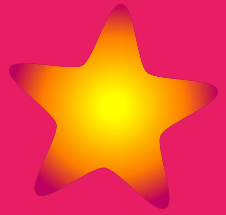


Hype Driven Development sucks?

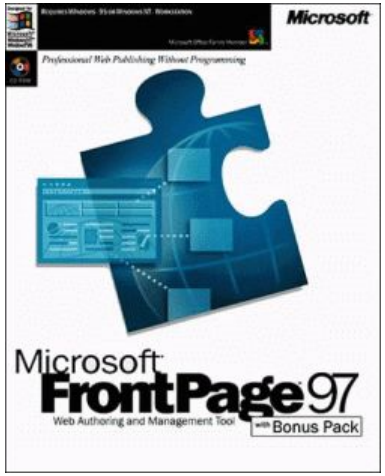
or rather, how to do it well

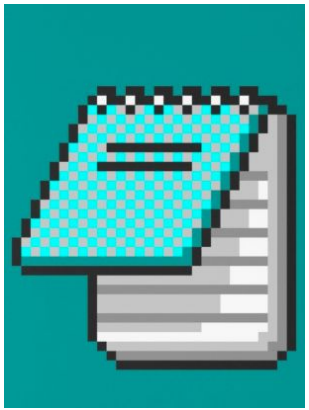
by Jacek Tomaszewski

Full-stack Web Developer for over 15 years

jtom.me / jtompl / medium.com/@jtomaszewski

Web Hypes History


Microsoft
Professional Web Publishing Without Programming
Microsoft
FrontPage 97
Web Authoring and Management Tool
with Bonus Pack




W3C
XHTML
1.0
W3C
CSS

2001

Time

SITE

News/Treść

CIEKAWE

News
Chat()
Księga Gości
Zobacz
Dodaj wpis
Forum
Linki
Link 2 us
R/S Club
Redakcja

R.I.P.

2003-07-04 22:16:49 - Autor: Wizard - Drukuj - Wyślij e-mailem

W dzisiejszym trzecim pisanym przeze mnie newsie oficjalnie ogłaszam, że POKéMON FRIENDS zostało zamknięte!! Mamy nadzieje, że nigdy nie zapomnicie tej strony i ludzi którzy nad nią pracowali... Jestem pewnien, że nie zapomnicie, a czemu?? Ponieważ nie rozstajemy się na długo, w najbliższych kilku dniach rusza bardzo ulepszona strona... Mamy nadzeje, że przypadnie wam do gustu równie, a moze i bardziej jak Pokémon Friends. Tutaj, narazie moecie nadal gadać na chacie i wpisywać się w księdze gości... newsy też na razie nie znikną. No więc... do zobaczenia.

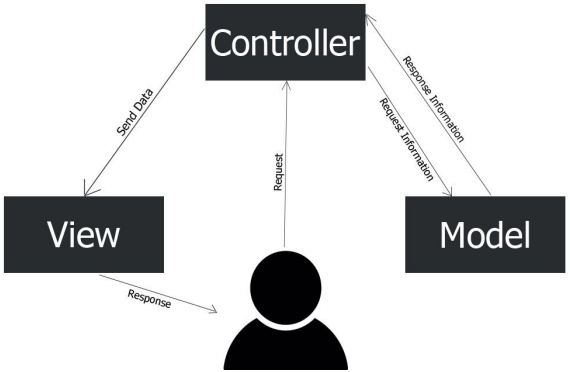
PS. Nikt ze składu Pokémon Friends nie odchodzi, wszyscy redaktorzy dalej będą spełniać swoje obowiązki ;)

SONDA

SONDA

Jakie newsy wolicie?

Model-View-Controller



ZALOGUJ SIĘ / ZAREJESTRUJ SIĘ | DODAJ OGŁOSZENIE

terazzaraz.pl

Praca w gronie znajomych

Lubię to! 300

WYSZUKIWARKA OFERT | GRUPY | LUDZIE | KARIERA

SZUKAJ PRACY

MAMY **91 733** AKTUALNE OFERTY

wpisz słowo kluczowe

podaj miejscowość

SZUKAJ

WIĘCEJ OPCJI

DLACZEGO WARTO ZAŁOŻYĆ KONTO W SERWISIE

04.

UCZESTNICZYSZ W CIEKAWYCH DYSKUSJACH

ZAŁÓŻ KONTO / ZALOGUJ SIĘ

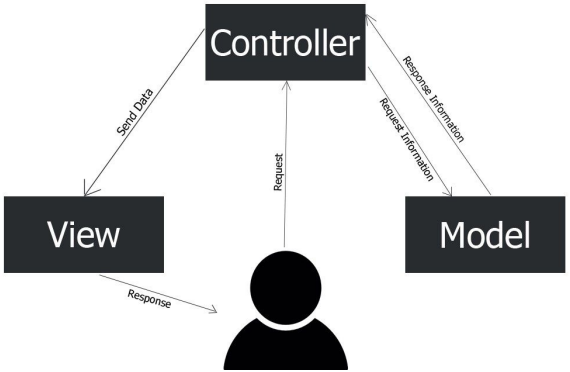
Nigdy nie spamujemy, nie pokazujemy Twojej aktywności w serwisach społecznościowych. Twoje dane z Facebooka to jedynie adres e-mail i po zalogowaniu możliwość udostępniania ogłoszeń oraz informacje o znajomych.

Zaloguj się przez facebook

Zaloguj się przez

RAILS

Model-View-Controller

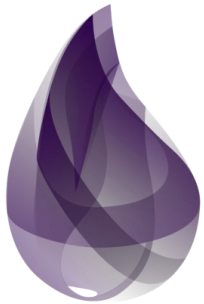


PostgreSQL

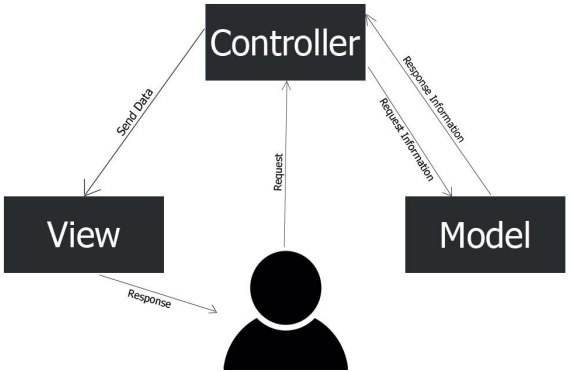
NGINX

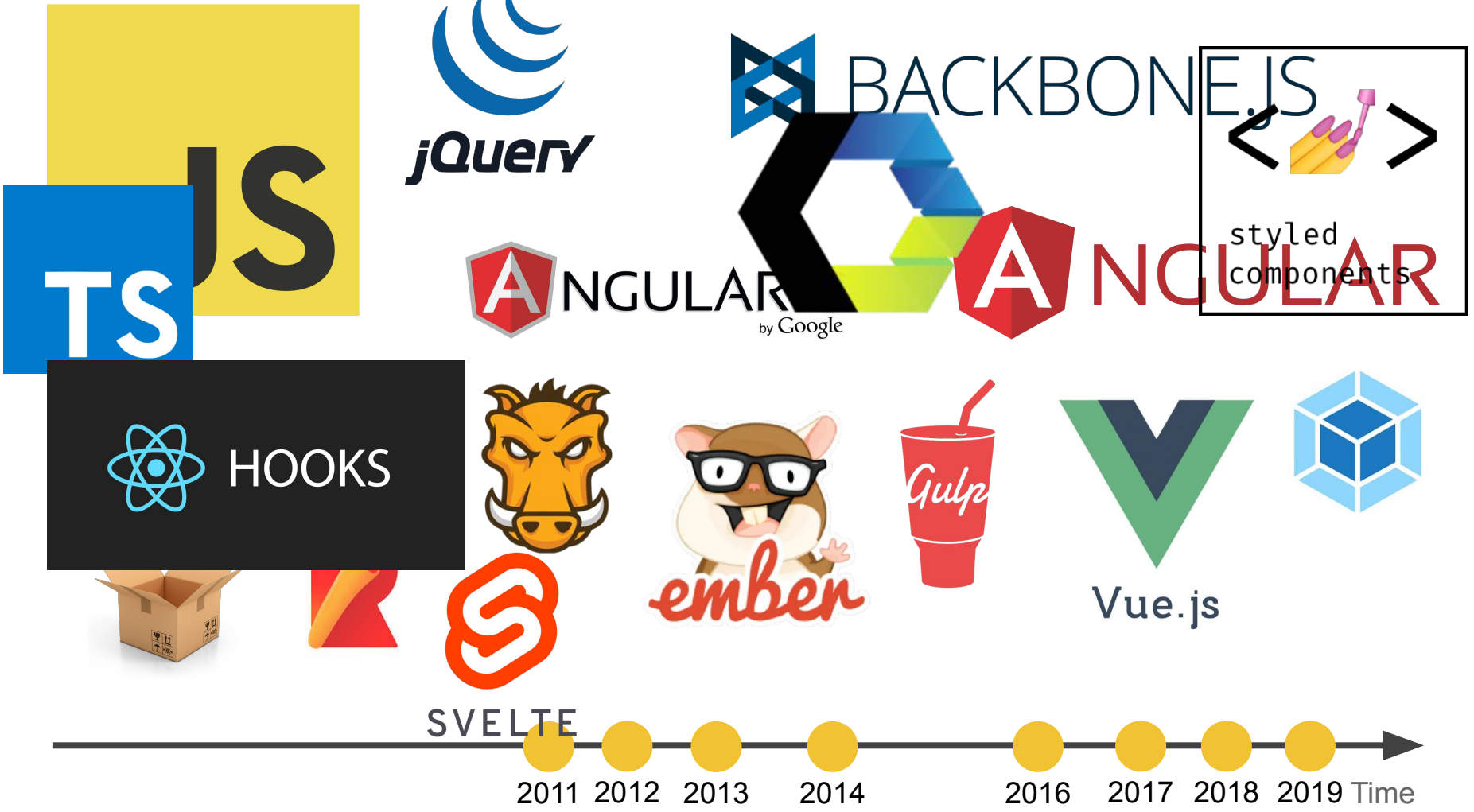
2013

Time



Model-View-Controller




JS
TS
jQuery
BACKBONE.JS
styled
components
ANGULAR
by Google
ANGULAR
HOOKS
Gulp
ember
Vue.js
SVELTE
2011
2012
2013
2014
2016
2017
2018
2019
Time

Still unanswered

1. Server-side render
2. Client-side render/hydration/enhancement
3. Logic in components / Redux / Mobx
4. Two-way binding vs one way flow
5. Dependency Injection: import/export singletons?
   Component Context (React)?
   Component DI context (Angular)?
6. Build setup: Webpack? Parcel? Rollup?
7. Framework: Angular? React? Vue? Ember? Svelte? Web Components?
8. CSS-in-JS? Sass or CSS3? BEM?

   ...

Thought-provoking question

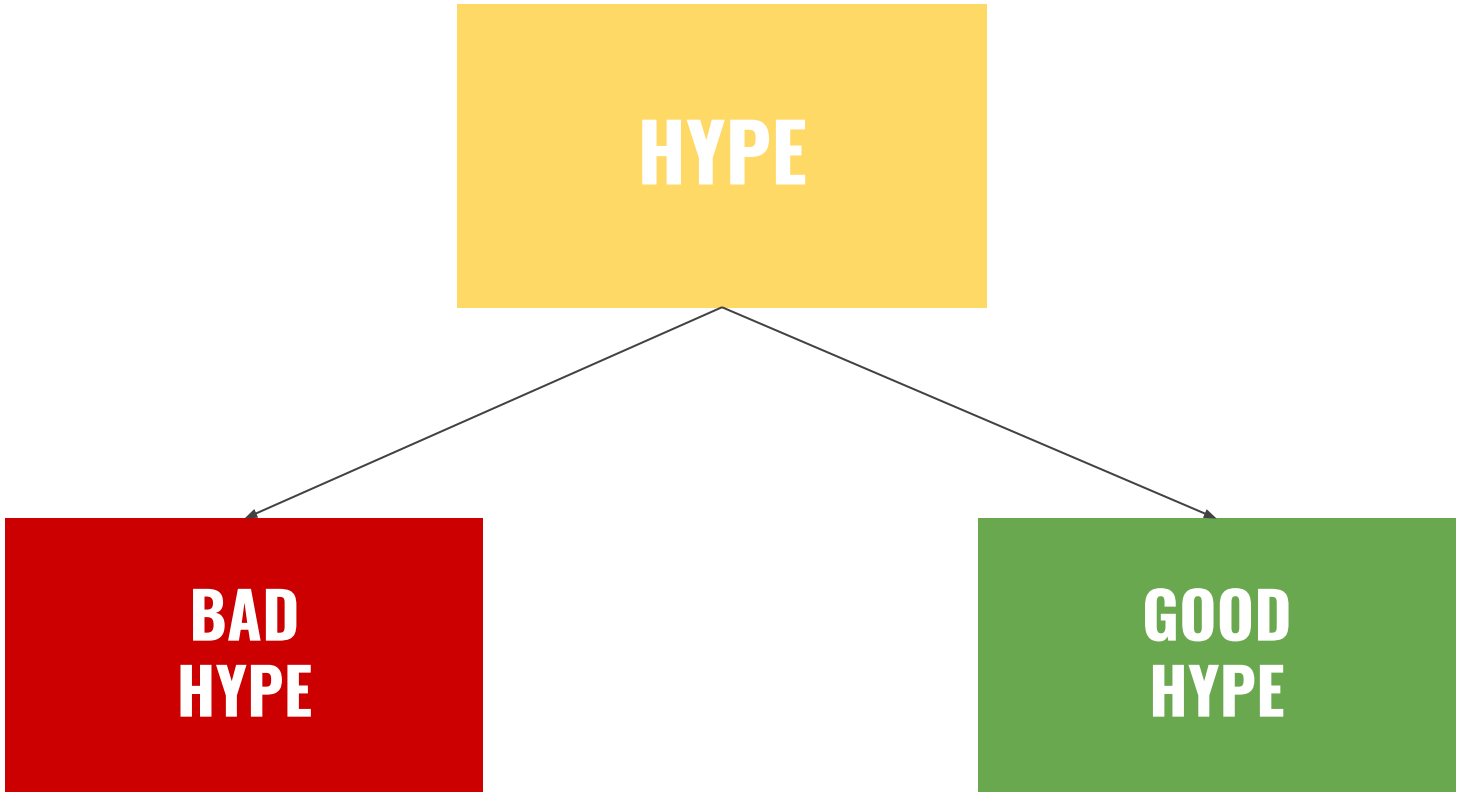
If you've created a website yesterday…

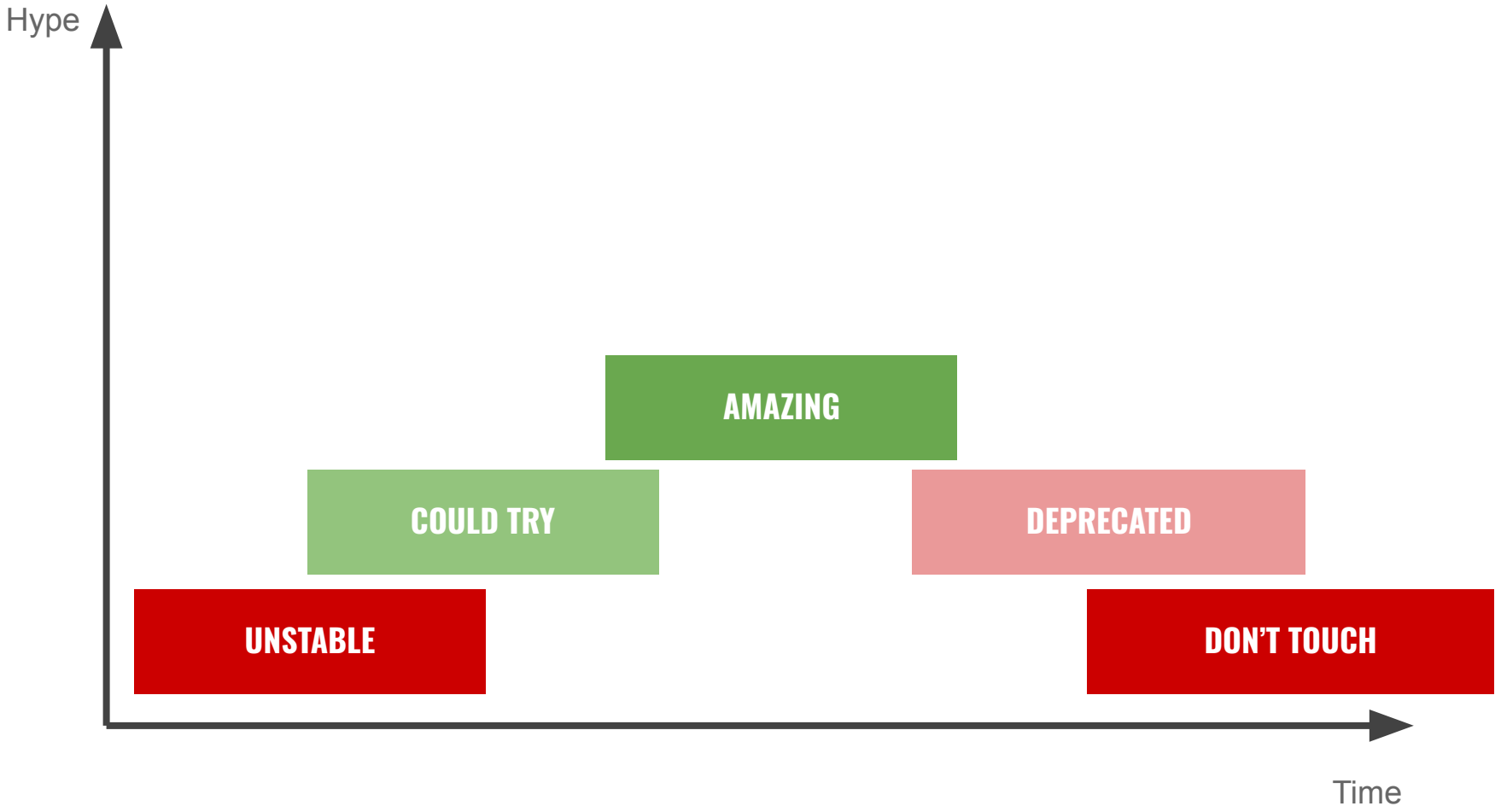
… will you do it more quickly today?

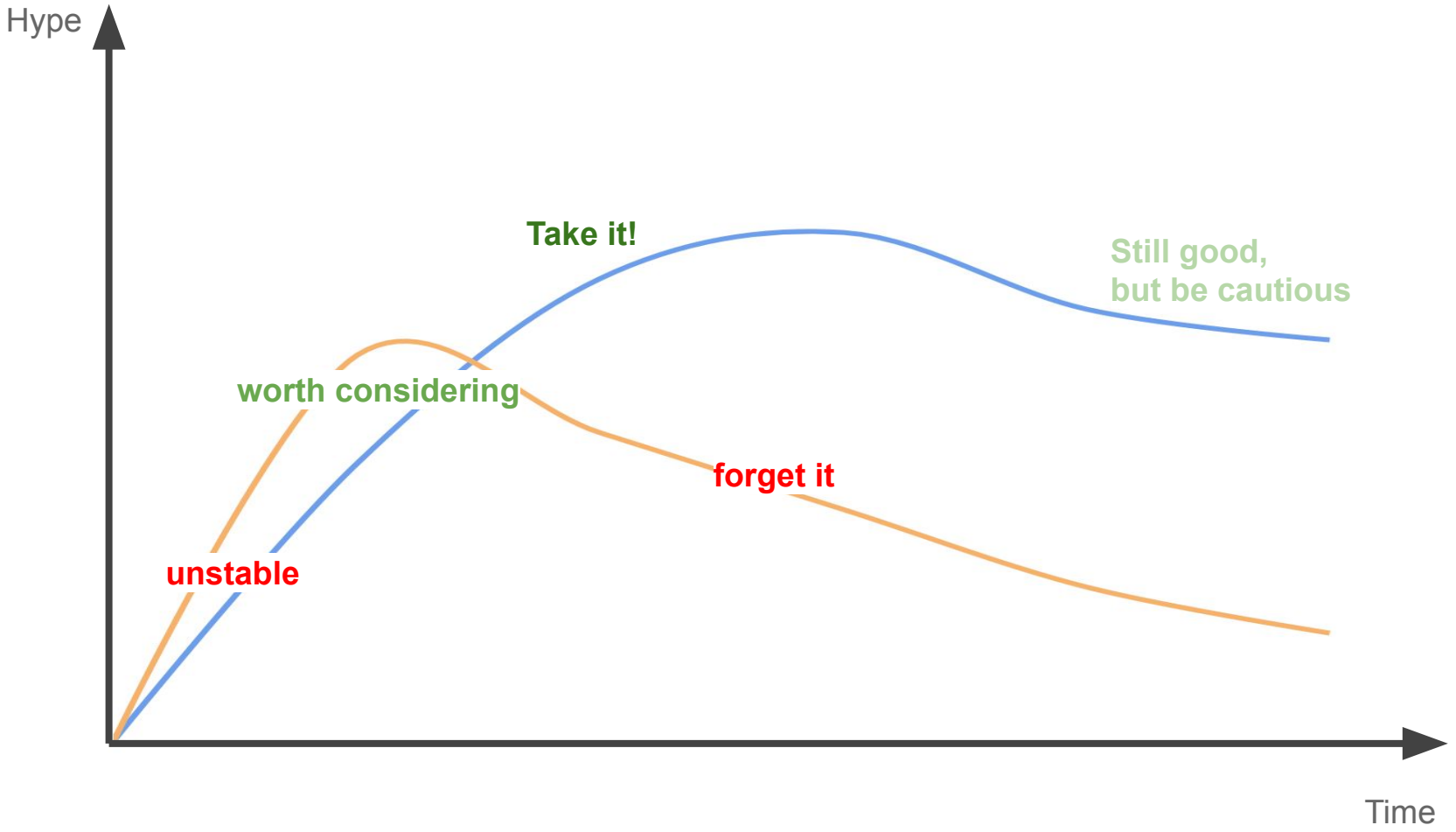
… will you use the same set of tools?

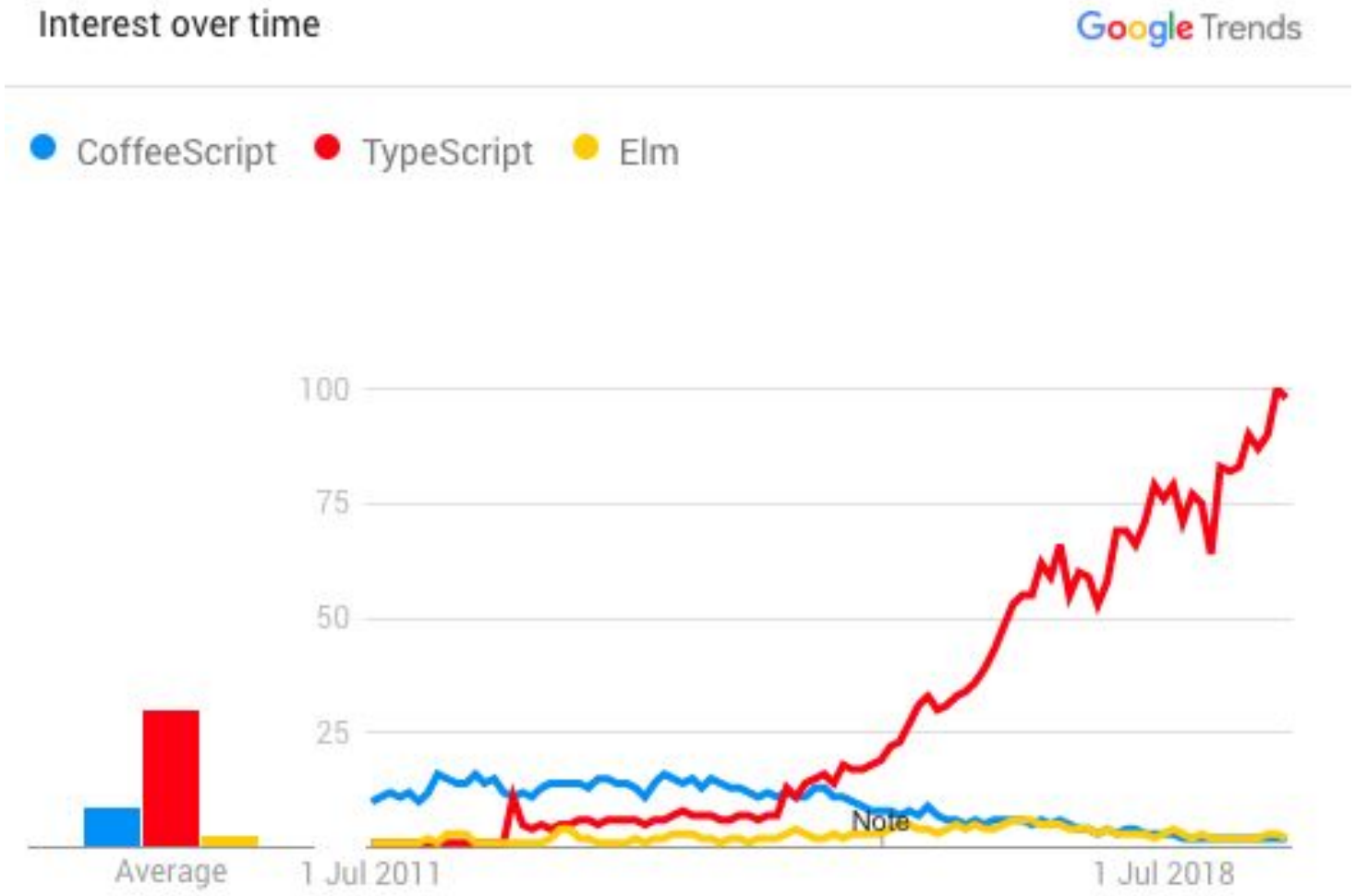
Truth is
we've been following hypes all along

(Which is good. If not, we'd still be doing jQuery)


HYPE
BAD HYPE
GOOD HYPE


Hype
AMAZING
COULD TRY
DEPRECATED
UNSTABLE
DON'T TOUCH
Time


Hype
Take it!
Still good,
but be cautious
worth considering
forget it
unstable
Time


Interest over time
Google Trends
CoffeeScript
TypeScript
Elm
100
75
50
25
Note
Average
1 Jul 2011
1 Jul 2018

~~Hype Driven Development sucks...?~~

Is *hype X* is bad or good?

~~Hype Driven Development sucks...?~~

~~Is *hype X* is bad or good?~~

Is it a good moment to join the *hype X*?

Should I join *hype X*?

~~Should I join *hype X*?~~

~~Should I join *hype X*?~~

I have *problem A*.

How can I solve it?

I have *problem A*.
How can I solve it?

I have *problem A*.
I can solve it with *X*,
but it has pros *a,b,c* and cons *d,e,f* .
Shall I do it?

Things to consider

when joining a hype

Community

- Popularity
  - GH stars
  - Google
  - Npmtrends.com
  - Local meetups

- Support
  - GH open issues
  - PRs merge frequency
  - Commit history
  - Release history

Documentation

- API docs
  - API fully and correctly documented?
- Usage examples
  - Up-to-date?
- Tests
  - Unit tests? E2E/Visual tests? Performance tests?
- Environment support
  - Legacy browsers?
  - Node.js (SSR)?
  - A11y?

Installation cost

- Setup time

- Edge cases, server-side, legacy browser support

Learning cost

- Your current / future team is familiar with it?

- Is easy to learn?
  - Simple API or a new programming concept?
  - “Don’t do it” scenarios?

Recruitment

- Attracts good devs?
- For how long?

Developers productivity

- Bugs probability
- Code simplicity

App performance

- **Meaningful** impact on the performance

Technology debt

- Complicates codebase?
  - Impacts a lot of files?

- Requires adding eslint/styleguide rules?

Deprecation / Abandonment risk

- After it’s abandoned, will it impact the project?
  - Stability over Time
  - Recruitment
  - Eventual Replacement Cost

- When will it get abandoned?
  - Maintainers
  - General Popularity
  - Business Popularity
  - Competition

Good Rules of

Safe Hype Driven Development

Avoid hypes that are about to die

- ~~ES6~~ ES7
- ~~lodash.map~~ Array.prototype.map
- ~~Flow~~ TypeScript

Prefer native solutions if they do the job

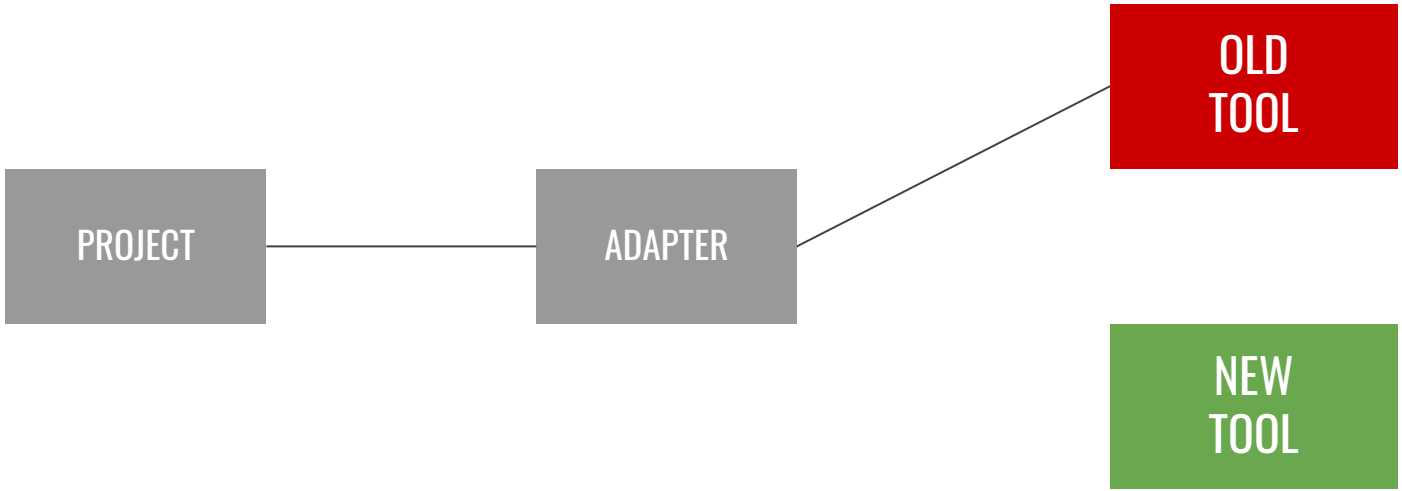
- Why `*axios*`, `*frisbee*`, if `*fetch*` is fine enough?
  - If you don’t have a specific reason for it, go with `fetch`!

Avoid hypes that you don’t need

- *“GraphQL/styled-components/redux everywhere!”*
  - AKA “Loudest guy in the room / on Twitter”

Use Adapter interface

- If the tool gets deprecated, you will need to switch it in only one place



Leave your personal taste aside

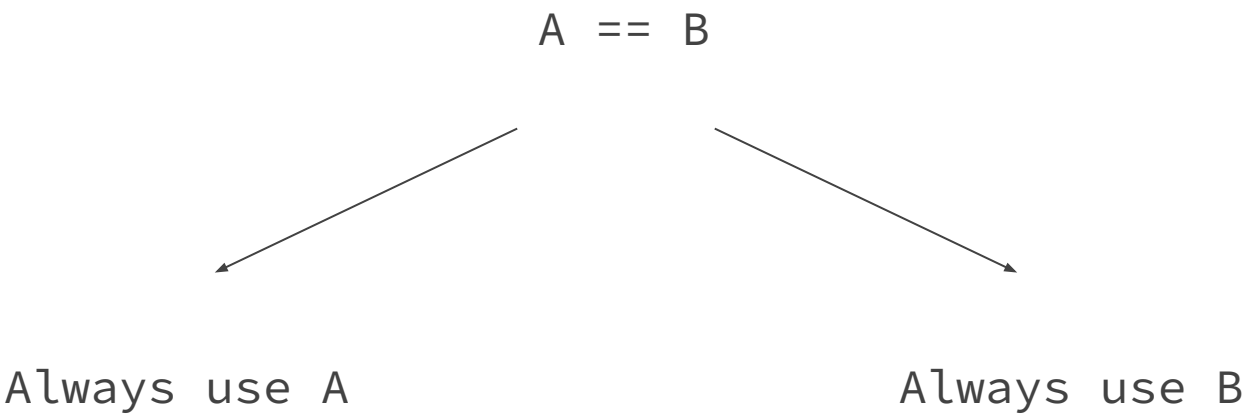
- Project’s Good > Your Personal Preference
- Avoid “*choose random*”, “*do what you like*”

Choose one thing over two

A > B

Always use A

Choose one thing over two



Choose one thing over two

A is usually better than B,
but in rare case C1, B is better

Always use A.
In cases like C1, always use B

Choose one thing over two

You've been using A,
But now B is better than A

Still always use A.

Leave A where it is.
From now on, always use B

Enforce the chosen way; Ban the other ones

- README.md, STYLEGUIDE.md
- ESLint rules

Iterate, not reinvent

- Keep backwards compatibility if possible
  - Unless you can reinvent everything in one commit, while not breaking anything, and not blocking anybody (P.S. This never happens in bigger projects)

Always be objective

- Give specific reasons why A is better than B
- Use facts, not emotions

Always be objective

- Bad Senior forces a solution
- Good Senior explains why the solution is better

Help yourself and the others be objective

- Understand the other side
- When hearing emotions, opinions, ask for their root cause
- Ask others for help
  - Teammates
  - Other teams
  - Community

Collaborate with others

- Raise a thought/question whenever you’re considering A/B
  - Slack, GH Issue
- Communicate **why** you chose what you chose

Why so brutal?

Don’t be selfish

1. It’s about the **long-term**
2. It’s about the **project**
3. It’s about the **team**

Don’t avoid the hypes

but know why you’re taking them

Always be objective

1. Leave emotions aside
2. Speak facts
3. Help others be objective

Good ~~Intuition~~
Decision-Making Process = Good Engineer

Related links

- “Coding isn’t art, it’s engineering” - Jacek Tomaszewski
- “Hype Driven Development” - Marek Kirejczyk
  - Also related: review by David Cassel
- “Questions to ask when adding new tech to your infrastructure” - Bensan Gorge
- Example of a hype that isn’t actually that often needed: “Should you be using Web Workers? (hint: probably not) - David Gilbertson
- Example of extracting out a framework to just a tool: “Using Micro-Frontends to Permantenly Solve the Legacy JavaScript Problem” - Beamery

Thanks! Questions?

Jacek Tomaszewski
Full-stack Web Developer for over 15 years
jtom.me / jtompl / jtomaszewski