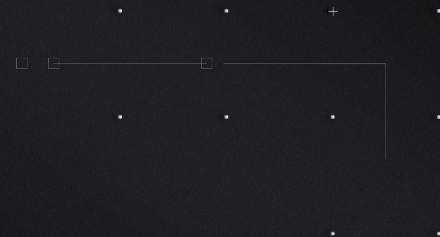


# FIAP

Hybrid Mobile App Development

Button, Text, TextInput

# Button





# Button

## Como funciona um botão no react-native?

É um componente básico do react-native. Ele possui um **callback** de quando é tocado, e um título.



PRESS ME

# Button

## Principais Props


- onPress (obrigatório)
- title (obrigatório)
- color
- disabled



# React

## Mas o que é mesmo uma prop?

As Props são parâmetros passados a um componente. Geralmente são valores que o componente precisa para se renderizar, ou são passados para componentes filhos.



```
<Button  
  onPress={() => navigation.navigate('Options')}  
  title="Options"  
  color="#00bfff"  
/>
```

# React

## Mas o que é mesmo um componente?

São os “blocos” fundamentais de um app React.  
Um componente React pode ou **não** ser traduzido em um componente visual.

Devem retornar um código **JSX** ou um componente filho no método especial **render**

Recebem informações através de Props, e podem manter informações através do State



```
class Classe extends React.PureComponent {  
  state = {  
    titleText: "Ninguém apertou o botão",  
  };  
  
  onPressedButton = (name) => {  
    this.setState({titleText: `${name} apertou o botão`});  
  };  
  
  render() {  
    return (  
      <View>  
        <Text>  
          {this.state.titleText}  
        </Text>  
        <Button  
          title="Stop capturing"  
          onPress={() => onPressedButton(this.props.name)}  
          color="#FF0000"  
        />  
      </View>  
    );  
  }  
}
```

# React

## E um functional component

É uma simplificação do componente. Usamos uma função para retornar um bloco JSX, ou componente filho

Os métodos de lifecycle são trocados por hooks

Trocamos o referencial interno **this** pelo escopo interno da própria função



```
const functionalComponent = (props) => {  
  const [titleText, setTitleText] = React.useState("Ninguém apertou o botão");  
  
  const onPressButton = (name) => {  
    setTitleText(`${name} apertou o botão`);  
  };  
  
  return (  
    <View>  
      <Text>  
        {titleText}  
      </Text>  
      <Button  
        title="Stop capturing"  
        onPress={() => onPressButton(props.name)}  
        color="#FF0000"  
      />  
    </View>  
  )  
}
```

# Arrow function

```
function funcaozinha(umaString) {  
  | return umaString + ' somei isso';  
}
```

```
const funcaozinha = (umaString) => {  
  | return umaString + ' somei isso';  
}
```

```
const funcaozinha = (umaString) => umaString + ' somei isso';
```



# Função anônima

É uma função que não é atribuída a uma variável. Ela é criada e utilizada na mesma linha, tal qual uma String ou int, e geralmente é utilizada em callbacks.

```
<Button  
  title="Stop capturing"  
  onPress={() => onPressButton(props.name)}  
  color="#FF0000"  
>
```

# Ternário

```
let greeting
if (isDay) {
  greeting = "Bom dia";
} else {
  greeting = "Boa noite"
}
```

```
const greeting = isDay ? "Bom dia" : "Boa noite";
```



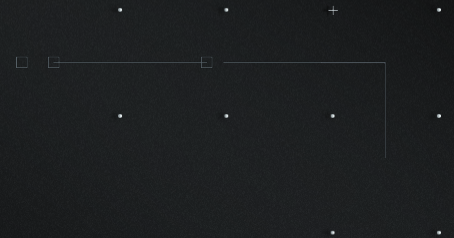
# Snack Expo

Outra possibilidade para desenvolvimento é a utilização do Snack Expo.

É uma IDE online com hot reload que permite desenvolvimento rápido e com feedback na hora

**`snack.expo.dev`**

# TextInput





# TextInput

É um componente básico do react-native, utilizado para o usuário submeter um texto.



# TextInput

## Principais Props

- value
- onChangeText (callback)
- inputMode
- maxLength
- multiline



# TextInput

## Controlando o Input

Para o nosso caso de uso, utilizaremos o TextInput como Controlled Component.

Isso significa que o estado dele será controlado pelo componente pai. Ou seja, iremos passar uma prop value, e atualizaremos ela com o estado do componente pai, utilizando o onChangeText

# Desafio

## Página de boas-vindas

Criar uma página inicial com pelo menos:

- 1 Text
- 1 TextInput
- 1 Button



# Desafio

## Página de boas-vindas

A página deverá ter um texto perguntando o nome do usuário. Ao inserir o nome e apertar o botão, o texto deve mudar, dando oi, e usando o nome do usuário

O botão deverá mudar para resetar, e ao apertado, o nome deverá ser apagado

# Desafio

Qual seu nome, amigo?

ENVIAR

Qual seu nome, amigo?

Joãozinho das Couves

ENVIAR

Oi, Joãozinho das Couves

RESETAR



# Desafio

## Dica:

- Utilizar 2 hooks useState. Um para o valor do input, e outro para salvar o nome do usuário que foi enviado. Ambos devem iniciar como string vazia

# Desafio

## Dica2:

- É possível escolher um componente dentre 2 a serem instanciados, utilizando lógica de JavaScript, como no exemplo abaixo

```
{  
    isNight ?  
    <Text> Boa Noite </Text> :  
    <Text> Bom dia </Text>  
}
```





# Desafio

## Dica3:

- As funções de callback não recebem parâmetro, mas é possível usar uma função anônima para passá-los

```
<Button  
  title="Stop capturing"  
  onPress={() => onPressButton(props.name)}  
  color="#FF0000"  
>
```

# Desafio

## Parte 2:

- Criar um novo botão, que agora mude o Style do app. De Dark para Light  
Para isso, é preciso declarar 2 styles, e mudar o backgroundColor da View e a color do Text





# Links

<https://reactnative.dev/docs/textinput>

<https://reactnative.dev/docs/button>

<https://snack.expo.dev/>

[https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions/Arrow\\_functions](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions/Arrow_functions)

<https://www.javascripttutorial.net/javascript-anonymous-functions/>

Dúvidas, anseios, desabafos?



FIAP