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

About Me

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

About Me (Mx. Kassian Wren!)

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

What even is WebAssembly?

What WebAssembly is NOT

— just a programming language or instruction set (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...

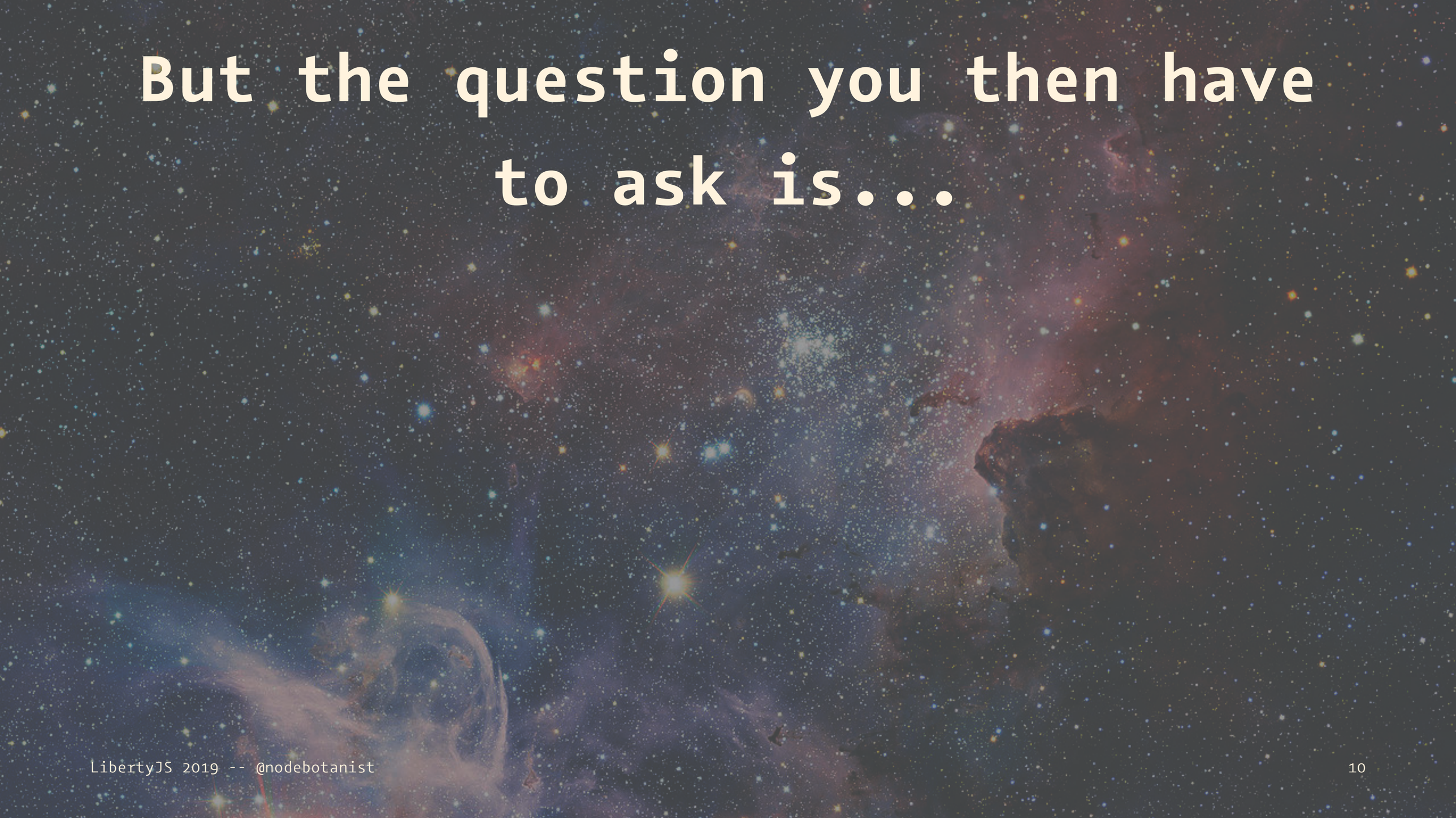
— Pretty literally* **magic(k)**

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

WebAssmebly is a compilation target

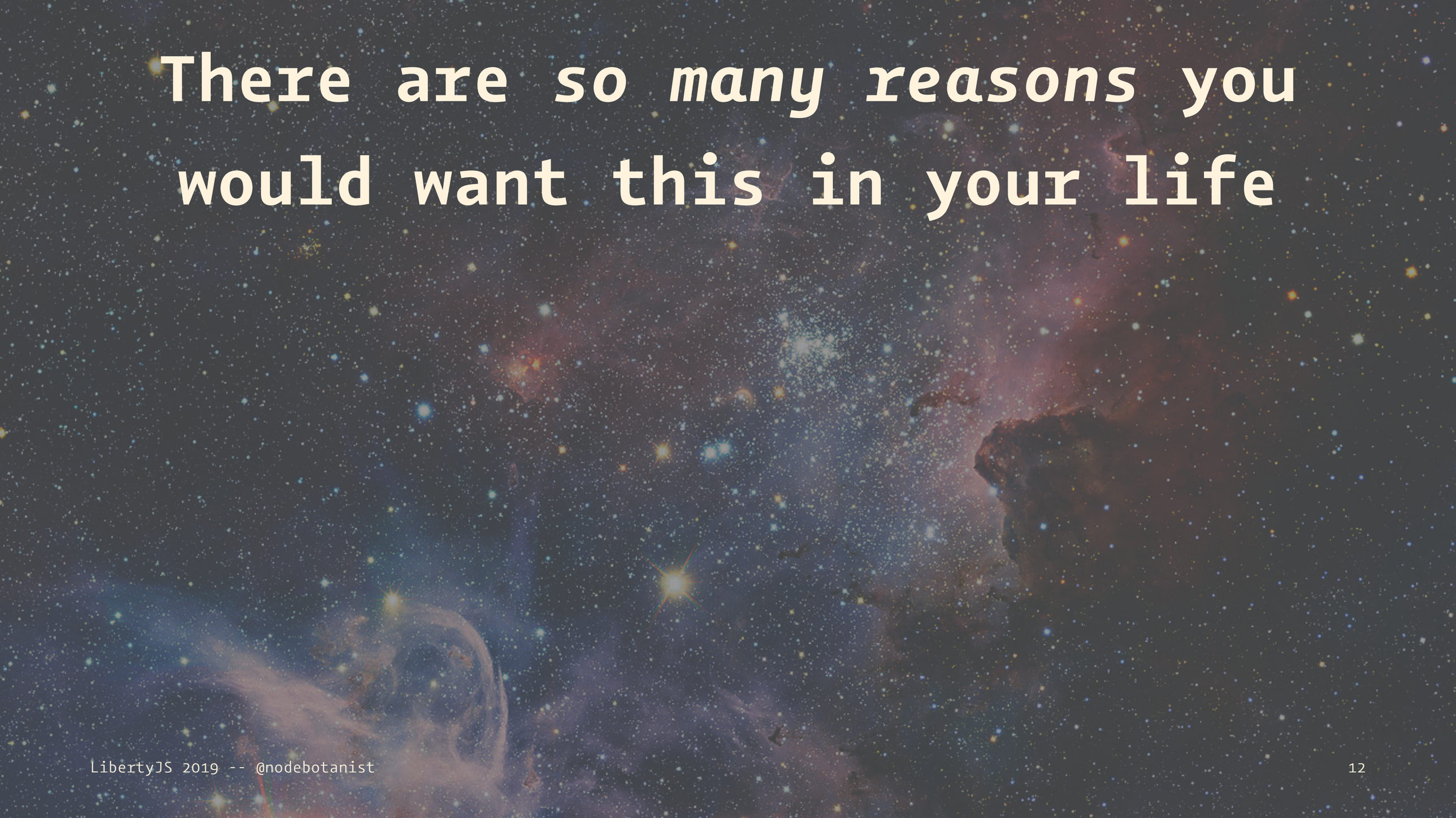
— You write code in other languages and compile them into WebAssembly

— 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
npm


WHY
npm

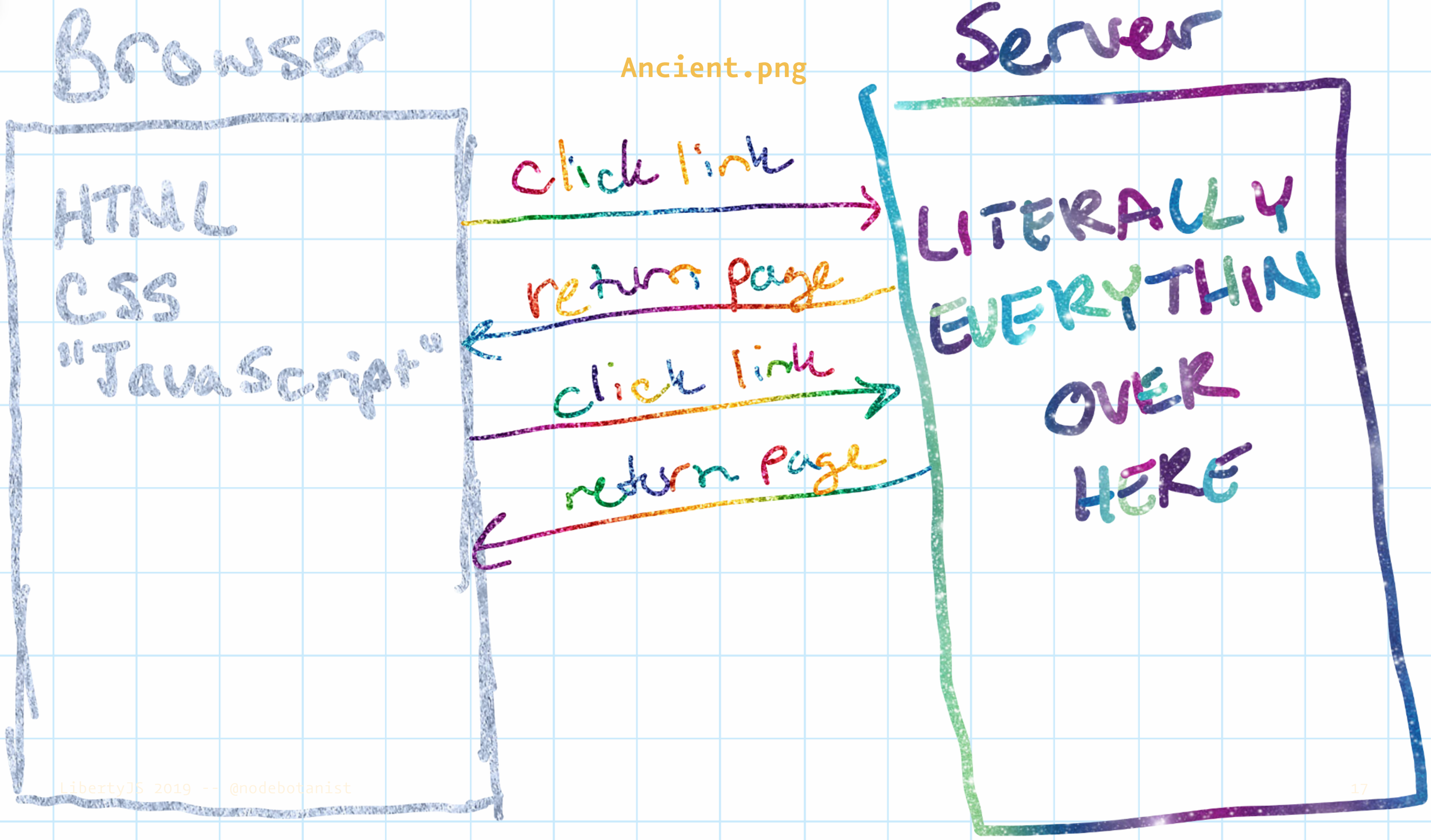

There are so many reasons you would want this in your life

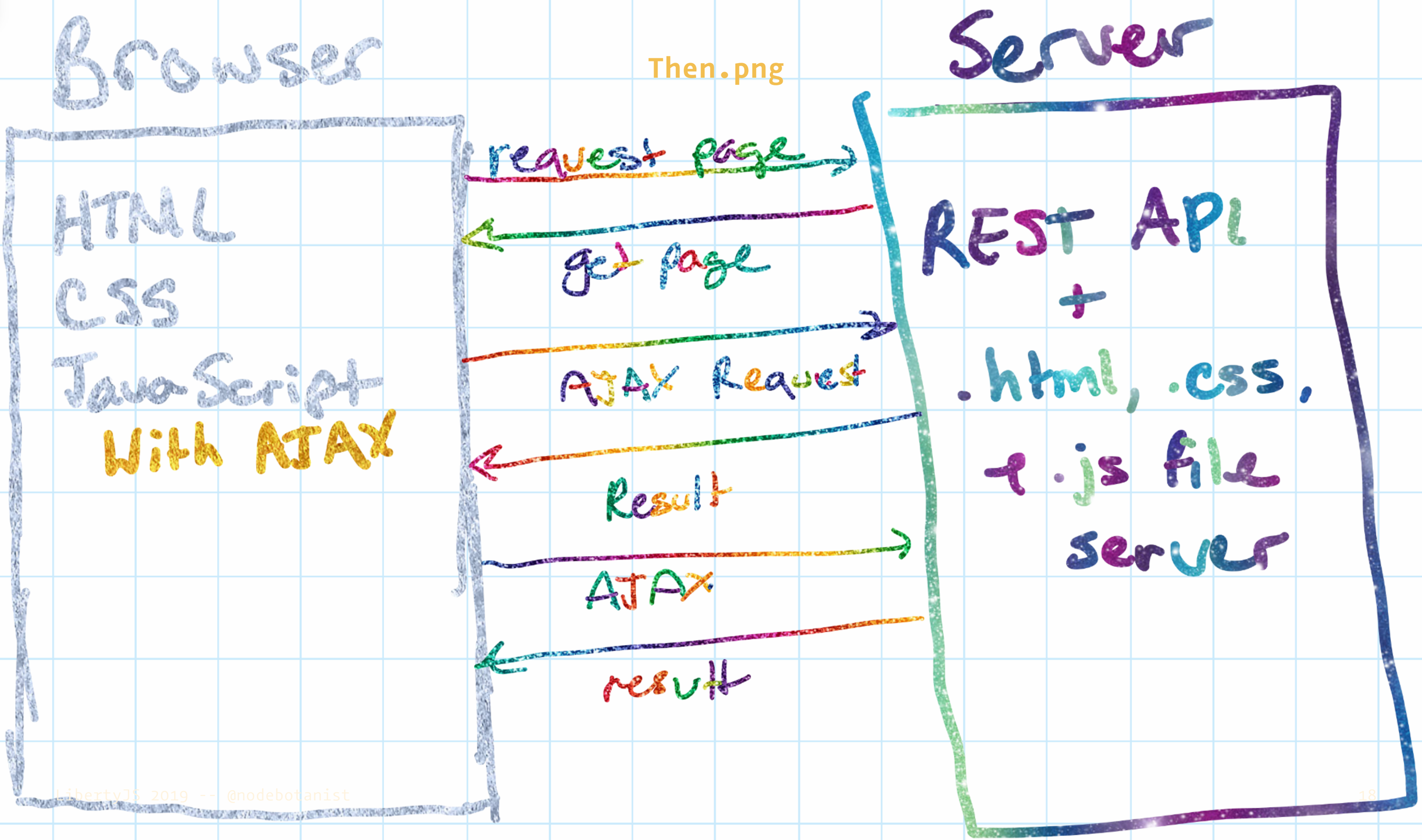

THIS IS A NEW ERA FOR THE WEB

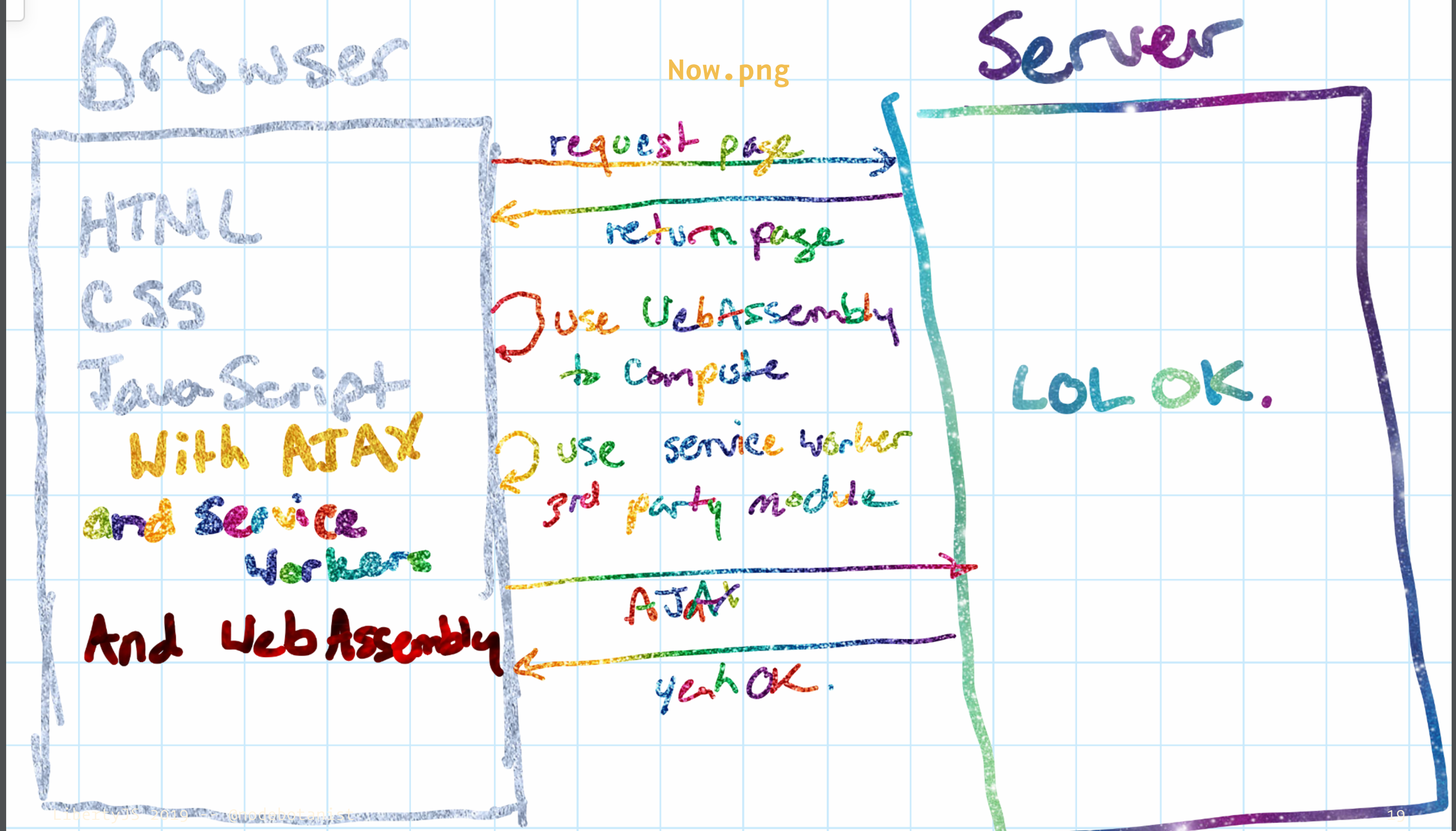
No but seriously

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

Ancient.png


Browser
HTML
CSS
"JavaScript"
click link
return page
click link
return page
Server
LITERALLY
EVERYTHIN
OVER
HERE


Then.png
Browser
HTML
CSS
JavaScript
With AJAX
request page
get page
AJAX Request
Result
AJAX
result
Server
REST API
+
.html, .css,
& .js file
server


Browser
HTML
CSS
JavaScript
With AJAX
and Service
Workers
And WebAssembly
Now.png
request page
return page
use WebAssembly
to compile
use service worker
3rd party module
AJAX
yeah OK.
Server
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 allllllllll 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
  - Especially the acciedental concat when you meant to add and vice versa.
- Anything that has to do with types:

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.

** - there are WebAssembly modules that can access the DOM and be used to manipulate the shadow DOM.

It makes the web better by creating better browser experiences

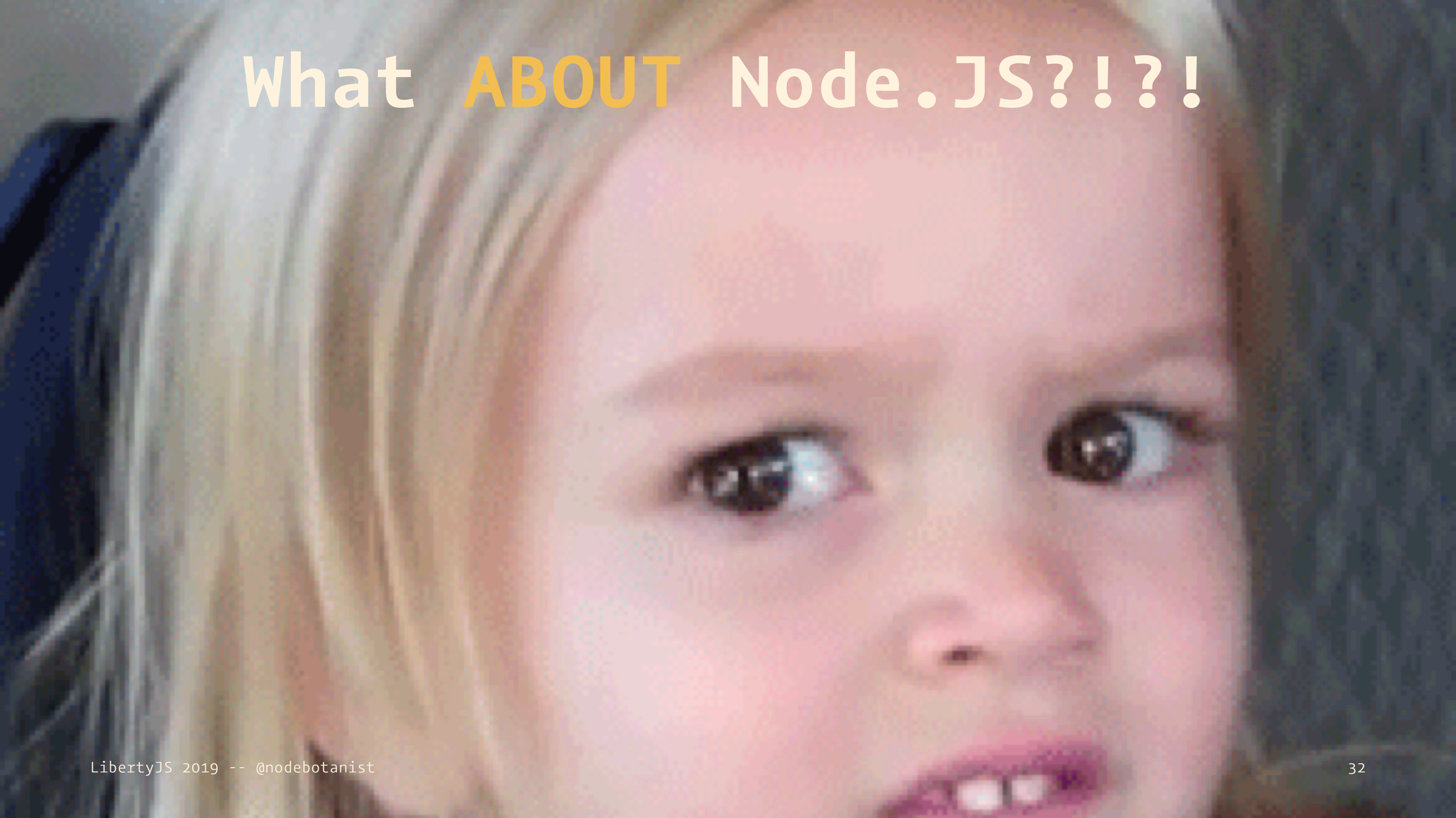

Let's take a closer look
with a demo

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 compiled on download for the architecture you're installing on

— 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


$V = \frac{1}{3}$
$\lim_{n \to \infty} \left(1 + \frac{1}{n}\right)^n = e$

WebAssembly Modules in Node.JS

— Are *precompiled binaries*, 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 [eventually] allow us to do this"

— Laurie Voss, a few weeks ago

WebAssembly is even invading serverless

CLOUDFLARE®

Workers

We have a free tier now!

(I'll show this again later)

How do we get to this magickal land?

- If you'd like to learn Rust, you can read the Rust Book and the Rust-wasm book:
- Rust book: https://doc.rust-lang.org/book/
- Rust-wasm book: https://rustwasm.github.io/book/
- If you'd like to use C/C++, check out https://emscripten.org
- C# fan? Try https://github.com/aspnet/Blazor

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/LibertyJS-2019