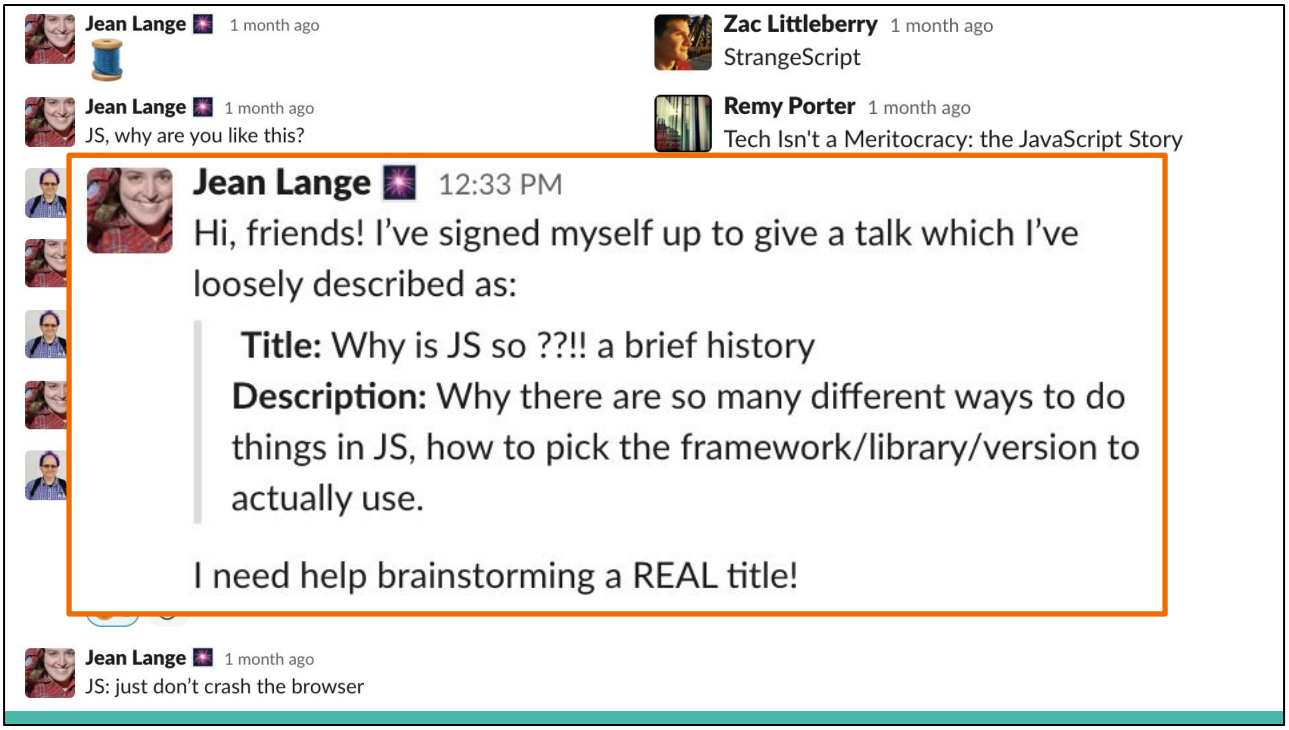
What the Heck-mascript?!

Jean Lange, PGH TechFest 2021

I promise you that the name of this talk is a very good joke and you will soon understand it!



Dan Oswalt brainstormed the name of this talk on Code & Supply chat - thank you, Dan, and I strongly recommend this method of titling your talks.

I'm Jean, here are some things I like!
These particular sticky notes are by Miro, which has been my go to sticky replacement in this remote time. Thank you, Miro.

You may be asking yourself - why is Jean, who likes Ruby, giving a talk that I thought was about JavaScript?
You're right! It's about JS!
The reason is that I know why Ruby is like it is. MINASWAN and all. Very into it. But I teach at Academy Pittsburgh, a local software development bootcamp (hey, we've got places open in our next session, that starts in September, and we've got really excellent grads for you to hire, hit me up!), and when we get to halfway through the 12-week course, having taught our students about C# and Ruby, and they start learning JS, they all say: WHY??!!! Why is it this way and causing me so much pain? Why is it so different from any other programming language?

And I wanted to find out.

Look, here's the story

1. JavaScript is a screwy language and has a screwy ecosystem.
2. But it is used everywhere! By everyone!
3. Let's take a look at its history.
4. And talk about how we can make good choices about JS in this world we find ourselves in now.

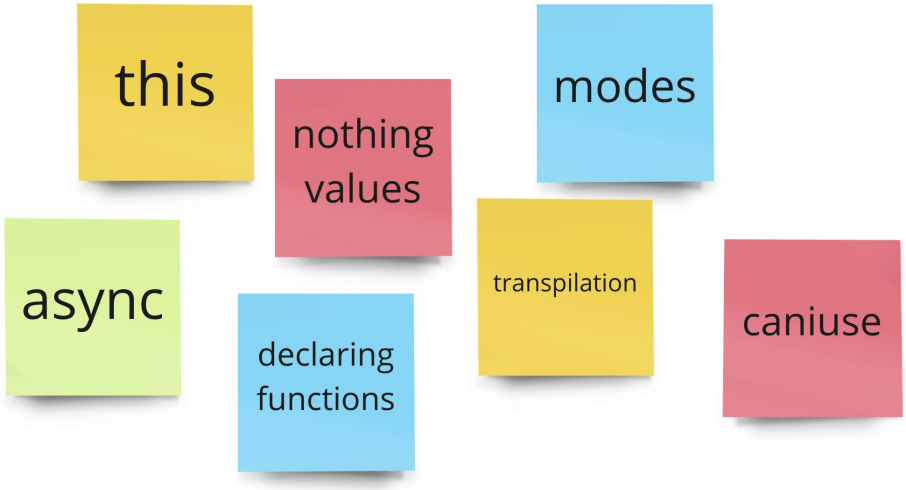
And here's the story that I found out about, that I'm going to present to you today.

Ok, let's get started!

JS is screwy

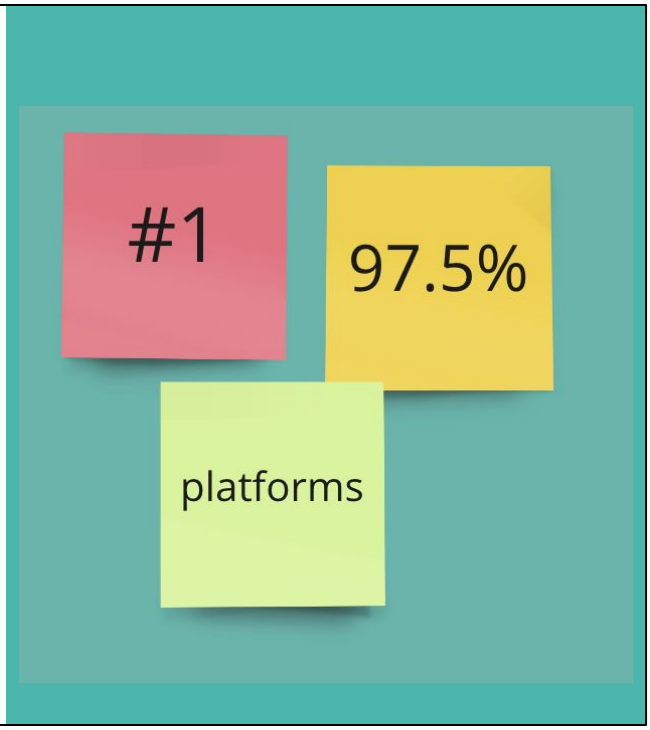
- What is JS all about?
  - Seems straightforward (another language, but for the web!), but there turn out to be all sorts of… quirks. Things we have to explain. Things we had to learn! Maybe you already know that JS is screwy, but in case you have been living immersed in it for too long, here's a refresher. I'd love to hear your favorite JS quirks in chat if you'd like to share those!
- Name
  - Java vs JS
  - ES vs JS. ES2015 = ES6.
- WAT - https://www.destroyallsoftware.com/talks/wat 1:20 - 2:42, especially
- Frameworks/package managers:
  - Let's play a game - how many frameworks can you think of for Ruby, C#, Python? What about JS?
  - What about package managers? Where do you go to get your libraries for Ruby? C#? Python? What about JS?

“some technical details”

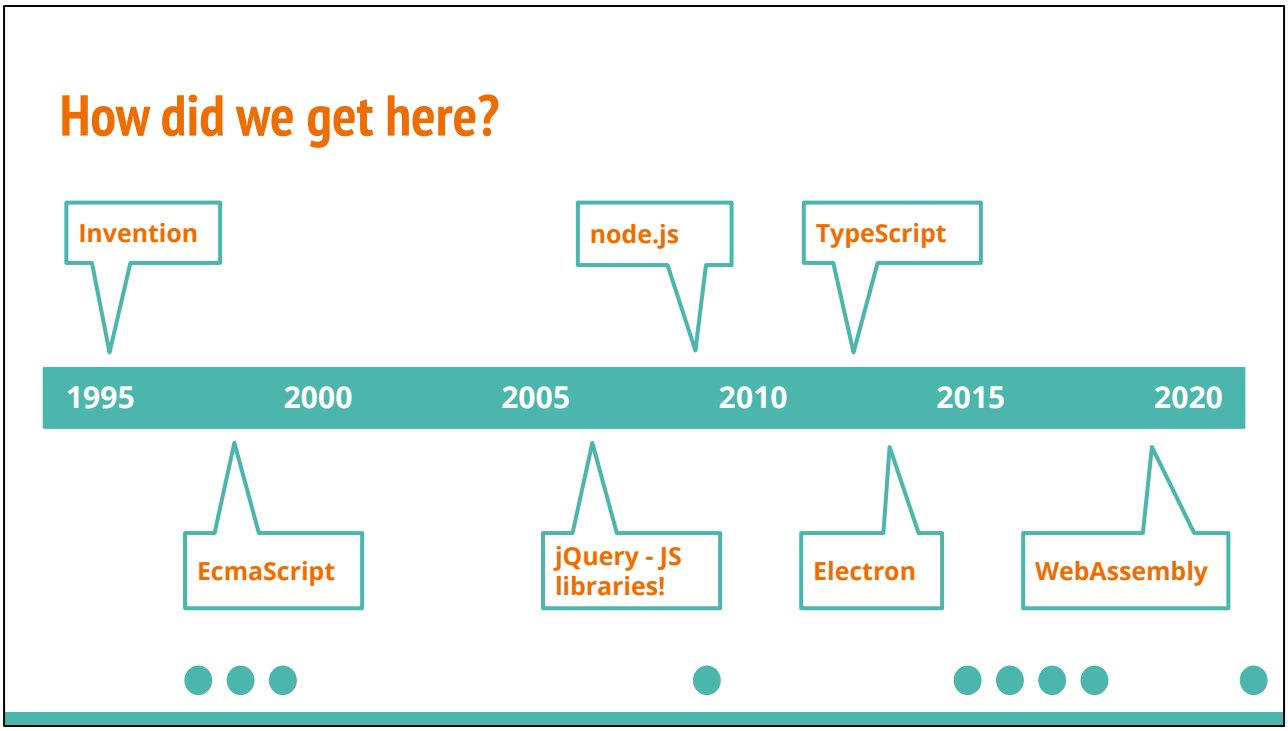


Some things that confuse our students when they get to JS - things that are different from what they expect!
Besides things that are different because it’s in a browser! Like the DOM & BOM & events-based programming...

BUT
JS is everywhere



- #1 most common programming language since 2012 (SO survey. This year's: https://insights.stackoverflow.com/survey/2021#most-popular-technologies-language)
- In use by 97.5% of website (https://w3techs.com/technologies/details/cp-javascript)
- Available not only on the web, where we think of JS living, but also
  - As a server-side language via node.js
  - For making desktop applications with frameworks like Electron
  - For making mobile applications with frameworks like React Native and Ionic



- Web invented in 1989 - static HTML, URIs.
- Invention (1995)
  - Brandon Eich at Netscape. “Easier Java”. 10 days. Mocha -> LiveScript -> JavaScript (marketing team)
  - You know how the <script> tag has that ‘lang’ attribute? JS was intended to be one of many! But it took over.
- EcmaScript (1997)
  - used to be European Computer Manufacturer’s Association - standards org.
  - The ES standard is also recognized by ISO under a different number, but is the same specification. Interestingly, the standard is NOT controlled by the W3C because they didn’t want to get into whole languages. https://www.quora.com/Why-was-JavaScript-standardized-by-ECMA-and-not-W3C
  - Ecma also responsible for (for example): language specs (C++, C#, Eiffel), file systems (FAT16/32, CD-ROM) and character sets.
  - Name of standard is because of copyright/trademark issues with the name JS.
  - JS has more than ES does - ES is a language specification, and JS adds DOM (Document Object Model - document.getElementById, eg) + BOM (Browser Object Model - location, history, back, eg)
  - There are other implementations of ES, like JScript (Microsoft) and ActionScript (Flash), but they are not nearly as popular.

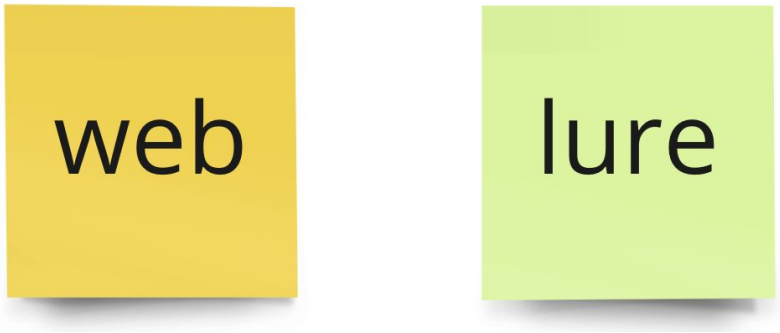
    - An illuminating SO question: https://stackoverflow.com/questions/912479/what-is-the-difference-between-javascript-and-ecmascript
    - Since its release in 1997, it's gone through periods of rapid and less rapid change! See https://www.w3schools.com/js/js_versions.asp. 1999 - 2009 for 3-5, then 2015 - 2018. ES2021 in June 2021
- jQuery (2006)
    - "vanilla"
    - First big megacollection of JS functionalities, unify functions that work on all browsers.
    - still used by 78% of websites (August 2021): https://w3techs.com/technologies/overview/javascript_library
    - This site also reports on other JS libraries, lots of familiar names in there! Includes:
        - graphical libraries like Bootstrap or Highcharts
        - Functionality libraries like lodash
        - Frameworks like React, Vue, and Angular
- Node.js (2009)
    - From OpenJs Foundation. Moves JavaScript to not-a-browser (using a browser engine, though!). JS runtime env, like ruby
    - Npm (ok, and yarn).
    - Express.js
    - Dev environment -> static, deployable assets.
- TypeScript (2012)
    - From Microsoft - another (early) implementation of ES2015 standard!
    - Adds optional typing. Strict superset of JS. Transpiles to JS.
    - Transcompilation
- Electron (2013)
    - Brought web tech (including JS) to desktop applications
- WebAssembly (2019)
    - W3C standard alongside HTML & CSS
    - New (assembly/binary) language for the web; can be written into from any language (in theory), major browsers support it with flags.
    - Focus on performance. Has some immaturities.

Summary trends to think about/notice:

- Proliferation of libraries
- SPAs (2012sish - squishy number)

Unless otherwise noted, info comes from Wikipedia pages.

Reframing questions about why JS is so ?!#*&!



These are my opinions!

- Web
  - Grew with it
  - DON’T BREAK THINGS
  - Falls into that singular tech bucket like HTML & CSS
- Lure
  - Attract non-devs
  - Gradually normal engineering practices surround it

How do I make choices?

- JS IS a reasonable - and pretty safe - choice.
- Same as any other software engineering choice!

- ilities
  - popularity/learnability/support/hireability/retainability/longevity - github stars
  - compatibility
- fit to your application - was it built to solve the kind of problems you're having?
  - react
  - angular - corporate style SPAs by google - like gmail
    - angular vs angular 2+ (copy the popularity of react)
  - Vue - lighter weight, easier to get started in. not supported by any major company.
  - svelte - goddammit, redux, but oooh the display of react. better datahandling react.
- New shiny?

Questions!

jean@academypgh.com
github.com/jeanlange/slides