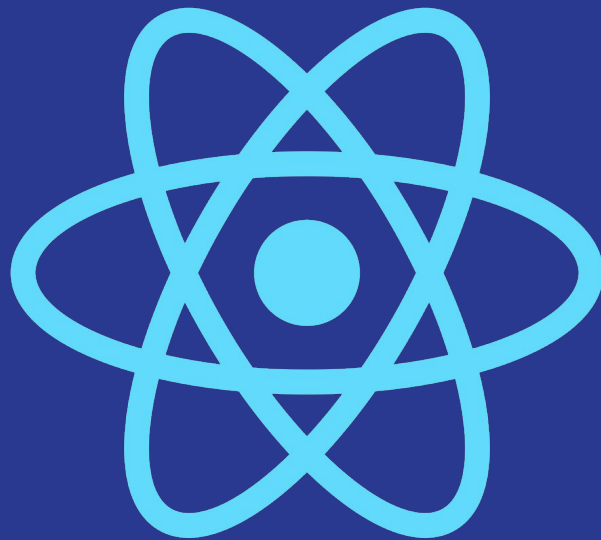


React Native

Munif Gebara Junior
munifgebara@gmail.com



React

Biblioteca Javascript para construção de interfaces (FACEBOOK, INSTAGRAM)

Declarativa

Baseada em Componentes

Eficiente, minimiza as alterações para manter o DOM atualizado.

Flexível, trabalha com frameworks conhecidos.

Utiliza JSX (XML-Like) para produzir HTML.



React

Angular Framework

React Biblioteca

Fácil de criar componentes reutilizáveis

Virtual DOM

Pode rodar do lado do servidor



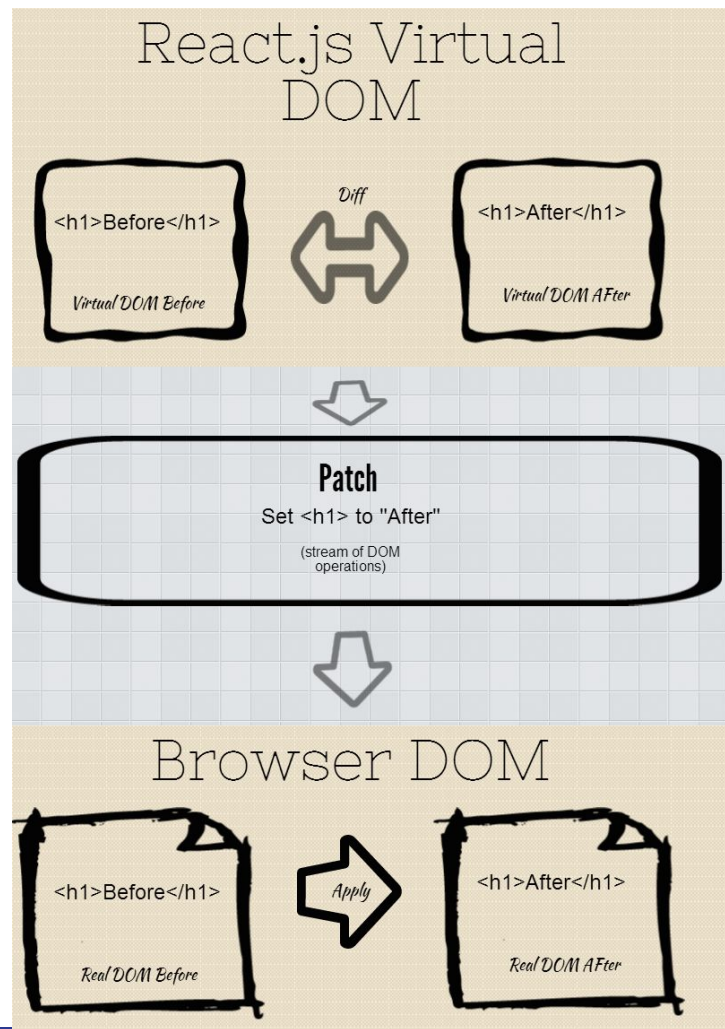
Virtual DOM

Calcula a diferença entre o DOM atual e o novo

Aplica apenas esta diferença

Ganho de performance

Alterar o DOM do navegador é lento



Primeiro Exemplo

```
class HelloMessage extends React.Component {  
  render() {  
    return (  
      <div>  
        Hello { this.props.name }  
      </div>  
    );  
  }  
}  
  
ReactDOM.render(  
  <HelloMessage name="Taylor" />,  
  mountNode  
);
```

Hello Taylor



Stateful

```
class Timer extends React.Component {  
  constructor(props) {  
    super(props);  
    this.state = { seconds: 0 };  
  }  
  tick() {  
    this.setState(prevState => ({  
      seconds: prevState.seconds + 1  
    }));  
  }  
  componentDidMount() {  
    this.interval = setInterval(() => this.tick(), 1000);  
  }  
}
```

Seconds:75

Quando o estado é alterado, o componente é atualizado executando o método render()

Render()

```
render() {  
  return (  
    <div className="MarkdownEditor">  
      <h3>Input</h3>  
      <textarea onChange={this.handleChange}  
defaultValue={this.state.value}/>  
      <h3>Output</h3>  
      <div  
        className="content"  
        dangerouslySetInnerHTML={this.state.html}/>  
    </div>  
  );  
}
```

método render() "desenha" o componente

JSX facilita a criação de estruturas:
Ex: `React.createElement('div')`.

Software

Xcode (Mac)

Android Studio

HomeBrew (Mac)

Node/NPM

watchman

react-native-cli (pode ser instalado na

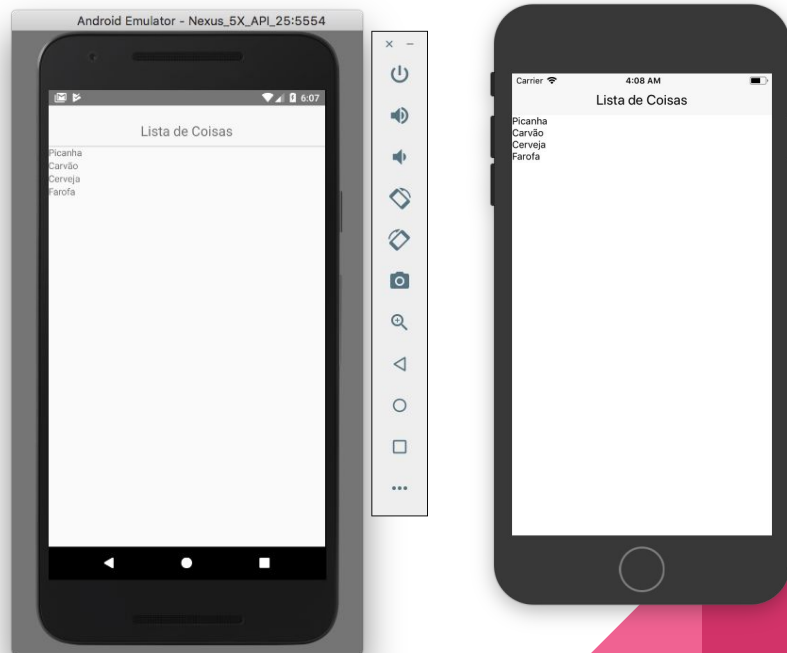
Editor de texto como o Atom, VisualStudio Code, etc...



Emulando

react-native run-android

react-native run-ios



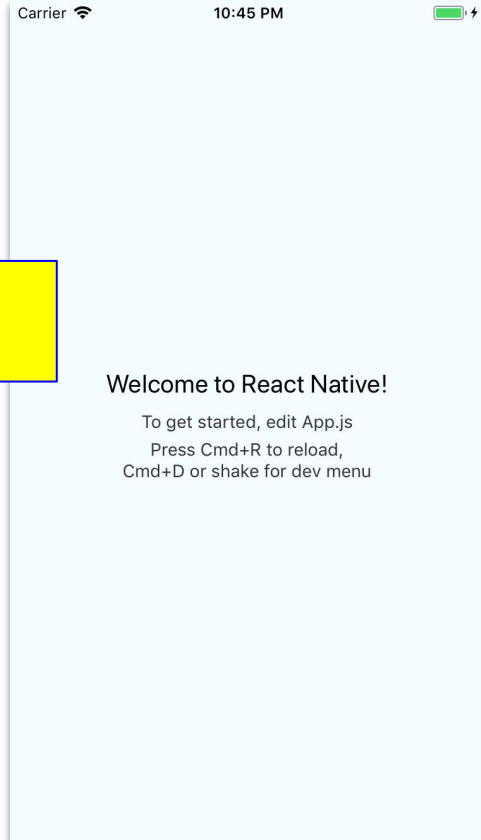
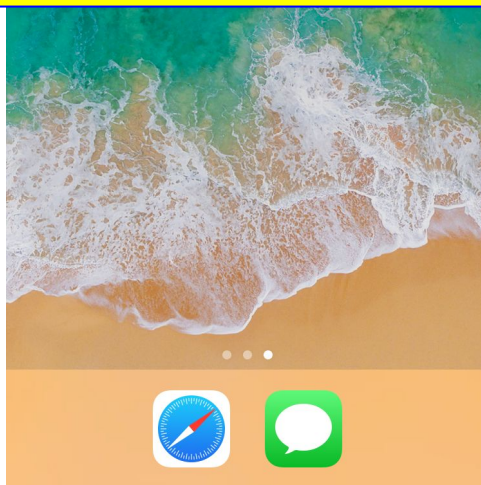
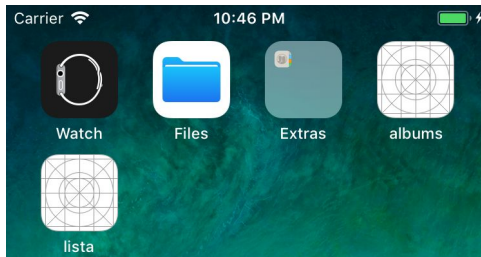
Primeiro Projeto

\$ react-native init lista

\$ cd lista

\$ react-native run-ios

Cria uma pasta e nela a estrutura de um projeto



Estrutura do Projeto

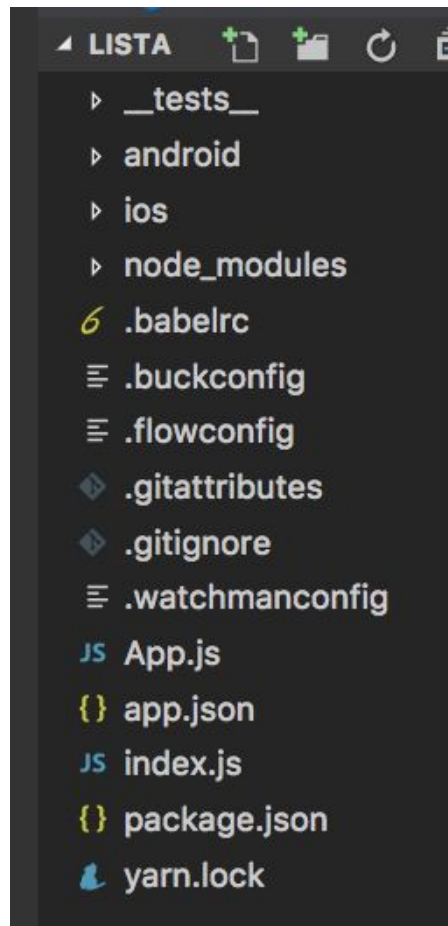
Adiciona por padrão IOS e Android

node_modules bibliotecas

diversos arquivos de configuração

package.json

.gitignore



index.js

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

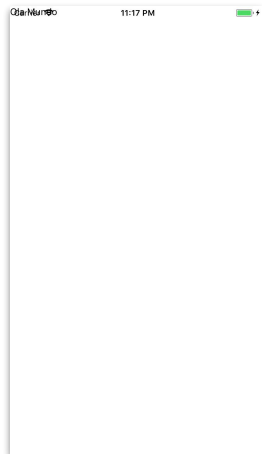
```
const App = () => {  
  return (  
    <Text>Ola Mundo</Text>  
  );  
};
```

Mesmo nome do App

```
AppRegistry.registerComponent('lista', () => App);
```

React
Comportamento
Combinar componentes

React Native
Renderiza componentes para
o dispositivo
Oferece vários componentes
prontos



titulo.js (componentizando)

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

const Titulo = () => {
  return (
    <Text>Lista</Text>
  );
};

export default Titulo;
```

index.js

```
import React from 'react';
import { Text, AppRegistry } from 'react-native';
import Titulo from './src/components/titulo';

const App = () => {
  return (
    <Titulo />
  );
};

AppRegistry.registerComponent('lista', () => App);
```

titulo.js (estilos)

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

const Titulo = () => {
  const { estiloText, estiloView } = estilos;
  return (
    <View style={estiloView}>
      <Text style={estiloText}>Lista</Text>
    </View>
  );
};
```

```
const estilos = {
  estiloView: {
    backgroundColor: '#F8F8F8',
    justifyContent: 'center',
    alignItems: 'center',
    height: 60,
    paddingTop: 15,
    shadowColor: '#000',
    shadowOffset: { width: 0, height: 2 },
    shadowOpacity: 0.2,
    elevation: 2,
    position: 'relative'
  },
  estiloText: {
    fontSize: 20
  }
};

export default Titulo;
```

titulo.js (propriedades)

```
import React from 'react';
import { Text, AppRegistry } from
'react-native';
import Titulo from '../src/components/titulo';

const App = () => {
  return (
    <Titulo texto={'Lista de Coisas'} />
  );
};

AppRegistry.registerComponent
('lista', () => App);
```

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

const Titulo = (props) => {
  const { estiloText, estiloView } =
estilos;

  return (
    <View style={estiloView}>
      <Text style={estiloText}>
{props.texto} </Text>
    </View>
  );
};

const estilos = {...};
export default Titulo;
```

Debug e console.log

```
titulo.js
....
const Titulo = (props) => {
  const { estiloText, estiloView } = estilos;
  console.log('Propriedades:', props);

  return (
    <View style={estiloView}>
      <Text
style={estiloText}>{props.texto}</Text>
    </View>
  );
};
...
```

command + D

React Native: Development (RCTCxxBridge
System JSC)

Reload

Debug JS Remotely

Enable Live Reload

Start Systrace

```
Running application lista ({
  initialProps = {
  };
  rootTag = 1;
})
```

```
Running application "lista" with appParams: {"root"
=== true, development-level warning are ON, perfor
```


```
Propriedades: ► {texto: "Lista de Coisas"}
```


lista-coisas.js (Componentes com Classes)

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

class ListaCoisas extends Component {
  componentWillMount() {
    console.log('ListaCoisas.componentWillMount()');
  }
  render() {
    return(
      <View>
        <Text>Lista de Coisas</Text>
      </View>
    );
  }
}

export default ListaCoisas;
```



lista-coisas.js (Classes, ciclo de vida)

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

```
class ListaCoisas extends Component {
  componentWillMount() {
    console.log('ListaCoisas.componentWillMount()');
  }
  render() {
    return(
      <View>
        <Text>Lista de Coisas</Text>
      </View>
    );
  }
}
```

```
export default ListaCoisas;
```

index.js

```
import React from 'react';
import { Text, AppRegistry, View } ...
import Titulo ...
import ListaCoisas ...
const App = () => {
  return (
    <View>
      <Titulo texto={'Lista de Coisas'} />
      <ListaCoisas />
    </View>
  );
};
```

lista-coisas.js (Estado)

```
class ListaCoisas extends Component {
```

<https://jsfiddle.net/Munif/ppta4xkk/1/>

```
  state = { lista: [] };
```

```
  componentWillMount() {  
    console.log('ListaCoisas.componentWillMount()');  
    this.carregaLista();  
  }
```

```
  carregaLista() {  
    setTimeout(()=>{  
      console.log('ListaCoisas.carregaLista()');  
      this.setState({lista: [{ produto: 'Arroz' },...]}  
    },1000);  
  }
```

```
}
```

lista-coisas.1.js (Lista)

```
class ListaCoisas extends Component {  
  
  renderCoisa() {  
    return this.state.lista.map(  
coisa => <Text key={coisa.produto}>{coisa.produto}</Text>);  
  }  
  
  render() {  
    console.log('ListaCoisas.render()');  
    console.log('ListaCoisas.state', this.state);  
    return (  
      <View>  
        {this.renderCoisa()}  
      </View>  
    );  
  }  
}
```

key para remover o warning!!

lista-coisas.2.js (HttpS)

HTTPS

```
class ListaCoisas extends Component {  
  carregaLista() {  
  
    fetch('https://raw.githubusercontent.com/munifgebara/reactnative/master/lista/lista.json')  
      .then(response => response.json().then(data=>{  
        this.setState({lista:data});  
      }))  
      .catch(error => {  
        this.setState({lista:[{produto:'Impossível carregar a  
lista'}]}));  
      });  
  
  }  
  
}
```

lista-coisas.2.js (Mostrando Imagem)

```
class ListaCoisas extends Component {
  renderCoisa() {
    const { imagemStyle, containerStyle } = estilos;
    return this.state.lista.map(coisa =>
      <View style={containerStyle} key={coisa.id}>
        <Text>{coisa.produto}</Text>
        <Image style={imagemStyle} source={{ uri: coisa.imagem }} />
      </View>
    );
  }
  render() {
    return (
      <ScrollView>
        {this.renderCoisa()}
      </ScrollView>
    );
  }
  ...
}
```

button.js (Botão)

children

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


const Button = ({ onPress, children }) => {
  const { buttonStyle, textStyle } = styles;
  return (
    <TouchableOpacity onPress={onPress} style={buttonStyle}>
      <Text style={textStyle}>
        {children}
      </Text>
    </TouchableOpacity>
  );
};

const styles = {...};

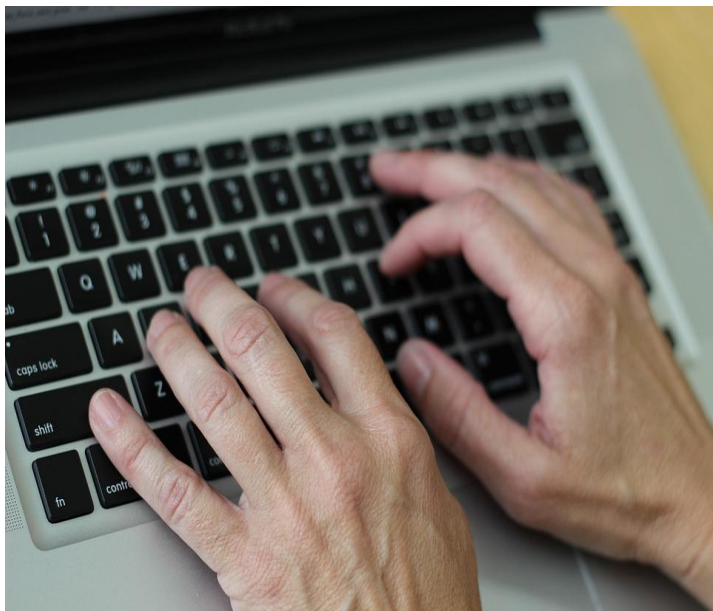
export default Button;
```

lista-coisas.2.js (Removendo)

```
class ListaCoisas extends Component {  
  
  comprar(coisa) {  
    let l = this.state.lista.slice(0);  
    let i=l.indexOf(coisa);  
    l.splice(i,1);  
    //alert(coisa.produto);  
    this.setState({lista:l});  
  }  
  renderCoisa() {  
    ...  
    <Button onPress={() => this.comprar(coisa)}>Remover</Button>  
  };  
}  
}
```



Mãos à Obra



O exemplo apresentado poderia ser mais "componentizado".

Altere o exemplo para utilizar mais componentes.

Observações:

Deve ser colocado em um repositório do GitHub;

Pode ser Feito em Duplas.