Moscow Node.js Meetup

13 июля 2016

22 октября 2015

2016-07-06, Version 6.3.0 (Current), @Fishrock123

Notable changes

- **buffer**: Added `buffer.swap64()` to compliment `swap16()` & `swap32()`. (Zach Bjornson) #7157
- **build**: New `configure` options have been added for building Node.js as a shared library. (Stefan Budeanu) #6994
  - The options are: `--shared`, `--without-v8-platform` & `--without-bundled-v8`.
- **crypto**: Root certificates have been updated. (Ben Noordhuis) #7363
- **debugger**: The server address is now configurable via `--debug=<address>:<port>`. (Ben Noordhuis) #3316
- **npm**: Upgraded npm to v3.10.3 (Kat Marchán) #7515 & (Rebecca Turner) #7410
- **readline**: Added the `prompt` option to the readline constructor. (Evan Lucas) #7125
- **repl / vm**: `sigint` / `ctrl+c` will now break out of infinite loops without stopping the Node.js instance. (Anna Henningsen) #6635
- **src**:
  - Added a `node::FreeEnvironment` public C++ API. (Cheng Zhao) #3098
  - Refactored `require('constants')`, constants are now available directly from their respective modules. (James M Snell) #6534
- **stream**: Improved `readable.read()` performance by up to 70%. (Brian White) #7077
- **timers**: `setImmediate()` is now up to 150% faster in some situations. (Andras) #6436
- **util**: Added a `breakLength` option to `util.inspect()` to control how objects are formatted across lines. (cjihrig) #7499
- **v8-inspector**: Experimental support has been added for debugging Node.js over the inspector protocol. (Ali Ijaz Sheikh) #6792
  - **Note: This feature is *experimental*, and it could be altered or removed.**
  - You can try this feature by running Node.js with the `--inspect` flag.

chromium / v8 / v8.git / **d08c0304c5779223d6c468373af4815ec3ccdb84**

commit d08c0304c5779223d6c468373af4815ec3ccdb84 [log] [tgz]
author caitpotter88 <caitpotter88@gmail.com> Tue May 17 00:26:53 2016
committer Commit bot <commit-bot@chromium.org> Tue May 17 00:27:51 2016
tree 9dfc56e59bace61772f5bd4a6b8b6531db092ae3
parent 5582e158e5d40eb61caf2f8968c37573b802cce0 [diff]

```
[esnext] prototype runtime implementation for async functions

BUG=v8:4483
LOG=N
R=littledan@chromium.org, adamk@chromium.org

Review-Url: https://codereview.chromium.org/1895603002
Cr-Commit-Position: refs/heads/master@{#36263}
```

BUILD.gn [diff]
src/bootstrapper.cc [diff]
src/builtins.cc [diff]
src/builtins.h [diff]
src/code-stubs.h [diff]
src/compiler.cc [diff]
src/contexts.h [diff]
src/factory.cc [diff]
src/globals.h [diff]
src/js/harmony-async-await.js [Added - diff]
src/js/prologue.js [diff]
src/js/promise.js [diff]
src/objects.cc [diff]
src/parsing/parser-base.h [diff]

Title	ES6 Module Interoperability
Author	@bmeck
Status	DRAFT
Date	2016-01-07

NOTE: `DRAFT` status does not mean ES6 modules will be implemented in Node core. Instead that this is the standard, should Node core decide to implement ES6 modules. At which time this draft would be moved to `ACCEPTED`.

The intent of this standard is to:

- implement interoperability for ES modules and Node's existing module system
- create a **Registry Object** (see WHATWG section below) compatible with the WHATWG Loader Registry

1. Purpose

1. Allow a common module syntax for Browser and Server.
2. Allow a common registry for inspection by Browser and Server environments/tools.
    - These will most likely be represented by metaproperties like `import.context`, but the spec is not yet fully in place.

2. Related

ECMA262 discusses the syntax and semantics of related syntax, and introduces:

2.1. Types

- **ModuleRecord**

Benjamin Coe @BenjaminCoe · Jun 4

@CollinEstes I heard a rumor that NASA uses Node.js for space-suits. I'm curious, do you use the npm ecosystem to develop these apps?

64 72

Collin Estes @CollinEstes · Jun 4

@BenjaminCoe You heard correctly, and yes we do.

38 87

Benjamin Coe @BenjaminCoe · Jun 4

@CollinEstes that's amazing, and I'm proud to help people to do such amazing work :)

1 12

Collin Estes
@CollinEstes

Follow

@BenjaminCoe Absolutely, you can say you are helping build the present and future systems supporting spacesuit operations and development.

RETWEETS **47** LIKES **121**

12:19 AM - 4 Jun 2016

47 121


Docker
@docker
Follow
.@docker is back at #GopherCon! The team will be there to run Hack Day on Wed 7/13 from 10a-5p. We'll see you then in Room 4C!
RETWEETS
4
LIKES
12
1:41 AM - 12 Jul 2016
4
12

MongoDB Unveils MongoDB Atlas, the New Industry Standard for Database as a Service

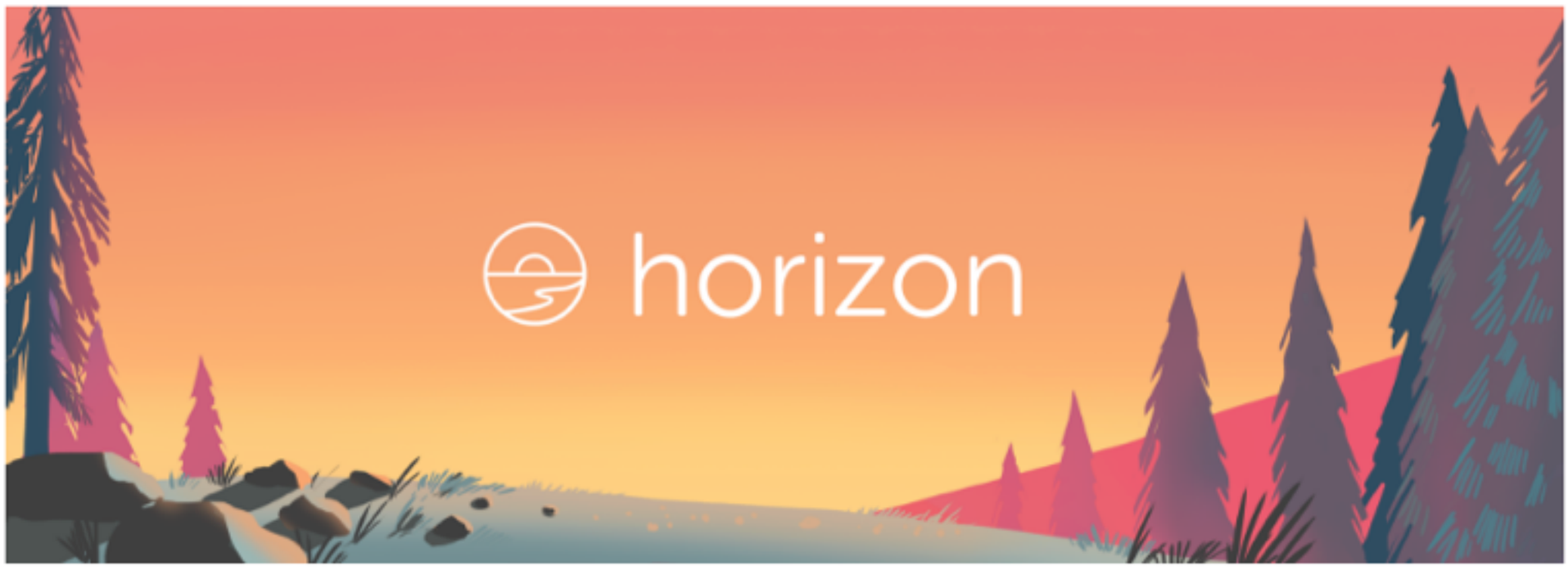
Operated by the Experts Who Design and Engineer MongoDB

Turnkey Service Automates Operations, Dramatically Increasing Developer Productivity

NEW YORK, NY, MongoDB World – June 28, 2016 – MongoDB, the database for giant ideas, today announced MongoDB Atlas, an elastic, on-demand cloud service that includes comprehensive infrastructure and turnkey management of its popular database. MongoDB Atlas will be available to users all over the world, on all popular cloud platforms. The service will initially be available on Amazon Web Services, followed by Microsoft Azure and then Google Cloud Platform. MongoDB has always made it easy to build applications, and now MongoDB Atlas makes it just as easy to run them – at scale, always on. To get access to MongoDB Atlas, visit mongodb.com/atlas.

Introducing Horizon: build realtime apps without writing backend code

Ryan Paul MAY 17, 2016



Today we are pleased to announce the first official release of Horizon, an open-source backend that lets developers build and scale realtime web applications. Horizon includes:

- A **backend server** built with Node.js and RethinkDB that supports data persistence, realtime streams, input validation, user authentication, and permissions
- A **JavaScript client library** that developers can use on the frontend to store JSON documents in the database, perform queries, and subscribe to live updates
- A **command-line tool** that can generate project templates, start up a local Horizon development server, and help you deploy your Horizon application to the cloud

TJ Holowaychuk Follow
Code. Photography. Art.
Jun 16 · 7 min read

Introducing Apex Ping

I'd like to introduce **Apex Ping**! A minimalistic, but powerful uptime & performance monitoring service for websites and APIs. In this post I'll cover features of Apex Ping, why I chose to build it first, and a few things I learned in the process. In future posts I'll be detailing the implementation for anyone who is curious.

I soft-launched Ping about a month ago, built almost 100% on AWS Lambda, with 500 users it has already detected over a million errors, and performed over 25 million checks. Big thanks to the early adopters who have checked it out!

Michael Hart Follow
Founder at SoothBooth, working on Uniqlo Mobile. github.com/mhart
Jul 7 · 8 min read

Introducing LambCI—a serverless build system

I'm excited to announce the first release of LambCI, an open-source continuous integration tool built on AWS Lambda 🎉

LambCI BOT 7:14 PM

Build #15 successful (4 sec)

Repository mhart/test-ci-project

Branch master

LambCI is a tool I began building over a year ago to run tests on our pull requests and branches at Uniqlo Mobile. Inspired at the inaugural ServerlessConf a few weeks ago, I recently put some work into hammering it into shape for public consumption.

SESSION

ECLAIRJS = NODE.JS + APACHE SPARK

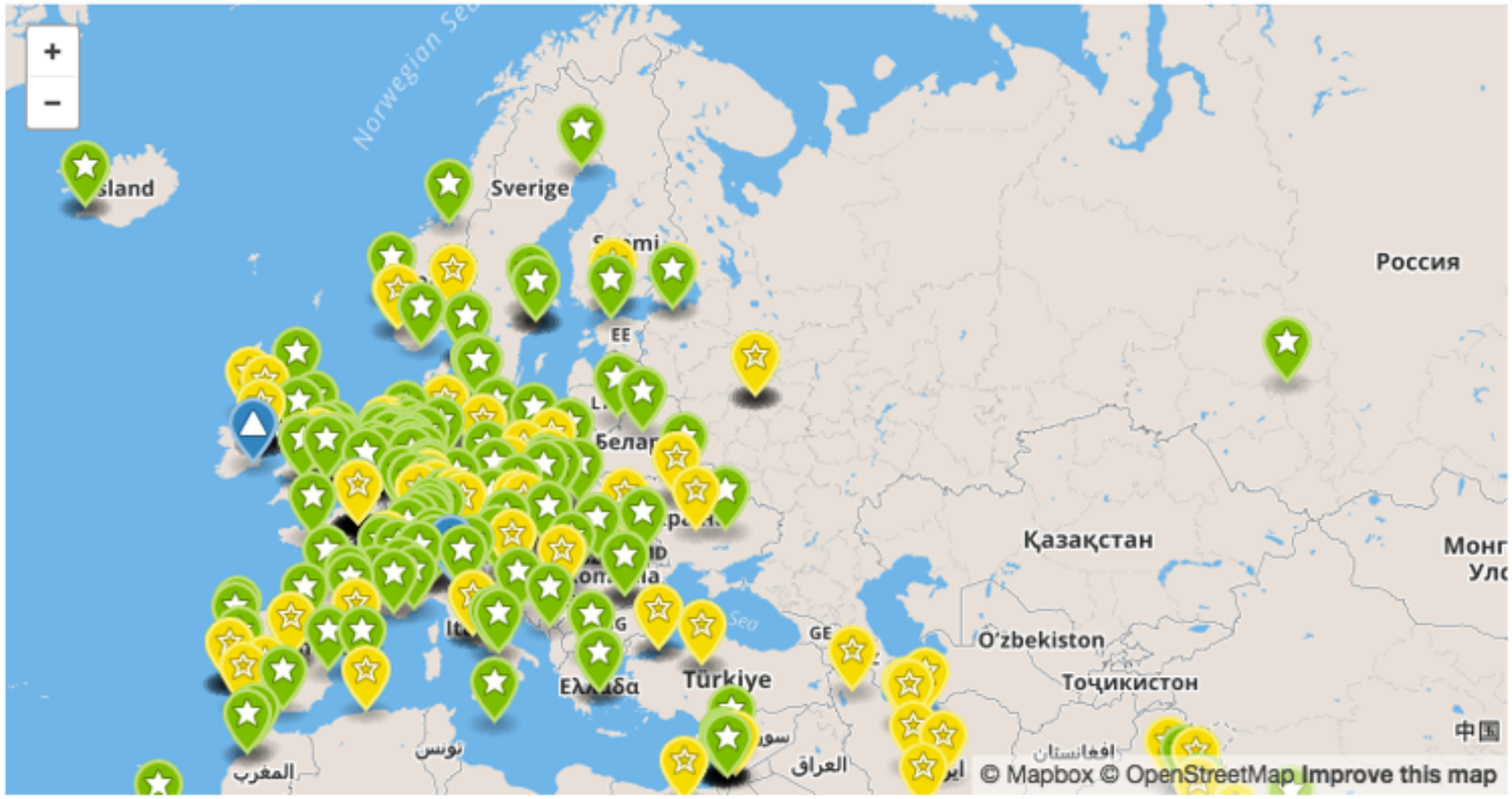
David Fallside (IBM)

Tuesday, June 7
5:25 PM – 5:55 PM
Ballroom B

SLIDES PDF VIDEO

Digital businesses are increasingly looking toward sophisticated analytics to improve their customer interactions, and today's interactive and user-facing applications are developed with technologies such as Node.js, Ruby on Rails, and PHP. Node.js in particular can handle very large numbers of simultaneous requests which contributes to its popularity, however it is not well-suited to the heavier workloads demanded by many analytics and so these workloads are typically handed off to back-end engines. Apache Spark is an ideal engine for this purpose because it is easily scalable, it provides a growing assortment of ML analytics, and within a single processing context it supports both static and streaming data, a SQL database, and a graph engine. Spark has no native API for Node.js or JavaScript and so the EclairJS project was started to provide developers with such an API. EclairJS enables developers to create Node.js applications and to call Spark functions in JavaScript, and to do so in a style that requires no special language constructions. Indeed, developers can write the lambda functions that are passed as arguments to Spark functions such as .map in JavaScript. In this presentation, I'll show an EclairJS streaming and analytical application, and use the code to frame a technical description of EclairJS including coding style and the use of Spark operators and datastructures. In addition, I'll describe the basic architectural components of EclairJS including the EclairJS-Node and EclairJS-Nashorn components and how they relate to Spark.

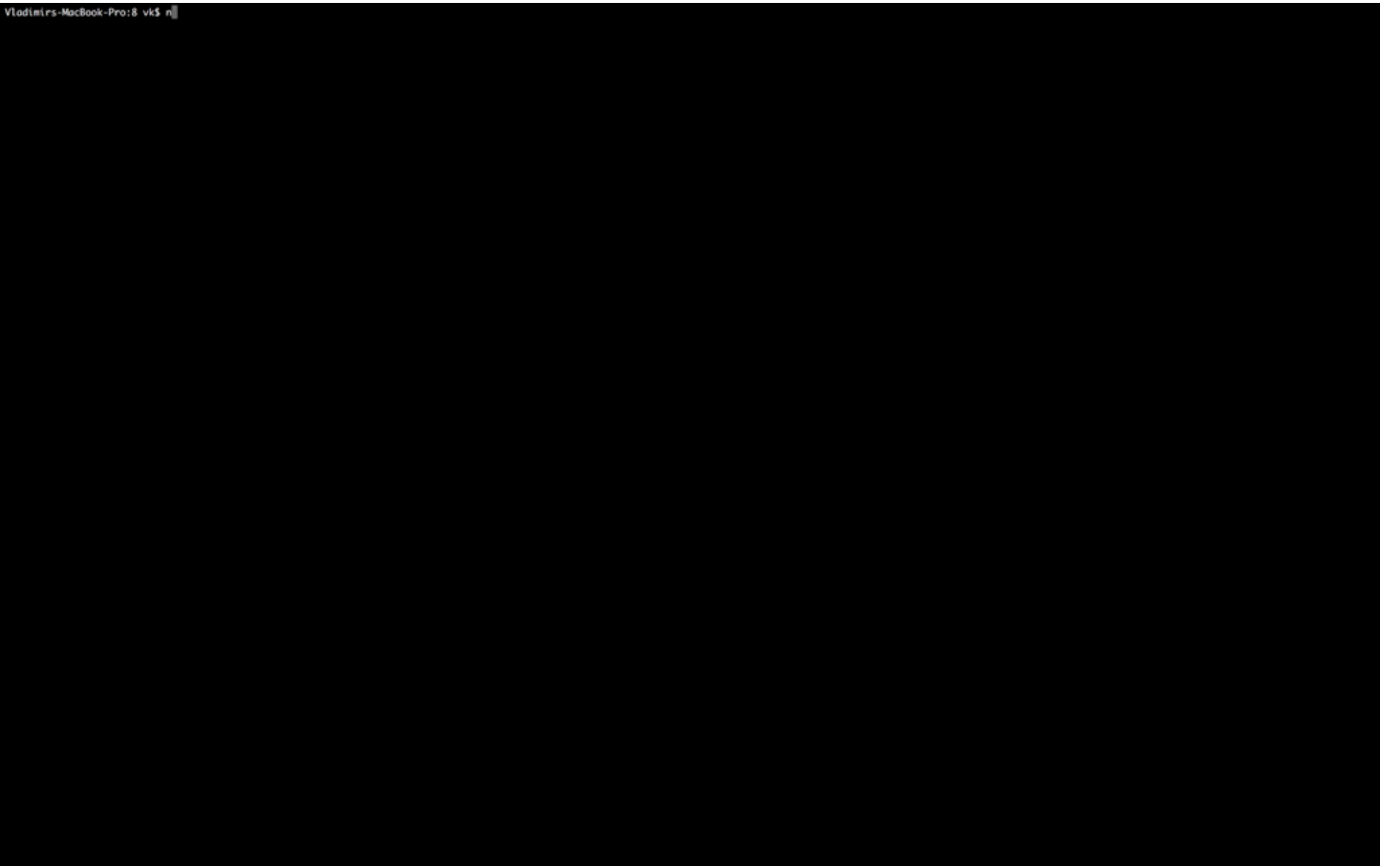
- Get Involved
- Code + Learn
- Contribute
- Development
- **Events**
- Conduct



Meetup (1102) NodeSchool (161) Conference (22)

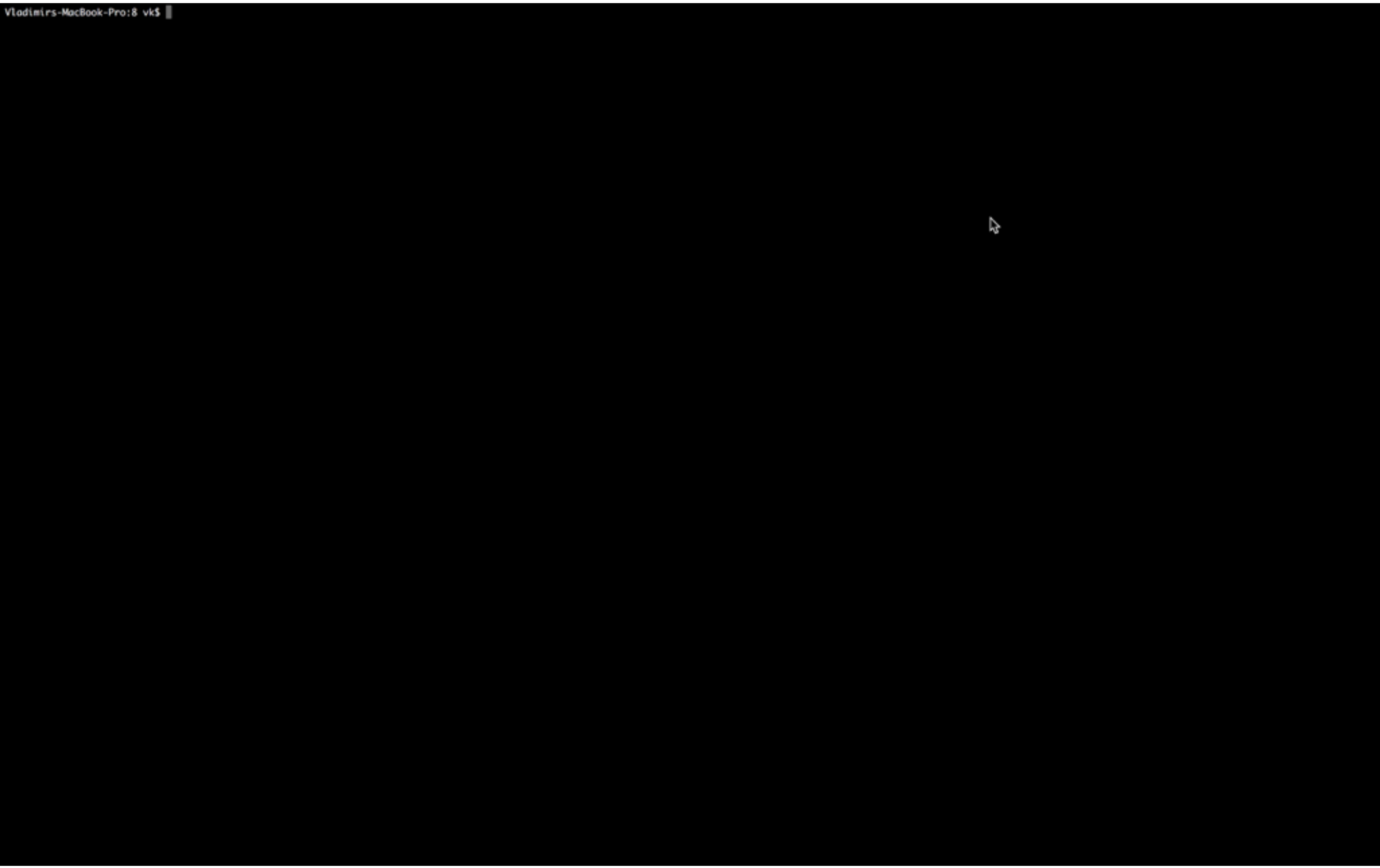
V8 Inspector

V8 Inspector



V8 async-await

V8 async-await



main.js	Initial commit.	13 days ago
package.json	pack and dist task	9 days ago
style.css	Silly font style cross-platform stuff	9 days ago

README.md

Node.js Installer

This is an Electron application for installing, updating and managing versions of Node.js.

The goal is to be far more humane and user friendly than traditional methods of installation.

To Develop

To clone and run this repository you'll need Git and Node.js (which comes with npm) installed on your computer. From your command line:

```
# Clone this repository
git clone https://github.com/nodejs/installer
# Go into the repository
cd installer
# Install dependencies and run the app
npm install && npm start
```

Learn more about Electron and its API in the documentation.

MJS

19:20 - Микросервисная архитектура на базе CoreOS и Kubernetes, Денис Измайлов (Startup Makers, CEO)

19:50 - Deis — облегчённая реализация модели PaaS, Владимир Махаев (Decision Mapper, Developer)

20:05 - Перерыв 15 минут

20:20 - Открытый микрофон

20:30 - Микросервисная архитектура с непрерывной интеграцией — пример из практики, Виталий Аминев (Makeomatic, CTO)

21:00 - Дискуссия

21:30 - After Party в Джон Донн (BeerJS)

Moscow Node.js Meetup

Show must go on!