Mage TITANS

MCR 2017

SHANE OSBOURNE

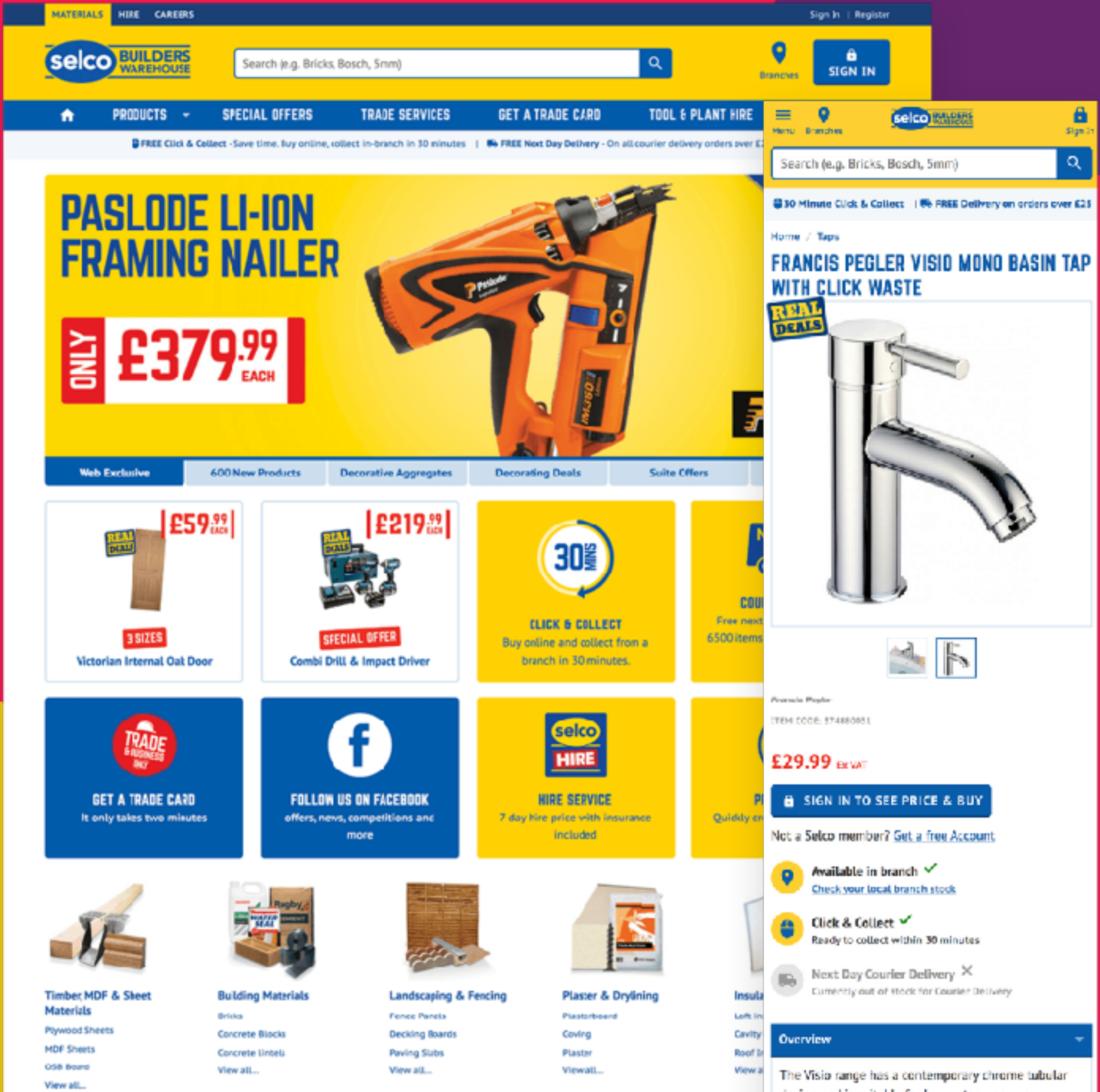
Lead Front-End Developer at JH

@shaneOsbourne

Using React-Native to build an M2 powered App on both iOS and Android

UK.MAGETITANS.COM

#MageTitansMCR @MageTitans



FREE Click & Collect - Save time. Buy online, collect in-branch in 30 minutes | FREE Next Day Delivery - On all courier delivery orders over £

PASLODE LI-ION FRAMING NAILER

ONLY £379.99 EACH

REAL DEALS £59.99 EACH
3 SIZES
Victorian Internal Oak Door

REAL DEALS £219.99 EACH
SPECIAL OFFER
Combi Drill & Impact Driver

30 MINS
CLICK & COLLECT
Buy online and collect from a branch in 30 minutes.

COU
Free next
6500 items

TRADE & BUSINESS ONLY
GET A TRADE CARD
It only takes two minutes

FOLLOW US ON FACEBOOK
offers, news, competitions and more

selco HIRE
HIRE SERVICE
7 day hire price with insurance included

P
Quickly cr

Timber, MDF & Sheet Materials
Plywood Sheets
MDF Sheets
OSB Board
View all...

Building Materials
Bricks
Concrete Blocks
Concrete lintels
View all...

Landscaping & Fencing
Fence Panels
Decking Boards
Paving Slabs
View all...

Plaster & Drylining
Plasterboard
Coving
Plaster
Viewall...

Insula
Loft In
Cavity
Roof In
View a

30 Minute Click & Collect | FREE Delivery on orders over £25

Home / Taps

FRANCIS PEGLER VISIO MONO BASIN TAP WITH CLICK WASTE

REAL DEALS

Francis Pegler

ITEM CODE: 374880051

£29.99 Ex VAT

SIGN IN TO SEE PRICE & BUY

Not a Selco member? Get a free Account

Available in branch ✓
Check your local branch stock

Click & Collect ✓
Ready to collect within 30 minutes

Next Day Courier Delivery ✕
Currently out of stock for Courier Delivery

Overview

The Visio range has a contemporary chrome tubular

php

JS

Required Skills

Design / UX ☑

API Development ☑

APP Development

Objective C / **Swift**

Java / **Kotlin**

React Native

iOS and Android Apps from JavaScript

No need to hire anyone new

No need to learn new languages

Who’s using it?

Why React Native

It's actively developed by Facebook

It uses modern tooling

We get to write JavaScript

90% of code shared across Platforms

It's **not** just a web view wrapper

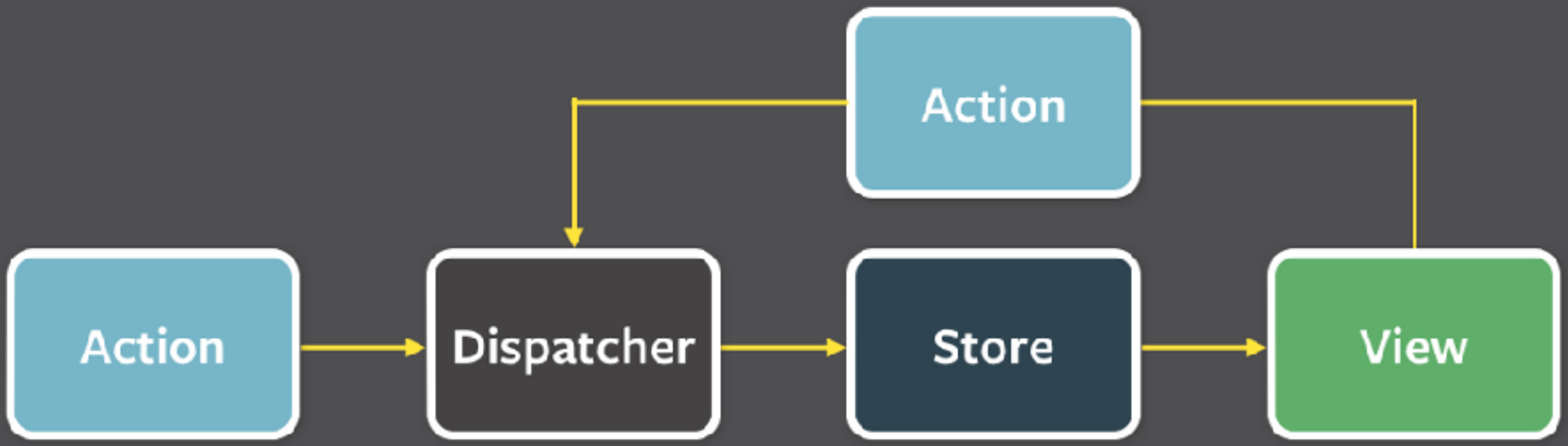
React Native
(not for beginners)

“Don’t ever get married to a particular library or framework, or even programming style - instead become an expert at the fundamentals”

- some wise developer

Redux

UNI-DIRECTIONAL DATA FLOW



Action

Action

Dispatcher

Store

View

Redux

UNI-DIRECTIONAL DATA FLOW

```
<Button onClick={() =>
  dispatch({
    type: 'SELECT_ITEM',
    payload: item.id
  })}
>
  {item.name}
</Button>
```

Redux

UNI-DIRECTIONAL DATA FLOW

```
function reducer(state = [], action) {
    switch (action.type) {
        case 'SELECT_ITEM': {
            return state
              .concat(action.payload);
        }
        default: return state;
    }
}
```

Redux

UNI-DIRECTIONAL DATA FLOW

```
function reducer(state = [], action) {
    switch (action.type) {
        case 'SELECT_ITEM': {
            return state
              .concat(action.payload);
        }
        default: return state;
    }
}
```

Redux

UNI-DIRECTIONAL DATA FLOW

```
function reducer(state = [], action) {
    switch (action.type) {
        case 'SELECT_ITEM': {
            return state
              .concat(action.payload);
        }
        default: return state;
    }
}
```

Redux

UNI-DIRECTIONAL DATA FLOW

```
function reducer(state = [], action) {
    switch (action.type) {
        case 'SELECT_ITEM': {
            return state
              .concat(action.payload);
        }
        default: return state;
    }
}
```

RxJS

COMPOSITION OF SIDE EFFECTS

```
dispatch({
  type: 'LOGIN',
  payload: {
    username: 'shakyshane',
    password: 'h4cker4life',
  }
})
```

RxJS

COMPOSITION OF SIDE EFFECTS

```
function login(action$) {
  return action$.ofType('LOGIN')
    .switchMap((creds) =>
      ajax('http://example.com/signin', creds)
        .takeUntil(action$.ofType('NAVIGATE'))
        .catch(err => of({type: 'LOGIN_FAIL'}))
        .mapTo({type: 'LOGIN_SUCCESS'})
    )
}
```

RxJS

COMPOSITION OF SIDE EFFECTS

```
function login(action$) {
  return action$.ofType('LOGIN')
    .switchMap((creds) =>
      ajax('http://example.com/signin', creds)
        .takeUntil(action$.ofType('NAVIGATE'))
        .catch(err => of({type: 'LOGIN_FAIL'}))
        .mapTo({type: 'LOGIN_SUCCESS'})
    )
}
```

RxJS

COMPOSITION OF SIDE EFFECTS

```
function login(action$) {
  return action$.ofType('LOGIN')
    .switchMap((creds) =>
      ajax('http://example.com/signin', creds)
        .takeUntil(action$.ofType('NAVIGATE'))
        .catch(err => of({type: 'LOGIN_FAIL'}))
        .mapTo({type: 'LOGIN_SUCCESS'})
    )
}
```

RxJS

COMPOSITION OF SIDE EFFECTS

```
function login(action$) {
  return action$.ofType('LOGIN')
    .switchMap((creds) =>
      ajax('http://example.com/signin', creds)
        .takeUntil(action$.ofType('NAVIGATE'))
        .catch(err => of({type: 'LOGIN_FAIL'}))
        .mapTo({type: 'LOGIN_SUCCESS'})
    )
}
```

RxJS

COMPOSITION OF SIDE EFFECTS

```
function login(action$) {
  return action$.ofType('LOGIN')
    .switchMap((creds) =>
      ajax('http://example.com/signin', creds)
        .takeUntil(action$.ofType('NAVIGATE'))
        .catch(err => of({type: 'LOGIN_FAIL'}))
        .mapTo({type: 'LOGIN_SUCCESS'})
    )
}
```

RxJS

COMPOSITION OF SIDE EFFECTS

```
function login(action$) {
  return action$.ofType('LOGIN')
    .switchMap((creds) =>
      ajax('http://example.com/signin', creds)
        .takeUntil(action$.ofType('NAVIGATE'))
        .catch(err => of({type: 'LOGIN_FAIL'}))
        .mapTo({type: 'LOGIN_SUCCESS'})
    )
}
```

Jest

TESTING APPLICATION LOGIC

```
const ajaxMock = jest.fn()
  .mockReturnsValue(of({status: 'success'}));

const store = configureStore({ajax: ajaxMock});

store.dispatch({
  type: 'LOGIN',
  payload: {
    username: 'shakyshane',
    password: 'kittenz'
  },
});

assert.equal(store.getState().auth.status, 'success');
```

Jest

TESTING APPLICATION LOGIC

```
const ajaxMock = jest.fn()
  .mockReturnsValue(of({status: 'success'}));

const store = configureStore({ajax: ajaxMock});

store.dispatch({
  type: 'LOGIN',
  payload: {
    username: 'shakyshane',
    password: 'kittenz'
  },
});

assert.equal(store.getState().auth.status, 'success');
```

Jest

TESTING APPLICATION LOGIC

```
const ajaxMock = jest.fn()
  .mockReturnsValue(of({status: 'success'}));

const store = configureStore({ajax: ajaxMock});

store.dispatch({
  type: 'LOGIN',
  payload: {
    username: 'shakyshane',
    password: 'kittenz'
  },
});

assert.equal(store.getState().auth.status, 'success');
```

Jest

TESTING APPLICATION LOGIC

```
const ajaxMock = jest.fn()
  .mockReturnsValue(of({status: 'success'}));

const store = configureStore({ajax: ajaxMock});

store.dispatch({
  type: 'LOGIN',
  payload: {
    username: 'shakyshane',
    password: 'kittenz'
  },
});

assert.equal(store.getState().auth.status, 'success');
```

Jest

TESTING APPLICATION LOGIC

```
const ajaxMock = jest.fn()
  .mockReturnsValue(of({status: 'success'}));

const store = configureStore({ajax: ajaxMock});

store.dispatch({
  type: 'LOGIN',
  payload: {
    username: 'shakyshane',
    password: 'kittenz'
  },
});

assert.equal(store.getState().auth.status, 'success');
```

ESLint

ENFORCE CODE QUALITY/STYLE

```
const name = "shakyShane"
```

```
$ eslint src/** --fix
```

```
const name = 'shakyShane';
```

Flow

STATIC TYPE CHECKER

```
// @flow
type User = {
    name: string,
    age: number
}


function getUser(name: string): Observable<User> {
    return ajax(`http://example.com/user/${name}`)
}
```

Flow

STATIC TYPE CHECKER

```
// @flow
type User = {
    name: string,
    age: number
}


function getUser(name: string): Observable<User> {
    return ajax(`http://example.com/user/${name}`)
}
```

Flow

STATIC TYPE CHECKER

```
// @flow
getUser('shakyshane')
    .subscribe(user => {
        console.log(user.name);
        console.log(user.age);
    });
```

What we learnt

Don’t become an expert in a single library/ framework.

Choose tools that make debugging easier

Use automation to lint & auto-fix your code

Write tests at the highest level possible

Get a grip on side-effects

@shaneosbourne

@shakyshane

egghead.io/instructors/shane-osbourne

wearejh.com