

EXPLORING THE ARCHITECTURE OF THE MEAN STACK

MongoDB, ExpressJS, AngularJS, NodeJS

Scott Davis

Web: <http://thirstyhead.com>

Twitter: [@scottdavis99](https://twitter.com/scottdavis99)

Slides: <http://my.thirstyhead.com>



ThirstyHead.com

training done right.



Scott Davis

@scottdavis99



HTML



developerWorks > Technical topics > Web development > Technical library >

Mastering MEAN: Introducing the MEAN stack

Develop modern, full-stack, twenty-first-century web projects from end-to-end

Build a modern web application with MongoDB, Express, AngularJS, and Node.js in this six-part series by web development expert Scott Davis. This first installment includes a demo, sample code, and full instructions for creating a basic MEAN application. You'll also learn about Yeoman generators that you can use to bootstrap a new MEAN application quickly and easily.

[→ View more content in this series](#) | [PDF \(721 KB\)](#) | [0 Comments](#)

Share:



In his 2002 book, David Weinberger described the burgeoning web's content as a collection of *[Small Pieces Loosely Joined](#)*. That metaphor stuck with me, because it's easy to get tricked into thinking of the web as a monolithic technology stack. Actually, every website you visit is the product of a unique mixture of libraries, languages, and web frameworks.

disciplined adj
led way.
sc jockey n a person who announces
recorded pop records on a radio programme or a
disco.
disclaim vb 1 to deny (responsibility for or knowl
edge of something). 2 to give up (any claim to).
disclaimer n a statement denying responsibility
or knowledge of something.
disclose vb -closing, -closed 1 to make
(information) known. 2 to allow to be seen: she
disclosed the contents of the box. disclose
-cos 1 a nightclub
an occasion
ards. 3







"WHO HAS BEEN TASTING MY SOUP?"

The Goldilocks Framework

This soup is too hot...
too cold...
just right...

A silhouette of a hand pointing towards the right, set against a background of a sunset sky over a body of water. The sky transitions from a deep blue at the top to a bright orange and yellow near the horizon, with the colors reflected in the water below. The hand is dark and detailed, with the index finger pointing forward.

Point #1

In software development, the order of operations should be solution \Rightarrow tools

The problem is we rarely understand the problem

Meet the **LAMP** stack

LAMP (software bundle)

From Wikipedia, the free encyclopedia

(Redirected from [Lamp stack](#))

The acronym **LAMP** refers to first letters of the four components of a [solution stack](#), composed entirely of [free and open-source software](#), suitable for building high-availability heavy-duty [dynamic web sites](#), and capable of serving tens of thousands of requests simultaneously.

The meaning of the LAMP acronym depends on which specific components are used as part of the actual bundle:

- **L**inux, the [operating system](#) (i.e. not just the [Linux kernel](#), but also [glibc](#) and some other essential components of an operating system);
- **A**pache HTTP Server, the [web server](#);
- **M**ySQL, **M**ariaDB or **M**ongoDB, the [database management system](#);
- **P**HP, **P**erl, or **P**ython, the [scripting languages](#) (respectively [programming languages](#)) used for [dynamic web pages](#) and [web development](#).

The exact combination of the software included in a LAMP stack is prone to variation, for example Apache web server can be replaced by some [other web server software](#). Though the original authors of these programs did not design them to work as a component of the LAMP stack, the development philosophy and tool sets are shared and were developed in close conjunction, so they work and scale very well together.

WIMP Stack

Variants and equivalents on other platforms [\[edit\]](#)

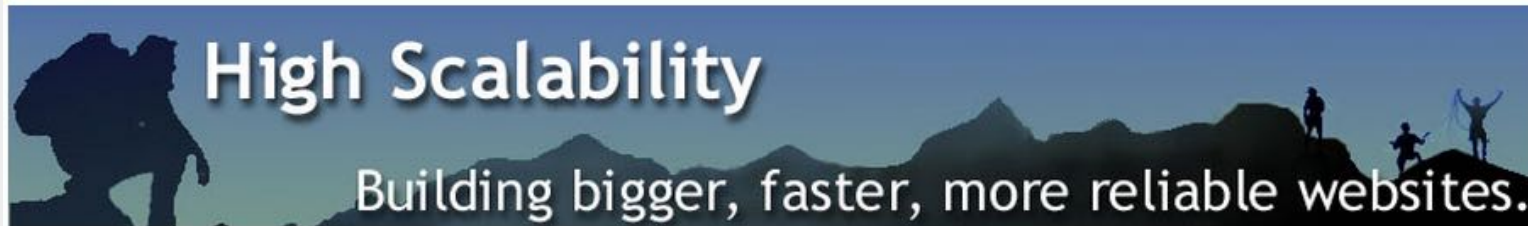
"WAMP" redirects here. For other uses, see [WAMP \(disambiguation\)](#).

Main article: [List of AMP packages](#)

See also: [List of web application frameworks](#)

With the growing use of LAMP, variations and [retronyms](#) appeared for other combinations of operating system, web server, database, and software language. For example the equivalent installation on a [Microsoft Windows](#) operating system is known as [WAMP](#). An alternative running [IIS](#) in place of Apache called [WIMP](#). Variants involving other operating systems include [MAMP](#) ([Macintosh](#)), [SAMP](#) ([Solaris](#)), [FAMP](#) ([FreeBSD](#)) and [iAMP](#) ([iSeries](#)).

<http://highscalability.com>



YouTube Architecture

WEDNESDAY, MARCH 12, 2008 AT 3:54PM

Update 2: [YouTube Reaches One Billion Views Per Day](#). *That's at least 11,574 views per second, 694,444 views per minute, and 41,666,667 views per hour.*

Information Sources

1. [Google Video](#)

Platform


1. Apache
2. Python
3. Linux (SuSe)
4. MySQL
5. psyco, a dynamic python->C compiler
6. lighttpd for video instead of Apache

The Lesser-Known **APLM** Stack...

Platform

1. Apache
2. Python
3. Linux (SuSe)
4. MySQL

http://netcraft.com



Site report for www.wikipedia.org

Search... →

Share: [f](#) [t](#) [in](#) [g+](#) [Y](#) [e8](#)

Lookup another URL:

Background

Network

Last Reboot (28 days ago)

Hosting History

Netblock owner	IP address	OS	Web server	Last seen
Wikimedia Foundation, Inc.	91.198.174.192	Linux	Apache	5-Mar-2015
Wikimedia Foundation, Inc.	91.198.174.225	Linux	Apache	5-Nov-2013
Wikimedia Foundation, Inc.	91.198.174.225	Linux	unknown	1-Nov-2013
Wikimedia Foundation, Inc.	91.198.174.225	Linux	Apache	30-Oct-2013
Wikimedia Foundation Inc. 149 New Montgomery Street 3rd Floor San Francisco CA US 94105	208.80.152.201	-	Apache	10-Oct-2012

Netcraft Extension

- Home
- Download Now!
- Report a Phish
- Site Report
- Top Reporters
- Incentives for reporters
- Phishiest TLDs
- Phishiest Countries
- Phishiest Hosters
- Phishing Map
- Takedown Map
- Most Popular Websites
- Branded Extensions
- Tell a Friend

Phishing & Fraud

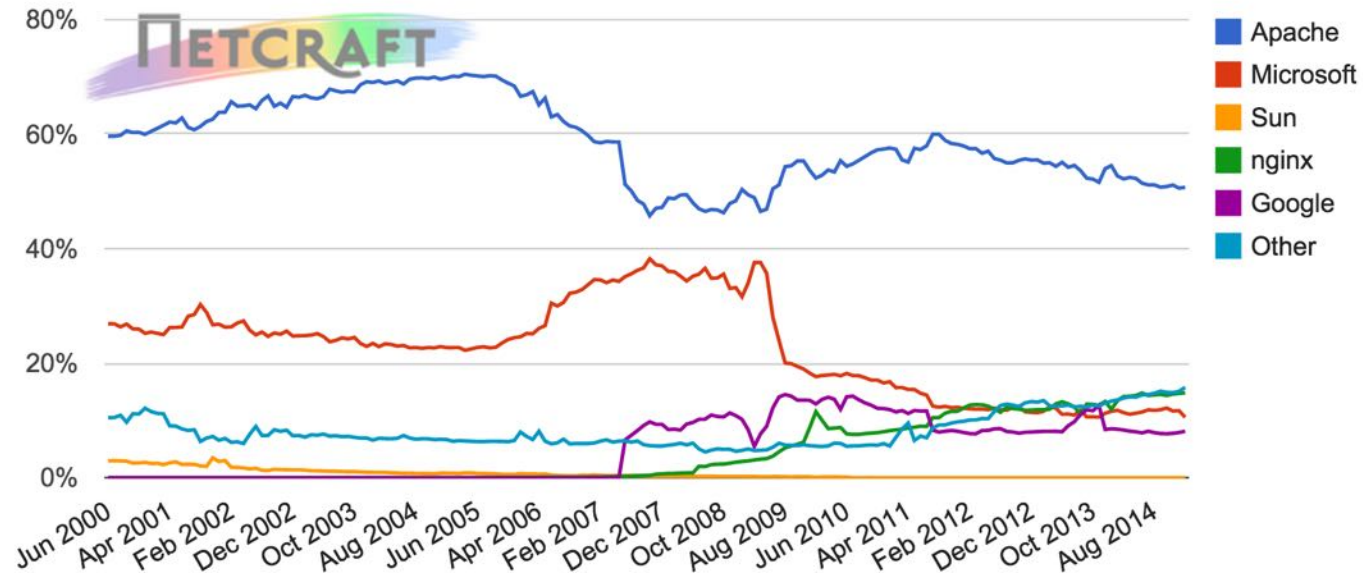
- Phishing Site Feed
- Hosting Phishing Alerts
- SSL CA Phishing Alerts
- Registry Phishing Alerts
- Domain Registration Risk
- Bank Fraud Detection
- Phishing Site Countermeasures

Extension Support

- FAQ

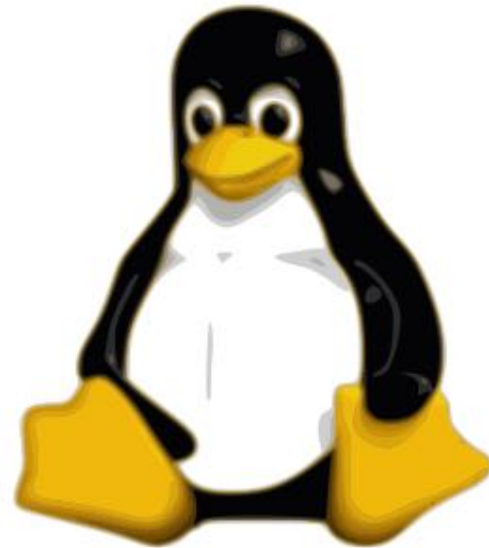
Netcraft Marketshare Report

Web server developers: Market share of active sites

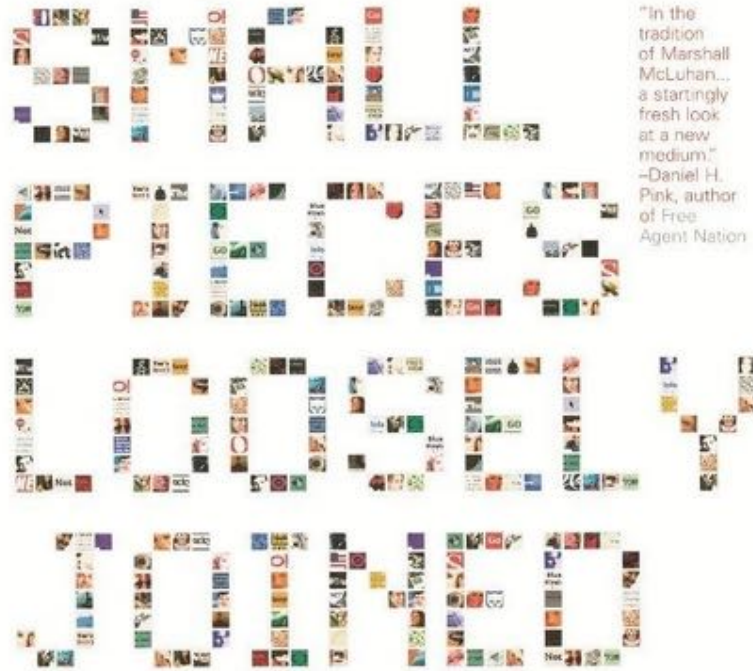


Developer	December 2014	Percent	January 2015	Percent	Change
Apache	90,846,940	50.57%	89,831,550	50.72%	0.15
nginx	26,466,559	14.73%	26,255,870	14.82%	0.09
Microsoft	21,057,292	11.72%	18,684,665	10.55%	-1.17
Google	14,184,320	7.90%	14,378,260	8.12%	0.22

For more information see [Active Sites](#)







"In the tradition of Marshall McLuhan... a startlingly fresh look at a new medium."
—Daniel H. Pink, author of Free Agent Nation

a unified theory of the
web

david weinberger

co-author of the *cluetrain manifesto*

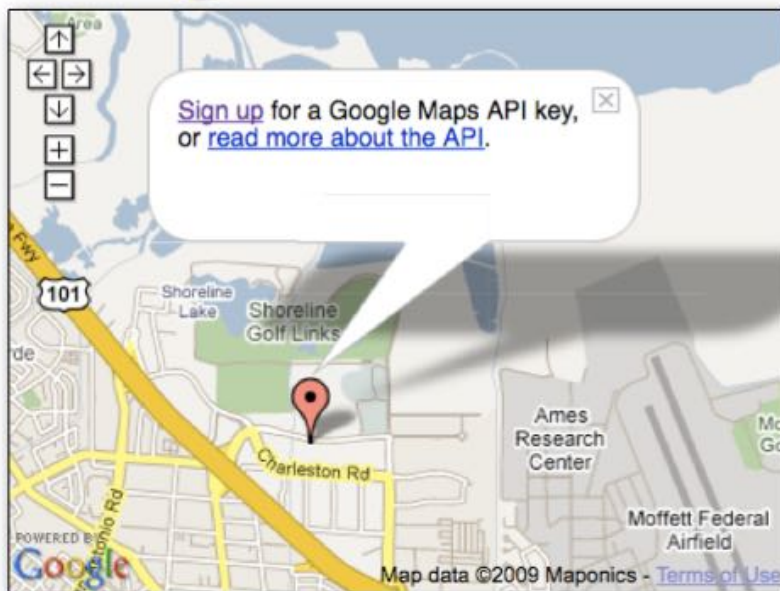
PayPal

Buy Now



Website Payments Standard
Accept credit cards on your website. ☺

Google



You Tube

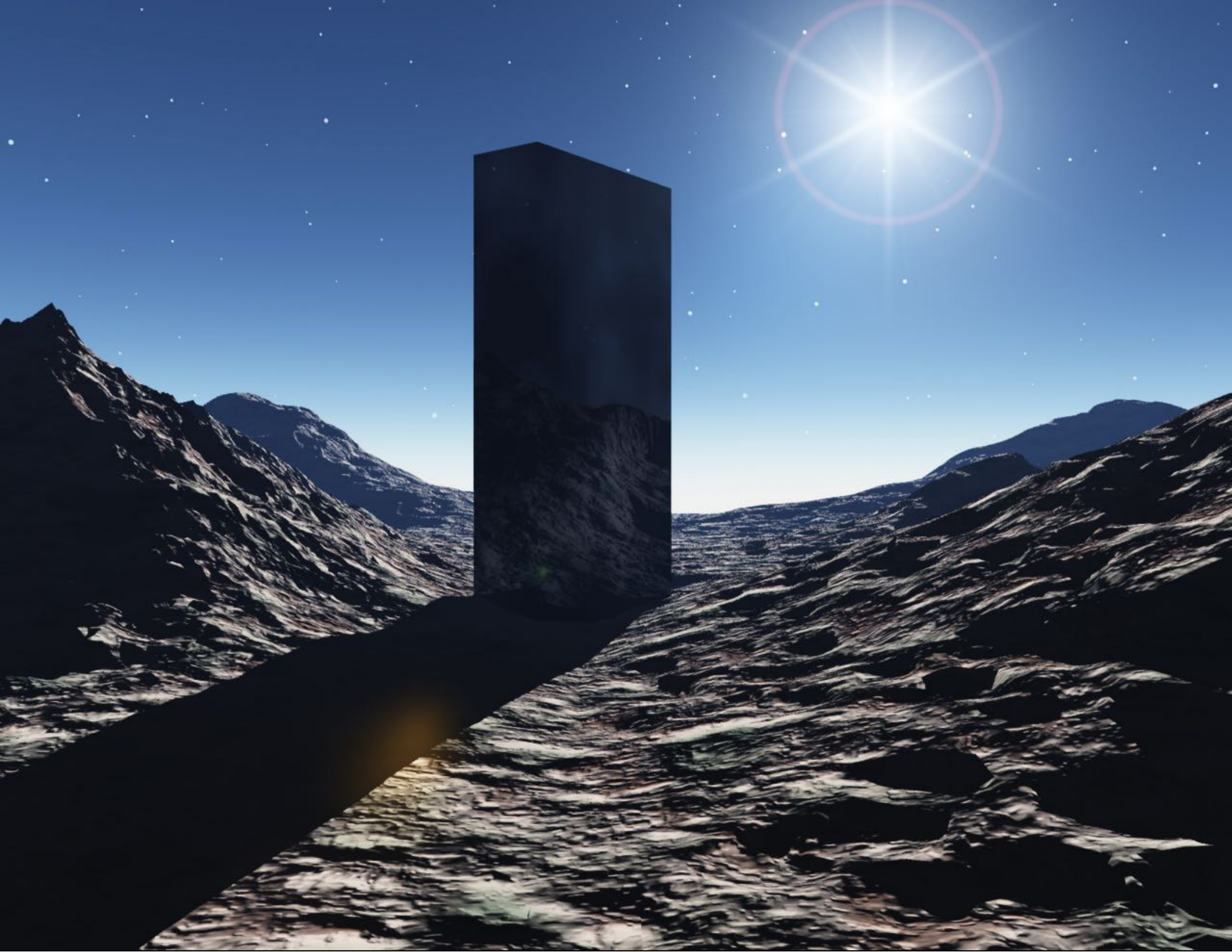
 Broadcast Yourself™
Worldwide | English

How to link to a single YouTube video

Go to the video that you want to share, and look for the **share details** section











Maslow's Hammer

If all you have is a hammer,
then **everything looks like a nail.**

Abraham Maslow
Maslow's Hierarchy of Needs

A silhouette of a hand pointing towards the right, set against a background of a sunset over a body of water. The sky transitions from blue at the top to orange and yellow near the horizon, with the water reflecting these colors. The hand is dark and positioned on the left side of the frame, pointing towards the text on the right.

Point #2

When evaluating web frameworks, favor:
extensibility $\lll \rightarrow$ completeness
plugins $\lll \rightarrow$ core features

And never forget:
lock-in is the new lock-out.

Introducing the **MEAN** stack

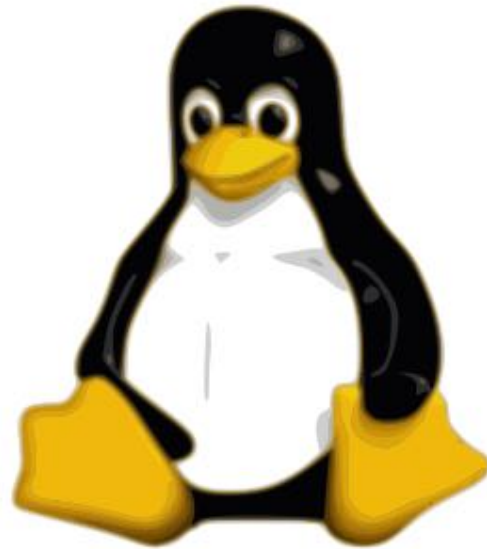
Meet the new boss...



express



...same as the old boss, **right?**





WRONG WAY

Yes, MEAN is:

- Free / open source web stack
- Hip / clever acronym

No, MEAN is not:

- A one-for-one letter swap

LAMP ≠ MEAN

Linux ≠ MongoDB

Apache ≠ ExpressJS

MySQL ≠ AngularJS

Perl ≠ NodeJS

LAMP ➡ NEMA

Linux ➡ NodeJS

Apache ➡ ExpressJS

MySQL ➡ MongoDB

Perl ➡ AngularJS

LAMP NEMA

Linux  NodeJS (Platform)

Apache  ExpressJS (Web Server)

MySQL  MongoDB (Persistence)

Perl  AngularJS (User Interface)



<http://meanjs.org/>

[Docs](#)[Yo Generator](#)[Modules](#)[Changelog](#)[Community](#)[Blog](#)

Open-Source Full-Stack Solution For MEAN Applications

What is MEAN.JS?

MEAN.JS is a full-stack JavaScript solution that helps you build fast, robust and maintainable production web applications using MongoDB, Express, AngularJS, and Node.js.

Why MEAN.JS?

MEAN.JS will help you getting started and avoid useless grunt work and common pitfalls, while keeping your application organized. Our goal is to create and maintain a simple and readable open-source solution that you can use and trust in your projects.



Community Experience Distilled

MEAN Web Development

Master real-time web application development using a mean combination of MongoDB, Express, AngularJS, and Node.js

Amos Q. Haviv

[PACKT] open source★
PUBLISHED

developerWorks > Technical topics > Web development > Technical library >

Mastering MEAN: Introducing the MEAN stack

Develop modern, full-stack, twenty-first-century web projects from end-to-end

Build a modern web application with MongoDB, Express, AngularJS, and Node.js in this six-part series by web development expert Scott Davis. This first installment includes a demo, sample code, and full instructions for creating a basic MEAN application. You'll also learn about Yeoman generators that you can use to bootstrap a new MEAN application quickly and easily.

[→ View more content in this series](#) | [PDF \(721 KB\)](#) | [0 Comments](#)

Share:



In his 2002 book, David Weinberger described the burgeoning web's content as a collection of *[Small Pieces Loosely Joined](#)*. That metaphor stuck with me, because it's easy to get tricked into thinking of the web as a monolithic technology stack. Actually, every website you visit is the product of a unique mixture of libraries, languages, and web frameworks.

MEAN.JS



mongoDB

express



ANGULARJS
by Google

node JS™



Scaffolding App

[Docs](#)[Yo Generator](#)[Modules](#)[Changelog](#)[Community](#)[Blog](#)[Overview](#)[Getting Started](#)[Application](#)[CRUD Module](#)[AngularJS Module](#)[AngularJS Route](#)[AngularJS Controller](#)[AngularJS View](#)[AngularJS Service](#)[AngularJS Directive](#)[AngularJS Filter](#)[AngularJS Config](#)[AngularJS Test](#)[Express Model](#)

Application Generator

The application generator will help you create a fresh copy of a MEAN.JS application in your working folder. To create your MEAN application, navigate to a new project folder, and then use `yo` to generate your application:

```
$ yo meanjs
```

The generator will ask you a few questions about your new application and will generate it for you. When the installation process is over, you will be able to use `grunt` to run your new MEAN application:

```
$ grunt
```

Now, the application generator does a great job scaffolding a whole application, but daily work requires us to repeat a lot of structured code. For this purpose we provided you with some sub-generators to help you speed up your development.

Scaffolding **CRUD**

[Docs](#)[Yo Generator](#)[Modules](#)[Changelog](#)[Community](#)[Blog](#)[Overview](#)[Getting Started](#)[Application](#)[CRUD Module](#)[AngularJS Module](#)[AngularJS Route](#)[AngularJS Controller](#)[AngularJS View](#)[AngularJS Service](#)[AngularJS Directive](#)[AngularJS Filter](#)

CRUD Module Sub-Generator

The CRUD module sub-generator will help you create a new CRUD module, similar to the article sample provided with the project. To create a new CRUD module you will need to use *yo* again:

```
$ yo meanjs:crud-module <module-name>
```

This will create both AngularJS and Express files supporting full CRUD functionality, and add the Karma and Mocha tests.

Note: Don't forget to use your module name as an argument when calling the CRUD module sub-generator.

MEAN.JS



mongoDB

express



ANGULARJS
by Google



node JS



YO



GRUNT



BOWER



Passport

Simple, unobtrusive authentication for Node.js.

mongoose

A silhouette of a hand pointing towards the right, set against a background of a sunset sky over a body of water. The sky transitions from a deep blue at the top to a bright orange and yellow near the horizon, with the colors reflected in the water below. The hand is dark and detailed, with the index finger extended forward.

Point #3

A good framework doesn't
move the goalposts;
it moves the starting line.

Pick one that includes
scaffolding, build scripts,
testing, deployment, etc.

NodeJS === Platform

http://nodejs.org



[HOME](#) | [DOWNLOADS](#) | [DOCS](#) | [COMMUNITY](#) | [ABOUT](#) | [JOBS](#) | [BLOG](#)

Node.js® is a platform built on **Chrome's JavaScript runtime** for easily building fast, 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.

Current Version: v0.12.0

[INSTALL](#)

[DOWNLOADS](#)

[API DOCS](#)



Node.js allows us to build our real-time cloud IDE with a single language front to back. It makes life easier for both us and our users to write, run, and debug code, anywhere, anytime.

[Rik Arends](#)

CTO



Node's evented I/O model freed us from worrying about locking and concurrency issues that are common with multithreaded async I/O.

[Subbu Allamarju](#)

Principal Member, Technical Staff



Node.js is the execution core of Manhattan. Allowing developers to build one code base using one language - that is the nirvana for developers.

[Renaud Waldura](#)

Sr. Product Manager, Cocktail



Node puts the magic in the right places. We write our application, and node delivers JSON over HTTP.

[Matt Ranney](#)

CTO

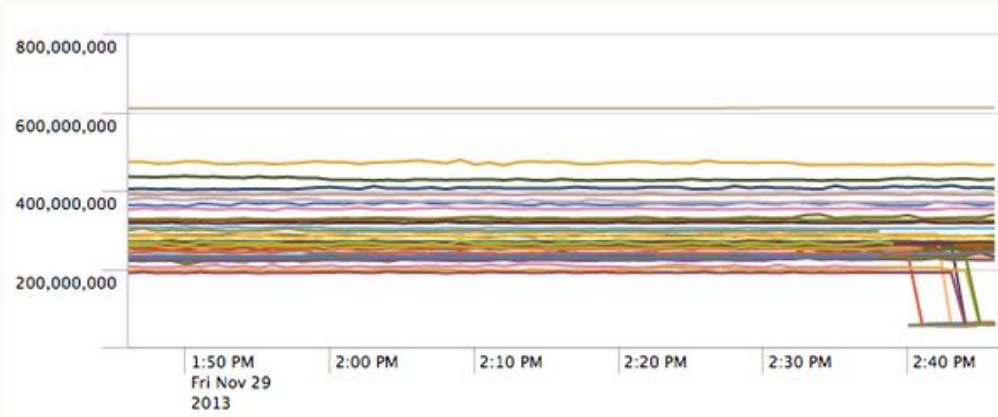
High-Performance

PayPal reported: **double the number of requests per-second and reduced response time by 35% or 200 milliseconds.**

WalMart Labs had a bumper launch with Node.js in 2013, where they put all of their Mobile traffic through Node.js on black-friday, the busiest shopping period of the year.


The team at WalMart Labs live tweeted against #nodebf tag showing the performance of the Node.js application.

On Black Friday the WalMart servers didn't go over 1% CPU utilisation and the team did a deploy with 200,000,000 users online.



Eran Hammer @eranhammer · Nov 30

I guess too nuts for anyone to guess: we felt so good about everything, we decided middle of Black Friday is perfect time for a release.

Linux  **NodeJS**

Downloads

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

Current version: **v0.12.0**

 Windows Installer node-v0.12.0-x86.msi	 Macintosh Installer node-v0.12.0.pkg	 Source Code node-v0.12.0.tar.gz
---	---	--

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.exe)	32-bit	64-bit
Mac OS X Installer (.pkg)	Universal	
Mac OS X Binaries (.tar.gz)	32-bit	64-bit
Linux Binaries (.tar.gz)	32-bit	64-bit
SunOS Binaries (.tar.gz)	32-bit	64-bit
Source Code	node-v0.12.0.tar.gz	

Note: Python 2.6 or 2.7 is required to build from source tarballs.

http://nodejs.org

[HOME](#)[DOWNLOADS](#)[DOCS](#)[COMMUNITY](#)[ABOUT](#)[JOBS](#)[BLOG](#)

Node.js® is a platform built on **Chrome's JavaScript runtime** for easily building fast, 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.

Current Version: v0.12.0

[INSTALL](#)[DOWNLOADS](#)[API DOCS](#)



Browser

***Rendering
Engine***
(HTML, CSS)

***Scripting
Engine***
(JavaScript)

Plugins
***(Flash,
Silverlight)***

YouTube drops Flash for HTML5 video as default

By [Rich McCormick](#) on January 27, 2015 10:22 pm [Email](#)

The slow death of Adobe Flash has been hastened — YouTube, which used the platform as the standard way to play its videos, [has dumped Flash in favor of HTML5 for its default web player](#). The site will now use HTML5 video as standard in Chrome, Internet Explorer 11, Safari 8, and in beta versions of Firefox. YouTube engineer Richard Leider said the time had come to ditch the aging Flash in favor of HTML5 as the latter, used in smart TVs and other streaming devices, had benefits that "extend beyond web browsers."

YouTube has spent years experimenting with HTML5, and engineer John Harding [wrote about its benefits in 2010](#). Harding said that although HTML5 let YouTube bring videos to devices that don't support Flash Player, such as the iPhone, it did not sufficiently meet the site's needs at the time. Almost five years later, the proliferation and advance of HTML5 means that YouTube can now use it for its default player in most modern browsers.

HTML5 IS NOW YOUTUBE'S DEFAULT ON CHROME, IE 11, SAFARI 8, AND FIREFOX BETAS

Exclusive: Adobe ceases development on mobile browser Flash, refocuses efforts on HTML5 (UPDATED)

Summary: *Adobe has briefed developers on the impending cessation of mobile flash browser plugin development.*



By [Jason Perlow](#) for Tech Broiler | November 8, 2011 -- 21:17 GMT (13:17 PST)

Sources close to Adobe that have been briefed on the company's future development plans have revealed this forthcoming announcement to ZDNet:

Our future work with Flash on mobile devices will be focused on enabling Flash developers to package native apps with Adobe AIR for all the major app stores. We will no longer adapt Flash Player for mobile devices to new browser, OS version or device configurations. Some of our source code licensees may opt to continue working on and releasing their own implementations. We will continue to support the current Android and PlayBook configurations with critical bug fixes and security updates.

Browser

***Rendering
Engine***

(HTML, CSS)

***Scripting
Engine***

(JavaScript)

Web browser engine

From Wikipedia, the free encyclopedia

A **web browser engine** (sometimes called **layout engine** or **rendering engine**) is a software **component** that takes **marked up** content (such as **HTML**, **XML**, **image** files, etc.) and formatting information (such as **CSS**, **XSL**, etc.) and displays the formatted content on the screen. It draws onto the content area of a window, which is displayed on a **monitor** or a **printer**. A layout engine is typically embedded in **web browsers**, **e-mail clients**, **e-book readers**, on-line help systems or other applications that require the displaying (and editing) of web content. Engines may wait for all data to be received before rendering a page, or may begin rendering before all data is received. This can result in pages changing as more data is received, such as images being filled in or a **flash of unstyled content** if rendering begins before formatting information is received.







Contents [hide]

- 1 Examples
- 2 Technical operation
- 3 Timeline
- 4 See also
- 5 References

Examples [edit]

KDE's open-source **KHTML** engine is used in KDE's **Konqueror** web browser and was the basis for **WebKit**, the rendering engine in **Apple's Safari** and **Google's Chrome** web browsers, which is now the most widely used browser engine according to **StatCounter**. Current versions of Chromium/Chrome (except iOS version) and **Opera** are based on **Blink**, a fork of WebKit.

Non-mobile web browser statistics on Wikipedia projects

Google Chrome, Opera and other variants (Blink)		43.00%
Internet Explorer (Trident)		25.80%
Firefox and other variants (Gecko)		18.22%
Safari and other variants (WebKit)		5.90%
Opera (Presto)		2.31%
Others		4.77%

Non-mobile web browser usage for Wikimedia visitors as of February 2014.^[1]

JavaScript engines [\[edit\]](#)

Active projects [\[edit\]](#)

- [Spidermonkey](#), the first-ever JavaScript engine, which powered [Netscape Navigator](#) and today powers [Firefox](#)
- [V8](#) - open source, developed by Google in Denmark, part of Google Chrome
- [JavaScriptCore](#) - open source, marketed as Nitro and developed by [Apple](#) for [Safari](#)
- [KJS](#) - KDE's ECMAScript/JavaScript engine originally developed by [Harri Porten](#) for the KDE project's Konqueror web browser
- [Chakra](#), for [Internet Explorer 9](#)^[16]
- [dyn.js](#), open source, written by Douglas Campos and others^[17]
- [Nashorn](#), open source as part of OpenJDK, written by Oracle Java Languages and Tool Group^[18]
- [Juce](#), a C++ [application framework](#), contains a custom embedded interpreter using part of JavaScript's syntax.
- [duktape](#), open source, embeddable, C, MIT-licensed Javascript engine, with a focus on portability and compact footprint
- [MuJS](#), open source, embeddable and extensible interpreter developed for [MuPDF](#).

Inactive projects [\[edit\]](#)

- [Tamarin](#), by [Adobe Labs](#)
- [Carakan](#), by [Opera Software](#), used by [Opera](#) web browser version 10.50 until switching to [V8](#) with Opera 14 (released in 2013).^{[19][20]}
- [Futhark](#), by [Opera Software](#), used by [Opera](#) web browser versions 9.50 to 10.10 until replaced by [Carakan](#) in [Opera 10.50](#) (released March 2010).
- [Narcissus](#) open source, written by [Brendan Eich](#), who also wrote [SpiderMonkey](#)
- [Rhino](#), managed by the [Mozilla Foundation](#), open source, developed entirely in [Java](#)



Firefox



Gecko
(HTML,
CSS)

**Spider
Monkey**
(JavaScript)

IE



Trident
(HTML,
CSS)

Chakra
(JavaScript)

Chrome



Blink
(HTML,
CSS)

V8
(JavaScript)



Opera

Blink
(HTML,
CSS)

V8
(JavaScript)

Safari




WebKit
(HTML,
CSS)

Nitro
(JavaScript)

Android 4.4 KitKat, the browser and the Chrome WebView

 66

 Tweet 198

 Like 87 people like this.




Android 4.4 has made a big change in the OS' internals for HTML5 development: it has replaced its original WebKit-based WebView with modern

Chromium. The new Android Browser is also powered by Chromium but it's not clear yet its future. Besides the good news, not everything looks exciting in these changes, let's see why.

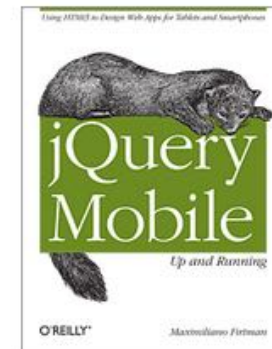
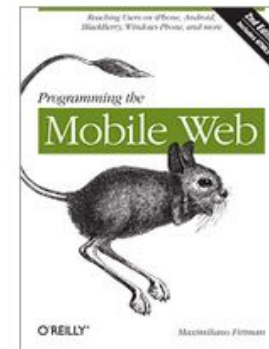
Every web developer that has played with native webapps, PhoneGap and the Android's WebView knows how terrible it was in terms of performance and HTML5 compatibility. The same problems that most web developers suffer right now with the Android Browser -reported to be 32% of the mobile web browsing market share, compared with just 5% of the modern Chrome for

Thoughts, researchs and samples about mobile web by [Maximiliano Firtman](#), speaker, trainer and author of O'Reilly books [Programming the Mobile Web](#) and [jQuery Mobile](#).

 Follow me @firt

 Follow this blog

MY BOOKS



WebKit

From Wikipedia, the free encyclopedia
(Redirected from [Webkit](#))

WebKit is a [layout engine software component](#) for rendering [web pages](#) in [web browsers](#). It powers [Apple's Safari web browser](#), and a [fork](#) of the project is used by [Google's Chrome web browser](#). By September 2013, [WebKit browser market share](#)^[5] was larger than that of both the [Trident engine](#) used by [Internet Explorer](#) and the [Gecko engine](#) used by [Firefox](#).

WebKit also forms the basis for the experimental browser included with the [Amazon Kindle e-book](#) reader, as well as the default browser in the [Apple iOS](#), [BlackBerry Browser](#) in OS 6 and above, and [Tizen](#) mobile operating systems. WebKit's [C++ application programming interface](#) provides a set of [classes](#) to display web content in windows, and implements browser features such as following links when clicked by the user, managing a back-forward list, and managing a history of pages recently visited.

WebKit



Original author(s)	KDE ^{[1][2]}
Developer(s)	Apple , Adobe , and others
Initial release	November 4, 1998; 16 years ago (KHTML released) June 7, 2005; 9 years ago (WebKit open sourced)
Preview release	Nightly ^[3]
Written in	C++
Operating system	Cross-platform ^[4]
Type	Layout engine

Hackers “Break” PS4 Firmware 1.76 – Webkit Exploit Now Available for the Console

GAMES 4 months ago by [Fahad Arif](#)

 Tweet 26

 Like Share 226



While Sony is getting a kick out of the immense success of its latest PlayStation 4 console, hackers are busy trying to break the code and find a tractable way into the system, and it looks like they have already done the trick. Following the PlayStation Vita webkit exploit that was released almost a week ago, two hackers have now successfully released PlayStation 4 webkit exploit by working on, and extending the hack that was used on Sony’s handheld console. The latest webkit exploit breaks the current firmware 1.76 of the PlayStation 4 through a vulnerability found in the web browser of the console.

 [Twitter](#)

 [Facebook](#)





PhantomJS

[SOURCE CODE](#)[DOCS](#)

Full web stack No browser required

PhantomJS is a headless WebKit scriptable with a JavaScript API. It has **fast** and **native** support for various web standards: DOM handling, CSS selector, JSON, Canvas, and SVG.

[Download v2.0](#)[Get started](#)



SlimerJS

Free - Open Source

A scriptable browser for Web developers

Download SlimerJS

All operating systems - Version 0.9.5

Compatible with [CasperJS 1.1 beta!!](#)

[Release notes](#)

SlimerJS allows you to interact with a web page through an external JS script

- > Opening a webpage,
- > Clicking on links,
- > Modifying the content...

SlimerJS is useful to do functional tests, page automation, network monitoring, screen capture, etc.

SlimerJS is similar to *PhantomJS*, except that it runs on top of **Gecko**, the browser engine of **Mozilla Firefox** (specifically, version 31), instead of **Webkit**, and is not yet truly headless.

Simple example

```
var webpage = require('webpage').create();
webpage
  .open('http://somewhere') // loads a page
  .then(function(){ // executed after loading
    // store a screenshot of the page
    webpage.viewportSize =
      { width:650, height:320 };
    webpage.render('page.png',
      {onlyViewport:true});
    // then open a second page
    return webpage.open('http://somewhere
```

http://nodejs.org



[HOME](#) | [DOWNLOADS](#) | [DOCS](#) | [COMMUNITY](#) | [ABOUT](#) | [JOBS](#) | [BLOG](#)

Node.js® is a platform built on **Chrome's JavaScript runtime** for easily building fast, 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.

Current Version: v0.12.0

[INSTALL](#)

[DOWNLOADS](#)

[API DOCS](#)



Point #4

Be familiar with browsers,
but think in terms of
rendering engines and
scripting engines

Truckers, don't be fooled
by the label on the can:
it's the fizzy pop inside
that matters...

V8 (JavaScript engine)

From Wikipedia, the free encyclopedia

The **V8 JavaScript Engine** is an [open source](#) JavaScript engine developed by [Google](#) for the [Google Chrome](#) web browser.^[4] It has since seen use in many other projects^[*citation needed*]. As of 2012, the head programmer is [Lars Bak](#).^[5] The first version of the V8 engine was released at the same time as the first version of Chrome, September 2, 2008.

V8 compiles JavaScript to native [machine code](#) (IA-32, x86-64, ARM, or MIPS ISAs)^{[3][6]} before executing it, instead of more traditional techniques such as [interpreting](#) bytecode or compiling the whole program to machine code and executing it from a filesystem. The compiled code is additionally optimized (and re-optimized) dynamically at runtime, based on heuristics of the code's execution profile. Optimization techniques used include [inlining](#), [elision](#) of expensive runtime properties, and [inline caching](#), among many others.

V8 JavaScript Engine



Developer(s)	Google
Stable release	4.1.0 ^[1] / March 3, 2015; 4 days ago
Development status	Active
Written in	C++, ^[2] JavaScript ^[2]
Operating system	Cross-platform
Platform	IA-32, x86-64, ARM, MIPS ^[3]
Type	JavaScript engine
License	BSD
Website	code.google.com/p/v8 



Lars Bak (computer programmer)

From Wikipedia, the free encyclopedia

Lars Bak is a Danish [computer programmer](#). He is known as a [JavaScript](#) expert and for his work on [virtual machines](#). He is currently employed by [Google](#), having contributed to the [Chrome browser](#) by developing the [V8 JavaScript engine](#). After years abroad, Lars Bak now lives near [Aarhus](#) in [Denmark](#).^[1]

Professional life [edit]

Bak studied at [Aarhus University](#) in Denmark, receiving an MS degree in [computer science](#) in 1988 after which he became active in designing and implementing object-oriented [virtual machines](#).

Virtual machines [edit]

After participating in the design and implementation of the [BETA Mjølner System](#), in 1991 he joined the Self group at [Sun Microsystems Laboratories](#) in [Cupertino, California](#). During his time there, he developed a programming environment for [Self](#) and added several enhancements to the virtual machine.

In 1994, he joined [LongView Technologies](#) LLC, where he designed and implemented high performance virtual machines for both [Smalltalk](#) and [Java](#). After Sun Microsystems acquired LongView in 1997, Bak became engineering manager and technical lead in the [HotSpot](#) team at Sun's Java Software Division where he developed a high-performance [Java virtual machine](#).^{[2][3]}

Java **!==** JavaScript

JavaScript sounds like it has something to do with Java. It doesn't.

Apart from some superficial syntactical similarities, they have nothing in common.

Java is to JavaScript as ham is to hamster.



Jeremy Keith

The A-Z of Programming Languages: JavaScript

Brendan Eich created JavaScript in 1995 with the aim to provide a "glue language" for Web designers and par
grown to become one of the most widely used languages on the planet.

Naomi Hamilton (Computerworld) | 31 July, 2008 21:04



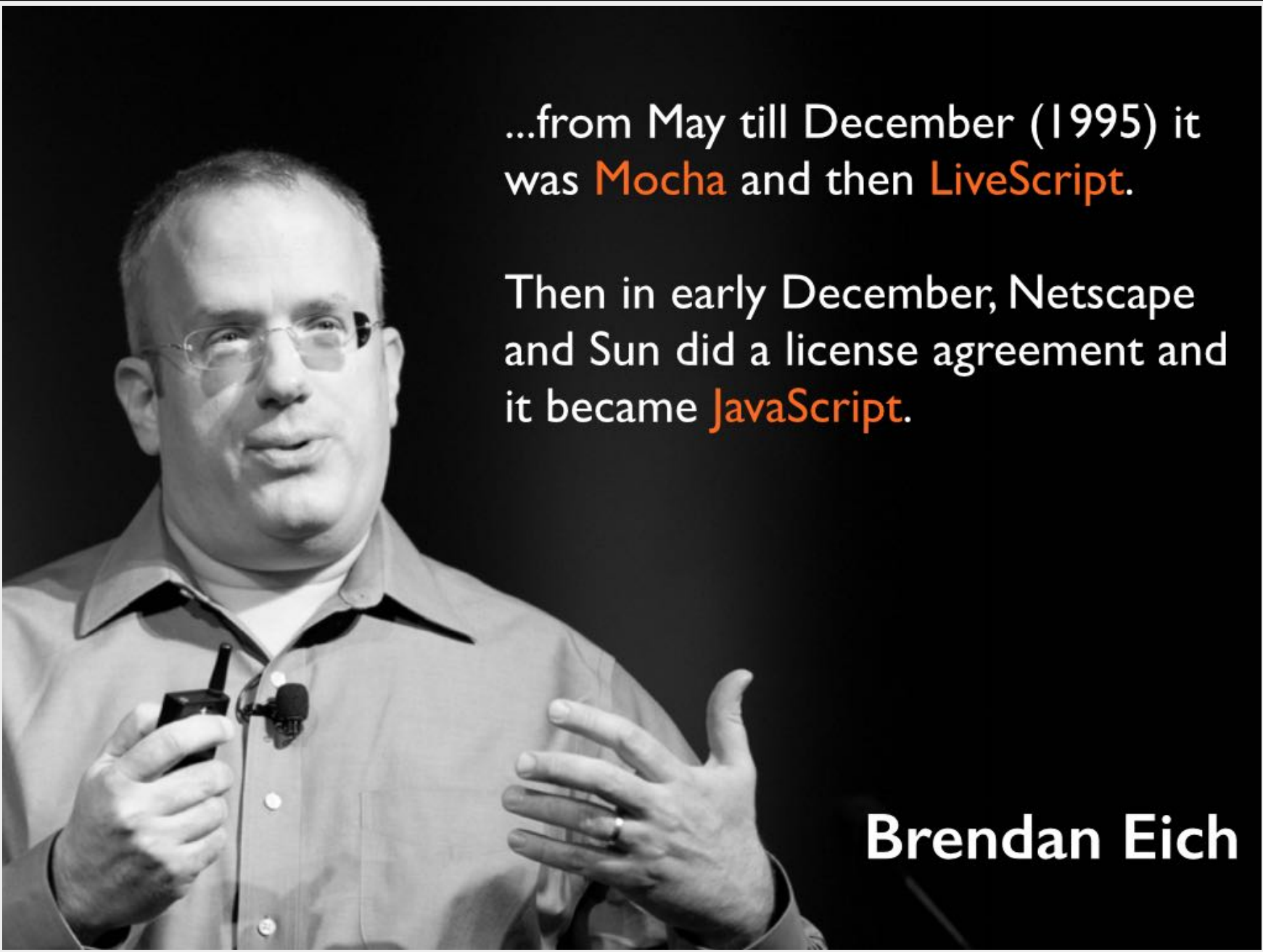
I joined Netscape on 4
April 1995, with the goal of
**embedding the Scheme
programming language,**
or something like it, into
Netscape's browser.

Scheme (programming language)

From Wikipedia, the free encyclopedia

This article is about the programming language. For other uses, see [Scheme](#).

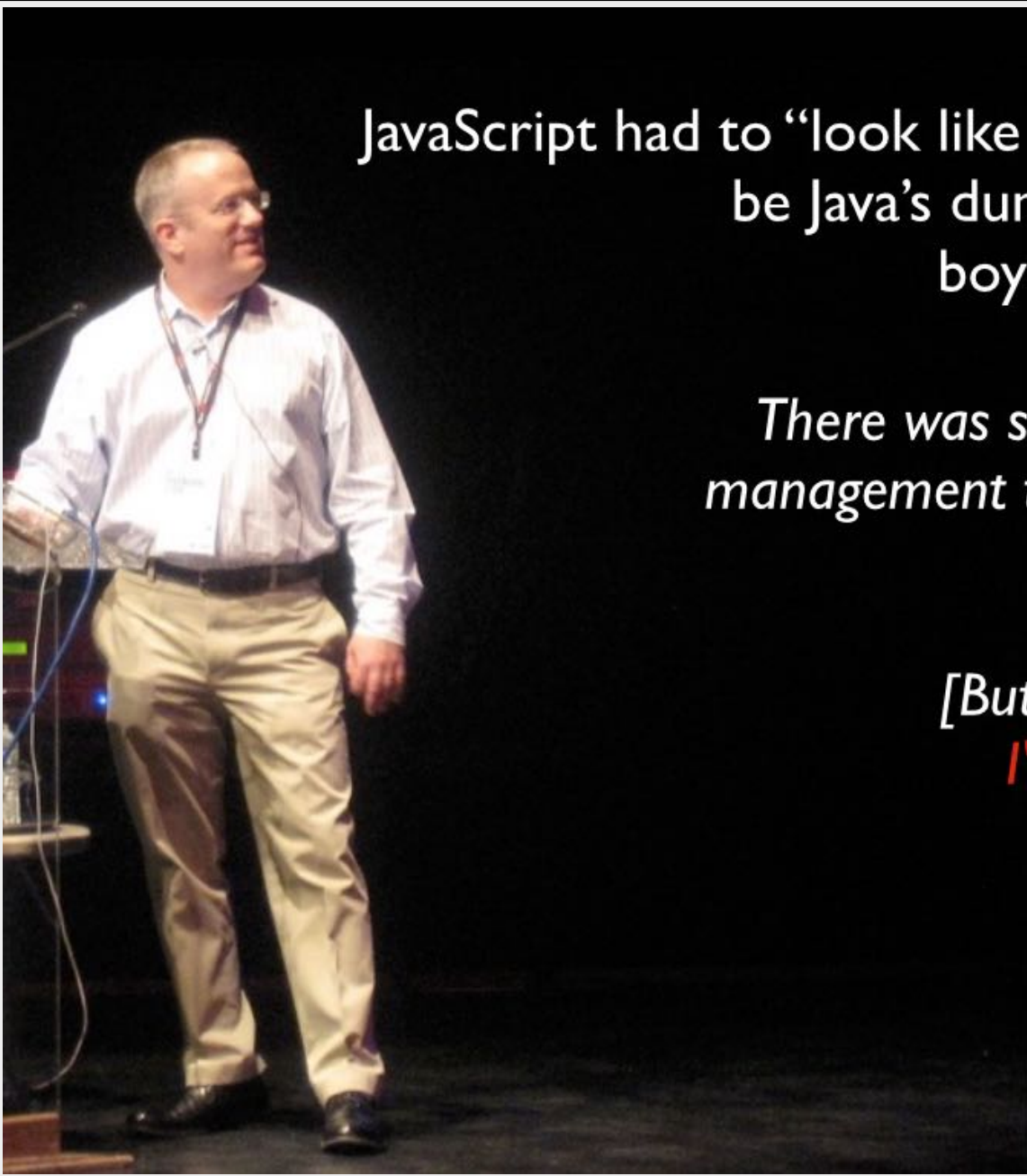
Scheme is one of the two main [dialects](#) of the programming language [Lisp](#). Unlike [Common Lisp](#), the other main dialect, Scheme follows a [minimalist](#) design philosophy specifying a small standard core with powerful tools for language extension. Its compactness and elegance have made it popular with educators, language designers, programmers, implementors, and hobbyists. The language's



...from May till December (1995) it was **Mocha** and then **LiveScript**.

Then in early December, Netscape and Sun did a license agreement and it became **JavaScript**.

Brendan Eich



JavaScript had to “look like Java” only less so,
be Java’s dumb kid brother or
boy-hostage sidekick.

*There was some pressure from
management to make the syntax
look like Java...*

*[But] if I put classes in,
I'd be in big trouble.*



Java was in some ways a negative influence.

I didn't want to have anything “classy.”

So I swerved from that and it caused me to look at **Self** and do **prototypes**.

Self (programming language)

From Wikipedia, the free encyclopedia

Self is an [object-oriented programming language](#) based on the concept of [prototypes](#). Essentially an extreme dialect of [Smalltalk](#), it was used mainly as an experimental test system for language design in the 1980s and 1990s. In 2006, Self was still being developed as part of the Klein project, which was a Self virtual machine written fully in Self. The latest version is 4.4, released in July 2010.

Prototype-based programming

From Wikipedia, the free encyclopedia

Prototype-based programming is a style of [object-oriented programming](#) in which [classes](#) are not present, and behavior reuse (known as [inheritance](#) in class-based languages) is performed via a process of [cloning](#) existing [objects](#) that serve as [prototypes](#). This model can also be known as *classless*, *prototype-oriented* or *instance-based* programming. [Delegation](#) is the language feature that supports prototype-based programming.



Lars Bak (computer programmer)

From Wikipedia, the free encyclopedia

Lars Bak is a Danish [computer programmer](#). He is known as a [JavaScript](#) expert and for his work on [virtual machines](#). He is currently employed by [Google](#), having contributed to the [Chrome browser](#) by developing the [V8 JavaScript engine](#). After years abroad, Lars Bak now lives near [Aarhus](#) in [Denmark](#).^[1]

Professional life [edit]

Bak studied at [Aarhus University](#) in Denmark, receiving an MS degree in [computer science](#) in 1988 after which he became active in designing and implementing object-oriented [virtual machines](#).

Virtual machines [edit]

After participating in the design and implementation of the [BETA Mjølner System](#), in 1991 he joined the Self group at [Sun Microsystems Laboratories](#) in [Cupertino, California](#). During his time there, he developed a programming environment for [Self](#) and added several enhancements to the virtual machine.

In 1994, he joined [LongView Technologies](#) LLC, where he designed and implemented high performance virtual machines for both [Smalltalk](#) and [Java](#). After Sun Microsystems acquired LongView in 1997, Bak became engineering manager and technical lead in the [HotSpot](#) team at Sun's Java Software Division where he developed a high-performance [Java virtual machine](#).^{[2][3]}

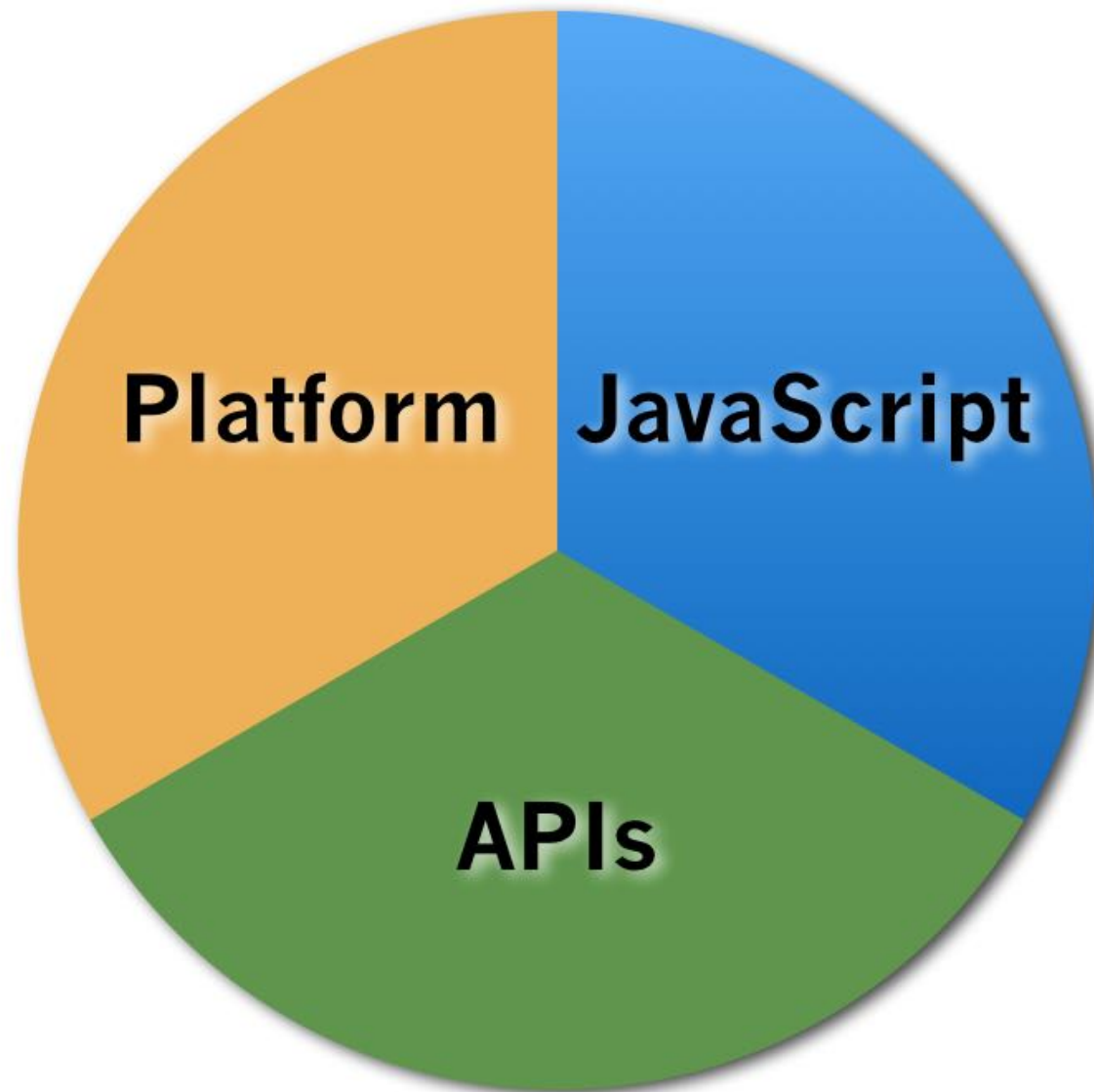
Headless JavaScript?

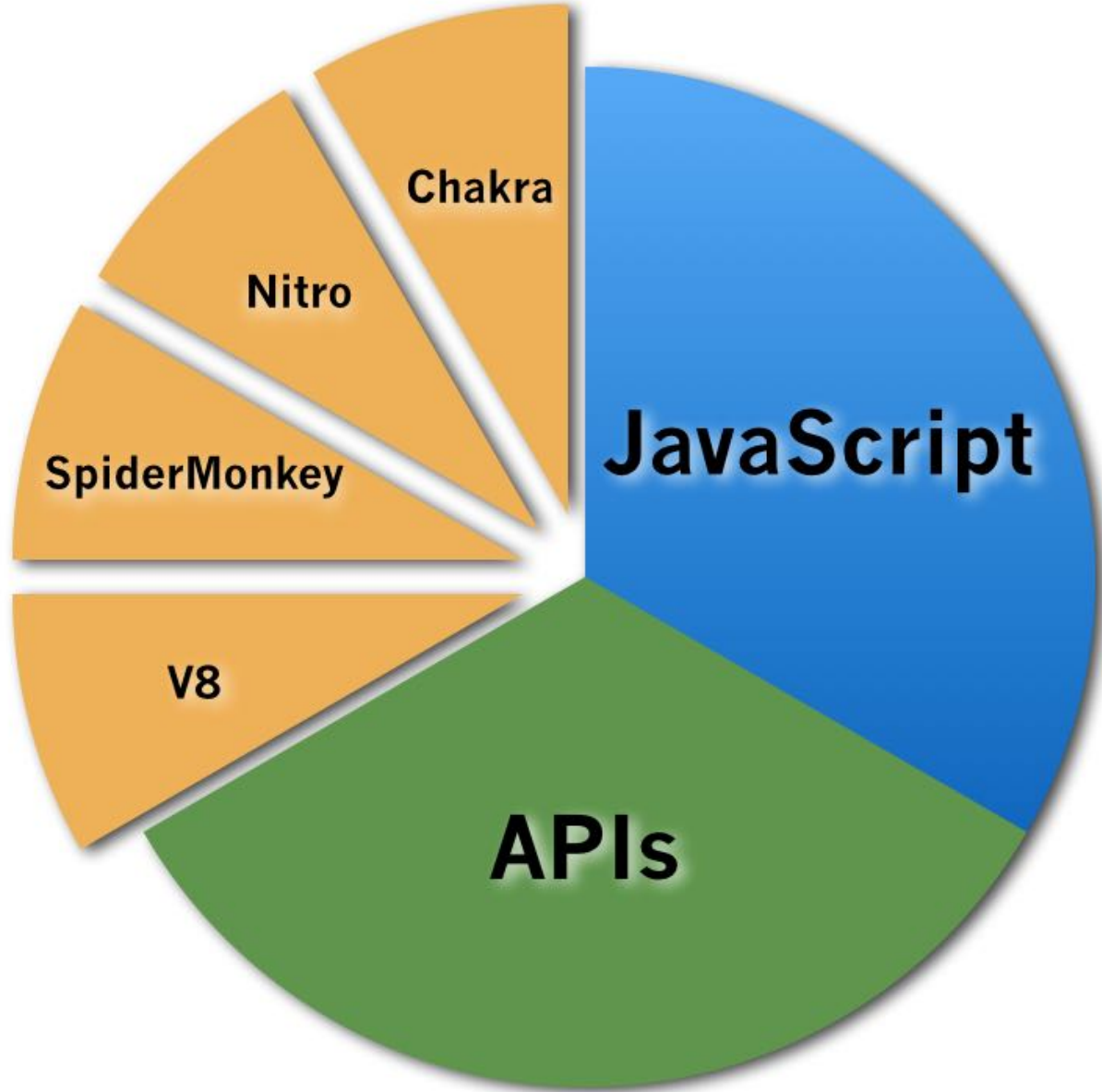
JavaScript **outside** of the browser?!?

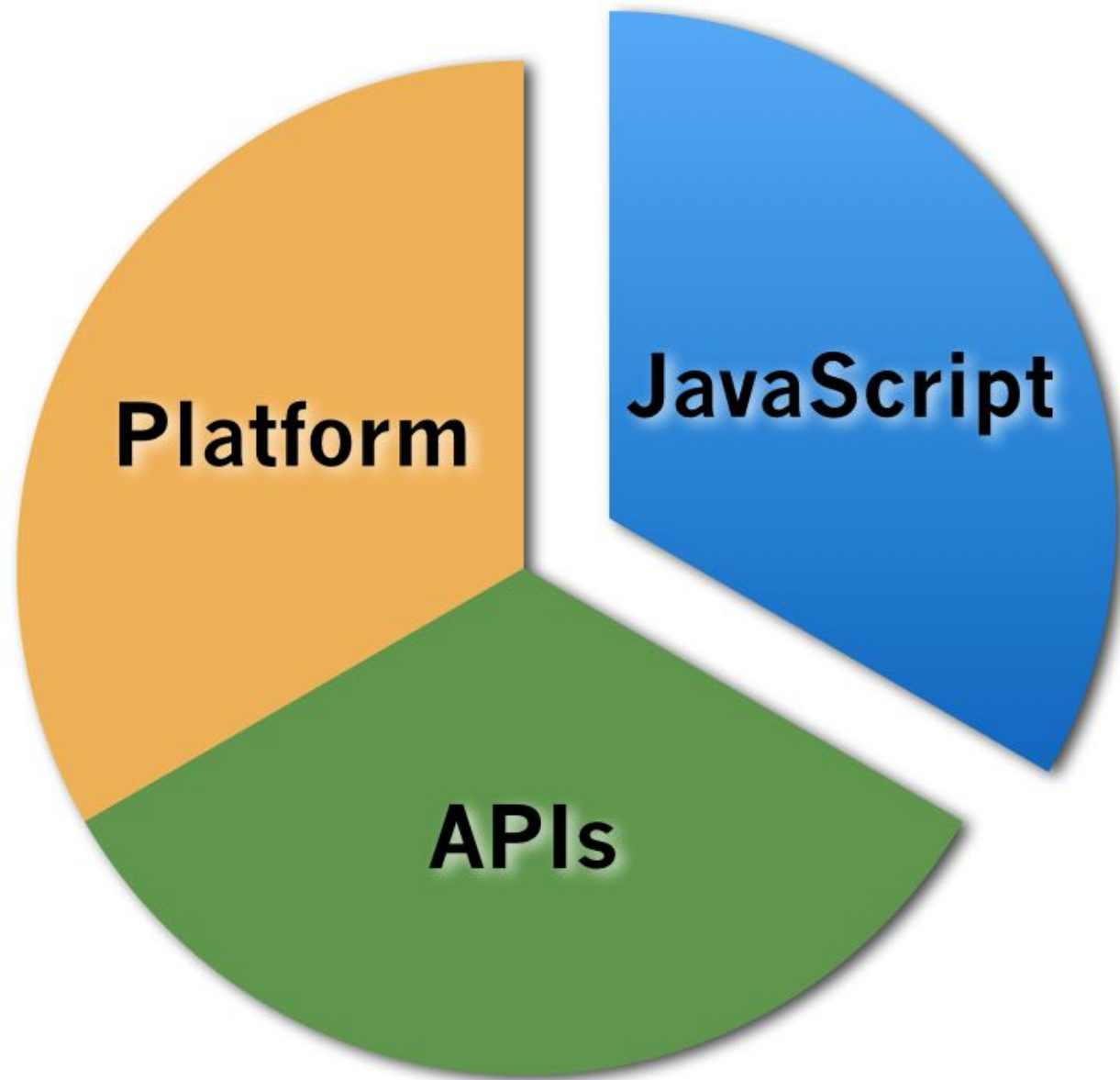
...but I thought that JavaScript was for
DOM manipulation and **browser events**.



JavaScript







7.6.1.1 Keywords

The following tokens are ECMAScript keywords and may not be used as *Identifiers* in ECMAScript programs.

Syntax

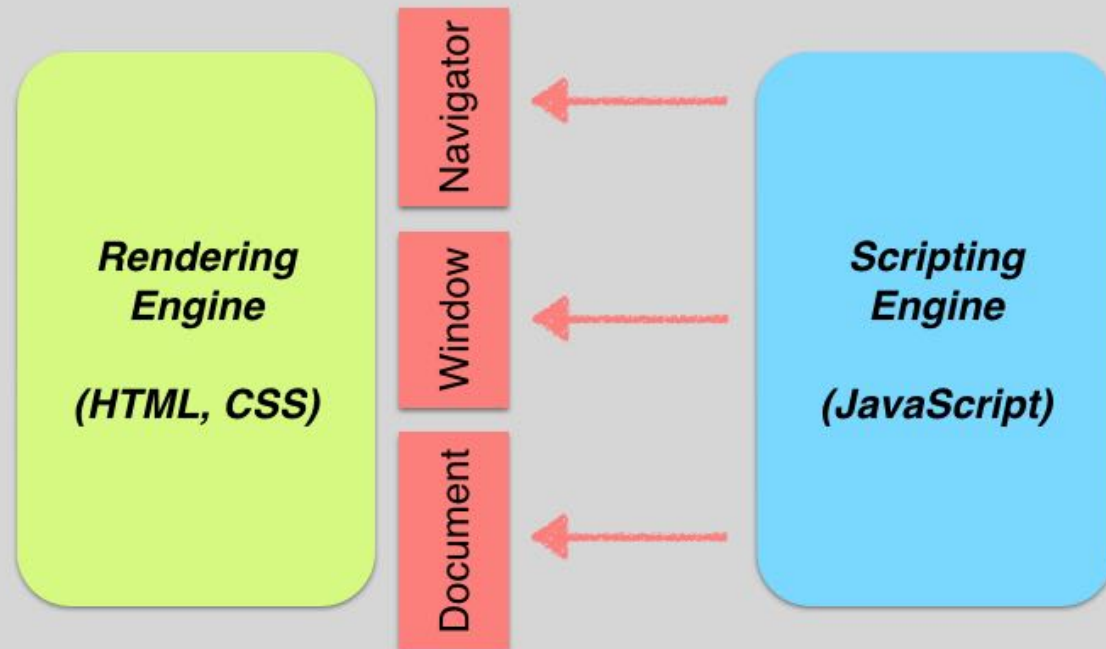
Keyword :: **one of**

break	do	instanceof	typeof
case	else	new	var
catch	finally	return	void
continue	for	switch	while
debugger	function	this	with
default	if	throw	
delete	in	try	

When most people think of **JavaScript**, they're most likely conflating it with a **browser-provided API**

- Navigator / Window
- AJAX / XHR
- `setTimeout()`
- DOM \rightsquigarrow `Document.getElementById()`

Browser





Misguided JavaScript Hate


Hate the **browser**,
not the **language**.

(e.g. render kit API
incompatibilities
among vendors)

W JavaScript - Wikipedia, the x Scott

en.wikipedia.org/wiki/JavaScript

Create account Log in



WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Featured content

Article Talk Read Edit View history Search

JavaScript

From Wikipedia, the free encyclopedia
(Redirected from Javascript)

Not to be confused with Java (programming language), Java (software platform), or Javanese script.

Elements Network Sources Timeline Profiles >>

<top frame> Preserve log

```
> navigator.appName
< "Netscape"
> navigator.appVersion
< "5.0 (Macintosh; Intel Mac OS X 10_10_2) AppleWebKit/537.36 (KHTML,
  like Gecko) Chrome/41.0.2272.76 Safari/537.36"
> |
```

Console Search Emulation Rendering

navigator-test.js

```
console.log(navigator.appName);  
console.log(navigator.appVersion);
```

Run it:

```
$ node navigator-test.js  
  
console.log(navigator.  
             ^  
ReferenceError: navigator is not defined  
    at Object.anonymous (~/navigator-test.js:1:75)  
    at Module._compile (module.js:460:26)  
    at Object.Module._extensions..js (module.js:478:10)  
    at Module.load (module.js:355:32)  
    at Function.Module._load (module.js:310:12)  
    at Function.Module.runMain (module.js:501:10)  
    at startup (node.js:129:16)  
    at node.js:814:3
```


Navigator

The **Navigator** interface represents the state and the identity of the user agent. It allows scripts to query it and to register themselves to carry on some activities.

A Navigator object can be retrieved using the read-only [Window.navigator](#) property.


Standard

[NavigatorID.appName](#)

Read only 

Returns a [DOMString](#) with the official name of the browser. Do not rely on this property to return the correct value.

[NavigatorID.appVersion](#)

Read only 

Returns the version of the browser as a [DOMString](#). Do not rely on this property to return the correct value.

Node.js v0.12.0 Manual & Documentation

[Index](#) | [View on single page](#) | [View as JSON](#)

Table of Contents

[About these Docs](#)

[Synopsis](#)

[Assertion Testing](#)

[Buffer](#)

[C/C++ Addons](#)

[Child Processes](#)

[Cluster](#)

[Console](#)

[Crypto](#)

[Debugger](#)

[DNS](#)

[Domain](#)

[Events](#)

[File System](#)

[Globals](#)

[HTTP](#)

[HTTPS](#)

[Modules](#)

[Net](#)

[OS](#)

[Path](#)

[Process](#)

[Punycode](#)

[Query Strings](#)

[Readline](#)

[REPL](#)

[Smalloc](#)

[Stream](#)

[String Decoder](#)

[Timers](#)

[TLS/SSL](#)

[TTY](#)

[UDP/Datagram](#)

[URL](#)

[Utilities](#)

[VM](#)

[ZLIB](#)

OS

Stability: 4 - API Frozen

Provides a few basic operating-system related utility functions.

Use `require('os')` to access this module.

`os.hostname()`

Returns the hostname of the operating system.

`os.type()`

Returns the operating system name.

`os.platform()`

Returns the operating system platform.

`os.arch()`

Returns the operating system CPU architecture. Possible values are `"x64"`, `"arm"` and `"ia32"`.

os-test.js

```
var os = require('os');  
  
console.log(os.hostname());  
console.log(os.type());  
console.log(os.platform());  
console.log(os.arch());
```

Run it:

```
$ node os-test.js  
  
fift33n.local  
Darwin  
darwin  
x64
```



Point #5

Don't hate JavaScript,
hate incompatible render kit
vendor APIs.

When you're running
JavaScript in all tiers
of your app
(Server, DB, Client),
you have to know which APIs
are available to you.

ExpressJS === Web Server

http://expressjs.com/

Express

Home

Getting started

Guide

API reference

Advanced topics

Resources

Express and Node.js Training
from StrongLoop

Express

Fast, unopinionated, minimalist
web framework for **Node.js**

```
$ npm install express --save
```

Web Applications

Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.

APIs

With a myriad of HTTP utility methods and middleware at your disposal, creating a robust API is quick and easy.

Performance

Express provides a thin layer of fundamental web application features, without obscuring Node features that you know and love.

http://nodejs.org



[HOME](#) | [DOWNLOADS](#) | [DOCS](#) | [COMMUNITY](#) | [ABOUT](#) | [JOBS](#) | [BLOG](#)

Node.js® is a platform built on **Chrome's JavaScript runtime** for easily building fast, 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.

Current Version: v0.12.0

[INSTALL](#)

[DOWNLOADS](#)

[API DOCS](#)

AN EXAMPLE: WEBSERVER

This simple web server written in Node responds with "Hello World" for every request.

```
var http = require('http');
http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/plain'});
  res.end('Hello World\n');
}).listen(1337, '127.0.0.1');
console.log('Server running at http://127.0.0.1:1337/');
```

To run the server, put the code into a file `example.js` and execute it with the `node` program from the command line:

```
% node example.js
Server running at http://127.0.0.1:1337/
```

NodeJS vs. Browser

There are two important things to notice in the next example:

- **Modules**
- **Asynchronous** Server Events


```
'use strict';
var http = require('http');
var url = require('url');
var PORT = process.env.PORT || 8888;
var HOSTNAME = process.env.HOSTNAME || 'localhost';
var server;

// override body with http://localhost?msg=Hola+Mundo
function requestHandler(req, res){
    var parsedUrl = url.parse(req.url, true);
    var msg = parsedUrl.query.msg || 'Hello World';

    res.writeHead(200, {'Content-Type': 'text/html'});
    res.end('<h1>' + msg + '</h1>');
}

function startupHandler(){
    var serverAddr = server.address();
    var outputMessage = 'Server running at ';
    outputMessage += 'http://' + serverAddr.address;
    outputMessage += ':' + serverAddr.port;
    console.log(outputMessage);
}

server = http.createServer(requestHandler);
server.listen(PORT, HOSTNAME, null, startupHandler);
```

Browser

Script:

```
<script src="angular.min.js">
```

Client-side Events:

```
var timeoutID;

function slowAlert(){
  window.alert("That was slow!");
}

timeoutID =
  window.setTimeout(slowAlert, 2000);
```

NodeJS

Module:

```
var express = require('express');
```

Server-side Events:

```
var server;


function requestHandler(req, res){
  res.writeHead(200, {...});
  res.end('<h1>Howdy</h1>');
}


server =
  http.createServer(requestHandler);
```

NodeJS !== Web Server

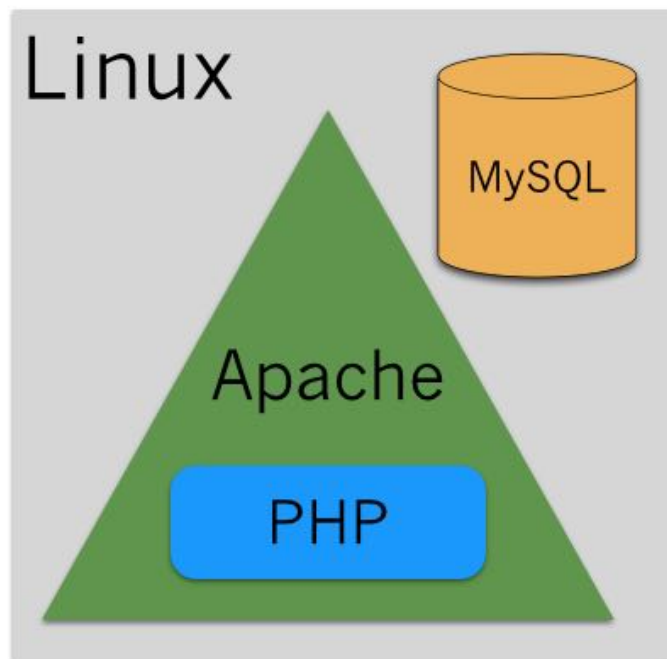
NodeJS is simply a platform that includes modules for networking like `net`, `http`, `https`, and `udp`.

ExpressJS is a **third-party module** that extends the core capabilities of `http`.

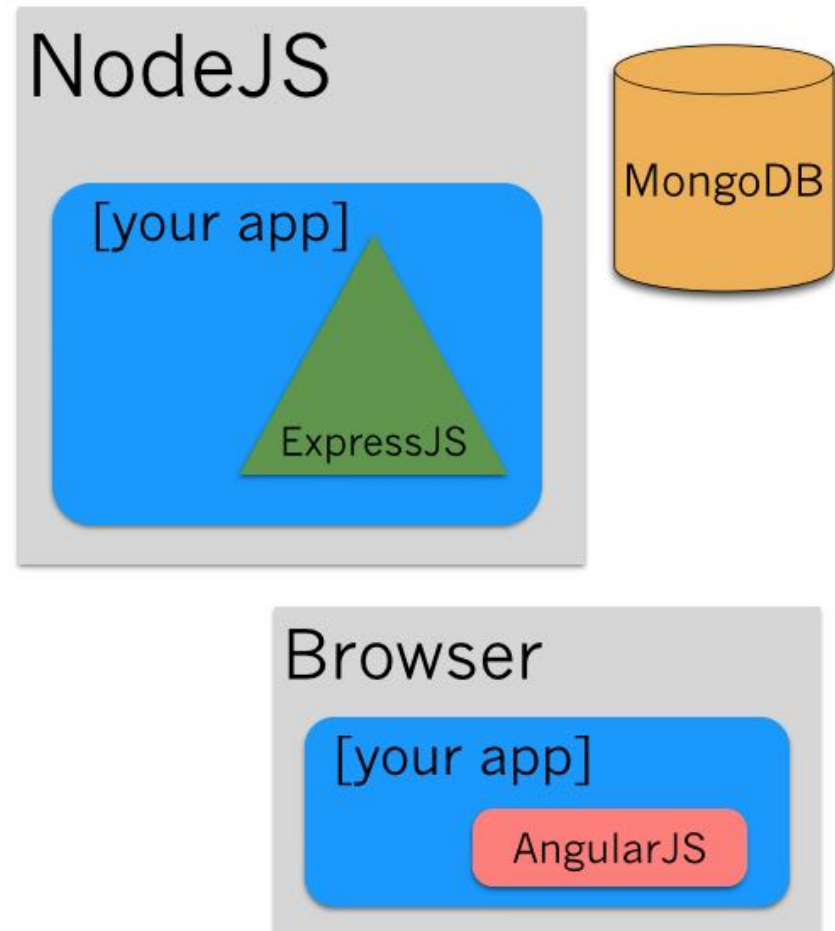
Linux  **NodeJS**

Apache  **ExpressJS**

LAMP



MEAN



Before we require ExpressJS in our app,
there are three key concepts we need to explore:

- **CommonJS Modules** [`require`]
- CommonJS Packages [`package.json`]
- npm

CommonJS

From Wikipedia, the free encyclopedia

CommonJS is a project with the goal of specifying an ecosystem for **JavaScript** outside the browser (for example, on the **server** or for native desktop applications).

History [\[edit\]](#)

The project was started by **Mozilla** engineer Kevin Dangoor in January 2009 and initially named **ServerJS**.^[1]

“ What I’m describing here is not a technical problem. It’s a matter of people getting together and making a decision to step forward and start building up something bigger and cooler together. ”

— Kevin Dangoor^[1]

In August 2009, the project was renamed *CommonJS* to show the broader applicability of the **APIs**.^[2] Specifications are created and approved in an open process. A specification is only considered *final* after it has been finished by multiple implementations.^[3] CommonJS is not affiliated with the **Ecma International** group TC39 working on **ECMAScript**, but some members of TC39 participate in the project.^[4]

Specifications [\[edit\]](#)

The list of specifications includes:^[6]

Current [\[edit\]](#)

- Modules/1.0 (Superseded by Modules/1.1)
- Modules/1.1
- Modules/1.1.1
- Packages/1.0
- Promises/B
- Promises/C
- System/1.0

Proposals [\[edit\]](#)

- Binary/B
- Binary/F
- Console
- Encodings/A
- Filesystem/A
- Filesystem/A/0
- Modules/Async/A
- Modules/Transport/B
- Packages/1.1
- Packages/Mappings
- Unit Testing/1.0

Implementations [\[edit\]](#)

- [Akshell](#)^[7]
- [Common Node](#)^[8]
- [CommonJS Compiler](#) - a command-line tool that makes Common JS modules suitable for in-browser use ^[9]
- [CommonJS for PHP](#) - a light-weight CommonJS implementation for PHP 5.3+ ^[10]
- [CouchDB](#)^[11]
- [Flusspferd](#)^[12]
- [GPSEE](#)^[13]
- [Jetpack](#)
- [Joyent Smart Platform](#)^[14]
- [JSBuild](#) ^[15]
- [MongoDB](#)^[16]
- [Narwhal \(JavaScript platform\)](#)^[17]
- [node.js](#)^[18]
- [Persevere](#)^[19]
- [PINF JavaScript Loader](#) ^[20]
- [RingoJS](#)^[21]
- [SilkJS](#)^[22]
- [SproutCore](#)^[23]
- [TeaJS](#)^[24]
- [Wakanda](#)^[25]
- [XULJet](#) ^[26]



branch: **master** ▾

express / **lib** / +

Fix regression where "Request aborted" is logged using res.sendFile ...



dougwilson authored 6 days ago

latest comm:

..

middleware	Fix typo in comment
router	Deprecate app.param(fn)
application.js	Merge tag '3.20.0'
express.js	Fix constructing application with non-configurable prototype properties
request.js	Merge tag '3.20.1'
response.js	Fix regression where "Request aborted" is logged using res.sendFile
utils.js	Merge tag '3.20.0'
view.js	Add support for app.set('views', array)

branch: master ▾

express / lib / express.js



dougwilson 7 days ago Fix constructing application with non-configurable prototype properties

12 contributors



94 lines (78 sloc) | 1.773 kb

Raw Blame Histc

```
1  /**
2   * Module dependencies.
3   */
4
5  var EventEmitter = require('events').EventEmitter;
6  var mixin = require('merge-descriptors');
7  var proto = require('./application');
8  var Route = require('./router/route');
9  var Router = require('./router');
10 var req = require('./request');
11 var res = require('./response');
12
13 /**
14  * Expose `createApplication()`.
15  */
16
17 exports = module.exports = createApplication;
18
19 /**
20  * Create an express application.
21  *
22  * @return {Function}
23  * @api public
24  */
25
```

CommonJS Modules

Key Points:

- `require` in other dependent modules
- `module.exports` === public API
- variables and functions not in `module.exports` are `private`

Before we require ExpressJS in our app,
there are three key concepts we need to explore:

- ✓ CommonJS Modules `[require]`
- CommonJS Packages `[package.json]`
- npm

CommonJS Packages


Packages

This specification describes the CommonJS package format for distributing CommonJS programs and libraries. A CommonJS package is a cohesive wrapping of a collection of modules, code and other assets into a single form. It provides the basis for convenient delivery, installation and management of CommonJS components.

This specifies the CommonJS package descriptor file and package file format. It does not specify a package catalogue file or format; this is an exercise for future specifications. The package descriptor file is a statement of known fact at the time the package is published and may not be modified without publishing a new release.

Package Descriptor File

Each package must provide a top-level package descriptor file called "package.json". This file is a JSON format file. Each package must provide all the following fields in its package descriptor file.

- name - the name of the package. This must be a unique, lowercase alpha-numeric name without spaces. It may include "." or "_" or "-" characters. It is otherwise opaque.
- version - a version string conforming to the Semantic Versioning requirements (<http://semver.org/> .

One of the following must also be in the package description file in order for it to be valid.

- main - module that must be loaded when `require(name)` is called. Definition must be relative to the package description file.
- directories.lib - directory of modules to be loaded under the packages namespace. `require(name/subfilename)` must return modules from this directory. Definition must be relative to the package description file.



package.json

AN INTERACTIVE GUIDE

This is an interactive guide for exploring various important properties of the [package.json](#) packaging format for [node.js](#) applications.

You can access information about properties by **mousing over** or **clicking** the property name.

```
{
  "name": "module-name",
  "version": "10.3.1",
  "description": "An example module to illustrate the usage of .",
  "author": "Your Name <you.name@example.org>", "main": "lib/foo",
  "repository": {
    "type": "git",
    "url": "https://github.com/nodejitsu/browsenpm.org"
  },

  "dependencies": {
    "primus": "*",
    "async": "~0.8.0",
    "express": "4.2.x",
    "winston": "git://github.com/flatiron/winston#master",
    "bigpipe": "bigpipe/pagelet",
    "plates": "https://github.com/flatiron/plates/tarball/maste
  },
  "devDependencies": {
    "vows": "^0.7.0",
    "assume": "<1.0.0 || >=2.3.1 <2.4.5 || >=2.5.2 <3.0.0"
```

version

The version of the package is specified by **Semantic Versioning**. Which assumes that a version number is written as **MAJOR.MINOR.PATCH** and you increment the:

1. **MAJOR** version when you make incompatible API changes
2. **MINOR** version when you add functionality in a backwards-compatible manner
3. **PATCH** version when you make backwards-compatible bug fixes

Fast, unopinionated, minimalist web framework for node. <http://expressjs.com>


 5,071 commits

 13 branches

 246 releases

 172 contributors



 branch: **master** ▾

express / +
















4.12.2



dougwilson authored 6 days ago

latest commit [dee9fbbda](#) 

 benchmarks	use 8 threads for benchmarks	a year ago
 examples	examples: fixes to mvc example	18 days ago
 lib	Fix regression where "Request aborted" is logged using res.sendFile	6 days ago
 test	Fix wrong code on aborted connections from res.sendFile	7 days ago
 .gitignore	build: misc. updates to packaging	5 months ago
 .travis.yml	Merge tag '3.20.1'	8 days ago
 Contributing.md	Merge tag '3.16.7'	7 months ago
 History.md	4.12.2	6 days ago
 LICENSE	Merge tag '3.20.0'	18 days ago
 Readme.md	build: add AppVeyor	18 days ago
 appveyor.yml	build: skip unnecessary dependency in AppVeyor	8 days ago
 index.js	build: remove lib-cov fork	10 months ago
 package.json	4.12.2	6 days ago

branch: master ▾

express / package.json



dougwilson 6 days ago 4.12.2

19 contributors



90 lines (89 sloc) 2.55 kb

Raw

Blame

His

```
1 {
2   "name": "express",
3   "description": "Fast, unopinionated, minimalist web framework",
4   "version": "4.12.2",
5   "author": "TJ Holowaychuk <tj@vision-media.ca>",
6   "contributors": [
7     "Aaron Heckmann <aaron.heckmann+github@gmail.com>",
8     "Ciaran Jessup <ciaranj@gmail.com>",
9     "Douglas Christopher Wilson <doug@somethingdoug.com>",
10    "Guillermo Rauch <rauchg@gmail.com>",
11    "Jonathan Ong <me@jongleberry.com>",
12    "Roman Shtylman <shtylman+expressjs@gmail.com>",
13    "Young Jae Sim <hanul@hanul.me>"
14  ],
15  "license": "MIT",
16  "repository": "strongloop/express",
17  "homepage": "http://expressjs.com/",
18  "keywords": [
19    "express",
20    "framework",
21    "sinatra",
22    "web",
23    "rest",
24    "restful",
25    "router"
```


branch: master

angular.js / package.json



juliemr 12 days ago chore(ci): update Karma to 0.12.32-beta.0

24 contributors



79 lines (78 sloc) | 2.214 kb

Raw

Blame

History

```
1 {
2   "name": "angularjs",
3   "branchVersion": "^1.4.0-beta.0",
4   "branchPattern": "1.4.*",
5   "repository": {
6     "type": "git",
7     "url": "https://github.com/angular/angular.js.git"
8   },
9   "engines": {
10    "node": "~0.10",
11    "npm": "~2.5"
12  },
13  "engineStrict": true,
14  "devDependencies": {
15    "angular-benchpress": "0.x.x",
16    "benchmark": "1.x.x",
17    "bower": "~1.3.9",
18    "browserstacktunnel-wrapper": "~1.3.1",
19    "canonical-path": "0.0.2",
20    "cheerio": "^0.17.0",
21    "dgeni": "^0.4.0",
22    "dgeni-packages": "^0.10.0",
23    "event-stream": "~3.1.0",
24    "grunt": "~0.4.2",
25    "grunt-bump": "~0.0.13",
```


<http://semver.org/>

Before we require ExpressJS in our app,
there are three key concepts we need to explore:

- ✓ CommonJS Modules [require]
- ✓ CommonJS Packages [package.json]
- npm

<https://www.npmjs.com/>



find packages



sign up or log in



★ express

Fast, unopinionated, minimalist web framework

express

npm v4.12.2 downloads 2M/month linux passing windows passing coverage 100%

```
var express = require('express')
var app = express()

app.get('/', function (req, res) {
  res.send('Hello World')
})

app.listen(3000)
```

Installation

```
$ npm install express
```

Features

- Robust routing
- Focus on high performance
- Super-high test coverage
- HTTP helpers (redirection, caching, etc)
- View system supporting 14+ template engines
- Content negotiation
- Executable for generating applications quickly

npm install express

dougwilson published 6 days ago

4.12.2 is the latest of 230 releases

github.com/strongloop/express

expressjs.com

MIT license

Maintainers



Stats

38,608 downloads in the last day

441,396 downloads in the last week

2,093,360 downloads in the last month

47 open issues on GitHub

11 open pull requests on GitHub

Keywords

express, framework, sinatra, web, rest, restful, router, app, api

Dependencies (24)

accepts, content-disposition, content-type, cookie-signature, debug, depd, escape-html, etag, finalhandler, fresh, merge-descriptors, methods, on-finished, parseurl, path-to-regexp, proxy-addr,

Before we require ExpressJS in our app,
there are three key concepts we need to explore:

- ✓ CommonJS Modules [require]
- ✓ CommonJS Packages [package.json]
- ✓ npm

Ready to install **ExpressJS**?

Installing

First, create a directory to hold your application, if you haven't already done so, and make that your working directory.

```
$ mkdir myapp  
$ cd myapp
```

Create a `package.json` file in the directory of interest, if it does not exist already, with the `npm init` command.

```
$ npm init
```

Install Express in the app directory and save it in the dependencies list:

```
$ npm install express --save
```

To install Express temporarily, and not add it to the dependencies list, omit the `--save` option:

```
$ npm install express
```

Node modules installed with the `--save` option are added to the `dependencies` list in the `package.json` file. Then using `npm install` in the app directory will automatically install modules in the dependencies list.

npm init

```
$ npm init
```

```
This utility will walk you through creating a package.json file.  
It only covers the most common items, and tries to guess sane defaults.
```

```
See `npm help json` for definitive documentation on these fields  
and exactly what they do.
```

```
Use `npm install <pkg> --save` afterwards to install a package and  
save it as a dependency in the package.json file.
```

```
Press ^C at any time to quit.
```

```
name: (example)
```

```
version: (1.0.0) 0.1.0
```

```
description: An example NodeJS app
```

```
entry point: (index.js) server.js
```

```
test command:
```

```
git repository:
```

```
keywords:
```

```
author: Scott Davis <scott@thirstyhead.com>
```

```
license: (ISC) BSD
```

```
About to write to ~/example/package.json:
```

package.json

```
{
  "name": "example",
  "version": "0.1.0",
  "description": "An example NodeJS app",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Scott Davis <scott@thirstyhead.com>",
  "license": "BSD"
}
```


npm install express --save

```
$ npm install express --save

express@4.12.2 node_modules/express
├─ merge-descriptors@1.0.0
├─ utils-merge@1.0.0
├─ methods@1.1.1
├─ cookie-signature@1.0.6
├─ cookie@0.1.2
├─ fresh@0.2.4
├─ escape-html@1.0.1
├─ range-parser@1.0.2
├─ content-type@1.0.1
├─ finalhandler@0.3.3
├─ vary@1.0.0
├─ parseurl@1.3.0
├─ serve-static@1.9.1
├─ content-disposition@0.5.0
├─ path-to-regexp@0.1.3
├─ depd@1.0.0
├─ qs@2.3.3
├─ on-finished@2.2.0 (ee-first@1.1.0)
├─ debug@2.1.2 (ms@0.7.0)
├─ etag@1.5.1 (crc@3.2.1)
├─ send@0.12.1 (destroy@1.0.3, ms@0.7.0, mime@1.3.4)
├─ proxy-addr@1.0.6 (forwarded@0.1.0, ipaddr.js@0.1.8)
├─ accepts@1.2.4 (negotiator@0.5.1, mime-types@2.0.9)
└─ type-is@1.6.0 (media-type@0.3.0, mime-types@2.0.9)
```

package.json (after)

```
{
  "name": "example",
  "version": "0.1.0",
  "description": "An example NodeJS app",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Scott Davis <scott@thirstyhead.com>",
  "license": "BSD",
  "dependencies": {
    "express": "^4.12.2"
  }
}
```

node_modules

Everything you `npm install --save` ends up in `node_modules` and `package.json`.

Be sure to add `node_modules` to your `.gitignore` file.

With an up-to-date `package.json`,
typing `npm install`
will re-download all dependencies.

server.js

```
'use strict';  
  
var express = require('express');  
var app = express();  
  
app.get('/', function (req, res) {  
  res.send('<h1>Hello Express</h1>');  
});  
  
app.listen(3000);
```

Your app

```
$ tree -L 2
.
├── node_modules
│   └── express
├── package.json
└── server.js
```

Start your app

```
$ npm start
```


ExpressJS gives our MEAN app **two important things**:

- Middleware
- Routing



Using middleware

An Express application is essentially a series of middleware calls.

Middleware is a function with access to the request object (`req`), the response object (`res`), and the next middleware in line in the request-response cycle of an Express application, commonly denoted by a variable named `next`.

Middleware can:

- Execute any code.
- Make changes to the request and the response objects.
- End the request-response cycle.
- Call the next middleware in the stack.

If the current middleware does not end the request-response cycle, it must call `next()` to pass control to the next middleware, otherwise the request will be left hanging.

With an optional mount path, middleware can be loaded at the application level or at the router level. Also, a series of middleware functions can be loaded together, creating a sub-stack of the middleware system at a mount point.



server.js

```
'use strict';
var init = require('./config/init')(),
    config = require('./config/config'),
    mongoose = require('mongoose'),
    chalk = require('chalk');

// Bootstrap db connection
var db = mongoose.connect(config.db, function(err) {
  if (err) {
    console.error(chalk.red('Could not connect to MongoDB!'));
    console.log(chalk.red(err));
  }
});

// Init the express application
var app = require('./config/express')(db);

// Start the app by listening on <port>
app.listen(config.port);

// Expose app
exports = module.exports = app;

// Logging initialization
console.log('MEAN.JS application started on port ' + config.port);
```

config/express.js

```
// Initialize express app
var app = express();

// Should be placed before express.static
app.use(compress({
  filter: function(req, res) {
    return (/json|text|javascript|css/).test(
      res.getHeader('Content-Type'));
  },
  level: 9
}));

// Request body parsing middleware should be above methodOverride
app.use(bodyParser.urlencoded({
  extended: true
}));
app.use(bodyParser.json());
app.use(methodOverride());

// CookieParser should be above session
app.use(cookieParser());

// Setting the app router and static folder
app.use(express.static(path.resolve('./public')));
```




config/express.js

```
'use strict';
var fs = require('fs'),
    http = require('http'),
    https = require('https'),
    express = require('express'),
    morgan = require('morgan'),
    bodyParser = require('body-parser'),
    session = require('express-session'),
    compress = require('compression'),
    methodOverride = require('method-override'),
    cookieParser = require('cookie-parser'),
    helmet = require('helmet'),
    passport = require('passport'),
    mongoStore = require('connect-mongo')({
      session: session
    }),
    flash = require('connect-flash'),
    config = require('./config'),
    consolidate = require('consolidate'),
    path = require('path');

module.exports = function(db) {
  // Initialize express app
  var app = express();
```



WIKIPEDIA
The Free Encyclopedia

- [Main page](#)
- [Contents](#)
- [Featured content](#)
- [Current events](#)
- [Random article](#)
- [Donate to Wikipedia](#)
- [Wikimedia Shop](#)

Interaction

- [Help](#)
- [About Wikipedia](#)

[Create account](#) [Log in](#)

[Article](#)

[Talk](#)

[Read](#)

[Edit](#)

[View history](#)



Coupling (computer programming)

From Wikipedia, the free encyclopedia

In **software engineering**, **coupling** is the manner and degree of interdependence between software modules; a measure of how closely connected two routines or modules are;^[1] the strength of the relationships between modules.^[2]

Coupling is usually contrasted with **cohesion**. **Low coupling** often correlates with high cohesion, and vice versa. Low coupling is often a sign of a well-structured **computer system** and a good design, and when combined with high cohesion, supports the general goals of high readability and maintainability.^[*citation needed*]



Routing

Routing refers to the definition of end points (URIs) to an application and how it responds to client requests.

A route is a combination of a URI, a HTTP request method (GET, POST, and so on), and one or more handlers for the endpoint. It takes the following structure `app.METHOD(path, [callback...], callback)`, where `app` is an instance of `express`, `METHOD` is an [HTTP request method](#), `path` is a path on the server, and `callback` is the function executed when the route is matched.

The following is an example of a very basic route.

```
var express = require('express')
var app = express()

// respond with "hello world" when a GET request is made to the homepage
app.get('/', function(req, res) {
  res.send('hello world')
})
```



Response methods

The methods on the response object (`res`) in the following table can send a response to the client and terminate the request response cycle. If none of them is called from a route handler, the client request will be left hanging.

Method	Description
res.download()	Prompt a file to be downloaded.
res.end()	End the response process.
res.json()	Send a JSON response.
res.jsonp()	Send a JSON response with JSONP support.
res.redirect()	Redirect a request.
res.render()	Render a view template.
res.send()	Send a response of various types.
res.sendFile	Send a file as an octet stream.
res.sendStatus()	Set the response status code and send its string representation as the response body.



articles.server.routes.js

```
'use strict';
var users = require('../..//app/controllers/users.server.controller'),
    articles = require('../..//app/controllers/articles.server.controller');

module.exports = function(app) {
  // Article Routes
  app.route('/articles')
    .get(articles.list)
    .post(users.requiresLogin, articles.create);

  app.route('/articles/:articleId')
    .get(articles.read)
    .put(users.requiresLogin,
        articles.hasAuthorization,
        articles.update)
    .delete(users.requiresLogin,
        articles.hasAuthorization,
        articles.delete);

  // Finish by binding the article middleware
  app.param('articleId', articles.articleByID);
};
```

ExpressJS gives our MEAN app **two important things**:

- Middleware
- Routing

A silhouette of a hand pointing towards the text on a sunset background. The hand is on the left side of the image, pointing towards the right. The background is a sunset over a body of water, with the sun low on the horizon, creating a warm orange and yellow glow that reflects on the water. The sky transitions from a deep blue at the top to a lighter blue and then to the orange of the sunset near the horizon.

Point #6

ExpressJS is the web server of the MEAN stack.

It brings highly cohesive, loosely coupled *middleware* and *routing* for your RESTful endpoints.

