React, Webpack, Flux

Isomorphism, Server Rendering, ES6, VDOM, Immutability...

Buzzwords... Buzzwords everywhere.

chriskjaer

Fullstack Udvikler @ Boligbesked

React?

"a javascript library for building user interfaces"

React Concepts

- Virtual DOM
- One-way databinding
- JSX

JSX

“Templates separate technologies, not concerns.”

Example

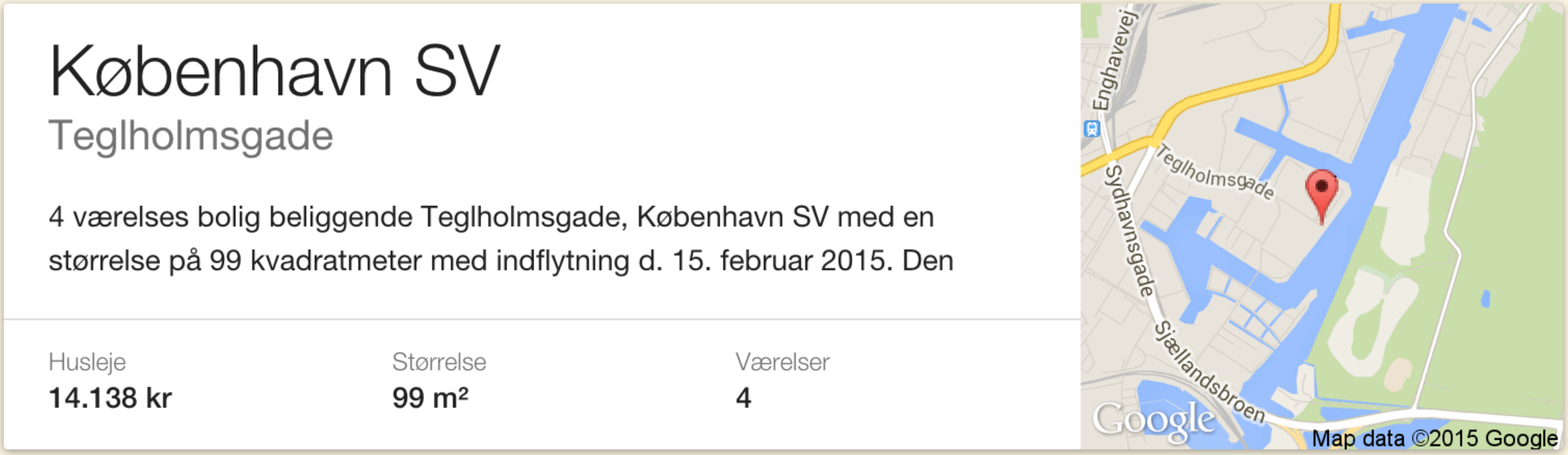
```
import React from 'react';

let Example = React.createClass({
  render: function() {
    return (
      <div>
        <h1>Haters Gonna Hate</h1>
        <p>Look {this.props.name}, HTML in my JavaScript!</p>
      </div>
    );
  }
});

React.render(<Example name='everybody!' />, document.body);
```

```
let data = {
  title: 'København SV',
  description: '4 værelses bolig beliggende Teglhomsgade...'
  ...
};

React.render( <Dwelling {...data}/>, document.body );
```



København SV

Teglholmsgade

4 værelses bolig beliggende Teglholmsgade, København SV med en størrelse på 99 kvadratmeter med indflytning d. 15. februar 2015. Den

Husleje	Størrelse	Værelser
14.138 kr	99 m²	4

Everything as a component

```
import GoogleMap from '../shared/google-map';
import Details from './details.component';
import Title from './title.component';
import Description from './description.component';

let Dwelling = React.createClass({
  render: function() {
    return (
      <div>
        <Title main={this.props.city} sub={this.props.street} />
        <Description text={this.props.teaser}/>
        <GoogleMap lat={this.props.lat} lng={this.props.lng} />
        <Details data={this.props.details}/>
      </div>
    );
  }
});
```

Cons

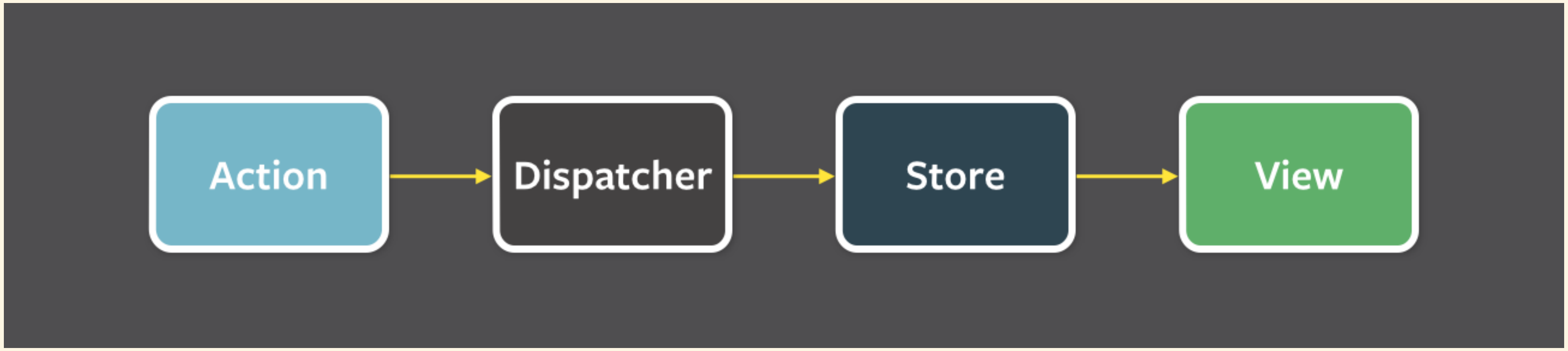
- New framework
- You decide the stack

Pros

- You decide the stack
- Minimal API to learn
- Encourages vanilla javascript
- Feels like writing javascript
- Server Side Rendering
- Testing

Flux

"Application architecture for building user interfaces"



Multiple Implementations

- Flux
- Fluxxor
- DeLorean
- Marty

RefluxJS

```
╔═══════════╗       ╔═════════╗       ╔═══════════════════╗
║  Actions  ║──────>║ Stores  ║──────>║ View Components   ║
╚═══════════╝       ╚═════════╝       ╚═══════════════════╝
      ^                                         │
      └─────────────────────────────────────────┘
```

Action

```
import Reflux from 'reflux';

let Actions = Reflux.createActions({
  fetchData: { asyncResult: true }
});

Actions.fetchData.listenAndPromise(
  num => xhr.get( `/latest?amount=${num}` )
);

export default Actions;
```

Store

```
import Reflux from 'reflux';
import Actions from './example.actions';

let Store = Reflux.createStore({
  listenables: Actions,

  onFetchDataCompleted: function(response) {
    this.data = response.data;
    this.trigger(this.data);
  },

  onFetchDataFailed:
    response => console.error(response.data)
});

export default Store;
```

View

```
import React from 'react';
import Reflux from 'reflux';
import Store from './example.store';
import { fetchData } from './example.actions';
import { Dwelling, Loading } from './dwelling.component';

React.createClass({
  mixins: [ Reflux.connect(Store, 'data') ],
  componentWillMount: fetchData,
  render: function() {
    if ('data' in this.state) {
      return this.state.data.map( data => <Dwelling {...data} /> );
    }
    else return <Loading />
  }
});
```

Cons

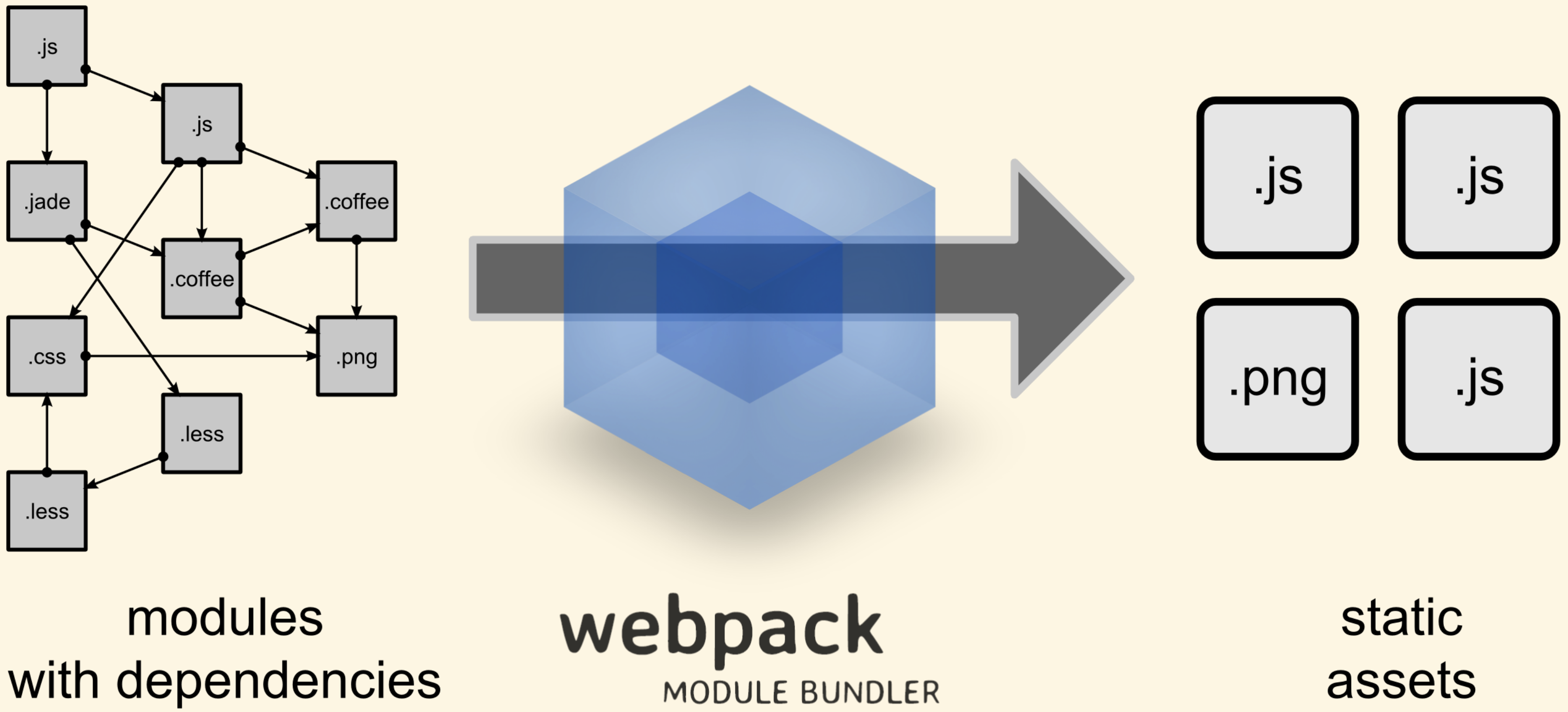
- No 'best practices'
- Sparse documentation
- Server side rendering can be tricky

Pros

- RefluxJS makes it easier to decouple domains
- Actions tells the truth
- Flat learning curve
- 'WYSIWYG'

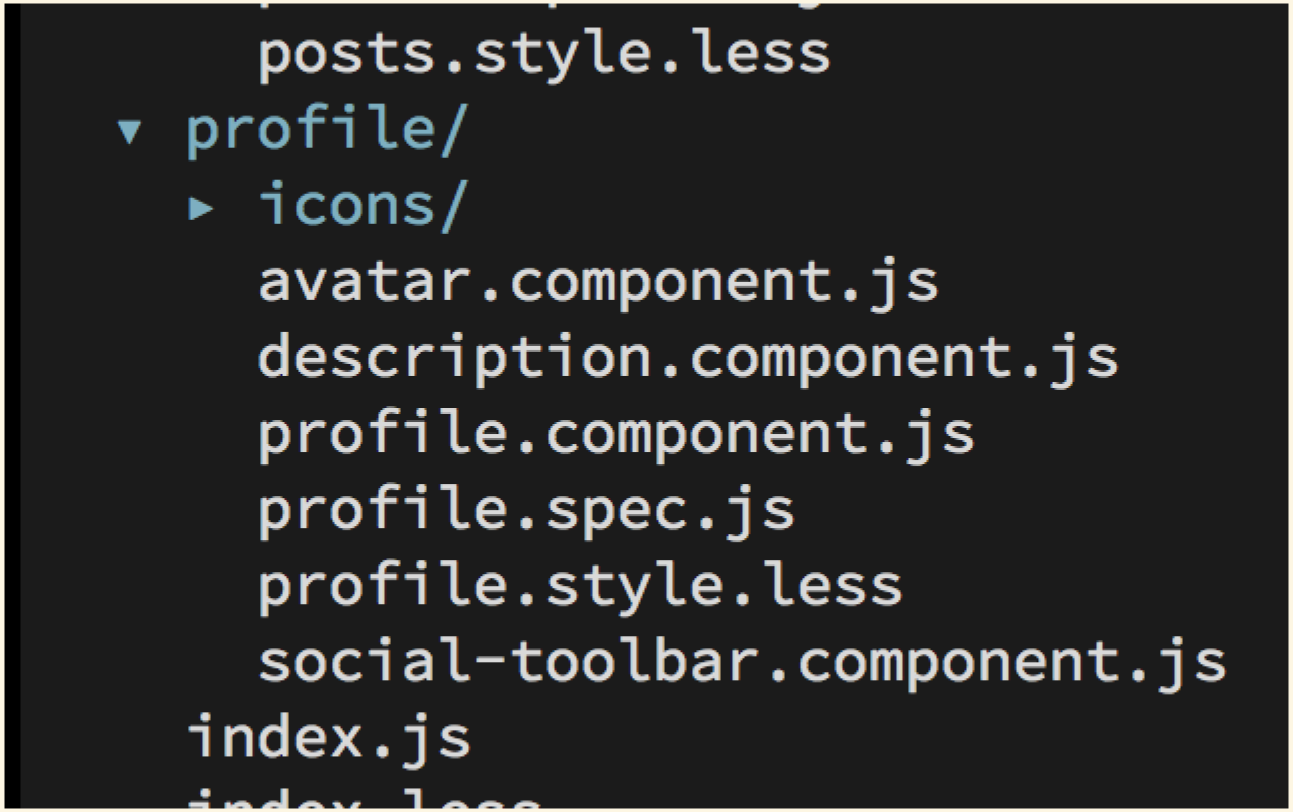
Webpack

module bundler


.js
.js
.jade
.coffee
.coffee
.css
.png
.less
.less
modules
with dependencies
webpack
MODULE BUNDLER
.js
.js
.png
.js
static
assets

- Like Browserify
- Transforms are called loaders
- Works with everything. Commonjs, AMD

Encapsulated Components



```
    posts.style.less
▾ profile/
  ▸ icons/
    avatar.component.js
    description.component.js
    profile.component.js
    profile.spec.js
    profile.style.less
    social-toolbar.component.js
  index.js
```

```
import 'example.style.less';
import github from 'github.icon.svg';
import React from 'react';

let Github = React.createClass({
  render: function() {
    return (
      <a href='https://github.com/chriskjaer'>
        <i dangerouslySetInnerHTML={{ __html: github }} />
        Github
      </a>
    );
  }
});
```

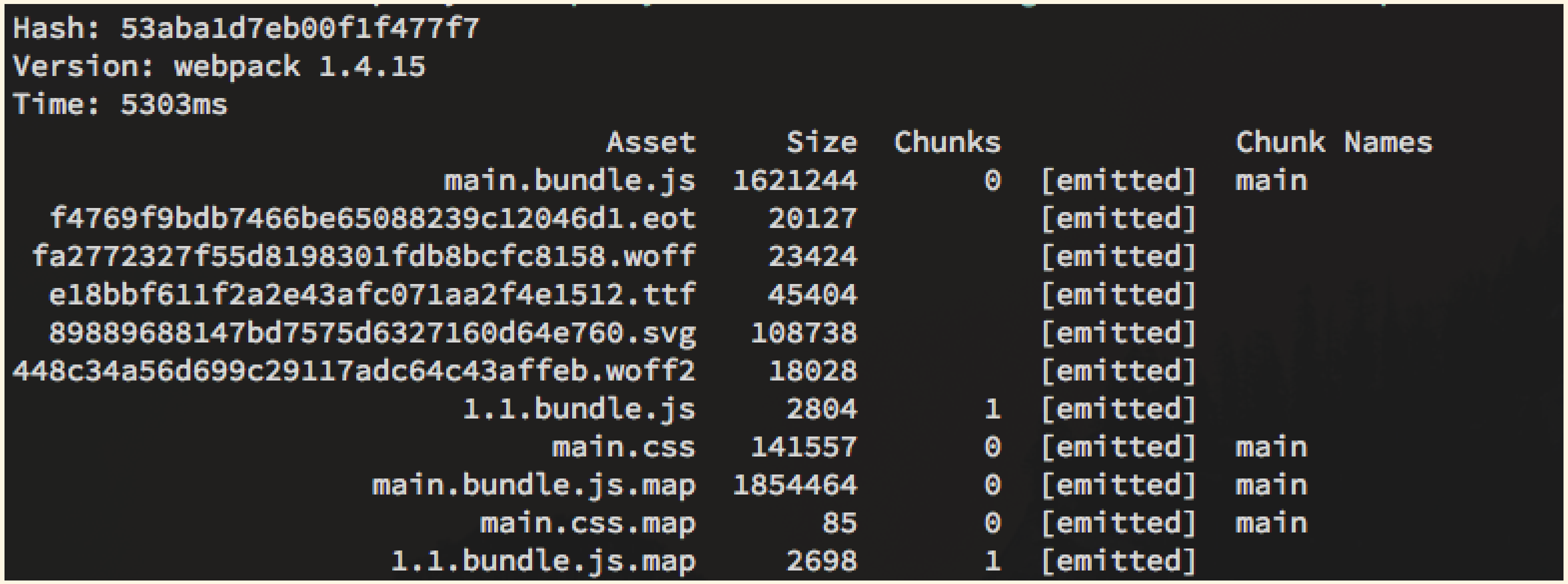
Code splitting & Async loading

```
import { Mixin } from 'react-proxy!./github.component';

let GithubAsync = React.createClass({
  mixins: [Mixin],

  renderUnavailable: function() {
    return <p>Loading...</p>;
  }
});
```

Gives



```
Hash: 53aba1d7eb00f1f477f7
Version: webpack 1.4.15
Time: 5303ms
                                Asset     Size  Chunks             Chunk Names
                       main.bundle.js  1621244       0  [emitted]  main
  f4769f9bdb7466be65088239c12046d1.eot    20127          [emitted]
 fa2772327f55d8198301fdb8bcfc8158.woff    23424          [emitted]
  e18bbf611f2a2e43afc071aa2f4e1512.ttf    45404          [emitted]
  89889688147bd7575d6327160d64e760.svg   108738          [emitted]
448c34a56d699c29117adc64c43affeb.woff2    18028          [emitted]
                          1.1.bundle.js     2804       1  [emitted]
                               main.css   141557       0  [emitted]  main
                   main.bundle.js.map  1854464       0  [emitted]  main
                           main.css.map       85       0  [emitted]  main
                      1.1.bundle.js.map     2698       1  [emitted]
```

Can also export your app as:

- node module
- webworker
- browser
- Commonjs / AMD / UMD

Cons

- Configuration over code (Like Grunt)
- Loaders strays from traditional require
- Documentation could be better

Pros

- Works out of the box with existing solutions
- Extremely good for SPA
- React-hot-loader
- Code splitting
- Webpack + Make
    - Eliminates the need for Grunt/Gulp

What did we learn?

Cons with this stack

- No large opensource examples
  - Best practices
  - Workflow
  - Composition
- Dependencies changes fast
- Running the app on the server
  - Async data
  - DOM Dependent API's

Pros with this stack

- (Re)flux - easy to reason about data
- Isomorphic
- Easy i18n (React Intl)
- Great Routing (React Router)
- Testing Components
- No Context Switching (javascript everywhere)
- Vanilla Javascript
- Multiple async app bundles (Webpack)

Near future

- React Native
- Relay + GraphQL
- Immutability.js

Thank you!

Github @chriskjaer

LinkedIn @chriskjaer

Twitter: @ckjaer