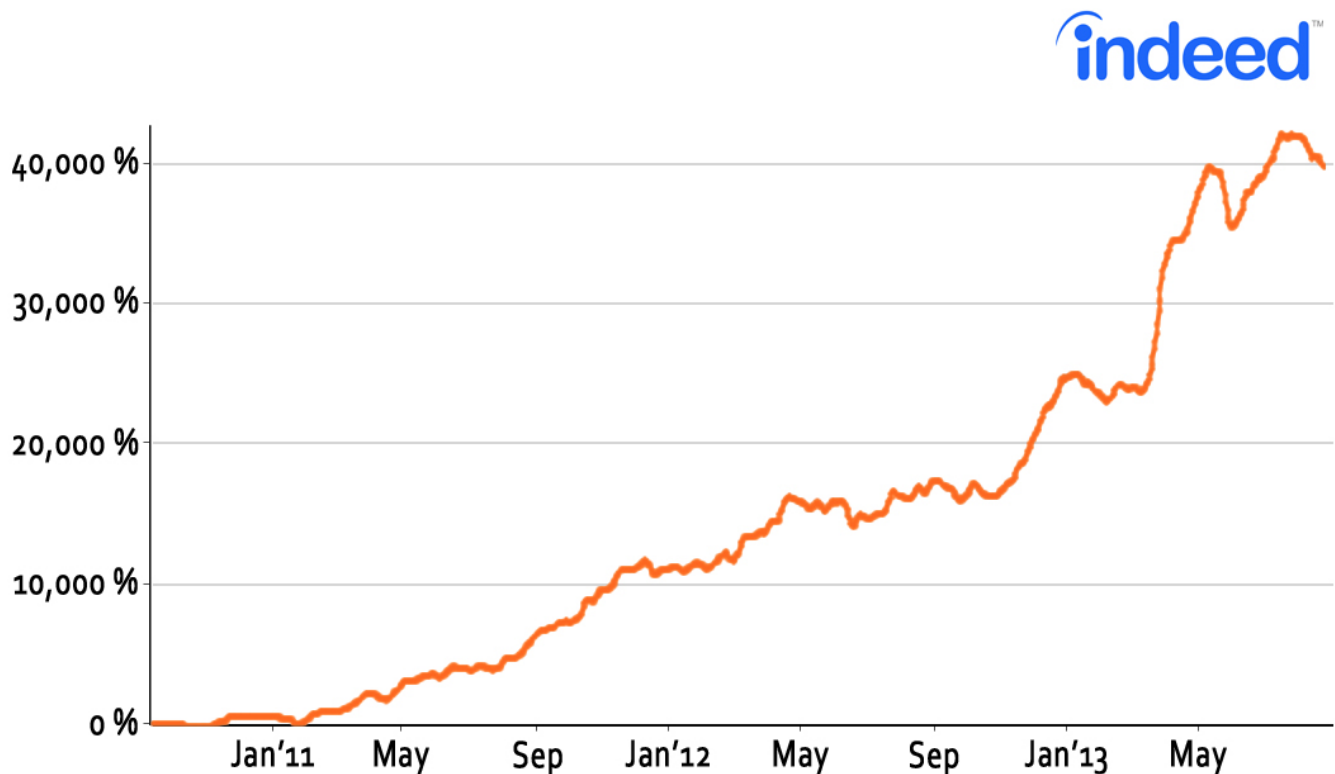


Figure 4 Node.js Job Trend from Indeed.com

Not only is Node.js the fastest growing job opportunity, but it's also the highest paying one. Indeed.com data shows that the average salary for Node.js jobs in all the top high-tech metros in the US beats the salary for similar jobs in Ruby on Rails, Java and PHP.

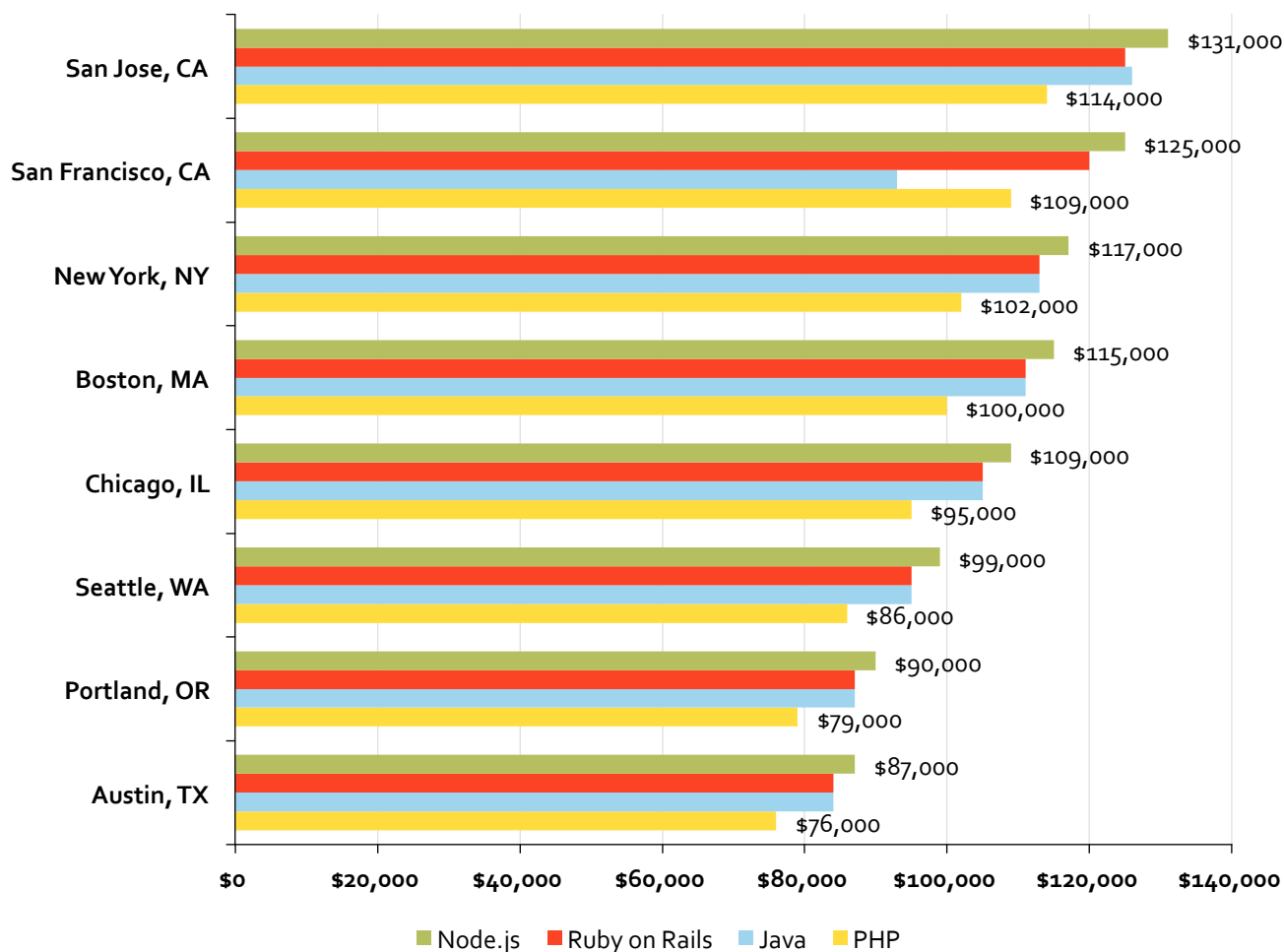


Figure 5 Average Salaries for Programming Jobs in US Major High-tech Metros

Node.js Momentum on Google

While leading companies and websites have embraced Node.js and its job market is heating up, more and more people started to pay attention to this new technology. An analysis on Google Trend demonstrates the growing interest of Node.js over the web. Surprisingly, people who are most interested in Node.js come from South Korea, while Americans only ranked at the 7th place, after Hong Kong, Russia, Taiwan and other regions. This reflects the broad opportunities in the North American region for the platform.

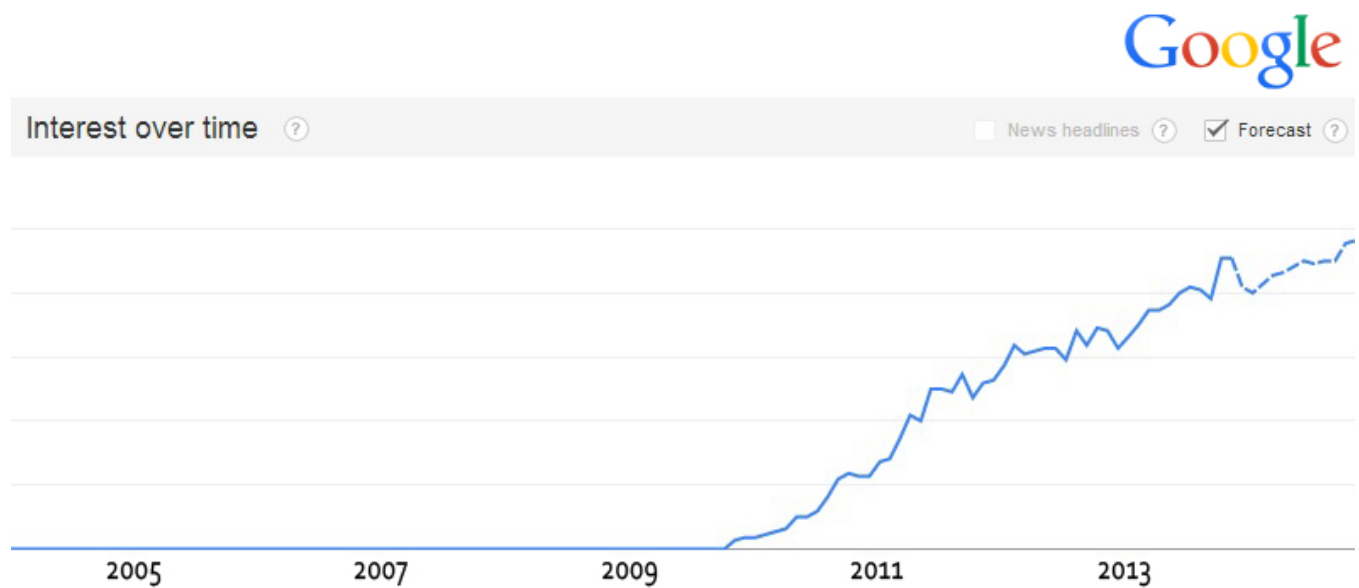


Figure 6 Node.js Interest on Google

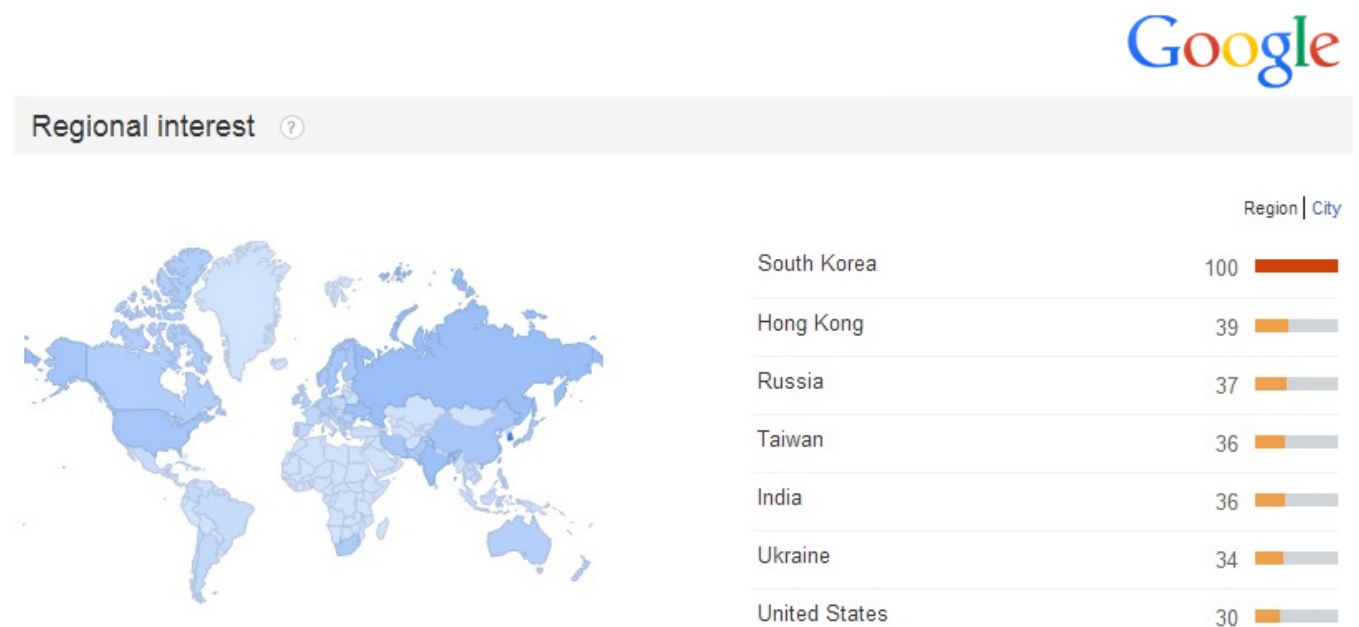
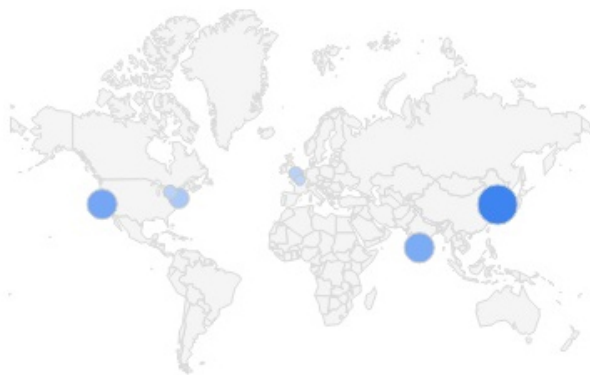


Figure 7 Node.js Interest on Google – by Regions



Regional interest ?

Region | City



Seoul	100	<div><div></div></div>
San Francisco	63	<div><div></div></div>
Bangalore	58	<div><div></div></div>
New York	26	<div><div></div></div>
Toronto	16	<div><div></div></div>
London	16	<div><div></div></div>
Paris	12	<div><div></div></div>

Figure 8 Node.js Interest on Google - by Cities

Node.js Meetup Groups Around the Globe

Local communities and Meetup Groups for Node are getting hot as well. As shown in the following map, over 200 Meetup groups with more than 50,000 members around the globe talk about Node.js.

**Groups**
216**Members**
50,394**Interested**
7,382**Cities**
144**Countries**
44

Top 5 Node.js Meetup Groups

MEETUP GROUP NAME	LOCATION	# OF MEMBERS
THE SF JAVASCRIPT MEETUP	San Francisco, CA, USA	5,770
NODE.JS CLUB SF	San Francisco, CA, USA	2,158
NODEJS	New York, NY, USA	1,710
JAVASCRIPTMN	Minneapolis, MN, USA	927
CHICAGO NODE.JS	Chicago, IL, USA	887

Case Study: LinkedIn

LinkedIn is the world's largest social networking website primarily targeted at users in professional occupations, mainly utilized for professional networking. As of this past June, LinkedIn had acquired over 250 million users worldwide in over 200 countries. With over 90% of users opting not to pay for the LinkedIn premium service, LinkedIn makes its revenue through a combination of premium subscriptions, marketing solutions and talent solutions.

Role of Node.js

According to the company's CEO, LinkedIn this year experienced 38% of its web traffic through its mobile application. Next year, it expects this number to climb to over 50%. LinkedIn had previously built its mobile app around Ruby on Rails, however for performance and scalability reasons, the company decided to replace its entire back-end mobile infrastructure with Node.js. The company was experiencing scalability issues at a time when less than 10% of their traffic was generated from mobile and they needed a solution.

According to Kiran Prasad, Director of Mobile Engineering, the company considered 3 separate solutions: Rails coupled with Event Machine, Python with Twisted, and Node.js. The company chose Node and Prasad detailed several key points in conversations with tech hub ArsTechnica and VentureBeat. Node.js was up to 20x faster in certain scenarios than Rails, the company decreased its server quantity by 90%, enabling more room for traffic growth, and front-end JS engineering teams were placed on back-end coding projects because of the language familiarity. Development was extremely fast as well, showcasing the learning curve of the platform.

While V8 was exceedingly fast, a former team member cautioned against rushing into projects with Node without considering its suitability for the individual project's particular purposes. Every project has different needs, and in many situations, crunching numbers and other heavy processing tasks for example, Node simply isn't practical.

Suggestions from the Development Team

(From the LinkedIn Engineering website)

- Avoid synchronous code
 - Turn off socket pooling
 - Don't use it for static assets
 - Render on the client-side
-

- Take advantage of Gzip
- Perform all blocking operations in parallel
- Manage the request/response cycle session-free, LinkedIn uses the Express framework, but removes the relevant configuration code
- Use binary modules instead of JavaScript modules
- Use V8 JS instead of client-side libraries
- Keep code small and light

Ecosystem: In Development, But A Lot to Offer

Frameworks

Sinatra-like

Express

A light-weight web application framework to help organize your web app into MVC architecture on the server side. You can use a variety of choices for your templating language (like ejs, jade, dustjs).

express
web application
framework for
node

Quick Facts

CURRENT VERSION	3.4.6
WEBSITE	http://expressjs.com
AUTHOR(S)	<ul style="list-style-type: none">■ TJ Holowaychuk (visionmedia)■ Ciaran Jessup (ciaranj)■ Aaron Heckmann (aheckmann)■ Guillermo Rauch (guille)
FEATURES	<ul style="list-style-type: none">■ Built on Connect, an extensible HTTP server framework for Node.■ Robust routing■ HTTP helpers (redirection, caching, etc.)■ View system supporting 14+ template engines■ Content negotiation■ Focus on high performance■ Environment based configuration■ Executable for generating applications quickly■ High test coverage
COMPANIES/WEBSITES THAT USES IT (PARTIAL)	<ul style="list-style-type: none">■ MySpace■ Storify■ Geekli.st■ Clipboard■ Ghost

OTHER FRAMEWORKS THAT ARE BUILT WITH IT (PARTIAL)	■ CompoundJS -- High-level MVC framework inspired by RoR
	■ Derby -- A real-time, collaborative application framework
	■ Tower.js -- Full Stack Web Framework for Node.js and the Browser. Modeled after Ruby on Rails. Built for the client and server from the ground up.
	■ Locomotive -- a framework that brings structure and MVC patterns to web applications

Rails-like

Compound.js (formally Railway.js)

Compound's formula is Express + structure + extensions. Where structure is the standard layout of directories, and extensions are node modules adding functionality to the framework. Compound's goal is to provide an obvious and well-organized interface for express compatible application development. This means that everything that works with express will work with compound.



Quick Facts

CURRENT VERSION	1.1.6
WEBSITE	http://compoundjs.com
AUTHOR(S)	■ Anatoliy Chakkaev ■ Daniel Lochrie

Sails.js

Sails.js make it easy to build custom, enterprise-grade Node.js apps. It is designed to mimic the MVC pattern of frameworks like Ruby on Rails, but with support for the requirements of modern apps: data-driven APIs with scalable, service-oriented architecture. It's especially good for building chat, real-time dashboards, or multiplayer games.



Quick Facts

CURRENT VERSION	0.9.7
WEBSITE	http://sailsjs.org
AUTHOR(S)	Mike McNeil

SUPPORTED BY	Balderdash
FEATURES	<ul style="list-style-type: none">■ Pure JavaScript. All the same APIs are available on the client and the server .■ Live page updates. Templates get updated automatically when data in the database changes. No more boilerplate redraw code to write. Supports any templating language.■ Clean, powerful data synchronization. Write your client code as if it were running on the server and had direct access to the database. No more loading your data from REST endpoints.■ Latency compensation. Users' screens are updated automatically when anyone else makes a change. The client is patched up with what actually happened If the server rejects the change request or executes it differently.■ Hot Code Pushes. Update your app while users are connected without disturbing them. When you push a new version, the new code is seamlessly injected into each browser frame in which the app is open.

Full-stack Frameworks

Meteor

Meteor is a next generation real-time framework that is built on top of Node.js. It speeds up the webs by transforming them into single-page applications. Its reactive philosophy ensures that any change in the data is automatically reflected everywhere throughout the application.

The Meteor logo, featuring the word "Meteor" in a bold, black, sans-serif font.

Quick Facts

CURRENT VERSION	Preview 0.6.6.3
WEBSITE	http://www.meteor.com
AUTHOR(S)	<ul style="list-style-type: none">■ Geoff Schmidt■ Matt DeBergalis■ Nick Martin■ David Greenspan
FUNDING	\$11.2 Million
INVESTOR	<ul style="list-style-type: none">■ Andreessen Horowitz■ Matrix Ventures■ James Lindenbaum (Heroku)■ Dustin Moskovitz (Facebook, Asana)■ Alexis Ohanian (Reddit)■ Paul Buchheit (Gmail)■ Rod Johnson (Spring)■ Peter Levine (XenSource)

- David Skok (JBoss, SilverStream)
- Maynard Webb
- Ron Conway
- Yuri Milner
- Y Combinator

FEATURES

- Database agnostic. Its ORM, Waterline, provides a simple data access layer that works, no matter what database you're using
- Sails.js automatically generates a RESTful JSON API for your app
- Realtime Socket.io requests are routed to your controllers the same way as everything else: with resourceful conventions and URL mappings.
- Sails.js provides basic security and role-based access control by default, and you can add as many custom policies as you like.
- Sails.js has automatic asset minification. you just put your files in the proper folder and they are automatically included in your layout.
- Sensitive code runs in a privileged environment. Write all of your code in JavaScript (if you want.) The user interface runs in your browser. The sensitive functions run in a privileged server environment.
- Fully self-contained application bundles. One command to compile your entire application into a tarball. Unpack it anywhere there's node.js, run one command, and you're on the air.
- Interoperability. You can connect anything to Meteor, from native mobile apps to legacy databases to Arduinos.
- Smart Packages. Small programs that can inject code into the client or the server, or even hook into the bundler to preprocess your source. Great care has been taken to give the core Meteor packages the minimal set of dependencies, so you can use your favorite templating, testing, or DOM manipulation frameworks.

Derby

Derby is a full-stack MVC framework making it easy to write real-time, collaborative applications for Node.js and browsers. It provides shared server and client rendering, automatic data synchronization, and real-time conflict resolution. Derby makes it simple to write applications that load as fast as a search engine, are as interactive as a document editor, and work offline.


 The word "Derby" is written in a large, bold, red sans-serif font.
Quick Facts






CURRENT VERSION	0.5.9
WEBSITE	http://derbyjs.com
AUTHOR(S)	<ul style="list-style-type: none"> ■ Nate Smith ■ Brian Noguchi
FEATURES	<ul style="list-style-type: none"> ■ Realtime collaboration: Powered by ShareJS's operational transformation of JSON and text, all conflicting data changes are automatically resolved. By default, every text input is a collaborative text editor, and every bit of data in the model can be collaboratively edited in

real-time or offline.

- Client and server routing: The same routes produce a single-page browser app and an Express server app. Links render instantly with push/pop state changes in modern browsers, while server rendering provides access to search engines and browsers without JavaScript.
- HTML templates: Handlebars-like templates are rendered into HTML on both the server and client. Because they render on the server, pages display immediately—even before any scripts are downloaded. Templates are mostly just HTML, so designers can understand and modify them.
- View bindings: In addition to HTML rendering, templates specify live bindings between the view and model. When model data change, the view updates the properties, text, or HTML necessary to reflect the new data. When users interact with the page—such as editing the value of a text input—the model data update.

Complete List of Frameworks is available at <http://nodeframework.com>

	EXPRESS JS	COMPOUND JS	SAILS JS	METEOR	DERBY
DEVELOPMENT PRINCIPLES	-	<ul style="list-style-type: none"> KISS Don't repeat yourself 	<ul style="list-style-type: none"> Test-driven development Configuration over convention 	-	<ul style="list-style-type: none"> Configuration over convention
DESIGN PATTERN	Model-View-Controller (MVC)	Hierarchical MVC	No	Resource View Presenter	MVC
LICENSE	MIT License	MIT License	MIT License	MIT License	MIT License
TARGET AUDIENCE	-	<ul style="list-style-type: none"> Web Development 	<ul style="list-style-type: none"> Web Development Enterprise 	<ul style="list-style-type: none"> Web Development 	<ul style="list-style-type: none"> Web Development Enterprise
MULTILINGUAL CONTENT	-	Yes	Yes	Yes	Yes
DEPENDENCY	-	<ul style="list-style-type: none"> Express Yii 	<ul style="list-style-type: none"> Express Yii 	Express	<ul style="list-style-type: none"> Express Yii Bootstrap
DATABASE	MongoDB	<ul style="list-style-type: none"> MongoDB Redis Neo4J Community MySQL PostgreSQL SQLite 	<ul style="list-style-type: none"> MongoDB MySQL Redis Flat File 	MongoDB	<ul style="list-style-type: none"> MongoDB Redis
MULTI-USER SYSTEM	Yes	Yes	Yes	Yes	Yes
EXTENTION/PLUG-IN	Yes	Yes	Yes	Yes	Yes
DATABASE MODEL	<ul style="list-style-type: none"> NoSQL Key-value Relational 	<ul style="list-style-type: none"> NoSQL Relational 	<ul style="list-style-type: none"> NoSQL Relational 	<ul style="list-style-type: none"> Document Oriented 	<ul style="list-style-type: none"> Document Oriented
TEMPLATE LANGUAGE	<ul style="list-style-type: none"> Jade EJS Plates Mustache Blade 	<ul style="list-style-type: none"> Jade EJS 	<ul style="list-style-type: none"> Jade EJS 	<ul style="list-style-type: none"> Handlebars JS 	<ul style="list-style-type: none"> Handlebars JS

	EXPRESS JS	COMPOUND JS	SAILS JS	METEOR	DERBY
REST-FUL	Yes	Yes	Yes	Conditional	Yes
SCRIPTING LANGUAGE SUPPORT	<ul style="list-style-type: none"> ■ Javascript ■ CoffeeScript 	<ul style="list-style-type: none"> ■ Javascript ■ CoffeeScript 	<ul style="list-style-type: none"> ■ Javascript ■ CoffeeScript 	<ul style="list-style-type: none"> ■ Javascript ■ CoffeeScript 	<ul style="list-style-type: none"> ■ Javascript ■ CoffeeScript
PERFORMANCE	★★★★	-	★★★★	★★★★	★★★★
WEBSOCKET SUPPORT	Yes	-	Yes	Yes	-
PROGRAMMING PARADIGM	<ul style="list-style-type: none"> ■ Event-driven ■ Object-oriented ■ Imperative programming 	<ul style="list-style-type: none"> ■ Event-driven ■ Object-oriented ■ Functional 	<ul style="list-style-type: none"> ■ Event-driven 	<ul style="list-style-type: none"> ■ Reactive Programming 	<ul style="list-style-type: none"> ■ Event-driven ■ Object-oriented ■ Reactive programming
CODE GENERATION	Yes	Yes	Yes	No	Yes
DOCUMENTATION LEVEL	★★★	★★★★	★★★★★	★★★★	★★★★
EASE OF USE	★★★★★	★★★★	★★★★★	★★★★	★★★★
FREE TO USE	Yes	Yes	Yes	Yes	Yes
DIFFICULTY LEVEL	<ul style="list-style-type: none"> ■ Intermediate ■ Advanced ■ Beginner 	<ul style="list-style-type: none"> ■ Beginner 	<ul style="list-style-type: none"> ■ Beginner 	<ul style="list-style-type: none"> ■ Intermediate ■ Beginner 	<ul style="list-style-type: none"> ■ Intermediate
IMPLEMENTATION FLEXIBILITY	 8% Votes	 33% Votes	 33% Votes	 8% Votes	 33% Votes

Data Source: <http://vschart.com>

Web Hosting

Managed Providers (Partial List)

Provide a simplified "Node Appliance" solution. Node and NPM will already be set up for you, and deploys are typically done via git push or similar method. You will have less control of your server, but everything will be set up for you.

NAME	NODE VERSION	IP/HOSTNAME	WEB SOCKETS	FREE PLAN	PAID PLANS
APFPOG	0.4.12, 0.6.17, 0.8.14	Yes, custom domains in paid plan only	No	Yes – up to 2G RAM for your applications	Yes – monthly subscriptions and enterprise support available
CLOUD FOUNDRY	0.4.12, 0.6.8, 0.8.2	No	No	Yes - Only during beta.	
DOTCLOUD	0.4.10	Yes	Paid Plan		Pro - \$99/month, 4 services. Enterprise - Unlimited services.
HEROKU	0.4.x, 0.6.x, 0.8.x, 0.10.x	Yes	Yes	Yes - 1 Dyno (512 MB Ram)	\$0.05/hour/dyno
NODEJITSU	0.4.12, 0.6.x, 0.8.x, 0.10.x	Yes	Yes	30 days sandbox	Yes
WINDOWS AZURE	0.6+	Yes (Worker Role)	Yes	3 month free trial 10 free web sites forever	Yes

Self-Managed Preconfigured

"Node Appliance" solutions where Node and NPM are already set up for you, and deploys are typically done via git push or similar method. You will have complete control over and responsibility for your servers, but everything will be set up for you.

NAME	NODE VERSION	IP/ HOSTNAME	WEB SOCKETS	FREE PLAN	PAID PLANS
CURE	0.6.2	Yes	Yes	Yes - One week trial. (Up to 1GB outgoing b/w, \$0.18 per GB after.)	\$12.95/month per server.
JOYENT	0.10.21	Yes	Yes	Yes	Plans starting from \$0.008/hour, up to \$2.56/hour, depending on instance size

Self-Managed

VPS providers, which often require you to set everything up yourself, including the operating system.

- [Host Stage](#)
- [A2 Hosting](#)
- [Amazon EC2](#)
- [BuyVM](#)
- [Digital Ocean](#) - \$5/mo. SSD VPS, New York and Amsterdam
- [DreamHostPS](#) - Need a total reconfiguration of virtual machines
- [Joyent](#)
- [MORE...](#)

DIY Platforms

Node.JS hosting platforms that allow you to host Node.JS apps on your own servers or clouds.

- [Nodester](#) - Open source Node.JS hosting platform and services
- [CloudFoundry](#) - Open source PaaS with support for NodeJS
- [OpenShift](#) - Open source polyglot PaaS with native support for Node.js
- [Raft](#) - Open source PaaS built on Node.js Looking for please to help out with the project.




Data Source: <https://github.com/joyent/node/wiki/node-hosting>

Node.js Top Contributors

Top 5 Individual Contributors

NAME	ROLE	COMPANY	LOCATION	# OF COMMITTS	LINES OF CODE ADDED/CHANGED
RYAN DAHL	Node.js Creator	Joyent	San Francisco, CA	2,941	5,218,608
ISAAC SCHLUETER	Author of NPM	Joyent	Oakland, CA	1,474	1,162,482
BEN NOORDHUIS		StrongLoop	Gouda, The Netherlands	1,317	1,265,667
BERT BELDER	Main developer for Node's Windows support	StrongLoop	Amsterdam, The Netherlands	510	450,038
FEDOR INDUTNY		Voxer	Russia	333	182,163

Top 5 Contributing Companies

NAME	PROFILE	WEBSITE	DESCRIPTION
 StrongLoop™	http://www.crunchbase.com/company/strongloop	http://strongloop.com	StrongLoop develops a leading Mobile API Tier called StrongLoop Suite. It's also the primary code contributor to Node.js.
 Joyent	http://www.crunchbase.com/company/joyent	http://www.joyent.com	Cloud infrastructure company offering solutions for building real-time web and mobile applications.
 voxer	http://www.crunchbase.com/company/voxer-llc	http://www.voxer.com	Provide real-time multimedia communication solutions to consumers and businesses.
 Microsoft	http://www.crunchbase.com/company/microsoft	http://www.microsoft.com	World's largest software company.
mozilla	http://www.crunchbase.com/company/mozilla	http://www.mozilla.org	A non-profit organization that develops and supports open source Mozilla products, such as Firefox browser.

Suggestions for Further Research

Node.js was originally created by Ryan Dahl, however the Node.js community is rapidly expanding at an exponential level. As time goes on, more and more resources will be readily available for aspiring developers and veterans alike. The following sources offer comprehensive information on the Node.js platform beyond the extent of this white paper.

<http://nodejs.org/>

The home site for the Node.js platform, this .org features a detailed, guided introduction to the Node.js platform, with documentation, API specifications, a blog, and videos from founder Dahl detailing the platform for newcomers. It features further resources as well for discussion, documentation and periodicals amongst the Node community.

<http://www.joyent.com/technology/nodejs>

Joyent is the self-proclaimed “corporate steward” of Node.js. The company offers debugging and performance analysis tools for Node-based applications, and the company's flagship software, “Joyent Cloud” will be a much-appreciated tool for future Node-based developers.

<https://www.codeschool.com/courses/real-time-web-with-nodejs>

Code School has a detailed, easily accessed introductory course for Node.js newcomers. The course, titled “Real Time Web, With Node.js,” serves as a terrific tutorial of the platform. Level 1 of 7 serves as a free preview, with an option to continue in more depth at affordable rates.

As previously mentioned, there are a variety of frameworks available. Each of the frameworks featured in this research paper has resources that can be easily retrieved from the homepages listed.

<http://www.nodebeginner.org/>

For those on a tight budget, the Node Beginner Book is a terrific resource developed by Manual Kiessling, a software architect and head of IT at MeinAuto.de, with a full 200+ pg. guide available at the affordable price of \$9.99

<http://visionmedia.github.io/masteringnode/>

Lastly, Mastering Node is another terrific, affordable option to learn the platform. Available as an open source ebook in pdf and other formats, Mastering Node is one of the most complete guides for introducing Node to the development community.

References

- [1] <http://www.toptal.com/nodejs/why-the-hell-would-i-use-node-js>
- [2] <http://w3techs.com/technologies/details/ws-nodejs/all/all>
- [3] <https://www.udemy.com/blog/learn-node-js/>
- [4] <https://digitalfireflymarketing.com/what-nodejs-good>
- [5] <http://readwrite.com/2013/11/07/what-you-need-to-know-about-nodejs#awesm=~omFREZoh1CbHwJ>
- [6] <http://johnkpaul.github.io/presentations/empirejs/javascript-bad-parts/#/>
- [7] <http://strongloop.com/node-republic/node-js-infographic/>
- [8] <http://www.appdynamics.com/blog/2013/10/17/an-example-of-how-node-js-is-faster-than-php/>
- [9] http://nodejs.org/cinco_de_node.pdf
- [10] <http://java.dzone.com/articles/performance-comparison-between>
- [11] <http://java.dzone.com/articles/performance-comparison-between>
- [12] <http://www.infoq.com/news/2012/10/Ruby-on-Rails-Node-js-LinkedIn>
- [13] <http://www.niwi.be/2013/02/18/python-node-http-performance/>
- [14] <http://blog.kgriffs.com/2012/10/23/python-vs-node-vs-pypy.html>
- [15] <http://nodeblode.wordpress.com/2012/04/10/the-great-node-js-versus-ruby-on-rails-speed-test/>
- [16] <http://w3techs.com/technologies/details/ws-nodejs/all/all>
- [17] <http://www.infoworld.com/d/application-development/nodejs-inventor-extends-javascript-programming-beyond-browsers-184963>
- [18] <http://caines.ca/blog/programming/the-node-js-community-is-quietly-changing-the-face-of-open-source/>
- [19] <http://www.quora.com/Node-js/What-companies-are-using-Node-js-in-production>
- [20] <http://github.com/ry/node>
- [21] <http://blog.stuartherbert.com/php/2012/05/21/if-node-js-is-so-hot-then-where-is-the-ecosystem/#sthash.j01HVzSk.dpuf>
- [22] <http://www.andrewmunsell.com/blog/the-odd-state-of-nodejs-and-its-frameworks/>
- [23] <http://vschart.com/compare/compoundjs/vs/tower-js/vs/meteor-web-framework>

[24] <https://github.com/joyent/node/wiki/Projects,-Applications,-and-Companies-Using-Node>

[25] <http://webdevrefinery.com/forums/topic/7867-the-simple-guide-to-nodejs-frameworks-and-libraries/>