Modern JavaScript and PhoneGap

PhoneGap Day EU 2017

Kerri Shotts • @kerrishotts

Image by Pete Linforth (https://pixabay.com/en/users/PeteLinforth-202249/), courtesy of Pixabay.com

About Me

- Used PhoneGap for over six years
- Authored Five books about PhoneGap
- Apache Cordova committer
- One of many moderators at:
  - Cordova Google Group
  - PhoneGap Adobe Forums
- I love retro technology and ST:TNG 😉

Modern JavaScript Versions

Remember ECMAScript 5?

Release year: 2009

- The version we all know and love (~ish?)
- Supported by all modern mobile web views[1]
  - iOS 6+, IE 10+, Edge (forever), Android 4.4+
- Reasonably modern (`map`, `reduce`, getters/setters, etc.)

1. http://caniuse.com/#feat=es5

Things have changed a lot since then...

ES2015 and beyond

2015[1]	Block-scoped `let` & `const`	Destructuring and named parms
	Default parameters	Rest and Spread operator (`...`)
	`for...of` loops and Iterators	Arrow functions (`=>`)
	Template strings & interpolation	Improved literals (object, `0b10`)
	Generators (`*` / `yield`)	Symbols, Maps & Sets, Promises
	`class` syntactic sugar & `super`	Modules (`import` , `export`)
2016[2]	Exponent (`**`)	`Array.prototype.incudes()`
2017[3]	`async` / `await`	String padding 😉
	Shared memory	Atomics

1. https://github.com/lukehoban/es6features#readme; the list here is not a complete representation of *all* features
2. http://www.2ality.com/2016/01/ecmascript-2016.html
3. http://www.2ality.com/2016/02/ecmascript-2017.html

Before we go any further…

Some Very Important Caveats!

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

Important Caveats

- ES2015+ is **NOT** a performance optimization
  - See https://kpdecker.github.io/six-speed/
- Typically requires a build step
- Debugging can be interesting
- Some of the syntax is a little *sharp* — use with care

		node 6.9.3	node 7.3.0	chrome 55	chrome 49	firefox 50	firefox 53	edge 14.14393	edge 15.14959	safari 10	webkit 604.1.2+
arrow tests	babel	1.2x slower	Identical	Identical	Identical	Identical	Identical	Identical	1.2x slower	Identical	Identical
	typescript	Identical	Identical	Identical	Identical	Identical	Identical	1.2x slower	1.2x faster	Identical	1.2x slower
	es6	Identical	Identical	Identical	Identical	Identical	Identical	Identical	1.2x faster	Identical	Identical
arrow-args tests	babel	Identical	Identical	Identical	Identical	Identical	Identical	Identical	1.5x slower	Identical	Identical
	typescript	Incorrect	Incorrect	Incorrect	Incorrect	Incorrect	Incorrect	Incorrect	Incorrect	Incorrect	Incorrect
	es6	Identical	Identical	Identical	Identical	Identical	Identical	Identical	Identical	69x slower	71x slower
arrow-declare tests	babel	1.6x slower	1.5x slower	1.4x slower	1.4x slower	30x slower	23x slower	1.3x slower	1.7x slower	Identical	Identical
	typescript	1.4x slower	1.3x slower	1.3x slower	1.4x slower	15x slower	23x slower	1.5x slower	1.4x slower	Identical	Identical
	es6	1.4x slower	1.3x slower	1.3x slower	1.3x slower	38x slower	56x slower	1.2x slower	1.5x slower	Identical	Identical
bindings tests	babel	Identical	Identical	Identical	Identical	Identical	Identical	Identical	1.4x slower	Identical	1.2x slower
	typescript	Identical	Identical	Identical	Identical	Identical	Identical	Identical	Identical	Identical	Identical
	es6	Identical	Identical	Identical	Identical	Identical	Identical	Identical	Identical	Identical	Identical
bindings-compound tests	babel	Identical	Identical	Identical	Identical	Identical	Identical	Identical	1.2x slower	Identical	Identical
	typescript	Identical	Identical	Identical	Identical	Identical	Identical	Identical	Identical	Identical	Identical
	es6	Identical	1.6x slower	1.6x slower	1.5x slower	Identical	Identical	1.2x slower	Identical	Identical	Identical
classes tests	babel	1.9x slower	9x slower	1.5x slower	1.4x slower	15x faster	33x slower	1.5x slower	1.2x slower	7x slower	7x slower
	babel-loose	1.7x slower	9x slower	1.6x slower	1.4x slower	14x faster	34x slower	1.7x slower	1.6x slower	8x slower	8x slower
	babel-runtime	1.8x slower	9x slower	1.5x slower	1.4x slower	13x faster	40x slower	1.5x slower	2.0x slower	7x slower	8x slower
	typescript	Identical	Identical	Identical	Identical	391x faster	Identical	Identical	Identical	Identical	Identical
	es6	Identical	Identical	Identical	2.4x slower	389x faster	Identical	1.4x slower	1.5x slower	Identical	Identical
defaults tests	babel	1.5x slower	1.5x slower	1.4x slower	1.5x slower	Identical	Identical	2.0x slower	575x slower	2.2x slower	10x slower
	typescript	Identical	Identical	Identical	Identical	Identical	Identical	Identical	Identical	Identical	Identical
	es6	Identical	Identical	Identical	3x slower	Identical	Identical	Identical	Identical	Identical	Identical
destructuring tests	babel	2.4x slower	1.3x slower	1.4x slower	1.8x slower	Identical	Identical	3x slower	3x slower	16x slower	1.3x slower
	babel-loose	Identical	Identical	Identical	Identical	Identical	Identical	Identical	1.3x slower	Identical	Identical
	babel-runtime	2.5x slower	1.4x slower	1.4x slower	1.4x slower	Identical	Identical	2.9x slower	4x slower	21x slower	7x slower
	typescript	Identical	Identical	Identical	Identical	Identical	Identical	1.2x faster	Identical	Identical	Identical
	es6	17x slower	15x slower	16x slower		83x slower	98x slower	20x slower	53x slower	9x slower	23x slower

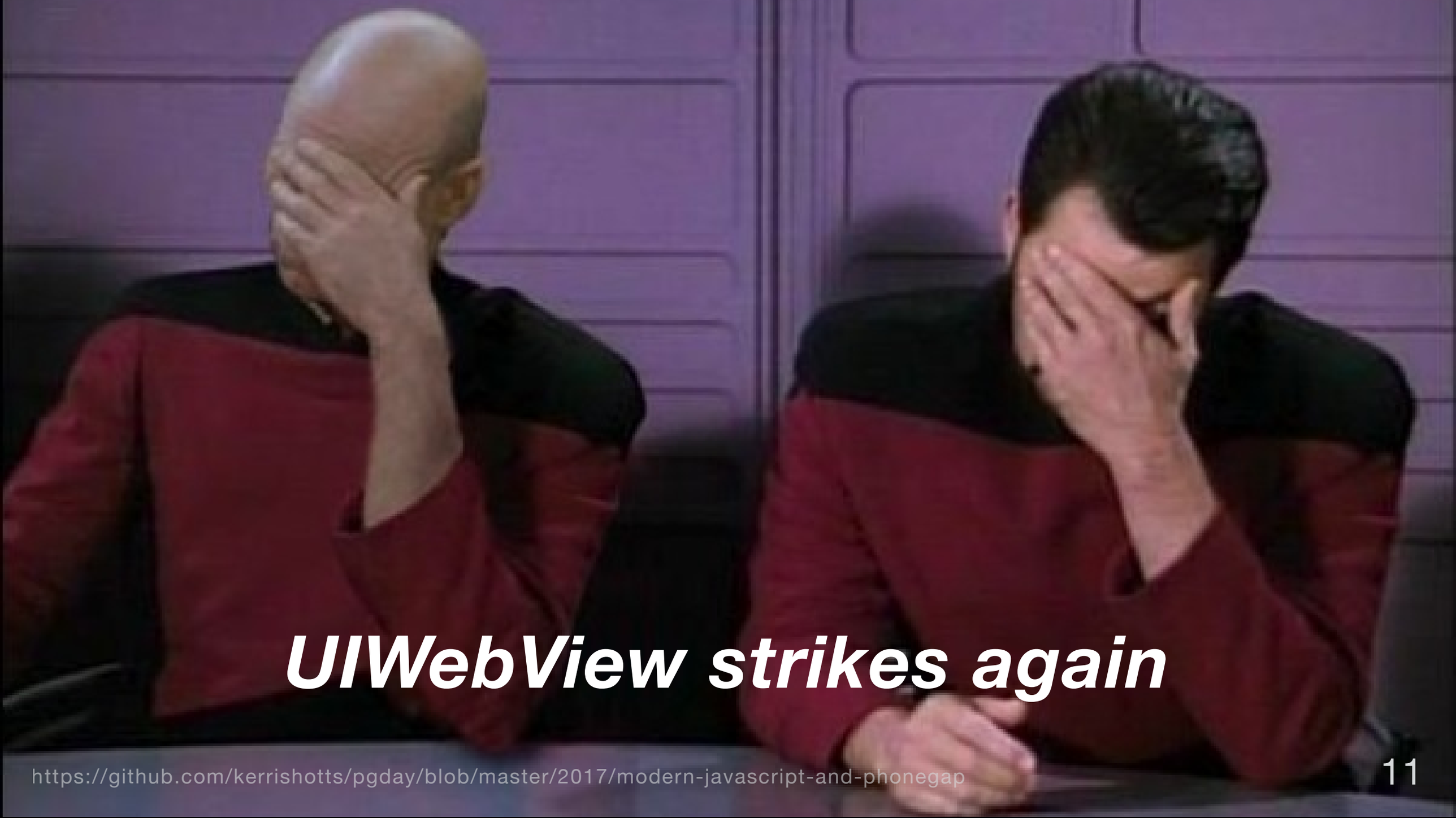
https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

Webviews & Performance

- **WKWebView** (iOS) single-core performance is **excellent**
- Compared to Safari on MBPr*
  - iPad Pro 12.9″: roughly equal
  - iPhone 6s: 2× slower
  - iPad Mini 4: 2.5× slower

- **Android Webview / Chrome** performance is **slow**; by roughly an order of magnitude
- Compared to Chrome on MBPr:
  - OnePlus One: ~10× slower
  - Samsung Tab S 8.4″: ~33× slower

Note: Results *highly sensitive* to the JavaScript features in use. *MacBook Pro: Late 2014, 2.2GHz i7 16GB RAM
Left Picture by TPSDave (Pixabay); Right Picture by leovalente (Pixabay)



UIWebView strikes again

Webviews & Performance (2)

- UIWebView: *ugh*
  - “Slower than molasses in January”
  - e.g: ~75× slower on an iPad Pro 12.9″
  - No JIT 😢

Note: Results *highly sensitive* to the JavaScript features in use. MacBook Pro: Late 2014, 2.2GHz i7 16GB RAM
Picture by Mhy (Pixabay)

Performance Takeaways

- Micro-benchmarks aren't the entire story
  - Performance is steadily improving
- Not "News" that most webviews are slow — true of ES5 too
  - Expected: iOS `WKWebView` is fast (JIT, faster single-core perf)
  - Expected: Android webviews aren't (slower single-core perf)
  - Expected: `UIWebView` is slower than a snail (no JIT)
- Frameworks are using ES2015 more
- More expressive & less boilerplate

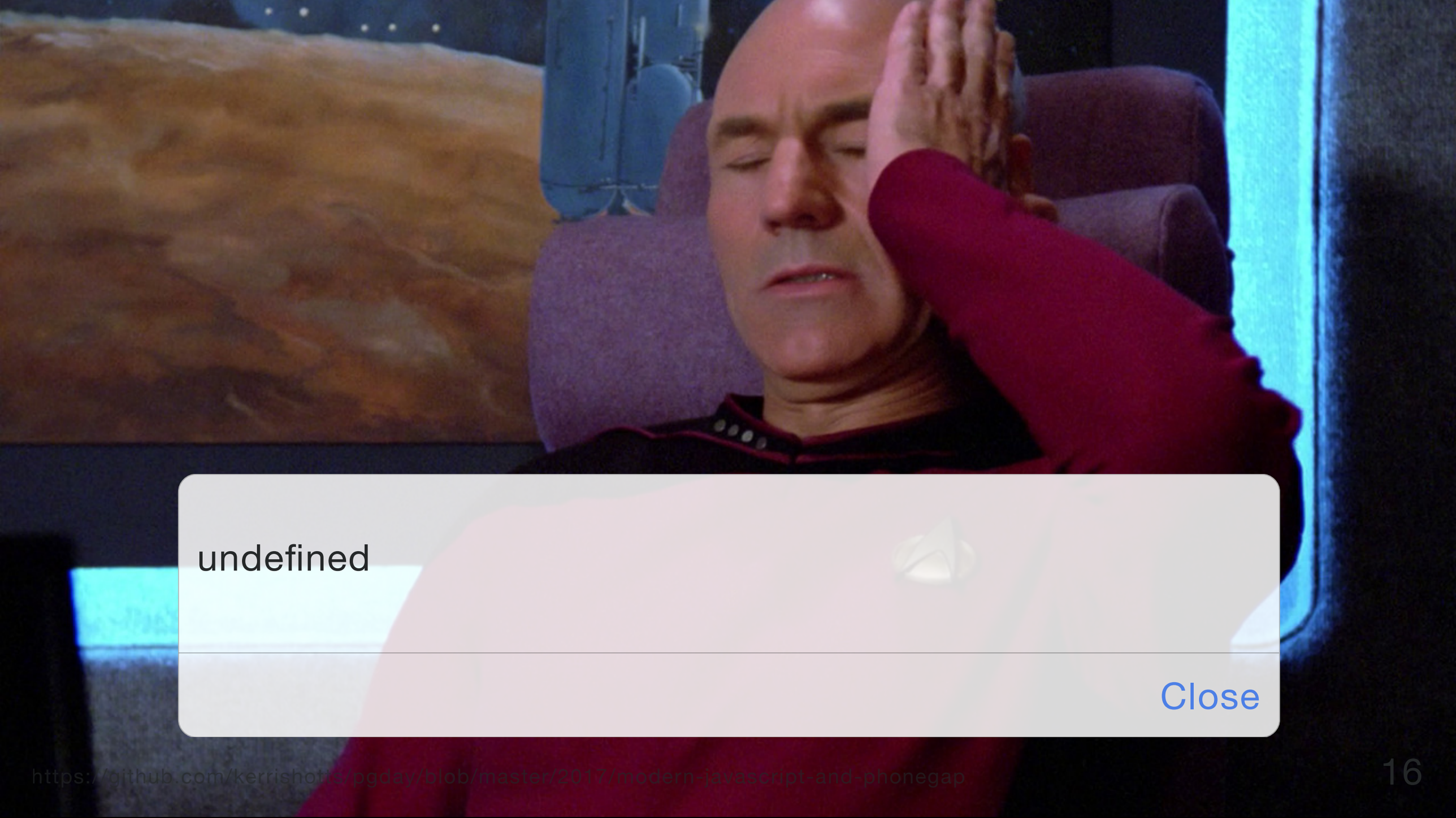
https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

A whirlwind tour

Dang it, *this!*

```
1   var app = {
2     text: "Hello, PG Day Attendees!",
3     sayHi: function() { alert(this.text); },
4     start: function() {
5       document.querySelector("#clickme")
6         .addEventListener("click", this.sayHi, false);
7     }
8   }
9
10  app.start();
```

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap



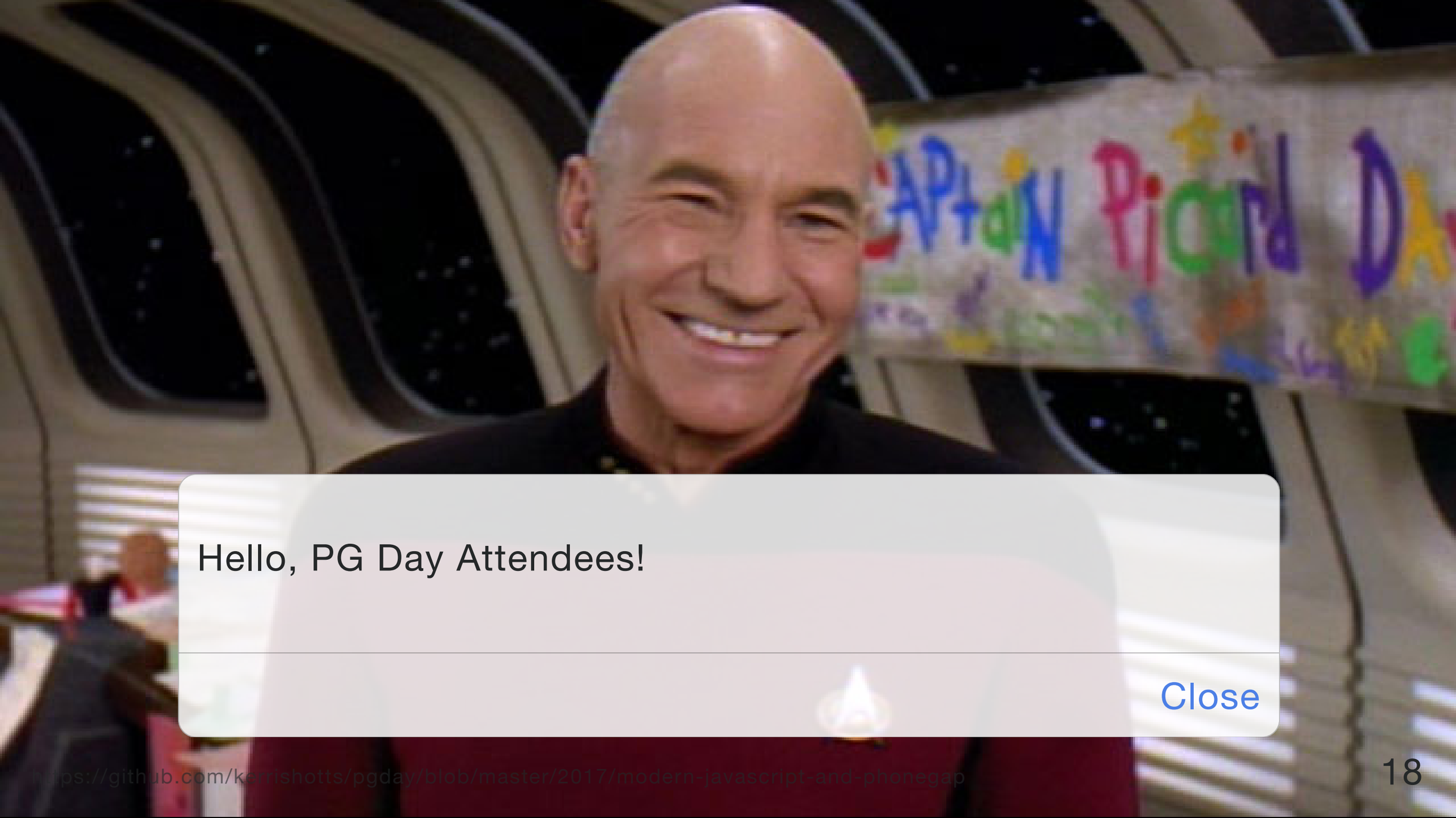
https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

Arrow functions (=>) & Classes

```
1  class App {
2    constructor() { this.text = "Hello, PG Day Attendees!"; }
3    sayHi() { alert(this.text); }
4    start() {
5      document.querySelector("#clickme")
6        .addEventListener("click",() => this.sayHi(), false);
7    }
8  }
9  const app = new App();
10 app.start();
```

Line 6 ES5 equivalent: `.addEventListener("click", (function() { this.sayHi(); }).bind(this), false)`

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap



https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

Array-like conversion

If only I had a € for every time I've written:

```
var elList = document.querySelectorAll("a"),
    elArr = [].slice.call(elList, 0);
```

ES2015+*:

```
let elArr = Array.from(document.querySelectorAll("a"));
```

* with the standard library

Spread/Rest is awesome (...)

Even shorter than `Array.from`:

```
let elArr = [...document.querySelectorAll("a")];
```

Easy variadic arguments:

```
function sum(start = 0, ...nums) {
  return nums.reduce((acc, val) => acc + val, start);
}
console.log(sum(1, 5, 10, 99)); /* 115 */
```

Destructuring

```
[a, b] = [b, a]  // swap!
```

“Multiple return values”:

```
function duplicate(str) {
  return {result: str + str,
          error: !str ? "no string" : null};
}
let {result, error} = someFunction("abc");
let {result:r, error:err} = someFunction("acb"); // you can rename
let {result} = someFunction("abc");              // or even ignore!
```

Named Parameters & Defaults

```
function getPicture({quality=50, width=512, height=512}={}) {
  return new Promise((resolve, reject) => {
    navigator.camera.getPicture(resolve, reject, {
      quality, targetWidth: width, targetHeight: height,
      allowEdit: false, correctOrientation: true });
  });
}
getPicture().then(/*...*/);
getPicture({quality:75}).then(/*...*/);
getPicture({height: 1024, width: 1024}).then(/*...*/)
```

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

Template Strings

```
function sayHello(name) {
  return `Hello, ${name}!`;
}
sayHello("World");  // Hello, World!
```

Complex expressions (use with care!):

```
function sayComplexHello(name) {
  return `Hello, ${name ? name : "Jane Doe"}!`;
}
```

Promises, promises

Hopefully already familiar to you...

```
function requestFileSystem({type = window.PERSISTENT,
                            quota = 5 * 1024 * 1024} = {}) {
  return new Promise((resolve, reject) => {
    window.requestFileSystem(type, quota, resolve, reject);
  });
}
```

async / await (ES2017)

```
async function getFile(name) {
  const fs = await requestFileSystem();
  return await fs.getFile(name);
}

async function start() {
  try {
    const file = await getFile("poem.txt"));
  } catch (err) {
    alert (`Error: ${err}`);
  }
}
```

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

Geolocation (ES2017)

```
function getPos(opts) {
  return new Promise((resolve, reject) => {
    navigator.geolocation.getCurrentPosition(resolve, reject, opts);
  });
}
async function start() {
  try {
    const {timestamp, coords:{latitude, longitude}} = await getPos();
    console.log(`At ${latitude}, ${longitude} on ${timestamp}`);
  } catch(err) {
    console.log(`Error ${err.code}: ${err.message}`);
  }
}
```

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

File Transfer (ES2017)

```
function uploadFile({source, target, options} = {}) {
  return new Promise((resolve, reject) => (new FileTransfer()).
    upload(url, to, resolve, reject, options));
}
async function start() {
  try {
   const {responseCode, response, bytesSent} = uploadFile({
      url: "cdvfile://localhost/persistent/test.txt",
      to:  "http://www.example.com/upload.php",
      options: { mimeType: "text/plain",
                 fileKey:  "file", fileName: "test" }});
  } catch (err) { /* do something with the error */ }
}
```

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

Modules

Static Analysis, FTW!

📄 math.js:

```
export function add(a, b) { return a+b; }
```

📄 index.js:

```
import {add} from "math.js";
console.log(add(4, 3));      // 7
```

Native support is a moving target

OS	ES2015	ES2016	ES2017
Android (Chrome)	97% (51+)	100% (55+)	53% (56+)
Edge 15	100%	100%	39%
Edge 14	93%	-	-
iOS 11*	100%	100%	98%
iOS 10	100%	61%	42%
iOS 9	54%	-	-

* Based on current status in Safari Technological Preview 11
Note: Some of the tests are based on existence, not completeness. **Sources**: ES2015, ES2016, ES2017

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

But, I want it everywhere!

ES2015: The Rise of the Transpilers

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

Common Transpilers

These can all transpile ES2015* (feature support may vary)

- Babel (née es6to5)
- TypeScript
- Bublé **
- Traceur

* **Note:** Not every ES2015+ feature can be transpiled effectively (if at all), such as proxies, shared memory, atomics, built-in subclassing, and tail call elimination. Also, most transpilers need core-js to polyfill the standard library.
** Doesn't attempt to transform non-performant or non-trivial ES6 features; *also very young*

Remember module support?

No Implementation! 😱

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

But we can fix that...

Module support using Bundling 🛍

Dependency management & `import` / `export` (and CommonJS, AMD, etc.) support

- Webpack
- JSPM
- Browserify

Execution Options

- Manual
  - Just run each tool's CLI… *every time*…
  - Error prone — you might forget!
- Automatic
  - Task runners (`gulp` or `grunt`; out-of-scope)
  - `npm` scripts
  - Plugin or Project hooks

Automating with npm scripts

- Pick your bundler and transpiler
  - Bundler: Webpack 2
  - Transpilers: TypeScript & Babel (showing both configs)
- Install Webpack & Transpiler
- Configure Webpack & Transpiler
- Add scripts to `package.json`

Install Webpack & Transpiler

```
[user@dev] $ npm install --save-dev webpack
```

Typescript:

```
[user@dev] $ npm install --save-dev ts-loader typescript core-js
```

Babel:

```
[user@dev] $ npm install --save-dev babel-loader babel-core babel-polyfill \
  babel-preset-es2015 babel-preset-es2016 babel-preset-es2017 \
  babel-plugin-transform-runtime
```

Note: `core-js` is a standard library polyfill; depending on your feature use and targets you may not need it.

Configure Transpiler

```
// tsconfig.json
{
  "compilerOptions": {
    "allowJs": true,
    "target": "es5",
    "module": "es2015", // DCR
    "lib": ["es2015", ...]
    "inlineSourceMap": true
  },
  "include":
    ["www(.src)/(es|ts)/**/*"]
}
```

```
// .babelrc
{
  "presets": [
    ["es2015", {
      "loose": true,
      "modules": false // DCR
    }],
    "es2016", "es2017"
  ],
  "plugins": ["transform-runtime"]
}
```

* Don't forget to import `core-js` (ts)/ `babel-polyfill` in your `index.?s` if targeting older runtimes. DCR = tree shaking

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

webpack.config.js

```
var src = path.resolve(__dirname, "www.src"),
    www = path.resolve(__dirname, "www"),
    loader = "ts-loader"; // or babel-loader
module.exports = {
  context: src, entry: "./es/index.js",
  devtool: "inline-source-map",
  output: { filename: "bundle.js", path: www + "/js" },
  module: {
    rules: [
      { test: /\.(ts|js|jsx)$/, exclude: /node_modules/,
         use: loader, options: { entryFileIsJs: true } //excl if babel
    } /*, ... other rules as needed */ ]}
}
```

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

Add Scripts

In package.json:

```
"scripts": {
  "build:ios": "webpack -d && cordova build ios",
  "build:release:ios": "webpack -p && cordova build ios --release",
  // below requires http-server & npm-run-all to be installed
  "_watch:webpack": "webpack -d -w",
  "_watch:serve": "http-server -p 8080 -c-1 www",
  "watch": "run-p _watch:*"
}
```

```
[user@dev] $ npm run build:ios
[user@dev] $ npm run watch
```

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

Automating with Plugin Hooks

cordova-plugin-webpack-transpiler transforms at `prepare` -time.

```
[user@dev] $ cordova plugin add --save                \
               cordova-plugin-webpack-transpiler      \
               --variable CONFIG=typescript|babel|...
```

Or, some useful templates (using project-level hooks):

- cordova-template-webpack-ts-scss
- cordova-template-webpack-babel-scss

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

What about...

- ... linting?
- ... tests?
- ... code coverage?

Linting

`eslint` works just fine with ES2015! (`tslint` for Typescript)

```
[user@dev] $ | npm install --save-dev eslint
```

`package.json`:

```
"scripts": {
    "lint": "eslint www.src"
}
```

```
[user@dev] $ | npm run lint     # or, write a plugin /
             |                  # project-level hook! ;-)
```

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

Tests

```
[user@dev] $ npm install --save-dev mocha chai
[user@dev] $ npm install --save-dev ts-node        # for TypeScript
[user@dev] $ npm install --save-dev babel-register # for Babel
```

Add test to package.json:scripts*

```
"test": "mocha" // TypeScript (needs ./test/_bootstrap.js)
"test": "mocha --compilers js:babel-register"     // Babel
```

Then npm test

* Assumes tests are in `./test`
_bootstrap.js: `require("ts-node").register();`

Code coverage (Babel)

First, `npm install --save-dev instanbul cross-env nyc`

```
{ // .babelrc
  "presets": ["es2015", ...],
  "plugins": [...],
  "env": {
    "cover": {
      "plugins": ["transform-es2015-modules-commonjs",
                  "istanbul"]
    }
  }
}
```

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

Code coverage (Babel, 2)

```
// package.json
"nyc": {
  "require": ["babel-register"],
  "reporter": ["text", "html"],
  "sourceMap": false,
  "instrument": false
},
"scripts": { //...,
  "cover": "cross-env NODE_ENV=cover nyc npm test"
}
```

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

Tips

Tips

- ES5 still works; adjust to ES2015+ at your own speed
- `var` hasn't gone away; it's OK to use it!
  - Use where performance is critical (e.g., tight nested loops)
- Try to declare `const` / `let` at the top of each scope
  - Benefits Chrome's optimizer
  - Good practice anyway
- Arrow functions aren't necessarily drop-in replacements
  - Can be risky with `describe` & `it` in your tests

Tips (2)

- Minify & remove dead code for release builds
  - Reduces bundle sizes and startup time
- Split code bundles
  - Vendor code can be separately bundled
  - Easier to blacklist in debuggers

That’s all, folks!

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

Thanks!

@kerrishotts

https://github.com/kerrishotts/pgday/blob/master/2017/modern-javascript-and-phonegap

This slide intentionally left blank