



Making ES6 available to all with ChakraCore

Chris Heilmann @codepo8, JFokus, Stockholm, February 2015

A black and white photograph of Chris Heilmann. He is a man with long, curly hair, wearing glasses and a dark t-shirt with a 'PLAN.JS' logo. He is looking down at a smartphone in his right hand and adjusting his ear with his left hand. The background is blurred, showing other people and bright lights.

CHRIS HEILMANN
@CODEPO8

LET'S TALK JAVASCRIPT...

- Old issues
- The learning process
- The library/framework issue
- The ES6 buffet
- Standards and interop
- Breaking monopolies

OLD ISSUES OF JAVASCRIPT

JAVASCRIPT CLIENT-SIDE HAS ISSUES...

- It is not fault-tolerant
- Many different parties mess with it
- You don't know the environment it runs in
- It has always been part of the browser and dependent on its release and upgrade cycle

JAVASCRIPT THE LANGUAGE HAS ISSUES (OPPORTUNITIES)...

- `typeof NaN === number`
- No type safety
- No classes
- "it feels rushed"

ENGINE TROUBLE:
JAVASCRIPT IS
HUNGRY



HTTPS://



.WTF

THE JAVASCRIPT LEARNING PROCESS

THE JAVASCRIPT LEARNING PROCESS HAS ALWAYS BEEN INTERESTING...

- Use view source to see what others are doing...
- Copy and paste the bits that look like they are responsible for some things
- Change some numbers around
- Run into errors
- Blame Internet Explorer

THIS, OF COURSE,
WAS WRONG AND
WE GOT MORE
PROFESSIONAL...

- Search for a solution on Stackoverflow
- Copy and paste the bits that look like they are responsible for some things
- Change some numbers around
- Run into errors
- Blame JavaScript for being terrible and not a real language
- For good measure, blame Internet Explorer.

IT'S TRUE...



Jordan Hall
@DivineOmega

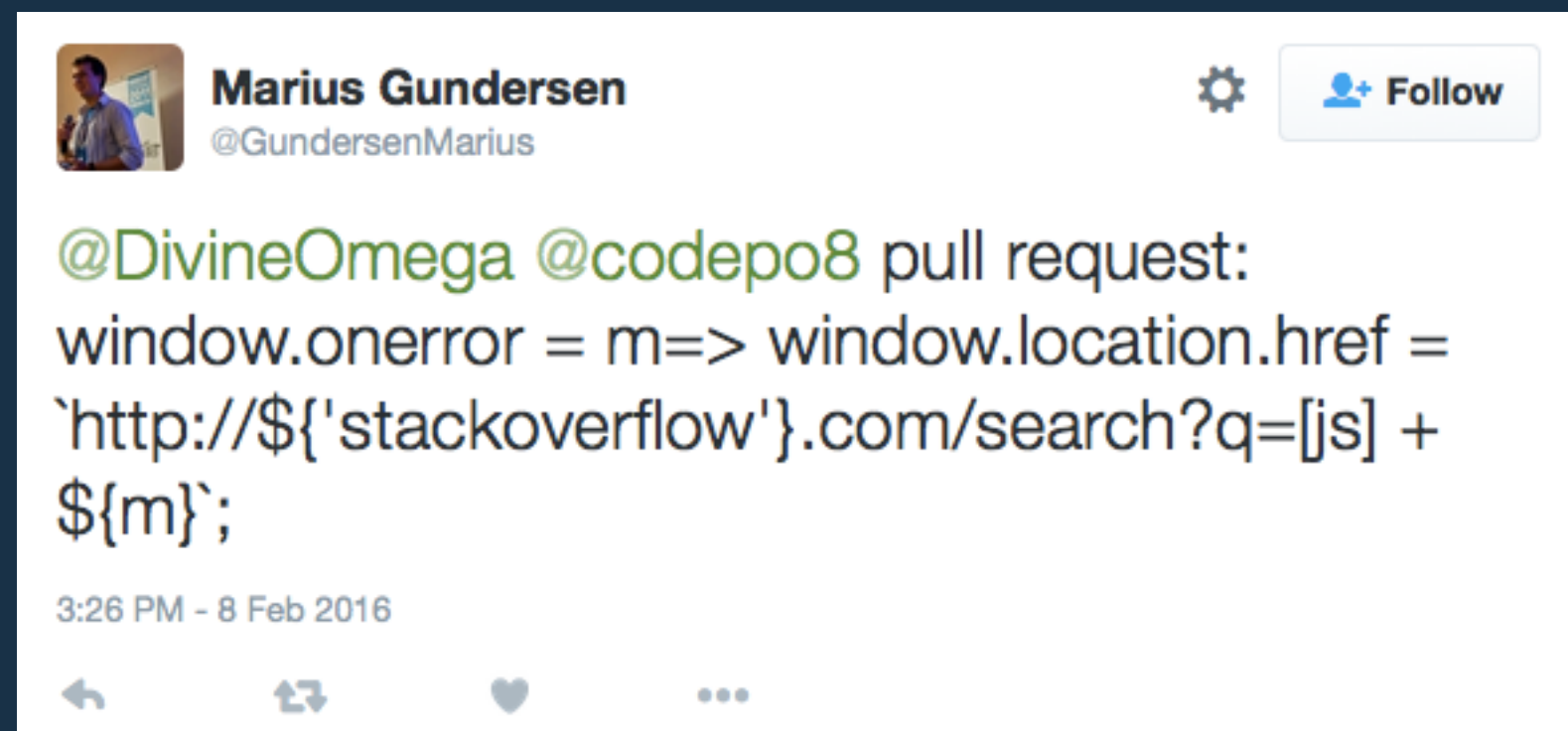



 Follow

The one true **#JavaScript** exception handler. ;)

```
try {  
  something  
} catch(e) {  
  window.location.href =  
    "http://stackoverflow.com/search?q=[js] + "  
    + e.message;  
}
```

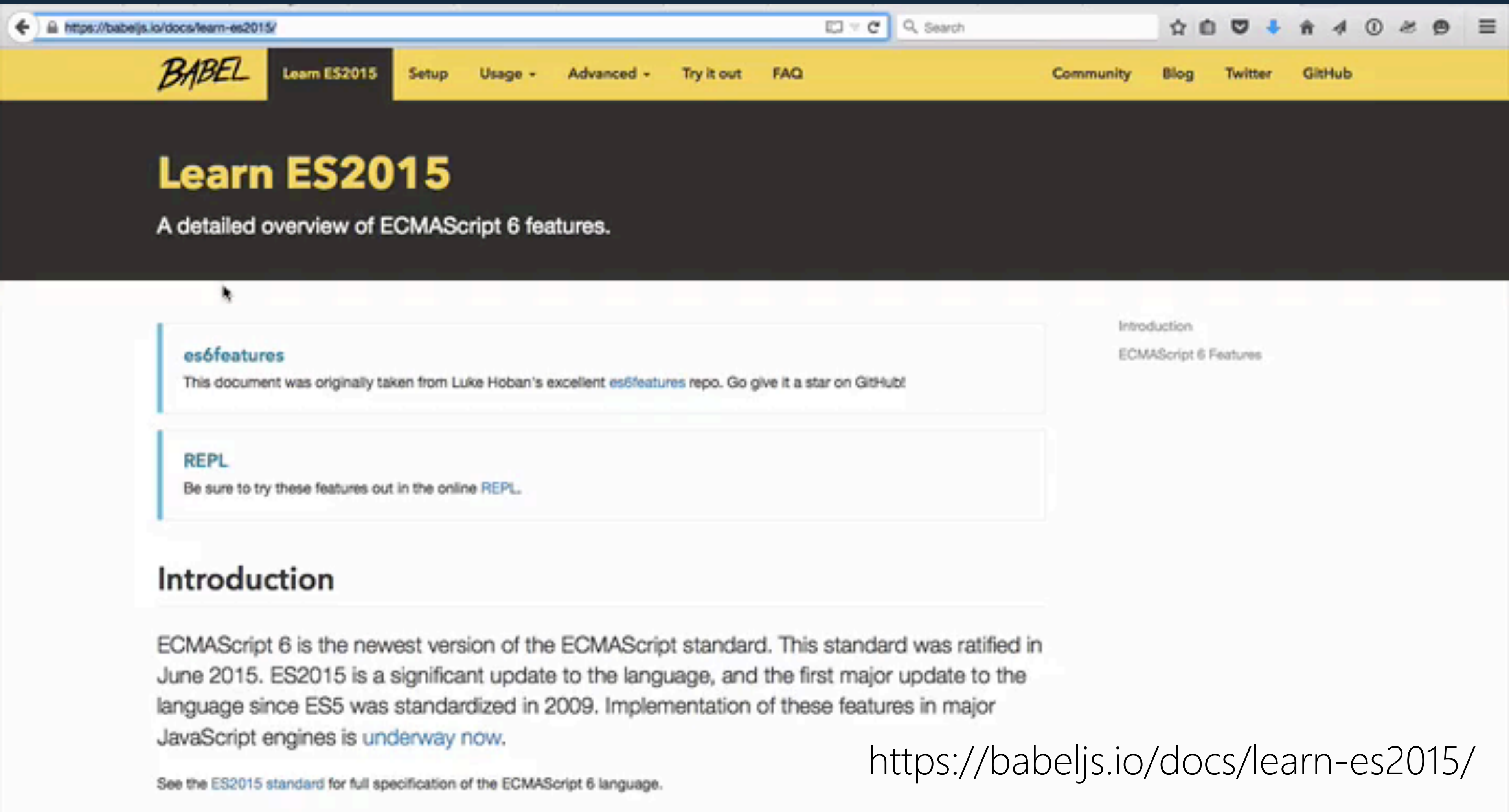

BUT OF COURSE IT CAN
BE IMPROVED...



A person's silhouette is shown in profile, interacting with a large digital screen. The screen displays a blue grid pattern, and the person's hand is visible, touching the screen. The background is dark, and the screen is the primary light source.

WITH ES6 WE HAVE
A CLEAN NEW
SLATE...
(AND YOU CAN'T BLAME IE ANY MORE)

SEE THE BABEL.JS DOCS AND TRY IT IN THE BROWSER...



The screenshot shows the Babel.js website with a yellow header. The header contains the Babel logo, a navigation menu with 'Learn ES2015' (highlighted), 'Setup', 'Usage', 'Advanced', 'Try it out', and 'FAQ', and social links for 'Community', 'Blog', 'Twitter', and 'GitHub'. The main content area has a dark background with the title 'Learn ES2015' in large yellow letters, followed by the subtitle 'A detailed overview of ECMAScript 6 features.' Below this, there are two light blue boxes: one for 'es6features' with a link to Luke Hoban's repo, and another for 'REPL' with a link to the online REPL. On the right side, there is a sidebar with links for 'Introduction' and 'ECMAScript 6 Features'. The main text area starts with the heading 'Introduction' and a paragraph about ECMAScript 6. At the bottom, there is a link to the ES2015 standard.

https://babeljs.io/docs/learn-es2015/

BABEL Learn ES2015 Setup Usage - Advanced - Try it out FAQ Community Blog Twitter GitHub

Learn ES2015

A detailed overview of ECMAScript 6 features.

es6features
This document was originally taken from Luke Hoban's excellent [es6features](#) repo. Go give it a star on GitHub!

REPL
Be sure to try these features out in the online [REPL](#).

Introduction

ECMAScript 6 is the newest version of the ECMAScript standard. This standard was ratified in June 2015. ES2015 is a significant update to the language, and the first major update to the language since ES5 was standardized in 2009. Implementation of these features in major JavaScript engines is [underway now](#).

See the [ES2015 standard](#) for full specification of the ECMAScript 6 language.

Introduction
ECMAScript 6 Features

https://babeljs.io/docs/learn-es2015/

350 BULLET POINTS

<https://babeljs.io/docs/learn-es2015/>



PONYFOO.COM

[Home](#) [History](#) [Feed](#) [About](#)

ES6 Overview in 350 Bullet Points

My **ES6 in Depth** series consists of 24 articles covering most syntax changes and features coming in ES6. This article aims to summarize all of those, providing you with practical insight into most of ES6, so that you can quickly get started. I've also linked to the articles in ES6 in Depth so that you can easily go deeper on any topic you're interested in.

I heard you like bullet points, so I made an article containing hundreds of those bad boys. To kick things off, here's a table of contents with all the topics covered. It has bullet points in it – **obviously**. Note that if you want these concepts to permeate your brain, you'll have a much

READ THE
EXCELLENT BOOK
EXPLORING ES6
FOR FREE

(OR BUY IT, AXEL DESERVES SUPPORT)

EXPLORING ES6

Upgrade to the next version of JavaScript



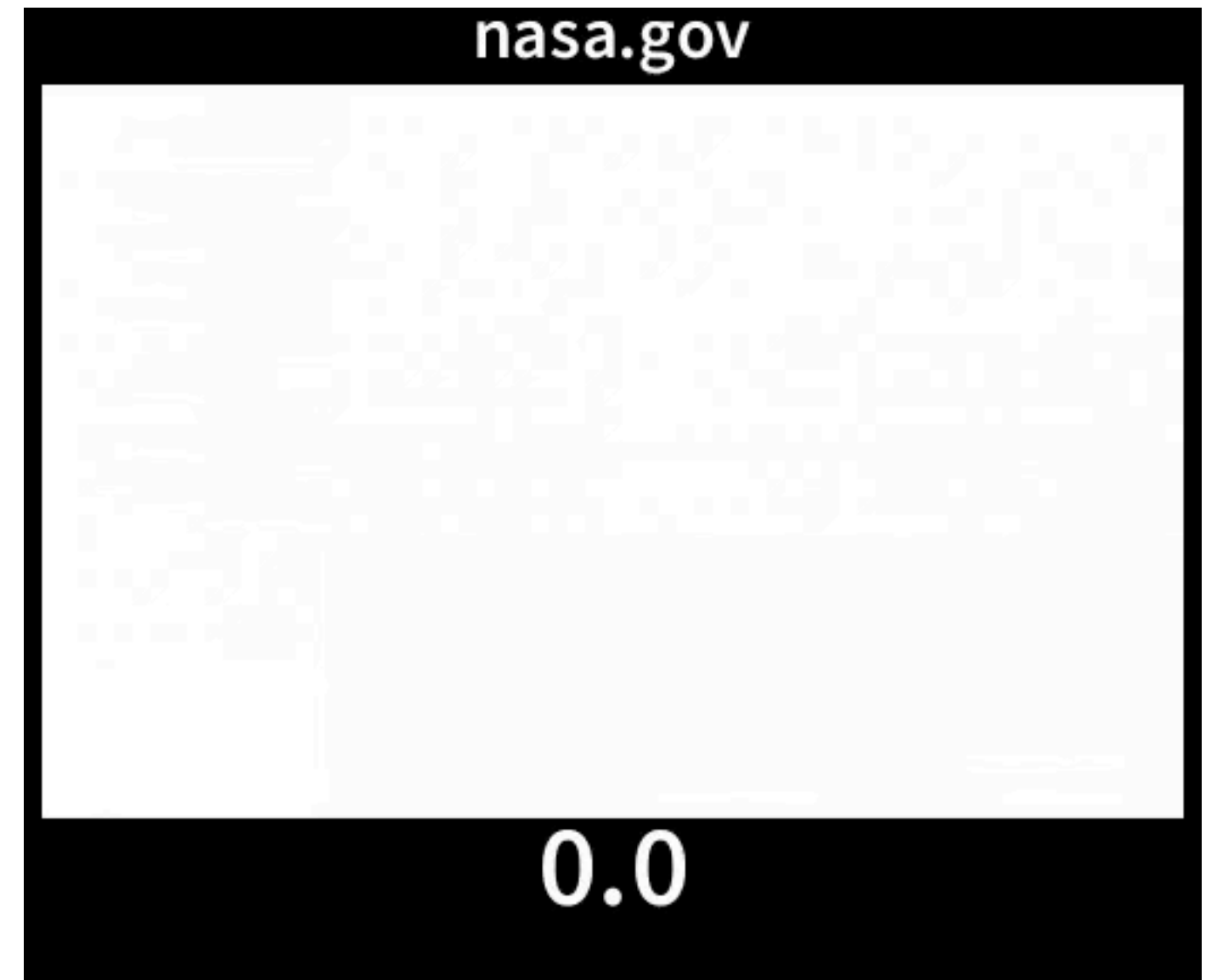
Dr. Axel Rauschmayer

Ecmanauten

<http://exploringjs.com/es6/>

THE LIBRARY/Framework ISSUE...

JAVASCRIPT ABUSE IS
RAMPANT...



SHOULD WE BUILD
EVERYTHING FROM
SCRATCH?

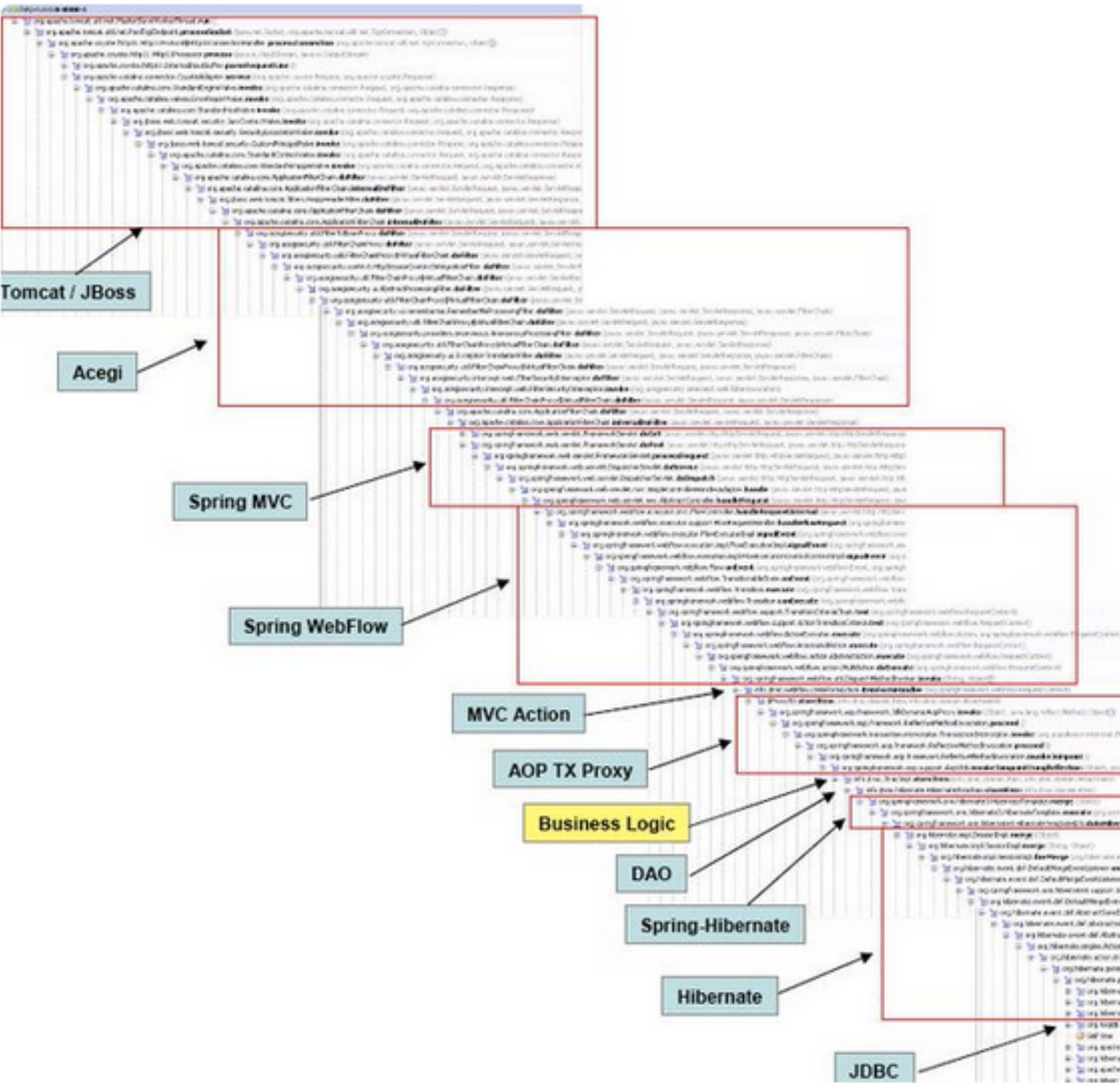




Vlad Yaroslavlev
@vladon

Follow

Modern comfortable programming language
#java



RETWEETS 2,064
FAVORITES 1,138



5:59 AM - 28 Oct 2015

OLD ISSUES, NOW IN
THE NEXT LANGUAGE...

IS DEPENDENCY HELL
A PROBLEM OF THE
TOO PRIVILEGED?

Stephan Bönnemann (JSConfEU 2015):
Dependency Hell Just Froze Over



Stephan Bönnemann: Dependency Hell Just Froze Over | JSConf EU 2015

<https://www.youtube.com/watch?v=PA139CERNbc>

WE'RE GOING
FULL SPEED ON
INNOVATION...

- Componentised Web
- Extensible Web Manifesto
- WebGL
- WebAssembly/ASM.js
- PostCSS
- Progressive Apps

BUILDING LIBRARIES
AND FRAMEWORKS
THAT MAGICALLY
FIX THINGS HAS
BECOME
FASHIONABLE...

- We work around browser issues
- We make web standards of tomorrow work today.
- We build solutions to clean up others and make them smaller
- And each of those comes with a “don’t use in production” label.

NSFW



THAT'S COOL -
GO NUTS AND
PLAY WITH ANY
TECHNOLOGY
YOU WANT...



FRAMEWORKS ARE GREAT...

- They are fun to use - achieve a lot very quickly
- You build complex apps in a matter of minutes
- They work around pesky browser bugs
- They are good on your CV

...BUT THEY ALSO
COME WITH
DEVELOPER COST

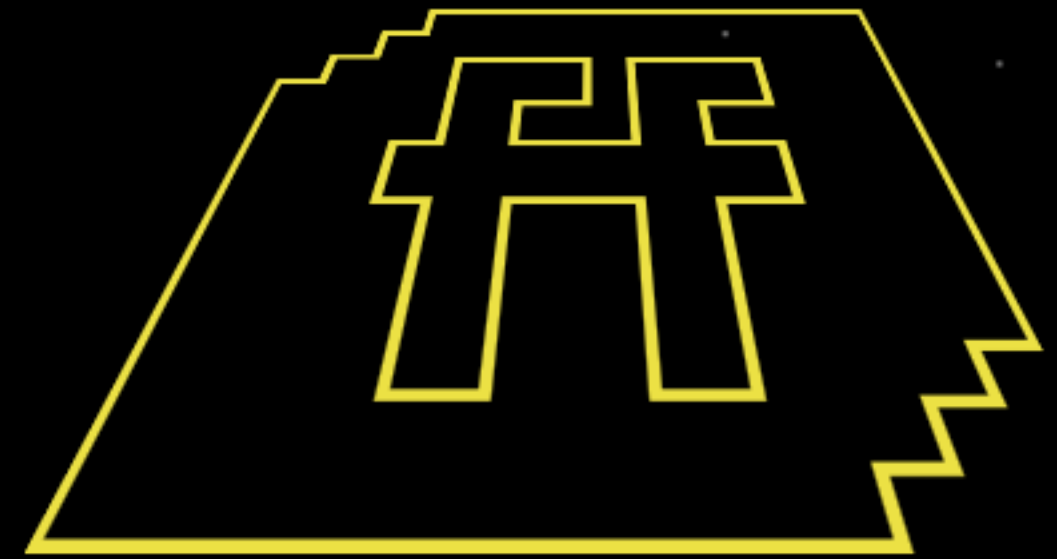
- Learning new frameworks
- Re-learning new frameworks
- Debugging frameworks
- Setting up developer environments
- Cutting down on possible hires/
adding to onboarding time

AND WE SHOULD
CONSIDER THE
EFFECTS WE HAVE
ON OUR END
USERS...

- Time to load / execute
- Bandwidth used
- CPU usage
- Frame rate (60 fps)
- Memory usage
- Battery hunger

HOMework / SNEAK PREVIEW OF GREAT INSIGHTS...

PAUL LEWIS
@AEROTWIST



NOV 6TH 2015 @ 9AM

BRIGHTON UK



YOU SHOULD USE <INSERT
LIBRARY/Framework>, IT'S THE BESTESTEST!

PAUL LEWIS

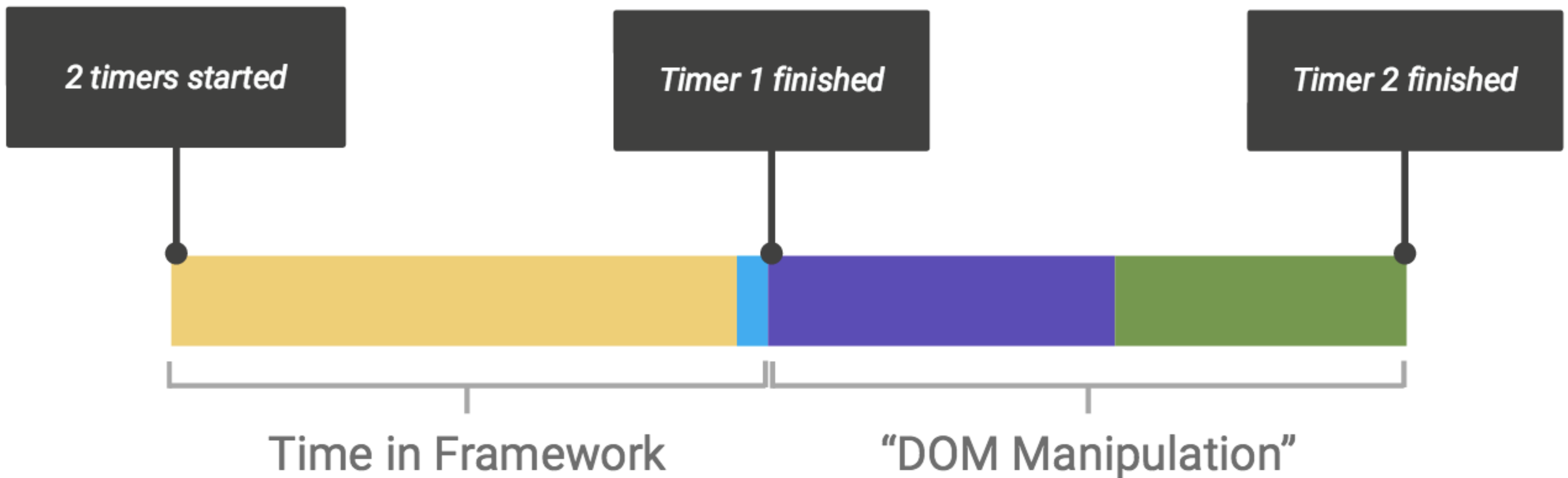
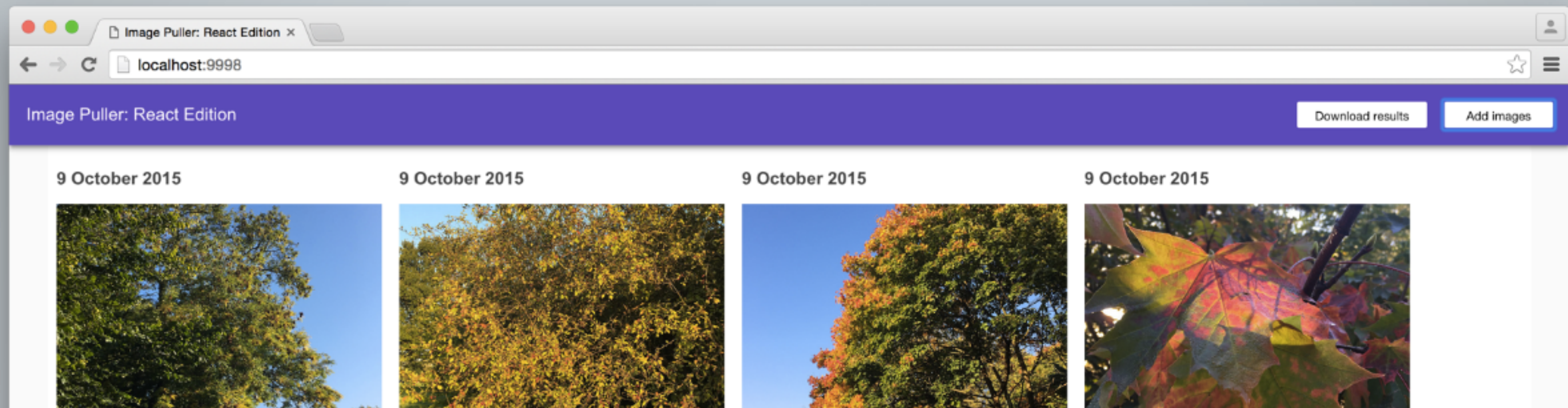
15:10 - 15:50

"You should use [insert library or framework here], it's the bestestest!" is pretty standard modern web development chatter. But how do you separate the hype from the reality? Let's have a look at how we can gather data to assess the fit of code to your projects.

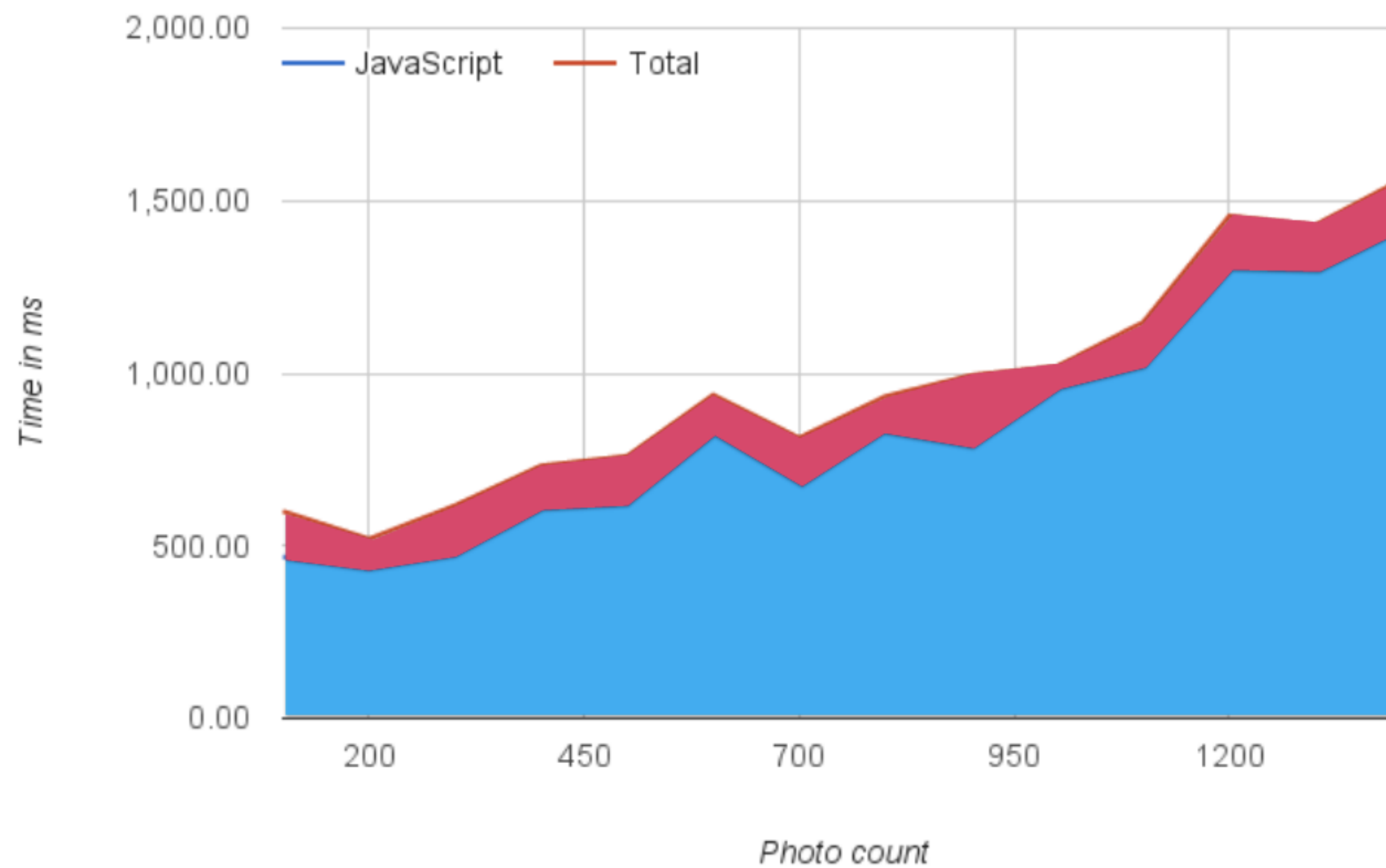
THE DOM IS SLOW?

“...the bottleneck is almost always the DOM mutation and not JS execution.

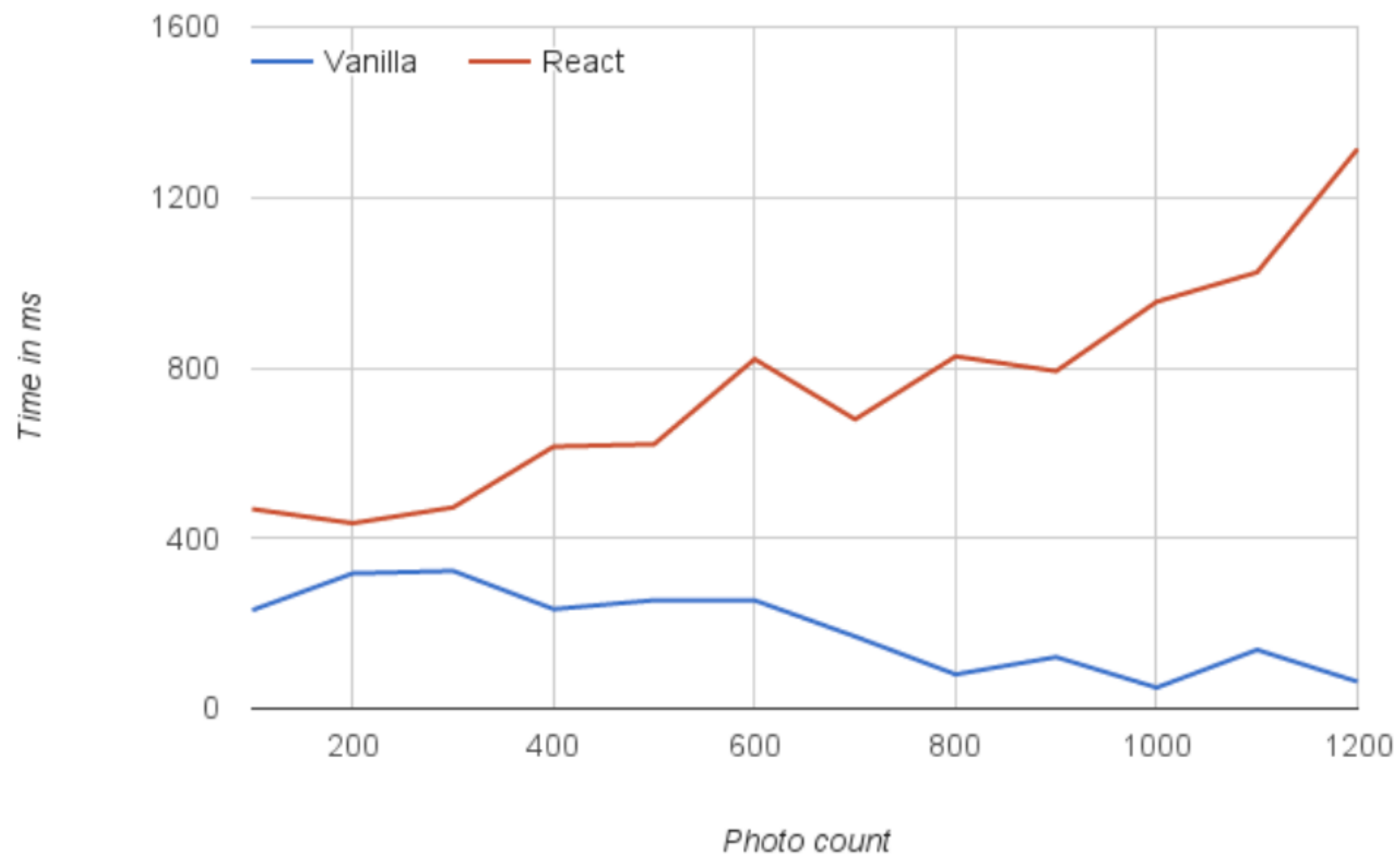
React docs



React - N5



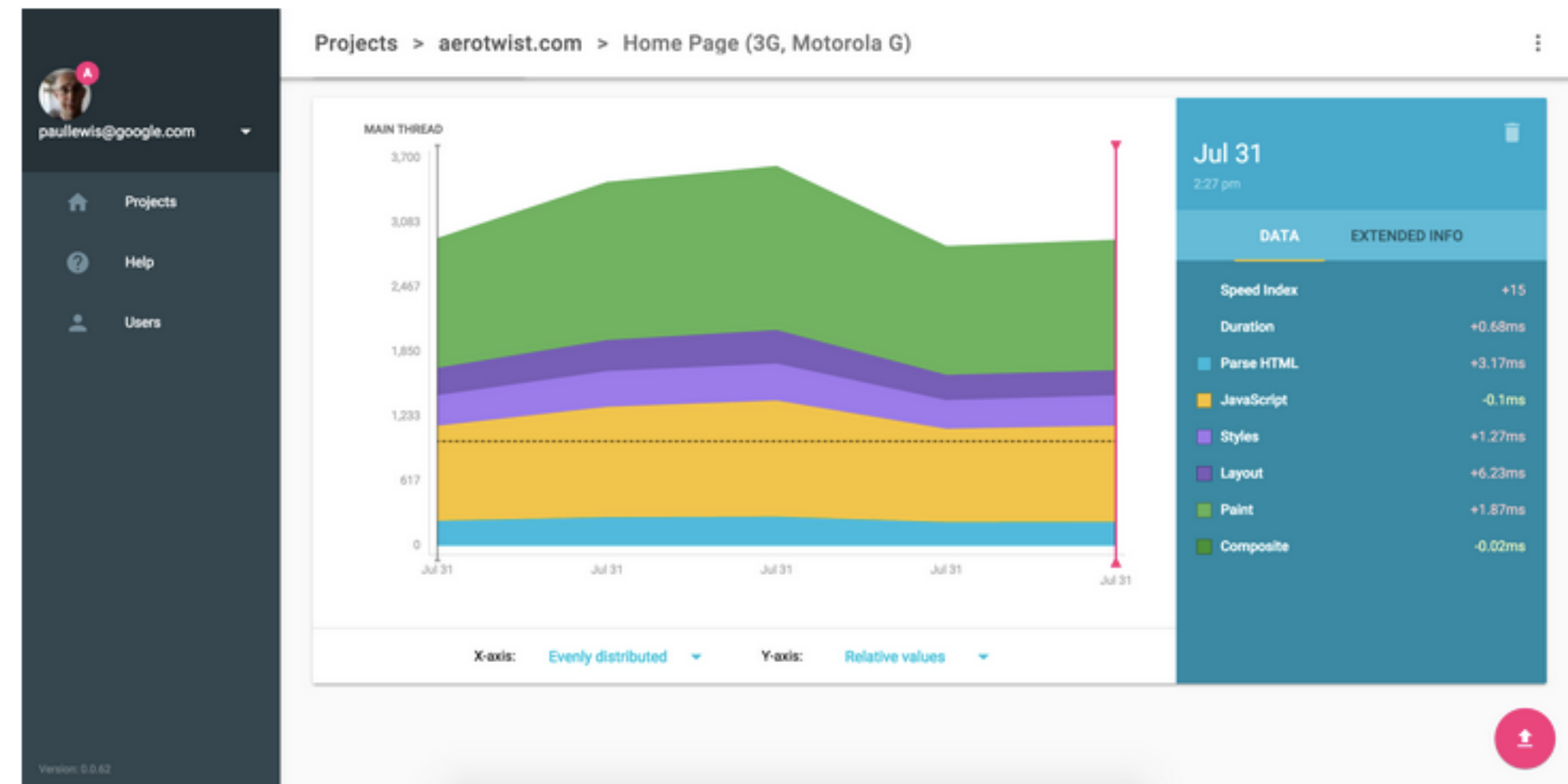
Vanilla vs React



ANALYSING BROWSER RESULTS...

Big Rig

Big Rig is an **experimental, proof-of-concept** system for generating and parsing Chrome's trace files. It has a web app dashboard that will allow you to track performance statistics over time, as well as a CLI version for integrating with other parts of your build system.



`npm install -g bigrig`

github.com/GoogleChrome/big-rig

github.com/GoogleChrome/node-big-rig

Framework	Size	Bootstrap time N5 ^{1,3}	Bootstrap time iPhone 5S ^{2,3}
Polymer v1.1.4	41KB ⁵	409ms	233ms
AngularJS v1.4.3	324KB	518ms	307ms
React v0.13.3	311KB	1,201ms	1,463ms
React v0.13.3 [JSX transformed via Babel] ⁴	162KB	509ms	282ms
Backbone v1.2.2 [inc. jQuery & Underscore]	139KB	248ms	168ms
Ember v1.10.0-beta.3	580KB	1,992ms	1,440ms
Vanilla	16KB	50ms	33ms

1. Tests done on a Nexus 5 running Chrome 47.

2. Tests done on an iPhone 5S running Safari 9.

3. All bootstrapping time includes handling initial todo list data.

4. JS Transformer stripped; JSX files transformed via Babel.

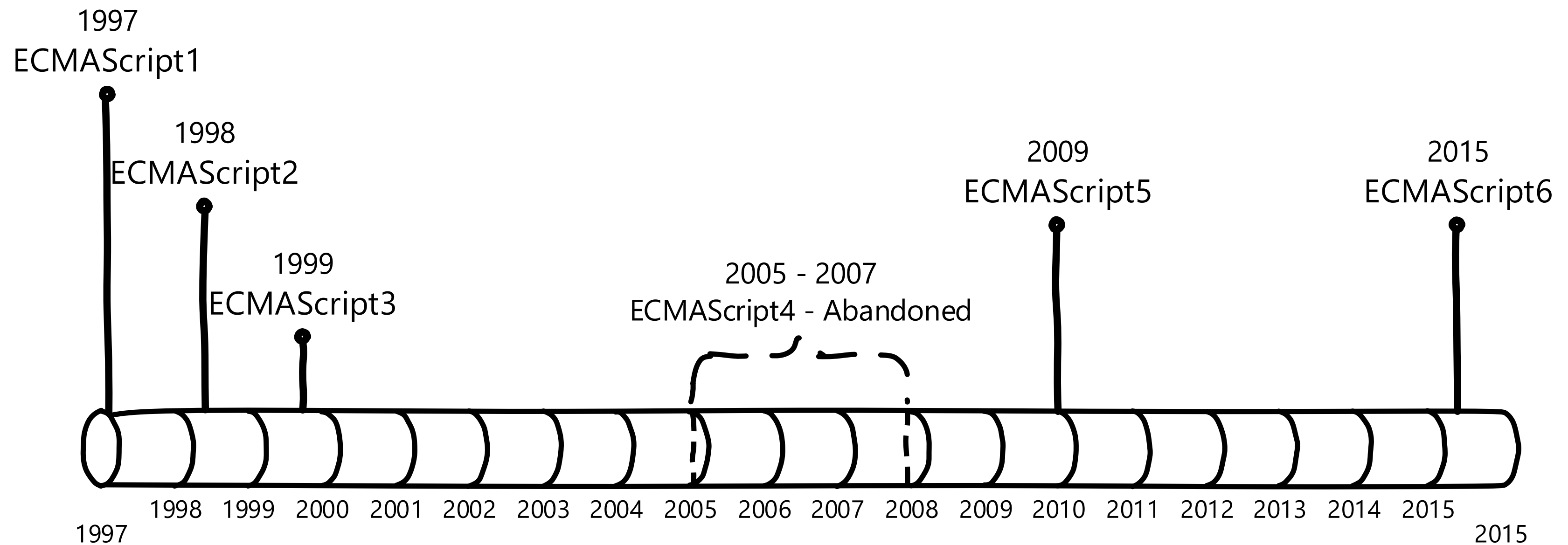
5. Excludes Web Components Polyfill (38KB)

WE USE CODE WE
DON'T UNDERSTAND
TO FIX ISSUES WE
DON'T HAVE...

- Learning libraries and frameworks beyond “hello world” costs time and money.
- Time you don't spend on looking at optimising your code
- In essence, we value developer convenience over user experience.

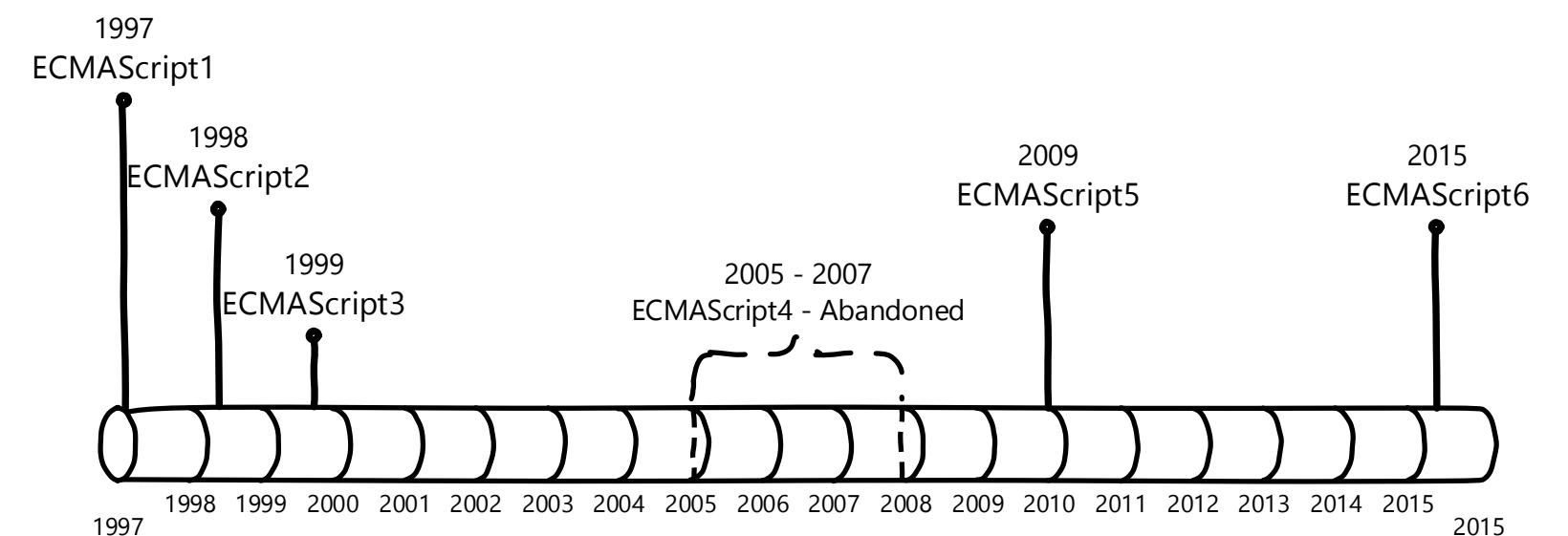
THE ES6 BUFFET...

JAVASCRIPT EVOLVES...



...NOW WE HAVE ES6!

- 5+ years since ES5 ratification
- Significant changes in 15+ years
- Backwards compatible
- Ratified June 2015



this IS COMPLICATED



Ölbaum
@oscherler



 Follow

JavaScript makes me want to flip the table and say “F***k this {***:”, but I can never be sure what “this” refers to.

RETWEETS
241

FAVORITES
220



11:03 AM - 30 Oct 2015

this IS COMPLICATED




Ölbaum
@oscherler

JavaScript makes me want to flip the table and say “F***k this :***:”, but I can never be sure what “this” refers to.

RETWEETS
241

FAVORITES
220




11:03 AM - 30 Oct 2015


```
function foo() {  
    setTimeout(function() {  
        console.log('id:', this.id);  
    }, 100);  
}  
foo.call( { id: 42 } ); // id: undefined
```


blog.getify.com/arrow-this/

this IS COMPLICATED



Ölbaum
@oscherler




 Follow

JavaScript makes me want to flip the table and say “F***k this :***:”, but I can never be sure what “this” refers to.

RETWEETS
241

FAVORITES
220



11:03 AM - 30 Oct 2015

```
function foo() {  
  var self = this;  
  setTimeout(function() {  
    console.log("id:", self.id);  
  }, 100);  
}  
  
foo.call( { id: 42 } ); // id: 42
```


this IS COMPLICATED




Ölbaum
@oscherler

JavaScript makes me want to flip the table and say “F***k this :***:”, but I can never be sure what “this” refers to.

RETWEETS
241


FAVORITES
220




11:03 AM - 30 Oct 2015


```
function foo() {  
    setTimeout(function() {  
        console.log("id:", this.id);  
    }.bind(this), 100);  
}  
  
foo.call( { id: 42 } ); // id: 42
```

this IS COMPLICATED



Ölbaum
@oscherler




 Follow

JavaScript makes me want to flip the table and say “F***k this :***:”, but I can never be sure what “this” refers to.

RETWEETS
241

FAVORITES
220



11:03 AM - 30 Oct 2015

```
function foo() {  
  setTimeout( () => {  
    console.log("id:", this.id);  
  }, 100);  
}  
  
foo.call( { id: 42 } ); // id: 42
```


SAVING KEYSTROKES

```
var a = [  
  "Hydrogen",  
  "Helium",  
  "Lithium",  
  "Beryllium"  
];  
  
var a2 = a.map(function(s){ return s.length });  
  
var a3 = a.map( s => s.length );
```

ES6 COMES WITH SO
MUCH GOODNESS,
TECHNICALLY IT HAS
TO BE FATTENING...

- Arrow functions as a short-hand version of an anonymous function.
- Block-level scope using `let` instead of `var` makes variables scoped to a block (`if`, `for`, `while`, etc.)
- Classes to encapsulate and extend code.
- Constants using the `const` keyword.
- Default parameters for functions like `foo(bar = 3, baz = 2)`
- Destructuring to assign values from arrays or objects into variables.
- Generators that create iterators using `function*` and the `yield` keyword.
- `Map`, a Dictionary type object that can be used to store key/value pairs. and `Set` as a collection object to store a list of data values.
- Modules as a way of organizing and loading code.
- Promises for async operations avoiding callback hell
- Rest parameters instead of using arguments to access functions arguments.
- Template Strings to build up string values including multi-line strings.

ALL OF THESE PARTS HAVE DIFFERENT AUDIENCES

Syntactic Sugar

Arrow functions
Enhanced object literals
Template strings
Destructuring
Rest, Spread, Default
String, Math, Number,
Object, RegExp APIs

Scalable Apps

let, const & block-
scoped bindings
Classes
Promises
Iterators
Generators
Typed arrays
Modules

Library Builders

map, set & weakmap
__proto__
Proxies
Symbols
Sub-classable built-ins

SUPPORT IS ENCOURAGING (EDGE, FIREFOX, CHROME, SAFARI (ON 9))

COMPAT

ES

ECMAScript

5

6

7

intl

non-standard

compatibility table

14

by kangax

Gratipay

& webbedspace

Please note that *some of these tests* represent existence, not functionality or full conformance.

Sort by number of features? ☐

Show obsolete platforms? ☐

Show unstable platforms? ☒

V8

SpiderMonkey

JavaScriptCore

Chakra

Carakan

KJS

Other

Minor difference (1 point)

Useful feature (2 point)

Significant feature (4 points)

Landmark feature (8 points)

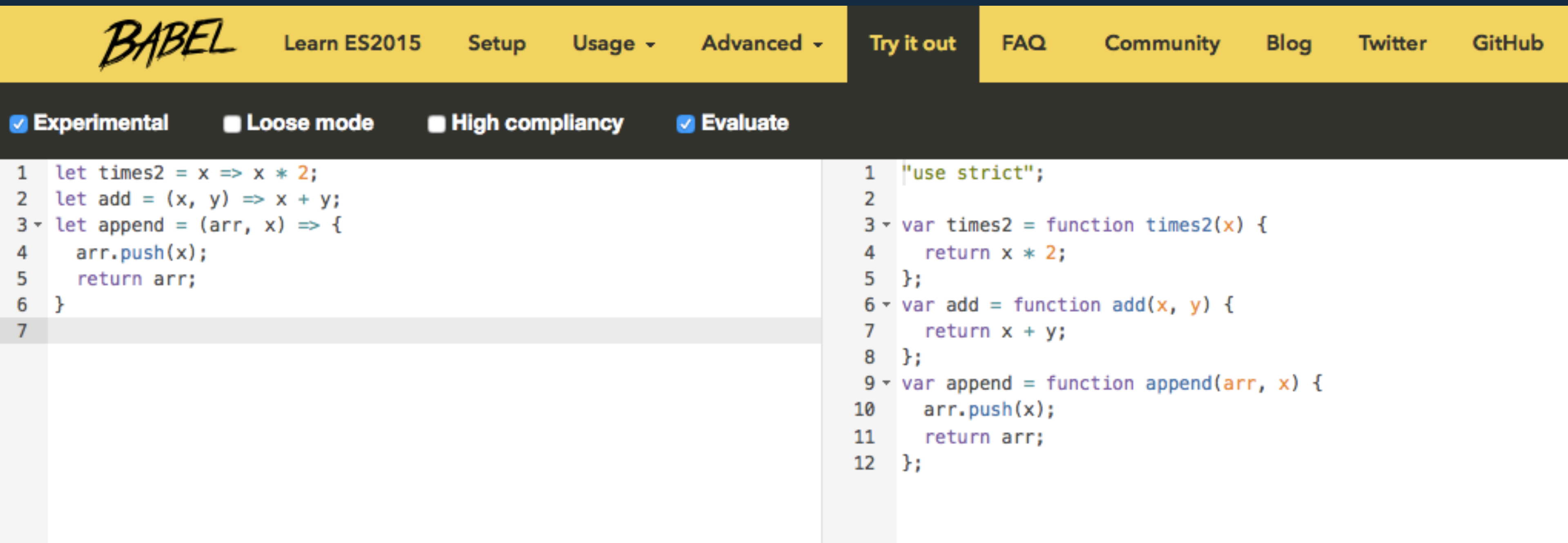
		Compilers/polyfills						Desktop browsers													Servers/runtimes				Mobile										
	70%		59%	71%	30%	18%	52%	17%	10%	20%	67%	80%	66%	70%	71%	58%	63%	63%	12%	21%	54%	66%	13%	11%	17%	53%	58%	12%	17%	29%	12%	21%	54%		
Feature name	Current browser	Traceur	Babel + core-js ^[1]	Closure	JSX ^[2]	Type-Script + core-js	es6-shim		IE 10	IE 11	Edge 12 ^[3]	Edge 13 ^[3]	FF 38 ESR	FF 41	FF 42	CH 45, OP 32 ^[4]	CH 46, OP 33 ^[4]	CH 47, OP 34 ^[4]	SF 6.1, SF 7	SF 7.1, SF 8	SF 9	WK	KQ 4.14 ^[5]	PJS	Node 0.12 ^[6]	Node 4.0 ^[6]	Echo JS	AN 4.4	AN 5.0	AN 5.1	iOS 7	iOS 8	iOS 9		
Optimisation																																			
<div><div></div><div>proper tail calls (tail call optimisation)</div></div>	0/2	0/2	1/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	
Syntax																																			
<div><div></div><div>default function parameters</div></div>	3/7	4/7	6/7	4/7	0/7	4/7	0/7	0/7	0/7	0/7	0/7	0/7	3/7	3/7	3/7	0/7	0/7	0/7	0/7	0/7	0/7	7/7	0/7	0/7	0/7	0/7	4/7	0/7	0/7	0/7	0/7	0/7	0/7	0/7	
<div><div></div><div>rest parameters</div></div>	4/5	4/5	4/5	2/5	3/5	3/5	0/5	0/5	0/5	0/5	5/5	5/5	4/5	4/5	4/5	0/5	0/5	0/5	0/5	0/5	0/5	0/5	0/5	0/5	0/5	0/5	2/5	0/5	0/5	0/5	0/5	0/5	0/5	0/5	
<div><div></div><div>spread (...) operator</div></div>	15/15	15/15	13/15	3/15	2/15	4/15	0/15	0/15	0/15	0/15	12/15	12/15	15/15	15/15	15/15	0/15	15/15	15/15	0/15	5/15	9/15	9/15	0/15	0/15	0/15	0/15	13/15	0/15	0/15	0/15	0/15	5/15	9/15	9/15	
<div><div></div><div>object literal extensions</div></div>	6/6	6/6	6/6	4/6	5/6	6/6	0/6	0/6	0/6	0/6	6/6	6/6	6/6	6/6	6/6	6/6	6/6	6/6	0/6	1/6	5/6	6/6	0/6	0/6	0/6	6/6	4/6	0/6	0/6	0/6	0/6	0/6	1/6	5/6	
<div><div></div><div>for...of loops</div></div>	7/9	9/9	9/9	6/9	2/9	3/9	0/9	0/9	0/9	0/9	6/9	6/9	7/9	7/9	7/9	7/9	7/9	7/9	0/9	2/9	8/9	8/9	0/9	0/9	7/9	7/9	6/9	0/9	0/9	7/9	0/9	2/9	8/9	8/9	
<div><div></div><div>octal and binary literals</div></div>	4/4	2/4	4/4	2/4	0/4	4/4	2/4	0/4	0/4	0/4	4/4	4/4	4/4	4/4	4/4	4/4	4/4	4/4	0/4	0/4	4/4	4/4	0/4	0/4	0/4	4/4	4/4	0/4	0/4	0/4	0/4	0/4	0/4	4/4	
<div><div></div><div>template strings</div></div>	5/5	4/5	4/5	3/5	4/5	3/5	0/5	0/5	0/5	0/5	4/5	4/5	5/5	5/5	5/5	5/5	5/5	5/5	0/5	0/5	5/5	5/5	0/5	0/5	0/5	5/5	5/5	0/5	0/5	0/5	0/5	0/5	0/5	5/5	
<div><div></div><div>RegExp "y" and "u" flags</div></div>	2/4	2/4	2/4	0/4	0/4	0/4	0/4	0/4	0/4	0/4	2/4	2/4	2/4	2/4	2/4	0/4	0/4	0/4	0/4	0/4	0/4	0/4	0/4	0/4	0/4	0/4	2/4	0/4	0/4	0/4	0/4	0/4	0/4	0/4	
<div><div></div><div>destructuring</div></div>	28/36	30/36	33/36	20/36	15/36	25/36	0/36	0/36	0/36	0/36	0/36	0/36	27/36	28/36	28/36	0/36	0/36	0/36	0/36	18/36	30/36	32/36	0/36	0/36	0/36	0/36	16/36	0/36	0/36	0/36	0/36	0/36	18/36	30/36	
<div><div></div><div>Unicode code point escapes</div></div>	1/2	1/2	1/2	1/2	0/2	1/2	0/2	0/2	0/2	0/2	2/2	2/2	0/2	1/2	1/2	2/2	2/2	2/2	0/2	0/2	2/2	2/2	0/2	0/2	0/2	2/2	1/2	0/2	0/2	0/2	0/2	0/2	0/2	2/2	
<div><div></div><div>new.target</div></div>	2/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	2/2	0/2	2/2	2/2	0/2	2/2	2/2	0/2	0/2	0/2	1/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	0/2	
Bindings																																			
<div><div></div><div>const</div></div>	8/8	6/8	6/8	6/8	0/8	8/8	0/8	0/8	8/8	8/8	8/8	8/8	8/8	8/8	8/8	5/8	5/8	5/8	1/8	1/8	1/8	8/8	2/8	1/8	1/8	5/8	6/8	1/8	1/8	1/8	1/8	1/8	1/8	1/8	
<div><div></div><div>let</div></div>	0/10	8/10	8/10	8/10	0/10	8/10	0/10	0/10	8/10	8/10	8/10	8/10	0/10	0/10	0/10	5/10	5/10	5/10	0/10	0/10	0/10	10/10	0/10	0/10	0/10	5/10	10/10	0/10	0/10	0/10	0/10	0/10	0/10	0/10	
<div><div></div><div>block-level function declaration^[13]</div></div>	No	Yes	Yes	Yes	No	No	No	No	Yes	Yes	Yes	Yes	No	No	No	Yes	Yes	Yes	No	No	No	No	No	No	Flag	Yes	No	No	No	No	No	No	No	No	No
Functions																																			
<div><div></div><div>arrow functions</div></div>	10/12	10/12	9/12	9/12	7/12	8/12	0/12	0/12	0/12	0/12	9/12	11/12	8/12	10/12	10/12	9/12	10/12	11/12	0/12	0/12	0/12	9/12	0/12	0/12	0/12	9/12	7/12	0/12	0/12	0/12	0/12	0/12	0/12	0/12	
<div><div></div><div>class</div></div>	0/23	16/23	19/23	9/23	15/23	15/23	0/23	0/23	0/23	0/23	0/23	23/23	0/23	0/23	0/23	0/23	0/23	0/23	0/23	0/23	15/23	22/23	0/23	0/23	0/23	0/23	16/23	0/23	0/23	0/23	0/23	0/23	0/23	15/23	
<div><div></div><div>super</div></div>	0/8	7/8	6/8	4/8	7/8	6/8	0/8	0/8	0/8	0/8	0/8	8/8	0/8	0/8	0/8	0/8	0/8	0/8	0/8	0/8	6/8	7/8	0/8	0/8	0/8	0/8	6/8	0/8	0/8	0/8	0/8	0/8	0/8	6/8	

THE PROBLEM: FOR
NON-SUPPORTING
ENGINES, ES6
FEATURES ARE
SYNTAX ERRORS...



READY.

THE SOLUTION: TRANSPILING INTO ES5...



The screenshot shows the Babel website interface. The top navigation bar includes the Babel logo, links to 'Learn ES2015', 'Setup', 'Usage', 'Advanced', 'Try it out', 'FAQ', 'Community', 'Blog', 'Twitter', and 'GitHub'. Below the navigation bar, there are four checkboxes: 'Experimental' (checked), 'Loose mode' (unchecked), 'High compliancy' (unchecked), and 'Evaluate' (checked). The main content area displays a side-by-side comparison of code. On the left, the input ES6 code is shown with line numbers 1 through 7. On the right, the output ES5 code is shown with line numbers 1 through 12. The ES6 code defines three variables: 'times2' as an arrow function, 'add' as an arrow function, and 'append' as a function that pushes an element to an array and returns it. The ES5 code is the transpiled version, using 'use strict', and converting the arrow functions into regular function declarations.

```
1 let times2 = x => x * 2;
2 let add = (x, y) => x + y;
3 let append = (arr, x) => {
4   arr.push(x);
5   return arr;
6 }
7
```

```
1 "use strict";
2
3 var times2 = function times2(x) {
4   return x * 2;
5 };
6 var add = function add(x, y) {
7   return x + y;
8 };
9 var append = function append(arr, x) {
10   arr.push(x);
11   return arr;
12 };
```


- Converts ES6 to older versions on the server or the client
- In use by Facebook and many others
- Used in editors and tool chains

<https://babeljs.io>


THE PROBLEMS WITH TRANSPILING:

- ✗ Extra step between writing code and running it in the browser.
- ✗ We don't run or debug the code we write.
- ✗ We hope the transpiler creates efficient code
- ✗ We create a lot of code
- ✗ Browsers that support ES6 will never get any.

TRANSPILED CODE...



Curtis Ovid Poe
@OvidPerl

 [Follow](#)

Reading JS code that ends with:

```
},{}],7:[function(require,module,exports){  
},{}],{},[1])(1)  
});
```


Perl has a bad punctuation reputation?

RETWEETS

35

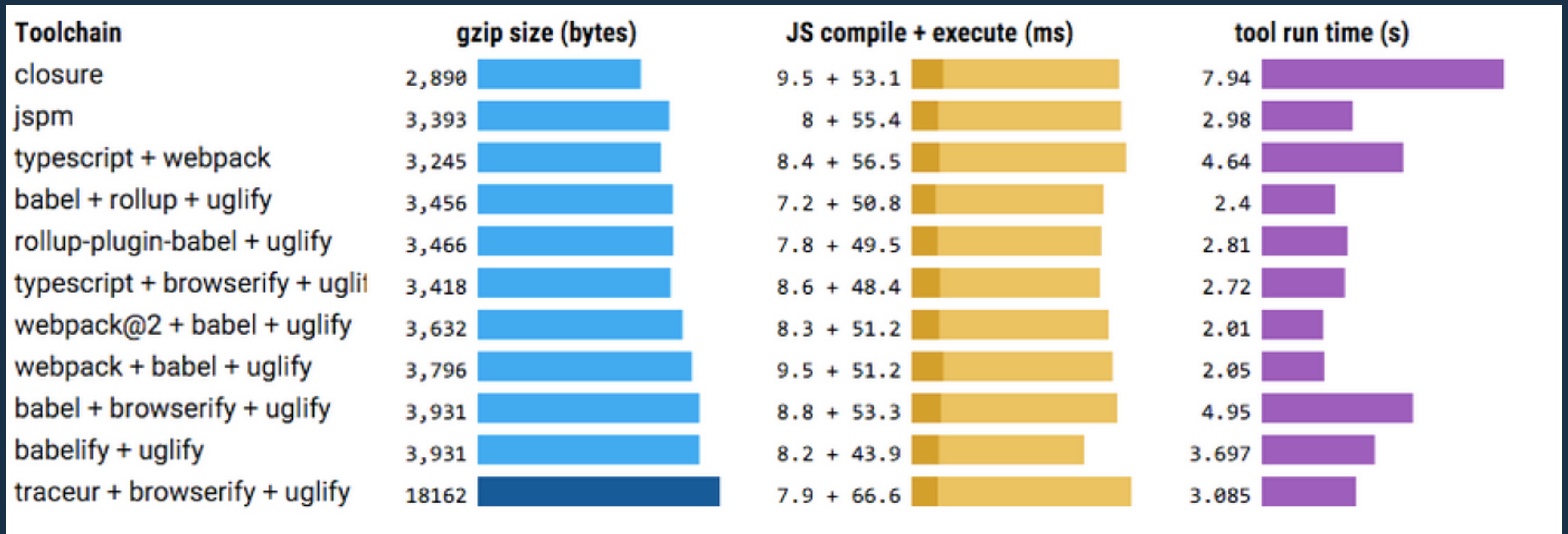
LIKES

23



9:09 PM - 6 Feb 2016

PICK YOUR TRANSPIRATION TOOLCHAIN...



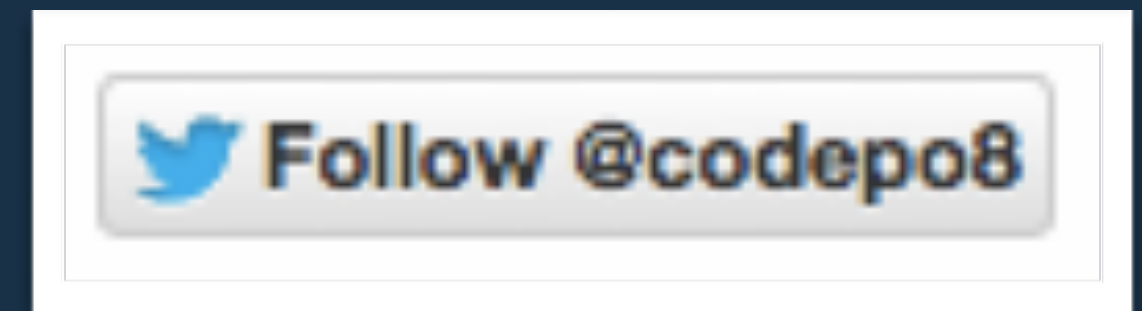
THE ES6 CONUNDRUM...

- We can't use it safely in the wild
- We can use TypeScript or transpile it
- We can feature test for it, but that can get complex quickly
- Browsers that support it, will not get any ES6 that way (but can use it internally)
- The performance is bad right now (which is a normal thing). To improve this, we need ES6 to be used in the wild...

STANDARDS AND INTEROP

WHAT DOES THIS CODE DO?

```
<!DOCTYPE html><html data-scribe="page:button"><head><meta charset="utf-8"><title>
Twitter Follow Button</title><base target="_blank"><link rel="dns-prefetch" href="
//twitter.com"><link rel="profile" href="http://microformats.org/profile/hcard"><
style type="text/css">html{padding:0;margin:0;font:normal normal normal 11px/18
px 'Helvetica Neue',Arial,sans-serif;color:#333;-webkit-user-select:none;-moz-
user-select:none;-ms-user-select:none;user-select:none}body{padding:0;margin:0;
background:transparent}a{text-decoration:none;outline:0}body.rtl{direction:rtl}
#widget{display:inline-block;overflow:hidden;text-align:left;white-space:nowrap
}.btn-o,.count-o,.btn,.btn .label,#count{display:inline-block;vertical-align:
top;zoom:1}.btn-o{max-width:100%}.btn{position:relative;height:18px;font-weight
:bold;color:#333;text-shadow:0 1px 0 rgba(255,255,255,0.5);cursor:pointer;
background-color:#eee;background-image:-webkit-linear-gradient(#fff,#ddd);back-
ground-image:linear-gradient(#fff,#dedede);border:#ccc solid 1px;border-radius:3px}.btn:focus,.btn:hover,.btn:active{background-color:#d9d9d9;background-image:-webkit-linear-gradient(#f8f8f8,#d9d9d9);background-image:linear-gradient(#f8f8f8,#d9d9d9);border-color:#bbb;box-shadow:none}.btn:active{box-shadow:inset 0 3px 5px rgba(0,0,0,0.1)}.btn i{position:absolute;right:50%;left:2px;width:16px;height:13px;margin-top:-5px;background:transparent;background-image:url(data:image/png;base64,iVBORw0KGgoAAAANSUUhEUgAAABq13MpAAAGcklEQVRYw+2YXUyTVxjHz4vJLiZGd7MtXi2LkZtdELM7lyzOG7Nk2RJvl8tFWqCFEGCAfJRC+SyltqWFgnwULIKAWB3y0VrAttQWC1ZC0i6ePc8LL74tVD6ly2KTf87J6TAAwEhMIi8h1MViEs0Jlai+we5oJFiGCX3D9X+fmKTma/f/rzkRLX5fzkmNPhLVaW2D01Ifv
Column 418
```



```
<a href="https://twitter.com/codepo8"
class="twitter-follow-button"
data-show-count="false">Follow @codepo8</a>

<script>!function(d,s,id) {
    var js,fjs=d.getElementsByTagName(s)[0],
        p=/^http:/.test(d.location)?'http':'https';
    if(!d.getElementById(id)){
        js=d.createElement(s);js.id=id;
        js.src=p+'://platform.twitter.com/widgets.js';
        fjs.parentNode.insertBefore(js,fjs);
    }
}(document, 'script', 'twitter-wjs');
</script>
```


REPLACING BUILT-IN FUNCTIONALITY FOR THE SAKE OF CONTROL...



Heydon
@heydonworks



Following

This is your life now. I hope that's okay with you.

```
/* components/submit-button.jsx */
import { Component } from 'react';
import styles from './submit-button.css';

export default class SubmitButton extends Component {
  render() {
    let className, text = "Submit"
    if (this.props.store.submissionInProgress) {
      className = styles.inProgress
      text = "Processing..."
    } else if (this.props.store.errorOccurred) {
      className = styles.error
    } else if (!this.props.form.valid) {
      className = styles.disabled
    } else {
      className = styles.normal
    }
    return <button className={className}>{text}</button>
  }
}
```

RETWEETS

9

FAVORITES

10




SIMPLY BECAUSE WE CAN FIX ANYTHING WITH
JAVASCRIPT DOES NOT MEAN WE SHOULD!



≠



THE COMPATIBILITY RACE...



Web Standards

THE VICIOUS INNOVATION CYCLE...



Christian Heilmann

@codepo8

Developers don't use features until browsers support them. Browsers don't add new features unless there is demand and use in the wild. Hmm...

RETWEETS

32

FAVORITES

36



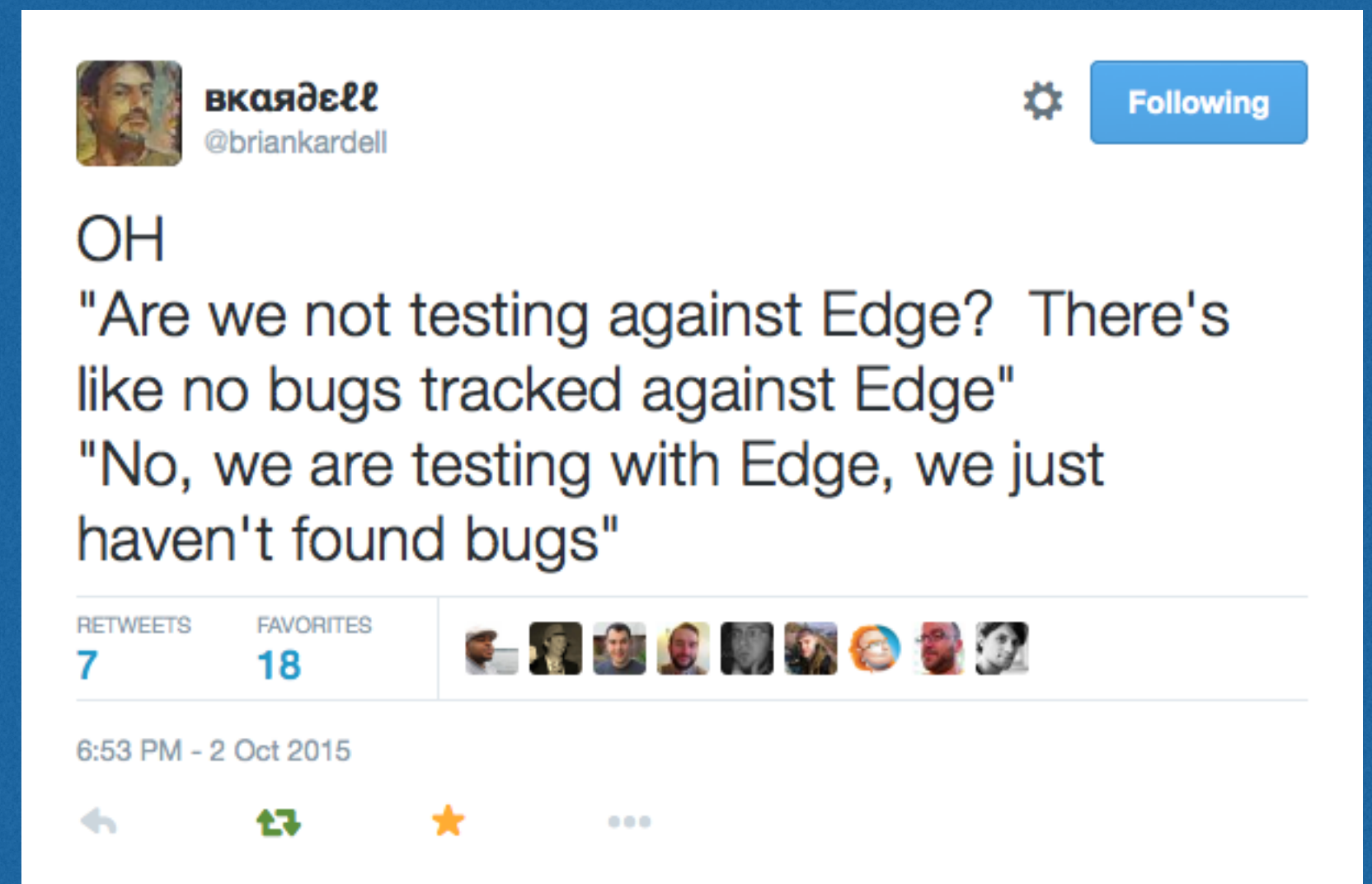
10:06 PM - 25 Oct 2015

📍 Bellevue, WA

OUT OF THE BOX BROWSER
DO NOT SUCK ANY LONGER!

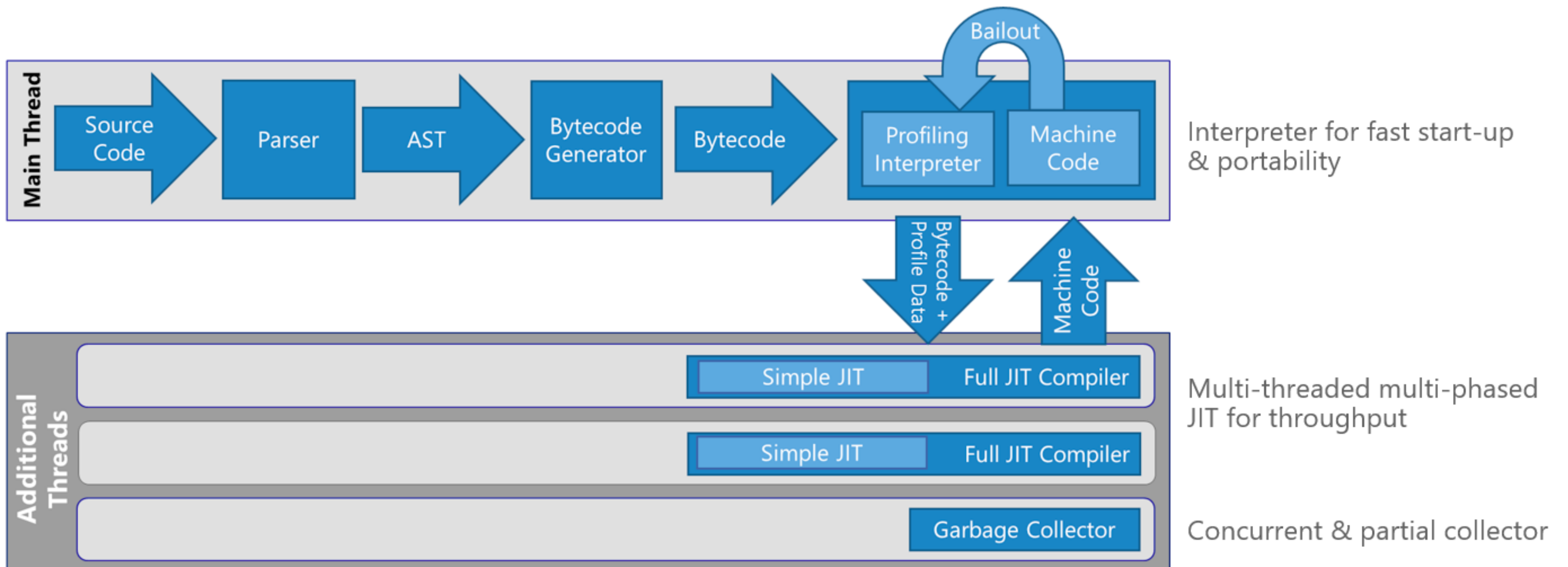


COMPATIBILITY IS ON
EVERY BROWSER
MAKER'S RADAR...



BREAKING MONOPOLIES

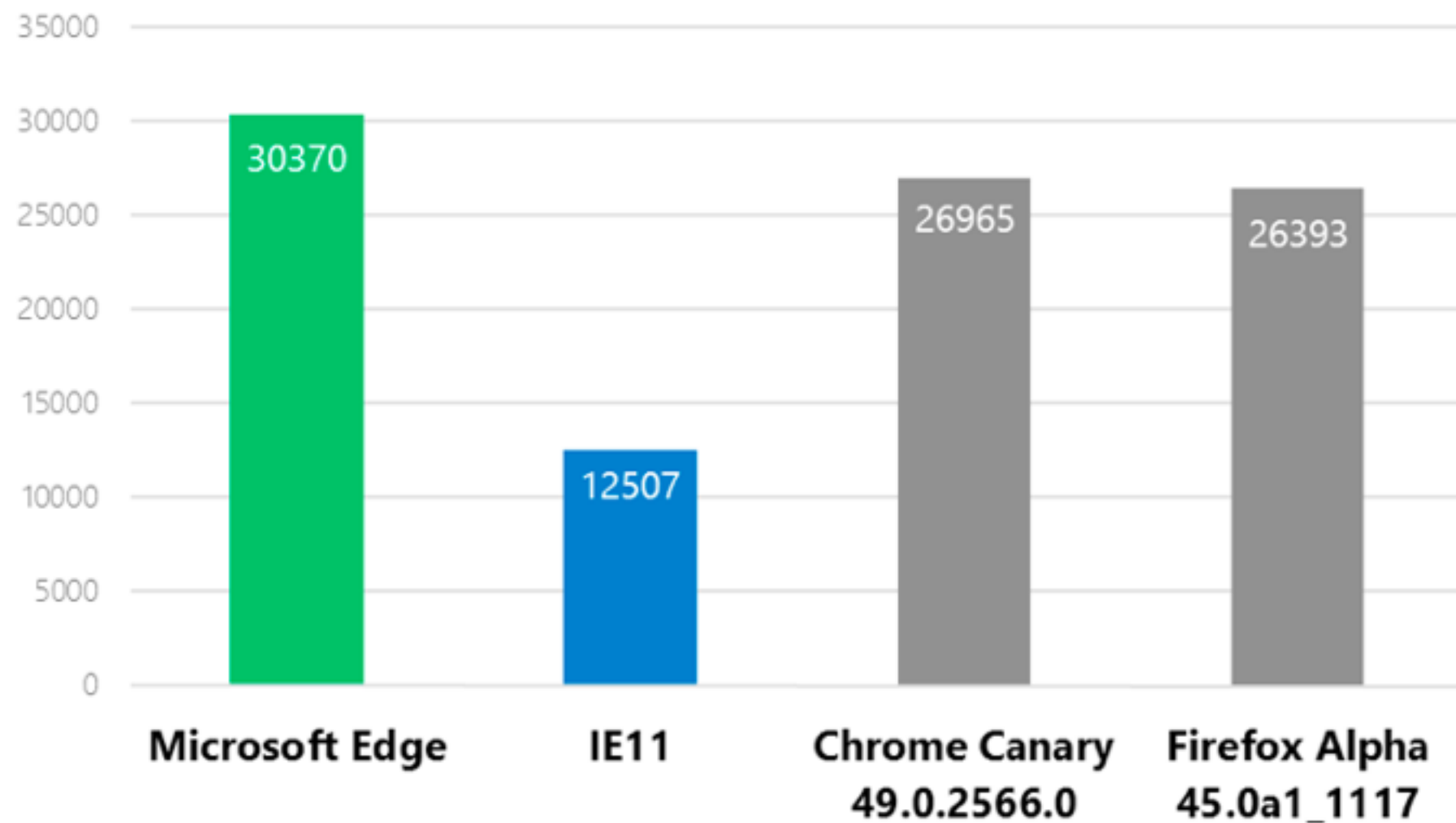
NEW BROWSER, NEW JS ENGINE



SPEED COMPARISONS...

OCTANE 2.0

(higher is better)

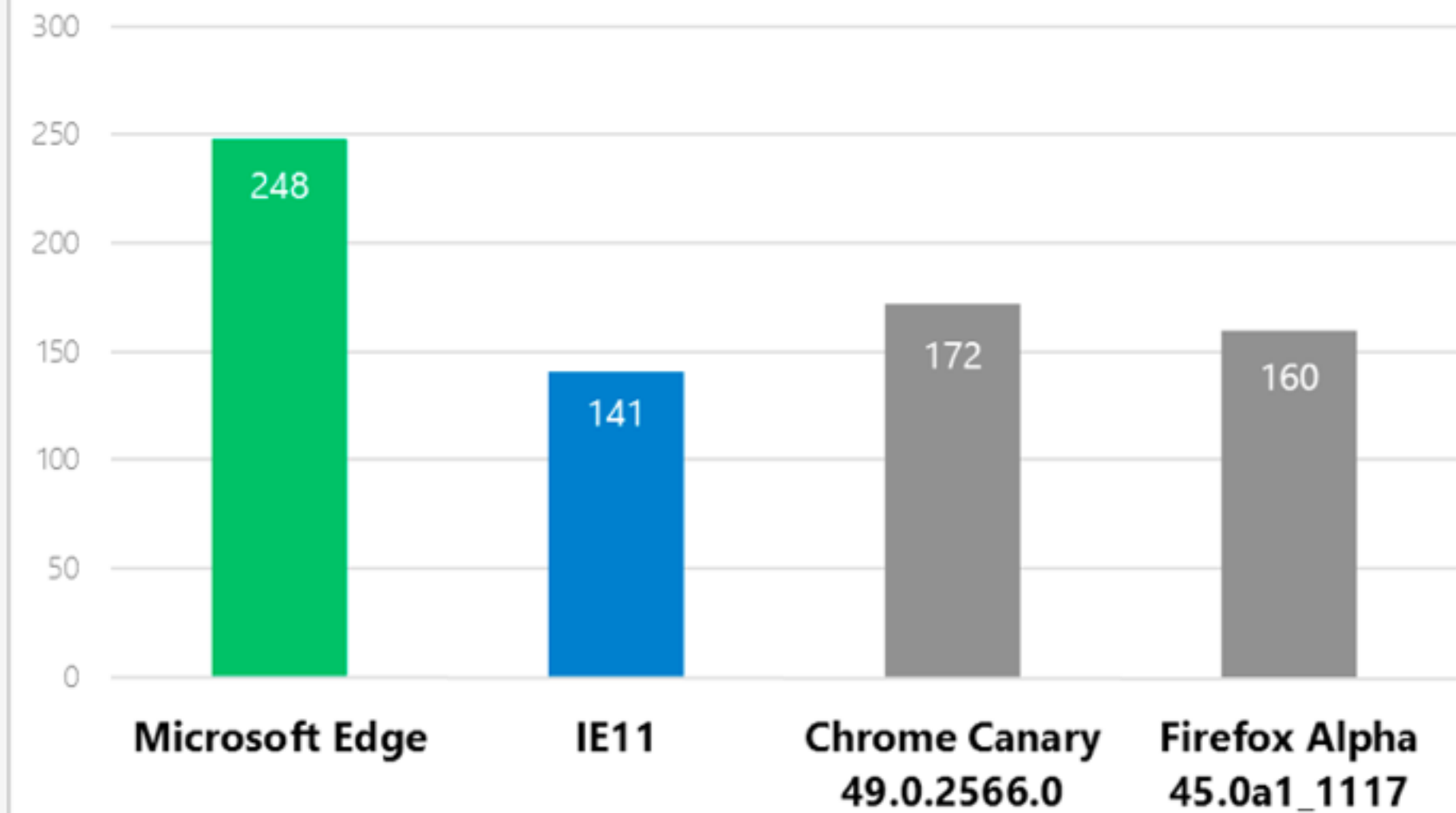


Owned by: Google

Edge leads closest competitor by 10+%

JET STREAM

(higher is better)



Owned by: Apple

Edge leads closest competitor by 40+%

SURPRISES HAPPEN...



meatcovered skeleton

@izs

+ Follow

42% of **@nodejs** users use Windows as their desktop env. More than any single *nix. If you want devs using your code, this matters.

RETWEETS

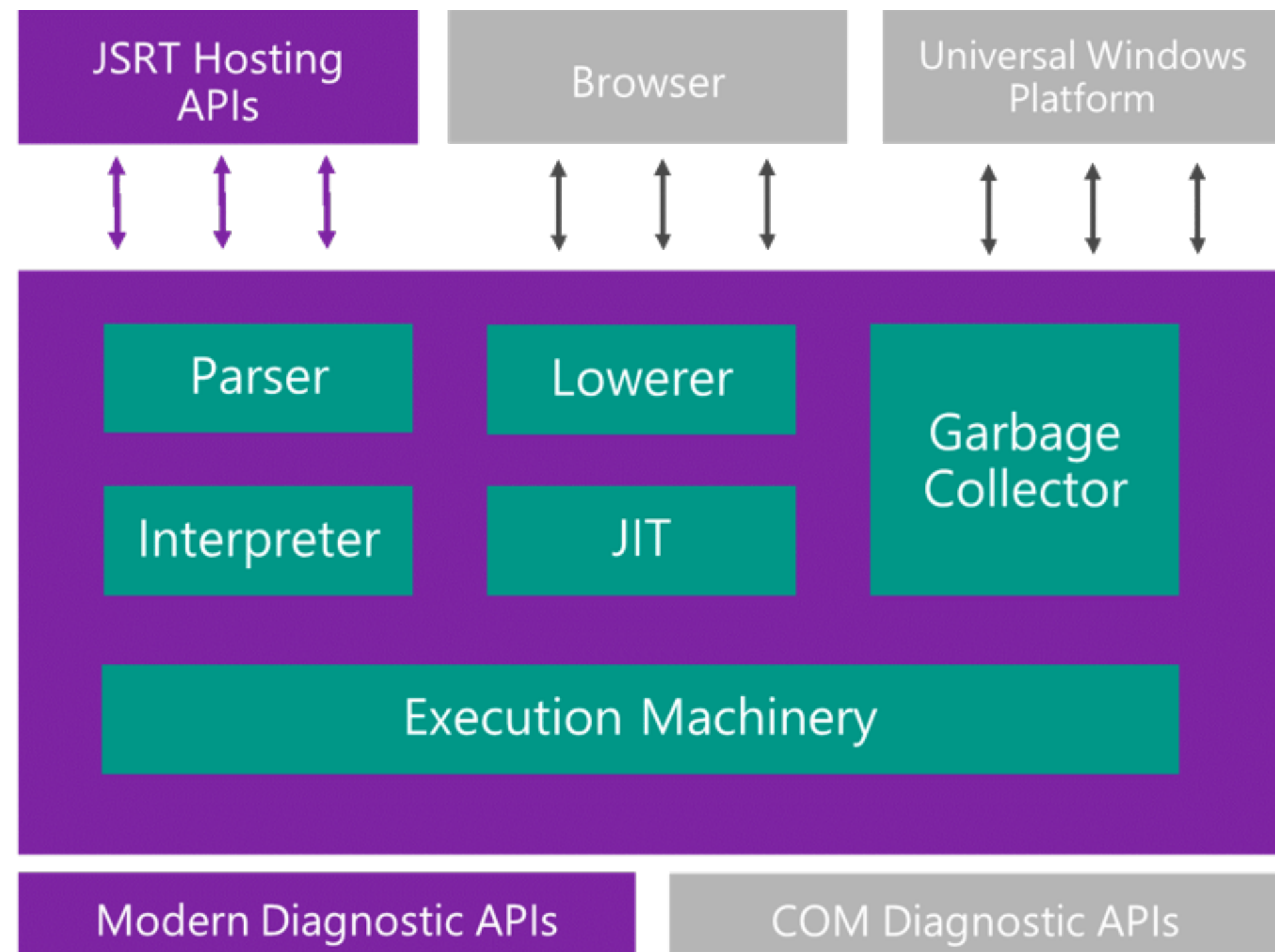
20

LIKES

15



CHAKRACORE VS CHAKRA



LEGEND:  ChakraCore

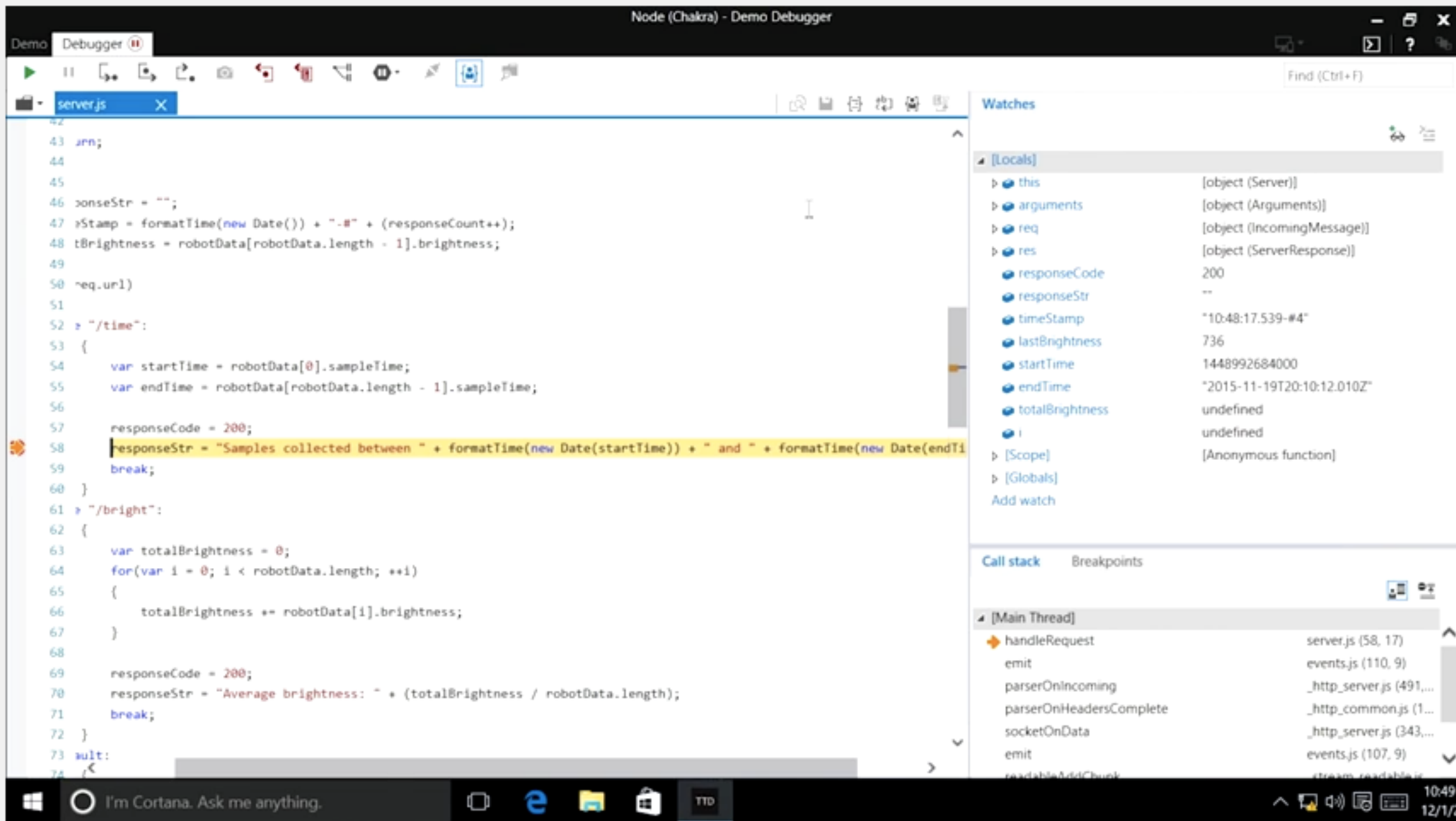
 +  Chakra

HTTPS://



.WTF

TIME TRAVEL DEBUGGING



The screenshot shows the Node.js (Chakra) Demo Debugger interface. The main window displays the source code of `server.js`. The code defines a `time` endpoint that calculates the time taken to process robot data. The debugger is paused at line 58, where the `responseStr` is being constructed. The `Watches` panel on the right shows the current state of variables, including `this`, `arguments`, `req`, `res`, `responseCode`, `responseStr`, `timeStamp`, `lastBrightness`, `startTime`, `endTime`, `totalBrightness`, and `i`. The `Call stack` panel shows the current call stack, including `handleRequest`, `emit`, `parserOnIncoming`, `parserOnHeadersComplete`, `socketOnData`, and `emit`.

ChakraCore OSS

Microsoft Visual Studio

Subscribe 30,656

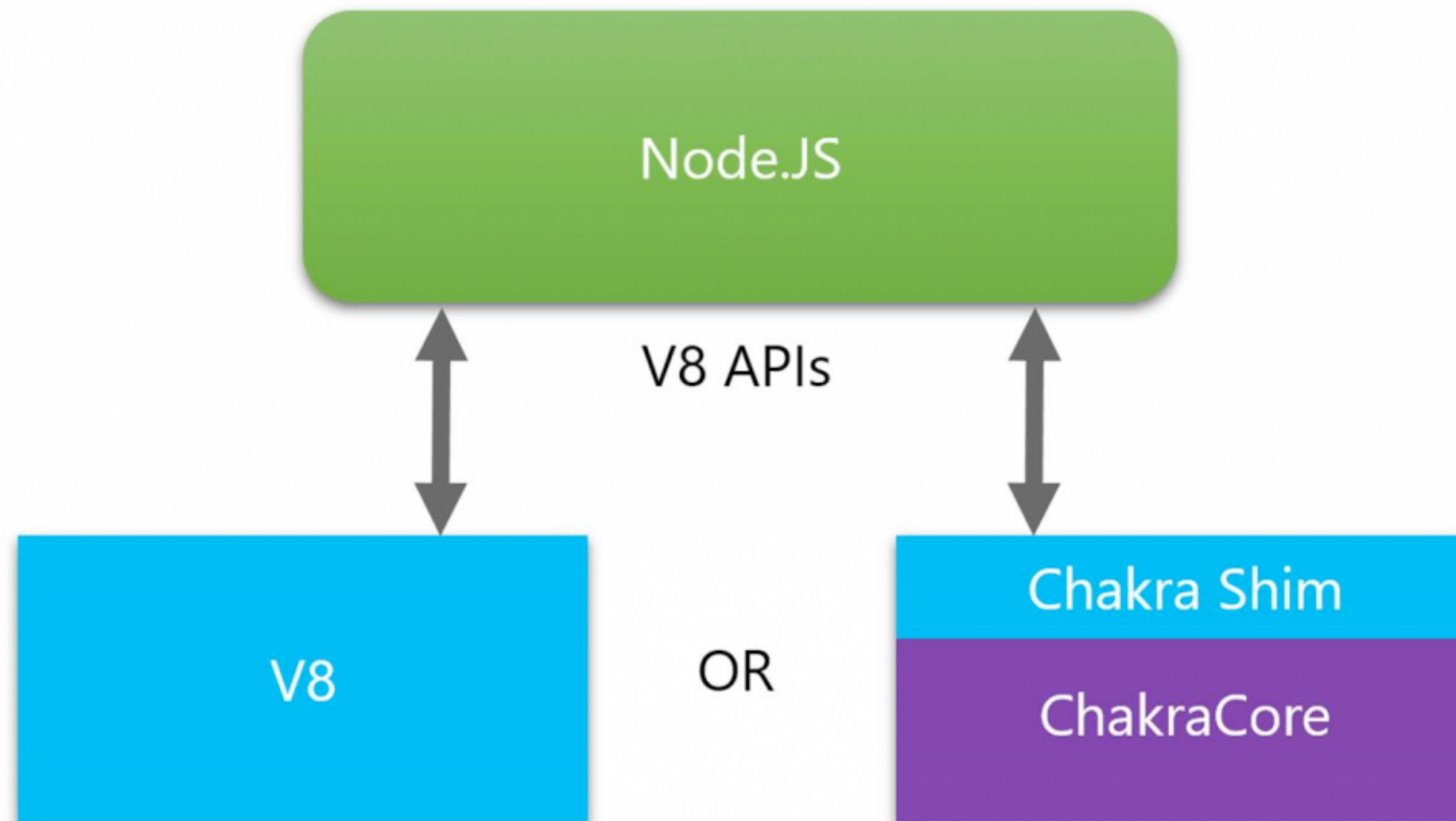
14,581

+ Add to Share ... More

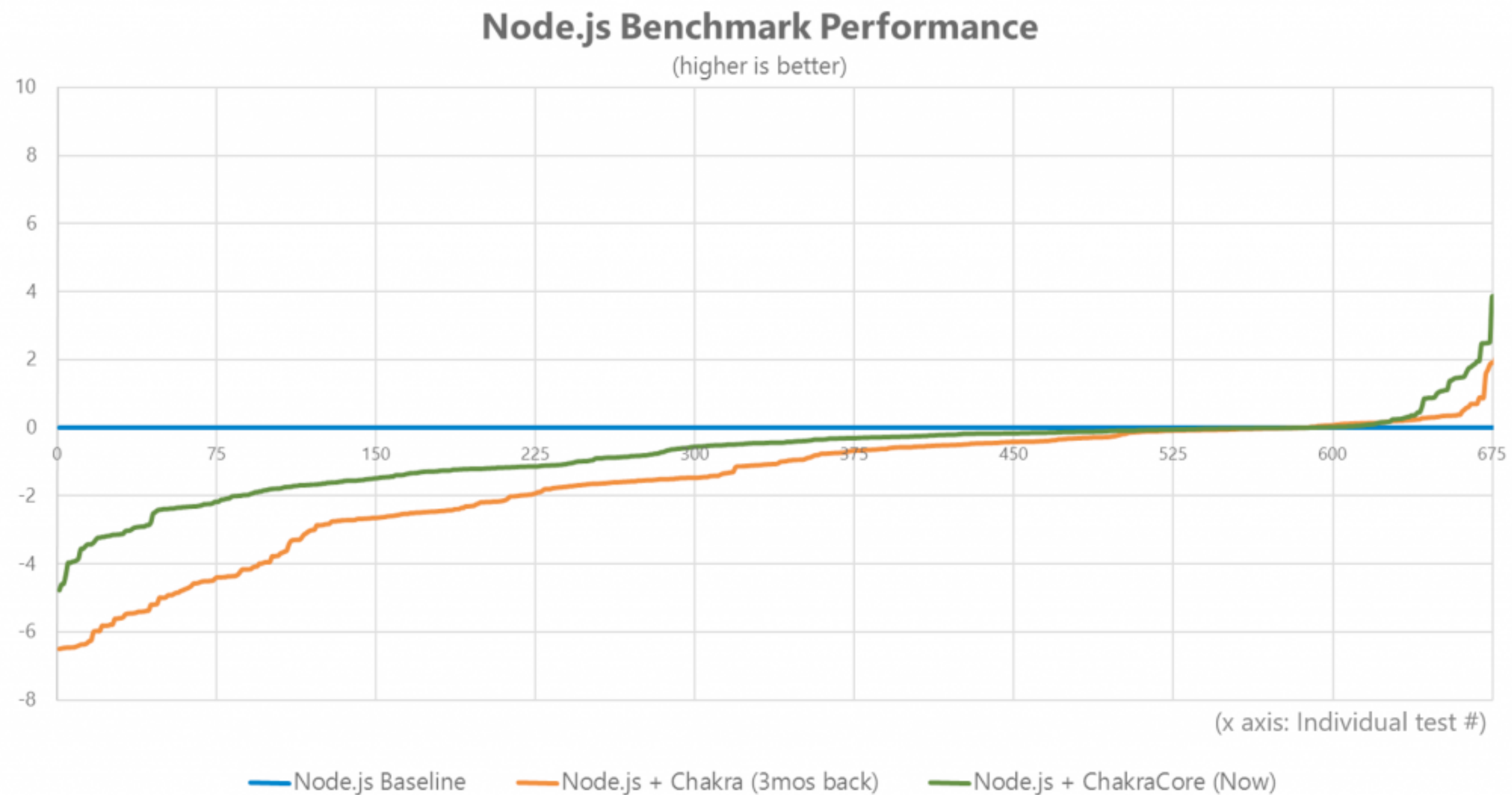
132 0

<https://www.youtube.com/watch?v=1bfDB3YPHF1>

NODE USING CHAKRACORE

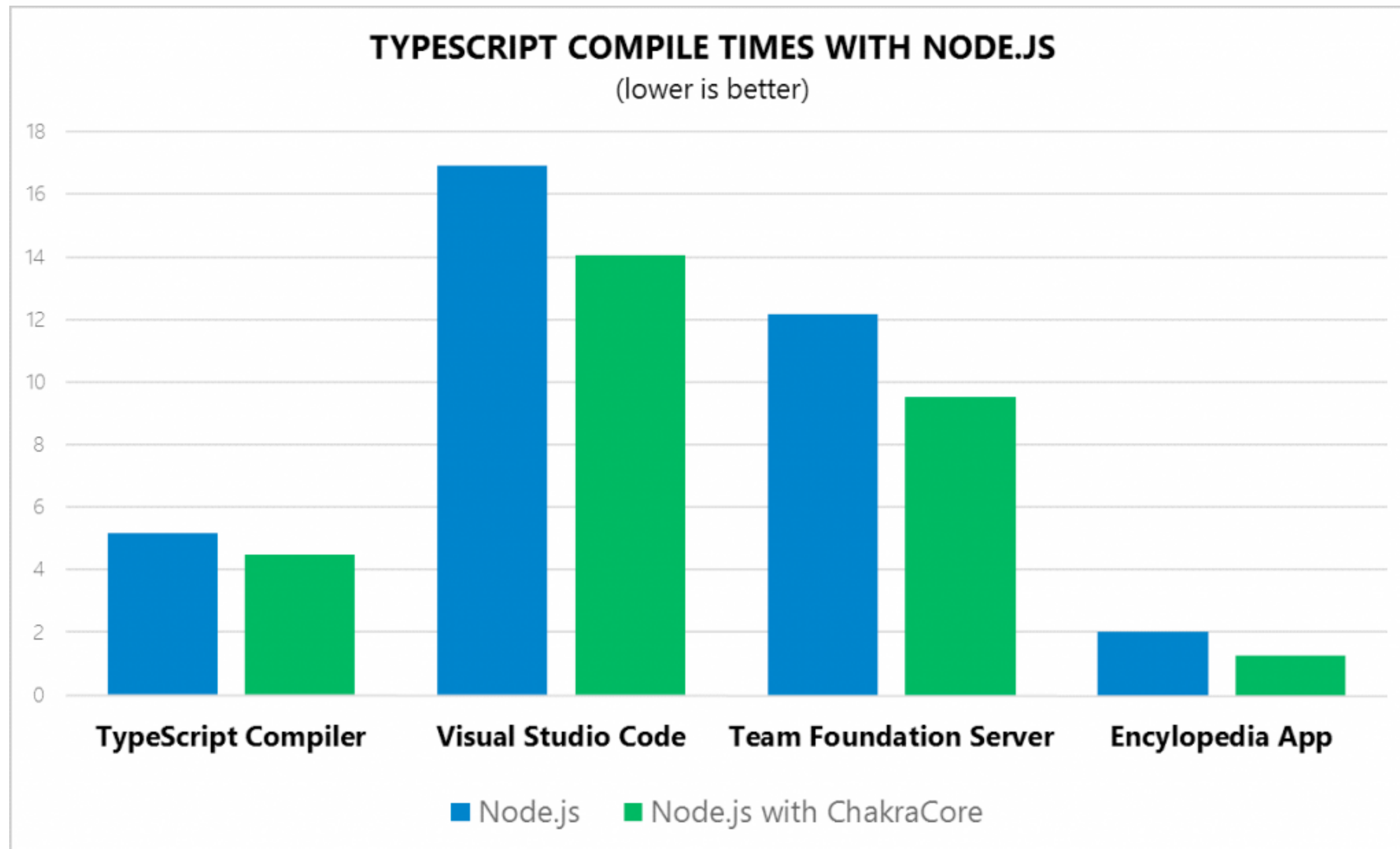


NODE USING CHAKRACORE



<https://blogs.windows.com/msedgedev/2016/01/19/nodejs-chakracore-mainline/>

NODE USING CHAKRACORE



<https://blogs.windows.com/msedgedev/2016/01/19/nodejs-chakracore-mainline/>

COME AND PLAY

ChakraCore

chat on [gitter](#)

ChakraCore is the core part of Chakra, the high-performance JavaScript engine that powers Microsoft Edge and Windows applications written in HTML/CSS/JS. ChakraCore supports Just-in-time (JIT) compilation of JavaScript for x86/x64/ARM, garbage collection, and a wide range of the latest JavaScript features. ChakraCore also supports the [JavaScript Runtime \(JSRT\) APIs](#), which allows you to easily embed ChakraCore in your applications.

You can stay up-to-date on progress by following the [MSEdge developer blog](#).

Build Status

	Debug	Test	Release
x86	build passing	build passing	build passing
x64	build passing	build passing	build passing
arm	build passing	build passing	build passing

Above is a table of our rolling build status. We run additional builds on a daily basis. See [Build Status](#) for the status of all builds.

<https://github.com/Microsoft/ChakraCore>



THANKS!

CHRIS HEILMANN

@CODEPO8

CHRISTIANHEILMANN.COM