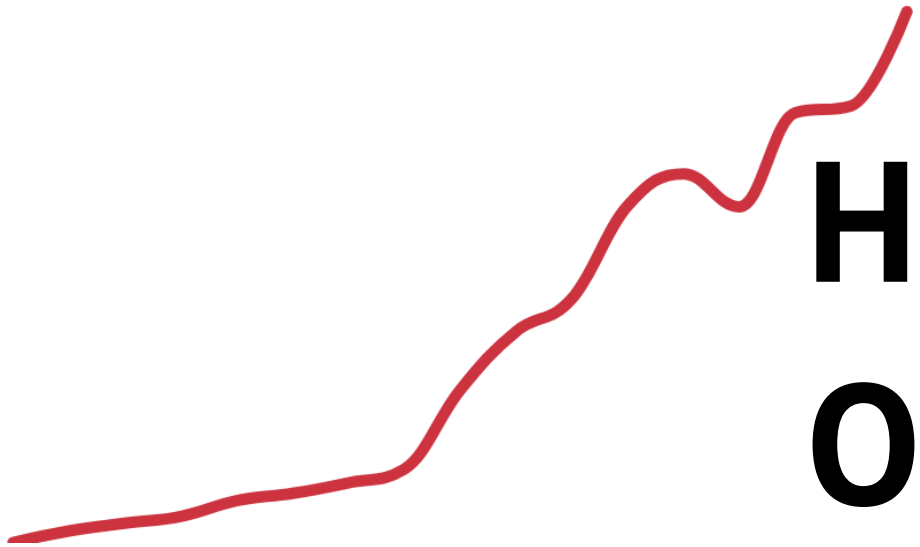
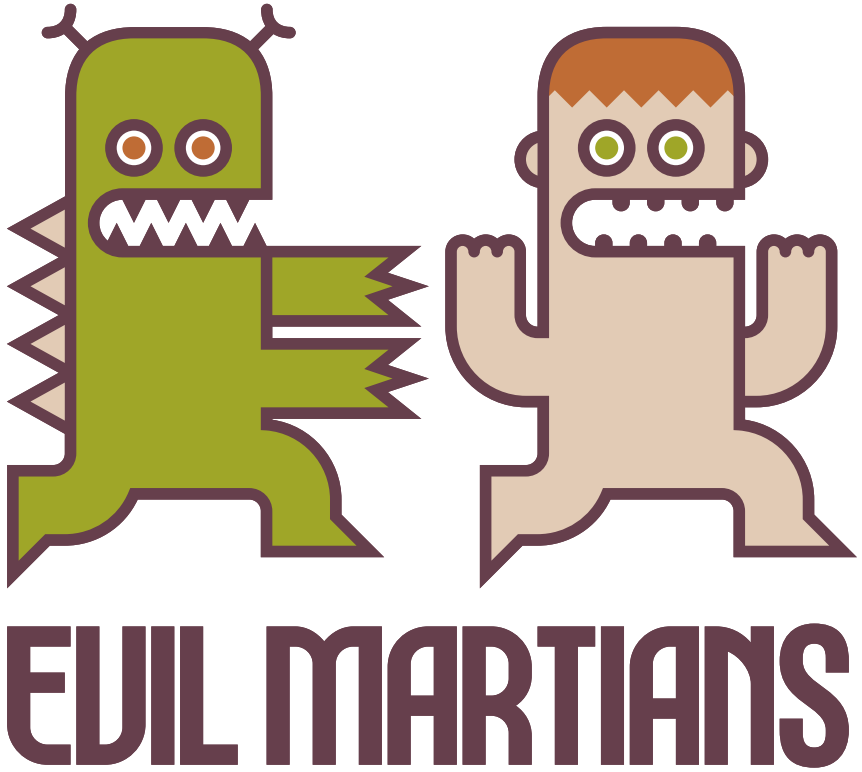


How to Make Popular Open Source Project

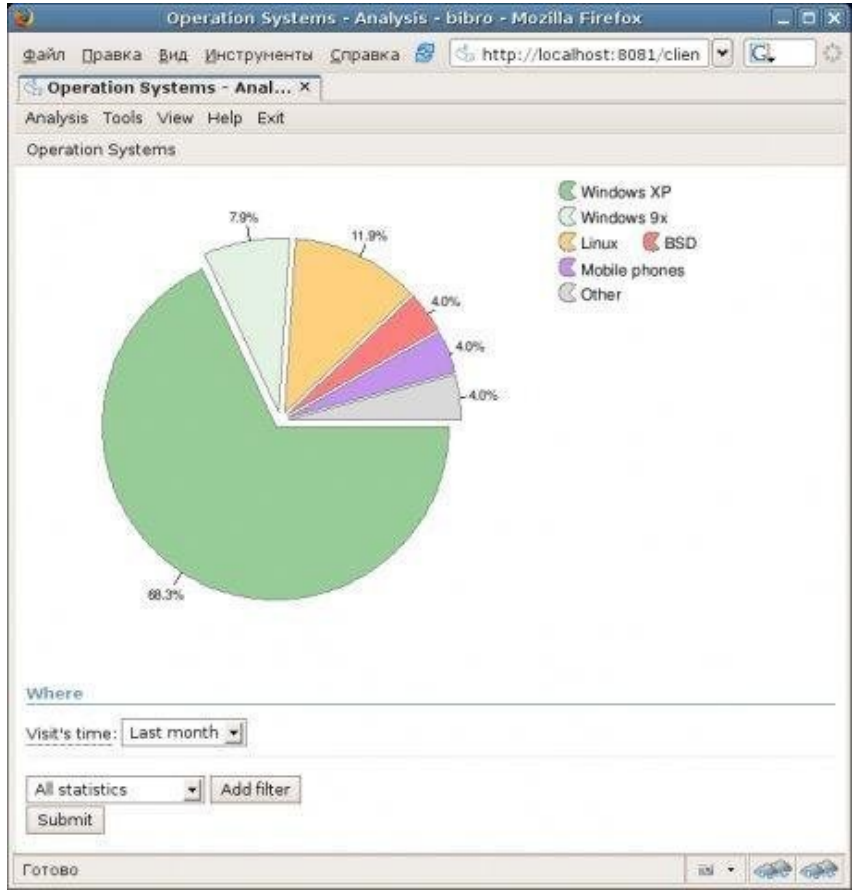
Andrey Sitnik, Evil Martians

I am working at



evilmartians.com

15 Years Experience in Open Source

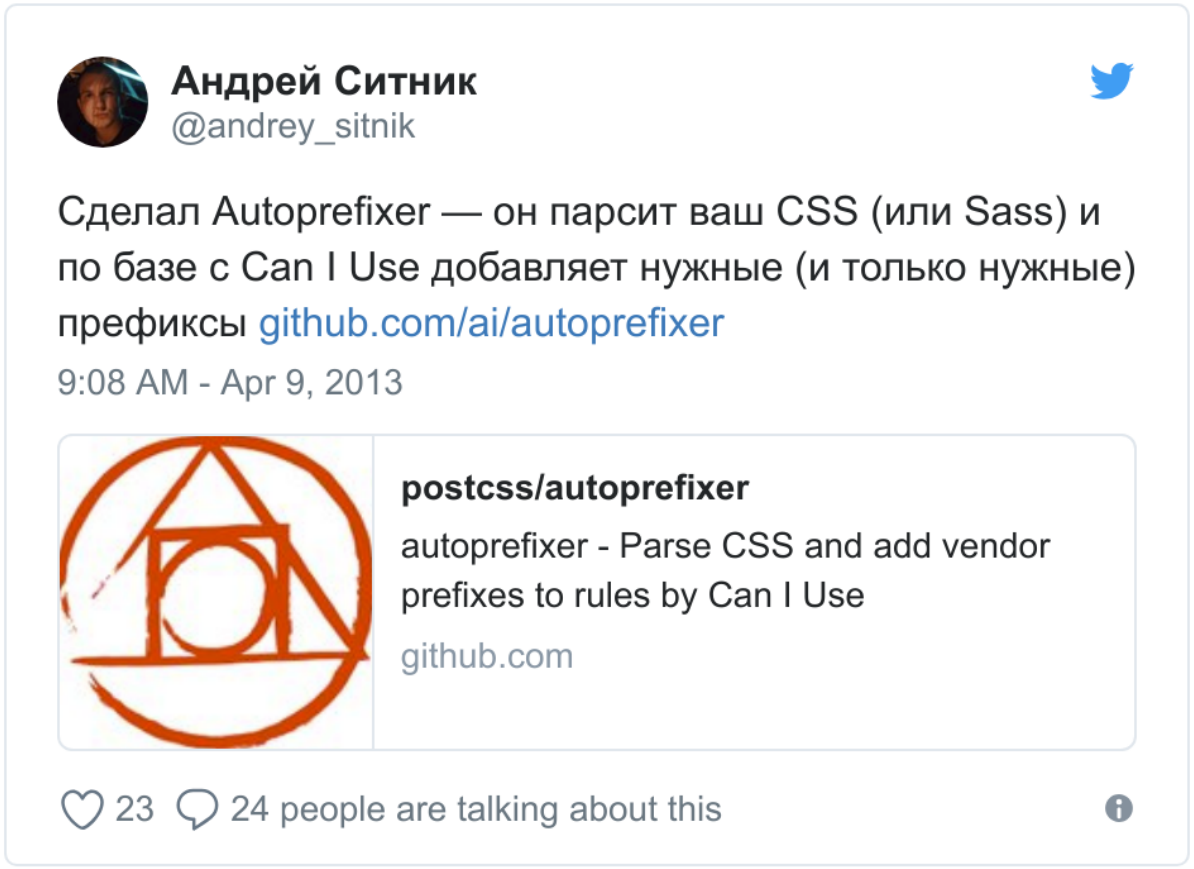


source forge

since 2004

sourceforge.net/projects/bibro/

5 Years Experience in Open Source Promoting



since 2013

twitter.com/andrey_sitnik/status/321655851565850624

Sharing Real Experience

30 M

Autoprefixer

80 M

PostCSS

50 M

Browserslist

3 M

Nano ID

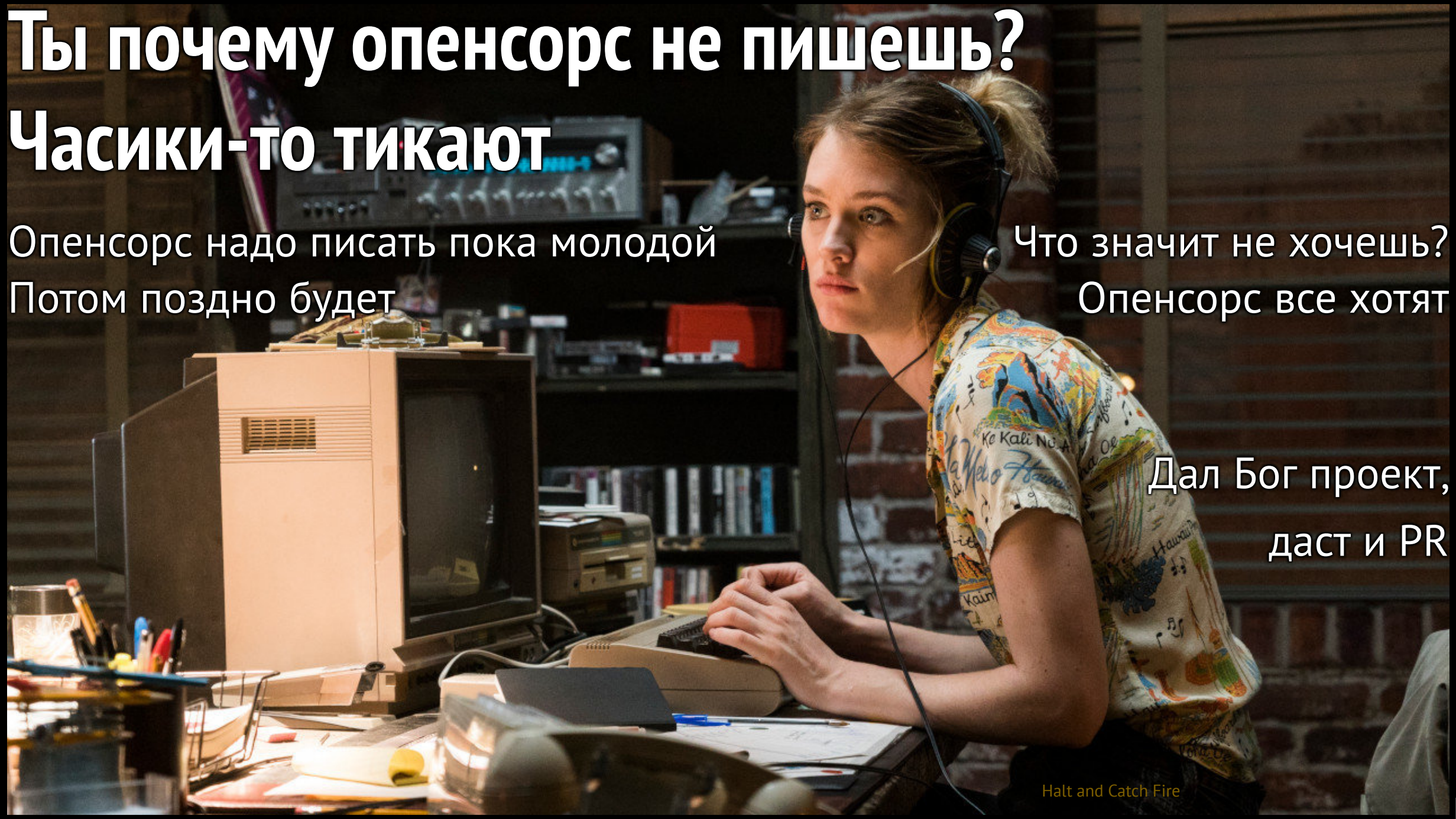
Downloads per month

Chapter 1

Why you should create open source project?

Ты почему опенсорс не пишешь?

Часики-то тикают



Опенсорс надо писать пока молодой

Потом поздно будет

Что значит не хочешь?

Опенсорс все хотят

Дал Бог проект,

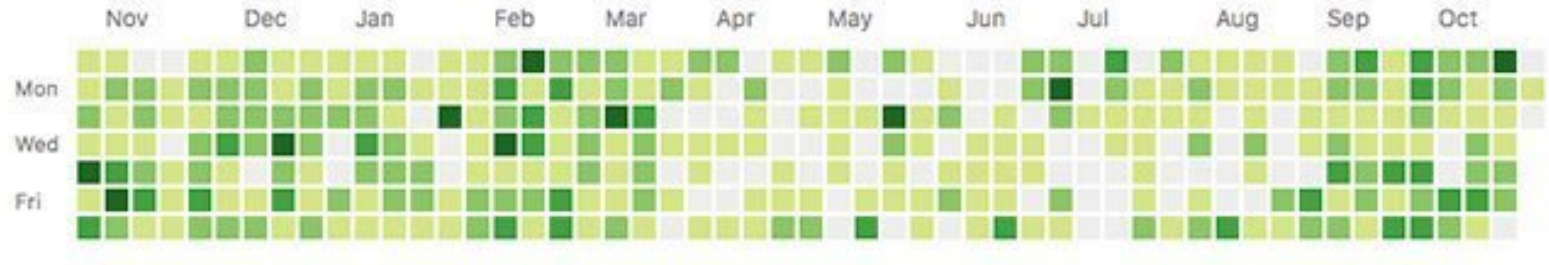
даст и PR

Halt and Catch Fire

“You must have OS project to find a job”

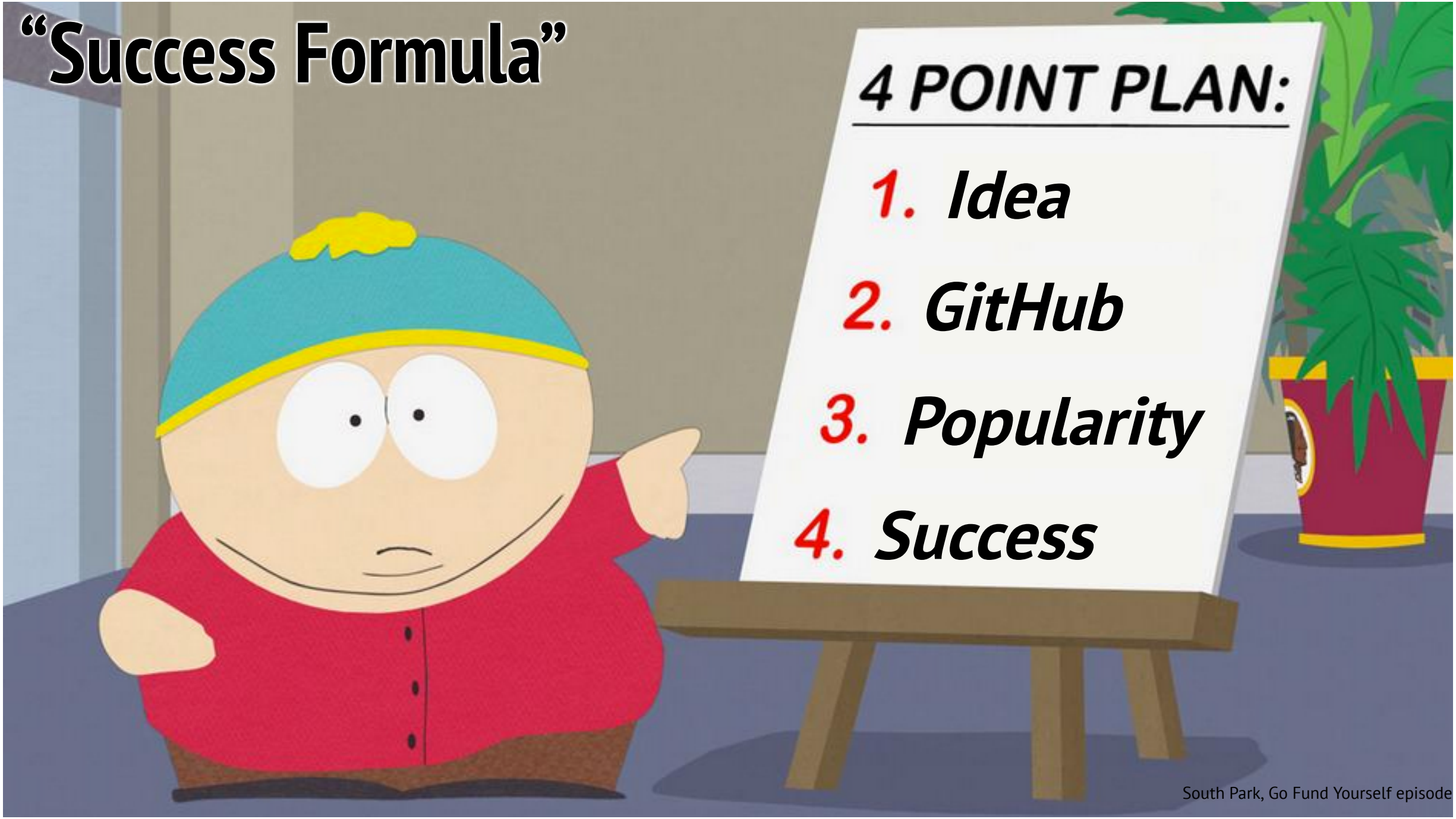
Me

The guy **HR** tells me not to worry about



twitter.com/Kiryous/status/790910619574730752

“Success Formula”



South Park, Go Fund Yourself episode

Open Source dream is a lie

Chompoo Baritone

Popular project ≠ many followers

rrodmit

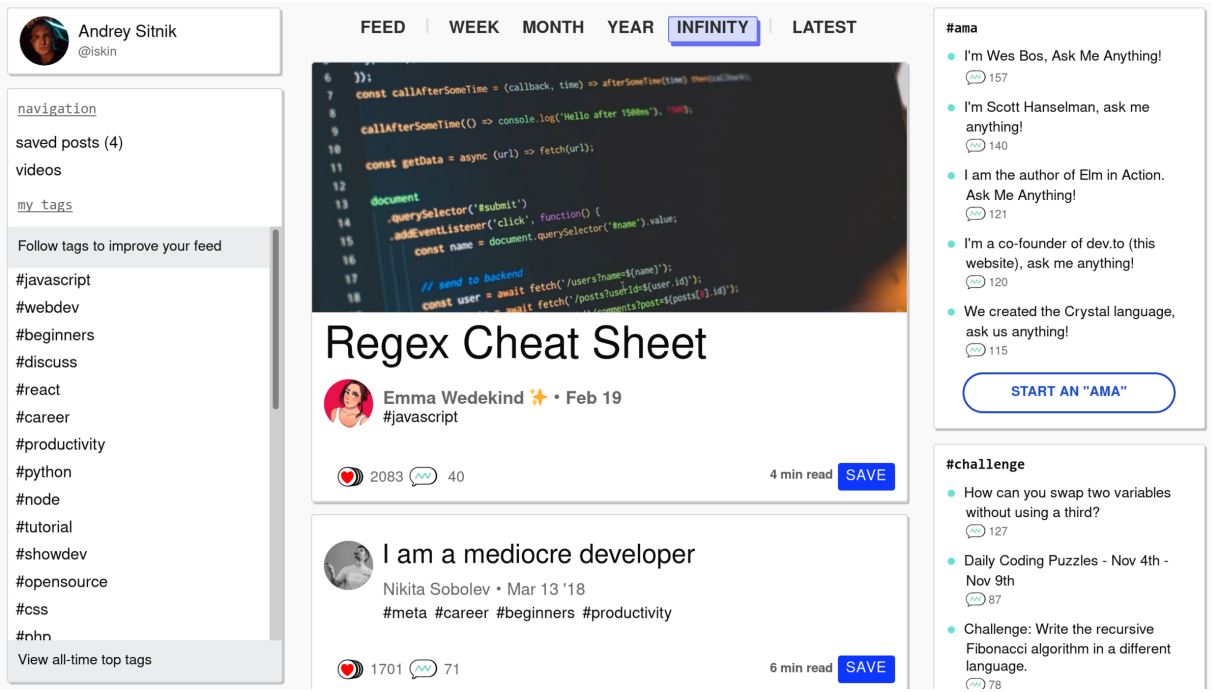
@timdorr

Maintainer of Redux and React Router.

4 075 followers

twitter.com/timdorr

Better ways to be popular

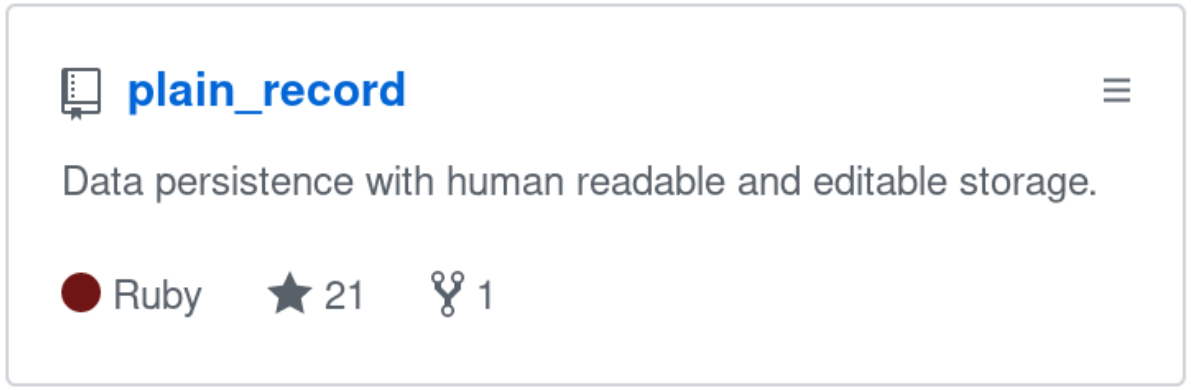


Articles

Talks

dev.to and Frontend Developer Love 2018

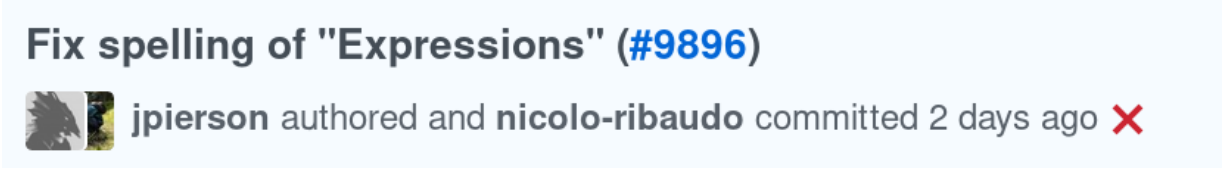
GitHub project ≠ good resume



1 month of work → few stars

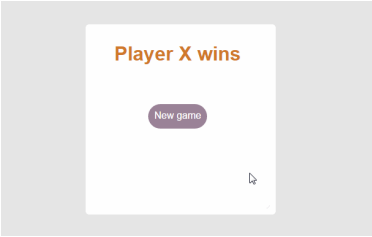
vs

2 weeks → commit to Babel



Better ways to be popular

- Create app



- Fix docs in popular projects
- Use beta version ⟶ fix bugs

App from slides.com/keziyahlewis/deck

The best reason: change the world

My reason for Autoprefixer: kill Compass

2 README.markdown

View

```
...  ...  @@ -1,5 +1,7 @@
1    1     # Compass Stylesheet Authoring Framework
2    2
     3    +**Depreciated:** Compass is no longer supported.
     4    +
3    5     Build Status: [![Build Status](https://travis-ci.org/Compass/compass.png)]
           (https://travis-ci.org/Compass/compass)
4    6
5    7     Code Quality: [![Code Climate]
           (https://codeclimate.com/github/Compass/compass.png)]
           (https://codeclimate.com/github/Compass/compass)
```

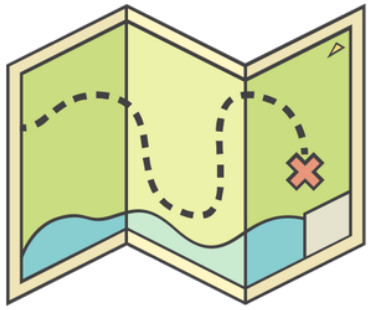
Source: github.com/Compass/compass/commit/dd74a1cfef478a896d03152b2c2a3b93839d168e

My reason for PostCSS: CSS tools diversity

Stylelint

cssnano

RTLCSS



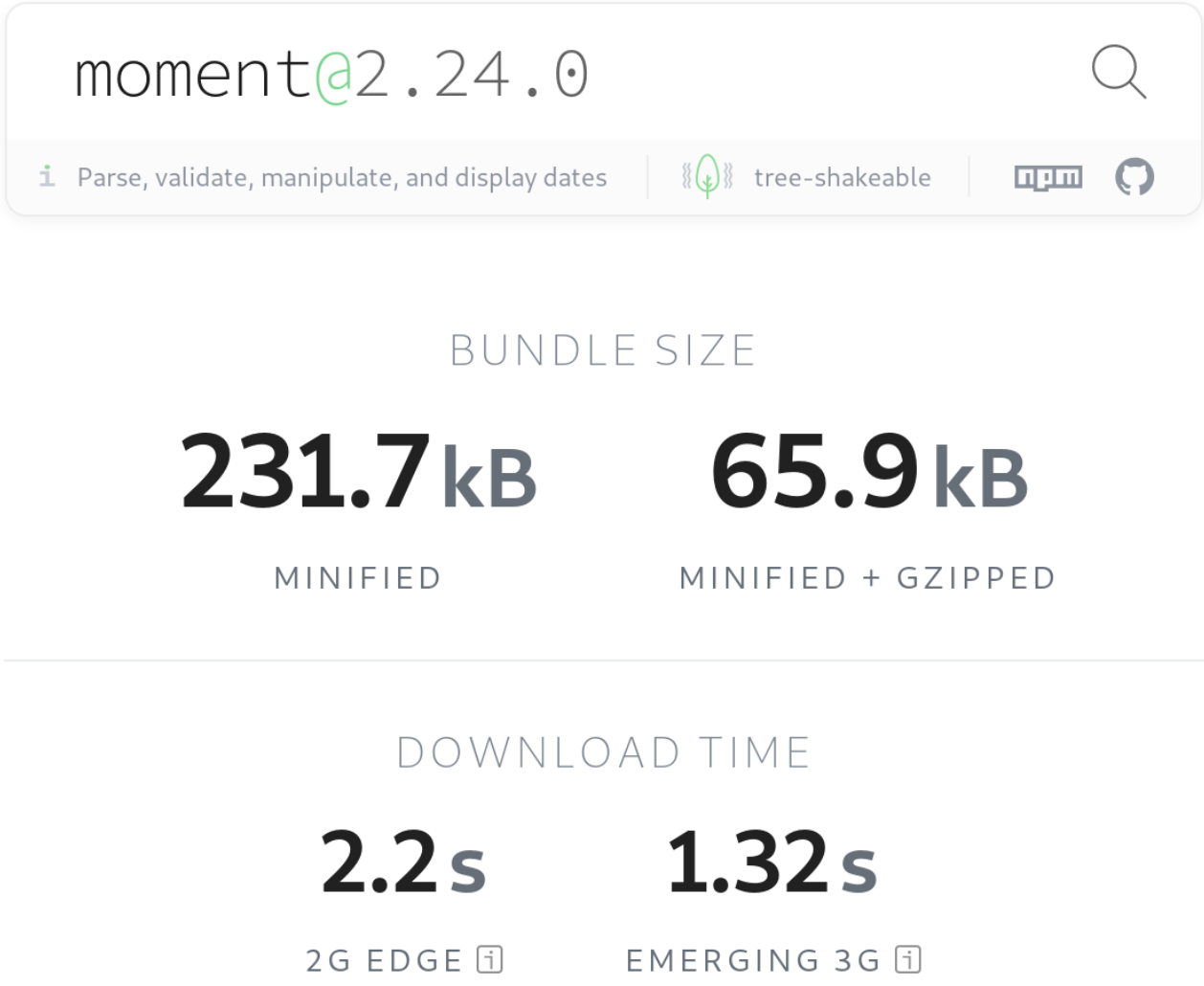
Lost Grid

@util

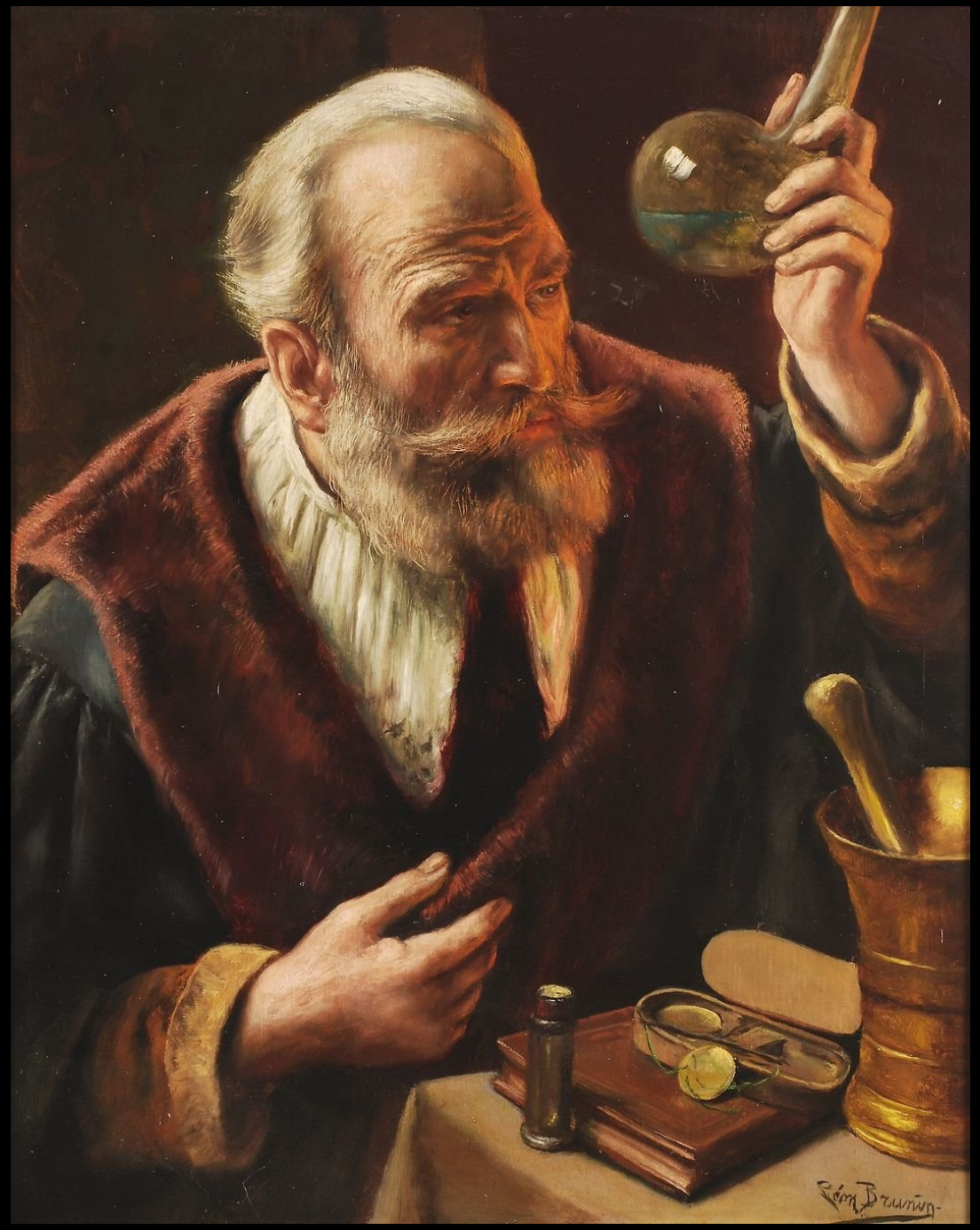
PostCSS utility library

CSS
MODULES

My reason for Size Limit: reduce JS lib size



bundlephobia.com/result?p=moment@2.24.0



Chapter 2

Success formula

False formula

Good idea → Popular project

My open source projects

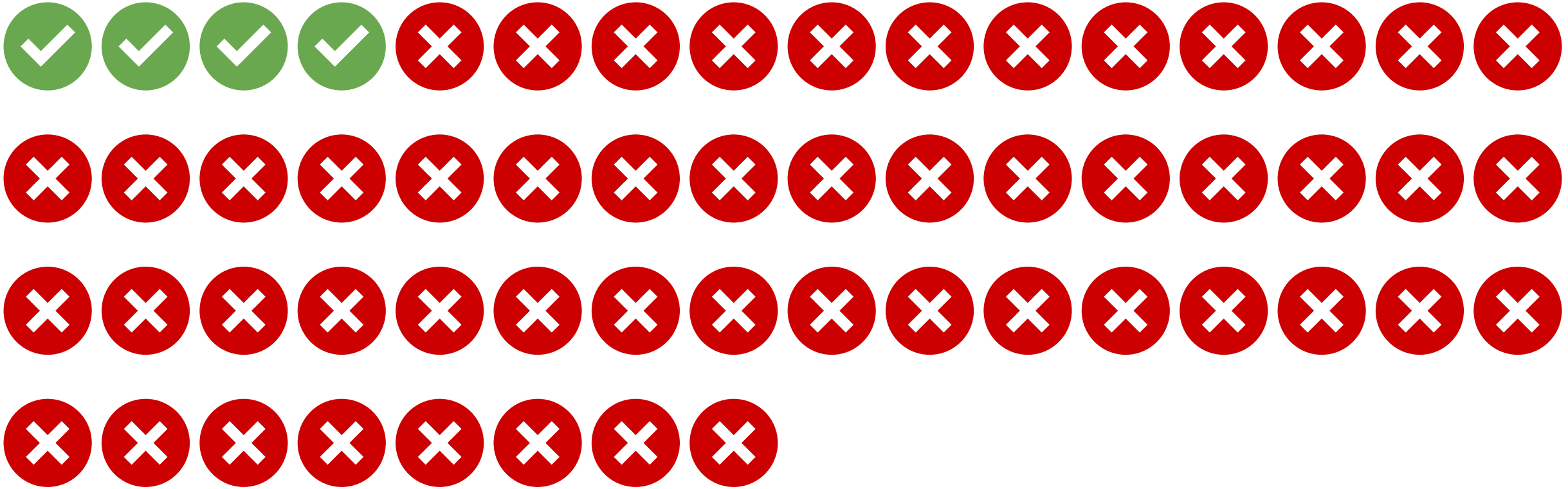
Autoprefixer

PostCSS

Browserslist

Nano ID

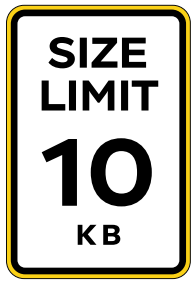
The real list of my projects



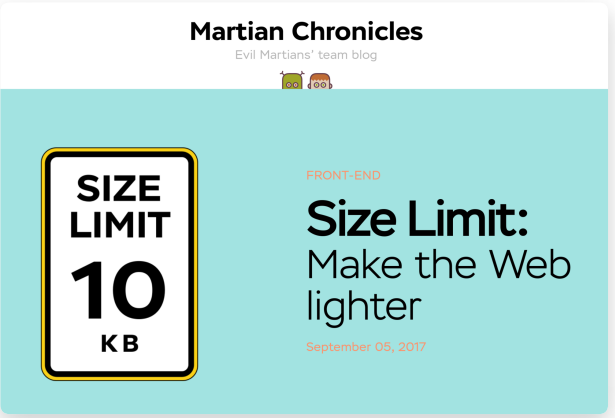
52 non-popular and only **4** popular

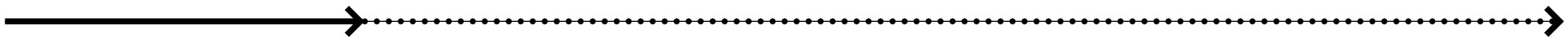
github.com/ai and sourceforge.net/u/iskin/profile/

Fail case 1: Size Limit



a week to write the article

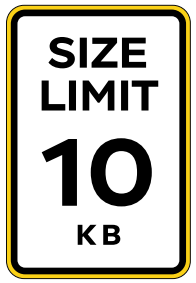


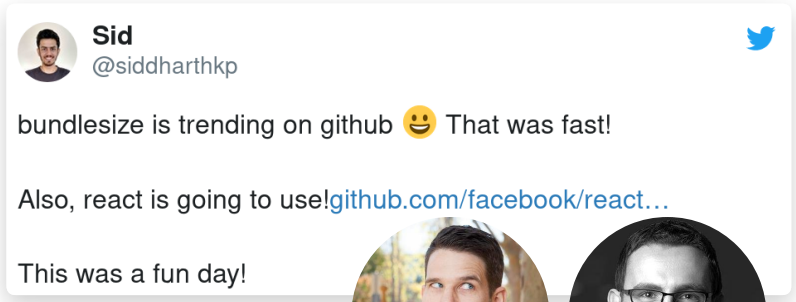


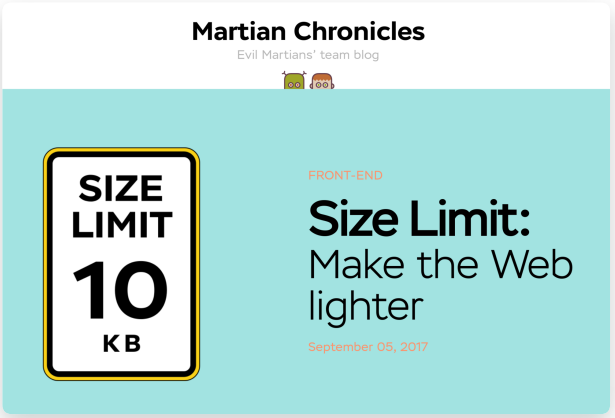
Size Limit 0.1

Article about it

bundlesize was released before my article



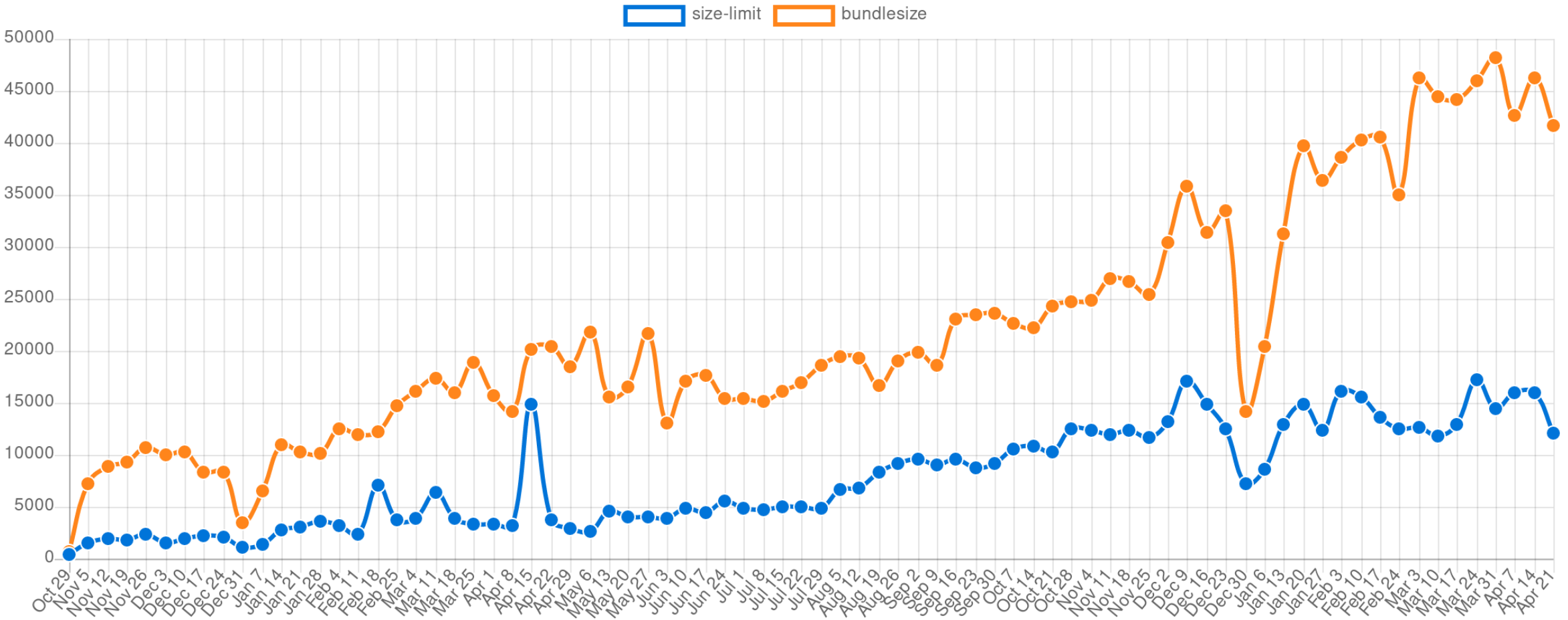




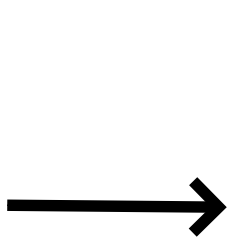
Size Limit 0.1

Article about it

Size Limit lost



Fail case 2: Autoprefixer



grid-template-rows: auto auto

Fail case 2: Autoprefixer

grid-template-rows: auto auto

The real success formula

Project popularity =

Your popularity **+**

Promotion **+**

Benefits for users **+**

Luck

Idea is not important

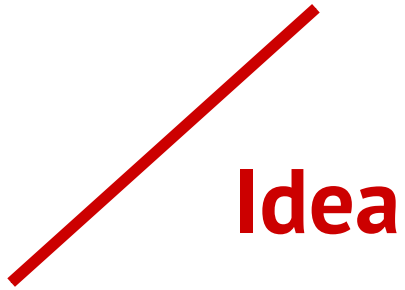
Project popularity =

Your popularity **+**

Promotion **+**

Benefits for users **+**

Luck



Useless project can’t be promoted

Project popularity =

Your popularity **+**

Promotion **+**

Benefits for users **+**

Luck

Chapter 3

Base axioms

Axiom 1: success formula

Project popularity =

Your popularity **+**

Promotion **+**

Benefits for users **+**

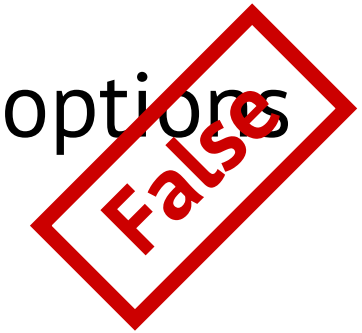
Luck

How we think people choose frameworks

Rational choose across **all** options

How we think people choose frameworks

Rational choose across **all** options



Axiom 2: How people choose frameworks

Irrational choose across **popular** options

How we think people read docs/tweets

thebuzzmagazines.com/articles/2017/12/cozy-december-reads

How we think people read docs/tweets

thebuzzmagazines.com/articles/2017/12/cozy-december-reads

Axiom 3: How people read docs/tweets

thenib.com/this-is-not-fine

Chapter 4

Preparations

Success formula has Popularity on both sides

Project popularity =

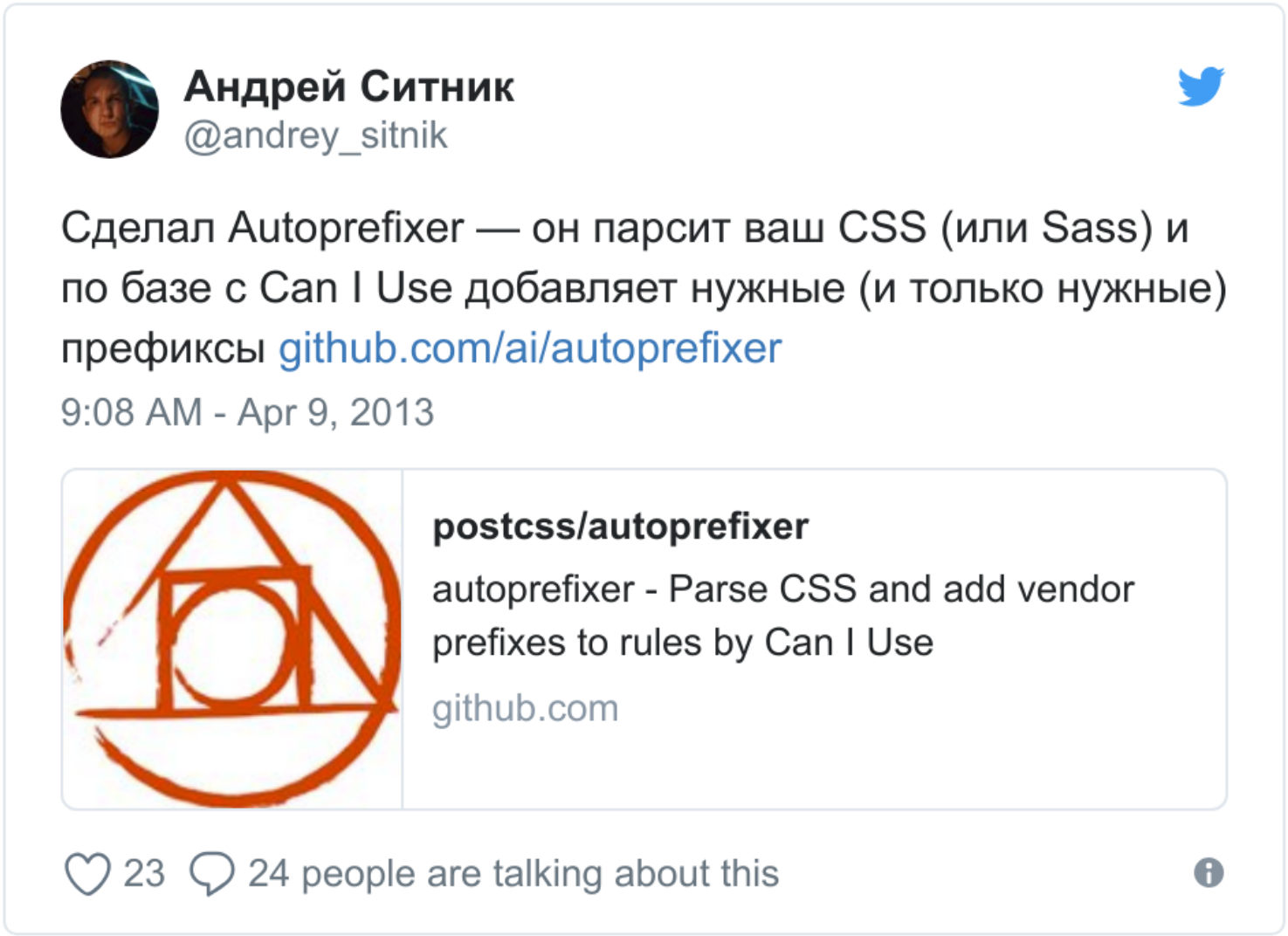
Your popularity +

Promotion +

Benefits for users +

Luck

My main mistake: only Russian announce



My main mistake: nobody to mention

Sara Soueidan
@SaraSoueidan

Follow

I'm so thrilled! Just installed Sublime Text's Autoprefixer plugin from @sindresorhus (github.com/sindresorhus/s …) It's like a touch of magic!

8:09 AM - 13 May 2013

Sindre Sorhus @sindresorhus · 4 Jun 2013

Announcing: sublime-autoprefixer - Sublime plugin to automagically add prefixes to your CSS so you don't have to

sindresorhus/sublime-autoprefixer

Sublime plugin to prefix your CSS. Contribute to sindresorhus/sublime-autoprefixer development by creating an account on GitHub.

github.com

4 45 39

Addy Osmani @addyosmani · 4 Jun 2013

@sindresorhus nice work, Sindre! Hopefully we'll be able to land github.com/yeoman/generat… soonish.

1 1

Sindre Sorhus
@sindresorhus

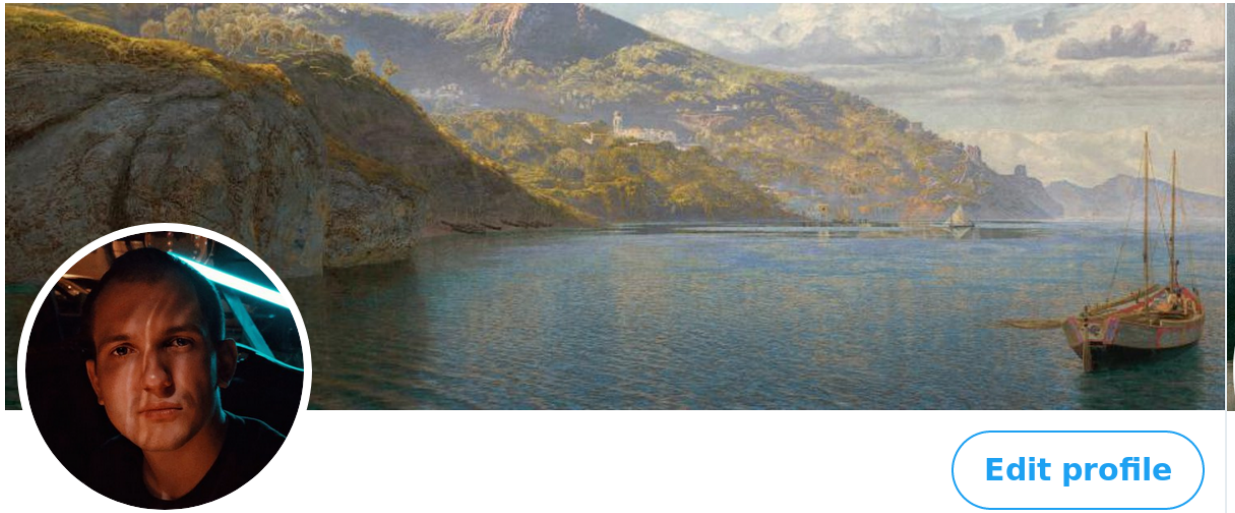
Following

Replying to @addyosmani

@addyosmani thanks!, though the biggest credit should go to the Autoprefixer lib.

11:25 AM - 4 Jun 2013

Advice 1: Create English Twitter account



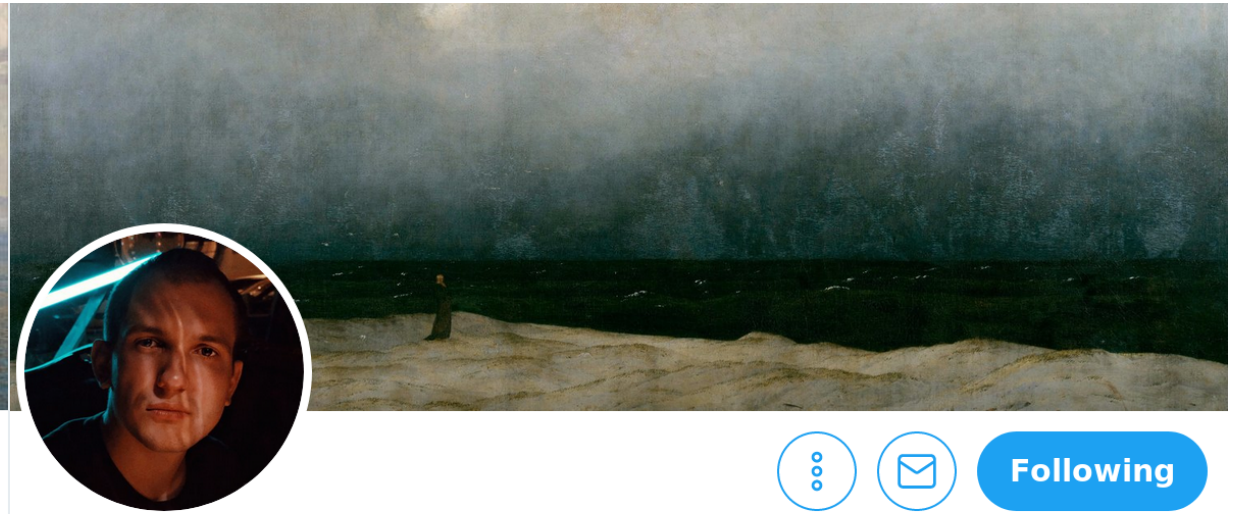
Edit profile

Андрей Ситник
@andrey_sitnik

Ведущий фронтендер в Злых марсианах. Создатель @PostCSS, @Autoprefixer и @Logux_io. In English: @andreysitnik

New York, USA sitnik.ru Born June 4, 1987
Joined August 2009

727 Following **7,254** Followers



Following

Andrey Sitnik
@andreysitnik Follows you

Author of @PostCSS and @Autoprefixer. Lead front-end developer in @EvilMartians. Account about code only: @sitnikcode. По-русски: @andrey_sitnik

New York, NY sitnik.ru Born June 4 Joined March 2014

379 Following **3,961** Followers

Success formula is based on luck

Project popularity **=**

Your popularity **+**

Promotion **+**

Benefits for users **+**

Luck

Advice 2: Be ready to try many times

Project popularity =
Your popularity +
Promotion +
Benefits for users +
Luck

Project popularity =
Your popularity +
Promotion +
Benefits for users +
Luck

Project popularity =
Your popularity +
Promotion +
Benefits for users +
Luck

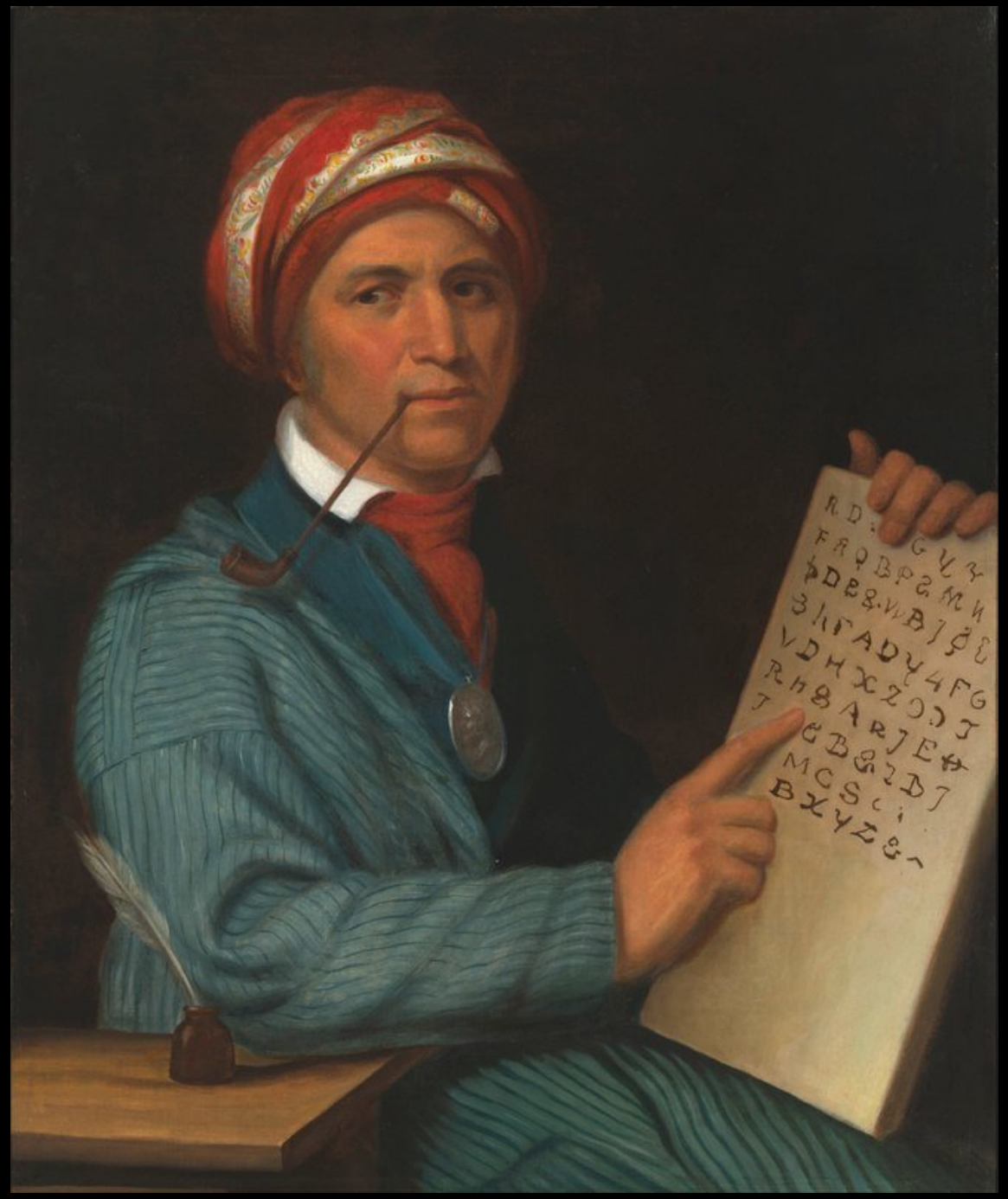
Project popularity =
Your popularity +
Promotion +
Benefits for users +
Luck

Project popularity =
Your popularity +
Promotion +
Benefits for users +
Luck

Project popularity =
Your popularity +
Promotion +
Benefits for users +
Luck

Project popularity =
Your popularity +
Promotion +
Benefits for users +
Luck

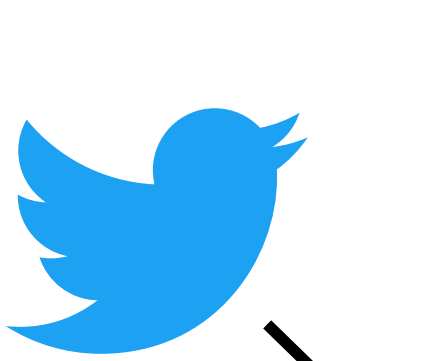
Project popularity =
Your popularity
Promotion
Benefits for users
Luck



Chapter 5

README.md

README.md is your main entry point

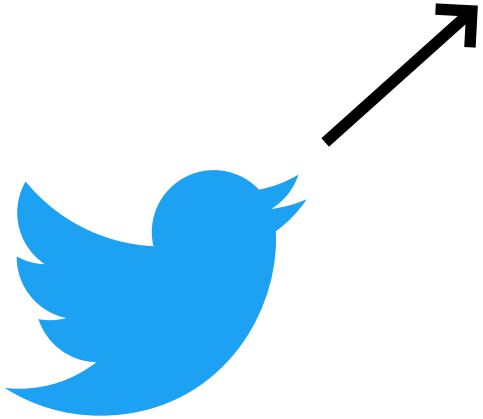


README.md

Size Limit Cult of Martians

Size Limit is a linter for your JS application or library performance. It calculates the real cost of your JS for end users and throws an error if the cost exceeds the limit.

- Size Limit calculates **the time** it would take a browser **to download and execute** your JS. Time is a much more accurate and understandable metric compared to the size in bytes.
- Size Limit calculations include **all dependencies and polyfills** used in your JS.
- Add Size Limit to **Travis CI**, **Circle CI**, or another CI system to know if a pull request adds a massive dependency.



```
ai@blackjack:~/Dev/my-js-lib

~/Dev/my-js-lib
➜ npx size-limit

  Total time limit has exceeded
  Time limit:   800 ms
  Package size: 36.46 KB with all dependencies, minified and gzipped
  Loading time: 730 ms   on slow 3G
  Running time: 200 ms   on Snapdragon 410
  Total time:   929 ms

  Try to reduce size or increase limit in .size-limit.json

~/Dev/my-js-lib
➜
```

How people read docs

What users need?

Project popularity =

Your popularity **+**

Promotion **+**

Benefits for users **+**

Luck

Advice 3: Start from the reason to read it

artstation.com/dianafranco

First block of PostCSS

Clear description

PostCSS is a tool for transforming styles with JS plugins.

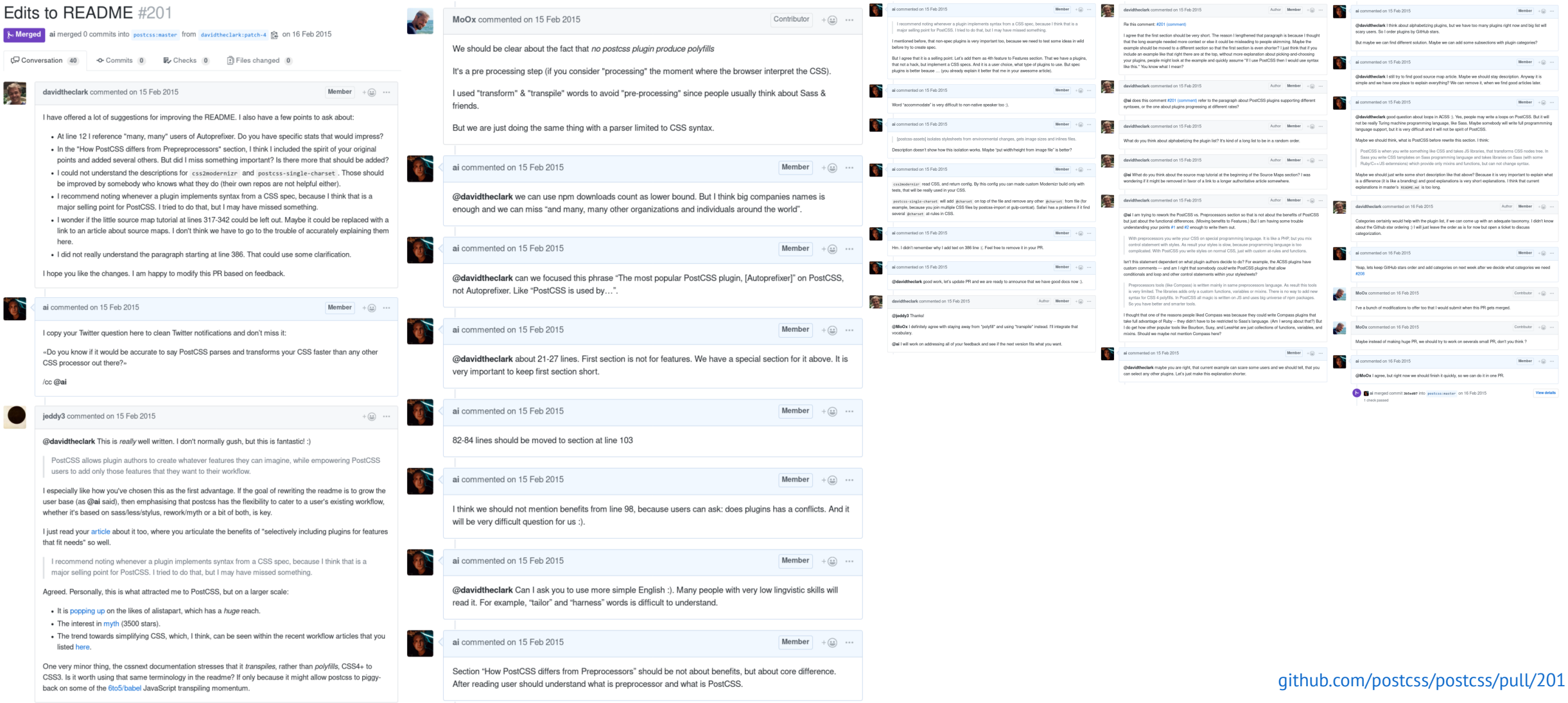
These plugins can lint your CSS, support variables and mixins, transpile future CSS syntax, inline images, and more.

How it could be useful

PostCSS is used by industry leaders including Wikipedia, Twitter, Alibaba, and JetBrains. The Autoprefixer PostCSS plugin is one of the most popular CSS processors.

Why it is important

It's OK to spend a week on the first block



Edits to README #201

Merged ai merged 0 commits into postcss:master from davidtheclark:patch-4 on 16 Feb 2015

Conversation 40 Commits 0 Checks 0 Files changed 0

davidtheclark commented on 15 Feb 2015 — Member

I have offered a lot of suggestions for improving the README. I also have a few points to ask about:

- At line 12 I reference "many, many" users of Autoprefixer. Do you have specific stats that would impress?
- In the "How PostCSS differs from Prepreprocessors" section, I think I included the spirit of your original points and added several others. But did I miss something important? Is there more that should be added?
- I could not understand the descriptions for `css2modernizr` and `postcss-single-charset`. Those should be improved by somebody who knows what they do (their own repos are not helpful either).
- I recommend noting whenever a plugin implements syntax from a CSS spec, because I think that is a major selling point for PostCSS. I tried to do that, but I may have missed something.
- I wonder if the little source map tutorial at lines 317-342 could be left out. Maybe it could be replaced with a link to an article about source maps. I don't think we have to go to the trouble of accurately explaining them here.
- I did not really understand the paragraph starting at line 386. That could use some clarification.

I hope you like the changes. I am happy to modify this PR based on feedback.

ai commented on 15 Feb 2015 — Member

I copy your Twitter question here to clean Twitter notifications and don't miss it:

«Do you know if it would be accurate to say PostCSS parses and transforms your CSS faster than any other CSS processor out there?»

/cc **@ai**

jeddy3 commented on 15 Feb 2015

@davidtheclark This is *really* well written. I don't normally gush, but this is fantastic! :)

> PostCSS allows plugin authors to create whatever features they can imagine, while empowering PostCSS users to add only those features that they want to their workflow.

I especially like how you've chosen this as the first advantage. If the goal of rewriting the readme is to grow the user base (as **@ai** said), then emphasising that postcss has the flexibility to cater to a user's existing workflow, whether it's based on sass/less/stylus, rework/myth or a bit of both, is key.

I just read your article about it too, where you articulate the benefits of "selectively including plugins for features that fit needs" so well.

> I recommend noting whenever a plugin implements syntax from a CSS spec, because I think that is a major selling point for PostCSS. I tried to do that, but I may have missed something.

Agreed. Personally, this is what attracted me to PostCSS, but on a larger scale:

- It is popping up on the likes of alistapart, which has a *huge* reach.
- The interest in myth (3500 stars).
- The trend towards simplifying CSS, which, I think, can be seen within the recent workflow articles that you listed here.

One very minor thing, the cssnext documentation stresses that it *transpiles*, rather than *polyfills*, CSS4+ to CSS3. Is it worth using that same terminology in the readme? If only because it might allow postcss to piggy-back on some of the 6to5/babel JavaScript transpiling momentum.

MoOx commented on 15 Feb 2015 — Contributor

We should be clear about the fact that *no postcss plugin produce polyfills*

It's a pre processing step (if you consider "processing" the moment where the browser interpret the CSS).

I used "transform" & "transpile" words to avoid "pre-processing" since people usually think about Sass & friends.

But we are just doing the same thing with a parser limited to CSS syntax.

ai commented on 15 Feb 2015 — Member

@davidtheclark we can use npm downloads count as lower bound. But I think big companies names is enough and we can miss "and many, many other organizations and individuals around the world".

ai commented on 15 Feb 2015 — Member

@davidtheclark can we focused this phrase "The most popular PostCSS plugin, [Autoprefixer]" on PostCSS, not Autoprefixer. Like "PostCSS is used by…".

ai commented on 15 Feb 2015 — Member

@davidtheclark about 21-27 lines. First section is not for features. We have a special section for it above. It is very important to keep first section short.

ai commented on 15 Feb 2015 — Member

82-84 lines should be moved to section at line 103

ai commented on 15 Feb 2015 — Member

I think we should not mention benefits from line 98, because users can ask: does plugins has a conflicts. And it will be very difficult question for us :).

ai commented on 15 Feb 2015 — Member

@davidtheclark Can I ask you to use more simple English :). Many people with very low lingvistic skills will read it. For example, "tailor" and "harness" words is difficult to understand.

ai commented on 15 Feb 2015 — Member

Section "How PostCSS differs from Preprocessors" should be not about benefits, but about core difference. After reading user should understand what is preprocessor and what is PostCSS.

github.com/postcss/postcss/pull/201

First block of Autoprefixer

Clear description

PostCSS plugin to parse CSS and add vendor prefixes to CSS rules using values from Can I Use.

It is recommended by Google and used in Twitter and Alibaba.

Why it is important

First block of Nano ID

Clear description

A tiny, secure, URL-friendly, unique string ID generator for JS.

- **Small.** 141 bytes (minified and gzipped). No dependencies.
- **Safe.** It uses cryptographically strong random APIs and tests distribution of symbols.
- **Fast.** It's 16% faster than UUID.
- **Compact.** It uses a larger alphabet than UUID (A-Za-z0-9_-).

Difference from analogs

What should be in first block

1. Clear description
2. How the product is useful for users
3. The difference from other products

Bad example of first sentence

Svelte is cybernetically enhanced web apps  Not clear

Svelte is JS framework with unique compiler which generate smaller JS files 

How to make good description


And then cut it 2-4 times

What do you think people read

Nano ID

A tiny, secure, URL-friendly, unique string ID generator for JavaScript.

- **Small.** 141 bytes (minified and gzipped). No dependencies. Size Limit controls the size.
- **Safe.** It uses cryptographically strong random APIs and tests distribution of symbols.
- **Fast.** It's 16% faster than UUID.
- **Compact.** It uses a larger alphabet than UUID (`A-Za-z0-9_-`). So ID size was reduced from 36 to 21 symbols.

The generator supports Node.js, React Native, and all browsers.

Security

See a good article about random generators theory: Secure random values (in Node.js)

Unpredictability

Instead of using the unsafe `Math.random()`, Nano ID uses the `crypto` module in Node.js and the Web Crypto API in browsers. These modules use unpredictable hardware random generator.

Uniformity

`random % alphabet` is a popular mistake to make when coding an ID generator. The spread will not be even; there will be a lower chance for some symbols to appear compared to others—so it will reduce the number of tries when brute-forcing.

Nano ID uses a better algorithm and is tested for uniformity.

What people actually read

Nano ID

A tiny, secure, URL-friendly, unique string ID generator for JavaScript.

- **Small.** 141 bytes (minified and gzipped). No dependencies. Size Limit controls the size.
- **Safe.** It uses cryptographically strong random APIs and tests distribution of symbols.
- **Fast.** It's 16% faster than UUID.
- **Compact.** It uses a larger alphabet than UUID (`A-Za-z0-9_-`). So ID size was reduced from 36 to 21 symbols.

The generator supports Node.js, React Native, and all browsers.

Security

See a good article about random generators theory: Secure random values (in Node.js)

Unpredictability

Instead of using the unsafe `Math.random()`, Nano ID uses the `crypto` module in Node.js and the Web Crypto API in browsers. These modules use unpredictable hardware random generator.

Uniformity

`random % alphabet` is a popular mistake to make when coding an ID generator. The spread will not be even; there will be a lower chance for some symbols to appear compared to others—so it will reduce the number of tries when brute-forcing.

Nano ID uses a better algorithm and is tested for uniformity.

Progressive JPEG

kyan.com/news/image-optimisation-techniques-for-web

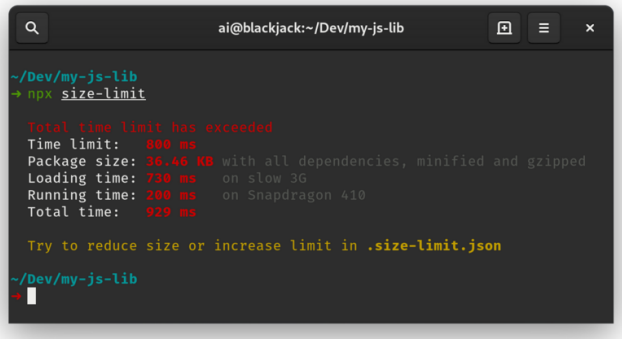
Advice 4: More illustration or code example

README.md

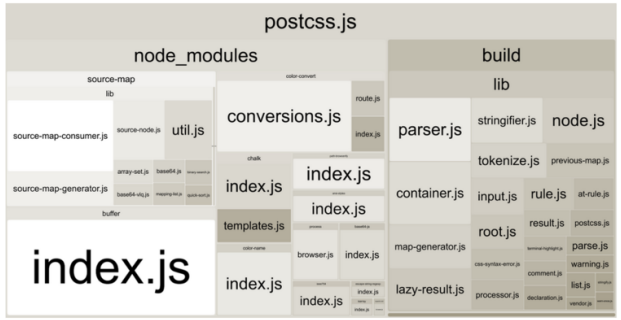
Size Limit

Size Limit is a linter for your JS application or library performance. It calculates the real cost of your JS for end users and throws an error if the cost exceeds the limit.

- Size Limit calculates **the time** it would take a browser **to download and execute** your JS. Time is a much more accurate and understandable metric compared to the size in bytes.
- Size Limit calculations include **all dependencies and polyfills** used in your JS.
- Add Size Limit to **Travis CI**, **Circle CI**, or another CI system to know if a pull request adds a massive dependency.



With `--why`, Size Limit can tell you *why* your library has this size and show the real cost of all your internal dependencies.



README.md

Storeon

A tiny event-based Redux-like state manager for React and Preact.

- **Small.** 173 bytes (minified and gzipped). No dependencies. It uses Size Limit to control size.
- **Fast.** It tracks what parts of state were changed and re-renders only components based on the changes.
- **Hooks.** The same Redux reducers. With hooks for **React** and **Preact**.
- **Modular.** API created to move business logic away from React components.

Read more about Storeon features in **our article**.

```
import createStore from 'storeon'

// Initial state, reducers and business logic are packed in independent modules
let increment = store => {
  // Initial state
  store.on('@init', () => ({ count: 0 }))
  // Reducers returns only changed part of the state
  store.on('inc', ({ count }) => ({ count: count + 1 }))
}

export const store = createStore([increment])
```

```
import useStoreon from 'storeon/react' // or storeon/preact

export default const Counter = () => {
  // Counter will be re-render only on `state.count` changes
  const { dispatch, count } = useStoreon('count')
  return <button onClick={() => dispatch('inc')}>{count}</button>
}
```

```
import StoreContext from 'storeon/react/context'

render(
  <StoreContext.Provider value={store}>
    <Counter />
  </StoreContext.Provider>,
  document.body
)
```

Advice 5: More lists and bolds

README.md

Storeon

A tiny event-based Redux-like state manager for React and Preact.

- **Small.** 173 bytes (minified and gzipped). No dependencies. It uses Size Limit to control size.
- **Fast.** It tracks what parts of state were changed and re-renders only components based on the changes.
- **Hooks.** The same Redux reducers. With hooks for **React** and **Preact**.
- **Modular.** API created to move business logic away from React components.

Read more about Storeon features in **our article**.

Important parts are highlighted

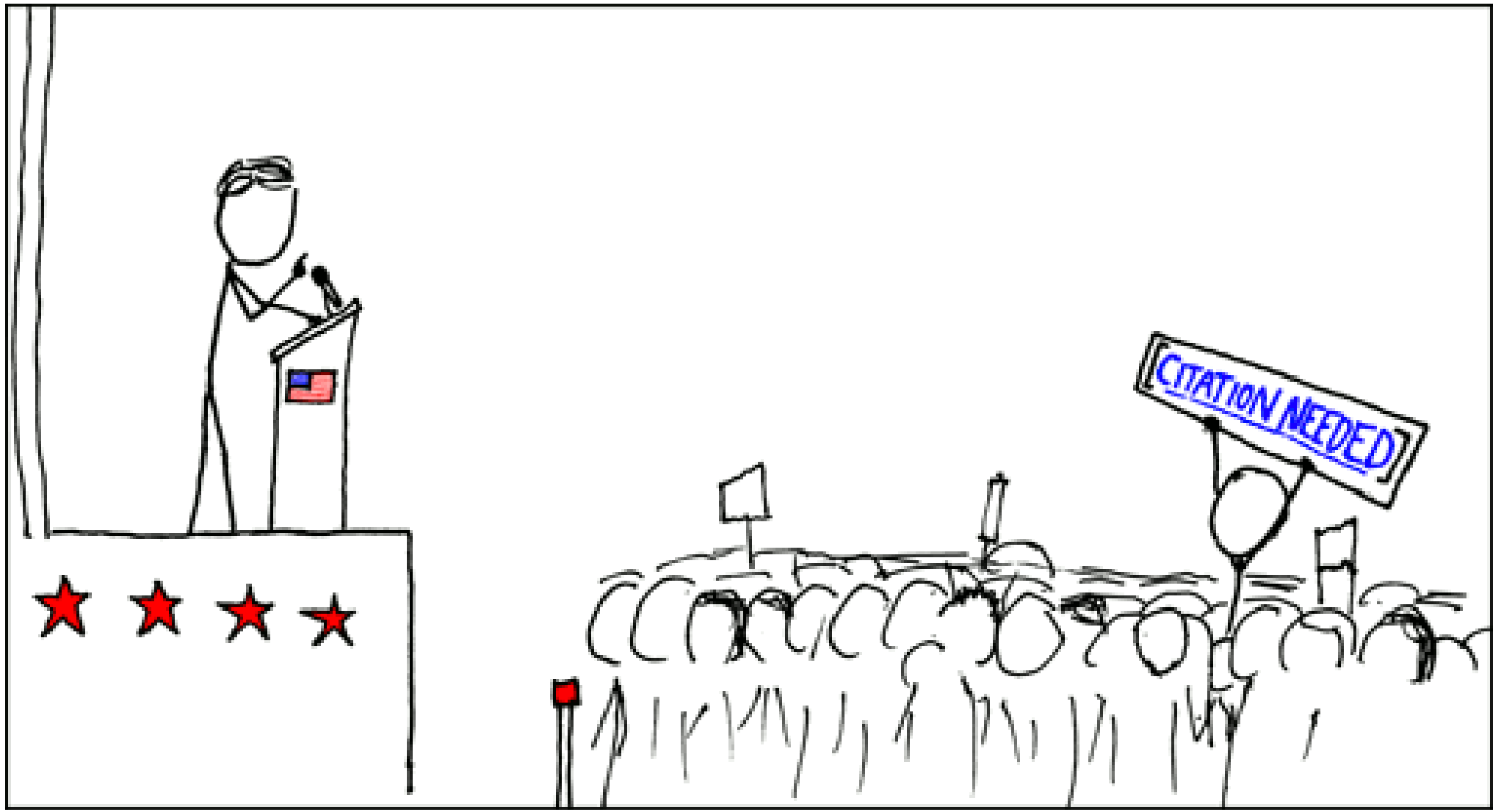
README.md

Storeon

A tiny event-based Redux-like state manager for React and Preact.

- **Small.**
- **Fast.**
- **Hooks.** **React** and **Preact**.
- **Modular.**

Advice 6: Use real proofs



Size Limit to prove “small”

README.md

Nano ID

A tiny, secure, URL-friendly, unique string ID generator for JavaScript.

- **Small.** 141 bytes (minified and gzipped). No dependencies. Size Limit controls the size.
- **Safe.** It uses cryptographically strong random APIs and tests distribution of symbols.
- **Fast.** It’s 16% faster than UUID.
- **Compact.** It uses a larger alphabet than UUID (`A-Za-z0-9_-`). So ID size was reduced from 36 to 21 symbols.

The generator supports Node.js, React Native, and all browsers.

Show benchmarks to prove “fast”

README.md

Nano ID

A tiny, secure, URL-friendly, unique string ID generator for JavaScript.

- **Small.** 141 bytes (minified and gzipped). No dependencies. Size Limit cont
- **Safe.** It uses cryptographically strong random APIs and tests distribution of
- **Fast.** It’s 16% faster than UUID.
- **Compact.** It uses a larger alphabet than UUID (`A-Za-z0-9_-`). So ID size w

The generator supports Node.js, React Native, and all browsers.

Benchmark

```
$ ./test/benchmark
nanoid                    693,132 ops/sec
nanoid/generate           624,291 ops/sec
uid.sync                  487,706 ops/sec
uuid/v4                   471,299 ops/sec
secure-random-string      448,386 ops/sec
shortid                    66,809 ops/sec

Async:
nanoid/async              105,024 ops/sec
nanoid/async/generate     106,682 ops/sec
secure-random-string       94,217 ops/sec
uid                        92,026 ops/sec

Non-secure:
nanoid/non-secure       2,555,814 ops/sec
rndm                    2,413,565 ops/sec
```

Show code to prove “better API”

```
// Redux
  switch (action.type) {
    case 'INCREMENT':
      return state + 1
    case 'DECREMENT':
      return state - 1
    default:
      return state
  }

// Storeon
store.on('INCREMENT', ({ count }) ⇒ ({ count: count + 1 }))
store.on('DECREMENT', ({ count }) ⇒ ({ count: count - 1 }))
```

Advice 7: Step-by-step starting guide

Usage

Step 1: convert all your JPEG/PNG images to WebP by Squoosh. Set checkbox on `Lossless` for PNG images and remove it for JPEG.

We recommend `Reduce palette` for most of the PNG images.

Save WebP images in the same places of JPEG/PNG images: `img/bg.png` → `img/bg.webp`.

Step 2: use `<picture>` to insert WebP images in HTML:

```
- <img src="/screenshot.jpg" alt="Screenshot">
+ <picture>
+   <source srcset="/screenshot.webp" type="image/webp">
+   <img src="/screenshot.jpg" alt="Screenshot">
+ </picture>
```

Step 3: install `webp-in-css`. For npm use:

```
npm install --save-dev webp-in-css
```

Step 4: add JS script to your client-side JS bundle:

```
+ require('webp-in-css')
```

Since JS script is very small (128 bytes), the best way for landings is to inline it to HTML:

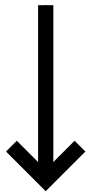
```
+   <script><%= readFile('node_modules/webp-in-css/index.js') %></script>
  </head>
```

Your users


OH MY GOD EVERY THINGS ON FIRE
WHAT THE HELL IS MY PROBLEM
OH MY GOD JESUS FUCK
WHAT THE FUCK WAS I EVEN THINKING

Be very specific

Step 5: use PostCSS.



Step 5: check do you use PostCSS already in your bundler. You can check `postcss.config.js` in the project root, `"postcss"` section in `package.json` or `postcss` in bundle config.

If you don't have it already, add PostCSS to your bundle:

- For webpack see postcss-loader docs.
- For Parcel create `postcss.config.js` file. It already has PostCSS support.
- For Gulp check gulp-postcss docs.

Help with choice

Step 5: check do you use PostCSS already in your bundler. You can check `postcss.config.js` in the project root, `"postcss"` section in `package.json` or `postcss` in bundle config.

If you don't have it already, add PostCSS to your bundle:

- For webpack see postcss-loader docs.
- For Parcel create `postcss.config.js` file. It already has PostCSS support.
- For Gulp check gulp-postcss docs.

Separated sections for specific use cases ❤️

Usage for Applications and Big Libraries

This guide is for two use cases:

- Application with bundler (like webpack or Parcel). Any React or Vue.js application is this use case.
- JS libraries, which use webpack or Rollup to build `dist/umd/lib.produciton.js`-kind file to put it to npm package. React is a good example.

Usage for Small Libraries

This guide is for small JS libraries with many small separated files in their npm package.
Nano ID or Storeon could be a good example.

Test your starting guide on new project

```
npm init
```

...

everything works

Chapter 6

Slow growing

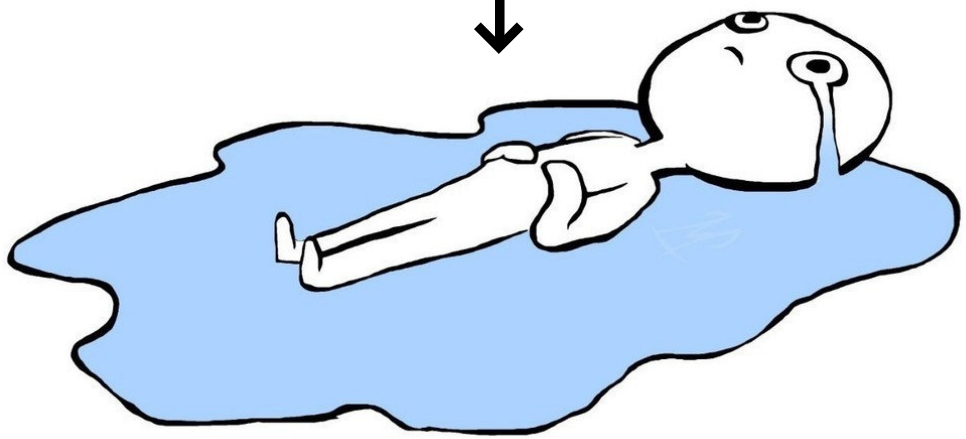
Wrong way to promote the project

Post 1 tweet

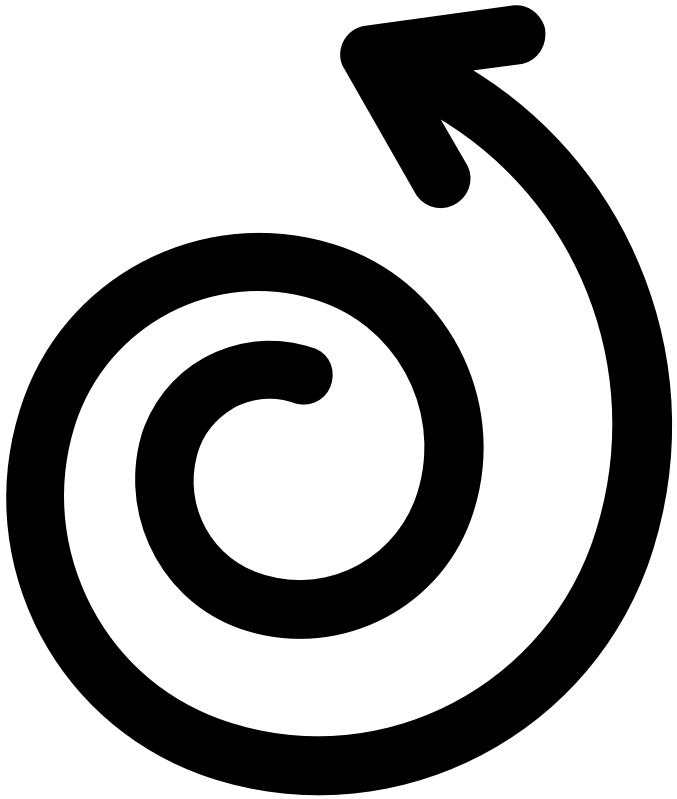
↓

Got 4 retweets

↓



Grows by small iterations



Each iteration contains

1. Action (feature, tweet, article, etc)
2. Feedback
3. Fixing the project according to feedback

Next iteration bigger than previous one

1. Bigger action
2. More feedback
3. More things to fix

1st iteration: explain the idea to friends

And find the way to describe the project

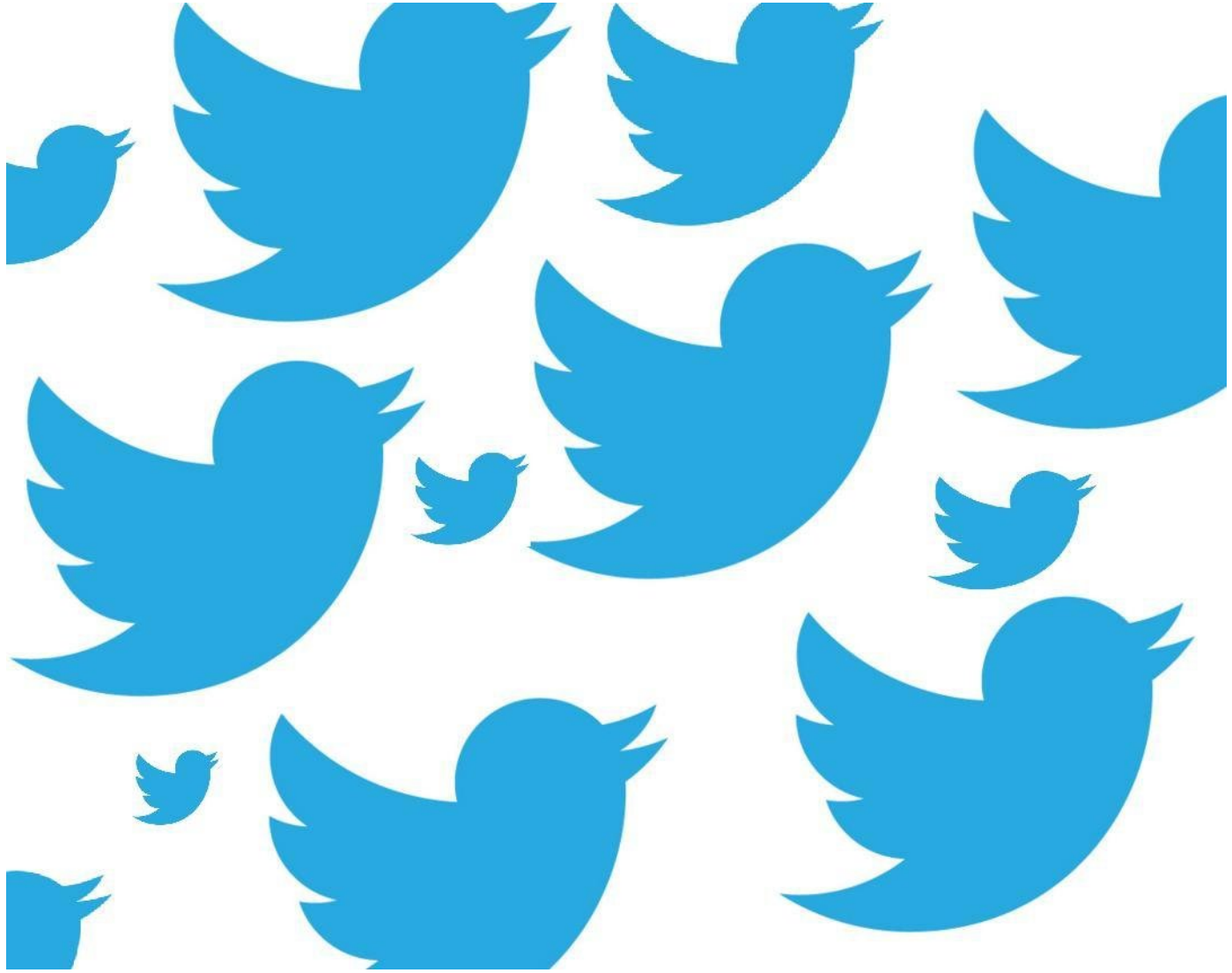
2nd iteration: convince colleagues to try it

Fix issues, DX,
and install docs

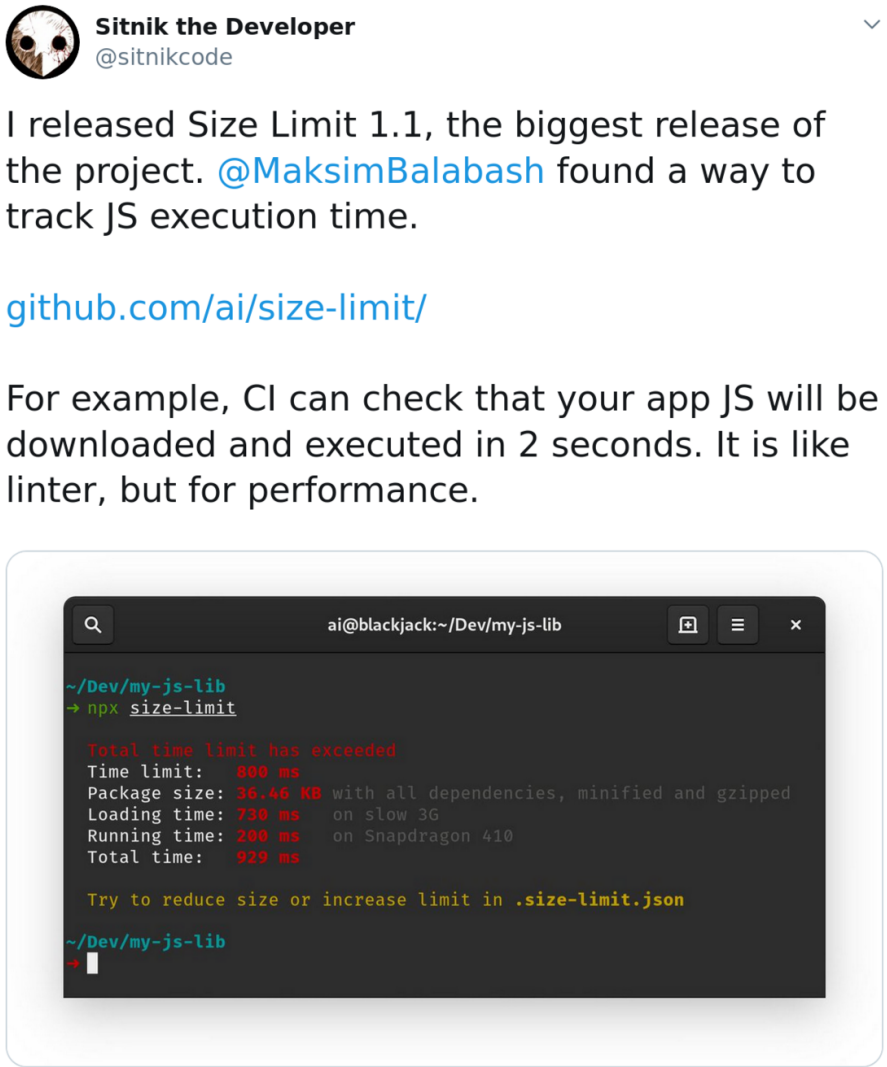
3rd iteration: post announce to the Twitter

And find the best way to describe projects

And tweet again each big feature



Good tweet



⟵ New feature which was added

⟵ Project description

⟵ Code example or screenshot

4th iteration: reddit and hacker

And fix compatibility issues

5th iteration: write article on your language

habr.com

6th iteration: write English article

dev.to

smashingmagazine.com

css-tricks.com

7th iteration: send PRs

Replace shortid to nanoid #9

Merged **jhen0409** merged 1 commit into `zalmoxisus:master` from `unknown repository` on 5 Nov 2017

Conversation 2 | Commits 1 | Checks 0 | Files changed 2

ai commented on 31 Oct 2017 • edited ▾ Contributor

What do you think of replacing `shortid` to `nanoid` (without big changes for users). Benefits:

1. `shortid` is not maintained anymore (and issue tracker has many important issues). I tried to took maintenance, but found that I can't fix `shortid` with keeping API.
2. `shortid` has problem with random generator (issue) and ID uniformity (issue). `nanoid` use hardware random generator and has special uniformity tests (docs).
3. `nanoid` is just 179 bytes. `shortid` is 2 times bigger (about 0.5 KB).
4. `nanoid` is 10x times faster than `shortid`.

@zalmoxisus what do you think?

👍 1

 `Replace shortid to nanoid` `19cf020`

jhen0409 approved these changes on 5 Nov 2017 **View changes**

 **jhen0409** merged commit `5f8b32d` into `zalmoxisus:master` on 5 Nov 2017

github.com/zalmoxisus/remotedev-utils/pull/9

Use previous success in next iterations

Who Uses Size Limit

- MobX
- Material-UI
- Autoprefixer
- PostCSS reduced 25% of the size.
- Browserslist reduced 25% of the size.
- EmojiMart reduced 20% of the size
- nanoid reduced 33% of the size.
- React Focus Lock reduced 32% of the size.
- Logux reduced 90% of the size.

Don’t step making interations

tenor.com/view/wheel-of-fortune-gif-5078993

Don’t spam: be regular

every 2 weeks or every month

Why you need to several tweets?

blog.newtonhq.com/the-problem-with-infinite-scrolling-feeds-e3d1aad2c078

Why you need many iterations?

Feedback will help to fix it

How people choose

Irrational choose across **popular** options

How long they will support the project?

How many resources project has behind?

Regular tweets answer to these questions

Irrational choose across **popular** options

How long they will support the project?

How many resources project has behind?

Chapter 7

Working with community

You did everything right

- Explain benefits in docs
- Good step-by-step intructions
- Regular tweets, articles, talks

And got issues from users

The circle of guilt

Many issues without answer

Blaming yourself for being bad maintaner

Depression and lack of productivity

Even more issues without answer

You don’t need to write code

ai commented on 23 Jan — Member

OK, let’s allow to pass a string. Do you want to send PR?

opnet commented on 23 Jan — Author

Sure, I can do that.

dmarkhas added a commit to dmarkhas/browserslist that referenced this issue on 19 Feb

`Accept string as config property, fixes browserslist#347` ··· Verified `619503b`

ai closed this in `5db376c` on 23 Feb

15 min in the morning to answer on new issues

Dropbox cinemagraphs

It's OK to ask wait

Don't use locale-aware comparison for browser names ✓ 196c739

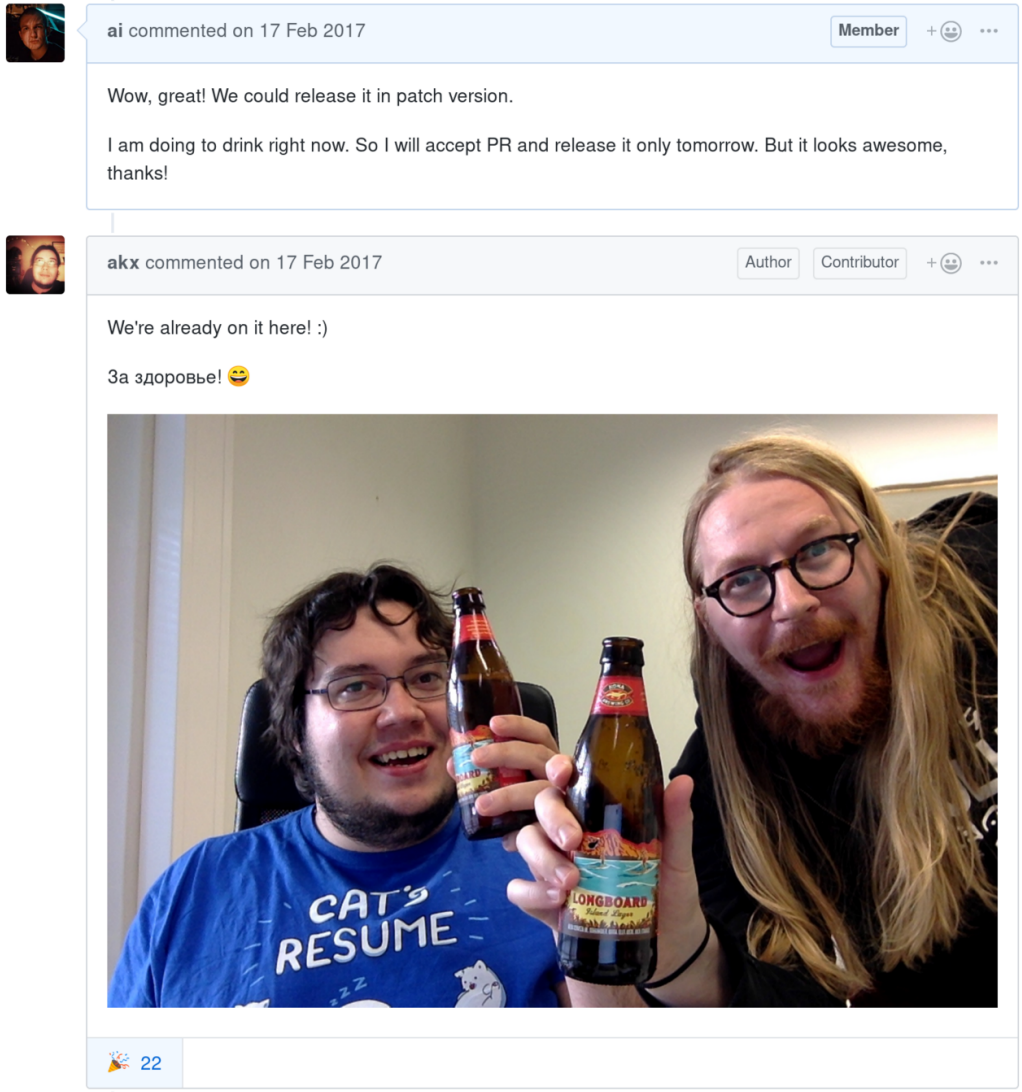
ai commented on 17 Feb 2017 **Member**

Wow, great! We could release it in patch version.

I am doing to drink right now. So I will accept PR and release it only tomorrow. But it looks awesome, thanks!

github.com/browserslist/browserslist/pull/123

Just an answer could make them your fans



github.com/browserslist/browserslist/pull/123

Do something to avoid issues in future

Add warning

```
try {
  parser.parse()
} catch (e) {
  if (process.env.NODE_ENV !== 'production') {
    if (e.name === 'CssSyntaxError' && opts && opts.from) {
      if (/\.scss$/i.test(opts.from)) {
        e.message += '\nYou tried to parse SCSS with ' +
                     'the standard CSS parser; ' +
                     'try again with the postcss-scss parser'
```

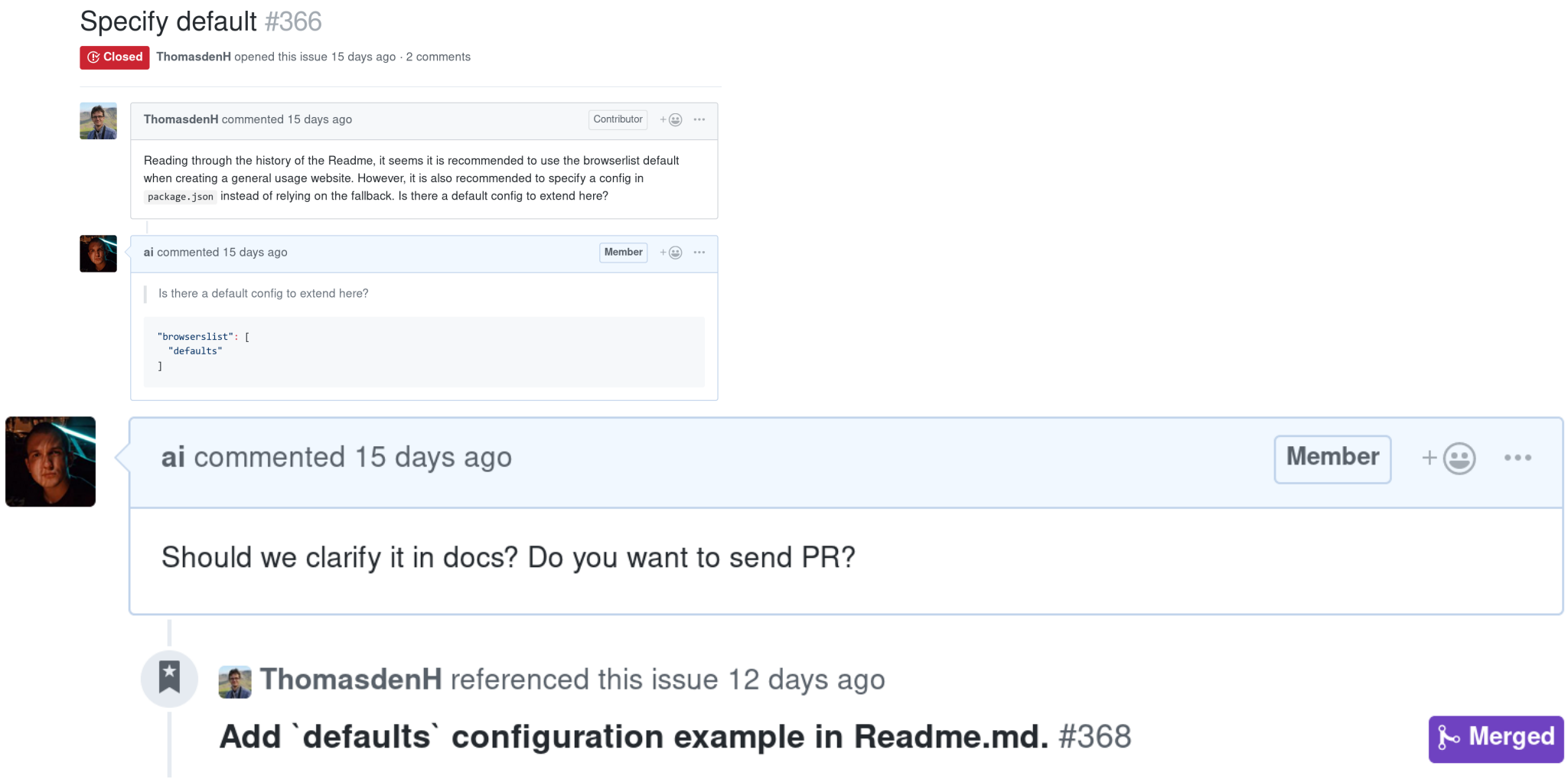
Stopped issues about using PostCSS for Sass sources

postcss/lib/parser.js

Add note to docs

- FAQ
  - Does Autoprefixer polyfill Grid Layout for IE?
  - No prefixes in production
  - What is the unprefixed version of `-webkit-min-device-pixel-ratio`?
  - Does it add polyfills?
  - Why doesn't Autoprefixer add prefixes to `border-radius`?
  - Why does Autoprefixer use unprefixed properties in `@-webkit-keyframes`?
  - How to work with legacy `-webkit-` only code?
  - Does Autoprefixer add `-epub-` prefix?
  - Why doesn't Autoprefixer transform generic font-family `system-ui`?

Ask issue authors to change docs



github.com/browserslist/browserslist/issues/366

How to deal with negative feedback?

bolk 16 апреля 2013 в 20:50 –5

> Эту проблему немного решил Stylus — у него синтаксис примесей не отличается от обычных свойств, так что префиксы добавляются невидимо. Впрочем, проблема актуальности и значений всё равно остались.

И как же решил проблему актуальности данный велосипед? Чем он лучше Стайлуса? Тем что грузит данные с другого адреса? Ну и что? Почему это лучше?

Александр Евгеньевич
чем он так упоролся
12 фев 2015 Ответить

Саша Саша ответила Андрею
Андрей, даже упомянул в статье))) чел то что ты делал логух которым никто не пользуется и который никому не зашёл или постцсс который тоже никому не нужен, не делает тебя каким то крутым челом, который даже упоминает в статье, ты ноунейм по сути, не выебуйся
25 апр в 7:03 Ответить 3

There is no ideal answer

Never Say No to Panda

My best way is to start asking questions

Андрей Андреев
Что старый логотип — хренотень, что новый.
6 янв 2016 Ответить 1

Андрей Ситник ответил Андрею
Андрей, согласен. Раз мы сейчас делаем новый сайт, то как раз может слегка подправить и форму логотипа. Можете сделать набросок?
7 янв 2016 Ответить

Андрей Андреев
Более приближённая к оригиналу заготовочка.

8 янв 2016 Ответить

Андрей Ситник ответил Андрею
Андрей, а создай issue в https://github.com/postcss/brand — пусть другие ребята тоже выскажутся
8 янв 2016 Ответить

vk.com/wall-90484126_1738

Sometimes, people just need to talk

They can become your fan after a talk

Chapter 8

Resume

Success formula

Project popularity **=**

Your popularity **+**

Promotion **+**

Benefits for users **+**

Luck

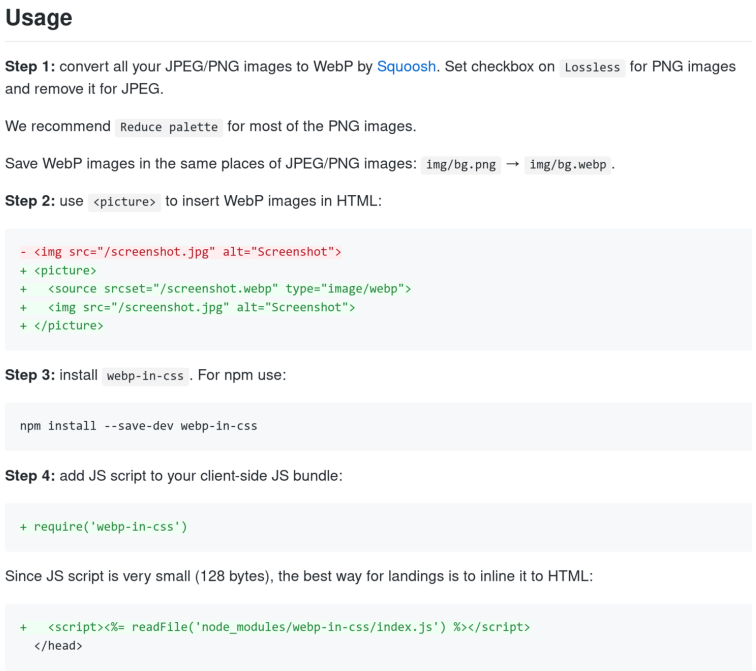
How people choose

Irrational choose across **popular** options

How people read

README.md is critical

1. Clear description
2. How the product is useful for users
3. The difference from other products
4. Easy step-by-step instruction



Usage

Step 1: convert all your JPEG/PNG images to WebP by Squoosh. Set checkbox on `Lossless` for PNG images and remove it for JPEG.

We recommend `Reduce palette` for most of the PNG images.

Save WebP images in the same places of JPEG/PNG images: `img/bg.png` → `img/bg.webp`.

Step 2: use `<picture>` to insert WebP images in HTML:

```
- <img src="/screenshot.jpg" alt="Screenshot">
+ <picture>
+   <source srcset="/screenshot.webp" type="image/webp">
+   <img src="/screenshot.jpg" alt="Screenshot">
+ </picture>
```

Step 3: install `webp-in-css`. For npm use:

```
npm install --save-dev webp-in-css
```

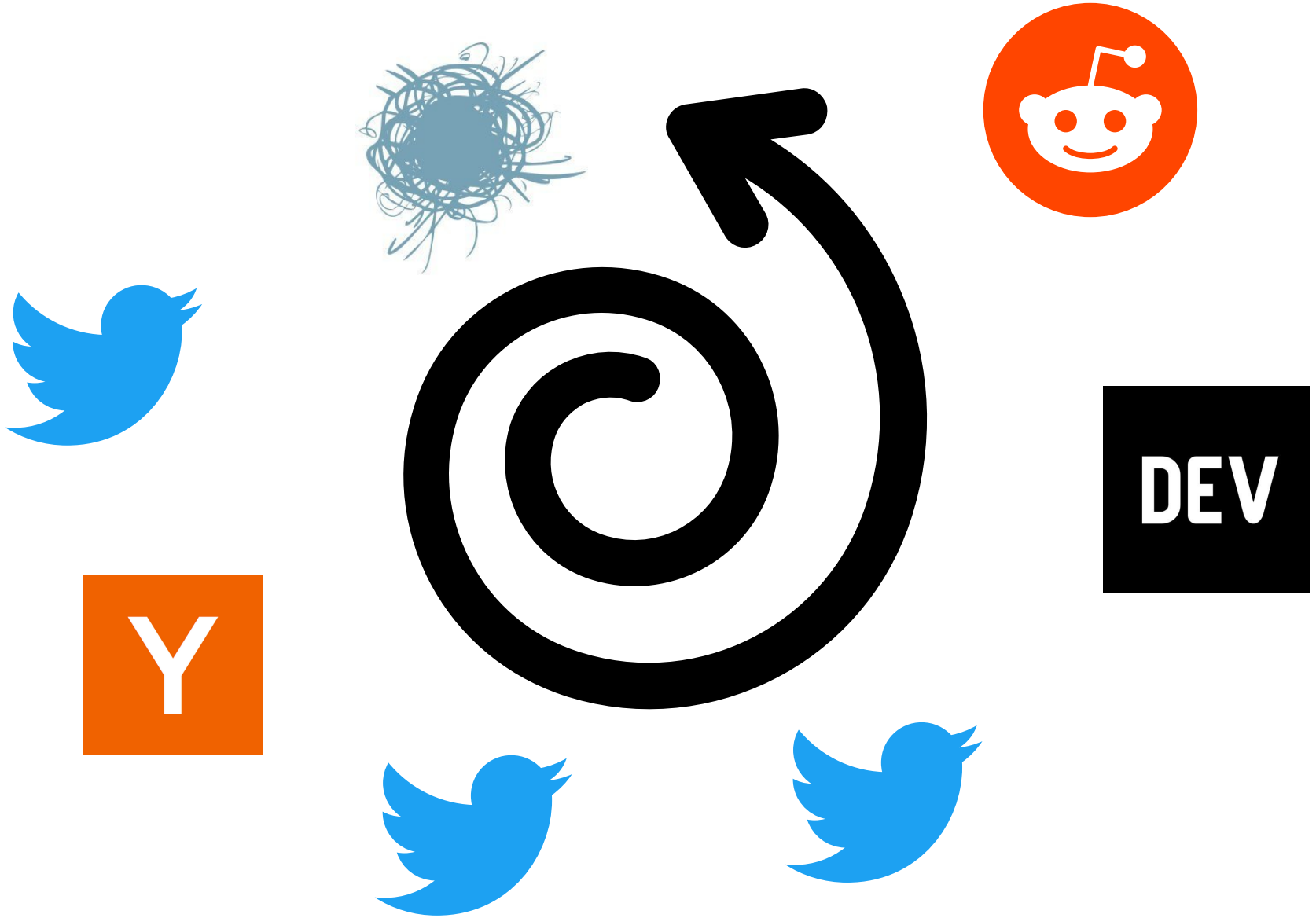
Step 4: add JS script to your client-side JS bundle:

```
+ require('webp-in-css')
```

Since JS script is very small (128 bytes), the best way for landings is to inline it to HTML:

```
+   <script><%= readFile('node_modules/webp-in-css/index.js') %></script>
  </head>
```

Regular tweets, articles, talks



Thanks

andrey_sitnik

DM me your open source

evilmartians_ru

evilmartians.com