Hier staan de logo's van een aantal frameworks en technieken. Welke kennen we allemaal…?

Het fenomeen 'ecosysteem' is momenteel erg in het nieuws: het ecosysteem namelijk van de Oostvaardersplassen. In hoeverre kan de omgeving het aantal individuen en het aantal soorten herbergen…? Net zoals in de Oostvaardersplassen is de vraag hoeveel JS 'het systeem' aankan.


Oostvaarders-ophef hoeft geen traditie te worden

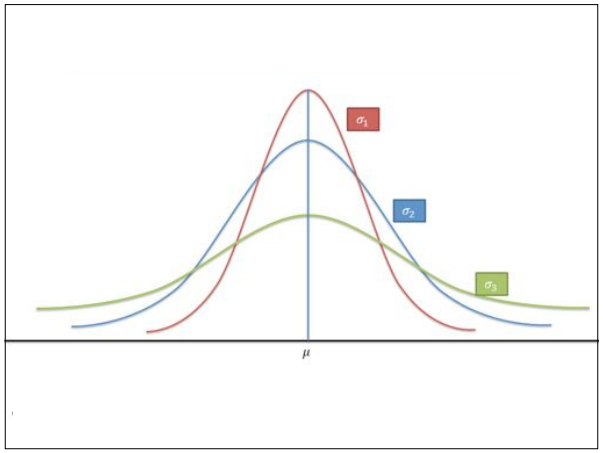
Trouw, 3 maart 2018

Als de omgeving niet het aantal individuen kan handhaven, als de biodiversiteit niet meer gezond is, dan kunnen we daar op verschillende manieren op ingrijpen. In hoeverre vertaalt zich dit naar het JS-ecosysteem?

---

1. Introductie

De ontwikkeling van een ecosysteem.
We zien dat een ecosysteem altijd een samenwerking is tussen de flora, de fauna en wat de omgeving kan dragen. Wat een gezonde diversiteit is, is hiervan afhankelijk.



Op de horizontaal de soorten, op de verticaal het aantal. Dat geldt voor een hele hoop zaken, zoals boeken, films, of diersoorten. De meeste diersoorten worden met uitsterven bedreigd, maar er zijn honderden miljoenen kippen.

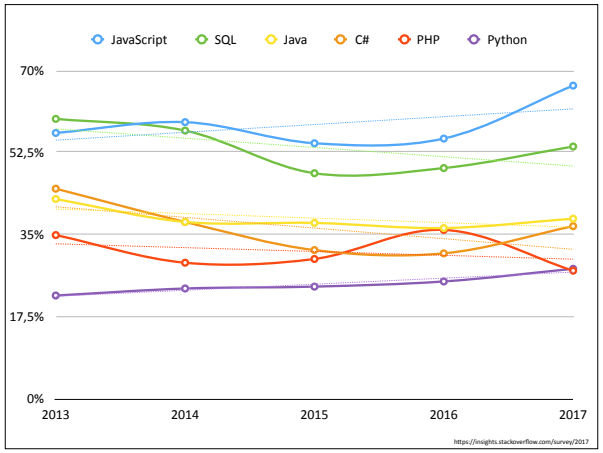
> Evolutie werkt door kleine variaties toe te staan en eventuele verbeteringen te belonen. Vaak zijn dat kleine stapjes, soms zijn het grote. In het berglandschap van het leven bevinden deze innovaties zich altijd in de voetheuvels, soms in de verste uithoeken. Een molshoop van nu kan uitgroeien tot het bergmassief van morgen. Zeldzame gebeurtenissen, ideeën en uitvindingen spelen een grotere rol in de geschiedenis dan we denken.

https://www.dub.uu.nl/sites/default/files/legacy/other/dijkgraaf.pdf

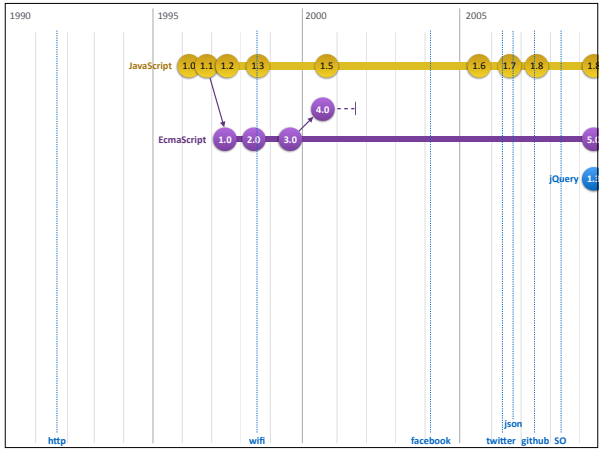
Een fraai citaat in dezen komt van Robbert Dijkgraaf. Dit idee kan als drijvende kracht achter dit praatje beschouwd worden.

---

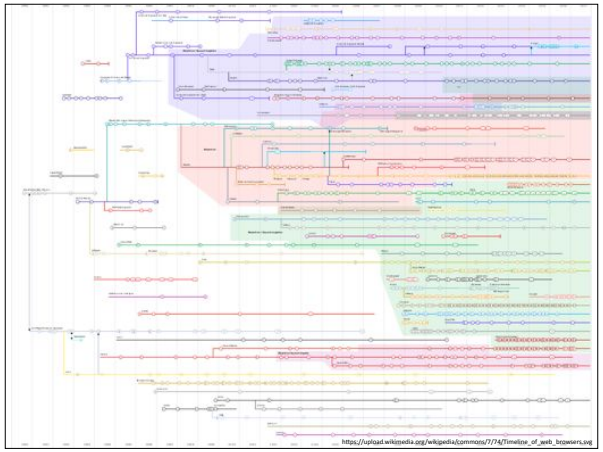
2. Het javascript ecosysteem



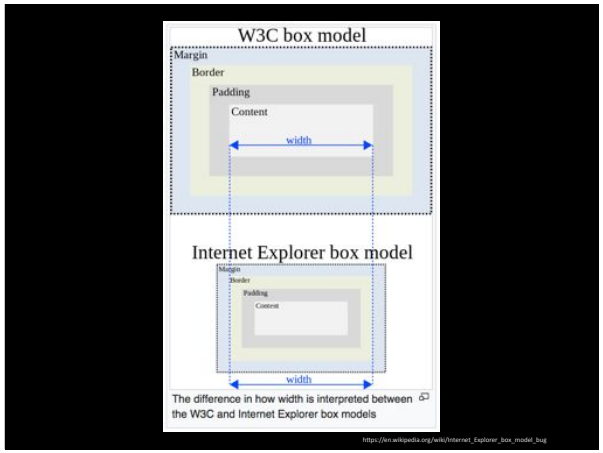
We zien dat met name de laatste paar jaar JavaScript zich op een toenemende belangstelling mag verheugen. Binnen welke omgeving is dit dan tot ontwikkeling gekomen? Laten we die omgeving eens proberen te schetsen.



Dit is de omgeving waarin JS opkwam (het is in tien dagen geschreven door Brendan Eich (Netscape)). Behoorlijk rustig, en opvallend is dat veel van de sites en technieken (json, wifi, facebook, twitter, stackoverflow) die we dagelijks gebruiken in deze periode zijn opgekomen.

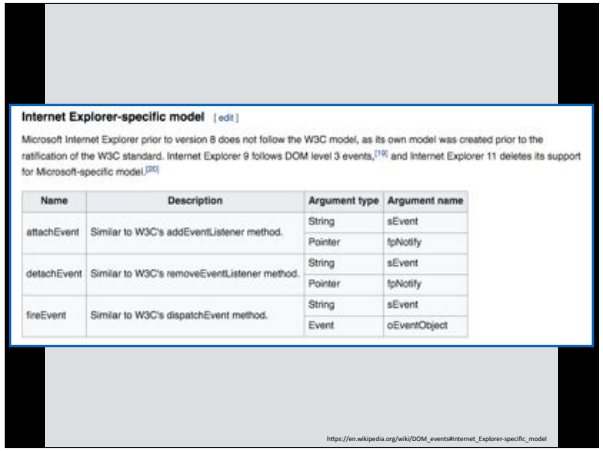


Dit was de tijd van de *browser wars*: we moesten toen veel moeite doen om onze code op alle mogelijke platformen en in alle mogelijke browsers goed te laten werken.



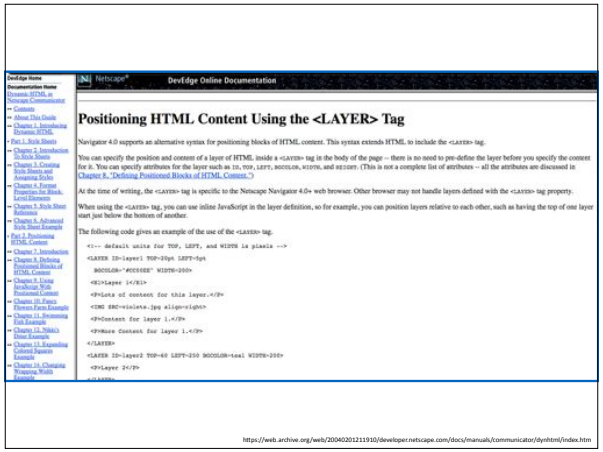
Wie herinnert zich bijvoorbeeld nog de eigen implementatie van het Box Model door Internet explorer?

(https://en.wikipedia.org/wiki/Internet_Explorer_box_model_bug)



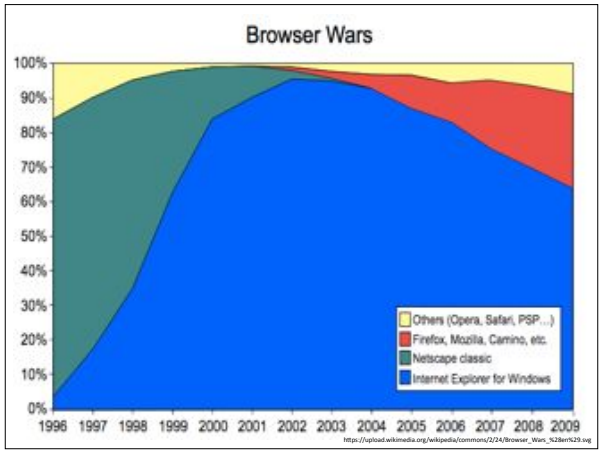
Of het eigen *event model* van Internet Explorer?

(https://en.wikipedia.org/wiki/DOM_events#Internet_Explorer-specific_model)



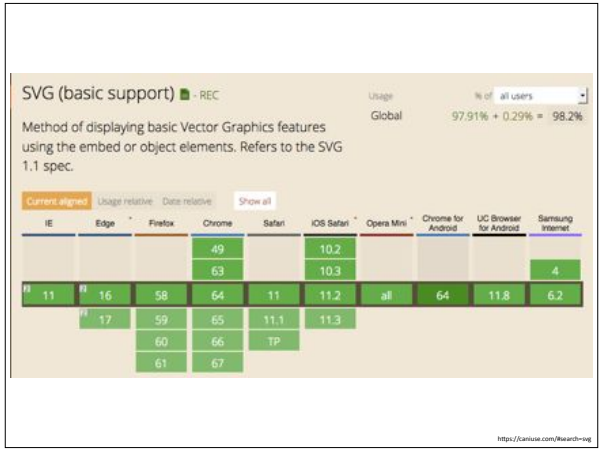
Of de *Layer tag* van *Netscape Navigator*? Dit waren allemaal specifieke dingen waar we als ontwikkelaar rekening mee moesten houden.

(https://web.archive.org/web/20040201211910/developer.netscape.com/docs/manuals/communicator/dynhtml/index.htm)



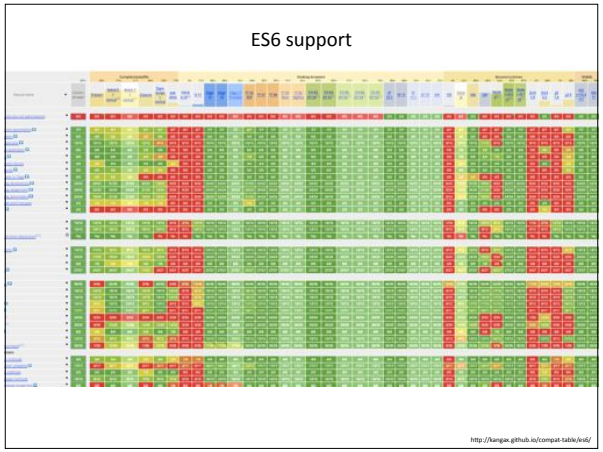
Dus in die tijd was er ook behoefte aan frameworks. Die frameworks abstraheerden (en abstraheren) al die ellende voor je weg.

(https://upload.wikimedia.org/wikipedia/commons/2/24/Browser_Wars_%28en%29.svg)



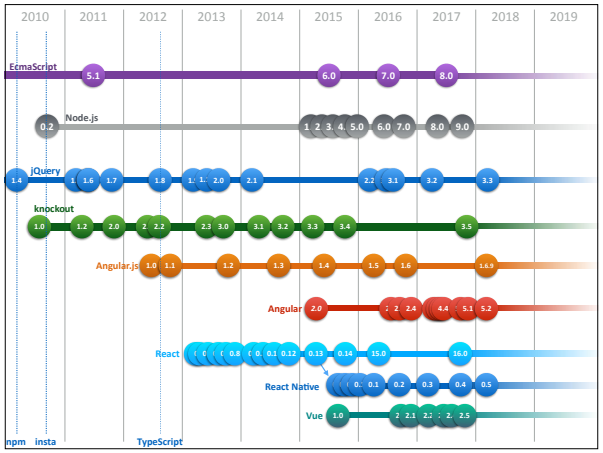
Maar inmiddels is die ellende ook wel een beetje uitgekristalliseerd en werken de meeste browsers gewoon goed samen en ondersteunen ze allemaal dezelfde technieken. Zoals hier het support voor svg… allemaal groen.

(https://caniuse.com/#search=svg)



Hetzelfde geldt voor ES-6 support. Deze dia is niet leesbaar, maar het gaat om de overeenstemming in rood en groen, en om de hoeveelheid groen.

(http://kangax.github.io/compat-table/es6/)



Maar dat heeft geen positieve ontwikkeling gehad op het aantal frameworks. Hoewel de omgeving een stuk rustiger is dan in de vijf, tien jaar daarvoor, is de wereld van de frameworks wat onrustiger aan het worden.

---

and there's more...

AMD, ActionHero, AdonisJS, Alt, Axios, Babel, Babelify, Backbone, Bluebird, Bower, Broccoli, Browserify, Clojure, CoffeeLint, CommonJS, Derby, ES2016+, Express, FeatherJS, Fetch, Flow, Flummox, Fluxible, Grunt, Gulp, Hapi, ImmutableJS, JFX, JSCS, JSDOM, JSHint, Jest, Koa, LoopBack, Loopback, Mean.js, Meteor, Mimosa, Mojito, Mongoose, Nightwatch, OCaml, PhantomJS, Plymer, Prettier, Ramda, Redux, Request, RxJS, Sails, Seneca, Socket.io, StandardJS, Stylelint, SystemJS, Total.js, Transpiling, TypeScript, UglifyJS, UnderscoreJS, Watchify, WebPack, XO, Zepto, asyn, es2015, eslint, lodash, moment, preact-cli, recompose, tslint,Aurelia,Keystone,

En er zijn nog wel wat meer technieken en frameworks en gists… Toegegeven, dit zit niet allemaal op hetzelfde niveau in de stack, maar het idee is wel duidelijk.

```
> cat package.json

{
  "name": "foto-app",
  "version": "1.0.0",
  "description": "Eerste versie van de foto'app",
  "main": "index.js",
  "scripts": {
    "debug": "watchify src/index.js -o www/bundle.js -t [ babelify --presets [ es2015 react stage-2 ] ] -v",
    "build": "browserify src/index.js -o www/bundle.js -t [ babelify --presets [ es2015 react stage-2 ] ]"
  },
  "keywords": [],
  "author": "Bart Barnard",
  "dependencies": {
    "react": "^16.2.0",
    "react-dom": "^16.2.0",
    "react-redux": "^5.0.6",
    "redux": "^3.7.2",
    "redux-thunk": "^2.2.0"
  },
  "devDependencies": {
    "babel": "^6.23.0",
    "babel-core": "^6.26.0",
    "babel-preset-es2015": "^6.24.1",
    "babel-preset-es2016": "^6.24.1",
    "babel-preset-react": "^6.24.1",
    "babel-preset-stage-2": "^6.24.1",
    "babelify": "^8.0.0",
    "browserify": "^15.1.0",
    "watchify": "^3.9.0"
  }
}
```

Dus ook voor een behoorlijk eenvoudig projectje moet je tegenwoordig al een relatief complexe infrastructuur opzetten, zoals blijkt uit dit package-bestand voor een behoorlijk eenvoudig ding.

I'm just going to move back to the backend. I just can't handle these many changes and versions and editions and compilers and transpilers. The JavaScript community is insane if it thinks anyone can keep up with this.

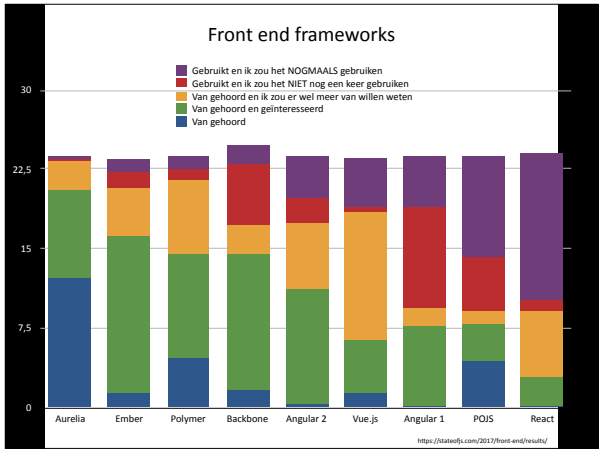
https://hackernoon.com/how-it-feels-to-learn-javascript-in-2016-d3a717dd577f

Wat een fraai citaat oplevert

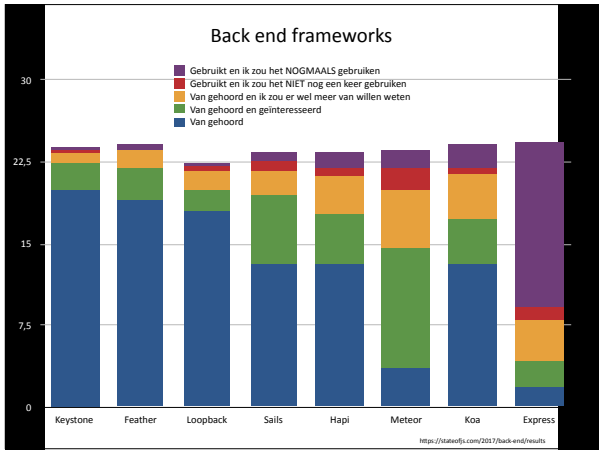
---

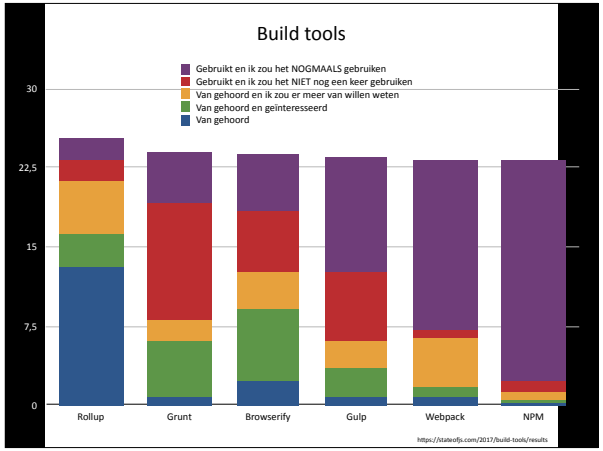
3. The state of JS

Indachtig de uitspraak van Dijkgraaf zou je dus kunnen denken dat er zich voldoende ontwikkeling aan de uiteinden afspeelt. Maar is dat wel echt zo? Polariseert het geheel ook hier niet naar het midden?



The state of javascript, vragenlijst ingevuld door meer dan 28000 respondenten van over de hele wereld.

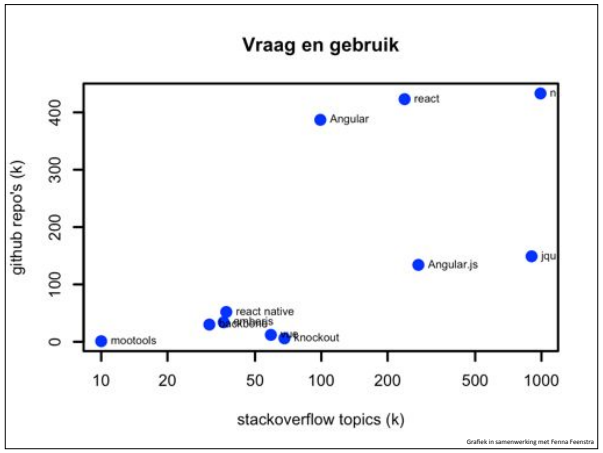




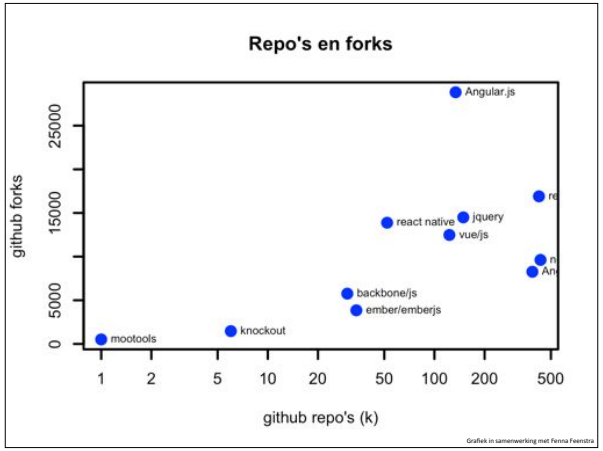
Deze dia is wat onduidelijk: de technieken die hierin genoemd worden zijn immers niet allemaal inwisselbaar…?

Framework	Anciënniteit	Releases (github)	Releases per jaar (μ)
Mootools	10	36	3,60
Backbone	7,5	30	4,00
Knockout	8	40	5,00
jquery	10	148	14,80
React	4,5	83	18,44
Angular.js	6	193	32,17
Emberjs	6	295	49,17
Nodejs	7,5	472	62,93
Angular	3	208	69,33
Vue	2,5	223	89,20
React native	2,5	253	101,20

Nog wel een aardige statistiek: als je de anciënniteit van de frameworks afzet tegenover het aantal releases op github en het resultaat daarop sorteert, zie je dat de hedendaagse frameworks het snelts een nieuwe release leveren.



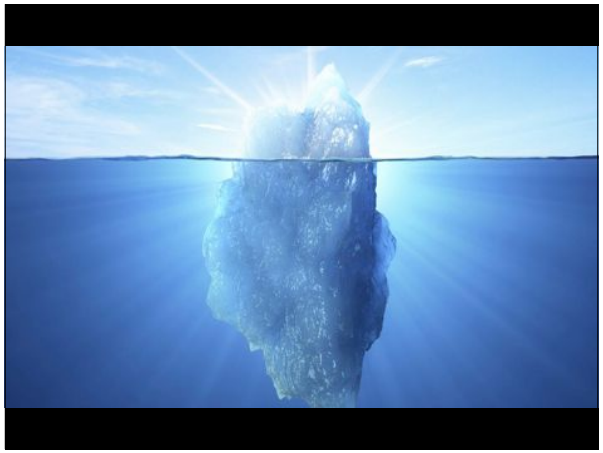
Of het aantal topics op stackoverflow plotten tegenover het aantal repo's op github, dat zegt ook wel iets (maar wat precies is natuurlijk punt van discussie). Merk op dat de horizontale as een logaritmische schaal heeft.



Een andere aardige statistiek is het aantal repo's tegenover het aantal forks op github.

---

4. De toekomst en de repercussies voor het onderwijs



JS Frameworks zijn op meerdere manieren een ijsberg: (1) je ziet maar een klein stukje, grootste deel zie je niet; (2) het abstraheert het grootste deel van de complexiteit; en (3) uiteindelijk smelten ze en verdwijnen ze.

```
Polyfills
Web Components
HTML templates
Custom Elements

Zero Framework Manifest
(demonstratie)
```

Terwijl de moderne browsers en de moderne DOM's ook van zichzelf al heel veel eigenschappen hebben die we kunnen gebruiken.

```
Async Functions
SharedMemory and Atomics
Object.values/Object.entries
String padding
Object.getOwnPropertyDescriptors()
Generators
Spread and Gathers
Destructuring
for..of
Modules
TypesArrays
WeakMaps
Asynchronous Iteration
...
```

En ook omdat veel methoden en technieken uit de frameworks inmiddels naar het reguliere ecmascript zijn doorgesijpeld; en er komt nog veel meer aan. Dus niet zomaar een framework gebruiken als het niet echt nodig is.

Geraadpleegde bronnen

Simpson, K (2015), *EcmaScript 6 & Beyond*. O'reilly

https://hacks.mozilla.org/2016/05/a-taste-of-javascripts-new-parallel-primitives/

https://github.com/tc39

http://kangax.github.io/

https://auth0.com/blog/a-brief-history-of-javascript/

https://upload.wikimedia.org/wikipedia/commons/7/74/Timeline_of_web_browsers.svg

https://en.wikipedia.org/wiki/JavaScript

https://www.telerik.com/campaigns/kendo-ui/wp-javascript-future-2018/thank-you

http://www.ecma-international.org/memento/TC39.htm