Bootstrapping a React project with ES6

@gabipurcaru / gabi.purcaru.com

Stage 0 Karma Babel Isparta WebPack Dev Server TypeScript Istanbul PhantomJS NPM Scripts NPM Bluebird PostCSS Bower React JSX Babel-Loader Hot Module Reloading ES2015 JSCS ES7+ Traceur ES6 Flow AMD JSLint ESLint Chai WebPack Gulp Mocha Grunt

Stage 0 Karma **Babel** Isparta **WebPack Dev Server** TypeScript Istanbul PhantomJS **NPM Scripts NPM** Bluebird PostCSS Bower **React JSX Babel-Loader** Hot Module Reloading Browserify ES2015 JSCS ES7+ **ES6** Flow AMD JSLint ESLint Chai **WebPack** Gulp Mocha Grunt

Yeoman / starter kits

Pro

★ Usor de folosit
★ Suport din partea comunitatii

Contra

★ In general destul de complexe
★ Inflexibile
★ Tot trebuie sa pricepi boilerplate-ul la un moment dat

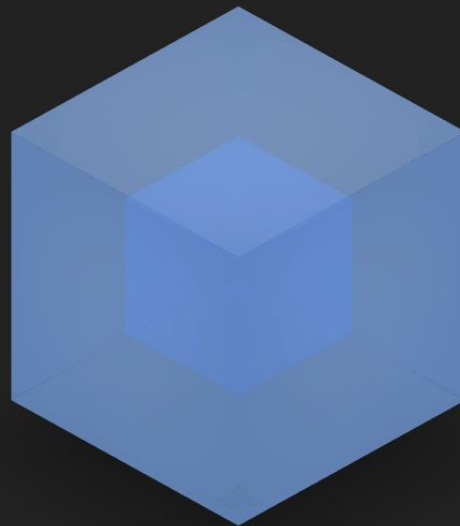# WebPack

★ Module bundler
  ○ JS
    ■ ES5/6/7/etc.
    ■ Flow
    ■ CommonJS / AMD
  ○ CSS, LESS, SASS, etc.
  ○ Imagini si alte asset-uri
★ Foarte modular
  ○ multe pluginuri

Babel

★ JavaScript compiler

  ○ Compileaza un dialect de JS (ES5/6/7+, Flow, JSX, etc.) in alt dialect de JS
  ○ Foarte modular (plugins + presets)
  ○ Babel-loader pentru WebPack
  ○ Polyfills

NPM

★ Nu doar pentru Node.js!
★ O alternativa foarte buna la Bower
★ NPM Scripts
★ Normal dependencies vs. development
   dependencies
★ Package.json

ES6

- Arrow functions
- Import / export syntax
- Array / object destructuring
- Classes
- Template literals
- etc.

ES6 - arrow functions

```
function(a) { return a + 2 }           a => a + 2

function(a, b) { return a * b }        (a, b) => a * b

(function(a) { return a + this.b       a => a + this.b
}).bind(this)

function(a) { return a[1] - a[0] }     ([a, b]) => b - a

function(o) { var c = o.a + o.b;       ({ a, b }) => { const c = a + b;
return c }                             return c }
```

ES6 - import / export

```
import React from 'react'          var React = require('react');

import { Card } from 'material-ui'  var Card =
                                     require('material-ui').Card

export default function f() { //    module.exports = function f() { }
... }

export const pi = 22 / 7;          module.exports.pi = 22 / 7
```

ES6 - classes

```
class A extends B {

    constructor() {

        super() // must be the first statement

    }

    static f() { // ...  }

}
```

ES6 - template literals

```
"GBP" + format(amount)          `GBP${format(amount)}`

"\\n"                           String.raw`\n`
```

Live Demo

Q&A