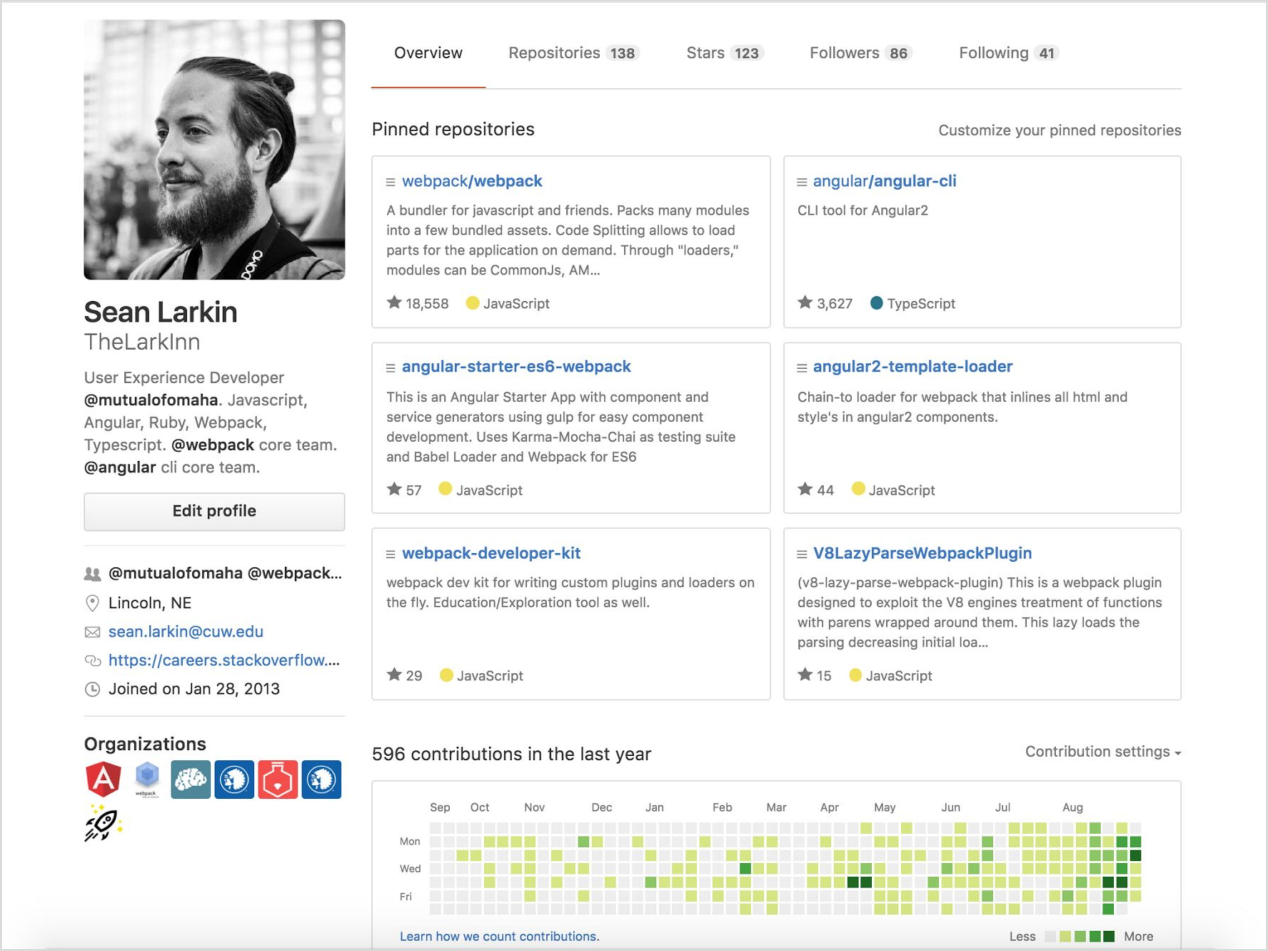
webpack: One Build Step To Rule Them All

Sean Larkin: webpack core team | @thelarkinn

QCon
SAN FRANCISCO



Sean Larkin

TheLarkInn

User Experience Developer **@mutualofomaha**. Javascript, Angular, Ruby, Webpack, Typescript. **@webpack** core team. **@angular** cli core team.

@mutualofomaha @webpack...

Lincoln, NE

sean.larkin@cuw.edu

https://careers.stackoverflow....

Joined on Jan 28, 2013

Organizations

Pinned repositories

webpack/webpack

A bundler for javascript and friends. Packs many modules into a few bundled assets. Code Splitting allows to load parts for the application on demand. Through "loaders," modules can be CommonJs, AM...

★ 18,558 ● JavaScript

angular/angular-cli

CLI tool for Angular2

★ 3,627 ● TypeScript

angular-starter-es6-webpack

This is an Angular Starter App with component and service generators using gulp for easy component development. Uses Karma-Mocha-Chai as testing suite and Babel Loader and Webpack for ES6

★ 57 ● JavaScript

angular2-template-loader

Chain-to loader for webpack that inlines all html and style's in angular2 components.

★ 44 ● JavaScript

webpack-developer-kit

webpack dev kit for writing custom plugins and loaders on the fly. Education/Exploration tool as well.

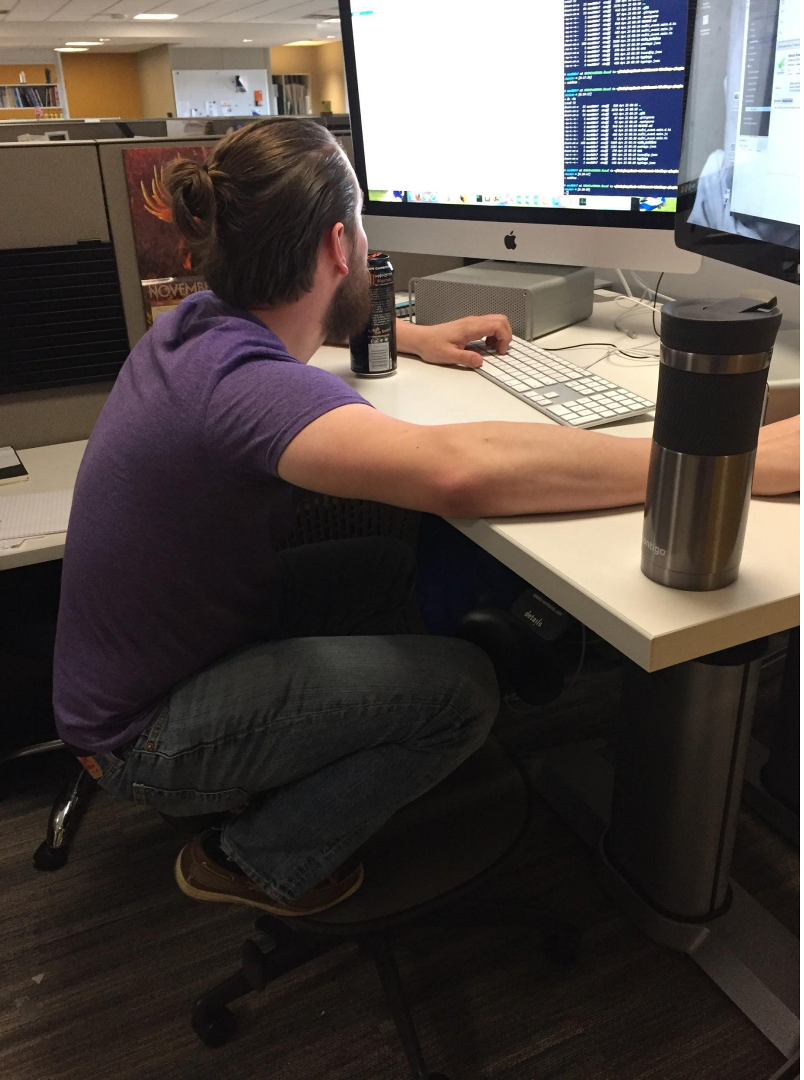
★ 29 ● JavaScript

V8LazyParseWebpackPlugin

(v8-lazy-parse-webpack-plugin) This is a webpack plugin designed to exploit the V8 engines treatment of functions with parens wrapped around them. This lazy loads the parsing decreasing initial loa...

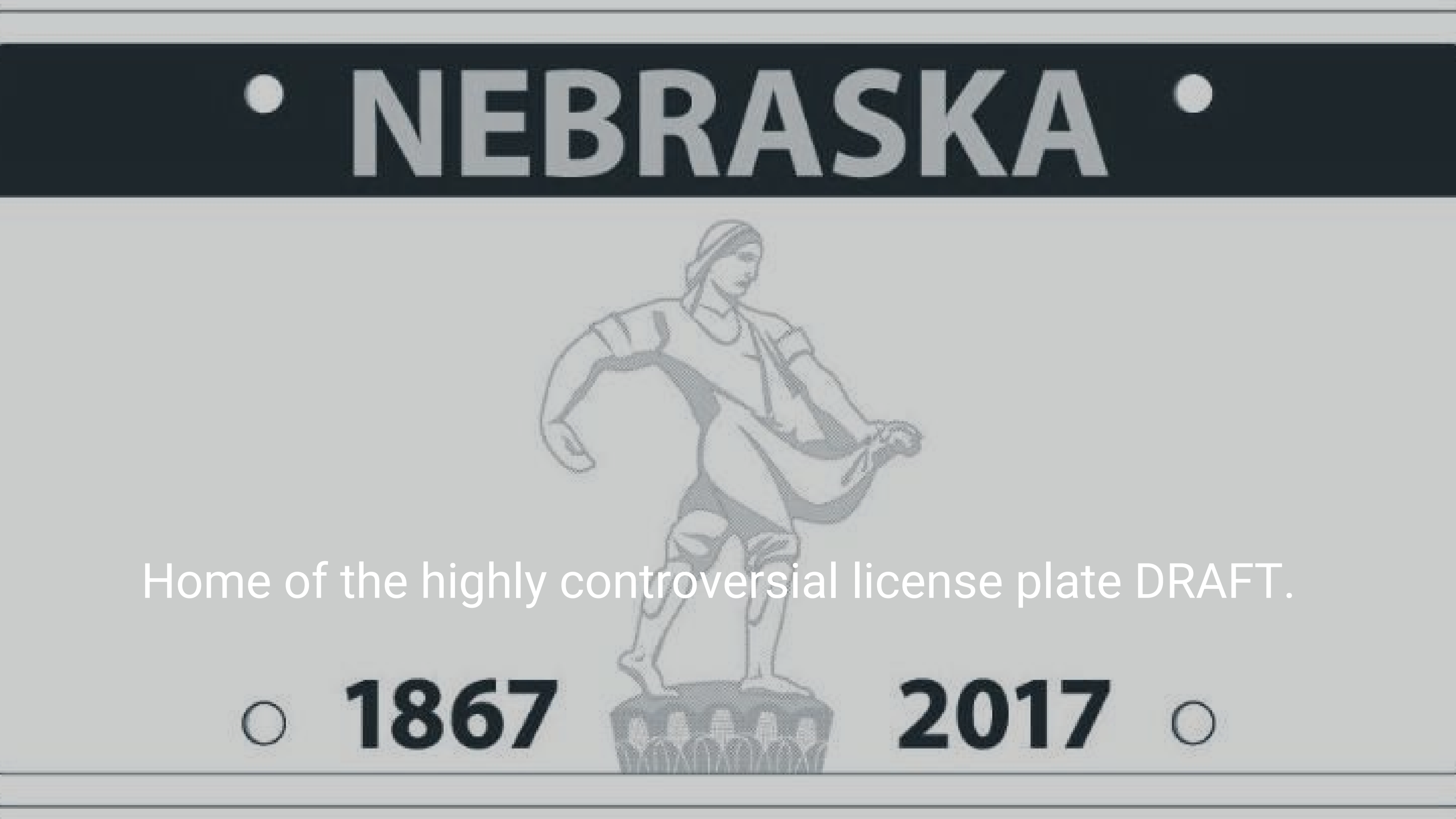
★ 15 ● JavaScript

596 contributions in the last year

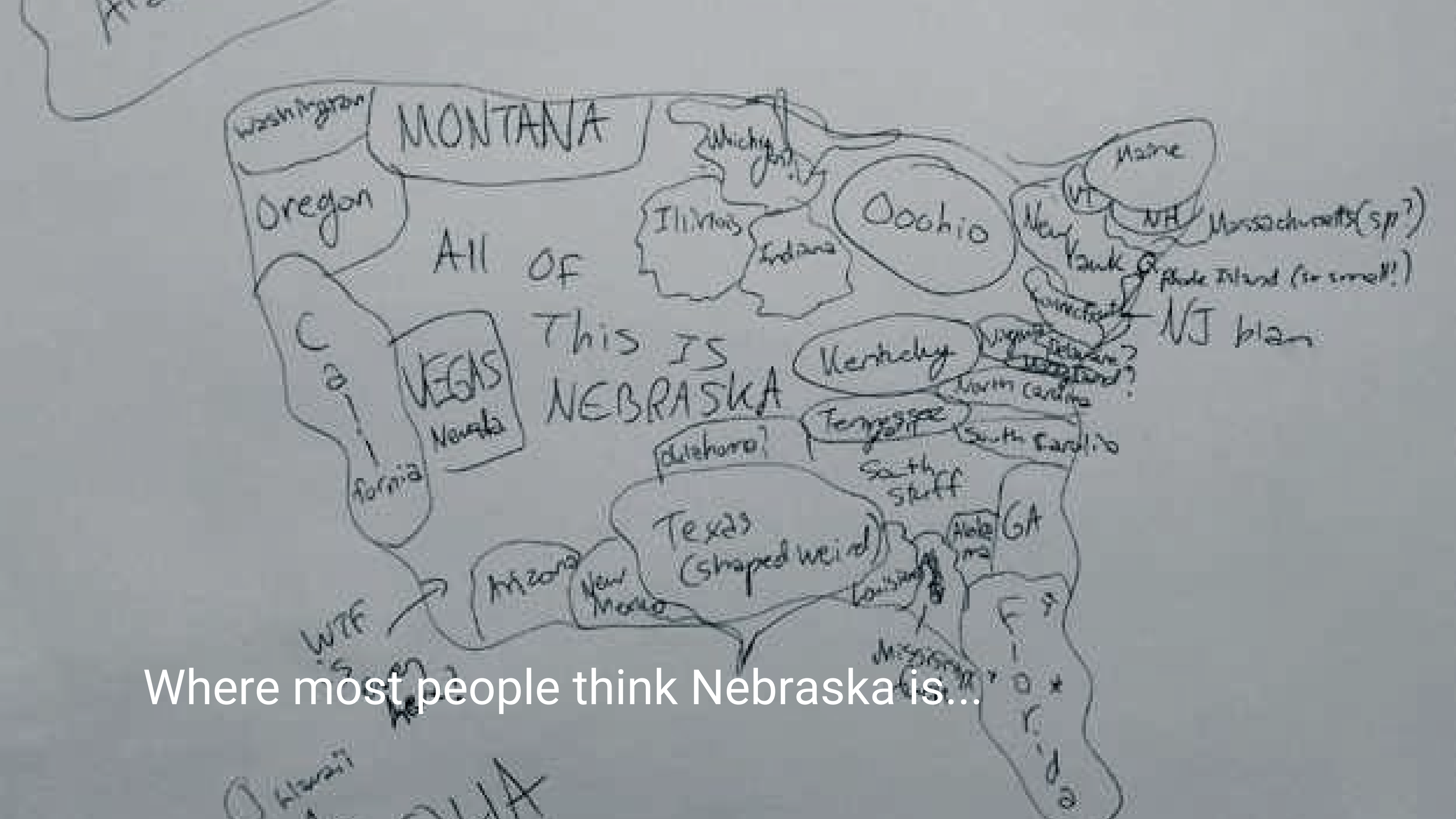


UX Developer @
Mutual of Omaha

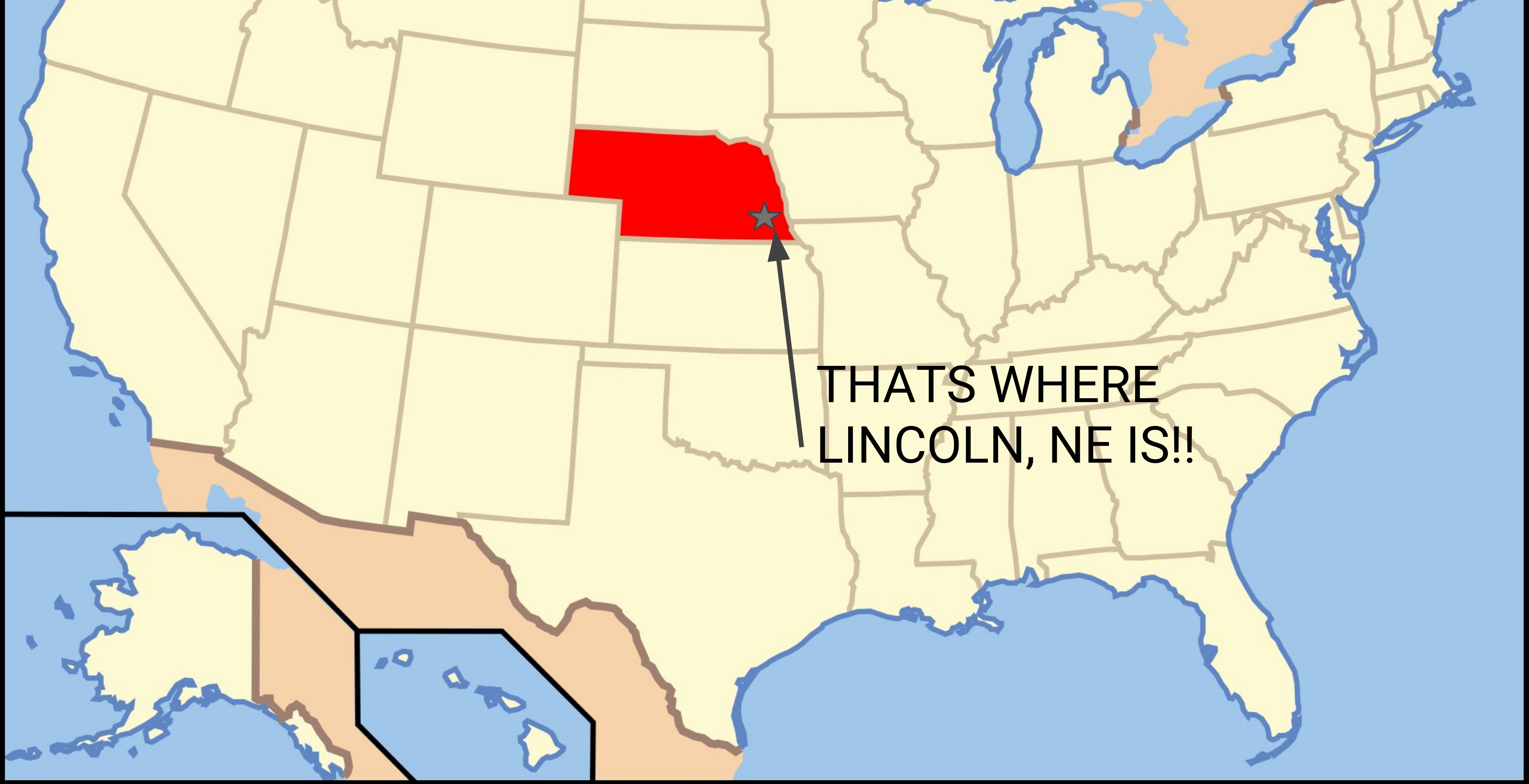
Come work here!



Home of the highly controversial license plate DRAFT.



Where most people think Nebraska is...



THATS WHERE
LINCOLN, NE IS!!

Where Nebraska really is...

Background

Former Tech Support Rep. gone rogue turned Software Engineer / Web Developer who got tired of never being able to really help the customer he served.

Written in…

Languages: Ruby, Objective-C, Swift, Javascript.

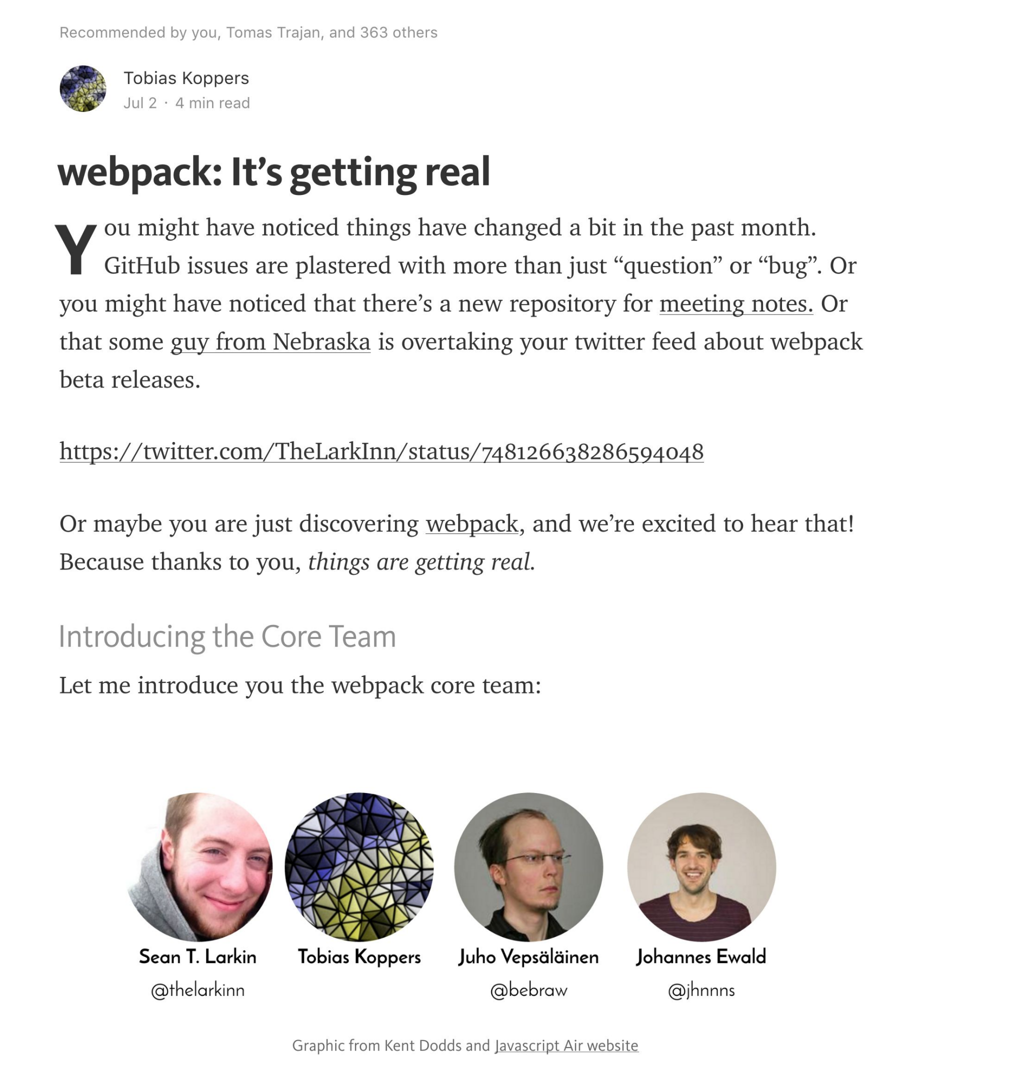
Also…

Woodworker, chicken farmer, IoT.

Projects

webpack core team

angular-cli core team



Recommended by you, Tomas Trajan, and 363 others

Tobias Koppers
Jul 2 · 4 min read

webpack: It’s getting real

You might have noticed things have changed a bit in the past month. GitHub issues are plastered with more than just “question” or “bug”. Or you might have noticed that there’s a new repository for meeting notes. Or that some guy from Nebraska is overtaking your twitter feed about webpack beta releases.

https://twitter.com/TheLarkInn/status/748126638286594048

Or maybe you are just discovering webpack, and we’re excited to hear that! Because thanks to you, *things are getting real.*

Introducing the Core Team

Let me introduce you the webpack core team:

Sean T. Larkin	Tobias Koppers	Juho Vepsäläinen	Johannes Ewald
@thelarkinn		@bebraw	@jhnnns

Graphic from Kent Dodds and Javascript Air website

@TheLarkInn

Github - Medium - Codepen - Stack Overflow - LinkedIn - Twitter

"Watching @TheLarkInn @qconsf teach me about #webpack has changed my life. #webpackAllTheThings"

ASK ME ANYTHING

http://github.com/thelarkinn/ama

JavaScript Modules

Qualities of JavaScript Modules:

Don’t pollute global scope

Reusable

Encapsulated

Organized

Convenient

How to use them?

Script Tag

Global Variable

Namespace (require/import)

JavaScript Modules...

What they look like...

CommonJS

```
//loading module
var _ = require(‘lodash’);

//declaring module
module.exports = someValue;
```

AMD

```
define(‘myAwesomeLib’, [‘lodash’,
‘someDep’],
   function (_, someDep) {
     return { … }
   }
);
```

AMD + CommonJS

```
define( function(require, exports, module) {
  var _ = require(‘lodash’);

  //..do things
  module.exports = someLib;
});
```

ES2015/TypeScript

```
import {Component} from ‘angular2/core’

@Component({
  selector:’info’
})
export class InfoComponent{}
```

So let’s talk about making them work together… ...for the browser.

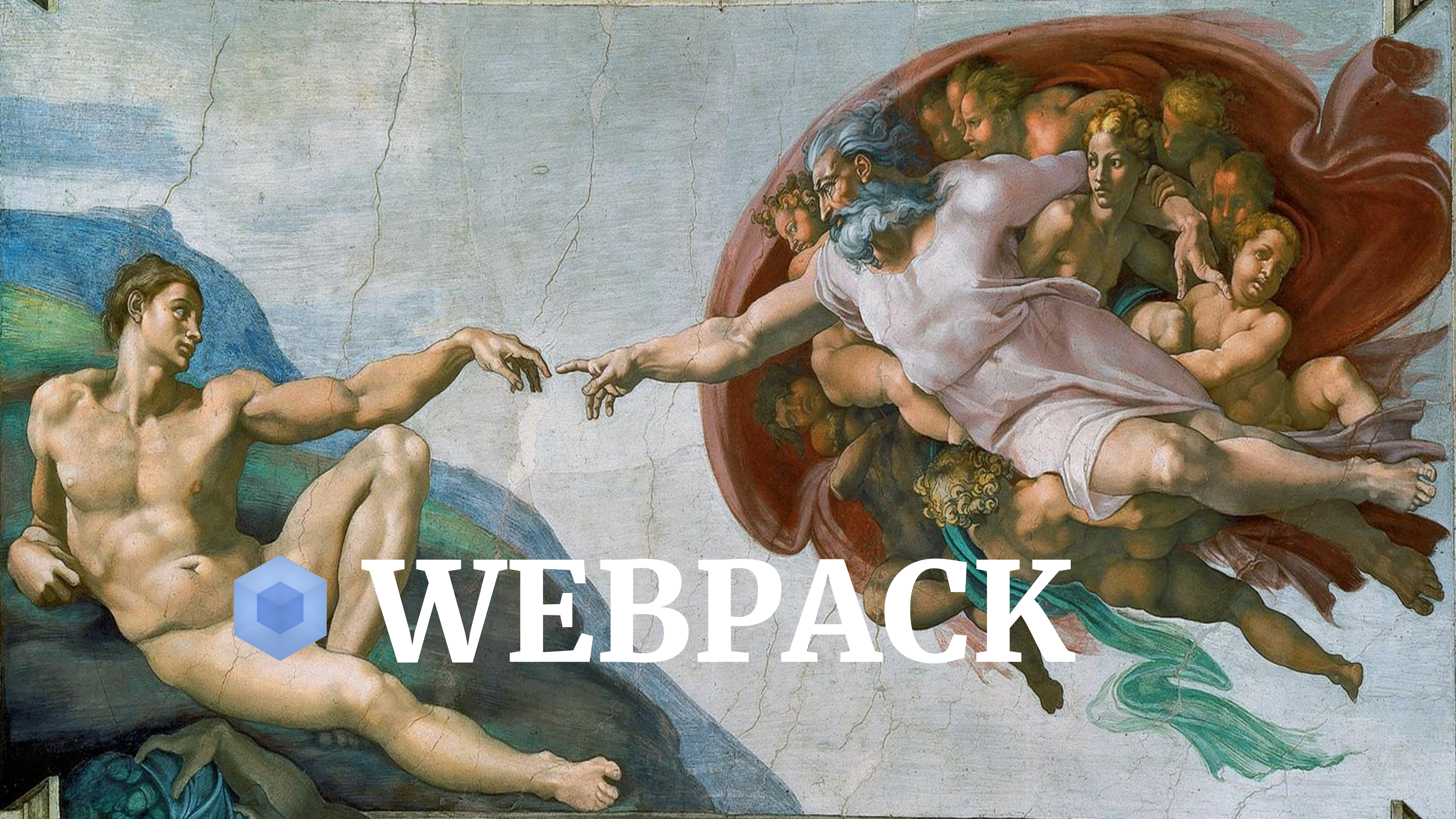
Every library is different...

And has their own loading requirements…
And their own module “shape”...
And their own way of using them in the “browser”

And this is just for JavaScript...

Each and every other filetype until now has had to have specific ways to process it.

Wouldn’t it be nice...



WEBPACK

Webpack is a module **bundler** and *not a task runner*.

Every asset is a dependency of another (js, css, html, jpg, icon, svg, etc...).

Static build tool (NOT A MODULE LOADER)!

But how?

Webpack - How to use it?

webpack.config.js

Yes, its a module too!!!

```
module.exports = {
  entry: {
    vendor: './src/vendors.ts',
    main: './src/main.browser.ts'
  },
  output: {
      path: 'dist/',
      filename: '[name].bundle.js',
      sourceMapFilename: '[name].map',
      chunkFilename: '[id].chunk.js'
  },
  resolve: {
    extensions: ['.ts', '.js'],
    modules: ['node_modules']
  },
  module: {
      {
        enforce: 'pre',
        test: /\.js$/,
        loader: 'source-map-loader',
        exclude: [
          // these packages have problems with their sourcemaps
          root('node_modules/rxjs')
        ]
      }

    ],
    loaders: [
```

Webpack - How to use it?

Webpack CLI

```
$> webpack <entry.js>
<result.js> --colors
--progress

$> webpack-dev-server
--port=9000
```

Webpack - How to use it?

Node API

```
var webpack = require("webpack");

// returns a Compiler instance
webpack({
   // configuration object here!
}, function(err, stats) {
  // …
  // compilerCallback
  console.error(err);
});
```

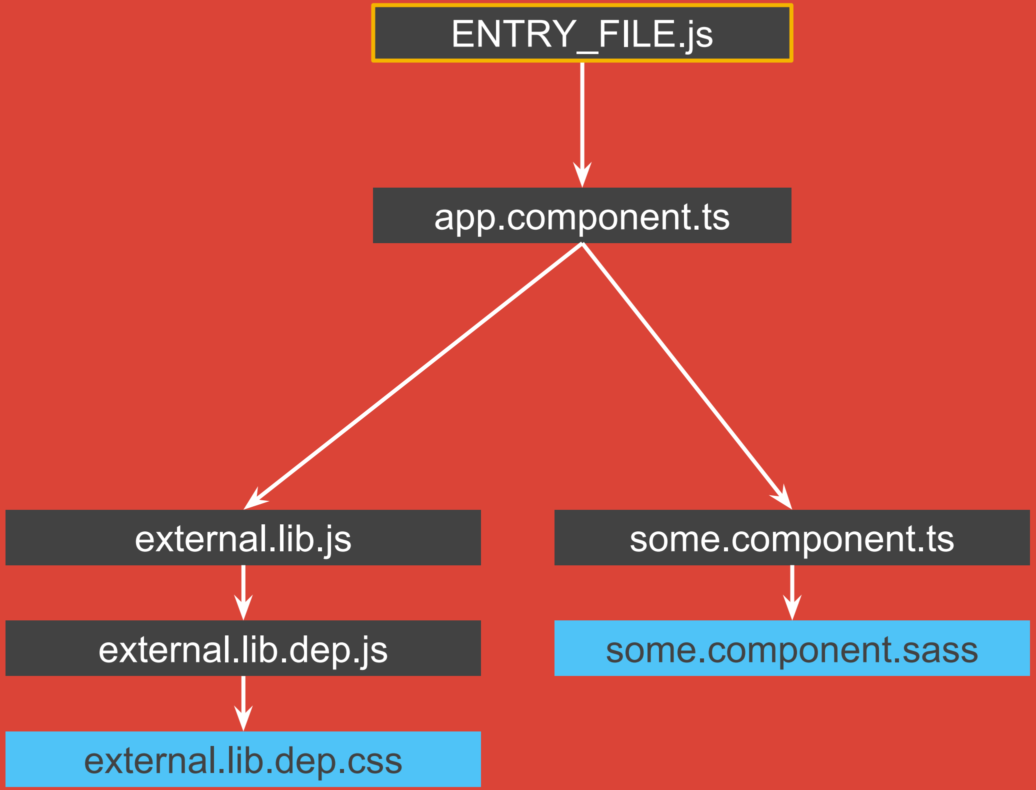
The Core Concepts

Entry

The Core Concepts: Entry

The “contextual root” of your application.

The first javascript file to load to “kick-off” your app in the browser.



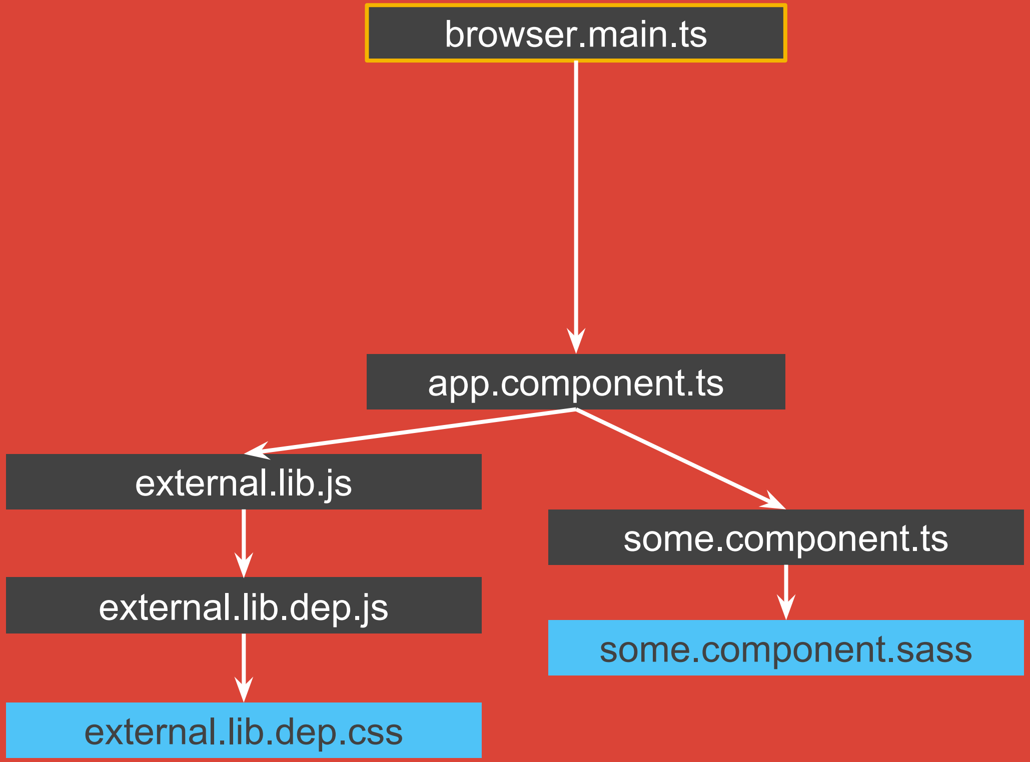
The Core Concepts: Entry

```
//webpack.config.js
module.exports = {
  entry: ‘./browser.main.ts’,
  //...
}

//browser.main.ts
import {
  Component
} from ‘@angular/core’;

import {
  App
} from ‘./app.component’;
bootstrap(App,[]);

//app.component.ts
@Component({...})
export class App {};
```



The Core Concepts: Entry

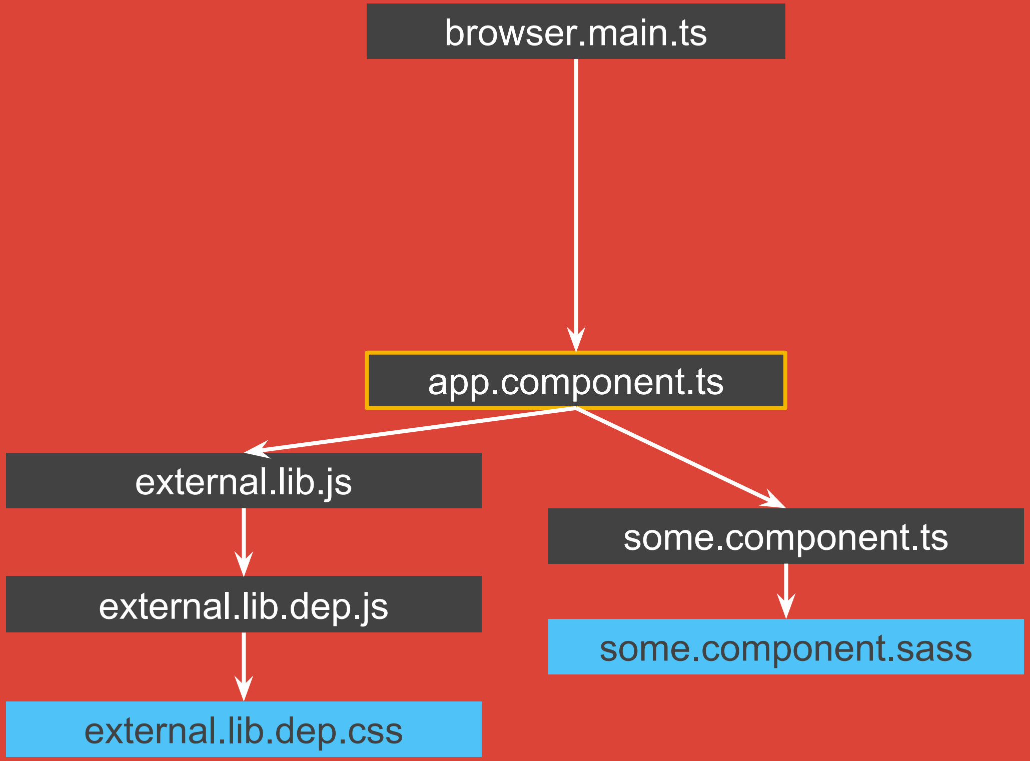
```
//webpack.config.js
module.exports = {
  entry: ‘./browser.main.ts’,
  //...
}

//browser.main.ts
import {Component} from
‘@angular/core’;

import {App} from
‘./app.component’;

bootstrap(App,[]);

//app.component.ts
@Component({...})
export class App {};
```



The Core Concepts

Entry

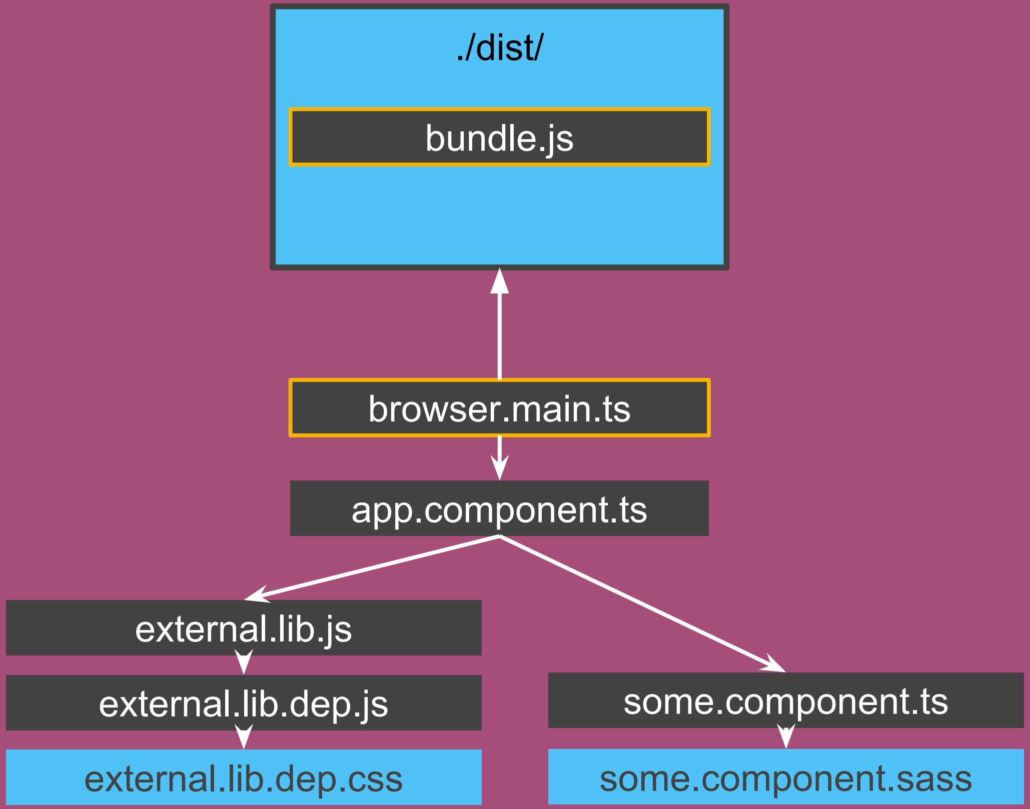
Tells webpack ***WHAT*** (files) to load for the browser; Compliments the *Output* property.

Output

The Core Concepts: Output

```
//webpack.config.js
module.exports = {
  entry: ‘./browser.main.ts’,
  output: {
    path: ‘./dist’,
    filename: ‘./bundle.js’,
  },
  //...
}

//Generates bundle.js
```



./dist/

bundle.js

browser.main.ts

app.component.ts

external.lib.js

external.lib.dep.js

external.lib.dep.css

some.component.ts

some.component.sass

The Core Concepts

Entry

Output

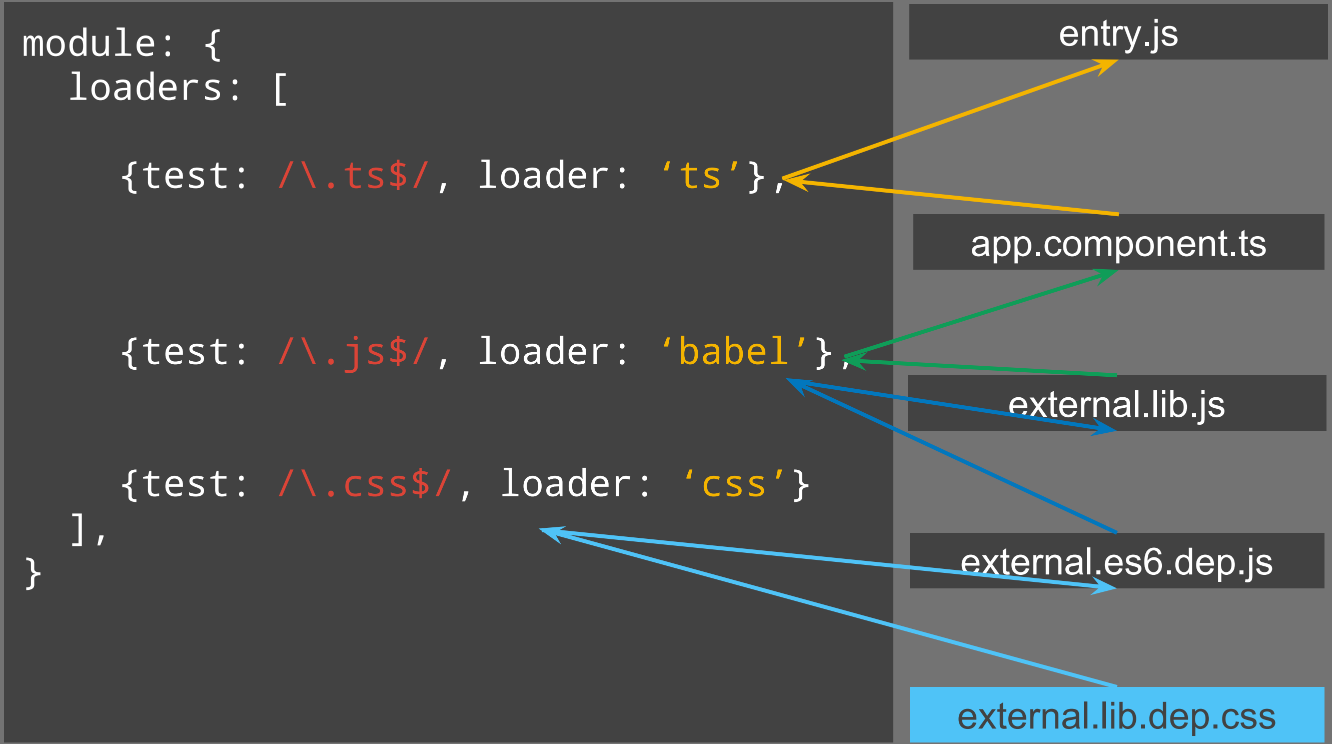
Tells Webpack WHERE and HOW to distribute bundles (compilations). Works with Entry.

Loaders

The Core Concepts: Loaders

Tells webpack how to load files in your content base.

Loaders are also javascript modules (*function*) that takes the source file, and returns it in a ‘loaded’ [modified] state.



```
module: {
  loaders: [

    {test: /\.ts$/, loader: ‘ts’},


    {test: /\.js$/, loader: ‘babel’},


    {test: /\.css$/, loader: ‘css’}
  ],
}
```

The Core Concepts: Loaders

```
module: {
  preLoaders:[], //lint
  loaders: [
    {
      test: regex,
      loader: string,
      loaders: Array<string>,
      include: Array<regex>,
      exclude: Array<regex>,
    },
  ],
  postLoaders:[] //coverage, docs, etc.
}
```

test

A *regular expression* that instructs the compiler which files to run the loader against.

loader

A *string* of the loader names you want to run.

loaders

An *array of strings* representing the modules you want to run. If using ‘*loader*’, provide a *string* of multiple loaders separated by ‘!’.

IE: ‘style!css!less`

The Core Concepts: Loaders

```
module: {
  preLoaders:[], //lint
  loaders: [
    {
      test: /\.ts$/,
      loader: string,
      loaders: [
        ‘awesome-typescript-loader’,
        ‘ng2-asset-loader`
      ],
      include: /some_dir_name/,
      exclude: [/\.(spec|e2e)\.ts$/],
    },
  ],
  postLoaders:[] //coverage, docs, etc.
}
```

include

An *array of regular expression* that instruct the compiler which folders/files to include. *Will only search paths provided with the include.*

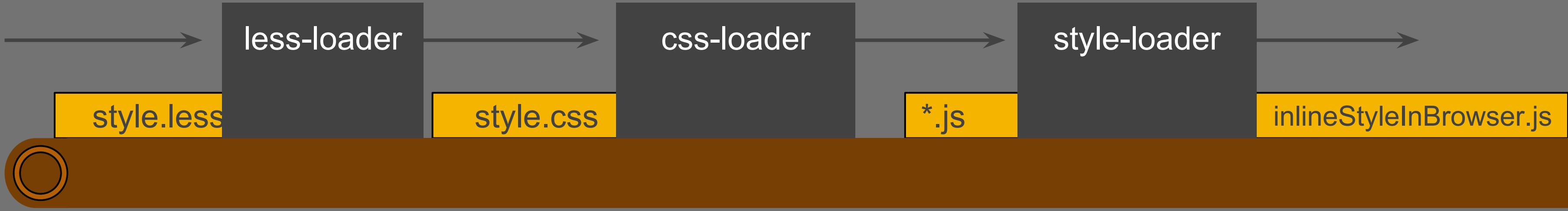
exclude

An *array of regular expression* that instructs the compiler which folders/files to ignore.

The Core Concepts: Loaders

Chaining Loaders

```
loaders: [
  { test: /\.less$/, loader:’style!css!less’ },
  { test: /\.less$/,
    loaders:[‘style’, ‘css’, less’]}
]
```



The Core Concepts: Loaders

json, hson, raw, val, to-string, imports, exports, expose, script, apply, callback, ifdef-loader, source-map, sourceMappingURL, checksum, cowsay, dsv, glsl, glsl-template, render-placement, xml, svg-react, svg-url, svg-as-symbol, symbol, base64, ng-annotate, node, required, icons, markup-inline, block-loader, bundler-configuration, console, solc, .sol, web3, includes, combine, regexp-replace, file, url, extract, worker, shared-worker, serviceworker, bundle, require.ensure, promise, async-module, bundle, require.ensure, react-proxy, react-hot, image, file, url, img, base64-image, responsive, srcset, svgo, svg-sprite, symbol, svg-fill, fill, line-art, baggage, polymer, uglify, html-minify, vue, tojson, zip-it, file, lzstring, modernizr, s3, path-replace, react-intl, require.ensure, font-subset, w3c-manifest, web-app-manifest, manifest-scope, coffee, coffee-jsx, coffee-redux, json5, es6, esnext, babel, regenerator, livescript, sweetjs, traceur, ts, typescript, awesome-typescript, webpack-typescript, purs, oj, elm-webpack, miel, wisp, sibilant, ion, html, dom, riot, pug, jade-html, jade-react, virtual-jade, virtual-dom, template-html, handlebars, handlebars-template-loader, dust, ractive, jsx, react-templates, em, ejs, ejs-html, mustache, yaml, yml, react-markdown, front-matter, markdown, remarkable, markdown-it, markdownattrs, ng-cache, ngtemplate, hamlc, haml, jinja, nunjucks, soy, smarty, swagger, template-string, ect, tmodjs, layout, swig, twig, mjml-, bootstrap-webpack, font-awesome-webpack, bootstrap-sass, bootstrap, bootstrap, font-awesome, style, isomorphic-style, style-loader, css, cess, less, sass, stylus, csso, rework, postcss, autoprefixer, namespace-css, fontgen, classnames, theo, bulma, css-to-string, css-loader, po, po2mo, format-message, jsxlate, angular-gettext, json, angular-gettext, webpack-angular-translate, angular-gettext-extract, .pot, gettext, preprocessor, amdi18n-loader, .json, .js, .coffee, sprockets-preloader, properties, transifex, mocha, coverjs, istanbul-instrumenter, ibrik-instrumenter, eslint, jshint, jscs, standard, inject, transform, falafel, image-size, csslint, coffeelint, tslint, parker, sjsp, amdcheck, manifest, gulp-rev, html-test, stylelint, stylefmt, scsslint, htmlhint, documentation, sassdoc, performance-loader

The Core Concepts

Entry

Output

Loaders

Tells Webpack HOW to interpret and translate files. They return *compilations.*

Plugins

The Core Concepts: Plugins

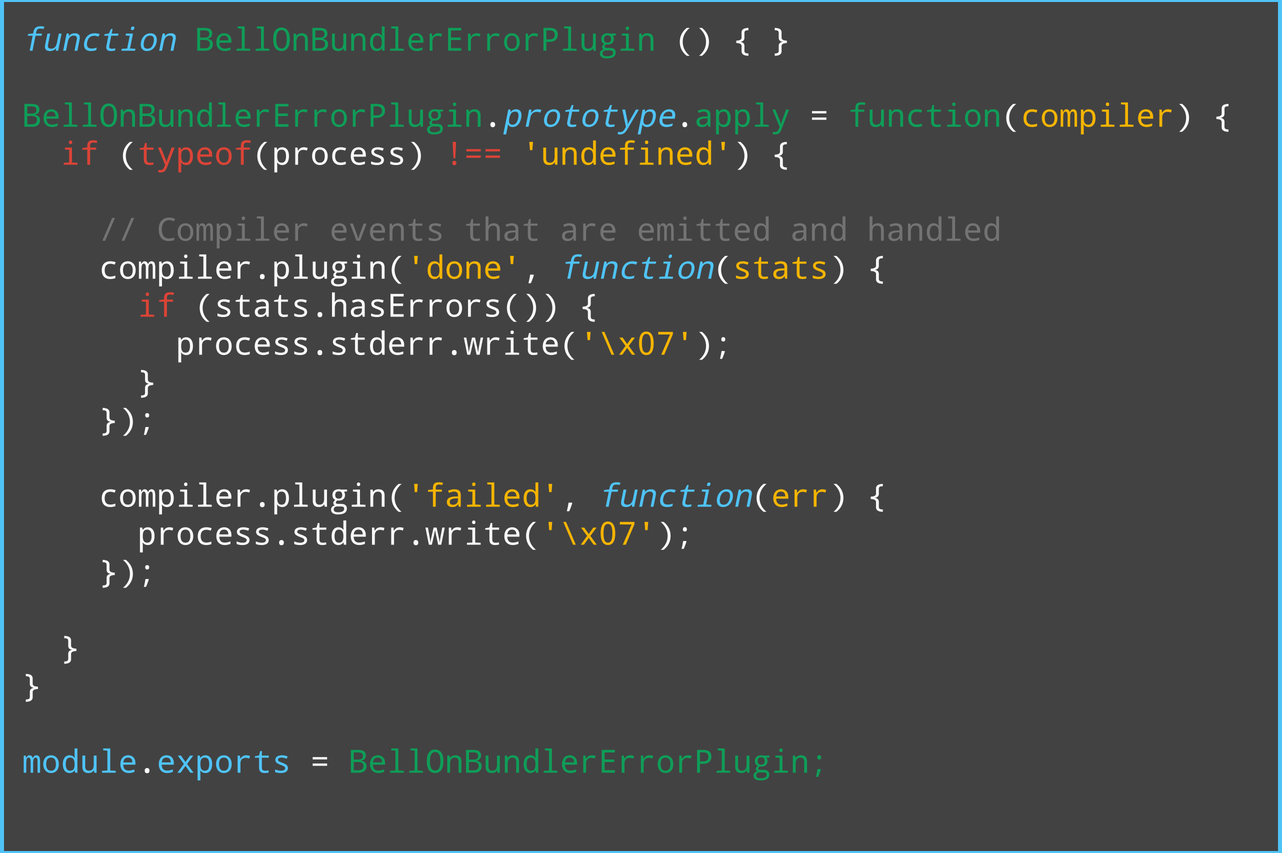
ES5 Classes

Apply functionality at the *compilation* level.

A *compilation* is a bundle of files processed by the webpack compiler. (Processed via loaders).

Webpack has a variety of built in plugins.

The Core Concepts: Plugins



```
function BellOnBundlerErrorPlugin () { }

BellOnBundlerErrorPlugin.prototype.apply = function(compiler) {
 if (typeof(process) !== 'undefined') {

  // Compiler events that are emitted and handled
  compiler.plugin('done', function(stats) {
   if (stats.hasErrors()) {
    process.stderr.write('\x07');
   }
  });

  compiler.plugin('failed', function(err) {
   process.stderr.write('\x07');
  });

 }
}

module.exports = BellOnBundlerErrorPlugin;
```

Basic Plugin Example

A plugin is an ES5 ‘class’ which implements an *apply* function.

The compiler uses it to emit events.

The Core Concepts: Plugins

```
// require() from node_modules or webpack or local file
var BellOnBundlerErrorPlugin = require(‘bell-on-error’);
var webpack = require(‘webpack’);

module.exports = {
  //...
  plugins: [
      new BellOnBundlerErrorPlugin(),

      // Just a few of the built in plugins
      new webpack.optimize.CommonsChunkPlugin(‘vendors’),
      new webpack.optimize.UglifyJsPlugin()
  ]
  //...
}
```

How to use Plugins

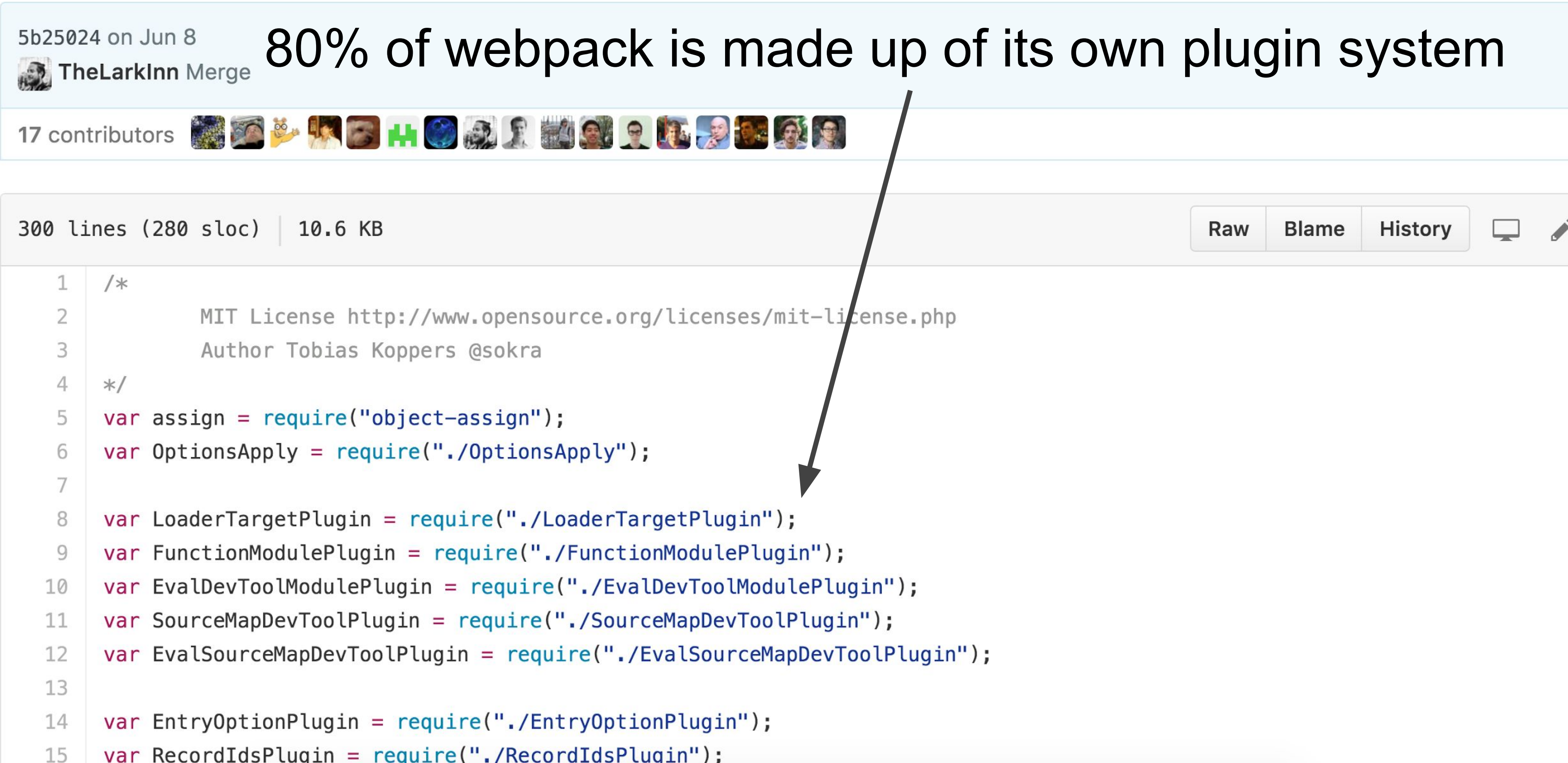
require() plugin from *node_modules* into config.

add ***new instance** of plugin to plugins* key in config object.

provide additional info for arguments

CLICK HERE TO SEE THE LIST OF PLUGINS

The Core Concepts: Plugins



80% of webpack is made up of its own plugin system

5b25024 on Jun 8

TheLarkInn Merge

17 contributors

300 lines (280 sloc) | 10.6 KB

```
/*
	MIT License http://www.opensource.org/licenses/mit-license.php
	Author Tobias Koppers @sokra
*/
var assign = require("object-assign");
var OptionsApply = require("./OptionsApply");

var LoaderTargetPlugin = require("./LoaderTargetPlugin");
var FunctionModulePlugin = require("./FunctionModulePlugin");
var EvalDevToolModulePlugin = require("./EvalDevToolModulePlugin");
var SourceMapDevToolPlugin = require("./SourceMapDevToolPlugin");
var EvalSourceMapDevToolPlugin = require("./EvalSourceMapDevToolPlugin");

var EntryOptionPlugin = require("./EntryOptionPlugin");
var RecordIdsPlugin = require("./RecordIdsPlugin");
```

The Core Concepts

Entry

Output

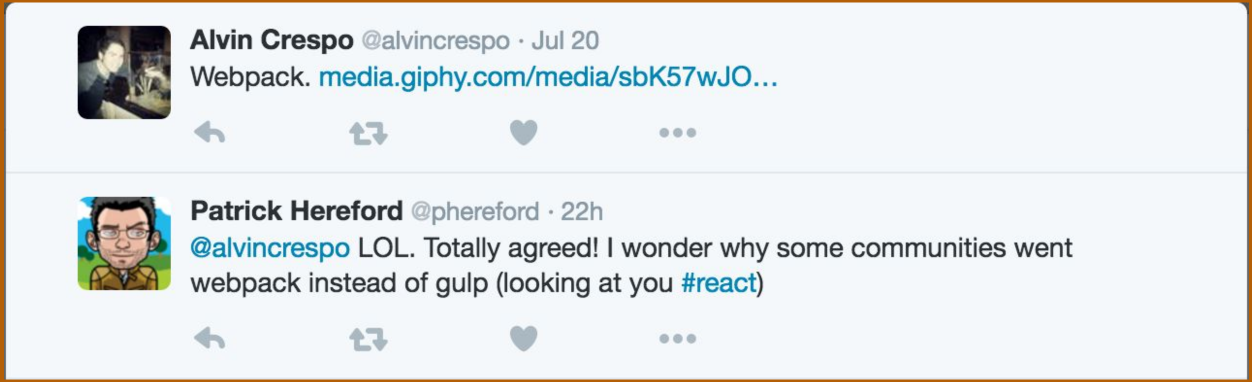
Loaders

Plugins

Adds additional functionality to *Compilations(optimized bundled modules)*. More powerful w/ more access to CompilerAPI. Does everything else you'd ever want to in webpack.

FAQ: Clear the Air

“Why use Webpack when I have grunt and gulp?



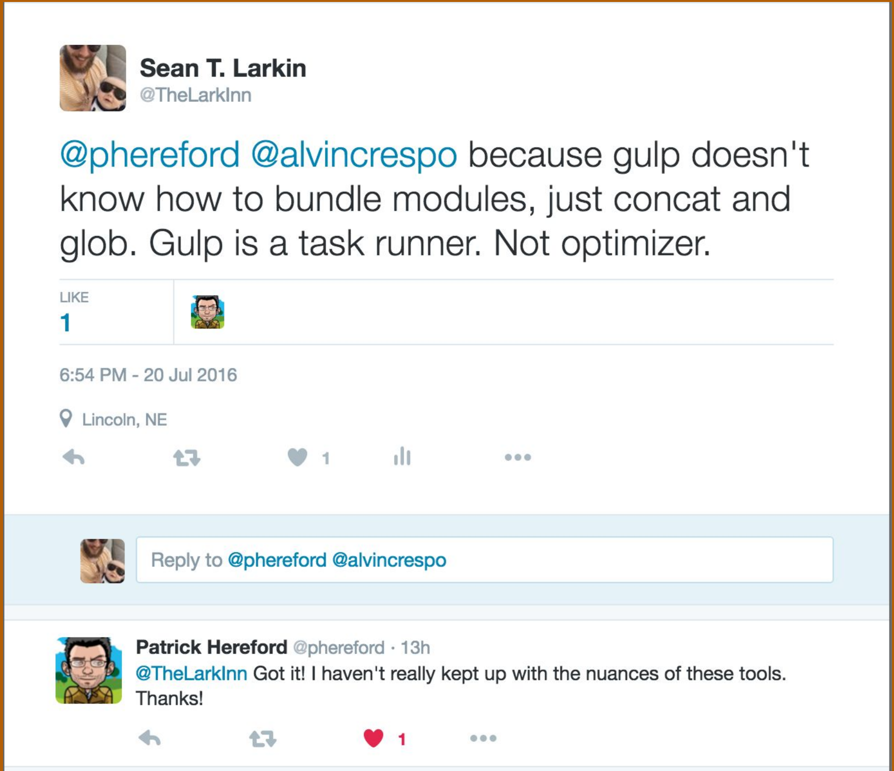
Alvin Crespo @alvincrespo · Jul 20

Webpack. media.giphy.com/media/sbK57wJO…

Patrick Hereford @phereford · 22h

@alvincrespo LOL. Totally agreed! I wonder why some communities went webpack instead of gulp (looking at you #react)

FAQ: Clear the Air



Sean T. Larkin
@TheLarkInn

@phereford @alvincrespo because gulp doesn't know how to bundle modules, just concat and glob. Gulp is a task runner. Not optimizer.

LIKE
1

6:54 PM - 20 Jul 2016

Lincoln, NE

1

Reply to @phereford @alvincrespo

Patrick Hereford @phereford · 13h
@TheLarkInn Got it! I haven't really kept up with the nuances of these tools. Thanks!

1

JS

MINIFIERS

LINTERS

COMPILE-TO-JS LANGUAGES

...

CSS

SASS

LESS

UNCSS

POSTCSS

...

ASSETS

GIFSICLE

PNGCRUSH

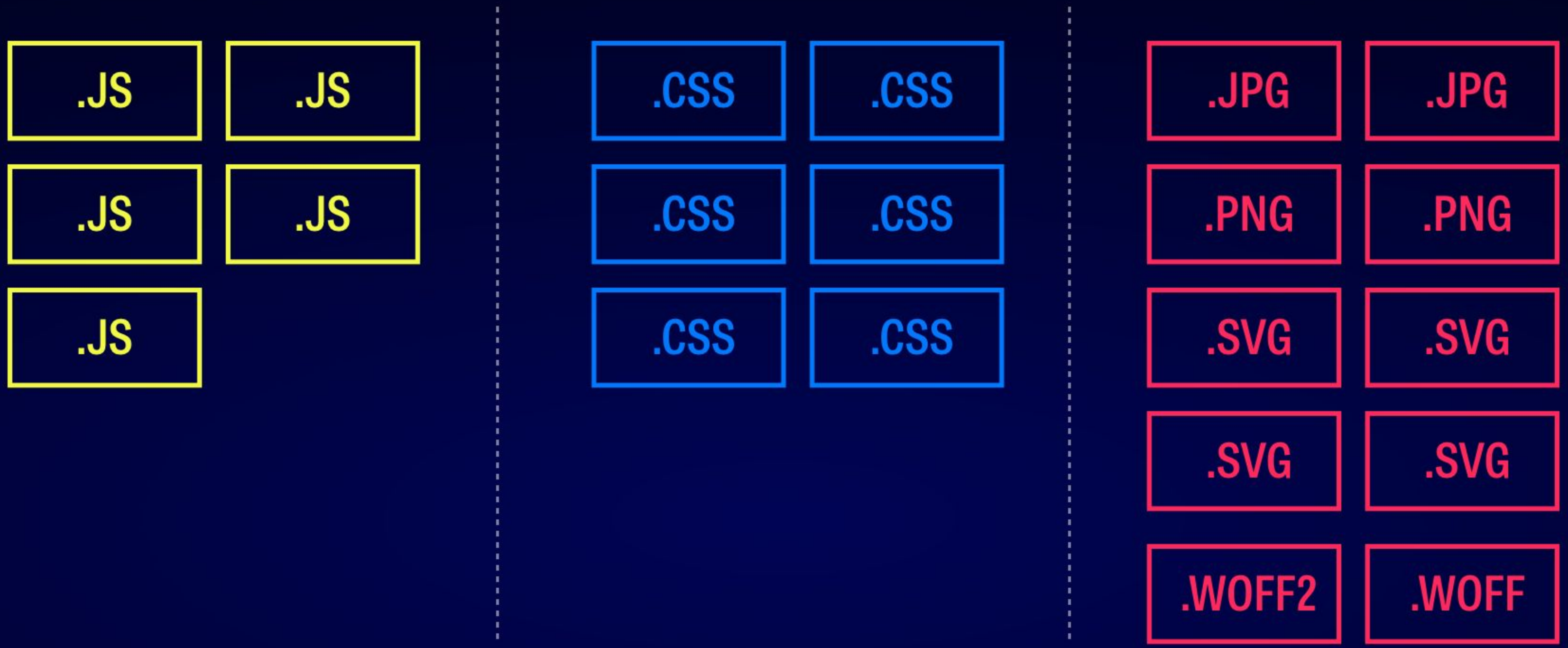
JPEGTRAN

SVGO

...

Thanks Glen Maddern! @glenmaddern (frontend.center)

GRUNT / GULP / ETC



.JS .JS

.JS .JS

.JS

.CSS .CSS

.CSS .CSS

.CSS .CSS

.JPG .JPG

.PNG .PNG

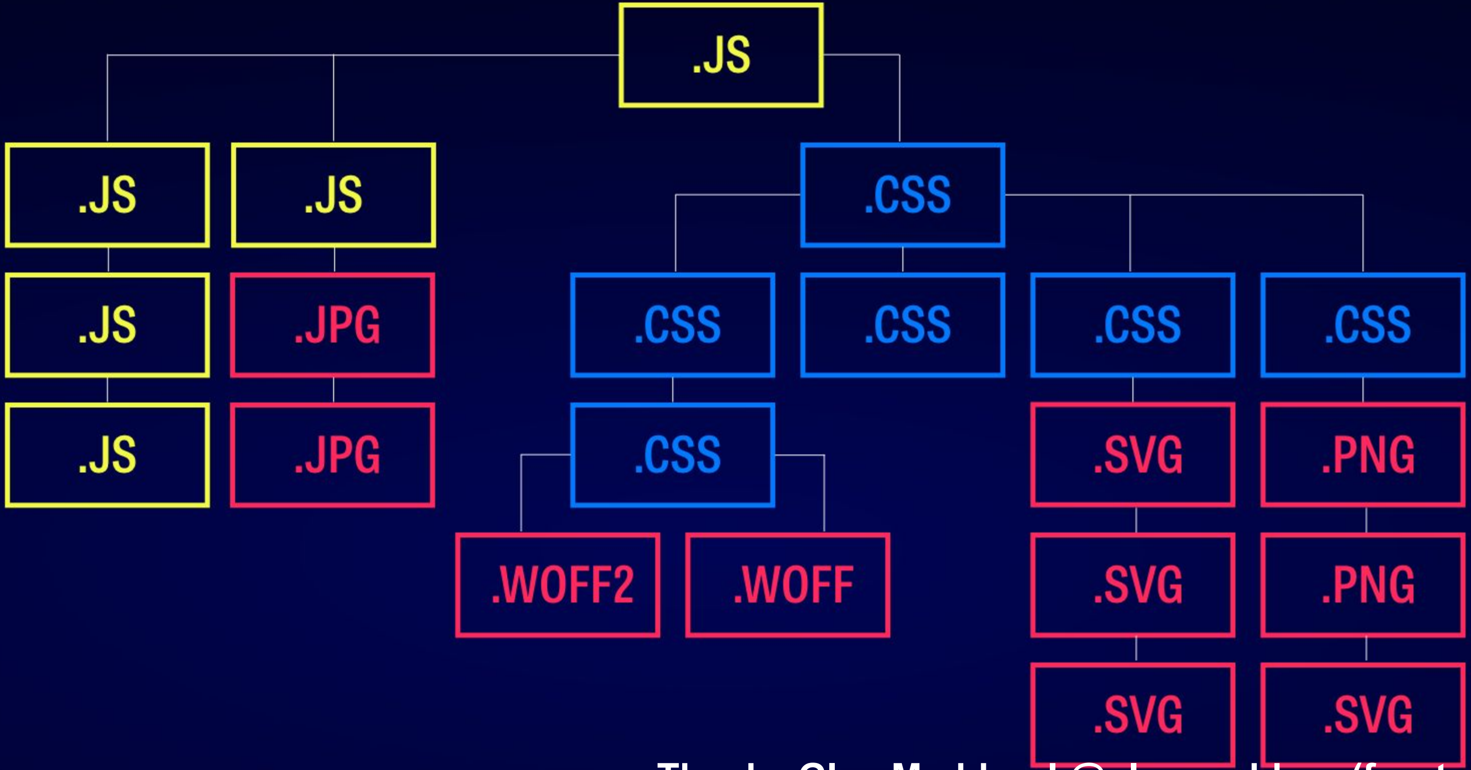
.SVG .SVG

.SVG .SVG

.WOFF2 .WOFF

Thanks Glen Maddern! @glenmaddern (frontend.center)

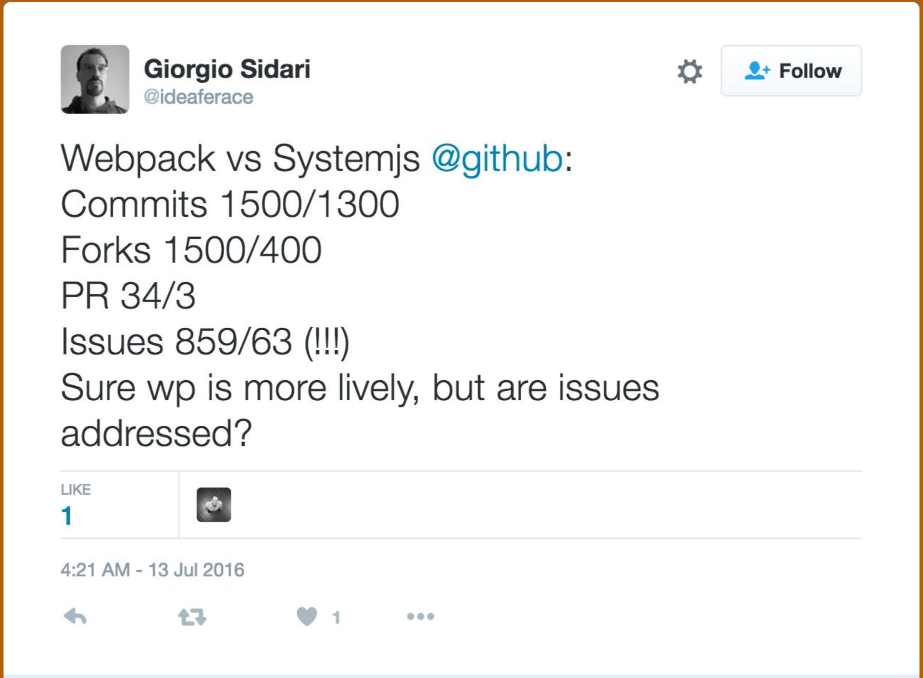
WEBPACK



Thanks Glen Maddern! @glenmaddern (frontend.center)

FAQ: Clear the Air

What's better, webpack or SystemJS?



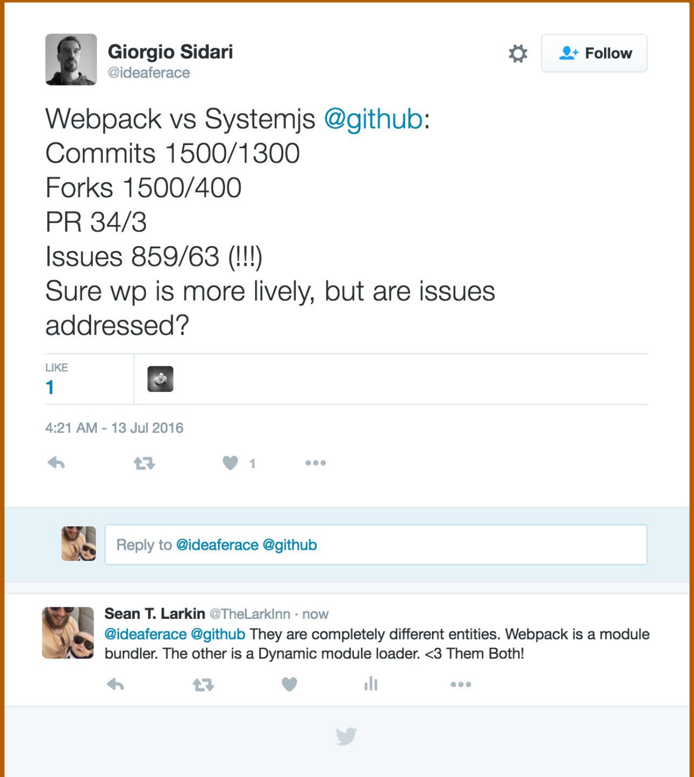
Giorgio Sidari
@ideaferace

Webpack vs Systemjs @github:
Commits 1500/1300
Forks 1500/400
PR 34/3
Issues 859/63 (!!!)
Sure wp is more lively, but are issues addressed?

4:21 AM - 13 Jul 2016

FAQ: Clear the Air

You still need to bundle your code, no server *(even with http2)* can handle the overhead of 500 assets/modules asynchronously being sent to the client.



Giorgio Sidari @ideaferace

Webpack vs Systemjs @github:
Commits 1500/1300
Forks 1500/400
PR 34/3
Issues 859/63 (!!!)
Sure wp is more lively, but are issues addressed?

LIKE
1

4:21 AM - 13 Jul 2016

Reply to @ideaferace @github

Sean T. Larkin @TheLarkInn · now
@ideaferace @github They are completely different entities. Webpack is a module bundler. The other is a Dynamic module loader. <3 Them Both!

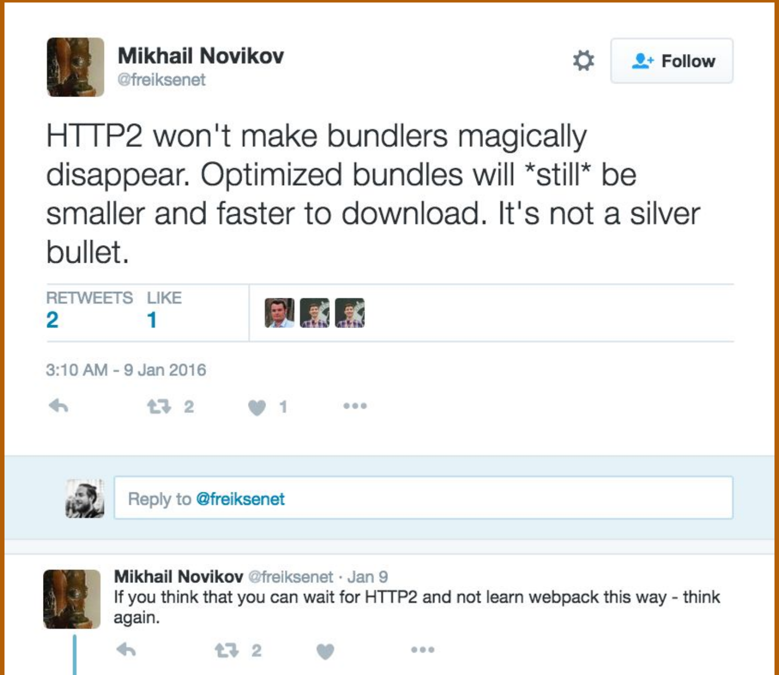
However it's better to compare SystemBundler or JSPM vs Webpack.

FAQ: Clear the Air

But HTTP2 will fix everything!

FAQ: Clear the Air

WRONG!



Mikhail Novikov
@freiksenet

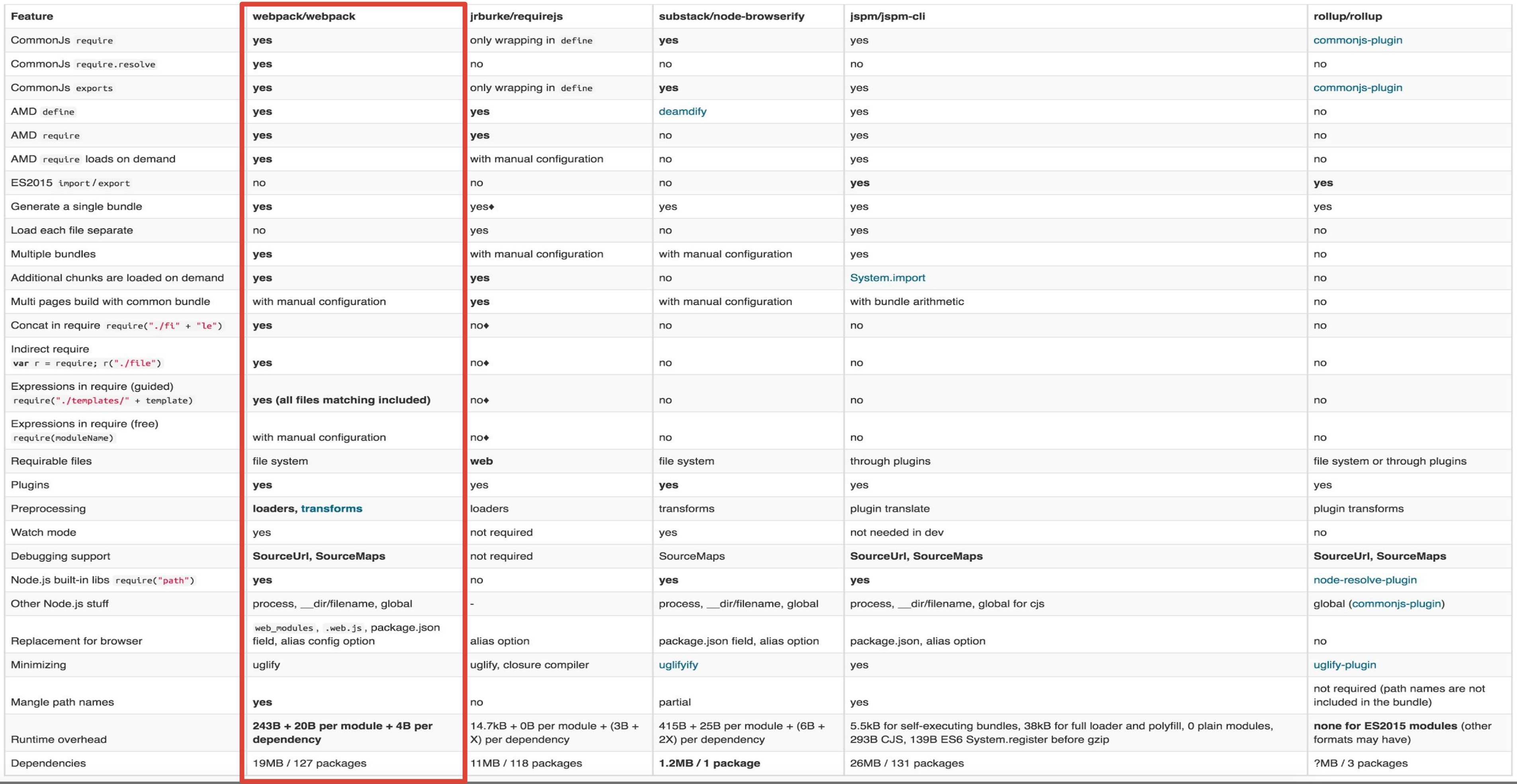
HTTP2 won't make bundlers magically disappear. Optimized bundles will *still* be smaller and faster to download. It's not a silver bullet.

RETWEETS 2 LIKE 1

3:10 AM - 9 Jan 2016

Mikhail Novikov @freiksenet · Jan 9
If you think that you can wait for HTTP2 and not learn webpack this way - think again.

Comparing the features...



Feature	webpack/webpack	jrburke/requirejs	substack/node-browserify	jspm/jspm-cli	rollup/rollup
CommonJs `require`	**yes**	only wrapping in `define`	**yes**	yes	commonjs-plugin
CommonJs `require.resolve`	**yes**	no	no	no	no
CommonJs `exports`	**yes**	only wrapping in `define`	**yes**	yes	commonjs-plugin
AMD `define`	**yes**	**yes**	deamdify	yes	no
AMD `require`	**yes**	**yes**	no	yes	no
AMD `require` loads on demand	**yes**	with manual configuration	no	yes	no
ES2015 `import` / `export`	no	no	no	**yes**	**yes**
Generate a single bundle	**yes**	yes♦	yes	yes	yes
Load each file separate	no	yes	no	yes	no
Multiple bundles	**yes**	with manual configuration	with manual configuration	yes	no
Additional chunks are loaded on demand	**yes**	**yes**	no	System.import	no
Multi pages build with common bundle	with manual configuration	**yes**	with manual configuration	with bundle arithmetic	no
Concat in require `require("./fi" + "le")`	**yes**	no♦	no	no	no
Indirect require `var r = require; r("./file")`	**yes**	no♦	no	no	no
Expressions in require (guided) `require("./templates/" + template)`	**yes (all files matching included)**	no♦	no	no	no
Expressions in require (free) `require(moduleName)`	with manual configuration	no♦	no	no	no
Requirable files	file system	**web**	file system	through plugins	file system or through plugins
Plugins	**yes**	yes	**yes**	yes	yes
Preprocessing	**loaders, transforms**	loaders	transforms	plugin translate	plugin transforms
Watch mode	yes	not required	yes	not needed in dev	no
Debugging support	**SourceUrl, SourceMaps**	not required	SourceMaps	**SourceUrl, SourceMaps**	**SourceUrl, SourceMaps**
Node.js built-in libs `require("path")`	**yes**	no	**yes**	**yes**	node-resolve-plugin
Other Node.js stuff	process, __dir/filename, global	-	process, __dir/filename, global	process, __dir/filename, global for cjs	global (commonjs-plugin)
Replacement for browser	`web_modules`, `.web.js`, package.json field, alias config option	alias option	package.json field, alias option	package.json, alias option	no
Minimizing	uglify	uglify, closure compiler	uglifyify	yes	uglify-plugin
Mangle path names	**yes**	no	partial	yes	not required (path names are not included in the bundle)
Runtime overhead	**243B + 20B per module + 4B per dependency**	14.7kB + 0B per module + (3B + X) per dependency	415B + 25B per module + (6B + 2X) per dependency	5.5kB for self-executing bundles, 38kB for full loader and polyfill, 0 plain modules, 293B CJS, 139B ES6 System.register before gzip	**none for ES2015 modules** (other formats may have)
Dependencies	19MB / 127 packages	11MB / 118 packages	**1.2MB / 1 package**	26MB / 131 packages	?MB / 3 packages

Webpack vs. the competition...

Just scratching the surface...

Dev Server

HMR (Hot Module Replacement)

Code Sharing &

Lazy Loaded Modules

Source Maps!!!!!

webpack 2

Native ES2015 Module Support
Tree Shaking
Faster Compilation
More Optimizations Built In
Better Loader Syntax
Configuration Validation

In beta until webpack.js.org milestone completion

A bunch more

Looking into the future...

Looking into the Future

- HTTP2: Webpack’s AgressiveSplittingPlugin (in latest!!)
- HTTP2: Dependency Tree driven Push Manifest
- Usability: Complete overhaul of the main interface
- Optimization: Module Inlining and Concatenation (Rollup)
- DevTools: Working with Multiple Browser Teams to Bring DevTools custom instrumentation and UI’s for webpack.
- (Crazy Ideas): Headless Chrome (timeline stats) + Machine Learning + Automatically Tweaked Configuration.
- (Crazy Idea): Bundler drive module spec?
- Accessiblity: How can we make testing easier w/ webpack

Wait, there’s more!

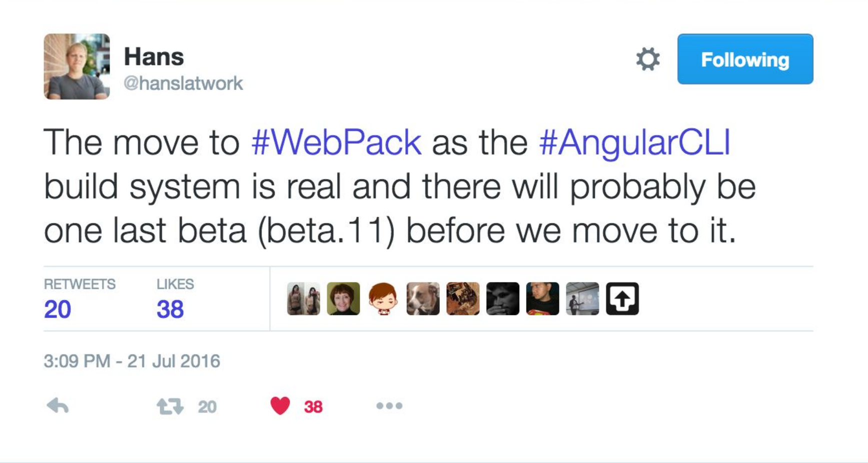
webpack

State of the Art

Who’s already using webpack?

State of the Art

angular/angular-cli



Hans
@hanslatwork

The move to #WebPack as the #AngularCLI build system is real and there will probably be one last beta (beta.11) before we move to it.

RETWEETS 20 LIKES 38

3:09 PM - 21 Jul 2016

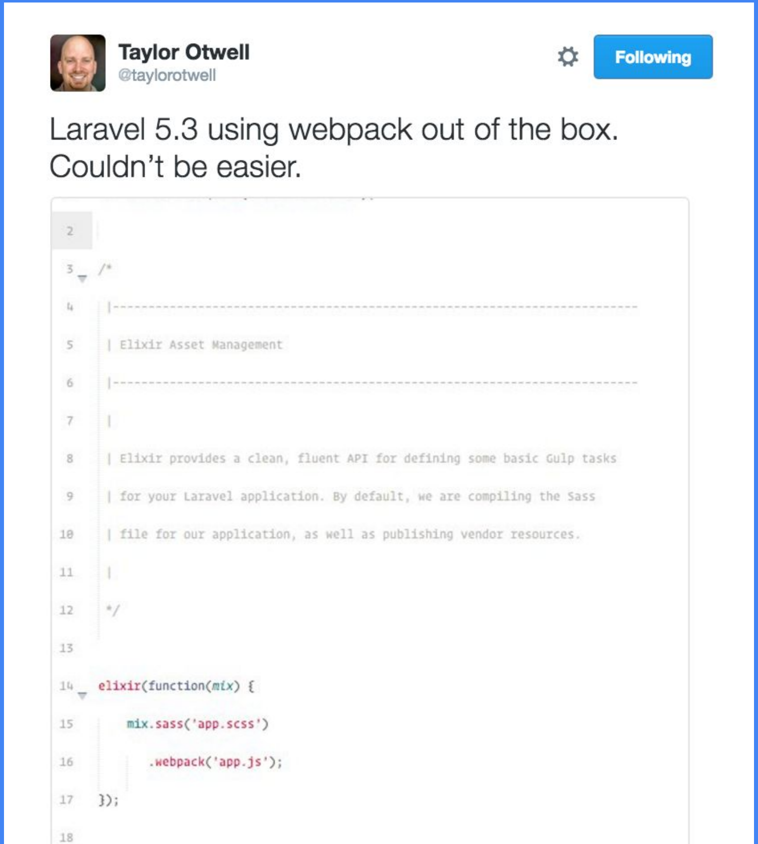
npm install -g angular-cli

State of the Art

facebookincubator/create-react-app

```
npm install -g create-react-app
```

State of the Art



Taylor Otwell
@taylorotwell

Laravel 5.3 using webpack out of the box. Couldn’t be easier.

```

/*
 |--------------------------------------------------------------------------
 | Elixir Asset Management
 |--------------------------------------------------------------------------
 |
 | Elixir provides a clean, fluent API for defining some basic Gulp tasks
 | for your Laravel application. By default, we are compiling the Sass
 | file for our application, as well as publishing vendor resources.
 |
 */

elixir(function(mix) {
    mix.sass('app.scss')
       .webpack('app.js');
});

```

State of the Art

Grails, Ruby on Rails, and more...
(drop in replacements)

State of the Art

JavaScript Services

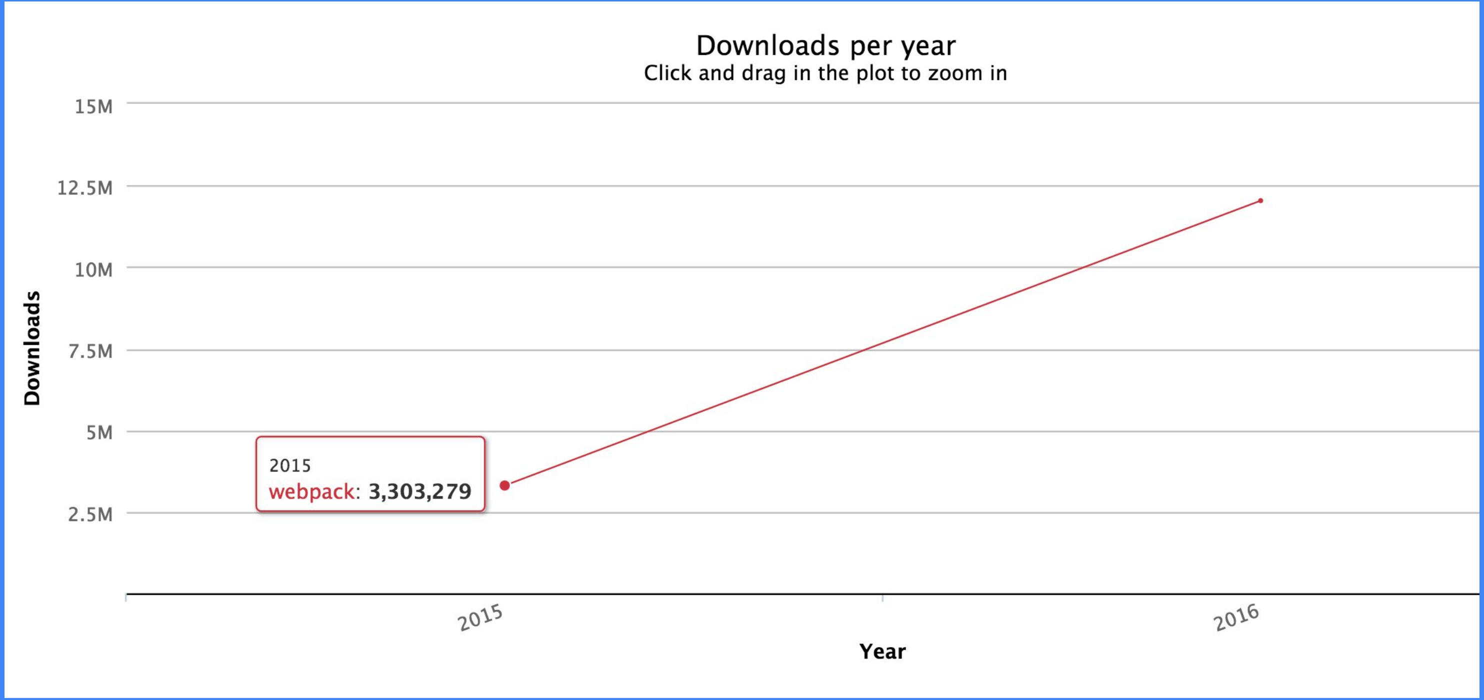
https://github.com/aspnet/JavaScriptServices

State of the Art

JHipster (Scaffolder for SpringBoot SpringMVC)
[WIP Angular2 + webpack setup]

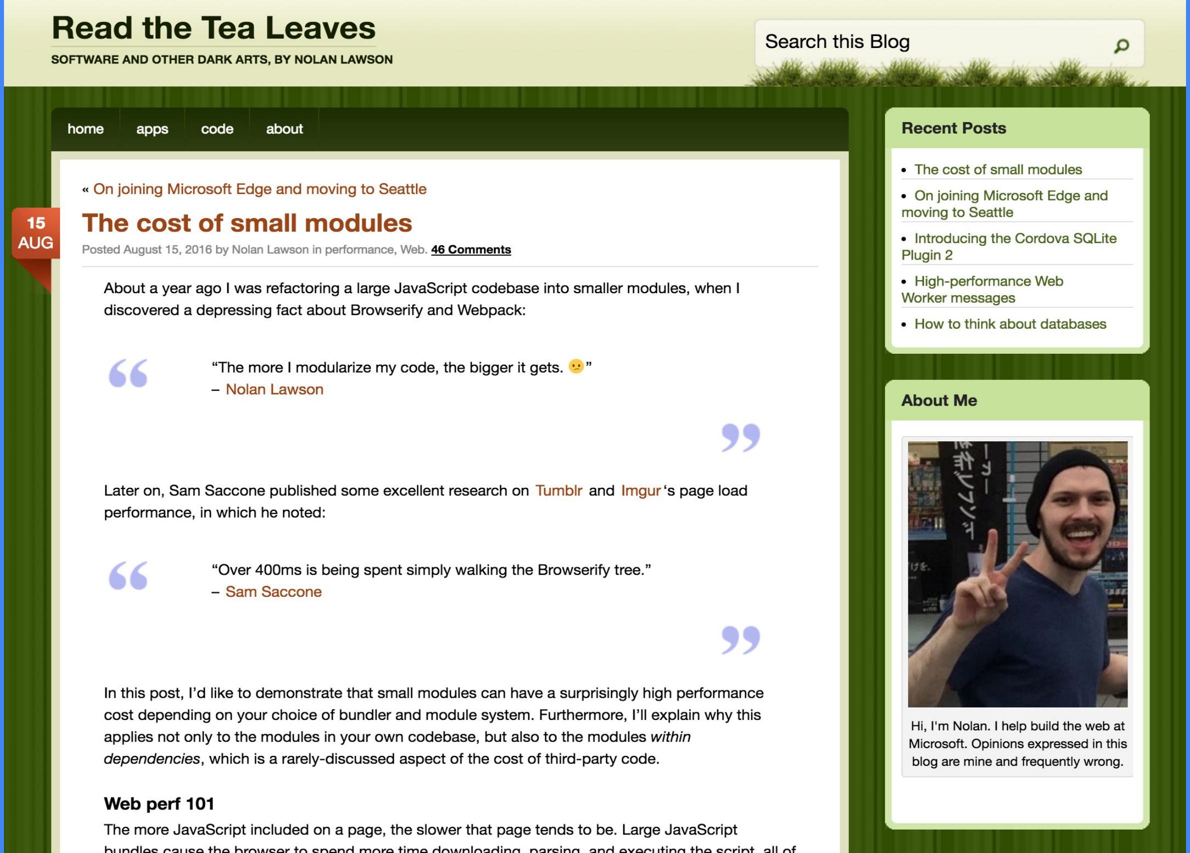
https://jhipster.github.io/

State of the Art



400% GROWTH IN 1 YEAR; 2M Monthly Downloads YTD

State of the Art



Read the Tea Leaves

SOFTWARE AND OTHER DARK ARTS, BY NOLAN LAWSON

« On joining Microsoft Edge and moving to Seattle

15 AUG

The cost of small modules

Posted August 15, 2016 by Nolan Lawson in performance, Web. 46 Comments

About a year ago I was refactoring a large JavaScript codebase into smaller modules, when I discovered a depressing fact about Browserify and Webpack:

> “The more I modularize my code, the bigger it gets. 😕”
> – Nolan Lawson

Later on, Sam Saccone published some excellent research on Tumblr and Imgur‘s page load performance, in which he noted:

> “Over 400ms is being spent simply walking the Browserify tree.”
> – Sam Saccone

In this post, I’d like to demonstrate that small modules can have a surprisingly high performance cost depending on your choice of bundler and module system. Furthermore, I’ll explain why this applies not only to the modules in your own codebase, but also to the modules *within dependencies*, which is a rarely-discussed aspect of the cost of third-party code.

Web perf 101

The more JavaScript included on a page, the slower that page tends to be. Large JavaScript bundles cause the browser to spend more time downloading, parsing, and executing the script, all of

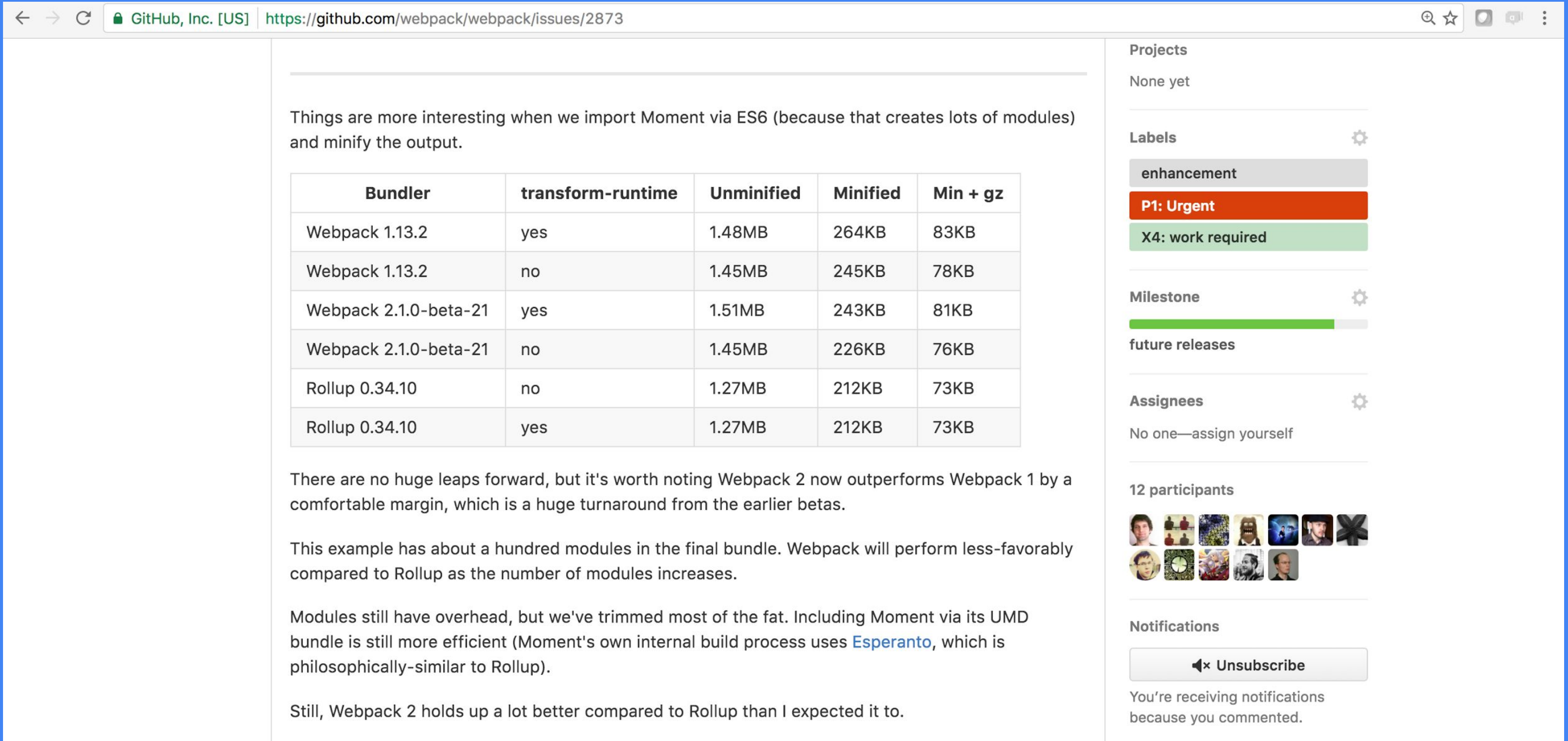
Recent Posts

- The cost of small modules
- On joining Microsoft Edge and moving to Seattle
- Introducing the Cordova SQLite Plugin 2
- High-performance Web Worker messages
- How to think about databases

About Me

Hi, I'm Nolan. I help build the web at Microsoft. Opinions expressed in this blog are mine and frequently wrong.

State of the Art



Things are more interesting when we import Moment via ES6 (because that creates lots of modules) and minify the output.

Bundler	transform-runtime	Unminified	Minified	Min + gz
Webpack 1.13.2	yes	1.48MB	264KB	83KB
Webpack 1.13.2	no	1.45MB	245KB	78KB
Webpack 2.1.0-beta-21	yes	1.51MB	243KB	81KB
Webpack 2.1.0-beta-21	no	1.45MB	226KB	76KB
Rollup 0.34.10	no	1.27MB	212KB	73KB
Rollup 0.34.10	yes	1.27MB	212KB	73KB

There are no huge leaps forward, but it's worth noting Webpack 2 now outperforms Webpack 1 by a comfortable margin, which is a huge turnaround from the earlier betas.

This example has about a hundred modules in the final bundle. Webpack will perform less-favorably compared to Rollup as the number of modules increases.

Modules still have overhead, but we've trimmed most of the fat. Including Moment via its UMD bundle is still more efficient (Moment's own internal build process uses Esperanto, which is philosophically-similar to Rollup).

Still, Webpack 2 holds up a lot better compared to Rollup than I expected it to.

Projects

None yet

Labels

enhancement

P1: Urgent

X4: work required

Milestone

future releases

Assignees

No one—assign yourself

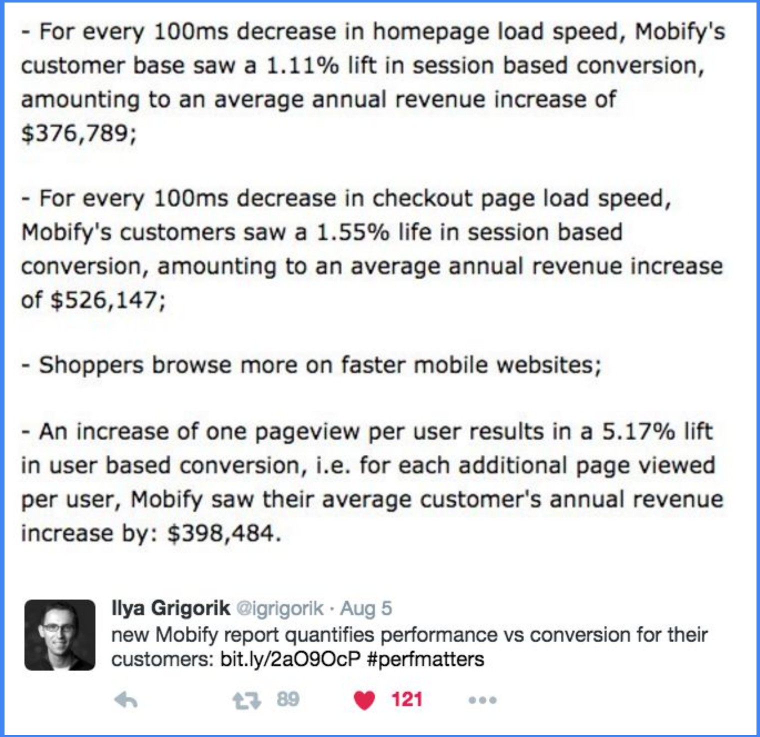
12 participants

Notifications

Unsubscribe

You're receiving notifications because you commented.

Why?



- For every 100ms decrease in homepage load speed, Mobify's customer base saw a 1.11% lift in session based conversion, amounting to an average annual revenue increase of $376,789;

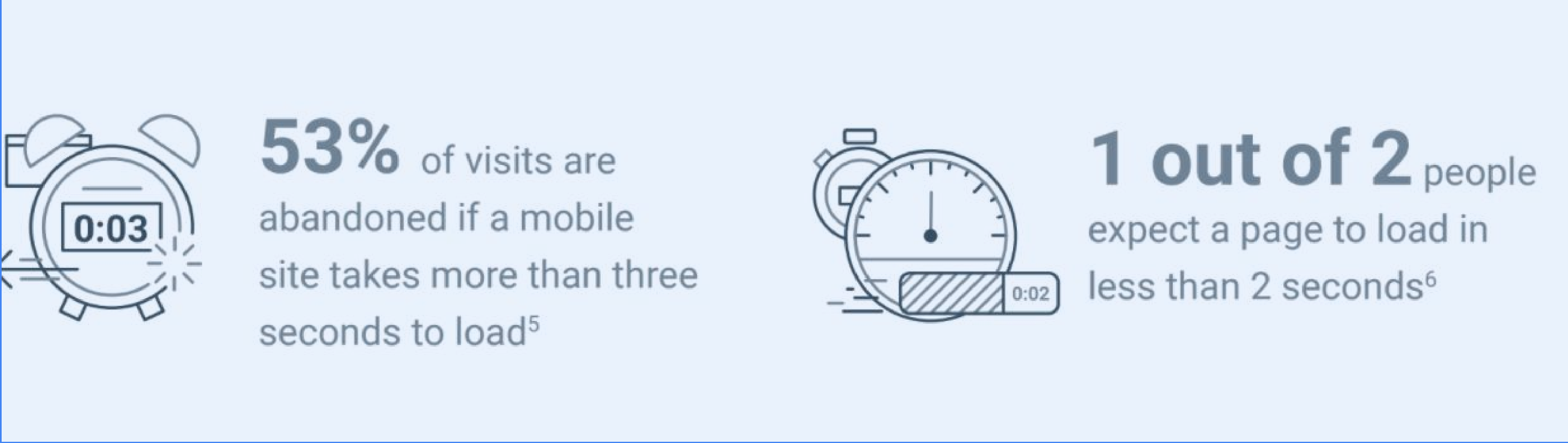
- For every 100ms decrease in checkout page load speed, Mobify's customers saw a 1.55% life in session based conversion, amounting to an average annual revenue increase of $526,147;

- Shoppers browse more on faster mobile websites;

- An increase of one pageview per user results in a 5.17% lift in user based conversion, i.e. for each additional page viewed per user, Mobify saw their average customer's annual revenue increase by: $398,484.

Ilya Grigorik @igrigorik · Aug 5

new Mobify report quantifies performance vs conversion for their customers: bit.ly/2aO9OcP #perfmatters

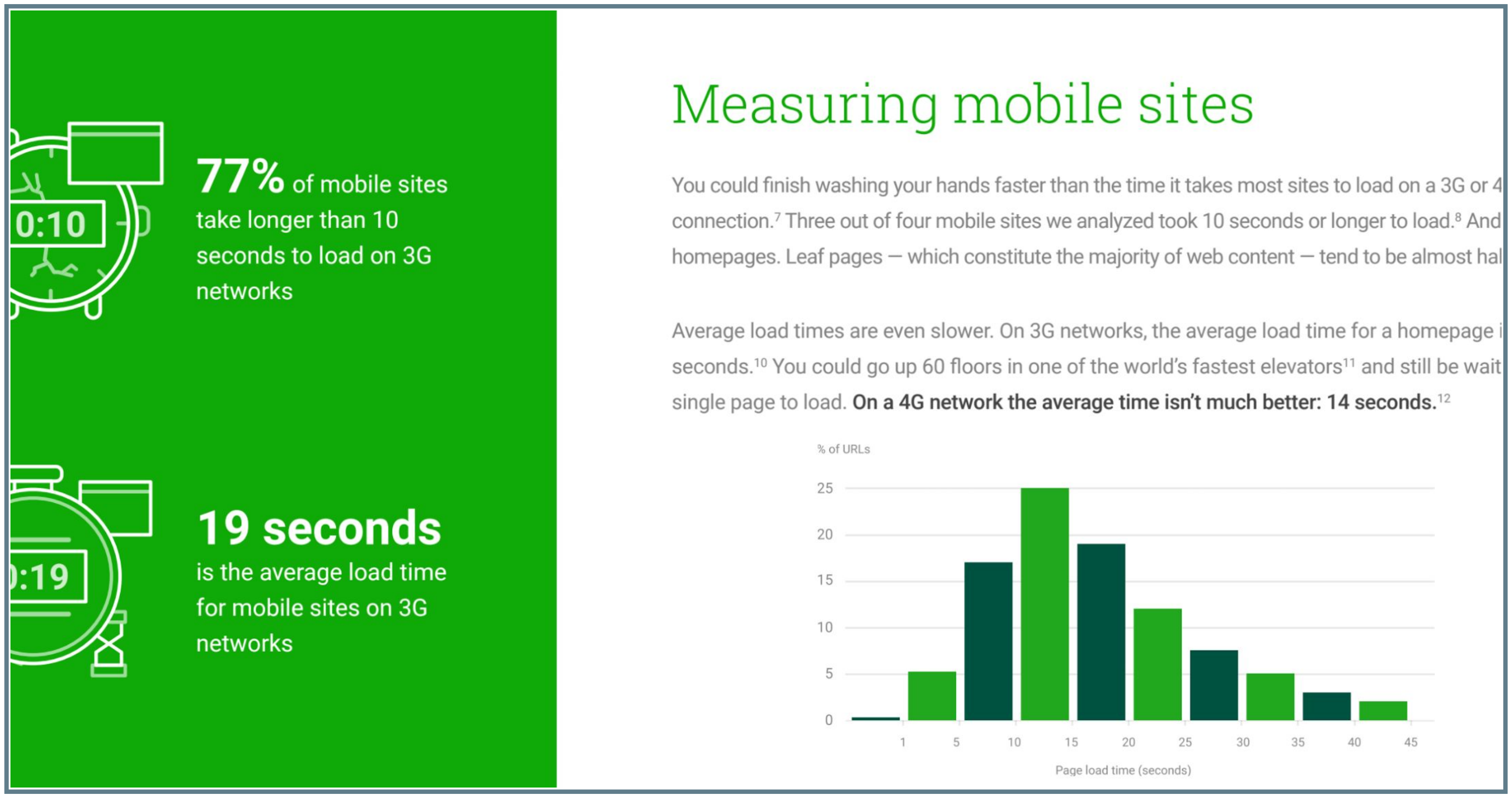


53% of visits are abandoned if a mobile site takes more than three seconds to load[5]

1 out of 2 people expect a page to load in less than 2 seconds[6]

https://docs.google.com/viewerng/viewer?url=https://storage.googleapis.com/doubleclick-prod/documents/The_Need_for_Mobile_Speed_-_FINAL.pdf

Why?



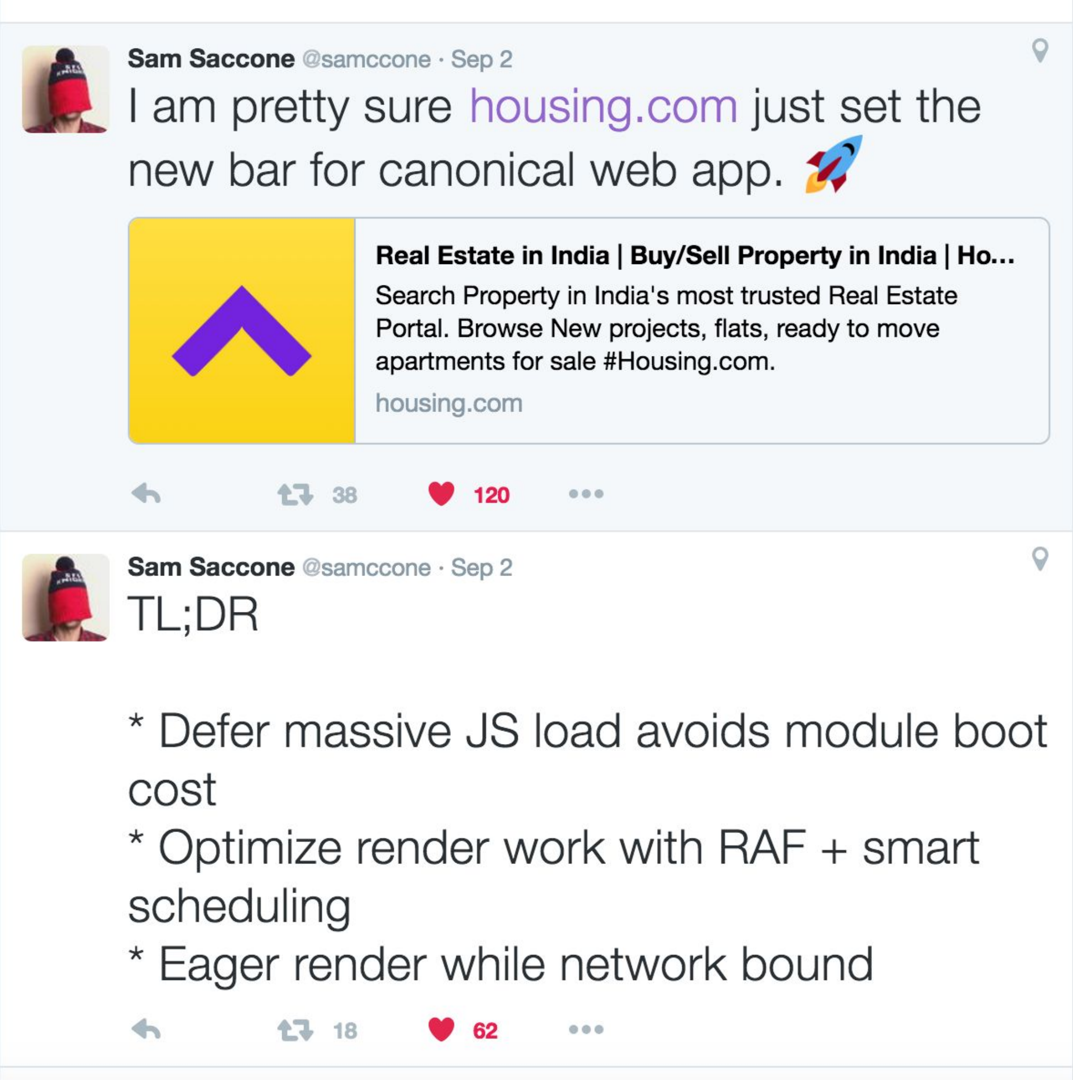
77% of mobile sites take longer than 10 seconds to load on 3G networks

19 seconds is the average load time for mobile sites on 3G networks

Measuring mobile sites

You could finish washing your hands faster than the time it takes most sites to load on a 3G or 4 connection.[7] Three out of four mobile sites we analyzed took 10 seconds or longer to load.[8] And homepages. Leaf pages — which constitute the majority of web content — tend to be almost hal

Average load times are even slower. On 3G networks, the average load time for a homepage i seconds.[10] You could go up 60 floors in one of the world's fastest elevators[11] and still be wait single page to load. **On a 4G network the average time isn't much better: 14 seconds.**[12]



Sam Saccone @samccone · Sep 2

I am pretty sure housing.com just set the new bar for canonical web app. 🚀

Real Estate in India | Buy/Sell Property in India | Ho...
Search Property in India's most trusted Real Estate Portal. Browse New projects, flats, ready to move apartments for sale #Housing.com.
housing.com

Sam Saccone @samccone · Sep 2

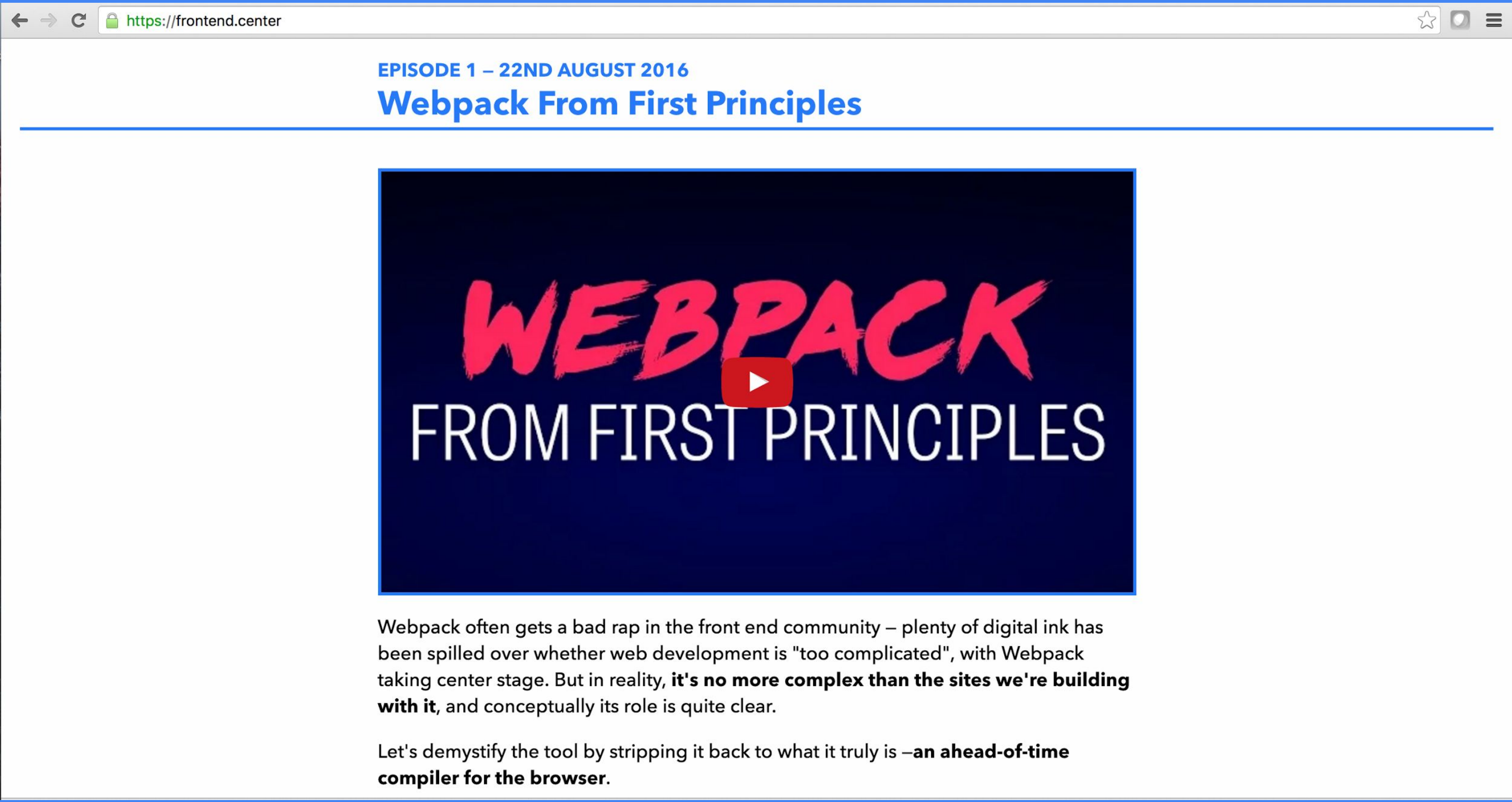
TL;DR

* Defer massive JS load avoids module boot cost
* Optimize render work with RAF + smart scheduling
* Eager render while network bound

Webpack is built by you...

and we give a shit...

Where can I start?



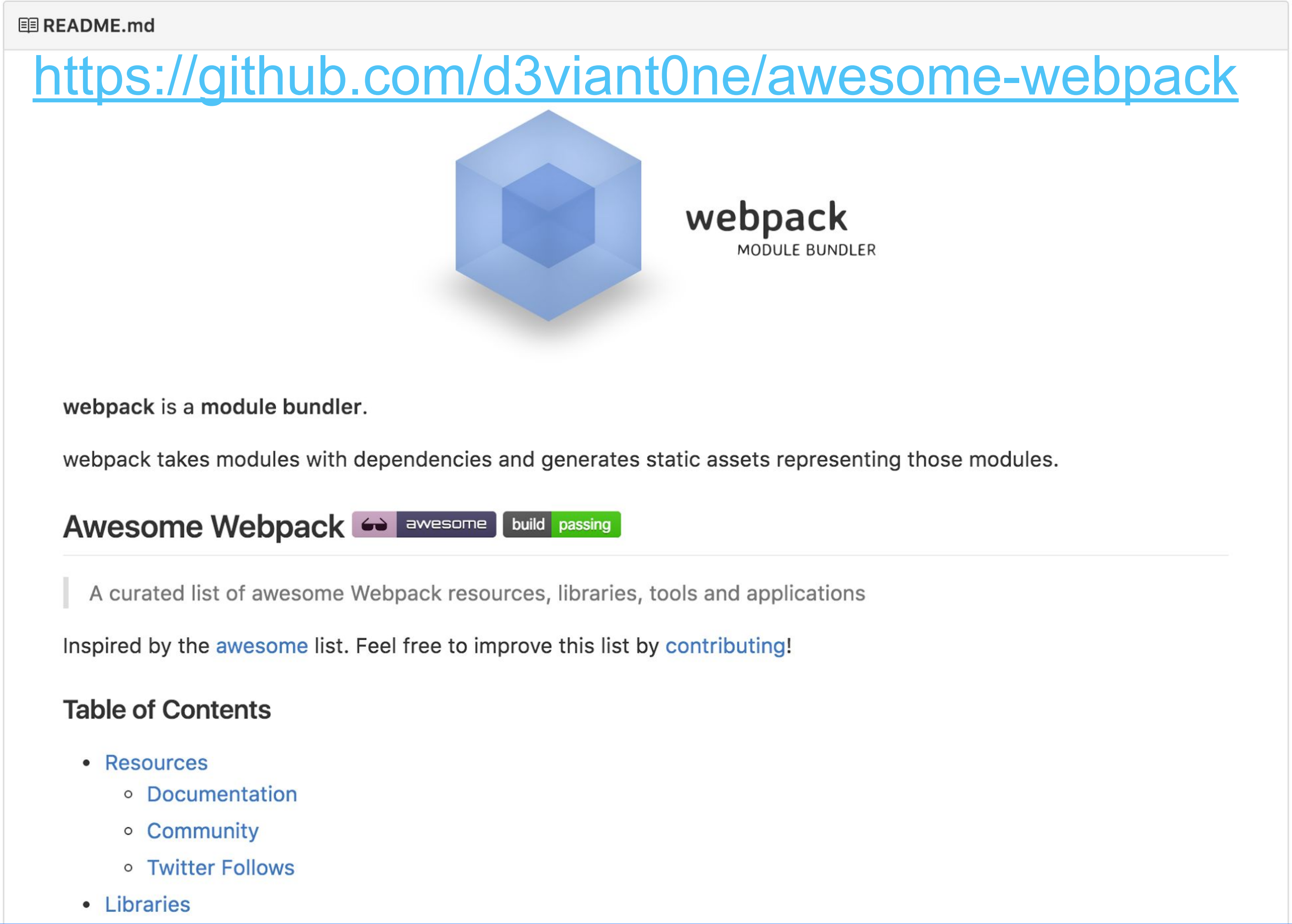
EPISODE 1 – 22ND AUGUST 2016

Webpack From First Principles

Webpack often gets a bad rap in the front end community – plenty of digital ink has been spilled over whether web development is "too complicated", with Webpack taking center stage. But in reality, **it's no more complex than the sites we're building with it**, and conceptually its role is quite clear.

Let's demystify the tool by stripping it back to what it truly is –**an ahead-of-time compiler for the browser**.

https://github.com/d3viant0ne/awesome-webpack



webpack is a **module bundler**.

webpack takes modules with dependencies and generates static assets representing those modules.

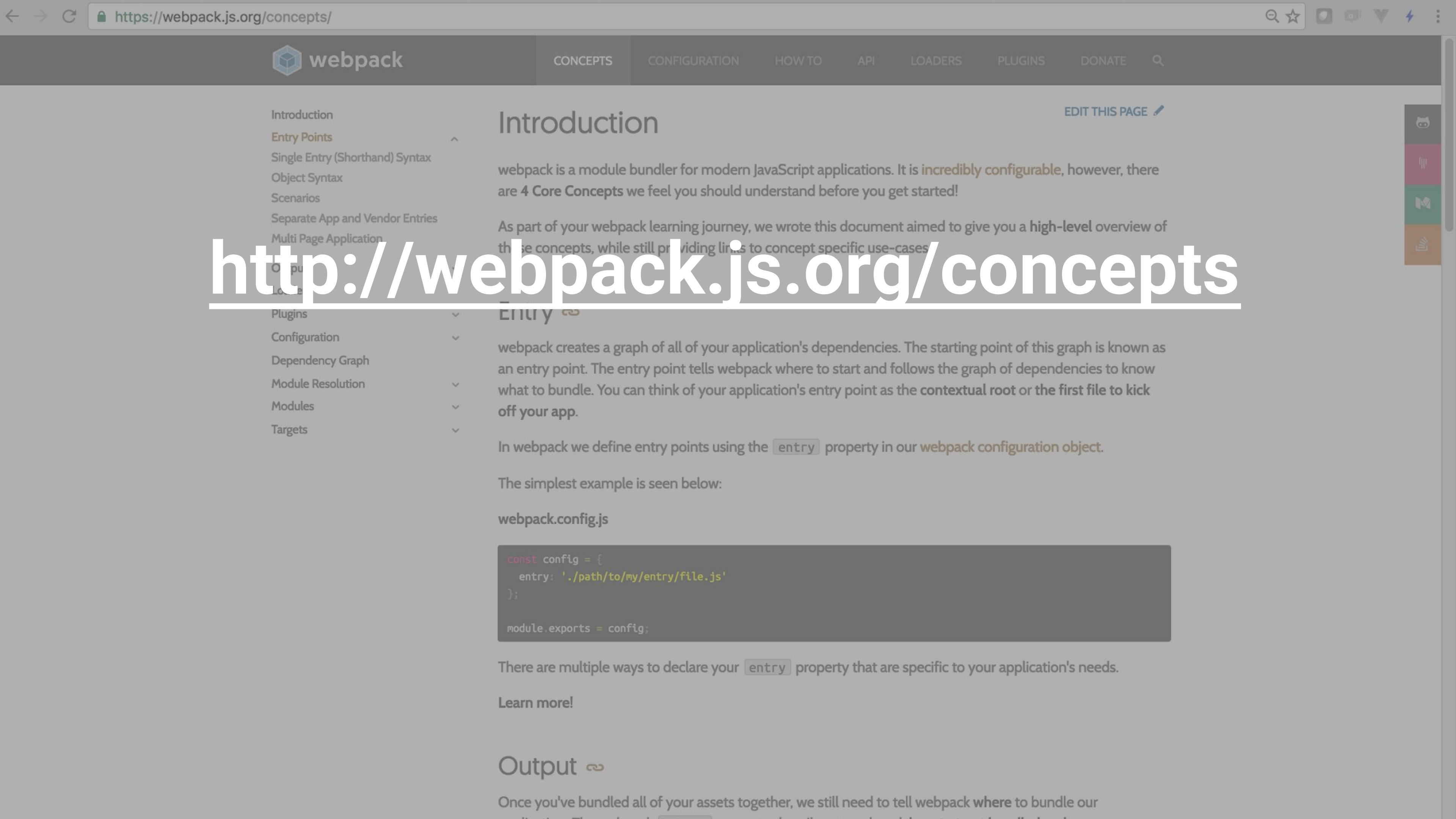
Awesome Webpack awesome build passing

> A curated list of awesome Webpack resources, libraries, tools and applications

Inspired by the awesome list. Feel free to improve this list by contributing!

Table of Contents

http://webpack.js.org/concepts



Introduction

webpack is a module bundler for modern JavaScript applications. It is incredibly configurable, however, there are **4 Core Concepts** we feel you should understand before you get started!

As part of your webpack learning journey, we wrote this document aimed to give you a **high-level** overview of these concepts, while still providing links to concept specific use-cases.

Entry

webpack creates a graph of all of your application's dependencies. The starting point of this graph is known as an entry point. The entry point tells webpack where to start and follows the graph of dependencies to know what to bundle. You can think of your application's entry point as the **contextual root** or **the first file to kick off your app**.

In webpack we define entry points using the `entry` property in our webpack configuration object.

The simplest example is seen below:

webpack.config.js

```
const config = {
  entry: './path/to/my/entry/file.js'
};

module.exports = config;
```

There are multiple ways to declare your `entry` property that are specific to your application's needs.

Learn more!

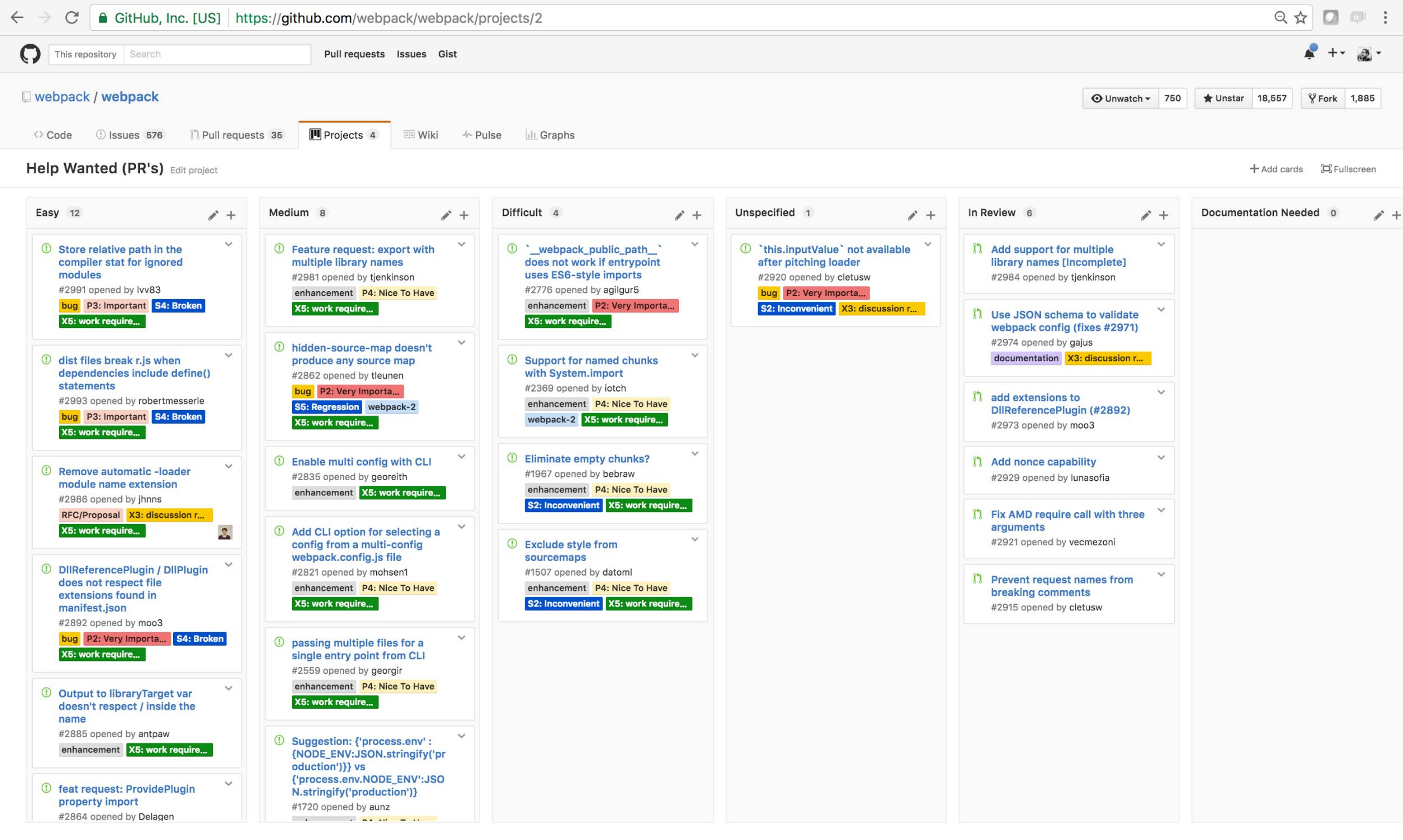
Output

Once you've bundled all of your assets together, we still need to tell webpack **where** to bundle our

How can I help?

triage

core loaders/plugins



Help Wanted (PR's)

Easy 12

- Store relative path in the compiler stat for ignored modules
  #2991 opened by lvv83
  bug · P3: Important · S4: Broken · X5: work require...
- dist files break r.js when dependencies include define() statements
  #2993 opened by robertmesserle
  bug · P3: Important · S4: Broken · X5: work require...
- Remove automatic -loader module name extension
  #2986 opened by jhnns
  RFC/Proposal · X3: discussion r... · X5: work require...
- DllReferencePlugin / DllPlugin does not respect file extensions found in manifest.json
  #2892 opened by moo3
  bug · P2: Very Importa... · S4: Broken · X5: work require...
- Output to libraryTarget var doesn't respect / inside the name
  #2885 opened by antpaw
  enhancement · X5: work require...
- feat request: ProvidePlugin property import
  #2864 opened by Delagen

Medium 8

- Feature request: export with multiple library names
  #2981 opened by tjenkinson
  enhancement · P4: Nice To Have · X5: work require...
- hidden-source-map doesn't produce any source map
  #2862 opened by tleunen
  bug · P2: Very Importa... · S5: Regression · webpack-2 · X5: work require...
- Enable multi config with CLI
  #2835 opened by georeith
  enhancement · X5: work require...
- Add CLI option for selecting a config from a multi-config webpack.config.js file
  #2821 opened by mohsen1
  enhancement · P4: Nice To Have · X5: work require...
- passing multiple files for a single entry point from CLI
  #2559 opened by georgir
  enhancement · P4: Nice To Have · X5: work require...
- Suggestion: {'process.env' : {NODE_ENV:JSON.stringify('production')}} vs {'process.env.NODE_ENV':JSON.stringify('production')}
  #1720 opened by aunz

Difficult 4

- `__webpack_public_path__` does not work if entrypoint uses ES6-style imports
  #2776 opened by agilgur5
  enhancement · P2: Very Importa... · X5: work require...
- Support for named chunks with System.import
  #2369 opened by iotch
  enhancement · P4: Nice To Have · webpack-2 · X5: work require...
- Eliminate empty chunks?
  #1967 opened by bebraw
  enhancement · P4: Nice To Have · S2: Inconvenient · X5: work require...
- Exclude style from sourcemaps
  #1507 opened by datoml
  enhancement · P4: Nice To Have · S2: Inconvenient · X5: work require...

Unspecified 1

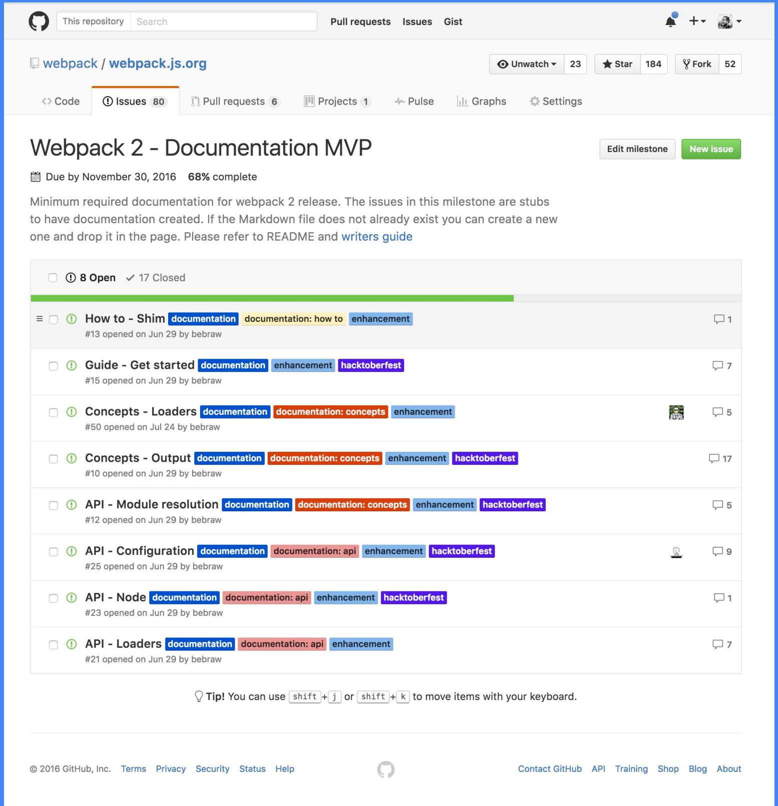
- `this.inputValue` not available after pitching loader
  #2920 opened by cletusw
  bug · P2: Very Importa... · S2: Inconvenient · X3: discussion r...

In Review 6

- Add support for multiple library names [Incomplete]
  #2984 opened by tjenkinson
- Use JSON schema to validate webpack config (fixes #2971)
  #2974 opened by gajus
  documentation · X3: discussion r...
- add extensions to DllReferencePlugin (#2892)
  #2973 opened by moo3
- Add nonce capability
  #2929 opened by lunasofia
- Fix AMD require call with three arguments
  #2921 opened by vecmezoni
- Prevent request names from breaking comments
  #2915 opened by cletusw

Documentation Needed 0

webpack/webpack.js.org



Webpack 2 - Documentation MVP

Due by November 30, 2016 **68%** complete

Minimum required documentation for webpack 2 release. The issues in this milestone are stubs to have documentation created. If the Markdown file does not already exist you can create a new one and drop it in the page. Please refer to README and writers guide

□ **8 Open** ✓ 17 Closed

- □ **How to - Shim** documentation documentation: how to enhancement — 1
#13 opened on Jun 29 by bebraw
- □ **Guide - Get started** documentation enhancement hacktoberfest — 7
#15 opened on Jun 29 by bebraw
- □ **Concepts - Loaders** documentation documentation: concepts enhancement — 5
#50 opened on Jul 24 by bebraw
- □ **Concepts - Output** documentation documentation: concepts enhancement hacktoberfest — 17
#10 opened on Jun 29 by bebraw
- □ **API - Module resolution** documentation documentation: concepts enhancement hacktoberfest — 5
#12 opened on Jun 29 by bebraw
- □ **API - Configuration** documentation documentation: api enhancement hacktoberfest — 9
#25 opened on Jun 29 by bebraw
- □ **API - Node** documentation documentation: api enhancement hacktoberfest — 1
#23 opened on Jun 29 by bebraw
- □ **API - Loaders** documentation documentation: api enhancement — 7
#21 opened on Jun 29 by bebraw

Tip! You can use `shift` + `j` or `shift` + `k` to move items with your keyboard.

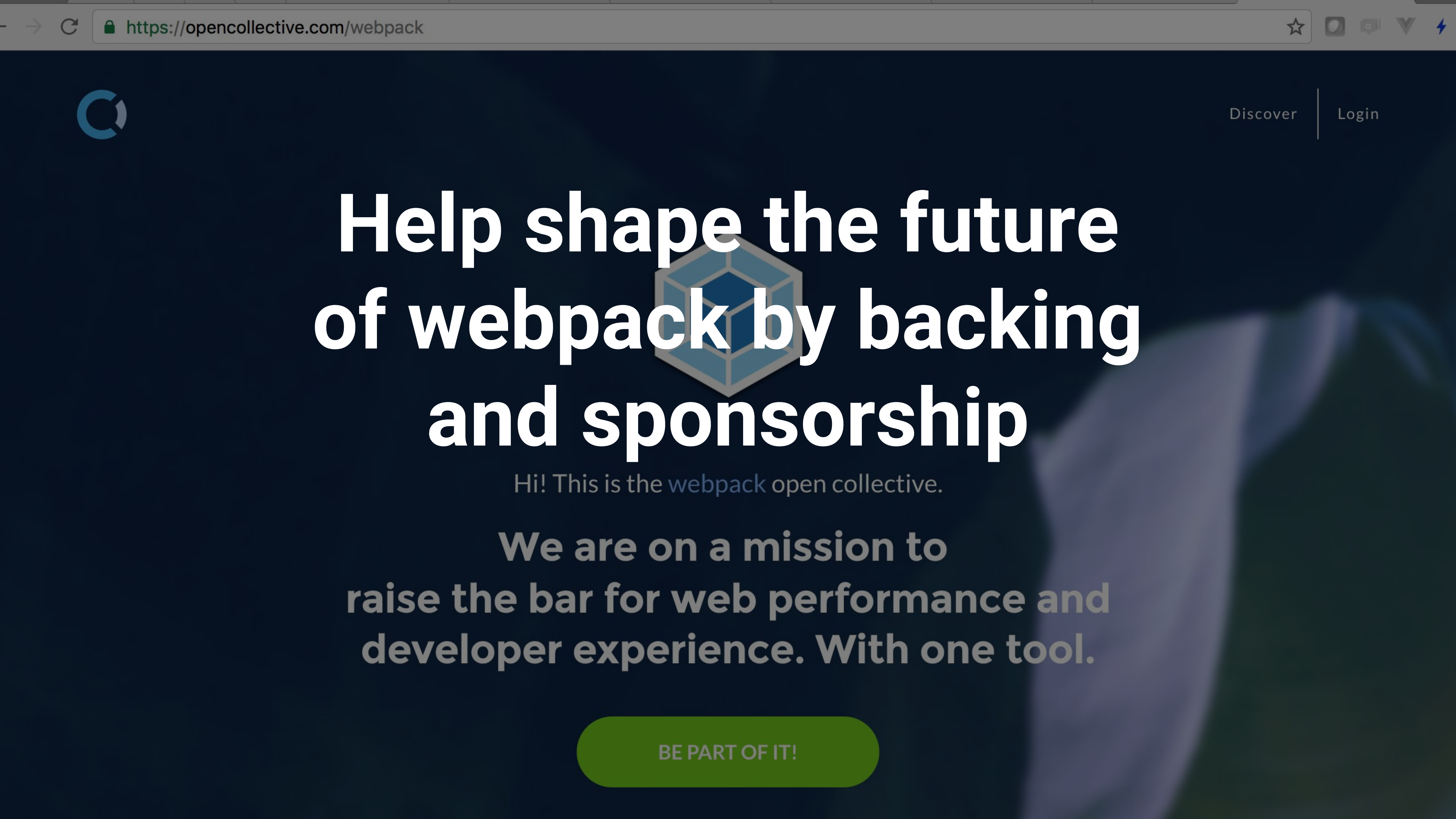
© 2016 GitHub, Inc. Terms Privacy Security Status Help

Contact GitHub API Training Shop Blog About

use webpack 2

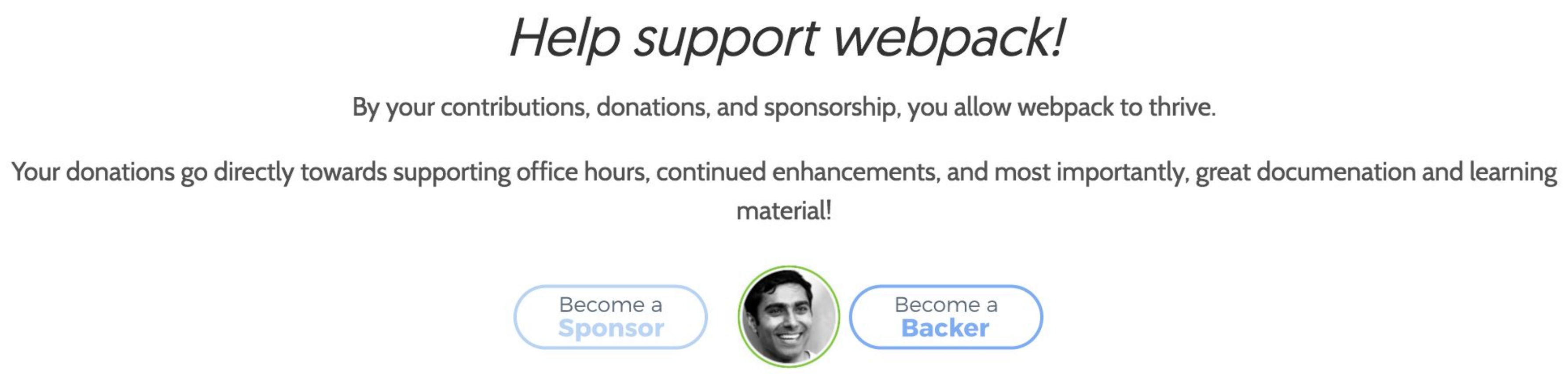
```
npm install webpack@2.1.0-beta.25
npm install webpack-dev-server@2.1.0-beta.9
```

Help shape the future of webpack by backing and sponsorship



Hi! This is the webpack open collective.

We are on a mission to raise the bar for web performance and developer experience. With one tool.



Help support webpack!

By your contributions, donations, and sponsorship, you allow webpack to thrive.

Your donations go directly towards supporting office hours, continued enhancements, and most importantly, great documenation and learning material!

opencollective.com/webpack

@TheLarkInn

Tweet me about this talk! I <3 Feedbac

#webpack

Tweet Questions, Gripes, Concerns, Frustrations