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

About Me

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

CatParent

What even is WebAssembly?

What WebAssembly is NOT

— a programming language

— the death of JS

— something you can just ignore cause it's gonna go away

What WebAssembly IS

— A compilation target for other languages to compile to

— An augmentation of the abilities of JS

— 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
npm
npm

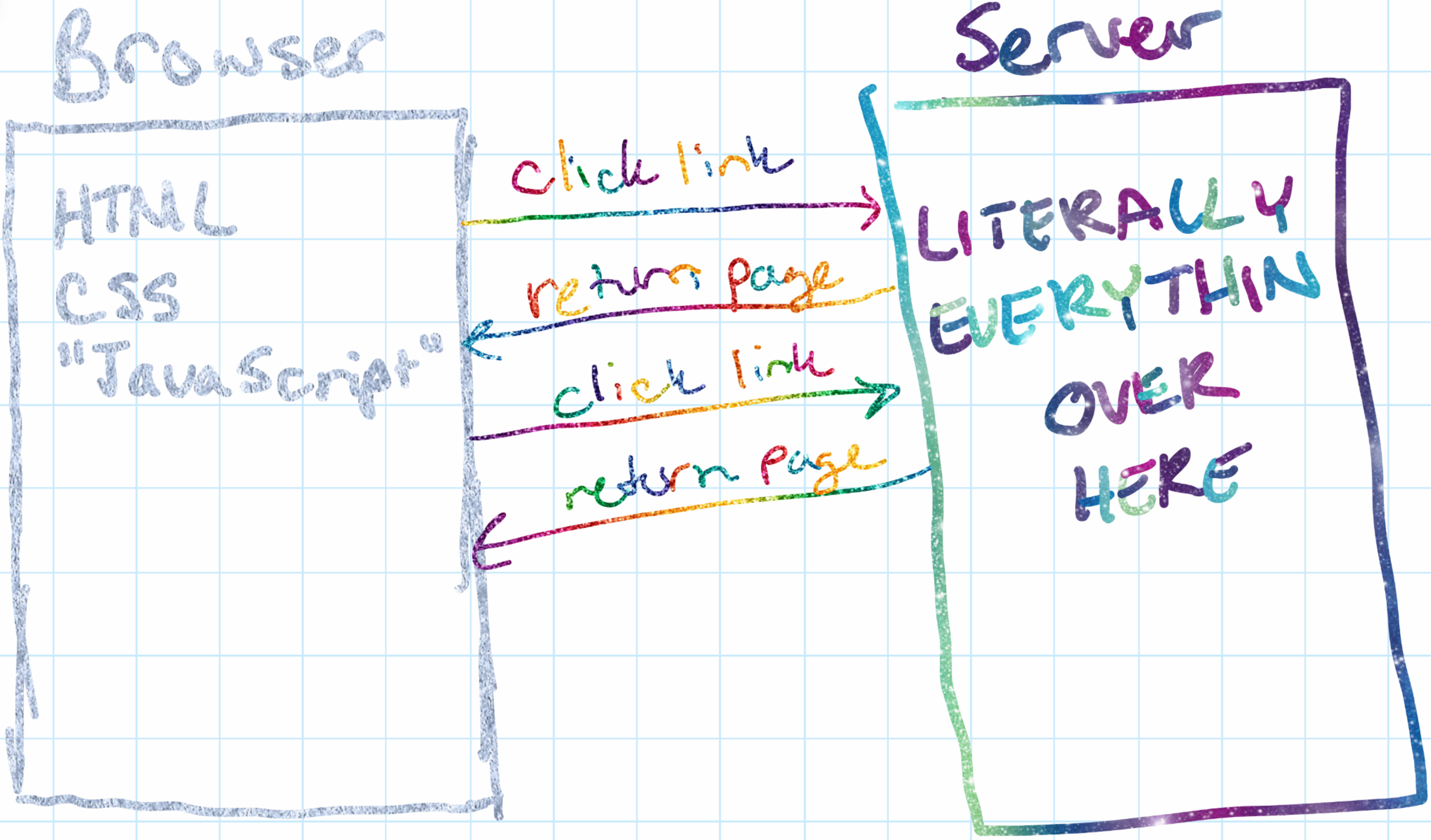

WHY
npm

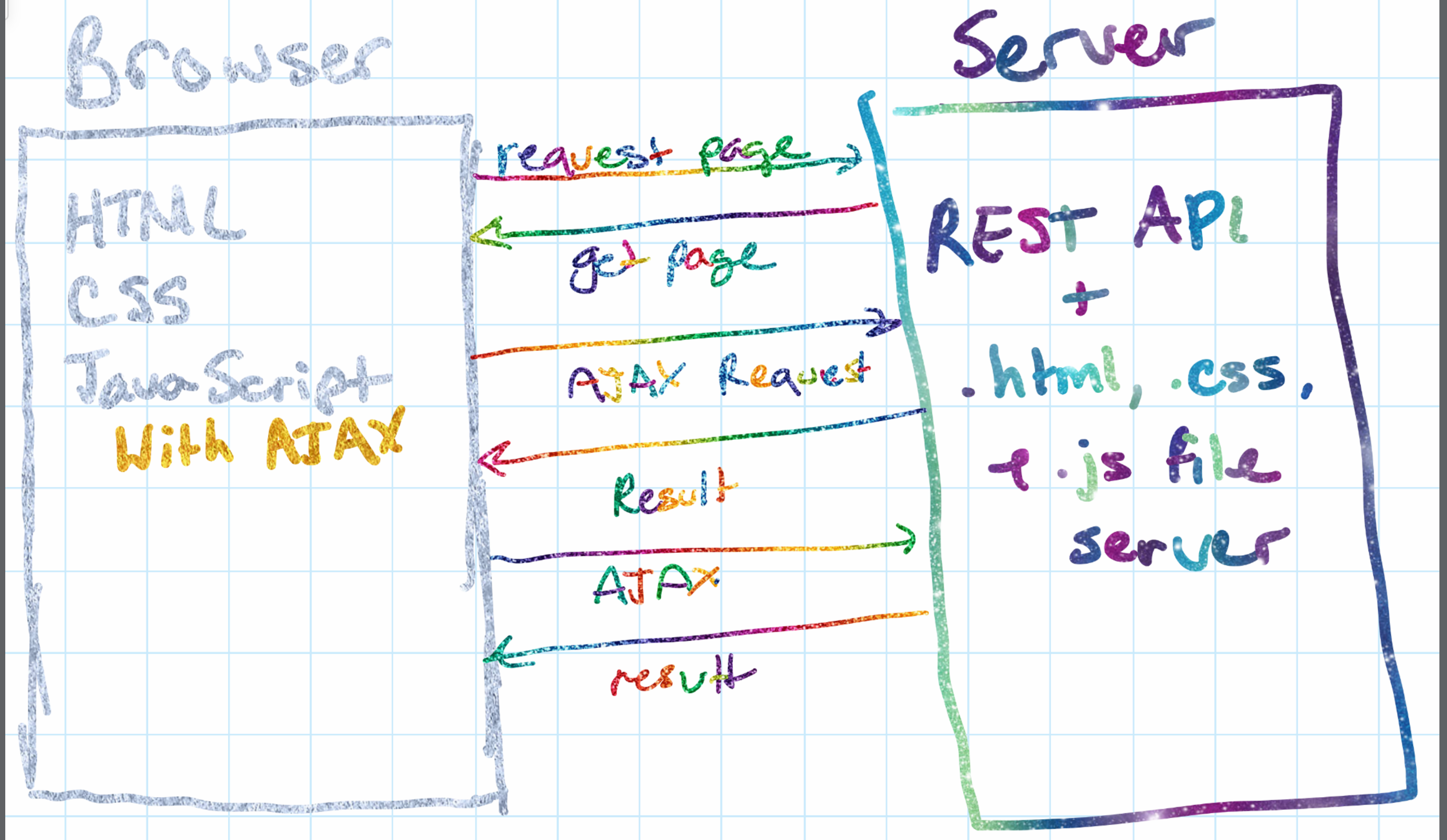
There are *so many reasons* you would want this in your life

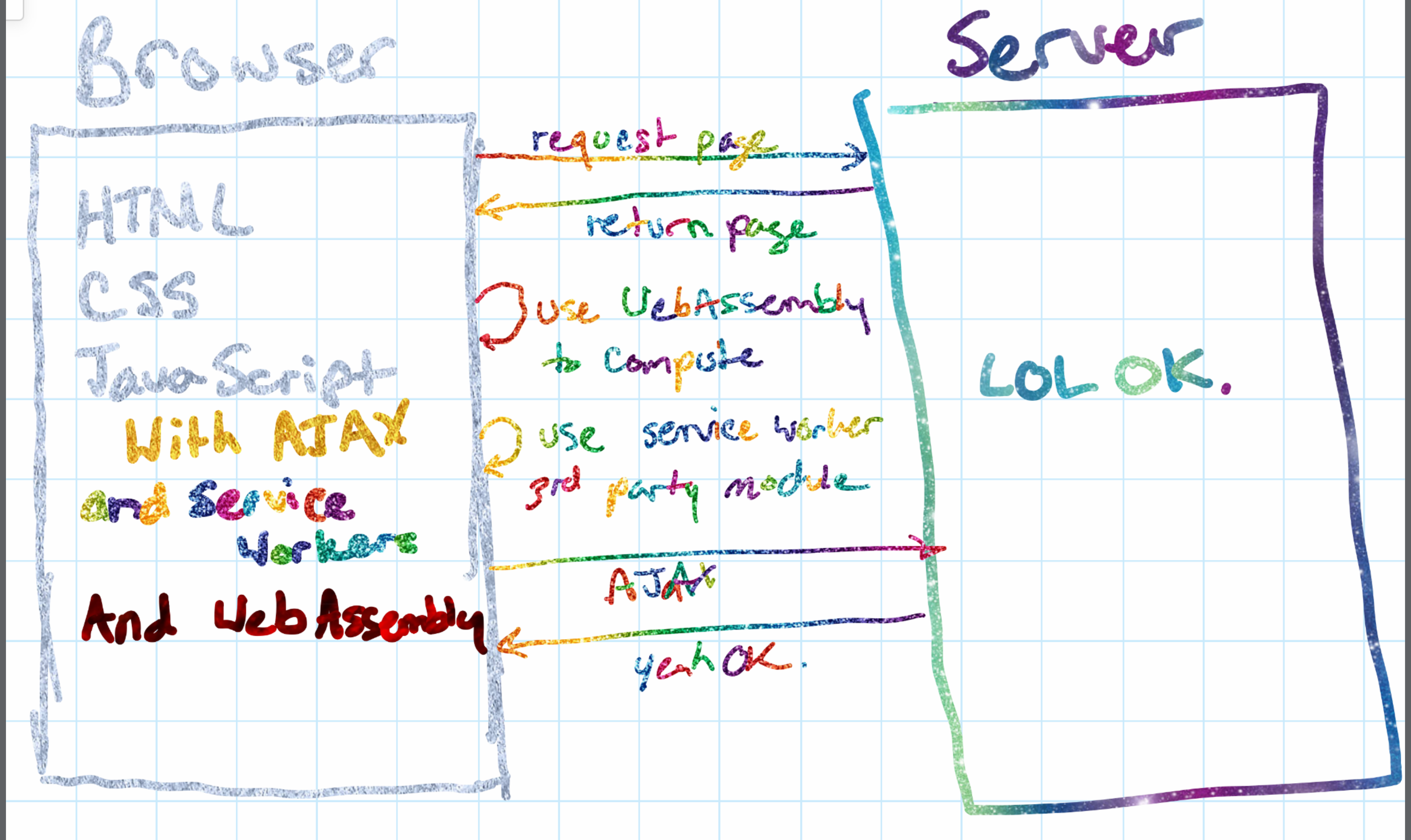
THIS IS A NEW ERA FOR THE WEB

No but seriously

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


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


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
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

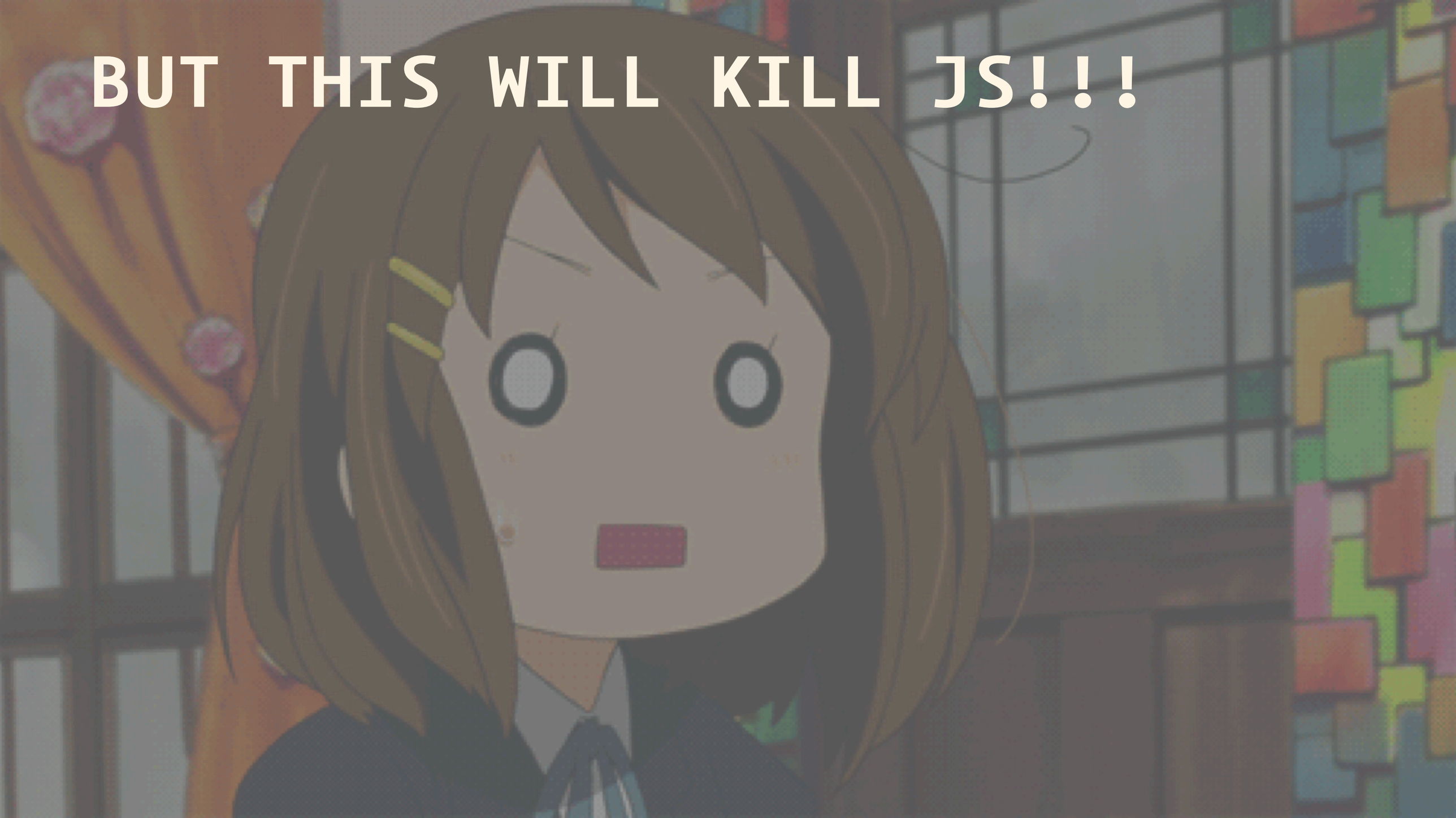
— API Overloads:

```
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!!!

No, it *makes JS better* by letting it do what it is good at and ignoring the rest.

It *makes the web better* by creating better browser experiences

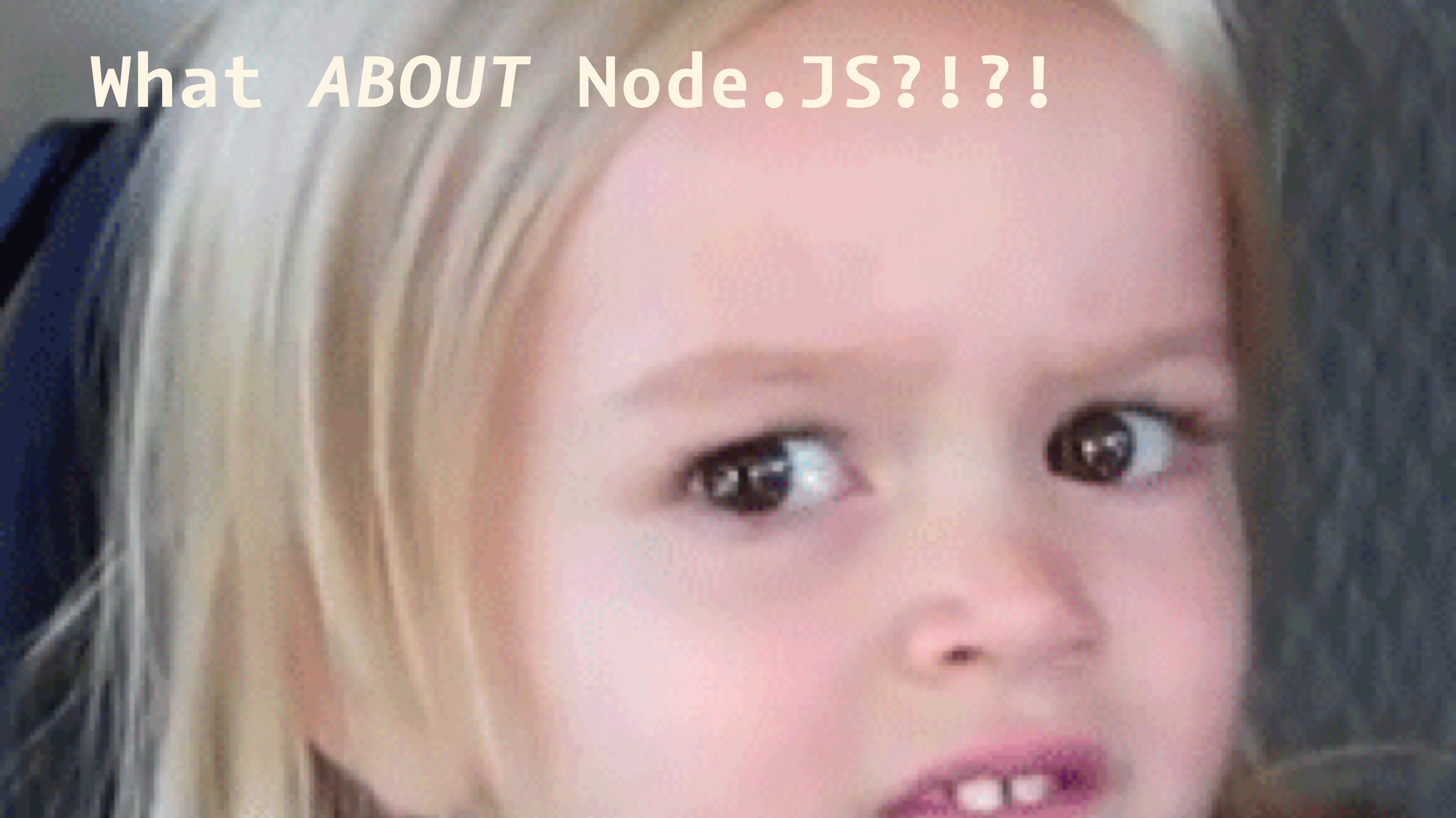
Let's *take a closer look* with a demo

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*


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

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, this morning

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