

V8, WebAssembly, and the Future of JS and a Multi-Language Web



About Me

- Robotics Author/Addict
- Developer 
- Twitch streams hardware/software @nodebotanist



What even is WebAssembly?

What WebAssembly is **NOT**

- just a programming language (it's so much more than that)
- the death of JS (probably)
- something you can just ignore cause it's gonna go away



What WebAssembly IS

- A compilation target for other languages to compile to, as well as a language in itself
- An augmentation of the abilities of JS by allowing other languages to operate in the browser
- But most importantly...

- Pretty literally* `magic(k)`



* - no not really literally but I'll explain later

WebAssembly is a **compilation target**

- You write code in other languages and compile them into WebAssembly
- Rust, C/C++, Go, C#; these are just a few of the languages with WebAssembly as a compile target

But the question you then have
to ask is...

WHAT



WHY



There are so many reasons you
would want this in your life

THIS IS A NEW ERA FOR THE
WEB



No but seriously

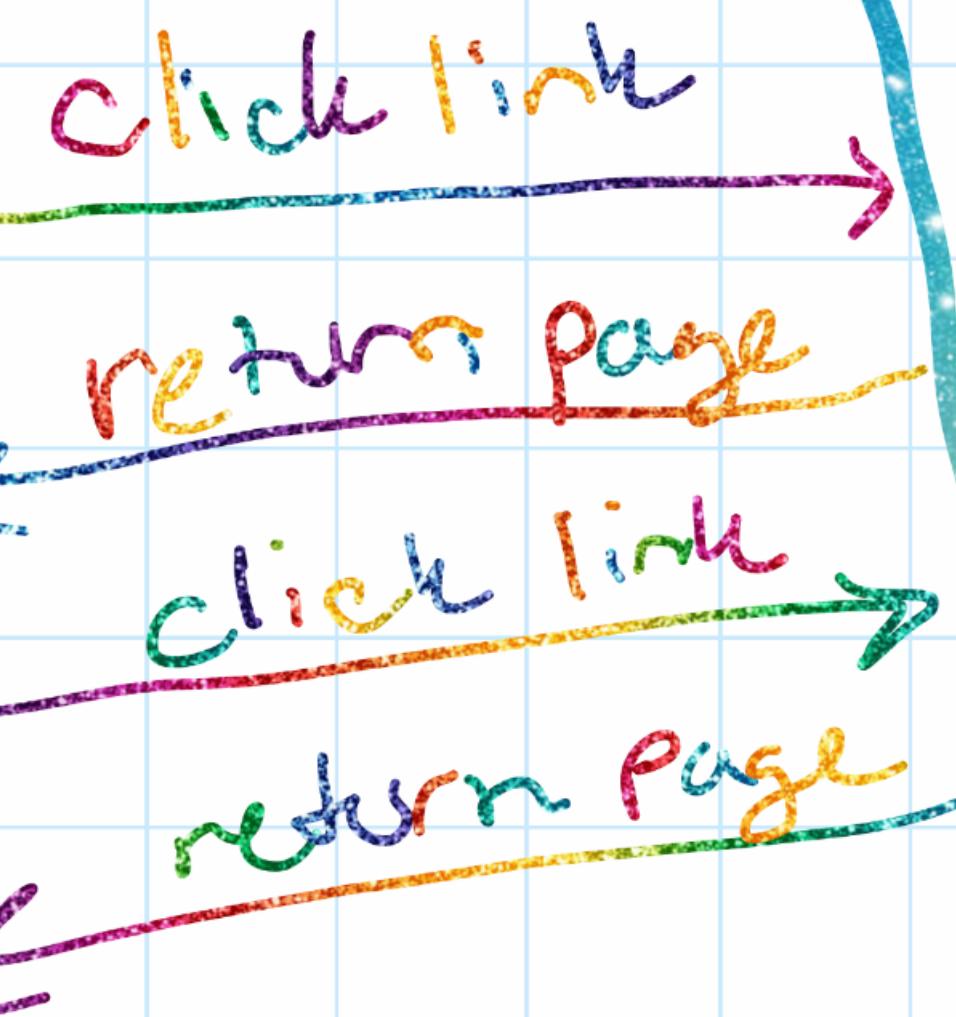
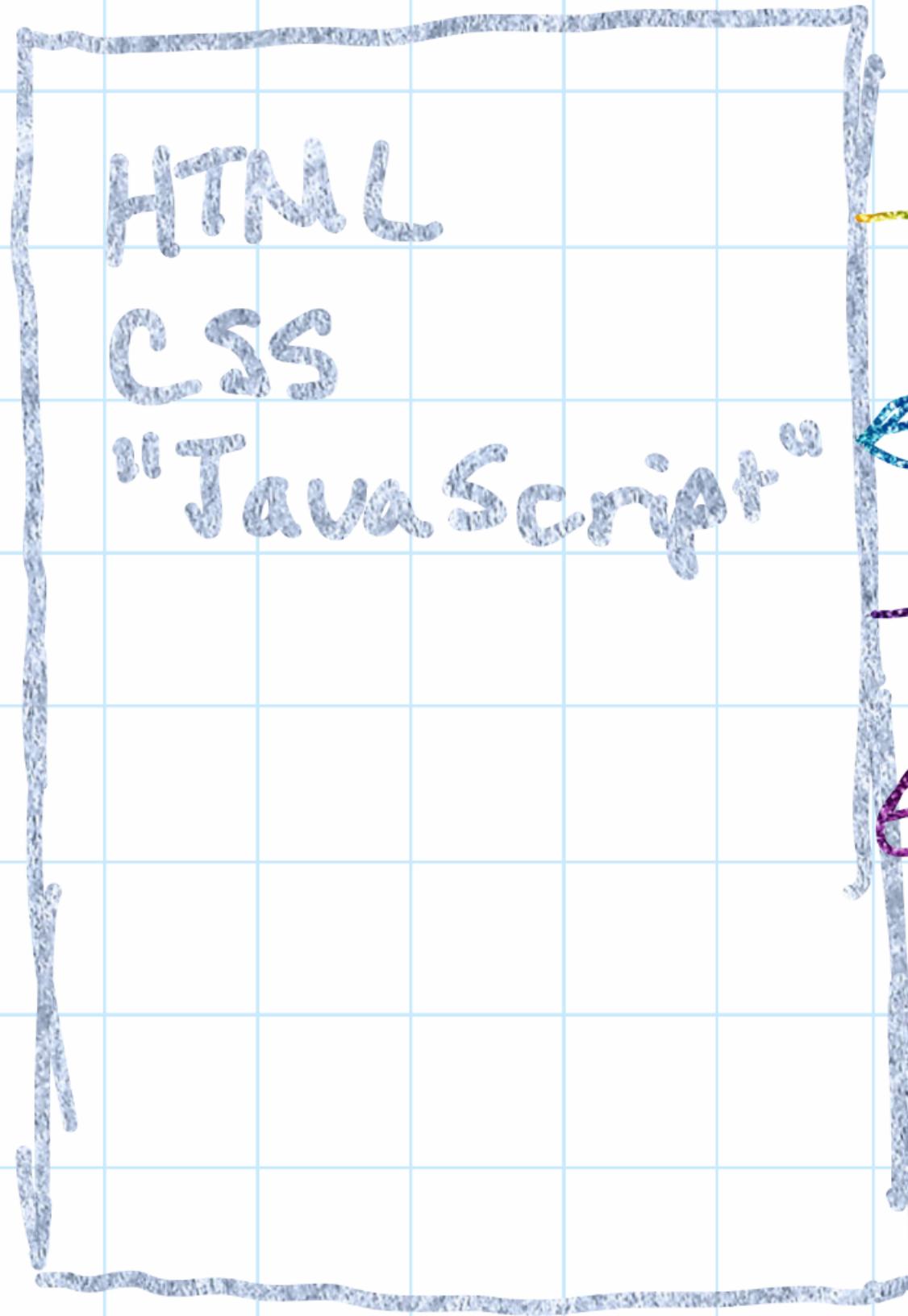
WebAssembly is comparable to bringing the power of the JVM and the subsequent LLVM toolchain **into the browser**, creating an **evolution of the web as we know it**.



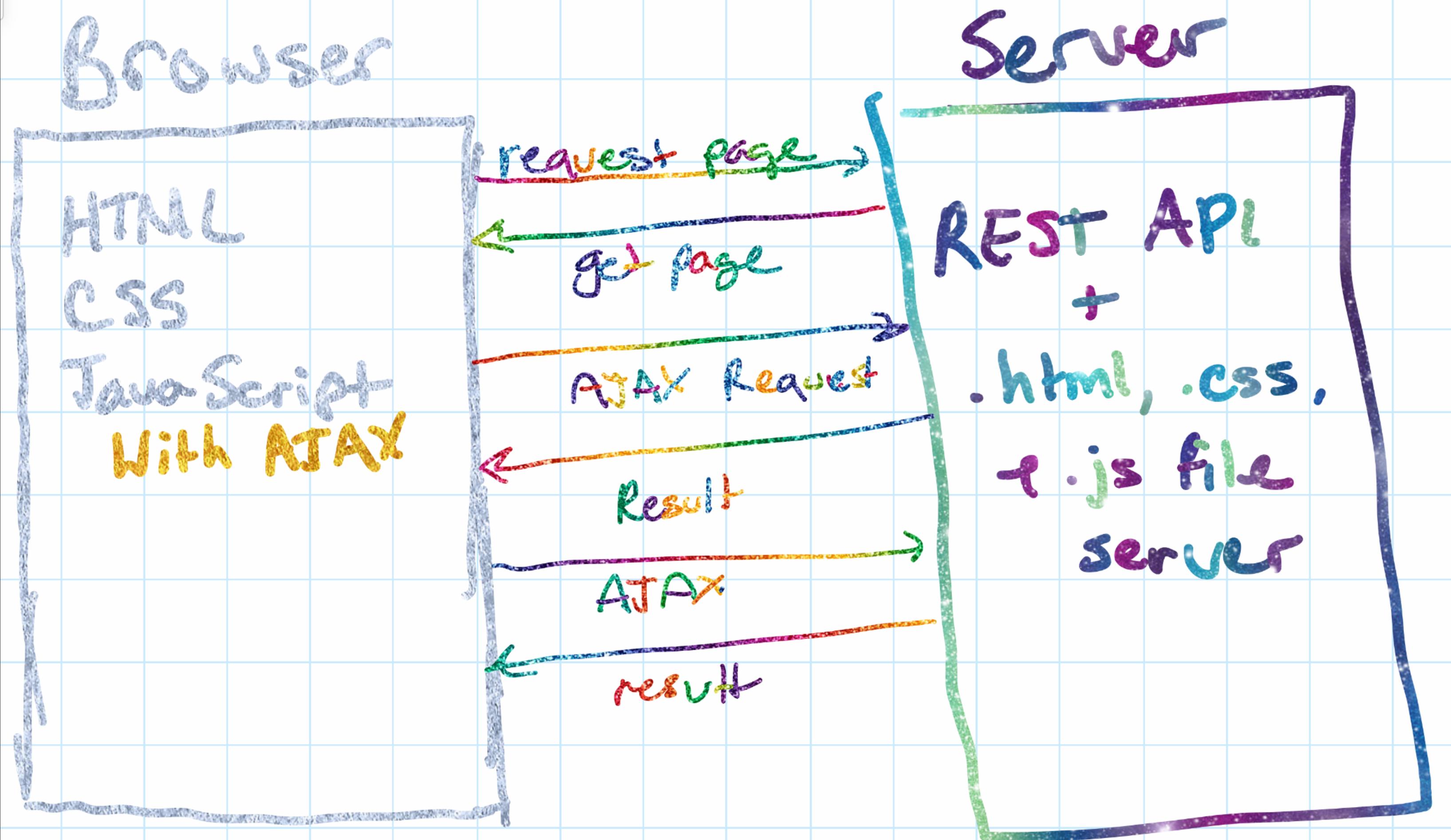


Browser

Server



LITERALLY
EVERYTHING
OVER
HERE



Browser

HTML

CSS

JavaScript

With AJAX
and Service
Workers

And WebAssembly

Server

request page

return page

use WebAssembly
to compute

use service worker
3rd party module

AJAX

yeahOK.

LOL OK.



Why does this matter?

- Augmenting JS at its not-so-strong points
- Not rewriting entire codebases to use them on the web
- Fewer calls to the server, less latency, faster web apps

Augmenting JS at its not-so-strong points

Who wants to write a banking app in JS?

If you're running *anything* that relies on mathematical numerical accuracy or speed that meant, until now, another AJAX call to have another language do alllllllll the math. With WebAssembly, we can do this in the browser, with, say, Rust.

Other JS not-so-strong points

- Type coercion side-effects: "" == 0 //true
- API Overloads:

```
1 + document.getElementById("inputElem").value; // Concatenates  
1 + Number(document.getElementById("inputElem").value); // Adds
```

- The mystery that is typeof

```
typeof {} === "object" //true  
typeof "" === "string" //true  
typeof [] === "array"; //false
```

*Using WebAssembly means using
the right tool for the job*

BUT THIS WILL KILL JS!!!



Probably not-- for most situations, it makes JS better by letting it do what it is good at and ignoring the rest.

However, WASM toolchains are gaining more and more abilities by the day, and some teams would like to have WASM be able to do everything JS does.

It makes the web better by creating better browser experiences



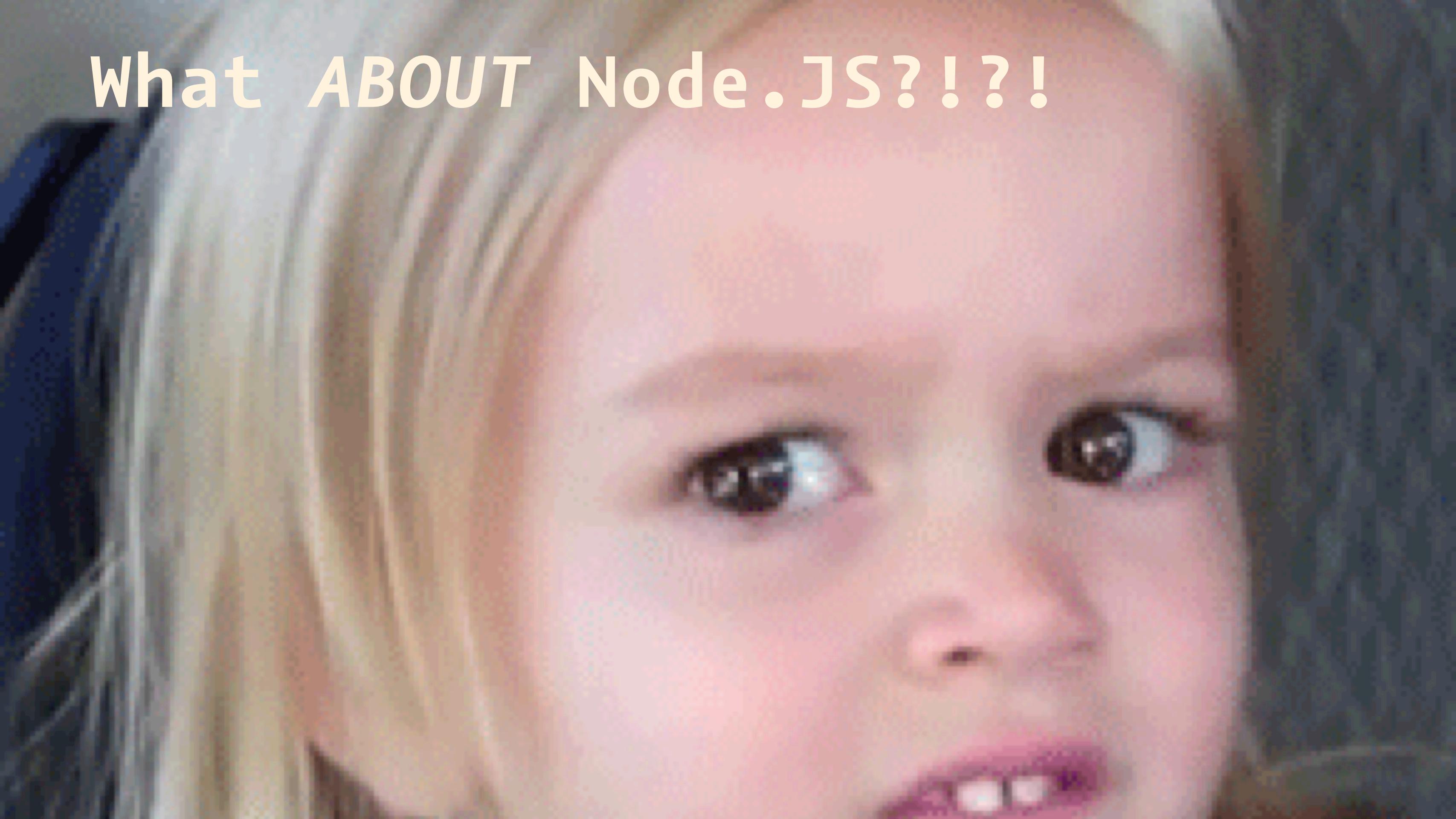
Let's *take* a closer look
with a demo

The Demo

- uses `wasm-imagemagick`
- manipulates images in the browser up to 10x faster than JS can
- Shows the real power of not having to rewrite code and being able to let us use the right tool for the job

But what about *Node.JS*?

Wait . . .



What *ABOUT* Node.js?!?!

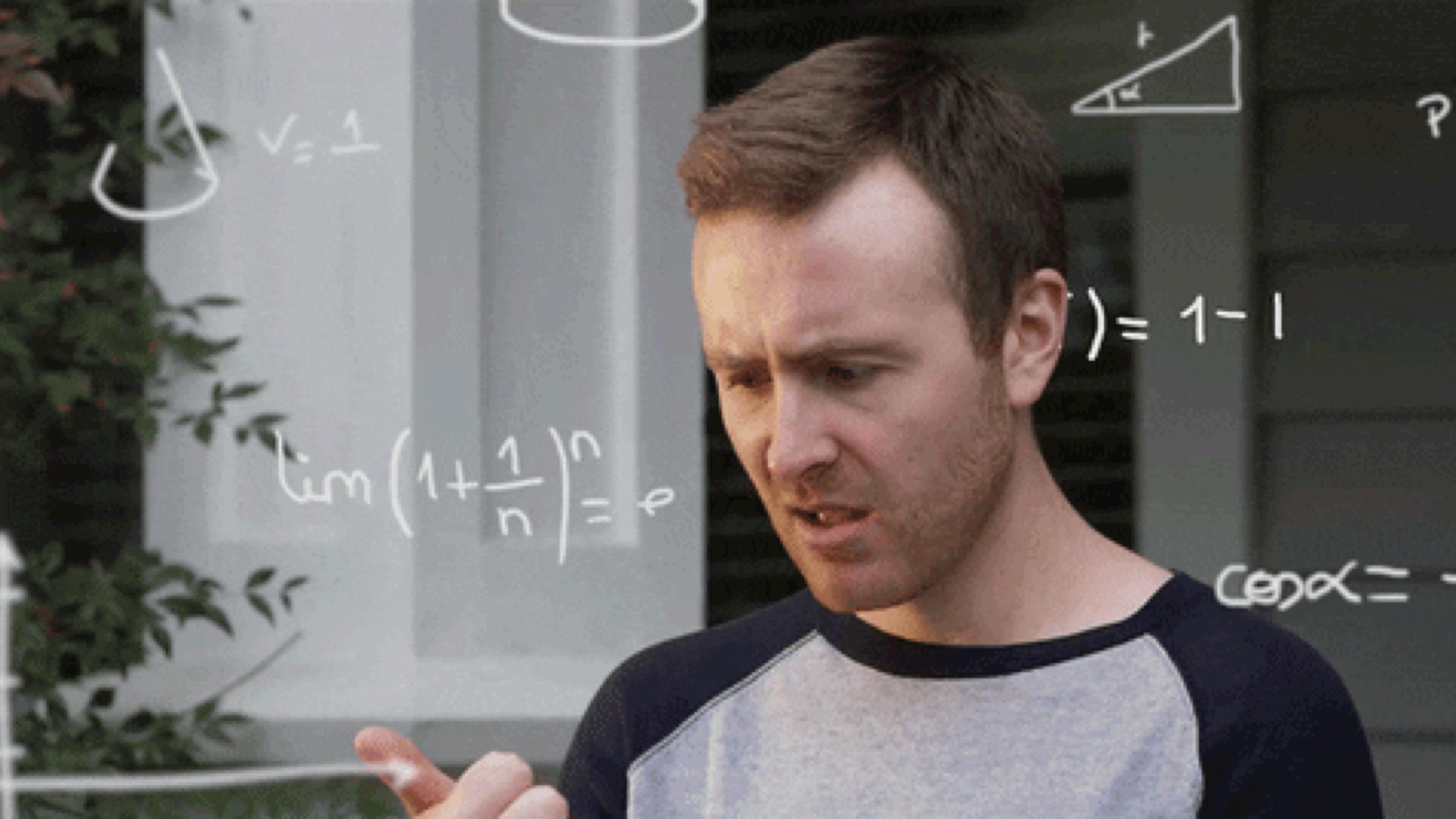
Native. Heccin. Modules.



Why native modules are such a pain

- They have to be recompiled on download
- They either have to compile on every platform OR leave off platforms from support
- Node-Gyp (disclaimer: I respecc the hecc out of their work.)

WebAssembly Works on Node >= 8.0



$$\lim_{n \rightarrow \infty} \left(1 + \frac{1}{n}\right)^n = e$$

$\sqrt{1}$

$$y = 1 - x$$

$$C_0 + \frac{C_1}{x-1} + \frac{C_2}{x+1}$$

WebAssembly Modules in Node.JS

- Are *precompiled for Node.JS*, so they're portable to *any platform that runs Node.JS*.
- No more recompilation on every download on every architecture.
- FOR REALS.

*"Everyone wants to
[deprecate] node-gyp and
WebAssembly would allow us
to do this"*

— Laurie Voss, a few weeks ago

How do we get to this magickal land?

- WASI (Web Assembly System Interface) is being actively worked on and allows WASM to make system-level calls. It's by no means complete yet, but worth keeping an eye on!
- WASI is helped along by the fact that the Node V8 sandbox is the same sandbox being worked on in a few major browsers.

WebAssembly is even invading serverless



CLOUDFLARE®
Workers

We have a free tier now!



(I'll show this again later)

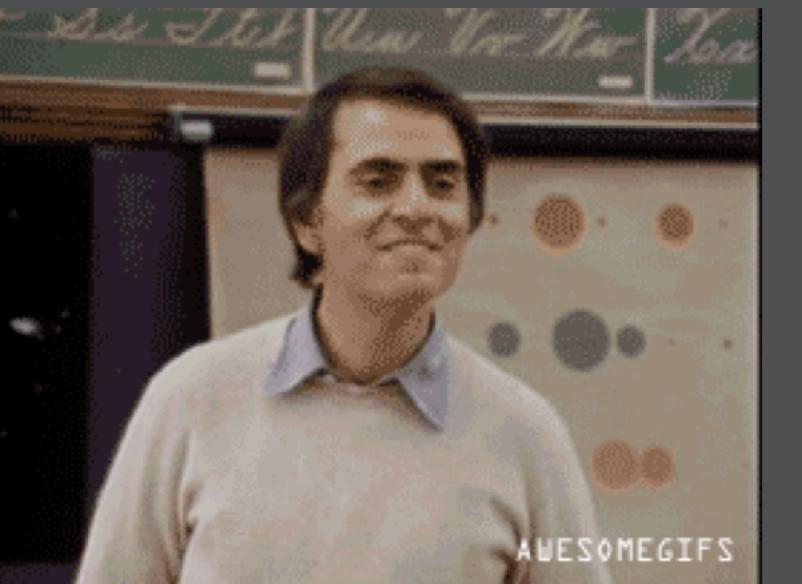
The point of this talk

- Try WebAssembly (I personally really like Rust)
- WebAssembly is the future of JS in all its forms
- If you are a hiring manager; *hire someone who is different from you.* Just go and do it.



JUST DO IT

Thanks for listening!



- kas@cloudflare.com
- @nodebotanist
- [https://github.com/
nodebotanist/austinjs-
july-2019](https://github.com/nodebotanist/austinjs-july-2019)

