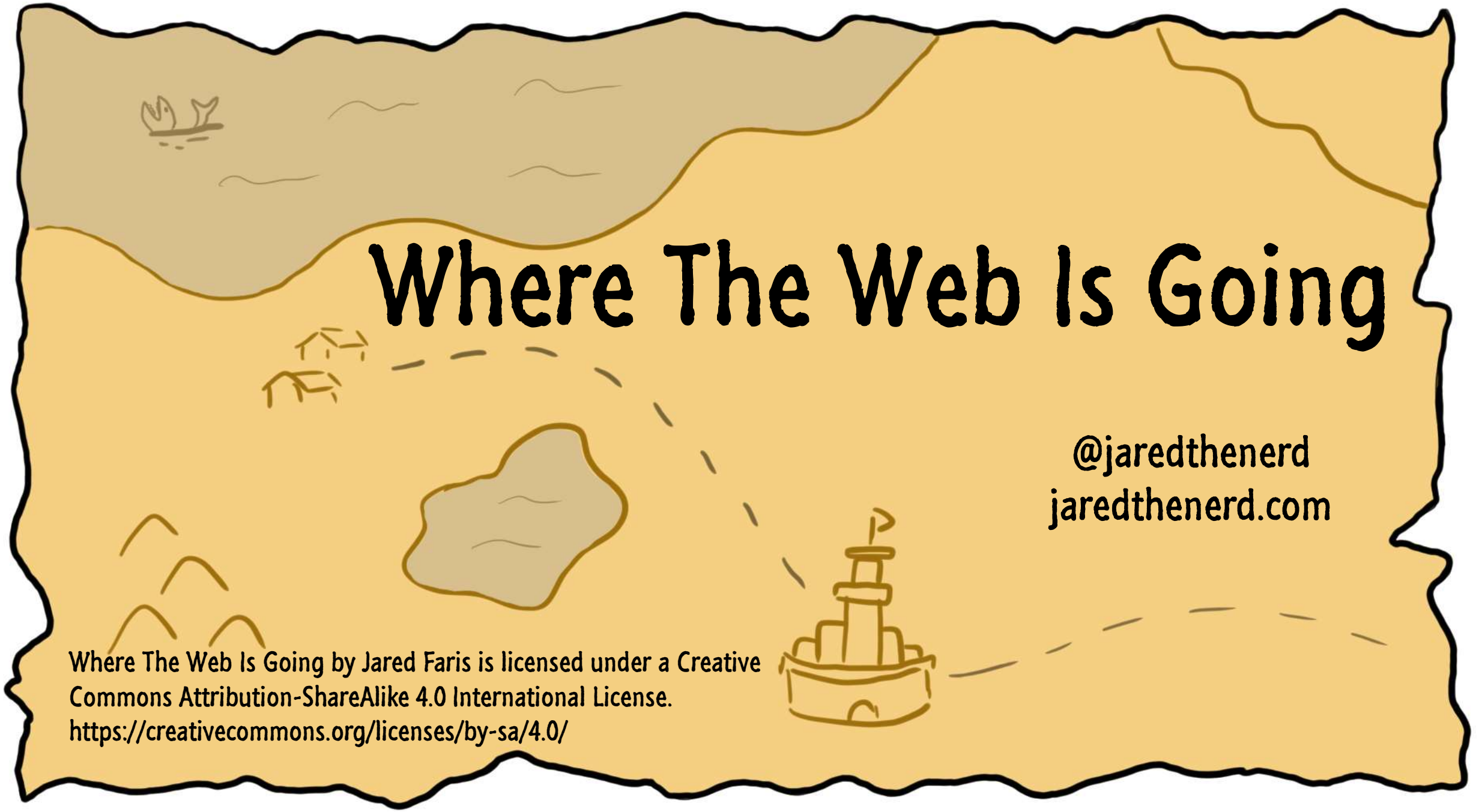
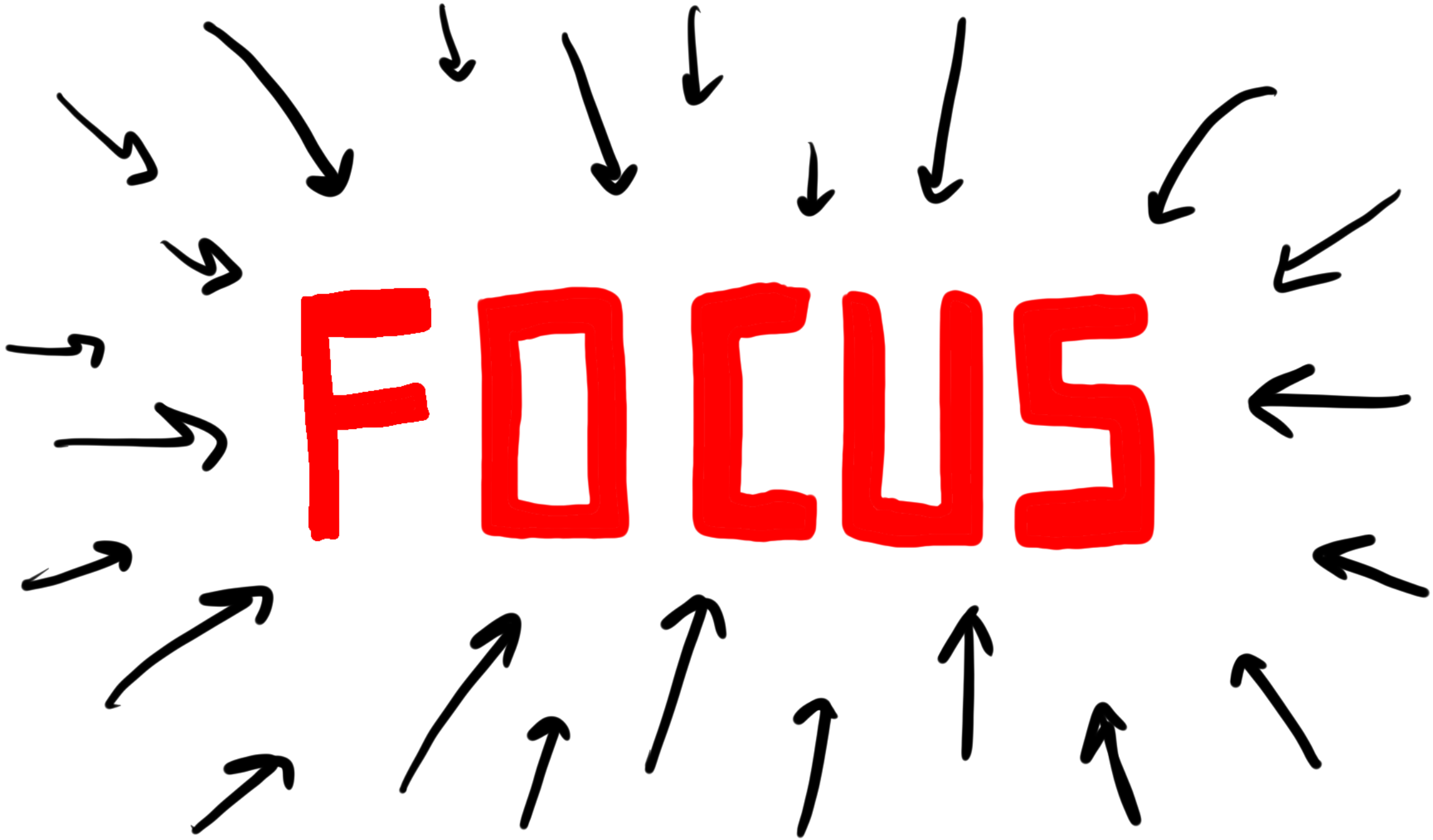
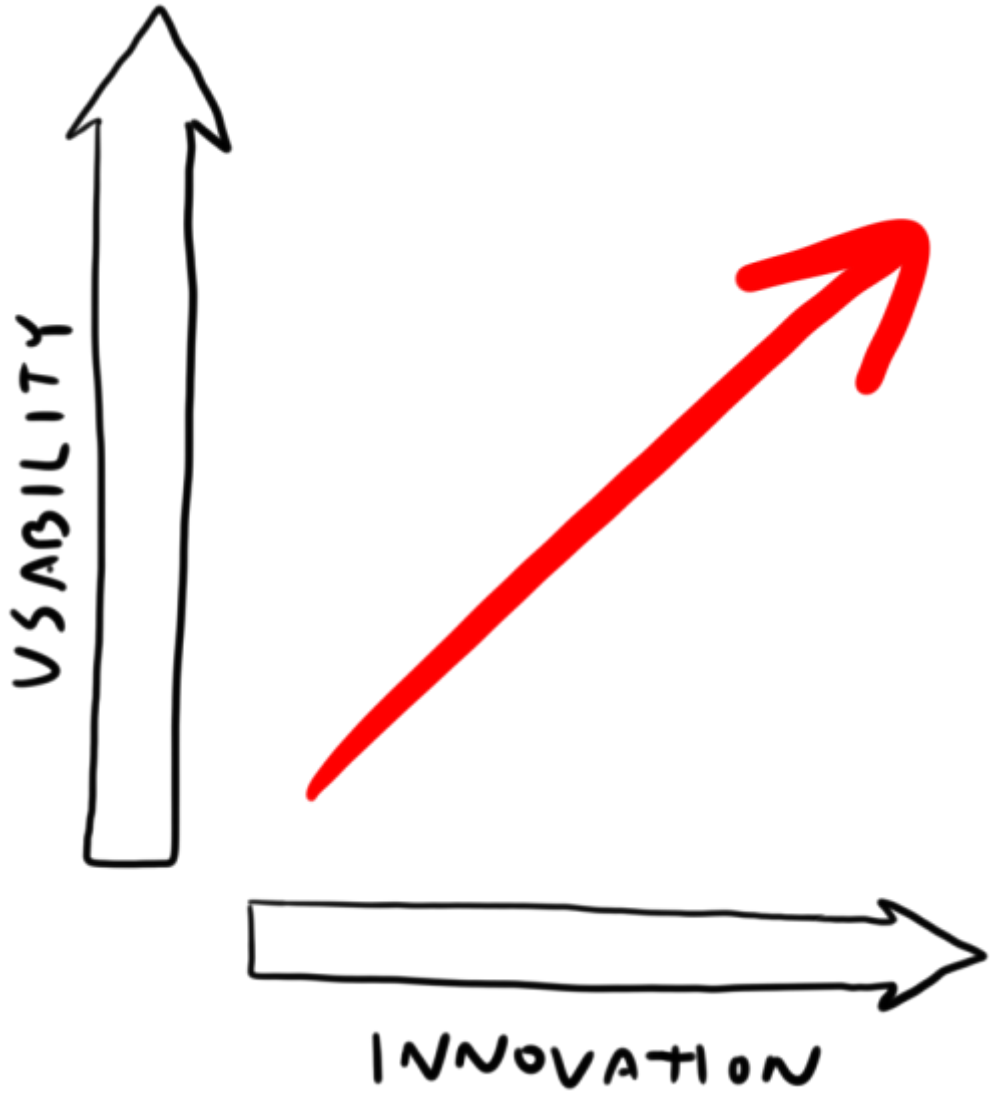
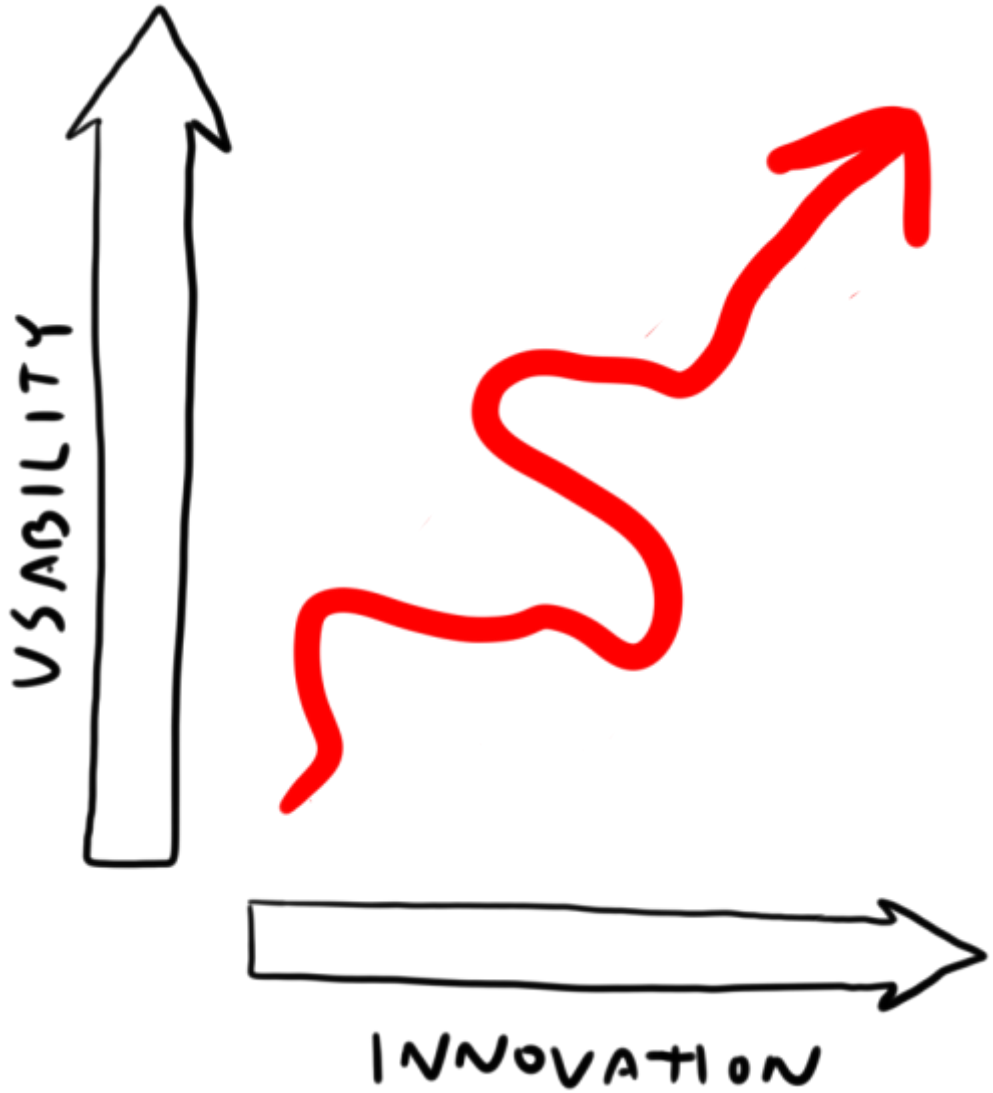


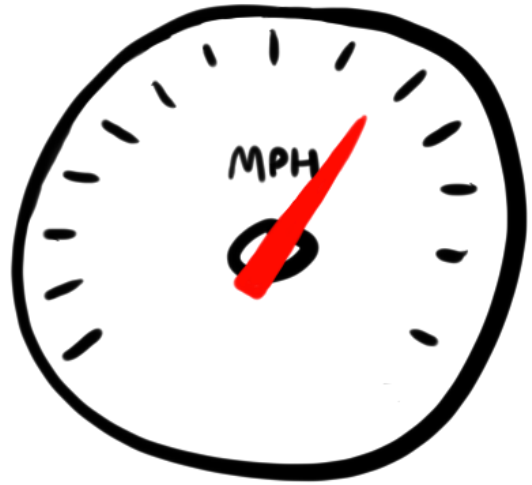
Where The Web Is Going by Jared Faris is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
https://creativecommons.org/licenses/by-sa/4.0/

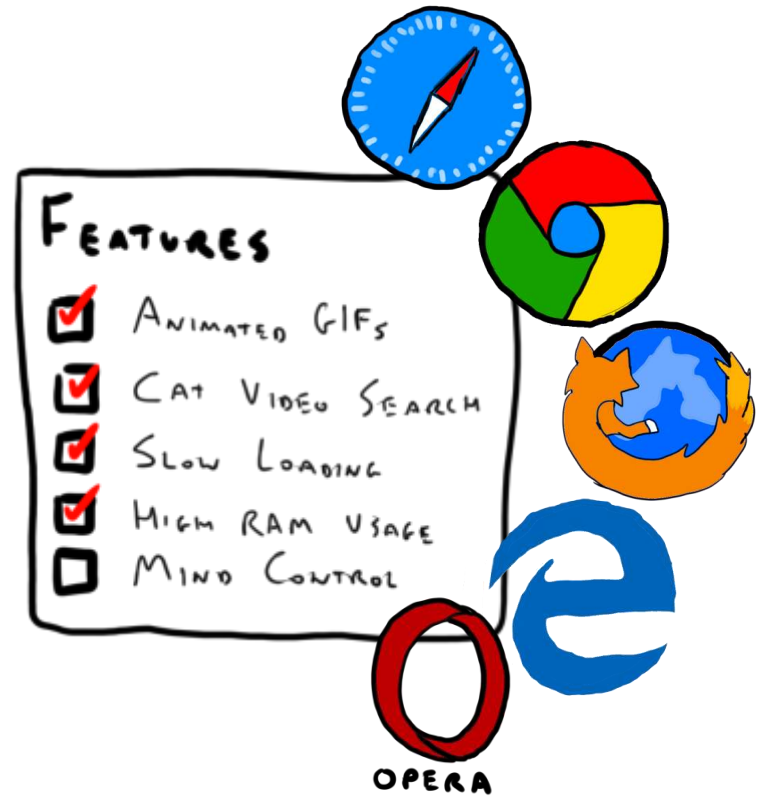

FOCUS


USABILITY
INNOVATION


USABILITY
INNOVATION

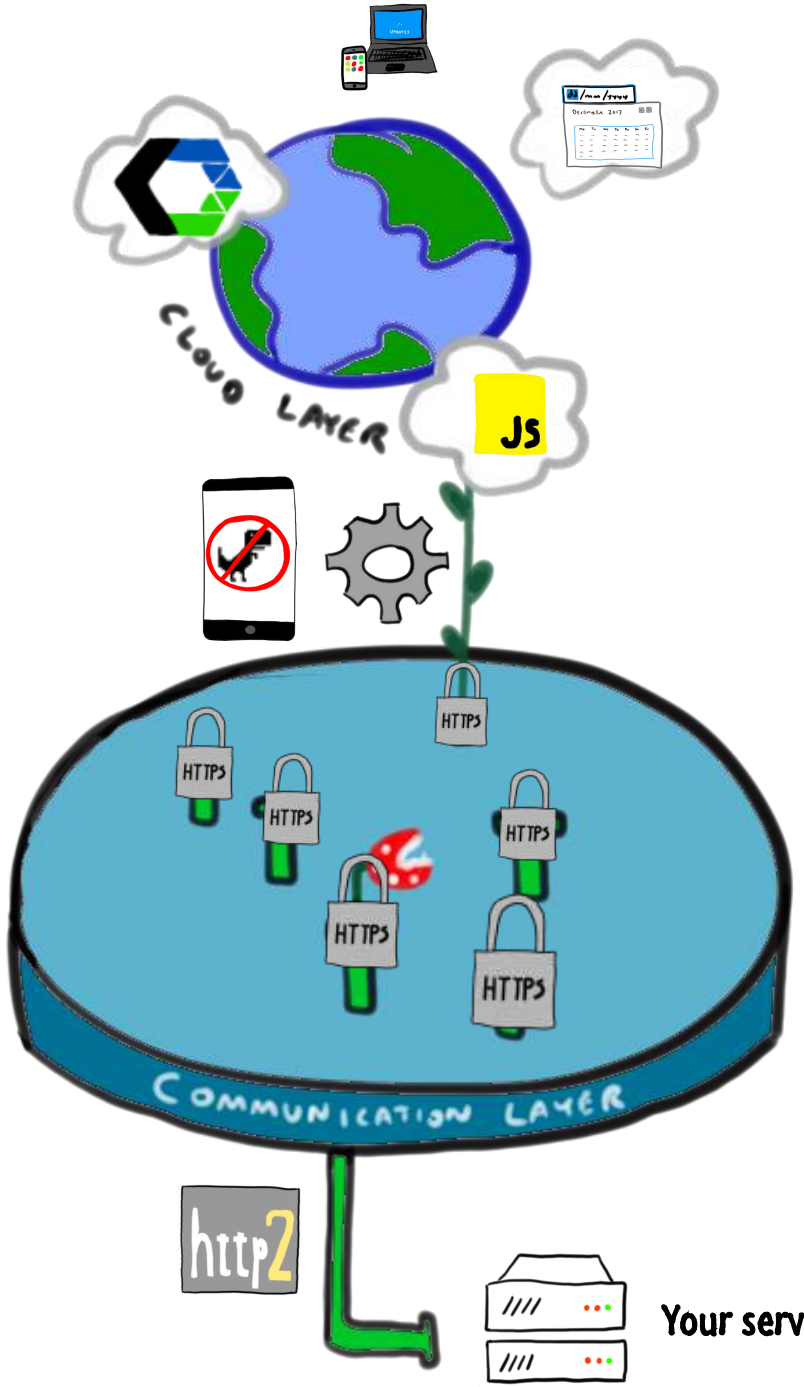
UPDATES


MPH


FEATURES
ANIMATED GIFS
CAT VIDEO SEARCH
SLOW LOADING
HIGH RAM USAGE
MIND CONTROL
OPERA

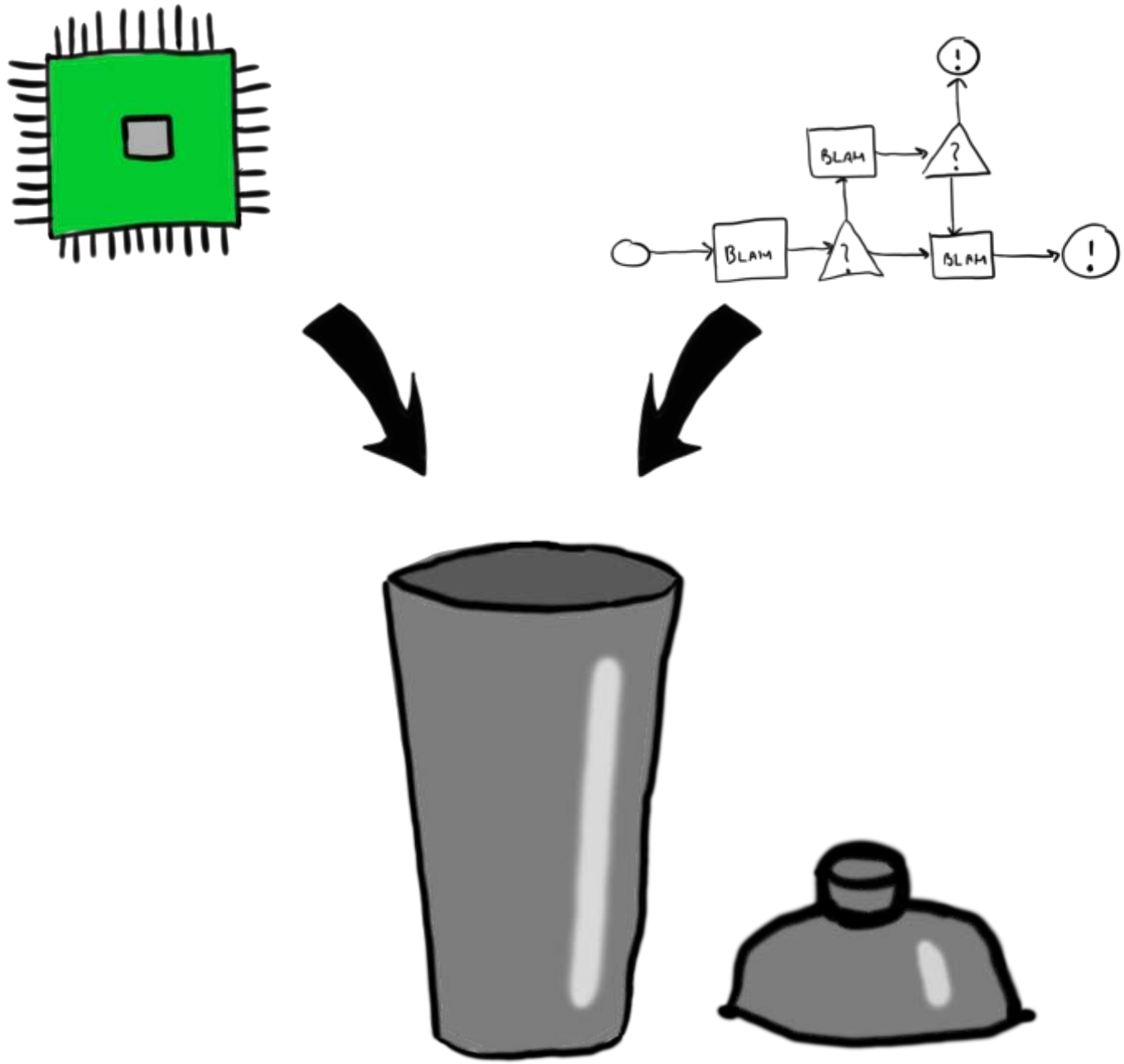
How the Web Literally* Works

and what we'll be talking about



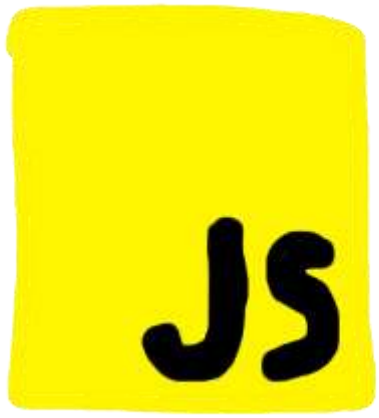
Your servers sit here

*figuratively


BLAM
?
!
BLAM
?
BLAM
!

HTML

CSS


JS

http2

W3C

ecma INTERNATIONAL

?

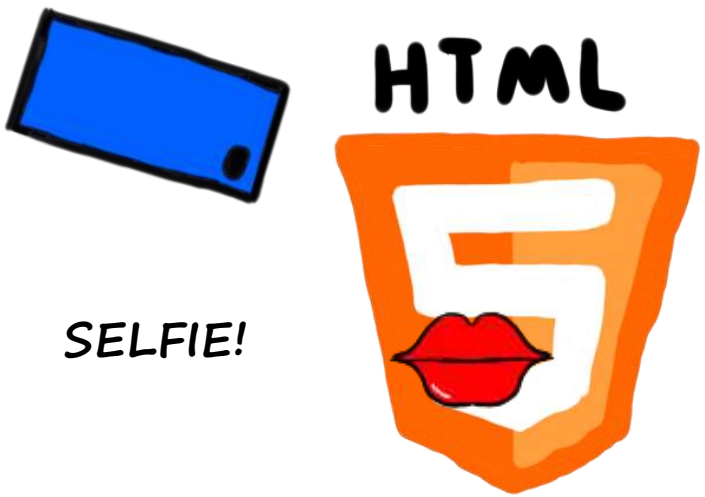
IETF

BBQLOL

W3C
HTML
5
?

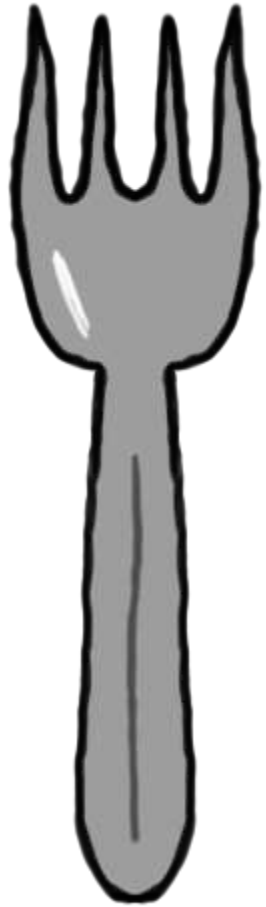
New Yorker – The Group That Rules The Web

http://bit.ly/RulesTheWeb

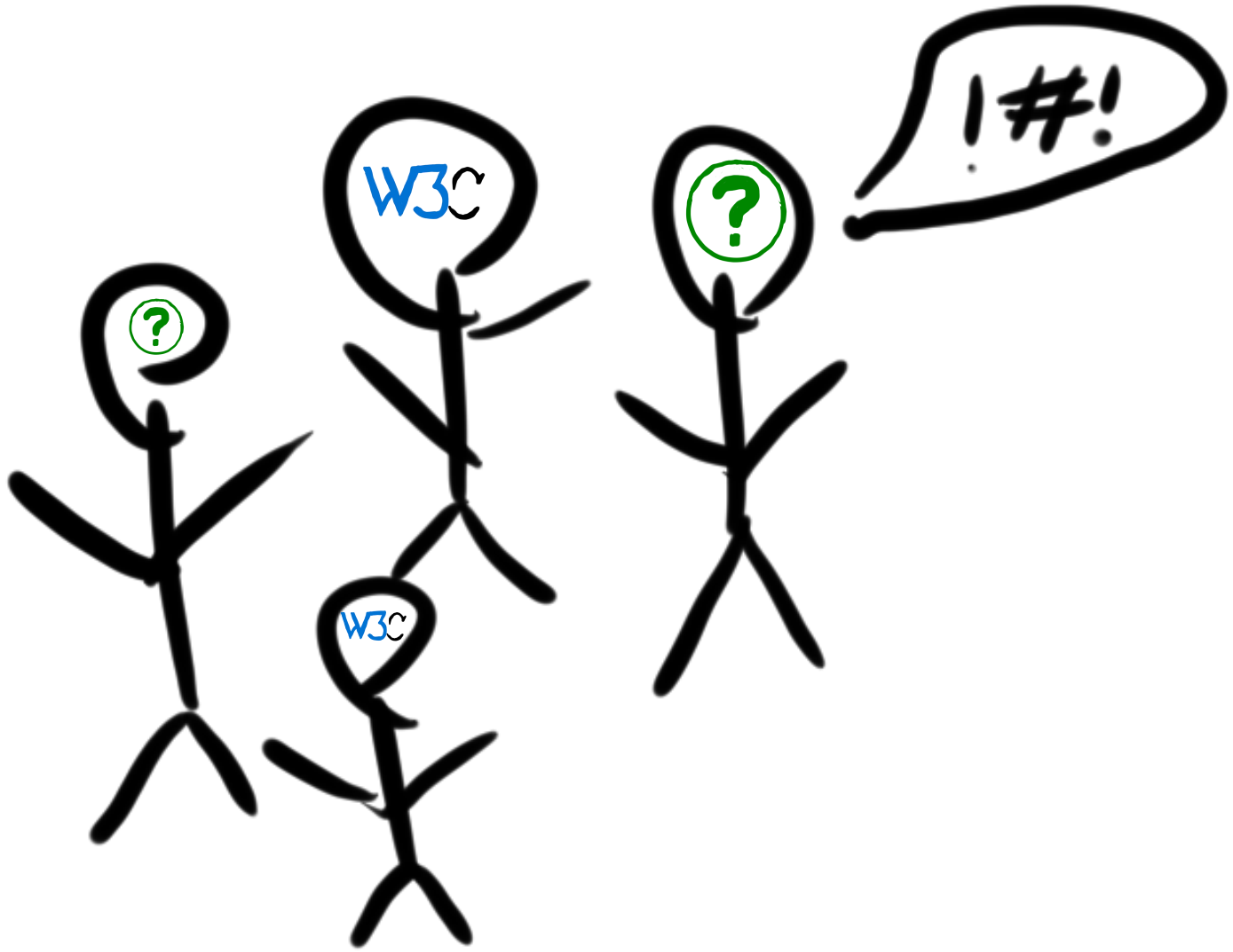


W3C

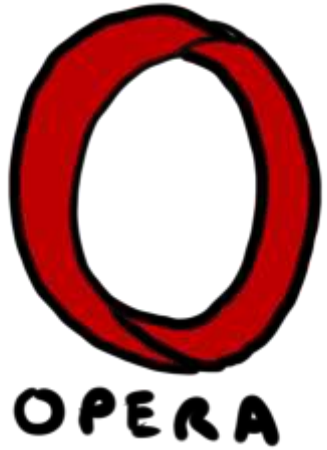
HTML



HTML


W3C
W3C
?
?
!#!

Mozilla
FOUNDATION


OPERA

"Consensus is not a useful value in Web spec development, because it doesn't accurately reflect the power dynamics at work.

In practice, implementers have the ultimate say, not spec editors, not authors, not users, and not interested bystander standardistas.

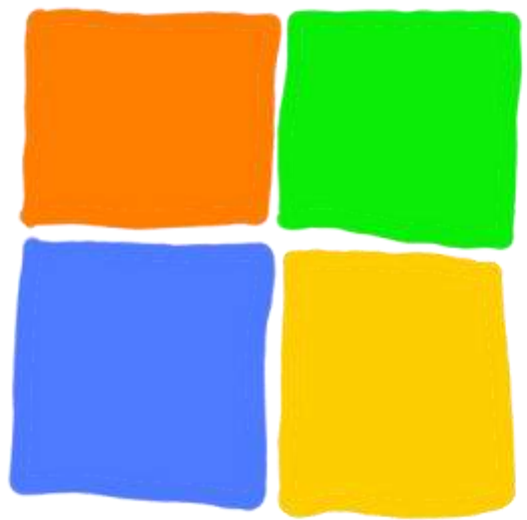
The W3C and the WHATWG thus have no power. (The WHATWG is built around that realization; the W3C, however, is built on the assumption that it does have power.

That's how it ends up making mistakes like RDF, XForms, or XHTML2.)"

Ian Hickson – April 23, 2014

W3C

<3 OPEN STAND



http://bit.ly/WHATWGIPR

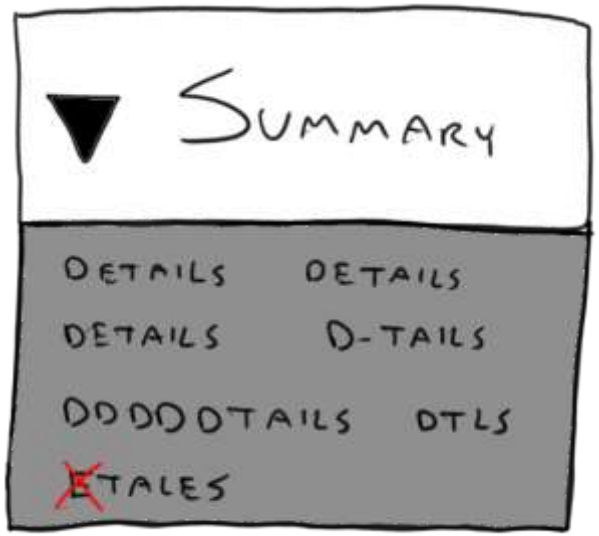
HTML 5.1

Form Validation & Forced Spellcheck

Date Pickers

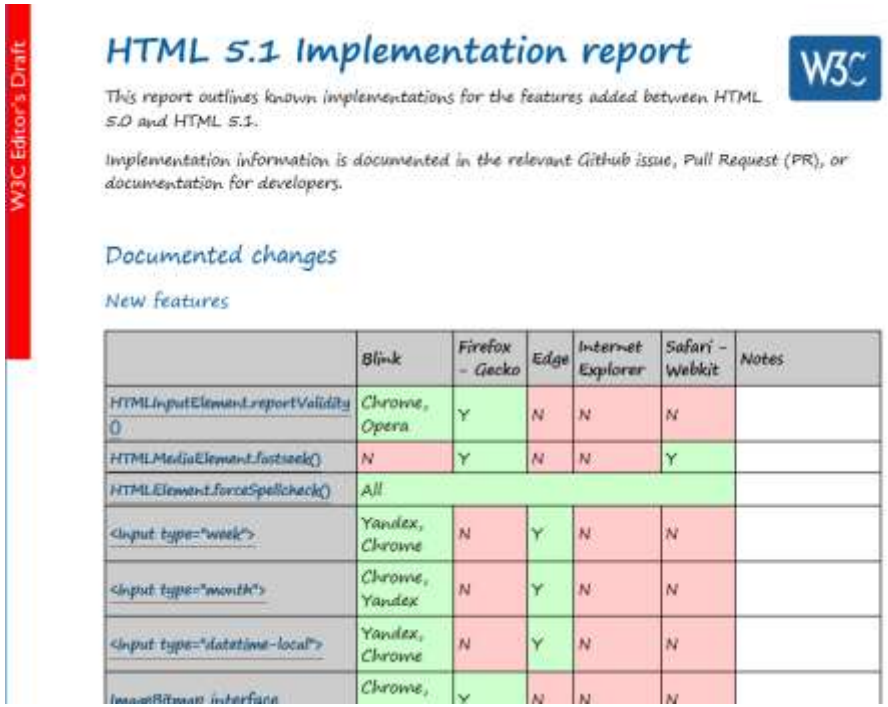
Context Menus

Expanding Detail Panels



.1

¯_(ツ)_/¯



W3C Editor's Draft

HTML 5.1 Implementation report

This report outlines known implementations for the features added between HTML 5.0 and HTML 5.1.

Implementation information is documented in the relevant Github issue, Pull Request (PR), or documentation for developers.

Documented changes

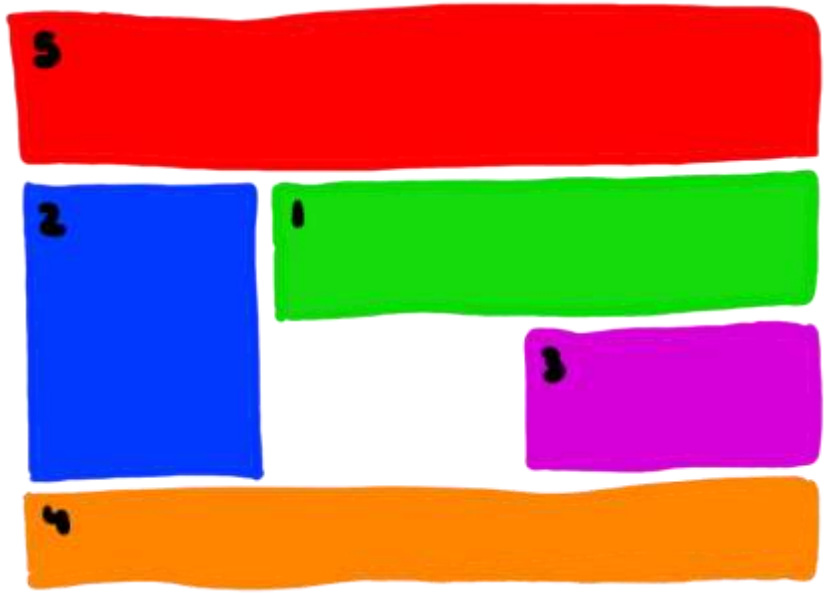
New features

	Blink	Firefox - Gecko	Edge	Internet Explorer	Safari - Webkit	Notes
HTMLInputElement.reportValidity()	Chrome, Opera	Y	N	N	N	
HTMLMediaElement.fastSeek()	N	Y	N	N	Y	
HTMLElement.forceSpellcheck()	All					
<input type="week">	Yandex, Chrome	N	Y	N	N	
<input type="month">	Chrome, Yandex	N	Y	N	N	
<input type="datetime-local">	Yandex, Chrome	N	Y	N	N	
ImageBitmap interface	Chrome,	Y	N	N	N	

http://bit.ly/HTML5Implementation

More Stuff

CSS Grid




```html
<div class="grid-container">
    <div class="item-1">1</div>
    <div class="item-2">2</div>
    <div class="item-3">3</div>
    <div class="item-4">4</div>
    <div class="item-5">5</div>
</div>
```

```css
#demo-two.example-2 .grid-container {
    display: grid;
    grid-template: repeat(4,1fr)/repeat(3,1fr);
    grid-gap: 20px;
    grid-template-areas:
        'header header header'
        'sidebar content content'
        'sidebar . content2'
        'footer footer footer';
}

#demo-two.example-2 .item-1 {
    grid-area: content;
}

#demo-two.example-2 .item-5 {
    grid-area: header;
}
```

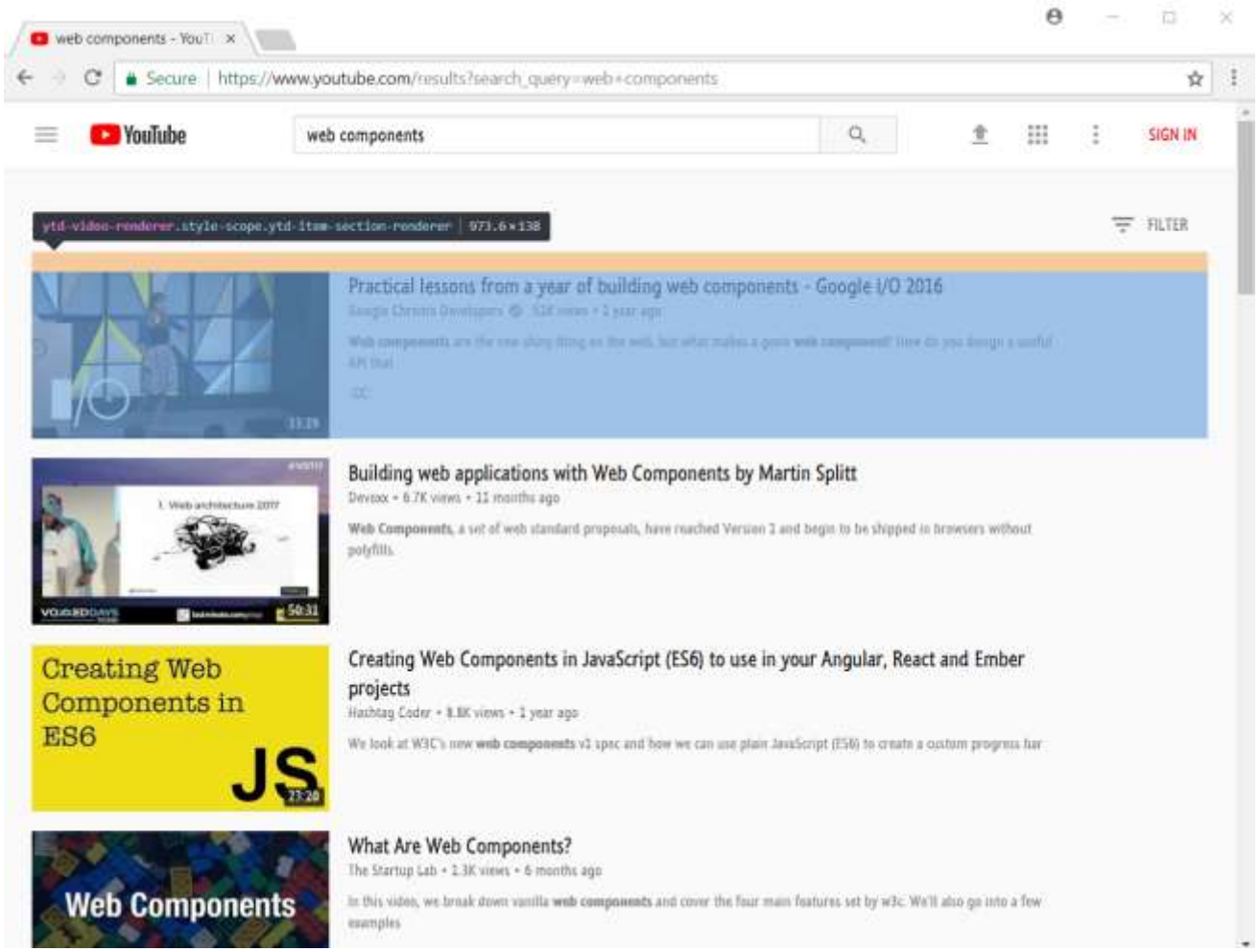
CSS Variables

```
:root {
    --my-color: red;
}

#someDiv {
    color: var(--my-color);
}
```

Web Components

- HTML Templates
- Shadow DOM
- Custom Elements
- HTML Imports

Web Components In Action

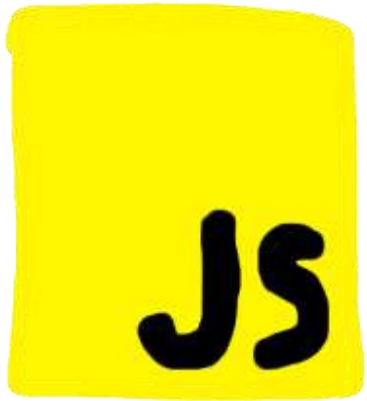


These get includes with Polymer and Shady DOM. That is its own talk!

Browser support	CHROME	OPERA	SAFARI	FIREFOX	EDGE
TEMPLATES	STABLE	STABLE	STABLE	STABLE	STABLE
CUSTOM ELEMENTS	STABLE	STABLE	STABLE	POLYFILL DEVELOPING	POLYFILL CONSIDERING
SHADOW DOM	STABLE	STABLE	STABLE	POLYFILL DEVELOPING	POLYFILL CONSIDERING
IMPORTS	STABLE	STABLE	POLYFILL ON HOLD	POLYFILL ON HOLD	POLYFILL CONSIDERING

https://www.webcomponents.org/

"ECMAScript was always an unwanted trade name that sounds like a skin disease." – Brendan Eich

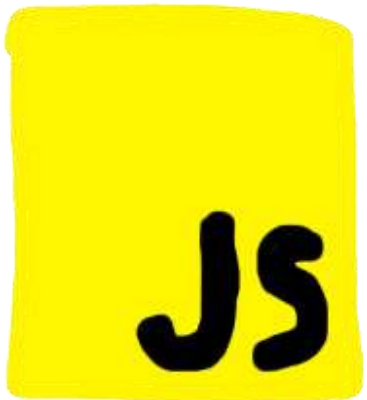


1995
Mocha
1996
Submitted to ECMA
2002
Firefox released
2008
Chrome released
2016
Edge released
LiveScript
ECMAScript 1
ECMAScript 2
ECMAScript 3
ECMAScript 5
ECMAScript 5.1
ECMAScript 6
ECMAScript 7
ECMAScript 8
ES.Next
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018

6/1997 - 6/1998
ECMAScript 1
6/1998 - 12/1999
ECMAScript 3
12/1999 - 12/2009
ECMAScript 5
1998
1999
2000

12/2009 - 6/2011
ECMAScript 5
6/2011 - 6/2015
ECMAScript 5.1
2010
2011
2012

2015
Edge released
6/2015 - 6/2016
ECMAScript 6
2016
2016 - ?
ECMAScript 7
2017 - ?
ECMAScript 8
ES...



ES 2016

Array.includes

** operator

Array.contains ???

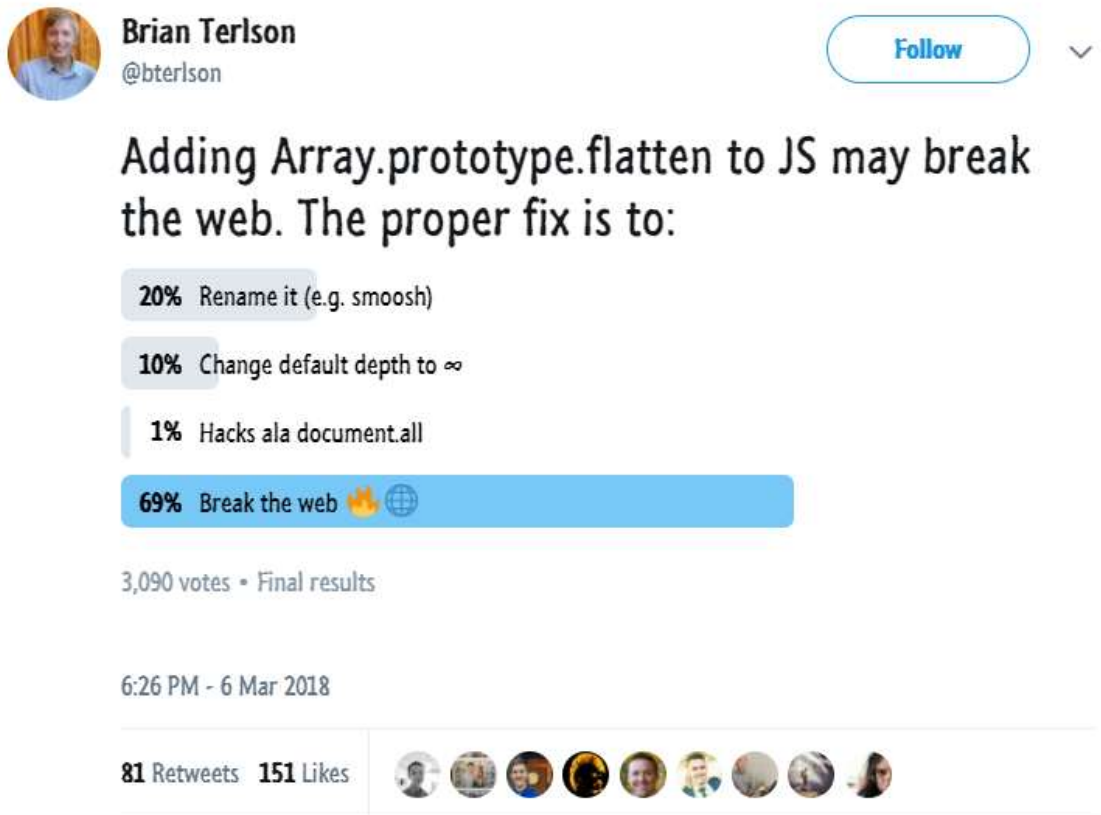
“You're telling me I should ship a browser that chokes on thousands of web sites that work fine today. That would be bad for our users, so I'm not planning on doing that."

"Patching 3.5 million websites is not a 'small fix' in any relevant sense.“

– Jason Orendorff (Mozilla)

moo tools

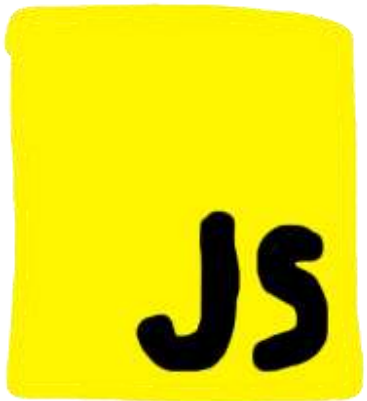
SMOOSH? Seriously?



prevent-smoosh

The purpose of this library is to prevent TC39 from adding Array.prototype.smoosh and Array.prototype.smooshMap as replacement names for Array.prototype.flatten and Array.prototype.flatMap. They usually don't listen to popular votes, but they do listen to the Web and usages of JavaScript in the wild. So here's your chance to vote with your code in production. Put this library in your production large-scale website (it's very tiny! won't hurt) and we'll have a compelling case for TC39 not to break backwards compatibility.

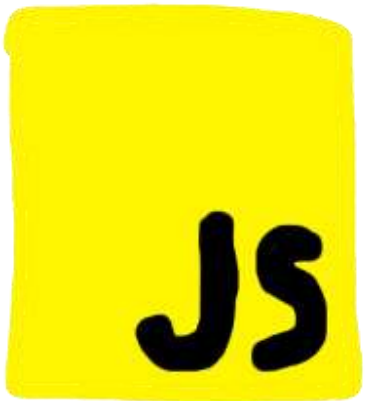
https://github.com/staltz/prevent-smoosh



ES 2016

Array.includes

** operator

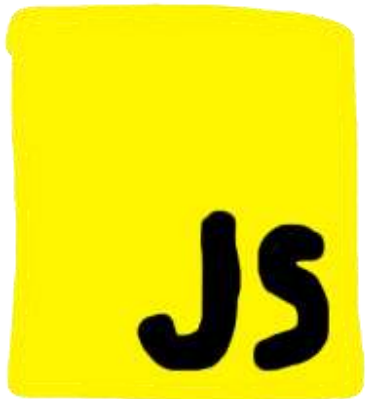


ES 2017

Async

Lots of little stuff

(Not an actual feature name.
Object.values, String Padding, not breaking on trailing commas, etc.)



ES 2018

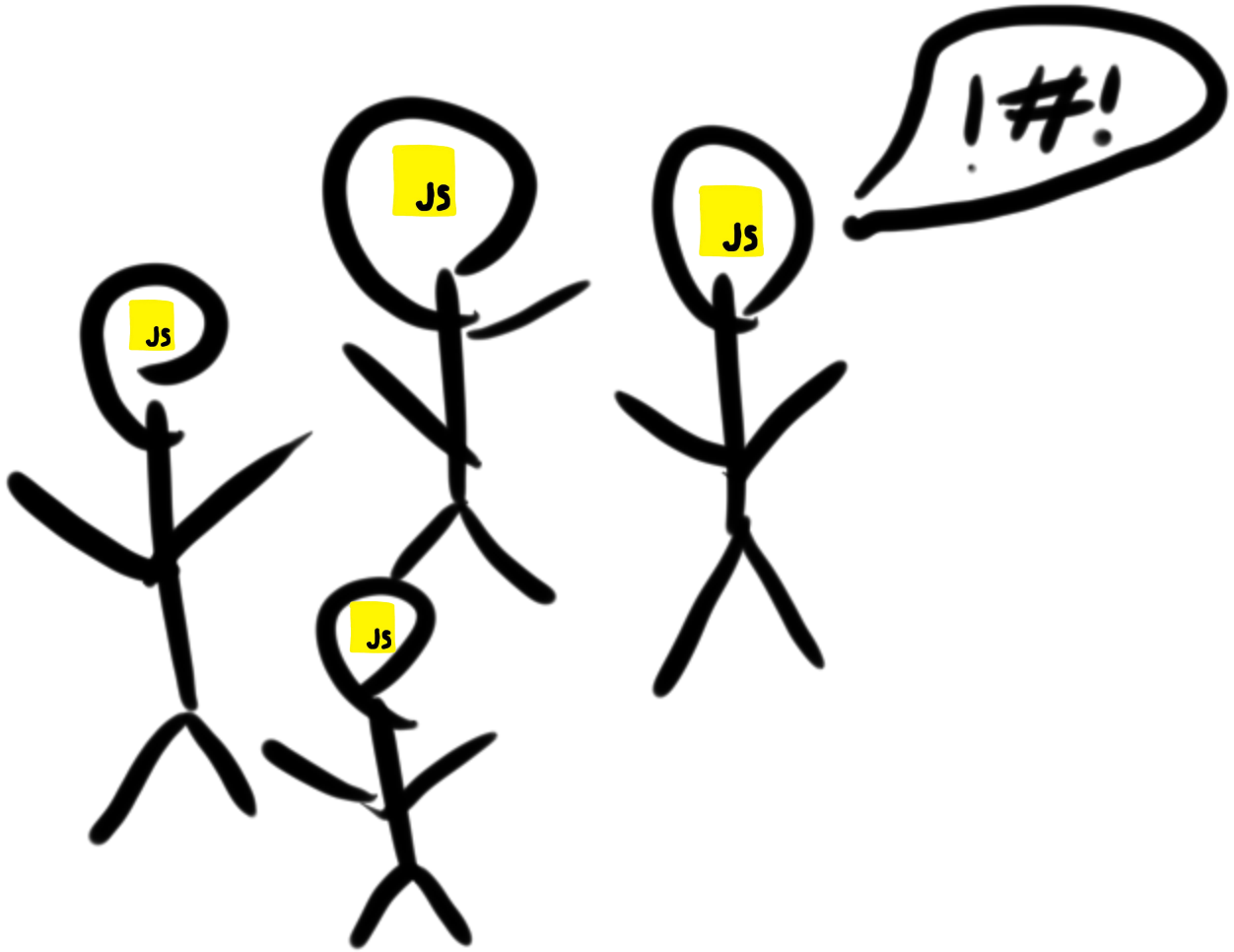
Shared Memory & Atomics

Async Iteration

Rest/Spread... operators

RegExp features

http://bit.ly/ESFeatures


JS
JS
JS
JS
!#!

Fetch

```
function SendButtonPush () {
    // NOTE - Look at the network traffic.

    debugger;

    fetch("/ButtonPushJSON", {
        method: "POST",
        body: JSON.stringify({ // Example of how you can send data
            secretCode: "12345",
            count: 1
        }),
        headers: {
            'Accept': 'application/json', // Telling the server this is JSON
            'Content-Type': 'application/json'
        }
    }).then(function(response) {
        return response; // I could use the response for something, but I don't here
    });

    // We don't need error handling. Let's live dangerously.
}
```

SEAPUNK commented on Dec 15, 2016

My apologizes in advance if I've missed something, but it seems this proposal has been abruptly withdrawn without much explanation. Is there a reason why?

28

domenic commented on Dec 15, 2016 Member

This proposal experienced significant opposition from within Google and so I am unable to continue working on it.

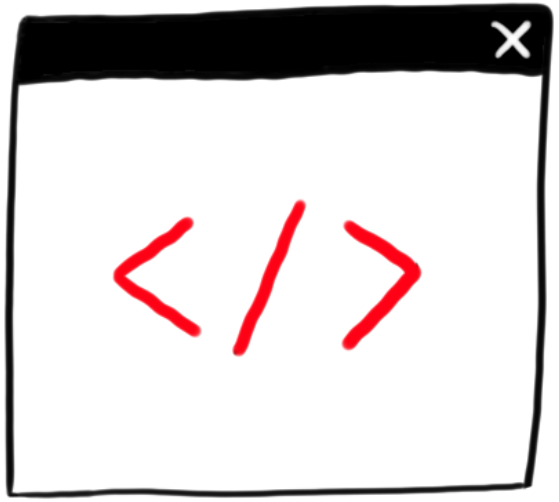
4 1 2 183

SEAPUNK commented on Dec 15, 2016

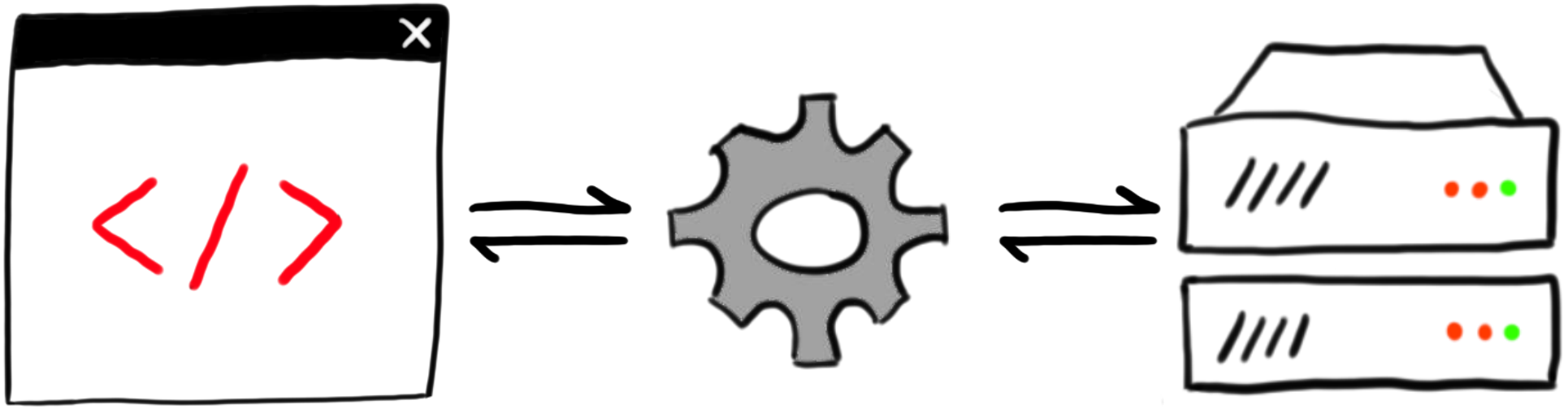
That sucks. In theory, somebody else could pick this proposal back up and champion it though, right?

domenic commented on Dec 15, 2016 Member

They could, but they would be blocked by other Googlers in TC39, so it would be fruitless.



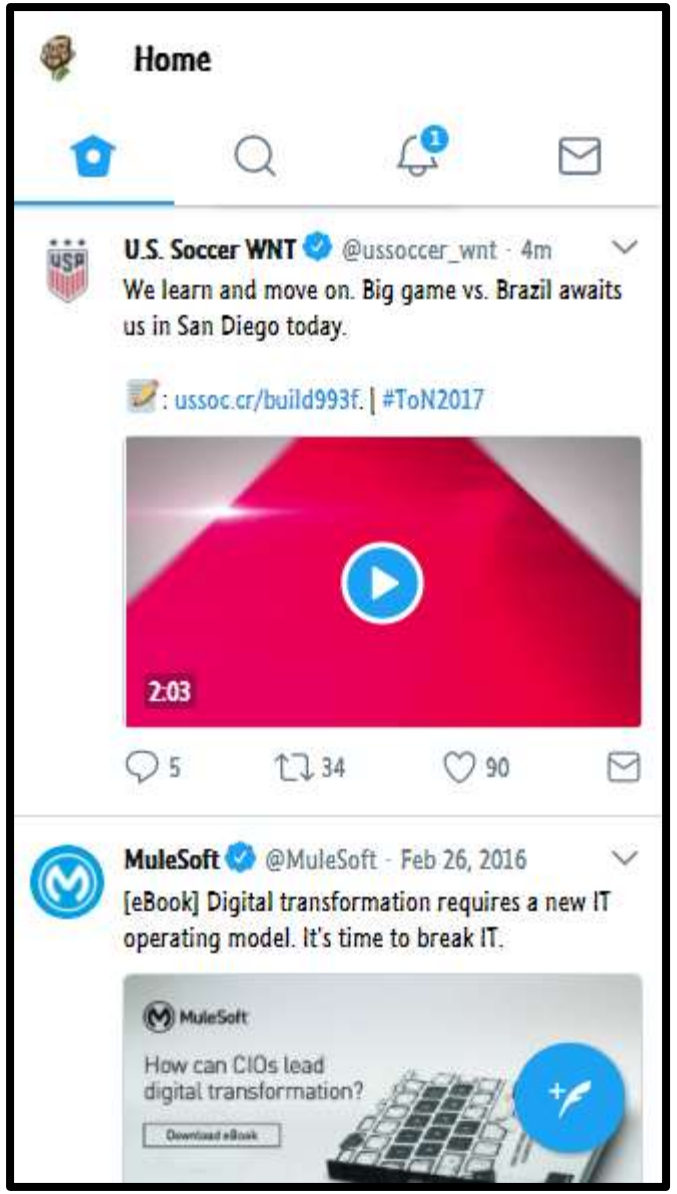
HTTP Requests



Service Worker

Progressive Web Apps

Google



Not Google

Twitter Is Complicated...

Here's a dumb example

https://github.com/jaredfaris/FetchAndServiceWorkerDemo

```
// Paths are important for caching files. Paths are relative to the SW
var filesToCache = [···

/* Fire off an event when the service worker installs so we can pre-cache files
or otherwise prep the SW */
self.addEventListener("install", function(event) {
    console.log("Installing the SW");

    // waitUntil does something a bit interesting. It waits for the promise to finish
    // before considering the SW installed. If the core cache doesn't load and this
    // fails, it won't consider it "installed" in a bad state.
    event.waitUntil(
        // Caches.open returns a promise that we'll chain off of
        caches.open("04a-cache")
            .then(function(cache) {
                // We'll add our files to the cache by passing in the array we defined before
                return cache.addAll(filesToCache);

            }).then(function() {
                console.log('SW Cache pre-configured');

            })
    );
});
```

```
// We want our cached objects to be returned from cache without any logic or timeouts.
self.addEventListener('fetch', event => {

    // We only want to serve cached results for a GET
    if (event.request.method === 'GET') {
        // We'll add Caching to allow us to serve up previously loaded files
        event.respondWith(
            // ignoreSearch will make the cache ignore ?stuff=things in the URL
            caches.match(event.request, {ignoreSearch:true}).then(response => {
                // If we have something in cache, we'll return that
                if(response)
                    return response;
                else // otherwise we'll fire off a fetch that won't get interdicted
                    return fetch(event.request);
            })
        );
    }
});
```

Offline?

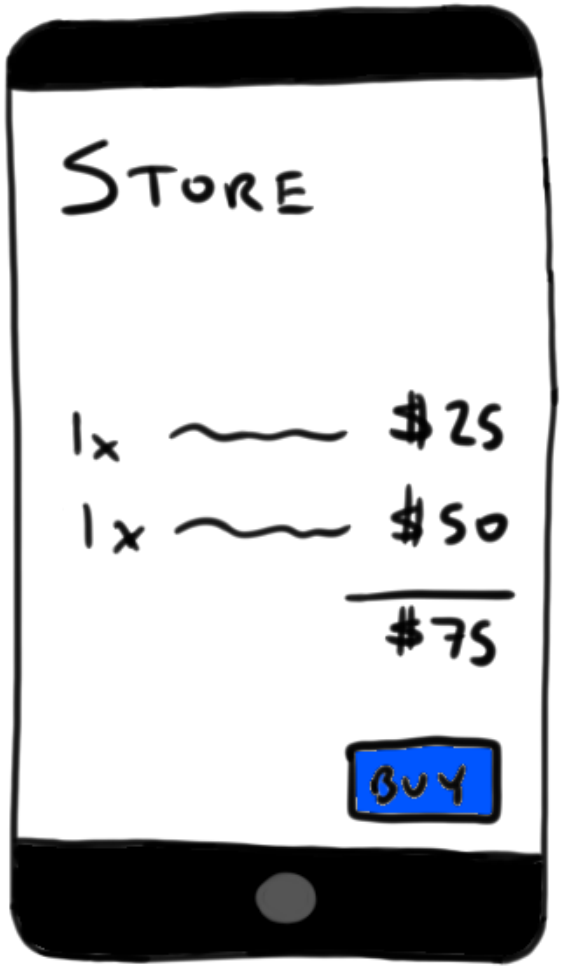
https://github.com/jaredfaris/FetchAndServiceWorkerDemo

```
// We're going to do some fancy background sync stuff here instead of just sending an AJAX request
document.getElementById("detonatorButton").addEventListener("click", function() {
    if(this.dataset.enabled !== "true")
        return;

    // Instead of POSTing data directly
    // SendButtonPush();

    // We're going to trigger a sync event that will do it
    addItemToOutbox().then(function() {
        reg.sync.register('buttonPush');
    })

});
```

```
// When we get a sync event, we'll pull anything that's been stored and POST it
// Then we'll clear out our storage
self.addEventListener('sync', function(event) {
    if(event.tag=="buttonPush") {
        // Get all of the stored button pushes
        getOutboxItems().then(function(buttonPushes) {
            var count = buttonPushes.length;

            fetch("/ButtonPushJSON", {
                method: "POST",
                body: JSON.stringify({ // Example of how you can send data
                    secretCode: "12345",
                    count: count
                }),
                headers: {
                    'Accept': 'application/json', // Telling the server this is JSON
                    'Content-Type': 'application/json'
                }
            }).then(function(res) {
                // Assuming this actually sends, let's clear out our outbox
                if(res.ok){
                    console.log("request successful", res);
                    clearOutbox();
                }
            })
        })
    }
});
```

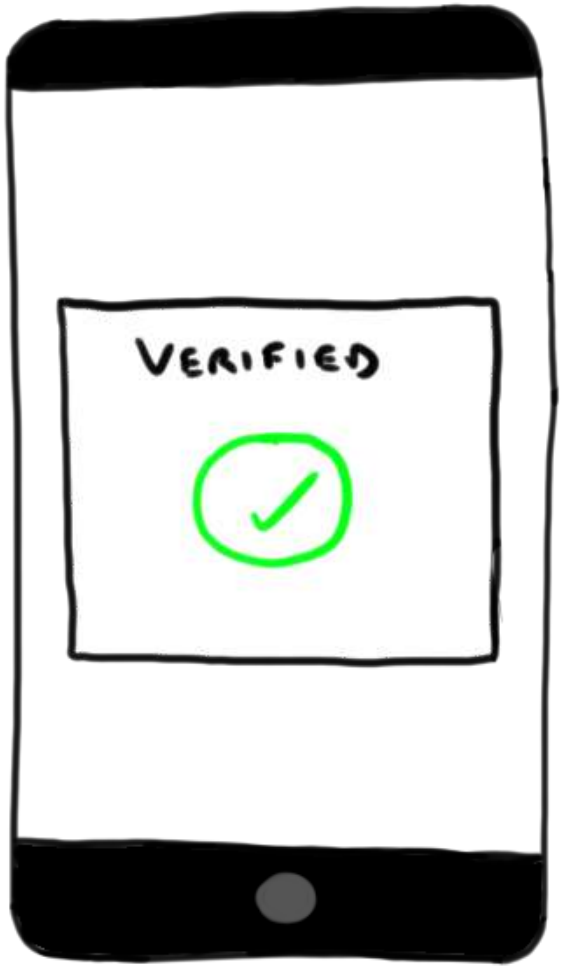
W3C

Web Payments



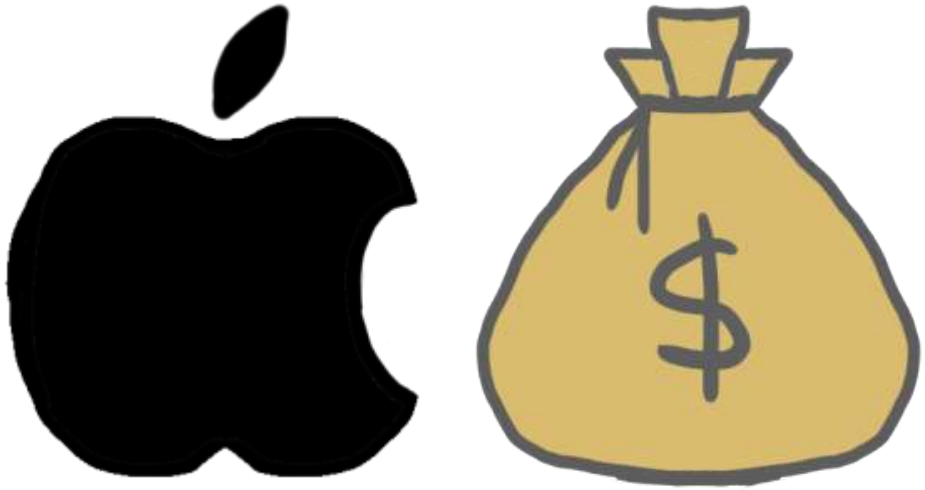
Video demo: http://bit.ly/WebPaymentDemo

Video demo: http://bit.ly/WebPaymentDemo



Video demo: http://bit.ly/WebPaymentDemo

Apple Pay?



Tube

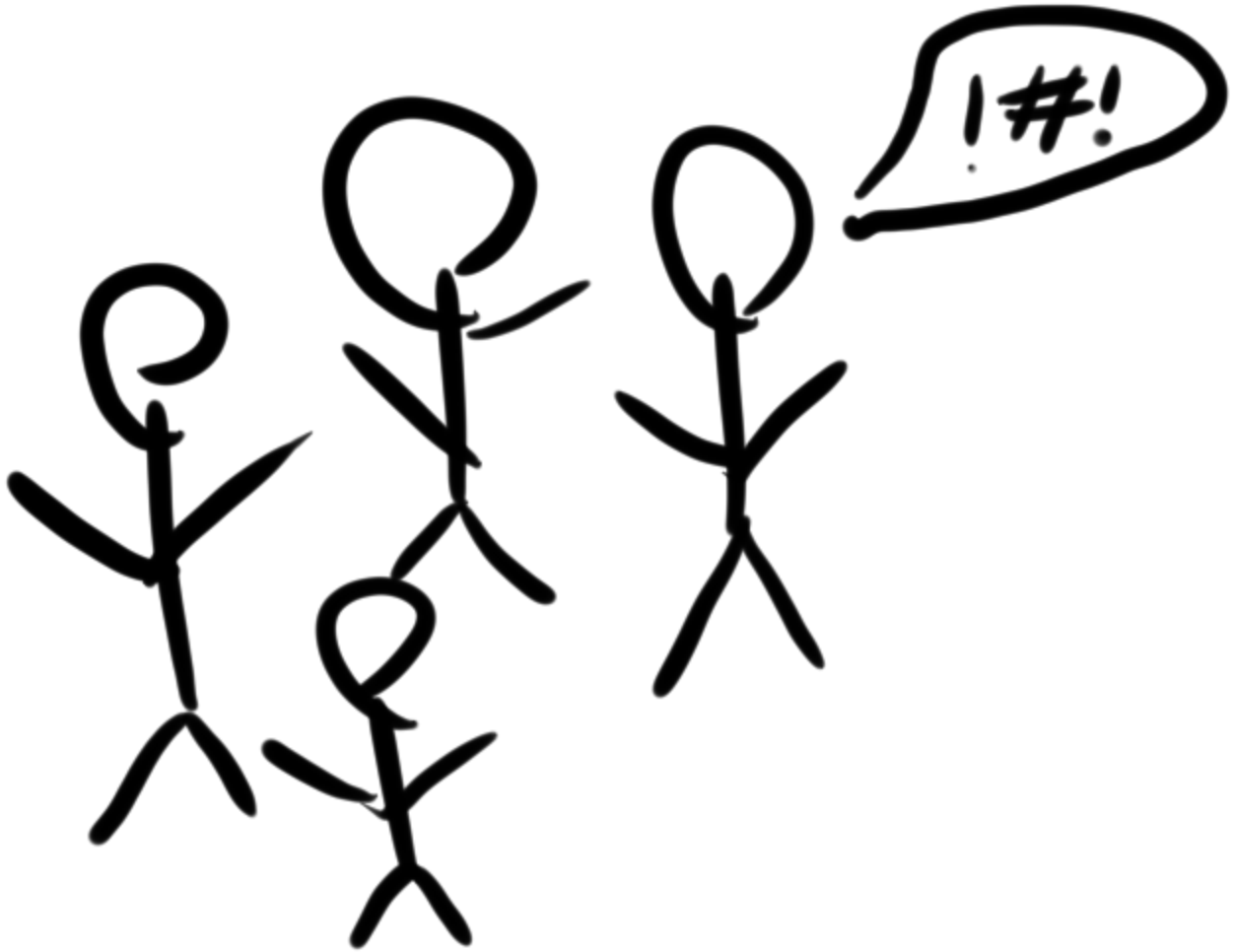
Internet Engineering ~~Task~~ Force

IETF

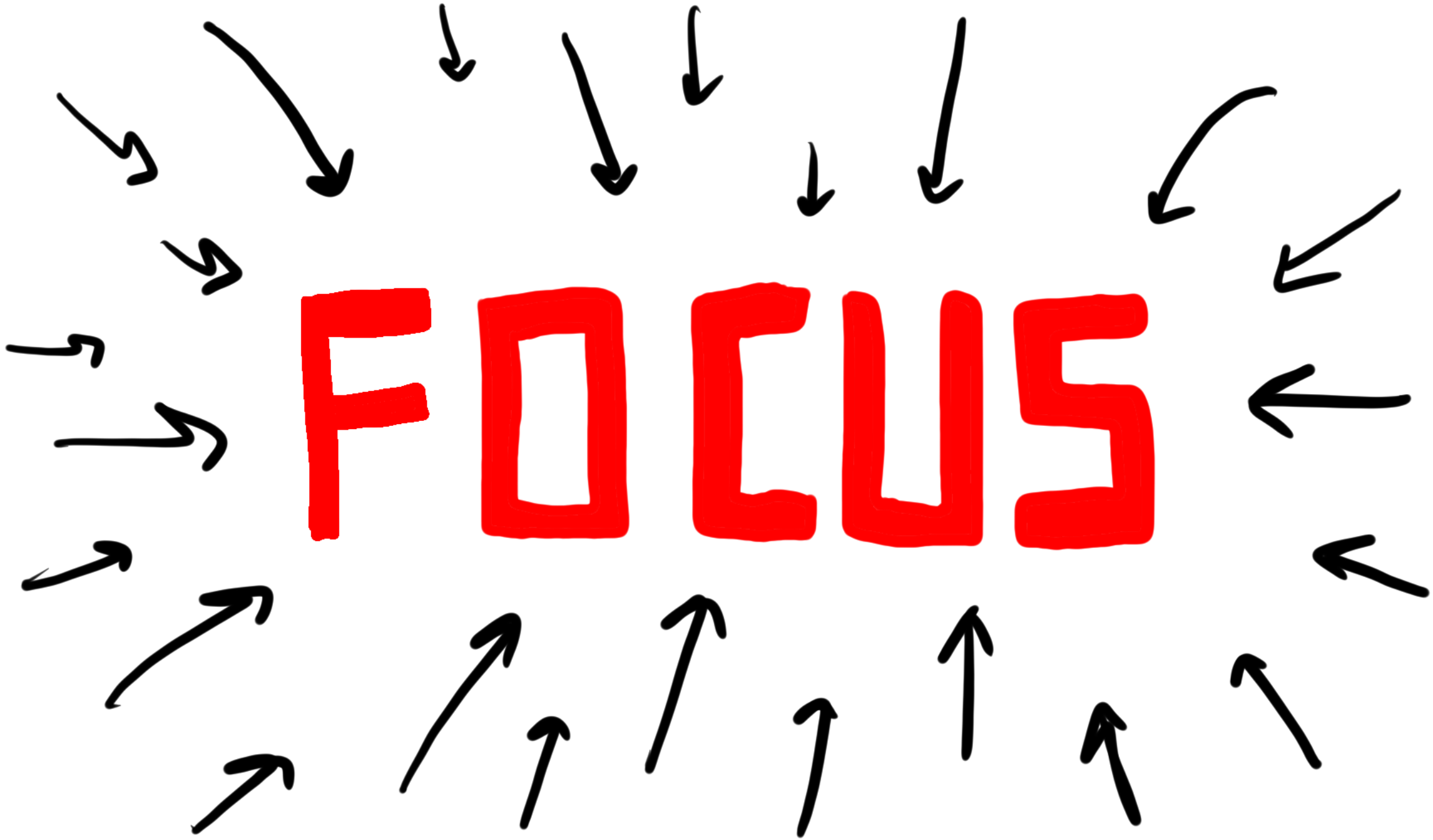
http2

Binary Framing Layer

¯_(ツ)_/¯



Security Researchers Swear A Lot


FOCUS

NOW GO AND LEARN NEW THINGS!

Thank You!

@jaredthenerd
jaredthenerd.com