HOW TO DEVELOP A REACT-NATIVE APP

COLOGNE.JS MEETUP, 9TH DECEMBER 2015, CHRISTOPH JEROLIMOV

ABOUT MYSELF
CHRISTOPH JEROLIMOV

Developer with passion, current focus on mobile.

Co-Founder bringnow and mobile.cologne meetup

Follow me @jerolimov

Contact me christoph@jerolimov.de

BENEFITS
REACT.JS

» Declarative view-only library for the web (one-way data flow simplifies data-binding)

» Define how the app should look based on a dataset

» React handles how the UI updates when the data changes

» Component model which makes encapsulated, reuseable and testable modules

BENEFITS
REACT-NATIVE

» Same as React.js!

» Write JS, renders native views (no WebView!)

» Share code and knowhow

» Better developer experience (DX)

» (Easy) Integration options in both directions

React

A JAVASCRIPT LIBRARY FOR BUILDING USER INTERFACES

Get Started Download React v0.14.3

JUST THE UI

Lots of people use React as the V in MVC. Since React makes no assumptions about the rest of your technology stack, it's easy to try it out on a small feature in an existing project.

VIRTUAL DOM

React abstracts away the DOM from you, giving a simpler programming model and better performance. React can also render on the server using Node, and it can power native apps using React Native.

DATA FLOW

React implements one-way reactive data flow which reduces boilerplate and is easier to reason about than traditional data binding.

A Simple Component

React components implement a `render()` method that takes input data and returns what to display. This example uses an XML-like syntax called JSX. Input data that is passed into the component can be accessed by `render()` via `this.props`.

JSX is optional and not required to use React. Try clicking on "Compiled JS" to see the raw JavaScript code produced by the JSX compiler.

React Native

A FRAMEWORK FOR BUILDING NATIVE APPS USING REACT

React Native enables you to build world-class application experiences on native platforms using a consistent developer experience based on JavaScript and React. The focus of React Native is on developer efficiency across all the platforms you care about — learn once, write anywhere. Facebook uses React Native in multiple production apps and will continue investing in React Native.

Get started with React Native

Native Components

With React Native, you can use the standard platform components such as UITabBar on iOS and Drawer on Android. This gives your app a consistent look and feel with the rest of the platform ecosystem, and keeps the quality bar high. These components are easily incorporated into your app using their React component counterparts, such as TabBarIOS and DrawerLayoutAndroid.

```
// iOS

var React = require('react-native');
var { TabBarIOS, NavigatorIOS } = React;

var App = React.createClass({
  render: function() {
    return (
      <TabBarIOS>
```

2007

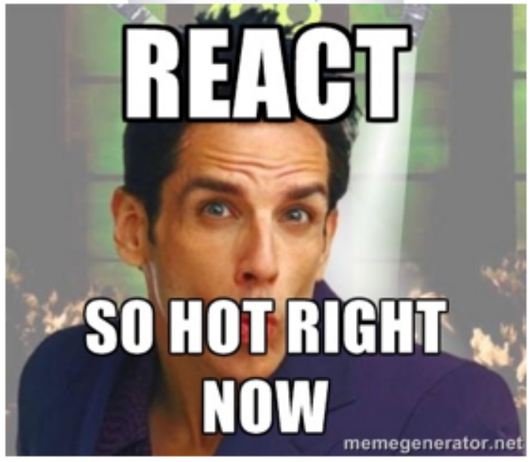
2009

2012

2013

2014

2015



REACT.JS (*SPRING 2013)

GitHub rank #8, ~32.500 stars, 570+ contributors

“HTML is just the beginning.”

REACT.JS (*SPRING 2013)

GitHub rank #8, ~32.500 stars, 570+ contributors

“HTML is just the beginning.”

REACT-NATIVE (*SPRING 2015)

GitHub rank #27, ~23.500 stars, 420+ contributors

angular #3, jquery #7, docker #18, atom #30

WRITE ONCE, RUN ANYWHERE
INITIALLY JAVA, BUT HTML5 TOO

~~WRITE ONCE, RUN ANYWHERE~~

LEARN ONCE, USE ANYWHERE

HISTORY
REACT-NATIVE

Announced 01/2015
v0.1 @ 03/2015 First public release
v0.5 @ 06/2015 Accessibility API
v0.6 @ 06/2015 View Inspector, Performance tools
v0.8 @ 07/2015 New Animation API
v0.11 @ 09/2015 Android support
v0.13 @ 10/2015 Linux and Windows support
v0.15 @ 11/2015 iOS requires Xcode 7+, Geolocation & Intents on Android

ROADMAP
REACT-NATIVE

» Implement missing views like Maps, Webview and others on Android.

» Android M permissions

» Improve the performance

» API improvements like a unified <ViewPager> <AndroidViewPager> and <ScrollView pagingEnabled={true}>

» ...

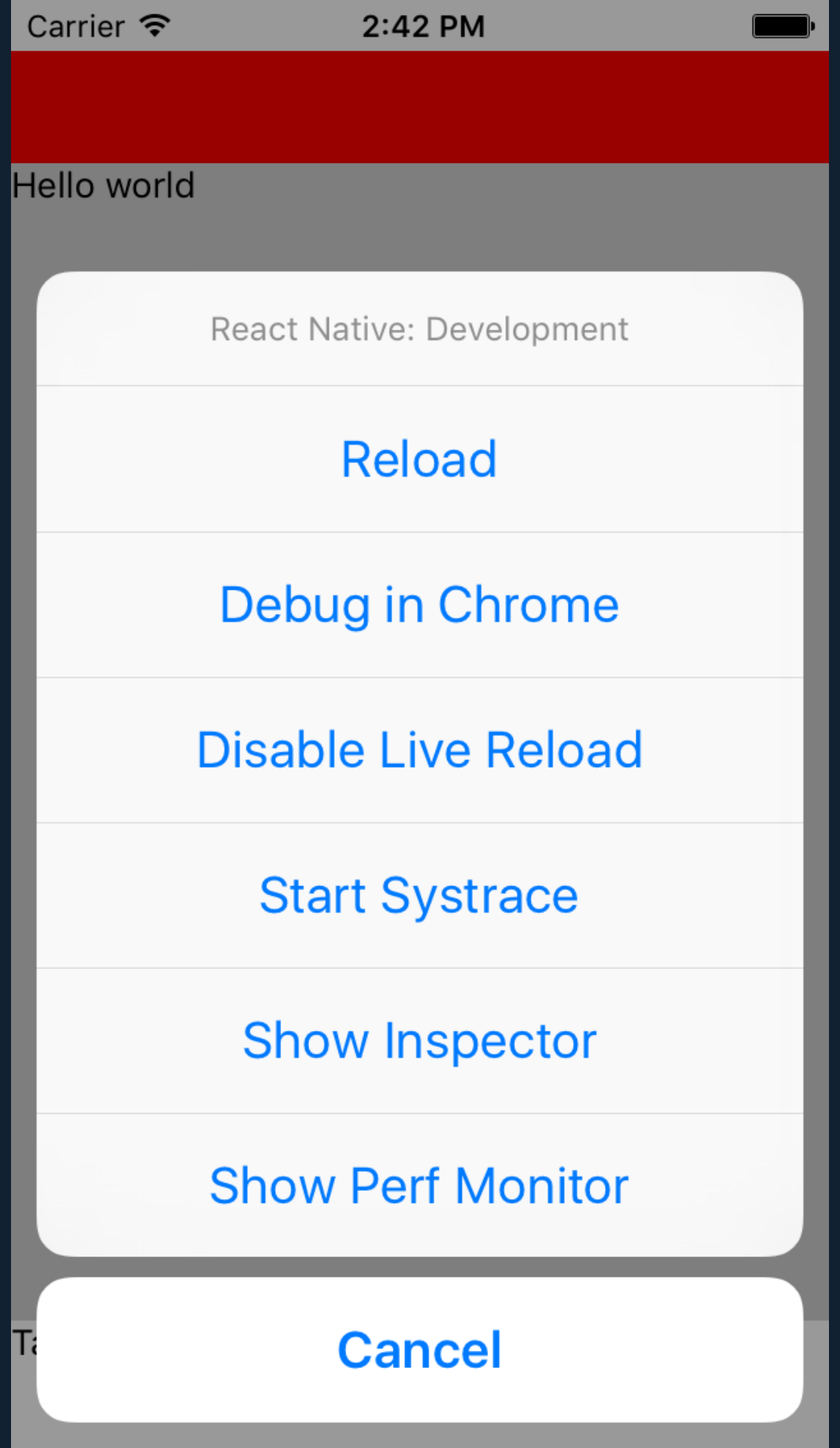
DEVELOPER EXPERIENCE
THE WEBSTUFF WE LOVE

» Modern HTML- & CSS-like => JSX + Flexbox

» Modern JavaScript w/ optional Flow (or TypeScript)

» Hot reloading (⌘R) & Live Reload

» Debugger, UI Inspector, Profiling


Carrier
2:42 PM
Hello world
React Native: Development
Reload
Debug in Chrome
Disable Live Reload
Start Systrace
Show Inspector
Show Perf Monitor
Cancel

HOW DOES IT WORK?
DEVELOPMENT

» You write javascript in your favorited editor

» App communicates with a local http server

Server starten:

```
npm start
```

HOW DOES IT WORK?
PRODUCTION

» Precompiled, minified JS bundled within the app

» Code updates are technical possible.. and allowed

Bundlen:

```
react-native bundle --entry-file index --platform ios --bundle-output main.jsbundle
```

HOW DOES IT WORK?
TECHNICAL

» Based on a minimal JS VM: JavaScriptCore
which is part of WebKit

» Android bundles the library w/ the app (3,5 MB)

» iOS 7 includes a shared version already

» JS <-> Native bridge is replaceable
Use a inter process model by default and can also run the app in a remote process, for example in the Chrome (for Debugging)

HOW DOES IT WORK?
SUPPORTED PLATFORMS

» Android 4.1+, >= 93 % [1]

» iOS 7+, >= 96 % [2] [3]

[1] https://developer.android.com/about/dashboards/index.html

[2] https://david-smith.org/iosversionstats/

[3] https://developer.apple.com/support/app-store/

PRO AND CONS
FOR A JS DEVELOPER

» + Native development (better UX than WebViews)

» + Same ecosystem (npm, babel, React.js, ...)

» + Code sharing (with web / backend)?!

» - More complex builds with "native problems"

» - Maybe less stable than Cordova

» - No media queries (but Flexbox and code switches)

PRO AND CONS
FOR A NATIVE DEVELOPER

» + Improved DX (Live Reloading!)

» + Cross Platform (learn once, use anywhere)

» + Code sharing between Android and iOS

» + Integration into / of other native components

» - Another programming language and maybe paradigma

» - Missing tool support (auto completion, etc.)

» - Maybe API limitation (but you can DIY)

PRO AND CONS
FOR THE PRODUCT OWNER

» + Knowhow sharing means more flexiblity

» + Code sharing

» + Native development (better UX than WebViews)

» + Solid ecosystem (node, npm, babel, redux, ...)

» - Future-proofness?

GETTING STARTED[4]
GENERAL SOFTWARE REQUIREMENTS

» OSX is the most common dev platform (at FB) and also required for iOS

» Linux and Windows should work[5]

» You need Node.js 4+, nvm is recommended

```
brew install watchman
npm install --global react-native
```

[4] http://facebook.github.io/react-native/docs/getting-started.html

[5] http://facebook.github.io/react-native/docs/linux-windows-support.html

GETTING STARTED
PLATFORM DEPENDENCIES

» Android SDK[6]

» Xcode 7+ for iOS

[6] http://facebook.github.io/react-native/docs/android-setup.html

GETTING STARTED
PROJECT SETUP

```
npm install -g react-native-cli

react-native init MyFirstApp

# IMHO missing by default:
cd MyFirstApp
npm install --save-dev react-native-cli
```

GETTING STARTED
ANDROID

Start an Android emulator or connect your device.

```
adb devices # should list at least one device

adb reverse tcp:8081 tcp:8081
react-native run-android
```

GETTING STARTED
IOS

Open the Xcode project from the ios folder and start the app via Project > Run.

```
open ios/*.xcodeproj
npm start
# run the app within Xcode
```

DEMO

JAVASCRIPT

ECMASCRIPT 2015/2016

~~OPTIONAL: TYPESCRIPT~~

ECMASCRIPT 2015
SPREAD OPERATOR

```
// var firstname = person.firstname;
// var lastname = person.lastname;
const { firstname, lastname } = person;

const person2 = {
    firstname // :firstname
    lastname // :lastname
}

// Object.assign({}, person2, { firstname: 'New name' })
const person3 = {
    ...person2,
    firstname: 'New name'
}
```

ECMASCRIPT 2015
CLASSES

```
class MyButton extends Button {
    constructor(props) {
        super(props)
    }

    componententDidMount() {
        super.componententDidMount();
        this.mounted = true;
    }
}
```

ECMASCRIPT 2015
CLOSURES

```
// var x = function() { ... }
const min = (a, b) => (a < b ? a : b);
const max = (a, b) => {
    return a > b ? a : b;
};
```

ECMASCRIPT 2015
IMPORT / EXPORT

```
import React from 'react'; // var React = require('react');

class MyButton extends React.Component {
}

export default MyButton;

// Short:
import React, { Component, View, Text } from 'react';

export default class MyButton extends Component {
}
```

REACT + JSX

REACT.JS
JSX EXAMPLE

```
class HelloWorld {
    render() {
        return <span>Hello World</span>;
    }
}
```

REACT.JS
JSX TRANSFORMATION

```
class HelloWorld {
    render() {
        return <Text>Hello World</Text>;
    }
}
```

Will be transformed with babel to:

```
class HelloWorld {
    render() {
        return React.createElement(Text, null, 'Hello World');
    }
}
```

REACT.JS
LOCAL STATE

```
class HelloWorld {
    constructor() {
        this.state = { say: 'Hello' };
    }

    render() {
        const goodbye = () => {
            this.setState({ say: 'Goodbye' });
        }

        return (
            <span onclick={ goodbye }>
                { this.state.say } World!
            </span>
        );
    }
}
```

REACT.JS
EXTERNAL PROPS

```
class HelloWorld {
    render() {
        return (
            <span onclick={ goodbye }>
                { this.state.say } { this.props.user }
            </span>;
        );
    }
}

class MyApp {
    render() {
        return <html><body><HelloWorld name="World" /></body></html>;
    }
}
```

REACT-NATIVE
UI COMPONENTS

» View

» Text

» Image

» TouchableOpacity

» ...

» ScrollView

» ListView

REACT-NATIVE
JSX HELLO WORLD

```
class HelloWorld {
    render() {
        return <Text>Hello World</Text>;
    }
}

AppRegistry.registerComponent('MyApp', () => HelloWorld);
```

REACT-NATIVE
A SIMPLE VIEW HIERACHY

```
<View>
    <Text>Hello World</Text>
    <Text>Hello World</Text>
</View>
```

Flexbox is enabled for any view!

Vertical layout is the default.

REACT-NATIVE
CHANGED FLEXBOX DIRECTION

```
<View style={{ flexDirection: 'row' }}>
    <Text>Hello World</Text>
    <Text>Hello World</Text>
</View>
```

REACT-NATIVE
STYLESHEETS, CSS-LIKE

```
const bold = {
    fontWeight: 'bold' // A string!
};

const styles = StyleSheet.create({
    bold: {
        fontWeight: 'bold'
    }
});

<View style={{ borderWidth: 1, borderColor: 'red' }}>
    <Text style={ bold }>Hello World</Text>
    <Text style={ styles.bold }>Hello World</Text>
</View>
```

REACT-NATIVE
FLEXBOX

```
// Grow 100% with childs 50%, 30% and 20%
<View style={{ flex: 1, flexDirection: 'row' }}>
    <View style={{ flex: 0.5, backgroundColor: 'red' }} />
    <View style={{ flex: 0.3, backgroundColor: 'blue' }} />
    <View style={{            backgroundColor: 'green' }} />
</View>;

// Grow 100% where first and last child is fix
<View style={{ flex: 1 }}>
    <View style={{ height: 64, backgroundColor: 'red' }} />
    <View style={{            backgroundColor: 'blue' }} />
    <View style={{ height: 50, backgroundColor: 'green' }} />
</View>;
```

DEMO

STRUCTURE YOUR APP
MANAGE THE STATE

- Flux and the single source of truth (state)
- Split your code into
  - "dump presentation" components and
  - "smarter containers"
- Persist the state (incl. the navigation stack)
  makes Auto Reloading finally awesome

```
npm install --save redux react-redux@3.1 redux-persist
```

STRUCTURE YOUR APP
VIEW STACK

» Navigator vs NavigatorIOS, both supports loops,...

» Recommended: Serializable routes (to persist them)

» Then render views based on a plain json route:

```
navigator.push({ identifier: 'UserDetail', userId: 1234 });

renderScene(route, navigator) {
    if (route.identifier === 'UserDetail') {
        return <UserDetail userId={ route.userId };
    } else {
        return <ViewNotFound />;
    }
}
```

STRUCTURE YOUR APP
PLATFORM SPECIFIC CODE

Auto-select component based on a file suffix:

```
Slider.android.js
Slider.ios.js
```

Or a good old platform switch:

```
import { Platform } from 'react-native';

if (Platform.OS === 'android') {
    // ...
} else {
    // ...
}
```

LINKS

- facebook.github.io/react-native (incl. API docs)
- facebook.github.io/react/blog/ (for both libs)
- github.com/facebook/react-native/releases
- www.reactnative.com (not the offi. blog)
- github.com/rackt (redux, react-redux, ..)
- github.com/jondot/awesome-react-native
- react.parts
- speakerdeck.com/frantic/under-the-hood 1
- speakerdeck.com/mkonicek/under-the-hood 2

THANK YOU. ANY QUESTIONS?