SCALE YOUR REACT APP FOR YOUR FIRST MILLION USERS

BEFORE WE START

visit http://git.io/vdrRR

clone the repo

run `npm install`

BEFORE WE START

```
// run an exercise
npm start
npm test
```

BEFORE WE START

```
~/Sites/Keynotes/react-days-berlin-2017 master
❯ npm start

> react-days-berlin-2017@1.0.0 start /Users/danielsneijers/Sites/Keynotes/react-days-ber
lin-2017
> node index.js

? Which exercise do you want to run? (Use arrow keys)
❯ Exercise 1: Higher Order Components
  Exercise 2: Functional Programming
  Exercise 3: State and Mobx
```

```
~/Sites/Keynotes/react-days-berlin-2017 master
❯ npm test

> react-days-berlin-2017@1.0.0 test /Users/danielsneijers/Sites/Keynotes/react-days-berl
in-2017
> node index.js --test

? Which test suite do you want to run? (Use arrow keys)
❯ Exercise 1: Higher Order Components
  Exercise 2: Functional Programming
  Exercise 3: State and Mobx
```

👋 I'M DANIEL

- Lead Frontend at Blendle
- 🦤 @furryrobot
- 🖥️ danielsneijers.com
- github?

Blendle

- Funded startup (2014) company from Holland
- Try to solve the problem of accessing and discovering quality journalism
- Access to 120+ newspapers and magazines
- Web: React app serving ~1,5 million users in 3 countries

Dinsdag, 10 oktober

Goedemiddag Daniel,

Hoe ambitie een islamitische agent verdacht maakte, waarom afkomen van antidepressiva een hel is, Thijs Zonneveld gaat los op teleurgestelde Oranjefans en niemand weet zich raad met autistische Daphne (27).

Staff Picks

Deze stukken vinden wij extra de moeite waard

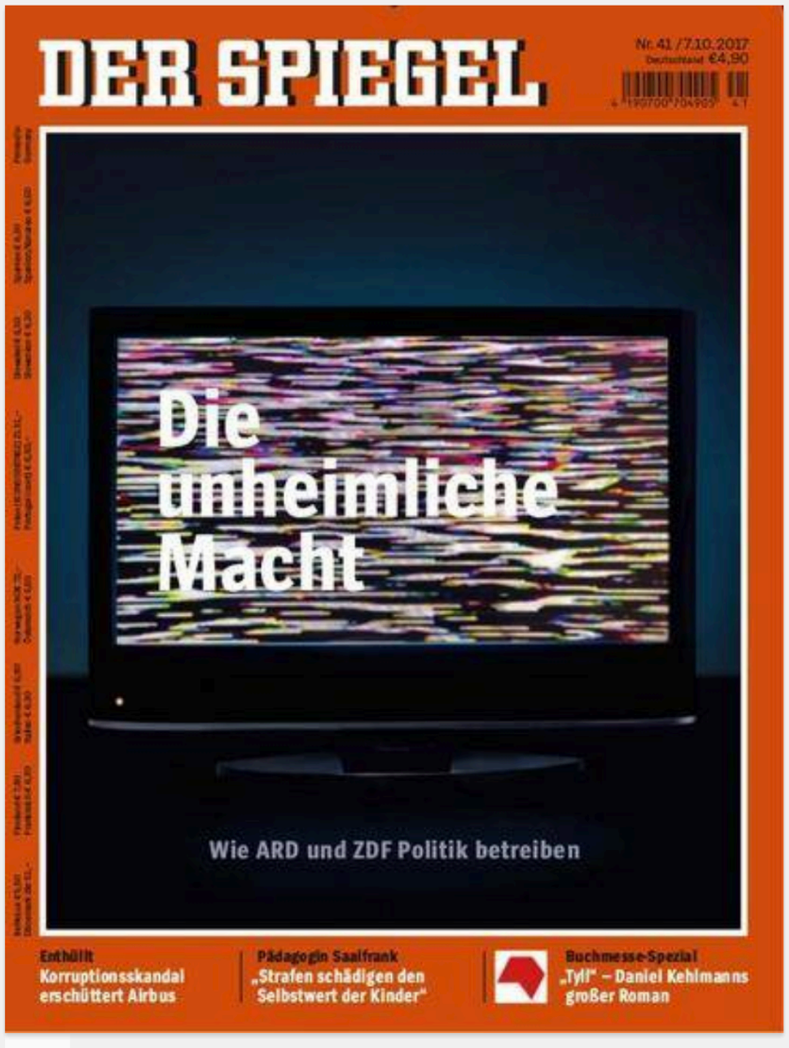
THE NEW YORKER

maandag 11 september 2017 · 31 minuten lezen

THE TRIALS OF A MUSLIM COP

Bobby Hadid joined the N.Y.P.D. after 9/11, to protect his new country. But when he questioned the force's tactics his life began to erode.

By Rachel Aviv


DER SPIEGEL
Nr. 41 / 7.10.2017
Deutschland €4,90
Die unheimliche Macht
Wie ARD und ZDF Politik betreiben
Enthüllt Korruptionsskandal erschüttert Airbus
Pädagogin Saalfrank „Strafen schädigen den Selbstwert der Kinder"
Buchmesse-Spezial „Tyll" – Daniel Kehlmanns großer Roman


Frankfurter Allgemeine
ZEITUNG FÜR DEUTSCHLAND
Berlin: Anklage Steudtners nicht akzeptabel
Anfang ohne Zauber
FDP und Grüne kritisieren Einigung der Union in der Flüchtlingspolitik
Obergrenze
Heute
Erfolgreiche Regierungsbildung in Den Haag
Robert Menasse erhält Deutschen Buchpreis
Madrid: Puigdemont ist ein Feigling
Durchsuchung bei AfD vor der Landtagswahl
Wirtschaftsnobelpreis für Richard Thaler


stern
STERN EXKLUSIV
DIE AKTE BECKE
Gläubiger fordern jetzt 61 Millionen Euro von ihm: wem er Geld schu
was er noch auf dem Konto hat – und was ihn noch retten kann

THIS IS YOUR BOSS

PHP CEO
@PHP_CEO

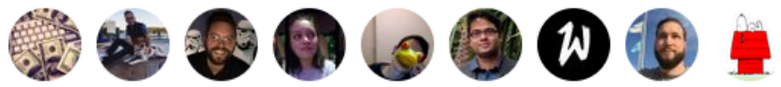
Following

I'VE ANALYSED THE MARKET

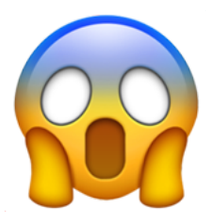
OUR NEXT PRODUCT WILL BE A MACHINE LEARNING POWERED EDUCATION PLATFORM FOR RECRUITING JAVASCRIPT SECURITY DEVOPS

1:39 PM - 21 Mar 2016

474 Retweets **554** Likes



13 474 554



HOW TO BUILD A UNICORN 10X ROCKSTAR NINJA APP 🦄

- Application setup
- State management at scale
- Legacy code and refactoring strategies
- Deployment strategies
- Where to go next

FOLDER STRUCTURE

APPLICATION SETUP

RAILS STYLE

```
actions/
    CommandActions.js
    UserActions.js
components/
    Header.js
    Sidebar.js
    Command.js
    CommandList.js
    CommandItem.js
    CommandHelper.js
    User.js
    UserProfile.js
    UserAvatar.js
containers/
    App.js
    Command.js
    User.js
reducers/
    index.js
    command.js
    user.js
routes.js
```

RAILS STYLE

```
actions/
    CommandActions.js
    ProductActions.js   ← Here
    UserActions.js
components/
    Header.js
    Sidebar.js
    Command.js
    CommandList.js
    CommandItem.js
    CommandHelper.js
    Product.js          ← Here
    ProductList.js      ← Here
    ProductItem.js      ← Here
    ProductImage.js     ← Here
     ...
containers/
    App.js
    Command.js
    Product.js          ← Here
    User.js
reducers/
    index.js
    foo.js
    bar.js
    product.js          ← Here
routes.js
```

RAILS STYLE

```
src/
    actions/
        CommandActions.js
        ProductActions.js
        UserActions.js
    components/
        Header.js
        Sidebar.js
        Command.js
         ...
    containers/
        App.js
        Command.js
        Product.js
        User.js
    reducers/
        index.js
        foo.js
        bar.js
        product.js
    routes.js
tests/
    actions/
        CommandActions.js
         ProductActions.js
        UserActions.js
    …
```

RAILS STYLE

Pros

- Familiar pattern
- Clear distinction of roles

Cons

- Heavily leaning towards MVC
- Related files are far away
- Adding files is a drag

"DUCKS" STYLE

```
components/
    Header.js
    Sidebar.js
    Command.js
    CommandList.js
    CommandItem.js
    CommandHelper.js
    User.js
    UserProfile.js
    UserAvatar.js
containers/
    App.js
    Command.js
    User.js
redux/
    modules/
       command.js
       user.js
    index.js
routes.js
```

"DUCKS" STYLE

```
components/
    Header.js
    Sidebar.js
    Command.js
    CommandList.js
    CommandItem.js
    CommandHelper.js
    User.js
    UserProfile.js
    UserAvatar.js
containers/
    App.js
    Command.js
    User.js
redux/
    modules/
       command.js
       user.js
    index.js
routes.js
```

"DUCKS" STYLE

Pros

- Clear distinction of roles
- Less clutter than Rails Style
- Better distinction between domain/shared components and logic

Cons

- Related files still far away
- Related files cas be far away

DOMAIN STYLE

```
command/
    actions.js
    component.js
    container.js
    index.js
    reducer.js
    style.css
    tests.js
user/
    actions.js
    component.js
    container.js
    index.js
    reducer.js
    style.css
    tests.js
```

DOMAIN STYLE

```
command/
    actions.js
    component.js
    container.js
    index.js
    reducer.js
    style.css
    tests.js
user/
    actions.js
    component.js
    container.js
    index.js
    reducer.js
    style.css
    tests.js
product/            ← Here
    actions.js      ← Here
    component.js    ← Here
    container.js    ← Here
    index.js        ← Here
    reducer.js      ← Here
    style.css       ← Here
    tests.js        ← Here
```

DOMAIN STYLE

```
// user/index.js
export { default as actions } from './actions';
export { default as reducer } from './reducer';

// or without babel
export * from './actions';
export * from ‘./reducer';
```

```
// Some other file
import { actions, reducer } from '../user';
```

DOMAIN STYLE

Pros

- Related files next to each other
- Short import paths
- Clear distinction of domains
- Easy to add files

Cons

- Errors in “Bucket files” are easy to make
- Not clear where to put shared components or logic

MIXED STYLE

```
shared/
    components/
        someComponent.js
    scenes/
        mainLayout.js
domains/
    user/
        __tests__/
        reducer.js
        actions.js
    components/
        __tests__/
            userProfile.js
            userAvatar.js
        userProfile.js
        userAvatar.js
        style.css
    scenes/
        userSettings.js
    actions.js
    index.js
    reducer.js
```

MIXED STYLE

Pros

- Related files next to each other
- Short import paths
- Clear distinction of domains
- Easy to add files
- Component/Scene separation

Cons

- Errors in “Bucket files” are easy to make
- Can be confusing for newcomers

ARCHITECTURE

APPLICATION SETUP

PHP CEO
@PHP_CEO

Following

ALL OF OUR SOFTWARE IS BUILT WITHOUT ANY DEPENDENCY ON STATE WE CALL IT OBJECTIVISM-ORIENTED PROGRAMMING

1:10 PM - 16 Aug 2015

223 Retweets **243** Likes

7 223 243

🤤 DUMB COMPONENTS

🤓 SMART CONTAINERS

CONTAINERS

- Are about how things work
- Provide (formatted) data
- Provide action handlers
- Don’t have to emit a DOM

COMPONENTS

- Are about how things look
- No dependencies on store/ actions
- Receive clear data and actions via Props
- Preferably have no state

MINIMAL COMPONENT

```jsx
import React from 'react';
import { node } from 'prop-types';

const Button = ({ children }) => <button>{children}</button>

Button.propTypes = {
  children: node.isRequired
};

export default Button
```

MINIMAL COMPONENT

```jsx
import React from 'react';
import { node } from 'prop-types';

const Button = ({ children }) => <button>{children}</button>

Button.propTypes = {
  children: node.isRequired
};

export default Button
```

MINIMAL COMPONENT

```jsx
import React from 'react';
import { node } from 'prop-types';

const Button = (props) => {
  return <button>{props.children}</button>;
};

Button.propTypes = {
  children: node.isRequired
};

export default Button
```

MINIMAL COMPONENT

```jsx
import React from 'react';
import { node, func, string } from 'prop-types';

const Button = ({ children, onClick, className }) => {
  return (
    <button className={className} onClick={onClick}>
      {children}
    </button>
  );
};

Button.propTypes = {
  children: node.isRequired,
  onClick: func.isRequired,
  className: string
};

export default Button
```

MINIMAL COMPONENT

```jsx
import React from 'react';
import { node, func, string } from 'prop-types';

const Button = ({ children, onClick, className }) => {
  return (
    <button className={className} onClick={onClick}>
      {children}
    </button>
  );
};

Button.propTypes = {
  children: node.isRequired,
  onClick: func.isRequired,
  className: string
};

export default Button
```

MINIMAL CONTAINER

```
import { logIn } from './actions'
import { bindActionCreators } from 'redux'

export function mapStateToProps (state) {
  const { authenticated } = state.auth
  return { authenticated }
}

export function mapDispatchToProps (dispatch) {
  const actions = { logIn }
  return bindActionCreators(actions, dispatch)
}
```

MINIMAL CONTAINER

```
import { logIn } from './actions'
import { bindActionCreators } from 'redux'

export function mapStateToProps (state) {
  const { authenticated } = state.auth
  return { authenticated }
}

export function mapDispatchToProps (dispatch) {
  const actions = { logIn }
  return bindActionCreators(actions, dispatch)
}
```

MINIMAL CONTAINER

```
import { logIn } from './actions'
import { bindActionCreators } from 'redux'

export function mapStateToProps (state) {
  const { authenticated } = state.auth
  return { authenticated }
}

export function mapDispatchToProps (dispatch) {
  const actions = { logIn }
  return bindActionCreators(actions, dispatch)
}
```

HIGHER ORDER COMPONENTS

- an HOC composes the original component
- wraps it in a container component
- a pure function with zero side-effects

HOC

```
function logProps(WrappedComponent) {
  return class extends React.Component {
    componentWillReceiveProps(nextProps) {
      console.log('Current props: ', this.props);
      console.log('Next props: ', nextProps);
    }

    render() {
      return <WrappedComponent {...this.props} />;
    }
  }
}
```

HOC

```
function logProps(WrappedComponent) {
  return class extends React.Component {
    componentWillReceiveProps(nextProps) {
      console.log('Current props: ', this.props);
      console.log('Next props: ', nextProps);
    }

    render() {
      return <WrappedComponent { ... this.props} />;
    }
  }
}
```

HOC

```
function logProps(WrappedComponent) {
  return class extends React.Component {
    componentWillReceiveProps(nextProps) {
      console.log('Current props: ', this.props);
      console.log('Next props: ', nextProps);
    }

    render() {
      return <WrappedComponent { ... this.props} />;
    }
  }
}
```

HOC

```
function logProps(WrappedComponent) {
  return class extends React.Component {
    componentWillReceiveProps(nextProps) {
      console.log('Current props: ', this.props);
      console.log('Next props: ', nextProps);
    }

    render() {
      return <WrappedComponent {...this.props} />;
    }
  }
}
```

HOC

```
function logProps(WrappedComponent) {
  return class extends React.Component {
    componentWillReceiveProps(nextProps) {
      console.log('Current props: ', this.props);
      console.log('Next props: ', nextProps);
    }

    render() {
      return <WrappedComponent {...this.props} />;
    }
  }
}
```

HOC

```
function logProps(WrappedComponent) {
  return class extends React.Component {
    componentWillReceiveProps(nextProps) {
      console.log('Current props: ', this.props);
      console.log('Next props: ', nextProps);
    }

    render() {
      return <WrappedComponent {...this.props} />;
    }
  }
}
```

HOC

```
const App = (props) ⟹ <h1>App</h1>;

export default logProps(App);
```

OBJECT ORIENTED PROGRAMMING

VS

FUNCTIONAL PROGRAMMING

“Should I use OOP or FP in this project?”

–Some developer

“Both!”

–Me

Dan Abramov
@dan_abramov

Following

The oddest thing I heard in a while: a company went “classless” and engineers come up with convoluted workarounds for class-based  features

11:58 AM - 28 Sep 2017

41 Retweets **196** Likes

19 41 196

WHEN YOU GOOGLE FUNCTIONAL PROGRAMMING...

$$
\begin{aligned}
\mathbf{return}: &\quad A \to S \to \mathrm{M}(A \times S) = a \mapsto s \mapsto \mathbf{return}\,(a, s) \\
\mathbf{bind}: &\quad (S \to \mathrm{M}(A \times S)) \to (A \to S \to \mathrm{M}(B \times S)) \to S \to \mathrm{M}(B \times S) = m \mapsto k \mapsto s \mapsto \mathbf{bind}\,(m\,s)\,((a, s') \mapsto k\,a\,s') \\
\mathbf{lift}: &\quad \mathrm{M}\,A \to S \to \mathrm{M}(A \times S) = m \mapsto s \mapsto \mathbf{bind}\,m\,(a \mapsto \mathbf{return}\,(a, s))
\end{aligned}
$$

ABOUT FUNCTIONAL PROGRAMMING

- Pure functions
- Higher order functions
- Composition of objects and functions
- Immutability
- Recursion

PURE FUNCTIONS

- Given the same input, will always return the same output.
- Produces no side effects.
- Relies on no external mutable state.

IMPURE FUNCTION

```
let number = 5;

function addToFiveAndLog(value) {
    number += value;
    console.log(number);
}

addToFiveAndLog(5) // 5
```

IMPURE FUNCTION

```
let number = 5;

function addToFiveAndLog(value) {
    number += value;
    console.log(number);
}

addToFiveAndLog(5) // 5
addToFiveAndLog(5) // ?
```

PURE FUNCTION

```
const number = 5;

function addToFiveAndLog(value) {
    const newValue = number + 5
    console.log(newValue);
}

addToFiveAndLog(5) // 5
addToFiveAndLog(5) // 5
addToFiveAndLog(5) // 5
```

HIGHER ORDER FUNCTIONS

- Basically a function that accepts a function as an argument

HIGHER ORDER FUNCTIONS

```
const array = [1, 2, 3];

// so hard to read!
for (let i = 0; i < array.length; i++) {
  var current = array[i];
  console.log(current);
}
```

HIGHER ORDER FUNCTIONS

```
const array = [1, 2, 3];

// so hard to read!
for (let i = 0; i < array.length; i++) {
  var current = array[i];
  console.log(current);
}

// like reading poetry
array.forEach(entry ⇒ console.log(entry));
```

HIGHER ORDER FUNCTIONS

```
const array = [1, 2, 3];

function doSomething (value) {
  console.log(value)
}

// so advanded !!
array.forEach(doSomething);
```

OBJECT COMPOSITION

```
const me = {
  name: 'Daniel',
  age: 30
}

const job = {
  title: 'developer',
  level: 'rockstar'
}

const profile = {  ...me, job }

//  {
//   name: 'Daniel',
//   age: 30,
//   job: {
//     title: 'developer',
//     level: 'rockstar unicorn'
//   }
// }
```

FUNCTION COMPOSITION

```
function double(n) {
  return n * 2;
}
```

```
function squared(n) {
  return n * n;
}

double(squared(5)); // 100
```

FUNCTION COMPOSITION

```
import { compose } from 'lodash'

function double(n) {
  return n * 2;
}

function squared(n) {
  return n * n;
}

const doubleSquared = compose(double, squared);

doubleSquared(5); // 100
```

“When would that ever be useful?”

–Some guy

FUNCTION COMPOSITION

```
const App = (props) ⟹ <h1>App</h1>;

export default logProps(App);
```

FUNCTION COMPOSITION

```
const App = (props) ⇒ <h1>App</h1>;

export default withUserData(logProps(App));
```

FUNCTION COMPOSITION

```
const App = (props) ⇒ <h1>App</h1>;

export default withRouter(withUserData(logProps(App)));
```

FUNCTION COMPOSITION

```
import { compose } from 'lodash';

const App = (props) ⇒ <h1>App</h1>;

const enhanced = compose(withRouter, withUserData, logProps);

export default enhanced(App);
```

ABOUT FUNCTIONAL PROGRAMMING

- ~~*Pure functions*~~
- ~~*Higher order functions*~~
- ~~*Composition of objects and functions*~~
- Immutability
- Recursion

LAST BUT NOT LEAST

- There are a lot of tools that’ll make your life easier
- Prettier => auto format your code
- ESLint => never forget a comma or closing bracket
- Jest => Unit tests without hassle
- Lint-staged => easy pre-commit hooks

AUTHENTICATION, WITH JWT

STATE MANAGEMENT AT SCALE

PHP CEO
@PHP_CEO

Following

WE SPENT 18 MONTHS MIGRATING FROM A MONOLITH TO MICROSERVICES

RESULT:

- GITHUB GETS PAID FOR MORE PRIVATE REPOS

- FIND/REPLACE IS HARDER

5:16 AM - 30 May 2015

1,872 Retweets **1,494** Likes

28 1.9K 1.5K

WHAT IS A JWT?

“JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object.”

– https://jwt.io/introduction/

- A base64 encoded string
- Standardised format (header.payload.signature)
- Stateless
- Reusable
- Secure: signed by the backend

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9.TJVA95OrM7E2cBab30RMHrHDcEfxjoYZgeFONFh7HgQ

Header: algorithm & token type

Payload: data

Signature

JWT HEADER

```
new Buffer(
  ‘eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9’,
  ‘base64’
).toString(‘ascii')


// Output
'{"alg":"HS256","typ":"JWT"}'
```

JWT PAYLOAD

```
new Buffer(
  ‘eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9’,
  ‘base64'
).toString('ascii')


// output
'{"sub":"1234567890","name":"John Doe","admin":true}'
```

JWT SIGNATURE

```
new Buffer(
  ‘TJVA95OrM7E2cBab30RMHrHDcEfxjoYZgeFONFh7HgQ’,
  ‘base64'
).toString('ascii')


// output
'L\u0015@w\u0013+316p\u0016\u001b_DL\u001e1CpGq\u000e\u0006\u0019\u0001aN4X{\u001e\u0004'
```

BONUS QUESTION

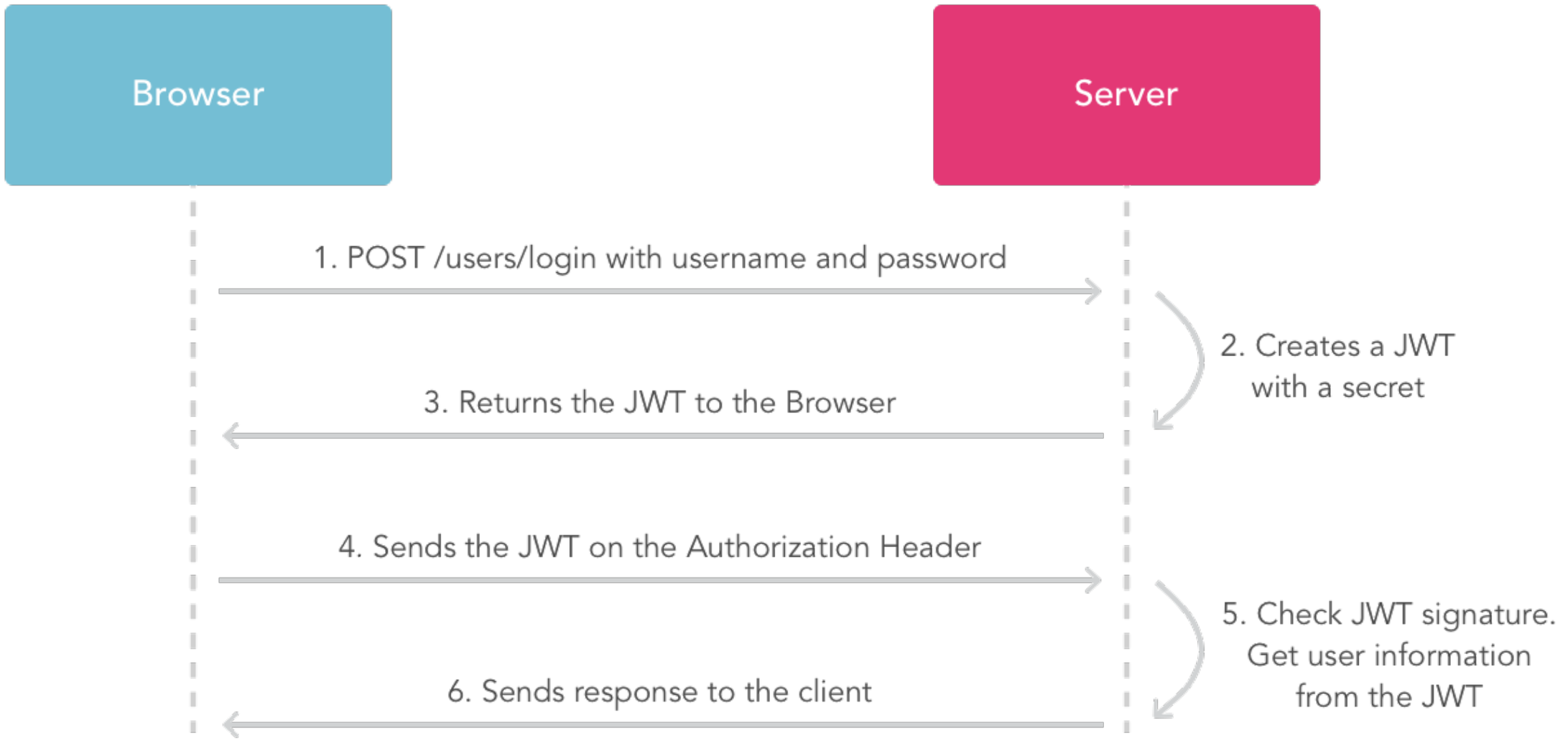
JWT SIGNATURE

```
new Buffer(‘ey’, ‘base64').toString('ascii')


// output
‘{'
```

JWT FLOW

source: https://jwt.io/introduction/



ADDING JWT TO YOUR CODE

```
const headers = new Headers({ Authorization: 'Bearer <token>' })

fetch('http://example.com', {
  method: 'get',
  headers
})
.then(doSomething)
.catch(doSomething)
```

ADDING JWT TO YOUR CODE

```
const headers = new Headers({ Authorization: 'Bearer <token>' })

fetch('http://example.com', {
  method: 'get',
  headers
})
.then(doSomething)
.catch(doSomething)
```

ADDING JWT TO YOUR CODE

```
const headers = new Headers({ Authorization: 'Bearer <token>' })

fetch('http://example.com', {
  method: 'get',
  headers
})
.then(doSomething)
.catch(doSomething)
```

STATE MANAGEMENT WITH MOBX

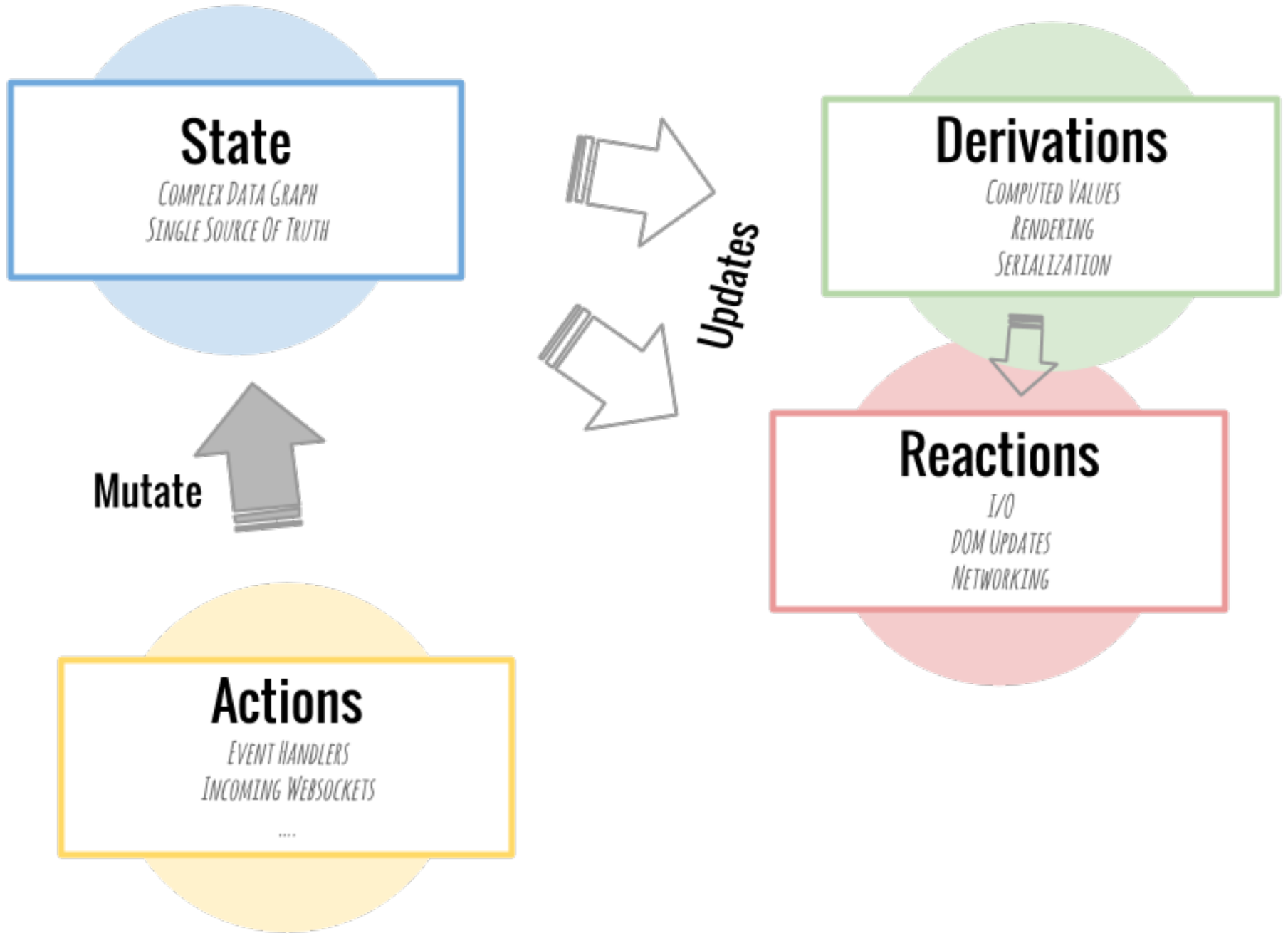
STATE MANAGEMENT AT SCALE

WHAT IS MOBX?

“Simple, scalable state management”

— https://github.com/mobxjs/mobx

WHAT IS MOBX?


State
Complex Data Graph
Single Source Of Truth
Derivations
Computed Values
Rendering
Serialization
Updates
Reactions
I/O
DOM Updates
Networking
Mutate
Actions
Event Handlers
Incoming Websockets
....

REDUX

- State is immutable
- One store
- Only actions mutate store
- actions.js > reducer.js

MOBX

- State is mutable
- Multiple stores
- Everyone can mutate store*
- store.js

A MOBX STORE

```
class CounterStore {
  @observable counter = 0;

  @computed
  get counterWithDecimals() {
    return this.counter.toFixed(2);
  }

  @action
  increaseCounter = () => {
    this.counter += 1;
  };
}
```

A MOBX STORE

```
class CounterStore {
  @observable counter = 0;

  @computed
  get counterWithDecimals() {
    return this.counter.toFixed(2);
  }

  @action
  increaseCounter = () => {
    this.counter += 1;
  };
}
```

A MOBX STORE

```
class CounterStore {
  @observable counter = 0;

  @computed
  get counterWithDecimals() {
    return this.counter.toFixed(2);
  }

  @action
  increaseCounter = () => {
    this.counter += 1;
  };
}
```

A MOBX STORE

```
class CounterStore {
  @observable counter = 0;

  @computed
  get counterWithDecimals() {
    return this.counter.toFixed(2);
  }

  @action
  increaseCounter = () => {
    this.counter += 1;
  };
}
```

STORE INJECTION

```
// React root
ReactDOM.render(
  <Provider counterStore={counterStore}>
    <App />
  </Provider>,
  rootEl,
);


// Some component
@inject('counterStore')
@observer
class SomeComponent extends Component {
  //  ...
}
```

STORE INJECTION

```
// React root
ReactDOM.render(
  <Provider counterStore={counterStore}>
    <App />
  </Provider>,
  rootEl,
);


// Some component
@inject('counterStore')
@observer
class SomeComponent extends Component {
  //  ...
}
```

STORE INJECTION

```
// React root
ReactDOM.render(
  <Provider counterStore={counterStore}>
    <App />
  </Provider>,
  rootEl,
);



// Some component
@inject('counterStore')
@observer
class SomeComponent extends Component {
  //  ...
}
```

STORE INJECTION

```
this.props.counterStore
```

MANAGING COMPLEX ASYNC

STATE MANAGEMENT AT SCALE

MULTIPLE OPTIONS

- redux-thunk or redux-promise
- Redux sagas
- RxJS
- Redux-observable

RXJS

- It’s kind of like lodash for events
- It’s kind of like all the array methods
- But it’s all async and reactive
- And it’s methods are called ‘operators’ (and there are a lot!)

RXJS

```
const button = document.querySelector('button');

button.addEventListener('click', () ⇒ console.log('Clicked!'));

// In react
<button onClick={() ⇒ { console.log('Clicked!')}} />
```

RXJS

```
const button = document.querySelector('button');

Rx.Observable.fromEvent(button, 'click')
  .subscribe(() => console.log('Clicked!'));
```

REDUX THUNK

```
export function someAction () {
  return dispatch ⇒ {
    dispatch(startRequest())

    fetch('http://example.com', { method: 'get', headers })
      .then(response ⇒ response.json())
      .then(data ⇒ {
        handleData(data)
        dispatch(handleSuccess(data))
        dispatch(push('/'))
      })
      .catch(error ⇒ dispatch(handleError(error)));
  }
}
```

RXJS

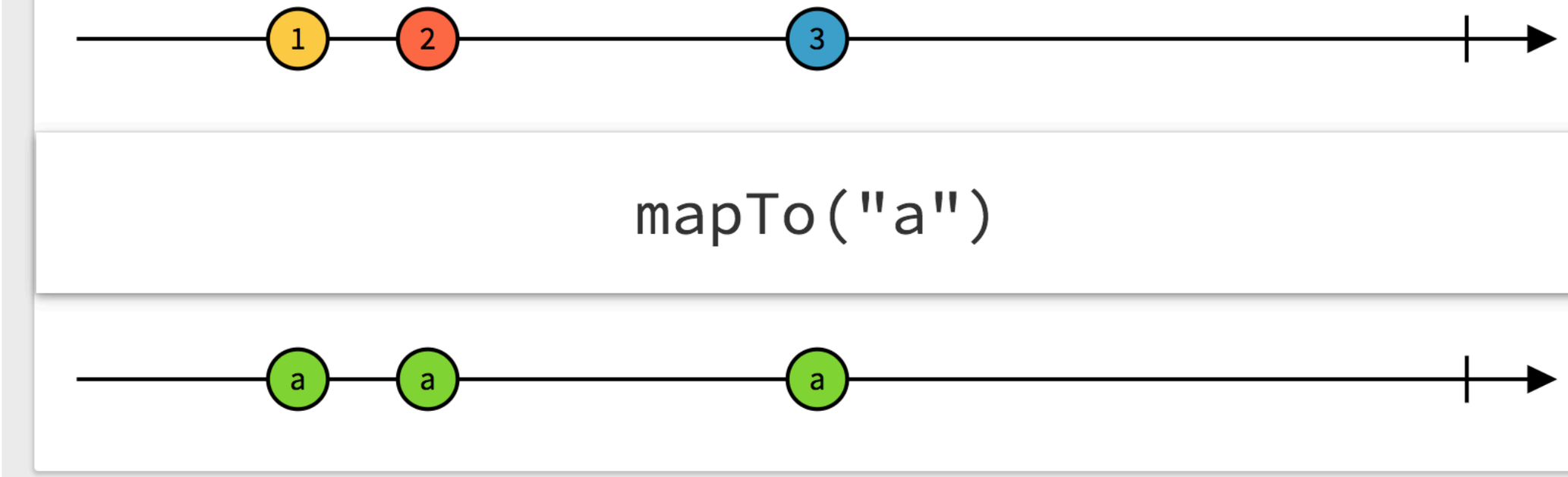
```
const button = document.querySelector('button');

const userClickedButton$ = Rx.Observable.fromEvent(button, 'click')

  userClickedButton$
    .map(() ⇒ Rx.Observable.fromPromise(
      fetch('https://example.com/')
    ))
    .subscribe(response ⇒ {
      // do something
    });
```

REDUX-OBSERVABLE

```
const pingEpic = action$ ⇒
  action$.ofType('PING')
    .delay(1000)
    .mapTo({ type: 'PONG' });
```



source: http://rxmarbles.com/#mapTo

RXJS

- It all starts with the observable
- Subscribe and operate on it when data arrives
- Keep it simple at first, use map, filter, pluck, take, etc

MANAGING DEPENDENCIES

LEGACY CODE AND REFACTORING STRATEGIES

BEFORE ADDING A DEPENDENCY, CHECK:

- Is it maintained actively?
- Are issues addressed?
- Are PR’s discussed actively?
- Is it well tested?
- What’s its footprint (in kb and performance)
- Does it really fit your needs?
- Is there an ES6+ alternative?

LOOK AT THE SOURCE

```
/*
  MIT License http://www.opensource.org/licenses/mit-license.php
  Author Tobias Koppers @sokra
*/
module.exports = function(content) {
  this.cacheable && this.cacheable();
  this.value = content;
  return "module.exports = " + JSON.stringify(content);
}
```

IS THERE AN ES6+ ALTERNATIVE?

```
const leftPad = require('left-pad')

leftPad('foo', 5)             // ⇒ "  foo"
leftPad('foobar', 6)          // ⇒ "foobar"
leftPad(1, 2, '0')            // ⇒ "01"
leftPad(17, 5, 0)             // ⇒ "00017"
```

IS THERE AN ES6+ ALTERNATIVE?

```
const leftPad = require('left-pad')

leftPad('foo', 5)           // ⇒ "  foo"
leftPad('foobar', 6)        // ⇒ "foobar"
leftPad(1, 2, '0')          // ⇒ "01"
leftPad(17, 5, 0)           // ⇒ "00017"

// Use ES6!
'foo'.padStart(5);          // ⇒ "  foo"
'foobar'.padStart(6);       // ⇒ "foobar"
'1'.padStart(2, "0");       // ⇒ "01"
'17'.padStart(5, "0");      // ⇒ "00017"
```

IS THERE AN ES6+ ALTERNATIVE?

```
const _ = require('lodash') // 17.6kb minified + gzipped

_.sum([4, 2, 8, 6]);
```

```
// or write your own, 67 bytes
function sum (values) {
  return values.reduce((total, current) => total + current, 0);
}

sum([4, 2, 8, 6]);
```

“I’ve got Webpack 3 with treeshaking, so it doesn’t matter!”

–Some of you

“Well, it depends…”

–Me

KEEP YOUR DEPENDENCIES UP TO DATE

- Try using tools like greenkeeper.io and gemnasium
- Reserve time for major version upgrades
- Consider your migration plan when introducing new dependencies
- … or use create-react-app 🙌

REFACTORING STRATEGIES

LEGACY CODE AND REFACTORING STRATEGIES

🔥 THE “BURN IT ALL”

👨‍🔬 THE “CLEVER ABSTRACTION”

🙃 THE “BOTTOM UP”

🔥 THE “BURN IT ALL” REFACTOR

- Throw it all away, build it from scratch
- Hardest to sell to the business
- Quite intense

🔥 THE “BURN IT ALL” REFACTOR

- Make a proposal
- Try to start a new route in application
- Enable it behind a feature flag
- Push code to production often
- Test a lot while developing!

👨‍🔬 THE “CLEVER ABSTRACTION” REFACTOR

- Isolate legacy code with an abstraction
- Gradually roll out in codebase
- Not as time consuming if well planned

👨‍🔬 THE "CLEVER ABSTRACTION" REFACTOR

- Make a proposal about the API
- Educate team
- Build the abstraction, use it in all new code
- Consider low profile parts of the codebase

🙃 THE “BOTTOM UP” REFACTOR

- Tool or dependency that coexists with current code
- Improves general quality of codebase
- Gradually roll out in codebase
- Can be used a lot

🙃 THE “BOTTOM UP” REFACTOR

- Educate team
- Use it in all new code
- Consider migration plan for older parts of code base
- Don’t try to fix a problem with multiple solutions at the same time

ALL REFACTORS

- Write tests
- Don’t forget to write some tests
- ~~Skip tests if you don’t have time~~
- **NO DON’T SKIP THIS, WRITE TESTS!!**

REMEMBER THIS GUY?

CONVINCING THE BUSINESS

- Help your boss/PO/team lead make informed decisions
- Explain what would happen if we don’t refactor, from a business perspective
- Explain the costs of delay from a technical perspective
- Explain it like I’m a five year old

DEPLOYMENT STRATEGIES

DEPLOYMENT STRATEGIES

GITHUB PAGES

GITHUB PAGES

- Free
- Easy to setup
- No server config
- No ftp, ssh
- High traffic, no problem

PAAS (LIKE NOW, BY ZEIT)

- Cheap
- No hassle
- Easy unlimited deployments
- Scaling/load balancing etc is handled for you
- No setup required for static websites, node.js and docker

DEPLOYING WITH NOW

```
$ now
```

PHP CEO
@PHP_CEO

Following

IF YOU'RE NOT USING DOCKER THEN BUDDY YOU'RE MISSING OUT ON THE BIGGEST REVOLUTION SINCE THE FRENCH ONE

ITS WHERE I MESSAGE ALL MY PALS NOW

5:59 PM - 9 Jul 2015

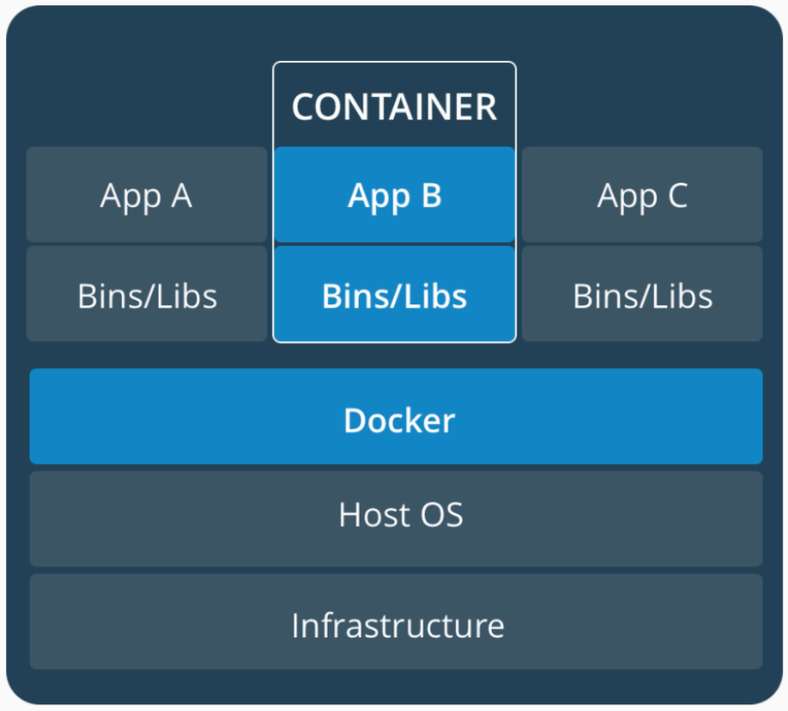
85 Retweets **81** Likes

1 85 81

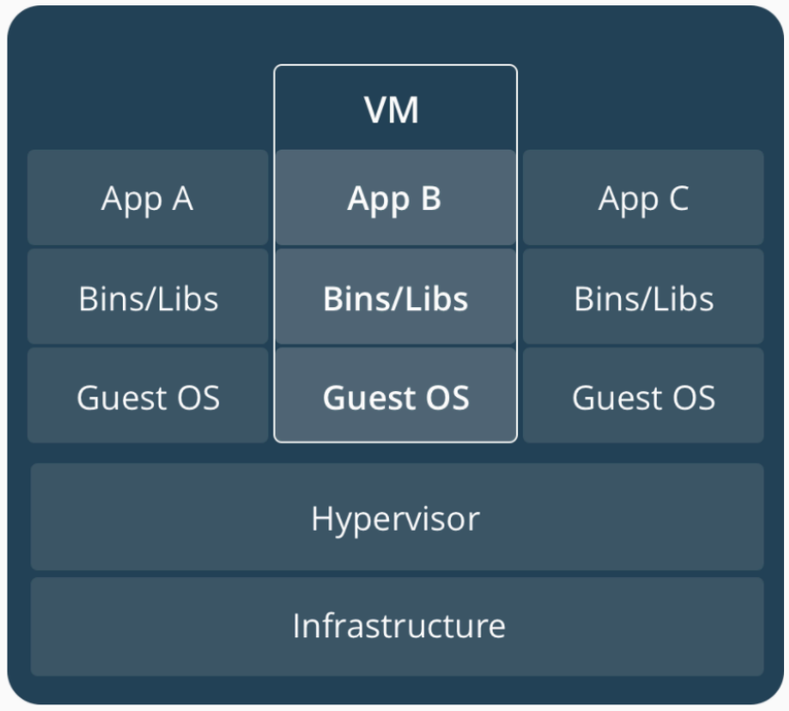
DOCKER, CLOUD SERVERS

- Can be either cheap or expensive
- A bit more setup
- Really flexible
- No compatibility issues between environments
- Works great with CI/CD

DOCKER INTRODUCTION



CONTAINERS



VIRTUAL MACHINES

CREATING A DOCKER FILE

```
# .Dockerfile in project root
FROM node:7
WORKDIR /app
COPY package.json /app
RUN npm install
COPY . /app
CMD node index.js
EXPOSE 8081
```

BUILDING A CONTAINER

```
$ docker build -t hello-world .
```

RUNNING A CONTAINER

```
$ docker run -p 8081:8081 hello-world
```

WHY A CI

- Automatically run your tests
- Automatically make builds
- Automatically deploy

Travis CI About Us Blog Status Help Daniel Sneijers

Search all repositories

My Repositories +

✓ danielsneijers/markslide # 210
Duration: 4 min 47 sec
Finished: about an hour ago

✓ blendle/emailcorrect # 75
Duration: 1 min 7 sec
Finished: about a month ago

✓ blendle/bunpack # 17
Duration: 1 min 1 sec
Finished: 2 years ago

danielsneijers / markslide

build passing

Current Branches Build History Pull Requests More options

✓ master CRON Merge pull request #6 from danielsneijers/move-to-redux

Move to redux

Commit c5f12a9
Branch master

Daniel Sneijers authored GitHub committed

#210 passed
Ran for 2 min 34 sec
Total time 4 min 47 sec
about an hour ago

Restart build

Build Jobs

✓	# 210.1	Node.js: 6	no environment variables set	2 min 13 sec
✓	# 210.2	Node.js: 7	no environment variables set	2 min 34 sec

```
Time:        62.497s
Ran all test suites.
-------------------------|----------|----------|----------|----------|----------------|
File                     |  % Stmts | % Branch |  % Funcs |  % Lines |Uncovered Lines |
-------------------------|----------|----------|----------|----------|----------------|
All files                |      100 |      100 |      100 |      100 |                |
 constants               |      100 |      100 |      100 |      100 |                |
  index.js               |      100 |      100 |      100 |      100 |                |
 higher-order-components |      100 |      100 |      100 |      100 |                |
  autoScaleContent.js    |      100 |      100 |      100 |      100 |                |
  fullScreenOption.js    |      100 |      100 |      100 |      100 |                |
  index.js               |      100 |      100 |      100 |      100 |                |
  keyboardNavigation.js  |      100 |      100 |      100 |      100 |                |
 modules/main            |      100 |      100 |      100 |      100 |                |
  component.js           |      100 |      100 |      100 |      100 |                |
  container.js           |      100 |      100 |      100 |      100 |                |
  selectors.js           |      100 |      100 |      100 |      100 |                |
 modules/progressbar     |      100 |      100 |      100 |      100 |                |
  component.js           |      100 |      100 |      100 |      100 |                |
  container.js           |      100 |      100 |      100 |      100 |                |
  index.js               |      100 |      100 |      100 |      100 |                |
 modules/slide           |      100 |      100 |      100 |      100 |                |
  reducer.js             |      100 |      100 |      100 |      100 |                |
  selectors.js           |      100 |      100 |      100 |      100 |                |
 modules/slide/code      |      100 |      100 |      100 |      100 |                |
  component.js           |      100 |      100 |      100 |      100 |                |
  container.js           |      100 |      100 |      100 |      100 |                |
  index.js               |      100 |      100 |      100 |      100 |                |
 modules/slide/default   |      100 |      100 |      100 |      100 |                |
  component.js           |      100 |      100 |      100 |      100 |                |
  container.js           |      100 |      100 |      100 |      100 |                |
  index.js               |      100 |      100 |      100 |      100 |                |
 utils                   |      100 |      100 |      100 |      100 |                |
  hoc.js                 |      100 |      100 |      100 |      100 |                |
  markdown.js            |      100 |      100 |      100 |      100 |                |
  metaData.js            |      100 |      100 |      100 |      100 |                |
  renderer.js            |      100 |      100 |      100 |      100 |                |
  slide.js               |      100 |      100 |      100 |      100 |                |
  text.js                |      100 |      100 |      100 |      100 |                |
  viewport.js            |      100 |      100 |      100 |      100 |                |
-------------------------|----------|----------|----------|----------|----------------|
No errors!
```

Top

WHAT YOU NEED

- No private keys in your codebase!
- environment variables
- A project on github
- A .travis.yml

.TRAVIS.YML

```yaml
language: node_js
node_js:
  - “6"
  - “8"
cache: yarn
after_success:
  - yarn run build:production
  - script/deploy
```

HOW TO CHOOSE CI/CD

- Keep it simple, start with Travis

- Once you feel comfortable, start playing with different ones (jenkins, circle, wercker, etc)

WHERE TO GO NEXT?

- Explore different frameworks
- Dive into micro frontends
- Build a component library with react-styleguidist or storybook
- Deep dive into RxJS
- Prepare a meetup talk
- Or write medium posts and share your insights!