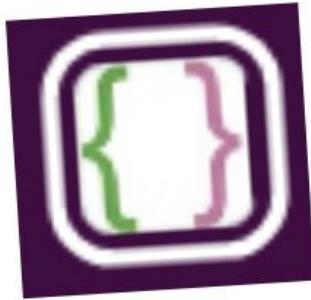


I'm Jack Harrhy

And this, is

why the web

a big thanks !



why the web

why the web?

why the web?!?!?!

similar to a previously delivered talk, but
geared to a more intermediate / senior audience



what the web

Google Developer Groups
St. Johns

why the web

some quick
questions for
the room!



working at a company
that has some web
presence

/

built websites previously



considers themselves
a "full-stack developer"



writes JavaScript/TypeScript
on a regular/semi-regular basis



enjoys writing CSS

for those that didn't
raise their hand, i hope this
is still an interesting talk to
listen to,

for those that did, hopefully
what i have to talk about here
might be useful to you

:)

Obligatory whoami slide:

My background:

Currently I work at C-CORE: Engineering Stuff

Previously I've worked for:

CoLab Software: GitHub for Engineers

Bluedrop ISM: Integrated Skills Management

And worked on some random projects:

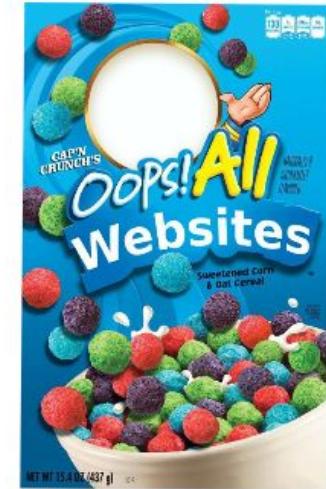
retrievium - discovery tool for chemistry

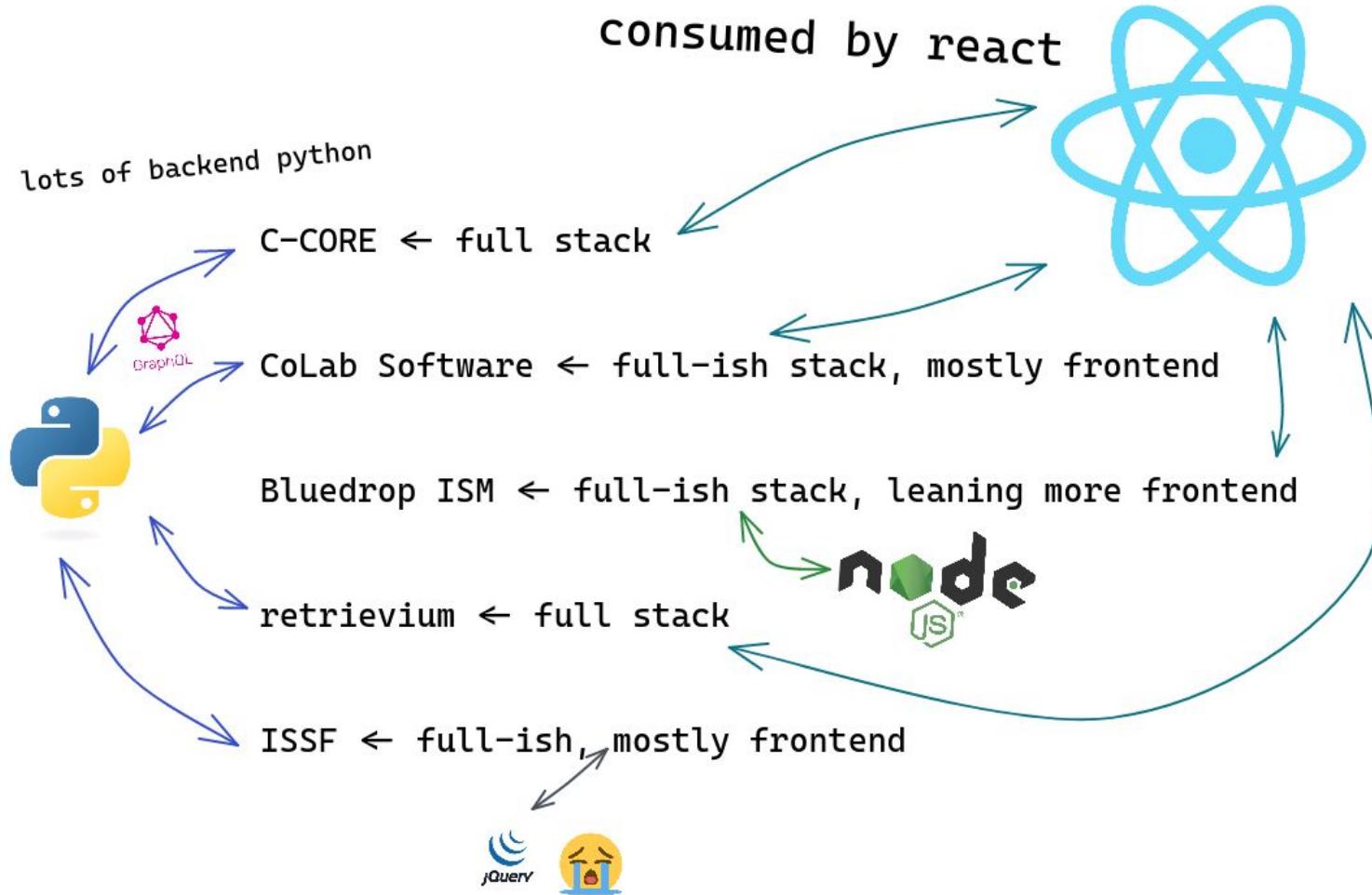
ISSF - Information System on Small-scale Fisheries



C-CORE
CoLab Software
Bluedrop ISM
retrievium
ISSF

Websites!





the technologies / tips & tricks / etc.
i'm about to talk about in this talk,
should be taken with a considerable
serving of salt



they have been useful in helping me
solve problems and build websites, but
the same may not always translate to
everyone here!

however ...

i am going to spend lots of this talk
talking about what i find useful, whenever
you see a slide formatted like this:

tip: *some amazing insight from
jack about web dev*

remember:



outline

- brief terminology
- problems with the web
- how did we get here?
- things aren't that bad
- tools, technologies, & tips

(brief overview of)

terminology

the "triforce" of web technologies



HTML: Hyper Text Markup Language

CSS: Cascading Style Sheets

JS: JavaScript

its related to java, right?

HTML: the skeleton

the first thing sent from
the server → client

holds everything together,
references the rest of the files

sometimes, usable / viewable
on its own just fine



CSS: the skin/hair/clothes

is applied to the html to
give it look & feel

not concerned with content,
but how to make content not
look 'default'

the 'style' of the page 🎨



JavaScript: the nervous system / brains

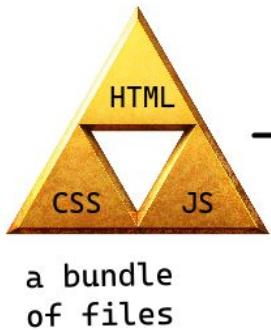
takes a static page, and
makes it interactive ☀

permits developers to add
logic to pages, i.e.

- when user clicks button,
send email
- after the page loads, fetch
more data from the server,
such as notifications
- has the ability to modify the
'DOM' (document object model),
so it can visually change the site



web browsers: the artist



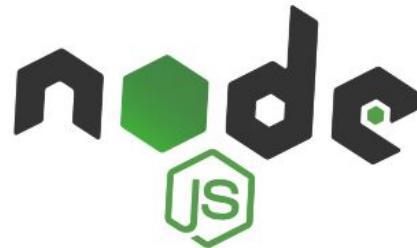
we take in html/css/js,
and render an interactive
browser window to the user!

the user,
you!

nodejs / node: javascript outside the browser

can be used to create
serverside applications
using frameworks like
express

commonly used in frontend
dev tooling as well



npm: node package manager



other peoples code, 'packaged' up
and made accessible to other people,
because when building a website you don't
want to reinvent the wheel!

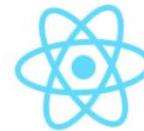
npm cli: what you use on the terminal

npm registry: the default (& defacto) package registry

npm, inc.: a company that oversees the above



vue



react



angular

"the big 3"

frontend(?) frameworks/libraries



svelte



solidjs



remix

NEXT.js

nextjs



jquery

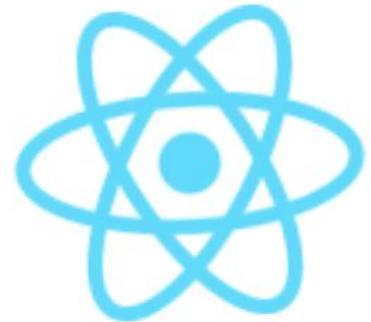


knockout

"new kids on the block"

"frontend?"

"old school"



react

remember:



that was our brief review of terminology, lets jump right into some problems

problems with the web:

build systems



so many browsers to support!
what if i want new features now?



is our 'triforce'
enough to solve these
problems?



slow connections are
common in some areas!
i can't just ship 50+ files
over a network separately ...

that sounds great, right?

an unfortunate increase in complexity



ill take your javascript
and make it browser compatible!



ill take your javascript
and squash it into tiny
files!



ill give you nicer css,
but i have a compile-time
step that requires bindings
to a C/C++ library!



webpack

ill take all of the above and produce the output,
but oh boy will you need to configure me!

before:

my-project/
index.html
style.css
script.js

i create a bunch of files,
and while im editing them,
i have the index.html file
open in my browser!

my-next-gen-project/

dist/ after build, some ugly
... mess of files for the
config/ browser

webpack.common.js

webpack will do the heavy lifting!

webpack.dev.js

but gotta setup all the plugins,
and give it definitions for dev & prod!

webpack.prod.js

public/

index.html

sort of looks like a html file, but usually
the pre-build html file that webpack processes
with its many plugins

src/

main.js

this src/ directory is what we wanted
from the start, but at what cost!

other-stuff.js

foo-bar-baz.js

random-stylesheet.css

package.json

package-lock.json

oh yeah, and since we need to install
build tool dependencies, we will need
these files to tell npm what to download too!

oh yeah, install these too!



That's right, I use React



R I can't

E Setup

A Node environment

C Help

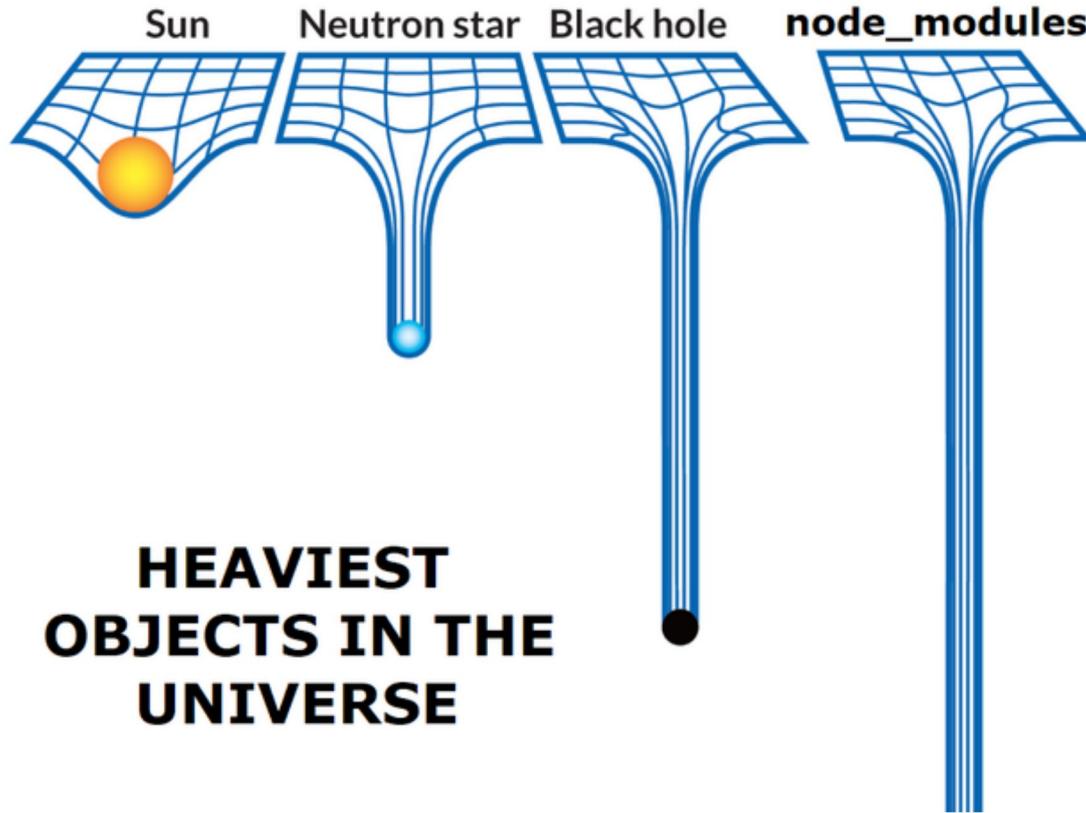
T

sometimes, this is someones
first introduction to web dev!



problems with the web:

node_modules



How to calculate size of all node_modules?

```
find . -type d -name node_modules -prune | tr '\n' '\0' | xargs -0 du -sch
```



```
fish /home/jack/repos
" 206M ./personal/jackharry.com/node_modules
 347M ./personal/bogos/frontend/node_modules
 27M ./personal/punch-the-keys/node_modules
 63M ./personal/punch-the-keys/web-component/node_modules
 202M ./personal/collarbone-softtissue/node_modules
 340M ./personal/james-website/node_modules
 194M ./personal/jack.camera/node_modules
 144M ./personal/scriptclick/stagehand/node_modules
 161M ./personal/scriptclick/node_modules
 324M ./personal/scriptclick/auditorium/node_modules
 92M ./personal/compsci-bois-tree/node_modules
 174M ./personal/yaMUN/node_modules
 2.0M ./personal/vent/node_modules
 124M ./personal/distributed-library/frontend/node_modules
 125M ./personal/stuff/frontend/node_modules
 87M ./personal/bullet/node_modules
 119M ./personal/infinite-pdf-theorem/frontend/node_modules
 49M ./personal/infinite-pdf-theorem/backend/node_modules
 88M ./personal/text-online/client/node_modules
 480M ./personal/www.cs.mun.ca-jaharry/node_modules
 848K ./personal/sunshine/node_modules
 165M ./personal/bolik/bolik-frontend/node_modules
 247M ./personal/MUNCS-Website/node_modules
 327M ./personal/yamun-cli/node_modules
 61M ./personal/card-animation/node_modules
 142M ./personal/ethos/frontend/node_modules
 313M ./personal/grackdb/frontend/node_modules
 830M ./personal/stickertrade/node_modules
 21M ./personal/audioVis/node_modules
 472M ./personal/feb2022-kattis-comp/slideshow/node_modules
 456M ./ret/retrievium/remix-app/node_modules
 19G total
jack@pop-os ~/repos>
```



19G!

problems with the web:

dependencies

leftpad

A man in Oakland, California, disrupted web development around the world last week by deleting 11 lines of code.

```
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;
}
```



Azer Koçulu

Image: GitHub

qz.com/646467/how-one-programmer-broke-the-internet-by-deleting-a-tiny-piece-of-code

Azer had many packages on the npm registry, including one called "kik", which was unrelated to the actual messaging app by the same name



kik reached out to Azer to take over the ability to publish to the kik package, he declined

"can we have?"



"no"

kik then reached out to npm directly, and they agreed to give up the package to them

"can we have?"

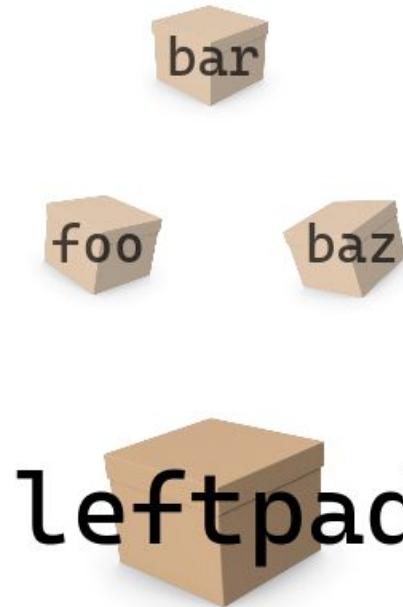


"sure"



Azer Koçulu

Image: GitHub



npmjs.org tells me that left-pad is not available (404 page) #4

Closed

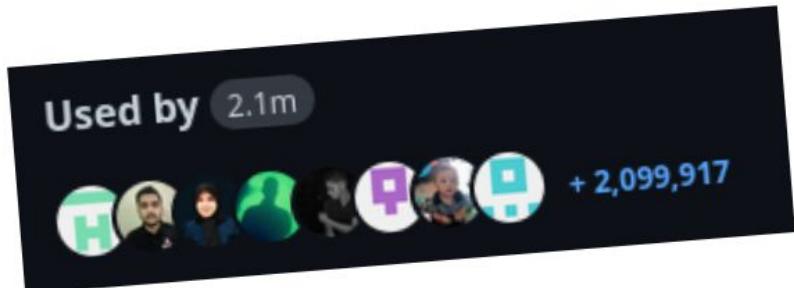
silkentrance opened this issue on Mar 22, 2016 · 193 comments

tonytamps commented on Mar 22, 2016

according to <https://registry.npmjs.org/left-pad>

```
unpublished: {  
  name: "azer",  
  time: "2016-03-22T21:27:15.696Z",  
  ...  
}
```

It's causing Babel to fail installation



core-js

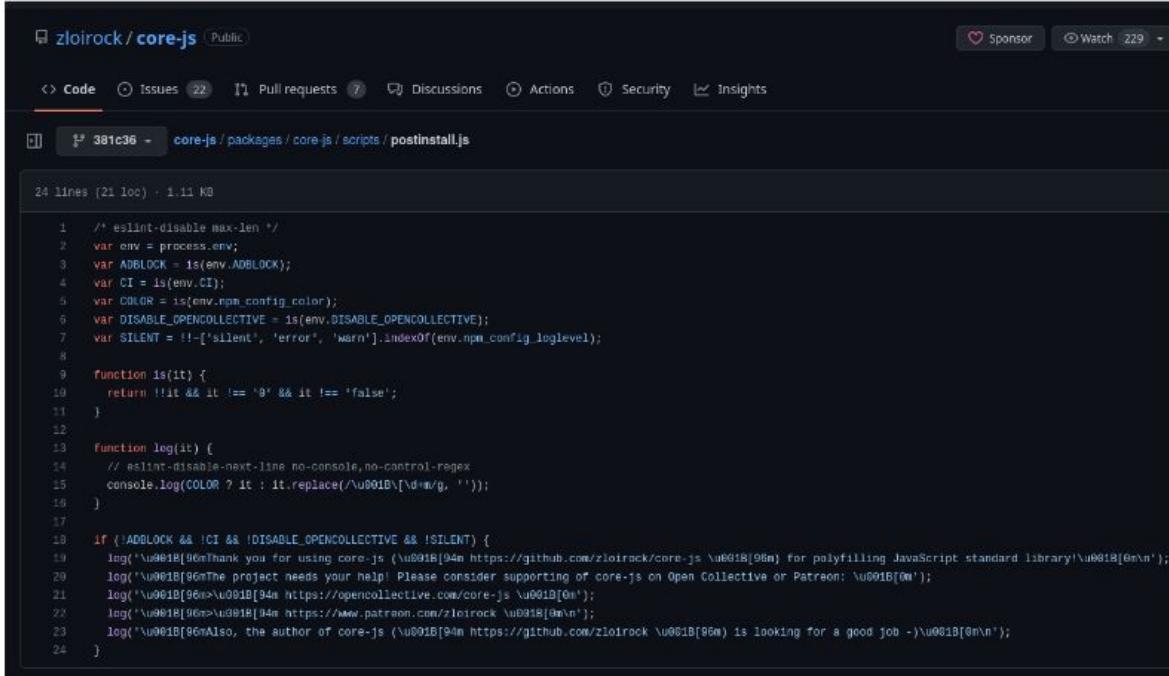


core-js

Modular standard library for JavaScript. Includes polyfills for ECMAScript up to 2023: promises, symbols, collections, iterators, typed arrays, many other features, ECMAScript proposals, some cross-platform WHATWG / W3C features and proposals like URL. You can load only required features or use it without global namespace pollution.

tl;dr: useful thing quite a few packages depend on

in june of 2019, the author of core-js added this to the 'post install' script within the package:

A screenshot of a GitHub repository page for 'zloirock / core-js'. The repository has 22 issues, 7 pull requests, and 229 watchers. The 'Code' tab is selected, showing the contents of the 'core-js/packages/core-js/scripts/postinstall.js' file. The file contains 24 lines of code, totaling 1.11 KB. The code is a script that logs messages to the console based on environment variables like ADBLOCK, CI, and COLOR. It includes logic to determine if a message is an error or warning, and it checks the npm config loglevel. It also includes a check for the Open Collective logo and links to the project's GitHub page and Open Collective page.

```
/* eslint-disable max-len */
var env = process.env;
var ADBLOCK = is(env.ADBLOCK);
var CI = is(env.CI);
var COLOR = is(env.npm_config_color);
var DISABLE_OPENCOLLECTIVE = is(env.DISABLE_OPENCOLLECTIVE);
var SILENT = !['silent', 'error', 'warn'].indexOf(env.npm_config_loglevel);

function is(it) {
  return !!it && it !== '0' && it !== 'false';
}

function log(it) {
  // eslint-disable-next-line no-console,no-control-regex
  console.log(COLOR ? it : it.replace(/\u001B\[\\d+m/g, ''));
}

if (!ADBLOCK && !CI && !DISABLE_OPENCOLLECTIVE && !SILENT) {
  log(`\u001B[96mThank you for using core-js (\u001B[94m https://github.com/zloirock/core-js \u001B[96m) for polyfilling JavaScript standard library!\u001B[0m\n`);
  log(`\u001B[96mThe project needs your help! Please consider supporting of core-js on Open Collective or Patreon: \u001B[0m`);
  log(`\u001B[96m>\u001B[94m https://opencollective.com/core-js \u001B[0m`);
  log(`\u001B[96m>\u001B[94m https://www.patreon.com/zloirock \u001B[0m\n`);
  log(`\u001B[96mAlso, the author of core-js (\u001B[94m https://github.com/zloirock \u001B[96m) is looking for a good job -)\u001B[0m\n`);
}
```

```
if (!ADBLOCK && !CI && !DISABLE_OPENCOLLECTIVE && !SILENT) {
  log(`\u001B[96mThank you for using core-js (\u001B[94m https://github.com/zloirock/core-js
  \u001B[96m) for polyfilling JavaScript standard library!\u001B[0m\n`);
  log(`\u001B[96mThe project needs your help! Please consider supporting of core-js on
  Open Collective or Patreon: \u001B[0m`);
  log(`\u001B[96m>\u001B[94m https://opencollective.com/core-js \u001B[0m`);
  log(`\u001B[96m>\u001B[94m https://www.patreon.com/zloirock \u001B[0m\n`);
  log(`\u001B[96mAlso, the author of core-js (\u001B[94m https://github.com/zloirock
  \u001B[96m) is looking for a good job -)\u001B[0m\n`);
}
```

```
> node scripts/postinstall || echo "ignore"
(node:9170) MaxListenersExceededWarning: Possible EventEmitter memory leak detected. 11 SIGINT listeners added to [process]. Use emitter.setMaxListeners() to increase limit
Thank you for using core-js ( https://github.com/zloirock/core-js ) for polyfilling JavaScript standard library!
The project needs your help! Please consider supporting of core-js on Open Collective or Patreon:
> https://opencollective.com/core-js
> https://www.patreon.com/zloirock
Also, the author of core-js ( https://github.com/zloirock ) is looking for a good job -)

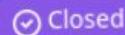
> core-js-pure@3.1.4 postinstall /src/stilltbd/clapp-webapp.wok/noob/node_modules/core-js-pure
> node scripts/postinstall || echo "ignore"

Thank you for using core-js ( https://github.com/zloirock/core-js ) for polyfilling JavaScript standard library!
The project needs your help! Please consider supporting of core-js on Open Collective or Patreon:
> https://opencollective.com/core-js
> https://www.patreon.com/zloirock
Also, the author of core-js ( https://github.com/zloirock ) is looking for a good job -)

> core-js@2.6.9 postinstall /src/stilltbd/clapp-webapp.wok/noob/node_modules/wait-on/node_modules/core-js
> node scripts/postinstall || echo "ignore"

Thank you for using core-js ( https://github.com/zloirock/core-js ) for polyfilling JavaScript standard library!
The project needs your help! Please consider supporting of core-js on Open Collective or Patreon:
> https://opencollective.com/core-js
> https://www.patreon.com/zloirock
Also, the author of core-js ( https://github.com/zloirock ) is looking for a good job -)
```

Get rid of postinstall message #548



Closed · jpike88 opened this issue on May 19, 2019 · 166 comments



jpike88 commented on May 19, 2019 · edited

...

My installation log isn't for advertising space. If you want donations, please do the right thing and promote yourself in places where appropriate. My logs are long enough as it is without this.

The right thing to do is to remove it.

EDIT: An unmaintained fork is available: [core-js-without-ads](#)



201



635



6



41



12

the notice was removed
from the repository

and some time passed ...

Did the author of core-js ever find a good job? #708

Closed

jannik-mohemian opened this issue on Nov 28, 2019 · 16 comments



jannik-mohemian commented on Nov 28, 2019

...

👉 I'm interested in whether you ever found a good job through the post-install message - it's been a while since this all started...



103



84



23



18



20



fuzzy76 commented on Jul 4, 2020

He is in prison. See #767



28



13



6

as of yesterday, the author of core-js posted a lengthy update called "So, what's next?"

<https://github.com/zloirock/core-js/blob/master/docs/2023-02-14-so-whats-next.md>

The screenshot shows a GitHub repository page for 'zloirock / core-js'. The 'Code' tab is selected, showing the file structure. A file named '2023-02-14-so-whats-next.md' is highlighted in the 'docs' directory. To the right, the contents of this file are displayed in a code editor window. The file begins with a logo consisting of a stylized insect and the text 'core-js'.

```
core-js / docs / 2023-02-14-so-whats-next.md
nanok Move donation information to the top (#1189) 45202c · 6 minutes ago
Preview Code Blame 568 lines (309 diff) · 74.4 KB
```

core-js

So, what's next?

Hi, I am (@zloirock) a full-time open-source developer. I don't like to write long posts, but it seems this is high time to do it. Initially, this post was supposed to be a post about the start of active development of the new major version of `core-js` and the roadmap [it was moved to the second half], however, due to recent events, became a really long post about many different things... I'm fucking tired. Free open-source software is fundamentally broken. I could stop working on this silently, but I want to give open-source one last chance.

sadly, these two stories
are hardly isolated incidents



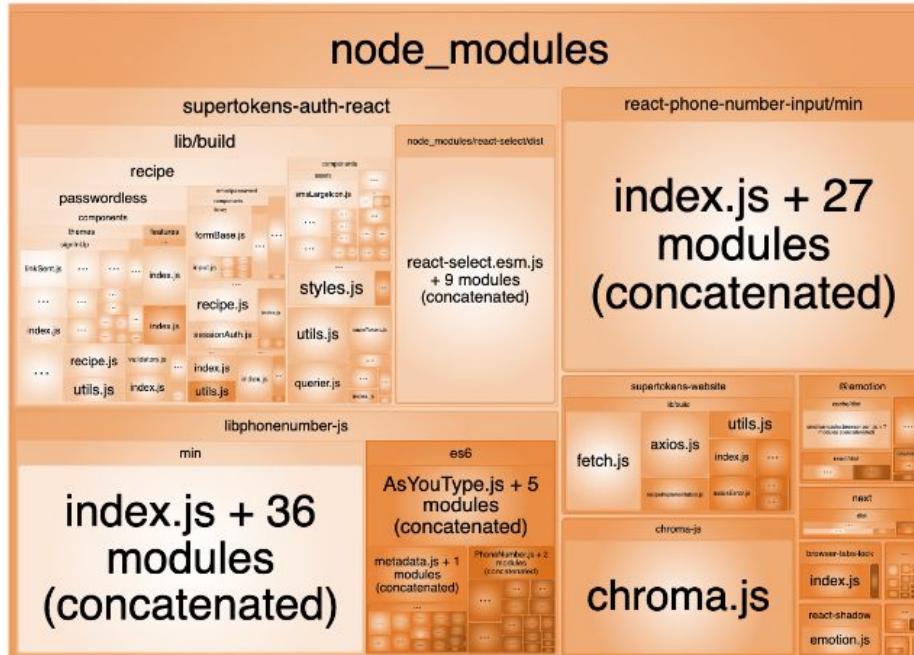
npm ecosystem

problems with the web:

single page
applications

```
fish /home/jack/repos/scratch/what-the-web/terminology/react-example/dist
jack@pop-os ~/r/s/w/t/r/dist> bat index.html
File: index.html
1  <!DOCTYPE html>
2  <html lang="en">
3      <head>
4          <meta charset="UTF-8" />
5          <link rel="icon" type="image/svg+xml" href="/vite.svg" />
6          <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7          <title>Vite + React</title>
8          <script type="module" crossorigin src="/assets/index-44ba7f54.js"></script>
9      </head>
10     <body>
11         <div id="root"></div>
12
13     </body>
14 </html>
jack@pop-os ~/r/s/w/t/r/dist> []
```


large client bundles!





Performance

Values are estimated and may vary. The [performance score is calculated](#) directly from these metrics. [See calculator.](#)



0-49



50-89



90-100

deepcrawl

Home > Blog > Webinars

Published on 20th December 2021 by Deepcrawl Editorial Team

Webinar On-Demand:
Human First — Digital
Ethics in Marketing &
Release

This website stores cookies on your computer. These cookies are used to collect information about how you interact with our website and remember you. We use this information in order to improve and customize your browsing experience and for analytics and metrics about our visitors both on this website and other media. To find out more about the cookies we use, see our Privacy Policy.

If you decline, your information won't be tracked when you visit this website. A single cookie will be used in your browser to remember your preference not to be tracked.

Accept Decline

problems with the web:

loading
spinners

 Fakebooks

Dashboard

Accounts

Sales

Expenses

Reports

Sales



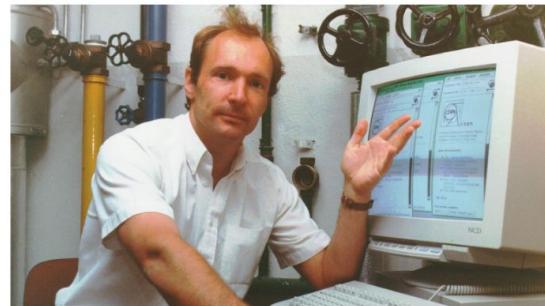
i'm looking at you, ClickUp ...



how did we get here?

The screenshot shows a Firefox browser window displaying the original homepage of the World Wide Web project. The title bar reads "The World Wide Web project". The main content area has a white background with black text. At the top left is the heading "World Wide Web". Below it is a paragraph about the W3 initiative. A section titled "What's out there?" lists links to subjects, servers, and help. Another section, "Software Products", lists various W3 components like Line Mode, Viola, NeXTStep, Servers, Tools, Mail robot, and Library. Further sections include Technical (protocols, formats, program internals), Bibliography (paper documentation), People (list of people), History (history of the project), How can I help? (support the web), and Getting code (anonymous FTP). The URL in the address bar is "info.cern.ch/hypertext/WWW/T1".

Tim Berners-Lee



HTML draft version timeline

October 1991

HTML Tags,^[6] an informal CERN document listing 18 HTML tag

June 1992

First informal draft of the HTML DTD,^[32] with seven^{[33][34][35]} s November 20, November 22)

November 1992

HTML DTD 1.1 (the first with a version number, based on RCS n

June 1993

Hypertext Markup Language^[36] was published by the IETF IIIR V replaced by a second version^[37] one month later.

November 1993

HTML+^[2] was published by the IETF as an Internet Draft and wa 1994.^[38]

November 1994

First draft (revision 00) of HTML 2.0 published by IETF itself^[39] (c RFC 1866^[2] in November 1995.^[41]

April 1995 (authored March 1995)

HTML 3.0^[42] was proposed as a standard to the IETF, but the pro included many of the capabilities that were in Raggett's HTML+ p complex mathematical formulas.^[43]

W3C began deve' its own Arena browser as a test bed f HTML 3 and Cascading Style Sheets,^{[44][45][46]} but HTML 3.0 was considered very large at 150 pages and the pace of browser development, as well as the number of interested parties, had o sources of the IETF.^[13] Browser vendors, including Microsoft and Netscape at the time, chose to implement different subsets of HT 3's draft features as well as to introduce their own extensions to it.^[13] (see *Browser wars*). These included extensions to control stylistic aspects of documents, such as color, texture, font size, and organization, which were definitely outside the scope of a language when their only intent was to specify how a document would be presented on a screen. Even a W3C Fellow for many years, has commented for example: "To a certain extent, Microsoft built its business on the Web by extending HTML features."^[13]



JavaScript

```
<form>
</form>
<div>
</div>
<div>
<form method="post" action="#" id="formvalue" onkeyup="drawChart()"/>
</form>
</div>
</div>

<script type="text/javascript" src="https://www.google.com/jsapi"></script>
<script type="text/javascript">
var bid = 43;
var ask = 21;

google.load("visualization", "1", {packages:['corechart']});
google.setOnLoadCallback(drawChart);
function drawChart() {
  var data = google.visualization.arrayToDataTable([
    ['Price', 'Quantity'],
    ['Value #1', 100],
    ['Value #2', ask],
    ['Value #3', ask]
  ]);
}
```

Screenshot of JavaScript source code with HTML

Paradigm

Multi-paradigm: event-driven, functional, imperative, procedural, object-oriented programming

Designed by

Brendan Eich of Netscape initially; others have also contributed to the ECMAScript standard

First appeared

December 4, 1995; 27 years ago^[1]

August 18, Nov

1 1.0), an info

ugh proposal

: Markup La

, that fin:

ember 1

: flow ar

Cascading Style Sheets (CSS)



The official logo of the latest version,
CSS 3

Filename extension

.css

Internet media type

text/css

Uniform Type Identifier (UTI)

public.css

Developed by

World Wide Web Consortium (W3C)

Initial release

17 December 1996; 25 years ago

Server-side rendering!



PHP: PHP Hypertext Preprocessor

PHP: Personal Home Page

First appeared: June 8, 1995; 27 years ago

One of the most significant breakthroughs for the web was going from this:

```
<html>
  <body>
    This document has been prepared ahead of time.
    Regards.
  </body>
</html>
```

To having easily embedded dynamic values:

```
<html>
  <body>
    Y2K? <?php echo time(); ?>
  </body>
</html>
```

Java applet

From Wikipedia, the free encyclopedia
(Redirected from [Java applets](#))

Java applets were small applications written in Java, a programming language, or another language that compiles to Java bytecode, and delivered via Java bytecode. The user launched the Java applet in a browser, and the applet was then executed within a process separate from the web browser. Java applets could appear in a frame of the web page, or in a separate window, Sun's AppletViewer, or a stand-alone tool.



ActiveX



Original author(s) Microsoft

Developer(s) Microsoft

Initial release 1996; 26 years ago

Adobe Flash



Developer(s)

Harman (2021-present for enterprise users)
[1][2][3][4]

Zhongcheng (2017-present in China)[5]

Adobe Inc.
(2005–2020)

Macromedia
(1996–2005)

FutureWave
(1993–1996)

Web browsers, iOS (via

Target



Initial release date: August 2004

django

Initial release date: July 21, 2005



Initial release date: May 27, 2009

the client side got busier over time, but browsers were frustrating when it came to shared consistent functionality

so, tools like jQuery and MooTools were created to help bridge the gap between browsers and give useful utilities for developers



Initial release date: August 26, 2006



Initial release: March 7, 2007; 15 years ago



ember[®]



METEOR





World Wide Web Consortium (W3C)

Founder: Tim Berners-Lee
Founded: October 1, 1994



Web Hypertext Application Technology Working Group (WHATWG)

Founded: June 4, 2004



https://developer.mozilla.org/en-US/docs/Web/HTML/Element/video

MDN Plus now available in your country! Support MDN and make it your own. Learn more ➔

mdn web docs References Guides MDN Plus

References > HTML > Elements > <video>

```
<time>
<title>
<tr>
<track>
<video>
<u>
<ul>
<var>
<wbr>
<xmp>

```

► Global attributes
► Attributes
► <input> types

<video>: The Video Embed element

The `<video>` HTML element embeds a media player which supports video playback into the document. You can use `<video>` for audio content as well, but the `<audio>` element may provide a more appropriate user experience.

Try it

HTML Demo: <video>

RESET

HTML CSS

```
1 <video controls width="250">
2   <source src="/media/cc0-videos/flower.webm"
3     type="video/webm">
4   <source src="/media/cc0-videos/flower.mp4"
5     type="video/mp4">
6 
```

OUTPUT

This screenshot shows a browser window displaying the MDN Web Docs page for the <video> element. The URL in the address bar is https://developer.mozilla.org/en-US/docs/Web/HTML/Element/video. The page header includes a 'MDN Plus' promotional message and navigation links for 'mdn web docs', 'References', 'Guides', and 'MDN Plus'. The main content area starts with a list of other HTML elements like <time>, <title>, <tr>, <track>, <u>, , <var>, <wbr>, and <xmp>. The <video> element is highlighted with a red background. Below this is a detailed description of the <video> element, mentioning its use for video playback and its compatibility with audio content. The 'Try it' section at the bottom features a code editor with sample HTML code and a preview window showing a flower video thumbnail.



thing aren't that bad

thing aren't that bad:

we (seem) to have
gotten better about
dependencies

Unpublishing packages from the registry

As a package owner or collaborator, if your package has no dependents, you can permanently remove it from the npm registry by using the CLI. You can [unpublish](#) within 72 hours of the initial publish; beyond 72 hours, you can still unpublish your package if [it meets certain criteria](#).

These criteria are set to avoid damaging the JavaScript package ecosystem. If you cannot unpublish your package, you can [deprecate it instead](#).



TIDELIFT

Maximize the health and security of the
open source powering your applications



A proactive approach to managing open source

Tidelift provides the tools, data, and strategies that help organizations assess risk and improve the health, security, and resilience of the open source used in their applications.

Backed by maintainers

Tidelift partners ensure the open source used in your enterprise stays healthy.

The screenshot shows the npm Docs homepage with a dark header featuring the npm logo and "npm Docs". Below the header is a search bar labeled "Search npm Docs". The main content area has a sidebar on the left with links: "About npm", "Getting started", "Packages and modules", and "Integrations". To the right of the sidebar, there's a section titled "npm-fund" with the subtext "Retrieve funding information". At the bottom, there's a dropdown menu for "Select CLI Version" with the option "Version 7.24.2 (Legacy Release)".



Developer loved, Security trusted.

Find and automatically fix vulnerabilities in your code, open source dependencies, containers, and infrastructure as code – all powered by Snyk's industry-leading security intelligence.

Get started with a free forever account, and scale up if needed.

A screenshot of a GitHub repository page for "Dependabot". The page shows a summary of dependency testing results, including a terminal window showing "snyk test" output and a GitHub status bar indicating 3 dependencies tested, 2 issues found, and 2 vulnerabilities. The GitHub interface includes a search bar, navigation buttons, and tabs for "Pull requests", "Issues", and "Codespaces". The repository card for "Dependabot" features a blue hexagonal icon with a white robot head, the name "Dependabot", a description of "Automated dependency updates built into GitHub", and stats like 586 followers and 11 people. Navigation links at the bottom include "Overview", "Repositories 23", "Projects", "Packages", and "People".

Dependabot

Automated dependency updates built into GitHub

586 followers United States of America https://github.com/features/security

Overview Repositories 23 Projects Packages People 11

but, sometimes the signal to noise ratio isn't that great ...

A screenshot of a Twitter post from Tom MacWright (@tmcw). The post contains the following text:

repeating that 99% of automated "dependency security" warnings in the javascript world are fake, totally unimportant, and cause useless busywork for maintainers

12:20 PM · Dec 7, 2022

A screenshot of a Twitter reply from Tim Oram (@timoram). The reply contains the following text:

This is not a new thought for me, but I am starting continuing to think services like Synk are a net-bad for our industry. Just hit this [issue](#) today as part of an update. Classified as a high severity issue, but in reality, it was a case of an undefined causing an error to be thrown.... (edited)

A screenshot of a GitHub repository interface showing a list of pull requests. The list includes the following items:

- Bump express from 4.16.4 to 4.17.3 in /server dependencies
- Bump qs and express in /server dependencies
- Bump engine.io and socket.io in /server dependencies
- Bump moment from 2.24.0 to 2.29.4 in /server dependencies
- Bump ajv from 6.8.1 to 6.12.3 in /server dependencies
- Bump node-sass from 4.11.0 to 7.0.0 in /client dependencies
- Bump three from 0.101.1 to 0.137.0 in /client dependencies
- Bump socket.io-parser from 3.3.0 to 3.3.2 in /server dependencies
- Bump lodash from 4.17.19 to 4.17.21 in /server dependencies

The list is filtered to show 9 open and 6 closed pull requests. The author is dependabot.

thing aren't that bad:

pnpm helps
solve the many
node_modules issue

Unity Benchmarks

7.x ▾  English ▾ 

pNpm

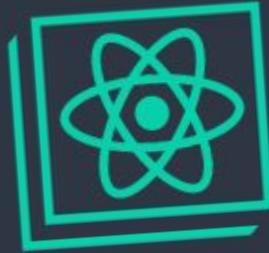
Fast, disk space efficient package manager

Get Started

 Star 20,906

thing aren't that bad:

the build system
story is a lot better
than it used to be



Create React App

Set up a modern web app by running one command.

[Get Started](#)



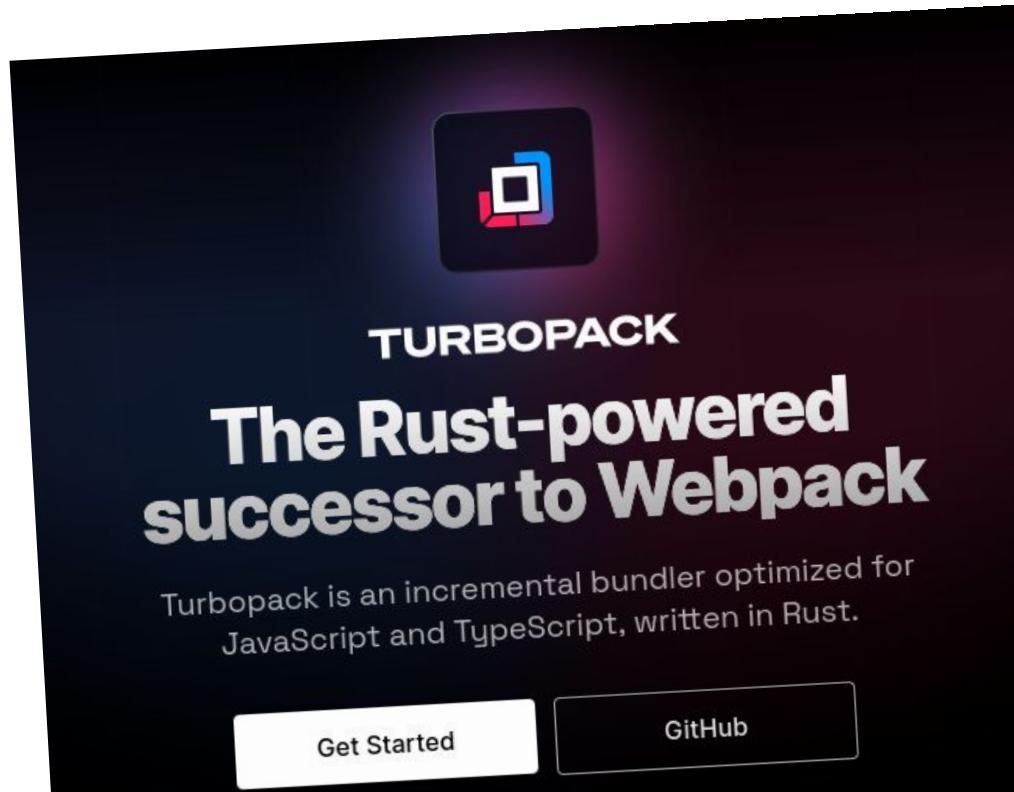
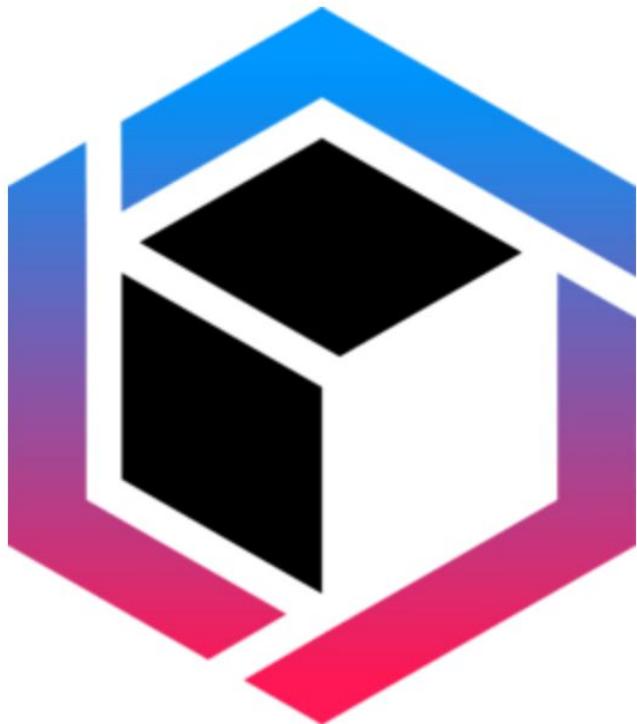
```
fish /home/jack/repos/scratch/what-the-web/terminology
jack@pop-os ~/r/s/w/terminology> npm create vite@latest
✓ Project name: ... react-example
✓ Select a framework: > React
✓ Select a variant: > JavaScript

Scaffolding project in /home/jack/repos/scratch/what-the-web/terminology/react-example...

Done. Now run:

  cd react-example
  npm install
  npm run dev

jack@pop-os ~/r/s/w/terminology> █
```



The landing page features a dark background with a glowing purple-to-blue gradient at the top. In the center is a black square icon with a white stylized 'T' or 'D' shape inside. Below the icon, the word "TURBOPACK" is written in white capital letters. Underneath that, the text "The Rust-powered successor to Webpack" is displayed in large, bold, white capital letters. At the bottom, a paragraph explains: "Turbopack is an incremental bundler optimized for JavaScript and TypeScript, written in Rust." Two white rectangular buttons with rounded corners are at the bottom: "Get Started" on the left and "GitHub" on the right.

TURBOPACK

**The Rust-powered
successor to Webpack**

Turbopack is an incremental bundler optimized for JavaScript and TypeScript, written in Rust.

Get Started

GitHub

thing aren't that bad:

you can still have
a fun dev experience
without shipping any js



A astro

thing aren't that bad:

the industry seems
less focused on
single page apps

NEXT.js



tip: try out Astro

of all of the static site generators out there right now, Astro has to be my favorite, both because of how simple it is out of the box, but also all the features it packs in too



src/components/DynamicHtml.astro

```
---
```

```
const items = ["Dog", "Cat", "Platypus"];
```

```
---
```

```
<ul>
  {items.map((item) => (
    <li>{item}</li>
  ))}
</ul>
```

src/components/MyComponent.astro

```
---
```

```
import SomeAstroComponent from '../components/SomeAstroComponent.astro';
import SomeReactComponent from '../components/SomeReactComponent.jsx';
import someData from '../data/pokemon.json';

// Access passed-in component props, like `<X title="Hello, World" />`
const {title} = Astro.props;
// Fetch external data, even from a private API or database
const data = await fetch('SOME_SECRET_API_URL/users').then(r => r.json());
---
```

```
<!-- Your template here! -->
```



Designed to feel familiar

Use the tools you already love.

Bring your own component framework. Astro offers first-class support for front-end tools like React, Vue, Svelte, and Tailwind CSS. Avoid confusing configuration and get started in seconds: just type `astro add`.

[Get started with integrations >](#)



```
---
```

```
import Mug from "../components/Mug.astro";
import MainLayout from "../layouts/MainLayout.astro";
---
```

```
<MainLayout title="about">
  <div class="mb-4">
    <Mug />
  </div>
  <ul>
    <li>
      <h1>hi, i'm jack</h1>
      <li class="pt-3">
        Currently i'm working at <a href="http://>, working on super neat web-based geo
      </li>
      <li class="pt-3">
        previously i've worked for
        <a href="https://bluedropism.com/">Blue
          href="https://www.colabsoftware.com/>, as a full stack developer
        </li>
      <li class="pt-3">
        instructor at <a href="https://www.get-c
      </li>
      <li class="text-sm pt-3">
        <span class="italic opacity-50">(planning
        undergrad at <a href="https://www.mun.ca/">
        <span class="italic opacity-50">(eventually
      </li>
      <li class="text-sm pt-3">
        active with <a href="https://ctsnl.ca/">CT
        <a href="https://www.meetup.com/NDevMeetup/>
      </li>
      <li class="text-sm pt-3">
        building things with pals in
        <a href="https://github.com/fogo-sh">fogo.sh
      </li>
      <li class="text-sm pt-3">from wales 🇬🇧, vibing in canada 🇨🇦
    </li>
  </ul>
</MainLayout>
```

The screenshot shows a web browser window with the URL <https://jackharry.dev>. The page content is as follows:

jack harrhy
about ←
now
projects

[github](#)
[twitter](#)
[instagram](#)
[steam](#)
[youtube](#)

hi, i'm jack
currently i'm working at [C-CORE](#), working on super neat web-based geospatial software
previously i've worked for [Bluedrop](#), and [CoLab Software](#), as a full stack developer

instructor at [Get Coding](#)
(planning on) finishing up my undergrad at [MUN](#) (eventually)
active with [CTSNL](#), and sometimes [NDev](#)
building things with pals in [fogo.sh](#)
from wales 🇬🇧, vibing in canada 🇨🇦
[my resume](#), built as a [modular latex resume](#)
contact me:
me@jackharry.com
[jack arthur null](https://i.stack.imgur.com/7539.html)

tip: use Vite

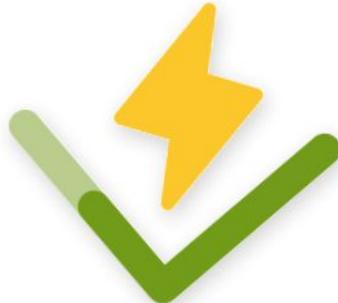
if you are planning on starting a new frontend project, or have an existing frontend project you want to speed up / move away from create-react-app/webpack, Vite is the way to go



The screenshot shows the official Vite website homepage. The header features the Vite logo and navigation links for Guide, Config, Plugins, Resources, Version, and Help. The main title is "Vite Next Generation Frontend Tooling". Below it, a subtitle reads "Get ready for a development environment that can finally catch up with you." Three call-to-action buttons are visible: "Get Started", "Why Vite?", and "View on GitHub". The page is divided into three main sections: "Instant Server Start" (with a lightbulb icon), "Lightning Fast HMR" (with a lightning bolt icon), and "Rich Features" (with a wrench and gear icon). Each section includes a brief description of its functionality.

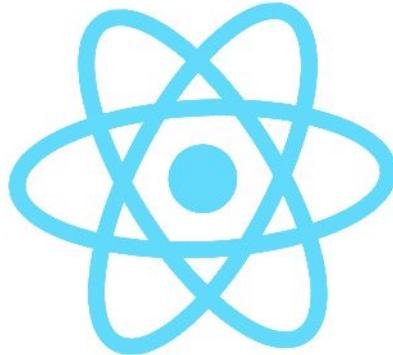
tip: use Vitest

in the same ecosystem as Vite, is Vitest, a test runner that comes out of the box with support for most things you could think of without configuration. Vitest is jest compatible, so if you're already using that, you're set.



React is still a safe bet as framework of choice

while there are other choices out there, and some newer frameworks with some pretty cool demos, as far as what I and most would recommend picking as a framework for some time, and now, is React



The screenshot shows the React Docs beta website. At the top left is the React logo. The header reads "React Docs BETA". Below the header is a search bar and two navigation tabs: "Learn" (which is underlined) and "Reference". A sidebar on the left lists categories: "Get Started", "Installation", "Quick Start", "Learn React", "Describing the UI", "Adding Interactivity", "Managing State", "Escape Hatches", "About React", "Community", and "Blog". To the right of the sidebar are two main sections: "Quick Start" and "API Reference". The "Quick Start" section contains text about learning React with step-by-step explanations and interactive examples, along with a "Read More" button. The "API Reference" section contains text about looking up the API of React Hooks with color-coded signatures, also with a "Read More" button. At the bottom, there's a section titled "What is this site?", a note about rewriting the documentation, and a footer stating "This beta website contains the current draft of the new docs." and "How much content is ready?".

tip: use the new beta.reactjs.org site

the old docs still reference class components everywhere,
the beta docs (soon hopefully to replace the official docs)
are much more improved and modern in comparison

The screenshot shows the React Docs beta interface. The top navigation bar includes the React logo, 'React Docs', and a 'BETA' indicator. Below the navigation is a search bar and a sidebar with links to 'Learn', 'Reference', 'useLayoutEffect', 'useMemo', 'useReducer', 'useRef', 'useState' (which is highlighted in blue), 'useSyncExternalStore', 'useTransition', and 'APIs'. The main content area has a dark background and features a 'Reference' header. Underneath it, the 'useState(initialState)' API is shown. A callout text states: 'Call useState at the top level of your component to declare a state variable.' Below this is a code snippet:

```
import { useState } from 'react';

function MyComponent() {
  const [age, setAge] = useState(20);
  const [name, setName] = useState('Taylor');
  const [todos, setTodos] = useState(() => createTodos());
  // ...
}
```

Further down, a note says: 'The convention is to name state variables like [something, setSomething] using array destructuring.' There's also a link to 'See more examples below.' The 'Parameters' section lists the 'initialState' parameter with its description: 'The value you want the state to be initially. It can be a value of any type, but there is a special behavior for functions. This argument is ignored after the initial render.' Another bullet point explains: 'If you pass a function as initialState, it will be treated as an initializer function. It should be pure, should take no arguments, and should return a value of any type. React will call your initializer function when initializing the component, and store its return value as the initial state. See an example below.'

**tip: use TanStack Query
for any & all data fetching**

the application you have, likely has to do some async stuff to fetch data, which react doesn't give you a perfect handle for yet

(Suspense is on the way, but still not perfect, and hardly concerns itself with state to the degree TanStack Query does)

TanStack Query v4

Powerful asynchronous state management for TS/JS, React, Solid, Vue and Svelte

```
// 1. Import *useState* and *useEffect*
import React, {useState, useEffect} from 'react';
import './App.css';

function App() {
    // 2. Create our *dogImage* variable as well as the *setDogImage* function via useState
    // We're setting the default value of dogImage to null, so that while we wait for the
    // fetch to complete, we don't attempt to render the image
    let [dogImage, setDogImage] = useState(null)

    // 3. Create our useEffect function
    useEffect(() => {
        fetch("https://dog.ceo/api/breeds/image/random")
        .then(response => response.json())
        // 4. Setting *dogImage* to the image url that we received from the response above
        .then(data => setDogImage(data.message))
    },[])
}

return (
    <div className="App">
        /* 5. Using *dogImage* as the *src* for our image*/
        {dogImage && <img src={dogImage}></img>}
        </div>
    );
}

export default App;
```



a single fetch
within a useEffect,
is sadly not enough

```
function Todos() {
  const { isLoading, isError, data, error } = useQuery({
    queryKey: ['todos'],
    queryFn: fetchTodoList,
  })

  if (isLoading) {
    return <span>Loading...</span>
  }

  if (isError) {
    return <span>Error: {error.message}</span>
  }

  // We can assume by this point that `isSuccess === true`
  return (
    <ul>
      {data.map((todo) => (
        <li key={todo.id}>{todo.title}</li>
      )))
    </ul>
  )
}
```



basic api, you can
pass it a proper
async function, and
it handles quite a
bit for you!

Out of the box, TanStack Query is configured with **aggressive but sane** defaults. **Sometimes these defaults can catch new users off guard or make learning/debugging difficult if they are unknown by the user.** Keep them in mind as you continue to learn and use TanStack Query:

- Query instances via ``useQuery`` or ``useInfiniteQuery`` by default **consider cached data as stale**.

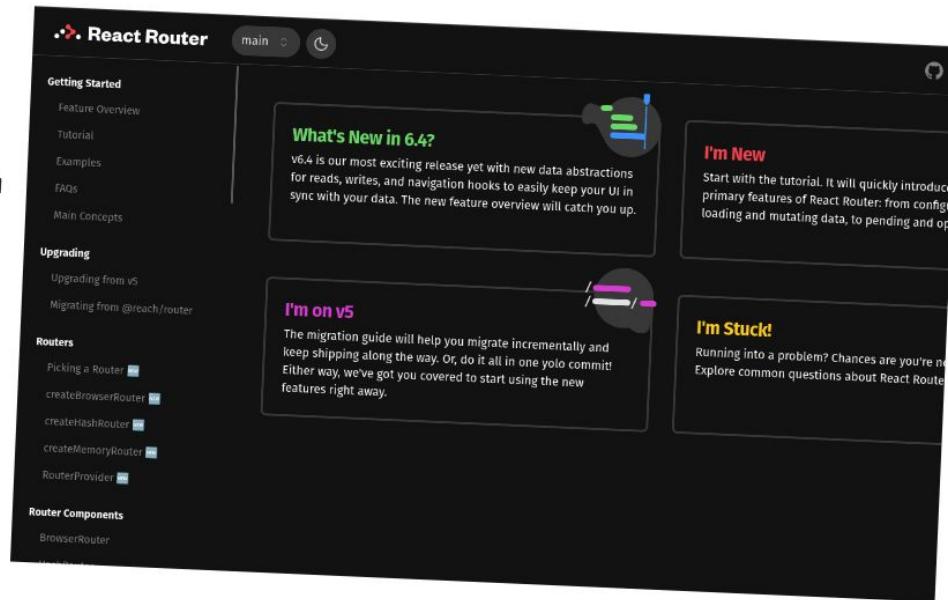
"To change this behavior, you can configure your queries both globally and per-query using the ``staleTime`` option. Specifying a longer ``staleTime`` means queries will not refetch their data as often"

- Stale queries are refetched automatically in the background when:
 - New instances of the query mount
 - The window is refocused
 - The network is reconnected
 - The query is optionally configured with a refetch interval

**tip: use React Router,
especially the 6.4+ features**

if you're writing a react app, chances are it already uses React Router to perform client side navigation, however there are some new features you can make use of that allows it to handle data loading/mutations :)

React Router



```
import { useLoaderData } from "react-router-dom";
import { getContacts } from "../contacts";

export async function loader() {
  const contacts = await getContacts();
  return { contacts };
}

export default function Contacts() {
  const { contacts } = useLoaderData();
  return (
    <>
      <div id="sidebar">
        <h1>React Router Contacts</h1>
        {/* Code to render contacts */}
      </div>
    </>
  );
}
```

look ma, component
with no loading state!

```
import { useLoaderData } from "react-router-dom";
import { getContacts } from "../contacts";

// ...
export async function action() {
  const contact = await createContact();
  return { contact };
}

export default function Contacts() {
  // ...
  return (
    <>
      <div id="sidebar">
        <h1>React Router Contacts</h1>
        {/* Code to render contacts */}
        <Form method="post">
          <button type="submit">New</button>
        </Form>
      </div>
    </>
  );
}
```

boring old forms, back
again in style!

use TypeScript



you should have a reason
you are not using TypeScript,
and it should be a good one,
if not you should figure out
a pathway for you/your team to
be using TypeScript





Kent C. Dodds
@kentcdodds

...

Unless I'm teaching you JavaScript fundamentals, everything I teach will be taught with TypeScript going forward.

TypeScript has won, and it's only a matter of time you're using it whether you like it or not.

10:53 PM · Feb 11, 2023 · 746.1K Views

tip: use Remix

while some would consider Next.js the bigger player here,
I myself am a big fan of the approach Remix to
server side React



Remix

Focused on web standards and modern web app UX, you're simply going to build better websites

Remix is a full stack web framework that lets you focus on the user interface and work back through web standards to deliver a fast, slick, and efficient user experience. Remix is built on top of Node.js and is designed to be easy to learn and use.

```
export async function getProject() { return {} }

export async function createProject() { const form = await document.querySelector('form').submit(); const response = await fetch('/api/projects', { method: 'POST', body: new FormData(form) }); const project = await response.json(); return project }

export default function Projects({ state }) { const projects = state.projects.map((project) => View) return <div>{projects}</div> }
```

```
● ● ●

export async function loader({ request }) {
  return getProjects();
}

export async function action({ request }) {
  const form = await request.formData();
  return createProject({ title: form.get("title") });
}

export default function Projects() {
  const projects = useLoaderData();
  const { state } = useTransition();
  const busy = state === "submitting";

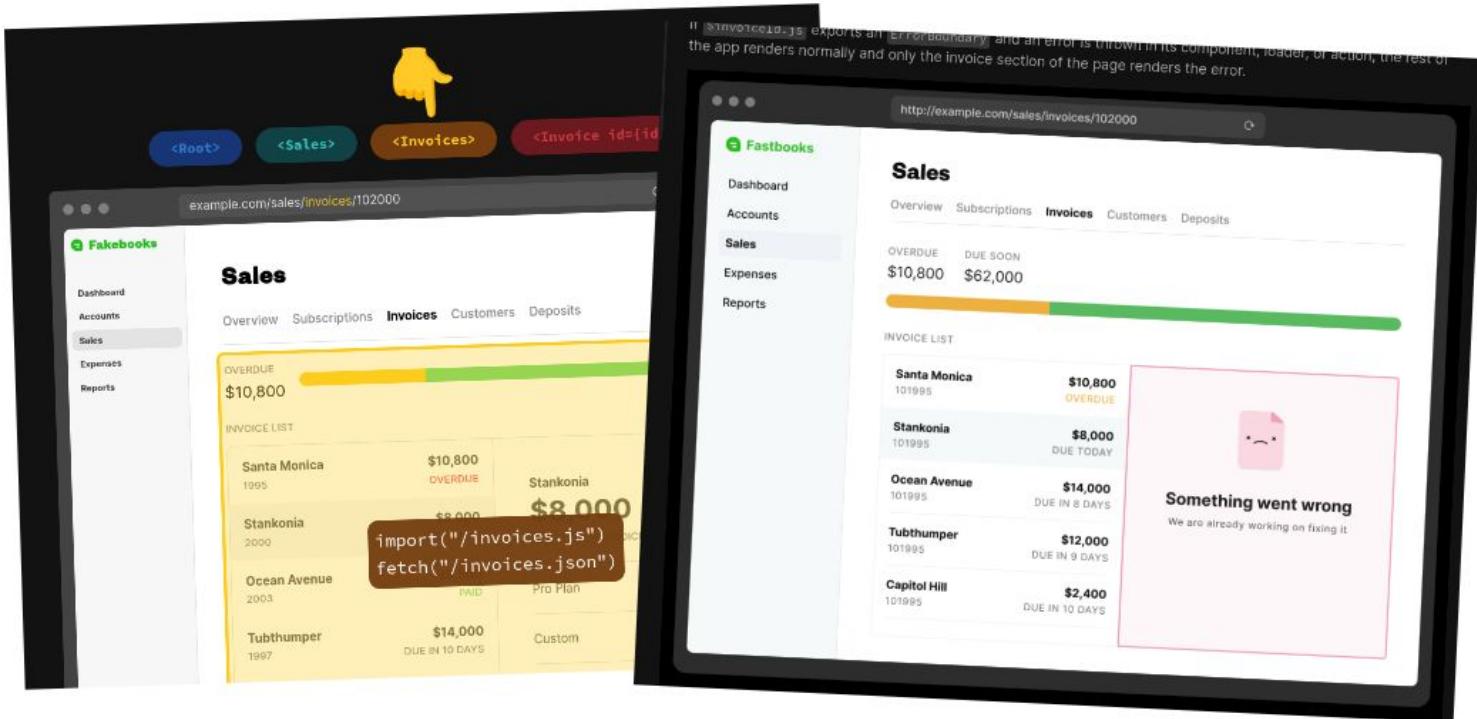
  return (
    <div>
      {projects.map((project) => (
        <Link to={project.slug}>{project.title}</Link>
      ))}

      <Form method="post">
        <input name="title" />
        <button type="submit" disabled={busy}>
          {busy ? "Creating..." : "Create New Project"}
        </button>
      </Form>
    </div>
  );
}
```

this looks quite similar to the previous react router example (mostly because its the same folk behind it),

HOWEVER, the loader and action functions you see here, are on the backend!

nested routing, nested error boundaries,
streaming, client only / server only code,
i could probably squeeze out a standalone Remix talk
in itself



tip: give Tailwind CSS a try,
its cursed, and you might love it

i haven't talked about styling a lot this talk, but i might drop a mention to Tailwind CSS as an interesting new approach to styling elements / building out a design system



The screenshot shows the official Tailwind CSS website. At the top, there's a navigation bar with the Tailwind CSS logo, a search bar, and links for "Docs", "Components", and "Blog". The main headline reads "Rapidly build modern websites without ever leaving your HTML." Below the headline, a subtext explains: "A utility-first CSS framework packed with classes like `flex`, `pt-4`, `text-center` and `rotate-90` that can be composed to build any design, directly in your markup." There are "Get started" and "Quick search..." buttons. A code snippet in a callout box shows how to create a figure component:

```
1 <figure class="bg-slate-100 rounded-xl p-8 dark:bg-slate-800">
2   
3   <div class="pt-2 space-y-4">
    <blockquote>
```

The screenshot shows a product page for a 'Classic Utility Jacket'. At the top, there are four navigation tabs: 'Simple', 'Playful', 'Elegant', and 'Brutalist'. Below the tabs is a grid pattern background. A woman wearing the jacket is shown on the left. The product title is 'Classic Utility Jacket' and the price is '\$110.00'. The status is 'In stock'. Size options include XS (selected), S, M, L, and XL. Buttons for 'Buy now' and 'Add to bag' are present, along with a heart icon for favorites. A note at the bottom states 'Free shipping on all continental US orders.'

```
1  <div class="flex font-sans">
2    <div class="flex-none w-48 relative">
3      
4    </div>
5    <form class="flex-auto p-6">
6      <div class="flex flex-wrap">
7        <h1 class="flex-auto text-lg font-semibold text-slate-900">Classic Utility Jacket</h1>
8        <div class="text-lg font-semibold text-slate-500">$110.00</div>
9        <div class="w-full flex-none text-sm font-medium text-slate-500">In stock</div>
10       <div class="flex items-baseline mt-4 mb-6 pb-6 border-b border-slate-500 rounded-md">
11         <div class="space-x-2 flex text-sm">
12           <label>
```

New Releases Top Rated Vincent's Picks

 **Prognosis Negative** ★ 2.66
PG-13 · 2021 · Comedy · 1h 46m
Simon Pegg, Zach Galifianakis

 **Rochelle, Rochelle** ★ 3.25
R · 2020 · Romance · 1h 56m
Emilia Clarke

 **Death Blow** ★ 4.95
18A · 2020 · Action · 2h 5m
Idris Elba, John Cena, Thandiwe Newton

Movies.js Nav.js NavItem.js List.js ListItem.js

```
1 import Nav from './Nav.js'
2 import NavItem from './NavItem.js'
3 import List from './List.js'
4 import ListItem from './ListItem.js'
5
6 export default function Movies({ movies }) {
7   return (
8     <div className="divide-y divide-slate-100">
9       <Nav>
10         <NavItem href="/new" isActive>New Releases</NavItem>
11         <NavItem href="/top">Top Rated</NavItem>
12         <NavItem href="/picks">Vincent's Picks</NavItem>
13       </Nav>
14       <List>
15         {movies.map((movie) => (
16           <ListItem key={movie.id} movie={movie} />
17         ))}
18       </List>
19     </div>
20   )
21 }
```

tip: use vim
keybindings btw



I'm Jack Harrhy

And this was

why the web

questions/comments/critiques?

interested in chatting?
feel free to reach out, always
interested in helping out
others with their technical
challenges :)



jackharry.dev



@jack@nfld.social



me@jackharry.com

slides will be available at
github.com/jackharry/why-the-web
sometime after the talk :)

again, thanks to
everyone involved!



tools shoutout



EXCALIDRAW



Google Slides