Minicurso de React-Native

@felipemfp e @chicobentojr
Célula **A**cadêmica de
Desenvolvimento de **S**oftware

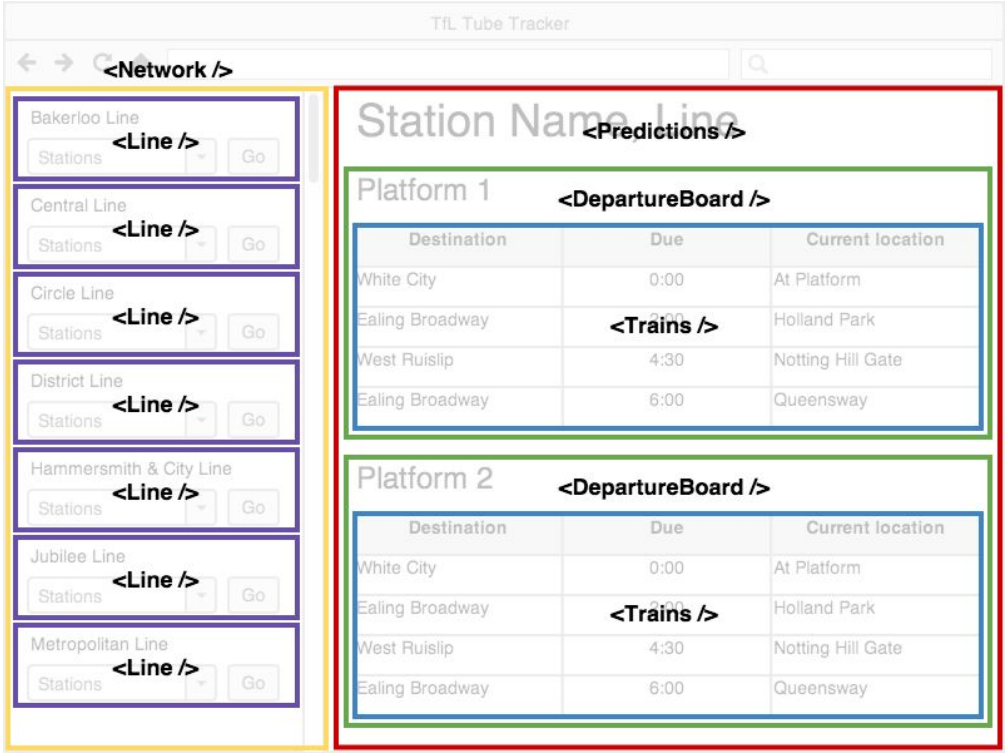
Agenda

- Introdução ao React
- Introdução ao React-Native
- Componentes Inclusos
- Componentes Open-Source
- Expo
- Prática

Introdução ao React

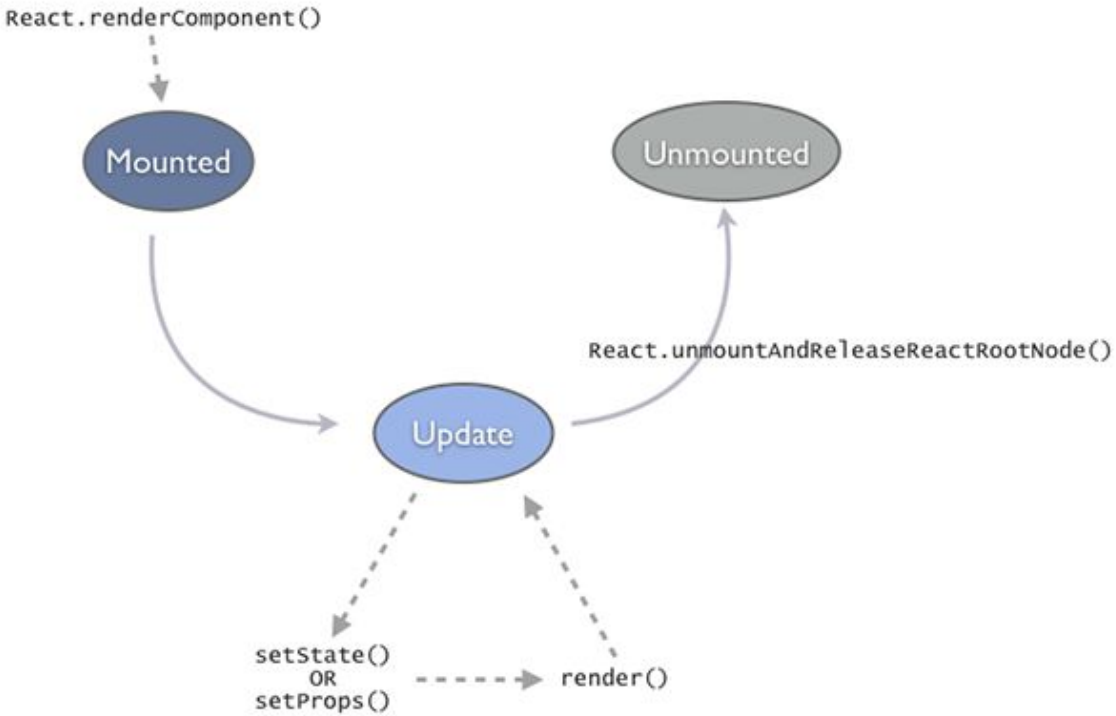
Apenas uma biblioteca

Tudo são componentes

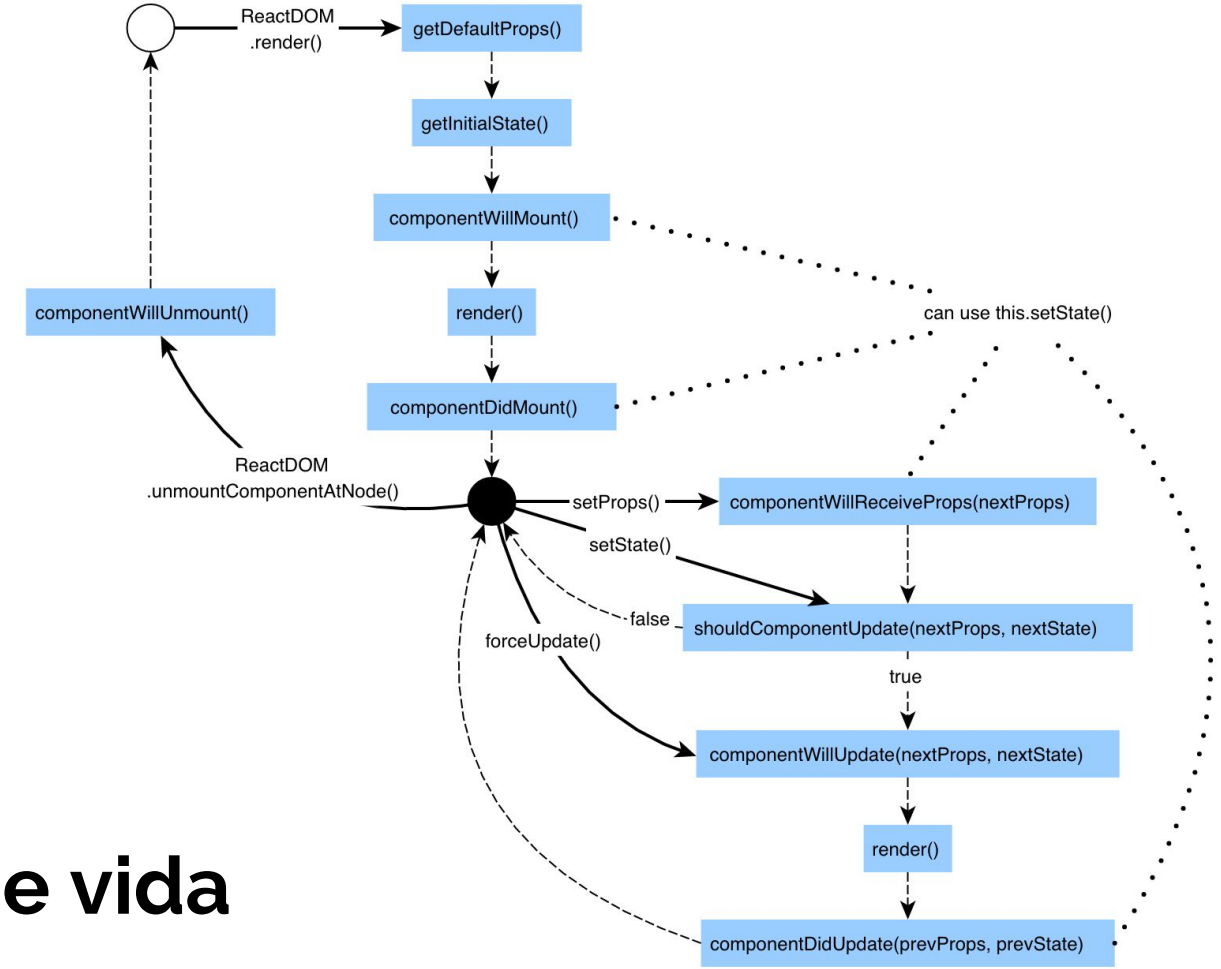


“Componentes interativos, testáveis e reutilizáveis”

Ciclo de vida simplificado



Ciclo de vida



Como escrevemos componentes em React?

- Componentes são objetos
- render() e JSX

JSX

```
var Hello = React.createClass({
    render: function() {
        return (
            <div>
                <h1>Hello from Kode</h1>
                <p>Maybe you prefer a hello from Adele</p>
            </div>
            );
    }
});

ReactDOM.render(<Hello/>, document.getElementById('react-target'));
```

Como escrevemos componentes em React?

- Possuem *props* e *state*
- Possuem funções:
  - *setState()*
  - *getInitialState()*
  - *getDefaultProps()*
  - [...]

props

```
1  import React from 'react'
2  import ReactDOM from 'react-dom'
3
4  var ComplimentMachine = React.createClass({
5    render: function () {
6      return (
7        <div>
8          <h1>What's your name?</h1>
9          {this.props.compliment}
10       </div>
11     )
12   }
13 })
14
15 ReactDOM.render(
16   <ComplimentMachine compliment="You're a great listener" />,
17   document.getElementById('root')
18 )
```

state


var CommentBox = React.createClass({
loadFromServer: function() {
return [
{id: 1, author: "Stephan Schmidt", text: "This is one comment"},
{id: 2, author: "Jens Schumann", text: "This is *another* comment"}
];
},
getInitialState: function() {
return {data: []};
},
componentDidMount: function() {
// Ajax code ....
this.setState({data: this.loadFromServer()});
},
render: function() {
return (

<h1>Comments</h1>
<CommentList data={this.state.data} />

);
}
});

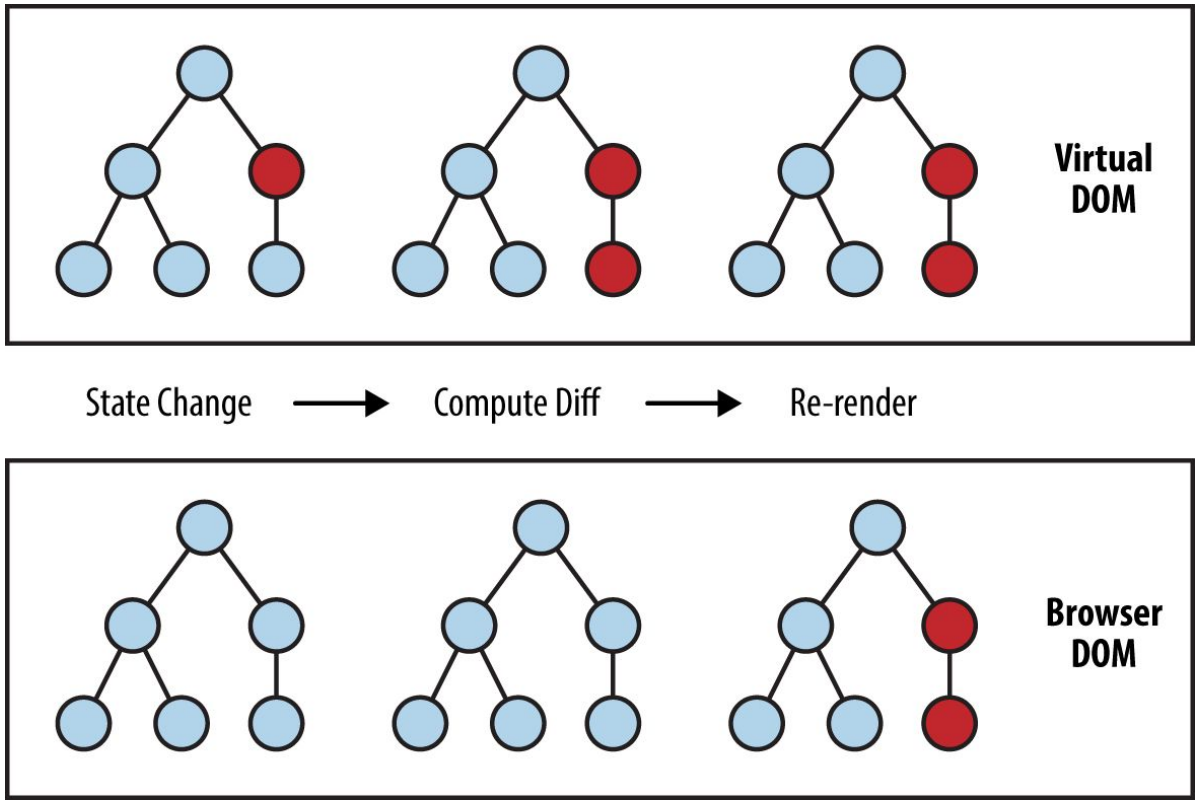
Componentes *stateless*

```
const Intro = () => {
    return <p>Hi there...</p>
}
```

Como escrevemos componentes em React?

- Dados fluem numa única direção
- Renderização conforme necessário (Virtual DOM)

Virtual DOM


Virtual
DOM
State Change
Compute Diff
Re-render
Browser
DOM

Introdução ao React-Native

React-Native é React

Introdução ao React-Native

- Componentes são renderizados como componentes nativos
- Os mesmos conceitos são utilizados para construir aplicações
- Suporte oficial para iOS e Android
- Suporte ainda para Universal Windows

Introdução ao React-Native

- Ciclo de vida (*props*, *state*...) do componentes são o mesmo
- Continuamos usando JSX

CSS não é o mesmo

Adicionando estilo aos componentes nativos

- “CSS” como *prop*
- StyleSheet é quase CSS

StyleSheet

```
var styles = StyleSheet.create({
  base: { width: 38, height: 38 },
  background: { backgroundColor: '#222222' },
  active: { borderWidth: 2, borderColor: '#00ff00' },
});

<Text style={styles.base} />
<View style={styles.background} />
```

Construindo layout com React-Native

- O mesmo Flexbox da web
- Com diferentes *defaults:*
  - flexDirection: column
  - alignItems: stretch

Componentes inclusos

Componentes inclusos

- **ActivityIndicator**
- **Button**
- **CheckBox**
- DatePickerIOS
- DrawerLayoutAndroid
- **FlatList**
- **Image**
- KeyboardAvoidingView
- ListView
- MaskedViewIOS
- Modal
- NavigatorIOS
- **Picker**
- Picker.Item
- PickerIOS
- ProgressBarAndroid
- ProgressViewIOS
- RefreshControl
- **ScrollView**
- **SectionList**
- SegmentedControlIOS
- Slider
- SnapshotViewIOS
- StatusBar

Componentes inclusos

- Switch
- TabBarIOS
- TabBarIOS.Item
- **Text**
- **TextInput**
- ToolbarAndroid
- TouchableHighlight
- TouchableNativeFeedback
- **TouchableOpacity**
- TouchableWithoutFeedback
- **View**
- ViewPagerAndroid
- VirtualizedList
- **WebView**

API inclusas

- AccessibilityInfo
- ActionSheetIOS
- Alert
- AlertIOS
- Animated
- AnimatedValue
- AnimatedValueXY
- AppRegistry
- AppState
- **AsyncStorage**
- BackAndroid
- BackHandler
- CameraRoll
- **Clipboard**
- DatePickerAndroid
- Dimensions
- Easing
- **Geolocation**
- ImageEditor
- ImagePickerIOS
- ImageStore
- InteractionManager
- Keyboard
- LayoutAnimation

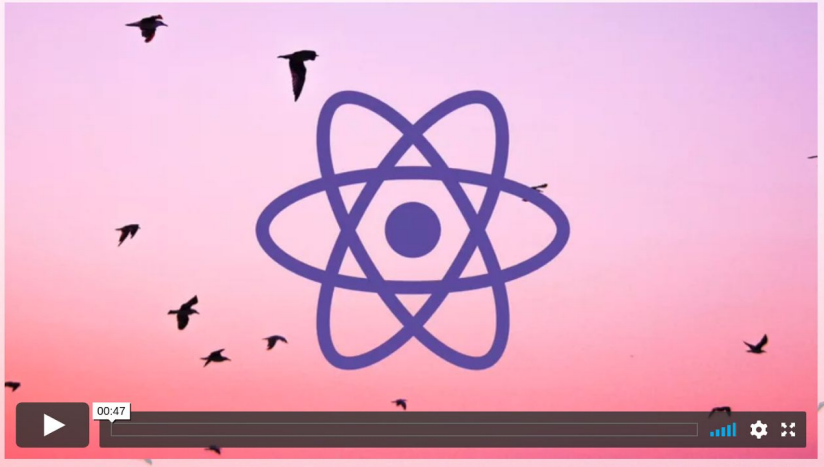
API inclusas

- **Linking**
- NetInfo
- PanResponder
- PermissionsAndroid
- PixelRatio
- PushNotificationIOS
- **Settings**
- **Share**
- StatusBarIOS
- StyleSheet
- Systrace
- TimePickerAndroid
- ToastAndroid
- **Vibration**
- VibrationIOS
- Layout Props
- Picker Style Props
- Shadow Props

Componentes Open-Source

http://www.awesome-react-native.com/

Navigation for React Native



Get Started →

DOCS GITHUB CONNECT START A PROJECT MARKET

NativeBase

Essential cross-platform UI components for React Native

100% open source

Currently v2.3.2

6,786 stars 731 forks

Without NativeBase

```
var style = StyleSheet.create({
    button: {
        backgroundColor: '#99AAFF',
        borderRadius: 5,
        borderWidth: 1,
        borderColor: '#000033'
    }
});

<TouchableOpacity style={style.button}>
    <Text style={{color: 'white'}}>Click me!</Text>
</TouchableOpacity>
```

With NativeBase

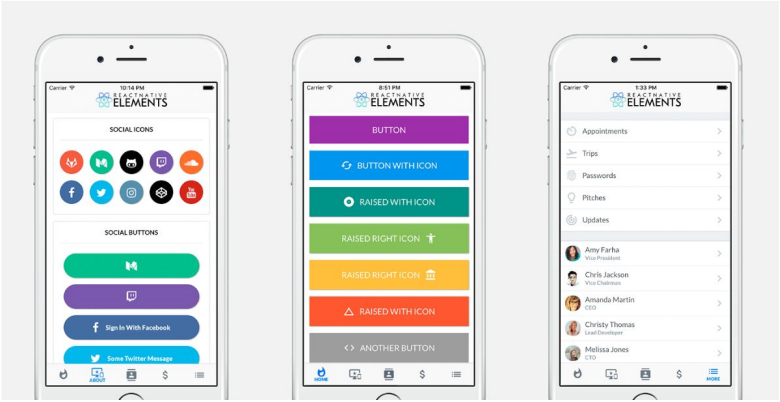
```
<Button>
  <Text>Button</Text>
</Button>
```

Home

REACTNATIVE ELEMENTS

React Native Elements

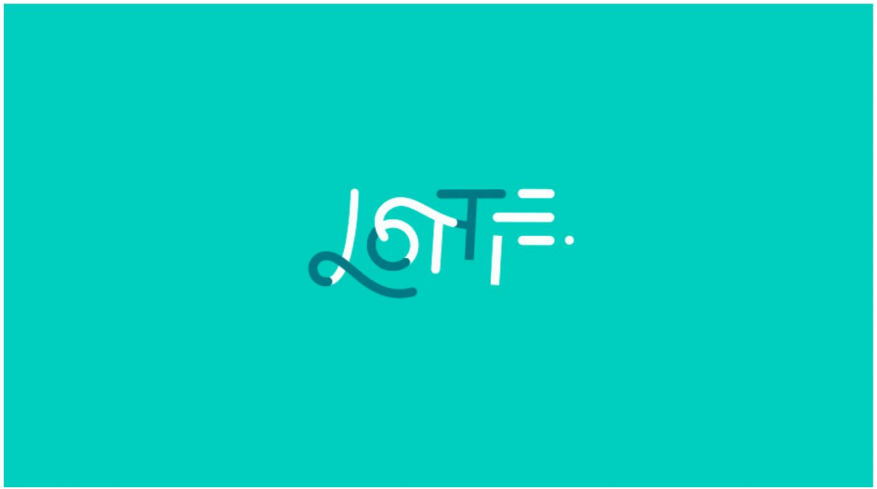
Cross Platform React Native UI Toolkit



Lottie for Android, iOS, React Native, and Web

Lottie is a mobile library for Android and iOS that parses Adobe After Effects animations exported as json with Bodymovin and renders them natively on mobile and on the web!

For the first time, designers can create **and ship** beautiful animations without an engineer painstakingly recreating it by hand. They say a picture is worth 1,000 words so here are 13,000:

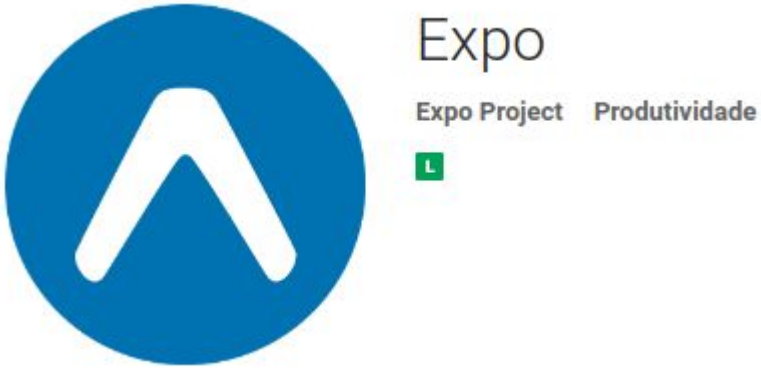


Expo

Expo é React-Native

- Apenas JavaScript
- Fácil prototipagem
- Rápido desenvolvimento

Expo para desenvolvedores e clientes



Prática

https://github.com/cades-ifrn/minicurso-react-native

Casa do Código

TADS

Passo 0 - Iniciar o projeto

- create-react-native-app wtads-2017
- cd wtads-2017
- yarn add react-navigation
- yarn start

Passo 1 - Criar tela de ínicio

- Utilizar componente Image
- Utilizar componente Button
- Estilizar componentes

Passo 2 - Adicionar React-Nativation

- Separar componente HomeScreen
- Utilizar componente StackNavigator

Passo 3 - Navegar entre telas

- Criar componente ProgramacaoScreen
- Adicionar ProgramacaoScreen no navegador raiz
- Adicionar evento para realizar navegação
- Corrigir barra de status do Android

Passo 4 - Apresentar programação

- Utilizar componente SectionList
  - Criar componentes Header e Item
- Carregar programação da internet

Passo 5.1 - Melhorar UX

- Utilizar componente TouchableOpacity
- Utilizar componente RefreshControl

Passo 5.2 - Melhorar UI

- Atualizar componente Header
- Atualizar componente Item

Passo 6 - Desafio

- Melhorar UX/UI
  - Sugestão: utilizar componente ActivityIndicator no ProgramacaoScreen
- Refatorar componentes
- Criar tela para listar minicursos
- Adicionar funcionalidade de compartilhar item da programação
  - Dica: utilizar Share API
- Adicionar persistência no primeiro acesso
  - Dica: utilizar AsyncStorage API

Próximos passos

Obrigado!

gh: @cades-ifrn | @felipemfp | @chicobentojr