Making a webapp like it’s 2004

How I made an NYC subway mobile app

p.b. to
The Recurse Center
2024-02-15

Making a webapp like it's 2004

How I made an NYC subway mobile app

p.b. to
The Recurse Center
2024-02-15

Email Password Sign In

/ Realtime Data Feeds

Realtime Data Feeds

Most of the MTA's open data is available without authentication in our open data portal. Our realtime GTFS-rt feeds get heavier usage, so we ask that users authenticate in order to access them.

Documentation

GTFS Realtime overview

GTFS - RT Reference for the New York City Subway

Integrating our key into your app

Support and Questions

We don't have a dedicated developer support team, but we actively monitor the MTA Developer Google Group. Send questions and comments to the group.

Join the Group

Useful Links

Sign into your Account

Create an Account

Terms and Conditions

Using MTA Logos, Maps or Symbols

Our data feeds are free to use. But to use our logos, maps, symbols or other intellectual property, you need a license. The MTA protects its brand identity and intellectual property on behalf of the public. Revenues generated from licensing go back into our budget, helping provide services for our customers. If an MTA-approved mobile application is provided to the public free of charge, the MTA requires a license but will not charge a license fee for the use of our intellectual property. Commercial and ad-sponsored applications may be charged a license fee.

Find out how to apply for a license

Previously:

checkins > P.B. To

JAN 14

P.B. To (she/they) (W2'24) EDITED 12:57 PM

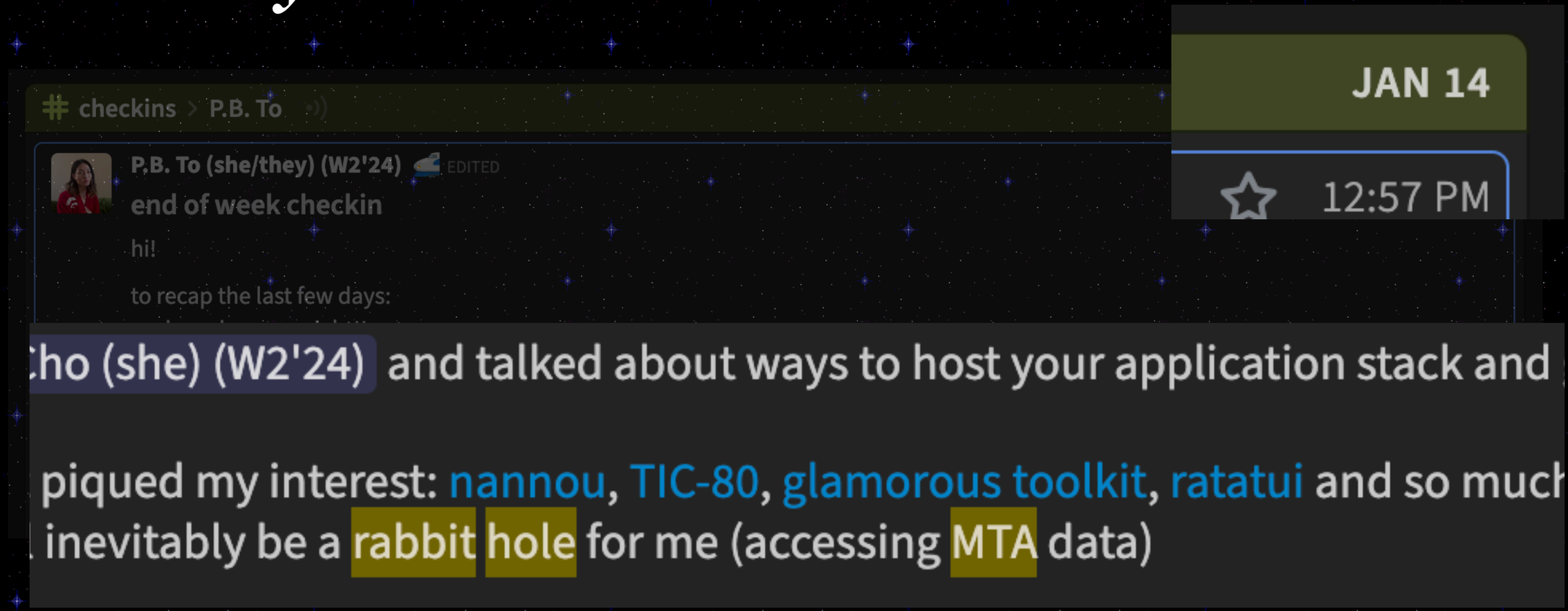
end of week checkin

hi!

to recap the last few days:

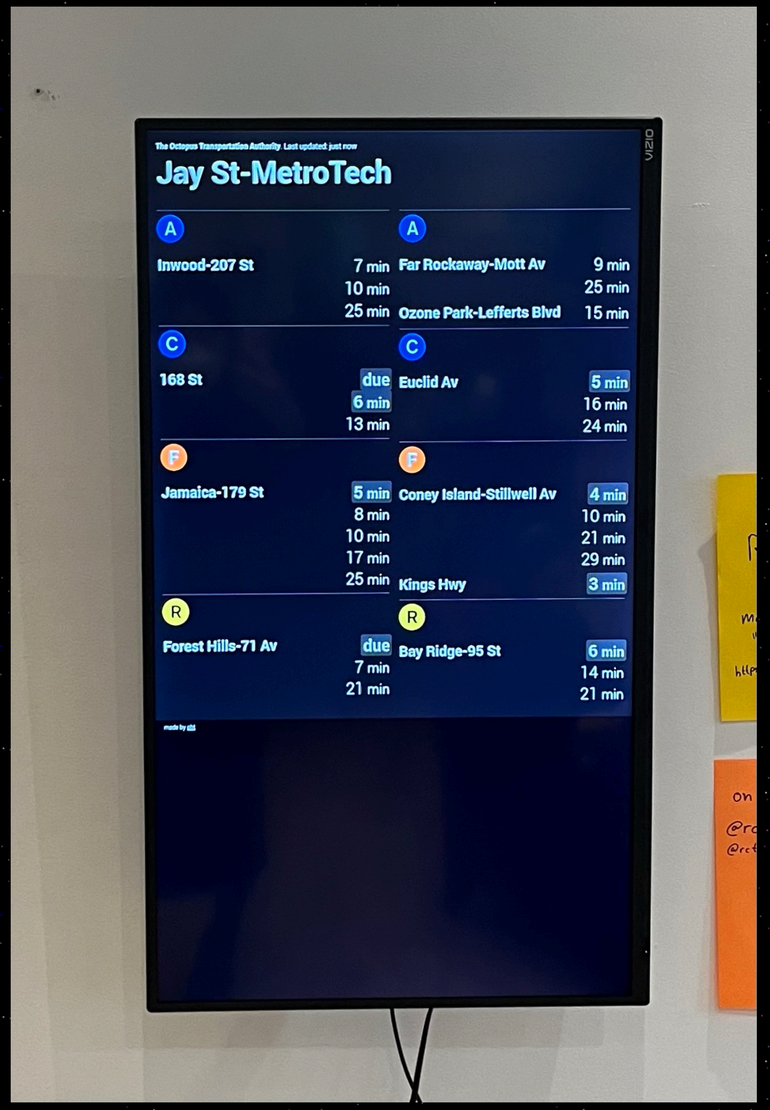
- board games night!!
- conversation with @Faisal Al Qasimi (he) (W1'23) about (edit: wait maybe i should actually write this in huh) having too much to learn
- paired with @Justina Cho (she) (W2'24) and talked about ways to host your application stack and getting started with React
- some things i liked that piqued my interest: nannou, TIC-80, glamorous toolkit, ratatui and so much more
- got started on what will inevitably be a rabbit hole for me (accessing MTA data)

Previously:

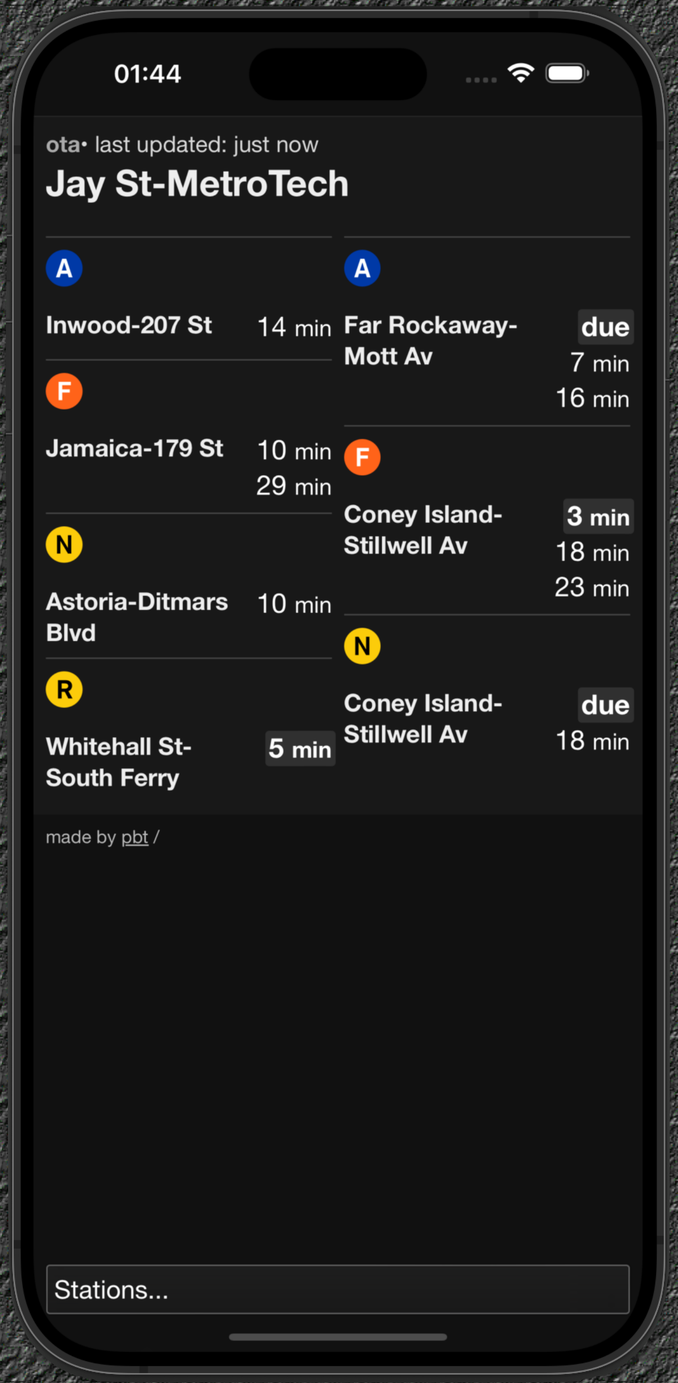

JAN 14
12:57 PM
checkins > P.B. To
P.B. To (she/they) (W2'24) EDITED
end of week checkin
hi!
to recap the last few days:
:ho (she) (W2'24) and talked about ways to host your application stack and
piqued my interest: nannou, TIC-80, glamorous toolkit, ratatui and so much
inevitably be a rabbit hole for me (accessing MTA data)

A few weeks later…

- I designed an MTA arrival board app for **RCTV**
- The initial goal I set for myself was to display arrival times for a single station in a clean manner
- But then I asked myself… ***what if I put this on my phone…***



A few more weeks later…



Demo!

In 2004, we:

- Assumed that internet is slow and spotty
- Assumed that you will intermittently go offline
- Assumed that you didn’t have JavaScript enabled

On the subway, you:

- Assume that internet is slow and spotty
- Assume that you will intermittently go offline
- … ok you probably have JS enabled on your phone

Architecture

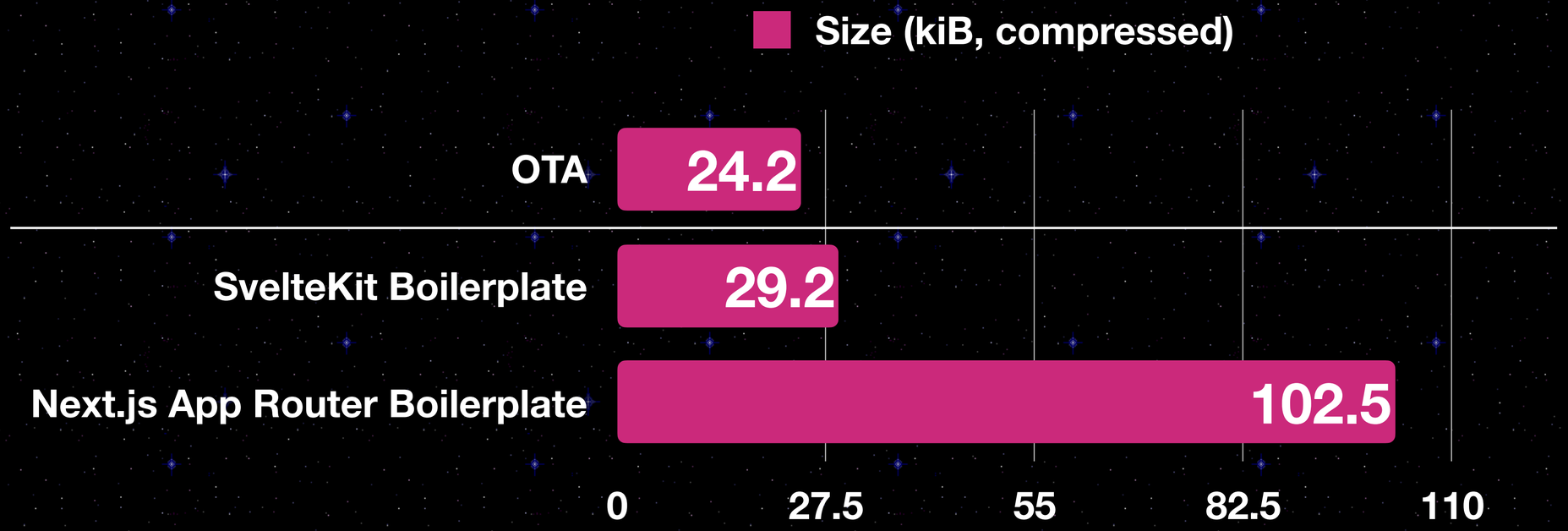
Backend

- A **flask app** that serves using **asyncio**
- In-memory cacheing because I can only do so much

Frontend

- **HTMX** on the front-end plus some vanilla JS for interactivity
- CSS

First-Load Size (code only, no assets)



Source: pagespeed insights

Progressive enhancement

- **Make sure it works with HTML & CSS first; use JavaScript later**

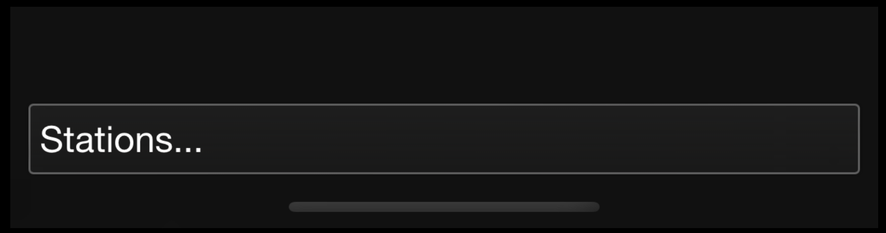
Implications:

- You get server-side rendering for free, because *you have to*
- *Your client-side code is mostly used to drive views, so it's very lightweight*
- You use *hypertext as the engine of application state*

HATEOAS

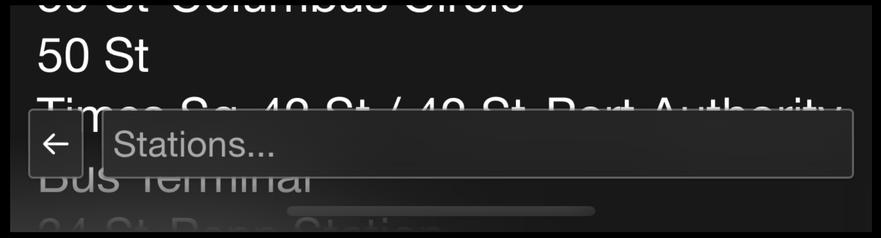
(Hypertext As The Engine Of Application State)

/?stop_ids=M06

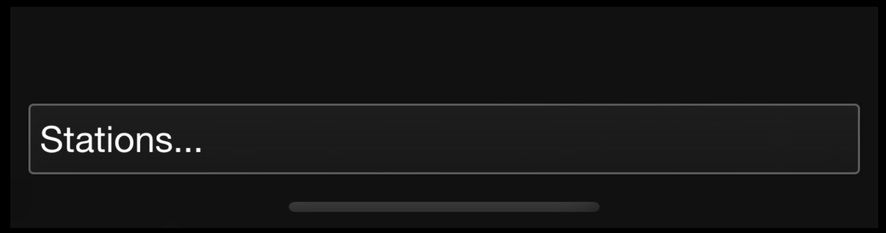


/?stop_ids

/stations

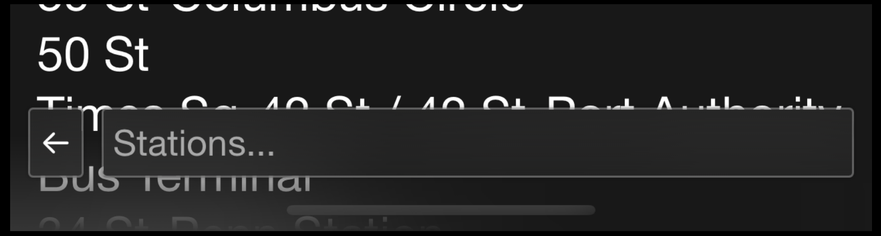


/?stop_ids=M06



/?stop_ids

/stations?prev=M06

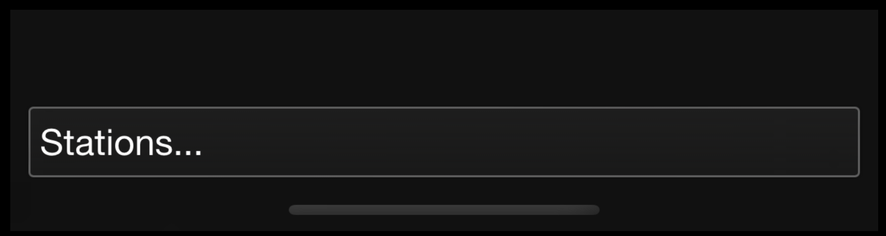


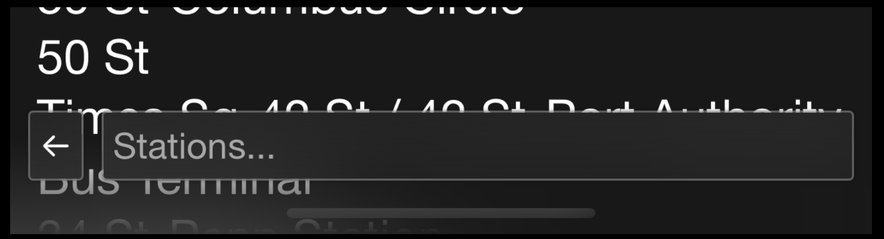
/stations?prev=M06

/stations?prev=M06

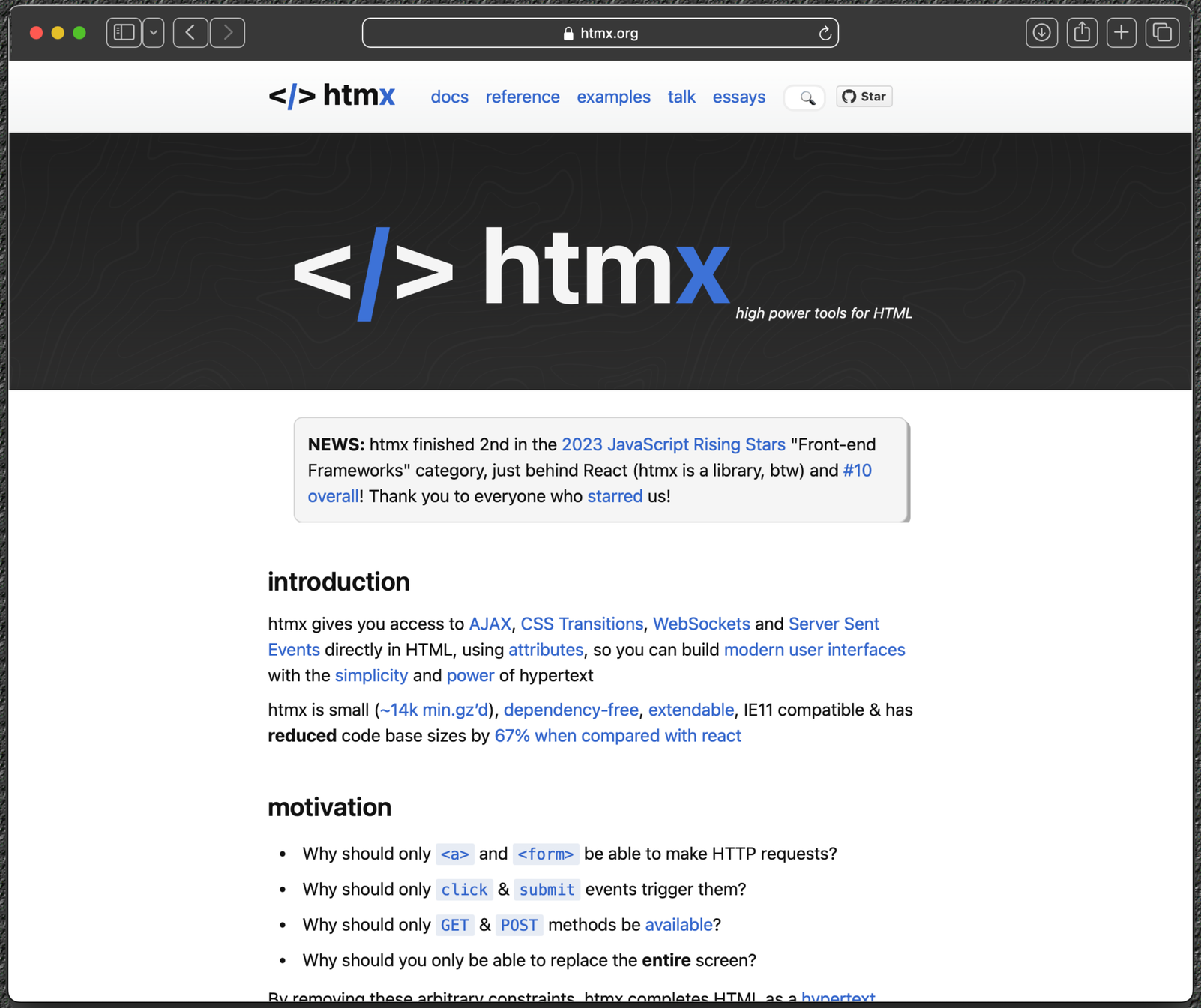
/?stop_ids=M06

/?stop_ids=M06





/stations?prev=M06


htmx.org
</> htmx
docs
reference
examples
talk
essays
Star
</> htmx
high power tools for HTML
NEWS: htmx finished 2nd in the 2023 JavaScript Rising Stars "Front-end Frameworks" category, just behind React (htmx is a library, btw) and #10 overall! Thank you to everyone who starred us!
introduction
htmx gives you access to AJAX, CSS Transitions, WebSockets and Server Sent Events directly in HTML, using attributes, so you can build modern user interfaces with the simplicity and power of hypertext
htmx is small (~14k min.gz'd), dependency-free, extendable, IE11 compatible & has reduced code base sizes by 67% when compared with react
motivation
Why should only <a> and <form> be able to make HTTP requests?
Why should only click & submit events trigger them?
Why should only GET & POST methods be available?
Why should you only be able to replace the entire screen?
By removing these arbitrary constraints, htmx completes HTML as a hypertext

Client-side refreshing

```
</header>
<main hx-get="{{url_for('arrivals', mode=mode, stop_ids=stop_ids)}}" hx-trigger="every 15s"
  hx-swap="morph:outerHTML" hx-select="main > *">
```

Live time updates

```
import dayjs from "./dayjs.min.mjs";
import relativeTime from "./dayjs.relativeTime.min.mjs";
import utc from "./dayjs.utc.min.mjs";
dayjs.extend(relativeTime);
dayjs.extend(utc);
setInterval(function() {
  document.querySelectorAll("time[data-arrival-time]").forEach(time => {
    const offset = dayjs(time.getAttribute("datetime")).diff(dayjs.utc(), 'm');

    if (offset <= 0) {
      time.innerText = "due";
    } else {
      time.querySelector(".arrival-time-mins").innerText = offset;
      if (offset < 5) {
        time.classList.add("arrival-time-due");
      }
    }
  });
  document.querySelectorAll("time[data-replace]").forEach(time => {
    const offsetSecs = dayjs(time.getAttribute("datetime")).diff(dayjs.utc(), 's')
    if (offsetSecs > -15) {
      time.innerText = 'just now';
    } else if (offsetSecs > -60) {
      time.innerText = `${Math.abs(offsetSecs)} seconds ago`;
    } else {
      time.innerText = dayjs(time.getAttribute("datetime")).fromNow();
    }
  })
}, 1000);
```

Making a webapp like it's 2024

Actually, we have a lot of nice things in the web now:

`await`

Async I/O

ES6

and

CSS3

SVGs

PWAs

Async I/O (in Python!)

```
Feeds = ("1", "A", "F", "G", "J", "L", "N", "SI")
NYCTFeeds = [NYCTFeed(feed, api_key=ApiKey, fetch_immediately=True) for feed in Feeds]
```

```
print(arrow.get(time), "refreshing")
await asyncio.gather(*(feed.refresh_async() for feed in NYCTFeeds))
```

ES6 and CSS3

```
#ota-search:has(input:focus) {
  clamp(0.5ch, 2vw, 2ch);
```

Discussion

- **Progressive enhancement is hard**
  - HTMX is a great library but as a frontend library it can only go so far.
  - There's a lot of state containers for JS, but comparatively few *shared* state containers
- **Web tech has gone really far**
  - It's never been easier to author a webapp with vanilla javascript

Discussion

- **Mobile web is hard**

Discussion

- **Mobile web is hard.....** *for iOS Safari*

```
* d9175f6 stop trying to autofocus it just won't happen
* eaf7bf9 i wonder if this would work
* 502e6fb hmm
* 9d572c1 Revert "oh no"
* aa22010 i think i have to give up lol 3
* 72f0470 i think i have to give up lol 2
* 69b72d2 i think i have to give up lol
* 73d987c hopefully
* 5ed8f02 ummm3
* f07db98 ummm2
* 4a78fbd ummm
* d2c6f7f oh no
```

```
* 011be96 big style change 2
* 72707e1 big style change
* 8f99c29 ugh9
* 8bb0602 ugh8
* 01417ba ugh7
* 3e5e827 ugh6
* 25f4c75 ugh5
* 6d9eca5 ugh4
* c781724 ugh3
* 977a56a i give up
* 9f7c25e ugh2
* be6c61d ugh
```

Future work/more yak-shaving opportunities

- Offline support with service workers

Would love to pair on this btw!

- Make the train view page look better
- …

Future work/more yak-shaving opportunities

- Offline support with service workers

Would love to pair on this btw!

- Make the train view page look better
- …
- Break the subway challenge world record (472 stations in 22 hours and 14 minutes)

Thank you so much

For your pairing and/or enthusiasm :)

Dan, Evan, Gus, Ivy, Julie, Matthew (W'14), Olivia, Sareena, Sydney, Taniya

Prior art

wheresthefuckingtrain.com

Transit Bandage app

Backgrounds

sadgrl.online