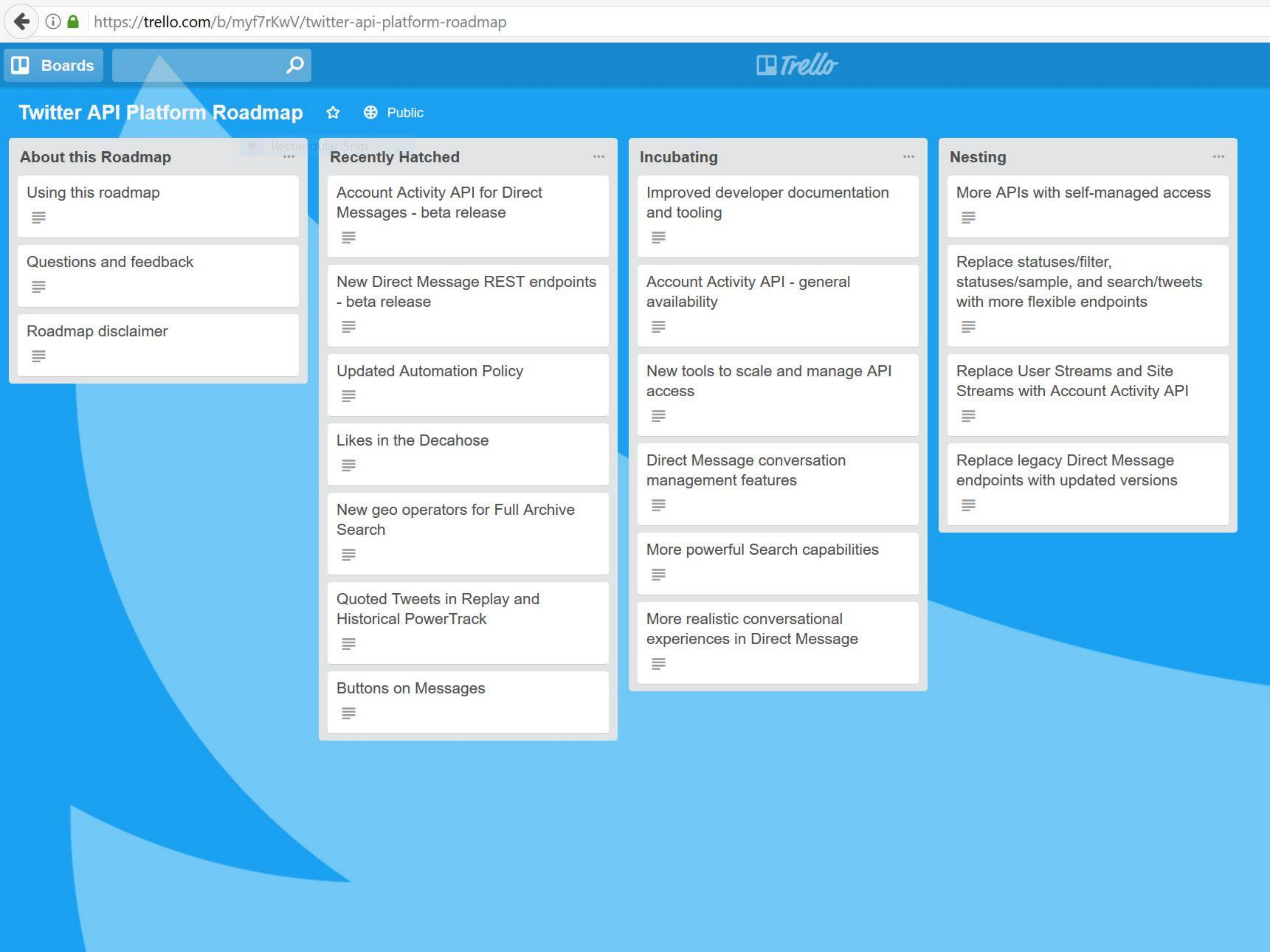
Last week

1. Review
2. Request/response roundtrip
3. RESTful APIs
4. Designing a REST API
5. Assignment 1 (and 2)
6. Workshop

This week

1. Some extras from last week
2. How does an application implement an API?
3. Inside the box – Node.js and ES2015
4. Concurrency (vs Parallelism)
5. State and statelessness
6. Persistence



Twitter API Platform Roadmap

About this Roadmap
- Using this roadmap
- Questions and feedback
- Roadmap disclaimer

Recently Hatched
- Account Activity API for Direct Messages - beta release
- New Direct Message REST endpoints - beta release
- Updated Automation Policy
- Likes in the Decahose
- New geo operators for Full Archive Search
- Quoted Tweets in Replay and Historical PowerTrack
- Buttons on Messages

Incubating
- Improved developer documentation and tooling
- Account Activity API - general availability
- New tools to scale and manage API access
- Direct Message conversation management features
- More powerful Search capabilities
- More realistic conversational experiences in Direct Message

Nesting
- More APIs with self-managed access
- Replace statuses/filter, statuses/sample, and search/tweets with more flexible endpoints
- Replace User Streams and Site Streams with Account Activity API
- Replace legacy Direct Message endpoints with updated versions

W3C Versioning

Consumer backward compatibility

- Consumer upgrades
- Producer downgrades - e.g., rollback

•Consumer forward compatibility

- Producer upgrades (or continue to support earlier versions)
- Support a given range of clients
- Assume don't change semantics, just add or subtract

From client perspective:

1. API is backward compatible if client can continue through service changes
2. Forward compatible if client can be changed without needing service change

2.10. Robustness Principle

TCP implementations should follow a general
principle of robustness: be conservative in what
you do, be liberal in what you accept from others.

http://tools.ietf.org/html/rfc761
1980

MustIgnore pattern
http://www.martinfowler.com/articles/consumerDrivenContracts.html

1. Caching - want different versions cached differently
2. Number or name?
    - If number, what format? Semver? Counter?

Semantic Versioning – semver.org

Given a version number MAJOR.MINOR.PATCH, increment the:

- MAJOR version when you make incompatible API changes,
- MINOR version when you add functionality in a backwards-compatible manner, and
- PATCH version when you make backwards-compatible bug fixes.

"Facebook's Marketing API now supports both versioning and migrations so that app builders can roll out changes over time. Read on to understand how you are affected by versions, how to use those versions in our Marketing APIs and Ads SDKs, and what migration windows are.

While Facebook's Platform has a core and extended versioning model, starting Oct 30th, 2014, Facebook's Marketing API will move to a versioning scheme to manage changes in the Marketing API. With Marketing API versioning, all breaking changes will be rolled up into a new version. Multiple versions of Marketing APIs or Ads SDKs can exist at the same time with different functionality in each version."

https://developers.facebook.com/docs/marketing-api/versions

Three approaches to specifying API version in request

- Query parameter
  - ?v=xx.xx, or ?version=xx.xx or ?Version=2015-10-01
  - e.g., Amazon, NetFlix
- URI
  - /v1/
  - e.g. Facebook
  - https://graph.facebook.com/v2.2/me/adaccounts
  - Semantically messy (implies version refers to version of object)
- Header
  - Accept header - hard to test - can't just click on link or type URL
  - Custom request header - duplicates Accept header function
  - https://developer.github.com/v3/media/
  - https://blog.pivotal.io/labs/labs/api-versioning

```
curl https://api.github.com/users/technoweenie -I \ -H "Accept: application/vnd.github.v3.full+json"
```

JSON “types”

Reuse standard type definitions

Credit cards, people, addresses,…

Microformats/microdata* or schemas

* In HTML markup (or JSON) to add semantics

Credibility

“If a developer doesn’t believe you, then good luck getting them to use your product.”

Support

“Developers are looking for signs of support. Something at some point is going to go wrong, so developers want to know they can solve their problems quickly.”

Success

“Look to share details about how your API can help them achieve something. A lot of companies don’t have that understanding that providing an API is going to create a win for both of you.”

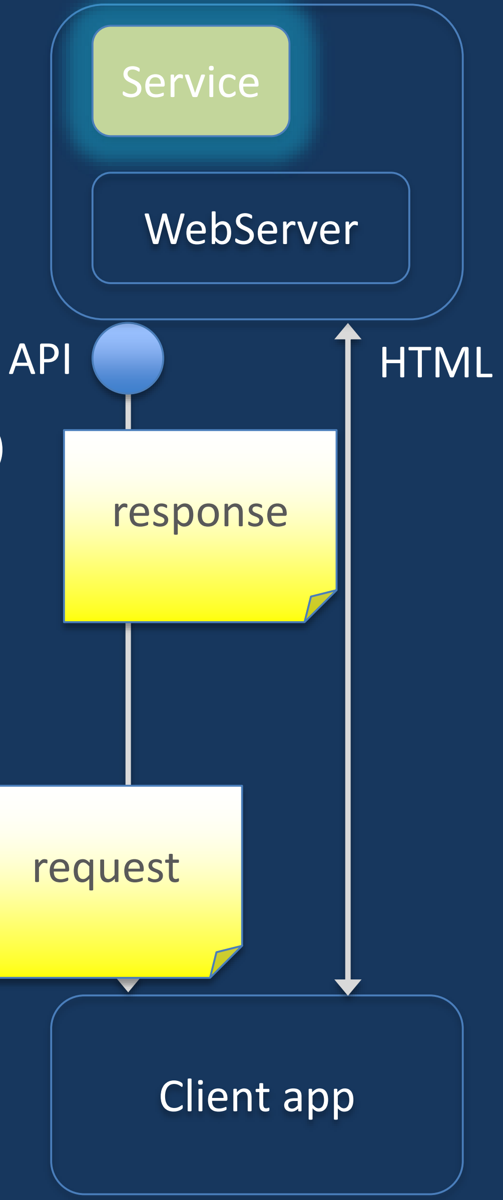
http://www.programmableweb.com/news/how-to-maximize-developer-adoption/analysis/2014/11/17

- Documentation - current, accurate, easy, guide/tutorial/directed (management tool generated)
- Direct access (no SDK required)
  - e.g., through Postman or curl (say, curl -L http://127.0.0.1:4001/v2/keys/message-XPUT -d value="Hello world")
- SDKs/Samples in developer preferred languages
  - Any SDK is just libraries to access REST/SOAP API, nothing more. Potentially an impediment to simply making use of the straight API.
  - Straightforward install and use
- Free/Freemium use for developers
- Instant API keys
- Simple sandbox to try things out for developers

- Before API available, establish API landing page on web to discover interest and potential user types

http://www.cutter.com/content-and-analysis/resource-centers/agile-project-management/sample-our-research/apmu1306.html

- Usage limits
- dev.abc.com or developer.abc.com



HTTP status (e.g., 200, 201, 500)

HTTP headers

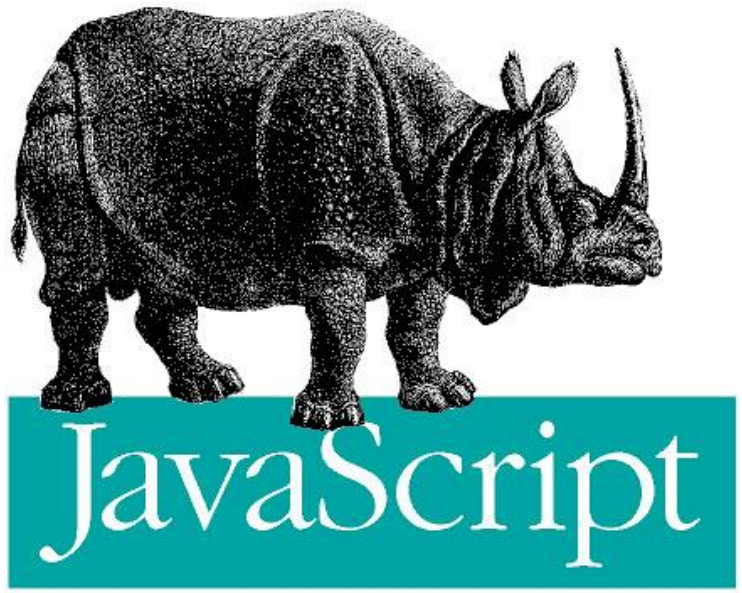
Body

scheme:[//[user[:password]@]host[:port]][/path][?query][#fragment]

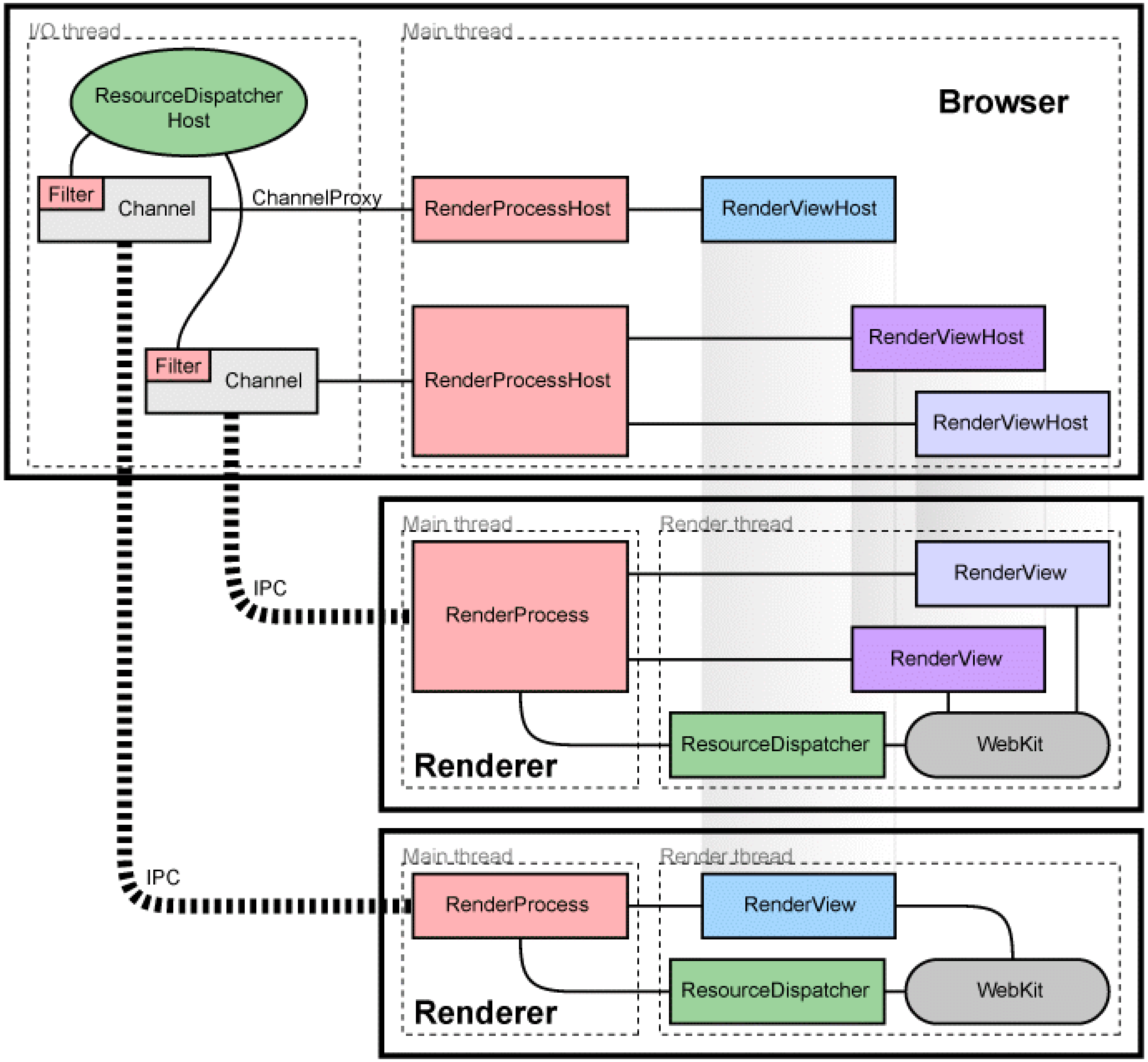
HTTP method (GET/POST/DELETE/PUT/PATCH)

HTTP headers

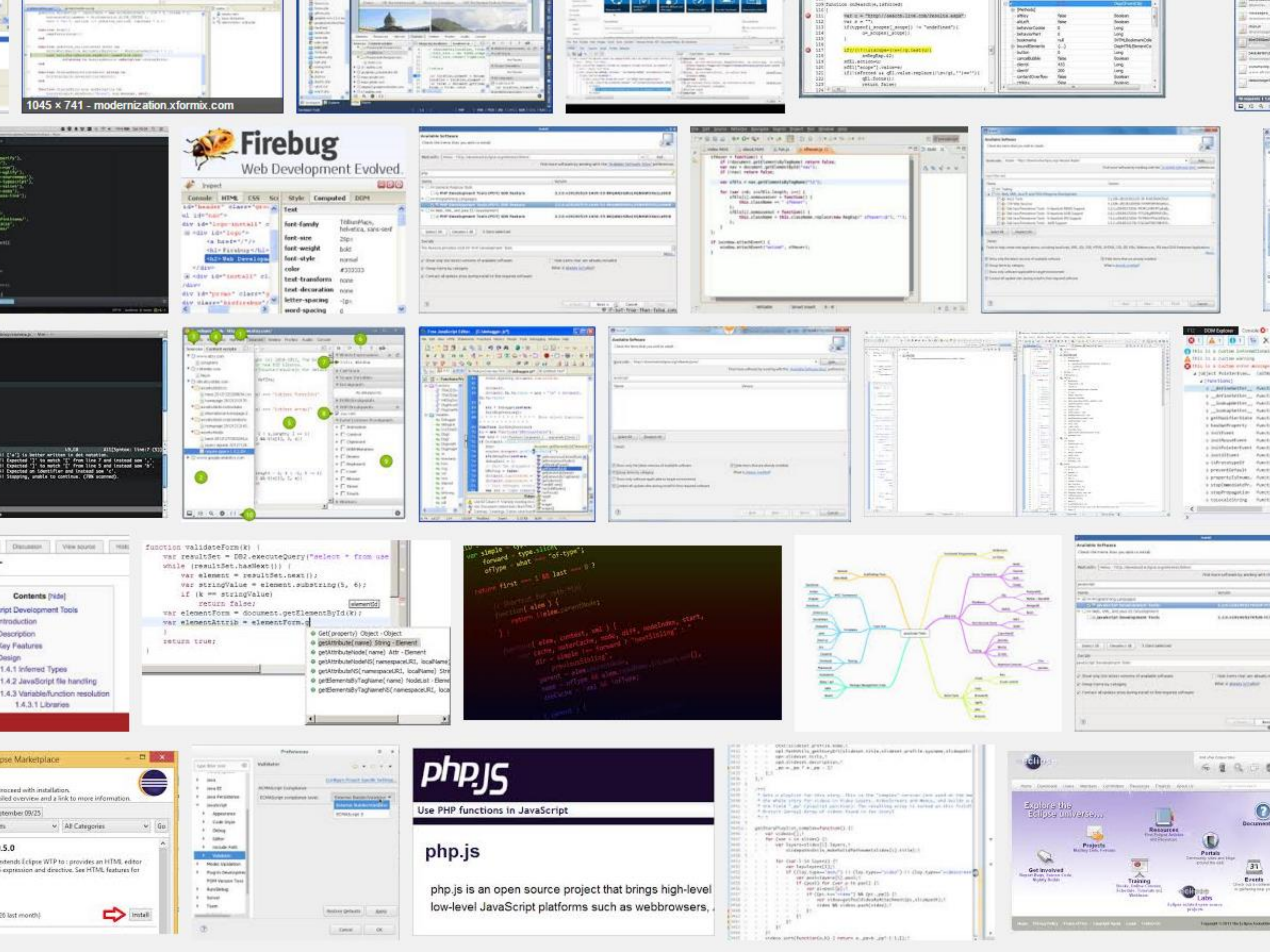
Body



"Once upon a time there was…"



https://www.chromium.org/developers/design-documents/multi-process-architecture



1045 × 741 - modernization.xformix.com

```
font-family     TitilliumMaps, helvetica, sans-serif
font-size       20px
font-weight     bold
font-style      normal
color           #333333
text-transform  none
text-decoration none
letter-spacing  -1px
word-spacing    0
```

```
'[a'] is better written in dot notation.
Expected ']' to match '[' from line 7 and instead saw ','.
Expected ')' to match '(' from line 5 and instead saw 'h'.
Expected an identifier and instead saw 'c'.
Stopping, unable to continue. (70% scanned).
```

Contents [hide]
ript Development Tools
ntroduction
Description
Key Features
Design
1.4.1 Inferred Types
1.4.2 JavaScript file handling
1.4.3 Variable/function resolution
1.4.3.1 Libraries

```
function validateForm(k) {
   var resultSet = DB2.executeQuery("select * from use
   while (resultSet.hasNext()) {
      var element = resultSet.next();
      var stringValue = element.substring(5, 6);
      if (k == stringValue)
         return false;
   var elementForm = document.getElementById(k);
   var elementAttrib = elementForm.g
   }
   return true;
}
```

```
Get(property) Object - Object
getAttribute(name) String - Element
getAttributeNode(name) Attr - Element
getAttributeNodeNS(namespaceURI, localName)
getAttributeNS(namespaceURI, localName) Str
getElementsByTagName(name) NodeList - Eleme
getElementsByTagNameNS(namespaceURI, loca
```

pse Marketplace

roceed with installation.
iled overview and a link to more information.

.5.0
xtends Eclipse WTP to : provides an HTML editor
expression and directive. See HTML features for

Install

php.js

Use PHP functions in JavaScript

php.js

php.js is an open source project that brings high-level
low-level JavaScript platforms such as webbrowsers,

- **Compile to JavaScript:**
- Coffescript/Typescript
- Google Web ToolKit (GWT) (Java)
- Google Dart

- **Plugins** e.g., Flash, Java
- Non-browser - Android/iOS **native**

- **WebAssembly** (future):
- C/C++ (anything in future ([demo]() using Unity)

- https://github.com/jashkenas/coffeescript/wiki/List-of-languages-that-compile-to-JS

```
+---------------------------------------------------+
|          Module and Application Ecosystem         |
+---------------------------------------------------+
     |                |            |              |
     |                | +----------------------+  |
     |                | |  Binary Abstraction  |  |
     |                | |        Layer         |  |
     |                | +----------------------+  |
     |                |     |             |       |
+----------------+    |     |             |       /
|  Node.js Core  |    |     |             |      /
|  Library API   |    |     |             |     /
+----------------+    |     |             |    /
|     js impl    |    |     |             |   /
+----------------+    |     |             |  /
         |            |     |             | /
+--------------------------------------+  |/
| Node.js Application Binary Interface |  |
+--------------------------------------|  |
|              C/C++ impl              |  |
+--------------------------------------+  |
                   |                      |
     +---------------------------------------+
     | Dependencies: v8, libuv, openssl, etc |
     +---------------------------------------+
```

Modules and npm

package.json

I/O (no sandbox)

No window, no document, no DOM

Modules and npm

package.json

I/O (no sandbox)

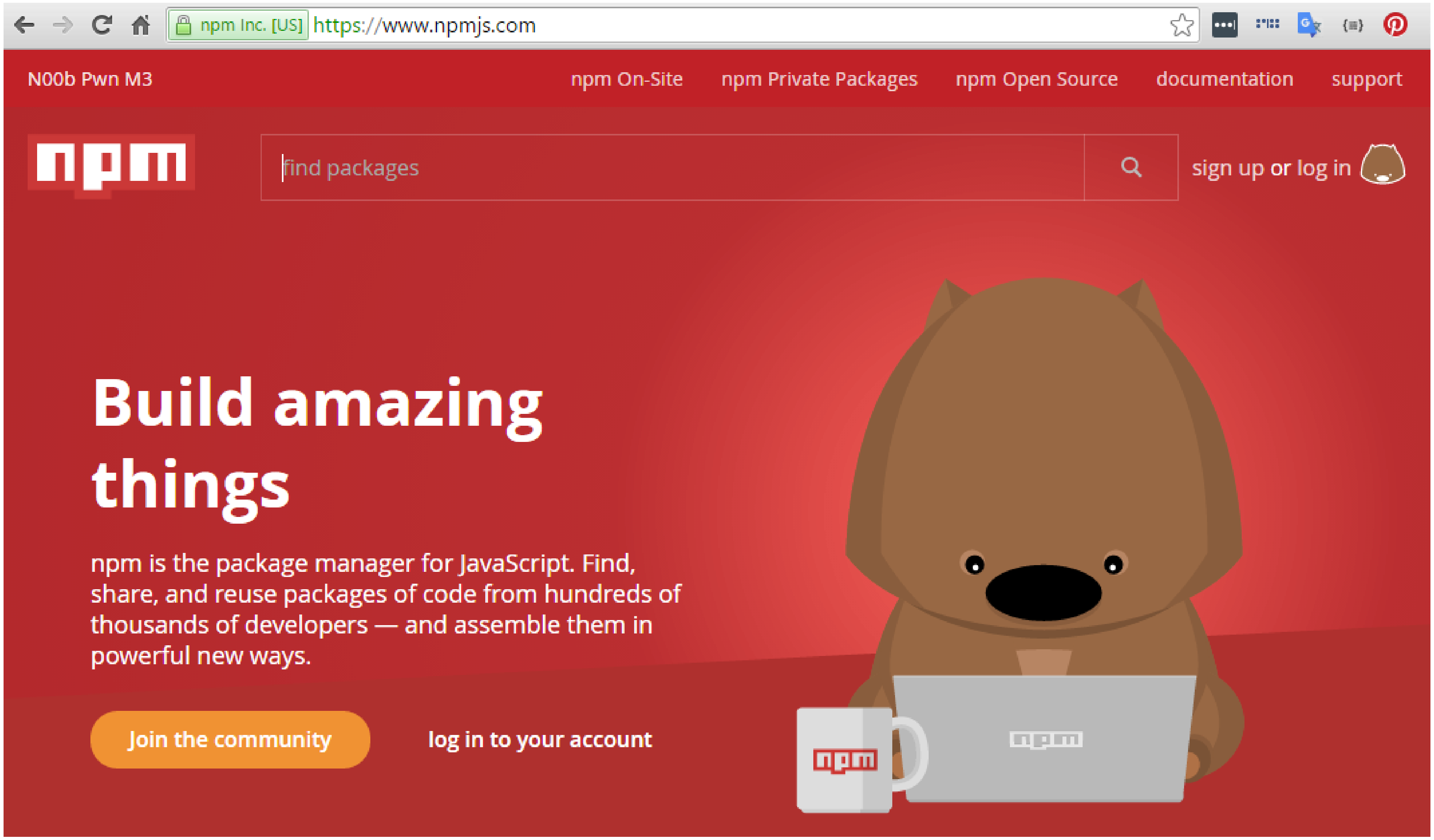
No window, no document, no DOM

But still:

- event loop
- single threaded
- asynchronous

$\left.\begin{array}{l}\text{event loop}\\\text{single threaded}\\\text{asynchronous}\end{array}\right\}$ V8 JavaScript Engine

```
node -p process.versions.v8
```



npm Inc. [US] https://www.npmjs.com

N00b Pwn M3

npm On-Site npm Private Packages npm Open Source documentation support

find packages

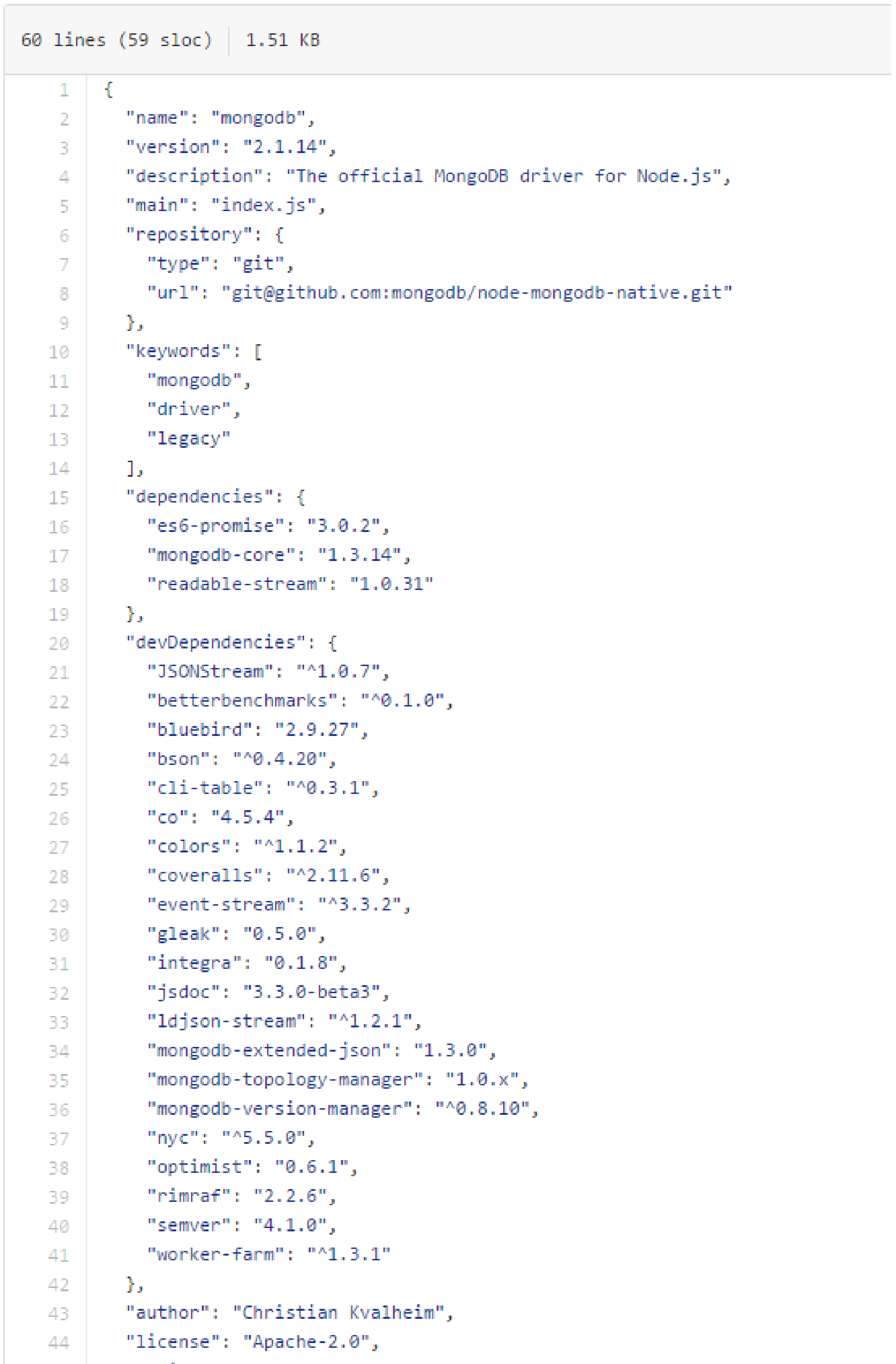
sign up or log in

Build amazing things

npm is the package manager for JavaScript. Find, share, and reuse packages of code from hundreds of thousands of developers — and assemble them in powerful new ways.

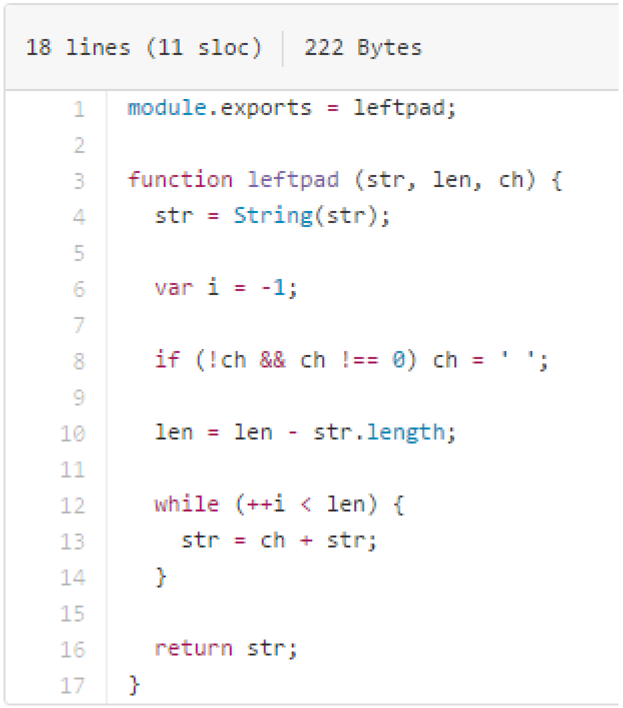
Join the community

log in to your account



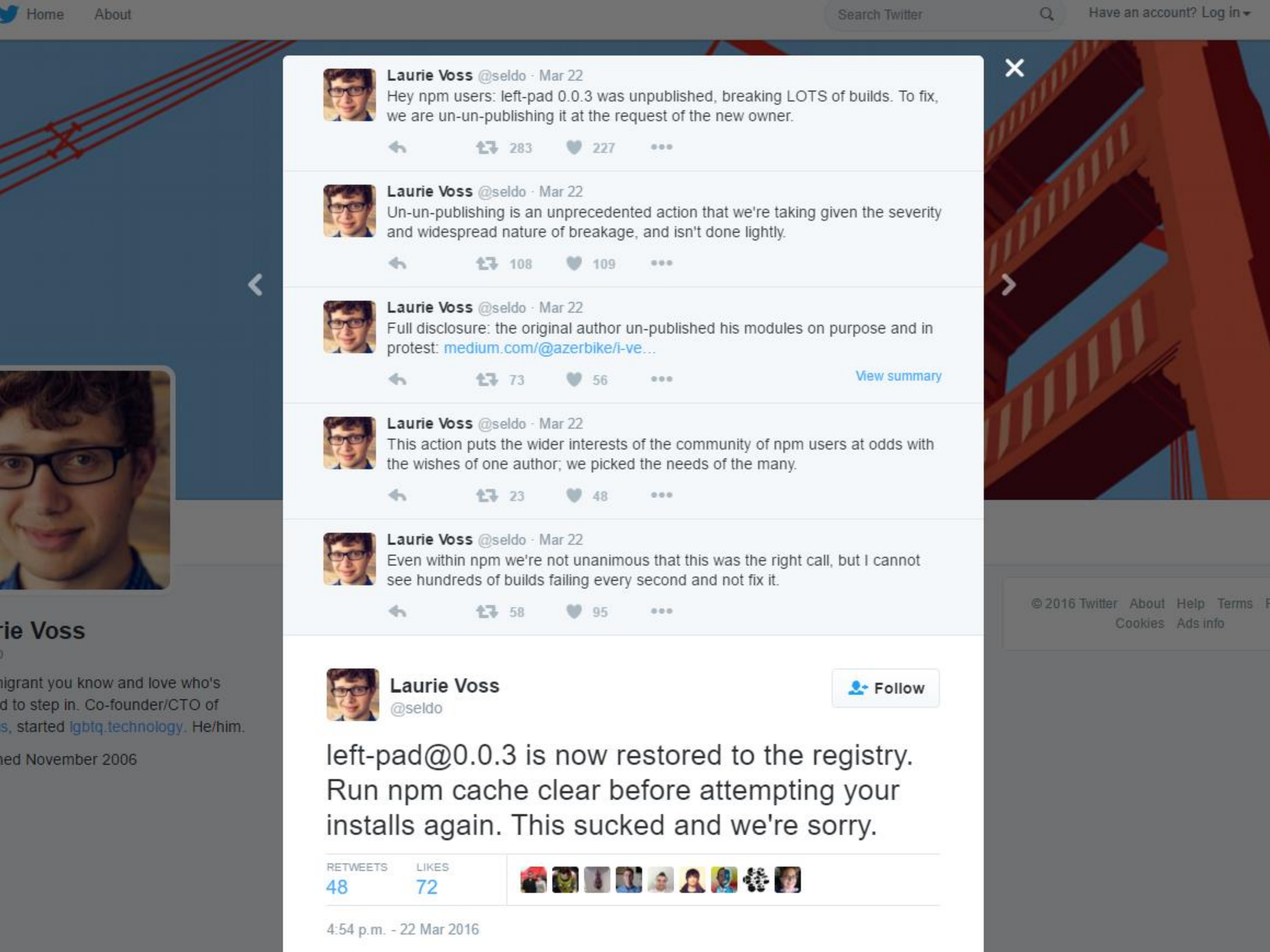
60 lines (59 sloc) | 1.51 KB

```
{
  "name": "mongodb",
  "version": "2.1.14",
  "description": "The official MongoDB driver for Node.js",
  "main": "index.js",
  "repository": {
    "type": "git",
    "url": "git@github.com:mongodb/node-mongodb-native.git"
  },
  "keywords": [
    "mongodb",
    "driver",
    "legacy"
  ],
  "dependencies": {
    "es6-promise": "3.0.2",
    "mongodb-core": "1.3.14",
    "readable-stream": "1.0.31"
  },
  "devDependencies": {
    "JSONStream": "^1.0.7",
    "betterbenchmarks": "^0.1.0",
    "bluebird": "2.9.27",
    "bson": "^0.4.20",
    "cli-table": "^0.3.1",
    "co": "4.5.4",
    "colors": "^1.1.2",
    "coveralls": "^2.11.6",
    "event-stream": "^3.3.2",
    "gleak": "0.5.0",
    "integra": "0.1.8",
    "jsdoc": "3.3.0-beta3",
    "ldjson-stream": "^1.2.1",
    "mongodb-extended-json": "1.3.0",
    "mongodb-topology-manager": "1.0.x",
    "mongodb-version-manager": "^0.8.10",
    "nyc": "^5.5.0",
    "optimist": "0.6.1",
    "rimraf": "2.2.6",
    "semver": "4.1.0",
    "worker-farm": "^1.3.1"
  },
  "author": "Christian Kvalheim",
  "license": "Apache-2.0",
```



18 lines (11 sloc) | 222 Bytes

```javascript
module.exports = leftpad;

function leftpad (str, len, ch) {
  str = String(str);

  var i = -1;

  if (!ch && ch !== 0) ch = ' ';

  len = len - str.length;

  while (++i < len) {
    str = ch + str;
  }

  return str;
}
```



Home About Search Twitter Have an account? Log in

Laurie Voss @seldo · Mar 22
Hey npm users: left-pad 0.0.3 was unpublished, breaking LOTS of builds. To fix, we are un-un-publishing it at the request of the new owner.
283 227

Laurie Voss @seldo · Mar 22
Un-un-publishing is an unprecedented action that we're taking given the severity and widespread nature of breakage, and isn't done lightly.
108 109

Laurie Voss @seldo · Mar 22
Full disclosure: the original author un-published his modules on purpose and in protest: medium.com/@azerbike/i-ve...
73 56 View summary

Laurie Voss @seldo · Mar 22
This action puts the wider interests of the community of npm users at odds with the wishes of one author; we picked the needs of the many.
23 48

Laurie Voss @seldo · Mar 22
Even within npm we're not unanimous that this was the right call, but I cannot see hundreds of builds failing every second and not fix it.
58 95

Laurie Voss
@seldo
Follow

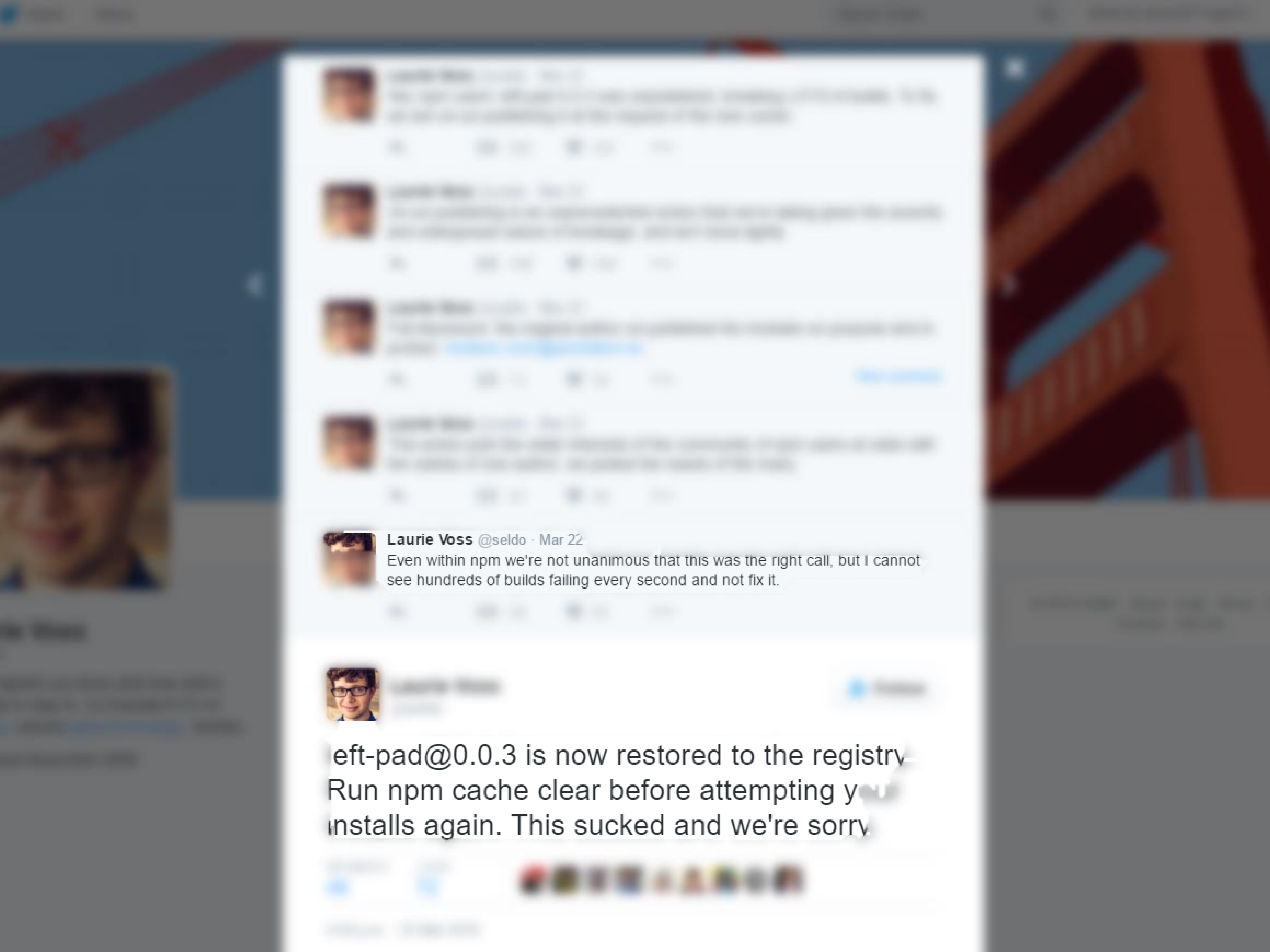
left-pad@0.0.3 is now restored to the registry. Run npm cache clear before attempting your installs again. This sucked and we're sorry.

RETWEETS 48 LIKES 72

4:54 p.m. - 22 Mar 2016

rie Voss
igrant you know and love who's
d to step in. Co-founder/CTO of
s, started lgbtq.technology. He/him.
ned November 2006

© 2016 Twitter About Help Terms
Cookies Ads info



Laurie Voss @seldo · Mar 22
Even within npm we're not unanimous that this was the right call, but I cannot see hundreds of builds failing every second and not fix it.

left-pad@0.0.3 is now restored to the registry. Run npm cache clear before attempting your installs again. This sucked and we're sorry.

CommonJS – Node.js

```
const $ = require(‘jQuery’);
const _ = require(‘lodash’);
const array = require('lodash/array');
npm install –g lodash
npm install --save express
```

AMD (Async Module Defn) – ES2015

```
import * as name from “module-name”;
import { member } from “module-name”;
```

HOW DOES AN APPLICATION RESPOND TO MULTIPLE OVERLAPPING REQUESTS?

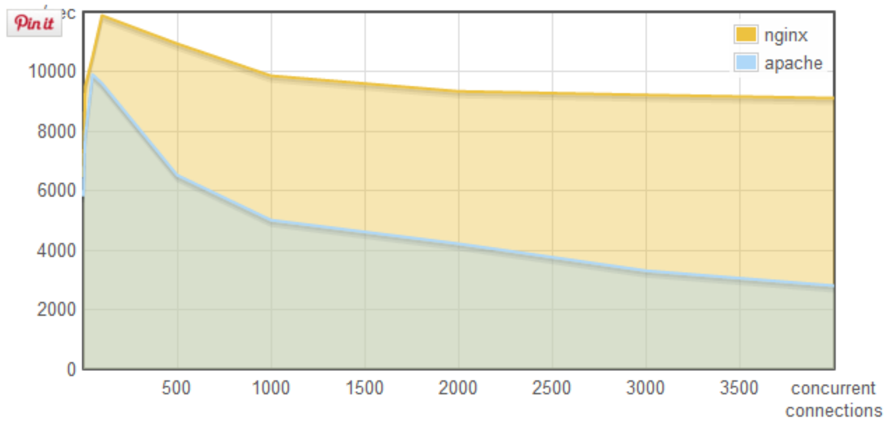
A little holiday present: 10,000 reqs/sec with Nginx!

Posted in Server setup December 18, 2008 by Remi D

Updated Dec 19 at 05:15 CDT (first posted Dec 18 at 06:01 CDT) by Remi

A few weeks ago we quietly started to configure our new machines with Nginx as the front web server instead of Apache (we still run Apache behind Nginx for people who need all the features from Apache).

Here is a little benchmark that I did to compare Nginx versus Apache (with the worker-MPM) for serving a small static file:

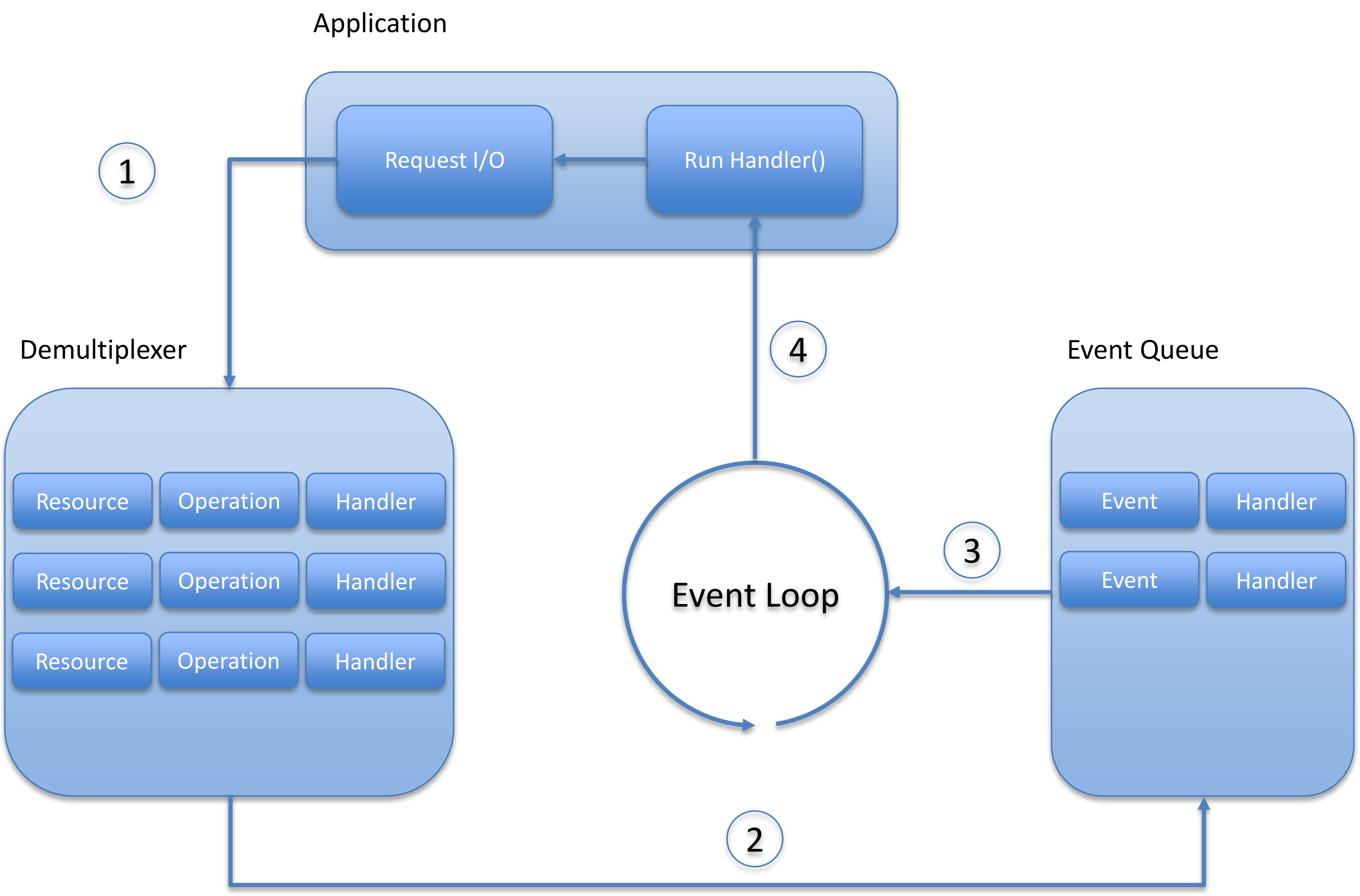


This benchmark is not representative of a real-world application because in my benchmark the web
servers were only serving a small static file from localhost (in real life your files would get served to

Concurrency (vs Parallelism)

"In programming, concurrency is the composition of independently executing processes, while parallelism is the simultaneous execution of (possibly related) computations. **Concurrency** is about **dealing** with lots of things at once. **Parallelism** is about **doing** lots of things at once." - Concurrency is not Parallelism, the GoLang blog

WHEN DOES MY CODE RUN?



Node.js Design Patterns, 2nd edition, Casciaro and Mammino, 2016, Packt Publishing

WHAT BINDINGS ARE IN EFFECT WHEN MY CODE RUNS?

8.1.2.1 GetIdentifierReference (lex, name, strict)

The abstract operation GetIdentifierReference is called with a Lexical Environment *lex*, a String *name*, and a Boolean flag *strict.* The value of *lex* may be **null**. When called, the following steps are performed:

1. If *lex* is the value **null**, then
    1. Return a value of type Reference whose base value is **undefined**, whose referenced name is *name*, and whose strict reference flag is *strict*.
2. Let *envRec* be *lex*'s EnvironmentRecord.
3. Let *exists* be *envRec*.HasBinding(*name*).
4. ReturnIfAbrupt(*exists*).
5. If *exists* is **true**, then
    1. Return a value of type Reference whose base value is *envRec*, whose referenced name is *name*, and whose strict reference flag is *strict.*
6. Else
    1. Let *outer* be the value of *lex's* outer environment reference.
    2. Return GetIdentifierReference(*outer*, *name*, *strict*).

http://www.ecma-international.org/ecma-262/6.0/#sec-getidentifierreference

Async

Callbacks

Scoping

Closures

this

Promises (ES2015)

1. Strict mode always
2. Use *let* and *const*, not *var* (or nothing)

“Closures are functions that refer to independent (free) variables. In other words, the function defined in the closure 'remembers' the environment in which it was created.”

https://developer.mozilla.org/en/docs/Web/JavaScript/Closures

“Closures are functions that refer to independent (free) variables. In other words, the <u>function</u> defined in the closure 'remembers' the <u>environment</u> in which it was created.”

https://developer.mozilla.org/en/docs/Web/JavaScript/Closures

“Closures are functions that refer to independent (free) variables. In other words, the function defined in the closure 'remembers' the environment in which it was created.”

Corollary of lexical scoping – functions are executed in scope in which they are <u>defined</u>, not scope in which executed (dynamic scope) *this* is a reference to the execution context when a function is called

1. Strict mode always
2. Use *let* and *const*, not *var* (or nothing)
3. Understand *this*, and be careful about context
   – ES5: use .bind or var self = this
   – ES6: =>

The Promise object is used for deferred and asynchronous computations.

A Promise represents an operation that hasn't completed yet, but is expected in the future.

- *pending*: initial state, not fulfilled or rejected.
- *fulfilled*: meaning that the operation completed successfully.
- *rejected*: meaning that the operation failed.

https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Promise

Standard for promises is Promises/A+

Previously as shim e.g., Q, Bluebird, when

Now in core API of many browsers, Node.js

https://github.com/promises-aplus/promises-spec

http://kangax.github.io/compat-table/es6/

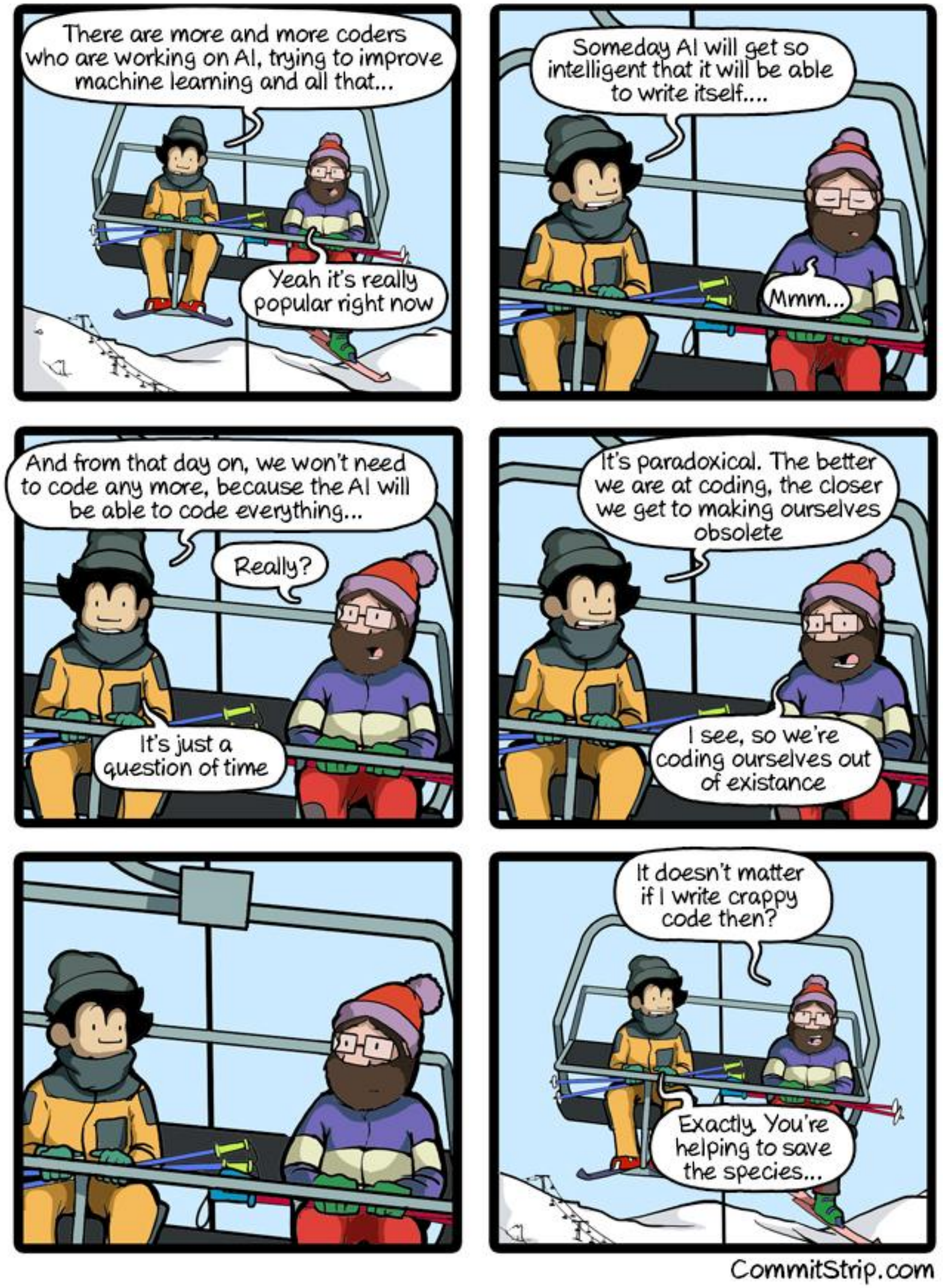
http://www.html5rocks.com/en/tutorials/es6/promises/

1. Strict mode always
2. Use *let* and *const*, not *var* (or nothing)
3. Understand *this*, and be careful about context
   – ES5: use .bind or var self = this
   – ES6: =>
4. Promises in preference to callbacks

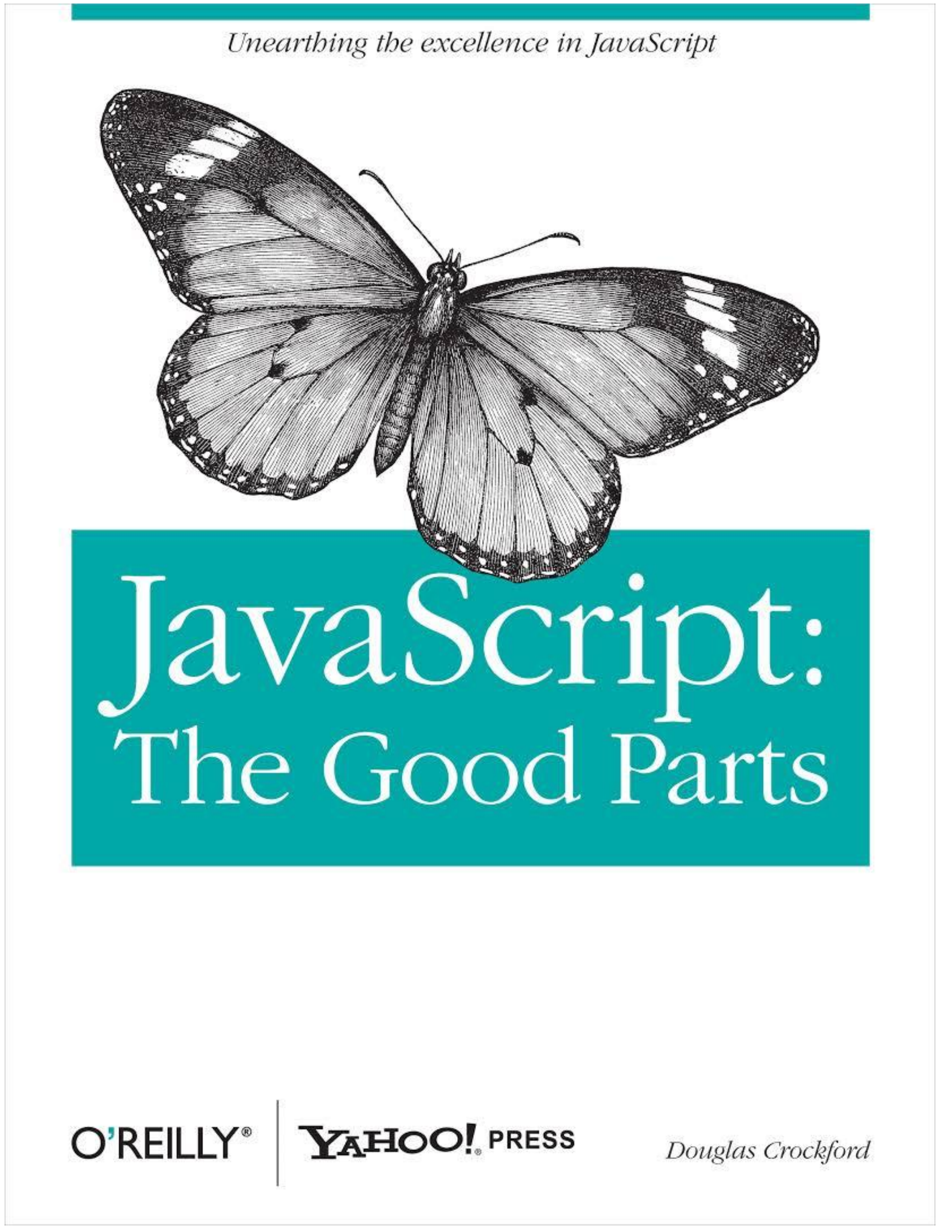
ES6/ES2015 Compatibility Table
Google Trends for JavaScript

http://arc.applause.com/2016/03/22/javascript-is-the-worlds-dominant-programming-language
/http://stackoverflow.com/research/developer-survey-2016#technology
https://github.com/blog/2047-language-trends-on-github

http://www.tiobe.com/tiobe_index


There are more and more coders who are working on AI, trying to improve machine learning and all that...
Yeah it's really popular right now
Someday AI will get so intelligent that it will be able to write itself....
Mmm...
And from that day on, we won't need to code any more, because the AI will be able to code everything...
Really?
It's just a question of time
It's paradoxical. The better we are at coding, the closer we get to making ourselves obsolete
I see, so we're coding ourselves out of existance
It doesn't matter if I write crappy code then?
Exactly. You're helping to save the species...
CommitStrip.com

http://www.commitstrip.com/en/2016/03/01/the-end-of-the-coders/


Unearthing the excellence in JavaScript
JavaScript: The Good Parts
O'REILLY | YAHOO! PRESS
Douglas Crockford

By Robert Claypool (Own work) [CC0], via Wikimedia Commons

https://github.com/douglascrockford/JSLint

You can always get better…

Local groups and meetups - chc.js, APN, http://www.meetup.com/Functional-Christchurch/, http://canterburysoftware.org.nz/

Github for samples

Style guides

- https://github.com/airbnb/javascript
- https://google.github.io/styleguide/javascriptguide.xml
- JSLint, ESLint
- JSRC – preset styles

Patterns - e.g., https://github.com/tfmontague/definitive-module-pattern

Blogs, e.g.,

- http://jrsinclair.com/articles/2016/gentle-introduction-to-functional-javascript-functions
- https://github.com/ericelliott/essential-javascript-links#essential-javascript-links

https://tc39.github.io/ecma262/

Next week

Prep for next week

How can access to an API be restricted? Controlled?

How can access that bypasses an API be prevented?

Further reading

- Course wiki – ES2015/Javascript
- Course wiki – Async design patterns

State and Statelessness

Memory of preceding "events"
Set of bindings

CHAPTER I

THE FAMILY OF Dashwood had been long settled in Sussex. Their estate was large, and their residence was at Norland Park, in the centre of their property, where for many generations, they had lived in so respectable a manner, as to engage the general good opinion of their surrounding acquaintance. The late owner of this estate was a single man, who lived to a very advanced age, and who for many years of his life, had a constant companion and housekeeper in his sister. But her death, which happened ten years before his own, produced a great alteration in his home; for to supply her loss, he invited and received into his house the family of his nephew Mr Henry Dashwood, the legal inheritor of the Norland estate, and the person to whom he intended to bequeath it. In the society of his nephew and niece, and their children, the old Gentleman's days were comfortably spent. His attachment to them all increased. The constant attention of Mr and Mrs Henry Dashwood to his wishes, which proceeded not merely from interest, but from goodness of heart, gave him every degree of solid comfort which his age could receive; and the cheerfulness of the children added a relish to his existence.

https://www.flickr.com/photos/wastes/6119262807/in/photolist-ajJPDa

State timescales

Individual HTTP request (stateless)

Business transaction

Session

Preferences

Record state

Session (state) information

For *web applications* (in contrast to public websites or webpages) there is a need to maintain some stateful information about the client

GET ? parameters

Maintain some kind of session variable in the parameter to the HTTP request

Variable does not contain the username and password, but a unique (‘random’) identifier

Include variable as a parameter in **each** network requests, e.g.

```
   GET www.example.com?sessionid=<var>
```

Why is this ‘bad practice’?

Why may something like this be needed, at times?

Cookie

Use cookies to maintain session information

The server issues a unique (‘random’) identifier in the cookie to the client, for that username & password

Client sends back the cookie with **each** network request to the server

- e.g. in the POST data

Note that the username and password are not sent (once the user is logged on).

Cookies

A small piece of data initially sent by the server to the client.

- Comprises name-value pairs
- Also has attributes (that are not sent back to the server)

Used to maintain state information

- e.g. items in a shopping basket
  (although this example may be better kept on the server)
- e.g. browser activity such as a ‘path’ through a registration process

Types of cookie

First-party cookie	A cookie set by the server to which the browser primarily connects.
Session cookie	Exists only for the duration of that browser session, and the browser typically deletes the cookie
Persistent cookie (aka tracking cookie)	Persistent data. The cookie is not deleted when the browser closes. Can be used by advertising to track user behaviour. Can be used to store credentials e.g. log in details.
Secure cookie	A cookie that can only be transmitted over an encrypted connection, such as HTTPS.
HTTPOnly cookie	Can only be transmitted through HTTP/S, and are not accessible through non-HTTP APIs such as JavaScript.
Third-party cookie	Cookies set by third-parties that serve content to the page e.g. advertising.

Sequence of cookie-ing

Request from browser

```
GET /index.htm HTTP/1.1
Host: www.example.com
```

Response from server

```
HTTP/1.1 200 OK
Content-type: text/html
Set-cookie: sessionToken=a1b2c3; Expires = [dat]
```

Follow-up request from browser

```
GET /profile.htm HTTP/1.1
Host: www.example.com
Cookie: sessionToken=a1b2c3
```

Those cookies (review)

Use cookies to maintain information

Send the cookie with **each** network request

- e.g. in the POST data

What's the security risk with including the cookie in the POST data?

How might we address this risk?

Limitations of cookies

Each browser maintains its own ‘cookie jar’

A cookie does not identify a person

A cookie identifies the combination of:

- User account
- Web browser
- Device

A cookie requires that the browser is cookie-enabled and is set to allow cookies

Standard session ID names

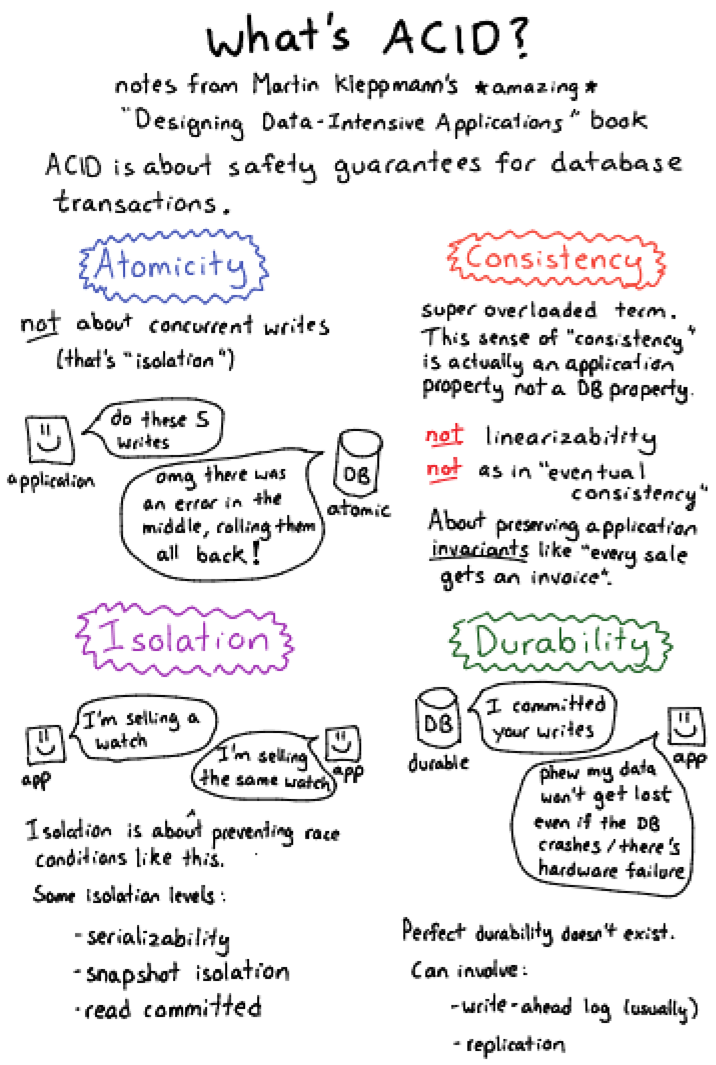
Examples of standard names for session IDs

JSESSIONID (Java EE)

PHPSESSID (PHP)

ASPSESSIONID (Microsoft ASP)

Why is it unwise to use standard names for variables in your application?



what's ACID?

notes from Martin Kleppmann's ★amazing★ "Designing Data-Intensive Applications" book

ACID is about safety guarantees for database transactions.

Atomicity

not about concurrent writes (that's "isolation")

application: do these 5 writes

DB atomic: omg there was an error in the middle, rolling them all back!

Consistency

super overloaded term. This sense of "consistency" is actually an application property not a DB property.

not linearizability

not as in "eventual consistency"

About preserving application invariants like "every sale gets an invoice".

Isolation

app: I'm selling a watch

app: I'm selling the same watch

Isolation is about preventing race conditions like this.

Some isolation levels:

- serializability
- snapshot isolation
- read committed

Durability

DB durable: I committed your writes

app: phew my data won't get lost even if the DB crashes / there's hardware failure

Perfect durability doesn't exist.

Can involve:

- write-ahead log (usually)
- replication

https://drawings.jvns.ca/drawings/acid.svg

PERSISTENCE

Long lived state

Data access patterns (Fowler)

DAO

ActiveRead

DataMapper

TableDataListener

TableModule