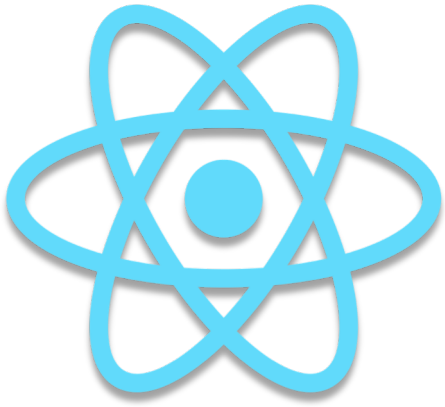
React Native in Production



My journey back to JavaScript

Oussama Ghalbzouri

- 32 years old
- Born & raised in Paris
- In Luxembourg since April 2015
- Technical Lead – Mobile Apps
- atHomegroup

- JavaScript is my first language
- Web development in 2008 (jQuery/PHP)
- Apps development in 2010 (iOS/Android)
- Tried JavaScript cross-platform apps development in 2011 (PhoneGap/Titanium)
- Continued with Objective-C and Java until 2017

atHome.lu
Luxembourg lives here

immoRegion.fr
Annonces immobilières régionales

atHome.de
Das Immobilienportal Ihrer Region

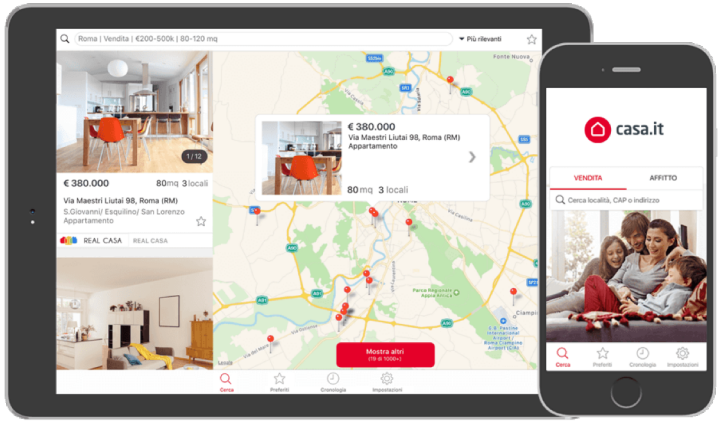
atHome Finance

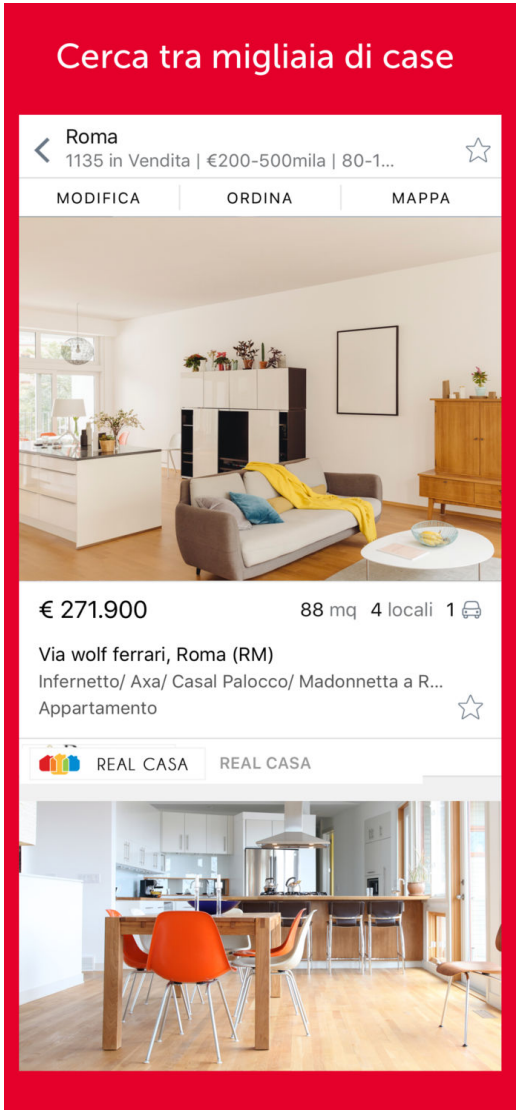
casa.it®

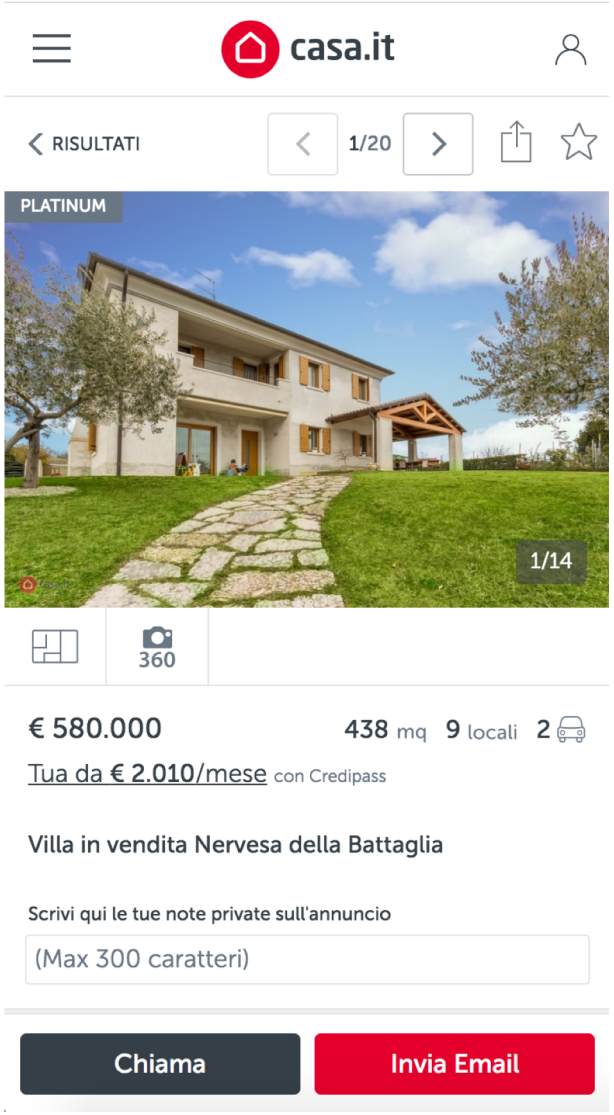
LUXAUTO.LU

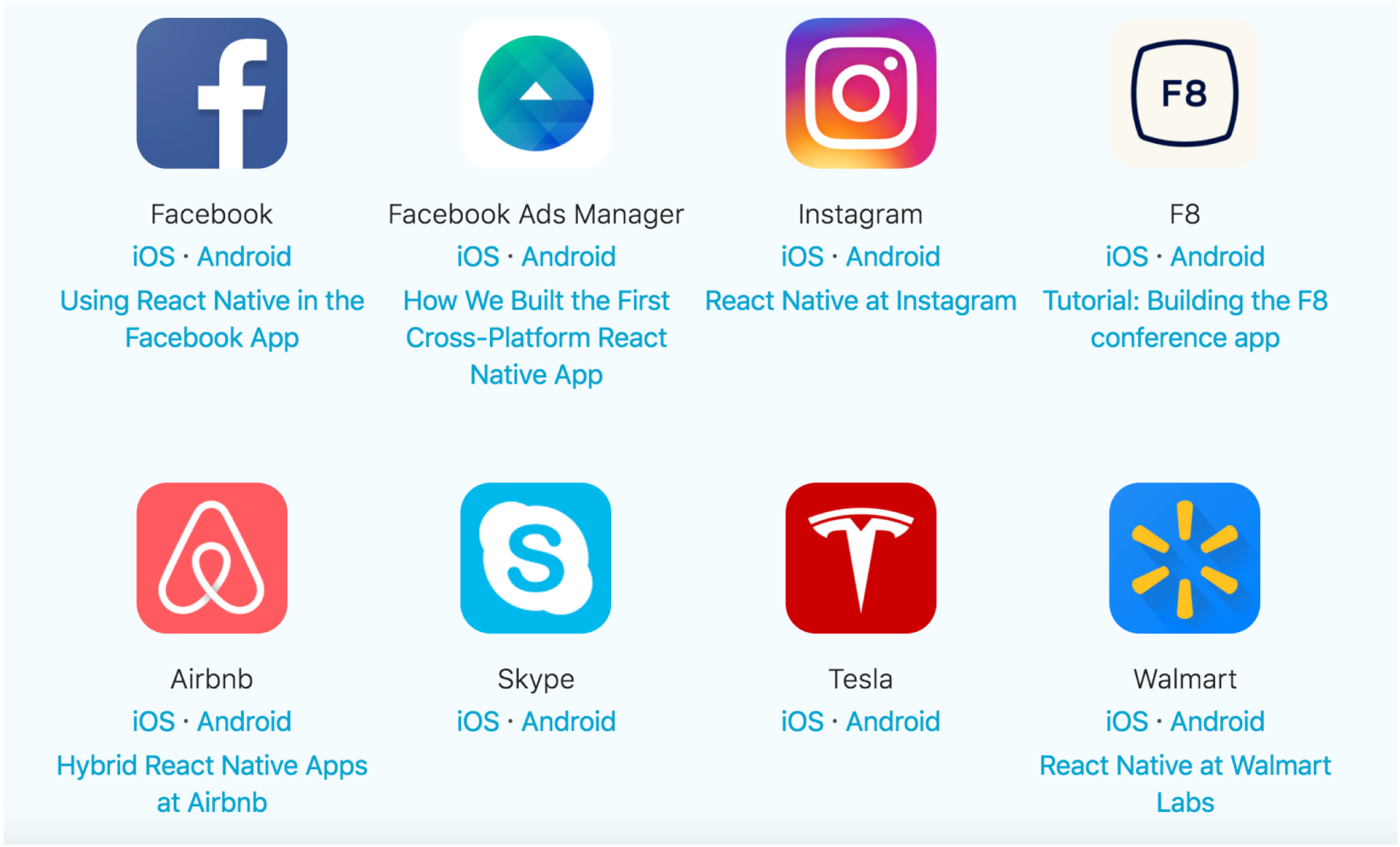
casa.it

- Successfully using React and React Native since 2016









Bloomberg, Soundcloud, Artsy, Adidas, Baidu, Uber, Wix…

http://facebook.github.io/react-native/showcase.html

ES6 JS Promise

#Javascript

flow-typed TypeScript

node webpack

JS

What is React Native?

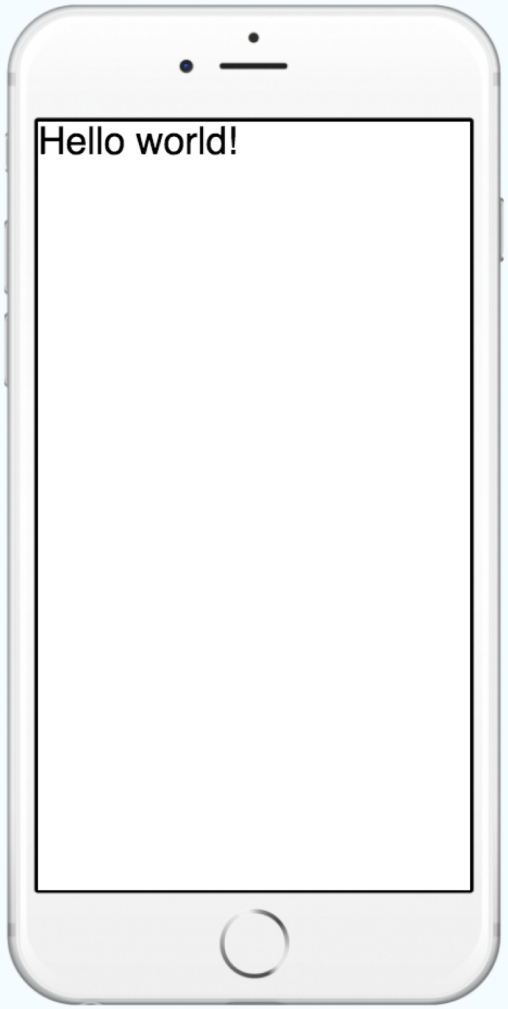
Presentation

- Mobile apps framework based on React
- Created and promoted by Facebook in March 2015
- Abstraction layer for native UI components
- Interprets JavaScript to Objective-C/Java via C/C++
- Live Reloading / Hot reloading
- Version 0.54

```
1  import React, { Component } from 'react';
2  import { Text } from 'react-native';
3
4  export default class HelloWorldApp extends Component {
5    render() {
6      return (
7        <Text>Hello world!</Text>
8      );
9    }
10 }
11
```

No Errors

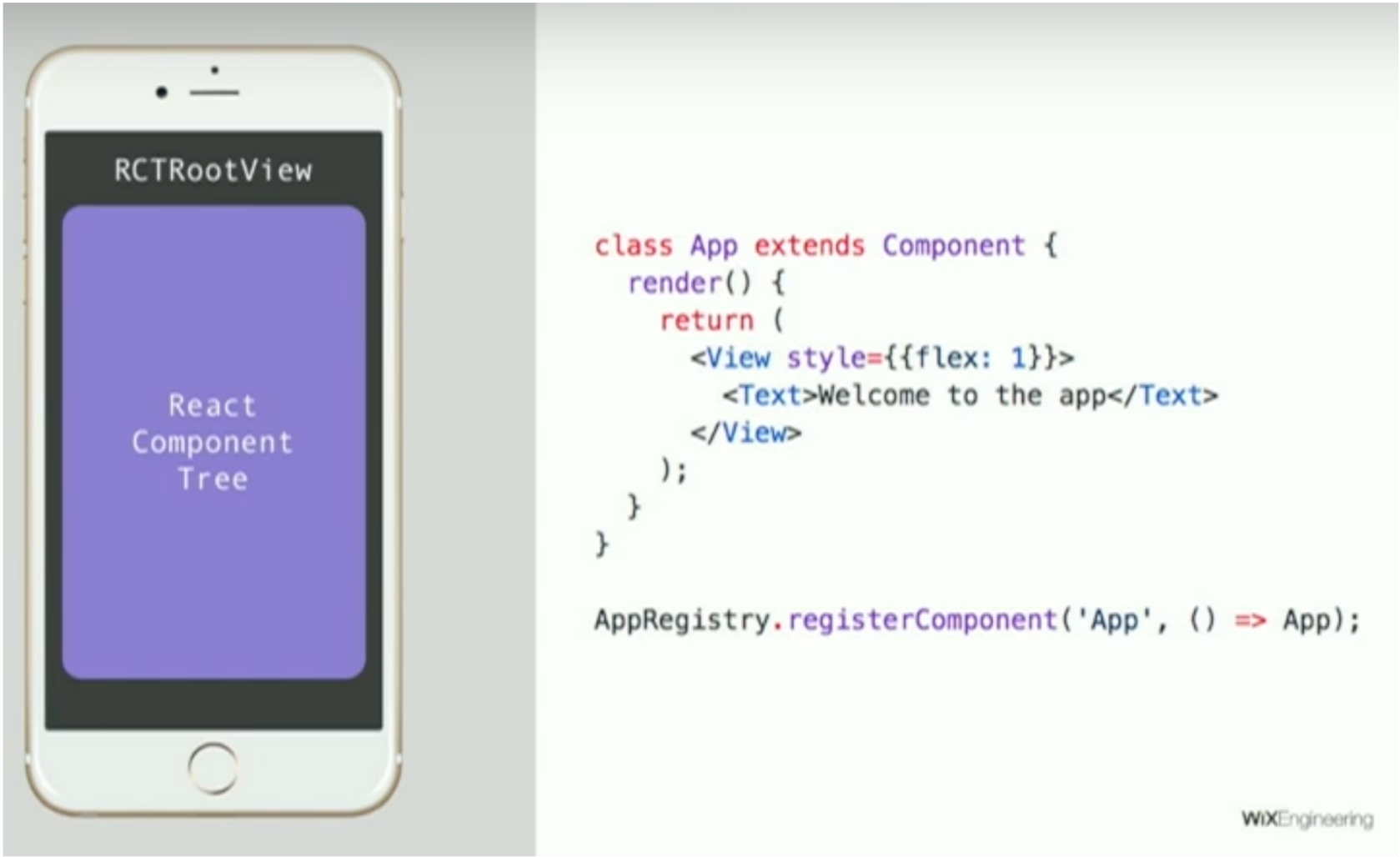
Show Details



Source: https://facebook.github.io/react-native/docs/tutorial.html

How does it work ?

Let’s dive under the hood…



Source: Tal Kol - Going Over The Speed Limit: Synchronous Rendering in React Native https://www.youtube.com/watch?v=HXKFQu2cP4c

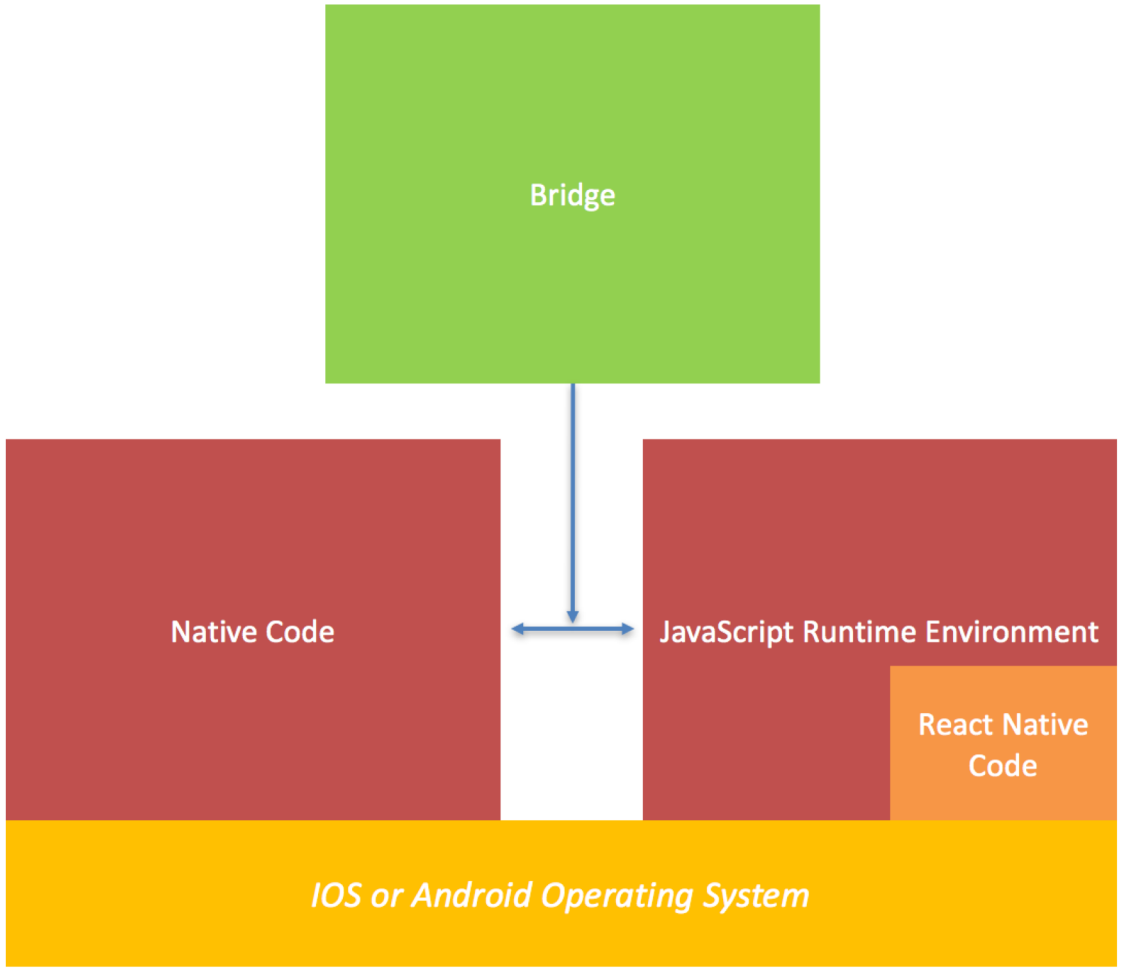
- iOS

```
RCTRootView *rootView = [[RCTRootView alloc] initWithBundleURL:jsCodeLocation
                                                    moduleName:@"App"
                                             initialProperties:nil
                                                 launchOptions:launchOptions];
```

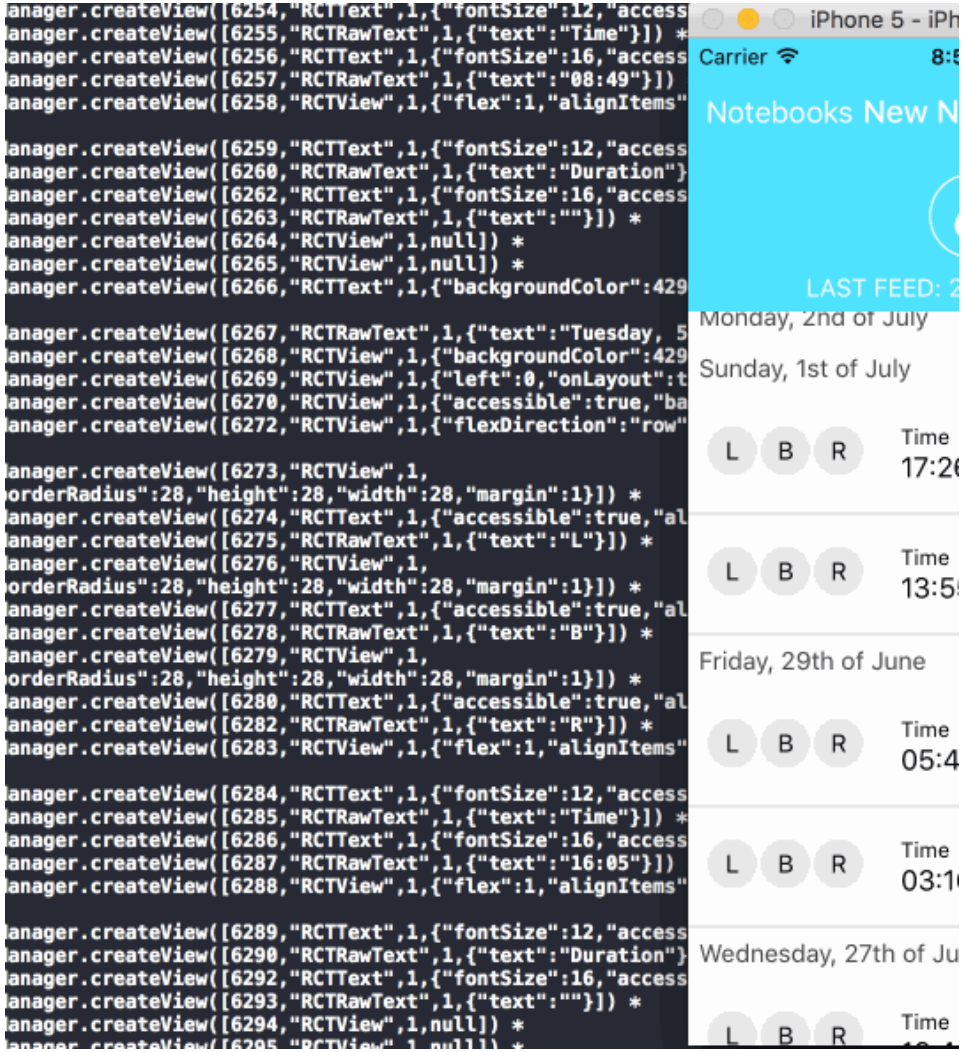
- Android

```
public class MainActivity extends ReactActivity {

  /**
   * Returns the name of the main component registered from JavaScript.
   * This is used to schedule rendering of the component.
   */
  @Override
  protected String getMainComponentName() {
    return "App";
  }
```

Architecture



Source: https://www.logicroom.co/react-native-architecture-explained/



Source: RN Snoopy https://github.com/jondot/rn-snoopy

Layout engine

- UIManager: converts declarative component tree into imperative instructions
- Yoga: cross-platform layout engine which implements Flexbox

Litho

ComponentKit

- JSX

```
public render() {
  return (
    <View>
      <Button title="Press me" onPress={() => console.log("Pressed")} />
    </View>
  );
}
```

- UIManager

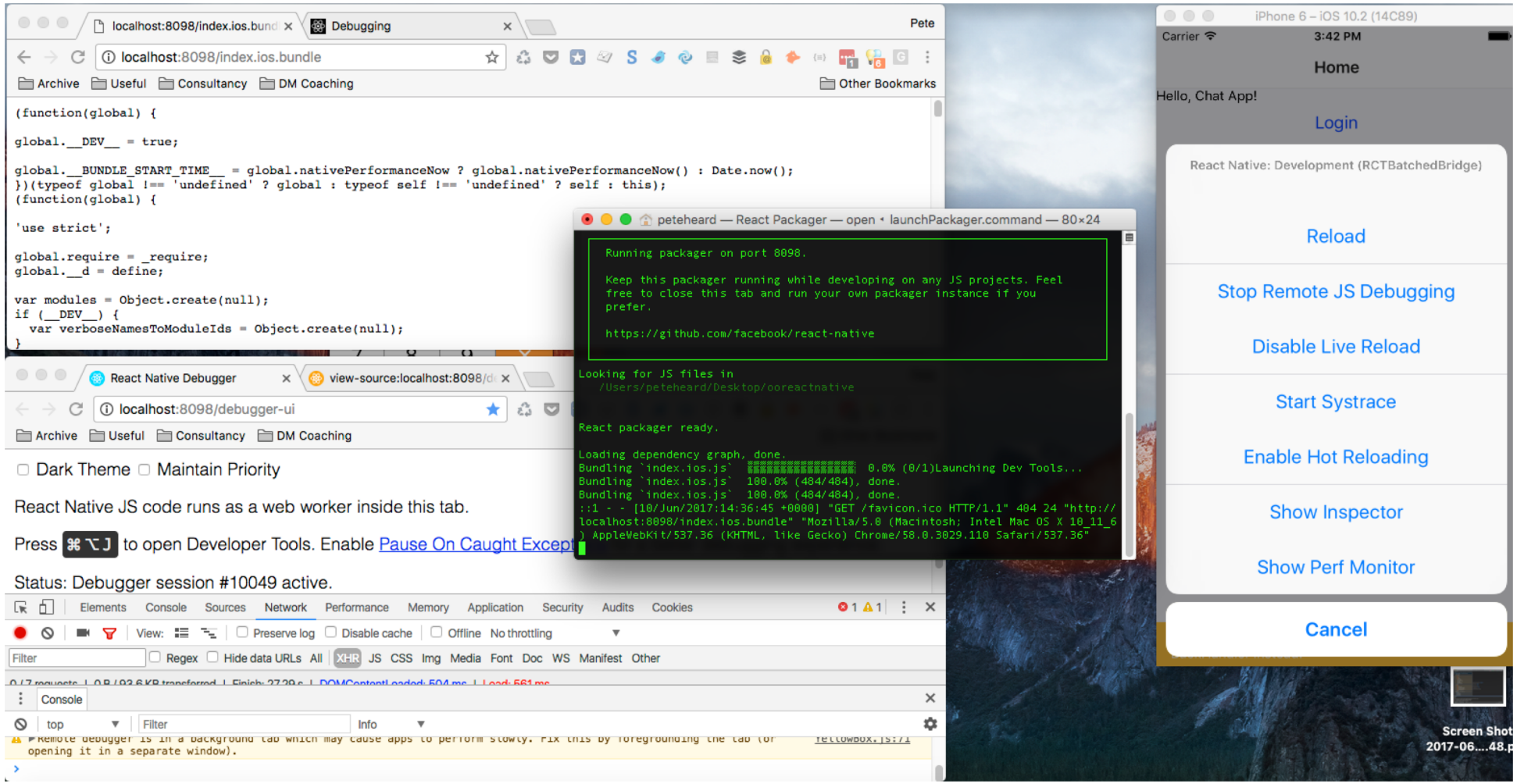
```
UIManager.createView([708, "RCTRawText", 41, {"text":"PRESS ME"}])
UIManager.createView([708, "RCTText", 41, {"textAlign" : "center"}])
UIManager.setChildren([708, [709]]);
```

- Yoga

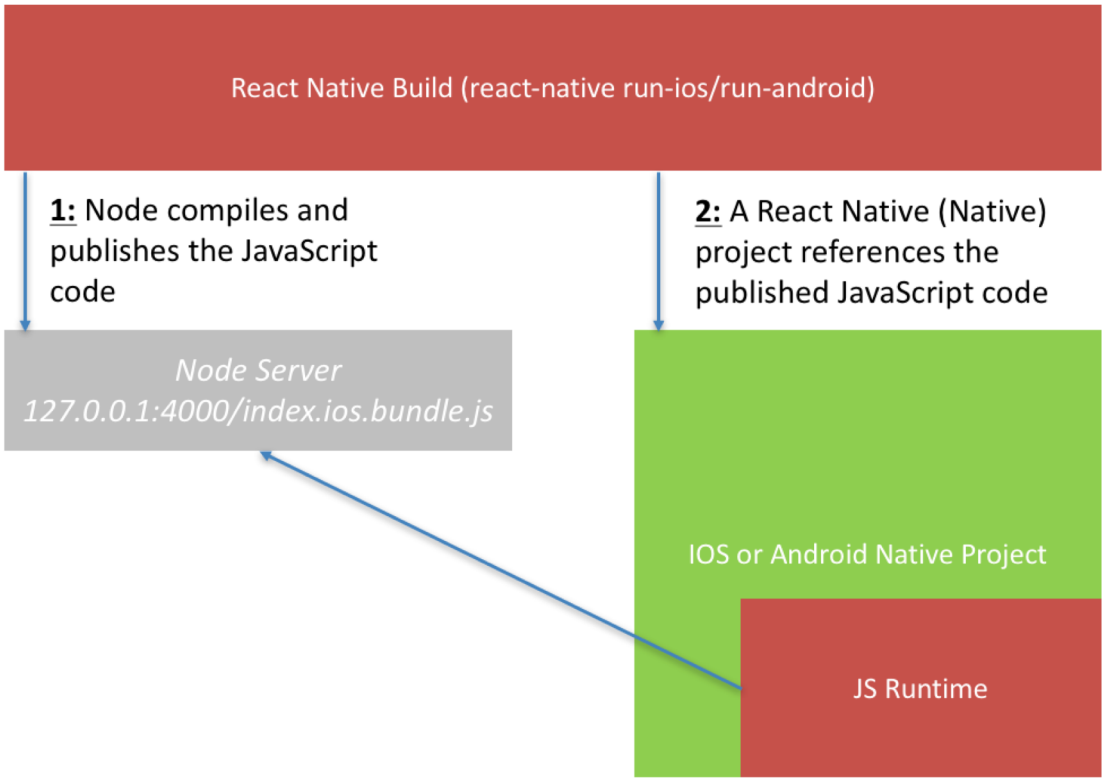
```
YGNodeRef container = YGNodeNew();
YGNodeStyleSetPadding(container, YGEdgeAll, 20);

YGNodeRef button = YGNodeNew();
YGNodeInsertChild(container, button, 0);

YGNodeRef text = YGNodeNew();
YGNodeSetMeasureFunc(text, &measureText);
YGNodeInsertChild(button, text, 0);

YGNodeCalculateLayout(
   container,
   screenWidth,
   screenHeight,
   layoutDirection);
```

Source: Emil Sjölander - React Native, The Native Bits https://www.youtube.com/watch?v=jFiQ6FxBDqY

Development Mode


localhost:8098/index.ios.bund
Debugging
Pete
localhost:8098/index.ios.bundle
Archive
Useful
Consultancy
DM Coaching
Other Bookmarks
(function(global) {
global.__DEV__ = true;
global.__BUNDLE_START_TIME__ = global.nativePerformanceNow ? global.nativePerformanceNow() : Date.now();
})(typeof global !== 'undefined' ? global : typeof self !== 'undefined' ? self : this);
(function(global) {
'use strict';
global.require = _require;
global.__d = define;
var modules = Object.create(null);
if (__DEV__) {
var verboseNamesToModuleIds = Object.create(null);
}
peteheard — React Packager — open • launchPackager.command — 80×24
Running packager on port 8098.
Keep this packager running while developing on any JS projects. Feel free to close this tab and run your own packager instance if you prefer.
https://github.com/facebook/react-native
Looking for JS files in
/Users/peteheard/Desktop/ooreactnative
React packager ready.
Loading dependency graph, done.
Bundling `index.ios.js` 0.0% (0/1)Launching Dev Tools...
Bundling `index.ios.js` 100.0% (484/484), done.
Bundling `index.ios.js` 100.0% (484/484), done.
::1 - - [10/Jun/2017:14:36:45 +0000] "GET /favicon.ico HTTP/1.1" 404 24 "http://localhost:8098/index.ios.bundle" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.36"
React Native Debugger
view-source:localhost:8098/d
localhost:8098/debugger-ui
Archive
Useful
Consultancy
DM Coaching
Dark Theme
Maintain Priority
React Native JS code runs as a web worker inside this tab.
Press ⌘⌥J to open Developer Tools. Enable Pause On Caught Except
Status: Debugger session #10049 active.
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Audits
Cookies
Preserve log
Disable cache
Offline
No throttling
Filter
Regex
Hide data URLs
All
XHR
JS
CSS
Img
Media
Font
Doc
WS
Manifest
Other
Console
top
Filter
Info
Remote debugger is in a background tab which may cause apps to perform slowly. Fix this by foregrounding the tab (or opening it in a separate window).
YellowBox.js:71
iPhone 6 – iOS 10.2 (14C89)
Carrier
3:42 PM
Home
Hello, Chat App!
Login
React Native: Development (RCTBatchedBridge)
Reload
Stop Remote JS Debugging
Disable Live Reload
Start Systrace
Enable Hot Reloading
Show Inspector
Show Perf Monitor
Cancel
Screen Shot
2017-06....48.p

- Architecture in Development mode



Source: https://www.logicroom.co/react-native-architecture-explained/

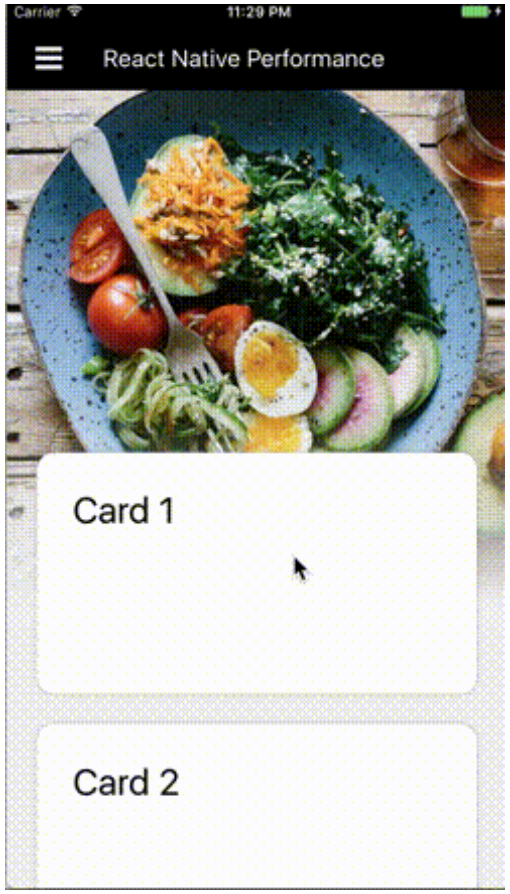
- Architecture in Development mode

```
#ifdef DEBUG
  jsCodeLocation = [[RCTBundleURLProvider sharedSettings] jsBundleURLForBundleRoot:@"index" fallbackResource:nil];
#else
  jsCodeLocation = [[NSBundle mainBundle] URLForResource:@"main" withExtension:@"jsbundle"];
#endif
```

Animations

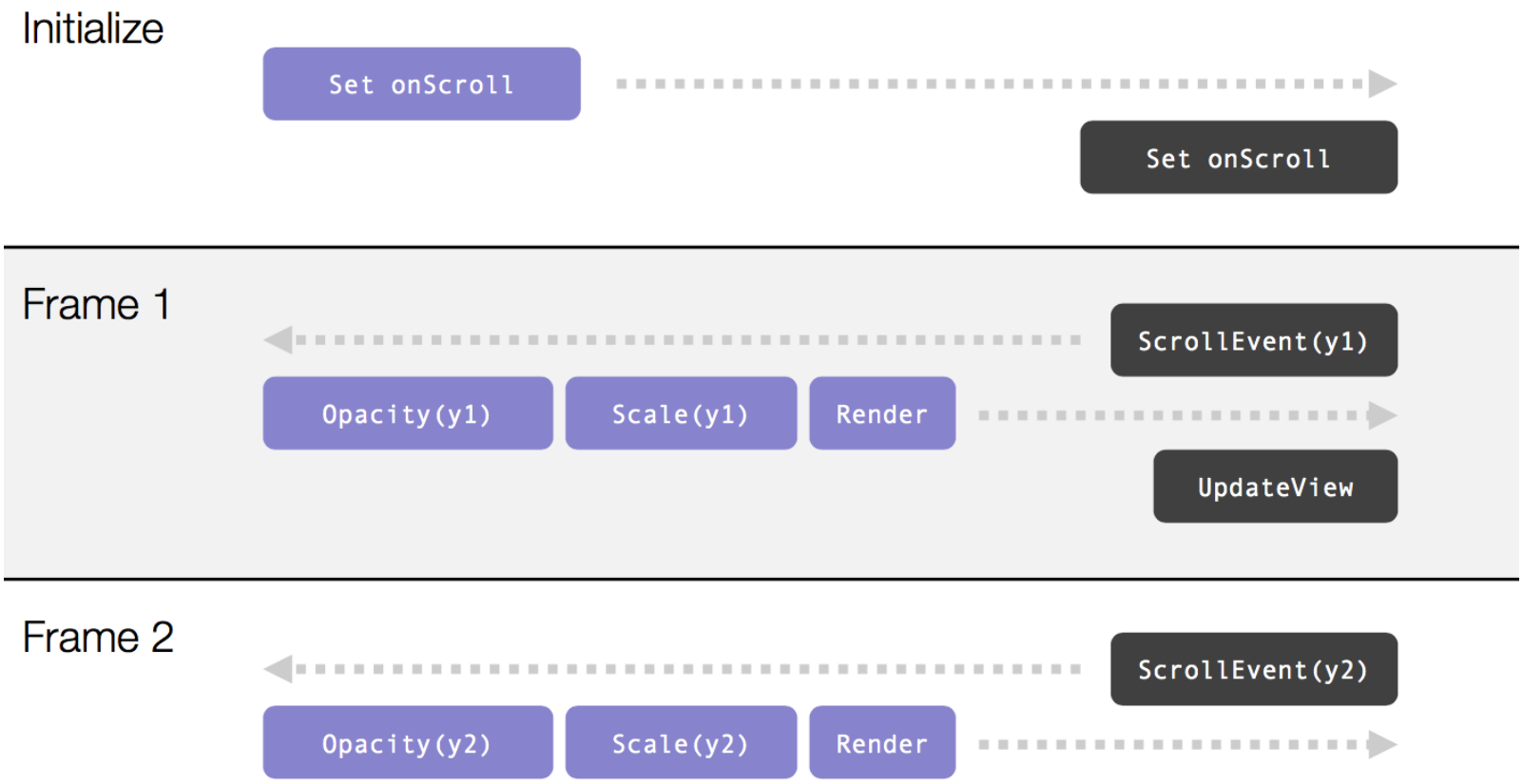
Animations

- Case study: Collapsible header
- 1st approach: for a given scroll event update position and opacity



```
<ScrollView
  {...props}
  scrollEventThrottle={16}
  onScroll={this.onScroll.bind(this)}
/>
```

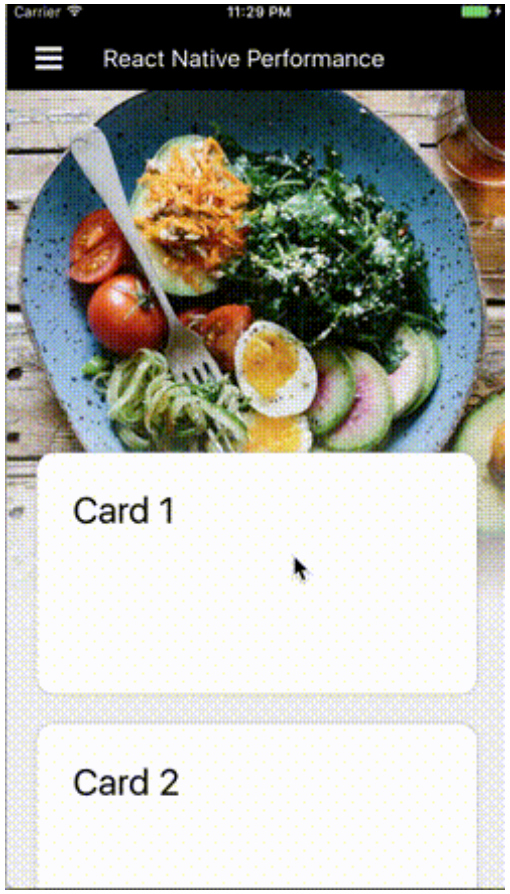
```
onScroll(event) {
  const scrollY = event.nativeEvent.contentOffset.y;
  if (scrollY >= 0) {
    let newOpacity = 1.0 - (scrollY / 250.0);
    if (newOpacity < 0) newOpacity = 0;
    this.setState({
      imageOpacity: newOpacity,
      imageScale: 1.0
    });
  } else {
    let newScale = 1.0 + 0.4*(-scrollY / 200.0);
    if (newScale > 1.4) newScale = 1.4;
    this.setState({
      imageOpacity: 1.0,
      imageScale: newScale
    });
  }
}
```



Source: https://hackernoon.com/react-native-performance-an-updated-example-6516bfde9c5c

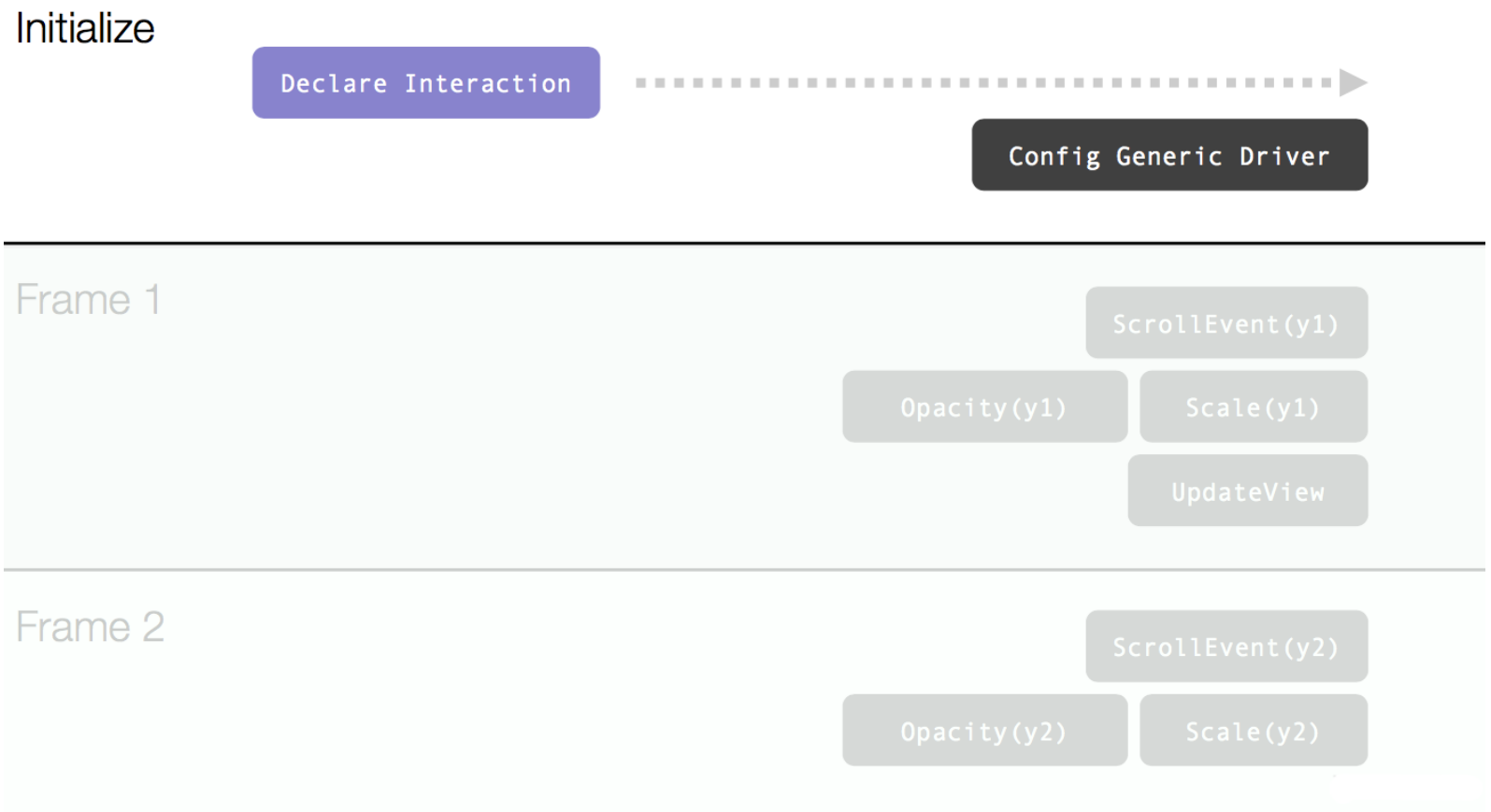
Animations

- Case study: Collapsible header
- 1st approach: for a given scroll event update position and opacity
- 2nd approach: use declarative API to avoid using the bridge in every frame



```
<Animated.ScrollView
  {...props}
  scrollEventThrottle={16}
  onScroll={
    Animated.event([{
      nativeEvent: { contentOffset: { y: this.state.scrollY } }
    }], {
      useNativeDriver: true
    })
  }
/>
```

```
<Animated.Image
  style={[styles.backgroundImage, {
    opacity: this.state.scrollY.interpolate({
      inputRange: [0, 250],
      outputRange: [1, 0]
    }),
    transform: [{
      scale: this.state.scrollY.interpolate({
        inputRange: [-200, 0, 1],
        outputRange: [1.4, 1, 1]
      })
    }]
  }]}
  source={require('../img/bg.jpg')}
/>
```



Source: https://hackernoon.com/react-native-performance-an-updated-example-6516bfde9c5c

Competition

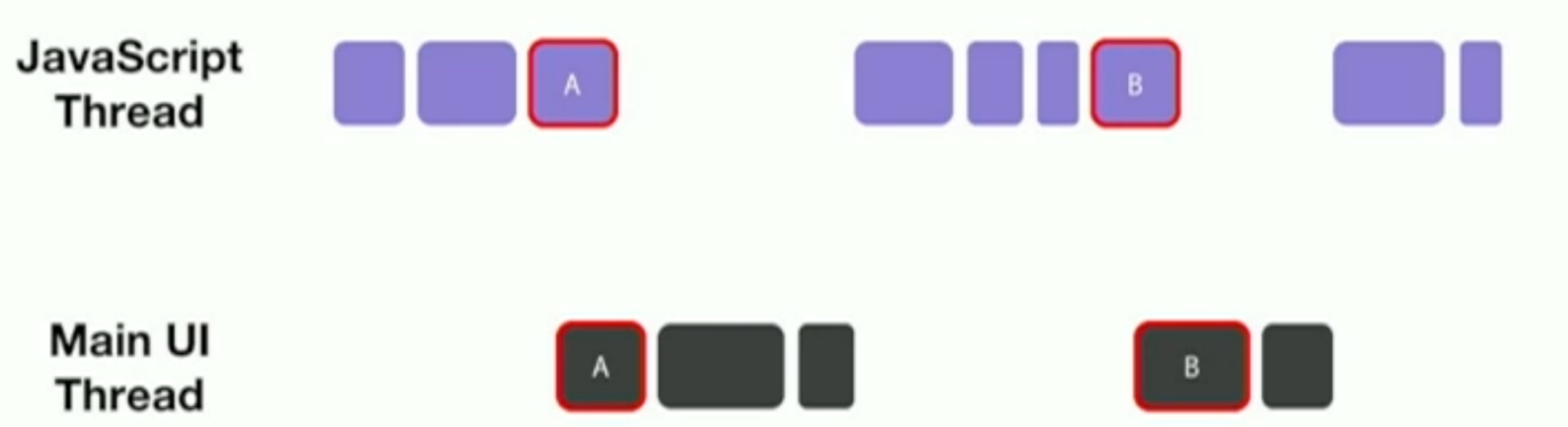
What about other frameworks ?

- Flutter:
  - embeds Dart runtime
  - renders its own UI components using a 2D engine in C++
  - can call native APIs using plugins in Java / Objective-C
- Cordova/Ionic:
  - renders using an embedded browser
  - can call native APIs via JavaScript using plugins in Java / Objective-C
- Titanium:
  - ~~translates~~ interprets JS to Objective-C and Java
  - can call native APIs via JavaScript (Titanium SDK + Hyperloop)

What about other frameworks ?

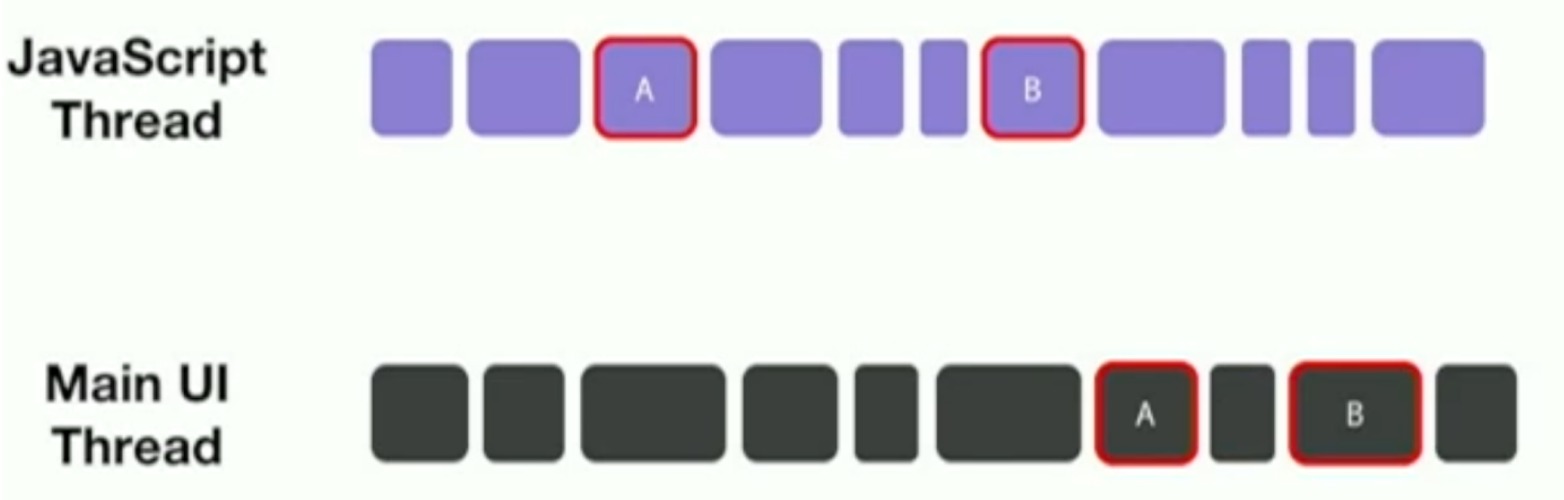
- Xamarin:
  - based on .NET
  - converts C# to intermediate language which calls native APIs
  - Xamarin Forms vs Xamarin Native
- NativeScript:
  - based on Angular
  - interprets JavaScript to Objective-C/Java via C/C++
  - can directly call native apis via JavaScript
  - different threading model

NativeScript threading: synchronous



Source: https://hackernoon.com/react-native-performance-an-updated-example-6516bfde9c5c

React Native threading: asynchronous

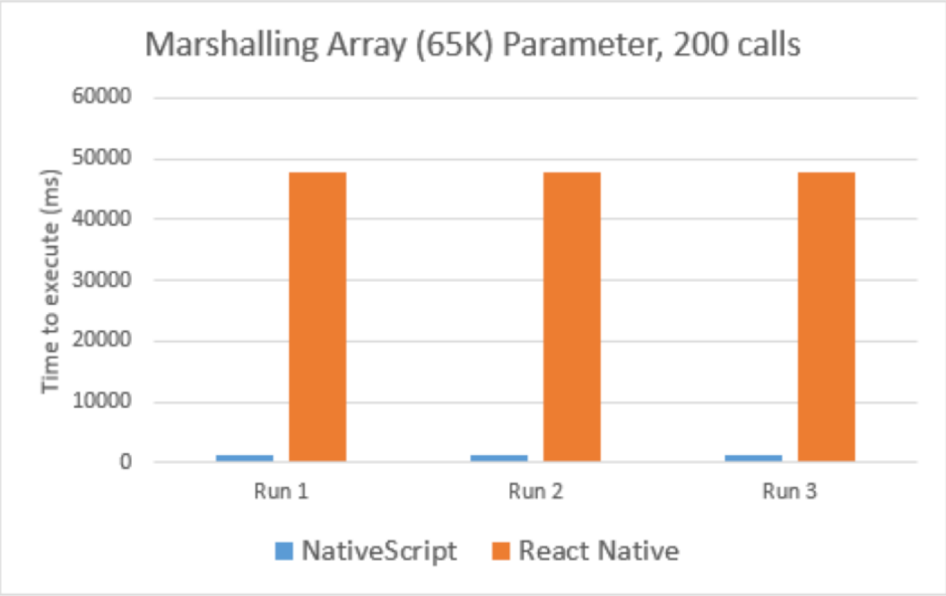


Source: https://hackernoon.com/react-native-performance-an-updated-example-6516bfde9c5c

MOBILE

The Benefits of NativeScript's Single Threading Model

Our synthetic runtime benchmarks demonstrate that. When compared with React Native, NativeScript, running on the main UI thread, is orders of magnitude faster calling native APIs, a very common behavior in JavaScript-driven native apps. *(Note: the below tests were performed on iOS8+)*



Source: https://developer.telerik.com/featured/benefits-single-threading-model-nativescript/

NativeScript threading: synchronous

NativeScript is Single Threaded: Did you know that all the UI and interactions are ran on the main UI thread? This means you are taxing your application for each container it has to render. The number of containers also affect the performance of the underlying native frameworks and their rendering and layout times.

Note: NativeScript does allow you to use background workers, but these are not meant for UI and are instead used for services such as http calls, large data manipulations, etc.

Source: https://www.nativescript.org/blog/nativescript-angular-performance-tips-tricks

Current Threading Model

For context, here's what we have now:

- **UI Thread**: standard main thread for the platforms
- **Shadow Thread**: Where measuring and layout occur. In a world where UI is constructed off the main thread, like in ComponentKit (an idea we're hoping to ship soon), this is where that occurs.
- **JS Thread**: Where React and Relay run.

Source: https://www.facebook.com/notes/andy-street/react-native-scheduling/10153916310914590/

Why we chose React Native

- New website using React
- Casa.it using React and React Native
- Strong community support
- Renders native UI components

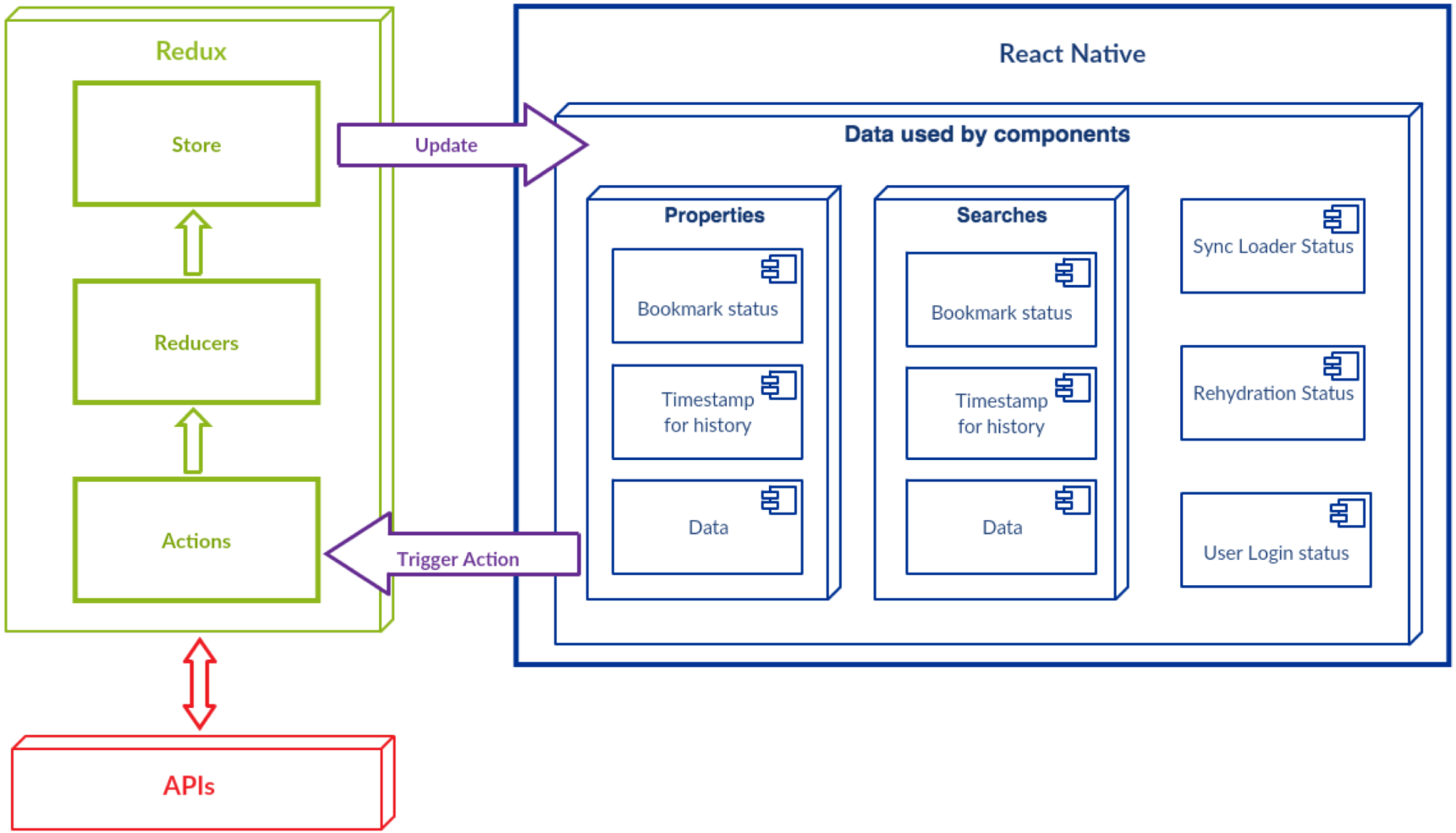
Starting choices

Choices

- TypeScript vs Flow vs …
- Visual Studio Code vs Atom vs …
- Multiple apps with the same codebase (multitenancy)
- Boilerplate ? Expo ? NativeBase ?
- Abstract UI with web: ReactXP ? react-native-web ? StyledComponents ?
- Inheritance (abstract) vs Composition (HOC)
- Single repository vs Multi repository
- Redux vs MobX vs …

```
public componentDidMount() {
  super.componentDidMount();

  // Code here
}
```

Redux


Redux
Store
Reducers
Actions
APIs
Update
Trigger Action
React Native
Data used by components
Properties
Bookmark status
Timestamp for history
Data
Searches
Bookmark status
Timestamp for history
Data
Sync Loader Status
Rehydration Status
User Login status

Choices

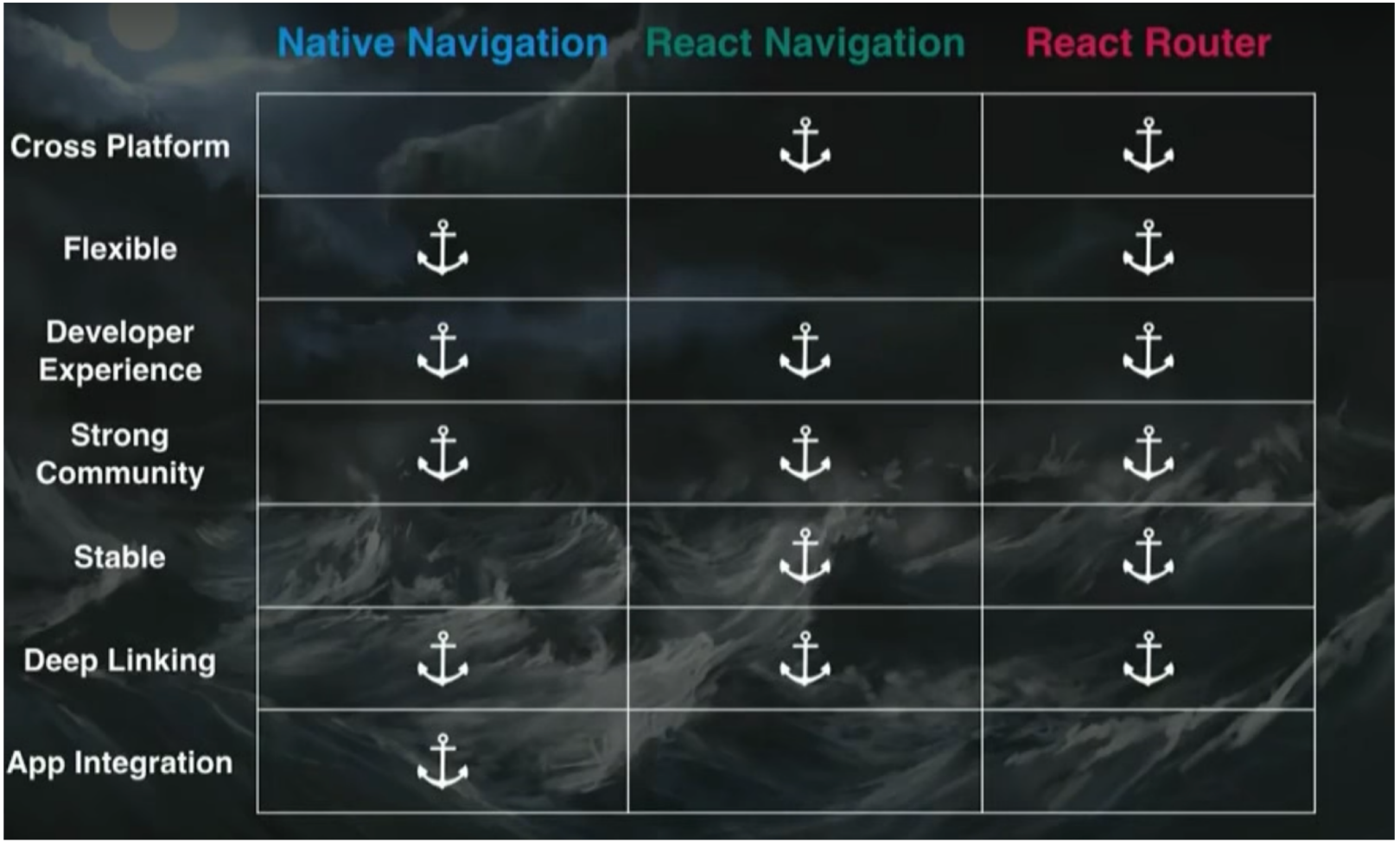
- TypeScript vs Flow vs …
- Visual Studio Code vs Atom vs …
- Inheritance (abstract) vs Composition (HOC)
- Multiple apps with the same codebase (multitenancy)
- Boilerplate ? Expo ? NativeBase ?
- Abstract UI with web: ReactXP ? react-native-web ? StyledComponents ?
- Single repository vs Multi repository
- Redux vs MobX vs …
- Brown field vs Green Field

Brown field vs Green field

- Integrate RN in existing app
- Native development skills
- RN used as the V from MVC
- Approach chosen by Facebook, Instagram, Airbnb…

- RN app from scratch
- Redux and shared logic in JS
- atHome requirements:
  - new API / data layer
  - new products / design

Navigating



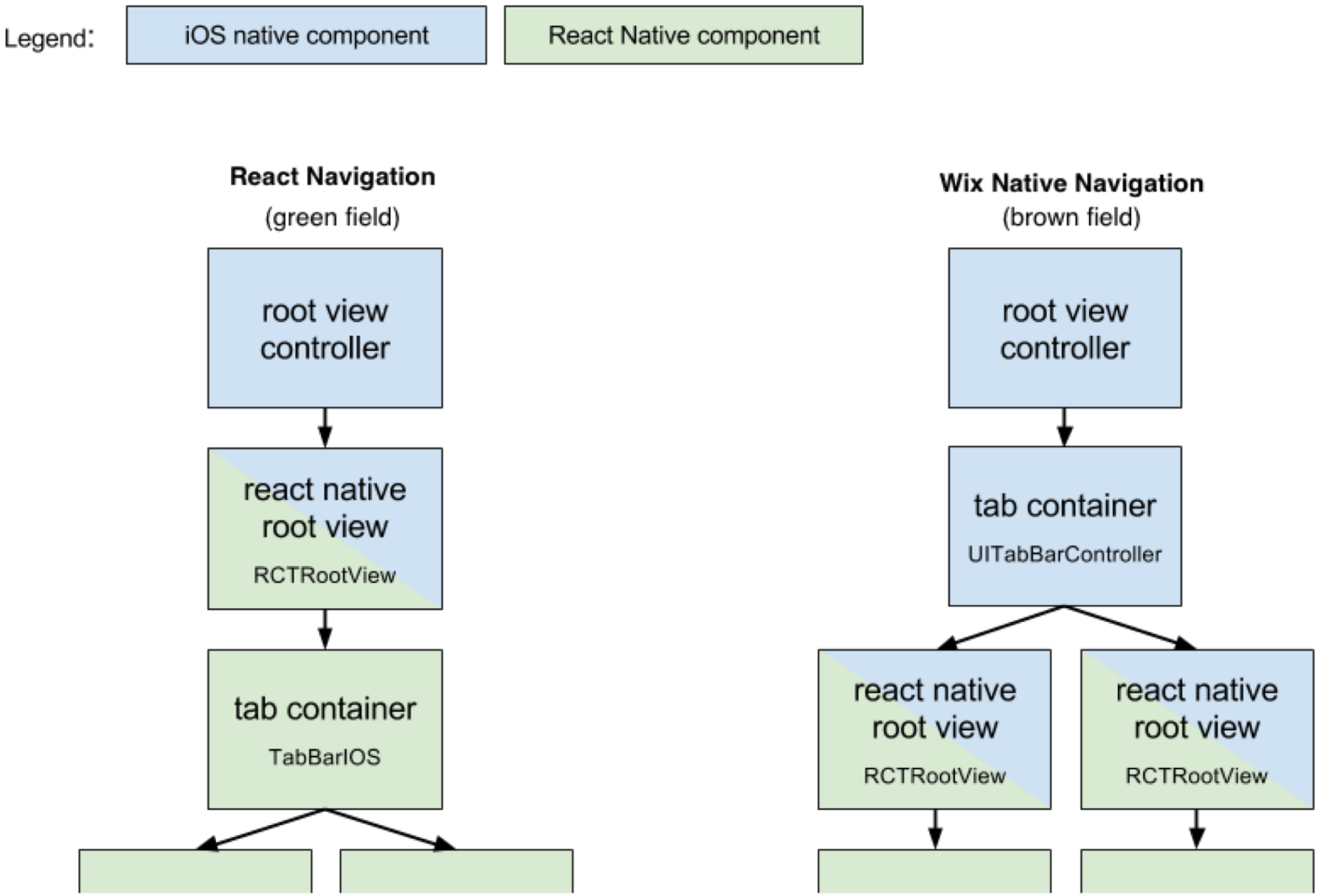
Source: Navigating React Native Navigation https://www.youtube.com/watch?v=42ogpJVwtw0w

Choosing the right navigation library

- React Router
  - similar to web
  - no transition animation out of the box
- Wix Native Navigation
  - Native implementation to native navigation component
  - v1 vs v2
- React Navigation
  - JS implementation with declarative animation API
  - suggested by RN official documentation

Problems with React Navigation

- Not using native navigation components
- Animation transitions declared in JavaScript
- Increase load on JS thread
- Workarounds to mimic native behaviour
- Do not handles duplicates (have to use throttle…)
- Header button problems with Redux
- Orientation bugs


Legend:
iOS native component
React Native component
React Navigation
(green field)
root view controller
react native root view
RCTRootView
tab container
TabBarIOS
Wix Native Navigation
(brown field)
root view controller
tab container
UITabBarController
react native root view
RCTRootView
react native root view
RCTRootView

Scrolling

Performance issues with large list

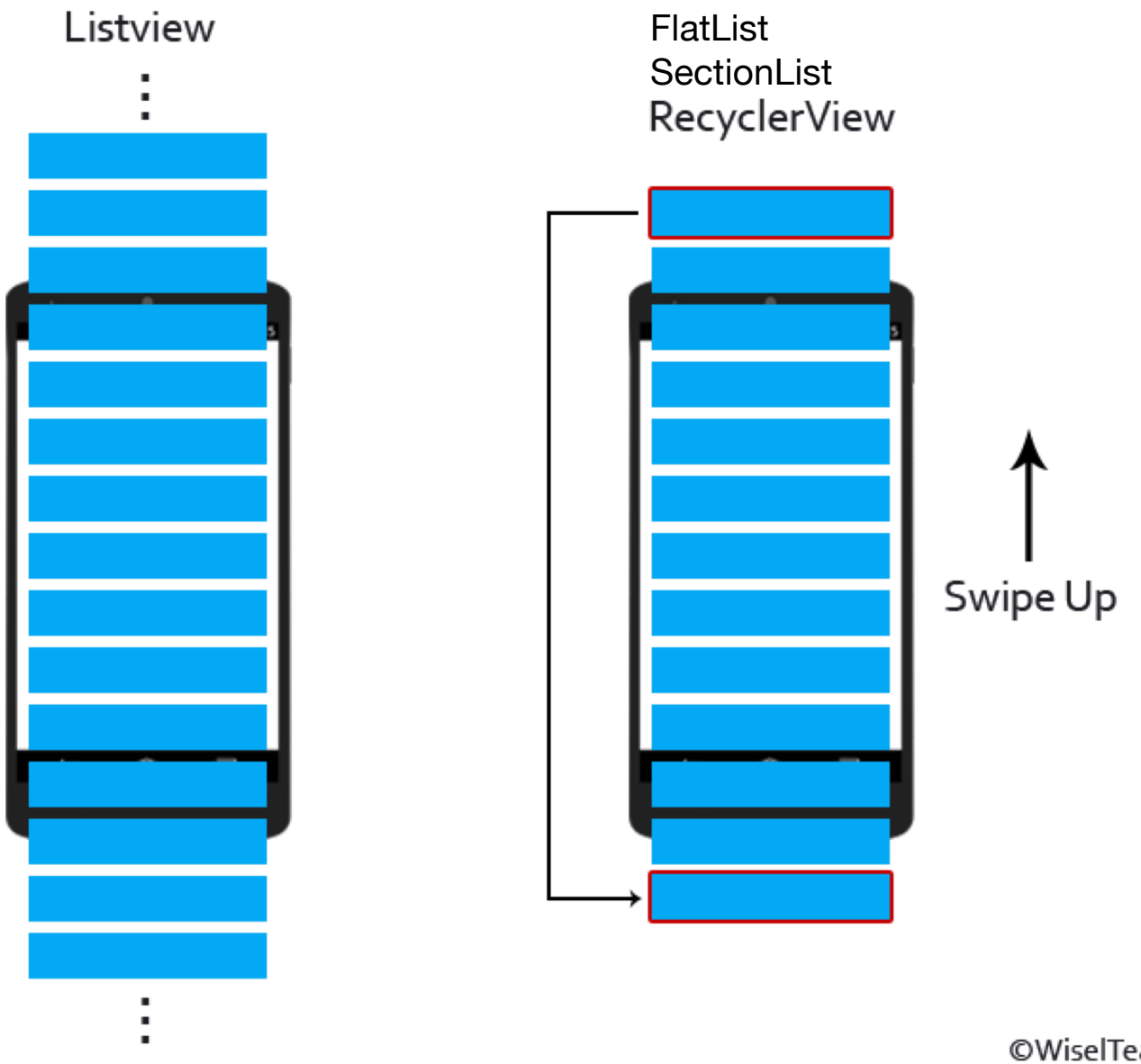
- React Native bridge limitation
- FlatList/SectionList vs ListView

`ListView` initial rendering is too slow or scroll performance is bad for large lists

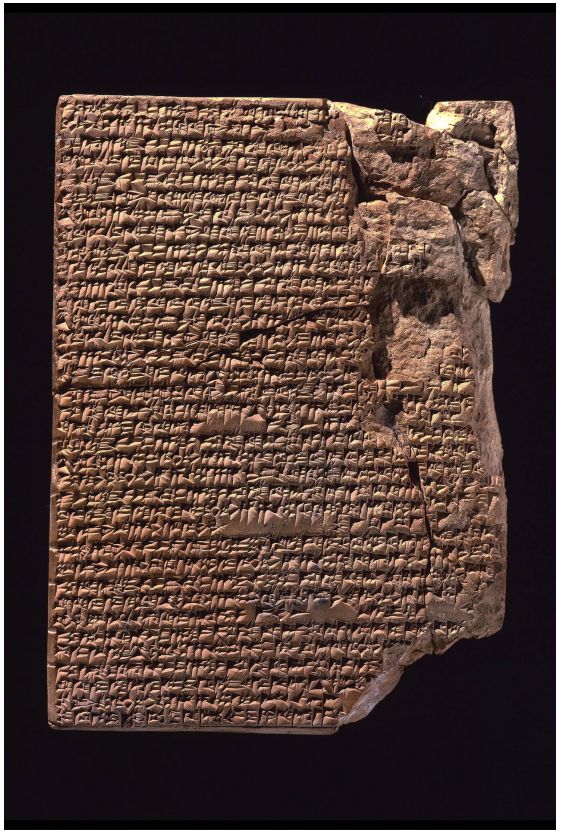
Use the new `FlatList` or `SectionList` component instead. Besides simplifying the API, the new list components also have significant performance enhancements, the main one being nearly constant memory usage for any number of rows.

If your `FlatList` is rendering slow, be sure that you've implemented `getItemLayout` to optimize rendering speed by skipping measurement of the rendered items.

Source: https://facebook.github.io/react-native/docs/performance.html


Listview
FlatList
SectionList
RecyclerView
Swipe Up
©WiselTeach

Persistence



How to persist data received from the API?

- AsyncStorage
  - Like web's LocalStorage API but asynchronous
  - Not suited for large data and relations
- Realm
  - Super fast mobile database for offline first and real time
  - Nested objects serialization is not supported
- SQLite
- Redux Persist
  - save redux store with a given storage driver
  - known issues with large data when using AsyncStorage
  - use FileSystemStorage instead
  - rehydrates asynchronously at app launch (can take a couple of seconds)

Interfacing with native code

Native module

```
$ npm install react-native-gemius --save
$ react-native link react-native-gemius
```

```
import RNReactNativeGemius from "react-native-gemius";

RNReactNativeGemius.sendPageViewedEvent();
```

Native module

```objc
RCT_EXPORT_METHOD(sendPageViewedEvent)
{
    GEMAudienceEvent *event = [GEMAudienceEvent new];
    [event sendEvent];
}
```

```java
@ReactMethod
public void sendPageViewedEvent() {
  AudienceEvent event = new AudienceEvent(reactContext);
  event.sendEvent();
}
```

Native module: asynchronous

```
async function measureLayout() {
  try {
    var {relativeX, relativeY, width, height} = await UIManager.measureLayout(
      100,
      100
    );

    console.log(relativeX + ':' + relativeY + ':' + width + ':' + height);
  } catch (e) {
    console.error(e);
  }
}

measureLayout();
```

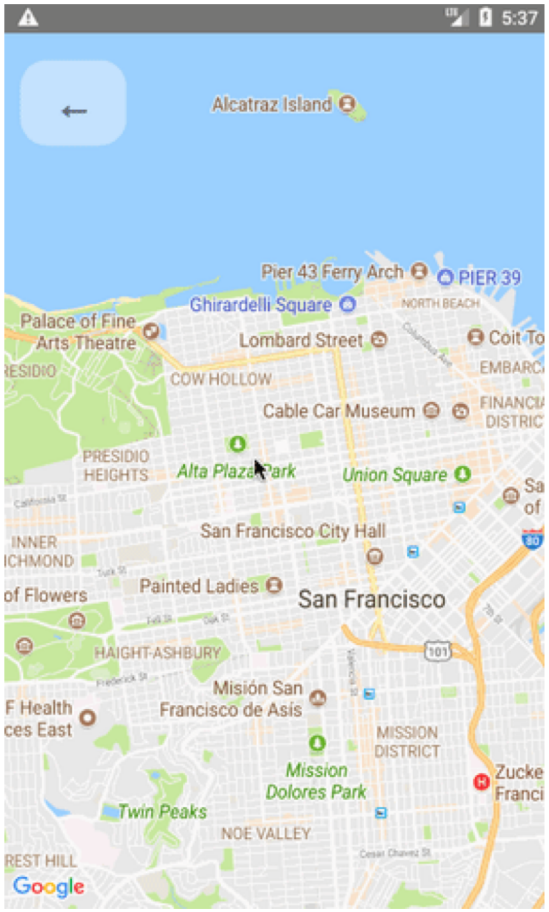
```java
import com.facebook.react.bridge.Promise;

public class UIManagerModule extends ReactContextBaseJavaModule {

...
  private static final String E_LAYOUT_ERROR = "E_LAYOUT_ERROR";
  @ReactMethod
  public void measureLayout(
      int tag,
      int ancestorTag,
      Promise promise) {
    try {
      measureLayout(tag, ancestorTag, mMeasureBuffer);

      WritableMap map = Arguments.createMap();

      map.putDouble("relativeX", PixelUtil.toDIPFromPixel(mMeasureBuffer[0]));
      map.putDouble("relativeY", PixelUtil.toDIPFromPixel(mMeasureBuffer[1]));
      map.putDouble("width", PixelUtil.toDIPFromPixel(mMeasureBuffer[2]));
      map.putDouble("height", PixelUtil.toDIPFromPixel(mMeasureBuffer[3]));

      promise.resolve(map);
    } catch (IllegalViewOperationException e) {
      promise.reject(E_LAYOUT_ERROR, e);
    }
  }
```

Native UI Component

```
<View >
  <ActivityIndicator size="small" color="#00ff00" />
</View>


<Image
  style={styles.stretch}
  source={require('/react-native/img/favicon.png')}
/>


 <Text style={{color: 'red'}}>
   and red
 </Text>
```

Native UI Component

```objc
@implementation RNTMapManager

RCT_EXPORT_MODULE()

- (UIView *)view
{
  return [[MKMapView alloc] init];
}

@end
```



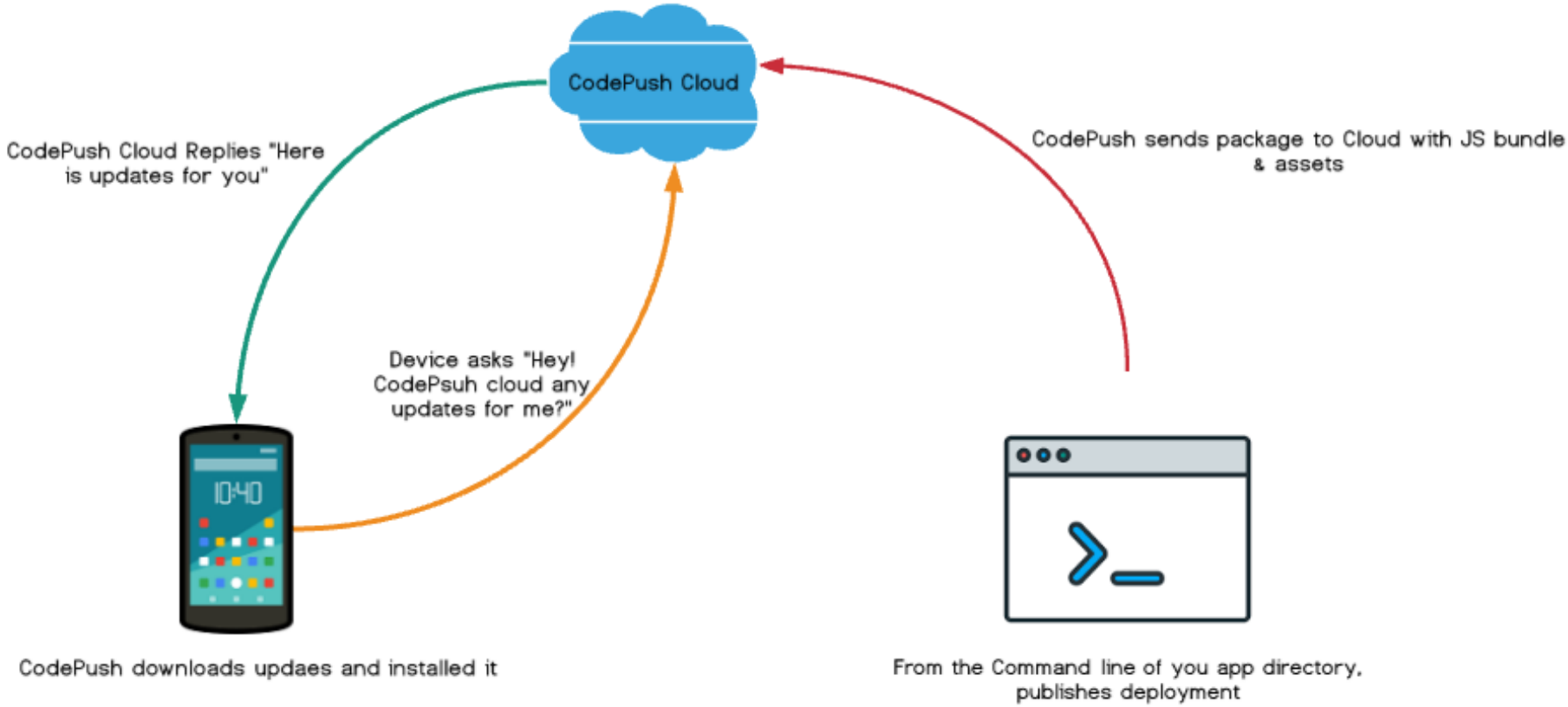
Native UI Component

```
// MapView.js

import { requireNativeComponent } from 'react-native';

// requireNativeComponent automatically resolves 'RNTMap' to 'RNTMapManager'
module.exports = requireNativeComponent('RNTMap', null);

// MyApp.js

import MapView from './MapView.js';

...

render() {
  return <MapView style={{ flex: 1 }} />;
}
```

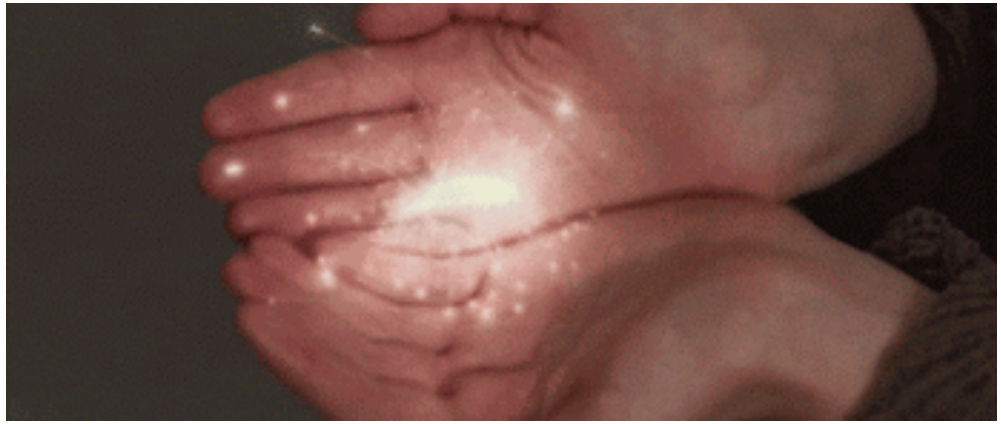
Google DFP Native ads

- DFP offers two solutions:
  - 1. Fetch data from DFP and render it
  - 2. Let DFP render the ads in WebViews
- Case 2 is perfect for Native UI Component
- Was not working in Android
- Ended up the Case 1 with a Native Module used a singleton to handle all the Native Ads instances

Update with CodePush


CodePush Cloud
CodePush Cloud Replies "Here is updates for you"
CodePush sends package to Cloud with JS bundle & assets
Device asks "Hey! CodePsuh cloud any updates for me?"
10:40
CodePush downloads updaes and installed it
From the Command line of you app directory, publishes deployment

Sharing logic



Code shared across platforms

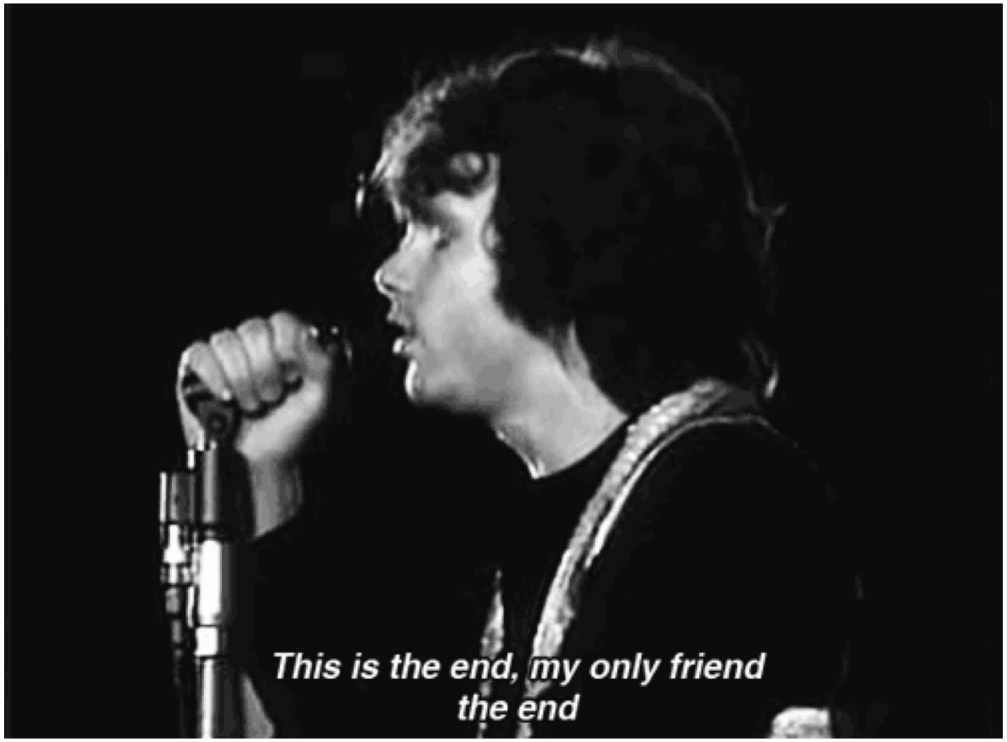
- API Client
- Data model
- Business logic (display texts, icons generation, domain conditional rendering…)
- Translation module
- iOS & android: 95% in common

Platform specific code

```
agencyAddress: {
  fontSize: 14,
  ...Platform.select({
    ios: { fontWeight: DimensionSizes.typefaces.light },
    android: { fontWeight: DimensionSizes.typefaces.ultraLight }
  }),
  fontStyle: "normal",
  lineHeight: 20.0,
  letterSpacing: 0.5,
  color: appColors.charcoalGreyTwo
} as StyleSheet.Style
```

```
const Touchable =
  Platform.OS === "android" ? TouchableNativeFeedback : TouchableOpacity;
```

Conclusion



Pros

- Productivity gains
- Shared logic
- No more siloed mobile team
- Bug/Crash reporting: Sentry + Fabric
- Build process: Bitrise + Fastlane
- Redux testing and debugging

Cons

- Monthly version updates (Breaking changes, TS definition files…)
- Some non spec compliant JS (Object.assign)
  - *TypeError: One of the sources for assign has an enumerable key on the prototype chain. Are you trying to assign a prototype property? We don't allow it*
  - *TypeError: In this environment the sources for assign MUST be an object. This error is a performance optimization and not spec compliant.*
- Minor differences between iOS and Android JS engines (Number.toLocaleString…)
- Licensing (MIT for React but not RN)
- Custom gestures and interactions
- **No native navigation component out of the box**

11
AVR.

mercredi 11 avril 2018, 18:30

First Mobile Apps Meetup in Luxembourg!

Organisé par Oussama G.

The location will be confirmed in a few days. It will be in Luxembourg-Ville. Doors open at 18:30. Talks will begin at 19:00. For the first meetup: ** About the mobile apps meetup - 5min Quick presentation of the goal of this meetup and how you can participate in the organization. ** React Native in Production - 30min by Oussama Ghalbzouri - https://github.com/ou2s This talk is obviously about the cross-platform mobile apps...

22 personnes y vont

Participer

Q-LEAP
13 Rue Beaumont · Luxembourg

3 commentaires

Thanks

github.com/ou2s