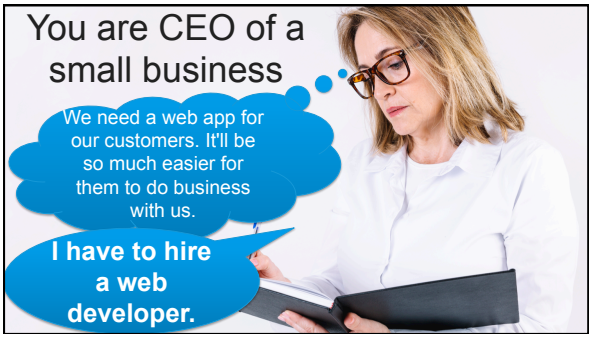
Hello React Native!

What is React Native?




Then customers say ...
We love your web app, but we'd have been here earlier if you were in the App Store.
How do I get into the App Store?
Hire an iOS dev who knows Swift


Then customers say ...
Hey, most of us have Android devices. Why aren't you in the Play Store?
How do I get into the Play Store?
Hire an Android dev who knows Java


Then customers say ...
We want you on our Blackberries!
Isn't there some way I can hire one person who can develop on all platforms?
But you can't afford three developers!

What is React Native?

"A framework and set of tooling that allows us to write applications with a single set of JavaScript code that will run on devices with different OSs, controlling both the presentation and behavior."

-- Rap Payne

Create apps for ...

The app is completely native, indistinguishable from one written in Swift, Objective-C, Java, or Kotlin

- Not in a WebView like Cordova or Ionic
- Not in a host
- Not a hybrid/partially native app like Xamarin.

Yet, you don't need ...

a Mac
xcode
Java/Kotlin
Objective-C/Swift

You only need the React Native compiler

When your app is ready, create an apk or ipa file which can be sent to the Google Play store or the App Store

Download on the App Store

ANDROID APP ON Google play

Of course, you'll still need to be registered to submit

Elements render natively

- React Native <Button>s render as iOS buttons on iOS and Android buttons on Android.
- Same with every element where it is possible
- (... in some places it is impossible)

Unfortunately we can't share anything with a web application

No HTML

No CSS

But you can share pure JavaScript libraries if you're careful.

And you can use most of the innumerable libraries in your code

As long as they don't depend on HTML, <canvas>, or SVG

- axios
- date-fns
- frisbee
- immutable
- lodash
- redux
- moment
- etc. etc.

Should we use React Native?

Maybe if ...

- You need iOS and Android apps in the stores
- Your team is already strong in functional JavaScript
- Or they're strong in React
- You value speed over quality

Probably not if ...

- The app is really complex -- like a game
- You already have iOS & Android devs
- The apps need to be different on the different platforms

But be careful. It isn't perfect

If you want the experience being different on both platforms, you must create different components that do the same things but look slightly different.

This is a maintenance problem and requires that you have people that know both iOS and Android.

Even if you want them to be the same ...

- There's a huge variety of Android devices and very few iOS devices. So you may end up doing checks if iOS => do one thing; else => do other things. The more of these you have, the worse RN fits your ideal solution.
- apk size is much bigger with RN (About 10Mb! as of 7/2018)
- RN lags behind iOS features and Android features. ie. When iOS adds a new feature like SafeAreaNavigation, it takes a while before it shows up in RN.
- When a bug occurs, it is really hard to tell if it is in iOS or in Android or in RN.

... and it is still new tech

- The framework rules seems inconsistent
- The tooling seems unreliable

- APIs are changing quickly
- Tooling is changing quickly
- Libraries are changing quickly

- You have iOS things that change (API, OS, etc) and Android things that change and you now have RN things that change and probably JavaScript libraries that it depends on that change. That's a lot of breaking changes and things that can be deprecated and require maintenance in one code base!

What's React Native code look like?

Categories of RN components

Category	Some components
Layout	Modal, View, SafeAreaView, ScrollView, RefreshControl, KeyboardAvoidingView, StatusBar, WebView
Single-value	Text, TextInput, Slider, Switch, Image
List	Picker, FlatList, SectionList
Touchable	Button, TouchableHighlight, TouchableNativeFeedback, TouchableOpacity, TouchableWithoutFeedback
Others	ActivityIndicator, Platform-specific components

React Native vs HTML

HTML	React Native
	<View>
<p>	<Text>
<img />	<Image />
<input type='text' />	<TextInput />
<input type="range" />	<Slider />
<button>	<Button />

A class-based component

MyComponent.js

```
import React from 'react';
import { View, Text } from 'react-native';

class MyComponent extends React.Component {
 render() {
    return (
      <View>
        <Text>Hello World</Text>
      </View>
    );
  }
}

// example usage

<MyComponent />
```

A stateless functional component

MyComponent.js

```
import React from 'react';
import { View, Text } from 'react-native';

const MyComponent = () => (
  <View>
    <Text>Hello World </Text>
  </View>
)

// example usage

<MyComponent />
```

MyComponent.js

```
class MyComponent extends React.Component {
  render() {
    return (
      <Notify message="Hello World" />
    );
  }
}
```

We use props and compose just like in React for the web

Notify.js

```
const Notify = props => <Text>{props.message}</Text>
```

tl;dr

- React Native allows developers to use JavaScript and React to create cross-platform apps
- They can be deployed to the Apple App Store and the Google Play Store because they are 100% native.
- No plugins
- No WebViews
- No tricks