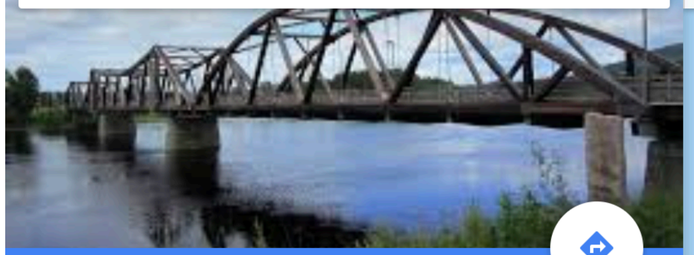
Full ~~Metal~~ Config Jacket

Or not, I’m not a cop.

Flisa



Flisa

2270

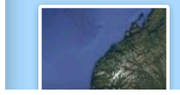
Cloudy · 6°C
11:36 AM

Directions

SAVE | NEARBY | SEND TO YOUR PHONE | SHARE

Quick facts

Flisa is a small town in south-eastern Norway, and the administrative centre of Åsnes municipality. Its population as of 1 January 2012 is 1,623. For some time the log driver statue was the town's only landmark. Wikipedia



UiT

THE ARCTIC
UNIVERSITY
OF NORWAY

https://academicpositions.no/uploads/2010/05/uit-logo.png

https://www.thousandwonders.net/Troms%C3%B8

webpack

«I’m not an expert.»

– Even Stensberg

- Creating the fundament for a Command Line Interface
- Abstractions & how to use them in your advantage
- Reducing entry requirements for new users

Creating a good baseline for a Command Line Interface

Our goal is to make libraries less hard to implement & use

A well established project increases chances of success with 0CJS

- Sets the baseline of your project
- Design Documents helps you to limit your project to its goal
- What is the first thing the user should see/do when they use my tool?
- Clarity in vision is less configuration - **If the developer doesn’t know how to set good defaults, how should the user know?**

https://github.com/webpack/webpack-cli/issues/101

Features definition v1

Sean Larkin edited this page on 5 Feb 2017 · 3 revisions

Overview

In this page, we draft out the direction in which we want to go in each of the features we will build for webpack-cli. This document will serve to guide us through the feature building.

Initial draft for tasks definition.

Compiler cli

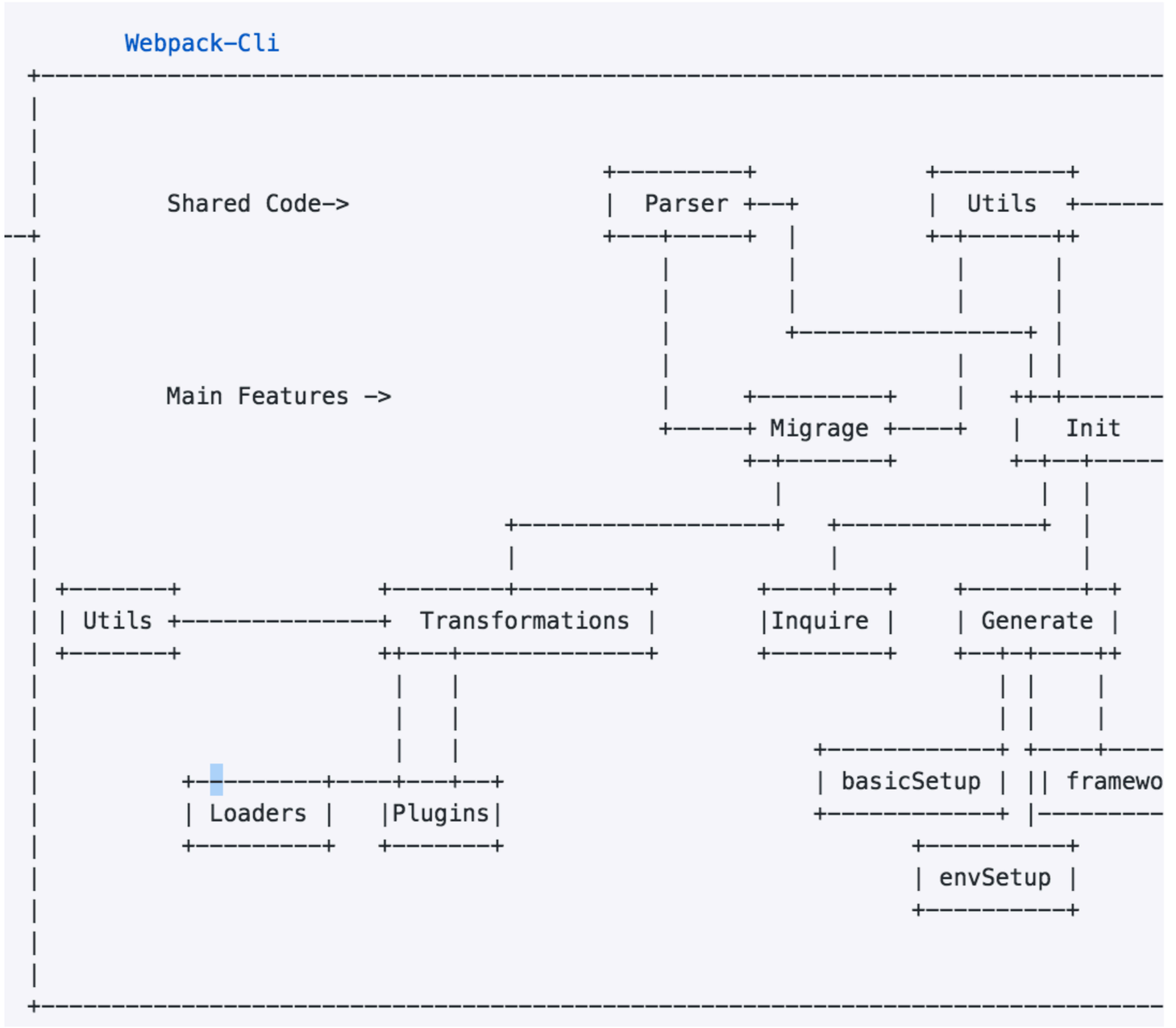
Goal: Have the current compiler cli options in the webpack-cli repo, living together with the new features.

Next steps

- Write tests.
- Add documentation in webpack.js.org

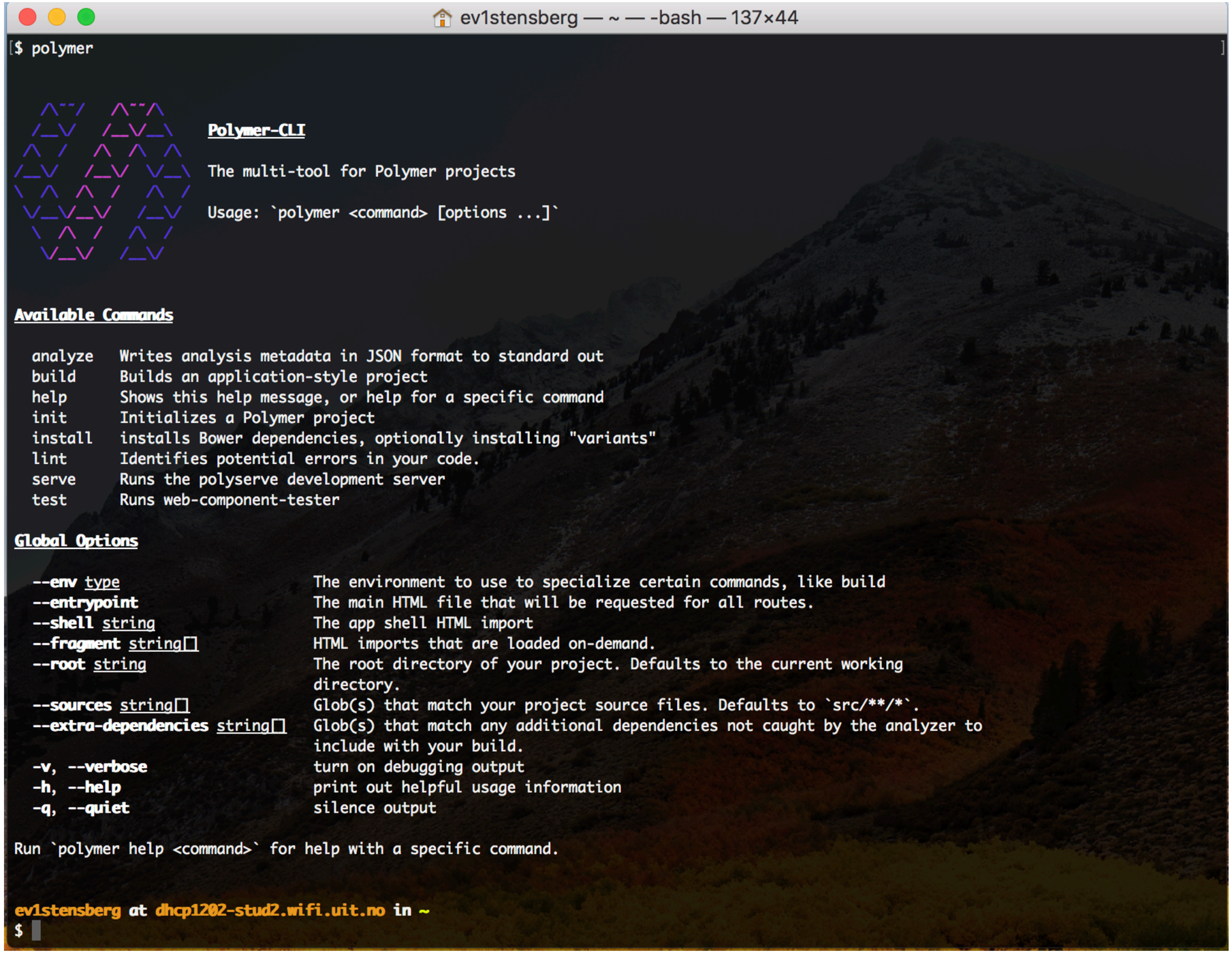
Transformations:

Goal: Read a given JS file, parse into AST, run transformations, write again into JS.


Webpack-Cli
Shared Code->
Parser
Utils
Main Features ->
Migrage
Init
Utils
Transformations
Inquire
Generate
Loaders
Plugins
basicSetup
framewo
envSetup

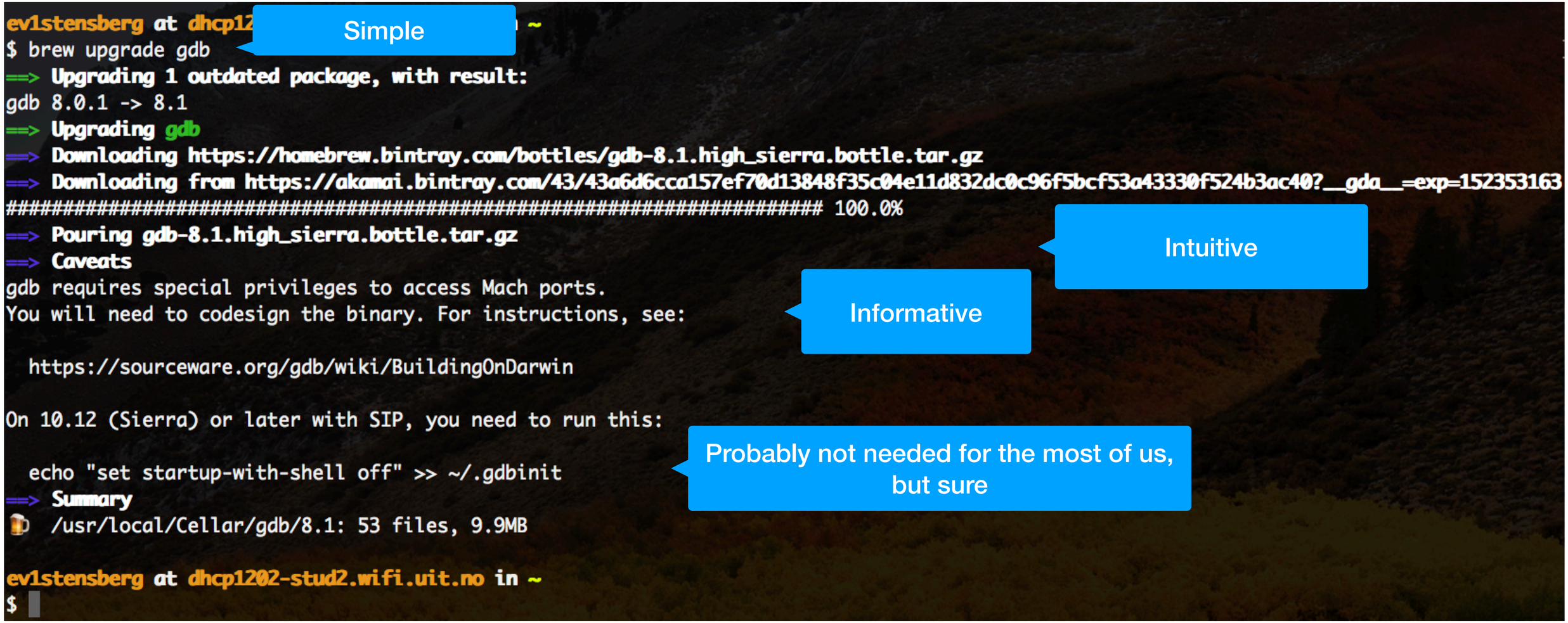
Project Structure

Polymer


ev1stensberg — ~ — -bash — 137×44
$ polymer
Polymer-CLI
The multi-tool for Polymer projects
Usage: `polymer <command> [options ...]`
Available Commands
analyze Writes analysis metadata in JSON format to standard out
build Builds an application-style project
help Shows this help message, or help for a specific command
init Initializes a Polymer project
install installs Bower dependencies, optionally installing "variants"
lint Identifies potential errors in your code.
serve Runs the polyserve development server
test Runs web-component-tester
Global Options
--env type The environment to use to specialize certain commands, like build
--entrypoint The main HTML file that will be requested for all routes.
--shell string The app shell HTML import
--fragment string[] HTML imports that are loaded on-demand.
--root string The root directory of your project. Defaults to the current working directory.
--sources string[] Glob(s) that match your project source files. Defaults to `src/**/*`.
--extra-dependencies string[] Glob(s) that match any additional dependencies not caught by the analyzer to include with your build.
-v, --verbose turn on debugging output
-h, --help print out helpful usage information
-q, --quiet silence output
Run `polymer help <command>` for help with a specific command.
ev1stensberg at dhcp1202-stud2.wifi.uit.no in ~
$

- Polymer parses its commands to a handler
- Handler runs a generator relative to its command
- «Almost» the entire source code is class based
- Intuitive Interface
- Simple & Specific commands
- ^ Easy for users to understand

Abstractions


ev1stensberg at dhcp12 ~
$ brew upgrade gdb
==> Upgrading 1 outdated package, with result:
gdb 8.0.1 -> 8.1
==> Upgrading gdb
==> Downloading https://homebrew.bintray.com/bottles/gdb-8.1.high_sierra.bottle.tar.gz
==> Downloading from https://akamai.bintray.com/43/43a6d6cca157ef70d13848f35c04e11d832dc0c96f5bcf53a43330f524b3ac40?__gda__=exp=152353163
100.0%
==> Pouring gdb-8.1.high_sierra.bottle.tar.gz
==> Caveats
gdb requires special privileges to access Mach ports.
You will need to codesign the binary. For instructions, see:
https://sourceware.org/gdb/wiki/BuildingOnDarwin
On 10.12 (Sierra) or later with SIP, you need to run this:
echo "set startup-with-shell off" >> ~/.gdbinit
==> Summary
/usr/local/Cellar/gdb/8.1: 53 files, 9.9MB
ev1stensberg at dhcp1202-stud2.wifi.uit.no in ~
$
Simple
Intuitive
Informative
Probably not needed for the most of us, but sure

```
1 warning generated.
ld: can't write output file: indexer for architecture x86_64
clang: error: linker command failed with exit code 1 (use -v to see invocation)
make: *** [indexer] Error 1
```

Okay?

What's «Error 1» ?

Why don't do this by default?

- Too much abstraction is overkill
- Too little makes source code messy
- Balance abstractions, try to generalize abstractions so that you can reuse them later.
- 📍`Tip`: **Balance the weight between imperative and functional programming**

https://medium.com/@isaace71295/bad-abstraction-is-the-problem-not-too-much-abstraction-5e5199603c56

Good

```
1  /**
2   * Copyright (c) 2013-present, Facebook, Inc.
3   *
4   * This source code is licensed under the MIT license found in the
5   * LICENSE file in the root directory of this source tree.
6   * @flow
7   */
8
9  import type {RefObject} from 'shared/ReactTypes';
10
11 // an immutable object with a single mutable value
12 export function createRef(): RefObject {
13   const refObject = {
14     current: null,
15   };
16   if (__DEV__) {
17     Object.seal(refObject);
18   }
19   return refObject;
20 }
```

You should add documentation here

Imports aren't over bloated

Named function

Clear code, easy to know what's happening

Bad *(Nested functions, 200LOC++)*

Named function, good.

```
37  export default function YamcsPlugin(options) {
38
39    const host = options.host || 'localhost';
40    const port = options.port || '8090';
41    const instance = options.instance || 'simulator';
42
43    const TELEMETRY = getDictionary().then(function(dictionary) {
44      return dictionary.map(function(param) {
45        return param;
46      });
47    });
48    function getDictionary() {
49      return axios
50        .get(`http://${host}:${port}/api/mdb/${instance}/parameters`)
51        .then(response => response.data.parameter);
52    }
```

Intuitive, but what happens when we want to fetch telemetry from another plugin?

Fallback values, good.

Could be abstracted.

Could be abstracted.

This file is over 200LOC

- Reusability is key
- 3rd party dependencies can reuse your code
- Abstractions reduce entry requirements for maintainers
- **more time to fix bugs & to tweet about it**

Focus on your library, not the tools used to get there

Create React App

- Sets Good Defaults
- Promotes Good Practice
- Wrapper Suite
- ***Feedback forms longer than your tax report*** (which is good)

The `sw-precache-webpack-plugin` is integrated into production configuration, and it will take care of generating a service worker file that will automatically precache all of your local assets and keep them up to date as you deploy updates. The service worker will use a cache-first strategy for handling all requests for local assets, including the initial HTML, ensuring that your web app is reliably fast, even on a slow or unreliable network.

```
Success! Created my-app at /Users/ev1stensberg/Documents/my-app
Inside that directory, you can run several commands:

  yarn start
    Starts the development server.

  yarn build
    Bundles the app into static files for production.

  yarn test
    Starts the test runner.

  yarn eject
    Removes this tool and copies build dependencies, configuration files
    and scripts into the app directory. If you do this, you can’t go back!

We suggest that you begin by typing:

  cd my-app
  yarn start

Happy hacking!
```

Performance

Progressive Web App

Accessibility

Best Practices

SEO

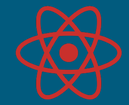
Ask for feedback, often.

- What did we do bad ❓ **What** ❗
- How can we make this better ❓ **How** ❗
- Did you struggle with our interface ❓ **Why** ❗
- Why did you find the tool hard to use ❓ **How come** ❗
- ⚠️ **User stories** - put yourself in the user's perspective and deduct

https://www.zazzle.co.uk/i_got_your_back_funny_stick_figures_postcard-239087971434308602

Platform Agnostic Tools

Ecosystems that enforce abstractions could hide complexity

- Yeoman 
- Babel 
- NPM 
- OpenMCT 
- *literally every React component there is* 

No Need to Reinvent The Wheel

- Commander - **maintainers** 
- Inquirer - **developers** ❓
- V8-Compile-Cache - **performance** 
- **Or just browse Sindre Sørhus's libraries at GitHub** 

Even Stensberg
@ev1stensberg

Adding v8-compile-cache for webpack v.4.0.0-alpha-1 on "minor source code":

webpack add (definePlugin):

no v8: 11492.589ms

v8: 6979.034ms

webpack-lighthouse-plugin demo (production mode, no plugins):

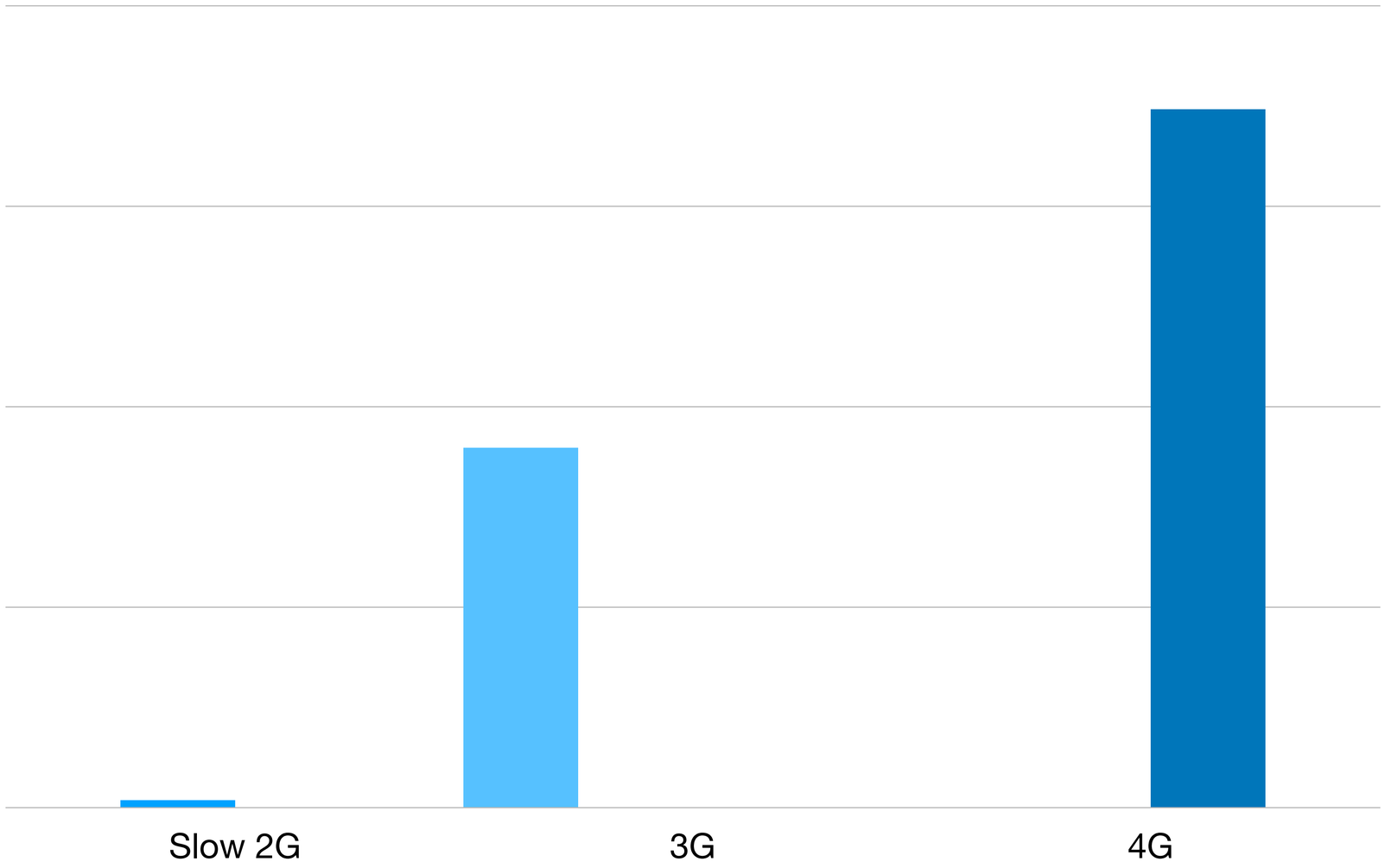
no v8: 1416.967ms

v8: 1385.388ms

7:27 AM - 21 Dec 2017

Don’t overuse dependencies

- What am I trying to accomplish?
- What dependencies do I really need?
- Could I create the function myself?

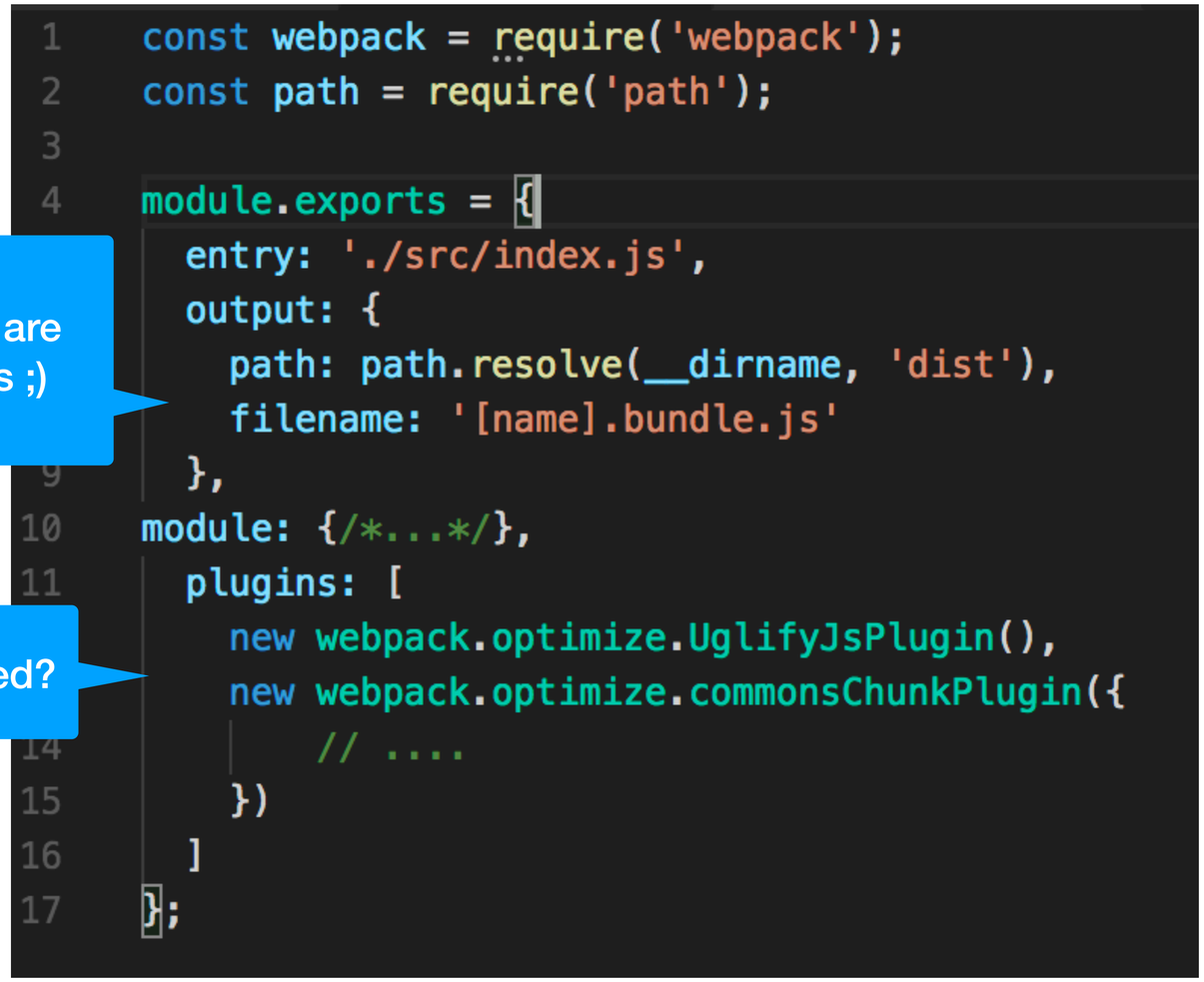


https://bigquery.cloud.google.com/results/my-chrome-ux-report-201008:US.bquijob_334858e3_162beb9a2cb

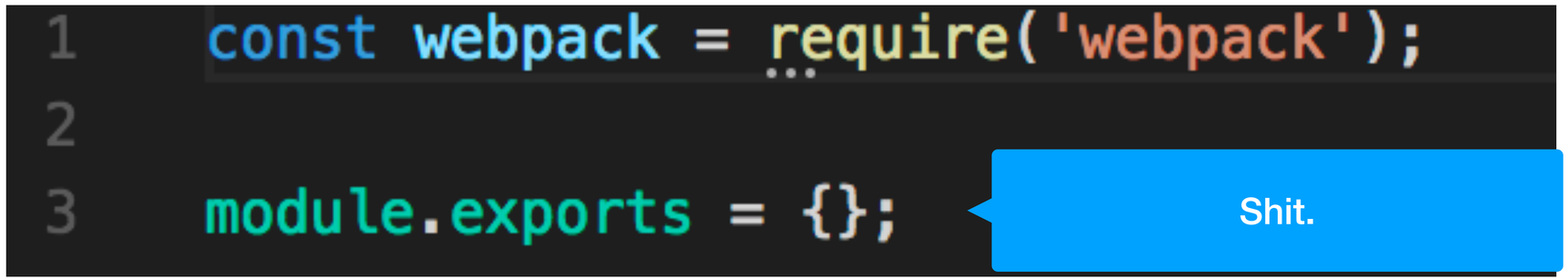
Reducing Entry Requirements

- Good defaults 👍🏽
- Minimalistic starting point 🔥
- Simple & intuitive interfaces ✅
- Scale complexity (*if ever needed*) over time 📈

Before


Entry & Output are always index.js ;)
Optimizations could be defaulted?
const webpack = require('webpack');
const path = require('path');
module.exports = {
entry: './src/index.js',
output: {
path: path.resolve(__dirname, 'dist'),
filename: '[name].bundle.js'
},
module: {/*...*/},
plugins: [
new webpack.optimize.UglifyJsPlugin(),
new webpack.optimize.commonsChunkPlugin({
// ....
})
]
};

After



Few tips on setting defaults

- Follow conventions
- Follow and set best practice, example: ***Google Developers***
- Remove unneeded arguments
- If you can code it without input from the user, strip it

We’ve been here
before, haven’t we?

0CJS is good for starting a project

But…

You lose visibility

Users might end up with a Black Box


0CJS
Full Config
Sweet Spot

0CJS isn’t only about no configuration

It’s about educating the developer

Don’t get on the hype train before knowing where to go

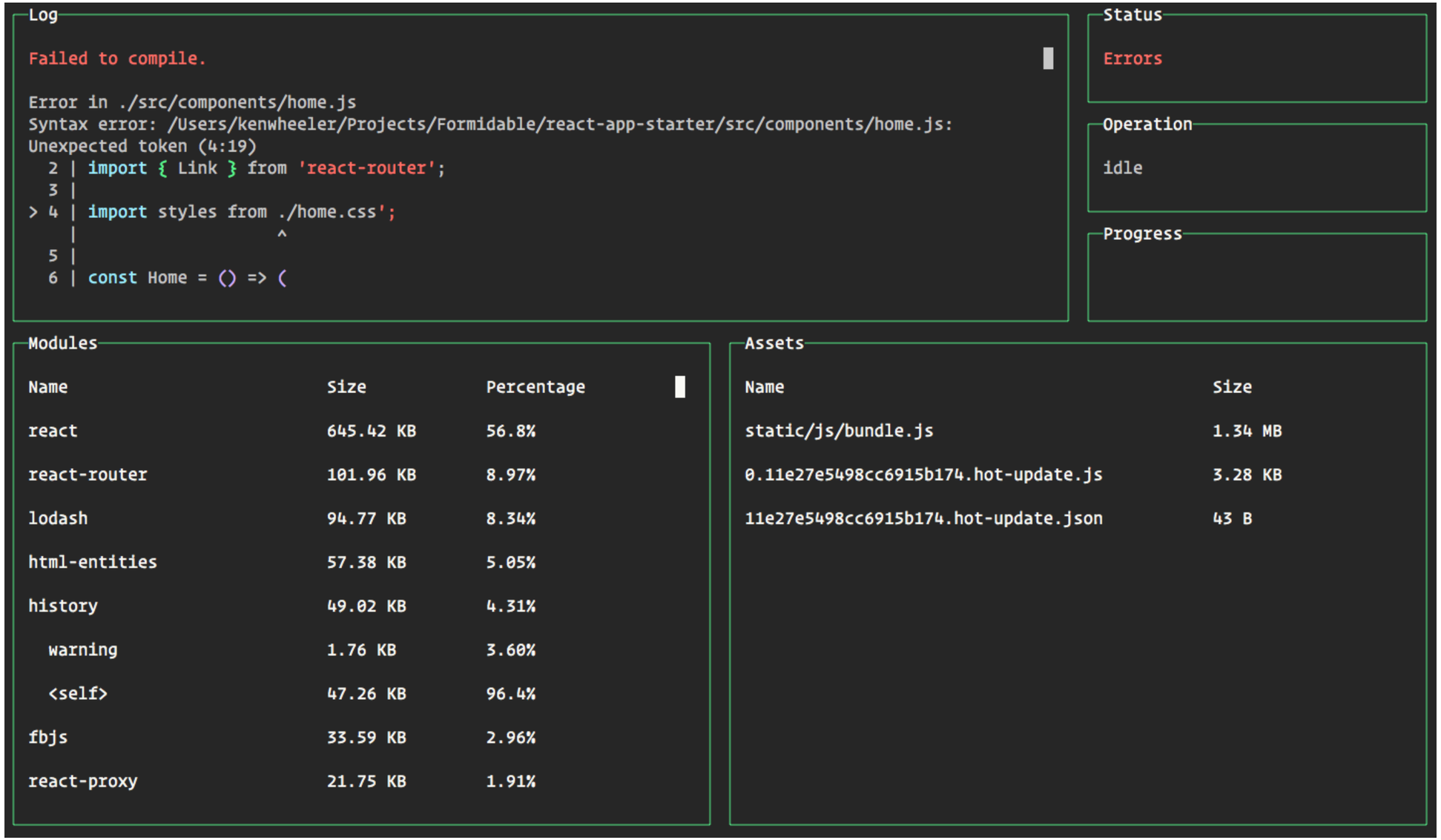
«We believe that ~~transparency~~ configurations are needed to create ~~trust~~ visibility, and it’s also needed to ~~create a dialogue~~ educate developers.»

– Julie Sweet

If 0CJS don’t help?

Use GUI’s / Visual representations

Don’t be stubborn, developers can save time at GUI’s, example -> GH desktop etc


Log
Failed to compile.
Error in ./src/components/home.js
Syntax error: /Users/kenwheeler/Projects/Formidable/react-app-starter/src/components/home.js:
Unexpected token (4:19)
2 | import { Link } from 'react-router';
3 |
> 4 | import styles from ./home.css';
| ^
5 |
6 | const Home = () => (
Status
Errors
Operation
idle
Progress
Modules
Name Size Percentage
react 645.42 KB 56.8%
react-router 101.96 KB 8.97%
lodash 94.77 KB 8.34%
html-entities 57.38 KB 5.05%
history 49.02 KB 4.31%
warning 1.76 KB 3.60%
<self> 47.26 KB 96.4%
fbjs 33.59 KB 2.96%
react-proxy 21.75 KB 1.91%
Assets
Name Size
static/js/bundle.js 1.34 MB
0.11e27e5498cc6915b174.hot-update.js 3.28 KB
11e27e5498cc6915b174.hot-update.json 43 B

Summary

- Think about the project - **Don’t over-engineer it**
- 0CJS is important - **But so is visibility**
- Reuse modules - **People use them**
- Use/Create ecosystems when you can - **Abstractions hide complexity**
- All this = **Low tech debt + Educating developers**