


ESNext, Service Workers, and the Future of the Web


ScotlandJS 2016

@JemYoung

June 17, 2015


JavaScript
The Definitive Guide
O'REILLY
JavaScript:
The Good Parts
O'REILLY
YAHOO! PRESS
Douglas Crockford

ES6

Proxies

class

modules

Promises

let

arrow functions

template literals

const

destructuring

spread

generators

Map

Iterables

Set

rest

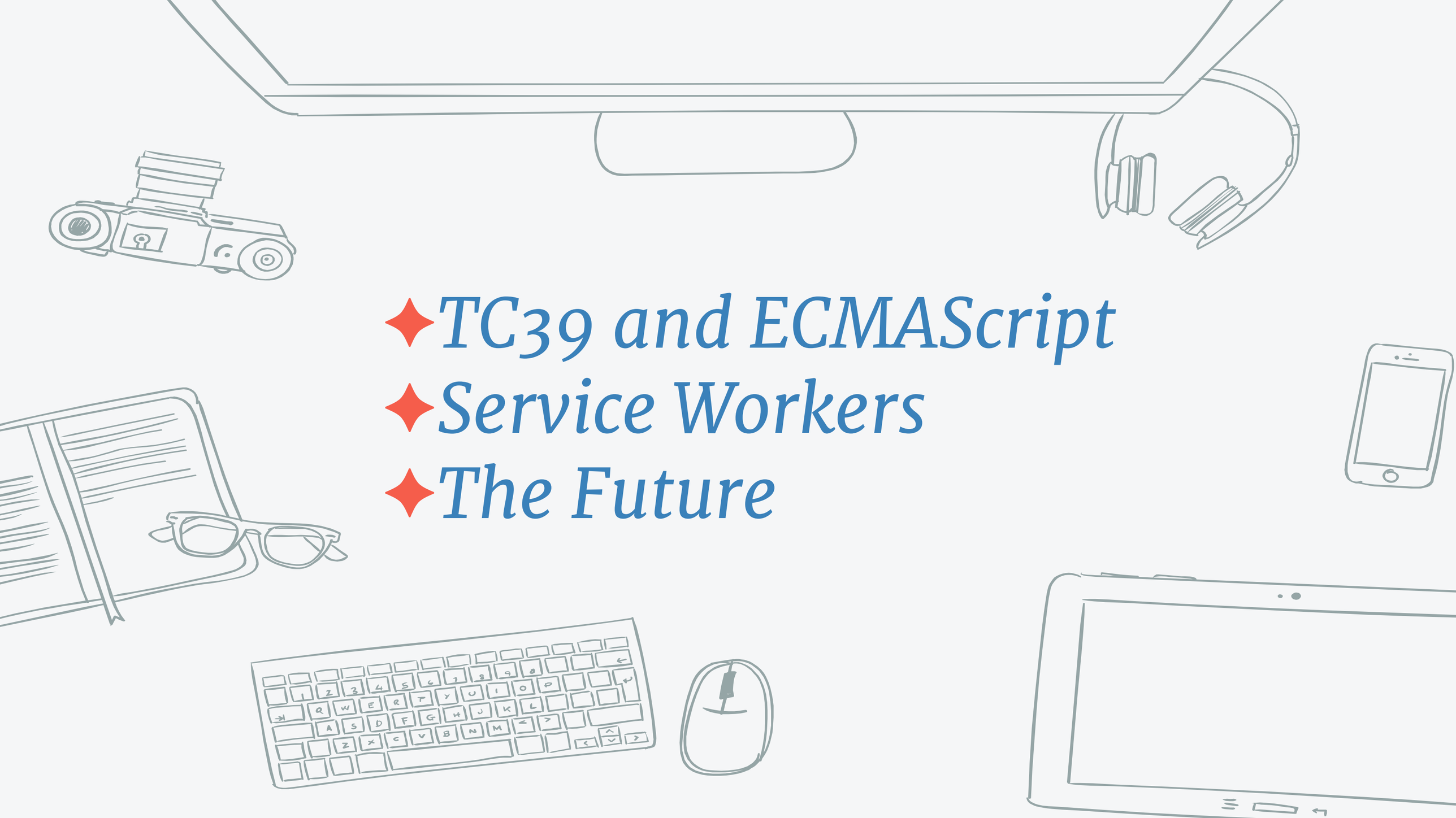
Symbols

WeakMap

default parameters

integer literals

Jem Young
Senior UI Engineer
@jemyoung



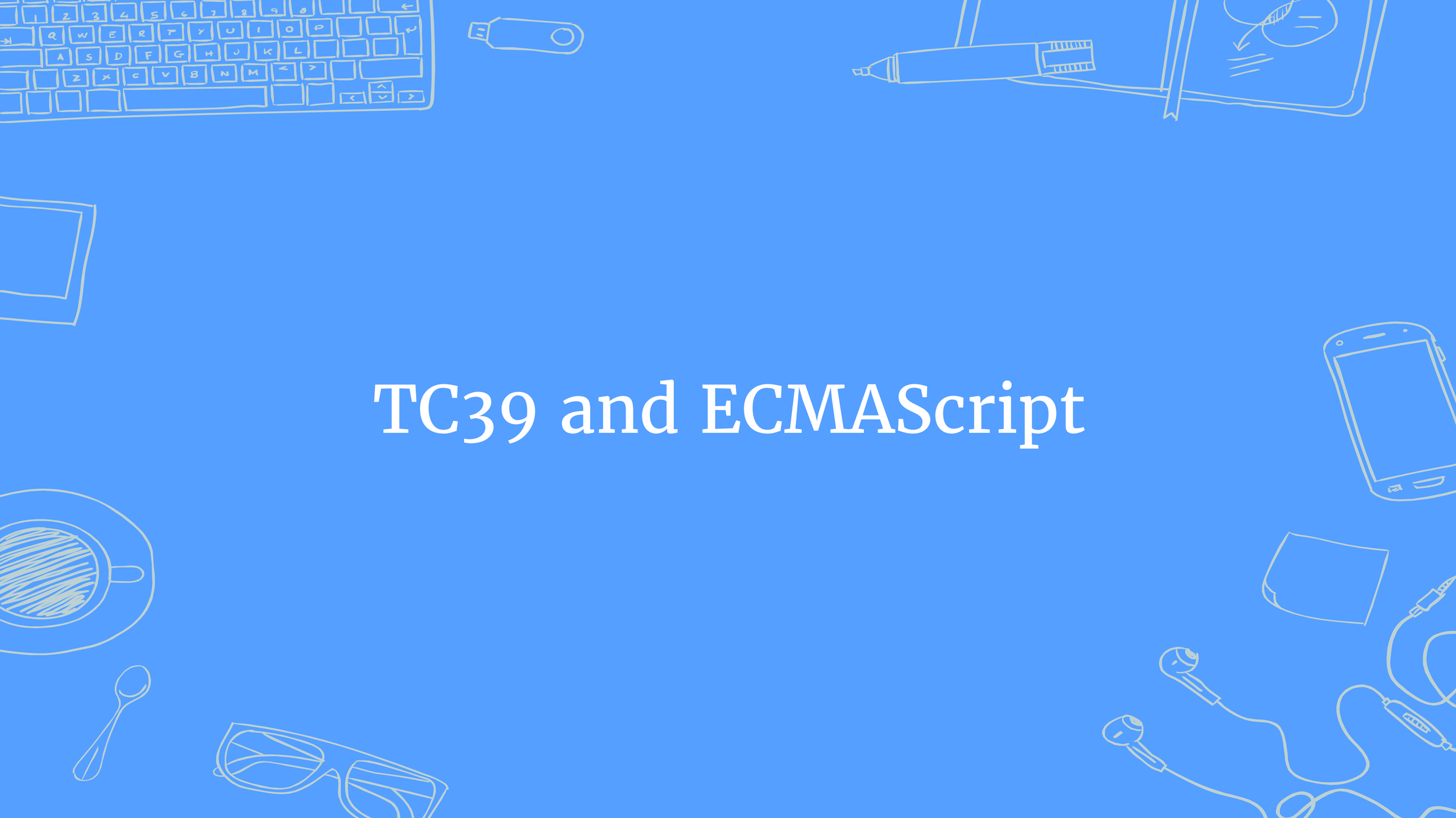
- TC39 and ECMAScript
- Service Workers
- The Future

Why should you listen to me?

https://young.github.io/service-worker/

Why should you listen to me?

TC39 and ECMAScript



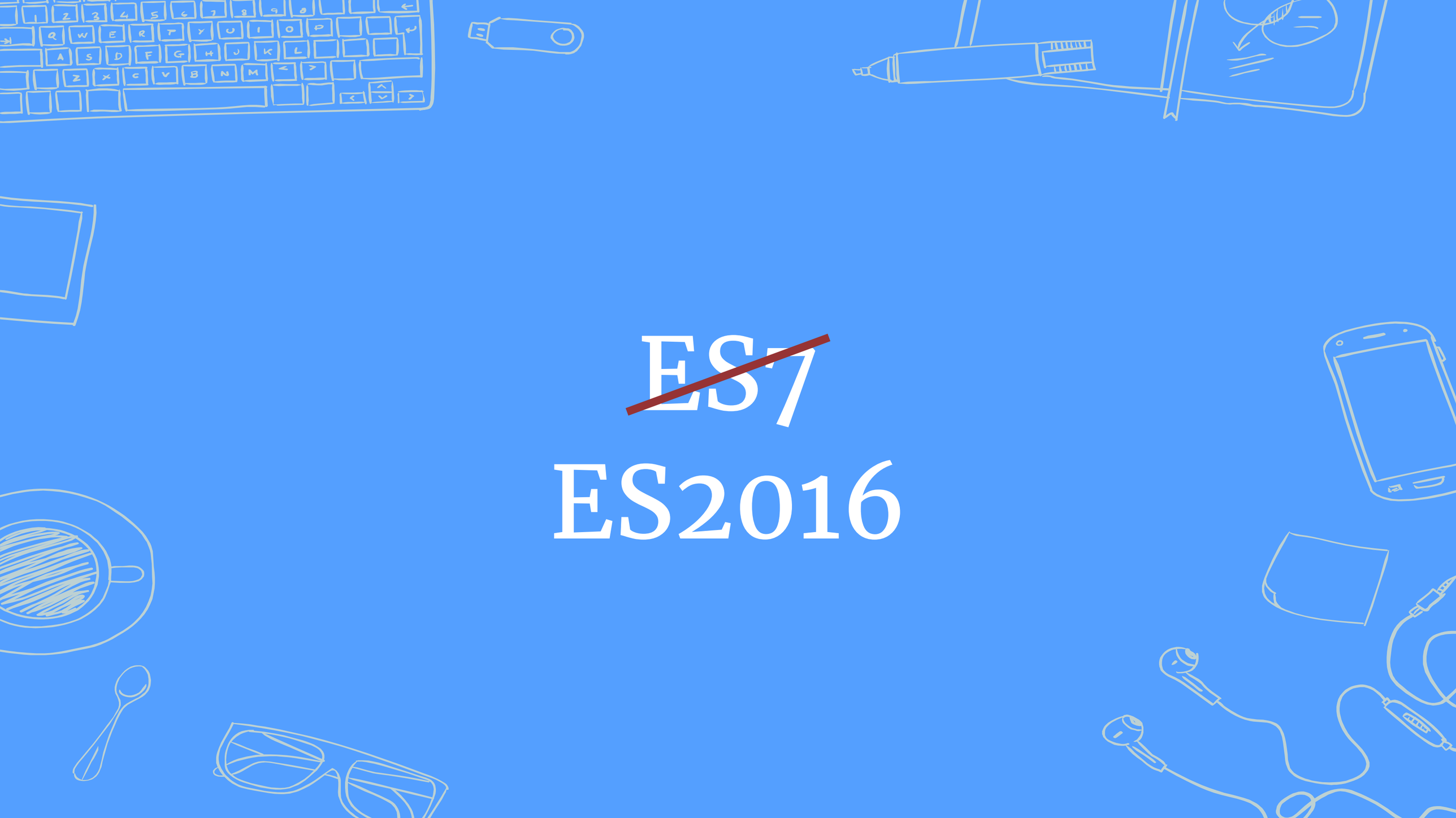
TC39

TC39

ECMAScript

~~ES7~~

ES2016



ES7

Array.includes

Exponentiation operator

ES7

Array.includes

```
const test = ['hello', 'friends'].includes('hello');
console.log(test); // true
```

```
const test1 = ['hello', NaN].indexOf(NaN);
console.log(test1); // false


const test2 = ['hello', NaN].includes(NaN);
console.log(test2); // true
```

ES7

Exponentiation operator

```
const e = 3**3;
console.log(e); // 27
```

```
Math.pow(3, 3) === 3**3; // true
```

ESNext

String.prototype.padStart

```
const test = 'JS';

console.log(test.padStart(4)); // '  JS'
console.log(test.padStart(4, 'o')); // 'ooJS'
console.log(test.padStart(10, 'Scotland')); // 'ScotlandJS'
```

String.prototype.padEnd

```
const test = 'JS';

console.log(test.padEnd(4)); // 'JS  '
console.log(test.padEnd(4, 'o')); // 'JSoo'
console.log(test.padEnd(10, 'Scotland')); // 'JSScotland'
```

ESNext

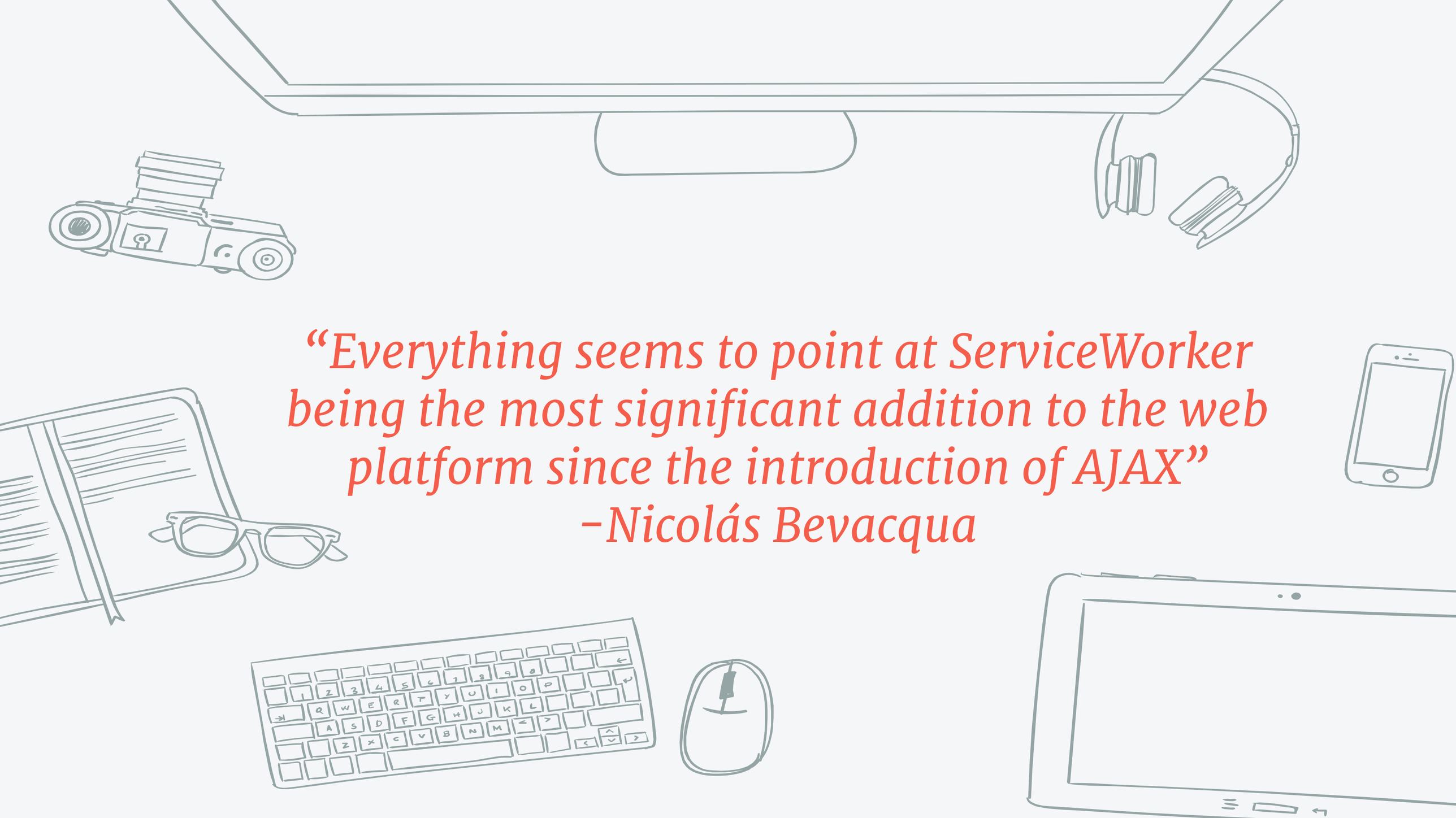
Object.values/Object.entries

```
const obj = { captain: 'picard', ship: 'USS Enterprise' };
console.log(Object.values(obj)); // ['picard', 'USS Enterprise' ]
console.log(Object.entries(obj)); // [ ['captain', 'picard'], ['ship', 'USS Enterprise'] ]
```

Object.getOwnPropertyDescriptors

```
const o = { foo: 42 };
Object.getOwnPropertyDescriptor(o, 'foo');
// Object {value: 42, writable: true, enumerable: true, configurable: true}
```

Service Workers



“Everything seems to point at ServiceWorker being the most significant addition to the web platform since the introduction of AJAX”
–Nicolás Bevacqua

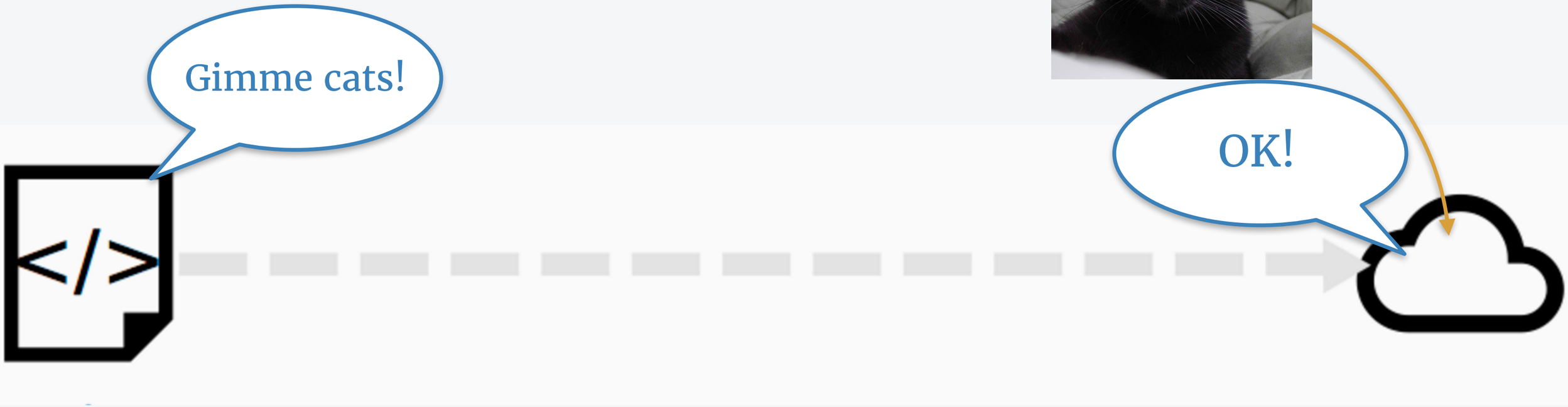
A persistent, fully asynchronous, separately threaded event based worker with the ability to intercept network requests and cache responses.

Service Worker

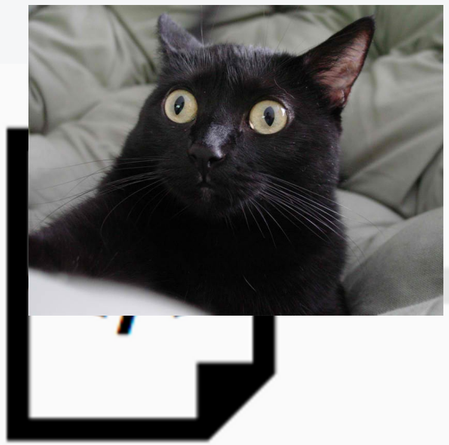
Web Worker

Service Worker

Service Worker



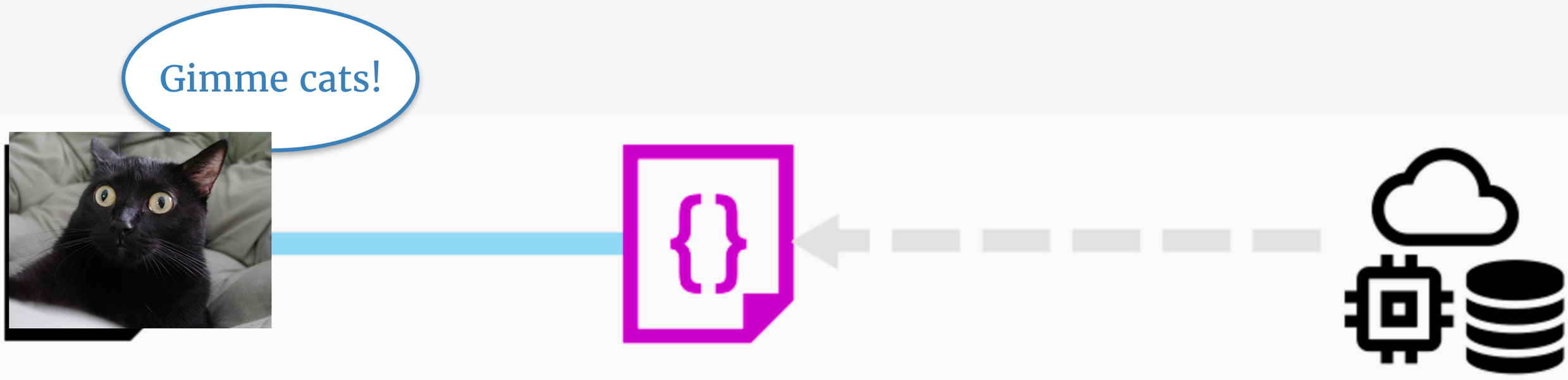
Service Worker



Web Page

Internet

Service Worker



[MDN – Service Workers]

Service Worker

A persistent, fully asynchronous, separately threaded event based worker with the ability to intercept network requests and cache responses.

Service Worker

persistent

- Exists outside of the browser tab
- Knows to when to expire

separately threaded

- Similar to Web Workers
- Does not die when the tab is closed

fully asynchronous

- Cannot use local storage or XHR
- Heavy utilization of Promises

event based

- No direct control from main thread
- install, activate, fetch, message

intercept network requests

- Proxy Fetch events
- Modify requests/responses

cache responses

- Persistent
- Cache Storage/Cache API

Service Worker

Events

event handler	event handler event type
oninstall	install
onactivate	activate
onfetch	fetch
onmessage	message

```
this.onactivate = () => {
  console.log('worker activated');
};
```

Or

```
this.addEventListener('activate', () => {
  console.log('worker activated');
});
```

W3C spec

Service Worker

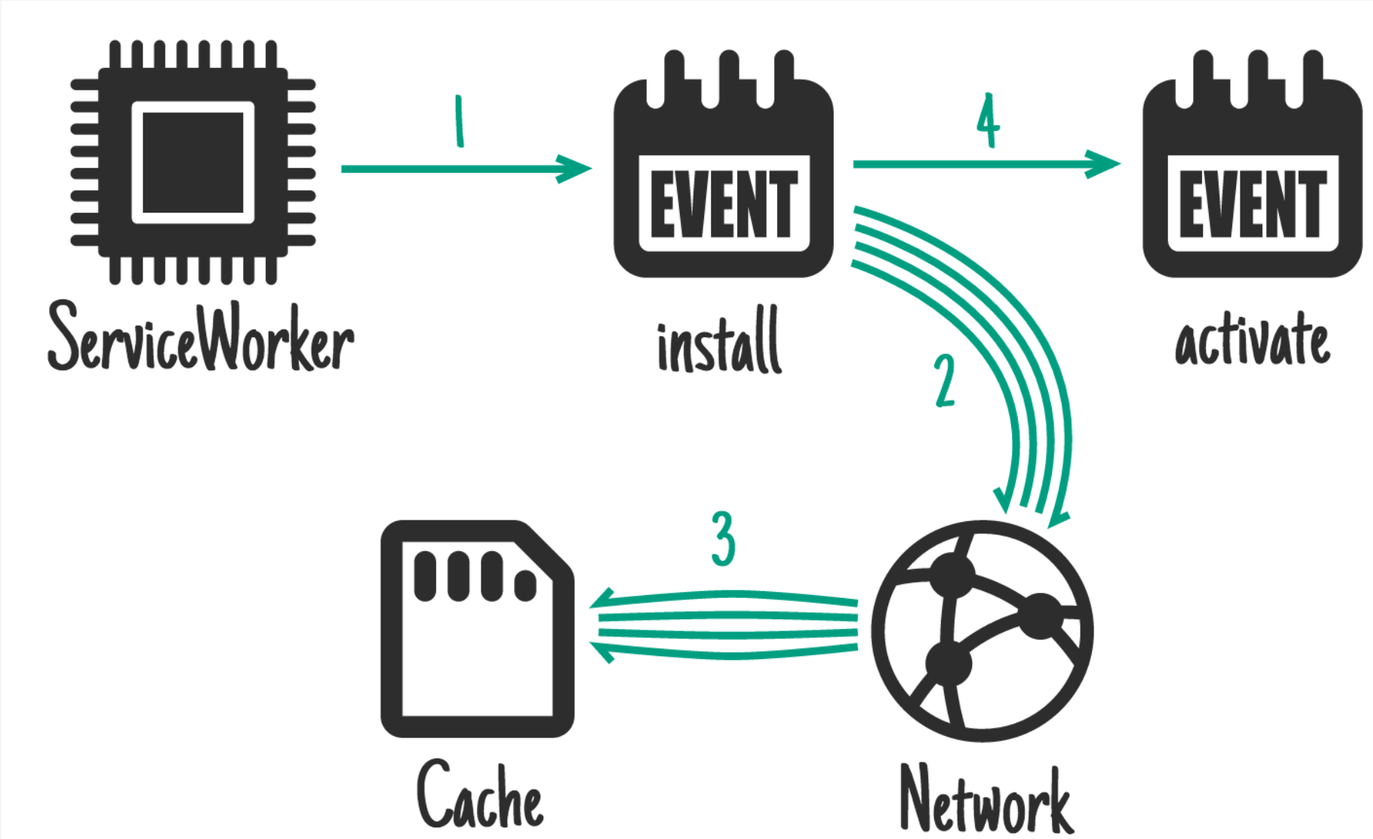
Initialization

```
<script type='text/javascript'>
    if ('serviceWorker' in navigator) {
      navigator.serviceWorker.register('./worker.js');
    }
</script>
```

Service Worker

Install

```
this.oninstall = (event) => {
  event.waitUntil(
    caches.open('v1')
      .then((cache) => {
        return cache.addAll([
          '/',
          '/index.html',
          '/images/cat1.jpg',
          '/images/cat2.jpg',
          '/images/cat3.jpg',
          '/images/cat4.jpg',
        ]);
      }).then(self.skipWaiting)
  );
};
```



[Offline cookbook]

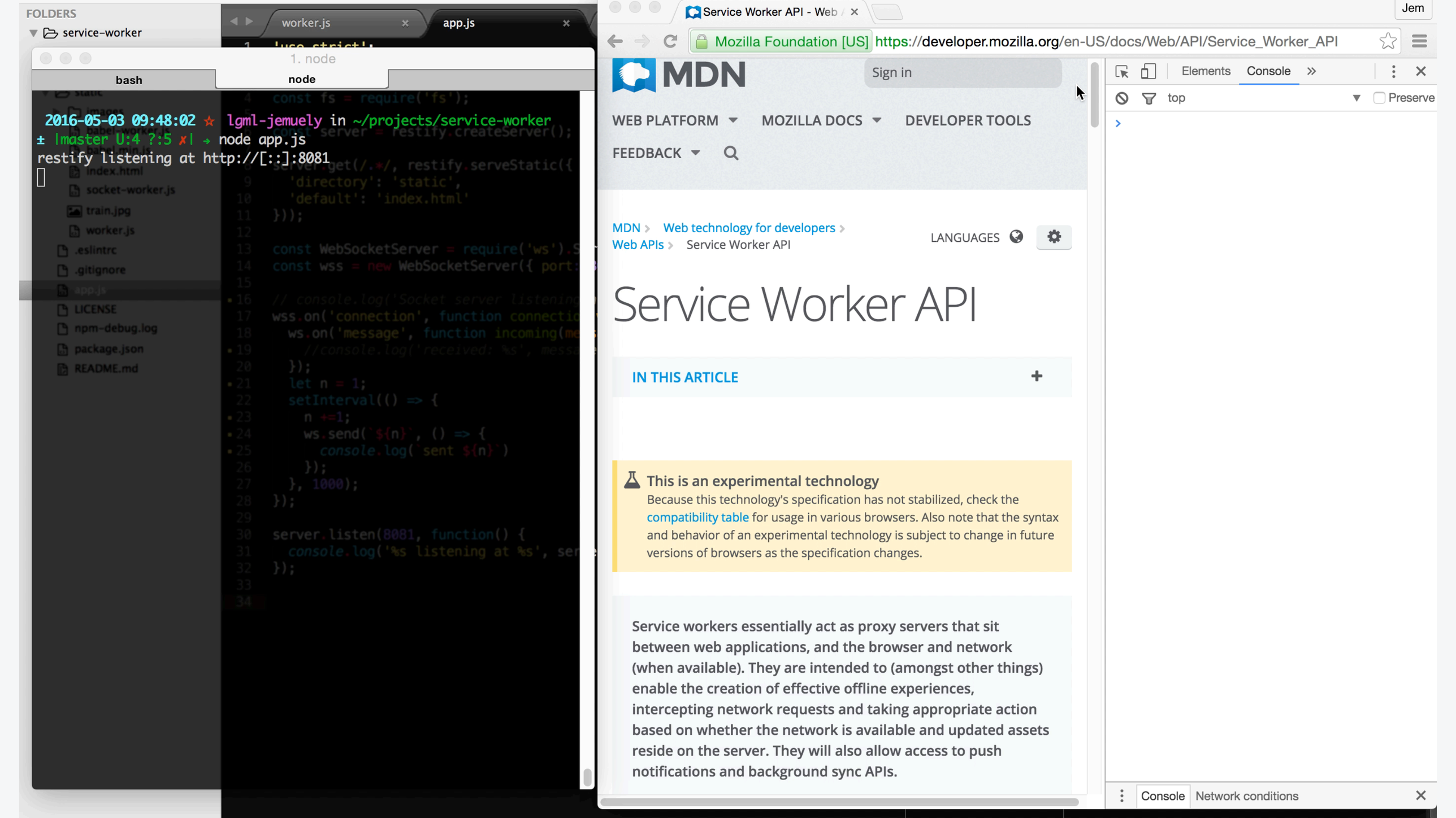
Service Worker

Service Worker

Fetch

```
this.onfetch = (event) => {
  const response =
    caches.match(event.request)
      .catch(() =>
        fetch(event.request)
          .then((res) => {
            const r = res.clone();
            caches.open('v1').then((cache) => {
              cache.put(event.request, r);
            });
          return res;
          }))
      .then((res) => res);

  event.respondWith(response);
};
```


FOLDERS
service-worker
worker.js
app.js
1. node
bash
node
2016-05-03 09:48:02 ☆ lgml-jemuely in ~/projects/service-worker
± |master U:4 ?:5 ✗| → node app.js
restify listening at http://[::]:8081
index.html
socket-worker.js
train.jpg
worker.js
.eslintrc
.gitignore
app.js
LICENSE
npm-debug.log
package.json
README.md
Service Worker API - Web
Jem
Mozilla Foundation [US] https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API
MDN
Sign in
WEB PLATFORM
MOZILLA DOCS
DEVELOPER TOOLS
FEEDBACK
MDN › Web technology for developers ›
Web APIs › Service Worker API
LANGUAGES
Service Worker API
IN THIS ARTICLE
This is an experimental technology
Because this technology's specification has not stabilized, check the compatibility table for usage in various browsers. Also note that the syntax and behavior of an experimental technology is subject to change in future versions of browsers as the specification changes.
Service workers essentially act as proxy servers that sit between web applications, and the browser and network (when available). They are intended to (amongst other things) enable the creation of effective offline experiences, intercepting network requests and taking appropriate action based on whether the network is available and updated assets reside on the server. They will also allow access to push notifications and background sync APIs.
Elements
Console
top
Preserve
Console
Network conditions

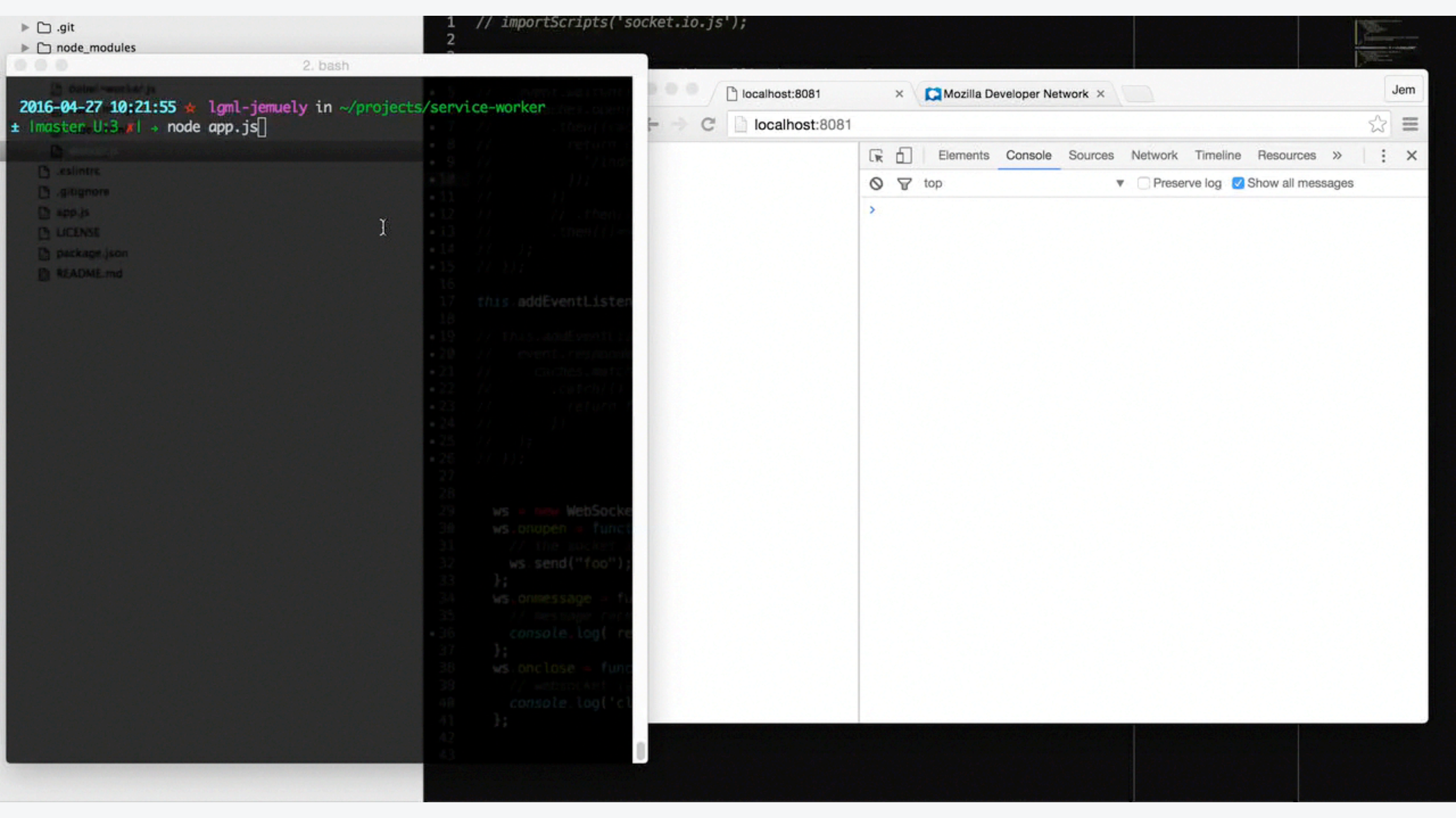
Service Worker

Ola Gasidlo – I’m Offline, Cool! What Now?

Service Worker

Service Worker

```
this.onactivate = () => {
  const ws = new WebSocket("ws://localhost:8082");
  ws.onopen = function() {
    // the socket is open
    ws.send("connected");
  };
  ws.onmessage = function (evt){
    // message received
    console.log(`received ${evt.data}`);
  };
  ws.onclose = function(e) {
    // websocket is closed.
    console.log('closed for', e);
  };
};
```


.git
node_modules
2. bash
2016-04-27 10:21:55 ☆ lgml-jemuely in ~/projects/service-worker
± |master U:3 ✗| → node app.js
.eslintrc
.gitignore
app.js
LICENSE
package.json
README.md
// importScripts('socket.io.js');
localhost:8081
Mozilla Developer Network
Jem
localhost:8081
Elements
Console
Sources
Network
Timeline
Resources
top
Preserve log
Show all messages

Service Worker

Service Worker

```
this.onfetch = (event) => {
  const isJS = isRequestJS(event.request);
  if (isJS && oldBrowser()) {
   event.respondWith(fetch(event.request)
    .then(response => response.blob())
    .then((response) => {
      const reader = new FileReader();
      const p = new Promise((resolve, reject) => {
        reader.onload = ({target: {result}}) => {
          const res = transform(result);
          resolve(res);
        };
      });
      reader.readAsText(response);
      return p;
    })
    .then(res => new Response(res))
    )

  } else {
    // Pass request through
    event.respondWith(fetch(event.request).then(response => response));
  }
};
```

Service Worker

https://github.com/young/live-transpile

Service Worker

```
class IMATest {
  constructor() {
    this.hello = {
      en: 'hello',
      es: 'hola'
    };
  }

  sayHello() {
    const {
      en,
      es
    } = this.hello;
    console.log(en);
    console.log(es);
  }
}
```

```
var _createClass = function() {
  function defineProperties(target, props) {
    for (var i = 0; i < props.length; i++) {
      var descriptor = props[i];
      descriptor.enumerable = descriptor.enumerable || false;
      descriptor.configurable = true;
      if ("value" in descriptor) descriptor.writable = true;
      Object.defineProperty(target, descriptor.key, descriptor);
    }
  }
  return function(Constructor, protoProps, staticProps) {
    if (protoProps) defineProperties(Constructor.prototype, protoProps);
    if (staticProps) defineProperties(Constructor, staticProps);
    return Constructor;
  };
}();

function _classCallCheck(instance, Constructor) {
  if (!(instance instanceof Constructor)) {
    throw new TypeError("Cannot call a class as a function");
  }
}

var IMATest = function() {
  function IMATest() {
    _classCallCheck(this, IMATest);

    this.hello = {
      en: 'hello',
      es: 'hola'
    };
  }

  _createClass(IMATest, [{
    key: 'sayHello',
    value: function sayHello() {
      var _hello = this.hello;
      var en = _hello.en;
      var es = _hello.es;

      console.log(en);
      console.log(es);
    }
  }]);

  return IMATest;
}();
```

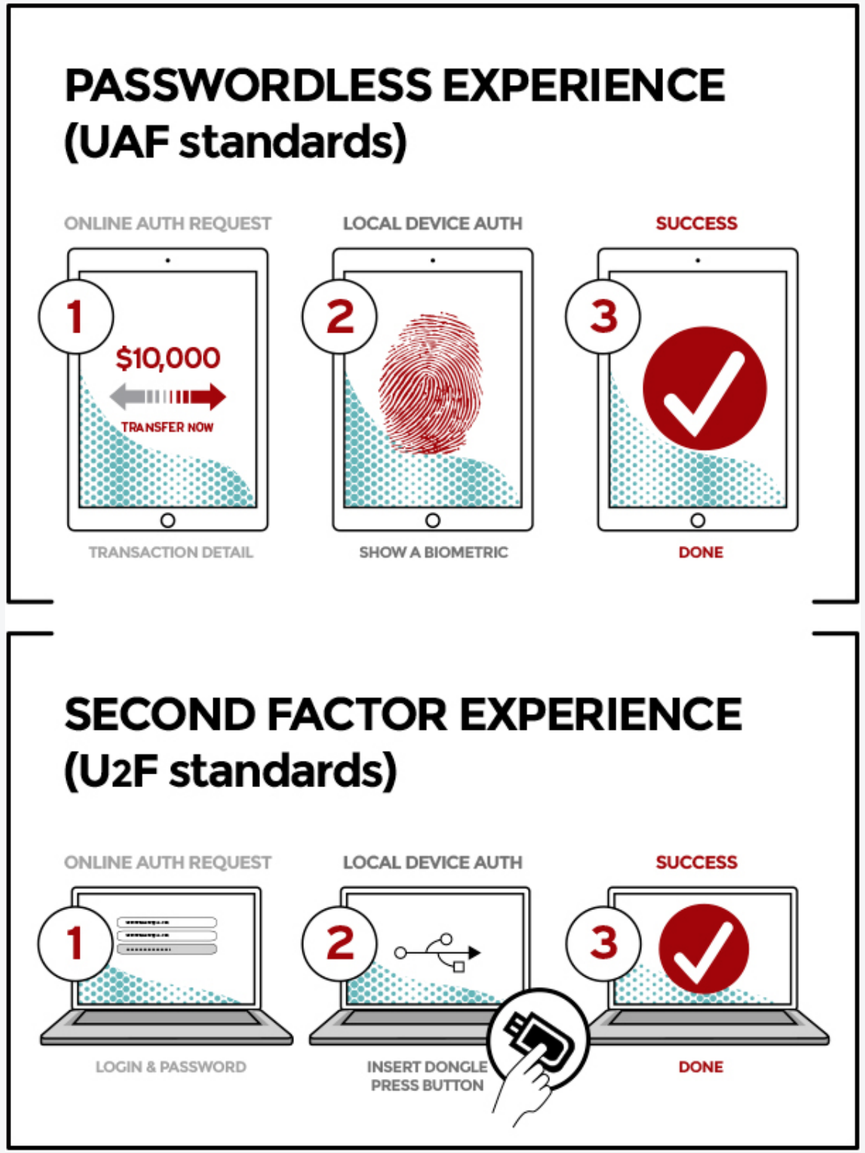
The Future of the Web

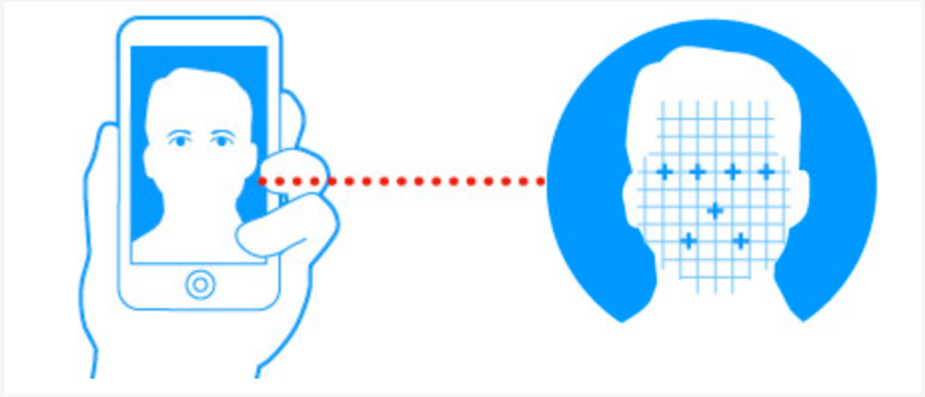
The Future

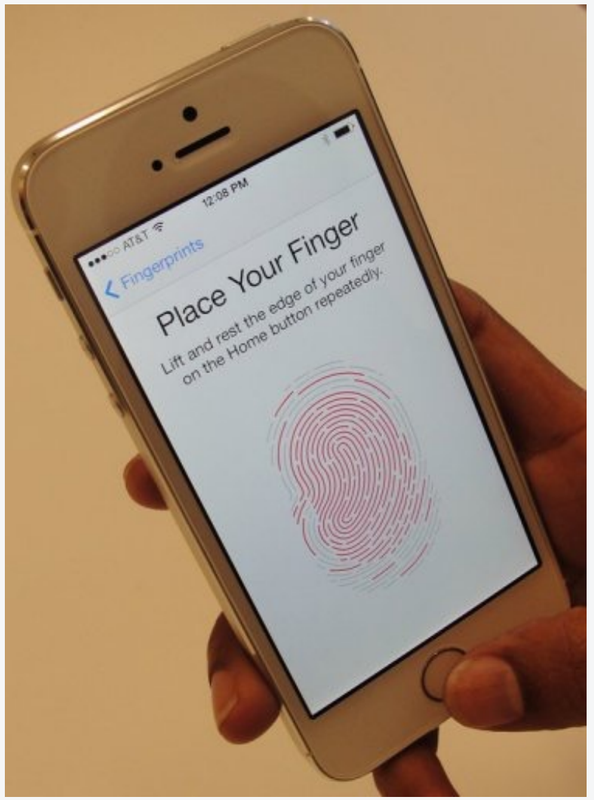
FIDO

The Future

Fast IDentity Online







The Future

Web Bluetooth

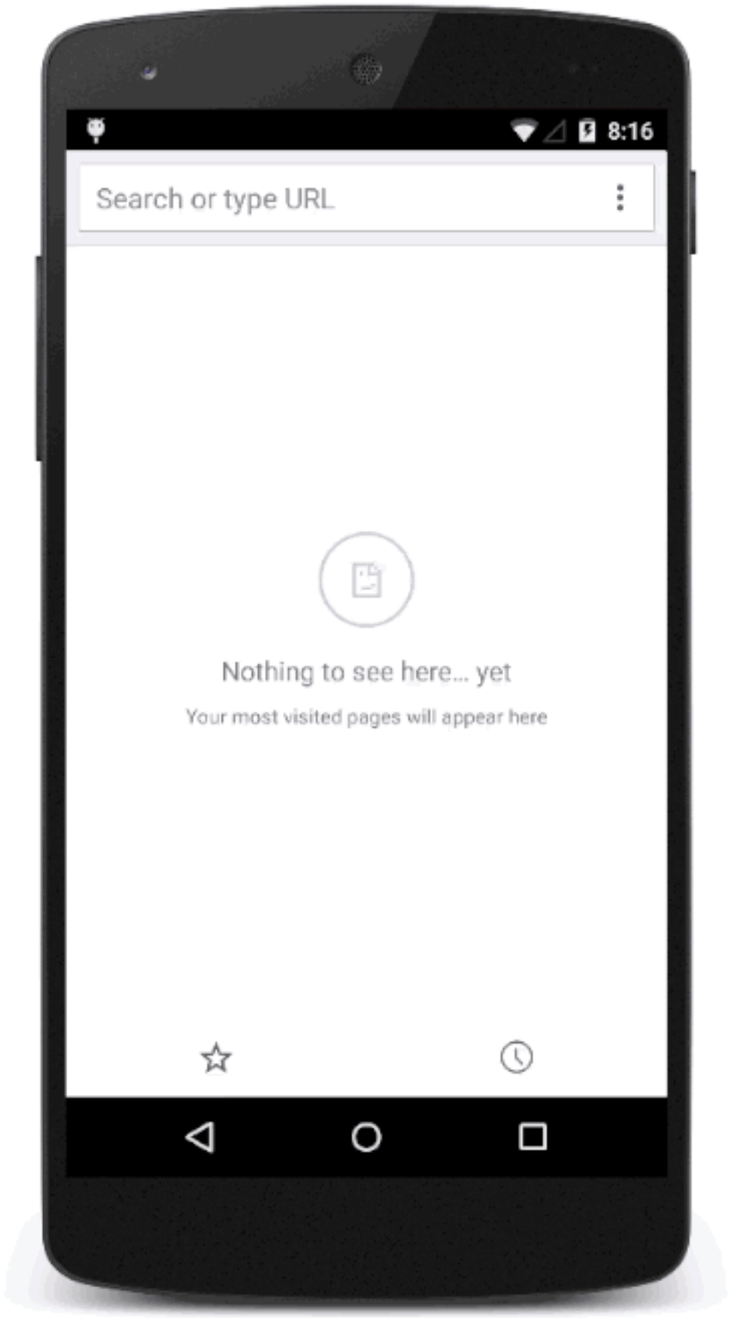
The Future

Web Bluetooth

```
navigator.bluetooth.requestDevice({ filters: [{ services: ['heart_rate'] }] })
.then(device => device.gatt.connect())
.then(server => server.getPrimaryService('heart_rate'))
.then(service => service.getCharacteristic('heart_rate_measurement'))
.then(characteristic => {
  return characteristic.startNotifications()
  .then(() => {
    characteristic.addEventListener('characteristicvaluechanged',
      handleNotifications);
  });
})
.then(() => {
  console.log('Notifications have been started.');
})
.catch(error => { console.log(error); });
```

Dan Jenkins – “Getting Physical with Web Bluetooth”

The Future

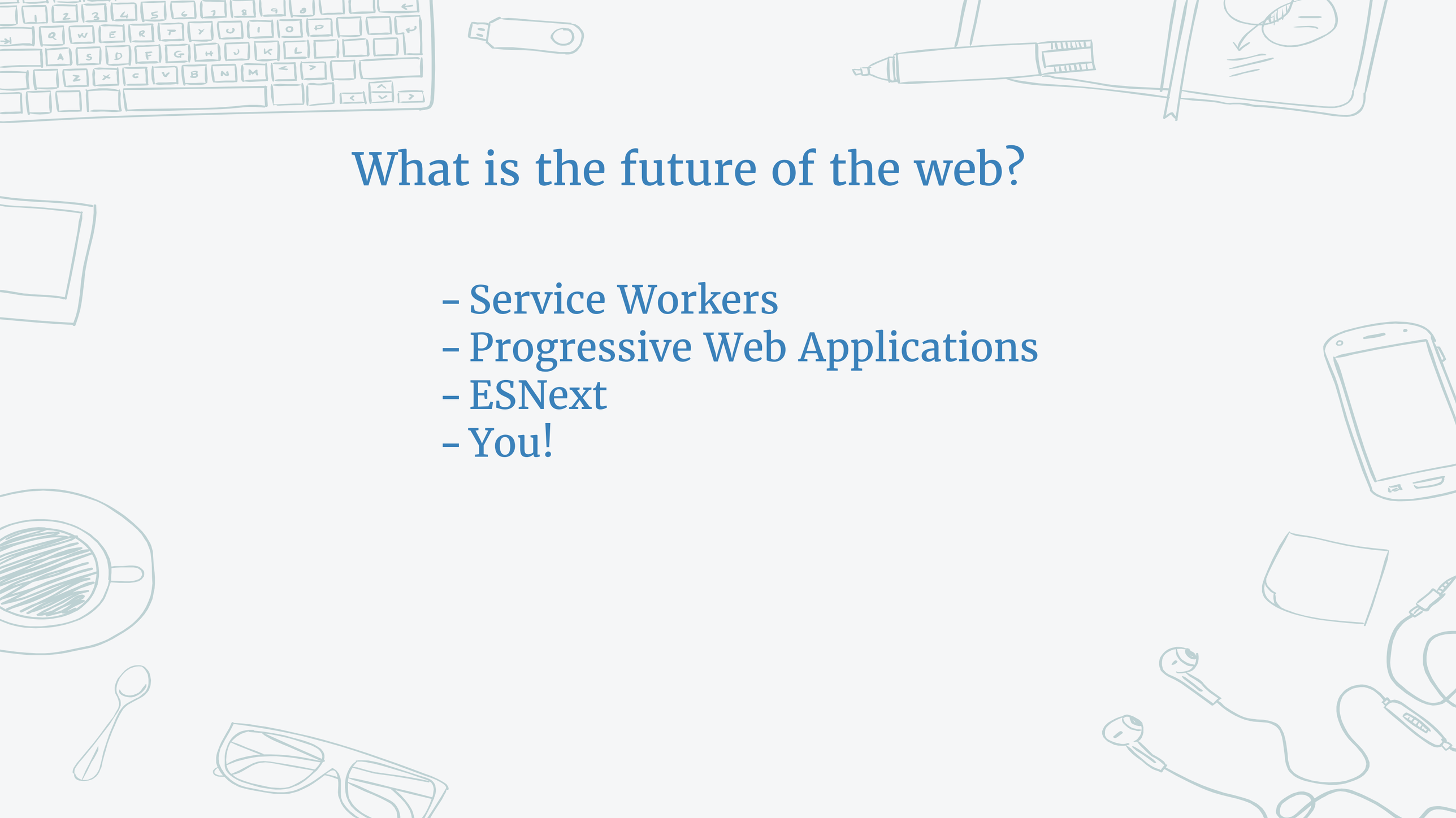

8:16
Search or type URL
Nothing to see here... yet
Your most visited pages will appear here


WHEN YOU DO THINGS RIGHT
PEOPLE WON'T BE SURE YOU'VE DONE ANYTHING AT ALL

What is the future of the web?

https://young.github.io/service-worker/

What is the future of the web?



What is the future of the web?

- Service Workers
- Progressive Web Applications
- ESNext
- You!

Links

Is Service Worker Ready?

TC39 proposals

Service Worker Experiments

Offline cookbook

MDN – Using Service Workers

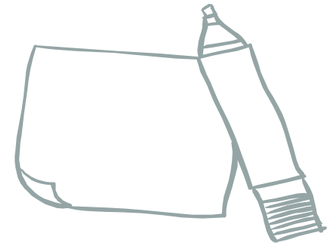
Service Worker Cookbook

website

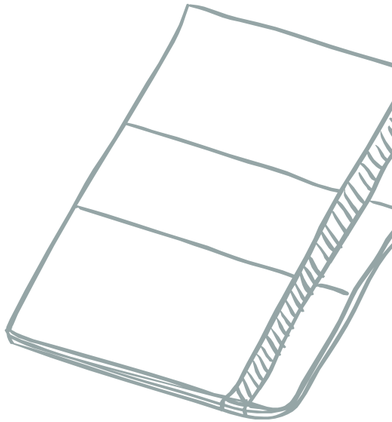
GitHub

@jemyoung.com

twitter









ESNext, Service Workers, and the Future of the Web


ScotlandJS 2016

@JemYoung