

# ReactNative

Programação para dispositivos móveis usando React

10 April 2021

Autor Ari Garcia

# Sobre o Autor

## Formação

- Engenheiro do Produção,
- Especialista em Engenharia de Software
- Mestre em Engenharia de Software e
- Doutorando em Engenharia de Software para Big Data

## IFPB

- Professor de Computação Distribuída e Segurança de Dados
- Pesquisador na área de Engenharia de Software e Aprendizado de Máquina

## Experiências

- Migração de sistemas legado para microserviços
- Restruturação de infraestrutura de servidores
- Desenvolvimento de aplicativos para smartphone
- Desenvolvimento de ferramentas para aplicações inteligentes

# Agenda

- A evolução da web como software (10min)
- Sobre React Framework (10min)
- Desenvolvimento baseado em Componentes (30min)
- Programação para React (~2h)
- React para Mobile (20min)
- Instalação do ReactNative (30min)
- Visão Geral sobre ReactNative (2h30min)
- Primeiro App (1h)

# Software para Web

# Web nos anos 90



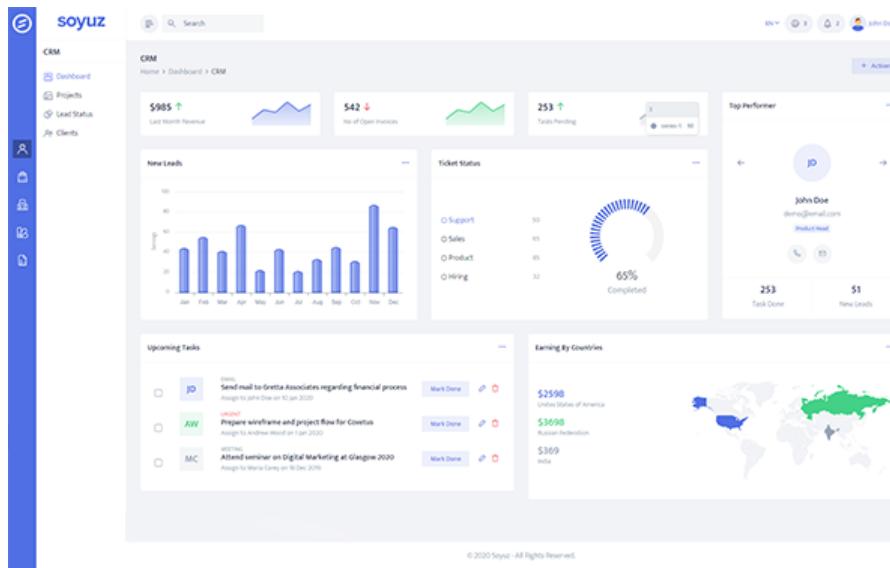
- mais dados e iteração baseada em navegação
- maior preocupação com carga de conteúdos
- estilização predominante sobre layout e textos

# Web em 2010:

The screenshot shows the homepage of TechTudo (www.techtudo.com.br) from 2010. The header features the 'techtudo' logo and the tagline 'a tecnologia descomplicada'. A search bar and a 'buscar' button are on the right. Below the header, a navigation menu includes 'REVIEWS', 'BLOGS', 'DICAS & TUTORIAIS', 'ARTIGOS', 'JOGOS', 'DOWNLOADS', and social media links for Twitter, Facebook, and RSS. The main content area has two sections: 'Destaques' (Features) and 'Blogs'. The 'Destaques' section includes articles about the Samsung Galaxy S2, LittleBigPlanet 2, and Solar Vox. The 'Blogs' section includes articles about Google's future and the iPod Nano. The overall design is clean and modern for its time.

- maior organização sobre os dados
- navegação mais intuitiva
- estilização sobre todos os elementos
- início do uso de aplicações web

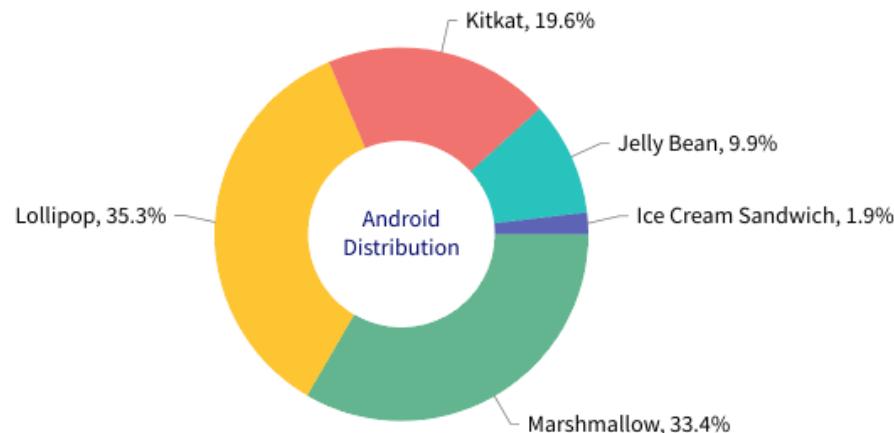
# Web em 2021:



- UX é tão importante quanto UI
- aplicação de princípios de design e de interação
- preocupação com performance de interfaces
- maior feedback para usuários
- focado em aplicativos

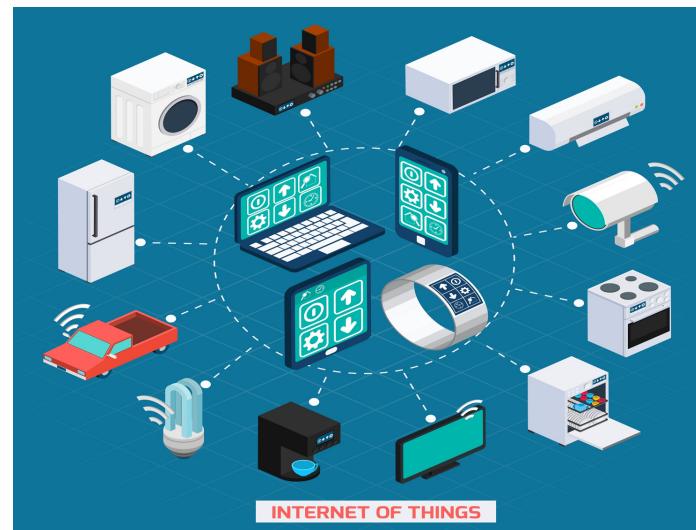
# Requisitos atuais de aplicativos web/mobile:

- efeitos visuais para promover microfeedback
- layout e comportamentos ajustáveis
- sincronização de dados ou invés de requisições constantes
- possibilidade de interações com microelementos



## Algumas perguntas:

- como desenvolver interfaces que possam apresentar e gerar dados, comportamentos, interações e feedbacks?
- o design para web não requer mais técnicas de web design?
- o desenvolvimento para web tornou-se um produto separado do servidor?
- com tantos requisitos não funcionais imperando nas aplicações web, qual a melhor maneira de desenvolver aplicações para este ambiente?



# Alguns frameworks web:

ANGULAR.JS	REACT.JS	VUE.JS
The Guardian 	AirBnB 	Alibaba 
Upwork 	Instagram 	Grammarly 
Pay Pal 	UberEats 	IPL dashboard 
Sony 	Dropbox 	Gitlab 

# React Framework

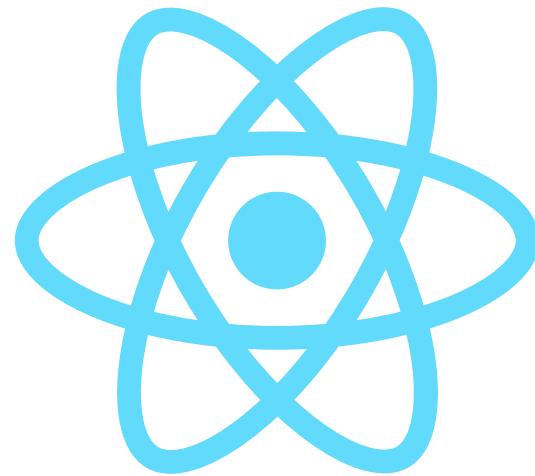
11

# React Framework #1/6

É um framework utilizado para criar componentes visuais de forma declarativa.

O react se preocupa em como fazer e o desenvolvedor diz o que deseja do componente.

"Torna fácil criar interfaces de usuário interativas"  
(fonte: <https://reactjs.org>)



## React Framework #2/6

Os componentes criados gerenciam seu próprio estado e podem ser usados para compor outros componentes mais complexos.

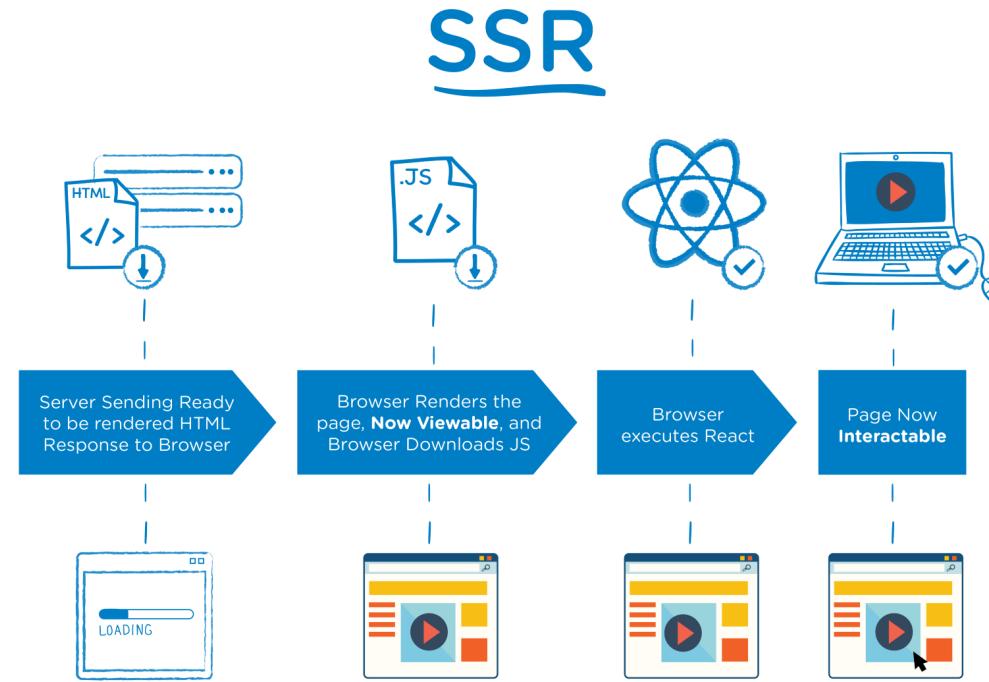
É possível utilizar o princípio de "dividir para conquistar" na construção de interfaces complexas.

A lógica dos componentes são escritas em JavaScript.

Dados podem ser passados através das propriedades dos componentes tornando possível manter o estado fora do DOM.

# React Framework #3/6

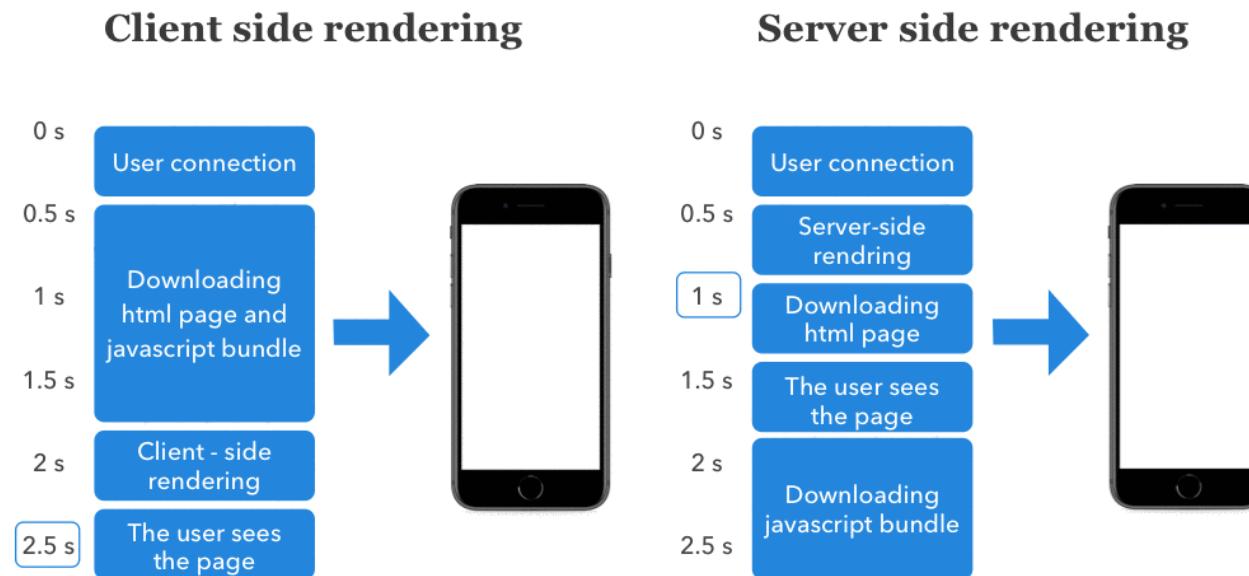
- React é agnóstico ao restante da stack de desenvolvimento, logo pode ser usado em conjunto com qualquer outra tecnologia.



- React pode ser renderizado tanto do lado do cliente (client-side) como do lado do servidor (server-side).

# React Framework #4/6

Client-side vs Server-side:



Para saber mais:

- <https://www.digitalocean.com/community/tutorials/react-server-side-rendering>
- <https://arkwright.github.io/scaling-react-server-side-rendering.html>

## React Framework #5/6

Um componente React é criado e renderizado originalmente sobre elementos DOM.

```
<!-- local onde será adicionado o componente raiz do React -->
<div id="root"></div>
<!-- código React para renderizar um componente -->
<script type="text/babel">

  ReactDOM.render(
    <h1>Hello, world!</h1>,
    document.getElementById('root')
  );

</script>
```

Resultado:

## React Framework #6/6

Para facilitar a produção de componentes visuais em React deve-se utilizar JSX.

JSX é uma sintaxe de programação para escrever tags HTML dentro de variáveis.

```
const element = <h1>Hello, world!</h1>;
```

Dessa forma é possível reescrever o código anterior da seguinte maneira:

```
//const element = React.createElement('h1', null, 'Hello, world without JSX!');  
const element = <h1>Hello, world with JSX!</h1>;  
ReactDOM.render(  
  element,  
  document.getElementById('root')  
);
```

Resultado:

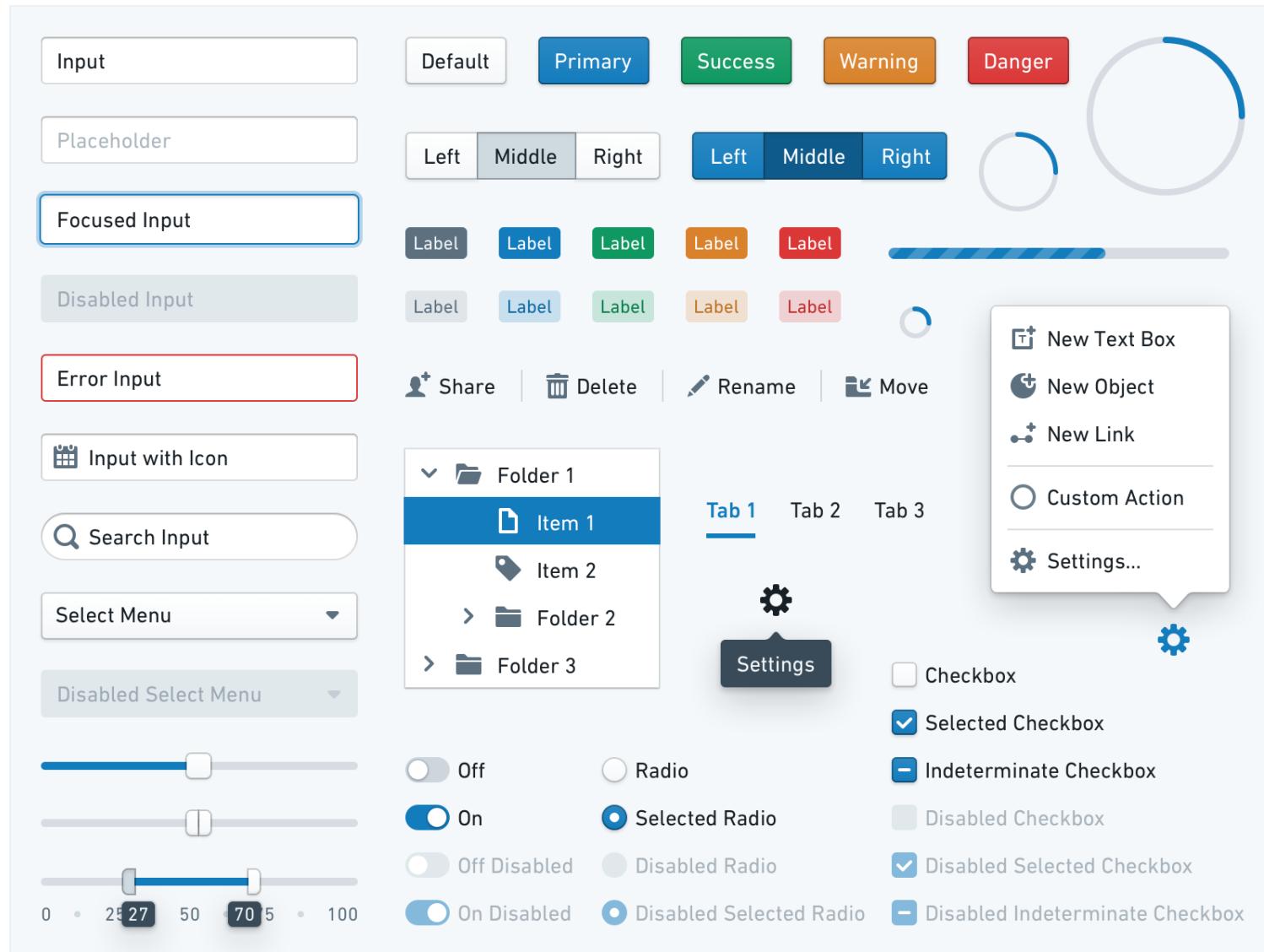
# Desenvolvimento baseado em Componentes

18

# Limitações e Premissas

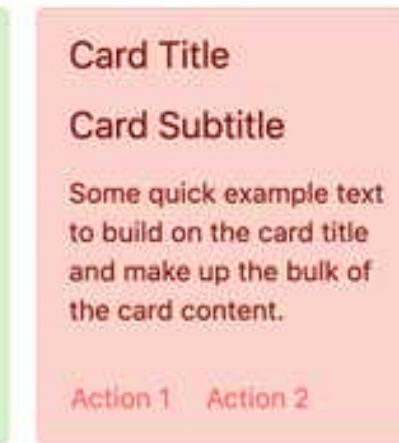
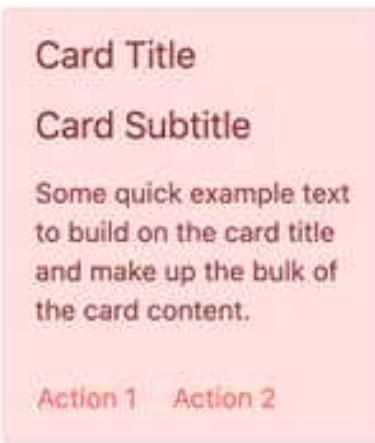
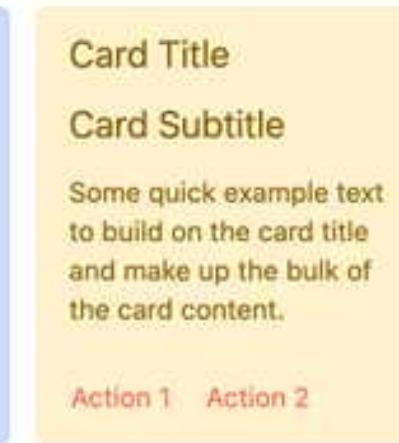
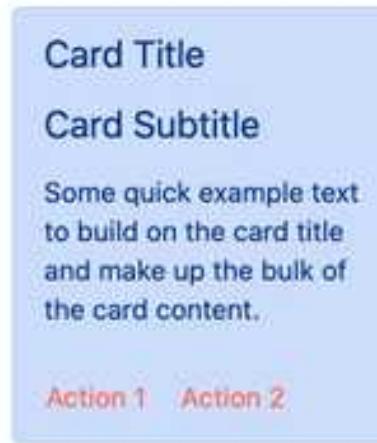
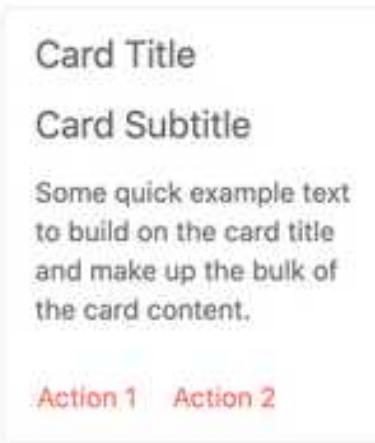
- Os conceitos apresentados aqui são direcionado para UI, mas não limitados a ele;
- Será considerado um componente qualquer elemento visual que possa ser reusado;
- Elementos visuais que possuam comunicação com o servidor serão chamados de componentes contextuais;
- Elementos visuais que limitados a construir uma organização da interface para acomodar outros componentes serão chamados de componentes de layout;
- Elementos não visuais responsáveis por apenas possibilitar organizar os demais em páginas serão chamados de componentes de roteamento;
- Não será usado outro framework neste estudo que não o React, apesar de possível.

# Componentes visuais #1/3



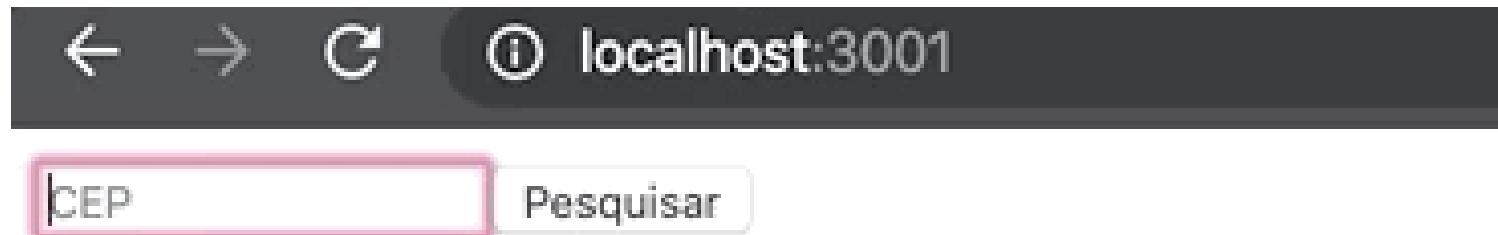
# Componentes visuais #2/3

Composição de componentes visuais:



## Componentes visuais #3/3

Componentes visuais que comunicam-se sozinhos com os servidores:



Fonte: <https://github.com/devarthurribeiro/react-via-cep>

22

# Criando UI a partir de componentes #1/3

Antes de continuar, uma pergunta:

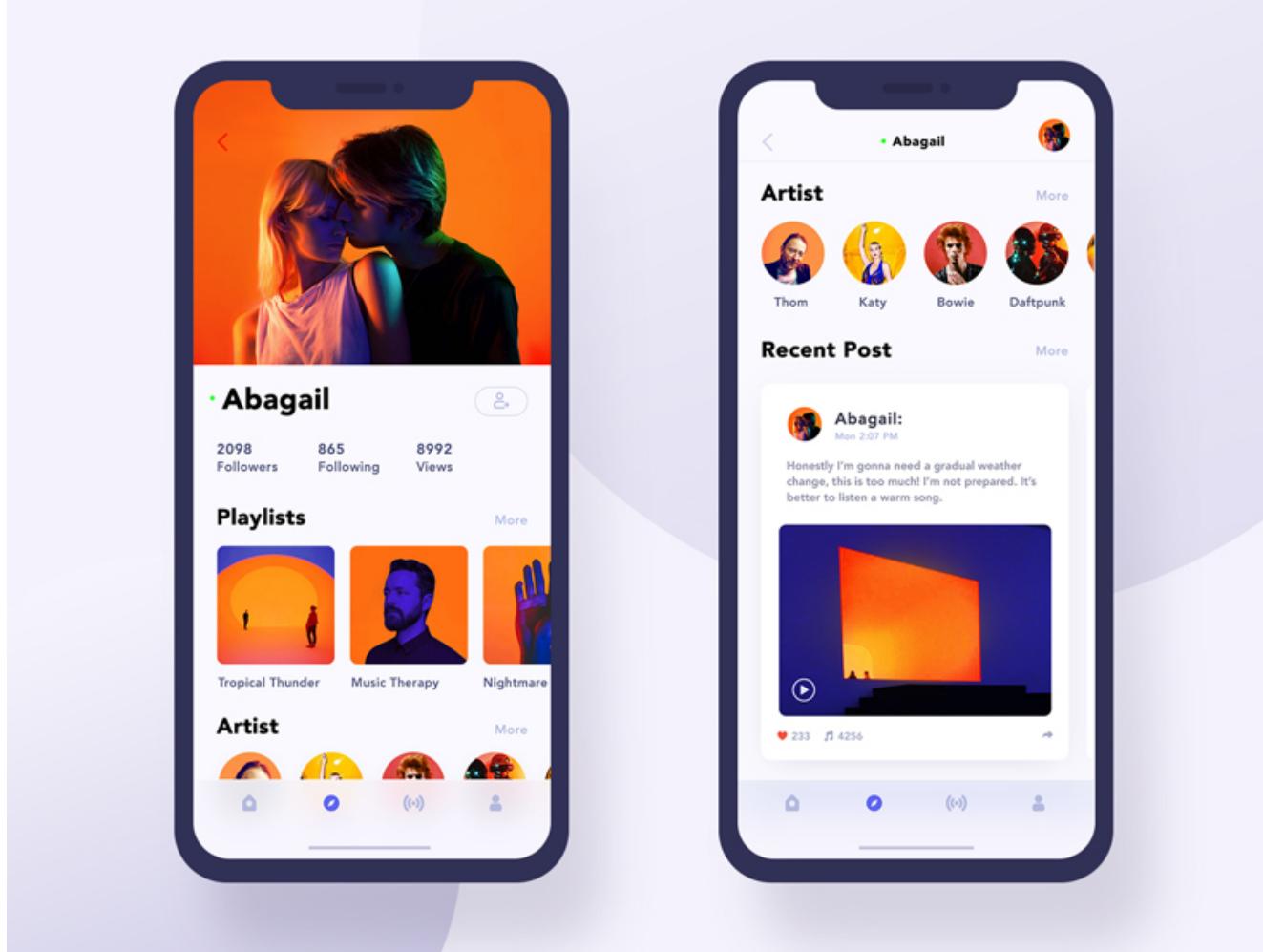
Como traduzir uma página web ou uma tela mobile em componentes visuais?



23

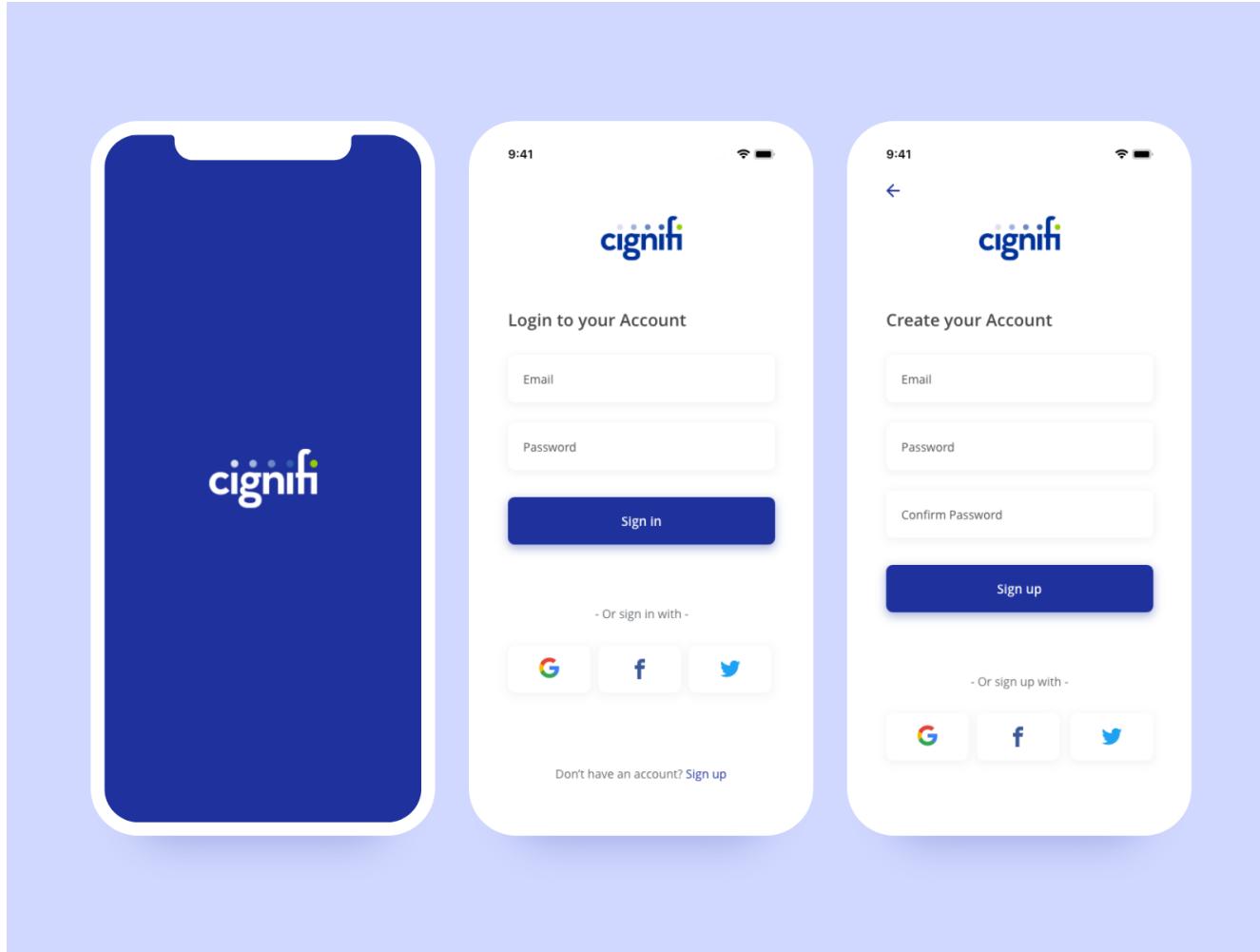
# Criando UI a partir de componentes #2/3

Pense um pouco e descreva como esta tela pode ser componentizada:



# Criando UI a partir de componentes #3/3

Pense um pouco e descreva como esta tela pode ser componentizada:



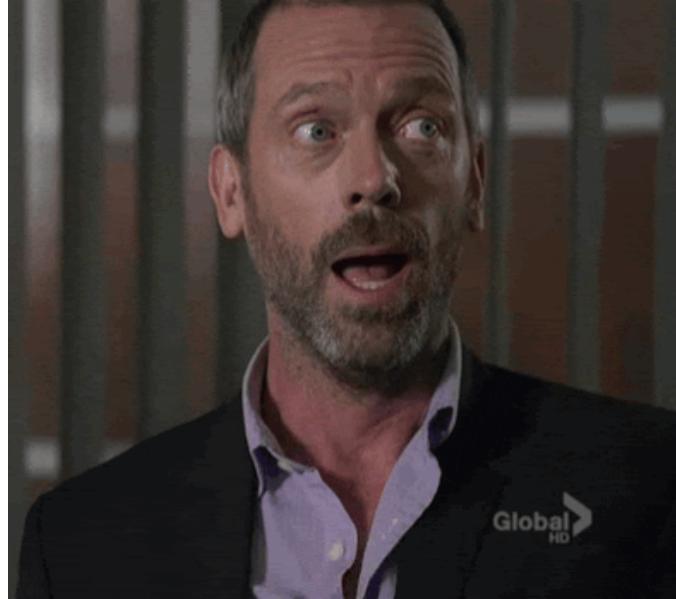
# Critérios para componentização #1/2

Quais dos critérios abaixo você usou para definir os componentes das telas anteriores?

- reuso no projeto
- reuso nos próximos projeto
- simplificar a construção
- dividir o projeto
- códigos já existentes (libs, por exemplo)

# Critérios para componentização #2/2

Na dúvida, não componentize!

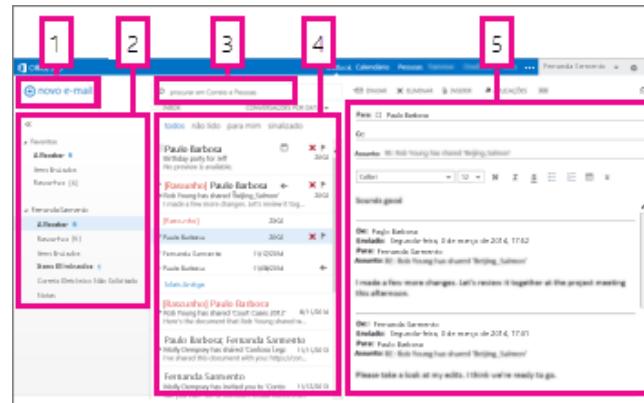


Por que?

27

# Web component #1/2

- os componentes da web são um conjunto de regras de tecnologia para a construção de um site que são empacotados (separados do resto do código do aplicativo) e reutilizáveis.
  - os componentes da web fornecem empacotamentos para reutilizar comportamentos, estilos, estados, propriedades e ciclo de vida.
  - alguns componentes da web geralmente exibem algumas APIs obrigatórias como componentes de vídeo da web podem exibir as funções *play()* e *pause()*.



Fonte: [https://www.theses.fi/bitstream/handle/10024/342721/Phan\\_Hong\\_Duc.pdf?sequence=2](https://www.theses.fi/bitstream/handle/10024/342721/Phan_Hong_Duc.pdf?sequence=2)

## Web component #2/2

- desenvolver um produto web baseado em componentização da UI requer um esforço inicial de planejamento e entendimento sobre que elementos deverão ser ou compor um componente;
- sem esse planejamento a vantagem de utilizar componentização pode ser perdida ou até impactar negativamente no projeto;
- se é uma simples página de site que não possua complexidade visual, não há justificativa para usar um framework como React;
- por um outro lado, o que se gasta em tempo com planejamento e melhor entendimento do projeto ganha-se velocidade de produção após todos os componentes terem sido desenvolvidos;

# Programação com React

30

# Preparação...

- accessem o site: <https://codesandbox.io>

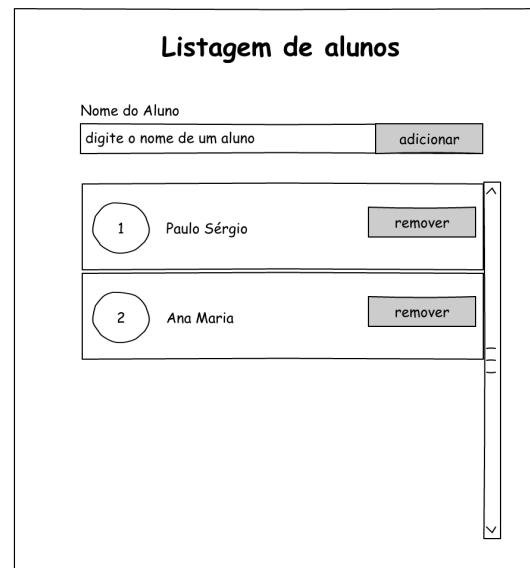
The screenshot shows the CodeSandbox web interface. On the left, there's a sidebar with navigation icons (File, Edit, Selection, View, Go, Help) and sections for Files (public, src, App.js, index.js, styles.css, package.json) and Dependencies (react 17.0.2, react-dom 17.0.2, react-scripts 4.0.0). Below these are sign-in options and a 'Sign In' button. The main workspace shows a file named 'App.js' with the following code:

```
1 import "./styles.css";
2
3 export default function App() {
4   return (
5     <div className="App">
6       <h1>Hello CodeSandbox</h1>
7       <h2>Start editing to see some magic happen!</h2>
8     </div>
9   );
10 }
11
```

To the right of the code editor is a preview window titled 'Hello CodeSandbox' containing the text 'Start editing to see some magic happen!'. At the bottom of the interface, there are tabs for Browser and Tests, and a URL bar showing 'https://9u7uf.csb.app/'. The footer includes links for Console, Problems, and React DevTools, along with status information: Ln 11, Col 1, Spaces: 2, UTF-8, LF, JavaScript.

# Primeiros passos...

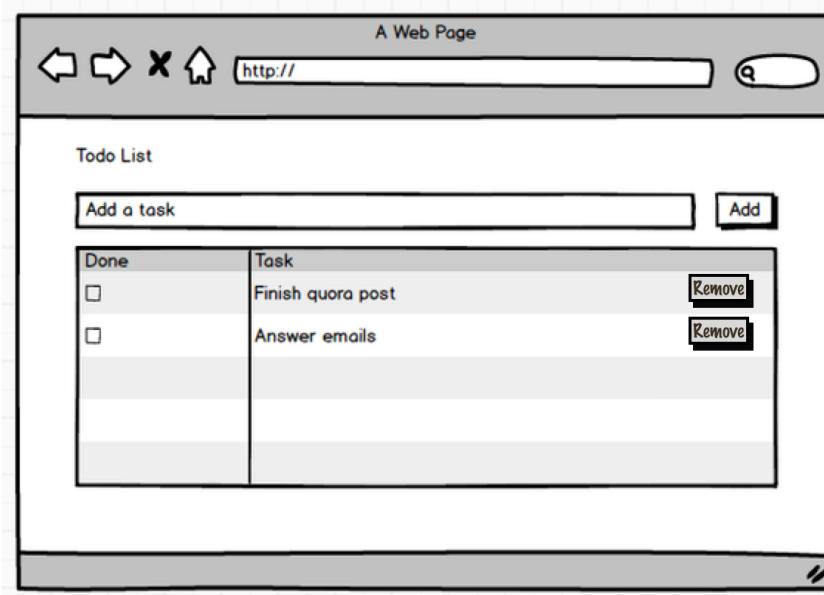
1. defina quais são os componentes desta tela
2. defina como será o layout desta tela
3. crie cada um dos componentes previamente
4. monte a tela
5. monte o comportamento da tela
6. estilize a tela (a seu gosto)



## Segundo aplicativo (tarefa de 30min):

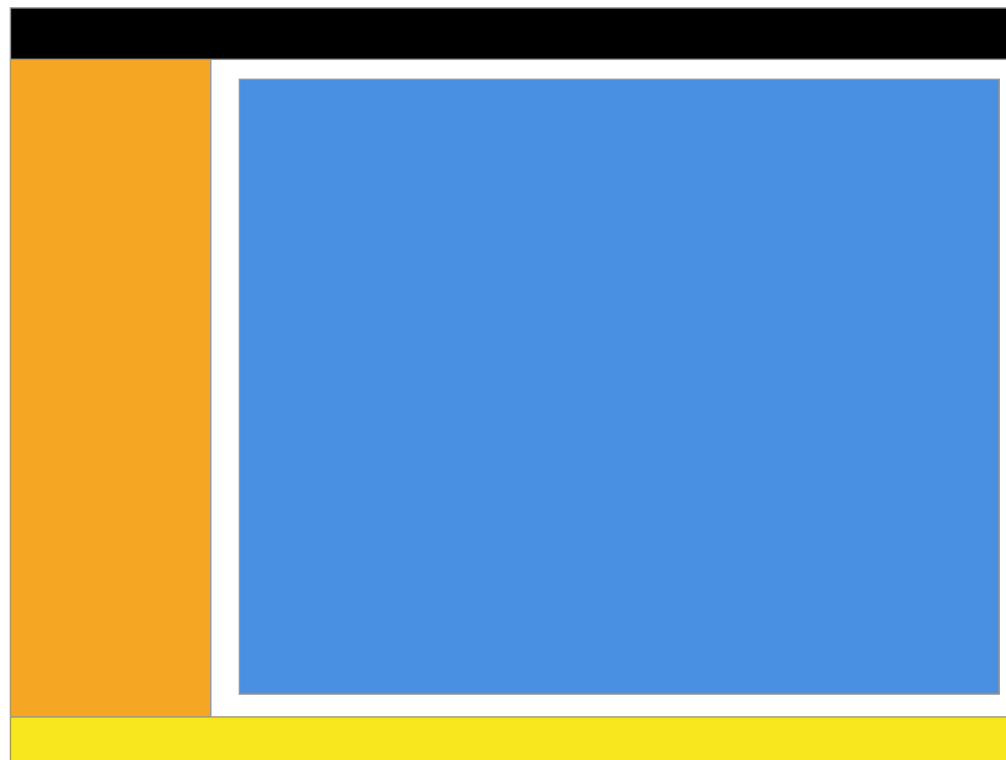
- siga os mesmos passos da atividades anterior
- siga os requisitos abaixo:

- 1) ao clicar no checkbox o o texto mudará de cor para azul e o botão "remove" some
- 2) ao clicar no botão remove o linha inteira some
- 3) ao cliente em "add" o texto digitado será adicionado na tabela
- 4) ao cliente em "add", caso não tenha texto no campo um alerta deverá ser exibido
- 5) estilize a seu gosto a página



## Trabalhando com layouts (tarefa de 30min):

- siga os mesmos passos da primeira atividade
- considerando o layout a seguir, crie os elementos que o compoem e adicione uma cor de background para cada elemento (side, top, footer e content)



34

# React para Mobile

35

# React para Mobile

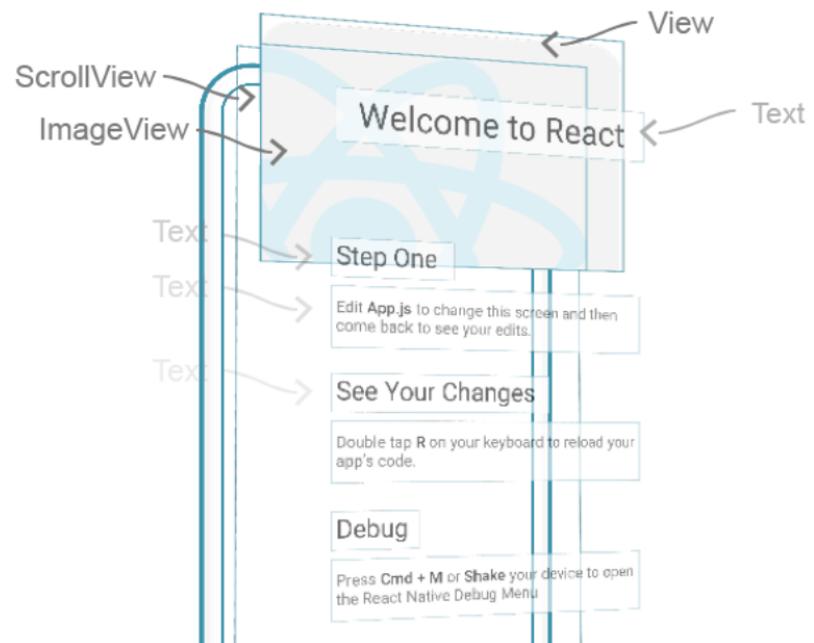
- Todo o conhecimento utilizados para produzir páginas web usando React podem ser utilizados para produção de aplicativos;
- Apesar do conhecimento ser reaproveitado, devido questões tecnológicas, o React possui uma versão própria para desenvolvimento de aplicativos para smartphone e iphone;
- O nome do framework do React para desenvolvimento de aplicações para mobile é React Native;
- Possui Tags (componentes) próprios;
- O termo nativo vem do fato que o React pode interagir nativamente com o dispositivo; 36

# ReactNative

- fornece um conjunto básico de componentes nativos independentes de plataforma, como:

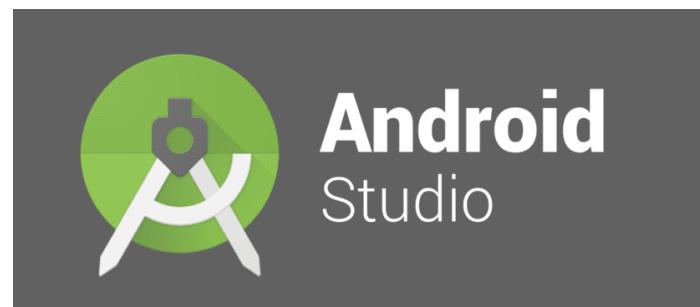
- View
- Text
- Image
- Button
- ScrollView

- Estes componentes são mapeados diretamente para os blocos de construção dos elementos UI nativos da plataforma (Android ou iOS)



# ReactNative

- para compilar e testar os códigos em uma plataforma específica é necessário ter o SDK da plataforma;
- para compilar para Android é necessário ter o Android Studio



- para compilar para iPhone é necessário ter o Xcode



# Instalação do ReactNative para Android

39

# Instalação do ReactNative para Android #1/2

Antes de começar:

- você precisa de nodejs e npm instalado:

<https://nodejs.org/en/>

- instalação do android-studio:

<https://developer.android.com/studio/>

- apontamento das variáveis de ambiente:

<https://www.youtube.com/watch?v=wnkynX7Yre0>

Outros links:

- AndroidStudio para W10: [https://www.youtube.com/watch?v=0zx\\_eFyHRU0](https://www.youtube.com/watch?v=0zx_eFyHRU0)
- Nodejs no Windows: <https://www.youtube.com/watch?v=-cLzUD0TQY0>

# Instalação do ReactNative para Android #2/2

Instruções para instalação do ReactNative:

```
https://reactnative.dev/docs/environment-setup
```

Teste de funcionamento:

```
npx react-native --version
```

Criando uma aplicação:

```
npx react-native init MeuApp
```

# Visão Geral sobre React Native

42

# Alguns elementos visuais #1/2

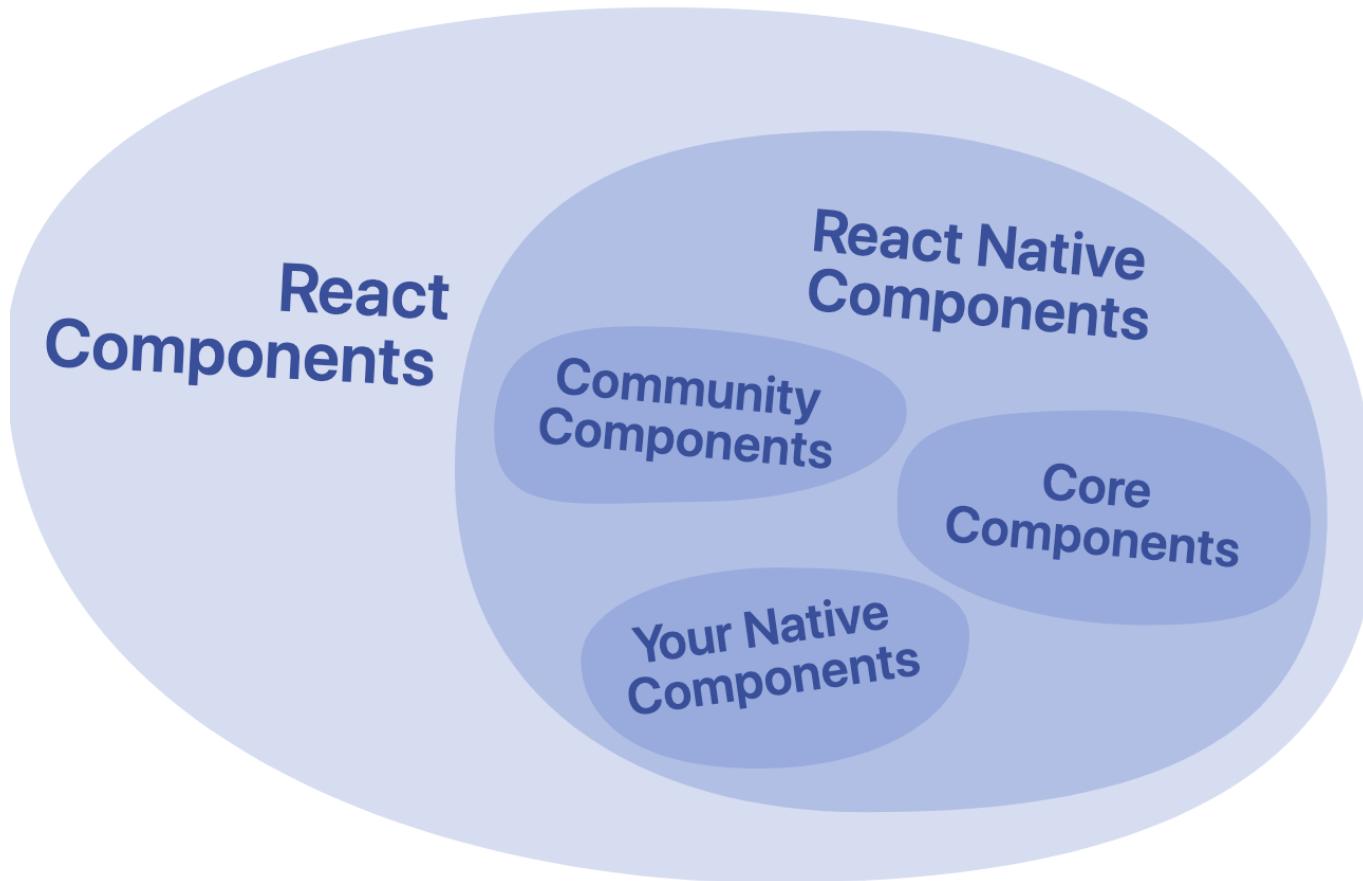
Um exemplo de mapeamento entre componentes React e Nativos:

REACT NATIVE UI COMPONENT	ANDROID VIEW	IOS VIEW	WEB ANALOG	DESCRIPTION
<code>&lt;View&gt;</code>	<code>&lt;ViewGroup&gt;</code>	<code>&lt;UIView&gt;</code>	<code>A non-scrolling &lt;div&gt;</code>	A container that supports layout with flexbox, style, some touch handling, and accessibility controls
<code>&lt;Text&gt;</code>	<code>&lt;TextView&gt;</code>	<code>&lt;UITextView&gt;</code>	<code>&lt;p&gt;</code>	Displays, styles, and nests strings of text and even handles touch events
<code>&lt;Image&gt;</code>	<code>&lt;ImageView&gt;</code>	<code>&lt;UIImageView&gt;</code>	<code>&lt;img&gt;</code>	Displays different types of images
<code>&lt;ScrollView&gt;</code>	<code>&lt;ScrollView&gt;</code>	<code>&lt;UIScrollView&gt;</code>	<code>&lt;div&gt;</code>	A generic scrolling container that can contain multiple components and views
<code>&lt;TextInput&gt;</code>	<code>&lt;EditText&gt;</code>	<code>&lt;UITextField&gt;</code>	<code>&lt;input type="text"&gt;</code>	Allows the user to enter text

## Alguns elementos visuais #2/2

"Como o React Native usa a mesma estrutura de API dos componentes React você precisa entender as APIs do componente React para começar"

(ReactNative, 2021) (<https://reactnative.dev/docs/intro-react-native-components>)



# Exemplo de código

aula-reactnative  
[Log in](#) to save your changes as you work ✓

Open files

- App.js

Project

- assets
- components

package.json

README.md

1 import \* as React from 'react';  
2 import { Text, View, StyleSheet } from  
'react-native';  
3 import Constants from 'expo-constants';  
4  
5 // You can import from local files  
6 import AssetExample from './components/  
AssetExample';  
7  
8 // or any pure javascript modules available in  
npm  
9 import { Card } from 'react-native-paper';  
10  
11 export default function App() {  
12 return (  
13 <View style={styles.container}>  
14 <Text style={styles.paragraph}>  
15 Change code in the editor and watch it  
change on your phone! Save to get a shareable  
url.  
16 </Text>  
17 <Card>  
18 <AssetExample />  
19 </Card>  
20 </View>  
21 );  
22 }  
23

Saved

My Device iOS Android Web



No errors

Prettier Editor Expo v40.0.0 Devices 0 Preview 45

## Estilização e Layout flex #1/3

- o ReactNative (RN) estiliza seus componentes utilizando **StyleSheet**
- como um componente cresce em complexidade, o StyleSheet pode ser utilizado para criar vários estilos
- Exemplo:

```
const styles = StyleSheet.create({  
  container: {  
    marginTop: 50,  
  },  
  bigBlue: {  
    color: 'blue',  
    fontWeight: 'bold',  
    fontSize: 30,  
  },  
  red: {  
    color: 'red',  
  },  
});
```

## Estilização e Layout flex #2/3

- apesar da similaridade, em alguns casos a estilização de componentes para mobile é diferente da web

([ReactNative Issues](https://github.com/facebook/react-native/issues/29308#issuecomment-792864162)) (<https://github.com/facebook/react-native/issues/29308#issuecomment-792864162>)

- a dimensionalidade dos componentes podem ser **fixas** ou **flexíveis** (flex)
- é recomendável deixar seu layout flexível e utilizar as regras de flexbox contidas no site <https://reactnative.dev/docs/flexbox>

([RN FlexBox](https://reactnative.dev/docs/flexbox)) (<https://reactnative.dev/docs/flexbox>)

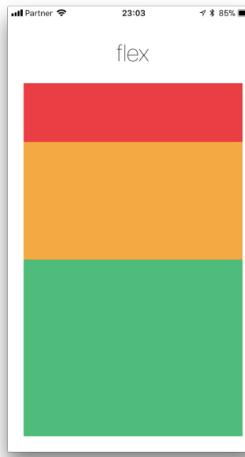
- o posicionamento dos componentes podem ser relativo (default) ou absoluto (uso apenas em casos absolutamente necessários)

([Mais sobre FlexBox](https://medium.com/wix-engineering/the-full-react-native-layout-cheat-sheet-a4147802405c)) (<https://medium.com/wix-engineering/the-full-react-native-layout-cheat-sheet-a4147802405c>)

47

## Estilização e Layout flex #3/3

- a separação de conteúdos pode ser pensada utilizando-se apenas flexbox



- o uso de imagens deve conter sempre a sua altura e largura fixa
- para usar imagens de background é recomendado o uso de **ImageBackground**

```
<ImageBackground source={...} style={{width: '100%', height: '100%'}}>
  <Text>Inside</Text>
</ImageBackground>
```

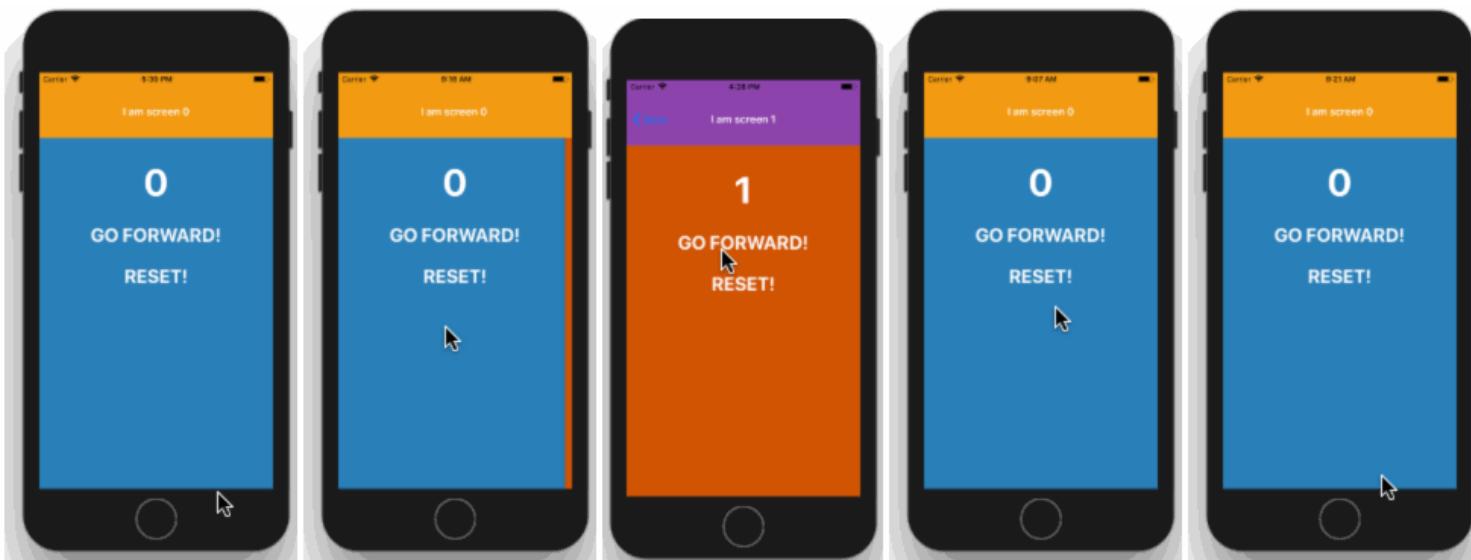
## Navegação entre Telas #1/2

- aplicativos móveis raramente são compostos por uma única tela, sendo necessário entender como navegar entre telas
- o gerenciamento da apresentação e da transição entre várias telas é normalmente feito por um **navegador**
- os padrões de navegação no mobile diferem da web e quase nunca serão realizados na mesma maneira
- algumas plataformas fornecem experiências próprias de navegação (iOS não possui um botão de retorno, por exemplo)
- um dos meios de se obter transparência sobre a navegação **react-native-navigation**

## Navegação entre Telas #2/2

Atualmente o **react-native-navigation** fornece 3 meios de navegação:

- tab: navegação baseada em abas
- stack: navegação baseada em empilhamento de telas
- drawer: navegação baseada em engavetamento de telas
- switch: *não mais suportado*



# Organização do Projeto

A organização dos códigos para o framework depende do próprio framework, no entanto a organização dos códigos de desenvolvimento (componentes, telas, recursos estáticos, etc) dependem do desenvolvedor. É sobre esta organização que estamos tratando neste ponto!

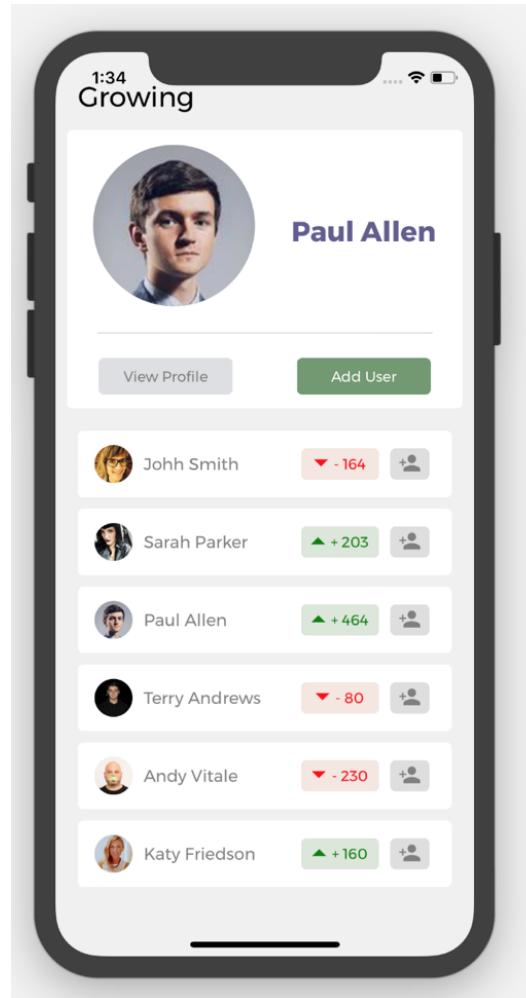
Existem guias de boas práticas exclusivos para React e outros para JavaScript.

O fato é que o uso de qualquer um desses guias ajuda a você ser desenvolvedor melhor!

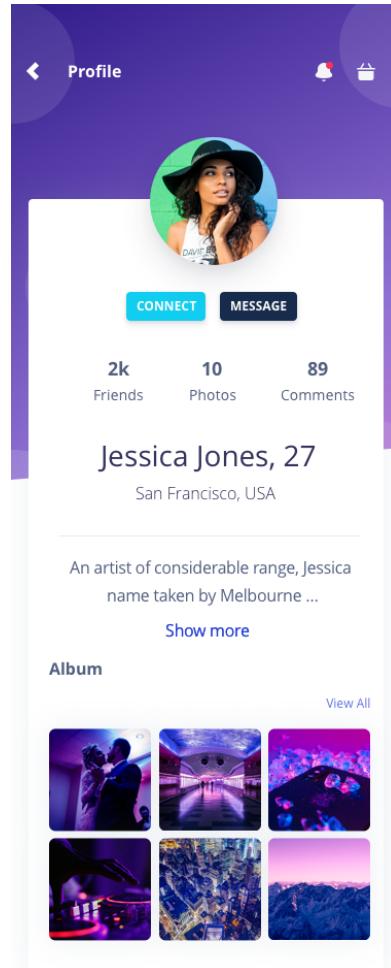
Alguns guias para dar uma olhada:

- <https://airbnb.io/javascript/react/>
- <https://airbnb.io/native-navigation/docs/guides/project-structure.html>
- <https://www.reactnative.guide/6-conventions-and-code-style/6.1-eslint.html>
- <https://www.reactnative.guide/5-project-structure-and-start-building-some-app/5.0-intro.html>

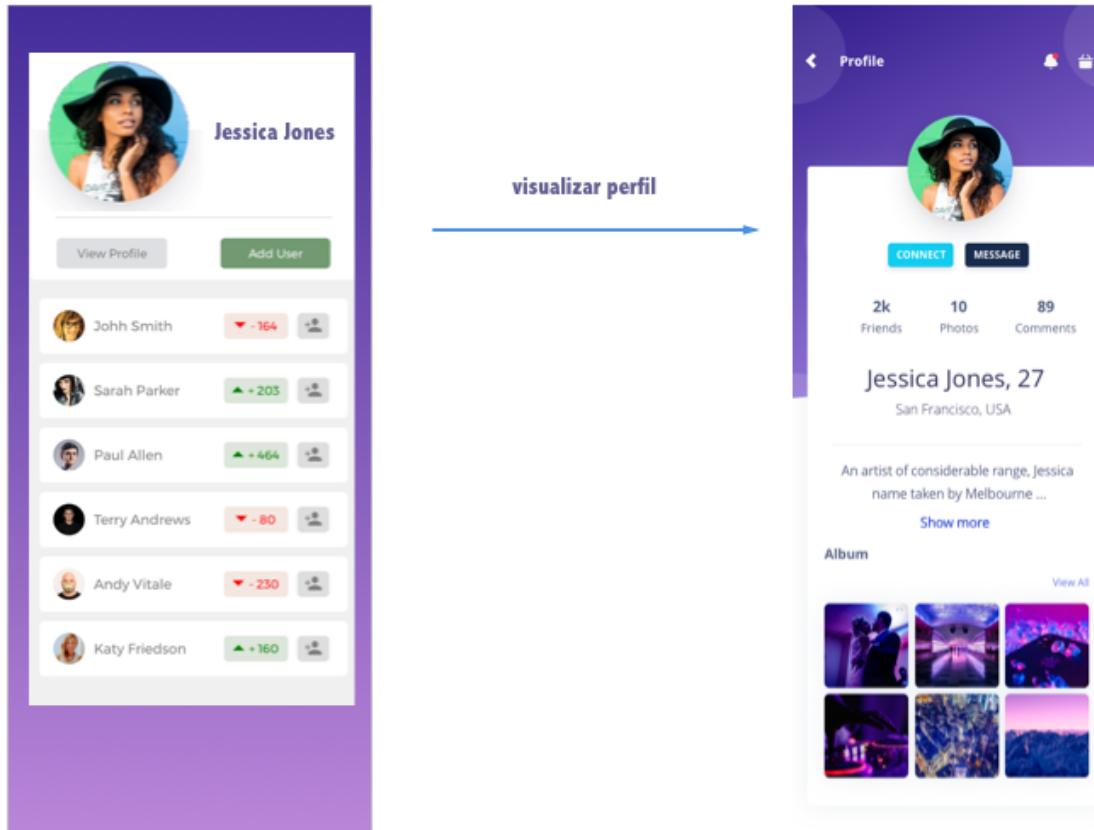
# Demo 1: Primeira tela



## Demo 2: Adicionando uma segunda tela



# Demo 3: Navegando entre telas



# Thank you

Autor Ari Garcia

[aristofanio@hotmail.com](mailto:aristofanio@hotmail.com) (<mailto:aristofanio@hotmail.com>)

<https://www.linkedin.com/in/aristofanio> (<https://www.linkedin.com/in/aristofanio>)

